CAPITULO I

FOTO 1

FUENTE: EL AUTOR

FUNDAMENTOS GENERALES

Se describe los siguientes.

1.1 ANALISIS HISTORICOS.- La historia ha permitido al ser humano tener pleno conocimiento de cómo ha ido evolucionando la ciencia de la computación, desde sus inicios, a continuación se realizará un breve recuento histórico del desarrollo de los computadores y las tarjetas de adquisición de datos. El ser humano con la innata curiosidad que le caracteriza, siempre estuvo en la búsqueda de nuevos instrumentos y procedimientos de medida y cálculo partiendo de lo mas elemental a lo más complicado.

En 1924, **Bull** en Francia patenta algunos dispositivos electromagnéticos para análisis numéricos, crea una sociedad que luego seria la firma **Bull** en Europa.

Con el descubrimiento de los semiconductores o transistores y coincidiendo con el desarrollo de la **Segunda Guerra Mundial** que causó una gran presión social, obligo a los científicos a mejorar la tecnología existente para adaptarla a procesos y trabajos bélicos, paralelamente se pusieron a punto los semiconductores o transistores cuyas propiedades conocían desde mucho antes.

En 1943 en la escuela de Ingeniería Electrónica de **Moore,** bajo la dirección de **John Mauchly** y **Presper Echert,** se construyó la **UNIVAC** esta computadora digital utilizaba diodos de cristal **.**

La IBM (Thomas Watson) en 1970 emplean los circuitos integrados y esta firma lanzó los primeros computadores personales (PC), así comenzó la revolución de este campo debido a los adelantos casi diarios que se van suscitando en la tecnología de la fabricación e integración.

En casa y en la oficina, el ordenador personal continúa con su progreso. El PC se ha establecido en un gran número de campos. Los componentes hardware y software están siendo cada vez más potentes y más rentables. Es lógico, por tanto, que la industria quiera tomar provecho de este hecho, para reducir costes y/o incrementar la productividad.

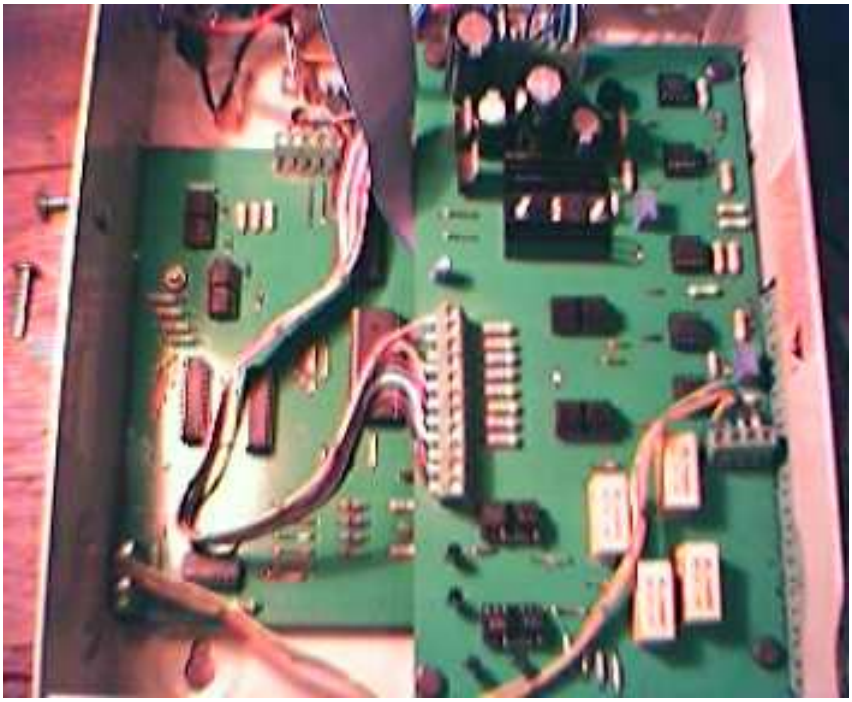
Ciertas tareas industriales están actualmente en manos de los ordenadores desde hace tiempo: desde emplear la tecnología Windows cuando se manejan pedidos

y/o se ajustan parámetros de maquinaria hasta preparar o visualizar datos prácticamente de cualquier tipo.

No hay que sorprenderse entonces, que los especialistas en automatización y los usuarios estén pensando ahora en qué forma se pueden transferir al PC otras tareas, para poder llegar a un mayor ahorro. Más recientemente un gran número de simuladores de PLC (controladores lógicos programables) por software ha aparecido en el mercado, que están ayudando a transferir el control de tareas al disco duro y presentan una automatización más efectiva en costes en una simple pieza de hardware (el PC).

TARJETAS DE EXPANSIÓN. Como el sistema operativo sólo puede proporcionar respuestas suaves en tiempo real lo más simple es emplear extensiones hardware para las tareas críticas (placas de expansión PC) y soluciones software para el resto de tareas. Esto nos lleva a una compatibilidad con futuros sistemas operativos y una solución totalmente factible actualmente. Estas tarjetas de expansión asumen las tareas críticas en tiempo real que el ordenador (PC) no puede atender, se está hablando de tarjetas que incorporan DSPs (Procesadores de Señales Digitales) o microcontroladores y que aportan una ayuda a la anterior “sobrecarga” mencionada para los ordenadores (PC).

FOTO 2



FUENTE: EL AUTOR

LA ESTRUCTURA ABIERTA. Aún no se ha establecido un estándar para poseer extensiones compatibles en tiempo real de sistemas operativos. De una forma estrictamente determinante, los sistemas estándar actuales deben ser modificados de forma general, así que la principal ventaja de un sistema basado en PC - su estructura abierta – pude llegar a ser un inconveniente. No obstante, la estructura abierta, permite a la empresa o el desarrollador más libertad en la elección de la herramienta adecuada para el análisis, diseño y programación del sistema SCADA. La solución comienza a ser propietaria nuevamente (cada empresa ofrece su solución) y la conversión a futuras generaciones de sistemas operativos lo hace más difícil.

FOTO 3

FUENTE: EL AUTOR

1.2 OBJETIVOS E HIPÓTESIS.-

GENERAL.- Dar un enfoque a la comunicación entre el computador y el ambiente externo a través de señales eléctricas que se transmiten del exterior.

ESPECÍFICOS:

- Dar una introducción a las comunicaciones especialmente de tipo paralelo entre la PC y dispositivos periféricos de entrada y salida, así como la descripción de sistemas de Adquisición de datos y Supervisión de control digital, especificando los tipos de interfaces de comunicación.

- Alcanzar un grado de desarrollo tecnológico que se ajuste a las nuevas exigencias de la industria de la automatización y del control estadísticos de procesos.

- **HIPÓTESIS.- .-** El diseño de un HARDWARE E IMPLEMENTARLO EN UN SOFTWARE permitirá la comunicación entre el computador y el medio externo a través de sensores de temperatura.

FOTO 4

FUENTE: EL AUTOR

FOTO 5

FUENTE: EL AUTOR

1.4 METODOLOGIA.- Para ello tenemos.

MÉTODOS Y TÉCNICAS.

Para la investigación se utilizará las siguientes técnicas.

A) MÉTODO INDUCTIVO.- Pues parte de lo particular a lo general al señalar que el sensor de temperatura podrá adaptarse a diversos ambientes pueden ser estos un edificio en construcción en donde activara una alarma en caso de incendio o en la Industria al vigilar la temperatura de calderos y así indicar si esta o no a nivel de la temperatura permitida.

B) OBSERVACIÓN DE CAMPO.- Al ver los requerimientos que podrían contribuir al bienestar social lo mejor es prevenir que lamentar . Muchas personas o empresas

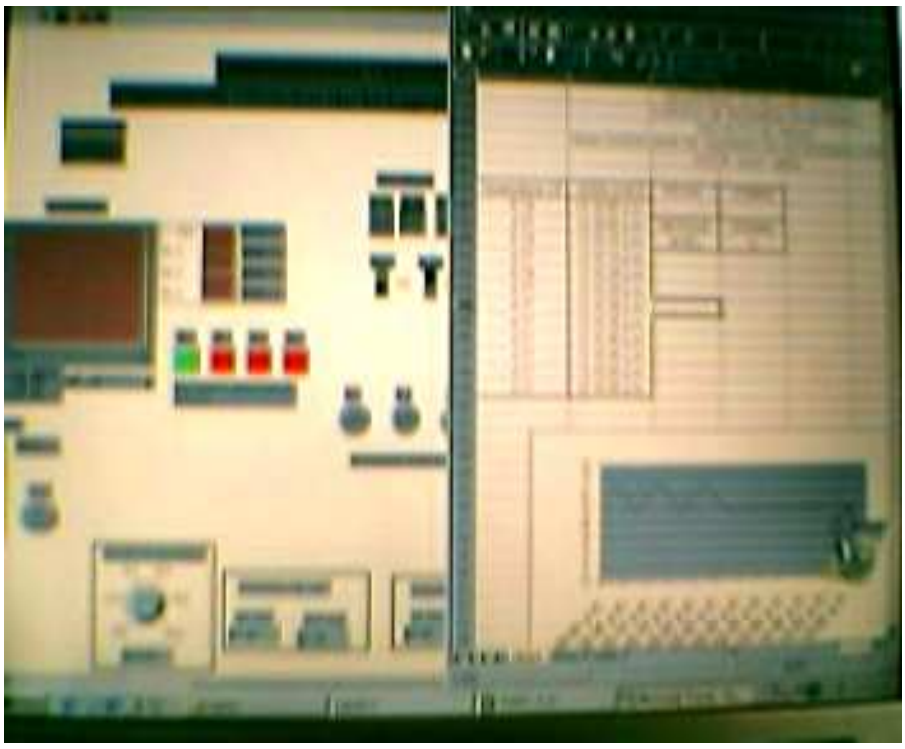
pueden perder su vida o posesiones debido a que no han sido advertidas convenientemente de peligro por este motivo se ha observado conveniente crear una tarjeta DAQ con censor de temperatura para que advierta de este peligro.

1.4.- SISTEMAS SCADA Y HMI[1] La respuesta nos aclara el concepto.

QUÉ ES EL SOFTWARE SCADA?

El software SCADA es aquél que permite ver en una pantalla el esquema de una instalación controlada por autómatas y sobre ésta se reflejan los valores clave y se pueden variar las consignas.

FOTO 6



FUENTE: EL AUTOR

[1] INSTRUPEDIA 99 , NATIONAL INSTRUMENTS , www.natinst.com .

Imaginemos un depósito de agua en el que hemos incorporado un sensor de temperatura y una resistencia.

Nuestro sistema intenta mantener el agua a la temperatura de consigna de 43º que hemos definido. En la pantalla veremos la temperatura real medida por el sensor y podremos programar una nueva consigna.

Si entra agua fría en el depósito, el sistema activará la resistencia para alcanzar de nuevo la temperatura deseada. Mientras tanto, por pantalla monitorizaremos todo el proceso.

FOTO 7

FUENTE: EL AUTOR

El sistema **SCADA** (*Supervisión Control y Adquisición de Datos*) es un sistema utilizado actualmente en el control de procesos, que puede ser desarrollado bajo

cualesquier **plataforma** que permita al usuario visualizar de manera fácil el proceso monitoreado, las ventajas que se obtienen al utilizar sistemas SCADA son:

- Interface gráfica para el usuario (GUI).
- Control de la tendencia histórica (Trending).
- Fijación de parámetros de control (PID) y alarmas.
- Intercambio dinámico de datos (DDE).
- Flexibilidad para manejo de protocolos como Modbus, Serial, etc.
- Procesamiento de datos OLE para control de procesos.

En necesario aclarar que el sistema SCADA involucra tanto software como hardware es decir se trata de un lazo completo de control que puede valerse de técnicas digitales como analógicas, no así los sistemas HMI que son netamente el lenguaje de programación gráfico.

La figura muestra un sistema SCADA, en el cual se pueden observar el empleo de diferentes plataformas para la adquisición de datos como Bus GPIB(*Bus de interface de propósito General*), plataforma SCXI (*Instrumentos extendidos y señales acondicionadas*), protocolos de puertos seriales (*Mod Bus* protocolo Modicom bus).

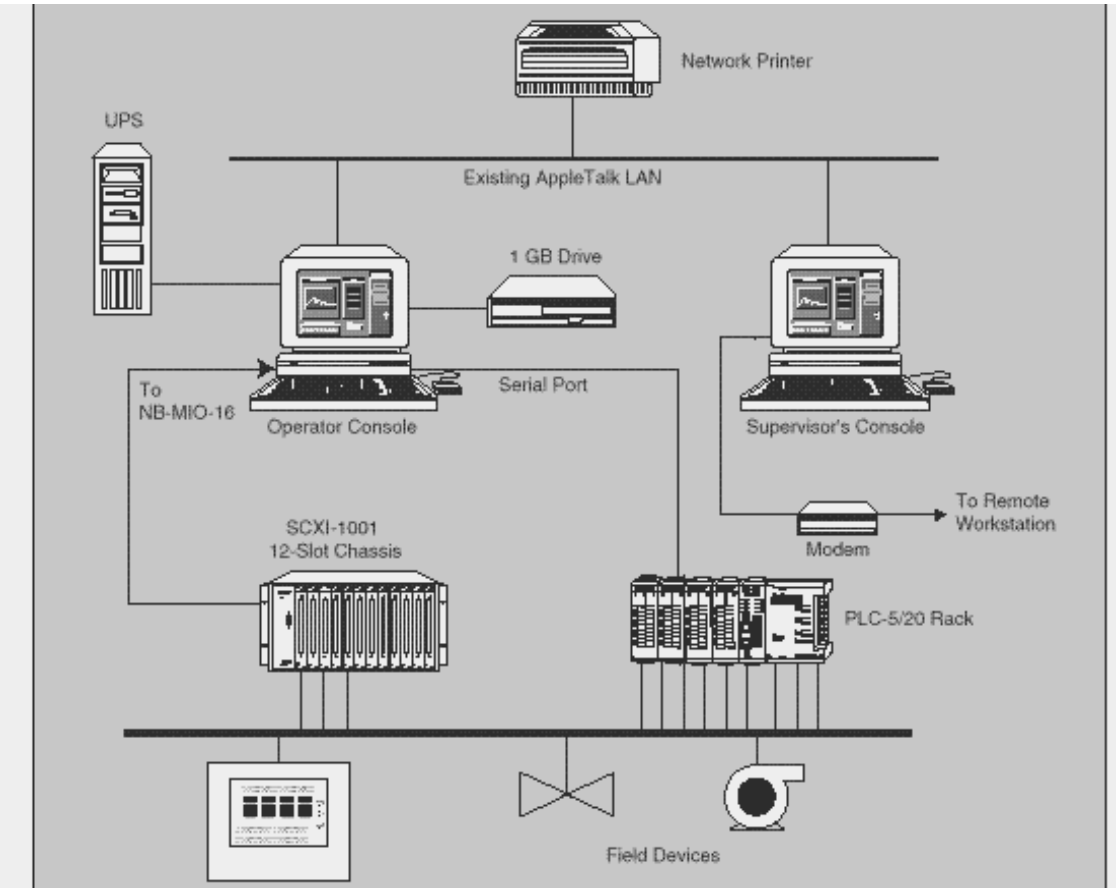


Figura 1.1 - Esquema típico de un sistema SCADA

Los sistemas **HMI (***Human-Machine Interface*) son plataformas gráficas de programación sobre las cuales se desarrollan los programas de aplicación (software), estos sistemas ofrecen al usuario monitorear en tiempo real un proceso industrial y además realizar un control estadístico del mismo (SPC *statistical process control*).

La figura muestra una plataforma de *interface gráfica para el usuario (GUI)* desarrollada bajo el software **Lookout** de **National Instruments** en la que se pueden observar las *tendencias históricas (Trending)* así como control estadístico del proceso (SPC).

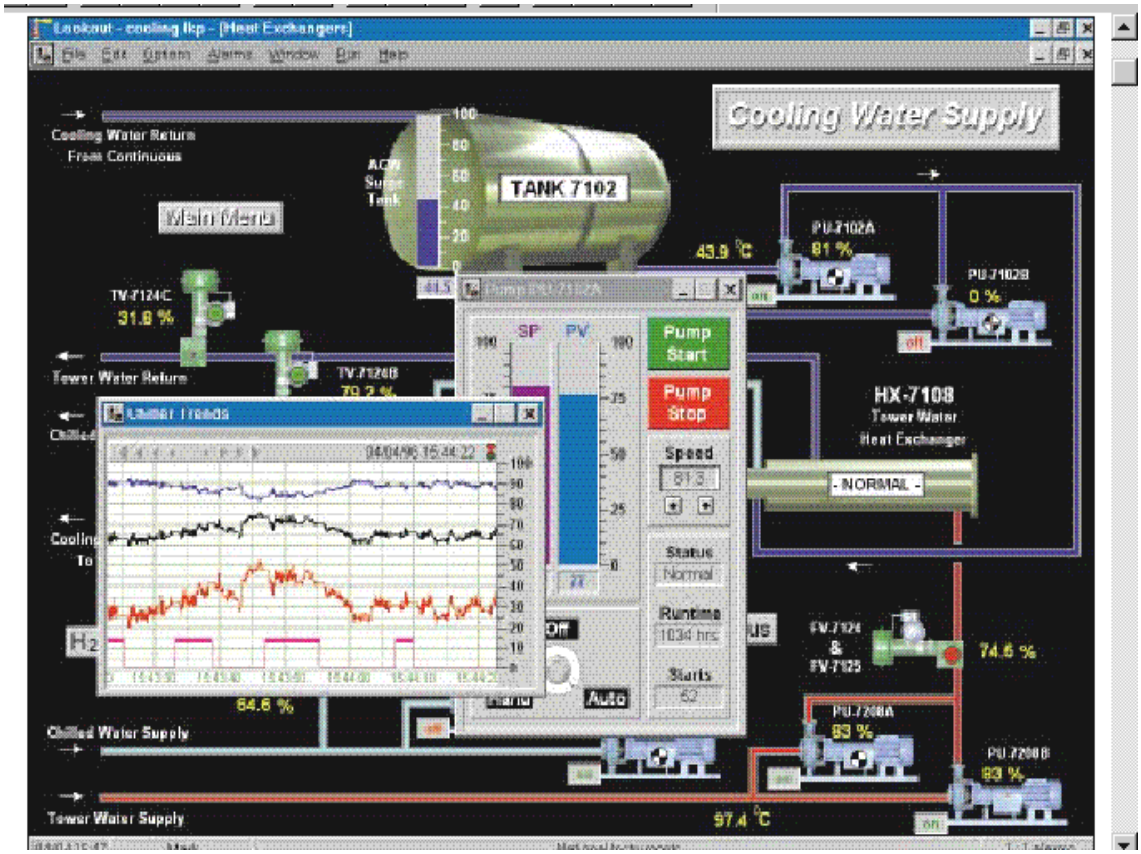


Figura 1.2- Ejemplo de una plataforma HMI desarrollada en Lookout.

1.5.- SISTEMA DE CONTROL DIGITAL DIRECTO (DDC)

El control digital directo tuvo sus inicios en los años 60, en este el computador llevaba a cabo todos los cálculos que efectuaban individualmente cada controlador, como controles P + I, P + I + D, los cuales generaban directamente las señales de control de las válvulas. Este tipo de control se denomina Control Digital Directo o DDC (Direct Digital Control) en el cual las técnicas digitales a más de fijar el valor

del setpoint y verificar el estado de la variable dinámica son utilizadas para realizar la comparación y dar una señal de corrección, las señales digitales son parte integrante del lazo de control como lo indica en la figura 1.3 . Este control realiza las siguientes funciones:

a. Explora las variables de entrada analógicas o digitales.

b. Las compara con el setpoint e introduce la señal de error en el algoritmo de control correspondiente.

c. Envía las señales de salida a los diferentes elementos de control final del proceso.

d. Se disponen de instrumentos analógicos en forma paralela con el computador en los puntos críticos y actúan como reserva en los casos de producirse fallas en el sistema[2].

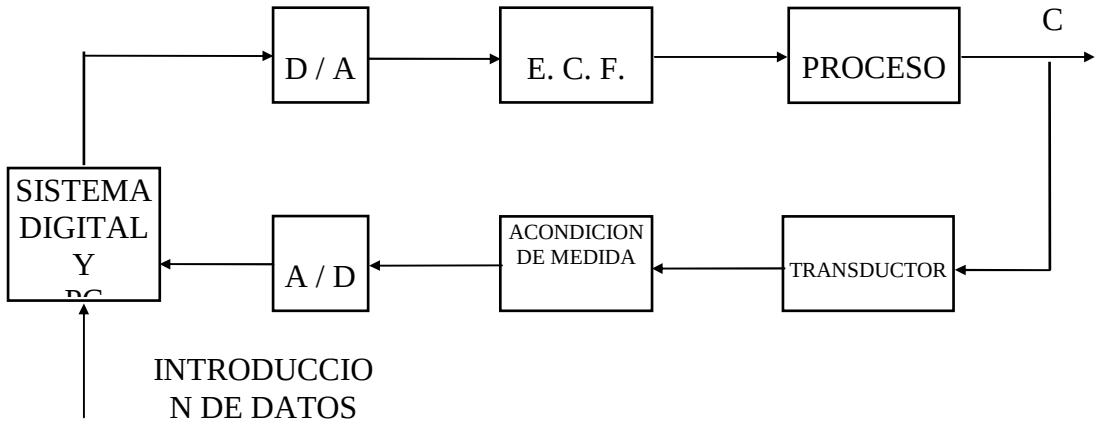


Figura 1.3 - Diagrama en bloques de un Control Digital Directo DDC.

1.6.- SISTEMA DE CONTROL DIGITAL POR SUPERVISION (DSC)

[2] CREUS Antonio "Instrumentación Industrial" Marcombo S.A. Barcelona España 1981 Págs. 553, 554.

Es una extensión natural de los sistemas de adquisición de datos (DAQ´s) en el que involucra el uso de una computadora en el proceso realimentado.

Las técnicas digitales son empleadas para verificar el estado de la variable dinámica y adicionalmente para fijar el valor del setpoint como lo muestra la figura 1.4.

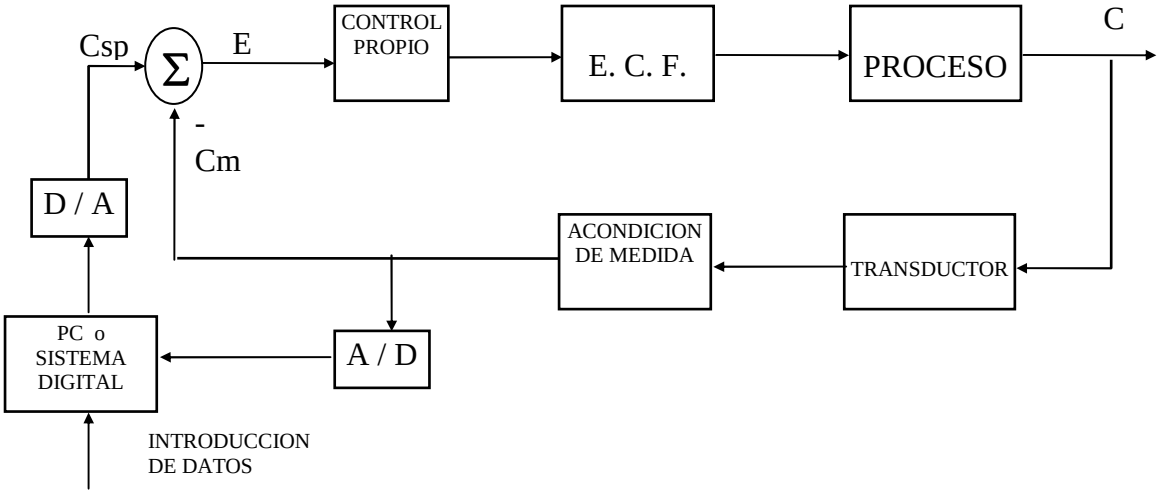


Figura 1.4- Diagrama de bloques de un sistema DSC

Varias cargas en ciertos procesos cambian y para aumentar la eficiencia y mantener la operación dentro de ciertos límites preestablecidos es ventajoso cambiar el valor del setpoint. El control digital por supervisión (DSC), también llamado **"control por puntos de consigna",** es una extensión del DATA LOGGING [3] .Con el desarrollo de las computadoras digitales de alta velocidad, con gran capacidad de almacenamiento se hace posible obtener datos interactivamente y en forma automática, monitoreando por medio de displays mediante un comando y ejecutando cálculos de los datos obtenidos para obtener una evaluación apropiada en las técnicas de control.

[3]CURIS Johnson, "Process Control Instrumentation Technology", Prentice Hall, Englewood, 1974

1.7.- TIPOS DE COMUNICACIÓN De los cuales tenemos.

1.7.1.- PUERTO SERIE[4]

El puerto serie o puerto RS-232 en los computadores personales es muy útil para controlar periféricos, interconexión con dispositivos tales como mouses, impresoras, microcontroladores, ordenadores, etc. Los canales seriales de la PC son programables, su comunicación es asíncrona y nos brinda la posibilidad de programar los bits de datos, bits de arranque, bits de parada, bits de paridad.

En la comunicación serial la información se transmite bit a bit por una o dos líneas de enlace de datos, lo que hace que disminuya significativamente el número de líneas; además la distancia de comunicación que puede ser extremadamente grande, para lo cual se ayuda de los equipos llamados módem. En la figura se presenta un ejemplo de comunicación serie.

Ventajas:

- Los canales seriales (COM1, COM2, etc.) son totalmente programables.
- Permite programar los bits de paridad y los bits de parada.
- Dispone de un sistema priorizador de interrupciones, que controla las interrupciones de transmisión, recepción, error y línea de estado.
- Existen recursos de diagnóstico a través de funciones loop back o lazo de realimentación.

Desventajas:

[4]RICARDO ZELENOVSKY "**IBM PC Para Ingenieros"** E.S.P.E (Facultad de Ingeniería Electrónica), Quito Ecuador, 1995 Cap. X.

- Permite solamente comunicación asíncrona.
- Requiere de más tiempo para la transmisión de datos.

Las señales del puerto serial de la PC están disponibles en un conector tipo D, macho de 9 o 25 pines, este conector generalmente se encuentra en la parte posterior del computador.

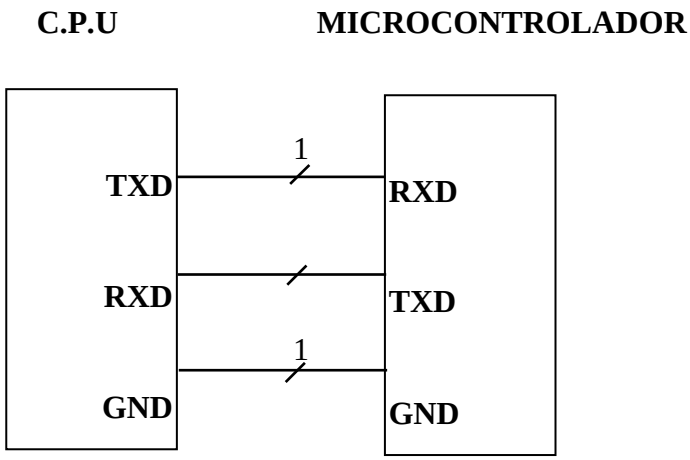


Figura 1.5 - Ejemplo de comunicación serial

1.7.2.- MODOS DE TRANSMISIÓN

Modo Símplex: Se transmite los datos en una sola dirección ejemplo del PC a la impresora.

Modo Dúplex (Half Dúplex): Se transmiten los datos en forma alterna, en este modo de comunicación la transmisión es en los dos sentidos pero no al mismo instante.

Modo Full Dúplex: Se transmite o se recibe datos al mismo tiempo por una sola línea virtual.

1.7.3.- PUERTO PARALELO

El puerto paralelo también conocido como el puerto de la impresora, permite la conexión con impresoras paralelas, también puede usarse como puerto I / O genérico y para comunicación entre PCs. En la figura 1.6 se observa un ejemplo.

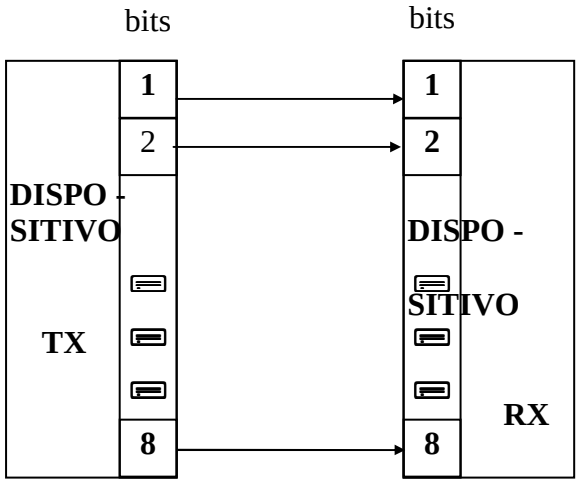


Figura 1.6- Ejemplo de comunicación paralelo.

Ventajas:

- Tiene una velocidad de comunicación muy alta (velocidad de compuerta).

Desventajas:

- Necesita un cable físico para cada línea.
- Se utiliza líneas independientes para la transmisión y recepción de datos.
- Sólo se puede utilizar a cortas distancias (máximo 7 metros).
- Necesita de un programa dedicado.

El puerto paralelo se representa por las letras LPT1 (puerto paralelo 1) LPT2 (puerto paralelo 2), todos los pines de I/O están disponibles en un conector tipo D de 25 pines hembra en la parte posterior del computador.

1.8 ACONDICIONAMIENTO DE SEÑALES. De los cuales tenemos.

1.8.1 MÓDULOS SCSI (SEÑALES ACONDICIONADAS PARA LA INSTRUMENTACIÓN)

Los dispositivos SCSI

En teoría, los dispositivos deben ser independientes de las operaciones del procesador. El ordenador solo necesita enviar los mandatos y los datos al periférico y esperar una respuesta (las impresoras. por ejemplo. tabajan de esta manera).

FOTO 8

FUENTE: EL AUTOR

En la actualidad, los puertos paralelo y serie son puertos específicos para dispositivos, de modo que el PC se limita a enviar la información al puerto, independientemente del dispositivo conectado, y no reconoce de forma automática (es decir, sin la intervención de controladores o del sistema operativo) cuál es el periférico que se está utilizando. Esto permite, por ejemplo, que dispositivos antiguos puedan funcionar en ordenadores recientes. En síntesis, es el concepto que emplea la interfaz SCSI. Los ordenadores y dispositivos se diseñan e integran sin la necesidad de asegurar compatibilidades de hardware concretas, que se aseguran por las especificaciones de la interfaz.

SCSI, que en términos de bus es un conjunto de cables, hilos y terminadores, cada uno con su nombre y propósito concretos. utiliza un conjunto limitado de instrucciones mandatos que permiten que tanto el sistema como el dispositivo se comuniquen a través del bus. Para ello se basan en el principio de independencia del dispositivo, característico de esta interfaz y una de sus propiedades principales. Por ejemplo, todas las unidades de disco duro se consideran semejantes en la interfaz SCSI (a excepción de su capacidad total) y, para una clase de dispositivos concreta, puede reemplazarse un disco instalado por otro nuevo, sin tener que modificar la configuración del sistema.

Dado que la iniciativa en la configuración y la comunicación reside en el dispositivo y no en el sistema, el ordenador es capaz de utilizar un reducido conjunto de mandatos estándar para garantizar la transferencia de datos hacia y desde el dispositivo.

Adaptadores SCSI

De forma similar al control de los discos duros para interfaz IDE, que se efectúa mediante una tarjeta específica (incorporada a la placa base en casi todos los modelos), los dispositivos SCSI utilizan un adaptador conocido como **controladora SCSI** o simplemente, tarjeta SCSI, que desde el punto de vista lógico de la interfaz se considera un dispositivo más. Actúa de puente entre el bus SCSI y el bus interno de entrada/salida del PC, enviando y recibiendo los mandatos necesarios y transfiriendo los datos de los dispositivos conectados al bus SCSI.

Iniciadores y objetivos

Básicamente, en un bus SCSI existen dos tipos de dispositivos: los **iniciadores** y los **objetivos**. Los primeros inician el proceso de comunicación cuando debe realizarse alguna tarea, mientras que los segundos responden a los mandatos e instrucciones de los iniciadores. Esta relación no se establece en un único sentido, ya que un dispositivo físico puede actuar a la vez como iniciador y como objetivo durante la transferencia de datos.

Un bus SCSI puede soportar varios dispositivos de forma simultánea pero en todo momento debe existir un iniciador y un objetivo. Aunque no sea necesariamente así siempre, en general, el adaptador SCSI (la tarjeta de expansión instalada en la ranura de la placa base) es el iniciador y los otros dispositivos (discos duros, lectoras o grabadoras de CD-ROM, por ejemplo) son los objetivos.

Identificadores

Un bus SCSI puede soportar un número elevado de dispositivos, pero para evitar cualquier conflicto en la comunicación, cada uno de ellos debe identificarse de manera individual. Si el bus admite hasta ocho dispositivos, cada uno de ellos se identifica con un número (ID) que puede tener un valor desde 0 hasta 7, o un valor comprendido entre 0 y 15, cuando son 16 los dispositivos soportados (incluyendo el adaptador, que se considera un dispositivo SCSI más).

La negociación

Dado que el cable utilizado para el bus SCSI es común a todos los dispositivos, cada uno de ellos debe obtener permiso de todos los restantes antes de tomar el control del bus, lo que se conoce como **fase de arbitraje**. En la **fase de selección** cuando uno de los dispositivos ha tomado el control (normalmente, el adaptador), procede a contactar con el dispositivo con el que debe comunicarse. Una vez establecido el contacto, comienza en realidad el proceso de comunicación.

Las distintas fases que intervienen en este proceso forman parte de lo que se conoce como **negociación**, que lleva a cabo un dispositivo de la cadena SCSI para acceder y utilizar el bus, y que comienza cuando el bus está libre. Esta circunstancia es conocida por todos los dispositivos mediante una señal específica, denominada BSY, que reciben por uno de los conductores del cable.

Se procede entonces a cambiar esta señal (de manera que el resto de dispositivos detecten el nuevo estado del bus) y se envía una señal por el conductor del cable correspondiente a su identificador.

Cables y conectores

La interfaz SCSI utiliza una topología de bus concreta. Cada uno de los dispositivos conectados al bus debe estar encadenado de forma lineal al dispositivo que le precede mediante un cable apropiado, por lo que el conjunto de dispositivos recibe el nombre de **cadena SCSI**. Se trata de una característica obligatoria. así que los dispositivos (incluyendo el adaptador) se conectan a uno o dos dispositivos más y nunca a un número mayor.

Los conectores SCSI

En la actualidad, existen ocho tipos distintos de conexiones SCSI (cuatro si no se diferencia entre conectores macho o hembra) incluyendo todas las aparecidas desde la publicación de las especificaciones oficiales de 1986, cuando la interfaz SASI se tomó como base para cl estándar SCSI.

FOTO 9

FUENTE : EL AUTOR

Los terminadores

Terminadores pasivos, que son los más antiguos y sencillos, indicados para los buses de velocidad reducida (SCSI simple).

Terminadores activos, similares a los anteriores pero con reguladores de voltaje que aportan mayor consistencia en la terminación del bus y que resultan indicados desde Fast SCSI hasta Ultra SCSI.

Por último, los terminadores **FPT** (*Forced Perfect Termination,* terminación perfecta forzada). Representan el tipo de terminadores más avanzado **y** además de reguladores de voltaje, utilizan diodos. Esto elimina prácticamente la posibilidad de que se refleje la señal. Ambos terminadores deben instalarse al final de la cadena SCSI, después de los últimos dispositivos, incluyendo los que puedan desconectarse

de la alimentación eléctrica mediante un interruptor o los que no se usen temporalmente.

CABLES SCSI

Cable A Cable original estándar con 50 conductores. Se utiliza para buses de ocho bits y en dos configuraciones diferentes:

normal (en general para dispositivos internos) y de alta densidad (más estrecho)

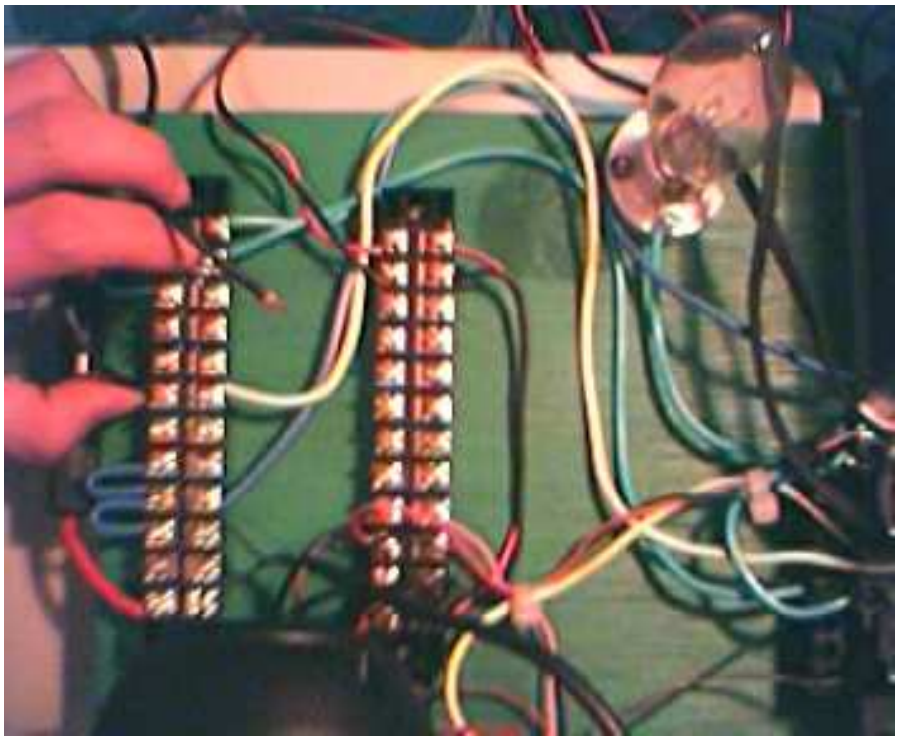
Cable B Para el estándar SCSI-2 y en los protocolos Wide SCSI. Prácticamente en desuso. Se combina con un cable A para poder disponer de los 68 conductores necesarios en buses de 16 bits.

Cable P Cable de 68 conductores, definido en el estándar SCSI-3. Reemplaza la combinación de cables A y B más antigua en los buses de 16 bits.

TIPOS DE DISPOSITIVOS

Acceso aleatorio	Escáneres
(discos duros)	Discos magnetoópticos
Acceso secuencial	Dispositivos de comunicación
(unidades de cinta)	Escáneres
Impresoras	Discos magnetoópticos
Unidades de CD-ROM	Dispositivos de comunicación

FOTO 10



FUENTE: EL AUTOR

Los estándares

Cada uno de los protocolos SCSI se engloba en uno de los estándares aprobados por el comité ANSI, cuyas especificaciones se detallan en la documentación oficial. Estos estándares son:

SCSI-1

Desde 1986. Es el más antiguo y se basa en buses de ocho bits y transferencia asíncrona para todos los comandos y datos. Utiliza tecnología bipolar y terminaciones pasivas. Los conectores, tanto internos como externos, son de 50 patillas (el conector externo se conoce como **Centronics 50**).

SCSI-2

Desde 1994. Utiliza transferencia asíncrona para el envío de comandos y síncrona para los datos. Permite terminadores pasivos y activos y señales diferenciales (tecnología denominada HVD, *High Voltage Differential* o diferencial de alto voltaje).

Para los protocolos Wide (de 16 bits), se usan conectores B de 68 patillas.

SCSI-3

A partir de 1996. Es una colección de documentos que cubren las especificaciones físicas, el protocolo básico de la intcrfaz y el conjunto de mandatos primarios. Cada uno de estos documentos tiene sus propias versiones y revisiones.

Los principales son:

SPI (SCSI Parallel Interface)

Define las conexiones y señales eléctricas del bus SCSI paralelo, incluyendo el cable y conector P de alta densidad, con 68 patillas.

Fast-20

Conocido también con el nombre de **Ultra SCSI**, contiene las secciones en las que se define la interfaz de bus ancho que duplica la transferencia de SPI.

Fast-40

Especificaciones de la interfaz SPI-2 (correspondiente al protocolo Ultra2 SCSI que alcanza un ancho de banda de hasta unos 80 MB/s, con una nueva interfaz eléctrica LVD (*Low Voltage Differential,* diferencial de bajo voltaje).

SPI-3

Cubre las especificaciones de la interfaz Fast-80DT conocida con el nombre de **Ultra3 SCSI**, en la que se introduce corrección de errores **CRC** (*Cyclic Redundancy Check,* comprobación de redundancia cíclica).

EPI (*Enhanced Parallel Interface,* interfaz paralela ampliada)

Documentación técnica que describe el diseño de sistemas SCSI. Entre otros aspectos, detalla la longitud de los cables y los parámetros eléctricos que permiten combinar en un mismo sistema dispositivos para bus normal (50 polos) y ancho (68 polos**).**

CARACTERISTICAS BASICAS

Cada especificación SCSI utiliza una velocidad de frecuencia y un ancho de bus característicos, que determinan la cantidad máxima de información que puede transmitirse, calculada mediante el producto de estos dos parámetros. La tabla siguiente recoge estos valores para los protocolos existentes.

Protocolo de interfaz	Estándar	Velocidad del bus (MHz)	Ancho del bus	Ratio de transferencia (MB/s)	Longitud máxima del cable	Número máximo de dispositivos

					SE (m)	por bus
SCSI	SCSI-l	5	8	4.77	6	8
Fast SCSI	SCSI-2	10	8	9.54	3	8
Wide SCSI	SCSI-2	5	16	9.54	6	16
Fast Wide SCSI	SCSI-2	10	16	19.07	3	16
Ultra SCSI	SCSI-3	20	8	19.07	1.5 (**)	8
Ultra Wide SCSI	SCSI-3	20	16	38.15	1.5 (**)	16
Ultra2 SCSI	SCSI-3	40	8	38.15	12 (LVD)	8
Ultra2 Wide SCSI	SCSI-3	40	16	76.29	12 (LVD)	16
Ultra3 SCSI	SCSI-3	40 (*)	16	152.59	12 (LVD)	16

(*) Ultra3 y la implementación Ultra 160/m. utilizan la misma frecuencia que Ultra2, si bien envían dos bytes por cada ciclo de reloj. Esto puede inducir a concluir erróneamente que utilizan buses de 32 bits.

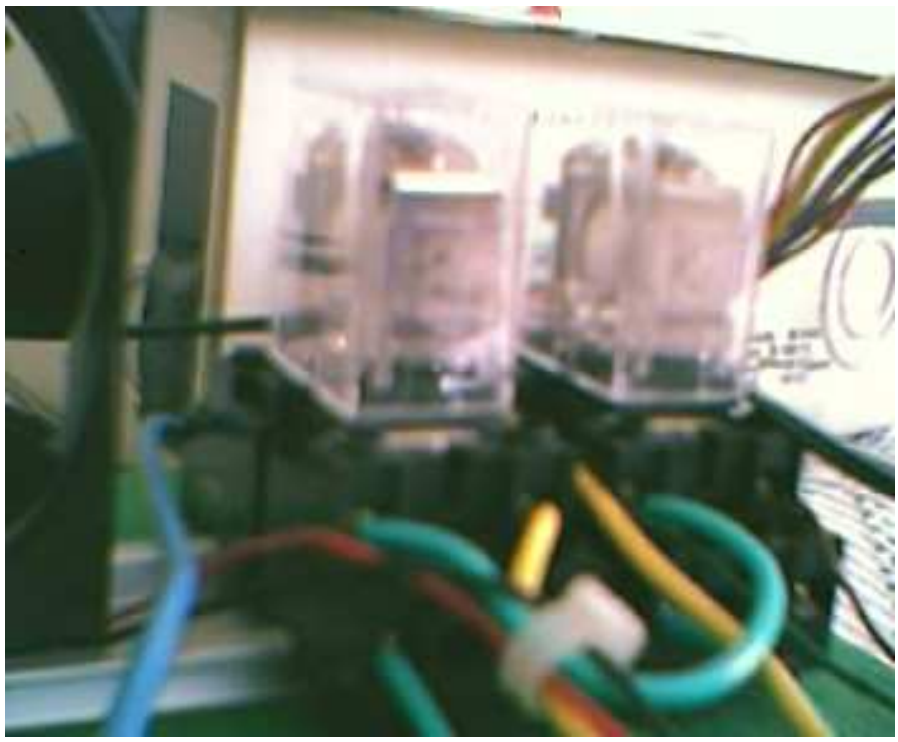
(**) Ultra SCSI y Ultra Wide SCSI pueden utilizar también cables con una distancia máxima entre dispositivos de cuatro metros, aunque en este caso el número máximo de éstos se reduce a cuatro y ocho respectivamente.

1.8.2 PLATAFORMAS VXI.- Los sistemas basados en VXI ofrecen otras ventajas que van más allá del alto número de puntos de medida. Hay una gama extraordinariamente amplia de conmutadores con diversas topologías y componentes de conmutación. Incluyen conmutadores de armadura, de láminas, de estado sólido, de RF y de microondas. Se pueden configurar en escáners, multiplexores y matrices o elementos de conmutación para usos generales. Se pueden optimizar para velocidad de conmutación, frecuencia, potencia, resistencia de contacto, fiabilidad o flexibilidad. El VXI ofrece dos tamaños de módulos, los "B" y "C". El tamaño "C" ofrece el mayor número de puntos de relés. El tamaño "B" más pequeño ofrece una densidad de relés reducida a un precio inferior. Los módulos del tamaño "B" funcionan también muy bien en un equipo principal de tamaño "C", mediante el uso de un adaptador.

En resumen, puede decirse que el VXI ofrece un medio excepcional de solucionar los problemas y aprovechar las oportunidades creadas por los cambios que se están produciendo en la industria electrónica. Proporciona una solución muy conveniente para tratar las aplicaciones de medidas existentes y de nueva aparición, que implican una gran variedad y un gran número de puntos y bloques asociados de información.

Su factor de forma proporciona la proximidad necesaria a los DUTs (dispositivos bajo prueba) y su excelente selección de topologías de conmutadores y configuraciones de relés la convierten en una solución muy atractiva.

FOTO 11



FUENTE: EL AUTOR

Esta capacidad de conmutación y la flexibilidad de sus soluciones de interconexión, combinadas con el excepcional entorno operativo estrechamente acoplado y la coordinación de la temporización, hacen que el VXI resulte una opción muy atractiva.

1.8.3 MÓDULOS PXI (Componentes periféricos para la instrumentación).

El diseño de la instrumentación es un campo competitivo, y los fabricantes se están esforzando constantemente aumentar funcionalidad mientras que simultáneamente reducen el tiempo-a-mercado y el coste. Para resolver estas demandas, los sistemas de la instrumentación se han desarrollado más de cerca a la palancada que existía, tecnologías disponibles. El estándar modular de la instrumentación de PXI, adoptado en 1998, es sí mismo un estudio en la tecnología existente del leveraging para entregar el alto rendimiento, instrumentación modular del bajo costo. Es estas calidades que condujeron 38 por ciento de crecimiento del mercado de PXI en 2001, según un informe de la industria por Frost y Sullivan, un año en que la mayoría de las otras plataformas de la prueba consideraron declinaciones grandes.

Dos elementos dominantes de un estándar modular de la instrumentación son la interconexión del módulo y el empaquetado (o forme el factor). En esta generación más reciente de instrumentos, los diseñadores dieron vuelta al autobús componente periférico de la interconexión (PCI) debido a su velocidad, su tarifa de transferencia de explosión de 132 Mbytes/sec es casi 10 por el de estándares anteriores, y sus modos numerosos para el control de módulo. Otra ventaja de usar el PCI era que los diseñadores podrían dibujar del mercado extenso del ordenador personal (PC) para los componentes del software y del bajo costo.

Aunque la PC era un recurso excelente, el empaquetado de escritorio de la PC no fue satisfecho a los ambientes industriales exigentes previstos para los nuevos diseños del instrumento. Las PC no fueron diseñadas para los ambientes industriales, y por lo tanto no proporcionan la conectividad, la ayuda que se refrescaba, la resistencia de la EMI, y la capacidad de la ranura necesitada para muchos sistemas de alto rendimiento de la instrumentación. Para solucionar estos problemas, los diseñadores adoptaron ruggedized el estándar de la Euro-Tarjeta para empaquetar.

Este acceso permitido a la historia larga del diseño y a la amplia ayuda del vendedor para los diseños del tablero y del recinto. Con la interconexión de menor importancia y los cambios de empaquetado, el estándar de CompactPCI fue propuesto y tiene desde aceptación en el mercado de aumento ganada.

Sin embargo la coordinación de las operaciones del multi-módulo exigió funcionalidad adicional. Las extensiones de VMEbus para el estándar de la instrumentación (VXI), un estándar modular anterior del instrumento, demostraron la ventaja de incluir señales auxiliares tales como una referencia común de la frecuencia del sistema, disparadores controlados de la sincronización, y un sistema genérico que

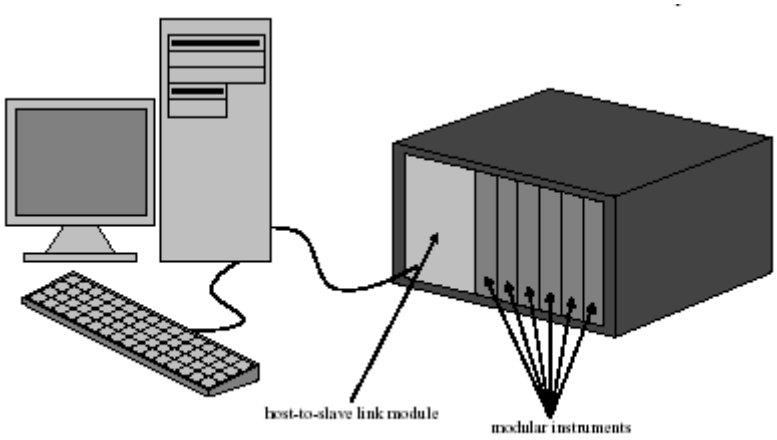
señalaba banderas. Otra vez con una cierta modificación, el sistema de la señal del auxiliar VXI fue agregado al estándar de CompactPCI para formar las extensiones del PCI para el estándar de la instrumentación (PXI). Por diseño, la mayoría de los productos de PXI son completamente inter-operables con los productos de CompactPCI.

FOTO 12

FUENTE: EL AUTOR

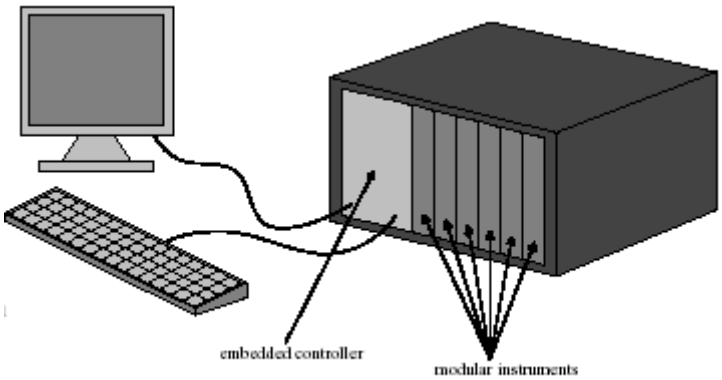
De sus raíces en el ordenador personal dinámico (PC), la Euro-Tarjeta rugosa, y los mercados de la herencia VXI, la especificación de PXI está proporcionando la dirección para una nueva generación de un rendimiento más alto, equipo de prueba de un costo más bajo.

¿Cómo utilizo PXI?



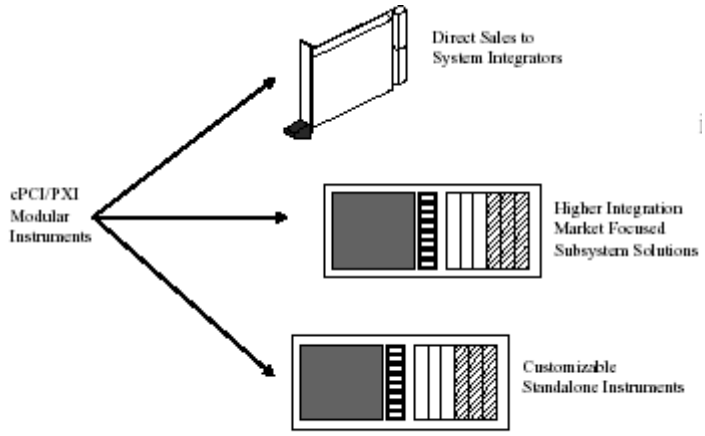
Host-Slave Operation

Los módulos de PXI se diseñan para poner funciones en ejecución específicas, tales como análisis de la captura de la señal análoga, de la señal del RF, o generación de la forma de onda. Para controlar estas funciones, un chasis de CompactPCI o de PXI y un regulador del sistema son necesarios. Según lo mencionado antes de que la mayoría de los módulos de PXI se puedan mezclar libremente con los módulos de CompactPCI, solamente las señales auxiliares de PXI tales como la referencia común de la frecuencia no estarán disponibles. Dos configuraciones están disponibles: operación auxiliar del chasis con un anfitrión de escritorio u operación independiente del chasis con un regulador encajado. En el desarrollo y usos superiores del instrumento del banco, la manera más económica de utilizar los instrumentos de PXI está con una PC de escritorio para el control y un acoplamiento del anfitrión-a-esclavo para la conexión a los módulos de PXI. El acoplamiento del anfitrión-a-esclavo actúa mientras que un puente normal del PCI y los módulos de PXI aparecen como dispositivos normales del PCI debajo del encargado del dispositivo de hardware de PC. Esta configuración permite el acceso completo al ambiente de la PC en un precio relativamente barato.



Standalone Operation

Para muchos otros usos, los instrumentos de PXI residirán en última instancia en un estante con el otro equipo tal como encendido un piso de la fábrica o se desea la operación alejada del sitio o portable. En estos ajustes el uso de las PC del tablero del escritorio no es factible y una PC encajada debe ser utilizada. Un monitor y un teclado pueden ser unidos al regulador para el desarrollo o eliminar errores y después ser quitados o ser substituidos por una conexión del LAN para un uso más último. Esta configuración es algo más alta en coste pero permite la mayoría de la flexibilidad de tamaño del ajuste operacional y de un sistema más pequeño. (Adaptado de un artículo por la visita de ZTEC Inc. el Web site de ZTEC en www.ztec-inc.com)



ZTEC's "PXI in 2002" Marketing Approach

1.9 INTERFACE DE COMUNICACIÓN GPIB IEEE486.-Encontramos

Instrumentación programada: En los tiempos actuales, el precio de la mano de obra supone uno de los costes más onerosos en los procesos de verificación y ensayo.

Además se debe tener en cuenta la posibilidad del error humano en un proceso de test medianamente duradero. La instrumentación programada ofrece una solución técnica y económica al problema, mediante la explotación de los interfaces de instrumentación más estándar como son GPIB, VXI y el puerto serie RS-232. El otro campo de trabajo de este grupo consiste en el diseño y/o explotación de los sistemas de adquisición de datos procedentes de variables físicas mediante transductores, dentro del cual se encuentran las tarjetas de adquisición de datos, registradores, etc.

La experiencia adquirida por este grupo de trabajo trata de alcanzar la solución óptima con el mínimo coste en equipamiento y mantenimiento, por parte del usuario.

Tal vez uno de los patrones más importantes es la interfase digital IEEE 488, para prueba de instrumentación programable y otros equipos. Estandarizar la interfase entre equipos de prueba permite la conexión entre piezas de equipo de prueba de laboratorio, sin importar su fabricación para crear avanzados sistemas de equipo de prueba automáticos ATE (Automatic Test Equipment).

FOTO 13

FUENTE: EL AUTOR

El bus GPIB se basa en la transmisión de palabras de datos de ocho bits, con un bus de datos paralelo de ocho líneas. Se utiliza varios bits de estado para aumentar el datos de ocho bits pero se transmiten por líneas separadas. El sistema 488 es básicamente un sistema de corta distancia para equipos de prueba montados en un gabinete dentro de una misma habitación y no para la transmisión a largas distancias, vía telefónica ni por cualquier otro medio de comunicación. Las 16 señales activas :

* Bus de datos (8 líneas)
* Bus de control de transferencia de datos Handshake (3 líneas).
* Bus para el control general de la interconexión (5 líneas).

CAPITULO II

EL PUERTO PARALELO O LPT1

Historia, evolución y tipos de puertos paralelos

Historia

En 1981, la IBM (International Business Machines) introdujo la Computadora Personal (PC). El puerto paralelo (Standart Parallel Port SPP) estaba incluido en el primer PC y se agregó a éste como una alternativa al bajo rendimiento del puerto serial, para utilizarlo como controlador de las impresoras de matriz de punto de alto desempeño. Este puerto tenía la capacidad de transmitir 8 bits de datos a la vez (del PC a la impresora), mientras que el puerto serial lo hacía de uno en uno. En el momento que el puerto paralelo fue presentado, las impresoras de punto fueron el principal dispositivo externo que se conecto a éste. Al hacerse extensamente utilizado, el puerto paralelo llegó a ser la respuesta para conectar dispositivos más rápidos.

Después de este inicio, tres grandes grupos de problemas aparecieron a los desarrolladores y usuarios de este puerto: Primero, aunque éste había aumentado su velocidad considerablemente, no había cambio en la arquitectura o desempeño. La máxima velocidad de transferencia alcanzable estaba por los 150 kbyte /seg. y era extremadamente dependiente del software. Segundo, no había un estándar para la interfase eléctrica. Esto causaba muchos problemas cuando se quería garantizar la operación en múltiples plataformas. Por último, la forma de diseño que le dieron, limitaba la distancia de los cables externos hasta un máximo de 1,8 metros. En 1991 hubo una reunión de fabricantes de modo que se pudiera desarrollar un nuevo

estándar para el control inteligente de impresoras a través de una red. Estos fabricantes, donde estaban incluidos Lexmark, IBM, Texas Instruments y otros, formaron la Network Printing Alliance (NPA), como una respuesta a estas necesidades.

Evolución

Desde la introducción del PC al mercado, el puerto paralelo ha sufrido varias modificaciones para hacerlo más veloz. Ya que el puerto original era unidireccional, se creó el puerto bidireccional. El puerto bidireccional fue introducido con el PS/2 compatible. Este permite una comunicación de 8 bits en ambas direcciones. Algo interesante de notar es que el puerto original tenía la posibilidad de ser bidireccional realizando una conexión entre dos pines de un componente electrónico que venía incluido en éste. (Dicho de otro modo, el puerto original es bidireccional en diseño básico, pero no en el diseño operacional). Finalmente se han creado el Enhanced Parallel Port (EPP) y el Extended Capability Port (ECP). Estos dos últimos son al puerto estándar como el Pentium al 286, además de ser bidireccionales.

Inicialmente el puerto paralelo se utilizó para la comunicación con impresoras. Actualmente se utiliza también para manejar otros periféricos como CD ROM, cintas de copia de respaldo, discos duros, tarjetas de red, protectores de copia, scanners, etc.

Tipos de puerto paralelo

En la actualidad se conoce cuatro tipos de puerto paralelo:

- Puerto paralelo estándar (Standart Parallel Port SPP)
- Puerto Paralelo PS/2 (bidireccional)

- Enhanced Parallel Port (EPP)
- Extended Capability Port (ECP)

En la siguiente tabla se muestra información sintetizada de cada uno de estos tipos de puertos.

	SPP	**PS/2**	**EPP**	**ECP**
Fecha de Introducción	1981	1987	1994	1994
Fabricante	IBM	IBM	Intel, Xircom y Zenith Data Systems	Hewlett Packard y Microsoft
Bidireccional	No	Si	Si	Si
DMA	No	No	No	Si
Velocidad	150 Kbyte/seg.	150 Kbytes/seg.	2 Mbytes/seg.	2 Mbytes/seg.

Manejo del puerto paralelo

En las aplicaciones que involucran el computador y la adquisición de datos es necesario emplear alguno de los puertos del computador, o bien realizar la aplicación enfocada a trabajar con una tarjeta de expansión, ya sea de estándar ISA (casi desaparecido) o bien con tecnología PCI. Estas dos ultimas opciones, a pesar de ser completamente posibles no se consideran adecuadas puesto que son tecnologías cambiantes, que obligan a una renovación del hardware de la aplicación, además de tener que destapar la caja del equipo para instalarlas. Son estas razones, además de la

facilidad en el manejo del puerto paralelo, la que nos motiva a trabajar apoyados en este.

En general el puerto paralelo de cualquier computador personal puede trabajar en varios modos dependiendo de aspectos como si se trata de entrada y salida de datos, si se requieren el uso de la memoria del puerto, etc. El modo mas empleado, que además lo poseen todos los computadores sin importar que tan viejo sea es el modo estándar o spp (standard parallel port).

El modo spp esta diseñado empleando un enfoque en el que el computador envía datos a través del puerto hacia una impresora y recibe únicamente algunas señales de control, por lo que el puerto se maneja empleando tres registros: el de datos, que funciona como salida de datos; el registro de control, que posee función mixta de entrada y salida; y el de estado que funciona como entrada.

Como ya se menciono, en modo SPP, estos pines están agrupados en tres registros y se encuentran mapeados como tal en el computador. La ubicación de estos registros varia de acuerdo a tres direcciones base usadas para direccionarlos como se indica en la siguiente tabla.

Dirección Reg. Datos	Dirección Reg. Status	Dirección Reg. Control
Base	Base + 1	Base + 2
3BCh	3BDh	3Beh
378h	379h	37Ah
278h	279h	27Ah

Puesto que ni el registro de estado ni el de control poseen por si mismos 8 bits libres para la entrada de datos es necesario realizar un arreglo empleando 5 bits de un registro y 3 del otro. En la siguiente tabla se indica como es la distribución de pines

que se empleara en el puerto para recibir el dato resultante de la conversión análogo-digital.

ADC 0808		Puerto Paralelo		
Bit	Pin #	Registro	Bit	Bit#
2^{-8}	17	Control	Strobe	0
2^{-7}	14	Control	AutoLF	1
2^{-6}	15	Control	Reset	2
2^{-5}	8	Estado	Error	3
2^{-4}	18	Estado	Select	4
2^{-3}	19	Estado	Paper	5
2^{-2}	20	Estado	Ack	6
2^{-1}	21	Estado	Busy	7

Esto supone que del bit 2-8 (el de menor peso en la salida del ADC) al 2-6 se reciben por el registro de control, mientras que los restantes 5 bits se reciben por el registro de estado. Es de tener en cuenta que, como se indica en los datos de la figura , se tienen algunos bits invertidos dentro de los registros empleados, por lo que es necesario invertirlos. Por la facilidad de manejo se determino realizar la adecuación de los datos, esto es generar mascaras para aislar los bits que nos interesan de cada registro e invertir los que sea necesario por software, asunto que se explica en la siguiente sección.

2.1 EL PUERTO DE SALIDA DE 8 BITS 378.-

La figura 2.1siguiente muestra como los registros están conectadas a los pines del conector.

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit3**	**Bit 2**	**Bit 1**	**Bit 0**
PINES	9	8	7	6	5	4	3	2

Figura 2.1 registros de 8 bits para salida.

Estos pines deberán ser capaces de “ source/sink” 2,6/24 mA. Una institución de OUT escribe directamente en los pines del conector. Al escribir 1 en un bit, resulta

un nivel TTL alto en la salida. Este registro también puede ser leído con una instrucción IN, lo que permite verificar si los datos están siendo correctamente transferidos. Notar que esto no funciona como entrada, ya que lee la salida del 373.

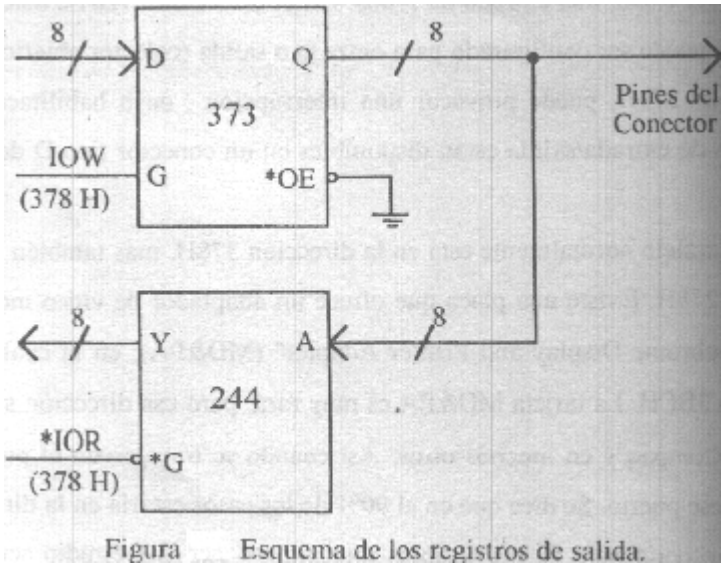


Figura 2.2 Esquema de los registros de salida.

2.2 EL PUERTO DE ENTRADA DE 5 BITS 379.- La figura 2.3 siguiente indica como este registro está ligado a los pines del conector.

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0.**
PINES	11 (L)	10	12	13	15	_	_	_

(L) indica una inversión.

Figura 2.3 Registro de 5 bits para entrada.

Una lectura desde esta dirección, con una instrucción IN, refleja el estado inmediato de estos pines. El pin 10 puede ser utilizado para provocar una interrupción

(IRQ7), con una transición de bajo para alto (↑). Es necesario que el bit 4 del registro 37AH esté en 1.

2.3 EL PUERTO BIDIRECCIONAL DE 4 BITS 37A.- La figura 2.4 siguiente indica como este registro está ligado a los pines del conector.

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
PINES	_	_	_	Hab IRQ7	17(L)	16(H)	14(L)	1(L)
RESET	_	_	_	0	1	0	1	1

Figura 2.4 (L) indica que el pin es invertido.

Registro de 4 bits bidireccional.

El bit es utilizado para controlar la habitación de IRQ 7. Cuando este bit es colocado en 1, la interrupción puede ocurrir. Estas salidas son accionadas por colector abierto. Esto les permite trabajar tanto como entrada como salida. Ellas están conectadas a Vcc por resistores de 4,7 K (pull-up); pueden recibir hasta 7 mA y todavía mantener un nivel bajo de 0.8 V.

Para funcionar como salida, se debe usar la instrucción OUT, para escribir en los pines. Para ser usada como entrada, es necesario antes programar todas las salidas en 1, esto va a permitir que elementos externos tengan para 0, cada una de las entradas. Al configurarse ese puerto para entrada, o sea, al escribir 1 en todos los pines, debe tomarse en cuenta que algunos pines están invertidos.

Cuando trabajan como salida, ese puerto (como cualquier salida a colector abierto) presenta una demora en las transiciones de bajo para alto, pues toda la carga

es hecha por el “pull up” pasivo de 4,7 K Ω. Es mas conveniente usar ese puerto como entrada.

2.4 PINES DEL PUERTO PARALELO. La tabla de la figura 2.5 muestra un resumen del puerto paralelo, ordenado de acuerdo a las direcciones, de los bits de los registros:

Pin No (D-Type 25)	Pin No (Centronics)	SPP Signal	Direction In/out	Register	Hardware Inverted
1	1	nStrobe	In/Out	Control	Yes
2	2	Data 0	Out	Data	
3	3	Data 1	Out	Data	
4	4	Data 2	Out	Data	
5	5	Data 3	Out	Data	
6	6	Data 4	Out	Data	
7	7	Data 5	Out	Data	
8	8	Data 6	Out	Data	
9	9	Data 7	Out	Data	
10	10	nAck	In	Status	
11	11	Busy	In	Status	Yes
12	12	Paper-Out PaperEnd	In	Status	
13	13	Select	In	Status	
14	14	nAuto-Linefeed	In/Out	Control	Yes
15	32	nError / nFault	In	Status	
16	31	nInitialize	In/Out	Control	
17	36	nSelect-Printer nSelect-In	In/Out	Control	Yes
18 - 25	19-30	Ground	Gnd		

Figura 2.5 Direcciones de los bits de los registros en puerto paralelo.

2.5 COMUNICACIÓN USANDO EL LPT1.- El puerto paralelo es una atractiva solución para comunicación entre computadoras. Esto se debe a dos motivos principales:

- Más veloz que el puerto serial,
- No necesita de ninguna interface.

Al lado de estas ventajas, algunos problemas existen:

- Confección del cable,
- Necesidad de un programa dedicado.

En este ítem se hace el estudio del problema y la presentación de una posible solución. Lo que se espera de una conexión entre dos computadores es que cada una de ellas pueda recibir o transmitir, inclusive simultáneamente, o sea, se busca una conexión “full duplex”. Por este motivo el cable paralelo debe tener dos vías de datos: una para recibir y otra para transmitir.

Para el envío de los datos, se necesita de dos líneas de “ handshake”: STROBE y ACKNOWLEDGE. Usando STROBE, el transmisor avisa al receptor de que hay un nuevo dato en la vía de datos. Usando ACNOWLEDGE, el receptor avisa al transmisor de que ya leyó el dato que está en la vía de datos.

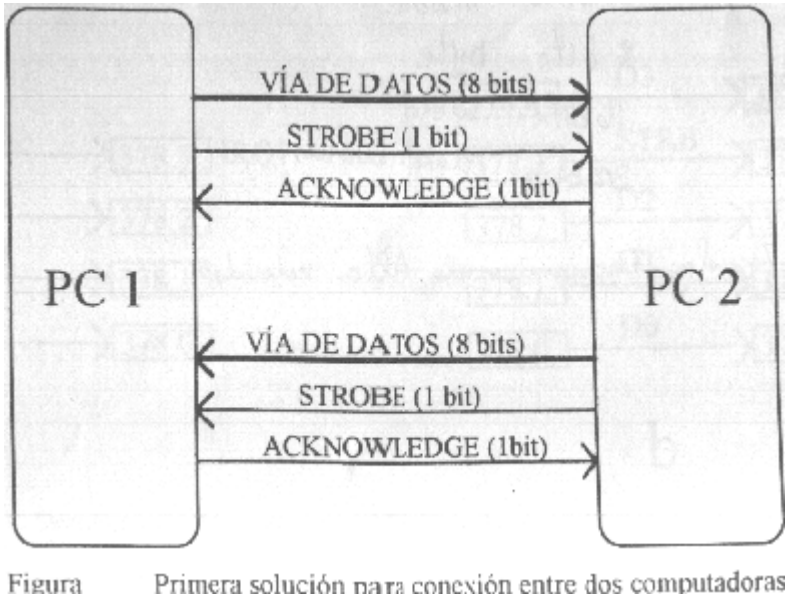


Figura Primera solución para conexión entre dos computadoras

Figura 2.6 Primera solución para conexión entre dos computadoras.

2.6 PROTOCOLOS DE ENLACE DEL LPT1.- Para realizar la transmisión y recepción de los datos, la solución mas sencilla es utilizar un protocolo semejado al de la impresora. El transmisor coloca la información en la vía de datos y pulsa la línea de STROBE para alertar al receptor; este al percibir un pulso de STROBE , lee los datos y pulsa la línea de ACKNOWLEDGE, indicando que la vía de datos está libre. Esto significa que cada computadora debe detectar dos pulsos: ACK_IN y STRB_IN. La única forma de detectar pulsos con seguridad es utilizando interrupción habilitada por flanco. En la PC hay apenas una interrupción destinada al puerto paralelo (IRQ7) y que ya está destinada a la línea STROBE. O sea, no hay como detectar con seguridad el pulso de ACKNOWLEDGE. Por eso se adopta un protocolo basado en niveles.

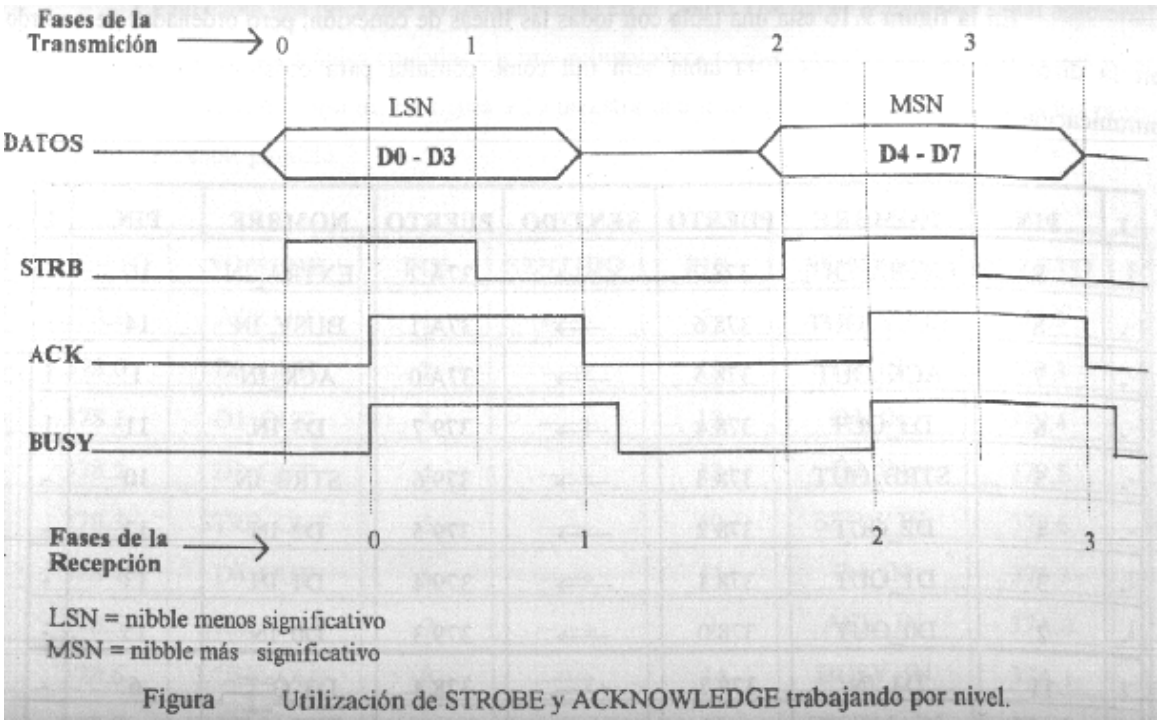


Figura 2.7 Utilización de STROBE y ACKNOWLEDGE trabajando por nivel

Para transmitir un byte, en este canal full duplex, se utiliza un protocolo de 4 fases. A continuación hay una descripción detallada de estas 4 fases, abordando en separado la transmisión y la recepción. Como el nivel de activación de las líneas es totalmente irrelevante, se utilizó la activación en nivel alto. En esta descripción T significa señal activada y F señal desactivada.

TRANSMISIÓN :

FASE 0: si BUSY=F, entonces escribir el LSN, STRB=T, fase=1;

FASE 1: si ACK=T, entonces STRB=F, fase=2;

FASE 3: si ACK=T, entonces STRB=F, fase=0;

RECEPCIÓN:

FASE 0: si STRB=T, entonces leer el LSN, ACK=T, BUSY=T, fase1;

FASE 1: si STRB=F, entonces ACK=F, BUSY=F, fase2;

FASE 3: si STRB=F, entonces ACK=F, BUSY=F, fase 0;

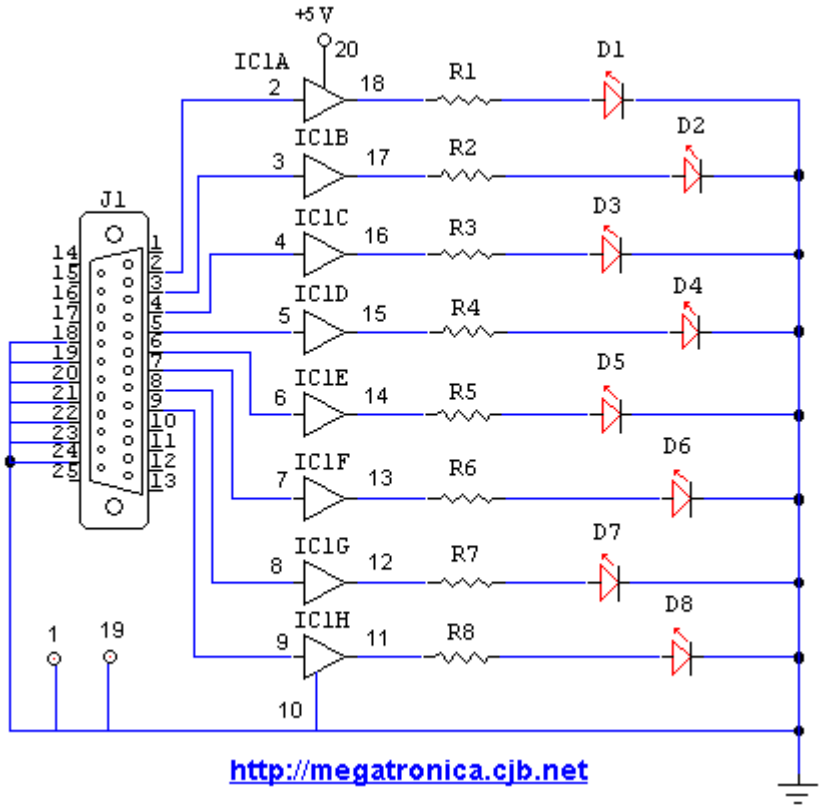
En los algoritmos de transmisión y recepción notase que hay una gran semejanza entre las fases 0,2 y 1,3. En este ejemplo la señal de BUSY no tiene función y solamente refleja la señal de ACKNOWLEDGE pero, esta línea puede ser usada para permitir protocolos de comunicación de mayor complejidad.

2.7 APLICACIONES.

Veamos cómo controlar el puerto a través de Visual Basic (cualquier versión de 32 bits). Para ello, necesitaremos tener en nuestro directorio de sistema

("C\windows\system") una librería especial llamada INPOUT32.DLL. Además de la mencionada librería se incluye un módulo (denominado "inpout32.bas") con las correspondientes llamadas a la DLL. Una vez descargada la librería basta agregar el módulo al proyecto para disponer entonces de la instrucción out, similar a BASIC.

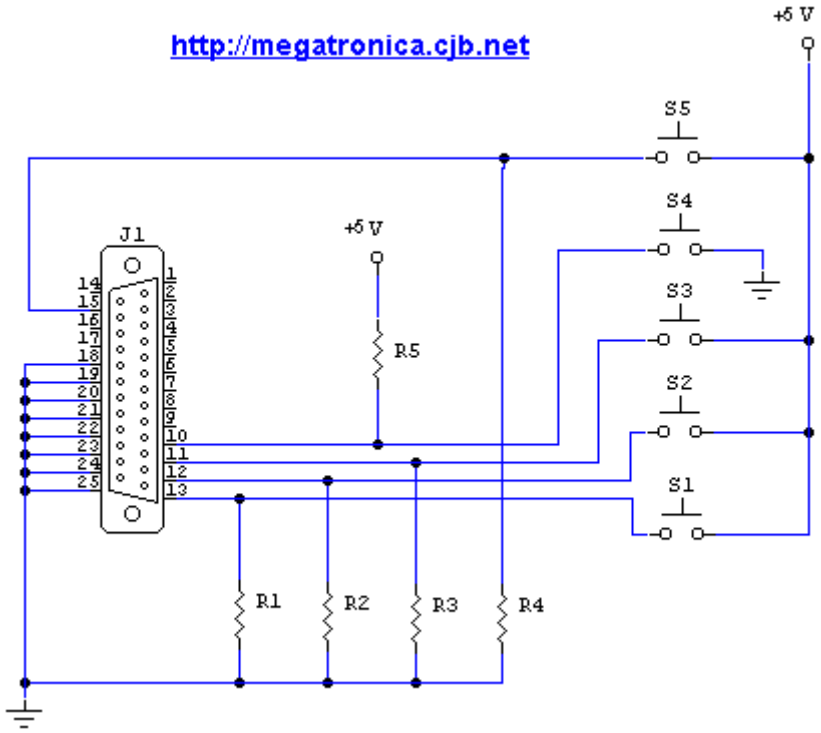
En este momento estamos en condiciones de desarrollar un primer circuito básico de prueba. Éste va a consistir en prender o apagar 8 Leds conectado cada uno a los pines D0 a D7. El diagrama esquemático es el siguiente:



En donde J1 es la ficha DB-25 del puerto paralelo, R1 a R8 son resistencias de 1/8W de 330 ohms. D1 a D8 son leds. Finalmente IC1 es un SN74AC541 o similar.

Una vez montado, este circuito será de gran utilidad para probar cualquier programa que utilice el puerto paralelo. Las distintas aplicaciones de éste son inmensas.

Para leer datos desde el puerto se utiliza otra instrucción que, como la anterior, posee una sintaxis similar en los diversos leguajes. Para BASIC y VISUAL BASIC, se escribe como inp. En "C" este comando recibe el nombre de inport o bien inportb (ambas en la cabecera "dos.h"). Finalmente en ensamblador la intrucción es inp. En todos los casos el único argumento de la instrucción es la dirección que, en este caso, corresponde con la del puerto más uno. Recordemos que se disponen 5 líneas de entrada y que una de ellas se encuentra negada. Un circuito típico para probar la lectura del puerto paralelo es el siguiente:



Nuevamente J1 es la ficha DB-25. R1 A R5 son resistencias de 1/8 W 10 kiloohms. S1 a S5 son pulsadores o llaves. Nótese que el circuito tiene en cuenta la línea negada, lo que "normaliza" la lectura.

Tomando como referencia a estos circuitos se puede diseñar y construir un sin número de circuitos y aplicaciones utilizando el puerto paralelo y cualquier lenguaje de programación.

Algunas aplicaciones son: Control de motores paso a paso, sistemas de alarma, generador de señales digital, fuente digital, efectos de luces, efectos sonoros,... y muchos más.

2.8 DIAGRAMA DE BLOQUES DE UNA LÍNEA DE DATOS POR EL PUERTO PARALELO.

El puerto paralelo está formado por 17 líneas de señales y 8 líneas de tierra. Las líneas de señales están formadas por tres grupos:

- 4 Líneas de control
- 5 Líneas de estado
- 8 Líneas de datos

En el diseño original las **líneas de control** son usadas para la interface, control e intercambio de mensajes desde el PC a la impresora.

Las **líneas de estado** son usadas para intercambio de mensajes, indicadores de estado desde la impresora al PC (falta papel, impresora ocupada, error en la impresora).

Las **líneas de datos** suministran los datos de impresión del PC hacia la impresora y solamente en esa dirección. Las nuevas implementaciones del puerto permiten una comunicación bidireccional mediante estas líneas.

Cada una de estas líneas (control, estado, datos) puede ser referenciada de modo independiente mediante un registro.

Los registros del puerto paralelo

Cada registro del puerto paralelo es accesado mediante una dirección. El puerto paralelo tiene tres registros:

- Registro de datos
- Registro de estado
- Registro de control

En la tabla que se muestra a continuación se muestra la relación que existe entre las líneas físicas del conector del PC y los registros.

Tabla general del puerto paralelo

DB25	Señal	Registro	Tipo	Activo	Sentido
1	Control 0	C0-	Salida	Bajo	Invertido
2	Dato 0	D0	Salida	Alto	Directo
3	Dato 1	D1	Salida	Alto	Directo
4	Dato 2	D2	Salida	Alto	Directo
5	Dato 3	D3	Salida	Alto	Directo
6	Dato 4	D4	Salida	Alto	Directo

7	Dato 5	D5	Salida	Alto	Directo
8	Dato 6	D6	Salida	Alto	Directo
9	Dato 7	D7	Salida	Alto	Directo
10	Estado 6	S6+	Entrada	Alto	Directo
11	Estado 7	S7-	Entrada	Bajo	Invertido
12	Estado 5	S5+	Entrada	Alto	Directo
13	Estado 4	S4+	Entrada	Alto	Directo
14	Control 1	C1-	Salida	Bajo	Invertido
15	Estado 3	S3+	Entrada	Alto	Directo
16	Control 2	C2+	Salida	Alto	Directo
17	Control 3	C3-	Salida	Bajo	Invertido
18-25	Tierra				

Notas:

Un dato en alto es un 1, un dato en bajo es un 0

La entrada y salida son desde el punto de vista del PC

Esquema

El puerto paralelo esquemáticamente, se describe a continuación. Nótese la conexión al bus ISA en la parte izquierda y los registros en la parte derecha.

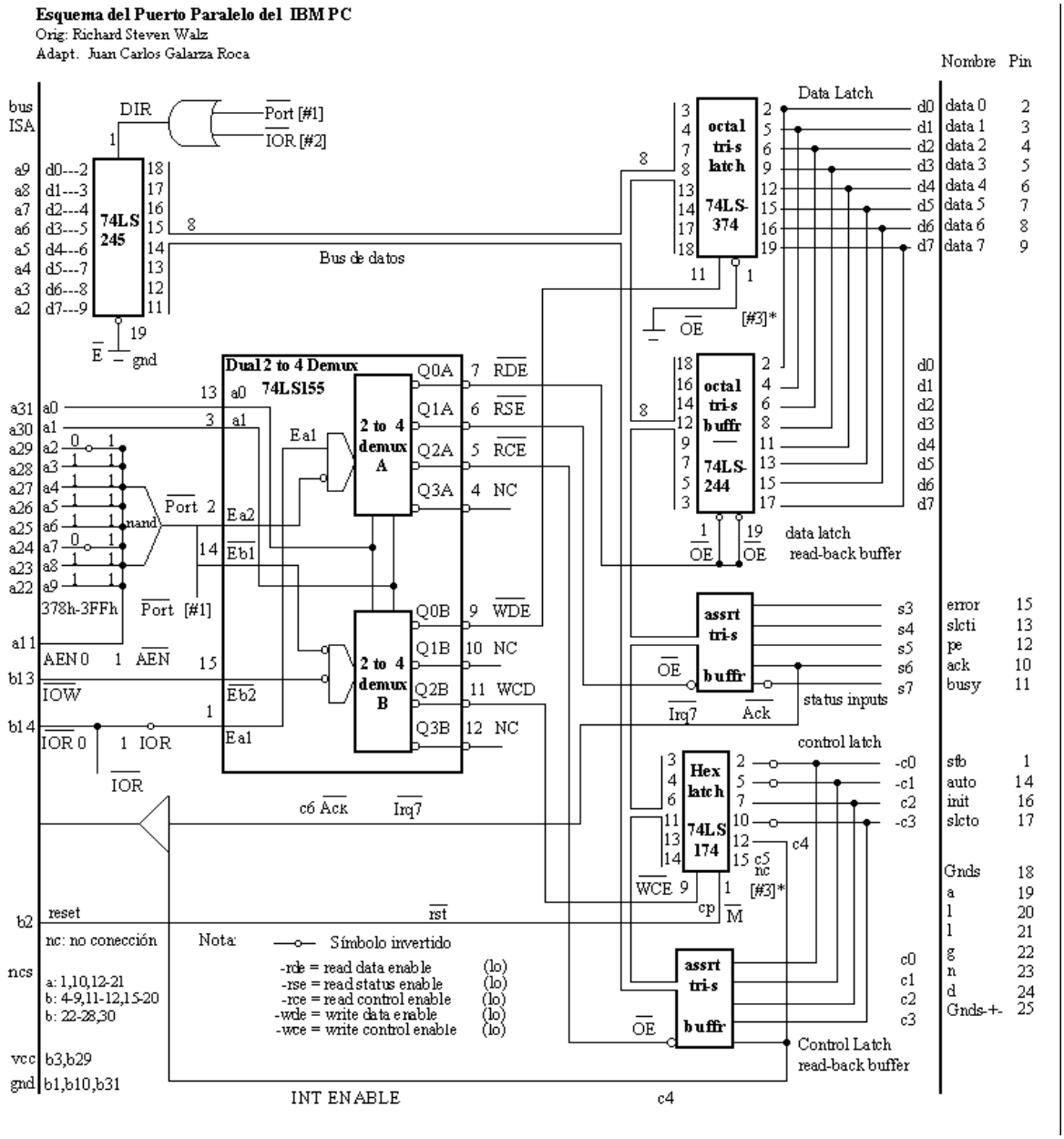


Figura 2.8 Esquema de un puerto paralelo.

Descripción de los componentes

El puerto paralelo originalmente estaba formado por los siguientes componentes:

- 1 **Latch** para manejar el registro de datos
- 1 **Buffer** para controlar la retroalimentación del registro de datos
- 1 **Buffer** para manejar el registro de estado

- 1 **Latch** para manejar el registro de control
- 1 **Buffer** para controlar la retroalimentación del registro de control
- 1 **Multiplexor** para direccionar los puertos en el bus ISA
- 1 **Driver bidireccional** para conectar con el bus ISA

Explicación del funcionamiento mediante el BIOS y el MS-DOS

IBM especificó direcciones base para el puerto paralelo estándar (dentro del espacio de direccionamiento de Entrada/Salida del 80x86). El adaptador de impresora podría usar la dirección base 3BCh, o más tarde 378h o 278h.

El BIOS (Basic Input Output System) de IBM crea en el momento de arranque o POST (Power On Seft Test) una tabla en el espacio de la memoria principal (RAM) para 4 direcciones base de puerto paralelo de impresora, estos se almacenan como 4 bytes empezando con la dirección de memoria 408h. Durante el arranque, el BIOS comprueba si hay puertos paralelos en las direcciones base 3BCh, 378h, y 278h, en ese orden, y almacena la dirección base de cualesquiera que hayan sido encontrados en posiciones consecutivas de la tabla. Las posiciones que no son usadas pueden estar en 0, o como algunos BIOS lo hacen, le colocan la dirección del primer puerto encontrado.

Algunos programas pueden ignorar esta tabla, pero esta es usada por lo menos por el propio BIOS (mediante la INT 17 de E/S de impresora) y por el MS-DOS.

El BIOS detecta estos puertos escribiendo AAh al registro de datos (en la dirección de E/S Base + 0), y luego si en el registro de datos se lee AAh. Significa que hay un puerto.

Normalmente la asignación de direcciones es como sigue:

Dirección	Nombre	Ubicación
3BCh	LPT1	Adaptador de impresión primario
378h	LPT2	Adaptador de impresión secundario

Las referencias a cada registro del puerto se realizan de la siguiente forma:

- Base (datos)=base+0
- Estado=base+1
- Control=base+2

Por ejemplo, si encontramos que la dirección base es 378h, entonces las direcciones del registro de datos, estado y control serán:

- Base (datos)=378h
- Estado=379h
- Control=37Ah

Cada una de ellas permite accesar a los siguientes bits (descritos en la tabla general):

- Base (datos)=D0, D1, D2, D3, D4, D5, D6, D7
- Estado=S3, S4, S5, S6, S7
- Control=C0, C1, C2, C3

CAPITITULO III

DIGITALIZACIÓN DE SEÑALES ANALÓGICAS EXTERNAS.

Para ello se describe el software de adquisición de datos.

SISTEMA DE ADQUISICIÓN DE DATOS A TRAVES DEL PUERTO PARALELO Y PLATAFORMA LABVIEW

Resumen: En el campo de la automatización electrónica y la instrumentación moderna es usual el empleo de soluciones apoyadas en ordenadores. A continuación se presenta una aplicación en la cual se realiza, de una manera sencilla y eficiente, un muestreo y digitalización de datos análogos empleando conversor análogo a digital, el puerto paralelo del PC y plataforma Labview.

Palabras clave: Interfaces, adquisición de datos, digitalización, conversor análogo a digital ADC, puerto paralelo, Labview, muestreo.

1. Introducción

En gran cantidad de aplicaciones, tanto a nivel académico, como a nivel industrial, es necesario registrar el valor de diferentes variables físicas que evolucionan en función del tiempo, para su posterior análisis y/o control.

En muchos casos, estas mediciones se realizan mediante medidores análogos, registrando de forma manual sus valores con respecto al tiempo. Sin duda, solo en

algunas pocas (muy pocas) aplicaciones, el procedimiento del operario con un reloj satisface los requerimientos de precisión tanto en tiempo, como en el valor de la variable registrada.

Es claro que en la gran mayoría de aplicaciones, es deseable tener un grado de precisión suficientemente alto,tanto en el tiempo entre muestra y muestra, así como en el valor registrado.

El objetivo de la presente aplicación es tener una herramienta para la adquisición de datos, que permita realizar un muestreo de un voltaje variable en el tiempo a intervalos de tiempo iguales y con la mayor precisión posible

2. Planteamiento del problema

En general el sistema se divide en tres bloques principales: sistema de conversión de datos análogos a datos digitales, sistema de interfaz a través del puerto paralelo, y sistema de control, visualización y registro de los datos en el computador.

Este planteamiento esta enfocado a una arquitectura por niveles que permitan el cambio de tecnología en cualquiera de los niveles, todo pensando en desarrollos posteriores.

Cada nivel tendrá funciones especificas dependiendo de las características de cada subsistema.

3. Diseño del sistema

Como ya se mostró, el sistema esta compuesto de tres partes básicas:

a) conversión de datos análogos a datos digitales.
b) Interfaz a través del puerto paralelo.
c) Control, visualización y registro de los datos en el computador.

A continuación se explica los requerimientos de cada parte y la solución propuesta en este desarrollo.

La mayoría de los transductores generan una señal de salida analógica. Por ejemplo un transductor de temperatura; si la temperatura varia de manera continua, la salida del transductor mostrara una variación continua y hay posiblemente una infinidad de números posibles para los valores de salida. El convertidor Analógico/Digital (convertidor A/D o ADC) es el dispositivo electrónico utilizado en los sistemas de adquisición de datos para convertir las señales analógicas de los transductores en el código digital utilizado por la computadora.

La representación digital del valor de salida del transductor es un código relacionado con la salida analógica del transductor, pero no describe exactamente la salida.

Por ejemplo, podría conectarse la salida de un transductor a un relevador que opera un foco; si el voltaje es menor o igual a 5volts el foco esta apagado, si el voltaje de salida del transductor es mayor a 5volts el foco se enciende. Los estados de prendido y apagado del foco son un código digital que representa la salida del transductor.

De manera similar la salida del transductor se podría representar por dos focos, por lo que existirían cuatro estados posibles para el par de focos:

	Foco 1	Foco 2
V<2.5	Apagado	Apagado
2.5≤V<5.0	Apagado	Encendido
5.0≤V<7.5	Encendido	Apagado
7.5≤V	Encendido	Encendido

Con dos focos la salida del transductor ahora se representa con mas precisión, en forma digital, que con un foco. Estos sistemas de focos representan formas primitivas de convertidores A/D. El sistema de un foco es un convertidor A/D de 1-bit y el sistema de dos focos es un convertidor A/D de 2-bits. Un bit tiene dos estados posibles encendido o apagado. Dentro de la computadora, se usan los flip-flops para representar tales estados.

En general, la salida de un convertidor A/D tiene 2^N valores posibles, donde N es el número de bits usado para representar la salida digital. El sistema de 1-bit tiene dos estados de salida posibles, 0 y 1, y el sistema de 2-bits tiene $2^2 = 4$ estados de salida posibles(00,01,10 y 11 en representación binaria). Los Sistemas de adquisición de datos computarizados normalmente utilizan convertidores A/D con 8-bits, de menos, donde el número de estados posibles es 2^8(igual a 256), estos estados se representan por números binarios con valores entre 00000000 y 11111111. Un ejemplo es 10000001,dado que existe una conversión directa entre números

binarios(base 2) y decimales(base 10), esta salida se establece como 129 en decimal. La representación física actual en la salida del ADC es, sin embargo, binaria.

Aunque la circuiteria de los ADC´s puede variar ampliamente, desde un punto de vista externo estos pueden ser descritos por tres características principales. La primera es el número de bits usado para representar la salida, entre más grande sea el número de bits mas grande será el número de estados posibles para la salida y la entrada analógica será representada con mayor precisión por la salida digital. La segunda característica es el rango de entrada; y la tercera es la velocidad de conversión, el tiempo que toma crear una salida digital después de que al dispositivo se le indica hacer la conversión.

El rango de entrada de un ADC es el rango de voltaje analógico de entrada sobre del cual el convertidor producirá una representación digital de salida. Los voltajes de entrada fuera de rango no producirán una representación digital significativa de la entrada. El rango de entrada de los ADC´s se clasifica como unipolar o bipolar. Un convertidor unipolar solo puede responder a entradas analógicas con el mismo signo (ejem. 0 a 5 volts o 0 a -10 volts), y uno bipolar puede convertir entradas positivas y negativas (valores típicos son ±5 y ±10 volts).

3.1 CONSIDERACIONES DEL HARDWARE. Se encuentran.

1.- PARÁMETROS ANÁLOGOS DE I/O.- Para configurar apropiadamente una tarjeta DAQ con entradas y salidas análogas, debe primero entenderse los efectos que; *resolución, rango y ganancia* tienen en la calidad de la señal digitalizada. Dependiendo del tipo de tarjeta que se va a utilizar, se puede configurar los parámetros mediante jumpers o software.

RESOLUCIÓN.- Nos referimos a resolución por el número de bits que el ADC utiliza para representar una señal análoga. Por ejemplo, un ADC de 3 bits divide el rango dentro de 8 divisiones. Un código digital binario entre 000 y 111 representa cada división. El ADC traslada cada medida de la señal análoga para una de esas divisiones. En la figura se indica la onda seno obtenida por 3 bits ADC. La señal digital no es una buena representación de la señal original porque la conversión tiene pocos niveles discretos para representar la variación de voltaje de la señal análoga. Pero incrementando la resolución a 16 bits, el número de divisiones se incrementa desde 8 a 65,536. Con lo cual se puede tener una mejor precisión en la representación de la onda analógica.

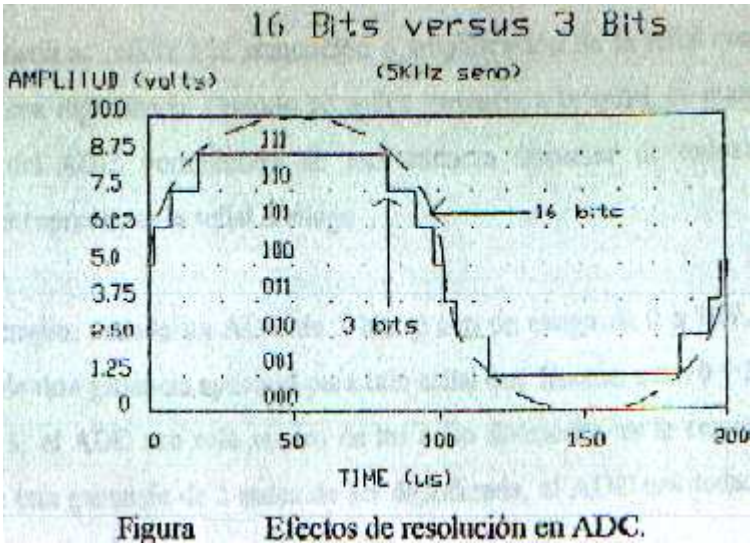


Figura 3.1 Efectos de resolución en ADC.

RANGO.- Los rangos son los niveles de voltaje máximo y mínimo que el ADC puede digitalizar. El ADC en muchas tarjetas DAQ tiene la característica de seleccionar los rangos, de esta manera tomar la más adecuada disponibilidad en base a la resolución. Por ejemplo, en la figura el ADC de 3 bits tiene 8 divisiones digitales dentro de un rango de 10V. Si seleccionamos el rango de –10V a +10V como indica la figura, el

mismo ADC ahora separa un rango de 20V dentro de ocho divisiones. El pequeño incremento de voltaje detectable, cambio de 1.25V a 2.5V, dando como resultado una menor precisión en la representación.

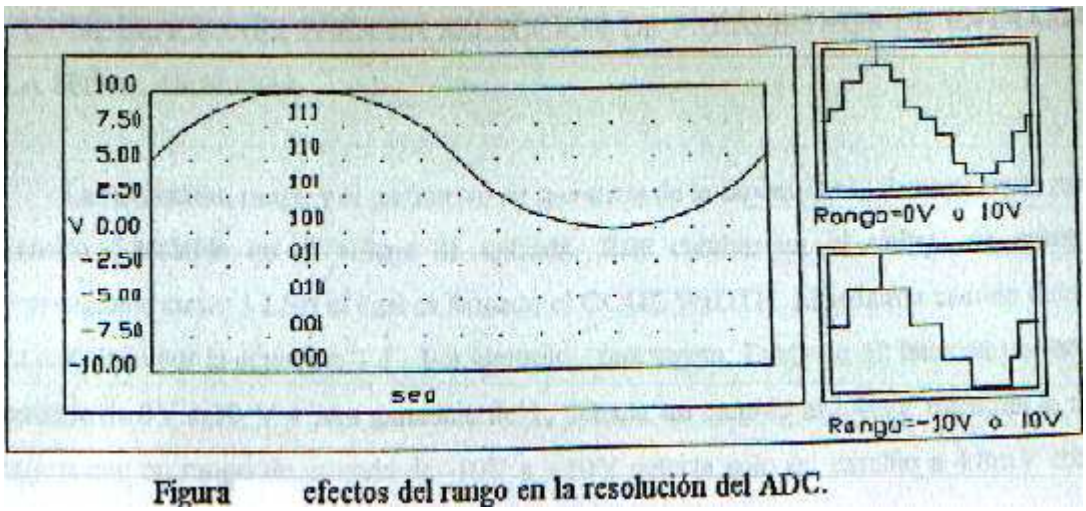


Figura efectos del rango en la resolución del ADC.

Figura 3.2 Efectos del rango en la resolución del ADC.

GANANCIA.- La ganancia se refiere a la atenuación o amplificación de la señal que puede ocurrir antes que la señal sea digitalizada. Cuando se aplica ganancia a la señal, se puede disminuir el rango de entrada del ADC, permitiendo de esta manera disponer de todos los niveles digitales posibles para representar la señal análoga.

Por ejemplo, usando un ADC de 3 bits y con un rango de 0 a 10V, la figura indica los efectos de una ganancia aplicada para una señal que fluctúa entre 0 y 5V.

Con una ganancia aplicada de 1, el ADC usa solo cuatro de las ocho divisiones en la conversión. Pero aplicando la señal con una ganancia de 2 antes de ser digitalizada, el ADC usa todas las 8 divisiones, y la representación digital es mucho más precisa.

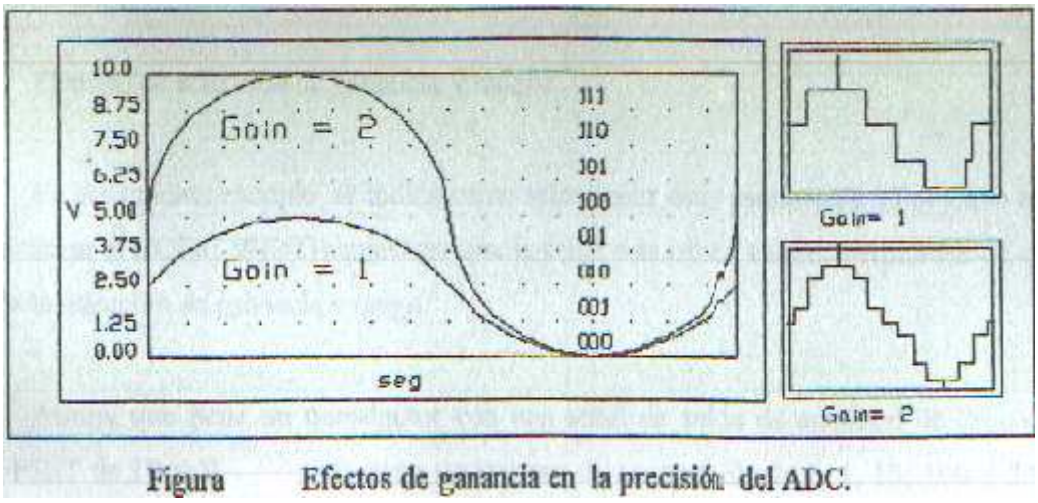


Figura 3.3 Efectos de ganancia en la precisión del ADC.

CONSIDERACIONES PARA LA SELECCIÓN DE PARÁMETROS DE ENTRADA DE LA SEÑAL ANÁLOGA.

La resolución, rango y el parámetro de ganancia de la tarjeta DAQ determinan el mínimo cambio detectable en el voltaje de entrada. Este cambio en el voltaje de entrada es representado como 1 LSB el cual es llamado el CODE WIDTH. El mínimo cambio detectable es calculado por la ecuación. Por ejemplo, una tarjeta DAQ de 12 bits con un rango de entrada 0V a 10V y una ganancia de 1, detecta un cambio a 2.4mV mientras la misma tarjeta con un rango de entrada de –10 a +10V detecta solo un cambio a 4.8 mV como se muestra en los cálculos.

$$\frac{\text{Rango}}{\text{Ganancia} \times 2^{\text{resolución}}} = \frac{10}{1 \times 2^{12}} = 2.5\ \text{mV}$$

$$\frac{20}{1 \times 2^{12}} = 4.8\ \text{mV}$$

Esto es importante para corregir el margen de rango de entrada de la tarjeta para la señal de entrada. Si los cambios de la señal de entrada son más pequeños que

la resolución de la tarjeta, se debería amplificar la señal. Por ejemplo, si es usada una ganancia de 10 en la tarjeta con un rango de 10V y una resolución de 12 bits la precisión se incrementa desde 2.4mV a 2.44 uV.

3.1.1. DETALLES DEL COMPUTADOR.- Dependiendo del tipo de tarjeta que disponga , se tiene que ubicar varios parámetros fijando swich y jumpers en la tarjeta, antes de instalar la tarjeta en el computador. Otros tipos de tarjetas DAQ como las PLUG and PLAY (PnP) fijan sus parámetros por medio de software. En este tema se discutirá varios parámetros de la tarjeta que se configuran en el hardware si la tarjeta no es PLUG and PLAY.

Tres son los parámetros que se debe ajustar en la tarjeta DAQ, que determinan como la tarjeta se puede comunicar con el computador. Esos parámetros son la dirección base I/O, el nivel de interrupción, y el canal de DMA.

1 EL DIRECCIONAMIENTO BASE I/O .- La tarjeta DAQ se comunica con el computador primeramente a través de registros. El software escribe hacia los registros de configuración en la tarjeta para configurar la tarjeta. El software lee los datos del registro en la tarjeta para obtener el estado de la tarjeta o una señal de medida. Los parámetros de direccionamiento base I/O determinan en que espacio de la memoria del computador los registros de la tarjeta residen.

2 NIVEL DE INTERRUPCION. – Otra forma de comunicar la tarjeta DAQ con el computador es a través del proceso de interrupción, las interrupciones son muy importantes en la operación de los computadores.

Ellos dan al procesador la habilidad de responder prontamente a los periféricos. Se puede pensar en un proceso de interrupción en una campana o un timbre. Si su puerta no tiene un timbre, debería ir periódicamente a la puerta para ver si algo a pasado en un tiempo en particular. Esto resulta muy ineficiente. Con un timbre, solo se necesita ir a la puerta cuando el timbre suena, y estaremos seguros de que algo esta allí esperando. Una tarjeta DAQ puede usar una interrupción como un timbre, para procesar la señal indicando que tiene datos esperando para ser leidos. A cada dispositivo que utiliza un proceso de interrupción se le debe asignar un nivel diferente de interrupción, caso contrario, si los dispositivos tienen los mismos niveles de interrupción pueden entrar en conflicto.

3 ACCESO DIRECTO A MEMORIA (DMA).

La tercera manera de comunicar la tarjeta DAQ hacia el computador es a través del ACCESO DIRECTO A MEMORIA (DMA). El DMA es un método de transferencia de datos, en el cual los datos son transferidos directamente desde los periféricos a la memoria del computador, no utilizando para ello el procesador. El DMA es usualmente requerido para realizar máxima velocidad de transferencia de datos, esto es muy útil para dispositivos que necesitan enviar datos a altas velocidades. A cada dispositivo que utiliza DMA debe asignarse un canal separado de DMA, caso contrario, entraran en conflicto con los dispositivos.

4 CONFIGURACIÓN DE LA TARJETA DAQ.

Dependiendo del computador utilizado, tres son los parámetros que son fijados en forma diferente.

PC/XT/AT Bus. Si su tarjeta DAQ no es PnP , jumpers y dip switches determinan el direccionamiento base I/O, el nivel de interrupción, y el canal(s) de DMA. Se debería verificar que el sistema del computador no tenga otro hardware con esos parámetros. Si se cambia los jumpers, se debe realizar el correspondiente cambio en el software de configuración.

5 CONFIGURACION DE LAS I/O ANÁLOGAS.

Las tarjetas DAQ tiene varios parámetros de configuración para las I/O análogas que controlan la operación del ADC Y DAC. Algunas tarjetas usan solo configuración por software para fijar los parámetros I/O, mientras otras pueden hacerlo por hardware. Una tarjeta que se puede configurar por software , se configura utilizando el software de configuración.

3.2 ACONDICIONAMIENTO DE SEÑALES ANALÓGICAS.

Tenemos a continuación.

3.2.1. AMPLIFICADORES ANALÓGICOS PARA LA INSTRUMENTACIÓN.

Los primeros amplificadores operacionales usaban el componente básico de su tiempo: la válvula de vacío. El uso generalizado de los AOs no comenzó realmente hasta los años 60, cuando empezaron a aplicarse las técnicas de estado sólido al diseño de circuitos amplificadores operacionales, fabricándose módulos que realizaban la circuitería interna del amplificador operacional mediante diseño discreto de estado sólido. Entonces, a mediados de los 60, se introdujeron los primeros amplificadores operacionales de circuito integrado. En unos pocos años los

amplificadores operacionales integrados se convirtieron en una herramienta estándar de diseño, abarcando aplicaciones mucho más allá del ámbito original de los computadores analógicos.

Con la posibilidad de producción en masa que las técnicas de fabricación de circuitos integrados proporcionan, los amplificadores operacionales integrados estuvieron disponibles en grandes cantidades, lo que, a su vez contribuyó a rebajar su coste. Hoy en día el precio de un amplificador operacional integrado de propósito general, con una ganancia de 100 dB, una tensión offset de entrada de 1 mV, una corriente de entrada de 100 nA. Y un ancho de banda de 1 MHz. es inferior a 1 euro.

El amplificador, que era un sistema formado antiguamente por muchos componentes discretos, ha evolucionado para convertirse en un componente discreto él mismo, una realidad que ha cambiado por completo el panorama del diseño de circuitos lineales.

Con componentes de ganancia altamente sofisticados disponibles al precio de los componentes pasivos, el diseño mediante componentes activos discretos se ha convertido en una pérdida de tiempo y de dinero para la mayoría de las aplicaciones dc y de baja frecuencia. Claramente, el amplificador operacional integrado ha redefinido las "reglas básicas" de los circuitos electrónicos acercando el diseño de circuitos al de sistemas. Lo que ahora debemos de hacer es a conocer bien los AOs, cómo funciona, cuáles son sus principios básicos y estudiar sus aplicaciones

3.2.2. CONFIGURACIONES BÁSICAS DE AMPLIFICADORES.

Los amplificadores operacionales se pueden conectar según dos circuitos amplificadores básicos: las configuraciones (1) ***inversora*** y (2) ***no inversora***. Casi

todos los demás circuitos con amplificadores operacionales están basados, de alguna forma, en estas dos configuraciones básicas. Además, existen variaciones estrechamente relacionadas de estos dos circuitos, más otro circuito básico que es una combinación de los dos primeros: el ***amplificador diferencial***.

3.3 ACONDICIONAMIENTO DE SEÑALES DIGITALES. Se hallan.

3.3.1. CONVERSORES DAC (DE DIGITAL A ANALÓGICO).

¿En que consiste un convertidor Digital/Analógico?

R. Un convertidor D/A. Es un dispositivo Electrónico que convierte una señal digital en una señal analógica.

Un convertidor D/A esta provisto de un arreglo de resistencias (R-2R), las cuales están conectadas a un amplificador sumador de tal manera que la resolución depende del numero de entradas digitales.

¿Cuántos pines tiene el DAC?

R. Este DAC tiene 20 pines; 8 para entrada de datos, 5 que funcionan como entradas de control, una entrada del voltaje de referencia, uno para la alimentación, 2 para conectarse a tierra, y además tiene tres pines los cuales van conectados a un amplificador operacional para convertir la corriente en voltaje de salida.

¿Por qué se requirió utilizar un Amplificador Operacional en esta práctica?

R. Para convertir la corriente de salida en voltaje.

¿Qué se observó en el osciloscopio? ¿y por qué?

R. En el osciloscopio se observó la salida del convertidor digital analógico en una forma escalonada este efecto observado es provocado porque el convertidor retiene el dato actual hasta que le llega otro dato nuevo, es decir, en este caso el DAC entrega a la salida lo que le entrega el convertidor analógico digital, pero como el convertidor analógico digital genera un tiempo de retardo, entonces el DAC tiene que retener el dato hasta que el ADC proporciona un dato nuevo. Esta operación de retardo es realizada por medio de unos registros internos del DAC.

¿Qué resolución tiene este convertidor?

R. La resolución esta directamente relacionada con el número de entradas digitales o bits en el DAC. Por ejemplo este DAC utilizado tiene 256 pasos y por lo tanto tiene una resolución de 8 bits.

Descripción general del DAC:

R. Este DAC esta hecho de Silicio y Cromo tiene una resolución de 8 bits, este dispositivo puede ser conectado directamente con el 8080, 8048, 8055, Z80 y otros populares microprocesadores.

El DAC utilizado contiene buffer doble los cuales permiten una salida de voltaje mientras se introduce la próxima palabra digital.

3.3.2 CONVERSIÓN DE DATOS ANÁLOGOS A DIGITAL.

Los requerimientos de este subsistema se acotan en dos aspectos básicos:

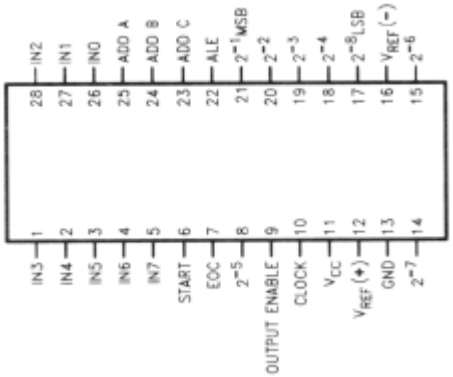
tiempo de muestreo, y regulación del tiempo entre muestra y muestra.

Como se desea un desarrollo de bajo costo y fácil implementación, se emplea un conversor análogo-digital comercial de la firma national semiconductor, el ADC 0808.

El circuito integrado ADC 0808 es un componente electrónico de tecnología CMOS con un conversor analógico a digital de 8 bits, un multiplexor análogo de ocho entradas y control lógico compatible con el funcionamiento de un microprocesador, característica que permite conectarlo directamente a un bus de expansión del computador. El conversor analógico digital de 8 bits emplea la técnica de aproximaciones sucesivas para realizar la conversión.

El diseño del ADC 0808 ofrece alta velocidad frente a su costo, gran desempeño, mínima dependencia de la temperatura, y mínimo consumo de potencia. La figura 3.4. se muestra un diagrama de ubicación de los pines.

Diagrama de la distribución de pines del ADC 0808



3.3.2.1 ADC DE RAMPA DIGITAL.- Tenemos.

Circuitos Convertidores A/D

ADC de rampa digital

Una de las versiones mas sencillas de ADC es la que emplea un contador binario como registro y permite que el reloj incremente el estado del contador un paso a la vez hasta que $V_{AX} >= V_A$. Este tipo de convertidor recibe el nombre de ADC de rampa digital debido a que la forma de onda de V_{AX} es una rampa (en realidad un escalera).

La figura 3.5 es el diagrama de un ADC de rampa digital . Como se observa, este contiene un contador, un DAC, un comparador analógico y una compuerta AND de control.

La salida del comparador también proporciona la señal de fin de conversión. Se supone que *Voltaje de referencia* es mayor la operación del mismo es la siguiente

Se aplico el pulso de Convertir para poner el contador en cero esto permito el paso de los pulsos del reloj por la compuerta AND. Cuando las entradas del DAC son todas cero, la salida de este es *Voltaje analógico* = 0. Dado que *Voltaje de referencia* > *Voltaje analógico* la salida es cero. A medida que cambia el contador, la salida del DAC, aumenta un paso a la vez, este proceso continua hasta que *Voltaje analógico* alcanza un paso que excede a *Voltaje de referencia* por una cantidad igual o mayor que Vt (por lo general de 10 a 100uV). En este momento la salida del comparador cambia de estado y no permite el flujo de pulsos hacia el contador, por el cual este deja de contar.

El contenido del contador es la representación digital del *Voltaje de Referencia.*

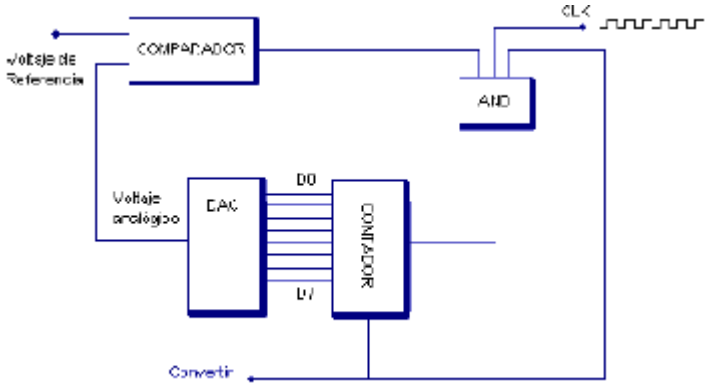


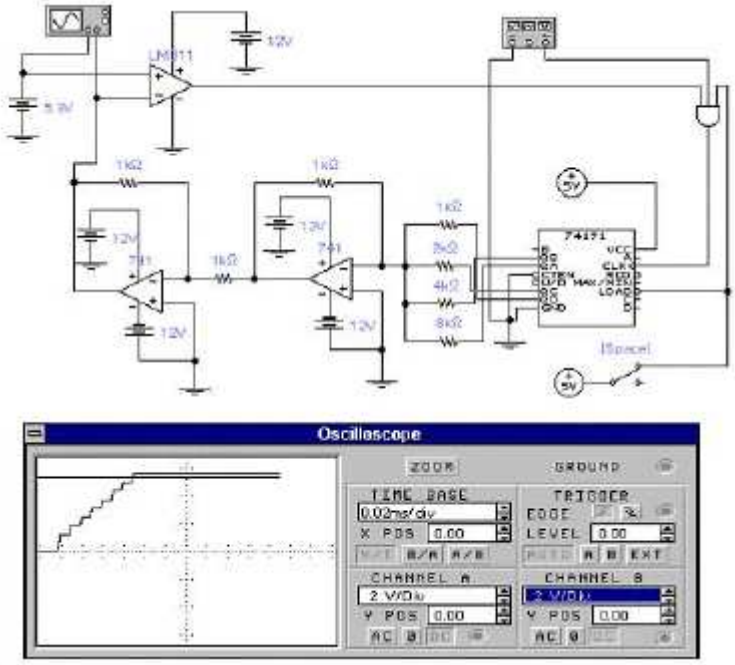
Figura Diagrama del convertidor A/D de rampa digital

Figura 3.5 Diagrama del convertidor A/D de rampa digital.

DESARROLLO Simulación e Implementación

Una vez establecida la teoría el paso siguiente es la simulación de un circuito como el mostrado en la figura . Con esto nos damos cuenta si el circuito próximo a implementar cumplirá con las características que buscamos.

El circuito simulado se muestra en la figura , en la cual se puede apreciar que consta con todos los elementos presentados en el diagrama es decir DAC, contador y comparador.



3.3.2.3 EL ADC 0808.- Analógico digital de 8 bits.

Manejo del conversor analógico digital ADC 0808.

En la figura se muestra el diagrama del circuito de conversión análogo a digital, la señal de clock (reloj) es generada mediante un circuito integrado LM555, el cual se configura como oscilador con una frecuencia aproximada a los 180 KHz. Si se desea, este oscilador puede ser reemplazado por uno de mayor frecuencia, pues el ADC 0808 puede tolerar frecuencias de reloj de hasta 400KHz.

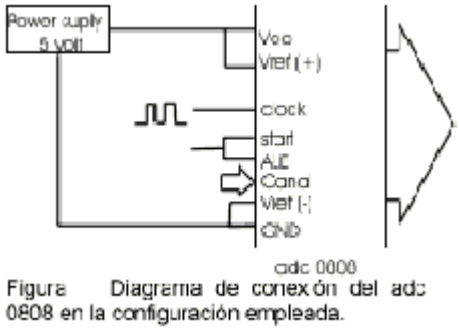


Figura Diagrama de conexión del adc 0808 en la configuración empleada.

Figura 3.6 Diagrama de conexión del adc 0808 en la configuración empleada.

Las señales de start (inicio) y ALE (habilitación de canal) se emplean ligadas como un solo pin de control que selecciona el canal del conversor que se emplea y, al mismo tiempo, le indica al circuito integrado que inicie el proceso de conversión.

Es importante tener en cuenta que el proceso de conversión toma de ocho a doce ciclos de reloj, lo que significa que el tiempo mínimo de muestreo debe ser mayor a 0.2 milisegundos, es decir, la frecuencia máxima de muestreo posible con el adc en la configuración descrita y reloj a 110 KHz es de 5 KHz.

Además de estas consideraciones el ADC posee un multiplexor que permite realizar el muestreo de diferentes variables con un solo equipo, pero no de forma estrictamente simultanea, pues si se realiza el muestreo en tres canales “a la ves”, el circuito integrado realizara el muestreo del primer canal, luego del segundo y finalmente el tercero, es decir solo es posible realizar un muestreo en cada instante.

Esto implica que si se requiere emplear dos canales del dispositivo con la frecuencia de reloj empleada, la máxima rata de muestreo es de unas 2500 muestras por segundo (se reduce a la mitad respecto a las mismas condiciones con solo un canal).

Es importante que la señal de reloj suministrada al dispositivo sea lo mas pura posible, pues cualquier oscilación puede ser tomada como un pulso generando errores en el funcionamiento del dispositivo.

3.3.2.3 CARACTERÍSTICAS DEL CONVERTIDOR ANALÓGICO A DIGITAL

A la salida digital de un convertidor analógico a digital ideal de 4 bits se gráfica en función del voltaje de entrada analógica. .

De manera análoga a lo que ocurre con los convertidores digital a analógico, la resolución de un convertidor analógico a digital se define de dos maneras. Primero, es el número máximo de códigos de salida digital.

Esta expresión de la resolución del convertidor es la misma que en el caso del convertidor analógico y se repite aquí :

resolución = 2^n

La resolución también se define como la razón de cambio del valor en el voltaje de entrada, V_i, que se necesita para cambiar en 1 LSB la salida digital.

Si se conoce el valor del voltaje de entrada a escala completa, V_{iFS}, que se requiere para producir una salida digital de todos los unos, es posible calcular la resolución mediante: resolución = $V_{IFS\ /\ -1}$

En su forma más simple, la ecuación de entrada-salida de un convertidor digital a analógico está dada por :

Código de salida digital = equivalente binario de D

donde D es igual al valor decimal de la salida digital ; o sea, D es igual al número de bits menos significativos en la salida digital y D se calcula a partir de

D = Vi/resolución

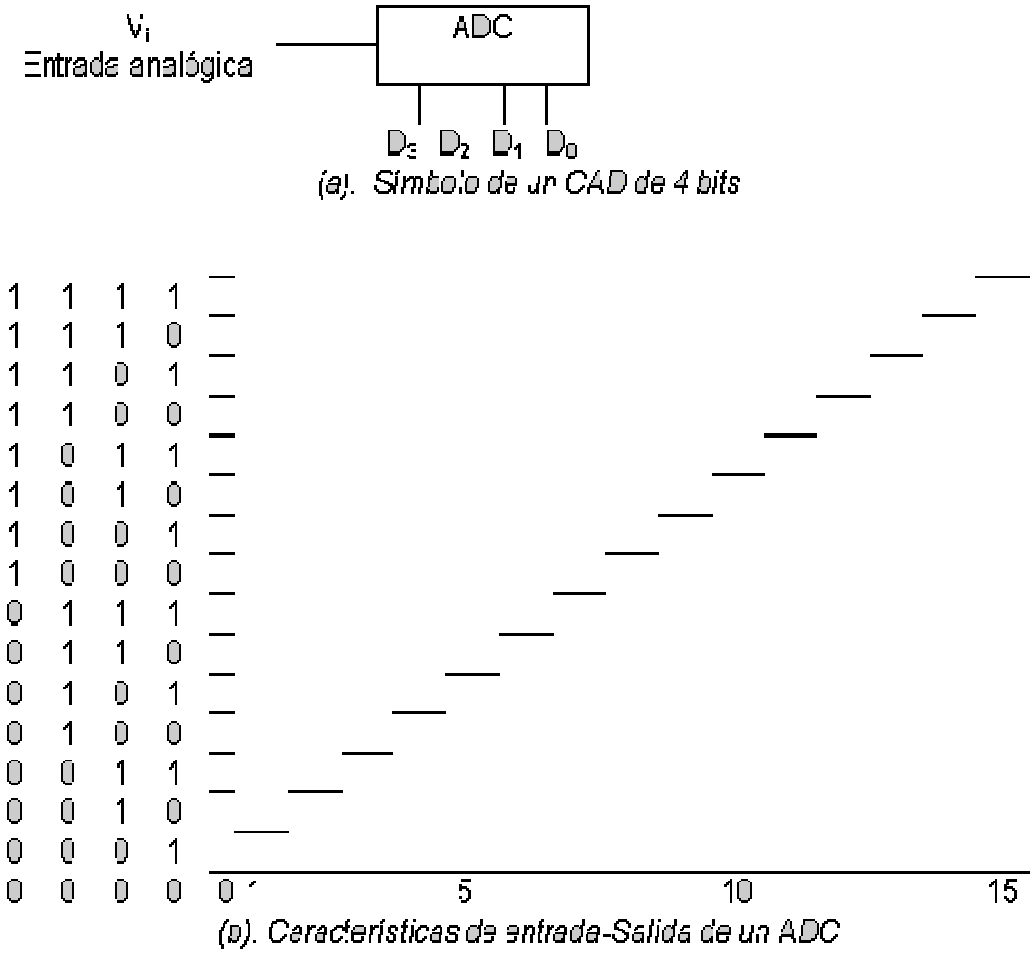


Figura 3.7 Características de entrada Salida de un ADC.

CAPITULO IV.

CONDICIONES PARA UN DISEÑO DE ADQUISICION DE DATOS (DAQ) .

Uno de los mas importantes parámetros de un sistema de entrada análoga es la velocidad cual la tarjeta muestrea una señal. Para determinar una apropiada velocidad de muestreo, se debería conocer el máximo rango de frecuencia de la señal que se necesita muestrear, la precisión que se requiere desde sus datos, algún ruido en el sistema que puede afectar a la señal análoga, y la compatibilidad de la tarjeta DAQ.

4.1 ENTORNO DE PROGRAMACIÓN GRAFICA.

La Programación Grafica surgió como resultado del caos producido cuando los lenguajes tradicionales se enfrentaron a los grandes problemas de software. Puesto que a medida que se van desarrollando los lenguajes, se van desarrollando también las posibilidades de resolver problemas cada vez más complejos.

La programación gráfica es una nueva forma o técnica de programación que se utiliza para desarrollar programas más eficientes y con gran fiabilidad.

4.1.1 VENTAJAS Y DESVENTAJAS DE LA PROGRAMACIÓN GRAFICA. Encontramos en la programación grafica y tradicional.

EN LA PROGRAMACIÓN GRÁFICA.- Los programas se dividen en entidades independientes que se relacionan con otras partes del programa, en donde cada entidad se considera un “gráfico” que contiene datos (sus características) y

procedimientos (su comportamiento). Es decir, un “grafico” es una entidad que contiene datos y los procedimientos que operan sobre esos datos. A los elementos de un grafico se les conoce como “miembros”, los procedimientos que operan sobre los gráficos se denominan “métodos” , y los datos se denominan “datos miembro”. Entonces , un programa consta de un número de gráficos, los cuales se comunican entre si mediante “métodos” a través del envío de “mensajes” que son acciones que debe ejecutar el gráfico.

EN LA PROGRAMACIÓN TRADICIONAL.- los programas constan de procedimientos y datos: donde cada procedimiento actúa como una “caja negra” que realiza una tarea específica sin preocuparse de lo que hace internamente. Esta técnica de programación permite desarrollar programas con empaquetamiento de código en procedimientos que los vuelven transportables y modulares: aunque los datos se tratan separadamente de los procedimientos, porque los datos utilizados son frecuentemente globales o se pasan en los parámetros de los procedimientos.

4.2 LABVIEW Y ADQUISICIÓN DE DATOS EN PUERTOS PARALELOS. Las funciones son.

LAS FUNCIONES IN PORT Y OUT PORT

Como su nombre lo indica, estas funciones o VI’s son las encargadas de recibir y enviar datos a través de los puertos; se encuentran en la paleta de funciones del diagrama, en la opción Advanced, en el recuadro de Port I/O y se distinguen por los iconos mostrados en la figura .

Figura Iconos de los VI's inport y outport.

Figura 4.1 Iconos de los VI's inport y outport.

La función InPort posee dos parámetros de entrada: La dirección de registro, que especifica la posición del registro empleado, por lo tanto es un dato numérico. El segundo parámetro de entrada consiste en un dato booleano que indica si el dato recibido desde la posición indicada es un byte (falso) o una palabra, es decir 2 bytes (verdadero). La salida de esta función consiste en un dato de tipo numérico.

La función OutPort tiene tres parámetros de entrada: La dirección de registro, que especifica la posición del registro empleado para enviar la información. El segundo parámetro de entrada consiste en un dato booleano que indica si el dato enviado a la posición indicada es un byte (falso) o una palabra (verdadero). El tercer parámetro es el dato que se va a enviar.

TRATAMIENTO DE DATOS

El VI de control de adc y puerto paralelo contiene los algoritmos necesarios para realizar la conversión del dato análogo a digital mediante el adc 0808 y la posterior adquisición del dato a través del puerto paralelo. Estos algoritmos se organizan en forma de secuencia de la siguiente forma:

- Envío de la señal de inicio de conversión.
- Tiempo de espera
- Recepción del dato a través del puerto paralelo.

El envío de la señal de inicio consiste en generar un pulso a través de un pin del puerto paralelo. En la aplicación diseñada esta operación toma 3 pasos de una secuencia con un total de cinco.

En cada uno se realiza la operación de salida de puerto, primero un cero luego un uno y de nuevo un cero, así se tendrá el pulso de inicio de la conversión.

En el paso cuatro de la secuencia se tiene un tiempo de espera de un milisegundo dado por la función de esperar mostrada en la figura .

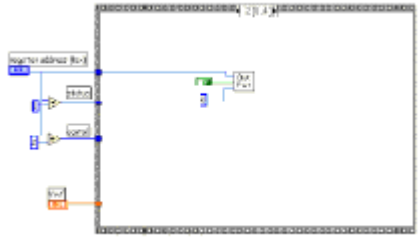


Figura . Parte de la secuencia.

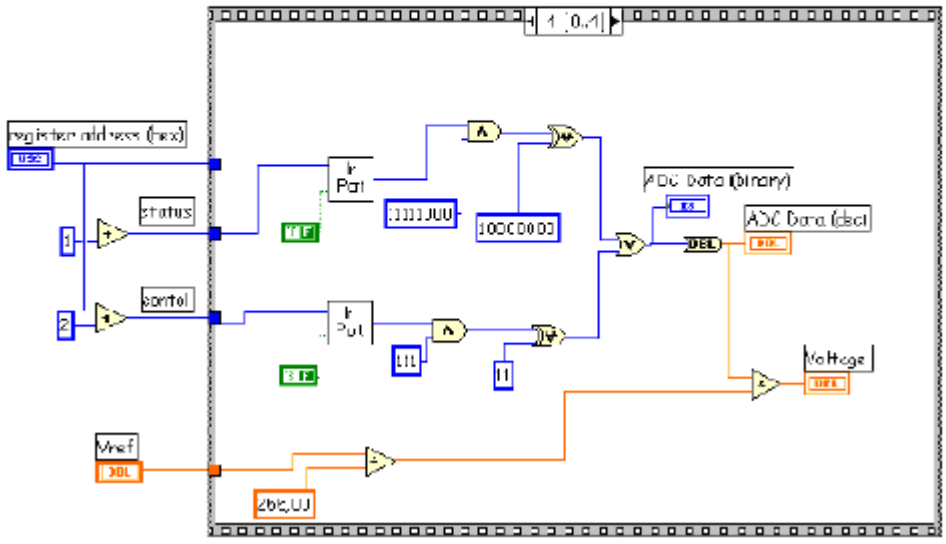
Figura . Función "esperar"

Figura 4.2 Función "esperar"

Para acondicionar los datos recibidos a través de los registros de estado y control del puerto paralelo es necesario aplicar mascaras realizando la operación AND del dato recibido por cada registro.

Para el registro de estado la mascara es el dato 11111000 binario, mientras que para el dato recibido través del registro de control es 00000111. Luego de esta operación resta invertir los bits que sea necesario y agrupar el dato, como puede apreciarse en la figura 4.3 .

Figura . Adquisición y acondicionamiento del dato proveniente del conversor analógico a digital a través del puerto paralelo



Por ultimo se incluye el voltaje de referencia (que debe coincidir con el Vref del ADC) para lograr finalmente un valor del voltaje registrado mediante el ADC.

4.2.1 USO DE LabVIEW

En este apartado se discuten los aspectos necesarios para familiarizarse con el uso de LabVIEW, incluyendo las ventanas Panel y Diagram, menús de LabVIEW y la ventana de jerarquía.

Asimismo se discuten otros aspectos necesarios como el uso de los modos **edit** y **run;** creación de objetos; herramientas y obtención de ayuda.

4.2.2 VENTANAS PANEL Y DIAGRAM

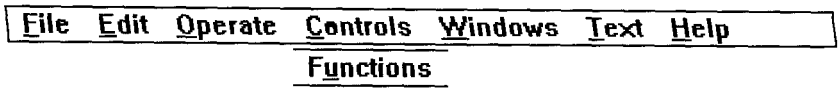
Cada VI tiene dos ventanas separadas, pero relacionadas entre sí. La ventana **Panel** contiene el panel frontal de nuestro Vi. La ventana **Diagram** es aquella en la cual se construye el diagrama de bloques. Se puede conmutar entre ambas pantallas con el

comando **Show Panel/Show Diagram** (Mostrar Panel/Mostrar Diagrama) de¡ menú **Windows** (Ventanas). Usando los comandos **Tile** (literalmente "baldosas"; podemos traducir por Parcelas), dentro de ese mismo menú, podemos posicionar las ventanas **Panel** y **Diagram** una al lado de la otra o una encima de la otra.

4.2.3 MENÚS DE LABVIEW

La programación en LabVIEW obliga a utilizar con frecuencia los diferentes menús. La barra de menús de la parte superior de la ventana de un Vi contiene diversos menús **pull-down** (desplegables). Cuando hacemos clic sobre un ítem o elemento de esta barra, aparece un menú por debajo de ella. Dicho menú contiene elementos comunes a otras aplicaciones Windows, como **Open** (Abrir), **Save** (Guardar) y **Paste** (Pegar), y muchas otras particulares de LabVIEW.

La siguiente figura muestra la barra de menús para la versión 3.1 cuando la ventana **Panel** está activa. El menú **Functions** reemplaza al **Controis** cuando la ventana **Diagram** está abierta.



File (Archivo) Sus opciones se usan básicamente para abrir, cerrar, guardar imprimir Vis.

Edit (Edición) Se usa principalmente para organizar el panel frontal y el diagrama de bloques y establecer nuestras preferencias.

Operate (Función) Sus comandos sirven para ejecutar el Vi.

Controls (Controles) Con este menú, podemos añadir controles e indicadores al panel frontal. Cada opción dentro de este menú visualiza una paleta con los controles e indicadores para esa opción. El menú **Controis** sólo está disponible cuando la ventana **Panel** está activa.

Functions (Funciones) Construimos el diagrama de bloques con este menú. Cada opción visualiza una paleta con sus ¡conos disponibles. El menú

Functions sólo está disponible cuando la ventana **Diagram** está activa.

Windows (Ventanas) Se usa para situar rápidamente las ventanas abiertas y para abrir ventanas de los diferentes subVIs.

Text (Texto) Se utiliza para cambiar la fuente, estilo y color del texto.

Heip (Ayuda) Presenta ayuda sobre los diferentes iconos y otros aspectos de LabVIEW.

File (Archivo) Misma función.

Edit (Edición) Misma función.

Operate (Función) Presenta nuevas opciones como pueden ser la impresión cuando

acaba la ejecución.

Project (Proyecto) Presenta los niveles de jerarquía, los subvis que lo integran, los que están sin abrir, busca Vis, etc.

Windows (Ventanas) Se utiliza básicamente para mostrar **(Show)** ventanas, como pueden ser las de información, historia¡, controles/funciones, herramientas, portapapeles, etc.

Help (Ayuda) Misma función.

El menú de LabVIEW que utilizaremos con más frecuencia es el menú **pop-up** (emergente) de objetos, al cual accedemos situando el cursor sobre el objeto en cuestión y pulsando el botón derecho del ratón. Si la pulsación se hace sobre un espacio vacío, el menú que se obtendrá vendrá en función de la herramienta seleccionada.

4.2.4 ADQUISICIÓN DE DATOS.- Un desarrollo en LabView se basa en el empleo de los "VI" o instrumentos virtuales, que en un sentido practico consiste en un objeto con entradas y salidas, y una función especificada dentro de este. La estructura de este lenguaje de programación denominado G permite llamar o incluir un VI dentro de otro, permitiendo jerarquizar los VI's.

La creación de un VI dentro de Labview supone dos partes: el diseño del panel frontal, en donde se encuentran las entradas y salidas (controles e indicadores) del VI,

y el diagrama de flujo de datos, en donde se configura el funcionamiento del VI; en un sentido practico es el algoritmo que define el comportamiento de la aplicación.

4.2.5 ANALISIS DE DATOS.- Para acceder a la paleta de controles dentro del panel de control basta con oprimir el botón derecho del ratón teniendo el puntero sobre la ventana correspondiente a este, de forma similar se logra el acceso a la paleta de funciones dentro del diagrama de flujo de datos (llamado simplemente diagrama).

LabView permite manipular distintas estructuras de datos, como boléanos (binarios), numéricos, cadenas, arreglos y clusters. Los clusters son una poderosa estructura que permite realizar arreglos compuestos por distintos tipos de datos. Dentro del Diagrama la estructura del dato se distingue por el color que toma el lazo correspondiente, por ejemplo, los boléanos se distinguen por el color verde

4.2.6 VISUALIZACIÓN DE DATOS.- Como ya se menciono con anterioridad, National Instruments LabView es la plataforma sobre la cual se desarrollo el software de esta aplicación. LabView es un software de programación grafica, obviamente enfocada al manejo de objetos, y desarrollado pensando en sistemas de instrumentación y control, lo que lo hace de gran utilidad para la adquisición y análisis de los datos.

Por ultimo es importante tener en cuenta que en el ambiente de programación G (LabView) el puntero del ratón cumple diversas funciones, distinguiéndose cada una por la figura que toma el puntero, por ejemplo si el cursor tiene forma de carrete de hilo funciona como herramienta de conexión.

4.2.7 CREACIÓN DE INSTRUMENTOS VIRTUALES. Tenemos.

Uso de los modos EDIT (Edición) y RUN (Ejecución)

Podernos crear o cambiar un VI cuando éste está en el modo **Edit.** En él, las

herramientas de edición se habilitan en la paleta de¡ modo **Edit,** por debajo de la barra de¡ menú de ventana, como se indica a continuación:

Cuando estamos listos para probar nuestro VI, hacemos clic sobre el botón de modo Q o seleccionamos **Change to Run Mode** (Cambio al Modo de Ejecución) desde el menú **Operate.** Haciendo esto compilamos el Vi y lo ponemos en el modo **Run.** En este punto podemos disponer de las opciones de depuración, ejecución del VI, diferentes modos de ejecución, impresión de datos, etc.

Si lo que queremos es ejecutar el Vi desde el modo **Edit** sin pasar al modo **Run,** hemos de hacer clic sobre la flecha de ejecución. Si fuese necesario, LabVIEW compilaría primero el VI, después conmuta al modo **Run,** ejecuta el Vi y vuelve al modo **Edit** una vez que el Vi se ha ejecutado.

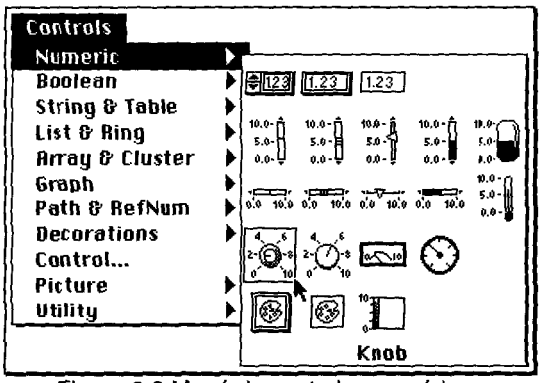
Éste es uno de los puntos que ha sufrido una mayor modificación en la versión Los iconos correspondientes a estos modos se indican a continuación:

Se puede observar que es aquí donde aparece el tratamiento de los diferentes tipos de letras y la alineación y distribución de objetos. Así mismo vemos que no aparece ninguna herramienta.

Otro aspecto a destacar es el botón **Pause** (pausa) IM. Al hacer clic en él se para la ejecución del VI y vamos al diagrama de bloques, parpadeando la siguiente secuencia que se ejecutará.

4.2.7.1 CREACIÓN DE OBJETOS

Para elaborar el panel frontal hemos de situar sobre él los objetos deseados mediante su selección desde el menú **Controls.** Creamos objetos sobre el diagrama de bloques seleccionándolos desde el menú **Functions.** Por ejemplo, si queremos crear un knob o botón rotatorio sobre el panel frontal, primero 'hemos de seleccionarlo desde la

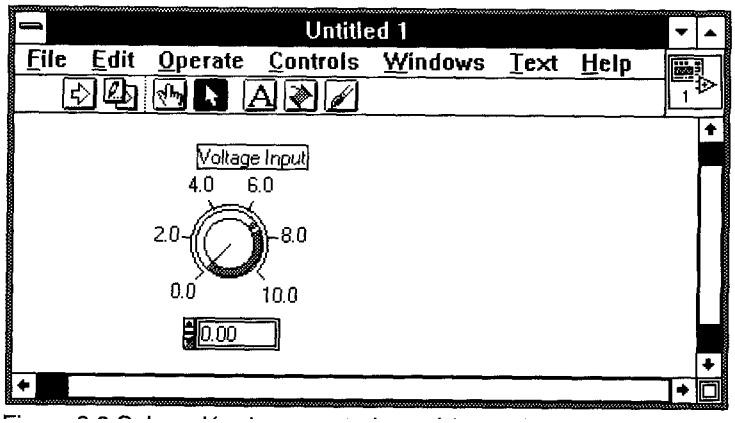


paleta **Numeric** (Numérico) del menú **Controls,** como se indica en la siguiente figura. El objeto aparecerá en la ventana **Panel** con un rectángulo negro o gris que representa una etiqueta de identificación o **Label.** Si queremos usarla en ese mismo momento, introduciremos el texto desde el teclado. Después de haberío hecho, cualquiera de las siguientes acciones completa la entrada:

- Pulsar < Shift + Enter >
- Pulsar < Enter > del **teclado numérico**.

· Clic sobre el botón Enter en la paleta de herramientas.

· Clic fuera de la etiqueta.

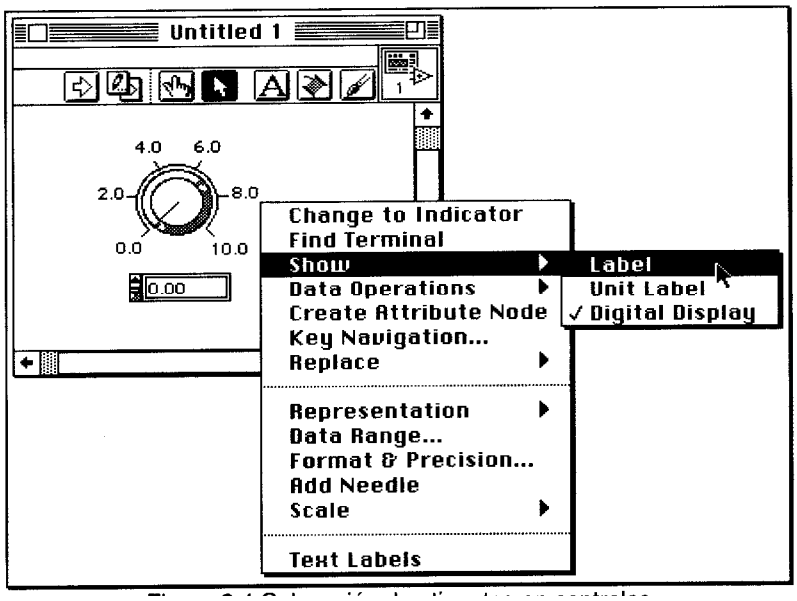


Cuando creamos un objeto sobre el panel frontal, al mismo tiempo se crea el terminal correspondiente sobre el diagrama de bloques. Este terminal se usa tanto para leer datos desde un control como para enviarlos a un indicador.

Si se selecciona **Show Diagrarn** (Mostrar Diagrama) desde el menú **Windows,** podremos ver el diagrama correspondiente al panel frontal. Este diagrama contendrá terminales para todos los controles e indicadores del panel frontal.

Todos los objetos en LabVIEW tienen asociados menús **pop-up,** los cuales podemos obtener pulsando el botón derecho del ratón sobre dicho objeto. Mediante la selección de sus diferentes opciones podremos actuar sobre determinados parámetros, como el aspecto o comportamiento de ese objeto.

Por ejemplo, si no hubiésemos introducido texto en la etiqueta del control anterior, ésta habría desaparecido al hacer clic en cualquier otro lado. Para volver a visualizarla tendríamos que obtener el menú **pop-up** de ese control y seleccionar Label del menú **Show** (figura).



Los menús **Controis** y **Functions** se hallan como ventanas flotantes que podemos tener visibles o no. Si no lo están, utilizaremos la opción **Show Controls Palette** (Mostrar paleta de controles) o **Show Functions Palette** (Mostrar paleta de funciones) del menú **Windows.**

Otra opción es hacer clic con el botón derecho de¡ ratón en cualquier área libre de la pantalla: Aparecerá el menú **Controls** o **Functions** según estemos en la ventana **Panel** o **Diagrarn,** respectivamente.

4.2.7.2 HERRAMIENTAS DE LabVIEW

Una herramienta es un modo de funcionamiento especial del ratón. Las usamos para llevar a cabo funciones específicas de edición o ejecución.

- La herramienta **Operating** (Funcionamiento) maneja los controles del panel frontal (y los indicadores en el modo **Edit).** Es la única herramienta disponible en el modo **Run.**
- La herramienta **Positioning 1** (Situación) selecciona, mueve y redimensiona objetos.
- La herramienta **Labeling** (Etiquetado) crea y edita textos.
- La herramienta **Wiring** (Cableado) enlaza objetos del diagrama de bloques y asigna a los terminales del conector del Vi los controles e indicadores del panel frontal.
- La herramienta **Coloring** (Coloración) colorea diversos objetos y los fondos.

Se puede cambiar de herramienta haciendo lo siguiente:

·Clic sobre el icono de la herramienta que queremos.

·Usando la tecla TAB para seleccionar la siguiente herramienta.

·Pulsando la tecla SPACE para cambiar entre la herramienta **Operatíng** y **Positioning** cuando la ventana **Panel** está activa, y entre las herramientas **Wiring** y **Positioning** cuando la ventana **Diagram** es la activa.

- **Operate Value** (Valor Operativo) H. Misma función que **Operating.**
- **Position/Size/Select** (SituaciónTamaño / Selección') M. Realiza la misma función que **Positioninq.**
- **Edit Text** (Edición de Texto) Misma función que **Labeling.**
- **Connect Wire** (Conexión de Cables) @. Misma función que **Wiring.**
- **Object Popup** (Menú pop-up del objeto) M. Función nueva. Despliega el menú pop-up asociado al objeto. Tiene el mismo efecto que si pulsamos el botón derecho del ratón sobre el objeto.

- **Scroll Window** (Desplazamiento de la pantalla) IJ. Función nueva. Desplaza la pantalla en la dirección que deseemos para ver posibles zonas ocultas.
- **Set/Clear Breakpoint** (Establecer/Quitar puntos de ruptura) ál. Función nueva. Permite poner tantos puntos de ruptura como deseemos a lo largo del diagrama de bloques. Cuando durante la ejecución se llega a uno de ellos, LabVIEW conmuta automáticamente al diagrama de bloques. Usamos esta misma herramienta para quitar los puntos.
- **Probe Data** (Sonda de datos).

4.2.7.3 TIPOS DE DATOS EN LABVIEW. CONTROLES E INDICADORES

LabVIEW ofrece una gran variedad de tipos de datos con los que podemos trabajar respondiendo a las necesidades reales con las que nos encontraremos. Uno de los aspectos más significativos de LabVIEW es la diferenciación que efectúa en el diagrama de bloques entre los diferentes tipos de controles o indicadores, basada en que cada uno de ellos tiene un color propio.

De esta manera, y como consecuencia de una memorización o asimilación práctica, nos será muy fácil identificarlos y reconocer inmediatamente si estamos trabajando con el tipo de datos adecuado. Distinguimos los siguientes tipos, los cuales pueden funcionar tanto como controles como indicadores (entre paréntesis queda reflejado el color con el que queda representado en el diagrama de bloques):

Boolean (verde claro)

Los tipos de datos booleanos son enteros de 16 bits. El bit más significativo contiene el valor Booleano. Si el bit 15 se pone a 1, entonces el valor del control o indicador es **true** (verdadero); por el contrario, si este bit 15 vale 0, el valor de la variable booleana será **false** (falso).

Numéricos: Hay diferentes tipos

Extended (naranja) Según el modelo de ordenador que estemos utilizando los números de coma flotante con precisión extendida presentan el siguiente formato:

Macintosh:96 bits (formato precisión extendida MC68881 - MC68882)
Windows: 80 bits (formato precisión extendida 80287)

Sun: Formato 128 bits

HP-UX: Son almacenados como los números en coma flotante de doble precisión.

Double (naranja) Los números en coma flotante de doble precisión cumplen con el formato de doble precisión IEEE de 64 bits. Es el valor por defecto de LabVIEW.

Single (naranja) Los números en coma flotante de precisión simple cumplen con el formato de precisión simple IEEE de 32 bits.

Long lnteger (azul) Los números enteros largos tienen un formato de 32 bits, con o sin signo.

Word lnteger (azul) Estos números tienen un formato de 16 bits, con o sin signo.

Byte lnteger (azul) Tienen un formato de 8 bits, con o sin signo.

Unsigned Long (azul) Entero largo sin signo.

Unsigned Word (azul) Palabra sin signo.

Unsigned Byte (azul) Byte sin signo.

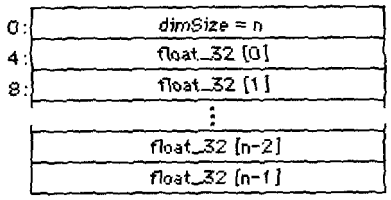
Complex Extended (naranja)Número complejo con precisión extendida.

Complex Double (naranja) Complejo con precisión doble.

Complex Single (naranja) Complejo con precisión simple.

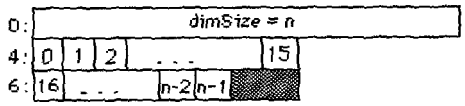
Arrays (depende del tipo de datos que contenga)

LabVIEW almacena el tamaño de cada dimensión de un array como **long integer** seguido por el dato. El ejemplo que sigue muestra un array unidimensional con números en coma flotante de precisión simple. Los números decimales a la izquierda

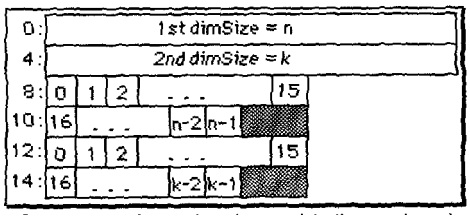


presentan el desplazamiento donde empieza cada array en la posición de memoria.

Los arrays booleanos se almacenan de manera diferente a los booleanos escalares. Estos arrays se almacenan como bits empaquetados. El tamaño de la dimensión viene dado en bits en lugar de bytes. El bit 0 se guarda en la posición más alta de memoria (215), y el bit 15 en la posición más baja (20).

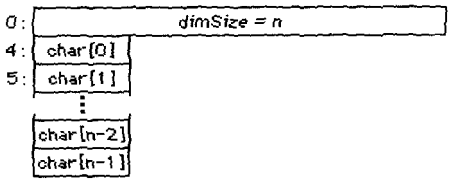


La figura muestra un ejemplo de un array booleano bi-dimensional. El elemento



0 de cada dimensión se almacena en una nueva palabra entera ignorándose los bits sin usar de las dimensiones previas.

Strings (rosa) LabVIEW almacena los strings como si fueran un array uni-



dimensional de bytes enteros (caracteres de 8 bits).

Handies Un handie es un puntero que apunta a un bloque de memoria relocalizable. Un handie sólo apunta a datos definidos por el usuario. LabVIEW no reconoce qué es lo que hay en ese bloque de memoria. Es especialmente útil para

pasar un bloque de datos por referencia entre nodos de interficie de código (Code Interface Nodes o CiNs).

Paths (verde oscuro) LabVIEW almacena las componentes tipo y número de un path en palabras enteras, seguidas inmediatamente por las componentes del path.

El tipo de path es 0 para un path absoluto y 1 para un path relativo. Cualquier otro valor indicaría que el path no es válido. Cada componente del path es una cadena Pascal (P-string), en la cual el primer byte es la longitud de la P-string (sin incluir el byte de longitud).

Clusters (marrón o rosa) Un cluster almacena diferentes tipos de datos de acuerdo a las siguientes normas:

Los datos escalares se almacenan directamente en el cluster; los arrays,strings, handies y paths se almacenan indirectamente. El cluster almacena un handie que apunta al área de memoria en la que LabVIEW ha almacenado realmente los datos.

Para conectar terminales se usa la herramienta **Wiring** (cableado).

4.2.8 PROGRAMACIÓN ESTRUCTURAL

A la hora de programar, muchas veces es necesario ejecutar un mismo conjunto de sentencias un número determinado de veces, o que éstas se repitan mientras se cumplan ciertas condiciones. También puede ocurrir que queramos ejecutar una u

otra sentencia dependiendo de las condiciones fijadas o simplemente forzar que unas se ejecuten siempre antes que otras.

Para ello LabVIEW dispone de cuatro estructuras fácilmente diferenciables por su apariencia y disponibles en la opción **Structures** del menú **Function** de la ventana **Diagram:**

4.2.8.1 ESTRUCTURAS ITERATIVAS: FOR LOOP Y WHILE LOOP

FORLOOP

Usaremos **For Loop** cuando queramos que una operación se repita un número determinado de veces. Su equivalente en lenguaje convencional es:

For i = 0 to N-1

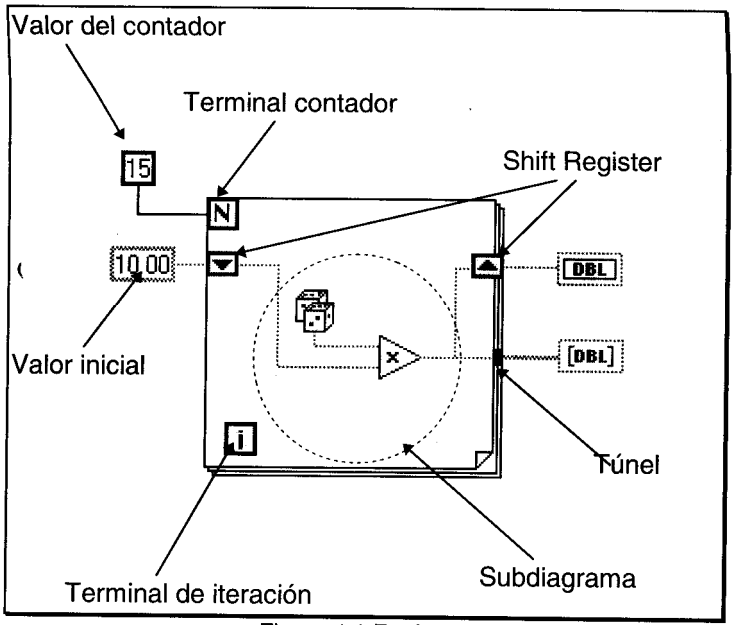
Ejecuta subdiagrama

Al colocar un **For Loop** en la ventana **Diagram** observamos que tiene asociados dos terminales:

1.- Terminar contador: Contiene el número de veces que se ejecutará el subdiagrama creado en el interior de la estructura. El valor del contador se fijará externamente (ver también **Arrays** en el capítulo 6).

2.- Terminal de iteración: Indica el número de veces que se ha ejecutado la estructura: Cero durante la primera iteración, uno duretnte la segunda y así hasta N-1.

Ambos terminales son accesibles desde el interior de la estructura, es decir, sus valores podrán formar parte del subdiagrama pero en ningún caso se podrán modificar.



WHILE LOOP

Usaremos **Whíle Loop** cuando queramos que una operación se repita mientras una determinada condición sea cierta. Su equivalente en lenguaje convencional es:

Do ejecutar subdiagrama **While** condición **is TRUE**

(Aunque esta estructura es más similar al comando *Repeat-Untíl*, ya que se repite como mínimo una vez, independientemente del estado de la condición).

Al igual que **For Loop** contiene dos terminales:

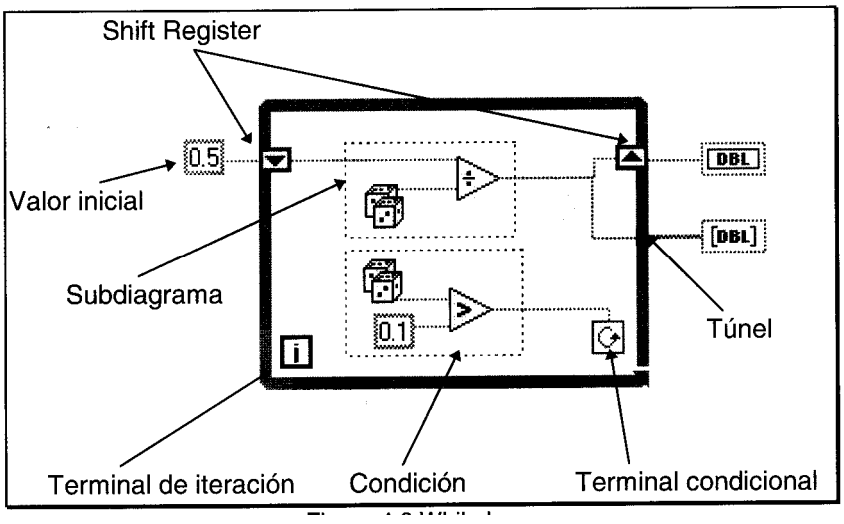
1.-Terminal condicional: A él conectaremos la condición que hará que se ejecute el subdiagrama. LabVIEW comprobará el estado de este termina¡ al final de cada iteración, si su valor es TRUE (verdadero) continuará, pero por el contrario si su valor es FALSE (falso) detendrá la ejecución.

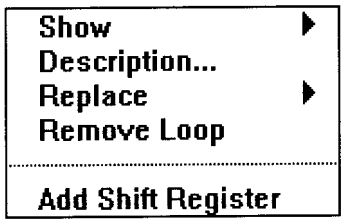
2.- Terminal de iteración: Indica el número de veces que se ha ejecutado el bucle y que, como mínimo, siempre será una (l=O).

Al hacer pop-up tanto en el **For Loop** como en el **Whíle Loop** se despliega el siguiente menú:

Show Label: Oculta o visualiza la etiqueta de identificación del **Loop y,** si no existe, permite ponerla.

- Description: Permite añadir comentarios.
- Replace: Cambia el **For Loop** o el **While Loop** por cualquier otra función de la paleta **Structs & Constants.**
- Remove Loop: Borra la estructura **While** o **For** pero sin eliminar el subdiagrama de su interior.
- **Add** Shift Register: Añade los **shift register** (registros de desplazamiento).





4.2.8.2 REGISTROS DE DESPLAZAMIENTO

Los registros de desplazamiento o **shift register** son variables locales, disponibles tanto en el **For Loop** como en el **While Loop,** que permiten transferir los valores del final de una iteración al principio de la siguiente.

Inicialmente **shift register** tiene un par de terminales colocados a ambos lados del **Loop;** el terminal de la derecha almacena el valor final de la iteración hasta que una nueva hace que este valor se desplace al terminal de la izquierda, quedando en el de la derecha el nuevo valor. Un mismo registro de, desplazamiento puede tener más de un terminal en el lado izquierdo; para añadirlo escogeremos la opción **Add Element** (añadir elemento) del menú pop-up. Cuantos más terminales tengamos en el lado izquierdo más valores de iteraciones anteriores podremos almacenar.

El menú pop-up tiene otros dos comandos:

- Remove element: Borra un terminal del lado izquierdo siempre y cuando el registro de desplazamiento tenga asociado más de uno.
- Remove All: Borra todo el registro de desplazamiento, tanto los termínales de la izquierda como el de la derecha.

Un mismo **Loop** puede tener varios registros de desplazamientos siendo conveniente inicializarlos, para que los terminales de la izquierda tengan el valor

deseado cuando se produzca la primera iteración. **Shift register** puede trabajar con cualquier tipo de datos siempre y cuando los datos que se conecten a cada terminal sean del mismo tipo.

Al finalizar la ejecución de todas las iteraciones el último valor quedará en el terminal de la derecha; uniéndolo a un indicador del mismo tipo de dato fuera del **Loop** podremos obtener su valor.

Pero existe otra posibilidad para pasar datos de forma automática desde el interior de la estructura al exterior. Cuando un cable atraviesa los límites del **Loop,** aparece en el borde un nuevo terminal llamado túnel que hace de conexión entre el interior y el exterior, de forma que los datos fluyen a través de él después de cada iteración del **Loop,** pudiendo guardar de esta manera no sólo el último valor de todas las iteraciones sino también los valores intermedios. A esta posibilidad que tienen tanto el **For** como el **While** de acumular arrays en sus límites automáticamente se le llama **auto-indexing** o autoindexado.

4.2.6 PROGRAMACIÓN ESTRUCTURADA

LabVIEW habilita por defecto **auto-indexing** en el **For Loop** ya que es más frecuente utilizar esta estructura para crear arrays que no el **While Loop,** en el cual esta opción está deshabilitada por defecto y cuya utilización podría provocar problemas de memoria debido a que no sabemos cuantas veces se va a ejecutar. No obstante, haciendo pop-up en el túnel se puede habilitar o deshabilitar esta opción.

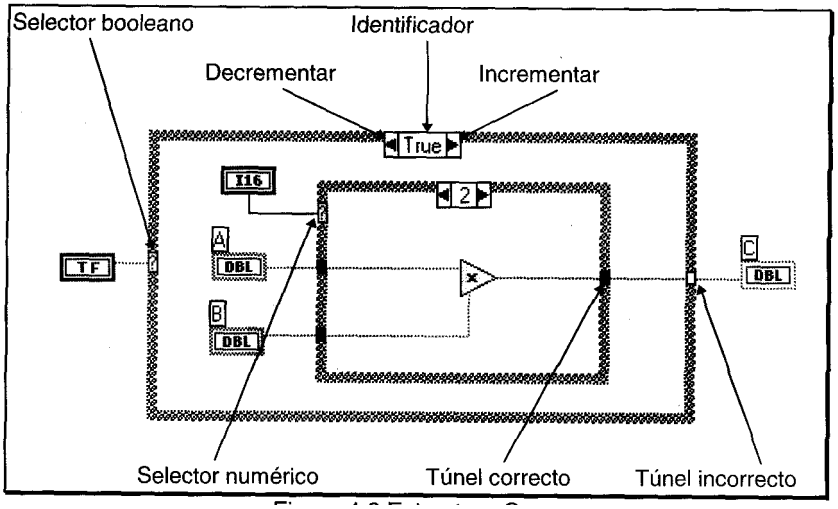
4.2.6.1 ESTRUCTURAS CASE Y SEQUENCE

Este tipo de estructuras se diferencia de las iterativas en que puede tener múltiples subdiagramas, de los cuales solamente uno es visible a la vez. En la parte superior de cada estructura existe una pequeña ventana que muestra el identificador del subdiagrama que se está mostrando. A ambos lados de esta ventana existen dos botones que decrementan o incrementan el identificador de forma que podamos ver el resto de subdiagramas.

CASE

Usaremos la estructura **Case** en aquellas situaciones en las que el número de alternativas disponibles sean dos o más. Según qué valor tome el selector dentro de los n valores posibles, se ejecutará en correspondencia uno de los n subdiagramas.

La estructura **Case** consta de un terminal llamado selector y un conjunto de subdiagramas, cada uno de los cuales está dentro de un case o suceso y etiquetado por un identificador del mismo tipo que el selector; éste será booleano o numérico. Si se conecta un valor booleano al selector, la estructura tendrá dos **Case:** False y True.



Pero si se conecta un valor numérico la estructura podrá tener hasta 214 **Case.**

En este caso la estructura **Case** engloba dos sentencias diferentes de otros lenguajes convencionales:

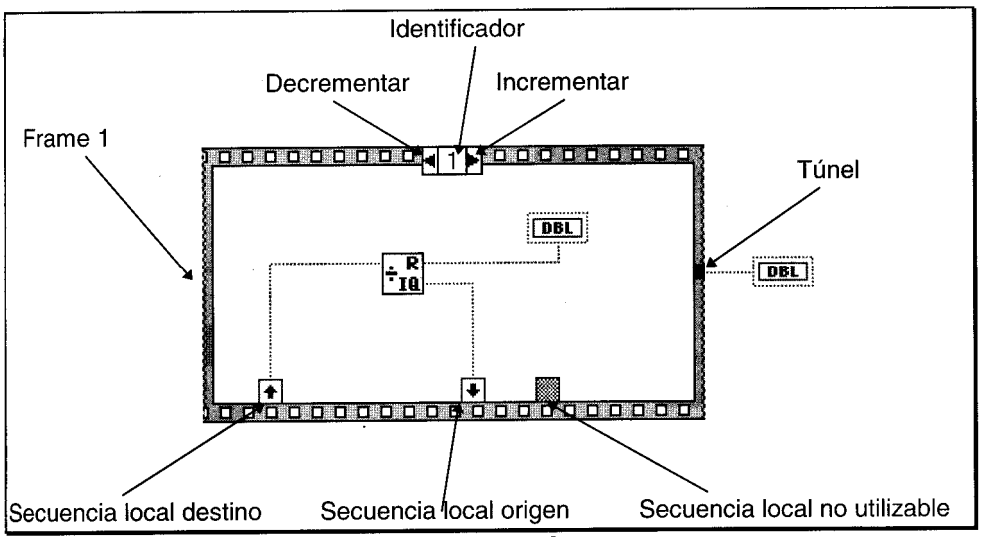
1.- lf condición true then ejecutar case true
    else ejecutar case false

2.- Case selector of
    l:ejecutar case 1;
    n:ejecutar case n
  end

Case no cuenta con los registros de desplazamiento de las estructuras iterativas pero sí podemos crear los túneles para sacar o introducir datos. Si un case o suceso proporciona un dato de salida a una determinada variable será necesario que todos los demás también lo hagan; si no ocurre de esta manera será imposible ejecutar el programa.

SEQUENCE

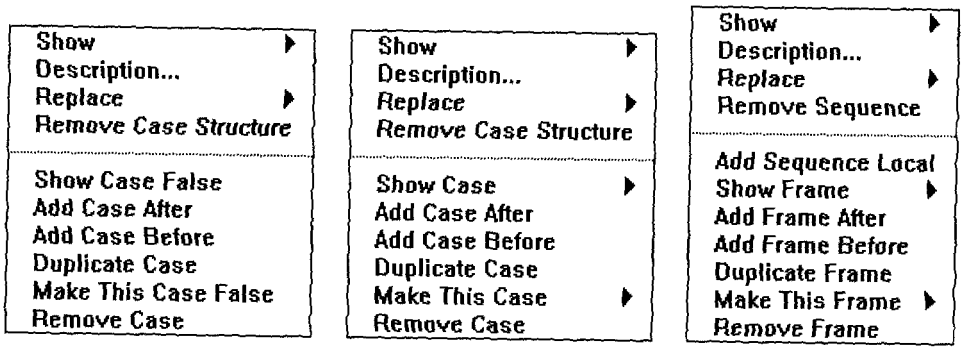
Esta estructura no tiene su homóloga en los diferentes lenguajes convencionales, ya que en éstos las sentencias se ejecutan en el orden de aparición pero, como ya sabemos, en LabVIEW una función se ejecuta cuando tiene disponible todos los datos de entrada. Se produce de esta manera una dependencia de datos que hace que la función que recibe un dato directa o indirectamente de otra se ejecute siempre después, creándose un flujo de programa.



Pero existen ocasiones en que esta dependencia de datos no existe y es necesario que un subdiagrama se ejecute antes que otro; es en estos casos cuando usaremos la estructura **Sequence** para forzar un determinado flujo de datos. Cada subdiagrama estará contenido en un **frame** o marco y estos se ejecutarán en orden de aparición: Primero el *frame 0 o marco* 0, después el *frame* 1 y así, sucesivamente, hasta el último.

Al contrario del **Case,** si un **frame** aporta un dato de salida a una variable los demás no tendrán por qué hacerlo. Pero tendremos que tener en cuenta que el dato estará solamente disponible cuando se ejecute el último **frame** y no cuando se ejecute el **frame** que transfiere el dato.

Debido a la similitud de los menús pop-up de la estructuras **Case** y **Sequence** vamos a estudiarlos de forma conjunta indicando en cada caso las posibles diferencias que puedan existir:

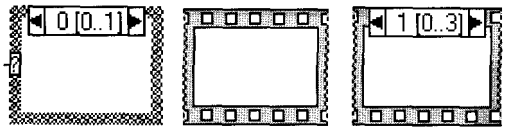


- Show Label: Oculta o visualiza la etiqueta de identificación de la estructura y, si no existe, permite ponerla.
- Description: Permite añadir comentarios.
- Replace: Cambia la estructura **Case** o **Sequence** por cualquier otra función de la paleta **Structs & Constants.**
- Remove Case Structure o Sequence: Borra completamente la estructura **Case** o **Sequence** y todos los subdiagramas menos el que se esté visualizando en el momento de la ejecución de este comando.
- Add Sequence Local (añadir secuencia local): Esta opción está sólo disponible en el menú de la estructura **Sequence** y se utiliza para pasar datos de un **frame** a otro.

Una pequeña flecha con la punta hacia el exterior de la estructura indica el **frame** de origen de la secuencia local, mientras que una flecha apuntando hacia el interior indica que la secuencia local contiene un dato de salida. Todos los **frames** posteriores al que contiene la secuencia local que origina el dato podrán disponer de él, no siendo así para los **frames** anteriores en los cuales aparecerá un cuadrado vacío que indicará que los datos no están disponibles.

- Show Case o Show Frame: Nos permite ir directamente al subdiagrama que queremos visualizar sin tener que pasar por todos los case o frame intermedios que pudiera haber. Al pulsar esta opción, un menú conteniendo todos los identificadores se desplegará y sólo tendremos que señalar con el cursor del ratón el que deseamos ver. Si sólo hubiese dos subdiagramas nos aparecerá directamente el nombre del único identificador que podemos visualizar, como es el caso del case con selector booleano.
- Add Case After o Add Frame After: Este comando inserta un subdiagrama vacío inmediatamente después del que se está visualizando.
- Add Case Before o Add Frame Before: Inserta un subdiagrama vacío justo un nivel por encima de¡ que se está visualizando.
- Duplicate Case o Duplicate Frame: Inserta una copia del subdiagrama visible inmediatamente después de él.
- Make This Case o Make This Frame: Mueve un subdiagrama a otra posición.
- Remove Case o Remove Frame: Borra el subdiagrama visible. Este comando no está disponible si solamente existe un **case** o un **frame.**

Se puede ver en el caso de las estructuras **Sequence** y **Case** numérico. En el primer caso, si sólo hay una secuencia, no aparece ningún identificador de **frame;** mientras que si hay más de uno, se nos indica en cuál estamos y cuántos hay. Lo mismo pasa con la estructura **Case,** sólo que, en este caso tendremos, como mínimo, dos posibles estados. Todo ello queda reflejado a continuación:



Formula Node o nodo de fórmula es una función de características similares a las estructuras vistas anteriormente, disponible en la paleta **Structs & Constants** del menú **Functions,** pero que, en lugar de contener un subdiagrama, contiene una o más fórmulas separadas por un punto y coma. Usaremos **Formula Node** cuando queramos ejecutar fórmulas matemáticas que serían complicadas de crear utilizando las diferentes herramientas matemáticas que LabVIEW incorpora en sus librerías.

Una vez escrita la fórmula en el interior del rectángulo sólo tendremos que añadir los terminales que harán la función de variables de entrada o de salida; para ello desplegaremos el menú pop-up de la estructura y ejecutaremos el comando **Add Input** (añadir entrada) o **Add Output** (añadir salida).

Menú pop-up Formula Node

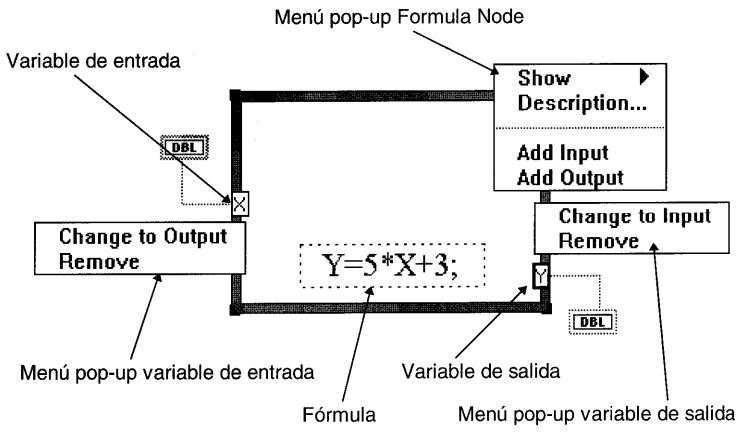


Figura 4.4 Formula Node

Cada variable, además, tendrá otro menú pop-up que permitirá definirla como de salida si anteriormente era de entrada, o de entrada si en un principio era de salida **(Change to Output** o Cambiar a Salida, **Change to Input** o Cambiar a Entrada). También podremos eliminarla mediante el comando **Remove.**

No hay límite para el número de variables o de fórmulas y nunca podrá haber dos entradas o dos salidas con el mismo nombre, aunque una salida sí podrá tener el mismo nombre que una entrada. Todas las variables de salida deberán estar asignadas a una fórmula por lo menos una vez.

La tabla muestra algunas de las funciones de **Formula Node:**

abs(x)	Devuelve el valor absoluto de x.
Acos(X)	Calcula el coseno inverso de x en radianes.,l
acosh(x)	Calcula el coseno hiperbólico inverso en radianes.
Asin(x)	Calcula el seno inverso de x en radiahes.
asinh(x)	Calcula el seno hiperbólico inverso en radianes.
atan(x,y)	Calcula la tangente inversa de y/x en radianeg,.
atanh(x)	Calcula la tangente hiperbólico inversa en radianes.
COS(X)	Calcula el coseno de x en radianes.
cosh(x)	Calcula el coseno hiperbólico de x en radianes.

cot(x)	Calcula la cotangente de x en radianes.
CSC(X)	Calcula la cosecante de x en radianes.
exp(x)	Calcula el valor de e elevado a x.
ln(x)	Calcula el logaritmo natural de x.
log(x)	Calcula el logaritmo de x en base 1 0.
log2(X)	Calcula el logaritmo de x en base 2.
max(X,Y)	Compara x con y, y devuelve el mayor valor.
min(x,y)	Compara x con y, y devuelve el menor valor.
mod(x@y)@	Calcula el cociente de x/y.
rando	Genera un número aleatorio entre 0 y 1.
Sic(x)	Calcula la secante de x en radianes.
sign(x)	Devuelve 1 si x es mayor que 0, 0 si x es igual a 0 y -1 si x es menor que cero.

Sin(x)	Calcula el seno de x en radianes.
sinc(x)	Calcula el seno de x dividido por'x en radianes.
sinh(X)@	Calcula el seno hiperbólico de x en radianes.
sqrt(x)	Calcula la raíz cuadrada de x.
Tan(x)	Calcula la tangente de x en radianes.
tanh(x)	Calcula la tangente hiperbólico de x en radianes.

4.2.6.2 VARIABLES LOCALES Y GLOBALES

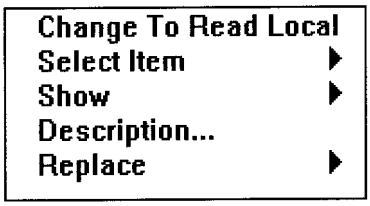
Las variables son imprescindibles en cualquier tipo de problemas, ya que permiten almacenar la información necesaria para su resolución.

En LabVIEW todos los controles introducidos en el Panel Frontal que generan un termina¡ en la ventana Diagrama van a ser variables, identificabas por el nombre asignado en la etiqueta. Pero puede ocurrir que queramos utilizar el valor de cierta variable en otro subdiagrama o en otro Vi o, simplemente, que queramos guardar un resultado intermedio. La forma mas sencilla de hacerlo es generando variables locales y/o globales dependiendo de la aplicación.

VARIABLES LOCALES

En las variables locales los datos se almacenan en algunos de los controles o indicadores existentes en el Panel Frontal del Vi creado; es por eso que estas variables **no** sirven para intercambiar datos entre VI's. La principal utilidad de estas variables radica en el hecho de que una vez creada la variable local no importa que proceda de un indicador o de un control, ya que se podrá utilizar en un mismo Diagrama tanto de entrada como de salida.

Las variables locales están disponibles en el menú **Structs & Constants** de la paleta **Function** y disponen de¡ siguiente menú pop-up:



- Change To Read Local o Change To Write Local: Permite escoger entre leer o escribir en el control.
- Select Item: Visualiza una lista con el nombre de todos los controles existentes en el Panel Frontal y de ella escogeremos el control al cual queremos que haga referencia nuestra variable. Es por esto que para poder crear la variable local será imprescindible que el control tenga asignado un nombre de identificación. Una vez creada la variable local, si en algún momento se cambia el nombre del control origen, será necesario cambiar también el nombre de la variable local ya que LabVIEW no actualiza los cambios.
- Show Label: Muestra una etiqueta con el nombre del Vi al que pertenece la variable local.
- Description: Permite añadir comentarios.
- Replace: Sustituye la variable local por cualquier otra función.

VARIABLES GLOBALES

Las variables globales son un tipo especial de VI, que únicamente dispone de Panel Frontal, en el cual se define el tipo de dato de la variable y el nombre de identificación imprescindible para después podernos referir a ella.

Cuando escogemos la función **Global** del menú **Structs & Constants** creamos un nuevo terminal en el Diagrama; este terminal corresponde a un VI que inicialmente no contiene ninguna variable. Para poderias añadir haremos doble clic en el terminal y se abrirá el panel frontal. Una vez abierto, las variables se definen igual que cualquier control o indicador de un VI normal. Podemos crear un Vi para cada variable global o definirlas todas en el mismo, que es la opción más indicada para cualquier aplicación. Cuando terminemos de colocar todas las variables grabaremos el VI y lo cerraremos. Si una vez cerrado queremos añadir nuevas variables, bastará con volverlo a abrir e introducir los cambios necesarios. Para añadir nuevos terminales que hagan referencia a las variables globales creadas, no volveremos a ejecutar la función *Global* ya que esto crearía un nuevo Vi sino que abriremos el ya existente mediante el comando VI.. del menú *Functíon y* seleccionaremos la variable en concreto a través del comando *Select ltem* del menú pop-up. Además, este mismo menú cuenta con otra opción que nos permite utilizar una variable ya creada para leer datos o para almacenarlos: Se trata del comando *Change To Read Global o Change To Wríte GlobaL*

ATTRIBUTE NODE

Los **attribute nodes** o nodos de atributos se pueden considerar como variables que dependen únicamente del terminal a partir del cual se han creado y que permiten

leer o modificar atributos del panel frontal de un control o indicador como, por ejemplo, cambiarlo de color, hacerlo invisible, desactivarlo, leer posiciones de cursores, cambiar escalas, etc. Para crear un **attribule node** basta con seleccionar la opción ***Create Attribute Node*** del menú pop-up de cualquier control del Panel Frontal o terminal del Diagrama de Bloques (se puede desplegar primero desplegamos el menú pop-up del objeto y a continuación tomamos la opción **Create.** Podremos crear un **attribute node** o variable local). Una vez creado aparece en el Diagrama un nuevo nodo que puede ser tanto de escritura como de lectura. Una pequeña flecha a la izquierda del nodo indica que éste es de escritura, mientras que una flecha a la derecha indica que es de lectura. Además los **attribute nodes** tienen su propio menú pop-up como se muestra a continuación.

- Change All To Read o Change All To Write: Dependiendo de si el nodo es de escritura o de lectura aparecerá una opción u otra que nos permitirá cambiar entre ambas. Debido a que un mismo **attribute node** puede tener más de un terminal

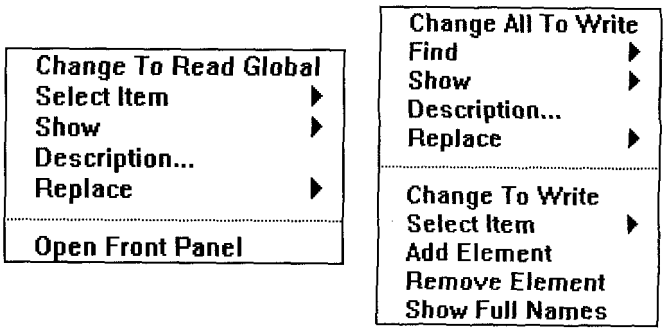


Figura Menú pop-up de una variable global

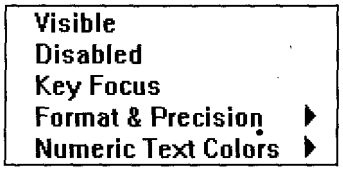
usaremos esta opción cuando queramos que todos ellos sean de escritura o de lectura.

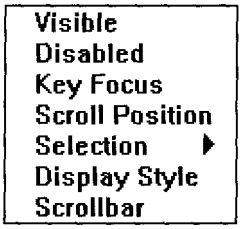
- Find Control: Encuentra el control asociado a dicho **attribute node** en el Panel Frontal.

- Find Terminal: Encuentra el terminal asociado a dicho **attríbute node** en el Diagrama de Bloques.

- Find Attribute Nodes: Muestra todos los **attribute nodes** existentes en el Diagrama y que están asociados a dicho control.
- Show Label: Oculta o visualiza la etiqueta identificativa del **atribuye node.**
- Description: Permite añadir comentarios.
- Replace: Sustituye el **attribute node** por cualquier otra función.
- Change To Read o Change To Write: Cambia a modo de escritura o de lectura únicamente el terminal seleccionado dejando los demás tal y como estaban.
- Select ltem: Visualiza todos los atributos disponibles para el control asociado al **attribute node** y permite cambiar un atributo por otro diferente. Podemos acceder directamente a esta opción colocándonos encima del atributo que deseamos cambiar **y** pulsando el botón izquierdo del ratón.

Por ejemplo, los atributos para un control numérico son:

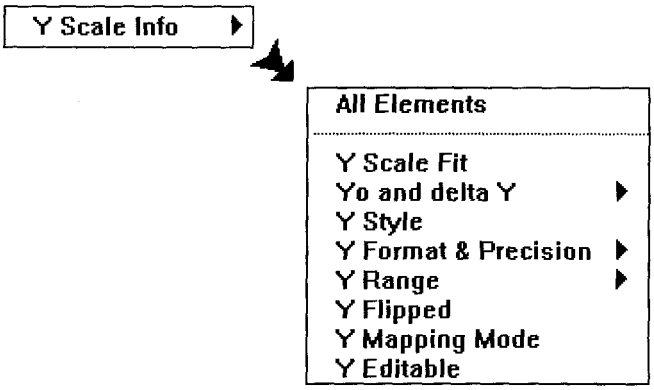




Y para un string :

Remove Element.- Borra el terminal seleccionado.

Add Element. Añade un nuevo terminal.



Algunos controles como los **graph** tienen un gran número de atributos. Muchos de estos atributos se agrupan en categorías, como es el caso de **Y Scale rifo** para un indicador **XY Graph.** Una pequeña flecha a la derecha del atributo nos indica que se trata de una categoría.

Se pueden seleccionar todos los atributos de una categoría de una sola vez mediante el comando **All elements** (todos los elementos), aunque también podemos seleccionarlos individualmente escogiendo el atributo específico.

INDICADORES GRAFICOS

En muchas ocasiones es necesario para una mayor comprensión de los resultados obtenidos representarlos gráficamente. Para ello LabVIEW dispone de cinco tipos de gráficos accesibles desde el menú *Controls* de¡ panel Frontal bajo e

ítem Graph, divididos en dos grupos: Los *indícadores charl* y los *indicadores graph.*

Un indicador *graph* o indicador gráfico es una representación bídimensional dE una o mas gráficas. El *graph* recibe los datos como un bloque. Un indicador

chartc de trazos también muestra gráficas, pero éste recibe los datos y los muestra puntc por punto o array por array, reteniendo un cierto número de puntos en pantalla mediante un buffer disponible para ello.

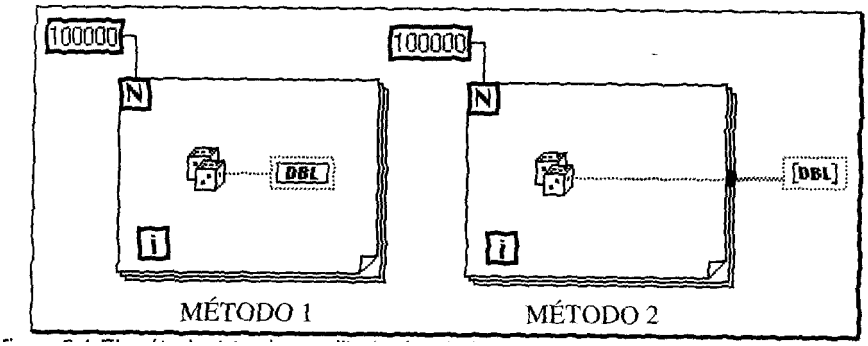
4.2.6.3 INDICADORES CHART Se encuentran los siguientes.

WAVEFORM CHART

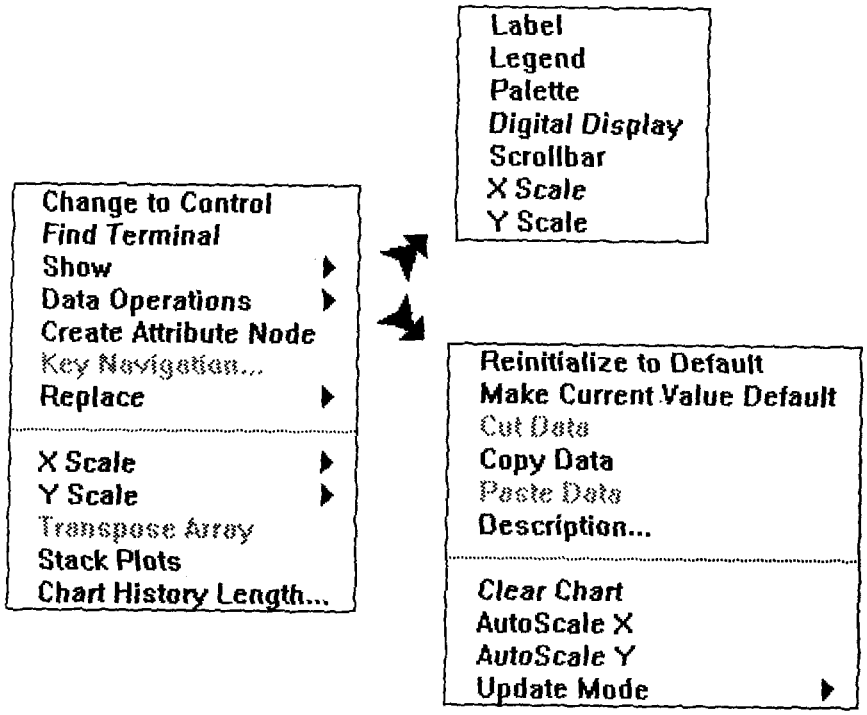
Waveform chart es un tipo especial de indicador numérico que muestra una o mas gráficas, reteniendo en pantalla un cierto número de datos definido por nosotros mismos. Los nuevos datos se añaden al lado de los ya existentes, de forma que se pueden comparar entre ellos.

Los datos se pueden pasar uno a uno al *chart* o mediante arrays, Evidentemente es mucho conveniente pasar múltiples puntos a la vez ya que de esta manera sólo es necesario redibujar la gráfica una vez y no una por cada punto (figura).

Es posible dibujar varias gráficas en un mismo *chart*, uniendo los datos de cada gráfica en un cluster de escalares numéricos de forma que cada escalar que contiene el cluster se considera como un punto de cada una de las gráficas para una misma abcisa. Se puede ahorrar tiempo uniendo los clusters en arrays y después pasando todo el array a la gráfica.



Desplegando el menú pop-up se tiene acceso a las siguientes opciones:



- Change to Control o Change to Indicator: Dependiendo de si la waveforrn es un control o un indicador nos permitirá cambiar entre ellas.
- Find Terminal: Muestra el terminal asociado en el Diagrama de bloques.
- Show Label: Permite poner una etiqueta de identificación a la waveform chart y, si ya existe, la visualiza.
- Show Legend: Permite poner una etiqueta de identificación a cada una de las gráficas.

- Show Palette: Activa una paleta que permite hacer zooms, desplazar la gráfica de forma rápida, ajustar automáticamente la escala de los ejes, cambiar el formato y la precisión de los indicadores numéricos y elegir entre escala lineal o logarítmica.
- Show Digital Display: Es un indicador que muestra el último valor que se ha cargado en pantalla. Hay un indicador por cada gráfica.
- Show Scrolibar: Permite ver los valores anteriores contenidos en el buffer.
- Show X Scale: Visualiza la escala del eje de abcisas.
- Show Y Scale: Visualiza la escala del eje de ordenadas.
- Reinitialize to Default: Actualiza el último punto obtenido al valor por defecto.
- Make Current Value Default: Convierte el último punto obtenido en el valor por defecto.
- Description: Permite añadir comentarios.
- Clear Chart: Borra el contenido del buffer.
- AutoScale X: Ajusta de forma automática el rango de valores de X para una correcta visualización.
- AutoScale Y: Ajusta de forma automática el rango de valores de Y para una correcta visualización.
- Update Mode: Permite escoger entre tres modos de visualizar los nuevosstrip chart, scope chart y sweep chart. El modo strip chart es el modo por defecto y consiste en que cada nuevo valor se coloca a la derecha del display, mientras que valores anteriores se desplazan hacia la izquierda. En el modo scope chart cada nuevo valor se coloca a la derecha del anterior, empezando por el margen izquierdo del display. Cuando se llega al margen derecho se borra todo el display y se comienza de nuevo desde la izquierda. El modo scope chart es mucho más rápido que el modo

strip chart ya que no es necesario realizar todo el proceso de desplazar la pantalla hacia la izquierda para cada nuevo punto. El modo sweep chart actúa como el modo scope chart, alvo que ahora cuando se llega al final de la pantalla, ésta no se borra y se S comienza de nuevo desde el principio, donde una línea vertical se mueve hacia la derecha cada vez que se añade un nuevo punto.

- Create Attribute Node: Crea un nodo asociado al terminal del que procede en el Diagrama de Bloques.
- Replace: Permite sustituir la waveform chart por cualquiera de los controles e indicadores del Panel Frontal.
- X Scale and Y Scale: Permite escoger el estilo de la escala, tipo de rejilla, punto inicial, incremento entre punto y punto, formato y precisión de estos puntos.
- Transpose Array: Cuando se representa más de una gráfica en una misma chart utilizando arrays, waveform chart interpreta por defecto las filas como gráficas diferentes. Pero si a nosotros nos interesa que sean las columnas las gráficas diferentes, utilizaremos este comando para convertir las columnas en filas.
- Stack Plots: Normalmente cuando se representan más de una gráfica todas ellas se sitúan en un mismo display. Pero puede ocurrir que las escalas de las ordenadas sean muy diferentes entre ellas o que simplemente nos interese representarlas por separado, cada una en un display. Para conseguir esto activaremos el comando Stack Plot de forma que cada gráfica aparecerá con su propia escala y su propio display. Cuando Stack Plots está activado, en su lugar aparece el comando Overlay Plot que es el que dibuja todas las gráficas en un mismo display.

Chart History Length: Mediante este control podemos fijar el número de puntos que waveform chart almacenará en el buffer que, por defecto, serán 1024.

INTENSITY CHART

Mediante *íntensíty chart* podemos mostrar datos tridimensionales colocando bloques de colores sobre planos cartesianos. Para ello crearemos arrays bidimensionales de números donde los índices de un elemento corresponderán a las coordenadas X e Y, y el contenido a la coordenada Z, que tendrá asociado un color para cada posible valor. Previamente será necesario definir la escala de colores que vamos a utilizar a través de los *atribuye nodes* mediante el ítem *Z Scale lnfo: Color Array o Color Table,* o a través de la rampa de colores visualizada junto a la gráfica.

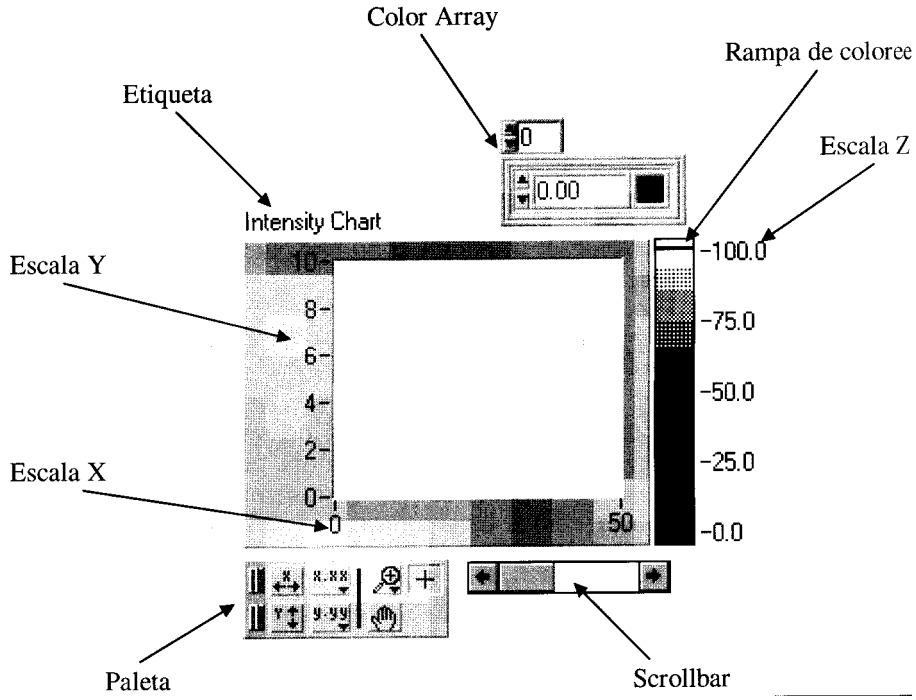


Figura 4.5 Indicador intensity chart

Evidentemente, la escala de colores que podamos visualizar dependerá de la resolución de nuestro monitor.

Cada vez que se envíe un nuevo conjunto de datos, estos aparecerán representados a la derecha de los ya existentes. *Intensíty chart* soporta los tres modos de visualización de *waveform charl* y también dispone de un buffer cuyo tamaño es, por defecto, de 128 puntos. Las opciones disponibles para *íntensity chart* son prácticamente las mismas que para *waveform chari.* únicamente, debido a que existe una nueva coordenada, aparecen en el menú opciones para ésta, como son:

- Show Ramp: Visualiza u oculta la rampa de colores.
- Show Color Array: Permite fijar los colores de la rampa.
- Show Z Scale: Visualiza u oculta la escala z.
- AutoScale Z: Ajusta de forma automática el rango de valores de z a la escala de colores.
- Z scale: Permite escoger el estilo de la escala, tipo de rejilla, punto inicial, incremento entre punto y punto, formato y precisión de estos puntos.

4.2.6.4 INDICADORES GRAPH Se encuentran los siguientes.

WAVEFORM GRAPH

Waveform graph representa una serie de valores Y equiespaciados dada siempre una distancia delta de X (ÁX) comenzando a partir de un valor inicial Xo. A un mismo punto X, sólo le puede corresponder un valor de Y,. Cuando se representa una nueva serie de datos, al contrario de lo que ocurría en los indicadores chart, estos datos reemplazan a los ya existentes en lugar de añadirse al lado, y pierden los valores representados con anterioridad.

Existen dos posibilidades a la hora de representar una única gráfica en una *waveform graph.* La primera consiste en unir un array de valores numéricos

directamente a la *graph* de forma que ésta interpreta cada valor como un nuevo punto comenzando en X=0 e incrementando X en 1 para cada punto.

La segunda consiste en crear un cluster en,.el cual, junto con el array de valores, se indica el valor inicial Xo y el incremento AX.

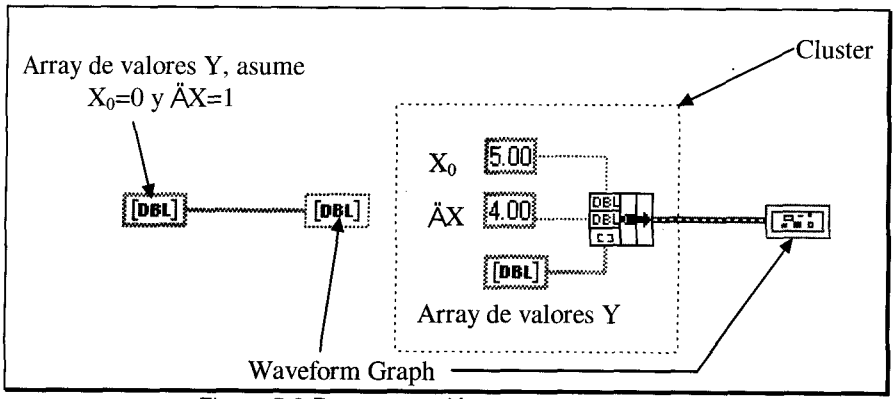


Figura 4.6 Representación de una sola gráfica

Existe la posibilidad de representar más de un gráfica en una misma *waveform graph.* Para ello es necesario unir los datos de las diferentes gráficas en un formato que LabVIEW sepa interpretar. Utilizar un formato u otro vendrá determinado principalmente por las características de las gráficas a mostrar. Así, si todas las gráficas tienen un mismo escalado X y un mismo número de puntos, bastará con crear un array bidimensional de valores numéricos donde cada fila de datos es una únicá gráfica. LabVIEW interpretará estos datos como puntos en la gráfica comenzando en X=0 e incrementándola en 1. Si nos interesa cambiar el punto inicial o el incremento de x, crearemos un cluster que contendrá el array bidimensional y los valores de xo y Áx.

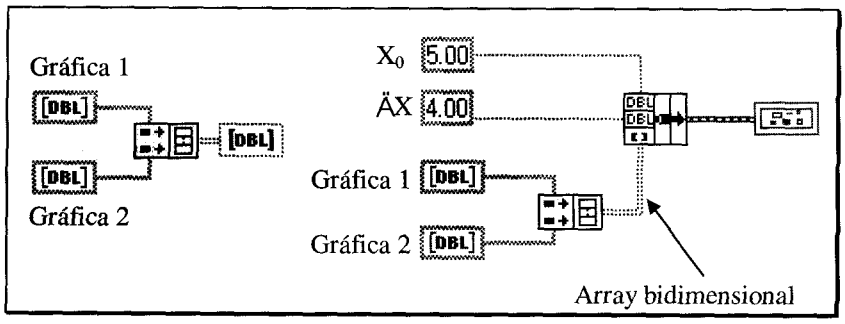


Figura 4.7 Representación de múltiples gráficas con el mismo número de puntos

Mediante el comando *Transpose Array* del menú pop-up podemos hacer que LabVIEW interprete las columnas como gráficas diferentes en lugar de las filas.

Puede ocurrir que el número de elementos de cada gráfica sea diferente. En ese caso es necesario crear un cluster para cada array de datos y después unir todos los clusters en un array. Esto es necesario debido a que LabVIEW no permite crear arrays de arrays. Al igual que anteriormente si nos interesa que el punto inicial sea diferente de cero o que el incremento sea diferente de 1, crearemos un cluster que contenga el array de clusters de array y los nuevos valores de Xo y ÁX.

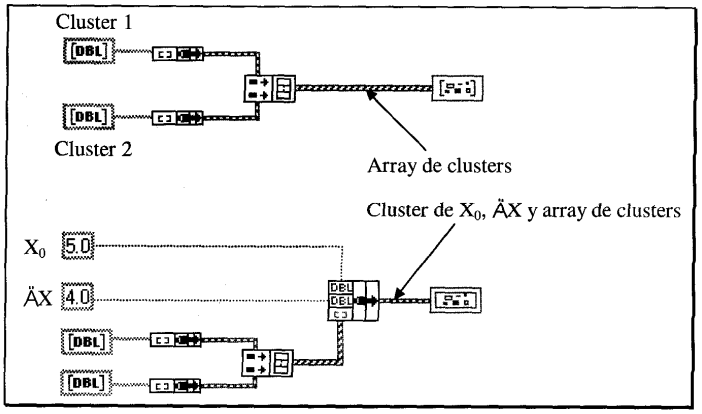


Figura 4.5 Representación de múltiples gráficas con diferente números de puntos.

Finalmente, si ni el escalado ni el número de puntos de la gráfica es el mismo para todas ellas, lo que haremos será crear un cluster por cada gráfica que contendrá un array de datos, un valor Xo y un valor ÁX. Y con todos los clusters de las diferentes gráficas crearemos un array. Este último formato es el más completo de todos porque permite fijar un valor Xo y un valor ÁX diferente para cada gráfica.

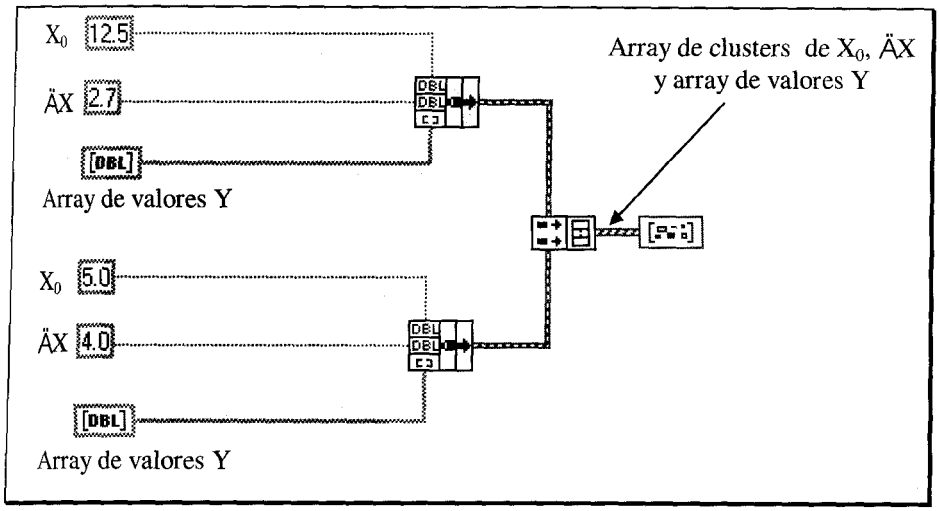


Figura 4.8 Representación de múltiples gráficas con diferente número de puntos y diferente escalado

XY GRAPH

En *XY Graph* un punto X, puede tener varios valores Y, lo que permite, por ejemplo, dibujar funciones circulares. *XY Greph* representa una coordenada (Xi, Y,) donde los valores de X no tienen porque estar equiespaciados como ocurría en *las waveform graph.*

Para representar una única gráfica en una *XY Graph* existen dos posibilidades.

La primera consiste en crear un cluster que contenga un array de datos X y un array de datos Y. La segunda consiste en crear un array de clusters, donde cada cluster contiene un valor de X y un valor de Y.

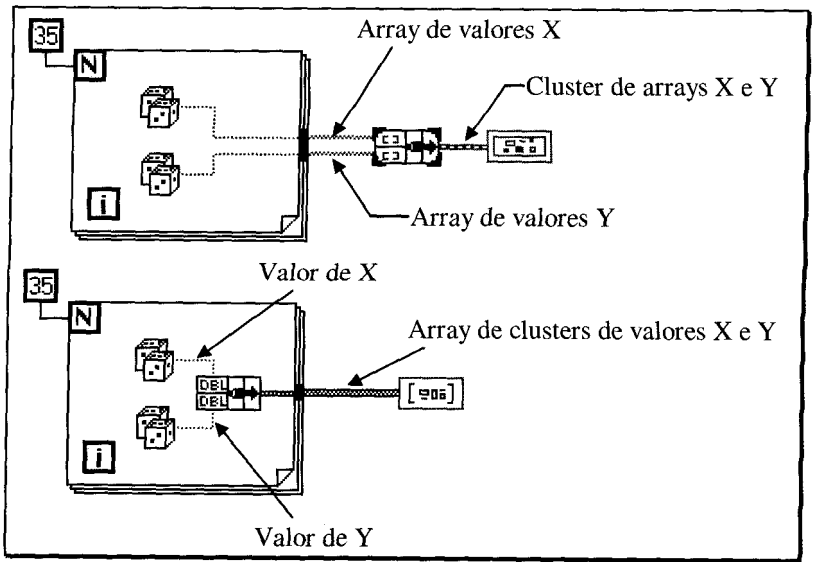


Figura 4.9 Posibles representaciones de una única XY Graph

Al igual que en las *waveform graph* existe la posibilidad de representar más de una gráfica en una misma *XY Graph* (figura). Pero, en este caso, tan sólo existen dos formatos posibles derivados de los dos formatos vistos anteriormente para una única gráfica. El primer formato es un array de gráficas, donde cada gráfica es un cluster de un array X y un array Y. Y el segundo formato es un array de clusters de gráficas, donde cada gráfica es, a su vez, otro array de clusters conteniendo un valor X y un valor Y.

INTENSITY GRAPH

Intensíty graph es exactamente igual que *íntensity chart* salvo que *intensity graph* no retiene valores anteriores, por lo que cuando un nuevo bloque de valores se carga, éstos sustituyen a los ya existentes.

Los comandos disponibles en los menús pop-up de los indicadores *graph* tienen las mismas utilidades que los descritos en los indicadores *chart*, por lo que no se han mencionado en este apartado. Solamente existe una diferencia importante y es que los indicadores graph disponen de cursores que nos permiten movernos por la gráfica.

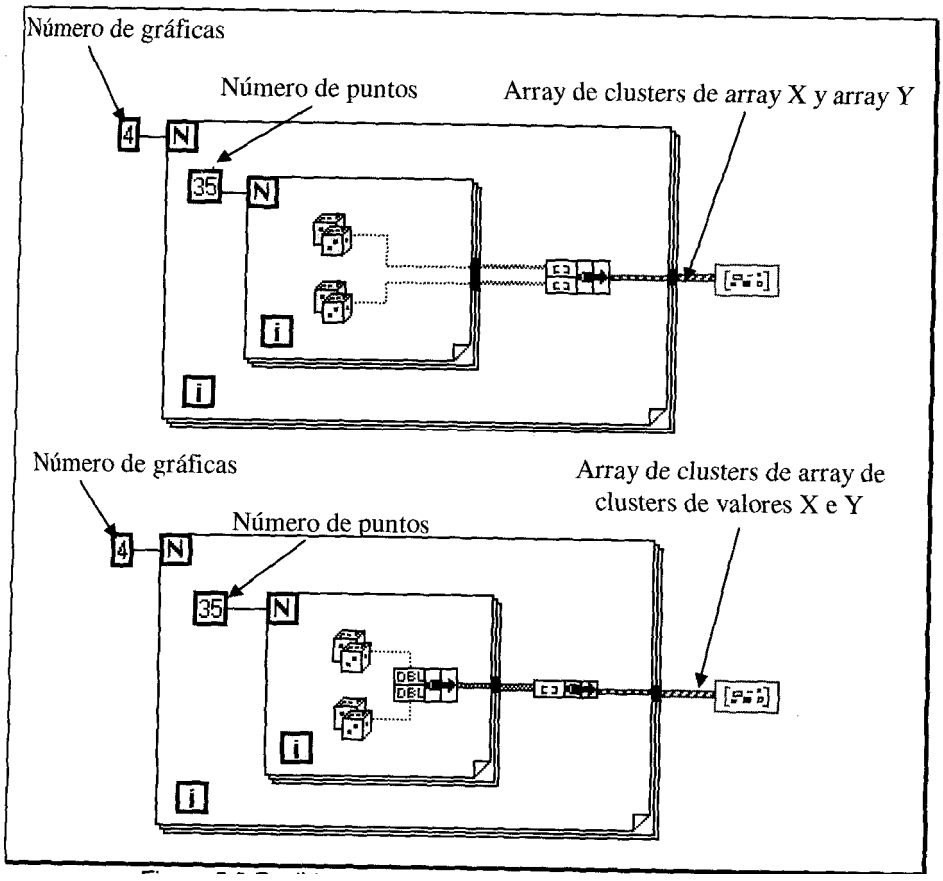


Figura 4.10 Posibles representaciones de múltiples XY Graph

GRAPH DURSORS

La paleta de cursores está disponible desde la opción *Show Cursor Display* del menú pop-up (figura 4.9).

- Nombre del cursor: Permite introducir una etiqueta de identificación del cursor. Podemos tener tantos cursores como deseemos.

- Posición X, Posición Y : Indica las coordenadas en las que se encuentra el cursor; en los indicadores *íntensíty* graph aparece también la coordenada Z. Podemos mover el cursor directamente a una posición concreta introduciendo las coordenadas del punto deseado.

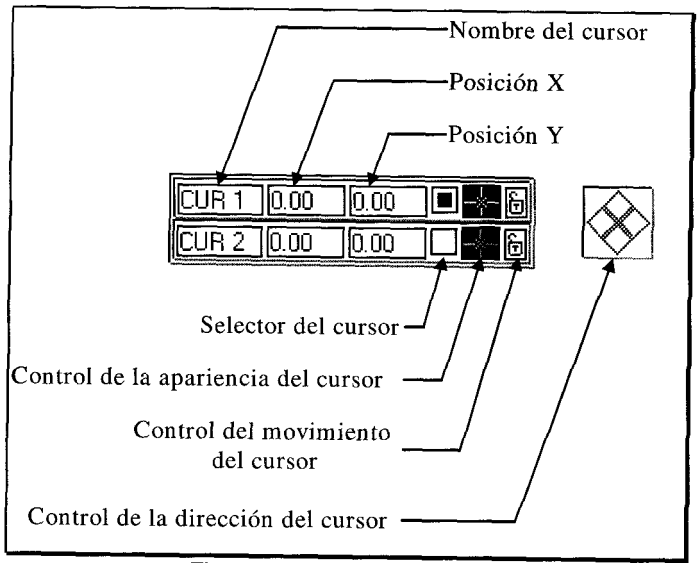
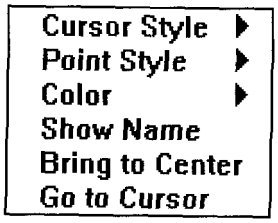


Figura 4.11 Paleta de cursores

- Selector del cursor: Selecciona el cursor a mover. Se pueden seleccionar a vez tantos cursores como deseemos.

- Control de la apariencia del cursor: Abriendo el menú mediante el bot izquierdo del ratón podemos modificar algunas características del cursor:



- Cursor Style: Selecciona la forma con la que se indica el punto sobre el cual encuentra el cursor.

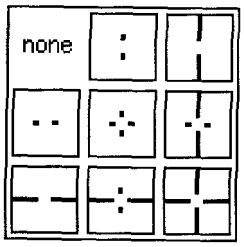


Figura 4.12 Submenú Cursor Style.

- Point Style: Selecciona el estilo del punto que marca la posición del cursor.

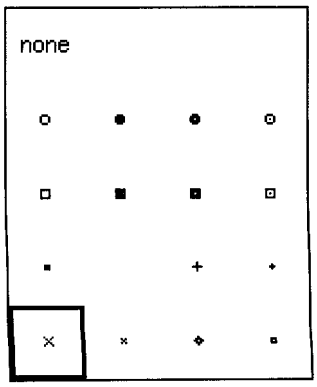
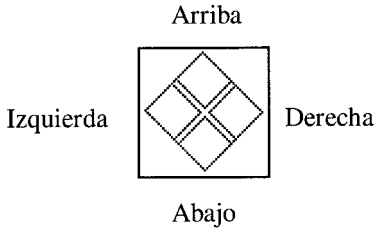


Figura 4.13 Submenú Point Style

- Color. Selecciona el color del cursor.
- 9 Show Name: Muestra el nombre del cursor sobre la gráfica.
- Bring to Center: Mueve el cursor hasta el centro de la pantalla cambiando las coordenadas de éste.
- Go to Cursor,. Modifica las escalas X e Y de forma que podamos ver el cursor, pero sin cambiar las coordenadas de éste.

- Control del movimiento del cursor: El candado cerrado indica que el cursor se moverá siguiendo la gráfica (opciones *Lock to plot y Snap to point),* mientras que el candado abierto indica que el cursor se moverá libremente (opción *Free).* Si hubiese más de una gráfica el menú nos permitirá escoger sobre cual de ellas queremos que se mueva el cursor. El comando *Allow Drag,* cuando está activo, permite desplazar la gráfica directamente con el puntero del ratón.

Control de la dirección del cursor: Mueve los cursores seleccionados punto por punto en la dirección indicada.



4.3 CONFIGURACION DEL PUERTO LPT1 A LA TARJETA DAQ

Para el presente trabajo es necesario configurar las salidas y entradas del puerto paralelo el cual como se analizo anteriormente tiene 3 registros los cuales serán configurados de la siguiente manera:

- El registro 378Hex que contiene 8 bits de salida será empleado como líneas de salida de datos para propósito general las cuales en primera instancia serán líneas para el control del ADC 0808 en la selección del canal analógico a digitalizar (A y B del 0808) el pulso de inicio de digitalización (Start y Ale del 0808) y finalmente separación de bits MSB y LSB en grupos de 4 bits cada uno para entrada de datos por el registro 379Hex. En segunda instancia estos 8 bits de salida serán utilizados para activación de 4 reles de 24Vdc (Out relé) y 4 salidas d voltaje cada una de 24Vdc

(Digital Out).Para lograr este propósito es necesario separar las líneas del bus común que se esta empleando del registro 378Hex esto se consigue mediante circuitos 74LS244 que son compuertas de tres estados (“1” lógico , “0” lógico y alta impedancia). Adicionalmente para las salidas a rele y voltaje es necesario memorizar los datos puesto que en un instante están controlando el ADC y luego provocando activación de salidas que puede producir conflictos o descoordinación en el control entonces es necesario utilizar circuitos 74LS373 (Tipo Latch) para grabar los datos de salida en especial los de activación de relés y voltaje.

- El registro 379Hex contiene 8 bits de entrada de datos pero solo 5 líneas disponibles de las cuales serán empleadas los bits 3,4,5 y 7 para entrada de datos de propósito general que en primera instancia ingresaran los bits del ADC 0808 en grupos de 4 bits cada uno (LSB y MSB) que son separados por el bit # 3 del registro 378Hex que se explico anteriormente el cual controla un circuito 74LS244 (tres estados) . En segunda instancia serán empleadas estas líneas para ingresar datos digitales de 24Vdc exteriores , esto se consigue utilizando otro circuito 74LS244 el cual es controlado por el bit # 2 del registro 37A Hex . El bit # 6 de este registro 379Hex se lo utiliza con la línea EOC del ADC 0808 (EOC=Fin de conversión) para indicar al programa (Software LabView) que se a terminado la digitalización del canal asignado y que se puede proceder a leer los adtos presentes en el registro 379Hex.
- El registro 37A Hex es un registro bidireccional (IN/OUT) que para nuestra tarjeta se lo utilizara solo como salida (OUT) y se empleara 3 bits de los 4 disponibles para líneas de control . El bit # 0 cumple la función de separar los datos de salida del bus común 378Hex como se explico anteriormente en control del ADC 0808 y salidas a relé / voltaje mediante controlar 2 circuitos 74LS244 como se indica en la figura a continuación .

El bit # 1 sirve para guardar o memorizar los datos de salida a rele y voltaje del 378Hex , esto se logra mediante producir un pulso en la línea Latch del circuito 74LS373 . El bit # 2 se emplea en la separación de lineas de los datos del ADC (bits LSB /MSB) y los datos exteriores (Digital In 24Vdc) mediante un circuito 74LS244 (tres estados).

En la figura siguiente a mas de mostrar la configuración del puerto paralelo LPT1 y su distribución o asignación de pines se indican también las fuentes de alimentación que se emplean que son de +5Vdc , +12 Vdc , -12Vdc , +24 Vdc y de la red 110VAC , tambien se muestra la asignación de pines del ADC 0808 ,el reloj (CLK) que esta formado por el circuito 555 y demás elementos para producir un frecuencia de 500 KHz , compuertas 74LS14 que son inversores smith trigger que sirven para generar formas de onda cuadradas con un mínimo de transición de alto a bajo y viceversa como es en el caso del CLK y para el control de los circuitos 74LS244, también se observan circuitos opto acopladores 3041 que nos sirven para proteger a la PC de voltajes superiores a los 5Vdc puesto que proveen aislamiento de 7000 voltios , y finalmente se observa el acondicionamiento de las senales analógicas (SCXI = Signal Conditioning eXtensions for Instrumentation) para el sensor de temperatura LM335 , utilizando para ello circuitos LM 741 que son amplificadores operacionales en varias configuraciones como seguidores de tensión , restadores , inversores , conversores de corriente a voltaje , este acondicionamiento es necesario puesto que el sensor de temperatura LM335 tiene una función de trasferencia de 10mV por grado Kelvin es decir a una temperatura de 0° grados Celsius se tendra un voltaje de 2.73Vdc. y a una temperatura de 100° grados Celsius el voltaje será de

3.73Vdc.Esto se logra convirtiendo los grados Celsius a grados kelvin mediante la ecuación:

$$°K = 273 + °C$$

Entonces si °C = 0 grados.

La temperatura kelvin es: °K = 273 grados.

Y el voltaje generado en el sensor LM 335 sera : Vs = 273 ° K*(10mV / °K)

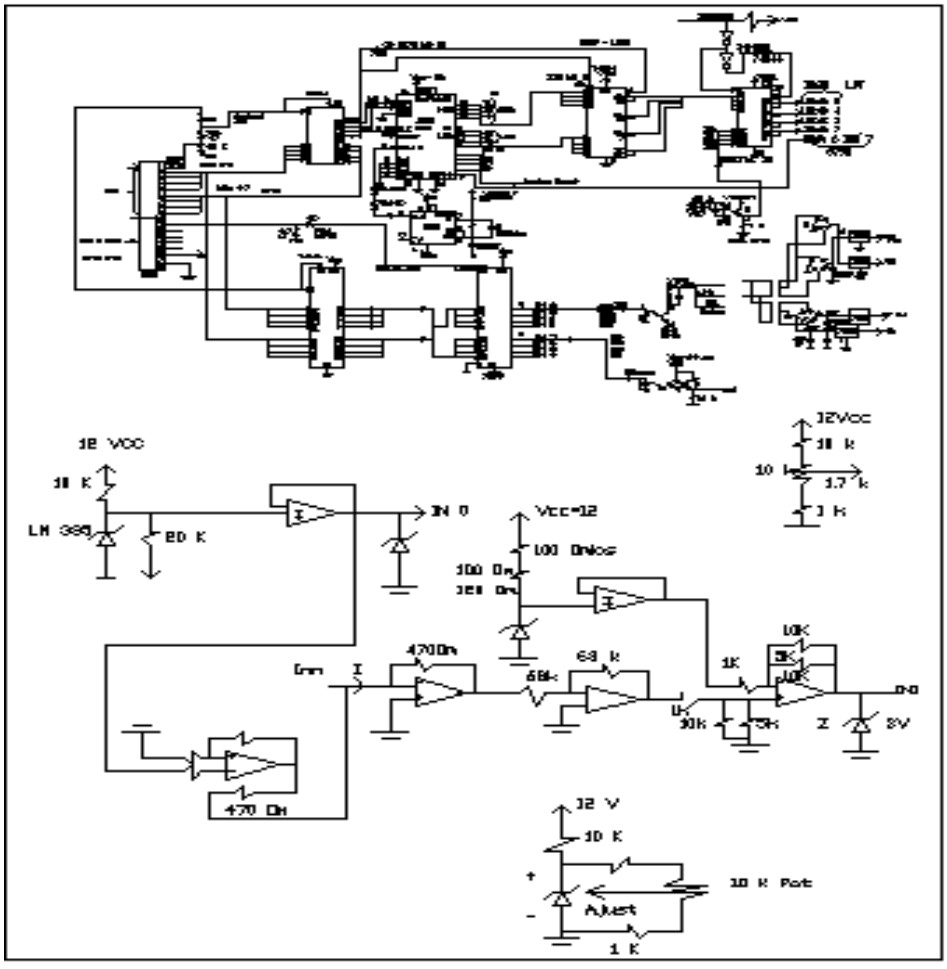
Vs = 2.73 Vcd.

Si °C = 100 grados

La temperatura kelvin será : °K = 273 + 100 , °K = 373 grados.

Y el voltaje generado es Vs = 373°K * (10mV / °K) , Vs = 3.73Vdc.

El procedimiento es similar para cualquier temperatura en grados Celsius para generar el voltaje en el sensor. NOTA (VER EN ANEXOS).



4.3.1 ASIGNACION DE PINES DE ENTRADA Y SALIDA DEL PUERTO LPT1

A continuación se describe la asignación de cada uno de los bits de los 3 registros del puerto paralelo con la respectiva función que cumple con un estado de “1 “ lógico o con “0” lógico según la descripción que se menciono arriba.

Pines	**Sentido**	**378 H bits**	**379 H bits**	**37^{A} H bits**	**Función**			
1	IN/OUT			0	**37A Bit 0 = “0”**		**37A Bit 0 = “1”**	
2	OUT	0			A Dir Ch 0808		Out rele 1 /NO/NC	
3	OUT	1			B Dir Ch 0808		Out rele 2 /NO/NC	
4	OUT	2			Start / Ale 0808		Out rele 3 /NO	
5	OUT	3			Msb / Lsb 0808		Out rele 4 /NO	
6	OUT	4			-----		Digital out 1 24Vdc	
7	OUT	5			-----		Digital out 2 24Vdc	
8	OUT	6			-----		Digital out 3 24Vdc	
9	OUT	7			-----		Digital out 4 24Vdc	
10	IN		6		EOC (fin de conversión) 0808			
Función del bit 3 del 378 H para separar lineas Msb/Lsb del ADC 0808 y bit 2 del 37A H para digital in , sobre el 379H					**378 Hex Bit 3=”0“**	**378 Hex Bit 3=”1“**	**37A Hex Bit 2=”0“**	**37A Hex Bit 2=”1“**
11	IN		7		M b7 adc	L b3 adc	Msb/Lsb	D in 3 24 V
12	IN		5		M b6 adc	L b2 adc	Msb/Lsb	D in 2 24V
13	IN		4		M b5 adc	L b1 adc	Msb/Lsb	D in 1 24V
14	IN/OUT			1	Latch 74373 =”0”		No Latch 74373= “1”	
15	IN		3		M b4 adc	L b0 adc	Msb/Lsb	Din 0 24V
16	IN/OUT			2	Msb/Lsb 0808=”0”		Digital In 24 V = “1”	

17	IN/OUT			3	------	------
18-25	GND				Referencia	Referencia

CAPITULO V

IMPLEMENTACION DE LA COMUNICACION TARJETA DAQ CON EL COMPUTADOR

Para ello analizaremos.

5.1 DISENO DEL SOFTWARE DE APLICACIÓN CON LA TARJETA DAQ

Luego de haber asignado los pines del puerto LPT 1 con la función respectiva que deben desempeñar cada uno y el momento (secuencia) y tiempo (duración) en que cada línea debe activarse o desactivarse se procede al diseño del programa bajo el software LabView de Natinal Instruments que es una herramienta diseñada como se ha explicado en capítulos anteriores para la instrumentación con el cual podemos acondicionar y controlar señales sean analógicas o digitales , realizar cálculos , controlar periféricos externos como tarjetas I/O , enlaces DDE sea a través de tarjetas Plug and Play , el puerto LPT1 , interfaces seriales como RS-232 , RS-485 todas estas operando en tiempo real , resumiendo en una sola palabra se puede construir instrumentos Virtuales con LabView es decir podemos hacer que la PC funciones o efectué las mismas operaciones que un instrumento real solo que no es tangible , pudiendo realizar como por ejemplo osciloscopios , voltímetros , controles de temperatura y cualquier variable (nivel, caudal, presión, velocidad, frecuencia, etc) y mas complejo aún supervisar y controlar plantas (Procesos productivos de una fabrica) enteras como control de motores , válvulas , análisis estadístico llamándose esto último los sistemas SCADA (supervisión control y adquisición de datos).

A continuación se procede a describir la programación realizada para lo cual es necesario recordar ciertas características que posee LabView y los datos que maneja que ya fueron descritos en capítulos anteriores pero que son necesarios tener en cuenta para comprender mejor el programa creado .

Entre las características tenemos:

- Programación totalmente gráfica (interconexión mediante iconos)
- Secuencial es decir ejecuta las operaciones condicionadas en orden jerarquico y retorna nuevamente al inicio o donde sea requerido.
- Trabaja en tiempo real con variables dinámicas exteriores.
- Puede intercambiar datos con diferentes programas como Excel, Visual Basic y otros a través de DDE (data dinamic exchange – intercambio dinamico de datos).

Entre los datos que maneja tenemos:

- Boleanos (TRUE or FALSE) que son de color verde en LabView.
- Enteros (Integer) con signo (Signed) y sin signo (Unsigned) que son de color azul que pueden ser representados en 8,16 o 32 bits (I8,I16,I32,U8,U16,U32).Además pueden expresados en formato decimal (0-9) hexadecimal (0-F) en octal (0-7) y binario(0-1).
- De precisión extendida y doble precisión (EXT ,DBL) que son de color tomate.
- Cadena de caracters tipo Ascii que son de color rosado.
- De tipo array (arreglo) colección de datos de un mismo tipo para manipulación múltiple que son de color café.

- Controles e indicadores , siendo un control aquel elemento que en el momento de correr el programa (RUN) puede el usuario manipular el valor y ser cambiado cuantas veces se requiera este se diferencia del indicador porque posee un contorno exterior mas visible o acentuado , mientras que el indicador su contorno no es acentuado es mas delgado y este sirve solo para visualización en modo RUN el usuario no puede manipular su valor, solamente un control puede variar a un indicador.

Luego de esta explicación procedemos a describir la función de cada Sub VI (progrmas locales o subrutinas del programa general).

5.1.1 DESCRIPCIÓN DEL PROGRAMA QUE CONTROLA AL ADC 0808.

Puesto que el ADC 0808 requiere de una secuencia lógica para su funcionamiento y la activación de ciertas líneas en un instante y en otro instante provoca la salida de datos ya digitalizados del canal seleccionado que igual requieren manipulación se creo el programa siguiente.

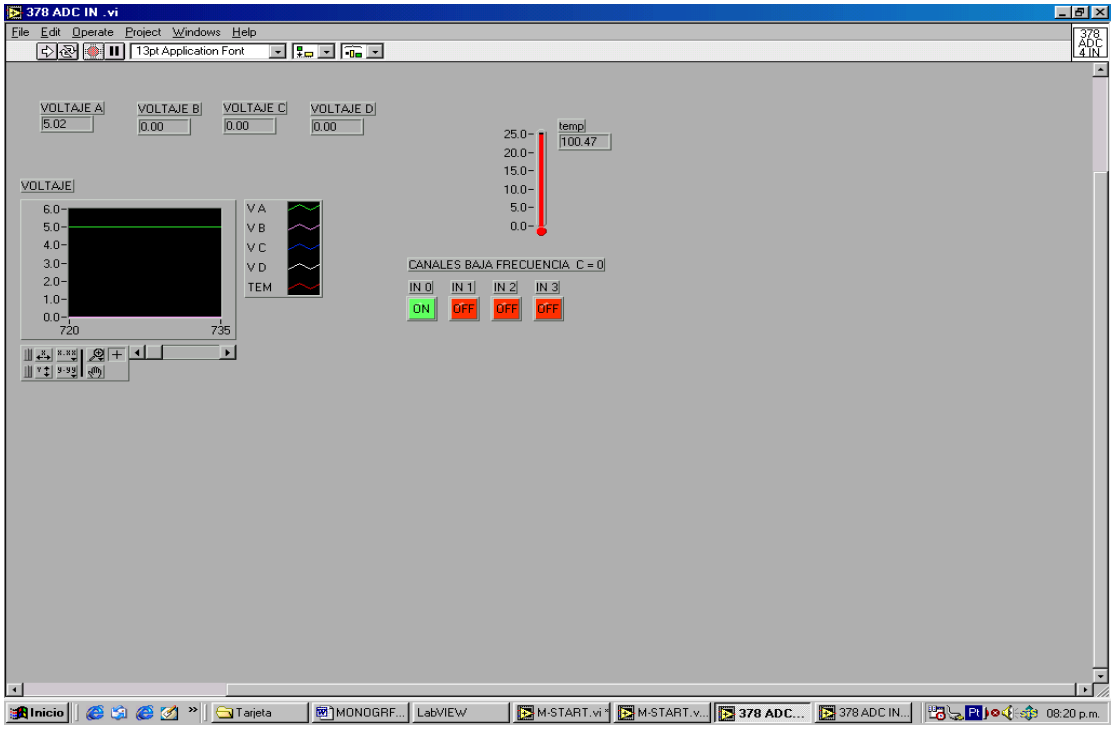


Figura 5.1

La figura 5.1 anterior es la ventana de edición en LabView del Sub VI 378ADC IN se observan los 4 canales del ADC 0808 que pueden ser seleccionados (IN 0, IN 1, IN 2 , IN 3) estos son controloes digitales (true or false), en este instante esta activado el canal IN 0 y el ADC 0808 indica una digitalización de 5.02Vdc que estan en la tarjeta exterior, la indicación de 5.02 se observa en el indicador VOLTAJE A que es un indicador de precisión extendida (EXT) también están los indicadores para los otros canales (voltaje B,C,D) ,como se logro esto en las siguientes figuras se detalla .Cabe mencionar que se puede elegir entre 1 a 4 canales simultaneos y pueden ser cualesquiera no importa el orden.

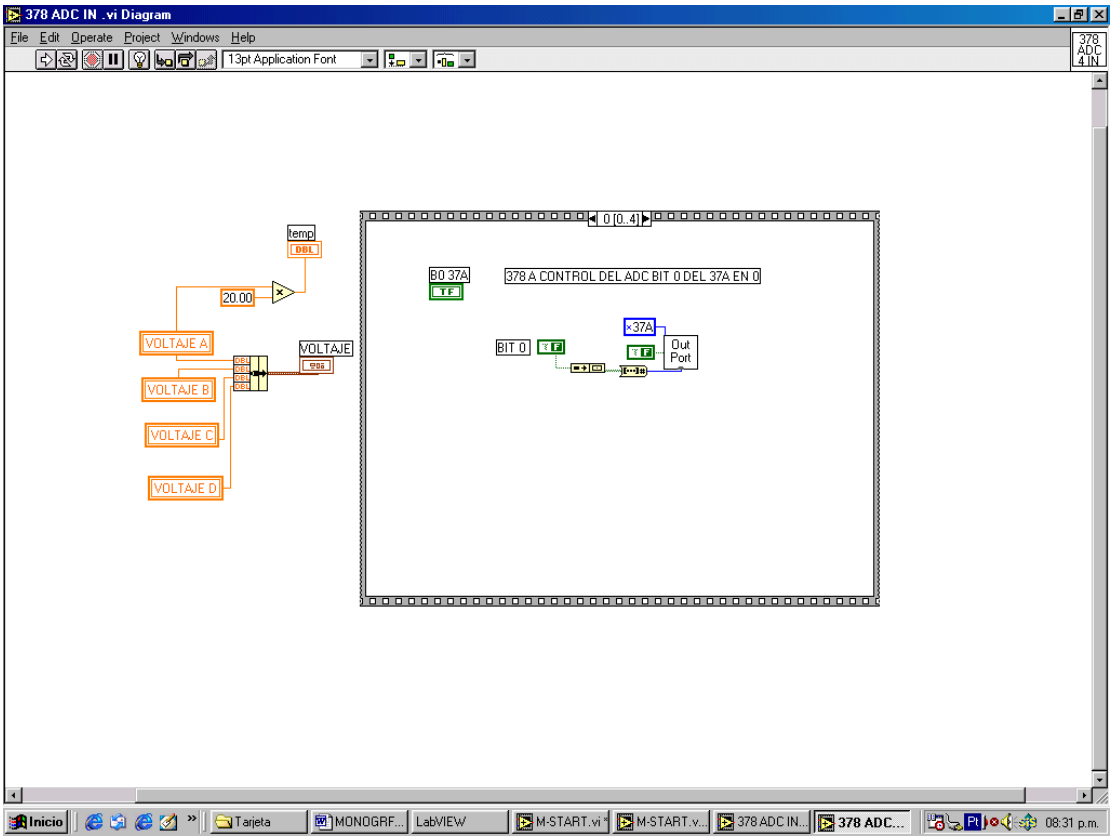


Figura 5.2

La figura 5.2 es la ventana de programación en LabView del SubVI anterior (378ADC IN) se observa los 4 canales de indicación en color tomate Voltaje A, B, C, D que son variables locales (extenciones de la variable principal) de escritura (Local Varible Write) que en este caso estan funcionando como controles (borde externo mas acentuado) a diferencia del indicador “Temp.” (borde externo no acentuado) mas adelante se indican las variables principales de voltaje A, B, C, D y se observara que son indicadores .Temp es el resultado de multiplicar Voltaje A por 20 como se observa.Voltaje es un array (color café) que en la pantalla de edición es una sub pantalla (waveform Chart) que grafica las 4 variables de precisión extendida voltaje A, B, C ,D . Tambien se observa una sentencia de secuencia que posee 5 pasos a ejecutar (0[0---4]) ,se encuentra ejecutando el paso # [0] dentro del cual se observa el Sub VI Out Port el cual es llamado para ejecutar la escritura en el puerto LPT 1 en este caso el registro (puerto) que se emplea es el 37A HEX y el dato que se esta

escribiendo el el bit 0 de este puerto es cero (“0” lógico / false) para ello es necesario emplear una conversión de boleano (0 /0/0/--) a tipo array boleano en primer lugar y luego de array boleano (00000---) a entero de sin signo de 16 bits (U 16) .

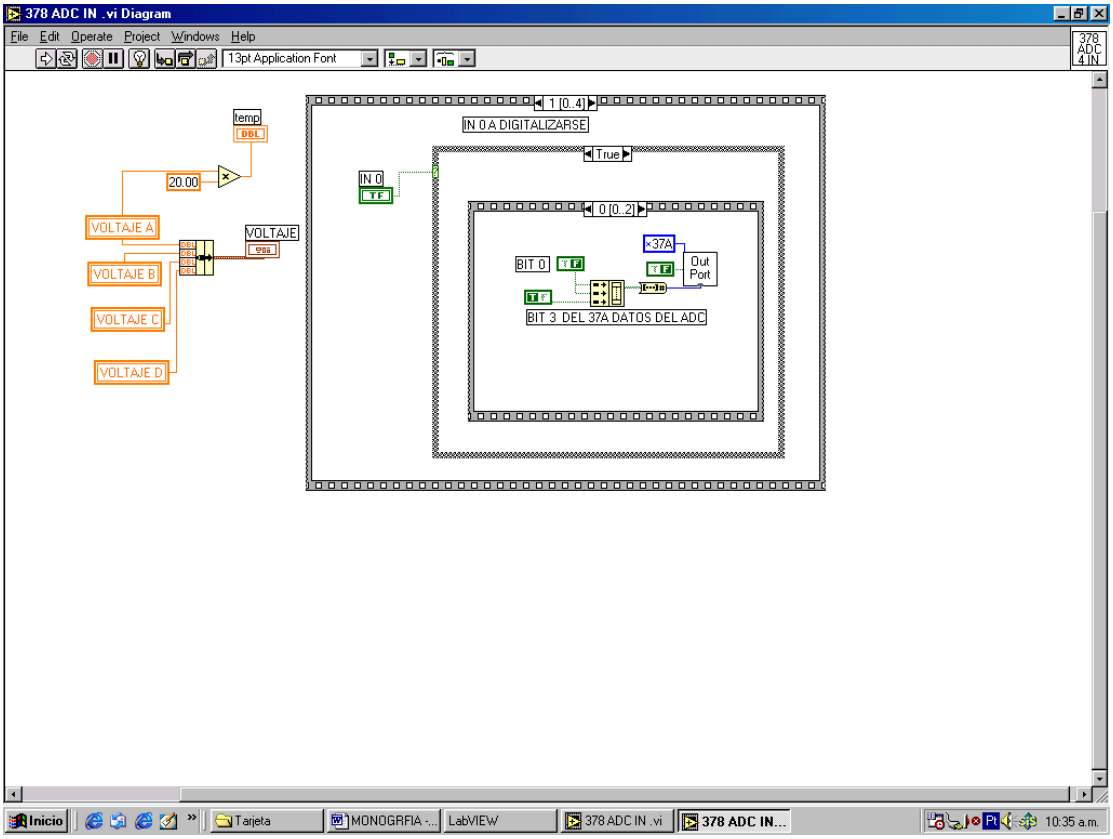


Figura 5.3

En la figura 5.3 se observa la secuencia # 1 (1[0..4]) en la cual se observan otras sentencias como por ejemplo la sentencia CASE (True / False) que es condicionada por el control boleano IN 0 y cuando es verdadero (True) ejecuta otra sentencia de secuencia de 0 a 2 (0[0..2]) , se observa en la secuencia 0[0..2] la escritura del bit 3 del puerto 37A Hex es decir el dato que se escribe en este instante es 100 en boleano y en decimal expresado en entero de 16 bits es cuatro 4 y en hexadecimal es 04.

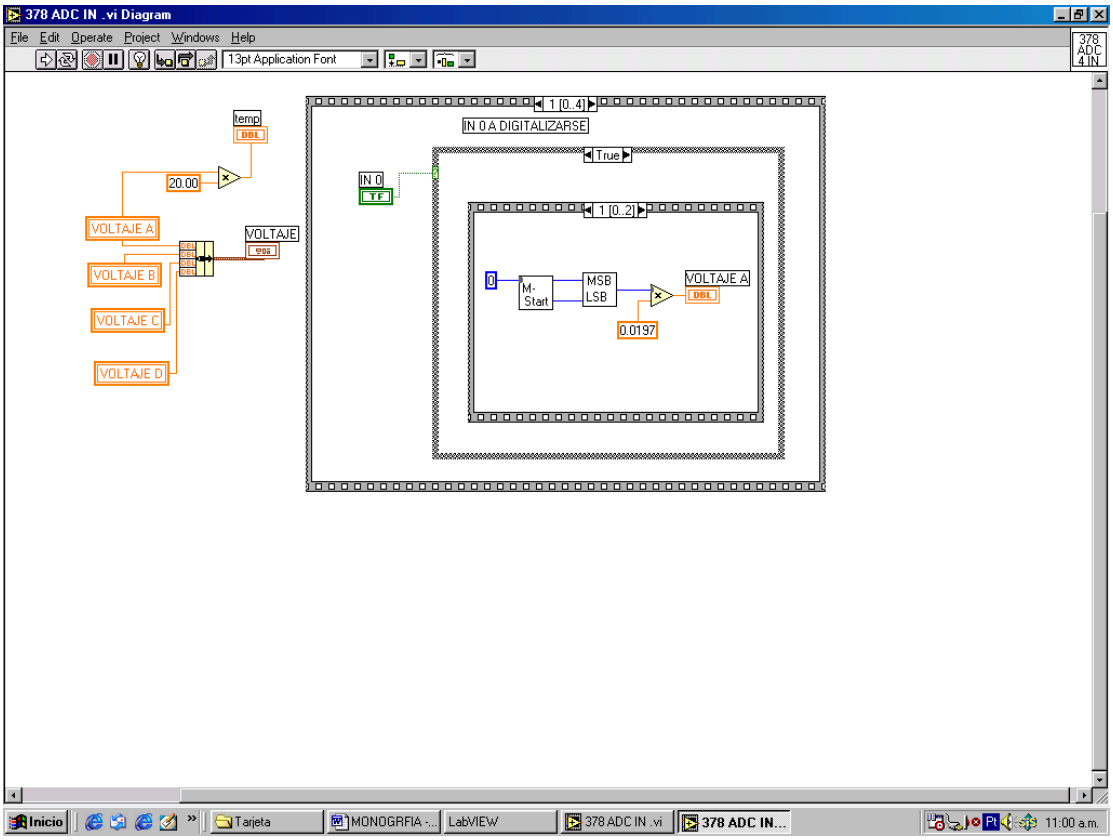


Figura 5.4

La figura 5.4 muestra la sentencia de secuencia 1[0..2] en la cual se observa que se ejecutan 2 Sub Vis el M-Start y el MSB-LSB , el sub VI M-Start requiere una entra en este caso el canal 0 y arroja 2 salidas posteriormente serán descritos que función desempeña este sub VI a igual que MSB-LSB cabe mencionar que estos 2 sub VI fueron diseñados totalmente para nuestra aplicación (propiedad intelectual) , la salida del sub VI MSB-LSB es un dato decimal de 16 bits que es multiplicado por una constante de precisión extendida (color tomate) cuyo valor es 0.0197 y finalmente escribe el valor en el indicador (variable tomate con contorno exterior no asentada) VOLTAJE A , en esta secuencia se observan tanto la variable principal Voltaje A como su variable local en modo de escritura (local variable write) que se explico en la figura 5.2. La razón de multiplicar el valor de salida del sub VI MSB-LSB por la constante 0.0197 es del calculo de convertir 0 voltios a 00 en Hex (decimal 0) hasta 5 voltios en FF en Hex (decimal 255) entonces si dividimos 5 voltios para 255 que sera

el valor máximo que arroja el ADC 0808 (1111 1111) al digitalizar los 5 voltios que es hex es FF y decimal es 255 se tiene un valor de 0.0197 es decir cada 19.7 mV (miliVoltios) la salida del ADC varia en un BIT , por ejemplo si deseamos digitalizar la temperatura ambiente con el sensor LM 335 el cual como se indico tiene una función de 10mV / grado Kelvin y arroja un voltaje de 2.93 Vcd a 20 grados Celsius la digitalización que arroja el ADC 0808 sera de 1001 0100 en boloeano en hexadecimal es 94 Hex y en decimal llega a ser 148 todo esto arroja el ADC según su resolución ,configuración y asignación de valores que se describieron en capítulos anteriores , entonces luego de ejecutarse los sub VI y después de todas estas transformaciones que estos efectúan el valor que se tiene de digitalización en decimal fue de 148 para trasformar a la escala de voltaje (0 a 5Vdc) es necesario multiplicar este valor por la constante 0.0197 y se tiene que es 2.9156 que aproximado es 2.92 VDC es decir hay una variación de 10 mv entre el valor real con el digitalizado en LabView (diferencia casi insignificante) ,el mismo proceso es par cualquier valor de temperatura a digitalizarse.

La figura 5.5 muestra la sentencia CASE en el estado de False igual es condicionada por el control boleano IN 0 y en este caso tiene que escribir en el indicador voltaje A un valor de 0 lo cual indica que no se esta empleando este canal del ADC 0808 y no se ejecutaran ninguna sentencia ni cálculos como cuando el canal es activo. Nótese que voltaje A es otra variable local ahora en modo lectura (local variable read) que a diferencia de la variable local en modo write (escritura) tiene el contorno externo no acentuado, pueden observarse las dos variables locales en este instante .

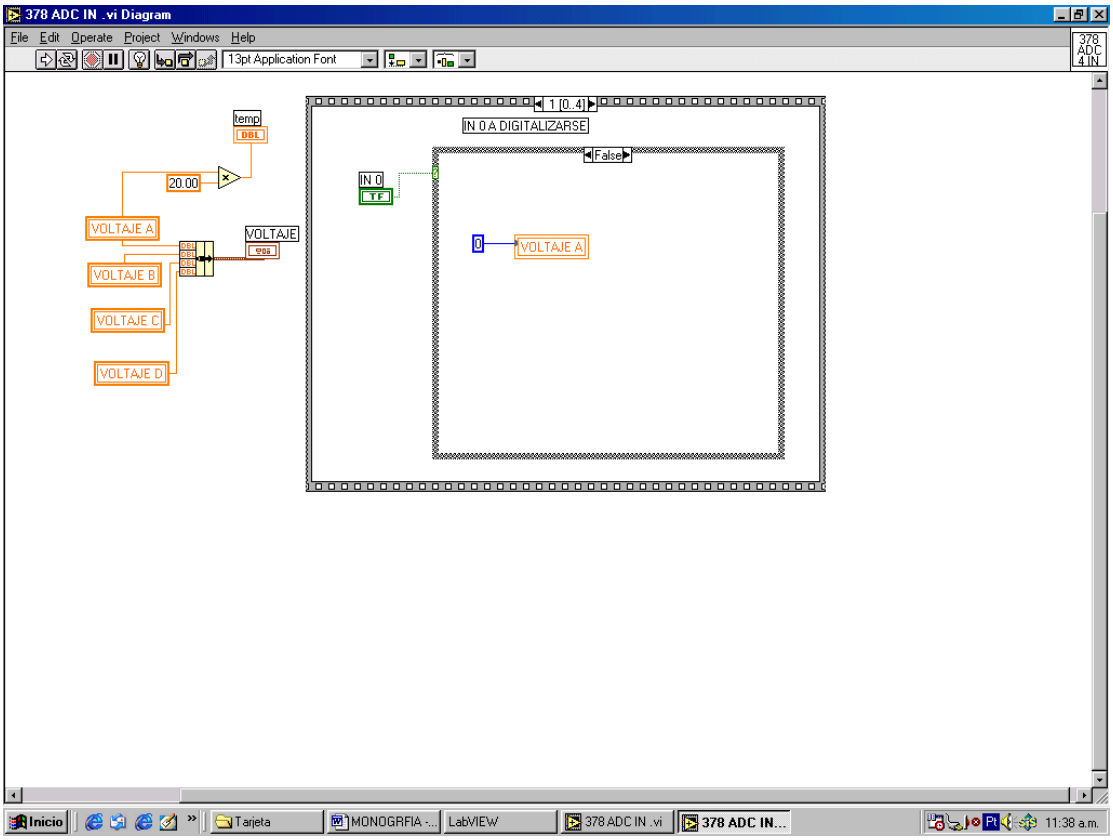


Figura 5.5

Luego de esta descripción del canal In 0 , todos los demás canales In 1 ,In 2 ,In3 siguen los mismos pasos descritos en las otras secuencias es decir en la 2[0...4] se ejecuta todo similar a lo descrito pero con las variables de controles e indicadores que corresponden al canal In 1 (voltaje B) , en la secuencia 3[0..4] se ejecuta todo con IN2 y Voltaje C , en 4[0..4] todo lo que es con In 3 y voltaje D.

A continuación se describen las funciones que efectúan los sub VI M-Start y el MSB-LSB que se indicaron anteriormente.

La figura 5.6 muestra el sub VI M START su pantalla de edición donde se observa los bits de entrada del registro 379 Hex que se utilizan en este caso son el b7, b5, b4, b3 el bit b6 es utilizado por la línea EOC del ADC 0808 y corresponde a IRQ 7 .

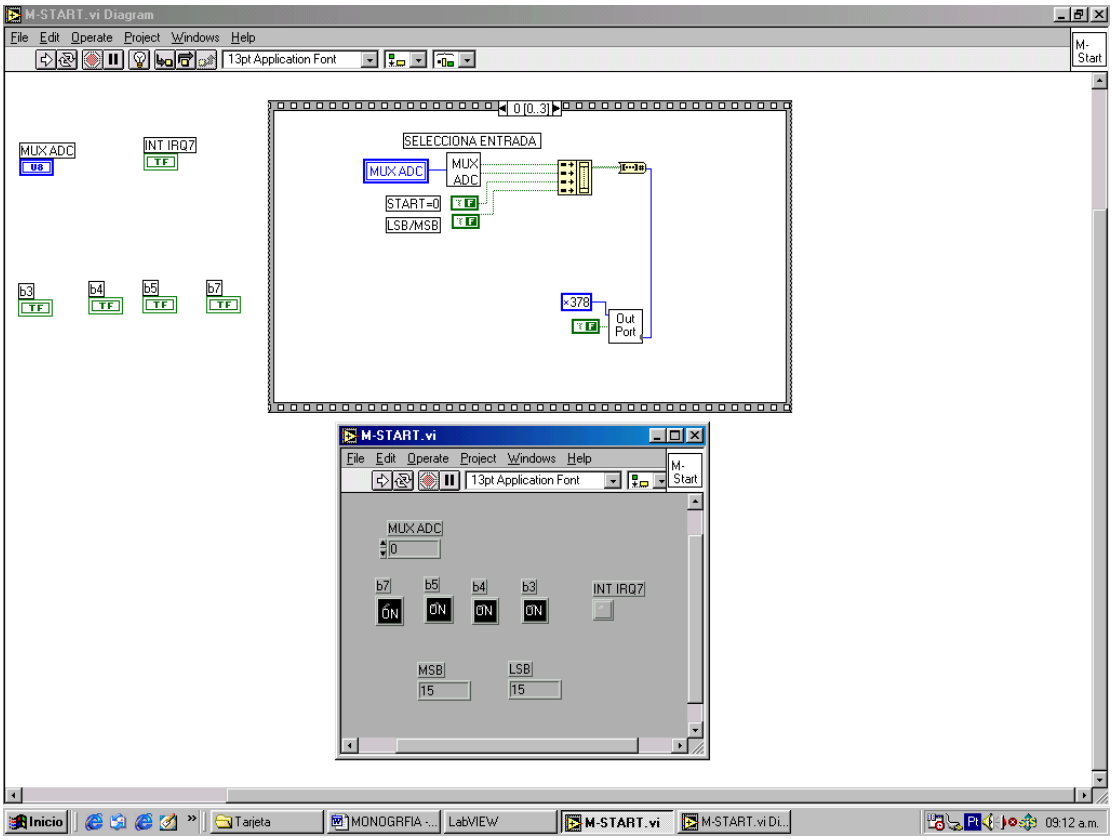


Figura 5.6

Además se observa el control Mux ADC y los indicadores de MSB LSB en estos últimos tenemos los datos ya digitalizados (15 MSB / 15 LSB) en hexadecimal en la pantalla de programación tenemos la secuencia 0[0..3] la cual activa 2 sub Vis como son el MUX ADC (diseñado para la aplicación) y el OUT PORT para escribir en el registro 378 Hex en los 4 bits ,los 2 primeros son para direccionar el canal del ADC a ser digitalizado utilizando el su VI MUX ADC, el tercero produce un pulso de START (“1” lógico) en la secuencia 1[0..3] y luego retorna a “0” en la secuencia 2[0..3] en donde se tiene un tiempo de conversión de 2 mili-Segundos en la secuencia 3[0..3] se tiene otras 3 sub secuencias como indica la figura 5.7 , se puede observar que se ha activado el bit 3 de LSB con lo cual podemos leer los 4 bit menos significativos que ha arrojado el ADC 0808 de su digitalización .

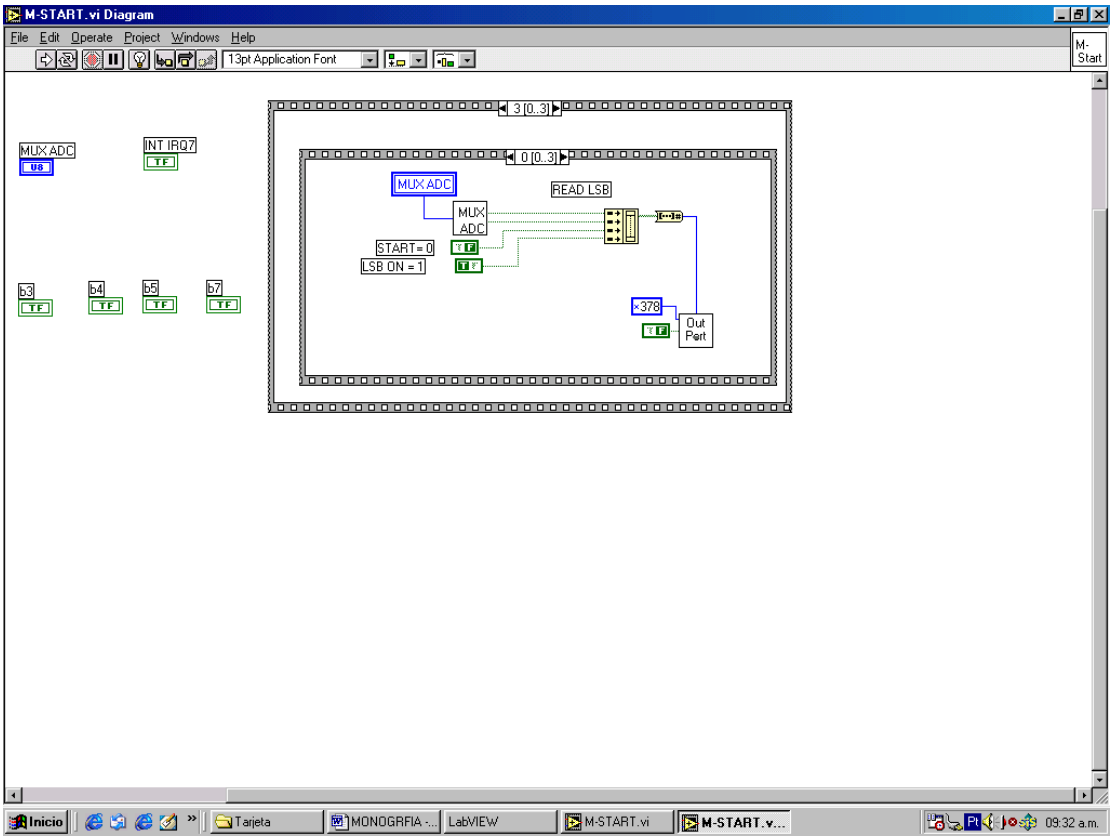


Figura 5.7

Luego en la siguiente secuencia 1[0..3] se observa la lectura del registro 379 Hex y algunas herramientas de programación que nos sirven para separar los bits que necesitamos como son el b7, b5, b4, b3 y finalmente se tiene el valor en hexadecimal en el indicador LSB como se muestra en la figura 5.8. Para leer los 4 siguientes bits mas significativos se cambia a false el bit 4 del registro 378 Hex y el proceso es similar al ya descrito.

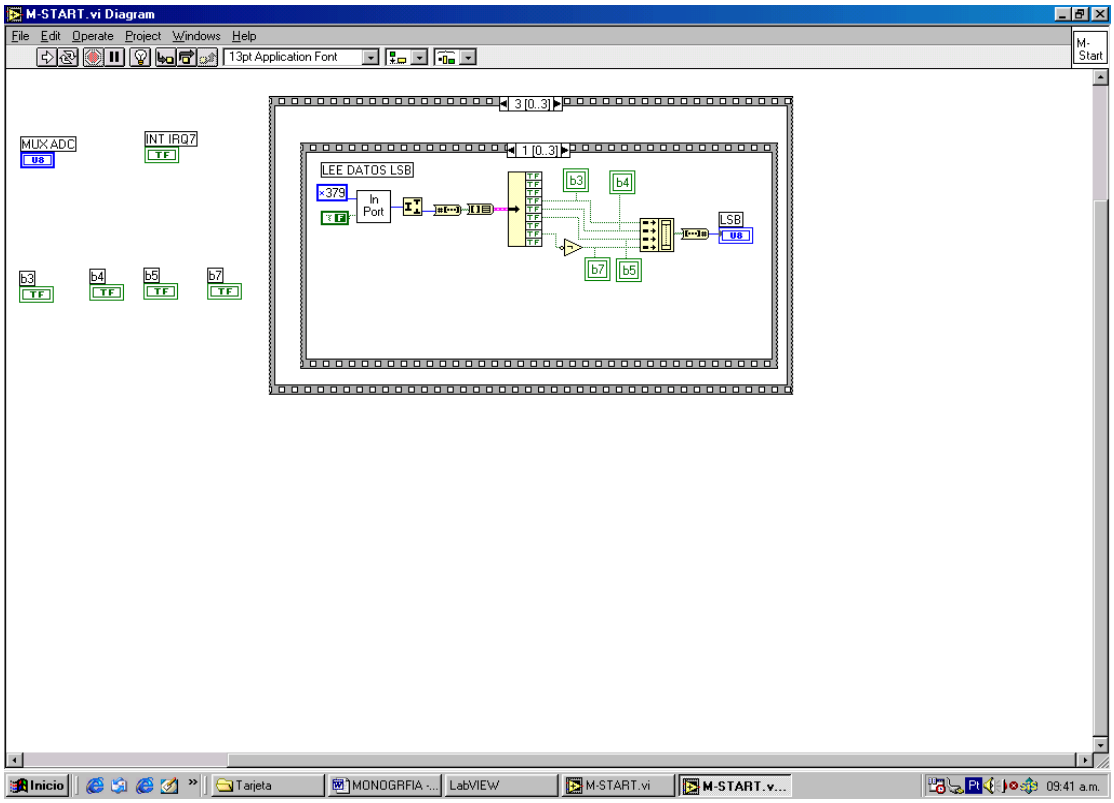


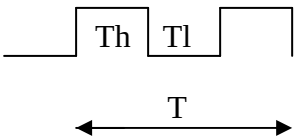
Figura 5.8

5.2 CONEXIÓN DE LA TARJETA DAQ CON EL COMPUTADOR

Para realizar la conexión de la tarjeta que contiene el ADC 0808 y demás elementos como reles de salida , voltajes de salida / entrada, y señales analógicas ,fue necesario construir un cable de trasmisión paralela con conectores DB 25 en un extremo tipo macho y en el otro extremo un tipo hembra similar al cable de impresora ,como se menciona con anterioridad es necesario la ayuda de circuitos auxiliares para direccionamiento de líneas como son circuitos 74LS244 que son compuertas 3 state es decir funcionan normalmente como las compuertas normales pueden producir estados de “1” lógico , “0” lógico y adicionalmente poseen otra línea de control que provoca un estado de alta impedancia es decir bloquea las salidas con una alta resistencia en el orden de los Mega ohms con esto podemos unir salidas de varios 74LS244 y construir un bus común como se tiene en esta aplicación en la salida de los datos MSB / LSB que arroja el ADC 0808 y al diseccionar las salidas del registro 378 Hex para

en un instante controlar al ADC y luego provocar las salidas de rele y voltaje. Toda la circuiteria interna auxiliar funciona con 5 voltios , otros circuitos funcionan con 12 VDC y voltaje negativo de 12 voltios ,rels y fuentes de salida funcionan con 24 Vdc y las entradas digitales son voltaje de 0 / 24 VDC. Tanto salidas como entradas están protegidas contra voltajes superiores a los 5 Vdc que son los niveles con los que trabaja el puerto paralelo para ello se emplea circuitos opto-acopladores 30 41 los cuales son transistores que son activados por un haz de luz que emite un diodo LED el aislamiento que producen son de 7000 voltios entonces la parte de potencia como reles de estado sólido (SSRs solid state relay) y contactores que manejan señales altas como 110 , 220 , 440Vac están completamente aislados y no existe el peligro de que se introduzcan en la PC a través del puerto paralelo. El ADC 0808 sus entradas a digitalizarse son de 5 Vdc es decir voltaje continuo y están protegidos por diodos Zener de 5 VDC con lo cual se asegura que las señales no destruyan al ADC por superar los 5 voltios . El oscilador que produce la señal de reloj (CLK) para el ADC 0808 es un circuito LM555 en configuración de Astable (Autodisparable por retroalimentación) esto se consigue mediante resistencias de 120 ohmios y un capacitor de 0.01uF (microfaradios 10 E-6) logrando con esto obtener una frecuencia de 600 KHz que es ideal para la digitalización esto se consigue mediante la ecuación que rige el funcionamiento del circuito astable que es:

T = Th + Tl (Periodo T = tiempo en alto Th + tiempo en bajo Tl)



Th = R * C * Ln 2 (R resistencia , C capacitor , Ln logaritmo natural)

Th = 120 * (0.01 10 E –6) * Ln 2

Th = 0.831776 10 E-6 segundos = 0.831776 uS (micro-segundos)

CAPITULO VI

ANALISIS DE PRUEBAS Y RESULTADOS

Se lo realiza en.

6.1 PRUEBAS DE TRANSMISIÓN Y RECEPCIÓN DE DATOS EN EL LPT1.

En el presente desarrollo del proyecto fue necesario realizar pruebas y comprobaciones como envío de datos y recepción de datos, que se detallan a continuación.

6.2 CODIFICACIÓN DE DATOS ADQUIRIDOS.

Para realizar la transmisión y recepción de los datos, la solución mas sencilla es utilizar un protocolo semejando al de la impresora. El transmisor coloca la información en la vía de datos y pulsa la línea de STROBE, para alertar al receptor; este al percibir un pulso de STROBE, lee los datos y pulsa la línea de ACKNOWLEDGE, indicando que la vía de datos está libre. Esto significa que cada computadora debe detectar dos pulsos: ACK_IN y STRB_IN. La única forma de detectar pulsos con seguridad es utilizando interrupción habilitada por flanco. En la PC hay una interrupción destinada al puerto paralelo (IRQ7) y que ya esta destinada a la línea de STROBE.

Para el envió de los datos, se necesita de dos líneas de “varias líneas”: STROBE y ACKNOWLEDGE. Usando STROBE, el transmisor avisa al receptor de que hay un nuevo dato en la vía de datos. Usando ACNOWLEDGE, el receptor avisa al transmisor de que ya leyó el dato que está en la vía de datos.

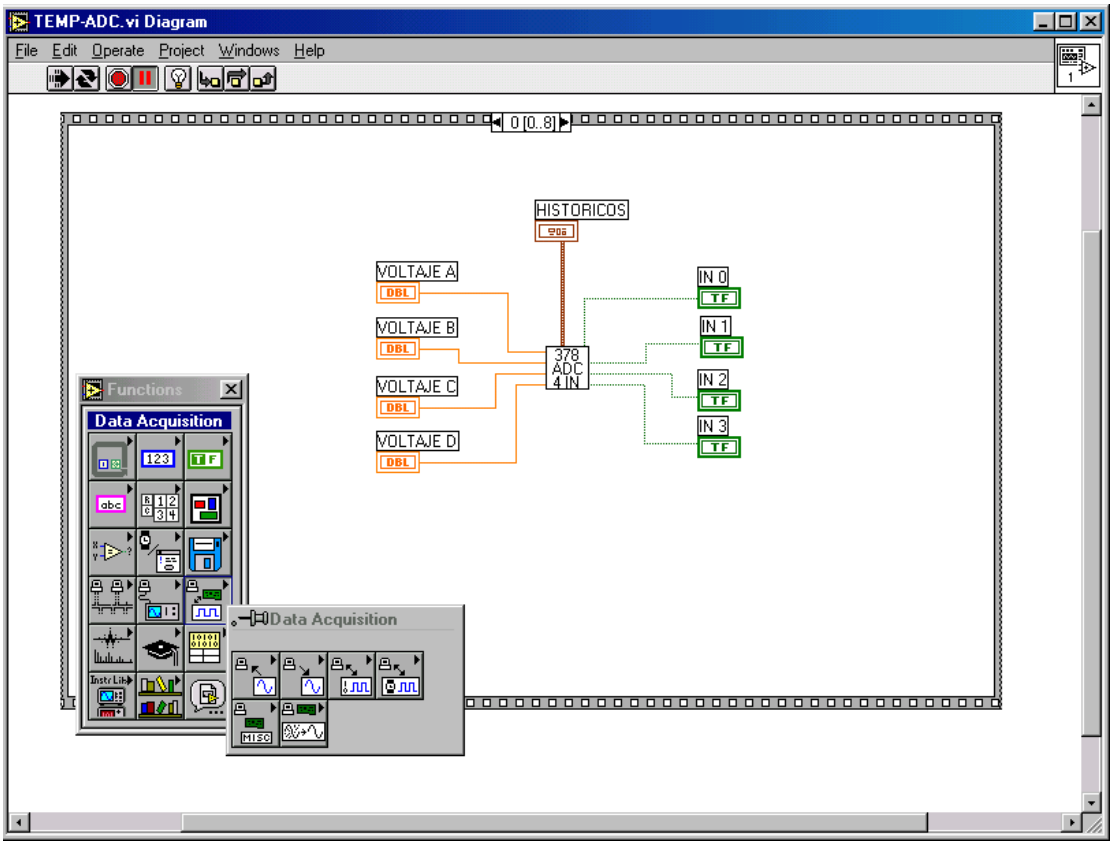
Con estos dos parámetros se inicia el proyecto del cable paralelo. El uso de una via de datos de 8 bits para cada computadora, lleva a un total de 20 líneas:

- 8 líneas para los datos a transmitir,
- 8 líneas de datos a recibir,
- 2 líneas para transmitir y recibir STROBE,
- 2 líneas para transmitir y recibir ACKNOWLEDGE.

El puerto paralelo ofrece 17 líneas:

- 8 líneas de salida,
- 5 líneas de entrada,
- 4 líneas vi direccionales (colector abierto).

El programa para la transmisión de datos fue desarrollado en el entorno de programación Lab VIEW de National Instruments el cual se ve a continuación.



6.3 CONCLUSIONES Y RECOMENDACIONES Tenemos.

A. CONCLUSIONES.

1.- Se ha logrado cumplir con los objetivos de esta tesis: El estudio de la tarjeta de adquisición de datos con sensores de temperatura y humo, así como sus aplicaciones en el control de flujo y monitoreo de señales para la activación de los instrumentos que prevengan un incendio.

2.- Con el uso de un computador, más la utilización de las tarjetas de adquisición de datos se puede crear sofisticados sistemas de adquisición y control, para implementarlos en el campo de la instrumentación o para la experimentación y estudio en laboratorios donde se necesita medir las señales análogas o digitales, también para realizar análisis estadísticos o monitoriar los comportamientos y la respuesta de los procesos.

3.- Particularmente se nota que el software labVIEW es mucho mas flexible para la adquisición, monitoreo y control de datos porque se ha probado su gran capacidad de adaptabilidad con cualesquier hardware.

4.-Ante los diferentes tipos de indicadores se plantea la necesidad de escoger entre uno u otro. Decir cuándo se debe utilizar cada uno es muy difícil ya que depende de cada aplicación y, además, puesto que en programación no hay nada imposible, podemos llegar a hacer que una gráfica simule el comportamiento de otra; sólo hace falta un poco de tiempo y paciencia

5.-Esta aplicación es una herramienta bastante útil pues permite monitorear un voltaje análogo proveniente de cualquier dispositivo electrónico desde el computador, lo que abre la posibilidad de guardar y analizar los datos e incluso tomar acciones de control desde el software.

Implementar el software en lenguaje G da la posibilidad al usuario de modificar el funcionamiento del mismo de una manera sencilla, lo que hace a la aplicación mas flexible y eficiente.

6.- Cuando se realice la programación (software) se debe tomar en cuenta la respuesta de los dispositivos, para no tomar datos erróneos.

7.- Cuando se realice la calibración de los diferentes lazos de control se debe tomar en cuenta el tiempo de respuesta de los diferentes elementos que forman dicho lazo y el tipo de respuesta de estos dispositivos.

8.- Una conclusión muy clara al finalizar el proyecto consiste en reconocer la gran utilidad que presenta el uso del computador en la actualidad, más en aquellos procesos en que el control debe tender a lo exacto y en los que el instrumento a supervisar esta muy alejado.

B. RECOMENDACIONES.

1.- Se recomienda a los usuarios que antes de efectuar el arranque del programa se verifique todas conexiones entre la PC y los módulos de procesos así como la

configuración de los mismos lo cual involucra analizar su dirección, prioridad, y demás parámetros que fueron mencionados en el capítulo II de este documento.

2.- En lo posible no debe alterarse el programa principal o uno de sus subprogramas (Sub Vis) que aparecen en la pantalla de programación de LabVIEW, porque una sola alteración o modificación hará que los controladores no respondan a los mensajes enviados o peor aún todo el sistema se desconfigurará totalmente.

3.- Cuando se utilicen las señales digitales de las tarjetas de adquisición de datos se recomienda utilizar un opto-aislador como medio para separar los niveles de referencia del sistema de adquisición de datos.

4.- En lo referente a la realización de proyectos como este consideramos que es necesario seguir ejecutándolos porque solo así se fomenta la investigación en los estudiantes y lo que es más se logra crear un criterio formado de los mismos, lo cual será beneficioso en su vida profesional.

5.- Este tipo de trabajo de investigación, se debería implementar en laboratorios o sitios donde el peligro de un incendio sea mas evidente y en un lugar no al alcance de personas sin autorización.

6.- Se recomienda utilizar un procesador de 700 Mhz o mas de velocidad para evitar lasos de perdida entre la señal y la comunicación exterior así como un mínimo de 32 Mb en RAM.

BIBLOGRAFIA.

- **Tom Sheldon**, Enciclopedia de Redes -LAN TIMES 1991.
- **THE FOXBORO COMPANY**: "Instruction Book 761CNA and 761CSA Single Station Micro Controllers", Edición 1995.
- "AC24AT/AC422AT USER´S GUIDE", [http://www.opto22.com] Edición Diciembre de 1998.
- **NATIONAL INSTRUMENTS:** "Measurement and Automation Catalogue", Edición 1999.
- Tutorial [http://members.es.tripod.de/Rendfield/tut1.html] Edición 1995.
- **SOISSOS** , "Instrumentación Industrial", Editorial Limusa México, Edición 1994.
- "USER´S GUIDE Controllers 761C Single Station" www.foxboro.com. 15 de septiembre 1999.
- **ZELENOVSKY**, RICARDO ,"IBM PC Para Ingenieros"E.S.P.E. Quito-Ecuador-1995.
- INSTRUPEDIA 99 NATIONAL INSTRUMENTS ,[http:// www natinst.com]. 15 de enero 1999
- **CREUS,** Antonio "Instrumentación Industrial" Marcombo S.A. Barcelona España -1995.
- **CURTIS**, Johnson, "Process Control Instrumentation Technology", Prentice Hall, Englewood, -1974
- National Instruments. Measurement and Automation Catalog. 2000
- National data Adquisition Databook.Nationmal semiconductor. 1995
- **Gary W Johnson 1994.** Labview Graphical programming.

- **Ronald J. Tocci** . Sistemas digitales
- **FLOYD, 97** Thomas L. Floyd. *Fundamentos de Sistemas Digitales*. Prentice Hall. 1997.
- **HAYES, 96** J. P. Hayes. *Introducción al diseño lógico digital*. Addison-Wesley iberoamericana.
- **LabVIEW for Windows,** User Manual; National Instruments; 1994.
- **LabVIEW for Windows,** Function Reference Manual, Ntional Instruments: 1994.
- **LabVIEW Data Adquisition Course Manual**, National Instruments; 1996.
- **LEWIN, 85** Douglas Lewin. *Design of Logic Systems*. Van Nostrand Reinhold. 1985.

ANEXOS.

- Anteproyecto.
- Circuito impreso para el conector de la tarjeta 1 Analogico y 2 Digital.
- Circuitos 1 y 2 de la asignación de pines I/O del puerto LPT1
- Front Panel y Block Diagram de la programación CASE en lenguaje G para el monitoreo y control del sistema automatizado con sensores en sistemas contra incendios.