ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE INGENIERÍA

"DISEÑO Y CONSTRUCCIÓN DE UN MÓDULO DE COMUNICACIÓN HALF-DUPLEX DE DATOS, PARA LOS PUERTOS SERIAL Y PARALELO DE UN COMPUTADOR, ASISTIDO POR UN PROGRAMA COMPUTACIONAL PARA EL MONITOREO Y CONTROL DE LA COMUNICACIÓN"

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO
DE INGENIERO EN ELECTRÓNICA Y CONTROL

SANTIAGO FERNANDO SANTAMARÍA CARRERA
PAVEL FERNANDO VALAREZO BASTIDAS

DIRECTOR: DR. LUIS CORRALES

Quito, Octubre 2003

DECLARACIÓN

Nosotros, Santiago Fernando Santamaría Carrera y Pavel Fernando Valarezo Bastidas, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

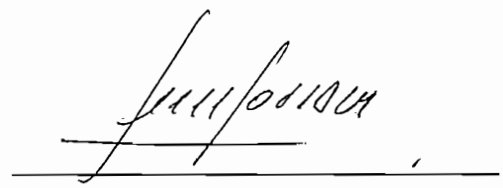
A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normatividad institucional vigente.

Santiago Fernando Santamaría Carrera

Pavel Fernando Valarezo Bastidas

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Santiago Fernando Santamaría Carrera y Pavel Fernando Valarezo Bastidas, bajo mi supervisión.



Dr. Luis Corrales
DIRECTOR DEL PROYECTO

AGRADECIMIENTOS

A nuestros amigos que desinteresadamente prestaron su ayuda .

Al Dr. Luis Corrales por su tiempo y soporte.

A todas las personas que colaboraron directa o indirectamente en el desarrollo y consecución de este proyecto.

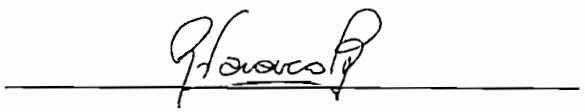
DEDICATORIA

Dedico el presente trabajo a mi Madre.

Santiago Fernando Santamaría Carrera

DEDICATORIA

Dedico este trabajo a mis Padres y Hermanos.



Pavel Fernando Valarezo Bastidas

CONTENIDO

CAPÍTULO 2

CAPÍTULO 3

CAPÍTULO 4

RESUMEN

En el presente trabajo se diseña y construye de un módulo de comunicación half-duplex de datos, para los puertos serial y paralelo de un computador asistido por un programa computacional para el monitoreo y control de la comunicación.

El módulo está construido a base de un microcontrolador PIC 16F877A, que maneja un visualizador LCD, un teclado y la interface necesaria que permite a un operador transmitir y recibir datos (half-duplex) en forma serial y paralela con un computador personal, que tenga previamente cargado el programa de pruebas de comunicación.

El programa de pruebas de comunicación se desarrolla en el lenguaje computacional con interface gráfica Visual Basic 6.0 SP5, desde el cual se podrá monitorear y establecer comunicación entre el PC y el dispositivo construido.

La comunicación se implementa bajo protocolos estandarizados, promulgados por IEEE1284 para el caso de transferencias paralelas y EIA232 para el caso transferencias seriales. Se considera el cumplimiento del handshake más no temporización.

El sistema esta restringido para la prueba de comunicación sobre computadores tipo PC o computador personal compatible con IMB, que corran bajo un sistema operativo Windows de 32bits.

PRESENTACIÓN

Hoy en día el computador es una herramienta indispensable en cuanto al monitoreo y control de procesos y variables se refiere; es así que muchos dispositivos valiéndose de interfaces estandarizadas se comunican con el PC (pórtico serial RS-232 o el paralelo), precisamente para satisfacer las necesidades antes mencionadas.

Resulta de utilidad entonces, disponer de una herramienta que permita verificar el establecimiento de comunicación entre el PC y otro dispositivo, de forma que mediante pruebas de transmisión, recepción y chequeo de bits, se garantice el correcto funcionamiento de los puertos.

Con este objetivo en mente, nuestro proyecto se desarrolla para satisfacer las necesidades de pruebas, tanto de comunicación como de señales para los puertos seriales y paralelos de los computadores compatibles con IMB, y que corran bajo sistemas operativos de 32 bits.

Se pretende entonces que con estas herramientas y con un conocimiento básico, se pueda determinar la existencia, el estado y la funcionalidad de los puertos seriales y paralelos de un computador.

CAPÍTULO 1

FUNDAMENTOS DE LA COMUNICACIÓN

1.1 SISTEMAS DE COMUNICACIÓN DIGITAL

Un sistema de comunicación digital completo es algo más complejo que solamente enviar señales que corresponden a datos binarios 1 y 0. Múltiples niveles de codificación, formato y protocolos preceden el envío de los bits de datos por el canal físico del transmisor, que en el otro extremo permiten al receptor recuperar la información original. Las ventajas de esta complejidad hacen que el sistema sea confiable, flexible y que pueda automáticamente manipular muchos problemas sin intervención de un operador.

En sistemas de comunicación digital simples, se representa los bits de datos con la presencia o ausencia de voltaje. Así un voltaje analógico de +5V es usado para representar un 1 binario; mientras que un voltaje de 0V se utiliza para representar un 0 binario. Un sistema así de simple provee un bajo rendimiento en una implementación real. Sistemas digitales prácticos realizan operaciones mucho más complejas sobre los patrones de bits, antes de que la señal sea transmitida.

Todos los sistemas de comunicación requieren un acuerdo previo entre el transmisor y el receptor. Si los dos extremos no están en acuerdo, la información transmitida carecerá de significado para el receptor; aun si el mensaje ha sido enviado correctamente. Esto se aplica tanto para sistemas análogos como para digitales.

En sistemas digitales hay muchos más aspectos que pueden ser cambiados y sobre los que se debe llegar a un acuerdo, de tal forma que estos contribuyan a un mejor rendimiento del sistema. En contraste, un sistema análogo tiene relativamente menos aspectos que pueden ser cambiados como: el ajuste del nivel de la señal, filtrado, selección de los parámetros de modulación, por mencionar los principales en cuanto a transmisión se refiere.

Las comunicaciones se desarrollan como una actividad que ocurre en múltiples niveles o múltiples capas. Así, una estructura de comunicación consta de los siguientes niveles: Protocolo, Codificación, Formato y Modulación e Interface Física.

1.1.1 MODOS DE COMUNICACIÓN

Un sistema de comunicaciones puede ser diseñado para transmitir información en *uno* o en dos sentidos.

1.1.1.1 Simplex

Un sistema simplex es aquel diseñado para enviar mensajes en un solo sentido. Esta clase de sistemas tienen un limitado interés en los sistemas de comunicación industrial, pués a menudo es requerido un canal reverso para confirmar la ejecución de una acción requerida.

1.1.1.2 Half Duplex

Las comunicaciones half-duplex se establecen cuando los datos fluyen en ambas direcciones; pero en un solo sentido a la vez. Por tanto para llevar a cabo una comunicación half duplex el receptor debe esperar a que el transmisor concluya el envío de datos para poder utilizar el canal.

1.1.1.3 Full Duplex

En un sistema full-duplex, la información puede viajar en ambos sentidos simultáneamente. Un sistema full-duplex utiliza dos canales separados para llevar a cabo la comunicación.

En esta configuración cada extremo en el canal de comunicaciones puede enviar información sin importar si el otro extremo esta enviando o no información al mismo tiempo.

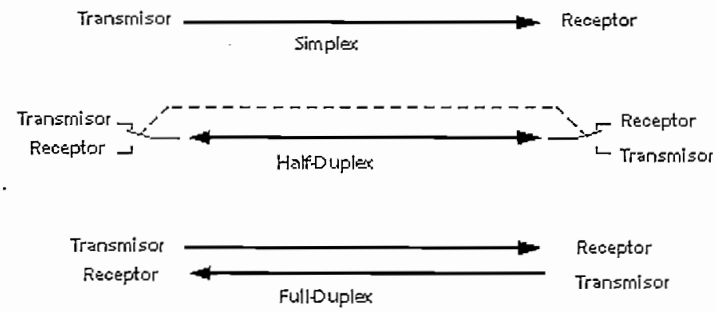


Figura 1.1
Tipos de canales

1.1.1.4 Consideraciones

Un sistema half duplex es usado en lugar de uno full-duplex, cuando hay la necesidad de un flujo de información en ambos sentidos, pero el ancho de banda, número de cables o número de canales de señal es una limitante. Por otro lado un canal full-duplex requiere un canal del doble de ancho de banda que aquel necesitado para establecer una comunicación half duplex, asumiendo que se va a transmitir la misma cantidad de información.

Por lo antes indicado se puede convenir entonces que en un sistema half duplex cada usuario comparte el mismo ancho de banda o cables, pero a diferente tiempo; mientras que en un sistema full-duplex los usuarios no deben tomar turno de envío.

Los sistemas de comunicaciones que utilizan un sistema full-duplex son más eficientes. El receptor puede en cualquier momento informar al transmisor acerca del estado de la comunicación y por tanto hacer que se tomen acciones correctivas, de ser el caso.

En un sistema half duplex, el receptor podrá dar parte del estado de la comunicación o de la información contenida en esta, solamente una vez que la transmisión haya sido realizada y el canal haya sido desocupado.

1.1.2 PROTOCOLOS

Un protocolo define las reglas que rigen la comunicación. Esto significa que un protocolo especifica cómo la comunicación inicia, termina y, lo más importante, indica cómo debe proceder ante situaciones especiales o anomalías que puedan ocurrir en el proceso. Dichas situaciones incluyen: encendido normal del sistema, recuperación ante un eventual apagado del mismo, y reacción frente a elementos externos que puedan alterarlo como el ruido. En suma, el protocolo define los pasos que deben seguir los elementos que conforman el sistema de comunicación; es decir, el transmisor y el receptor, cuando los datos son recibidos o transmitidos.

La elección del protocolo adecuado, depende del tipo de aplicación y de los datos que serán enviados, además se toma en cuenta cuán críticos son los datos y cuán importante es que sean recibidos sin errores. Así, cuando una tasa de error es aceptable es permitido el uso de protocolos relativamente simples y que no necesitan retransmisión de datos. Por otro lado, cuando los datos a enviar son críticos se necesita de un protocolo más completo que tenga la capacidad de corregir errores, inclusive bit por bit.

Los protocolos son necesarios aún cuando el objetivo principal es recabar información; es decir, recibir datos. El receptor debe tener alguna forma de indicar al transmisor que la información fue recibida y cómo esta se recibió. Por lo tanto, en este tipo de aplicaciones, los sistemas de comunicación usan canales half-duplex o full-duplex. Los sistemas simplex, donde no hay comunicación de dos vías, usualmente no utilizan un protocolo (o probablemente utilicen uno muy simple).

Los protocolos a su vez son divididos en dos clases: stop and go y continuos.
Los canales half-duplex deben usar protocolos *stop and go,* mientras que los canales full-duplex pueden usar cualquiera de los dos protocolos (*stop and go o continuo*).

Para el caso de *stop and go*, cuando se conoce que se ha recibido el fin del mensaje, el receptor determina si el mensaje tuvo o no errores; de no existir errores, este envía un mensaje de reconocimiento (ACK, Acuse de recibo) indicando que el mensaje fue recibido con éxito. Si el mensaje fue recibido con errores, el receptor envía de vuelta un mensaje de reconocimiento negativo (NAK, Acuse de recibo negativo), indiciando al transmisor lo sucedido.

Cuando un ACK es recibido por el transmisor, este procede a enviar el próximo bloque del mensaje. Por el contrario si recibe un NAK, el transmisor repite el último mensaje. Este protocolo simple de confirmación de mensajes es efectivo pero muy ineficiente en cuanto al uso de tiempo de un canal. Luego de cada bloque de mensajes, la trasmisión de la nueva información se detiene hasta que el ACK o el NAK es recibido.

En canales full-duplex se logra una mayor eficiencia debido a que estos pueden usar un protocolo continuo. Esto es, el transmisor, después enviar cada bloque de mensajes , en lugar de detener la transmisión y esperar por el ACK o NAK, envía continuamente datos hasta completar la transmisión en proceso. Solo cuando esto sucede verifica los acuses de recibo que envió el receptor. En este protocolo el receptor envía continuamente un ACK o un NAK por cada bloque recibido, mientras recibe simultáneamente el próximo bloque de bits.

Cuando el receptor detecta un error, este envía un NAK junto con el número del último bloque recibido correctamente. El transmisor, al detectar un NAK y el número de bloque, completa la transmisión actual y luego procede a transmitir de nuevo, empezando por el bloque que fue recibido con errores.

De esta forma, la mayor eficiencia del protocolo continuo se logra debido a que el sistema solo debe retransmitir cuando un error se presenta, mientras que al usar stop and go el sistema debe detenerse después de cada bloque independientemente de si ocurrió o no un error.

1.1.3 CODIFICACIÓN

El primer paso para preparar los datos para una comunicación con un sistema digital es la codificación.

La codificación se encarga de tomar los bits de datos desde la fuente y convertirlos en una forma estandarizada para su transmisión. En este punto es necesario que el transmisor y el receptor acuerden el patrón específico digital que *representará* la información a ser transmitida.

Muchos códigos diferentes son usados comúnmente para esta tarea, y cada uno provee ventajas en aplicaciones específicas.

Nonprintable Control Characters		Special Symbols	Numbers, Special Symbols	Upper-Case Alphabet		Lower-Case Alphabet	
00 NUL	10 DLE	20 SP	30 0	40 @	50 P	60 .	70 p
01 SOH	11 DC1	21 !	31 1	41 A	51 Q	61 a	71 q
02 STX	12 DC2	22 "	32 2	42 B	52 R	62 b	72 r
03 ETX	13 DC3	23 =	33 3	43 C	53 S	63 c	73 s
04 EOT	14 DC4	24 $	34 4	44 D	54 T	64 d	74 t
05 ENG	15 NAK	25 %	35 5	45 E	55 U	65 e	75 u
06 ACK	16 SYN	26 &	36 6	46 F	56 V	66 f	76 v
07 BEL	17 ETB	27 '	37 7	47 G	57 W	67 g	77 w
08 BS	18 CAN	28 (	38 8	48 H	58 X	68 h	78 x
09 HT	19 EM	29)	39 9	49 I	59 Y	69 i	79 y
0A LF	1A SUB	2A *	3A :	4A J	5A Z	6A j	7A z
0B VT	1B ESC	2B +	3B ;	4B K	5B [	6B k	7B {
0C FF	1C FS	2C ,	3C <	4C L	5C \	6C l	7C |
0D CR	1D GS	2D _	3D =	4D M	5D]	6D m	7D }
0E SO	1E RS	2E .	3E >	4E N	5E ^	6E n	7E ~
0F SI	1F VS	2F /	3F ?	4F O	5F #	6F o	7F DEL

Tabla 1.1

Caracteres ASCII

Uno de los códigos más ampliamente utilizados para representar información es el código ASCII (American Estandar Code for Information Interchange). El código ASCII asigna un campo binario de 7 bits para representar cada caracter; dando así un total de 2 ^ 7= 128 únicos caracteres que pueden ser representados. Algunos sistemas usan una versión expandida del código ASCII con 8 bits (256 caracteres).

En la convención del formato ASCII, el bit menos significativo de cada caracter es transmitido en primer lugar.

Los espacios entre palabras o números en el mensaje, son representados por el código ASCII del caracter de espacio (20H). Existe además el código llamado nulo (null, todos ceros), el cual por definición no origina ninguna acción, solamente consume tiempo; este es necesitado para propósitos de prueba y algunas veces para permitir al sistema receptor asegurar la información transmitida.

Los códigos ASCII para letras, números y símbolos de puntuación usan más de la mitad de los 128 códigos de caracteres y son llamados códigos imprimibles. Esto debido a que pueden ser impresos, o ser visibles en una pantalla terminal de un computador.

Los códigos restantes de la totalidad de los 128 caracteres, son llamados códigos no imprimibles o códigos de control. Estos son símbolos ASCII que originan acciones específicas pero no imprimen un caracter tangible en la pantalla o en un papel. Los códigos no imprimibles indican acciones como alimentación de línea, cambio, fin de línea, etc. Estos son usados para controlar las actividades de un equipo y son muy importantes para una operación adecuada del sistema.

Cuando la totalidad de datos son numéricos no es necesario el uso de códigos ASCII. En este caso es preferible el uso de códigos binarios. La ventaja de usar código binario radica en que pocos bytes son necesarios para enviar un número, comparados con el envío del mismo número en código ASCII. Esta es la razón de que algunos sistemas que transmiten datos numéricos usen el código binario directamente. Sin embargo, este sistema estará limitado al envío de mensajes con datos solo numéricos; esto hace que se presenten dificultades cuando se prueba, se configura y se mantiene un sistema de comunicaciones, pues algunas veces es necesario también enviar letras, caracteres de comunicación o palabras *como* parte del mensaje.

Los sistemas de comunicación normalmente no usan los valores binarios directos, en su lugar usan el código ASCII o códigos similares para transmitir información numérica.

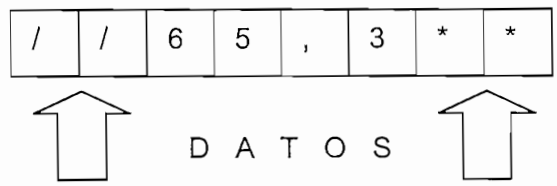
La principal aplicación para los valores binarios es donde la tasa de transferencia y eficiencia son críticas, o donde la cantidad de datos a transmitir es conocida y no requiere cambios.

La mayoría de terminales de computadores son diseñadas para aceptar la representación de caracteres ASCII. El terminal recibe el patrón de 7 bits y despliega el carácter correspondiente en pantalla.

1.1.4 FORMATO

El formato define la información adicional que necesita el mensaje para ser transmitido, recibido y entendido. El rango de los formatos utilizados varía desde formatos muy simples hasta formatos muy complejos. La elección de un formato depende de la necesidades de diferentes aplicaciones.

Un ejemplo de transmisión con formato simple es una lectura de voltaje enviada desde un dispositivo al computador. Cada lectura de voltaje empieza con uno o más caracteres ASCII de inicio, seguido del valor, y finaliza con uno o más caracteres ASCII de terminación. De esta forma, el valor de voltaje 65,3 puede ser transmitido de la siguiente forma: //65,3**



Delimitador Inicio

Delimitador Terminación

Figura 1.2
Formato Simple

Un formato más avanzado define a la totalidad del mensaje como trama. Una trama genérica tiene secciones denominadas campos, y cada campo esta formado por bytes. Los nombres de los campos son los siguientes: campo de inicio de trama, campo de dirección, campo de longitud / tipo, campo de datos, campo de secuencia de verificación de trama y campo de fin de trama.

El campo de inicio de trama le(s) indica a el(los) dispositivo(s) receptor(res) que se va a enviar una trama de datos. Esta compuesto por una secuencia de bytes de inicio y señalización.

El campo de dirección contiene la información de denominación, como por ejemplo, el nombre del dispositivo origen y el nombre del dispositivo destino.

El campo de longitud / tipo especifica la longitud exacta de la trama a enviar y su tipo.

El campo de datos contiene el paquete de datos que se desea enviar. Junto con estos datos, también se deben enviar bytes adicionales llamados bytes de relleno, que a veces se agregan para que las tramas tengan una longitud mínima con fines de temporización.

El campo de secuencia de verificación de trama contiene un número calculado por el dispositivo origen basado en los datos de la trama. El dispositivo destino al recibir la trama calcula también este número; si los dos números son distintos, se da por sentado que ha ocurrido un error, se descarta la trama y se pide al origen que vuelva a realizar la transmisión.

El campo de fin de trama está conformado generalmente por una secuencia formal de bytes que se denomina delimitador de fin de trama e indica donde termina la trama.

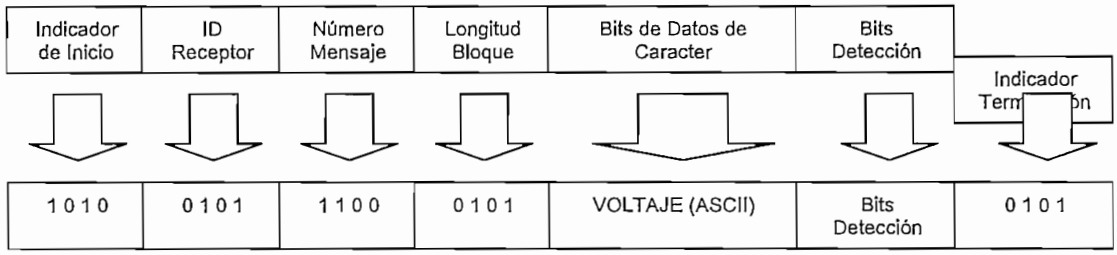


Figura 1.3
Trama

1.1.5 MODULACIÓN E INTERFAZ FÍSICA

La Modulación se encarga de transformar los niveles de voltaje presentes, en una señal compatible con el canal de forma que los datos puedan transmitirse.

1.1.5.1 Interfaz Física

Sin importar el número de niveles en un sistema de comunicación y el procesamiento llevado a cabo en cada nivel, la señal debe ser alimentada al enlace del sistema. El enlace en la forma de un cable, fibra óptica o antena es llamada la capa física. Esta es la capa más tangible de un sistema de comunicación, pues en este punto una señal puede ser fácilmente observada o medida.

Esta es una capa muy crítica para el rendimiento total del sistema, pues la calidad de la señal debe ser mantenida dentro de ciertos parámetros, de forma que el receptor sea capaz de recuperar los bits de información sin error. A diferencia de las actividades de las otras capas de comunicación, las cuales son completamente internas al sistema y se encargan de manipular los bits de datos usando circuitería de hardware y además software, esta capa representa el canal físico externo y el procesamiento que debe hacerse a los datos para que puedan acoplarse y viajar por el mismo.
La capa física es encargada de llevar a cabo la modulación, que hace que la señal que representa la información sea compatible con el enlace usado.

La comunicación física más simple usa dos valores de voltaje para representar los bits de datos. Usa 0V para representar un 0 binario, y +5V para representar 1 binario (esquema unipolar). Este método es efectivo cuando las distancias entre transmisor y receptor son cortas, y tanto el transmisor como receptor comparten el mismo chasis y el mismo cable de referencia. La ventaja de esta interface es que se necesita una circuitería adicional muy simple para poner la señal digital en el cable, o recuperar la señal desde el receptor.

Las compuertas lógicas digitales estándar no son una buena elección para circuitería de la interfaz física. Su circuitería de salida no tiene la capacidad de manejar un cable y su capacitancia. Aún a bajas tasas de transferencia, las salidas de las compuertas no están protegidas en contra de cualquier clase de problemas eléctricos que ocurren cuando la circuitería está conectada a otros circuitos, como por ejemplo: conexión / desconexión mientras la energía está conectada, cortos a la fuente de poder e interferencias.

Para atender estos problemas, circuitos digitales especiales llamados manejadores de línea y, correspondientemente, receptores de línea, son usados como las interfaces físicas entre la circuitería digital y el cable.

Los manejadores y receptores de línea toman las señales de voltaje y las mantienen adecuadas para la comunicación.

Adicionalmente al esquema unipolar se usan voltajes bipolares. Típicamente estos son de +5 y –5 V o +15 o –15 V, aunque algunos sistemas pueden utilizar otros valores. En este caso los manejadores y receptores de línea deben primeramente convertir los datos a voltajes bipolares y proveer capacidades de manejo y protección necesarias cuando se realizan las interfaces a cables.

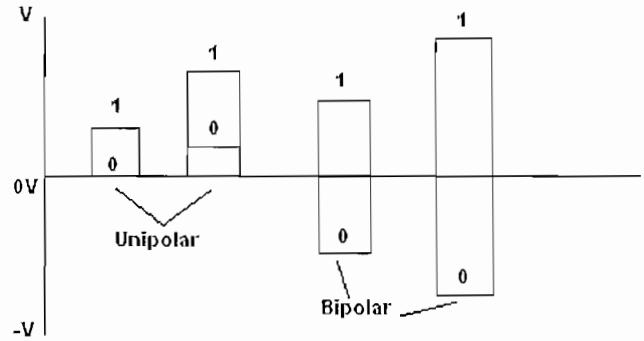


Figura 1.4
Señales unipolares y bipolares

1.1.5.2 Niveles DC

Al usar señales unipolares o bipolares por lo general existe un igual número de ceros y unos. Por lo tanto el voltaje medio será el promedio de los valores de voltaje usados para representar cero y uno binario. Por ejemplo para un sistema unipolar 0V y + 5V, el voltaje promedio es 2.5V, mientras que para un sistema bipolar se tendrá un valor promedio de 0 voltios. Este voltaje medio es llamado voltaje dc, o componente dc de el dato.

Muchos enlaces de comunicación no permiten un valor medio diferente de cero, lo que obliga a la inserción de capacitores en la línea que, a su vez, introducen problemas con la constante de tiempo que puede distorsionar los bits.
El formato digital sin retorno a cero NRZ es el mas sencillo y se caracteriza por una señal alta y una señal baja (a menudo +5v a +3.3V para un 1 binario y 0 V para un 0 binario).

El formato digital Manchester es mas complejo, pero es inmune al ruido y es mejor para mantener la sincronización. En este caso los bits se codifican como transiciones; el 0 binario se codifica como una transición de bajo a alto y el 1 binario como una transición de alto a bajo. Dado que tanto los ceros como los

unos dan como resultado una transición en la señal, el reloj se puede recuperar de forma eficaz en el receptor.

1.1.5.3 Tasas de Datos

El bit rate de un enlace corresponde al número de bits por segundo transmitidos o recibidos por unidad de tiempo, usualmente expresados como bits por segundo (bps). Baud rate es el número de eventos posibles, o transiciones de datos, por segundo. Los dos valores son a menudo idénticos, debido a que en la mayoría de enlaces cada periodo de transición representa un nuevo bit, por lo que la tecnología industrial utiliza cualquiera de los dos términos como sinónimos, a menos que se especifique explícitamente lo contrario.

Las tasas de bits que pueden implementarse dependen de muchos factores, tales como: cuanta información va a ser enviada, el tipo del canal de comunicación, la complejidad de la circuitería, y la tasa de error aceptable. Señales transmitidas a bajas tasas de velocidad, son más tolerantes a problemas de tiempo y ruido, y requieren menos ancho de banda del canal. Pero por el otro lado requieren mayor tiempo para enviar una cantidad de información.

Los valores estándar utilizados en la industria son los siguientes: 300, 600, 1200, 2400, 4800, 9600, 19600 y 28800 baudios.

1.2 SISTEMAS SINCRÓNICOS Y ASINCRÓNICOS

Uno de los problemas que se presenta cuando se transmiten señales binarias es el de la sincronización entre el transmisor y el receptor. Se han implementado para resolver este problema dos tipos de sistemas de comunicación: sincrónicos y asincrónicos. De estos el preferido suele ser el asincrónico, que no exige la adición de una línea extra que lleve la señal de reloj para sincronizar la transmisión y recepción. Sin embargo, algún modo de sincronización se requiere.

1.2.1 SISTEMA SINCRÓNICO

En una transmisión sincrónica, todos los dispositivos utilizan un reloj común generado por uno de los dispositivos o una fuente externa de reloj. El reloj puede tener una frecuencia fija o la puede cambiar a intervalos regulares. Todos los bits transmitidos son sincronizados con el reloj.

Cada bit transmitido es válido en un definido tiempo después de la transición del reloj; esto es en el borde de subida o bajada de la señal de reloj. El receptor entonces utiliza las transiciones del reloj para saber cuando los datos son válidos y por tanto leer los bits entrantes. Los detalles exactos del protocolo pueden variar. Por ejemplo, un receptor puede leer los datos entrantes en el borde de subida o bajada del reloj, o ante la detección de un nivel lógico alto o bajo del mismo. El receptor no leerá nuevamente el canal de datos, sino hasta que un nuevo pulso de reloj sea recibido.

Las interfaces sincrónicas son útiles en enlaces cortos, con cables de 15 pies o menos, o entre componentes que están en una sola tarjeta. Para enlaces que requieren mayores distancias, los formatos sincrónicos no son prácticos debido a que necesitan transmitir la señal de reloj, lo que implica una línea extra, que encarece la conexión y puede estar sujeta a ruido.

En términos de tasas de datos y protocolos, un sistema sincrónico provee un alto rendimiento. Este toma un conjunto de caracteres y los envía como un bloque continuo, manejados por un protocolo. La longitud de los bloques típicamente es de 64 caracteres (512 bits si cada carácter tiene un código de 8 bits). Además de este bloque de datos se añade un preámbulo y un campo de terminación. El objetivo de un diseño sincrónico es mantener los datos transmitidos a altas velocidades sin ningún tiempo muerto entre caracteres. La eficiencia del sistema esta determinada por la relación entre el número de bits que corresponden a los datos sobre el número total de bits. Normalmente el preámbulo y el campo de terminación son pequeños por lo tanto la eficiencia normalmente es alta.

Para asegurar una consistencia interna, algunos sistemas sincrónicos usan un bloque de longitud fija con el mismo número de caracteres para cada transmisión. De no haber nuevos caracteres para enviar, el protocolo llena con caracteres nulos el campo de datos.

A medida que más datos son generados, el próximo bloque será enviado con muy pocos caracteres nulos y más reales.

En un sistema sincrónico no existe tiempo de inactividad entre los caracteres; sin embargo hay un tiempo entre bloques, y este tiempo varia de acuerdo al diseño del sistema.

Los protocolos con sistemas sincrónicos son usualmente más avanzados. Estos son capaces de enviar más datos y de mantener el flujo de datos, el protocolo debe manipular muy cuidadosamente errores y ruido.

La circuitería para protocolo sincrónico es más compleja pero existen circuitos integrados estándar (IC) que contienen las reglas de protocolo y estados y lo implementan automáticamente.

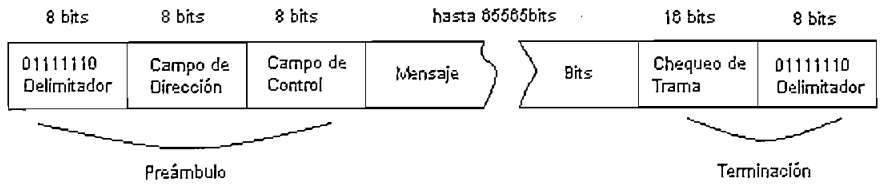


Figura 1.5
Diagrama Sincrónico

1.2.2 SISTEMA ASINCRÓNICO

Un sistema asincrónico es diseñado para enviar los caracteres de un mensaje al bit rate especificado, sin ninguna relación de tiempo fija o convenida entre cada carácter.

Este sistema es muy útil cuando la fuente transmisora genera caracteres a intervalos esporádicos, sin un intervalo preciso. Así, a medida que los caracteres son generados, estos son codificados y luego transmitidos.

Debido a la naturaleza esporádica de los caracteres asincrónicos, la implementación de un protocolo asincrónico es relativamente simple. La transmisión asincrónica no necesita de una línea de reloj adicional que mantenga la sincronía en la comunicación, pues cada extremo en el enlace de comunicación provee su propio reloj. El transmisor y receptor acuerdan previamente una frecuencia de reloj y todos los relojes del sistema deben coincidir dentro de un determinado porcentaje. Cada byte transmitido incluye un bit de inicio (Start) para sincronizar los relojes, y uno o más bits de parada (Stop) para indicar la finalización de la transmisión de un caracter.

El formato más común en una transmisión sincrónica es 8,N,1 donde el transmisor envía cada byte de datos como un conjunto de bits que incluyen: 1 bit de inicio, 8 bits de datos (empezando por el bit menos significativo) y 1 bit de parada. El símbolo N indica no chequeo de paridad.

En otros formatos se incluye un bit de paridad. Esta es una forma simple de añadir chequeo de errores. La paridad puede ser: par , impar, marca y espacio.

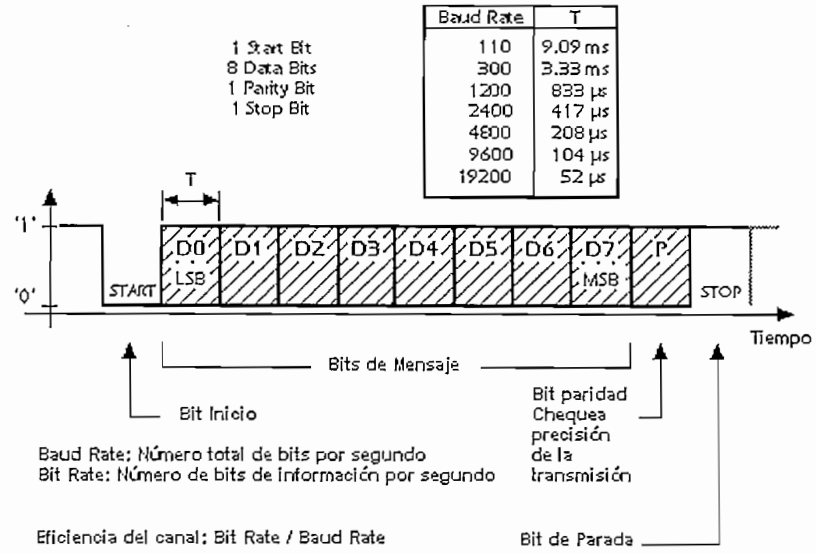


Figura 1.6
Diagrama asincrónico

Si se elige paridad *par*, el bit de paridad será fijado a uno o a cero, de forma que la suma de los bits de datos más el bit de paridad, arroje un número par de unos (1).

Al elegir paridad *impar*, el bit de paridad será fijado a uno o a cero, de forma que la suma de los bits de datos más el bit de paridad, arroje un número impar, de unos (1).

Si no se recibió el valor esperado, el receptor examina el dato recibido e informa al transmisor que hubo un error.

Paridad de *marca* y *espacio* son otras formas de paridad. Con paridad de *marca*, el bit de paridad es siempre uno, y con paridad de *espacio* el bit de paridad es siempre cero. Estos formatos son menos útiles como indicadores de error.

Otros formatos menos comunes aún, usan diferentes números de bits de datos. Algunos protocolos seriales permiten formatos de 5 hasta 8 bits de datos, además del bit de paridad.

El formato de transmisión 8-N-1 fijado a un bit rate de 9600 bps, transmitirá eficazmente 960 bytes por segundo. Esto se explica al analizar que este formato añade un bit de inicio y un bit de parada los cuales incrementan el tiempo de transmisión de cada byte. Es decir, se toma en cuenta que en total se transmitirán 10 bits para poder representar un solo byte de información.

Si el receptor requiere un tiempo extra para leer los datos recibidos, el transmisor puede extender el bit stop a un ancho equivalente a 1.5 o 2 bits.

En suma, el formato asincrónico provee una simple y conveniente manera de enviar datos a bajas tasas sin necesidad de un protocolo complicado.

1.3 LÍNEAS BALANCEADAS Y NO BALANCEADAS

La elección entre líneas balanceadas y no balanceadas es una importante consideración al seleccionar un sistema de comunicación.

1.3.1 LÍNEAS NO BALANCEADAS

Una señal de voltaje no balanceada se compone de dos conductores donde uno de ellos es encargado de llevar la señal de voltaje y el segundo conductor es el cable de señal común, llamado también señal de tierra. La señal transmitida es la diferencia de voltaje entre el conductor de señal y el conductor común de referencia.
Al usar este tipo de líneas, el conductor de tierra ó común es compartido por otros circuitos.

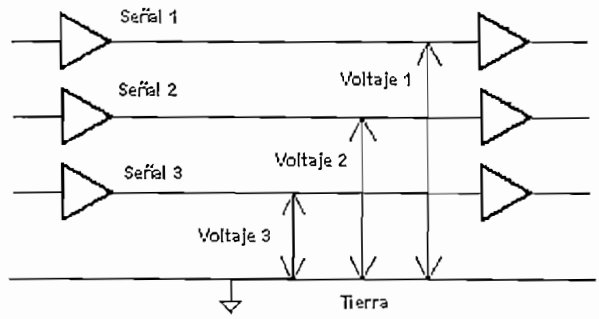


Figura 1.7
Líneas no Balanceadas

En teoría este sistema debiera funcionar bien, y en algunos casos lo hace. Sin embargo un simple conductor de tierra en el sistema no es perfecto y a menudo está muy lejos de serlo. Una perfecta referencia o tierra tiene el mismo potencial (voltaje) a lo largo de la longitud del cableado, sin ninguna diferencia de potencial entre dos puntos cualquiera.

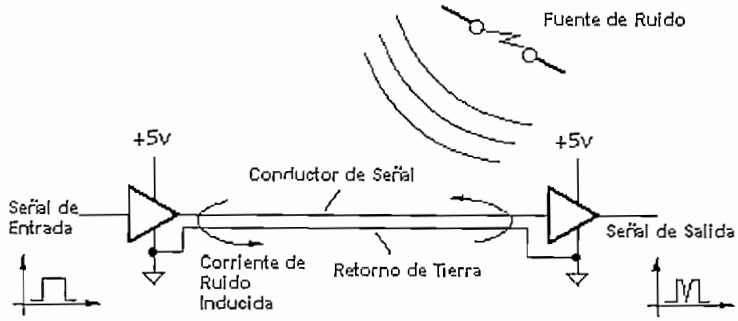


Figura 1.8
Agentes Externos

Un conductor real de tierra tiene resistencia, inductancia y capacitancia, que dan como resultado una diferencia de potencia de un punto a otro. Esto es un inconveniente debido a que un cable imperfecto puede captar ruido y, pueden inducirse corrientes dentro del conductor y por tanto diferencias de voltaje pueden originarse a raíz de esto.

Por tanto, el valor de la señal transmitida dependerá en suma, del punto donde se tome la medición. El efecto derivado de este efecto, es parecido a lo causado por las pérdidas de las caídas debido a IR (pérdidas por la resistencia intrínseca del cable); es decir, el voltaje caerá a lo largo de la longitud del cable a medida que la corriente fluya a través de la resistencia del cable de tierra.

La alternativa para solucionar este problema es no usar la tierra como referencia para la señal transmitida, lo que se logra utilizando no solamente un cable sino dos cables para cada señal a ser transmitida. Este método es conocido como diferencial o balanceado.

1.3.2 LÍNEAS BALANCEADAS

Una línea balanceada requiere dos conductores para transmitir cada señal. El voltaje en el receptor entonces es medido como la diferencia de voltaje entre estos dos conductores.

Este esquema es adecuado para eliminar la mayoría de problemas originados por el uso de un cable imperfecto, debido a que no asume que ninguno de los dos cables es ideal. Por el contrario, asume que los dos conductores tienen características similares y que por tanto ambos son afectados por igual por el ruido, caídas de voltaje y señales inducidas.

Debido a que ambos cables son afectados igualmente, la diferencia de voltaje entre ellos se mantendrá a lo largo de la longitud de los conductores, en cualquier punto en que estos sean medidos.

Un conductor es llamado señal alta (Hi), mientras que el otro se denomina señal baja (Lo).

En el método balanceado las señales de los dos cables son medidas con respecto a tierra.

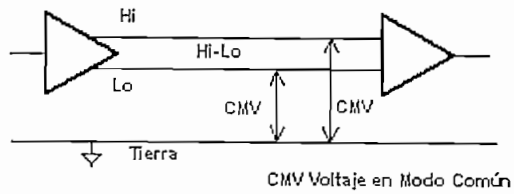


Figura 1.9
Línea Balanceada

En suma, el método diferencial es utilizado cuando hay problemas de captación de ruido, cuando los cables son imperfectos o cuando los niveles de las señales son de bajo voltaje.

1.4 TIPOS DE SALIDAS

Las configuraciones de salida comunes a la lógica digital son de open-collector / open drain, totem-pole, push-pull y 3-state.

1.4.1 OPEN-COLLECTOR (Colector-Abierto) y OPEN DRAIN (Drenaje-Abierto)

En este tipo de salidas, el colector del transistor de salida esta abierto o no esta conectado a ningún circuito en el chip. Entonces para usar la salida, se debe añadir un resistor de pull-up conectado a la fuente de alimentación (+5 V).

El transistor es utilizado como switch, de tal forma que cuando el transistor esta encendido (ON), la baja resistencia desde el pin de salida a tierra, da como resultado una salida lógica baja. Cuando el transistor esta apagado (OFF), el resistor de pull-up lleva el pin de salida al valor de la fuente de alimentación (+5V), dando como resultado un estado lógico alto.

Una de las ventajas de la lógica de colector-abierto (open-collector), es la capacidad de combinar dos o más salidas. Al tener dos salidas conectadas entre sí; cuando una de las salidas es baja, la baja resistencia desde el pin de salida a tierra hace que la salida combinada sea baja.

Este arreglo es algunas veces llamado wired-OR output (salida cableada OR), aunque actualmente se comporta como una compuerta OR solo si se asume una lógica negativa, donde un voltaje bajo es un 1 lógico y un voltaje alto es un 0 lógico.

Usando lógica positiva, si las compuertas individuales son buffers no invertidos, el circuito se comporta como una compuerta AND, es decir que cualquier entrada baja lleva la salida combinada a bajo. Si las compuertas son invertidas, el circuito es una compuerta NOR; cualquier entrada alta lleva la salida combinada a bajo.

Se puede entonces combinar las salidas para obtener una línea de datos bidireccional.

Por otro lado, una de las desventajas de la lógica de colector abierto, es su baja velocidad de conmutación (slow switching speed).

Cuando una salida pasa de bajo a alto, la capacitancia del cable se tiene que cargar a través de la resistencia de pull-up. Mientras mayor sea la resistencia mas lento será el cambio de voltaje de salida.

En componentes CMOS, el equivalente a las salidas de colector-abierto (open-collector) son las salidas de drenaje-abierto (open-drain).

Los dispositivos etiquetados con HCT son una muestra de este tipo de salidas. La tecnología es diferente, pero la operación es la misma o muy parecida.

Algunos dispositivos NMOS y CMOS tienen salidas que se comportan de manera similar a las de colector-abierto. Pero en lugar de pull-up pasivas externas, tienen un transistor interno con una alta resistencia que actúa como una débil y activa pull-up. Por tanto, tal como en la lógica de colector abierto, escribiendo 1 lógico a este tipo de salidas permite leer el estado lógico externo de un dispositivo conectado a estas. Otro nombre para estas salidas es cuasi-bidireccional.

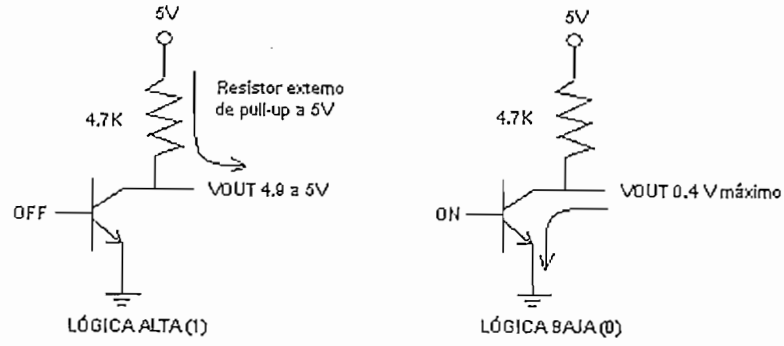


Figura 1.10

Salidas de colector abierto

1.4.2 TOTEM POLE

A diferencia de la lógica de colector-abierto, muchos dispositivos LSTTL usan el tipo de salida llamada totem-pole (pila totémica), con dos transistores puestos uno encima de otro. Cuando la salida es baja, el transistor inferior conduce, creando un camino de baja resistencia desde el pin de salida a tierra. Cuando la salida es alta, es el transistor superior quien conduce, creando un camino de baja resistencia a +5V. Una salida totem-pole puede drenar mas corriente a tierra que la que podría suministrar desde +5V.

La baja resistencia de salida da como resultado que una salida totem-pole pueda conmutar mas rápidamente que una salida de colector abierto. Pero esto también significa que sus salidas no pueden ser usadas para propósitos bidireccionales.

Si se combinan dos salidas totem-pole, siendo la una alta y la otra baja, se obtiene lo siguiente:

Siendo la una salida con una baja resistencia hacia +5V y la otra salida con una baja resistencia hacia tierra, el resultado es un estado lógico impredecible, pudiendo las altas corrientes generadas destruir los componentes envueltos.

Combinar una salida totem-pole y una salida de colector abierto da resultado, siempre y cuando la salida de colector abierto se mantenga en alto. Si la salida de colector abierto es baja y la salida tótem pole es alta; se pueden dar altas corrientes con resultado impredecible.

Si se conecta una salida totem-pole a una salida de colector-abierto, una resistencia en serie de 330 ohms en la línea puede proteger los circuitos (aunque esto daría como resultado una merma en la velocidad de conmutación).

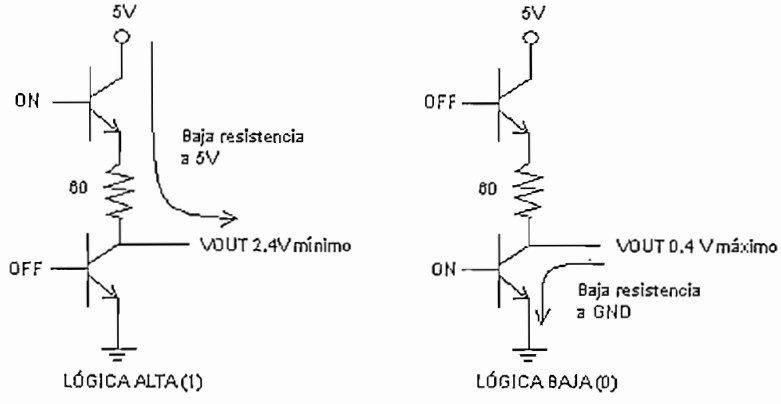


Figura 1.11
Salidas Totem Pole

1.4.3 PUSH PULL

Muchos de los circuitos CMOS, tienen salidas complementarias que son similares a las totem-pole, excepto que las capacidades de suministro y drenaje de corriente son iguales. Este tipo de salidas son denominadas push-pull.

1.4.4 3-STATE

Otro tipo de salidas son las denominadas 3-state (tres estados), las cuales tienen una señal de control que deshabilita las salidas por completo. Para propósitos prácticos el deshabilitar o poner en alta impedancia (tri-stating) una salida, da como resultado una desconexión eléctrica de la salida hacia cualquier circuito a la que este conectada físicamente.

Cuando la línea OE esta baja, las salidas siguen el estado lógico de las entradas. Por el contrario cuando OE esta alta, ambos transistores son apagados y las salidas no tienen efecto sobre los circuitos externos.

Las salidas que conectan los buses de la computadora son a menudo de tres-estados, con circuitos decodificadores de dirección que controlan los pines de habilitación. Esto permite a los chips de memoria y otros componentes compartir el bus de datos, habilitando a cada componente solo cuando la computadora selecciona la dirección del componente.

Tal como sucede con la lógica totem-pole, si se conecta dos salidas de tres-estados, el resultado puede ser impredecible. Por tanto si no se puede garantizar el comportamiento de las salidas en los circuitos; los dispositivos de colector abierto siguen siendo la mejor elección.

Por último, las salidas de tres-estados requieren una entrada adicional para controlar cada set de salidas. Con la lógica de colector abierto, se configura fácilmente las salidas como entradas sin necesidad de líneas de control extra.

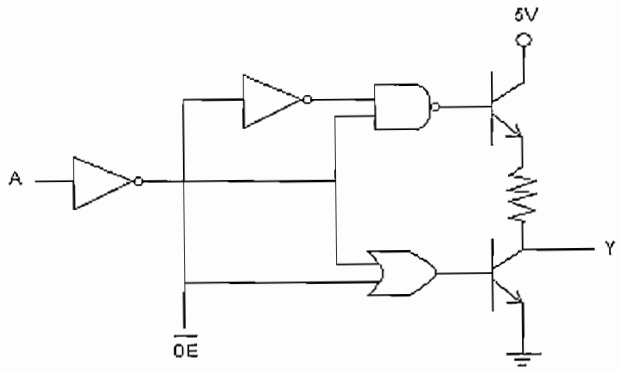


Figura 1.12
Salidas 3-State

1.5 ESTÁNDARES DE COMUNICACIÓN SERIAL Y PARALELA

A medida que computadores y dispositivos se conectan entre sí para intercambiar información, se hace necesario definir estándares que garanticen que la interface usada, sea esta física (hardware) o lógica (software), esté en concordancia y garantice una correcta comunicación entre los elementos actuantes.

Existen muchas ventajas tanto para usuarios del sistema como para los productores, el usar estándares bien definidos y aceptados para esas interfaces.

Los estándares de comunicación caen en dos categorías: los Promulgados y los De facto.

Los estándares de comunicación promulgados, son establecidos oficialmente por varias organizaciones internacionales, tales como:

La CCITT (Consultative Committe for Telegraphy and Telephony)
La IEC (International Electrotechnical Comission)
El IEEE (Institute of Electrical and Electronics Engineers)
La EIA (Electronic Industries Association)

Estas instituciones definen públicas especificaciones a través de las cuales los fabricantes pueden establecer esquemas de comunicación que permita la compatibilidad con una gran cantidad de diferentes fabricantes de productos. Estándares tales como el IEEE-488 para bus de instrumentación, el IEEE1284 para el puerto paralelo, el EIA RS-232 y el EIA-485, son algunos ejemplos de estándares bien definidos y proclamados.

Los estándares De facto (que no tienen la aprobación de algún cuerpo internacional de control), son métodos de interface que han ganado popularidad a través de su amplio uso. Aunque estos populares estándares han sido adoptados por toda la industria, ellos no tienen una definición especial. Debido a que no son apropiadamente definidos, algunos estándares De facto causan problemas de

interferencia; sin embargo, otros estándares, tal como el lazo de corriente de 20mA, son buenos y bien definidos.

Por otro lado, el proceso de transferencia de información puede ser categorizado ampliamente en dos diferentes modos y dos diferentes formas de conexión: Transferencia serial y Transferencia paralela; Conexión radial (punto-a punto) y Conexión highway (multipunto o bus)
La transferencia serial y paralela difieren en esencia solamente en la velocidad global de transferencia. En el modo paralelo, todo o una porción de la unidad de información (carácter o palabra) es puesta en el sistema de una sola vez. En el más elemental modo serial, solamente la unidad fundamental de información, 1 bit, está disponible al mismo tiempo. Entonces, en teoría la transferencia paralela de palabras o bytes pueden tomar lugar tan rápidamente como un enlace serial puede transferir bits. Sin embargo, existen diferencias de esta simplista extrapolación, que no son precisamente regidas por el número de caminos en una conexión. Claramente se puede ver que un modo paralelo demanda mayor gasto de conexión, debido al gran número de conductores separados requeridos. La conexión punto a punto es entonces más costosa que una conexión multipunto puesto que los caminos tenderían a ser duplicados.

1.5.1 ESTÁNDAR IEEE1284

El estándar IEEE 1284 es un documento que define y describe los protocolos y convenciones para las comunicaciones a través del puerto paralelo.
El estándar fue publicado en 1994, y fue desarrollado por un comité del Instituto de Ingenieros Eléctricos y Electrónicos (Institute of Electrical and Electronic Engineers) IEEE.

El IEEE es una organización de ingenieros y de compañías de ingeniería. Una de sus actividades es desarrollar y publicar estándares o documentos que recomiendan especificaciones para practicas de ingeniería, incluyendo las interfaces a computadoras.

El propósito de desarrollar estos estándares es reducir la confusión e incompatibilidad que resultan cuando cada fabricante desarrolla productos independientemente.

La designación para el estándar del puerto paralelo es IEEE Std 1284-1994 y el título completo se describe como Standard Signaling Method for a Bi-directional Parallel Interface for Personal Computers.

El estándar describe los modos de transferencia Compatible, Nibble, Byte, EPP, ECP y el protocolo de negociación, que inquiere al periférico por los modos que soporta.

Además define las señales del puerto paralelo y sus usos en los diferentes modos, incluyendo las especificaciones de tiempo. También describe conectores, interface eléctrica y cables, incluyendo los originales y los nuevos tipos de alto rendimiento.

El IEEE 1284 no cubre todo cuanto se espera. Lo mas notable es que no dice nada acerca de cómo programar o accesar la interface en un PC o periférico. No menciona los registros del puerto paralelo y como usar estos para configurar, leer y escribir al puerto.

Sin embargo, otras fuentes han documentado convenciones para solventar estos asuntos. Un documento publicado por Microsoft describe el uso de los registros y protocolos para puertos ECPs. Por otro lado las hojas de datos incluidos con los controladores de puerto paralelo describen el uso de los registros del puerto paralelo. Otro comité de IEEE esta desarrollando un estándar BIOS, o API, para el uso con los nuevos modos.

El estándar identifica tres tipos de conectores para una interface 1284:
Conector de 25 pines 1284 Tipo A
Conector de 36 pines 1284 Tipo B
Conector de 36 pines (compacto) 1284 Tipo C

1.5.2 ESTÁNDAR EIA232

El RS-232 fue desarrollado a principio de los años 60, por un comité de estandarización conocido hoy en día como Asociación de Industrias Electrónicas (Electronic Industries Association EIA), para satisfacer las necesidades crecientes de comunicación de datos entre dispositivos ubicados remotamente, conectados a líneas telefónicas de voz, por medio de modems.

Debido a las altas tasas de error que ocurren al transmitir datos a través de un canal análogo, el estándar fue creado en primera instancia para asegurar comunicaciones confiables y permitir la interconexión de equipos producidos por otros fabricantes.

Este estándar especifica las señales de voltaje, temporización, función de las señales y un protocolo para intercambio de información y los conectores mecánicos.

Luego de que este estándar fue desarrollado, la EIA publicó tres modificaciones, siendo la más reciente la EIA232E introducida en 1991. Además de que el nombre fue cambiado de RS-232 a EIA232, también algunas líneas de señal fueron renombradas y otras nuevas definidas, incluyendo el conductor de blindaje.

Durante este largo periodo de más de 40 años, con el rápido crecimiento de la electrónica, fabricantes adoptaron versiones simplificadas de esta interface para aplicaciones que fue imposible preverlas en los 60. Hoy en día las interfaces seriales son muy parecidas a la EIA232, en cuanto a sus señales de voltaje, protocolos y conectores, sin tomar en cuenta si un modem es usado o no.

La implementación completa del estándar EIA232, define en un extremo de la conexión un dispositivo llamado DTE (Data Terminal Equipment, usualmente un computador o terminal), que tiene un conector macho DB25, y utiliza 22 de los 25 pines disponibles para señales o referencia. El siguiente equipo en conexión es llamado DCE (Data Circuit-terminating Equipment, usualmente un modem), tiene

un conector DB25 hembra, y utiliza los mismas 22 pines disponibles para señales y referencia. El cable de enlace entre DTE y DCE, es un cable paralelo directo.

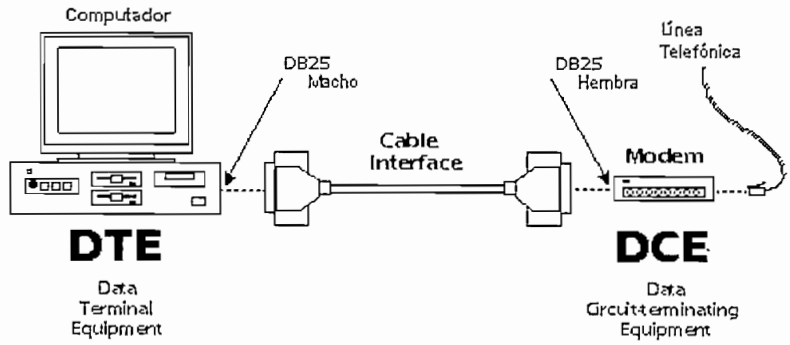


Figura 1.13
Interface EIA232

CAPÍTULO 2
PUERTO PARALELO

2.1 INTRODUCCIÓN

Desde su origen, concebido como una interface para impresoras, el puerto paralelo de los computadores personales ha llegado a ser el enlace desde donde una gran cantidad de dispositivos pueden ser conectados al computador.

Casi todos los computadores personales compatibles con IBM, tienen un puerto paralelo al cual se puede tener acceso por medio de un conector localizado detrás del panel.

El propósito original del puerto paralelo, fue el proporcionar una interface para la impresora, y este sin duda sigue siendo el uso más común. Sin embargo debido a su versatilidad, su aplicación se ha extendido, por lo que a nivel experimental las líneas del puerto se emplean para tareas básicas de monitoreo, control, transferencia de datos, y otras aplicaciones.

En general, el puerto paralelo ofrece una rápida solución para proyectos de pequeña escala que requieran comunicación entre el computador y un dispositivo externo.

2.2 DEFINICIÓN

El puerto paralelo es un arreglo de líneas de señal, que el microprocesador o CPU usan para intercambiar información con otros componentes.

Típicamente el puerto paralelo es usado para comunicarse con impresoras, teclados y displays. Un puerto paralelo transfiere múltiples bits a la vez, a diferencia de un puerto serial, que transfiere un bit a la vez.

El puerto paralelo se encuentra comúnmente en la parte posterior del PC como un conector tipo DB-25 hembra.

Este trabajo se desarrolla sobre un tipo específico de puerto paralelo: el encontrado en cada PC o computador personal compatible con IBM.

En nuevos PCs, se puede encontrar puertos tales como el USB (Universal Serial Bus) SCSI (Small Computer System Interface), e incluso IrDA; pero el puerto paralelo se mantiene popular debido a su capacidad, flexibilidad, y debido a que todo PC tiene uno.

El término PC-Compatible toma como referencia al computador de IBM e identifica a cualquier computador personal derivado de este, que pueda correr el sistema operativo MS-DOS de Microsoft y cuyo bus de expansión sea compatible con el bus ISA original. La categoría incluye al XT, AT, PS/2, y la mayoría de computadores con 80x86, Pentium, y CPUs compatibles.

2.3 TIPOS DE PUERTOS PARALELOS

El puerto paralelo original tiene ocho salidas (registro de Datos), cinco entradas (registro de Estado) y cuatro líneas bidireccionales (registro de Control). Estas son suficientes para la comunicación con cualquier tipo de periférico.

Por otro lado, como el puerto paralelo fue diseñado como un puerto de impresora muchos de los nombres originales de las señales reflejan ese uso.

Muchos fabricantes han introducido versiones mejoradas de puerto paralelo. Los nuevos tipos son compatibles con el diseño original, pero tienen nuevas capacidades; principalmente incremento de velocidad. En las nuevas PCs las ocho salidas pueden también servir como entradas, para una mayor velocidad de comunicación con escáneres y otros dispositivos que envían datos al PC.

La velocidad es importante a medida que el trabajo que realizan los computadores y periféricos se vuelve más complicado y la cantidad de información que deben intercambiar se incrementa.

En términos generales, cuanto más rápido pueda el computador transmitir información, más rápido podrá la impresora, si es que posee las características para aquello, procesar la información e imprimir el resultado.

A continuación se desarrolla un sumario de los puertos paralelos disponibles hoy en día:

2.3.1 SPP (Standard Parallel Port)

El puerto paralelo original de las PCs IBM y cualquier puerto que emule el diseño del puerto original, es llamado SPP o puerto paralelo estándar (Standard Parallel Port). Otros nombres usados son: tipo-AT o ISA-compatible.

El SPP puede transferir ocho bits de datos a la vez a un dispositivo periférico, usando un protocolo similar al de la Interface Centronics original. El SPP no tiene puerto de entrada de ocho bits completo (byte-wide input), pero puede usar el modo de transferencia Nibble en el que se manipulan cuatro bits a la vez. Aunque el modo Nibble es lento, ha llegado a ser muy popular al utilizar el puerto paralelo como entrada de datos.

2.3.2 PS2 (Simple Bidirectional)

Una de las primeras mejoras al puerto paralelo fue precisamente hacer bidireccional al registro de datos y fue introducido en el modelo PS/2 de IBM. El registro bidireccional permite a un periférico transmitir ocho bits a la vez hacia un PC. El término PS/2 refiere a todos aquellos puertos paralelos que tienen un puerto bidireccional de datos pero que no soporta los modos EPP y ECP que se describen a continuación. El modo "Byte" es un protocolo de transferencia de ocho bits de datos que usa el registro tipo PS/2 para transmitir datos desde un periférico al PC.

2.3.3 EPP (Enhanced Parallel Port)

El puerto paralelo mejorado EPP, al igual que el tipo PS2, tiene líneas de datos bidireccionales.

Pero un EPP puede leer o escribir un byte de datos en un ciclo del bus de expansión ISA (1us), incluido el handshaking, comparado con 4 ciclos para SPP o PS/2.

Un puerto EPP puede conmutar direcciones rápidamente, por lo que es muy eficiente al usar con dispositivos que usan transferencia bidireccional como discos y manejadores de cinta.

Un EPP puede emular un SPP, y algunos EPPs pueden incluso emular un puerto tipo PS/2.

2.3.4 ECP (Extended Capabilities Port)

El puerto de capacidades extendidas ECP, es también un puerto bidireccional, y como el EPP, puede transferir datos a la velocidad del bus ISA.

Los ECPs tienen buffers y soporte para transferencias DMA (Direct Memory Access) y compresión de datos.

Las transferencias ECP son muy convenientes para impresoras, escáneres, y otro tipo de periféricos que transfieren grandes bloques de datos.

Un ECP puede emular un SPP o PS/2, e incluso algunos ECPs pueden emular un EPP.

2.3.5 MULTIMODE PORTS

Muchos de los nuevos puertos son multimodo; esto significa que pueden emular algunos o todos los tipos de puerto antes mencionados. A menudo incluyen

opciones de configuración desde donde se pueden habilitar todos los tipos de puerto nombrados o por el contrario bloquear algunos de ellos.

2.3.6 RECURSOS QUE EMPLEAN LOS PORTICOS

El puerto paralelo se vale de una variedad de recursos del computador. Así:

- Cada puerto usa un rango de direcciones (aunque el número y localización varían).
- Muchos tienen asignados niveles de IRQs (peticiones de interrupción)
- A los ECPs se les puede asignar un canal DMA (Direct Memory Access).

Los recursos asignados a un puerto no pueden estar en conflicto con aquellos usados por otros componentes del sistema, incluyendo otros puertos paralelos.

2.3.6.1 Direccionamiento

El puerto paralelo estándar usa tres direcciones contiguas, en uno de estos rangos:

Reg. Datos	Reg. Estado	Reg. Control
3BCh	3BDh	3BEh
378h	379h	37Ah
278h	279h	27Ah

Tabla 2.1
Direcciones Base del Puerto Paralelo

Donde la primera dirección corresponde a la llamada dirección base del puerto. La siguiente dirección es el registro de estado y la dirección final se reserva para el registro de control.

Los puertos EPP y ECP apartan direcciones adicionales para cada puerto.

El puerto EPP adiciona cinco registros desde la dirección base + 3 hasta la dirección base + 7.

Un ECP añade tres registros desde la dirección base + 400h hasta la dirección base + 402h.

Los primeros puertos paralelos tienen una dirección base en 3BCh. En los nuevos sistemas, el puerto paralelo está más a menudo en 378h. De cualquier forma, tres direcciones se encuentran apartadas para el puerto paralelo, y si el hardware lo permite se puede configurar el puerto en cualquiera de estas direcciones.

El puerto PS/2 tipo 3 de IBM, también tiene tres registros adicionales en la dirección base +3 hasta la dirección base + 5 y permite una dirección base de 1278h o 1378h.

Muy a menudo DOS y WINDOWS se refieren al primer puerto en orden numérico como LPT1, el segundo LPT2 y el tercero LPT3. Así, si alguno de ellos o los tres estuviesen presentes se debe considerar que:

LPT1 puede estar en cualquiera de las tres direcciones.
LPT2 puede estar en 378h o 278h.
LPT3 puede estar solo en 278h.

Varias técnicas de configuración pueden cambiar estas asignaciones, por lo tanto no todos los sistemas siguen esta convención. Sin embargo la denominación de un puerto como LPT se mantiene para nombrar al puerto de la impresora.

Si el hardware de puerto lo permite, se puede añadir un puerto en cualquier dirección no estandar; pero no todo software reconoce los puertos ubicados en direcciones distintas a las citadas como puertos LPTs, sin embargo se puede acceder a ellos mediante un software que escriba directamente en las direcciones asignadas.

2.3.6.2 Interrupciones

La mayoría de puertos paralelos, son capaces de detectar señales de interrupción desde un periférico.

Normalmente un periférico utiliza una interrupción para indicarle al puerto que está listo para recibir un byte, o a su vez que tiene un byte para enviar. Para usar dichas interrupciones el puerto paralelo debe tener asignado un nivel de petición de interrupción conocido como IRQ (Interrupt Request).

Convencionalmente, el LPT1 usa la interrupción IRQ7 y el LPT2 usa la IRQ5. Pero IRQ5 es también usada por algunas tarjetas de sonido, y debido a que las IRQs pueden ser escasas en el sistema; incluso la IRQ7 puede ser reservada por otro dispositivo. Algunos puertos permiten elegir otros niveles de IRQs además de a dos anteriores.

Muchos manejadores de impresoras y muchas otras aplicaciones que acceden al puerto paralelo, no requieren de interrupciones para efectuar la comunicación.

Si no se selecciona un nivel de IRQ para un puerto paralelo, este seguirá trabajando, en la mayoría de los casos, aunque algunas veces no tan eficientemente. Se puede entonces hacer uso de la interrupción libre para otra tarea.

2.3.6.3 Canales DMA (Direct Memory Access)

Como se indico, los ECPs hacen uso del acceso directo a memoria o DMA, para la transferencia de datos del puerto paralelo.

Durante una transferencia DMA, el CPU esta libre para realizar otras tareas; por tanto las transferencias DMA pueden resultar mucho más convenientes al ofrecer mayor velocidad y rendimiento para el sistema.

Para usar un DMA, el puerto debe tener asignado un canal DMA, en el rango de cero (0) a tres (3).

2.4 CONFIGURACIÓN

El puerto paralelo que viene con cada PC tiene asignada una dirección, posiblemente una IRQ y hasta un canal DMA. Por otro lado los puertos multimodo pueden ser configurados con modos específicos. De esta forma se puede cambiar algunas o todas de estas asignaciones hasta llegar a la adecuada.

Al añadir un nuevo puerto, se lo debe configurar asegurándose que no se encuentre en conflicto con alguno de los recursos del sistema mencionados anteriormente.

No hay un método estándar para configurar un puerto paralelo. Muchos puertos, especialmente los antiguos, usan bloques de puentes (jumper) o interruptores (switches) para seleccionar las diferentes opciones. Otros se los puede configurar mediante el software que provee el fabricante.

Un puerto sobre una tarjeta principal o tarjeta madre puede tener opciones de configuración en la pantalla setup del sistema (CMOS-setup), a las cuales se puede acceder al iniciar el mismo.

Para puertos multimodo se necesita tomar muy en cuenta las consideraciones antes mencionadas, pues tampoco existe un método estándar para configurarlos y es posible además que exista una gran variedad de chips controladores de dichos puertos.

En el caso de puertos que soportan el estandar plug-and-play de Microsoft, Windows puede asignarles automáticamente una dirección estandar disponible y un nivel IRQ.

Si el puerto soporta transferencias ECP, en lo posible se debe asignar además un canal DMA. Esto es necesario puesto que la mayoría de manejadores usan estos recursos, y de no estar disponibles, el manejador o controlador recurrirá al modo de transferencia más lento.

Se debe chequear la documentación referente al puerto a instalar para encontrar especificaciones de cómo configurarlo adecuadamente.

2.5 ACCESO A LOS PUERTOS PARALELOS

La forma más directa de acceder a un puerto es leyendo y escribiendo las direcciones de los registros asignados al mismo. La mayoría de lenguajes de programación incluyen esta opción, o al menos permiten añadirla.

Uno de los lenguajes que permiten el manejo de puertos escribiendo directamente sobre sus registros es Visual Basic. Windows y DOS ofrecen también algunos métodos para realizar este cometido.

2.5.1 SEÑALES

La Tabla 3 muestra las funciones de cada uno de los 25 pines del conector hembra DB25 del puerto paralelo.

La mayoría de nombres de señal y función anteriormente descritos, están basados sobre la convención establecida por Centronics Data Computer Corporation, un primer fabricante de impresoras de matriz de puntos.

DB-25 Pin	SEÑAL	FUNCIÓN
1	nStrobe	Validación de datos D0-D7
2	D0	Bit 0 de datos
3	D1	Bit 1 de datos
4	D2	Bit 2 de datos
5	D3	Bit 3 de datos
6	D4	Bit 4 de datos
7	D5	Bit 5 de datos
8	D6	Bit 6 de daros
9	D7	Bit 7 de datos
10	nAck	Acuse de recibo de datos (puede disparar una interrupción)
11	Busy	Impresora ocupada
12	PaperEnd	Fin de papel, vacío
13	Select	Impresora seleccionada (on line)
14	nAutoLF	Generación automática de suministro de línea después de retorno del carro
15	nError	Error
16	nInit	Inicializa impresora (reset)
17	nSelectIn	Selección de impresora (ubicar en línea)
18	Gnd	Tierra de retorno para nStrobe, D0
19	Gnd	Tierra de retorno para D1, D2
20	Gnd	Tierra de retorno para D3, D4
21	Gnd	Tierra de retorno para D5, D6
22	Gnd	Tierra de retorno para D7, Ack
23	Gnd	Tierra de retorno para nSelectIn
24	Gnd	Tierra de retorno para Busy
25	Gnd	Tierra de retorno para nInit

Tabla 2.2

Asignación de Pines y Señales del Puerto Paralelo

Cada señal y su correspondiente función pueden variar de acuerdo al modo elegido para el puerto paralelo.

2.5.2 ACUERDO DE NOMBRES DE LOS REGISTROS

Se indicó que el puerto paralelo estándar usa tres registros de 8 bits cada uno. El PC accede a las señales del puerto paralelo leyendo o escribiendo directamente los registros llamados: Registro de Datos, Registro de Estado y Registro de Control.

Cada una de la señales tiene un nombre que sugiere su función en una interface de impresión. Sin embrago, en interfaces con otro tipo de periféricos, estas señales no necesariamente se utilizan con su propósito original.

Debido a que el objetivo de esta tesis es la interface con otro dispositivo (microcontrolador), que no es la impresora, en adelante se usaran nombres más genéricos para identificar las señales de los registros mencionados.

Registro	Nro de Bits	Denominación	Dirección
Registro de Datos	8 bits de datos	D0-D7	Dirección Base
Registro de Estado	5 bits de estado	S3-S7	Dirección Base + 1
Registro de control	4 bits de control	C0-C3	Dirección Base + 2

Tabla 2.3

Acuerdo de Nombres de los Registros

Las letras entonces identifican el registro del puerto y los números denotan la posición del bit en cada registro.

El hardware del puerto invierte 4 de las señales entre el conector y el correspondiente bit del registro.

Para S7, C0, C1 y C3, el estado lógico en el conector es el complemento o inverso del estado lógico del correspondiente bit del registro.

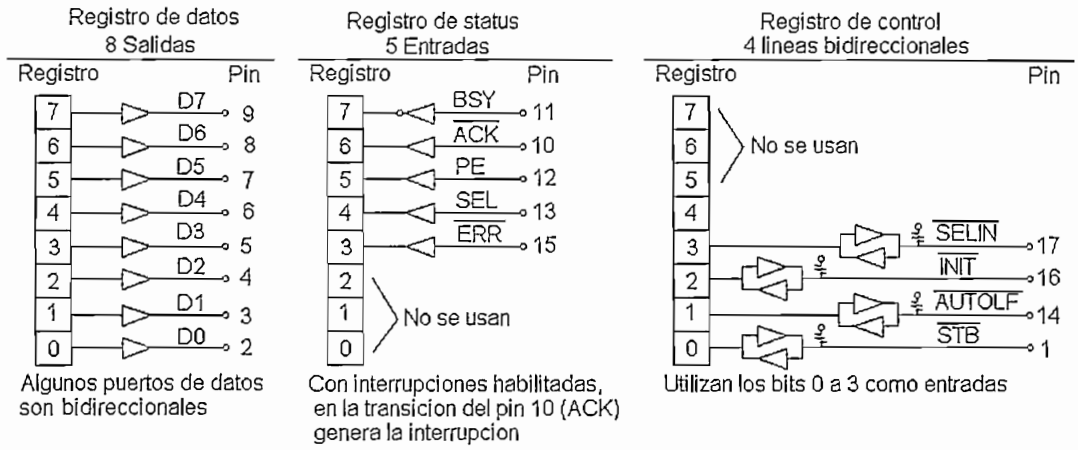


Figura 2.1

Registros del Puerto Paralelo

2.5.3 REGISTRO DE DATOS

El registro de datos (D0-D7), mantiene el byte escrito a la salida de datos.

En puertos bidireccionales de datos, cuando el puerto es configurado como entrada; el registro de datos mantiene el byte leído en los pines de datos del conector.

El estándar IEEE-1284 se refiere a las líneas de datos como D1-D8.

Dirección Base			
Bit	DB-25 Pin	Nombre señal	Invertido en el conector
D0	2	Bit 0 de datos	No
D1	3	Bit 1 de datos	No
D2	4	Bit 2 de datos	No
D3	5	Bit 3 de datos	No
D4	6	Bit 4 de datos	No
D5	7	Bit 5 de datos	No
D6	8	Bit 7 de datos	No
D7	9	Bit 7 de datos	No

Tabla 2.4

Registro de Datos

Algunos puertos de datos son bidireccionales.

2.5.4 REGISTRO DE ESTADO

El registro de estado mantiene los estados lógicos de las entradas S3, S4, S5, S6 y complementa el valor leído en el conector para S7.

Los bits S0 a S2 no aparecen en el conector. El registro de estado es solo de lectura, excepto por S0, el cuál es una bandera de desbordamiento (timeout) en puertos que soportan transferencias EPP, y puede ser borrado por software.

En algunos puertos, las entradas de estado tienen resistores de pull-up.

En usos convencionales, los bits de estado tienen las siguientes funciones:

- *S0: timeout.* En modo EPP este se pone en alto para indicar el timeout de una transferencia de datos EPP.
  De otro modo no es usado. No aparece en el conector.

- *S1: no usado.*
- *S2: no usado.* Excepto por algunos pocos puertos donde este bit indica el estado de interrupción del puerto paralelo (PIRQ). Si el bit es cero ha ocurrido una interrupción. Si es uno no ha ocurrido una interrupción.
  En estos puertos, leyendo el registro de estado se pone a 1 = PIRQ.
- *S3: nError.* Pasa a bajo cuando la impresora detecta un error o falla.
- *S4: Select.* Pasa a alto cuando la impresora esta on-line (cuando las entradas de datos de la impresora están habilitadas).
- *S5: PaperEnd.* Alto cuando falta papel en la impresora.
- *S6: nAck.* Pasa a bajo cuando la impresora recibe un byte. Cuando las interrupciones están habilitadas (C4 = 1), una transición de alto a bajo en este pin, genera una interrupción.
- *S7: Busy* .Bajo cuando la impresora no esta lista para aceptar un nuevo dato. Este señal aparece invertida en el conector.

Dirección Base + 1			
Bit	DB-25 Pin	Nombre señal	Invertido en el conector
S3	15	nError(nFault)	No
S4	13	Select	No
S5	12	PaperEnd	No
S6	10	NAck	No
S7	11	Busy	Si

Tabla 2.5
Registro de Estado

Los bits adicionales no están disponibles en el conector:
Bit 0: puede indicar timeout (1 = timeout)
Bits 1,2: no usados

2.5.5 REGISTRO DE CONTROL

El registro de control, mantiene el estado leído en el conector para C2 y complementa C0,C1 y C3. Entonces en el conector del puerto se tiene: C0, C1, C2 y C3

Convencionalmente estos bits son usados como salidas.

En muchos SPPs, sin embargo los bits de control son de colector abierto, o de drenaje abierto, lo cual significa que pueden también funcionar como entradas. Para leer una señal externa lógica en un bit de control, se debe previamente escribir un 1L al bit del conector del puerto correspondiente y luego leer el bit desde el registro. Pero en muchos puertos que soportan modos EPP y ECP, con el propósito de mejorar la velocidad de conmutación, las salidas de control son del tipo push-pull y por tanto no pueden ser usadas como entradas.

En algunos puertos multimodo, los bits de control tienen salidas push-pull para los modos avanzados, y por compatibilidad estos conmutan a salidas de colector abierto o drenaje abierto cuando emulan un SPP.

Los bits C4 a C7 no aparecen en el conector. En su uso convencional los bits de control tienen las siguientes funciones.

- *C0: nStrobe.* (Invertida en el conector) Un pulso bajo en esta señal del conector indica a la impresora que puede leer D0-D7. Normalmente esta señal se encuentra en 1L al encender el computador.
- *C1: AutoLF.* (Invertida en el conector) Un pulso bajo en este pin del conector le indica a la impresora que genere el retorno automático de línea, luego de cada retorno de carro. Luego de iniciar el PC, normalmente se encuentra en alto en el conector.
- *C2: nInit.* Pulsa a bajo para resetear la impresora y limpiar su buffer. Mínimo ancho de pulso 50 useg. Luego de iniciar el PC, normalmente alto en el conector.

- *C3: nSelectIn.*(Invertido en el conector) Valida a alto para indicar a la impresora que habilite sus entradas de datos. Luego de iniciar el PC normalmente bajo en el conector.
- *C4: Enable Interrupt request.* Se pone en alto para habilitar la interrupción que genera nAck (S6). Si C4 es alto y si los niveles de interrupción del puerto están habilitadas en el control de interrupciones, una transición en nAck (S6), origina una petición de interrupción de hardware. No aparece en el conector.
- *C5: Direction control.* En puertos bidireccionales, decide la dirección del puerto de datos. Si es cero (0) habilita la salida de datos. Si es uno (1) habilita la entrada de datos y en este caso deshabilita la salida. Usualmente primero se debe configurar el puerto para uso bidireccional (modo PS/2), para asegurar que el cambio en este bit tenga efecto. No aparece en el conector. No usado en SPP.
- *C6: Unused.* No usada.
- *C7: Unused.* No usada; eexcepto por unos pocos puertos donde este bit lleva a cabo la configuración de la dirección del puerto de datos, hecha normalmente por C5.

Cuando cualquiera de estos bits está en alto, el PC puede leer entradas externas a través del Puerto de Control (Sólo SPP).

Dirección Base + 2			
Bit	DB-25 Pin	Nombre señal	Invertido en el conector
C0	1	nStrobe	Si
C1	14	nAutoLF	Si
C2	16	nInit	No
C3	17	nSelectIn	Si

Tabla 2.6

Registro de Control

2.6 HARDWARE DEL PUERTO

El hardware del puerto paralelo incluye: el conector posterior del panel y los circuitos y cables entre el conector y el bus de expansión del sistema. El procesador del PC usa los datos, direcciones y líneas de control del bus de expansión para transferir información entre el puerto y el CPU, la memoria y otros componentes del sistema.

2.6.1 CONECTORES

El estándar IEEE 1284 define 3 tipos de conectores:

IEEE 1284-A	25 pines
IEEE 1284-B	36 pines
IEEE 1284-C	36 pines (compacto)

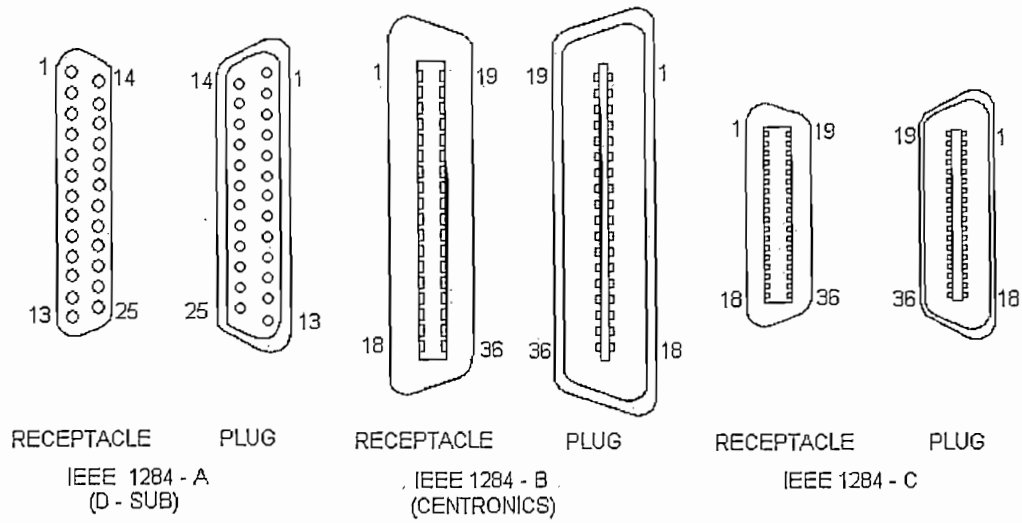


Figura 2.2
Tipos de Conectores

El conector del panel posterior del PC, en la mayoría de puertos paralelos es un conector hembra de 25 contactos sub-D. Otros nombres para este conector son el subminiatura D, DB25, o solamente conector D. El estándar IEEE 1284 para puerto paralelo lo denomina conector IEEE 1284-A.

Comúnmente en los periféricos se utiliza el conector hembra llamado Centronics (IEEE 1284-B).

El estándar describe estos dos tipos de conectores pero no recomienda un conector en particular para cada dispositivo.

Un último conector incluido es el nuevo y compacto IEEE 1284-C. Este es un conector de 36 contactos similar al Centronics. Con este conector, el estándar recomienda el uso de receptáculos hembras tanto para el PC como para el periférico, con conectores machos para el cable. La recomendación para los nuevos diseños es el conector tipo C. Es común que el conector DB25 del puerto paralelo se confunda a primera vista con el conector de puerto serial o, incluso, con algunos conectores de la interfaz SCSI, que también tienen 25 contactos.

Dentro del computador, los circuitos del puerto paralelo pueden estar sobre la tarjeta principal o sobre una tarjeta externa que se coloca en una ranura del bus de expansión del sistema.

A continuación se muestra una tabla con la asignación de pines para los tres tipos de conectores según IEEE1284:

Nombre de Señal	Bit del Registro	Pin de Señal			Pin de Retorno de Señal		
		IEEE 1284-A	IEEE 1284-B	IEEE 1284-C	IEEE 1284-A	IEEE 1284-B	IEEE 1284-C
Bit de Datos 0	D0	2	2	6	19	20	24
Bit de Datos 1	D1	3	3	7	19	21	25
Bit de Datos 2	D2	4	4	8	20	22	26
Bit de Datos 3	D3	5	5	9	20	23	27
Bit de Datos 4	D4	6	6	10	21	24	28
Bit de Datos 5	D5	7	7	11	21	25	29
Bit de Datos 6	D6	8	8	12	22	26	30
Bit de Datos 7	D7	9	9	13	22	27	31
nError	S3	15	32	4	23	29	22
Select	S4	13	13	2	24	28	20
PaperEnd	S5	12	12	5	24	28	23
nack	S6	10	10	3	24	28	21
Busy	S7	11	11	1	23	29	19
nStrobe	C0	1	1	15	18	19	33
nAutoLF	C1	14	14	17	25	30	35
nInit	C2	16	31	14	25	30	32
nSelectIn	C3	17	36	16	25	30	34
HostLogicHigh				18			18
PeriphLogicHigh				36			36

Tabla 2.7

Asignación de Pines DB25, CENTRONICS, IEEE1284C

2.6.2 CABLES

La mayor parte de los cables de impresión tienen un conector macho de 25 pines (DB25) en el un extremo y un conector macho de 36 pines en el otro. Este conector de 36 pines es conocido como conector Centronics, debido a que es del mismo tipo que el usado en las impresoras Centronics. Otros nombres para este cable son: Conector de interfaz paralela o simplemente conector de impresora. La IEEE 1284 lo llama conector 1284-B.

Por otro lado existen diferentes tipos de periféricos que pueden usar diferentes conectores y por tanto diferentes cables. Así, algunos usan un conector DB-25 similar al del PC. Un dispositivo que requiere solo pocas señales puede usar un conector telefónico; ya sea de 4 conductores RJ11, o uno de 8 conductores RJ45. Nuevos periféricos pueden tener un conector 1284-C de 36 contactos.

En cualquier caso, debido a que las salidas del puerto paralelo no son diseñadas para transmitir sobre largas distancias, es recomendado que el cable se mantenga dentro de una distancia entre 1,82 y 3m; o en su defecto puede llegar a tener hasta 10m, siempre y cuando se ciña a IEEE-1284.

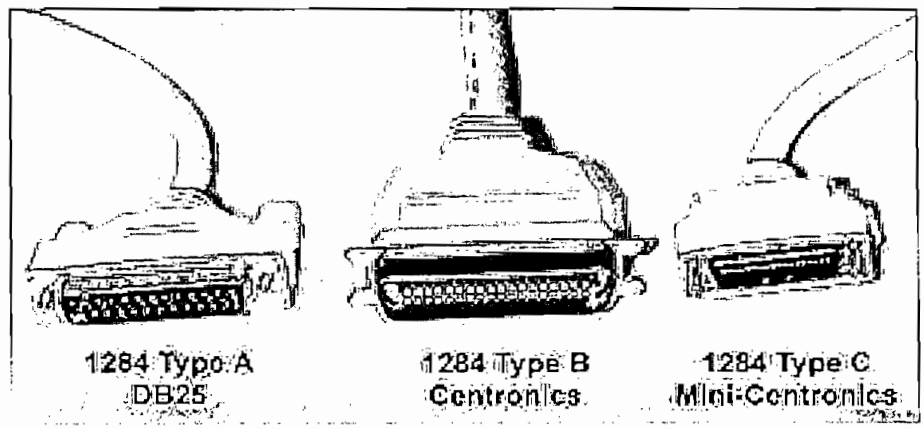


Figura 2.3
Conectores en Cables

Para un enlace corto, de baja velocidad y que no sea muy crítico el uso de cualquier cable y conectores no representa mayor problema, debido a que se asume que las señales cambian lentamente.

Al utilizar cables más largos y trabajar a altas velocidades, el diseño del cable y conectores hacen la diferencia entre un enlace que trabaja confiablemente y uno que no lo hace.

2.6.3 TIERRA DE RETORNO

Como es conocido, en todos los circuitos la corriente debe retornar a la fuente. Un enlace con cables no es la excepción, los cables de tierra proveen un camino de retorno para la corriente. Aunque aparentemente un cable de tierra no tiene voltaje, la impedancia del mismo hace que ante la presencia de corriente se induzca voltaje.

Cuando múltiples señales comparten una tierra de retorno, cualquiera de las entradas puede receptar voltajes de tierra causadas por las otras. En la interface Centronics original, la mayoría de señales tienen su propia tierra de retorno formando un par trenzado en el cable.

El conector DB-25 de los PCs puede alojar solamente 8 contactos de tierra. Por lo tanto, solo pocos contactos son diseñados como retornos de tierra para una señal en particular, mientras que otros tienen retorno de tierra para dos señales.

Debido a que la corriente de tierra toma el camino de menor resistencia, no hay forma de garantizar que la corriente fluirá por algún cable en particular.

Sin embargo múltiples cables de tierra harán que la impedancia total sea menor, lo que reducirá las corrientes de tierra.

La IEEE 1284 introdujo un nuevo cable de puerto paralelo. El cable contiene 18 pares trenzados, es decir cada línea de señal en par con su retorno de tierra. Esto provee distancias más largas de cable 10m (33 pies), comparado con los 10 pies del cable paralelo original.

En este tipo de cables, el par 18 tiene dos cables con nuevas funciones que utilizan el computador y el periférico para detectar la presencia del otro dispositivo. Si se usa el nuevo cable con conectores 1284-C, cada contacto esta conectado a un cable.

Par del Cable	Computador IEEE 1284-A (D-sub)		Periférico IEEE 1284-B (Centronics)	
	Señal	Pin	Pin	Señal
1	S7 (Busy)	11	11	S7 (Busy)
	Tierra (S7, S3)	23	29	Señal de Tierra (S7)
2	S4 (Select)	13	13	S4 (Select)
	Tierra (S4, S5, S6)	24	28	Señal de Tierra (S4)
3	S6 (nAck)	10	10	S6 (nAck)
	Tierra (S4, S5, S6)	24	28	Señal de Tierra (S6)
4	S3 (nError)	15	32	S3 (nError)
	Tierra (S4, S5, S6)	23	29	Señal de Tierra (S3)
5	S5 (PaperEnd)	12	12	S5 (PaperEnd)
	Tierra (S4, S5, S6)	24	28	Señal de Tierra (S5)
6	Bit de Datos 0 (D0)	2	2	Data Bit 0 (D0)
	Tierra (D0, D1)	19	20	Señal de Tierra (D0)
7	Bit de Datos 1 (D1)	3	3	Bit de Datos 1 (D1)
	Tierra (D0, D1)	19	21	Señal de Tierra (D1)
8	Bit de Datos 2 (D2)	4	4	Bit de Datos 2 (D2)
	Tierra (D2, D3)	20	22	Señal de Tierra (D2)
9	Bit de Datos 3 (D3)	5	5	Bit de Datos 3 (D3)
	Tierra (D2, D3)	20	23	Señal de Tierra (D3)
10	Bit de Datos 4 (D4)	6	6	Data Bit 4 (D4)
	Tierra (D4, D5)	21	24	Señal de Tierra (D4)
11	Bit de Datos 5 (D5)	7	7	Bit de Datos 5 (D5)
	Tierra (D4, D5)	21	25	Señal de Tierra (D5)
12	Bit de Datos 6 (D6)	8	8	Bit de Datos 6 (D6)
	Tierra (D6, D7)	22	26	Señal de Tierra (D6)
13	Bit de Datos 7 (D7)	9	9	Bit de Datos 7 (D7)
	Tierra (D6, D7)	22	27	Señal de Tierra (D7)
14	C2 (nInit)	16	31	C2 (nInit)
	Tierra (C1, C2, C3)	25	30	Señal de Tierra (C2)
15	C0 (nStrobe)	1	1	C0 (nStrobe)

	Tierra (C0)	18	19	Señal de Tierra (C0)
16	C3 (nSelectIn)	17	36	C3 (nSelectIn)
	Tierra (C1, C2, C3)	25	30	Señal de Tierra (C3)
17	C1 (nAutoFd)	14	14	C1 (nAutoFd)
	Tierra (C1, C2, C3)	25	30	Señal de Tierra (C1)
18	Conectados juntos, sin conexión en el PC		18	Lógica Alta (Computador)
			36	Lógica Alta (Periférico)
-	Blindaje			Blindaje

Tabla 2.8

Conexiones entre IEEE 1284-A (DB-25) e IEEE 1284-B (CENTRONICS) con cable par trenzado

Por otro lado, si se usa el nuevo cable con conector 1284-A en un extremo y 1284-B en el otro, en este caso los retornos de tierra para dos o más señales son conectados a un único contacto en el conector. Esto es debido a que, aunque el conector Centronics tiene 36 contactos, el uso convencional no incluye un retorno de tierra para cada señal

Uno de los problemas mas comunes en los cables es la interferencia producida por el acoplamiento de las señales. Métodos de reducción de interferencia incluyen el blindaje de las señales que entran o salen de un cable y la reducción de la amplitud de las señales de interferencia.

2.6.4 BLINDAJE

El blindaje metálico es una efectiva alternativa para bloquear ruido debido al acoplamiento capacitivo, electromagnético y acoplamiento de alta frecuencia. Un buen cable paralelo tiene un blindaje metálico rodeando los conductores y extendiéndose hasta la chasis del conector en si.

Los cables IEEE-1284-compliant, tienen dos capas de blindaje. Un forro sólido de aluminio o poliéster rodea los conductores y este es a su vez es rodeado por blindaje trenzado .

El estándar recomienda el uso de cables de calibre AWG28 o menor.

2.6.5 PAR TRENZADO

Otra forma de reducir interferencia es el uso de pares trenzados.
Un par trenzado tiene dos cables aislados que se trenzan uno con otro a lo largo de la longitud del cable.

La IEEE 1284 especifica un máximo de 36 trenzados por metro de longitud.

El trenzado reduce interferencia de acoplamiento magnético, especialmente para señales de baja frecuencia. (El cambio de voltaje en los cables origina que se produzca un campo magnético, lo cual induce corriente en los cables dentro de dicho campo).

Los campos originados por el cable de señal y su tierra de retorno, tienen polaridades opuestas. Cada trenzado permite a los cables intercambiar posiciones físicamente. Esto causa que los campos magnéticos opuestos generados por cada uno de estos cables se anulen entre sí. De forma similar el trenzado también reduce la radiación electromagnética emitida por el par.

Un cable que cumpla los requerimientos del estándar es etiquetado como IEEE std 1284-1994 compliant .

2.6.6 MANEJADORES Y RECEPTORES

La mayoría de los circuitos de puertos paralelos usan lógica TTL, y en el mejor de los casos usan manejadores y buffers. El puerto paralelo original tiene un flip-flop 74LS374 para manejar las ocho líneas de Datos, un inversor de colector abierto

7405 para manejar las líneas de Control, y las entradas de las líneas de Estado son conectadas a compuertas lógicas LSTTL.

Hoy en día no hay una forma exacta de conocer los componentes que el PC o un periférico puedan usar para los circuitos del puerto paralelo. Sin embargo todos los puertos paralelos tienen las mismas 17 líneas de señal, y estas pueden diferir en características como impedancia de salida e inmunidad al ruido.

Las salidas de todos los puertos paralelos deben tener al menos la misma capacidad de manejo de corriente que mantenía el puerto original.

El estándar IEEE 1284 especifica las características para los drivers y receivers del puerto paralelo. Esta describe dos tipos de dispositivos:

- Dispositivos de Nivel 1, que son similares a los usados en el diseño del puerto paralelo original.
- Dispositivos de Nivel 2, que ofrecen un mejor rendimiento mientras se mantienen compatibles con la interface original.

Un puerto con drivers y receivers de Nivel 2, puede conectarse a un puerto con drivers y receivers de Nivel 1 sin inconvenientes; aunque no se tendrán los mismos beneficios que si se estuviera trabajando solo con dispositivos de Nivel 2. Ambos niveles asumen un voltaje de fuente de +5V.

2.6.6.1 Dispositivos de Nivel 1

Las especificaciones para drivers y receivers de Nivel 1 son cumplidas por componentes LSTTL, TTL y HCTMOS, incluidos en el puerto paralelo original.

2.6.6.1.1 Drivers

- Salidas de estado lógico alto: +2.4V mínimo, con un suministro de corriente de 0.32mA.

- Salidas de estado lógico bajo: +0.4V máximo, con un drenaje de corriente de 12mA.
- Resistores de Pull-Up (si se usaran): 1.8K mínimo, sobre las líneas de estado y control, 1K mínimo en líneas de datos.

En operación normal, las salidas no proveen sus máximas tasas de corriente, pero la habilidad para suministrar y drenar altas capacidades de corriente son un indicativo de que las salidas tienen baja impedancia, lo que significa que la salida puede conmutar rápidamente. A medida que una salida conmuta, el voltaje debe cargarse o descargarse a través de la capacitancia del cable por tanto, mientras más baja sea la impedancia, más rápido cambiará el voltaje.

Las compuertas lógicas LSTTL garantizan cumplir con los requerimientos de Nivel 1. Las compuertas TTL por su parte no cumplen todos estos requerimientos.

Aunque la familia de CIs HCMOS son equivalentes a la mayoría de chips LSTTL, las hojas de datos no especifican suficiente información que garantice que estos chips cumplen el requerimiento del Nivel 1.

2.6.6.1.2 Receivers

- Entradas de estado lógico alto: 2.0V máximo, con un drenaje de corriente de 0.32mA.
- Entradas de estado lógico bajo: 0.8V mínimo, con un suministro de corriente de 12mA.
- Resistores de Pull-Up (si se usaran): los valores mínimos recomendados son 470 ohms sobre las líneas de estado y control, 1K en las líneas de datos.
- Tiempo de subida y bajada (entre 0.8V y 2.0V): 120ns máximo.
- Limites de entrada: las entradas deben soportar trascientes de voltaje desde –2.0V a +7.0V.

Cualquier entrada LSTTL o HCTMOS, cumple los requerimientos arriba sugeridos.

Los chips HCMOS no son una buena elección, debido a que el mínimo voltaje garantizado para una entrada lógica alta es 3.5V, es decir 1.5V más altos que los 2V (TTL-compatible) requeridos. De esta forma si se usa un chip HCMOS, es necesario añadir una resistencia de pull-up desde la entrada a +5V. Los dispositivos HCTMOS tienen entradas TTL compatible, lo que significa que no necesitan resistencias de pull-up.
A pesar de que las especificaciones no lo mencionan, las entradas Schmitt-trigger, proporcionan una alta inmunidad al ruido, por su característica de histéresis.

2.6.6.2 Dispositivos de Nivel 2

Los dispositivos de Nivel 2 tienen drivers de gran capacidad y entradas con histéresis.

2.6.6.2.1 Drivers

- Salidas de estado lógico alto: +2.4V mínimo, con un suministro de corriente de 12mA. Esta capacidad es mayor que para los Niveles 1, con solo 0.32mA de corriente.
- Salidas de estado lógico bajo: +0.4V máximo, con un drenaje de corriente de 12mA. Este es el mismo drenaje que para la especificación de Nivel 1.
- Impedancia de salida de los manejadores: 45-55 ohms entre ($V_{OH} - V_{OL}$).
- Tasa de cambio del manejador: 0.05 a 0.40 V/nsec.

Los drivers LSTTL no pueden drenar suficiente cantidad de corriente para cumplir la especificación.

HC(T)MOS tienen una característica de igual suministro y drenaje de corriente, pero no son lo suficientemente fuertes para cumplir el mínimo estándar.

Las salidas de los nuevos chips controladores incluidos SMC y National, cumplen los requerimientos del Nivel 2

2.6.6.2.2 Receivers

- Entradas de estado lógico alto: 2.0V máximo, con un drenaje de corriente de 20uA.
- Entradas de estado lógico bajo: 0.8V mínimo, con un suministro de corriente de 20uA.
- Histéresis del Receptor: 0.2V mínimo. Una mayor histéresis, hasta 1.2V, ofrecerá mayor inmunidad al ruido.

2.7 MODOS DE TRANSFERENCIA DE DATOS

El estándar IEEE1284 definen cinco tipos de transferencia de datos: Compatible, Nibble, Byte, EPP y ECP. Todos estos estuvieron en uso antes de esta publicación, pero el estándar provee una referencia para que los diseñadores de circuitos y programadores puedan implementarlo en sus productos.

El estándar describe además una negociación de software (protocolo de negociación). Esta hace posible que el PC y el periférico decidan sobre que protocolo usar para la transferencia de datos.

Desafortunadamente, algunos pocos aspectos como ajustes básicos y procedimientos de configuración del puerto no están estandarizados y varían con el chip controlador del puerto.

La transferencia de datos del puerto paralelo es normalmente asincrónica. Los dispositivos involucrados en la transferencia no comparten un reloj común y la transferencia de datos se realiza de manera esporádica.

La IEEE 1284 usa los términos *canal directo* para referirse a transferencias desde el PC (host) al periférico, y *canal reverso* para referirse a transferencias desde el periférico al PC.

2.7.1 DISPOSITIVOS COMPATIBLES Y EN CONCORDANCIA

El estándar 1284 define los términos *compatible (compatible)* y *compliant (en concordancia)*.

De acuerdo a este estándar, dispositivos *compatibles 1284* incluyen cualquier dispositivo que pueda usar el modo compatible. Esto incluye el SPP y todos aquellos que emulan su comportamiento con sus variantes.

Por otro lado, un dispositivo denominado *en concordancia 1284* debe también soportar el protocolo de negociación IEEE 1284. El único requerimiento adicional que debe cumplir este dispositivo en acordancia es que debe soportar el modo Nibble.

Un SPP u otro dispositivo compatible puede también usar el modo Nibble, pero no soporta el protocolo de negociación, de forma que el PC y el periférico deben tener otra manera de seleccionar el modo de transferencia.

De acuerdo con las definiciones de IEEE 1284, ninguno de los dos tipos de dispositivos, ya sean compatibles o en concordancia, tienen que soportar los modos EPP o ECP.

2.7.2 MODO DE NEGOCIACIÓN

Cuando el host y un periférico desean comunicarse, estos necesitan una forma para decidir en que modo deben hacerlo. La fase de negociación IEEE1284, permite a los dispositivos llegar a un acuerdo sobre el mejor modo a usar.

A través de la negociación, el PC puede determinar que modos soporta el periférico. Si un periférico soporta múltiples modos, la negociación le indica al periférico que modo a establecido el PC.

Un dispositivo en concordancia con IEEE1284, negociará de acuerdo al estándar.

Si el PC tratara de negociar con un dispositivo que no responde a la negociación, el PC debe elegir por defecto el modo compatible.

2.7.2.1 Protocolo de Negociación

Los siguientes pasos describen el protocolo de negociación:

1. El PC pregunta por un modo, escribiendo un valor de petición a las líneas de datos (D0-D7), luego lleva 1284 Activo (C3) a alto, y HostBusy (C1) a bajo.
2. En respuesta, el periférico 1284-compliant (en concordancia) lleva a bajo PtrClk (S6), y pone las señales AckDataReq (S5), XFlag (S4) y nDataAvail (S3) a alto. Sobre un periférico que no sea 1284-compliant es improbable que estos eventos ocurran, pues en un dispositivo que no soporta negociación, el llevar a alto la señal S3 (PaperEnd), originan que S5 (nError) vaya a bajo. Por tanto, si las señales PaperEnd (S3) y nError (S5) no son llevadas a alto, el periférico no soporta negociación IEEE1284, entonces el host lleva 1284Active (C3) a bajo para terminar la negociación. El enlace puede entonces usar el modo compatible para la transferencia de datos.
3. Si el periférico soporta negociación, el PC lleva HostClk (C0) a bajo para indicar al periférico que puede leer el valor de petición de protocolo. El PC luego lleva HostClk (C0) y HostBusy (C1) a alto.
4. En respuesta, el periférico lleva AckDataReq (S5) a bajo. Si soporta el modo requerido, entonces mantiene XFlag (S4) en alto. La excepción es el modo Nibble, para el cual el XFlag (S4) va a bajo. Por otra parte si el

periférico tiene que enviar datos al PC, indica esta condición llevando a nDataAvail a bajo.

5. El periférico lleva PtrClk (S6) a alto para completar la negociación.

Si el modo solicitado no es soportado por el periférico, el PC retorna al modo Compatible y puede tratar nuevamente de negociar utilizando un byte diferente de petición de modo.

Todos los puertos paralelos de los PCs, pueden usar dos modos de comunicación: Modo Compatible, que realiza transferencias de 8 bits de PC a periférico y Modo Nibble, para transferencias de cuatro bits de periférico a PC.

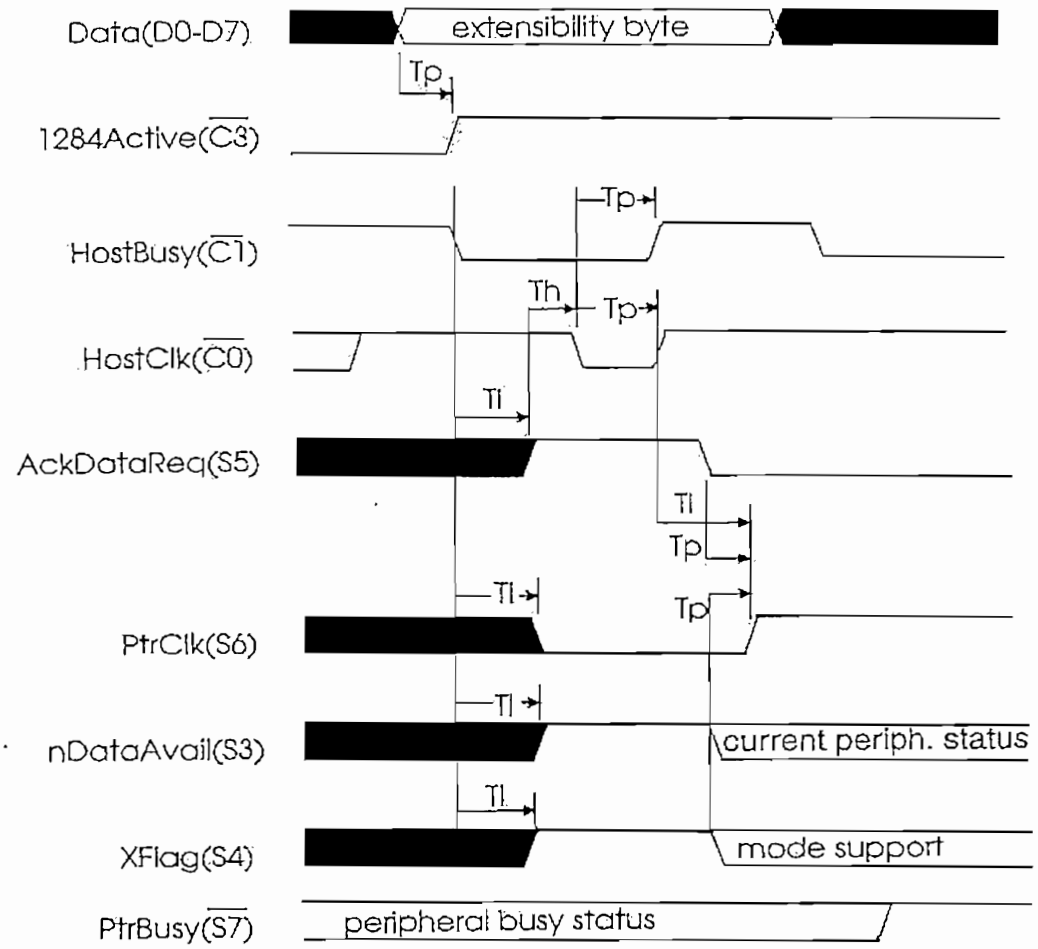


Figura 2.4

Protocolo de Negociación

2.7.3 MODO COMPATIBLE

El Modo Compatible es el modo por defecto para envío de datos desde el PC al periférico. El PC escribe bytes al puerto de datos, y los bits de Estado y Control proveen handshaking. El handshaking para .modo Compatible descrito en el IEEE1284, es compatible con el handshaking de la rutina BIOS original del PC para transferencias de puerto paralelo.

2.7.3.1 Handshaking Modo Compatible

Tres de las señales del puerto paralelo son usadas para handshakig.

El handshaking realiza algunas funciones; así, la línea Busy (S7) le indica al PC que el periférico está listo para recibir datos. Por su parte, la línea nStrobe (C0) del PC anuncia al periférico que un byte en las líneas de Datos (D0-D7) está listo para ser leído. Luego de que el periférico lee el byte, envía un pulso nAck (S6) para indicar al PC que el byte fue recibido.

No siempre es necesario utilizar las señales de handshaking. Normalmente son adecuadas cuando el dispositivo receptor requiere de un aviso previo que le indique que debe prepararse para recibir datos, o cuando se debe enviar acuses de recibo por cada byte recibido.

A continuación se describe una transferencia de datos con handshaking para el Modo Compatible:

1. Cuando el PC quiere comunicarse, lleva SelectIn (C3) a bajo. En respuesta, el periférico lleva Select (S4) a alto. El PC entonces, lee el puerto de Estado y verifica que Select (S4) sea alto y Busy (S7) sea bajo.
2. Si las condiciones antes mencionadas se cumplen, el PC escribe el byte a enviar en D0-D7.
3. Luego de un retardo de al menos 0.75 useg, el PC pulsa a bajo nStrobe (C0). Este pulso es típicamente de 1 a 5 useg, pero puede ser configurado

de 0.75 a 500 useg. El PC mantiene D0-D7 validos por al menos 0.75 useg luego de que nStrobe (C0) retorna a alto.

4. En el flanco de bajada de nStrobe (C0), el periférico lee y almacena los datos recibidos en D0-D7. Una vez hecho esto, el periférico lleva Busy (S7) a alto, dentro de 0.5 useg desde que nStrobe (C0) pasó a bajo. Esto indica al computador que no envíe mas datos.
5. Cuando el periférico ha leído y almacenado los datos, pulsa nAck (S6) a bajo para indicarle al PC ésta circunstancia. El pulso de nAck (S6) es típicamente 5 useg, pero puede ser desde 0.5 a 10 useg. El PC puede usar nAck (S6) como una interrupción que le indique al PC cuando escribir el próximo byte de datos. Si el periférico esta listo para recibir otro byte, lleva otra vez Busy (S7) a bajo, y una nueva transferencia puede empezar.

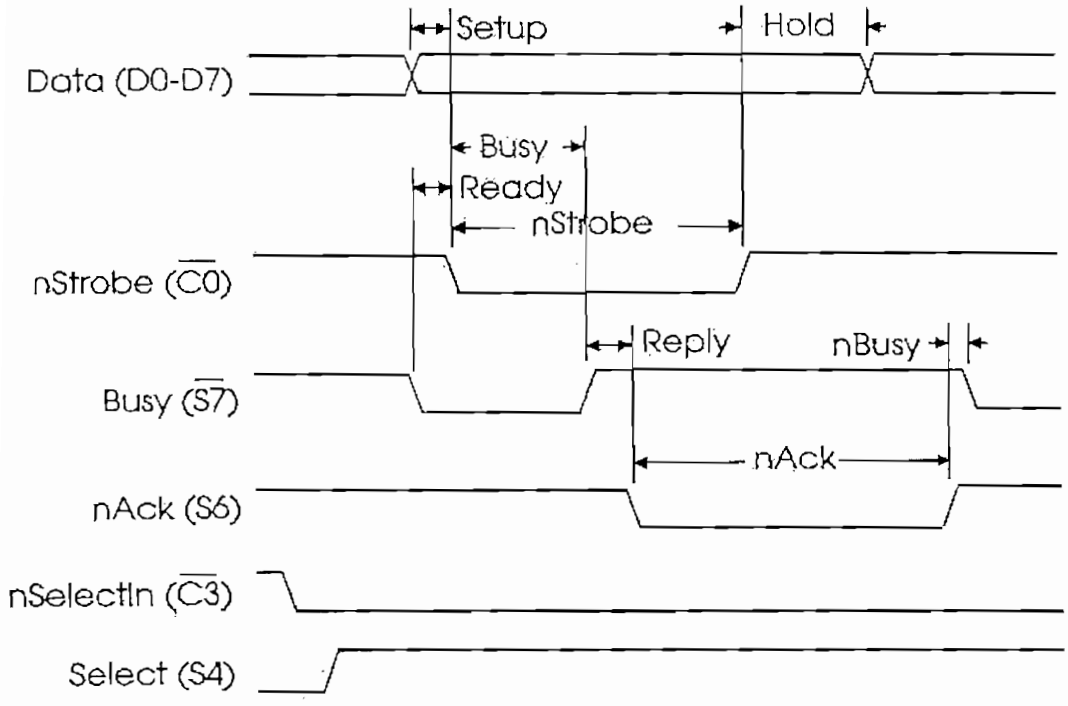


Figura 2.5
Transferencia en Modo Compatible

2.7.3.2 Variaciones

Debido a que IBM no documentó completamente los requerimientos de tiempo y algunos otros detalles del puerto paralelo de los PCs originales, fabricantes de computadoras compatibles y periféricos tuvieron que recurrir al diseño original del puerto paralelo y de las funciones BIOS del PC, para diseñar puertos que fueran compatibles con este.

Con muchos fabricantes haciendo PCs, impresoras y otros dispositivos de puerto paralelo, algunas variaciones fueron implementadas sobre el diseño original, por lo que no se puede asegurar un solo estándar de tiempo para todos los puertos desarrollados. Hoy en día, el relativamente nuevo estándar IEEE1284 puede servir como base para los nuevos diseñadores.

Por lo antes mencionado resulta difícil desarrollar un programa que pueda manejar el handshaking de modo Compatible para varios puertos. Una alternativa para este propósito es implementar un programa que obvie la temporización, y que se ciña solamente a cumplir el handshake en cuanto a las señales de control se refiere, donde cada señal de control permanece validada hasta que el extremo opuesto la reconoce. Esta es la consideración tomada para el desarrollo del presente proyecto de titulación.

2.7.4 MODO NIBBLE

El Modo Nibble permite a cualquier puerto paralelo recibir bytes de datos desde un periférico. El periférico usa las cuatro líneas de Estado para enviar un byte de datos dividido en dos Nibble. Las transferencias Nibble estuvieron en uso mucho antes de la promulgación del IEE1284, pero este estándar formaliza un protocolo.

2.7.4.1 Handshaking Modo Nibble

Existen dos fases asociadas con el modo Nibble. La fase de transferencia de datos, que incluye la escritura de un byte desde el periférico al PC, y la fase de inactividad, que define los estados de señal cuando no ocurre una transferencia.

A continuación se describe el protocolo para una transferencia en modo Nibble:

1. El PC lleva HostBusy (C1) para indicar que esta listo para aceptar el primer Nibble desde el periférico.
2. El periférico escribe los bits de datos D0-D3 en las líneas S3, S4, S5 y S7. Y lleva PtrClk (S6) a bajo para indicar que el Nibble esta listo para ser leído.
3. El PC lee los cuatro bits de datos y lleva HostBusy (C1) a alto para indicar que ha recibido el Nibble.
4. El periférico lleva PtrClk (S6) a alto.
5. Cuando el PC esta listo para el segundo Nibble, lleva HostBusy (C1) a bajo.
6. El periférico ubica los bits de datos D4-D7 sobre las líneas de Estado S3, S4,S5 y S7.Y lleva PtrClk (S6) a bajo para indicar que el siguiente Nibble esta listo para ser leído.
7. El PC lee los cuatro bits de datos y lleva HostBusy (C1) a alto para indicar que ha recibido el Nibble.
8. El periférico fija los bits de Estado como sigue:
   PtrBusy (S7), alto si el periférico está ocupado y bajo si no lo esta
   nDataAvail (S3), bajo si hay otro byte para enviar y alto si no lo hay
   AckDataReq (S5), igual que S3
   XFlag (S4), igual que S5
9. El periférico lleva PtrClk (S6) a alto
10. El PC lee nDataAvail (S3) para determinar si hay algún byte disponible para ser leído, y PtrBusy (S7) para averiguar si el periférico esta ocupado. De haber recibido un nuevo byte, el PC puede hacer lo siguiente:
    Lleva HostBusy (C1) a bajo y aguarda por más datos.

Mantiene HostBusy (C1) en alto para prevenir que el periférico envíe otro Nibble.

Lleva 1284Active (C3) a bajo para regresar al modo Compatible.

Bit de Estado	Nibble 1	Nibble 2
S3	D0	D4
S4	D1	D5
S5	D2	D6
S7	D3	D7

Tabla 2.9

Bits de Datos Modo Nibble

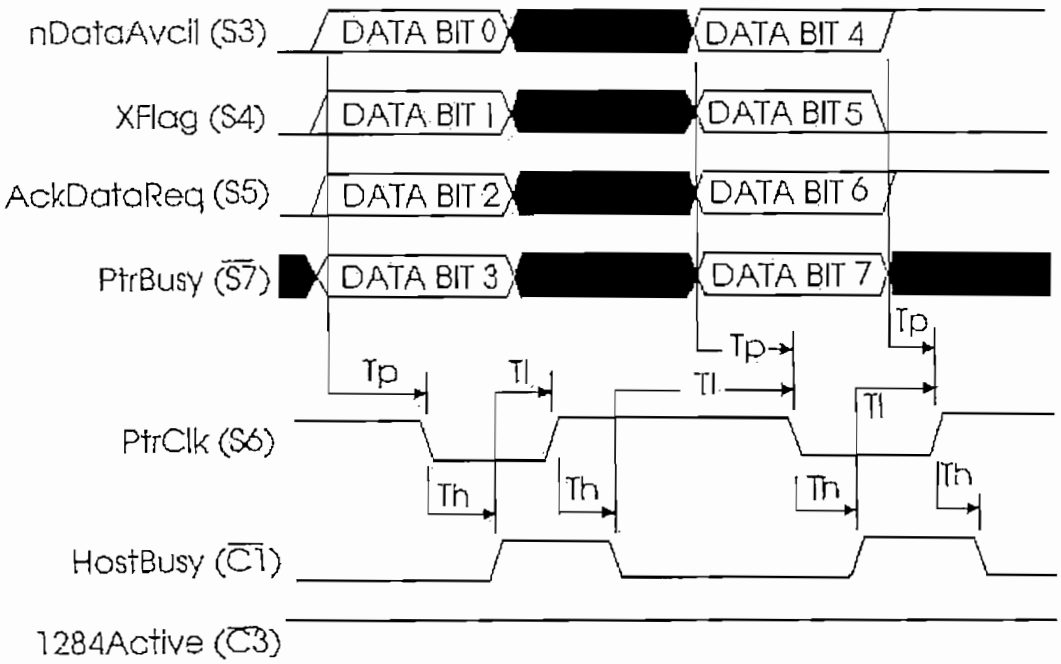


Figura 2.6

Transferencia Modo Nibble

Si se esta programando una interface para ambos extremos, no es necesario cumplir a cabalidad la asignación de bits que propone IEEE1284. Por ejemplo, se

puede usar los bits de Estado S4-S7 como bits de Datos y S3 para Handshaking. Además las consideraciones de temporización, como en el caso anterior, pueden también ser obviadas, preocupándose solamente de cumplir el handshaking para las señales de Control.

2.7.5 MODO BYTE

Muchos puertos paralelos incluyendo el tipo PS2, EPPs y ECPs, tienen puertos bidireccionales de datos. Estos puertos pueden usar el modo Byte para transferencias de periférico a PC de 8 bits, donde el periférico escribe un byte a la vez al puerto de Datos, en lugar de tener que dividir el byte en dos Nibble y escribirlos en secuencia por el puerto de Estado.

2.7.5.1 Handshaking Modo Byte

Para usar el modo Byte, el PC debe tener un puerto de Datos bidireccional y el periférico debe ser capaz de escribir un byte a sus líneas de Datos. El Modo Byte IEEE1284, describe un protocolo de handshaking para transferencias en Modo Byte.

Al igual que el handshake para el modo Compatible, el handshake de modo Byte incluye la señal Busy para indicarle al periférico cuando es posible enviar un byte, y la señal Strobe, para indicar al PC que un dato esta disponible.

En transferencias de modo Byte, los requerimientos de temporización no son rigurosos, pues una señal de control permanece validada hasta ser reconocida por el otro extremo.

A continuación se describe el protocolo de transferencia en Modo Byte:

1. EL Host deshabilita las salidas D0-D7 . En la mayoría de puertos bidireccionales, llevando el bit C5 a alto se logra esto.
2. El PC lleva HostBusy (C1) a bajo, para indicar que está listo para recibir datos.

3. El periférico ubica los datos sobre D0-D7, y lleva PtrClk (S6) a bajo.
4. En respuesta el PC lee D0-D7 y lleva HostBusy (C1) a alto. Luego el PC lleva HostClk (C0) a bajo.
5. El periférico fija los bits de Estado como sigue:
   PtrBusy (S7), alto si el periférico está ocupado y bajo si no lo esta
   nDataAvail (S3), bajo si hay otro byte para enviar y alto si no lo hay
   AckDataReq (S5), igual que S3
   XFlag (S4), igual que S5
6. El periférico lleva PtrClk (S6) a alto.
7. El PC lleva HostClk (C0) a alto para indicar que ha recibido el byte.
   De haber recibido un nuevo byte, el PC puede hacer lo siguiente:
   Lleva HostBusy (C1) a bajo y aguarda por más datos.
   Mantiene HostBusy (C1) en alto para prevenir que el periférico envíe otro Byte.
   Lleva 1284Active (C3) a bajo para regresar al modo Compatible.

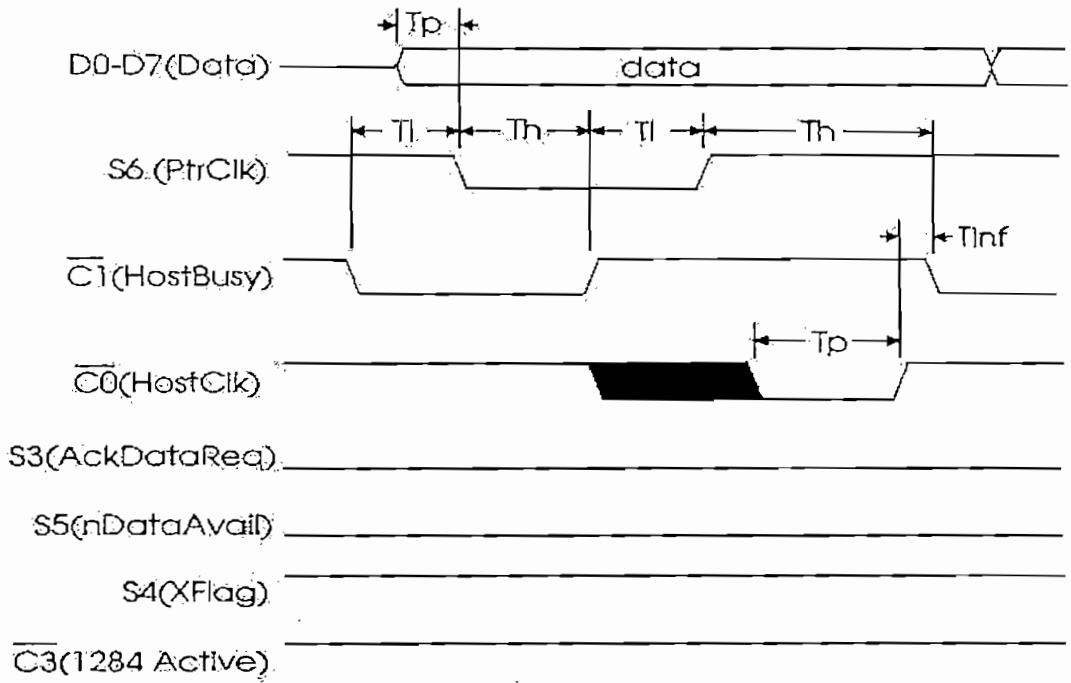


Figura 2.7

Transferencia Modo Byte

Como se analizó para los modos anteriores, no es necesario cumplir a cabalidad las consideraciones de temporización, bastaría con cumplir el handshaking para las señales de Control.

2.7.6 MODO EPP (Enhanced Parallel Port)

Un EPP puede transferir datos a altas velocidades en ambas direcciones y puede distinguir entre dos tipos de información: datos y direcciones. Debido a que pueden conmutar direcciones rápidamente, el EPP es excelente para dispositivos que intercambian pequeños bloques de datos con frecuentes cambios de dirección, como manejadores de disco externos o interfaces de red.

Un EPP puede leer o escribir un byte de datos en un solo ciclo de bus ISA, o en alrededor de 1us, incluyendo handshaking. Las líneas de datos son bidireccionales, y una señal de control determina la dirección del puerto de datos. Otras dos señales de control distinguen si la información en las líneas de datos corresponden a direcciones ó datos.

Las transferencias EPP difieren de transferencias de modo Compatible, Nibble y Byte en que el hardware del puerto genera automáticamente las señales de control y detecta la respuesta del lado opuesto (periférico conectado). Esto significa que no se necesitan instrucciones en el programa para cambiar una salida Strobe o leer una entrada Busy. El handshake es generado automáticamente por el hardware del puerto y permite a un EPP leer o escribir un byte con handshaking, en un ciclo del Bus ISA, en lugar de los cuatro ciclos que toma hacerlo en modo Compatible o en el modo Byte.

Las operaciones de datos y direcciones del EPP usan diferentes señales de control para retener (latching) los bytes dentro del dispositivo receptor. La lectura y escritura de direcciones usan nAStrobe (C3), mientras que la lectura y escritura de datos usan nDStrobe (C1). Esto proporciona al dispositivo receptor una forma simple de distinguir entre los dos tipos de información.

El estándar IEEE1284 no documenta todos los aspectos del EPP. Así, no menciona las características que son específicas a los PCs, tales como: las señales y temporizadores sobre el bus de expansión y los registros adicionales del puerto.

Un EPP usa 8 registros, es decir 5 más que los que utiliza el puerto paralelo original. Los primeros tres registros son muy parecidos a los registros de datos, estado y control de un SPP. Una diferencia es que en el puerto de estado, el bit S0 usualmente indica un timeout de un ciclo EPP.

Algunos EPPs también emulan puertos tipo PS/2, donde poniendo el bit de control C5 a 1 deshabilita las salidas de datos y habilita el uso de las líneas de datos para entradas. En otros EPPs las líneas de datos son de entrada solamente (tipo SPP), excepto cuando se realizan transferencias EPP.

Para transferencias EPP el puerto usa registros adicionales. Para escribir un byte de Datos en modo EPP, se escribe al registro de datos EPP (dirección base + 4), en lugar de escribir a la dirección base.

Escribir al registro de datos del EPP, origina que el puerto inicie un ciclo completo de dato-escritura. El hardware del puerto ubica el byte a escribir en D0-D7, luego realiza automáticamente el handshaking y detecta las respuestas del periférico. De igual forma, al leer un byte desde el registro de datos del EPP, se inicia un ciclo completo de dato-lectura en el puerto.

La transferencia de direcciones es muy parecida a la descrita para la transferencia de datos, excepto que se escribe o lee el registro de direcciones del EPP (dirección base + 3), lo cual origina que el puerto inicie un ciclo completo de dirección–escritura o dirección–lectura, según sea el caso. Los ciclos son idénticos a los ciclos de datos del EPP, con la diferencia de que estos usan una señal de control distinta para transferir el byte dentro del dispositivo receptor.

La función de los registros en dirección base + 5 hasta dirección base + 7 varía. En algunos puertos se puede usar operaciones de lectura o escritura de 16 o 32 bits para acceder al puerto, y estos registros mantendrán él o los bytes adicionales escritos o leídos, con el puerto transfiriendo cada byte en secuencia.

Registros EPP y Funciones		
Nombre del Registro	OFFSET	Uso
SPP/PS2 DATA	0	Lee o escribe a las líneas de datos sin handshaking
SPP Status	1	Lee las 5 líneas de estado (S3-S7). En modo EPP, un bit adicional (S0), indica timeout
SPP Control	2	Lee o escribe las 4 líneas de control (C0-C3); también contiene bits de configuración para habilitación de interrupciones (C4) y control de dirección para el modo byte (C5)
EPP Address	3	Lee o escribe las líneas de datos con handshaking de ciclo de direcciones
EPP Data	4	Lee o escribe las líneas de datos con handshaking de ciclo de datos
(Varies)	5	Puede ser usado para transferencias de datos de 16/32 bits, configuración del puerto o definida por el usuario
(Varies)	6	Puede ser usado para transferencias de datos de 16/32 bits, configuración del puerto o definida por el usuario
(Varies)	7	Puede ser usado para transferencias de datos de 16/32 bit, configuración del puerto o definida por el usuario

Tabla 2.10

Registros EPP y Funciones

La dirección base de un EPP es normalmente 378h o 278h, con el puerto usando un rango de direcciones 378h-37Fh o 278h-27Fh según corresponda.
Los EPPs normalmente no usan la dirección base 3BCh, debido a que el display de video puede usar los bytes siguientes a 3BEH y esto ocasionaría conflictos.

2.7.6.1 Handshaking Modo EPP

El modo EPP soporta 4 operaciones: escritura de direcciones, escritura de datos, lectura de direcciones y lectura de datos. Cada una tiene un handshake distinto.

En muchos puertos, antes de acceder a los registros EPP e iniciar una transferencia, los bits del puerto de control C0,C1 y C3 deben estar altos (hay que recordar que el hardware del puerto invierte estos bits, entonces para llevarlos a alto se debe escribir 0 a los bits correspondientes de registro).

Estos son los pasos en los cuatro tipos de transferencia EPP, descrita en IEEE 1284.

2.7.6.2 Escritura de Direcciones (Transferencia Directa)

1. Las salidas de datos del periférico son deshabilitadas y nWait (S7) es baja. El PC lleva nWrite (C0) a bajo y escribe una dirección en el registro de direcciones EPP, lo cuál origina que el byte aparezca en D0-D7, para luego llevar nAStrobe (C3) a bajo.
2. El periférico lleva nWait (S7) a alto, para señalar que está listo para retener las direcciones.
3. El PC lleva nAStrobe (C3) a alto, causando que el periférico retenga la dirección.
4. Cuando el periférico esta listo para otro byte, lleva nWait (S7) a bajo.

La figura muestra las señales para un ciclo escritura-dirección.

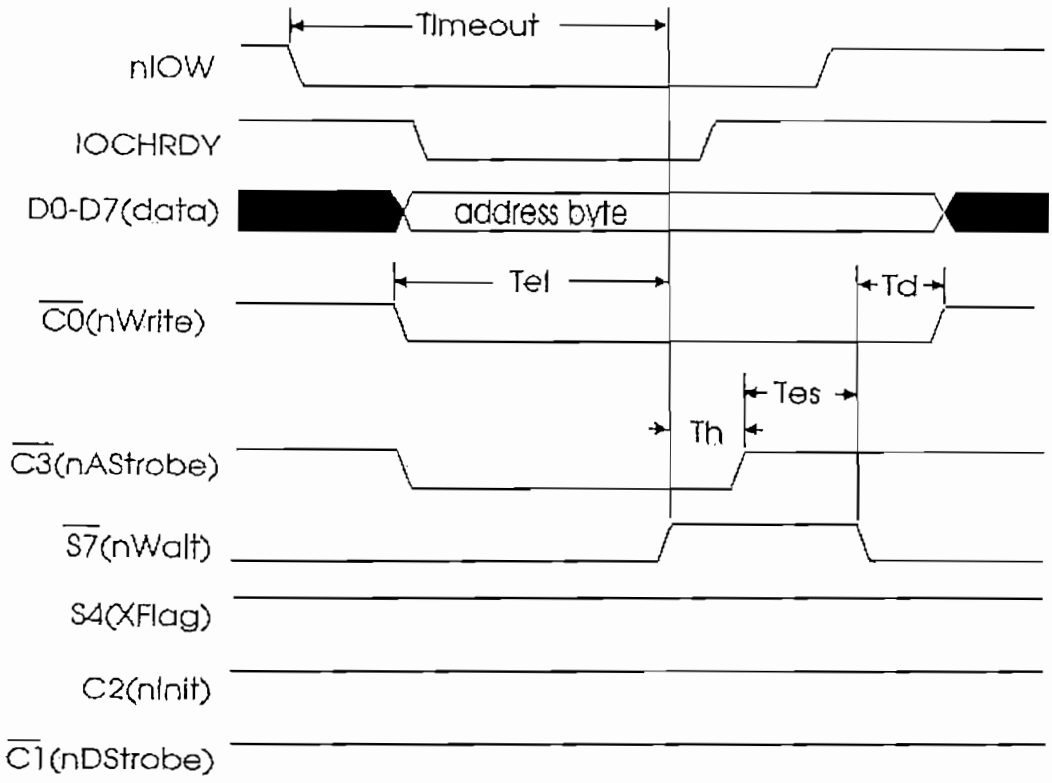


Figura 2.8

Ciclo de Escritura de Direcciones EPP

2.7.6.3 Escritura de Datos (Transferencia Directa)

La escritura de datos es idéntica a la escritura de direcciones, excepto que el PC usa nDStrobe en vez de nAStrobe.

1. Las salidas de datos del periférico son deshabilitadas y nWait (S7) es baja. El PC lleva nWrite (C0) a bajo, y escribe un dato en el registro de datos EPP, lo cuál origina que el byte aparezca en D0-D7, para luego llevar nDStrobe (C1) a bajo.
2. El periférico lleva nWait (S7) a alto, para indicar que está listo para retener el dato.
3. El PC lleva nDStrobe (C1) a alto, causando que el periférico retenga los datos.
4. Cuando el periférico esta listo para otro byte, lleva nWait (S7) a bajo.

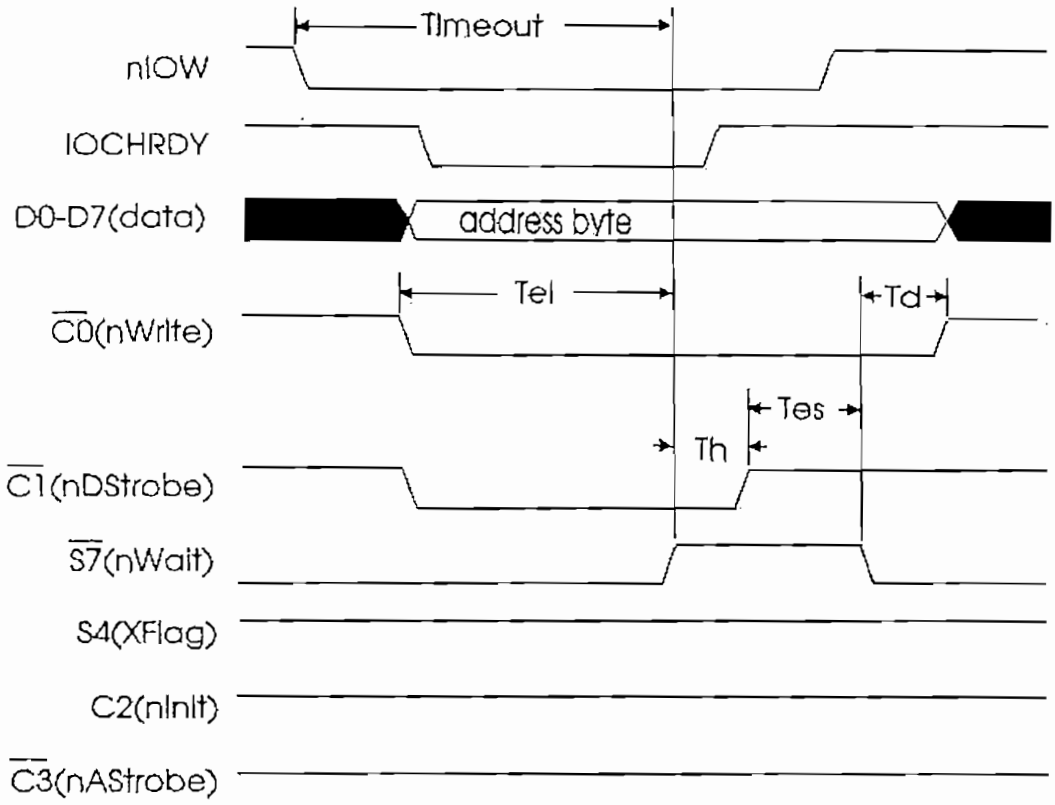


Figura 2.9

Ciclo de Escritura de Datos EPP

2.7.6.4 Lectura de Direcciones (Transferencia Reversa)

Para un ciclo lectura-dirección, el PC usa nAStrobe como en una escritura de dirección, pero lee el registro de dirección del EPP en lugar de escribir en él.

1. El nWait (S7) del periférico es bajo. El PC lleva nWrite (C0) a alto, deshabilita las salidas D0-D7, y lleva nAStrobe (C3) a bajo.
2. El periférico habilita sus salidas D0-D7, escribe una dirección, y lleva nWait (S7) a alto para indicar al PC que la dirección está disponible para ser leída.
3. El PC lee D0-D7 en el registro de direcciones EPP y lleva nAStrobe (C3) a alto.
4. El periférico deshabilita las salidas D0-D7 y lleva nWait (S7) a bajo.

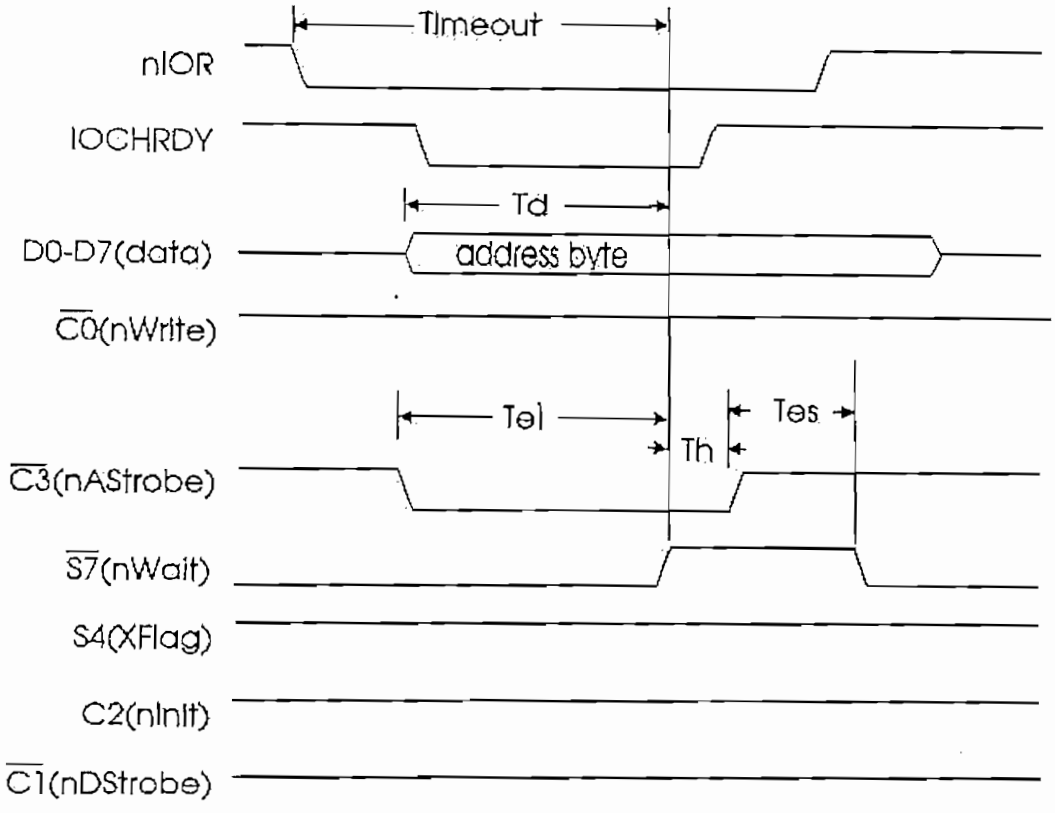


Figura 2.10

Ciclo de Lectura de Direcciones EPP

2.7.6.5 Lectura de Datos (Transferencia Reversa)

La lectura de datos es idéntica a la lectura de direcciones, excepto que el PC usa nDStrobe (C1) en lugar de nAStrobe (C3).

1. El nWait (S7) del periférico debe estar bajo. El PC lleva nWrite (C0) a alto, deshabilita las salidas D0-D7, y lleva nDStrobe (C1)a bajo.
2. El periférico habilita sus salidas D0-D7, escribe un dato, y lleva nWait (S7) a alto para indicar al PC que el dato está disponible para ser leído.
3. El PC lee D0-D7 en el registro de datos EPP y lleva nDStrobe (C1) a alto.
4. El periférico deshabilita las salidas D0-D7 y lleva nWait (S7) a bajo.

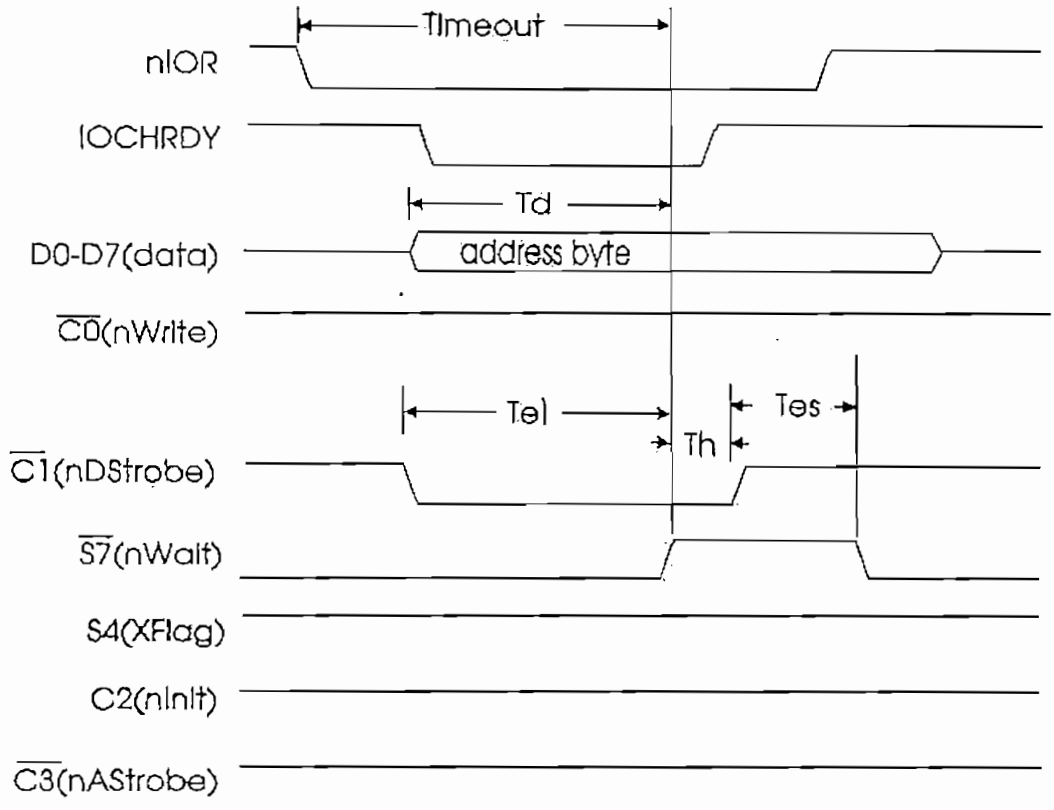


Figura 2.11

Ciclo de Lectura de Datos EPP

Para mayor información acerca de Puertos EPP refiérase al ANEXO A

2.7.7 MODO ECP (Extended Capabilities Port)

El Puerto de Capacidades Extendidas o ECP, provee otra forma rápida de transferencia de datos a través del puerto paralelo. Al igual que en un EPP, las transferencias ECP se realizan en un ciclo del bus ISA, o alrededor de 1useg.

Convencionalmente un ECP tiene un buffer de 16 bytes para almacenar los datos a ser enviados y recibidos. Para aumentar la velocidad de transferencia, un ECP usa compresión de datos para empacar la información dentro de unos pocos bytes, y también usa acceso directo a memoria (DMA). El DMA permite al CPU

realizar otras tareas mientras los datos son transferidos entre el buffer y la memoria.

El handshaking de hardware que utiliza un puerto ECP, no implementa timeout automático, por tanto, la velocidad con que se realizan las transferencias puede adecuarse a la velocidad del periférico usado. Un ECP puede emular puertos tipo SPP y PS/2. Además incluye el modo Fast Centronics, para mejorar las transferencias con periféricos SPP. Algunos ECPs pueden realizar transferencias EPP.

Como sucede con el EPP, muchas de las convenciones para puertos ECPs no están documentadas en el estándar IEEE1284; por ejemplo, no se menciona los registros del ECP o el uso de FIFO.

La información adicional está contenida en un documento de Microsoft titulado "The IEEE1284 Extended Capabilities Port Protocol and ISA Interface Standard".

El puerto ECP utiliza señales de control para distinguir entre bytes de datos y comandos.

Para transferencias directas, la señal de control es C1 (HostAck). Para transferencias reversas, la señal de control es S7 (Periph-Ack).

En ambos casos la señal es llevada a alto cuando el dispositivo esta enviando datos y es llevada a bajo cuando esta enviando comandos.

Cuando los bytes son de comando, si el bit D7 es 1L, los bits D0-D6 actúan como un canal de direcciones; y, si el bit D7 es 0L, los bits D0-D6 son un contador de longitud usado en la compresión de datos. Además de los tres registros base SPP, un ECP añade tres registros adicionales en las direcciones base + 400H hasta base + 402H.

La función de algunos de estos registros varia dependiendo del modo interno ECP seleccionado. Así, en modo ECP 001, una escritura a la dirección base origina que el puerto intente un ciclo de lectura de direcciones ECP. Para escritura simple en la dirección base, el ECP debe estar en modo 000 ó 001 (SPP ó PS/2).

Nombre	Offset	Modos ECP	Descripción
Data	000	000, 001	Datos SPP/PS2
EcpAFIFO		011	Direcciones FIFO ECP
DSR	001	Todos	Estado SPP
DCR	002	Todos	Control SPP
CFIFO	400	010	FIFO de Datos del Puerto Paralelo (Fast Centronics)
EcpDFIFO		011	FIFO de Datos ECP
TFIFO		110	FIFO de Test ECP
CnfgA		111	Configuración A
CnfgB	401	111	Configuración B
ECR	402	Todos	Registro de Control Extendido

Tabla 2.11
Registros ECP

Un puerto que soporta los tipos ECP y EPP usa once registros en total: los tres registros SPP, los cinco registros EPP en la dirección base + 3 hasta la dirección base + 7 y los tres registros ECP desde la dirección base + 400H hasta la dirección base + 402H.

Los puertos ECP soportan transferencias SPP y PS/2, así como transferencias ECP. También los ECPs soportan el modo Fast Centronics, el cual mejora las

transferencias con un SPP. Muchos ECP pueden soportar también transferencias EPP. La Tabla siguiente muestra los modos internos configurables para un puerto ECP.

Modo (bits 7, 6, 6 de ECR)	Descripción
000	SPP (original)
001	PS/2 (Byte, Bidireccional)
010	Fast Centronics
011	ECP
100	EPP
101	Reverso
110	Test
111	Configuración

Tabla 2.12
Modos Internos ECP

La selección de un modo específico se lleva a cabo a través de los bits 7, 6 y 5 del ECR (dirección base + 402H). En modo 000, ECP se comporta como un SPP. En modo 001 se comporta como un PS/2 (puerto de datos bidireccional). Muchos ECPs soportan el modo 100, el cual hace que el puerto actúe como un EPP.

2.7.7.1 Transferencias ECP

Un ECP puede realizar transferencias directas (PC a periférico) y transferencias reversas (periférico a PC). Para ambas direcciones el byte transferido puede ser de datos ó de comandos.

2.7.7.1.1 Transferencia Directa ECP

Una transferencia ECP se desarrolla como sigue:

1. El periférico debe mantener nAckReverse (S5) en alto. Luego si no está ocupado, mantiene PeriphAck (S7) es bajo.
2. En el PC por su parte HostClk (C0) está en alto. Se debe recordar que el bit 5 del registro de control, (C5) debe ser 0, de forma que estén habilitadas las salidas de datos del puerto.
3. El PC escribe un byte al FIFO de datos ECP o al FIFO de direcciones ECP. Si no hay otros bytes en el FIFO precediendo a éste, entonces el byte es ubicado en D0-D7. En caso de que el byte sea un dato, el PC lleva HostAck (C1) a alto ó lo lleva a bajo si es un comando de direcciones. Por último, el PC lleva HostClk (C0) a bajo.
4. El periférico lleva PeriphAck (S7) a alto.
5. El PC lleva HostClk (C0) a alto.
6. El periférico lee D0-D7 y lleva PeriphAck (S7) a bajo para completar la transferencia.

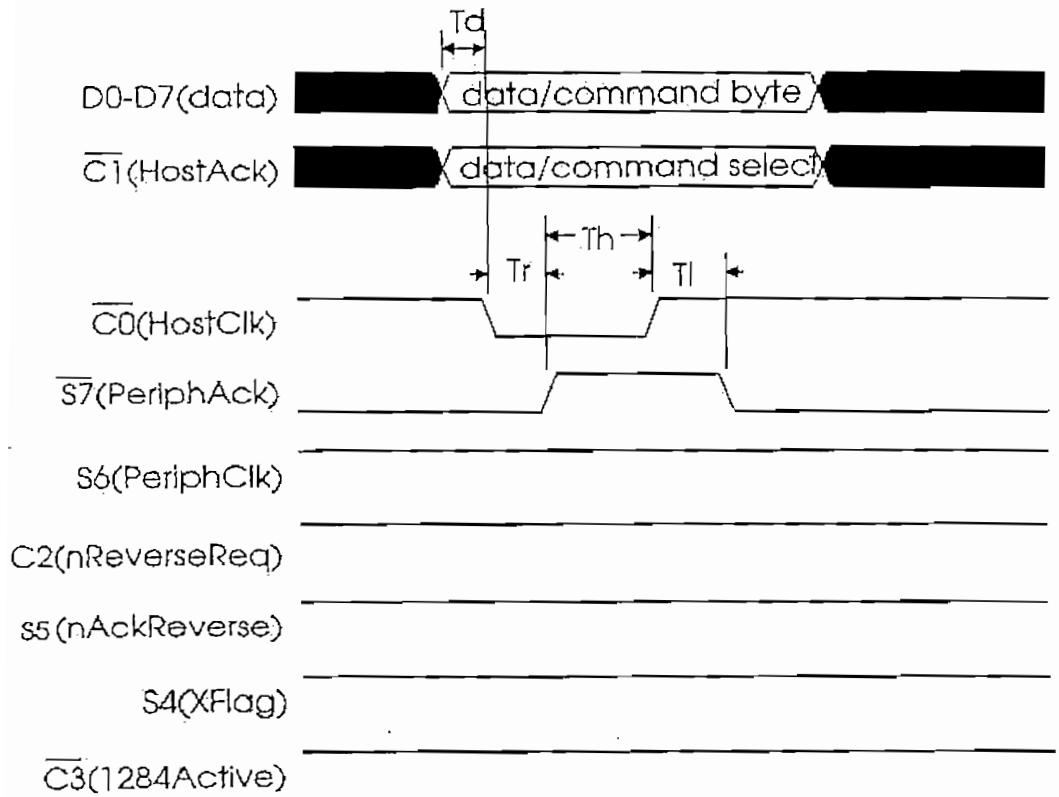


Figura 2.12

Transferencia Directa ECP

2.7.7.1.2 Transferencia Reversa ECP

La transferencia reversa ECP se realiza de la siguiente manera:

1. En el periférico, nAckReverse (S5) es alta y PeriphAck (S7) es baja cuando el periférico no está ocupado.
2. En el PC HostClk (C0) y HostAck (C1) son altas.
3. El PC lleva el bit 5 de control C5 a alto para deshabilitar las salidas de datos. Luego lleva HostAck (C1) a bajo.
4. Luego de al menos 0.5 useg el PC lleva nReverseReq (C2) a bajo.
5. El periférico lleva nAckReverse (S5) a bajo.
6. El periférico habilita sus salidas de datos y escribe un byte al FIFO de datos ECP o al FIFO de direcciones ECP según sea el caso. Si no hay otros bytes en el FIFO precediendo a éste, entonces el byte es ubicado en D0-D7. En caso de que el byte sea un dato, el periférico lleva PeriphAck (S7) a alto ó lo lleva a bajo si es un comando de direcciones. Por último, el periférico lleva PeriphClk (S6) a bajo.
7. El PC lleva HostAck (C1) a alto.
8. El periférico lleva PeriphClk (S6) a alto.
9. El PC lee D0-D7 y lleva HostAck (C1) a bajo para completar la transferencia.

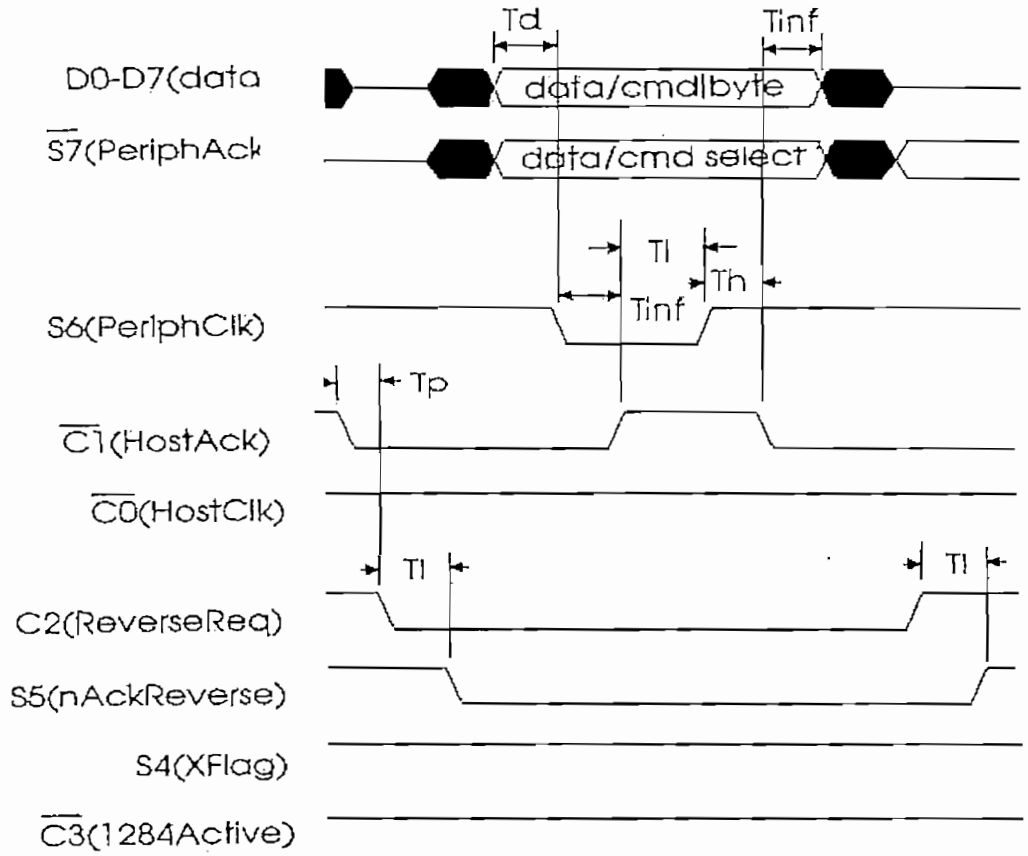


Figura 2.13

Transferencia Reversa ECP

Para mayor información acerca de Puertos ECP refiérase al ANEXO B

CAPÍTULO 3
PUERTO SERIAL

3.1 INTRODUCCIÓN

Al igual que los puertos paralelos, los puertos seriales han sido parte del PC desde sus inicios.

Un puerto serial (COM) frecuentemente tiene una interface RS-232, RS-485 o puede ser usado por el MODEM interno u otro dispositivo. Un PC también puede tener otros tipos de puertos seriales: como el USB, Firewire, I^2C u otros; pero estos usan protocolos diferentes y requieren componentes también diferentes. Las nuevas interfaces seriales son más rápidas y tienen otras ventajas. Es así que si bien las recomendaciones de Microsoft PC 98 permiten el uso de los puertos RS-232, sugieren implementar puertos USB en lugar de estos cuando sea posible.

Sin embargo de estas consideraciones, las interfaces RS-232 continúan en vigencia para aplicaciones como el monitoreo y control de sistemas.

El RS-232 es muy popular debido a que es ampliamente disponible, barato y puede usar cables más largos que la mayoría de otras opciones. El RS-485 es también barato, fácil de añadir y soporta largas distancias, altas velocidades y más nodos que un RS-232.

La interface IrDA (Infrared Data Association) puede usar el mismo UART y formato de datos tal como el RS-232; pero los datos se transmiten como luz infraroja sobre enlaces inalámbricos. El IrDA es adecuado para enlaces cortos con línea de vista entre los dispositivos, siempre que el cableado represente un inconveniente.

La interface MIDI(Musical Instrument Digital Interface) es usado para la transferencia de señales de instrumentos musicales, equipo de control de teatro y otra máquinas controladoras.

Microwire, SPI e I^2C son interfaces seriales sincrónicas, adecuadas para cortas distancias. Muchos microcontroladores tienen una o más de estas interfaces internamente construidas.

USB (Universal Serial Bus) y Firewire (IEEE-1384) son nuevas interfaces inteligentes de alta velocidad para conectar PCs y otros computadores a varios periféricos.

USB fue desarrollado para reemplazar al estándar RS-232 y los puertos de impresora Centronics; así como una interface de Modem y otros periféricos estándar Firewire fue diseñado para transferencias rápidas de video, audio y grandes bloques de datos.

3.2 DEFINICIÓN

Cada puerto de comunicación (COM) en un PC es un puerto serial asincrónico manejado por un UART (Universal Asynchronous Receiver Transmitter).

La implementación serial mas común encontrada en un computador, es la RS232 (EIA232E). Por tanto, en adelante para definir el puerto serial de un PC se referirá a esta interfaz.

La comunicación serial desarrollada en el estándar RS232, define el uso de un conductor para cada dirección de flujo de datos y especifica el envío de los bits del mensaje en secuencia de un bit a la vez, con un tiempo determinado entre bits. La información debe ser separada en palabras de datos, donde la longitud de la palabra es variable y no existe un tiempo fijado entre el envío de cada palabra.

En los PCs se puede elegir la longitud de palabras de entre 5 y 8 bits.

Para una adecuada transferencia de la información, se añaden bits adicionales que sirven para propósitos de sincronización y corrección de errores.

El requerimiento para el envío de información, exige un acuerdo previo entre el transmisor y receptor sobre el baud rate y el formato de transmisión a utilizar; de otra forma, la información puede ser malinterpretada o no reconocida del todo.

A pesar de que el estándar define un sistema serial con tan solo un conductor para cada dirección, también permite el uso de señales adicionales entre dispositivos DTE y DCE. Estas señales adicionales son usadas para manejar la interface de datos e indicar el estado de la comunicación.

La especificación RS232 es adecuada para proveer comunicación confiable hasta una distancia de 50 pies, a tasas de hasta 20000 bps.

El estándar no especifica que patrones de bits se utilizarán para representar la información a ser transmitida. Sin embargo la representación ASCII es la más común y la mas usada.

El RS232 es primariamente un estándar para interfaces DTE a DCE, que define señales de voltaje y tiempo. Esta no discute la forma en que los datos deben ser presentados, formato y protocolos. La definición de estos aspectos son encargados a los desarrolladores de los sistemas de comunicación. De esta forma el estándar RS232 es muy flexible, pero presenta problemas cuando el formato de mensaje y protocolo difieren entre los dispositivos interconectados, aun cuando los niveles de señal y tiempo coincidan.

3.3 EL UART

El UART (Universal Asynchronous Receiver Transmitter) convierte datos paralelos en seriales y viceversa. Así, en la transmisión, el UART convierte datos paralelos

del bus del sistema del PC en datos seriales, y en la recepción, el UART convierte los datos seriales a datos paralelos que pueden ser leídos en el bus del sistema.

El UART soporta comunicaciones full-duplex, half-duplex y simplex. Además de las líneas de datos, soporta el handshake RS-232 y señales de control como RTS, CTS, DTR, DSR, RI, CD.

En los PCs IBM originales, el UART que controlaba el puerto serial fue el 8250, con una velocidad máxima de 57600 bits por segundo. Los UART en los nuevos PCs han emulado y mejorado el desarrollo original en cuanto al buffering, velocidad y otras características. Hoy en día, el UART en un PC es parte de un chip multifunción que contiene uno o más UARTs además del soporte para puertos paralelos y otros componentes del sistema.

La primera mejora del UART 8250 fue el 16450, el cual soporta velocidades de hasta 115200 bps. Añade el scratch pad, que no es más un byte de almacenaje en el UART sin una función asignada. Por otro lado, el 16550 añade buffers de transmisión y recepción. Los nuevos PCs tienen el equivalente de un 16550 o mejor.

Cada uno de los buffers del 16550, pueden almacenar 16 bytes. Los buffers son FIFO (first-in, first-out), lo que significa que los datos son leídos desde el buffer en el mismo orden como fueron recibidos. La característica de los buffers es que mejoran la eficiencia de la transferencia de datos.

En el lado receptor, el CPU no tienen que preocuparse de leer cada byte antes de que el próximo arribe. Si el CPU está ocupado, el buffer almacena los datos recibidos y el CPU puede leerlos a su conveniencia. El bus de datos del CPU es mucho mas rápido que la tasa de bits del puerto serial; de forma que puede leer todos los 16 bytes en una sola operación (en una fracción de tiempo que les toma a los datos arribar).

En el lado transmisor, el CPU puede escribir hasta 16 bytes al UART, y este se encarga del envío en secuencia.

Los nuevos UART se continúan construyendo sobre las características del 16550. Así, Texas Instruments TL16C750 tiene un FIFO de 64 bytes que trabaja a 5 V o 3 V. El chip soporta bit rates de hasta 1 Mbps, cuando trabaja con un cristal de 16 MHz. Este además tiene soporte para handshake automático RTS/CTS. En modo Auto-CTS, el UART transmite solo cuando el CTS está validado, liberando al software de chequear el estado de esta señal. En modo Auto-RTS, el UART automáticamente valida el RTS, cuando el FIFO de recepción tiene menos bytes que el número de bytes definidos en el umbral del buffer. Esto le indica al periférico que debe enviar más datos, lo que ayuda a evitar que el FIFO se quede vacío.

Otro ejemplo de los nuevos UART es el Exar ST16C50A. Este tiene un FIFO de 32 bytes, soporta bit rates de hasta 1.5 Mbps e incluye un codificador-decodificador IrDA para enlace infrarrojo. Este soporta handshake automático RTS/CTS, así como handshake automático de software. Sin embargo, las altas tasas de transferencias en los nuevos UART no están disponibles cuando el PC usa el reloj convencional de 1.8432 MHz, y las otras características avanzadas no están en uso a menos que el software conozca como habilitarlas y usarlas.

3.3.1 DENTRO DEL UART

La siguiente tabla muestra las funciones y direcciones de los registros internos de tres de los tipos de los UART originales:

Add-ress	Access	Name	Abb-rev	Bit Number							
				7	6	5	4	3	2	1	0
0	DLAB=0 Read only	Receive buffer	RVR	Received data							
	DLAB=0 Write only	Transmit holding register	THR	Transmit data							
	DLAB=0 Read/Write	Divisor Latch, low byte	DLL	Baud rate divisor low byte							
1	DLAB=0 Read/Write	Interrupt enable	IER	0	0	0	0	Modem status	Receiver Line status	Transmit Holding Register empty	Received Data available
	DLAB=1 Read/Write	Divisor Latch, high byte	DLM	Baud rate divisor high byte							
2	Read only	Interrupt Identify	IIR	FIFOs enabled**: 11 if FCR bit 7=1 00 if FCR bit 7=0		0	0	Interrupt ID: 011=receive line status 010=received dada avail 110=character timeout 001=TR hold reg. empty 000=modem status			Interrupt Pending
	Write only**	FIFO control**	FCR**	Receive FIFO trigger level** : 00=1 byte 01=4 bytes 10=8 bytes 11=14 bytes		received**	received**	DMA mode select**	Transmit FIFO reset**	Receive FIFO reset**	FFIFO enable**
3	Read/Write	Line control	LCR	Divisor latch access bit (DLAB)	Break set	Stick parity set	Even parity set	Parity enable	Stop bits: 0=1bit 1=2bits	Word length: 00=5 bits 01=6 bits 10=7 bits 11=8 bits	
4	Read/Write	Modem control	MCR	0	0	0	Loopback mode	OUT2 (IRQ enable on PCs)	OUT1	Request to send (RTS	Data termin al ready (DTR
5	Read only	Line Status	LSR	Error in receive FIFO	Transmit buffer empty	Transmit Holding Reg.	Break Interrupt	Framing Error	Parity Error	Overrun Error	Data Ready
6	Read only	Modem Status	MSR	Data carrier detect (CD)	Ring Indicator (RI)	Data set ready (DSR)	Creal to send (CTS)	Change in CD	RS232 falling edge at RI	Change in DSR	Chang e in CTS
7	Read/Write	Scratch*	SCR	Scratch Register, no designated function							

*16450 y 16550, **16550

Tabla 3.1

Registros del UART 8250, 164450 y 16550

Los UART 16550 y otros similares tienen doce registros de 8 bits. Es en estos registros donde se mantiene el byte a transmitir, el último byte recibido, el bit rate y otras configuraciones del puerto. Además información de control y estado del handshaking, uso del FIFO e interrupciones.

Los lenguajes de programación o las funciones API del sistema, a menudo incluyen funciones para configuración y uso del puerto; de esta manera se evita el acceso directo a los registros y se asegura un manejo más fácil y rápido del UART.

El UART tiene doce registros internos y dos buffers FIFO, pero solo ocho direcciones de puerto son necesarias. Esto se debe a que múltiples registros comparten una dirección, y el registro accesado depende del valor de un bit en otro registro o inclusive de si la operación es de lectura o escritura. Los FIFOs son internos al UART y no requieren direcciones de puerto.

La dirección de los UART son relativas a su dirección base. Así la dirección cero está en la dirección base del puerto con las otras direcciones siguiendo en secuencia. Por ejemplo en un COM1, la dirección cero está usualmente en 3F8h, y la dirección 7 está en 3FFh.

La dirección base tiene tres registros. Un registro de solo escritura, que mantiene el próximo byte a transmitir. Un registro de solo lectura, que mantiene el último byte recibido. Y un registro de lectura / escritura, que tiene el byte bajo del latch divisor.

Al ocurrir un reset, el bit DLAB (bit 7, dirección base + 3) es cero. Al leer la dirección base (buffer de recepción), se obtiene el más reciente dato recibido en el pin SIN del UART. Al escribir un byte en la dirección base (buffer de transmisión) se transmite un byte en formato serial en el pin SOUT.

El latch divisor tiene un valor de 16 bits que divide la frecuencia del cristal del UART para el bit rate deseado. Llevar DLAB a 1 permite usar el latch divisor (byte bajo: base address y byte alto: base address + 1) para ajustar el bit rate del puerto

La siguiente tabla muestra los valores del latch divisor para los diferentes bit rates asumiendo que el cristal estándar del UART es 1.8432 MHz:

Bit Rate (bps)	High Byte(Hex)	Low Byte(Hex)
300	01	80
1200	00	60
2400	00	30
9600	00	0C
19200	00	06
38400	00	03
115200	00	01

Tabla 3.2
Valores del Latch Divisor para ajustar el Bit Rate

En los antiguos 8250, el máximo bit rate permitido es 57600, con el latch divisor ajustado a 2. Otros UARTs pueden transmitir y recibir hasta 115200 baudios. Con cristales más rápidos, los nuevos UART pueden manejar más altas velocidades.

Solamente para ajustar o leer el bit rate, se habilita DLAB a 1. En adelante, para seguir operando el UART, el bit DLAB debe ser llevado a 0.

3.3.2 FUENTES DE INTERRUPCIÓN

Cuando DLAB es igual a 0, en el registro de Habilitación de Interrupciones (IER, dirección base + 1) se puede habilitar hasta cuatro fuentes de interrupción.

Cuando una interrupción ocurre, los bits 1, 2 y 3 del registro de Identificación de Interrupciones (IIR, dirección base + 2) discriminan la fuente de interrupción.

Muchas aplicaciones no usan todas las fuentes de interrupción. La fuente de interrupción más comúnmente usada es Received Data Available (bit 0 del registro IER). Cuando este bit es llevado a 1, el UART genera una interrupción cuando se ha recibido un nuevo dato. Leyendo el buffer de recepción se borra la interrupción, hasta que el próximo byte arribe.

Cuando el bit 1 del IER es ajustado a 1 lógico, una interrupción ocurre solamente cuando el buffer de transmisión está vacío. Esto le indica al CPU que debe escribir más datos al buffer.

Cuando el bit 2 del IER es ajustado a 1 lógico, una interrupción ocurre al detectar un error de transmisión. Esta fuente de interrupción es señalada por un cambio en los bits 1-4 del registro 5 (LSR). Los cuatro errores de transmisión que se pueden detectar son los siguientes:

- Overrun: Cuando un nuevo dato arriba al buffer antes de que el previo sea leído.
- Parity: Si paridad esta habilitada, el bit de paridad del byte del FIFO de recepción fue incorrecto.
- Framing: Cuando el carácter recibido no tiene el bit de parada (Stop bit). Este error ocurre también cuando los bit rates del transmisor y receptor son distintos.
- Break interrupt: Los datos recibidos se mantienen en 0 lógico por un tiempo mayor al tiempo de transmisión de un carácter.

Cuando el bit 3 del IER es ajustado a 1 lógico, una interrupción ocurre al detectar un cambio en una de las entradas de control del conector serial. Los bits 4-7 del registro de Estado del Modem (MSR, dirección base + 6), mantienen el estado de las entradas de Control, y los bits 0-3 indican cuales entradas cambiaron desde la última vez que se leyó el registro.

Los bits 4-7 se invierten dos veces entre el registro y el conector RS232. La interface RS232 invierte la señal una vez, y las entradas en el UART son los complementos de los valores correspondientes en el registro.

Así, cuando el bit4 es 0 lógico, el pin CTS del UART tiene un estado lógico alto, y el pin CTS en el conector RS232 es complemento.

3.3.3 REGISTROS DE CONTROL

El registro de Control de Línea (LCR, dirección base + 3), almacena información de configuración, tal como: los bits de parada, datos y paridad usados en cada transmisión. Por ejemplo, para la configuración más popular 8,n,1; los bits 5-0 del LCR serán 000111.

Por su parte el registro de Control del Modem (MCR, dirección base + 4) tiene algunas funciones importantes. El bit 3 (OUT2) es una salida de propósito general del UART. En los PCs este bit habilita la línea IRQ en el puerto. Ajustando OUT2 a 0 lógico, se deshabilita la salida que genera la petición de interrupción para el puerto. Aun si el UART detecta una interrupción, el controlador de interrupciones y el CPU nunca la verán.

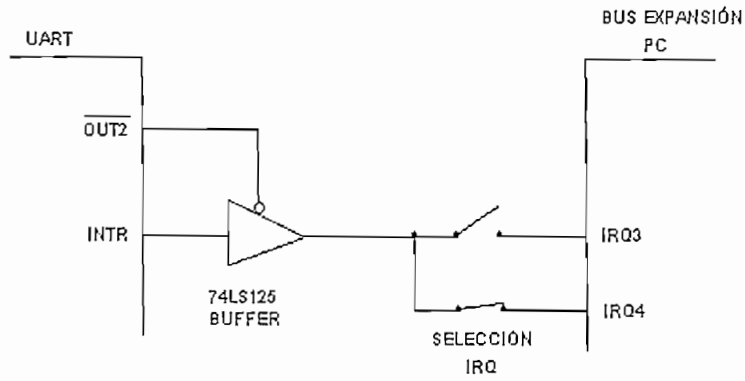


Figura 3.1
Hardware de Interrupción del UART

Los bits 0 y 1 del MCR controlan DTR y RTS respectivamente en el conector serial. El bit 2 es otra salida de propósito general. Esta no tiene una función definida en los PCs.

Ajustando el bit 4 de MCR se lleva al UART al modo de Loopback, lo que permite la lectura de los bytes transmitidos, en el buffer de recepción. El Loopback permite probar un puerto o incluso su presencia.

Algunos de los registros y bits referidos anteriormente no están disponibles en todas las versiones de UARTs. Así, el 16450 y el 16550 tienen un registro Scratch (dirección base + 7) que no tiene una función definida. De manera que en los programas este registro puede ser usado a conveniencia.
Otros bits controlan el FIFO del 16550. Ajustando el bit 0 del registro de Control del FIFO (dirección base + 2), habilita el FIFO y lleva a 1 lógico los bits 6 y 7 de este. Por tanto, llevando a 1 lógico el bit 0 y luego leyendo los bits 6 y 7, se obtiene una forma rápida de probar la presencia de un FIFO (pocos de los primeros FIFOs fallan este test). En UARTs que no tienen FIFO, los bits 6 y 7 siempre se leerán como 0 lógico.

3.4 RECURSOS DEL PUERTO

Cada puerto serial reserva un conjunto de direcciones, y la mayoría tienen asignados una línea de petición de interrupción (IRQ). Los puertos son designados como COM1, COM2, COM3, y así sucesivamente.

El determinar cuantos puertos seriales tiene un computador es una tarea algo más complicada que solamente verificar el número de conectores RS-232 disponibles en la parte posterior del PC. Un ejemplo de esto es un modem interno que muestra como su único conector un jack telefónico.

Existen algunas formas de encontrar información acerca de los puertos en el sistema, una es a través de los recursos que ofrece Windows mediante el panel

de control; allí se puede verificar la configuración del puerto, la dirección base, la tasa de bits, las IRQ asignadas, y otras características del puerto.

La configuración del puerto requiere que se asigne una IRQ a cada puerto. Convencionalmente se asignan cuatro direcciones base para los puertos con su correspondiente IRQ . Sin embargo los puertos no necesariamente deben cumplir esa condición; algunos usan cualquier dirección y cualquier IRQ que sea soportada por el hardware.

Cada puerto reserva 8 direcciones secuenciales que empiezan desde la dirección base.

La siguiente tabla muestra las direcciones base de puerto y la IRQ asignada:

Puerto	Dirección	IRQ
COM1	3F8H	4
COM2	2F8H	3
COM3	3E8H	4 o 11
COM4	2E8H	3 o 10

Tabla 3.3
Direcciones de Puerto Serial y líneas IRQ

Una de las características útiles de Windows, es que almacena las direcciones base y la líneas IRQ para cada puerto en el registro del sistema de Windows, de esta forma una aplicación puede accesar los puertos utilizando funciones de un lenguaje de programación o del Windows API (Application Programming Interface). Las funciones utilizadas por estos programas, hacen una llamada al puerto especificando su nombre (COM1, COM2, etc.) y Windows se encarga de encontrarlo y usarlo correctamente.

Bajo DOS se pueden encontrar las direcciones de los puertos COM en un área de memoria llamada área de datos BIOS. Al encenderse el computador, una

rutina en el BIOS busca la presencia de puertos seriales en las direcciones bases antes especificadas. En el área de datos del BIOS se pueden almacenar hasta cuatro direcciones base de 16 bits, empezando en la dirección 40:00.

3.4.1 CONFIGURACION

Muchos puertos seriales tiene jumpers, switchs, o utilidades de configuración que permiten seleccionar una dirección de puerto y una línea IRQ. La pantalla de setup, a la que se puede acceder al encender el sistema, permite configurar los puertos que residen en la tarjeta principal del sistema.

La cantidad de elecciones en la configuración varía; así algunos puertos permiten direcciones base y líneas IRQ distintas a las convencionales. Bajo Windows usar una dirección de puerto no convencional no presenta inconvenientes. Bajo DOS no es posible acceder a un puerto en una dirección no convencional, pues el BIOS no la detecta. Por otro lado el área de datos BIOS no almacenan las líneas IRQ asignadas a un puerto.

A pesar de que las recomendaciones de Microsoft PC 98 no sugieren el uso de puertos COM heredados, sin embargo los permiten con los siguientes requerimientos:

- El puerto debe tener un equivalente a un UART 16550A o mejor, y debe soportar tasas de bit de hasta 115200 bps.
- El puerto debe ser capaz de ser reconfigurado y deshabilitado completamente mediante software.
- El puerto debe soportar las direcciones y líneas IRQ convencionales.
- Cada puerto debe permitir una elección de al menos dos líneas IRQ.
- En caso de la existencia de dos puertos, la recomendación es elegir las líneas IRQ4 y IRQ11 para el un puerto, y las líneas IRQ3 y IRQ10 para el otro.
- Un puerto adaptador infrarrojo puede ser ubicado en lugar de un puerto serial.

3.4.2 INTERRUPCIONES

Uno de los principales problemas al usar múltiples puertos seriales, es que solo se dispone de dos líneas IRQ reservadas para el uso de los puertos. Convencionalmente COM1 y COM3 usan la IRQ4; mientras que COM2 y COM4 usan IRQ3. Por tanto se debe tener presente que solo se pueden utilizar dos puertos seriales a la vez.

Las interrupciones de hardware son muy utilizadas en algunas aplicaciones, debido a que permiten transferencias mucho más rápidas.

Una interrupción es una señal que indica al CPU que una tarea necesita servicio inmediato. El PC original de IBM soporta 8 líneas de interrupción. El modelo AT incrementó este número a 16, y este es el número que se ha mantenido a pesar de que el número de dispositivos que usan interrupciones se ha incrementado.

Cada línea IRQ corresponde a una señal que conecta la fuente de interrupción, al controlador de interrupciones del PC.

Un programa que usa interrupciones de hardware, debe ser capaz de proveer una rutina de servicio a interrupciones (ISR), esta se encargada de ejecutar las acciones correspondientes cuando una interrupción ocurre.

Cuando un dispositivo activa una línea de interrupción, el controlador de interrupciones del puerto detecta la condición e informa al CPU que una petición está pendiente. Entonces el CPU deja cualquier tarea que se este ejecutando y atiende la rutina programada en el servicio de interrupciones. Una vez atendida la interrupción el CPU reanuda las tareas que se estaban ejecutando cuando el servicio de interrupciones fue invocado.

En Visual Basic, el control MSCOMM se encarga de los detalles de instalación y habilitación de las rutinas de servicio de interrupción.

3.5 OPERACIÓN DEL RS232

El RS232 (EIA232E) define tres áreas de un estándar de comunicación: señales de voltaje, el uso de las líneas de señal y temporización de señales y bits. Estas son las definiciones básicas que se puede encontrar para un protocolo de comunicación, pero representan una base para protocolos más complejos e intercambio de mensajes.

3.5.1 NIVELES DE SEÑAL

Para representar un 1 ó un 0 binario, son requeridas dos señales de voltaje. De esta forma, un 0 binario (también conocido como espacio) se representa por cualquier voltaje entre +3 y +25 V. Por otro lado un 1 binario (marca) es un voltaje entre –3 y –25 V.

El intervalo de voltaje entre +3 a –3, es indefinido y no debe existir en ningún sistema que use el estándar RS232.

Cuando se efectúa una comunicación, el transmisor envía datos binarios con voltajes para 1 binario entre -5 y -25 V y para 0 binario con +5 y +25 V. El receptor por su parte decide que se ha recibido un 1 si detecta voltajes de entre –3 a –25, o que ha recibido un 0 binario si lee voltajes entre +3 a +25 V.

Esta brecha de niveles de voltaje, válidos para el transmisor y receptor, permite una caída de voltaje entre los extremos del enlace.

Por otro lado el uso de niveles de voltaje altos asegura una alta resistencia al ruido y menor impacto ante pérdidas de voltaje.

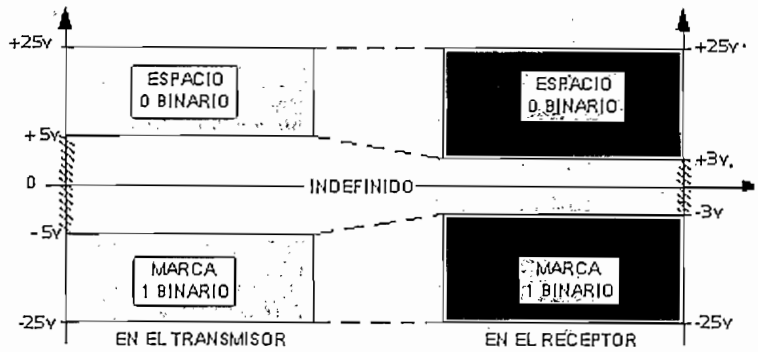


Figura 3.2
Niveles de Voltaje RS232

Para una interconexión muy corta, niveles RS232 de +/- 15 V o aún +/- 5 V son suficientes.

3.5.2 LÍNEAS DE SEÑAL

La implementación completa del estándar EIA232E (RS232) define en un extremo de la conexión un dispositivo llamado DTE (Data Terminal Equipment, usualmente un computador o terminal), que tiene un conector macho DB25, y utiliza 22 de los 25 pines disponibles para señales o referencia. El siguiente equipo en conexión es llamado DCE (Data Circuit-terminating Equipment, usualmente un modem), tiene un conector DB25 hembra, y utiliza los mismos 22 pines disponibles para señales y referencia.

El cable de enlace entre DTE y DCE es un cable paralelo directo.

Muchas de las 22 líneas de señal del estándar EIA232 corresponden a conexiones donde el dispositivo DCE es un modem.

Para cualquier dispositivo DCE que no sea un modem, o cuando dos dispositivos DTE son directamente conectados, el uso de estas líneas no es necesario, solamente unas pocas son utilizadas.

Seguidamente se presenta la asignación de pines y las señales del estándar EIA232E (RS232).

PIN	DESCRIPCIÓN
1	Protective ground
2	Transmitted data
3	Received data
4	Request to send
5	Clear to send
6	Data set ready
7	Signal ground (common return)
8	Received line signal detector (data carrier detect)
9	Reserved for data set testing
10	Reserved for data set testing
11	Unassigned
12	Secondary received line signal detector
13	Secondary clear to send
14	Secondary transmitted data
15	Transmission signal element timing (DCE source)
16	Secondary received data
17	Receiver signal element timing (DCE source)
18	Unassigned
19	Secondary request to send
20	Data terminal ready
21	Signal quality detector
22	Ring indicator
23	Data signal rate selector (DTE/DCE source)
24	Transmit signal element timing (DTE source)
25	Unassigned

Tabla 3.4

Líneas de Señal EIA232E

3.5.2.1 Conectores y Longitud de Cables

Seguidamente se muestran dos figuras con los conectores y señales para los dispositivos DTE y DCE, según la definición completa EIA232E.

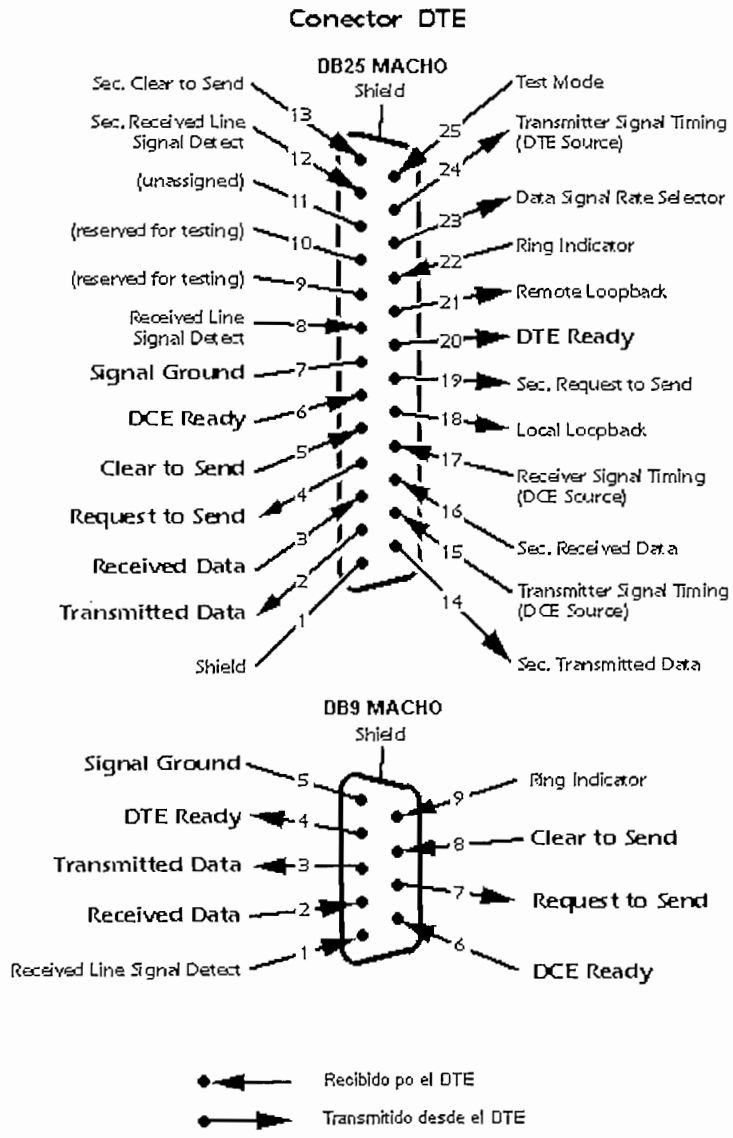


Figura 3.3

Señales del Conector DTE EIA232

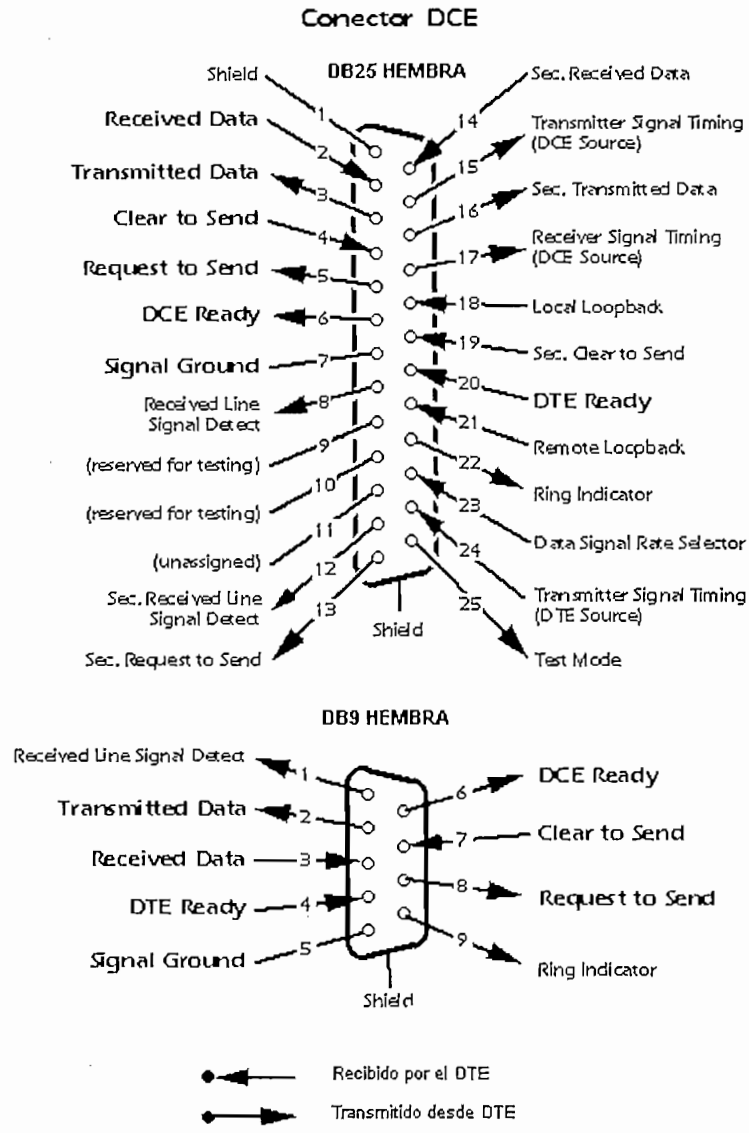


Figura 3.4
Señales del Conector DCE EIA232

Como se observa en los diagramas de pines para los conectores DB25, existe un canal secundario, el cual incluye un conjunto duplicado de señales de control de flujo. Este canal secundario es útil para el manejo de un modem remoto, permitiendo cambiar los baud rates mientras la comunicación está en transcurso, ante peticiones de retransmisión al detectar un error de paridad y otras funciones de control. Cuando este canal secundario es usado, típicamente opera a un bajo

baud rate, en comparación con el canal primario; lo que asegura confiabilidad. Además puede operar como canal simplex, halfduplex o fullduplex, dependiendo de las capacidades de modem.

Las señales de temporización de transmisión y recepción (pines 15,17 y 24) son usadas solamente en protocolos de transmisión sincrónica. Para comunicaciones asincrónicas con el protocolo de 8 bits, las señales de temporización externa no son necesarias.

Es importante notar que los nombres de señal que implican una dirección, tales como transmisión y recepción, son nombradas desde el punto de vista del dispositivo DTE. Con esta consideración, si el estándar EIA232 (RS232) fuera estrictamente seguido, estas señales deberían tener el mismo nombre para el mismo número de pin en el lado del dispositivo DCE. Desafortunadamente esto no se cumple.

Hoy en día, debido a que muchos de los 25 conductores especificados en el estándar no son usados, se implementó el uso de un conector de 9 pines (DB9) para el extremo del DTE. Sin embargo, algunos dispositivos DCE continúan utilizando el conector DB25, aunque esto está cambiando.

La longitud del cable no está especificada en el estándar; sin embargo las primeras versiones recomendaron una longitud de cable de hasta 50 pies a tasas de hasta 20000 bps. Las últimas versiones se refieren más bien al cumplimiento de una capacitancia de hasta 2500 pF en el receptor.

3.5.2.2 Definición de Señales

Las señales en el estándar EIA232 pueden ser subdivididas en seis categorías:

1. Señal de referencia y blindaje
2. Canal de comunicación primario. Este es usado para intercambio de datos e incluye señales de control de flujo.

3. Canal de comunicaciones secundario. Cuando se implementa, se usa para controlar el modem remoto, peticiones de retransmisión ante la ocurrencia de errores y manejo de la configuración del canal primario.
4. Estado del modem y señales de control. Estas señales indican el estado del modem y proveen verificadores intermedios ante el establecimiento de un canal telefónico de voz.
5. Señales de temporización de transmisión y recepción. Si un protocolo sincrónico es usado, estas señales proveen información de temporización para el transmisor y el receptor, los cuales pueden operar a diferentes baud rates.
6. Señales de prueba de canal. Antes del intercambio de datos, se puede probar la integridad del canal y se puede ajustar automáticamente el baud rate al máximo que el canal pueda soportar.

3.5.2.2.1 Señal de Tierra y Blindaje

Pines 7,1 y el chasis. A pesar que externamente los conductores proveen un camino separado para cada uno de estos pines, internamente están interconectados.

Pin 7 GROUND. Todas las señales son referidas a una tierra común, el pin 7 del conector. El conductor de tierra puede o no ser conectado al pin de protección de tierra del DCE.

3.5.2.2.2 Canal de Comunicación Primario

Pin 2 Transmitted data (TxD). Esta señal es activa cuando los datos son transmitidos desde el DTE al DCE. Cuando no hay datos a transmitir, la señal es mantenida en la condición de marca. (1 lógico, voltaje negativo).
Sobre el DCE, el pin 2 es comúnmente etiquetado como "Received Data". Sin embargo el EIA232 mantiene su nombre como "Transmitted data".

Pin 3 Received data (RxD). Esta señal es activa cuando el DTE recibe datos desde el DCE. Cuando no hay datos transmitidos, la señal es mantenida en la condición de marca. (1 lógico, voltaje negativo).
Sobre el DCE, el pin 3 es comúnmente etiquetado como "Transmitted data". Sin embargo el EIA232 mantiene su nombre como Received data.

Pin 4 Request to send (RTS). Esta señal es validada (0 lógico, voltaje positivo) para preparar al DCE para aceptar los datos transmitidos desde el DTE. Tal señal puede ser usada para habilitar los circuitos receptores, o configurar la dirección del canal en aplicaciones half-duplex. Cuando el DCE está listo, envía una señal de reconocimiento validando Clear to send.
Sobre el DCE, el pin 4 es comúnmente etiquetado como "Clear to send". Sin embargo el EIA232 mantiene su nombre como Request to send.

Pin 5 Clear to send (CTS). Esta señal es validada (0 lógico, voltaje positivo) por el DCE, para informar al DTE que la transmisión puede empezar. RTS y CTS son comúnmente usadas como señales de handshake, para controlar el flujo de datos dentro del DCE.
Sobre el DCE, el pin 5 es comúnmente etiquetado como "Request to send". Sin embargo el EIA232 mantiene su nombre como Clear to send.

3.5.2.2.3 Canal de Comunicación Secundario

Las señales en el canal secundario son equivalentes a sus correspondientes señales en el canal primario. Sin embargo el baud rate, es típicamente mas bajo en el canal secundario para incrementar confiabilidad.

Pin 14 Secondary Transmitted data (STxD)
Pin 16 Secondary Received data (SRxD)
Pin 19 Secondary Request to send (SRTS)
Pin 13 Secondary Clear to send (SCTS)

3.5.2.2.4 Señales de Control y Estado del Modem

Pin 6 DCE Ready (DSR). Cuando se origina desde un modem, esta señal es validada (0 lógico, voltaje positivo).

Si la señal DCE Ready se origina desde un dispositivo distinto a un modem, esta puede se validada para indicar que el dispositivo ha sido encendido y está listo para funcionar, o puede no ser usada.

De no usar esta señal debe permanecer validada (0 lógico, voltaje positivo). Alternativamente, el DTE puede ser programado para ignorar esta señal.

Pin 20 DTE Ready (DTR). Esta señal es validada (0 lógico, voltaje positivo) por el DTE, cuando este requiere abrir un canal de comunicación. Si el dispositivo DCE es un modem, la validación de esta señal prepara al modem para que se conecte al circuito telefónico, y una vez conectado mantenga la conexión.
Cuando el DTE Ready es desactiva (1 lógico, voltaje negativo), el modem termina la conexión telefónica.
Si el dispositivo DCE no es un modem, este puede requerir que la señal DTE sea validada antes de que el DCE pueda ser usado. De todas formas si no se va ha usar esta señal, puede ser ignorada.

Pin 8 Received Line Signal Detector (CD). También llamada Carrier Detect. Esta señal es relevante cuando el DCE es un modem. El modem valida (0 lógico, voltaje positivo) la señal, solo cuando la línea telefónica esta descolgada, una conexión ha sido establecida y un tono de respuesta esta siendo recibido desde el modem remoto. Esta señal es desactivada cuando no hay un tono de respuesta, o cuando el tono de respuesta es de inadecuada calidad para cumplir los requerimientos del modem.

Pin 12 Secondary Received Line Signal Detector (SCD). Esta señal es equivalente a Received Line Signal Detector (pin 8), pero referente al canal secundario.

Pin 22 Ring Indicator (RI). Esta señal es relevante cuando el DCE es un modem. Se valida (0 lógico, voltaje positivo) cuando una señal de timbrado es recibida desde la línea telefónica. El tiempo de validación de esta señal será aproximadamente igual a la duración de la señal de timbrado, y será desactivada entre timbrados o cuando no hay timbrado presente.

Pin 23 Data Signal Rate Selector. Esta señal puede ser originada en un DTE o en un DCE, pero no en ambos a la vez,. Es usada para seleccionar uno o dos baud rates predefinidos. Ante la validación de esta señal (0 lógico, voltaje positivo) se selecciona el baud rate mas alto.

3.5.2.2.5 Señales de Temporización de Transmisión y Recepción

Pin 15 Transmitter Signal Element Timing (TC). También llamada Transmitter Clock. Esta señal es relevante solamente cuando el DCE es un modem y está operando con un protocolo sincrónico. El modem genera esta señal de reloj para controlar exactamente la tasa a la cual los datos son enviados desde el pin 2 Transmitted data del DTE, al comunicarse con el DCE. La transición de voltaje que ocurre al cambiar de lógica 1 a lógica 0 (voltaje negativo a positivo), origina la transmisión del próximo elemento de datos sobre la línea de transmisión.
El modem genera esta señal continuamente, excepto cuando está realizando funciones de diagnóstico interno.

Pin 17 Receiver Signal Element Timing (RC). También llamada Receiver Clock. Esta señal es similar a la TC descrita anteriormente. Excepto que provee información de temporización para el receptor DTE.

Pin 24 Transmitter Signal Element Timing (ETC). También llamada External Transmitter Clock. En este caso las señales de temporización son generadas por el DTE, y son convenientes cuando se usa un modem. Esta señal se utiliza cuando TC y RC no son usadas.

3.5.2.2.6 Señales de Prueba del Canal

Pin 18 Local Loopback (LL). Esta señal es generada por el DTE y origina que el modem entre en un estado de prueba. Cuando se valida (0 lógico, voltaje positivo), el modem redirecciona su señal de salida modulada (la cual es normalmente alimentada a la línea telefónica) devuelta hacia su circuitería de recepción. Esto hace posible que los datos generados por el DTE, sean enviados y recibidos de vuelta, para probar la circuitería del modem. El modem valida su señal Test Mode en el pin 25 para indicar está en la condición de Local Loopback.

Pin 21 Remote Loopback (RL). Esta señal es generada por el DTE y es usada para poner el modem remoto en un estado de Test. Cuando Remote Loopback es validado (0 lógico, voltaje positivo), el modem remoto redirecciona los datos recibidos a su entrada de datos transmitidos, lo que significa que remodula los datos recibidos y los devuelve a su fuente. Cuando el DTE inicia un Test, los datos transmitidos pasan a través del modem local, la línea telefónica y el modem remoto y regresan de vuelta para probar el canal y comprobar su integridad. El modem remoto indica al modem local que el Test Remote Loopback va a ser ejecutado, validando Test Mode en el pin 25.

Pin 25 test Mode (TM). Esta señal es útil solamente cuando el DCE es un modem. La validación de esta señal (0 lógico, voltaje positivo), indica que el modem se encuentra en una condición de Local Loopback o Remote Loopback.

3.5.3 TEMPORIZACIÓN

El siguiente asunto a tratar es sobre la presentación de los bits en un medio asincrónico.

Cuando no hay una señal presente (estado de inactividad), la línea de transmisión de datos RS232 se encuentra en el estado de marca (1 lógico, voltaje negativo). Cuando existen datos a enviar, la línea va a su estado de espacio (0 lógico, voltaje positivo) durante un período de bit llamado bit de inicio (Start Bit). Esta

transición, indicando que una nueva serie de bits de datos van a ser enviados, es censada por la circuitería del receptor RS232, y es una advertencia que lo prepara para recibir los bits entrantes.

Dependiendo de la configuración del diseño del sistema, el bit de inicio es seguido de 5, 6, 7 o hasta 8 bits de datos (7 y 8 bits son los comunes), en donde el bit menos significativo (LSB) de los bit de datos es enviado primeramente.

Luego del último bit de datos, opcionalmente se puede agregar un bit de paridad, con el propósito de detección de errores. La secuencia completa de los bits de datos y paridad termina con uno o mas bits de parada (Stop Bit), donde la línea de datos va al estado de marca (1 lógico, voltaje negativo), por 1, 1.5 o 2 períodos de bits. Esto indica que se ha completado la secuencia de los bits de datos.

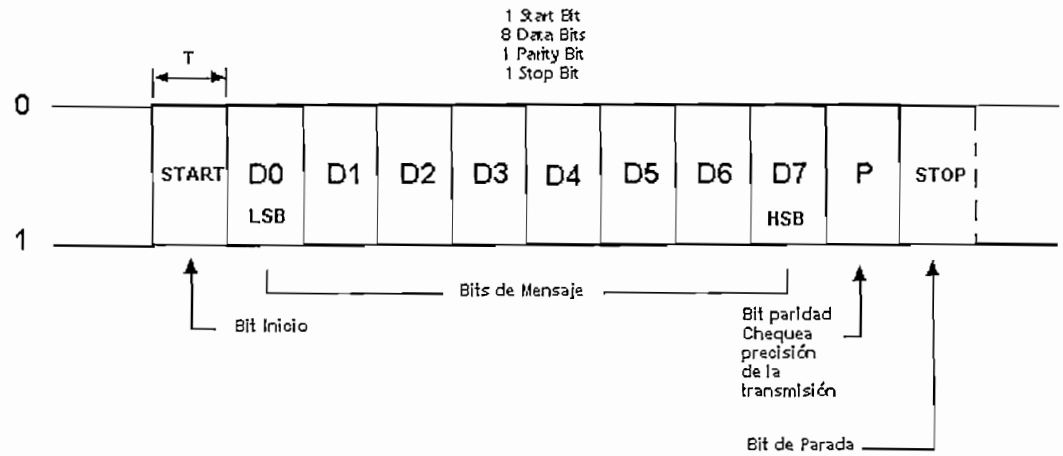


Figura 3.5
Secuencia de Bits de Datos RS232

Aunque el estándar no lo especifica, la representación de los datos se la hace comúnmente en código ASCII.

El estándar RS232 permite algunos valores de baud rate, hasta un máximo de 20000 baud. Las tasas más comúnmente usadas son: 300, 1200, 2400, 4800,

9600 y 19200 bit/s. La tasa de 300 bit/s es usada para teletypes mecánicos, o canales de bajo ruido y bajo ancho de banda.

Las tasas de 1200, 2400, 4800 y 9600 baud, son utilizados por la mayoría de sistemas electrónicos. A pesar de que 19200 baud es el valor estandar más rápido dentro de los 20000 baud especificados por el estandar RS232, algunos sistemas se comunican hasta 38400 baud. Esta última tasa funciona en sistemas interconectados a distancias menores a 50 pies, bajo situaciones cuidadosamente controladas, sin embargo, no es una aplicación frecuente de la interface RS232.

Un período de bit es el inverso de baud rate, así, para un baud rate de 1200 el período de bit es de 0.83 mseg.

3.5.4 CONTROL DE FLUJO

El control de flujo se implementa en un enlace para evitar la pérdida de datos debido principalmente al desbordamiento del buffer de recepción.

El control de flujo se puede realizar por Hardware o por Software. Ambos necesitan un programa que realice las tareas de handshaking. El control de flujo por software se lleva a cabo usando las líneas de comunicación estándar RxD y TxD, mientras que el control de flujo por hardware, utiliza líneas de control adicionales para realizar el handshaking.

3.5.4.1 Control de Flujo por Software

Para implementar el control de flujo por Software, también llamado Xon/Xoff, se utiliza dos códigos específicos de los caracteres ASCII. Uno de estos códigos es denominado Xon, y está normalmente designado por el carácter ASCII 17. Este carácter es utilizado por el DTE ó por el DCE para indicar al otro dispositivo que está listo para enviar o recibir nuevos caracteres. El segundo código llamado Xoff, representado por el código ASCII 19, es utilizado para indicar que el buffer de recepción está lleno y que no se deben enviar nuevos caracteres.

Xoff y Xon deben su nombre al hecho de que ellos pueden parar y reanudar una transmisión.

En una comunicación que implementa control de flujo por Software, el DTE ó DCE no transmite ningún dato a menos que haya recibido un código Xon. Cuando el DTE o DCE está enviando datos, circuitería especial y software continuamente monitorean la línea de recepción de datos para verificar si algún carácter Xoff ha sido recibido. Si efectivamente ha sido recibido un Xoff, el software detiene el envío de nuevos caracteres. Una vez que el buffer es vaciado, se envía el carácter Xon para indicar que se puede reanudar la transmisión. Un Xoff es también transmitido para indicar que el envío de datos se ha concluido.

Este tipo de control de flujo tiene la ventaja de que no requiere líneas adicionales además de TxD y RxD. Sin embargo tiene la desventaja de que hace uso del canal de datos entre los dos dispositivos para transmitir Xon/Xoff, lo cual origina una reducción del ancho de banda, debido a que cada uno de estos caracteres requiere diez bits (8,n,1).

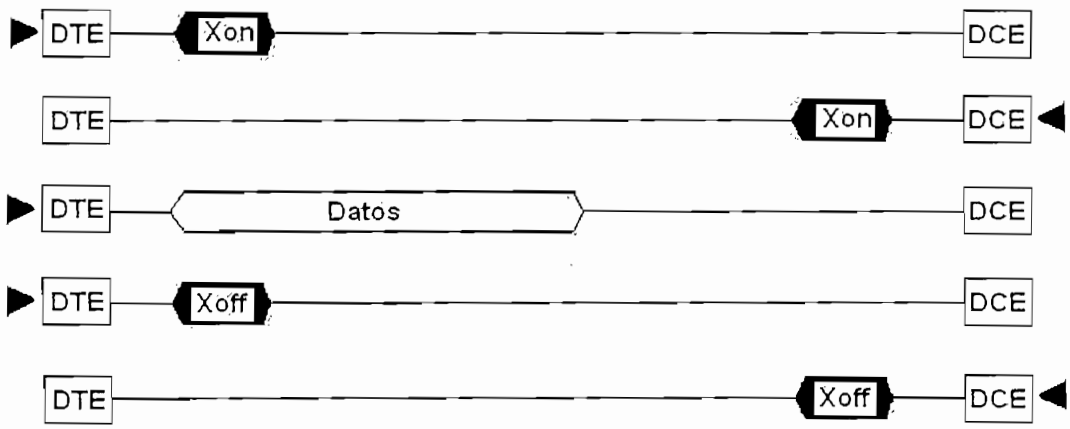


Figura 3.6
Control de Flujo por Software

Las comunicaciones seriales que usan control de flujo por Software son aceptables únicamente cuando las velocidades de comunicación no son altas, y la probabilidad de que el buffer se desborde o los caracteres de datos sean alterados por agentes externos, como ruido, sea mínima.

3.5.4.2 Control de Flujo por Hardware

El control de flujo por Hardware es también conocido como control de flujo RTS/CTS. Este usa dos conductores del cable serial para llevar la información de handshaking, en lugar de caracteres extra transmitidos por las líneas de datos (TxD y RxD).

El control de flujo RTS/CTS es usado principalmente para el handshaking entre un computador y un modem.

El handshake de hardware se lleva a cabo de la siguiente forma:

Cuando el computador desea transmitir datos, valida (0 lógico, voltaje positivo) su línea RTS para indicar al dispositivo este propósito. El dispositivo chequea si tiene espacio en su buffer para recibir la información, de ser así valida la línea CTS para indicar al computador que inicie la transferencia. El DTE entonces comienza el envío de los bits del carácter al baud rate especificado. Luego de cada carácter, el DCE usa la línea CTS para indicarle al DTE que está procesando el carácter recibido y que le es imposible aceptar otro. El DTE indica que tiene otro carácter para enviar, validando (0 lógico, voltaje positivo) la línea RTS. Una vez que el DCE está en capacidad de aceptar más bits, la línea CTS es validada y solo entonces el DTE reanuda el envío.

Las líneas comúnmente usadas para realizar el handshake de hardware son CTS y RTS, sin embargo en algunos enlaces se utilizan también las líneas DSR y DTR para ampliar el handshaking.

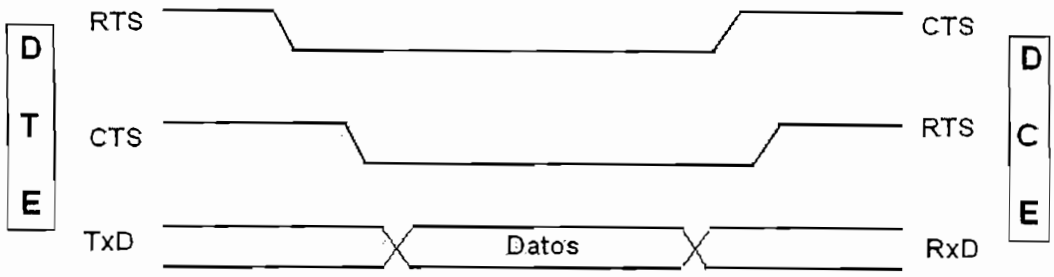


Figura 3.7
Control de Flujo por Hardware

3.5.4.2.1 Null Modem

Cuando se enlazan dos dispositivos DTE, la conexión es llamada Null Modem, y el concepto de handshake de hardware puede ser implementado de manera similar; con la diferencia de que se utiliza un cable cruzado en lugar de uno directo, como ocurre al conectar dispositivos DTE con DCE.

Existen algunas variaciones en cuanto a las conexiones Null Modem; así, para una conexión simple, tres líneas de un cable que incluyen TxD, RxD y GND son suficientes. Esta es una conexión *Null Modem Simple* sin handshaking.

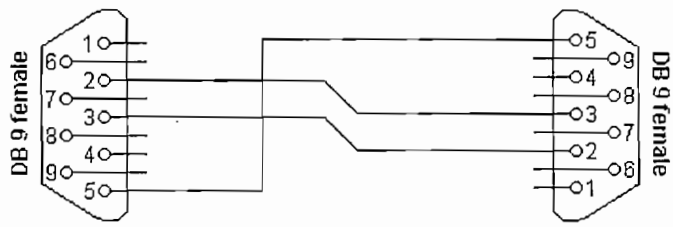


Figura 3.8
Simple Null Modem sin Handshaking

Un simple cable Null Modem sin Handshaking puede presentar inconvenientes dependiendo del software usado; por lo tanto, muchas veces es necesario implementar alguna clase de handshaking. Cables *Null Modem con Handshaking* pueden ser definidos en numerosas formas: con Handshaking de Loopback en

cada PC, o con Handshaking Completo entre los dos sistemas. Sin embargo, para algunos programas, solo es necesario un Handshaking Parcial. A continuación se muestran las figuras correspondientes de los tipos de cable con Handshaking:

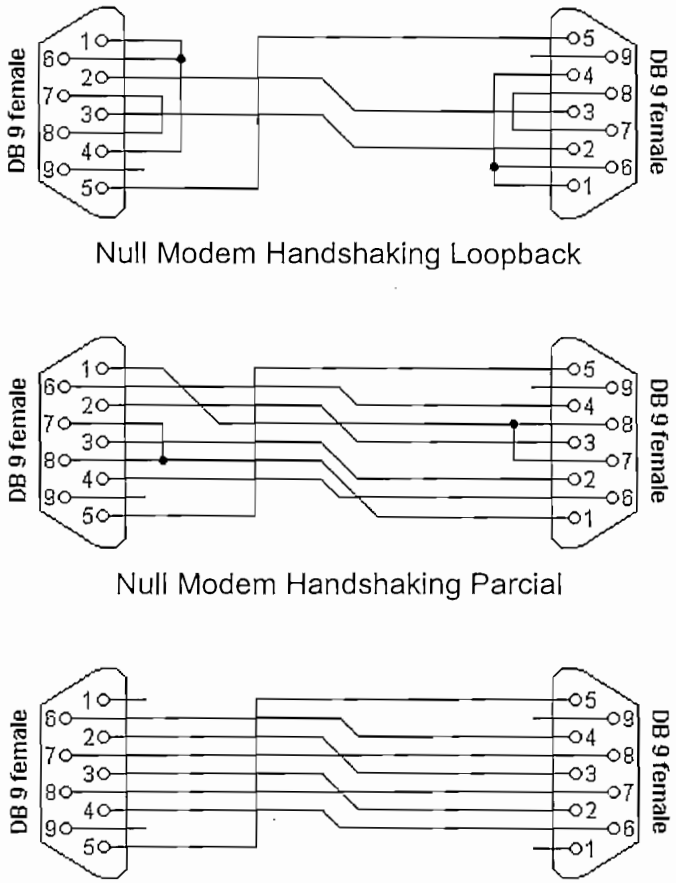


Figura 3.9

Null Modem Con Handshaking

3.5.5 BUFFERING

El control de flujo por Hardware o RTS/CTS es ineficiente debido a que se consume tiempo valioso validando RTS y chequeando CTS. Aun a altos baudrates, el envío efectivo de caracteres es bajo debido al intervalo entre

caracteres. Las transferencias continuas y el buffering son usadas para mejorar la eficiencia y el throughput (se refiere al número de caracteres de datos útiles enviados, recibidos y procesados por segundo).

Una operación más eficiente es posible mediante el uso de un buffer, el cual es un área de memoria de almacenamiento interna presente tanto en el DCE como en el DTE. EL buffer, también conocido como memoria FIFO, permite al DCE aceptar caracteres desde el DTE aun si no ha procesado aquellos previamente recibidos.

Cuando el DTE envía una cadena de caracteres a alta velocidad y llena el buffer de recepción del DCE, el DTE puede realizar otras tareas sin preocuparse del manejo de la interfaz de comunicación, pues el buffer de transmisión se encarga de este cometido, enviando los caracteres almacenados sin intervalos.

A medida que el buffer de recepción se vacía, el DCE valida la línea CTS para indicar al DTE esta situación. Cuando el buffer de recepción esta casi lleno, deshabilita la línea CTS, para que el DTE no envíe nuevos caracteres. El indicador CTS se deshabilita antes de que el buffer de recepción esté totalmente lleno, lo que asegura que cualquier carácter en proceso de envío sea completado. Una vez que el DCE procesa los caracteres recibidos y vacía el buffer, indica nuevamente al DTE que está listo para recibir nuevos caracteres.

CAPÍTULO 4

DISEÑO DEL SISTEMA

Para diseñar este sistema se partió de que el objeto de este proyecto es construir un módulo que, a través de un software de soporte, realice pruebas en los puertos serial y paralelo de cualquier computador. Para este fin se dedujo que el módulo debía ser portátil y que el programa de soporte corra en cualquier sistema operativo Windows. La Figura 4.1 muestra un esquema simplificado de lo que aquí se desea construir.

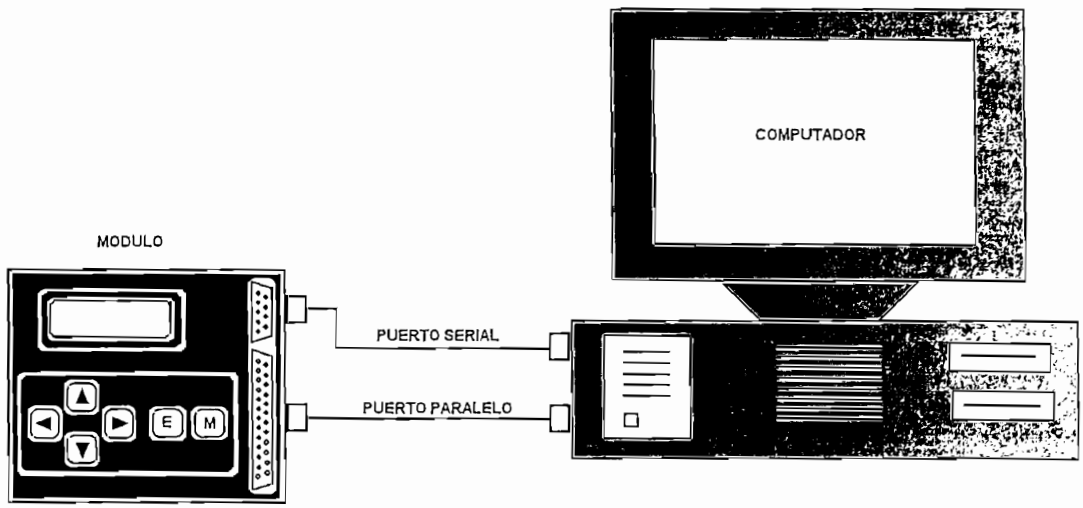


Figura 4.1
Modulo - Computador

Se desea llamar la atención al hecho que el proyecto requirió de una investigación completa en cuanto a la compatibilidad tanto del hardware que se conecta a los puertos, como del software que se tenía que instalar en los diferentes sistemas operativos Windows, a más de un conocimiento exhaustivo de los estándares para la comunicación tanto serial como paralela.

La necesidad de conocer al detalle sobre aspectos como: tiempo de reconocimiento de señales de handshake, tiempo de ejecución de instrucciones para realizar la transmisión y recepción de datos tanto en el programa del

microcontrolador como en el software de soporte, características eléctricas y respuesta de frecuencia de los circuitos integrados que conforman el hardware del módulo, elección de una adecuada librería dinámica para el manejo del puerto paralelo, requerimientos de utilitarios adicionales a los contenidos en Visual Basic para asegurar la correcta instalación y carga del programa de soporte, depuración y personalización de los programas del módulo y del computador para lograr una adecuada compatibilidad entre ambos, entre las más relevantes, dan fe de lo expuesto al inicio de este capítulo .

A continuación se presentan el software de soporte, el programa del microcontrolador PIC y el hardware construido.

4.1 DISEÑO DEL SOFTWARE DE SOPORTE

El programa de soporte para el sistema de pruebas de comunicación, está desarrollado en Visual Basic 6.0.

El ambiente grafico de desarrollo de aplicaciones para el sistema operativo de Microsoft Windows proporcionado por Visual Basic, ayuda a crear aplicaciones con una interfaz de usuario sencilla y amigable. Esta es la razón por la que se eligió Visual Basic 6.0 (VB6.0) para la aplicación desarrollada.

La primera consideración para el desarrollo de la aplicación, se concentró en las funciones disponibles en VB6.0 para el manejo de los puertos.

Se determinó que, aunque VB6.0 es una herramienta potente para crear aplicaciones con interface de usuario de fácil uso, tiene un limitante importante en cuanto al manejo del hardware de la computadora. Esto se ve reflejado al no tener implementadas funciones de acceso directo a los puertos y que, como se verá más adelante, son indispensables para el manejo en particular del puerto paralelo.

Esta dificultad se resuelve mediante el uso de una Librería de Enlace Dinámico (DLL) desarrollada en lenguaje C, que si maneja el hardware de la computadora directamente para el acceso a los puertos. Existen múltiples librerías dinámicas para el acceso a los puertos publicadas como software de libre disponibilidad. Es así que, para nuestros propósitos, se utilizo una librería desarrollada para sistemas de 32 bits que se encuentra disponible en la pagina web www.logix4u.cjb.net

Un DLL no es otra cosa que un archivo ejecutable independiente, que contiene funciones y recursos que pueden ser llamados por los programas y por otras DLLs para realizar ciertos trabajos.

Un DLL no puede ser ejecutado de forma independiente; entra en acción hasta que un programa u otra DLL llama a una de las funciones de la librería. El término enlace dinámico se refiere al hecho de que el código que contiene la DLL se incorpora al programa ejecutable que la llama solo hasta el momento en que es requerida, en tiempo de ejecución.

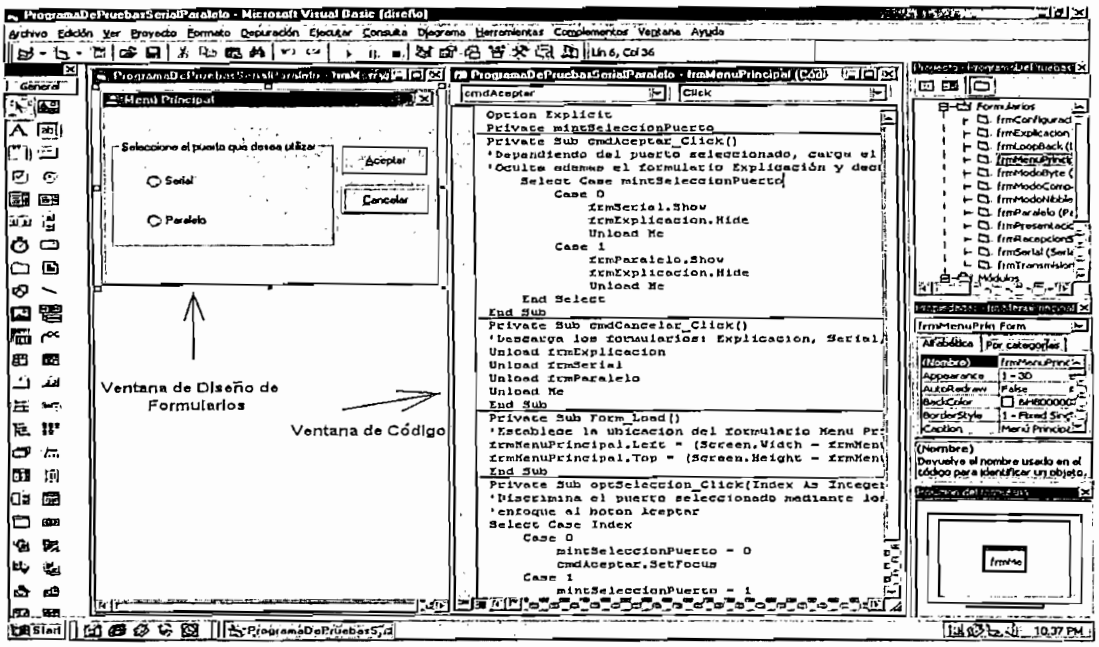
La DLL utilizada (inpout32.dll), contiene funciones Inp y Out. Estas funciones son agregadas al programa fuente de la aplicación y ejecutan de manera transparente las tareas de escritura y lectura de los puertos.

La utilidad de esta librería se incorporó al programa de aplicación, solamente para el desarrollo de las rutinas de manejo del puerto paralelo. Para el acceso al puerto serial no hubo necesidad de utilizar sus capacidades, pues en VB existe un control propio que maneja esta interface llamado MSComm y que se encarga del manejo de dicho puerto.

4.1.1 ENTORNO DE DESARROLLO DE VISUAL BASIC

El entorno de desarrollo de aplicaciones de VB, define básicamente el uso de dos interfaces para la escritura de código de programa: Módulos de Formulario y Módulos Estandar.

Los Módulos de Formulario ofrecen un medio de comunicación entre el usuario y la aplicación. Un Módulo de Formulario consta de dos ventanas: Ventana de Diseño de Formularios, donde se agregan los controles con los que el usuario interactúa, y una Ventana de Código, que es donde se escriben las líneas de programa que responderán a los eventos de los controles (Ver Figura 4.2).

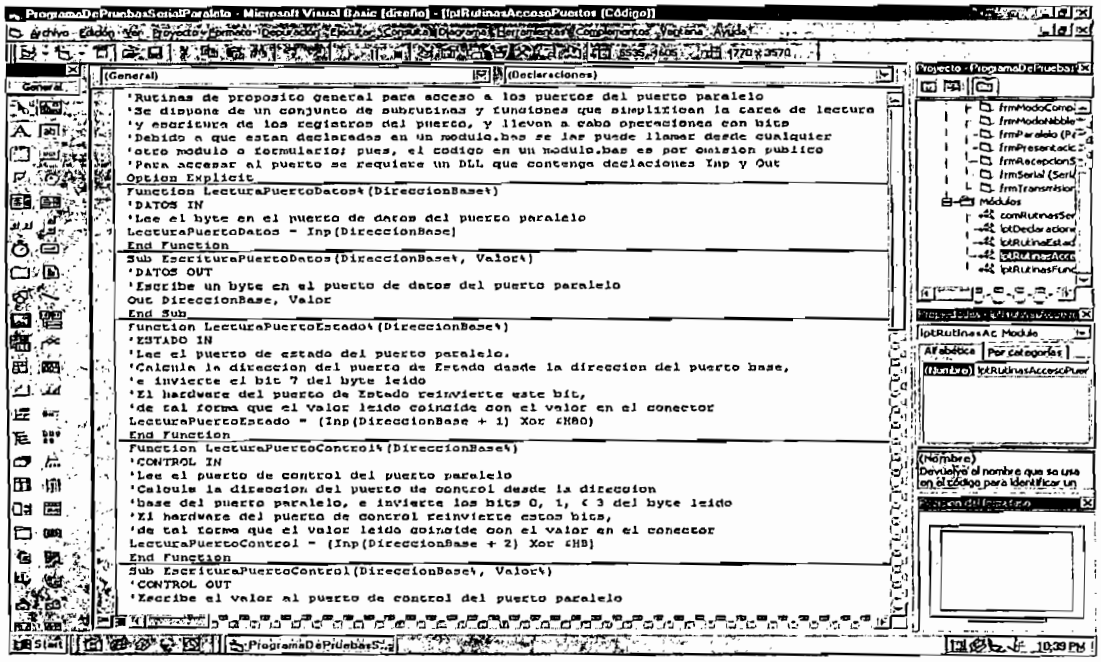


Ventana de Diseño de Formulario y Ventana de Código
Figura 4.2

El código escrito en un formulario es denominado "Local", lo que significa que solo sirve para el formulario en particular y no puede ser utilizado ni llamado por otro formulario. Cualquier rutina, función o subrutina escrita dentro de un formulario es también local.

Los Módulos Estándar por su parte, son ventanas que contienen solamente código, el cual por definición es llamado "Publico" (Ver Figura 4.3). A diferencia del código Local, el código Publico tiene un alcance global, lo que significa que puede ser llamado y utilizado por cualquier modulo o formulario. Esto facilita el

desarrollo de aplicaciones que comparten un mismo código para llevar a cabo algunas tareas.



Módulos

Figura 4.3

Tomando en cuenta estas herramientas de desarrollo de aplicaciones que provee Visual Basic, el programa se estructura a base de formularios y módulos estándar.

Para establecer una programación funcional, se definieron rutinas de propósito general, que incluyen funciones y subrutinas que son compartidas por toda la aplicación y que se declaran en Módulos Estándar.

Se diseña un total de 12 formularios denominados como sigue:

Presentación
Explicación
Menú Principal
Puerto Serial

Prueba de LoopBack
Transmisión Serial
Recepción Serial
Configuración del Puerto Serial
Puerto Paralelo
Transmisión en Modo Compatible
Recepción en Modo Nibble
Recepción en Modo Byte

y las rutinas implementadas son:

Rutina de Acceso a los Puertos
Rutina de Funciones Especiales
Rutina de Estado de los Puertos
Declaraciones Inp Out
Rutinas Seriales

donde las cuatro primeras tienen que ver con el manejo del Puerto Paralelo, la ultima con el manejo del Puerto Serial.

El programa desarrollado es hábil para manejar los puertos de PCs que utilizan sistemas operativos de 32 bits. Windows XP, 2000 y NT son ejemplos de tales sistemas.

En cuanto al hardware, los computadores que usan procesadores Pentium en adelante utilizan buses de 32 bits y son aptos para correr la aplicación realizada. Adicionalmente se recomienda espacio en el disco de al menos 30 MB y una memoria RAM de 256MB.

4.1.2 PROGRAMA DEL PUERTO PARALELO

El programa para el puerto paralelo esta concebido para realizar las siguientes tareas:

- Identificar el o los puertos paralelos instalados y reconocidos por Windows, y realizar una prueba de existencia y tipo de tales puertos. La identificación de los puertos muestra la dirección base del puerto y el tipo de puerto.
- Leer y/o escribir en los Registros de Datos, Estado y Control mediante el ingreso, recepción de un byte de información, o mediante la manipulación de los bits individuales de cada puerto.
- Como recurso adicional se provee herramientas de transferencias de datos de hasta 10 caracteres utilizando los protocolos de comunicación propuestos por IEEE 1284 para una transferencia directa en Modo Compatible y para el caso de transferencia inversa en Modo Nibble y Modo Byte.

4.1.2.1 Herramientas de Programación

El programa de soporte para el acceso del puerto paralelo se desarrolla utilizando la técnica de escritura y lectura directa a los registros del puerto. De esta forma las operaciones implementadas ofrecen un control completo, incluso a nivel de bits, para los registros de Datos, Estado y Control.

Las rutinas de propósito general para este caso, contienen funciones comunes para la escritura y lectura de los registros y bits individuales del puerto, prueba de existencia y tipos de puertos y chequeo del estado de los mismos.

Además, debido a que se utiliza el inpout32.dll, se incluye la declaración de las funciones Inp y Out disponibles en el DLL.

Se implementan 3 modos básicos para transferencia de datos entre el computador y el modulo: Modo Compatible, Modo Nibble y Modo Byte.

Para una transferencia directa, se utiliza el Modo Compatible, y para transferencias inversas, se utilizan el Modo Nibble y el Modo Byte.

Se definió un limite máximo de 10 caracteres para la trasmisión y para la recepción de datos. La codificación ASCII es utilizada como estándar de Facto para los datos a transferir.

Aunque existen dos modos de transferencia adicionales (Modo EPP y Modo ECP), el programa incluye solamente los modos de transferencia más comúnmente utilizados y accesibles por la mayoría de PCs.

Debido a las variedades en el diseño de puertos paralelos y a que IBM no documentó completamente los requerimientos de tiempo para los protocolos de transferencia, se implementa un programa que obvia la temporización, y que se ciñe solamente a cumplir el handshake en cuanto a las señales de control se refiere, donde cada señal de control permanece validada hasta que el extremo opuesto la reconoce. Esta es la consideración fundamental tomada para el desarrollo de los modos de comunicación del puerto paralelo.

Se definió un tiempo máximo para la transferencia de datos de 30 segundos, luego de lo cual se desborda un temporizador e indica transferencia fallida.

El programa para el puerto paralelo esta compuesto de 4 formularios y 4 módulos estándar.

FORMULARIOS:

- Puerto Paralelo
- Transmisión en Modo Compatible
- Recepción en Modo Nibble
- Recepción en Modo Byte

MODULOS ESTANDAR

- Rutina de Acceso a los Puertos
- Rutina de Funciones Especiales

Rutina de Estado de los Puertos
Declaraciones Inp Out

FORMULARIOS:

Absolutamente todos los formularios en alguna parte del código de programa, utilizan llamadas a funciones y subrutinas definidas en las tres Rutinas declaradas en los Módulos Estándar.

Puerto Paralelo

- Inicialmente identifica el o los puertos paralelos instalados y reconocidos por Windows, y se realiza una prueba de existencia y tipo de tales puertos. La identificación de los puertos se muestra mediante controles de texto que incluyen la dirección base del puerto y el tipo de puerto. Estas tareas se inician al llamar una subrutina local denominada EncontrarPuertosParalelos.
- El formulario aloja también controles y código para las Pruebas Bit a Bit del Puerto Paralelo. Estas pruebas llevan a cabo fundamentalmente tareas de lectura / escritura de los registros de datos, estado y control del puerto. Las funciones y subrutinas de propósito general utilizadas son: EscrituraPuertoDatos, LecturaPuertoEstado y EscrituraPuertoControl.
- Se incluyen también: un menú que llama a los formularios que se encargan de la transferencia de datos y un menú para llamar al formulario de Puerto Serial.

Transmisión en Modo Compatible

- Formulario para realizar transferencias directas en Modo Compatible.
- Para asegurar la transferencia, primeramente verifica que un puerto exista y posteriormente habilita los controles para hacer posible la transmisión.

- Inicializa el puerto de control al valor requerido para empezar una transmisión (9H).
- Llama a una subrutina local denominada EscribirBytePuertoMC, cuya tarea es escribir hasta 10 caracteres ingresados en un cuadro de texto, a las líneas de datos del puerto. La escritura de los caracteres se realiza en secuencia y bajo el protocolo de Modo Compatible.
- Se prevee un tiempo máximo de transferencia de 30 segundos, luego de lo cual se considerara la transmisión fallida.

Recepción en Modo Nibble

- Formulario para realizar transferencias reversas en Modo Nibble.
- Para asegurar la transferencia, primeramente verifica que un puerto exista y posteriormente habilita los controles para hacer posible la recepción
- Inicializa el puerto de control al valor requerido para empezar la recepción (2H).
- Verifica que el modulo este listo y configurado en el modo adecuado.
- Llama a la subrutina local RecibirMN que se encarga de recibir hasta 10 caracteres ingresados por las líneas de estado del puerto.
- La recepción de los caracteres se realiza en secuencia y bajo el protocolo de Modo Nibble.
- Se prevee un tiempo máximo de transferencia de 30 segundos, luego de lo cual se considerara la recepción fallida.

Recepción en Modo Byte

- Formulario para realizar transferencias reversas en Modo Byte.
- Para asegurar la transferencia, primeramente verifica que un puerto exista y posteriormente habilita los controles para hacer posible la recepción.
- Inicializa el puerto de control al valor requerido para empezar la recepción (23H).
- Verifica que el modulo este listo y configurado en el modo adecuado.

- Llama a la subrutina local RecibirMB que se encarga de recibir hasta 10 caracteres ingresados por las líneas de datos del puerto.
- La recepción de los caracteres se realiza en secuencia y bajo el protocolo de Modo Byte.
- Se prevee un tiempo máximo de transferencia de 30 segundos, luego de lo cual se considerara la recepción fallida.

MODULOS ESTÁNDAR:

Las funciones y subrutinas definidas en los Módulos Estándar son utilizadas en su momento por todos los formularios del programa.

Rutina de Acceso a los Puertos

- Se desarrolla para la escritura / lectura de los registros o bits del puerto.
- Utiliza las funciones Inp y Out definidas en inpout32.dll .
- Calcula automáticamente la dirección de cada uno de los registros del puerto (registros de Datos, Control y Estado), a partir de las tres direcciones base estándar y carga en ellos los valores a leer y/o escribir.
- Incluye tareas de lectura / escritura que reinvierten las señales del bit 7 del registro de estado y de los bits 0, 1 y 3 del registro de Control que son invertidas en el conector.
- Para la lectura del Registro de estado y para la escritura del Registro de Control se utiliza enmascaramiento, mediante una operación OR exclusiva (XOR) entre el valor leído / escrito en el registro correspondiente y el valor hexadecimal 80H para el registro de estado, o, BH para el registro de control.
- El enmascaramiento mediante la operación XOR complementa el o los bits que son invertidos en el conector y previene que cambien los bits del registro que no lo son.
- Las funciones definidas aquí son llamadas y utilizadas por todos los formularios del puerto paralelo, pues para llevar a cabo cualquier tarea

sobre el mismo, siempre se deberá leer y/o escribir a sus registros y estas funciones se encargan de la operación.

Las operaciones de lectura / escritura de registros se desarrollo mediante las siguientes funciones y subrutinas:

Lectura del Registro de Estado:
LecturaPuertoEstado = (Inp(DirecciónBase + 1) Xor &H80)

Escritura del registro de control:
Out DirecciónBase + 2, Valor Xor &HB

Para la lectura y/o escritura del Registro de Datos, no es necesario ningún enmascaramiento debido a que este registro no tiene ningún bit invertido en el conector.

Lectura del Registro de Datos:
LecturaPuertoDatos = Inp(DirecciónBase)

Escritura del Registro de Datos:
Out DirecciónBase, Valor

Rutina de Funciones Especiales

- Contienen código de programa que se encarga de verificar la existencia y el tipo de los puertos paralelos del computador, en cada una de las tres direcciones base establecidas.
- Esta rutina se ejecuta cada vez que se carga el programa de puerto paralelo, de forma que se asegura que cualquier cambio ocurrido sea detectado por el programa.

Al verificar la existencia y tipo de puertos, surge un inconveniente en el orden en que se debe llevar a cabo esta tarea. Es común pensar que el primer paso

es el buscar un SPP, y de allí en adelante probar si es un PS2, un EPP o un ECP. Sin embargo esto no es correcto, pues si el puerto del computador es un ECP, y sucede que esta configurado internamente como un SPP, el puerto fallara las pruebas bidireccionales (PS2). Esta es la razón por la que se comienza probando en primer lugar si es un ECP, luego un EPP, seguidamente un SPP y, de no resultar ninguno de estos, finalmente un PS2.

Prueba de un ECP

Un ECP tiene registros adicionales, uno de estos registros, el ECR en la dirección base + 402H, se utiliza para la prueba de la existencia de un ECP.

El procedimiento implementado para probar la presencia de un ECP, se detalla a continuación.

El primer paso es leer el registro ECR y verificar que el bit 0 (FIFO empty) sea 1, y el bit 1 (FIFO full) sea 0. Estos bits deben ser distintos del bit 0 y bit 1 del registro de Control, pues el test consiste en cambiar el estado de dichos bits y luego comprobar que los bits correspondientes en el registro ECR no hayan cambiado. Adicionalmente para completar esta prueba, se escribe el valor 34H al registro ECR. Como los bits 0 y 1 son solo de lectura, al leer de vuelta ECR se debe conseguir el valor 35H, pues el bit 0 no cambia. Si esto se cumple entonces el puerto es con certeza un ECP.

Si el puerto existe, se puede leer y escribir el modo interno del ECP en los bit 5, 6 y 7 del ECR.

Prueba de un EPP

Si el puerto fallo el test ECP, el programa busca entonces un puerto EPP.

Tal como sucede con el ECP el EPP también tiene registros adicionales. En el EPP estos registros están en la dirección base + 3 hasta la dirección base + 6.

Los registros adicionales y el bit Timeout del EPP son los que ofrecen una alternativa para probar la presencia de un EPP.

La prueba consiste en escribir un valor en uno de los registros del puerto EPP (dirección base + 3 ó dirección base + 4) y luego leerlo de vuelta. Si no hay un periférico EPP compatible conectado al puerto, el sistema no es capaz de completar el handshake EPP y el bit de timeout se activa al terminar el tiempo de transferencia inhabilitando la lectura del registro EPP. Si al borrar este bit es posible leer de vuelta el valor escrito en el registro, el puerto es un EPP. Este procedimiento se realiza con dos valores distintos para asegurar su efectividad.

Prueba de un SPP

Si el puerto falla las dos pruebas precedentes, el siguiente paso es probar la presencia de un SPP. Esto se lleva a cabo escribiendo dos valores al Puerto de Datos y leyéndolos de vuelta. Si los valores leídos coinciden, el puerto existe y es un SPP. De otra forma el puerto no existe o no esta trabajando apropiadamente.

Prueba de un PS2

Este test se realiza una vez que el puerto ha pasado la prueba SPP. El programa primeramente configura el puerto de Datos como entrada, escribiendo 1 en el bit 5 del Registro de Control (dirección base + 2). Si el puerto es bidireccional, esta acción origina que las salidas del Puerto de Datos se pongan en alta impedancia, permitiendo que trabaje como entrada. Por tanto, si los valores leídos no coinciden con los escritos, el puerto es bidireccional (PS2).

Esta prueba debe ser realizada, solo cuando se aseguro que el puerto existe, pues si se trata de leer los valores del puerto de Datos de un puerto que no

existe, se tendrá el mismo resultado anterior (los valores leídos no coinciden con los escritos).

Las funciones definidas en esta Rutina, corresponden a métodos estándar de detección de puertos paralelos y se desarrollaron de acuerdo a las recomendaciones hechas en el libro "Parallel Port Complete" de Jan Axelson, que a su vez son referidas de los documentos de Microsoft para el Puerto Paralelo.

Rutina de Estado de los Puertos

- Esta rutina se desarrolla para leer los registros de Datos, Estado y Control del puerto seleccionado y mostrar los valores del byte leído y de cada bit dentro del byte. Cuando entra en ejecución, lee los valores presentes en el conector DB25 y actualiza los controles del formulario Paralelo.
- Es llamada recurrentemente en todos los formularios del programa paralelo

Declaraciones Inp Out

- Contiene las declaraciones que se deben añadir a cualquier programa que use las subrutinas (Out) y funciones (Inp) disponibles en el inpout32.dll

4.1.3 PROGRAMA DEL PUERTO SERIAL

El programa para el puerto serial se desarrolla para llevar a cabo las siguientes tareas:

- Identificar el o los puertos seriales instalados y reconocidos por Windows, y realizar una prueba de existencia de tales puertos.
- Realizar pruebas bit a bit de las 5 señales más utilizadas para el handshake RS232, presentes en un conector DB9.
- Como recurso adicional, puede realizar transmisión y recepción de datos de hasta 10 caracteres según el estándar EIA232 y pruebas de LoopBack.

Las transferencias de datos se realizan con cable Null Modem, considerando a ambos dispositivos como DTE.

4.1.3.1 Herramientas de Programación

Los lenguajes de programación o las funciones API del sistema, a menudo incluyen funciones para configuración y uso del puerto; de esta manera se evita el acceso directo a los registros y se asegura un manejo más fácil y rápido del UART.

Debido a que Windows almacena las direcciones y las líneas IRQ para cada puerto, la aplicación no debe preocuparse de este fin. El programa accede al puerto usando las funciones propias del lenguaje de programación, que en este caso las proporciona el control MSComm.

Los puertos son llamados COM1, COM2, COM3, etc, por tanto, para detectar la presencia de un puerto usando MSComm basta con referirlo por su nombre y Windows se encarga de reconocerlo y encontrarlo.

El control MSComm utiliza la función PortOpen para abrir un puerto. Esta función es útil para la detección del mismo. De esta manera, se elige un nombre de puerto y luego se intenta abrirlo, si un puerto se abre, entonces existe; caso contrario no. Es así como se procedió para determinar la presencia de puertos seriales.

El programa busca la existencia de hasta 16 puertos, iniciando en COM1, y finalizando en COM16.

Esta búsqueda se ejecuta cada vez que se carga el programa de puerto serial, de forma que se asegura que cualquier cambio ocurrido sea detectado por el programa.

Se define una rutina de propósito general (en modulo estándar) que contiene funciones de uso común para la inicialización del puerto serial, y funciones para la escritura y lectura de los bits del puerto.

Para las transferencias de datos entre el computador y el modulo, se implementan transferencias halfduplex, que se ciñen al protocolo EIA232.

Antes de realizar una transmisión o recepción, es necesario configurar previamente parámetros como baudrate, control de flujo, formato, bits de datos, inicio, parada y paridad.

Se definió por defecto para toda transferencia el protocolo 8,n,1 (8 bits de datos, sin paridad, 1 bit de parada) y se implementaron 2 tipos de Control de Flujo y 2 Formatos para los datos.

Las opciones de baudrate disponibles son : 300, 1200, 2400, 9600, 19200 y 28800.

En lo que concierne al Formato, se tiene dos alternativas: formato texto y formato binario. En formato texto los caracteres son enviados y transmitidos como códigos ASCII. En formato binario se transfieren datos en forma binaria, con una capacidad de manejo de un byte.

El control de flujo se realizo por hardware y por software. Para el control de flujo por hardware se considero un handshake ampliado; es decir utilizando las líneas DTR, DSR, RTS y CTS. En el caso de control de flujo por software se usan los caracteres de Xon y Xoff.

El control de flujo se desarrolla simulando un buffer de transmisión y recepción de un solo byte y se realiza handshake con chequeo de control de flujo por cada byte transferido. Aunque el MSComm permite esta opción, no permite usar las líneas de handshake DTR y DSR como se considero para la programación de la interface, por tanto para este propósito no se utilizo las herramientas de MSComm

y se desarrollo código de programa independiente. Además esta consideración se extiende al PIC, que maneja solamente un buffer de transmisión de tamaño de un byte y de dos bytes de buffer de recepción; por lo que la necesidad de un control personalizado fue requerido.

Como herramienta adicional se desarrolla una prueba de LoopBack del puerto, valiéndose del uso de un conector del mismo nombre. El conector de LoopBack interconecta las líneas de transmisión y recepción del puerto, y las señales de handshake de forma que cualquier dato enviado es enrutado a la línea de recepción inmediatamente. De esta forma se puede realizar una prueba de transmisión y recepción, con control de flujo sin necesidad de conectarse al modulo.

Se definió un tiempo máximo para la transferencia de datos de 30 segundos, luego de lo cual se desborda un temporizador e indica transferencia fallida.

El programa para el puerto serial esta compuesto de 5 formularios y 1 módulo estándar.

FORMULARIOS:

- Puerto Serial
- Prueba de LoopBack
- Transmisión Serial
- Recepción Serial
- Configuración del Puerto Serial

MODULO ESTANDAR

- Rutinas Seriales

FORMULARIOS:

Puerto Serial

- Inicialmente identifica el o los puertos seriales instalados y reconocidos por Windows, y se realiza una prueba de existencia de tales puertos. La identificación de los puertos se muestra mediante controles de texto que incluyen la denominación hecha por Windows para el puerto (COMx). Estas tareas se inician al llamar una subrutina local denominada EncontrarPuertosSeriales. La técnica empleada es determinar la existencia de puertos, abriendo consecutivamente hasta 16 puertos. Si el puerto se abre, entonces existe.
- El formulario aloja también controles y código para las Pruebas Bit a Bit del Puerto Serial. Estas pruebas llevan a cabo fundamentalmente tareas de lectura / escritura de los bits individuales del puerto, valiéndose del uso de las funciones propias del control MSComm disponible en Visual Basic para el manejo del puerto serial.
- Se incluyen también: un menú que llama al formulario para las Pruebas de LoopBack, un menú que llama a los formularios que se encargan de la transferencia de datos y un menú para llamar al formulario de Puerto Paralelo.

Prueba de LoopBack

- Formulario para realizar pruebas de transmisión y recepción del puerto.
- Como valor necesario para esta prueba se configura: Formato Texto y Control de Flujo por Hardware.
- Por otro lado por defecto se establece: 9600 bps.
- Llama a una subrutina local denominada EnviarBytePuerto, cuya tarea es enviar 10 caracteres ingresados en un cuadro de texto, y mostrar los caracteres que están disponibles en la línea de recepción cada vez que se envían datos.

- Se prevee un tiempo máximo de transferencia de 1 segundo, luego de lo cual se considerara la transmisión / recepción fallida.

Transmisión Serial

- Formulario para transmisión de datos (Computador-Modulo) .
- Por defecto, la configuración para la transferencia se establece como sigue protocolo 8,n,1, 9600bps, control de flujo por hardware y formato texto.
- El código de programa en este formulario llama a una subrutina local denominada EnviarBytePuerto, que se encarga de enviar hasta 10 caracteres ingresados en un cuadro de texto, según el protocolo, control de flujo y formato definidos por defecto o en el formulario Configuración del Puerto Serial.
- Se prevee un tiempo máximo de transferencia de 30 segundos, luego de lo cual se considerara la transmisión fallida.

Recepción Serial

- Formulario para recepción de datos (Modulo-Computador).
- Por defecto, la configuración para la transferencia se establece como sigue protocolo 8,n,1, 9600bps, control de flujo por hardware y formato texto.
- El código de programa en este formulario llama a una subrutina local denominada RecibirS, que se encarga de recibir hasta 10 caracteres, según el protocolo, control de flujo y formato definidos por defecto o en el formulario Configuración del Puerto Serial.
- Se prevee un tiempo máximo de transferencia de 30 segundos, luego de lo cual se considerara la recepción fallida.

Configuración del Puerto Serial

- Formulario para la configuración del puerto serial del computador.
- Por defecto, la configuración se establece como sigue protocolo 8,n,1, 9600bps, control de flujo por hardware y formato texto.

- Desde este formulario se pueden establecer ajustes del puerto como: puerto a usar, baudrate, control de flujo y formato.
- Las opciones de puertos, dependen de los que se tengan instalados.
- Se disponen de 6 opciones de velocidad: 300, 9600, 1200, 2400, 9600, 19200 y 28800 bps.
- En cuanto al control de flujo, se dispone de dos opciones: Control de Flujo por Hardware y Control de Flujo por software. También se incluye la posibilidad de no usar control de flujo alguno (None).
- El formato se puede elegir para ser de Texto o Binario.

RUTINAS:

Rutinas Seriales

- Las Rutinas Seriales del modulo estándar se desarrollan para inicializar el puerto y verificar y establecer el estado de los bits individuales.
- En lo que respecta a la inicialización del puerto, el código se encarga de definir el puerto serial a usar, el baudrate, bits de datos, paridad, parada y los valores por defecto como el tamaño del buffer de transmisión y recepción, así como el estado inicial de las líneas de señal del puerto (DTR, DSR, RTS, CTS y CD).
- Las funciones de inicialización en esta rutina son ejecutadas al iniciar el programa de puerto serial y cada vez que se ha realizado un cambio en la configuración del puerto.
- Por su parte las funciones de manejo de los bits individuales son llamadas a necesidad del programa para realizar lectura y/o escritura de los bits del puerto y para actualizar los controles.

4.2 PROGRAMACION DEL MICROCONTROLADOR PIC

El PIC 16F877A responde a un grupo de solo 35 instrucciones de 14 bits cada una, que se graban en su memoria de programa tipo Flash de 8K; la cual puede ser programada y borrada eléctricamente, facilitando el desarrollo de programas y la experimentación. La memoria Flash de programa esta dividida en cuatro bloques, de 2K cada uno.

El numero de instrucciones a las cuales responde un microcontrolador PIC es pequeño, comparado con las de un microprocesador de la familia MCS-51/52 de INTEL por ejemplo. Esto implica una mayor dificultad en cuanto al desarrollo del código de un programa; pero realiza instrucciones de 14 bits de tamaño en lugar de 8 bits. Esta capacidad le permite al PIC realizar un cometido en menor número de instrucciones que en otro dispositivo similar, pues una instrucción de 14 bits provee mayor información que una de 8 bits.

El microcontrolador posee un bloque de memoria RAM de Datos dividido en 4 particiones, las cuales contienen Registros de Propósito general y Registros de Funciones Especiales. Los Registros de Propósito General son usados para guardar los datos temporales de la tarea que se esta ejecutando, y los Registros de Funciones Especiales sirven para configurar las diferentes capacidades que posee el microcontrolador.

También existe un bloque de memoria EEPROM, que puede ser usada para almacenar datos que no se desee perder al desconectar la alimentación del microcontrolador, pero para nuestros fines no es necesaria.

El LCD se maneja a través de los 6 pines del Puerto A del microcontrolador. Se utilizan 4 bits para transmitir los bytes de instrucciones y datos, y 2 bits para manejar las señales de habilitación y control del LCD, usando su modo de operación de 4 bits.

El programa esta estructurado de la siguiente manera:

- Definición de Variables
- Rutina de Atención a la Interrupción
- Subrutinas de Tareas Específicas
  - Subrutinas de Retardo
  - Subrutina de Conversión Hexadecimal a Decimal
  - Subrutina de envío de Datos y señales de Control al LCD
  - Subrutina de envío de Carácter y Posición al LCD
  - Subrutina de envío de Mensajes
  - Subrutina de Transmisión Serial de 10 Caracteres
  - Subrutinas de Configuración I/O del Puerto D
- Programa Principal
  - Rutinas de Configuración Inicial
  - Rutinas de Menús
    - Rutinas del Menú Principal
      - Rutina de opción Comunicación Serial
      - Rutina de opción Comunicación Paralela
    - Rutinas del Menú Serial
      - Rutina de opción Transmisión Serial
      - Rutina de opción Recepción Serial
      - Rutina de opción Prueba Pin a Pin Serial
      - Rutinas de opción Configuración Serial
        - Rutinas de opción Baudrate
        - Rutinas de opción Control de Flujo
    - Rutinas del Menu Paralelo
      - Rutinas de opción Transmisión Paralela
      - Rutina de opción Recepción Paralela
      - Rutina de opción Prueba Pin a Pin Paralela
  - Rutinas de Ejecución
    - Rutina de Ingreso de Datos
    - Rutinas de Transmisión de Datos
      - Rutinas de Transmisión Serial
        - Rutina de Transmisión con Control de Flujo por Hardware

Rutina de Transmisión con Control de Flujo por Software

Rutina de Transmisión sin Control de Flujo

Rutinas de Transmisión Paralela

Rutina de Transmisión Modo Nibble

Rutina de Transmisión Modo Byte

Rutinas de Recepción de Datos

Rutinas de Recepción Serial

Rutina de Recepción con Control de Flujo por Hardware

Rutina de Recepción con Control de Flujo por Software

Rutina de Recepción sin Control de Flujo

Rutina de Recepción Paralela Modo Compatible

Rutinas de Pruebas Pin a Pin

Rutina de Prueba Pin a Pin Serial

Rutina de Prueba Pin a Pin Paralela

Subrutinas para Cargar Mensajes

Definición de Variables

- Se definen los nombres y direcciones de los Registros de Propósito General utilizados como variables.
- Define los nombres de los pines a utilizar como señales de habilitación y control.

Rutina de Atención a la Interrupción

Esta rutina se ejecuta cada vez que cambian de estado los bits RB5:RB7 del puerto B del microcontrolador.

- La rutina lee los bits RB5:RB7, donde esta conectado físicamente el teclado, y guarda este valor en el registro TECLA.

Subrutinas de Tareas Específicas

Las Subrutinas de Tareas Especificas pueden ser llamadas en cualquier parte del programa principal. Realizan la tarea para la cual fueron creadas y regresan a la dirección en la cual fueron llamadas mas uno. Se describen a continuación.

Subrutinas de Retardo

- Generan retardos necesarios para que el LCD reconozca las señales de control RS y E, utilizadas para discriminar entre bytes de Control y Datos.

- Generan retardos para eliminar los rebotes producidos al presionar y al soltar una tecla (se utilizan para la Subrutina de Atención a la Interrupción).
- Generan retardos necesarios para el reconocimiento de las señales de handshake y datos en la transmisión y recepción.
- Generan retardos luego del envío de mensajes al LCD para su visualización.

Subrutina de Conversión Hexadecimal a Decimal

Esta subrutina se utiliza para transformar el valor leído en el puerto D, que maneja el puerto paralelo del PC, en decimal.

- Convierte un valor Hexadecimal almacenado en el registro de trabajo en decimal, y lo almacena en los registros Unidad, Decena y Centena.

Subrutina de envío de Datos y señales de Control al LCD

- Genera la secuencia necesaria de RS y E para que el LCD discrimine entre un byte de Datos y un byte de Control.
- Escribe en el puerto A del microcontrolador el byte de Datos o Control a enviar.

Subrutina de envío de Carácter y Posición al LCD

- Envía el carácter almacenado en el registro CAR a la posición especificada en el registro POS al LCD.

Subrutina de envío de Mensajes

- Envía 40 caracteres almacenados en los registros CAR01:CAR40 al LCD para mostrar un mensaje completo.

Subrutina de Transmisión Serial de 10 Caracteres

- Transmite vía serial los caracteres almacenados en los registros DAT01:DAT10 sin control de flujo.

Subrutinas de Configuración I/O del Puerto D

- Configuran el puerto D del microcontrolador como Entrada para leer los registros de Control y Datos del PC.
- Configuran el puerto D del microcontrolador como Salida para escribir en los registros de Estado y Datos del PC.

Programa Principal

Rutinas de Configuración Inicial

- Configura los puertos del PIC y los Registros de Función Especial de acuerdo al estado inicial que se requiere.
- Da valores iniciales a los registros designados para almacenar los caracteres a transmitir y recibir, y demás Registros de Propósito General.
- Carga valores por defecto de baudrate, control de flujo y modo de transmisión paralela.

Rutinas de Menús

Todas las Rutinas de Menús realizan la siguiente secuencia:

- Presentan el mensaje correspondiente a la opción del menú.
- Leen el registro TECLA y de acuerdo a este valor avanzan a la opción siguiente, regresan a la opción anterior, saltan a una de las Rutinas de Ejecución o salen a las opciones del menú anterior.

Las rutinas de las opciones Baudrate, Control de Flujo y Transmisión Paralela además guardan en registros designados para este fin el baudrate, control de flujo y modo de transmisión elegidos respectivamente.

Las primeras opciones de los menús Serial y Paralelo guardan "S" o "P" en un registro para discriminar el puerto por el cual se va a transmitir o recibir.

Rutinas de Ejecución

Estas rutinas ejecutan la opción seleccionada en las Rutinas de Menús.

Rutina de Ingreso de Datos

- Presenta el mensaje para el ingreso de datos.
- Permite el ingreso de hasta 10 caracteres ASCII por teclado. De acuerdo al valor que contiene el registro TECLA, cada uno de los registros designados para almacenar los caracteres a enviar cambia de 20H a 5AH, pasando solo por los valores correspondientes a los valores ASCII de espacio, 0 al 9 y A a la Z.
- Guarda los caracteres ingresados en las localidades DAT01:DAT10 para envío de datos.
- A través de las Subrutinas de envío de Datos y señales de Control al LCD y envío de Carácter y Posición al LCD, permite la visualización de los caracteres ingresados por teclado en el LCD.

- Lee el registro INDTX, que guarda el puerto por el cual se va a transmitir, y salta a la Rutina de Transmisión que corresponda.

Rutinas de Transmisión de Datos

Estas rutinas transmiten hasta 10 caracteres vía puerto serial o paralelo.

Rutinas de Transmisión Serial

Rutina de Transmisión con Control de Flujo por Hardware

- Lee el registro FLUJO que contiene el control de flujo seleccionado para la transmisión, si es "H" continua, de lo contrario salta a Rutina de Transmisión con Control de Flujo por Software.
- Presenta mensaje de transmisión con control de flujo por Hardware, junto con el baudrate elegido y los datos a transmitir.
- Carga el baudrate en el registro de configuración y habilita el puerto serial.
- Transmite los caracteres ingresados vía serial usando control de flujo por Hardware.
- Presenta mensaje de transmisión exitosa junto con los caracteres enviados si la transmisión se completó.
- Lee registro TECLA y dependiendo de este valor salta a la Rutina de Ingreso de Datos para realizar una nueva transmisión o sale a la opción Transmisión Serial.

Rutina de Transmisión con Control de Flujo por Software

- Lee el registro FLUJO que contiene el control de flujo seleccionado para la transmisión, si es "S" continua, de lo contrario salta a Rutina de Transmisión sin Control de Flujo.
- Presenta mensaje de transmisión con control de flujo por Software, junto con el baudrate elegido y los datos a transmitir.
- Carga el baudrate en el registro de configuración y habilita el puerto serial.
- Transmite los caracteres ingresados vía serial usando control de flujo por Software.
- Presenta mensaje de transmisión exitosa junto con los caracteres enviados o de lo contrario un mensaje de error de handshake si este no se cumple.

- Lee registro TECLA y dependiendo de este valor salta a la Rutina de Ingreso de Datos para realizar una nueva transmisión o sale a la opción Transmisión Serial.

Rutina de Transmisión sin Control de Flujo

- Presenta mensaje de transmisión sin control de flujo, junto con el baudrate elegido y los datos a transmitir.
- Carga el baudrate en el registro de configuración y habilita el puerto serial.
- Transmite los caracteres ingresados vía serial sin control de flujo usando la subrutina Transmisión Serial de 10 Caracteres.
- Presenta mensaje de transmisión exitosa junto con los caracteres enviados si la transmisión se completó.
- Lee registro TECLA y dependiendo de este valor salta a la Rutina de Ingreso de Datos para realizar una nueva transmisión o sale a la opción Transmisión Serial.

Rutinas de Transmisión Paralela

Rutina de Transmisión Modo Nibble

- Lee el registro MTPP que contiene el modo seleccionado para la transmisión, si es "N" continua, de lo contrario salta a Rutina de Transmisión Modo Byte..
- Presenta mensaje de transmisión Modo Nibble junto con datos a transmitir.
- Transmite los caracteres ingresados usando el handshake descrito para Modo Nibble.
- Presenta mensaje de transmisión exitosa junto con los caracteres enviados si la transmisión se completó.
- Lee registro TECLA y dependiendo de este valor salta a la Rutina de Ingreso de Datos para realizar una nueva transmisión o sale a la opción Transmisión Paralela.

Rutina de Transmisión Modo Byte

- Presenta mensaje de transmisión Modo Byte junto con datos a transmitir.
- Transmite los caracteres ingresados usando el handshake descrito para Modo Byte.

- Presenta mensaje de transmisión exitosa junto con los caracteres enviados si la transmisión se completó.
- Lee registro TECLA y dependiendo de este valor salta a la Rutina de Ingreso de Datos para realizar una nueva transmisión o sale a la opción Transmisión Paralela.

Rutinas de Recepción de Datos

Estas rutinas reciben hasta 10 caracteres vía puerto serial o paralela.

Rutinas de Recepción Serial

Rutina de Recepción con Control de Flujo por Hardware

- Lee el registro FLUJO que contiene el control de flujo seleccionado para la recepción, si es "H" continua, de lo contrario salta a Rutina de Recepción con Control de Flujo por Software.
- Presenta mensaje de recepción con control de flujo por Hardware, junto con el baudrate elegido y los últimos caracteres a recibidos.
- Carga el baudrate en el registro de configuración, habilita el puerto serial y espera que el transmisor inicie el handshake.
- Recibe los caracteres enviados vía serial usando control de flujo por Hardware.
- Presenta mensaje de recepción exitosa junto con los caracteres recibidos si la recepción se completó.
- Lee registro TECLA y dependiendo de este valor la rutina inicia de nuevo o sale a la opción Recepción Serial.

Rutina de Recepción con Control de Flujo por Software

- Lee el registro FLUJO que contiene el control de flujo seleccionado para la recepción, si es "S" continua, de lo contrario salta a Rutina de Recepción sin Control de Flujo.
- Presenta mensaje de recepción con control de flujo por Software, junto con el baudrate elegido y los últimos caracteres recibidos.
- Carga el baudrate en el registro de configuración, habilita el puerto serial y espera que el transmisor inicie el handshake.
- Recibe los caracteres enviados vía serial usando control de flujo por Software.

- Presenta mensaje de recepción exitosa junto con los caracteres recibidos si la recepción se completó o de lo contrario presenta un mensaje de error de handshake.
- Lee registro TECLA y dependiendo de este valor la rutina inicia de nuevo o sale a la opción Recepción Serial.

Rutina de Transmisión sin Control de Flujo

- Presenta mensaje de recepción sin control de flujo, junto con el baudrate elegido y los últimos caracteres recibidos.
- Carga el baudrate en el registro de configuración y habilita el puerto serial.
- Recibe los caracteres enviados vía serial sin control de flujo.
- Presenta mensaje de recepción exitosa junto con los caracteres recibidos si la recepción se completó.
- Lee registro TECLA y dependiendo de este valor la rutina inicia de nuevo o sale a la opción Recepción Serial.

Rutina de Recepción Paralela Modo Compatible

- Presenta mensaje de recepción Modo Compatible junto con los últimos caracteres recibidos.
- Recibe los caracteres enviados usando el handshake descrito para Modo Compatible.
- Presenta mensaje de Recepción exitosa junto con los caracteres recibidos si la recepción se completó.
- Lee registro TECLA y dependiendo de este valor la rutina inicia de nuevo o sale a la opción Recepción Paralela.

Rutinas de Pruebas Pin a Pin

Estas rutinas leen o escriben los pines del puerto serial o paralelo para probar su funcionamiento.

Rutina de Prueba Pin a Pin Serial

- Lee los pines de puerto C del microcontrolador designados para las líneas CTS y DSR.
- Lee el registro TECLA y de acuerdo a este valor cambia el estado de los pines del puerto C designados para las líneas RTS, DTR y CD o salta a la opción Prueba Pin a Pin Serial.

- Presenta el estado de todas las líneas de handshake en el LCD.

Rutina de Prueba Pin a Pin Paralela

- Lee el estado de los bits del registro de Datos del PC y los presenta en binario y decimal en el LCD.
- Lee el estado de los bits del registro de Control del PC y los presenta en binario y decimal en el LCD.
- Lee el registro TECLA y de acuerdo a este valor cambia el estado de los bits del registro de Estado del PC y los presenta en binario y decimal en el LCD o salta a la opción Prueba Pin a Pin Paralela.

Subrutinas para Cargar Mensajes

- Cargan los registros CAR01:CAR40 con caracteres que corresponden a un mensaje determinado, para luego ser enviados al LCD mediante la Subrutina de envío de Mensajes.

4.2.1 PROGRAMA DEL PUERTO PARALELO

El programa maneja las señales del Puerto Paralelo del PC a través del Puerto D del microcontrolador utilizando la técnica del multiplexaje. Configura el puerto como entrada o como salida de acuerdo a la operación que se desea realizar (lectura de las señales de Datos y Control / escritura de las señales de Datos y Estado). Algunos de los pines de los Puertos B y E son empleados como señales de habilitación de los drivers y latchs utilizados para el multiplexaje.

Para la transmisión de datos en Modo Nibble y Modo Byte, se leen los registros donde se encuentran almacenados los caracteres a transmitir, para luego enviarlos por el Puerto D del microcontrolador utilizando el handshake descrito para cada modo; esto es, leyendo las señales Control y escribiendo las señales de Datos y/o Estado del puerto paralelo del PC.

Para la recepción de datos en Modo Compatible, se guarda cada valor leído del puerto de Datos del PC en las localidades designadas para este fin, y se escribe y lee las señales de Control y Estado respectivamente, cumpliendo el handshake descrito para Modo Compatible.

Para la prueba Bit a Bit del puerto paralelo del computador, se lleva a 1L o a 0L cada una de las señales del puerto de Estado, y se leen las señales de los puertos de Datos y Control del PC. La Subrutina de conversión Hexadecimal a Decimal transforma el valor Hexadecimal leído o escrito en el Puerto D, en cada caso, a decimal, y se presentan este valor y el estado de cada pin a través de la Subrutina de envío de Carácter y Posición al LCD.

4.2.2 PROGRAMA DEL PUERTO SERIAL

Aprovechando que los pines 7 y 6 del Puerto C (RxD y TxD) son usados por el USART (Universal Synchronous Asynchronous Receiver Transmiter) del microcontrolador, las señales del Puerto Serial del PC se manejan utilizando dicho Puerto.

Antes de realizar una transmisión o recepción Serial se debe configurar el USART del microcontrolador. La velocidad de transmisión se selecciona cargando en el registro de configuración del baudrate un valor decimal proveniente de la formula Baud Rate = Fosc / 64(X + 1), donde X es el valor decimal y Fosc es la frecuencia de oscilación del cristal u oscilador . Entonces, para cambiar el valor del baudrate, simplemente se carga el valor X correspondiente a la velocidad deseada en el registro de configuración del baudrate.

El USART del microcontrolador no realiza por si solo el control de flujo de la comunicación, por lo que se implementa el handshake de Software y Hardware mediante código de programa siguiendo el procedimiento indicado. En el caso del control de flujo por Hardware se utilizan los pines restantes del Puerto C del microcontrolador.

Para desarrollar una transmisión serial de datos se leen las localidades de memoria designadas para almacenar los caracteres a enviar, se escriben dichos caracteres uno a uno en el registro de envió de datos serial y se chequea la bandera de estado del buffer de transmisión para comprobar su envío. Los

caracteres transmitidos son mostrados en la pantalla del LCD junto con un mensaje de transmisión exitosa.

Para recibir datos vía puerto Serial se chequea el estado de la bandera del buffer de recepción y se almacenan los datos recibidos uno a uno en las localidades de memoria designadas para este propósito. Los caracteres recibidos son mostrados en la pantalla del LCD junto con un mensaje de recepción exitosa.

Las pruebas bit a bit de las señales de handshake se efectúan leyendo o escribiendo en dichas líneas del puerto Serial del computador. El programa lee el estado de los bits del puerto y los presenta a través de la Subrutina de envío de Carácter y Posición al LCD.

4.3 DISEÑO DEL HARDWARE DEL SISTEMA

4.3.1 CONSIDERACIONES DE DISEÑO

Para la construcción del hardware del modulo, se debieron analizar algunas alternativas en cuanto a los dispositivos a usar, ya que su elección esta íntimamente ligada al hardware del puerto paralelo y serial de los computadores, a los requerimientos y consideraciones de los protocolos de comunicación y a la programación.

Como elemento inteligente, el módulo contiene un microcontrolador PIC 16F877A de Microchip, el cual se encarga del manejo de las capacidades de todo el hardware. Para la elección de microcontrolador PIC 16F877A se consideraron principalmente: sus múltiples puertos digitales, su USART (Universal Synchronous Asynchronous Receiver Transmiter), sus fuentes de interrupción externa, su capacidad de memoria de programa, su memoria tipo Flash de programación, su bajo consumo de potencia y su bajo costo.

El microcontrolador PIC se basa en la arquitectura Harvard, en la cual el programa y los datos se pueden trabajar desde memorias separadas, lo que

posibilita que las instrucciones y los datos posean longitudes diferentes. Esta misma estructura es la que permite la superposición de los ciclos de búsqueda y ejecución de las instrucciones, lo cual se ve reflejado en una mayor velocidad del microcontrolador.

El PIC 16F877A posee 5 puertos que pueden ser configurados alternativamente como entradas o salidas digitales mediante software. Esta capacidad permite utilizar un solo puerto del microcontrolador para el manejo de las señales de entrada (bits de Datos y Estado) y salida (bits de Datos y Control) del puerto paralelo del computador, utilizando la técnica de multiplexaje. Además la capacidad de manejo bit a bit de sus puertos permite utilizar sus pines como señales de habilitación para latchs y drivers/buffers, y como señales de handshake para control de flujo de una comunicación.

Este microcontrolador posee un USART de manejo muy sencillo. Simplemente escribiendo o leyendo los registros de transmisión o recepción es posible comunicarse vía Serial con otro dispositivo. La velocidad de transmisión esta determinada por el valor Baud Rate = Fosc/(64(X+1)), donde Fosc es la frecuencia de oscilación del cristal u oscilador y X el numero que se debe cargar en el registro de configuración del baudrate. Si se reemplaza Fosc = 3686400 en la formula, se obtienen valores exactos de baudrate (300, 1200, 2400, 9600, 19200, 28800), es esta la razón por la que se elige un oscilador de 3.6864 MHz.

Los pines RB4:RB7 del puerto B del PIC pueden ser configurados como fuentes de interrupción externa. Un cambio de estado en cualquiera de estos pines hará que el contador de programa salte a la dirección 04H de memoria de programa (vector de interrupción), donde se realizan rutinas de atención a la interrupción. Esta fuente de interrupción es útil para el manejo de teclado, y es para este fin que se la utiliza.

Este PIC tiene 8K de memoria de programa tipo Flash. Este tipo de memoria se puede programar y borrar eléctricamente, lo que facilita el desarrollo de programas y la experimentación.

Para la elección de los dispositivos conectados directamente al puerto paralelo, se toman las siguientes consideraciones:

En el puerto paralelo original, un flip-flop 74LS373 manejaba las ocho líneas de datos. Inversores de colector abierto 7504 manejaban las líneas de control y las líneas de estado eran conectadas a entradas de compuertas lógicas LSTTL. Pero hoy en día es muy difícil determinar con certeza que componentes son utilizados para los circuitos del puerto paralelo; pues su diseño varia con los fabricantes y la IEEE no define exactamente los tipos adecuados y un estándar a seguir.

Se tomo en cuenta que todo puerto paralelo tiene los mismos 17 bits, y sabiendo que estos pueden diferir en aspectos como la inmunidad al ruido y la impedancia de salida; se tiene bien claro que cada salida del puerto paralelo debe cumplir al menos con los mínimos requerimientos de suministro y drenaje de corriente del puerto original.

Por otro lado, analizando los tipos de salidas existentes y su característica de suministro y drenaje de corriente, además de las posibilidades de conexión entre estas se llego a la siguiente determinación:

Se prefiere el uso de circuitos integrados de la familia HCT para el caso de los drivers/buffers conectados a las salidas de los puertos de datos y control y a la entrada del puerto de estado, por las siguientes razones:

- Los dispositivos HCT tienen voltajes de entrada compatibles con la lógica TTL, lo que en primer lugar asegura compatibilidad con los puertos de salida del puerto paralelo.
- Las salidas de los circuitos integrados HCT pueden suministrar y drenar suficiente cantidad de corriente, como para asegurar el encendido de los leds.

- Las salidas de los dispositivos HCT tienen comportamiento similar a las salidas de colector-abierto. Lo que permite mayor seguridad y protección de los circuitos.
- Aunque no tienen compuertas de entrada Smith-Trigger, como los LSTTL, proveen una baja inmunidad al ruido (característica CMOS) y un bajo consumo de corriente.

Por otro lado, las resistencias conectadas a las entradas de datos del puerto, controladas por las salidas del Buffer Octal 74HCT244, son para salvaguardar el puerto en caso de que ocurriera que tanto las salidas propias del puerto de datos como las salidas del Buffer Octal estuvieran a la vez en alto. Esta resistencia limita la corriente en cada línea bajo los 15 mA.

Se aclara que aunque en los primeros puertos, las salidas del puerto de control podían ser utilizadas como entradas (los primeros diseños utilizaban salidas de colector abierto), en la propuesta se descarta esta posibilidad por considerarla insegura; pues, muchos de los nuevos puertos para mejorar la velocidad han cambiado estos circuitos por circuitos tipo push-pull, lo que imposibilitaría el uso de las salidas como entradas.

Considerando entonces que el modulo de pruebas puede ser utilizado eventualmente sobre cualquier PC, no se usan las salidas del puerto de control como entradas.

Para la elección de los dispositivos conectados al puerto serial, se toma en cuenta lo siguiente:

Debido a que el puerto serial de los computadores trabaja con niveles RS232 y el microcontrolador PIC trabaja con lógica TTL y CMOS, es necesario utilizar un convertidor RS232 a TTL y viceversa, para la interfaz entre el puerto serial del computador y el puerto serial del PIC.

Se considera el uso de capacitores conectados entre VCC y GND de cada circuito integrado, debido a que almacenan la energía requerida por las compuertas. Todos los integrados atraen corriente a medida que responden a los cambios en sus entradas. Cuando la corriente esta disponible en un condensador cercado, la compuerta puede cambiar rápidamente, sin ocasionar picos en la fuente de energía o en las líneas de tierra.

Para mostrar el estado de las señales de los bits de los puertos serial y paralelo del computador se disponen de arreglos de leds para cada puerto.

Estos leds son manejados a través de buffers 74HCT244 y latchs 74LS373, los cuales mantienen el último estado lógico escrito desde el PC o desde el PIC, según la dirección de las líneas de los puertos.

Para la interfaz de visualización del modulo se eligió una pantalla LCD de 2*20, por ser suficiente para nuestros propósitos.

Las Figuras 4.4 y 4.5 presentan un esquema del hardware que contiene el módulo para el puerto Serial y Paralelo respectivamente.

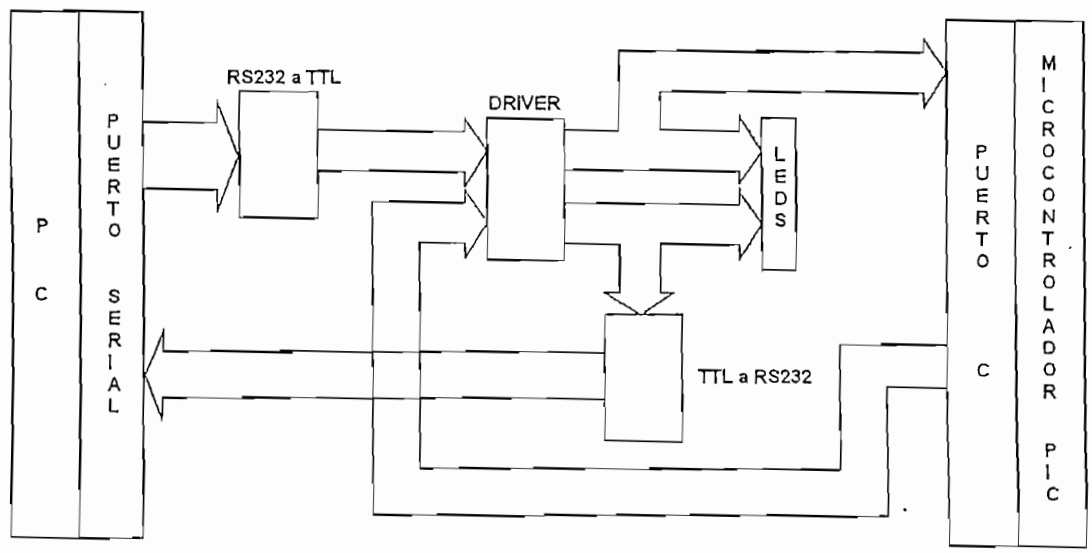


Figura 4.4
Esquema del hardware del Módulo para el Puerto Serial

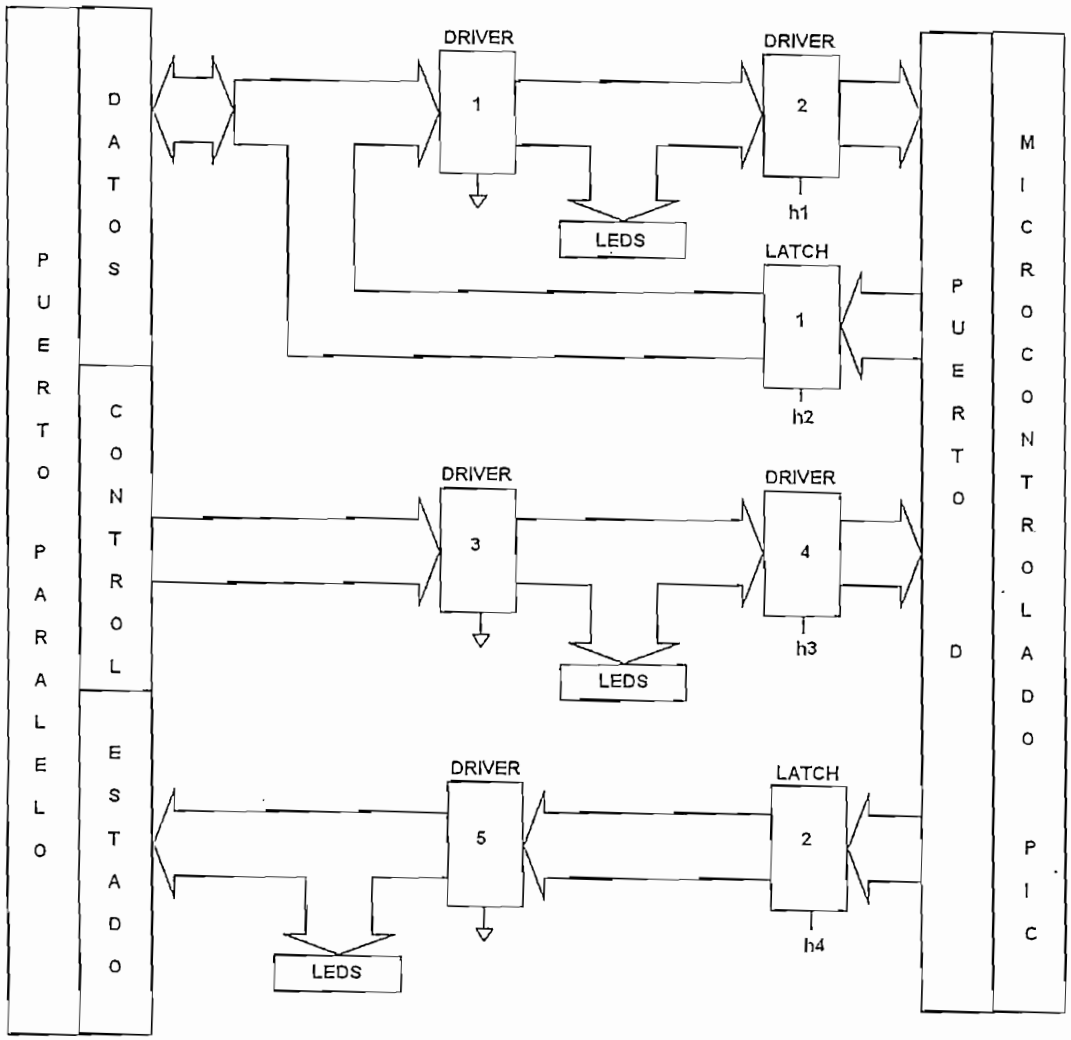
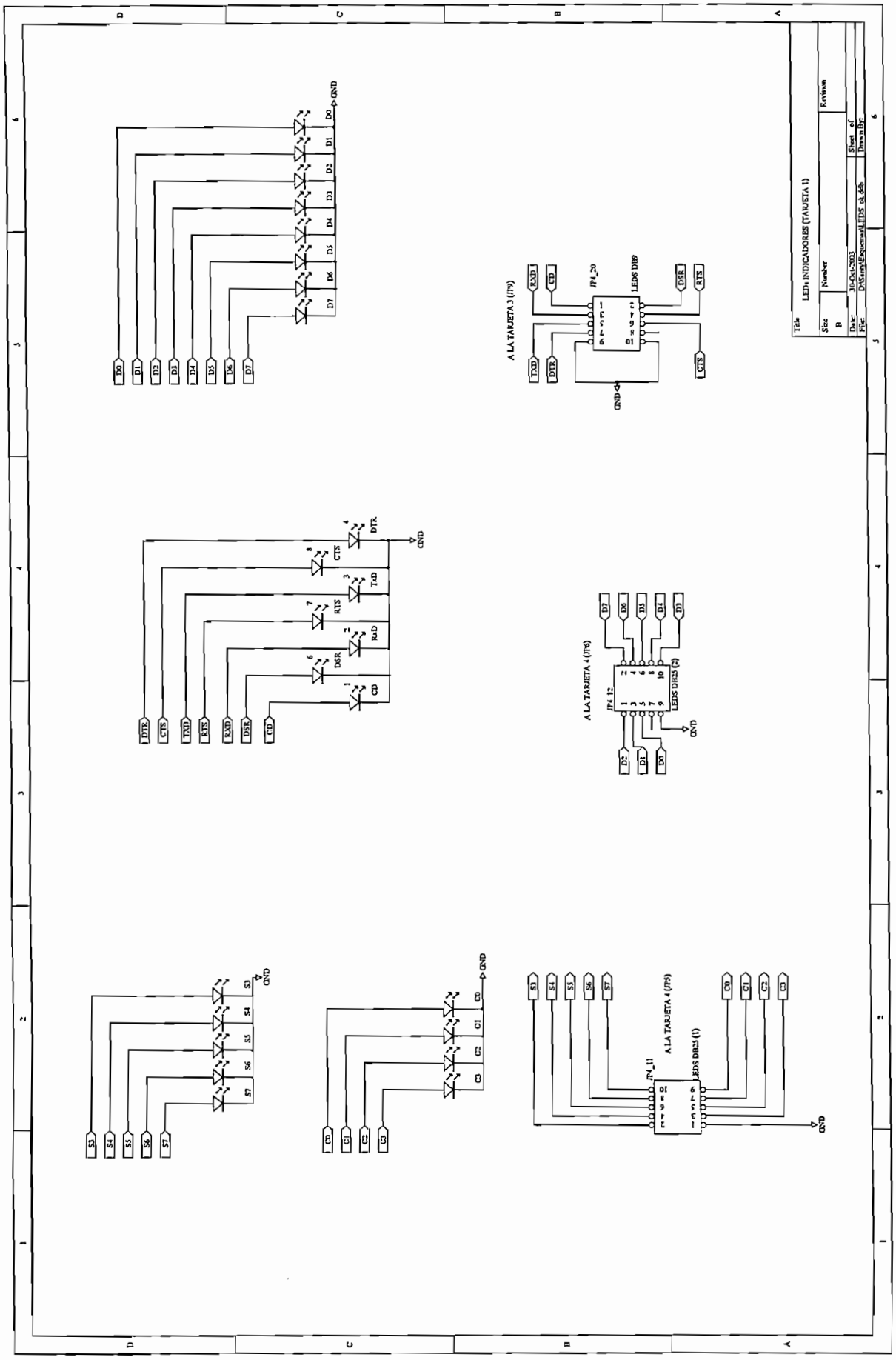


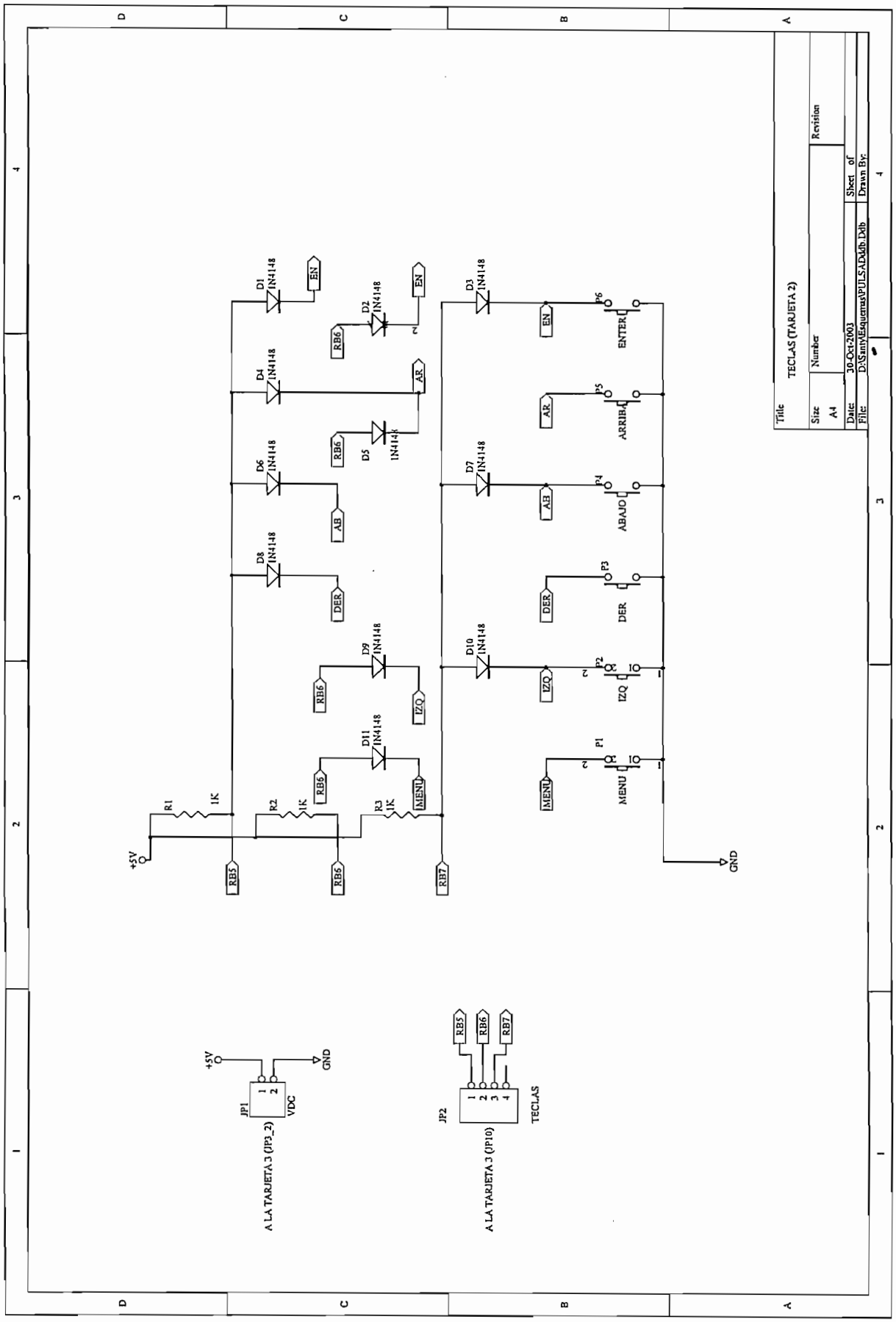
Figura 4.5

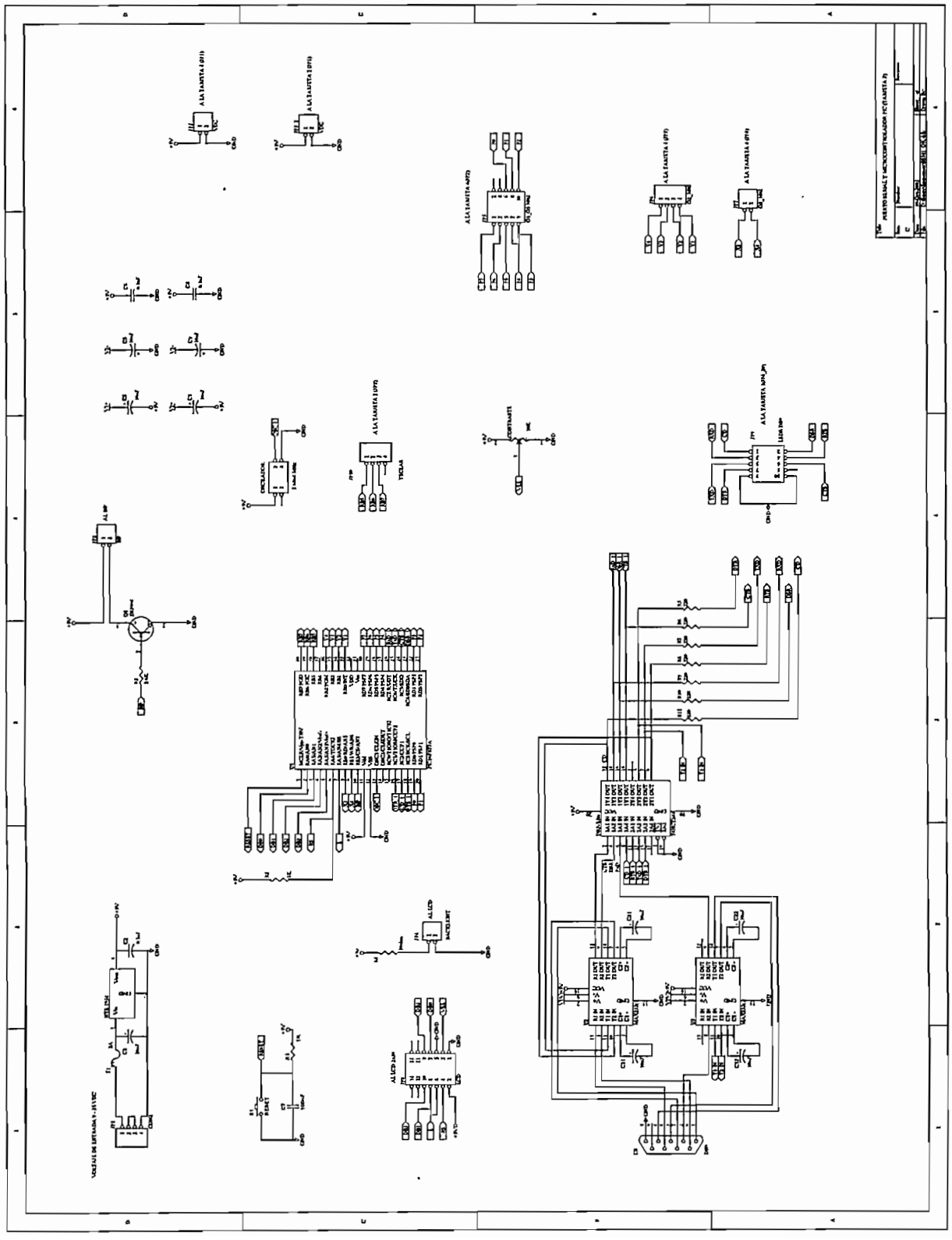
Esquema del hardware del Módulo para el Puerto Paralelo

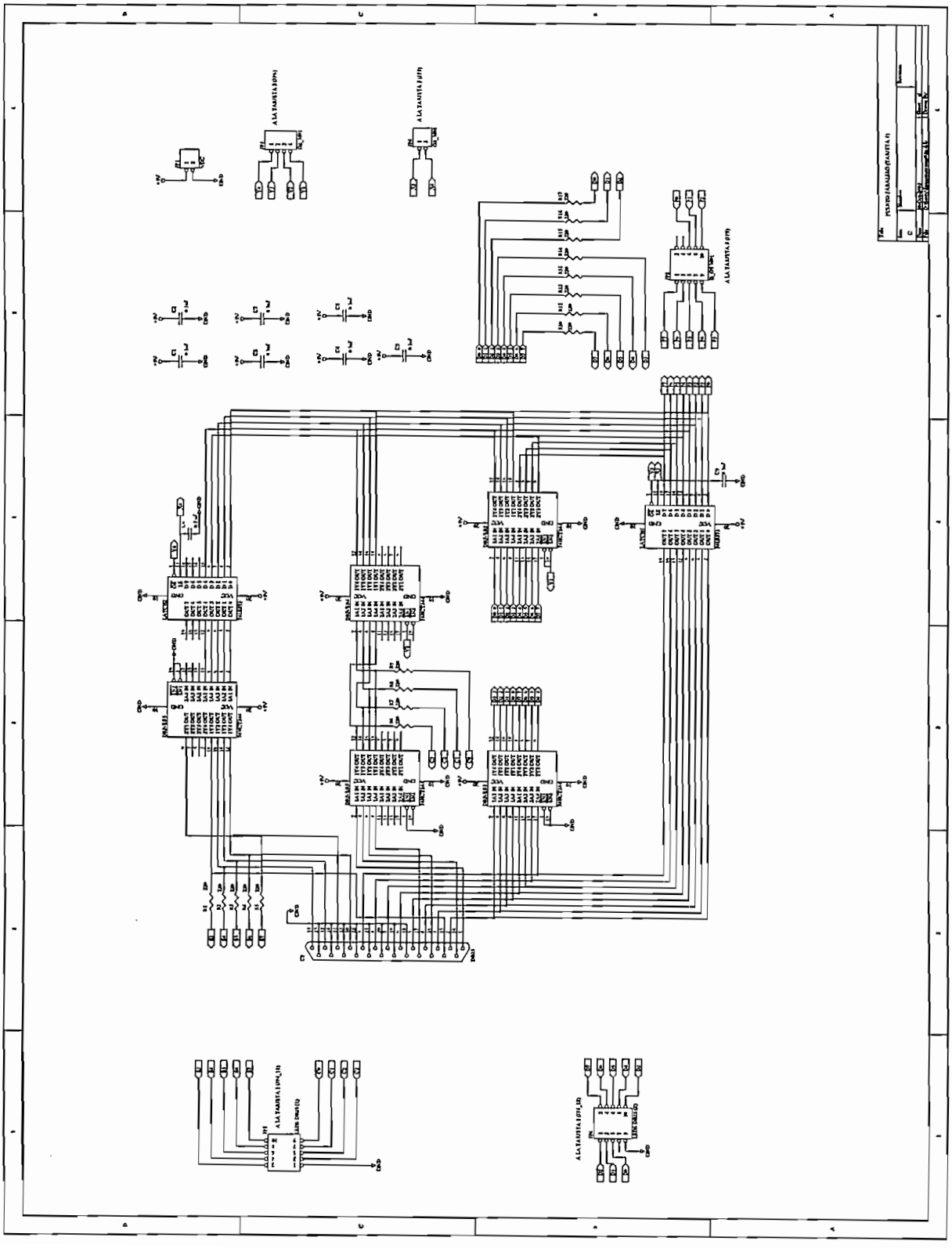
4.3.2 ESQUEMATICOS

A continuación se presentan los esquemáticos de las tarjetas realizadas.


LEDs INDICADORES (TARJETA 1)
A LA TARJETA 3 (J19)
A LA TARJETA 4 (J16)
A LA TARJETA 4 (J9)
LEDS D0-D7
LEDS D8-D15 (2)
LEDS D16-D23 (1)
GND

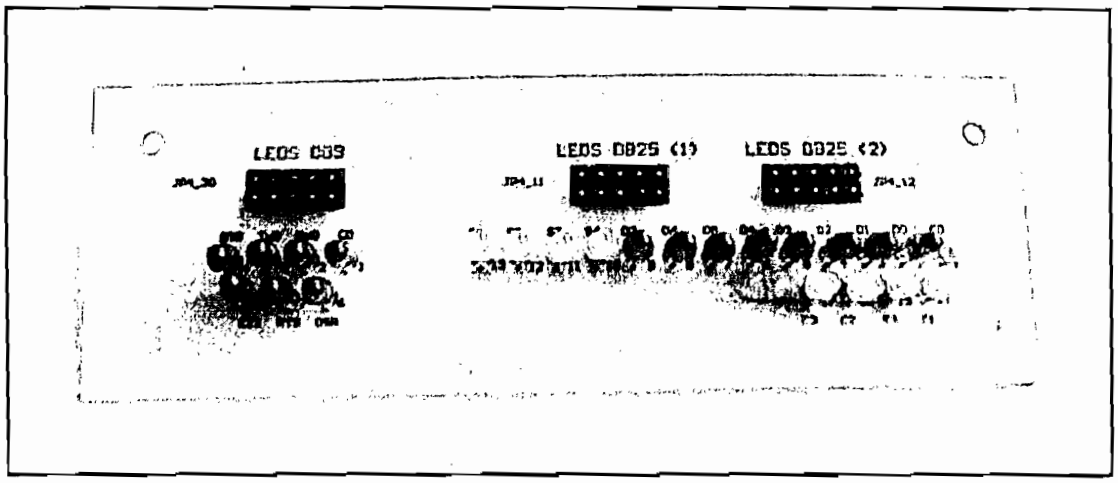

D1 IN4148
D2 IN4148
D3 IN4148
D4 IN4148
D5 IN4148
D6 IN4148
D7 IN4148
D8 IN4148
D9 IN4148
D10 IN4148
D11 IN4148
EN
AR
AB
DER
IZQ
MENU
RB5
RB6
RB7
R1 1K
R2 1K
R3 1K
+5V
GND
P1 MENU
P2 IZQ
P3 DER
P4 ABAJO
P5 ARRIBA
P6 ENTER
JP1
VDC
A LA TARJETA J (JP3_2)
JP2
TECLAS
A LA TARJETA J (JP10)
Title TECLAS (TARJETA 2)
Size A4
Number
Revision
Date: 30-Oct-2003
File: D:\Santy\Esquemas\PIJ1-SAD\db.Ddb
Sheet of
Drawn By:



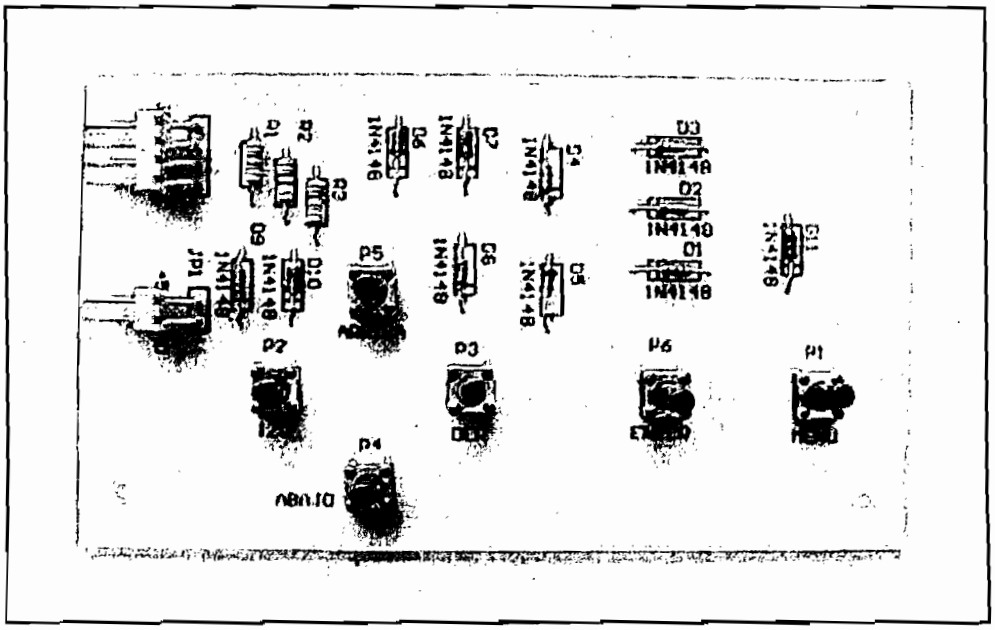


4.3.3 TARJETAS

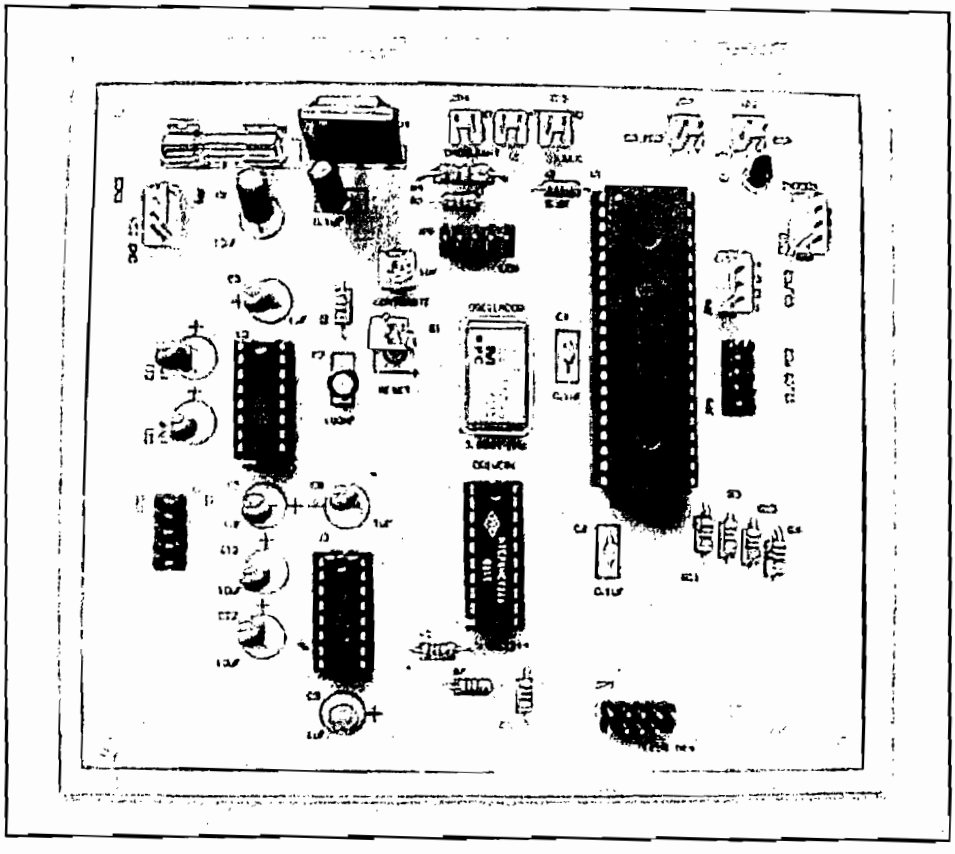
A continuación se presentan las tarjetas que conforman el hardware del módulo.



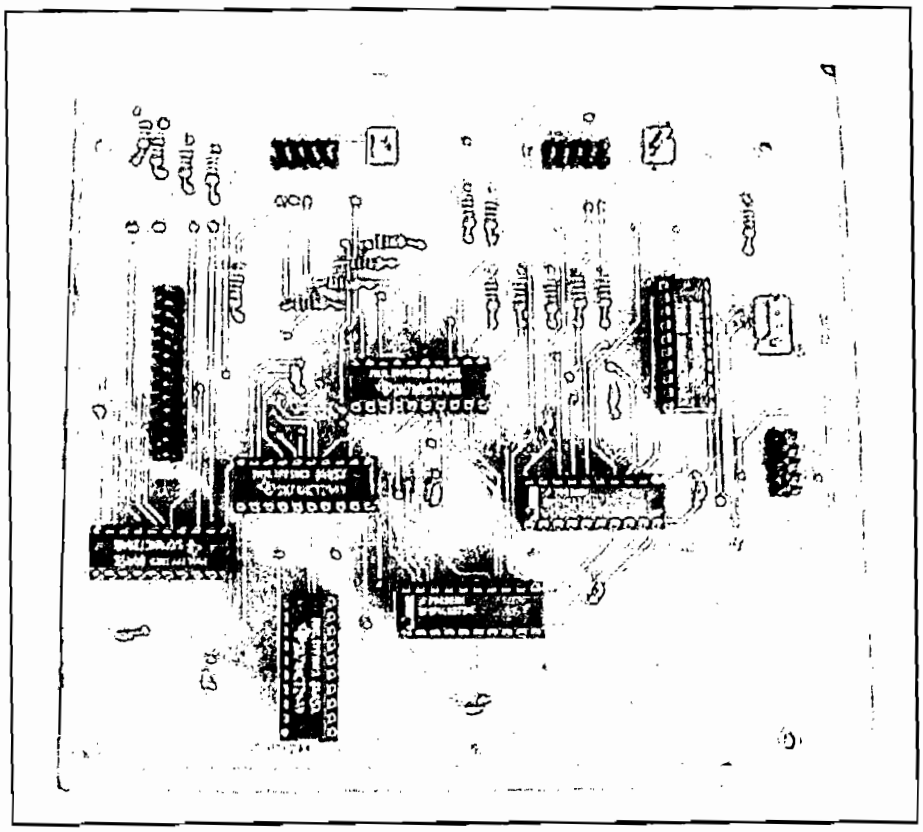
Leds Indicadores
Tarjeta 1



Teclas
Tarjeta 2



Puerto Serial y Microcontrolador PIC
Tarjeta 3



Puerto Paralelo
Tarjeta 4

4.4 FUNCIONAMIENTO DEL SISTEMA

Como ya se indicó con anterioridad, el sistema de prueba de puertos paralelo y serial de un computador consta de un Módulo de Comunicaciones y un Programa de Soporte en el computador.

4.4.1 MÓDULO DE COMUNICACIONES

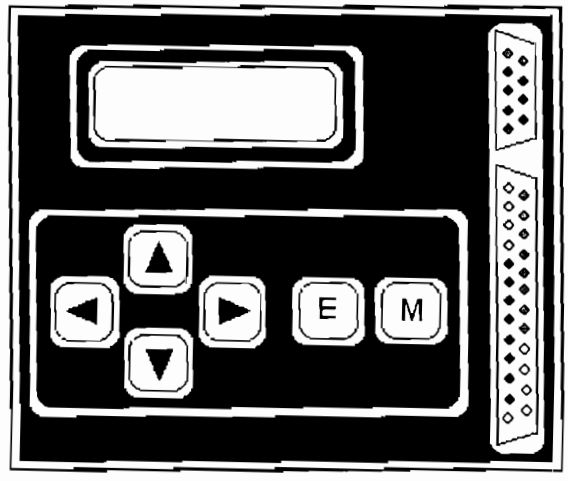


Figura 4.6
Módulo de Comunicaciones

El Módulo (Ver Figura 4.6) consta de una pantalla de cristal líquido LCD de 2x20, 24 leds indicadores del estado de las señales del puerto serial y paralelo del PC, 6 teclas, 1 conector DB9 macho para el puerto serial (Ver Figura 4.7), 1 conector Centronics hembra para el puerto paralelo (Ver Figura 4.8), un pulsante de Reset, un switch de encendido ON/OFF, un jack para conectar la fuente de alimentación y un adaptador AC/DC de 12VDC/2A.

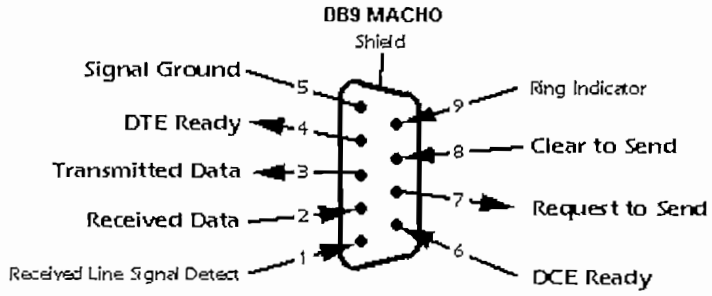


Figura 4.7

Conector DB9 para Puerto Serial

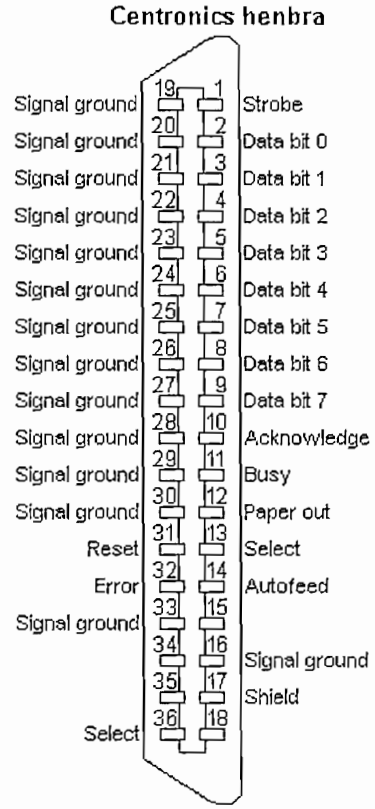


Figura 4.8

Conector Centronics para Puerto Paralelo

Además se incluyen: 1 conector DB9 para la prueba de Loopback (Ver Figura 4.9), 1 cable cruzado DB9 para el puerto Serial (Ver Figura 4.10)y 1 cable directo con un conector DB25 en un extremo y un conector Centronics en el otro para el puerto Paralelo (Ver Figura 4.11).

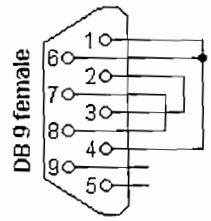


Figura 4.9

Conector hembra DB9 Loopback

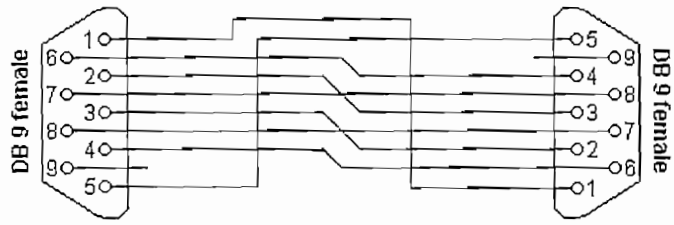


Figura 4.10

Cable cruzado Serial

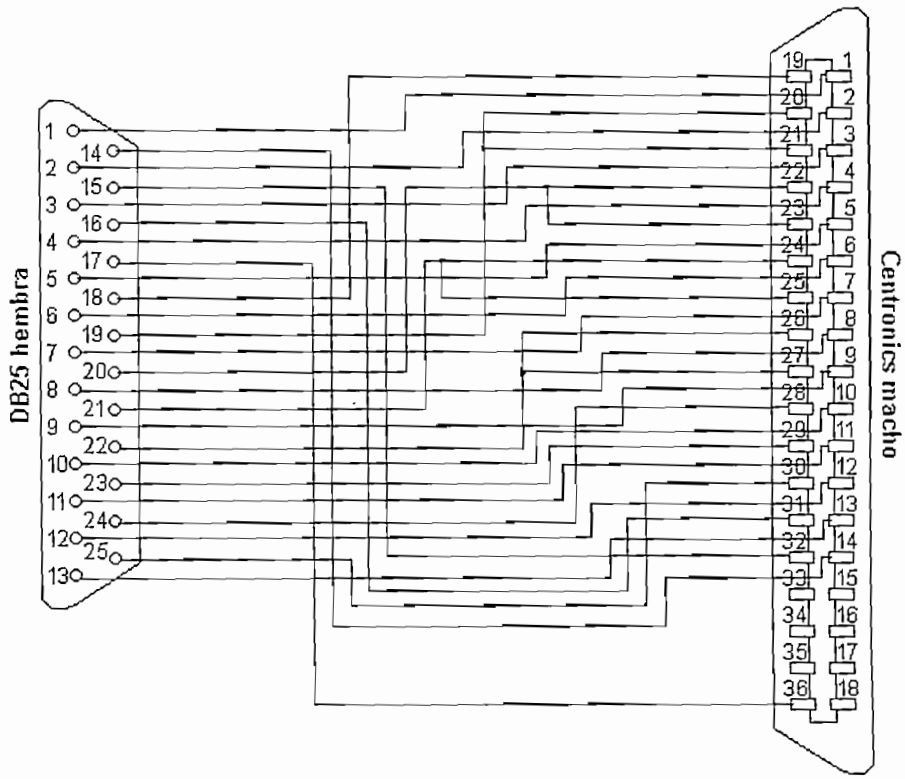


Figura 4.11

Cable DB25 a Centronics Paralelo

A continuación se presentan las teclas del menú y su función en el Módulo:

Arriba: Con el cursor parpadeante, cambia los caracteres de: " " a "Z".
Tecla para viajar en los menús. Muestra bits Datos, Estado y Control.

Abajo: Con el cursor parpadeante, cambia los caracteres de: "Z" a " ".
Tecla para viajar en los menús. Muestra bits Datos, Estado y Control

Izquierda: Desplaza hacia la izquierda el cursor parpadeante en el LCD.

Derecha: Desplaza hacia la derecha el cursor parpadeante en el LCD.

Enter: Selecciona la opción escogida en un menú. Cambia el estado lógico del bit parpadeante.

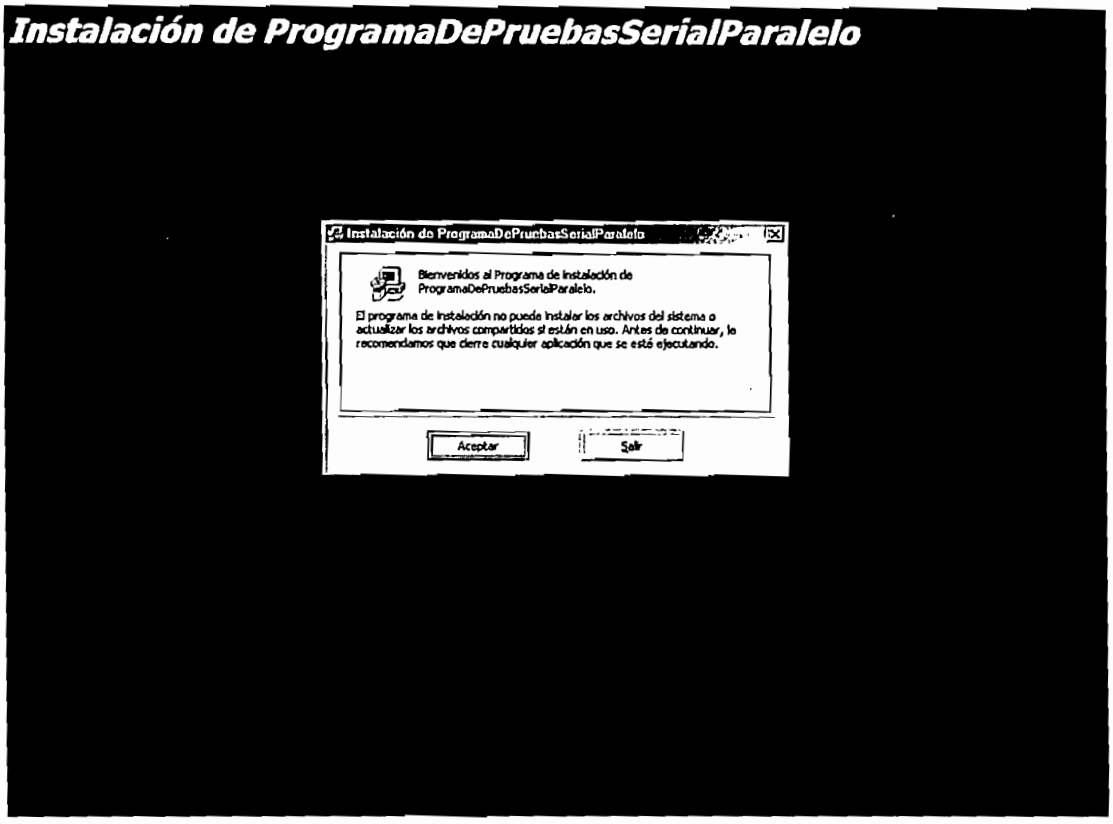
Menú: Despliega la pantalla del menú principal. Retorno al menú anterior.

Se puede ingresar los caracteres " ","0","1",.....,"9","A","B",.....,"X","Y","Z" como datos para ser transmitidos.

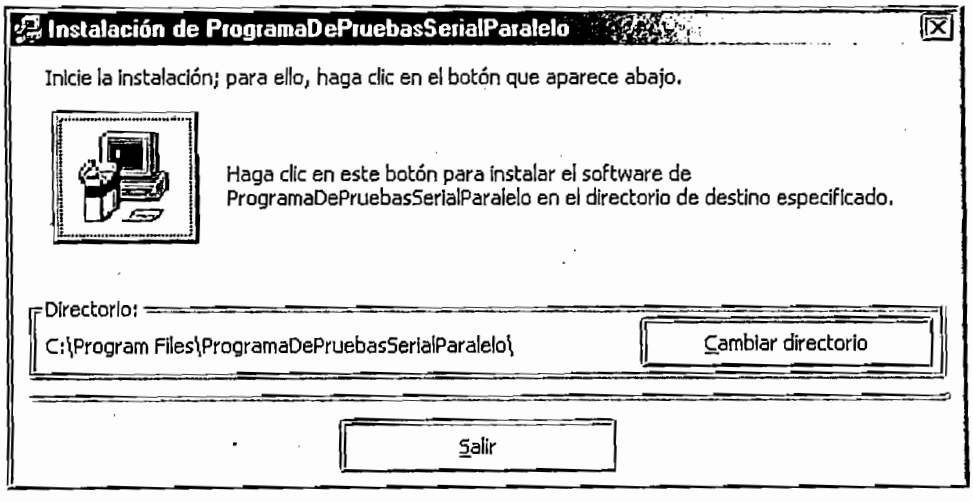
4.4.2 INSTALACIÓN Y EJECUCIÓN DEL PROGRAMA DE SOPORTE

El programa de Prueba de Comunicaciones corre bajo sistemas de 32 bits (NT/2000/XP) y por tanto requiere al menos una PC con procesador INTEL Pentium.

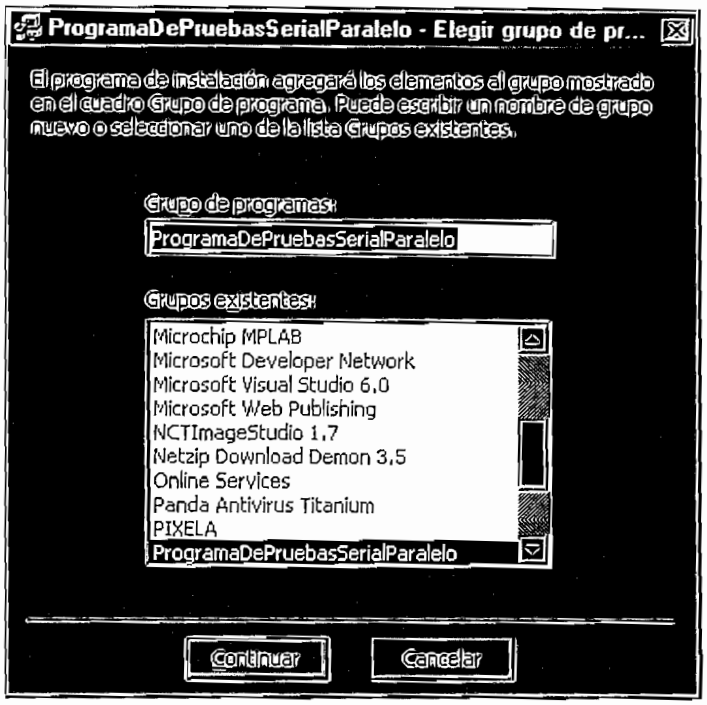
Para instalar el software de soporte, inserte el CD de instalación y haga doble clic en el archivo SETUP.EXE o utilice la opción - Agregar o quitar programas - que se encuentra en - Configuración - del menú de Inicio de Windows.
Al ejecutarse SETUP.EXE inicia el programa de instalación.


Instalación de ProgramaDePruebasSerialParalelo
Instalación de ProgramaDePruebasSerialParalelo
Bienvenidos al Programa de instalación de ProgramaDePruebasSerialParalelo.
El programa de instalación no puede instalar los archivos del sistema o actualizar los archivos compartidos si están en uso. Antes de continuar, le recomendamos que cierre cualquier aplicación que se esté ejecutando.
Aceptar
Salir

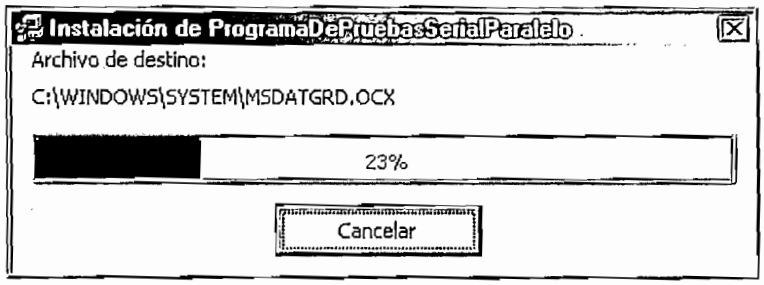
Presione 
Aceptar
.


Instalación de ProgramaDePruebasSerialParalelo
Inicie la instalación; para ello, haga clic en el botón que aparece abajo.
Haga clic en este botón para instalar el software de ProgramaDePruebasSerialParalelo en el directorio de destino especificado.
Directorio:
C:\Program Files\ProgramaDePruebasSerialParalelo\
Cambiar directorio
Salir

Elija el directorio donde desea instalar el programa y presione  .

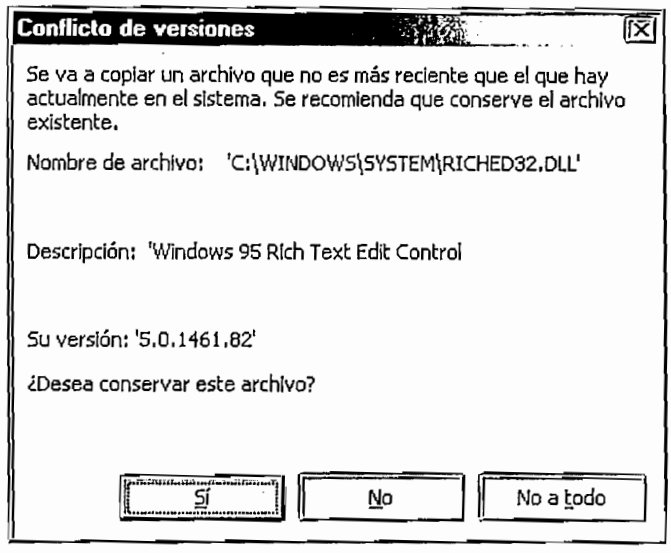


Presione  para iniciar la instalación y espere un momento mientras corre la secuencia de instalación del programa.



Luego de terminar la instalación reinicie el computador.

Puede ocurrir que ya existan en el sistema operativo del computador versiones actualizadas de los archivos que se están instalando.



De ser el caso se recomienda conservar el archivo existente.

4.4.3 INICIO DEL SISTEMA

Realice la siguiente secuencia para iniciar el sistema:

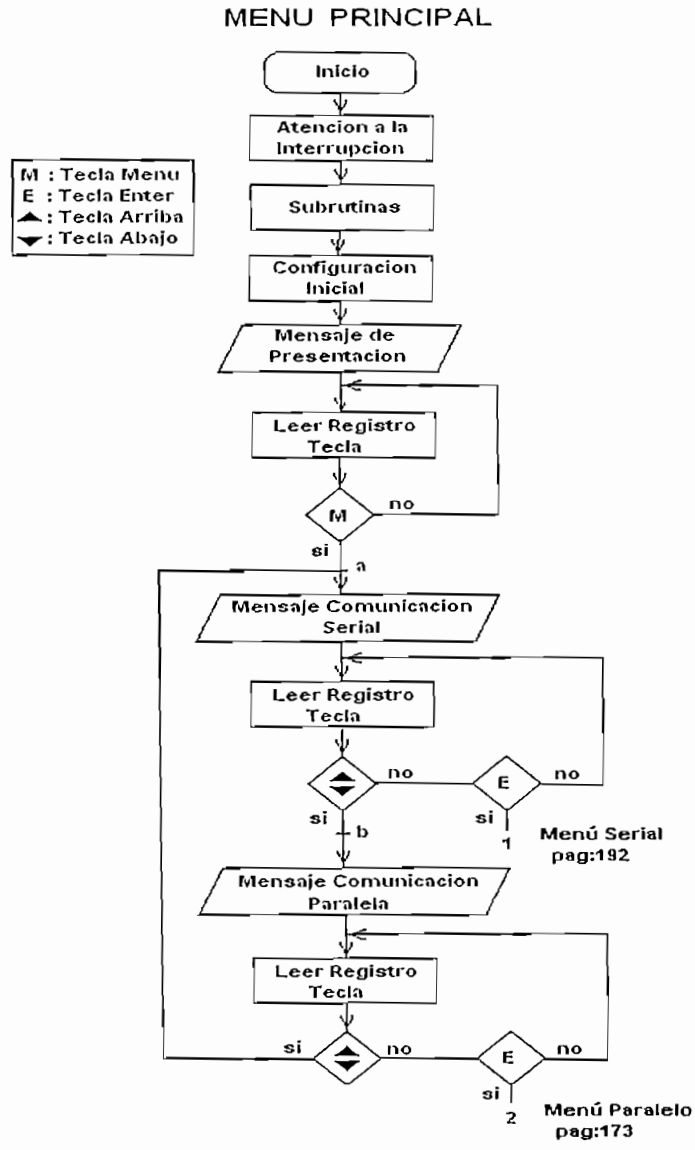
- Encienda el computador y el Módulo, asegurándose de que no estén conectados ninguno de los cables en los puertos de comunicación.
- Abra el menú de Inicio de Windows en el computador, ingrese a "Programas ", y ejecute "Programa de Pruebas Serial Paralelo".
- Tome el cable serial y el cable paralelo e insértelos en los puertos serial y paralelo del PC y Módulo respectivamente.

Es importante recordar que no se deben interconectar el computador y el módulo si no están ambos encendidos, pues al someter el hardware de los puertos a voltajes sin estar alimentados, puede causar averías o daños permanentes en los mismos.

En adelante para realizar cualquiera de las pruebas, se debe elegir la misma opción tanto en los menús del Módulo como en los del programa de soporte en el PC. De lo contrario aparecerán ventanas de Error en el PC y/o el Módulo y ninguna de las pruebas se realizará con éxito.

Cada prueba de los puertos del PC se realiza en conjunto con el Módulo. A continuación se describe la configuración individual de cada uno de estos.

En el Módulo:



Inicio

- Declaración del PIC a utilizar (PIC16F877A).
- Declaración del registro añadido que define los Registros de Función Especial (p16f877a.inc).
- Definición de Variables a utilizar en el programa (Registros de Propósito General).
- Definición de dirección Flash de inicio de escritura del programa.

Atención a la interrupción

Cada vez que se presiona una tecla el programa salta a esta rutina (Vector de Interrupción).

- Salvar registros de trabajo, estatus y bits más significativos del contador de programa.
- Añadir un retardo para evitar el transitorio o rebote al presionar una tecla.
- Leer la tecla presionada y almacenar el valor en el registro Tecla.
- Recuperar registro de trabajo, estatus y bits más significativos del contador de programa.
- Regresar a la localidad de flash ROM desde donde saltó mas uno.

Subrutinas

El programa utiliza subrutinas que realizan tareas específicas que son llamadas desde cualquier parte del mismo. A continuación se las describe.

Configuración Inicial

- Configurar los puertos y habilitar la interrupción por cambio de RB4:RB7.
- Deshabilitar pines que controlan latchs y drivers.
- Inicializar valores de localidades de memoria que guardan caracteres a transmitir, recibir y presentar.
- Enviar bytes de control al LCD.

Menú Principal

Mensaje de Presentación

- Enviar mensaje de Presentación al LCD.

MODULO DE COMUNICACIONES

- Esperar hasta que se presione la tecla *Menu* .

Mensaje Comunicación Serial

- Presentar mensaje de la opción 1 del Menú Principal.

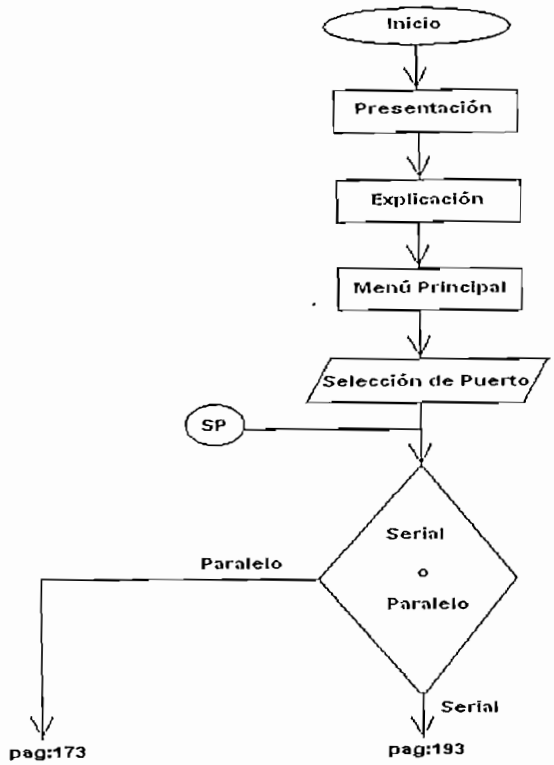
- Revisar que tecla se presiona.
- Tecla *Arriba* o *Abajo* : saltar a Mensaje de Comunicación Paralela.
- Tecla *Enter :* saltar al Menú Serial.

Mensaje Comunicación Paralela

- Presentar mensaje de la opción 2 del Menú Principal.

- Revisar que tecla se presiona.
- Tecla *Arriba* o *Abajo* : saltar a Mensaje Comunicación Serial.
- tecla *Enter* : saltar a Menú Paralelo.

En el Computador:



Inicio

Presentación

- Carga el formulario Presentación, que muestra información del Proyecto de Titulación (Ver Figura 4.12).

Explicación

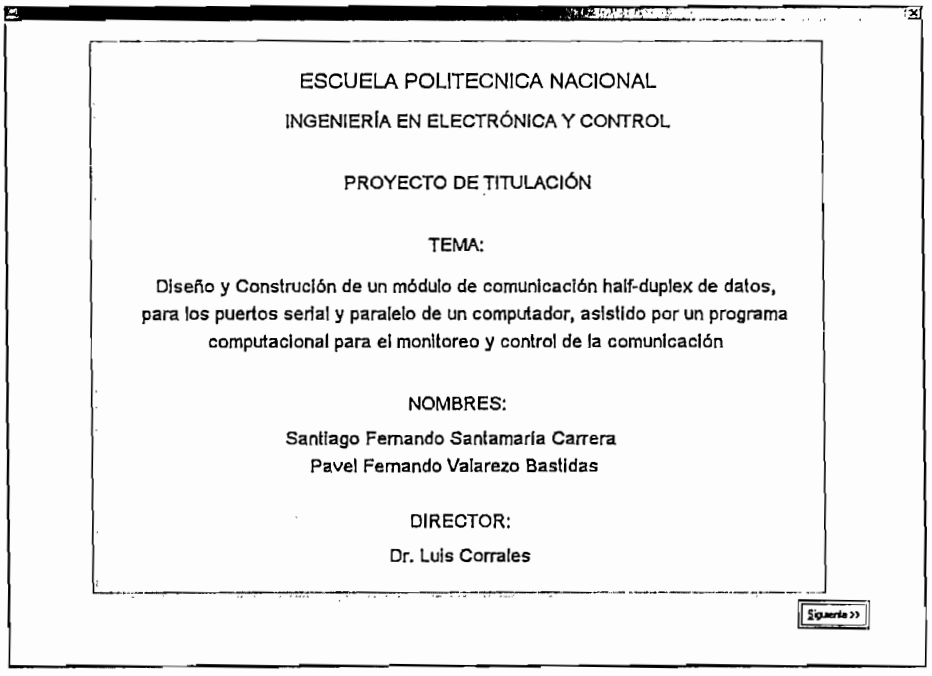
- Carga el formulario Explicación, que muestra en pantalla una explicación de las herramientas disponibles para el programa de soporte (Ver Figura 4.13).

Menú Principal

- Carga el formulario Menú Principal y muestra la pantalla de selección de puertos (Ver Figura 4.14).

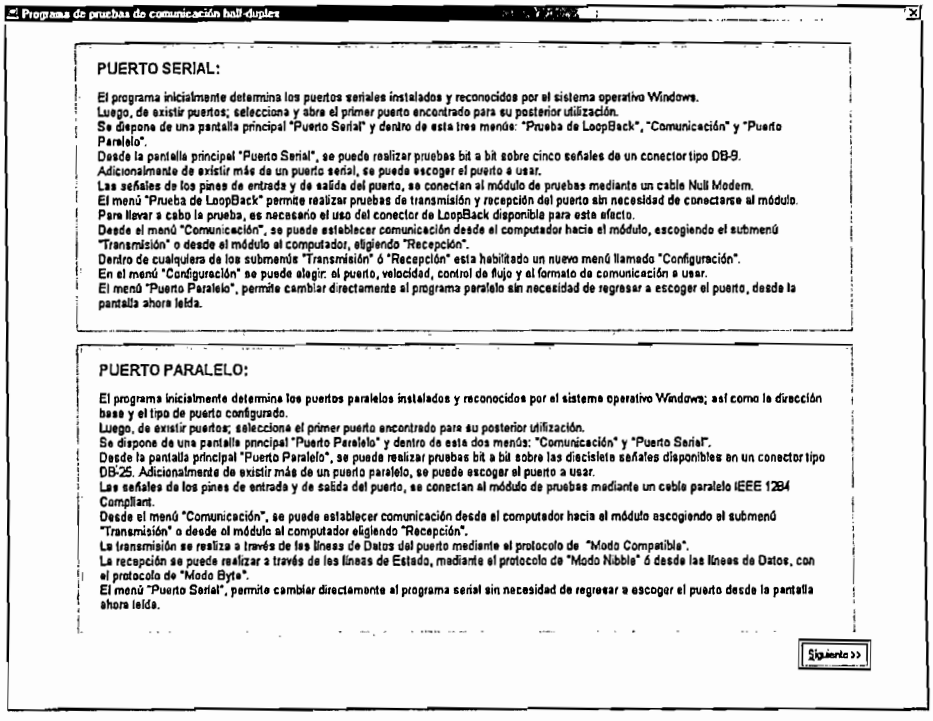
Selección de Puertos

- Se elige el puerto a usar, chequeando un cuadro de control con la opción adecuada.



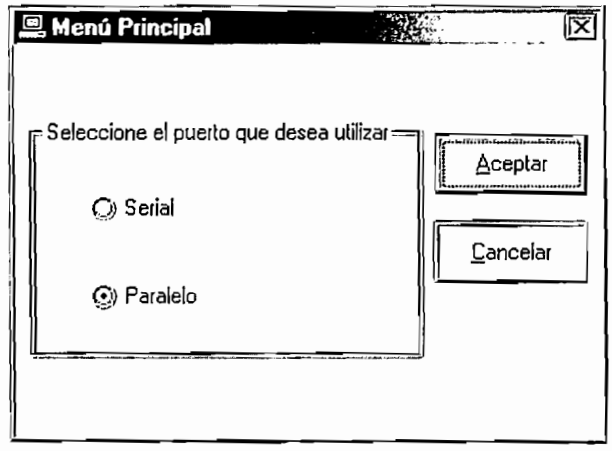
Pantalla de Presentación

Figura 4.12



Pantalla de Explicación

Figura 4.13

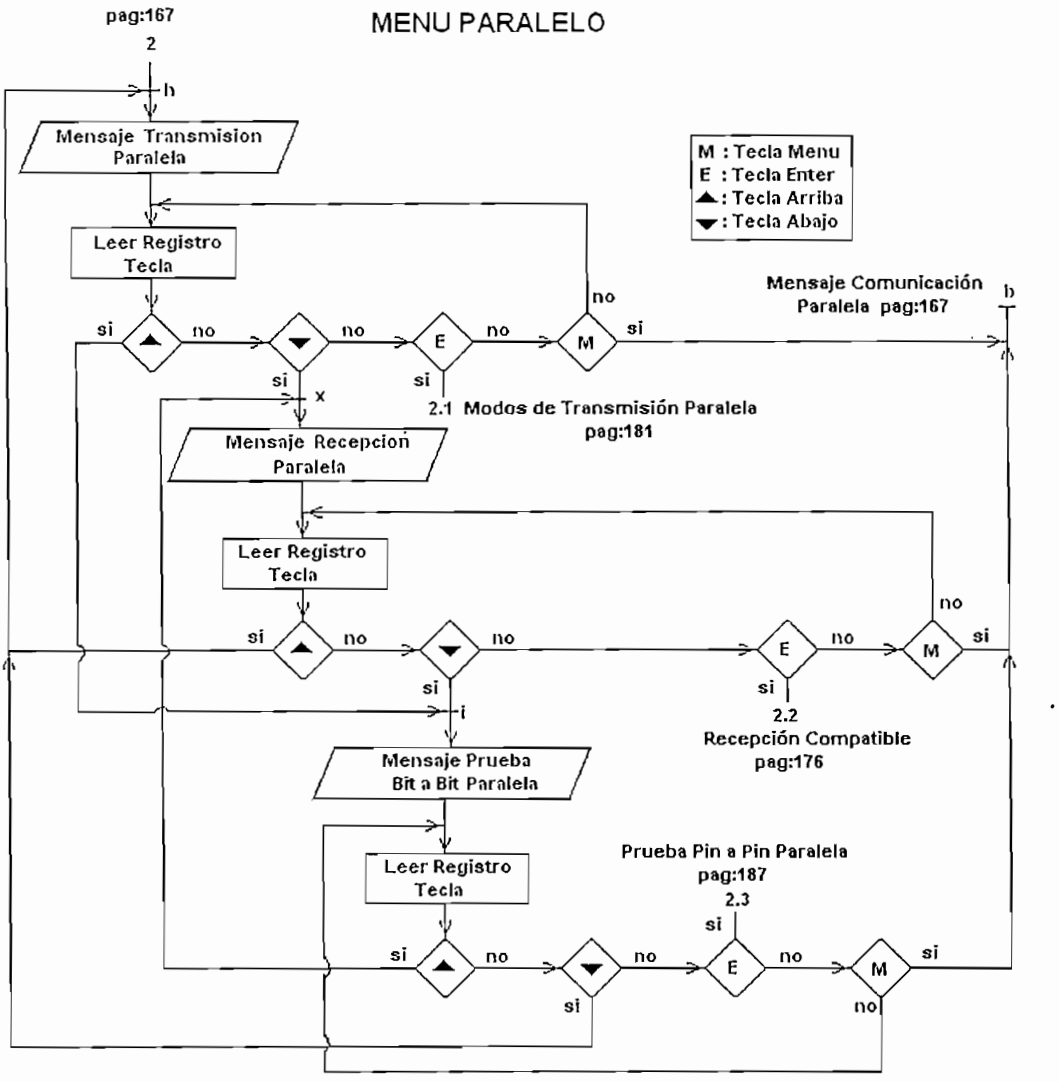


Ventana de selección de Puerto

Figura 4.14

4.4.4 PUERTO PARALELO

En el Módulo:



Menú Paralelo

Mensaje Transmisión Paralela

- Presentar mensaje de la opción 1 del Menú Paralelo.

<1> TRANSMISION VIA PUERTO PARALELO

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Prueba Pin a Pin Paralela.

- Tecla *Abajo*: saltar a Mensaje Recepción Paralela.
- Tecla *Enter* : saltar a Menú Transmisión Paralela.
- Tecla *Menu*: saltar a Mensaje Comunicación Paralela.

Mensaje Recepción Paralela

- Presentar mensaje de la opción 2 del Menú Paralelo.

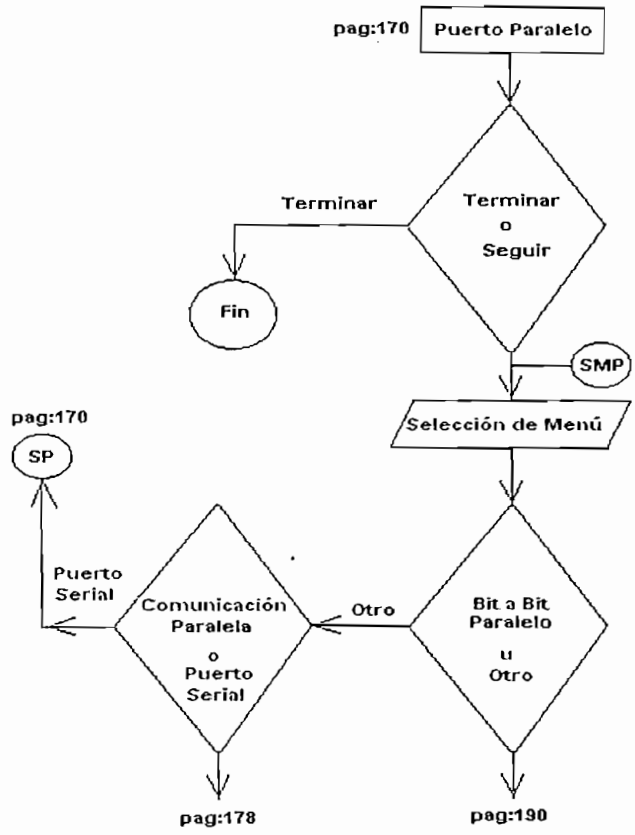
- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Transmisión Paralela.
- Tecla *Abajo*: saltar a Mensaje Prueba Pin a Pin Paralela.
- Tecla *Enter* : saltar a Recepción Paralela.
- Tecla *Menu*: saltar a Mensaje Comunicación Paralela.

Mensaje Prueba Pin a Pin Paralela

- Presentar mensaje de la opción 3 del Menú Paralelo.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Recepción Paralela.
- Tecla *Abajo*: saltar a Mensaje Transmisión Paralela.
- Tecla *Enter* : saltar a Prueba Pin a Pin Paralela.
- Tecla *Menu*: saltar a Mensaje Comunicación Paralela.

En el Computador:



Puerto Paralelo

- Carga el formulario Puerto Paralelo (Ver Figura 4.15).
- En el proceso de carga se ejecutan rutinas para determinar la existencia y tipo de puertos.
- De existir puertos, selecciona el primer puerto encontrado para su utilización y muestra la dirección base y el tipo de puertos encontrados.
- Si existen varios puertos se puede elegir el puerto paralelo a usar.
- En esta pantalla esta disponible la Prueba Bit a Bit del puerto y se muestran los menús para Comunicación y Puerto Serial.

Fin

- Termina la ejecución del programa.

Selección de Menú

- Escoge entre las alternativas para las pruebas del puerto.
- Pruebas Bit a Bit, Comunicación o ir al formulario del Puerto Serial.

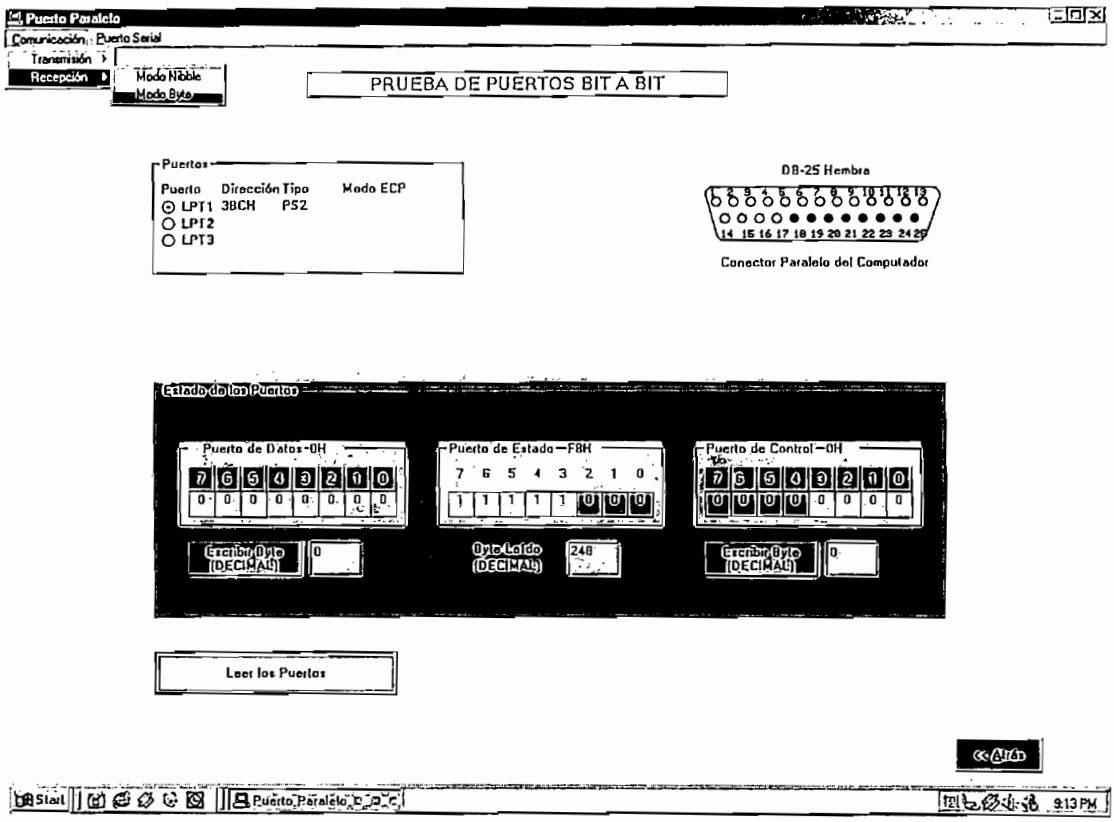
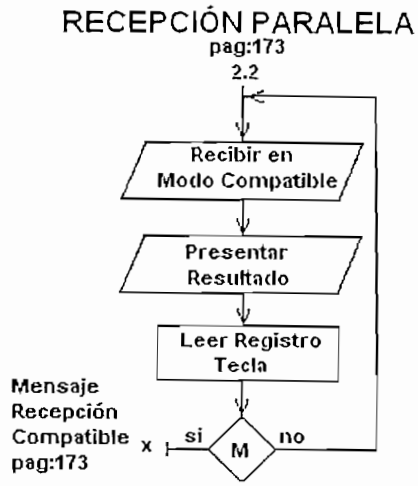


Figura 4.15

4.4.4.1 Transmisión Paralela Computador – Módulo

En el Módulo:



Recepción Paralela

Recibir en Modo Compatible

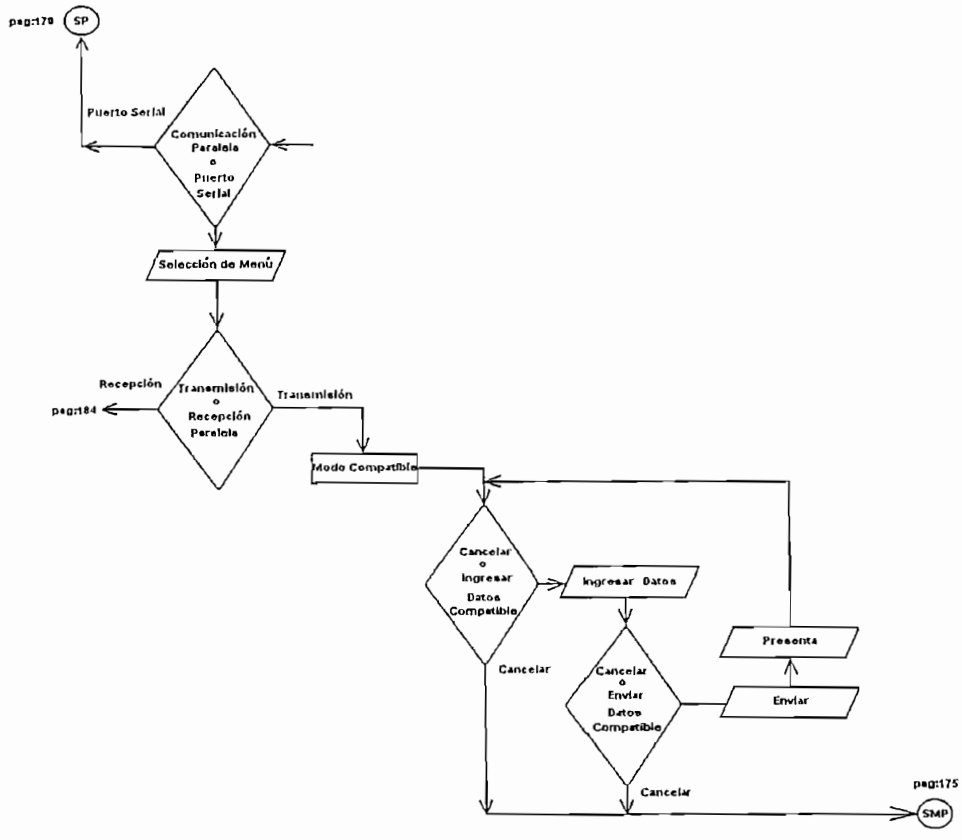
- Recibe hasta 10 caracteres en modo Compatible y los almacena en 10 localidades de memoria designadas para este fin.

Presentar Resultado

- Si se cumple el handshake, presentar un mensaje de recepción exitosa junto con los caracteres recibidos.

- Revisar que tecla se presiona.
- Tecla *Menu*: saltar a Mensaje Recepción Paralela, de lo contrario saltar a Recibir en Modo Compatible para realizar una nueva recepción.

En el Computador:



Selección de Menú

- Escoge entre las alternativas para las pruebas de comunicación del puerto.
- Transmisión o Recepción (Ver Figura 4.16).

Modo Compatible

- Carga el formulario "Modo Compatible" y muestra en la pantalla los controles para la prueba de comunicación (Ver Figura 4.17).
- Inhabilita el formulario Puerto Paralelo.
- Inicializa los valores por defecto de las señales del puerto para el handshake.

Ingresar Datos

- Recepta hasta 10 caracteres alfanuméricos ingresados por el usuario, en un cuadro de texto.

Enviar

- Envía hasta 10 caracteres en secuencia, utilizando en protocolo de Modo Compatible.
- El handshake y escritura de datos se muestran en una barra de tareas dentro del formulario.

Presenta

- Muestra una pantalla de mensaje que indica el resultado de la transmisión (Ver Figura 4.18).
- Una barra de estado en la parte inferior, muestra el proceso de las líneas de handshake, así como la dirección del puerto paralelo utilizado y su tipo (Ver Figura 4.19).

SMP

- Habilita el formulario Puerto Paralelo
- Regresa a la pantalla Puerto Paralelo.

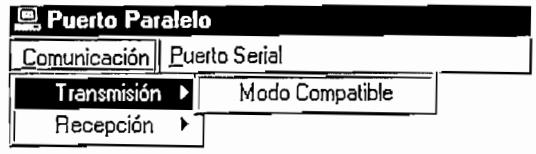


Figura 4.16

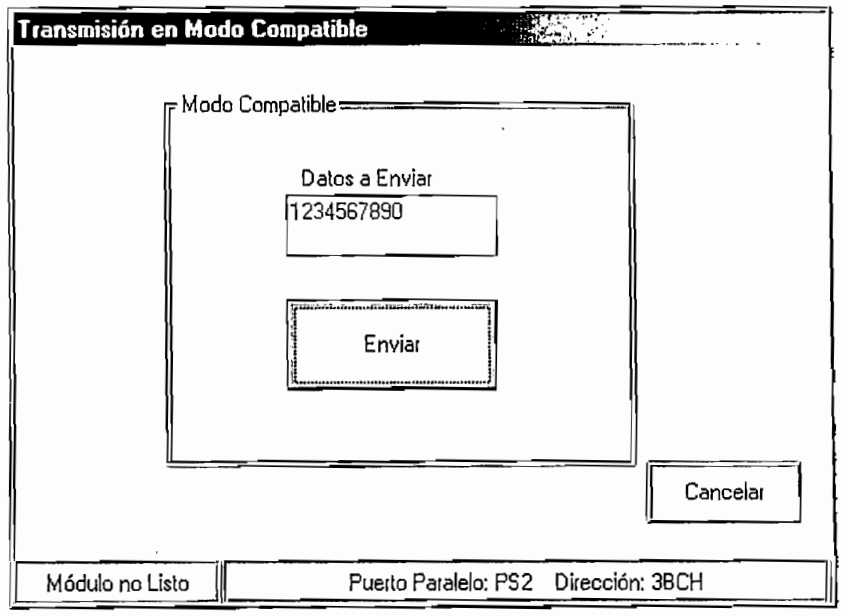


Figura 4.17

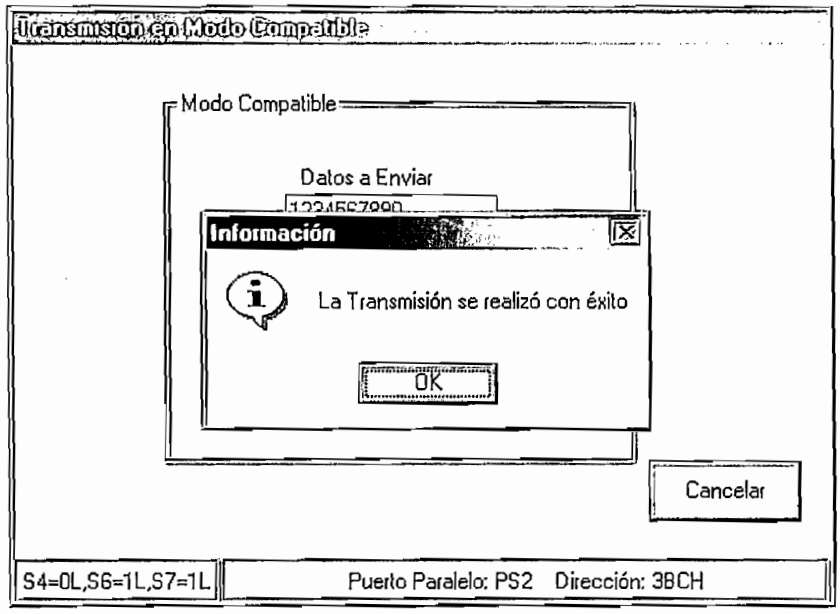


Figura 4.18

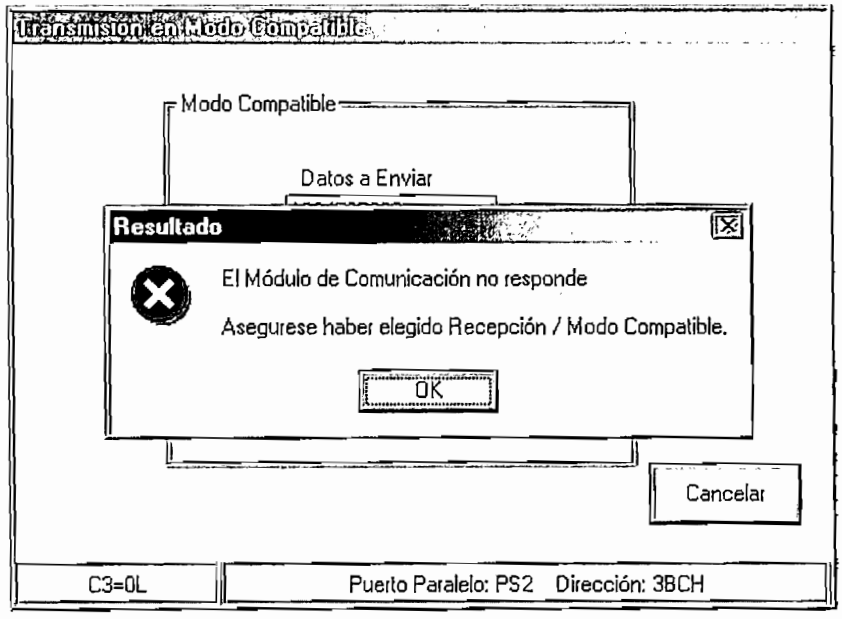
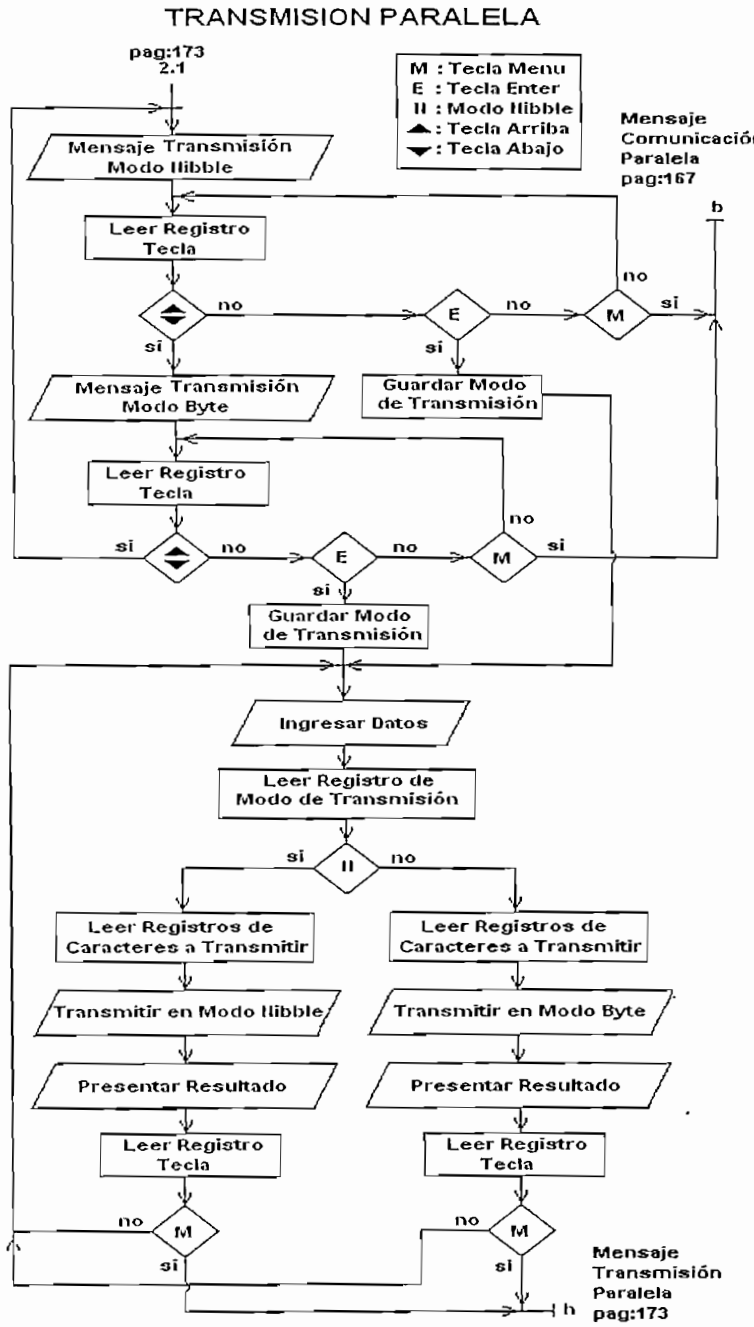


Figura 4.19

4.4.4.2 Recepción Paralela Computador - Módulo

La recepción de datos en el computador vía puerto paralelo se la realiza utilizando dos modos: Nibble y Byte.

En el Módulo:


TRANSMISION PARALELA
pag:173
2.1
M : Tecla Menu
E : Tecla Enter
II : Modo Hibble
▲ : Tecla Arriba
▼ : Tecla Abajo
Mensaje Comunicación Paralela pag:167
Mensaje Transmisión Modo Hibble
Leer Registro Tecla
b
no
si
Mensaje Transmisión Modo Byte
Guardar Modo de Transmisión
Ingresar Datos
Leer Registro de Modo de Transmisión
Leer Registros de Caracteres a Transmitir
Transmitir en Modo Hibble
Transmitir en Modo Byte
Presentar Resultado
h
Mensaje Transmisión Paralela pag:173

Menú Transmisión Paralela

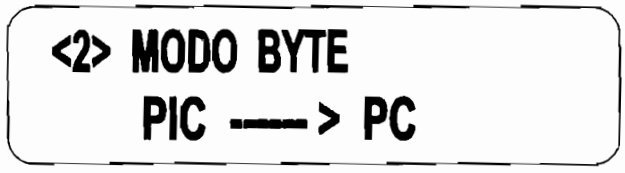
Mensaje Transmisión Modo Nibble

- Presentar mensaje de la opción 1 del Menú Transmisión Paralela.

- Revisar que tecla se presiona.
- Tecla *Arriba* o *Abajo*: saltar a Mensaje Transmisión Byte.
- Tecla *Enter*: guardar modo de transmisión paralela y saltar a Ingresar Datos.
- Tecla *Menu*: saltar a Mensaje Transmisión Paralela.

Mensaje Transmisión Modo Byte

- Presentar mensaje de la opción 2 del Menú Transmisión Paralela.



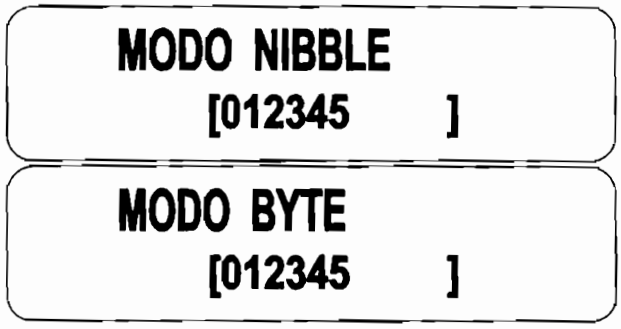
- Revisar que tecla se presiona.
- Tecla *Arriba* o *Abajo*: saltar a Mensaje Transmisión Nibble.
- Tecla *Enter*: guardar modo de transmisión paralela y saltar a Ingresar Datos.
- Tecla *Menu*: saltar a Mensaje Transmisión Paralela.

Ingresar Datos

- Almacena 10 caracteres ingresados a través del teclado en 10 localidades de memoria designadas para este fin.

Transmitir en Modo Nibble o en Modo Byte

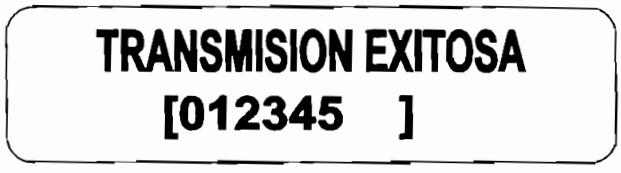
- Leer los registros que almacenan los caracteres a transmitir.
- Transmitir hasta 10 caracteres almacenados en las localidades de memoria designadas para este fin.



- Configurar el puerto D que maneja las señales del puerto paralelo del PC como entrada o como salida de acuerdo al handshake.

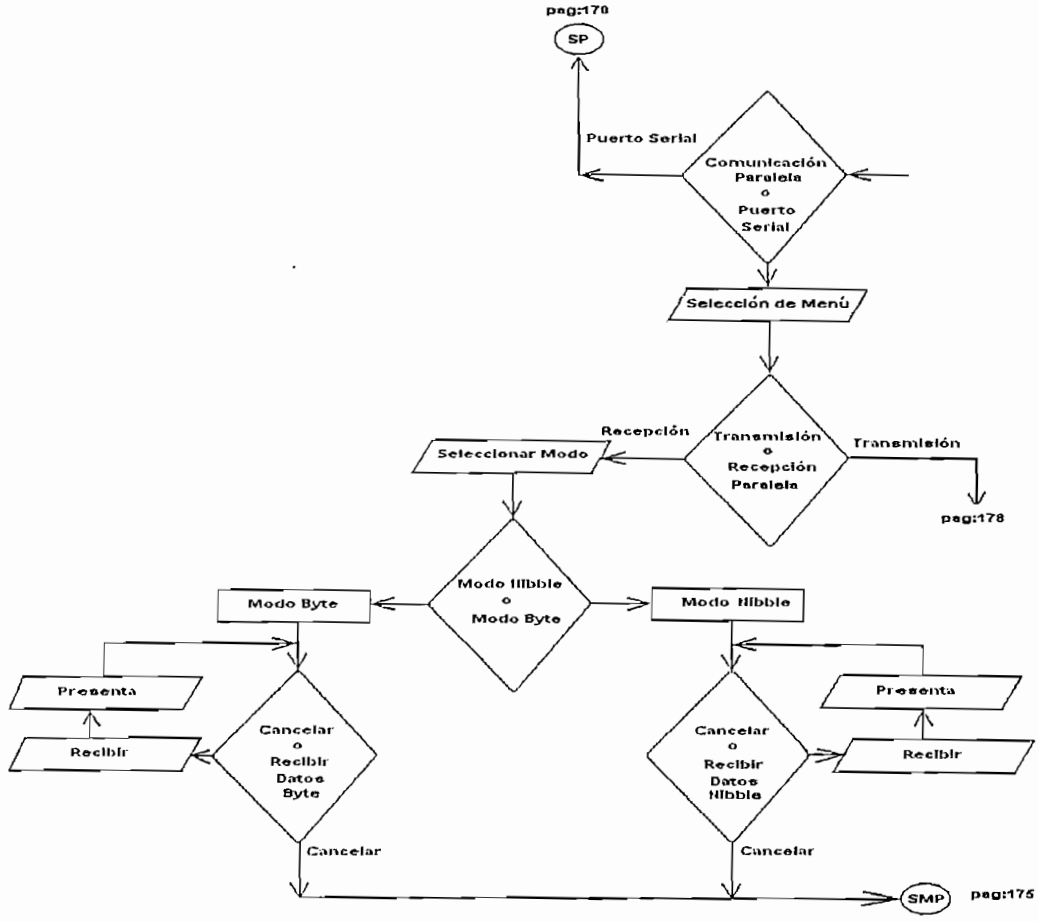
Presentar Resultados

- Si se cumple el handshake, presentar un mensaje de Transmisión exitosa junto con los caracteres Transmitidos.



- Revisar que tecla se presiona.
- Tecla *Menu*: saltar a Mensaje Transmisión Paralela, de lo contrario saltar a Ingresar Datos para realizar una nueva Transmisión

En el Computador:



Selección de Menú

- Escoge entre las alternativas para las pruebas de comunicación del puerto.
- Transmisión o Recepción (Ver Figura 4.20).

Seleccionar Modo

- Elige uno de los Modos de Recepción disponibles. Modo Nibble o Modo Byte.

Modo Nibble

- Carga el formulario Modo Nibble y muestra en la pantalla los controles para la prueba de comunicación (Ver Figura 4.21).
- Inhabilita el formulario Puerto Paralelo.
- Determina si el módulo esta listo o no.

- Si esta listo, inicializa los valores por defecto de las señales del puerto para el handshake.

Recibir

- Recibe hasta 10 caracteres en secuencia, utilizando en protocolo de Modo Nibble.
- El handshake y recepción de datos se muestran en una barra de tareas dentro del formulario.

Presenta

- Muestra una pantalla de mensaje que indica el resultado de la recepción (Ver Figura 4.22 y 4.23).
- Una barra de estado en la parte inferior, muestra el proceso de las líneas de handshake, así como la dirección del puerto paralelo utilizado y su tipo.

Modo Byte

- Carga el formulario Modo Byte y muestra en la pantalla los controles para la prueba de comunicación (Ver Figura 4.24).
- Inhabilita el formulario Puerto Paralelo.
- Determina si el módulo esta listo o no.
- Si esta listo Inicializa los valores por defecto de las señales del puerto para el handshake.

Recibir

- Recibe hasta 10 caracteres alfanuméricos en secuencia utilizando en protocolo de Modo Byte.
- El handshake y recepción de datos se muestran en una barra de tareas dentro del formulario.

Presenta

- Muestra una pantalla de mensaje que indica el resultado de la recepción (Ver Figuras 4.25 y 4.26).
- Una barra de estado en la parte inferior, muestra el proceso de las líneas de handshake, así como la dirección del puerto paralelo utilizado y su tipo.

SMP

- Habilita el formulario Puerto Paralelo
- Regresa a la pantalla Puerto Paralelo.

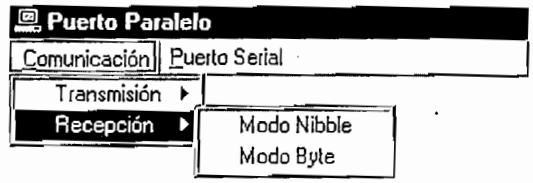


Figura 4.20

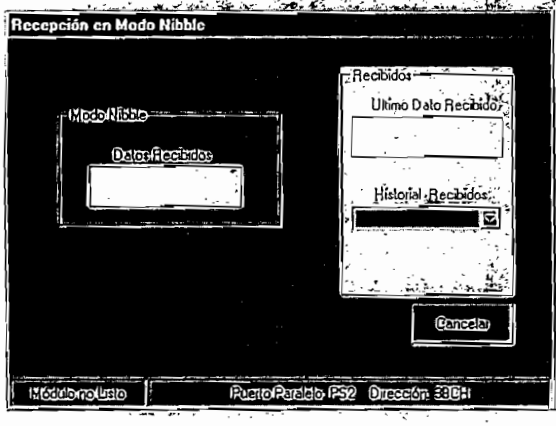


Figura 4.21

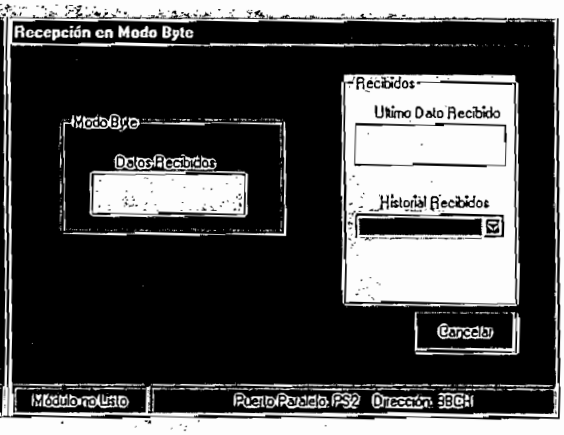


Figura 4.24

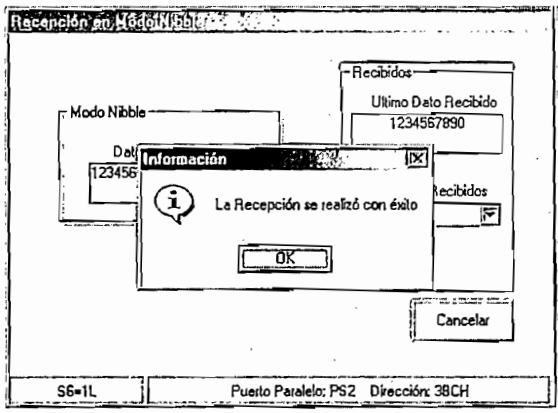


Figura 4.22

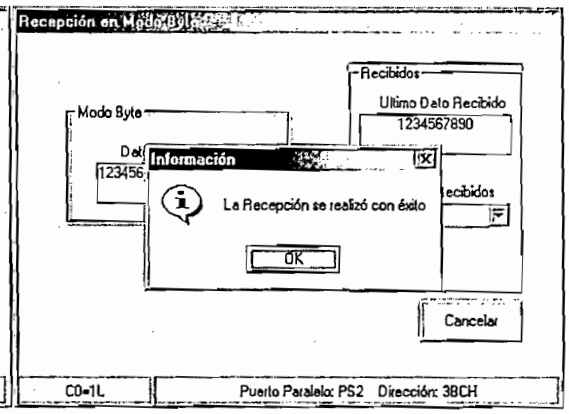


Figura 4.25

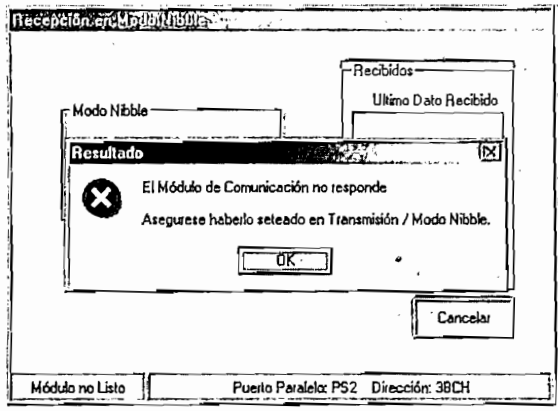


Figura 4.23

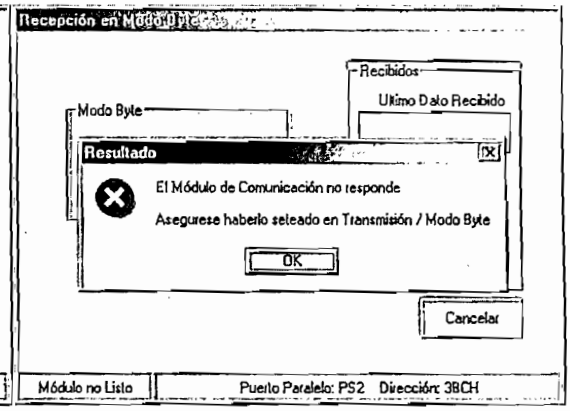
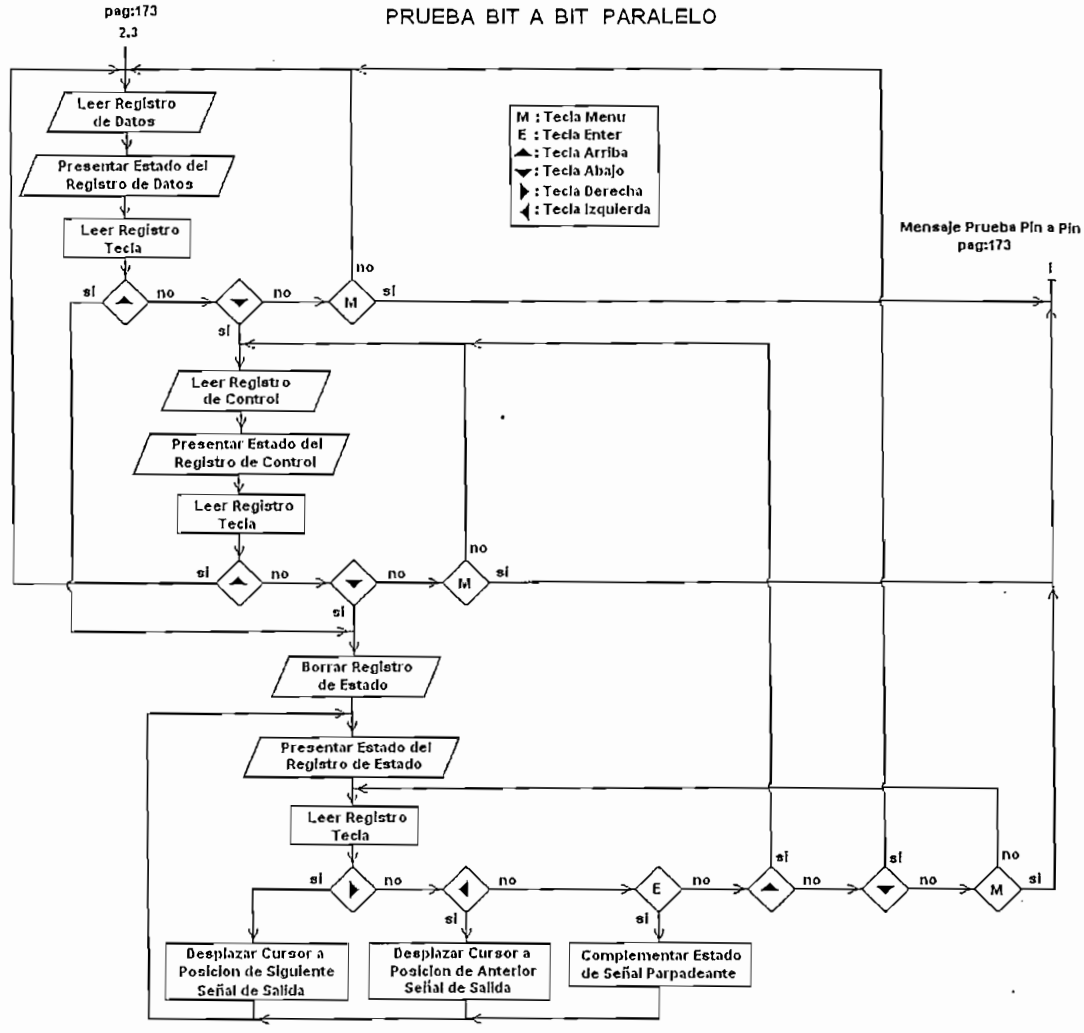


Figura4.26

4.4.4.3 Prueba Bit-a-Bit del Puerto Paralelo

En el Módulo:



Prueba Pin a Pin Paralela

Leer Registro de Datos

- Configurar como entrada el puerto D del PIC y leer el registro de Datos del puerto Paralelo del PC.

Presentar Estado del Registro de Datos

- Presentar el estado leído en el puerto de cada bit del registro de Datos en Binario y en Decimal en el LCD.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Borrar Registro de Estado.
- Tecla *Abajo*: saltar Leer Registro de Control.
- Tecla *Menu*: saltar a Mensaje Prueba Pin a Pin Serial.
- Regresar a Leer Registro de Datos

Leer Registro de Control

- Configurar como entrada el puerto D del PIC y leer el registro Control del puerto Paralelo del PC.

Presentar Estado del Registro de Control

- Presentar el estado leído en el puerto de cada bit del registro de Control en Binario y en Decimal en el LCD.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Leer Registro de Datos.
- Tecla *Abajo*: saltar Borrar Registro de Estado.
- Tecla *Menu*: saltar a Mensaje Prueba Pin a Pin Serial.
- Regresar a Leer el Registro de Control.

.Borrar Registro de Estado

- Configurar el puerto D como salida y llevar a 0L los bits que corresponden al puerto de Estado del Puerto del PC.

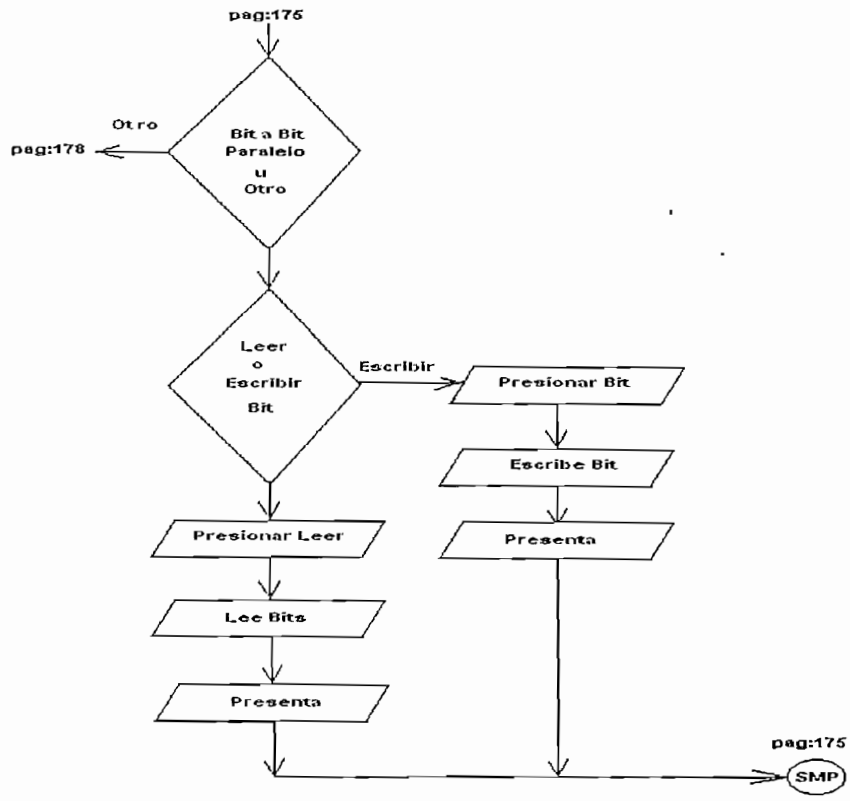
Presentar Estado del registro de Estado

- Presentar el estado leído en el puerto de cada bit del registro de Estado en Binario y en Decimal en el LCD.

- Revisar que tecla se presiona.
- Tecla *Derecha*: desplazar cursor a siguiente señal de salida.
- Tecla *Izquierda*: desplazar cursor a anterior señal de salida
- Tecla *Enter*: complementar el estado de la señal parpadeante
- Tecla *Arriba*: saltar a Leer Registro de Control
- Tecla *Abajo*: saltar a Leer Registro de Datos
- Tecla *Menu*: saltar a Mensaje Prueba Pin a Pin Serial.
- Regresar a Presentar Estado del Registro de Estado.

El estado de cada una de las señales del puerto paralelo es mostrado por los leds indicadores dispuestos en la forma de un conector DB25.

En el Computador:



Presionar Bit

- Ingreso de información, escogiendo el bit a escribir y actuando sobre su control (Ver Figura 4.27).

Escribe Bit

- Escribe los bits de los registros de datos y/o control del puerto
- Complementa el estado lógico del bit determinado y lo escribe al puerto paralelo correspondiente.

Presenta

- Actualiza el estado de las señales leídas del conector y las presenta en pantalla (Ver Figura 4.28).
- Se puede también visualizar el estado de las señales del conector.

Presionar Leer

- Indica operación de lectura de los bits del puerto.

Lee Bits

Lee Bits

- Lee el valor de los bits de todos los registros del puerto.

Presenta

- Actualiza el estado de las señales leídas del conector y las presenta en pantalla (Ver Figura 4.28).
- Se puede también visualizar el estado de las señales del conector.

SMP

Regresa a la pantalla Puerto Paralelo.

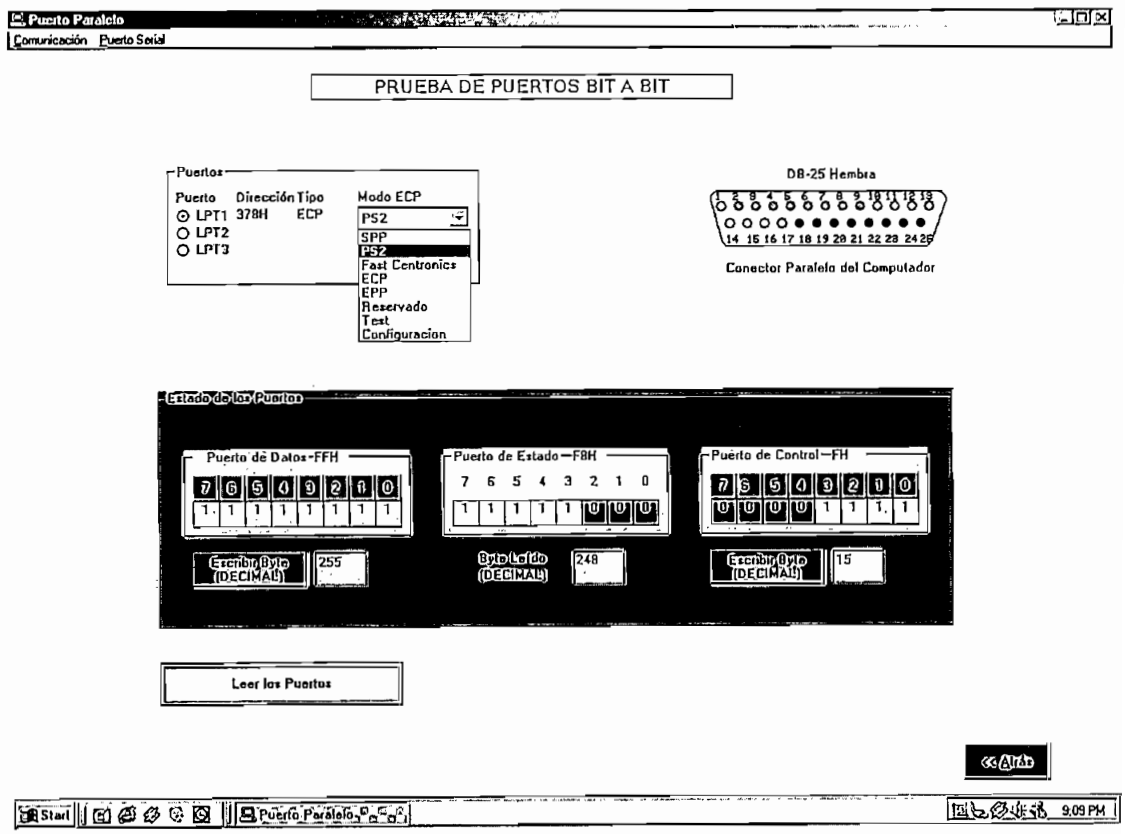


Figura 4.27

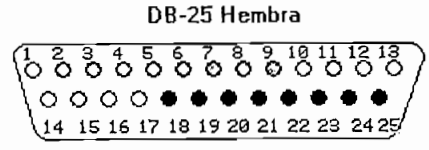
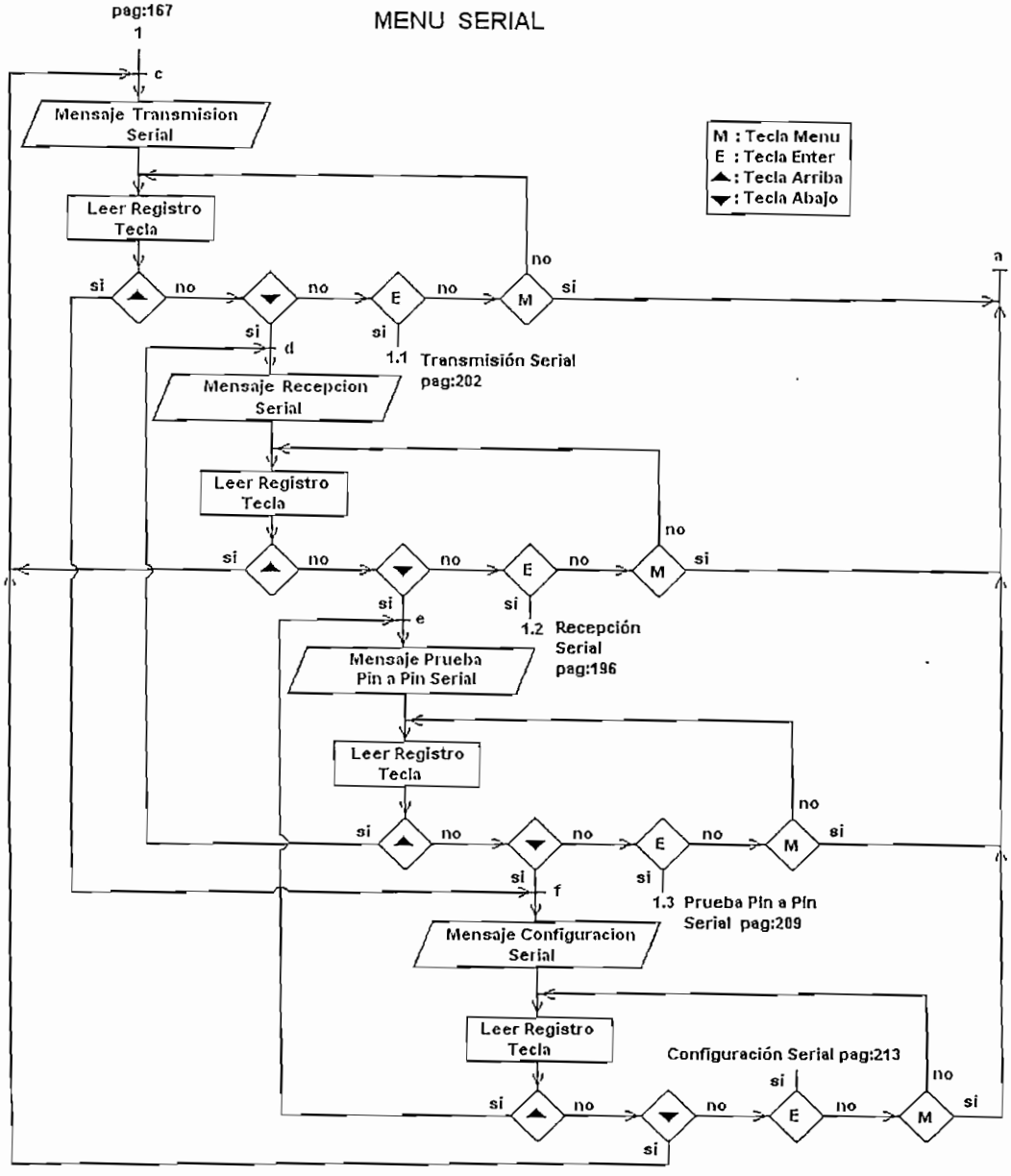


Figura 4.28

4.4.5 PUERTO SERIAL

En el Módulo:



Menú Serial

Mensaje Transmisión Serial

- Presentar mensaje de la opción 1 del Menú Serial.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Configuración Serial.
- Tecla *Abajo*: saltar a Mensaje Recepción Serial
- Tecla *Enter*: saltar a Transmisión Serial.
- Tecla *Menu*: saltar a Mensaje Comunicación Serial.

Mensaje Recepción Serial

- Presentar mensaje de la opción 2 del Menú Serial.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Transmisión Serial.
- Tecla *Abajo*: saltar a Mensaje Prueba Pin a Pin Serial.
- Tecla *Enter*: saltar a Recepción Serial.
- Tecla *Menu*: saltar a Mensaje Comunicación Serial.

Mensaje Prueba Pin a Pin Serial

- Presentar mensaje de la opción 3 del Menú Serial.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Recepción Serial.
- Tecla *Abajo*: saltar a Mensaje Configuración Serial.

- Tecla *Enter*: saltar a Prueba Pin a Pin Serial.
- Tecla *Menu*: saltar a Mensaje Comunicación Serial.

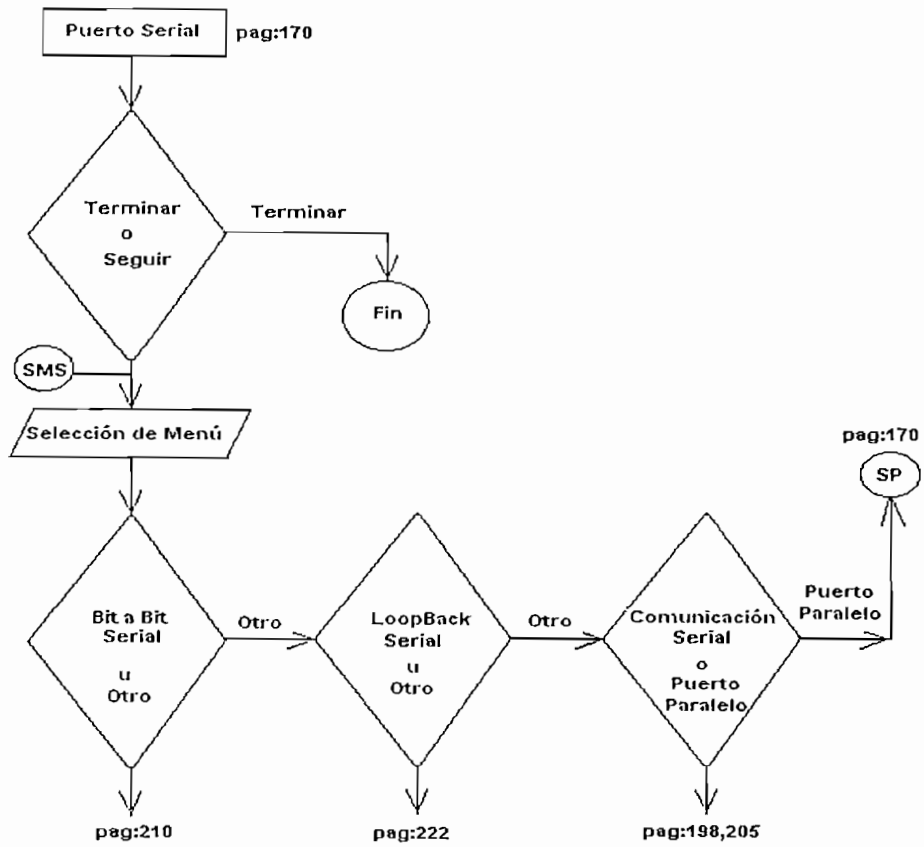
Mensaje Configuración Serial

- Presentar mensaje de la opción 4 del Menú Serial.

<4> CONFIGURAR EL PUERTO SERIAL

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Prueba Pin a Pin Serial.
- Tecla *Abajo*: saltar a Mensaje Transmisión Serial.
- Tecla *Enter*: saltar a Menú Configuración Serial.

En el Computador:



Puerto Serial

- Carga el formulario Puerto Serial (Ver Figura 4.29).
- En el proceso de carga se ejecutan rutinas para determinar la existencia de puertos.
- De existir, selecciona y abre el primer puerto encontrado para su utilización y muestra los puertos de comunicaciones encontrados.
- De existir varios puertos, se puede elegir el puerto serial a usar.
- En esta pantalla esta disponible la Prueba Bit a Bit del puerto y se muestran los menús de LoopBack, Comunicación y Puerto Paralelo.

Fin

- Termina la ejecución del programa.

Selección de Menú

- Escoge entre las alternativas para las pruebas del puerto.
- Pruebas Bit a Bit, Prueba de LoopBack, Comunicación o ir al Puerto Paralelo.

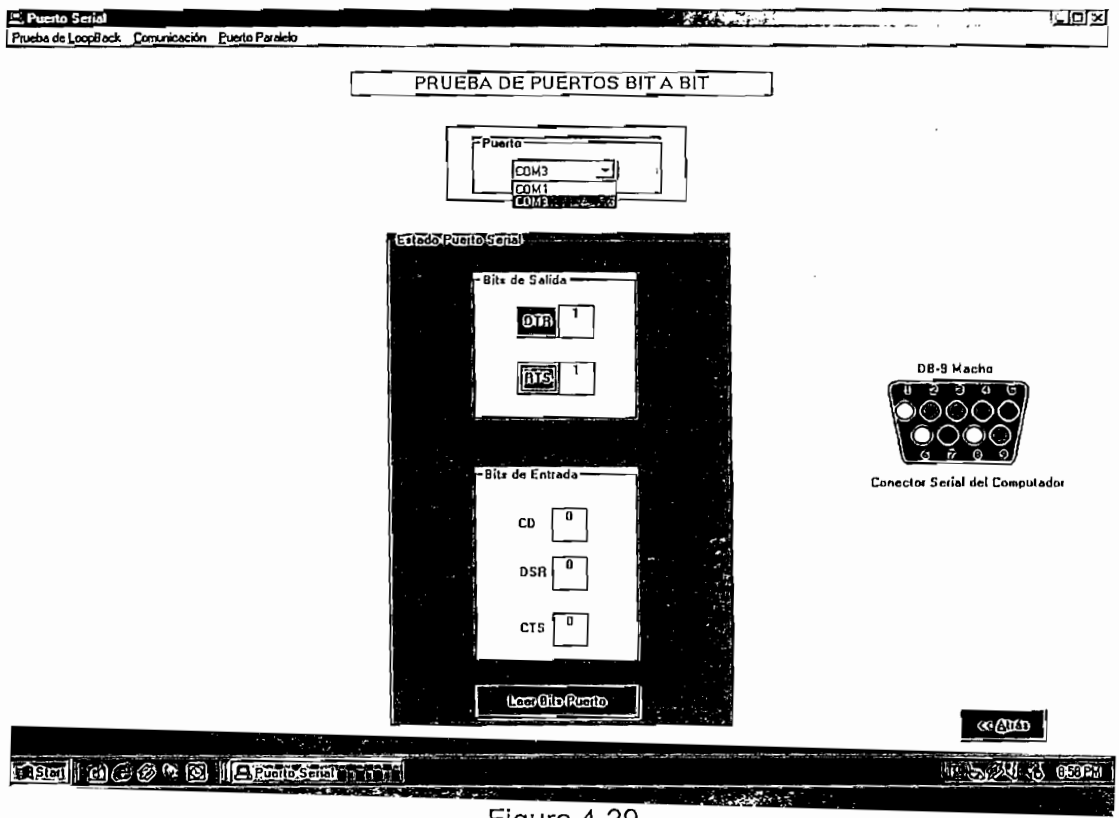
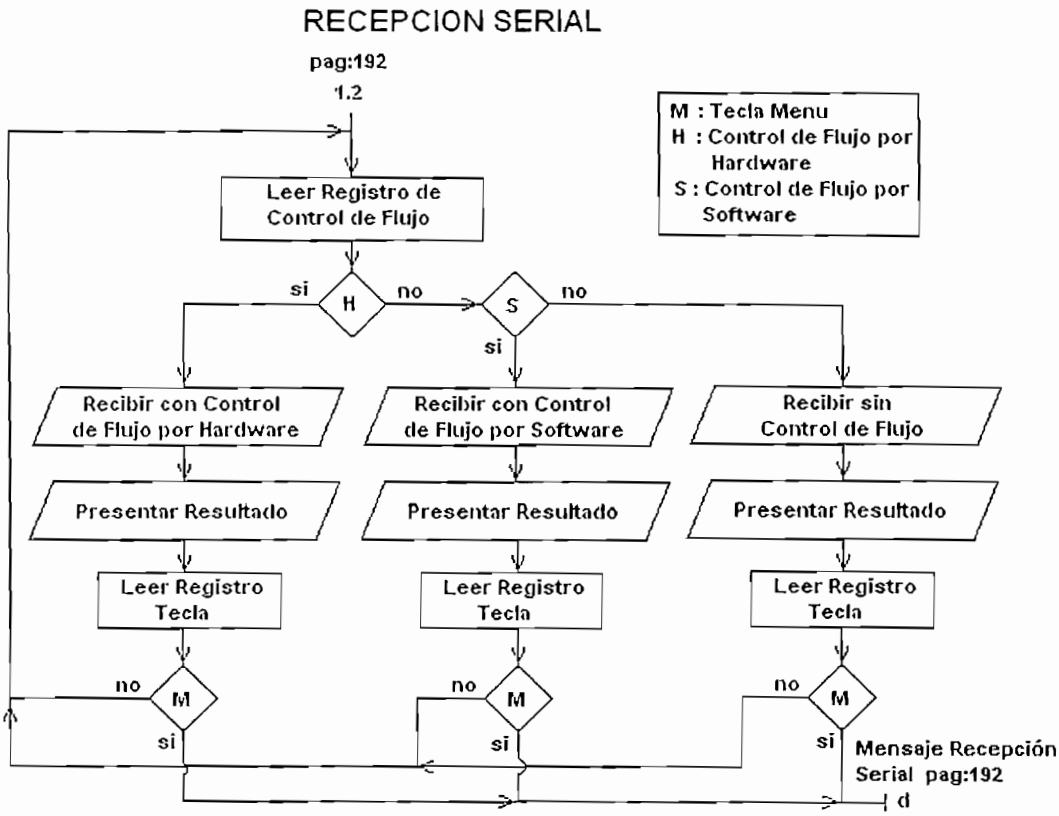


Figura 4.29

4.4.5.1 Transmisión Serial Computador – Módulo

En el Módulo:

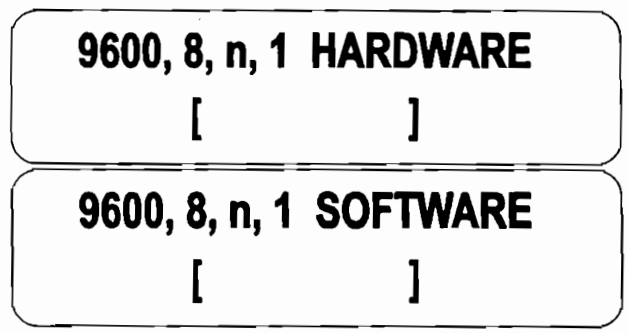


Recepción Serial

- Leer registro donde se guarda el control de flujo.
- De acuerdo a este valor, saltar a Recibir con Control de Flujo por Hardware, Software o sin Control de Flujo.

Recepción con Control de Flujo por Hardware y por Software

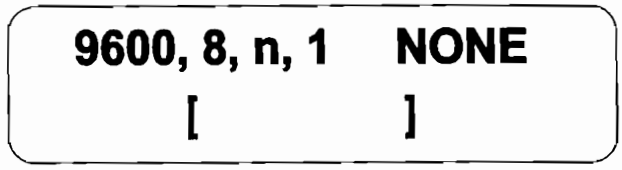
- Presentar mensaje de recepción serial con el control de flujo determinado.



- Configurar el puerto serial y cargar el baudrate en el registro de configuración de baudrate.
- Recibir los caracteres usando el Control de Flujo configurado, y almacenarlos en localidades de memoria designadas para este fin.

Recepción Sin Control de Flujo

- Presentar mensaje de recepción serial sin control de flujo.



- Configurar el puerto serial y cargar el baudrate en el registro de configuración de baudrate.
- Recibir los caracteres sin usar Control de Flujo, y almacenarlos en localidades de memoria designadas para este fin.

Presentar Resultado

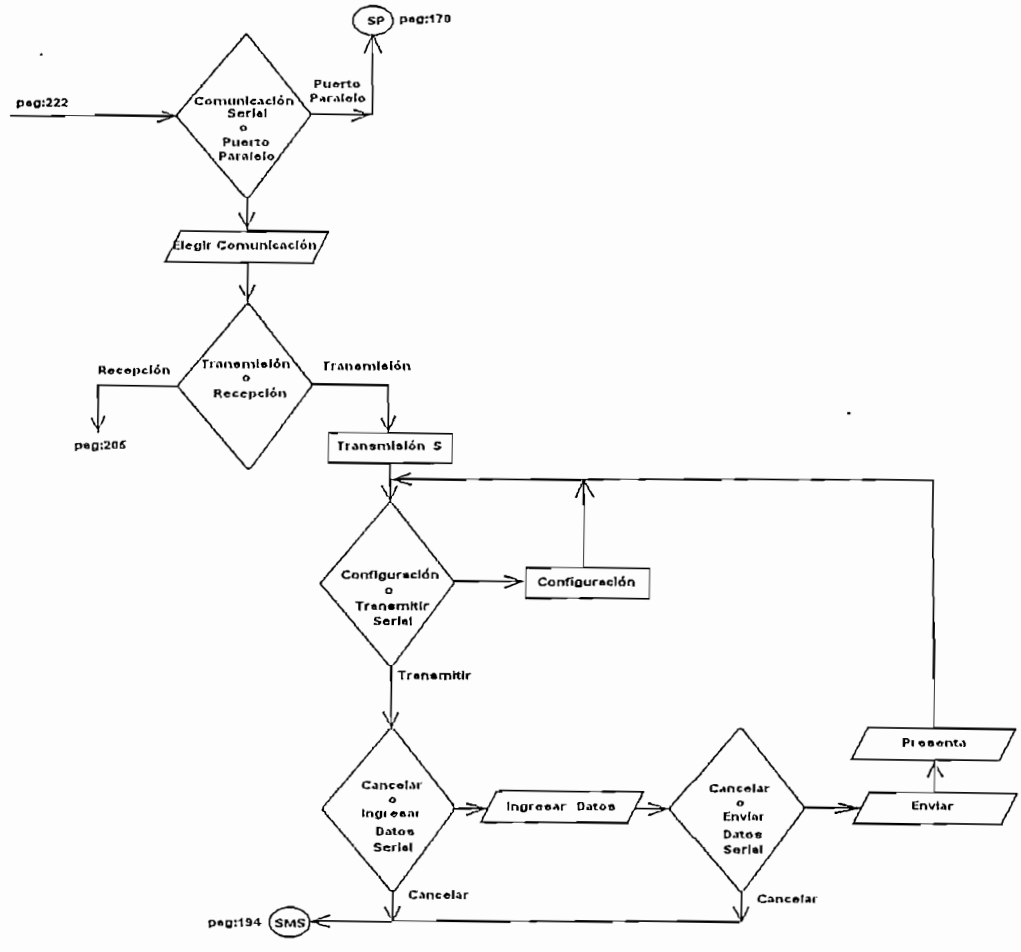
- Si se cumple el handshake, presentar mensaje de recepción exitosa junto con los caracteres recibidos y saltar a Leer Registro de Control de Flujo para realizar una nueva recepción.

- Si no cumple el handshake, presentar un mensaje de error de handshake.

- Si se transmite sin control de flujo, presentar el mensaje de transmisión exitosa junto con los caracteres recibidos y saltar a Leer Registro de Control de Flujo para realizar una nueva transmisión.
- Revisar que tecla se presiona.
- Tecla *Menu*: saltar a Mensaje Transmisión Serial.

El Módulo de Comunicaciones acepta solamente datos en formato Texto , en codificación ACSII. Sin embargo, en el programa del computador se pueden elegir alternativamente los formatos Texto y Binario. Al trabajar con el formato binario, el computador envía los números hexadecimales directamente sin necesidad de convertir cada dígito a su correspondiente código ASCII.

En el Computador:



Elegir Comunicación

- Escoge entre las alternativas para las pruebas de comunicación del puerto.
- Transmisión o Recepción (Ver Figura 4.30).

Transmisión S

- Carga el formulario Transmisión Serial y muestra en la pantalla los controles para la prueba de comunicación.

- Inhabilita el formulario Puerto Serial.
- Inicializa los valores por defecto de las señales del puerto para el handshake.
- Carga los valores del protocolo de comunicación (8,n,1), el baudrate, tipo de handshake y formato, según la última configuración ajustada.

Configuración

- Descarga el formulario Transmisión Serial.
- Carga el formulario Configuración del Puerto Serial.
- Carga los valores del protocolo de comunicación (8,n,1), el baud rate, tipo de handshake y formato, según la configuración seleccionada mediante los controles del formulario.
- Carga el formulario Transmisión Serial.

Ingresar Datos

- Recepta hasta 10 caracteres alfanuméricos ingresados por el usuario, en un cuadro de texto.

Enviar

- Envía hasta 10 caracteres en secuencia utilizando en protocolo 8,n,1; baudrate, handshake y formato fijados.
- El baudrate, handshake, formato, y escritura de datos se muestran en una barra de tareas dentro del formulario.

Presenta

- Muestra una pantalla de mensaje que indica el resultado de la transmisión (Ver Figuras 4.31, 4.32 y 4.33).
- Una barra de estado en la parte inferior, muestra el proceso de las líneas de handshake, el puerto COM por el cual se van a transmitir los datos, la velocidad de transmisión, el control de flujo y el formato de los datos a transmitir.

SMS

Habilita el formulario Puerto Serial.

Regresa a la pantalla Puerto Serial.

Figura 4.30

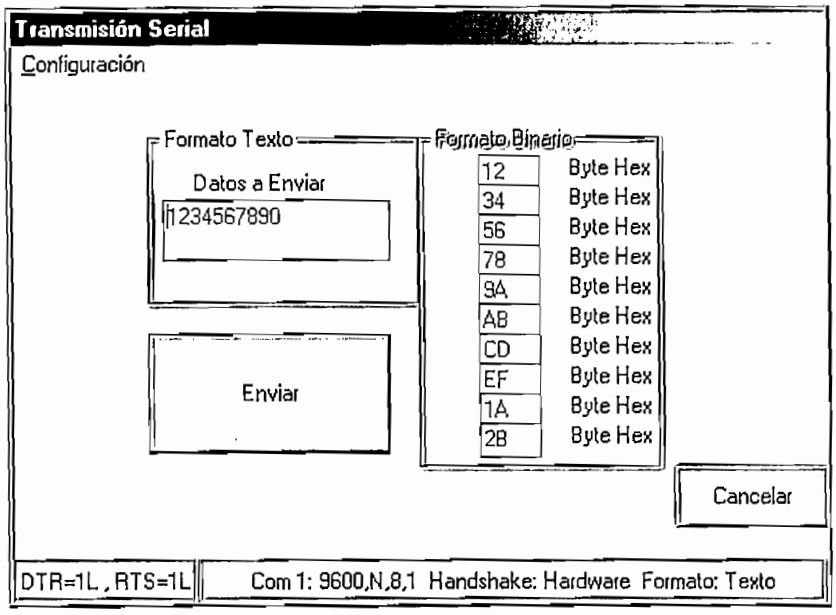


Figura 4.31

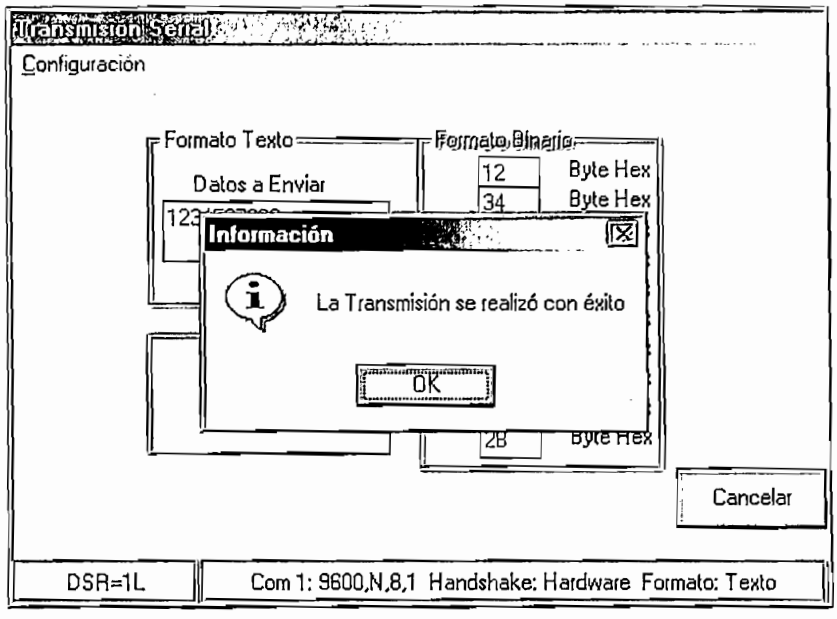


Figura 4.32

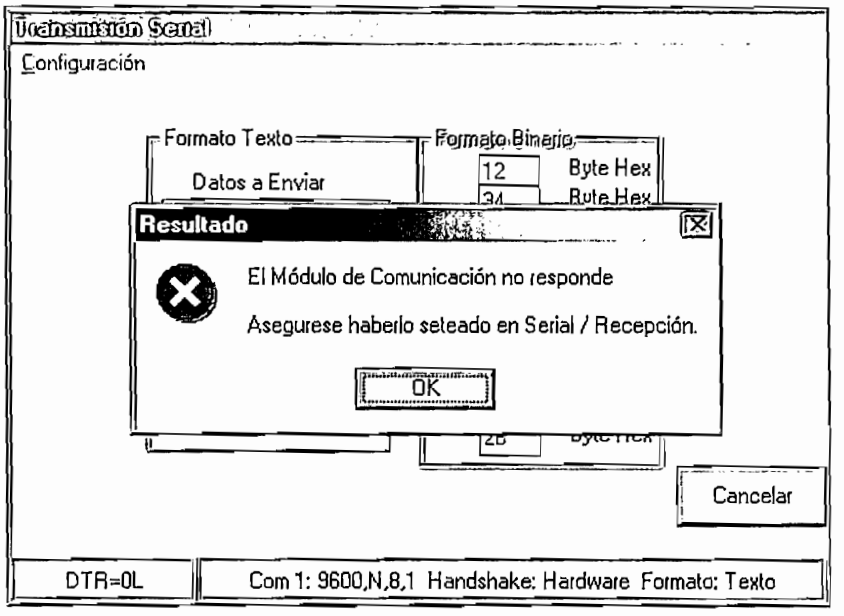
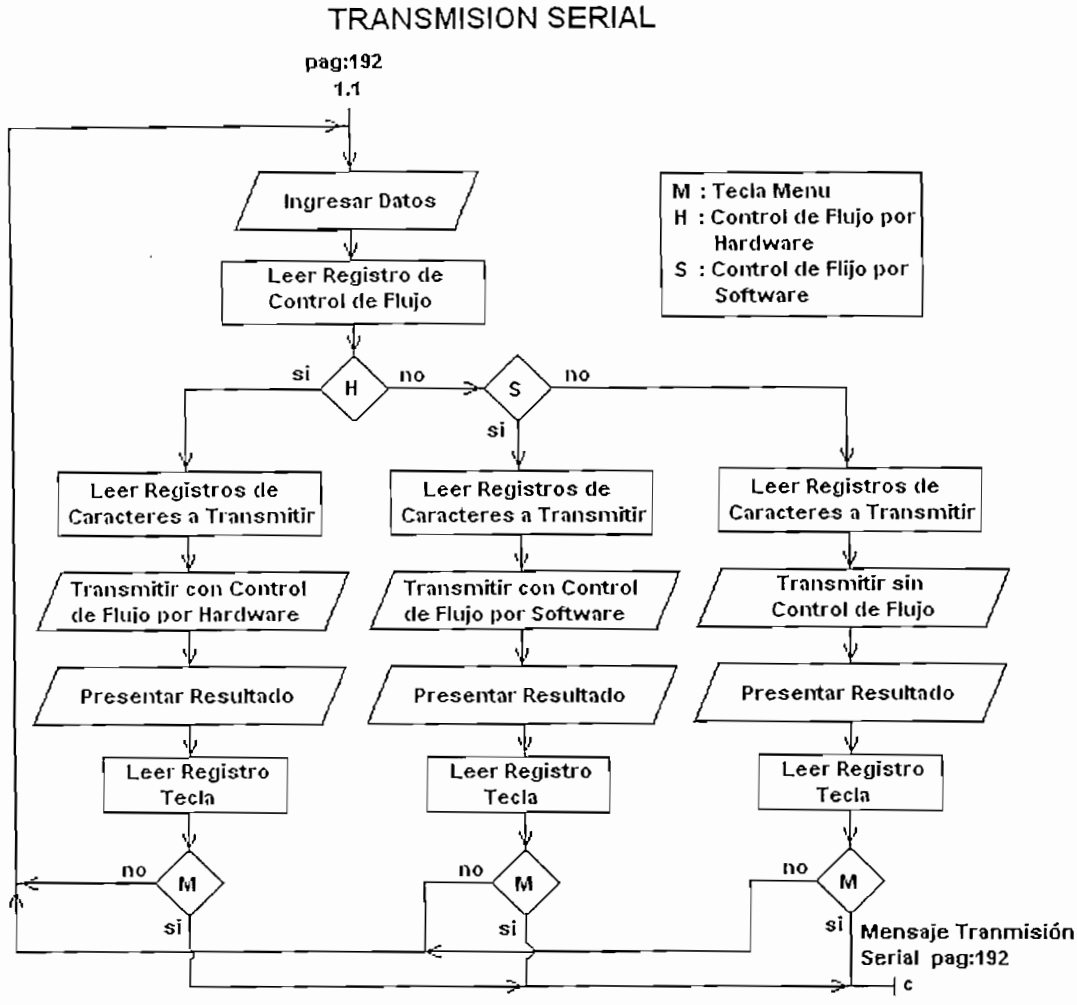

Transmisión Serial
Configuración
Formato Texto
Datos a Enviar
Formato Binario
12
Byte Hex
Resultado
El Módulo de Comunicación no responde
Asegurese haberlo seteado en Serial / Recepción.
OK
Cancelar
DTR=0L
Com 1: 9600,N,8,1 Handshake: Hardware Formato: Texto

Figura 4.33

4.4.5.2 Recepción Serial Computador - Módulo

En el Módulo:



Transmisión Serial

Ingresar Datos

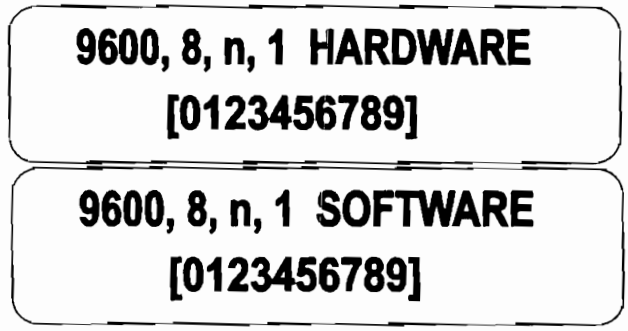
- Almacena 10 caracteres ingresados a través del teclado en 10 localidades de memoria designadas para este fin.

- Leer registro donde se guarda el control de flujo.

- De acuerdo a este valor, saltar a Transmitir con Control de Flujo por Hardware, Software o sin Control de Flujo.

Transmisión con Control de Flujo por Hardware y por Software

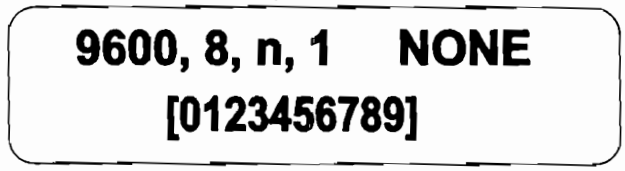
- Presentar mensaje de transmisión serial con el control de flujo determinado.



- Configurar el puerto serial y cargar el baudrate en el registro de configuración de baudrate.
- Transmitir los caracteres ingresados usando el Control de Flujo configurado.

Transmisión Sin Control de Flujo

- Presentar mensaje de transmisión serial sin Control de Flujo.



- Configurar el puerto serial y cargar el baudrate en el registro de configuración de baudrate.
- Transmitir los caracteres ingresados sin usar Control de Flujo.

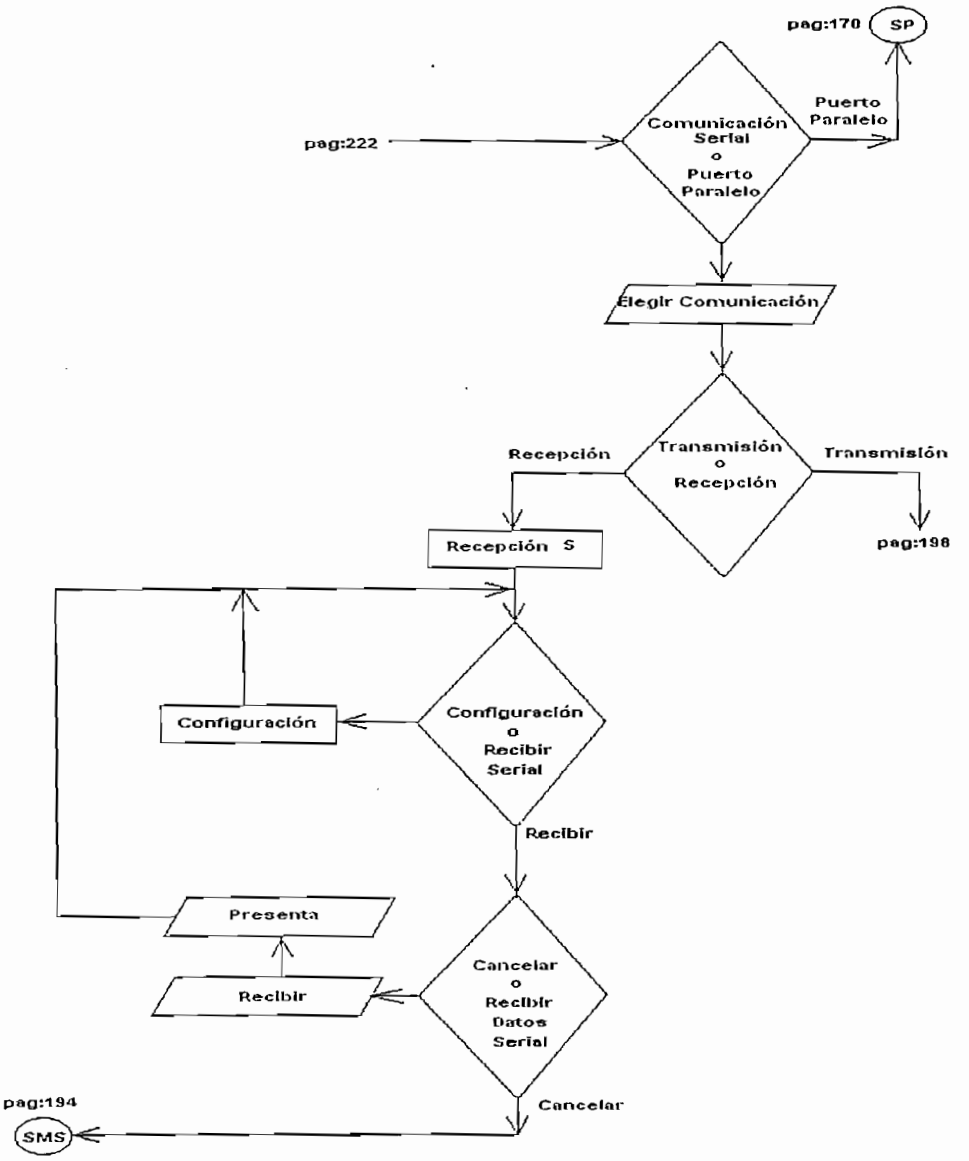
Presentar Resultado

- Si se cumple el handshake, presentar mensaje de transmisión exitosa junto a los caracteres transmitidos y saltar a Ingresar Datos para realizar una nueva transmisión.

- Si no cumple el handshake, presentar un mensaje de error de handshake.

- Si se transmite sin control de flujo, presentar el mensaje de transmisión exitosa junto con los caracteres transmitidos y saltar a Ingresar Datos para realizar una nueva transmisión.
- Revisar que tecla se presiona.
- Tecla *Menu*: saltar a Mensaje Transmisión Serial

En el Computador:



Elegir Comunicación

- Escoge entre las alternativas para las pruebas de comunicación del puerto.
- Transmisión o Recepción (Ver Figura 4.34).

Recepción S

- Carga el formulario Recepción Serial y muestra en la pantalla los controles para la prueba de comunicación.
- Inicializa los valores por defecto de las señales del puerto para el handshake.

- Carga los valores del protocolo de comunicación (8,n,1), el baudrate, tipo de handshake y formato, según la última configuración ajustada.
- Inhabilita el formulario Puerto Serial.

Configuración.

- Descarga el formulario Recepción Serial (Ver Figura 4.35).
- Carga el formulario Configuración del Puerto Serial.
- Carga los valores del protocolo de comunicación (8,n,1), el baud rate, tipo de handshake y formato, según la configuración seleccionada mediante los controles del formulario.
- Carga el formulario Recepción Serial.

Recibir

- Recibe hasta 10 caracteres en secuencia utilizando en protocolo 8,n,1; baudrate, handshake y formato fijados.
- El baudrate, handshake, formato, y recepción de datos se muestran en una barra de tareas dentro del formulario.
- Cada grupo de datos recibido es almacenado en un historial.

Presenta

- Muestra una pantalla de mensaje que indica el resultado de la recepción (Ver Figuras 4.36, 4.37 y 4.38).
- Una barra de estado en la parte inferior, muestra el proceso de las líneas de handshake, el puerto COM por el cual se van a transmitir los datos, la velocidad de transmisión, el control de flujo y el formato de los datos a transmitir.

SMS

- Habilita el formulario Puerto Serial.
- Regresa a la pantalla Puerto Serial.

Figura 4.34

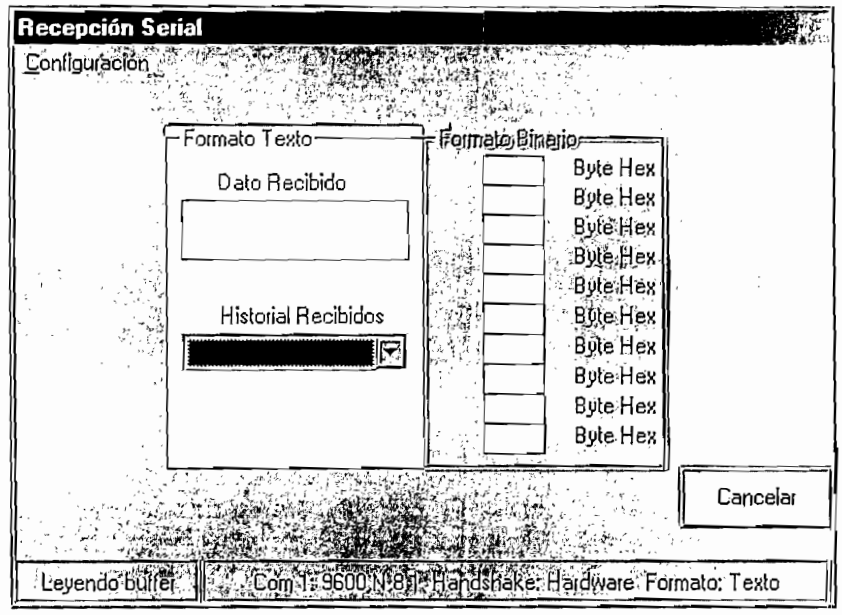


Figura 4.35

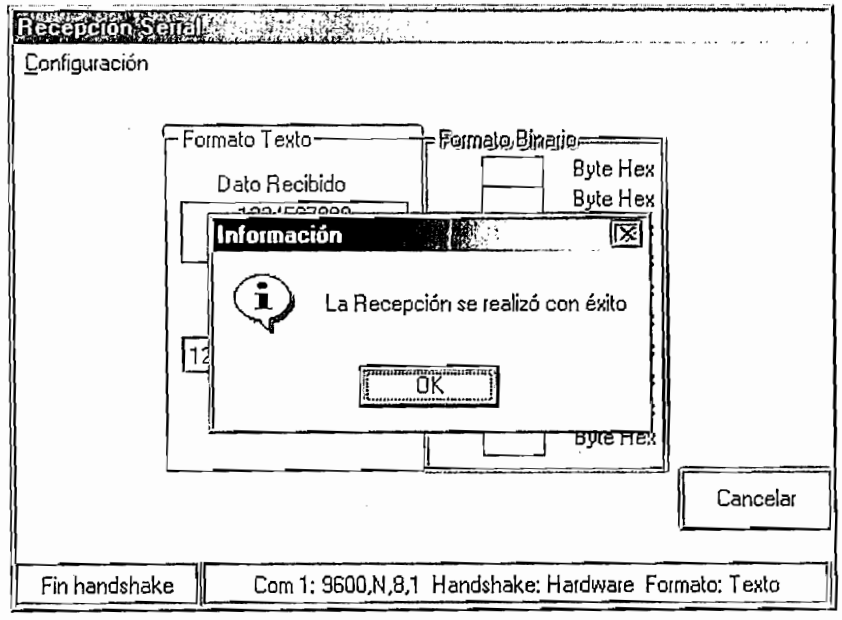


Figura 4.36

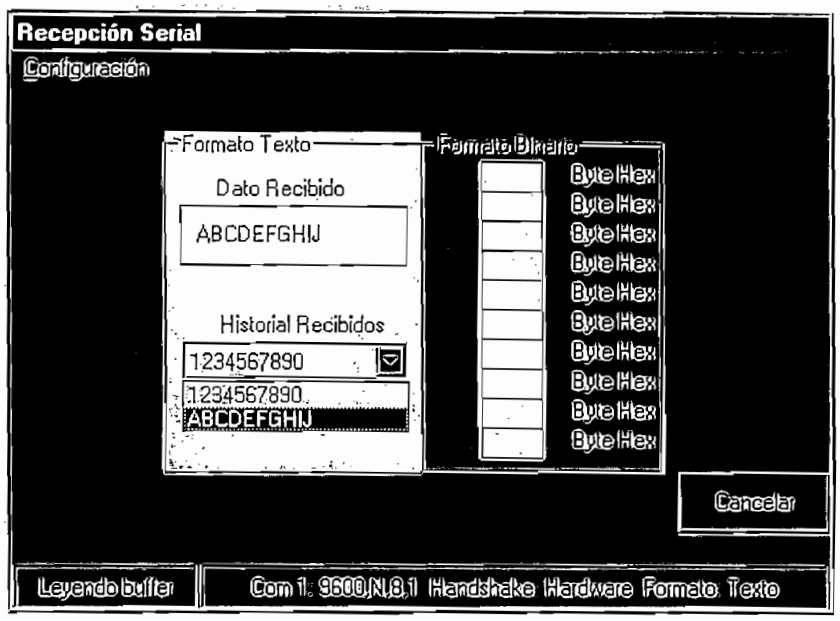
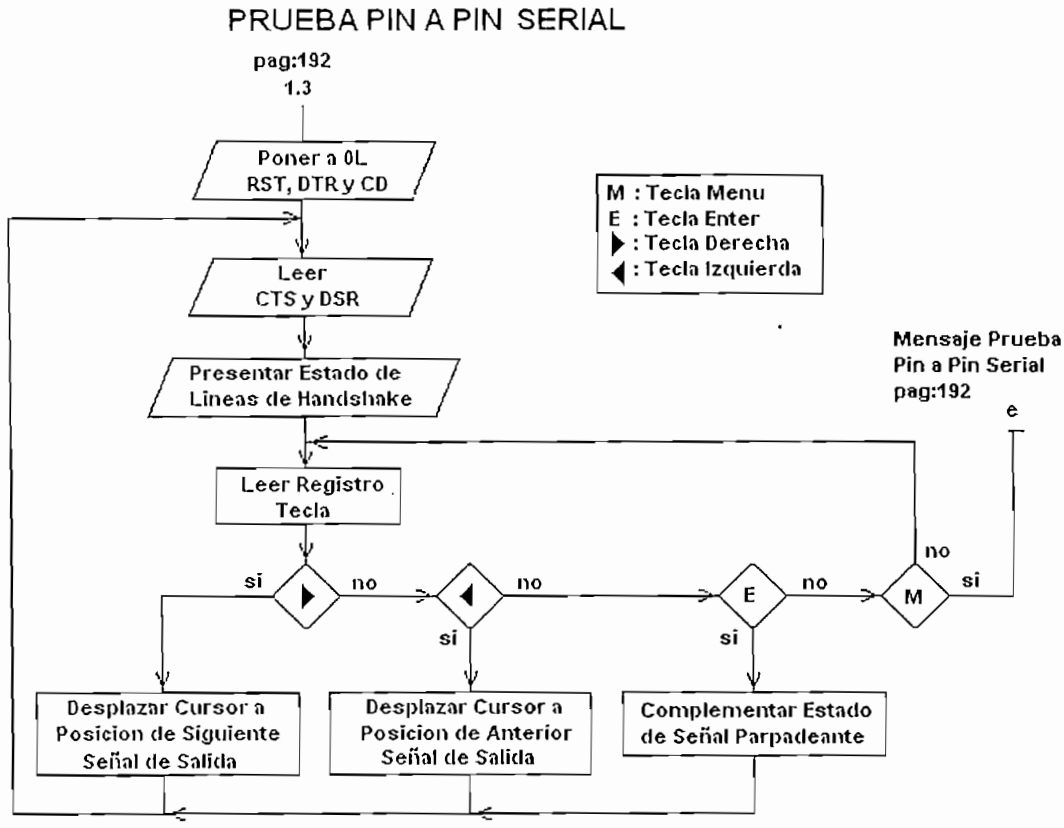


Figura 4.37

Figura 4.38

4.4.5.3 Prueba Pin a Pin del Puerto Serial

En el Módulo:



Prueba Pin a Pin Serial

- Borrar los pines del puerto C del microcontrolador que corresponden a RTS, DTR y CD del puerto serial del PC.
- Leer los pines del puerto C del PIC que corresponden a las señales CTS y DSR.

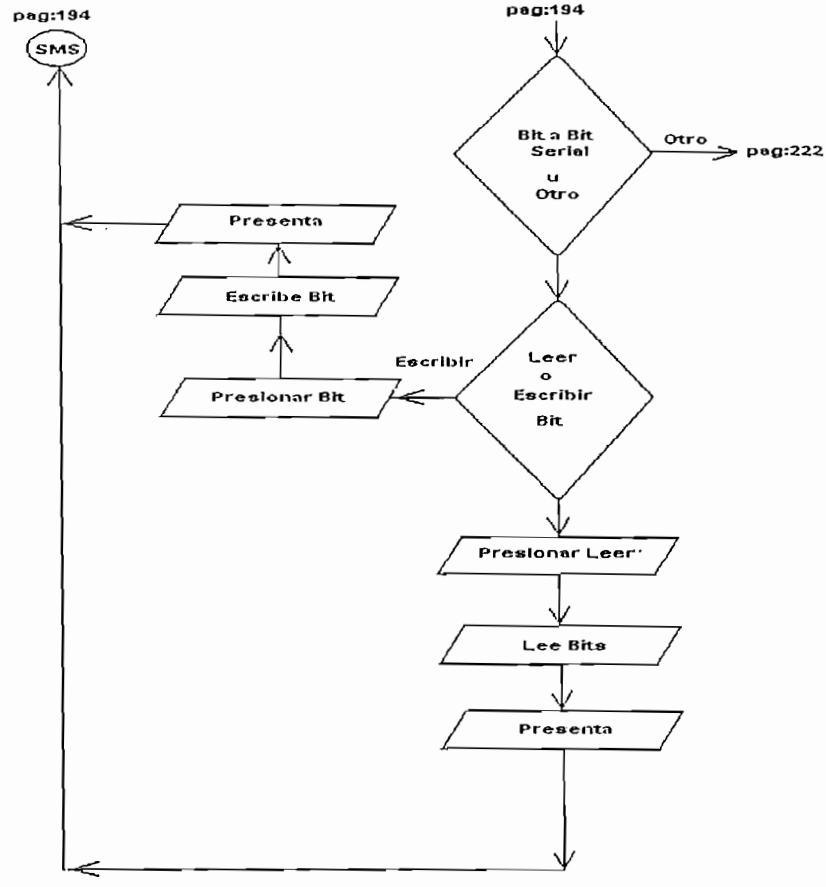
Presentar Estado de Líneas de Handshake.

- Presentar en el LCD el estado de los pines del puerto del microcontrolador que corresponden a las líneas de handshake.

- Leer que tecla se presiona.
- Tecla *Derecha*: desplazar cursor a la posición de la siguiente línea0 de salida, y regresa a leer las señales de entrada CTS y DSR.
- Tecla *Izquierda*: desplazar cursor a la posición de la anterior línea de salida, y regresa a leer las señales de entrada CTS y DSR.
- Tecla *Enter*: complementar el estado lógico de la señal parpadeante, y regresar a Leer CTS y DSR.
- Tecla *Menu*: saltar a Mensaje Prueba Pin a Pin Serial.

El estado de cada una de las señales del puerto serial es mostrado por los leds indicadores dispuestos en la forma de un conector DB9.

En el Computador:



Presionar Bit

- Ingreso de información, escogiendo el bit a escribir y actuando sobre su control (Ver Figura 4.39).

Escribe Bit

- Escribe las líneas DTR y RTS.
- Complementa el estado lógico del bit determinado y lo escribe al puerto serial correspondiente (Ver Figura 4.39).

Presionar Leer

- Indica operación de lectura de los bits del puerto.

Lee Bits

- Lee el estado de las señales de entrada del conector (CD, DSR y CTS) (Ver Figura 4.39)..

Presenta

- Actualiza el estado de las señales leídas del conector y las presenta en pantalla (Ver Figura 4.40).
- Se puede también visualizar el estado de las señales del conector.

SMS

- Regresa a la pantalla Puerto Serial.

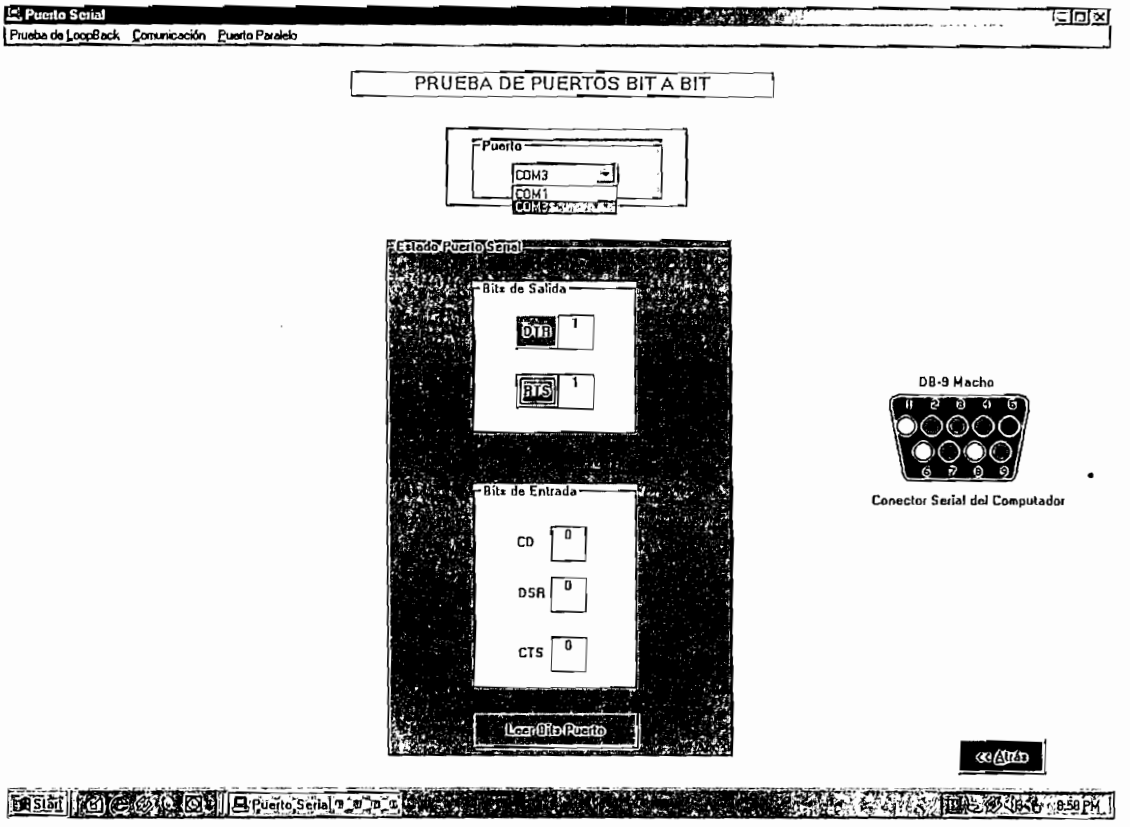


Figura 4.39

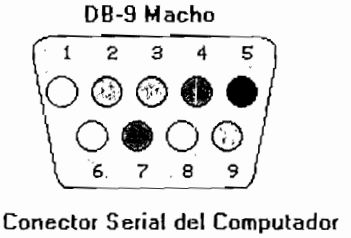
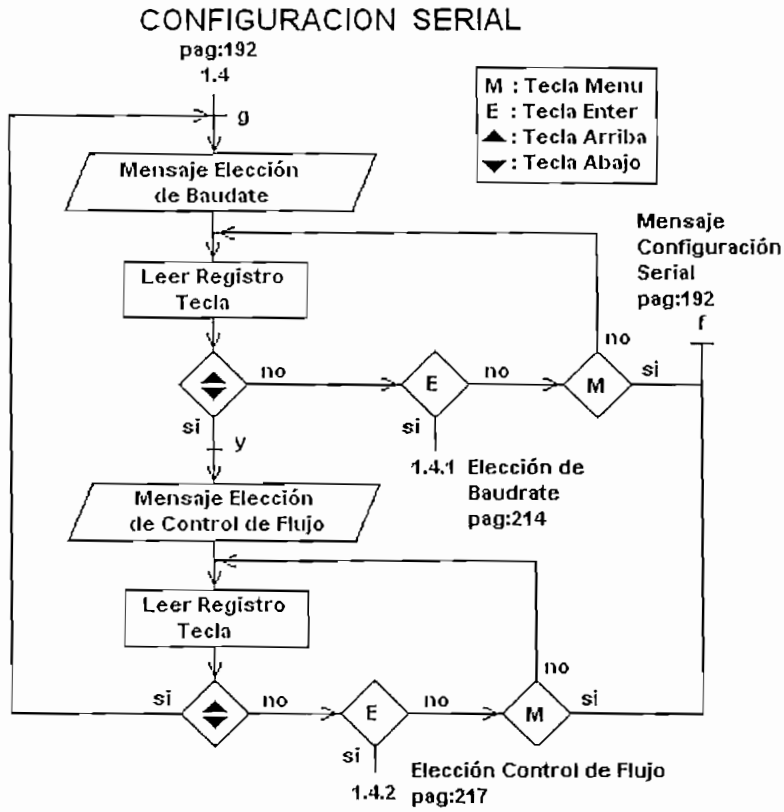


Figura 4.40

4.4.5.4 Configuración del Puerto Serial

Antes de transmitir o recibir datos por el puerto serial, se debe configurar la velocidad de transmisión o baudrate y el control de flujo. Por defecto la transferencia se realiza a 9600bps y con Control de Flujo por Hardware.

En el Módulo:



Menú Configuración Serial

Mensaje Elección de Baudrate

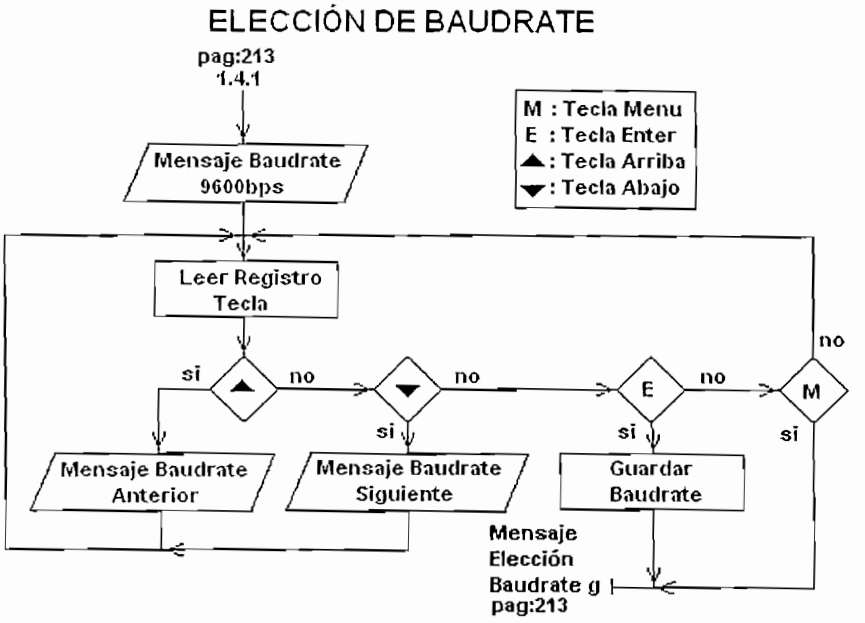
- Presentar mensaje de la opción 2 del Menú Configuración Serial.

- Revisar que tecla se presiona.
- Tecla *Arriba* o *Abajo*: saltar a Mensaje Erección de Control de Flujo.
- Tecla *Enter*: saltar a Mensaje Baudrate 9600bps.
- Tecla *Menu*: saltar a Mensaje Configuración Serial.

Mensaje Elección de Control de Flujo

- Presentar mensaje de la opción 1 del Menú Configuración Serial.

- Revisar que tecla se presiona.
- Tecla *Arriba* o *Abajo*: saltar a Mensaje Elección de Baudrate.
- Tecla *Enter*: saltar a Mensaje Control de Flujo por Hardware.
- Tecla *Menu*: saltar a Mensaje Configuración Serial.



Mensaje Baudrate 300bps

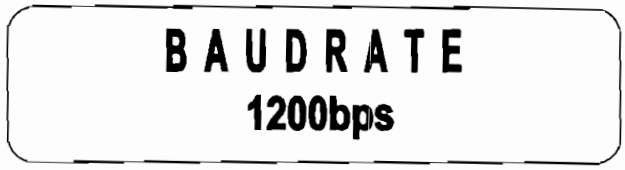
- Presentar mensaje de la opción 1 de baudrate.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Baudrate 28800bps.
- Tecla *Abajo*: saltar a Mensaje Baudrate1200bps.

- Tecla *Enter*: guardar el valor a cargar en el registro de configuración de baudrate y saltar a Mensaje Elección Baudrate.
- Tecla *Menu*: saltar a Mensaje Elección Baudrate.

Mensaje Baudrate 1200bps

- Presentar mensaje de la opción 2 de baudrate.



- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Baudrate 300bps.
- Tecla *Abajo*: saltar a Mensaje Baudrate2400bps.
- Tecla *Enter*: guardar el valor a cargar en el registro de configuración de baudrate y saltar a Mensaje Elección Baudrate.
- Tecla *Menu*: saltar a Mensaje Elección Baudrate.

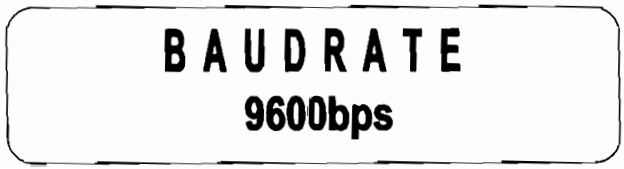
Mensaje Baudrate 2400bps

- Presentar mensaje de la opción 3 de baudrate.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Baudrate 1200bps.
- Tecla *Abajo*: saltar a Mensaje Baudrate 9600bps.
- Tecla *Enter*: guardar el valor a cargar en el registro de configuración de baudrate y saltar a Mensaje Elección Baudrate.
- Tecla *Menu*: saltar a Mensaje Elección Baudrate.

Mensaje Baudrate 9600bps

- Presentar mensaje de la opción 4 de baudrate.



- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Baudrate 2400bps.
- Tecla *Abajo*: saltar a Mensaje Baudrate19200bps.
- Tecla *Enter*: guardar el valor a cargar en el registro de configuración de baudrate y saltar a Mensaje Elección Baudrate.
- Tecla *Menu*: salta a Mensaje Elección Baudrate.

Mensaje Baudrate 19200bps

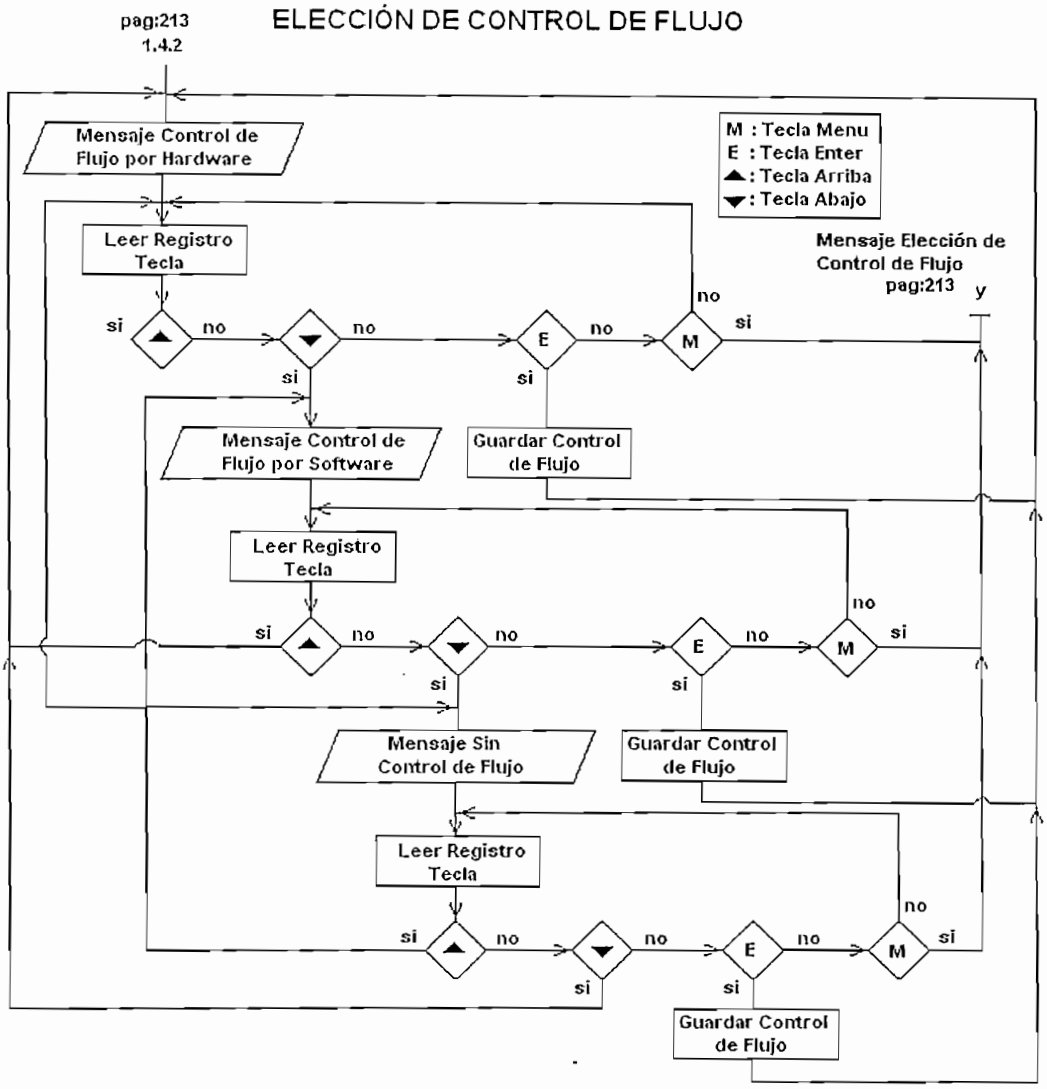
- Presentar mensaje de la opción 5 de baudrate.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Baudrate 9600bps.
- Tecla *Abajo*: saltar a Mensaje Baudrate28800bps.
- Tecla *Enter*: guarda el valor a cargar en el registro de configuración de baudrate y saltar a Mensaje Elección Baudrate.
- Tecla *Menu*: saltar a Mensaje Elección Baudrate.

Mensaje Baudrate 28800bps

- Presentar mensaje de la opción 6 de baudrate.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Baudrate 19200bps.
- Tecla *Abajo*: saltar a Mensaje Baudrate 300bps.
- Tecla *Enter*: guarda el valor a cargar en el registro de configuración de baudrate y saltar a Mensaje Elección Baudrate.
- Tecla *Menu*: saltar a Mensaje Elección Baudrate.



Control de Flujo

Mensaje Control de Flujo por Hardware

- Presentar mensaje de la opción 1 de control de flujo.

<1> CONTROL DE FLUJO
POR HARDWARE

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje sin Control de Flujo.
- Tecla *Abajo*: saltar a Mensaje Control de Flujo por Software.
- Tecla *Enter*: guardar el valor el Control de Flujo elegido y saltar a Mensaje Elección de Control de Flujo.
- Tecla *Menu*: saltar a Mensaje Elección de Control de Flujo.

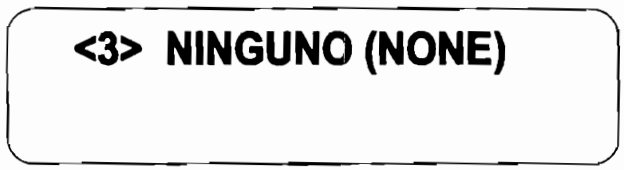
Mensaje Control de Flujo por Software

- Presentar mensaje de la opción 2 de control de flujo.

- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Control de Flujo por Hardware.
- tecla *Abajo*: saltar a Mensaje Sin Control de Flujo.
- Tecla *Enter*: guardar el valor el Control de Flujo elegido y saltar a Mensaje Elección de Control de Flujo.
- Tecla *Menu*: saltar a Mensaje Elección de Control de Flujo.

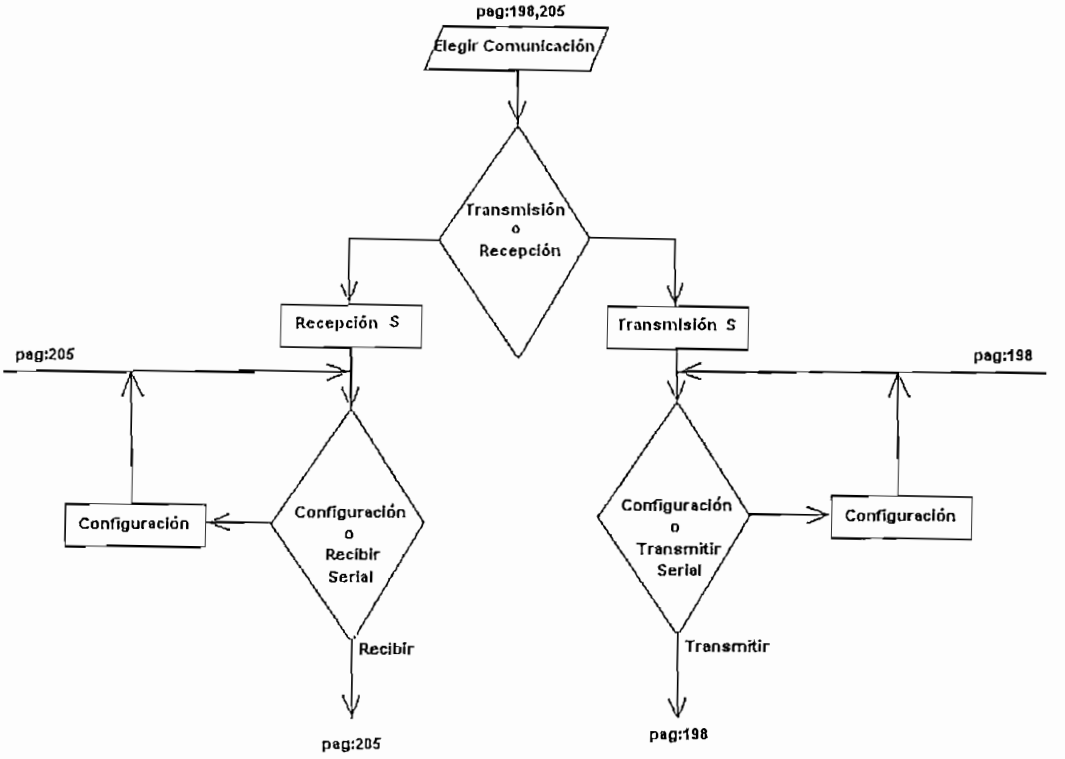
Mensaje Sin Control de Flujo

- Presentar mensaje de la opción 3 de control de flujo.



- Revisar que tecla se presiona.
- Tecla *Arriba*: saltar a Mensaje Control de Flujo por Software.
- Tecla *Abajo*: saltar a Mensaje Control de Flujo por Hardware.
- Tecla *Enter*: guardar el valor el Control de Flujo elegido y saltar a Mensaje Elección de Control de Flujo.
- Tecla *Menu*: salta a Mensaje Elección de Control de Flujo.

En el Computador:



Elegir Comunicación

- Escoge entre las alternativas para las pruebas de comunicación del puerto.
- Transmisión o Recepción (Ver Figura 4.41).

Transmisión S o Recepción S

- Carga el formulario "Transmisión Serial" o "Recepción Serial" y muestra en la pantalla los controles para la prueba de comunicación (Ver Figura 4.42).
- Inicializa los valores por defecto de las señales del puerto para el handshake.
- Carga los valores del protocolo de comunicación (8,n,1), el baudrate, tipo de handshake y formato, según la última configuración ajustada.
- Inhabilita el formulario Puerto Serial.

Configuración.

- Descarga el formulario "Transmisión Serial" o "Recepción Serial" (Ver Figuras 4.43, 4.44, 4.45 y 4.46).
- Carga el formulario Configuración del Puerto Serial, con los últimos valores ajustados para el baudrate, control de flujo y formato.

- Se puede elegir el puerto RS232 donde desea realizar las pruebas, la velocidad de comunicación o baudrate, el control de flujo en la comunicación y el formato de los datos a transmitir o recibir.
- Los valores por defecto del baudrate, control de flujo y formato son 9600 bps, Hardware y Texto.
- Los valores del protocolo de comunicación (8,n,1), son fijos y se mantendrán para cualquier transferencia.
- Carga nuevamente el formulario Transmisión Serial o Recepción Serial dependiendo desde donde se llamo al formulario Configuración del puerto serial.

Figura 4.41

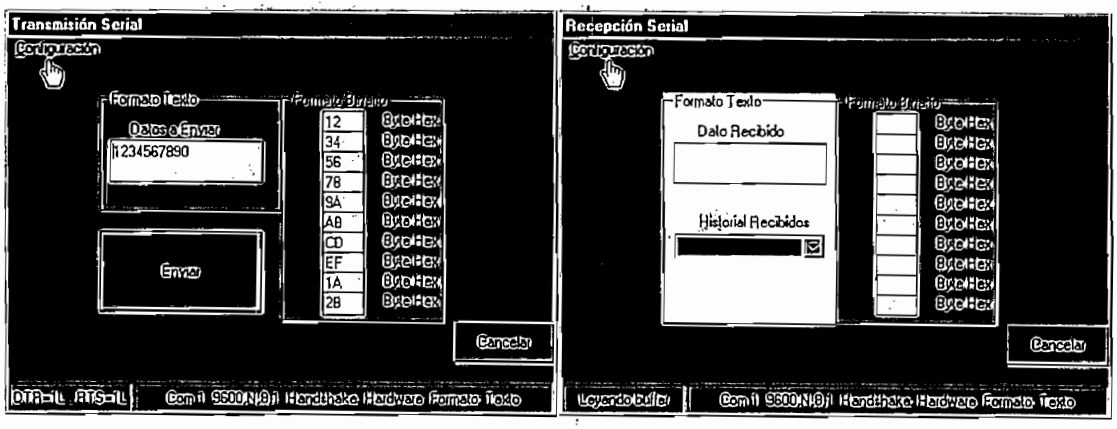


Figura 4.42

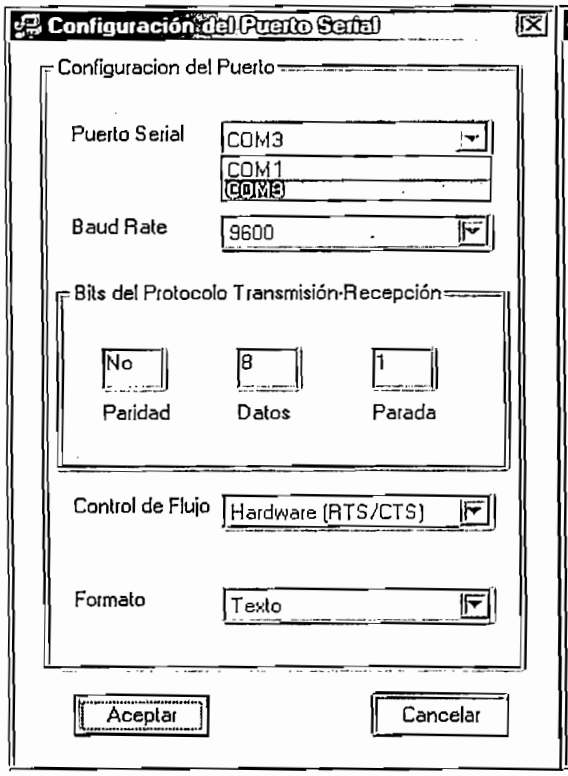


Figura 4.43

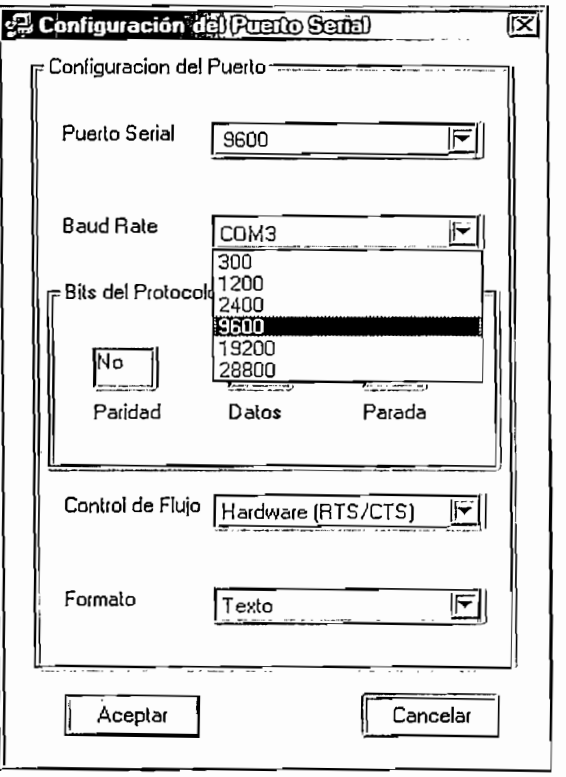


Figura 4.44

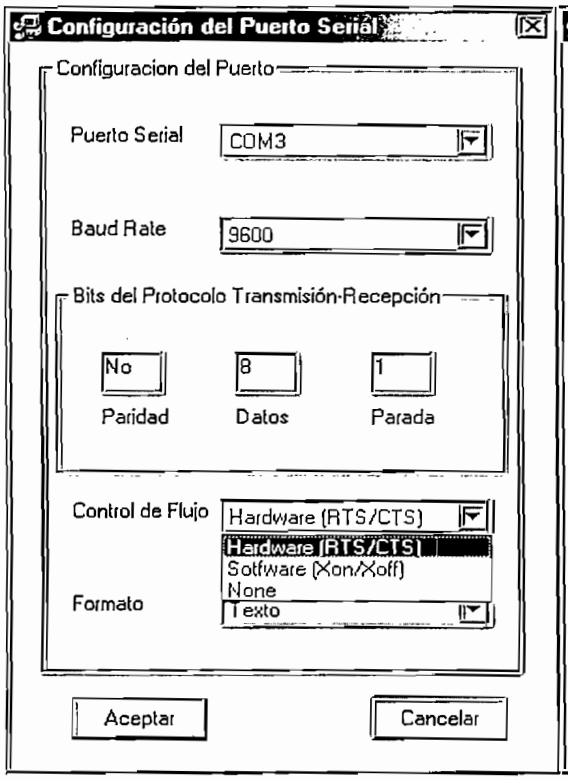


Figura 4.45

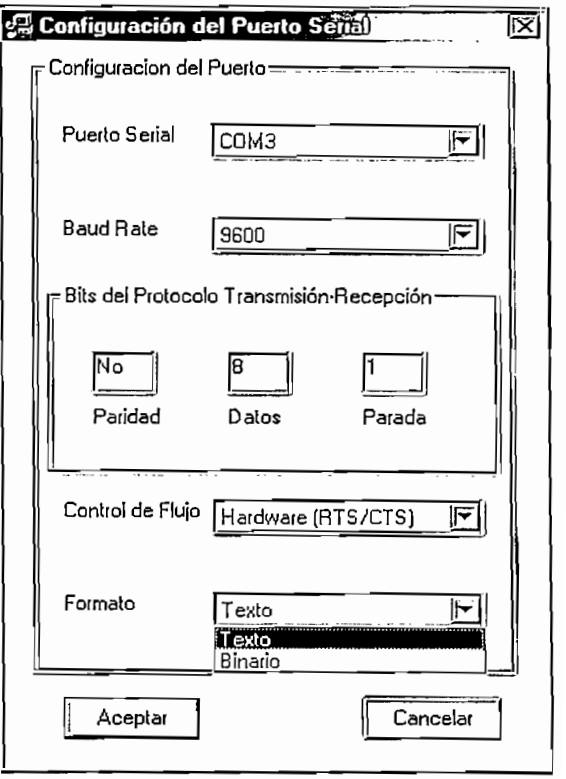
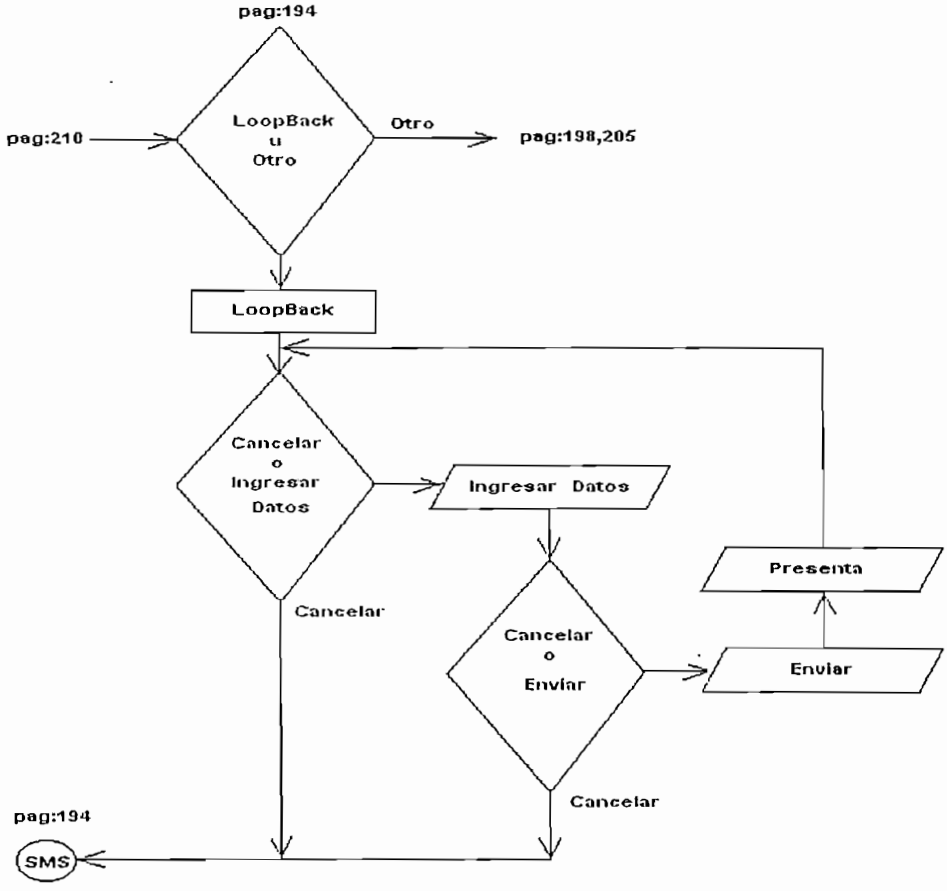


Figura4.46

4.4.5.5 Prueba de Loopback del Puerto Serial

La prueba de Loopback del puerto serial RS232, consisten transmitir una cadena de caracteres desde el computador y recibirlos en el mismo computador utilizando handshake de Hardware. Para este propósito se requiere un conector DB9 de Loopback conectado al puerto.



LoopBack

- Carga el formulario Prueba de LoopBack y muestra en la pantalla los controles para la prueba de comunicación (Ver Figura 4.47).
- Inicializa los valores por defecto de las señales del puerto para el handshake.
- Inhabilita el formulario Puerto Paralelo.

Ingresar Datos

- Recepta 10 caracteres alfanuméricos ingresados por el usuario en un cuadro de texto.

Enviar

- Envía 10 caracteres alfanuméricos en secuencia utilizando en protocolo 8,n,1 con handshake de Hardware, y con formato Texto.
- El handshake, formato y escritura de datos se muestran en una barra de tareas dentro del formulario.

Presenta

- Muestra una pantalla de mensaje que indica el resultado de la transmisión y presenta los datos recibidos mediante la prueba de Loopback (Ver Figuras 4.48, 4.49 y 4.50).
- En la barra de estado, localizada en la parte inferior del formulario, se muestra el proceso de handshake, así como el puerto elegido , baudrate, control de flujo y formato.

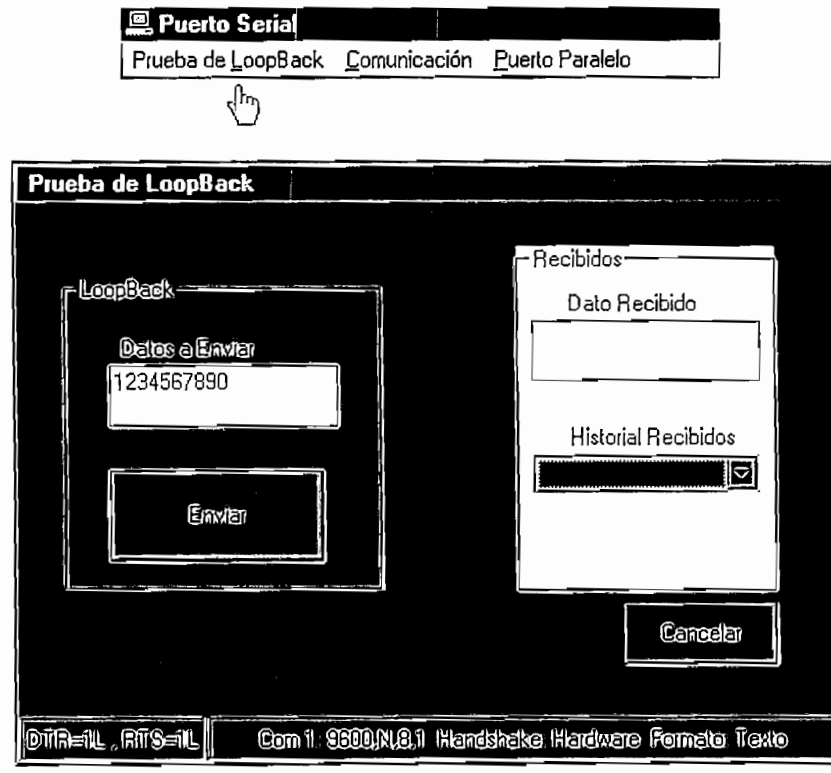


Figura 4.47

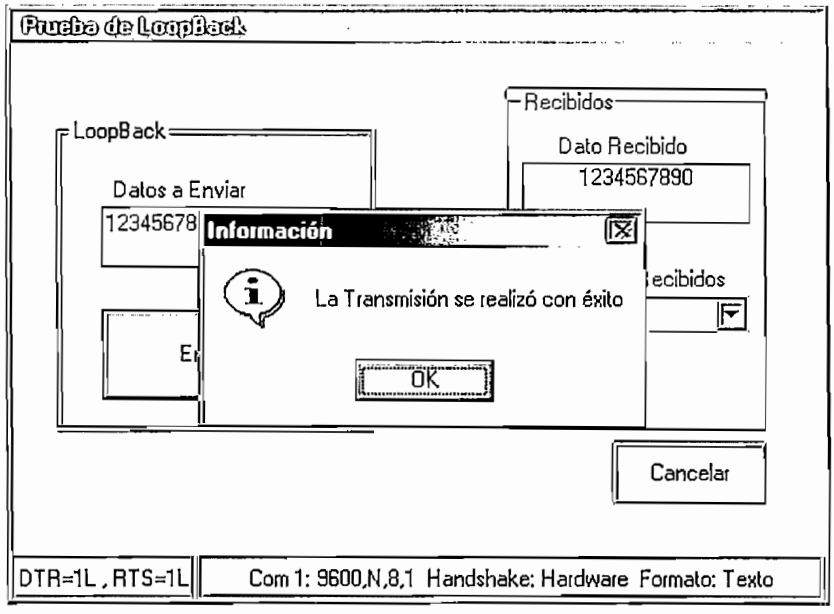


Figura 4.48

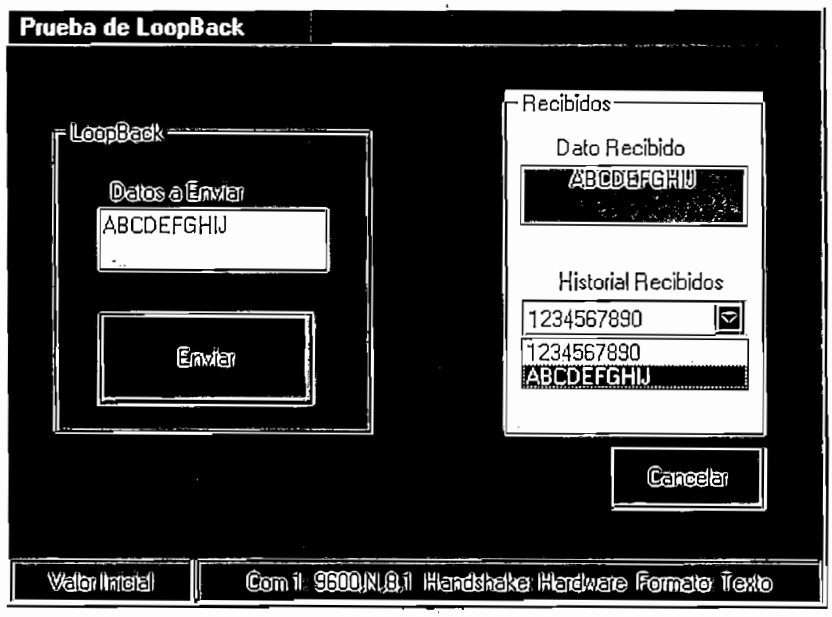


Figura 4.49

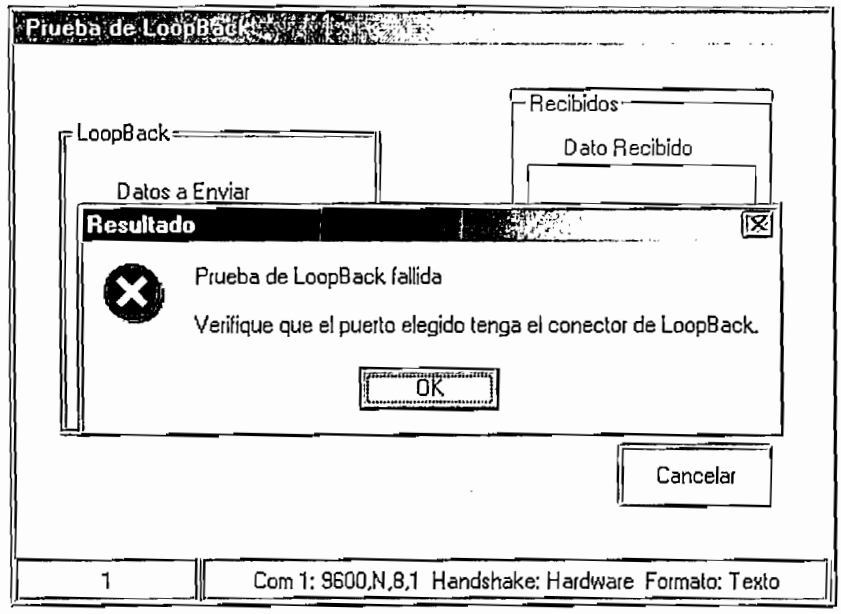

Prueba de LoopBack
Recibidos
LoopBack
Dato Recibido
Datos a Enviar
Resultado
Prueba de LoopBack fallida
Verifique que el puerto elegido tenga el conector de LoopBack.
OK
Cancelar
1
Com 1: 9600,N,8,1 Handshake: Hardware Formato: Texto

Figura 4.50

CAPÍTULO 5
PRUEBAS Y RESULTADOS

En este capítulo se diseñan y corren las pruebas para comprobar el funcionamiento de este sistema.

El primer paso para poder realizar las pruebas del sistema es instalar el programa de soporte en un PC. Esta tarea es sumamente sencilla, pues se dispone de una utilidad de instalación, y el procedimiento de la misma se indica mediante cuadros de dialogo.

Una primera contrariedad surgió en este proceso. Mientras que con sistemas operativos Windows 98 y Millennium, el programa se instalaba y corría sin dificultad, en sistemas operativos Windows NT, 2000 y XP se presentaban inconvenientes. Un cuadro de dialogo anunciaba que algunos de los archivos que se trataban de instalar no estaban actualizados, y que el sistema debía reiniciarse para actualizarlos, sin embargo, luego de reinicializar el sistema, el software no se instalaba.

Para solucionar este percance se debió instalar el Service Pack 5 (SP5) para Visual Basic, que se lo consiguió de la pagina web de Microsoft. El SP5 (que no es mas que un "parche") contiene mejoras y recursos adicionales para la primera versión de VB6.0 que aseguran que las aplicaciones puedan correr sobre sistemas operativos NT.

Con estos recursos adicionales provistos por el SP5, el programa debió ser compilado nuevamente y pudo ser instalado sobre sistemas operativos: 98/Millennium/2000/XP/NT.

Una vez listos el modulo y el programa de soporte, se llevaron a cabo pruebas de transmisión, recepción y bit a bit en los puertos de varios computadores

personales. Se eligieron PCs tanto de escritorio como Laptops que funcionen bajo el sistema operativo Windows.

A continuación se detalla las características de los computadores en los cuales se llevaron a cabo las pruebas:

- Laptop COMPAC Armada 7400, Sistema Operativo Microsoft Windows Me, Procesador Intel Pentium II, 256 MB RAM, Disco Duro 6.06 GB.
- Laptop COMPAC Pesario 700, Sistema Operativo Microsoft Windows XP, Procesador AMD, 256 MB RAM, Disco Duro 40 GB.
- Laptop IMB, Sistema Operativo Microsoft Windows 98, Procesador Intel Pentium II, 256 MB RAM, Disco Duro 10GB.
- PC de escritorio CLON, Sistema Operativo Microsoft Windows 98, Procesador Intel Pentium III, 256 MB RAM, Disco Duro 40 GB.
- PC de escritorio COMPAC , Sistema Operativo Microsoft Windows 2000, Procesador Intel Pentium IV, 512 MB RAM, Disco Duro 80 GB.
- PC de escritorio CLON, Sistema Operativo Microsoft Windows 2000, Procesador Intel Pentium IV, 96 MB RAM, Disco Duro 40 GB.
- PC de escritorio CLON , Sistema Operativo Microsoft Windows NT, X86 Family 6 Model 5 Stepping 3, 416 MB RAM, Disco Duro 40 GB.
- PC de escritorio CLON, Sistema Operativo Microsoft Windows NT, Procesador Intel Pentium IV, 256 MB RAM, Disco Duro 40 GB.
- Laptop IBM ThinkPad 600 , Sistema Operativo Microsoft Windows 98, Procesador Intel Pentium I, 64 MB RAM, Disco Duro 20 GB.
- Laptop Toshiba, sistema Operativo Microsoft Windows XP, Procesador Intel Pentium IV, 256 MB RAM, Disco Duro 40 GB.

5.1 PRUEBAS Y RESULTADOS DE LA COMUNICACIÓN SERIAL

Las pruebas para el puerto serial resultaron más simples que para el puerto paralelo, debido a que se tienen menos señales que controlar y las transferencias se realizan por medio de dos conductores solamente.

El programa del puerto serial inicia reconociendo los puertos instalados, configurados y reconocidos por el sistema operativo. Sobre todos los computadores que se probó el sistema, se verificó que el reconocimiento de puertos seriales hecho por el programa coincida con el hardware instalado.

Una vez elegido un puerto serial, se prosiguió con la siguiente prueba que consiste en manejar las señales de handshake del puerto. El resultado fue optimo, las pruebas resultaron exitosas para todas las señales previstas y no se presentaron dificultades ni errores en ninguno de los computadores sobre los que se experimentó.

En cuanto a la transmisión serial de datos se refiere (PC a PIC y PIC a PC), se presentó un inconveniente importante. Tanto el PIC y como Computador no leían todos los datos transferidos, ambos fallaban en la lectura del ultimo byte.

Se estudió este asunto y se determinó que estaba vinculado con en tiempo de lectura de las señales de handshake y del tiempo de procesamiento de las instrucciones de los programas desarrollados.

Antes de enviar un dato, el transmisor chequea la señal de handshake que le indica si el receptor esta listo para recibirlo. Por su parte el receptor antes de leer un dato, chequea la señal de handshake que le indica si todavía hay datos que recibir.

Debido al tiempo de procesamiento de las instrucciones de ambos dispositivos antes de leer el último byte, se genera un desfase entre el último dato transmitido y la recepción de este. Cuando el receptor esta listo para leer el dato, el transmisor ya genero la señal de fin de datos transmitidos; entonces, el receptor da por terminada la recepción y, como consecuencia, se pierde el último dato transmitido.

Se solucionó este desfase manteniendo las señales de handshake el tiempo suficiente para que puedan ser reconocidas. Por lo descrito anteriormente, para determinar este tiempo se realizaron pruebas en un PC Pentium I, para asegurar que las transferencias en versiones superiores se realicen correctamente. Se determinó entonces que el tiempo adecuado es poco menos de 40 mseg.

Solventado este contratiempo la comunicación pudo establecerse, y los datos pudieron ser transmitidos y recibidos a todas las velocidades implementadas sin perdida de información.

Se realizaron 45 transmisiones y 45 recepciones con cada velocidad de transferencia: 15 con Control de Flujo por Hardware, 15 con Control de Flujo por Software y 15 sin Control de Flujo en una Laptop COMPAC Armada 7400, Sistema Operativo Microsoft Windows Me, Procesador Intel Pentium II, 256 MB RAM, Disco Duro 6.06 GB. El resultado fue óptimo, todas las transferencias se realizaron con éxito.

Las pruebas de LoopBack no presentaron problema alguno. Solamente se debe asegurar que el conector de loopback esté conectado adecuadamente en el puerto elegido. Se realizaron 45 pruebas con cada velocidad de transferencia: 15 con Control de Flujo por Hardware, 15 con Control de Flujo por Software y 15 sin Control de Flujo en una Laptop COMPAC Armada 7400, Sistema Operativo Microsoft Windows Me, Procesador Intel Pentium II, 256 MB RAM, Disco Duro 6.06 GB. Todas las pruebas se realizaron de manera exitosa.

En cada una de las computadoras mencionadas al inicio del capítulo se realizaron 15 transmisiones y 15 recepciones con cada velocidad de transferencia: 5 con Control de Flujo por Hardware, 5 con Control de Flujo por Software y 5 sin Control de Flujo, además de 5 pruebas de Loopback. Todas exitosas.

5.2 PRUEBAS Y RESULTADOS DE LA COMUNICACIÓN PARALELA

Se mencionó que el programa es apto para correr en sistemas operativos de 32 bits, con procesadores Pentium, memoria RAM de 256KB y espacio en disco de al menos 30 MB. Sin embargo, en un principio el programa no siempre pudo manejar el puerto paralelo de computadores que cumplían con estos requerimientos. Así, mientras en sistemas operativos Windows 98/Millennium el programa corrió sin problemas, en sistemas como Windows 2000, XP y NT, no se pudo lograr este propósito.

La razón de este inconveniente radicó en las funciones implementadas en el DLL (inpout32.dll); pues, aunque el DLL elegido en un principio especificaba el manejo de programas de 32 bits, no incluía un driver Kernel Mode necesario para el manejo de sistemas NT. Por lo que se recurrió a otra librería que indicaba explícitamente el manejo de sistemas Windows 2000/NT/XP y además hacia referencia al uso del manejador de dispositivos kernel-mode ya embebido en el propio DLL.

Con esta nueva librería, que tiene el mismo nombre (inpout32.dll), fue posible correr el programa del puerto paralelo en todos los sistemas operativos señalados y seguir adelante con las pruebas.

El programa del puerto paralelo inicia reconociendo los puertos instalados, configurados y reconocidos por el sistema operativo. Sobre todos los computadores que se corrió el programa, se verificó que el reconocimiento de los puertos paralelos hechos por el programa coincida con el hardware instalado en el PC.

En seguida se procedió con las pruebas bit a bit de las señales del puerto, sin presentarse ninguna dificultad. En el programa del PC, en la pantalla LCD del modulo y en los leds indicadores las señales concordaron.

A continuación se llevaron a cabo las pruebas de comunicación. Aquí surgió un nuevo problema. Para los valores 7 y W trasmitidos desde el PIC al PC (Modo Byte), las lecturas correspondientes mostraban otros valores y no siempre los mismos.

Se determino, mediante ensayos de prueba y error, que el latch conectado al puerto paralelo en el modo byte no se llegaba a habilitar a tiempo y, por tanto, el PC leía los valores de alta impedancia del mismo (indeterminados). Esto sucedió debido a que, para estos caracteres, el PIC requería poner varios unos en el puerto a la vez, llegando casi a su máximo umbral de suministro de corriente (recordar que todos los puertos del PIC están siendo cargados con dispositivos que consumen corriente).

Este problema se solucionó conectando un capacitor de 0.1uF entre el pin del PIC que habilita de latch y la referencia, de esta forma el capacitor almacena la energía requerida para activar el pin de habilitación.

Como se esperaba los resultados fueron óptimos. El latch respondió ante la señal de habilitación del PIC y los valores transmitidos en Modo Byte fueron recibidos correctamente.

Solucionada esta dificultad, se consiguió transmitir y recibir datos en todos los modos implementados (Modo Compatible, Modo Nibble y Modo Byte) de manera exitosa. Se realizaron 45 transferencias: 15 en Modo Nibble, 15 en Modo Byte y 15 Modo Compatible en una Laptop COMPAC Armada 7400, Sistema Operativo Microsoft Windows Me, Procesador Intel Pentium II, 256 MB RAM, Disco Duro 6.06 GB.

En cada una de las computadoras descritas al inicio del capítulo se realizaron 15 transferencias: 5 en Modo Nibble, 5 en Modo Byte y 5 en Modo Compatible. Todas exitosas.

Se consiguió entonces correr el programa en sistemas operativos Windows 98/2000/Me/NT/XP con éxito.

5.3 COSTOS DEL SISTEMA

Luego de realizar una lista de materiales, se obtuvieron cotizaciones en varios lugares de venta de material electrónico y se escogió una alternativa considerando la calidad y el costo de los materiales.

A continuación se presenta la alternativa elegida.

MATERIALES			
CANT.	**DESCRIPCIÓN**	**VALOR UNITARIO**	**VALOR TOTAL**
1	Microcontrolador PIC16F877A	12.80	12.80
1	LCD 2x20	19.50	19.50
6	Driver NTE 74HCT244	2.43	14.58
2	Latch 74LS373	0.60	1.20
2	Convertidores TTL a RS232 MAX232N	2.00	4.00
1	Regulador NTE1934	11.80	11.80
1	Oscilador 3.6864MHz	1.20	1.20
3	Metros cable Plano de 50 pines	3.95	11.85
2	Metros cable UTP	0.50	1.00
1	Cable de Impresora	2.50	2.50
1	Metro cable multifilar 12h	0.95	0.95
8	Zócalos maquinado 20 pines	1.00	8.00
2	Zócalos 16 pines	0.07	0.14
1	Zócalo 40 pines	0.12	0.12
24	Leds pequeños	0.09	2.16
31	Resistencias 220,1K,10,1/4W	0.03	0.93
8	Capacitores de Tantalio 10uF	0.70	5.60
13	Capacitores 0.1uF,10uF,1uF	0.20	2.60

9	Conectores 2 pines	0.35	3.15
5	Conectores 4 pines	0.45	2.25
9	Conectores de cable Plano 10 pines	0.50	4.50
1	Conector de cable Plano 14 pines	0.55	0.55
1	Conector de cable Plano 26 pines	0.70	0.70
4	Header 40 pines	1.20	4.80
3	Conectores DB9	1.90	5.70
1	Conector DB9 Macho	1.90	1.90
1	Conector Centronics Hembra	2.25	2.25
6	Pulsantes pequeños	0.35	2.10
1	Pulsante	0.45	0.45
1	Switch ON/OFF	1.10	1.10
1	Adaptador AC/DC 12V,1.5A	6.60	6.60
4	Tarjetas electrónicas terminadas	----------	98.50
1	Caja terminada	25.00	27.00
1	Pasta para suelda	10.00	10.00
1	Carrete Estaño	2.20	2.30
		VALOR TOTAL	274.78

El proyecto se realizó en 6 meses, por lo que, considerando un salario de 400 dólares por mes por cada ingeniero, el costo de ingeniería asciende a 4800 dólares; teniéndose un valor total del proyecto de 5074.78 dólares.

CAPÍTULO 6

CONCLUSIONES Y RECOMENDACIONES

6.1 CONCLUSIONES

- Se determinó que, aunque VB6.0 es una herramienta potente para crear aplicaciones con interface de usuario de fácil uso, tiene un limitante importante en cuanto al manejo del hardware de la computadora. La incorporación de librerías dinámicas proporciona recursos para realizar programas que requieran el manejo de los puertos del computador.
- No se puede asegurar un solo estándar de temporización para las señales de handshake de todos los puertos paralelos desarrollados, debido a que IBM no documentó completamente los requerimientos de tiempo y algunos otros detalles del puerto paralelo de los PCs originales. Hoy en día, el relativamente nuevo estándar IEEE1284, puede servir como base para los nuevos diseñadores.
- No es necesario cumplir a cabalidad la asignación de bits que propone IEEE1284, si se esta programando una interfaz para ambos extremos, basta con que los dos dispositivos establezcan un mismo protocolo con anterioridad.
- Resulta difícil determinar con certeza que circuitos son usados en un puerto paralelo, pues, con muchos fabricantes haciendo puertos, varias implementaciones fueron desarrolladas.
- Aunque las nuevas interfaces seriales son más rápidas e incluyen otras ventajas, la interfaz RS232 se mantiene vigente debido a que es ampliamente difundida, barata, de fácil implementación y todo PC tiene una.
- Aunque la interfaz USB es más rápida que la RS232 (hasta 12Mbp), esta última es preferida, ya que, a pesar que funciona a menor velocidad (hasta 115Kbps), alcanza distancias de 50 a 100 pies en comparación de los 16 pies de una USB.

- A medida que se requiere transferencias de datos a mayor velocidad y distancia, se debe considerar el enlace como una línea de transmisión, de forma que es necesario implementar terminadores de línea para no recaer en problemas de líneas reflejadas, latencia y atenuación.
- En dispositivos que no dispongan de buffers de transmisión y recepción, es necesario utilizar control de flujo para asegurar que la información no se pierda al realizar una transferencia.
- Para tareas sencillas de monitoreo y control, los recursos que proporcionan los puertos paralelo y serial son suficientes.
- Aunque a simple vista un enlace paralelo parece más conveniente que uno serial, a la postre resulta con mayores inconvenientes en cuanto a costo e implementación; pues, el utilizar más líneas de datos acarrea problemas de acople capacitivo entre los conductores, y a medida que la longitud aumenta el costo de la implementación aumenta.
- Las pruebas demostraron que el sistema de pruebas implementado constituye una herramienta útil para probar las capacidades básicas de los puertos Serial y Paralelo de un computador, una herramienta necesaria hoy en día.

6.2 RECOMENDACIONES

- En general aunque el puerto paralelo entrega señales TTL, el hardware del puerto esta limitado en cuanto al manejo de corriente, por tanto el uso de drivers/buffers en la interface entre el computador y un periférico son muy adecuadas para prevenir el daño del puerto; pues un cortocircuito podría afectar y dañar inclusive la tarjeta madre.
- No se debe tratar de manejar cargas directamente desde las líneas del puerto paralelo. Utilice para este fin drivers/buffers en la interfaz entre el puerto y la carga, pues es muy probable que el puerto sufra daño parcial o incluso total.
- Cuando se desea implementar una interfaz para puertos paralelos, se debe tener especial cuidado en el diseño del hardware y la carga que se va a

conectar; pues, con muchos fabricantes implementando nuevos componentes, resulta difícil determinar con certeza qué circuitos son usados en el puerto paralelo. Por tanto el conectar una carga indebida puede dañar el puerto.

- Como la mayoría de los circuitos de puertos paralelos usan lógica TTL y en el mejor de los casos manejadores y buffers, se recomienda el uso de circuitos integrados que sean compatibles con esta lógica, de tal forma que los circuitos conectados al puerto puedan acoplarse y manejarse adecuadamente. Uno de estos integrados es el HCT244.
- Cuando se vaya a elegir un dispositivo externo que vaya a ser conectado al puerto paralelo, es conveniente averiguar sus características tanto de software como de hardware para asegurar compatibilidad con el puerto paralelo. Esta información por lo general esta disponible en las hojas del fabricante.
- Cuando no se pueda asegurar que el puerto paralelo de datos es bidireccional, se recomienda emplearlo únicamente como salida; pues, al conectar un dispositivo que ingrese datos a un puerto paralelo que tenga la capacidad de trabajar solo como salida, ocasionaría daños en el puerto. De allí que no se debe conectar el cable paralelo entre PCs hasta que el puerto de datos de uno de ellos haya sido configurado como entrada.
- Aunque el puerto de control en los primeros puertos paralelos (SPP) esta conformado por circuitos de colector abierto y puede trabajar en forma bidireccional, en los nuevos diseños de puerto paralelo no se puede asegurar que esta condición se mantenga; pues, en algunos puertos multimodo, para trabajar en los modos avanzados de transferencia de datos, los bits de control tienen salidas push pull, por lo que el utilizarlos como entrada resultaría en un daño del puerto.
- La determinación de un puerto serial en un computador no es tan simple como buscar un conector RS232 en el panel posterior; pues, por ejemplo, un puerto serial esta internamente conectado a un modem y el único conector visible en el panel posterior es un RJ11.
- En dispositivos que posean buffers pequeños de recepción y transmisión de datos o que no los tengan, se recomienda emplear técnicas de control

de flujo para evitar la perdida de información; por ejemplo, la técnica envío de acuse de recibo por cada byte recibido.

- La distancia de un cable en una interfaz es relevante cuando se pretende transmitir datos a altas velocidades y distancias grandes. Es así, que para una interfaz RS232 a 9600bps se recomienda no sobrepasar distancias de 50 pies.
- El estándar EIA232 recomienda un conector DB25 hembra para el dispositivo DTE , pero permite el uso de conectores DB9 hembra.
- El estándar IEEE1284 define tres tipos de conectores: IEEE1284 DB25 tipo A, IEEE1284 tipo B (Centronics) y IEEE1284 tipo C (Centronics compacto). Para los nuevos diseños se recomienda el uso del conector compacto Centronics tipo C.
- Se recomienda encender el PC y el módulo de comunicaciones, correr el programa de soporte en el PC, configurar ambos DTEs y luego conectarlos, para evitar daños en los puertos.
- Se recomienda dar a conocer este producto, pues es de mucha utilidad en el medio industrial.

REFERENCIAS BIBLIOGRÁFICAS

LIBROS Y MANUALES

- AXELSON, Jan. "Serial Port Complete". Lakeview Research. 1998
- AXELSON, Jan. "Parallel Port Complete". Lakeview Research. 1997
- -SCHWEBER, William. "Electronic Comunication System". Segunda Edición. Prentice Hall. 1996
- Triangle Research International. "User's Manual" of PLCs
- REVISTAS
- MOON, Robert. "Use a serial port for discrete I/O operation". Microcomputer Journal. 1994
- MATTA, Dany. "Fundamentos de programación en Visual Basic"

DIRECCIONES ELECTRÓNICAS

- www.geocities.com/TimesSquare/Chasm/7990/modem.htm
- www.camiresearch.com/Data_Com_Basic/RS232_standard.html
- www.airborn.com.au/rs232.html
- www.microchip.com/1000/pline/picmicro/category/embctr/14kbytes/device/16f877a
- www.jameco.com
- Jameco. Chips, components, and parallel port-cards, switch boxes, and extenders.
- www.jdr.com
- JDR. Chips, components, and parallel port-cards, switch boxes, and extenders.
- www.fapo.com

APUNTES PERSONALES DE:

- Interfaces de comunicación industrial
- Hardware de computadoras y redes
- Microcontroladores
- Control con microcontroladores
- Curso de certificación CCNA

ANEXO A

INFORMACIÓN ADICIONAL DE PUERTOS EPP

Cambio de Direcciones

En modo EPP, cambiar direcciones no requiere ninguna negociación especial. Se puede mezclar ciclos de lectura y escritura sin pasos extra para ajustar la dirección. Cuando nWrite está baja, el periférico debe inmediatamente deshabilitar sus salidas D0-D7 entonces el PC puede escribir al periférico, y el periférico debe responder al strobe (habilitación) del PC leyendo D0-D7. Siempre que nWrite es alto, el PC debe deshabilitar inmediatamente sus salidas de datos, entonces el periférico puede escribir PC, y el periférico debe responder a la señal de strobe (habilitación) escribiendo a D0-D7.

Si se está diseñando una interface periférica propia y el hardware no soporta cambio (switching) automático de dirección, se puede usar líneas inusuales, o controlar la dirección mediante comandos de software. El resultado no será una interface estándar EPP completa, pero se podrá seguir usando el modo EPP para las transferencias.

Aunque IEEE1284 no especifica esto como un requerimiento, el hardware EPP del PC automáticamente cambia (toggles) las salidas nStrobe o nDStrobe, habilita o deshabilita las salidas de datos cuando es apropiado , y monitorea la entrada nWait para saber si el periférico está listo para una nueva transferencia. No hay necesidad de hacer nada de esto en software.

Temporización

Las especificaciones mínimas de tiempo para transferencias EPP son muy cortas, por lo tanto las transferencias EPP pueden ser muy rápidas. La única restricción en la duración de nAStrobe y nDStrobe, es que ellas sean del ancho suficiente para que el dispositivo receptor pueda detectarlas. En un PC, una transferencia EPP completa puede tomar lugar dentro de un ciclo de I/0 del bus ISA, así el

acceso al puerto puede tener lugar a la velocidad del bus, o alrededor de 1MHz en la mayoría de los sistemas.

Si es necesario, una transferencia EPP puede tomar mucho mas tiempo. Las especificaciones del tiempo permiten que los pulsos nDStrobe by nAStrobe sean tan largos como 1 segundo.

Un importante limite de temporización, es que el periférico debe llevar nWait a alto dentro de los 10 useg después de que nDStrobe o nAStrobe vayan a bajo. Si nWait no va a alto dentro del tiempo permitido, el bit timeout (S0, o bit 0 del puerto de estado) será puesto a 1 y la transferencia será abortada.

Las razones para esta restricción de tiempo, es que a diferencia de una simple lectura y escritura del puerto, en una transferencia EPP, el ciclo de lectura o escritura del bus del sistema de I/O no se completa hasta que nWait esté alto. El ciclo debe completarse dentro de 15 useg para permitir el refresco de la memoria. Otros recursos del sistema pueden estar esperando para acceder al bus del sistema también.

Si se está diseñando circuitos periféricos propios, para prevenir un timeout, un gradual descenso sobre una entrada strobe debe causar que el periférico lleve nWait alto durante 10us.

Variaciones EPP

Entre EPPs hay variaciones en la temporización (timing) de la trasferencia de datos en el ciclo de I/O del PC, en como borrar el bit timeout, y en el uso del bit de dirección del registro de control.

Una variante en EPPs es el resultado de una diferencia entre el protocolo EPP IEEE1284 y el EPP original como el implementado en el chip 82360SL de Intel. El tipo original es algunas veces llamado EPP tipo 1.7, mientras los puertos

compatibles con las señalización IEEE1284 son EPP tipo 1.9. Muchos nuevos EPPs pueden emular ambos.

Si se tiene problema con la transferencia EPP, y el chip controlador soporta ambos tipos de EPP, cambiar por el otro tipo de EPP podría ayudar. El tipo 1.9 está usualmente por defecto.

Borrado de un Timeout

IEEE 1284 no especifica esto, pero el bit 0 de estado es un bit timeout que indica una transferencia EPP fallida. Desafortunadamente, el método para borrar el bit timeout varía con el chip controlador. En los súper controladores de I/O 665 y 666 de SMC, se limpia S0 escribiéndole 1 . Escribirle un 0 no tiene efecto. En los súper I/O de National Semiconductor, después de leer 1 en el bit 0, otra lectura del registro de estado limpia el bit.

Si una aplicación va a correr en diferentes sistemas , o si no se está seguro de qué método usar, se puede hacer ambos: escribir 1 al bit timeout y luego leerlo de nuevo. Se podría escribir 0 al bit para intentar borrarlo, en caso de que ningún chip use el método convencional para borrar el bit.

Borrar el bit timeout es esencial. En 665 y 666 de SMC (y posiblemente en otros chips), un bit timeout bloqueará todas las lecturas y escritura al puerto, en cualquier modo, hasta que el bit sea borrado por software o un reset del sistema. IEEE 1284 especifica que el periférico debe llevar nWait a bajo dentro de 125 ns después de que nDStrobe o nAStrobe retorna a alto. Si el periférico tarda más tiempo y el host es un EPP tipo 1.7, el software del host debería chequear nWait antes de empezar un ciclo de lectura o escritura.

Control de Dirección

Muchos EPPs permiten el control por software o automático del bit 5 del puerto de Control. Como en modo PS/2, cuando C5 es 0, las salidas de datos son

habilitadas, y cuando es 1, las salidas de datos son llevadas a alta impedancia y el PC puede leer señales externas de las líneas de datos.

ANEXO B

INFORMACIÓN ADICIONAL DE PUERTOS ECP

FIFO (First-In First-Out)

En el PC, un ECP tiene típicamente un buffer FIFO de 16 bytes. Las siglas FIFO provienen de First-In First-out, lo que significa que el primer byte en entrar es el primero en salir del buffer.

En una transferencia, el CPU escribe una serie de bytes al buffer, y los circuitos del puerto se encargan de escribirlos en secuencia a las salidas del puerto de Datos. Así, el CPU pasa la responsabilidad del envió de datos al FIFO. En la dirección opuesta, el buffer en cambio almacena la serie de los bytes recibidos, por tanto, el CPU no tiene que preocuparse de leer cada byte antes de que el próximo arribe. Cuando el software lee los bytes desde el FIFO, lo hace en el mismo orden en que el buffer los recibió.

Si el periférico que recibe los datos opera a menor velocidad que el PC, el computador puede escribir los bytes de datos en el FIFO, de forma que se transfiera automáticamente a medida que el periférico este listo para recibirlos. Como la responsabilidad del envío se paso al buffer FIFO, el CPU queda libre para realizar otras tareas.

Cuando se utilizan periféricos muy rápidos, el FIFO también es de utilidad, pues almacena los bytes recibidos en el buffer, permitiendo que el PC los lea cuando se encuentre listo para hacerlo.

Comandos y Datos

El puerto ECP utiliza señales de control para distinguir entre bytes de datos y comandos.

Para transferencias directas, la señal de control es C1 (HostAck).

Para transferencias reversas, la señal de control es S7 (Periph-Ack).

En ambos casos la señal llevada a alto cuando el dispositivo esta enviando datos y es llevada a bajo cuando esta enviando comandos.

Cuando los bytes son de comando, si el bit D7 es 1L, los bits D0-D6 actúan como un canal de direcciones; y, si el bit D7 es 0L, los bits D0-D6 son un contador de longitud usado en la compresión de datos.

Compresión de Datos

Un método de compresión de datos es efectivo cuando los datos contienen bytes idénticos en secuencia, así como muchos archivos gráficos.

Cuando se lleva a cabo un método de compresión, en lugar de la transferencia de los bytes idénticos individualmente, el dispositivo transmisor envía un byte de comando indicando cuantas veces se repite el byte, y luego escribe el byte de datos. Por ejemplo, en lugar de enviar el byte F0H cinco veces, el transmisor escribe primero 5 al FIFO de direcciones del ECP (dirección base). Esto le indica al receptor que debe almacenar cinco copias del próximo byte. Luego el transmisor escribe F0H al FIFO de datos (dirección base + 400H). Esto evita que el receptor tenga que leer cinco veces el mismo dato.
Muchos chips ECP, tales como el súper controlador I/O SMC, soportan la descompresión de datos en el extremo del receptor. En el extremo del transmisor el software es quien debe comprimir los datos.

Uso de canal DMA

Los puertos tipo 3 PS/2 de IBM, fueron los primeros que incluyeron soporte para acceso directo a memoria (DMA), al igual que los puertos ECP.

Todos los PCs tienen un controlador DMA que puede transferir bloques de datos entre la memoria y los puertos. Para realizar una operación de escritura DMA, el

software escribe en el controlador DMA la dirección de inicio de la fuente y el número de bytes. Entonces, el controlador pide el control del bus del sistema llevando la entrada Hold del CPU a alto. Cuando el CPU responde llevando la salida HldA a alto, el controlador DMA escribe cada byte en secuencia en el puerto.

Una operación de lectura DMA es similar, el controlador almacena la dirección de inicio del destinatario y el número de bytes a leer.

Durante una transferencia DMA el CPU puede realizar operaciones internas, pero no puede accesar el bus del sistema hasta que la transferencia DMA se complete y el controlador DMA lleve Hold a bajo de nuevo, devolviendo el control del bus del sistema al CPU.

Debido a que el CPU debe refrescar la RAM del sistema cada 15 useg, las transferencias DMA deben devolver el control del bus del sistema al CPU antes de que expire este tiempo.

Registros de Control Extendido

Los registros de control extendido ECR contienen información de la configuración ECP, incluyendo el modo actual seleccionado. En la siguiente tabla se muestran las funciones de cada uno de los bits del registro.

Bit	Nombre	Read/ Write	Descripción
0	FIFOEmpty	Read only	1= vacío 0= al menos 1 byte de datos presente
1	FIFOFull	Read only	1= lleno 0= al menos queda 1 byte libre
2	ServiceIntr	R/W	1= deshabilitados DMA y servicio de interrupciones 0= Habilitado servicio de interrupciones.

			Puesto a 1: Si dmaEn = 1, durante transferencias DMA Si dmaEn = 0 y direction = 0, cuando el número de bytes libres en el FIFO es igual o más grande que el umbral del FIFO. Si dmaEn = 0 y direction = 1, cuando el número de bytes en el FIFO es igual o más grande que el umbral del FIFO. Después de que el bit ha sido puesto a 1, este puede ser puesto a 0 para habilitar nuevamente las interrupciones.
3	dmaEn	R/W	1= Habilitado DMA 0= Deshabilitado DMA
4	nErrIntrEn	R/W	1= Sin interrupción por nError (S3). 0= Flanco de bajada en nError genera interrupción.
5, 6, 7	ECP mode select	R/W	Ver tabla modos ECP.

Registros de Control Extendido ECP

Temporización

Tal como sucede en el modo EPP, el hardware del puerto realiza automáticamente el handshaking en el PC, de forma que se puede escribir o leer un byte en un ciclo I/O.

Los requerimientos de temporización para el periférico son indefinidos.

En una transferencia directa no hay un límite en cuanto al tiempo que debe tomar el periférico para llevar PeriphAck a alto en respuesta cuando HostClk va a bajo. Por otro lado cuando HostClk retorna a alto el periférico puede tardar hasta 35 mseg para llevar PeriphAck a bajo.

En una transferencia reversa, luego de que el periférico lleva PeriphClk a bajo para enviar un byte y el PC lleva en respuesta HostAck a alto, el periférico puede tomar hasta 35 mseg para llevar PeriphClk nuevamente a alto. En realidad, el hardware dl puerto del PC no monitorea los tiempos, por lo que el software puede permitir al periférico tomar el tiempo que necesite para responder al PC.

Si el periférico no completa una transferencia dentro de un tiempo determinado, el PC debe abortar la Transferencia y retornar al estado en el que estuvo antes de intentar la transferencia. El PC puede determinar si una transferencia se completó leyendo el número de bytes presentes en el FIFO. El chequeo del timeout asegura que el puerto del PC no se bloquee mientras espera la respuesta del periférico. De todas formas el período del timeout es indefinido.

Uso de Interrupciones

En el modo ECP algunos eventos pueden originar una interrupción de hardware en el puerto paralelo.

Para usar una interrupción: un nivel IRQ debe ser seleccionado previamente, habilitado en el controlador de interrupciones del PC y además en el puerto paralelo.

A continuación se enumeran los eventos que pueden generar una interrupción:

1. Transferencias DMA: Cuando serviceIntr = 0, dmaEn = 1, y el contador de terminal DMA (TC) es validado en un ciclo DMA.
2. Salidas ECP: Cuando las señales del ECP serviceIntr = 0, dmaEn = 0, C5 = 0 y el número de bytes libres en el FIFO es igual o mayor que el umbral determinado del mismo.
3. Entradas ECP: Cuando las señales del ECP serviceIntr = 0, dmaEn = 0, C5 = 1 y el número de bytes libres en el FIFO es igual o mayor que el umbral determinado del mismo.

4. Al ocurrir un error: Cuando nErrIntrEn = 0 y nError va a bajo, o si nError es bajo cuando nErrIntrEn va a bajo .
5. En un Acknowlegde: Cuando C4 = 1 y nAck va a alto. Esta es la interrupción convencional del puerto paralelo.

Uso del FIFO

Las transferencias ECP requieren una forma para monitorear el estado del FIFO. Para transferencias directas el PC necesita saber si hay o no espacio en el FIFO para otro byte; y para transferencias reversas, el PC necesita saber cuando hay bytes en espera de ser leídos.
Existen tres maneras de determinar el estado del FIFO. La primera es mediante escrutinios periódicos de I/O, en donde en PC lee constantemente el estado del FIFO desde el registro ECR del EPP. La segunda es programando un manejador de interrupciones, donde el ECP genera una interrupción cuando el FIFO alcanza su umbral, lo que origina que el PC lea o escriba al FIFO. La tercera forma es utilizando los canales DMA, donde el controlador DMA del PC es programado para transferir datos hacia y desde el FIFO.

La configuración de dmaEn (bit 3) y serviceIntr (bit 2) del ECR determina que método está habilitado.

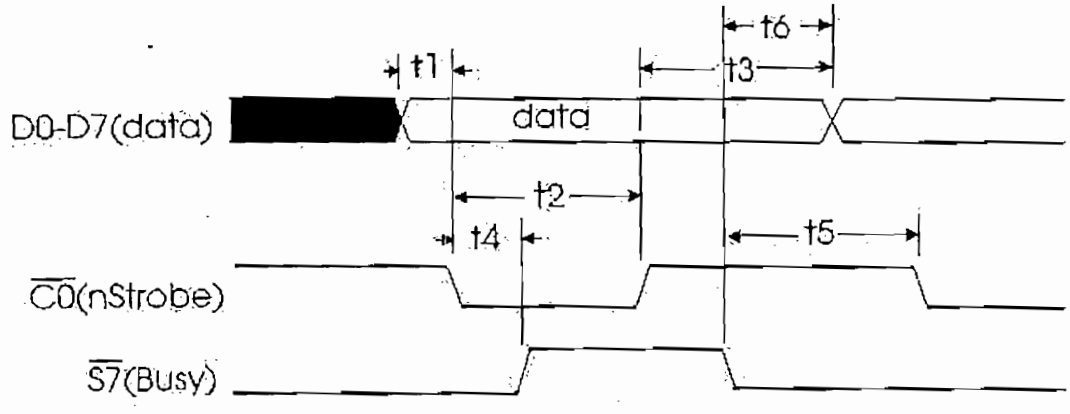
Otros Modos ECP

Los modosa adicionales ECP son Fast Centronics, Test FIFO y Configuración.

Modo Fast Centronics

Un Fast Centronics, o también llamado modo rápido o modo FIFO del puerto Paralelo (010), permite rápidas transferencias de datos en comunicaciones con periféricos SPP.

En este modo el PC escribe datos al FIFO y el hardware lleva a cabo el handshake, escribiendo un byte y pulsando nStrobe (C0) a bajo mientras Busy (S7) esta también en bajo.



Transferencia Fast Centronics

En el modo Fast Centronics, no hay necesidad de hacer el handshaking en software. El uso del FIFO permite al PC escribir una serie de bytes sin tener que esperar la respuesta del periférico.

El condicionante para llevar para una transferencia Fast Centronics es que el periférico esté en modo SPP.

En modo Fast Centronics se realizan únicamente transferencias directas (PC a periférico) . El estándar IEE1284 no menciona el modo Fast Centronics, sin embargo la documentación de Microsoft para ECP describe este modo y los chips controladores ECP lo incluyen.

Modo Test

El modo Test permite leer y escribir al FIFO sin generar señales de handshakig o escritura a los pines del puerto. Este modo es implementado para propósitos de

prueba de la velocidad del puerto, y para determinar a que tasa el PC debe escribir o leer al FIFO para prevenir su desbordamiento.

Modo De Configuración

El modo ECP incluye dos registros de configuración que recolectan la información acerca de la compresión de datos que soporta el chip, el uso de interrupciones y DMA, y los umbrales y estado actual del FIFO. Algunas de las funciones del registro varían dependiendo del chip, mientras otras deben ser las mismas en todos los puertos ECP.

Registro A de Configuración

Es solo de lectura. Devuelve 10H, lo que indica que el puerto tiene una implementación de 8 bits.

Registro de Configuración B

Bit7: Compresión. Solo de lectura. Un cero en este bit indica que el hardware no soporta compresión RLE (sin embargo el chip puede soportar descompresión RLE)
Bit6: IntrValue. Solo de lectura. Contiene el estado de la línea IRQ del puerto.
Bits 5, 4, 3: En algunos chips selecciona un nivel IRQ.
Bits 2, 1, 0: En algunos chip selecciona un canal DMA.

ANEXO C

PIC16F87XA

FLASH Memory Programming Specification

This document includes programming specifications for the following devices:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

1.0 PROGRAMMING THE PIC16F87XA

The PIC16F87XA is programmed using a serial method. The Serial mode will allow the PIC16F87XA to be programmed while in the user's system. This allows for increased design flexibility. This programming specification applies to PIC16F87XA devices in all packages.

1.1 Programming Algorithm Requirements

The programming algorithm used depends on the operating voltage (V_{DD}) of the PIC16F87XA device, or whether internal or external timing is desired.

Algorithm #	V_{DD} Range	Timing
1	$2.0V \le V_{DD} < 5.5V$	Internal; 4 ms/op
2	$4.5V \le V_{DD} \le 5.5V$	External; 1 ms/op

Both algorithms can be used with the two available programming entry methods. The first method follows the normal Microchip Programming mode entry of holding pins RB6 and RB7 low, while raising $\overline{MCLR}$ pin from V_{IL} to V_{IHH} (13V ± 0.5V). The second method, called Low Voltage ICSP™ or LVP for short, applies V_{DD} to $\overline{MCLR}$ and uses the I/O pin RB3 to enter Programming mode. When RB3 is driven to V_{DD} from ground, the PIC16F87XA device enters Programming mode.

1.2 Programming Mode

The Programming mode for the PIC16F87XA allows programming of user program memory, data memory, special locations used for ID, and the configuration word.

Pin Diagrams

PDIP, SOIC

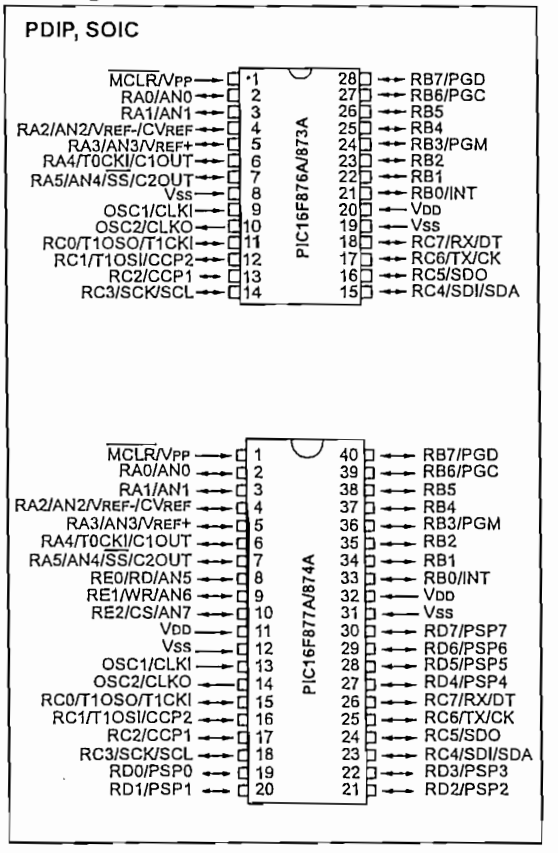


TABLE 1-1: PIN DESCRIPTIONS (DURING PROGRAMMING): PIC16F87XA

Pin Name	During Programming		
	Function	Pin Type	Pin Description
RB3	PGM	I	Low voltage ICSP programming input if LVP configuration bit equals '1'
RB6	CLOCK	I	Clock input
RB7	DATA	I/O	Data input/output
$\overline{MCLR}$	$V_{TEST\ MODE}$	P*	Program Mode Select
V_{DD}	V_{DD}	P	Power Supply
V_{SS}	V_{SS}	P	Ground

Legend: I = Input, O = Output, P = Power

* To activate the Programming mode, high voltage needs to be applied to the $\overline{MCLR}$ input. Since $\overline{MCLR}$ is used for a level source, this means that $\overline{MCLR}$ does not draw any significant current.

2.0 PROGRAM MODE ENTRY

2.1 User Program Memory Map

The user memory space extends from 0000h to 1FFFh (8 K words). In Programming mode, the program memory space extends from 0000h to 3FFFh, with the first half (0000h - 1FFFh) being user program memory and the second half (2000h - 3FFFh) being configuration memory. The PC will increment from 0000h to 1FFFh and wrap around to 0000h. From 2000h, the PC will increment up to 3FFFh and wrap around to 2000h (not to 0000h). Once in configuration memory, the highest bit of the PC stays a '1', thus always pointing to the configuration memory. The only way to point to user program memory is to reset the part and re-enter Program/Verify mode, as described in Section 2.4.

In the configuration memory space, 2000h - 200Fh are physically implemented. However, only locations 2000h through 2007h are available. Other locations are reserved. Locations beyond 200Fh will physically access user memory (see Figure 2-1).

2.2 Data EEPROM Memory

The EEPROM data memory space is a separate block of high endurance memory that the user accesses, using a special sequence of instructions. The amount of data EEPROM memory depends on the device and is shown below in number of bytes.

Device	# of Bytes
PIC16F873A	128
PIC16F874A	128
PIC16F876A	256
PIC16F877A	256

The contents of data EEPROM memory have the capability to be embedded into the HEX file.

The programmer should be able to read data EEPROM information from a HEX file and conversely (as an option), write data EEPROM contents to a HEX file, along with program memory information and configuration bit information.

The 256 data memory locations are logically mapped starting at address 2100h. The format for data memory storage is one data byte per address location, LSB aligned.

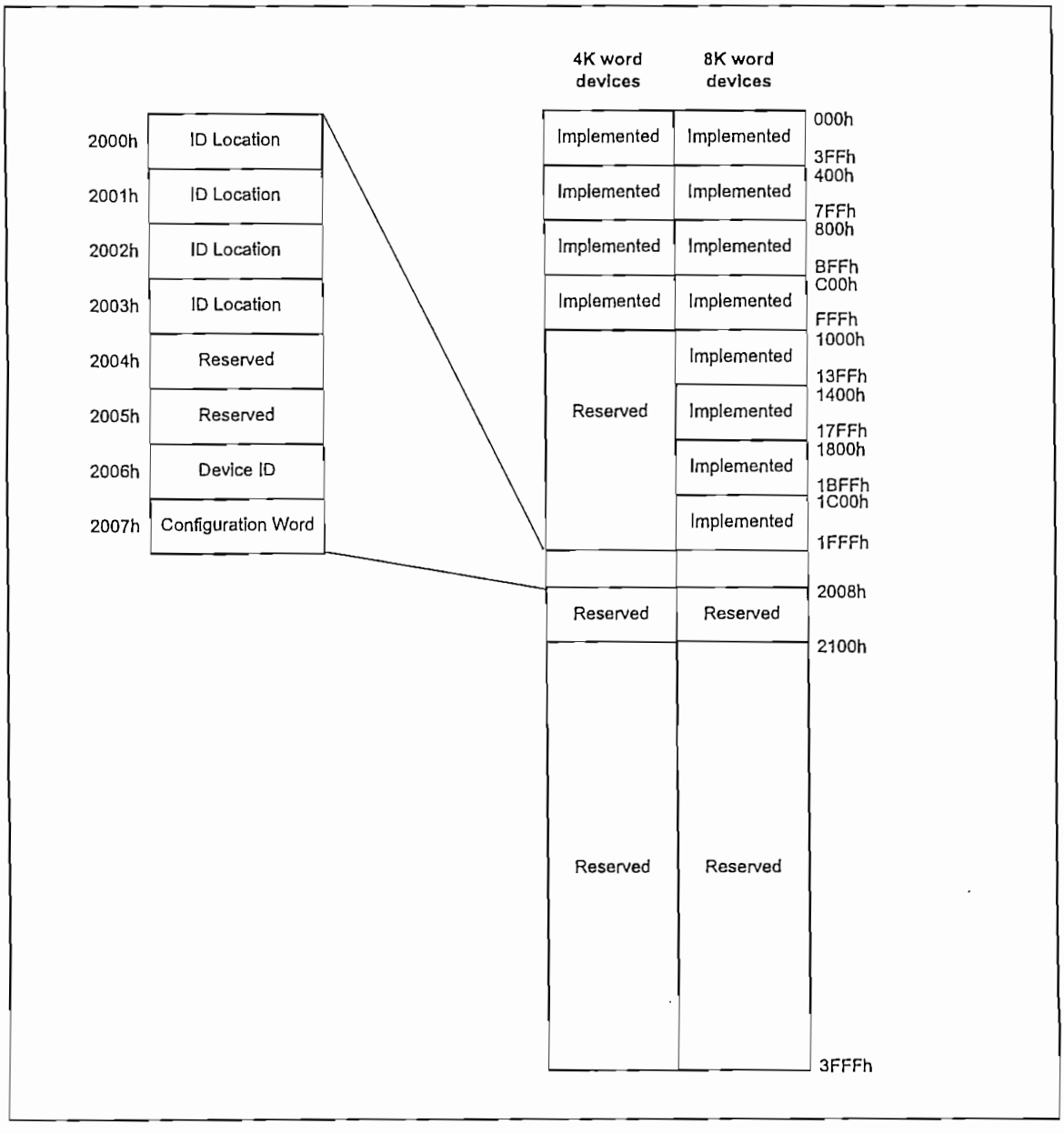
2.3 ID Locations

A user may store identification information (ID) in four ID locations. The ID locations are mapped in addresses 2000h - 2003h. It is recommended that the user use only the four Least Significant bits of each ID location. In some devices, the ID locations read out in an unscrambled fashion after code protection is enabled.

For these devices, it is recommended that ID location is written as "11 1111 1000 bbbb", where 'bbbb' is ID information.

In other devices, the ID locations read out normally, even after code protection. To understand how the devices behave, refer to Table 5-1.

FIGURE 2-1: PIC16F87XA PROGRAM MEMORY MAPPING



2.4 Program/Verify Mode

The Program/Verify mode is entered by holding pins RB6 and RB7 low, while raising MCLR pin from VIL to VIHH (high voltage). In this mode, the state of the RB3 pin does not effect programming. Low Voltage ICSP Programming mode is entered by raising RB3 from VIL to VDD, and then applying VDD to MCLR. Once in this mode, the user program memory and the configuration memory can be accessed and programmed in serial fashion. The mode of operation is serial, and the memory accessed is the user program memory. RB6 and RB7 are Schmitt Trigger inputs in this mode.

Note:	The OSC must not have 72 osc clocks while the device MCLR is between VIL and VIHH.

The sequence that enters the device into the Programming/Verify mode places all other logic into the RESET state (the MCLR pin was initially at VIL). This means all I/O are in the RESET state (high impedance inputs).

A device RESET will clear the PC and set the address to '0'. The 'Increment Address' command will increment the PC. The 'Load Configuration' command will set the PC to 2000h. The available commands are shown in Table 2-1.

The normal sequence for programming eight program memory words at a time is as follows:

1. Load a word at the current program memory address using the 'Load Data' command.
2. Issue an 'Increment Address' command.
3. Load a word at the current program memory address using the 'Load Data' command.
4. Repeat Step 2 and Step 3 six times.
5. Issue a 'Begin Programming' command to begin programming.
6. Wait tprog (about 1 ms).
7. Issue an 'End Programming' command.
8. Increment to the next address.
9. Repeat this sequence as required to write program and configuration memory.

The alternative sequence for programming one program memory word at a time is as follows:

1. Set a word for the current memory location using the 'Load Data' command.
2. Issue a 'Begin Programming Only' command to begin programming.
3. Wait tprog.
4. Issue an 'End Programming' command.
5. Increment to the next address.
6. Repeat this alternative sequence as required to write program and configuration memory.

The address and program counter are reset to 0000h by resetting the device (taking MCLR below VIL) and re-entering Programming mode. Program and configuration memory may then be read or verified using the 'Read Data' and 'Increment Address' commands.

2.4.1 LOW VOLTAGE ICSP PROGRAMMING MODE

Low Voltage ICSP Programming mode allows a PIC16F87XA device to be programmed using VDD only. However, when this mode is enabled by a configuration bit (LVP), the PIC16F87XA device dedicates RB3 to control entry/exit into Programming mode.

When LVP bit is set to '1', the low voltage ICSP programming entry is enabled. Since the LVP configuration bit allows low voltage ICSP programming entry in its erased state, an erased device will have the LVP bit enabled at the factory. While LVP is '1', RB3 is dedicated to low voltage ICSP programming. Bring RB3 and then, MCLR to VDD to enter Programming mode. All other specifications for high voltage ICSP apply.

To disable Low Voltage ICSP mode, the LVP bit must be programmed to '0'. This must be done while entered with the High Voltage Entry mode (LVP bit = '1'). RB3 is now a general purpose I/O pin.

2.4.2 SERIAL PROGRAM/VERIFY OPERATION

The RB6 pin is used as a clock input pin, and the RB7 pin is used to enter command bits, and to input or output data during serial operation. To input a command, the clock pin (RB6) is cycled six times. Each command bit is latched on the falling edge of the clock, with the Least Significant bit (LSb) of the command being input first. The data on RB7 is required to have a minimum setup (tset1) and hold (thold1) time (see AC/DC specifications), with respect to the falling edge of the clock. Commands with associated data (read and load) are specified to have a minimum delay (tdly1) of 1 µs between the command and the data. After this delay, the clock pin is cycled 16 times, with the first cycle being a START bit (0) and the last cycle being a STOP bit (0). Data is transferred LSb first.

During a read operation, the LSb will be transmitted onto RB7 on the rising edge of the second cycle, and during a load operation, the LSb will be latched on the falling edge of the second cycle. A minimum 1 µs delay (tdly2) is specified between consecutive commands.

All commands and data words are transmitted LSb first. The data is transmitted on the rising edge, and latched on the falling edge of the clock. To allow decoding of commands and reversal of data pin configuration, a time separation of at least 1 µs (tdly1) is required between a command and a data word, or another command.

The available commands are described in the following paragraphs and listed in Table 2-1.

2.4.2.1 Load Configuration

After receiving this command, the program counter (PC) will be set to 2000h. By then applying 16 cycles to the clock pin, the chip will load 14 bits in a "data word," as described above, to be programmed into the configuration memory. A description of the memory mapping schemes of the program memory for normal operation and configuration mode operation is shown in Figure 2-1. After the configuration memory is entered, the only way to get back to the user program memory is to exit the Program/Verify Test mode by taking MCLR low (V_{IL}).

2.4.2.2 Load Data for Program Memory

After receiving this command, the chip will load one word (with 14 bits as a "data word") to be programmed into user program memory when 16 cycles are applied. A timing diagram for this command is shown in Figure 6-1.

2.4.2.3 Load Data for Data Memory

After receiving this command, the chip will load in a 14-bit "data word" when 16 cycles are applied. However, the data memory is only 8-bits wide, and thus, only the first 8 bits of data after the START bit will be programmed into the data memory. It is still necessary to cycle the clock the full 16 cycles in order to allow the internal circuitry to reset properly. The data memory contains up to 256 bytes. If the device is code protected, the data is read as all zeros. A timing diagram for this command is shown in Figure 6-2.

2.4.2.4 Read Data from Program Memory

After receiving this command, the chip will transmit data bits out of the program memory (user or configuration) currently accessed, starting with the second rising edge of the clock input. The RB7 pin will go into Output mode on the second rising clock edge, and it will revert back to Input mode (hi-impedance) after the 16th rising edge. A timing diagram of this command is shown in Figure 6-3.

2.4.2.5 Read Data from Data Memory

After receiving this command, the chip will transmit data bits out of the data memory, starting with the second rising edge of the clock input. The RB7 pin will go into Output mode on the second rising edge, and it will revert back to Input mode (hi-impedance) after the 16th rising edge. As previously stated, the data memory is 8-bits wide, and therefore, only the first 8 bits that are output are actual data. A timing diagram for this command is shown in Figure 6-4.

2.4.2.6 Increment Address

The PC is incremented when this command is received. A timing diagram of this command is shown in Figure 6-5.

2.4.2.7 Begin Erase/Program Cycle

Eight locations must be loaded before every 'Begin Erase/Programming' command. After this command is received and decoded, eight words of program memory will be erased and programmed with the values contained in the program data latches. The PC address will decode which eight words are programmed. The lower three bits of the PC are ignored, so if the PC points to address 003h, then all eight locations from 000h to 007h are written.

An internal timing mechanism executes an erase before write. The user must allow the combined time for erase and programming, as specified in the electrical specs, for programming to complete. No 'End Programming' command is required.

1. If the address is pointing to user memory, the user memory alone will be affected.
2. If the address is pointing to the physically implemented test memory (2000h - 201Fh), test memory will be written. The configuration word will not be written unless the address is specifically pointing to 2007h.

This command can be used to perform programming over the entire V_{DD} range of the device.

Note 1:	The code protect bits cannot be erased with this command.
2:	All Begin Erase/Programming operations can take place over the entire V_{DD} range.

A timing diagram for this command is shown in Figure 6-6.

2.4.2.8 Begin Programming Only

Note:	Begin Programming Only operations must take place at the 4.5V to 5.5V V_{DD} range.

This command is similar to the 'Erase/Programming Cycle' command, except that a word erase is not done, and the internal timer is not used. Programming of program and data memory will begin after this command is received and decoded. The user must allow the time for programming, as specified in the electrical specs, for programming to complete. An 'End Programming' command is required.

The internal timer is not used for this command, so the 'End Programming' command must be used to stop programming.

1. If the address is pointing to user memory, the user memory alone will be affected.
2. If the address is pointing to the physically implemented test memory (2000h - 201Fh), the test memory will be written. The configuration word will not be written unless the address is specifically pointing to 2007h.

A timing diagram for this command is shown in Figure 6-7.

2.4.2.9 End Programming

After receiving this command, the chip stops programming the memory (test program memory or user program memory) that it was programming at the time.

Note:	This command will also set the write data shift latches to all '1's to avoid issues with downloading only one word before the write.

TABLE 2-1: COMMAND MAPPING FOR PIC16F87XA

Command	Mapping (MSB ... LSB)					Data	Voltage Range
Load Configuration	0	0	0	0	0	0, data (14), 0	2.2V - 5.5V
Load Data for Program Memory	0	0	0	1	0	0, data (14), 0	2.2V - 5.5V
Read Data from Program Memory	0	0	1	0	0	0, data (14), 0	2.2V - 5.5V
Increment Address	0	0	1	1	0		2.2V - 5.5V
Begin Erase/Programming Cycle	0	1	0	0	0	4 ms typical, internally timed	2.2V - 5.5V
Begin Programming Only Cycle	1	1	0	0	0	1 ms typical, externally timed	4.5V - 5.5V
Bulk Erase Program Memory	0	1	0	0	1	4 ms typical, internally timed	4.5V - 5.5V
Bulk Erase Data Memory	0	1	0	1	1	4 ms typical, internally timed	4.5V - 5.5V
Chip Erase	1	1	1	1	1	4 ms typical, internally timed	4.5V - 5.5V
Load Data for Data Memory	0	0	0	1	1	0, data (14), 0	2.2V - 5.5V
Read Data from Data Memory	0	0	1	0	1	0, data (14), 0	2.2V - 5.5V
End Programming	1	0	1	1	1		

2.5 Erasing Program and Data Memory

Depending on the state of the code protection bits, program and data memory will be erased using different methods. The first two commands are used when both program and data memories are not code protected. The third command is used when either memory is code protected, or If you want to also erase the fuse locations, including the code protect bits. A device programmer should determine the state of the code protection bits and then apply the proper command to erase the desired memory.

2.5.1 ERASING NON-CODE PROTECTED PROGRAM AND DATA MEMORY

When both program and data memories are not code protected, they must be individually erased using the following commands. The only way that both memories are erased using a single command is if code protection is enabled for one of the memories. These commands do not erase the configuration word or ID locations.

2.5.1.1 Bulk Erase Program Memory

When this command is performed, and is followed by a 'Begin Erase/Programming' command, the entire program memory will be erased.

If the address is pointing to user memory, only the user memory will be erased.

If the address is pointing to the test program memory (2000h - 201Fh), then both the user memory and the test memory will be erased. The configuration word will not be erased, even if the address is pointing to location 2007h.

Previously, a load data with 0FFh command was recommended before any Bulk Erase. On these devices, this will not be required.

The Bulk Erase command is disabled when the CP bit is programmed to '0' enabling code protect.

A timing diagram for this command is shown in Figure 6-8.

2.5.1.2 Bulk Erase Data Memory

When this command is performed, and is followed by a 'Begin Erase/Programming' command, the entire data memory will be erased.

The Bulk Erase Data command is disabled when the CPD bit is programmed to '0' enabling protected data memory. A timing diagram for this command is shown in Figure 6-9.

Note:	All Bulk Erase operations must take place at the 4.5V to 5.5V VDD range.

2.5.1.3 Chip Erase

This command, when performed, will erase the program memory, EE data memory, and all of the fuse locations, including the code protection bits. All on-chip FLASH and EEPROM memory is erased, regardless of the address contained in the PC.

When a Chip Erase command is issued and the PC points to (0000h - 1FFFh), the configuration word and the user program memory will be erased, but not the test row (see Section 2.5.2.1). Chip Erase can also be used to erase code protected memory, as described in Section 2.5.2.

This command will also erase the code protect and code protect data fuses if they are programmed. This is the only command that allows a user to erase the code protect fuses.

The Chip Erase is internally self-timed to ensure that all program and data memory is erased before the code protect bits are erased. A timing diagram for this command is shown in Figure 6-10.

Note:	The Chip Erase operation must take place at the 4.5V to 5.5V VDD range.

2.5.2 ERASING CODE PROTECTED MEMORY

For the PIC16F87XA devices, once code protection is enabled, all protected program and data memory locations read all '0's and further programming is disabled. The ID locations and configuration word read out unscrambled and can be reprogrammed normally. The only command to erase a code protected PIC16F87XA device is the Chip Erase. This erases program memory, data memory, configuration bits and ID locations. **Since all data within the program and data memory will be erased when this command is executed, the security of the data or code is not compromised.**

2.5.2.1 Chip Erase

This command, when performed, will erase the program memory, data EEPROM, and all of the fuse locations, including the code protection bits, code protect fuses, and code protect data fuses. All on-chip FLASH and EEPROM memory is erased, regardless of the address contained in the PC.

If the PC points to user memory, the test row (2000h through 201Fh) is not erased with a Chip Erase command, except for the configuration word (at 2007h). If the test row is to be completely erased, the address in the PC must point to configuration memory.

When the PC points to 2000h - 201Fh, the configuration word, test program memory, and the user program memory will all be erased with a Chip Erase command. This allows the user to erase all program and configuration content, including the code protect bits, without compromising the user ID bits (2000h through 2004h), or any pass codes stored in the test row.

The Chip Erase is internally self-timed to ensure that all program and data memory is erased before the code protect bits are erased.

A timing diagram for this command is shown in Figure 6-10.

Note:	The Chip Erase operation must take place at the 4.5V to 5.5V VDD range.

FIGURE 2-2: ALGORITHM 1 FLOW CHART – PROGRAM MEMORY (2.0V ≤ VDD < 5.5V)

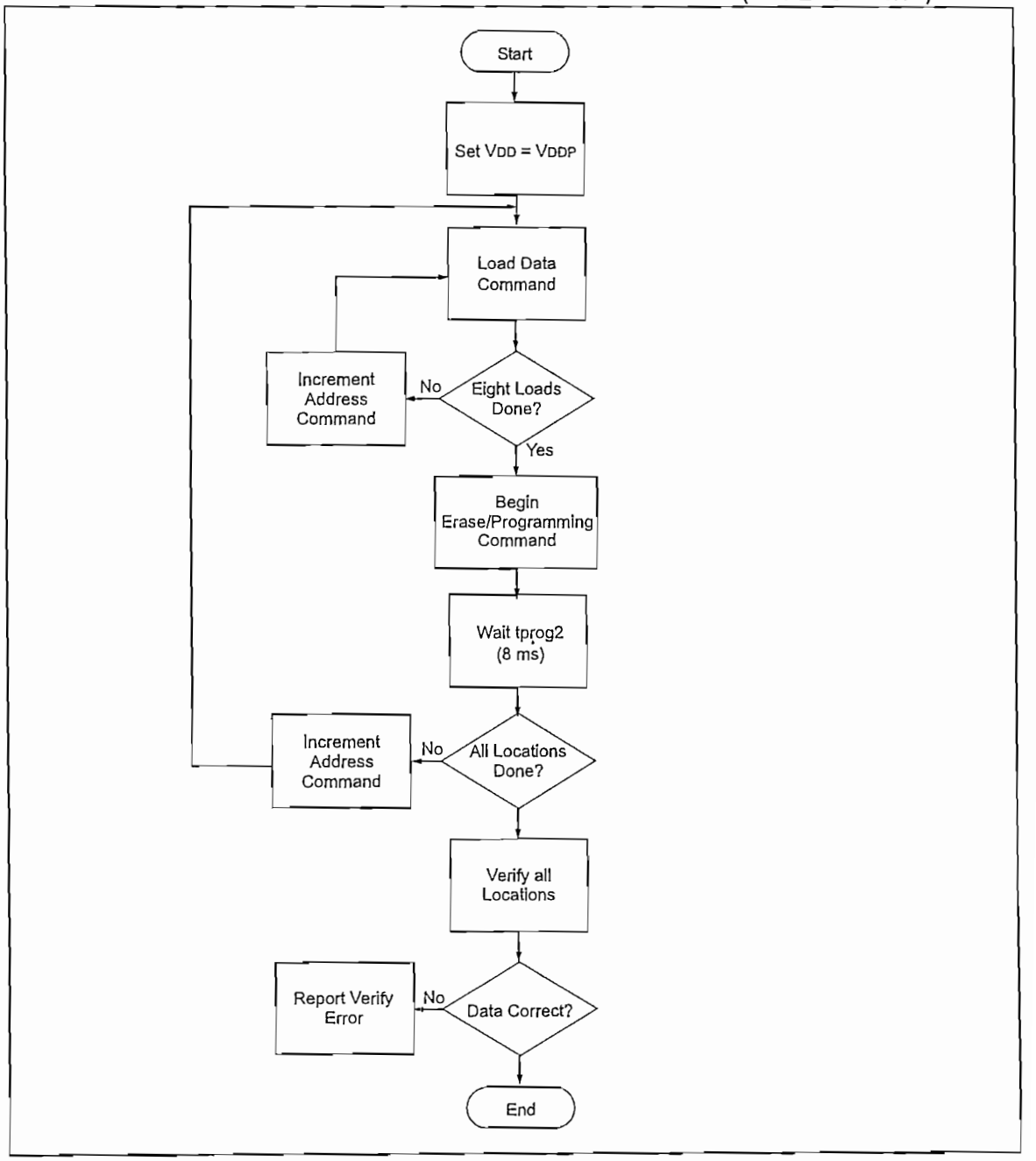


FIGURE 2-3: ALGORITHM 2 FLOW CHART – PROGRAM MEMORY (4.5V ≤ VDD ≤ 5.5V)

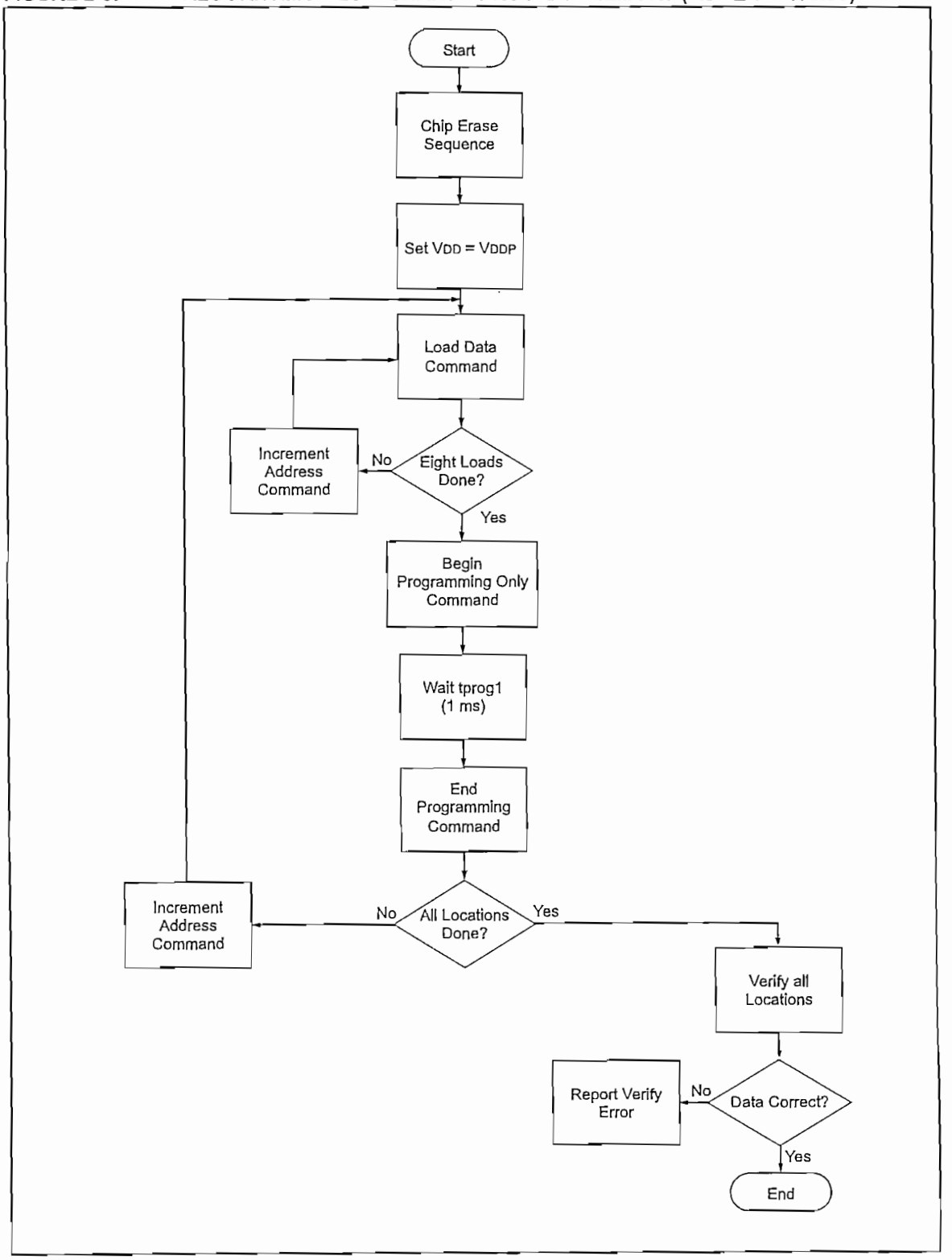


FIGURE 2-4: FLOW CHART – PIC16F87XA CONFIGURATION MEMORY (2.0V ≤ VDD < 5.5V)

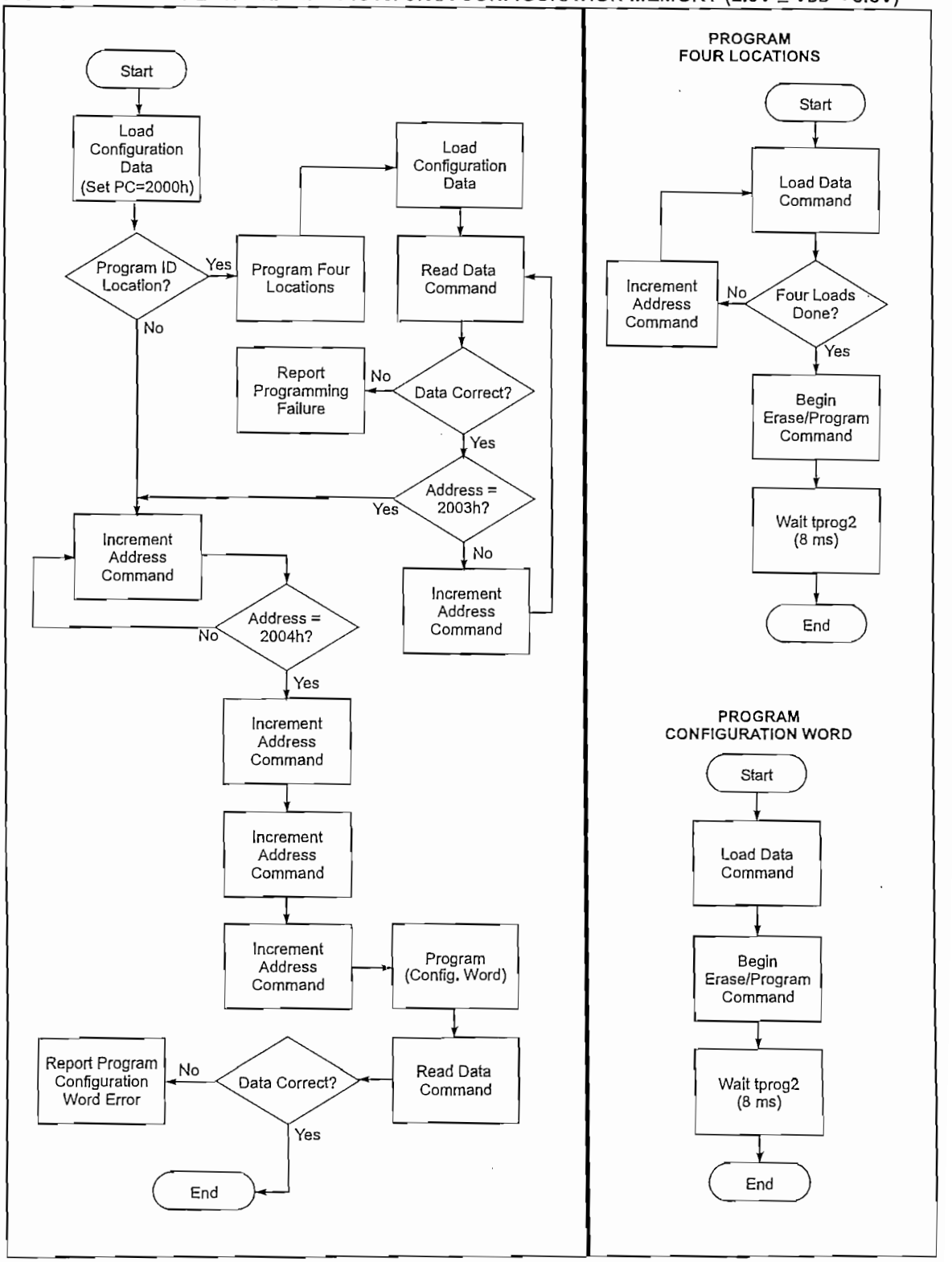
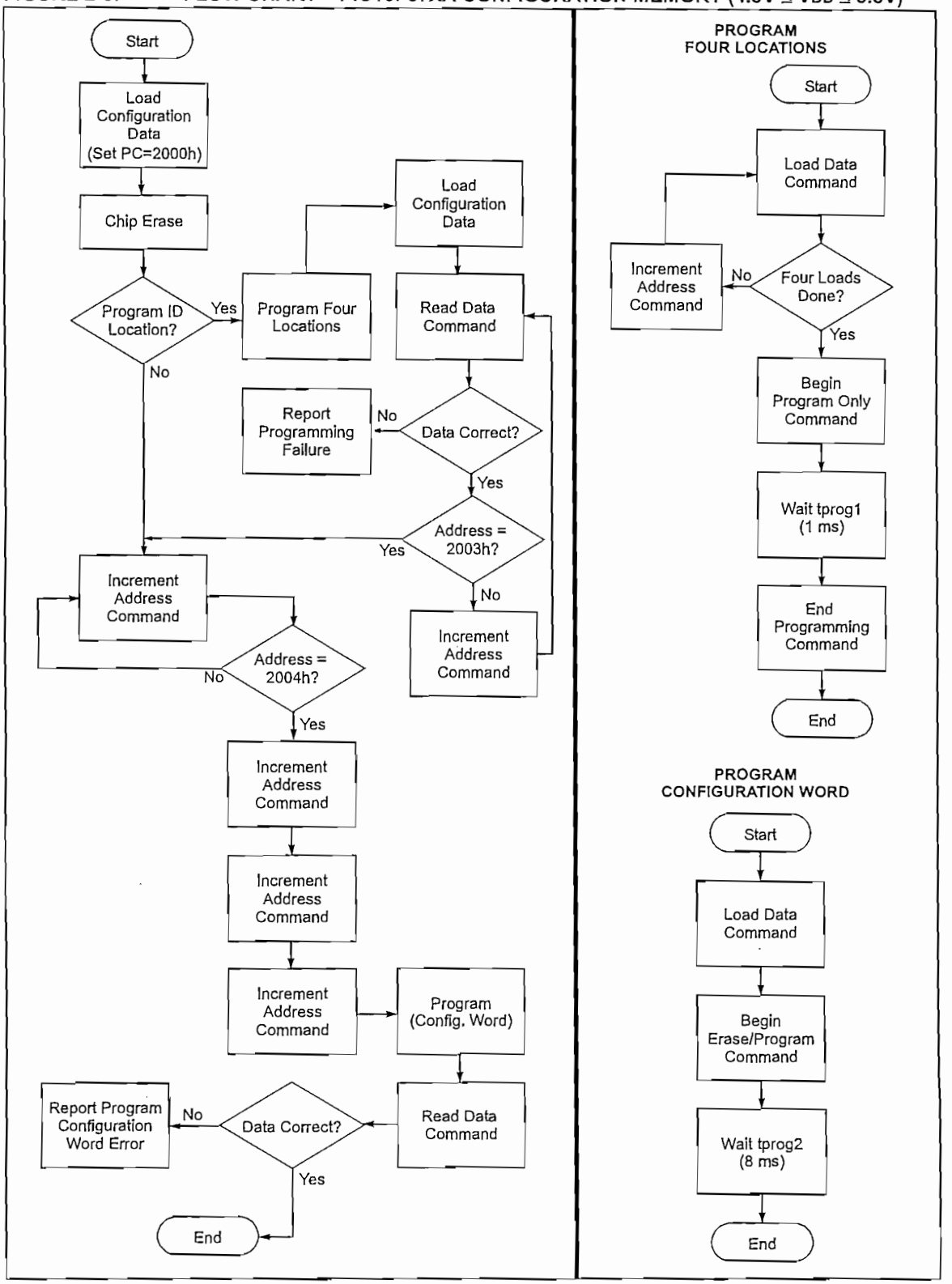


FIGURE 2-5: FLOW CHART – PIC16F87XA CONFIGURATION MEMORY (4.5V ≤ VDD ≤ 5.5V)



3.0 CONFIGURATION WORD

The PIC16F87XA has several configuration bits. These bits can be set (reads '0'), or left unchanged (reads '1'), to select various device configurations.

3.1 Device ID Word

The device ID word for the PIC16F87XA is located at 2006h.

TABLE 3-1: DEVICE ID VALUE

Device	Device ID Value	
	Dev	Rev
PIC16F873A	00 1110 0100	XXXX
PIC16F874A	00 1110 0110	XXXX
PIC16F876A	00 1110 0000	XXXX
PIC16F877A	00 1110 0010	XXXX

REGISTER 3-1: CONFIGURATION WORD REGISTER

R/P-1	U-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	U-1	U-1	R/P-1	R/P-1	R/P-1	R/P-1
CP	—	DEBUG	WRT1	WRT0	CPD	LVP	BOREN	—	—	PWRTEN	WDTEN	FOSC1	FOSC0
bit 13													bit 0

bit 13 **CP:** FLASH Program Memory Code Protection bit
(PIC16F877A/876A):
1 = Code protection off
0 = 0000h to 1FFFh code protected
(PIC16F874A/873A):
1 = Code protection off
0 = 0000h to 0FFFh code protected
1000h to 1FFFh wraps to 0000h to 0FFFh

bit 12 **Unimplemented:** Read as '1'

bit 11 **DEBUG:** Background Debugger Mode bit
1 = Background debugger functions not enabled
0 = Background debugger functional

bit 10-9 **WRT<1:0>:** FLASH Program Memory Write Enable bits
(PIC16F877A/876A):
11 = Write protection off
10 = 0000h to 00FFh write protected, 0100h to 1FFFh may be modified by EECON control
01 = 0000h to 07FFh write protected, 0800h to 1FFFh may be modified by EECON control
00 = 0000h to 0FFFh write protected, 1000h to 1FFFh may be modified by EECON control
(PIC16F874A/873A):
11 = Write protection off
10 = 0000h to 00FFh write protected, 0100h to 0FFFh may be modified by EECON control
01 = 0000h to 03FFh write protected, 0400h to 0FFFh may be modified by EECON control
00 = 0000h to 07FFh write protected, 0800h to 1FFFh may be modified by EECON control

bit 8 **CPD:** Data EE Memory Code Protection bit
1 = Code protection off
0 = Data EE memory code protected

bit 7 **LVP:** Low Voltage Programming Enable bit
1 = RB3/PGM pin has PGM function, low voltage programming enabled
0 = RB3 is digital I/O, HV on MCLR must be used for programming

bit 6 **BOREN:** Brown-out Reset Enable bit
1 = BOR enabled
0 = BOR disabled

bit 5-4 **Unimplemented:** Read as '1'

bit 3 **PWRTEN:** Power-up Timer Enable bit
1 = PWRT disabled
0 = PWRT enabled

bit 2 **WDTEN:** Watchdog Timer Enable bit
1 = WDT enabled
0 = WDT disabled

bit 1-0 **FOSC<1:0>:** Oscillator Selection bits
11 = RC oscillator
10 = HS oscillator
01 = XT oscillator
00 = LP oscillator

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '1'	
-n = Default value	1 = Bit is erased	0 = Bit is programmed	x = Bit is unknown

4.0 EMBEDDING CONFIGURATION WORD AND ID INFORMATION IN HEX FILE

To allow portability of code, the programmer is required to read the configuration word and ID locations from the HEX file when loading the HEX file. If configuration word information was not present in the HEX file, then a simple warning message may be issued. Similarly, while saving a HEX file, configuration word and ID information must be included. An option to not include this information may be provided.

Specifically for the PIC16F87XA, the EEPROM data memory should also be embedded in the HEX file (see Section 2.2).

Microchip Technology Inc. feels strongly that this feature is important for the benefit of the end customer.

5.0 CHECKSUM COMPUTATION

Checksum is calculated by reading the contents of the PIC16F87XA memory locations and adding up the opcodes up to the maximum user addressable location, e.g., 0x1FF for the PIC16F87XA. Any carry bits exceeding 16-bits are neglected. Finally, the configuration word (appropriately masked) is added to the checksum. Checksum computation for each member of the PIC16F87XA devices is shown in Table 5-1.

The checksum is calculated by summing the following:

- The contents of all program memory locations
- The configuration word, appropriately masked
- Masked ID locations (when applicable)

The Least Significant 16 bits of this sum are the checksum.

The following table describes how to calculate the checksum for each device. Note that the checksum calculation differs depending on the code protect setting. Since the program memory locations read out differently depending on the code protect setting, the table describes how to manipulate the actual program memory values to simulate the values that would be read from a protected device. When calculating a checksum by reading a device, the entire program memory can simply be read and summed. The configuration word and ID locations can always be read.

Note that some older devices have an additional value added in the checksum. This is to maintain compatibility with older device programmer checksums.

TABLE 5-1: CHECKSUM COMPUTATION

Device	Code Protect	Checksum*	Blank Value	25E6h at 0 and max address
PIC16F873A	OFF	SUM[0000:0FFF] + (CFGW & 2FCF)	1FCF	EB9D
	ON	(CFGW & 2FCF) + SUM_ID	4F9E	1B6C
PIC16F874A	OFF	SUM[0000:0FFF] + (CFGW & 2FCF)	1FCF	EB9D
	ON	(CFGW & 2FCF) + SUM_ID	4F9E	1B6C
PIC16F876A	OFF	SUM[0000:1FFF] + (CFGW & 2FCF)	0FCF	DB9D
	ON	(CFGW & 2FCF) + SUM_ID	1F9E	EB6C
PIC16F877A	OFF	SUM[0000:1FFF] + (CFGW & 2FCF)	0FCF	DB9D
	ON	(CFGW & 2FCF) + SUM_ID	1F9E	EB6C

Legend: CFGW = Configuration Word
SUM[a:b] = [Sum of locations a to b inclusive]
SUM_ID = ID locations masked by 0Fh then made into a 16-bit value with ID0 as the most significant nibble. For example, ID0 = 01h, ID1 = 02h, ID3 = 03h, ID4 = 04h, then SUM_ID = 1234h
*Checksum = [Sum of all the individual expressions] MODULO [FFFFh]
+ = Addition
& = Bitwise AND

6.0 PROGRAM/VERIFY MODE ELECTRICAL CHARACTERISTICS

TABLE 6-1: TIMING REQUIREMENTS FOR PROGRAM/VERIFY MODE

AC/DC CHARACTERISTICS POWER SUPPLY PINS	Standard Operating Procedure (unless otherwise stated) Operating temperature 0 ≤ TA ≤ +70°C Operating Voltage 2.0V ≤ VDD ≤ 5.5V					
Characteristics	**Sym**	**Min**	**Typ**	**Max**	**Units**	**Conditions/Comments**
General						
VDD level for Begin Erase/Program operations and EECON write of program memory	VDD	2.0		5.5	V	
VDD level for Begin Erase/Program operations and EECON write of data memory	VDD	2.0		5.5	V	
VDD level for Bulk Erase/Write, Chip Erase, and Begin Program operations, of program and data memory	VDD	4.5		5.5	V	
Begin Programming Only cycle time	tprog1	1			ms	Externally Timed
Begin Erase/Programming	tprog2		4	8	ms	Internally Timed
Chip Erase cycle time	tprog3		4	8	ms	Internally Timed
High voltage on $\overline{MCLR}$ and RA4/T0CKI for Test mode entry	VIHH	VDD + 3.5		13.5	V	
$\overline{MCLR}$ rise time (Vss to VHH) for Test mode entry	tVHHR			1.0	µs	
(RB6, RB7) input high level	VIH1	0.8 VDD			V	Schmitt Trigger input
(RB6, RB7) input low level	VIL1	0.2 VDD			V	Schmitt Trigger input
RB<7:4> setup time before $\overline{MCLR}$↑ (Test mode selection pattern setup time)	tset0	100			ns	
RB<7:4> hold time after $\overline{MCLR}$↑ (Test mode selection pattern setup time)	thld0	5			µs	
Serial Program/Verify						
Data in setup time before clock↓	tset1	100			ns	
Data in hold time after clock↓	thld1	100			ns	
Data input not driven to next clock input (delay required between command/data or command/command)	tdly1	1.0			µs	2.0V ≤ VDD < 4.5V
		100			ns	4.5V ≤ VDD ≤ 5.5V
Delay between clock↓ to clock↑ of next command or data	tdly2	1.0			µs	2.0V ≤ VDD < 4.5V
		100			ns	4.5V ≤ VDD ≤ 5.5V
Clock↑ to data out valid (during read data)	tdly3	80			ns	

FIGURE 6-1: LOAD DATA FOR USER PROGRAM MEMORY COMMAND (PROGRAM/VERIFY)

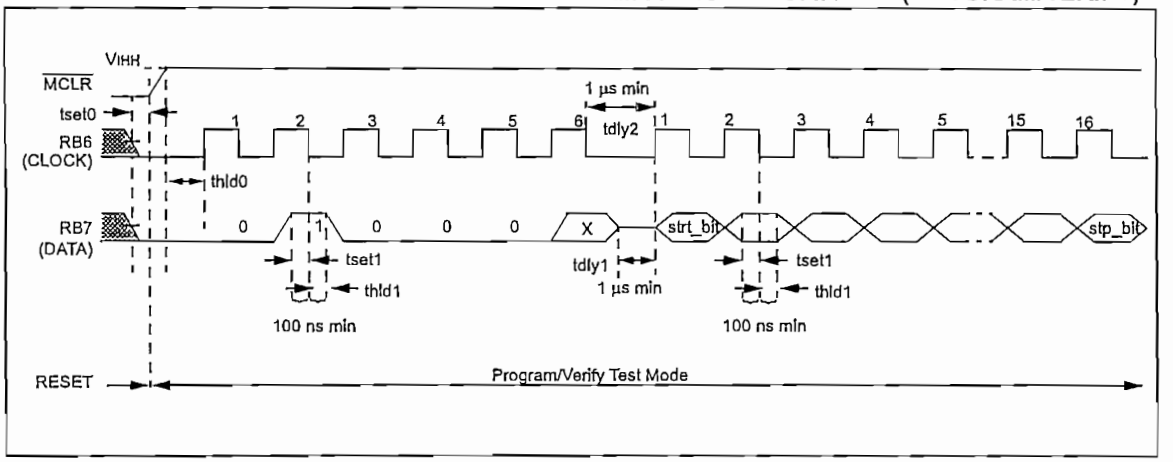


FIGURE 6-2: LOAD DATA FOR USER DATA MEMORY COMMAND (PROGRAM/VERIFY)

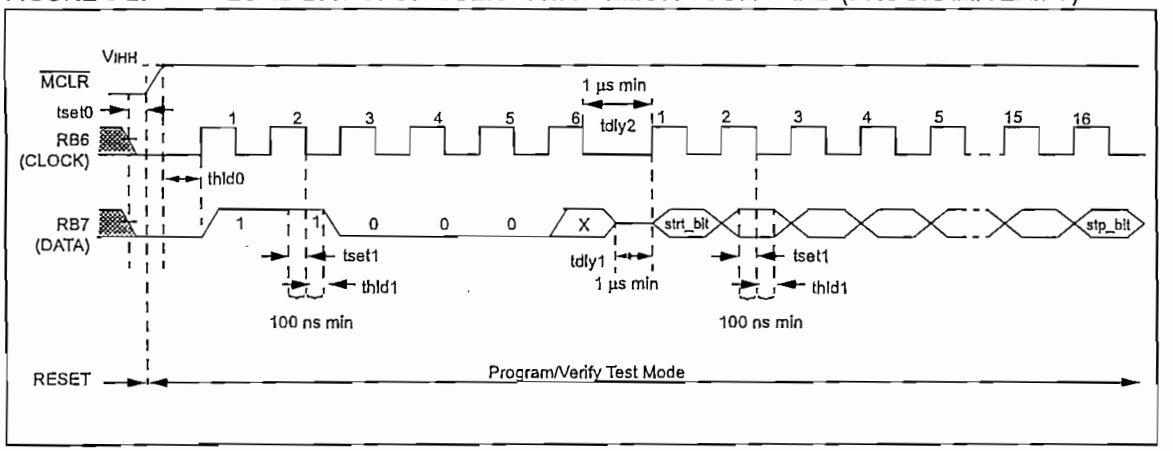


FIGURE 6-3: READ DATA FROM PROGRAM MEMORY COMMAND (PROGRAM/VERIFY)

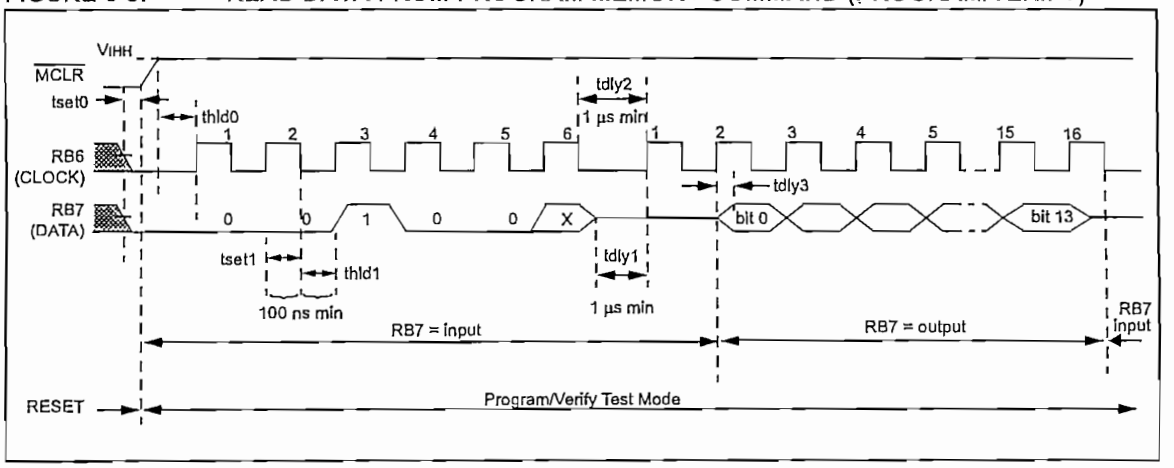


FIGURE 6-4: READ DATA FROM DATA MEMORY COMMAND (PROGRAM/VERIFY)

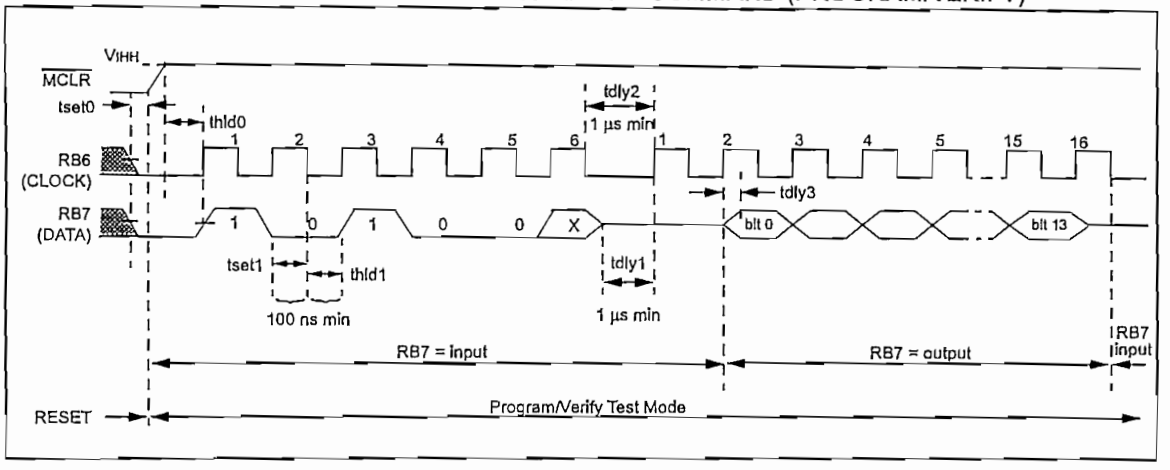


FIGURE 6-5: INCREMENT ADDRESS COMMAND (PROGRAM/VERIFY)

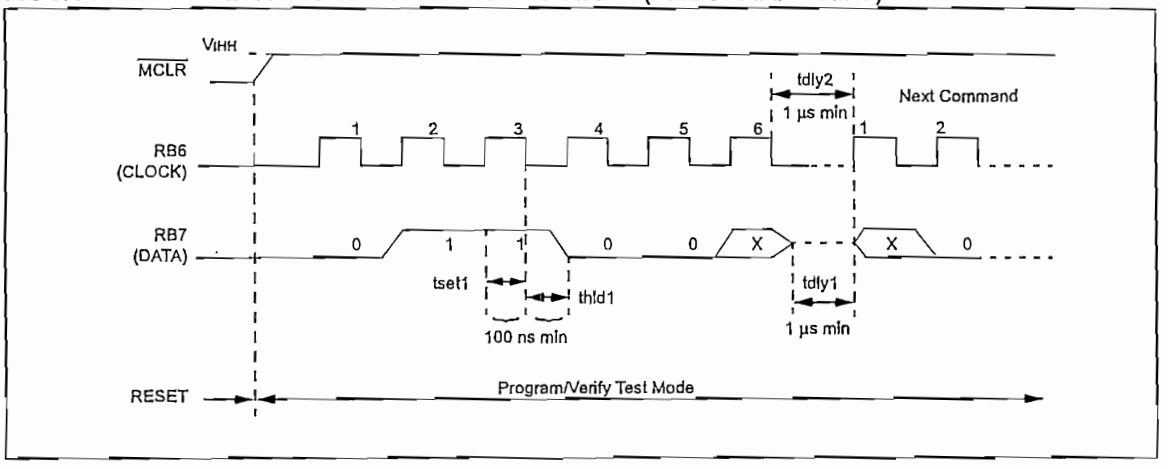


FIGURE 6-6: BEGIN ERASE/PROGRAMING COMMAND (PROGRAM/VERIFY)

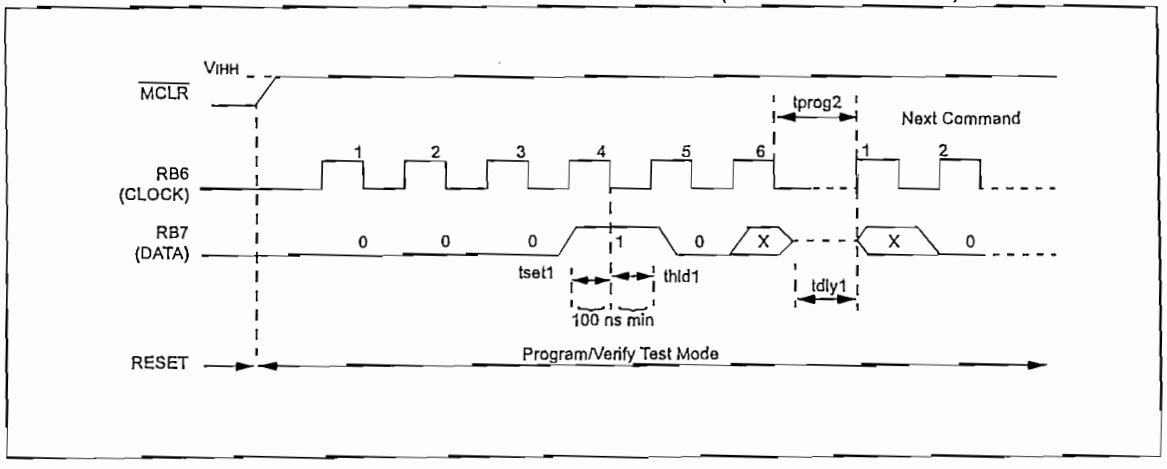


FIGURE 6-7: BEGIN PROGRAMING ONLY COMMAND (PROGRAM/VERIFY)

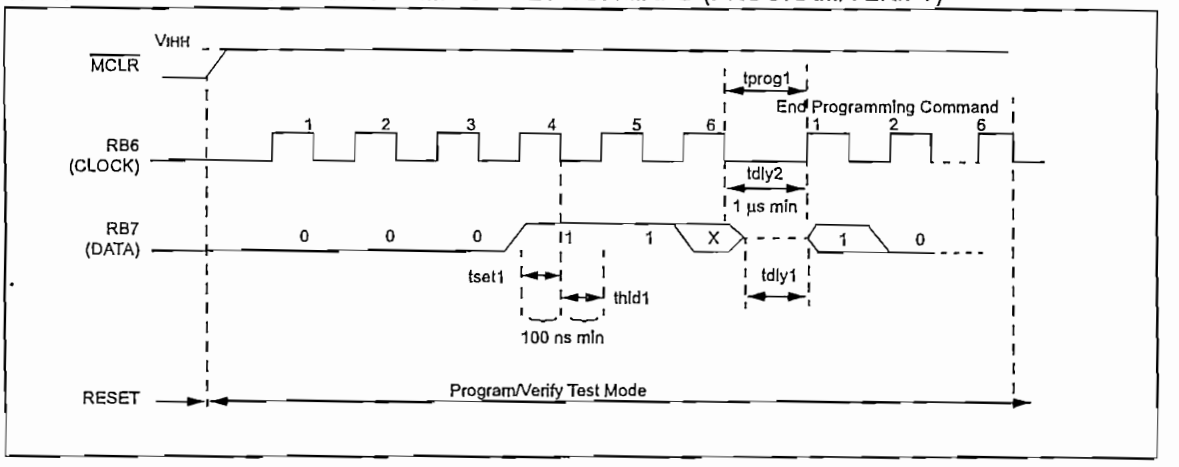


FIGURE 6-8: BULK ERASE PROGRAM MEMORY COMMAND (PROGRAM/VERIFY)

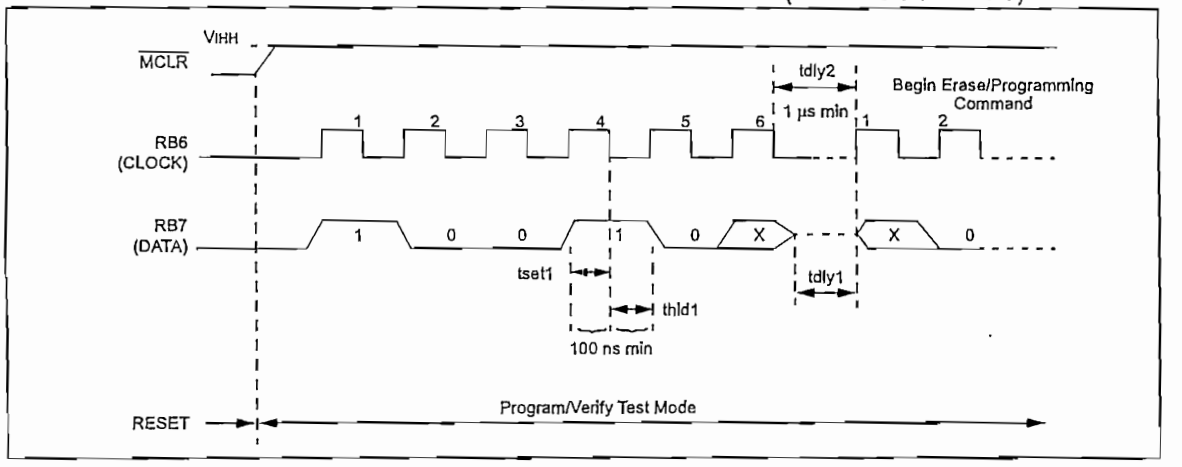


FIGURE 6-9: BULK ERASE DATA MEMORY COMMAND (PROGRAM/VERIFY)

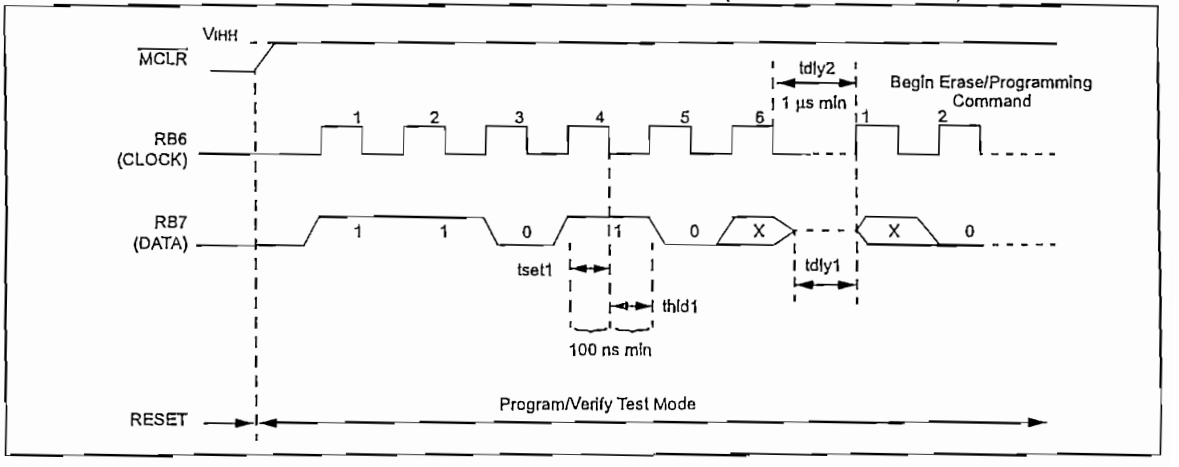
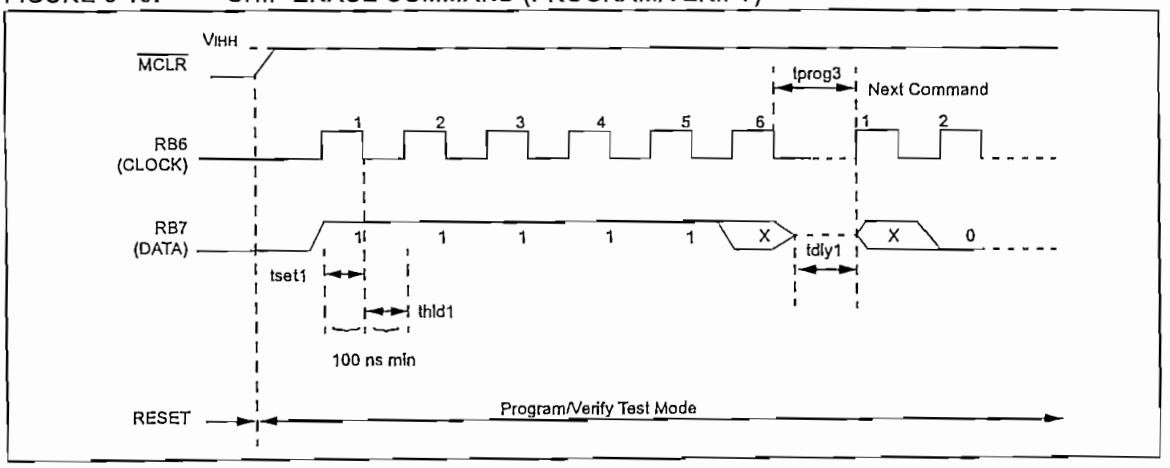


FIGURE 6-10: CHIP ERASE COMMAND (PROGRAM/VERIFY)



Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable".
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

DNV Certification, Inc. USA — DNV MSC The Netherlands Accredited by the RvA — ANSI-RAB QMS ACCREDITED — DNV — ISO 9001 / QS-9000 REGISTERED FIRM

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999 and Mountain View, California in March 2002. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200 Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

Rocky Mountain
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-4338

Atlanta
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770-640-0034 Fax: 770-640-0307

Boston
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848 Fax: 978-692-3821

Chicago
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

Dallas
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423 Fax: 972-818-2924

Detroit
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

Kokomo
2767 S. Albright Road
Kokomo, Indiana 46902
Tel: 765-864-8360 Fax: 765-864-8387

Los Angeles
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

San Jose
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950 Fax: 408-436-7955

Toronto
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699 Fax: 905-673-6509

ASIA/PACIFIC

Australia
Microchip Technology Australia Pty Ltd
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

China - Beijing
Microchip Technology Consulting (Shanghai) Co., Ltd., Beijing Liaison Office
Unit 915
Bei Hai Wan Tai Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

China - Chengdu
Microchip Technology Consulting (Shanghai) Co., Ltd., Chengdu Liaison Office
Rm. 2401, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-86766200 Fax: 86-28-86766599

China - Fuzhou
Microchip Technology Consulting (Shanghai) Co., Ltd., Fuzhou Liaison Office
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506 Fax: 86-591-7503521

China - Shanghai
Microchip Technology Consulting (Shanghai) Co., Ltd.
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

China - Shenzhen
Microchip Technology Consulting (Shanghai) Co., Ltd., Shenzhen Liaison Office
Rm. 1315, 13/F, Shenzhen Kerry Centre,
Renminnan Lu
Shenzhen 518001, China
Tel: 86-755-82350361 Fax: 86-755-82366086

China - Hong Kong SAR
Microchip Technology Hongkong Ltd.
Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

India
Microchip Technology Inc.
India Liaison Office
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan
Microchip Technology Japan K.K.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

Korea
Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Singapore
Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-6334-8870 Fax: 65-6334-8850

Taiwan
Microchip Technology (Barbados) Inc., Taiwan Branch
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Austria
Microchip Technology Austria GmbH
Durisolstrasse 2
A-4600 Wels
Austria
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark
Microchip Technology Nordic ApS
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

France
Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

Germany
Microchip Technology GmbH
Steinheilstrasse 10
D-85737 Ismaning, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Italy
Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

United Kingdom
Microchip Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

10/18/02

ANEXO D

Texas Instruments

Data sheet acquired from Harris Semiconductor
SCHS167D

November 1997 - Revised August 2003

CD54/74HC240, CD54/74HCT240, CD74HC241, CD54/74HCT241, CD54/74HC244, CD54/74HCT244

High-Speed CMOS Logic
Octal Buffer/Line Drivers, Three-State

Features

- HC/HCT240 Inverting
- HC/HCT241 Non-Inverting
- HC/HCT244 Non-Inverting
- Typical Propagation Delay = 8ns at V_{CC} = 5V, C_L = 15pF, T_A = 25°C for HC240
- Three-State Outputs
- Buffered Inputs
- High-Current Bus Driver Outputs
- Fanout (Over Temperature Range)
  - Standard Outputs . . . . . 10 LSTTL Loads
  - Bus Driver Outputs . . . . . 15 LSTTL Loads
- Wide Operating Temperature Range . . . -55°C to 125°C
- Balanced Propagation Delay and Transition Times
- Significant Power Reduction Compared to LSTTL Logic ICs
- HC Types
  - 2V to 6V Operation
  - High Noise Immunity: N_{IL} = 30%, N_{IH} = 30% of V_{CC} at V_{CC} = 5V
- HCT Types
  - 4.5V to 5.5V Operation
  - Direct LSTTL Input Logic Compatibility, V_{IL}= 0.8V (Max), V_{IH} = 2V (Min)
  - CMOS Input Compatibility, $I_I \leq 1\mu A$ at V_{OL}, V_{OH}

Description

The 'HC240 and 'HCT240 are inverting three-state buffers having two active-low output enables. The CD74HC241, 'HCT241, 'HC244 and 'HCT244 are non-inverting three-state buffers that differ only in that the 241 has one active-high and one active-low output enable, and the 244 has two active-low output enables. All three types have identical pinouts.

Ordering Information

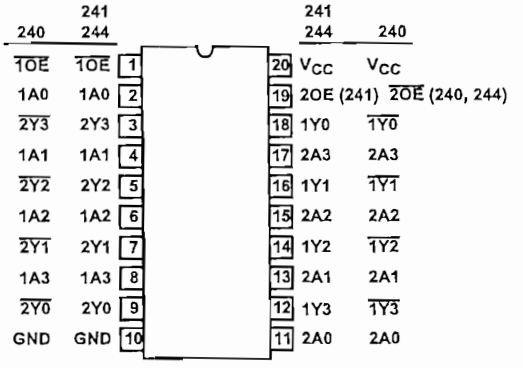
PART NUMBER	TEMP. RANGE (°C)	PACKAGE
CD54HC240F3A	-55 to 125	20 Ld CERDIP
CD54HC244F3A	-55 to 125	20 Ld CERDIP
CD54HCT240F3A	-55 to 125	20 Ld CERDIP
CD54HCT241F3A	-55 to 125	20 Ld CERDIP
CD54HCT244F3A	-55 to 125	20 Ld CERDIP
CD74HC240E	-55 to 125	20 Ld PDIP
CD74HC240M	-55 to 125	20 Ld SOIC
CD74HC240M96	-55 to 125	20 Ld SOIC
CD74HC241E	-55 to 125	20 Ld PDIP
CD74HC241M	-55 to 125	20 Ld SOIC
CD74HC241M96	-55 to 125	20 Ld SOIC
CD74HC244E	-55 to 125	20 Ld PDIP
CD74HC244M	-55 to 125	20 Ld SOIC
CD74HC244M96	-55 to 125	20 Ld SOIC
CD74HCT240E	-55 to 125	20 Ld PDIP
CD74HCT240M	-55 to 125	20 Ld SOIC
CD74HCT240M96	-55 to 125	20 Ld SOIC
CD74HCT240PW	-55 to 125	20 Ld TSSOP
CD74HCT240PWR	-55 to 125	20 Ld TSSOP
CD74HCT240PWT	-55 to 125	20 Ld TSSOP
CD74HCT241E	-55 to 125	20 Ld PDIP
CD74HCT241M96	-55 to 125	20 Ld SOIC
CD74HCT244E	-55 to 125	20 Ld PDIP
CD74HCT244M	-55 to 125	20 Ld SOIC
CD74HCT244M96	-55 to 125	20 Ld SOIC

NOTE: When ordering, use the entire part number. The suffixes 96 and R denote tape and reel. The suffix T denotes a small-quantity reel of 250.

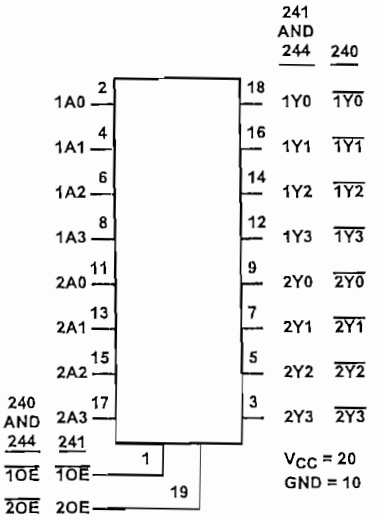
Pinout

CD54HC240, CD54HCT240, CD54HCT241,
CD54HC244, CD54HCT244
(CERDIP)
CD74HC240, CD74HC241, CD74HCT241,
CD74HC244, CD74HCT244
(PDIP, SOIC)
CD74HCT240,
(PDIP, SOIC, TSSOP)

TOP VIEW



Functional Diagram



Absolute Maximum Ratings

DC Supply Voltage, V_{CC} ..... -0.5V to 7V
DC Input Diode Current, I_{IK}
For $V_I < -0.5V$ or $V_I > V_{CC} + 0.5V$ ..... ±20mA
DC Output Diode Current, I_{OK}
For $V_O < -0.5V$ or $V_O > V_{CC} + 0.5V$ ..... ±20mA
DC Drain Current, per Output, I_O
For $-0.5V < V_O < V_{CC} + 0.5V$ ..... ±35mA
DC Output Source or Sink Current per Output Pin, I_O
For $V_O > -0.5V$ or $V_O < V_{CC} + 0.5V$ ..... ±25mA
DC V_{CC} or Ground Current, I_{CC} ..... ±70mA

Operating Conditions

Temperature Range (T_A) ..... -55°C to 125°C
Supply Voltage Range, V_{CC}
HC Types ..... 2V to 6V
HCT Types ..... 4.5V to 5.5V
DC Input or Output Voltage, V_I, V_O ..... 0V to V_{CC}
Input Rise and Fall Time
2V ..... 1000ns (Max)
4.5V ..... 500ns (Max)
6V ..... 400ns (Max)

Thermal Information

Thermal Resistance (Typical, Note 1)	θ_{JA}
E (PDIP) Package	69°C/W
M (SOIC) Package	58°C/W
PW (TSSOP) Package	83°C/W

Maximum Junction Temperature ..... 150°C
Maximum Storage Temperature Range ..... -65°C to 150°C
Maximum Lead Temperature (Soldering 10s) ..... 300°C
(SOIC - Lead Tips Only)

CAUTION: Stresses above those listed in "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress only rating and operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.

NOTE:

1. The package thermal impedance is calculated in accordance with JESD 51-7.

DC Electrical Specifications

PARAMETER	SYMBOL	TEST CONDITIONS		V_{CC} (V)	25°C			-40°C TO 85°C		-55°C TO 125°C		UNITS
		V_I (V)	I_O (mA)		MIN	TYP	MAX	MIN	MAX	MIN	MAX	
HC TYPES												
High Level Input Voltage	V_{IH}	-	-	2	1.5	-	-	1.5	-	1.5	-	V
				4.5	3.15	-	-	3.15	-	3.15	-	V
				6	4.2	-	-	4.2	-	4.2	-	V
Low Level Input Voltage	V_{IL}	-	-	2	-	-	0.5	-	0.5	-	0.5	V
				4.5	-	-	1.35	-	1.35	-	1.35	V
				6	-	-	1.8	-	1.8	-	1.8	V
High Level Output Voltage CMOS Loads	V_{OH}	V_{IH} or V_{IL}	-0.02	2	1.9	-	-	1.9	-	1.9	-	V
			-0.02	4.5	4.4	-	-	4.4	-	4.4	-	V
			-0.02	6	5.9	-	-	5.9	-	5.9	-	V
High Level Output Voltage TTL Loads			-6	4.5	3.98	-	-	3.84	-	3.7	-	V
			-7.8	6	5.48	-	-	5.34	-	5.2	-	V
Low Level Output Voltage CMOS Loads	V_{OL}	V_{IH} or V_{IL}	0.02	2	-	-	0.1	-	0.1	-	0.1	V
			0.02	4.5	-	-	0.1	-	0.1	-	0.1	V
			0.02	6	-	-	0.1	-	0.1	-	0.1	V
Low Level Output Voltage TTL Loads			6	4.5	-	-	0.26	-	0.33	-	0.4	V
			7.8	6	-	-	0.26	-	0.33	-	0.4	V
Input Leakage Current	I_I	V_{CC} or GND	-	6	-	-	±0.1	-	±1	-	±1	µA
Quiescent Device Current	I_{CC}	V_{CC} or GND	0	6	-	-	8	-	80	-	160	µA

DC Electrical Specifications (Continued)

PARAMETER	SYMBOL	TEST CONDITIONS		V_{CC} (V)	25°C			-40°C TO 85°C		-55°C TO 125°C		UNITS
		V_I (V)	I_O (mA)		MIN	TYP	MAX	MIN	MAX	MIN	MAX	
Three-State Leakage Current	I_{OZ}	V_{IL} or V_{IH}	-	6	-	-	±0.5	-	±0.5	-	±10	µA
HCT TYPES												
High Level Input Voltage	V_{IH}	-	-	4.5 to 5.5	2	-	-	2	-	2	-	V
Low Level Input Voltage	V_{IL}	-	-	4.5 to 5.5	-	-	0.8	-	0.8	-	0.8	V
High Level Output Voltage CMOS Loads	V_{OH}	V_{IH} or V_{IL}	-0.02	4.5	4.4	-	-	4.4	-	4.4	-	V
High Level Output Voltage TTL Loads			-6	4.5	3.98	-	-	3.84	-	3.7	-	V
Low Level Output Voltage CMOS Loads	V_{OL}	V_{IH} or V_{IL}	0.02	4.5	-	-	0.1	-	0.1	-	0.1	V
Low Level Output Voltage TTL Loads			6	4.5	-	-	0.26	-	0.33	-	0.4	V
Input Leakage Current	I_I	V_{CC} to GND	0	5.5	-	-	±0.1	-	±1	-	±1	µA
Quiescent Device Current	I_{CC}	V_{CC} or GND	0	5.5	-	-	8	-	80	-	160	µA
Additional Quiescent Device Current Per Input Pin: 1 Unit Load	ΔI_{CC} (Note 2)	V_{CC} -2.1	-	4.5 to 5.5	-	100	360	-	450	-	490	µA
Three-State Leakage Current	I_{OZ}	V_{IL} or V_{IH}	-	5.5	-	-	±0.5	-	±5	-	±10	µA

NOTE:

2. For dual-supply systems theoretical worst case (V_I = 2.4V, V_{CC} = 5.5V) specification is 1.8mA.

HCT Input Loading Table

INPUT	UNIT LOADS
HCT240	
nA0-A3	1.5
$\overline{1OE}$	0.7
$\overline{2OE}$	0.7
HCT241	
nA0-A3	0.7
$\overline{1OE}$	0.7
2OE	1.5
HCT244	
nA0-A3	0.7
$\overline{1OE}$	0.7
$\overline{2OE}$	0.7

NOTE: Unit Load is ΔI_{CC} limit specified in DC Electrical Specifications table, e.g., 360µA max at 25°C.

Switching Specifications $C_L = 50pF$, Input t_r, $t_f = 6ns$

PARAMETER	SYMBOL	TEST CONDITIONS	V_{CC} (V)	25°C			-40°C TO 85°C			-55°C TO 125°C			UNITS
				MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	
HC TYPES													
Propagation Delay Data to Outputs HC240	t_{PLH}, t_{PHL}	$C_L = 50pF$	2	-	-	100	-	-	125	-	-	150	ns
			4.5	-	-	20	-	-	25	-	-	30	ns
		$C_L = 15pF$	5	-	8	-	-	-	-	-	-	-	ns
		$C_L = 50pF$	6	-	-	17	-	-	21	-	-	26	ns
Data to Outputs HC241	t_{PLH}, t_{PHL}	$C_L = 50pF$	2	-	-	110	-	-	140	-	-	165	ns
			4.5	-	-	22	-	-	28	-	-	33	ns
		$C_L = 15pF$	5	-	9	-	-	-	-	-	-	-	ns
		$C_L = 50pF$	6	-	-	19	-	-	24	-	-	28	ns
Data to Outputs HC244	t_{PLH}, t_{PHL}	$C_L = 50pF$	2	-	-	110	-	-	140	-	-	165	ns
			4.5	-	-	22	-	-	28	-	-	33	ns
		$C_L = 15pF$	5	-	9	-	-	-	-	-	-	-	ns
		$C_L = 50pF$	6	-	-	19	-	-	24	-	-	28	ns
Output Enable and Disable Time	t_{THL}, t_{TLH}	$C_L = 50pF$	2	-	-	150	-	-	190	-	-	225	ns
			4.5	-	-	30	-	-	38	-	-	45	ns
			5	-	12	-	-	-	-	-	-	-	ns
			6	-	-	26	-	-	33	-	-	38	ns
Output Transition Time	t_{TLH}, t_{THL}	$C_L = 50pF$	2	-	-	60	-	-	75	-	-	90	ns
			4.5	-	-	12	-	-	15	-	-	18	ns
			6	-	-	10	-	-	13	-	-	15	ns
Input Capacitance	C_I	$C_L = 50pF$	-	10	-	10	-	-	10	-	-	10	pF
Three-State Output Capacitance	C_O	$C_L = 50pF$	-	-	-	20	-	-	20	-	-	20	pF
Power Dissipation Capacitance (Notes 3, 4)	C_{PD}	$C_L = 15pF$											
HC240			5	-	38	-	-	-	-	-	-	-	pF
HC241			5	-	34	-	-	-	-	-	-	-	pF
HC244			5	-	46	-	-	-	-	-	-	-	pF
HCT TYPES													
Propagation Delay Data to Outputs HCT240	t_{PHL}, t_{PLH}	$C_L = 50pF$	4.5	-	-	22	-	-	28	-	-	33	ns
		$C_L = 15pF$	5	-	9	-	-	-	-	-	-	-	ns
Data to Outputs HCT241	t_{PHL}, t_{PLH}	$C_L = 50pF$	4.5	-	-	25	-	-	31	-	-	38	ns
		$C_L = 15pF$	5	-	10	-	-	-	-	-	-	-	ns
Data to Outputs HCT244	t_{PHL}, t_{PLH}	$C_L = 50pF$	4.5	-	-	25	-	-	31	-	-	38	ns
		$C_L = 15pF$	5	-	10	-	-	-	-	-	-	-	ns

Switching Specifications $C_L = 50pF$, Input t_r, $t_f = 6ns$ **(Continued)**

PARAMETER	SYMBOL	TEST CONDITIONS	V_{CC} (V)	25°C			-40°C TO 85°C			-55°C TO 125°C			UNITS
				MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	
Output Enable and Disable Times	t_{TLH}, t_{THL}	$C_L = 50pF$	4.5	-	-	30	-	-	38	-	-	45	ns
Output Transition Time	t_{THL}, t_{TLH}	$C_L = 50pF$	4.5	-	-	12	-	-	15	-	-	18	ns
Input Capacitance	C_I	$C_L = 50pF$	-	10	-	10	-	-	10	-	-	10	pF
Power Dissipation Capacitance (*Notes* 3, 4)	C_{PD}												
HCT240		-	5	-	40	-	-	-	-	-	-	-	pF
HCT241		-	5	-	38	-	-	-	-	-	-	-	pF
HCT244		-	5	-	40	-	-	-	-	-	-	-	pF

NOTES:

3. C_{PD} is used to determine the dynamic power consumption, per channel.
4. $P_D = V_{CC}^2 f_i (C_{PD} + C_L)$ where f_i = Input Frequency, f_O = Output Frequency, C_L = Output Load Capacitance, V_{CC} = Supply Voltage.

Test Circuits and Waveforms

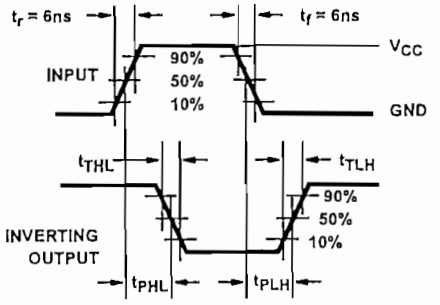


FIGURE 1. HC TRANSITION TIMES AND PROPAGATION DELAY TIMES, COMBINATION LOGIC

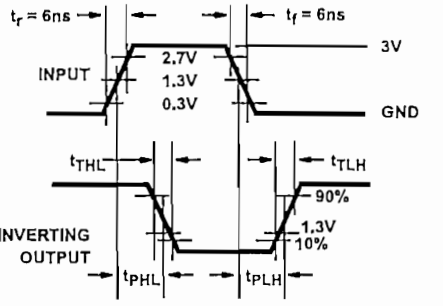


FIGURE 2. HCT TRANSITION TIMES AND PROPAGATION DELAY TIMES, COMBINATION LOGIC

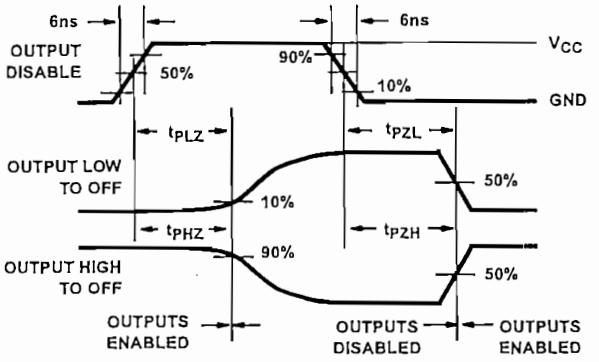


FIGURE 3. HC THREE-STATE PROPAGATION DELAY WAVEFORM

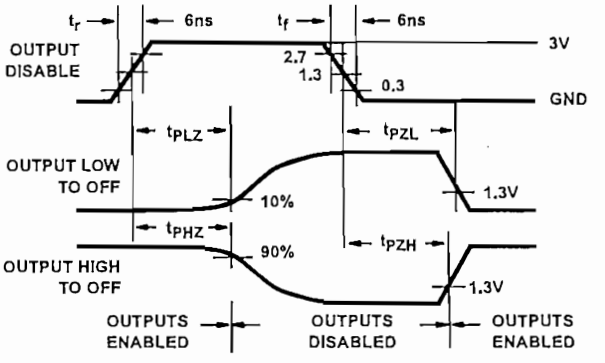
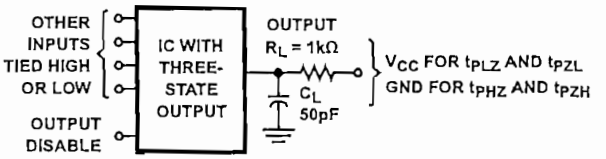


FIGURE 4. HCT THREE-STATE PROPAGATION DELAY WAVEFORM

Test Circuits and Waveforms (Continued)



NOTE: Open drain waveforms t_{PLZ} and t_{PZL} are the same as those for three-state shown on the left. The test circuit is Output R_L = 1kΩ to V_{CC}, C_L = 50pF.

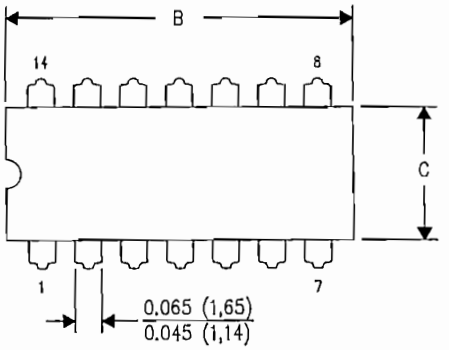
FIGURE 5. HC AND HCT THREE-STATE PROPAGATION DELAY TEST CIRCUIT

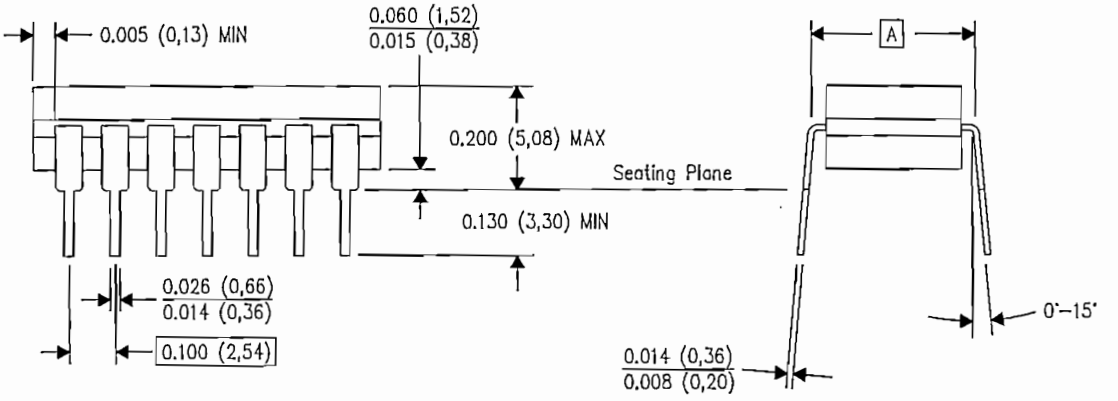
J (R–GDIP–T**)

14 LEADS SHOWN

CERAMIC DUAL IN–LINE PACKAGE

DIM \ PINS **	14	16	18	20
A	0.300 (7,62) BSC	0.300 (7,62) BSC	0.300 (7,62) BSC	0.300 (7,62) BSC
B MAX	0.785 (19,94)	.840 (21,34)	0.960 (24,38)	1.060 (26,92)
B MIN	—	—	—	—
C MAX	0.300 (7,62)	0.300 (7,62)	0.310 (7,87)	0.300 (7,62)
C MIN	0.245 (6,22)	0.245 (6,22)	0.220 (5,59)	0.245 (6,22)





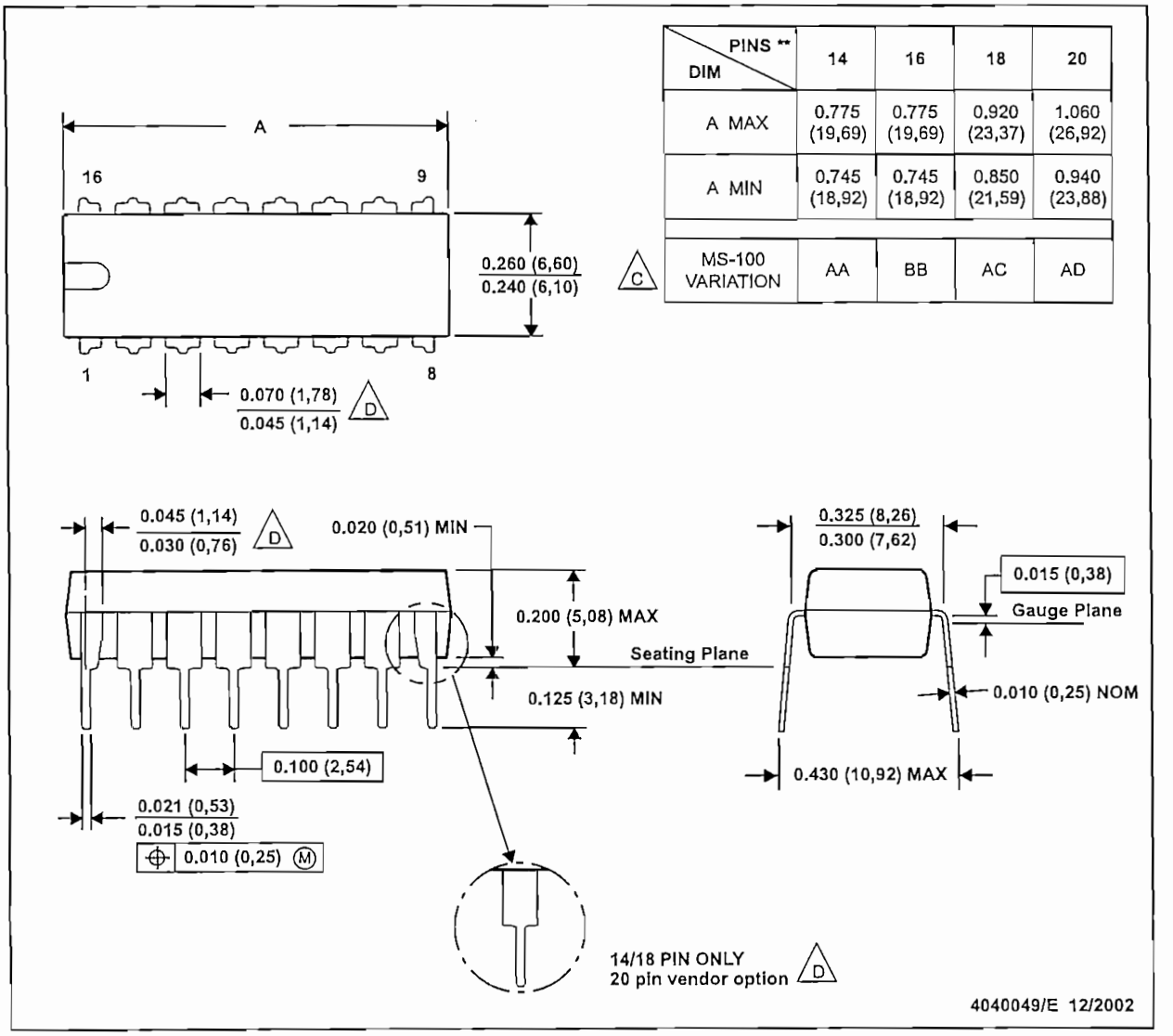
4040083/F 03/03

NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. This package is hermetically sealed with a ceramic lid using glass frit.
D. Index point is provided on cap for terminal identification only on press ceramic glass frit seal only.
E. Falls within MIL STD 1835 GDIP1–T14, GDIP1–T16, GDIP1–T18 and GDIP1–T20.

MECHANICAL

N (R-PDIP-T**) PLASTIC DUAL-IN-LINE PACKAGE

16 PINS SHOWN



DIM \ PINS **	14	16	18	20
A MAX	0.775 (19,69)	0.775 (19,69)	0.920 (23,37)	1.060 (26,92)
A MIN	0.745 (18,92)	0.745 (18,92)	0.850 (21,59)	0.940 (23,88)
MS-100 VARIATION	AA	BB	AC	AD

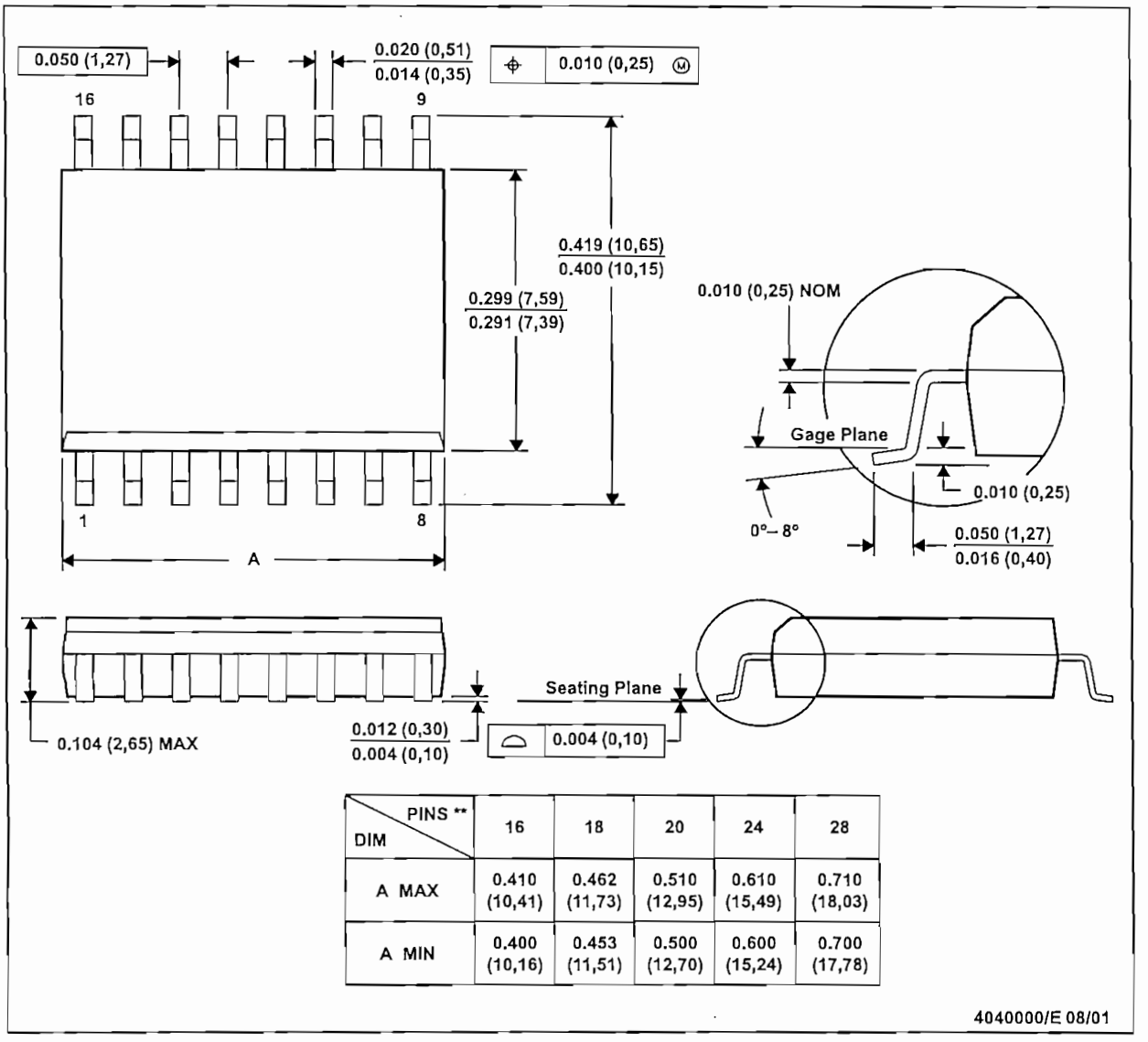
NOTES: A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Falls within JEDEC MS-001, except 18 and 20 pin minimum body length (Dim A).
D. The 20 pin end lead shoulder width is a vendor option, either half or full width.

MSOI003E – JANUARY 1995 – REVISED SEPTEMBER 2001

DW (R-PDSO-G**)

PLASTIC SMALL-OUTLINE PACKAGE

16 PINS SHOWN



DIM \ PINS **	16	18	20	24	28
A MAX	0.410 (10,41)	0.462 (11,73)	0.510 (12,95)	0.610 (15,49)	0.710 (18,03)
A MIN	0.400 (10,16)	0.453 (11,51)	0.500 (12,70)	0.600 (15,24)	0.700 (17,78)

4040000/E 08/01

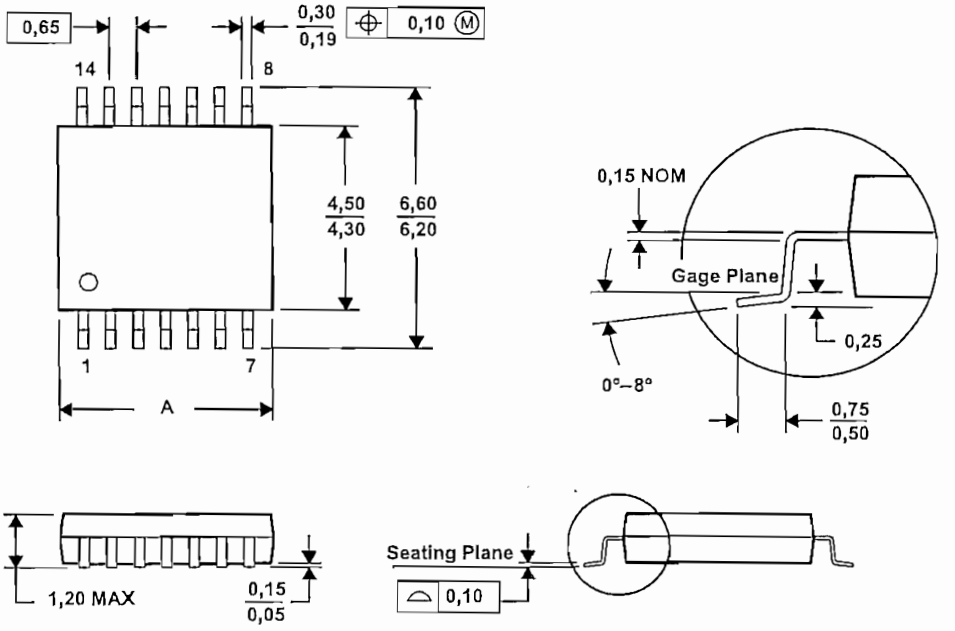
NOTES: A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Body dimensions do not include mold flash or protrusion not to exceed 0.006 (0,15).
D. Falls within JEDEC MS-013

MTSS001C – JANUARY 1995 – REVISED FEBRUARY 1999

PW (R-PDSO-G**)

PLASTIC SMALL-OUTLINE PACKAGE

14 PINS SHOWN



DIM \ PINS **	8	14	16	20	24	28
A MAX	3,10	5,10	5,10	6,60	7,90	9,80
A MIN	2,90	4,90	4,90	6,40	7,70	9,60

4040064/F 01/97

NOTES: A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Body dimensions do not include mold flash or protrusion not to exceed 0,15.
D. Falls within JEDEC MO-153

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

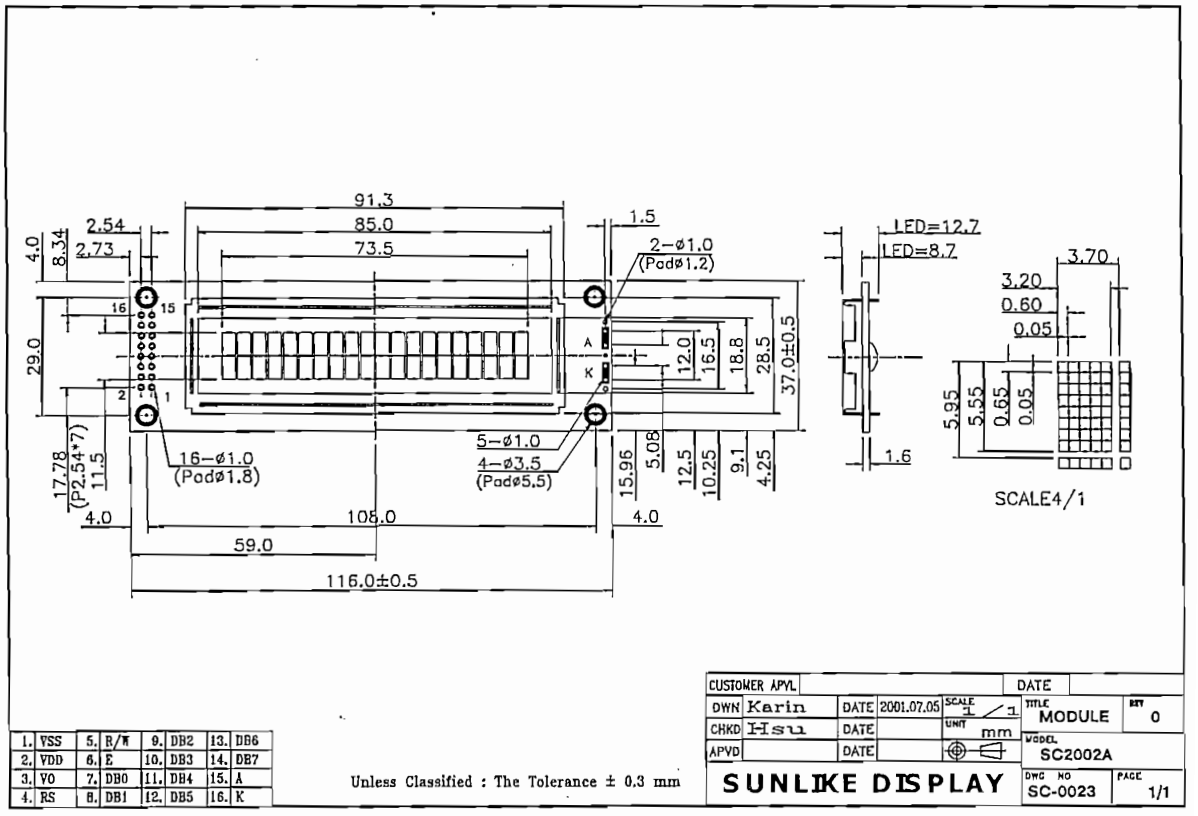
Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		Applications	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265

ANEXO E

LCD 2x20



1.	VSS	5.	R/W	9.	DB2	13.	DB6
2.	VDD	6.	E	10.	DB3	14.	DB7
3.	VO	7.	DB0	11.	DB4	15.	A
4.	RS	8.	DB1	12.	DB5	16.	K

Unless Classified : The Tolerance ± 0.3 mm

CUSTOMER APVL			DATE		
DWN	Karin	DATE	2001.07.05	SCALE 1/1	TITLE MODULE / REV 0
CHKD	Hsu	DATE		UNIT mm	MODEL SC2002A
APVD		DATE		SUNLIKE DISPLAY	DWG NO SC-0023 / PAGE 1/1

ABSOLUTE MAXIMUM RATING

(1) Electrical Absolute Ratings

Item	Symbol	Min.	Max.	Unit	Note
Power Supply for Logic	V_{DD}-V_{SS}	-0.3	7.0	Volt	
Power Supply for LCD	V_{DD}-V_O	-0.3	12.0	Volt	
Input Voltage	V_I	-0.3	V_{DD}	Volt	
LED Power Dissipation	P_{AD}	-	1.5	W	
LED Forward current	I_{AF}	-	315	mA	
LED Reverse Voltage	V_R	-	8	V	

(2) Environmental Absolute Maximum Ratings

Item	Normal Temperature				Wide Temperature			
	Operating		Storage		Operating		Storage	
	Min,	Max.	Min,	Max.	Min,	Max.	Min,	Max.
Ambient Temperature	0	+50	-20	+70	-20	+70	-30	+80
Humidity(without condensation)	Note 2,4		Note 3,5		Note 4,5		Note 4,6	

Note 2 Ta 50 : 80% RH max
Ta>50 : Absolute humidity must be lower than the humidity of 85%RH at 50

Note 3 Ta at -20 will be<48hrs at 70 will be <120hrs when humidity is higher than 70%.

Note 4 Background color changes slightly depending on ambient temperature. This phenomenon is reversible.

Note 5 Ta 70 : 75RH max
Ta>70 : absolute humidity must be lower than the humidity of 75%RH at 70

Note 6 Ta at -30 will be <48hrs, at 80 will be <120hrs when humidity is higher than 70%.

ELECTRICAL CHARACTERISTICS

Item	Symbol	Condition	Min.	Typ	Max.	Unit	note
Power Supply for Logic	V_{DD}-V_{SS}	-	4.5	5.0	5.5	Volt	
Input Voltage	V_{IL}	L level	0	-	0.6	Volt	
	V_{IH}	H level	2.2	-	V_{DD}	Volt	
LCM Recommend LCD Module Driving Voltage	V_{DD} –V_O	Ta 0	-	-	-	Volt	
		Ta 25	5.2	6.5	7.0		
		Ta 50	-	-	-		
Power Supply Current for LCM	I_{DD}	V_{DD} =5.0V V_{DD}-V_O =4.5V	-	2.0	3.0	mA	
LED For Ward Doltage	V_F	If 210 mA	-	4.2	4.6	Volt	
LED For Ward Current	I_F	-	-	210	-	mA	
LED Reverse Current	I_R	VR=8V	-	-	0.2	mA	

OPTICAL CHARACTERISTICS

Item	Symbol	Condition	Min.	Typ	Max.	Unit	note
Viewing angle range	f(12 o'clock)	When Cr 1.4	-	10	-	Degree	9,10
	b(6 o'clock)		-	30	-		
	l(9 o'clock)		-	30	-		
	r(3 o'clock)		-	30	-		
Rise Time	Tr	V_{DD}-V_O =4.5V Ta=25	-	200		mS	
Fall Time	Tf		-	250			
Frame frequency	Frm		-	64	-	Hz	8,10
Contrast	Cr		-	3.0	-		7
The Brightness Of Backlight	L	IF=210 mA	110	170	-	cd/	
Peak Emission Wavelength	P		-	570	-	nm	

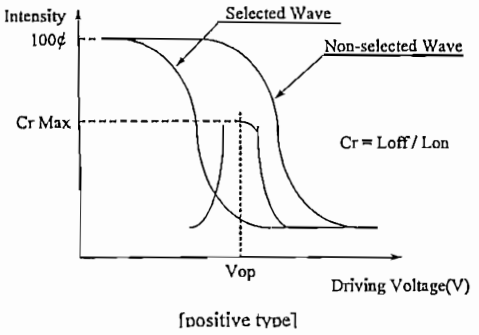
MECHANICAL SPECIFICATION

ITEM	DESCRIPTION
Product No.	
Module Size	116.0(W)×37.0(H)×8.8(LED=12.7) max(D)
Dot Size	0.60 (W)mm×0.65(H)mm
Dot Pitch	0.65(W)mm×0.70(H)mm
Display Format	20 characters (W)×2 lines (H)
Duty Ratio	1/16 Duty
Controller	KS0066 or Equivalent

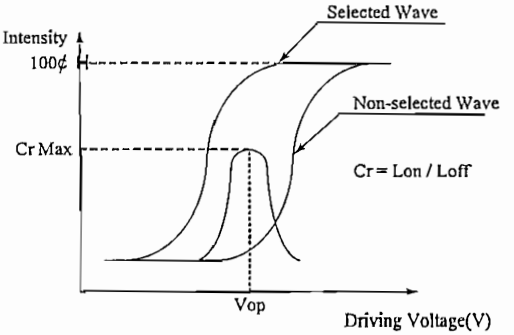
INTERFACE PIN ASSIGNMENT

Pin No.	Pin Out	Level	Description
1	VSS	0V	Power Supply Ground
2	VDD	5V	Power Supply Voltage
3	Vo	---	Contrast Adj
4	RS	H/L	Register Select
5	R/W	H/L	Read / Write
6	E	H,H L	Enable Signal
7	DB0	H/L	Data Bit 0
8	DB1	H/L	Data Bit 1
9	DB2	H/L	Data Bit 2
10	DB3	H/L	Data Bit 3
11	DB4	H/L	Data Bit 4
12	DB5	H/L	Data Bit 5
13	DB6	H/L	Data Bit 6
14	DB7	H/L	Data Bit 7
15	A	4.2V	LED Power Supply ( )
16	K	0V	LED Power Supply ( )

[Note 7] Definition of Operation Voltage (Vop)

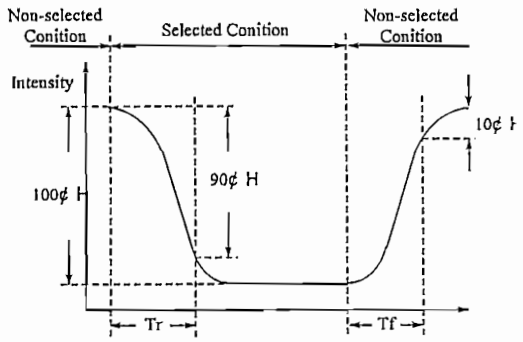


[positive type]

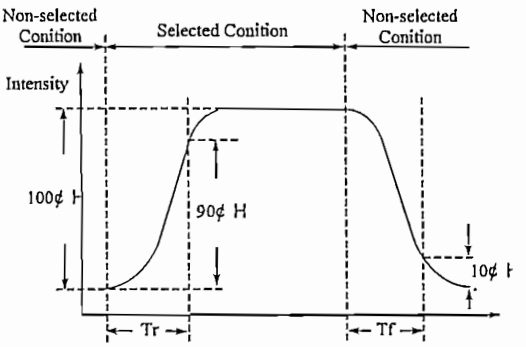


[Negative type]

[Note 8] Definition of Response Time (Tr, Tf)



[positive type]



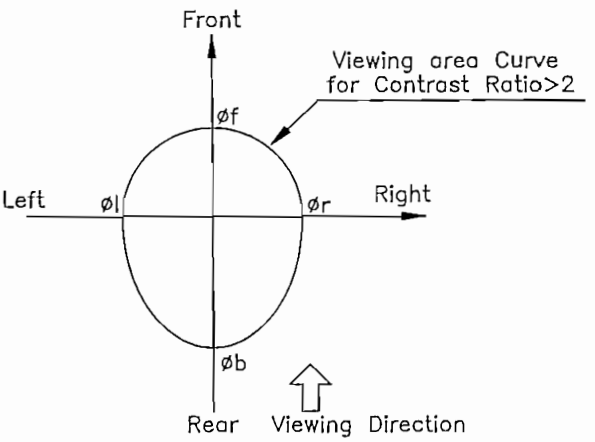
[Negative type]

Conditions:

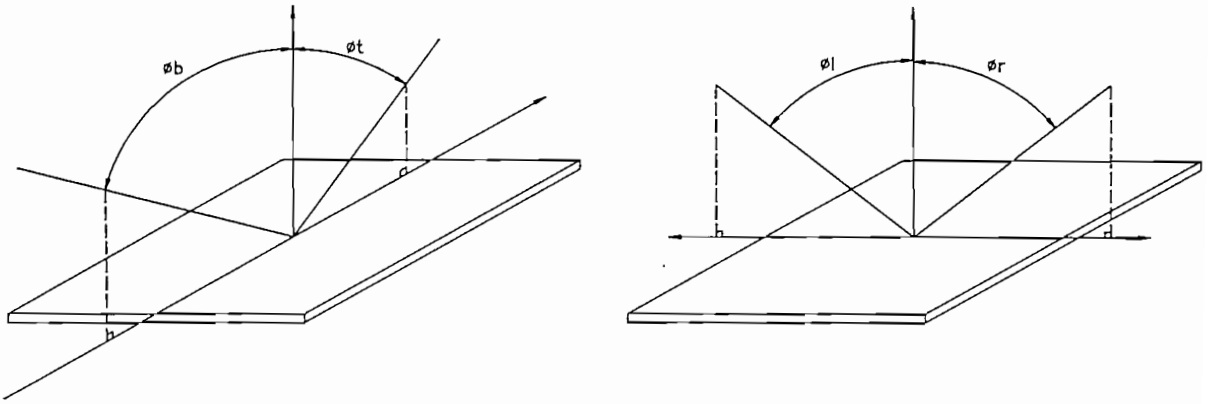
Operating Voltage : Vop **Viewing Angle(θ, φ): 0° , 0°**

Frame Frequency : 64 Hz **Driving Wave form : 1/N duty, 1/a bias**

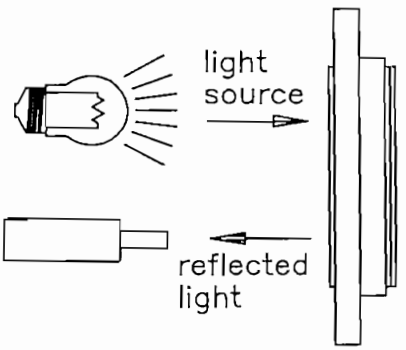
[Note 9] Definition of Viewing Direction



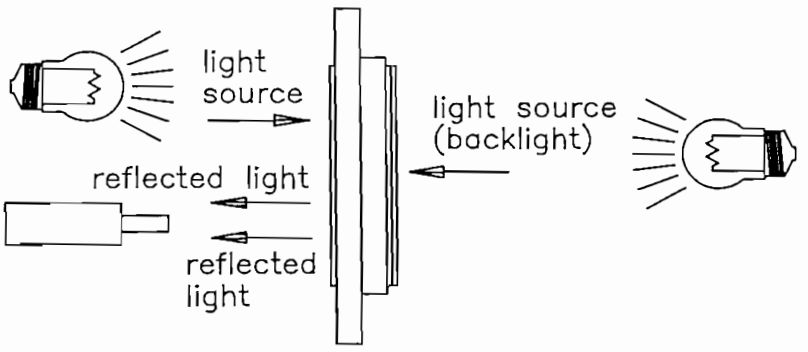
[Note 10] Definition of viewing angle



[Note 11] Description of Measuring Equipment



Reflective type



Transflective type