

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES

"CUAUTITLAN"

10  
20/1



INTERFAZ ENTRE UN SISTEMA DE COMPUTO DE  
PROPOSITO GENERAL Y UN SISTEMA DE  
DESARROLLO DE MICROPROCESADORES

T E S I S

QUE PARA OBTENER EL TITULO DE;  
INGENIERO MECANICO ELECTRICISTA  
P R E S E N T A N

MARIA ALICIA CISNEROS PATIÑO

JORGE ALBERTO LIMON JIMENEZ

DIRECTOR DE TESIS: M. EN C MOISES MOSHINSKY AGUILAR  
MEXICO 1984



Universidad Nacional  
Autónoma de México



**UNAM – Dirección General de Bibliotecas**  
**Tesis Digitales**  
**Restricciones de uso**

**DERECHOS RESERVADOS ©**  
**PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL**

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

	página
1 <u>INTRODUCCION</u>	
1.1 <u>ANTECEDENTES</u>	
1.1.1 Definición del problema	1
1.1.2 Un sistema de cómputo visto como periférico	3
1.2 <u>MICROCOMPUTADORA INTELLEC-MDS</u>	
1.2.1 Descripción general	6
1.2.2 Conformación	7
1.3 <u>COMPUTADORA HEWLETT-PACKARD 3000</u>	
1.3.1 Generalidades	11
1.3.2 Nociones sobre el sistema operativo	13
1.3.3 Comunicación al exterior	17
1.4 <u>INTERCOMUNICACION ENTRE SISTEMAS</u>	
1.4.1 Conexión física	18
1.4.2 Control de la comunicación	20
2 <u>COMUNICACION A DISTANCIA</u>	
2.1 <u>TIPOS DE COMUNICACION</u>	
2.1.1 Transmisiones en serie y paralelo	21
2.1.2 Generalidades sobre comunicación síncrona y asíncrona	23
2.2 <u>LA INTERFAZ EIA-RS-232-C</u>	
2.2.1 Aspectos generales	30
2.2.2 Definición de funciones EIA	32
2.2.3 Conexiones de la interfaz EIA	38
2.3 <u>EJEMPLOS DE COMUNICACION A DISTANCIA</u>	
2.3.1 La unidad moduladora demoduladora (modem)	39
2.3.2 Conexión mediante modems	42
3 <u>DISEÑO DE LA INTERFAZ EN LA MICROCOMPUTADORA</u>	
3.1 <u>ARQUITECTURA DE LA MICROCOMPUTADORA</u>	
3.1.1 Diagrama de bloques del sistema	45

3.1.2	El microprocesador 8080	51
3.1.3	Interfaz entre el microprocesador y el exterior	57
3.2	<u>EL TRANSMISOR/RECEPTOR UNIVERSAL SINCRONO/ASINCRONO</u>	
3.2.1	Formatos de comunicación	61
3.2.2	Configuración interna del USART y señales de interfaz	62
3.2.3	Selección de modo y manejo de datos	70
3.3	<u>SUMINISTRO DE UNA INTERFAZ DE ENTRADA/SALIDA</u>	
3.3.1	Base de tiempo	75
3.3.2	Líneas referidas al CPU	83
3.3.3	Líneas referidas a la interfaz	93
3.3.4	La tarjeta de interfaz	99
3.4	<u>PROGRAMACION DE LA INTERFAZ</u>	
3.4.1	Instrucciones del microprocesador 8080	104
3.4.2	Selección de formato de comunicación	105
3.4.3	Subrutinas de control de la interfaz	108
4	<u>ENLACE DE LOS SISTEMAS</u>	
4.1	<u>CONEXIONES FINALES</u>	
4.1.1	Conexiones en la HP 3000	114
4.1.2	Conexiones en la microcomputadora	117
4.2	<u>CONTROL DE LA COMUNICACION</u>	
4.2.1	Configuración del puerto en la HP 3000	120
4.2.2	Programa de control de la interfaz	121

4.3	<u>TRANSFERENCIA DE ARCHIVOS</u>	
4.3.1	El sistema operativo del Intellec-MDS	128
4.3.2	Interfaz entre el ensamblador 8080 y el sistema operativo	135
4.3.3	El usuario de la interfaz entre los sistemas	142
5	<u>CONCLUSIONES</u>	145
6	<u>APENDICES</u>	
	APENDICE A - Instrucciones del 8080.	147
	APENDICE B - Código ASCII	166
	APENDICE C - Respuesta en el tiempo de los canales de datos y direcciones del 8080 en el momento de un "reset".	168
	APENDICE D - Subrutina de control del USART (inicialización, entrada y salida).	169
	APENDICE E - Programa de control de la interfaz.	173
	APENDICE F - Configuración del puerto de la computadora HP 3000.	185
	<u>INDICE DE FIGURAS</u>	190
	<u>BIBLIOGRAFIA</u>	193

\* \* \* \* \*

# I N T R O D U C C I O N

### 1.1.1 Definición del problema.

En la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, existen actualmente un sistema monousuario de desarrollo de microprocesadores y una computadora de propósito general, esta última con capacidad para varias terminales.

Un sistema de desarrollo de microprocesadores (SDM) es una herramienta poderosa y económica, pero el hecho de ser un sistema monousuario plantea serias limitaciones respecto al tiempo real aprovechado. En las condiciones actuales de la Universidad, se estima que en un futuro cercano el sistema quedará saturado debido a la creciente demanda de usuarios. Al respecto se ha notado que gran parte del tiempo empleado por el usuario frente al sistema se utiliza en la edición de programas.

Sabiendo que se cuenta con una computadora que puede atender a varios usuarios a la vez, surge la idea de llevar a cabo esta tarea fuera del sistema de microprocesadores. De este modo, el interesado podría ocurrir a cualquier terminal de la computadora de propósito general y generar archivos fuente. Cuando disponga de tiempo en el SDM, sólo se limitaría a transferirlos a éste para continuar con el proceso de desarrollo.

La finalidad de este trabajo será diseñar y construir una interfaz entre la computadora de propósito general y el sistema de desarrollo de microprocesadores (figura 1.1), con lo cual será posible posteriormente efectuar las transferencias de archivos.

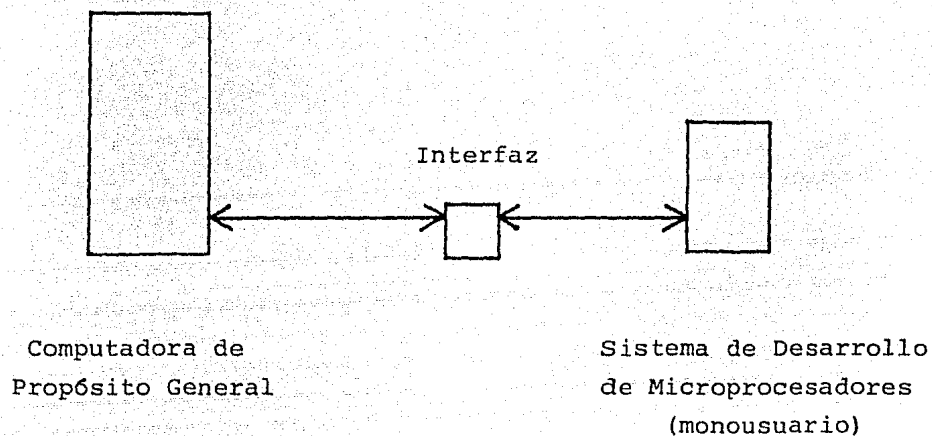


Figura 1.1 Los dos sistemas de computación existentes en la Universidad Autónoma Metropolitana comunicándose entre sí mediante una interfaz.



### 1.1.2 Un sistema de cómputo visto como periférico. 3

Si bien cada sistema posee su propia filosofía operativa, la interfaz es factible de realizarse, pues tanto la computadora de propósito general como el sistema de desarrollo de microprocesadores se apegan a las normas internacionales de comunicaciones.

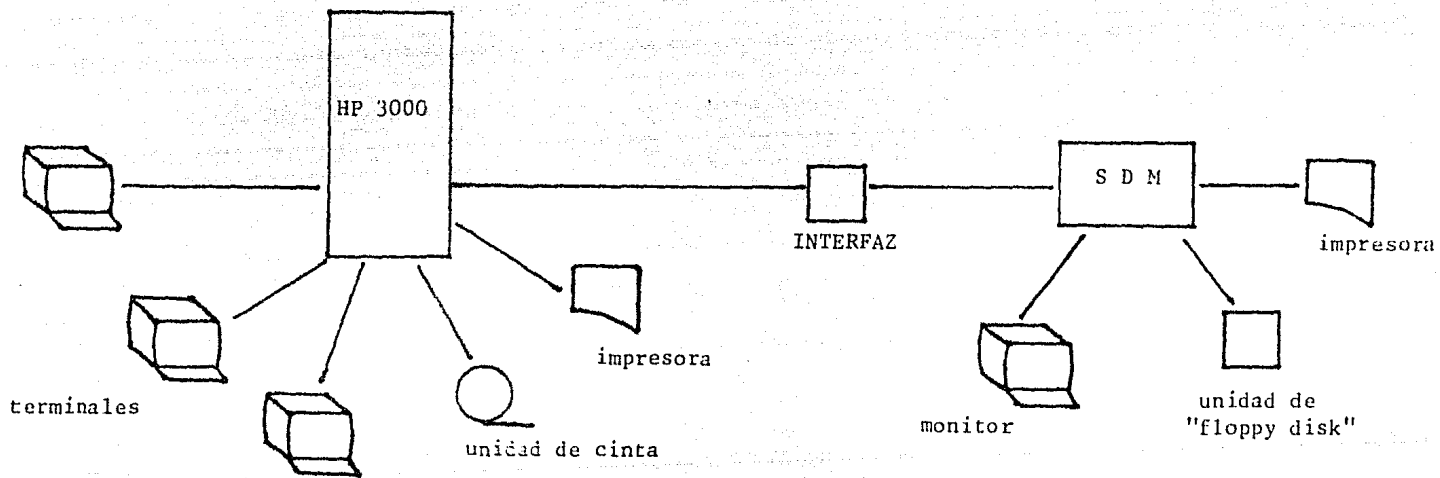
Por otra parte, en todo equipo de computación se deben tener en cuenta dos aspectos primordiales, el equipo físico y circuitería o "hardware", y la programación asociada o "software".

A partir de estas consideraciones sólo resta determinar la naturaleza de la interfaz. Para ello, se describirá nuevamente la secuencia que seguiría un usuario del SDM:

En primer lugar, editaría el archivo fuente a través de una de las terminales de la computadora de propósito general. Cuando dispusiera de tiempo en el SDM, transferiría su archivo fuente desde la computadora donde lo generó al SDM, continuando con el proceso de desarrollo.

Mencionar una "transferencia" de archivos de la computadora de propósito general a la microcomputadora implica una conexión física y un protocolo de comunicación entre ambos sistemas.

Dadas las características de los sistemas existentes en la U.A.M., se diseñará la interfaz considerando que cada computadora pueda ver a la otra como un periférico propio (figura 1.2).



Computadora de propósito general

Sistema de Desarrollo de Microprocesadores

Figura 1.2 La interfaz como periférico propio de cada sistema.

Por lo tanto, el lado de la interfaz conectado a la computadora de propósito general, deberá ser compatible con ella, y lo mismo debe cumplirse para el lado conectado al sistema de desarrollo de microprocesadores.

La interfaz se construirá en el SDM, pues como se verá en la sección 1.2, la filosofía de diseño del SDM permite la integración de un dispositivo de tal naturaleza. Con esto, la programación asociada a la interfaz, deberá efectuarse también en el SDM.

### 1.2.1 Descripción general.

El sistema de desarrollo de microprocesadores Intellec MDS es una herramienta completa que permite el desarrollo de "hardware" y "software".

La operación del sistema se lleva a cabo bajo el control de un microprocesador 8080 de Intel, el cual supervisa los recursos del sistema, tales como: memoria, dispositivos periféricos de entrada/salida, y facilidades opcionales como emulación (ICE) y acceso directo a memoria.

En otras palabras, el Intellec MDS es un sistema modular que posee todos los elementos necesarios para su operación.

El sistema estándar consiste de cuatro módulos, un gabinete para interconectar circuitos impresos, fuentes de poder, ventiladores, un chasis y un panel frontal.

Los cuatro módulos son: módulo del CPU 8080, módulo con 16 K de memoria RAM, módulo de control del panel frontal y módulo del monitor (figura 1.3). Catorce conectores adicionales en el gabinete permiten expandir la capacidad de este sistema modular.

- Módulo del CPU 8080.

Este módulo contiene un microprocesador 8080, tipo n-MOS de 8 bits, un reloj de tiempo real, lógica para inicializar el CPU y lógica de control.

- Módulo del monitor.

Contiene el "hardware" para la interfaz del monitor y de los periféricos del MDS, así como todos los contadores, controles y circuitos de transferencia de datos necesarios para la interfaz de los siguientes periféricos: terminal de impresión, CRT, lectora rápida de cinta de papel, perforadora rápida de cinta de papel, impresora de líneas y programador universal de EPROM (figura 1.4).

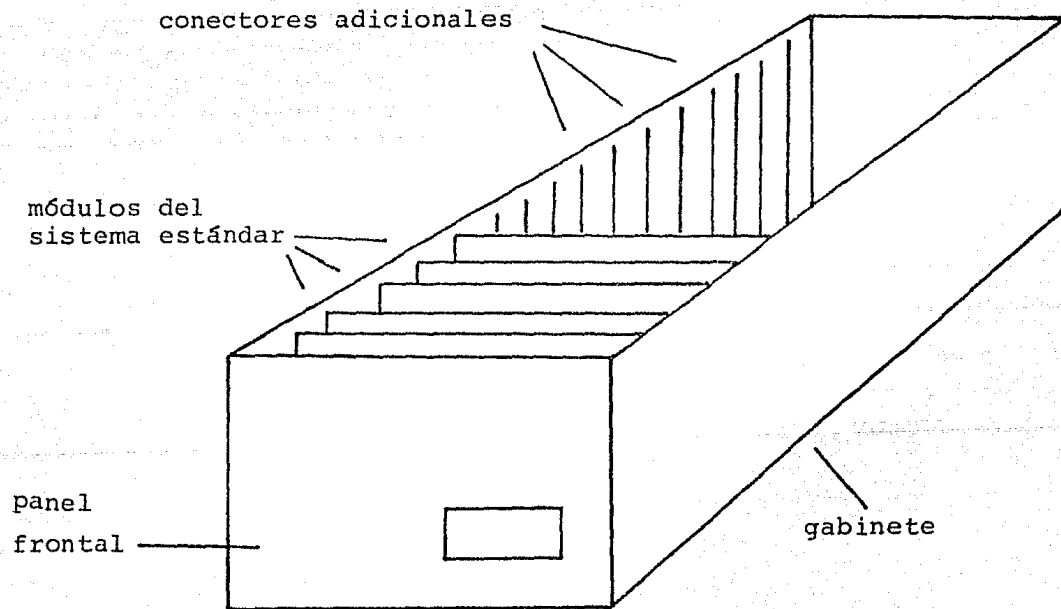


Figura 1.3 Sistema estándar Intellec MDS.

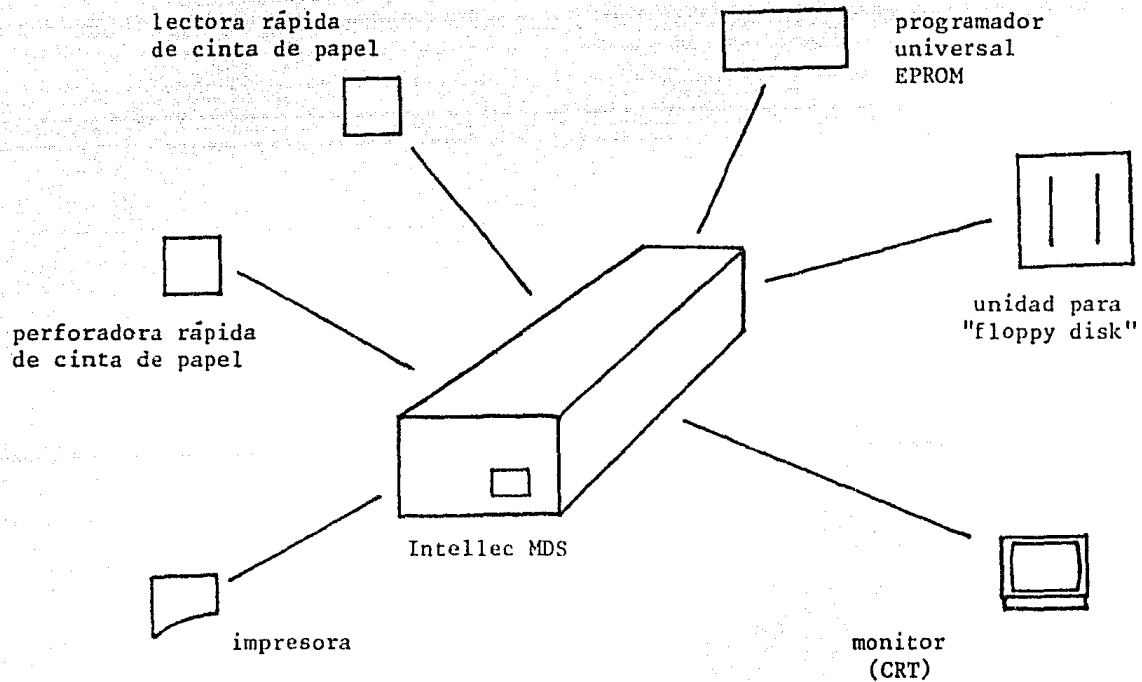


Figura 1.4 Periféricos del Intellec MDS.

- Módulo de control del panel frontal.

Este módulo contiene circuitería para el control de operaciones del panel frontal, a la vez que suministra respaldo para las líneas de control, generación de tiempos y el programa de inicialización.

- Módulo de memoria.

Contiene 16 K de memoria RAM, factible de expanderse mediante módulos adicionales.

-Módulos opcionales.

El sistema puede admitir, recordando que existen 14 conectores adicionales en el gabinete, cualquiera de los siguientes módulos:

El emulador ICE, unidades para disco blando, facilidades entrada/salida para el usuario, memorias PROM o RAM para expanderlas en incrementos de 6 K (PROM) o 16 K (RAM), hasta un máximo de 12 K de ROM y 64 K de RAM, y finalmente, acceso directo a memoria ("DMA").



## 1.3.1 Generalidades.

El sistema de computación HP 3000 es una computadora de propósito general que acepta varios lenguajes y es multiprogramable. Puede manejar simultáneamente procesos interactivos, tiempo compartido y operaciones encadenadas ("batch") en cualesquiera de sus lenguajes de programación.

El "hardware" y el "software" están estrechamente relacionados entre sí, de modo que la circuitería realiza muchas de las operaciones que convencionalmente se llevan a cabo por programación. Cada usuario es independiente respecto a los otros para el sistema, siendo posible interactuar con otros usuarios debido a que el sistema operativo suministra las facilidades necesarias para compartir información.

A nivel general, el sistema HP 3000 consiste de cuatro elementos: el "hardware", subsistemas de "software", los programas del usuario y el sistema operativo MPE (Multi-programming Executive III).

a) El sistema de "hardware" comprende al CPU, la memoria principal y los periféricos disponibles.

b) Los subsistemas de "software" incluyen un juego de programas utilitarios, que son el EDIT-3000 (editor de textos, poderoso y fácil de usar), el FCOPY-3000 (programa para copiado y transferencia de archivos), el SORT-3000 (para

ordenar grabaciones en un archivo), biblioteca compiladora (juego de subrutinas que suministran operaciones comunes necesarias para los compiladores de lenguajes), y biblioteca científica (juegos de subrutinas que suministran funciones matemáticas frecuentemente usadas). Además se ofrecen seis lenguajes de programación de alto nivel, COBOL, RPG, BASIC, FORTRAN, SPL y APL, así como los subsistemas DEL-3000 (Data Entry Library), KSAM-3000 (Keyed Sequential Acces Method) y DBMS (Data Base Management System).

c) Los programas del usuario permiten a éste interactuar en el sistema, mediante compiladores y utilidades a través del MPE, usualmente por la vía del sistema de archivos, con el objeto de emplear el "hardware" y los periféricos del sistema. Generalmente, las tareas que realiza un usuario pertenecen a una de tres categorías: desarrollo de programas, ejecución o aplicación de programas y manejo del sistema.

d) El sistema operativo MPE es el sistema de disco de propósito general que supervisa el proceso de los programas que se corren en la computadora HP 3000, suministrando también un extenso y poderoso conjunto de funciones del sistema.

El MPE III (Multiprogramming Executive), como ya se ha mencionado, es el sistema operativo basado en disco y de propósito general que supervisa el proceso de los programas del usuario de los sistemas de cómputo HP 3000 (figura 1.5). El MPE ofrece dos alternativas para construir una aplicación de operación orientada, bien sea control central, en el cual un programa de aplicación maneja terminales múltiples, o control individual, en el que se da a cada terminal la posibilidad de correr programas separados.

El MPE libera al usuario de muchos programas de control, entrada/salida y otros tipos de responsabilidades al monitorear y controlar la programación de entrada, compilación, preparación para correr, carga, ejecución y salida. También regula el orden en que los programas son ejecutados y asigna los recursos de "hardware" y "software" que requieren.

Los recursos del sistema operativo tales como memoria principal, procesador central y dispositivos y canales periféricos, se encuentran dinámicamente asignados a cada programa según sea necesario.

U S U A R I O S

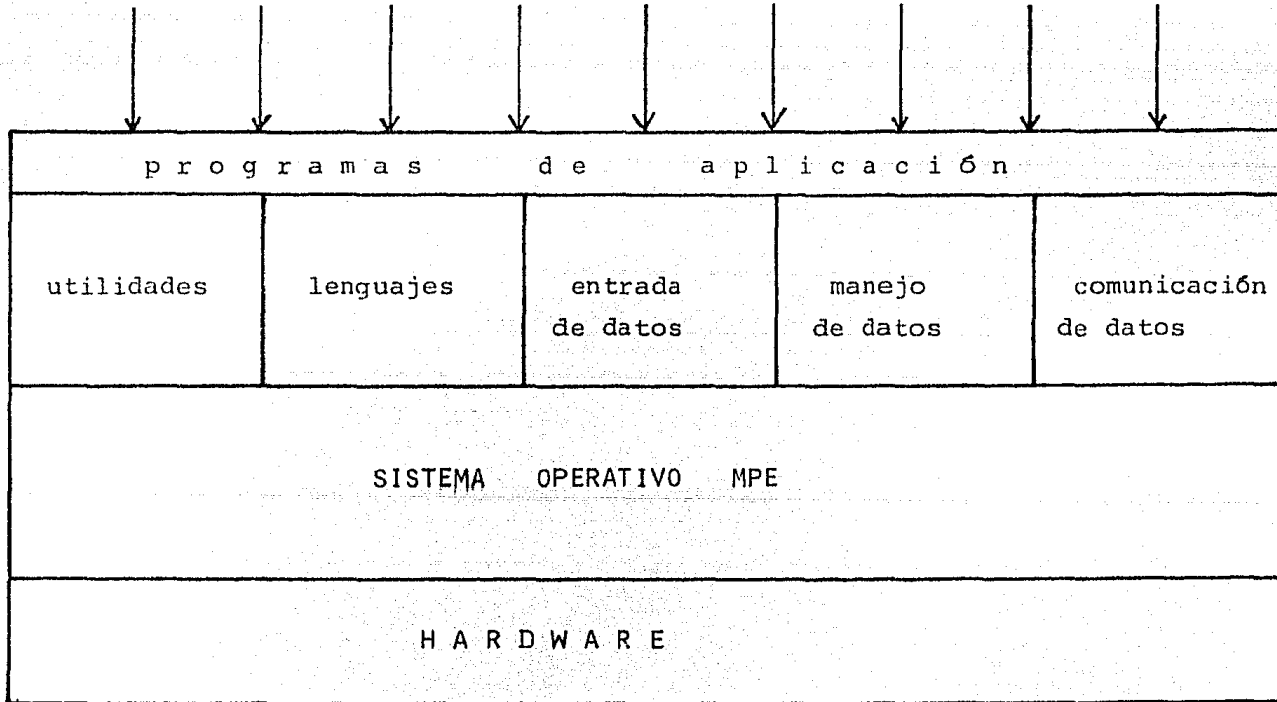


Figura 1.5 El sistema operativo y los componentes principales del sistema HP 3000.

Todos los dispositivos físicos de entrada/salida (periféricos) son manejados por el sistema MPE I/O, el cual recibe solicitudes de ent/sal del "software" de otro sistema, las forma ordenadamente si es necesario, y realiza la transferencia de datos desde o hacia el dispositivo. Ya que todos los dispositivos de ent/sal son tratados como archivos por el MPE, es posible escribir programas sin precisar de inmediato la fuente física de entrada o el destino de salida, y correrlos en modo interactivo o en procesamiento secuencial("batch") sin tener que cambiar los nombres de los archivos a los que se hace referencia.

Las comunicaciones asíncronas de terminales se manejan automáticamente por el MPE. Comunicaciones síncronas de datos a terminales, a otros sistemas de computación HP (incluyendo HP 3000) y a computadoras que no sean HP, se suministran a través de subsistemas opcionales de "software", los cuales incrementan las posibilidades de comunicación de datos del MPE.

El sistema operativo provee pues muchas herramientas para todo el manejo y control del sistema, incluyendo una rutina automática de arranque en caso de fallar el suministro de energía eléctrica.

A continuación se enuncian los principales aspectos del sistema operativo MPE:

- Multiprogramación: proceso de solicitud concurrente, tiempo compartido, proceso secuencial.
- Memoria virtual.
- Arquitectura en columna: separación de instrucciones y datos, segmentación de longitud variable, apiladores de datos.
- Capacidad concurrente de multilenguajes: COBOL, RPG, BASIC, FORTRAN, APL, y SPL.
- Un sistema de archivos uniforme, independiente de dispositivo y lenguaje, con archivo de respaldo y seguridad.
- Sistema de seguridad y contabilidad completa de recursos.
- Poderoso lenguaje de comando que incluye comandos de usuario definidos, control de trabajo condicional, la facilidad HELP y mensajes de error significativos.
- Independencia de archivos y dispositivos.
- Conveniencias de ent/sal: embobinado de entrada y salida, volúmenes de disco privado, etiquetas de cinta magnética.
- Manejo de terminal completo y automático, local y remoto.
- Inclusión automática al sistema (bajo el control de la dirección de las instalaciones).
- Adecuaciones particulares del sistema.
- Arranque automático para fallas de energía eléctrica.

La computadora HP 3000 tiene gran capacidad para establecer de manera eficaz y funcional una comunicación con terminales. La comunicación puede ser síncrona o asíncrona en eslabonamientos de punto a punto y multipunto. Los eslabonamientos se pueden alambrear directamente, o bien emplear MODEMS, siendo posibles varias velocidades en ambos casos.

Aunque la computadora puede enlazarse a terminales con una velocidad de transmisión hasta de 9600 bps en general, en lo que se refiere a las comunicaciones punto a punto, el controlador terminal asíncrono (ATC) permite enlazar hasta 16 terminales asíncronas a la HP 3000, a velocidades hasta de 2400 bps. El ATC es capaz de determinar automáticamente la velocidad para conexiones con modems.

Las terminales cuya interfaz se realiza a través del ATC pueden ser configuradas ante el sistema operativo MPE como terminales de entrada de datos bajo el programa de control, o como terminales "log-on" para tener acceso a todas las funciones de la HP 3000.

Sólo resta destacar que una conexión punto a punto es aquélla en la cual se conectan entre sí dos y solamente dos instalaciones terminales. Por otra parte, cuando se conectan dos o más terminales a una sola línea de comunicaciones, el eslabonamiento resultante recibe el nombre de multipunto.

## 1.4.1 Conexión física.

Como ya se ha visto, la computadora HP 3000 es muy versátil en el manejo de terminales, pudiendo establecer una comunicación hasta de 2400 bits/seg con terminales asíncronas en eslabonamientos punto a punto. Por consiguiente, desde el punto de vista de la HP 3000 y en lo que a "hardware" se refiere, el trabajo se limita a alambrear el conector adecuado e insertarlo en el puerto de la HP 3000 que designe el operador.

En cuanto a la microcomputadora Intellec MDS, su diseño modular y las facilidades de ent/sal para el usuario confirman el camino a seguir para conectar ambos sistemas entre sí.

En efecto, la interfaz referida será un módulo opcional tal como se describe en la sección 1.2.2. Este módulo opcional no es otra cosa que una tarjeta de circuito impreso, que una vez insertada en uno de los conectores adicionales del gabinete, funcionará como un puerto análogo al de la computadora HP 3000 (figura 1.6).

Parte del trabajo consistirá en diseñar y construir esta tarjeta de interfaz.



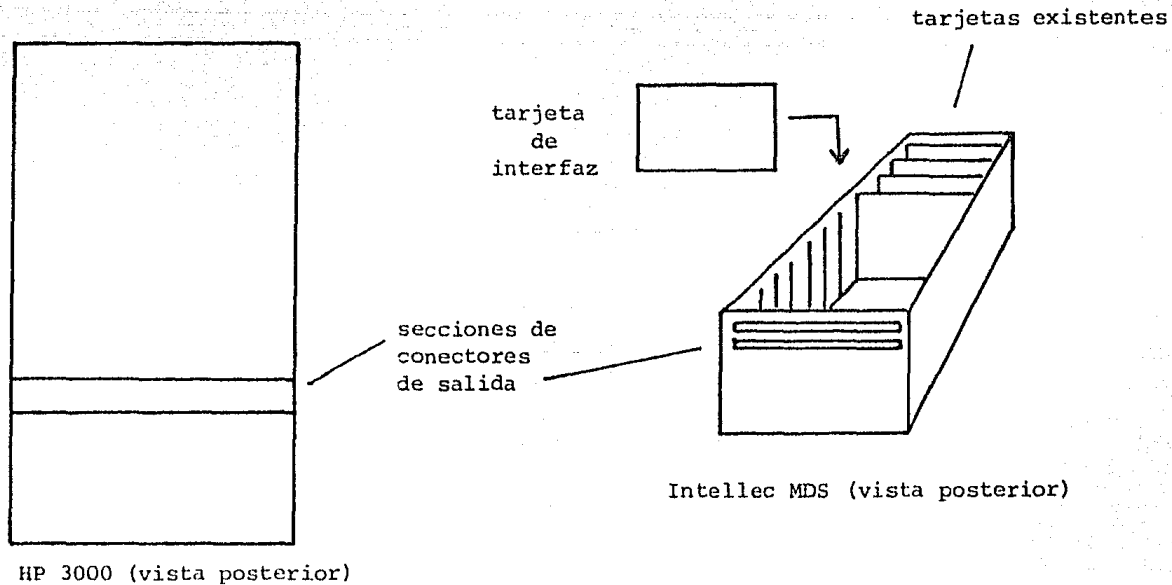


Figura 1.6 Puertos de salida en la computadora HP 3000 y en la microcomputadora Intellec MDS.

En la microcomputadora, el puerto de salida es de hecho la tarjeta de interfaz, que irá alamburada a uno de los conectores de salida existentes en la parte posterior del gabinete del Intellec MDS.

Una vez enlazadas físicamente ambas computadoras, sólo resta considerar el problema de la comunicación en sí, por lo que es conveniente recordar que en sistemas digitales existen dos tipos de comunicación, síncrona y asíncrona.

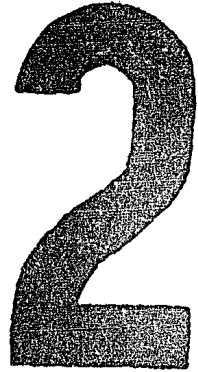
Tal como fue planteado en la sección 1.1.2, cada computadora deberá ver a la otra como un periférico propio. De igual modo, debido a que la computadora HP 3000 no va a sufrir ninguna modificación física, el tipo de comunicación que se llevará a cabo, será desde el punto de vista de la HP 3000. Como ésta se comunica con sus terminales en forma asíncrona, éste será el tipo de transmisión entre ambos sistemas.

De este modo, el diseño de la interfaz debe incluir los requerimientos necesarios para ser compatible con la HP 3000. Es decir, la naturaleza de la interfaz debe ser tal que la HP 3000 "vea" a una de sus terminales.

Por ser la interfaz un módulo más del Intellec MDS, será necesaria la elaboración de un programa de control de la interfaz, o sea, un programa que defina las características de la transmisión, la selección del puerto (desde el punto de vista de la microcomputadora) y una clara distinción entre transmisión y recepción de datos.

Análogamente, el operador de la HP 3000 deberá configurar el puerto de salida de la computadora a nivel de "software" y de acuerdo a las especificaciones indicadas en el manual de HP requeridas para este proyecto, las cuales, lógicamente, deberán ser las mismas especificaciones de diseño de la tarjeta de interfaz.

\* \* \* \* \*



C O M U N I C A C I O N

A

D I S T A N C I A

## 2.1.1 Transmisiones en serie y en paralelo.

Los bits de datos binarios son comúnmente transmitidos entre dispositivos electrónicos mediante cambios de corriente o de voltaje. Los datos pueden ser transferidos en "serie" mediante una línea simple (más el retorno), o en "paralelo" usando varias líneas a la vez. En ambos casos, las transferencias pueden ser "síncronas", de tal modo que se puede predecir con exactitud la partida o la llegada de cada bit de información, o pueden ser "asíncronas", caso en el cual los datos no se transmiten con periodicidad uniforme (sección 2.1.2).

En la transmisión en paralelo, cada bit o juego de bits que representan un caracter tienen su propio alambre. Una línea adicional llamada "strobe" o "clock" es la que notifica a la unidad receptora el momento en que todos los bits están presentes en sus respectivos cables para que los voltajes puedan ser muestreados (figura 2.1).

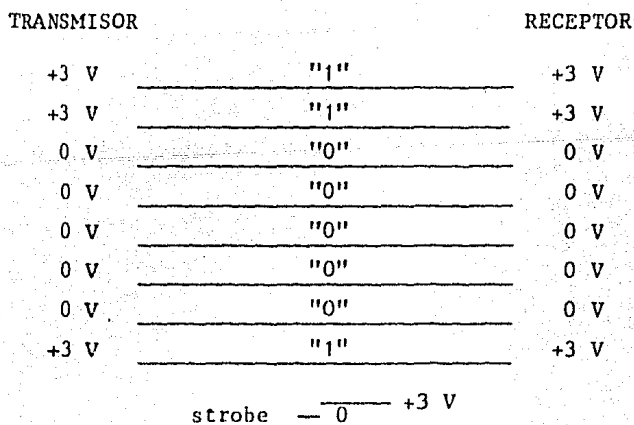


Figura 2.1 Transferencia de datos en paralelo.

La figura muestra la transmisión del caracter de 8 bits 11000001. En el momento en que el "strobe" pasa de su valor inicial de "0" a "1", el receptor efectúa el muestreo de los voltajes presentes en las líneas.

En la transmisión en serie, los bits de información son enviados uno tras otro a través de una simple línea.

Las computadoras y otros sistemas digitales de alta velocidad generalmente operan con datos en paralelo, sin importar dónde se encuentran los dispositivos, siempre y cuando estén físicamente cercanos. Esto se debe a que a medida que la distancia entre los dispositivos se incrementa, el alambre múltiple no solo deviene más costoso, sino que además los impulsores y los receptores de línea ("drivers") se vuelven más complejos debido a la dificultad creciente de impulsar y recibir adecuadamente señales en alambres de gran longitud.

El empleo de la transmisión en serie se llevará a cabo cuando el costo de las líneas entre los equipos físicos sea suficientemente elevado, a lo que habría que añadir la disponibilidad de líneas telefónicas. Las conversiones paralelo/serie y serie/paralelo se efectúan con registros de corrimiento.

Por todo esto, en la mayoría de las aplicaciones de comunicación de datos, se prefiere la transmisión en serie a la transmisión en paralelo.

La transferencia en serie de datos hacia o desde una terminal o interfaz de computadora, se describe en términos de bits/segundo.

La razón de cambio de facilidades de transmisión se describe en bauds, de este modo:

$$1 \frac{\text{bit}}{\text{segundo}} = 1 \text{ baud}$$

si y solo si cada elemento de señalización en una transmisión conduce un bit de información.

A causa del diseño mecánico en los primeros teletipos, y para facilitar las operaciones sin fallo, los sistemas de teletipos que transmiten en serie, han adoptado una convención que dice lo siguiente:

Una línea desocupada (por la cual no se están enviando datos) debe ser de tal naturaleza que esté fluyendo corriente por ella. Esto significa que la transmisión de datos deberá ocurrir cuando la corriente en la línea sea interrumpida en una forma especificada.

Por convención, el estado en el cual la línea está desocupada (con corriente fluyendo) es el estado "1" o condición de "MARK", y la ausencia de corriente en la línea es el estado "0" o condición de "SPACE".

Para activar el mecanismo del teletipo receptor, la línea que se encuentra desocupada (en estado "1"), se lleva a "0" durante el tiempo que dura un bit. Este bit es llamado "START" bit. Los bits siguientes, que pueden ser de 5 a 8 dependiendo del tipo de código, representan el carácter que se está enviando, es decir, se trata de la información original que se desea transmitir, por lo que la línea puede variar del estado "1" al estado "0" o viceversa, de acuerdo al carácter presente.

Para preparar al teletipo para el siguiente carácter, es necesario llevarlo nuevamente al estado "1", o sea, a la condición de línea desocupada, de tal modo que pueda

esperar al siguiente "start" bit. Esto se efectúa enviando el estado "1" durante el tiempo de uno o más bits (en la práctica: 1, 1.5 o 2 bits), período llamado de "STOP" bit (figura 2.2).

El intervalo del "stop" bit debe aparecer forzosa-mente después del caracter, y una vez terminado este intervalo de "stop", el siguiente caracter puede aparecer en cualquier momento. La ausencia de una concordancia continua de sincronización entre el transmisor y el receptor, o más específicamente, la ausencia de una señal de reloj dentro del canal de datos o acompañándolo, es la causa de que este tipo de transmisión se llame asíncrona, literalmente hablando, sin sincronización.

En vista de que no hay una sincronización durante la transmisión, la estación receptora debe saber a qué velocidad está enviando los bits la estación transmisora para poder muestrearlos con la velocidad correcta.

Por otra parte, aunque los receptores asíncronos modernos ya no requieren del intervalo de "stop" para el funcionamiento correcto del mecanismo, sigue siendo necesario para garantizar que cada caracter comenzará con la transición 1 a 0, aún cuando el caracter precedente haya sido formado completamente por ceros. La necesidad de esta transición para indicar el principio de cada caracter, ocasiona que un caracter de 8 bits requiera en realidad de 10 bits para ser transmitido (1 start bit + 8 bits de datos + 1 stop bit = 10 bits).

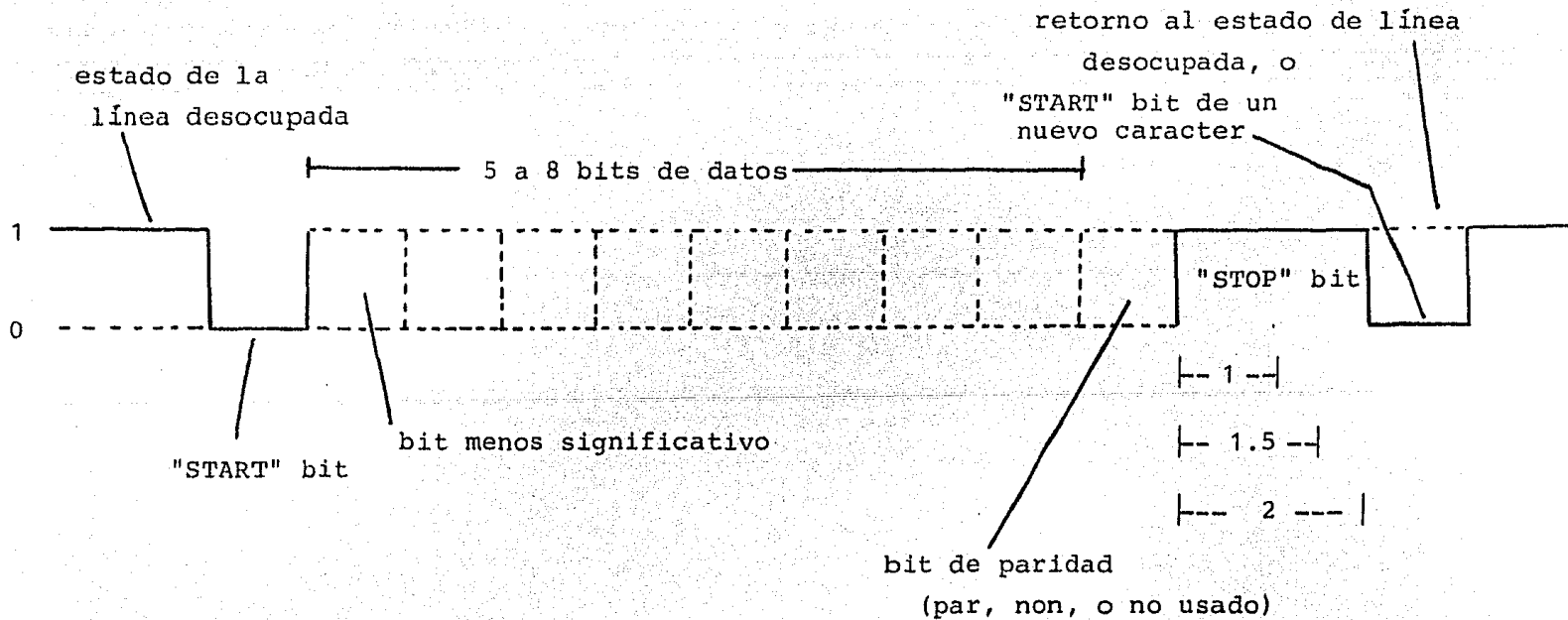


Figura 2.2 Formato de un Caracter de Datos Asíncrono.



Analizando este hecho, se desprende que el 20% del tiempo de la línea se emplea estrictamente en propósitos de identificación de caracteres cuando la comunicación es asíncrona.

La comunicación síncrona requiere un tiempo de reloj que puede mandarse por un cable separado del de los datos desde el punto de transmisión al punto de recepción, o que bien puede modularse durante el proceso de codificación de los datos, para después ser demodulado y separado en otra línea en la estación receptora. Este tiempo de reloj es el que indica a la circuitería de comunicación de datos (típicamente una interfaz de computadora) el instante apropiado para muestrear el dato recibido.

Ya que los bits "start" y "stop" no son requeridos en comunicación síncrona, todos los bits son usados para transmitir datos, con lo que se elimina la pérdida del 20% típica de la comunicación asíncrona. De cualquier modo, como la información suministrada por el "start" bit y el "stop" bit no existe más, debe haber otro método para determinar qué grupo de bits constituyen un carácter.

En la figura 2.3 se muestra el formato de un carácter de datos síncrono junto al tiempo de reloj asociado. Los bits del carácter de datos que aparecen en los tiempos del 1 al 8 podrían formar un carácter, de tal modo

que los bits que aparecen del 9 al 13 formen el siguiente caracter. Pero también sería posible que el tiempo 1 fuera el último bit de un caracter, y por consiguiente, los bits 2 al 9 serían el siguiente, los bits del 10 al 13 un tercer caracter, y así sucesivamente. Todo esto significa que se debe contar con una delimitación precisa de cada caracter, lo que se logra definiendo un caracter de sincronización, comúnmente llamado "SYNC".

El caracter "sync" se escoge de tal manera que su arreglo de bits sea significativamente diferente de los caracteres de 8 bits que se transmiten con regularidad, y que además presente un modelo irregular (que no sea por ejemplo 01010101). Por lo tanto, ninguno de los caracteres que preceden o siguen al caracter "sync" se parecerá a éste, resultando fácilmente reconocible.

El caracter síncrono usado en el ASCII de 1968 es 10010110 y corresponde al 226 octal.

En cuanto a la forma de operación de los receptores síncronos, éstos se colocan en modo de búsqueda mediante "hardware" o "software", dependiendo del sistema en particular, ya sea cuando comienza una transmisión o cuando ha ocurrido una pérdida de datos, de tal modo que el "hardware" o el "software" determinan que es necesaria una nueva sincronización.

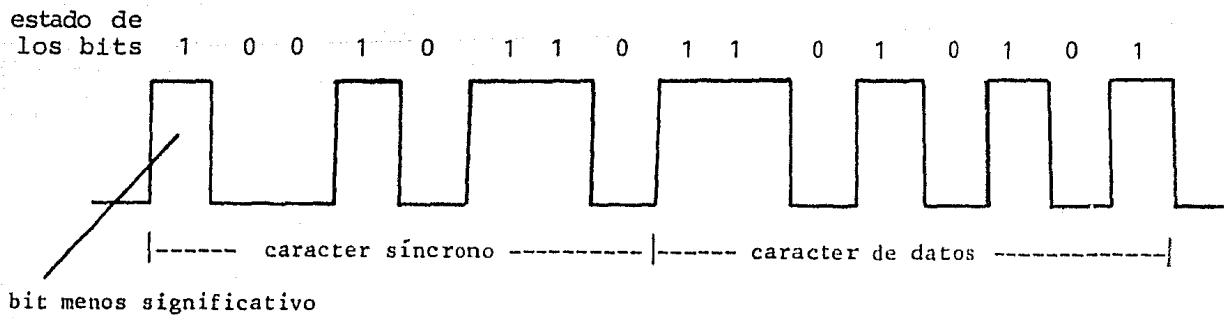
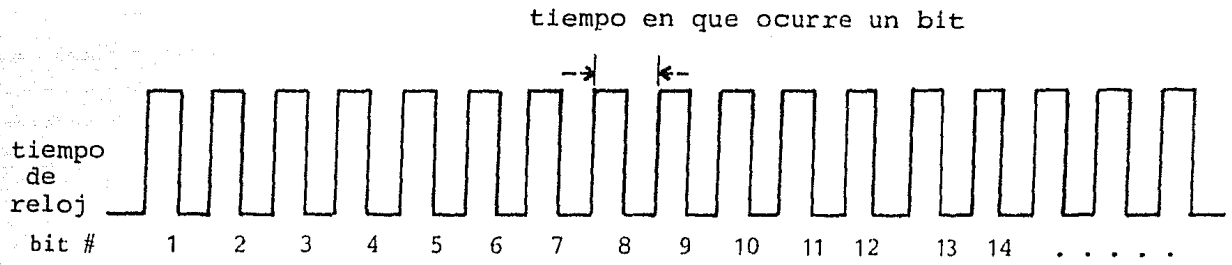


Figura 2.3 Formato de un Caracter de Datos Síncrono.

Durante el procedimiento de búsqueda, el "hardware" coloca un bit en el registro de corrimiento receptor, y lo compara con el primer bit del carácter "sync" previamente almacenado en otro registro, y así sucesivamente hasta que sea identificado el carácter "sync", con lo cual el receptor comienza a admitir bits y levanta una bandera indicando "carácter disponible" cada ocho bits. Por el contrario, si no identifica el carácter "sync", toma uno a uno los bits de la información que va recibiendo colocándolos en el registro de corrimiento y efectuando una y otra vez la comparación con el carácter "sync" almacenado, hasta que aparezca éste.

Para asegurar mayor certeza en el proceso de sincronización, la mayoría de los sistemas de comunicación requieren que el receptor identifique dos caracteres síncronos sucesivos antes de levantar la bandera de "carácter disponible" cada ocho bits.

Por último, es sumamente importante hacer notar que no siempre es cierta la aseveración en el sentido de que asíncrono significa lento, y síncrono significa rápido. Hay terminales que operan a distancias cortas a una velocidad asíncrona de 38,400 bits/seg. Existe también un número importante de terminales en uso, operando en forma síncrona a 1,200 bits/seg.

## 2.2.1 Aspectos generales.

Esta interfaz fue creada por la Asociación de Industria Eléctrica (EIA) de los Estados Unidos, en respuesta a la problemática planteada por la ausencia de un estándar en la comunicación de datos.

En un principio, las compañías asociadas bajo el nombre de "American Telephone & Telegraph Company", eran prácticamente las únicas que suministraban servicio en el área de comunicaciones. Es por esto que el único estándar de la industria lo constituían los modems desarrollados por los ingenieros de los Laboratorios Bell. Sin embargo, los productores independientes de modems, de equipos terminales y de comunicación, y de interfaces para estos equipos, se vieron en la necesidad de conocer las características eléctricas de los modems Bell de interfaz. Fue así como la Asociación de Industria Eléctrica en cooperación con el sistema Bell, y los productores independientes de modems y computadoras, desarrollaron un estándar para la interfaz entre Equipo Terminal de Datos y Equipo de Comunicación de Datos de intercambio binario en serie.

El estándar se llama RS-232-C, donde la "C" hace alusión a la última revisión.

El contenido del estándar EIA es el siguiente:

- Características de las señales eléctricas.
- Características mecánicas de la interfaz.
- Descripción funcional de los circuitos de intercambio.
- Una lista de estándares de circuitos específicos de intercambio para grupos específicos de aplicaciones de sistemas de comunicación.

Para los objetivos del presente trabajo, sólo será necesario conocer la descripción funcional de los circuitos con los cuales se realiza la interfaz, descripción que se estudiará en la sección 2.2.2.

Para realizar una interfaz con un modem en una línea privada, tan sólo se requiere de un número limitado de circuitos de interfaz (figura 2.4). Las definiciones de estas funciones se especifican a continuación.

Circuito AA ("Protective Ground")

La conexión de este conductor se efectúa en el chasis del equipo, pudiéndose conectar a tierras externas de acuerdo a las necesidades existentes.

Circuito AB ("Signal Ground" o "Common Return")

Este conductor establece la referencia de potencial común para todos los circuitos de intercambio, con excepción del circuito AA. Dentro del equipo de comunicación de datos será posible conectar este circuito al circuito AA por medio de un alambre. Esto puede ser de gran utilidad para minimizar la intromisión de ruido en los circuitos electrónicos.

Circuito BA ("Transmitted Data" o "TxD")

Las señales en este circuito son generadas por el equipo terminal de datos y transferidas al convertidor local de señales. De aquí son transmitidas a otro equipo terminal de datos remoto.

El equipo terminal de datos deberá mantener al circuito BA en condición "MARK" durante los intervalos entre

caracteres o palabras, y en todo tiempo que no se estén transmitiendo datos.

En todos los sistemas, el equipo terminal de datos no efectuará ninguna transmisión a menos que esté presente la condición de encendido en todos y cada uno de los cuatro circuitos siguientes, siempre y cuando estén implementados:

Circuito CA ("Request to Send")

Circuito CB ("Clear to Send")

Circuito CC ("Data Set Ready")

Circuito CD ("Data Terminal Ready")

Entonces pues, cuando las cuatro señales anteriores estén activas, toda señal transmitida a través de la interfaz mediante el circuito de intercambio BA ("TxD"), será transmitida al canal de comunicación, es decir, al exterior.

Circuito BB ("Received Data" o "RxD")

Las señales de este circuito se generan como respuesta a la información recibida desde el equipo terminal de datos remoto.

El circuito BB deberá mantener la condición "MARK" durante todo el tiempo que el circuito CF ("Received Line Signal Detector") esté apagado, o sea, mientras no esté presente una portadora.



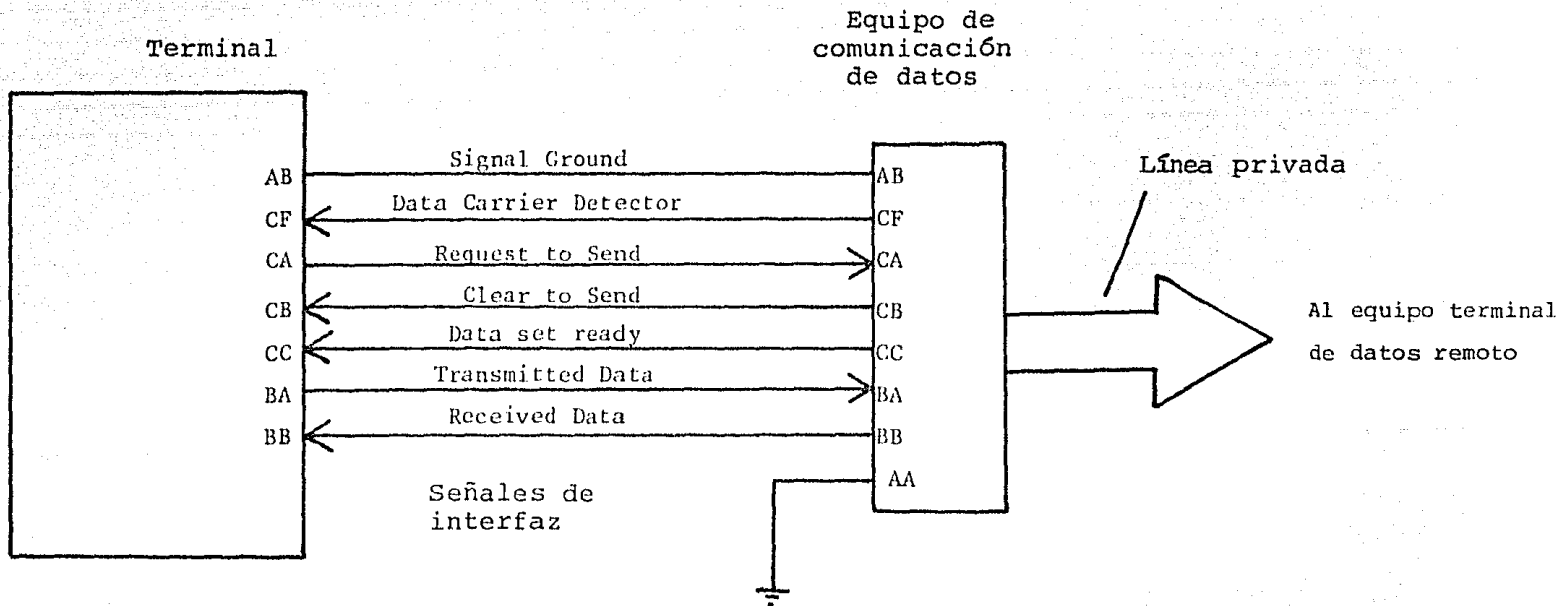


Figura 2.4 Interfaz entre una terminal y un equipo de comunicación de datos (modem).

### Circuito CA ("Request to Send" o "RTS")

Este circuito se emplea para condicionar al equipo local de comunicación de datos a la transmisión, y en el caso de una canal "half-duplex" (sección 2.3.1), controlar la dirección de la transmisión de datos del equipo local.

### Circuito CB ("Clear to Send" o "CTS")

Las señales de este circuito son generadas por el Equipo de Comunicación de Datos para indicar si está o no está preparado para comenzar una transmisión de datos.

La condición de encendido en este circuito, en combinación con el encendido de los circuitos de intercambio CA, CC Y CD (este último, "Data Terminal Ready" o "DTR", sólo se toma en cuenta si está implementado), indica que las señales presentes en el circuito BA ("TxD") serán transmitidas al canal de comunicación.

La condición de apagado significa que el equipo terminal no transferirá datos a través del circuito de intercambio BA de la interfaz.

La condición de encendido del circuito CB es una respuesta a la concurrencia simultánea del encendido de los circuitos CC ("Data Set Ready") y CA ("Request to Send").

### Circuito CC ("Data Set Ready" o "DSR")

Las señales de este circuito se usan para indicar el estado del equipo local.

Cuando la condición de encendido se encuentra presente en este circuito, se debe a lo siguiente:

- El equipo local de comunicación de datos se halla conectado a un canal de comunicación.
- El equipo no se encuentra en modo de prueba.
- El equipo ha completado, en ciertos casos, funciones de tiempo o transmisiones de tonos de respuesta.

El circuito CC sólo indica el estado del equipo local, y por tanto la condición de encendido no será interpretada como el establecimiento de un canal de comunicación, ni como indicador del estado de un equipo remoto.

La condición de apagado estará presente en los demás casos, lo que significa que el equipo terminal ignorará cualquier señal que aparezca en los demás circuitos de intercambio.

Circuito CF ("Received Line Signal Detector" o "Data Carrier Detector")

La condición de encendido se presenta cuando el equipo de comunicación de datos está recibiendo una señal que cumple con sus características de diseño. Estas características son establecidas por el fabricante de equipos de comunicación.

La condición de apagado indica que no se está recibiendo señal alguna, o que ésta no se puede adecuar para demodularse.

La condición de apagado en el circuito CF provocará además que el circuito BB ("RxD") quede fijo a "1" (condición de "mark").

#### Circuito CD ("Data Terminal Ready")

Las señales de este circuito se usan para controlar la integración del equipo de comunicación de datos al canal de comunicaciones.

La condición de encendido prepara al equipo de comunicación de datos para conectarse al canal de comunicaciones, manteniendo la conexión establecida.

La condición de apagado remueve al equipo del canal de comunicación.

Esta señal no es necesaria para modems asíncronos en líneas privadas, requiriéndose en el caso de redes telefónicas, para lo cual deberá contar además con la señal CE ("Ring Indicator").

Una gran parte de los equipos compatibles con las normas de la interfaz EIA se pueden conectar físicamente de la siguiente manera:

Número de contacto	Circuito de intercambio en	Nombre
1	AA	Protective Ground
2	BA	Transmitted Data
3	BB	Received Data
4	CA	Request to Send
5	CB	Clear to Send
6	CC	Data Set Ready
7	AB	Signal Ground
8	CF	Data Carrier Detector

Los conectores usados en este tipo de interfaz son de 25 contactos, de los cuales sólo se requiere alambrear los ocho primeros, tal como puede apreciarse en la tabla anterior.

Los contactos restantes pueden estar libres o suministrar otro tipo de circuitos, dependiendo del equipo.

El conector de salida del equipo de comunicación de datos es de 9 contactos, de los cuales generalmente sólo se alambra los dos o cuatro primeros (correspondientes a transmisión y recepción de datos). En vez de este conector, algunos equipos ofrecen una tira terminal con cuatro tornillos.

En cualquier caso, la información necesaria para efectuar las conexiones debe ser proporcionada por el fabricante de los equipos.

## 2.3.1 La unidad moduladora demoduladora (MODEM).

MODEM es el acrónimo de una unidad MODuladora DEMo-  
duladora, cuya función consiste en covertir datos de una  
forma que es compatible con el equipo de procesamiento de  
datos a una forma que es compatible con la facilidad de  
transmisión y viceversa.

Esto significa que el modem actuará recibiendo la in-  
formación de los sistemas de acuerdo a las características  
propias de cada uno, convirtiendo posteriormente esa in-  
formación, sin falsearla, para que sea transmisible de a-  
cuerdo a los medios existentes (figura 2.5).

En términos generales, la circuitería interna de un  
modem, aparte de su fuente de poder, se encarga de esta-  
blecer las velocidades de transmisión, operaciones de mo-  
dulación y de demodulación, frecuencias de transmisión,  
modos de operación, niveles de transmisión y recepción,  
y otras funciones para diversas aplicaciones.

Las velocidades de transmisión de un modem son espe-  
cificadas por el fabricante, y se definen con el rango en  
el cual el dispositivo responde a la transmisión. Por es-  
to es posible encontrar un modem para cada necesidad, des-  
de modems que cubren de 50 a 300 bauds, hasta aquellos  
que alcanzan los 9600 bauds. La forma de transmisión pue-  
de ser síncrona o asíncrona, dependiendo del dispositivo  
elegido.

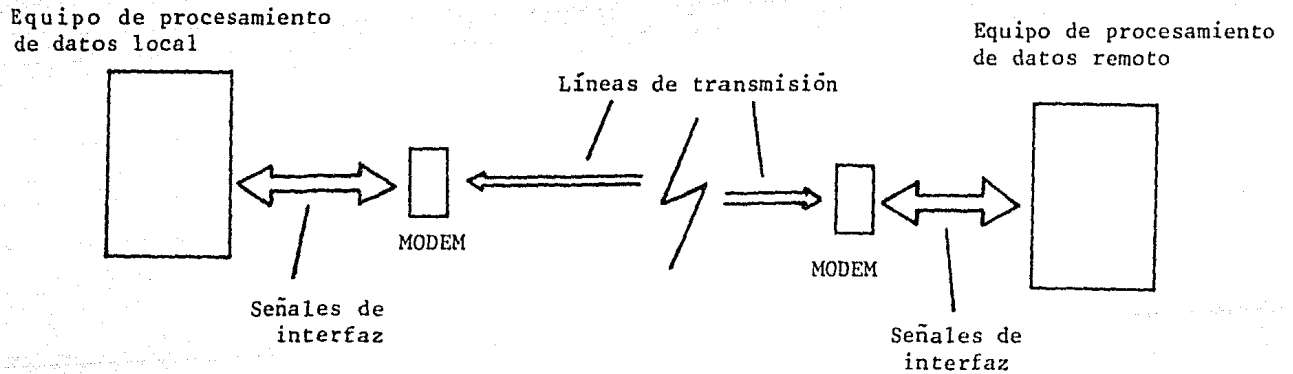


Figura 2.5 El modem recibe la información de cada equipo y la convierte para hacerla compatible con las facilidades de transmisión.

En general, los modems ofrecen dos alternativas de transmisión, "half-duplex" y "full-duplex". En el primer caso, el modem efectúa alternativamente las operaciones de transmisión y recepción, esto es, cuando una función está presente, por ejemplo la recepción de datos, automáticamente se inhibe la transmisión y viceversa.

La facilidad "full-duplex" permite efectuar ambas operaciones, transmisión y recepción, sea que ocurran al mismo tiempo o no.

La transmisión en "half-duplex" requiere de dos hilos, mientras que en "full-duplex" se requieren cuatro. No obstante, existen modems que permiten operar en "full-duplex" con dos hilos.



Para efectuar la conexión entre un equipo de procesamiento de datos local y uno remoto mediante modems, se deberá contar con dos cables de interfaz y la línea de transmisión.

La transmisión en "half-duplex" o "full-duplex" estará supeditada al número de hilos que tenga la línea de transmisión, así como al equipo de comunicación de datos (modem) con que se cuente, según se vió en la sección 2.3.1.

Respecto a los cables de interfaz, éstos enlazarán cada equipo de procesamiento de datos con el modem respectivo. Es conveniente hacer notar que algunos equipos comerciales de comunicación de datos trabajan en modo de respuesta ("answer mode") y en modo de origen ("originate mode"). En caso de contar con modems de este tipo, se debe habilitar en modo de origen al modem local, y en modo de respuesta al modem remoto.

La figura 2.6 muestra una conexión mediante modems entre dos equipos de procesamiento de datos, transmitiendo en "full-duplex" sobre cuatro hilos.

La figura 2.7 muestra como alambrar un cable de interfaz para conectar entre sí a dos equipos de procesamiento de datos sin el concurso de modems.

Este tipo de conexión sólo se recomienda cuando los equipos estén físicamente cercanos el uno del otro.

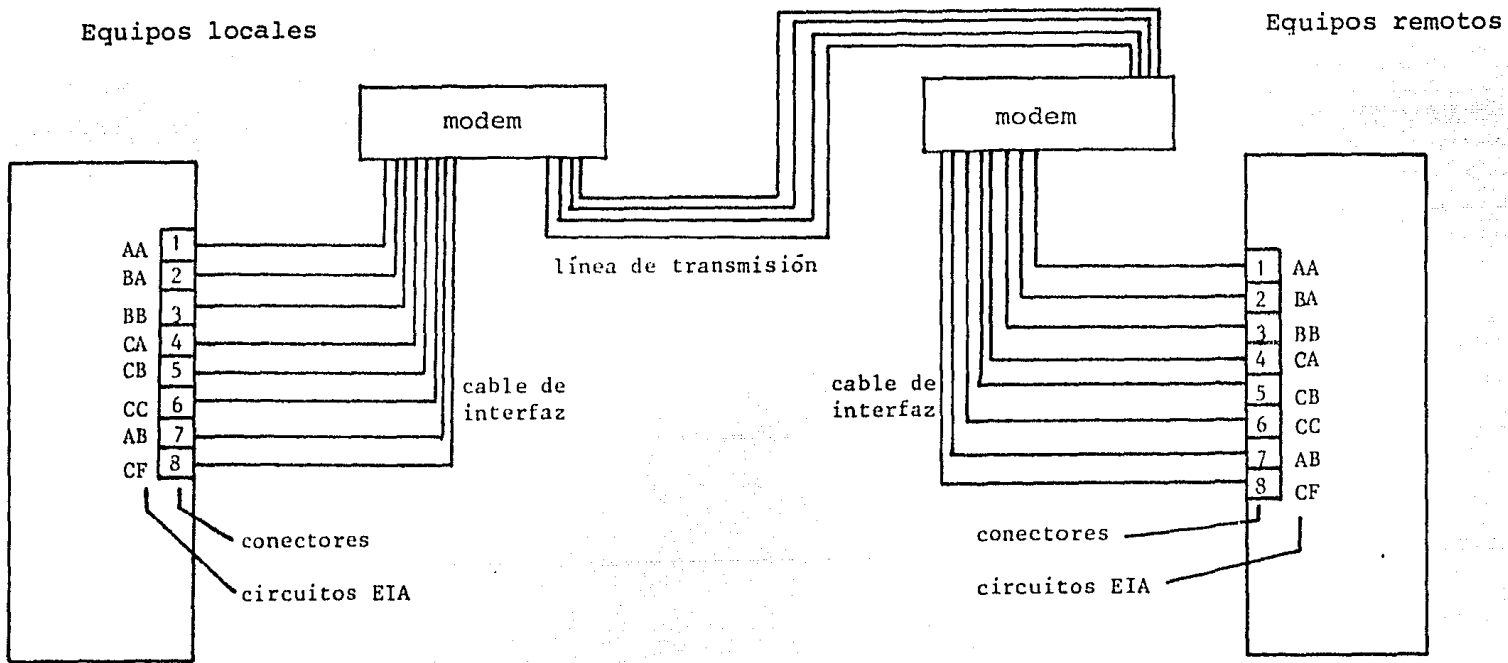


Figura 2.6 Conexión mediante modems entre dos equipos de procesamiento de datos.

La transmisión es "full-duplex" sobre 4 hilos.

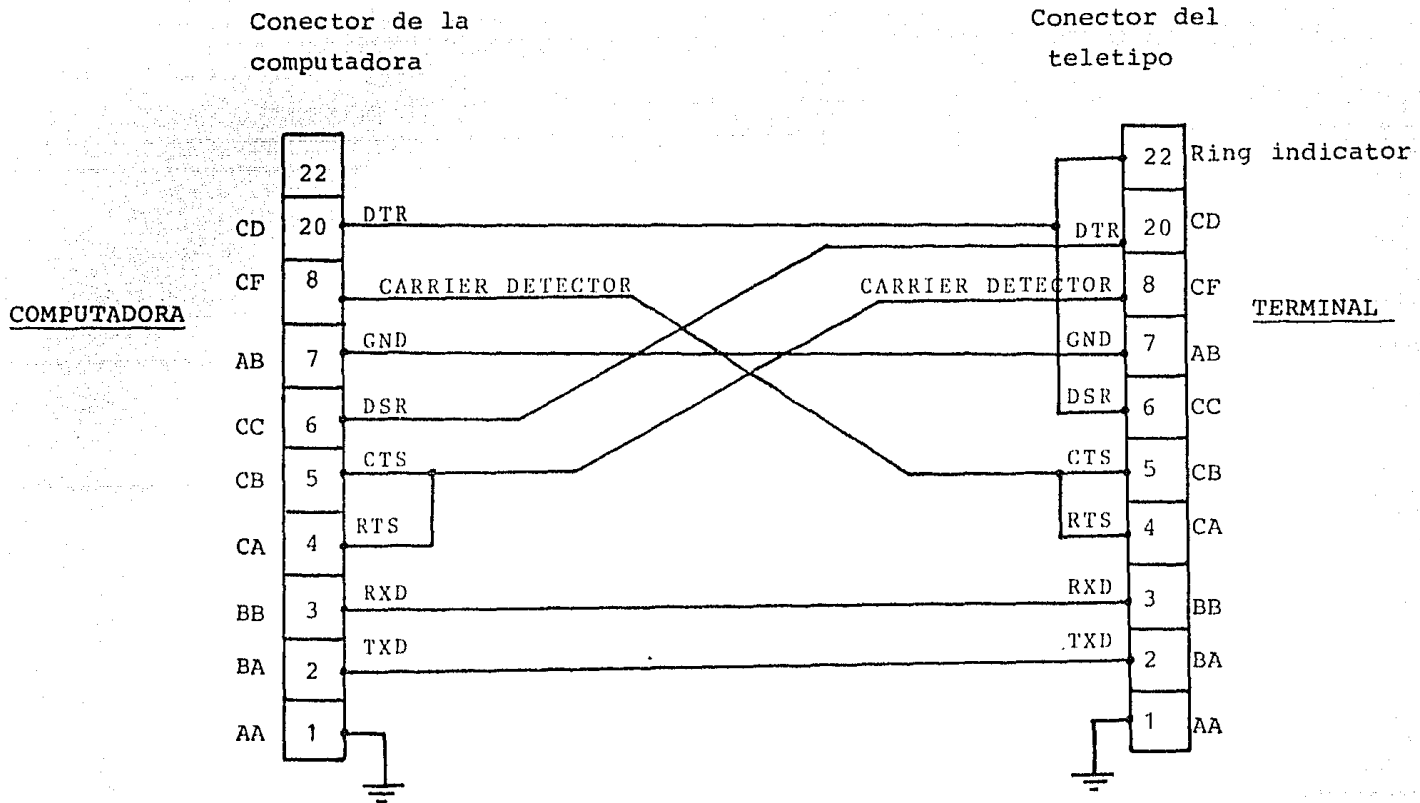


Figura 2.7 Conexión sin modems entre una computadora y una terminal asíncrona.

# 3

\* \* \* \* \*

D I S E Ñ O   D E   L A  
I N T E R F A Z   E N   L A  
M I C R O C O M P U T A D O R A

### 3.1.1 Diagrama de bloques del sistema.

Cualquier sistema de computación consiste básicamente de tres bloques, tal como se muestra en la figura 3.1. Estos bloques son:

MEMORIA

UNIDAD CENTRAL DE PROCESO

PUERTOS DE ENTRADA/SALIDA

#### Bloque de la unidad central de proceso

Contiene a la unidad central de proceso (CPU), la base de tiempo del sistema, circuitería de interfaz para la memoria y dispositivos de entrada/salida. El CPU puede tener acceso rápidamente a cualquier dato almacenado en la memoria.

#### Bloque de la memoria

En el caso de nuestra microcomputadora, contiene "Read Only Memory" (ROM) y "Read/Write Memory" (RAM) para almacenamiento de datos y programas.

#### Bloque de entrada/salida

Contiene circuitería que permite al sistema de computación comunicarse con dispositivos o estructuras existentes fuera del CPU o del arreglo de memoria. Por ejemplo, teclados, discos blandos, cinta de papel perforado y diversos dispositivos.

Los tres bloques anteriormente descritos se encuentran interconectados por tres tipos de canales:

CANAL DE DATOS

CANAL DE DIRECCIONES

CANAL DE CONTROL

#### Canal de datos

Es un camino bidireccional a través del cual hay un flujo de datos entre el CPU y la memoria o los dispositivos de ent/sal.

#### Canal de direcciones

Se trata de un grupo unidireccional de líneas, cuya función es identificar una localidad de memoria en particular, o un dispositivo específico de ent/sal.

#### Canal de control

Nuevamente, es un grupo unidireccional de señales que indican el tipo de actividad presente durante el proceso. El tipo de actividad puede pertenecer a una de cinco categorías:

- escritura en memoria
- lectura de memoria
- escritura de ent/sal
- lectura de ent/sal
- reconocimiento de interrupciones

El comportamiento en el tiempo de estos tres tipos de canales se haya descrito en el apéndice C.

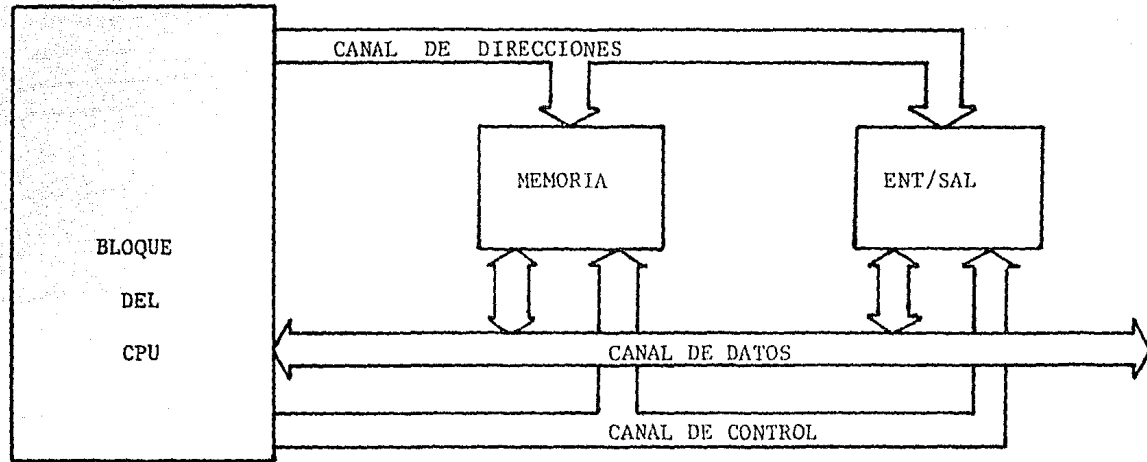


Figura 3.1 Diagrama de bloques de un sistema de computación básico.

La operación básica del sistema es la siguiente:

1 El CPU efectúa una actividad de comando en el canal de control.

2 El CPU emite un código binario en el canal de direcciones para identificar cuál localidad de memoria o dispositivo de ent/sal será involucrado en el proceso.

3 El CPU recibe o transmite datos desde o hacia la localidad seleccionada de memoria o dispositivo de ent/sal.

4 El CPU regresa al punto 1 y emite el siguiente comando.

En la figura 3.2 se muestra en detalle el bloque de la Unidad Central de Proceso, en donde se puede apreciar que está formada por tres elementos, el CPU propiamente, un generador de tiempo con un impulsor de alto nivel, y un impulsor de canal bidireccional con un controlador del sistema.

#### CPU 8080

La unidad central de proceso 8080 se estudiará en detalle en la sección 3.1.2.

#### Generador de tiempo e impulsor de alto nivel

El 8080 es un dispositivo dinámico, lo que significa que sus elementos internos de almacenamiento y la circuitería lógica requieren de una referencia de tiempo, sumi-



nistrada por circuitería externa. El 8080 requiere dos de tales referencias, de tal modo que sus formas de onda no se traslapen y cumplan con las especificaciones del 8080.

El generador de tiempo es un cristal oscilador de 20 MHz controlado. El nivel de voltaje de esta señal no es compatible con TTL como las otras señales que llegan al 8080, es por esto que la base de tiempo requiere de impulsores de alto nivel para efectuar la interfaz con el 8080.

#### Impulsor de canal bidireccional y controlador del sistema

La memoria del sistema y los dispositivos de ent/sal se comunican con el CPU a través del canal de datos bidireccional. El controlador del sistema se emplea para colocar y remover datos del canal de datos dentro de las secuencias de tiempo correctas que indica el CPU.

Las líneas de datos del CPU 8080, memoria y dispositivos de ent/sal son de tres estados, es decir, sus impulsores de salida tienen la facultad de ser forzados a un tercer estado de alta impedancia (diferente de "0" y "1" lógicos), con lo cual los datos pueden ser efectivamente removidos del circuito.

La técnica de un canal de tres estados permite al diseñador construir un sencillo canal de datos bidireccional de ocho bits en paralelo, en el cual simplemente coloca o retira información seleccionando o no la memoria o los dispositivos de ent/sal desde el canal de control, es decir, mediante el controlador del sistema.

Sólo resta añadir que los elementos básicos de un sistema de computación descritos en esta sección, CPU, memoria y ent/sal, se encuentran físicamente en la microcomputadora Intellec MDS, distribuidos en sus respectivos módulos según se expuso en la sección 1.2.2.

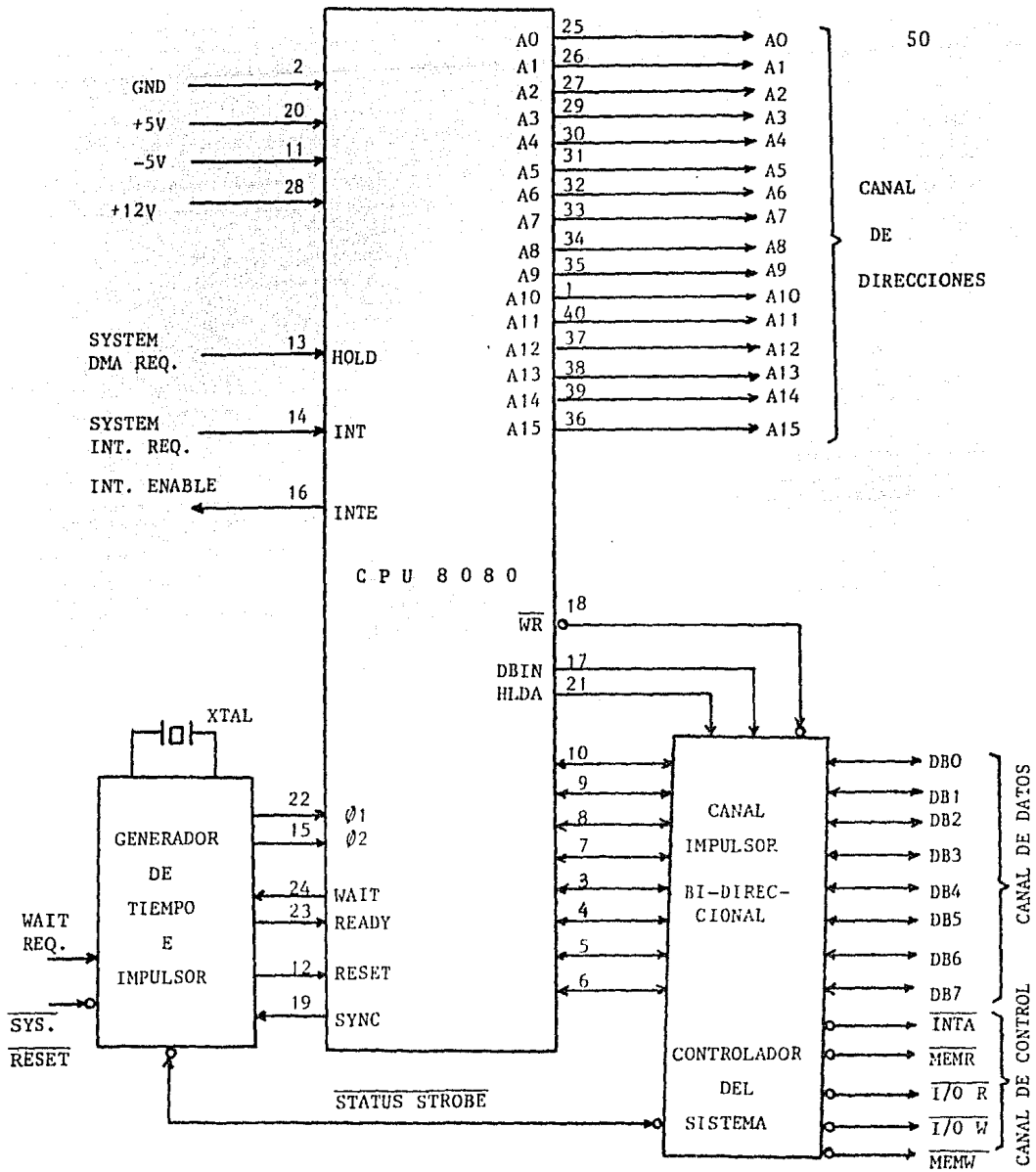


Figura 3.2 El bloque del CPU en detalle, formado por el CPU, Generador de Tiempo y Controlador del Sistema. La numeración del 1 al 40 son los contactos del circuito integrado 8080.

El microprocesador 8080 es una Unidad Central de Procesamiento (CPU) de 8 bits en paralelo de propósito general (figura 3.3).

- Transfiere datos e información de estado interna por una línea de datos bidireccional (canal de datos), representada por las líneas  $D_0 - D_7$ .
- Las direcciones de la memoria y de los dispositivos periféricos se transmiten por el canal de direcciones, representado por las líneas  $A_0 - A_{15}$ .
- Consta de seis salidas de control y de tiempo: SYNC, DBIN, WAIT,  $\overline{WR}$ , HLDA, INTE.
- Recibe cuatro entradas de alimentación: +12 V, +5 V, -5 V, GND.
- Acepta cuatro entradas de control: READY, HOLD, INT, RESET.
- Recibe también dos entradas de reloj:  $\emptyset 1$ ,  $\emptyset 2$ .

La arquitectura del CPU 8080 está formada por las siguientes unidades funcionales:

- El arreglo de registros y la lógica direccional.
- La unidad aritmética lógica (ALU).
- El registro de instrucciones y la sección de control.
- Canal de datos bidireccional de tres estados.

La figura 3.4 muestra el diagrama de bloques de la estructura interna del CPU 8080.

#### Arreglo de registros

La sección de registros consiste de un arreglo estático de RAM, organizado en seis registros de 16 bits cada uno. Estos registros son:

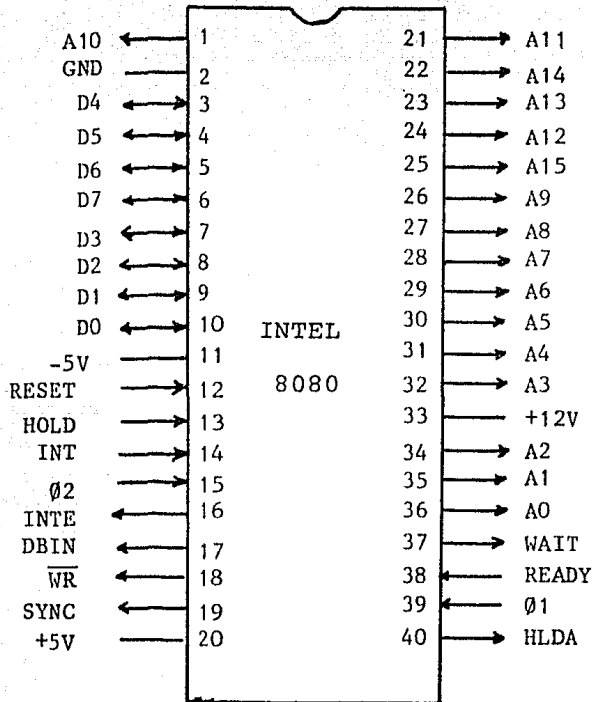


Figura 3.3 Contactos de entrada, salida,  
y alimentación del CPU 8080.

- El contador de programa (PC)
- El apuntador del apilador (SP)
- Seis registros de 8 bits cada uno de propósito general, dispuestos por parejas: (B,C), (D,E), (H,L).
- Un registro temporal par: (W,Z).

El contador de programa (PC) conserva la dirección de memoria del programa corriente de instrucciones, siendo automáticamente incrementado durante la búsqueda de cada nueva instrucción.

El apuntador del apilador (SP) mantiene la dirección de la siguiente localidad disponible de la pila en la memoria. El apuntador del apilador puede ser inicializado para usar cualquier porción de RAM como una pila de memoria. El apuntador del apilador es decrementado cuando se "empuja" un dato en la pila, y es incrementado cuando el dato se "saca" de la pila.

Los seis registros de propósito general pueden usarse como registros sencillos de 8 bits o como registros pares de 16 bits.

El registro temporal par W,Z no se puede direccionar mediante el programa del usuario, y su uso se limita a la ejecución interna de instrucciones.

Los bytes de datos de 8 bits pueden ser transferidos entre el canal interno y el arreglo de registros mediante el multiplexor y el selector de registros.

Las transferencias de 16 bits son factibles entre el arreglo de registros y el "latch" de direcciones o el circuito incrementador/decrementador.

El "latch" de direcciones recibe datos de cualquiera de los tres registros pares e impulsa al "buffer" de salida de direcciones ( $A_0 - A_{15}$ ), así como al circuito incrementador/decrementador.

El circuito incrementador/decrementador recibe datos del "latch" de direcciones y los envía al arreglo de registros.

Resumiendo, los datos de 16 bits pueden ser incrementados, decrementados, o simplemente transferidos de unos registros a otros.

### Unidad Aritmética Lógica (ALU)

Esta unidad consta de los siguientes registros:

- Un acumulador de 8 bits
- Un acumulador temporal de 8 bits (ACT)
- Un registro de banderas de 5 bits: cero, acarreo, signo, paridad y acarreo auxiliar.
- Un registro temporal de 8 bits (TMP)

En la unidad aritmética lógica se efectúan las operaciones aritméticas, lógicas y de rotación.

La unidad es alimentada por el registro temporal (TMP), el acumulador temporal (ACT) y el flip-flop de acarreo. El resultado de las operaciones puede ser transferido al canal interno o al acumulador. De igual modo, las operaciones efectuadas alimentan de manera automática al registro de banderas.

El registro temporal (TMP) recibe información del canal interno, y puede enviarla toda o en partes a la unidad aritmética lógica, al registro de banderas o al canal interno.

El acumulador puede ser cargado mediante el ALU o mediante el canal interno, y puede transferir datos al acumulador temporal (ACT) o al canal interno.

El contenido del acumulador y el flip-flop de acarreo auxiliar pueden ser probados para efectuar correcciones decimales.

### Registro de instrucciones y control

Durante la búsqueda de instrucciones, el primer byte de una instrucción es transferido del canal interno al registro de instrucciones de 8 bits.

El contenido del registro de instrucciones va quedando a disposición del decodificador de instrucciones. La salida del decodificador, en combinación con las señales de tiempo, suministra las señales de control para el arreglo de registros, el ALU y el "buffer" de datos. Además, las salidas del decodificador de instrucciones y las señales externas de control, alimentan la sección de tiempo y control, la cual, a su vez, genera las señales de estado y produce los ciclos de tiempo.

### "Buffer" del canal de datos

El "buffer" de 8 bits bidireccional de tres estados se emplea para aislar el canal interno del CPU del canal externo de datos ( $D_0 - D_7$ ).

En modo de salida, el contenido del canal interno es cargado en un "latch" de 8 bits, que en la debida oportunidad impulsa los "buffers" de salida del canal de datos.

Durante el modo de entrada, o cuando no existen operaciones de transferencia, los "buffers" son apagados.

En el modo de entrada, los datos del canal externo son transferidos al canal interno.

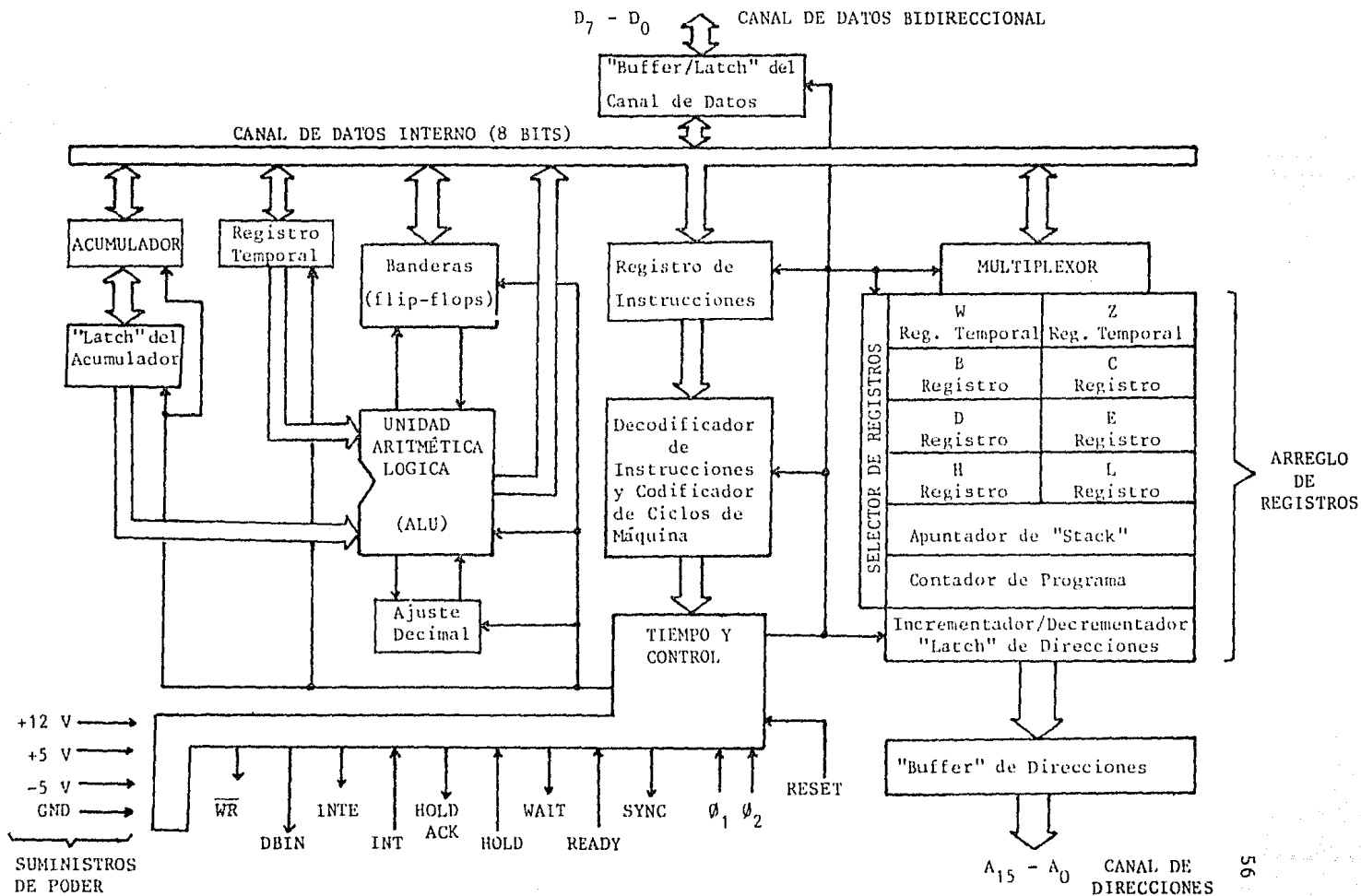


Figura 3.4 Arquitectura del microprocesador 8080.



y el exterior.

Como en cualquier otro sistema de computación, el CPU 8080 puede comunicarse con dispositivos o estructuras fuera de su arreglo normal de memoria. Dispositivos tales como teclados, cinta de papel, discos blandos, impresoras, "displays" y otras estructuras de control se usan para mostrar o almacenar los resultados de la actividad computacional.

El método más común para configurar un dispositivo de ent/sal logrando la máxima eficiencia y reduciendo el número de componentes, consiste en decodificar el canal de direcciones a través de "chip selects" exclusivos que habilitan al dispositivo de ent/sal direccionado.

La figura 3.5 es un ejemplo de un sistema típico de ent/sal, empleando tres componentes: 8212, 8255 y 8251. Este sistema puede emplearse para la interfaz de periféricos o para comunicación de datos.

El circuito integrado 8212 es un puerto de ent/sal de 8 bits, y el 8255 es una interfaz periódica programable de propósito general. Estos circuitos no son necesarios para el desarrollo del presente trabajo, por lo mismo, no se estudiarán en detalle y sólo se mencionan a manera de ejemplo.

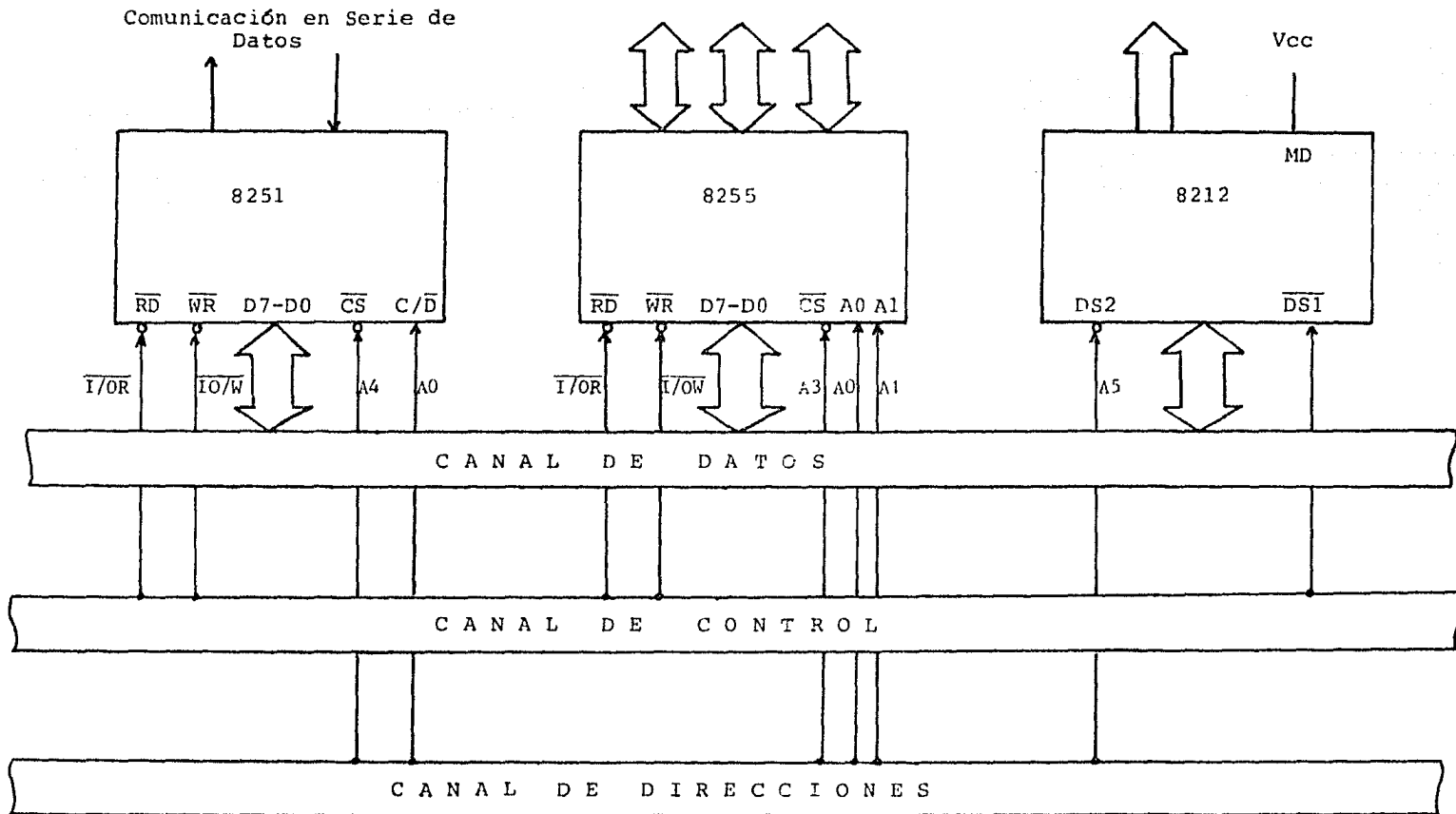


Figura 3.5 Ejemplo de un sistema típico de interfaz de entrada/salida.

El 8251 es una interfaz programable de comunicación, circuito diseñado para comunicación de datos en sistemas de microcomputadoras, por lo cual resulta el elemento idóneo para el diseño de la interfaz entre la microcomputadora y la computadora de propósito general.

El 8251 es un Transmisor/Receptor Universal Síncrono/Asíncrono, el cual se programa mediante el CPU para operar con casi cualquier técnica de transmisión de datos en serie, ya que acepta caracteres en paralelo del CPU y los serializa. De igual modo, acepta datos en serie del equipo de comunicación de datos y los convierte en paralelo para ser transmitidos al CPU.

Por ser este componente de vital importancia para el diseño de la interfaz, se estudiará en detalle a lo largo de la sección 3.2.

## 3.2 EL TRANSMISOR/RECEPTOR UNIVERSAL SÍNCRONO/ASÍNCRONO

### 3.2.1 Formatos de comunicación.

El diseño de una tarjeta de interfaz para comunicar a la microcomputadora con el exterior se lleva a cabo eficazmente gracias al circuito integrado 8251 de Intel (figura 3.6).

El 8251 es un transmisor/receptor universal síncrono/asíncrono (USART) que opera con tres diferentes formatos serie de comunicaciones, a la vez que permite realizar la interfaz con diversas clases de periféricos.

Esto significa que el USART permitirá transmisiones en "half-duplex" y "full-duplex" si se emplean modems para cualquiera de los tres formatos serie: síncrono, asíncrono e isosíncrono (la transmisión isosíncrona ocurre cuando se transmiten datos en formato asíncrono usando un modem síncrono).

Tanto en modo síncrono como asíncrono, el USART puede operar con caracteres de 5, 6, 7 u 8 bits. Es posible elegir entre paridades par e impar y checar posteriormente mediante "software" esta condición.

En modo asíncrono se agrega un "start bit" al dato, y se añaden además 1, 1.5 o 2 "stop bit" según se requiera.

El USART se programa para aceptar velocidades de reloj de 16 o 64 veces la velocidad requerida en bauds.

El USART no provee todas las señales de la interfaz EIA-RS-232-C, y tampoco es compatible con los niveles de voltaje de ésta.

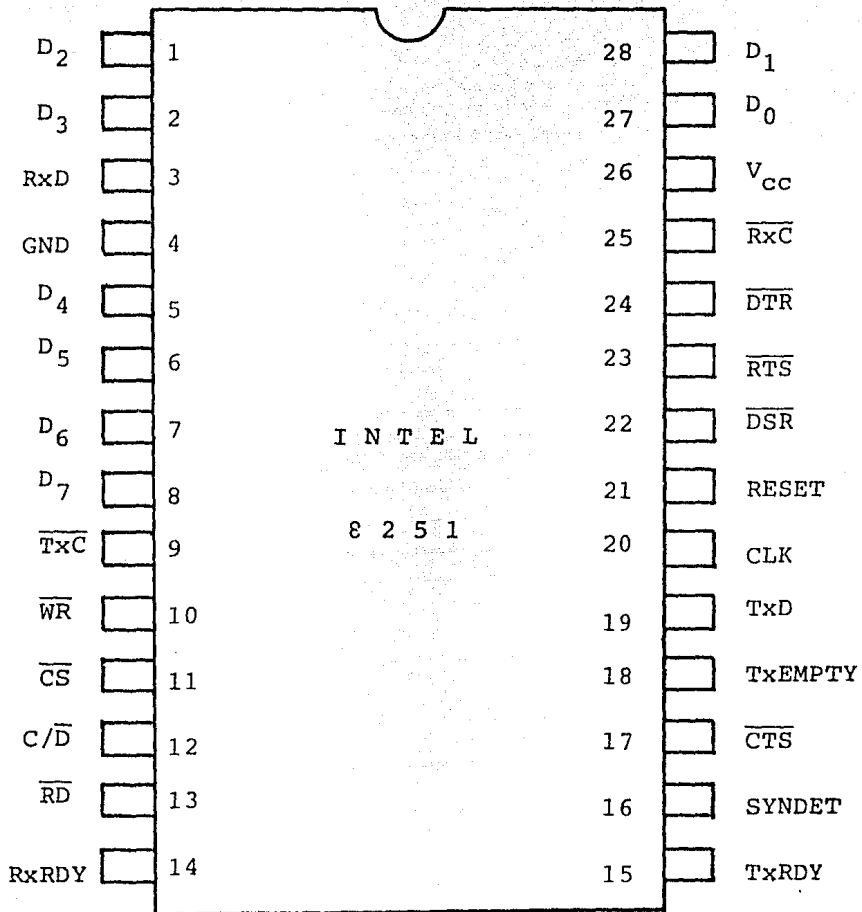


Figura 3.6 Contactos del Transmisor/Receptor  
 Universal Sincrono/Asincrono 8251  
 de Intel.

El USART consiste básicamente de cinco secciones, cada una de las cuales se comunica con las otras mediante un canal de datos interno (figura 3.7). Estas secciones son las siguientes:

RECEPTOR

TRANSMISOR

CONTROL DEL MODEM

CONTROL DE LECTURA/ESCRITURA

"BUFFER" DE ENTRADA/SALIDA

#### RECEPTOR

La función del receptor es admitir datos en serie en el contacto RxD, los cuales transforma posteriormente en datos en paralelo con el formato apropiado.

Cuando el USART está programado en modo asíncrono, se encuentra a la espera de un nivel bajo en RxD, siempre y cuando esté listo para aceptar un carácter, es decir, que no se encuentre en el proceso de estarlo recibiendo.

Cuando aparece el nivel bajo, el USART asume que se trata de un "start bit", con lo cual habilita un contador interno. Transcurrido el tiempo equivalente a medio bit, se vuelve a muestrear la línea RxD, y si continúa en nivel bajo, probablemente ha sido recibido un "start bit" válido, con lo que el 8251 procede a ensamblar el carácter.

Si por el contrario, la línea RxD está a nivel alto durante el muestreo, ocurrió un pulso de ruido, o bien, el receptor fue habilitado a la mitad de la transmisión de un carácter. En ambos casos, el receptor aborta la operación y se prepara para aceptar un nuevo carácter.

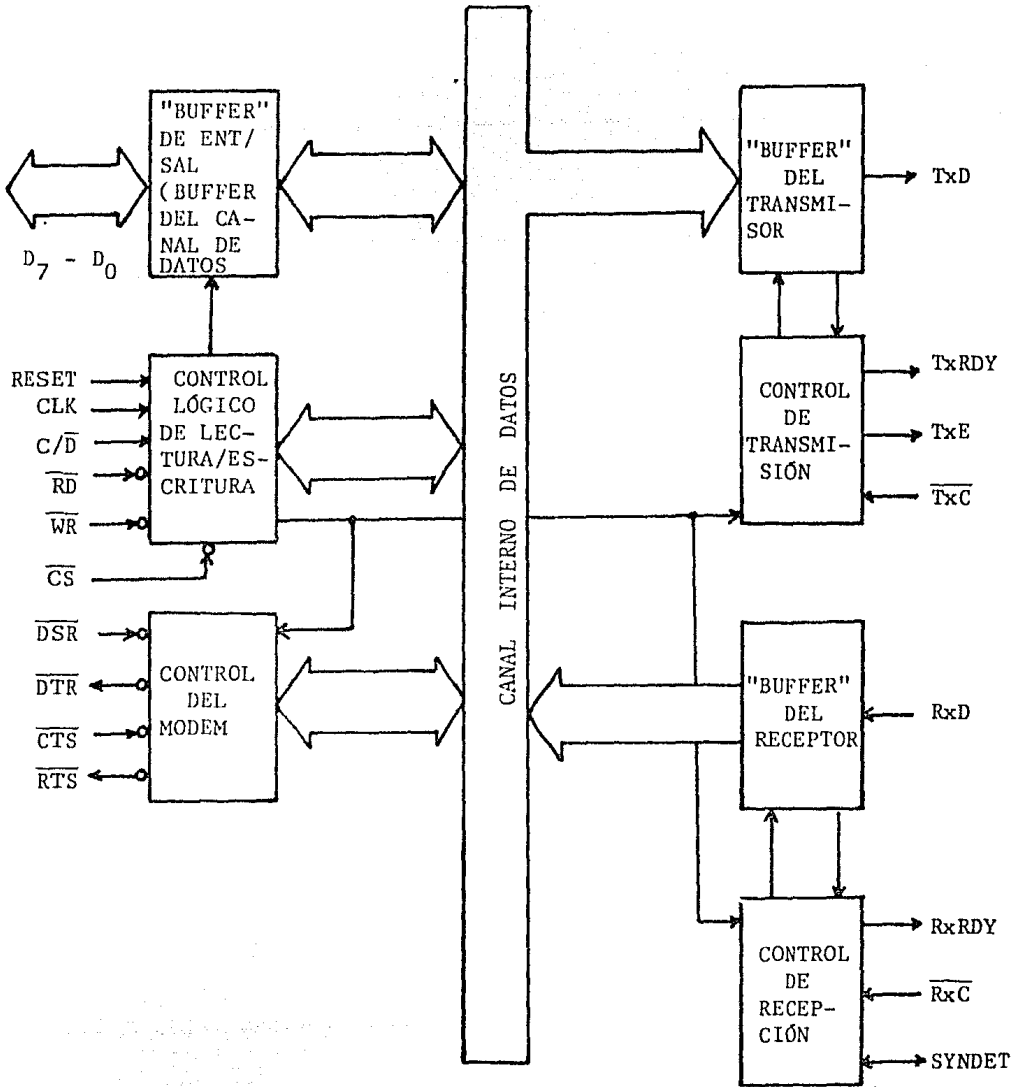


Figura 3.7 Diagrama de bloques de la configuración interna del USART 8251.

Tras la recepción exitosa de un "start bit", el 8251 registra los bits de datos, paridad y "stop", procediendo a transmitir los datos por el canal interno de datos hacia el registro de recepción de datos. La señal RxRDY es validada para indicar que un caracter se encuentra disponible.

El receptor admite las señales RxRDY, SYNDY,  $\overline{\text{RxC}}$  y RxD.

### TRANSMISOR

El transmisor acepta datos en paralelo del procesador, añade la información apropiada y la serializa para transmitirla por el contacto TxD.

En la transmisión asíncrona, siempre se añaden un "start bit", 1, 1.5 o 2 "stop bit", y opcionalmente, un bit de paridad, la cual puede ser par o impar.

Tanto en modo asíncrono como síncrono, la transmisión es inhibida mientras no se habilite la entrada  $\overline{\text{CTS}}$ .

El transmisor acepta las señales TxD, TxRDY, TxE y  $\overline{\text{TxC}}$ .

### CONTROL DEL MODEM

Esta sección permite la generación de la señal  $\overline{\text{RTS}}$  y la detección de la señal  $\overline{\text{CTS}}$ . Proporciona además una entrada y una salida de propósito general, denominadas  $\overline{\text{DSR}}$  y  $\overline{\text{DTR}}$  respectivamente.

La señal  $\overline{\text{DTR}}$  se habilita mediante programación, y corresponde al bit 2 de la instrucción de comando (sección 3.2.3). Por otra parte, la señal  $\overline{\text{DSR}}$  puede ser detectada y corresponde al bit 7 del registro de estado (sección 3.2.3).

El USART no da significado específico a estas señales ( $\overline{\text{DSR}}$ ,  $\overline{\text{DTR}}$ ). Normalmente DTR se asigna al modem para indicar que la terminal está lista para comunicarse, y DSR es una señal proveniente del modem indicando que éste está listo para efectuar la comunicación.



### CONTROL DE LECTURA/ESCRITURA

Es el control lógico correspondiente a la entrada y a la salida del USART, siendo su función decodificar las señales provenientes del 8080 por el canal de control en señales que admiten o envían datos a través del canal interno del USART, a la vez que controlan el canal externo de ent/sal ( $DB_0 - DB_7$ ).

A continuación se muestra la tabla de verdad de estas operaciones, de donde se desprende que una de las dos señales,  $\overline{READ}$  o  $\overline{WRITE}$ , deberá estar puesta a cero para que se lleven a cabo funciones de entrada/salida. Por otra parte, si ambas señales fuesen cero al mismo tiempo, se obtendría un estado ilegal de resultados indefinidos.

CS	C/D	READ	WRITE	Función
0	0	0	1	El CPU lee datos del USART
0	1	0	1	El CPU lee el estado del USART
0	0	1	0	El CPU escribe datos al USART
0	1	1	0	El CPU escribe comando al USART
1	X	X	X	Canal del USART fuera de operación

Respecto a la señal  $C/\overline{D}$ , un uno lógico habilita el control, con lo cual el CPU leerá el estado del USART o emitirá un comando, mientras que un cero lógico habilitará la lectura o la escritura de datos.

Si el USART no ha sido seleccionado ( $CS = 1$ ), no importará el estado de las líneas  $C/\overline{D}$ ,  $\overline{READ}$ ,  $\overline{WRITE}$ .

### "BUFFER" DE ENTRADA/SALIDA

Está formado por tres elementos, el "buffer" de estado, el "buffer" receptor de datos, y el "buffer" de transmisión de datos/comando (figura 3.8).

Mientras que existen dos registros que almacenan datos para ser transmitidos al CPU (el de estado y el de

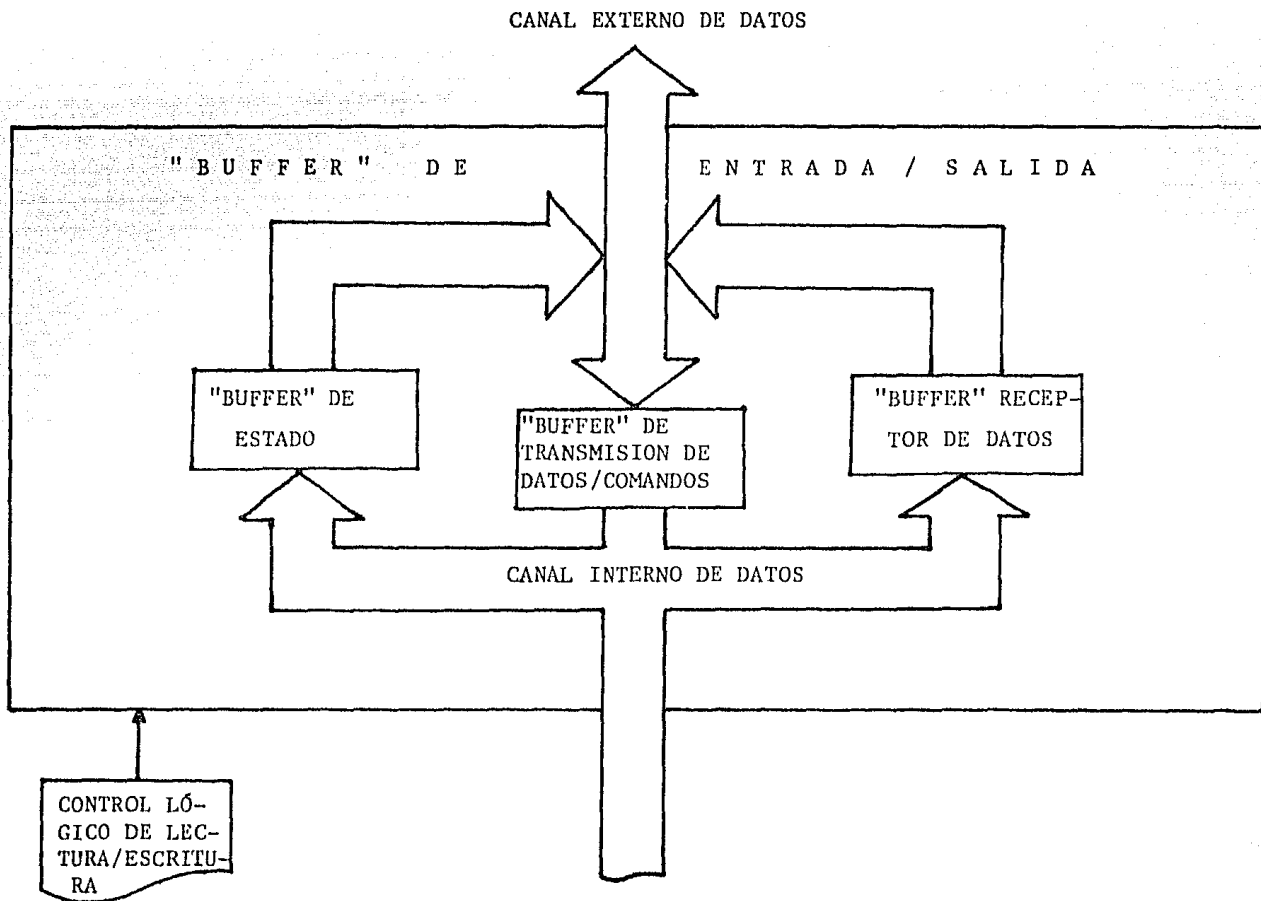


Figura 3.8 Diagrama de bloques del "buffer" de entrada/salida.

recepción), solamente hay un registro que almacena los datos que llegan al USART provenientes del CPU. La partición de este registro para aceptar lo mismo datos que comandos implica la necesidad de asegurar que el USART no tiene datos almacenados en este registro antes de enviar un comando. Esto puede hacerse con la señal TxRDY, la cual, estando en nivel bajo, indica que no deben transferirse datos ni comandos al USART.

Esta prueba no es indispensable para el funcionamiento del USART, pero si no se efectúa, se corre el riesgo de transmitir datos o comandos equivocadamente.

Las señales de interfaz del USART 8251 pueden ser clasificadas en dos grupos:

- Señales referidas al CPU
- Señales referidas al dispositivo

El primer grupo está especialmente diseñado para acoplarse a la microcomputadora, mientras que el segundo está pensado para realizar una interfaz con un modem o algún dispositivo similar, como un teleimpresor (TTY) o una terminal de video (CRT), empleando las características de la interfaz estándar EIA-RS-232-C.

Las señales de interfaz del USART 8251 se describen a continuación.

Nombre	Número de contacto	Dirección	Descripción
TxRDY	15	S	"Transmitter Ready". Esta señal de salida indica al CPU que el USART está listo para aceptar un dato o comando.
TxE	18	S	"Transmitter Empty". Un 1 en la salida indica que la conversión paralelo/serie ha sido realizada.
RxRDY	14	S	"Receiver Ready". Esta salida se pone a 1 para indicar que el 8251 ha recibido un caracter en la entrada serie y está listo para transferirlo al CPU.
SYNDET	16	E/S	"Synch Detect". Esta línea sólo se usa en el modo síncrono.
$\overline{\text{DTR}}$	24	S	"Data Terminal Ready". Señal de salida de propósito general que se puede poner a 0 programando un 1 en el bit 1 de la instrucción de comando.
DSR	22	E	"Data Set Ready". Señal de entrada de propósito general cuyo estado puede ser probado por el CPU. Corresponde al bit 7 del registro de estado.
RTS	23	S	"Request to Send". Salida de propósito general equivalente a $\overline{\text{DTR}}$ . Se emplea para habilitar al modem para transmitir. Se programa poniendo a 1 el bit 5 de la instrucción de comando.
$\overline{\text{CTS}}$	17	E	"Clear to Send". Un 0 en esta entrada habilita al USART para transmitir. Esta señal es normalmente generada por el modem en respuesta a $\overline{\text{CTS}}$ .
$\overline{\text{RxC}}$	25	E	"Receiver Clock". Este reloj controla la velocidad de recepción de los datos que llegan al USART.
RxD	3	E	"Receiver Data". Los caracteres recibidos en este contacto en forma serie son ensamblados en paralelo.
$\overline{\text{TxC}}$	9	E	"Transmitter Clock". Controla la velocidad a la cual los caracteres son transmitidos por el USART.
TxD	19	S	"Transmitter Data". Los caracteres en paralelo provenientes del CPU son transmitidos en serie por esta línea.

Nombre	Número de contacto	Dirección	Descripción
V <sub>CC</sub>	26	E	Suministro de +5 volts
GND	4	E	Señal de tierra
CLK	20	E	Esta entrada genera el tiempo interno del dispositivo. La frecuencia debe ser por lo menos 4.5 veces mayor que la del reloj del transmisor y del receptor en modo asíncrono.
RESET	21	E	Un 1 lógico en esta entrada efectúa un "reset" en el 8251.
DB <sub>7</sub> -DB <sub>0</sub>	8,7,6,5, 2,1,28, 27	E/S	Las señales DB son un canal de tres estados compatible con el canal de datos del CPU. Este canal maneja control, estado y datos.
CS	11	E	"Chip Select" o "Chip Enable". Un cero en esta entrada habilita la comunicación entre el CPU y el USART.
C/D	12	E	"Control/Data". Durante la operación de lectura, el CPU lee el estado con 1 y los datos con 0. En la operación de escritura el USART interpreta con 1 comando y con 0 datos.
$\overline{RD}$	13	E	Un 0 en la entrada indica al USART poner el estado o el dato en el canal de datos.
$\overline{WR}$	10	E	Un 0 en la entrada indica al USART aceptar la información del canal de datos como comando o como dato.

Según se mencionó en la sección 3.2.1, el USART se puede operar en modo síncrono o en modo asíncrono, llevándose a cabo la selección de uno u otro modos por medio de una serie de salidas de control.

La programación de las salidas de control debe ocurrir entre el tiempo en que el USART recibe un "reset" y el tiempo empleado en la transferencia de datos. Ya que el USART necesita esta información para estructurar su lógica interna, es esencial completar la inicialización de éste antes de que se realice cualquier tipo de transferencia de datos, incluida la lectura de estado.

La primera operación a seguir después del "reset" deberá ser, por tanto, la carga del Registro de Modo de Control. Esta carga se efectúa asumiendo que la primera salida de control siguiendo a un "reset" ( $\overline{CS}=0$ ,  $C/\overline{D}=1$ ,  $\overline{RD}=1$ ,  $\overline{WR}=1$ ), corresponde a la información del Registro de Modo de Control.

La instrucción del formato de modo de control puede ser considerada como cuatro campos de dos bits. El primer campo de dos bits ( $D_1D_0$ ) determina el modo de operación del USART (síncrono o asíncrono). El segundo campo ( $D_3D_2$ ) especifica el número de bits en el carácter de datos. El tercer campo ( $D_5D_4$ ) controla la paridad del carácter de datos. El cuarto campo ( $D_7D_6$ ) tiene dos significados, según se vaya a operar en modo síncrono o asíncrono. En el primer caso, este campo controla los procesos de sin-

cronización, mientras que en modo asíncrono controla el número de "stop bit" asociados al carácter de datos.

Una vez cargado el registro de modo de control, la siguiente operación consiste en enviar un carácter de comando al USART. Esta instrucción de comando lleva a cabo el control de la operación del USART dentro del marco establecido por la instrucción de modo.

En cualquier momento se puede determinar el estado del USART, el cual puede ser leído por el programa de control. El estado se encuentra siempre presente en el Registro de Estado.

La habilidad para cambiar el modo de operación del USART por medio de "software", hace que éste sea un dispositivo ideal para la implementación de enlaces de comunicación en serie.

El diseño de este tipo de enlaces debe incluir factores tales como la clase de transmisión y el formato de seados, los requerimientos adecuados de verificación de errores, la operación "half-duplex" o "full-duplex" según se desee, y por supuesto, la implementación física del eslabonamiento.

A continuación se describen los formatos de los registros del USART 8251:

- Registro del Modo de Control
- Registro de Instrucción de Comando
- Registro de Estado

Formato de la instrucción de modo:

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------

D<sub>1</sub> - D<sub>0</sub> ----- Factor de cambio en bauds:

---

00 = modo síncrono  
 01 = asíncrono x 1  
 10 = asíncrono x 16  
 11 = asíncrono x 64

---

D<sub>3</sub> - D<sub>2</sub> ----- Longitud del caracter:

---

00 = 5 bits  
 01 = 6 bits  
 10 = 7 bits  
 11 = 8 bits

---

D<sub>5</sub> - D<sub>4</sub> ----- Control de paridad:

---

X0 = sin paridad  
 01 = impar  
 11 = par

---

D<sub>7</sub> - D<sub>6</sub> ----- Control de armadura (asíncrono):

---

00 = no válido  
 01 = 1 stop bit  
 10 = 1.5 stop bit  
 11 = 2 stop bit

---

Control de caracter "SYN" (síncrono):

---

X0 = SYN interno  
 X1 = SYN externo  
 0X = doble caracter SYN  
 1X = un solo caracter SYN

---



Formato de la instrucción de comando:

EH	IR	RTS	ER	SBRK	RxE	DTR	TxE
D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

- TxE ----- Habilitación para transmitir:  
 1 = habilitado  
 0 = no habilitado
- 
- DTR ----- Data Terminal Ready:  
 una señal alta provocará que DTR tenga una salida de cero
- 
- RxE ----- Habilitación para recibir:  
 1 = habilita RxRDY  
 0 = deshabilita RxRDY
- 
- SBRK ----- Envío de caracter "break":  
 1 = provoca una señal baja en TxD  
 0 = operación normal
- 
- ER ----- "Reset" de un error:  
 1 = ejecuta un "reset" en TODAS las banderas de error: FE, OE, PE
- 
- RTS ----- Request to Send:  
 una señal alta produce en RTS una salida de cero
- 
- IR ----- "Reset" interno:  
 una señal alta pone al 8251 en el formato de la instrucción de modo nuevamente
- 
- EH ----- Entrada de modo de búsqueda ("hunt"):  
 1 = habilita la búsqueda del caracter síncrono "SYN"
-

Formato del registro de estado:

DSR	SYNDET	FE	OE	PE	TxE	RxRDY	TxRDY
D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

DSR, SYNDET, TxE, RxRDY, TxRDY, se definen igual que sus respectivos contactos de entrada/salida (sección 3.2.2), exceptuando que TxRDY no está condicionado por  $\overline{\text{CTS}}$  o por TxENable.

---

FE ----- Error de armadura (sólo en modo asíncrono):

La bandera FE es puesta en "set" cuando no se detecta ningún "stop" bit válido al final de cada carácter. Se pone en "reset" mediante el bit ER del comando de instrucción. FE no inhibe la operación del 8251.

---

OE ----- Error de exceso ("overrun"):

La bandera de OE es puesta en "set" cuando el CPU no ha leído un carácter antes de que el siguiente esté disponible. Es puesta en "reset" por el bit ER de la instrucción de comando. OE no inhibe la operación del 8251, pero se pierde el carácter previo a "overrun".

---

PE ----- Error de paridad:

La bandera de PE es puesta en "set" cuando un error de paridad es detectado. Es puesta en "reset" por el bit ER de la instrucción de comando. PE no inhibe la operación del 8251.

---

## 3.3.1 Base de tiempo.

El primer planteamiento para diseñar una interfaz es la elección de una base de tiempo adecuada para el USART, la cual estará en función de la velocidad de transmisión entre la microcomputadora y la computadora de propósito general.

Los teletipos más sencillos funcionan a velocidades que oscilan entre los 100 y los 300 bauds aproximadamente, mientras que la computadora HP 3000 puede transmitir, en modo asíncrono, a velocidades hasta de 2400 bauds, tal como se vió en la sección 1.3.3.

A partir de estos parámetros se deduce que una base de tiempo que no necesariamente exceda de 2400 bauds, y que posea la flexibilidad de transmitir por debajo de esta velocidad, satisface ampliamente los requerimientos de diseño.

Según se expuso en la sección 3.3.2, entre las señales de interfaz del USART se encuentran tres señales de reloj:

- $\overline{\text{RxC}}$  que controla la velocidad de recepción de los datos que llegan al USART.

- $\overline{\text{TxC}}$  que es el control de la velocidad a la cual los datos serán transmitidos por el USART.

- CLK es la entrada que genera el tiempo interno del dispositivo, y cuya frecuencia deberá ser por lo menos 4.5

veces mayor que las frecuencias de  $\overline{RxC}$  y  $\overline{TxC}$  en modo a-síncrono.

Habiendo establecido la velocidad de transmisión en un intervalo de 300 a 2400 bauds, y sabiendo que el Sistema de Desarrollo de Microprocesadores suministra una frecuencia de 9.8304 MHz, la solución práctica para dotar de una base de tiempo al USART, será configurar un circuito que permita reducir la frecuencia del sistema a una frecuencia afín con la velocidad de transmisión deseada.

Este circuito puede implementarse a partir del contador binario de 4 bits SN7493A (figura 3.9). El 7493A es un contador monolítico que contiene cuatro flip-flops "master-slave" y dos entradas que proveen, respectivamente, un contador divisor-entre-dos y un contador binario de tres etapas divisor-entre-ocho (figura 3.10).

Para usar la longitud máxima de cuenta, se conecta la salida  $Q_A$  a la entrada B, aplicando los pulsos de cuenta a la entrada A. Las salidas se describen en la tabla de funciones:

ENTRADAS DE "RESET"		SALIDAS			
$R_{O(1)}$	$R_{O(2)}$	$Q_D$	$Q_C$	$Q_B$	$Q_A$
H	H	L	L	L	L
L	X	CUENTA			
X	L	CUENTA			

H = Nivel Alto      L = Nivel Bajo      X = Irrelevante

Secuencia de conteo (con la salida  $Q_A$  conectada a la entrada B):

CUENTA	S A L I D A S			
	$Q_D$	$Q_C$	$Q_B$	$Q_A$
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	L	H	L	H
6	L	H	H	L
7	L	H	H	H
8	H	L	L	L
9	H	L	L	H
10	H	L	H	L
11	H	L	H	H
12	H	H	L	L
13	H	H	L	H
14	H	H	H	L
15	H	H	H	H

Colocando un "latch" de alta velocidad entre los contadores y la señal de alta frecuencia del sistema, se logra mejorar notablemente la forma de onda de la señal de tiempo.

El circuito integrado 3404 de Intel contiene seis "latches" de alta velocidad, organizados en dos grupos independientes, uno de cuatro bits y otro de dos bits (figura 3.11). La aplicación principal de un "latch" es en registros de memoria, dirección, u otros elementos de almacenamiento.

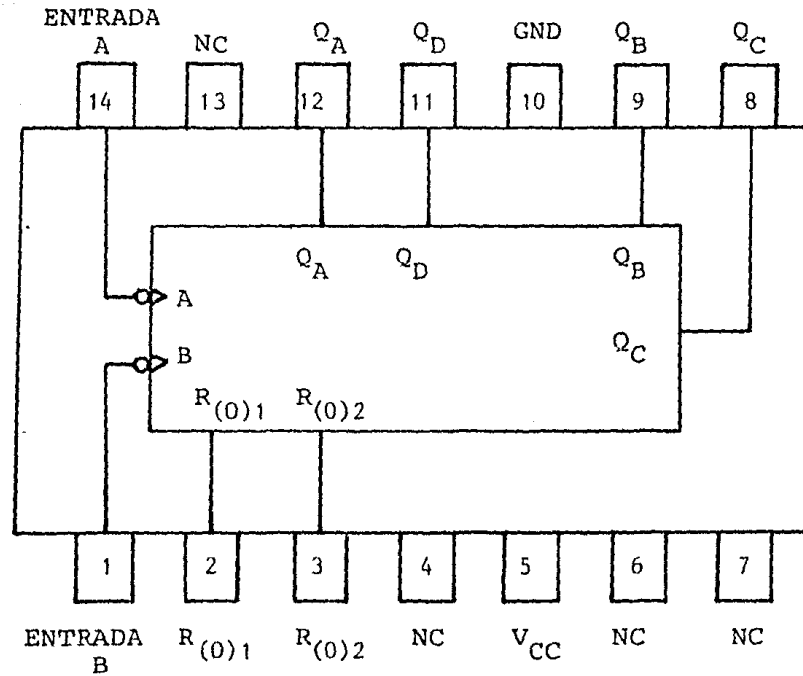
Los "latches" actúan como inversores de alta velocidad cuando la entrada "WRITE" ( $\overline{W}_1$ ) se encuentra en nivel bajo.

La figura 3.12 muestra el circuito completo de la base de tiempo. El efecto de cada flip-flop es dividir entre dos la frecuencia de entrada, y por consiguiente, el efecto total de cada 7493A, es dividir entre 16 la frecuencia de entrada, en caso de usarse la máxima longitud de cuenta.

En la sección 2.1.1 se definió en bauds a la razón de cambio de facilidades. Asumiendo que en cada segundo se transmite un bit de información, expresarse en términos de Hertz (frecuencias), será equivalente a expresarse en términos de Bauds (velocidades de transmisión). Así por ejemplo, en la salida  $Q_D$  del último 7493A, la frecuencia obtenida de 4.8 KHz es equivalente a decir 4800 bauds.

Aún alambrando completamente los tres circuitos 7493A de la figura 3.12, la frecuencia resultante de dividir tres veces sucesivas 9.8304 MHz entre 16 sería de 2400 Hz, o si se prefiere, 2400 bauds.

Esto nos da el límite superior del rango de velocidad de transmisión establecido. Para obtener velocidades inferiores, incluso 150 bauds, se puede agregar otro contador 7493A, o mejor aún, emplear las facilidades de "software" del USART (Registro de Modo de Control, sección 3.2.3), lo cual se expone en la sección 3.4.2.



NC = "No Connection"

Figura 3.9 Contador Binario de 4 Bits SN7493A.

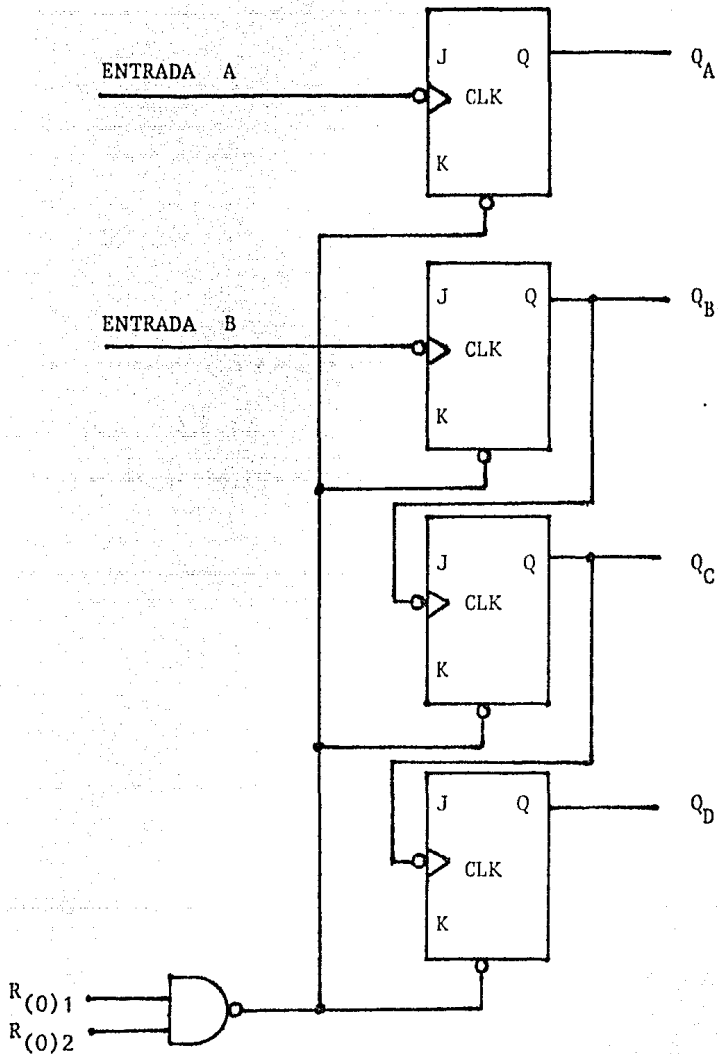


Figura 3.10 Configuración l6gica del 7493A.

Las entradas J, K, se muestran como referencia y se encuentran siempre a nivel alto.



3 4 0 4

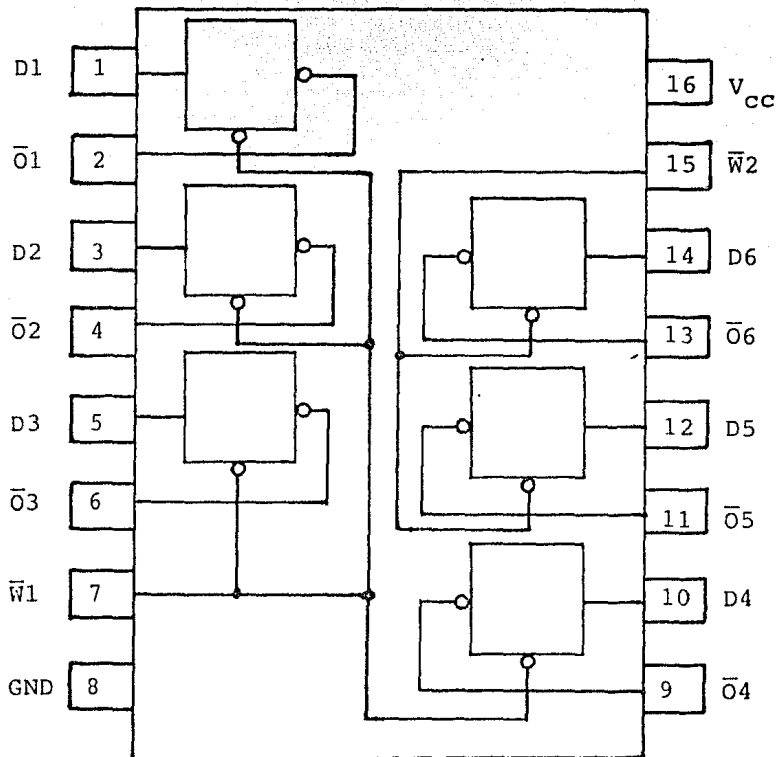


Figura 3.11 "Latch" de alta velocidad 3404.

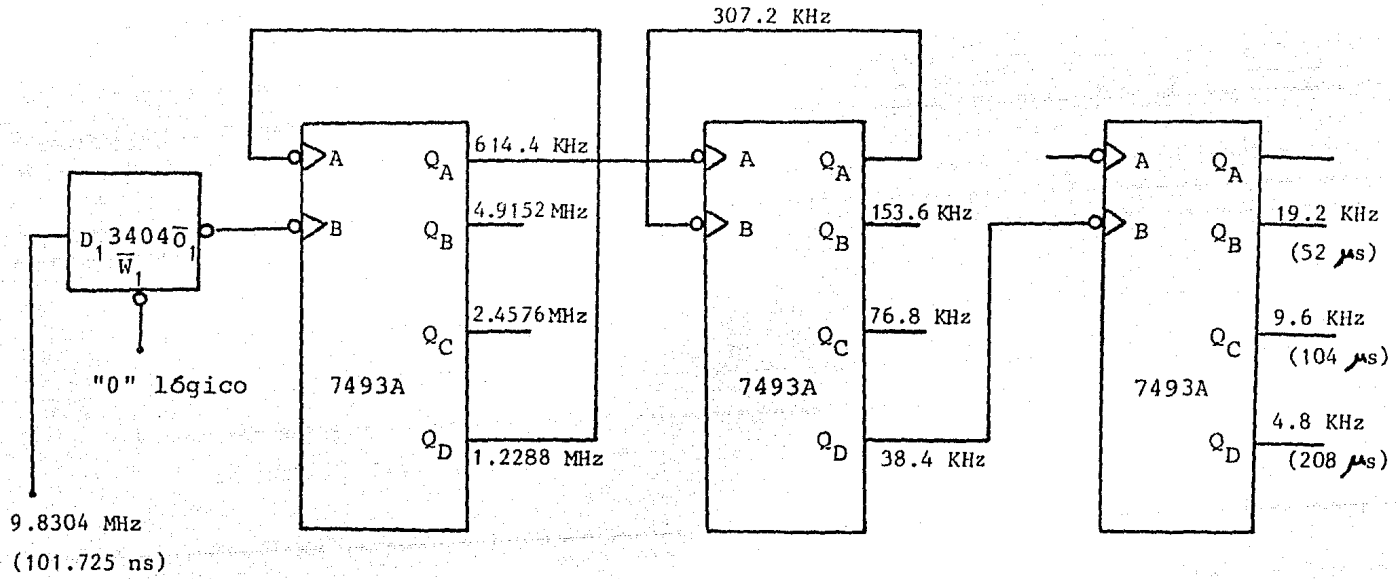


Figura 3.12 Base de tiempo para el USART.

Las señales del USART referidas al CPU pueden considerarse en dos grupos, líneas del canal de datos y líneas de control y direccionamiento del USART.

#### El Canal de Datos

Los microprocesadores como el 8080 son dispositivos MOS generalmente compatibles con dispositivos TTL. Aún cuando el acoplamiento directo es factible en sistemas pequeños con pocos componentes, a menudo es conveniente colocar "buffers" entre el microprocesador y la memoria si se piensa en añadir componentes o expandir el sistema a varias tarjetas.

De igual modo, es útil suministrar "buffers" a los componentes de ent/sal, al tiempo que se mantiene una interfaz directa y común con el canal de datos bidireccional.

El circuito integrado 8226 es una canal bidireccional de 4 bits impulsor/receptor diseñado para servir de "buffer" entre componentes de microcomputadoras (figura 3.13).

La figura 3.14 muestra el diagrama lógico del 8226. Cada línea del dispositivo consiste de dos "buffers" separados de tres estados, con el objeto de lograr un acoplamiento directo y capacidad bidireccional. De un lado del impulsor la salida de un "buffer" y la entrada del otro están unidas (DB), siendo éste el lado del dispositivo empleado para enlazar al sistema componentes tales como memorias o ent/sal, ya que su interfaz es compatible

con TTL.

Del otro lado del 8226 la entradas y salidas están separadas para obtener mayor flexibilidad, siendo posible conectarlas juntas para actuar como "buffer" del lado de un canal de datos bidireccional, tal como el canal de datos del 8080.

La entrada  $\overline{CS}$  es el selector del dispositivo. Cuando se encuentra en nivel alto los impulsores de salida son forzados a su estado de alta impedancia, mientras que la presencia de un cero lógico habilitará al dispositivo, siendo la dirección del flujo de datos función de la entrada  $\overline{DIEN}$ .

Para efectuar una interfaz del 8226 con un componente de entrada/salida, la señal  $\overline{DIEN}$  se conecta a la señal  $\overline{I/O R}$  del 8080, con lo que se asegura el flujo correcto de datos entre el dispositivo de ent/sal y el canal bidireccional de datos del sistema.

En la figura 3.15 se puede ver como enlazar el canal de datos del 8080 con el canal externo de datos del USART empleando dos componentes 8226.

#### Control y direccionamiento del USART

La mayoría de los dispositivos se habilitan mediante la entrada CS ("chip select"), lo cual, en un sistema grande supondría una confusión para distinguir que dispositivo debe ser habilitado y cuál no.

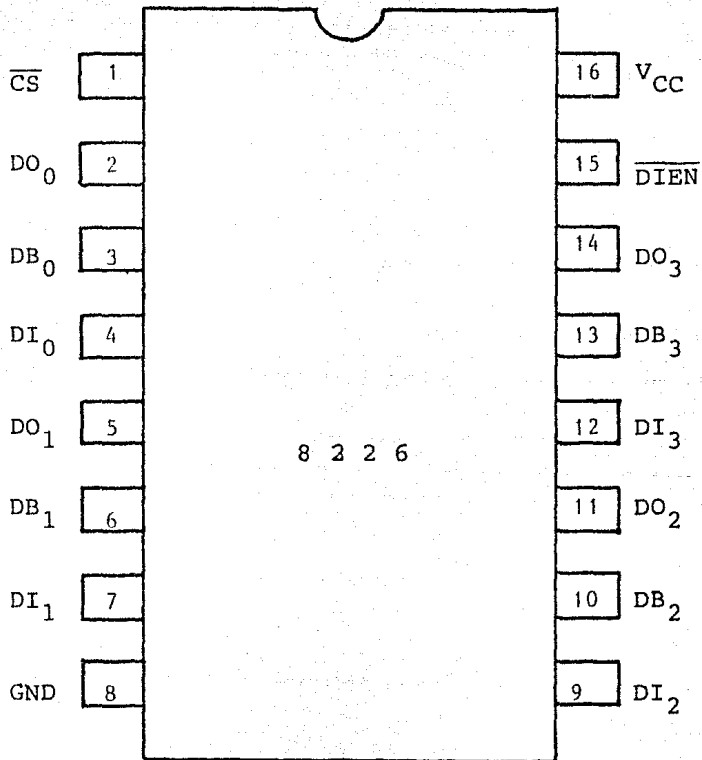


Figura 3.13 Impulsor de línea bidireccional de 4 bits en paralelo 8226.

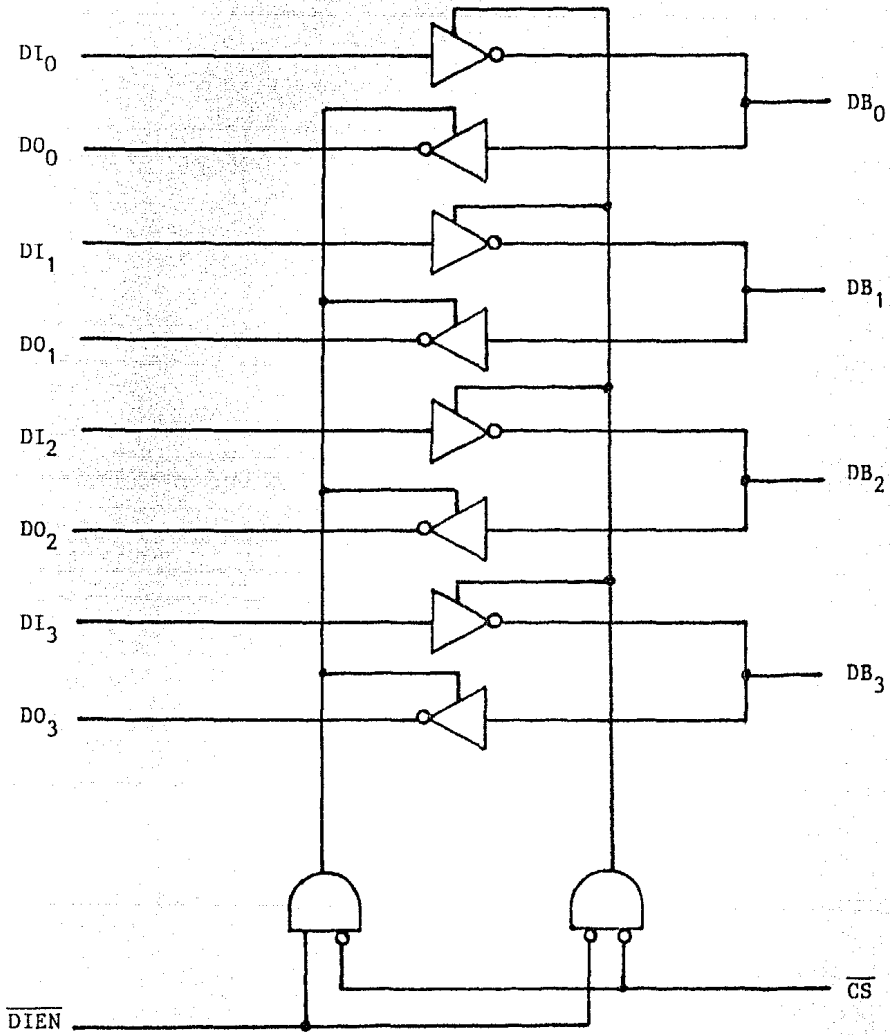


Figura 3.14 Diagrama l6gico del 8226.

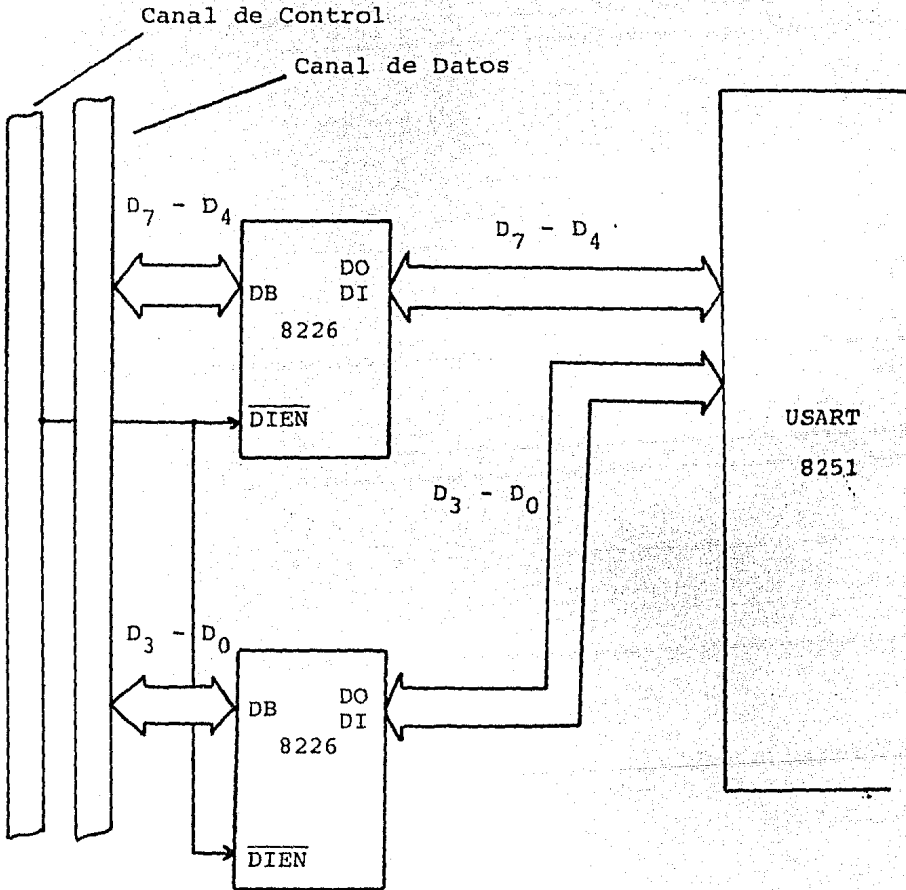


Figura 3.15 El 8226 como "buffer" entre el canal de datos del sistema y el canal de datos externo del USART.

Este problema se resuelve fácilmente creando una sola señal de habilitación para un componente o grupo de componentes a partir de una información decodificada, generalmente del canal de direcciones o de algunas líneas de éste.

Para tal efecto, se puede emplear el dispositivo 3205 (o bien el 8205, que es idéntico), pues se trata de un decodificador cuya aplicación se encuentra en la expansión de sistemas que utilizan puertos de entrada, puertos de salida, y componentes de memoria con entrada de "chip select" activa baja ( $\overline{CS}$ ) (figura 3.16).

En el caso de sistemas muy grandes, varios decodificadores 3205 pueden ser conectados en cascada, de tal modo que cada decodificador pueda impulsar desde uno hasta ocho decodificadores para expansiones arbitrarias de memoria.

El 3205 suministra una salida binaria decodificada de entre ocho salidas. Esta salida exclusiva se crea colocando un código binario de tres bits en las entradas  $A_0$ ,  $A_1$ ,  $A_2$ , y habilitando el dispositivo mediante las entradas  $\overline{E}_1$ ,  $\overline{E}_2$ ,  $E_3$ .

La figura 3.17 muestra el diagrama lógico del 3205 y su tabla de verdad, de la cual se desprende que el dispositivo no producirá una salida a menos que esté habilitado, y cuando ésto ocurra, sólo se producirá una salida (activa baja) en un contacto a la vez, dependiendo del código de entrada.



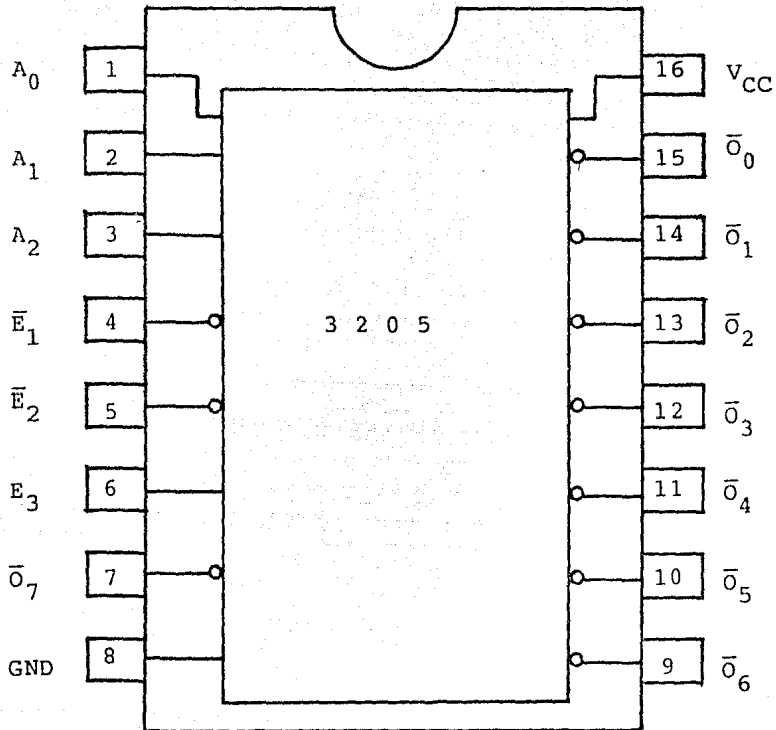
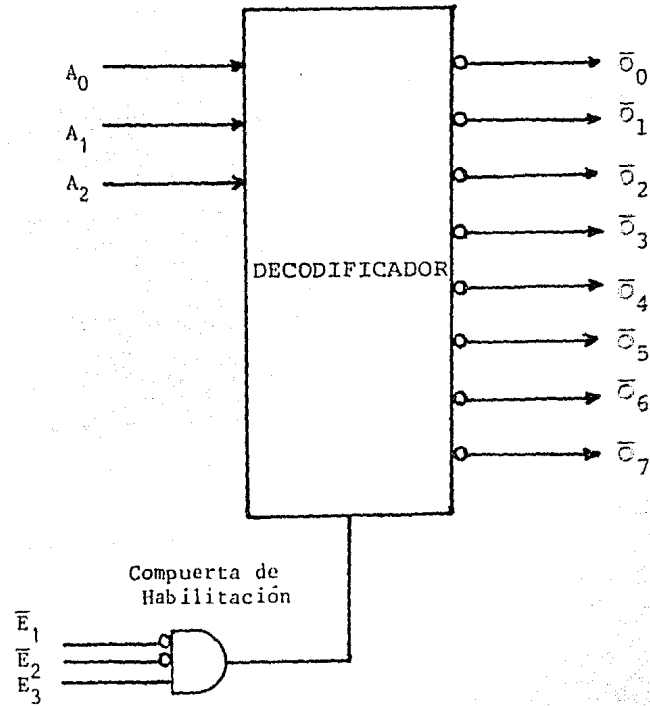


Figura 3.16 Decodificador Binario 3205.



DIRECCIÓN			HABILITACIÓN			S A L I D A S							
A <sub>0</sub>	A <sub>1</sub>	A <sub>2</sub>	E <sub>1</sub>	E <sub>2</sub>	E <sub>3</sub>	0	1	2	3	4	5	6	7
L	L	L	L	L	H	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
L	H	L	L	L	H	H	H	L	H	H	H	H	H
H	H	L	L	L	H	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
H	L	H	L	L	H	H	H	H	H	H	L	H	H
L	H	H	L	L	H	H	H	H	H	H	H	L	H
H	H	H	L	L	H	H	H	H	H	H	H	H	L
X	X	X	L	L	L	H	H	H	H	H	H	H	H
X	X	X	H	L	L	H	H	H	H	H	H	H	H
X	X	X	L	H	L	H	H	H	H	H	H	H	H
X	X	X	H	H	L	H	H	H	H	H	H	H	H
X	X	X	H	L	H	H	H	H	H	H	H	H	H
X	X	X	L	H	H	H	H	H	H	H	H	H	H
X	X	X	H	H	H	H	H	H	H	H	H	H	H

Figura 3.17 Diagrama lógico y tabla de verdad del Decodificador Binario 3205.

Mediante dos decodificadores 3205 en cascada se puede obtener una señal decodificada para habilitar al USART y a los impulsores 8226. Esta conexión aparece en la figura 3.18, donde se aprecia que el primer dispositivo se habilitará o no dependiendo del nivel presente en  $E_3$ , pues  $\bar{E}_1$  y  $\bar{E}_2$  se encuentran permanentemente a cero.

Tan pronto sea habilitado el decodificador aparecerá alguna señal activa baja en alguna de las salidas, dependiendo del código de entrada.

Esta señal baja se alimenta en la entrada  $\bar{E}_1$  del segundo decodificador para habilitarlo. Las entradas  $\bar{E}_2$  y  $E_3$  se encuentran permanentemente conectadas a cero y uno lógicos respectivamente, por lo que la habilitación del dispositivo sólo depende de  $\bar{E}_1$ .

Tan pronto queda habilitado el segundo decodificador aparecerá en éste una señal activa baja en alguna salida, de acuerdo al código de entrada seleccionado.

Esta última señal generada por el segundo decodificador será la que alimente las entradas  $\overline{CS}$  de los componentes que lo requieran (8251 y 8226) habilitándolos.

Las líneas de control del USART se conectarán como se indica en la sección 3.3.4.

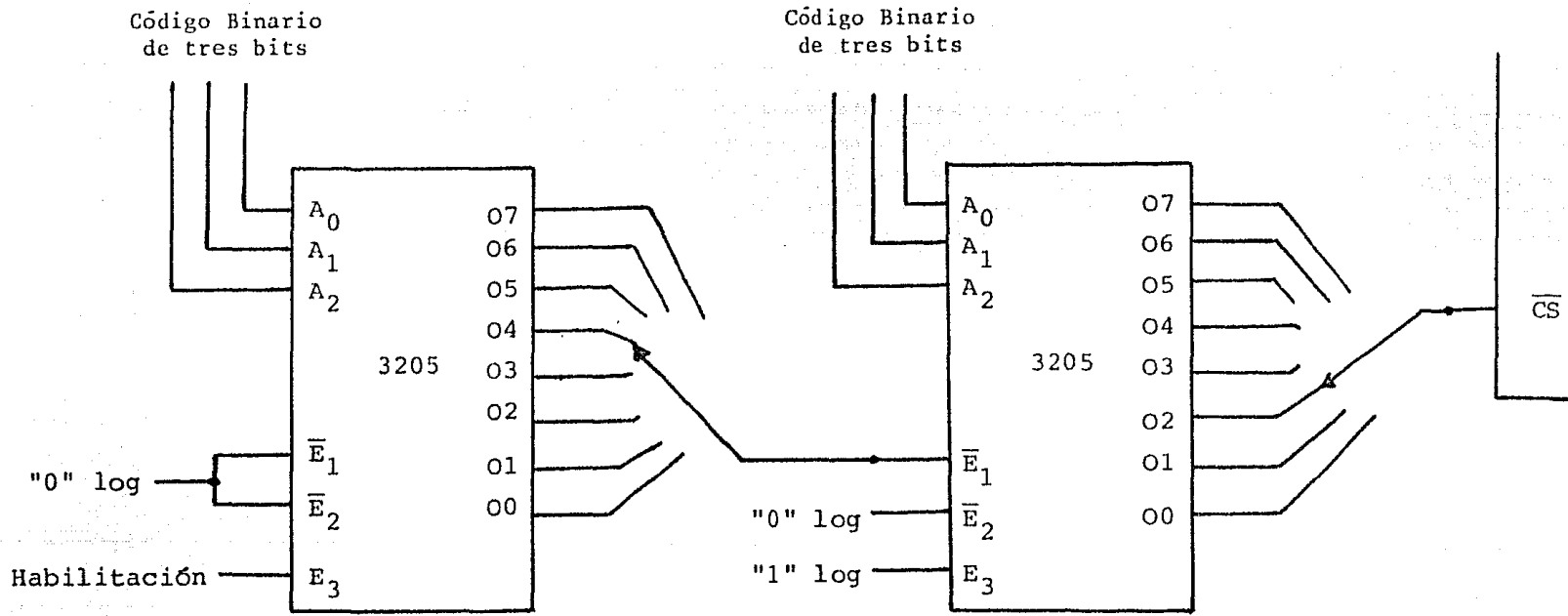


Figura 3.18 Conexión de dos decodificadores en cascada, para obtener una señal decodificada que habilite el "chip select" del sistema.

Las señales del USART referidas a la interfaz son TxD, RxD,  $\overline{\text{RTS}}$ ,  $\overline{\text{CTS}}$ ,  $\overline{\text{DSR}}$ ,  $\overline{\text{DTR}}$ , RxRDY, TxRDY, TxEMP, SYNDET. De estas señales, las primeras cinco son suficientes para la realización del presente proyecto.

RxD	Recepción de datos
TxD	Transmisión de datos
$\overline{\text{RTS}}$	Habilitación del modem para transmitir
$\overline{\text{CTS}}$	Habilitación del USART para transmitir
$\overline{\text{DSR}}$	Detección de la portadora

La descripción detallada de estas señales se efectuó en la sección 2.2.2 (definición de funciones EIA) y en la sección 3.2.2 (señales de interfaz del USART).

Las cinco restantes señales de interfaz que suministra el USART no serán usadas y por tanto no irán físicamente conectadas a ninguna parte.

Tal como se indicó al final de la sección 3.2.1, las señales de interfaz que provee el USART no son compatibles con los niveles de voltaje de la interfaz estándar EIA, lo cual implica la adición de impulsores y receptores de línea entre el USART y el modem.

El 8T15 es un impulsor de línea doble para comunicaciones, y suministra los niveles de voltaje adecuados para la transmisión de datos entre el equipo de comunicación de datos y el equipo terminal. El 8T15 requiere de un suministro nominal de +12 y -12 volts.

Según puede verse en la figura 3.19, cada impulsor

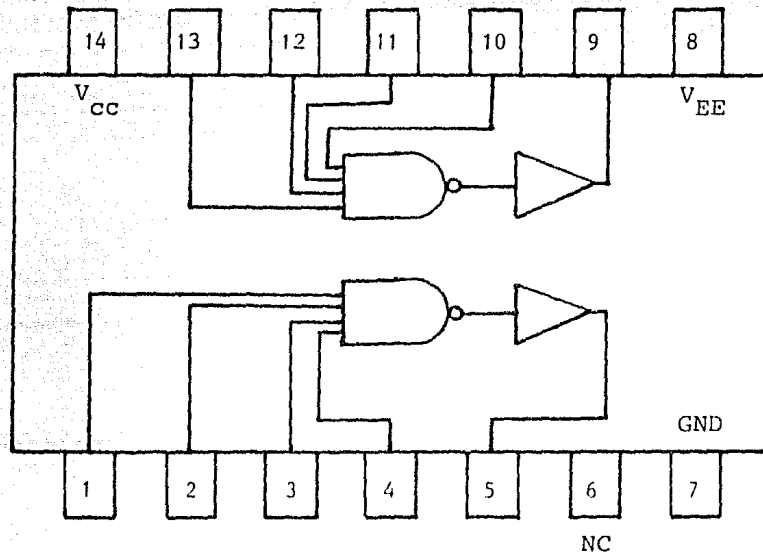


Figura 3.19 Circuito Integrado 8T15. Contiene dos impulsores de línea.

realiza una función lógica NAND a partir de cuatro entradas, las cuales pueden ser niveles lógicos estándar TTL. A la salida de la compuerta NAND está colocado un "buffer" para impulsar líneas de interfaz a niveles nominales de +6 y -6 volts.

El 8T15 estará asociado con las señales TxD y  $\overline{\text{RTS}}$ , cuya dirección es del USART hacia el modem.

El 8T16 es un receptor de línea doble, diseñado para la interfaz entre equipo de comunicación de datos y equipo terminal, con características EIA-RS-232-C o MIL STD-188B. Cada receptor está dotado de dos entradas MIL (MIL+ y MIL-) y de una entrada EIA, siendo posible controlar a cada compuerta independientemente mediante las entradas de histéresis y "strobe" (figura 3.20). El 8T16 opera a partir de un suministro sencillo de +5 volts.

La entrada "strobe" opera de la siguiente manera:

- Un cero lógico en la entrada "strobe" permite la transferencia de datos.

- Un uno lógico en la entrada "strobe" mantiene la salida en nivel alto.

Este comportamiento del dispositivo está dentro de la convención de lógica negativa, un 1 se refiere a un voltaje bajo, y un 0 se refiere a un voltaje alto.

El 8T16 acepta una señal simple (EIA o MIL) o una señal diferencial (MIL), convirtiéndola a nivel estándar TTL.

El circuito emplea la histéresis para obtener reducción de ruido en condiciones adversas, y conectando a

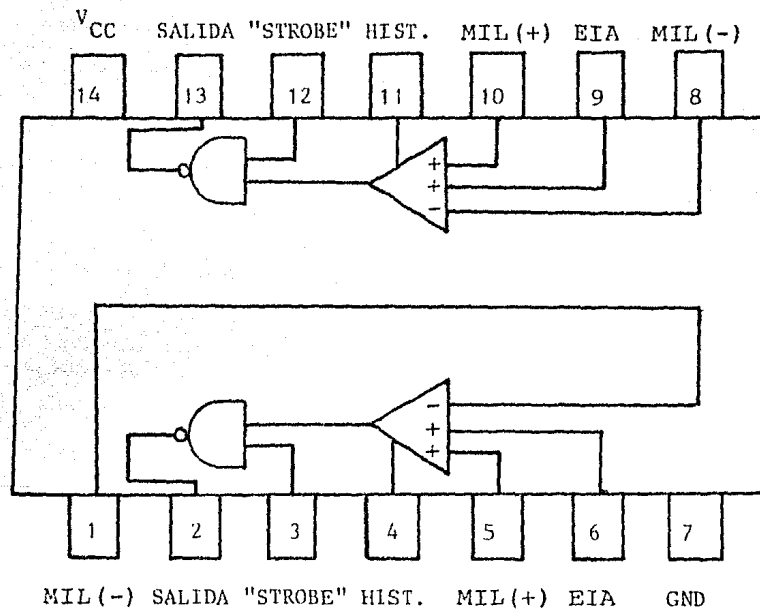


Figura 3.20 Circuito Integrado 8T16. Contiene dos receptores de línea.



tierra la histéresis se logra que el voltaje mínimo esté ligeramente arriba de 0 volts.

La entrada EIA puede dejarse sin conectar cuando se usan una o ambas entradas MIL, pero cuando sólo se usa la entrada EIA, ambas entradas MIL deben ser conectadas a tierra.

Las señales de interfaz asociadas al 8T16 serán Rx $\overline{D}$ ,  $\overline{CTS}$  y  $\overline{DSR}$ , cuya dirección es del modem hacia el USART.

La figura 3.21 muestra la configuración lógica de las señales de interfaz del USART hacia el modem empleando el impulsor de línea 8T15 y el receptor de línea 8T16.

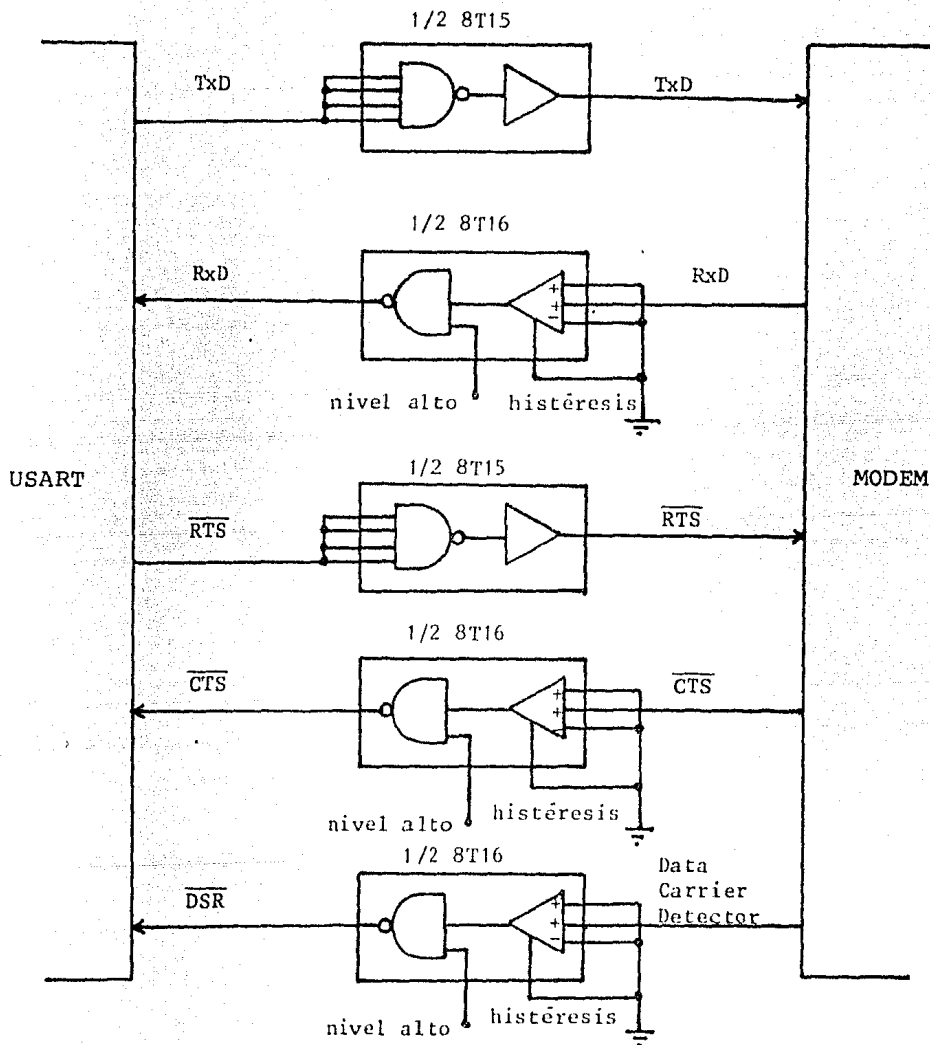


Figura 3.21 Acoplamiento de las señales de interfaz a niveles de voltaje compatibles con las características EIA mediante impulsores y receptores de línea.

Con los elementos descritos en las secciones anteriores (3.3.1, 3.3.2, 3.3.3), se puede construir la tarjeta de interfaz a la cual se hizo referencia en la sección 1.4.1, y que será el puerto de entrada/salida del Sistema de Desarrollo de Microprocesadores que establezca la comunicación con la computadora HP 3000.

La figura 3.22 muestra la distribución aproximada de los componentes en la tarjeta de interfaz. Los componentes numerados 13 y 14 son dos receptáculos para circuitos integrados de ocho patas. Un interruptor permite escoger entre dos velocidades de transmisión, 300 o 2400 bauds, sin tener que modificar la programación del USART.

La figura 3.23 corresponde al diagrama completo de conexiones en la tarjeta de interfaz.

Las líneas de control del USART se conectan a las líneas del sistema como se indica a continuación:

- $\overline{RD}$  se conecta a  $\overline{I/O R}$
- $\overline{WR}$  se conecta a  $\overline{I/O W}$
- $C/\overline{D}$  se conecta a la línea 0 del canal de direcciones.
- RESET se conecta mediante un "latch" a la línea de inicialización del sistema.

Los relojes del USART  $\overline{Rx\overline{C}}$  y  $\overline{TxC}$  se conectan juntos al interruptor, y éste, a las bases de tiempo indicadas en el diagrama. El ajuste adecuado para obtener una velocidad real de 2400 bauds se hará mediante "software". La señal CLK se conecta a la salida de 2.457 MHz.

Los receptáculos para circuito integrado se usan en combinación con los decodificadores 3205 para activar las señales  $\overline{CS}$  del USART y de los "buffers" del canal de datos. Las salidas del 3205 son conectadas al lado izquierdo del receptáculo, mientras que del lado derecho de éste se cortocircuitan todos los contactos entre sí y se conectan a los "chip select" del 8251 y de los 8226.

Mediante un alambre se puentean ambos lados de cada receptáculo como se indica en el diagrama, actuando así como interruptores a la salida de cada decodificador.

Conectados estos alambres tal como se muestra en el diagrama, las señales  $\overline{CS}$  serán seleccionadas cuando por el canal de direcciones aparezca la información siguiente:

1	0	1	1	1	0	1	X
$A_7$	$A_6$	$A_5$	$A_4$	$A_3$	$A_2$	$A_1$	$A_0$

El bit  $A_1$  habilita al primer decodificador en combinación con el código de los bits  $A_2$ ,  $A_3$ ,  $A_4$ , produciendo una salida de nivel bajo en el contacto puentado.

Esta salida habilita al segundo decodificador, el cual en combinación con el código de los bits  $A_5$ ,  $A_6$ ,  $A_7$ , produce en este segundo decodificador una salida baja en el contacto puentado respectivo, el cual está conectado a las señales  $\overline{CS}$  y por tanto habilitará a los dispositivos.

El bit  $A_0$  está conectado a la señal  $C/\overline{D}$ , por tanto, si es 1 el USART asumirá que a continuación aparecerá un comando de control, y si es 0, que aparecerán datos.

Es importante hacer notar que la información en el canal de direcciones se está recibiendo invertida, por lo tanto, el programa debe enviar la información complementaria, que sería:

#### PUERTO DE CONTROL

hexadecimal	4				4																			
binario	<table border="1"> <tr> <td>0</td><td>1</td><td>0</td><td>0</td> <td>0</td><td>1</td><td>0</td><td>0</td> </tr> <tr> <td>A<sub>7</sub></td><td>A<sub>6</sub></td><td>A<sub>5</sub></td><td>A<sub>4</sub></td> <td>A<sub>3</sub></td><td>A<sub>2</sub></td><td>A<sub>1</sub></td><td>A<sub>0</sub></td> </tr> </table>								0	1	0	0	0	1	0	0	A <sub>7</sub>	A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>
0	1	0	0	0	1	0	0																	
A <sub>7</sub>	A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>																	

#### PUERTO DE DATOS

hexadecimal	4				5																			
binario	<table border="1"> <tr> <td>0</td><td>1</td><td>0</td><td>0</td> <td>0</td><td>1</td><td>0</td><td>1</td> </tr> <tr> <td>A<sub>7</sub></td><td>A<sub>6</sub></td><td>A<sub>5</sub></td><td>A<sub>4</sub></td> <td>A<sub>3</sub></td><td>A<sub>2</sub></td><td>A<sub>1</sub></td><td>A<sub>0</sub></td> </tr> </table>								0	1	0	0	0	1	0	1	A <sub>7</sub>	A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>
0	1	0	0	0	1	0	1																	
A <sub>7</sub>	A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>																	

Por último, todas las señales referidas al CPU, así como alimentaciones y tierra de todos los componentes, van alambradas al conector inferior de la tarjeta, mientras que las cinco señales de interfaz se alambran al conector superior, a partir del cual se efectúa otra conexión a un tercer conector que se instala en la parte trasera del gabinete que contiene a las tarjetas de la microcomputadora.

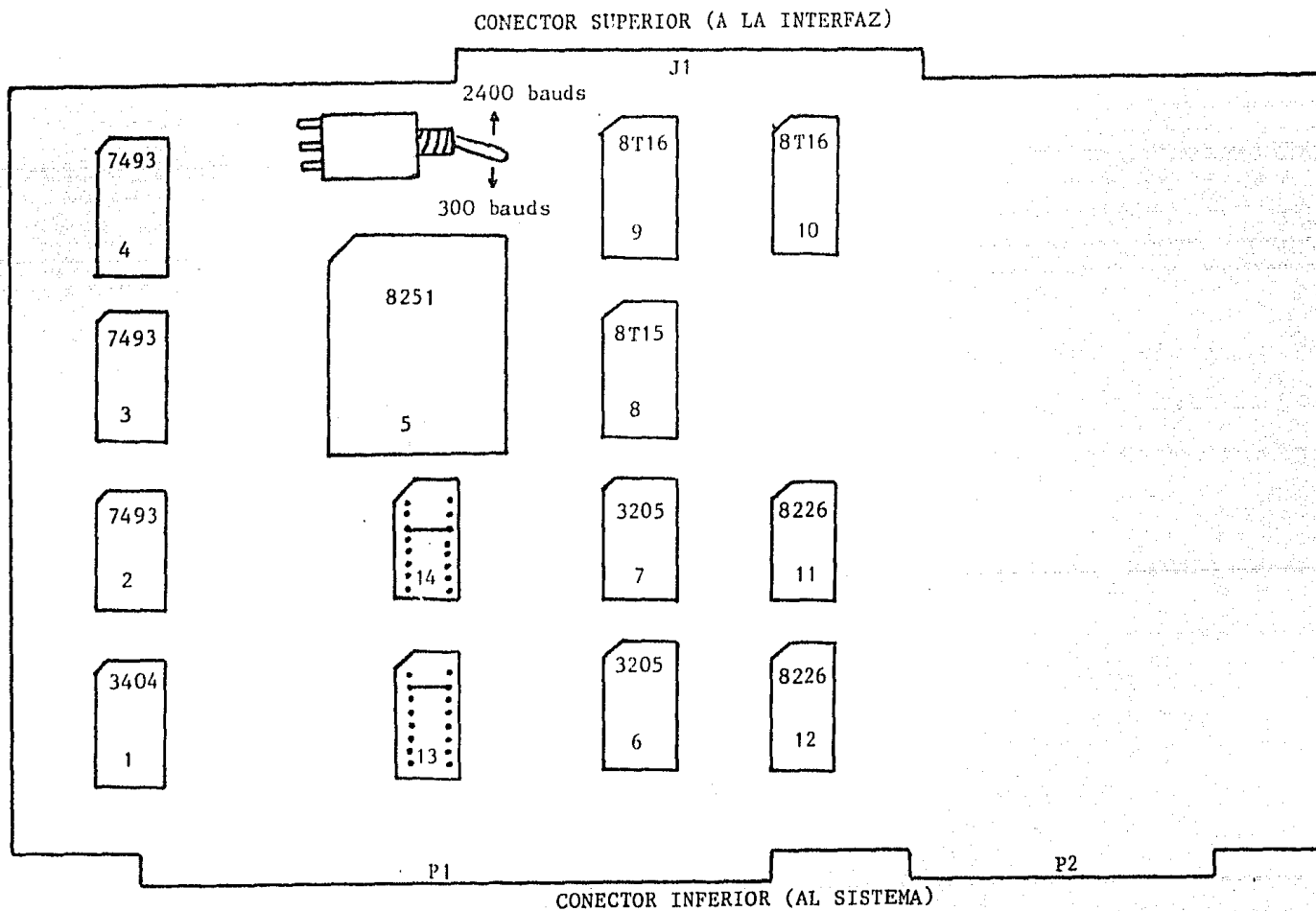


Figura 3.22 Distribución aproximada de componentes en la tarjeta.

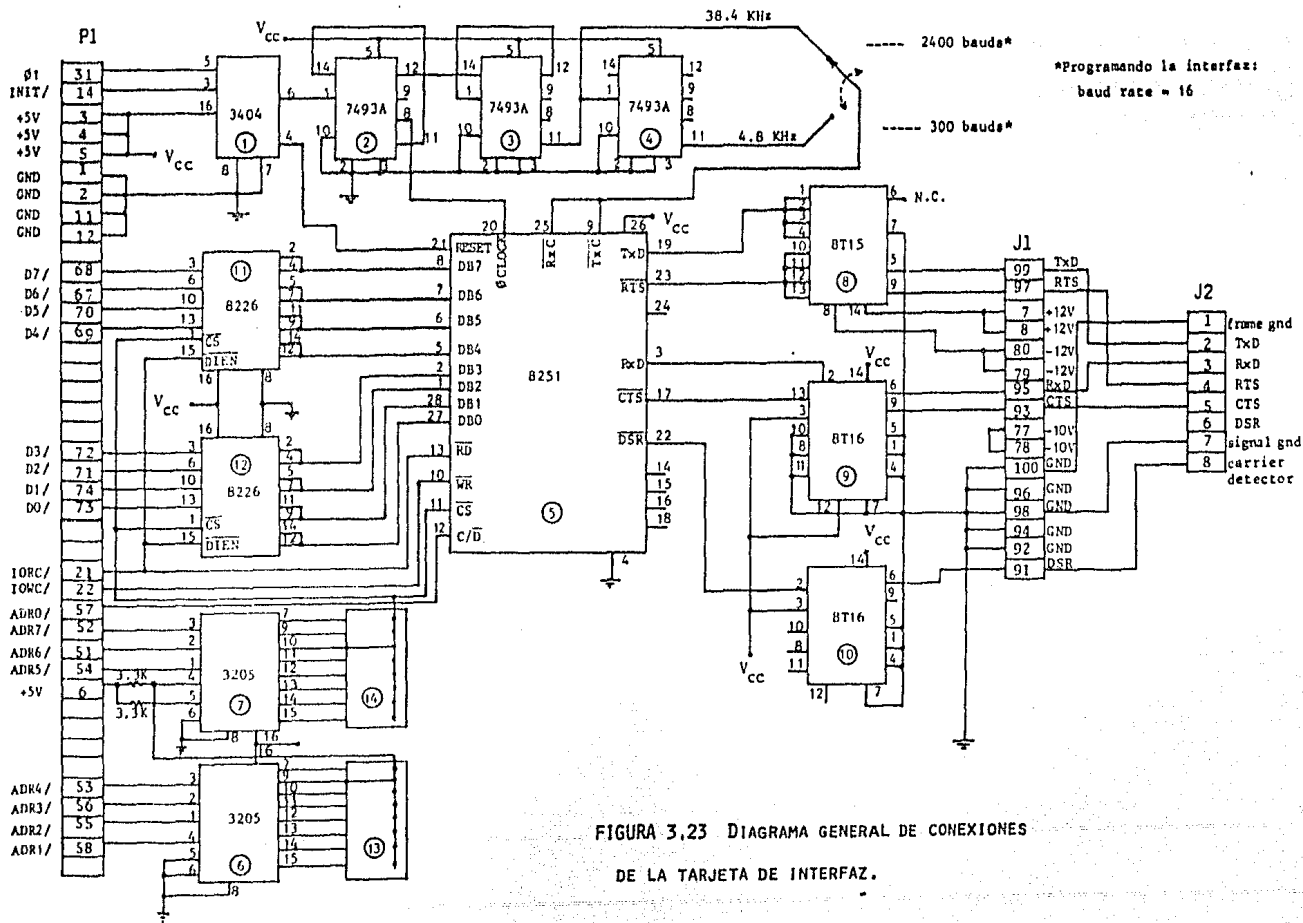


FIGURA 3.23 DIAGRAMA GENERAL DE CONEXIONES DE LA TARJETA DE INTERFAZ.

#### 3.4.1 Instrucciones del microprocesador 8080.

El juego de instrucciones del 8080 incluye cinco diferentes tipos de instrucciones:

- Grupo de transferencia de datos.

Mueven datos entre registros o entre memoria y registros.

- Grupo aritmético.

Suman, restan, incrementan o decreamentan datos almacenados en memoria o en registros.

- Grupo lógico.

Funciones AND, OR, EXCLUSIVE-OR, comparación, rotación o complementación, actúan sobre datos que se encuentran en registros o en memoria.

- Grupo de ramificación.

Contiene instrucciones condicionales e incondicionales de salto, instrucciones de llamada a subrutina e instrucciones de retorno.

- Grupo de apilación, ent/sal y control.

Incluye instrucciones de ent/sal, así como instrucciones para mantener al apilador interno y a las banderas de control internas.

En el apéndice A se definen brevemente las instrucciones del 8080, enunciando el código mnemotécnico, el campo de operandos, el nombre de la instrucción y una descripción de la función que realizan.



De acuerdo a los conceptos expuestos en la sección 3.2.3, la inicialización del USART se llevará a cabo como se describe a continuación.

Las condiciones deseadas para realizar la interfaz entre la computadora y la microcomputadora son:

- Transmisión asíncrona.
- Caracteres de datos de 8 bits.
- Hacer caso omiso de la paridad en el código ASCII.
- Añadir dos "stop" bit al final de cada carácter.
- Disponer de dos velocidades de transmisión, una a 300 bauds y la otra a 2400 bauds.

El primer carácter que debe llegar al USART después del "reset" de todo el sistema es el Registro de Modo de Control, seguido por el Registro de Instrucción de Comando.

#### Registro de modo de control

En la figura 3.23 se puede ver como las líneas de  $\overline{\text{RxC}}$  y  $\overline{\text{TxC}}$ , que son las señales de control de velocidad para recepción y transmisión de datos respectivamente, se pueden conectar mediante el interruptor a cualquiera de dos frecuencias, 38.4 KHz o 4.8 KHz (para obtener respectivamente 38400 bauds y 4800 bauds). Si se deseara obtener otra frecuencia se puede conectar el interruptor a la salida elegida de los flip-flops.

A partir de las frecuencias de 38.4 KHz y 4.8 KHz, se ve que dividiéndolas entre 16 se obtienen 2.4 KHz y 300 Hz, que corresponden a las velocidades pre-establecidas de 2400 bauds y 300 bauds.

Por esto, para el factor de cambio en bauds, correspondiente a los bits  $D_1 - D_0$  del registro de modo de control, se especificará el código binario 10, transmisión asíncrona x 16.

Para una longitud de carácter de 8 bits, se requiere el código binario 11 en los bits  $D_3 - D_2$  del registro de modo de control.

Los bits  $D_5 - D_4$ , con código X0, determinan "sin paridad", sea entonces el código 00.

Para añadir dos "stop" bit se requiere el código 11 en los bits  $D_7 - D_6$ .

Con estos datos, el registro de modo de control queda constituido de la siguiente manera:

hexadecimal	C				E											
binario	<table border="1"> <tr> <td>1</td><td>1</td><td>0</td><td>0</td> <td>1</td><td>1</td><td>1</td><td>0</td> </tr> </table>								1	1	0	0	1	1	1	0
1	1	0	0	1	1	1	0									
	$D_7$	$D_6$	$D_5$	$D_4$	$D_3$	$D_2$	$D_1$	$D_0$								

La información del registro de modo de control será, en código hexadecimal, CE, y en la subrutina de inicialización del USART será designada mediante la variable MIFO.

#### Registro de instrucción de comando

Una vez inicializado el USART mediante el registro de modo de control, se procede a habilitarlo para su funcionamiento con el registro de instrucción de comando.

El bit  $D_0$  indicará que se habilita la transmisión, haciendo TxEN = 1.

Como la señal  $\overline{DTR}$  no será empleada en este proyecto, quedará deshabilitada haciendo  $DTR = 0$  en el bit  $D_1$ .

Mediante el bit  $D_2$  se habilitará la recepción, con  $RxE = 1$ .

No se empleará el caracter "break", quedando el bit  $D_3$  en 0.

Haciendo  $ER = 1$  en el bit  $D_4$ , se obtendrá un "reset" en las banderas de error del registro de estado (PE, OE y FE).

Con  $RTS = 1$  en el bit  $D_5$ , se obtiene el nivel bajo requerido para esta señal.

En el bit  $D_6$  no se requiere un "reset" interno, por tanto  $IR = 0$ .

En el bit  $D_7$  se hará  $EH = 0$ , pues no se usará transmisión síncrona.

El registro de instrucción de comando quedará constituido de la siguiente manera:

hexadecimal	3				5																			
binario	<table border="1" style="border-collapse: collapse; text-align: center; width: 100%;"> <tr> <td style="width: 12.5%;">0</td> <td style="width: 12.5%;">0</td> <td style="width: 12.5%;">1</td> <td style="width: 12.5%;">1</td> <td style="width: 12.5%;">0</td> <td style="width: 12.5%;">1</td> <td style="width: 12.5%;">0</td> <td style="width: 12.5%;">1</td> </tr> <tr> <td><math>D_7</math></td> <td><math>D_6</math></td> <td><math>D_5</math></td> <td><math>D_4</math></td> <td><math>D_3</math></td> <td><math>D_2</math></td> <td><math>D_1</math></td> <td><math>D_0</math></td> </tr> </table>								0	0	1	1	0	1	0	1	$D_7$	$D_6$	$D_5$	$D_4$	$D_3$	$D_2$	$D_1$	$D_0$
0	0	1	1	0	1	0	1																	
$D_7$	$D_6$	$D_5$	$D_4$	$D_3$	$D_2$	$D_1$	$D_0$																	

La información del registro de instrucción de comando será 35 en código hexadecimal, y en la subrutina de inicialización del USART será designada mediante la variable CIFO.

El programa del usuario deberá cubrir necesariamente tres aspectos que involucran el control del nuevo módulo de interfaz del SDM.

En primer lugar, se requiere una subrutina para inicializar al USART, la cual debe suministrar la información del registro de modo de control y la información del registro de instrucción de comando.

Deben haber además dos subrutinas que deben enviar información del sistema al exterior, y aceptar información externa dentro del sistema, respectivamente, a través de la interfaz.

En la sección 3.3.4 se definió al puerto de control con el código hexadecimal 44, y al puerto de datos con el código hexadecimal 45.

El puerto de control tendrá dos funciones, suministrar al USART el modo de control y la instrucción de comando por una parte, y por otra, leer del USART el registro de estado cada vez que se necesite.

El puerto de datos también tendrá dos funciones, enviar datos del canal de datos al USART, y colocar datos provenientes del exterior que llegan al USART en el canal interno de datos del sistema.

Los códigos hexadecimales de los puertos no variarán, pero se designarán de diferente manera en el programa del usuario, de acuerdo a la función que estén desempeñando en un momento dado.

Las designaciones mnemotécnicas serán:

código operacional (hexadecimal)	código mnemotécnico	f u n c i ó n
44	TELCTL	Puerto de control del USART
44	TELS	Puerto de lectura del registro de estado del USART
45	TELO	Puerto de salida de datos del USART
45	TELI	Puerto de entrada de datos del USART
CE	MIFO	Información del registro de modo de control
35	CIFO	Información del registro de instrucción de comando

A partir de estos datos se pueden programar las tres subrutinas de control de la interfaz.

La figura 3.24 muestra el diagrama de flujo de la subrutina de inicialización del USART. El programa correspondiente aparece a continuación:

```

MVI A, MIFO
OUT TELCTL
MVI A, CIFO
OUT TELCTL

```

La figura 3.25 muestra el diagrama de flujo de la subrutina para transmitir datos del canal interno de datos del sistema a la interfaz, siendo el programa correspondiente:

```
TELOUT:   IN  TELS
          ANI TRDY
          JZ  TELOUT
          MOV A,C
          OUT TELO
          RET
```

La figura 3.26 corresponde al diagrama de flujo de la subrutina para colocar datos que llegan a la interfaz del exterior en el canal interno bidireccional de datos del sistema. El programa asociado es el siguiente:

```
TELIN:   IN  TELS
          ANI RBR
          JZ  TELIN
          IN  TELI
          RET
```

Los listados con la codificación correspondiente a estas tres subrutinas aparecen en el apéndice D.

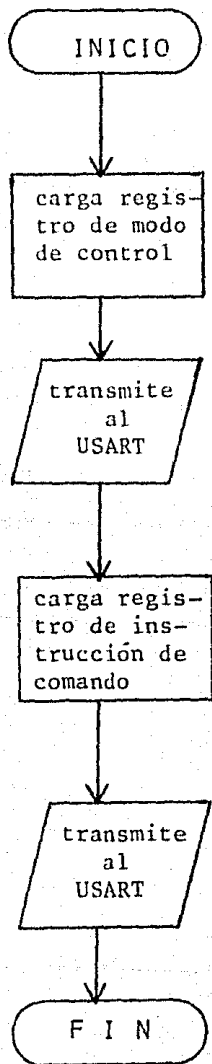


Figura 3.24 Diagrama de flujo de la subrutina de inicialización del USART.

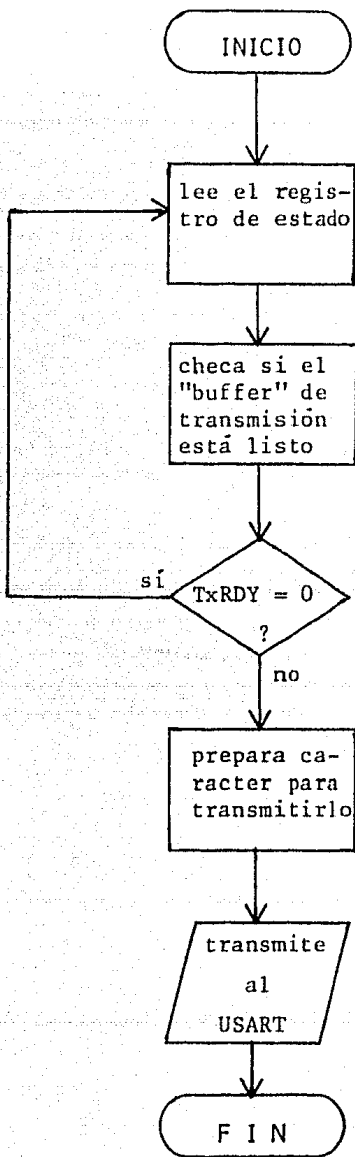


Figura 3.25 Diagrama de flujo de la subrutina que transmite datos del sistema al exterior.



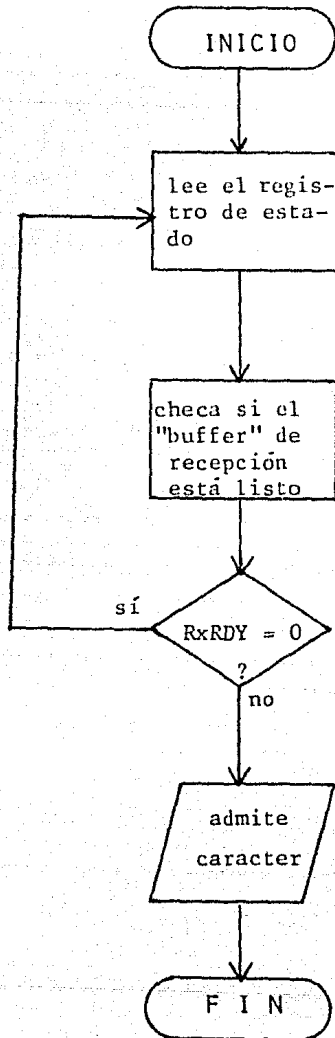
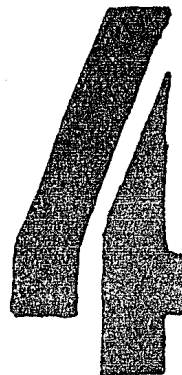


Figura 3.26 Diagrama de flujo de la subrutina que toma del USART los datos que llegan a este último provenientes del exterior.

\* \* \* \* \*



E N L A C E D E  
L O S S I S T E M A S

## 4.1.1 Conexiones en la HP 3000.

Según se expuso en la sección 1.4.1, las adiciones de "hardware" a la HP 3000 son en extremo sencillas.

El controlador terminal asíncrono de la HP 3000 suministra los siguientes circuitos EIA.

<u>contacto</u>	<u>circuito</u>	<u>definición</u>
2	BB	RECEIVE DATA
3	BA	SEND DATA
4	CF	CARRIER DETECT
6	CD	DATA TERMINAL READY
7	AB	COMMON RETURN
8	CA	REQUEST TO SEND
11	SCF	SECONDARY RECEIVE LINE SIGNAL DETECTOR
12	SCA	SECONDARY REQUEST TO SEND
20	CC	DATA SET READY
22	CB	CLEAR TO SEND
23	CH	FREQUENCY SELECT

De entre estos circuitos, sólo se usarán los indicados en la figura 4.1, donde se muestra como alambrar el cable de interfaz a los conectores del modem y de la computadora.

La figura 4.2 muestra la conexión final de las líneas de interfaz para la HP 3000. El conector designado para el proyecto es el J13 de la computadora HP 3000.

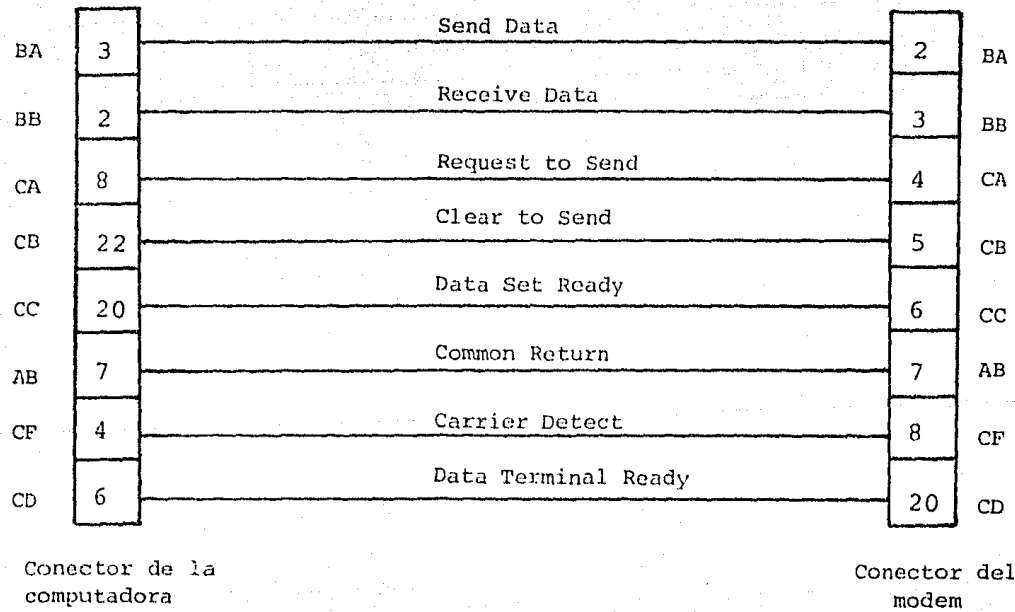


Figura 4.1 Cable de interfaz para el modem de la HP 3000.

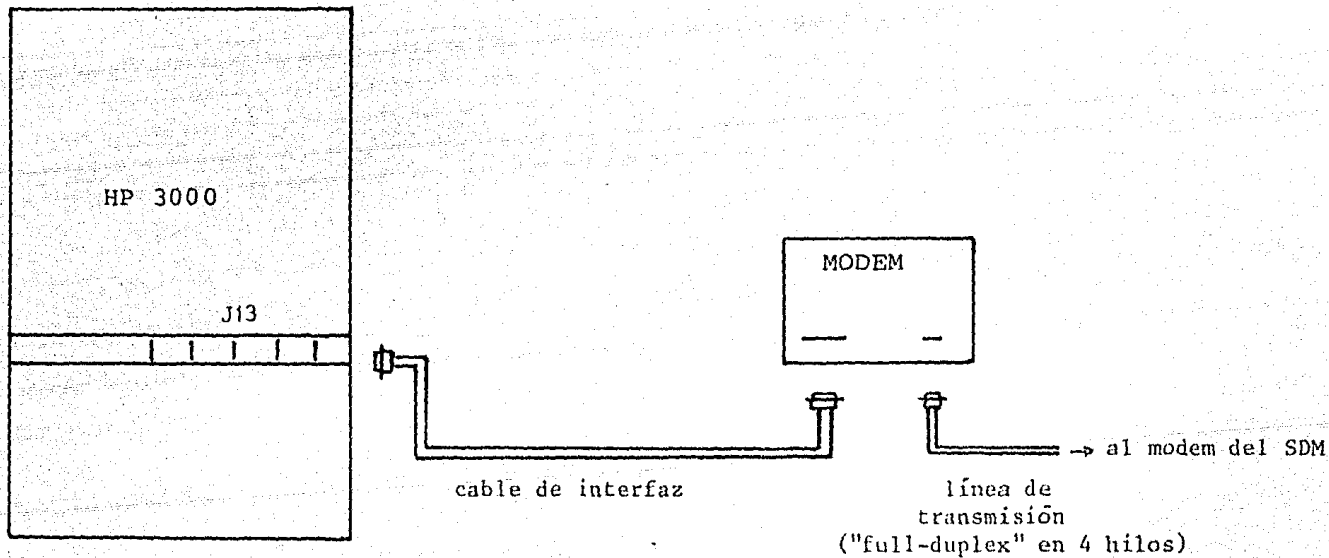


Figura 4.2 Conexiones de interfaz del lado de la computadora HP 3000.

La tarjeta de interfaz presentada en la sección 3.3.4 se conecta al sistema de desarrollo de microprocesadores según se expuso en la sección 1.4.1.

Del conector superior de la tarjeta de interfaz, ya sea directamente a ésta o mediante un conector, se llevan las señales de interfaz a otro conector que pueda instalarse en la parte trasera del gabinete.

Entre este último conector y el modem, se conecta el cable de interfaz correspondiente a la microcomputadora, el cual se alambra tal como se muestra en la figura 4.3.

En el momento de introducir la tarjeta de interfaz en el gabinete, todas las líneas referidas al CPU quedan automáticamente conectadas al sistema.

El interruptor montado en la tarjeta se colocará para habilitar la transmisión a 2400 bauds.

La figura 4.4 muestra el aspecto general de las conexiones finales del lado de la microcomputadora.

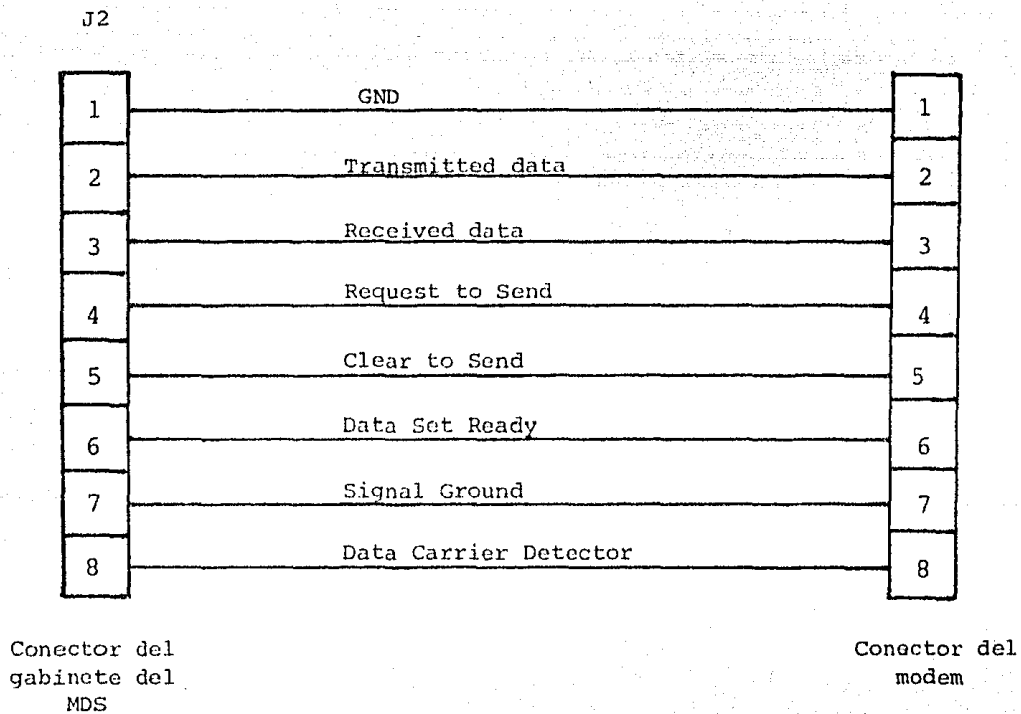


Figura 4.3 Cable de interfaz de la microcomputadora.

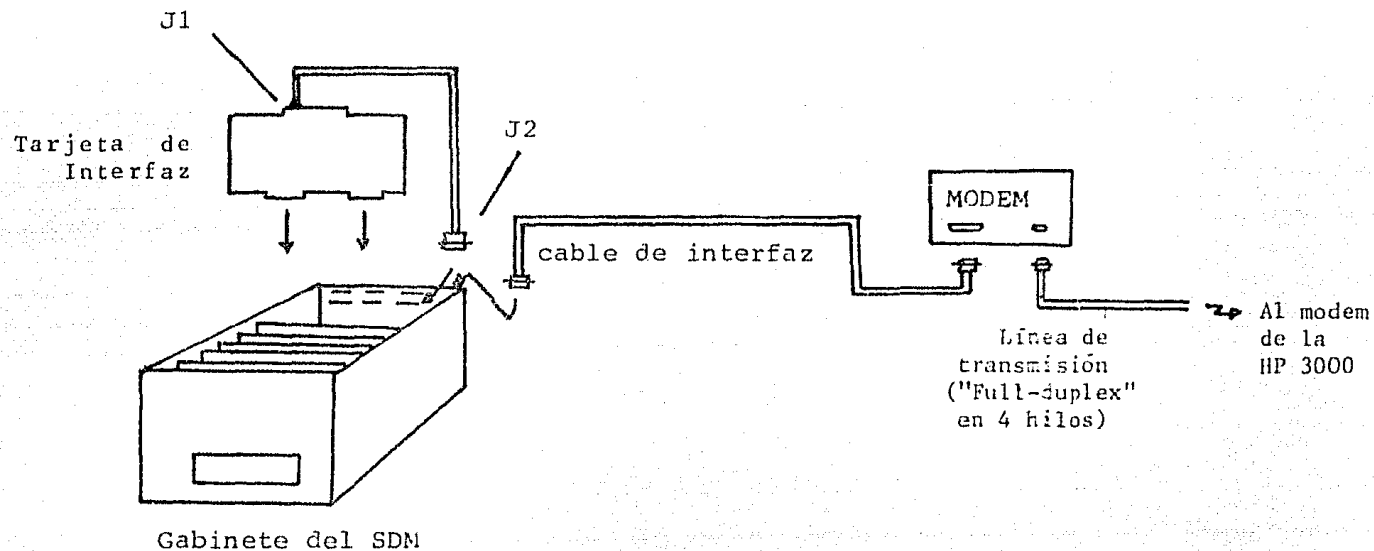


Figura 4.4 Conexiones de interfaz del lado de la microcomputadora.



## 4.2.1 Configuración del puerto de la HP 3000.

Tal como se indicó en la sección 1.4.2, corresponde al operador de la HP 3000 la tarea de configurar el puerto de la computadora.

Para este proyecto, el dispositivo lógico asignado es el número 33, correspondiente al conector J13. Con este dato, y sabiendo que la HP 3000 considerará a la interfaz de la microcomputadora como una terminal asíncrona propia, el operador puede proceder a la configuración del puerto.

En el apéndice F se muestra la información al respecto, y se puede apreciar como queda configurado el dispositivo lógico 33 con respecto a los demás dispositivos.

El programa de control de la interfaz será el encargado de establecer la comunicación a través de la tarjeta de interfaz entre la microcomputadora y la computadora de propósito general.

Los aspectos básicos que debe contemplar un programa de tal naturaleza son los siguientes:

- Inicializar el USART.
- Poseer las subrutinas de entrada y salida del puerto (sección 3.4.3).
- Enviar a la HP la señal adecuada para el "log-on".
- Enviar a la HP la señal adecuada para el "log-off".
- Admitir la posibilidad de que el usuario del SDM pueda borrar caracteres o líneas antes de transmitir su información a la HP.
- Producir los ecos de los caracteres del usuario y de la información proveniente de la HP.
- Terminada la ejecución del programa, restituir el control al sistema operativo del SDM.

En las páginas siguientes se muestra el diagrama de flujo del programa (figura 4.5).

En el apéndice E se puede ver el listado de este programa con la codificación correspondiente.

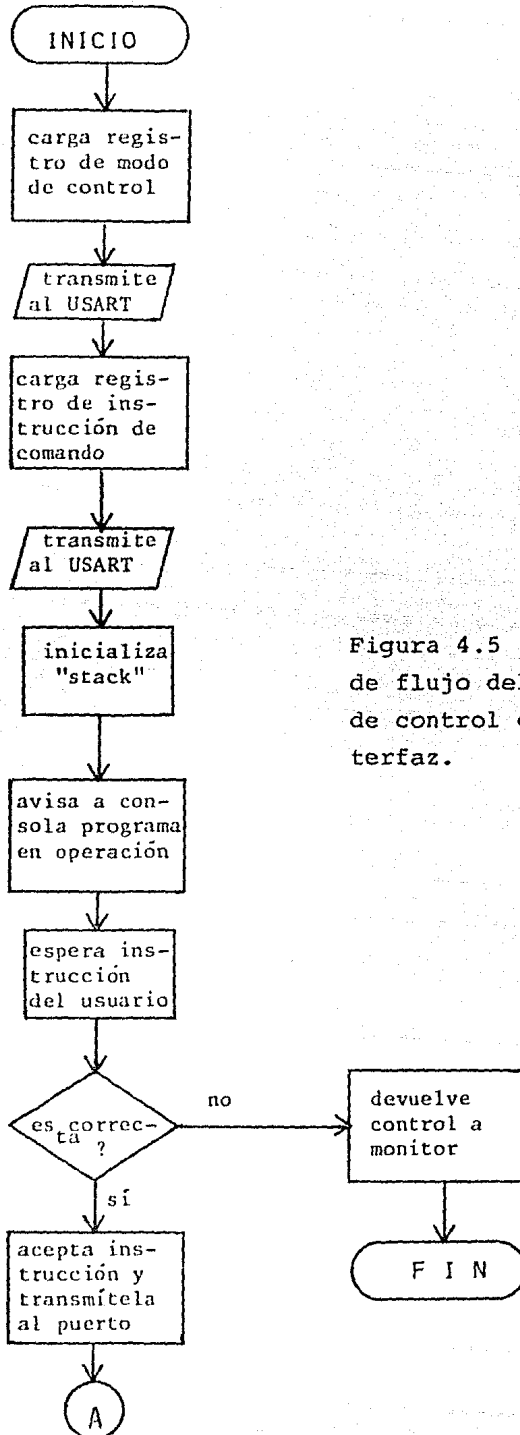


Figura 4.5 Diagrama de flujo del programa de control de la interfaz.

Figura 4.5  
(continuación)

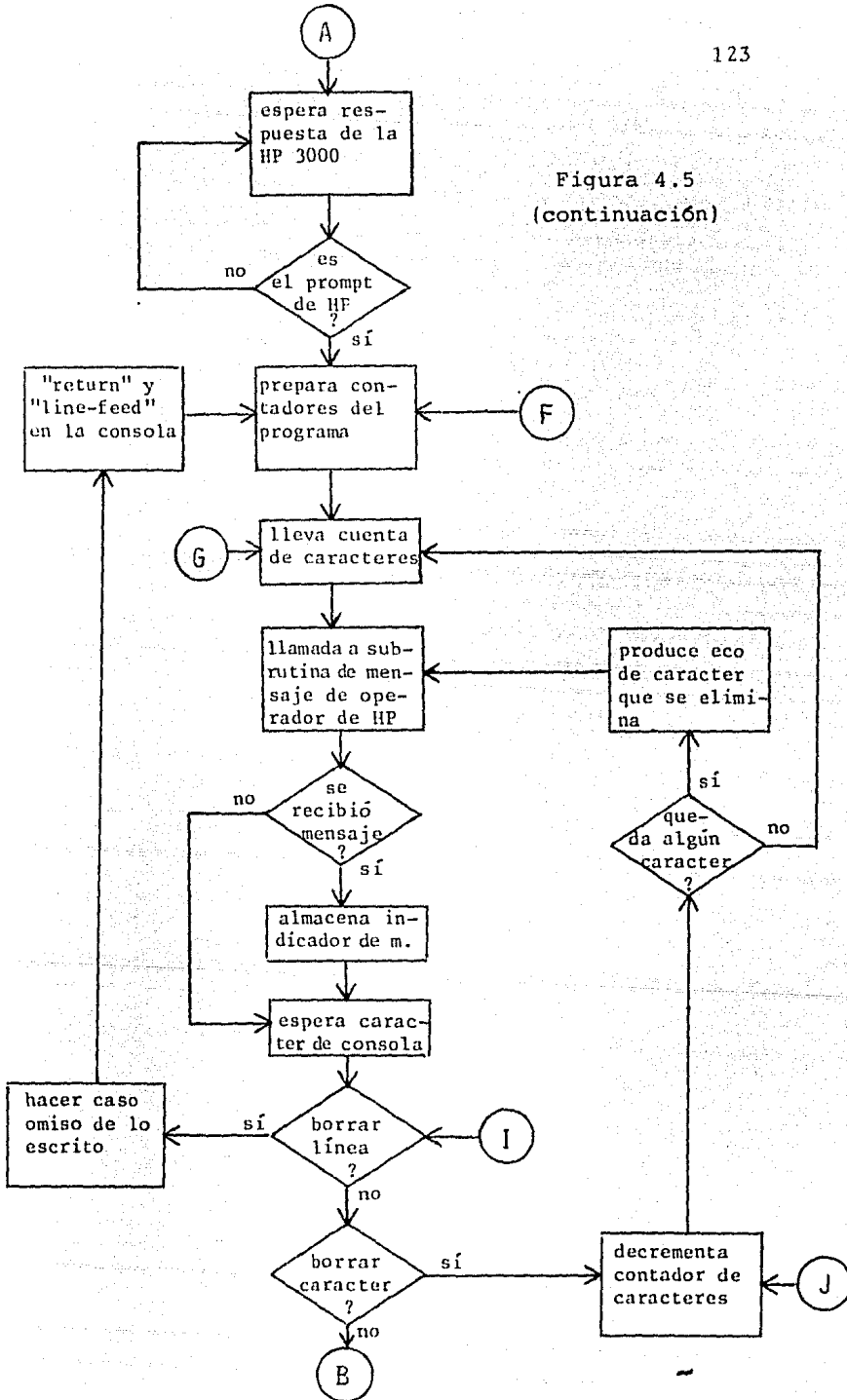
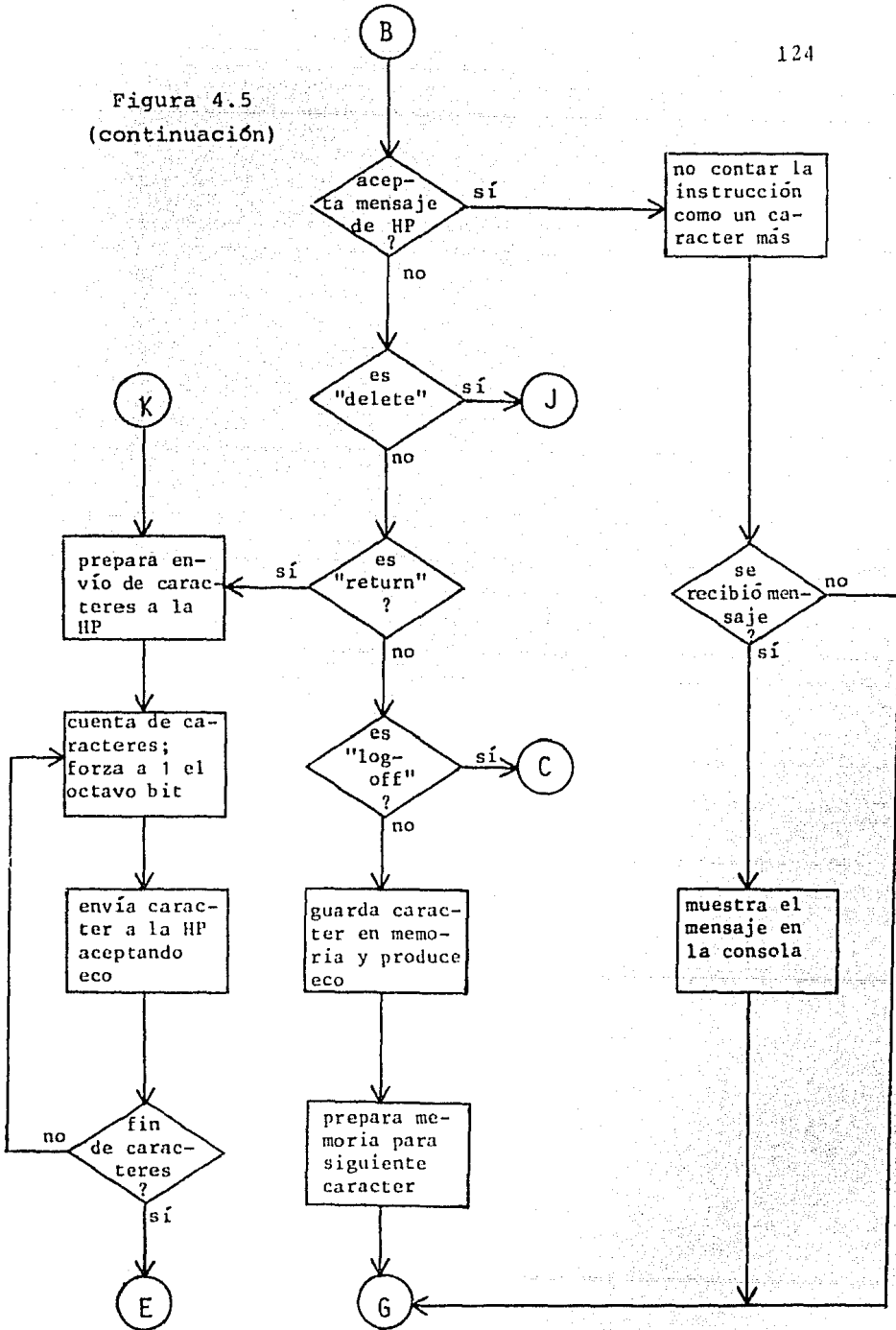


Figura 4.5  
(continuación)



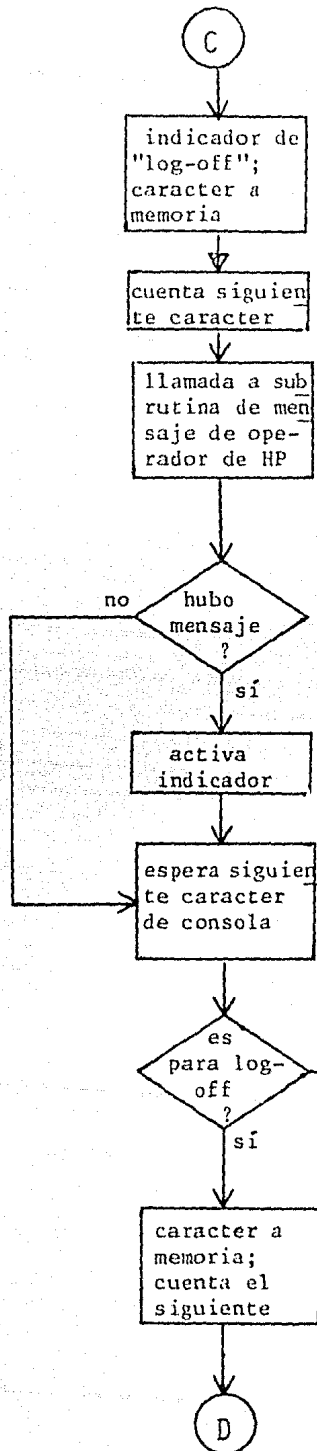


Figura 4.5  
(continuación)

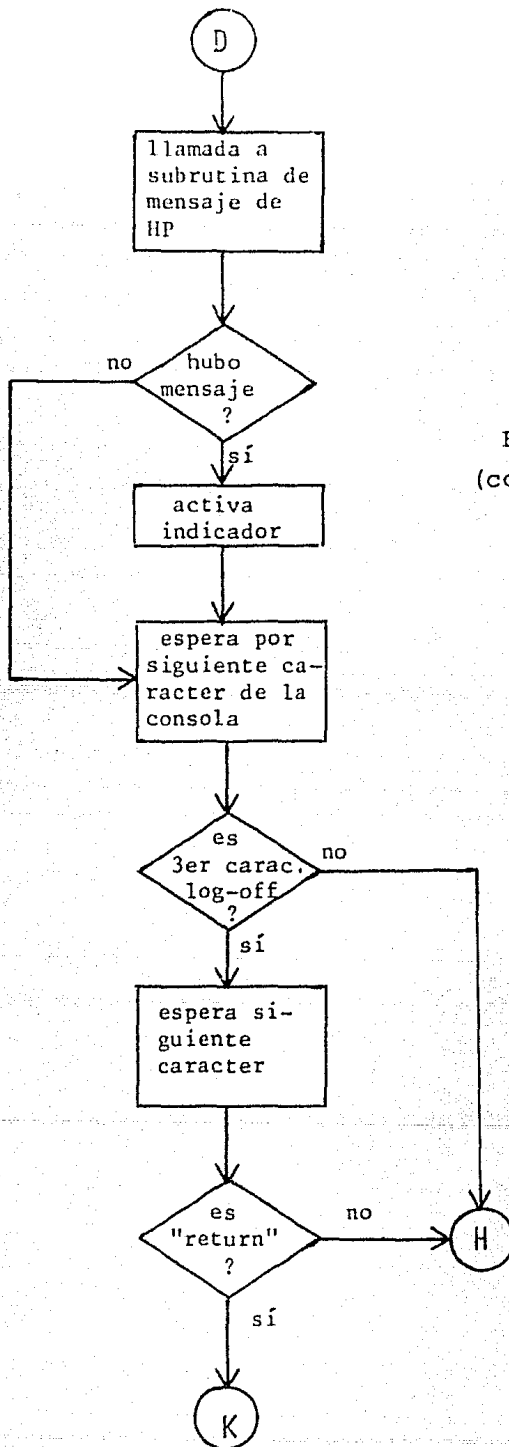


Figura 4.5  
(continuación)

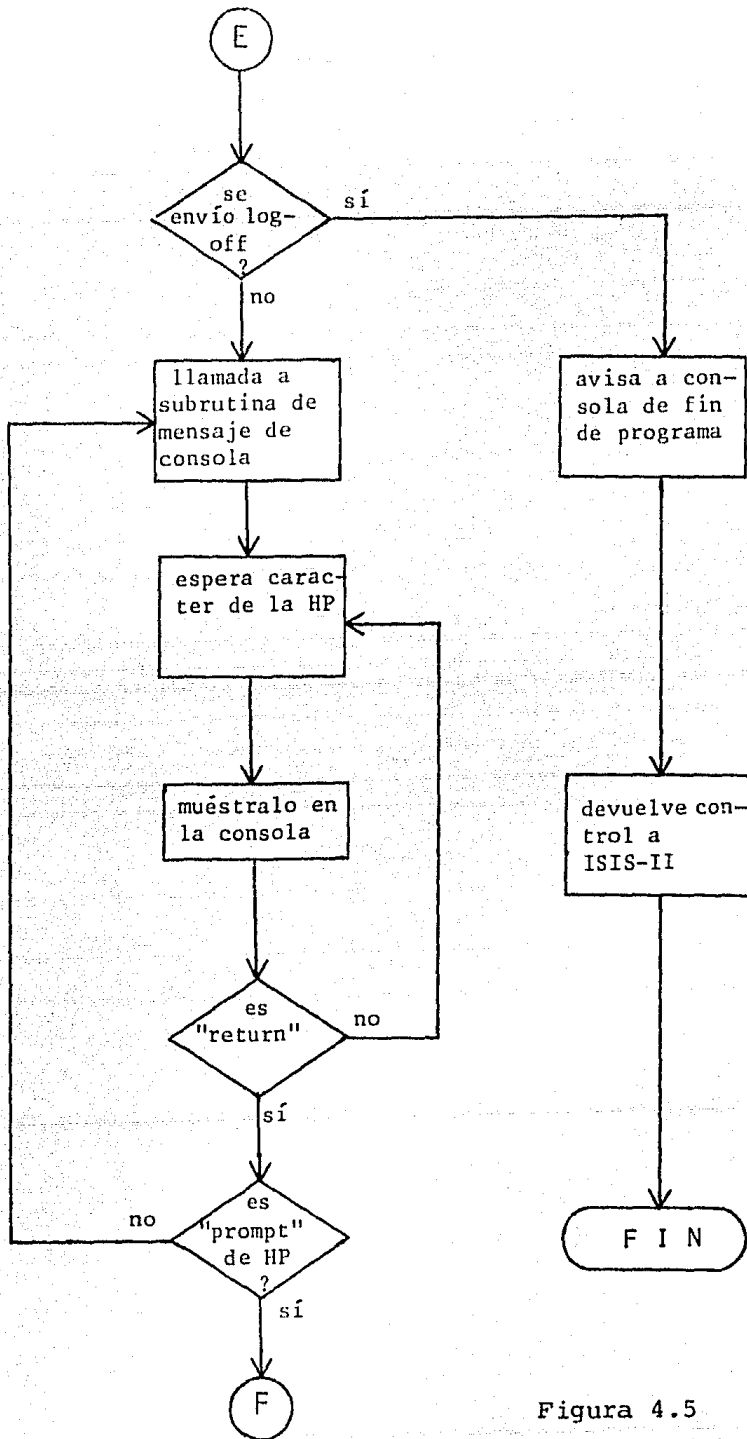


Figura 4.5  
(continuación)



## 4.3.1 El sistema operativo del Intellec-MDS.

El Supervisor de Sistemas de Implementación de Intel (ISIS-II) trabaja en combinación con el sistema de disco blando ("diskette") del SDM, suministrando interfaz de "software" entre el "diskette" y cualquier otro dispositivo periférico estándar. La comunicación con ISIS-II puede efectuarse mediante comandos en el sistema de consola, o mediante llamadas de programas escritos en ensamblador del 8080.

La operación del sistema de discos requiere de la configuración de "hardware" siguiente:

- El sistema Intellec, incluido el monitor.
- 32K a 64K bytes de memoria RAM.
- Sistema de consola (CRT o teletipo).
- Cualquier dispositivo soportado por el monitor.

Los componentes de "hardware" del sistema de discos son los siguientes:

Una o dos unidades de "diskette" con capacidad para uno o dos discos, y hasta un máximo de cuatro discos.

Dos módulos de circuito impreso para cada unidad de "diskette" consistentes en un módulo de interfaz y un módulo de canal, instalados en el gabinete del Intellec.

Un "diskette" conteniendo el sistema de "software" de ISIS-II, el editor de textos, el ensamblador 8080, y los sistemas ICE-80 y UPM.

Dos cables de conexión para cada unidad de discos.

En lo referente a los requerimientos de "software", ISIS-II exige un sistema Intellec con un mínimo de 32K de memoria RAM. Los primeros 12K de memoria están reservados a la parte residente de ISIS-II, mientras que, por arriba de los 12K, el uso de la memoria es alternativo entre partes no residentes de programas de ISIS-II y programas del usuario.

El monitor queda subordinado a ISIS-II cuando el sistema Intellec está corriendo con un sistema de discos. Por tanto, el proceso de inicialización del sistema le permitirá correr bajo control de ISIS-II siempre y cuando el "diskette" del sistema se encuentre en el impulsor 0 de la unidad de discos, y el sistema operativo esté correctamente habilitado.

La aplicación principal del sistema operativo ISIS-II se encuentra en la creación y manejo de archivos.

El sistema ISIS-II ofrece los siguientes servicios, los cuales pueden ser llamados por el programa del usuario:

- Operaciones de ent/sal para el "diskette" y para los periféricos estándar del Intellec, con excepción del Programador PROM Universal. Los servicios disponibles son OPEN, CLOSE, READ, WRITE, SEEK, RESCAN.

- Mantenimiento del directorio del "diskette" (ATTRIB, DELETE, RENAME).

- Asignación del dispositivo de consola y salida de mensaje de error (CONSOL, WHOCON, ERROR).

- Carga y ejecución de programa, así como regreso al supervisor (servicios LOAD y EXIT).

La interfaz con estos servicios de ISIS-II es un llamado al supervisor de ISIS-II, en el cual se especifica el servicio deseado y la dirección de la lista de parámetros que el supervisor va a acceder. Llamar un servicio de ISIS-II no afecta el apilador del programa del usuario, pues el sistema operativo emplea su propio apilador, pero sí borra el contenido de los registros del CPU.

Existen dos variables asociadas a los archivos, debidas a las llamadas a ciertos servicios de ISIS-II, éstas son Longitud y Marca, donde longitud corresponde al número de bytes en un archivo, y marca es el número de bytes leídos o escritos en ese archivo en un instante determinado. Esto significa que el parámetro marca actúa como un apuntador de archivo.

A continuación se exponen algunos de los servicios disponibles en términos de su operación, indicando los parámetros que debe especificar el programa del usuario para poder llamarlos.

Los parámetros tienen nombres simbólicos, que no necesariamente deben ser los que emplee el usuario en su programa.

Inicializa archivo para operaciones de ent/sal.

La llamada de sistema OPEN inicializa tablas del sistema y prepara impulsores que se requieren para el procesamiento de ent/sal de un archivo especificado.

La lista de parámetros que el programa del usuario debe especificar para un OPEN consiste en 5 campos de 2 bytes cada uno, y son los siguientes:

AFTNPTR - El sistema mantiene una tabla de archivos activos, donde a cada archivo abierto se le asigna un número llamado AFTN, el cual es regresado al programa del usuario en el lugar indicado por el campo AFTNPTR, usándose para identificar al archivo abierto en llamadas subsiguientes al sistema. Los valores 0 y 1 para AFTN corresponden a los archivos :CO: y :CI: respectivamente ("console output" y "console input"), los cuales siempre están abiertos.

FILEPTR - Este parámetro contiene la dirección de la cuerda ASCII que especifica el nombre del archivo que se va a abrir. La cuerda ASCII puede comenzar con espacios en blanco, pero no debe haberlos entre caracteres, debiendo terminar con un caracter diferente de una letra, dígito, dos puntos (:) o punto (.).

ACCESS - El valor de este parámetro indica como se va a acceder el archivo una vez abierto. Si ACCESS = 1, se abre para entrada (READ); si ACCESS = 2, se abre para salida (WRITE); y si ACCESS = 3 se pueden efectuar ambas operaciones (READ/WRITE).

Entrada.- Cuando un archivo se abre para entrada, marca se pone a 0 y longitud no se altera. Si el archivo especificado no existe, ocurre un error no fatal.

Salida.- En este caso, marca y longitud se ponen a 0. Si el archivo de salida especificado no existe, se crea uno con el nombre y atributos especificados.

Entrada/salida.- Marca se pone a 0. Si el archivo existe, longitud no se altera, si no existe, se crea un nuevo archivo con el nombre y atributos especificados.

ECHOAFTN - Contiene el AFTN del archivo eco en caso que se se desee editar alguna línea. El archivo de eco debe ser un archivo que se encuentre abierto en modo de salida. Si el parámetro ECHOAFTN contiene 0 no es posible la edición.

STATUS - Contiene la dirección de la localidad de memoria para el retorno de errores no fatales.

### READ

Transfiere datos de un archivo a memoria.

La llamada READ transfiere datos de un archivo abierto a una localidad de memoria especificada por el programa que efectúa la llamada.

Los parámetros que deben ser especificados para un READ contienen 5 campos de 2 bytes:

AFTN - Contiene el AFTN del archivo abierto para lectura o escritura. Es el primer parámetro requerido para READ y vale 1 para :CI:.

**BUFFER** - Contiene la dirección de la localidad de memoria adonde son transferidos los datos, y debe ser mayor o igual que **COUNT**.

**COUNT** - Es el número de bytes que serán transferidos del archivo a la localidad de memoria dada por **BUFFER**.

**ACTUAL** - Es la dirección de una localidad de memoria para el regreso del número de bytes exitosamente transmitidos, y lleva la misma cuenta que el parámetro marca.

**STATUS** - Contiene la dirección de memoria para el regreso del número de error no fatal.

### WRITE

Transfiere datos de memoria a un archivo.

La llamada **WRITE** transfiere datos de una localidad de memoria especificada a un archivo abierto.

Se requieren 4 parámetros de 2 bytes:

**AFTN** - Contiene el **AFTN** de un archivo abierto para lectura o ent/sal, es el primer parámetro requerido por el **WRITE** y vale 0 para **:CO:**.

**BUFFER** - Contiene la dirección de la localidad de memoria de la cual se transfieren los datos.

**COUNT** - Especifica el número de bytes que se van a transferir entre la memoria y el archivo abierto. El valor de **COUNT** es sumado a **marca**.

**STATUS** - Contiene la dirección de memoria para el regreso del número de error no fatal.

Termina operaciones de entrada/salida.

La llamada CLOSE remueve un archivo de las tablas de ent/sal del sistema y libera los impulsores (BUFFERS) que había seleccionado la llamada OPEN. Todos los archivos deben cerrarse cuando se ha completado el proceso de ent/sal.

Los parámetros que debe contener el programa del usuario son:

AFTN - Contiene el AFTN del archivo abierto, y si especifica :CO: o :CI:, los cuales están siempre abiertos, no se realiza ninguna acción, respondiendo con STATUS = 0 al programa que efectúa la llamada.

STATUS - Contiene la dirección de la localidad de memoria para el retorno de errores no fatales.

### EXIT

Termina programa y regresa a ISIS-II.

La llamada EXIT puede ser usada por el programa del usuario para terminar la ejecución de éste y regresar al supervisor. Todos los archivos abiertos se cierran, con excepción de :CO: y :CI:. La asignación de consola no se cambia.

El único parámetro requerido por EXIT es STATUS, que contiene la dirección de memoria para el regreso de errores no fatales.

La interfaz entre el lenguaje ensamblador 8080 y el sistema operativo ISIS-II se efectúa llamando un punto de entrada (llamado ISIS) y dando sus parámetros.

El primer parámetro es un número que identifica la llamada del sistema, y el segundo es la dirección de un bloque de control que contiene los parámetros adicionales requeridos por la llamada del sistema, parámetros que fueron descritos en la sección 4.3.1.

El primer parámetro de la interfaz es pasado al registro C, mientras que la dirección del bloque de control es colocada en el registro par DE.

La forma general de la interfaz es:

```
MVI C, llamada-deseada
LXI D, parámetro-bloque-dirección
CALL ISIS
```

Los números para identificar a las diferentes llamadas del sistema pueden ser definidos mediante declaraciones EQUATE antes de ser usados en el programa, lo que permite referirse a estas llamadas simbólicamente.

Sólo es necesario definir las llamadas que va a emplear el programa del usuario, pudiendo omitir la definición de las que no serán empleadas.

Los detalles para las llamadas OPEN, WRITE, READ, CLOSE, EXIT, con sus respectivos parámetros, se muestran a continuación en código de lenguaje ensamblador fuente.



## ASSEMBLY LANGUAGE CODE FOR ISIS-II SYSTEM CALLS

```

;
;*****
;
; SYSTEM CALL IDENTIFIERS
;
OPEN      EQU      0
CLOSE     EQU      1
DELETE    EQU      2
READ      EQU      3
WRITE     EQU      4
SEEK      EQU      5
LOAD      EQU      6
RENAME    EQU      7
CONSOL    EQU      8
EXIT      EQU      9
ATTRIB    EQU     10
RESCAN    EQU     11
ERROR     EQU     12
WHOCON    EQU     13
;
          EXTRN    ISIS      ;LINK TO ISIS ENTRY POINT
;
;*****
;
;OPEN
;
          MVI     C,OPEN    ;SYSTEM CALL IDENTIFIER
          LXI     D,OBLK    ;ADDRESS OF PARAMETER BLOCK
          CALL    ISIS

```

```

LDA    OSTAT    ;TEST ERROR STATUS
ORA    A
JNZ    EXCEPT ;BRANCH TO EXCEPTION ROUTINE
;...
OBLK:                ;PARAMETER BLOCK FOR OPEN
      DW    OAFT    ;POINTER TO AFT
      DW    OFILE   ;POINTER TO FILENAME
ACCESS: DW    1     ;ACCESS, READ=1, WRITE=2,
      ;UPDATE=3
ECHO:   DW    0     ;IF ECHO < 0
      ;ECHO=AFTN OF ECHO OUTPUT FILE
      DW    OSTAT   ;POINTER TO STATUS
OAFT:   DS    2     ;AFTN (RETURNED)
OSTAT:  DS    2     ;STATUS (RETURNED)
OFILE:  DB    ':F0:FILE.EXT';FILE TO BE OPENED
;
;*****
;
;CLOSE
;
      MVI    C,CLOSE ;SYSTEM CALL IDENTIFIER
      LXI    D,CBLK ;ADDRESS OF PARAMETER BLOCK
      CALL   ISIS
      LDA    CSTAT   ;TEST ERROR STATUS
      ORA    A
      JNZ    EXCEPT ;BRANCH TO EXCEPTION ROUTINE
;...
CBLK:                ;PARAMETER BLOCK FOR CLOSE
CAFT:   DS    2     ;FILE AFTN
      DW    CSTAT   ;POINTER TO STATUS
;
CSTAT:  DS    2     ;STATUS (RETURNED)
;
;*****
;
;READ
;
      MVI    C,READ  ;SYSTEM CALL IDENTIFIER
      LXI    D,RBLK ;ADDRESS OF PARAMETER BLOCK
      CALL   ISIS
      LDA    RSTAT   ;TEST ERROR STATUS
      ORA    A
      JNZ    EXCEPT ;BRANCH TO EXCEPTION ROUTINE
;...
RBLK:                ;PARAMETER BLOCK FOR READ
RAFT:   DS    2     ;FILE AFTN
      DW    IBUF    ;ADDRESS OF INPUT BUFFER
RCNT:   DW    128   ;LENGTH OF READ REQUESTED
      DW    ACTUAL  ;POINTER TO ACTUAL
      DW    RSTAT   ;POINTER TO STATUS
;

```

```

ACTUAL: DS      2          ;COUNT OF BYTES READY
RSTAT:  DS      2          ;STATUS (RETURNED)
IBUF:   DS     128        ;INPUT BUFFER
;
;*****
;
;WRITE
;
        MVI     C,WRITE    ;SYSTEM CALL IDENTIFIER
        LXI     D,WBLK     ;ADDRESS OF PARAMETER BLOCK
        CALL    ISIS
        LDA     WSTAT      ;TEST ERROR STATUS
        ORA     A
        JNZ     EXCEPT   ;BRANCH TO EXCEPTION ROUTINE
;...
WBLK:   DS          ;PARAMETER BLOCK FOR WRITE
WAFT:   DS      2      ;FILE AFTN
        DW     OBUF      ;ADDRESS OF OUTPUT BUFFER
WCNT:   DW     128
        DW     WSTAT     ;POINTER TO STATUS
;
WSTAT:  DS      2      ;STATUS (RETURNED)
OBUF:   DS     128    ;OUTPUT BUFFER
;
;*****
;
;EXIT
;
        MVI     C,EXIT     ;SYSTEM CALL IDENTIFIER
        LXI     D,EBLK     ;ADDRESS OF PARAMETER BLOCK
        CALL    ISIS
;...
EBLK:   DW          ;PARAMETER BLOCK FOR EXIT
        DW     ESTAT     ;POINTER TO STATUS
;
ESTAT:  DS      2      ;STATUS (RETURNED)
;

```

Con los conceptos expuestos en esta sección y en la sección 4.3.1, es posible dar un ejemplo de la utilidad de las llamadas a ISIS-II a partir de un programa escrito en lenguaje ensamblador.

El programa de ejemplo, que será denominado TYPE, tendrá por objeto mostrar un archivo determinado en el dispositivo asignado como consola.

Para llamarlo se utilizaría el comando:

```
TYPE archivo
```

y es equivalente al programa ya existente definido como:

```
COPY archivo TO :CO:
```

Una vez que se ha completado sin errores el ensamble del programa, éste debe ser eslabonado a la biblioteca del sistema (SYSTEM.LIB), la cual contiene el símbolo público ISIS. La salida de este eslabonamiento (LINK) debe ser puesta en una localidad absoluta antes de ser ejecutado el programa.

```

;
; PROGRAMA DE EJEMPLO
; LLAMADO MEDIANTE EL
; COMANDO TYPE
;
OPEN EQU 0
CLOSE EQU 1
READ EQU 3
WRITE EQU 4
EXIT EQU 9
ERROR EQU 12
;
; EXTRN ISIS
;

```

```

;
; CSEG ; INICIO DEL SEGMENTO DE CODIGOS
BEGIN:
LXI SP,STACK+4
MVI C,READ ;LEE LA CONSOLA
LXI D,RBLK
CALL ISIS
LDA STATUS
ORA A
JNZ ERR
;
MVI C,OPEN ;ABRE EL ARCHIVO DE ENTRADA
LXI D,OBLK
CALL ISIS
LDA STATUS
ORA A
JNZ ERR
LHLD AFT
SHLD CAFT
;
LOOP:
MVI C,READ ;LEE EL ARCHIVO DE ENTRADA
LXI D,RBLK
CALL ISIS
LDA STATUS
ORA A
JNZ ERR
LHLD ACTUAL
MOVE A,H
ORA L
JZ DONE
MVI C,WRITE ;ESCRIBE EN LA CONSOLA
LXI D,WBLK
CALL ISIS
LDA STATUS
ORA A
JNZ ERR
JMP LOOP
DONE:
MVI C,CLOSE ;CIERRA EL ARCHIVO DE ENTRADA
LXI D,CBLK
CALL ISIS
MVI C,EXIT ;SALIDA NORMAL
LXI D,XBLK
CALL ISIS
;
ERR:
MVI C,ERROR ;MENSAJE DE ERROR
LXI D,EBLK
CALL ISIS

```

```

MVI    C,EXIT    ;SALIDA DE ERROR
LXI    D,XBLK
CALL   ISIS
;
OBLK:  DSEG      ;INICIO DEL SEGMENTO DE DATOS
        DW      AFT
        DW      BUFFER
        DW      1    ;LEE ACCESO
        DW      0    ;NO ECO
        DW      STATUS
;
CBLK:
CAFT:  DS      1
        DW      STATUS
;
RBLK:
AFT:   DW      1
        DW      BUFFER
        DW      128
        DW      ACTUAL
        DW      STATUS
;
WBLK:
        DW      0
        DW      BUFFER
ACTUAL: DS      2
        DW      STATUS
;
XBLK:
        DW      STATUS
;
EBLK:
STATUS: DS      2
        DW      STATUS
;
BUFFER: DS      128
;
STACK:  DS      4
;
END     BEGIN

```

El usuario de la interfaz entre la HP 3000 y la microcomputadora Intellec-MDS cuenta ya con el "hardware" en su totalidad (la tarjeta de interfaz, sección 3.3.4) y el soporte general de "software" (programa de control de la interfaz, sección 4.2.2).

En base a lo expuesto en las secciones 4.3.1 y 4.3.2, el usuario puede programar en base a sus necesidades particulares, creando un programa análogo al programa de ejemplo TYPE de la sección anterior.

La transferencia de un archivo editado en la HP 3000 hacia la microcomputadora exige, en suma, el programa que habilita la interfaz, la conecta a la HP y da aviso al usuario que se ha establecido la comunicación como primer aspecto, y como segundo, el programa del usuario donde se especifique el nombre del archivo presente en la HP y que se desea copiar a un archivo en "diskette", actividades que se llevan a cabo mediante las llamadas del sistema OPEN, READ, CLOSE, WRITE, etc.

Por otra parte, el sistema operativo MPE-III de la HP 3000 cuenta, según se indicó en la sección 1.3.1, con subsistemas de "software" que incluyen un juego de programas utilitarios.

Entre estos programas, el FCOPY-3000 es el indicado para la transferencia de archivos desde la HP hacia el SDM.

La secuencia de ejecución del programa FCOPY-3000 es la siguiente:

- Esperar por el "prompt" de la HP, que es el signo de dos puntos (:).

- Transmitir el comando:

```
:RUN FCOPY.PUB.SYS
```

- La HP responde con otro "prompt", que es el signo >. Esto indica que el programa utilitario FCOPY está corriendo, y espera el comando correspondiente.

- A continuación se transmite el comando de transferencia:

```
>FROM=archivo1;TO=archivo2
```

donde archivo1 es el archivo existente en la HP, y que va a ser copiado en el archivo2, creado mediante ISIS-II para recibir la información de la HP.

- Para dejar el programa utilitario FCOPY se transmite el comando:

```
>EXIT
```

- Se vuelve a recibir el "prompt" de la HP (:) indicando nuevamente la presencia del sistema operativo MPE-III.

- El "log-on" con la HP se hace transmitiendo tan solo un "carriage return", y el "log-off" mediante el



comando:

:BYE

Estos casos (EXIT, BYE, "carriage return") están contemplados en el programa de control de la interfaz.

El usuario de la interfaz determinará en que momento debe emplearse el comando FCOPY dentro de su programa codificado en lenguaje ensamblador del 8080. De igual modo, dará nombre a sus archivos en sustitución de los nombres simbólicos empleados en la sección anterior, si así lo desea.

El usuario de la interfaz no debe olvidar que las compañías de computación constantemente actualizan los sistemas de "software" de sus equipos, por lo que deberá atender a la última versión implementada en el sistema antes de proceder a la programación.

Con esto se quiere decir que el usuario no sólo puede actualizar sus programas una y otra vez, sino que debe hacerlo, optimizando el programa de control de la interfaz al adecuarlo a sus justas necesidades.

5

\* \* \* \* \*

C O N C L U S I O N E S

Los primeros frutos de un proyecto de tal naturaleza son los conocimientos prácticos adquiridos al trabajar en un sistema de cómputo real y no en un modelo teórico.

Efectivamente, las adiciones de "hardware" y la creación de programas para la interfaz son la llave para comprender el funcionamiento de otros tipos de microcomputadoras.

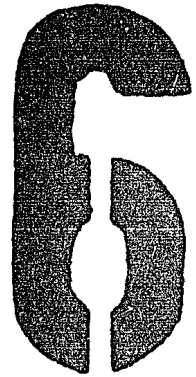
Otro aspecto de vital importancia, y rara vez tratado en los programas universitarios, es el referente a la comunicación de datos. El conocimiento práctico de los circuitos de interfaz y de los tipos de transmisión son las facetas técnicas de la ingeniería sin las cuales sería imposible conectar equipos entre sí físicamente.

El desarrollo de este proyecto es un prelude a la actividad profesional, pues establece una analogía exacta con los problemas que podría enfrentar el profesionista, desde su planteamiento y comprensión, hasta la búsqueda de un método de solución, lo que implica una consulta continua a textos y manuales, así como a profesionistas relacionados con el área que pudieran aportar datos indispensables para optimizar la solución del problema propuesto.

Este trabajo no puede ser una guía completa y exhaustiva sobre como diseñar una interfaz, pero sí un manual introductorio a la interfaz que se construyó.

La construcción de esta interfaz, una entre tantas, es un paso más para enlazar equipos digitales entre sí, y una invitación para el estudio de las infinitas posibilidades de interfaz entre el mundo analógico y el mundo digital.

\* \* \* \* \*



A P E N D I C E S

INSTRUCCIONES DEL 8080

- Código mnemotécnico
- Breve descripción de cada instrucción

Grupo de transferencia de datos

Grupo aritmético

Grupo lógico

Grupo de ramificación

Grupo de apilación, entrada/salida  
y control

JUMP	CALL	RETURN	RESTART	ROTATE	MOVE (cont)	ACCUMULATOR*	CONSTANT DEFINITION
C3 JMP	CD CALL	C9 RET	C7 RST 0	07 RLC	58 MOV E,B	80 ADD B	A8 XRA B
C2 JNZ	C4 CNZ	C0 RNZ	CF RST 1	0F RRC	59 MOV E,C	81 ADD C	A9 XRA C
CA JZ	CC CZ	C8 RZ	D7 RST 2	17 RAL	5A MOV E,D	82 ADD D	AA XRA D
D2 JNC	D4 CNC	D0 RNC	DF RST 3	1F RAR	5B MOV E,E	83 ADD E	AB XRA E
DA JC	DC CC	D8 RC	E7 RST 4		5C MOV E,H	84 ADD H	AC XRA H
E2 JPO	E4 CPO	E0 RPO	EF RST 5		5D MOV E,L	85 ADD L	AD XRA L
EA JPE	EC CPE	E8 RPE	F7 RST 6		5E MOV E,M	86 ADD M	AE XRA M
F2 JP	F4 CP	F0 RP	FF RST 7	CONTROL	5F MOV E,A	87 ADD A	AF XRA A
FA JM	FC CM	F8 RM					
E9 PCHL							
				00 NOP	60 MOV H,B	88 ADC B	B0 ORA B
				78 HLT	61 MOV H,C	89 ADC C	B1 ORA C
				F3 DI	62 MOV H,D	8A ADC D	B2 ORA D
				FB EI	63 MOV H,E	8B ADC E	B3 ORA E
					64 MOV H,H	8C ADC H	B4 ORA H
					65 MOV H,L	8D ADC L	B5 ORA L
					66 MOV H,M	8E ADC M	B6 ORA M
					67 MOV H,A	8F ADC A	B7 ORA A
				40 MOV B,B	68 MOV L,B	90 SUB B	B8 CMP B
				41 MOV B,C	69 MOV L,C	91 SUB C	B9 CMP C
				42 MOV B,D	6A MOV L,D	92 SUB D	BA CMP D
				43 MOV B,E	6B MOV L,E	93 SUB E	BB CMP E
				44 MOV B,H	6C MOV L,H	94 SUB H	BC CMP H
				45 MOV B,L	6D MOV L,L	95 SUB L	BD CMP L
				46 MOV B,M	6E MOV L,M	96 SUB M	BE CMP M
				47 MOV B,A	6F MOV L,A	97 SUB A	BF CMP A
				48 MOV C,B	70 MOV M,B	98 SBB B	
				49 MOV C,C	71 MOV M,C	99 SBB C	
				4A MOV C,D	72 MOV M,D	9A SBB D	
				4B MOV C,E	73 MOV M,E	9B SBB E	
				4C MOV C,H	74 MOV M,H	9C SBB H	
				4D MOV C,L	75 MOV M,L	9D SBB L	
				4E MOV C,M		9E SBB M	
				4F MOV C,A	77 MOV M,A	9F SBB A	
				50 MOV D,B	78 MOV A,B	A0 ANA B	ORG Adr
				51 MOV D,C	79 MOV A,C	A1 ANA C	END
				52 MOV D,D	7A MOV A,D	A2 ANA D	EQU D16
				53 MOV D,E	7B MOV A,E	A3 ANA E	SET D16
				54 MOV D,H	7C MOV A,H	A4 ANA H	DS D16
				55 MOV D,L	7D MOV A,L	A5 ANA L	DB DB
				56 MOV D,M	7E MOV A,M	A6 ANA M	DW D16
				57 MOV D,A	7F MOV A,A	A7 ANA A	IF D16
							ENDIF
							MACRO
							ENDM
							FLAG BYTE
							STACK FORMAT
							7 6 5 4 3 2 1 0
							S Z O C O P I C

DB = constant, or logical/arithmetic expression that evaluates to an 8 bit data quantity.

D16 = constant, or logical/arithmetic expression that evaluates to a 16 bit data quantity

Adr = 16 bit address

\*\* = all flags except CARRY affected; (exception: INX & DCX affect no flags)

† = only CARRY affected

\* all Flags (C, Z, S, P, AC) affected

MOV r1,r2 (Move Register)

El contenido del registro r2 es movido al registro r1.

MOV r,M (Move from Memory)

El contenido de la localidad de memoria, cuya dirección se encuentra en los registros H y L, es movido al registro r.

MOV M,r (Move to Memory)

El contenido del registro r, es movido a la localidad de memoria cuya dirección está en los registros H y L.

MVI r,data (Move Immediate)

El contenido del byte 2 de la instrucción (data), es movido al registro r.

MVI M,data (Move to memory immediate)

El contenido del byte 2 de la instrucción (data), es movido a la localidad de memoria cuya dirección se encuentra en los registros H y L.

LXI rp,data 16 (Load register pair immediate)

El byte 3 de la instrucción es movido al registro de alto orden del registro par rp. El byte 2 de la instrucción es movido al registro de bajo orden del registro par rp.



LDA address (Load Accumulator direct)

150

El contenido de la localidad de memoria, cuya dirección está especificada en los bytes 2 y 3 de la instrucción, es movido al registro A.

STA address (Store Accumulator direct)

El contenido del acumulador es movido a la localidad de memoria cuya dirección está especificada en los bytes 2 y 3 de la instrucción.

LHLD address (Load H and L direct)

El contenido de la localidad de memoria, cuya dirección está especificada en los bytes 2 y 3 de la instrucción, es movido al registro L. El contenido de la localidad de memoria de la dirección siguiente, es movido al registro H.

SHLD address (Store H and L direct)

El contenido del registro L es movido a la localidad de memoria cuya dirección está dada en los bytes 2 y 3. El contenido del registro H es movido a la siguiente localidad de memoria.

LDAX rp (Load accumulator indirect)

El contenido de la localidad de memoria, cuya dirección está en el registro par rp, es movido al registro A.  
Nota: sólo los registros pares B,C, o D,E, pueden ser especificados.

STAX rp            (Store accumulator indirect)            151

El contenido del registro A es movido a la localidad de memoria cuya dirección está en el registro par rp. Nota: sólo los registro pares B,C y D,E, pueden ser especificados.

XCHG                (Exchange H and L with D and E)

El contenido de los registros H y L es intercambiado con el contenido de los registros D y E.

Ninguna instrucción de este grupo, afectará a ninguna de las banderas de condición.

ADD r            (Add Register)

El contenido del registro r es sumado al contenido del acumulador. El resultado permanece en el acumulador.

ADD M

El contenido de la localidad de memoria cuya dirección está contenida en los registros H y L, es sumada al contenido del acumulador. El resultado se coloca en el acumulador.

ADI data        (Add immediate)

El contenido del segundo byte de la instrucción es sumado al contenido del acumulador. El resultado queda en el acumulador.

ADC r            (Add Register with carry)

El contenido del registro r y el contenido del bit de carry son sumados al contenido del acumulador. El resultado queda en el acumulador.

ADC M            (Add memory with carry)

El contenido de la localidad de memoria cuya dirección está contenida en los registros H y L, y el contenido del bit de carry, son sumados al acumulador. El resultado queda en el acumulador.

ACI data (Add immediate with carry) 153

El contenido del segundo byte de la instrucción y el contenido de la bandera CY son sumados al acumulador. El resultado queda en el acumulador.

SUB r (Subtract Register)

El contenido del registro r es restado del contenido del acumulador. El resultado queda en el acumulador.

SUB M (Subtract memory)

El contenido de la localidad de memoria cuya dirección está contenida en los registros H y L, es restado del contenido del acumulador. El resultado queda en el acumulador.

SUI data (Subtract immediate)

El contenido del byte 2 de la instrucción es restado del contenido del acumulador. El resultado queda en el acumulador.

SBB r (Subtract Register with borrow)

El contenido del registro r y el contenido de la bandera CY son restados del acumulador. El resultado queda en el acumulador.

SBB M (Subtract memory with borrow)

El contenido de la localidad de memoria cuya dirección está en los registros H y L, y el contenido del bit de carry, son restados del contenido del acumulador. El resultado queda en el acumulador.

SBI data (Subtract immediate with borrow) 154

El contenido del segundo byte de la instrucción y el contenido de la bandera CY, son restados del acumulador. El resultado queda en el acumulador.

INR r (Increment Register)

El contenido del registro r es incrementado en uno.

Nota: todas las banderas de condición, excepto CY, son afectadas.

INR M (Increment memory)

El contenido de la localidad de memoria cuya dirección está dada por los registros H y L, es incrementado en uno. Nota: todas las banderas, excepto CY, son afectadas.

DCR r (Decrement Register)

El contenido del registro r es decrementado en uno.

Nota: todas la banderas son afectadas, excepto CY.

DCR M (Decrement memory)

El contenido de la localidad de memoria cuya dirección está en los registros H y L, es decrementado en uno. Nota: todas las banderas son afectadas, excepto CY.

INX rp (Increment register pair)

El contenido del registro par rp es incrementado en uno. Nota: No se afecta ninguna bandera.

DCX rp

(Decrement register pair)

155

El contenido del registro par rp es decrementado en uno. Nota: no se afecta ninguna bandera.

DAD rp

(Add register pair to H and L)

El contenido del registro par rp es sumado al contenido del registro par H,L. El resultado queda en el registro H,L. Nota: sólo la bandera CY es afectada.

DAA

(Decimal Adjust Accumulator)

El número de ocho bits en el acumulador, es ajustado para formar dos números de cuatro bits Binarios codificados en Decimal.

A menos que se haya indicado otra cosa, todas las instrucciones de este grupo afectan a todas las banderas, de acuerdo a las reglas estándar.

Banderas:

Cero Z

Signo S

Paridad P

Acarreo CY

Acarreo Auxiliar AC

ANA r (AND Register)

Se efectúa una función AND entre el contenido del registro r y el contenido del acumulador. El resultado queda en el acumulador. La bandera CY se borra.

ANA M (AND memory)

Se efectúa una función AND entre el contenido de la localidad de memoria cuya dirección está dada por los registros H y L, y el contenido del acumulador. El resultado queda en el acumulador. La bandera CY se borra.

ANI data (AND immediate)

Se realiza una operación AND entre el segundo byte de la instrucción y el contenido del acumulador. El resultado queda en el acumulador. Las banderas CY y AC se borran.

XRA r (Exclusive OR Register)

Se realiza una operación OR-exclusiva entre el contenido del registro r y el contenido del acumulador. El resultado queda en el acumulador. Las banderas CY y AC se borran.

XRA M (Exclusive OR memory)

Se realiza una operación OR-exclusiva entre el contenido de la localidad de memoria cuya dirección está dada por los registros H y L, y el contenido del acumulador. Las banderas CY y AC se borran. El resultado queda en el acumulador.

XRI data (Exclusive OR immediate) 157

Se realiza una operación OR-exclusiva entre el byte 2 de la instrucción y el acumulador. El resultado queda en el acumulador. Las banderas CY y AC se borran.

ORA r (OR Register)

Se realiza una operación OR entre el contenido del registro r y el contenido del acumulador. El resultado queda en el acumulador. Las banderas CY y AC se borran.

ORA M (OR memory)

Se realiza una operación OR entre el contenido de la localidad de memoria cuya dirección está dada por los registros H y L, y el contenido del acumulador. El resultado queda en el acumulador. Las banderas CY y AC se borran.

ORI data (OR immediate)

Se realiza una operación OR entre el byte 2 de la instrucción y el contenido del acumulador. El resultado queda en el acumulador. Las banderas CY y AC se borran.

CMP r (Compare Register)

El contenido del registro r es restado del acumulador. El acumulador permanece inalterado. Las banderas se afectan como resultado de la substracción. La bandera Z se pone a 1 si  $(A)=(r)$ . La bandera CY se pone a 1 si  $(A)<(r)$ .



El contenido de la localidad de memoria cuya dirección está dada por los registros H y L, es restado del acumulador. El acumulador permanece inalterado. Las banderas se afectan como resultado de la substracción. La bandera Z se pone a 1 si  $(A)=[(H)(L)]$ . La bandera CY se pone a 1 si  $(A)<[(H)(L)]$ .

## CPI data (Compare immediate)

El contenido del segundo byte de la instrucción es restado del acumulador. El acumulador permanece inalterado. Las banderas se afectan como resultado de la substracción. La bandera Z se pone a 1 si  $(A)=(\text{byte } 2)$ . La bandera CY se pone a 1 si  $(A)<(\text{byte } 2)$ .

## RLC (Rotate left)

El contenido del acumulador se rota hacia la izquierda una posición. El bit menos significativo y la bandera CY toman el valor del bit más significativo. Sólo la bandera CY es afectada.

## RRC (Rotate right)

El contenido del acumulador se rota hacia la derecha una posición. El bit más significativo y la bandera CY toman el valor del bit menos significativo. Sólo la bandera CY es afectada.

RAL

(Rotate left through carry)

159

El contenido del acumulador es rotado una posición hacia la izquierda, incluyendo al bit CY. El bit menos significativo toma el valor del acarreo, y el acarreo toma el valor del bit más significativo. Sólo se afecta la bandera CY.

RAR

(Rotate right through carry)

El contenido del acumulador es rotado una posición hacia la derecha, incluyendo al bit CY. El bit más significativo toma el valor del acarreo, y el acarreo toma el valor del bit menos significativo. Sólo se afecta la bandera CY.

CMA

(Complement accumulator)

Los bits del acumulador son complementados (los "0" se vuelven "1" y viceversa). No se afecta ninguna bandera.

CMC

(Complement carry)

La bandera CY es complementada. No se afectan las demás.

STC

(Set carry)

La bandera CY es puesta a 1. No se afectan las demás.

A menos que se haya indicado lo contrario, todas las instrucciones de este grupo afectan a todas las banderas.

Este grupo de instrucciones altera la secuencia normal del flujo del programa. Ninguna bandera es alterada por estas instrucciones.

En este grupo, las instrucciones pueden ser condicionales o incondicionales. Las transferencias incondicionales tan sólo realizan la operación especificada en el registro PC (Contador del Programa). Las transferencias condicionales examinan el estado de una de una de cuatro banderas (Z, S, P, CY) para determinar si se ejecuta o no la transferencia. Las condiciones pueden ser:

NZ - no cero (Z=0)

Z - cero (Z=1)

NC - no acarreo (CY=0)

C - acarreo (CY=1)

PO - paridad non (P=0)

PE - paridad par (P=1)

P - positivo (S=0)

M - negativo (S=1)

JMP address (Jump)

El control es transferido a la instrucción cuya dirección está dada por los bytes 2 y 3 de la instrucción JMP.

Jcondition address (Conditional jump)

Si la condición especificada es verdadera, el control es transferido a la instrucción cuya dirección está dada por

los bytes 2 y 3 de la instrucción presente. En caso contrario, el control continúa secuencialmente.

#### CALL address (Call)

Los 8 bits de alto orden de la dirección de la siguiente instrucción, son movidos a la localidad de memoria cuya dirección está indicada por el contenido del registro SP, pero decrementado en uno. Los 8 bits de bajo orden de la dirección de la siguiente instrucción, son movidos a la localidad de memoria cuya dirección está indicada por el contenido del registro SP, pero decrementado en dos. El control es transferido a la instrucción cuya dirección está especificada en los bytes 3 y 2 de la instrucción presente.

#### Ccondition address (Condition call)

Si la condición especificada es verdadera, se llevan a cabo las acciones especificadas en la instrucción CALL. De otro modo, el control continúa secuencialmente.

#### RET (Return)

El contenido de la localidad de memoria cuya dirección está especificada en el registro SP, es movido a los 8 bits de bajo orden del registro PC. El contenido de la localidad de memoria cuya dirección está especificada por el contenido del registro SP incrementado en uno, es movido a los 8 bits de alto orden del registro PC. El contenido del registro SP es incrementado en dos.

Rcondition (Conditional return)

162

Si la condición especificada es verdadera, se llevan cabo las acciones especificadas en la instrucción RET. De otro modo, el control continúa secuencialmente.

RST n (Restart)

Se llevan a cabo las acciones descritas para la instrucción CALL, sólo que el control es transferido a la instrucción cuya dirección es ocho veces el contenido de NNN, donde NNN es la representación binaria de 000 hasta 111, para los números del 1 al 7 respectivamente.

PCHL (Jump H and L indirect - move H and L to PC)

El contenido del registro H se mueve a los 8 bits de alto orden del registro PC. El contenido del registro L se mueve a los 8 bits de bajo orden del registro PC.

PUSH rp            (Push)

El contenido del registro de alto orden del registro par rp es movido a la localidad de memoria cuya dirección está dada por el contenido del registro SP menos uno. El contenido del registro de bajo orden del registro par rp es movido a la localidad de memoria cuya dirección está dada por el contenido del registro SP menos dos. El registro SP es decrementado en dos. Nota: no puede especificarse  $rp = SP$ .

PUSH PSW            (Push processor status word)

El contenido del registro A es movido a la localidad de memoria cuya dirección está dada por el registro SP menos uno. El contenido de las banderas de condición es ensamblado en una palabra de estado, y esta palabra es movida a la localidad de memoria cuya dirección está dada por el registro SP menos dos. El contenido del registro SP es decrementado en dos.

POP rp            (Pop)

El contenido de la localidad de memoria, cuya dirección está dada por el contenido del registro SP, es movido al registro de bajo orden del registro par rp. El contenido de la localidad de memoria, cuya dirección está dada por el registro SP más uno, es movido al registro de alto orden del registro par rp. El registro SP es incrementado en 2

El contenido de la localidad de memoria cuya dirección está dada por el contenido del registro SP, se usa para restaurar las banderas de condición. El contenido de la localidad de memoria cuya dirección está dada por el contenido del registro SP más uno, es movido al registro A. El contenido del registro SP es incrementado en 2.

XTHL (Exchange stack top with H and L)

El contenido del registro L es intercambiado con el contenido de la localidad de memoria direccionada por el contenido del registro SP. El contenido del registro H es intercambiado con el contenido de la localidad de memoria direccionada por el registro SP más uno.

SPHL (Move HL to SP)

El contenido de los registros H y L (16 bits) es movido al registro SP.

IN port (Input)

El dato colocado en el canal bidireccional de datos de 8 bits, a través del puerto especificado, es movido al registro A.

OUT port (Output)

El contenido del registro A es colocado en el canal de datos bidireccional de 8 bits, para ser transmitido al puerto especificado.

EI (Enable interrupts)

165

El sistema de interrupciones es habilitado una vez que se haya ejecutado la instrucción siguiente.

DI (Disable interrupts)

El sistema de interrupciones es deshabilitado inmediatamente después de la instrucción DI.

HLT (Halt)

El procesador es detenido. Banderas y registros no son afectados.

NOP (No operation)

No se realiza ninguna operación. Banderas y registros no son afectados.

A menos que se haya indicado lo contrario, las instrucciones de este grupo no afectan a las banderas de condición.

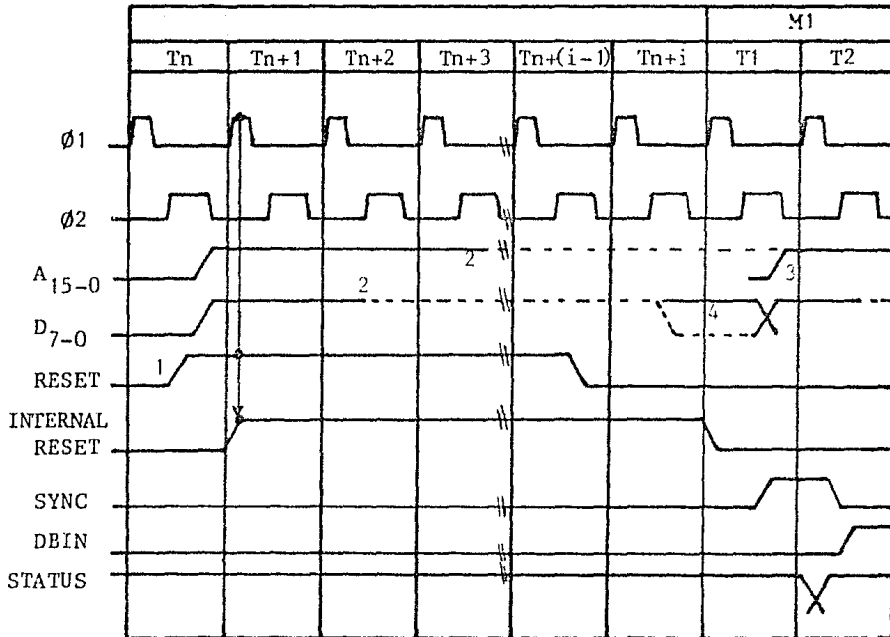


NOTACION HEXADECIMAL  
DEL CODIGO  
ASCII

Notación Hexadecimal del Código ASCII 167

00	NUL	1B	ESC	36	6	51	Q	6C	l
01	SOH	1C	FS	37	7	52	R	6D	m
02	STX	1D	GS	38	8	53	S	6E	n
03	ETX	1E	RS	39	9	54	T	6F	o
04	EOT	1F	US	3A	:	55	U	70	p
05	ENQ	20	SP	3B	;	56	V	71	q
06	ACK	21	!	3C	<	57	W	72	r
07	BEL	22	"	3D	=	58	X	73	s
08	BS	23	#	3E	>	59	Y	74	t
09	HT	24	\$	3F	?	5A	Z	75	u
0A	LF	25	%	40	@	5B	[	76	v
0B	VT	26	&	41	A	5C	\	77	w
0C	FF	27	^	42	B	5D	]	78	x
0D	CR	28	(	43	C	5E	^	79	y
0E	SO	29	)	44	D	5F	_	7A	z
0F	SI	2A	*	45	E	60	`	7B	{
10	DLE	2B	+	46	F	61	a	7C	
11	DC1 (X-ON)	2C	,	47	G	62	b	7D	}
12	DC2 (TAPE)	2D	-	48	H	63	c	7E	~ (ALT MODE)
13	DC3 (X-OFF)	2E	.	49	I	64	d	7F	DEL (RUB OUT)
14	DC4 (TAPE)	2F	/	4A	J	65	e		
15	NAK	30	0	4B	K	66	f		
16	SYN	31	1	4C	L	67	g		
17	ETB	32	2	4D	M	68	h		
18	CAN	33	3	4E	N	69	i		
19	EM	34	4	4F	O	6A	j		
1A	SUB	35	5	50	P	6B	k		

- Respuesta en el tiempo de los canales de datos y direcciones del 8080 en el momento de un "reset".



1 Cuando se activa la señal "reset", todas las señales de salida de control recibirán un "reset" inmediatamente o algunos períodos de reloj después. La señal "reset" debe permanecer activa por un mínimo de tres ciclos de reloj. Las variables "n" e "i" pueden tomar cualquier valor entero.

2 Señales flotando.

3 Contador de programa = 0.

4 Estado desconocido.

- Subrutina de inicialización del USART.
- Subrutina para transmitir datos del USART al exterior.
- Subrutina para admitir en el USART datos provenientes del exterior.



```

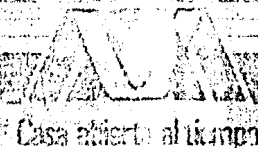
LOC  LOC      LINE      SOURCE STATEMENT
-----
1 ***** T E L Q U T *****
2
3 FUNCTION: TEOULT
4 DESCRIPTION: TEOULT WAITS UNTIL THE TELEPRINTER IS READY TO ACCEPT A
5 CHARACTER AND THEN SENDS THE INPUT ARGUMENT TO THE TELEPRINTER.
6
7 ORG      S800H
8 TEOULT: IN      TELS          IGET STATUS OF TELEPRINTER
9          JNDY          ICHECK FOR TRANSMIT BUFFER READY
10         JZ      TEOULT        ILOOP UNTIL READY
11         MOV     A,C
12         OUT    TEO          OUTPUT CHARACTER
13         RET
14
15 *****
16
17 TELEPRINTER EQUATES
18
19 MIFO EQU 000H          MODE INSTRUCTION FORMAT
20 CIFO EQU 035H          COMMAND INSTRUCTION FORMAT
21 TRDY EQU 00000001B     TRANSMIT READY MASK
22 RBR  EQU 00000010B     RECEIVE BUFFER READY MASK
23 DSR  EQU 10000000B     DATA SET READY MASK
24 TELCTL EQU 044H        TELEPRINTER (USART) CONTROL PORT
25 TELS  EQU 044H        TELEPRINTER INPUT STATUS PORT
26 TELI  EQU 045H        TELEPRINTER INPUT DATA PORT
27 TELQ  EQU 045H        TELEPRINTER OUTPUT DATA PORT
28      END
    
```

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS  
MIFO A 000H    DSR    A 0080    MIFO    A 000H    RBR    A 0002    TELCTL A 0044    TELI    A 0045    TELQ    A 0045  
TEOULT A S800    TELS    A 0044    TRDY    A 0001

ASSEMBLY COMPLETE. NO ERRORS.



LOC OBJ LINE SOURCE STATEMENT

```

1 ***** TELIN *****
2
3 FUNCTION: TELIN
4 DESCRIPTION: TELIN WAITS UNTIL A CHARACTER HAS BEEN ENTERED AT THE TELE-
5 PRINTER AND THEN RETURNS THE CHARACTER VIA THE A REGISTER, TO THE
6 CALLING ROUTINE.
7
8 ORG 5700H
9
10 TELIN: IN     TELS      ;GET STATUS OF TELEPRINTER
11      ANI     RBR       ;CHECK FOR RECEIVE BUFFER READY
12      JZ      TELIN     ;LOOP UNTIL READY
13      IN     TELS      ;READY SO GET CHARACTER
14      RET

```

\*\*\*\*\*

```

17 TELEPRINTER EQUATES
18 MIFO EQU 0C0H ;MODE INSTRUCTION FORMAT
19 CIFO EQU 0B5H ;COMMAND INSTRUCTION FORMAT
20 TRDY EQU 0000001B ;TRANSMIT READY MASK
21 RBR EQU 0000010B ;RECEIVE BUFFER READY MASK
22 DSR EQU 1000000B ;DATA SET READY MASK
23 TELCTL EQU 044H ;TELEPRINTER (USART) CONTROL PORT
24 TELS EQU 044H ;TELEPRINTER INPUT STATUS PORT
25 TELI EQU 045H ;TELEPRINTER INPUT DATA PORT
26 TELS EQU 045H ;TELEPRINTER OUTPUT DATA PORT
27
28 END

```

USER SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS  
MIFO A 00C0 DSR A 00B0 MIFO A 00C0 RBR A 0002 TELCTL A 0C44 TELI A 0045 TELIN A 5700  
TELD A 0045 TELS A 0044 TRDY A 0001

ASSEMBLY COMPLETE NO ERRORS

Programa de control de la interfaz.



LOC	OBJ	LINE	SOURCE STATEMENT
		1	PROGRAMA LOGON
		2	
		3	
		4	ESTE PROGRAMA PERMITE LA INTERCOMUNICACION ENTRE EL MOS Y LA COMPUTADORA
		5	HP-3000.
		6	CUANDO SE ESCRIBE EN LA CONSOLA EL CPU DEVUELVE CADA ECO.
		7	PARA SACAR LOS CARACTERES AL PUERTO TECLAR UN RETURN
		8	LOS CARACTERES PROVENIENTES DE LA HP VAN APARECIENDO EN LA CONSOLA
		9	CONFORME VAN LLEGANDO AL PUERTO.
		10	EL ULTIMO CARACTER DEBE SER EL 'PROMPT' DE LA HP INDICANDO QUE SE HA
		11	VUELTO A MODO DE CONSOLA.
		12	EN EL MODO DE CONSOLA EXISTEN LAS SIGUIENTES FUNCIONES.
		13	PARA BORRAR UNA LINEA TECLAR CONTROL-X, EL ECO ES UN \$.
		14	PARA BORRAR UN CARACTER TECLAR CONTROL-H O DELETE. ESTO BORRA DE LA
		15	MEMORIA EL CARACTER REPITIENDO SU ECO.
		16	
		17	U. A. M.
		18	
		19	
		20	
		21	----- USART INITIALIZATION CODE -----
		22	
		23	MODE INSTRUCTION
		24	
		25	2 STOP BITS
		26	NO PARITY
		27	8 BIT CHARACTERS
		28	BAUD RATE FACTOR OF 16
		29	
		30	COMMAND INSTRUCTION
		31	
		32	NO HUNT MODE
		33	NOT (RTS) FORCED TO 0
		34	RESET ALL ERROR FLAGS
		35	RECEIVE ENABLED
		36	TRANSMIT ENABLED
		37	
5555		38	ORG 5555H
5555 3E3E		39	MVI A, MIFO ;MODE INSTRUCTION
5557 D344		40	OUT TELCTL ;OUTPUT MODE SET TO USART
5559 3E35		41	MVI A, CIFO ;COMMAND INSTRUCTION
555B D344		42	OUT TELCTL ;OUTPUT COMMAND WORD TO USART
		43	EJECT

LOC	OBJ	LINE	SOURCE STATEMENT
		44	;
		45	;..... INICIO DEL PROGRAMA *****
		46	;
555D	31F859	47	LXI SP,APU ;APUNTA A LA CONSOLA
5560	0E28	48	MVI C,'+' ;AVISAR A CONSOLA PROGRAMA LISTO PARA
		49	;INTERCONEXION.
5562	1EFF	50	MVI E,MENOS ;CONTADOR INICIAL PARA LOG-OFF
5564	CD09FB	51	CALL CO
5567	CD03F8	52	CALL CI ;ESPERA INSTRUCCION PARA ACTUAR
556A	E67F	53	ANI PRY ;QUITA PARIDAD
556C	FE0D	54	CPI CR ;ES LA INSTRUCCION CORRECTA?
556E	CA7255	55	JZ SIGUE ;SI ES REALIZA INTERCONEXION
5571	C7	56	RST 0 ;NO ES LA INSTRUCCION DEBIDA
		57	;VE A MONITOR.
		58	;
		59	;
		60	;
		61	;EN ESTA PARTE DEL PROGRAMA SE REALIZA LA INTERCONEXION CON LA HP.
		62	;
5572	47	63	SIGUE: MOV B,A ;GUARDA INSTRUCCION
5573	CD0256	64	CALL RETLF ;REGRESA CURSOR EN LA CONSOLA
5576	48	65	MOV C,B ;SACA INSTRUCCION AL PUERTO
5577	CDE256	66	CALL TELOUT
557A	CDD856	67	AQUI: CALL TELIN ;ESPERA POR CARACTER
557D	E67F	68	ANI PRY ;QUITA PARIDAD
557F	47	69	MOV B,A ;GUARDA CARACTER
5580	4F	70	MOV C,A ;PRODUCE SU ECO
5581	CD09FB	71	CALL CO
5584	FE3A	72	CPI ':' ;ES PROMPT DE LA HP?
5586	C27A55	73	JNZ AQUI ;CONTINUA SECUENCIA SI NO LO ES
5589	C3C555	74	JMP INI ;CONTINUA PROGRAMA SI SI ES
		75	;
558C	CDE256	76	START: CALL TELOUT
558F	3E00	77	MVI A,CONT ;CHECA SI ES LOG-OFF
5591	BB	78	CMP E
5592	CAB656	79	JZ FIN ;TERMINA SI ES LOG-OFF
		80	EJECT

LOC	ORG	LINE	SOURCE STATEMENT
		81	;
		82	;
		83	;
		84	;RECEPCION DE MENSAJE DE LA HP.
		85	;
5595	CD1357	86	ATRÁS: CALL LOOP2 ;ESPERA CARACTER DE CONSOLA
5598	CDD856	87	CALL TELIN ;ESPERA CARACTER DE HP
559B	E67F	88	ANI PRTY ;QUITA PARIDAD
559D	47	89	MOV B,A ;GUARDA CARACTER
559E	4F	90	MOV C,A ;PRODUCE SU ECO
559F	CDC9FB	91	CALL CO
55A2	78	92	MOV A,B ;RESTITUYE CARACTER
55A3	FE0A	93	CPI RETRN ;SE APROXIMA FIN DE MENSAJE?
55A5	C29555	94	JNZ ATRAS ;CONTINUA SECUENCIA SI NO LO ES
55A6	CDD656	95	CALL TELIN ;ESPERA POR CARACTER FINAL
55AB	E67F	96	ANI PRTY ;QUITA PARIDAD
55AD	47	97	MOV B,A ;GUARDA CARACTER
55AE	4F	98	MOV C,A ;PRODUCE SU ECO
55AF	CDC9FB	99	CALL CO
55B2	78	100	MOV A,B ;RESTITUYE CARACTER
55B3	FE3A	101	CPI ':' ;ES "PROMPT" DE LA H.P.?
55B5	CAC555	102	JZ INI ;CAMBIA DE MODO SI ASI ES
55B8	FE3E	103	CPI '>' ;IDEM
55BA	CAC555	104	JZ INI
55BD	FE2F	105	CPI '/' ;IDEM
55BF	CAC555	106	JZ INI
55C2	C29555	107	JNZ ATRAS ;CONTINUA SECUENCIA SI NO LO ES
		108	\$ EJECT

LOC	OBJ	LINE	SOURCE STATEMENT
		109	;
		110	;
		111	;
		112	; CAMBIO A MODO DE CONSOLA.
		113	;
55C5	212857	114	INI: LXI H, MEM ;ALMACEN DE MEMORIA
55C8	31F859	115	LXI SP, APU ;APUNTAADOR DE STACK
55CB	1600	116	MVI D, CONT ;CONTADOR PARA CARACTERES
55CD	1EFF	117	MVI E, MENOS ;INDICADOR PARA LOG-OFF
55CF	14	118	NEXT: INR D ;VE CONTANDO CARACTERES
55D0	CDED56	119	LAST: CALL LOOP; ;RECEPCION DE MENSAJE DE OPERADOR
55D3	3E02	120	MVI A, STX ;CHECA SI HUBO MENSAJE
55D5	B9	121	CMP C
55D6	C2DA55	122	JNZ CONIN ;SALTA SI NO HUBO
55D9	05	123	PUSH B ;GUARDA INDICADOR DE MENSAJE
55DA	CD03F8	124	CONIN: CALL CI ;ESPERA CARACTER DE CONSOLA
55DD	E67F	125	ANI PRTY ;QUITA PARIDAD
55DF	FE18	126	ANTES: CPI CNTLX ;ES CONTROL X ?
55E1	C9F56	127	JZ OTRA ;SI, BORRA LINEA
55E4	FE08	128	CPI CNTLH ;ES CONTROL H ?
55E6	C9A556	129	JZ BORR
55E9	FE02	130	CPI STX ;INSTRUCCION PARA ACEPTAR MENSAJE DE HP
55EB	C93056	131	JZ MESSG
55EE	FE7F	132	CPI DEL ;ES 'DELETE'?
55F0	C9A553	133	JZ BORR ;SI, BORRA CARACTER
55F3	FE0D	134	CPI CR ;ES RETURN?
55F5	C9D656	135	JZ RETU ;SALTA SI LO ES
55F8	FE42	136	CPI 'B' ;PRIMER CARACTER PARA LOG-OFF
55FA	CAE056	137	JZ UNO ;INDICA SI ES
55FD	77	138	MOV M, A ;GUARDA CARACTER EN MEMORIA
55FE	4F	139	MOV C, A ;PREPARA ECO
55FF	23	140	INX H ;PREPARA MEMORIA PARA SIGUIENTE CARACTER
5606	CD03F8	141	CALL CO ;PRODUCE ECO EN CONSOLA
5603	C3CF55	142	JMP NEXT ;VE POR SIGUIENTE CARACTER
		143	;

LOC	OBJ	LINE	SOURCE STATEMENT
		144	;
		145	;
		146	;
		147	ENVIO DEL MENSAJE A LA HP
		148	;
		149	;
5606	0E0D	150	RETU: MVI C,CR ;PREPARA CONSOLA
5608	0E09FB	151	CALL CO
5608	0E20	152	MVI C,ESPA ;ESPACIO EN BLANCO
560D	0C09FB	153	CALL CO
5610	212E57	154	LVI H,MEM ;PREPARA CARACTERES
5613	15	155	LOOP: DCP D ;LLEVA CUENTA DE CARACTERES
5614	CA2B56	156	JZ MODE ;AVISA CUANDO TERMINE
5617	7F	157	MOV A,M ;PREPARA CARACTER
5618	F660	158	CPI MASK ;FORZA A 1 EL OCTAVO BIT
561A	4F	159	MOV C,A ;PREPARA ENVIO
561B	0CE256	160	CALL TELCUT
561E	23	161	INX H ;PREPARA MEMORIA
561F	0DD856	162	CALL TELIN ;RECIBE ECO DE LA HP
5622	E67F	163	ANI PRY ;QUITA PARIDAD
5624	4F	164	MOV C,A ;PRODUCE ECO EN LA CONSOLA
5625	0C09FB	165	CALL CO
5628	C31356	166	JMP LOOP
		167	;
562B	0F0D	168	MODE: MVI C,CR ;INSTRUCCION FINAL PARA HP
562D	C38C55	169	JMP START ;PREPARA PUERTO PARA RESPUESTA
		170	;
		171	;
		172	;
		173	Mensaje de operador de la HP.
		174	;
5630	15	175	MSGG: DCR D ;NO CONTAR EL STX
5631	C1	176	POP B ;CHECA SI HUBO RECEPCION
5632	79	177	MOV A,C
5633	F02	178	CPI STX
5635	C2CF55	179	JNZ NEXT ;CONTINUA PROGRAMA PRINCIPAL SI NO HUBO
		180	RECEPCION
5638	0DC256	181	CALL RETLF_____ push H ;PREPARA CONSOLA
563B	21785E	182	IXI H,BUFF ;DIRECCION INICIAL DE MENSAJE
563E	05	183	OTRO: DCR B ;LLEVA CUENTA DE CARACTERES
563F	CA4A56	184	JZ TERMI ;SALTA CUANDO TERMINE
5642	4E	185	MOV C,M ;PRODUCE ECO EN CONSOLA
5643	0C09FB	186	CALL CO
5646	23	187	INX H ;PREPARA SIGUIENTE CARACTER
5647	C33E56	188	JMP OTRO ;CONTINUA
		189	;
564A	0DC256	190	TERMI: CALL RETLF_____ pop H ;PREPARA CONSOLA
564D	C3CF55	191	JMP NEXT ;CONTINUA PROGRAMA PRINCIPAL
		192	* EJECT

LOC	OBJ	LINE	SOURCE STATEMENT
		193	;
		194	;
		195	;
		196	EN ESTA PARTE DEL PROGRAMA SE CHECA SI HAY UN LOG-OFF PARA LA HP.
		197	;
5650	1C	198 UNO:	INR E ;INDICADOR PARA LOG-OFF
5651	77	199	MOV M,A ;GUARDA CARACTER EN MEMORIA
5652	4F	200	MOV C,A ;PREPARA ECO
5653	CD09FE	201	CALL CO
5656	23	202	INX H ;PREPARA MEMORIA
		203	;
5657	14	204	INR D ;CUENTA SIGUIENTE CARACTER
5658	CDE056	205	CALL LOOP1 ;ESPERA MENSAJE DEL OPERADOR
5658	3E02	206	MVI A,STX ;CHECA SI HUBO MENSAJE
565D	B9	207	CMR C
565E	C26256	208	JNZ CNIN ;SALTA SI NO HUBO
5661	C5	209	PUSH B ;GUARDA INDICADOR SI HUBO
5662	CD03FB	210 CNIN:	CALL C1 ;ESPERA POR CARACTER DE CONSOLA
5665	E67F	211	ANI PRTY ;QUITA PARIDAD
5667	FE59	212	CPI 'Y' ;ES SEGUNDO CARACTER PARA LOG-OFF?
5669	C29B56	213	JNZ FALSE ;RESTITUYE INDICADOR SI NO ES
566C	77	214	MOV M,A ;GUARDA CARACTER EN MEMORIA
566D	4F	215	MOV C,A ;PREPARA ECO
566E	CD09FB	216	CALL CO
5671	23	217	INX H ;PREPARA MEMORIA
		218	;
5672	14	219	INR D ;CUENTA SIGUIENTE CARACTER
5673	CDE056	220	CALL LOOP1 ;ESPERA MENSAJE DEL OPERADOR
5676	3E02	221	MVI A,STX ;CHECA SI HUBO MENSAJE
5678	B9	222	CMR C
5679	C27D56	223	JNZ CONI ;SALTA SI NO HUBO
567C	C5	224	PUSH B ;GUARDA INDICADOR SI HUBO
567D	CD03FE	225 CONI:	CALL C1 ;ESPERA CARACTER DE LA CONSOLA
5680	E67F	226	ANI PRTY ;QUITA PARIDAD
5682	FE45	227	CPI 'E' ;ES TERCER CARACTER PARA LOG-OFF?
5684	C29B56	228	JNZ FALSE ;RESTITUYE INDICADOR SI NO ES
5687	77	229	MOV M,A ;GUARDA CARACTER EN LA MEMORIA
5688	4F	230	MOV C,A ;PREPARA ECO
5689	CD09FB	231	CALL CO
568C	23	232	INX H ;PREPARA MEMORIA
		233	;
568D	14	234	INR D ;CUENTA SIGUIENTE CARACTER
568E	CD03FE	235	CALL C1 ;ESPERALO
5691	E67F	236	ANI PRTY ;QUITA PARIDAD
5693	FE0D	237	CPI CR ;ES INSTRUCCION PARA EJECUTAR LOG-OFF?
5695	C29B56	238	JNZ FALSE ;RESTITUYE INDICADOR SI NO ES
5698	C29B56	239	JMP RETU ;EJECUTA LOG-OFF
		240	;
569B	1D	241 FALSE:	DCR E ;RESTITUYE INDICADOR A POSICION INICIAL
569C	C3DF55	242	JMP ANTES ;PROCESA INSTRUCCION
		243	↓

LOC	ORG	LINE	SOURCE STATEMENT
		244 ;	
		245 ;	
		246 ;	CORRECCION DE UNA LINEA
569F	0E24	247	OTRA: MVI C, '4' ; PRODUCE ECO DE ESCAPE
56A1	CD09FB	248	CALL CO
56A4	CD0256	249	CALL RETLF ; PREPARA CONSOLA
56A7	C3C555	250	JMP INI ; DEJA LA LINEA Y COMIENZA
		251 ;	
		252 ;	
		253 ;	CORRECCION DE UN CARACTER
56AA	15	254	BORA: DCR D ; RESTITUYE CUENTA DE CARACTERES
56AB	CACFB5	255	JZ NEXT ; RECESA SI NO HAY CARACTERES EN MEMORIA
56AE	28	256	DCX H ; PREPARA ECO DE CARACTER INCORRECTO
56AF	4E	257	MOV C, M ; SACALO
56B0	CD09FB	258	CALL CO
56B3	C30053	259	JMP LAST ; VE POR CARACTER CORRECTO
		260 ;	
		261 ;	
		262 ;	FIN DEL PROGRAMA
56B6	CD0556	263	FIN: CALL AVISO ; AVISA A CONSOLA DE FIN DE PROGRAMA
56B9	0E40	264	MVI C, '0'
56BB	CD09FB	265	CALL CO
56BE	CD0256	266	CALL RETLF ; PREPARA CONSOLA
56C1	CF	267	RST 1 ; CONTROL A ISIS-II
		268 ;	
		269 ;	
		270 ;	
		271 ;	
		272 ;	
56C2	0E0D	273	RETLF: MVI C, CR ; RETURN A CONSOLA
56C4	CD09FB	274	CALL CO
56C7	0E0A	275	MVI C, LF ; LINE FEED A CONSOLA
56C9	CD09FB	276	CALL CO
56CC	C9	277	RET
		278 ;	
56CD	0E07	279	AVISO: MVI C, BEL ; CAMPANA PARA CONSOLA
56CF	CD09FB	280	CALL CO
56D2	0E07	281	MVI C, BEL
56D4	CD09FB	282	CALL CO
56D7	C9	283	RET
		284 \$	EJECT

LOC	OBJ	LINE	SOURCE STATEMENT
		285	;
		286	;
		287	TELIN
		288	;FUNCION: TELIN
		289	;DESCRIPCION: TELIN ESPERA HASTA QUE UN CARACTER HA ENTRADO AL PUERTO. ENTONCES
		290	;LO DEVUELVE MEDIANTE EL REGISTRO A LA RUTINA QUE LO SOLICITA.
		291	;
56DB	D844	292	TELIN: IN TELS ;STATUS DEL PUERTO
56DA	E802	293	ANI RBR ;CHECA POR BUFFER DE RECEPCION LISTO
56DC	C8C856	294	JZ TELIN ;HAZ LOOP HASTA QUE ESTE LISTO
56DF	D845	295	IN TELI ;SI ESTA LISTO, ACEPTA CARACTER
56E1	C9	296	RET
		297	;
		298	TELOUT
		299	;
		300	;FUNCION: TEOULT
		301	;DESCRIPCION: TEOULT ESPERA HASTA QUE EL PUERTO ESTA LISTO PARA ACEPTAR UN
		302	;CARACTER. ENTONCES LO ENVIA A ESTE.
		303	;
56E2	DB44	304	TELOUT: IN TELS ;STATUS DEL PUERTO
56E4	E801	305	ANI TROY ;CHECA POR BUFFER DE TRANSMISION LISTO
56E6	C8E256	306	JZ TEOULT ;HAZ LOOP HASTA QUE ESTE LISTO
56E9	79	307	MOV A,C ;SI ESTA LISTO SACCA CARACTER
56EA	D345	308	OUT TEO ;CARACTER DE SALIDA
56EC	C9	309	RET
		310	;
		311	LOOP1
		312	;
56ED	D830	313	LOOP1: MVI B,CUENTI ;RETARDO PARA MENSAJE DE OPERADOR
56EF	05	314	DCRE: DCR B ;EFECTUA RETRASO
56F0	08	315	RZ ;REGRESA SI NO LLEGO MENSAJE
56F1	D244	316	IN TELS ;STATUS DEL PUERTO
56F3	E802	317	RBR ;BUFFER DE RECEPCION LISTO?
56F5	C8E256	318	JZ DCRE ;CONTINUA RETARDO SI NO
56F8	DCD56	319	CALL AVISO ;AVISAS SI LLEGA MENSAJE
56FB	E5	320	PUSH H ;GUARDA DIRECCION DE MEMORIA
56FC	0800	321	MVI B,CONT ;CONTADOR DE CARACTERES DE MENSAJE DE OP.
56FE	217859	322	LXI H,BUFF ;POSICION INICIAL DE MEMORIA PARA MENSAJE
5701	04	323	FOLLOW: INR B ;CUENTA CARACTER
5702	D845	324	IN TELI ;ADMITE CARACTER
5704	E87F	325	ANI PRY ;QUITAS PARIDAD
5706	77	326	MOV M,A ;GUARDA CARACTER EN MEMORIA
5707	23	327	INX H ;PREPARAS MEMORIA
5708	DB44	328	IN TELS ;STATUS DEL PUERTO
570A	E802	329	RBR ;BUFFER DE RECEPCION LISTO?
570C	D20157	330	FOLLOW JNZ ;CONTINUA SI ASI ES
570F	E1	331	POP H ;RESTITUYE POSICION DE MEMORIA
5710	0E02	332	MVI C,STX ;INDICADOR DE MENSAJE
5712	C9	333	RET
		334	EJECT



LOC	COB	LINE	SOURCE STATEMENT
		335	;
		336	;
		337	LOOP2
5713	0430	338	MUI B,CUENT1 ;RETARDO PARA MENSAJE DE CONSOLA
5715	05	339	ACR B ;EFECTUA RETRASO
5716	08	339	RZ ;REGRESA SI NO LLEGA MENSAJE
5717	08F7	340	IN CRTS ;STATUS DEL PUERTO
5719	0A02	341	ANI RCR ;SI ESTE DE RECEPCION LISTO?
5718	0A1007	342	J7 RCR ;CONTINUA RETARDO SI NO
5718	0B40	343	IN TELL ;ADMITE CARACTER
5720	0E10	344	CPI BREAK ;ES UN BREAK?
5722	00	345	RNZ ;REGRESA SI NO
5723	0F	346	MOV C,A ;ENVIAR SI ES
5724	0DE256	347	CALL TELOUT
5727	09	348	RET
		349	;
			EJECT

LOC	OBJ	LINE	SOURCE STATEMENT
		350	; EQUIVALENCIAS
		351	;
FB09		352	CO EQU 0FB09H ;SUBROUTINA DE SALIDA DE CONSOLA
FB03		353	CI EQU 0FB03H ;SUBROUTINA DE ENTRADA DE CONSOLA
0044		354	TELCTL EQU 044H ;PUERTO DE CONTROL (USART)
0044		355	TELS EQU 044H ;PUERTO DE STATUS
0045		356	TELI EQU 045H ;PUERTO DE ENTRADA DE DATOS
0045		357	TELO EQU 046H ;PUERTO DE SALIDA DE DATOS
0CF6		358	CRT1 EQU 0CF6H ;PUERTO DE ENTRADA DE DATOS (CRT)
00F7		359	CRTS EQU 0CF7H ;PUERTO DE STATUS (CRT)
00CE		360	MIFO EQU 0CEH ;"MODE INSTRUCTION FORMAT"
0035		361	CIFO EQU 035H ;"COMMAND INSTRUCTION FORMAT"
		362	;
0001		363	TRDY EQU 0000001B ;MASCARA PARA BUFFER DE TRANSMISION
0002		364	PRR EQU 0000010B ;MASCARA PARA BUFFER DE RECEPCION
		365	;
00E0		366	MASK EQU 80H ;MASCARA PARA PONER UN 1 EN EL OCTAVO BIT
007F		367	PRTY EQU 07FH ;MASCARA PARA ELIMINAR PARIDAD
		368	;
0007		369	BEL EQU 07H ;ASCII DE LA CAMPANA
001E		370	BREK EQU 18H ;ASCII DEL ESCAPE
000D		371	CR EQU 0DH ;ASCII DEL 'CARRIAGE RETURN'
007F		372	DEL EQU 7FH ;ASCII DEL 'DELETE'
0018		373	CNTLX EQU 18H ;ASCII DEL CONTROL-X
0008		374	CNTLH EQU 08H ;ASCII DEL CONTROL-H
0002		375	STX EQU 02H ;ASCII DEL 'START TEXT'
0020		376	ESPA EQU 20H ;ASCII DEL ESPACIO
000A		377	LF EQU 0AH ;ASCII DEL 'LINE FEED'
0000		378	CONT EQU 00H ;CONTADOR PARA CARACTERES
0030		379	CUENTI EQU 030H ;RETRASO PARA MENSAJE DE HP
00FF		380	MENDS EQU 0FFH ;CONTADOR INICIAL PARA LOG-OFF
000A		381	RETRN EQU 0FH ;ASCII DEL LINE FEED
		382	\$ EJECT

LOC	OBJ	LINE	SOURCE STATEMENT
		383	;
		384	;
		385	;
		386	MEM: DS 150H ;ESPACIO PARA MEMORIA
5723		387	BUF: DS 150H ;BUFFER DE MEMORIA
5878		388	ESTAK: DS 30H ;ESPACIO PARA STACK
59CB		389	APU EQU \$ ;DIRECCION INICIAL APUNTAADOR DE STACK
59F6		390	END 5555H

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS

ACA	A 5715	ANTES	A 550F	APU	A 59F8	AQUI	A 557A	ATRAS	A 5395	AVISO	A 56CD	BEL	A 0007
BORR	A 56AA	BREAK	A 001B	BUF	A 5878	CI	A FB03	CIFO	A 0035	CNIN	A 5662	CNTLH	A 0008
CNTLX	A 0018	CO	A FB09	CON1	A 567D	CONIN	A 55DA	CONT	A 0000	CR	A 000D	CRT1	A 00F6
CRTS	A 00F7	CUENT1	A 0030	DECRE	A 56EF	DEL	A 007F	ESPA	A 0020	ESTAK	A 59CB	FALSE	A 5698
FIN	A 5686	FOLLOW	A 5701	INI	A 55C5	LAST	A 55D0	LF	A 000A	LOOP	A 5613	LODP1	A 56ED
LOOP2	A 5713	MASK	A 0080	MEM	A 5728	MENOS	A 00FF	MESSG	A 5630	MIFO	A 00CE	MODE	A 5628
NEXT	A 55CF	OTRA	A 569F	OTRO	A 563E	PRTY	A 007F	R8R	A 0002	RETLF	A 56C2	RETRN	A 000A
RETU	A 5606	SIGUE	A 5572	START	A 558C	STX	A 0002	TELCTL	A 0044	TELL	A 0043	TELIN	A 56D8
TELO	A 0045	TELOUT	A 56E2	TELS	A 0044	TERMI	A 564A	TRDY	A 0001	UNO	A 5650		

ASSEMBLY COMPLETE, NO ERRORS

Configuración del puerto de la computadora  
HP 3000.

SYSTEM-ID = HP320028.00.02

MEMORY SIZE (K WORDS) 256  
 LINKED MEMORY (WORDS) 218128  
 FIXED MEMORY (WORDS) 44016


HIGHEST DRT# 25

HIGHEST LOEV# 72

LOG DEV #	DRT #	U	C	T	SUB	TERM	REC	OUTPUT	MODE	DRIVER	DEVICE
#	#	I	A	H	Y	TYPE	WIDTH	DEV		NAME	CLASSES
		T	P	E							
1	4	0	0	0	9		128	0		*IOMDISC1	DISC SPOOL SYSDISC
2	4	1	0	0	9		128	0		IOMDISC1	DISC SPOOL
5	13	0	0	8	0		40	LP	JA	S IOCDR00	CARD
6	14	0	0	32	2		66	0		S IOLPRT0	LP
7	6	0	0	24	0		128	0		IOTAPE0	TAPE
8	6	1	0	24	0		128	0		IOTAPE0	TAPE
9	6	2	0	24	0		128	0		IOTAPE0	TAPE
10	6	3	0	24	0		128	LP	JA	IOTAPE0	JOBTAPE
12	18	0	0	18	0		0	0		CSSBSC0	SSLC1
13	19	0	0	18	0		0	0		CSSBSC0	SSLC2
20	7	0	0	16	0	10	40	20	JAID	IOTERM0	TERM
21	7	1	0	16	0	10	40	21	JAID	IOTERM0	TERM
22	7	2	0	16	0	10	40	22	JAID	IOTERM0	TERM
23	7	3	0	16	0	10	40	23	JAID	IOTERM0	TERM
24	7	4	0	16	0	10	40	24	JAID	IOTERM0	TERM
25	7	5	0	16	0	10	40	25	JAID	IOTERM0	TERM
26	7	6	0	16	0	10	40	26	JAID	IOTERM0	TERM
27	7	7	0	16	0	10	40	27	JAID	IOTERM0	TERM
28	7	8	0	16	0	10	40	28	JAID	IOTERM0	TERM
29	7	9	0	16	0	??	100	29		IOTERM0	PLOT
30	7	10	0	16	0	10	40	30	JAID	IOTERM0	TERM
31	7	11	0	16	0	10	40	31	JAID	IOTERM0	TERM
32	7	12	0	16	1	11	40	32	JAID	IOTERM0	TERM
33	7	13	0	16	0	0	66	33	JAID	IOTERM0	TERM
34	7	14	0	16	0	10	40	34	JAID	IOTERM0	TERM
35	7	15	0	16	0	10	40	35	JAID	IOTERM0	TERM
50#	12	0	0	41	0		128	0	I	I0DS0	SDS1
51#	13	0	0	41	0		128	0	I	I0DS0	SDS2
60#	12	0	0	16	0	??	40	60	J ID	I0DSTRM0	DSTERM
61#	12	1	0	16	0	??	40	61	J ID	I0DSTRM0	DSTERM
62#	12	2	0	16	0	??	40	62	J ID	I0DSTRM0	DSTERM
70#	13	0	0	16	0	??	40	70	J ID	I0DSTRM0	DSTERM
71#	13	1	0	16	0	??	40	71	J ID	I0DSTRM0	DSTERM
72#	13	2	0	16	0	??	40	72	J ID	I0DSTRM0	DSTERM

LDN	PH	PRT	LCL MOD	TC	RCV TMOUT	LCL TMOUT	CON TMOUT	MODE	TRANSMIT SPEED	TH	BUFFER SIZE	D C	DRIVER OPTIONS
12	0	1	1	1	20	60	180		240	0	576	N	0
13	0	1	1	1	20	60	180		240	0	576	N	0

CLASS NAME	ACCESS TYPE	LOGICAL DEVICES
CARD	IN	5
DISC	DA	1,2
DSTERM	I/O,C	60,61,62,70,71,72
JOBTAPE	I/O,NC	10
LP	OUT	6
PLOT	I/O,C	29
SDS1	41	50
SDS2	41	51
SPOOL	DA	1,2
SSLC1	18	12
SSLC2	18	13
SYSDISC	DA	1
TAPE	I/O,NC	7,8,9
TERM	I/O,C	20,21,22,23,24,25,26,27,28,31,34,35,30,33,32



TIME LIMIT (45)	150	STACK SIZE	4000
TPRI =	152	MAX STACK SIZE	31252
CPRI =	160	MAX XOS SIZE	32767
DPRI =	200	MAX NUMBER XOS	9
JOB LIMIT	4	MAX CODE SEG SIZE	16384
SESSION LIMIT	10	MAX # CODE SEGS	63
SECS TO LOGON	120		

LOG REC SIZE	2	SPOOL EXTENT SIZE:	768
LOG FILE SIZE	1023	MAX SPOOL SECTORS	256
LOG FILE #	462	MAX OPEN SPODFLES	60

TYPE	EVENT	STATUS
------	-------	--------

1	LOGGING ENABLED	ON
2	JOB INITIATION	ON
3	JOB TERMINATION	ON
4	PROCESS TERMINATION	ON
5	FILE CLOSE	ON
6	SYSTEM SHUTDOWN	OFF
7	POWER FAIL	OFF
8	SPOOLING	ON
9	LINE DISCONNECTION	OFF
10	LINE CLOSE	OFF
11	I/O ERROR	OFF
12	VOLUME MOUNT	OFF
13	VOLUME SET MOUNT	OFF
14	TAPE LABELS	OFF

#CST ENTRIES	192	TERM BUFFERS	150
#DST ENTRIES	300	SYSTEM BUFFERS	40
#PCB ENTRIES	96	ICS	512
#IDQ ENTRIES	144	UCDP REQ QDE	32
#XCST ENTRIES	400	HAN TABLE SIZE	768
BREAKPOINT ENTRIES	32	TIMER REQ LIST	32
LOCAL RINS	60	VIRTUAL SECTIONS	24576
GLOBAL RINS	48	DIRECTORY SECTIONS	1536



Figura		página
1.1	Los dos sistemas de computación existentes en la Universidad Autónoma Metropolitana comunicándose entre sí mediante una interfaz.	2
1.2	La interfaz como periférico propio de cada sistema.	4
1.3	Sistema estándar Intellec MDS.	8
1.4	Periféricos del Intellec MDS.	9
1.5	El sistema operativo y los componentes principales del sistema HP 3000.	14
1.6	Puertos de salida en la computadora HP 3000 y en la microcomputadora Intellec MDS.	19
2.1	Transferencia de datos en paralelo.	21
2.2	Formato de un caracter de datos asíncrono.	25
2.3	Formato de un caracter de datos síncrono.	28
2.4	Interfaz entre una terminal y un equipo de comunicación de datos (modem).	34
2.5	El modem recibe la información de cada equipo y la convierte para hacerla compatible con las facilidades de transmisión.	40
2.6	Conexión mediante modems entre dos equipos de procesamiento de datos.	43
2.7	Conexión sin modems entre una computadora y una terminal asíncrona.	44
3.1	Diagrama de bloques de un sistema de computación básico.	47
3.2	El bloque del CPU en detalle.	50
3.3	Contactos de entrada, salida y alimentación del CPU 8080.	52
3.4	Arquitectura del microprocesador 8080.	56

Figura		página
3.5	Ejemplo de un sistema típico de interfaz de entrada/salida.	58
3.6	Contactos del Transmisor/Receptor Universal Síncrono/Asíncrono 8251 de Intel.	61
3.7	Diagrama de bloques de la configuración interna del USART 8251.	63
3.8	Diagrama de bloques del "buffer" de entrada/salida.	66
3.9	Contador binario de 4 bits SN7493A.	79
3.10	Configuración lógica del 7493A.	80
3.11	"Latch" de alta velocidad 3404.	81
3.12	Base de tiempo para el USART.	82
3.13	Impulsor de línea bidireccional de 4 bits en paralelo 8226.	85
3.14	Diagrama lógico del 8226.	86
3.15	El 8226 como "buffer" entre el canal de datos del sistema y el canal de datos externo del USART.	87
3.16	Decodificador binario 3205.	89
3.17	Diagrama lógico y tabla de verdad del decodificador binario 3205.	90
3.18	Conexión de dos decodificadores en cascada.	92
3.19	Circuito integrado 8T15.	94
3.20	Circuito integrado 8T16.	96
3.21	Acoplamiento de las señales de interfaz a niveles de voltaje compatibles con las características EIA, mediante impulsores y receptores de línea.	98
3.22	Distribución aproximada de componentes en la tarjeta.	102

Figura	página
3.23 Diagrama general de conexiones de la tarjeta de interfaz.	103
3.24 Diagrama de flujo de la subrutina de inicialización del USART.	111
3.25 Diagrama de flujo de la subrutina que transmite datos del sistema al exterior.	112
3.26 Diagrama de flujo de la subrutina que toma del USART los datos que llegan a este último provenientes del exterior.	113
4.1 Cable de interfaz para el modem de la computadora HP 3000.	115
4.2 Conexiones de la interfaz del lado de la computadora HP 3000.	116
4.3 Cable de interfaz de la microcomputadora.	118
4.4 Conexiones de interfaz del lado de la microcomputadora.	119
4.5 Diagrama de flujo del programa de control de la interfaz.	122 123 124 125 126 127

- |   |  |
|---|--|
| A Beginner's Guide to Computers<br>& Microprocessors              | Charles K. Adams   |
| An Introduction to Microcomputers<br>Volume 0 The Beginner's Book | Adam Osborne   |
| Computer Dictionary   | Charles J. Sippl &<br>Charles P. Sippl                         |
| Computer Logic Design   | M. Morris Mano   |
| Cuaderno de Trabajo   | Lance A. Leventhal   |
| Customer Training (General<br>Information Manual)                 | Hewlett Packard  |
| Data Catalog 1977   | Intel  |
| Digital Integrated Electronics                                    | Herbert Taub &<br>Donald Schilling                             |
| EDIT/3000 Reference Manual  | Hewlett Packard  |
| EIA Standard  | Electronic Industries<br>Association<br>Engineering Department |
| HP 3000 Computer Systems Utilities                                | Hewlett Packard  |
| Intellec MDS Monitor Listing V. 2.0                               | Intel  |
| ISIS-II System User's Guide                                       | Intel  |
| MDS-8080 INTELLEC MDS<br>Operator's Manual                        | Intel  |

Microcomputer Systems User's Manual	Intel
Signetics Digital, Linear and Mos Applications	Signetics Corporation
Technical Aspects of Data Communications	John E. McNamara
The TTL Data Book for Design Engineers	Texas Instruments
Using the 8251 Universal Synchronous/Asynchronous Receiver/transmitter	Lionel Smith (Intel)
Using the HP 3000	Hewlett Packard
2604B/N/S Display Terminal Reference Manual	Hewlett Packard
8080/8085 Assembly Language Programming Manual	Intel