3

Interfaces serie y paralelo

3.1 INTRODUCCIÓN

La transferencia de información entre dos sistemas digitales, por ejemplo, un microcornputador y un terminal, periférico u otro microcomputador, se realiza generalmente carácter a carácter utilizando códigos binarios (ASCII, EBCDIC, BAUDOT, CDC-Científico, ...). Otras veces la información que se transfiere no corresponde a ninguna codificación de caracteres numéricos ó alfanurnéricos sino que es puramente binaria, por ejemplo, cuando se efectúan cargas de programas objeto sobre la memoria del ordenador.

De una forma o de otra la información se transmite en unidades de información denominadas palabras, que suelen ser de 8,16 o 32 bits. Existen dos formas de realizar la transmisión de estas palabras:

- Método paralelo: Transmitiendo simultáneamente, por líneas separadas, todos los bits de la palabra, junto con una señal de reloj que indica el momento en que está presente una palabra de información en las líneas de datos.
- Método serie: Transmitiendo en forma secuencial en el tiempo todos los bits de la palabra, uno tras otro, por una sola línea de datos.

Eventualmente puede existir una línea adicional de reloj que marca los tiempos de bit. El método paralelo es utilizado para transmisiones a alta velocidad entre dos sistemas; no obstante cuando la distancia entre ambos aumenta, el coste de la línea y el de los amplificadores de transmisión y recepción puede llegar a crecer de forma tal que, desde el punto de vista económico, sea preferible utilizar un sistema serie de comunicaciones. Por otra parte, los sistemas de comunicaciones serie han alcanzado un alto grado de estandarización desde hace tiempo. Existen normas universalmente aceptadas que fijan completamente todos los detalles de la comunicación, incluyendo aspectos mecánicos (tipo de conector y distribución de señales en las patillas del mismo), aspectos eléctricos (niveles y formas de las señales) y aspectos lógicos (sistemas de codificación y sincronización, y descripción de todos los circuitos de datos, control y temporizado).

Estos estándares han conducido a que la mayoría de fabricantes de procesadores y periféricos incorporen en sus equipos interfaces serie que cumplen las normas especificadas, de forma que se pueda realizar con toda facilidad la conexión indistinta de cualquier terminal o

periférico con cualquier procesador. Así, se utilizan interfaces serie para conectar periféricos, como terminales de pantalla o impresoras, a computadores aunque su distancia sea reducida y puedan, por tanto, usarse interfaces de tipo paralelo.

Las normas referidas a las interfaces paralelas, han aparecido más recientemente ya que durante más tiempo han permanecido como interfaces propietarias de los distintos fabricantes. Actualmente el nivel de normalización de éstos los situan a la altura de las interfaces serie.

Por último, un tercer campo en que se utilizan sistemas de manipulación de datos en serie es el de los controladores de unidades de almacenamiento de informaciones digitales sobre soportes magnéticos (discos, cassettes y diskettes). En ellos se graban y se leen los datos en forma serie, presentándose problemas de codificación comunes con los sistemas de comunicaciones serie. Recuérdese como en el tema dedicado a dispositivos de almacenamiento, los datos son almacenados en serie, por lo que aunque el dispositivo se conecte al sistema central mediante una interfaz paralela, el canal de lectura/escritura trabaja siempre en modo serie.

3.2 PROBLEMAS EN LAS TRANSMISIONES SERIE

Cuando se transmiten informaciones a través de una línea serie es necesario utilizar un sistema de codificación que permita resolver los siguientes problemas:

a) Sincronización de bit
b) Sincronización de carácter
c) Sincronización de mensaje

3.2.1 Sincronización de bit

El receptor necesita saber exactamente donde empieza y donde termina cada bit en la señal recibida para efectuar el muestreo de la misma en el centro de la celda de bit. Considérese el caso de transmisión en serie de la información 01110010. Si se utilizase un método NRZ como el explicado en el tema dedicado a dispositivos de almacenamiento (no retorno a cero, en que los bits 1 ó 0 se representan por niveles 1 ó 0 respectivamente), la señal en la línea sería como la representada en la figura (3.1).

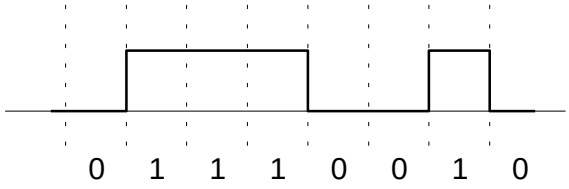


Fig. 3.1 Codificación NRZ

La presencia de varios bits iguales, por ejemplo 3 «unos», hace que la línea no efectúe ninguna transición y el receptor puede perder la referencia de donde empieza y donde acaba cada bit. Si el número de bits iguales aumenta, se observa que la dificultad de reconstruir las celdas de bit aumenta también.

Para resolver el problema de la sincronización de bit pueden usarse varios métodos:

a) Enviar por una línea independiente de la de datos una señal de reloj que indique el centro o el inicio de las celdas de bits de la línea de datos.
b) Enviar junto con cada bit transmitido en serie y por la misma línea una información adicional que permita al receptor extraer una señal de reloj. Esto se puede conseguir fundamentalmente

de dos formas: mediante códigos autoreloj o mediante pulsos de sincronismo como en la señal de vieo compuesto.

c) Lograr, mediante algún procedimiento, que los relojes de transmisión y recepción se mantengan en fase continuamente.

Para ilustrar con mayor claridad estos conceptos veamos un proceso de transmisión y recepción con algún detalle:

- Se desea transmitir 8 bits en serie, por ejemplo 01110010.
- El transmisor, en cada flanco de subida de un reloj, envía un bit de información por la línea.
- Las señales por la línea, a lo largo del tiempo, contienen unas celdas de bit, que dependen de la frecuencia del reloj, y dentro de cada celda, el transmisor coloca un bit de información codificado según algún procedimiento, por ejemplo:

  - RZ. Una celda de bit contiene un 1 si hay un impulso positivo y un 0 si no lo hay.
  - NRZ. La celda de bit contiene un 1 (o un 0) según que el nivel de la señal sea 1 (ó 0).
  - NRZI. La celda de bit contiene un 1 si hay una transición y un 0 si no la hay.

Para que el receptor pueda interpretar adecuadamente estas señales, debe ser capaz de obtener o crear un reloj que se mantenga en perfecto sincronismo con el del transmisor. Este reloj marcará las celdas de bit y analizándolas verá si contienen un bit 1 ó 0. En este caso los datos no contienen información de reloj. Efectivamente, las secuencias de ceros, en cualquiera de los sistemas (RZ, NRZ, NRZI), y las secuencias de unos, en el sistema NRZ, no contienen ninguna transición que permita al receptor determinar la situación de las celdas de bit.

Estos sistemas se dice que no son auto-reloj y plantean el mismo problema que en los dispositivos de almacenamiento. La sincronización de bit en tales sistemas se consigue utilizando en la recepción el propio reloj de transmisión, enviado por una línea independiente de los datos, o bien utilizando relojes de precisión y con dispositivos adicionales que aseguren que se mantiene a la misma frecuencia y fase que el de transmisión. Alguno de estos sistemas de reloj se describe más adelante.

Frente a estos sistemas de codificación se encuentran los de auto-reloj (self-clock), que transmiten información de forma tal que permiten al receptor deducir la situación exacta de las celdas de bit y por tanto los datos, sin necesidad de disponer de un reloj síncrono con el de transmisión.

Hay varios métodos auto-reloj, siendo los más conocidos: PE, codificación de fase; FSK, codificación por cambio de frecuencia; FM, modulación de frecuencia; o MFM, modulación de frecuencia modificada. En estos sistemas, el envío de la información adicional para determinación del reloj se hace a costa de la disminución de la cantidad de información útil enviada para un mismo ancho de banda. Aunque como se vió en el tema de dispositivos de almacenamiento el método MFM no requiere aumentar el ancho de banda.

Dado que las características de una línea o canal de transmisión limitan la frecuencia máxima de la señal que se puede enviar por él, la cantidad que es posible enviar mediante una codificación «no auto-reloj» es normalmente mayor que mediante una codificación «auto-reloj». No obstante hay campos de aplicación idóneos para cada método.

Sin embargo, cuando el problema es de transmisión de una información serie entre dos puntos, es posible la utilización de una codificación «no auto-reloj», realizando la sincronización de bit con el propio reloj de transmisión o generando un reloj sincronizado con aquél.

Cuando el problema es de grabación de información serie en un soporte magnético giratorio (discos, cintas, etc.), para posterior reproducción o lectura, la posibilidad de utilizar el reloj usado

en la grabación, o sincronizar un reloj de recepción, se hace muy difícil al introducirse un agente perturbador como es el de las fluctuaciones de las velocidades de giro del soporte magnético en los instantes de grabación y lectura.

En tales casos es preferible utilizar un método auto-reloj y disponer en el receptor un circuito que extraiga el reloj de recepción de los datos, haciéndose insensible a posibles derivas del reloj de grabación y de las velocidades del soporte.

3.2.2 Sincronización de carácter

La información en serie se transmite, por definición, bit a bit, pero la misma tiene sentido en palabras, por ejemplo de 8 bits. El sistema de codificación usado debe permitir distinguir sin ambigüedades dentro de una corriente de bits cuáles son los 8 bits que forman una palabra. Normalmente se resuelve enviando los bits de cada carácter separados por alguna señal de sincronismo.

Por ejemplo, la siguiente información en serie: 0100110001001100100
Puede tener distintas interpretaciones según como se agrupen los 8 bits para formar las palabras.
01, 00110001, 00110010, 0 o también 010, 01100010, 01100100.

La primera agrupación representa los caracteres 1 y 2 según una codificación ASCII, la segunda representa, según la misma codificación, los caracteres b y d.

Para obtener la sincronización de carácter pueden utilizarse diversos sistemas, unos se basan en la utilización de líneas adicionales a las de datos para enviar impulsos que indican el inicio de un bloque de caracteres. Tal impulso identifica el primer bit del primer carácter de un bloque o mensaje, y luego, por contaje de bits y caracteres se determinan todas las fronteras de los datos del bloque.

Otros sistemas, utilizados usualmente en los sistemas de comunicaciones serie, son:

Asíncrono: Cada carácter va señalizado mediante dos bits, uno al principio, bit de arranque, y otro al final, bit de parada. Estos bits permiten reconocer las fronteras de los caracteres.

Síncrono: Cada mensaje o bloque de transmisión va precedido por unos caracteres de sincronismo. Cuando el receptor identifica una configuración de bits igual a la de los caracteres de sincronismo, da por detectado el inicio de los datos y a continuación, por contaje de bits y caracteres identifica todos los caracteres del bloque.

3.2.3 Sincronización de mensaje

En un sistema de comunicaciones generalmente las informaciones se transmiten en bloques de caracteres. Por sincronización de mensaje entendemos el mecanismo por el cual un conjunto de palabras es interpretado correctamente. Este problema normalmente no incumbe a los circuitos de codificación, sino al procesador que lo utiliza. El conjunto de reglas (protocolo) que permiten interpretar correctamente los mensajes suele estar controlado por una tarea software (un programa) que ejecuta el ordenador, aunque actualmente hay ciertos circuitos integrados LSI que efectúan alguna de estas tareas.

3.3 MÉTODOS DE E/S PARA COMUNICACIONES SERIE

En lo que sigue se utiliza el nombre genérico de terminales para designar a los sistemas que se comunican utilizando un procedimiento serie de entrada/salida. Un terminal puede ser un ordenador, un microcomputador, un periférico, etc. La comunicación entre terminales se hace utilizando líneas o canales de transmisión, que pueden ser:

Simplex: cuando son capaces de transmitir información en un solo sentido.

Semiduplex (half-duplex): cuando son capaces de transmitir información en ambos sentidos pero no de forma simultánea.

Dúplex (full-duplex): cuando son capaces de transmitir simultáneamente información en ambos sentidos.

La codificación de las señales en estos sistemas se hace mediante uno de los siguientes métodos: asíncrono o síncrono.

3.3.1 Método asíncrono

En el método asíncrono la transmisión se controla por bits de inicio y de final que enmarcan cada carácter transmitido, son los denominados bits de inicio ('start') y parada ('stop') y son utilizados por el terminal receptor para sincronizar su reloj con el del transmisor en cada carácter. La especificación RS404 de EIA (Electronic Industries Association) define las características del método asíncrono de transmisión serie. La transmisión en asíncrono se basa en las siguientes reglas:

a) Cuando no se envían datos por la línea, ésta se mantiene en estado 1.
b) Cuando se desea transmitir un carácter se envía primero un bit de inicio, que pone la línea a cero durante el tiempo de 1 bit.
c) A continuación se envían todos los bits del carácter a transmitir con los intervalos que marca el reloj de transmisión.
d) A continuación del último bit del carácter se envía el bit de final que hace que la línea se ponga a 1 por lo menos durante el tiempo de 1 bit.

Los datos codificados según estas reglas pueden ser detectados fácilmente por el receptor. Para ello deben seguirse los siguientes pasos:

1) Esperar una transición de 1 a 0 en la señal recibida.
2) Activar un reloj de frecuencia igual a la del transmisor.
3) Muestrear la señal recibida al ritmo de este reloj para formar el carácter.
4) Leer un bit más de la línea y comprobar si es 1 para confirmar que no ha habido error de sincronización.

El bit de final tiene la misión de llevar la línea a estado 1 para que el bit de inicio del próximo carácter provoque la transición de 1 a 0 que permita al receptor sincronizar el siguiente carácter. El bit de final sirve también para dar tiempo a que el sistema receptor acepte el dato recibido. De todas formas, actualmente se utiliza siempre una memoria de tipo FIFO que almacena el dato recibido mientras el receptor está recibiendo el siguiente, de forma que el procesador dispone del tiempo de todo un carácter para recogerlo.

El método asíncrono de transmisión presenta las siguientes ventajas:

1) Permite enviar caracteres a ritmos variables ya que cada uno de ellos lleva incorporada la información de sincronismo.
2) Existen circuitos integrados de bajo costo, las UART (Universal Asynchronous Receiver Transmiter), que simplifican enormemente la realización de sistemas de entrada/salida en este formato.
3) Es un método de comunicaciones estándar entre ordenadores y terminales de pantalla, impresoras lentas, ratones, modems, etc.

Entre sus inconvenientes se puede citar, como más importante, su ineficiencia, ya que cada carácter va lastrado con dos bits de sincronización que no contienen información útil. Suponiendo caracteres de 8 bits, es necesario enviar por la línea 10 bits para enviar un carácter, es decir sólo un 80% de la información transmitida es válida.

3.3.2 Método síncrono

En el método síncrono, en vez de añadirse bits de sincronismo a cada palabra, lo que se, hace es añadir caracteres de sincronismo a cada bloque de datos. Los caracteres se transmiten en serie, bit a bit, y sin ninguna separación entre uno y otro, no obstante, delante de cada bloque de datos se colocan unos caracteres de sincronismo que sirven al receptor para realizar la sincronización de carácter, es decir, conocer las fronteras de carácter en una corriente de bits. La sincronización de bit se consigue normalmente utilizando una señal externa de reloj. En una comunicación local entre dos dispositivos, el transmisor envía por una línea independiente de la de datos su señal de reloj, que es utilizada por el receptor como reloj de recepción. La sincronización de bit queda de esta forma resuelta, ya que el mismo reloj que el transmisor utiliza para serializar los bits de información sobre la línea de datos, es utilizada por el receptor para leer los datos recibidos. Será necesario únicamente tener en cuenta que el receptor debe muestrear la línea de datos con el flanco de reloj contrario al que el transmisor utilizó para enviarlos, para que así el muestreo se efectúe en el centro de la celda de bit.

El método de comunicaciones síncrono se utiliza cuando el volumen de información a enviar es importante, debido a su mayor eficiencia respecto al método asíncrono. Como ya se ha comentado, en modo asíncrono cada palabra se envía precedida por un bit de inicio y seguida por 1 ó 2 bits de final. Suponiendo palabras de 8 bits y utilización de 1 bit de final, se necesitan 10 bits para enviar una palabra de 8 bits. En modo síncrono, cada mensaje se envía precedido por unos caracteres de sincronismo, normalmente dos caracteres SYN (ASCII Nº 22).

Para enviar un mensaje de N palabras serán necesarios (N + 2) x 8 bits en síncrono y 10xN bits en asíncrono. Comparando ambas cifras se observa que para mensajes superiores a 8 bits el sistema síncrono es más eficiente, y para mensajes de 512 octetos la eficiencia del método síncrono es un 25 % superior a la del método asíncrono.

3.3.3 Regeneración del reloj en el receptor

En una comunicación remota utilizando modems, la señal de reloj es extraída del canal de datos por el modem; para ello utiliza un reloj de la misma frecuencia que el transmisor y que mediante circuitos de sincronización lo mantiene en la misma fase. El sistema es inherente al principio de funcionamiento del módem. Existen los llamados modems síncronos y modems asíncronos. Los modems asíncronos utilizan sistemas de codificación FSK cuya misión es generar una señal de distinta frecuencia para la marca "1" y el espacio "0", esta señal debe ser de frecuencia apropiada para que pueda transmitiese a través de la red telefónica, ya que ésta sólo permite la banda de audiofrecuencia. El módem receptor recibe la señal de la línea telefónica y discrimina los dos tonos generando las señales marca y espacio que reconstruyen la señal digital primitiva. Debido a este modo de funcionamiento, el módem en sí no está ligado a la frecuencia de

transmisión de los datos y admite, sin necesidad de ningún ajuste, señales de frecuencias de transmisión comprendidas entre cero y el máximo.

Otra forma de sincronizar los dos sistemas es el emplear en el receptor un oscilador de una frecuencia varias veces superior (normalmente x16) y que es el método que se emplea por ejemplo en los circuitos que incorporan los PC's para los puertos serie. Según este método, el receptor genera una señal de frecuencia 16 veces superior a la empleada por el emisor para enviar los datos y toma la muestra en el centro, es decir cuando la señal de recepción haya completado 8 ciclos, tal y como se muestra en la figura (3.2). Los flancos de la señal recibida se emplean para resincronizar el oscilador local de recepción. Esto es necesario, porque no se puede garantizar que el oscilador de recepción sea exactamente 16 veces el de emisión. Si existe alguna pequeña deriva, y por ejemplo el oscilador de recepción va ligeramente más lento de lo que debería, el siguiente flanco de señal llega antes de que complete la cuenta de 16 ciclos. Si esto sucede, la cuenta vuelve a comenzar desde cero (aunque no hubiese completado la anterior). Esto se puede conseguir con el circuito que se muestra en la figura (3.3) donde se utiliza un contador que reinicia la cuenta cada vez que llega un flanco en la señal de datos.

El objetivo es conseguir que el reloj del receptor esté en fase lo más exactamente posible con el reloj del transmisor y para ello se aprovecha cada flanco de la señal recibida para reiniciar el ciclo del reloj de recepción. La salida de peso más alto del contador de 4 bits es el reloj de frecuencia *f*. Esta señal tiene el flanco de subida en el momento en que el contador se pone a 0. La sincronización se efectúa mediante la puesta a 0 asíncrona del contador cada vez que se detecta un flanco en la línea de datos. En el momento del flanco, el contador debe estar a 0; si no lo está, significa que el reloj tiende a adelantarse o atrasarse, y se aprovecha este momento para ponerlo en sincronismo.

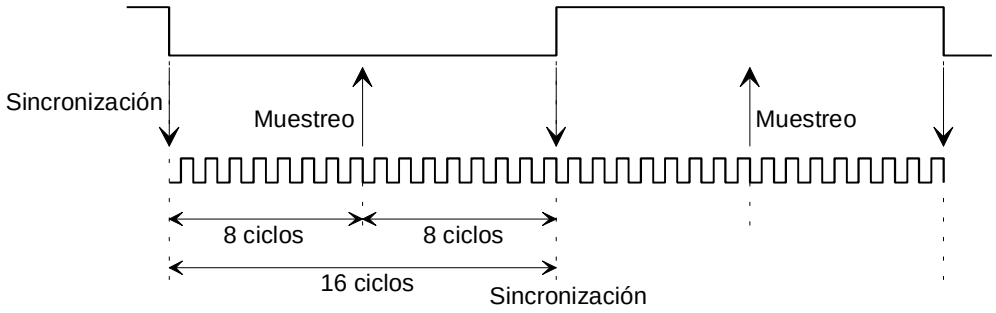


Fig. 3.2 Método de sincronizar emisor y receptor con un reloj de recepción 16 veces más rápido que el de emisión.

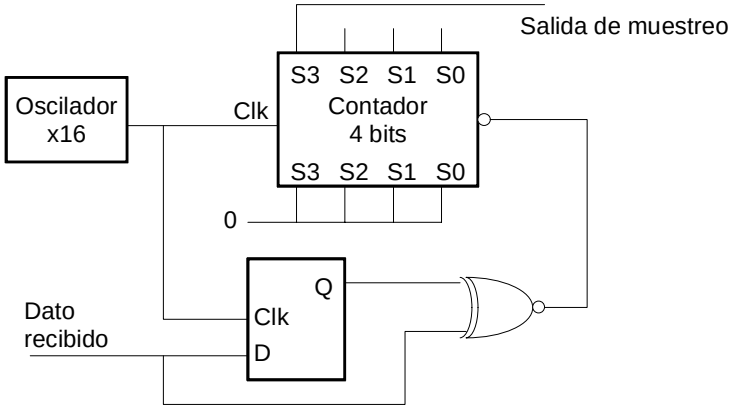


Fig. 3.3 Circuito de sincronización del reloj de recepción.

El sistema de detección de flancos utiliza una báscula D para generar una salida retrasada 1/16 de bit respecto a la señal de entrada, la puerta NOR exclusiva genera un impulso de 0 en los instantes en que la señal D varía.

Otro sistema es el basado en un lazo de sincronización de fase digital (DPLL: Digital Phase Locked Loop). Por ejemplo, el 8273, controlador de comunicaciones para protocolos tipo HDLC y SDIC, incorpora un PLL digital que funciona de la siguiente forma:

Se utiliza un reloj de frecuencia 32 veces superior a la de transmisión. A partir de este reloj 32 x R y del reloj recibido, el DPLL genera un impulso que está centrado en las celdas de bit. El DPLL reacciona contra derivas y distorsiones de fase en los datos recibidos haciendo correcciones en la fase del impulso de reloj a base de incrementos discretos. El impulso del DPLL se genera a partir de un contaje de 32 impulsos del reloj 32 x R, con una corrección de -2, -1, +1 ó +2 según el cuadrante en que se detecta el flanco de la señal recibida. En el ejemplo de la figura (3.4) el flanco de la señal se detecta en el 2º cuadrante indicando que el impulso no estaba en el centro de la celda de bit sino marcadamente desfasado hacia la derecha; por tanto el sistema correctivo actúa haciendo que el próximo impulso en vez de salir 32 tiempos de 32 x R más tarde, salga sólo 31 tiempos más tarde, iniciándose el proceso de centraje del impulso en el centro de la celda de bit. Puede demostrarse que, partiendo de una señal recibida en reposo, el DPLL tarda 12 tiempos de bit en sincronizarse en el peor caso. Para ello es necesario enviar antes de cada bloque unos caracteres que posean suficiente número de flancos para que el DPLL pueda conseguir la sincronización de bit de forma rápida.

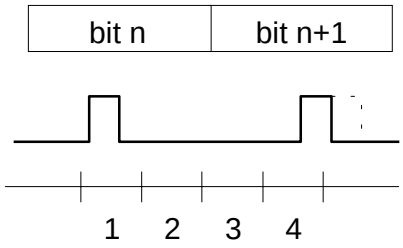


Fig. 3.4 Resincronización utilizando un PLL digital

3.4 ESTÁNDAR DE COMUNICACIÓN SERIE RS-232

Los sistemas de comunicaciones serie tienen a su disposición un conjunto de recomendaciones elaboradas por asociaciones e institutos de normalización (ISO, EIA, CCITT, etc.) que especifican con precisión todas las características del sistema de comunicaciones. Las normas para comunicaciones serie están clasificadas por niveles; aquí interesa resaltar únicamente el NIVEL 1, que hace referencia a:

a) Las características eléctricas de las señales
b) Las características mecánicas de la interfaz
c) La descripción funcional de las señales.

La norma más ampliamente aceptada es la EIA RS-232-C, que define las características funcionales, eléctricas y mecánicas de la interfaz entre un ordenador o terminal y un equipo de comunicaciones (por ejemplo un módem). La norma RS-232-C puede ser aplicada a la conexión entre dos ordenadores, aunque no se utilicen modems, como se verá a continuación. Las especificaciones funcionales de la RS-232-C coinciden con la recomendación V.24 del CCITT

(Comité Consultatif Internacional Telephonique et Telegraphique) y definen 21 circuitos con el significado que se muestra en la tabla (3.1). (Se utiliza la numeración de circuitos según CCITT):

Estas son 21 señales que RS-232-C y V.24 especifican para la comunicación entre un terminal y un módem. Para la comunicación entre dos terminales, sin utilización de modems, se utiliza un subconjunto de 3, 5 ó 7 señales solamente, aunque se respetan sus especificaciones funcionales, eléctricas y mecánicas. A este tipo de conexión se le denomina modem nulo para resaltar la ausencia de éste. En cuanto a especificaciones mecánicas, la norma RS-232-C establece un conector de 25 patillas y fija todas sus dimensiones, así como la distribución sobre el mismo de las 21 señales (Tabla 3.1) Algunos sistemas utilizan un subconjunto de estas señales y emplean un conector de 9 contactos.

Pin	Abrev.	Función
1		Tierra de protección
2	TxD	Transmisión desde el terminal
3	RxD	Recepción desde el módem
4	RTS	Petición de envío
5	CTS	Listo para enviar
6	DSR	Dato preparado
7		Masa de señal común
8	DCD	Detector de portadora
9		Reservado
10		Reservado
11		Sin asignar
12		Detector secundario de portadora
13		Listo para enviar secundario
14		Transmisión de datos secundario
15		Reloj de transmisión desde el módem
16		Recepción de datos secundario
17		Reloj de recepción
18		Sin asignar
19		Petición de envío secundario
20	DTR	Terminal de datos preparado
21		Detector de calidad de señal
22	RI	Indicador de llamada (Timbre)
23	DSRD	Selector de velocidad de la señal de datos
24		Reloj de transmisión desde el terminal
25		Sin asignar

Tabla 3.1 Circuitos especificados por CCITT en la recomendación V.24

Uno de los elementos principales que hay que fijar para cualquier comunicación son los niveles eléctricos de la señal, si emplea lógica positiva o negativa, si la transferencia es por niveles de tensión o de corriente, si es en modo diferencial o no, etc. En las comunicaciones serie, los niveles más habituales son los TTL y los RS-232.

Señales TTL: Los niveles de tensión más inmediatos para la transmisión de señales son los correspondientes a las señales TTL ya que son los que habitualmente emplean internamente los distintos sistemas. Sin embargo no son adecuados para transmisión de datos por diversos motivos y deben emplearse otras soluciones alternativas. La comunicación basada en señales TTL está basada en el envío directo por una línea unifilar o por pares trenzados de las señales de salida de las puertas TTL. No es aconsejable su utilización para distancias mayores de 5 metros lo cual restringe considerablemente su rango de aplicación. La figura (3.5) muestra los niveles de tensión

correspondientes a estas señales. Una puerta TTL envía un '1' lógico poniendo en su salida una tensión entre 2.4 y 5 Voltios. y envía un '0' lógico poniendo en su salida una tensión inferior a 0.4 Voltios. Por otra parte, una puerta TTL interpreta la entrada como '1' lógico cuando la tensión de entrada es superior a 2 Voltios y un '0' lógico cuando la entrada es inferior a 0.8 Voltios.

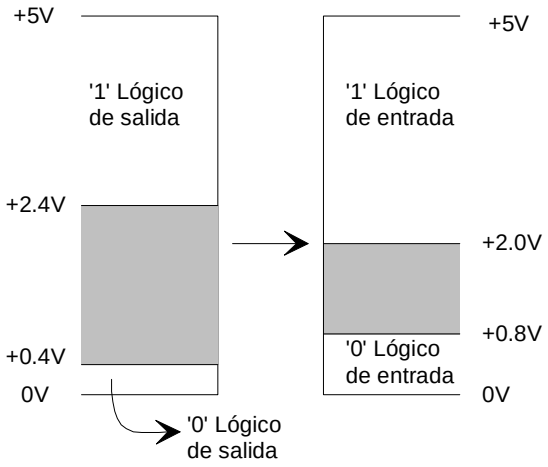


Fig. 3.5 Niveles de tensión correspondientes al '0' y al '1' en la salida y en la entrada de una puerta TTL

Señales RS-232: La figura (3.6) muestra los niveles eléctricos estándar de la RS-232 que emplea lógica negativa. Un '1' lógico corresponde a valores de tensión entre -5 y -15 V, es decir 'LO'. Un '0' lógico corresponde a valores de tensión entre +5 y +15 V, es decir, 'HI'. Estos niveles son para circuitos cargados, en vacío los niveles pueden variar entre ±25 V. El receptor admite rangos de +3 a +25 V para el '0' lógico y de -3 a -25 V para el '1'. La amplia región entre ±3 V minimiza los problemas de ruido y permite una operación fiable hasta 15 metros de distancia.

Bucle de corriente: Permite realizar comunicaciones a mayores distancias, hasta 300 metros según la velocidad (normalmente 1200 bps a 30 y 10 bps a 300 m). Los niveles 1 y 0 se codifican por la ausencia o presencia de una corriente unidireccional de 20 mA en la línea.

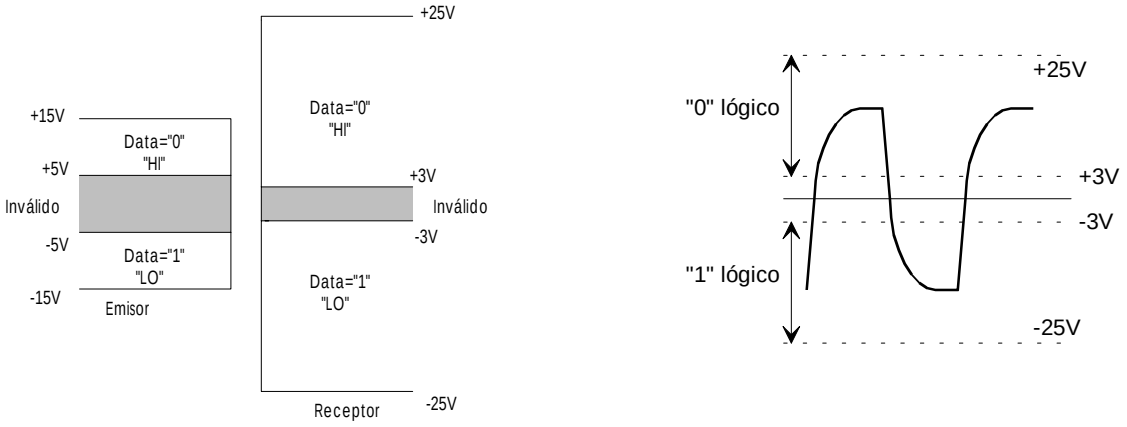


Fig. 3.6 Niveles de tensión en el emisor y receptor según la norma RS-232

Utiliza un conector estándar de 25 terminales estando adoptado por todos los fabricantes el tipo DB-25 con la asignación de pines que se muestra en la tabla (3.1). El módem (equipo de comunicación de datos o DCE) incorpora un conector hembra, mientras que el terminal o DTE (normalmente un ordenador) dispone de un conector macho. De las 21 señales definidas,

generalmente sólo se utilizan nueve y a menudo sólo son tres las empleadas: emisión, recepción y masa de señal que se corresponden con los terminales 2, 3 y 7 del conector de 25 terminales.

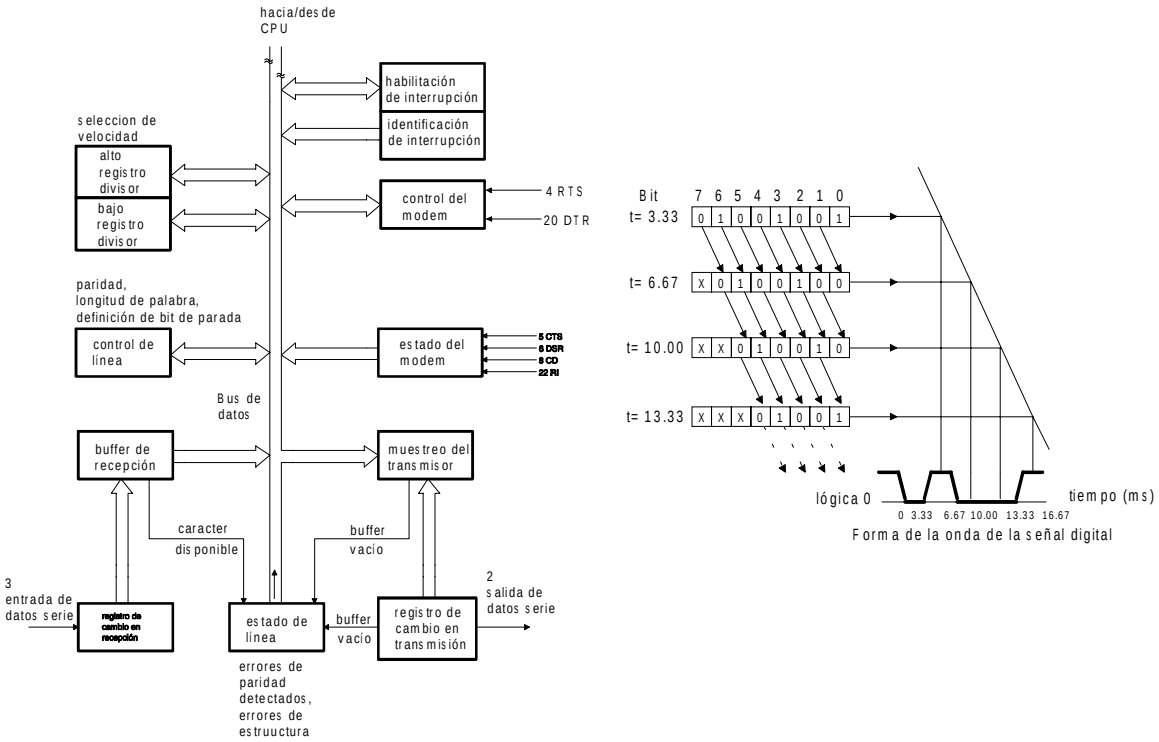


Fig. 3.7 Esquema de funcionamiento del integrado NS8250

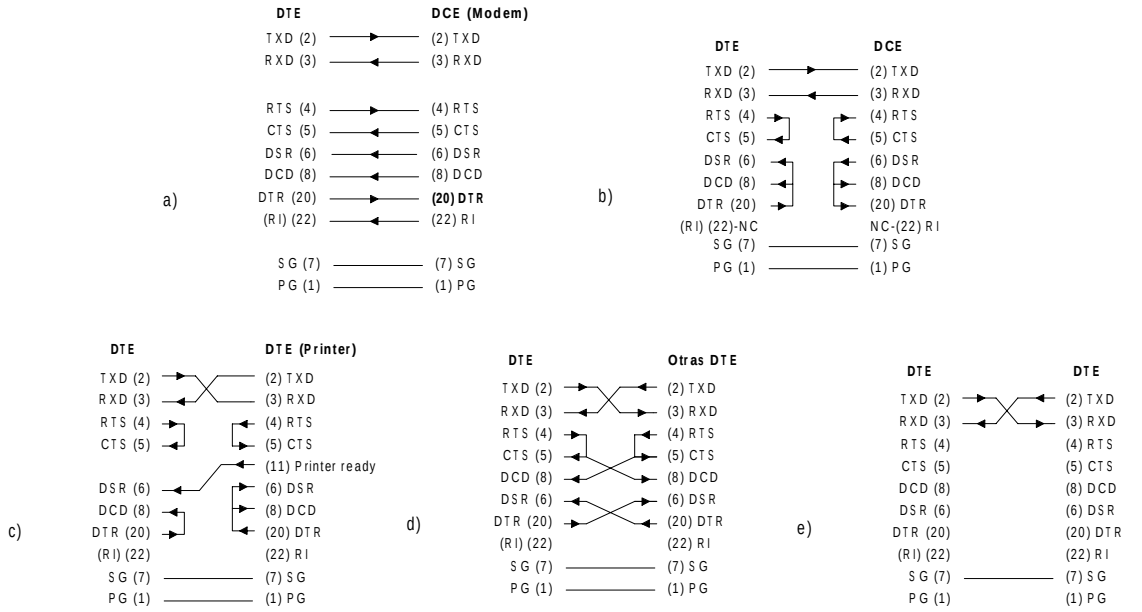


Fig. 3.8 Distintas configuraciones de conexión con RS-232

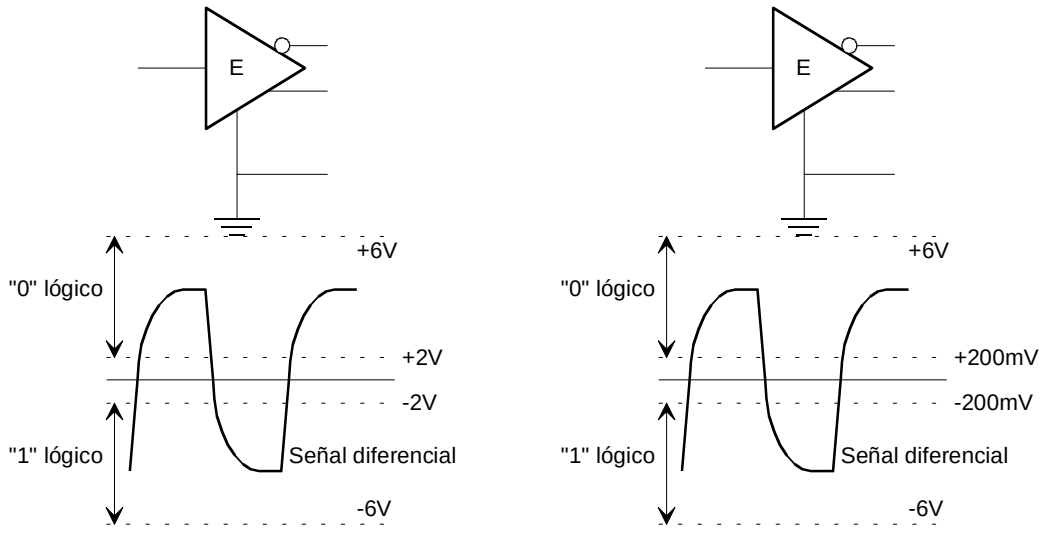
El tipo de circuitos que realizan este tipo de comunicaciones reciben el nombre de UART (Universal Asynchronous Receiver Transmiter). El interfaz RS-232C es implementado en el adaptador asíncrono de comunicaciones del PC, usando el chip de National Semiconductor INS8250 UART, cuyo esquema de funcionamiento se muestra en la figura (3.7). Este adaptador es capaz de comunicar a diferentes velocidades desde 50 a 9600 baudios. El usuario puede controlar la velocidad de transmisión, la longitud de caracteres, paridad y bits de parada a través del sistema

operativo o programas de usuario. Versiones más recientes del integrado permiten velocidades muy superiores. Por ejemplo el integrado 16550 alcanza los 115200 bps e incluye una memoria FIFO de 16 caracteres para emisión y otra de idéntica longitud para recepción. En la figura (3.8) se muestran distintas configuraciones de conexión con RS-232.

3.4.1 Variantes RS-422, 423 y 485

RS-232-C utiliza emisores y receptores no balanceados, la señal '1' es una tensión ≤ -3V y la señal '0' es una tensión ≥ +3V. Se utiliza normalmente una señal de +12 y -12 V (la especificación indica ±3V a ±25 V.) La velocidad de subida de la señal se limita a 30V/µs. esta interfaz está especificada para una velocidad máxima de transmisión de 20 kbps y una distancia de 15 m.

Cuando se requieren velocidades mayores de transmisión que las que ofrecen los anteriores sistemas es necesario utilizar un sistema de transmisión diferencial, para evitar los efectos del ruido que aparecen con tensiones en modo común en las salidas del emisor o a la entrada del receptor. La norma RS-422 fue definida por la EIA para este propósito permitiendo veocidades de transmisión de has 10Mbits/s y hasta una longitud de 1200m. Los dispositivos emisores que cumplen esta norma son capaces de transmitir señales diferenciales con un mínimo de 2V de diferencia sobre un par de líneas trenzadas y terminadas con una impedancia de 100 Ohms. Los receptores deben ser capaces de detectar una señal diferencial de ±200mV. en presencia de una señal común de ±7V. Se utilizan señales de hasta 6 V y el receptor tiene un umbral de disparo de 200mV.



Emisor y tensión de salida del mismo

Receptor y tensiones válidas a la entrada

Fig. 3.9 Niveles de tensión de salida y de entrada de los interfaces RS-422 y RS-485

Una ventaja de esta norma frente a RS-232-C es que en aplicaciones de bus, permite que un solo emisor pueda comunicar con varios receptores aunque tiene la limitación de que los restantes receptores deben tener una alta impedancia de entrada para no cargar el bus. Un problema que presentan ambos interfaces es el de la contención. Es decir, no permite que varios emisores transmitan información simultáneamente. Cuando esto ocurre, la excesiva corriente producida por la tensión en modo común generada, puede llevar a la destrucción del circuito emisor, puesto que no existen limitaciones para evitarla.

Una situación intermedia entre las dos normas comentadas es la propuesta en la norma RS-423. Ésta utiliza un receptor diferencial y un emisor que no lo es; de esta forma se permite su

interconexión con emisores o receptores RS-232C y RS-422 indistintamente. Las prestaciones que se consiguen son: 300Kbps a 12 m y 3Kbps a 1200 m.

La principal ventaja de las normas 422 y 423 es que utilizan recepción en par diferencial o transmisión balanceada, lo que las hace más inmunes al ruido. Esto es debido a que las variaciones introducidas por el ruido debido a interferencias electromágnéticas, afectarán por igual a las dos señales, y como el receptor toma el dato de la diferencia entre ambas, no se verá afectado por esta situación. Lo único que se requiere es que el receptor tenga un alto factor de rechazo al modo común, es decir que sea insensible a las variaciones conjuntas de sus dos terminales de entrada.

Para solventar algunos problemas que presentaban las anteriores normas, la EIA definió un nuevo estándar: la norma RS-485. Se considera como una interfaz multipunto (ver figura 3.10) y permite la comunicación de hasta 32 pares de emisores-receptores en un bus de datos común satisfaciendo al mismo tiempo los requerimientos de la RS-422. Las diferencias fundamentales son las siguientes:

- Margen de tensiones ampliado hasta -7V a +12V frente a -0.25 a +7 de la RS-422
- El emisor dispone de protección frente al problema de la contención.
- El margen de tensiones en el receptor va desde ±7V a ±12V manteniendo una sensibilidad de ±200mV.
- Incremento de la impedancia de entrada del receptor hasta 12 Kohms.

Para concluir este apartado hay que resaltar que las diferencias entre las distintas variantes, están únicamente en la capa física. Es decir los circuitos de tipo UART pueden ser idénticos, con lo que la programación es independiente de la variante que se esté utilizando. El emplear una u otra depende de los circuitos excitadores de línea y de los correspondientes circuitos receptores. La tabla (3.2) resume las características de las distintas variantes.

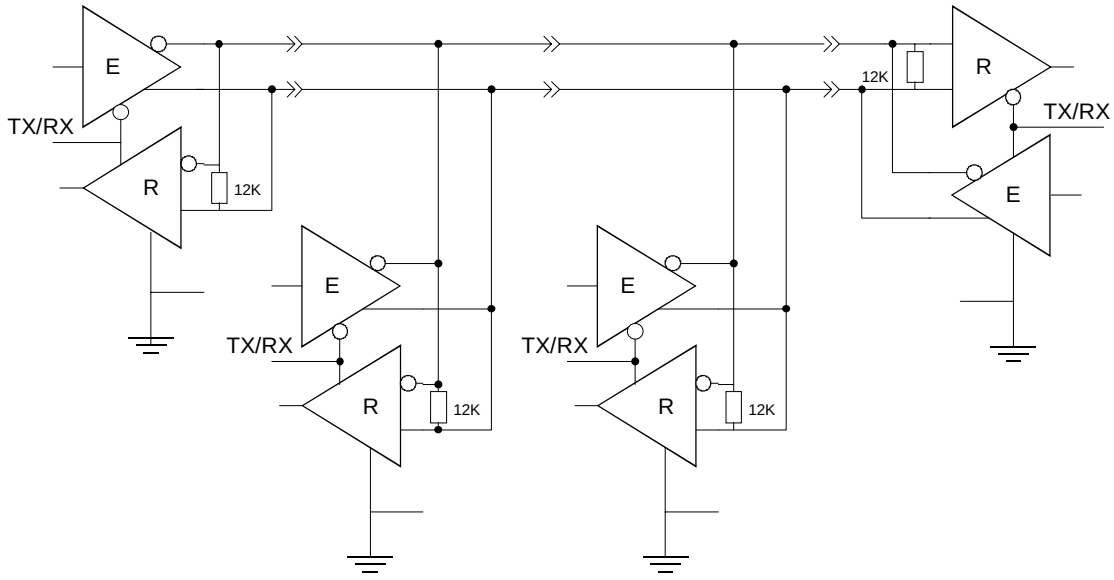


Fig. 3.10 Configuración típica de RS-485 mostrando 4 equipos conectados simultáneamente al mismo bus

Parámetro	RS-232	RS-422	RS-423	RS-485
Modo de trabajo	Simple	Diferencial	Simple	Diferencial
Nº de emisores permitidos	1	1	1	32
Nº de receptores permitidos	1	10	10	32
Longitud máxima del cable	15m	1200m	1200m	1200m
Vel. de transmisión (Bps)	20K	100K	10M	10M
Tensión en modo común	±25	+6V -0.25	±6	+12 -7
Tensión de salida	±5 V mín ±15 V máx	±2 V	±3,6 V mín ±6.0 V máx	±1.5 V mín
Carga de excitación	3kΩ-7kΩ	100Ω	450Ω mín	60Ω
Pendiente de subida	30V/μx máx	X	Determinado por la longitud del cable	X
Impedancia de entrada	3kΩ-7kΩ	4kΩ	4kΩ	12kΩ
Sensibilidad del receptor	±3 V	±200 mV sobre un modo común de ±7 V	±200 mV	±200 mV sobre un modo común de ±12 V

Tabla 3.2 Resumen de características de las distintas variantes de la interfaz RS-232

3.5 EL INTERFAZ MIDI

3.5.1 Un poco de historia

El interfaz MIDI fué diseñado originalmente para la interconexión de instrumentos musicales digitales entre sí y de estos con un ordenador. No es de extrañar por tanto que su gestación y sus características estén estrechamente relacionadas con el mundo de los instrumentos musicales y de la música y el espectáculo en general.

A comienzos de los setenta comenzaron a aparecer los primeros sintetizadores electrónicos de tipo analógico. Un sintetizador es un instrumento que genera los sonidos musicales a partir de elementos electrónicos básicos, como osciladores, generadores de envolvente o de rampa, filtros, etc. Este primer tipo de sintetizadores se podían conectar entre sí de forma analógica con una señal que proporcionaba una tensión de 1 voltio por octava. De esta forma, señales que se diferenciaban entre sí en un voltio, representaban la misma nota pero de octavas adyacentes. Con este sencillo interfaz electrónico, se podía gobernar un sintetizador de un fabricante con un teclado de otro.

Rápidamente se comenzó a introducir la digitalización y el control por ordenador de los múltiples osciladores de los sintetizadores polifónicos. A partir de este momento, el sencillo interfaz analógico de un voltio por octava dejó de ser aplicable y los instrumentos volvieron a estar incomunicados entre sí. Algunas compañías, a la vista de las limitaciones impuetas a los usuarios, comenzaron a desarrollar estructuras de bus capaces de permitir distintas expansiones. Algunas de ellas empleaban el sistema de bus serie, con objeto de rebajar el coste, mientras otras elegían el bus paralelo debido a su mayor rapidez. En lo único que estaban todas de acuerdo era en la necesidad de desarrollar una interfaz apropiada y común a todas ellas.

En diciembre de 1982, Sequential Circuits lnc. (fabricante del Prophet, primer sintetizador polifónico de difusión masiva) lanzó las primeras unidades del Prophet 600. Una de sus

características más interesantes era que disponía de una conexión de interfaz serie, y que Dave Smith, presidente de Sequential denominó entonces como Universal Synthesizer Interface (USI). Durante la Feria de Invierno de Fabricantes de Música que se desarrolló aquel mismo año, técnicos de Sequential, Yamaha y algunos otros fabricantes celebraron una reunión informal en la que comenzaron a discutir las bases de una posible estandarización. Como resultado de estas conversaciones apareció un protocolo muy similar al USI de Dave Smith, y que ofrecía la mejor relación entre velocidad, simplicidad y bajo costo.

El junio de 1983 se conectó un Prophet 6000 a un Yamaha DX-7 (instrumento basado en el afortunado, y no por ello menos importante descubrimiento de John Chowning: la técnica de sintetizado en FM. Técnica que modificaría el mundo de los teclados para siempre). El resultado careció por sí mismo de espectacularidad pero motivó el que en agosto de 1983, representantes de Sequential, Roland, Yamaha, Korg y Kawai sentaran en Tokio las bases de la norma «MIDI 1.0» (Musical Instrument Digital Interface).

Esta interfaz es quizá la mas extendida y prácticamente única dentro de su campo de aplicación pese a no haber sido respaldada por ningún organismo internacional de normalización, pero no hay instrumento musical electrónico que se precie, que no disponga de una conexión MIDI, e incluso resulta muy económico incorporar este tipo de interfaz a cualquier ordenador. De hecho, casi cualquier tarjeta de sonido convencional incluye uno.

3.5.2 El hardware MIDI

La característica técnica más importante del MIDI reside en que, para abaratar los cables y las conexiones, se ha usado un protocolo serie, básicamente el mismo que el RS-232 con un bit de comienzo (START), 8 bits de datos y dos bits de fin (STOP). Funciona a una velocidad de 31,25 kilobaudios, lo que a primera vista sin duda parece un poco extraño. Pero 31,25 Kbd. no es una velocidad tan extraña si consideramos el aún popular (y bastante barato hoy día) adaptador de comunicaciones asincrónicas 6850, cuyos registros de control internos actúan en modo división por 64, y son controlados externamente por un reloj de transmisión/recepción que funciona a 2 MHz., lo que arroja un resultado de 31,25 Kbd. A diferencia del RS-232 (que usa una tensión bipolar), el MIDI utiliza un bucle de corriente de 1,5 mA con un optoacoplador en la entrada del receptor (ver figura 3.11). En esta figura se muestra la entrada MIDI, la salida MIDI y una salida especial denominada MIDI Thru. Si observamos la figura, observamos que esta salida no es más que una réplica de la entrada convenientemente optoaislada de esta. Si no existiera esta salida, tan sólo se podrían conectar dos equipos. Como los datos emitidos por este tercer puerto son una réplica de los recibidos por el dispositivo en MIDI IN, su uso permite el encadenamiento de varios dispositivos. Aunque en teoría, la interconexión via MIDI Thru es transparente, en la práctica se produce una distorsión que puede acarrear la pérdida de mensajes después de más de tres enlaces. Por este motivo, en entornos complejos con múltiples dispositivos, es aconsejable emplear un dispositivo especial denominado expansor MIDI o "MIDI patch bay".

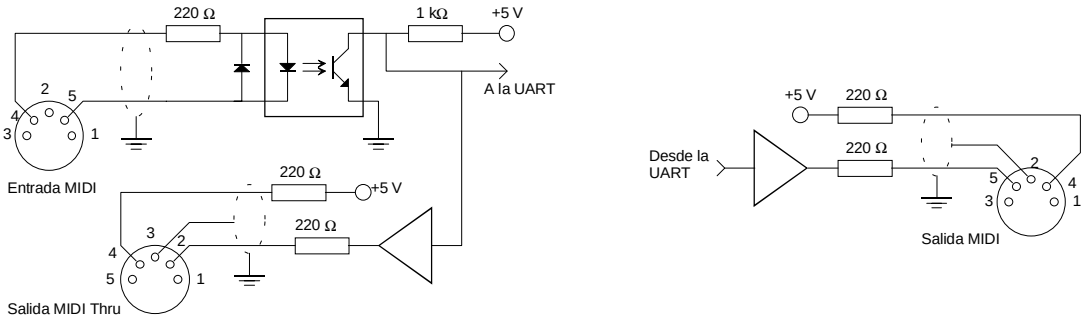


Fig. 3.11 Entradas y salidas en un interfaz MIDI

En cuanto a los conectores, el MIDI usa los del tipo DIN de 5 patillas, con hembras en los equipos y machos en los cables. Este tipo de conectores son perfectamente válidos para usarse tanto en equipos de comsumo como de estudio. Tampoco pasa desapercibido el hecho de que, por otra parte, los conectores pueden siempre ser sustituidos a la hora de la verdad por los de clase XLR profesional.

El cable que se usa en los equipos de MIDI suele ser un par trenzado y blindado cuya longitud no exceda de 15 metros. Como sólo necesita dos hilos y la pantalla, resultan conexiones económicas. Sólo hay una precaución que tomar: hay en el mercado cables terminados en conectores de tipo DIN de 5 patillas y destinados a equipos de audio. Debido a su bajo precio podemos caer en la tentación de usarlos como sucedáneo de los cables auténticos de MIDI. En la mayoría de los casos no tendremos problemas, pero el riesgo existe. ¿Por qué? Si un cable no es más que un conjunto de hilos. Pues NO, no siempre es así. El problema surge debido a dos causas distintas, cada una de ellas insignificante por sí sola. Debido a que la mayoría de los conectores DIN van soldados sobre una placa tiene sentido que éstos vayan soldados a masa. En algunos equipos la masa es también la tierra del sistema (lo cual parece apropiado), y en la mayoría de las tomas DIN la carcasa va conectada electrónicamente al blindaje. El que estos cables de audio lleven conectados a una de sus patillas no es problema, pero sí lo es el hecho de que sus blindajes estén interconectados a través del apantallado del cable. Si uno de los cables se usa para conectar dos partes de un equipo cuyos diseñadores han puesto a tierra las carcasas de los conectores, el resultado es una realimentación instantánea, y no se trata sólo de un posible zumbido de audio; sino que éste está casi garantizado. Esto se producirá por lo que se conoce como lazos de tierra que consiste en que a través de los blindajes de los cables que interconectan los distintos equipos pasa una corriente nada despreciable consecuencia de que no todos los equipos tienen el mismo potencial de tierra.

Centrándonos en la figura (3.11), y recordando que una de las premisas fundamentales del sistema MIDI es prevenir dichas realimentaciones antes de que ocurran, observemos que ninguno de los pines del conector MIDI-IN está conectado a masa, mientras que en el conector MIDI-OUT sólo el pin 2 va a masa. Esta es una característica particularmente importante en los sistemas MIDI: no hay masa común entre equipos a través del cable de conexión.

A pesar de que la figura (3.11) muestra tres tipos de conectores MIDI (IN, OUT y THRU), la mayoría de equipos sólo incorporan los de tipo IN y OUT. Algunos módulos de expansión de voces sin teclado sólo montan el IN, mientras que otros como los controladores de teclado o las bases de tiempo pueden llevar sólo el de tipo OUT. Otros equipos de mayores prestaciones incorporan el conector THRU. Dicho conector es simplemente una salida protegida que a su vez es copia directa de la señal presente en el conector MIDI-IN. En la figura (3.12) se muestra un ejemplo de interconexión mediante interfaz MIDI.

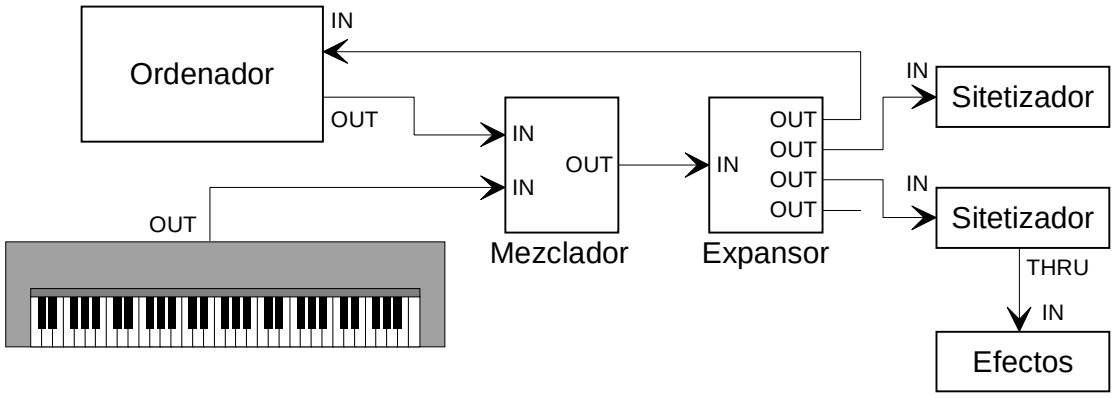


Fig. 3.12 Ejemplo de interconexión de distintos sistemas mediante interfaz MIDI

Algunos equipos incorporan fuentes de entrada múltiples con la consecuente mezcla del flujo de datos, lo cual es una operación algo compleja como se puede suponer, y exige un cuidado proceso de manera que la mezcla se realice en la secuencia adecuada. También podemos encontrar equipos con salidas múltiples, bien de tipo copia de la salida anterior, o bien tomas con la suficiente complejidad como para poder separar canales de forma autónoma. Las posibilidades de interconexión de equipos MIDI son enormes.

La configuración en cadena es la más simple que podemos formar con los equipos MIDI, y no por ello la única, o dicho de otro modo, la mejor. La configuración en anillo puede darnos excelentes resultados con equipos de la última generación, aunque también puede resultar catastrófica si son de tipo antiguo.

3.5.3 Protocolo de mensajes de MIDI

El interfaz MIDI, al contrario que el RS-232, incorpora como parte de la norma que lo define, todo el protocolo de intercambio de información entre los distintos dispositivos. Esto incluye el tipo de mensajes que intercambian los equipos y el significado de cada uno de ellos. Este protocolo es muy completo y versátil y permite grandes posibilidades en cuanto a interconexión de instrumentos musicales. Su principal inconveniente, es debido precisamente a este alto grado de normalización que lo hace esclavo del tipo de aplicaciones para las que fue pensado. Es por esto que pese a ser un interfaz económico y con múltiples posibilidades de interconexión entre equipos, como se muestra en la figura (3.12) no se emplee fuera del ámbito de la música. Sin embargo, este ámbito no se restringe únicamente a los instrumentos musicales, sino que también podemos conectar por este tipo de interfaz, generadores de efectos, mesas de mezclas digitales, equipos de grabación, bases de tiempo para sincronización de mesas de edición de audio y video o sistemas de control de iluminación, consiguiendo de esta forma efectos luminosos e incluso pirotécnicos, perfectamente sincronizados con las notas que genera un determinado intérprete en el escenario.

Aunque uno de los objetivos fundamentales de los equipos MIDI era sustituir el antiguo sistema de interfaz mediante control por voltaje, parecía lógico que sus descubridores pretendieran de él que fuese capaz de algo más que enviar el mensaje de tocar una determinada nota. Por ejemplo, mientras que los primeros sintetizadores eran casi como los órganos electrónicos de la época en que una tecla no era mucho más que un interruptor abierto o cerrado, pronto aparecieron los teclados que permitían un cierto control según la presión ejercida en ellos. Esta sensación de velocidad dio a los teclados un tacto mucho más «natural» y proporciona más «sentimiento» a la interpretación.

El protocolo MIDI está constituído por el envío de mensajes que indican al resto de equipos que realicen alguna acción, como activar o desactivar una nota concreta, cambiar el banco de sonidos, aplicar un determinado efecto, etc. Estos mensajes conforman el lenguaje a través del cual se comunican todos los dispositivos.

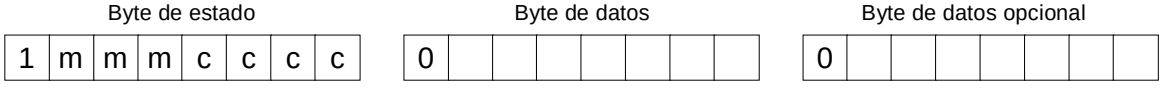


Fig. 3.13 Estructura binaria de un mensaje MIDI. Todos los mensaje comienzan por un byte de estado, que incluye 3 bits con el tipo de mensaje y 4 con el número de canal. A continuación va uno o dos bytes de datos dependiendo del mensaje que se envíe.

Los mensajes (figura 3.13) están compuestos por un byte de estado y uno o más bytes de datos. Los bytes de estado proporcionan información sobre el tipo de acción a efectuar, y seleccionan el canal en el que se realizará esa acción, mientras que los bytes de datos subsiguientes

especifican dicha acción. Por ejemplo, el byte de estado puede contener «activar una nota», y el byte de datos le indica que la nota es «un DO en la cuarta octava». En la norma MIDI los bytes de estado tienen siempre a 1 su bit más significativo (MSB), mientras que el bit más significativo de los bytes de datos siempre está a 0. La figura (3.13) muestra la estructura binaria de un mensaje MIDI.

El byte de estado incluye tres bits que especifican el tipo de mensaje, lo que permite ocho tipos distintos de mensaje. Los cuatro bits menos significativos indican el número de canal al que se dirige el mensaje. En una configuración MIDI, cada dispositivo puede ser asignado a un canal y múltiples dispositivos pueden ser asignados al mismo canal. Al hacer esto, el dispositivo responderá a los mensajes que lleven la etiqueta correspondiente a ese canal.

Byte de estado	Nº bytes de datos	Descripción	Byte de datos 1	Byte de datos 2
1000nnnn	2	Desactivación de nota	Altura	Velocidad
1001nnnn	2	Activación de nota	Altura	Velocidad
1010nnnn	2	Expresión de nota Post-pulsación	Altura	Presión
1011nnnn	2	Cambio de control	Tipo de control	Intensidad
1100nnnn	1	Cambio de programa	Programa	
1101nnnn	1	Expresión de canal	Presión	
1110nnnn	2	Cambio de tono "Pitch Bend"	MSB	LSB
1111xxxx	variable	Mesajes de sistema		

Tabla 3.3 Resumen de los distintos tipos de mensajes MIDI

Ya hemos dicho anteriormente que la mayoría de los bytes de status llevan un número de canal, y como se puede suponer, los mensajes que van precedidos por el número de canal se denominan mensajes de canal. La norma MIDI prevé también mensajes sin número de canal y destinados a causar una respuesta similar en cualquier equipo que se sitúe en el bus. Los bytes de status cuyo prefijo de mensaje lleva los cuatro bits de mayor orden a 1 son los mensajes de sistema. Es preferible que comencemos a referirnos a ellos en notación hexadecimal, por lo cual, los mensajes de sistema serán de la forma $Fx, y donde la parte $x será el mensaje específico (el símbolo $ indica base hexadecimal).

En situación normal, los mensajes de canal constan de un byte de estado seguido por uno o dos bytes de datos (y cuyos MSB están a cero). El resumen de estos bytes puede verse en la tabla (3.3). El mensaje más comúnmente usado en los sistemas musicales es el que hace tocar una nota. Para ello están previstos dos bytes de status; los mensajes de canal NOTE ON y NOTE OFF. Vamos a centrarnos primero en el NOTE ON. Su codificación es $9n, donde $n representa el número de canal formado por los 4 bits, y requiere 2 bytes de datos. El primer byte de datos indica la nota a ser tocada. Debido a que por definición los MSB deben ser 0, hay combinaciones para 128 notas diferentes. Un piano tiene sólo 88 teclas, por lo que parecen ser suficientes, aunque algunos equipos de ultimísima generación trabajan con partituras micro-tonales, en las que se usan tonos intermedios de cada nota base del piano, por lo que en estos casos 128 combinaciones no serían suficientes. El segundo byte de datos se ocupa de especificar la velocidad, o lo que es igual, la dureza de la pulsación sobre la tecla. Si el equipo no incorpora teclado sentivo, y aunque la norma MIDI especifica que la velocidad debe ser de un valor igual a $20, los instrumentos envían normalmente una señal cuyo valor suele centrarse sobre $40. Es necesario que se envíen ambos bytes de datos, aunque el dispositivo no lo tenga implementado.

El mensaje de canal NOTE OFF (cuyo prefijo es $8n) consta también de dos bytes; el número de nota y la velocidad de «tecla soltada». Pero MIDI permite que una nota sea también desactivada mediante el envío de una nueva orden de NOTE ON, cuya velocidad sea igual a 0, y la razón de ello es simplemente permitir el estado de funcionamiento continuo.

Mientras que la mayoría de los bytes de un mensaje deben ser enviados necesariamente, no ocurre lo mismo con el byte de status. Sobre todo en el caso de que el nuevo byte sea idéntico al anterior, caso en el cual la norma MIDI nos permite omitirlo. Así por ejemplo, en el caso de que desactivemos una nota mediante un mensaje NOTE ON (con V=0), no precisaremos usar los bytes de status. Si tocamos un conjunto de tres notas en el teclado, los mensajes no usarán una longitud de 18 bytes (3 triples byte de NOTE ON y 3 triples byte de NOTE OFF), sino que nos bastará con usar sólo 13 bytes. Como vemos, en caso de que todos los mensajes se sitúen en el mismo canal conseguiremos un ahorro de un 33 por 100 aproximado sobre el ancho de banda del bus. (Un «atasco» en la información enviada por MIDI puede traducirse, en la práctica, en retrasos perceptibles entre el momento de pulsar una tecla y el instante en que suena la nota correspondiente; problema éste bastante grave en los grandes sistemas, aunque hay maneras de atenuarlo).

3.6 INTERFACES PARALELO

En términos generales, una interfaz es un nexo de conexión que facilita la comunicación entre dos dispositivos.

Un gran número de unidades de disco tienen un interfaz estándar aunque algunos fabricantes, emplean interfaces no estándar en sus equipos. Afortunadamente, este tipo de interfaces propietarios y exclusivos de un fabricnate son cada vez menos habituales y actualmente casi todos los dispositivos se pueden conectar a través de interfaces normalizadas, documentadas y públicas. Las interfaces las podemos dividir en dos clases; aquellas que están entre el mecanismo del dispositivo y su controlador, y aquéllas que están entre la unidad básica y el controlador. El controlador puede ser una unidad separada o puede estar incluido en el dispositivo. En este último caso, la interfaz entre el dispositivo y la controladora es usualmente inaccesible y es la situación prácticamente universal hoy dia. El controlador es ocasionalmente empaquetado en la unidad básica del ordenador; en este caso, el estándar 'lógico' de interfaz entre la unidad básica y el controlador se mantiene. Por lo tanto, el software puede ser usado igualmente, aunque la interfaz no exista físicamente. Aquí se entiende por controlador el circuito o dispositivo que se comunica directamente con la CPU mediante comandos y en base a esto produce unas señales de control que actuan sobre el dispositivo. Por ejemplo, un comando a un disco puede ser el leer un sector y las señales al disco son de activar un motor, mover la cabeza, comprobar posicionado, leer secuencia de patrones magnéticos y tras su decodificación enviarlos a la CPU a través del interfaz. Un comando similar podría enviarse a una impresora, sin embargo, las señales de control serían muy distintas: desplazar la cabeza de impresión, activar determinados inyectores, etc.

3.7 EL INTERFAZ ST-506/412

3.7.1 Generalidades

Entre los interfaces controladores de dispositivos uno de los primeros y más conocidos es el ST-506[1]. El separador de datos está en el controlador, el cual define un rango de datos de 5 Mbits por segundo. El separador de datos es el circuito que tomando como entrada la señal proveniente de la cabeza magnética, extrae por una parte los datos y por otra la señal de reloj embebida. El

[1] Su hoja de especificaciones puede encontrarse en Peripheal INTEL vol. II

dispositivo está preparado para controlar un motor paso a paso con el fin de desplazar las cabezas pista a pista, y las pistas son seleccionadas por pulsos de motor a razón de un pulso por milisegundo. Cada uno de los pulsos mueve la cabeza a una pista, siendo el tiempo medio de acceso grande si el dispositivo tiene muchas pistas. Pueden ser seleccionadas 16 cabezas, permitiendo 4 discos. El servocontrol de la localización de la pista no se usa. Una variante de ésta es la interfaz ST-412, que contiene un buffer de pulsos. El controlador puede enviar pulsos de paso (step) más rápidamente de lo que puede desplazarse la cabeza, y el buffer los pasará, entonces, a la velocidad adecuada para el motor. De esta forma podemos usar motores paso a paso más rápidos según las necesidades. El ST-506/412 está diseñado para sistemas de bajas prestaciones y bajo costo, como los primeros PC's. Es utilizado en muchos dispositivos de 3 y 1/2" y 5 y 1/4", aunque a menudo en los PC's éstos están integrados con el controlador. Puede manejar 4 unidades de disco. Se trata de un interfaz obsoleto pero que permite ver las distintas capas de abstracción de los interfaces y la evolución hacia sistemas más modernos y mucho más completos como SCSI.

3.7.2 Cableado

Esta interfaz eléctrica está dividida en dos cables separados (etiquetados habitualmente como J1 y J2) más la alimentación (J3). J1 es un cable plano de 34 vías y J2 es también un cable plano de 20 vías. Las señales del conector J1 son de control. Todas ellas son digitales (colector abierto y niveles TTL) y funcionan en configuración simple, es decir, la información se obtiene de ellos tomando la diferencia de tensión entre la señal y una referencia común.

El cable J2 contiene señales de datos, que son diferenciales, es decir, la información se obtiene de ellos tomando la diferencia de tensión entre dos hilos de señales conjugadas. Por ejemplo, el dato leído será 1 o 0 dependiendo de la diferencia de voltaje entre *MFM_Read_Data* y $\overline{MFM\_Read\_Data}$. Esta controladora realiza todo el trabajo y el disco lo único que hace es leer y escribir los patrones magnéticos tras acondicionar debidamente la señal. El mismo disco podía ser utilizado con codificación RLL, ya que la codificación y decodificación se realizaba en la controladora.

El conector J1 se conecta a todas las unidades en cadena, en tanto que el conector J2 se conecta por separado a cada una de ellas, como muestra la figura (3.14).

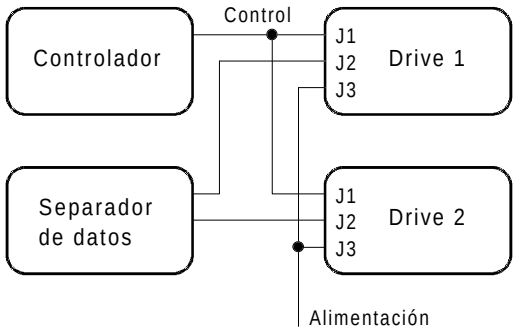


Fig. 3.14 Cableado entre el controlador, el separador de datos y los dispositivos en el ST-506/412

3.7.3 Señales y funcionalidad

CONECTOR J1

1 $\overline{Head\_Select}$ (0:3) (O)
Estas cuatro líneas seleccionan en binario una cabeza determinada. El hecho de que sean activas a nivel bajo implica que la selección se hará con lógica negativa, es decir, la combinación 1101 en estas señales seleccionarán la cabeza número 2

2 $\overline{Write\_Gate}$ (O)
Se activa para indicar una operación de escritura, activa la corriente de escritura en la cabeza

3 $\overline{Direction\_In}$ (O)
Indica la dirección de la cabeza durante los pulsos de step (1 hacia afuera, 0 hacia adentro)

4 $\overline{Step}$ (O)
A través de esta señal se le aplica un pulso activo a nivel bajo por cada movimiento de pista a pista que se desee mover la cabeza

5 $\overline{Drive\_Select}$ (1:4) (O)
A través de estas cuatro señales se selecciona uno entre cuatro posibles unidades de disco que esta interfaz puede manejar. No está codificada, sino que que cada disco tiene una línea, con lo cual la activación de más de una señal será considerada como activación errónea.

6 $\overline{Track0}$ (I)
Se activa cuando la cabeza está posicionada sobre la pista 0, que es la más exterior.

7 $\overline{Index}$ (I)
Es activado por el disco cuando la cabeza está posicionada al comienzo de una pista.

8 $\overline{Ready}$ (I)
Indica al controlador que en el dispositivo no existe ninguna condición de error, y por tanto, la operación en curso (lectura o escritura) puede continuar.

9 $\overline{Seek\_Complete}$ (I)
Indica al controlador que la operación de posicionamiento sobre la pista deseada se ha completado y puede proceder a lectura o escritura.

10 $\overline{Write\_Fault}$ (I)
Indica al controlador que la operación de escritura se ha detectado como errónea. Lo que ocurre no es que se haya escrito mal, sino que no se ha escrito debido a un fallo drástico del hardware ya que es lo único que la controladora puede detectar. Cuando un dato se ha escrito mal no se entera nadie hasta que se lee.

CONECTOR J2

1 $\overline{MFM\_Write\_Data}$ y *MFM_Write_Data* (O)
Esta pareja de señales constituyen el dato a escribir en el disco codificado en MFM

2 $\overline{MFM\_Read\_Data}$ y *MFM_Read_Data* (O)
A través de estas dos señales el disco envía a la controladora la secuencia de bits leídos

3 $\overline{Drive\_Selected}$ (I)
Con esta señal, el dispositivo indica al controlador que ha reconocido la selección. Es lógico que esta señal no forma parte del conector J1, puesto que notifica una situación particular de cada cable. Si al controlador le llegara una señal de *Drive_Selected* por el cable J1 (que está conectado a todos los dispositivos) no sabría quién le está contestando.

3.7.4 Ejemplo de implementación: La tarjeta controladora WD1003-WAH

Aquí se verá un ejemplo de una tarjeta controladora para la interfaz ST-506/412. El diagrama de bloques de esta tarjeta puede verse en la figura (3.15)

La solución que se comenta, como la mayoría de ellas, se basa en el uso de un conjunto de circuitos integrados, que consiste en dividir la función a realizar en cometidos más simples que puedan ser abordados por integrados de no muy alta complejidad. Estos chips están pensados para conectarse entre sí casi pin a pin. El resultado es que se tiene una tarjeta con 4 ó 5 chips LSI o VLSI (que realizan entre todos la función compleja) y varios chips SSI (puertas lógicas y 'drivers' excitadores de línea). Actualmente estas controladoras tienden a realizarse a medida con lo que se englobaría todo en un solo chip. La tarjeta en cuestión es la WD1003-WAH y controla hasta dos discos winchester de hasta 16 cabezas y 2048 cilindros cada uno. La interfaz con la unidad básica que presenta esta tarjeta, es el slot de expansión de los PC's.

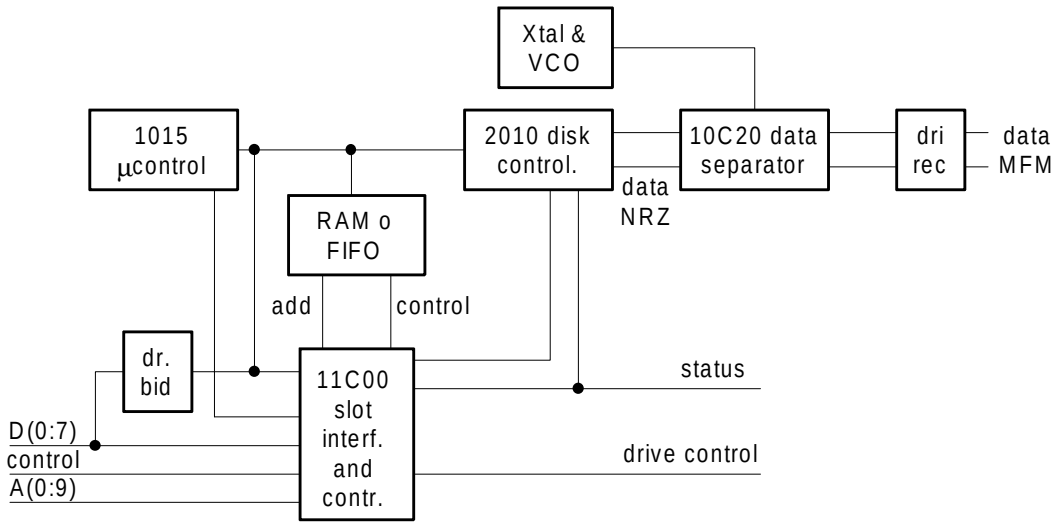


Fig. 3.15 Diagrama de bloques de una tarjeta controladora para el ST-506/412

Está basada en el conjunto de integrados formado por los siguientes circuitos:

1 WD2010A-05
Es el controlador de discos winchester. Suministra la mayoría de las señales de la interfaz ST-506. La única función que deja para otro chip es la de separador de datos (realizada por el WD10C20). Por el lado de la unidad básica se conecta con una memoria tampón (una FIFO o una RAM y un contador) y un controlador de esta memoria. Además suministra el control de interrupciones y transferencias por DMA.

2 WD11C00C-22
Es el controlador de la interfaz con la unidad básica, en este caso el PC. Además de esta interfaz física, ejecuta algunas otras funciones como el direccionamiento del buffer para las transferencias en escritura con la unidad básica, separa en bytes la palabra (word) que le envía la unidad básica, realiza él la selección de cabeza, controla el led de actividad del disco, etc.

3 WD1015-27
Es un microcontrolador de 8 bits que controla y coordina el buen funcionamiento de los dos LSI anteriores. Recibe y envía información de comandos y de estado a través del bus interno (que está multiplexado) de la tarjeta. El 'firmare' de control reside en una ROM interna de 2K que posee el propio micro.

4 WD10C20
Es el separador de datos. En lectura, realiza la sincronización de los datos del disco, suministra al controlador (WD2010A-05) el reloj con el que debe muestrear los datos para decodificar la secuencia MFM. En escritura, realiza la precompensación. Hay que suministrarle por el exterior algunos elementos analógicos para el VCO (Oscilador controlado por voltaje) interno.

Esta tarjeta presenta a la unidad básica 7 registros de lectura/escritura internos al WD2010A-05 y otros 3 más externos. Los registros internos al controlador se llaman registros de tarea. En ellos se escribirán los comandos y los parámetros de las operaciones a realizar. Como ejemplo del tipo de información que manejan estos registros, se enumeran en la tabla (3.4) los registros de tarea.

Dirección	Lectura	Escritura
1F1 h	Registro de errores	Precompensación
1F2 h	Cálculo del sector	Cálculo del sector
1F3 h	Cilindro (byte bajo)	Cilindro (byte bajo)
1F4 h	Cilindro (byte alto)	Cilindro (byte alto)
1F5 h	Dispositivo/Cabeza	Dispositivo/Cabeza
1F6 h	Estado	Comando

Tabla 3.4 Registros del controlador WD2010A-05

El controlador soporta 8 comandos posibles, que se escribirán para su ejecución en el registro 1F6h. Los comandos son los siguientes:

1 *Restore*
Mover la cabeza hasta la pista 0

2 *Seek*
Mover la cabeza un número de pasos determinado

3 *Read Sector*
Leer un número de sectores de 1 a 256 a partir de uno dado (el número es el Sector Count)

4 *Write Sector*
Análogo al anterior pero de escritura.

5 *Read Verify*
Leer un sector para la comprobación del CRC sin pasarlo a la unidad básica

6 *Format Pista*
Formatear una pista

7 *Diagnose*
Ejecuta una rutina de autotest de la tarjeta y comunica el resultado a la unidad básica

8 *Set Parameter*
Fijar parámetros, como el número de sectores por pista, el número de cabezas, etc.

La ejecución típica de comandos se lleva a cabo con la siguiente secuencia de operaciones:

- En reposo, el controlador tiene las señales de control desactivadas y su registro de estado indica la situación de Ready

- La unidad básica escribe los parámetros de la operación a realizar en los registros de tarea. Luego escribe el comando. Si la operación es de escritura, debe escribir el sector a escribir en la memoria tampón. En este caso el WD11C00C (el controlador de la interfaz con la unidad básica) sabe que el comando requiere que se rellene el buffer y controla esta transferencia por DMA

- Cuando termina esta transferencia (si alguna vez empezó), el microprocesador (WD1015-27) notifica al controlador que puede empezar la ejecución del comando, pues todo está listo.

- A la terminación del comando el controlador lanza una interrupción al microcontrolador, éste examina el registro de estado y si decide que la información es coherente con el comando que se ejecutó, entonces lanza a la unidad básica la interrupción y se retorna al estado de reposo.

Aunque este esquema de funcionamiento pueda parecer farragoso, responde al criterio de división de tareas y responsabilidades. Este modo de operar tiene la ventaja de que se puede mejorar el sistema mejorando cualquiera de sus componentes. Por ejemplo, se puede cambiar el circuito codificador MFM por otro RLL manteniendo el mismo disco con lo la capacidad se verá aumentada en un 50% como ya se comentó en el capítulo anterior.

3.8 INTERFAZ ESDI

Posteriormente las investigaciones han ido encaminadas a dispositivos que incorporen el separador de datos, porque esto da al diseñador más posibilidades para incrementar la capacidad y el rendimiento. El más popular de los dispositivos para control de pequeños discos con separador de datos es el "Extended Small Device Interface" o ESDI. Esto no restringe el rango de datos, y la mayoría de los controladores manejan cualquier rango sobre 10 Mbits por segundo (algunos 20 Mbits/seg). La selección de la pista se realiza transmitiendo por las direcciones a razón de un pulso por pista. Hay previsión para que el controlador interrogue al dispositivo, el cual informará sobre los parámetros necesitados por el controlador tales como número de cabezas, cilindros, bytes por pista, etc.

3.9 BUS SCSI

3.9.1 Generalidades

La interfaz entre la unidad básica y el controlador es a menudo diseñada para seguir las especificaciones de la unidad básica (por ejemplo, el IBM PC). Sin embargo, existe actualmente una tendencia hacia el uso de interfaces independientes de la unidad básica estándar, de los cuales la más popular es la 'Small Computer System Interface' o SCSI, que es un desarrollo del SASI producido por la firma 'Shugart Associates Standard Interface'. SCSI está ahora reconocida como estándar ANSI bajo el nombre ANSI X3.131-1986. Soporta muchos tipos de periféricos, no sólo discos, y con simples adaptadores puede conectarse a la mayoría de las unidades básicas. Algunas unidades básicas de computadora recientes soportan SCSI y no necesitar ni siquiera adaptador.

SCSI puede ser usada de dos maneras: como interfaz de la unidad básica construida en una interfaz inteligente, y como interfaz de la unidad básica con un controlador separado, que conecta uno o más dispositivos. La primera forma es normalmente la más barata cuando se utiliza un único dispositivo. La segunda forma se utiliza cuando hay un grupo de dispositivos (o una mezcla de dispositivos periféricos). La especificación SCSI tiene muchas características opcionales. Una de ellas permite al controlador dirigirse al dispositivo para iniciar un movimiento de la cabeza, y entonces desconectar el dispositivo del controlador y conectarlo a otro dispositivo cuya cabeza está ya en la pista correcta: puede transferir datos durante un tiempo comparativamente largo antes de que el primer dispositivo esté preparado para realizar eso. Cuando hay más de un programa, o más de una unidad básica, usando un grupo de discos, este rasgo puede aumentar el rendimiento del sistema. Sin embargo, el gran número de opciones y alternativas permitidas por el estándar SCSI puede dar problemas de compatibilidad puesto que algunos fabricantes no implementa un desarrollo completo de la norma.

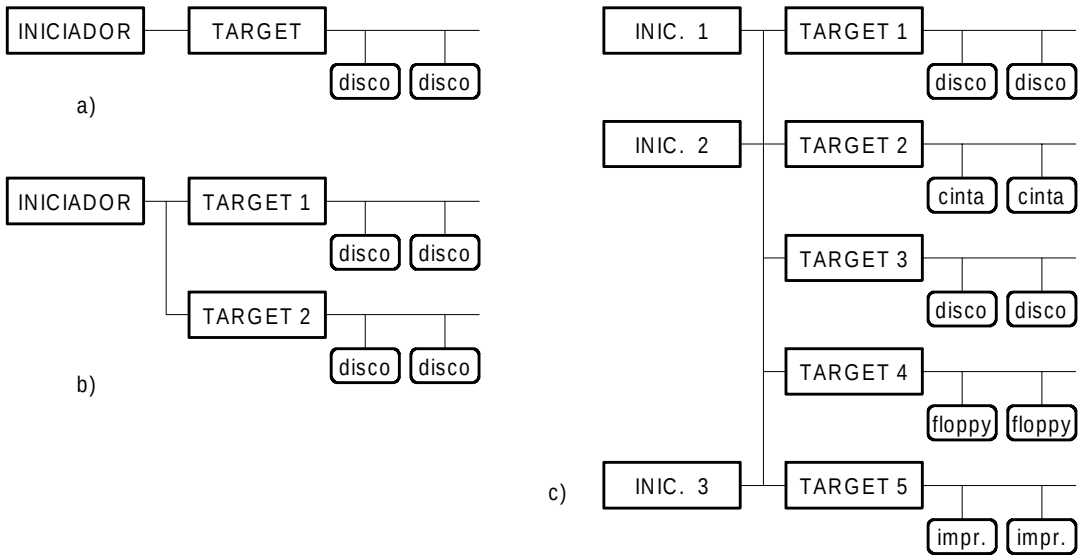


Fig. 3.16 Algunas configuraciones permitidas por la norma SCSI. a) Un único iniciador y un único target, b) Un único iniciador y varios target, c) Varios iniciadores y varios target. En todos los casos se han incluído varios dispositivos lógicos en cada periférico.

Según especifica la norma, existen dos tipos de dispositivos que se pueden conectar al bus SCSI, el iniciador (generalmente un procesador principal) y el target (generalmente un periférico). Al bus se pueden conectar un máximo de 8 dispositivos (iniciadores o targets), de los cuales al menos uno debe ser un iniciador y al menos uno debe ser un target. En la figura (3.16) se muestran algunas configuraciones posibles. Un determinado periférico SCSI puede contener internamente varios dispositivos, como por ejemplo varios discos, o varios lectores de CD-ROM y ser considerados como un único periférico por el sistema central. Estos elementos, reciben el nombre

de dispositivos lógicos y se les asigna un segundo identificador o LUN (Logic Unit Number). Dentro de un periférico SCSI que tiene varios dispositivos lógicos, éstos deben ser iguales y aceptar los mismos comandos. Salvo en los grandes sistemas, un periférico SCSI tiene normalmente un único dispositivo al que se le asigna el LUN cero.

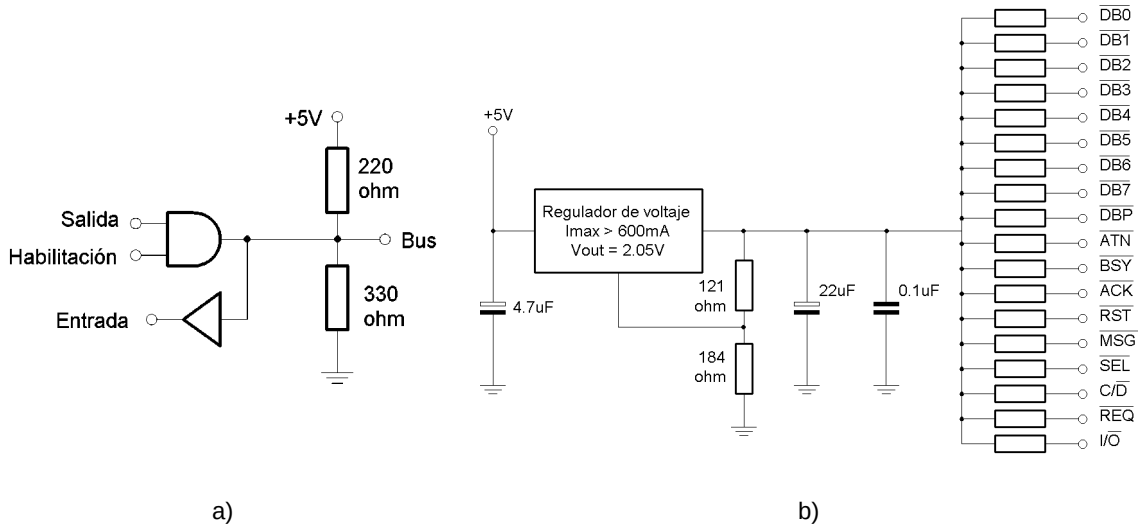
Toda comunicación (síncrona o asíncrona) a través del bus se realiza siguiendo una secuencia determinada de eventos. Estos eventos están agrupados por la funcionalidad en lo que se denominan fases. El bus en cada instante sólo se encontrará en una fase concreta. La fase está determinada por el estado de las señales de control del bus.

A continuación se dará una relación de las señales y la funcionalidad de la interfaz SCSI. Luego se describen someramente cada una de las fases, sin hacer referencia a los tiempos que especifica la norma[2], considerando en primer lugar la transferencia asíncrona de datos y la transferencia síncrona de datos, asi como la descripción de las fases de transferencia de información. Finalmente, se describen las condiciones especiales del bus.

3.9.2 Señales y funcionalidad

Esta interfaz tiene un total de 18 señales. Nueve son usadas para el control y las nueve restantes son para datos. El bus de datos es por tanto paralelo de 8 bits más uno de paridad.

Existen dos modos de implementación eléctrica, una simple (con referencia a una tierra común) o 'single-ended' (SE) y otra diferencial o 'differential-ended' (DE) donde cada señal es transmitida por un par de hilos conjugados. En la configuración simple, el nivel lógico se obtiene de la diferencia de potencial entre el hilo de señal y la tierra común. En la configuración diferencial, se obtiene de la diferencia de potencial entre cada par de hilos conjugados. El modo simple es más cómodo y barato de implementar pero está limitado a un cable de 6 metros (que es suficiente si se implementa como bus local). El cable diferencial puede llegar hasta 25 metros. Estas dos implementaciones son incompatibles entre si, y por lo tanto, en un sistema todos los dispositivos deberán ser del mismo tipo, no pudiéndose mezclar dispositivos 'single-ended' con dispositivos 'differential-ended'. No obstante, en un mismo sistema pueden convivir más de un bus con lo que un sistema puede trabajar simultáneamente con dispositivos SE y DE siempre que se conecten a buses distintos. En los sistemas pequeños o medianos, habitulmente sólo se emplea conexión SE.



a) b)

Fig. 3.17 Circuitos terminadores del bus en modo simple. a) en SCSI-1 b) alternativa en SCSI-2

---

[2] Aquéllos que estén interesados en los tiempos pueden dirigirse a las normas ANSI X3.131-1986 (SCSI-1) y ANSI X3.131-1994 (SCSI-2)

Tanto una como otra implementación deben incorporar un terminador en los extremos del bus. Estos terminadores son necesarios ya que al trabajar a frecuencias elevadas, hay que tener en cuenta las posibles reflexiones de señal que se pueden producir en los extremos de los cables causadas por una desadaptación de impedancias. Hay que tener en cuenta que cuando por una línea eléctrica se propagan señales de alta frecuencia, no pueden considerarse simplemente señales, sino que deben considerarse como ondas electromagnéticas que se propagan a través de una línea de trnasmisión. Al comportarse como ondas, pueden producirse reflexiones en los extremos de un cable de la misma forma que se reflejan las ondas de agua en las paredes de un estanque. Un estudio detallado de este fenómeno requeriría entrar en la teoría de líneas de transmisión y queda por tanto fuera del alcance de este curso. Basta tener en cuenta que si se produjesen reflexiones en la señal transmitida ha través del bus, podrían producirse interferencias entre la señal emitida y la reflejada desde el extremo lo que provocaría un mal funcionamiento del bus. Los terminadores tienen por tanto la función de simular la continuación del bus como si de una línea infinita se tratase. En las figuras (3.17) y (3.18) se muestran los circuitos terminales que deben colocarse en los extremos del bus para una y otra configuraciones.

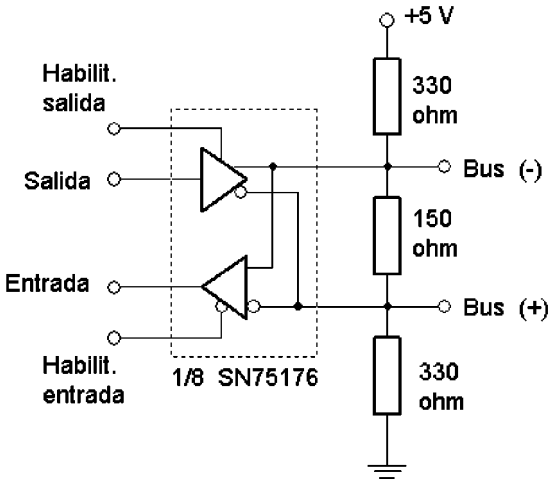


Fig. 3.18 Circuito terminador para SCSI en modo diferencial.

Las señales del bus SCSI son las siguientes:

1 $\overline{BSY}$
La señal *busy* indica que el bus está siendo usado. Cuando $\overline{BSY}$ está activada, nadie puede acceder al bus excepto los dos dispositivos interlocutores.

2 $\overline{SEL}$
La señal *select* la emplea un iniciador para seleccionar un target o bien un target para seleccionar un iniciador.

3 $C / \overline{D}$
La señal *control/data* la maneja el target para indicar si por el bus de datos circula información de control o de datos.

4 $I / \overline{O}$
La señal *input/output* la maneja el target para indicar la dirección de los datos en el bus (cuando es verdadera indica entrada al iniciador). Esta señal es usada también para distinguir entre las fases de selección y reselección.

5 $\overline{MSG}$
La señal *message* es la señal con la que el target indica que la información en curso en el bus es un mensaje.

6 $\overline{REQ}$
La señal de *Request* (solicitud o petición) es la que el target emplea para indicar una petición en el protocolo de transferencia de datos por *REQ/ACK*

7 $\overline{ACK}$
La señal *Acknowledge* (reconocimiento) es la que maneja el iniciador para indicar un reconocimiento en el protocolo de transferencia de datos REQ/ACK

8 $\overline{ATN}$
La señal *Attention* es la señal con la cual el iniciador indica la condición de atención.

9 $\overline{RST}$
La señal *Reset* indica la condición de reset. Puede ser activada por cualquier dispositivo en cualquier momento y coloca al bus en un estado inicial.

10 $\overline{DB}$ (0:7), *DBP*
Las señales *Data Bus* son ocho señales de datos y una señal de paridad y conforman el bus de datos. *DB7* es el bit de más peso (más significativo). La paridad *DBP* de los datos es impar. El empleo de la paridad es una opción del sistema.

El cable de conexión es una cinta plana de 50 vías, que está conectada en cadena a todos los dispositivos.

3.9.3 Fases del bus SCSI

FASE DE BUS LIBRE

Se emplea para indicar que ningún dispositivo se encuentra utilizando el bus y que se encuentra disponible para los demás. Los equipos conectados no detectan la fase de bus libre hasta que $\overline{SEL}$ y $\overline{BSY}$ permanecen falsos durante un cierto tiempo (>400ns)

A esta fase se llega cuando se conecta la alimentación o después de un RESET del bus. También se llega a esta fase durante una operación normal del bus si se envía alguno de los mensajes: COMMAND COMPLETE, DISCONNECT, ABORT, BUS DEVICE RESET, RELEASE RECOVERY, ABORT TAG y CLEAR QUEUE.

FASE DE ARBITRAJE

La fase de arbitraje permite a un dispositivo (iniciador o target) ganar el control del bus para que pueda comunicarse con otro (target o iniciador).

La implementación de la fase de arbitraje es una opción del sistema. Aquellos que no implementan esta opción tienen un solo iniciador y un solo target, que siempre estarán lógicamente conectados al bus. Para ganar el bus, cada dispositivo tiene asignado un identificativo que coincidirá con uno de los bits del bus de datos, de ahí que sólo se puedan conectar 8 dispositivos al bus. En el arbitraje, si dos o más dispositivos colisionan para obtener el control, siempre lo obtendrá el que tenga el identificativo de más peso.

El procedimiento que sigue un dispositivo para obtener el control del bus SCSI es como sigue:

- El dispositivo esperará primero a que ocurra un fase de bus libre. La fase de bus libre se detecta cuando $\overline{BSY}$ y $\overline{SEL}$ son ambas simultánea y continuamente falsas durante cierto tiempo (>400 ns).

- El dispositivo esperará otro retraso después de la detección de bus libre y antes de introducir ninguna señal.

- A continuación, el dispositivo ya puede arbitrar introduciendo la señal $\overline{BSY}$ y las correspondientes a su propio identificativo.

- Después de esperar un cierto tiempo, el dispositivo examinará el bus de datos. Si un bit ID SCSI de mayor prioridad es verdadero, entonces el dispositivo ha perdido el arbitraje y tiene que desactivar sus señales y volver a esperar a que ocurra una fase de bus libre. Si no hay un bit ID más prioritario, entonces el dispositivo ha ganado el arbitraje y activará $\overline{SEL}$. Los demás dispositivos que participaron en el arbitraje han perdido y desactivan sus señales $\overline{BSY}$ y su ID y volverán a esperar a que ocurra una fase de bus libre. El bit de paridad no es válido durante el arbitraje.

FASE DE SELECCIÓN

La fase de selección permite a un iniciador seleccionar un target con el propósito de iniciar alguna función. Se llega a esta fase cuando la fase de arbitraje ha sido ganada por un iniciador. Si es ganada por un target, se entra en la fase de RESELECCIÓN, existiendo por tanto una simetría entre ambas fases.

El dispositivo del bus SCSI que gana el arbitraje tiene $\overline{BSY}$ y $\overline{SEL}$ validadas; a continuación pondrá el bus de datos a un valor que será la OR de su bit y el bit ID del seleccionado. De esta forma el dispositivo seleccionado puede saber quien lo ha hecho. El dispositivo esperará un cierto tiempo y desactivará $\overline{BSY}$.

El seleccionado se enterará de la selección cuando $\overline{SEL}$ y su bit ID estén activos, y $\overline{BSY}$ inactiva. El seleccionado podrá examinar el bus de datos para determinar el ID del ganador del bus (a menos que se emplee la opción de no arbitraje, en cuyo caso se sabe quién le habla) y activará $\overline{BSY}$ para indicar al seleccionador (dispositivo que ganó el arbitraje) que se ha dado cuenta que ha sido seleccionado y por quien. Si hay más de dos bits en el bus de datos no se responderá a la selección.

Después de que el seleccionador detecta que $\overline{BSY}$ es verdadera, liberará $\overline{SEL}$ y podrá continuar el proceso. En la figura (3.19) se esquematiza el secuenciamiento de una fase de selección.

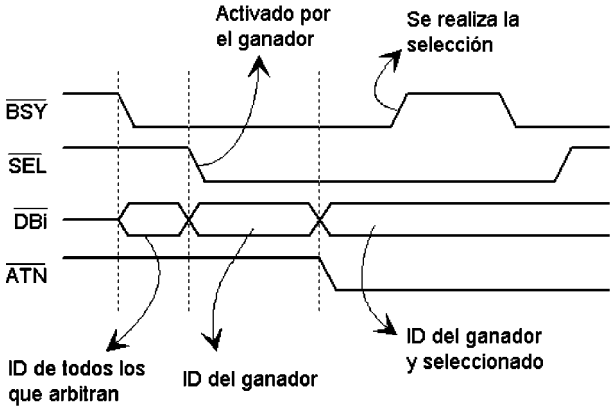


Fig. 3.19 Cronograma de una fase de selección tras la fase de arbitraje

FASE DE RESELECCIÓN

La reselección, al igual que la de selección, sólo puede ser usada en sistemas que tienen la fase de arbitraje implementada. Sólo se diferencia de la fase de selección en que el ganador ha sido un target y va a seleccionar a un iniciador.

Para avisar al iniciador que le selecciona un target, éste activa $I/\overline{O}$ en esta fase, y permanecerá activa hasta el final de la misma. Salvo esta señal, el resto es idéntico a la fase de selección.

3.9.4 Fases de transferencia de información

Los conceptos de entrada/salida están referidos aquí siempre al iniciador. Es decir cuando se habla de entrada (salida) de datos se entiende que los datos son enviados por el target (iniciador) y recibidos por el iniciador (target).

Las señales $C/\overline{D}$, $I/\overline{O}$ y $\overline{MSG}$ son usadas para distinguir entre las distintas fases de transferencia de información. El target establece estas tres señales y por tanto controla todos los cambios de una fase a otra. El iniciador se ve obligado a responder a ellas. Lo único que le está permitido es provocar una reinicialización o una condición de atención, que se verá más adelante en el apartado dedicado a las condiciones especiales del bus.

Las fases de transferencia de información emplean un protocolo *REQ/ACK* para controlar la comunicación. En cada *REQ/ACK* se transfiere un byte de información. Durante las fases de transferencia de información $\overline{BSY}$ quedará activa para que nadie interfiera la comunicación. En la tabla (3.5) se resume la codificación de las fases de transferencia de información.

$\overline{MSG}$	$C/\overline{D}$	$I/\overline{O}$	**Nombre de fase**	**Dirección**
1	1	1	Salida de Datos	Iniciador → Target
1	1	0	Entrada de Datos	Target → Iniciador
1	0	1	Comando	Iniciador → Target
1	0	0	Estado	Target → Iniciador
0	1	1	*	Reservado
0	1	0	*	Reservado
0	0	1	Salida de Mensaje	Iniciador → Target
0	0	0	Entrada de Mensaje	Target → Iniciador

Tabla 3.5 Tabla de fases de transferencia de información.

FASE DE COMANDOS

La fase de comandos permite al target pedir un comando al iniciador. El target activará la señal $C/\overline{D}$ y negará $I/\overline{O}$ y $\overline{MSG}$ durante los intercambios *REQ/ACK* de esta fase.

FASE DE DATOS

Es un término que engloba la fase de entrada de datos y la fase de salida de datos.

- La fase de entrada de datos permite al target pedir que el iniciador acepte datos desde el target. El target activará la señal $I/\overline{O}$ y negará las señales $C/\overline{D}$ y $\overline{MSG}$ durante esta fase.
- La fase de salida de datos permite al target pedir que los datos sean enviados desde el iniciador al target. El target negará las señales $C/\overline{D}$, $I/\overline{O}$ y $\overline{MSG}$ durante el protocolo *REQ/ACK* de esta fase.

FASE DE ESTADO

Permite al target enviar información de estado (normalmente de éxito o fracaso de la ejecución de un comando) al iniciador. El target activará $C/\overline{D}$ e $I/\overline{O}$ y negará la señal $\overline{MSG}$ durante esta fase. La información de estado siempre es de 1 byte. Al contrario de un mensaje que puede ser enviado en cualquier momento durante la fase de comando, el estado solo se envía cuando el comando se ha completado, se ha interrumpido o ha sido rechazado.

FASE DE MENSAJE

La fase de mensaje también engloba la de entrada de mensaje y la de salida de mensaje

- La fase de entrada de mensaje permite al target pedir que el iniciador acepte un mensaje, es decir, información de control no ejecutable ni que es consecuencia directa de la ejecución de un comando. El target activará $C/\overline{D}$, $I/\overline{O}$ y $\overline{MSG}$.
- La fase de salida de mensajes permite al target pedir que el mensaje sea enviado desde el iniciador al target. El target activará $C/\overline{D}$ y $\overline{MSG}$ y negará $I/\overline{O}$.

El iniciador puede solicitar esta fase provocando la condición de atención.

Como resumen de todas las fases, en la tabla (3.6) se muestra el origen de las señales en todas ellas. En la figura (3.20) se muestra el diagrama de transición entre las distintas fases del bus. Este esquema puede interpretarse como un diagrama de transición de estados y diseñar su circuito secuencial equivalente.

Nombre de fase	$\overline{BSY}$	$\overline{SEL}$	$C/\overline{D}, I/\overline{O}, \overline{MSG}, \overline{REQ}$	$\overline{ACK}$, $\overline{ATN}$	$\overline{DB0}-\overline{DB7}, \overline{DBP}$
Bus libre	Nadie	Nadie	Nadie	Nadie	Nadie
Arbitraje	Todos	Ganador	Nadie	Nadie	ID
Selección	I y T	Ganador	Nadie	Iniciador	Iniciador
Reselección	I y T	Target	Target	Iniciador	Target
Comando	Target	Nadie	Target	Iniciador	Iniciador
Entrada de datos	Target	Nadie	Target	Iniciador	Target
Salida de Datos	Target	Nadie	Target	Iniciador	Iniciador
Estado	Target	Nadie	Target	Iniciador	Target
Entrada de mensaje	Target	Nadie	Target	Iniciador	Iniciador
Salida de mensaje	Target	Nadie	Target	Iniciador	Target

Tabla 3.6 Origen de las señales en cada una de las fases.

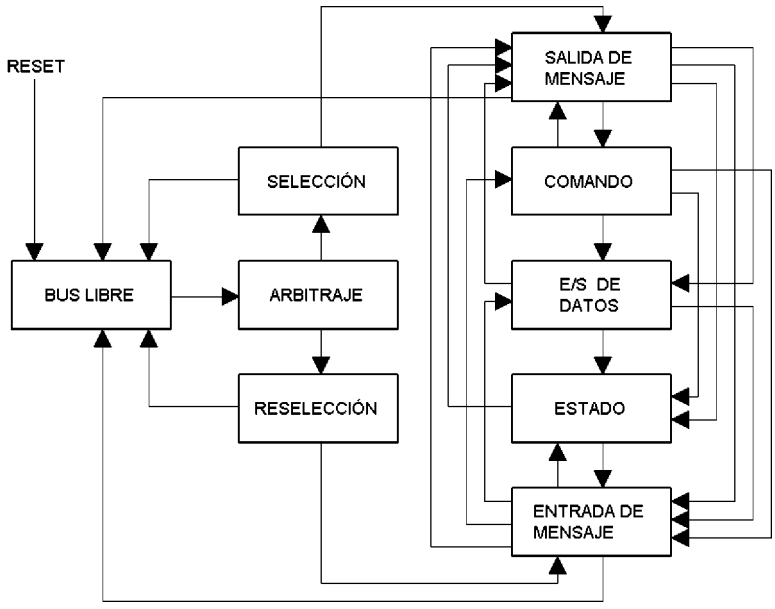


Fig. 3.20 Transición de fases del bus SCSI

3.9.5 Variantes síncrona y ancha

Existe la posibilidad de un modo de transferencia en la cual el protocolo *REQ/ACK* se modifica para que sea más rápido. Este modo es opcional y se le conoce como SCSI rápido (Fast SCSI) o modo síncrono.

Para poder utilizarla, ambos dialogantes se deben haber puesto previamente de acuerdo en una serie de cosas usando mensajes. Este modo consiste básicamente en que no se espera el *ACK* del iniciador para continuar con el siguiente dato. Como existe acuerdo de anchura mínima y separación mínima entre pulsos y número máximo de $\overline{REQ}$ sin esperar *ACK*, se envía un tren de pulsos y bytes sin esperar contestación y se van contando las contestaciones recibidas. Si al final coinciden se podrá continuar. Con este modo se consigue subir la velocidad de transferencia de 1.5 Mbytes/s hasta 4 Mbytes/s.

Existe una segunda variante del interfaz SCSI conocida como Wide-SCSI. Este interfaz es formalmente idéntico al SCSI normal, pero en lugar de trabajar con un bus de 8 bits de datos y uno de paridad, trabaja con 16 bits de datos y 2 de paridad o 32 bits de datos y 4 de paridad (1 bit de paridad para cada byte de la doble palabra de 16 o 32 bits).

Por compatibilidad con el SCSI normal, un segundo cable es añadido para llevar todas las señales adicionales. A este segundo cable se le llama cable B para distinguirlo del cable normal que en los sistemas Wide SCSI se denomina cable A. Este segundo cable (o cable B), consta de 68 hilos con sus correspondientes conectores. Actualmente, el comité de normalización encargado del bus SCSI propone sustituir estos dos cables por uno para transferencias de 8 o 16 bits y otro alternativo para transferencias a 32 bits con lo que el segundo cable queda eliminado en las últimas versiones de la norma.

En un mismo sistema se pueden mezclar dispositivos SCSI normales con dispositivos Wide-SCSI. Como las señales adicionales van por otro cable independiente, puede suceder que su longitud difiera de las del bus SCSI normal, por lo que en este cable adicional se han incluido dos señales nuevas, aparte de las ya mencionadas de datos y paridad: $\overline{REQB}$ y $\overline{ACKB}$ que tienen el mismo significado que $\overline{REQ}$ y $\overline{ACK}$ en el cable A, pero referidas a las señales que van por el cable B. Las señales $\overline{REQ}$ y $\overline{REQB}$ solicitan ('request') algún tipo de atención y $\overline{ACK}$ y $\overline{ACKB}$ envían el reconocimiento de llegada del dato ('acknoledgement'). Con la eliminación del segundo cable, esto ya no es necesario.

3.9.6 Condiciones especiales del bus

El bus SCSI tiene dos condiciones asíncronas y que por lo tanto quedan fuera del esquema de fases síncronas descrito anteriormente y esquematizado en la figura (3.20): la condición de ATENCIÓN y la condición de RESET. Estas condiciones provocan que el dispositivo lleve a cabo algunas acciones peculiares y se altere la secuencia de fases.

CONDICIÓN DE ATENCIÓN

Permite a un iniciador informar al target de que tiene un mensaje preparado, El target debe responder con la fase de salida de mensaje.

El iniciador crea una condición de atención activando $\overline{ATN}$ en cualquier momento, excepto en las fases de arbitraje y de bus libre. El iniciador niega $\overline{ATN}$ durante el último intercambio *REQ/ACK*.

CONDICIÓN DE RESET

Se utiliza para realizar un reset. Prevalece sobre todas las demás fases y condiciones. Cualquier dispositivo puede provocar un reset simplemente activando la señal $\overline{RST}$, el estado de todas las señales del bus SCSI distintas de $\overline{RST}$ mientras esta señal está activa no está definido.

3.10 LOS INTERFACES CENTRONICS E IEEE-1284

3.10.1 Introducción y necesidad de la norma

Uno de los interfaces más extendidos en todos los sistemas computadores, e implementados en gran número de dispositivos periféricos es el interfaz de tipo paralelo Centronics. En algunos casos se le conoce simplemente como puerto paralelo, aunque existen otros interfaces de tipo paralelo distintos y ampliamente difundidos, como SCSI o GPIB. El tipo de dispositivos que han incluido, y siguen incluyendo este tipo de interfaz son los dispositivos de salida, que podríamos catalogar como lentos, fundamentalmente impresoras y plotters. Esto es debido a que la sencillez del hardware de este interfaz, no permitía grandes posibilidades de gestión de la comunicación, y por lo tanto, todo el trabajo debía recaer sobre un programa de control. Esto hace que el protocolo sea tremendamente lento, si tenemos en cuenta que se trata de un interfaz implementado con un bus paralelo de 8 bits. Actualemente el protocolo se ha mejorado con una mayor velocidad lo que permite ampliar el tipo de periféricos conectables mediante este tipo de interfaz.

Las características fundamentales de este interfaz, son que se trata de un interfaz con bus de datos paralelo (8 líneas) un bus de control (4 líneas) y 5 líneas de estado, cuyos significados están intimamente relacionados con las impresoras. Es fundamentalmente unidireccional, y punto a punto, es decir que solo puede comunicar un ordenador con un periférico, ya que no implementa la posibilidad de conectar más dispositivos simultáneamente, por lo que no se puede emplear el término de 'bus ' para referirnos a el como sí sucede con SCSI o GPIB.

Cuando IBM introdujo en el mercado el PC, a mediados de 1981, incluyó el puerto paralelo como una alternativa rápida al interfaz serie, que era el más habitual para trabajar con impresoras, y terminales de datos. Este aumento de velocidad se debía al hecho de que Centronics es un interfaz paralelo en lugar de serie, lo que permitía enviar un byte (8 bits) completo de datos cada vez. Por contra tiene el inconveniente de no permitir distancias elevadas entre el ordenador y el dispositivo periférico. Las distancias máximas, asi como otras características no fueron recogidas en ninguna norma estándar, por lo que los fabricantes lo han implementado con ciertas variaciones, que si bien no son muy importantes, si son muy numerosas. La influencia de estas numerosas variantes ha sido relativamente pequeña debido fundamentalmente al bajo nivel de requisitos que se le ha exigido a este tipo de interfaz, como lo demuestra el hecho de que se haya empleado casi exclusivamente para enviar datos a impresoras u otros dispositivos de salida con tiempos de respuesta elevados, y que habitualmente están próximos al sistema computador.

No obstante, cuando comenzó a popularizarse el uso de los ordenadores portátiles surgió la necesidad de mejorar este interfaz para hacerlo útil en el uso de otros tipos de dispositivos de mayor velocidad y nivel de requerimientos. Esto fue debido a que, mientras que en un sistema central, no es muy problemático añadir un interfaz específico, en un sistema portátil, si que lo es. Por este motivo se empezaron a diseñar dispositivos periféricos que empleasen el puerto paralelo Centronics, ya que está presente en casi todos los sistemas, para añadir cualquier otro tipo de periféricos, como por ejemplo discos duros, CD-ROM, conexiones a red local, etc..

Esto provocó que el interfaz entrara en una fase de mejoras, donde cada fabricante introducía las suyas propias. Estas mejoras se encaminaban a dos objetivos fundamentales: aumentar la velocidad de transferencia, y conseguir un interfaz verdaderamente bidireccional ya que estas eran las dos grandes limitaciones del interfaz. Estas mejoras individualistas no tardaron en traer como consecuencia problemas de compatibilidad, lo que provocó que varias empresas

como Lexmark, IBM, Texas Instrument y otras, fuertemente introducidas en el campo de la microinformática y por extensión, en la informática portátil, hicieran frente común y crearan la Network Printing Alliance (NPA). Esta asociación de fabricantes definió una serie de parámetros para permitir una completa comunicación entre impresoras y sistemas computadores. Un requisito que consideró esencial fue el de dotar al interfaz de un modo verdaderamente bidireccional, lo que permitiría ampliar considerablemente el tipo de dispositivos que se podrían conectar. Una restricción era que la nueva interfaz debía ser compatible con la interfaz Centronics existente, debido al gran número de sistemas que lo estaban empleando. Prácticamente todos los ordenadores disponían de uno ya que su sencillez hacía que el coste del mismo fuera muy bajo. Además la gran mayoría de impresoras en funcionamiento no disponían de ningún otro tipo de conexión por lo que un cambio significativo las dejaría 'desconectadas'.

Esta asociación propuso al IEEE (Institute of Electrical and Electronic Engeneering) la creación de un comité para desarrollar un nuevo estándar para el 'puerto paralelo' que lo hiciese de alta velocidad y bidireccional, llegando al menos a 1Mbyte por segundo de velocidad de transferencia en ambas direcciones. Este comité fue el IEEE-1284, por lo que el nuevo estándar para el puerto paralelo se conoce como IEEE-1284.

Este estándar define cinco modos de funcionamiento, el más simple de los cuales corresponde con el interfaz Centronics convencional y recibe el nombre de modo compatible (Compatibility mode). El resto son el modo Nibble, de Byte y los modos de altas prestaciones EPP (Enhanced Parallel Port) y ECP (Extended Capability Port). Estos dos últimos, elevan considerablemente la velocidad de transferencia, ya que gestionan el protocolo por hardware. En el interfaz Centronics convencional, es el propio software el que se encarga de gestionar completamente la comunicación, activando y desactivando por programa todas las señales de control. A continuación se dará una breve descripción de los distintos modos. Hay que señalar que la norma designa con distintos nombres a las señales en cada uno de los modos ya que su significado varía de un modo a otro.

Grupo	Señal SPP	E/S	Descripción
Control	nSTROBE	S	Activa en baja. Indica que hay un dato válido en las líneas de datos.
	nAUTOFEED	S	Activa en baja. Indica a la impresora que añada un avance de línea de forma automática por cada retorno de carro
	nSELECTIN	S	Activa en baja. Usada para indicar a la impresora que está seleccionada.
	nINIT	S	Activa en baja. Reinicializa la impresora. Reinicialización hardware.
Estado	nACK	E	Un pulso hacia abajo indica que el carácter fue recibido por la impresora.
	BUSY	E	En alta indica que la impresora está ocupada y no puede admitir más datos.
	PE	E	La impresora está sin papel (Paper Empty)
	SELECT	E	En alta indica que la impresora está lista (on line)
	nERROR	E	Indica que la impresora está en un estado e error.
Datos	DATA (1:8)	S	8 líneas de datos. En el modo SPP es únicamente de salida.

Tabla 3.7 Señales que intervienen en el Modo Centronics estándar.

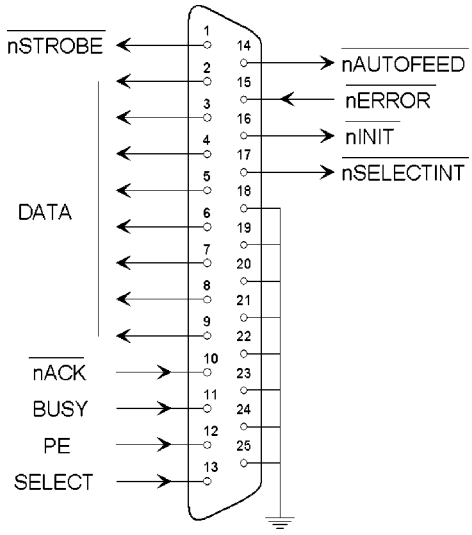


Fig. 3.21 Conector del interfaz Centronics convencional.
Las flechas indican el sentido de la información respecto del ordenador

3.10.2 Modo compatible (Centronics convencional)

El modo compatible, recibe también del nombre de puerto paralelo estándar (Standard Parallel Port ó SPP). En la Tabla (3.7) se muestran las distintas señales que intervienen, asi como su significado y si son de entrada o salida al ordenador. En esta tabla se puede comprobar como la interfaz Centronics estaba muy orientada al manejo de impresoras, por los significados que tienen las señales de control y de estado. La descripción de estas señales y de las que aparecerán en las tablas correspondientes a los distintos modos se refieren a la fase de transmisión de datos, pero algunas de estas se utilizan para el cambio de modo o como señales de estado y control adicionales. En la figura (3.21) se muestra la distribución de estas señales sobre el conector.

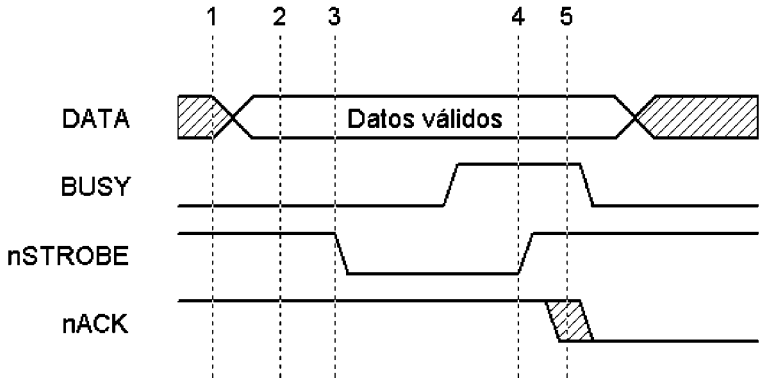


Fig. 3.22 Cronograma del ciclo de transferencia en el modo compatible

En la figura (3.22) se muestra el diagrama de tiempos de un ciclo de transferencia de datos en el modo compatible, sobre el que se pueden distinguir cinco fases:

1.- Escritura de los datos en el registro de datos de salida
2.- El programa lee el registro de estado, para ver si la impresora está ocupada (señal Busy).
3.- Si no está ocupada, escribe en el registro de control para dar la señal de disparo (STROBE)
4.- Nuevamente se escribe en el registro de control para cancelar la señal de disparo y se prepara para enviar un nuevo dato.
5.- La impresora reconoce la llegada del dato.

Como puede verse, el envío de un byte de datos requiere al menos cuatro instrucciones, esto hace que la velocidad de transferencia sea del orden de 150 Kbytes por segundo. Ha de tenerse en cuenta que al ser un protocolo con una gran componente software, será muy dependiente de la velocidad de la máquina. Esta velocidad es suficiente para comunicar con la mayoría de las impresoras de matriz de agujas y las de tecnología láser antiguas, pero no es suficiente para las impresoras láser de nueva generación y mucho menos para adaptadores de red local, discos extraíbles, etc. Aunque los ordenadores modernos pueden alcanzar velocidades muy superiores no deja de ser una fuerte restrcción la dependencia de la velocidad de transmisión del tipo de procesador, además de sobrecargar la CPU con una tarea que debería realizarse de forma autónoma.

Algunos fabricantes han implementado un modo que usa una FIFO para transferir los datos en este modo a más alta velocidad. Este modo se conoce como Centronics rápido (Fast Centronics) o puerto paralelo con modo FIFO (Parallel Port FIFO mode). Cuando se emplea este modo, los datos se escriben en la FIFO y el hardware del controlador es el que se encarga de gestionar el protocolo aumentando la velocidad hasta unos 500Kbytes por segundo. Sin embargo, este modo no está contemplado en la norma IEEE-1284.

Señal SPP	Nombre en el modo Nibble	E/S	Descripción
nSTROBE	nSTROBE	S	No usado para la transferencia inversa
nAUTOFEED	HostBusy	S	En baja indica que el ordenador está listo para recibir un nibble. En alta indica que el nibble ha sido recibido
nSELECTIN	1284Active	S	En alta cuando está en un modo 1284
nINIT	nINIT	S	No usado para la transferencia inversa
nACK	PtrClk	E	En baja indica que hay un nibble válido. Se pone en alta en respuesta al flanco de subida de la señal HostBusy
BUSY	PtrBusy	E	Usado para el bit 3, 7
PE	AckDataReq	E	Usado para el bit 2, 6
SELECT	Xflag	E	Usado para el bit 1, 5
nERROR	nDataAvail	E	Usado para el bit 0, 4
DATA (1:8)	No se usan		

Tabla 3.8 mostrando las señales empleadas en el modo Nibble.

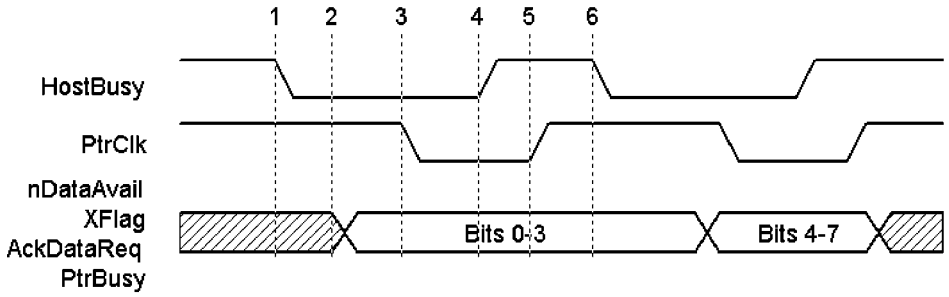


Fig. 3.23 Diagrama de tiempos durante una transferencia en el modo Nibble.

3.10.3 Modo nibble

Este modo es el más común para retornar datos desde la impresora al ordenador, e incluso para conectar dos ordenadores. Combinado con el modo compatible, permite una comunicación bidireccional sencilla. El estándar proporciona cinco líneas de información desde el periférico (normalmente una impresora) al ordenador que son usadas como señales de estado del dispositivo. Usando estas líneas, un periférico puede enviar un byte dividido en dos envíos de 4 bits (1 Nibble)

cada uno. El inconveniente de este modo, es que el ordenador debe formar nuevamente el byte leyendo dos veces en el registro de estado. La tabla (3.8) muestra las señales que intervienen en este modo y su correspondencia con las líneas del modo compatible (SPP).

En la figura (3.23) se muestran los puntos más relevantes durante la transferencia de un byte en el modo Nibble:

1.- El ordenador habilita la señal HostBusy para indicar que puede comenzar la transferencia.
2.- El periférico responde colocando el primer nibble en las líneas de estado.
3.- El periférico señala que los datos son válidos bajando la señal PtrClk
4.- El ordenador pone en alta la señal HostBusy para indicar que ha recibido el nibble e indica que todavía no está preparado para el siguiente.
5.- El dispositivo pone PtrClk en alta para indicar al host que hay un nuevo nibble preparado.
6.- Se repiten nuevamente las fases 1-5 para el segundo nibble.

El modo nibble, para la transmisión inversa (del periférico al ordenador) necesita muchas más instrucciones software que el modo compatible por lo que tiene una limitación de unos 50Kbytes por segundo de velocidad de transferencia. La principal ventaja de este modo es que no presupone la existencia de ninguna circuitería especial, y por lo tanto está disponible en todos los puertos paralelos Centronics. Este modo es útil en dispositivos que no requieren enviar muchos datos al ordenador como sucede con las impresoras, pero resulta inaceptable para adaptadores de red, CD-ROM, etc.

3.10.4 Modo byte

Al diseñar IBM su serie PS/2, modificó las etapas de excitación de las líneas de datos del puerto paralelo para permitir que pudiesen funcionar tanto para entrada como para salida de datos. Esto permite al dispositivo enviar un byte completo de datos empleando las mismas líneas por las que recibe los datos. Al poder enviar los bytes completos sin necesidad de partirlos en dos nibbles para enviarlos multiplexados en tiempo, la velocidad en modo inverso (del periférico al ordenador) se eleva considerablemente, igualándose a las velocidades de transferencia en sentido directo (del ordenador al periférico).

En la tabla (3.9) se muestran las señales que intervienen junto con una breve descripción de su significado, asi como su correspondencia con las equivalentes al modo SPP y si son de entrada, salida o bidireccionales respecto al ordenador.

Señal SPP	Nombre en el modo Byte	E/S	Descripción
nSTROBE	HostClk	S	Es una señal de reconocimiento da un pulso bajo para indicar que el byte ha sido recibido
nAUTOFEED	HostBusy	S	En estado bajo indica que el ordenador está preparado para recibir un byte. Pasa a estado alto para indicar que el byte ha sido recibido
nSELECTIN	1284Active	S	En alto indica que está en un modo 1284
nINIT	nINIT	S	No usado. Debe estar en alta
nACK	PtrClk	E	En estado bajo indica que hay datos válidos en las líneas de datos. Pasa a alta en respuesta al flanco de subida de HostBusy
BUSY	PtrBusy	E	Línea de estado.
PE	AckDataReq	E	No usado
SELECT	Xflag	E	No usado en el modo Byte
nERROR	nDataAvail	E	Pasa a estado bajo para indicar que el byte está listo
DATA (1:8)	DATA	E/S	8 bits de datos del dispositivo al computador

Tabla 3.9 Señales que intervienen en el modo Byte.

La figura (3.24) muestra la transferencia en el modo byte, en la que se pueden distinguir los siguientes sucesos:

1.- El ordenador indica que puede recibir datos poniendo HostBusy en estado bajo
2.- El periférico responde colocando los datos en las líneas correspondientes
3.- El dispositivo indica que los datos son válidos mediante un pulso a través de PtrClk
4.- El ordenador pasa HostBusy a estado alto para indicar que ha recibido el dato y que todavía no está preparado para el siguiente envío.
5.- El periférico pasa CtrClk a estado alto como señal de reconocimiento al host y este responde bajando la señal HostClk

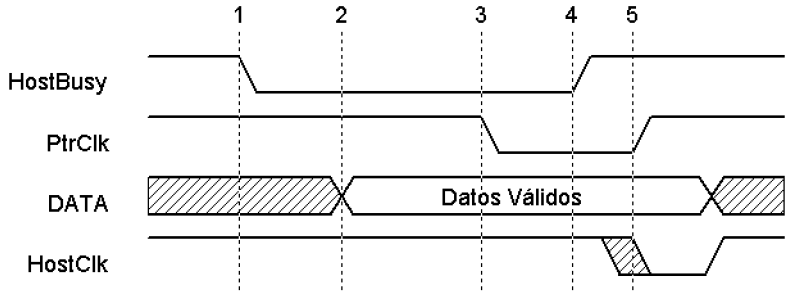


Fig. 3.24 Diagrama de tiempos durante una transferencia en el modo Byte

3.10.5 Modo EPP (Enhanced Parallel Port)

Este modo de funcionamiento fue desarrollado inicialmente por Intel, Xircom y Zenith Data Systems, para proporcionar un puerto paralelo de altas prestaciones y compatible con el Centronics convencional. Este protocolo fue implementado por Intel en el juego de circuitos integrados de soporte de la familia 386SL (integrado 82360 I/O).

El modo EPP ofrece grandes posibilidades sobre el interfaz estándar y fue adoptado rápidamente por numerosos fabricantes. Este protocolo, proporciona cuatro tipos de transferencia de datos:

1.- Ciclo de escritura de datos
2.- Ciclo de lectura de datos
3.- Ciclo de escritura de dirección
4.- Ciclo de lectura de dirección

Los ciclos de dirección están pensados para pasar direcciones, números de canal, comandos o información de control, y los de datos, para la transferencia de datos propiamente dicha. En la tabla se muestran las señales del modo EPP, y como en los modos anteriores, su correspondencia con el modo SPP.

Una de las características más sobresalientes de este modo es que la transferencia del byte está gestionada por el hardware del propio interfaz, con lo que el envío de un byte se reduce a una simple instrucción de salida. Esto permite elevar la velocidad de transferencia desde los 500Kbytes/s a los 2Mbytes por segundo. Los modos Nibble, Byte, EPP y ECP utilizan la técnica de protocolo con interbloqueo de forma que la transferencia se realiza siempre a la velocidad del elemento más lento, ya sea el periférico o el ordenador. Esto hace que tengamos una velocidad de transferencia adaptativa que resulta transparente tanto para el ordenador como para el dispositivo periférico.

Señal SPP	Nombre en el modo EPP	E/S	Descripción
nSTROBE	nWRITE	S	En baja indica una operación de escritura. En alta operación de lectura.
nAUTOFEED	nDATASTB	S	Activa en baja. Indica que una operación de lectura o escritura de datos está en proceso.
nSELECTIN	nADDRSTB	S	Activa en baja. Indica que una operación de lectura o escritura de dirección está en proceso.
nINIT	nRESET	S	Activa en baja. Reinicializa al dispositivo.
nACK	nINTR	E	Interrupción. Utilizada por el periférico para producir una interrupción en el ordenador y solicitar así su atención.
BUSY	nWAIT	E	En baja indica que esta listo para comenzar un ciclo. En alta indica que está listo para terminar.
PE	Definidas por el usuario	E	Estas tres señales pueden ser utilizadas de forma diferente por cada periférico.
SELECT		E	
nERROR		E	
DATA (1:8)	AD	E/S	Para enviar o recibir tanto los datos propiamente dichos como las direcciones.

Tabla 3.10 Señales del protocolo EPP

Esta característica de interbloqueo hace referencia a que cada una de las señales de control es reconocida por el lado opuesto del interfaz. Esto se ve más claro si nos fijamos en la transferencia de un dato en el modo EPP, como se muestra en la figura (3.25).

1.- El programa ejecuta una instrucción de salida, representada por una señal *IOW* en el bus del sistema.
2.- La línea nWrite pasa a estado bajo para indicar que se trata de una escritura, y los datos pasan a la salida.
3.- Se da la señal de disparo (bajando nDataStrobe) para indicar que los datos han sido colocados, y permanecerá así hasta que nWAIT pase a estado bajo.
4.- El puerto del ordenador (no un programa) espera que el dispositivo cambie el estado de nWAIT.
5.- La señal de disparo es retirada (pasa nuevamente a alta) y el ciclo termina.
6.- Finaliza el ciclo de ejecución de la instrucción de salida.
7.- nWAIT vuelve a estado bajo par indicar que el siguiente ciclo puede comenzar.

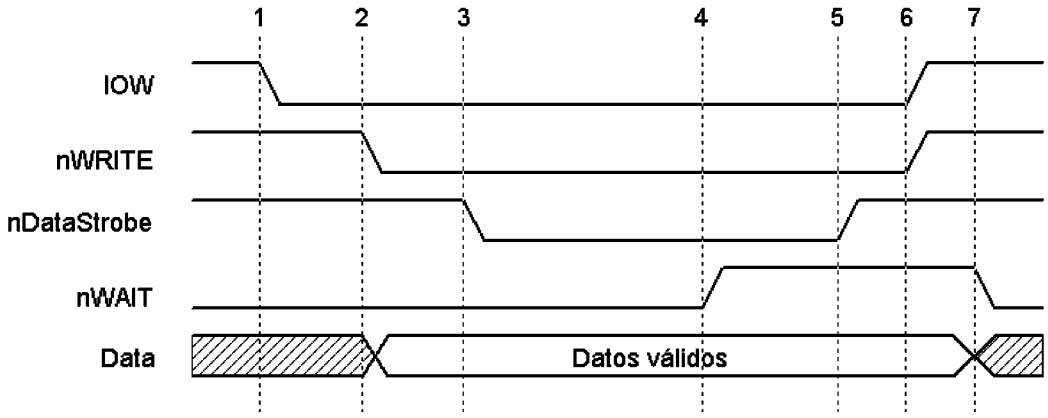


Fig. 3.25 Diagrama de tiempos durante una operación de escritura de datos en modo EPP

De esta manera el periférico puede controlar el tiempo de set-up (tiempo entre 3 y 4) que necesita para su correcto funcionamiento. Esta característica de interbloqueo, también hace que la transferencia sea independiente de la longitud del cable. Sin embargo, la longitud del cable puede hacer que la comunicación se haga lenta debido a que los flancos de subida o bajada queden muy suaves y también que se produzcan errores debido a que la contaminación con ruido será mayor cuanto mayor sea la longitud del cable. La tabla (3.10) muestra las señales en este modo.

El diagrama de tiempos para un ciclo de lectura, sería idéntico pero con la señal nWRITE en estado alto en lugar de bajo. Para los modos de escritura y lectura de direcciones, tendríamos unos diagramas de tiempo idénticos con la única salvedad de que ahora la señal de disparo la dará la señal nADDRSTB en lugar de nDATASTB. Con el modo EPP se proporcionan nuevos registros, que permiten incluso transferencias de 16 o 32 bits. En la tabla (3.11) se muestran estos registros.

Nombre del puerto	Offset	Modo	R/W	Descripción
SPP Data Port	0	SPP/EPP	W	Puerto de salida de datos estándar. Sin auto-disparo -> necesita dar el disparo escribiendo en el registro de control.
SPP Status Port	1	SPP/EPP	R	Registro de estado
SPP Control Port	2	SPP/EPP	W	Registro de control del protocolo
EPP Address Port	3	EPP	R/W	Realiza una lectura o escritura de dirección con interbloqueo (auto-disparo)
EPP Data Port	4	EPP	R/W	Realiza una lectura o escritura de dato con interbloqueo (auto-disparo)
No definido	5-7	EPP	R/W	Los emplean algunas implementaciones para permitir transferencias de 16 o 32 bits

Tabla 3.11 Registros que utiliza el protocolo EPP.
Se muestran también cuales son compartidos con el modo Centronics convencional o SPP.

3.10.6 Modo ECP (Extended Capability Port)

Este protocolo fue propuesto por Hewlett Packard y Microsoft como un modo avanzado para la comunicación con impresoras gráficas y escáner fundamentalmente. Una característica de estos dos tipos de dispositivos, es que normalmente precisan la transferencia de grandes cantidades de datos entre los que hay una gran redundancia. Este hecho sugiere el empleo de alguna técnica de compresión. El método escogido es el RLE (Run Length Encoding). Otras características que incorpora este protocolo es una FIFO en cada extremo del interfaz y acceso DMA.

Este modo también incorpora direccionamiento, aunque es conceptualmente diferente al del modo ECP. Este modo está orientado a manejar varias unidades lógicas dentro de un mismo dispositivo. Esto permite que se tengan un fax, una impresora y un módem conectados como un único dispositivo y se direccionen lógicamente. De esta forma, se pueden recibir datos del módem mientras la impresora está procesando datos. En el modo compatible, si la impresora está ocupada, activa la señal de BUSY y el interfaz queda paralizado, sin embargo con el direccionamiento, basta con direccionar otro dispositivo para realizar alguna transferencia con él y esperar a más adelante para continuar con el trabajo de impresión. Hay que tener en cuenta que estos distintos periféricos constituyen un solo equipo que se conecta mediante un único cable a la unidad central.

Como en el resto de modos IEEE-1284, el protocolo ECP redefine las señales SPP dándoles un nuevo significado. Al contrario que con los otros modos, cuando se propuso el estándar para este modo no sólo se proponía el significado e interpretación de las señales, sino también varios registros para permitir la comunicación. En la tabla (3.12) se muestran las señales en el modo ECP y en la tabla (3.13) los registros que emplea.

Señal SPP	Nombre en el modo ECP	E/S	Descripción
nSTROBE	HostClk	S	Utilizado con PeriphAck para transferir datos en sentido directo.
nAUTOFEED	HostAck	S	Proporciona el estado comando/dato en sentido directo.
nSELECTIN	1284Active	S	En alta indica que se trabaja en algún modo IEEE-1284
nINIT	nReverseReq	S	Pasa a estado bajo para indicar el sentido inverso.
nACK	PeriphClk	E	Utilizado con HostAck para transferir datos en sentido inverso.
BUSY	PeriphAck	E	Utilizado con HostClk para transferir datos o direcciones en sentido directo. Proporciona el estado comando/dato en sentido inverso.
PE	nAckReverse	E	Pasa a baja como reconocimiento a nReverseReq
SELECT	Xflag	E	Bandera de extensión.
nERROR	nPeriphReq	E	En estado bajo indica que un dato está listo para ser enviado del periférico al ordenador.
DATA (1:8)	Data	E-S	Datos bidireccionales.

Tabla 3.12 Señales en el modo ECP.

El protocolo ECP proporciona dos tipos de transferencia de información, tanto en sentido directo, como inverso, que son el ciclo de datos y el de comandos. En la figura (3.26) se muestra el diagrama de tiempos correspondiente a dos ciclos de transferencia en sentido directo (uno de datos y otro de comando) con los siguientes eventos:

1.- El ordenador coloca un byte sobre las líneas de datos, y señala que se trata de un dato (en lugar de un comando) poniendo en alta la señal HostAck.
2.- El ordenador pasa a baja la señal HostClk para indicar que el dato es válido.
3.- El periférico envía el reconocimiento al ordenador poniendo PeriphAck en alta.
4.- El ordenador pone en alta HostClk. Este flanco debería ser usado por el periférico para tomar el dato.
5.- El periférico pone PeriphAck en baja para indicar que está preparado para el siguiente byte.
6.- El ciclo se repite, pero ahora el byte enviado corresponde a un comando, ya que HostAck está en baja.

Nombre	R/W	Modo ECP	Función
Data	R/W	000-001	Registro de datos
ecpAfifo	R/W	011	Dirección de la FIFO ECP
dst	R/W	todos	Registro de estado
dcr	R/W	todos	Registro de control
cfifo	R/W	010	FIFO del puerto paralelo
ecpDfifo	R/W	011	FIFO de datos ECP
tfifo	R/W	110	Test FIFO
cnfgA	R	111	Registro de configuración A
cnfgB	R/W	111	Registro de configuración B
ecr	R/W	todos	Registro de control ampliado

Tabla 3.13 Registros empleados en el modo ECP

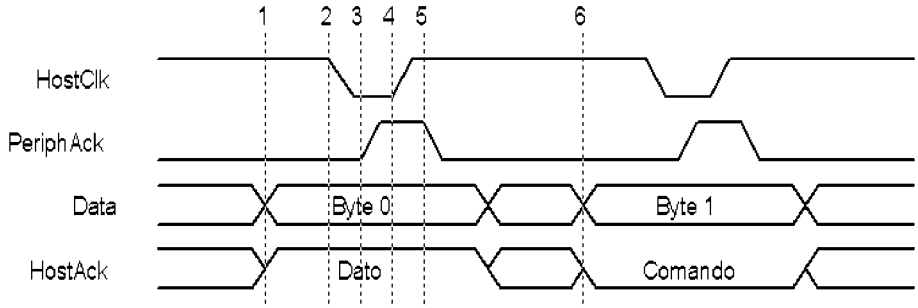


Fig. 3.26 Envío de un dato y luego un comando en sentido directo.

En el protocolo EPP el software de control puede enviar o recibir datos desde el periférico, sin mayor problema, sin embargo, en el modo ECP el cambio de dirección debe ser negociado. Esto se muestra en la figura (3.27) donde puede verse un diagrama de tiempos similar, pero ahora corresponde a un envío en sentido inverso. Los puntos relevantes de esta transferencia son:

1.- El ordenador solicita invertir el canal de comunicaciones poniendo nReverseReq en estado bajo.
2.- El periférico contesta que está listo para la comunicación inversa bajando nAckReverse.
3.- El periférico coloca el byte sobre las líneas de datos e indica que se trata de un dato (no un comando) poniendo PeriphAck en estado alto.
4.- El periférico pone PeriphClk en baja para indicar un dato válido.
5.- El ordenador envía la contestación de reconocimiento poniendo HostAck en alta.
6.- El periférico pone PeriphClk en alta. Este flanco debe ser usado por el ordenador para tomar el dato.
7.- El ordenador pone HostAck en baja para indicar que está preparado para el siguiente byte.
8.- El ciclo se repite, pero en este caso se trata de enviar un comando porque PeriphAck esta en estado bajo.

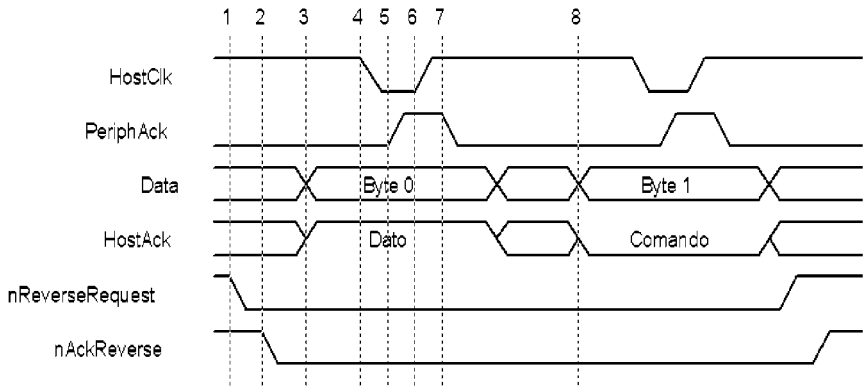


Fig. 3.27 Envío de un dato y luego un comando en sentido inverso.

3.10.7 Negociación de modo

Hasta ahora hemos descrito los distintos modos del interfaz IEEE-1284. Los periféricos no tienen porque tener implementados todos los modos anteriores, con lo que se hace necesario un método para determinar cuales son las posibilidades del dispositivo conectado al puerto y que permita al ordenador establecer el modo apropiado de funcionamiento. Para solventar el problema, se introdujo el concepto de negociación de modo. Mediante el proceso de negociación, un ordenador establece comunicación con el periférico para conocer los modos que éste implementa y elegir uno de ellos.

La negociación es una secuencia de eventos que debe realizarse a través del interfaz, entre el ordenador y el periférico, pero que no debe tener efecto sobre un periférico antiguo, ajeno a los nuevos modos del estándar. Es decir, un dispositivo más viejo que solo soporta el modo compatible, correspondiente al modo Centronics convencional, no responderá al proceso de negociación.

El byte de extensión se utiliza durante la negociación para que el periférico entre en un modo determinado. La tabla (3.14) muestra los valores permitidos para este byte. La señal Xflag es utilizada por el periférico para indicar que el modo solicitado está disponible. Esta señal estará el estado alto para todos los modos, salvo para el modo Nibble que está presente, como ya se señaló, en todos los dispositivos, incluidos los más antiguos. El bit de enlace (Request Extensibility Link) se utiliza como una forma de contemplar posibles ampliaciones futuras y no se usa.

Bit	**Descripción**	**Valores válidos** (7654 3210)
7	Request Extensibility Link	1000 0000
6	Request EPP Mode	0100 0000
5	Request ECP Mode with RLE	0011 0000
4	Request ECP Mode without RLE	0001 0000
3	Reservado	0000 1000
2	Request Device ID	Modo de retorno de datos: Nibble Mode 0000 0100 Byte Mode 0000 0101 ECP Mode without RLE 0001 0100 ECP Mode with RLE 0011 0100
1	Reservado	0000 0010
0	Byte Mode	0000 0001
ninguno	Nibble Mode	0000 0000

Tabla 3.14. Valores del byte de extensión.

1.- El ordenador coloca en las líneas de datos el byte de extensión para solicitar un determinado modo.
2.- Una vez hecho esto, pone nSelectIn en alta y nAutoFeed en baja para indicar que comienza una secuencia de negociación. Recuérdese que en el puerto Centronics convencional nSelectIn en alta significa que la impresora no está seleccionada, con lo que un dispositivo antiguo no se dará por aludido y no responderá a los siguientes eventos.
3.- Un periférico IEEE-1284 responderá poniendo nAck en estado bajo, y nError, PE y Select en alto. Un dispositivo que no sea IEEE-1284 no responderá.
4.- El ordenador establece nStrobe en estado bajo como señal de disparo indicando al dispositivo que el byte de extensión está disponible sobre las líneas de datos.
5.- El ordenador ahora, sube a estado alto tanto nStrobe como nAutoFeed para indicar al periférico que lo ha reconocido como dispositivo IEEE-1284.
6.- El periférico responde bajando PE; Y pone nError en estado bajo si el periférico dispone de canal inverso, e indica que el modo no está disponible poniendo Select en estado bajo.
7.- Por último el periférico pone nAck en estado alto para indicar que la secuencia de negociación ha terminado y que todas las señales están en el estado solicitado, si éste es soportado.

En la figura (3.28) se muestra el diagrama de tiempos de la fase de negociación:

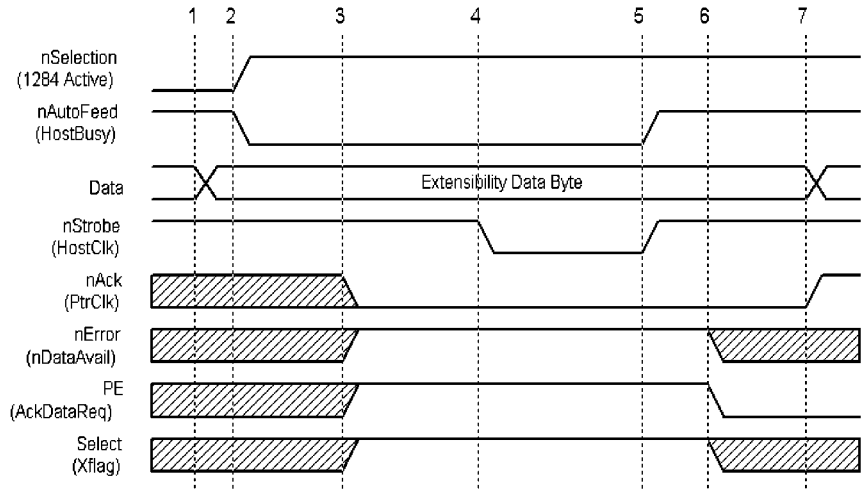


Fig. 3.28. Diagrama de tiempos correspondiente a una negociación de modo IEEE-1284.

3.11 BUS IEEE-488

La interfaz IEEE-488 es el resultado de la normalización de un bus propietario de la compañía Hewlett-Packard. Esta empresa comenzó el desarrollo del bus en 1965, para la interconexión de los instrumentos de laboratorio de la misma. El objeto era un bus de propósito general, destinado a simplificar el diseño y la integración de equipos de medida entre sí y especialmente de estos con el ordenador. Dicha simplificación se consigue al reducir al mínimo los problemas tanto eléctricos como mecánicos y de compatibilidad funcional entre equipos, poseyendo la suficiente flexibilidad para acomodar un amplio y creciente número de productos. El nombre inicial fué el de HPIB (Hewlett Packard Interface Bus). Rápidamente, numerosas empresas empezaron a comercializar equipos que incorporaban este tipo de interfaz con el nombre más habitual de GPIB (General Purpose Interface Bus). Estos trabajos iniciales fueron del interés de la comisión Electrotécnica Internacional (IEC) y del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) y respaldaron el borrador propuesto por Hewlett-Packard. El IEC-625 y el IEEE-488/1978 son los nombres oficiales de las normas publicadas por los dos organimos anteriormente mencionados.

La interfaz IEEE-488 se aplica a sistemas de interconexión de instrumentos en los cuales:

a) El intercambio de información entre los equipos interconectados, sea de naturaleza digital. Este aspecto parece obvio, puesto que todos los periféricos que hemos tratado hasta ahora han sido de naturaleza digital. Sin embargo, este bus se concibió para interconectar instrumentos de medida, como generadores de señal, osciloscopios, analizadores de espectro, fuentes de alimentación, multímetros, etc. que en principio tienen una naturaleza analógica.
b) El número de equipos a interconectar no exceda de 15. La limitación a 15 aparatos se debe a que por ser un procedimiento asíncrono la sobrecarga del bus hace que el funcionamiento no sea fiable para una carga mayor.
c) Las longitudes totales de transmisión sobre los cables de interconexión no exceda de 20 metros o de dos metros por equipo, cuando no se utilicen técnicas especiales de ampliación del bus.
d) La velocidad de los datos en la interconexión y en cualquier línea de la misma no supere la cantidad de 1 MByte/segundo. Consideraciones prácticas hacen que el límite de velocidad de transmisión de datos sea del orden de 250 KBytes/s. No obstante, la revisión IEEE-488.2

garantiza velocidades de 1MB/segundo y casi cualquier interfaz comercial supera con creces esta velociad.

Las normas IEEE-488 e IEC-625 son totalmente compatibles a nivel funcional y eléctrico, pero no a nivel mecánico. La propuesta mecánica de IEC contempla la utilización de un conector de 25 contactos, exactametne igual al utilizado por los interfaces RS-232 (CCITT V.24) mientras que la norma de IEEE propone un conector de 24 contactos tipo Ribbon. Sin embargo, esta diferencia mecánica es fácil de subsanar utilizando los adaptadores adecuados. La principal ventaja de los conectores Ribbon es que incorporan un macho y una hembra de tal forma que por el un lado se conecta a un determinado instrumento u ordenador y por el otro lado se puede conectar un nuevo cable que enlace al siguiente equipo.

3.11.1 Estructura del bus

El sistema de interface IEEE-488 utiliza una estructura de bus de línea compartida, es decir los equipos comparten las líneas de señal. La estructura del bus consiste en 16 líneas de señal (ocho de datos y ocho de control), y ocho líneas de masa en una configuración paralela y manteniendo un flujo ordenado de información entre equipos e interconexión. Cualquier equipo conectado al GPIB puede ejecutar una o más de las siguientes funciones:

A) 'Talker' = Locutor. Equipo capaz de transmitir datos si es direccionado. Aunque más de un equipo pueda tener esta funcionalidad, en cada momento sólo puede haber un locutor activo conectado al bus.

B) 'Listener' = Oyente. Instrumento direccionado para recepción de datos. Varios escuchas pueden estar activos sobre un mismo interfaz simultáneamente. Esto permite enviar datos a varios dispositivos simultaneamente y en este caso la transferencia será a la velocidad del equipo más lento.

C) 'Controller'= Controlador. Unidad destinada a direccionar los instrumentos conectados al interfaz, En la mayoría de los casos será un ordenador. Asimismo puede definir una unidad como hablador o como escucha si aquella es funcionalmente ambivalente. Lógicamente el controlador puede asumir también funciones de habla y escucha. La única restricción es que sólo puede existir un controlador activo, de forma simultánea en el bus, de forma que en sistemas multimaestros, uno de los controladores toma el bus, mientras los demás permanecen pasivos o adoptan la apariencia funcional de escuchas o habladores.

La figura (3.29) muestra la estructura de conexión de diferentes equipos entre sí por medio de este bus.

En general la denominación de controlador es atribuíble a uno o varios instrumentos con capacidades de control sobre todo el sistema de medida, es decir las funciones propias de cada equipo locutor u oyente y las funciones propias del interfaz. Generalmente, el controlador toma la forma de un ordenador, aunque esto no es imprescindible.

Desde el punto de vista del bus, el controlador asume el pilotaje de todas las funciones propias del mismo, cuidando que las transferencias de datos a su través se efectúen correctamente. Otra función básica del controlador consiste en determinar qué instrumentos actúan como locutores, cuales como oyentes y en que momento lo hace cada uno.

En un sistema pueden coexistir diversos controladores, pero solamente uno de ellos tendrá oficio de controlador general del sistema, ya que la especificación del interfaz no permite la existencia de más de un controlador maestro. Únicamente el controlador central del sistema puede

activar los circuitos de validación con la señal REN (Remote ENable) y de invalidación IFC (Interface Clear) del interfaz.

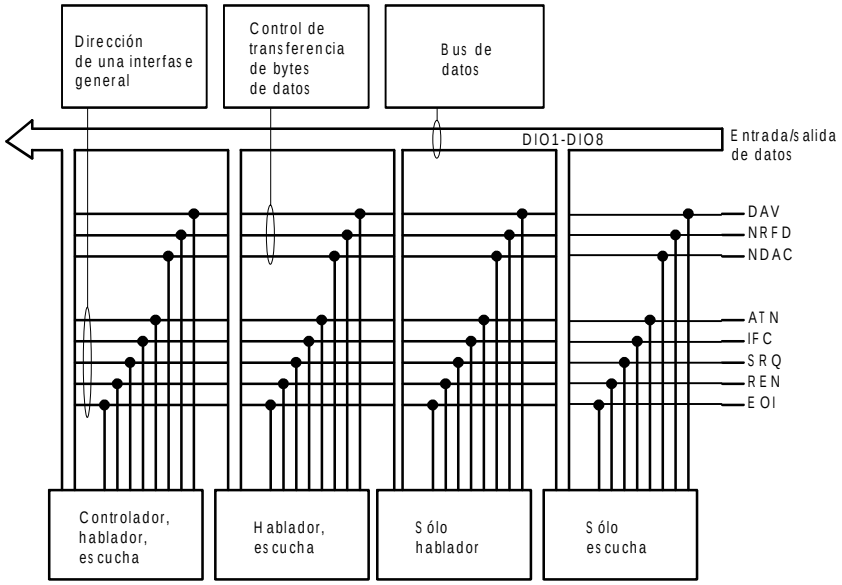


Fig. 3.29 Estructura de una conexión por GPIB

Para sistemas pequeños, en los que no se requieran posibilidades de reconfiguración dinámica de equipos o en aquellos otros donde no existe procesado de las señales medidas, es posible prescindidr del controlador, siempre que se configuren las funciones de los instrumentos de forma manual. Una solución de este tipo es evidentemente rígida y solamente será válida en sistemas extremadamente simples. Un ejemplo de esta situación se presenta, cuando se tiene por ejemplo un osciloscopio y queremos imprimir la pantalla en una impresora o plotter. A la impresora, normalmente no hay que hacerle nada, puesto que siempre está configurada como "listen only" (Sólo escucha); sin embargo un osciloscopio moderno, puede realizar cualquier función. Por eso, cuando vayamos a imprimir deberemos configurar el osciloscopio como "talk only" (Sólo habla). Esto se puede hacer normalmente desde el panel de control del propio instrumento, o a través de los menús de pantalla si el equipo incorpora pantalla táctil.

3.11.2 Examen funcional del bus

Todas las líneas de señal del bus de la interfaz funcionan en lógica negativa y niveles TTL y están cargadas por circuitos en colector abierto (voltaje menor de 0.8 V es un '1' y superior a 2.5 V es un '0'). Estas líneas se pueden clasificar en tres grupos bien diferenciados (ver figura 3.30)

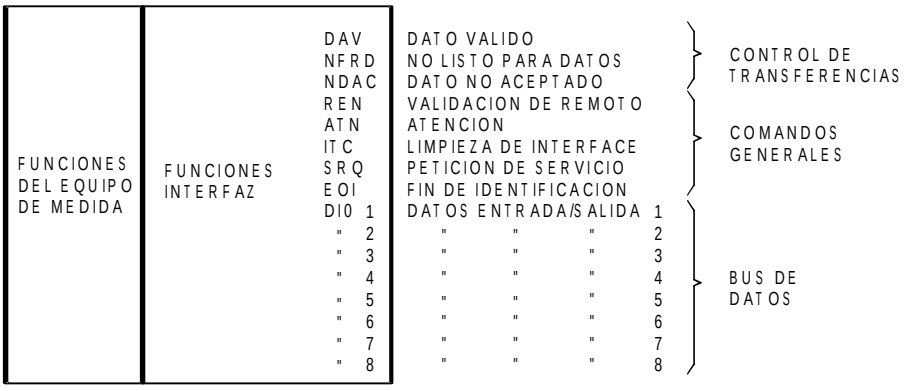


Fig. 3.30 Líneas del bus

a) Líneas de datos

Existen ocho líneas bidireccionales (DIO1 - DIO8) que se utilizan para la transferencia de datos entre un equipo que los envía ('talker') y tantos otros como estén en ese momento recibiéndolos ('listeners'). Normalmente se utiliza un código ASCII normalizado de 7 bits, con el octavo disponible para el control de paridad. La información transferida incluye los comandos de control de la interface, direcciones y datos dependientes de los equipos. Los equipos deben poseer registros de almacenamiento de lectura a fin de asegurar su recepción.

Las ocho líneas bidireccionales son utilizadas para:
- las medidas
- las instrucciones de programación de equipos
- las direcciones
- las palabras de estado
- comandos universales multilínea

b) Líneas de control

Existen ocho líneas de control, de las que tres (DAV, NRFD y NDAC) son líneas de protocolo y se usan para coordinar el intercambio de información entre los equipos e instrumentos conectados al interfaz. El objetivo de las líneas de protocolo es facilitar el manejo del bus por el controlador, así como permitir una gran flexibilidad en la conexión de equipos e instrumentos de diversa índole. Gracias a las líneas de protocolo la transferencia de datos puede ser asíncrona y la velocidad de transferencia puede ajustarse automáticamente a la velocidad del equipo activo más lento. Cabe asimismo la posibilidad de que se produzca aceptación de datos por más de un equipo de forma simultánea. Por otra parte, cada una de las cinco líneas restantes presenta una función específica entre el controlador y el resto de equipos conectados al sistema.

DAV DATA VALID (Dato válido). Es una de las tres líneas de control de la transmisión de datos. Un '1' (< 0.8V) indica que el dato está disponible en las líneas de datos. Se controla por el locutor activo o por el controlador

NRFD NOT READY FOR DATA (No preparado para recibir datos). Es otra de las tres líneas de control mencionadas anteriormente. Un '0' indica que los equipos que reciben 'listeners' están dispuestos a admitir los datos. Es controlada por todos los escuchas activos o por aquel equipo que esté recibiendo los comandos del bus.

NDAC NOT DATA ACCEPTED (Dato no aceptado). Es la tercera de las líneas mencionadas anteriormente, indica al equipo emisor que los datos han sido leídos. Un '0' indica que todos los receptores han leído los datos. Esta línea viene controlada por los escuchas activos o por todos los dispositivos que estén recibiendo comandos del interfaz.

ATN ATTENTION (Atención). Esta señal es utilizada por el controlador del bus para indicar que se va a mandar una orden. Un '1' en ATN indica que los datos enviados son órdenes. Todos los equipos deben monitorizar esta señal de forma continua y responder a ella antes de 200 ns. Cuando esta línea es activa, la señal ATN coloca al bus en modo comando, de modo que todos los equipos acepten datos y los interpreten como comandos. Todos los equipos son receptores de comandos.
Cuando la línea está desactivada, la señal ATN coloca al interfaz en modo dato. En este modo un hablador activo proporciona datos solo a los escuchas activos, el resto de escuchas ignoran los datos.

IFC INTERFACE CLEAR (Liberar interfaz). Esta señal se usa para inicializar todos los aparatos del bus, es decir situarlos en un estado no direccionado y no activo, a fin de conseguir una situación transparente de los mismos antes de iniciar una nueva secuencia de operaciones en el bus. Todos los equipos deben monitorizar constantemente esta señal y responder a ella antes de 100 μs. Un '1' causa la vuelta a las condiciones iniciales de todos los aparatos conectados al bus.

REN REMOTE ENABLE (Permiso remoto). Esta línea es usada por el controlador del sistema para disponer los equipos conectados al bus en el modo de programación remota. Cuando la línea es activa, todos los escuchas se colocan en operación remota cuando se les direcciona como tales. Cuando la línea está desactivada, todos los equipos vuelven al modo local.
Cualquier equipo capaz de operar de forma remota y local debe necesariamente monitorizar en todo momento la línea REN y debe ser capaz de responder a un cambio de nivel en la misma antes de 100 μs. Un '0' permite a todos los aparatos del bus ser controlados por el bus GPIB.

SRQ SERVICE REQUEST (Solicitud o Demanda de servicio). Esta señal se usa para que los equipos que requieren servicio se lo indiquen al bus (p. ej. han completado una tarea, se ha producido un error, etc.). Cuando el controlador detecta un '0' en SRQ muestrea los aparatos en búsqueda del que está requiriendo sus servicios para atenderlo a continuación.

EOI END OF IDENTIFY (Final de identificación) Esta señal tiene dos funciones. Cualquier equipo puede poner un '0' en EOI indicando que ha terminado la transmisión. El controlador puede usar EOI para iniciar un muestreo en paralelo. (Cuando se envían juntos un ATN y un EOI, los aparatos conectados al bus presentan sus bits de estado en las líneas de datos).

Aclaremos ahora algunos conceptos introducidos en la descripción de las señales precedentes. En primer lugar, hay que tener en cuenta que las salidas de los equipos son en colector abierto, lo que permite realizar una OR cableada. Esto resulta especialemente interesante cuando un equipo emite y varios reciben. La línea que indica que los datos han sido recibidos sólo pasará a estado alto cuando todos los dispositivos pongan su salida en alto, con lo que cuando el emisor detecta un estado alto en esta línea de reconocimiento tiene la seguridad de que todos los dispositivos oyentes han aceptado el dato. Ver figura (3.31).

El segundo aspecto a tener en cuenta es el de modo local o remoto. La mayoría de equipos diseñados para ser conectados a través del bus GPIB son relativamente complejos y pueden funcionar de forma autónoma y ser controlados manualmente a través de su panel de control. Pensemos por ejemplo en un osciloscopio, un generador de señal o una fuente de alimentación digital. Estos equipos pueden desarrollar toda su funcionalidad sin necesidad de ser conectados a otros equipos, al contrario de lo que sucede con un disco duro, o una impresora, que no tienen ninguna utilidad si no los conectamos a un ordenador. Sin embargo, para aumentar sus posibilidades, puede que deseemos conectar esos equipos entre sí o a un ordenador central. Cuando varios de estos equipos son conectados al bus, podemos controlarlos a través de su panel frontal (modo local) o a través del bus (modo remoto). Habrá situaciones en las que queramos que el control remoto (realizado desde un ordenador a través del bus) no se vea interferido por manipulaciones del panel de control del equipo, para lo que resultará conveniente bloquear este modo de funcionamiento. Esto se consigue, como se comentará más adelante, con el comando: 'Local Lockout' (Bloqueo del Modo Local).

3.11.3 Protocolo de operación

Los datos se transmiten en el GPIB byte a byte. Las líneas de control DAV, NRFD y NDAC manejan la transferencia de bytes realizando un intercambio entre los equipos transmisor y receptor(es). El procedimiento de intercambio asegura que un byte no es enviado hasta que todos los receptores están preparados, que cada receptor sólo lee el bus cuando el byte válido está allí, y que el emisor mantiene su dato en el bus hasta que ha sido leído por todos los receptores. El diagrama de tiempos de la figura (3.31) ilustra como se usan las señales DAV, NRFD y NDAC para realizar esto.

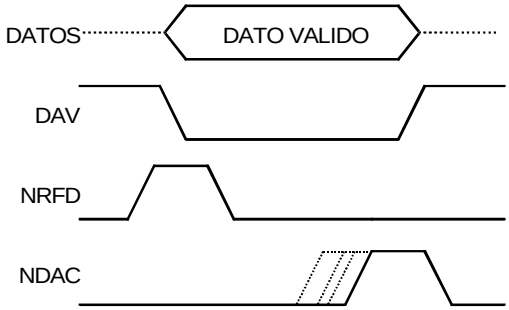


Fig. 3.31 Diagrama de tiempos de las líneas de control del GPIB. En la última de las curvas correspondiente a la señal NDAC se muestra el efecto de la OR cableada sobre el bus. Las líneas punteadas muestran cuando los distintos equipos reconocen la aceptación del dato, pero la línea no pasa a estado alto hasta que todos ellos lo han hecho.

Antes de poner un dato en el bus el equipo transmisor debe esperar a que la señal NRFD se ponga a '0' (> 2.5 V). Como está conectada en colector abierto, esto quiere decir que todos los aparatos del bus están dispuestos para recibir datos, es decir, que ninguno va a utilizar el bus para transmisión. Una vez la línea NRFD está en alto, el emisor puede poner sus datos en el bus. Además pone la señal DAV a bajo nivel (dato válido en el bus). Cuando el receptor detecta un nivel bajo en DAV, lee el dato del bus. Cuando cada receptor termina de leer el dato libera NDAC. Cuando el último libera NDAC, ésta va a '0' (nivel alto), señalando al emisor que el dato ha sido aceptado por todos los receptores. Una vez que ha sido aceptado por todos los aparatos del bus, el emisor puede quitar el dato del bus y liberar DAV.

Los equipos conectados al bus pueden enviar datos, recibirlos o controlar el bus o cualquier combinación de estas funciones. El controlador debe fijar qué equipo es el emisor y cuales los receptores, además el controlador ejecuta otras funciones de control del bus. Esto lo realiza por medio de las órdenes del bus, que se envían de forma análoga a los datos pero con la señal ATN puesta a '1'. Las órdenes son leídas por todos los equipos.

El controlador puede enviar cuatro tipos de órdenes o comandos: direcciones ('adress'), escuche ('listen'), hable ('talk') y universal ('universal'). Para dar estas órdenes sólo se usan los siete primeros bits del bus de datos. El tipo de comando se da en los bits 5, 6 y 7 (tabla 3.15). Existen también comandos secundarios que se usan para configurar un equipo para un muestreo paralelo o enviar direcciones secundarias.

b7	b6	b5	Tipo de comando
0	0	0	De dirección
0	0	1	Universal
0	1	x	De habla
1	0	x	De escucha
1	1	x	Secundario

Tabla 3.15 Código del tipo comando del GPIB

Comandos hable y escuche

Estos comandos sirven para indicar el paso al estado de emisor o receptor de un determinado equipo. Los últimos cinco bits permiten direccionar el equipo a que nos referimos, éstos pueden tener bien por hardware o software asignadas direcciones en el bus de 0 a 30 (la dirección 31 se usa en los comandos UTN ('untalk') y UNL ('unlisten') que mandan a todos los equipos a estado de reposo.

Comandos universales

Son los que afectan a todos los equipos del bus. Existen cinco tipos de comandos universales:

LLO 'Local Lockout' (11H) (bloqueo del modo local). Esta orden inutiliza los controles manuales de los equipos a fin de que no exista conflicto entre las instrucciones enviadas por el bus y los mandos locales de los equipos.

DCL 'Device Clear' (14H) (inicialización de todos los equipos). Esta orden reinicializa todos los equipos en el bus.

PPU 'Parallel Poll Unconfigure' (15H). Esta orden resetea las respuestas a un muestreo o encuesta paralelo, permitiendo realizar un nuevo muestreo.

SPE SPD 'Serial Poll Enable' (18H) y 'Serial Poll Disable' (19H). Estas órdenes se usan para ejecutar un muestreo serie de los equipos del bus. Cuando el comando detecta una demanda de servicio, debe determinar qué equipo es el que lo demanda antes de actuar. El controlador envía un SPE y a continuación manda un comando 'hable' a cada equipo (uno detrás de otro) y éstos contestan con un byte de estado. Cuando encuentra el equipo que quiere emitir, el controlador envía un SPD que causa la vuelta al estado normal de todos los equipos y así el controlador puede pasar a atender el servicio requerido.

Comandos de direcciones

Estas órdenes sólo afectan a equipos a los que previamente se les ha enviado una orden de 'escuche' y por ello sólo afectan a ciertos equipos. Existen cinco comandos de direcciones:

GTL 'Go To Local' (001). Esta orden cancela el comando universal LLO volviendo el control de los equipos que están recibiendo a modo local.

SDC 'Selected Device Clear' (004). Esta orden libera todos los equipos que han recibido la orden 'escuche'.

GET 'Group Trigger' (008). Esta orden sincroniza la operación de un cierto número de equipos. Estos han de ser previamente programados en su actuación, y cuando reciben un GET comienzan simultáneamente sus tareas.

TCT 'Take Control' (009). Esta orden transfiere el control del bus de un controlador a otro, que ha sido previamente puesto en condición de recibir. Tras recibir esta orden toma el control del bus el nuevo controlador y comienza a enviar órdenes.

PPC 'Parallel Poll Configure' (005). Esta orden prepara a cualquier equipo para participar en un muestreo paralelo. Un muestreo paralelo se efectúa para conocer qué equipo está requiriendo servicio cuando SQR se pone a '0'.
El muestreo paralelo se inicia cuando el controlador envía un ATN y un EOI simultáneamente; esto causa que los equipos que han sido configurados para un muestreo paralelo pongan un bit de estado en una de las líneas del bus. El controlador a continuación examinará el bus para determinar qué equipo requiere servicio. El comando PPC se usa para especificar qué bit usará el equipo para especificar su estado. Después del PPC, el controlador envía al equipo un comando secundario cuyos tres bits menos significativos indican el bit que será usado para especificar su estado (p. ej. 010 indica que será el bit tercero), el cuarto bit señalará cómo se indicará el estado del equipo (si es '0' el bit de estado se pondrá a '0' para indicar que se desea servicio, y si es '1' se pondrá a '1').
Durante un muestreo paralelo más de un equipo puede usar el mismo bit para indicar su estado. Si este bit es '0' en los dos, la línea en el bus será '0' si uno de ellos requiere servicio. Si fuera '1' la línea del bus será '1' si los dos equipos requieren servicio.

Implementación del interfaz GPIB

Existen numerosos fabricantes que suministran tarjetas de interfaz GPIB para casi cualquier tipo de sistema. Suelen estar basadas en circuitos integrados VLSI GPIB, por ejemplo los Intel 8291 talker/listener, 8296 GPIB controller, y un par de 8293 GPIB transceivers. También Texas Instruments (TMS9914), Signetics (HEF4738) y Motorola 6848. Un interfaz GPIB permite a un ordenador ser talker/listener/controller y ejercer el control del bus GPIB con muy poco software ya que el interfaz maneja todos los protocolos y en general permite interrumpir al procesador activando una línea de interrupción.

El bus IEEE-488 tiene tres características fundamentales que lo hacen especialmente adcuado a un entorno experimental o de laboratorio. La primera es que los equipos, pueden conectarse entre sí con una gran flexibilidad. A cada equipo llega un cable que se conecta mediante un conector tipo Ribbon. Desde este mismo punto, se puede conectar un segundo cable a un segundo equipo, pero como este nuevo cable tendrá nuevamente un macho por una parte y una hembra por la otra, volvemos a tener la posibilidad de seguir conectando equipos. Es decir, de cualquier equipo pueden partir cables a un número variado de otros equipos, desde uno hasta el máximo permitido (15), si aplicamos esto a todos los equipos veremos que la flexibilidad es considerable, aspecto que resulta fundamental en un entorno de laboratorio, donde los equipos se conectan y desconectan de forma frecuente. Otras interfaces, como SCSI, exigen que todos los equipos estén conectados en cadena.

La segunda característica es que los datos que envía un locutor pueden ser recogidos por varios oyentes simultánemente. Esto permite por ejemplo, que un voltímetro esté enviando datos a un ordenador para análisis y almacenamiento y simultáneamente, esos datos estén siendo recogidos por un filtro digital, o estén controlando la salida de una fuente de alimentación o múltiples cosas de forma simultánea. Para que esto funcione, se emplean las señales de NRFD y NDAC que se conectan al bus mediante la técnica de colector abierto. Según esta técnica, basta que un equipo mantenga su salida en baja, para que la línea correspondiente del bus también lo esté. Esta señal sólo pasará a alta cuando todos los equipos hayan puesto su salida en alta. La señal NRFD indica cuando los equipos están preparados para recibir un dato y la NDAC indica que todos los equipos ya lo han recibido.

La tercera característica, es que permite sincronizar acciones entre distintos equipos, mediante el envío de comandos de disparo. De esta forma, un generador de señal puede comezar a proporcionar una determinada salida para un equipo bajo estudio y simultáneamente, un

osciloscopio comienza a registrar las señales a la entrada y salida de dicho equipo, y ambas acciones comenzarán en los distintos equipos, y pueden ser todos, de forma sincronizada.

Como se ve son características que lo hacen ideal en entornos de instrumentación y de laboratorio y en este campo, es de utilización prácticamente universal y todos los equipos de laboratorio profesionales lo incorporan o bien de serie o al menos como opción.