

TEMA 6: COMUNICACIÓN CON EL EXTERIOR (Resumen)

- 6.1. [Introducción. Las unidades de E/S y los dispositivos periféricos.](#)
- 6.2. [Unidades de entrada/salida.](#)
 - 6.2.1. [Objetivos y necesidades de las unidades de E/S.](#)
 - 6.2.2. [Técnicas de transferencia.](#)
 - 6.2.2.1. [E/S Programada.](#)
 - 6.2.2.2. [E/S por acceso directo a memoria.](#)
 - 6.2.3. [Señales de control y estado de los periféricos.](#)
 - 6.2.4. [Generalidades sobre prioridades](#)
 - 6.2.4.1. [Gestión distribuida de prioridades.](#)
 - 6.2.4.1.1. [Encadenamiento o Daisy-Chain.](#)
 - 6.2.4.1.2. [Lógica distribuida.](#)
 - 6.2.4.2. [Gestión de prioridad centralizada.](#)
 - 6.2.4.3. [Gestión de prioridad híbrida.](#)
 - 6.2.5. [Interrupciones. Selección de la rutina de tratamiento de la interrupción.](#)
 - 6.2.5.1. [Prioridades y niveles de interrupción.](#)
 - 6.2.6. [Canales de E/S, procesadores de E/S \(IOP\) o unidad periférica de proceso \(PPU\).](#)
 - 6.2.7. [E/S y sistema operativo.](#)

Los sistemas de entrada/salida constituyen uno de los temas de mayor interés y complejidad en el diseño y utilización de los computadores. Abarca desde los cronogramas y mecanismos de sincronismo de las señales empleadas en la interconexión de los diferentes dispositivos periféricos, hasta los procedimientos empleados por el sistema operativo para ejecutar las diversas solicitudes de entrada/salida, pasando por la estructura de los elementos de interconexión y el desarrollo de los programas de accionamiento de los mismos.

En este tema se pretende estudiar los mecanismos básicos de transmisión de información entre CPU y periféricos, y en la organización de las operaciones de E/S. Además se repasan las técnicas para la gestión de prioridades cuando más de un peticionario solicita un mismo recurso y las distintas formas de generar una interrupción. También se repasarán conceptos relacionados con los canales de E/S atendiendo a su clasificación, la intervención del Sistema Operativo en una operación completa de E/S y por último se verá un ejemplo de diseño de un sistema de E/S.

6.1. INTRODUCCIÓN. LAS UNIDADES DE E/S Y LOS DISPOSITIVOS PERIFÉRICOS

Hasta ahora se ha estudiado el sistema computador como una máquina que puede ejecutar un programa inicialmente registrado en la Memoria Principal, sobre datos registrados en la misma Memoria Principal y almacenar los resultados en dicha memoria a medida que se obtienen. Ahora es preciso dotarla de medios para comunicarse con el exterior: este es el papel reservado a las *Unidades de Entrada/Salida* y a los *Dispositivos Periféricos*.

Existen dos grandes clases de *Unidades Periféricas*:

- a) Las *Unidades de Comunicación* (impresora, pantalla, teclado, etc.) que permiten el diálogo con el exterior.
- b) Las *Unidades de Almacenamiento* (discos, cintas magnéticas, etc.), cuyas capacidades de almacenamiento son muy superiores a la de la Memoria Principal.

Las Unidades Periféricas se conectan a la Unidad Central de Proceso o directamente a la Memoria Principal, a través de unidades especializadas en la gestión de las transferencias de información. Estas unidades de intercambio de información con el exterior se llaman *Unidades de E/S*. La Unidad de Control se comunica con estas Unidades de E/S mediante instrucciones que pueden ser específicas de E/S o generales de transferencia, dependiendo del sistema computador.

En este tema se estudiarán las Unidades de entrada/salida, dejando para un curso posterior el estudio de los dispositivos periféricos.

6.2. UNIDADES DE ENTRADA/SALIDA

6.2.1. Objetivos y necesidades de las unidades de E/S

El objetivo de las Unidades de Entrada/Salida (E/S) también llamadas de intercambio de información, es realizar la conexión y la adaptación de la Unidad Central de Proceso (CPU) del computador con los dispositivos periféricos. Esta conexión presenta unas características muy especiales como son las siguientes:

- Los periféricos tienen unas velocidades de transmisión que abarcan desde unos pocos de bytes por segundo, hasta varios millones de bytes por segundo. Sin embargo esta velocidad es sensiblemente inferior a la de la CPU (10 MIPS a 10 GIPS).

- Los periféricos suelen tener un ancho de palabra de un byte, lo que no coincide con el ancho de palabra de la mayoría de los computadores.
- Algunos periféricos son de lectura, otros de escritura y, por último, otros son simultáneamente de lectura y escritura.

El problema del intercambio de información entre la CPU y los periféricos es relativamente complejo. Por un lado, hay que establecer mecanismos que permitan transmitir la información, mecanismos que deben considerar aspectos tales como el direccionamiento o selección del periférico, la forma de establecer el camino para el envío de datos, la posible conversión serie-paralelo, la posible conversión de códigos, etc. Por otro lado hay que establecer un mecanismo de control que determine el origen y el destino de la información, la cantidad de ella a transmitir, los códigos de protección, etc. Estos mecanismos se reparten entre el controlador del periférico, la unidad de control de la CPU y los programas de entrada/salida que, normalmente, forman parte del sistema operativo.

Por todo ello hay que diferenciar entre lo que es la transferencia elemental de información, la transferencia de bloque y la operación de entrada/salida.

- La **TRANSFERENCIA ELEMENTAL** de información tiene por objeto el envío o la recepción de una única unidad de información (un byte o una palabra) ya sea ésta un dato o una palabra de control. Suele ser el resultado de la ejecución de alguna instrucción de Entrada-Salida (mapa de memoria no común), por ejemplo INPUT, OUTPUT; o de transferencia de datos (mapa de memoria común), como por ejemplo las instrucciones MOVE, STORE, LOAD.
- La **TRANSFERENCIA DE BLOQUE** se encarga de enviar o recibir un bloque de información, como puede ser un sector de un disco, un registro de una cinta o una trama de un dispositivo de comunicación hacia o desde posiciones consecutivas de memoria principal. Un bloque de información consiste en un conjunto de unidades de información (byte o palabra) que ocupan posiciones consecutivas.
- La **OPERACIÓN DE E/S** tiene por objeto el proceso global de transferencia de un conjunto de datos, garantizando que ésta se produzca correctamente. Para ello gestiona la transferencia de bloques y supervisa el estado del periférico, realiza el tratamiento de los errores y lleva la cuenta de los bloques transferidos. También se encarga de la conversión de código en el caso de que los datos estén representados en el periférico en un código diferente al requerido por el programa.

Las funciones más importantes de cada una de estas operaciones están reflejadas en la [Figura 6.1](#).

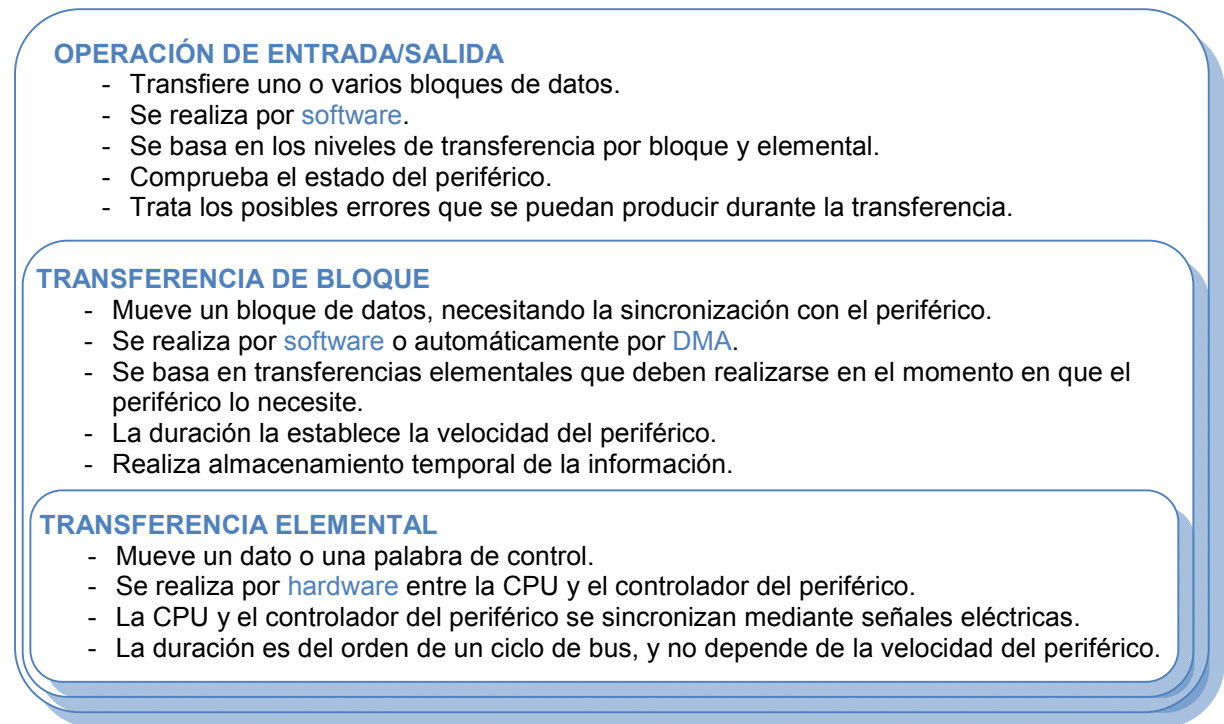


Figura 6.1. Estructuración de los niveles de transferencia de E/S.

6.2.2. Técnicas de transferencia

Las operaciones de E/S se basan en transferencias elementales, por tanto, es necesario conocer los procedimientos empleados para transmitir una información elemental entre el periférico y la CPU. Esta información puede ser, según sea el caso, una palabra, un byte o un bit. Existen dos alternativas para hacer una transferencia elemental:

- E/S programada, por ejecución de una instrucción.
- Por acceso directo a memoria (DMA).

6.2.2.1. E/S programada

Las instrucciones de E/S son fundamentalmente instrucciones de transferencia donde uno de los operandos implicados está almacenado en un registro del controlador del periférico y el otro operando en un registro general de la CPU. La CPU suministra información al controlador del periférico para que éste genere las señales necesarias que permiten ejecución de la transferencia. Esta información que la CPU suministra al controlador consiste en:

- Información de dirección para elegir el registro del periférico con el que se realiza la transferencia.

- Tipo de operación, por ejemplo escritura o lectura de un registro de un periférico.
- Temporización. La transferencia exige que se establezca un camino físico entre los dos elementos de almacenamiento y que se genere una señal de flanco cuando los datos estén estabilizados a la entrada del destino. Las señales de la CPU y del periférico han de tener una temporización coherente para que la transferencia se haga correctamente.

Al decodificar la instrucción de E/S, la unidad de control del computador suministra al exterior estas informaciones y envía el dato o se dispone a recibirlo, dependiendo del tipo de acceso. La comunicación se realiza mediante las siguientes señales:

- Señales de dirección para seleccionar el periférico.
- Señales de datos que permiten el intercambio de datos.
- Señales de control que permiten que el controlador del periférico y la CPU se pongan de acuerdo.

SEÑALES DE DIRECCIÓN

La información de la dirección se compone por p bits lo que permite establecer 2^p direcciones distintas, sirviendo cada una de ellas para especificar lo que se llamará una **puerta de entrada/salida**. Estas puertas de entrada/salida sirven para enviar y/o recibir información a tamaño de byte o palabra. Cada periférico empleará una o varias puertas para comunicarse con la CPU. Al conjunto de las 2^p direcciones se le llama **mapa de E/S**.

Es necesario establecer un mecanismo de decodificación que convierta la dirección en la señal de selección correspondiente. Para ello se puede emplear la decodificación centralizada o la decodificación independiente.

SEÑALES DE DATOS

Los datos pueden transmitirse por un conjunto de líneas bidireccionales o dos conjuntos de líneas de dirección única. Las técnicas básicas de conexión son dos: **Conexión a un bus común** y **Conexión mediante Multiplexor/Demultiplexor**.

SEÑALES DE CONTROL

Las señales de control deben especificar el tipo de transferencia, de entrada o de salida, y deben establecer su temporización. Existen dos métodos, denominados **síncrono** y **asíncrono**, para establecer esta temporización.

6.2.2.2. E/S por acceso directo a memoria (DMA)

En el acceso directo a memoria (DMA: *Direct Memory Access*) el controlador del periférico se comunica directamente con la memoria principal del computador, sin

intervención de la CPU. La transferencia elemental se realiza sin que se ejecute ninguna instrucción en la CPU, siendo el controlador del periférico el que se encarga de generar las señales de control. El DMA exige que el computador envíe al controlador del periférico un determinado tipo de información para que éste sepa lo que tiene que hacer. Esta información se denomina **bloque de control** y consiste en:

- Dirección en memoria principal, para generar las señales de dirección en el acceso a memoria.
- Tipo de operación: lectura o escritura.
- Número de datos a transferir.
- Dirección del periférico.

Para hacer llegar el bloque de control al controlador del periférico se dotará al mismo de una o varias puertas de E/S programada. Basta con ejecutar un programa que contenga tantas instrucciones de salida como palabras tenga el bloque de control. Una vez recibida la información, el controlador de forma independiente, realiza los necesarios ciclos de DMA. Finalizada la operación de E/S, el controlador del periférico envía una señal de interrupción a la CPU. Esto hace que este tipo de transferencia tenga sentido cuando se envíe o se reciba un bloque de datos. En este sentido, el DMA admite dos tipos de transferencia, la transferencia de bloque y la transferencia elemental. Se hablará de operación de DMA para referirnos a la transferencia de un bloque, y de transferencia por DMA, para referirnos al intercambio de una palabra.

Las principales aplicaciones de las operaciones de E/S por DMA son:

- Controladoras de discos y disquetes.
- Operaciones de descanso.
- Adquisición de datos.
- Transferencia de memoria a memoria.
- Búsqueda en memoria.
- Almacenamiento de seguridad.
- Sistema de buses paralelos.
- Conexión con fibra óptica.
- Transferencia de bloques en multiproceso y multiprograma.

La desventaja de realizar operaciones por DMA es que la CPU puede quedar inactiva (no siempre) perdiendo parcial o totalmente el control del sistema de buses, en consecuencia no hay servicio de interrupciones y abandono de refrescos.

Las operaciones de E/S por DMA pueden dar lugar a conflictos de acceso, ya que varios clientes (CPU y controladores) pueden solicitar accesos a memoria simultáneos, que es

un recurso único. Por tanto es necesario disponer de un mecanismo de prioridad. Existen dos formas básicas de resolver estos conflictos:

- Mediante Memoria Multipuerta.
- Mediante Robo de Ciclo.

MEMORIA MULTIPUERTA

La memoria multipuerta permite que los periféricos dispongan de la memoria principal sin intervención de la CPU. Al estar dividida en varios bloques de memoria, permite accesos paralelos, es decir puede gestionar peticiones de acceso en paralelo siempre que estas peticiones simultáneas no sean operaciones de escritura o lectura y escritura en un mismo bloque de memoria. El mayor inconveniente de la memoria multipuerta es su coste, puesto que además de la propia lógica de cada puerta, debe tener un dispositivo de gestión de prioridades para resolver las colisiones, es decir las peticiones simultáneas a un mismo bloque de memoria.

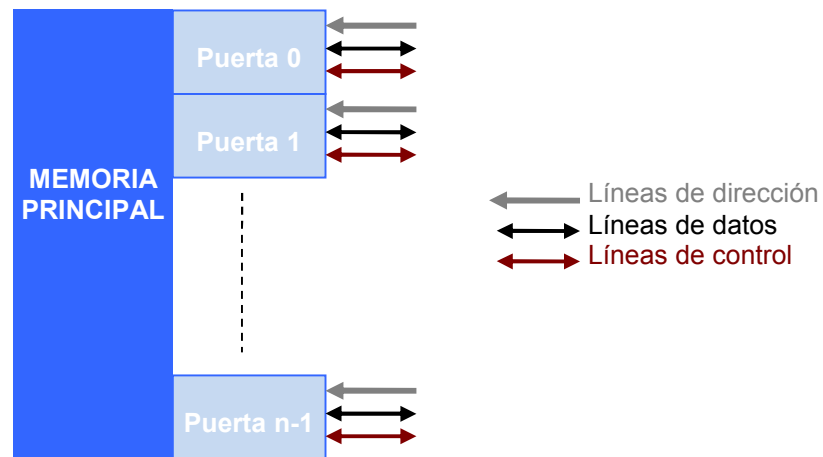


Figura 6.9. Acceso Directo a Memoria con memoria Multipuerta.

La información que debe definir el controlador del periférico para comunicarse con la memoria es:

- La Dirección de la Memoria Principal de la posición a la que pretende acceder.
- El Dato, si la operación es de escritura.
- El inicio de ciclo de memoria, para indicar a la memoria el comienzo de la operación.
- La orden de Lectura o Escritura.

Las señales con las que contesta la Memoria Principal son:

- El Dato, si la operación es de lectura.
- El fin de ciclo de memoria, cuando finaliza la operación.

Dado el coste que implica disponer de una memoria multipuerta, se suelen conectar a cada una de las puertas varios dispositivos periféricos, de forma que el tráfico total de

información que soporten se acerque a su máximo. Para evitar los conflictos de acceso simultáneo que pueden ocurrir en una misma puerta con varios periféricos conectados, se suele emplear un mecanismo adicional de prioridad.

ROBO DE CICLO

El robo del ciclo es una forma más económica de realizar el acceso directo a memoria, puesto que la memoria va a tener una sola puerta que debe ser compartida por la CPU y por los periféricos. La transferencia exige que la CPU y el controlador del periférico se pongan de acuerdo para obtener el control de los buses y las señales de control de la única puerta de memoria.

Los pasos requeridos son los siguientes:

- El controlador del periférico solicita a la CPU el acceso a la Memoria Principal mediante la señal de control BREQ (Bus Request).
- La CPU contesta mediante la señal BACK (Bus Acknowledge) concediendo la solicitud. Al mismo tiempo pone en estado de alta impedancia los buses y señales de control de acceso a la memoria. La CPU debe esperar a terminar la fase de instrucción en curso para conceder el robo de ciclo solicitado.
- El controlador del periférico realiza el acceso a memoria, activando las señales de control y generando las direcciones correspondientes.
- Cuando el controlador del periférico ha terminado, devuelve a la CPU el control de los buses y demás señales desactivando la señal BREQ.
- La CPU desactiva la señal BACK y recupera el control de los buses.
- Si en el acceso a memoria, se ha finalizado la transferencia de un bloque completo de información, el controlador envía a la CPU la señal INT, generando una interrupción, para la avisarla de que ha finalizado la transferencia del bloque completo.
- A continuación, y una vez que la CPU finaliza la ejecución de la instrucción que se encontraba realizando cuando se produjo el robo de ciclo, ésta responde con la señal INTA para comunicar la aceptación de la interrupción provocada

Las señales que el controlador del periférico ha de enviar a la CPU son las siguientes.

- BREQ, para solicitar ciclo.
- INT, señal de solicitud de interrupción.

Las señales que el controlador del periférico ha de enviar a la Memoria son:

- CM, señal de inicio de ciclo de memoria.
- L, señal de nivel que indica ciclo de lectura.

- E, señal de nivel que indica ciclo de escritura.

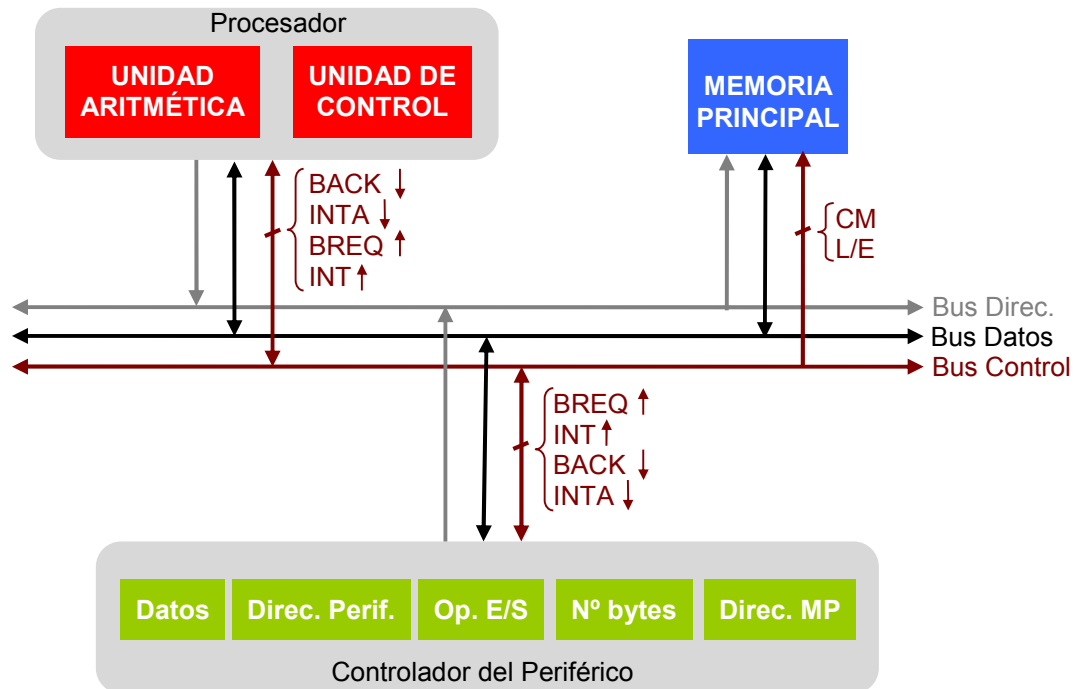


Figura 6.10. Acceso Directo a Memoria por Robo de Ciclo.

Por otro lado, debe interpretar las señales generadas por la CPU:

- BACK, de concesión de los buses.
- INTA, señal de aceptación de la interrupción.

A su vez, el controlador debe ser capaz de acceder a sus propios registros como son:

- El registro de datos, para cargar/dejar su contenido sobre el bus de datos, según sea una operación de lectura/escritura de memoria.
- El registro que contiene la dirección del periférico implicado en la operación de E/S.
- El registro que indica si la operación es de Entrada o de Salida.
- El registro que indica el número de bytes que se van a transferir durante la operación de E/S. Después de cada acceso a memoria, el controlador decrementa este registro.
- El registro de direcciones, que contiene la dirección de memoria a la que se va a acceder. Además el controlador debe ser capaz de incrementar este registro después de cada acceso a una posición de memoria. En una transferencia de bloque, la información de memoria está almacenada en posiciones consecutivas.

Generalmente el periférico que solicita el robo del ciclo es más lento que la CPU, por lo que va intercalando, esporádicamente, robos de ciclo para hacer transferencias

individuales. Sin embargo, puede darse el caso de que una vez obtenido el ciclo, el controlador realice múltiples transferencias, robos de ciclo de tipo ráfaga, antes de desactivar la señal de petición.

Los modos de operación del Controlador de Acceso Directo a Memoria (DMAC) son tres.

- Modo BYTE. El DMAC transfiere un solo byte mientras que el periférico está activo. La CPU toma el control cuando se ha transferido el byte.
- Modo demanda (BURST). El DMA transfiere varios bytes hasta que la CPU retoma el control.
- Modo continuo (BLOCK). El DMA mantiene ocupado los buses todo el tiempo necesario hasta transferir el bloque completo.

El robo del ciclo hace que la duración de la ejecución de la instrucción no sea fija, ya que ésta dependerá de si se realizan robos de ciclo. Este efecto debe tenerse en cuenta si se quieren hacer consideraciones temporales en función de los tiempos de ejecución de las instrucciones.

La sincronización de la transferencia viene reflejada en la [Figura 6.11](#).

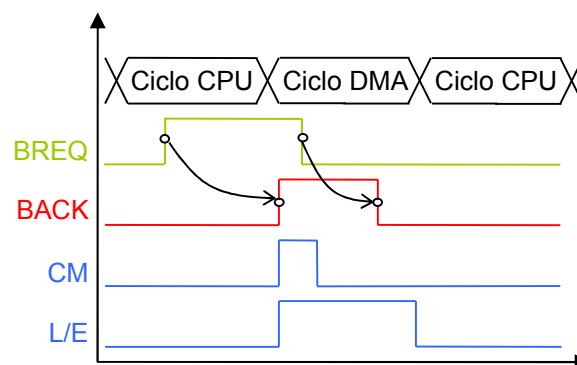


Figura 6.11. Cronograma Acceso Directo a Memoria por Robo de Ciclo.

El controlador del periférico activa la señal BREQ, solicitando el robo del ciclo, a lo que contesta la CPU activando la señal BACK. Esta señal la utiliza el controlador para que el registro de direcciones ponga en el bus de direcciones la dirección correspondiente a la posición de memoria deseada. Finalizado el acceso, el controlador del periférico desactiva BREQ y la CPU desactiva BACK. El controlador de DMA debe conocer las direcciones de origen y destino, el sentido de la transferencia y el número de bytes a transmitir.

6.2.3. Señales de control y estado de los periféricos

La mayor parte de los periféricos disponen de un conjunto de señales de control y estado, que permiten conocer su situación interna y simplificar su gobierno.

Señales de control:

- Encender o apagar.
- Seleccionar opciones.
- Saltar de páginas en impresoras.
- Buscar marca en cintas magnéticas.

Señales de estado:

- Periférico encendido o apagado.
- Periférico en funcionamiento.
- Periférico operativo o no.
- Error de operación.
- Error de paridad.

El tratamiento de las señales de control y estado forma parte de las operaciones de E/S.

Para que la CPU pueda acceder a las señales de estado y pueda enviar las señales de control, existen dos procedimientos:

- Manejar las informaciones como si fuesen datos y transmitirlos mediante E/S programadas. Supone tener periféricos con puertas distintas para datos y para control y estado.
- Mediante diferenciación de un mapa de direcciones de control y estado, añadiendo una señal que diferencia entre E/S de datos y E/S de control. Exige instrucciones especiales como TEST (para leer una puerta de control) o CONTROL (para escribir en una puerta de control).

6.2.4. Generalidades sobre prioridades

El problema de prioridades se presenta siempre que dos o más elementos tienen que compartir un mismo recurso, pudiendo existir peticiones simultáneas. Por tanto hay que establecer un mecanismo para determinar cuál de los dispositivos es atendido primero, haciendo esperar a los demás.

Se llamará PETICIONARIOS a los elementos que comparten el recurso común, y FASE DE SERVICIO al tiempo empleado en atender a cada uno de ellos.

El problema se plantea en situaciones tales como:

- Varios periféricos solicitan al mismo tiempo un robo de ciclo. En este caso el robo de ciclo es la fase de servicio.
- Varios periféricos solicitan al mismo tiempo una interrupción.

- Varios periféricos conectados a un mismo bus solicitan de forma simultánea un ciclo de bus.
- Varias puertas de memoria multipuerta tratan de acceder simultáneamente al mismo módulo de memoria.

Existen dos formas de abordar este problema:

- **GESTIÓN DISTRIBUIDA DE PRIORIDADES.** En este caso los peticionarios han de ponerse de acuerdo entre sí para determinar quién se queda con el recurso.
- **GESTIÓN CENTRALIZADA DE PRIORIDADES.** En este caso existe un elemento maestro que se encarga de decidir la cesión del recurso.

6.2.4.1. Gestión Distribuida de Prioridades

Este método es recomendado cuando los distintos peticionarios están conectados al recurso mediante un bus. Las técnicas que emplean gestión distribuida de prioridades son:

6.2.4.1.1. Encadenamiento o Daisy-Chain. Exige disponer de un elemento maestro que decide cuándo se concede el recurso, aunque son los peticionarios quienes determinan cuál de ellos se queda con la fase de servicio.

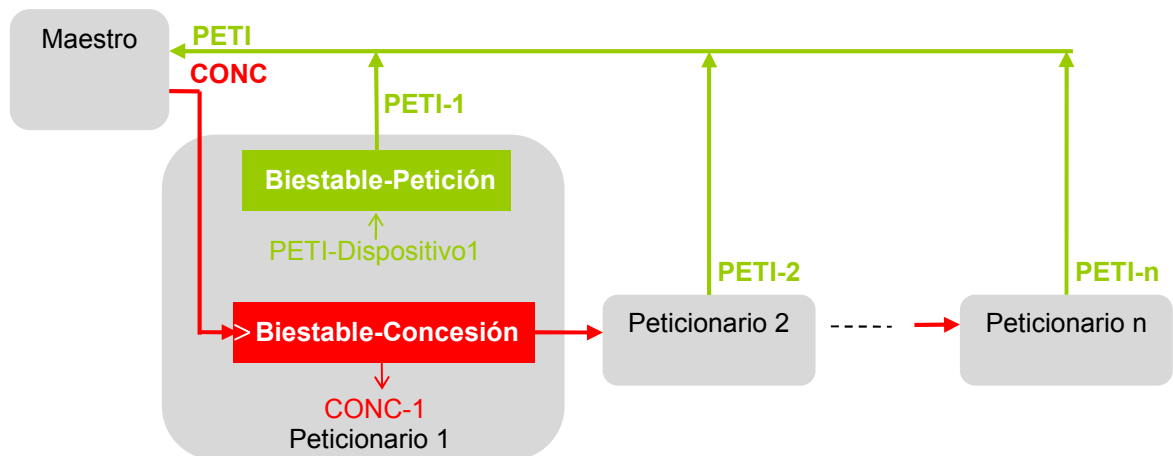


Figura 6.12. Esquema de Conexión para Encadenamiento o Daisy-Chain.

Muchas veces será el propio recurso el que actúe como maestro, otras veces el maestro y el recurso solicitado son distintos. Por ejemplo cuando un periférico solicita a la CPU el robo de ciclo para el acceso a la memoria.

Las características fundamentales del *Encadenamiento* son:

- Es muy sencilla y económica.

- Existe una única línea de petición (PETI) a la que se conectan todos los dispositivos. Esta línea debe permitir que se realicen peticiones simultáneas,
- Existe una única línea de concesión (CONC) que asegura la concesión del recurso. La conexión de los dispositivos es en forma de cadena, tal que el pulso de concesión recorre una serie de dispositivos, según sea el orden en que están conectados en la cadena. El pulso de concesión es tomado por el primero de los dispositivos de la cadena que desea el recurso.
- Debe establecerse un mecanismo de enclavamiento para garantizar que solamente un dispositivo tome el flanco y que no pueda conectar una fase de servicio cuando ésta ya ha sido concedida.
- La prioridad es fija y viene definida por el orden en que los dispositivos se conectan a la señal de concesión. Cuanto más próximo en la cadena se está del maestro, se tiene una mayor opción a quedarse con la concesión.
- No es muy rápida, puesto que se debe dejar tiempo para que el testigo recorra el máximo número de dispositivos peticionarios previstos.

6.2.4.1.2. Lógica Distribuida. Este método no necesita que el maestro actúe de árbitro en la gestión de la prioridad, por lo que son los propios peticionarios quienes se ponen de acuerdo para decidir quién se queda con el servicio. Un ejemplo de Lógica Distribuida de Prioridades presenta el bus S-100, empleado para interconectar distintas placas: existen 4 líneas de prioridad (DMA0-DMA3) por lo que pueden existir hasta 16 códigos de prioridad. Cada peticionario coloca su nivel de prioridad al mismo tiempo que observa el código presente. Si éste es mayor que el suyo, se retira.

6.2.4.2. Gestión de Prioridad Centralizada

Este tipo de gestión de prioridad puede ser empleado bien cuando la conexión de los dispositivos es mediante multiplexor/demultiplexor o bien mediante bus. Cada peticionario se comunica con el gestor de prioridades mediante una señal de petición (PETI-n) y concesión (CONC-n). Puesto que sólo existe una lógica de control de prioridades, es más fácil poder establecer políticas más complejas que las de prioridad fija, empleada casi exclusivamente en la gestión distribuida.

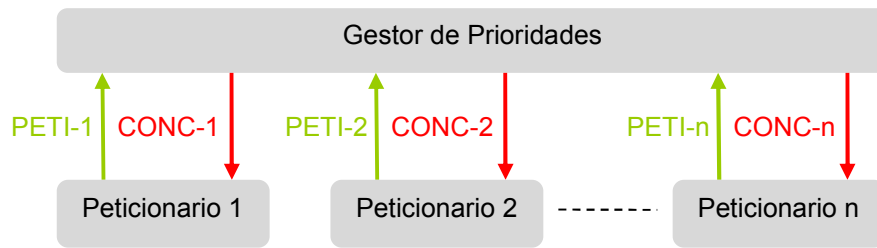


Figura 6.14. Esquema de Gestión Centralizada de Prioridades.

6.2.4.3. Gestión de Prioridad Híbrida

Este método emplea simultáneamente los conceptos de gestión distribuida (mediante el empleo de líneas comunes) y de gestión centralizada (mediante líneas individuales). El mecanismo centralizado establece una serie de niveles de prioridad, a cada uno de los cuales se pueden conectar una serie de peticionarios. Las prioridades dentro de cada nivel se gestionan, por ejemplo, mediante el esquema de encadenamiento.

Un ejemplo de aplicación de este método puede encontrarse en las memorias multipuertas, a excepción de la puerta asignada a la CPU, donde cada puerta sirve para conectar varios peticionarios, por lo que se establece un mecanismo de prioridad entre ellos. Los peticionarios de cada puerta suelen conectarse en forma de bus con prioridad por encadenamiento.

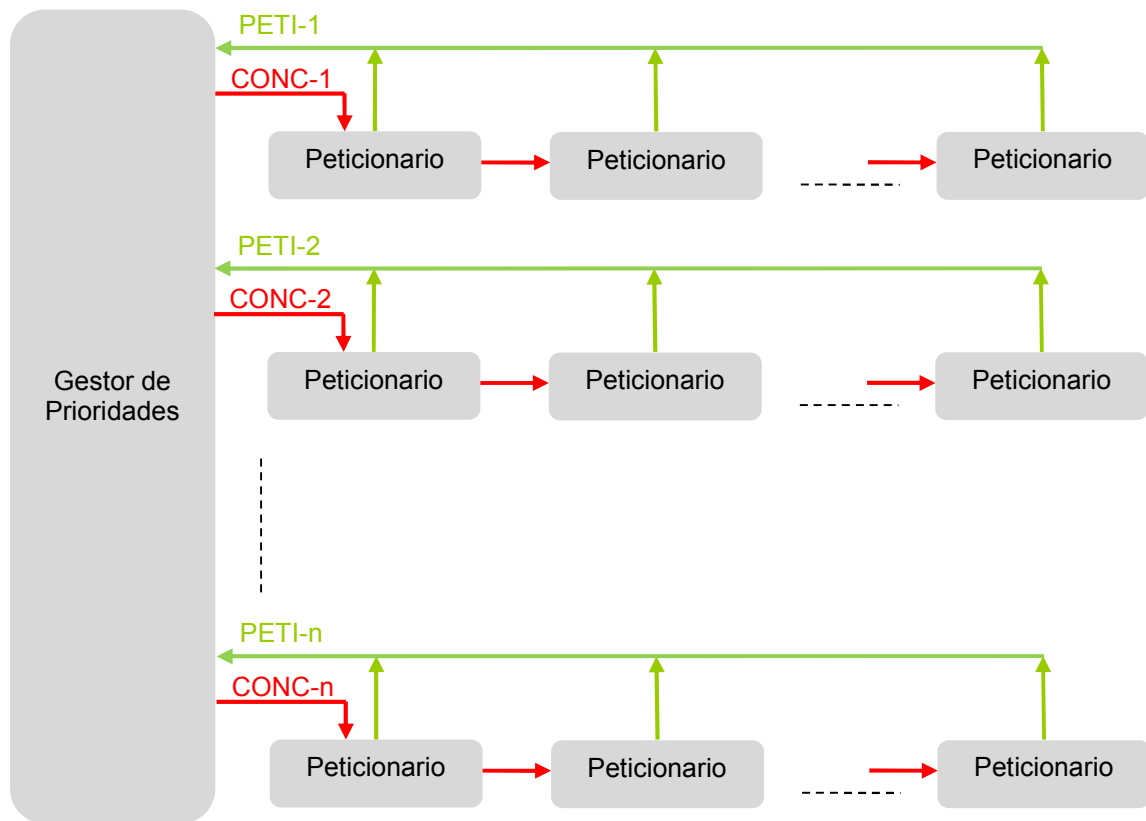


Figura 6.15. Esquema de Gestión Híbrida de Prioridades.

6.2.5. Interrupciones. Selección de la rutina de tratamiento de interrupciones

Las interrupciones son rupturas de secuencia no programadas provocadas por señales externas a la CPU. El objetivo que tiene el dispositivo que provoca la interrupción es reclamar la atención de la CPU sobre algún acontecimiento externo, pidiendo que se ejecute un programa específico que trate adecuadamente dicho acontecimiento. A dicho programa se le denomina **Rutina de Servicio a la Interrupción (RSI)**.

Las etapas de una interrupción son dos:

- Ciclo de concesión de interrupción: Se produce el diálogo entre el dispositivo y la CPU. El dispositivo solicita la interrupción y la CPU acepta dicha interrupción y bifurca a la rutina de servicio a la interrupción. En este sentido, el ciclo de concesión de la interrupción es muy parecido a una instrucción máquina y su duración será equivalente a la de éstas.
- Tratamiento de la interrupción. Consiste en un programa que realiza las operaciones para atender al dispositivo que ha solicitado la interrupción.

La primera fase de la rutina de servicio a la interrupción consiste en salvar el estado del computador, para luego volver a dar el control al programa interrumpido. Dependiendo

del computador, las funciones que realiza directamente el hardware en el ciclo de concesión de interrupción pueden ser más o menos complicadas. La rutina de servicio deberá completar estas funciones.

La CPU comprueba las solicitudes de interrupción al finalizar la ejecución de cada instrucción. Para evitar ciclos infinitos de peticiones de interrupción, una vez que se acepta una interrupción, se inhiben las interrupciones posteriores. Así, cuando se acepta una interrupción se activa un biestable que impide que se acepte ninguna otra. Más tarde, la rutina de servicio a la interrupción vuelve a desactivar ese biestable, permitiendo otra vez la aceptación de interrupciones.

Los mecanismos básicos de conexión para petición de interrupciones han de resolver los siguientes problemas:

- Identificación del peticionario.
- Selección de la rutina de servicio.
- Desactivación de la solicitud de interrupción.
- Superposición de peticiones. El esquema de conexionado ha de permitir que más de un dispositivo tengan activadas sus solicitudes en un mismo instante.
- Resolución de prioridades. En caso de peticiones simultáneas, hay que seleccionar a quien se atiende.
- Anidamiento de interrupciones. El anidamiento correcto ha de permitir que una rutina de servicio solo se pueda interrumpir por peticiones más prioritarias.

Las formas básicas de conexión de dispositivos para producir interrupción son:

- Línea de interrupción única.
- Líneas de interrupción y aceptación.
- Interrupciones vectorizadas.

LÍNEA DE INTERRUPCIÓN ÚNICA

En este caso, el computador dispone de una única línea de interrupción (INT), que sirve para que cualquier dispositivo solicite una interrupción. Adicionalmente se dispone de un biestable general de inhibición de interrupciones (BGII).

En la [Figura 6.16](#) puede observarse que se han agrupado los biestables de interrupción de varios dispositivos en la misma puerta de control representada mediante un buffer triestado, de forma que se acelere el proceso de interrogación.

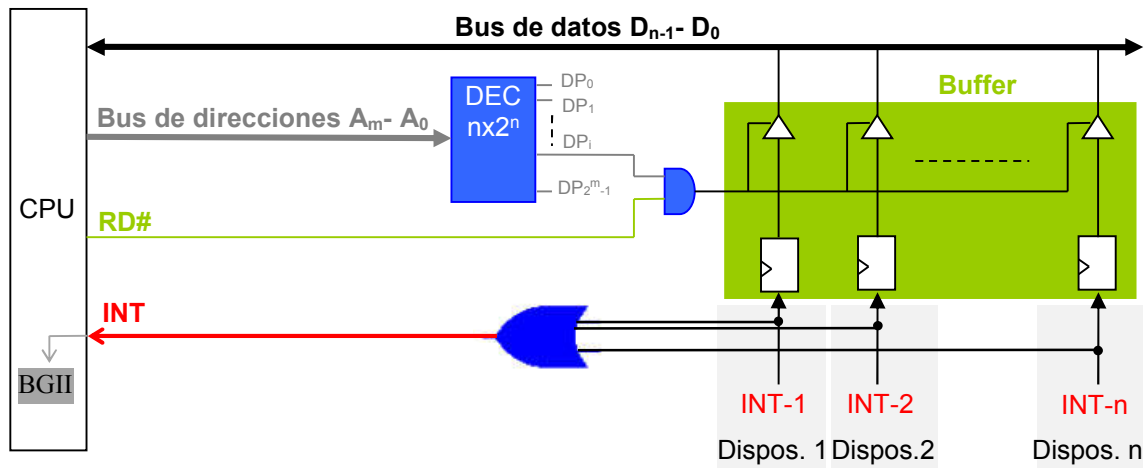


Figura 6.16. Conexión de dispositivos mediante Línea de Interrupción Única.

LÍNEAS DE INTERRUPCIÓN Y ACEPTACIÓN

Esta solución se considera como una extensión de la anterior, a la que se le añade una línea de aceptación de interrupción (INTA). Esta línea de aceptación de interrupción tiene origen en la CPU y llega a cada uno de los dispositivos que pueden generar una interrupción. La activación de esta señal actúa a modo de testigo, que se queda el dispositivo más prioritario.

INTERRUPCIONES VECTORIZADAS

Se habla de interrupciones vectorizadas cuando son los dispositivos que solicitan la interrupción los que proporcionan la dirección de comienzo de la RSI.

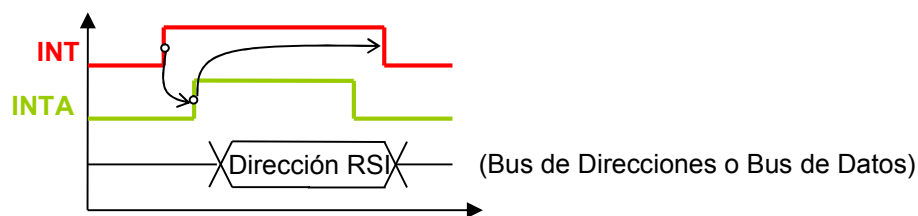


Figura 6.17. Cronograma de Aceptación de Interrupción Vectorizada.

En la [Figura 6.17](#) se muestra un ejemplo de cronograma en el que se aprovecha el flanco de bajada de la señal de aceptación de interrupción INTA para que el dispositivo petionario cancele la señal de petición de interrupción INT.

6.2.5.1. Prioridades y niveles de interrupción

En todo sistema computador con más de un dispositivo capaz de interrumpir debe establecerse un mecanismo de gestión de prioridad. Las funciones que debe resolver este mecanismo son:

- Determinación del dispositivo al que se concede la interrupción cuando existan peticiones simultáneas.
- Permitir que cada programa defina los tipos de interrupciones que puede tolerar, ya que pueden darse las siguientes situaciones:
 - a. Que un programa deba ejecutarse sin sufrir ningún tipo de interrupción.
 - b. Que un programa pueda ser interrumpido por un cierto número o clase de dispositivos.
 - c. Que un programa pueda ser interrumpido por todos los dispositivos.
 - d. Que un programa tenga fases de cada una de las tres categorías anteriores.

Para poder realizar, de forma flexible, la inhibición de las distintas interrupciones, se suelen establecer tres mecanismos distintos:

- Mediante el biestable general de inhibición (BGII).
- Asociando a cada nivel de interrupción, un biestable, llamado máscara, que permite inhibir (enmascarar) de forma selectiva cada nivel.
- Mediante un registro, común a todos a todos los niveles, que contiene el valor del menor nivel que puede interrumpir. Un decodificador se encargará de desactivar todos los niveles de menor prioridad.

Es conveniente emplear los tres mecanismos conjuntamente para dar mayor flexibilidad al sistema. Por ejemplo, se puede asignar a cada programa o proceso un nivel de prioridad, que no pueda automodificar. El nivel de un proceso es fijo pero, si necesita eliminar algunas interrupciones puede usar el mecanismo de la máscara o el de registro.

En la [Figura 6.18](#) se contemplan n niveles de prioridad, cada uno asociado a la pareja de señales INT- i e INTA- i . Contiene los tres mecanismos de inhibición mencionados anteriormente: el registro de máscara (RM), de nivel (RN) y el biestable de inhibición (BGII). Estos elementos representan la información de prioridad de la máquina y pueden leerse o escribirse mediante instrucciones especiales.

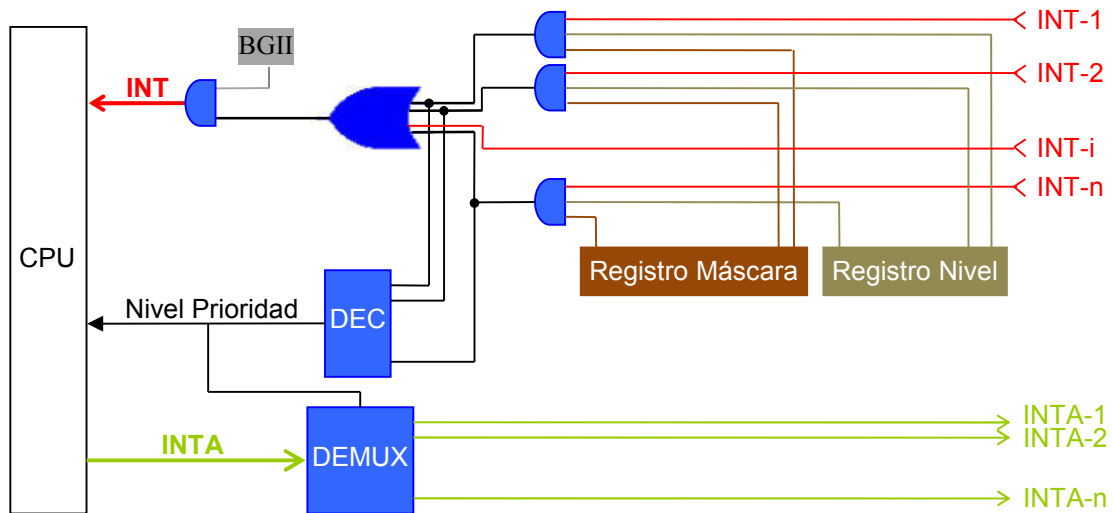


Figura 6.18. Esquema de Generación de Interrupciones y Gestión de Prioridad.

6.2.6. Canales de entrada/salida, procesadores de entrada/salida o unidad periférica de proceso (PPU)

Un canal es una unidad de E/S automática que funciona por acceso directo a memoria (DMA). En este sentido, el controlador necesario para la lectura de la unidad de disquette por robo de ciclo es un canal. Sin embargo, en esta sección se aplicará el término de canal de forma más restrictiva, puesto que se impondrán las siguientes condiciones:

- Acceso directo a memoria, preferentemente por propia puerta.
- Comunicación con la CPU a través de memoria y no por E/S programada.

El mecanismo de comunicación entre la CPU y el canal suele organizarse de forma que se dispone de un puntero PCANAL, ubicado en una posición fija de memoria, que contiene la dirección de memoria a partir de donde se encuentran los DATOS implicados en la operación de E/S.

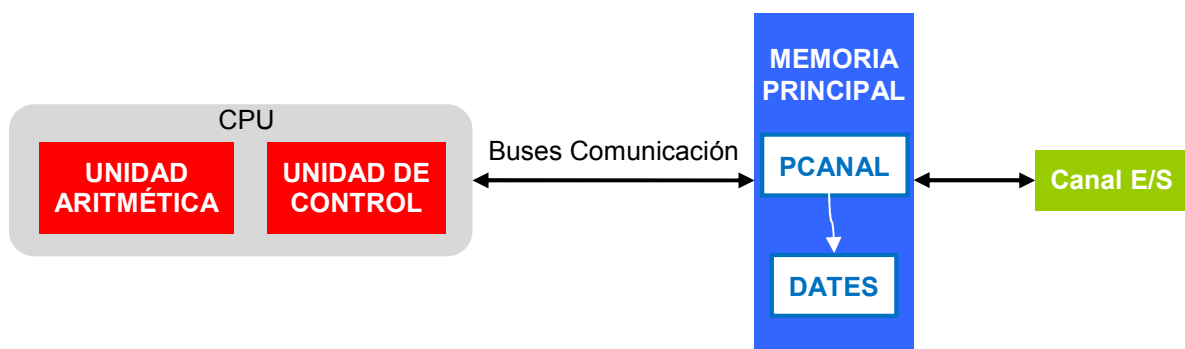


Figura 6.19. Comunicación CPU-CANAL.

Estos datos están ubicados en la zona llamada DATES y consisten en:

- La dirección de memoria principal a partir de donde se leerá/almacenará la información correspondiente a la operación que realiza el canal.
- La dirección asociada al periférico (e indicación del periférico en el caso de existir varios periféricos conectados a un mismo canal).
- El número de bytes implicados en la operación.
- El tipo de operación (entrada o salida).
- El bit de interrupción (si está activado, el canal interrumpe cuando ha finalizado la operación).
- El bit de encadenamiento.
- El balance de la operación, esta información la define el canal al finalizar la operación de E/S correspondiente.

ENCADENAMIENTO DE LAS OPERACIONES DE E/S

El encadenamiento se puede realizar sin más que añadir al final de la primera zona de datos DATES1, otra nueva zona DATES2 ([Figura 6.20](#)). El canal, al terminar la operación con DATES1 introduce el balance de dicha operación y lee la información almacenada en DATES2. El proceso se repite hasta encontrar la indicación de fin. Las zonas DATES1, DATES2,..., DATESn se encadenan mediante un bit de encadenamiento que está activo mientras existan nuevas zonas de datos consecutivas. Al conjunto de DATES se le suele llamar **PROGRAMA CANAL**. La CPU va generando dinámicamente el programa de canal, a medida que van surgiendo las necesidades de E/S.

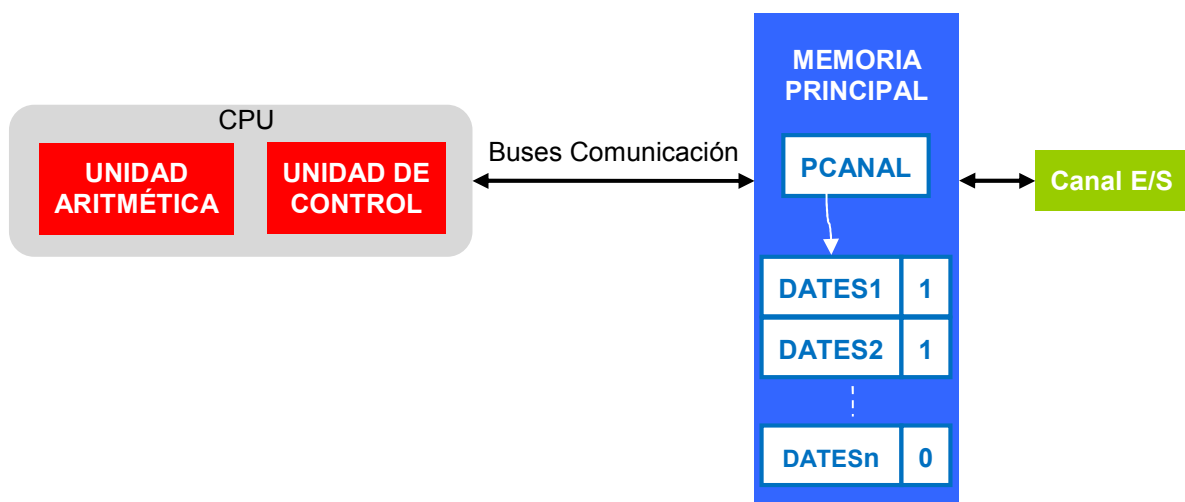


Figura 6.20. Encadenamiento de órdenes de canal.

6.2.7. Entrada/salida y sistema operativo

En este apartado se destacan aspectos que revisten la ejecución y control de una operación completa de E/S.

Supóngase que se trata de un fichero de datos con los que se pretende trabajar. El programador se limita a escribir una sentencia de apertura del fichero y, seguidamente, programará las instrucciones de lectura o de escritura. Las sentencias de alto nivel, que expresan operaciones de E/S, se compilan como llamadas al sistema operativo, siendo éste el responsable de llevar a cabo estas operaciones. Para ello, el sistema operativo ha de ser capaz de realizar las siguientes funciones:

- Manejar los directorios de los periféricos.
 - a. Convertir el nombre simbólico en direcciones físicas.
 - b. Insertar nuevos ficheros.
 - c. Borrar un fichero.
- Generar las órdenes a los periféricos, elaborando en su caso los programas de canal.
- Gobernar los periféricos, enviándoles órdenes y atendiendo a sus necesidades y problemas.
- Ordenar las peticiones de E/S, manteniendo las colas necesarias y generando mensajes a los programas peticionarios.
- Tratar las situaciones de error, repitiendo las operaciones en caso necesario y avisar al usuario.
- Asignar las zonas de memoria necesarias para estas operaciones, realizar la conversión de código y agrupación de octetos en palabra (y viceversa).

Las rutinas de E/S del sistema operativo son, entre otras:

- Leer el sector x del dispositivo y .
- Escribir en el sector x del dispositivo y .
- Leer un carácter del teclado.
- Leer un carácter de una puerta serie.
- Enviar un carácter a una puerta serie.
- Borrar una pantalla.
- Rebobinar una cinta magnética.

El usuario debe construirse sus programas de E/S con las rutinas que facilita el sistema operativo. A la rutina de manejo de cada periférico se le suele llamar manejador o driver.

Por último, indicar que el sistema operativo debe ser capaz de realizar la función “SPOOL” (Simultaneous Peripheral Operation On Line). Esta función se emplea en las

operaciones con periféricos lentos. Consiste en usar un almacenamiento intermedio o buffer ubicado en el disco, con el objeto de hacer un desacoplo entre las velocidades de operación de E/S y el programa que la solicita.

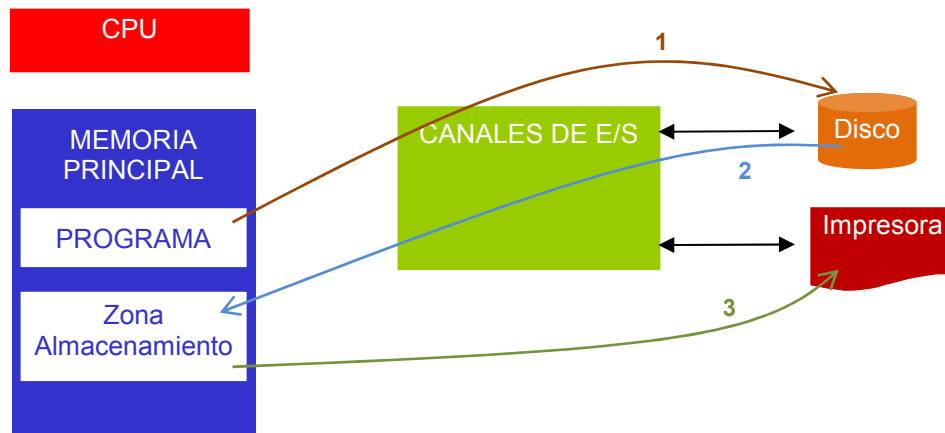


Figura 6.26. Función SPOOL del Sistema Operativo.