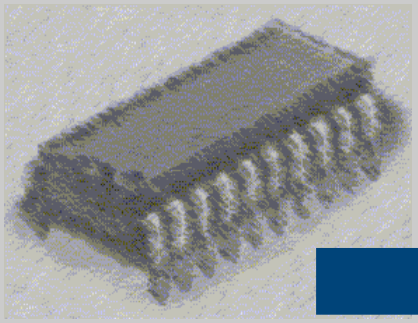
Soluciones a los problemas impares

Tema 6. El sistema de entrada-salida

Arquitectura de Computadores I



I. T. Informática de Sistemas

Curso 2009-2010

Base teórica

Al diseñar un computador, uno de los puntos a tener en cuenta es el diseño del sistema de entrada-salida, ya que de él dependerá el rendimiento con el que el computador se comunicará con los periféricos, y en definitiva, con el mundo exterior, no digital (figura 1)

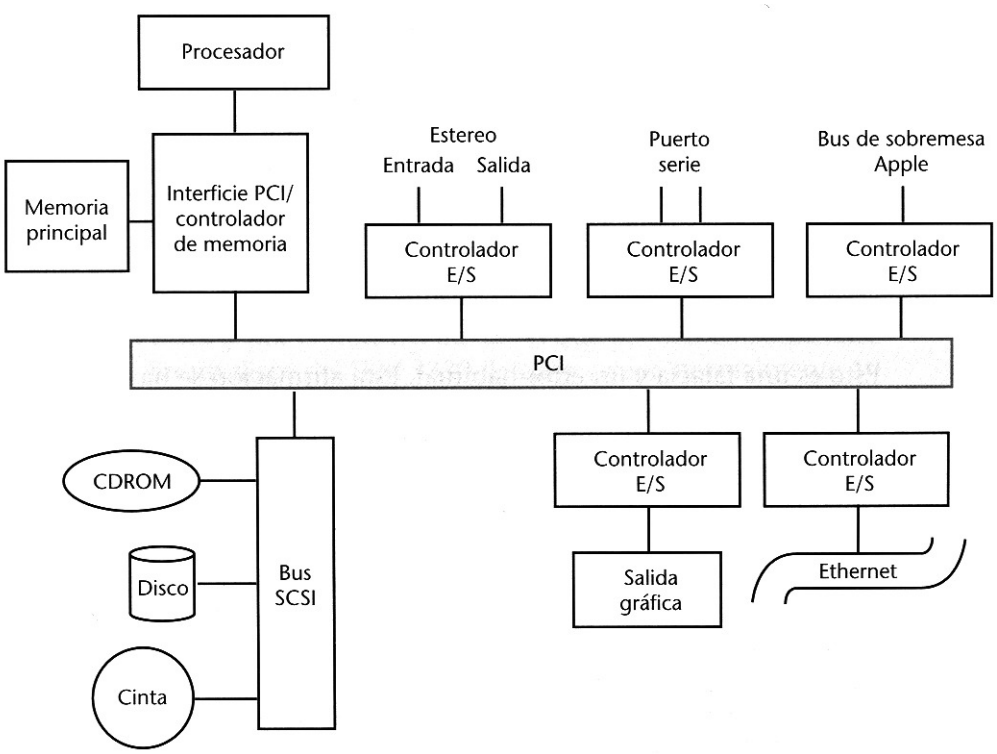


Figura 1. Sistema de entrada-salida del Apple Macintosh 7200

Los periféricos los hay de entrada, de salida y de entrada-salida. Destinados a la comunicación con el ser humano o con otros computadores, pero todos ellos constan de dos partes claramente diferenciadas, el controlador y el dispositivo.

El controlador del periférico es el encargado de la comunicación con la CPU. El que lleva a cabo el protocolo de comunicación y el que realiza la transferencia de datos propiamente dicha.

El dispositivo es realmente el periférico, electrónico una memoria USB, electromecánico como un disco duro, un trasductor como en el caso de los sensores de temperatura o movimiento.

Gestión de la comunicación CPU-periféricos

En la gestión de la comunicación de la entrada salida se deben tener en cuenta tres aspectos:

- **Selección del periférico:** identificar el periférico entre todos los disponibles mediante el direccionamiento de los registros de datos y de control
- **Sincronización con la CPU:** evitar los problemas surgidos de las diferentes velocidades de trabajo de la CPU y de los periféricos. Evitar inundar al periférico con datos provenientes de la CPU y evitar que ésta se quede esperando los datos del periférico
- **Gestión de las señales de control:** determinar para cada tipos de sincronización y cada tipo de selección de periférico el conjunto de señales de control que deberán ser tenidos en cuenta por la Unidad de Control para la correcta gestión de la comunicación

Selección del periférico

Para elegir un periférico y comunicarse con él se emplean los puertos de entrada-salida que en su versión más simple se corresponden con un registro

Para comunicarse con un periférico la CPU debe:

- **Especificar la dirección:** identificar el puerto de entrada-salida entre todos los que tenga (normalmente tienen varios puertos de entrada salida)
- **Indicar el sentido de la comunicación:** especificar si se trata de una lectura o de una escritura.
- Enviar el dato: si se trata de una escritura

Para especificar las direcciones de los puertos existen dos métodos:

- **Espacio de direcciones separado:** los espacios de direccionamiento de puertos y de direcciones de memoria son diferentes por lo que se deberán incluir instrucciones propias de entrada-salida. Por ejemplo, en el i80x86 instrucciones IN y OUT
- **Espacio de direcciones común:** tanto direcciones de memoria como de puertos de entrada-salida forman un espacio de direcciones común por lo que las instrucciones del repertorio son las mismas para ambos tipos de transferencia

Sincronización con la CPU

El objetivo de la sincronización es ajustar la diferencia de velocidades de proceso de la CPU y de los periféricos. Los mecanismos más empleados según el uso de CPU (de más implicación a menos) son:

- **Entrada-salida programada:** la CPU gestiona la comunicación con los periféricos mediante la ejecución de un programa (figura 2)

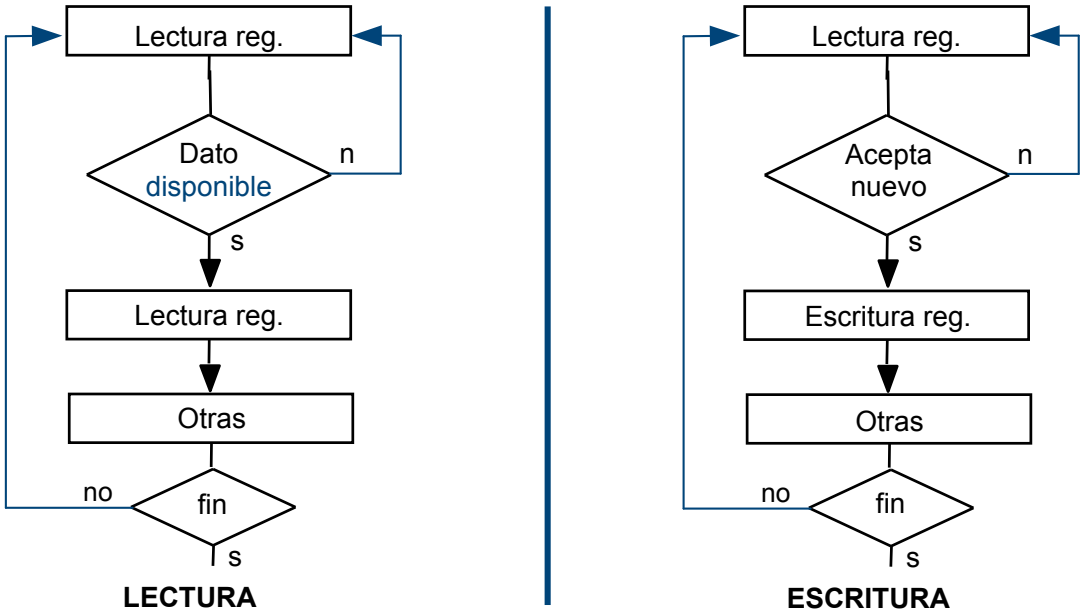


Figura 2. Cronograma de los programas de lectura y escritura mediante entrada-salida programada para un único periférico

- **Entrada-salida por interrupciones:** los periféricos avisan a la CPU cuando están listos para ser atendidos. Cuando la interrupción es aceptada por la CPU, ésta abandona momentáneamente el programa principal para ejecutar la rutina de tratamiento de la interrupción y realiza la transferencia. Dependiendo del número de líneas que existen para comunicarse con la CPU se pueden dar los siguientes casos:
    - Todos los periféricos solicitan ser atendidos por la misma línea. Por lo tanto la dirección de la rutina de tratamiento de la interrupción es fija y común para todos ellos. Dicha rutina de tratamiento identifica mediante sondeo cuál de los periféricos ha interrumpido a la CPU. La prioridad, en el caso de varios periféricos se da por el orden en el que se sondea a éstos.
    - CPU con varias líneas de interrupción. En este esquema cada periférico solicita ser atendido por una línea diferente. La dirección de la rutina de tratamiento de la interrupción es fija pero diferente para cada línea de interrupción. La prioridad la determina internamente la CPU
    - CPU con una línea de interrupción y otra de aceptación. La CPU cuenta con una línea de entrada INT por la que todos los periféricos solicitan ser atendidos. La CPU tiene una línea de salida INTA para dar el reconocimiento de la interrupción al periférico. Se debe determinar cómo se conectan los periféricos y la gestión de las prioridades en caso de solicitud simultánea. Asimismo se debe determinar la dirección de la rutina de tratamiento de interrupción. La resolución de prioridades puede darse mediante daisy-chain, por la posición que ocupan o mediante la ayuda de un PIC.
- **Entrada-salida mediante acceso directo a memoria (DMA):** los periféricos trabajan directamente con la memoria escribiendo o leyendo bloques de datos

- **Entrada-salida mediante procesadores de entrada-salida:** emplean una CPU secundaria dedicada a la gestión de la operación de entrada-salida.

Buses

Un bus es un canal de comunicación compartido que emplea un conjunto de cables para conectar los diferentes subsistemas. Están formados por:

- Un conjunto de señales de control
- Un conjunto de líneas de datos

Ventajas:

- Versatilidad, ya que una vez definido el sistema de conexión se pueden añadir nuevos dispositivos
- Bajo coste, puesto que los cables que lo forman se pueden compartir de diferentes maneras

Inconvenientes:

- Es un cuello de botella que limita la máxima productividad de entrada-salida del sistema
- Dificultad de diseño debida a que depende del número de elementos que se conectarán y a la longitud del bus

Los buses se clasifican en tres tipos

- **Buses procesador-memoria**. Son de longitud pequeña, de alta velocidad y adaptados al sistema de memoria para maximizar el ancho de banda
- **Buses backplane.** Suelen recibir este nombre por estar construidos en el plano posterior. Es una estructura de conexión en el chasis y la memoria, la CPU y los periféricos lo emplean para comunicarse
- **Buses de entrada-salida.** Pueden tener una longitud grande y sirven para conectar muchos dispositivos con anchos de banda muy diferentes

entre si. Además para comunicarse con la memoria suelen emplear o buses procesador-memoria o buses backplane

Entre las cosas a tener en cuenta para mejorar el ancho de banda del bus y que en su mayor parte vienen determinadas casi por la elección de un protocolo síncrono o asíncrono y por las características de temporización del bus, existen otros factores que afectan al ancho de banda y que son:

- **Anchura del bus de datos.** Al aumentar la anchura del bus de datos se pueden transmitir varias palabras en menos ciclos de bus
- **Líneas de datos y direcciones separadas o multiplexadas.** Si las líneas están separadas se aumenta el ancho de banda en las escrituras
- **Transferencias por bloques.** Si se transfieren múltiples palabras en ciclos consecutivos de bus sin enviar la dirección o sin liberar el bus se reduce el tiempo de transmitir un bloque grande de datos

La temporización se refiere a la manera en que se realizan las fases de direccionamiento y de datos durante la transferencia. Pueden ser:

- **Bus de ciclo completo.** Las fases de direccionamiento y de datos van seguidas sin abandonar el bus. El diseño es sencillo, pero presenta una menor latencia y un ancho de banda más pequeño
- **Bus de ciclo partido.** El bus se abandona entre las fases de direccionamiento y de datos. Permite la presencia de varios maestros de bus. Aumenta la latencia y el ancho de banda y que se transmita la identidad del maestro de bus. Por último, los exclavos deben ser capaces de solicitar el bus

En los buses se debe tener un control de acceso al mismo, de manera que no se produzcan conflictos al acceder varios elementos simultáneamente. El control de acceso es una manera de reservar el bus por parte de un dispositivo. El mecanismo tener uno o más maestros del bus y varios esclavos.

- **Maestro de bus.** Controla el acceso al bus. Debe ser capaz de iniciar y controlar todas las peticiones de bus.
- **Esclavo de bus.** Responde a las peticiones del maestro, o de los maestros, pero nunca genera peticiones.

También se debe determinar un arbitraje del bus cuando tenemos un esquema de varios maestros y se debe decidir cuál es el siguiente maestro que puede usar el bus.

Se debe garantizar que el maestro más prioritario sea atendido cuanto antes, pero que el menos prioritario no se quede sin su acceso al bus

Los métodos de arbitraje más empleados son:

- **Arbitraje en daisy-chain.** Una línea de concesión recorre todos los dispositivos
- **Arbitraje centralizado.** Un árbitro centralizado selecciona al dispositivo y le nombra maestro del bus
- **Arbitraje distribuido por auto selección.** Los dispositivos indican la prioridad de manera que el más prioritario se erige en maestro
- **Arbitraje distribuido por detección de colisión.** Una vez detectada la colisión se emplea un esquema para seleccionar al maestro entre los dispositivos que causaron la colisión

La relación rendimiento-coste se refleja en el resumen siguiente (figura 3):

Opción	Alto rendimiento	Bajo coste
Anchura de bus	Líneas separadas de datos y direcciones	Multiplexado de las líneas
Anchura de datos	Más ancho implica más rápido	Más estrecho implica más barato
Tamaño de la transferencia	Transferir múltiples palabras disminuye la sobrecarga del bus	Transferir una única palabra es más sencillo

Opción	Alto rendimiento	Bajo coste
Número de maestros del bus	Múltiples maestros, requiere arbitraje	Un único maestro
Sincronización	Síncrono	Asíncrono
Temporización	Bus de ciclo partido	Bus de ciclo completo

Figura 3. Relación rendimiento-coste para las propiedades de los buses

Entrada-salida y sistema operativo

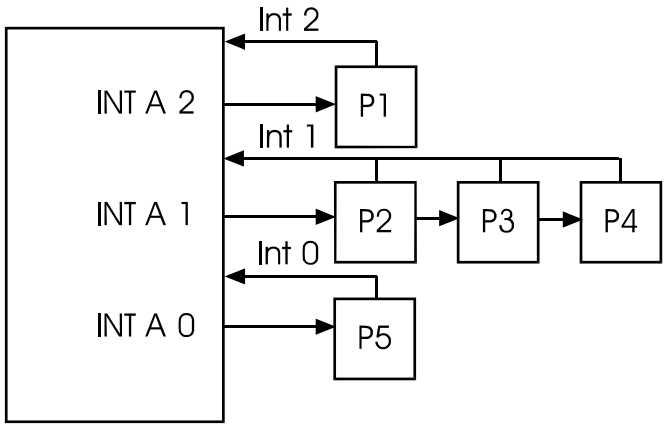
El sistema operativo debe tener en cuenta:

- Que el sistema de entrada-salida es compartido por múltiples programas que usan el procesador
- Que un programa de usuario solamente accede a los dispositivos de entrada-salida sobre los que el usuario tiene acceso
- Que los sistemas de entrada-salida emplean frecuentemente interrupciones para comunicar información sobre las operaciones de entrada-salida y que deberán ser atendidas por el sistema operativo
- Que el control de bajo nivel de un dispositivo es complejo y que deben ser simplificados al programador mediante las llamadas al sistema operativo
- Que se dé un acceso equitativo a los recursos de entrada-salida compartidos e intentar planificar los accesos para mantener la productividad del sistema

Importante, en los sistemas de entrada-salida se emplea la base 10 en vez de la 2 para hacer referencia a los 1 KB (1000 bytes), 1 MB = 10^6 bytes)

1. Sea un computador que dispone de un sistema de interrupciones con tres niveles de prioridad, correspondiendo el nivel 2 a la máxima y el 0 a la mínima. Cuando se atiende una interrupción de nivel *n* se inhibe automáticamente la atención a las interrupciones de ese nivel y de todos los niveles inferiores, y se activa durante una unidad de tiempo la señal de reconocimiento de interrupción de ese nivel INTA *n* a la vez que se ejecuta la primera instrucción de la rutina de servicio correspondiente RUT *n*.

   A este computador se han conectado cinco periféricos según el esquema siguiente:



   El periférico P1 solicita interrupción en instantes arbitrarios, mientras que el resto lo hacen cuando tienen un nuevo dato que suministrar. Todos los periféricos mantienen activa la señal de solicitud de interrupción hasta que llegue la señal de reconocimiento correspondiente. El esquema de conexión de los periféricos P2, P3 y P4 es un *daisy-chain.*

Las rutinas de servicio de interrupción correspondientes a cada nivel son las siguientes:

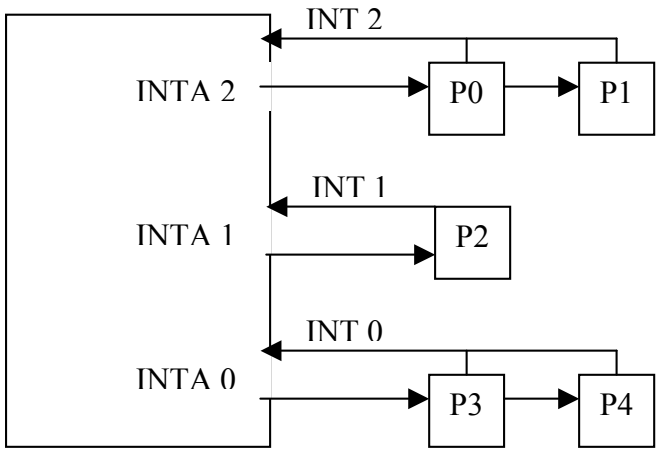
Nº.	Pseudocódigo de la instrucción	Rutina
1	Guardar el registro A en la pila	Rut 2
2	Cargar A con el valor del contador	
3	Incrementar A	
4	Actualizar el contador con el valor de A	
5	Recuperar el valor de A de la pila	
6	Habilitar las interrupciones de nivel 2	
7	Retorno de interrupción	
10	Guardar el registro A en la pila	Rut 1
11	Guardar los flags en la pila	
12	Leer del puerto de entrada y almacenar en el registro A	
13	Cargar en MEM_P2 el valor del registro A	
14	Incrementar A	
15	Habilitar las interrupciones de los niveles 1 y 2	
16	Leer del puerto de entrada y almacenar en el registro A	
17	Incrementar A	
18	Cargar en MEM_P2 el valor del registro A	
19	Recuperar los *flags* de la pila	
20	Recuperar el valor de A de la pila	
21	Retorno de interrupción	
30	Guardar el registro A en la pila	Rut 0
31	Cargar en A el valor de MEM_P4	
32	Leer el puerto de entrada y almacenar en el registro A	
33	Cargar en MEM_P4 el valor del registro A	
34	Recuperar el valor de A de la pila	
35	Habilitar todas las interrupciones	
36	Retorno de interrupción	
100	Habilitar las interrupciones de nivel 2	Principal
101	Poner a cero la variable contador	
102	Cargar la variable MEM_P2 con 100	
103	Cargar la variable MEM_P2 con 150	
104	Cargar la variable MEM_P2 con 200	
105	Habilitar todas las interrupciones	
106	Sumar los registros A y D	
107	Mostrar resultado en pantalla	

Los periféricos solicitan interrupciones en los siguientes tiempos:

Periférico	Instante 1ª. Solicitud	Tiempo entre solicitudes
P_1	20'2	20'2 – 24'7 – 50
P_2	4'5	Cada 40
P_3	3'25	Cada 42
P_4	0'8	Cada 35
P_5	2	Cada 49

a) Sabiendo que todas las instrucciones duran una unidad de tiempo, realizar la traza de ejecución entre los instantes 0 y 31, suponiendo que la CPU comienza a ejecutar la primera instrucción del siguiente programa en el instante 0.

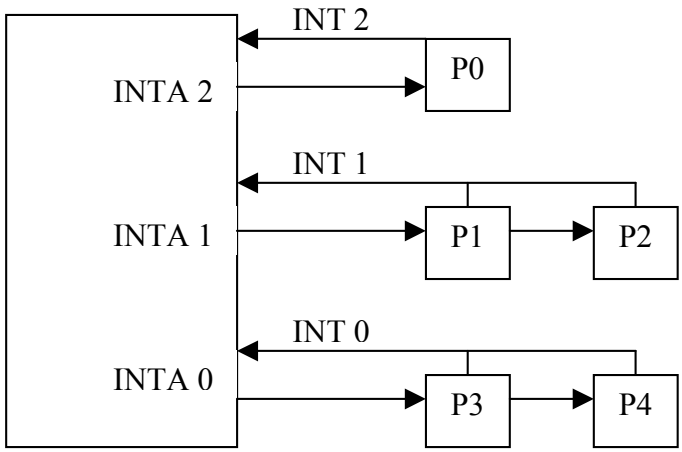
2. Repetir el ejercicio anterior entre los instantes 0 y 31 pero asumiendo el siguiente esquema de conexión de los periféricos



3. Sea un computador que dispone de un sistema de interrupciones con tres niveles de prioridad, correspondiendo el nivel 2 a la máxima y el 0 a la mínima. Cuando se atiende una interrupción de nivel *n* se inhibe automáticamente la atención a las interrupciones de ese nivel y de todos los niveles inferiores, y se activa durante una unidad de tiempo la señal de reconocimiento de interrupción de ese nivel INTA *n* a la vez

que se ejecuta la primera instrucción de la rutina de servicio correspondiente RUT *n*.

A este computador se han conectado cinco periféricos según el esquema siguiente:



El periférico P0 solicita interrupción en instantes arbitrarios, mientras que el resto lo hacen cuando tienen un nuevo dato que suministrar. Todos los periféricos mantienen activa la señal de solicitud de interrupción hasta que llegue la señal de reconocimiento correspondiente. El esquema de conexión de los periféricos P1 y P2 y P3 y P4 es un daisy-chain respectivamente.

Las rutinas de servicio de interrupción correspondientes a cada nivel y el programa principal se muestran en la tabla siguiente:

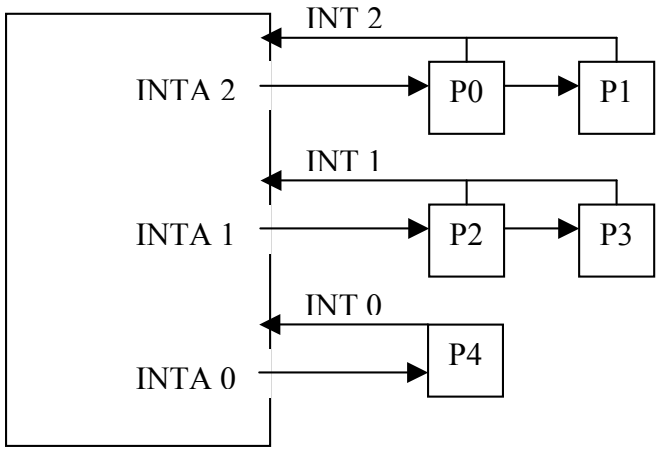
Nº	Pseudocódigo	Rutina
1	Cargar A con el valor del contador	Rut 2
2	Actualizar el contador con el valor de A	
3	Habilitar las interrupciones de nivel 2	
4	Retorno de interrupción	
11	Leer del puerto de entrada y almacenar en el registro A	Rut 1
12	Habilitar las interrupciones de los niveles 1 y 2	
13	Incrementar A	
14	Retorno de interrupción	
21	Leer el puerto de entrada y almacenar en el registro A	Rut 0
22	Cargar en Índice el valor del registro A	
23	Habilitar todas las interrupciones	
24	Retorno de interrupción	
100	Habilitar las interrupciones de nivel 2	Programa Principal
101	Poner a cero la variable *contador*	
102	Poner a cero la variable *Índice*	
103	Habilitar todas las interrupciones	
104	Sumar los registros A y D	
105	Mostrar resultado en pantalla	
106	Borrar variables temporales	
107	Terminar programa	

Los periféricos solicitan interrupciones en los siguientes tiempos:

Periférico	Instante 1ª. Solicitud	Tiempo entre solicitudes
P_0	2	Cada 49
P_1	22’2	50
P_2	6’5	Cada 40
P_3	3’25	Cada 42
P_4	0’8	Cada 20

Sabiendo que todas las instrucciones duran una unidad de tiempo, realizar la traza de ejecución entre los instantes 0 y 31, suponiendo que la CPU comienza a ejecutar la primera instrucción del programa principal en el instante 0.

4. Repetir el ejercicio anterior entre los instantes 0 y 31 pero asumiendo el siguiente esquema de conexión de los periféricos



5. Sea un computador capaz de ejecutar 10 MIPS (10^7 instrucciones por segundo) Se desea conectar al computador, únicamente un periférico con una velocidad de transferencia de 20.000 bytes/sg. y sobre el que se realizan operaciones de lectura de bloques de 1.024 bytes.

   Se pretende ver el comportamiento de la pareja computador-periférico ante las diferentes técnicas de entrada-salida (programada, mediante interrupciones y por DMA)

   Se sabe que:

   - La rutina de transferencia de E/S programada consta de 10 instrucciones.
   - La rutina de tratamiento de interrupción en la E/S mediante interrupciones consta de 20 instruciones.
   - La rutina de inicializacón del DMA consta de 8 instrucciones. Y en cada operación de escritura de un dato en memoria el controlador ocupa los buses durante 500 ns.

Se pide:

Indicar el número de instrucciones de otros procesos que puede realizar el computador durante cada uno de los tipos de E/S previstos.

---

6. Se desea dotar a un edificio de investigación de un sistema de alarma que cuente con 112 sensores de movimiento. Dichos sensores proporcionan una salida de 0 ó 3 voltios. Para ahorrar costes se ha pensado controlar los sensores con un i8086, dado que la velocidad de un posible intruso es muy inferior a la de un microprocesador. Dicho procesador es un micro de 8 bits.

   Se pide:

   Diseñar la estructura de E / S del sistema, tanto el mapa de E / S como las posibles conexiones. Emplee para ello los elementos que crea necesarios

---

7. Se desea controlar con un i8086 un sistema de control de 2.000 sensores, tipo abierto-cerrado, de una fábrica de motores para lavadora. Si cada sensor tiene asociado un biestable cuya información es la que se desea leer.

   Se pide:

   Diseñar la estructura de E / S del sistema, tanto el mapa de E / S como las posibles conexiones. Emplee para ello los elementos que crea necesarios

---

8. Se desea controlar la temperatura de una bodega de ron añejo. Para ello, se han colocado 1023 sensores de temperatura repartidos por toda la bodega. El objeto de estos sensores es enviar una salida de entre 0 y 5 voltios si se supera una determinada temperatura (asignada por

hardware y sin relevancia para el problema) Para ahorrar costes se ha pensado controlar los sensores con un i8086, dado que la temperatura no cambia demasiado rápidamente dentro de la bodega se empleará este micro de 16 bits.

Se pide:

Diseñar la estructura de E / S del sistema, tanto el mapa de E / S como las posibles conexiones. Emplee para ello los elementos que crea necesarios

9. Se tiene un sistema computador con las características siguientes:
    - Una CPU que ejecuta 300 millones de instrucciones por segundo.
    - Un bus de memoria con una velocidad de transferencia de 100 MB/sg.
    - Controladores SCSI-2 con velocidad de transferencia de 20 Mb/sg que permiten la conexión de hasta siete discos
    - Unidades de disco con ancho de banda de lectura/escritura de 5 Mb/sg.

    Suponiendo que:

    - La carga de trabajo consiste en lecturas de bloques de 64Kb y que un programa de usuario necesita 100.000 instrucciones por cada operación de E/S.
    - Que cada operación de E/S emplea una media de 50.000 instrucciones de sistema operativo.
    - Y que las lecturas siempre se pueden realizar en un disco inactivo, si es que existe (es decir, se deben ignorar los conflictos en los discos)

Se pide encontrar la máxima velocidad de E/S que puede mantenerse y el número de discos duros y controladores SCSI-2 necesarios en ese caso.

10. Calcular el tiempo medio de lectura o escritura de un sector de 512 bytes en un disco duro que gira a 5.400 r.p.m. suponiendo que:
    - El tiempo medio de posicionado es de 18 ms.
    - La velocidad de transferencia es de 5 MB/sg.
    - La sobrecarga debida al controlador es de 3 ms.
    - No existe tiempo de espera porque el disco está desocupado

11. Calcular el tiempo medio de lectura o escritura de un sector de 1024 bytes en un disco duro que gira a 3.600 r.p.m. suponiendo que:
    - El tiempo medio de posicionado es de 12 ms.
    - La velocidad de transferencia es de 10 MB/sg.
    - La sobrecarga debida al controlador es de 2 ms.
    - No existe tiempo de espera porque el disco está desocupado

12. Se desea comparar los anchos de banda máximos de un bus síncrono y otro asíncrono. El bus síncrono tiene un tiempo de ciclo de reloj de 40 ns. Y cada transacción requiere 1 ciclo de reloj. El bus asíncrono requiere 40 ns. Para el protocolo de *handshaking*. En ambos buses, la sección de datos tiene una anchura de 16 bits.

Se pide:

Calcular el ancho de banda de ambos buses cuando realizan lecturas de una memoria de 200 ns. suponiendo que las lecturas siempre son de una palabra

13. Se desea comparar los anchos de banda máximos de un bus síncrono y otro asíncrono. El bus síncrono tiene un tiempo de ciclo de reloj de 50 ns. Y cada transacción requiere 1 ciclo de reloj. El bus asíncrono requiere 20 ns. Para el protocolo de *handshaking*. En ambos buses, la sección de datos tiene una anchura de 32 bits.

    Se pide:

    Calcular el ancho de banda de ambos buses cuando realizan lecturas de una memoria de 100 ns. suponiendo que las lecturas siempre son de una palabra

14. Se desea comparar los anchos de banda máximos de un bus síncrono y otro asíncrono. El bus síncrono tiene un tiempo de ciclo de reloj de 20 ns. Y cada transacción requiere 1 ciclo de reloj. El bus asíncrono requiere 20 ns. Para el protocolo de handshaking. En ambos buses, la sección de datos tiene una anchura de 16 bits.

    Se pide:

    Calcular el ancho de banda de ambos buses cuando realizan lecturas de una memoria de 30 ns. suponiendo que las lecturas siempre son de una palabra

15. Se desea comparar los anchos de banda máximos de un bus síncrono y otro asíncrono. El bus síncrono tiene un tiempo de ciclo de reloj de 30 ns. Y cada transacción requiere 1 ciclo de reloj. El bus asíncrono

requiere 25 ns. Para el protocolo de handshaking. En ambos buses, la sección de datos tiene una anchura de 32 bits.

Se pide:

Calcular el ancho de banda de ambos buses cuando realizan lecturas de una memoria de 60 ns. suponiendo que las lecturas siempre son de una palabra

Solución ejercicio 1

La capacidad de la memoria principal es de 2^{32} bytes, y se encuentra estructurada en bloques de tamaño 16 bytes = 2^4 bytes/bloque, así se puede calcular que el número de bloques de memoria principal es de:

Rut 2															
Rut 1							**10**	**11**	**12**	**13**	**14**	**15**	*10*	*11*	*12*	*13*
Rut 0																
Principal	100	101	102	103	104	105										
Instante	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Solicitud		**P_4**	**P_5**		**P_3**	**P_2**										
Mín. Int.	**2**	**2**	**2**	**2**	**2**	**0**	**2**	**2**	**2**	**2**	**2**	**1**	**2**	**2**	**2**	**2**

Rut 2							**1**	**2**	**3**	**4**	**5**	**6**	*1*	*2*	*3*	*4*
Rut 1	*14*	*15*	**10**	**11**	**12**	**13**										
Rut 0																
Principal																
Instante	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Solicitud						**P_1**				**P_1**						
Mín. Int.	**2**	**1**	**2**	**2**	**2**	**2**	**X**	**X**	**X**	**X**	**X**	**2**	**X**	**X**	**X**	**X**

Como se observa, en la franja de tiempos pedida, ninguno de los periféricos logra terminar sus peticiones

Solución ejercicio 3

Rut 2				1	2	3	4									
Rut 1									**11**	**12**	**13**	**14**				
Rut 0														**21**	**22**	**23**
Principal	100	101	102					103					104			
Instante	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Solicitud		**P_4**	**P_0**		**P_3**			**P_2**								
Mín. Int.																

Rut 2																
Rut 1									11	12	13	14				
Rut 0	*21*	*22*	*23*	*24*	**24**		*21*	*22*					*23*	*24*		
Principal						105									106	107
Instante	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Solicitud						**P_4**		**P_1**								
Mín. Int.																

En este caso todos los periféricos son atendidos entre los instantes 0 y 31

Solución ejercicio 5

El periférico suministra 20.000 bytes/sg. Enviará un dato cada 50 µsg.

Como los bloques son de 1.024 bytes, una operación de E/S durará:

50µsg/dato. x 1.024 datos = 51.200µsg.

E/S programada

El periférico proporcionará un dato cada 50µsg. La CPU en ese tiempo es capaz de ejecutar 500 instrucciones (ya que el computador trabaja a 10 MIPS). La mayor parte de las instrucciones estarán dedicadas a realizar el muestreo sobre el periférico.

La rutina de E/S programada consta de 10 instrucciones, pero el tiempo de ejecución es despreciable frente las 500 instrucciones que es capaz de realizar la CPU en el tiempo de transmitir un dato. Y dado que en transmitir los 1.024 bytes se emplean 51.200µsg.las instrucciones que podría haber realizado la CPU serían 512.000. Además, al ser entrada-salida programada, la CPU está plenamente dedicada a ejecutar las instrucciones de la rutina de entrada-salida, con lo que no es capaz de ejecutar ninguna instrucción de otro proceso.

E/S mediante interrupciones

La rutina de interrupción consta de 20 instrucciones, y se ejecutará cada vez que está lista para entregarnos un dato nuevo. Al ser los bloques de 1.024 bytes, ejecutaremos un total de 20 x 1.024 =20.480 instrucciones dedicadas a la transferencia de datos, con lo que la CPU podría dedicar tiempo a otros procesos.

El número de instrucciones de otros procesos que puede ejecutar la CPU es de:

512.000 instrucciones - 20.480 instrucciones = 491.520

E/S mediante DMA

La CPU dedicará el tiempo necesario para poder programar el controlador de DMA. El DMA emplea robo de ciclo, con lo que el periférico ocupará los buses 500 ns. por byte. Con lo que para leer 1.024 bytes, tardará: 500 ns. x 1.024 bytes = 512.000 ns. = 512μsg. Dado que la rutina de inicialización del controlador del DMA consta de 8 instrucciones, podemos aproximar el cálculo a los 512μsg. que se corresponden con los robos de ciclo.

En ese tiempo, la CPU habrá ejecutado 5.120 instrucciones, que se corresponderán con el hecho de que durante el robo de ciclo por cada dato, la CPU no puede acceder a los buses.

La CPU podrá ejecutar 512.000 – 5.120 = 506.880 instrucciones correspondientes a otros procesos.

Resumiendo:

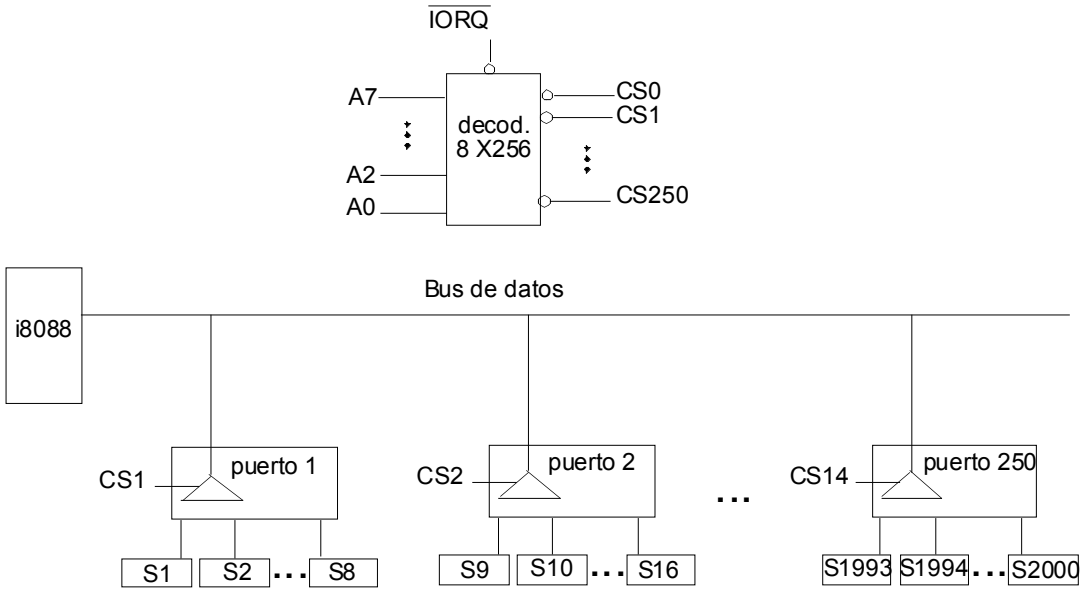
Tipo de E/S	Instrucciones totales	Instrucciones para otros procesos
E/S programada	512.000	0
E/S mediante interrupciones	20.480	491.520
E/S mediante DMA	5.120	506.880

Solución ejercicio 7

Dado que la palabra del i8086 es de 8 bits se podrán leer al mismo tiempo 8 sensores con los estados de 0 ó 3 voltios. Como queremos monitorizar 112 sensores, se necesitarán:

$$\frac{2.000\ sensores}{8\ bits} = 250\ puertos$$

Con lo que se agruparán en 250 puertos de entrada. Como tenemos 8 bits podremos direccionar 2^8 = 256 emplearemos las líneas A_0 a A_7 para direccionarlos. Además se empleará un decodificador de 8 x 256 para determinar el grupo de sensores que se explorarán.



Solución ejercicio 9

Los componentes sobre los que se tiene que comenzar el estudio son la CPU y el bus de memoria para ver cuál de los elementos se comporta como cuello de botella

CPU(máxima velocidad de E/S) será:

Velocidad de ejecución de instrucciones / Instrucciones por E/S =

$$\frac{300x10^6}{150.000} = 2000 \text{ operaciones de E/S por segundo}$$

Bus(máxima velocidad de E/S) será:

Ancho de banda del bus / Instrucciones por E/S = $\frac{100x10^6}{64x10^3} = 1562$ operaciones de E/S por segundo

Se ve que el bus es el cuello de botella por lo tanto se debe configurar el resto del sistema para que funcione al ritmo que marca el bus

Tiempo por operación de E/S de disco = tiempo de transferencia = 64Kb/ 5Mb/sg = 12,8 ms

Eso implica que cada disco hará 78,125 operaciones de E/S por segundo. Con lo que para poder llegar a tener 1562 operaciones e necesitarán 1562/78,125 =19,99 = 20 discos

Para saber el número de SCSI que se necesitan hace falta saber la velocidad media de transferencia por disco:

Velocidad de transferencia = tamaño transferencia /tiempo transferencia =

64Kb/12,8ms=4,88 Mb/sg con lo que como máximo se podrán conectar a cada SCSI 20Mb/sg / 4,88 Mb/sg = 4,09 = 4 discos por controladora.

De esta forma se necesitarían 20 / 4 = 5 buses y controladoras SCSI.

Solución ejercicio 11

El tiempo medio de lectura o de escritura, es decir, el tiempo medio de acceso al disco es la suma de:

- El tiempo de posicionado
- El retardo medio de rotación
- El tiempo de transferencia
- El tiempo introducido por la sobrecarga del controlador

En nuestro caso:

- El tiempo de posicionado: 12 ms.
- El retardo medio de rotación: 0,5 rotación / 3.600 r.p.m = 0,5/(3.600 r.p.m. / 60 sg./min) = 0,0083 sg. = 8,3 ms. =
- El tiempo de transferencia: 1 KB / 10 MB/sg. = 0,1 ms.
- El tiempo introducido por la sobrecarga del controlador: 2 ms.

Por tanto, el tiempo medio de acceso será de:

12 ms. + 8,3 ms. + 0,1 ms. + 2 ms. = 22,4 ms.

Solución ejercicio 13

Bus síncrono

Tiene un ciclo de bus de 50 ns.

El bus síncrono deberá:

- Enviar la dirección a la memoria: 50 ns.
- Leer la memoria: 100 ns.
- Enviar los datos al dispositivo: 50 ns.

El tiempo total será de 200 ns.

El ancho de banda máximo para transmitir 32 bits (4 bytes) cada 200 ns. será 4 bytes / 200 ns. = 4 MB / 0,2 sg. = 20 MB / sg.

Bus asíncrono

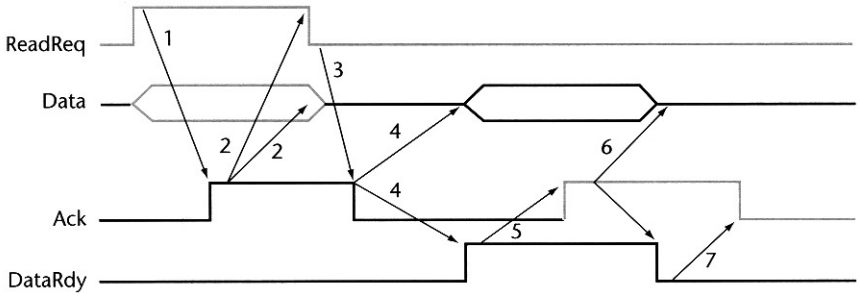


Figura 4. Protocolo de handshaking para un bus asíncrono

Tal y como vemos, en el protocolo de *handshaking* hacen falta 7 pasos, cada uno de ellos de 20 ns. pero sin embargo, algunos de los pasos pueden solaparse:

- Paso 1: la memoria detecta la activación de ReadReq, lee la dirección que hay en el bus y activa Ack. Tiempo 20 ns.
- Pasos 2, 3 y 4: el periférico libera los buses y desactiva ReadReq, la memoria activa Ack para indicar que va a leer y activa DataRdy para indicar que tiene el dato. Tiempo el máximo (3 x 20 ns., 100 ns.) = 100 ns.
- Pasos 5, 6 y 7: el periférico detecta DataRdy y lee los datos, activando al terminar Ack. La memoria ve la activación y libera el bus de datos y desactiva DataRdy. El periférico desactiva su señal de Ack indicando que la transferencia ha terminado. Tiempo 3 x 20 ns. = 60 ns.

El tiempo total es de: 180 ns.

Por lo tanto el ancho de banda máximo para transmitir 4 bytes cada 180 ns. será: 4 bytes / 180 ns. = 4 MB / 0,18 sg. = 22,22 MB / sg.

Solución ejercicio 15

Bus síncrono

Tiene un ciclo de bus de 30 ns.

El bus síncrono deberá:

- Enviar la dirección a la memoria: 30 ns.
- Leer la memoria: 60 ns.
- Enviar los datos al dispositivo: 30 ns.

El tiempo total será de 120 ns. El ancho de banda máximo para transmitir 32 bits (4 bytes) cada 120 ns. será de 4 bytes / 120 ns. = 33,3333 MB / sg.

Bus asíncrono

Tal y como vemos, en el protocolo de *handshaking* hacen falta 7 pasos, cada uno de ellos de 25 ns., pero los pasos 2, 3 y 4 pueden solaparse con el acceso a memoria

- Paso 1: la memoria detecta la activación de ***ReadReq***, lee la dirección que hay en el bus y activa ***Ack***. Tiempo 25 ns.
- Pasos 2, 3 y 4: el periférico libera los buses y desactiva ***ReadReq***, la memoria activa ***Ack*** para indicar que va a leer y activa ***DataRdy*** para indicar que tiene el dato. Tiempo el **máximo** (3 x 25 ns., 60 ns.) = 75 ns.
- Pasos 5, 6 y 7: el periférico detecta ***DataRdy*** y lee los datos, activando al terminar ***Ack***. La memoria ve la activación y libera el bus de datos y desactiva ***DataRdy***. El periférico desactiva su señal de ***Ack*** indicando que la transferencia ha terminado. Tiempo 3 x 25 ns. = 75 ns.

El tiempo total es de: 175 ns. Por lo tanto el ancho de banda máximo para transmitir 4 bytes cada 180 ns. será: 4 bytes / 175 ns. = 22,857 MB / sg.