Clase 1

Conceptos de arquitectura de computadoras

Concepto de **PROGRAMA** — SECUENCIA DE DATOS → FUNCIONES ARITMÉTICO/LÓGICAS → RESULTADOS

Antes se tenían sistemas cableados

- Programación en hardware: cuando cambiamos las tareas, debemos cambiar el hardware.

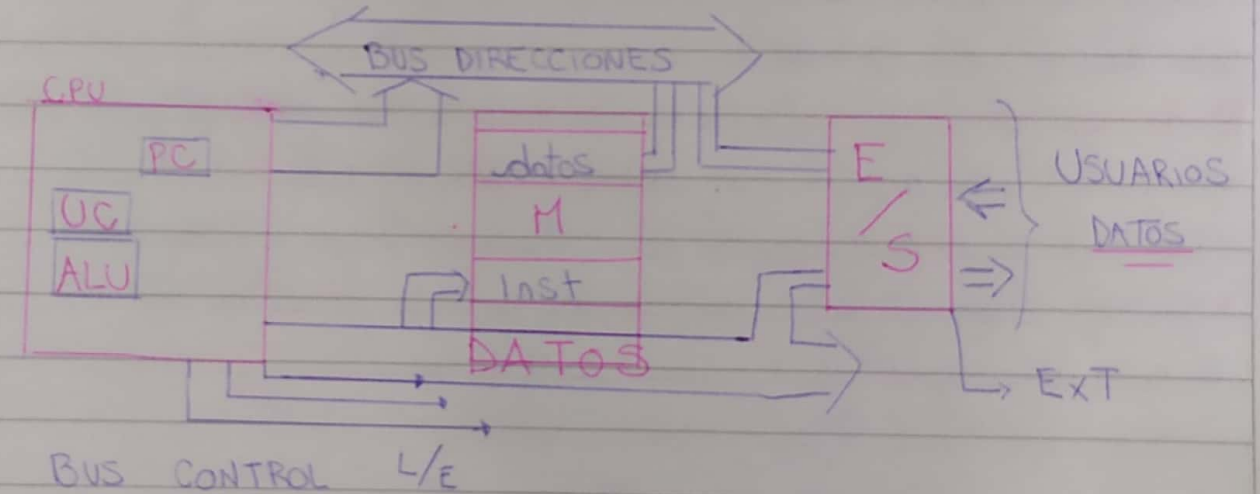
AHORA:

Código de instrucción
↓
Intérprete de instrucción
↓
Señales de control
↓
Datos → Funciones aritmético/lógicas → Resultados

- Programación en SOFTWARE: en cada paso se efectúa alguna operación sobre los datos.
- Para cada paso se necesita un nuevo conjunto de señales de control.
- Las instrucciones proporcionan esas señales de control.
- Aparece el nuevo concepto de programación. NO HAY QUE CAMBIAR EL HARDWARE

ARQUITECTURA VON NEUMANN

- La unidad central de procesamiento (CPU) está constituida por la Unidad de control (UC), encargada de dirigir las operaciones, y por la unidad aritmético-lógica (ALU), encargada de procesar los datos.
- E/S: los datos e instrucciones se introducen en el sistema mediante la Unidad de entrada. Los resultados se envían a través de la Unidad de Salida.
- La MEMORIA PRINCIPAL almacena temporalmente datos e instrucciones.



Su funcionamiento está dado por el CICLO DE INSTRUCCIÓN, el cual tiene 2 pasos:

• CICLO DE BÚSQUEDA (captación): común a todas las instrucciones.
La CPU (UC) capta (busca) una instrucción en memoria. El PC almacena la dirección de la próxima instrucción a buscar, por esto la CPU incrementa su valor. La instrucción buscada se carga dentro del registro IR de la CPU.

• CICLO DE EJECUCIÓN: depende de la instrucción, puede implicar varias operaciones. La CPU (UC) interpreta cada instrucción (que está en código binario) y lleva a cabo las acciones requeridas. La ejecución del programa se interrumpe solo si la máquina se apaga, hay un error o una instrucción que interrumpa a la computadora.

INSTRUCCIÓN elemento autocontenido que posee toda la información necesaria para llevar a cabo un ciclo de instrucción. Contiene:
- Código de operación: (C) qué hacer?
- Referencia a un operando } DIRECCIONES (C) dónde están?
- Referencia a otro operando }
- Referencia a la próxima instrucción. (C) cómo sigo?
- Referencia a resultado. (C) dónde reflejo el resultado?

MÁQUINA TEÓRICA DE 4 DIRECCIONES.
En una máquina de 3 direcciones no existe la referencia a la próxima instrucción, CPU usa registro PC de forma secuencial.
MÁQUINA DE 2 DIRECCIONES: CPU con registro PC y menos operaciones: MOV → COPIA la dirección del operando 1 es la misma dirección del resultado.
IAS: MÁQUINA DE 1 DIRECCIÓN: CPU con registro PC y registro acumulador y menos operaciones: LOAD, STORE
Solo dirección del operando 2.

ACCIONES POSIBLES:
- Procesador-memoria (transf de datos CPU - memoria)
- Procesador-E/S (transf de datos CPU y módulo de E/S)
- Procesamiento de datos (alguna op. aritmética o lógica con los datos)
- Control: alteración de la secuencia de ejecución (inst. de salto)
- COMBINACIÓN DE LAS ACCIONES ANTERIORES

* DECODIFICAC

REPERTORIO DE INSTRUCCIONES: conjunto completo de instrucciones que se realizan en una CPU (código máquina binario), representando simbólicamente por un conjunto de códigos de ensamblaje:

- De operaciones: ADD (sumar), SUB (restar), LOAD (cargar datos en un registro).
- De operandos: ADD BX, PEPE, sumar contenidos de reg BX y direc PEPE, guardar en BX

ALTO NIVEL A MÁQUINA:

Prog. en Alto Nivel → COMPILADOR → Prog en leng. Ensamblador → ENSAMBLADOR → objeto: Módulo en leng. máquina

objeto: librerías de rutinas en leng. máquina

→ Linkado ←

Ejecutable: Prog en lenguaje máquina → S.O. Loader → MEMORIA

¿Dónde se almacenan operandos?

- MEMORIA PRINCIPAL: o memoria virtual o memoria cache.
- REGISTRO DE LA CPU
- DISPOSITIVO DE E/S

Alternativas de almacenamiento

- Tipo PILA
- Tipo ACUMULADOR
- Tipo MEMORIA-MEMORIA
- Tipo REGISTRO-REGISTRO

Tipos de instrucciones:

- Procesamiento de datos (aritmético-lógicas)
- Almacenamiento de datos (de memoria)
- Transferencia de datos (de E/S).
- Control (de tests y flujo del programa)

¿Cuántas direcciones?

- + direcciones por instrucción: inst + complejas; + registros (+ rápida), - inst por programa.
- - direcciones por instrucción: inst. - compleja; + inst. por programa; la captación/ejecución de las inst. es + rápida.

DECISIONES EN EL DISEÑO DEL CONJUNTO DE INSTRUCCIONES

- Tipos de operandos (datos).
- Repertorio de operaciones (cuántas, cuáles, complejidad).

- Formatos de instrucciones (longitud, Nº de direc., tamaño de los campos).
- Registros (Nº de reg. de la CPU referenciables, en cuáles se pueden ejecutar qué operaciones)
- Modos de direccionamiento (¿cómo es especificada la ubi de un operando o una inst.?
- RISC contrapuesto a CISC.

TIPOS DE OPERANDOS: direcciones; números (punto fijo o flotante); caracteres (ASCII, EBCDIC); datos lógicos (bits 1 o 0).

Little endian: el byte menos significativo en la dirección con valor numérica más baja.

Big endian: el byte más significativo

SI SE PERMITEN LOS ACCESOS NO ALINEADOS EN MEMORIA SON MÁS LENTOS

TIPOS DE OPERACIONES: transferencias de datos; aritméticas; lógicas; conversión; entrada/salida; control del sistema; control de flujo.

1. Transferencia de datos: se debe especificar la ubicación del operando fuente y destino, el tamaño de los datos a ser transferidos y el modo de direccionamiento. Reg-Reg; Reg-Mem o Mem-Reg. MOV destino, Fuente.

2. Aritméticas: operaciones básicas (ADD, SUB, MUL, DIV). Números enteros sin/con signo. Números en punto flotante. Pueden incluirse otras operaciones: INC/DEC (uni); NEGate (Ca2); ABSolute; SHIFT LEFT/RIGHT (desplaza bits un lugar).

3. Lógicas - Conversión: operaciones que manipulan bits individualmente. Operaciones booleanas (AND, OR, XOR, NOT) otras (Rotate left/right) y operaciones para cambiar formatos de datos (conversión de binario a decimal o de EBCDIC a ASCII).

4. Entrada/Salida: pocas instrucciones para dar acciones específicas (IN/OUT). Se pueden realizar usando instrucciones de movimiento de datos (MOV). Se pueden realizar a través de un controlador aparte: DMA

5. Control de flujo: modifican el valor contenido en el registro PC: saltos incondicionales/condicionales) con retorno a llamada a subrutina.

MODOS DE DIRECCIONAMIENTO:

- Inmediato: el operando contiene la información sobre lo que hay que operar (MOV AX, 1000h)
- Directo de memoria o absoluto: la instrucción contiene la dirección de memoria exacta donde se encuentra el operando; el operando se encuentra en memoria (MOV BL, VAR_byte).
- Directo de registro: el operando está contenido en un registro (MOV AX, BX)
- Indirecto de memoria (en desuso).

- Indirecto con registro: la instrucción contiene una dirección que se emplea para leer en memoria una dirección intermedia que será la verdadera dirección del dato buscado. El operando se encuentra en memoria (MOV AX, [BX])
- Indirecto con desplazamiento: basado, indexado o relativo al PC; PILA (o relativo al SP).

SUBRUTINAS: innovación en lenguajes de programación.

- Programa auto-contenido
- Puede invocarse desde cualquier punto de un programa con la instrucción CALL
- Brinda economía (código usado varias veces) y modularidad (subdivisión en unidades pequeñas)
- Requiere pasaje de argumentos (parámetros) por valor o referencia
  - Vía registros: el Nº de reg. es la principal limitación; es importante documentar qué registros se usan.
  - Vía memoria: se usa un área definida de memoria (RAM); difícil de estandarizar.
  - Vía Pila: es el método más ampliamente usado, es independiente de memoria y registros, es usada por el usuario y por el sistema.

Funcionamiento de una Pila: el operando está de forma implícita en la cabeza de la Pila; se requiere un registro Puntero de Pila (SP) apunta al último lugar usado. OPERACIONES: PUSH apilar POP desapilar. Mueve los datos (reg-mem / mem-reg) y modifica el puntero SP.

PILA: estructura de almacenamiento dinámica/ por demanda, caracterizada por tener un puntero (STACK POINTER) indicador del último elemento. Se desarrolla en la memoria.

INTERRUPCIONES mecanismo mediante el cual se puede cambiar el procesamiento normal de la CPU. Pueden ser de origen interno o externo.

¿Por qué interrumpir?

- Resultado de una ejecución de una instrucción (E) overFlow)
- Temporizador interno del procesador (permite al S.O. realizar ciertas funciones de manera regular).
- Operación de E/S (E): indicar finalización normal de una operación).
- Fallo de hardware (E): pérdida de energía, error de paridad en la memoria)

En caso de una interrupción se transfiere el control al GESTOR, antes de esto hay que salvar el "estado del procesador".

GESTOR: rutina, secuencia de instrucciones que responden a las causas que ocasionó la solicitud de interrupción. Guardar/restaurar su valor/modificar valores en registros. Retorno a la ejecución normal del programa interrumpido.

Las interrupciones pueden ser:

- NO enmascarables: No pueden ignorarse; indican eventos peligrosos o de alta prioridad, requieren respuesta eficiente y rápida.
- Enmascarables: pueden ser ignorados; sus eventos no configuran peligro o pueden esperar, la posible solicitud de interrupción puede inhibirse con instrucciones especiales.

INTERRUPCIONES POR HARDWARE (interrupt request)

- Generados por dispositivos de E/S.
- Son los "verdaderos" interrupciones.
- El sistema de cómputo tiene que manejar estos eventos externos "no planeados" o "asincrónicos".
- No están relacionados con el proceso en ejecución en ese momento.

Traps/excepciones: interrupciones por hardware creadas por el procesador mediante en respuesta a ciertos eventos internos:

- Condiciones excepcionales: overFlow en ALU de punto flotante.
- Falla de Programa: Tratar de ejecutar una instrucción no definida.
- Fallos de hardware: error de paridad de memoria.
- Accesos no alineados o a zonas de memoria protegidas.

INTERRUPCIONES POR SOFTWARE: instrucciones explícitas que afectan al procesador de la misma manera que las interrupciones por hardware.

- Permiten depurar los gestores de interrupción
- Pueden ser usados para invocar funciones del S.O. (permite que los subrutinas del S.O. se carguen en "algún" lugar y puedan utilizarse).
- No requieren conocer la dirección de la rutina en tiempo de ejecución.
- Sin ellas deberíamos escribir todas las funciones que necesitemos o al cargar un programa habría que "mirar" todas las llamadas a funciones del BIOS y S.O. y reemplazar en el código las direcciones de todas estas funciones invocadas.

CICLO DE INTERRUPCIÓN

1. Se comprueba si se ha solicitado alguna interrupción (indicada por una señal, flag, de pedido de interrupción).
2. Si no hay señal se coge la siguiente instrucción
3. Si hay algún pedido de interrupción pendiente:
   - 3i. Se suspende la ejecución del programa en curso.
   - 3ii. Guarda su contexto (básicamente instrucción a ejecutar y el estado del procesador).
   - 3iii. Carga el PC con la dirección de comienzo del gestor. Se inhiben otros pedidos de interrupción
   - 3iv. Finalizada la rutina de gestión, el procesador retoma la ejecución del programa del usuario en el punto de interrupción.

INTERRUPCIONES MÚLTIPLES

Con interrupciones inhabilitadas:

- El procesador puede y debe ignorar la señal de petición de interrupción si se produce una interrupción en ese momento.
- Si se hubiera generado una interrupción se mantiene pendiente y se examinará luego una vez que se hayan habilitado nuevamente.

Con interrupciones habilitadas:

- Ante una solicitud, se inhabilitan; se gestiona la misma y luego se habilitan otra vez.

Por lo tanto las interrupciones se manejan en un orden secuencial estricto:

Con prioridades:

- Una interrupción de prioridad más alta puede interrumpir a un gestor de interrupción de prioridad menor.
- Cuando se ha gestionado la interrupción de prioridad más alta, el procesador vuelve a las interrupciones previas (de menor prioridad).
- Terminadas todas las rutinas de gestión de interrupciones se retoma el programa del usuario.

Las interrupciones se manejan anidando gestores.

RECONOCIMIENTO DE INTERRUPCIONES

Interrupciones multinivel: Cada dispositivo que puede provocar interrupción tiene una entrada física de interrupción conectada a la CPU. Es sencillo pero muy cara

· Línea de interrupción única: una sola entrada física de pedido de interrupción a la que están conectados todos los dispositivos. Se debe "preguntar" a cada dispositivo si ha producido el pedido de interrupción. (técnica POLLING / encuesta).

· Interrupciones vectorizadas: el dispositivo que quiere interrumpir además de la señal de pedido de interrupción, debe colocar en el bus de datos un identificador (vector); lo coloca el periférico directamente o controlador de interrupciones (que se ocupa de todo).

Si el procesador tiene una única entrada de pedido de interrupciones y si tenemos varios productores de interrupciones lo solucionamos con el PIC, dispositivo controlador programable de interrupciones. Es un secretario (no procesa) La CPU le programa su funcionamiento

Tabla de vectores de interrupción: es el nexo el tipo de interrupción (0..255) y el procedimiento designado para atenderla. Cada entrada es una doble palabra (4 bytes): dirección del procedimiento que brinda el servicio (Ej: aaaa:yyyy donde yyyy es la dirección lógica / física). Tiene vectores preasignados:

- Tipo 0: Finaliza ejecución de programa.
- Tipo 3: punto de parada para depuración / seguimiento.
- Tipo 6: lectura de entrada std. Requiere el uso de BX
- Tipo 7: escritura de salida std. Requiere BX y AL

Registros Internos del PIC

- EOI: para comandos (para Fin de Int escribir 20H)
- IMR: máscara de int. (enmascara con 1)
- IRR: registro 8 bits enmascara con 1 la posición del pedido.
- ISR: int en servicio (indica con 1 la posición que se está atendiendo).
- INT0...INT7: c/u con su vector.

Se sitúan a partir de la dirección 20H. Son accedidos con operaciones lectura y escritura en el espacio de E/S (IN y OUT).

Interrupciones hardware asignadas:

- INT 0 → tecla F10
- INT 1 → timer
- INT 2 → handshake
- INT 3 → DMA
- INT 4 a INT 7 no usadas

MÓDULO DE ENTRADA/SALIDA

PROBLEMAS: • Gran variedad de periféricos con varios métodos de operación transmisión de diferentes cantidades de datos; a diferentes velocidades; usan diferentes formatos de dato y tamaños de palabras.
• Son todos más lentos que la CPU y la RAM.
• Necesidad de módulos de E/S con alguna "inteligencia".

FUNCIÓN: realizar la interfaz entre el procesador y la memoria (bus) y los periféricos. Pueden manejar 1 o + periféricos (e/s básicos como mouse, monitor, teclado; de almacenamiento; de impresión; de comunicación con dispositivos remotos; multimedia; de automatización y control)

COMPONENTES:
• Registro para DATOS: LEÍDO y ESCRITO por la CPU y quizás por los periféricos.
• Registro para CONTROL: la CPU ESCRIBE.
• Registro STATUS: puede ser LEÍDO por la CPU para conocer el estado del periférico.
• PUERTOS USB: interfase e/ periférico y módulo de E/S. Envía señales:
- Señal de CONTROL: función a realizar (Ej: INPUT o READ, OUTPUT o WRITE).
- Señal de ESTADO: READY / NOT READY
- Control lógico: manejo de direccionamiento.
- Transductor: conversión de datos
- BUFFER: adaptación (1, 8 o 16 bits).

FUNCIONES: • Control y temporización de 1 o + dispositivos externos.
• Interpretar las órdenes que recibe de CPU y transmitirlas al periférico.
• Comunicación con la CPU (registros) y Memoria.
• Controlar las transferencias de datos e/ CPU y el periférico (convertir formatos, adaptar velocidades).
• Comunicación con los dispositivos (periféricos)
• Informar a la CPU del estado del periférico.
• Almacenamiento temporal (buffering) de datos.
• Detección de errores.
• Bloque de LÓGICA: encargado de decir qué se conecta con qué. Es administrado por el bus de direcciones.

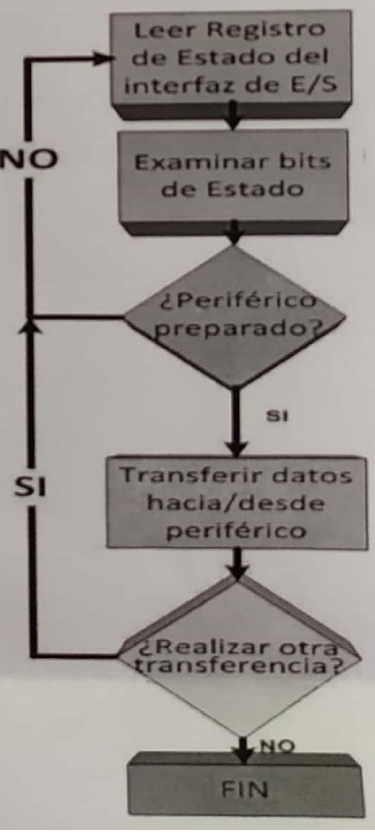
CAPACIDADES: • Ocuparse de 1 o varios dispositivos
• Ocultar las propiedades del dispositivo a la CPU (temporizados, formatos, etc.)
• Controlar ono las funciones del dispositivo

OPERACIÓN:

1. DIRECCIONAMIENTO: cómo la CPU identifica el destino
   a. Mapeada en memoria: dispositivos de E/S y memoria comparten un único espacio de direcciones. E/S se trata a la memoria de lectura/escritura. No hay órdenes específicas para E/S. Variedad de órdenes de acceso a memoria (programación eficiente).
   b. E/S aislada: espacios de direcciones separados. Necesidad de líneas especiales de E/S y de memoria. Órdenes específicas para E/S. IN y OUT.
2. TRANSFERENCIA DE DATOS: envío de la orden (leer o escribir).
3. GESTIÓN DE E/S:
   a. Programada con espera de respuesta • Intercambio de datos e/ CPU y módulo.
   • La CPU tiene control directo sobre la operación de E/S.
     - Comprobación del estado del dispositivo
     - Envío de comandos de lectura/escritura
     - Transferencia de datos
   • La CPU espera que el módulo E/S termine la operación
   • La CPU permanece ociosa durante un periodo de tiempo (sigue leyendo el reg → llegó la respuesta; sigue haciendo cosas pero nada productivo: NO DESEABLE).

Detalles de la E/S programada

- La CPU solicita la operación de E/S al módulo.
- El módulo E/S realiza la operación.
- El módulo E/S activa los bits de estado del dispositivo direccionado y espera.
- La CPU comprueba periódicamente el estado de esos bits, hasta que detecta que la operación fue completada.
- En caso contrario la CPU espera y vuelve a comprobarlo más tarde.



mecanismo igualador de los tiempos de RTA

b. E/S por interrupciones: • La CPU no tiene que esperar la finalización de la tarea de E/S, puede seguir procesando
• No se repite la comprobación de los estados de los módulos.
• El módulo envía un pedido de interrupción a la CPU cuando está listo nuevamente.

1. CPU: envía una orden de lectura (READ).
MÓDULO E/S: obtiene los datos del periférico mientras la CPU realiza otro trabajo.
2. CPU: Chequea si hay pedidos de interrupciones pendientes al final de cada ciclo de instrucción.
MÓDULO E/S: emite un pedido de interrupción a la CPU.
3 CPU: detecta el pedido, guarda el contexto, interrumpe el proceso y realiza la gestión de la interrupción.
4. CPU: solicita los datos.
MÓDULO E/S: transfiere los datos.

Identificación del módulo que interrumpe
- DIFERENTES LÍNEAS PARA CADA MÓDULO (computadoras personales; limita el número de dispositivos).
- CONSULTA SOFTWARE (POLL O ENCUESTA) ocurrido un pedido de interrupción la CPU consulta a cada módulo para determinar quién fue el demandante. Resulta lenta.
- CONEXIÓN EN CADENA (daisy chain) "hard Poll": la línea de reconocimiento de interrupción se conecta encadenando los módulos, la línea de pedido es compartida. Una vez enviada la confirmación de parte de la CPU el módulo responderá colocando un vector (palabra) en el bus q lo identificará. La CPU emplea el vector como puntero para acceder a la rutina de servicio. Todas las líneas de interrupción tienen un orden de prioridad.

Se debe usar un árbitro o gestor de interrupciones externo PIC que tiene 8 líneas de interrupción, podrá manejar 8 módulos de E/S. Usando conexión en cascada hasta 64. Puede ser útil tener una Interfase de Periféricos Programable PIO, es un módulo de E/S de propósito general, posee 24 líneas de E/S programable vía los registros de control.

C. E/S con acceso directo a memoria (DMA): se utiliza un CONTROLADOR DE DMA (dispositivo capaz de controlar una transferencia de datos e/ un periférico y memoria sin intervención de la CPU). Debe actuar como maestro del bus durante la transferencia DMA y ser capaz de:
- Solicitar el uso del bus mediante las señales y las lógicas de arbitraje necesarios
- Especificar la dirección de memoria sobre la que se realiza la transferencia
- Generar las señales de control del bus (sincronización de la transferencia y tipo de operación lectura/escritura).

ETAPAS: INICIALIZACIÓN: la CPU debe enviar al interfaz del periférico y al DMAC los parámetros de la transferencia.

INICIALIZACIÓN DEL INTERFAZ (Bus master: CPU - Bus slave: Interfaz)
- Nº bytes a transferir. • Tipo de operación (lectura/escritura)
- Otra información de control (Pista, sector, etc).

INICIALIZACIÓN CONTROLADOR DMA (Bus master: CPU - Bus slave: DMAC)
- Nº bytes o palabras a transferir. • Tipo de transferencia (lectura / escritura)
- Dirección de memoria inicial para la transferencia
- Nº de canal (para DMA= con varios canales).

Después de la inicialización la CPU retorna a sus tareas y no se preocupa más de la ejecución de la transferencia

REALIZACIÓN DE LA TRANSFERENCIA: cuando el periférico está listo se lo indica al DMAC, quien pide el control del bus y se realiza la transferencia e/ periférico y memoria.

Bus master: DMAC + memoria - Bus slave: Memoria

Después de la transferencia de cada palabra se actualizan los registros del DMAC (Nº bytes o palabras a transferir y dirección de memoria).

FINALIZACIÓN DE LA TRANSFERENCIA: el DMAC libera el bus y devuelve el control a la CPU, suele activar una señal de interrupción para indicar a la CPU la finalización de la operación de E/S solicitada.

PROBLEMAS • Se puede degradar el rendimiento de la CPU si el DMAC hace uso intensivo del bus. La CPU no puede acceder a memoria para leer inst./datos.
↳ Solución: uso de memoria caché (la CPU deja de necesitar el bus porque las inst. de la caché; el DMAC aprovecha estos intervalos que la CPU no usa el bus para realizar las transferencias.

En caso de computadoras sin caché el procesador no utiliza el bus en todas las fases de la ejecución de una instrucción, el DMAC puede aprovechar las fases de ejecución de una instrucción en las que la CPU no utiliza el bus para realizar sus transferencias.

Si el DMAC sólo toma el control del bus durante los intervalos de tiempo en los que la CPU no hace uso del mismo el rendimiento del sistema no sufrirá degradación alguna.

2. TIPOS DE TRANSFERENCIA:

DMA MODO RÁFAGA: el DMAC solicita el control del bus a la CPU. Cuando la CPU concede el bus, el DMAC no lo libera hasta haber finalizado la transferencia de todo el bloque de datos completo.

Ventaja: la transferencia se realiza de forma rápida.

Desventaja: durante el tiempo que dura la transferencia la CPU no puede utilizar el bus con memoria, lo que puede degradar el rendimiento del sistema

DNA MODO ROBO DEL CICLO: el DMAC solicita el control del bus a la CPU. Cuando la CPU concede el bus al DMAC, se realiza la transferencia de una única palabra y después el DMAC libera el bus. El DMAC solicita el control del bus tantas veces como sea necesario hasta finalizar la transferencia del bloque completo.

Ventaja: no se degrada el rendimiento del sistema.

Desventaja: la transferencia tarda más tiempo en realizarse.

Para la CPU no es una interrupción (el procesador no debe guardar el contexto). Si bien el trabajo de la CPU es lento, no será tanto como si ella realizara la transferencia. Por lo tanto, para transferencia de E/S de múltiples palabras, es la técnica más eficiente.