Entrada/Salida

OC1 - 1er Cuatrimestre 2018

I/O: Conexión al mundo exterior

Vimos

- Cómputo con los registros
- Cargar datos en los registros desde la memoria
- Guardar datos en la memoria desde los registros

¿De dónde provienen los datos que hay en la memoria?

¿Cómo exponemos la información afuera de la computadora para que la entendamos?

Por qué estudiar I/O

- Ley de Moore: se **duplica el número de transistores** en un circuito integrado aproximadamente **cada dos años**

  Moore, Gordon E. *Cramming more components onto integrated circuits*, Proceedings of the IEEE 86.1 (1998): 82-85.

- Ley de Amdahl: *speed-up* está limitado por el elemento más lento.

  Amdahl, Gene M. *Validity of the single processor approach to achieving large scale computing capabilities*, Proceedings of the April 18-20, 1967, spring joint computer conference. ACM, 1967.

- Adivinen cuál es el elemento más lento...

Dispositivos de I/O

Por comportamiento

- input: teclado, detector de movimiento, placa de red
- output: monitor, impresora, placa de red
- almacenamiento: disco rígido, memoria flash, CD-ROM

Por velocidades

- teclado: 100 bytes/s
- memoria USB: 30 MB/s
- disco rígido: 6 Gb/s
- red: 1 Mb/s - 1 Gb/s

Comunicación con los dispositivos

Comunicación a través de registros

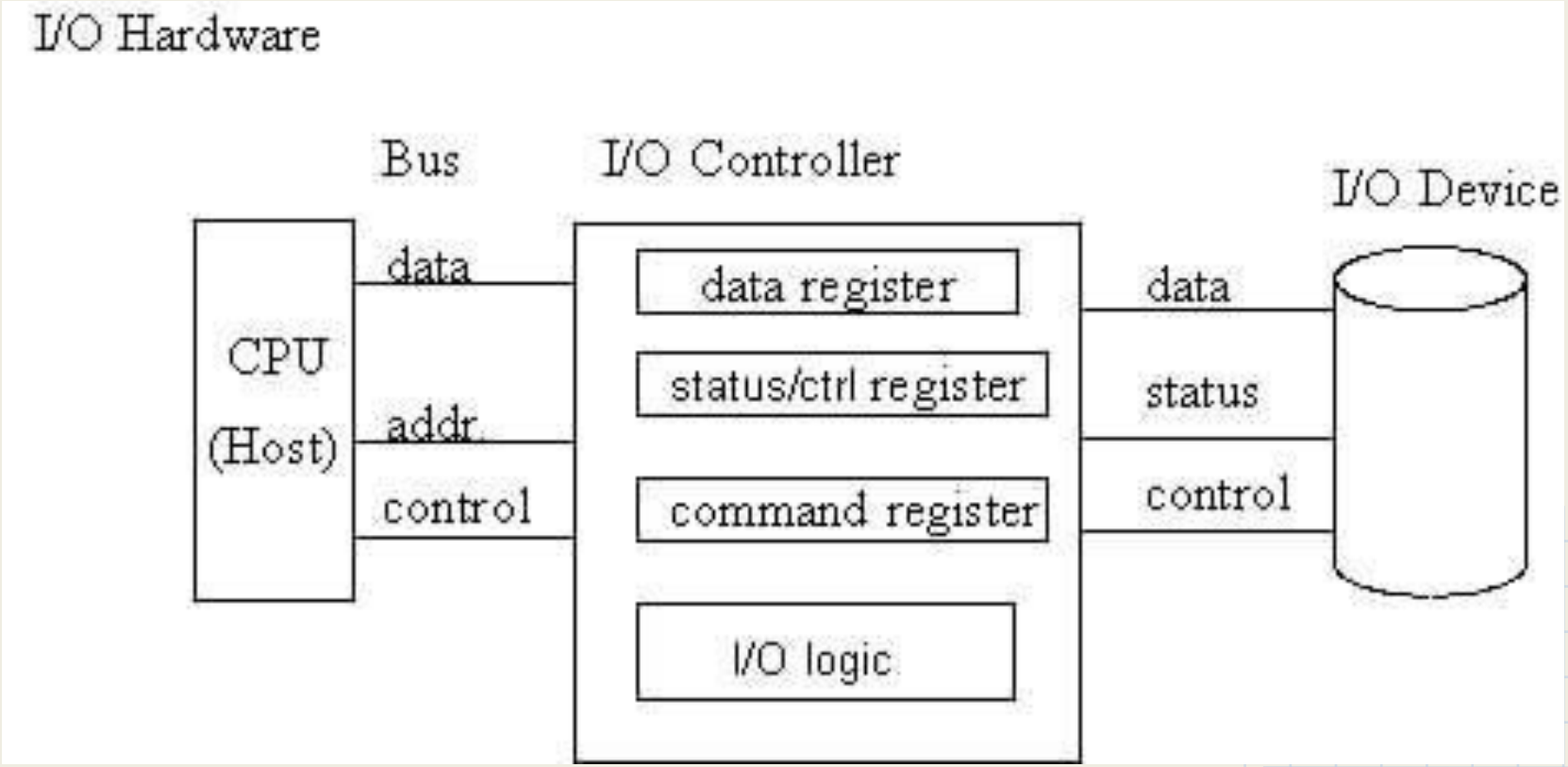
Registros de Control/Estado

- CPU dice qué hacer -- escribe el registro de control
- CPU verifica la tarea -- lee el registro de estado

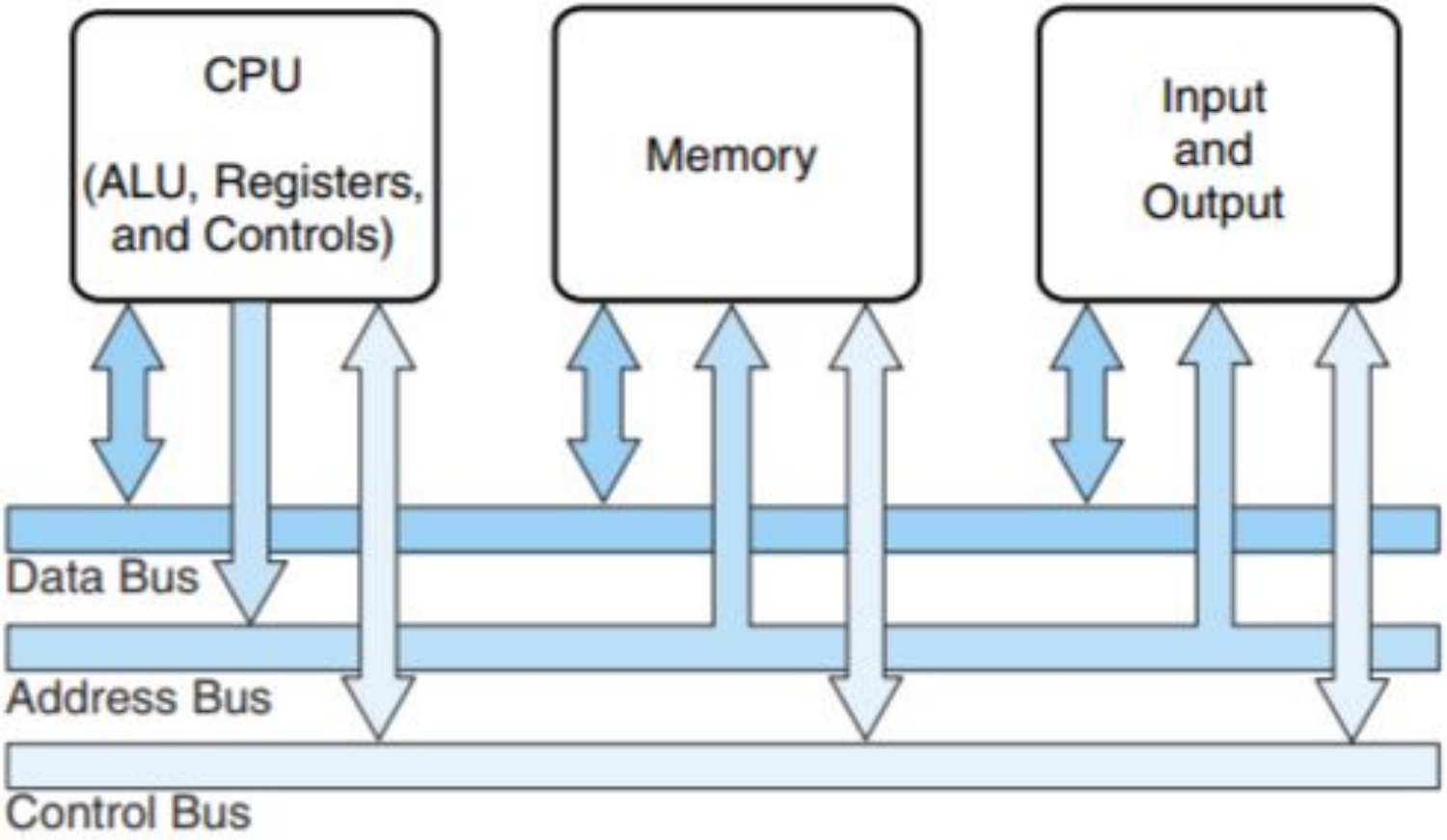
Registros de Datos

- CPU transfiere datos desde/hacia el dispositivo
- La electrónica resuelve la operación
- Ejemplos de transferencias: pixels a la pantalla, bits desde/hacia el disco, caracteres desde el teclado, etc.

Registros de un dispositivo



Buses de E/S


CPU
(ALU, Registers, and Controls)
Memory
Input and Output
Data Bus
Address Bus
Control Bus

Programming interface

¿Cómo identifico los registros de cada dispositivo?

- **Mapeado a memoria** vs. **instrucciones especiales**

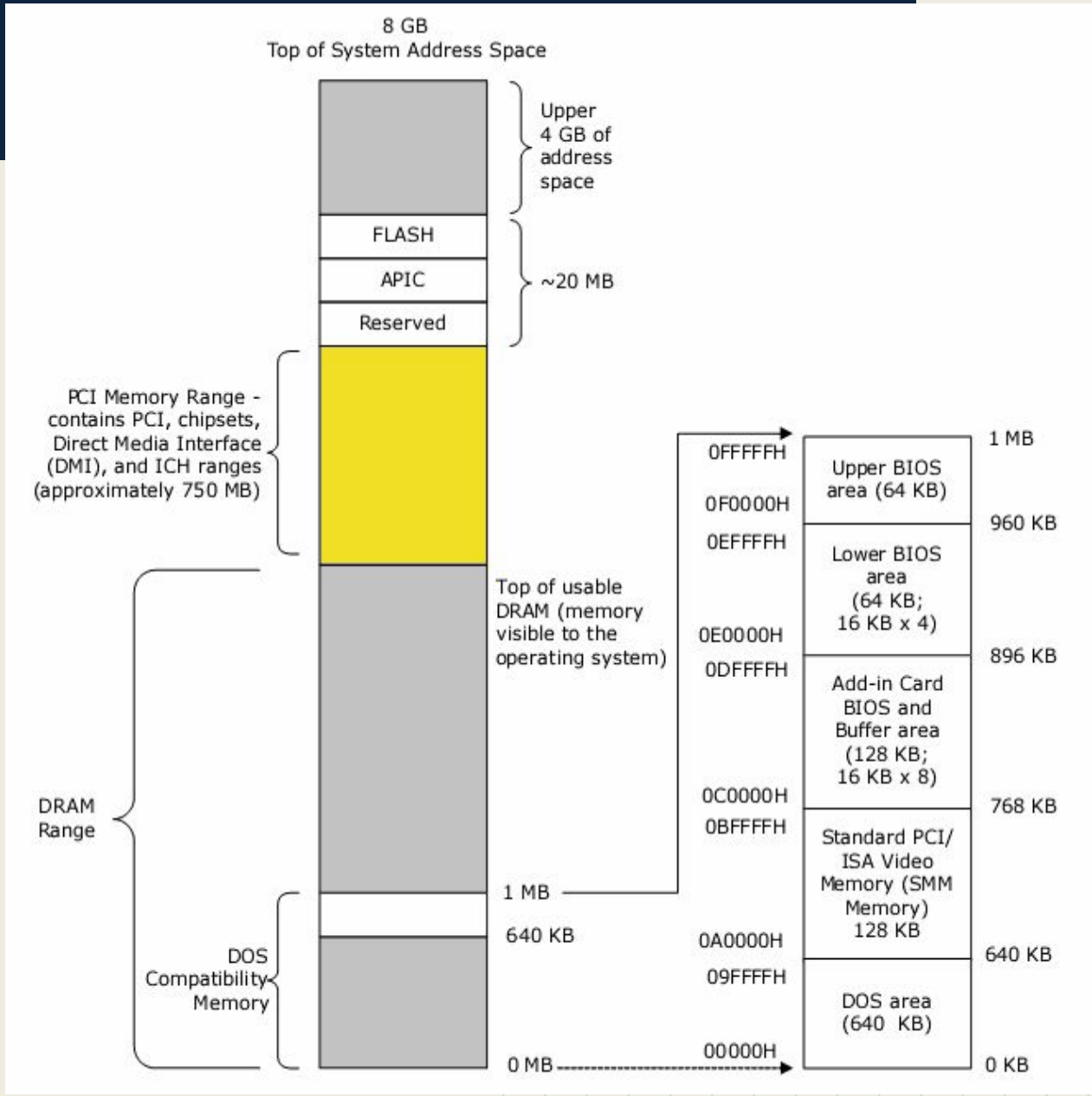
¿Cómo manejo los tiempos de las transferencias?

- **Sincrónico** vs **Asincrónico**

¿Quién controla la transferencia?

- CPU (**polling**) vs dispositivo (**interrupciones**)

Mapeado a memoria (INTEL)



Instrucciones especiales (INTEL)

Instrucciones especiales

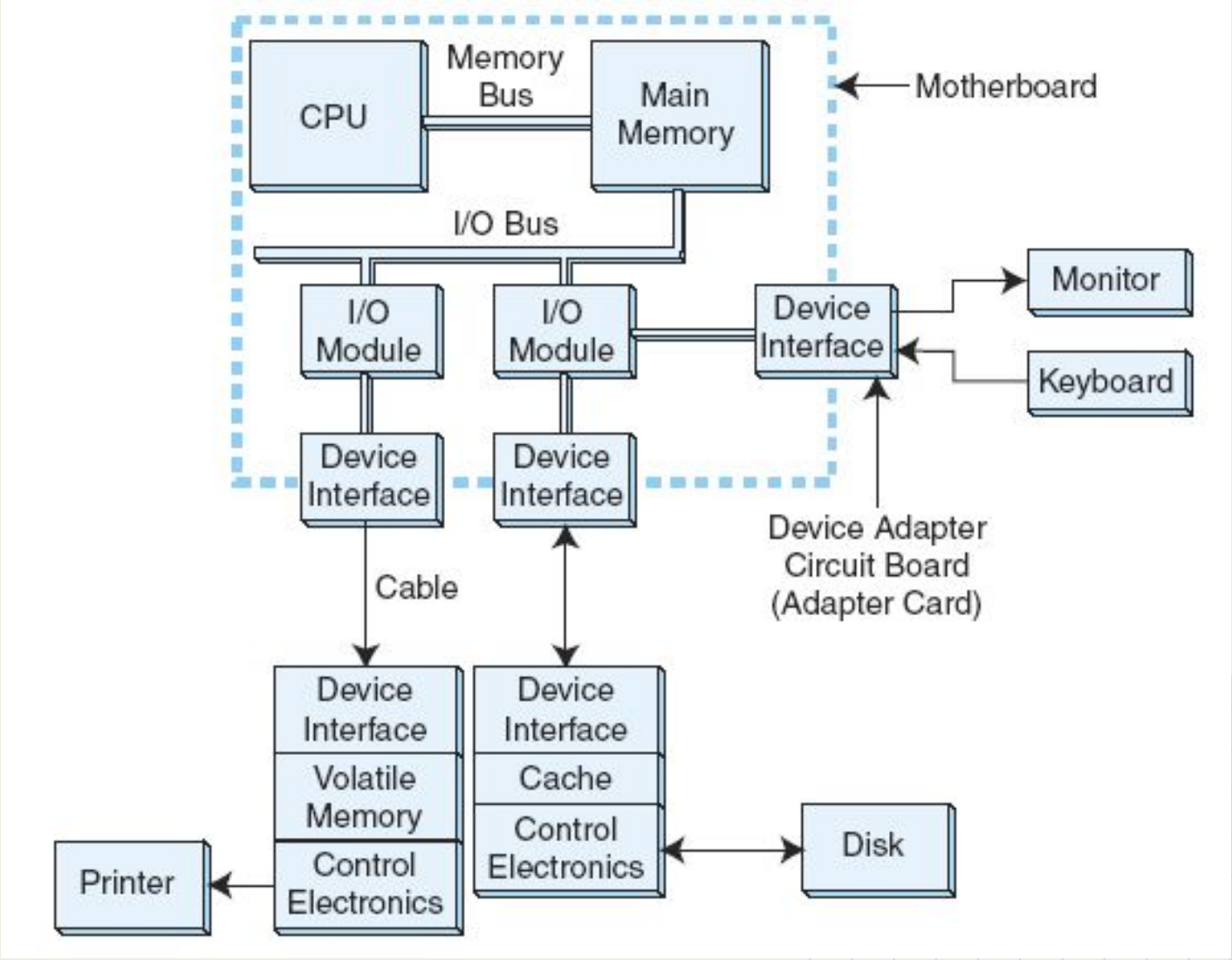
- IN
- OUT

```
IN   AL,  19H     ;8-bits  are  saved  to  AL  from  I/O  port  19H.
IN   EAX, DX      ;32-bits  are  saved  to  EAX.
OUT  DX,  EAX     ;32-bits  are  written  to  port  DX  from  EAX.
OUT  19H, AX      ;16-bits  are  written  to  I/O  port  0019H.
```

Mapeado vs Instrucciones especiales

- Instructions especiales: opcodes para I/O, operaciones *encubiertas* en la instrucción.
- Mapeado a memoria: direcciones asignadas (saca lugar a la memoria), usa instrucciones de movimiento de datos para TODAS las operaciones (I/O y control)

Modelo de I/O: Mapeado a memoria



Polling vs Interrupciones

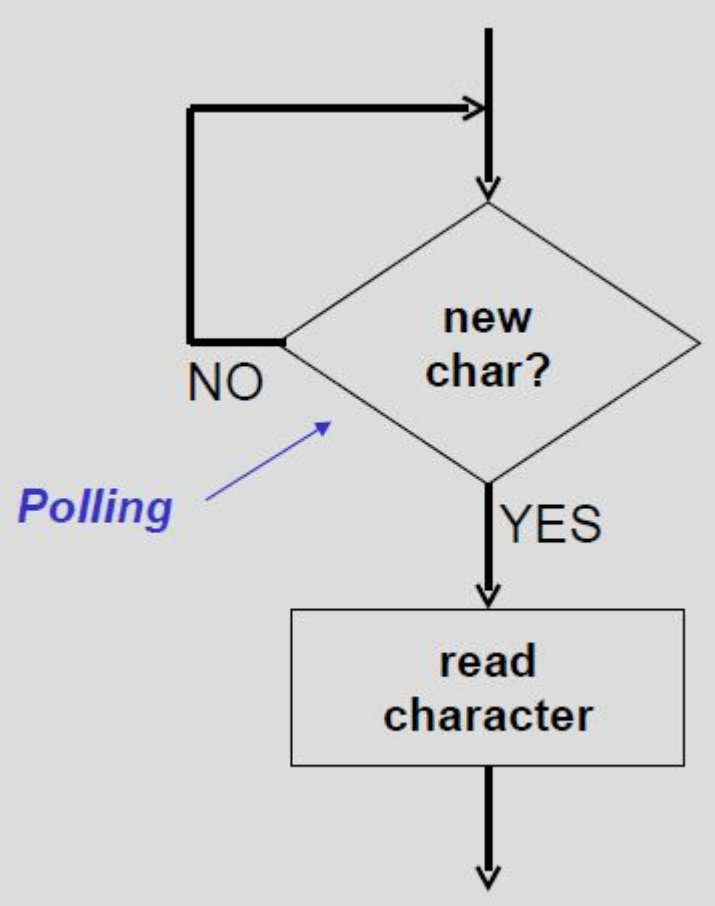
Polling

- I/O *controlado por* la CPU
- el dispositivo de I/O es consultado periódicamente por la CPU.

Interrupciones

- I/O *controlado por* el dispositivo
- el dispositivo I/O interrumpe la CPU cuando quiere comunicarse.

Ejemplo de polling: teclado



Supongamos que:

- el registro de estado está en 0x10
- el registro de datos está en 0x11

```
# New char? ... polling
poll:   IN R0, 0x0010
        CMP R0,0x0001
        JNE poll

# Yes, new char. Read it!
ready:  IN R1, 0x0011
        ...
```

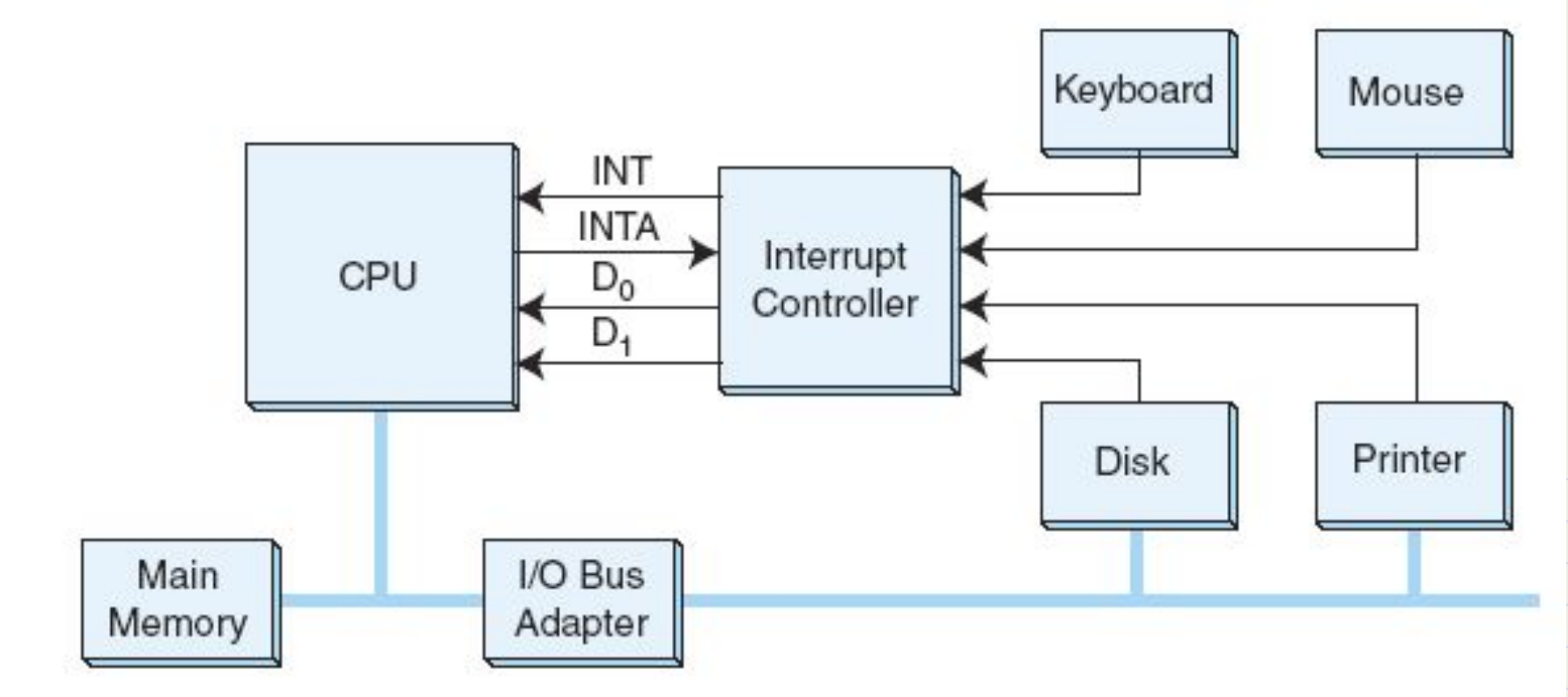
Interrupciones

Polling: la interacción entre CPU y dispositivo es controlada por la CPU

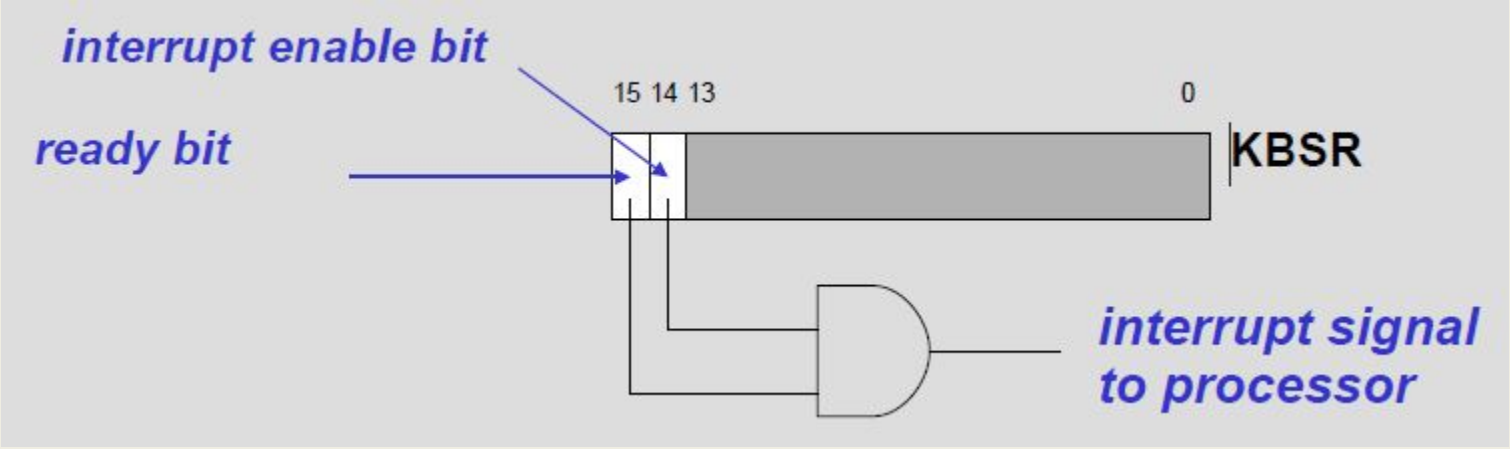
Interrupciones: la interacción entre CPU y dispositivo es controlada por el dispositivo. Al terminar, necesita:

1. Forzar a suspender el programa en ejecución
2. Indicar al procesador que resuelva las necesidades del dispositivo
3. Retomar ejecución del programa suspendido

Modelo de I/O con interrupciones



Implementando INTR



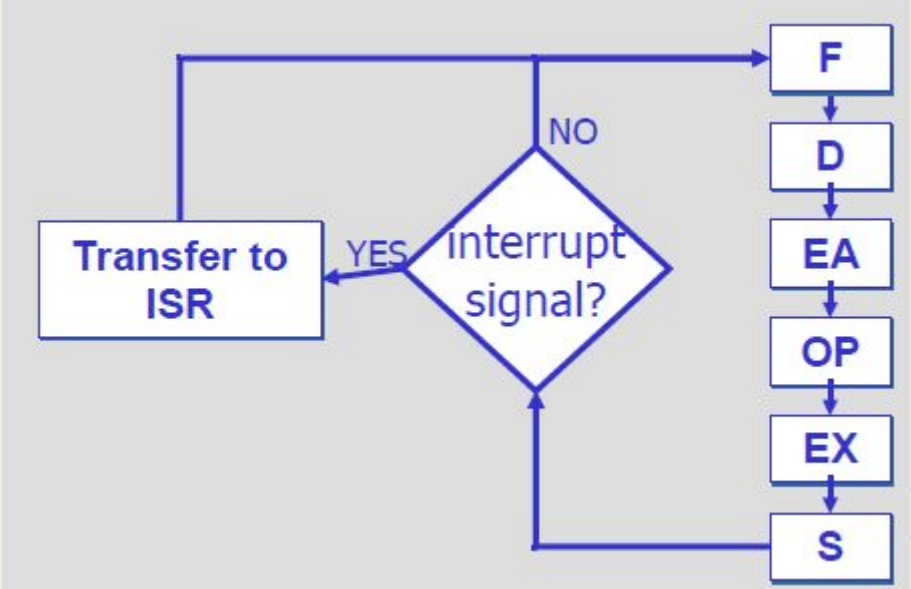
Implementación de interrupciones

Necesitamos

– Que un dispositivo pueda indicar un evento a la CPU (INTR)

– Que la CPU verifique si alguien indicó una interrupción

- CPU verifica la señal entre el STORE y el FETCH
- **Nuevo Flag**: IF (hab./deshab. interrupciones)



Rutina de Atención

Pasos **HARDWARE**:

1. Dispositivo I/O activa señal de INTR
2. CPU termina de ejecutar instrucción y verifica INTR (si IF está en 1)
3. CPU acusa recibo (INTA) e identifica el dispositivo
4. Dispositivo envía identificación
5. CPU guarda contexto en el stack (PSW y PC)
6. CPU deshabilita interrupciones (IF = 0)
7. CPU salta a rutina de atención

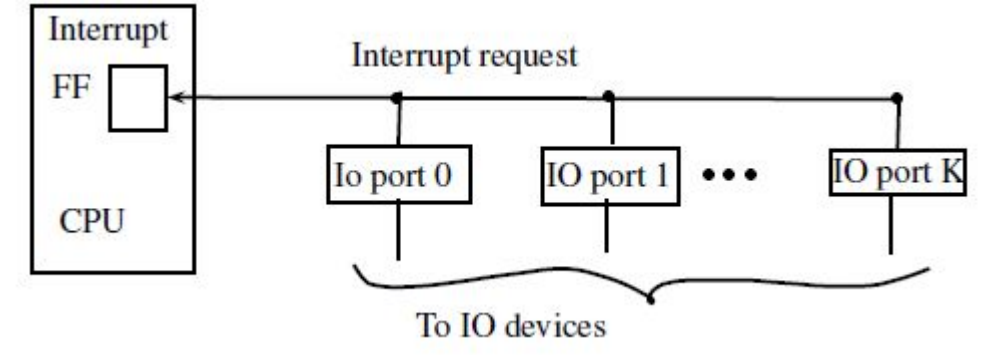
Rutina de Atención

Pasos **SOFTWARE**:

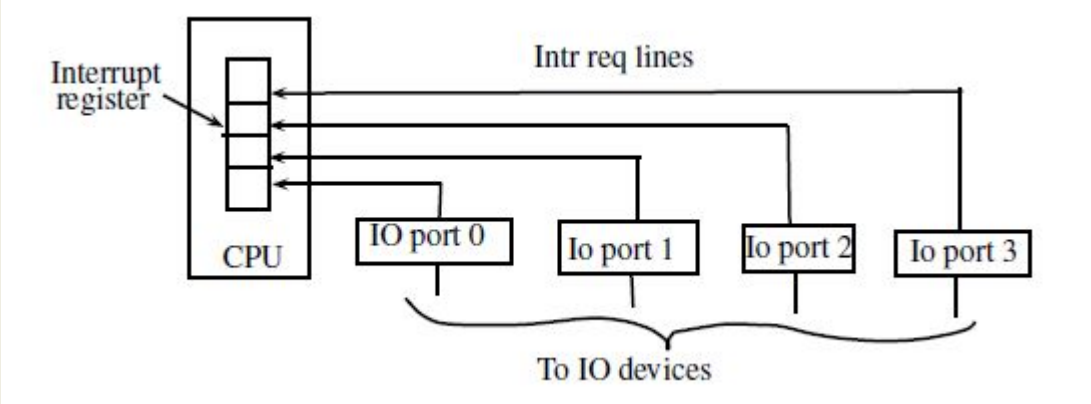
1. ISR (Interrupt Service Routine) guarda máscara de interrupciones
2. Se modifica la máscara para permitir interrupciones de alta prioridad
3. Se habilitan las interrupciones (IF=1)
4. **SE EJECUTA ISR**
5. IF = 0
6. Se recupera máscara de interrupciones
7. Retorno de la ISR con IRET

Múltiples interrupciones

Una única señal de interrupción

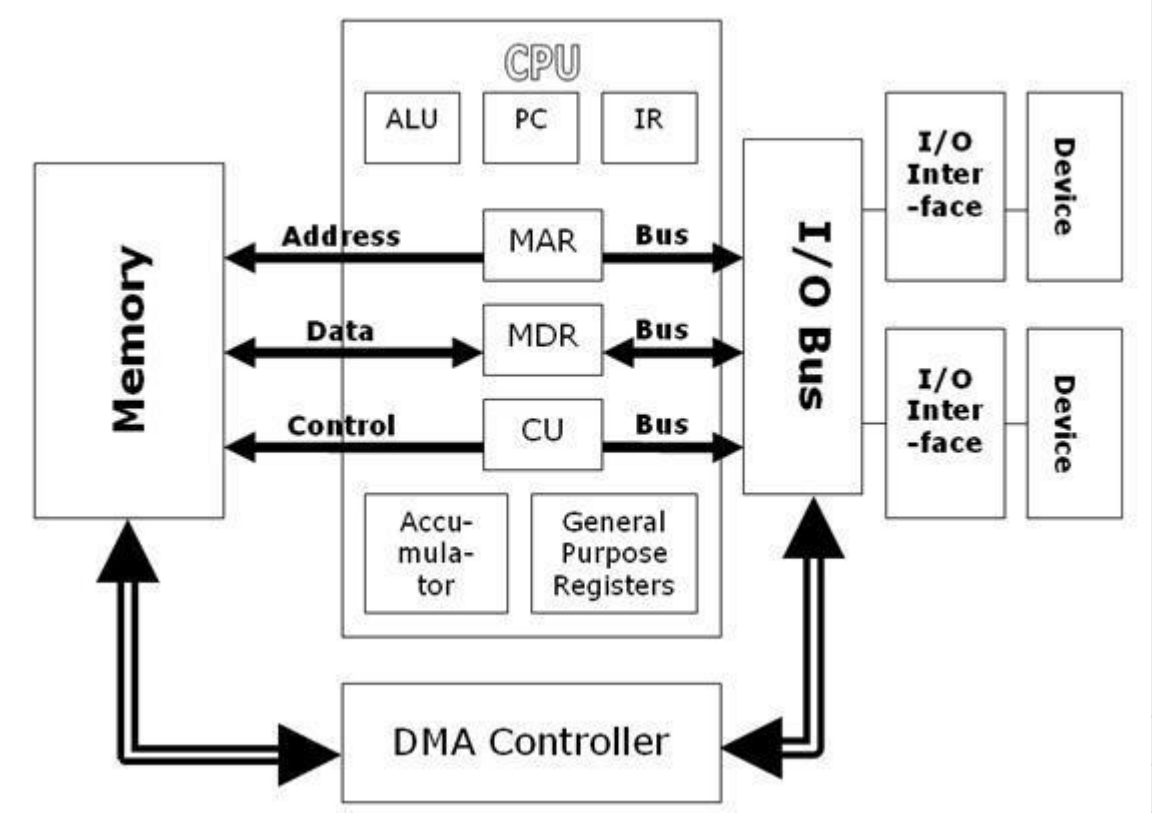


Múltiples señales

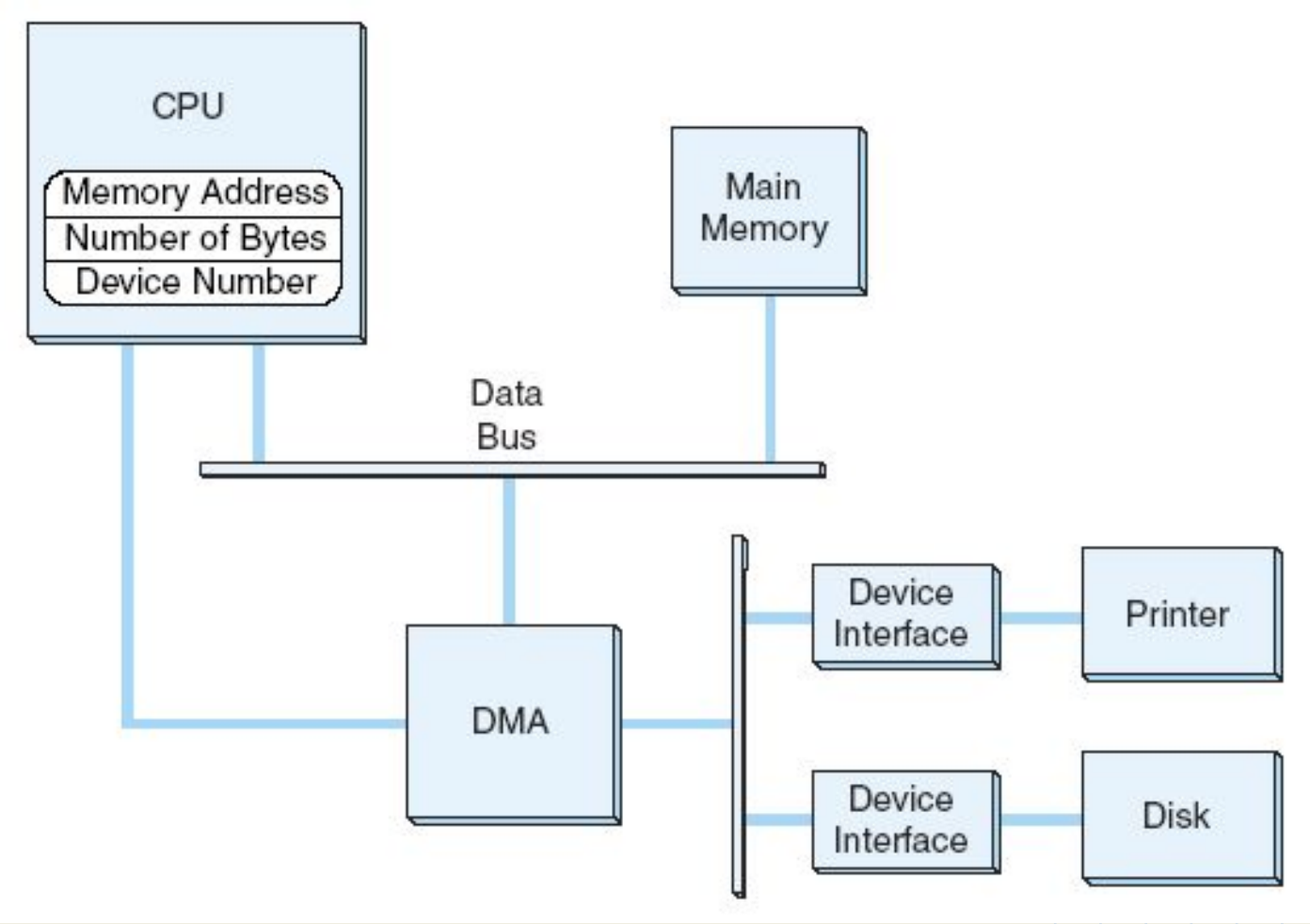


DMA

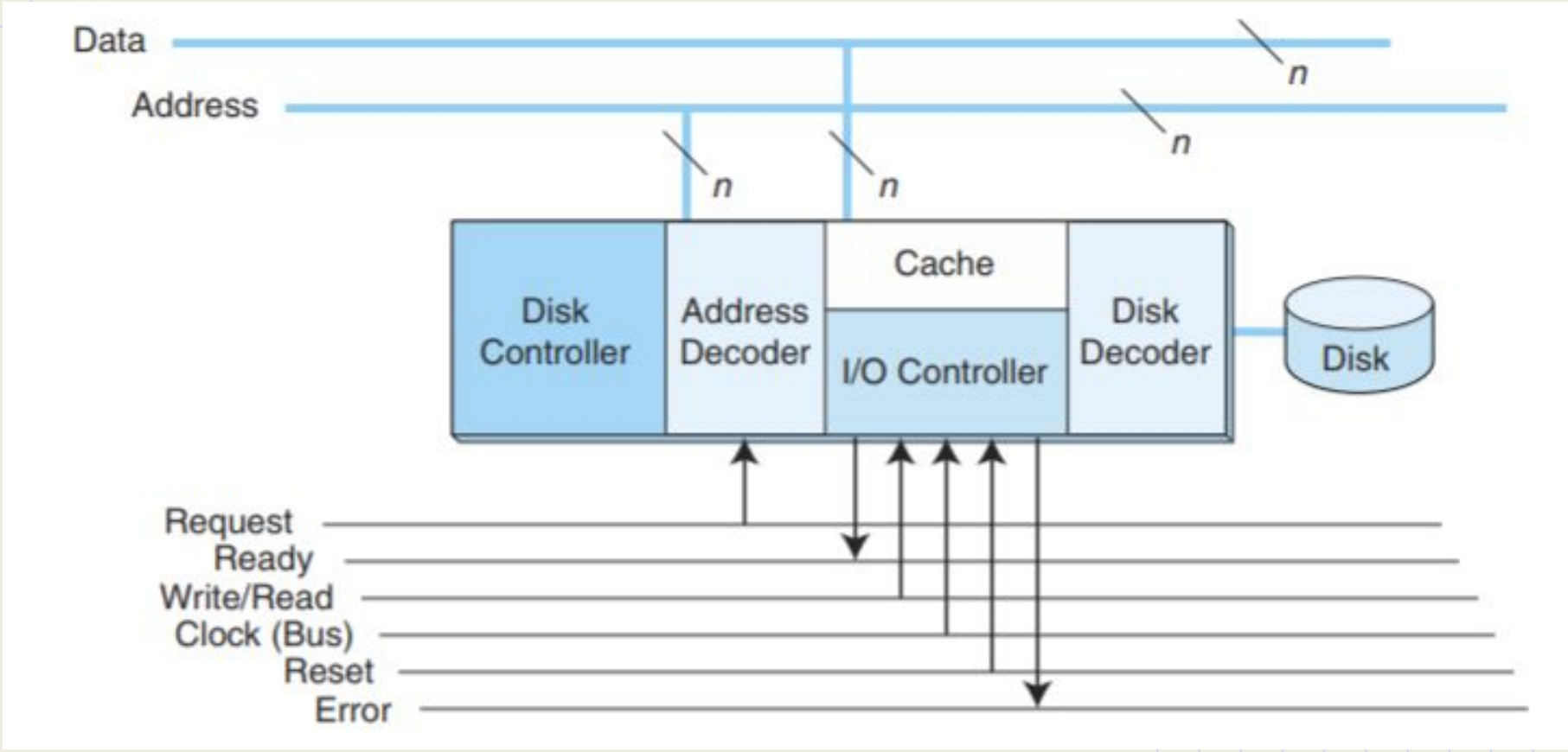
Motivación: ¿Qué pasa si quiero transferir una gran cantidad de datos desde dispositivo a memoria?



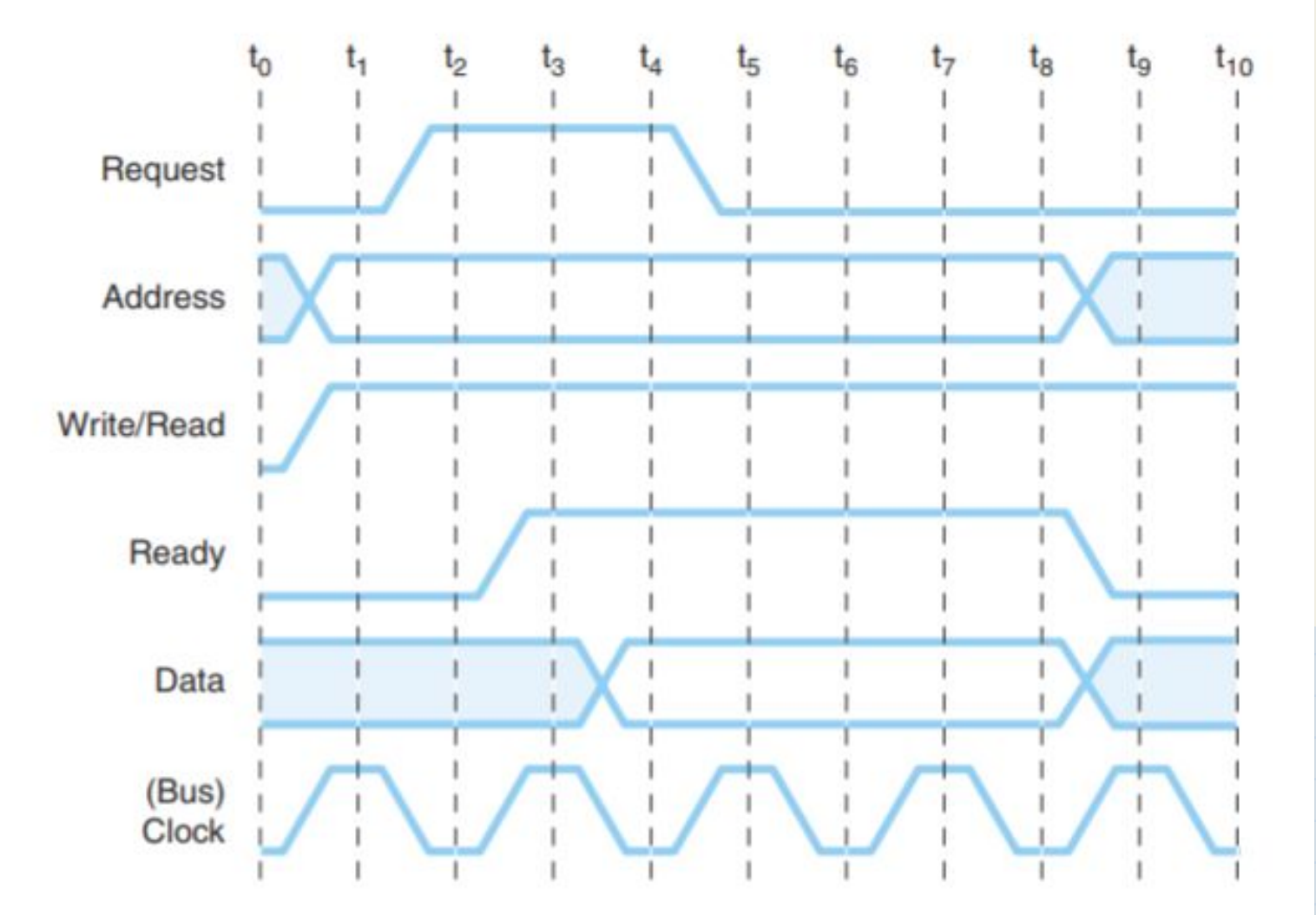
Modelo de DMA



Un ejemplo de un DMA mínimo



Un ejemplo de un DMA mínimo

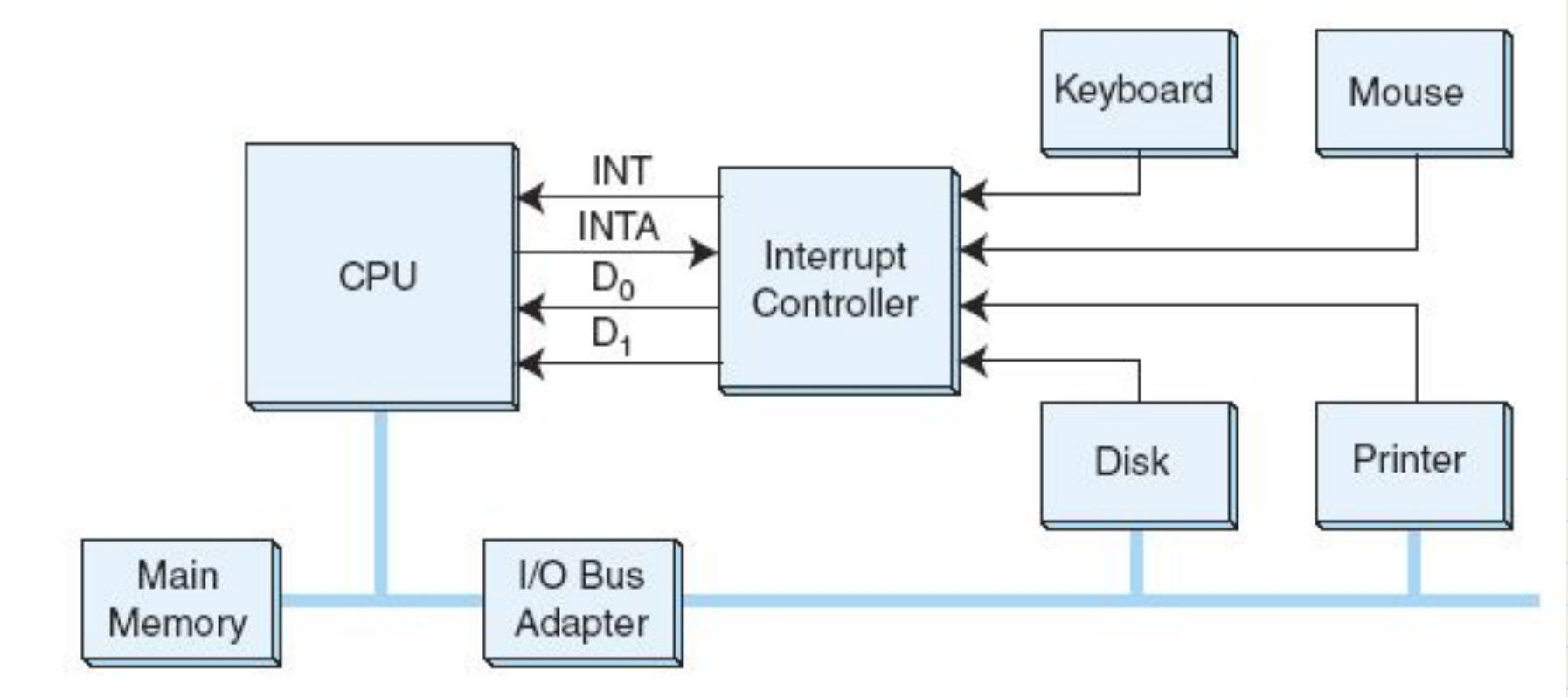


Resumen

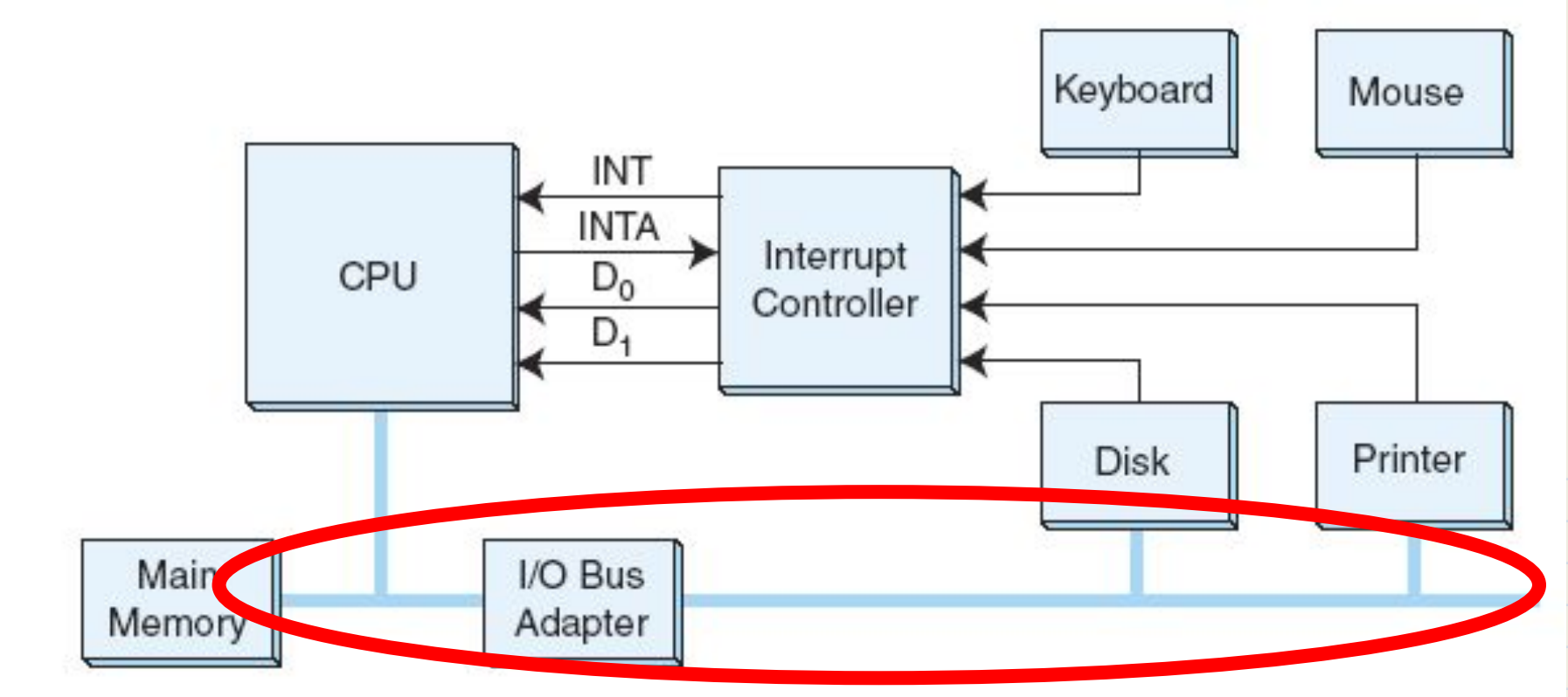
Polling: I/O controlado por CPU
Interrupciones: I/O controlado por dispositivo
DMA: transferencias grandes directas a memoria

Mecanismo I/O	Complejidad HW	Complejidad SW	Velocidad
Polling	+	+++	+
Interrupciones	++	++	++
DMA	+++	+	+++

Modelo de I/O de ORGA1i

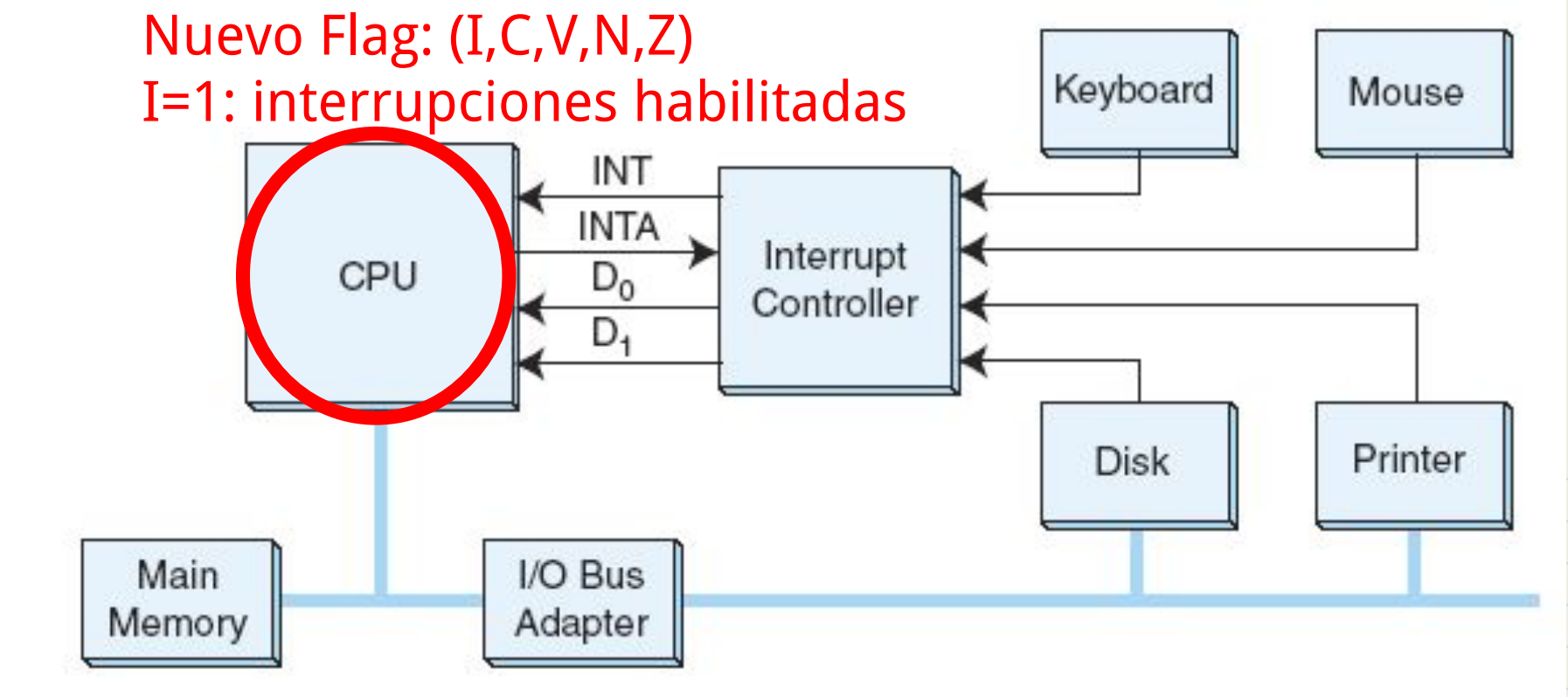


Modelo de I/O de ORGA1i

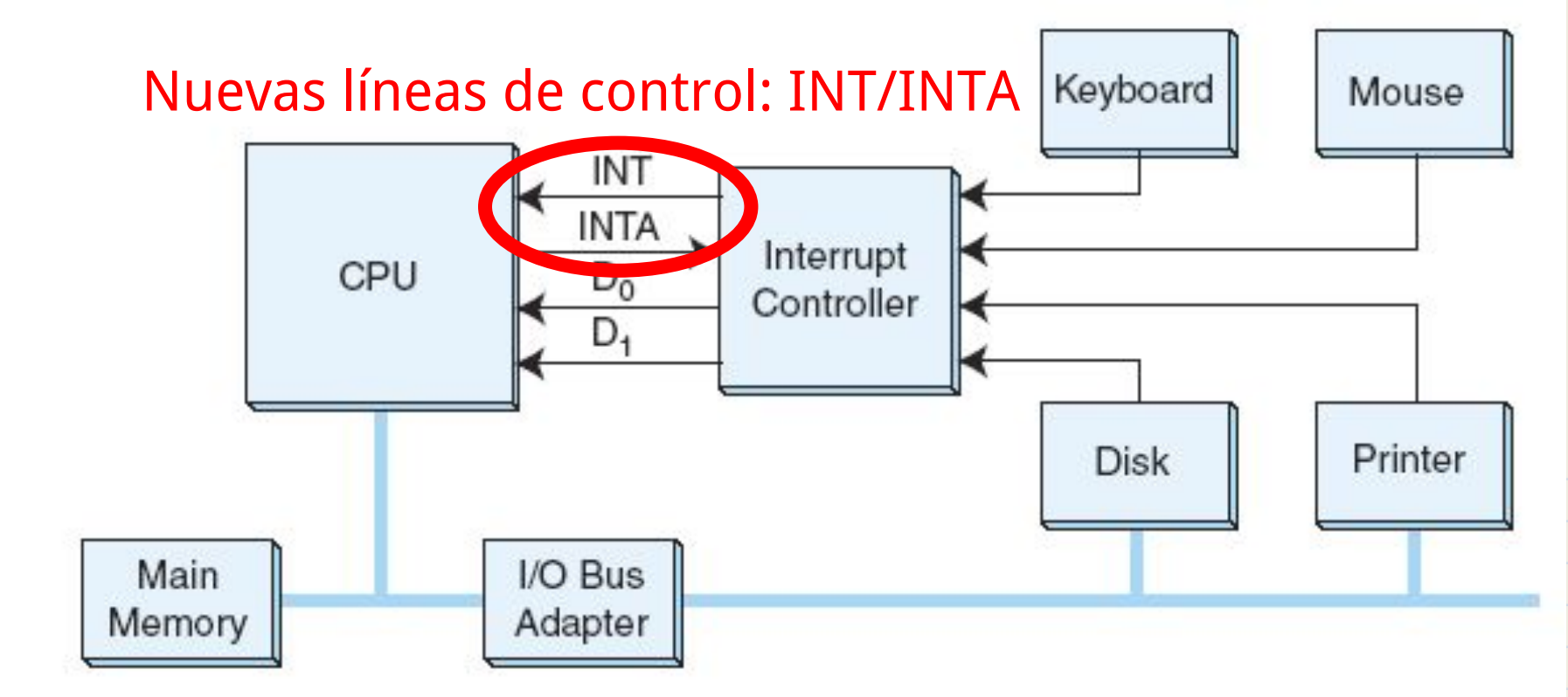


Mapeado a memoria: 0xFFF0 - 0xFFFF.

Modelo de I/O de ORGA1i

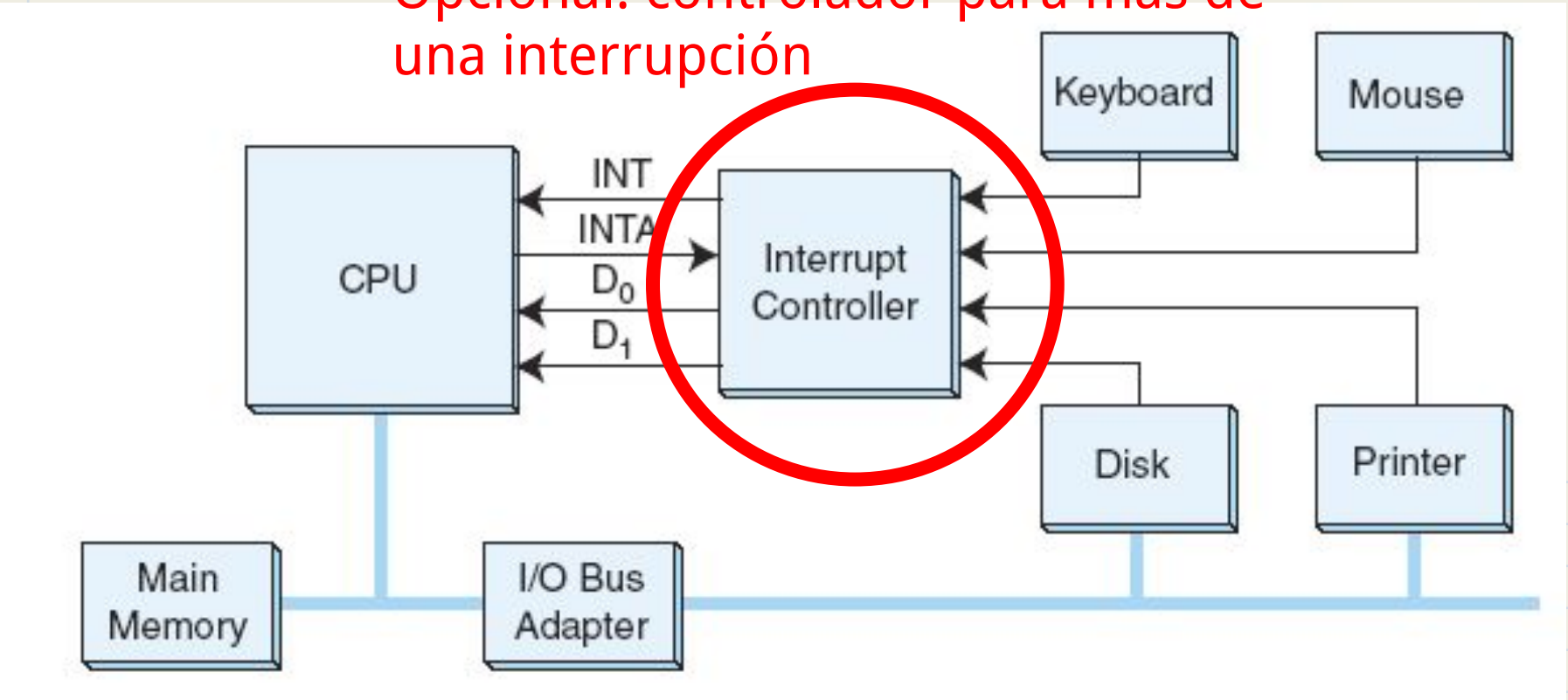


Modelo de I/O de ORGA1i



Modelo de I/O de ORGA1i (opcional)

Opcional: controlador para más de una interrupción



Rutina de atención de ORGA1i

La posición de memoria **0x0000 se reserva** para almacenar la **dirección de la rutina de atención** de la interrupción del dispositivo de E/S.

Al atender la interrupción, ORGA1i realiza **atómicamente:**

1. Coloca [SP]=PSW, y decrementa el SP (SP=SP-1)
2. Coloca [SP]=PC, y decrementa el SP (SP=SP-1)
3. Coloca I=0 para que el procesador no interrumpa
4. Coloca PC=[0x0000]
5. Activa la señal INTA para indicarle al dispositivo que atendió su pedido

Rutina de atención de ORGA1i

Nuevas operaciones disponibles:

- **CLI** que coloca el flag I=0
- **STI** que coloca el flag I=1
- **PUSH Ri** , cuyo efecto es [SP]=Ri y luego SP=SP-1
- **POP Ri** , cuyo efecto es SP=SP+1 y luego Ri=[SP]
- **IRET**, cuyo efecto es PC=[SP],PSW=[SP+1],SP=SP+2

Buses

¿Qué es un bus?

- Un bus es un medio de comunicación entre componentes.
- De **uso compartido**.

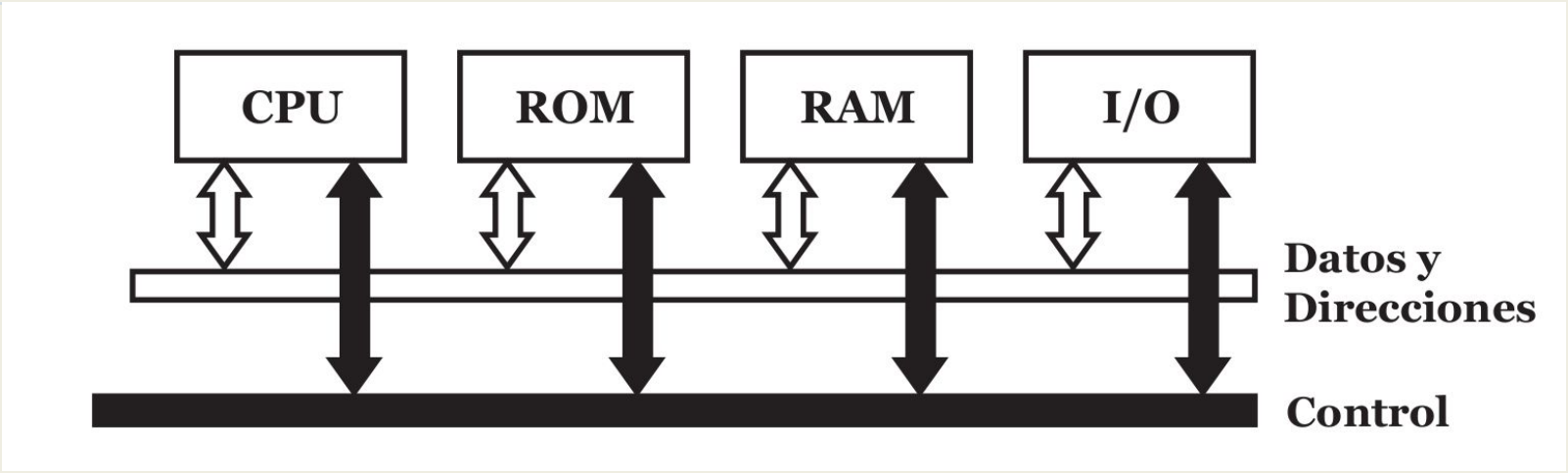
Participantes de un bus

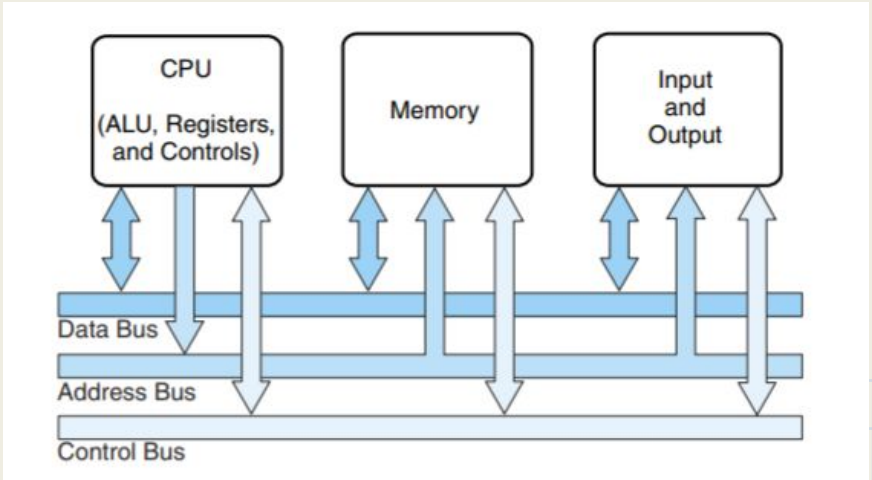
- Hay uno que inicia la comunicación, llamado **master**.
- Otro responde, llamado **slave**.
- Hablamos de **lectura** cuando el **master** quiere **recibir** datos.
- Y de **escritura** cuando el **master** quiere **enviar** datos.

Protocolos

- Hay que **evitar las escrituras simultáneas**
- Los protocolos **sincrónicos** necesitan de un **reloj** para medir el paso del tiempo.
- Los **asincrónicos** no, pero sí necesitan **líneas de control especializadas**.
- Además, los **buses** pueden ser **multiplexados** (varios tipos de buses sobre los mismos cables) o **dedicados** (cada tipo en su propia pista)
  - Los buses dedicados requieren más hardware, pero sus protocolos son más simples.
  - Los buses multiplexados, al revés. Por ende, pueden ser más lentos.

Multiplexado vs dedicado

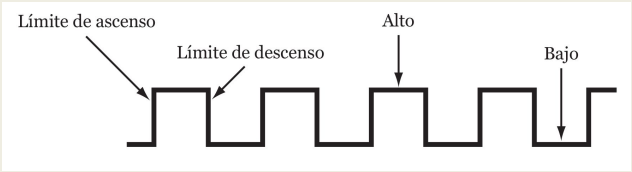




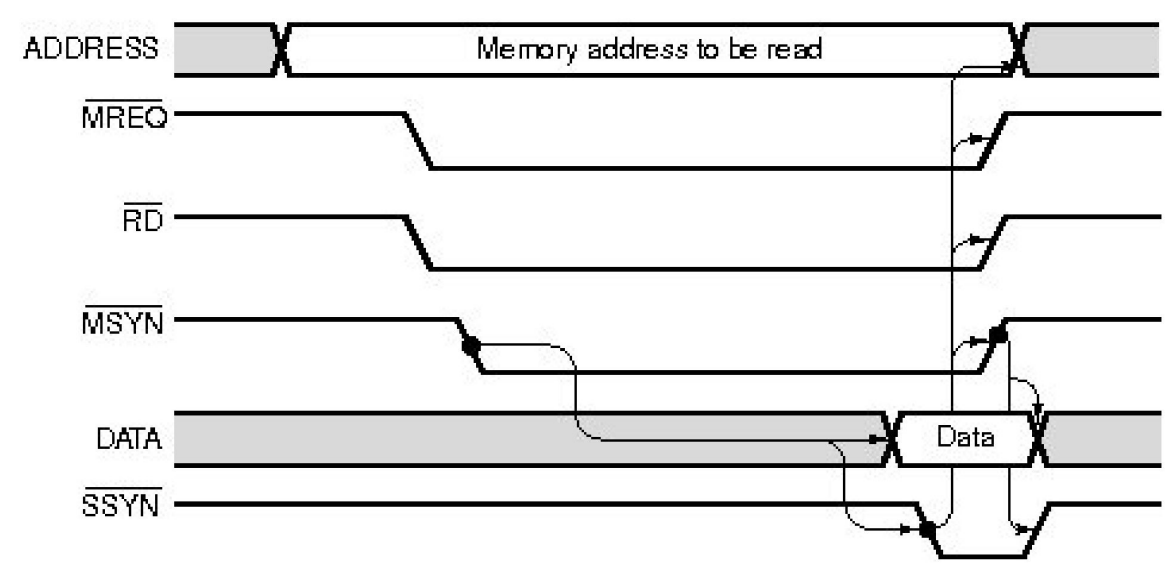
Sincrónico vs asincrónico

Sincrónico: incluye reloj.

- Los participantes escriben durante el ciclo alto.
- Los participantes leen durante el ciclo bajo



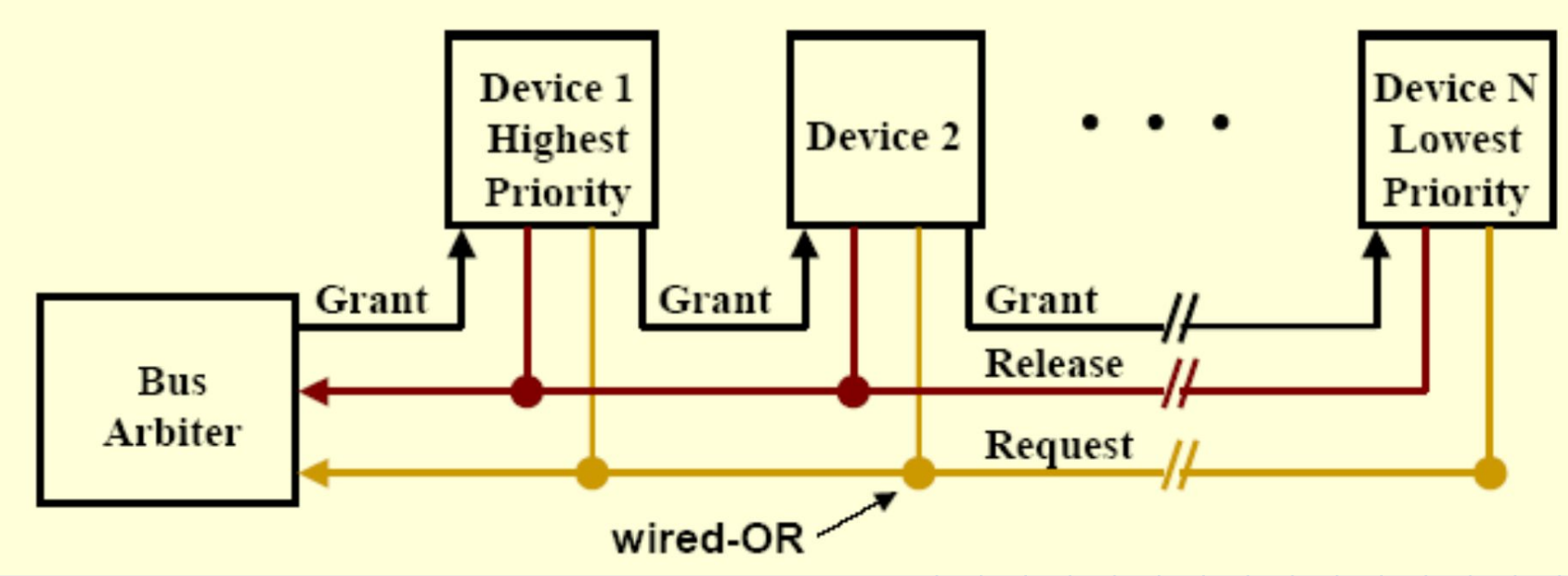
Asincrónico: los eventos que suceden en el bus provocan nuevos eventos



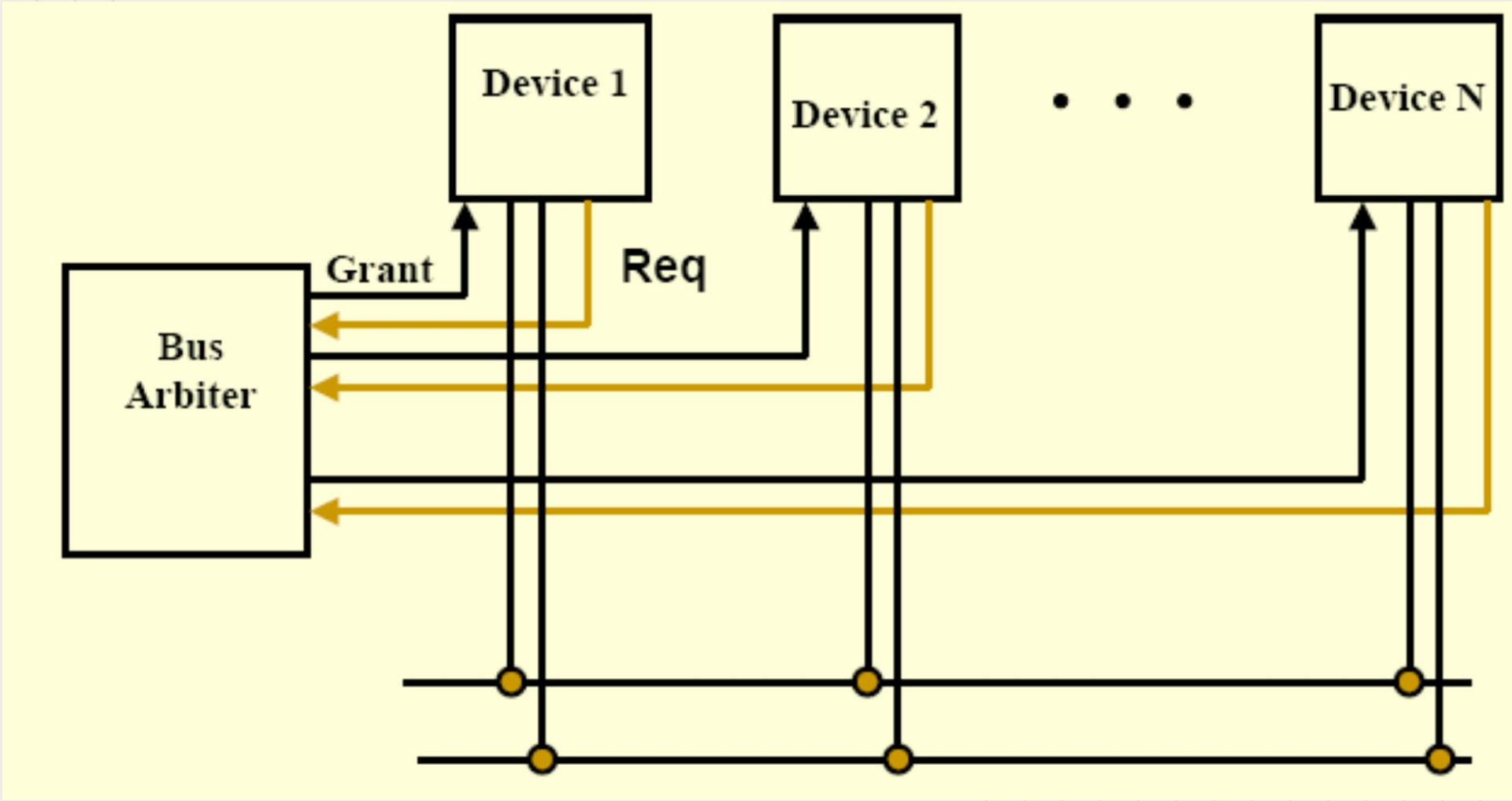
Arbitraje

Cuando hay **muchos dispositivos master**, puede haber **conflicto** en los pedidos

Solución centralizada (con Daisy-Chain)



Arbitraje centralizado



Transmisión de datos

El ancho se define por el número de líneas del bus.

- Ancho del bus de datos → Nro. de accesos a memoria.
- Ancho del bus de direcciones → cantidad direcciones.

Serie

- La información se envía bit a bit en forma secuencial

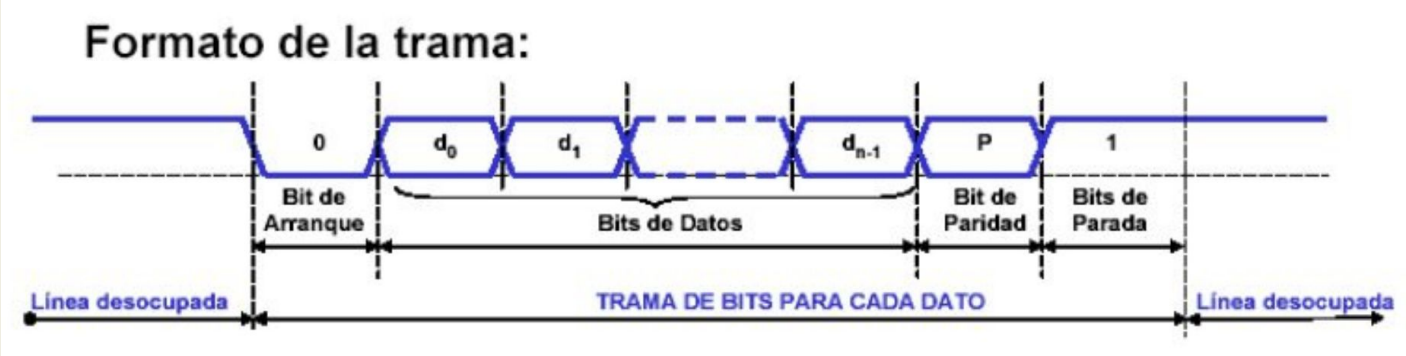
Paralelo

- El bus tiene un ancho predefinido que se transmite en simultáneo

Bus serie

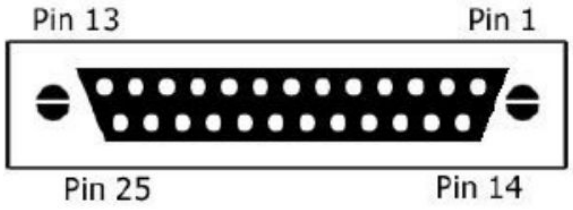
RS232

- Se crea en los años 60.
- La idea es transmitir bit por bit de forma secuencial.
- Además de los bits de datos, existen bit de arranque, de paridad y de parada.



Bus paralelo

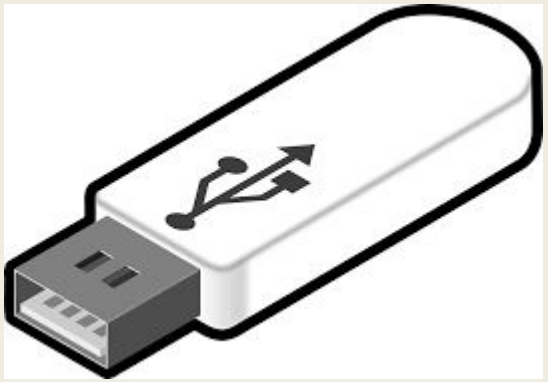
- SPP: Standard Parallel Port
  - 1er estándar bidireccional (IBM 1987)



PIN	Señal	PIN	Señal
1	nSTROBE	10	nACK
2	D0	11	BUSY
3	D1	12	PE
4	D2	13	SELECTED
5	D3	14	nAUTOFEED
6	D4	15	nERROR
7	D5	16	nINIT
8	D6	17	nSELECTIN
9	D7	18-25	GND

USB

- USB (Universal Serial Bus), creado por un consorcio de empresas, entre ellas Intel.
- Creado para sustituir a los buses series (RS-232) y paralelo (IEEE 1284).
- Modos de transferencia:
  - low speed: 1.5 MB/s (USB 1.1)
  - full speed: 12 MB/s (USB 1.1)
  - high speed: 480 MB/s (USB 2.0)
  - USB 3.1: 1.25 GB/s



- Necesita de un host.
- Se pueden conectar hasta 127 dispositivos por host.

Hoy vimos

E/S

- Modelos de I/O: dedicado/mapeado
- Polling: I/O controlado por CPU
- Interrupciones: I/O controlado por dispositivo
- DMA: transferencias grandes directas a memoria
- Modelo de I/O de ORGA1i

Buses

- Multiplexado/dedicado
- Sincrónico/asincrónico
- Arbitraje
- Serie/Paralelo

Bilbiografía: L. Null, Cap. 7 (7.1 a 7.5)