Arquitectura de computadors

Controladors d'un computador

- Introducció
- Classificació dels dispositius d'E/S
- Programació

Introducció

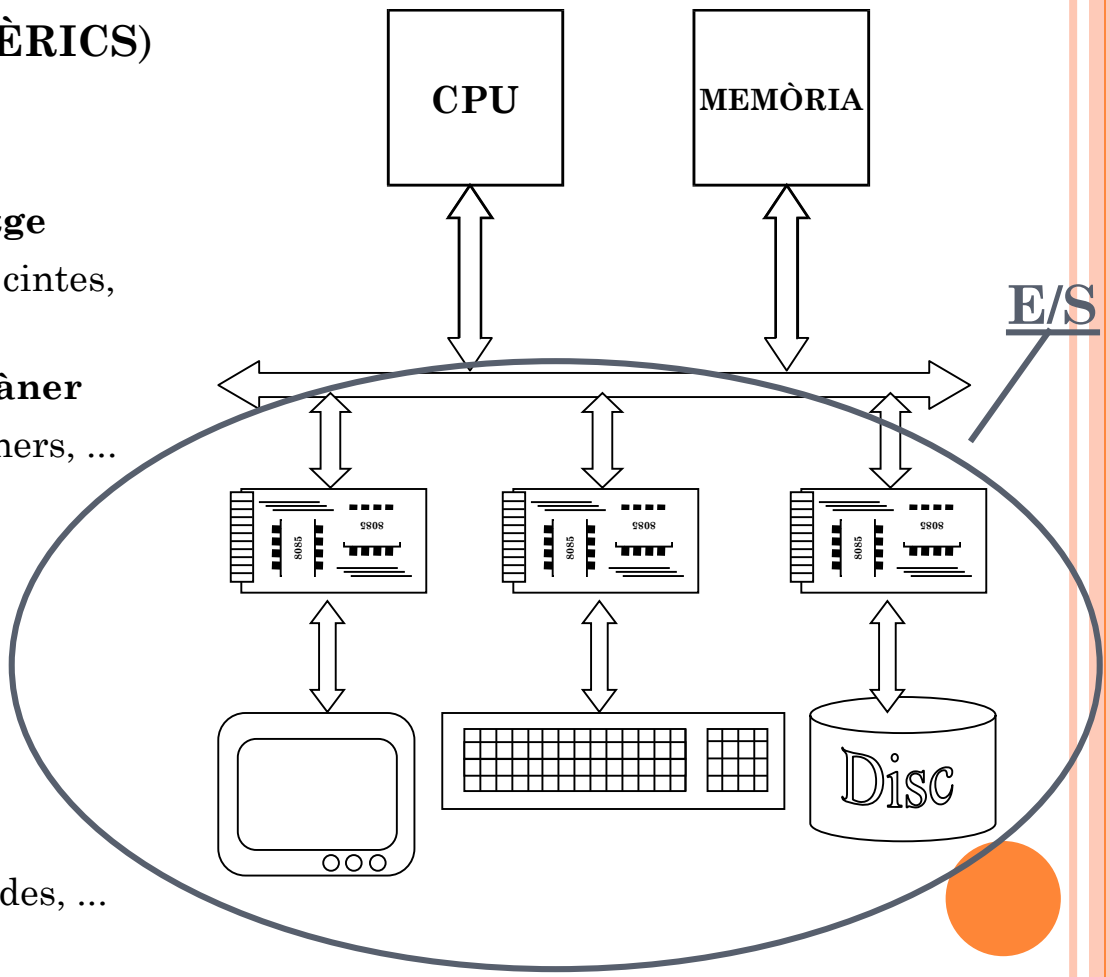
- Objectius:
  - Identificar les funcions bàsiques que necessita una unitat d'E/S independent del perifèric connectat.
  - Analitzar els mecanismes de sincronització entre la unitat d'E/S i la CPU.
  - Estudiar el sistema d'accés directe a memòria (DMA) utilitzat quan la velocitat i el volum de dades és elevat.

- Contingut:
  - Funcions implicades en les operacions d'entrada/sortida
  - Estructura del sistema d'E/S: mòduls d'e/s i controladors de dispositius
  - Mecanismes bàsics d'e/s: sincronització
    - E/S controlada por programa
    - E/S por interrupció: gestió de les interrupcions
    - E/S por accés directe a memòria (*DMA): motivació*

CLASSIFICACIÓ DELS DISPOSITIUS D'E/S

Necessitat de les E/S

- L'E/S permet al computador interactuar amb el mon exterior
- Dispositius típics d'E/S (**PERIFÈRICS**)
  - **Dispositius d'E/S bàsics**
    - teclat, ratolí, pantalla
  - **Dispositius d'emmagatzematge**
    - discos, disquets, CD-ROM, cintes, discos magnetoòptics, ...
  - **Dispositius d'impressió i escàner**
    - impressores, plotters, secaners, ...
  - **Dispositius de comunicació**
    - xarxes, mòdems, ...
  - **Dispositius multimèdia**
    - àudio, vídeo, ...
  - **Dispositius d'automatització i control**
    - sensors, alarmes, sistemes d'adquisició de dades, ...



CLASSIFICACIÓ DELS DISPOSITIUS D'E/S

Varietat de dispositius d'E/S

- **Comportament**
  - Entrada: Només es llegeix un cop.
  - Sortida: Només es pot escriure.
  - Emmagatzematge: Es pot llegir i escriure diferents cops
- **Interlocutor:** Persona o maquina que està a l'altra banda del dispositiu d'E/S i que li envia dades a la seva entrada o llegeix dades de la seva sortida
- **Velocitat de transferència de dades:** ritme màxim de transferència entre el dispositiu d'E/S i la memòria principal o el processador. És útil conèixer, en dissenyar un sistema d'E/S, quina és la demanda màxima que el dispositiu pot generar.

- Exemple:
  - ✓ Un teclat és un dispositiu d'entrada amb una velocitat de transferència de dades màxim d'uns 10 bytes per segon.

Dispositiu	Comportament	Interlocutor	Velocitat de transferència de dades (Mbits/s)
Ratolí	Entrada	Humà	0,0038 (3800 bits/s)
Entrada de veu	Entrada	Humà	0,2640 (264000 bits/s)
Entrada de so	Entrada	Màquina	3.0000
Escàner	Entrada	Humà	3.2000
Pantalla gràfica	Sortida	Humà	Entre 800 i 8.000
LAN i LAN sense fil	E/S	Màquina	Entre 100 i 1.000 // 11 i 54
Disc magnètic	E/S	Màquina	Entre 240 i 2.560

CLASSIFICACIÓ DELS DISPOSITIUS D'E/S

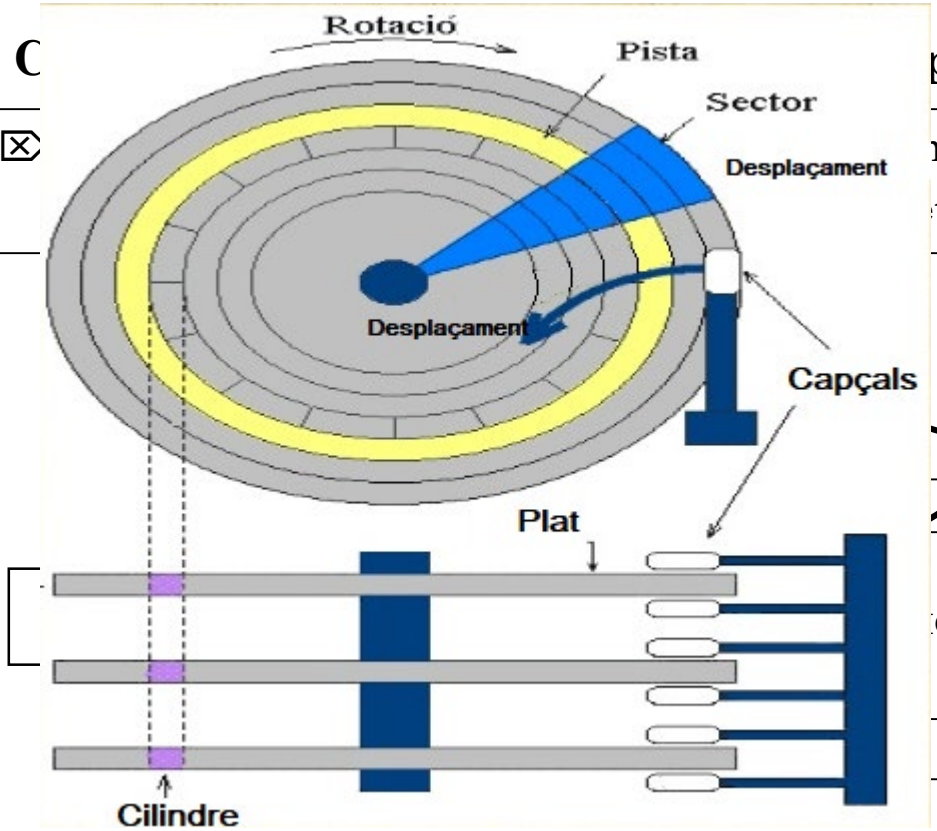
Funcions bàsiques del sistema d'E/S

- **Adreçament**
  - És necessari seleccionar el dispositiu d'E/S amb el que es vol fer la transferència
- **Transferència de dades** entre el computador i el perifèric
  - Tipus de transferència
    - Lectura: computador ← perifèric
    - Escriptura: computador → perifèric
  - Pot necessitar conversions de format de les dades
    - Conversió de nivells elèctrics (CMOS 1 → V>2/3Vcc; 0 → V<1/3Vcc)
      - ✓ TTL: 1 → V > 2,0 Volts; 0 → V < 0,8 Volts
      - ✓ RS-232-C: 1 → V < -3.0 Volts; 0 → V > +3.0 Volts
    - Conversió del tipus de codificació
      - ✓ Caràcters (ASCII, EBCDIC)
      - ✓ Sencers (magnitud i signe, C'1, C'2, ...)
      - ✓ Reals (punt fix, punt flotant, simple precisió, doble precisió, ...)
    - Conversió sèrie - paral·lel / paral·lel - sèrie
    - Conversió digital-analògic / analògic-digital
- **Sincronització i control de la transferència**
  - Necessari un mecanisme de sincronització de la transferència
    - El computador ha de saber
      - ✓ Si el perifèric està preparat per enviar o rebre dades
      - ✓ Si el perifèric ha acabat de realitzar una transferència i en pot iniciar una de nova

PROGRAMACIÓ DELS DISPOSITIUS D'E/S

C…putador

- …nputador per un **INTERFÍCIE d'E/S**
  …ta d'E/S



Exemple: Lectura del disc

Ordres CPU → Interfície
Llegir N bytes a partir de
Plat P
Cilindre C
Sector S

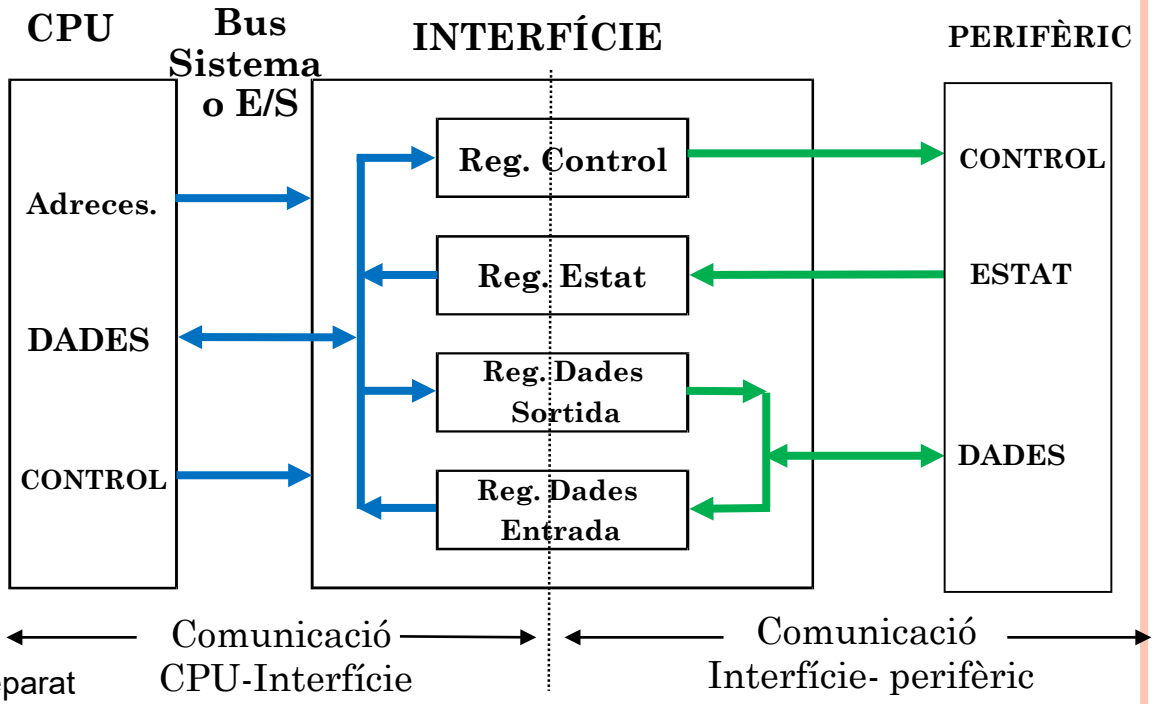
Ordres Interfície → perifèric
Posicionar capçals al cilindre C
Posicionar capçals al sector S
Seleccionar capçal de plat P
Llegir N bytes
Retirar capçals

Funcions del interfície d'E/S

- Interpretar les comandes que rep de la CPU i transmetre-les al perifèric
- Controlar la transferència de dades entre la CPU i el perifèric
  - Conversió de formats
  - Adaptar la diferència de velocitats entre CPU i perifèric (mitjançant buffers d'emmagatzematge)
- Informar a la CPU de l'estat del perifèric

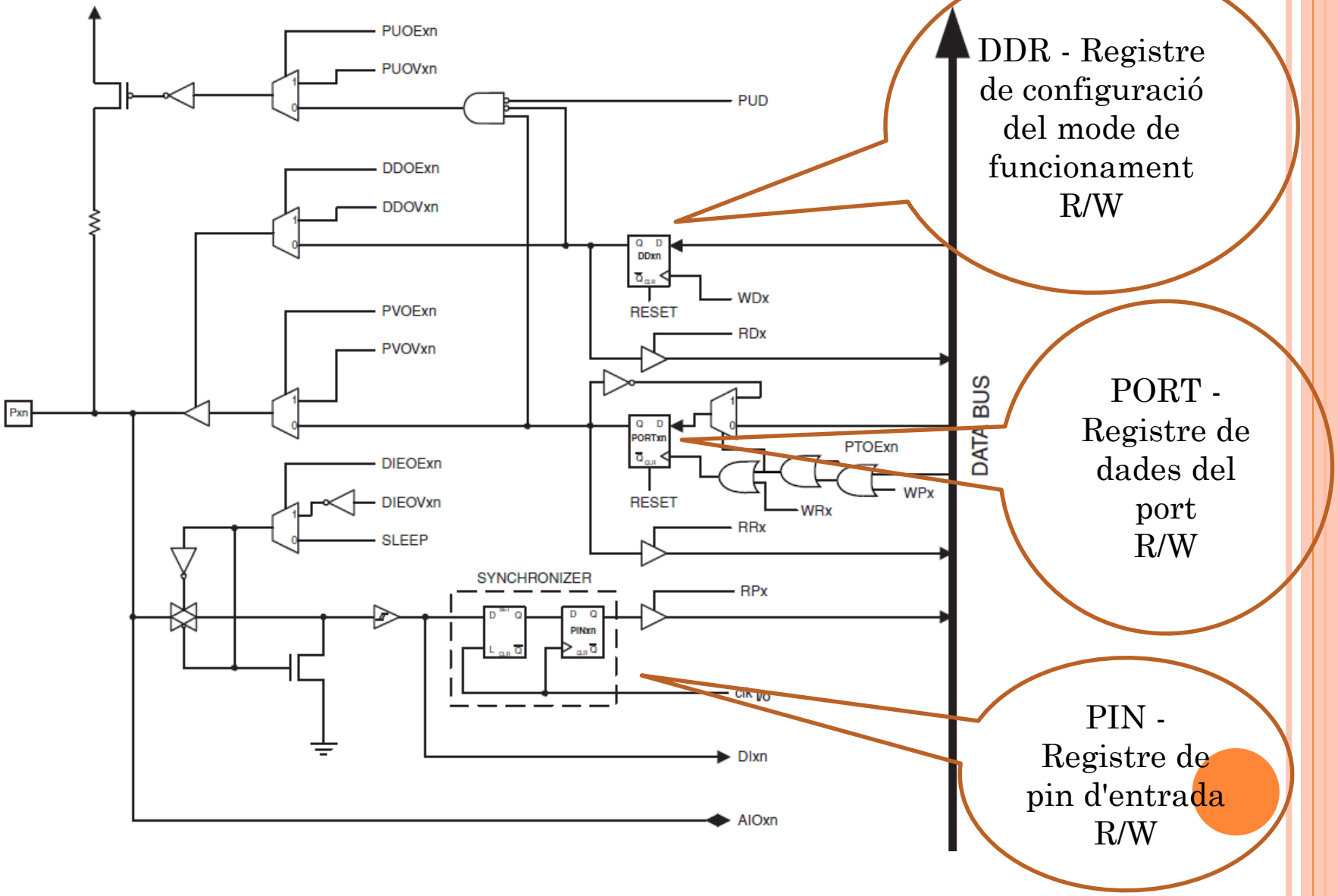
Estructura dels interfícies d'E/S

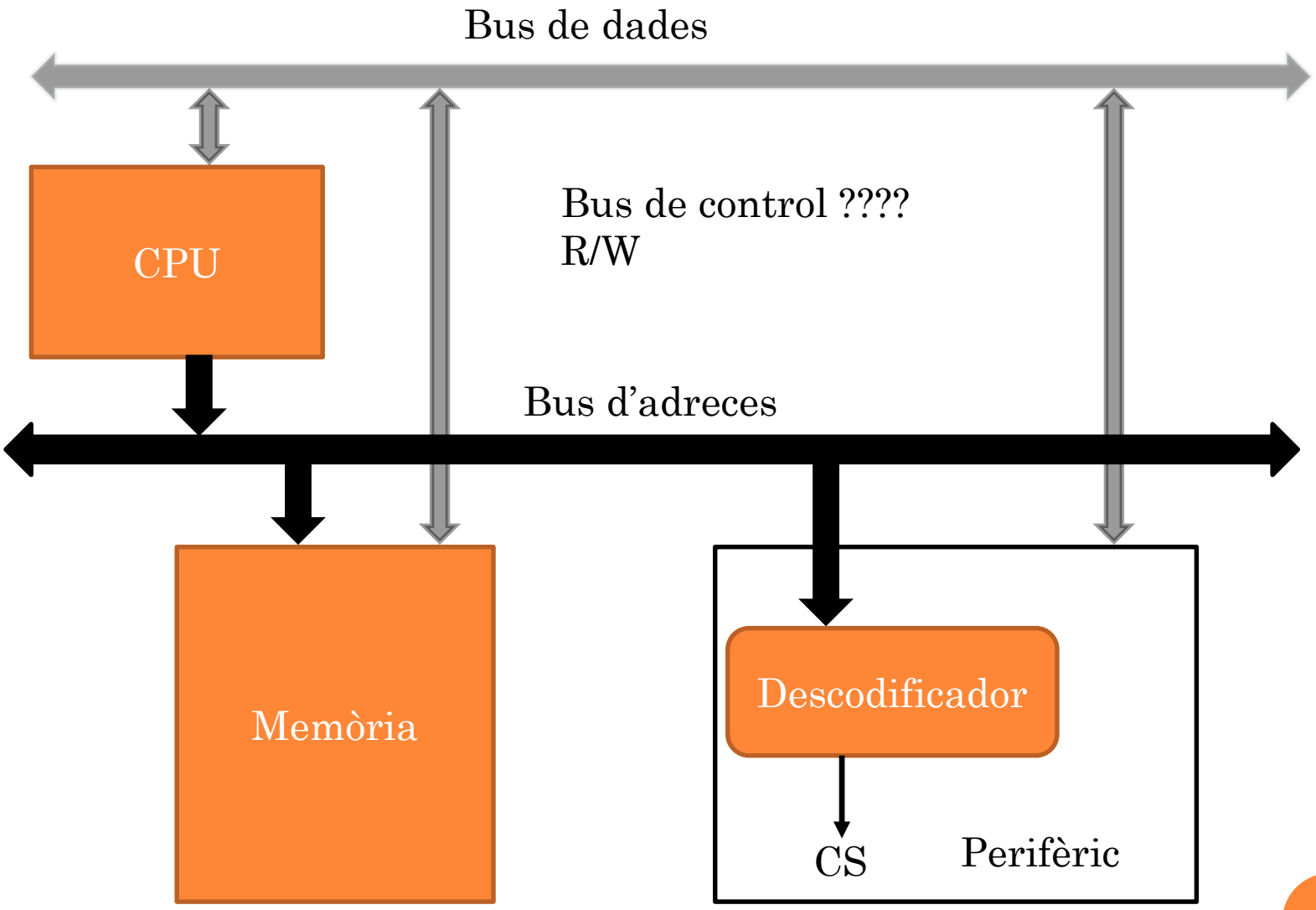
La comunicació entre la CPU i el perifèric es realitza mitjançant els registres del interfície



- **Registre de dades de sortida**
  - Quan la CPU vol enviar dades al perifèric les escriu sobre el registre(s) de dades de sortida
- **Registre de dades d'entrada**
  - Quan la CPU vol rebre dades del perifèric les llegeix des del registre(s) de dades d'entrada
- **Registre d'estat**
  - Quan la CPU vol saber l'estat del perifèric, ho fa llegint el registre d'estat:
    - Dispositiu preparat o no preparat
    - Registre de dades ple o buit
    - Transferència finalitzada o no finalitzada, etc.
- **Registre de control**
  - Quan la CPU vol transmetre una comanda al perifèric ho fa escrivint al registre de control
    - Llegir/escriure N bytes al cilindre C, pista P, sector S (per discos)
    - Rebobinar / avançar / llegir N bytes (per cintes)
    - Imprimir caràcter / saltar de línea / saltar de pàgina (per impressores), etc.

ATmega328P: Port

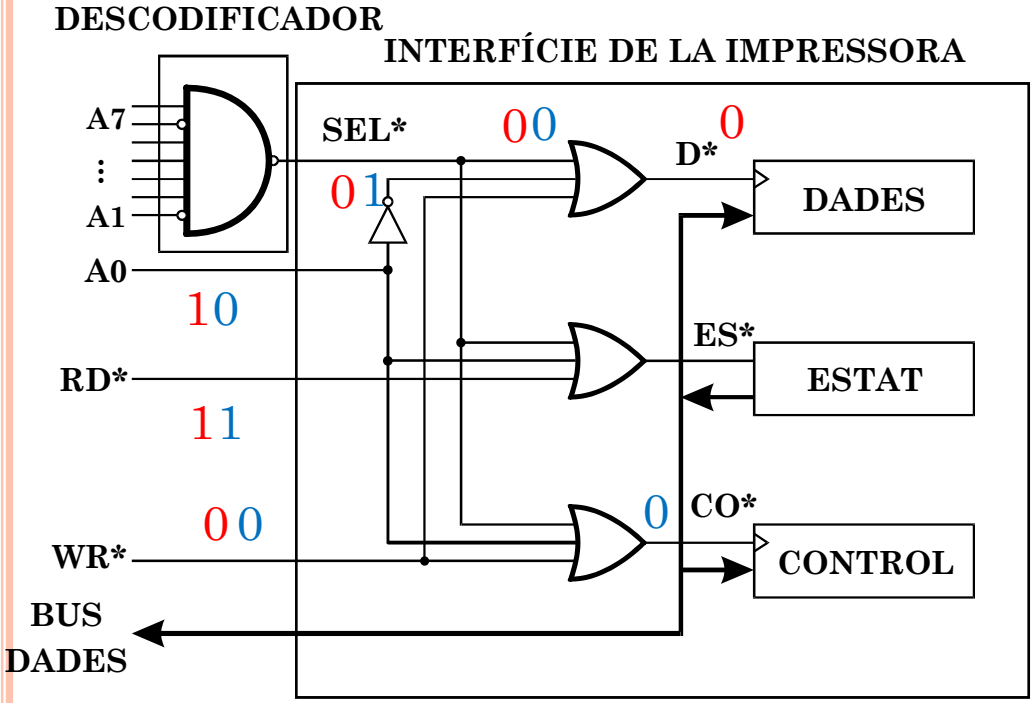



Bus de dades
CPU
Bus de control ????
R/W
Bus d'adreces
Memòria
Descodificador
CS
Perifèric

PROGRAMACIÓ DELS DISPOSITIUS D'E/S

Exemple de connexió d'un interfície d'E/S al bus

0xBD = 1011 1101 0xBD = 1011 1100



- Enviar un caràcter emmagatzemat al registre R1 a la impressora
  - MOVE o SB R1, $BD
  - OUT R1, $BD
- Enviar una comanda emmagatzemada al registre R2 a la impressora
  - MOVE o SB R2, $BC
  - OUT R2,$BC
- Llegir l'estat de la impressora i guardar-lo al registre R3
  - MOVE o LB $BC, R3
  - IN $BC, R3

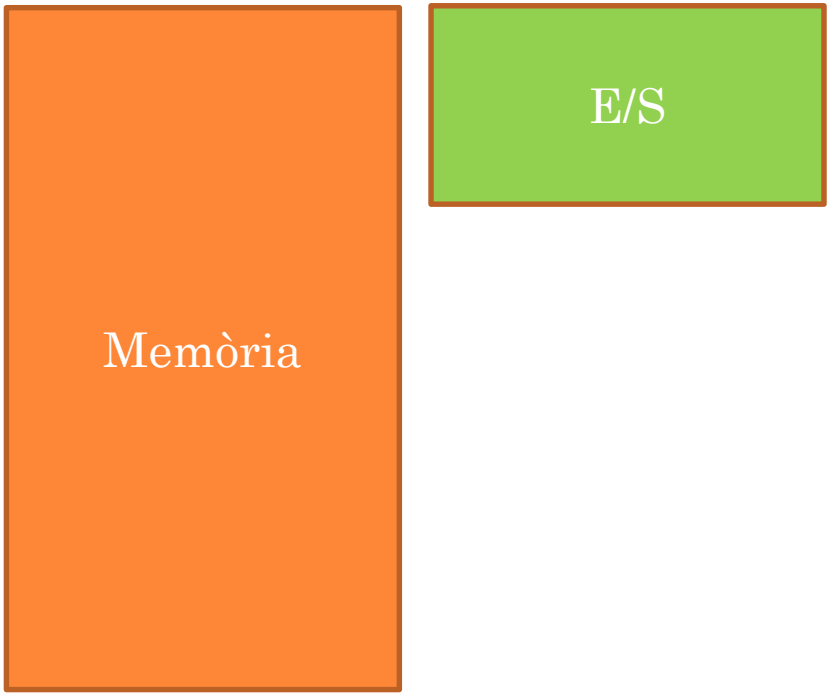
Adreça:	Registre Estat/Control:	10111100 ($BC)
Adreça:	Registre Dades:	10111101 ($BD)

Nota: Els registres del interfície d'E/S també s'anomenen ***ports d'E/S***

Programació dels dispositius d'E/S

Alternatives de disseny de sistemes d'E/S
- **E/S aïllada**
- **E/S localitzada a memòria**

E/S aïllada



E/S localitzada a memòria

E/S aïllada

- ***L'E/S i la memòria utilitzen un espai d'adreçament diferent***
  - El conjunt d'adreces que utilitza la memòria i el que utilitzen les E/S son independents
- ***Disposen d'un conjunt d'instruccions específiques d'E/S***
  - IN dir_E/S, Ri (CPU ← Perifèric)
  - OUT Ri, dir_E/S (Perifèric ← CPU)
- ***El bus disposa de línees de control específiques*** (MEM/IO*) per indicar si es tracta d'una operació amb memòria o d'una operació d'E/S
  - Si MEM/IO* = 1
    - Operació de memòria (MOVE, LOAD, STORE) ⇒ L'adreça del bus es correspon a una posició de memòria
  - Si MEM/IO* = 0
    - Operació d'E/S (IN, OUT) ⇒ L'adreça del bus es correspon a un port d'E/S
  - Un port d'E/S pot tenir assignada la mateixa adreça que una posició de memòria valida
    - És impossible que hi hagi ambigüitat gracies a l'existència de la línea MEM/IO*
- ***Exemples***
  - i8086 i altres computadores de la família intel x86

E/S localitzada a memòria

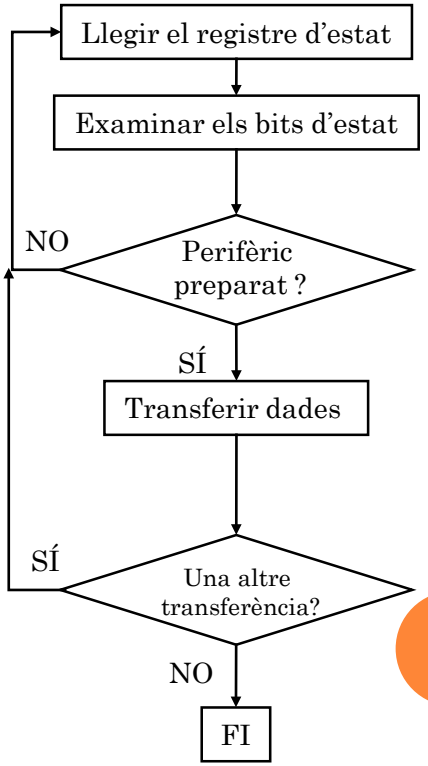
- ***Les E/S i la memòria comparteixen el mateix espai d'adreces***
- ***No es necessiten instruccions específiques d'E/S***
  - Les mateixes instruccions que s'utilitzen per fer transferències de dades amb la memòria (LOAD i STORE) es poden utilitzar per fer les operacions d'E/S
    - LB Ri, adreça d'E/S (CPU ← Perifèric)
    - SB Ri, adreça d'E/S (Perifèric ← CPU)
- ***Al bus no hi ha una línea especial*** per diferenciar operacions amb la memòria de les operacions d'E/S
  - Un port d'E/S no pot tenir assignada la mateixa adreça que una posició de memòria vàlida
  - Normalment s'assigna als dispositius d'E/S una porció contigua de l'espai d'adreces que no s'utilitzi per la memòria
- ***Avantatges de l'E/S localitzada a memòria***
  - És més flexible que l'E/S aïllada ja que permet realitzar diferents tipus d'operacions sobre els ports d'E/S (aritmètiques, lògiques, manipulació de bits, etc.) i no només de moviment de dades
- ***Exemple***
  - MC68000
    - 24 Línies d'adreces ⇒ espai d'adreces de 16 Mbytes
    - Es pot dividir l'espai d'adreces en dues parts, per exemple:
      - ✓ Adreces assignades a memòria: de $000000 a $BFFFFF (primers 12 Mbytes)
      - ✓ Adreces assignades a E/S: de $C00000 a $FFFFFF (últims 4 Mbytes)

Sincronització de les E/S

- Quan la CPU vol enviar/rebre dada/des d'un perifèric s'ha de garantir que el dispositiu està preparat per realitzar la transferència: **sincronització entre la CPU i el dispositiu d'E/S**
- Existeixen dos mecanismes bàsics de sincronització de l'E/S
  - E/S programada per enquesta
  - E/S per interrupcions

E/S programada per enquesta

- Cada vegada que la CPU vol fer una transferència entra en una iteració en la que es consulta l'estat del perifèric fins que aquest està preparat per realitzar la transferència
- **Problemes**
  - La CPU no pot està fent treball útil durant l'enquesta
    - Amb dispositius lents l'enquesta podria repetir-se moltes vegades
  - La dinàmica del programa s'atura durant l'operació d'E/S
    - Exemple: en un videojoc no es pot aturar la dinàmica del joc esperant que l'usuari premi una tecla o mogui el *joystick*
  - Dificultats per atendre a varis perifèrics
    - Mentre s'espera a que un perifèric estigui apunt per fer la transmissió, no es pot atendre a un altre perifèric



Programació dels dispositius d'E/S

Exemple: E/S programada per enquesta

- Ratolí: el processador verifica periòdicament l'estat del ratolí, per veure si l'usuari l'ha mogut, si ho ha fet, el S.O. informa al programa associat del canvi de situació.
- Normalment, els bucles d'espera activa (enquesta continuada) només es poden utilitzar en dispositius dedicats (sistemes enquestats)
- Per computadors normals (S.O. de temps compartit), és més habitual l'enquesta periòdica (per exemple, pel ratolí)

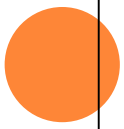
- Suposem que en una CPU a 500 MHz una operació d'enquesta necessita 400 cicles de rellotge (cridar a la rutina, accedir al dispositiu i retornar). Determineu el % de temps que la CPU gasta realitzant sondejos pels següents dispositius:
  - Ratolí: s'ha de fer 30 vegades per segon per no perdre el moviment que pugui fer l'usuari.
  - Disquet: transfereix dades en unitats de 2 bytes i té una amplada de banda de 50 KiB/s. No es poden perdre dades.
  - Dics dur: transfereix dades en unitats de 16 bytes i té una amplada de banda de 8 MiB/s. No es poden perdre dades.

KiB -> Kibibyte -> 2^{10} MiB -> Mebibyte -> 2^{20}

Programació dels dispositius d'E/S

Exemple: E/S programada per enquesta

- Cicles per segon consumits en el sondeig del ratolí:
  - 30 * 400 = 12.000 cicles/s consumits
  - % CPU: 12.000/(500*10^6)*100 = 0,0024% → Impacte menyspreable
- Cicles per segon consumits en el sondeig del disquet:
  - 50 KiB/s / 2 bytes per transferència = 25.600 enquestes per segon
  - 25.600 * 400 = 10.240.000 cicles/s consumits
  - % CPU: 10.240.000/(500*10^6)*100 = 2,048% → Impacte apreciable, però permissible si no hi ha molts dispositius.
- Cicles per segon consumits en el sondeig del disc dur:
  - 8 MiB/s / 16 bytes per transferència = 524.288 enquestes per segon
  - 524.288 * 400 = 209.715.200 cicles/s consumits
  - % CPU: 209.715.200 /(500*10^6)*100 = 41,95% → Impacte inacceptable



- 1KiB = 2^{10} = 1.024 Bytes // 1MiB = 2^{20} = 1.048.576 Bytes

Interrupcions

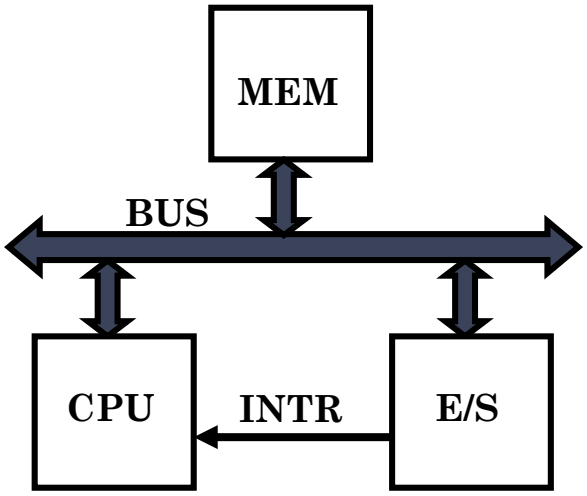
- Gestió
  - Seqüència d'esdeveniments
  - Peticions
  - Inhibició
  - Identificació
  - Interrupcions multi nivell

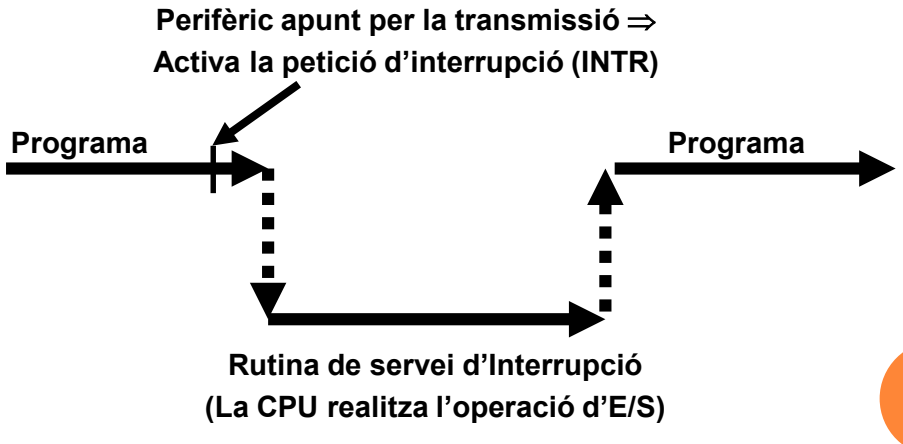
- Exemple
  - ✓ Intel
  - ✓ MIPS

PROGRAMACIÓ DELS DISPOSITIUS D'E/S

E/S per interrupció

- No hi ha el bucle d'espera
- Quan un perifèric està apunt per transmetre ho comunica a la CPU activant una línia especial del bus de control denominada **LÍNIA DE PETICIÓ D'INTERRUPCIÓ (INT o INTR)**
- **RUTINA DE SERVEI D'INTERRUPCIONS (RSI)**
  - Quan la CPU rep una petició d'interrupció salta a executar una **RSI**
  - La RSI s'encarrega de atendre al perifèric que ha fet la petició i realitzar l'operació d'E/S

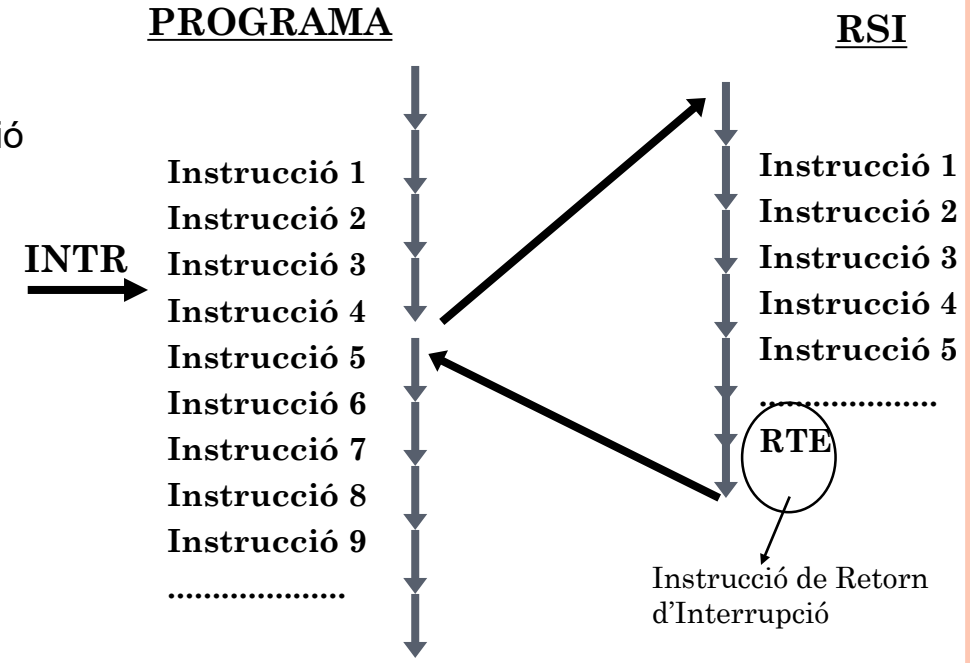




PROGRAMACIÓ DELS DISPOSITIUS D'E/S

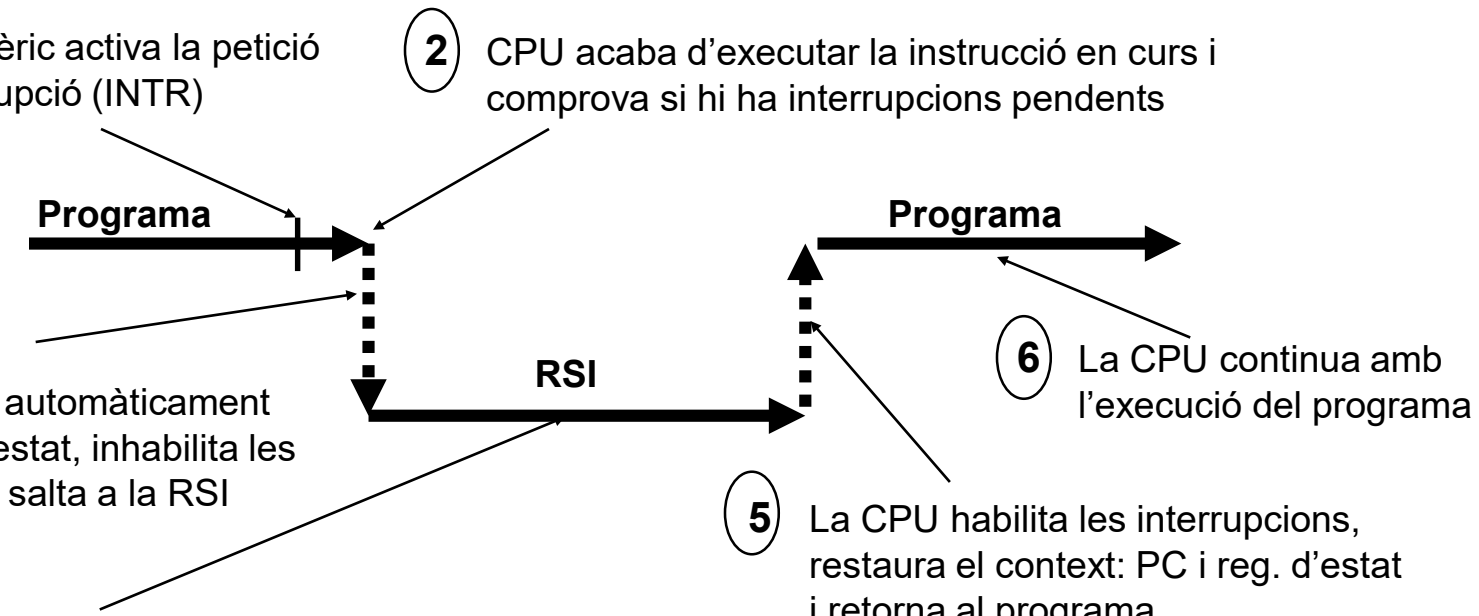
E/S per interrupció

- **Analogies entre una subrutina i una RSI**
  - Es trenca la seqüència normal d'execució
  - Quan acaba l'execució s'ha de retornar al punt on estava.
    - ⇨ S'ha de **guardar el PC** en els dos casos
- **Diferencies entre una subrutina i una RSI**
  - En una subrutina el programador sap en quin punt exacte es trenca la seqüència.
  - Una RSI es pot executar en qualsevol moment.
    - ⇨ Necessari **guardar el registre d'estat** i restaurar-lo al retornar de la RSI
      - ✓ Normalment es realitza automàticament
    - ⇨ Necessari **guardar els registres** que utilitza la RSI a la pila i restaurar-los al retornar de la RSI
      - ✓ Normalment s'ha de fer manualment



Gestió d'interrupcions

Seqüència d'esdeveniments en el tractament d'una interrupció



4 La CPU executa la RSI, i:

- Informa al perifèric que s'ha reconegut la seva interrupció (per programa o per maquinari)
- Salva a la memòria tots els registres de dades i/o adreces utilitzades per la RSI (manual)
- Realitza l'operació d'E/S amb el perifèric
- Restaura els registres de dades/adreces
- Executa la instrucció de retorn d'interrupció (RTE)

Questions plantejades

- Quan comprova la CPU si hi ha interrupcions pendents?
- Per què és necessari inhabilitar les interrupcions?
- Com s'informa al perifèric que s'ha reconegut la seva interrupció?
  - ⇒ *Identificació de la font d'interrupció*
- Què succeeix si es produeix una segona interrupció durant l'execució de la RSI?
  - ⇒ *Interrupcions multi nivell i encadenament d'interrupcions*

GESTIÓ D'INTERRUPCIONS

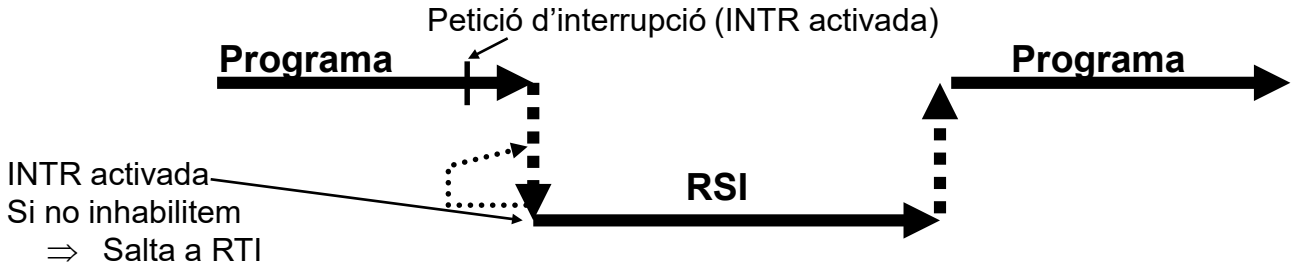
Comprovació de peticions d'interrupció pendents

- La CPU comprova si hi ha interrupcions pendents (línia INTR activada) al final de l'execució de cada instrucció
  - **Motiu:**
    - Només és necessari guardar el PC, el reg. d'estat i els registres que modifica el programa (registres de dades i/o adreces)
    - Si es pugues interrompre una instrucció a la meitat de l'execució seria necessari guardar el valor de tots els registres interns de la CPU
      - ✓ Reg. de instrucció, registres d'adreces, registres de dades de memòria, etc.
  - **Excepcions:**
    - Instruccions de llarga duració
      - ✓ Per exemple, en instruccions de moviment múltiple (MOVEM), es comprova si hi ha interrupcions pendents després de moure cadascuna de les paraules
    - Interrupcions molt prioritàries
      - ✓ Per exemple, una interrupció per fallada de pàgina, en la que s'ha d'accedir a disc per portar els operants a memòria

GESTIÓ D'INTERRUPCIONS

Inhibició de les interrupcions

- Abans de saltar a la RSI és necessari inhibir les interrupcions
  - **Motiu:** Si no es prohibeixen la CPU pot entrar en un bucle infinit
    - Quan s'entra a la RSI el perifèric encara no ha desactivat la seva petició
    - Si les interrupcions estan habilitades ⇒ la CPU detecta una interrupció pendent i torna a saltar a la RSI.
    - Abans de finalitzar la RSI s'ha de verificar que el perifèric ha desactivat la línia de petició INTR
      - ✓ **Per programa:** accedint al registre d'estat o de dades del interfície
      - ✓ **Per maquinari:** activant un senyal de reconeixement d'interrupció (INTA)

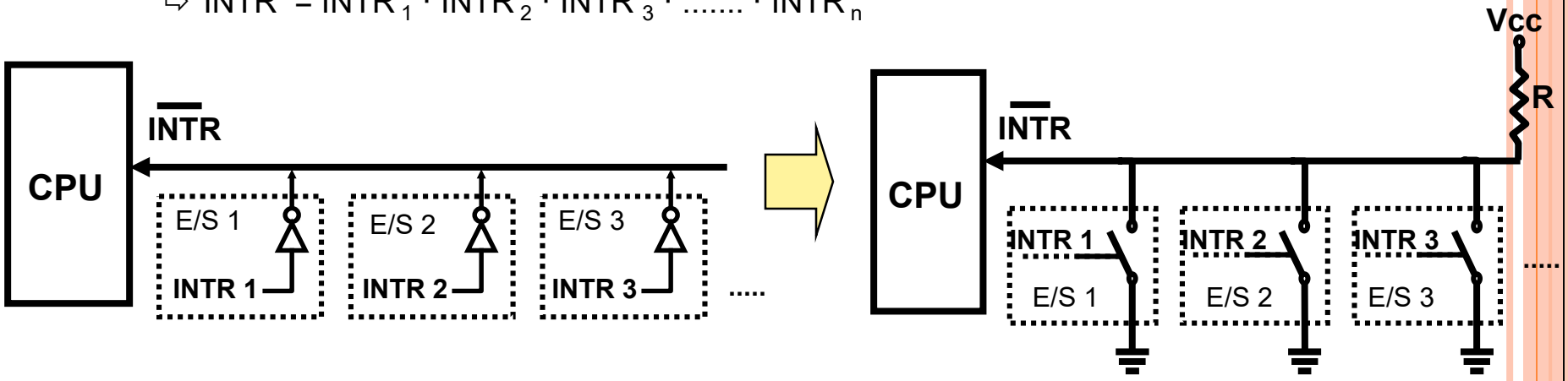


- Alternatives
  - **Inhabilitació global**
    - S'inhabiliten totes les interrupcions ⇒ cap altre perifèric podrà interrompre durant l'execució de la RSI
  - **Inhabilitació o màscares selectives**
    - Quan hi ha varis nivells d'interrupció es poden inhabilitar les interrupcions pel nivell que genera la interrupció, però no necessàriament pel la resta de nivells més prioritaris
    - Interrupcions multi nivell i encadenament d'interrupcions (més endavant)

GESTIÓ D'INTERRUPCIONS

Identificació de la font d'interrupció

- A una mateixa línia d'interrupció és possible connectar varis perifèrics
  - Normalment s'utilitza lògica negada (INTR*) i cablejada (en col·lector obert, *"open collector"*)
    - Sí INTR* = 1 ⇒ No hi ha interrupcions pendents
    - Sí INTR* = 0 ⇒ Sí hi ha interrupcions pendents
  - El senyal INTR* se calcula com la "I" lògica cablejada de cadascuna de les línies de petició d'interrupció individuals:
    - $\overline{INTR} = \overline{INTR}_1 \cdot \overline{INTR}_2 \cdot \overline{INTR}_3 \cdot ..... \cdot \overline{INTR}_n$

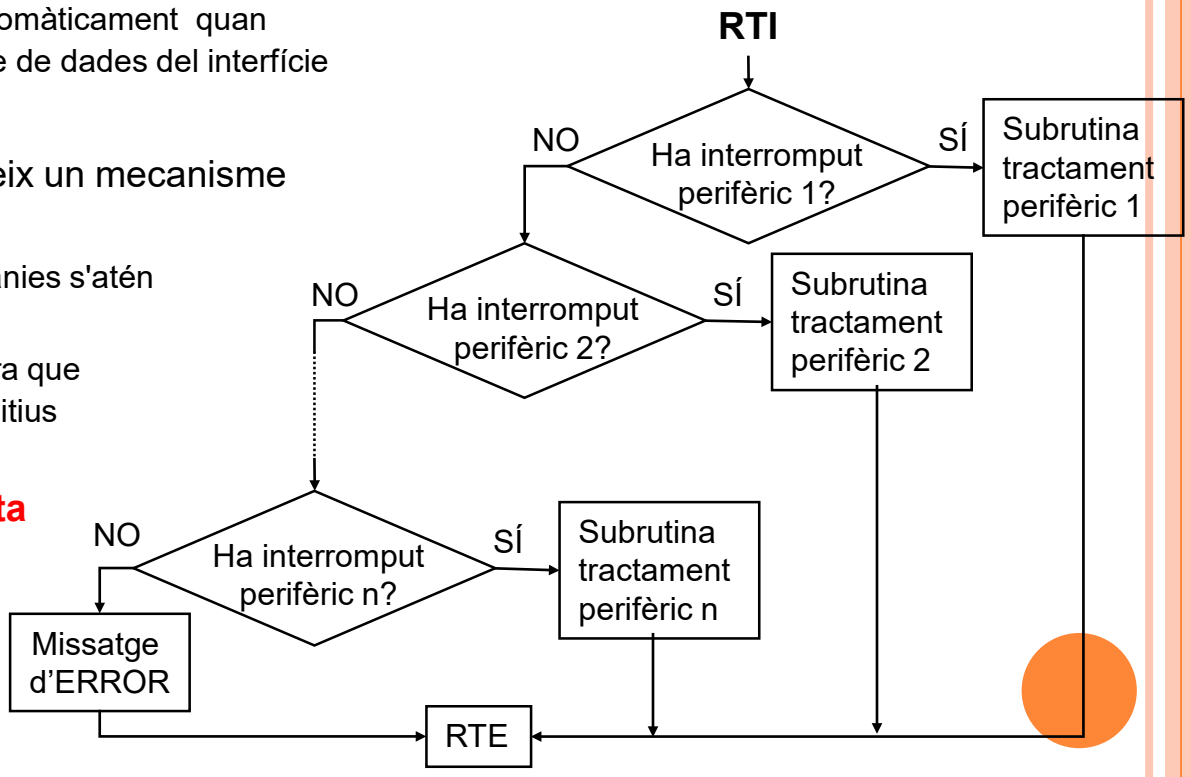


- Quan existeixen varies fonts d'interrupció és necessari un mecanisme per identificar al perifèric que ha generat la interrupció i executar la RSI adequada per atendre el perifèric que l'ha generat
  - **Identificació per programa:** per enquesta (polling)
  - **Identificació per maquinari:** per vectors

GESTIÓ D'INTERRUPCIONS

Identificació per programa (enquesta) (polling)

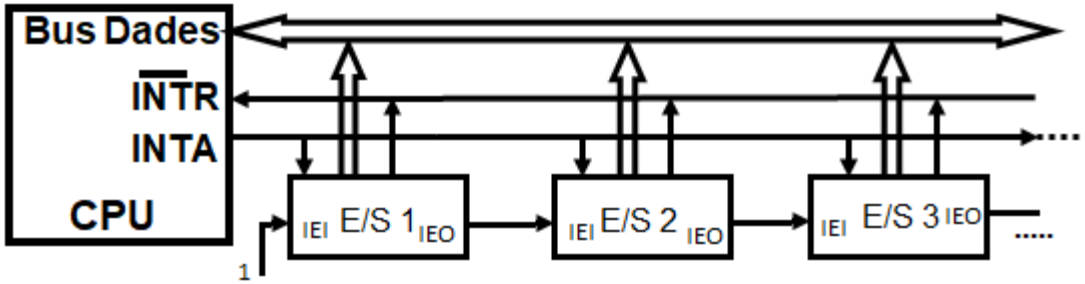
- La RSI examina un a un els bits d'estat de cada perifèric fins que troba el que té activat el seu bit de petició d'interrupció
  - Un cop s'ha detectat el perifèric que ha fet la sol·licitud d'interrupció s'executa una subrutina particular per atendre a aquest perifèric
  - Durant l'execució d'aquesta rutina s'ha de desactivar el bit de petició d'interrupció del perifèric
    - Normalment es desactiva automàticament quan es llegeix o s'escriu al registre de dades del interfície
- **Prioritats**
  - El mètode d'enquesta introdueix un mecanisme de prioritats per programa
    - En el cas de peticions simultànies s'atén per ordre de l'enquesta
    - La RSI es dissenya de manera que es pregunta primer als dispositius més prioritaris
- **Problemes del mètode d'enquesta**
  - Es perd temps consultant els dispositius que no han sol·licitat el servei



GESTIÓ D'INTERRUPCIONS

Identificació maquinari per vectors: *Vectors d'interrupció*

- El perifèric que fa la sol·licitud d'interrupció envia un codi o **numero de vector** a la CPU a partir del qual es pot calcular l'adreça d'inici de la RSI d'aquest perifèric
  - Quan el perifèric rep un senyal de confirmació o reconeixement d'interrupció **INTA** (*"Interruption Ack."*) envia el nº de vector pel bus de dades
  - A partir del **nº de vector** es calcula l'adreça de memòria (vector) on hi ha emmagatzemada l'adreça d'inici de la RSI



- Seqüència d'esdeveniments en el tractament d'una interrupció vectoritzada
  1. El perifèric activa el senyal d'interrupció **(INTR* = 0)** i **(IEO = 0)**
  2. La CPU activa el senyal de confirmació d'interrupció **(INTA=1)** que es propaga a tots els dispositius.
  3. Un perifèric que no ha sol·licitat la interrupció, transfereix el valor de IEI a IEO (Cadena daisy chain)
  4. Quan el perifèric que ha fet la interrupció rep el senyal INTA i l'entrada IEI val 1, volca el seu número de vector sobre el bus de dades, desactiva el senyal de petició d'interrupció i activa IEO (1)
  5. La CPU calcula l'adreça d'inici de la RSI a partir del nº de vector
  6. La CPU salva el context a la pila (PC i registres d'estat) i salta a la RSI
  7. Es guarden els registres accessibles per programa, s'executa l'operació d'E/S i es retorna de la interrupció al programa principal restaurant prèviament tot el context.

GESTIÓ D'INTERRUPCIONS

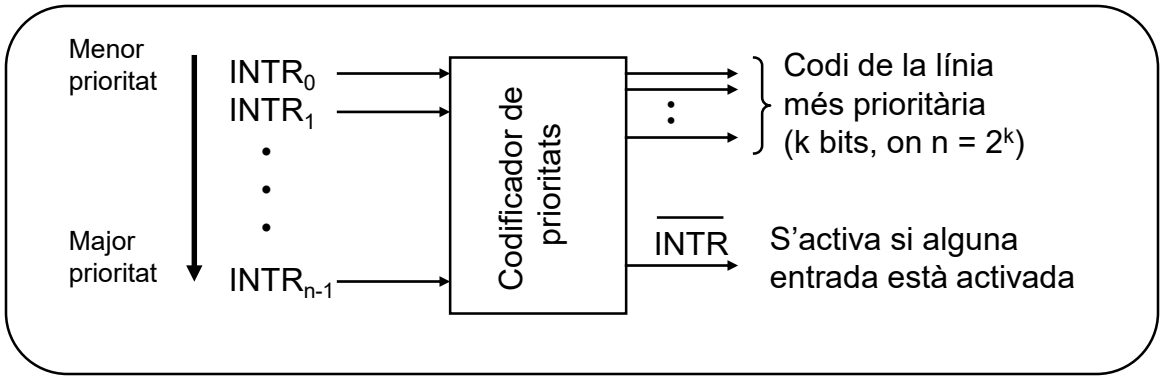
Identificació maquinari per vectors

- **Avantatges**
  - La transmissió de INTA està feta només amb maquinari ⇒ és molt més ràpid que l'enquesta
- **Inconvenients**
  - El nº de dispositius que es poden identificar amb aquest mètode depèn del nº de bits que utilitzem pel el nº vector
    - Exemple: amb un nº de vector de 4 bits es poden identificar 16 dispositius
  - ***Solució:*** es poden utilitzar codis de grup
    - Un mateix nº de vector es pot utilitzar per identificar a un grup de varis dispositius
    - Quan la CPU rep un nº de vector de grup, la RSI ha d'identificar al dispositiu d'aquest grup amb una enquesta

GESTIÓ D'INTERRUPCIONS

Interrupcions multi nivell i encadenament d'interrupcions

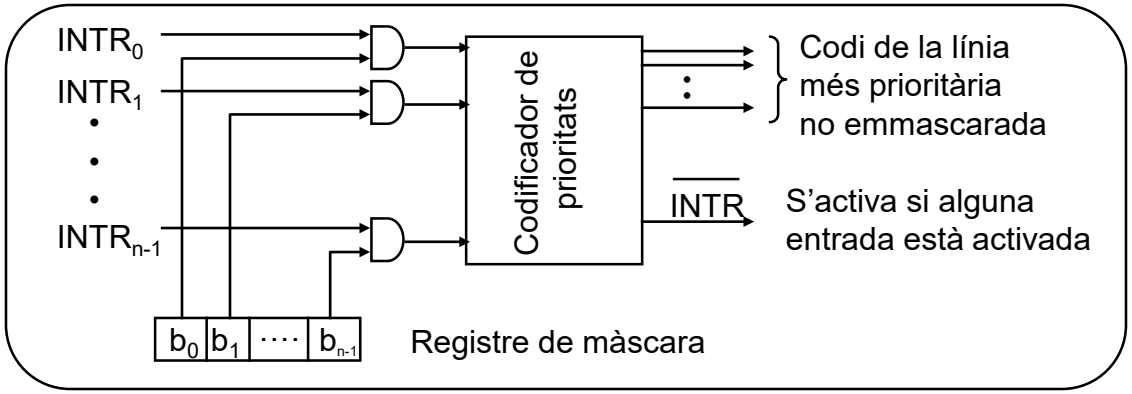
- **Interrupcions multi nivell**
  - Existeixen varies línies o nivells de petició d'interrupció
  - Cada nivell té assignat una prioritat diferent
  - A cada línia d'interrupció s'hi poden connectar un o més dispositius
- **Resolució de conflictes de peticions d'interrupció simultànies**
  - Peticions simultànies per la mateixa línia
    - ⇨ Es resolt amb algun dels mecanismes explicats
      - ✓ Mitjançant l'enquesta (per programa)
      - ✓ Mitjançant vectors (per maquinari)
  - Peticiones simultànies per línies diferents
    - ⇨ Es pot resoldre amb un **codificador de prioritats** ⇒ S'atén a la línia més prioritària



Interrupcions multi nivell i encadenament d'interrupcions

Màscares selectives de nivells d'interrupció

- Els sistemes d'interrupcions multi nivell permeten emmascarar selectivament les interrupcions per determinats nivells
- S'utilitza un ***registre de màscara***
  - 1 bit de màscara b_k per nivell
    - Si $b_k = 1 \rightarrow$ Nivell $INTR_k$ habilitat
    - Si $b_k = 0 \rightarrow$ Nivell $INTR_k$ emmascarat o inhabilitat



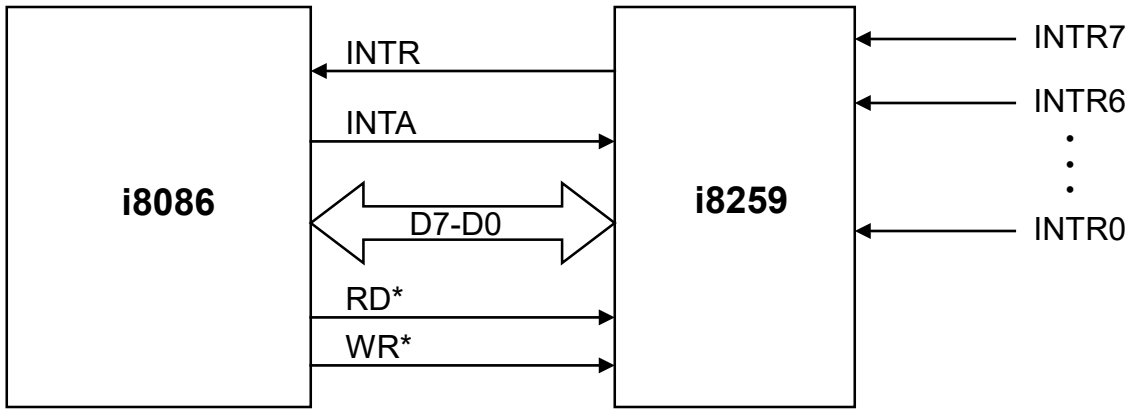
Encadenament d'interrupcions

- En general, en els sistemes d'interrupcions multi nivell es permet l'encadenament d'interrupcions
  - Mentre s'executa la RSI d'un determinat nivell s'inhibeixen les interrupcions pel mateix nivell o inferiors, preo es poden atendre peticions d'interrupció d'una nivell superior
- L'encadenament es controla amb el registre de màscara
  - Quan es produeix una interrupció de prioritat P_k s'emmascaren totes les interrupcions de prioritat $P \leq P_k$

EXEMPLES: I8086

Controlador d'interrupcions i8259

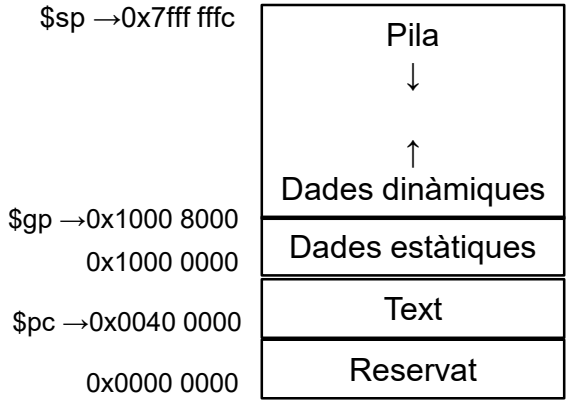
- Permet convertir la línia de petició d'interrupció emmascarable (INTR) en 8 nivells d'interrupció diferents (INTR0-INTR7)
- Cada línia té assignat un nivell de prioritat que pot ser fixa o circular
- El controlador i8259 disposa d'un conjunt de registres per emmagatzemar un nº de vector per cada nivell
  - Aquest registres son programables
  - Quan es produeix una interrupció per un determinat nivell el controlador posa el nº vector assignat al nivell al bus de dades
- El controlador i8259 permet encadenar en cascada dos nivells de controladores
  - Poden haver-hi fins a 64 nivells de prioritat diferents (8x8)



EXEMPLES: MIPS

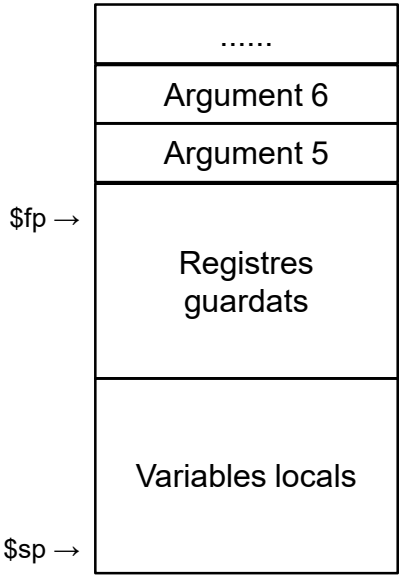
Gestió de les entrades i sortides

- E/S localitzada a memòria
- Dos modes de funcionament: Nucli i usuari
- La meitat superior de l'espai adreçable només és accessible amb el mode nucli:
  - Adreces a partir de 0x80000000
  - Aquesta regió inclou:
    - ✓ Codi pel tractament de les excepcions (RSI)
    - ✓ Dades només accessibles al SO
    - ✓ Adreces d'E/S assignades a partir de : 0xffff0000



Organització de la memòria

sp : Punter de pila
fp : Punter marc



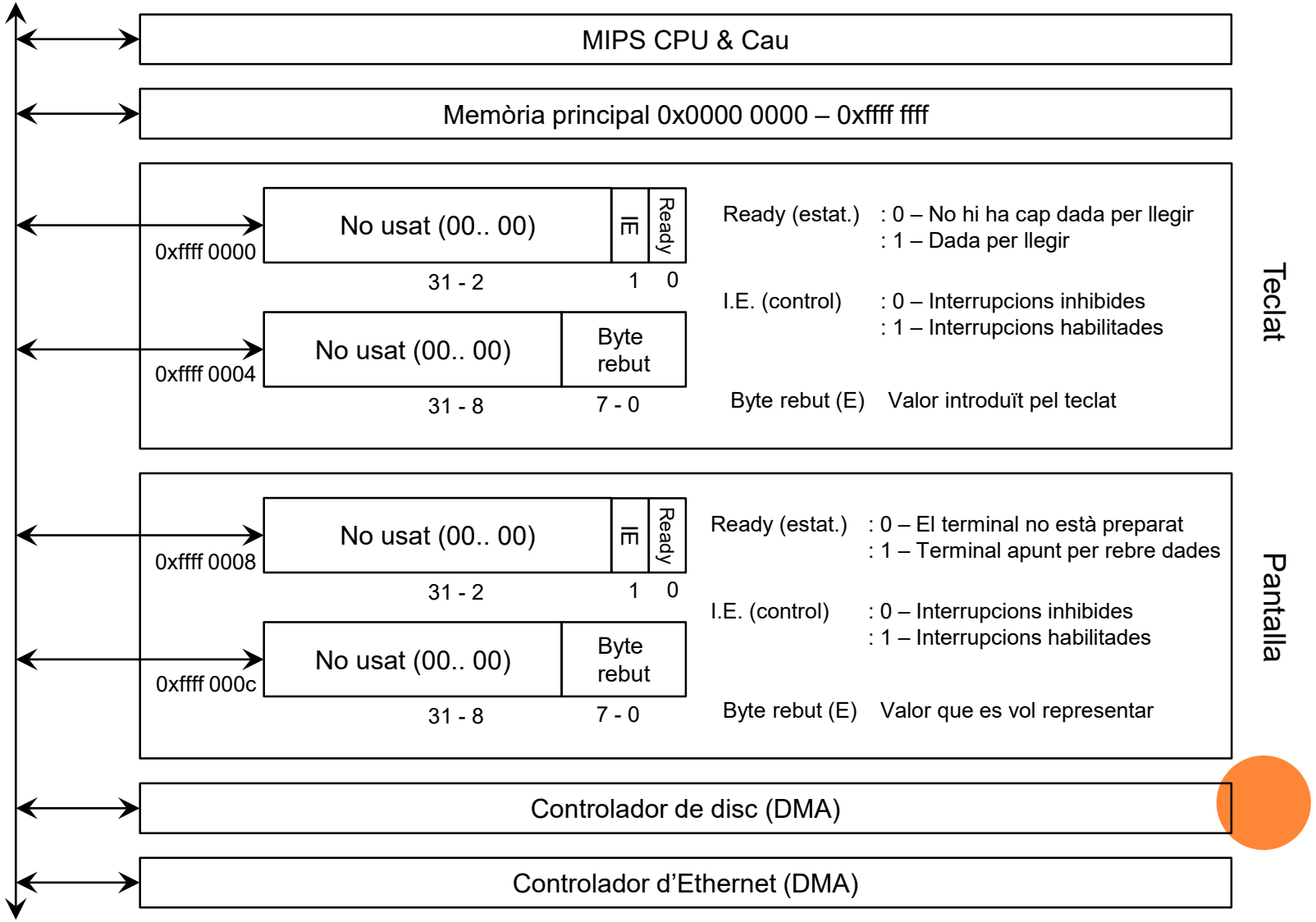
Part alta de la memòria

Part baixa de la memòria

Organització de la pila

EXEMPLES: MIPS

Gestió de les entrades i sortides



EXEMPLES: MIPS

Entrada i sortida amb interrupcions

- Esdeveniments o situacions excepcionals en el flux d'execució del MIPS
  - Petició d'un dispositiu d'E/S: Origen extern (interrupció)
  - Crida al sistema des de l'espai d'usuari: Origen intern (excepció)
  - Desbordament aritmètic: Origen intern (excepció)
  - Us d'una instrucció no definida: Origen intern. (excepció)
  - Problema de maquinari. Origen intern o extern. (excepció o interrupció)
- Pel tractament de les excepcions i interrupcions el MIPS utilitza el coprocessador 0
- El motiu de l'excepció es determina pel registre de causa ($13) del coprocessador 0 (no utilitza interrupcions vectoritzades)
  - Camp de codi d'excepció (ExCod):

    - 0: INT: Interrupció externa
    - 2: TLB_L: Fallada de lectura a TLB (Memòria Virtual)
    - 3: TLB_E: Fallada d'escriptura a TLB
    - 4: ADDRL: Excepció de càrrega d'una dada o instrucció per una adreça no alineada
    - 5: ADDRS: Excepció per intentar guardar una dada en una adreça no alineada
    - 6: IBUS: Excepció per valor del PC erroni (fora de rang)
    - 7: DBUS: Igual que l'anterior, però per dades
    - 10: INV: Codi d'instrucció no vàlid
    - 12: OVF: Desbordament aritmètic
  - Camp IP (interrupcions pendents): Son 8 bits que senyales diferents tipus d'interrupcions pendents. Aquest camp ocupa les mateixes posicions que el camp IM del registre d'estat per poder fer una màscara i atendre les interrupcions pendents. El bit 8 senyala les interrupcions del teclat.

EXEMPLES: MIPS

Entrada i sortida amb interrupcions

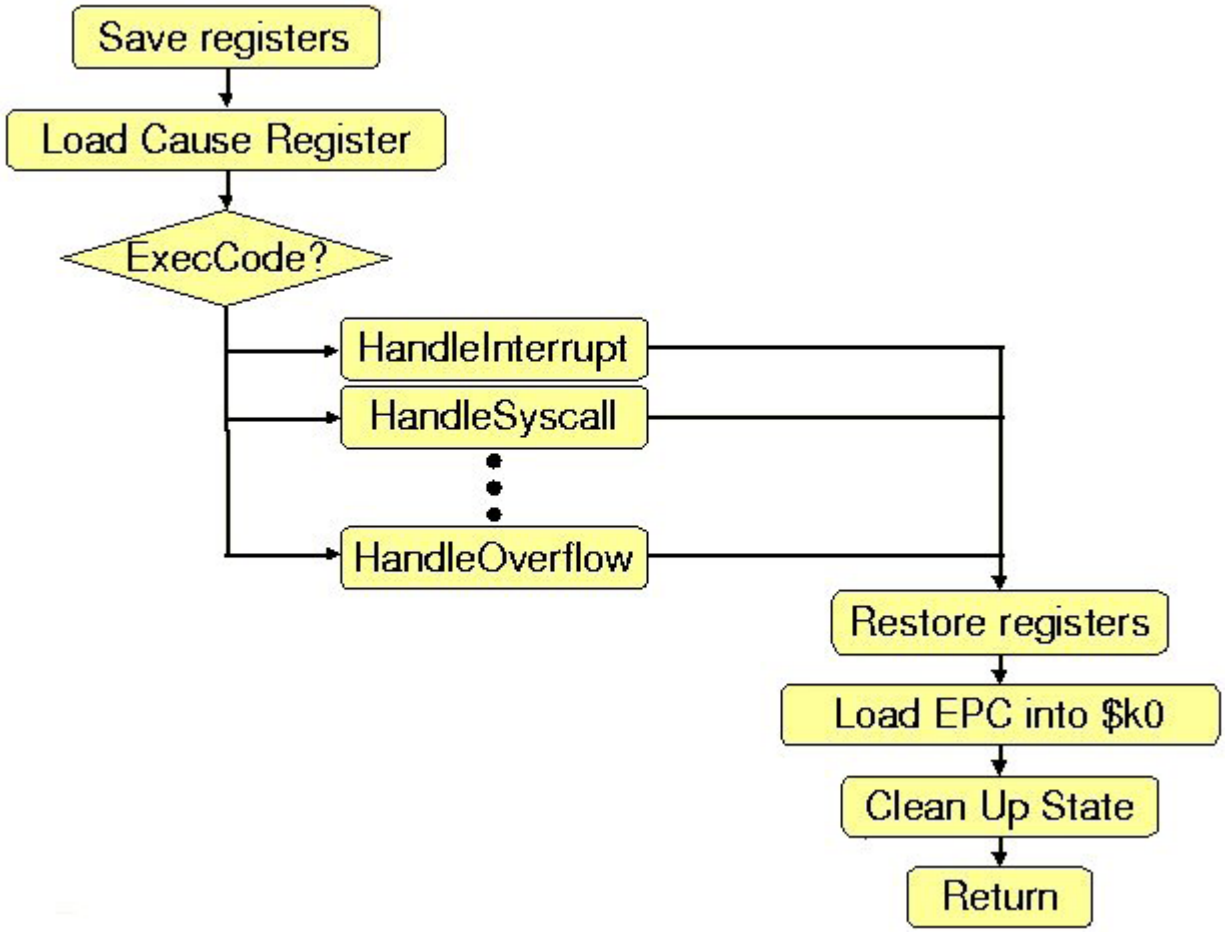
- Registre d'estat ($12) per habilitar o inhabilitar les interrupcions
  - Bits 15-8 (IM: mascara d'interrupció) Mascara de 8 bits que serveix per habilitar selectivament els diferents tipus d'interrupcions disponibles. El bit 8 es correspon al teclat
  - El bit 4 (UM, mode de treball) Amb un 1, la CPU treballa en mode usuari (per executar programes normals). Amb un zero, la CPU treballa en mode nucli (per executar codi del SO)
  - Bit 1 (EXL, Nivell d'excepció) S'activa automàticament al produir-se una excepció o interrupció. S'inhibeixen les interrupcions i salta a mode nucli)
  - Bit 0 (IE, Habilitació d'interrupció) Amb un 1, s'habiliten les interrupcions senyalades pel camp IM. Si s'hi posa un 0, les interrupcions queden inhabilitades.

- El registre EPC ($14) Excepció del Comptador de programa que guarda l'adreça on s'ha de tornar quan s'acabi la RSI
- Per accedir als registres del coprocessador 0 s'ha de fer mitjançant les instruccions:
  - mfc0 d, f que mou els valors dels registres del coprocessador 0 als registres de la CPU
  - mtc0 f, d que mou els valor dels registres de la CPU als registres del coprocessador 0
- La instrucció rfe (retorn d'excepció) habilita les interrupcions (bit IE del registre d'estat) abans de retorna el control després d'executar una RSI

EXEMPLES: MIPS

Entrada i sortida amb interrupcions

- L'arquitectura MIPS fixa una adreça de memòria pel inici del programa que ha de tractar les excepcions.
- L'adreça de memòria és la 8000 0180 h



Programació dels dispositius d'E/S

Interrupcions: Rendiment (consum de CPU amb interrupcions)

- Suposem que en una CPU a 500 MHz la RSI per realitzar una transferència necessita 500 cicles de rellotge (temps d'execució de la RSI). Determineu el % de temps que la CPU gasta realitzant la transferència:
  - Dics dur: transfereix dades en unitats de 16 bytes i té una amplada de banda de 8 MiB/s. No es poden perdre dades. El disc dur, s'utilitza durant un 5 % del temps.
  - Interrupcions de disc dur/s:
  - ✓ 8 MiB/s / 16 bytes per transferència = 524.288 interrupcions/s
  - Cicles/s: 524.288 * 500 = 262.144.000 cicles/s.
  - % CPU (durant la transferència): 262.144.000 / ($500*10^6$) * 100 = 52,4 %
  - % CPU (total: només el 5% de temps actiu): 52,4% * 5 % = 2,62 %

Nota: en l'exemple de l'enquesta suposàvem que estàvem sondejant sempre (no només el 5% del temps): No és molt realista pel disc dur (ja que sabem quan hi ha peticions pendents),
però . . .
per exemple per la targeta de xarxa poden arribar dades en qualsevol moment → hauríem d'estar fent l'enquesta constantment.

Programació dels dispositius d'E/S

Interrupcions: Rendiment (consum de CPU amb interrupcions)

- L'E/S per enquesta o per interrupció resulta inadequada per perifèrics d'alta velocitat, sobre tot si s'ha de transferir una gran quantitat de dades
- **Exemple perifèric lent**
  - Processador a 200 MHz (temps cicle = 5 ns.; Cicle mig per instrucció: CPI = 2 cicles
    - ⇨ Una instrucció tarda en promig 2 x 5 ns = 10 ns ⇒ el computador pot executar ~100 MIPS
  - Volem imprimir un fitxer de 10 KiB en una impressora làser de 20 pagines per minut
  - 1 pàgina ≅ 3.072 caràcters (1 caràcter = 1 byte)
    - ⇨ La impressora imprimeix (20*3.072) = 61.440 Bytes per minut = 1024 Bytes/s = 1 KiB/s

a) E/S per enquesta

- La CPU entra en un bucle i envia un nou byte cada vegada que la impressora està preparada per rebre'l.
  - ⇨ La impressora tarda 10 s en imprimir 10 KiB
  - ⇨ **La CPU està ocupada amb la operació d'E/S durant 10 s**
    (en aquest temps la CPU podria haver executat 1000 milions d'instruccions)

b) E/S per interrupcions

- La impressora genera una interrupció cada vegada que està preparada per rebre un nou byte
  - ⇨ Suposem que la RSI té 10 instruccions (guardar registres, comprovar l'estat, transferir el byte, restaurar registres i retornar)
  - ⇨ Per transferir 10 KiB s'ha d'executar 10.240 vegades la RSI
    ⇒ s'han d'executar 102.400 instruccions per atendre al perifèric ⇒ la CPU tarda 0,001024 s
  - ⇨ **La CPU està ocupada amb la operació d'E/S durant 1,024 ms**

CONCLUSIÓ

- L'E/S per interrupcions redueix en 10.000 vegades el temps que la CPU està ocupada gestionant la impressora

Programació dels dispositius d'E/S

- Una interrupció d'un dispositiu d'E/S és com una excepció (necessita una rutina de tractament de dades, capacitat de recuperació, etc..), però:
  - La interrupció és **asíncrona** respecte al programa (no se sap en quin moment pot produir-se; no està associada a cap tipus d'instrucció concreta).
  - Necessitem saber quin dispositiu l'ha provocat.
- Terminologia (molt variable a la literatura):
  - **Excepció**: una instrucció provoca una situació anormal o no permesa durant l'execució. (exemple: violació d'accés, desbordament, ..).
  - **Interrupció**: un dispositiu extern a la CPU diu a la CPU que requereix la seva atenció.
  - **Trap**: és una instrucció especial que el programador pot utilitzar per provocar una crida a una rutina de servei (crida al sistema operatiu)
- L'enquesta i les interrupcions son idònies per dispositius amb una amplada de banda petita: es redueix el cost de la controladora i del interfície, ja que la CPU i el SO realitzen quasi ve tot el treball:
  - Cada transferència d'una paraula necessita una seqüència d'instruccions de la CPU per fer la transferència (mitjançant una RSI o una enquesta)
- Amb dispositius d'una gran amplada de banda les transferències solen ser de grans blocs de dades
  - La transferència per enquesta o per interrupcions no son bones, ja que estarien ocupant el processador durant massa temps.
- Accés directe a memòria (DMA): Dispositiu per aconseguir que la transferència de dades entre el perifèric i la memòria es realitzi sense la intervenció del processador.

Accés directe a memòria (DMA)

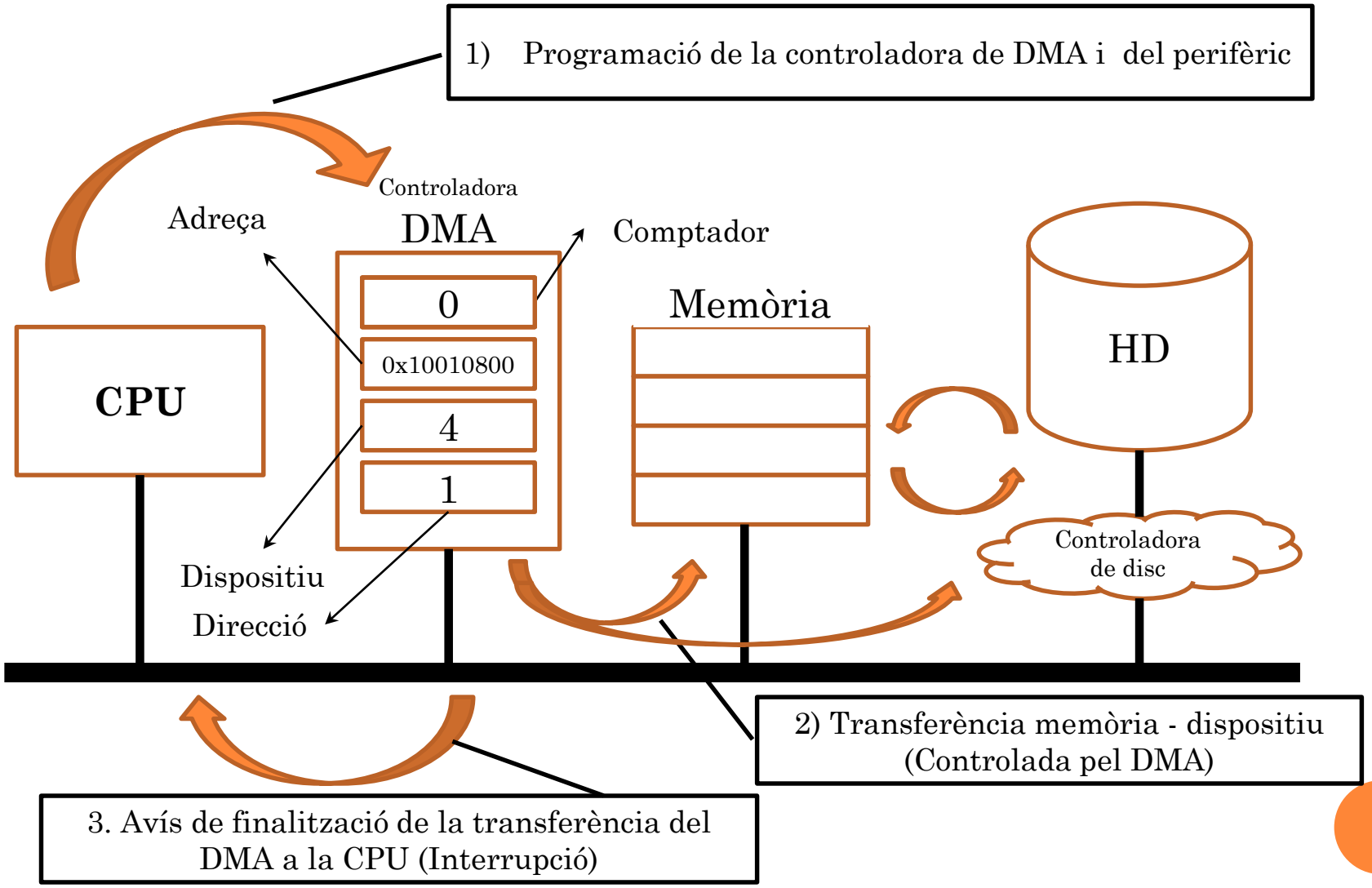
Programació dels dispositius d'E/S

E/S per DMA

- El DMA s'implementa amb un xip especialitzat (controladora de DMA), que transfereix dades entre un dispositiu d'E/S i la memòria de forma autònoma (independent del processador).
- Ha de controlar part del maquinari:
  - Bus: múltiples amos (CPU i el DMA) → sistema d'arbitratge
- Passos en una transferència per DMA:
  - El processador inicialitza (programa) la controladora de DMA:
    - Identitat del dispositiu
    - Operació a realitzar
    - Adreça de memòria on ha de llegir o escriure
    - Número de bytes a transferir
    - Sentit del desplaçament.
  - El dispositiu d'E/S avisa a la controladora de DMA que està preparat.
  - La controladora de DMA demana el bus i, quan l'aconsegueix, realitza tantes operacions de transferència entre el dispositiu i la memòria com siguin necessàries.
  - Per últim, la controladora de DMA genera una interrupció informant al processador de la finalització de la transferència o d'una condició d'error.

PROGRAMACIÓ DELS DISPOSITIUS D'E/S

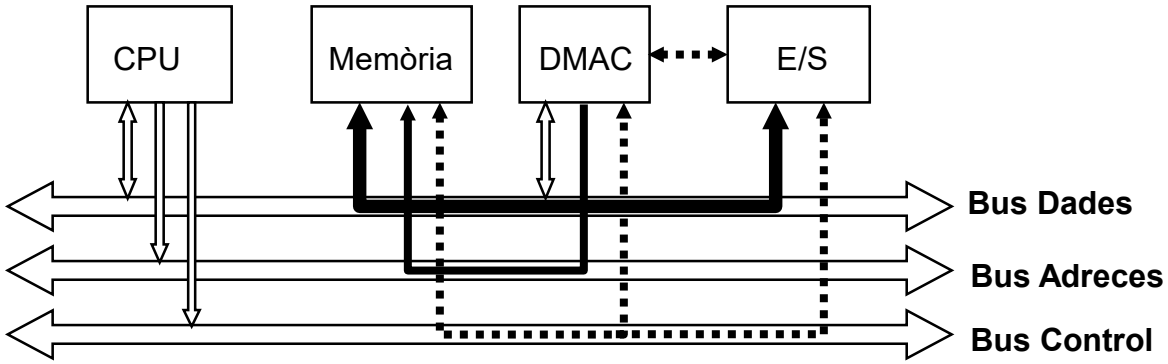
E/S per DMA



Accés directe a memòria (DMA)

El controlador de DMA (DMAC)

- El controlador de DMA és un dispositiu que pot controlar una transferència de dades entre un perifèric i la memòria sense la intervenció de la CPU



- El DMAC ha d'actuar com a máster del bus durant la transferència DMA i ha de ser capaç de:
  - Sol·licitar l'ús del bus mitjançant els senyals i la lògica d'arbitratge necessàries
  - Especificar l'adreça de memòria sobre la que es realitza la transferència
  - Generar els senyals de control del bus
    - Tipus d'operació (lectura/escriptura)
    - Senyals de sincronització de la transferència

TRANSFERÈNCIES DMA

Etapes d'una transferència per DMA

- **Inicialització de la transferència**
  - La CPU envia al interfície del perifèric i al DMAC els paràmetres de la transferència

  Inicialització del interfície (Bus master: CPU - Bus slave: Interfície)
    - Nº de bytes a transferir
    - Tipus de transferència (lectura/escriptura)
    - Més informació de control (pista, sector, etc.)

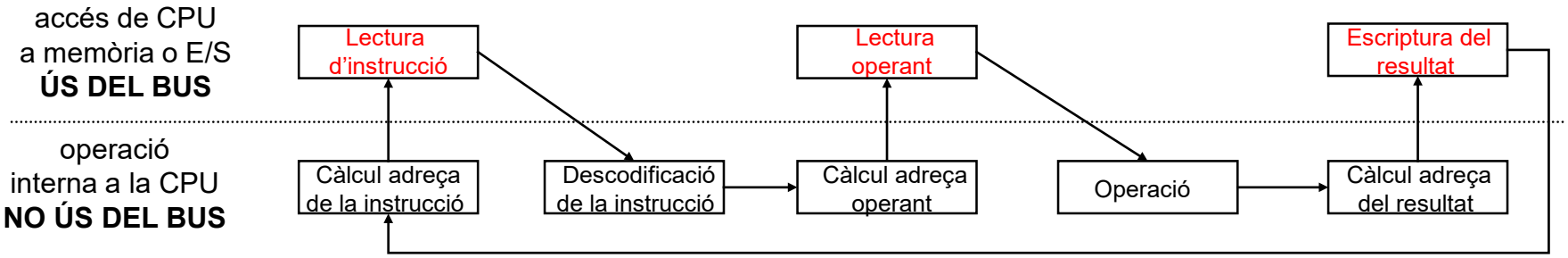
  Inicialització del controlador DMA (Bus master: CPU - Bus slave: DMAC)
    - Nº de bytes o paraules a transferir
    - Tipus de transferència (lectura/escriptura)
    - Adreça de memòria inicial per la transferència
    - Nº de canal (per DMAs amb varis canals)
  - Després de la inicialització la CPU retorna a les seves tasques i no es preocupa més de l'evolució de la transferència
- **Realització de la transferència**
  - Quan el perifèric està apunt per realitzar la transferència ho diu al DMAC
  - El DMAC demana el control del bus i es realitza la transferència entre el perifèric i la memòria
    - **Bus máster:** DMAC + Perifèric - **Bus slave:** Memòria
    - Després de la transferència de cada paraula s'actualitzen els registres del DMA
      - ✓ Nº de bytes o paraules a transferir
      - ✓ Adreça de memòria
- **Finalització de la transferència**
  - El DMAC allibera el bus i retorna el control a la CPU
  - El DMAC sol activar un senyal d'interrupció per indicar a la CPU la finalització de l'operació d'E/S sol·licitada

Transferències DMA

Tipus de transferències

⌦ Problema que pot presentar el DMA

- Pot degradar el rendiment de la CPU si el DMAC fa un ús intensiu del bus
  - ⇨ Si el bus està ocupat en una transferència DMA, la CPU no pot accedir a memòria per llegir instruccions/dades
- Aquest problema es redueix amb l'ús de la memòria cau
  - ⇨ La major part del temps, la CPU llegeix instruccions de la cau, pel que no necessita usar el bus de memòria
  - ⇨ El DMAC pot aprofitar aquest intervals en els que la CPU està llegint instruccions de la cau (i per tant no usa el bus de memòria) per realitzar les transferències
- En el cas de computadores sense cau
  - ⇨ El processador no utilitza el bus en totes les fases de l'execució d'una instrucció
  - ⇨ El DMAC pot aprofitar les fases d'execució d'una instrucció en les que la CPU no utilitza el bus per realitzar les seves transferències



⌦ Conclusió

- Si el DMAC només agafa el control del bus durant els intervals de temps en els que la CPU no l'utilitza
  ⇒ *el rendiment del sistema no tindrà cap degradació*
- Es distingeixen, dos tipus de transferències ⇒
  - → **Por ràfegues (burst)**
  - → **Por robatori de cicle (cycle-stealing)**

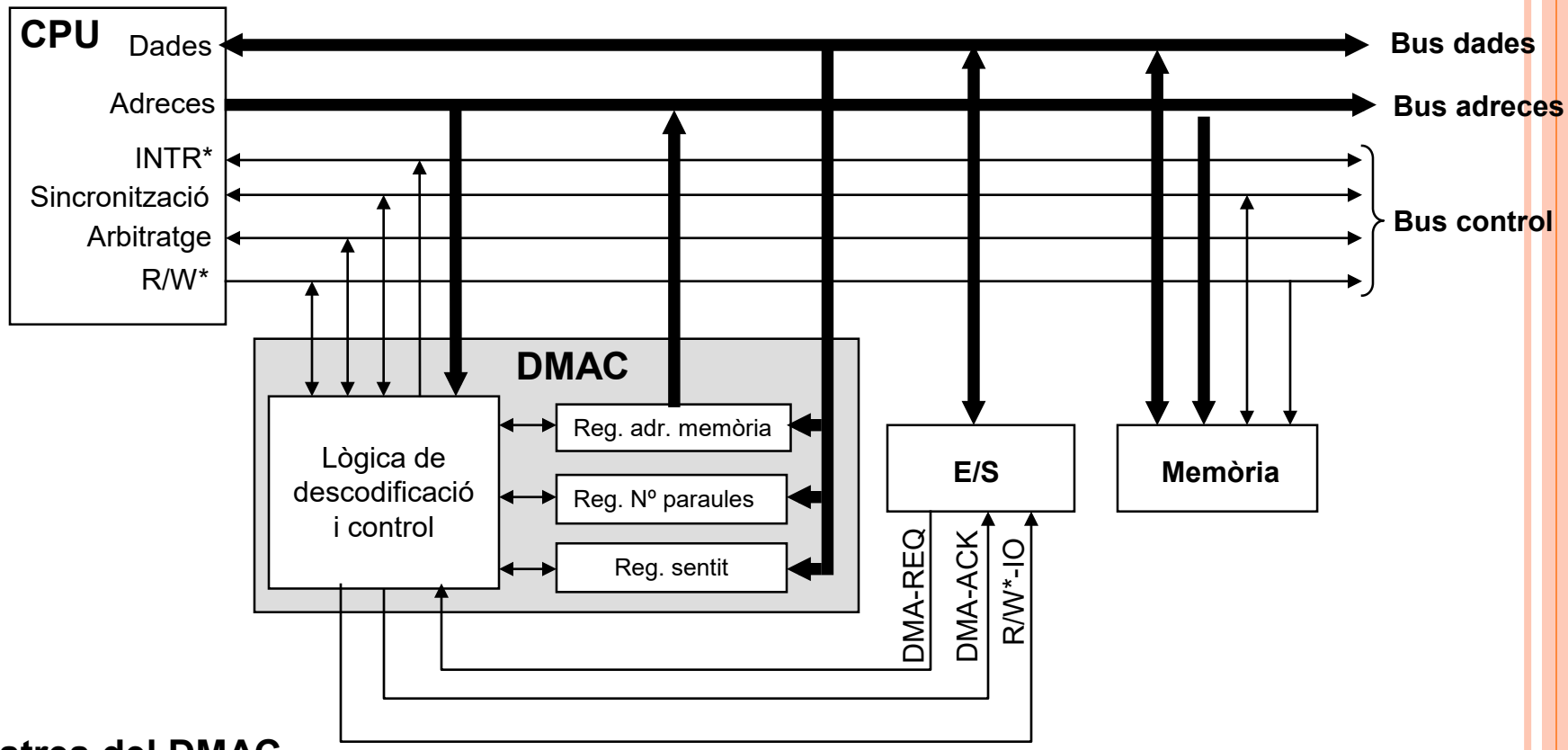
Transferències DMA

Tipus de transferències (cont.)

- **Transferència DMA mode ràfega**
  - El DMAC sol·licita el control del bus a la CPU
  - Quan la CPU concedeix el bus el DMAC no l'allibera fins que no ha acabat la transferència de tot el bloc de dades
  - **AVANTATGES:**
    - La transferència es realitza de forma ràpida
  - **INCONVENIENTS:**
    - Durant el temps que dura la transferència la CPU no pot utilitzar el bus per llegir la memòria, fet que pot degradar el rendiment del sistema
- **Transferència DMA mode robatori de cicle**
  - El DMAC sol·licita el control del bus a la CPU
  - Quan la CPU concedeix el bus al DMAC, es realitza la transferència d'una única paraula i després el DMAC allibera el bus
  - El DMAC torna a sol·licitar el control del bus tantes vegades com sigui necessari fins que finalitza la transferència de tot el bloc
  - La CPU cedeix el control del bus durant els cicles que no l'utilitza
  - **AVANTATGES:**
    - No es degrada el rendiment del sistema
  - **INCONVENIENTS:**
    - La transferència tarda més temps

El controlador DMA (DMAC)

Estructura del maquinari d'un DMAC



Registres del DMAC

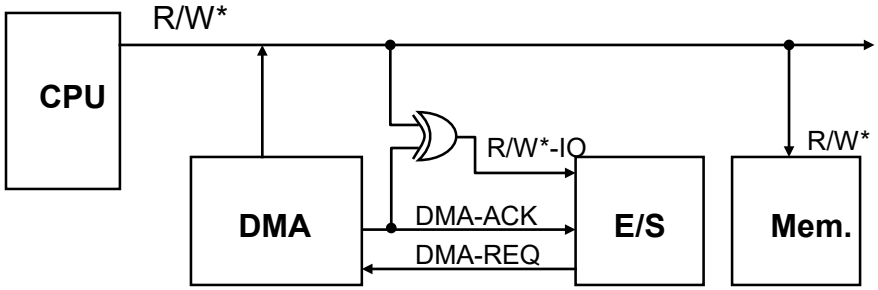
- **Reg. adr. memòria:** emmagatzema l'adreça inicial de memòria i s'incrementa/decrementa després de transferir cada paraula
- **Reg. Nº paraules:** emmagatzema el número de paraules a transferir i es decrementa després de transferir cada paraula
- **Reg. sentit:** emmagatzema el sentit de la transferència (lectura o escriptura)

El controlador DMA (DMAC)

Senyals de control del perifèric

- **DMA-REQ:** sol·licitud del servei de DMA
  - L'activa el perifèric per indicar al DMAC que està apunt per transmetre/rebre
- **DMA-ACK:** Concessió del servei de DMA
  - L'activa el DMAC per indicar al perifèric que pot realitzar la transferència
  - Abans d'activar aquest senyal el DMAC ha de tenir el control del bus
- **R/W*-IO:** Sentit de la transferència pel perifèric
  - El perifèric no pot utilitzar el mateix senyal de R/W* que la memòria, ja que la transferència té sentits oposats des del punt de vista d'un i l'altre dispositiu
  - **Operació DMA de lectura (memòria → perifèric)**
    - Per la memòria és una lectura: R/W* = 1
    - Pel perifèric és una escriptura: R/W*-IO = 0
  - **Operació DMA d'escriptura (perifèric → memòria)**
    - Per la memòria és una escriptura: R/W* = 0
    - Pel perifèric és una lectura: R/W*-IO = 1

Possible esquema de connexió de R/W* y R/W*-IO



DMA-ACK	R/W*	R/W*-IO	
0	0	0	Operació d'E/S normal (controlada per la CPU) R/W*-IO = R/W*
0	1	1	
1	0	1	Operació d'E/S controlada per DMA R/W*-IO = NOT (R/W*)
1	1	0	

El controlador DMA (DMAC)

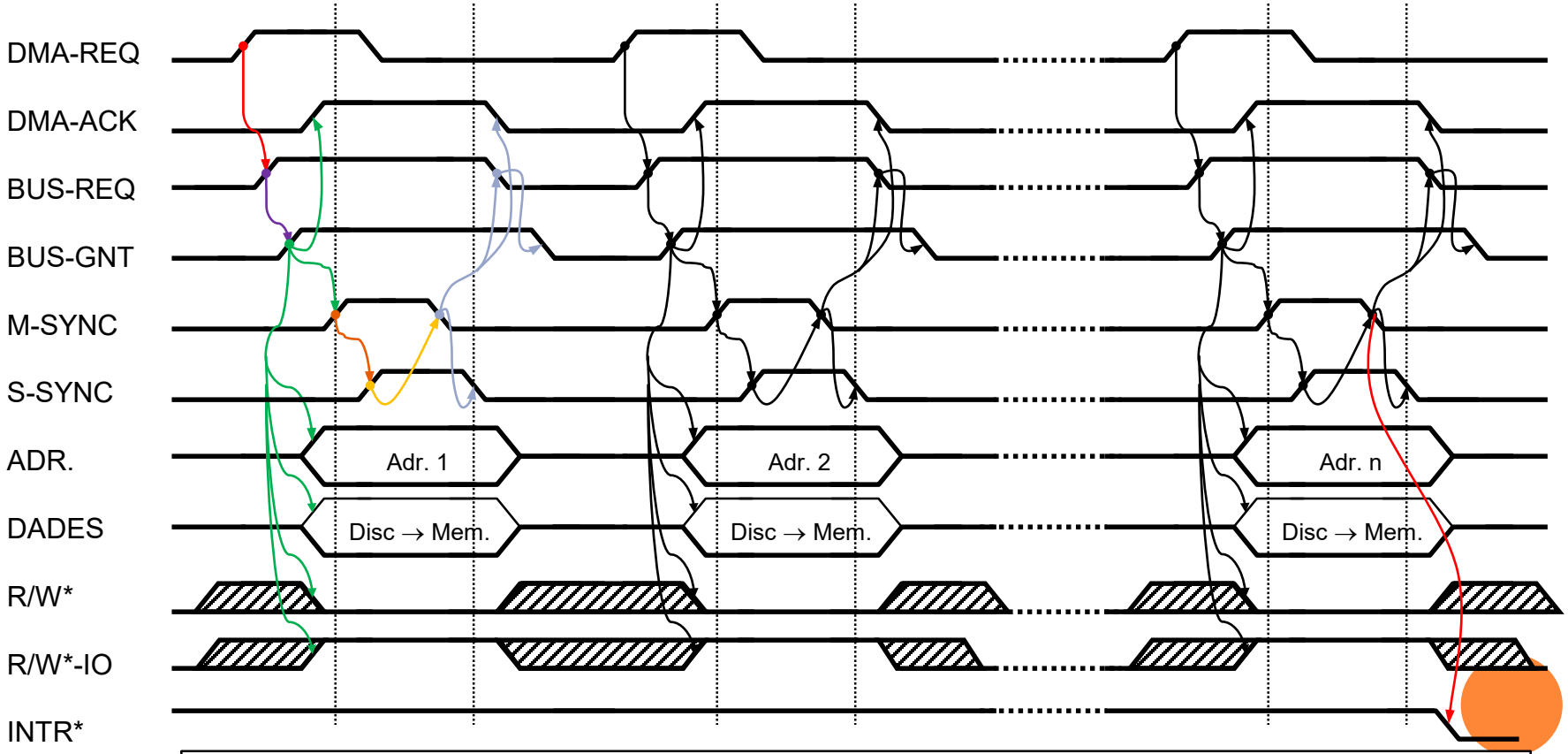
Funcionament d'una transferència per DMA

- **Fase d'inicialització**
  - La CPU indica al interfície del perifèric l'operació a realitzar
  - La CPU accedeix als registres del DMAC per indicar els paràmetres de la transferència
    - Adreça inicial de memòria → *Reg. adr. memòria*
    - Número de paraules a transferir → *Reg. Nº paraules*
    - Sentit de la transferència → *Reg. Sentit*
  - La CPU retorna a les seves tasques
- **Fase de transferència**
  - Quan el perifèric està apunt per fer la transferència activa el senyal ***DMA-REQ*** per comunicar-ho al DMAC
  - El DMAC sol·licita el control del bus mitjançant les línies d'arbitratge (**BUS-REQ**)
  - El DMAC rep la concessió del bus (**BUS_GNT**) i activa el senyal **DMA-ACK** per indicar al perifèric que pot iniciar la transferència.
  - El DMAC ha de generar i processar els senyals del bus adequades
    - Adreça de memòria sobre la que es realitza la transferència
    - Senyals de sincronització de la transferència (master-syncro, slave-syncro, etc.)
    - Senyals de lectura/escriptura (R/W* y R/W*-IO)
  - Després de transferir cada paraula el DMAC ha d'actualitzar els seus registres
    - Decrementar el registre del nº de paraules
    - Incrementar/decrementar el registre d'adreces de memòria (poden ser creixents o decreixents)
  - En les transferències de robatori de cicle
    - S'allibera el bus després de transferir cada paraula i es torna a sol·licitar per transferir la següent
  - En les transferències en mode ràfega
    - El DMAC manté el control del bus fins que s'ha transferit el bloc complert
- **Fase de finalització de la transferència**
  - Quan el registre del nº de paraules arriba a zero el DMAC activa el senyal d'interrupció per notificar-ho a la CPU

El controlador DMA (DMAC)

Exemple d'una transferència per DMA

- Transferència en mode robatori de cicle de disc a memòria (llegir disc – escriure a memòria)
- Protocol de bus asíncron i protocol d'arbitratge de dos fils



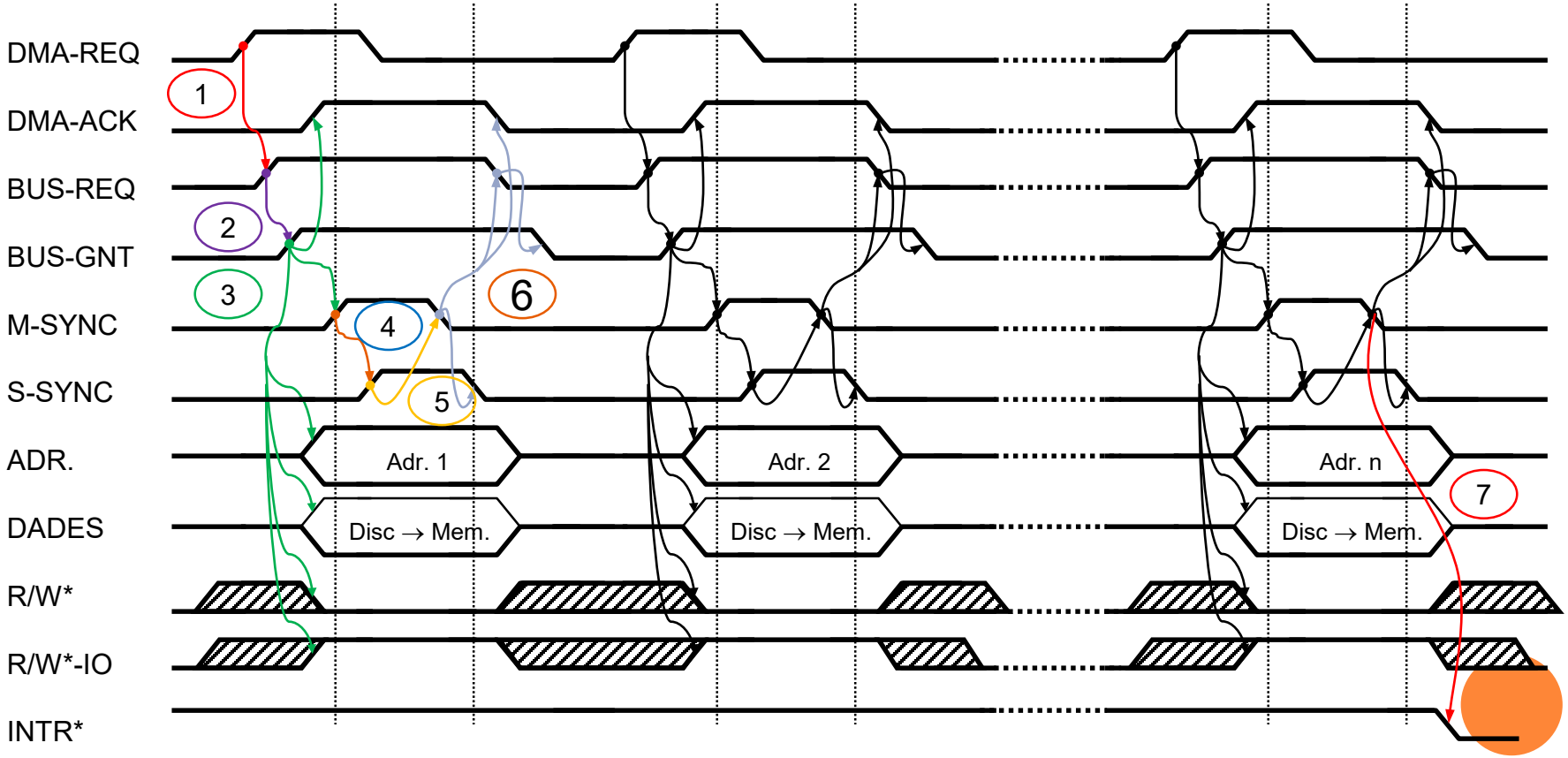
Les altres transferències....

L'última genera una interrupció

El controlador DMA (DMAC)

Exemple d'una transferència per DMA

- Transferència en mode robatori de cicle de disc a memòria (llegir disc – escriure a memòria)
- Protocol de bus asíncron i protocol d'arbitratge de dos fils



El controlador DMA (DMAC)

Exemple d'una transferència per DMA

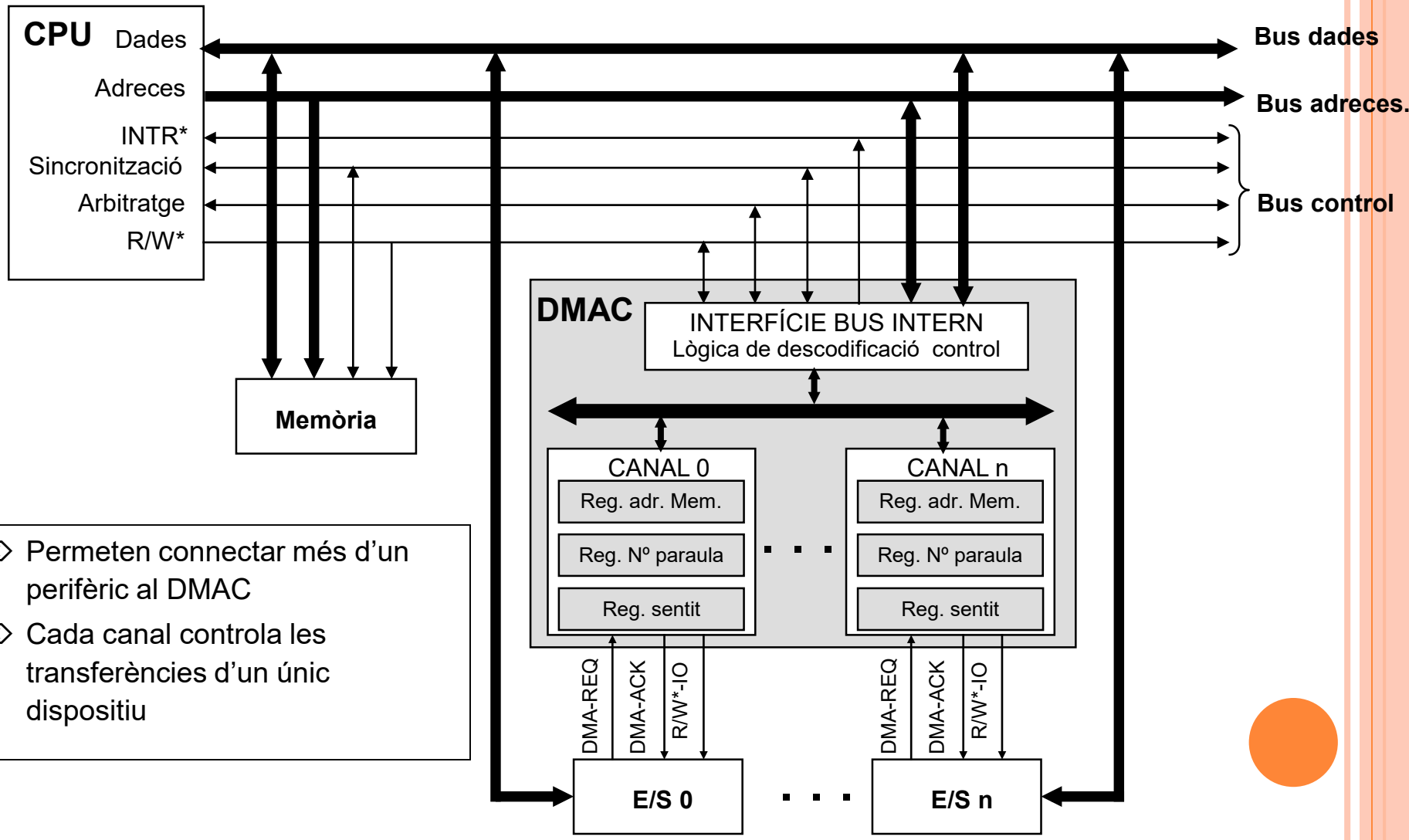
1. El perifèric diu al DMAC que està preparat
2. El DMAC demana a la CPU el control del bus
3. La CPU concedeix el control del bus i elDMAC agafa el control del bus i posa l'adreça al bus d'adreces, diu al perifèric que pot deixar la paraula al bus de dades,lectura del perifèric, escriptura a memòria, ..
4. S'activa el Bus Master per llegir la paraula que hi ha al bus
5. S'activa el Bus Slave per gravar a memòria la paraula del bus
6. Finalitza la transferència de la paraula, desactivant totes les línies

Les altres transferències....

7. L'última genera una interrupció

El controlador DMA (DMAC)

Controlador de DMA de varis canals



- Permeten connectar més d'un perifèric al DMAC
- Cada canal controla les transferències d'un únic dispositiu

Programació dels dispositius d'E/S

E/S per DMA

- Suposem la mateix CPU i disc dur dels exemples anteriors.
- Suposem que el procés d'inici de la controladora de DMA (programació dels ports corresponents) tarda 1.000 cicles i que la RSI de tractament un cop ha finalitzat la transferència tarda 500 cicles.
- Suposem que cada transferència de DMA és de 4KB.
- Casos:
  - Si el disc està contínuament transmeten:
    - 8MB/s / 4KB per transferència = 2048 transferències de DMA per segon.
    - Cicles CPU/s: (1000+500) * 2048 = 3.072.000 cicles/s
    - % de CPU (durant les transferències): $3.072.000/500 \cdot 10^6 = 0{,}6144\%$
  - Com que només transfereix el 5 % del temps:
    - % CPU (total): 0,61 % * 5 % = 0,031 % (molt petit)

Nota: a l'exemple se suposa que el DMA no interfereix amb la CPU, cosa que no és molt realista, ja que els accessos a memòria de la CPU poden xocar amb els del DMA.