Tema 8. Excepcions i interrupcions

Joan Manuel Parcerisa

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

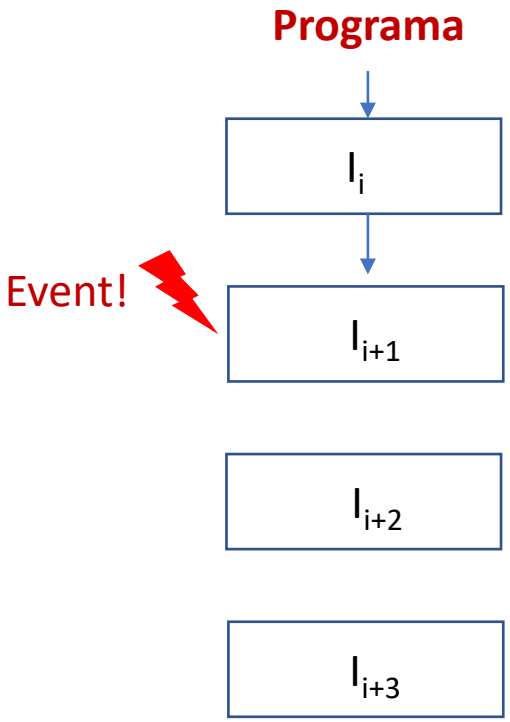
FIB

Excepcions i Interrupcions

- Introducció
  - Tipologia d'interrupcions i excepcions
  - Cronologia d'una excepció
- Implementació en MIPS
- Tres exemples concrets

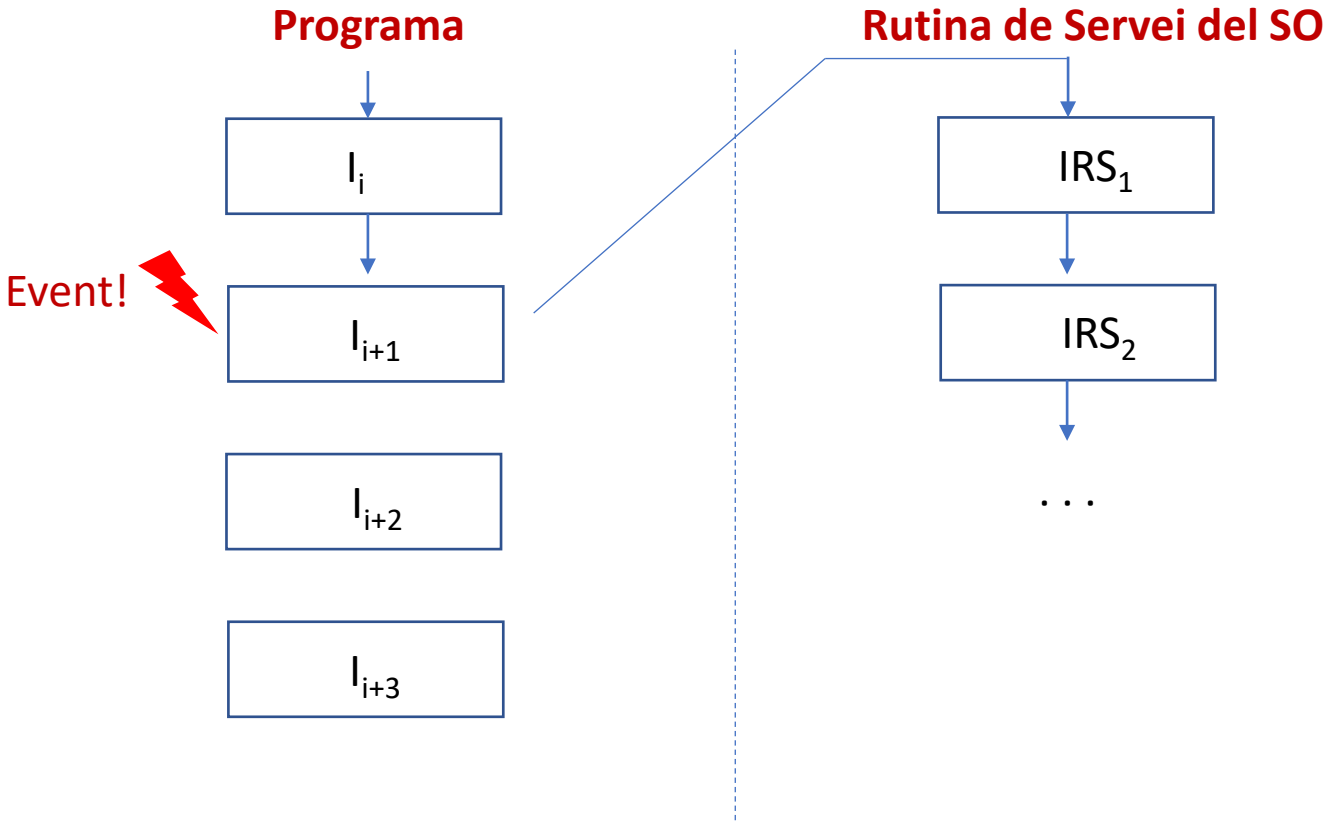
Concepte

- **Interrupcions**: mecanisme del processador que altera el flux de control normal del programa sense intervenir cap instrucció de salt
  - Degut a un *event* inesperat o rar (extern o intern)



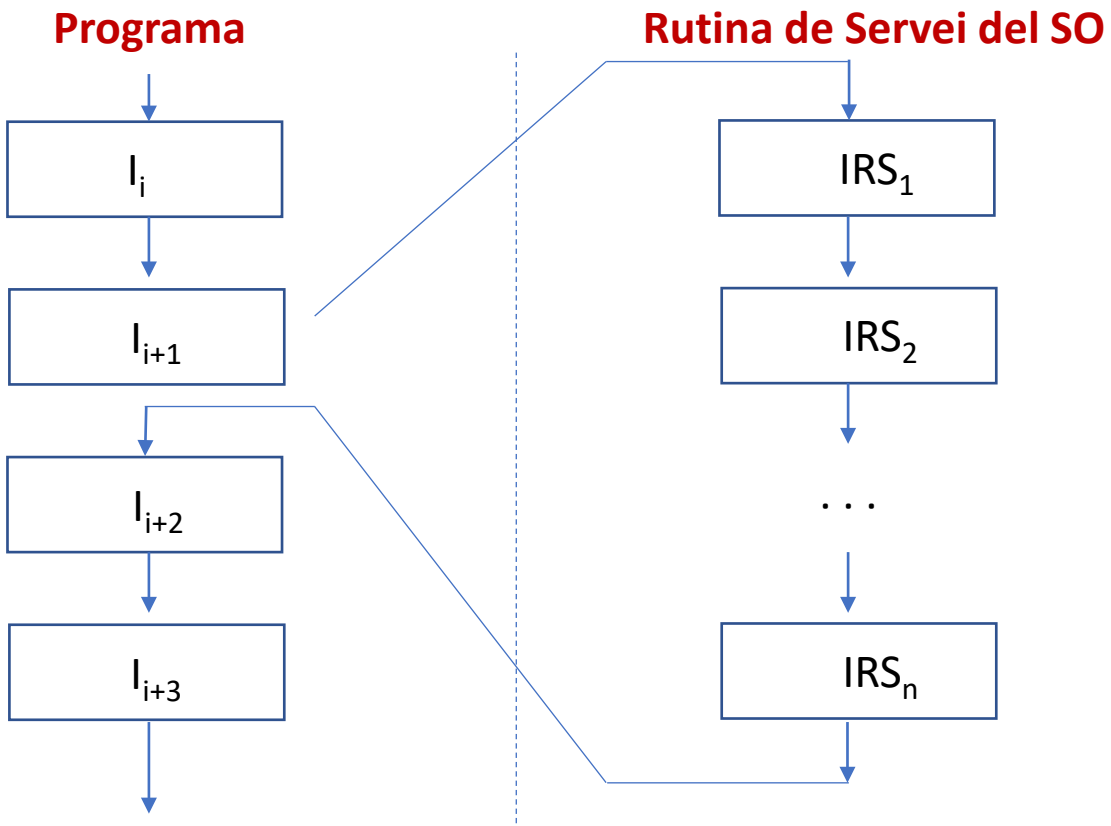
Concepte

- **Interrupcions**: mecanisme del processador que altera el flux de control normal del programa sense intervenir cap instrucció de salt
  - Degut a un *event* inesperat o rar (extern o intern)
  - Necessita ser tractat per una Rutina de Servei del Sistema Operatiu



Concepte

- **Interrupcions**: mecanisme del processador que altera el flux de control normal del programa sense intervenir cap instrucció de salt
  - Degut a un *event* inesperat o rar (extern o intern)
  - Necessita ser tractat per una Rutina de Servei del Sistema Operatiu



Tipologia d'interrupcions i excepcions

- ***Excepcions*** (o *interrupcions síncrones*)
  - Excepcions per violació d'una restricció
    - Instrucció privilegiada, opcode invàlid
    - Overflow aritmètic, divisió per zero, adreça no-alineada

Tipologia d'interrupcions i excepcions

- ***Excepcions*** (o *interrupcions síncrones*)
  - Excepcions per violació d'una restricció
    - Instrucció privilegiada, opcode invàlid
    - Overflow aritmètic, divisió per zero, adreça no-alineada
  - Fallades de memòria virtual
    - Fallada de pàgina, fallada de TLB, violació de proteccions

Tipologia d'interrupcions i excepcions

- ***Excepcions*** (o *interrupcions síncrones*)
  - Excepcions per violació d'una restricció
    - Instrucció privilegiada, opcode invàlid
    - Overflow aritmètic, divisió per zero, adreça no-alineada
  - Fallades de memòria virtual
    - Fallada de pàgina, fallada de TLB, violació de proteccions
  - Causades expressament per una instrucció
    - Crides al sistema (invocades amb *syscall*)
    - Traps: breakpoint, debug

Tipologia d'interrupcions i excepcions

- ***Excepcions*** (o *interrupcions síncrones*)
  - Excepcions per violació d'una restricció
    - Instrucció privilegiada, opcode invàlid
    - Overflow aritmètic, divisió per zero, adreça no-alineada
  - Fallades de memòria virtual
    - Fallada de pàgina, fallada de TLB, violació de proteccions
  - Causades expressament per una instrucció
    - Crides al sistema (invocades amb *syscall*)
    - Traps: breakpoint, debug
- **Interrupcions** (o *interrupcions asíncrones*)
  - Degudes a events *externs* al processador
    - Petició de servei d'un dispositiu d'E/S
    - Expiració del temporitzador
    - Anomalia del hardware, tall d'energia

Cronologia d'una excepció

1. Es detecta una excepció o interrupció
   - Al decodificador: opcode il·legal, instruc. privilegiada, syscall
   - A la MMU: fallades de memòria virtual
   - A la ALU: adreça no-alineada, overflow, divisió per zero
   - Un dispositiu d'E/S ha activat el senyal extern INT: interrupció

Cronologia d'una excepció

1. Es detecta una excepció o interrupció
   - Al decodificador: opcode il·legal, instruc. privilegiada, syscall
   - A la MMU: fallades de memòria virtual
   - A la ALU: adreça no-alineada, overflow, divisió per zero
   - Un dispositiu d'E/S ha activat el senyal extern INT: interrupció
2. La Unitat de Control de la CPU decideix...
   - Si és una excepció: avorta la instrucció en curs, inhibint l'escriptura en registres, memòria, etc.
   - En cas d'interrupció: espera que finalitzi la instrucció en curs

Cronologia d'una excepció

1. Es detecta una excepció o interrupció
   - Al decodificador: opcode il·legal, instruc. privilegiada, syscall
   - A la MMU: fallades de memòria virtual
   - A la ALU: adreça no-alineada, overflow, divisió per zero
   - Un dispositiu d'E/S ha activat el senyal extern INT: interrupció
2. La Unitat de Control de la CPU decideix...
   - Si és una excepció: avorta la instrucció en curs, inhibint l'escriptura en registres, memòria, etc.
   - En cas d'interrupció: espera que finalitzi la instrucció en curs
3. A continuació, salta a la RSE
   - És la *Rutina de Servei d'Excepcions* o *Exception Handler*
   - És el codi del SO que fa el tractament d'excepcions i interrupcions
   - Com se salta? → Posant al PC l'adreça inicial de la RSE

Cronologia d'una excepció

1. Es detecta una excepció o interrupció
   - Al decodificador: opcode il·legal, instruc. privilegiada, syscall
   - A la MMU: fallades de memòria virtual
   - A la ALU: adreça no-alineada, overflow, divisió per zero
   - Un dispositiu d'E/S ha activat el senyal extern INT: interrupció
2. La Unitat de Control de la CPU decideix...
   - Si és una excepció: avorta la instrucció en curs, inhibint l'escriptura en registres, memòria, etc.
   - En cas d'interrupció: espera que finalitzi la instrucció en curs
3. A continuació, salta a la RSE
   - És la *Rutina de Servei d'Excepcions* o *Exception Handler*
   - És el codi del SO que fa el tractament d'excepcions i interrupcions
   - Com se salta? → Posant al PC l'adreça inicial de la RSE
4. I un cop acaba la RSE
   - Pot retornar al programa o avortar-lo, segons el tipus d'excepció

Funcions de la RSE: violació de restriccions

- Excepcions per violació d'una restricció

Funcions de la RSE: violació de restriccions

- Excepcions per violació d'una restricció
- Normalment no hi ha tractament
  - La RSE *avorta el programa*

Funcions de la RSE: violació de restriccions

- Excepcions per violació d'una restricció
- Normalment no hi ha tractament
  - La RSE *avorta el programa*

- En alguns casos la RSE pot resoldre el problema
  - Per exemple, si el SO emula per software la instrucció

Funcions de la RSE: violació de restriccions

- Excepcions per violació d'una restricció
- Normalment no hi ha tractament
  - La RSE *avorta el programa*
- En alguns casos la RSE pot resoldre el problema
  - Per exemple, si el SO emula per software la instrucció
- Un cop finalitza la RSE
  - *Se salta a la instrucció següent* a la que ha causat l'excepció

Funcions de la RSE: memòria virtual

- Excepcions de fallada de pàgina o de TLB

Funcions de la RSE: memòria virtual

- Excepcions de fallada de pàgina o de TLB

- Fallada de TLB
  - Causa excepció en MIPS (altres ho resolen per hw)
  - La rutina de servei copia l'entrada de la TP al TLB

Funcions de la RSE: memòria virtual

- Excepcions de fallada de pàgina o de TLB

- Fallada de TLB
  - Causa excepció en MIPS (altres ho resolen per hw)
  - La rutina de servei copia l'entrada de la TP al TLB

- Fallada de pàgina
  - Causa excepció en tots els processadors
  - La RSE copia la pàgina del disc a un marc lliure en MP, i actualitza la TP i el TLB

Funcions de la RSE: memòria virtual

- Excepcions de fallada de pàgina o de TLB

- Fallada de TLB
  - Causa excepció en MIPS (altres ho resolen per hw)
  - La rutina de servei copia l'entrada de la TP al TLB

- Fallada de pàgina
  - Causa excepció en tots els processadors
  - La RSE copia la pàgina del disc a un marc lliure en MP, i actualitza la TP i el TLB

- Un cop finalitza la rutina de servei
  - E*s reexecuta la instrucció* que ha causat l'excepció

Funcions de la RSE: crides al sistema

- Excepció de crida al sistema i trap (p.ex. *syscall*)

Funcions de la RSE: crides al sistema

- Excepció de crida al sistema i trap (p.ex. *syscall*)

- És similar a una crida a subrutina
  - Amb pas de paràmetres i retorn de resultats
  - La RSE executa el servei sol·licitat del SO

Funcions de la RSE: crides al sistema

- Excepció de crida al sistema i trap (p.ex. *syscall*)

- És similar a una crida a subrutina
  - Amb pas de paràmetres i retorn de resultats
  - La RSE executa el servei sol·licitat del SO

- Un cop finalitza la RSE
  - S'incrementa l'EPC i se salta *a la instrucció següent*, per evitar un bucle infinit

Funcions de la RSE: interrupcions

- Interrupcions
  - Causades per l'activació del senyal hardware INT (petició d'interrupció) d'algun dispositiu d'E/S

Funcions de la RSE: interrupcions

- Interrupcions
  - Causades per l'activació del senyal hardware INT (petició d'interrupció) d'algun dispositiu d'E/S

- La petició no s'atén fins que acaba la instrucció en curs
  - Mentrestant, queda registrada en un registre de peticions pendents
  - Quan s'atén, la RSE sol transferir dades al/del dispositiu
  - El SO pot planificar altres processos mentre dura l'operació d'E/S

Funcions de la RSE: interrupcions

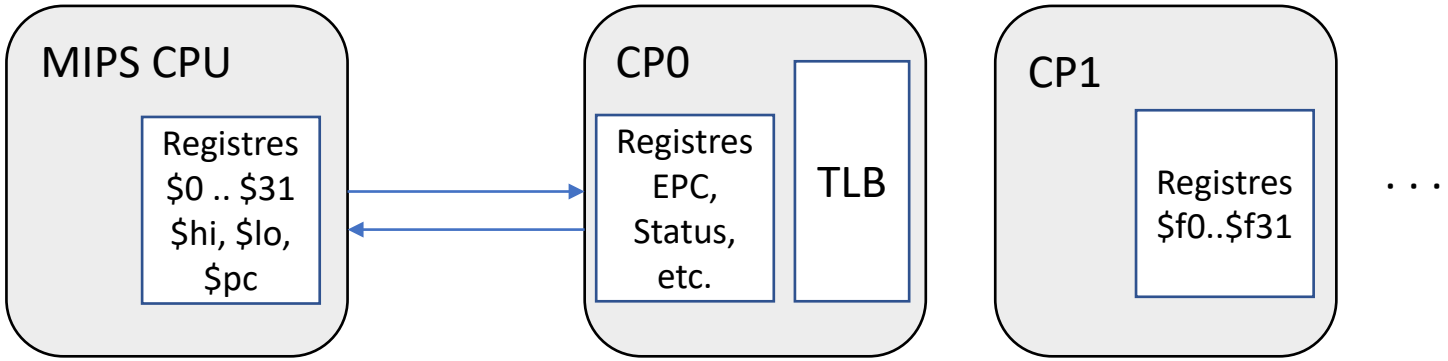
- Interrupcions
  - Causades per l'activació del senyal hardware INT (petició d'interrupció) d'algun dispositiu d'E/S
- La petició no s'atén fins que acaba la instrucció en curs
  - Mentrestant, queda registrada en un registre de peticions pendents
  - Quan s'atén, la RSE sol transferir dades al/del dispositiu
  - El SO pot planificar altres processos mentre dura l'operació d'E/S
- Un cop finalitza la RSE
  - *Se segueix executant el programa*

Excepcions i Interrupcions

- Introducció
- Implementació en MIPS
  - El coprocessador de sistema CP0
  - Accions del hardware en una excepció
  - Accions del software (RSE) en una excepció
- Tres exemples concrets

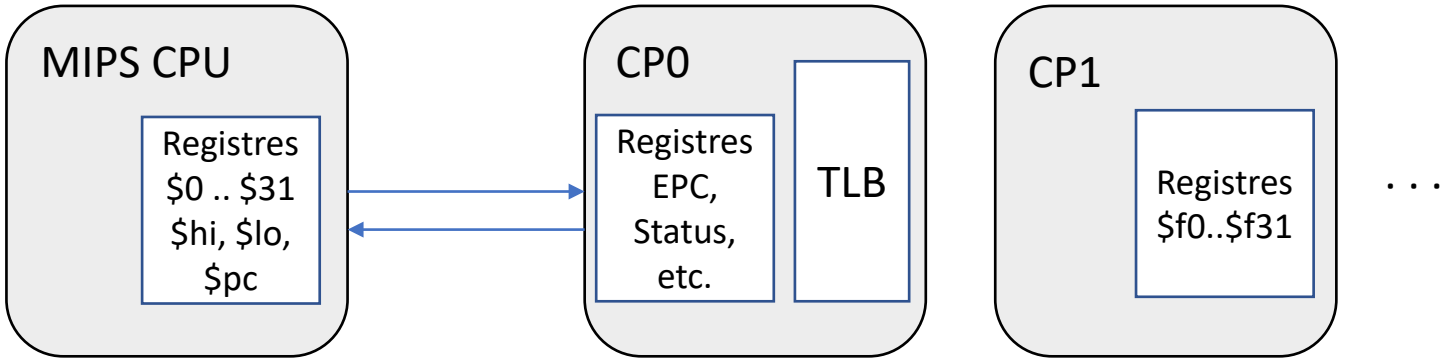
El coprocessador de sistema CP0

- Coprocessador del sistema CP0
  - Controla el tractament d'excepcions i la traducció amb TLB



El coprocessador de sistema CP0

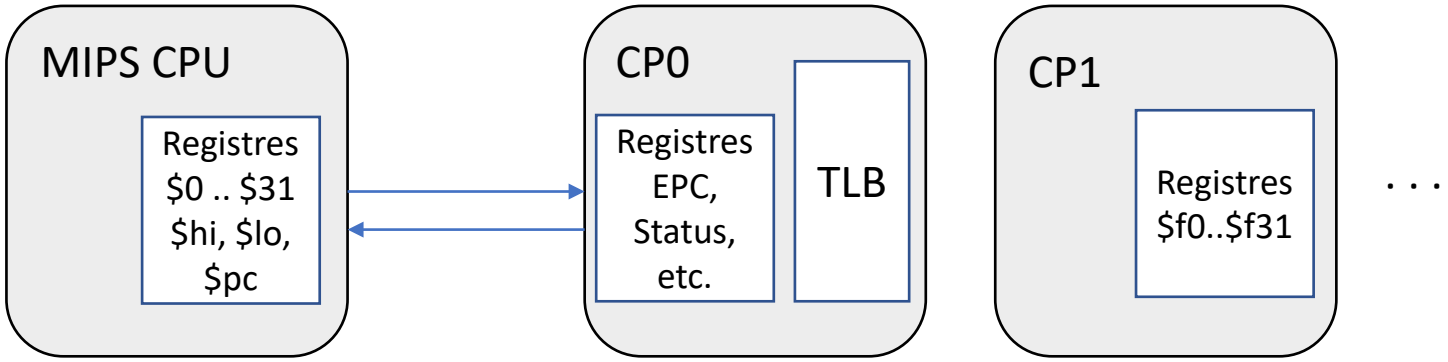
- Coprocessador del sistema CP0
  - Controla el tractament d'excepcions i la traducció amb TLB
- Afegeix instruccions addicionals
  - Privilegiades: sols es poden executar en mode sistema



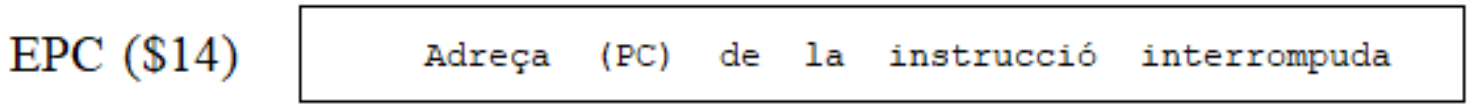
El coprocessador de sistema CP0

- Coprocessador del sistema CP0
  - Controla el tractament d'excepcions i la traducció amb TLB
- Afegeix instruccions addicionals
  - Privilegiades: sols es poden executar en mode sistema
- Té un banc de registres específic
  - Només són accessibles per les instruccions privilegiades
  - Per exemple, per copiar-los de CP0 a CPU o viceversa:

```
mfc0 rt, rd_c0      # rt ← rd_c0
mtc0 rt, rd_c0      # rt → rd_c0
```



El coprocessador de sistema CP0



Registre EPC (Exception Program Counter, $14)

- Registre de lectura/escriptura
- Abans d'invocar la RSE, el processador guarda el valor actual del PC en EPC, serà l'adreça de retorn

El coprocessador de sistema CP0

EPC ($14)	Adreça (PC) de la instrucció interrompuda

Registre EPC (Exception Program Counter, $14)

- Registre de lectura/escriptura
- Abans d'invocar la RSE, el processador guarda el valor actual del PC en EPC, serà l'adreça de retorn

- Una pregunta de test...
  - Per què no es pot guardar l'adreça de retorn en $ra?

El coprocessador de sistema CP0

EPC ($14)	Adreça (PC) de la instrucció interrompuda

Registre EPC (Exception Program Counter, $14)

- Registre de lectura/escriptura
- Abans d'invocar la RSE, el processador guarda el valor actual del PC en EPC, serà l'adreça de retorn

- Una pregunta de test...
  - Per què no es pot guardar l'adreça de retorn en $ra?

- On apunta el PC que es guarda a l'EPC?
  - En cas d'excepció, la instrucció en curs s'avorta
    → el PC encara apunta a la instrucció en curs

El coprocessador de sistema CP0

EPC ($14)	Adreça (PC) de la instrucció interrompuda

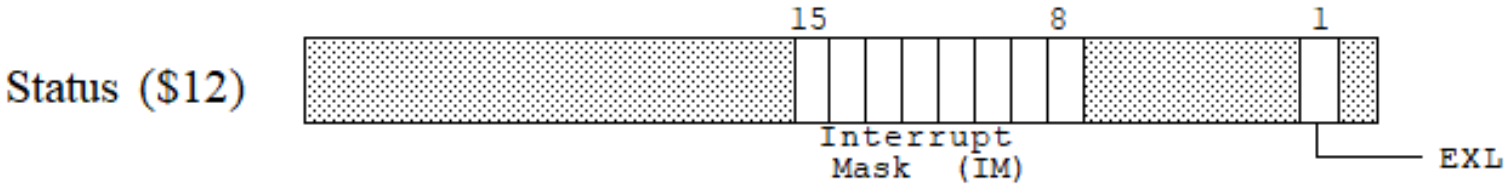
Registre EPC (Exception Program Counter, $14)

- Registre de lectura/escriptura
- Abans d'invocar la RSE, el processador guarda el valor actual del PC en EPC, serà l'adreça de retorn

- Una pregunta de test...
  - Per què no es pot guardar l'adreça de retorn en $ra?

- On apunta el PC que es guarda a l'EPC?
  - En cas d'excepció, la instrucció en curs s'avorta
    → el PC encara apunta a la instrucció en curs
  - En cas d'interrupció: la instrucció en curs finalitza
    → el PC s'ha incrementat i apunta a la següent

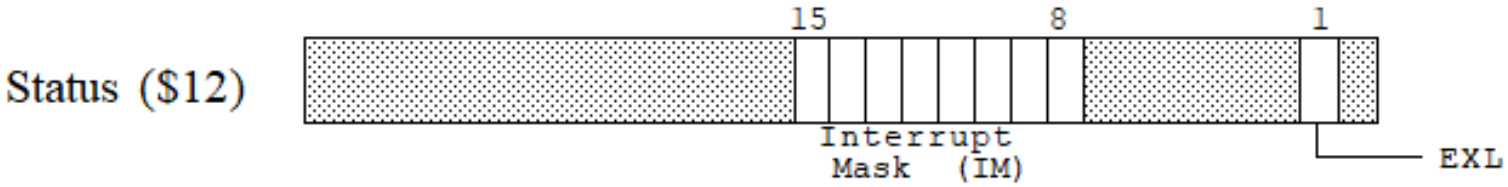
El coprocessador de sistema CP0



Registre Status ($12):

- Bit EXL (Exception Level), té 2 funcions:
  - Mode usuari / mode sistema
    - Si EXL=0, el processador està en mode usuari
    - Si EXL=1, el processador està en mode sistema, té accés a les adreces del SO, i pot executar instruccions privilegiades
    - S'activa quan es produeix una excepció o interrupció

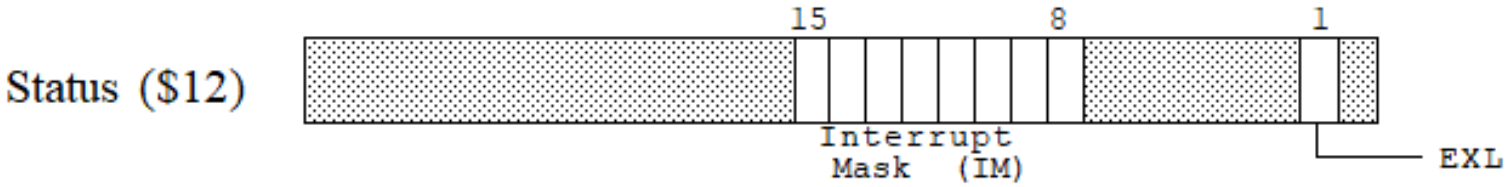
El coprocessador de sistema CP0



Registre Status ($12):

- Bit EXL (Exception Level), té 2 funcions:
  - Mode usuari / mode sistema
    - Si EXL=0, el processador està en mode usuari
    - Si EXL=1, el processador està en mode sistema, té accés a les adreces del SO, i pot executar instruccions privilegiades
    - S'activa quan es produeix una excepció o interrupció
  - Si EXL=1, totes les interrupcions estan inhibides i s'ignoren
    - Però no es poden inhibir les excepcions!

El coprocessador de sistema CP0



Registre Status ($12):

- Bit EXL (Exception Level), té 2 funcions:
  - Mode usuari / mode sistema
    - Si EXL=0, el processador està en mode usuari
    - Si EXL=1, el processador està en mode sistema, té accés a les adreces del SO, i pot executar instruccions privilegiades
    - S'activa quan es produeix una excepció o interrupció
  - Si EXL=1, totes les interrupcions estan inhibides i s'ignoren
    - Però no es poden inhibir les excepcions!
- 8 bits de IM (Interrupt Mask)
  - 1 bit per a cada tipus d'interrupció
  - Si el bit val 1, les interrupcions d'aquest tipus estan habilitades

El coprocessador de sistema CP0

Registre Cause ($13), de sols lectura

- Camp ExcCode, codifica la causa de l'excepció, p.ex:

Núm	Nom	Causa
0	Int	interrupció d'un dispositiu hardware d'E/S
1	Mod	fallada de TLB per pàgina modificada (primera escriptura a pàgina)
2	TLBL	fallada de TLB (o fallada de pàgina) per lectura
3	TLBS	fallada de TLB (o fallada de pàgina) per escriptura
4	AdEL	error d'adreça per lectura (accés mal alineat, o a espai de sistema no permès)
5	AdES	error d'adreça per escriptura (accés mal alineat, o a espai de sistema no permès)
6	IBE	error de bus (instruction fetch), p.ex. adreça física inexistent
7	DBE	error de bus (dades de load/store), p.ex. adreça física inexistent
8	Sys	crida al sistema (causada per la instrucció syscall)
9	Bp	breakpoint (causada per la instrucció break)
10	RI	instrucció reservada. p.ex. codis d'operació inexistents
11	CpU	coprocessador no implementat, p.ex. accés a CP0 en mode usuari
12	Ov	overflow aritmètic d'enters (instruccions add, sub, addi)
13	Tr	trap (causada per una instrucció de trap)
15	FPE	excepció de coma flotant: consultar detalls als registres del coprocessador CP1

El coprocessador de sistema CP0

- 8 bits de IP (Interrupt Pending)
  - Un per cada tipus d'interrupció
  - Quan es rep una petició d'interrupció s'activa el bit corresponent
  - El bit roman activat mentre el dispositiu mantingui el senyal de petició activat

Accions del hardware en una excepció

1. Quan el hardware detecta...
   - ... una excepció
     - La instrucció en curs s'avorta: no escriu en registres o memòria
     - No s'incrementa el PC

Accions del hardware en una excepció

1. Quan el hardware detecta...
   - ... una excepció
     - La instrucció en curs s'avorta: no escriu en registres o memòria
     - No s'incrementa el PC
   - ... una interrupció (i les interrupcions estan permeses, EXL=0)
     - Finalitza la instrucció en curs
     - Es comproven les peticions pendents (IP_i=1) i habilitades (IM_i=1)
     - Si n'hi ha més d'una, se selecciona la més prioritària

Accions del hardware en una excepció

1. Quan el hardware detecta...
   - ... una excepció
     - La instrucció en curs s'avorta: no escriu en registres o memòria
     - No s'incrementa el PC
   - ... una interrupció (i les interrupcions estan permeses, EXL=0)
     - Finalitza la instrucció en curs
     - Es comproven les peticions pendents (IP_i=1) i habilitades (IM_i=1)
     - Si n'hi ha més d'una, se selecciona la més prioritària
2. Guarda el PC en el registre EPC

Accions del hardware en una excepció

1. Quan el hardware detecta...
   - ... una excepció
     - La instrucció en curs s'avorta: no escriu en registres o memòria
     - No s'incrementa el PC
   - ... una interrupció (i les interrupcions estan permeses, EXL=0)
     - Finalitza la instrucció en curs
     - Es comproven les peticions pendents (IP_i=1) i habilitades (IM_i=1)
     - Si n'hi ha més d'una, se selecciona la més prioritària
2. Guarda el PC en el registre EPC
3. Escriu la causa en el camp ExcCode del registre Cause

Accions del hardware en una excepció

1. Quan el hardware detecta...
   - ... una excepció
     - La instrucció en curs s'avorta: no escriu en registres o memòria
     - No s'incrementa el PC
   - ... una interrupció (i les interrupcions estan permeses, EXL=0)
     - Finalitza la instrucció en curs
     - Es comproven les peticions pendents (IP_i=1) i habilitades (IM_i=1)
     - Si n'hi ha més d'una, se selecciona la més prioritària
2. Guarda el PC en el registre EPC
3. Escriu la causa en el camp ExcCode del registre Cause
4. Posa a 1 el bit EXL (mode sistema, interrupc. inhibides)

Accions del hardware en una excepció

1. Quan el hardware detecta...
   - ... una excepció
     - La instrucció en curs s'avorta: no escriu en registres o memòria
     - No s'incrementa el PC
   - ... una interrupció (i les interrupcions estan permeses, EXL=0)
     - Finalitza la instrucció en curs
     - Es comproven les peticions pendents (IP_i=1) i habilitades (IM_i=1)
     - Si n'hi ha més d'una, se selecciona la més prioritària
2. Guarda el PC en el registre EPC
3. Escriu la causa en el camp ExcCode del registre Cause
4. Posa a 1 el bit EXL (mode sistema, interrupc. inhibides)
5. Escriu en el PC l'adreça base de la rutina de servei

   (en MIPS, hi ha 2 rutines de servei diferents)
   - RSE (genèrica), PC = 0x800000180
   - TLBmiss (sols fallades de TLB), PC = 0x80000000

Accions software en una excepció (RSE)

1. La RSE ha de preservar l'estat del programa interromput
   - Salva **TOTS** els registres ($0 - $31) a la pila (excepte $k0, $k1)
   - No cal salvar els registres de coma flotant ($f0 - $f31)
     - Sols si invoca alguna subrutina específica on es modifiquin

Accions software en una excepció (RSE)

1. La RSE ha de preservar l'estat del programa interromput
   - Salva **TOTS** els registres ($0 - $31) a la pila (excepte $k0, $k1)
   - No cal salvar els registres de coma flotant ($f0 - $f31)
     - Sols si invoca alguna subrutina específica on es modifiquin
2. Identificar la causa (ExcCode en el registre Cause)

Accions software en una excepció (RSE)

1. La RSE ha de preservar l'estat del programa interromput
   - Salva **TOTS** els registres ($0 - $31) a la pila (excepte $k0, $k1)
   - No cal salvar els registres de coma flotant ($f0 - $f31)
     - Sols si invoca alguna subrutina específica on es modifiquin
2. Identificar la causa (ExcCode en el registre Cause)
3. Saltar a una subrutina específica, la qual pot...
   - ... avortar el programa
   - ... bloquejar-lo temporalment, donant pas a altres programes
   - ... o solucionar el problema i continuar l'execució del programa

Accions software en una excepció (RSE)

1. La RSE ha de preservar l'estat del programa interromput
   - Salva **TOTS** els registres ($0 - $31) a la pila (excepte $k0, $k1)
   - No cal salvar els registres de coma flotant ($f0 - $f31)
     - Sols si invoca alguna subrutina específica on es modifiquin
2. Identificar la causa (ExcCode en el registre Cause)
3. Saltar a una subrutina específica, la qual pot...
   - ... avortar el programa
   - ... bloquejar-lo temporalment, donant pas a altres programes
   - ... o solucionar el problema i continuar l'execució del programa
4. En cas de syscall, sumar 4 a EPC per no reexecutar-la

Accions software en una excepció (RSE)

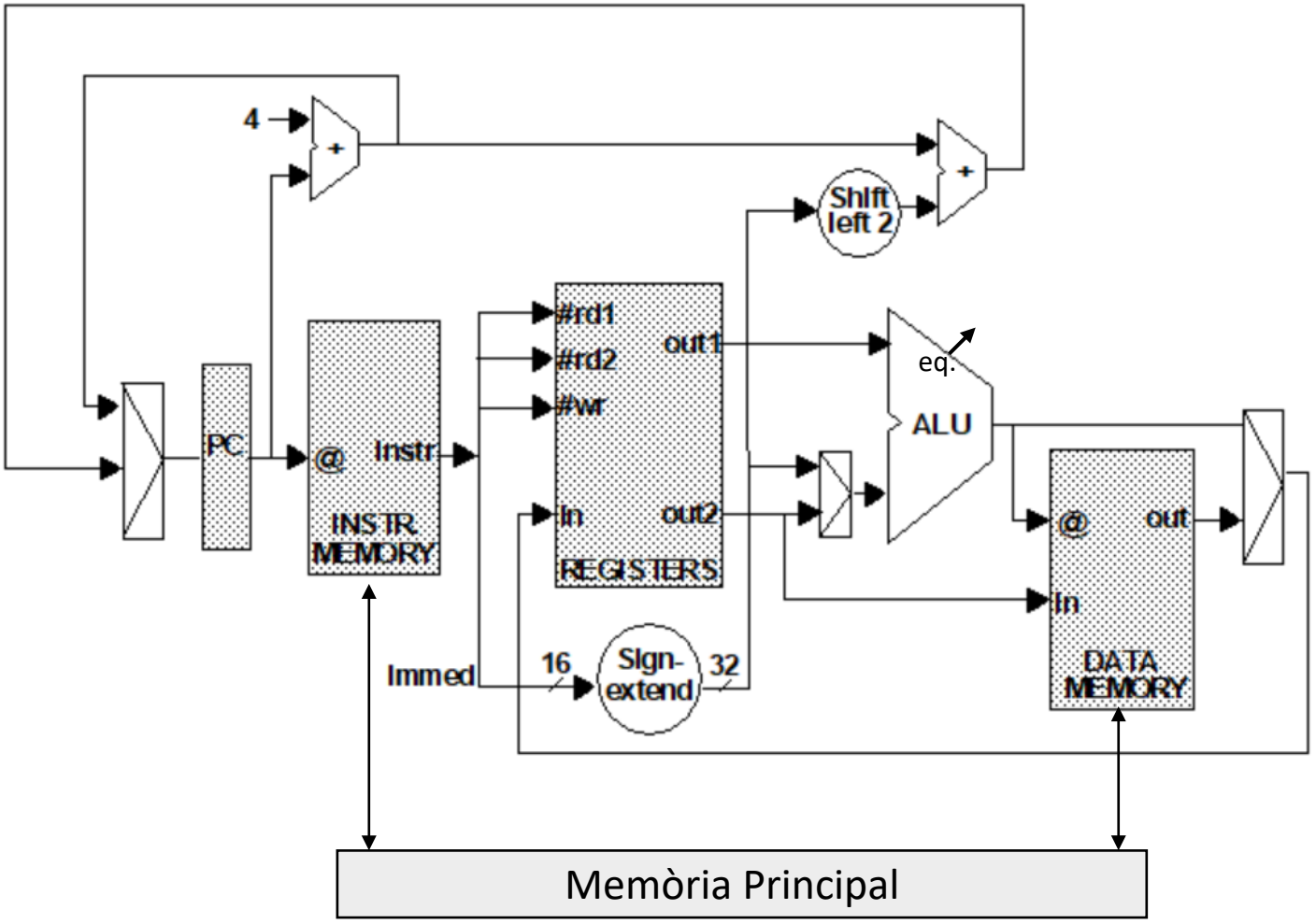
1. La RSE ha de preservar l'estat del programa interromput
   - Salva **TOTS** els registres ($0 - $31) a la pila (excepte $k0, $k1)
   - No cal salvar els registres de coma flotant ($f0 - $f31)
     - Sols si invoca alguna subrutina específica on es modifiquin
2. Identificar la causa (ExcCode en el registre Cause)
3. Saltar a una subrutina específica, la qual pot...
   - ... avortar el programa
   - ... bloquejar-lo temporalment, donant pas a altres programes
   - ... o solucionar el problema i continuar l'execució del programa
4. En cas de syscall, sumar 4 a EPC per no reexecutar-la
5. Restaurar tots els registres salvats a la pila

Accions software en una excepció (RSE)

1. La RSE ha de preservar l'estat del programa interromput
   - Salva **TOTS** els registres ($0 - $31) a la pila (excepte $k0, $k1)
   - No cal salvar els registres de coma flotant ($f0 - $f31)
     - Sols si invoca alguna subrutina específica on es modifiquin
2. Identificar la causa (ExcCode en el registre Cause)
3. Saltar a una subrutina específica, la qual pot...
   - ... avortar el programa
   - ... bloquejar-lo temporalment, donant pas a altres programes
   - ... o solucionar el problema i continuar l'execució del programa
4. En cas de syscall, sumar 4 a EPC per no reexecutar-la
5. Restaurar tots els registres salvats a la pila
6. Retornar, usant la instrucció eret (exception return)
   - Posa EXL=0 (mode usuari, interrupcions permeses)
   - Copia EPC en el PC (salta a l'adreça de retorn)

Canvis al camí de dades del MIPS

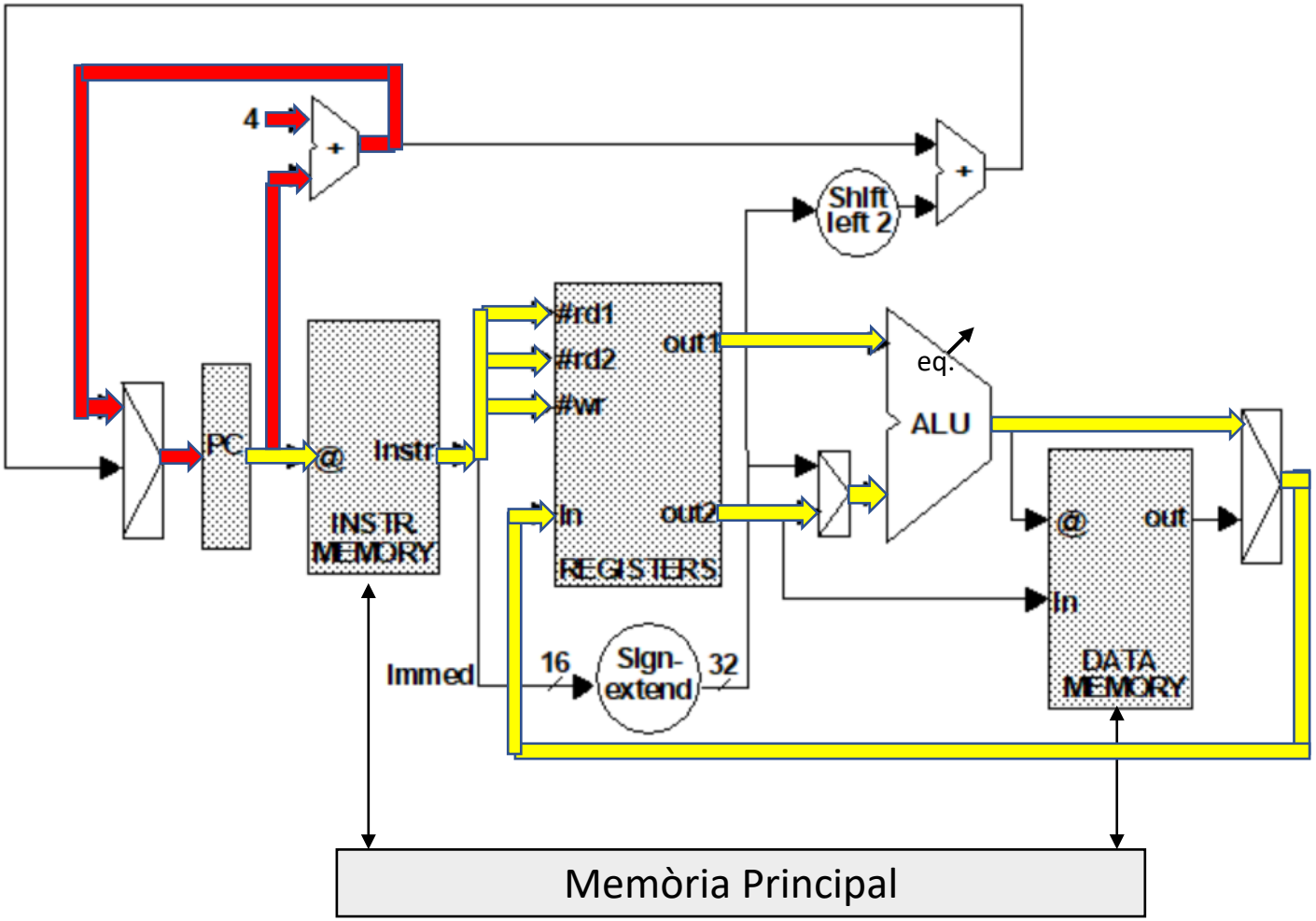
- Possible disseny d'un MIPS



Canvis al camí de dades del MIPS

- Possible disseny d'un MIPS

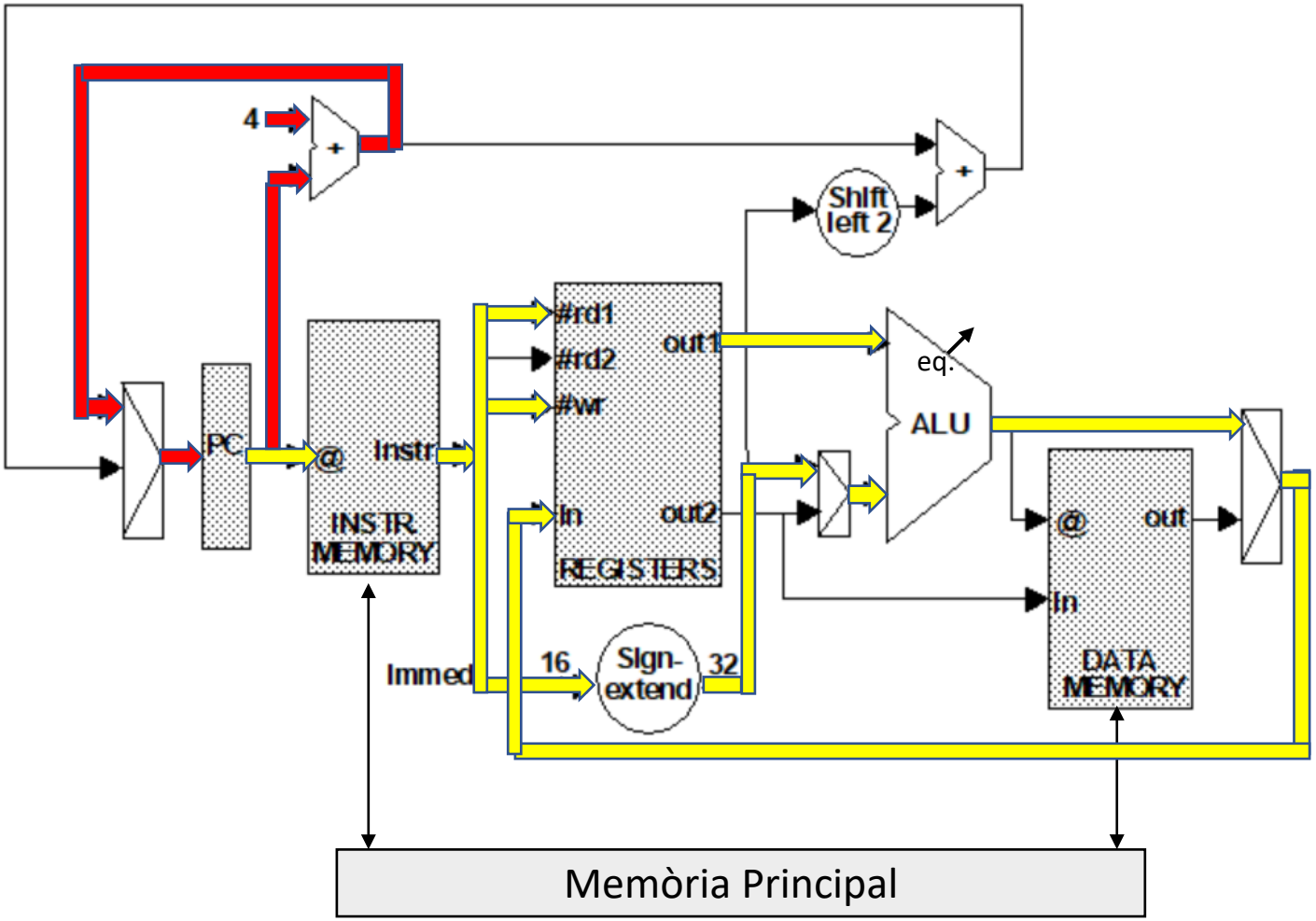
Exemple: `addu`



Canvis al camí de dades del MIPS

- Possible disseny d'un MIPS

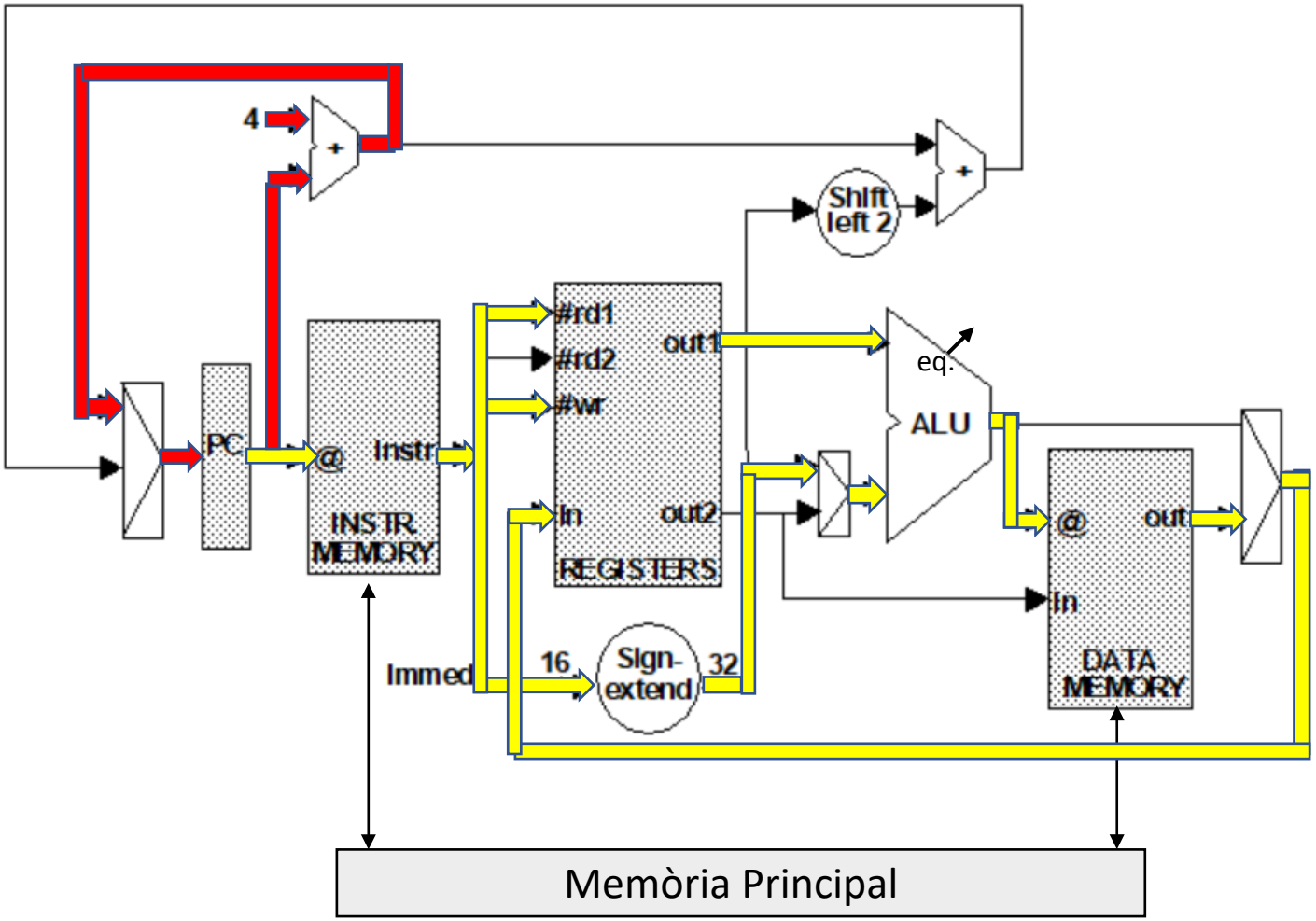
Exemple: `addiu`



Canvis al camí de dades del MIPS

- Possible disseny d'un MIPS

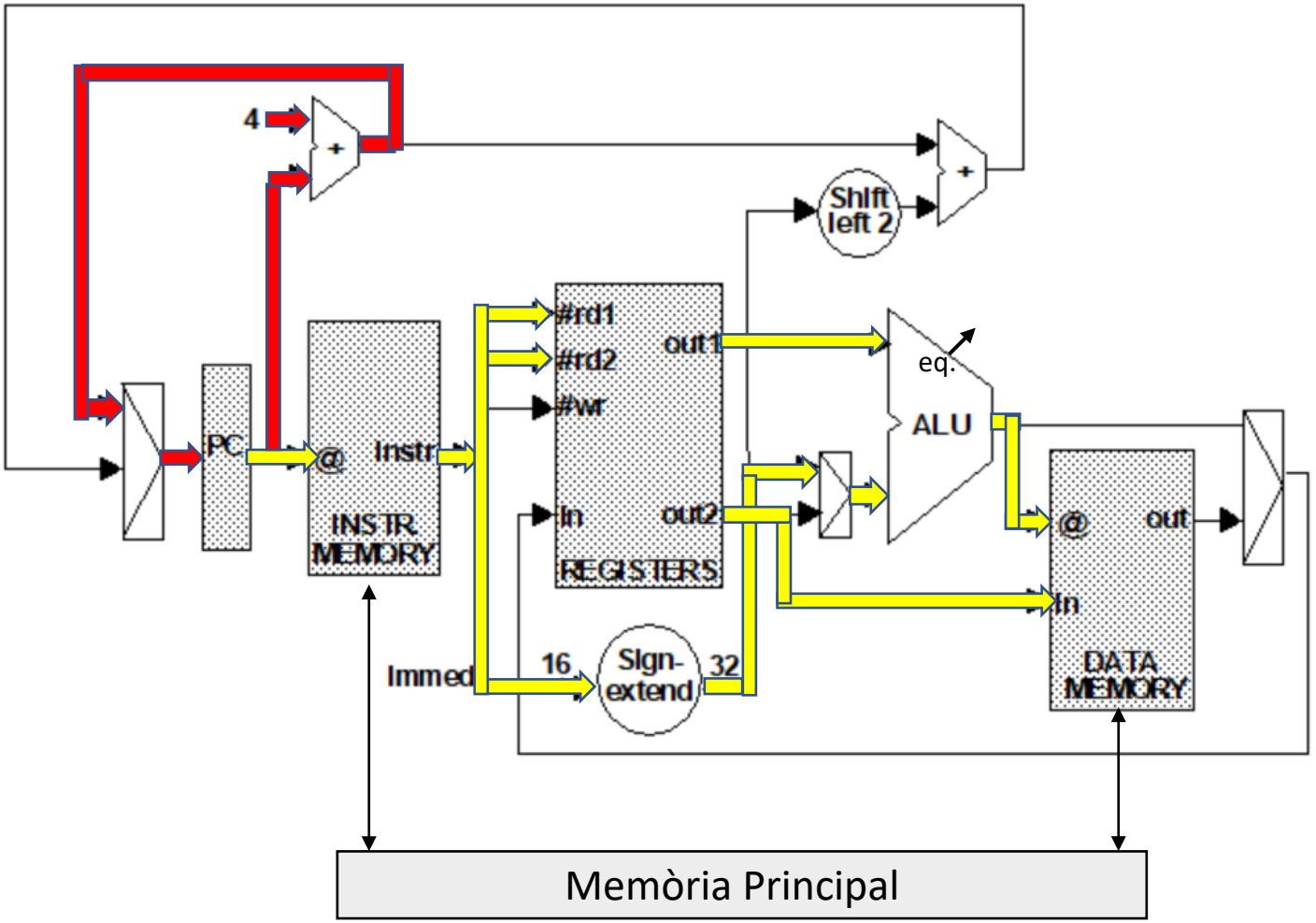
Exemple: `lw`



Canvis al camí de dades del MIPS

- Possible disseny d'un MIPS

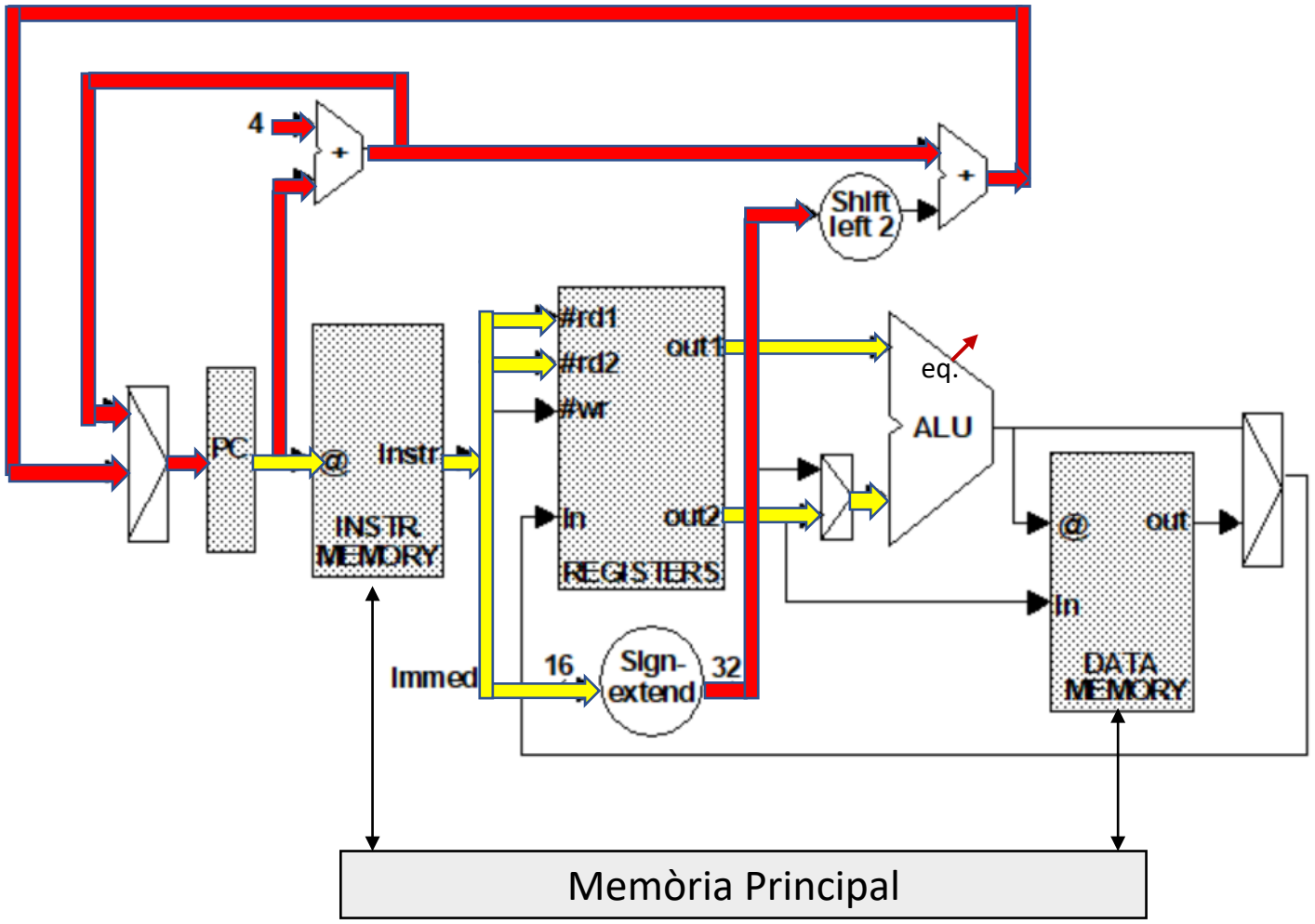
Exemple: `sw`



Canvis al camí de dades del MIPS

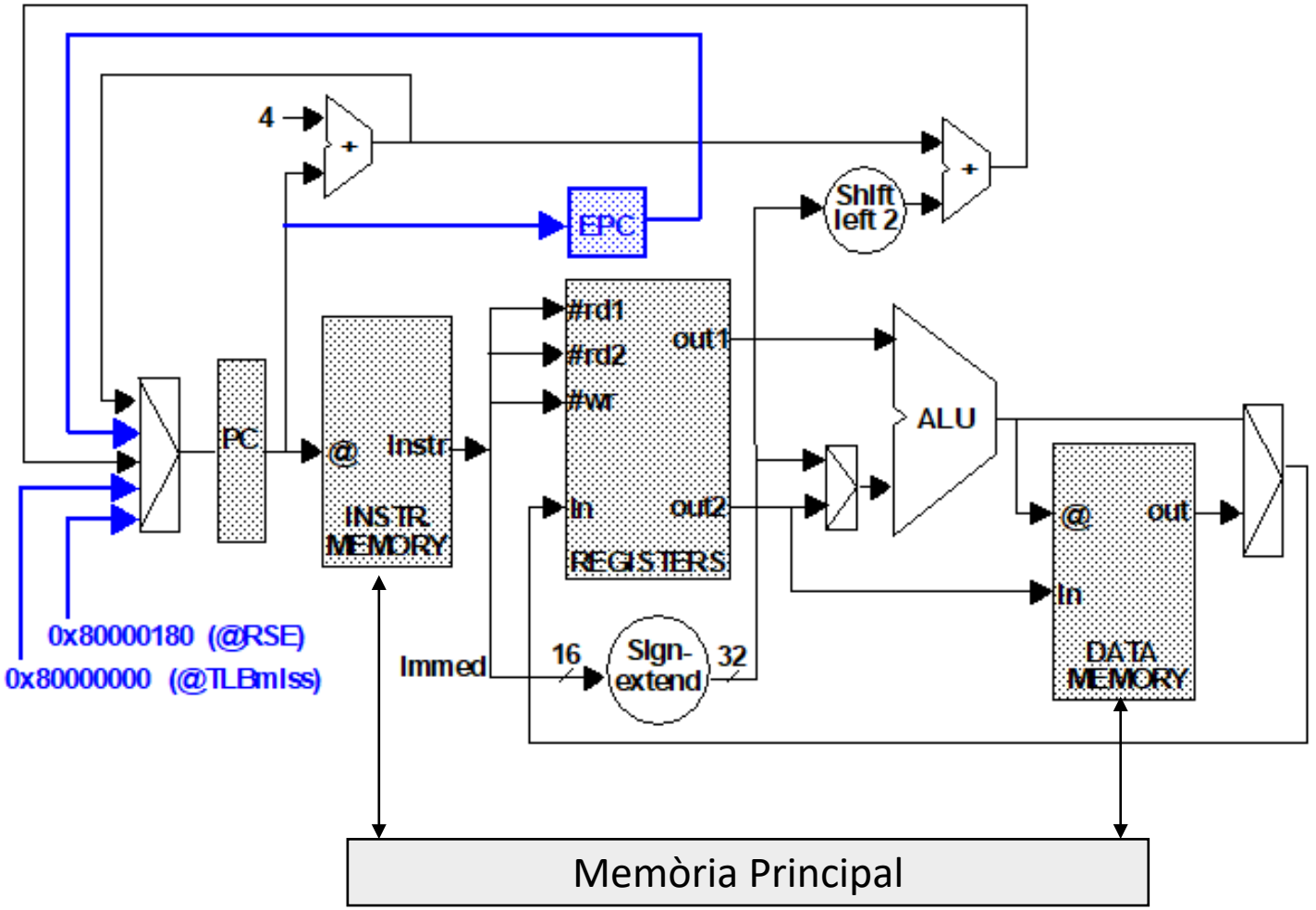
- Possible disseny d'un MIPS

Exemple: `beq`



Canvis al camí de dades del MIPS

- Possible disseny d'un MIPS incloent suport d'excepcions



Un exemple senzill de RSE (1)

```
      .ktext 0x80000180          # secció "kernel text"
RSE:
      # Salvar TOTS els registres incloent $hi, $lo però no $k0, $k1, $sp
      addiu  $sp, $sp, -128
      sw     $1, 0($sp)
      sw     $2, 4($sp)
      sw     $3, 8($sp)
      sw     $4, 12($sp)
      . . .
      sw     $31, 124($sp)

      # Passar paràmetres Cause i EPC a una rutina específica
      mfc0   $a0, $13           # Passa Cause Register (amb ExcCode)
      mfc0   $a1, $14           # Passa EPC
      jal    handler_dispatcher # invocarà el handler que correspongui
                                # i decidirà on ha de retornar la RSE
      mtc0   $v0, $14           # copiar resultat a EPC

      # Restaurar registres i pila
      addiu  $sp, $sp, -128
      lw     $1, 0($sp)
      lw     $2, 4($sp)
      lw     $3, 8($sp)
      lw     $4, 12($sp)
      . . .
```

Un exemple senzill de RSE (2)

```
sw      $3, 8($sp)
sw      $4, 12($sp)
. . .
sw      $31, 124($sp)

# Passar paràmetres Cause i EPC a una rutina específica
mfc0    $a0, $13              # Passa Cause Register (amb ExcCode)
mfc0    $a1, $14              # Passa EPC
jal     handler_dispatcher    # invocarà el handler que correspongui
                              # i decidirà on ha de retornar la RSE
mtc0    $v0, $14              # copiar resultat a EPC

# Restaurar registres i pila
addiu   $sp, $sp, -128
lw      $1, 0($sp)
lw      $2, 4($sp)
lw      $3, 8($sp)
lw      $4, 12($sp)
. . .
lw      $31, 124($sp)
addiu   $sp, $sp, 128

# Retornar al programa d'usuari, posant EXL=0 i PC=EPC
eret
```

Un exemple senzill de RSE (3)

```
sw      $3, 8($sp)
sw      $4, 12($sp)
. . .
sw      $31, 124($sp)

# Passar paràmetres Cause i EPC a una rutina específica
mfc0    $a0, $13              # Passa Cause Register (amb ExcCode)
mfc0    $a1, $14              # Passa EPC
jal     handler_dispatcher    # invocarà el handler que correspongui
                              # i decidirà on ha de retornar la RSE
mtc0    $v0, $14              # copiar resultat a EPC
```

```
# Restaurar registres i pila

lw      $1, 0($sp)
lw      $2, 4($sp)
lw      $3, 8($sp)
lw      $4, 12($sp)
. . .
lw      $31, 124($sp)
addiu   $sp, $sp, 128
```

```
# Retornar al programa d'usuari, posant EXL=0 i PC=EPC
eret
```

Un exemple senzill de RSE (4)

```
sw      $3, 8($sp)
sw      $4, 12($sp)
. . .
sw      $31, 124($sp)

# Passar paràmetres Cause i EPC a una rutina específica
mfc0    $a0, $13            # Passa Cause Register (amb ExcCode)
mfc0    $a1, $14            # Passa EPC
jal     handler_dispatcher  # invocarà el handler que correspongui
                            # i decidirà on ha de retornar la RSE
mtc0    $v0, $14            # copiar resultat a EPC

# Restaurar registres i pila

lw      $1, 0($sp)
lw      $2, 4($sp)
lw      $3, 8($sp)
lw      $4, 12($sp)
. . .
lw      $31, 124($sp)
addiu   $sp, $sp, 128

# Retornar al programa d'usuari, posant EXL=0 i PC=EPC
eret
```

Excepcions i Interrupcions

- Introducció
- Implementació en MIPS
- Tres exemples concrets
  - L'excepció de fallada de TLB
  - L'excepció de crida al sistema
  - Les interrupcions d'E/S

Fallada de TLB

- Fallada de TLB
  - La CPU accedeix a una adreça i el TLB no en té la traducció
- Tractament
  - Buscar entrada lliure al TLB (o seleccionar-ne una a reemplaçar)

Fallada de TLB

- Fallada de TLB
  - La CPU accedeix a una adreça i el TLB no en té la traducció
- Tractament
  - Buscar entrada lliure al TLB (o seleccionar-ne una a reemplaçar)
  - Copiar-hi l'entrada corresponent de la TP (PTE)

Fallada de TLB

- Fallada de TLB
  - La CPU accedeix a una adreça i el TLB no en té la traducció
- Tractament
  - Buscar entrada lliure al TLB (o seleccionar-ne una a reemplaçar)
  - Copiar-hi l'entrada corresponent de la TP (PTE)
  - Inclou els bits: Marc de pàgina (PPN), Presència (V) i Dirty (D)

Fallada de TLB

- Fallada de TLB
  - La CPU accedeix a una adreça i el TLB no en té la traducció
- Tractament
  - Buscar entrada lliure al TLB (o seleccionar-ne una a reemplaçar)
  - Copiar-hi l'entrada corresponent de la TP (PTE)
  - Inclou els bits: Marc de pàgina (PPN), Presència (V) i Dirty (D)
  - Afegir-hi el número de pàgina VPN (etiqueta de la traducció)

Fallada de TLB en MIPS

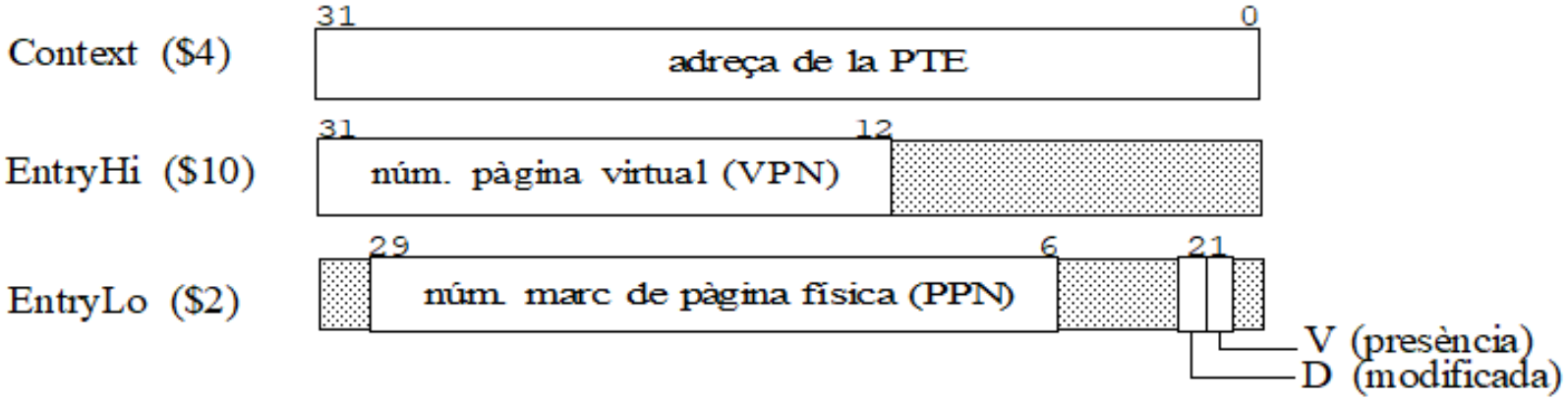
- En MIPS, les fallades de TLB causen excepció
  - Les tracta una subrutina del SO (software)
  - Són relativament freqüents, cal tractar-les eficientment
  - Però la RSE genèrica (@ = 0x80000180) és massa lenta (salvar tots els registres, etc...) !!

Fallada de TLB en MIPS

- En MIPS, les fallades de TLB causen excepció
  - Les tracta una subrutina del SO (software)
  - Són relativament freqüents, cal tractar-les eficientment
  - Però la RSE genèrica (@ = 0x80000180) és massa lenta (salvar tots els registres, etc...) !!
- Rutina específica
  - TLBmiss (@ = 0x80000000) $\rightarrow$ molt ràpida (menys de 13 cicles)
  - No salva registres a la pila (sols modifica $k1, que no cal salvar perquè està reservat per al SO)

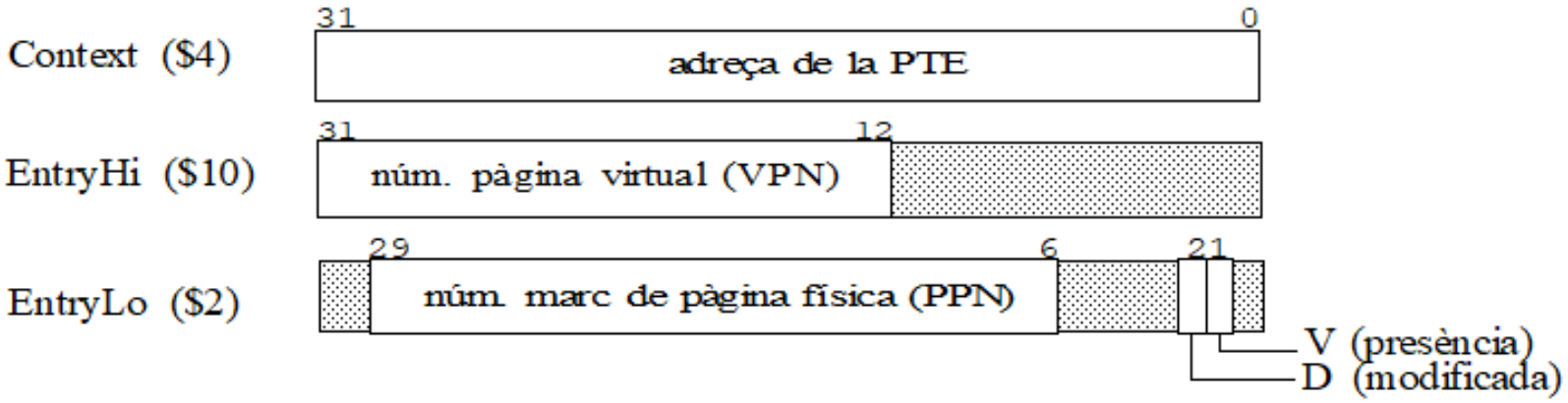
Fallada de TLB en MIPS

- En MIPS, les fallades de TLB causen excepció
  - Les tracta una subrutina del SO (software)
  - Són relativament freqüents, cal tractar-les eficientment
  - Però la RSE genèrica (@ = 0x80000180) és massa lenta (salvar tots els registres, etc...) !!
- Rutina específica
  - TLBmiss (@ = 0x80000000) → molt ràpida (menys de 13 cicles)
  - No salva registres a la pila (sols modifica $k1, que no cal salvar perquè està reservat per al SO)
- Registres del CP0 per a la gestió del TLB



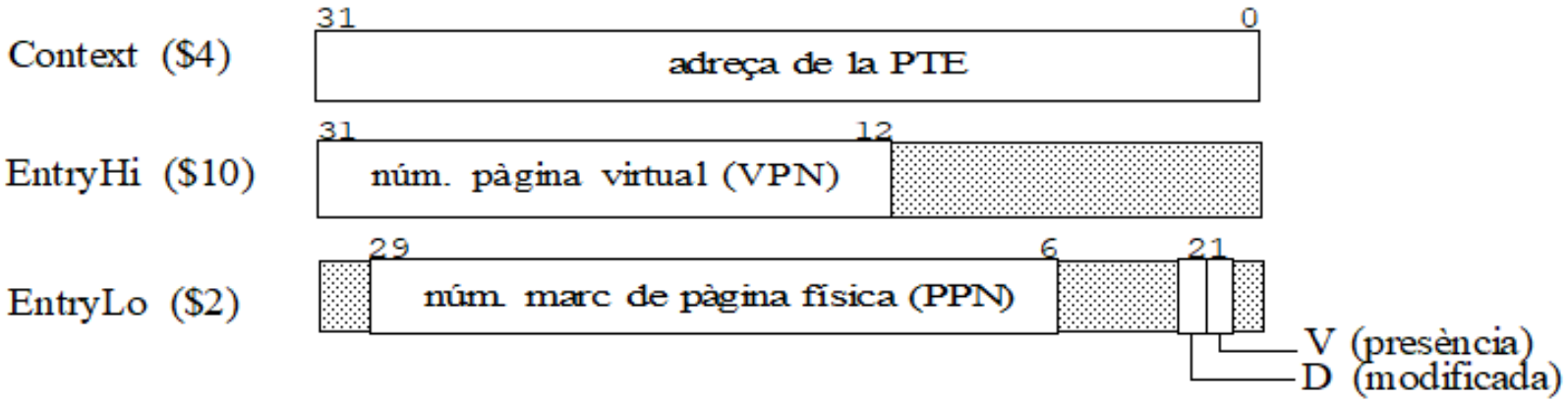
Fallada de TLB en MIPS

- Quan el hardware detecta la fallada de TLB...
  - ... escriu l'adreça de l'entrada de la TP (PTE) al registre Context
  - ... Escriu els 20 bits del VPN al registre EntryHi



Fallada de TLB en MIPS

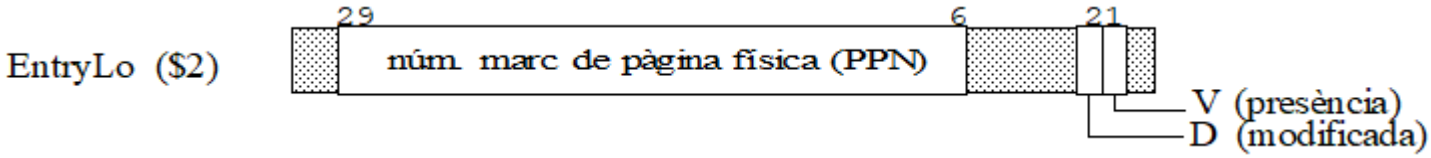
- Quan el hardware detecta la fallada de TLB...
  - ... escriu l'adreça de l'entrada de la TP (PTE) al registre Context
  - ... Escriu els 20 bits del VPN al registre EntryHi
- I a més a més...
  - ... escriu en ExcCode el codi TLBL (fetch o load) o TLBS (store) (com s'ha explicat abans)
  - ... copia el PC en EPC, i posa el bit EXL=1
  - ... escriu en el PC l'adreça base de TLBmiss (= 0x80000000) (i s'executa TLBmiss)



Exemple de rutina TLBmiss

```
    .ktext 0x80000000
TLBmiss:
    mfc0   $k1, $4        # Copia Context (adreça de la PTE) en $k1
    lw     $k1, 0($k1)    # Llegeix la PTE (traduccció) en $k1
    mtc0   $k1, $2        # Copia la traducció a EntryLo
    tlbwr                 # Escriu EntryHi|EntryLo al TLB
    eret                  # Retorna al programa
```

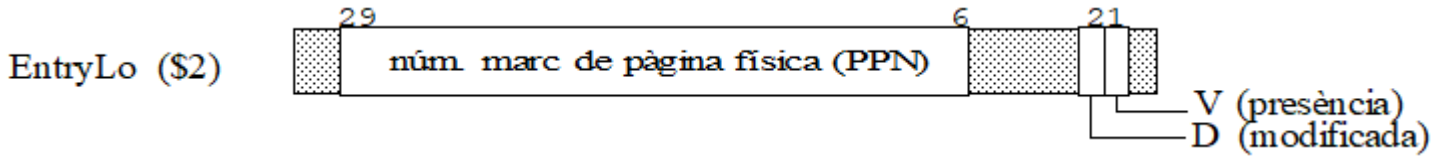
- Fa còpia de la PTE (32 bits) al registre EntryLo



Exemple de rutina TLBmiss

```
      .ktext 0x80000000
TLBmiss:
      mfc0   $k1, $4        # Copia Context (adreça de la PTE) en $k1
      lw     $k1, 0($k1)    # Llegeix la PTE (traducció) en $k1
      mtc0   $k1, $2        # Copia la traducció a EntryLo
      tlbwr                 # Escriu EntryHi|EntryLo al TLB
      eret                  # Retorna al programa
```

- Fa còpia de la PTE (32 bits) al registre EntryLo

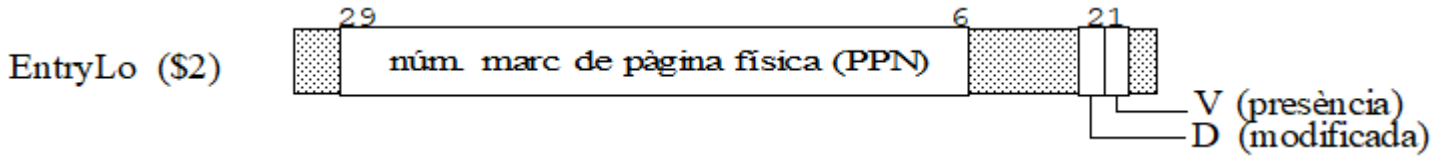


- **`TLBWR`** = "TLB Write Random"
  - Selecciona una entrada del TLB aleatòriament
  - Hi escriu el contingut de EntryHi (VPN) i EntryLo (PPN, V, D)

Exemple de rutina TLBmiss

```
      .ktext 0x80000000
TLBmiss:
      mfc0   $k1, $4        # Copia Context (adreça de la PTE) en $k1
      lw     $k1, 0($k1)    # Llegeix la PTE (traducció) en $k1
      mtc0   $k1, $2        # Copia la traducció a EntryLo
      tlbwr                 # Escriu EntryHi|EntryLo al TLB
      eret                  # Retorna al programa
```

- Fa còpia de la PTE (32 bits) al registre EntryLo



- `TLBWR` = "TLB Write Random"
  - Selecciona una entrada del TLB aleatòriament
  - Hi escriu el contingut de EntryHi (VPN) i EntryLo (PPN, V, D)

- **`ERET`** = "Exception Return"
  - Restaura el PC: PC ← EPC
  - Restaura EXL: EXL ← 0

Fallada de TLB en MIPS

Què passa si en una fallada de TLB el bit V val 0?

- Durant una fallada de TLB, la rutina TLBmiss
  - Copia la traducció de la TP al TLB
  - Però **no** comprova el bit de presència V
  - Quan acaba, es reexecuta la instrucció que ha fallat

Fallada de TLB en MIPS

Què passa si en una fallada de TLB el bit V val 0?

- Durant una fallada de TLB, la rutina TLBmiss
  - Copia la traducció de la TP al TLB
  - Però **no** comprova el bit de presència V
  - Quan acaba, es reexecuta la instrucció que ha fallat
- Quan la instrucció es reexecuta
  - Hi ha encert de TLB, i es comprova el bit de presència V
  - Si V=1, la pàgina està present en MP...
    ...i la instrucció s'executa normalment
  - Si V=0, hi ha excepció de fallada de pàgina...
    ...i s'invoca la RSE

El TLB de MIPS (recordatori)

Funció del bit V del TLB

- És una còpia del bit de presència (P) de la TP
  - Indica si la pàgina està en MP o solament en disc

El TLB de MIPS (recordatori)

Funció del bit V del TLB

- És una còpia del bit de presència (P) de la TP
  - Indica si la pàgina està en MP o solament en disc
  - No intervé per determinar si és encert de TLB o no
    - L'encert sols depèn de si el VPN està en alguna entrada del TLB

El TLB de MIPS (recordatori)

Funció del bit V del TLB

- És una còpia del bit de presència (P) de la TP
  - Indica si la pàgina està en MP o solament en disc
  - No intervé per determinar si és encert de TLB o no
    - L'encert sols depèn de si el VPN està en alguna entrada del TLB
- El bit V intervé quan reemplacem una entrada del TLB
  - L'algorisme dóna preferència a ocupar entrades amb V=0

El TLB de MIPS (recordatori)

Funció del bit D del TLB

- Quan en una fallada de pàgina la copiem del disc a MP
  - S'inicialitza el bit D a la TP i al TLB (0 per lectura, 1 per escriptura)

El TLB de MIPS (recordatori)

Funció del bit D del TLB

- Quan en una fallada de pàgina la copiem del disc a MP
  - S'inicialitza el bit D a la TP i al TLB (0 per lectura, 1 per escriptura)
  - El bit D indica si la pàgina en MP s'ha modificat respecte al disc
    - Podem dir que “els stores escriuen en disc amb *escriptura retardada”*

El TLB de MIPS (recordatori)

Funció del bit D del TLB

- Quan en una fallada de pàgina la copiem del disc a MP
  - S'inicialitza el bit D a la TP i al TLB (0 per lectura, 1 per escriptura)
  - El bit D indica si la pàgina en MP s'ha modificat respecte al disc
    - Podem dir que “els stores escriuen en disc amb *escriptura retardada”*
- Si un store fa encert al TLB i D==0
  - És el 1er store a la pàgina: es posa D=1 al TLB

El TLB de MIPS (recordatori)

Funció del bit D del TLB

- Quan en una fallada de pàgina la copiem del disc a MP
  - S'inicialitza el bit D a la TP i al TLB (0 per lectura, 1 per escriptura)
  - El bit D indica si la pàgina en MP s'ha modificat respecte al disc
    - Podem dir que “els stores escriuen en disc amb *escriptura retardada”*
- Si un store fa encert al TLB i D==0
  - És el 1er store a la pàgina: es posa D=1 al TLB
  - I es genera una excepció de "pàgina modificada“
    - El SO comprova els permisos d'escriptura, i posa D=1 a la TP

El TLB de MIPS (recordatori)

Funció del bit D del TLB

- Quan en una fallada de pàgina la copiem del disc a MP
  - S'inicialitza el bit D a la TP i al TLB (0 per lectura, 1 per escriptura)
  - El bit D indica si la pàgina en MP s'ha modificat respecte al disc
    - Podem dir que “els stores escriuen en disc amb *escriptura retardada”*
- Si un store fa encert al TLB i D==0
  - És el 1er store a la pàgina: es posa D=1 al TLB
  - I es genera una excepció de "pàgina modificada“
    - El SO comprova els permisos d'escriptura, i posa D=1 a la TP
  - És a dir: el bit D al TLB i a la TP es mantenen *consistents*
    - Simplifica el posterior reemplaçament d'entrades al TLB

El TLB de MIPS (recordatori)

Funció del bit D del TLB

- Quan en una fallada de pàgina la copiem del disc a MP
  - S'inicialitza el bit D a la TP i al TLB (0 per lectura, 1 per escriptura)
  - El bit D indica si la pàgina en MP s'ha modificat respecte al disc
    - Podem dir que “els stores escriuen en disc amb *escriptura retardada”*
- Si un store fa encert al TLB i D==0
  - És el 1er store a la pàgina: es posa D=1 al TLB
  - I es genera una excepció de "pàgina modificada“
    - El SO comprova els permisos d'escriptura, i posa D=1 a la TP
  - És a dir: el bit D al TLB i a la TP es mantenen *consistents*
    - Simplifica el posterior reemplaçament d'entrades al TLB
    - Podem dir que “D s’escriu a MP amb *escriptura immediata”*
    - És l’únic bit del TLB que el programa pot modificar (posar a 1)

L'excepció de crida al sistema

- El sistema operatiu (SO) proporciona accés segur i eficient als recursos compartits
  - Memòria física (DRAM)
  - Dispositius d'E/S (Disc, SSD, USB, pantalla, teclat, xarxa, etc.)
  - Processador (execució concurrent de múltiples programes)

L'excepció de crida al sistema

- El sistema operatiu (SO) proporciona accés segur i eficient als recursos compartits
  - Memòria física (DRAM)
  - Dispositius d'E/S (Disc, SSD, USB, pantalla, teclat, xarxa, etc.)
  - Processador (execució concurrent de múltiples programes)
- El SO s'executa en mode sistema (EXL=1)
  - Li permet accedir a la *memòria protegida* (codi/dades) del SO
    - En MIPS: adreces virtuals amb bit 31 igual a 1
    - L'accés en mode usuari causa l'excepció ExcCode=AdEL o AdES

L'excepció de crida al sistema

- El sistema operatiu (SO) proporciona accés segur i eficient als recursos compartits
  - Memòria física (DRAM)
  - Dispositius d'E/S (Disc, SSD, USB, pantalla, teclat, xarxa, etc.)
  - Processador (execució concurrent de múltiples programes)
- El SO s'executa en mode sistema (EXL=1)
  - Li permet accedir a la *memòria protegida* (codi/dades) del SO
    - En MIPS: adreces virtuals amb bit 31 igual a 1
    - L'accés en mode usuari causa l'excepció ExcCode=AdEL o AdES
  - Li permet executar *instruccions privilegiades*
    - Com les que operen sobre el CP0
    - L'execució en mode usuari causa l'excepció ExcCode=CpU

L'excepció de crida al sistema

- L'accés als serveis del SO es fa amb una excepció
  - En MIPS, amb la instrucció syscall
  - L'excepció (codi ExcCode=Sys) activa el mode sistema (EXL=1)
  - I salta a la RSE, que és part del SO

L'excepció de crida al sistema

- L'accés als serveis del SO es fa amb una excepció
  - En MIPS, amb la instrucció syscall
  - L'excepció (codi ExcCode=Sys) activa el mode sistema (EXL=1)
  - I salta a la RSE, que és part del SO
- Funciona com una crida a subrutina...
  - Es posa en $v0 el codi del servei que es sol·licita
    - El SO té una subrutina per a cada servei
  - Els paràmetres, en $a0-$a3 (o $f12 i $f14 si són *float*)
  - Es fa la crida amb syscall (no pas amb jal !)
  - El valor de retorn (si n'hi ha) es passa en $v0 (o $f0 si és *float*)

L'excepció de crida al sistema

- L'accés als serveis del SO es fa amb una excepció
  - En MIPS, amb la instrucció syscall
  - L'excepció (codi ExcCode=Sys) activa el mode sistema (EXL=1)
  - I salta a la RSE, que és part del SO
- Funciona com una crida a subrutina...
  - Es posa en $v0 el codi del servei que es sol·licita
    - El SO té una subrutina per a cada servei
  - Els paràmetres, en $a0-$a3 (o $f12 i $f14 si són *float*)
  - Es fa la crida amb syscall (no pas amb jal !)
  - El valor de retorn (si n'hi ha) es passa en $v0 (o $f0 si és *float*)
- A diferència de les subrutines...
  - L'excepció salta a un punt d'entrada únic del SO (la RSE)
  - L'excepció activa el mode sistema, i inhibeix interrupcions

L'excepció de crida al sistema

- Alguns codis de servei del SO (per posar en $v0)

Servei	Codi	Paràmetres	Resultat
print_int	1	`$a0 = int`	
print_float	2	`$f12 = float`	
print_double	3	`$f12 = double`	
print_string	4	`$a0 = string`	
read_int	5		$v0 = integer
read_float	6		$f0 = float
read_double	7		$f0 = double
read_string	8	`$a0 = buffer, $a1 = length`	
sbrk	9	`$a0 = amount`	$v0 = address
exit	10		
print_char	11	`$a0 = char`	
read_char	12		$v0 = char
open	13	`$a0 = filename (string)` `$a1 = flags, $a2 = mode`	$a0 = file descriptor
read	14	`$a0 = file descriptor,` `$a1 = buffer, $a2 = length`	$a0 = num chars read
write	15	`$a0 = file descriptor,` `$a1 = buffer, $a2 = length`	$a0 = num chars written
close	16	`$a0 = file descriptor`	
exit2	17	`$a0 = result`	

L'excepció de crida al sistema

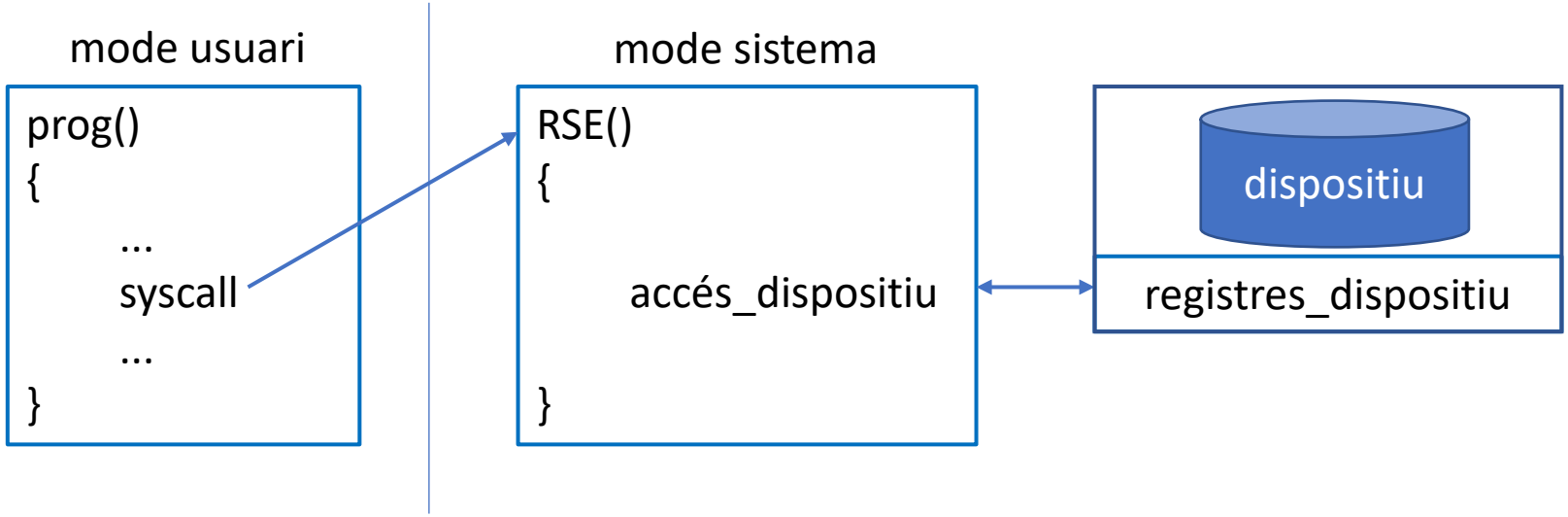
- Programa exemple que fa crides al sistema

```
        .data
cadena:     .asciiz "Introdueix una frase\n"
        .text
        ...
        li      $v0, 4          # crida num 4: print_string(char *p)
        la      $a0, cadena
        syscall

        li      $v0, 10         # crida num 10: exit()
        syscall
```

Les interrupcions d'E/S

- Els dispositius d'E/S només són accessibles per al SO
  - Els programes d'usuari hi poden accedir sol·licitant-ho al SO
  - Per mitjà d'una crida al sistema



Les interrupcions d'E/S

- Les operacions d'E/S tenen latències enormes
  - La comunicació amb la CPU requereix sincronització

Les interrupcions d'E/S

- Les operacions d'E/S tenen latències enormes
  - La comunicació amb la CPU requereix sincronització
- Sincronització per enquesta
  - El programa espera que el dispositiu estigui llest, consultant repetidament el seu estat (espera activa)

```
do {
     estat = consultar_estat_dispositiu();
while (estat != READY);

transferir_dades_dispositiu;
```

Les interrupcions d'E/S

- Les operacions d'E/S tenen latències enormes
  - La comunicació amb la CPU requereix sincronització
- Sincronització per enquesta
  - El programa espera que el dispositiu estigui llest, consultant repetidament el seu estat (espera activa)

```
do {
     estat = consultar_estat_dispositiu();
while (estat != READY);

transferir_dades_dispositiu;
```

- Sincronització per interrupcions
  - El SO executa altres programes mentre el dispositiu estigui ocupat (millor utilització de la CPU)

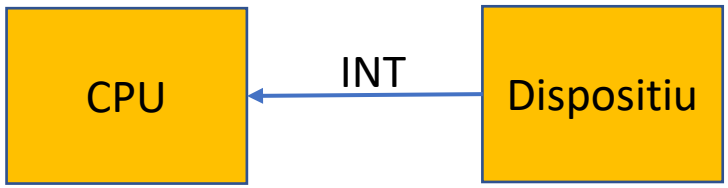
Les interrupcions d'E/S

- Les operacions d'E/S tenen latències enormes
  - La comunicació amb la CPU requereix sincronització
- Sincronització per enquesta
  - El programa espera que el dispositiu estigui llest, consultant repetidament el seu estat (espera activa)

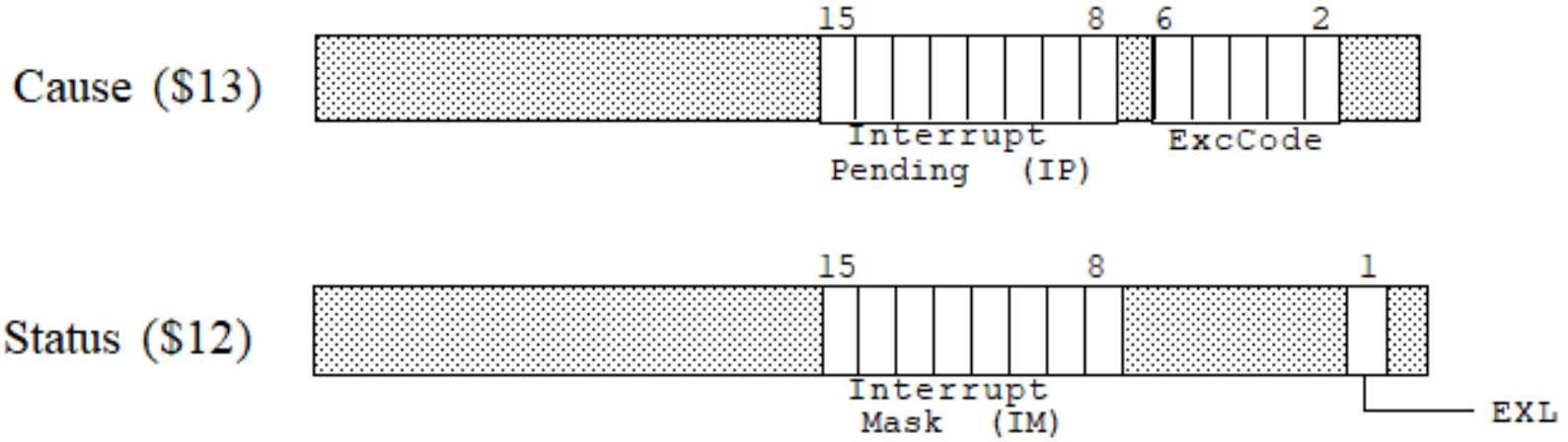
```
do {
     estat = consultar_estat_dispositiu();
while (estat != READY);

transferir_dades_dispositiu;
```

- Sincronització per interrupcions
  - El SO executa altres programes mentre el dispositiu estigui ocupat (millor utilització de la CPU)
  - Quan el dispositiu està llest, avisa a la CPU, activant un senyal de petició d'interrupció (INT)

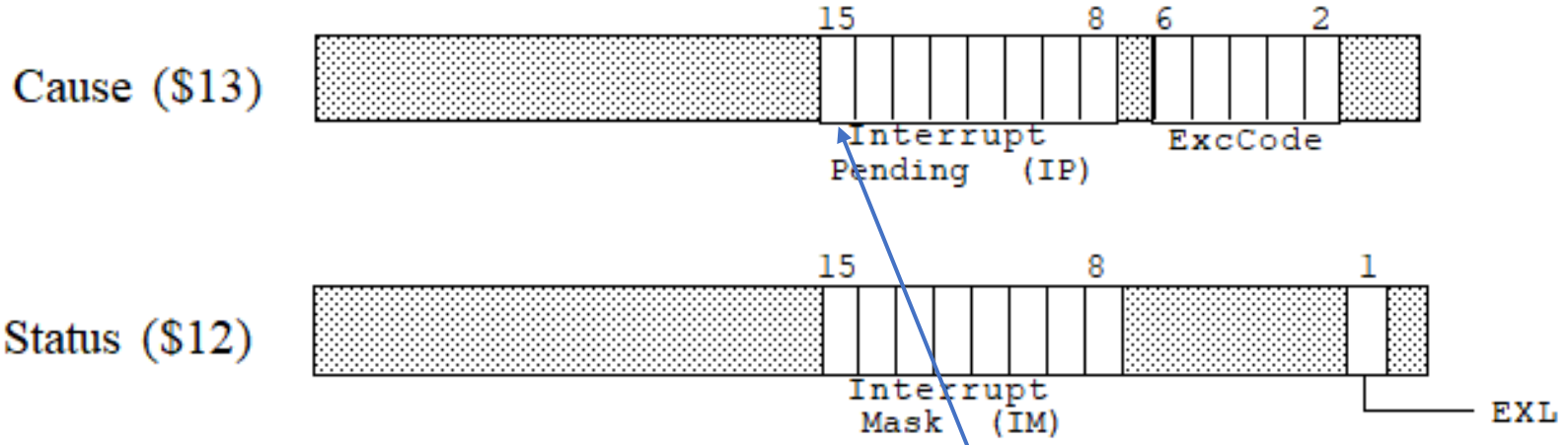


Les interrupcions d'E/S en MIPS



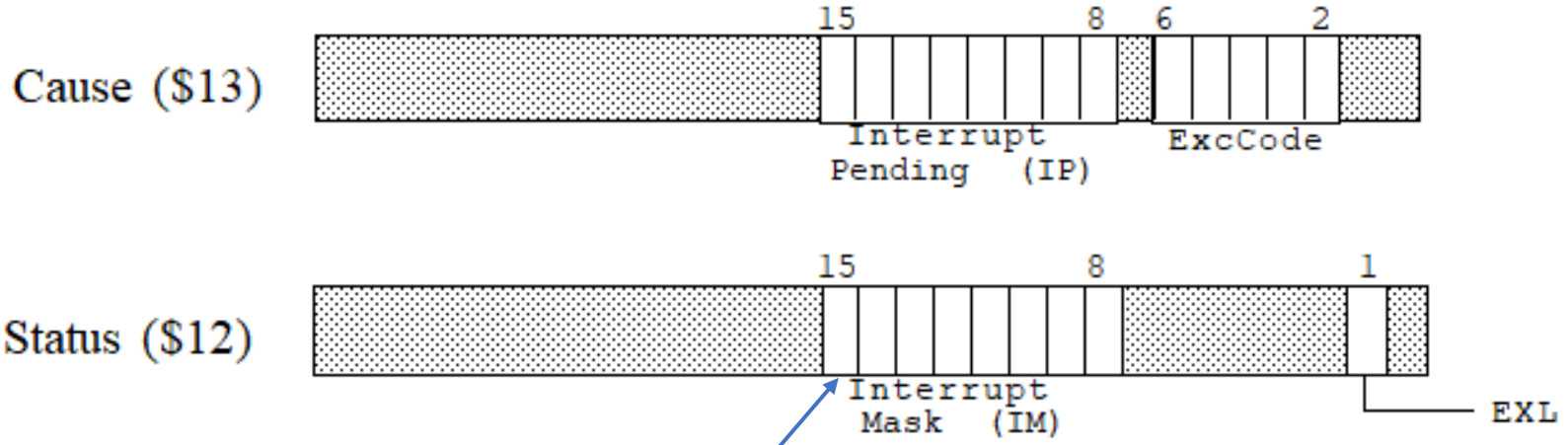
- Quan un dispositiu necessita atenció
  - Sol·licita interrupció activant el senyal INT (asíncron)
  - La instrucció en curs d'execució no es pot interrompre

Les interrupcions d'E/S en MIPS



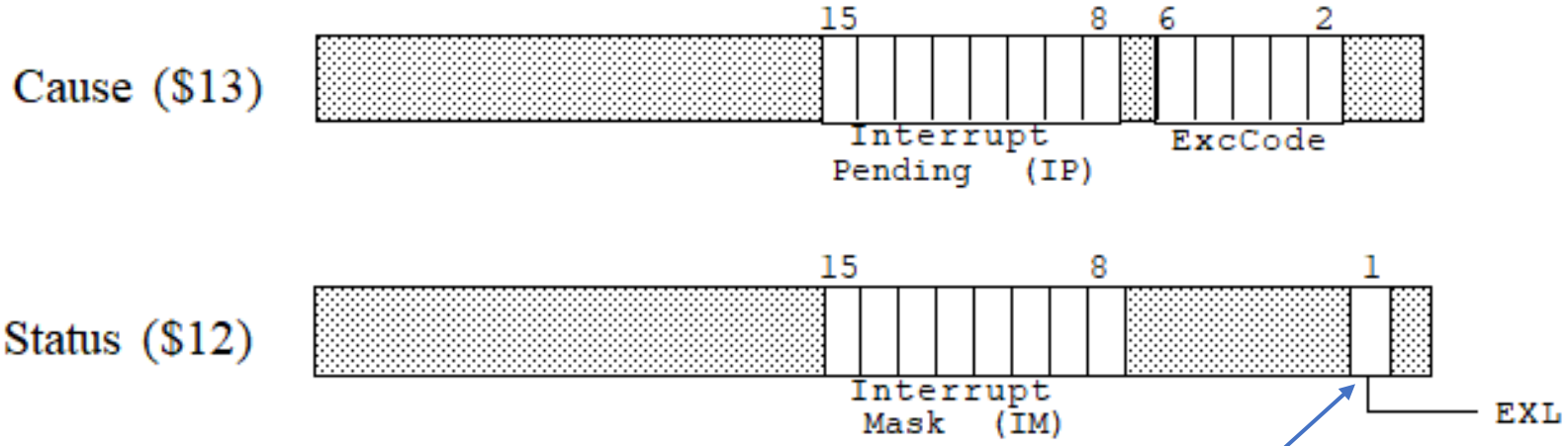
- Quan un dispositiu necessita atenció
  - Sol·licita interrupció activant el senyal INT (asíncron)
  - La instrucció en curs d'execució no es pot interrompre
  - La petició queda pendent i anotada al bit IP_i associat al dispositiu
  - Múltiples peticions poden quedar registrades al camp IP

Les interrupcions d'E/S en MIPS



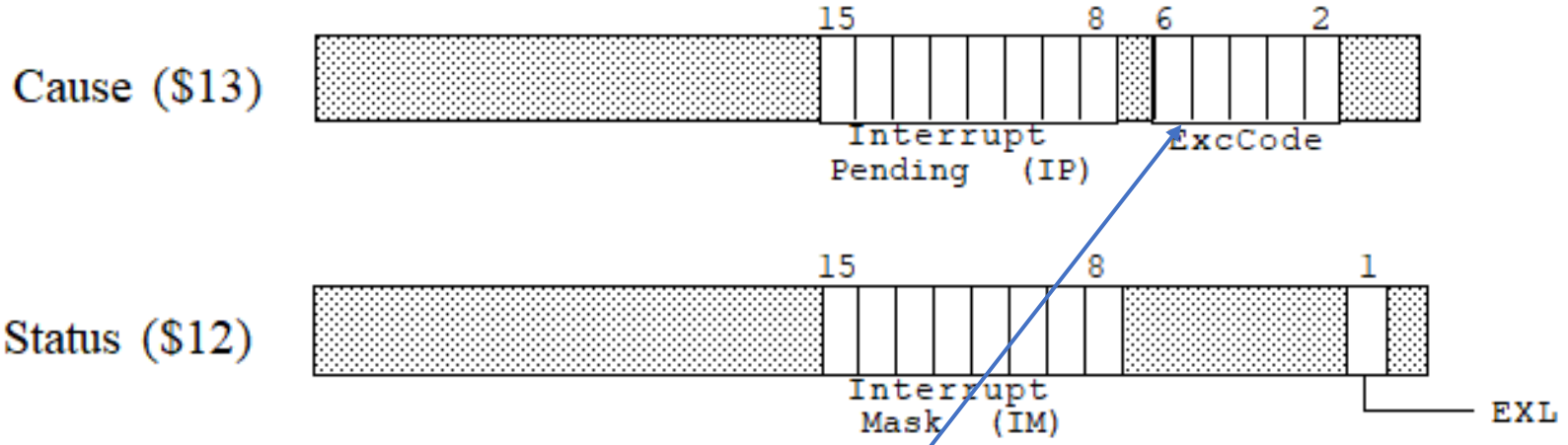
- Quan un dispositiu necessita atenció
  - Sol·licita interrupció activant el senyal INT (asíncron)
  - La instrucció en curs d'execució no es pot interrompre
  - La petició queda pendent i anotada al bit IP_i associat al dispositiu
  - Múltiples peticions poden quedar registrades al camp IP
- El SO pot deshabilitar selectivament les interrupcions
  - Posant $IM_i = 0$ es deshabiliten (s'ignoren) les peticions del dispositiu *i*

Les interrupcions d'E/S en MIPS



- Quan finalitza la instrucció en curs i s'incrementa el PC
  - Si les interrupcions estan habilitades (EXL=0), el processador comprova si hi ha peticions pendents

Les interrupcions d'E/S en MIPS



- Quan finalitza la instrucció en curs i s'incrementa el PC
  - Si les interrupcions estan habilitades (EXL=0), el processador comprova si hi ha peticions pendents
  - Si alguna petició està pendent IP_i=1 i habilitada IM_i=1, es genera una excepció (ExcCode=Int)

Les interrupcions d'E/S en MIPS

- Quan s'invoca la RSE
  - Es posa el bit EXL=1
    - La rutina s'executa en mode sistema
    - Mentre EXL=1, queden deshabilitades TOTES les interrupcions

Les interrupcions d'E/S en MIPS

- Quan s'invoca la RSE
  - Es posa el bit EXL=1
    - La rutina s'executa en mode sistema
    - Mentre EXL=1, queden deshabilitades TOTES les interrupcions
  - La RSE comprova els bits IP i IM per identificar el dispositiu
  - Si hi ha múltiples peticions, estableix un sistema de prioritats

Les interrupcions d'E/S en MIPS

- Quan s'invoca la RSE
  - Es posa el bit EXL=1
    - La rutina s'executa en mode sistema
    - Mentre EXL=1, queden deshabilitades TOTES les interrupcions
  - La RSE comprova els bits IP i IM per identificar el dispositiu
  - Si hi ha múltiples peticions, estableix un sistema de prioritats

- Sincronització
  - La CPU notifica "assabentat" al dispositiu *i* (senyal INTA)
  - El dispositiu *i* desactiva la petició (senyal INT)
  - El bit IP_i es desactiva