***Pràctica 4: Introducció al RISC 5-Stage Processor en Ripes***
(1 sessió)

**Objectiu**
L'objectiu de la pràctica és visualitzar els diferents passos que ha de fer un microprocessador per tal d'executar una instrucció.

**Introducció**
El número de cicles dividit per la freqüència de funcionament del processador ens defineix el temps d'execució de la instrucció. Les diferents accions que es realitzen en cada cicle formen el que es coneixen com microinstruccions.

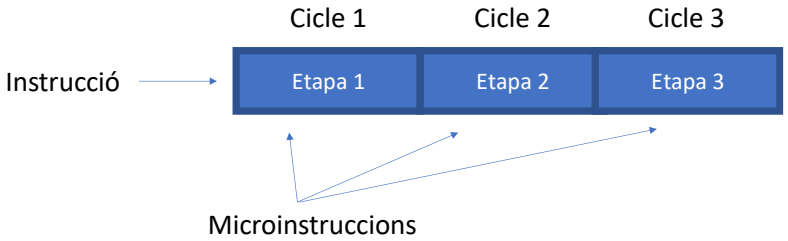



Figura 1. Concepte de microistrucció

Com ja vàrem veure a l'apartat *Estructura bàsica de la CPU* d'aquest guió de pràctiques de l'assignatura, hi ha processadors que han d'executar totes les etapes d'una instrucció abans d'executar la següent, però també n'hi ha d'altres que a costa de multiplicar recursos de hardware poden executar diferents etapes d'una instrucció com a una cadena de muntage, o *pipeline*.

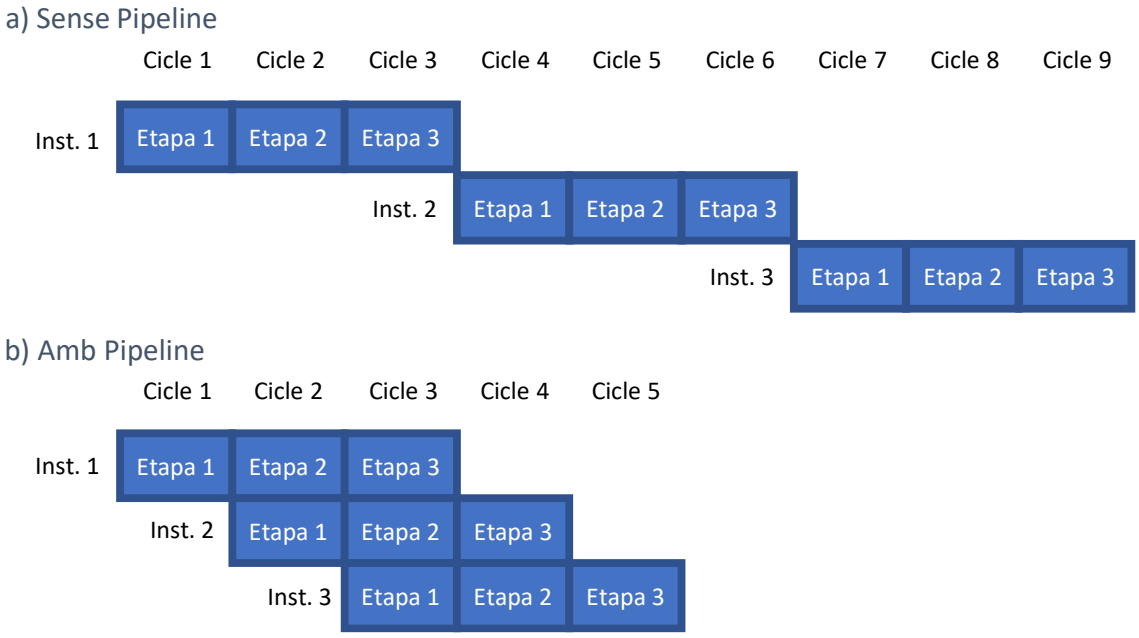



Figura 2. Cronograma d'execució d'instruccions en un processador a) sense *pipeline* i b) amb *pipeline*. El temps avança seguint un eix imaginari horitzontal de esquerra a dreta. Pel que fa a les instruccions, aquestes s'incrementen seguint un eix imaginari vertical que va de dalt a baix.

En ciència de computadors, el *pipeline* RISC es una solució de microarquitectura que té com a propòsit la execució d'una instrucció per circle de rellotge. Aquest *pipeline* executa una instrucció en cinc etapes: Instruction Fetch, Instruction Decode, Execute, Memory Access i Writeback. A continuació s'expliquen les accions que realitza el processador en cadascuna d'aquestes etapes:

- **Instruction Fetch (IF)**: LLegiex la següent instrucció de memòria i incrementa el comptador de programa.
- **Instruction Decode (ID)**: Decodifica la instrucció. Llegeix els registres d'operants i computa direccions de salt (si escau).
- **Execute (EX)**: Executa una operació aritmètico-lògica o realiza un salt.
- **Memory Access (MEM)**: Realitza accessos a memòria de lectura o escriptura.
- **Writeback (WB)**: Escriu els results de les operacions a un registre.

Alguns dels processadors que comparteixen aquesta microarquitectura són: MIPS (Microprocessor without Interlocked Pipelined Stages), SPARC (Scalable Processor Architecture), Motorola 88000 (o m88k) i, és clar, el RISC-V (Reduced Instruction Set Computer). A la següent figura podem veure el típic pipeline de cinc etapes d'una màquina RISC.

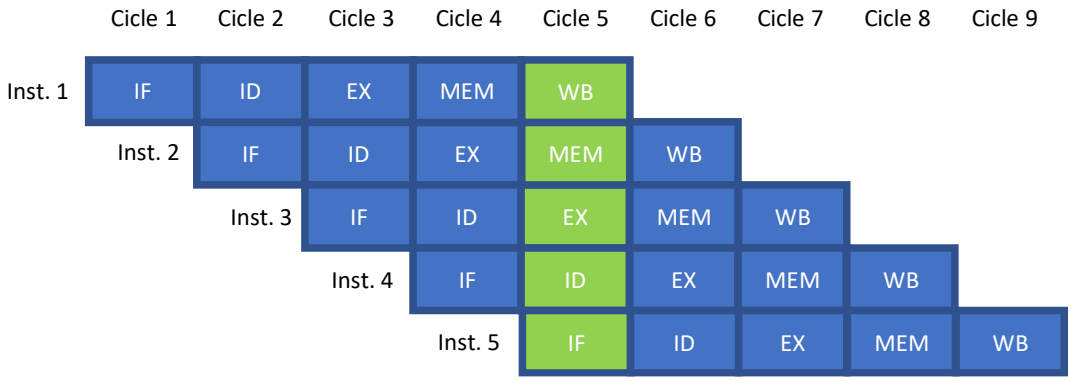



Figura 3. Pipeline de cinc etapes bàsic a una màquina RISC. Veiem que al cinquè cicle de rellotge (columna verda) la primera instrucció s'acaba d'executar, la segona instrucció es troba a l'etapa IF, la tercera a EX, la quarta a ID, i la cinquena a IF. Noteu que si no hi hagués esperes (stall(s) deguts a un hazard), a partir del cinquè cicle cada instrucció s'executaria en un cicle de rellotge.

**Exercici guiat**

Haurem d'executar els següents codis, cicle a cicle (microinstrucció a microinstrucció) utilitzant com a micro el *5-Stage Processor* al simulador Ripes:

**Codi 1**

```
.data
resultat: .word 0
.text
main:
add  a3, zero, zero
add  a7, zero, zero
addi a2, zero, 4
add  a3, a2, a3
addi a7, a7, -1
la   a0, resultat
sw   a3, 0(a0)
```

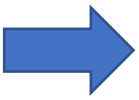
**Codi 2**

```
.data
resultat: .word 0
.text
main:
add  a3, zero, zero
add  a7, zero, zero
addi a2, zero, 4
add  a3, a2, a3
addi a7, a7, -1
blt  a7, zero, salta
salta:
la   a0, resultat
sw   a3, 0(a0)
```

1) Abans d'executar els codis tracta d'esbrinar la seva funcionalitat. Faran el mateix? Creus què trigaran el mateix nombre de cicles en executar-se?
2) Ves a la finestra del Ripes dedicada al procesador (Processor). Busca entre les opcions del Simulator control, la *Pipeline table*.
3) Executa els dos codis cicle a cicle fixant-te com les intruccions van passant per les diferents etapes del pipeline. Compta els nombre de cicles que es necesiten per executar cadascun dels codis i compara les Pipeline tables.
4) Perquè les etapes de la instrucció **auipc x10 0x 10000** al pipeline son **IF ID IF ID EX MEM WB** al Codi 2?
5) Com afectaria al nombre total de cicles d'execució el següent canvi en el codi:

**Codi 1**

```
.data
resultat: .word 0
.text
main:
add  a3,  zero,  zero
add  a7,  zero,  zero
addi  a2,  zero,  4
add   a3,  a2,  a3
addi   a7,  a7,  -1
la   a0, resultat
sw  a3, 0(a0)
```



**Codi 3**

```
.data
resultat: .word 0
.text
main:
add   a3,  zero,  zero
add   a7,  zero,  zero
addi  a2,  zero,  4
add   a3,  a2,  a3
addi  a7,  a7,  -1
bgt    a7,  zero,  salta
salta:
la     a0,  resultat
sw    a3,  0(a0)
```

**Realització de la pràctica (Cal entregar)**
***Ripes*:** Executa el següent programa microinstrucció a microinstrucció:

**Codi 4**

```
.data
valorDada: .word 2
guardaResultat: .word 0
.text
main:
la   a0, valorDada
lw  a7,  0(a0)
addi  a2,  zero,  9
add  a3,  zero,  zero
loop:
add  a3,  a2,  a3
addi  a7, a7,  -1
bgt  a7,  zero,  loop
la   a0,  guardaResultat
sw  a3,  0(a0)
```

**Informe**

Explica detalladament la pràctica realitzada. Fes els diagrames necessaris per entendre i mostrar el cicle d'execució dels diferents tipus d'instruccions al simulador. L'informe ha de constar d'uns apartats d'objectius i conclusions. Cada qüestió del següent apartat ben resolta val 1 punt a l'informe. Presentació, estructura i claretat de les explicacions valen el punt que resta fins arribar a 10.

**Preguntes sobre el simulador RIPES:**

Per resoldre aquestes qüestions és necessari mirar l'estat del pipeline o utilizar la *Pipeline table*:

1) Quin es l'estat de casdascuna de les cinc etapes del pipeline al cicle 6? I al 8?
2) Quins senyals de control s'activen en el cicle 4 a la segona etapa del pipeline? A quines instruccions del codi corresponen?
3) Quins són els valors a les sortides dels multiplexors assenyalats a la figura al cicle 9:

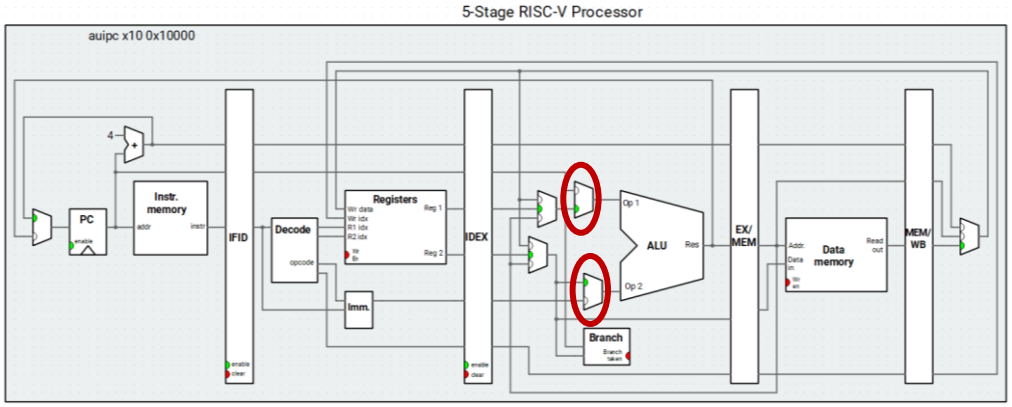



Figura 4. Detall de les etapes del RISC-V en el simudor RIPES.
Les el·lipses vermelles assenyalen els multiplexors que s'utilitzen a la qüestió 3.

4) Què està calculant la ALU al cicle 9?
5) Llegeix amb cura la part del codi amb que s'implementa el loop. Tenint en compte el que has vist a la qüestió 3), justifica perquè el pipeline al cicle 10 presenta aquest estat:

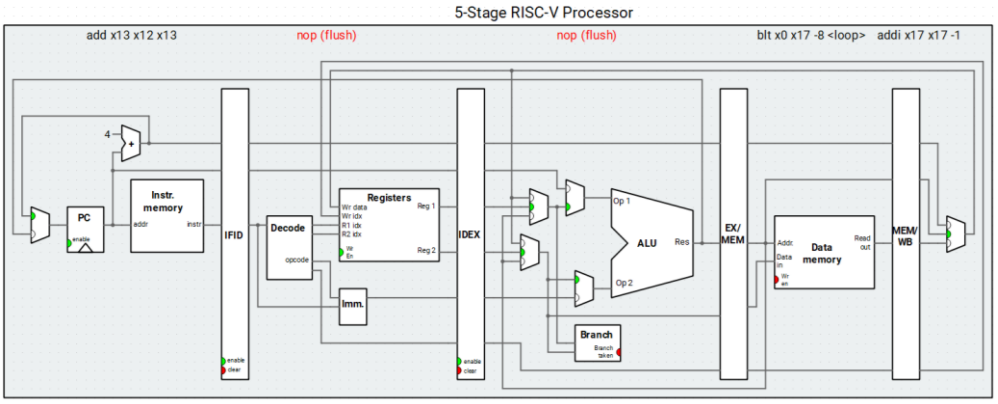



Figura 5. Estat del pipeline al cicle 10.

6) Quan NO es produeix el salt, quantes etapes de la instrucció **auipc x10 0x 10000** s'executen al pipeline?
7) Quan s'està executant la instrucció de salt (etapa EX), però el salt NO es produeix, a quina posició apunta el program counter (PC)?
8) Quan es produeix el salt, quantes etapes de la instrucció **auipc x10 0x 10000** s'executen al pipeline?
9) Quan s'està executant la instrucción de salt (etapa EX), i el salt es produeix, a quina posició apunta el program counter (PC)?