Master 1 - SEM

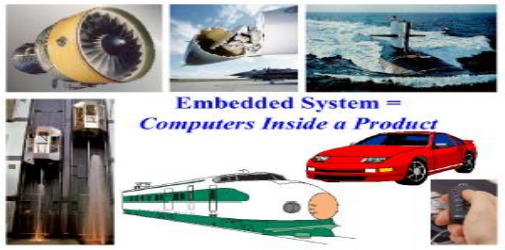
Processeurs embarqués

Cours 5.1

Pipeline

Année 2020-2021

Pr. R. BOUDOUR



Performance de processeur

Pour améliorer la rapidité de calcul d'un processeur, plusieurs dispositifs sont mis en œuvre :

- intégrer des pipelines ;
- intégrer plusieurs unités de calcul ;
- augmenter la taille des opérations ;
- intégrer de la mémoire cache ;
- réduire le jeu d'instructions ;
- intégrer un coprocesseur.

Contrairement à ce que l'on pourrait penser, la simple augmentation de la fréquence d'horloge du processeur ne suffit pas !

L'architecture 32 bits actuelle est progressivement supplantée par une nouvelle architecture dite « 64 bits ».

Performance de processeur

- L'ensemble des améliorations des microprocesseurs visent à diminuer le temps d'exécution du programme :
  - La première idée qui vient à l'esprit est d'augmenter tout simplement la fréquence de l'horloge du microprocesseur. Mais l'accélération des fréquences provoque un surcroît de consommation ce qui entraine une élévation de température. On est alors amené à équiper les processeurs de systèmes de refroidissement ou à diminuer la tension d'alimentation.
  - Une autre possibilité d'augmenter la puissance de traitement d'un microprocesseur est de diminuer le nombre moyen de cycles d'horloge nécessaire à l'exécution d'une instruction.
    - Dans le cas d'une programmation en langage de haut niveau, cette amélioration peut se faire en optimisant le **compilateur**. Il faut qu'il soit capable de sélectionner les séquences d'instructions minimisant le nombre moyen de cycles par instruction.
    - Une autre solution est d'utiliser une **architecture** de microprocesseur qui réduise le nombre de cycles par instruction.

Modes de travail

Sequential

Parallel

Pipelined

Mode de travail : Piplining

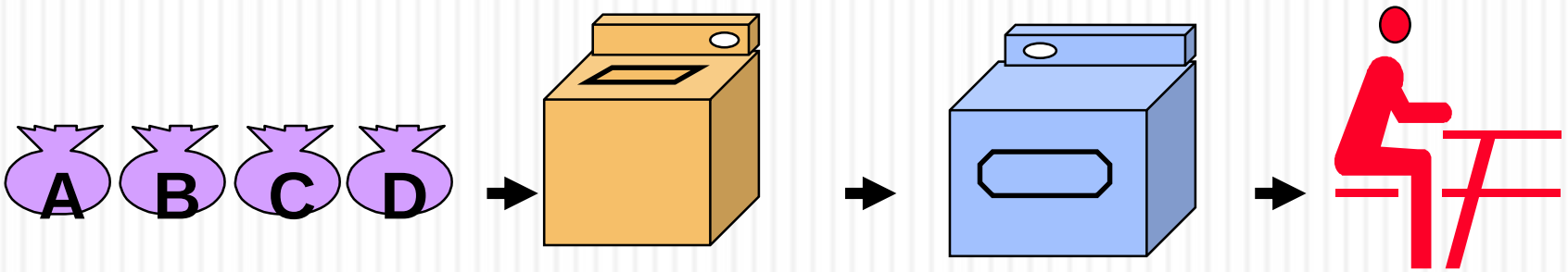
Auto

Cola

Exemple 1 : Problème de la lessive

- Le problème de la lessive !
  - 4 personnes ont un paquet de linge à laver, sécher et repasser.
  - Une machine à laver (durée : 30 minutes)
  - Un sèche linge (durée : 40 minutes)
  - Une planche à repasser (durée : 20 minutes)
    - durée d'un lavage séquentiel : 6 heures
    - si on utilise une machine dès qu'elle est libre : 3.5 heures !

Exemple 2 : Problème de la lessive



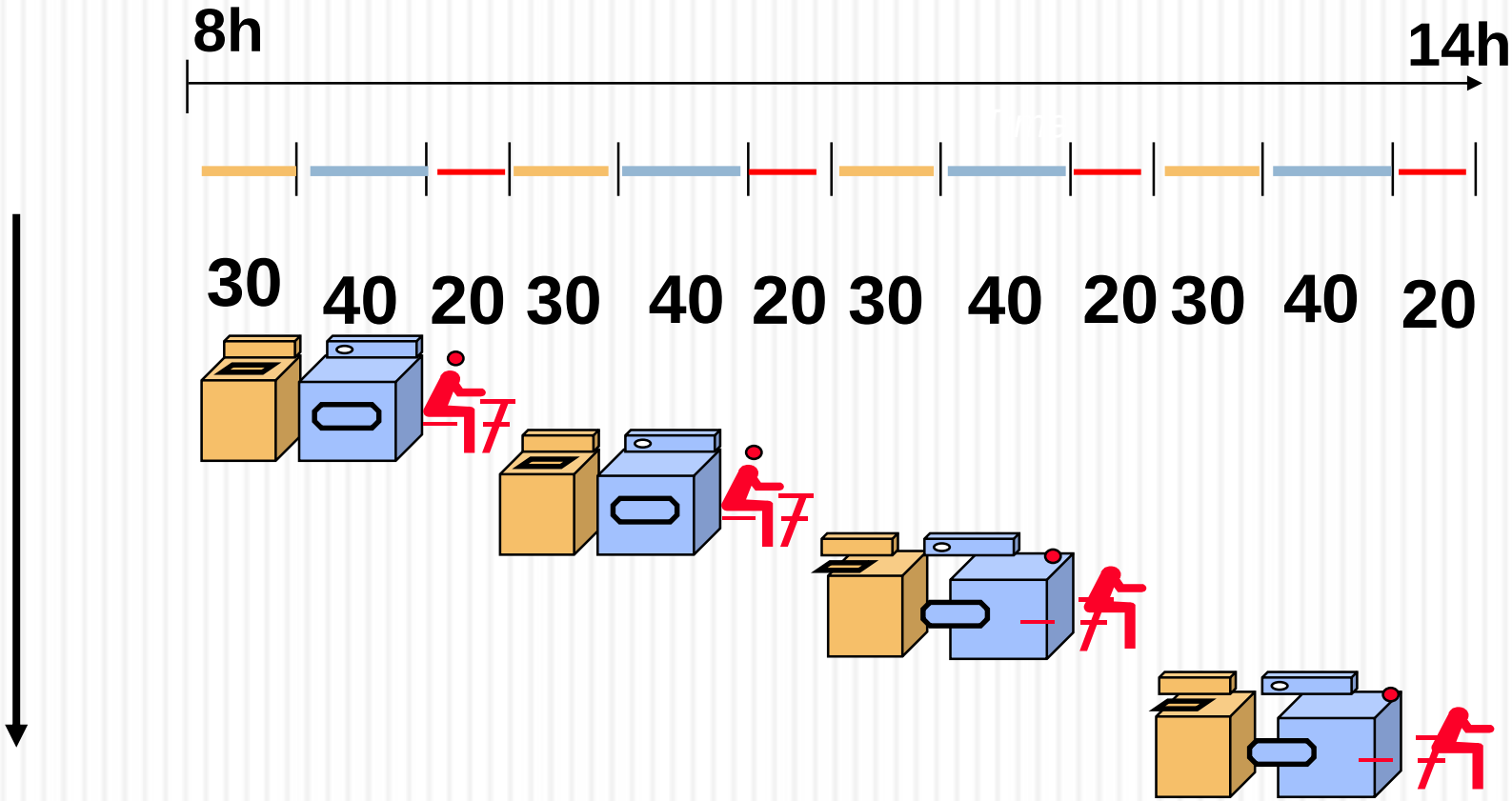
4 Paquets

Laver
30 mn

Essorer
40 mn

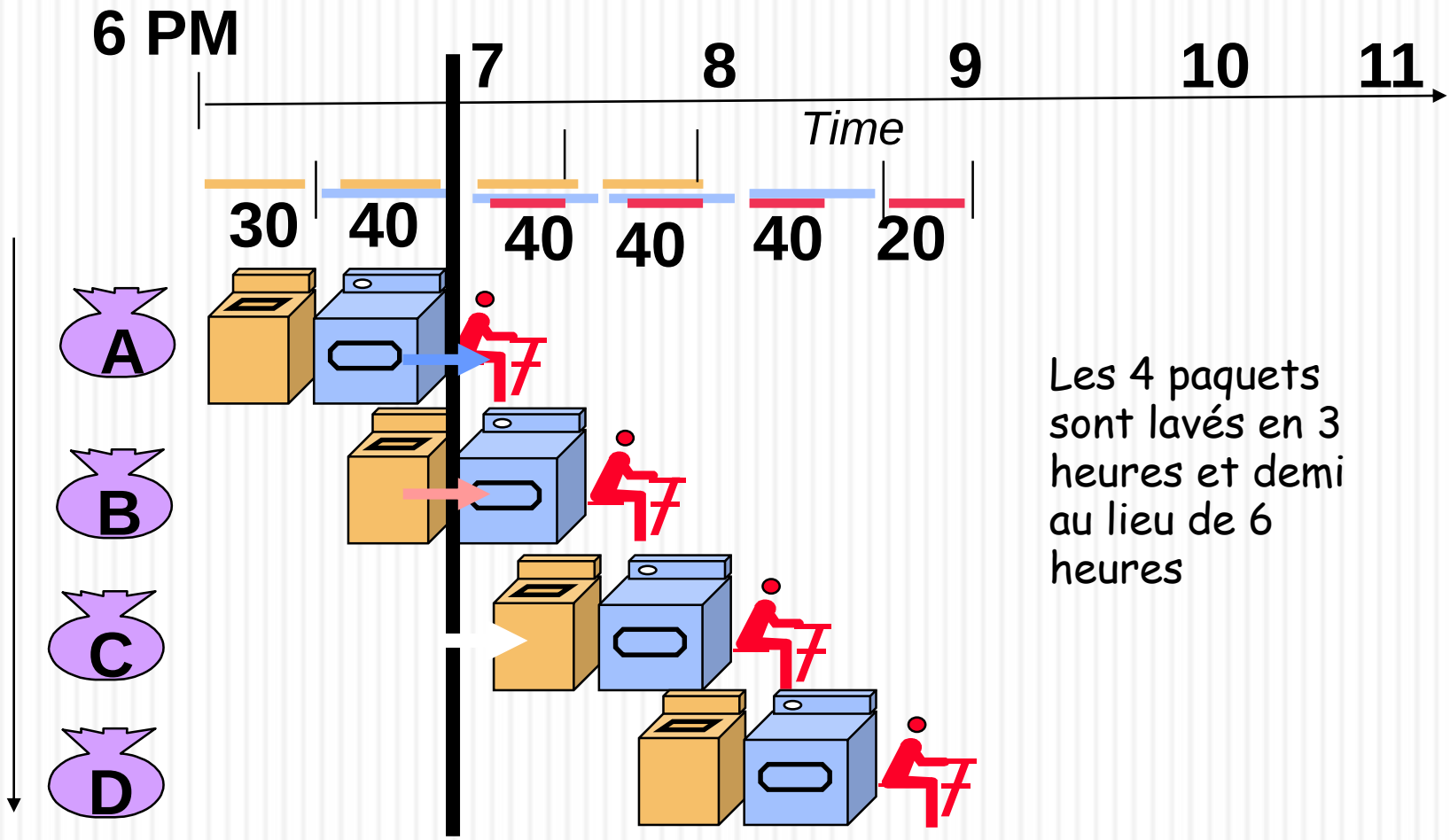
Plier
20 mn

Solution naïve : Un paquet à la fois



6 heures pour laver les 4 paquets

Laver les vêtements en pipeline



Les 4 paquets sont lavés en 3 heures et demi au lieu de 6 heures

Définition du pipeline

- Technique utilisée pour optimiser le temps d'exécution d'une opération répétitive.
  - **Principes**
    - Lancer le traitement d'une opération avant que la précédente ne soit terminée ⇒ recouvrement des opérations.
    - A chaque étape de l'opération est affectée une ressource indépendante, de façon à pouvoir exécuter plusieurs opérations en parallèle, chacune à une étape différente
      ⇒ exploite le parallélisme entre les opérations d'un flot d'opérations séquentielles.
    - Optimiser le temps d'utilisation des différents éléments
  - **Découpage des opérations en sous-parties élémentaires**
    - En relation avec les étapes de traitement de l'opération.
    - Définition des étages du pipeline.
    - Ttravail à la chaine.

Où se situe le gain du pipeline ?

- Le gain se situe au niveau du débit
  - Le temps de traitement d'une instruction n'est pas réduit
  - Il est même souvent augmenté
    - Gestion du passage entre les étages
    - Temps de stabilisation

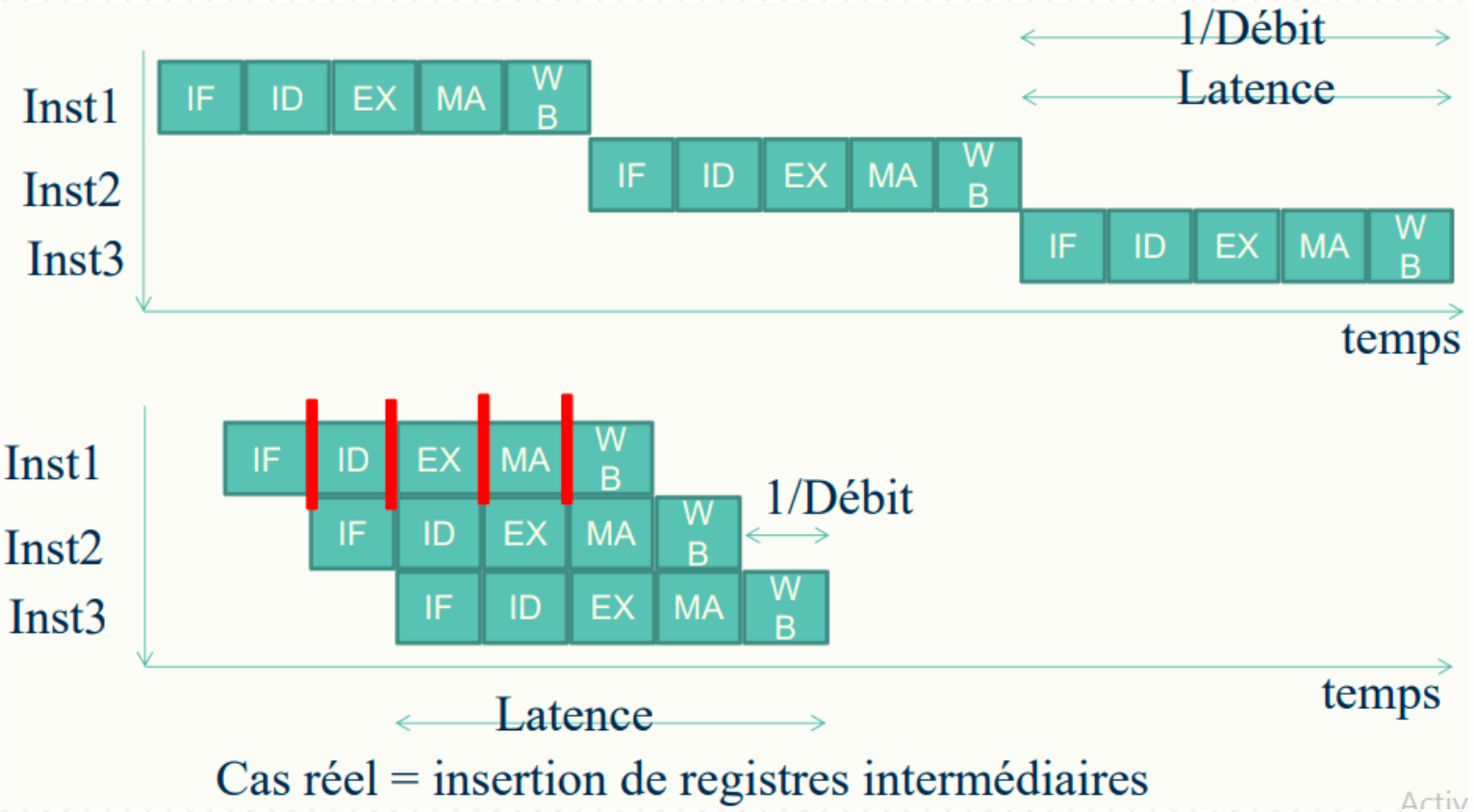
Terminologie

- **Temps de cycle** (Cycle time)
  - Temps de traitement de chaque étape
  - il est égal à une ou plusieurs périodes d'horloge
- **Latence** (Flowthrough time)
  - Temps de latence, d'attente du premier résultat
  - (n cycles)
- **Débit** (Pipeline throughput)
  - Nombre d'instructions par seconde traitées dans le pipeline
  - typiquement égal à 1/ temps de cycle

Performances du pipeline

- Deux paramètres sont souvent utilisé pour mesurer la performance d'un processeur:
  - La **latence**: Le temps de réponse ou temps d'exécution d'une certaine tâche:
    - Temps écoulé entre le début et la fin d'exécution de la tâche (instruction).
  - Le **débit**: quantité totale de travail réalisé dans un certain emps.
    - Nombre d'opérations (tâches, instructions) exécutées par unités de temps.
- L'amélioration du temps de réponse implique toujours une amélioration du débit. Le contraire n'est pas toujours vrai.
  - Par exemple, augmenter le nombre de processeurs augmentera le débit mais pas le temps d'exécution.
- **L'accélération**: nombre de fois plus vite qu'en séquentiel.

Performances du pipeline



Performances du pipeline

- Calcul du **temps total d'exécution des instructions**:
  - Si T_e , le temps d'exécution d'un élément,
  - n le nombre d'éléments que compose une instruction
  - T_p , le temps d'exécution d'une instruction
  - m le nombre d'instruction à effectuer
  - Et T_t , le temps total

- **Séquentiel**

  $\Rightarrow T_t = m * T_p$, avec $T_p = n * T_e$

  $\Rightarrow \boldsymbol{T_t = m * n * T_e}$

- **Pipeline**

  $\Rightarrow T_t = T_p + (m-1) * T_e$ Fin de la première instruction à $T_p = n * T_e$

  $\Rightarrow T_t = n * T_e + (m-1) * T_e$ Toutes les unités de temps suivantes

  $\Rightarrow \boldsymbol{T_t = T_e * (n + m - 1)}$ => fin d'une nouvelle instruction.

Si m est beaucoup plus grand que n, alors on peut considérer $\boldsymbol{T_t \approx m * T_e}$

Performances du pipeline

- Calcul de **l'accélération**:

$$A = T_{seq}/T_{pip}$$

$$A = \frac{m*n*T_e}{(n+m-1)*T_e}$$

$$\boldsymbol{A = \frac{m*n}{n+m-1}}$$

Si m très grand alors $\boldsymbol{A = n}$

$\Rightarrow$ **cas idéal,** $\boldsymbol{T_{pip} = \frac{T_{seq}}{profondeur\ du\ pipeline}}$

- Calcul du **débit** du pipeline
  - $D = \frac{1}{T_{exi}}$, avec $T_{exi} = \frac{T_t}{m}$, temps exécution total ramené à 1 instruction
  - $D = \frac{m}{(n+m-1)*T_e}$
  - $\boldsymbol{D \approx \frac{1}{T_e}}$

$\Rightarrow \boldsymbol{\frac{1}{T_e}}$ **est souvent appelé le cycle machine**

Accélération du pipeline

- **Dans un cas idéal**
  - L'accélération est donée par le nombre d'étages du pipeline
  - Temps moyen instruction$_{séq}$ / Temps moyen instruction$_{pip}$
- Le pipeline **peut diminuer** selon les approches adoptées :
  - Le nombre de cycles par instruction (CPI)
  - Le temps de cycle
  - une combinaison des deux

Exemples de profondeurs de pipeline

Quelques profondeurs de pipeline

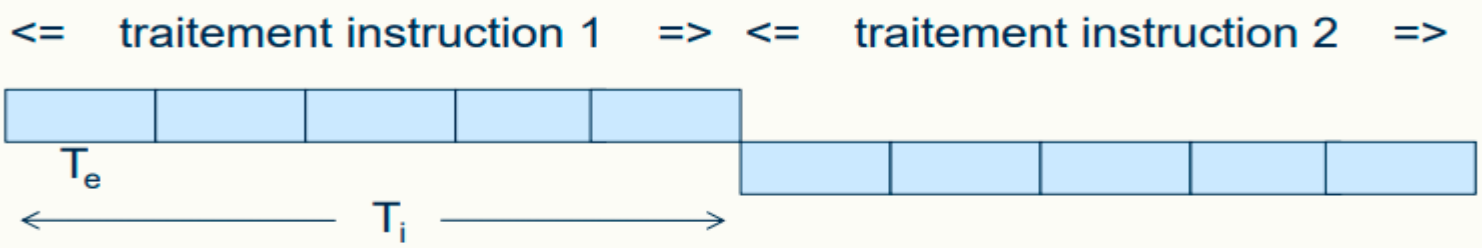
Aujourd'hui, tous les microprocesseurs sont pipelinés

Processeur	Profondeur du pipeline
Intel Pentium 4 Prescott	31
Intel Pentium 4	20
AMD K10	16
Intel Core 2 Duo	14
Intel Pentium II	14
AMD Opteron 1xx	12
Intel Pentium III	10
AMD ATHLON	10
Power PC G4(PPC 7450)	7
IBM Power4	12
IBM Power5	16
IBM PowerPC 970	16
Sun UltraSparc III et UltraSparc IV	14
Intel Itanium	10
MIPS R4400	8

Traitement d'une instruction

- Le traitement d'une instruction s'effectue généralement en plusieurs étapes

- Exemple sans pipeline



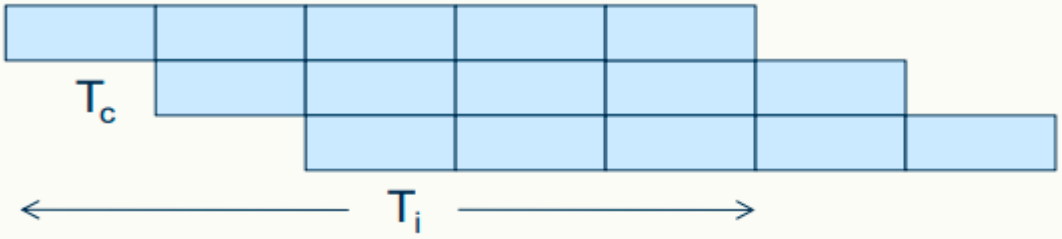
- soit n le nombre d'étapes par instruction
- et Te le temps de traitement d'un étape $T_i = nT_e$
- pour m instructions $T_t = m\,T_i = mnT_e$

Traitement d'une instruction

- **Exemple avec pipeline**

<= traitement instruction 1 =>

<= traitement instruction 2 =>



- **pour m instructions**

  $T_t = T_i + (m-1)T_e = nT_e + (m-1)T_e = (n+m-1)T_e$

- **pour un grand nombre d'instructions**

  $T_t = m\ T_e$ soit n fois moins que le traitement sans pipeline

Traitement des instructions

- Suivant les types de processeurs :
  - **Pipeline à 3 niveaux :**
    - Fetch, decode , execute
  - **Pipeline à 5 niveaux** (processeurs ARM 9, MIPS) :
    - IF, ID, EX, MEM, WB
  - **Pipeline à 20 niveaux** ! (20 niveaux pour le Pentium 4)

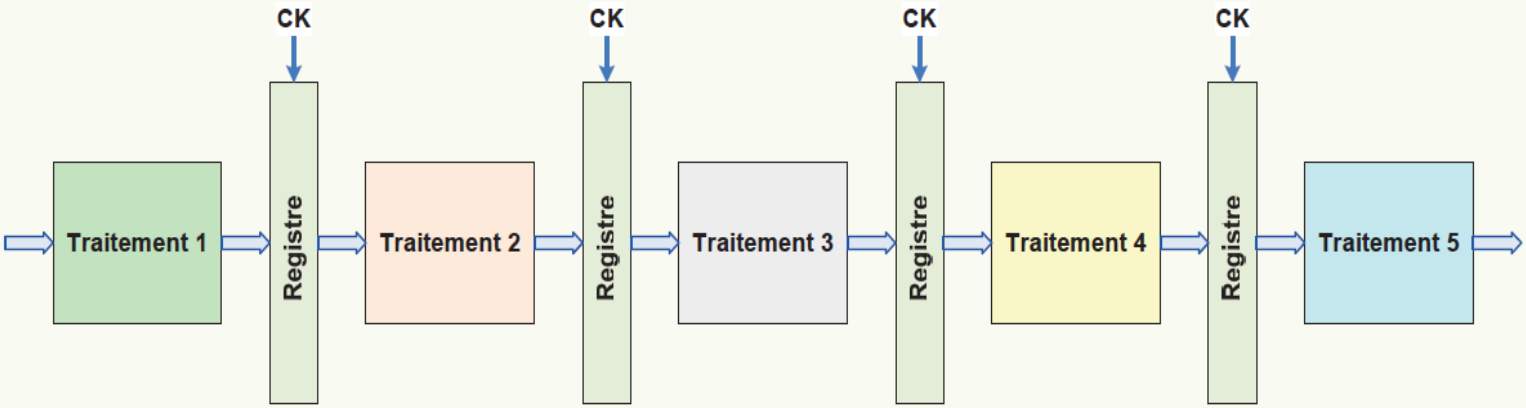
Pipeline à 5 étages

- **Phase IF** (Instruction Fetch) : recherche de l'instruction
- **Phase ID** (Instruction Decode): décodage de l'instruction et lecture des registres opérandes
- **Phase EX** (Execution): exécution de l'opération ou calcul de l'adresse de mémoire
- **Phase MEM** (Memory): accès de la mémoire ou écriture dans le PC de l'adresse de saut
- **Phase WB** (Write Back): écriture dans un registre du résultat de l'opération

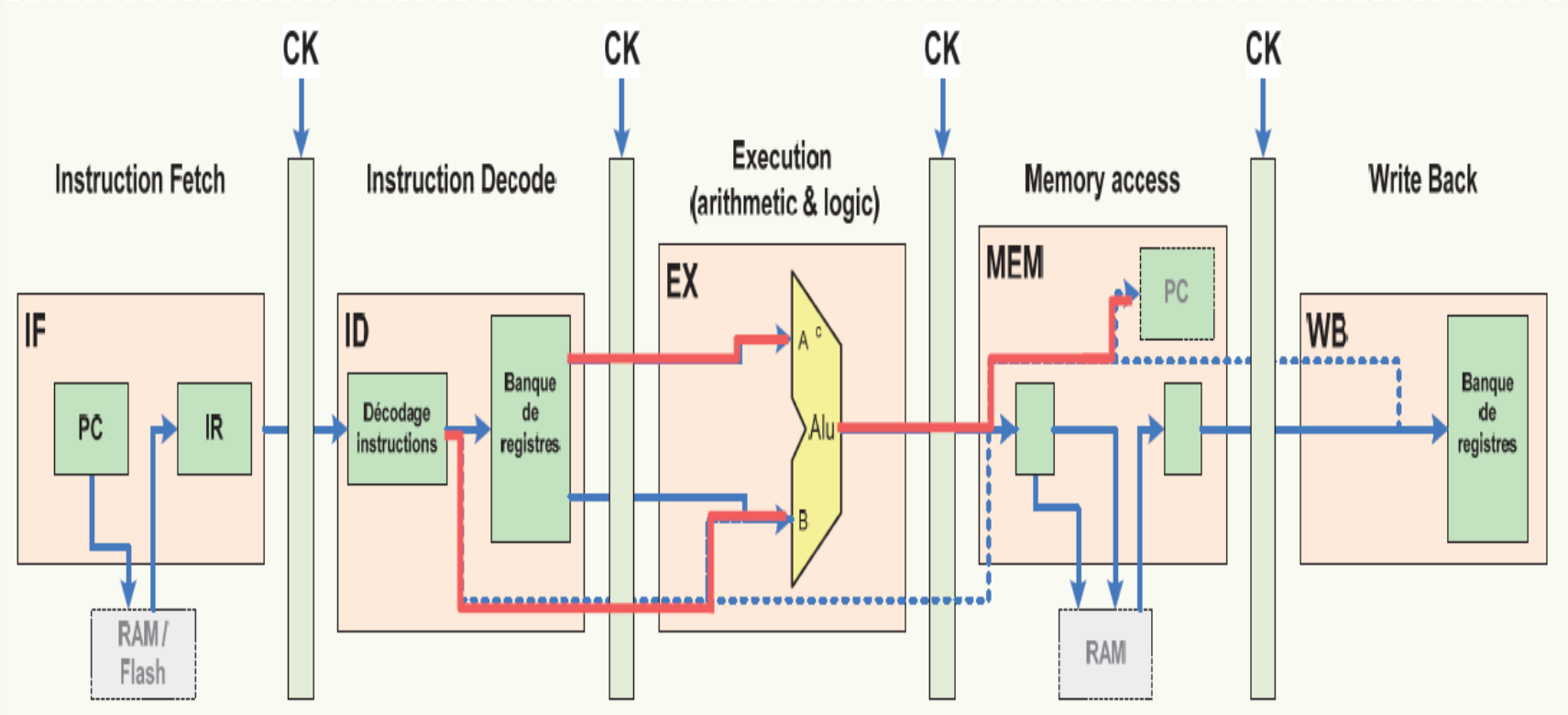
Exemple de processeur pipeliné

- Exemple de pipeline à 5 étages ou niveaux

- Le traitement s'effectue en 5 étapes (ou 5 cycles)
- Chaque étage est isolé par un registre
- Temps de cycle identique pour chaque étape (ou phase)
- Les traitements 1, 2, 3, 4 et 5 s'effectuent en parallèle



Exemple de processeur pipeliné



Etage et cycle

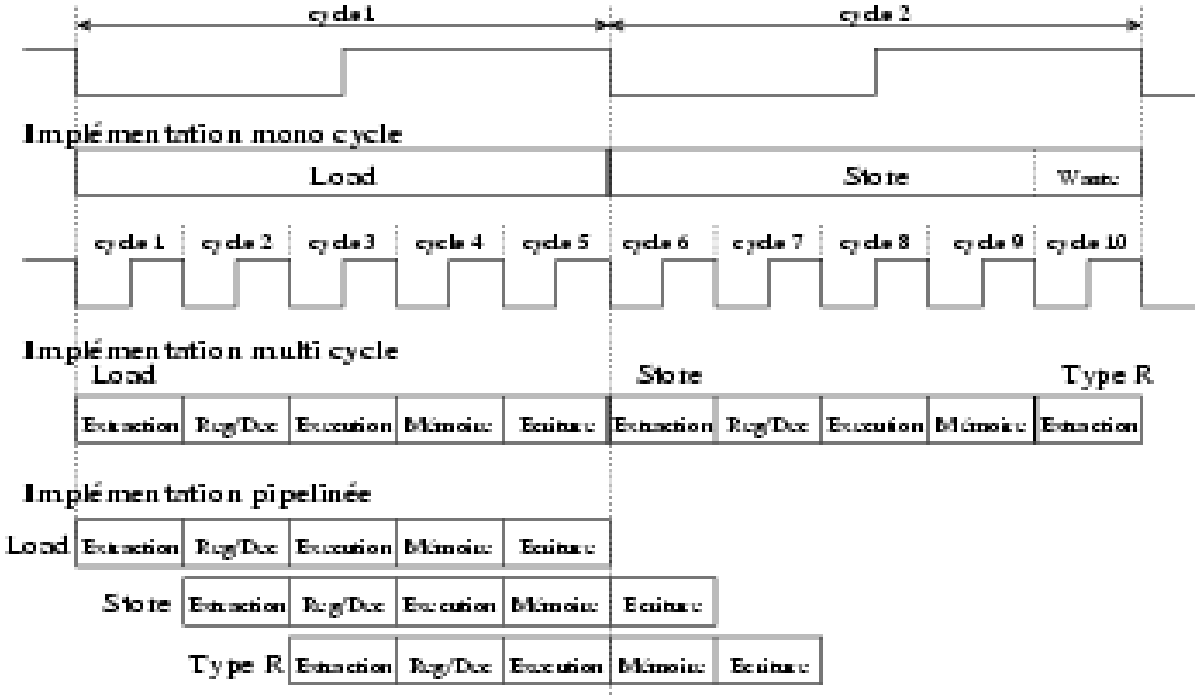
- Le temps passé par une instruction dans un étage est appelé **temps de cycle**
  - La longueur d'un cycle est déterminée par l'étage le plus lent
  - Le temps de cycle est égal à un cycle horloge, parfois deux
- Une bonne conception équilibre la longueur des étages du pipeline
  - Sinon les étages les plus rapides attendent les plus lents
  - Ce n'est pas optimal
- Insertion de registres intermédiaires entre étages
  - appelés Registres pipelines

A quoi sert un registre pipeline ?

- Il est nécessaire pour réaliser un pipeline
- Il prévient l'interférence entre instructions à différents étages du pipeline
- Il permet une isolation des instructions
- Les noms des registres :
  - IF/ID
  - ID/EX
  - EX/MEM
  - MEM/WB

Différences

- Monocycle, Multicycle et Pipeline



Différences

- **Performance Monocycle, Pipeline**
  - M1 : monocycle, temps de cycle : $10 + 8 + 10 + 10 + 7 = 45\text{ns}$;
  - M2 : pipeline, temps de cycle 11 ns (10ns (temps le plus long) + surcoût de 1ns pour le pipeline)

$$Acceleration = \frac{Temps\ sur\ M1}{Temps\ moyen\ sur\ M2} = \frac{45ns}{11ns} = 4,1 fois$$

Différences

- Performance Multicycle, Pipeline

  - M1 : multicycle, temps de cycle : 10ns ;
    - 4 cycles pour les types R (40%) et les branchements(20%), 5 cycles pour les accès mémoire (40%)
  - M2 : pipeline, temps de cycle 10 + 1 ns (surcoût de 1ns pour le pipeline)

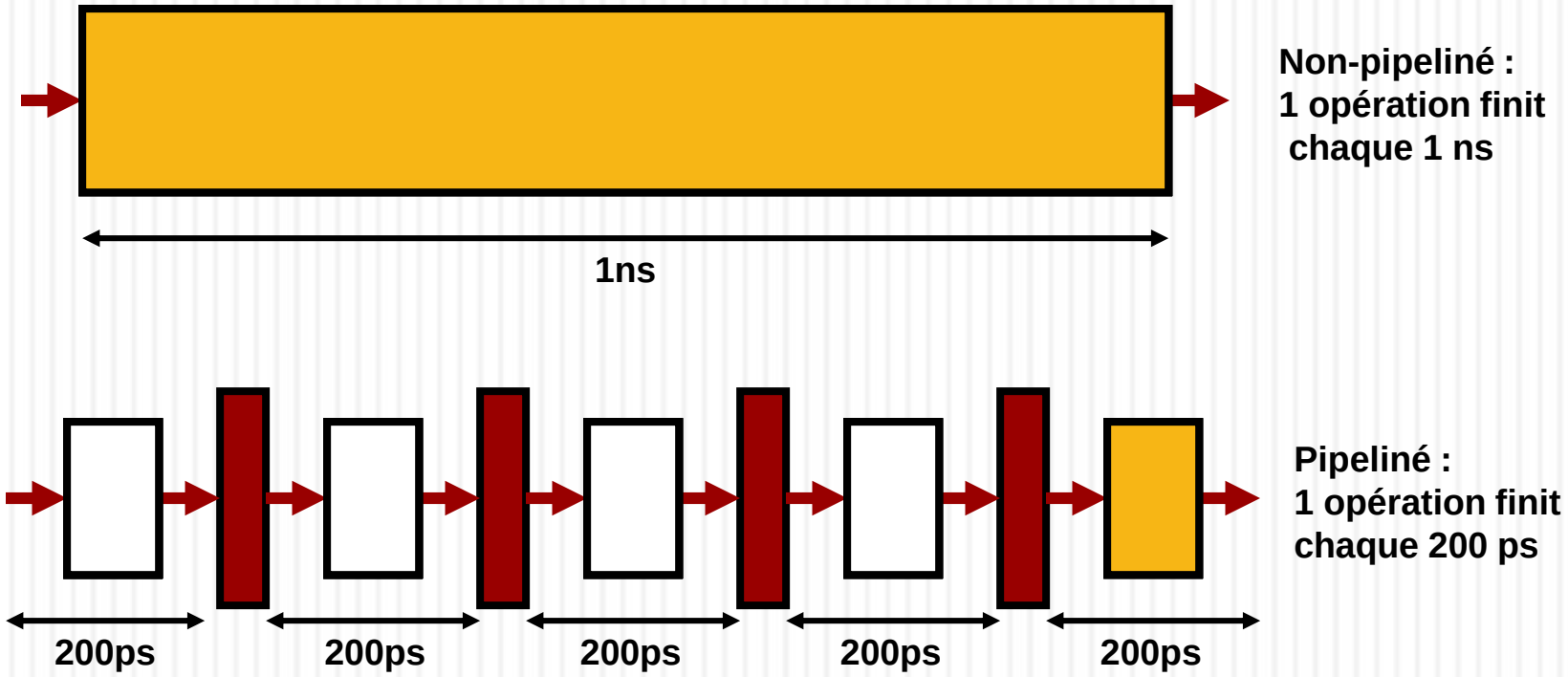
Temps sur M1 :

$$Cycle \times CPI\ moyen = 10ns \times ((0.4+0.2) \times 4 + 0.4 \times 5) = 44ns$$

$$Acceleration = \frac{Temps\ sur\ M1}{Temps\ moyen\ sur\ M2} = \frac{44ns}{11ns} = 4 fois$$

Différences

- Pourquoi faisons ceci ? Il est plus rapide pour les traitements répétitifs



Remarques sur le pipelining

- Pipelining augmente le débit (throughput), mais pas la latence
  - Résultat disponible chaque 200 ps, alors que
  - un traitement mono reste prenne 1ns
- Inconvénients :
  - Les traitements doivent être divisibles en taille étage
  - Les registres pipeline ajoutent un surcoût

Aléas du pipeline

- **Pourquoi ne peut-on pas obtenir un CPI = 1 ?**
- Plus le pipeline est long, plus le nombre de cas où il n'est pas possible d'atteindre la performance maximale est élevé
- Il existe 3 principaux cas où la performance d'un processeur pipeliné peut être dégradé. Ces cas de dégradation de performance sont appelés aléas. Ils empêchent l'instruction suivante de s'exécuter au cycle prochain comme prévu :
  - **Les aléas structurels** : Ils interviennent lors des conflits de ressource.
  - **Les aléas de données** : Dépendance de données
    Ils interviennent lorsqu'une instruction dépend du résultat d'une instruction précédente.
  - **Les aléas de contrôle** : Ils interviennent lors de l'exécution des instructions de branchement et des instructions modifiant le PC

Questions

Q1. Pourquoi le pipelining améliore-t-il la performance ?

Q2. Quelles sont les limites de l'amélioration de performance offerte par le pipelining ?

Q3. Donner deux techniques proposant des solutions aux aléas : l'une logicielle et l'autre matérielle


THANK YOU