Systèmes à microprocesseurs

Chapitre 1 : Introduction aux Systèmes à microprocesseurs

J.ZAIDOUNI
Université Mohammed Premier
ENSA Oujda, 2017/2018

Sommaire

Sommaire

Sommaire

Introduction

- L'INFORMATIQUE, contraction d'INFORmation et autoMATIQUE, est la science du traitement de l'information.
- Apparue au milieu du 20ème siècle, elle a connu une évolution extrêmement rapide.
- Motivation initiale : faciliter et d'accélérer le calcul.
- Se sont ajoutées de nombreuses fonctionnalités comme : l'automatisation, le contrôle et la commande de processus, la communication ou le partage de l'information.

Introduction

- Le cours d'architecture des systèmes à microprocesseurs expose les principes de base du **traitement programmé de l'information**.
- La mise en œuvre de ces systèmes s'appuie sur :
  - Le matériel (hardware) correspond à l'aspect concret du système : unité centrale, mémoire, organes d'entrées-sorties, etc...
  - Le logiciel (software) correspond à un ensemble d'instructions , appelé programme, qui sont contenues dans les différentes mémoires du système et qui définissent les actions effectuées par le matériel.

Définition de l'architecture

- L'architecture d'un système à microprocesseur représente l'organisation de ses différentes unités et de leurs interconnexions.
- Le choix d'une architecture est toujours le résultat d'un compromis :
  - entre performances et coûts
  - entre efficacité et facilité de construction
  - entre performances et facilité de programmation
  - etc ...

Définition du microprocesseur

- Un microprocesseur est un circuit intégré complexe.
- Il résulte de l'intégration sur une puce de fonctions logiques combinatoires (logiques et/ou arithmétique) et séquentielles (registres, compteur, etc...). Il est capable d'interpréter et d'exécuter les instructions d'un programme.
- Le premier microprocesseur de Intel (1971), le 4004, qui était une unité de calcul 4 bits fonctionnant à 108 kHz. (intégration d'environ 2300 transistors).

Rappels sur le codage de l'information

- Les informations traitées par un microprocesseur sont de différents types (nombres, instructions, images, vidéo, caractères , etc. . .) mais elles sont toujours représentées sous un format binaire.
- Seul le codage changera suivant les différents types de données à traiter. Elles sont représentées physiquement par 2 niveaux de tensions différents.
- En binaire, une information élémentaire est appelé bit et ne peut prendre que deux valeurs différentes : 0 ou 1.
- Une information plus complexe sera codée sur plusieurs bit.
- On appelle cet ensemble un mot. Un mot de 8 bits est appelé un octet.

Rappels sur le codage de l'information

- Représentation d'un nombre entier en binaire : Les nombres sont exprimés par des chiffres pouvant prendre deux valeurs 0 ou 1.
  - Exemple : le nombre binaire $N = (101)_2$ est équivalent au nombre décimal :$N = 1 * 2^2 + 0 * 2^1 + 1 * 2^0 = (5)_{10}$
- Représentation d'un nombre entier en hexadécimal : Lorsqu'une donnée est représentée sur plus de 4 bits, on préfère souvent l'exprimer en hexadécimal.
- Les nombres sont exprimés par des chiffres et des lettres pouvant prendre 16 valeurs : 0 1 2 3 4 5 6 7 8 9 A B C D E F
- A chaque chiffre est affecté un poids exprimé en puissance de 16.

Rappels sur le codage de l'information

- le nombre hexadécimal $N = (9A)_{16}$ est équivalent au nombre décimal :$N = 9 * 16^1 + 10 * 16^0 = (154)_{10}$
- Attention :
  - 1 Kilobit $= 2^{10}$ bit $= 1024$ bit
  - 1 Mégabit $= 2^{10}$ kbit $= 1024$ Kbit
  - 1 Gigabit $= 2^{10}$ Mbit $= 1024$ Mbit

Rappels sur le codage de l'information

- Le code ASCII représente les caractères sur 7 bits.
- Les codes (du 20 au 7F), sont utilisés pour coder les caractères alphabétiques (majuscules ou minuscules) et les caractères de ponctuation. $A = (41)_{16} = (65)_{10}$, $a = (61)_{16} = (97)_{10}$, les caractères sont consécutifs dans le code.
- Pour passer des majuscules aux minuscules on ajoute 32.
- On a étendu le code ASCII à 8 bits ce qui permet de représenter certains caractères accentués de certains pays européens ou l'alphabet grec (é, à, ...).

Applications des systèmes à microprocesseurs

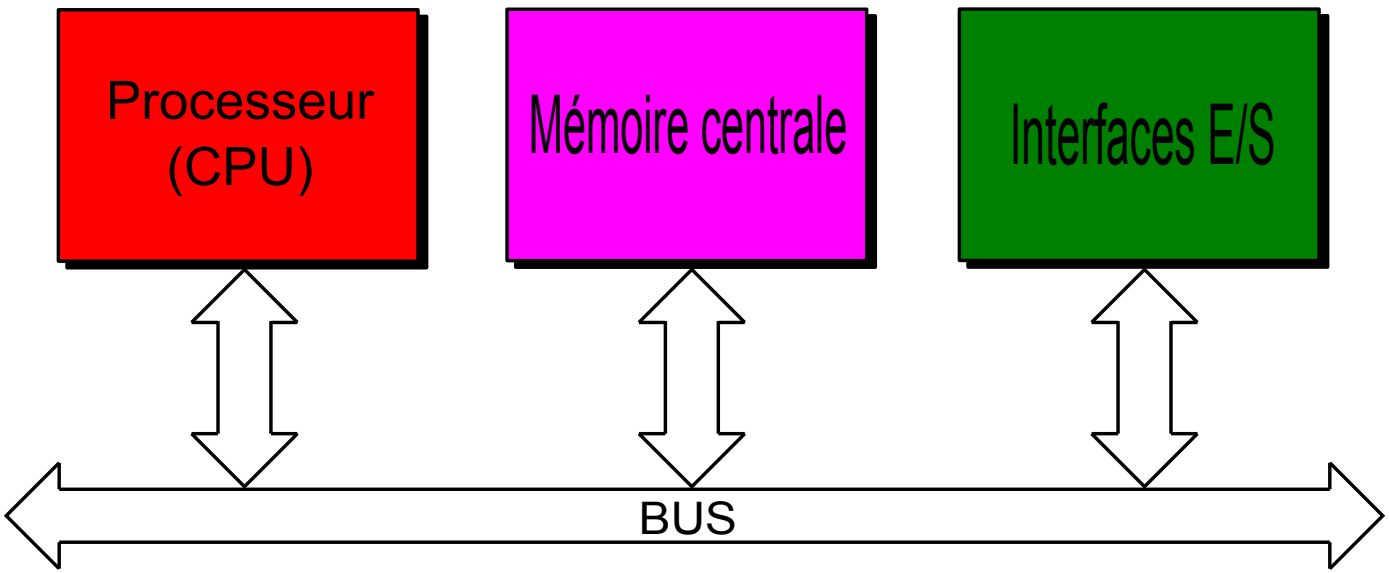
- Les applications des systèmes à microprocesseurs sont multiples et variées :
  - Ordinateur,
  - PDA
  - console de jeux
  - calculatrice
  - télévision
  - téléphone portable
  - distributeur automatique d'argent
  - robotique
  - lecteur carte à puce, code barre
  - automobile
  - instrumentation
  - etc. . .

Architecture de base : Modèle de Von Neumann

- Pour traiter une information, un microprocesseur seul ne suffit pas, il faut l'insérer au sein d'un système minimum de traitement programmé de l'information.
- John Von Neumann est à l'origine d'un modèle de machine universelle de traitement programmé de l'information (1946). Cette architecture sert de base à la plupart des systèmes à microprocesseur actuel. Elle est composé des éléments suivants :
  - une unité centrale (CPU, Microprocesseur)
  - une mémoire principale
  - des interfaces d'entrées/sorties

Architecture de base : Modèle de von Neumann

- Les différents organes du système sont reliés par une voie de communication appelée BUS.



Microprocesseur

- Le microprocesseur est chargé d'interpréter et d'exécuter les instructions d'un programme, de lire ou de sauvegarder les résultats dans la mémoire et de communiquer avec les unités d'échange (interfaces E/S).
- Toutes les activités du microprocesseur sont cadencées par une horloge.
- On caractérise le microprocesseur par :
  - sa fréquence d'horloge : en MHz ou GHz
  - le nombre d'instructions par secondes qu'il est capable d'exécuter : en MIPS
  - la taille des données qu'il est capable de traiter : en bits

Mémoire principale

- Elle contient les instructions des programmes en cours d'exécution et les données associées à ces programme.
- Physiquement, elle se décompose souvent en :
  - une mémoire morte (ROM = Read Only Memory) chargée de stocker le programme. C'est une mémoire à lecture seule.
  - une mémoire vive (RAM = Random Access Memory) chargée de stocker les données intermédiaires ou les résultats de calculs. On peut lire ou écrire des données dedans, ces données sont perdues à la mise hors tension.
- Remarque : Les disques durs, disquettes, CDROM, etc. . . sont des périphériques de stockage et sont considérés comme des mémoires secondaires.

Interfaces d'entrées/sorties (E/S), et bus

- Les interfaces d'entrées/sorties :
  - Elles permettent d'assurer la communication entre le microprocesseur et les périphériques. (capteur, clavier, moniteur ou afficheur, imprimante, modem, etc...).
- Les bus :
  - Un bus est un ensemble de fils qui assure la transmission de l'information.
  - On retrouve trois types de bus véhiculant des informations en parallèle dans un système de traitement programmé de l'information.
- 1. un bus de données : bidirectionnel qui assure le transfert des informations entre le microprocesseur et son environnement, et inversement. Son nombre de lignes est égal à la capacité de traitement du microprocesseur. Ce nombre est *m bits*.

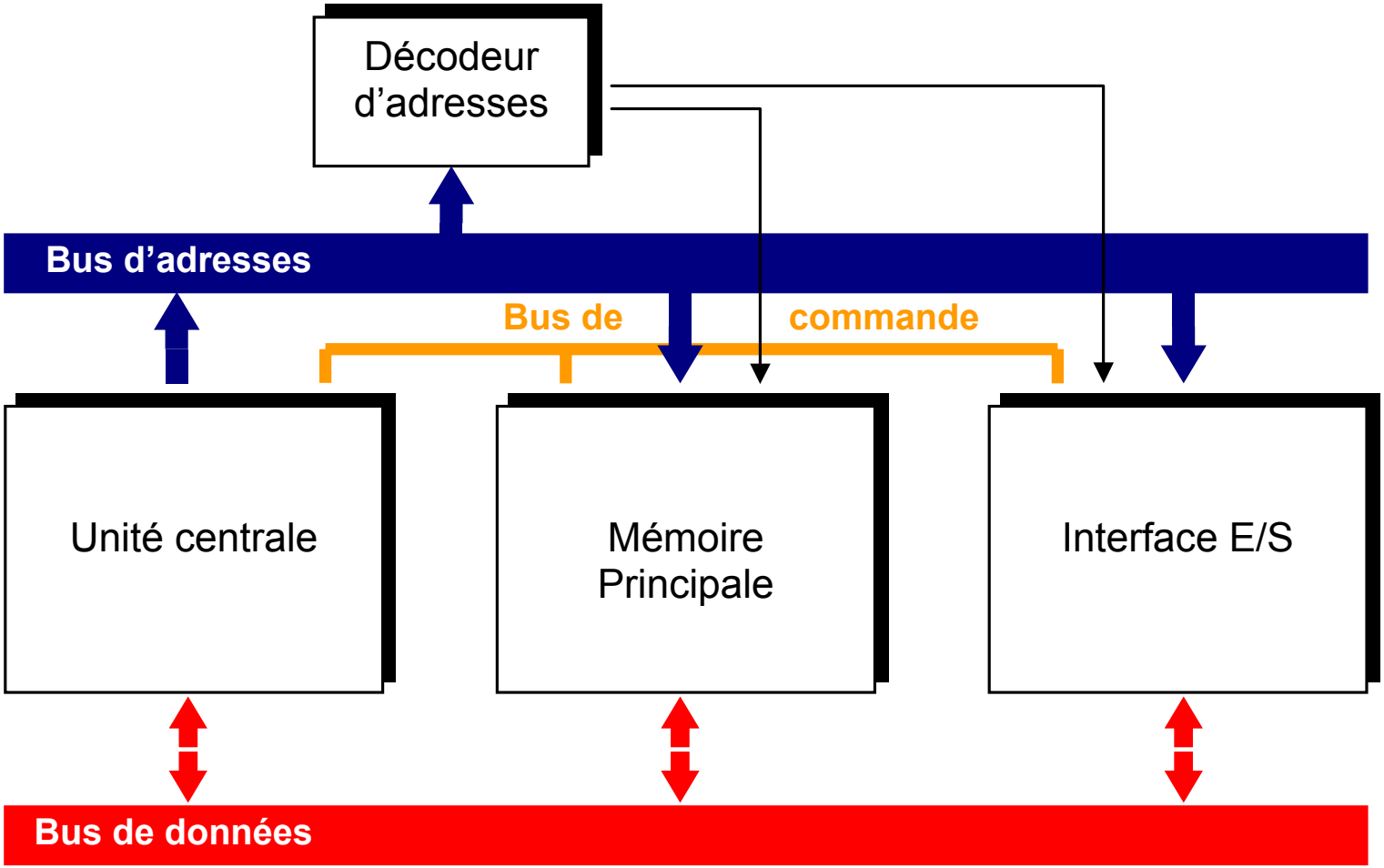
Interfaces d'entrées/sorties (E/S), et bus

- 2. un bus d'adresses : unidirectionnel qui permet la sélection des informations à traiter dans un espace mémoire (ou espace adressable) qui peut avoir 2^n emplacements, avec n = nombre de conducteurs du bus d'adresses.
- 3. un bus de commande : constitué par quelques conducteurs qui assurent la synchronisation des flux d'informations sur les bus des données et des adresses.

Décodage d'adresses

- La multiplication des périphériques autour du microprocesseur oblige la présence d'un décodeur d'adresse chargé d'aiguiller les données présentes sur le bus de données.
- En effet, le microprocesseur peut communiquer avec les différentes mémoires et les différents boîtier d'interface. Ceux-ci sont tous reliés sur le même bus de données et afin d'éviter des conflits, un seul composant doit être sélectionné à la fois.
- Lorsqu'on réalise un système microprogrammé, on attribue donc à chaque périphérique une zone d'adresse et une fonction" décodage d'adresses" est donc nécessaire afin de fournir les signaux de sélection de chacun des composants.

Décodage d'adresses



Décodage d'adresses

Remarque : lorsqu'un composant n'est pas sélectionné, ses sorties sont mises à l'état " haute impédance " afin de ne pas perturber les données circulant sur le bus. (elle présente une impédance de sortie très élevée = circuit ouvert).

Mémoires

- Une mémoire est un circuit à semi-conducteur permettant d'enregistrer, de conserver et de restituer des informations (instructions et variables).
- Les informations peuvent être écrites ou lues :
  - Il y a écriture lorsqu'on enregistre des informations en mémoire
  - Il y a écriture lecture lorsqu'on récupère des informations précédemment enregistrées.

Mémoires

- Avec une adresse de *n bits* il est possible de référencer au plus 2^n cases mémoire. Soit $Adresse \in [0, 2^n - 1]$.
- Chaque case est remplie par un mot de données (sa longueur m est toujours une puissance de 2).
- Un boîtier mémoire comprend en plus :
  - une entrée de commande ($R/\overline{W}$) qui permet de définir le type d'action que l'on effectue avec la mémoire (lecture/écriture
  - une entrée de sélection ($\overline{CS}$) qui sélectionner cette mémoire.

Mémoires

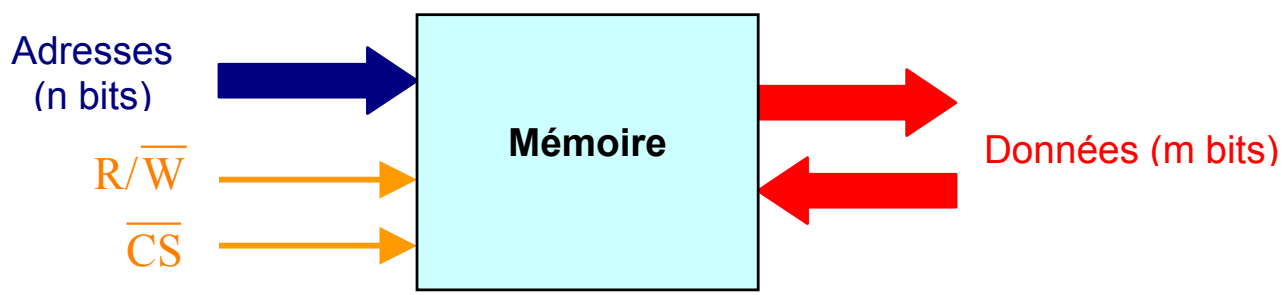
- Exemple : Mémoire $8 \times 8\ bits = 2^3 \times 8\ bits = 64\ bits$ ($n = 3$ et $m = 8$), un mot est codé sur 8 bits = 1 octet.

Adresse	Case mémoire
7 = 111	
6 = 110	
5 = 101	
4 = 100	
3 = 011	
2 = 010	
1 = 001	
0 = 000	0001 1010

Mémoires

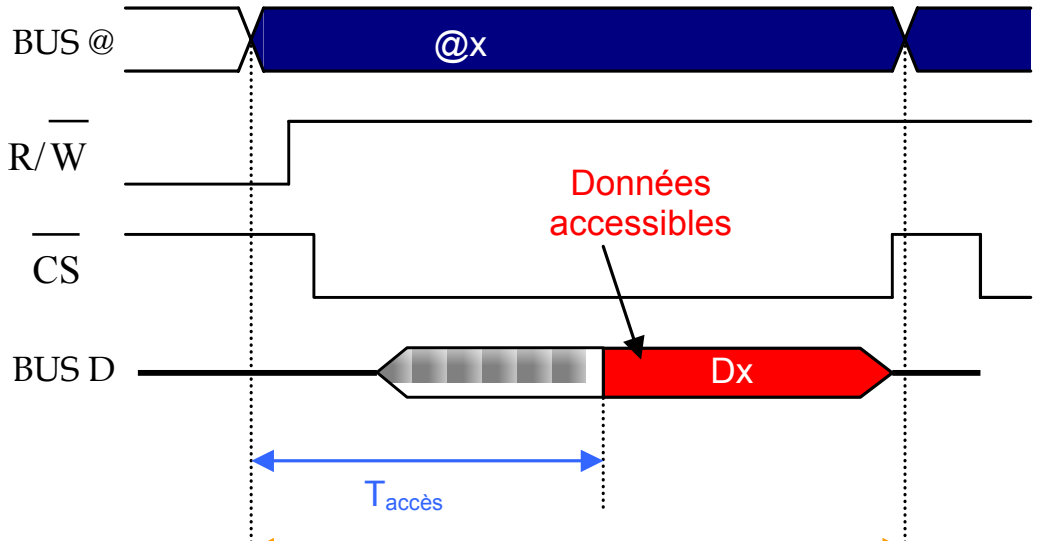
On peut donc schématiser un circuit mémoire par la figure suivante où l'on peut distinguer :

- les entrées d'adresses
- les entrées et sorties de données (souvent regroupées sur des bornes bidirectionnelles)
- les entrées de commandes :
  - une entrée de sélection de lecture ou d'écriture. ($R/\overline{W}$)
  - une entrée de sélection du circuit. ($\overline{CS}$)



Mémoires

- Une opération de lecture ou d'écriture de la mémoire suit toujours le même cycle :
  1. sélection de l'adresse
  2. choix de l'opération à effectuer ($R/\overline{W} = 1$ pour une lecture, $R/\overline{W} = 0$ pour une écriture)
  3. sélection de la mémoire ($\overline{CS} = 0$)
  4. lecture ou écriture de la donnée
- Exemple : Chronogramme d'un cycle de lecture



Caractéristiques d'une mémoire

- La capacité : c'est le nombre total de bits que contient la mémoire. Elle s'exprime aussi souvent en octet.
- Le format des données : c'est le nombre de bits que l'on peut mémoriser par case mémoire. On dit aussi que c'est la largeur du mot mémorisable.
- Le temps d'accès : c'est le temps qui s'écoule entre l'instant où a été lancée une opération de lecture en mémoire et l'instant où l'information est disponible sur le bus de données.

Caractéristiques d'une mémoire

- Le temps de cycle : il représente l'intervalle minimum qui doit séparer deux demandes successives de lecture ou d'écriture.
- Le débit : c'est le nombre maximum d'informations lues ou écrites par seconde.
- Volatilité : elle caractérise la permanence des informations dans la mémoire.
  - Mémoire est volatile si son contenu est effacé lors de la mise hors tension (les RAM)
  - Mémoire est non volatile si son contenu est maintenu lors de la mise hors tension (les ROM)

Notion de hiérarchie mémoire

- Une mémoire idéale serait une mémoire de grande capacité, capable de stocker un maximum d'informations et possédant un temps d'accès très faible afin de pouvoir travailler rapidement sur ces informations.
- Mais il se trouve que les mémoires de grande capacité sont souvent très lente et que les mémoire rapides sont très chères.
- Et pourtant, la vitesse d'accès à la mémoire conditionne dans une large mesure les performances d'un système.
- En effet, c'est là que se trouve le goulot d'étranglement entre un microprocesseur capable de traiter des informations très rapidement et une mémoire beaucoup plus lente (ex : processeur actuel à 3Ghz et mémoire à 400MHz).

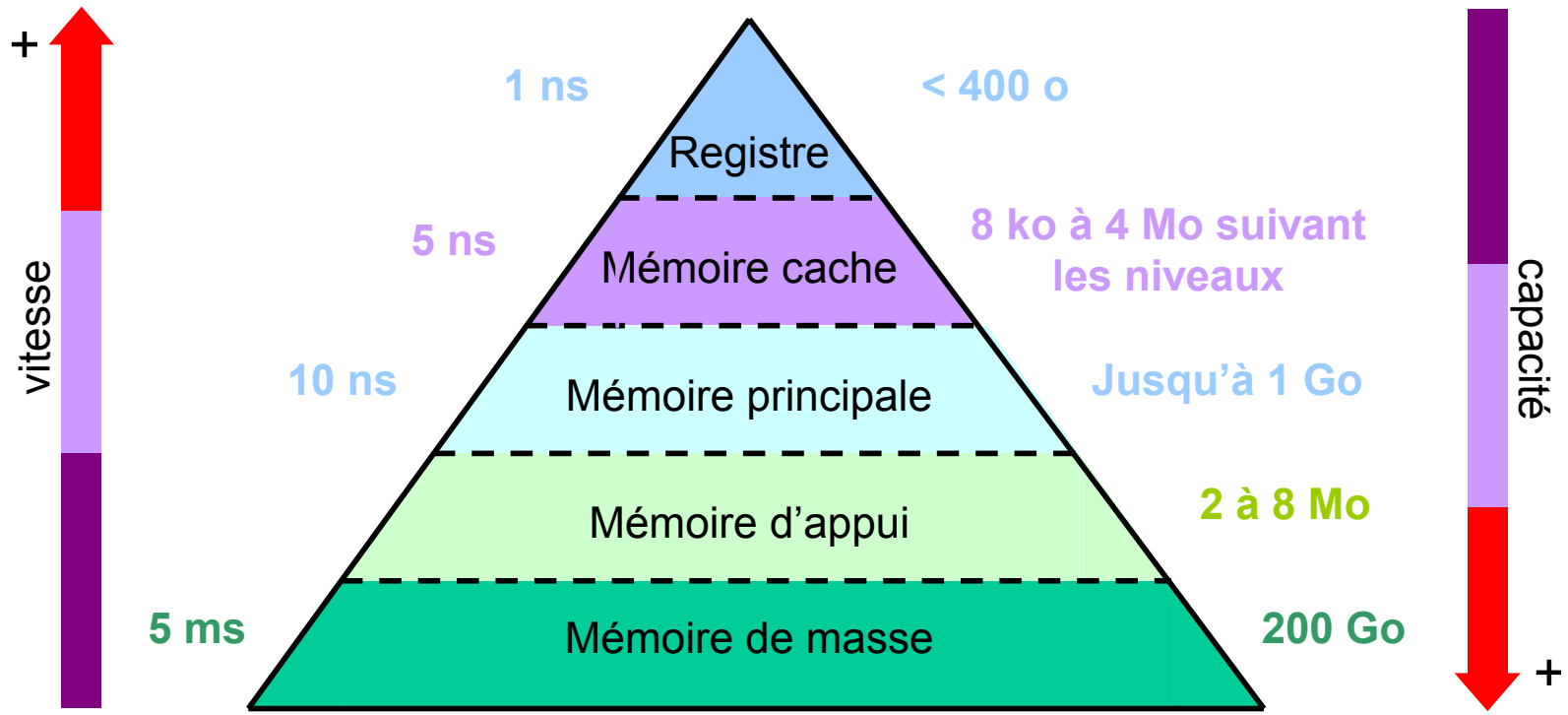
Notion de hiérarchie mémoire

- Or, on n'a jamais besoin de toutes les informations au même moment. Afin d'obtenir le meilleur compromis coût-performance, on définie donc une hiérarchie mémoire.
- On utilise des mémoires de faible capacité mais très rapides pour stocker les informations dont le microprocesseur se sert le plus et on utilise des mémoires de capacité importante mais beaucoup plus lente pour stocker les informations dont le microprocesseur se sert le moins.
- Ainsi, plus on s'éloigne du microprocesseur et plus la capacité et le temps d'accès des mémoire vont augmenter.

Notion de hiérarchie mémoire

- Les types de mémoires :
  - Les registres sont les éléments de mémoire les plus rapides. Ils sont situés au niveau du processeur et servent au stockage des opérandes et des résultats intermédiaires.
  - La mémoire cache est une mémoire rapide de faible capacité destinée à accélérer l'accès à la mémoire centrale en stockant les données les plus utilisées.
  - La mémoire principale est l'organe principal de rangement des informations. Elle contient les programmes (instructions et données) et est plus lente que les deux mémoires précédentes.
  - La mémoire d'appui sert de mémoire intermédiaire entre la mémoire centrale et les mémoires de masse. Elle joue le même rôle que la mémoire cache.
  - La mémoire de masse est une mémoire périphérique de grande capacité utilisée pour le stockage permanent ou la sauvegarde des informations. Elle utilise pour cela des supports magnétiques (disque dur) ou optiques (CDROM, DVDROM).

Notion de hiérarchie mémoire



Microprocesseur

- Un microprocesseur est un circuit intégré complexe caractérisé par une très grande intégration et doté des facultés d'interprétation et d'exécution des instructions d'un programme.
- Il est chargé d'organiser les tâches précisées par le programme et d'assurer leur exécution.
- Il doit aussi prendre en compte les informations extérieures au système et assurer leur traitement.
- A l'heure actuelle, un microprocesseur regroupe sur quelques millimètre carré des fonctionnalités toujours plus complexes.

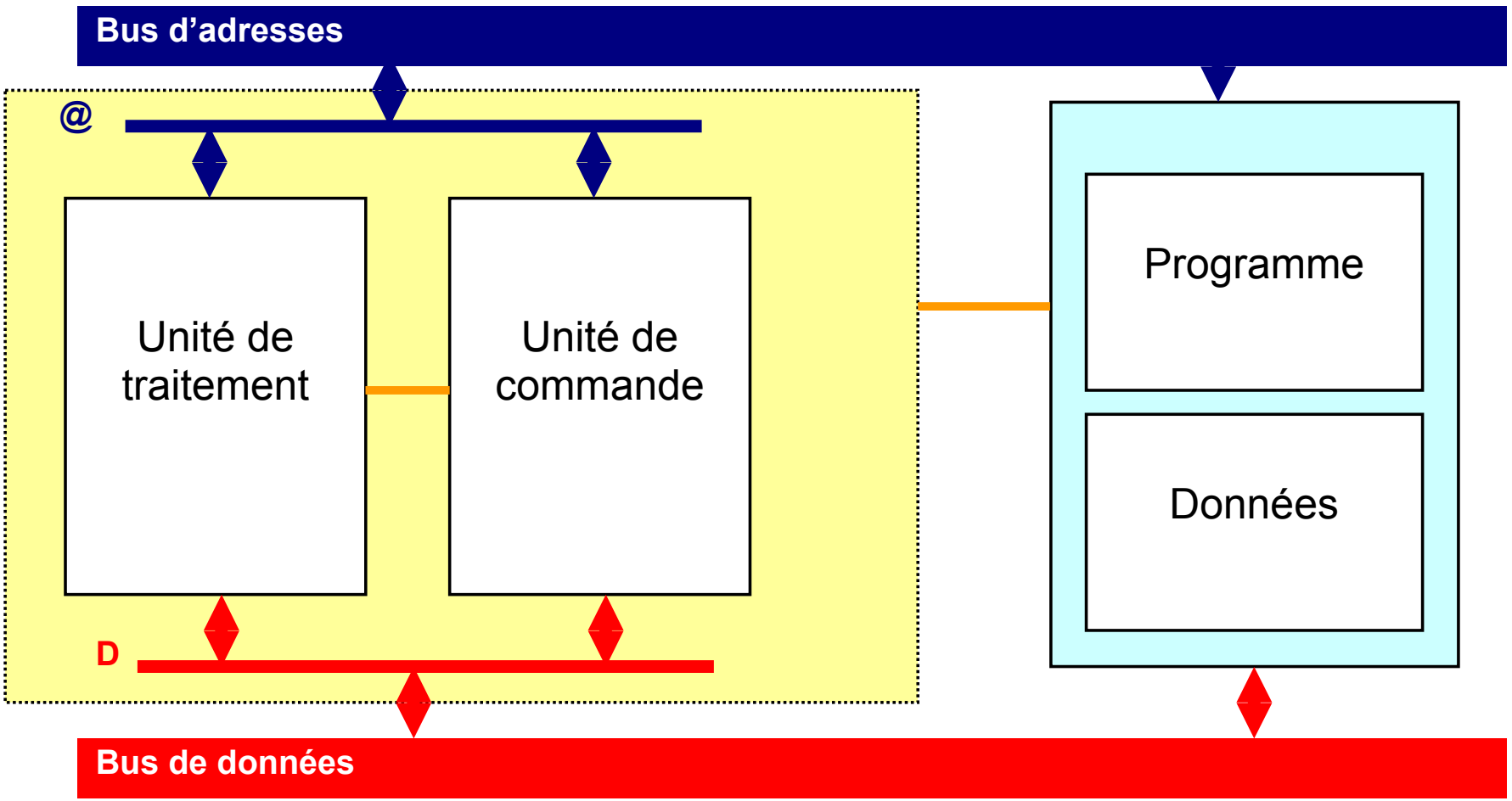
Architecture de base d'un microprocesseur

- Un microprocesseur est construit autour de deux éléments principaux :
  1. Une unité de commande
  2. Une unité de traitement
- Ces deux éléments sont associés à des registres chargés de stocker les différentes informations à traiter.
- Les trois éléments sont reliés entre eux par des bus internes permettant les échanges d'informations.

Architecture de base d'un microprocesseur

- Il existe deux types de registres :
  - les registres d'usage général permettent à l'unité de traitement de manipuler des données à vitesse élevée. Ils sont connectés au bus données interne au microprocesseur.
  - les registres d'adresses (pointeurs) connectés sur le bus adresses.

Architecture de base d'un microprocesseur



Unité de commande

- Elle permet de séquencer le déroulement des instructions.
- Elle effectue la recherche en mémoire de l'instruction.
- Comme chaque instruction est codée sous forme binaire, elle en assure le décodage pour enfin réaliser son exécution puis effectue la préparation de l'instruction suivante.
- Pour cela, elle est composée par :
- le compteur de programme constitué par un registre dont le contenu est initialisé avec l'adresse de la première instruction du programme. Il contient toujours l'adresse de l'instruction à exécuter.

Unité de commande

- le registre d'instruction et le décodeur d'instruction : chacune des instructions à exécuter est rangée dans le registre instruction puis est décodée par le décodeur d'instruction.
- Bloc logique de commande (ou séquenceur) : Il organise l'exécution des instructions au rythme d'une horloge. Il élabore tous les signaux de synchronisation internes ou externes (bus de commande) du microprocesseur en fonction des divers signaux de commande provenant du décodeur d'instruction ou du registre d'état par exemple. Il s'agit d'un automate (machine à états) réalisé soit de façon câblée, soit de façon micro-programmée.

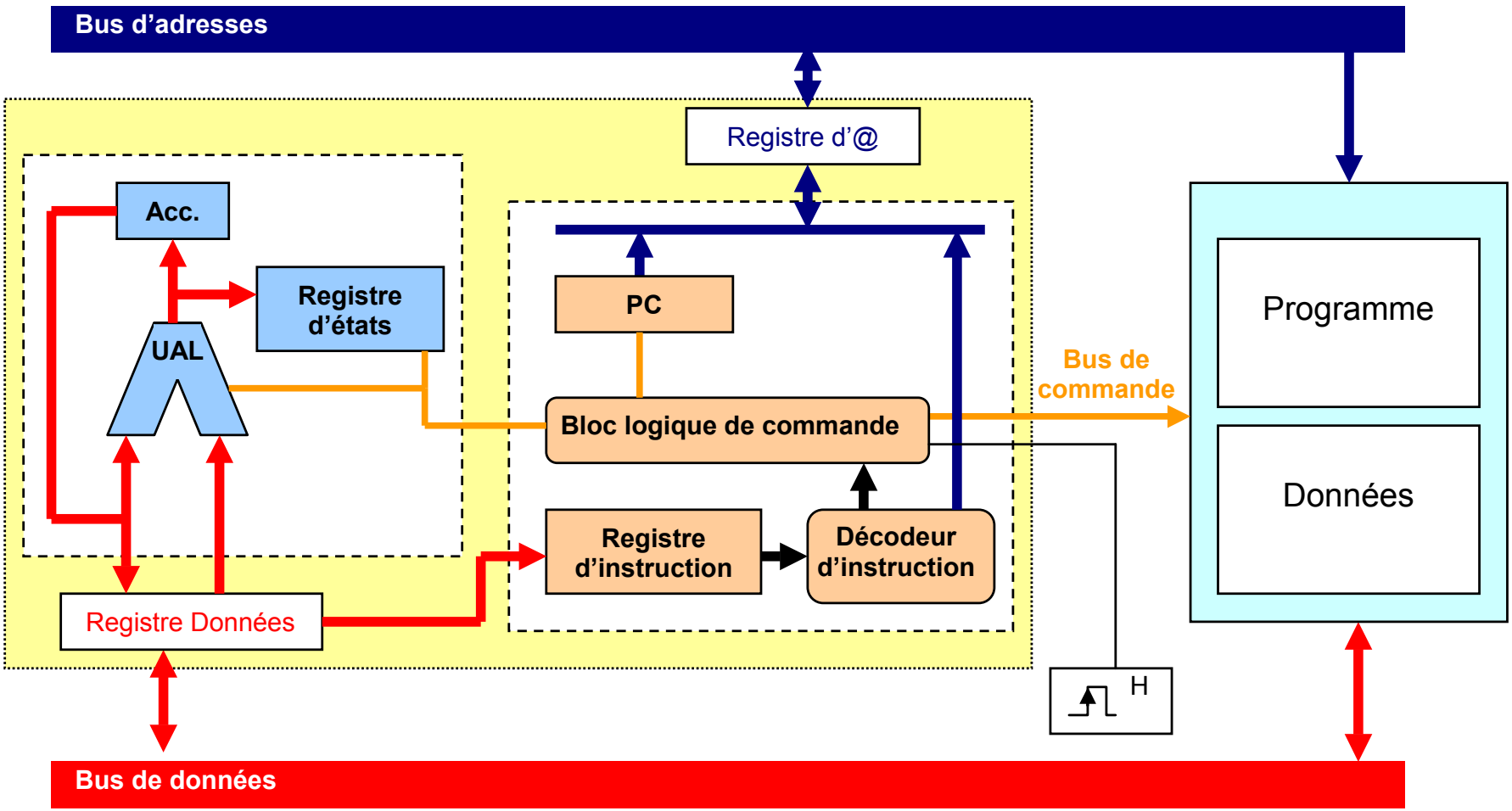
Unité de traitement

- Elle regroupe les circuits qui assurent les traitements nécessaires à l'exécution des instructions :
  - L'Unité Arithmétique et Logique (UAL) est un circuit complexe qui assure les fonctions logiques (ET, OU, Comparaison, Décalage , etc. . .) ou arithmétique (Addition, soustraction).
  - Le registre d'état est généralement composé de 8 bits à considérer individuellement. Chacun de ces bits est un indicateur dont l'état dépend du résultat de la dernière opération effectuée par l'UAL. On les appelle indicateur d'état ou flag ou drapeaux. Dans un programme le résultat du test de leur état conditionne souvent le déroulement de la suite du programme.
  - Les accumulateurs sont des registres de travail qui servent à stocker une opérande au début d'une opération arithmétique et le résultat à la fin de l'opération.

Unité de traitement

- On peut citer par exemple les indicateurs du registre d'état :
  - retenue (carry : C)
  - retenue intermédiaire (Auxiliary-Carry : AC)
  - signe (Sign : S)
  - débordement (overflow : OV ou V)
  - zéro (Z)
  - parité (Parity : P)

Architecture de base détaillée

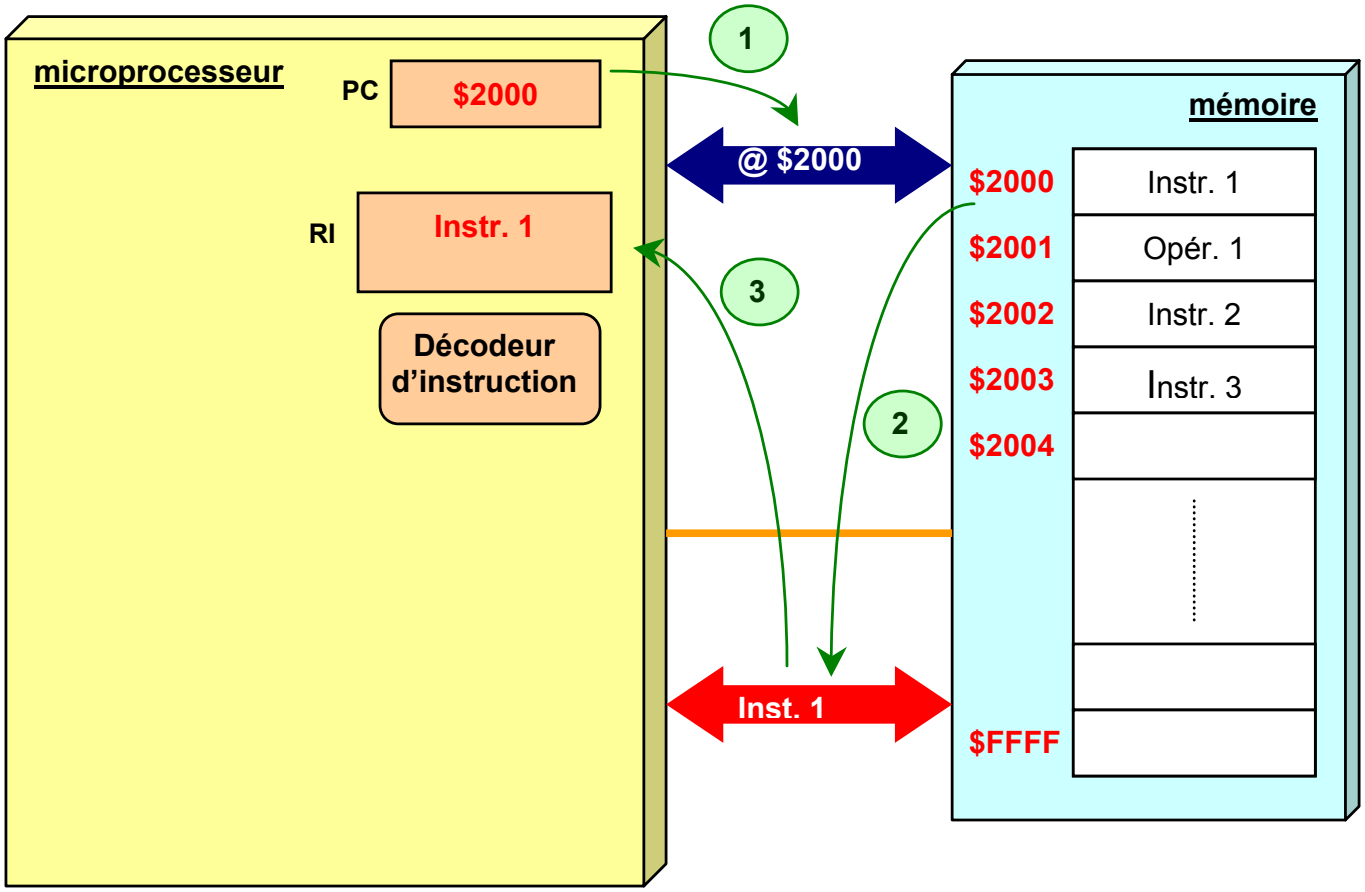


Cycle d'exécution d'une instruction

- Le microprocesseur ne comprend qu'un certain nombre d'instructions qui sont codées en binaire. Le traitement d'une instruction peut être décomposé en trois phases.
- Phase 1 : Recherche de l'instruction à traiter (fetch)
  1. Le PC contient l'adresse de l'instruction suivante du programme. Cette valeur est placée sur le bus d'adresses par l'unité de commande qui émet un ordre de lecture.
  2. Au bout d'un certain temps (temps d'accès à la mémoire), le contenu de la case mémoire sélectionnée est disponible sur le bus de données.
  3. L'instruction est stockée dans le registre instruction du processeur.

Cycle d'exécution d'une instruction

- Phase 1 : Recherche de l'instruction à traiter (fetch)



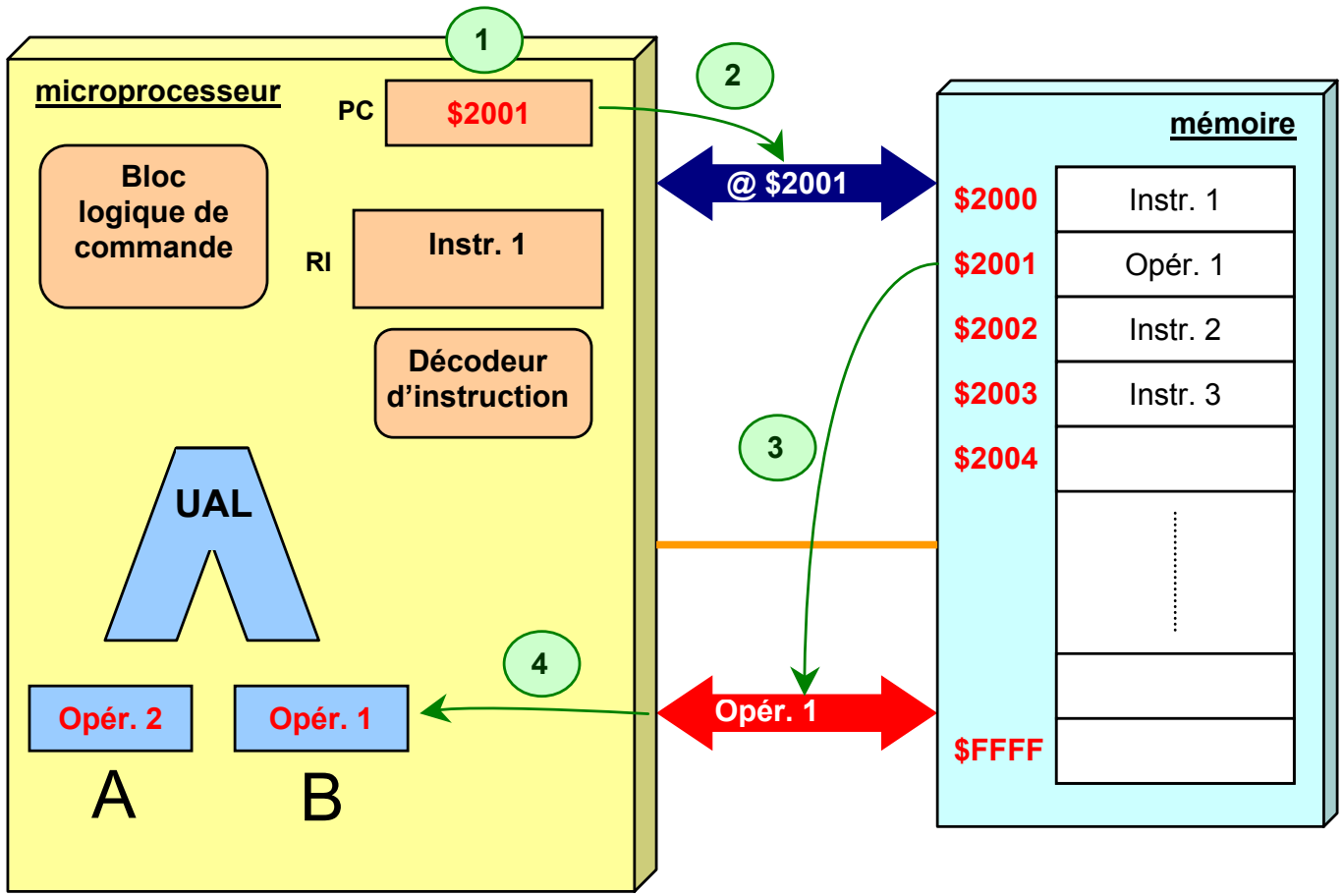
Cycle d'exécution d'une instruction

- Phase 2 : Décodage de l'instruction et recherche de l'opérande (decode) : L'unité de commande transforme l'instruction en une suite de commandes élémentaires nécessaires au traitement de l'instruction.

1. Incrémenter le PC. Si l'instruction nécessite une donnée (opérande 1) en provenance de la mémoire, Le PC contient l'adresse de de cette donnée.
2. Cette valeur est placée sur le bus d'adresses par l'unité de commande qui émet un ordre de lecture.
3. Au bout d'un certain temps (temps d'accès à la mémoire), le contenu de la case mémoire sélectionnée est disponible sur le bus de données.
4. L'opérande est stockée dans le registre B.

Cycle d'exécution d'une instruction

- Phase 2 : Décodage de l'instruction et recherche de l'opérande (decode)

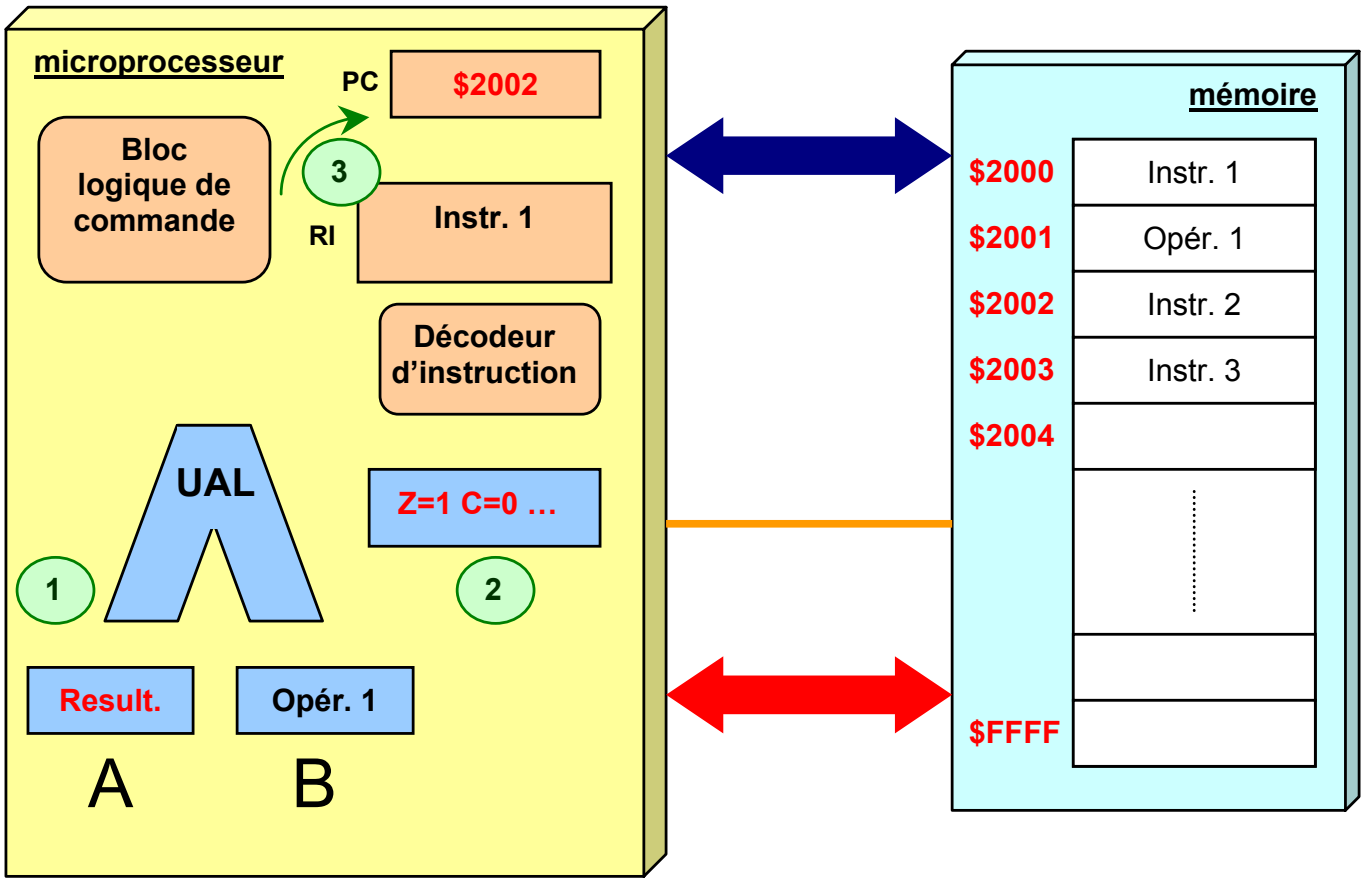


Cycle d'exécution d'une instruction

- Phase 3 : Exécution de l'instruction (execute)
  1. Le micro-programme réalisant l'instruction est exécuté.
  2. Les drapeaux sont positionnés (registre d'état).
  3. L'unité de commande incrémente le PC pour l'instruction suivante.

Cycle d'exécution d'une instruction

- Phase 3 : Exécution de l'instruction (execute)



Jeu d'instructions

- La première étape de la conception d'un microprocesseur est la définition de son jeu d'instructions.
- Le jeu d'instructions décrit l'ensemble des instructions que le microprocesseur pourra exécuter.
- Déterminer l'architecture du microprocesseur à réaliser et notamment celle du séquenceur.
- A un même jeu d'instructions peut correspondre un grand nombre d'implémentations différentes du microprocesseur

Jeu d'instructions

- Les instructions que l'on retrouve dans chaque microprocesseur peuvent être classées en 4 groupes :
  1. Transfert de données pour charger ou sauver en mémoire, effectuer des transferts de registre à registre, etc...
  2. Opérations arithmétiques : addition, soustraction, division, multiplication
  3. Opérations logiques : ET, OU, NON, NAND, comparaison, test, etc...
  4. Contrôle de séquence : branchement, test, etc...

Jeu d'instructions, Codage

- Les instructions et leurs opérandes (paramètres) sont stockés en mémoire principale.
- La taille totale d'une instruction (nombre de bits nécessaires pour la représenter en mémoire) dépend du type d'instruction et aussi du type d'opérande. Chaque instruction est toujours codée sur un nombre entier d'octets afin de faciliter son décodage par le processeur. Une instruction est composée de deux champs :
  - le code instruction, qui indique au processeur quelle instruction réaliser
  - le champ opérande qui contient la donnée, ou la référence à une donnée en mémoire (son adresse).

Jeu d'instructions, Codage

- Exemple :

Code instruction	Code opérande
1001 0011	0011 1110

- Le nombre d'instructions du jeu d'instructions est directement lié au format du code instruction. Ainsi un octet permet de distinguer au maximum 256 instructions différentes.

Jeu d'instructions, Mode d'adressage

- Un mode d'adressage définit la manière dont le microprocesseur va accéder à la donnée.
- Les différents modes d'adressage dépendent des microprocesseurs mais on retrouve en général :
  1. Adressage de registre : le registre contient la donnée à traiter par l'instruction.
  2. Adressage immédiat : la valeur de la donnée est définit immédiatement dans l'instruction.
  3. Adressage direct : on spécifit dans l'instruction l'adresse de la donnée à traiter.
  4. Adressage indirect : on spécifit dans l'instruction l'adresse de l'adresse de la donnée (adresse du pointeur) à traiter.
- Selon le mode d'adressage de la donnée, une instruction sera codée par 1 ou plusieurs octets.

Jeu d'instructions, Temps d'exécution

- Chaque instruction nécessite un certain nombre de cycles d'horloges pour s'effectuer.
- Le nombre de cycles dépend de la complexité de l'instruction et aussi du mode d'adressage.
- Il est plus long d'accéder à la mémoire principale qu'à un registre du processeur.
- La durée d'un cycle d'horloge = l'inverse de la fréquence d'horloge du processeur.

Langage de programmation

- Le langage machine est le langage compris par le microprocesseur.
- Ce langage est difficile à maîtriser puisque chaque instruction est codée par une séquence propre de bits.
- Afin de faciliter la tâche du programmeur, on a créé différents langages plus ou moins évolués.

Langage de programmation

- Le langage assembleur est le langage le plus "proche" du langage machine. Il est composé par des instructions que l'on appelle des mnémoniques.
- Ce sont essentiellement des opérations de transfert de données entre les registres et l'extérieur du microprocesseur (mémoire ou périphérique), ou des opérations arithmétiques ou logiques.
- Chaque instruction représente un code machine différent.
- Chaque microprocesseur posséde un assembleur différent.

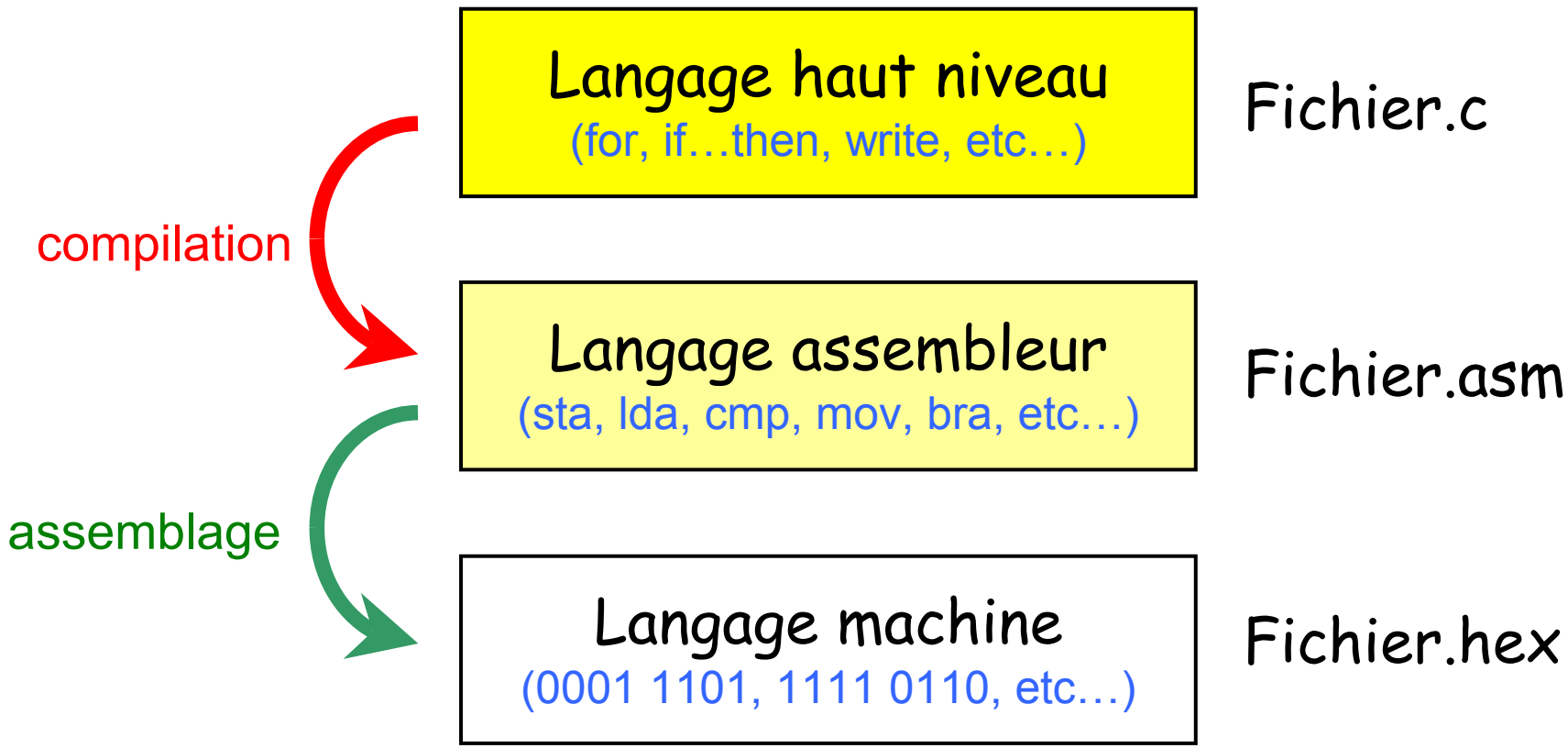
Langage de programmation

- La difficulté de mise en œuvre de ce type de langage, et leur forte dépendance avec la machine a nécessité la conception de langages de haut niveau, plus adaptés à l'homme, et aux applications qu'il cherche à développer.
- Faisant abstraction de toute architecture de machine, ces langages permettent l'expression d'algorithmes sous une forme plus facile à comprendre, à apprendre, et à maîtriser (C, Pascal, Java, etc. . .).

Langage de programmation

- Chaque instruction en langage de haut niveau correspondra à une succession d'instructions en langage assembleur. Une fois développé, le programme en langage de haut niveau n'est donc pas compréhensible par le microprocesseur.
- Il faut le compiler pour le traduire en assembleur puis l'assembler pour le convertir en code machine compréhensible par le microprocesseur. Ces opérations sont réalisées à partir de logiciels spécialisés appelés compilateur et assembleur.

Langage de programmation



Langage de programmation

Code Machine (68HC11)	Assembleur (68HC11)	Code Source (Langage C)
@00 C6 64 @01 B6 00 @03 1B @04 5A @05 26 03	LDAB #100 LDAA #0 ret ABA DECB BNE ret	A=0 ; for (i=1 ; i<101 ; i++) A=A+i ;

Code Machine (μC 8051)	Assembleur (μC 8051)	Code Source (Langage C)
@0000 E4 @0001 7864 @0003 28 @0004 D8FD	org 0 CLR A; MOV R0,#100; boucle: ADD A, R0; DJNZ R0,boucle;	A=0 ; for (i=1 ; i<101 ; i++) A=A+i ;

Performances d'un microprocesseur

- On peut caractériser la puissance d'un microprocesseur par le nombre d'instructions qu'il est capable de traiter par seconde.
- Pour cela, on définit :
  - le CPI (Cycle Par Instruction) qui représente le nombre moyen de cycles d'horloge nécessaire pour l'exécution d'une instruction pour un microprocesseur donné.
  - le MIPS (Millions d'Instructions Par Seconde) qui représente la puissance de traitement du microprocesseur

$$MIPS = \frac{F_h}{CPI}$$

- Pour augmenter les performances d'un microprocesseur, on peut donc soit augmenter la fréquence d'horloge (limitation matérielle), soit diminuer le CPI (choix d'un jeu d'instruction adapté).

Améliorations de l'architecture de base

- 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 entraîne une élévation de température.
  - On est alors amené à équiper les processeurs de systèmes de refroidissement ou à diminuer la tension d'alimentation.

Améliorations de l'architecture de base

- 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.

Architecture pipeline

- L'exécution d'une instruction est décomposée en une succession d'étapes et chaque étape correspond à l'utilisation d'une des fonctions du microprocesseur.
- Lorsqu'une instruction se trouve dans l'une des étapes, les composants associés aux autres étapes ne sont pas utilisés.
- Le fonctionnement d'un microprocesseur simple n'est donc pas efficace.
- L'architecture pipeline permet d'améliorer l'efficacité du microprocesseur.

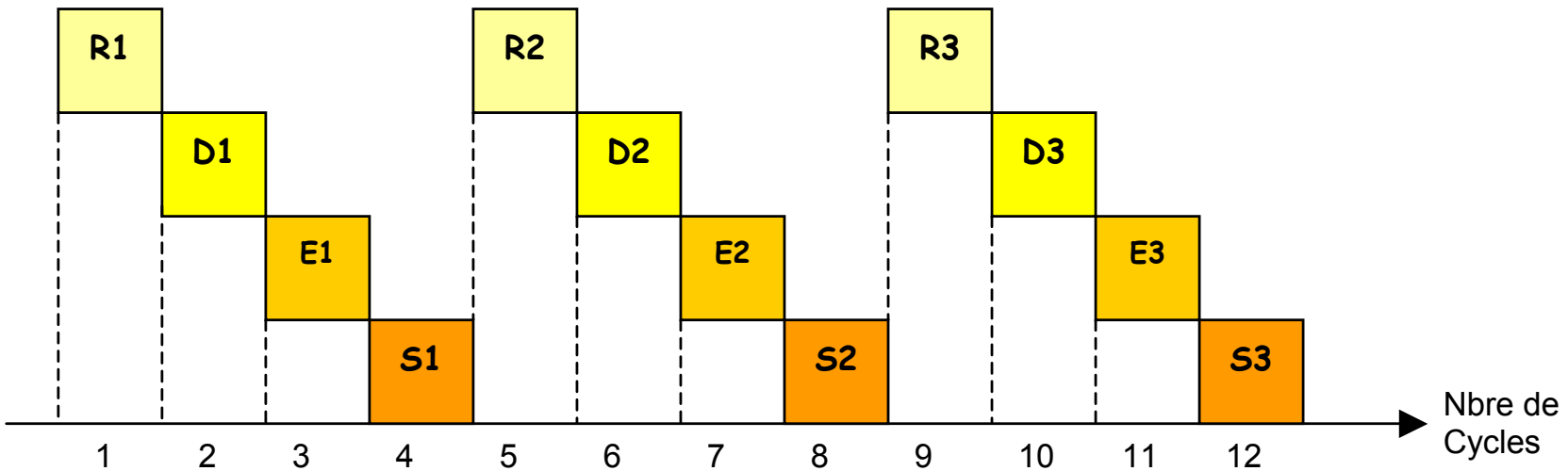
Architecture pipeline

- En effet, lorsque la première étape de l'exécution d'une instruction est achevée, l'instruction entre dans la seconde étape de son exécution et la première phase de l'exécution de l'instruction suivante débute.
- Il peut donc y avoir une instruction en cours d'exécution dans chacune des étapes et chacun des composants du microprocesseur peut être utilisé à chaque cycle d'horloge.
- L'efficacité est maximale.
- Le temps d'exécution d'une instruction n'est pas réduit mais le débit d'exécution des instructions est considérablement augmenté.
- Une machine pipeline se caractérise par le nombre d'étapes utilisées pour l'exécution d'une instruction, on appelle aussi ce nombre d'étapes le nombre d'étages du pipeline.

Architecture pipeline

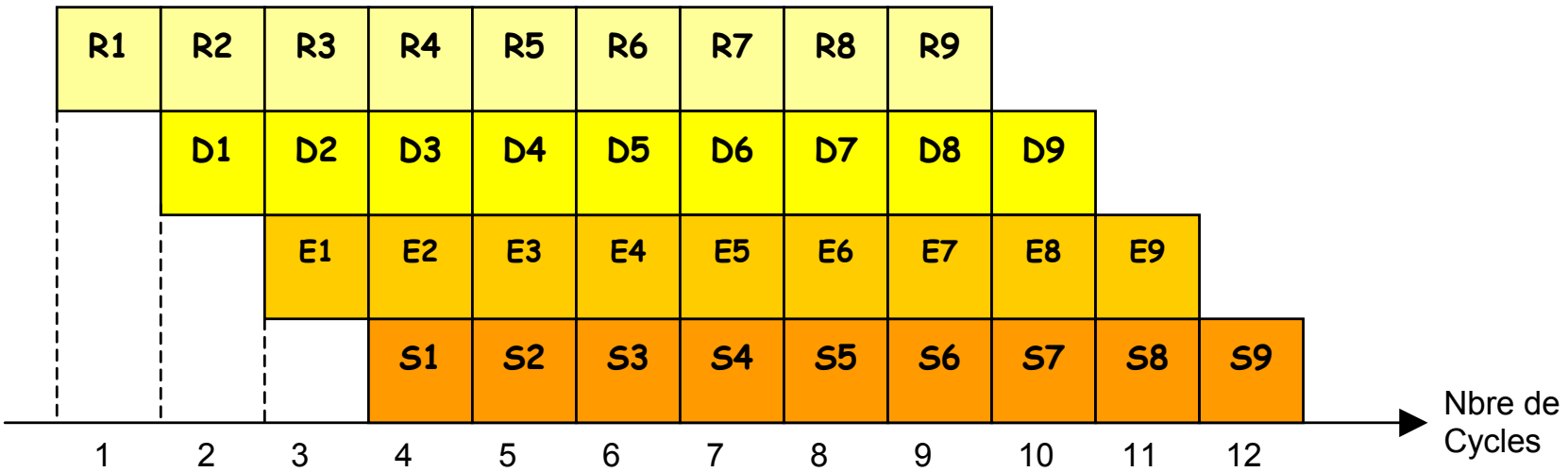
Exemple de l'exécution en 4 phases d'une instruction :

Modèle classique :



Architecture pipeline

Modèle pipeliné :



Architecture pipeline, Gain de performance

- Dans cette structure, la machine débute l'exécution d'une instruction à chaque cycle et le pipeline est pleinement occupé à partir du quatrième cycle. Le gain obtenu dépend donc du nombre d'étages du pipeline.
- En effet, pour exécuter n instructions, en supposant que chaque instruction s'exécute en k cycles d'horloge, il faut :
  - $n.k$ cycles d'horloge pour une exécution séquentielle.
  - k cycles d'horloge pour exécuter la première instruction puis $n-1$ cycles pour les $n-1$ instructions suivantes (càd 1 instruction/cycle) si on utilise un pipeline de k étages.

Architecture pipeline, Gain de performance

- Le gain obtenu est donc de :

$$G = \frac{n.k}{k + n - 1}$$

- Donc lorsque le nombre n d'instructions à exécuter est grand par rapport à k, on peut admettre qu'on divise le temps d'exécution par k :

$$G = k$$

Architecture pipeline

- Remarque : Le temps de traitement dans chaque unité doit être à peu près égal sinon les unités rapides doivent attendre les unités lentes.
- Exemples :
  - L'Athlon d'AMD comprend un pipeline de 11 étages.
  - Les Pentium 2, 3 et 4 d'Intel comprennent respectivement un pipeline de 12, 10 et 20 étages.

Architecture pipeline, Problèmes

- La mise en place d'un pipeline pose plusieurs problèmes.
- En fait, 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 (appelés des aléas) où la performance d'un processeur pipeliné peut être dégradé :
- (1) aléa structurel : qui correspond au cas où deux instructions ont besoin d'utiliser la même ressource du processeur (conflit de dépendance),

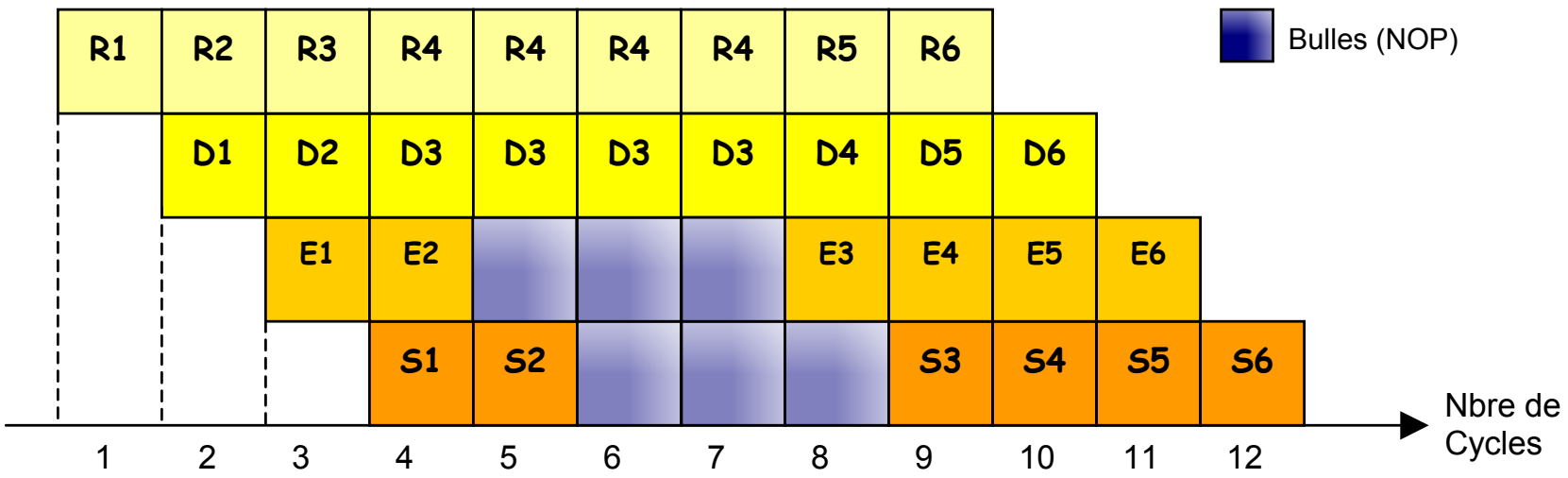
Architecture pipeline, Problèmes

- (2) aléa de données : qui intervient lorsqu'une instruction produit un résultat et que l'instruction suivante utilise ce résultat.
- (3) aléa de contrôle : qui se produit chaque fois qu'une instruction de branchement est exécutée. Lorsqu'une instruction de branchement est chargée, il faut normalement attendre de connaître l'adresse de destination du branchement pour pouvoir charger l'instruction suivante.

Architecture pipeline, Solution

- Lorsqu'un aléa se produit, cela signifie qu'une instruction ne peut continuer à progresser dans le pipeline.
- Pendant un ou plusieurs cycles, l'instruction va rester bloquée dans un étage du pipeline, mais les instructions situées plus en avant pourront continuer à s'exécuter jusqu'à ce que l'aléa ait disparu.
- Le compilateur génère une instruction NOP (No OPeration) émise à la place de l'instruction bloquée.
- Plus le pipeline possède d'étages, plus la pénalité est grande.

Architecture pipeline, Solution



Notion de cache mémoire, Problème posé

- La mémoire n'est plus en mesure de délivrer des informations aussi rapidement que le processeur est capable de les traiter.
- Depuis le début des années 80, une des solutions utilisées pour masquer cette latence est de disposer une mémoire très rapide entre le microprocesseur et la mémoire.
- Elle est appelée cache mémoire.
- On compense ainsi la faible vitesse relative de la mémoire en permettant au microprocesseur d'acquérir les données à sa vitesse propre.

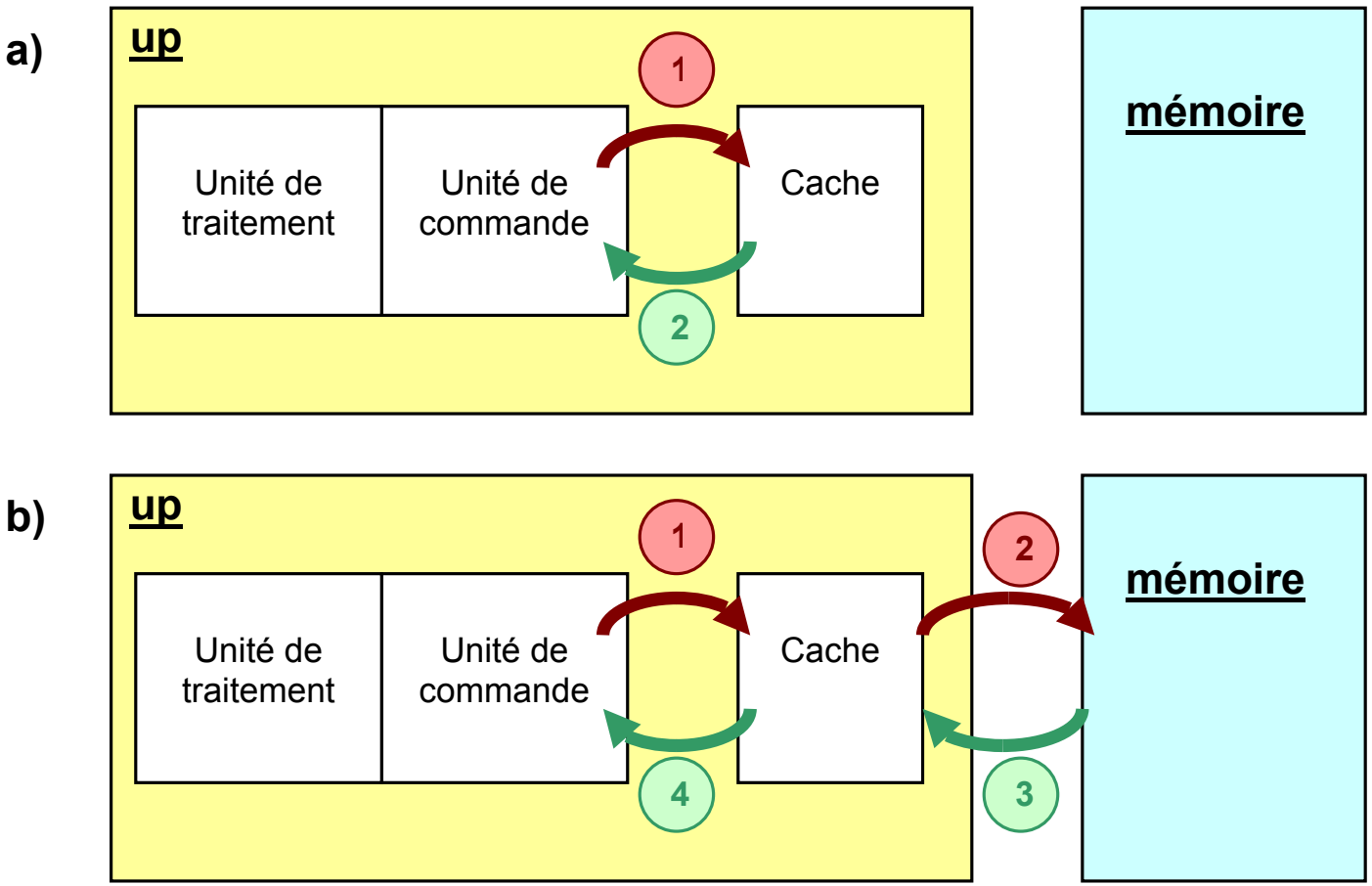
Notion de cache mémoire, Principe

- On la réalise à partir de cellule SRAM (RAM statique) de taille réduite (à cause du coût)
- Le bit mémoire d'une SRAM est composé d'une bascule.
- Sa capacité mémoire est donc très inférieure à celle de la mémoire principale et sa fonction est de stocker les informations les plus récentes ou les plus souvent utilisées par le microprocesseur.
- Au départ cette mémoire était intégrée en dehors du microprocesseur mais elle fait maintenant partie intégrante du microprocesseur et se décline même sur plusieurs niveaux.

Notion de cache mémoire, Principe

- Le principe de cache est très simple : le microprocesseur n'a pas conscience de sa présence et lui envoie toutes ses requêtes comme s'il agissait de la mémoire principale :
  - Soit la donnée ou l'instruction requise est présente dans le cache et elle est alors envoyée directement au microprocesseur. On parle de succès de cache (a).
  - soit la donnée ou l'instruction n'est pas dans le cache, et le contrôleur de cache envoie alors une requête à la mémoire principale. Une fois l'information récupérée, il la renvoie au microprocesseur tout en la stockant dans le cache. On parle de défaut de cache (b).

Notion de cache mémoire, Principe



Notion de cache mémoire, Principe

- Bien entendu, le cache mémoire n'apporte un gain de performance que dans le premier cas.
- Sa performance est donc entièrement liée à son taux de succès. Il est courant de rencontrer des taux de succès moyen de l'ordre de 80 à 90%.

Remarques :

- Un cache utilisera une carte pour savoir quels sont les mots de la mémoire principale dont il possède une copie. Cette carte devra avoir une structure simple.

Notion de cache mémoire, Principe

Remarques (suite) :

- Il existe dans le système deux copies de la même information : l'originale dans la mémoire principale et la copie dans le cache. Si le microprocesseur modifie la donnée présente dans le cache, il faudra prévoir une mise à jour de la mémoire principale.
- Lorsque le cache doit stocker une donnée, il est amené à en effacer une autre. Il existe donc un contrôleur permettant de savoir quand les données ont été utilisées pour la dernière fois. La plus ancienne non utilisée est alors remplacée par la nouvelle.

Processeurs spéciaux : Microcontrôleur

- Ce sont des systèmes à microprocesseurs minimum sur une seule puce.
- Ils contiennent un CPU, de la RAM, de la ROM et des ports d'Entrée/Sorties (parallèles, séries, I2C, etc..).
- Ils comportent parfois aussi des fonctions spécifiques comme des compteurs programmables pour effectuer des mesures de durées, des CAN voir des CNA pour s'insérer au sein de chaînes d'acquisition, des interfaces pour réseaux de terrain, etc ...

Processeurs spéciaux : Microcontrôleur

- Il est adapté pour répondre au mieux aux besoin des systèmes embarquées (appareil électroménagers, chaîne d'acquisition, lecteur carte à puce, etc...).
- Il est par contre généralement moins puissant en terme de rapidité, de taille de données traitables ou de taille de mémoire adressable qu'un microprocesseur.

Processeurs spéciaux : Processeur de signal (DSP)

- Le processeur de signal est beaucoup plus spécialisé.
- Alors qu'un microprocesseur n'est pas conçu pour une application spécifique, le processeur DSP (Digital Signal Processor) est optimisé pour effectuer du traitement numérique du signal (calcul de FFT, convolution, filtrage numérique, etc...).
- Les domaines d'application des D.S.P étaient à l'origine les télécommunications et le secteur militaire.
- Aujourd'hui, les applications se sont diversifiées vers le multimédia (lecteur CD, MP3, etc..) l'électronique grand public (télévision numérique, téléphone portable, etc. . .), l'automatique, l'instrumentation, l'électronique automobile, etc. . .

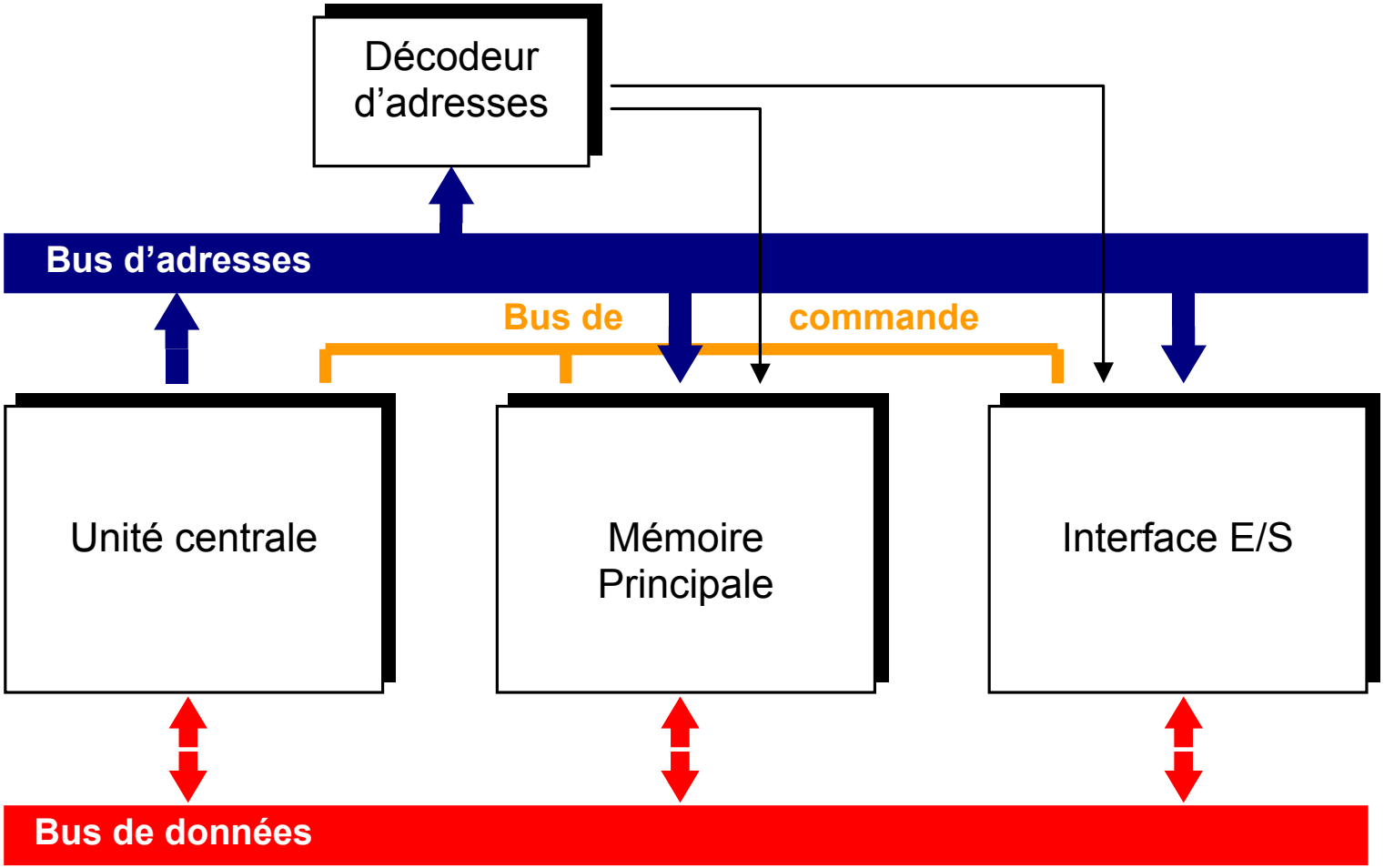
Échanges de données

- La fonction d'un système à microprocesseurs, quel qu'il soit, est le traitement de l'information.
- Il est donc évident qu'il doit acquérir l'information fournie par son environnement et restituer les résultats de ses traitements.
- Chaque système est donc équipé d'une ou plusieurs interfaces d'entrées/sorties permettant d'assurer la communication entre le microprocesseur et le monde extérieur.
- Les techniques d'entrées/sorties sont très importantes pour les performances du système.

Échanges de données

- Rien ne sert d'avoir un microprocesseur calculant très rapidement s'il doit souvent perdre son temps pour lire des données ou écrire ses résultats.
- Durant une opération d'entrée/sortie, l'information est échangée entre la mémoire principale et un périphérique relié au système.
- Cet échange nécessite une interface (ou contrôleur) pour gérer la connexion.
- Plusieurs techniques sont employées pour effectuer ces échanges.

Échanges de données



Interface d'entrée/sortie

- Chaque périphérique sera relié au système par l'intermédiaire d'une interface (ou contrôleur) dont le rôle est de :
  - Connecter le périphérique au bus de données
  - Gérer les échanges entre le microprocesseur et le périphérique
- Pour cela, l'interface est constituée par :
  - Un registre de commande dans lequel le processeur décrit le travail à effectuer (sens de transfert, mode de transfert).
  - Un ou plusieurs registres de données qui contiennent les mots à échanger entre le périphérique et la mémoire
  - Un registre d'état qui indique si l'unité d'échange est prête, si l'échange s'est bien déroulé, etc...
- On accède aux données de l'interface par le biais d'un espace d'adresses d'entrées/sorties.

Techniques d'échange de données

- Avant d'envoyer ou de recevoir des informations, le microprocesseur doit connaître l'état du périphérique. En effet, le microprocesseur doit savoir si un périphérique est prêt à recevoir ou à transmettre une information pour que la transmission se fasse correctement. Il existe 2 modes d'échange d'information :
  1. Le mode programmé par scrutation ou interruption où le microprocesseur sert d'intermédiaire entre la mémoire et le périphérique
  2. Le mode en accès direct à la mémoire (DMA) où le microprocesseur ne se charge pas de l'échange de données.

Échange programmé : Scrutation

- a.Scrutation :
  - Dans la version la plus rudimentaire, le microprocesseur interroge l'interface pour savoir si des transferts sont prêts.
  - Tant que des transferts ne sont pas prêts, le microprocesseur attend. L'inconvénient majeur est que le microprocesseur se retrouve souvent en phase d'attente.
  - Il est complètement occupé par l'interface d'entrée/sortie.
  - De plus, l'initiative de l'échange de données est dépendante du programme exécuté par le microprocesseur.
  - Il peut donc arriver que des requêtes d'échange ne soient pas traitées immédiatement car le microprocesseur ne se trouve pas encore dans la boucle de scrutation.
  - Ce type d'échange est très lent.

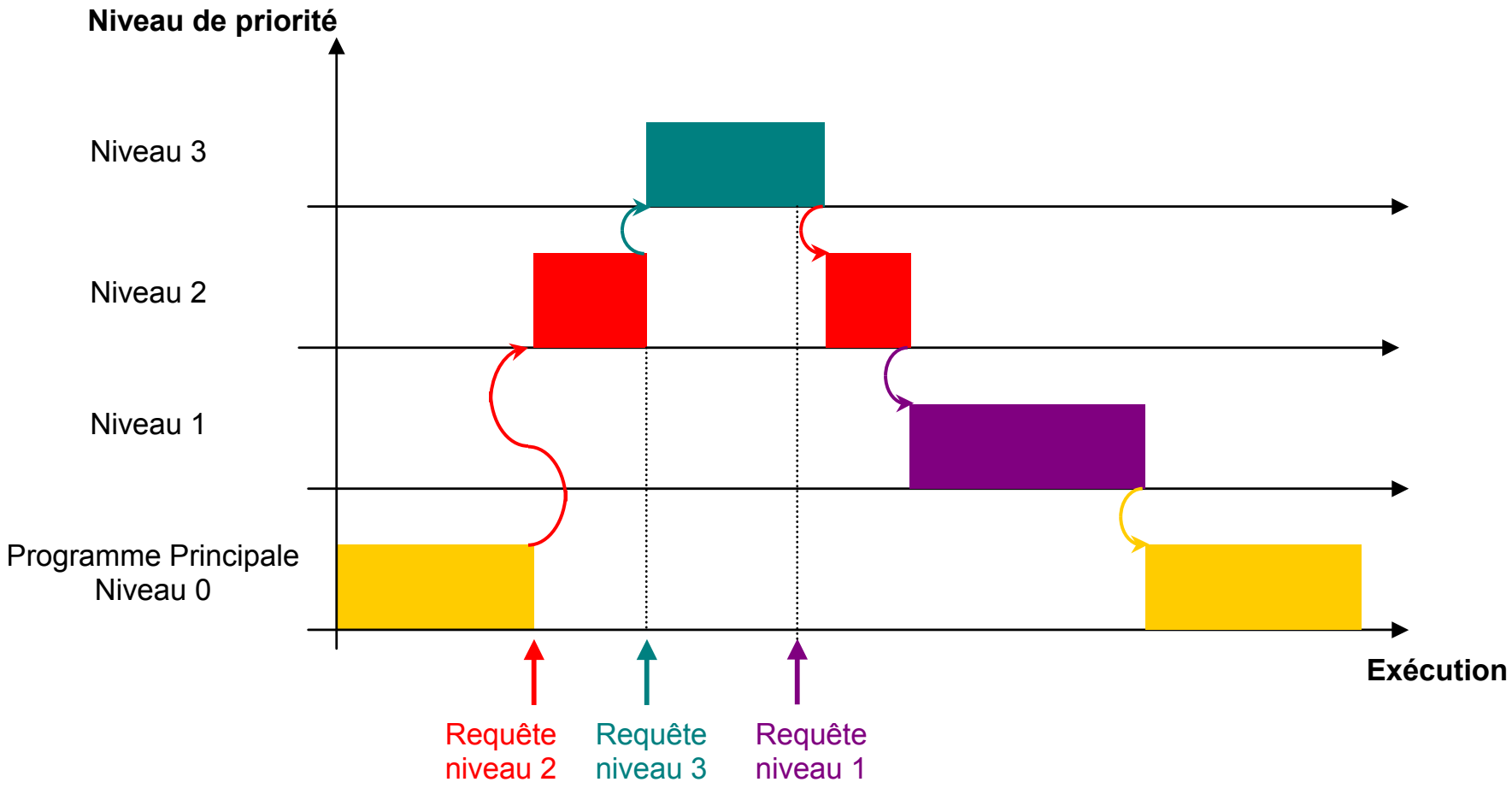
Échange programmé : Interruption

- b.Interruption :
  - Une interruption est un signal, généralement asynchrone au programme en cours, pouvant être émis par tout dispositif externe au microprocesseur.
  - Le microprocesseur possède une ou plusieurs entrées réservées à cet effet.
  - Sous réserve de certaines conditions, elle peut interrompre le travail courant du microprocesseur pour forcer l'exécution d'un programme traitant la cause de l'interruption.
  - Dans un échange de données par interruption, le microprocesseur exécute donc son programme principal jusqu'à ce qu'il reçoive un signal sur sa ligne de requête d'interruption.
  - Il se charge alors d'effectuer le transfert de données entre l'interface et la mémoire.

Principe de fonctionnement d'une interruption

- Avant chaque exécution d'instructions, le microprocesseur examine si il y a eu une requête sur sa ligne d'interruption.
- Si c'est le cas, il interrompt toutes ces activités et sauvegarde l'état présent (registres, PC, accumulateurs, registre d'état) dans un registre particulier appelé pile.
- Les données y sont ''entassées'' comme on empile des livres (la première donnée sauvegardée sera donc la dernière à être restituée ou LIFO).
- Ensuite, il exécute le programme d'interruption puis restitue l'état sauvegardé avant de reprendre le programme principale.

Principe de fonctionnement d'une interruption



Principe de fonctionnement d'une interruption

- Remarques :
  - Certaine source d'interruption possède leur propre autorisation de fonctionnement sous la forme d'un bit à positionner, on l'appelle le masque d'interruption.
  - On peut donc interdire ou autoriser certaines sources d'interruptions, on les appelle les interruptions masquables.
  - Chaque source d'interruption possède un vecteur d'interruption où est sauvegardé l'adresse de départ du programme à exécuter.
  - Les interruptions sont classées par ordre de priorité.
  - Dans le cas où plusieurs interruptions se présentent en même temps, le microprocesseur traite d'abord celle avec la priorité la plus élevée.

Échange direct avec la mémoire (DMA)

- Ce mode permet le transfert de blocs de données entre la mémoire et un périphérique sans passer par le microprocesseur.
- Pour cela, un circuit appelé contrôleur de DMA (Direct Memory Access) prend en charge les différentes opérations.
- Le DMA se charge entièrement du transfert d'un bloc de données. Le microprocesseur doit tout de même :
  - Initialiser l'échange en donnant au DMA l'identification du périphérique concerné
  - Donner le sens du transfert
  - Fournir l'adresse du premier et du dernier mot concernés par le transfert

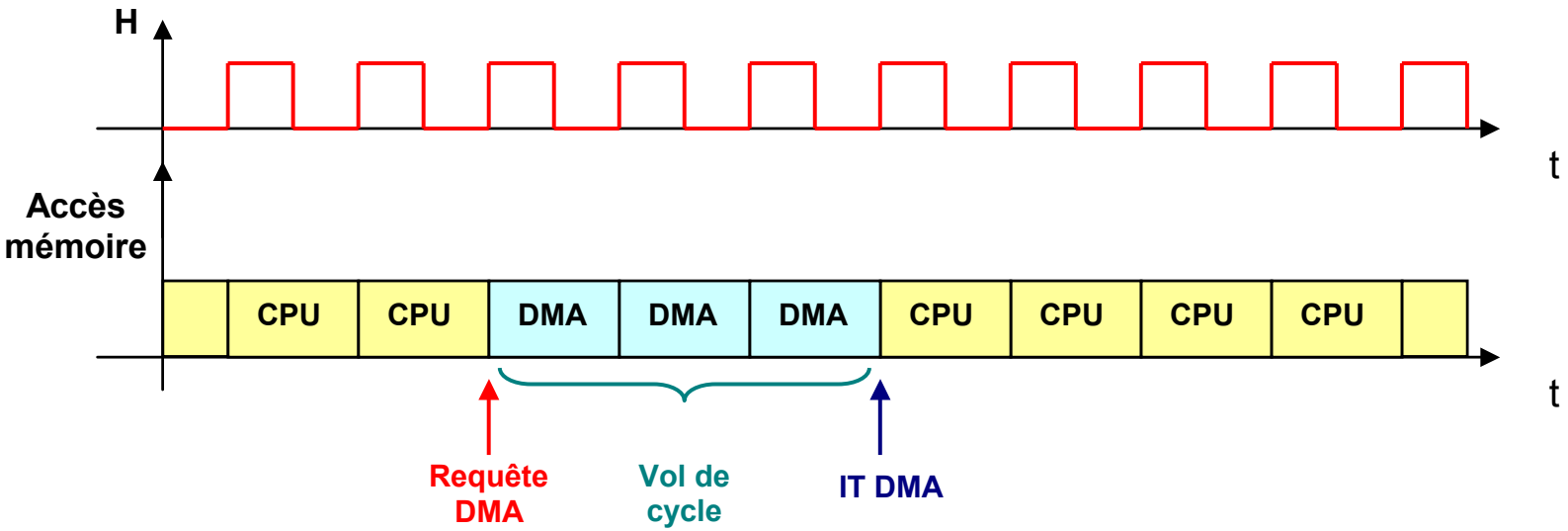
Échange direct avec la mémoire (DMA)

- Un contrôleur de DMA est doté d'un registre d'adresse, d'un registre de donnée, d'un compteur et d'un dispositif de commande (logique câblée).
- Pour chaque mot échangée, le DMA demande au microprocesseur le contrôle du bus, effectue la lecture ou l'écriture mémoire à l'adresse contenue dans son registre et libère le bus.
- Il incrémente ensuite cette adresse et décrémente son compteur.
- Lorsque le compteur atteint zéro, le dispositif informe le processeur de la fin du transfert par une ligne d'interruption.

Échange direct avec la mémoire (DMA)

- Le principal avantage est que pendant toute la durée du transfert, le processeur est libre d'effectuer un traitement quelconque.
- La seule contrainte est une limitation de ses propres accès mémoire pendant toute la durée de l'opération
- Puisqu'il doit parfois retarder certains de ses accès pour permettre au dispositif d'accès direct à la mémoire d'effectuer les siens :
  - il y a apparition de vols de cycle.

Échange direct avec la mémoire (DMA)



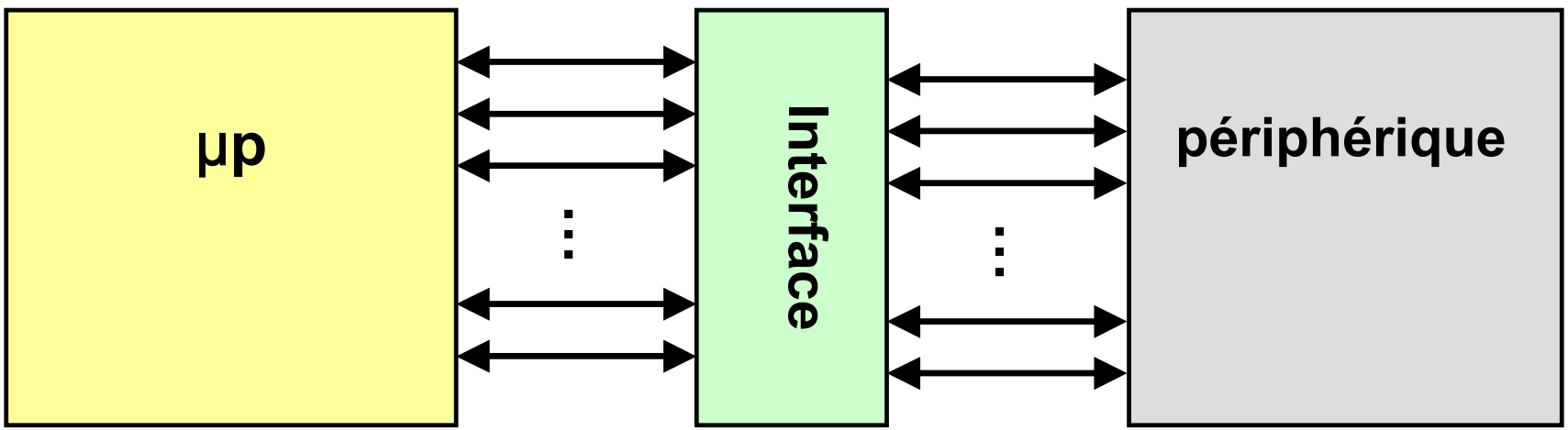
Types de liaisons

- Les systèmes à microprocesseur utilisent deux types de liaison différentes pour se connecter à des périphériques :
  1. Liaison parallèle
  2. Liaison série
- On caractérise un type de liaison par sa vitesse de transmission ou débit (en bit/s).

Liaison parallèle

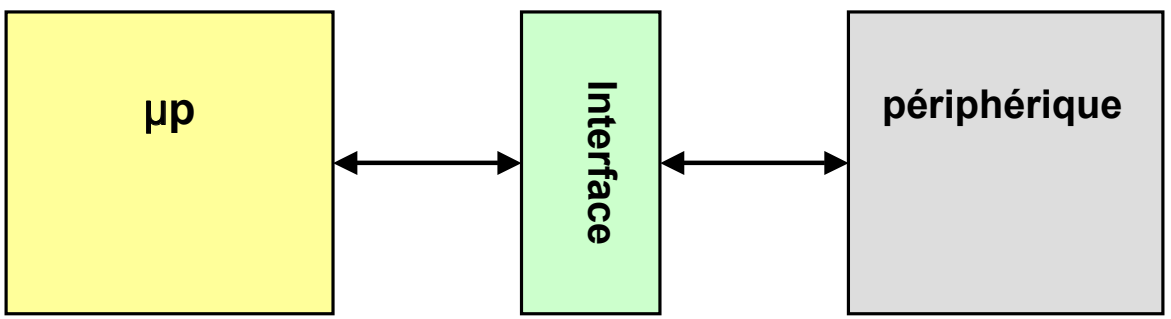
- Dans ce type de liaison, tous les bits d'un mot sont transmis simultanément.
- Ce type de transmission permet des transferts rapides mais reste limitée à de faibles distances de transmission à cause du nombre important de lignes nécessaires (coût et encombrement) et des problèmes d'interférence électromagnétique entre chaque ligne (fiabilité).
- La transmission est cadencée par une horloge
- Exemple : Bus PCI, AGP dans un PC.

Liaison parallèle



Liaison série

- Dans ce type de liaison, les bits constitutifs d'un mot sont transmis les uns après les autres sur un seul fil.
- Les distances de transmission peuvent donc être plus beaucoup plus importantes mais la vitesse de transmission est plus faible.
- Sur des distances supérieures à quelques dizaines de mètres, on utilisera des modems aux extrémités de la liaison.



Liaison série

- La transmission de données en série peut se concevoir de deux façons différentes :
  1. Mode synchrone : l'émetteur et le récepteur possède une horloge synchronisée qui cadence la transmission. Le flot de données peut être ininterrompu.
  2. Mode asynchrone : la transmission s'effectue au rythme de la présence des données. Les caractères envoyés sont encadrés par un signal start et un signal stop.

Principe de base d'une liaison série asynchrone

- Afin que les éléments communicants puissent se comprendre, il est nécessaire d'établir un protocole de transmission. Ce protocole devra être le même pour chaque élément.
- Paramètres rentrant en jeu :
- Longueur des mots transmis : 7 bits (code ASCII) ou 8 bits
- Vitesse de transmission : les vitesses varient de 110 bit/s à 128000 bit/s et détermine les fréquences d'horloge de l'émetteur et du récepteur.

Principe de base d'une liaison série asynchrone

- Parité : le mot transmis peut être suivis ou non d'un bit de parité qui sert à détecter les erreurs éventuelles de transmission. Il existe deux types de parité : paire ou impaire. Si on fixe une parité paire, le nombre total de bits à 1 transmis (bit de parité inclus) doit être paire. C'est l'inverse pour une parité impaire.
- Bit de start : la ligne au repos est à l'état 1 (permet de tester une coupure de la ligne). Le passage à l'état bas de la ligne va indiquer qu'un transfert va commencer. Cela permet de synchroniser l'horloge de réception.
- Bit de stop : après la transmission, la ligne est positionnée à un niveau 1 pendant un certains nombre de bit afin de spécifier la fin du transfert. En principe, on transmet un et ou 2 bits de stop.

Principe de base d'une liaison série asynchrone

- Déroulement d'une transmission :
- Les paramètres du protocole de transmission doivent toujours être fixés avant la transmission.
- En l'absence de transmission, la liaison est au repos au niveau haut pour détecter une éventuelle coupure sur le support de transmission.
- Une transmission s'effectue de la manière suivante :
  1. L'émetteur positionne la ligne à l'état bas : c'est le bit de start.
  2. Les bits sont transmis les un après les autres, en commençant par le bit de poids fort.
  3. Le bit de parité est éventuellement transmis.
  4. L'émetteur positionne la ligne à l'état haut : c'est le bit de stop.

Principe de base d'une liaison série asynchrone

- Exemple : transmission d'un mot de 7 bits $(0110100)_2$ – Parité impaire (nombre 1 est impair) – 1 bit de Stop
- Horloge : $F = \frac{1}{\triangle}\ Hz$, Vitesse de transmission (débit) : $v = \frac{1}{\triangle}\ bits/s$