



# 1. Architecture d'une machine numérique

# + Introduction

- Ordinateur = machine de traitement de l'information
- Un ordinateur est capable:
  - D'acquérir l'information
  - De stocker l'information
  - De transformer l'information en effectuant des traitements
  - De restituer l'information sous une autre forme
- Information = tout ensemble de données (texte, son , images, mais aussi instructions constituant un programme informatique)
- Toute information est manipulée sous forme binaire

# + Introduction

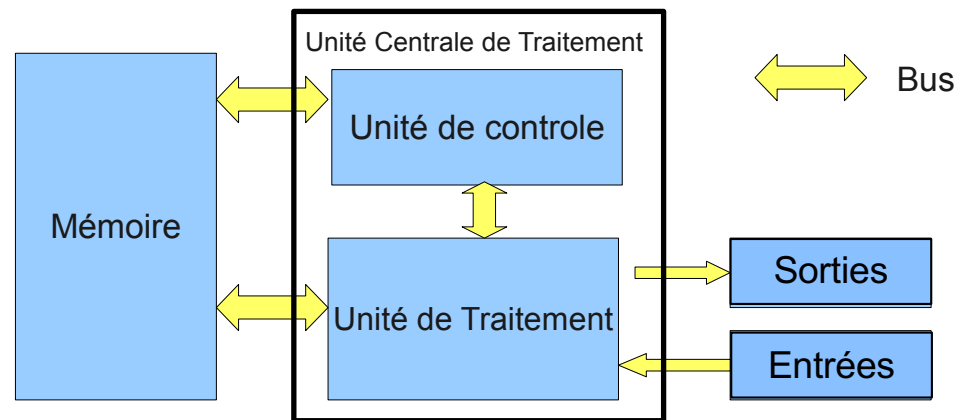
- L'architecture d'un calculateur numérique tel qu'un ordinateur, correspond à l'organisation interne de ses constituants
- Fonctionnellement l'architecture d'un ordinateur peut être décomposée en 3 sous-systèmes:
  - L'unité centrale de traitement (UCT) ou processeur
  - La mémoire
  - Les systèmes d'entrée-sortie
- Ces éléments sont reliés entre eux par un ensemble de bus (système de communication entre composants)

# + Introduction

- L'UCT permet l'exécution d'un programme constitué d'un ensemble d'instructions pour traiter des données
- La mémoire stocke l'ensemble des instructions constituant les programmes ainsi que les données sur lesquelles les instructions vont être exécutées
- Les systèmes d'entrée-sortie permettent au processeur et à la mémoire de communiquer avec des éléments externes (disques durs, cartes graphiques, ...)
- Les bus permettent à ces trois éléments de communiquer entre eux pour des échanges de données

# + Architecture de Von Neumann

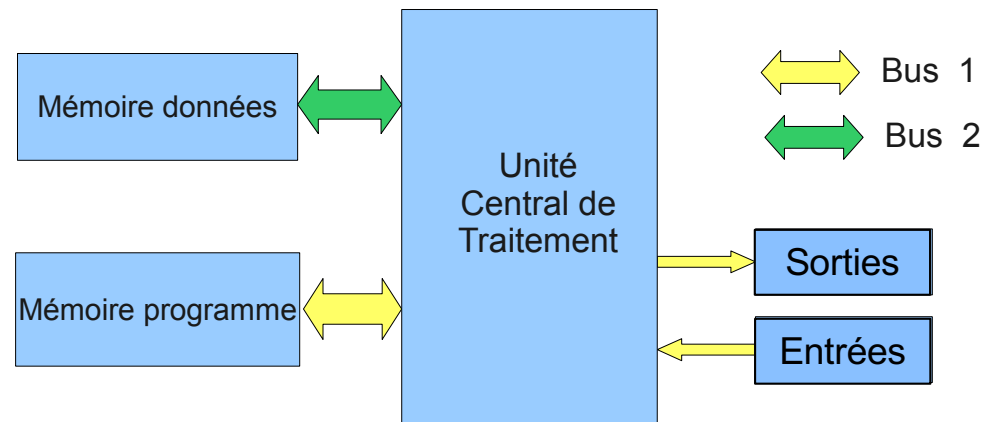
- Architecture de base d'un ordinateur pensée par John Von Neumann (concepteur de l'ENIAC) en 1945



- Séparation du processeur et de la mémoire, reliés par un bus de communication
- L'architecture de Von Neuman stocke dans une même mémoire les données et les instructions à exécuter

# + Architecture de Harvard

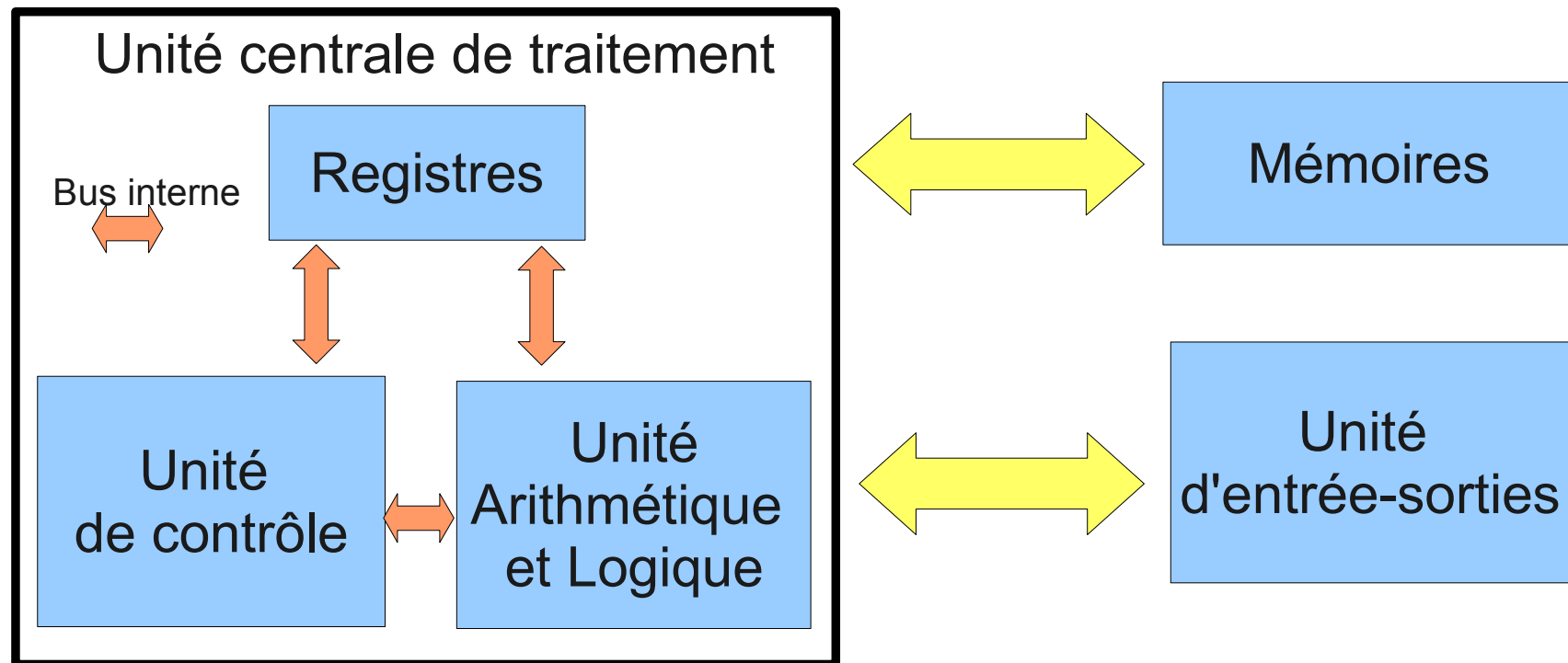
- Séparation physique des mémoires contenant les données des mémoires contenant les programmes (instructions)
- Les deux mémoires sont accessibles par des bus distincts



# + Architecture de Von Neumann vs architecture de Harvard

- Avantages et inconvénients des deux architectures:
  - Vitesse de transfert de la mémoire à UCT limitée par la vitesse des bus
  - L'architecture de Harvard permet une augmentation du débit des données par l'accès simultanée via les 2 bus
  - L'architecture de Harvard est de ce fait adaptée aux applications nécessitant un traitement à la volée des données

# + Structure d'un système de Von Neumann





# + L'unité centrale de traitement

- Elle est constituée de:
  - L'unité arithmétique et logique (UAL) qui effectue les opérations
  - Registres permettant le stockage temporaire de données issues ou allant vers l'UAL
  - L'unité de contrôle permettant de gérer les ordres à envoyer aux autres éléments de l'UCT
  - Le bus interne permettant de véhiculer les informations entre les éléments constituant l'UCT

# + La mémoire

- Entité permettant de stocker temporairement ou à plus long terme des informations (données, instructions ou résultats intermédiaires de calculs)
- Les différentes catégories de mémoire sont:
  - Registres: type de mémoire à capacité réduite présente dans l'UCT
  - Mémoire cache: mémoire intermédiaire accessible directement par le microprocesseur
  - Mémoire centrale: mémoire principale accessible via le bus
  - Mémoire de masse: accessible par les dispositifs d'entrée-sortie avec un temps d'accès plus long

# + Choix de la mémoire

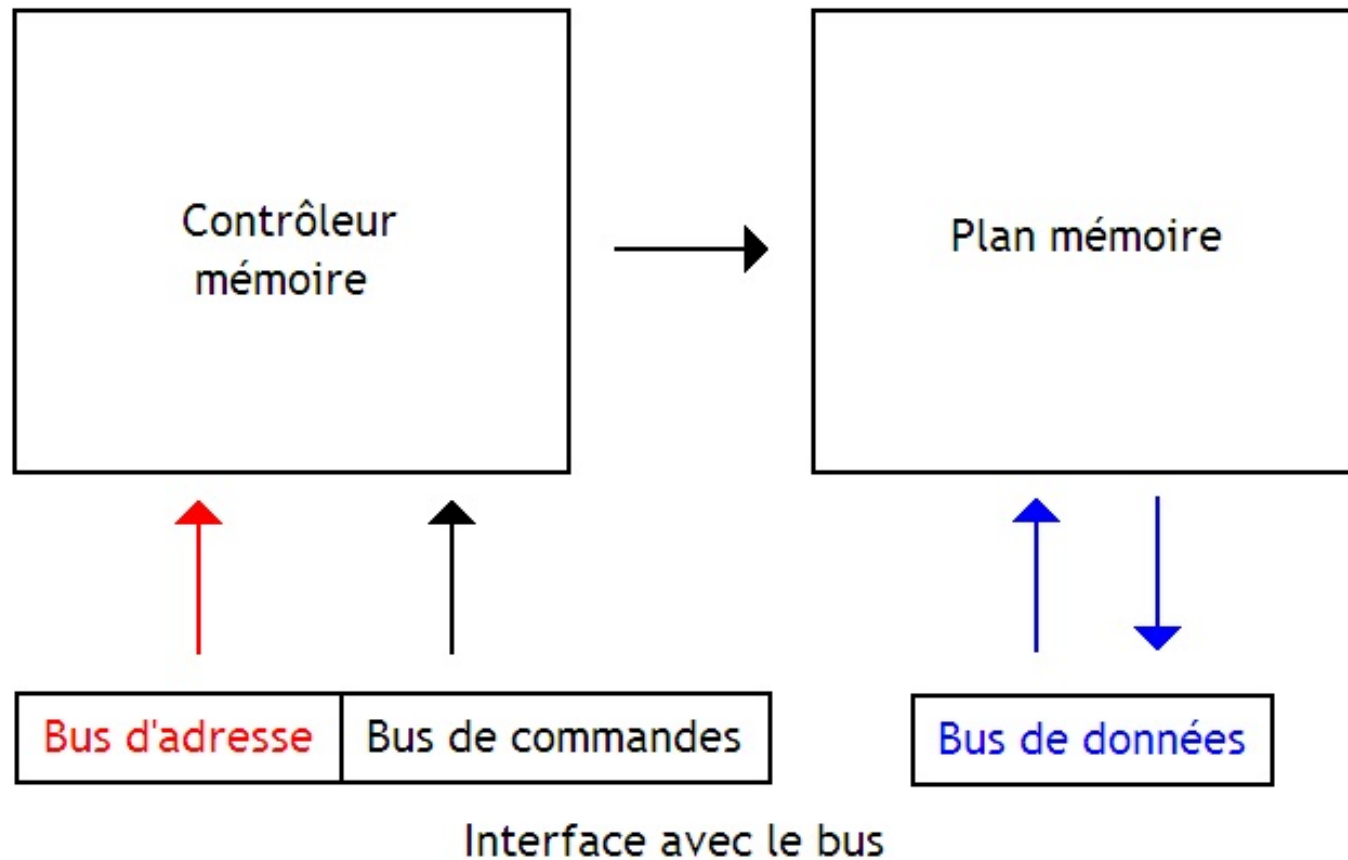
- Le choix d'un type de mémoire résulte d'un compromis entre:
  - La capacité mémoire (quantité d'information binaire que l'on peut stocker)
  - Le temps de réponse (lié également à l'accessibilité et à la proximité à l'UCT)
  - Le coût

Ce choix revient au concepteur de l'architecture

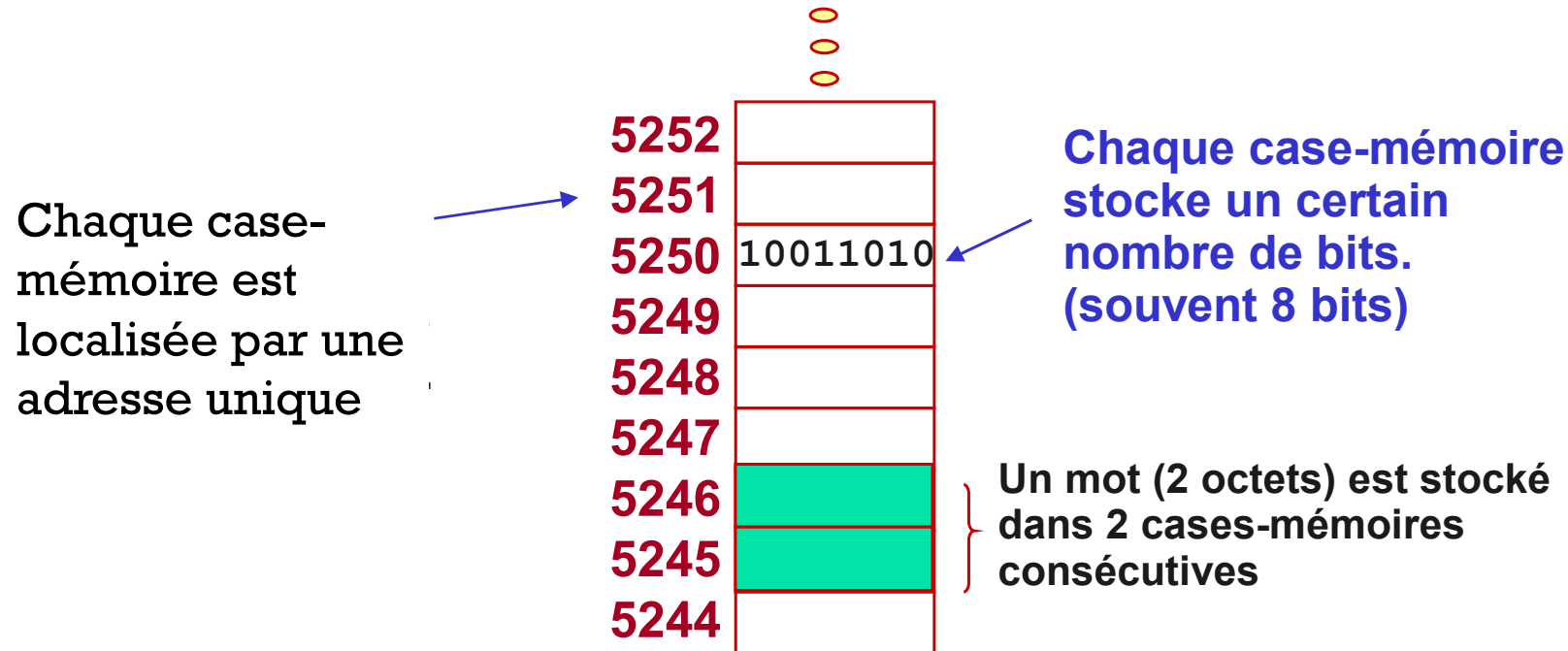
# + Structure de la mémoire

- La mémoire est composée de cases mémoire de taille fixe (généralement 1 octet, c.a.d 8 bits)
- Les cases mémoire sont les briques élémentaires de la mémoire
- L'ensemble des cases mémoire permet de stocker des instructions et des données (information codée en binaire)
- Chaque case mémoire est repérée par une position ou numéro que l'on appelle "adresse" et contient une information sur un octet
- Si une information codée sur plusieurs octets doit être stockée, alors plusieurs cases mémoire consécutives sont utilisées

# + Structure de la mémoire



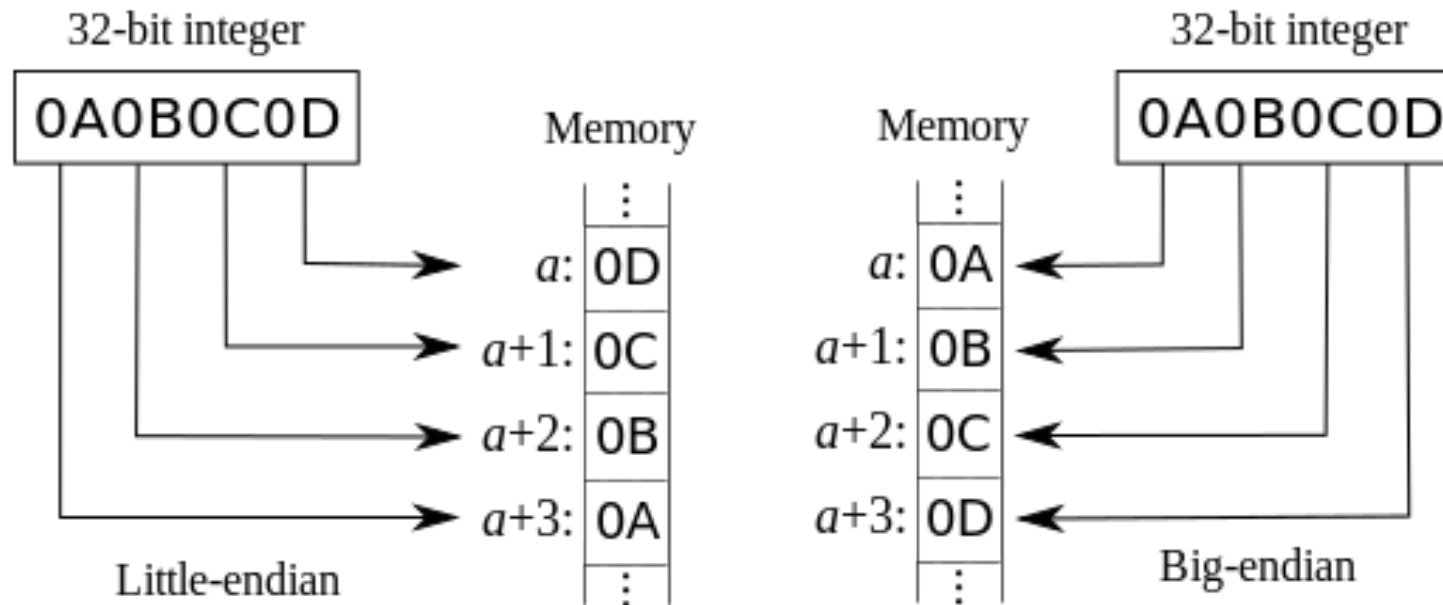
# + Structure de la mémoire



# + Stockage en mémoire

- La mémoire est organisée en cellule élémentaire d'un octet, mais il est possible d'y stocker une information de 16 bits voire 32 bits, en utilisant des cases mémoire consécutives d'adresses croissantes
- Il y a deux manières (deux sens) de stocker les octets d'information:
  - Mode big-endian: octet de poids fort stocké avant l'octet de poids faible
  - Mode little-endian: octet de poids faible stocké avant l'octet de poids fort

# + Stockage en mémoire





# + Stockage en mémoire

- Le mode *big endian* accélère les opérations qui nécessitant de regarder en premier les bits de poids forts (exemple: recherche du signe, comparaison de deux entiers,...)
- Le mode *little endian* accélère les opérations qui nécessitant de regarder en premier les bits de poids faible (exemple: l'addition)

# + Stockage en mémoire

- Pour accéder à une information en mémoire, le bus d'adresse devra contenir l'adresse la plus basse où est stockée cette information
- Idéalement les adresses doivent être adaptées à la taille du bus de données.  
Exemple: Pour un bus de données de 32 bits, les adresses contenant une information sont des multiples de 4 octets

# + Stockage de l'information en mémoire

- Exemple: on stocke l'information (09ABCDEF) exprimée en hexadécimal sur 4 octets à l'adresse 1000h

1006h	
1005h	
1004h	
1003h	09
1002h	AB
1001h	CD
1000h	EF
0FFFh	

mode little-endian

1006h	
1005h	
1004h	
1003h	EF
1002h	CD
1001h	AB
1000h	09
0FFFh	

mode big-endian

# + Les bus

- Un bus est défini comme un ensemble de support de l'information (fils électriques par exemple) véhiculant une information binaire
- Sur  $n$  fils conducteurs peuvent être transportés  $n$  signaux binaires
- Si on lit en binaire l'information fournie par un bus de taille  $n$ , on obtient un nombre allant de 0 à  $2^n - 1$

# + Les bus

- On distingue 3 types de bus:
  - Le **bus de données** qui est un bus bidirectionnel véhiculant les données du microprocesseur vers un composant ou d'un composant vers le microprocesseur
  - Le **bus de commande** (unidirectionnel) constitué d'une nappe de fils de « commande » permettant la synchronisation et la commande des boîtiers mémoire (sélection et ordres de lecture/écriture) et entrée/sortie par le microprocesseur
  - Le **bus d'adresse** qui est un bus unidirectionnel (du microprocesseur vers les autres composants) véhiculant l'adresse d'une brique élémentaire de la mémoire

# + Les bus: mécanisme d'échange avec la mémoire

- Le transfert d'information entre l'UCT et la mémoire peut se faire selon 2 sens:
  - En lecture: l'UCT souhaite lire une information (instruction d'un programme ou donnée) dans la mémoire
  - En écriture: l'UCT souhaite écrire une information présente sur le bus de données, dans la mémoire

# + Les bus: mécanisme d'échange avec la mémoire

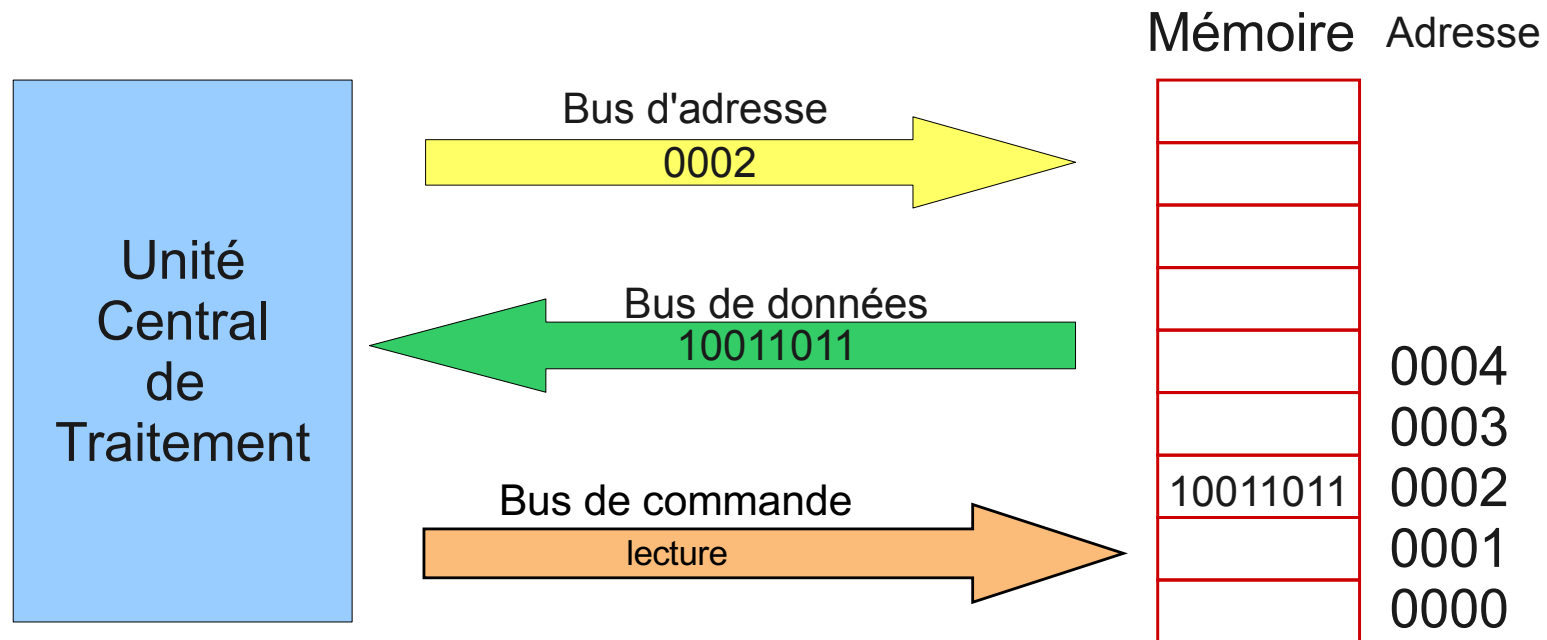
## ■ Mécanisme de lecture:

- Lorsque le microprocesseur veut lire le contenu d'une case mémoire il doit indiquer à quelle adresse se trouve cette case en mettant cette adresse sur le bus d'adresse
- Le contenu de la case mémoire sélectionnée par l'adresse indiquée dans le bus d'adresse, est mis sur le bus de données

## ■ Mécanisme d'écriture quasi identique à celui de la lecture:

- La donnée à stocker est mise sur le bus de données
- L'adresse de la case mémoire où doit être stockée cette donnée, sur le bus d'adresse

# + Les bus: mécanisme d'échange avec la mémoire



Exemple de lecture en mémoire



# + Adressage mémoire

- La taille de la mémoire adressable est limitée par la taille du bus d'adresse
- Suivant les microprocesseurs le bus d'adresse est composé de 16, 32 ou 64 fils
- Avec un bus d'adresse de 16 bits on peut adresser une mémoire de 64 kilo-octets, 1 Mo avec un bus de 20 bits, et 4 Go avec un bus de 32 bits.
- Il faut distinguer largeur du bus d'adresses et largeur du bus de données qui peuvent être différentes
- **Attention: quand on parle de processeur n bits, on parle de la taille de ses registres !**
- Exemple: évolution des microprocesseurs INTEL
  - 8086 (1978): microprocesseur 16 bits
  - 80386 (1985): microprocesseur 32 bits
  - Core i5 / i7: microprocesseur 64 bits

# + Adressage mémoire

- Un accès mémoire se fait selon une certaine taille ce qui permet d'accéder à plusieurs octets contigus en une seule fois
- Si on désire lire 2 octets à l'adresse 123h sur une architecture utilisant un mode de stockage little-endian, l'octet de poids faible est lu à l'adresse 123h et l'octet de poids fort à l'adresse 124h