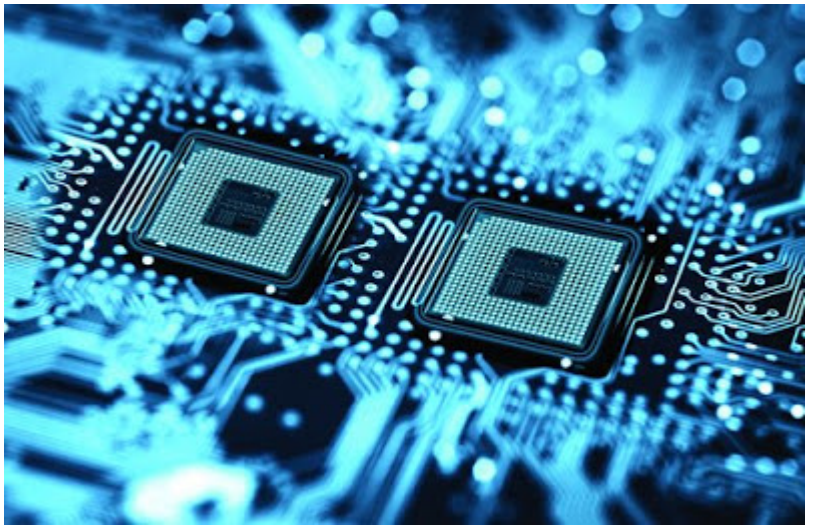
ARCHITECTURE MATERIELLE



https://www.tutorialhall.com/2017/12/shift-micro-operation-in-computer-architecture.html

Objectifs de la séquence

- Les différents *modèles d'architectures matérielles* existantes, les composants d'un ordinateur et leur rôle ;
- Le *codage de l'information* ;
- Le *lien* entre l'*architecture matérielle* et le fonctionnement d'un *système d'exploitation* ;
- Le fonctionnement d'*un processeur* et les principales techniques d'optimisations existantes.

Agenda d’aujourd’hui

Notions abordées :

- Les composants fondamentaux d’un ordinateur ;
- L’idée générale du fonctionnement d’un ordinateur ;
- Modèles standards d’architecture des ordinateurs
  - Van Neumann
  - Harvard
  - Harvard modifiée

Un peu de vocabulaire avant de commencer

Qu’y-a-t-il à l’intérieur d’un ordinateur ?

Un peu de vocabulaire avant de commencer

Carte mère

La carte mère est un circuit imprimé qui permet la communication
Des éléments de l'ordinateur entre eux

Un peu de vocabulaire avant de commencer

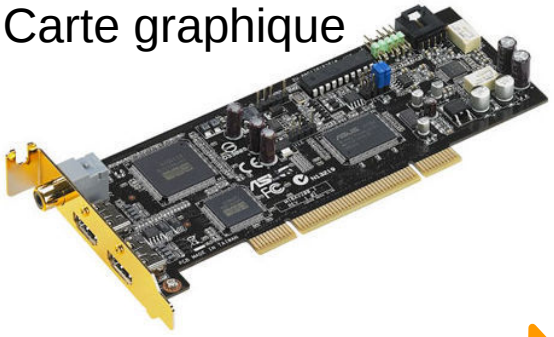
Carte graphique

PCI-E

Carte mère

Socket

Processeur



Carte son

Mémoire de masse

Mémoire Vive

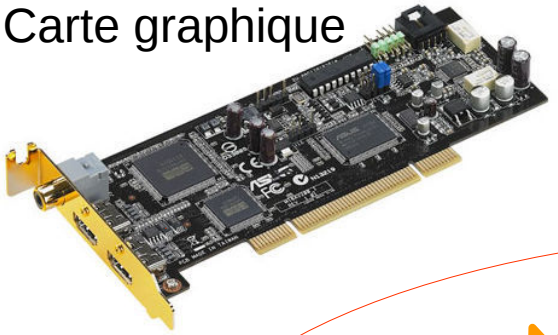
Un peu de vocabulaire avant de commencer

Carte graphique

PCI-E

Socket

Processeur



Carte son

Carte mère

Contient vos applications
Et vos fichiers

Mémoire de masse

Mémoire Vive

Un peu de vocabulaire avant de commencer

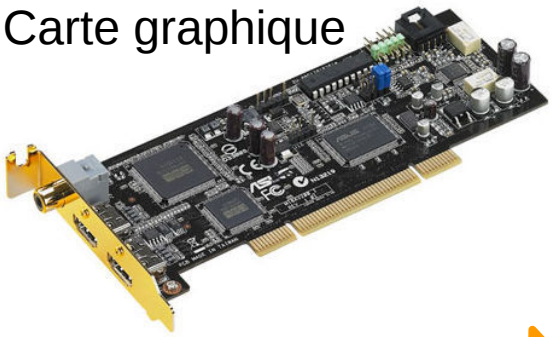
Carte graphique

PCI-E

Carte mère

Socket

Processeur



Carte son

Mémoire de masse

Contient les logiciels
En fonctionnement
+ données temporaires
(page web, images,..)

Mémoire Vive

Un peu de vocabulaire avant de commencer

Exécute les applications
Fait des calculs sur les données

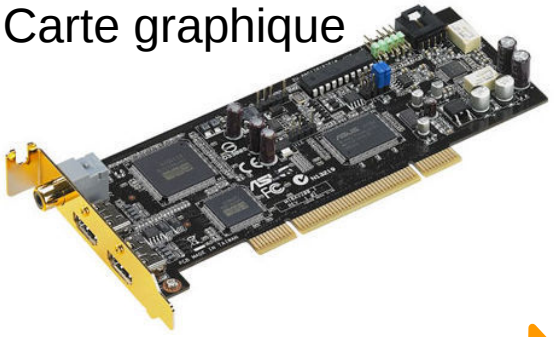
Carte graphique

PCI-E

Carte mère

Socket

Processeur



Carte son

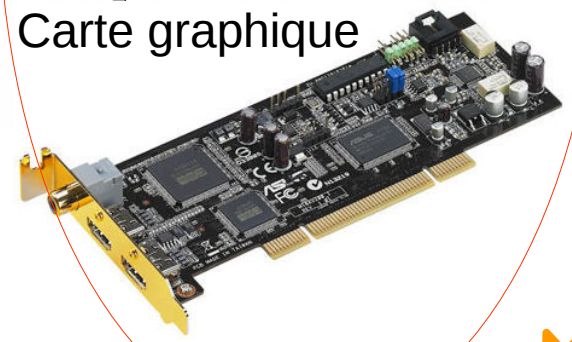
Mémoire de masse

Mémoire Vive

Un peu de vocabulaire avant de commencer

Communique avec l'utilisateur

Carte graphique



Carte son

PCI-E

Carte mère

Socket

Processeur

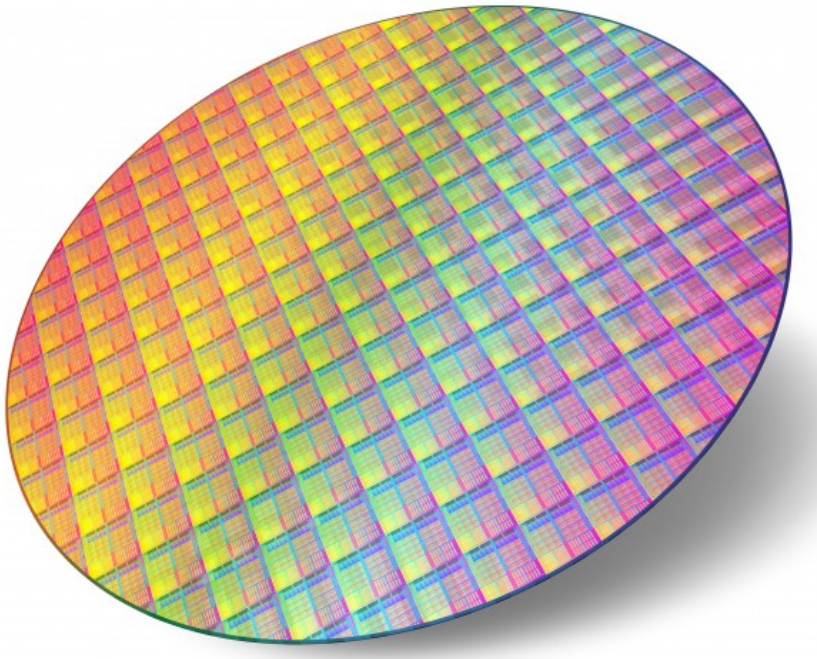
Mémoire de masse

Mémoire Vive

Fabrication des circuits imprimés

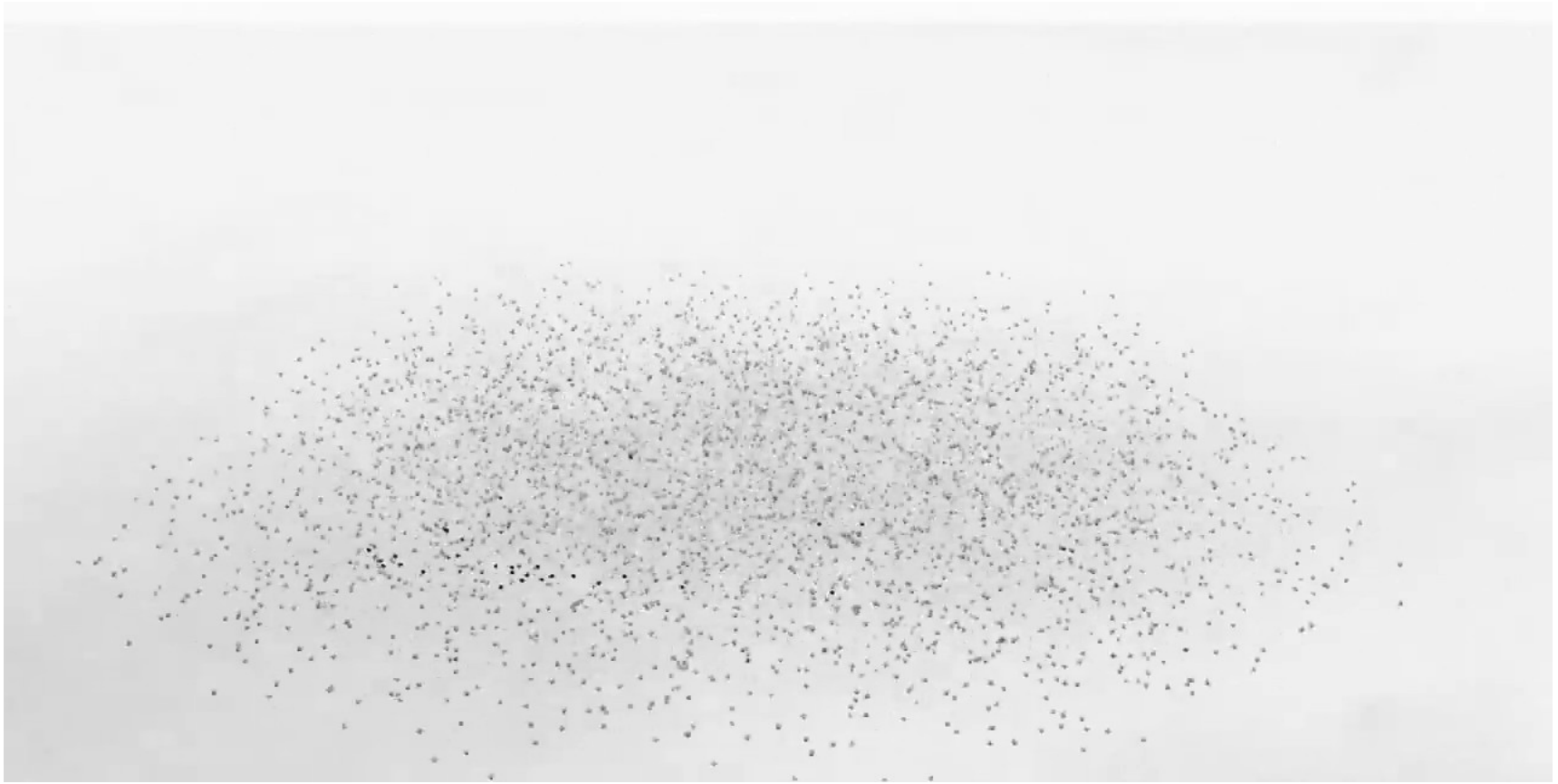
- Wafer = plaque de semi-conducteurs, généralement en silicium
- Le silicium est abondant :

  1/4 de la masse de la croûte terrestre.



- Peut être dopé pour fabriquer des semi-conducteurs = introduction de Phosphore ou de Bore
- Fabrication de processeurs est complexe (peut atteindre plus de 300 étapes intermédiaires)

Fabrication de circuits imprimés

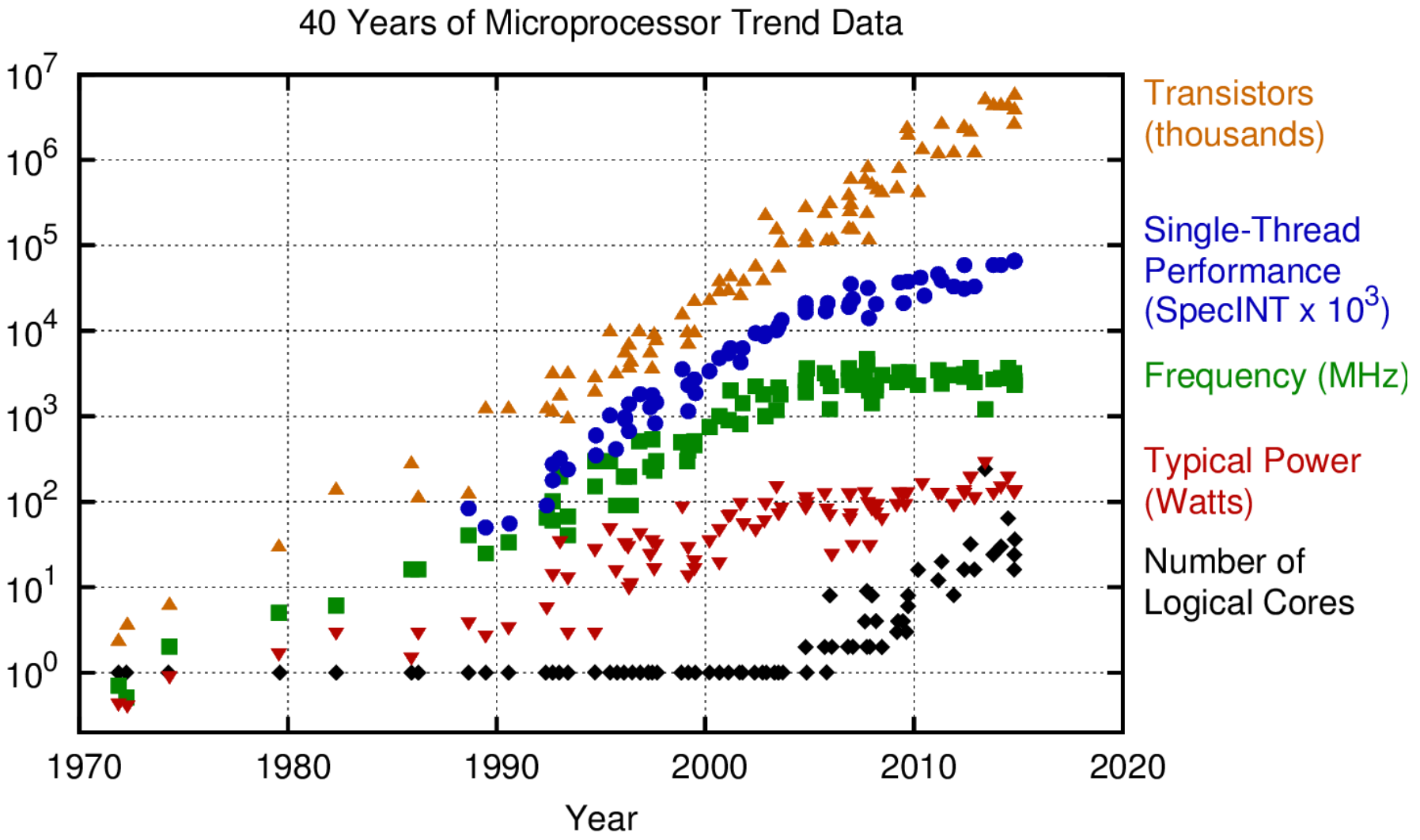


https://www.youtube.com/watch?v=d9SWNLZvA8g&feature=youtu.be

Introduction générale

Evolution et nouvelles tendances dans le domaine de l’informatique

Evolution des processeurs depuis 40 ans



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp

Récapitulatif

- L’évolution des processeurs, c’était (avant 2005) :
  - La course à la densité de transistors ;
  - L’augmentation de la fréquence de fonctionnement.
- Aujourd’hui, c’est :
  - L’augmentation du nombre de cœurs ;
  - L’efficacité énergétique ;
  - L’ajout d’accélérateurs matériels (GPU, FPGA, …) ;
  - Le calcul déporté.

Nouvelles tendances : Le calcul déporté

- Architecture traditionnelle

Carte graphique dernier cri	500 €
Processeur dernière génération	200 €
Carte mère adaptée à la configuration de l'ordinateur	230 €

- Architecture déportée

  Exemple Nvidia Grid VCA

  Machine à 40.000 €

Nouvelles tendances : Le calcul déporté

- Domaines d'application :
  - Entreprises
  - Services cloud
  - Jeux vidéos

- Infiniband
  - Réseau ultra rapide pour le calcul déporté.

    Horizon 2023 : 3.000 Gbit/s

    (Ethernet : max 400

- Architecture déportée

  Exemple Nvidia Grid VCA

  Machine à 40.000 €

Les différents systèmes d’exploitation

- Propriétaires (le code source n’est pas disponible librement) :

- Open Source (code disponible et amélioré par une communauté) :

Linux

Intérêt d’un OS

- Un ordinateur sans logiciel = Amas de silicium.
- Implémentation bare metal (sans OS) :
  - Permet d’exécuter du code de base.

```
int main()
{
        while(1)
        {
                char input;
                printf("Select command:\n");
                getchar(&input);
                switch(input)
                {
                        case 'A':
                                function_A();
                        break;
                        case 'B':
                                function_B();
                        break;
                        case 'C':
                                function_C();
                        break;
                        case 'D':
                                function_D();
                        break;

                }
        }
}
```

Vocabulaire

- Logiciel d’application :
  - Logiciel permettant de résoudre un problème spécifique.
  - Calculatrice → faire des calculs simples.
  - Suite bureautique → Dématérialisation des documents.
  - Navigateur web → Accéder à des services distants.
- Logiciel utilitaire :
  - Permet de développer des applications.
  - Environnement de développement (IDE):
    - Code::Blocks
    - Eclipse
    - Bloc Note !
  - Compilateurs :
    - Traduit un langage dit de haut niveau (Langage C,Java,Ada,Python,…) en langage machine
- Langage machine :
  - Très bas niveau, opérations de base (addition, soustraction, lecture/ecriture dans une mémoire).
  - Tout part du langage machine !

Un ordinateur, une histoire d'abstraction

Utilisateur lambda

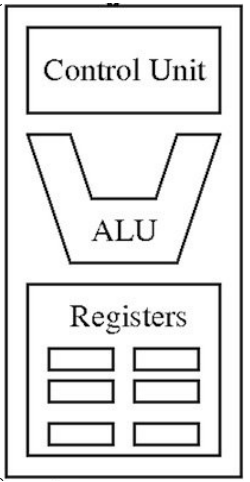
Logiciel d'application

Logiciel Utilitaire

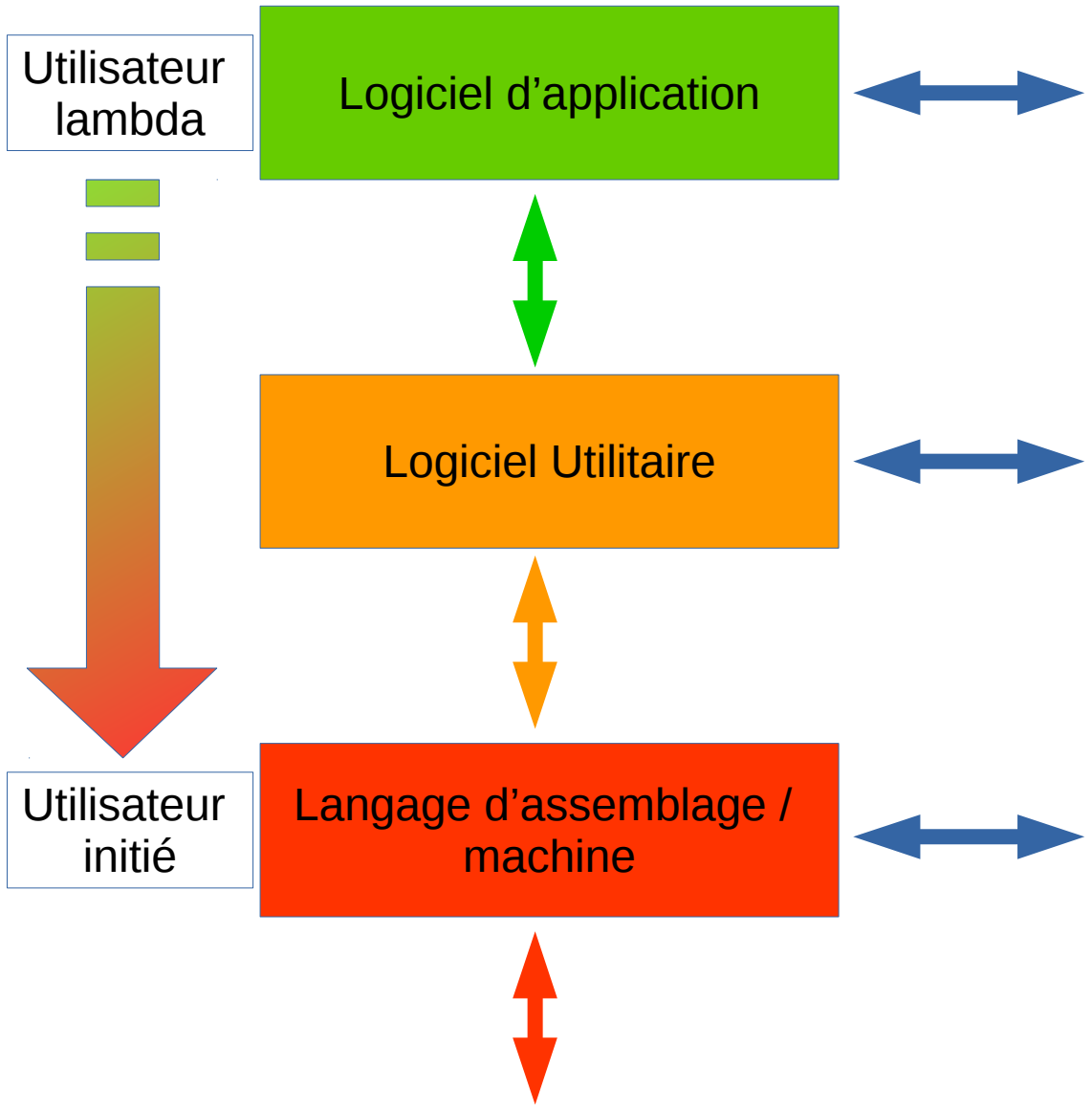
Algorithmes, fonctionnalités

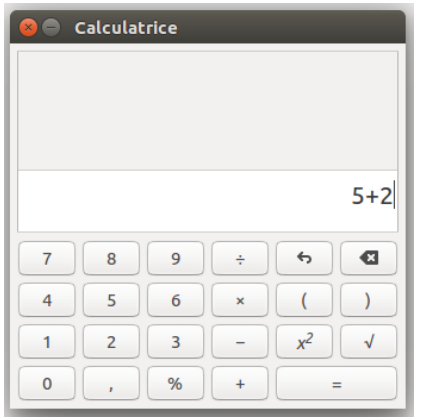
Utilisateur initié

Langage d'assemblage / machine



Un ordinateur, une histoire d'abstraction






```
uint32_t add (uint32_t a, uint32_t b)
{
        uint32_t c;
        c = a + b;
        return c
}
```

```
  PUSH {R3,R4,R5,R6,R7,PC}

  LDR R3,[R1,#0]
  LDR R4,[R2,#0]
  ADD R3,R3,R4
  STR R3,[R0,#0]

  POP  {R3,R4,R5,R6,R7,LR}
```

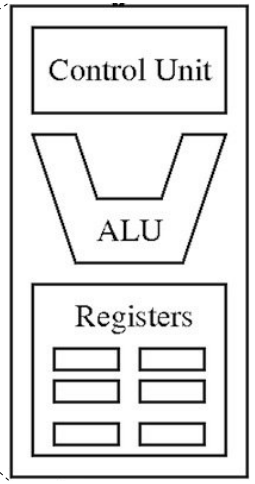
Pourquoi peut-on exécuter plusieurs applications ?

- C’est le rôle du système d’exploitation de gérer la multiprogrammation.

L’architecture d’un ordinateur

- Il existe 3 architectures d’ordinateurs standards :
  - L’architecture Von Neumann
  - L’architecture Harvard
  - L’architecture Harvard modifiée

Architecture Von Neumann, en bref



Processeur (CPU)

Unité arithmétique et logique → Réalise des opérations de base sur les données

Unité de contrôle → Envoi de signaux De contrôle aux Autres unités

Mémoire → Garde en mémoire Les données et Les programmes

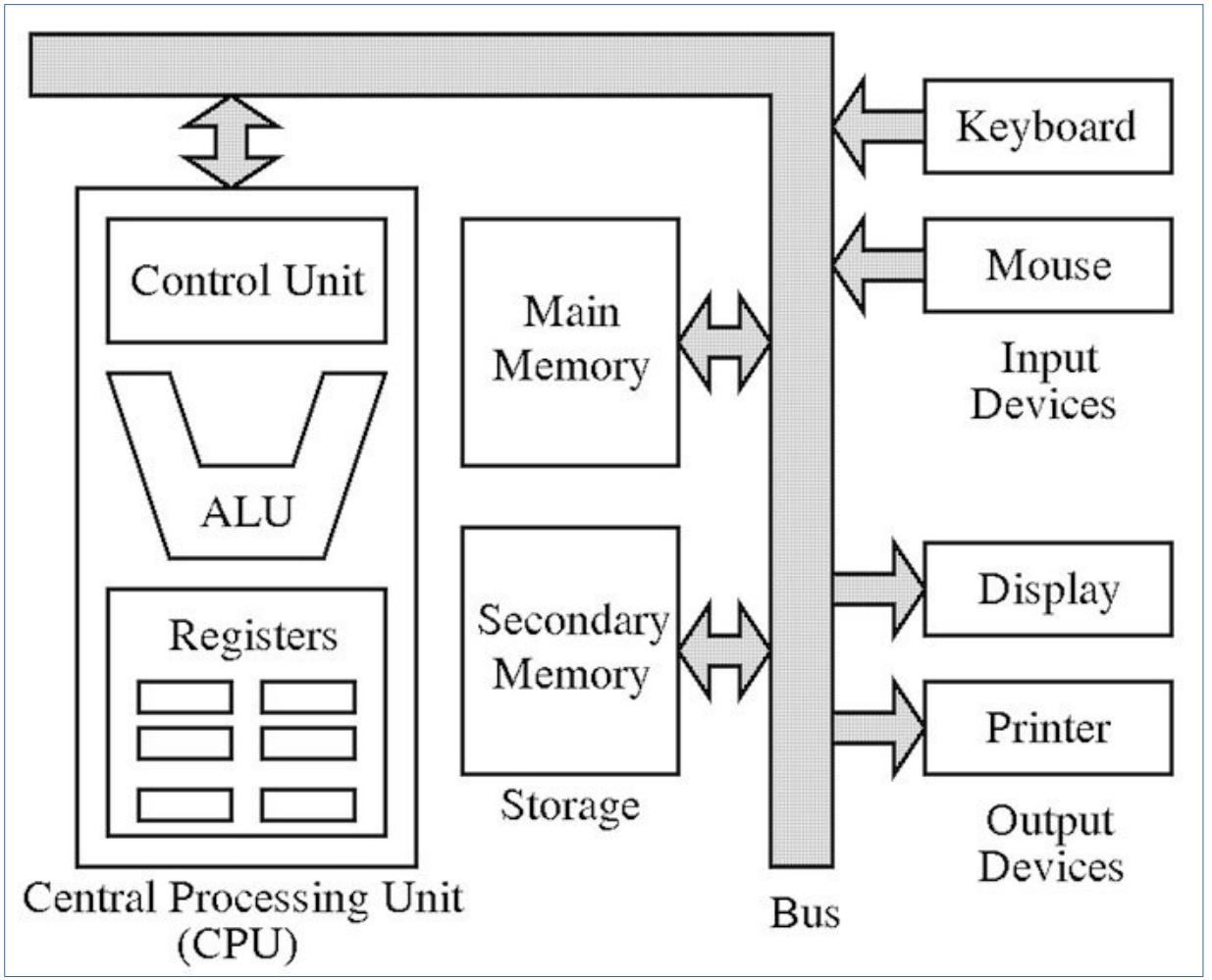
Ordinateur

Dispositif d'entrée

Dispositif de sortie

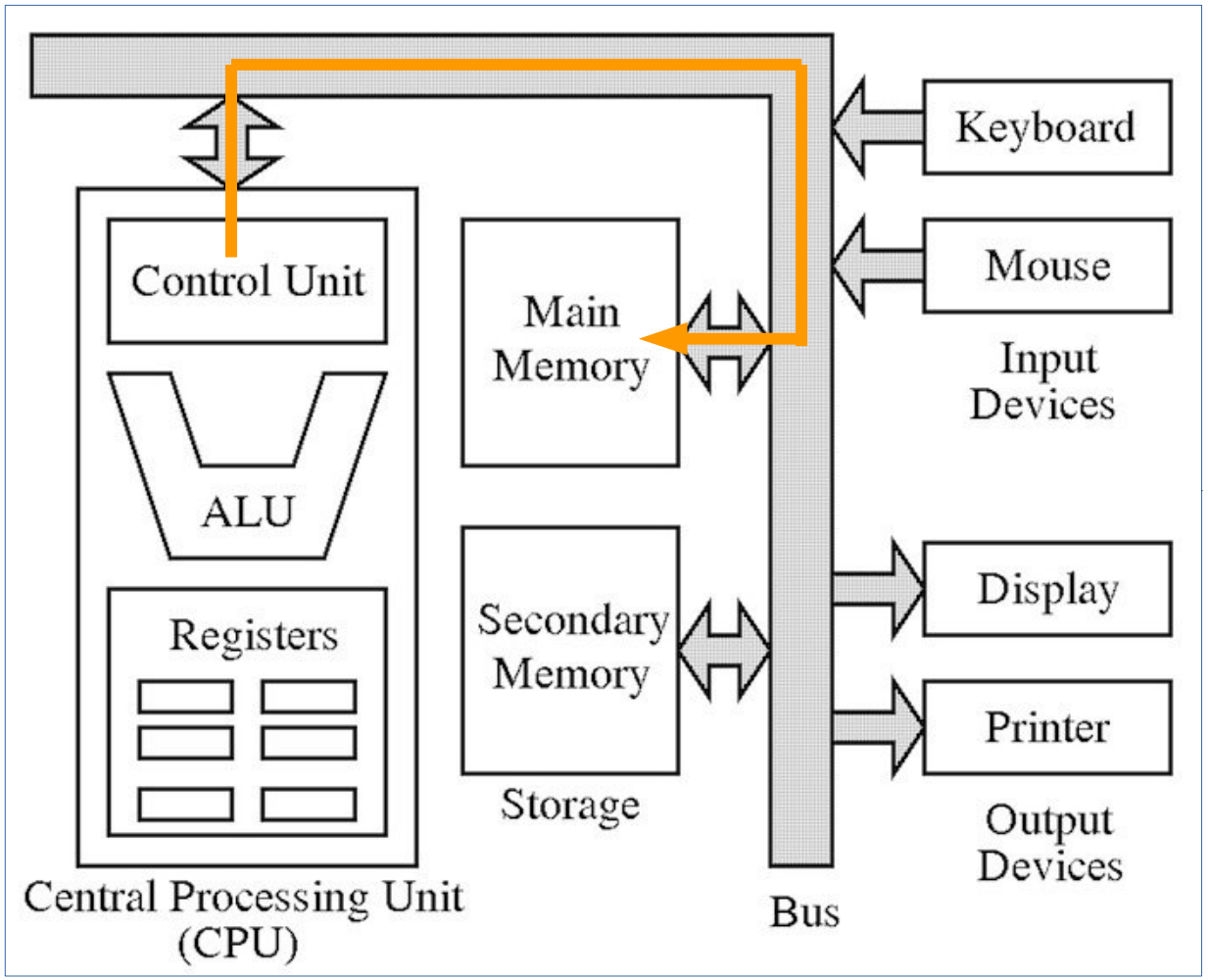
→ Permet la communication Avec l'exterieur

Architecture Von Neumann, en bref



Exemple d'utilisation : le *keylogger*

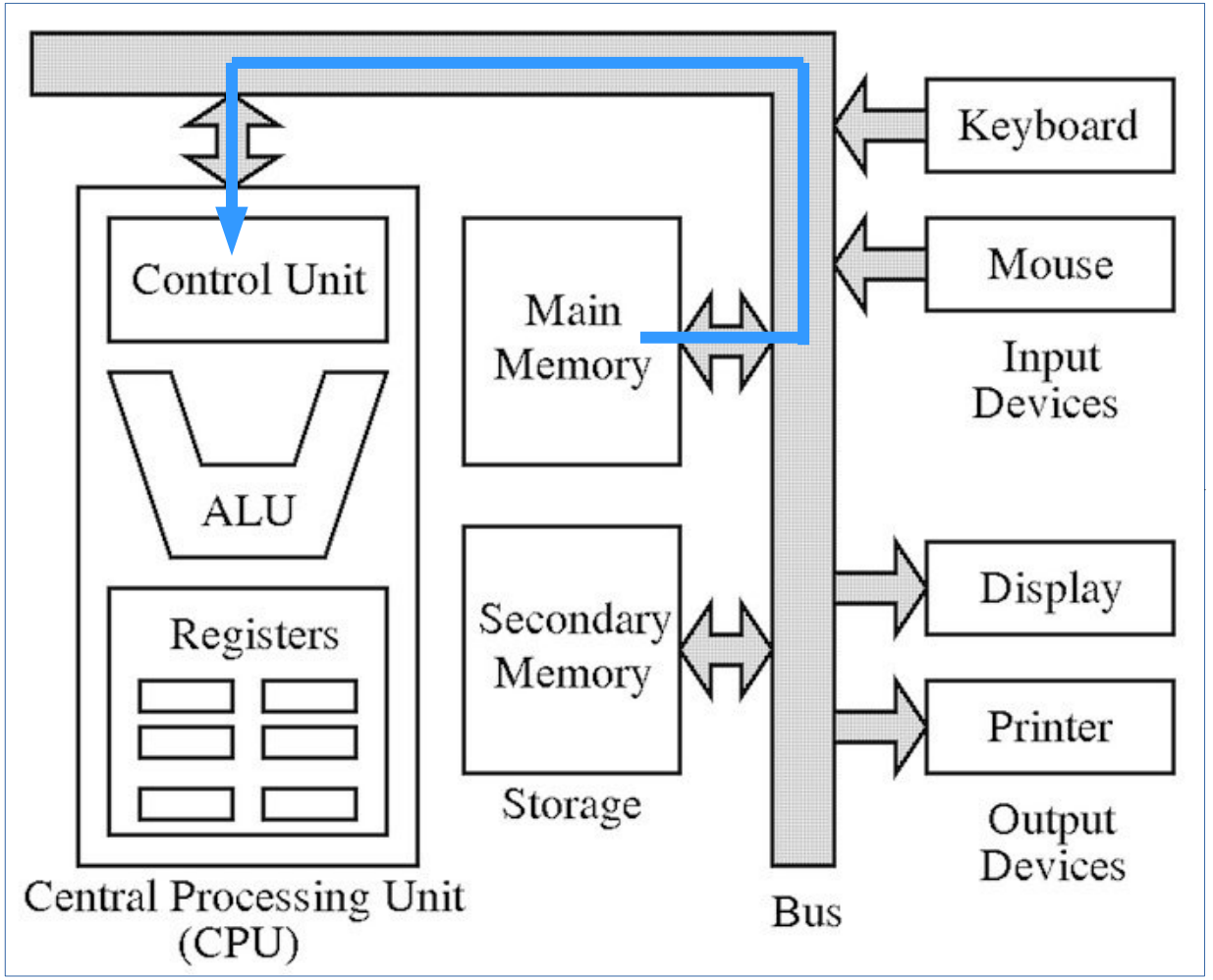
Demande de lecture de l'adresse du programme



- **Lecture du programme du *keylogger* en mémoire**
-
-
-

Exemple d'utilisation : le *keylogger*

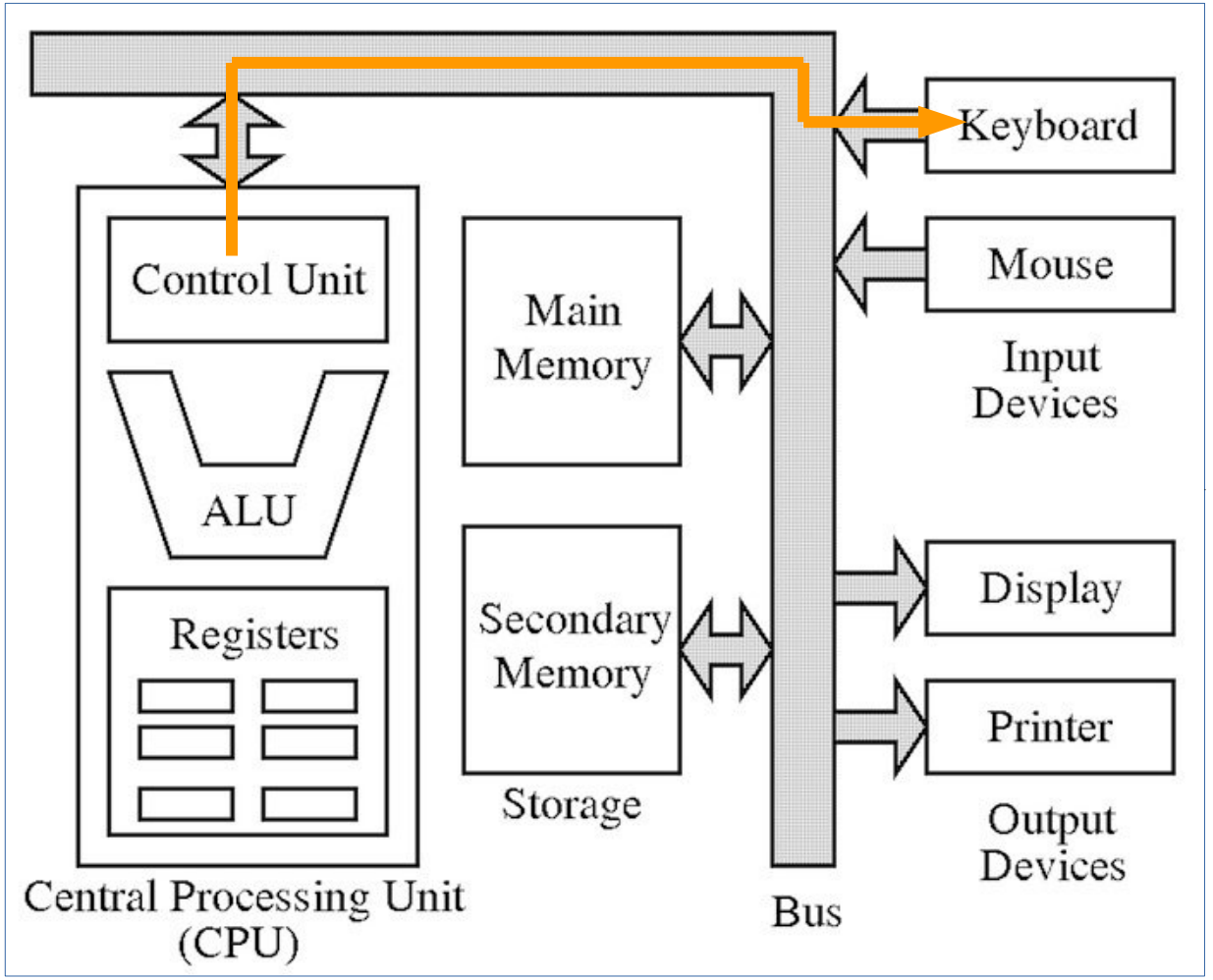
Récupération de l'opération à faire (ici lire la valeur clavier)



- **Lecture du programme du *keylogger* en mémoire**
-
-
-

Exemple d'utilisation : le *keylogger*

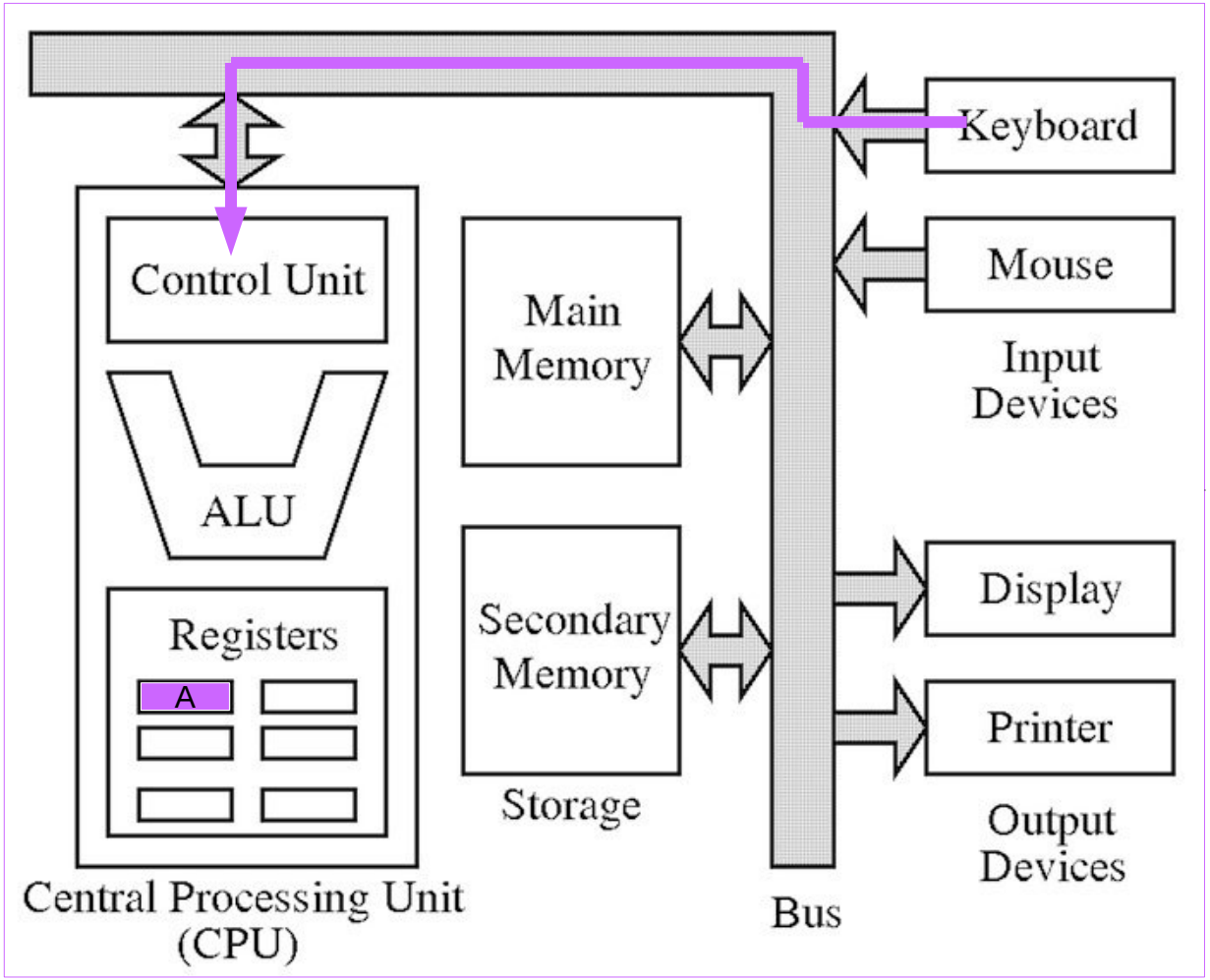
Demande de la valeur du clavier (exemple touche A)



- **Lecture du programme du *keylogger* en mémoire**
- **Récupération de la valeur du clavier**
-
-

Exemple d'utilisation : le *keylogger*

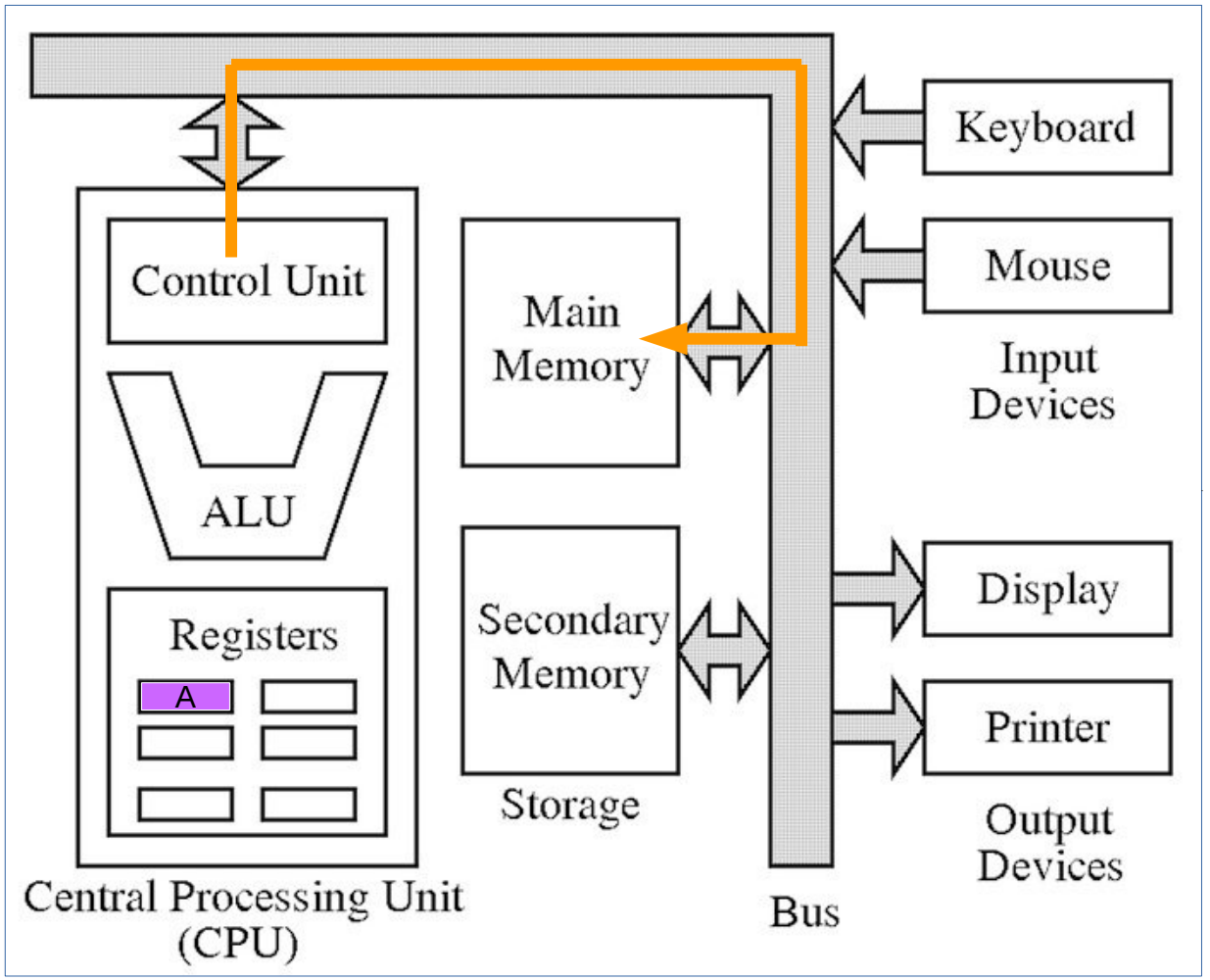
Réponse du périphérique (touche A appuyée)



- Lecture du programme du *keylogger* en mémoire
- Récupération de la valeur du clavier
-
-

Exemple d'utilisation : le *keylogger*

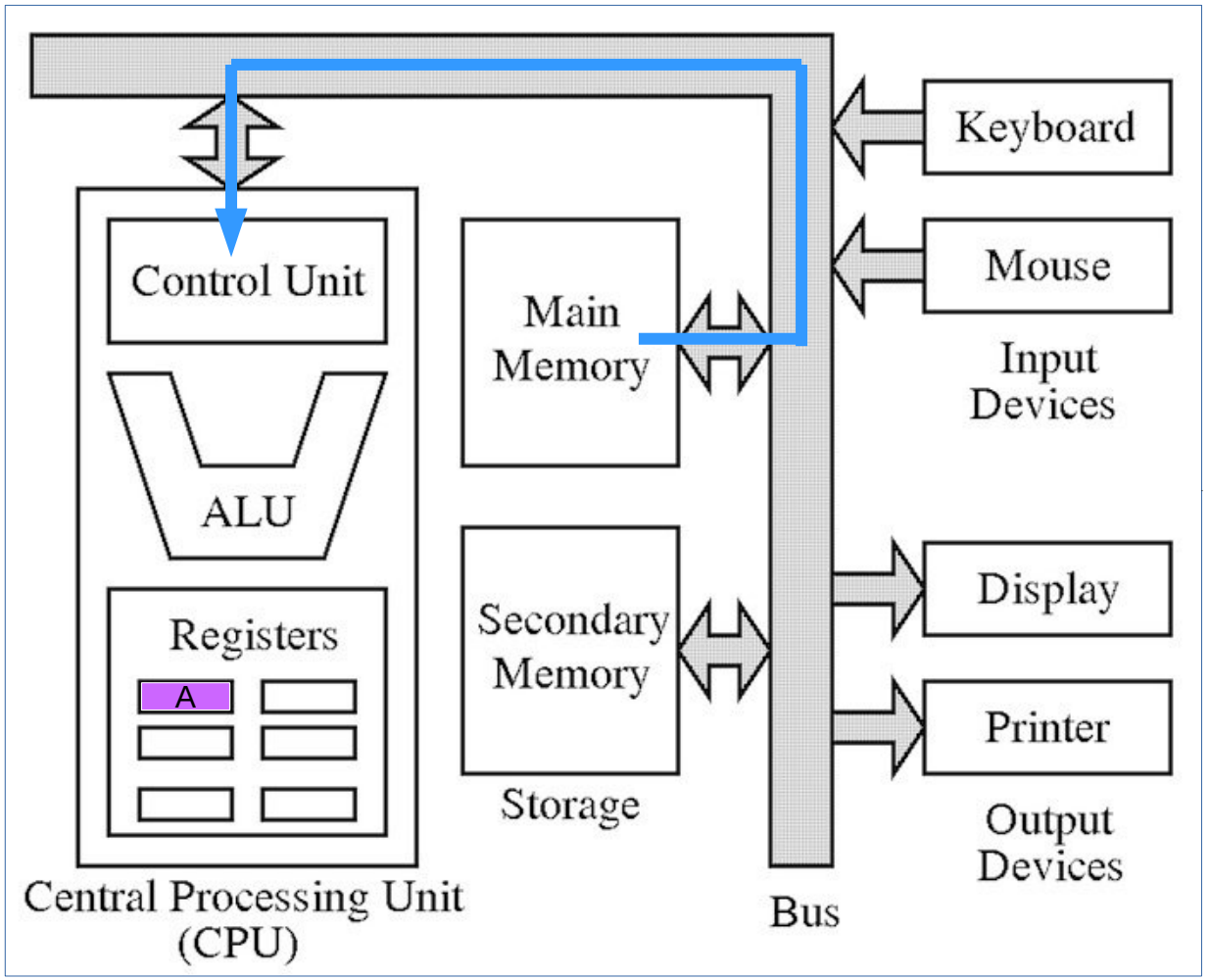
Demande de lecture de l'adresse du programme (+1)



- **Lecture du programme du *keylogger* en mémoire**
- **Récupération de la valeur du clavier**
- **Lecture du programme du *keylogger* en mémoire**
-

Exemple d'utilisation : le *keylogger*

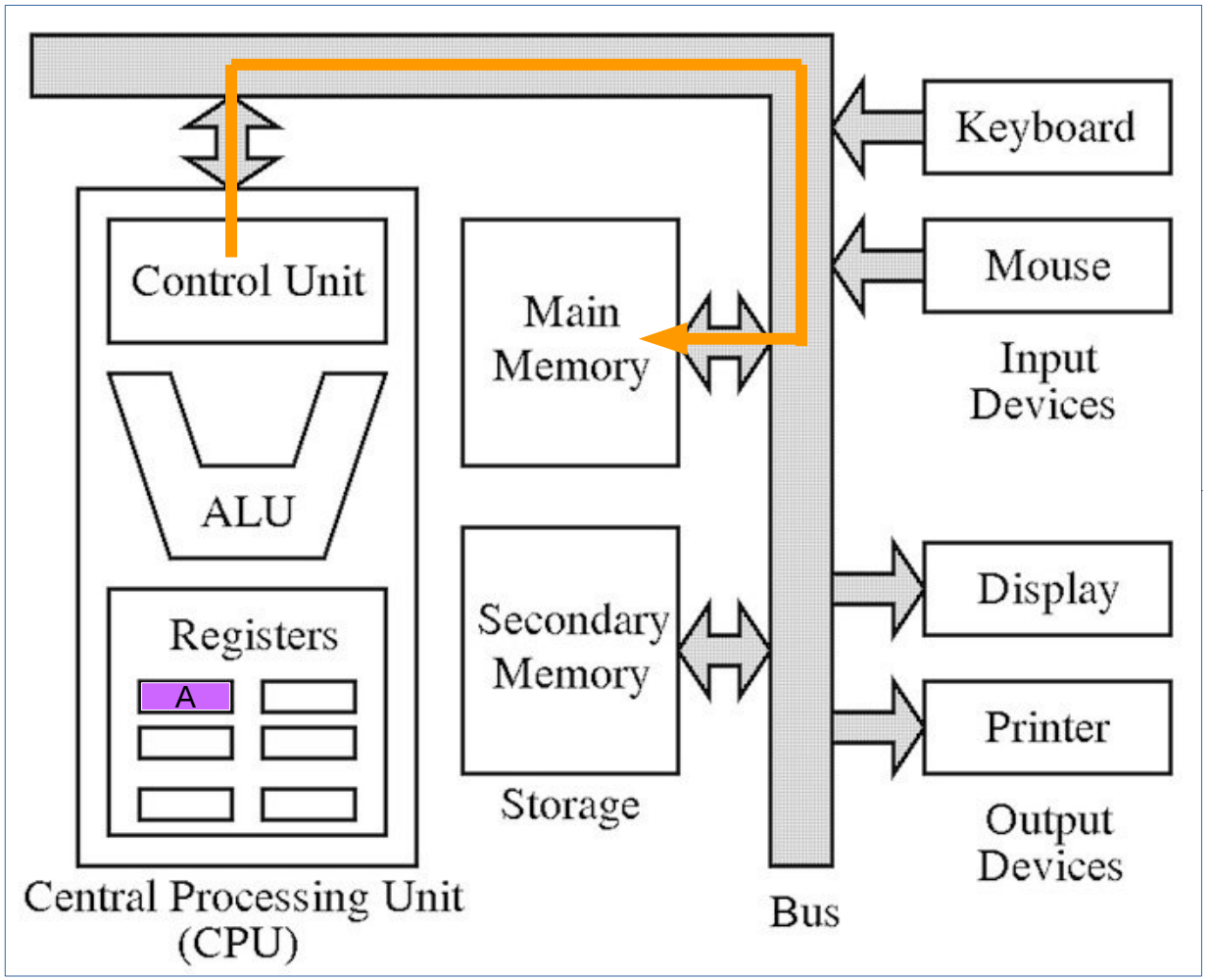
Récupération de l'opération à faire (ici stocker la valeur clavier)



- **Lecture du programme du *keylogger* en mémoire**
- **Récupération de la valeur du clavier**
- **Lecture du programme du *keylogger* en mémoire**
-

Exemple d'utilisation : le *keylogger*

Demande d'écriture d'une adresse en mémoire



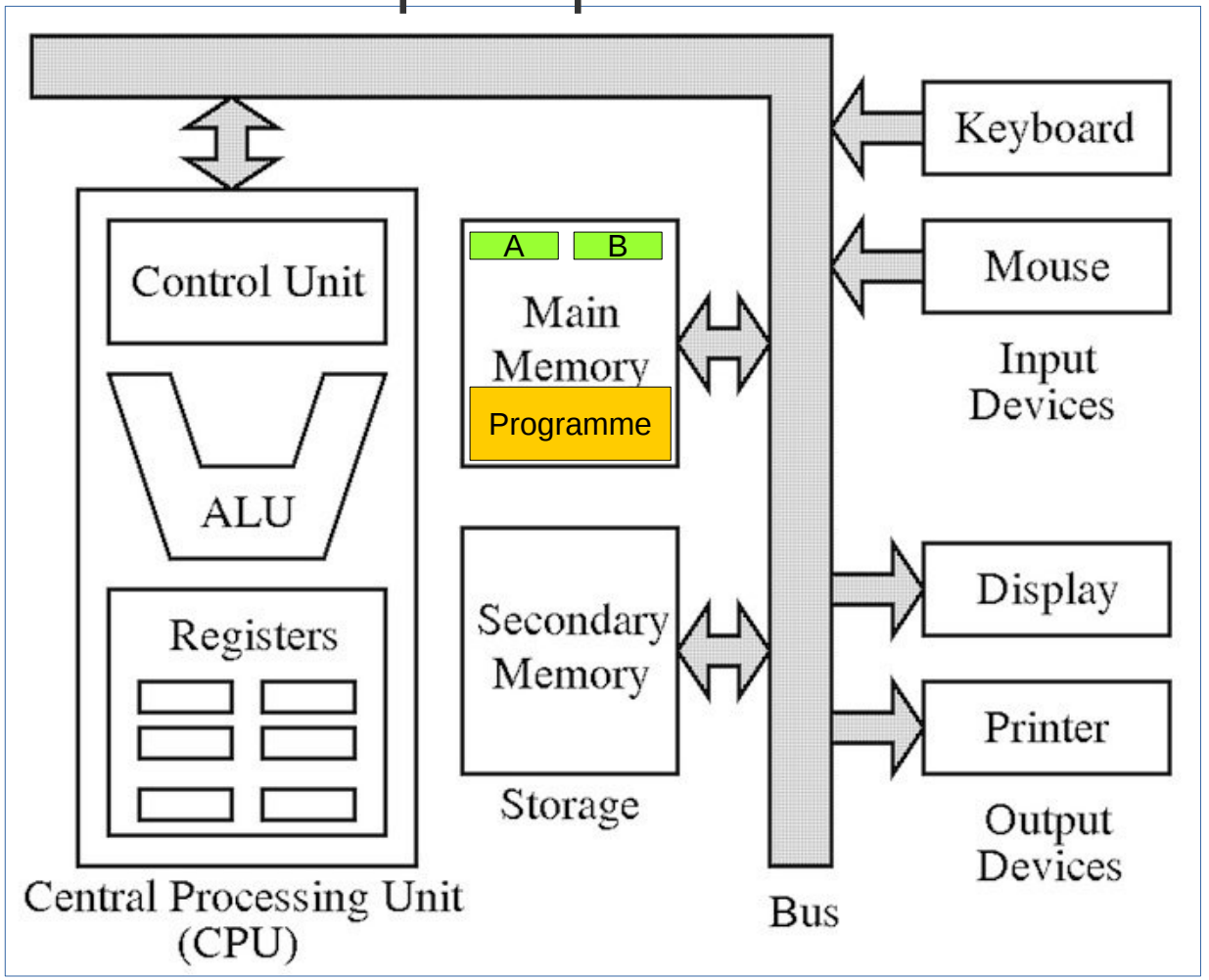
- **Lecture du programme du *keylogger* en mémoire**
- **Récupération de la valeur du clavier**
- **Lecture du programme du *keylogger* en mémoire**
- **Stockage de la valeur dans la mémoire**

Récapitulatif

- L’architecture Von Neumann décrit les composants essentiels d’un ordinateur.
- L’**unité de contrôle pilote le fonctionnement de toutes les autres unités**.
- Ce que doit faire l’**unité de contrôle** est décrit dans la **mémoire**. Cela s’appelle **une instruction**. Un ensemble d’instruction permet de réaliser un programme.
- L’**unité de contrôle** fait **un accès mémoire à chaque nouvelle instruction** !

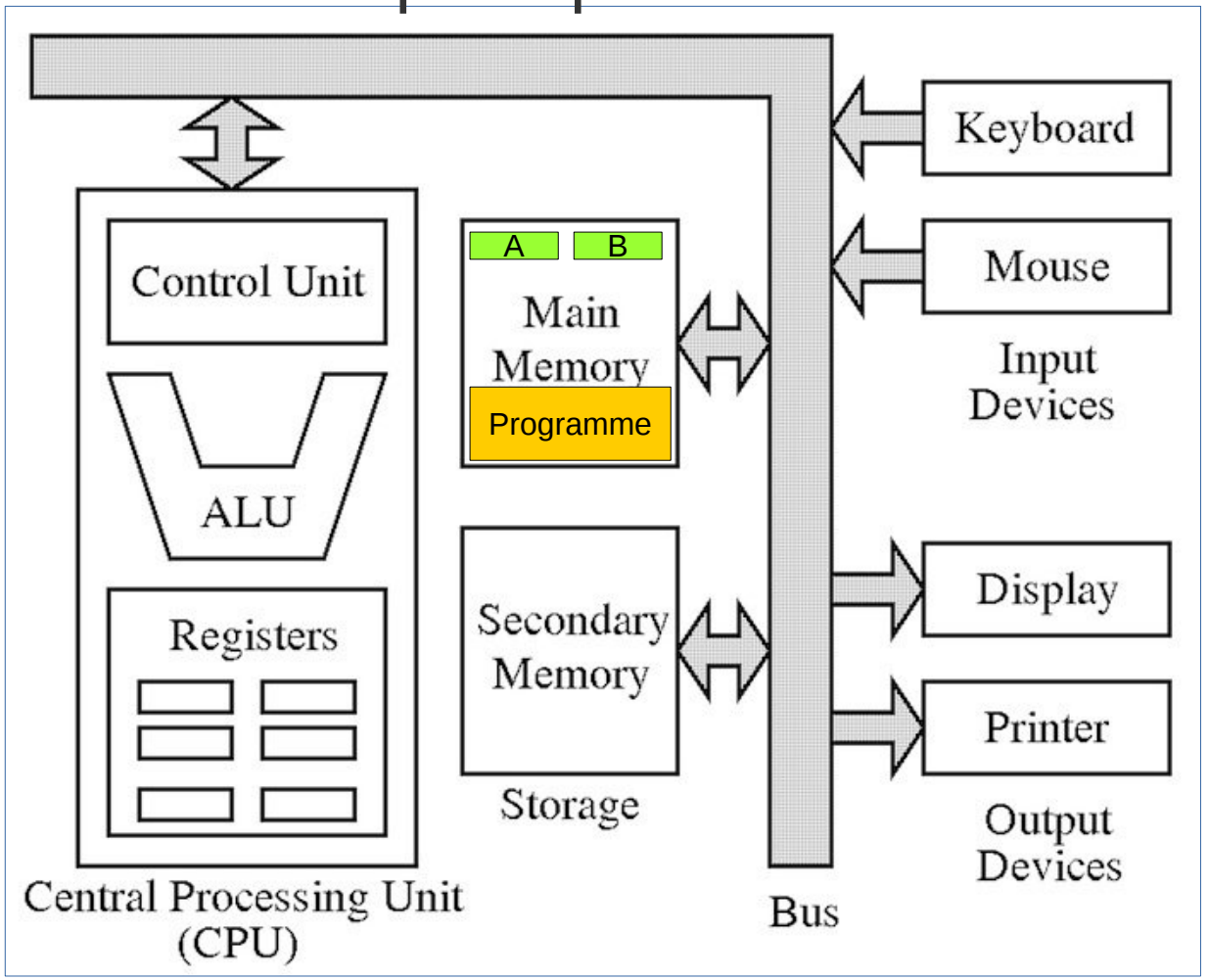
A vous de jouer !

- Programme : récupération de deux entiers A et B en mémoire, calcul de A+B = C, stockage du résultat dans la mémoire principale.



A vous de jouer !

- Programme : récupération de deux entiers A et B en mémoire, calcul de A+B = C, stockage du résultat dans la mémoire principale.



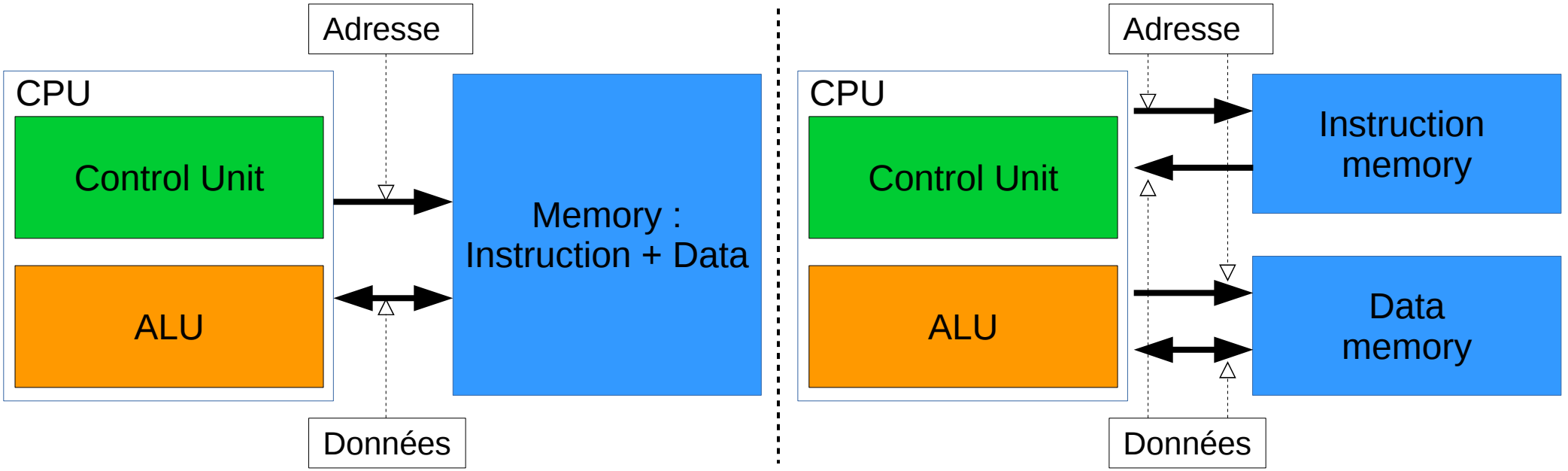
- Lecture de l'instruction du programme.
- Lecture de A et stockage temporaire du résultat dans un registre.
- Lecture de l'instruction suivante.
- Lecture de B et stockage temporaire du résultat dans un registre.
- Lecture de l'instruction suivante.
- Addition de A par B en utilisant l'ALU, stockage temporaire du résultat dans un registre (C).
- Lecture de l'instruction suivante.
- Ecriture du résultat dans la mémoire principale.

Limitation principale de l'architecture Von Neumann

- La principale **limitation** du modèle de Von Neumann est la présence sur **une unique mémoire pour le programme et les données**.
- Nous avons donc un séquençage
  → Lecture d'une instruction → Lecture de données
  → Lecture d'une instruction → Lecture de données
  → Lecture d'une instruction → ...
- On préfère donc utiliser en pratique d'autres architectures, à savoir l'architecture Harvard et Harvard modifiée, afin d'effectuer la lecture des instructions en parallèle du reste du fonctionnement.

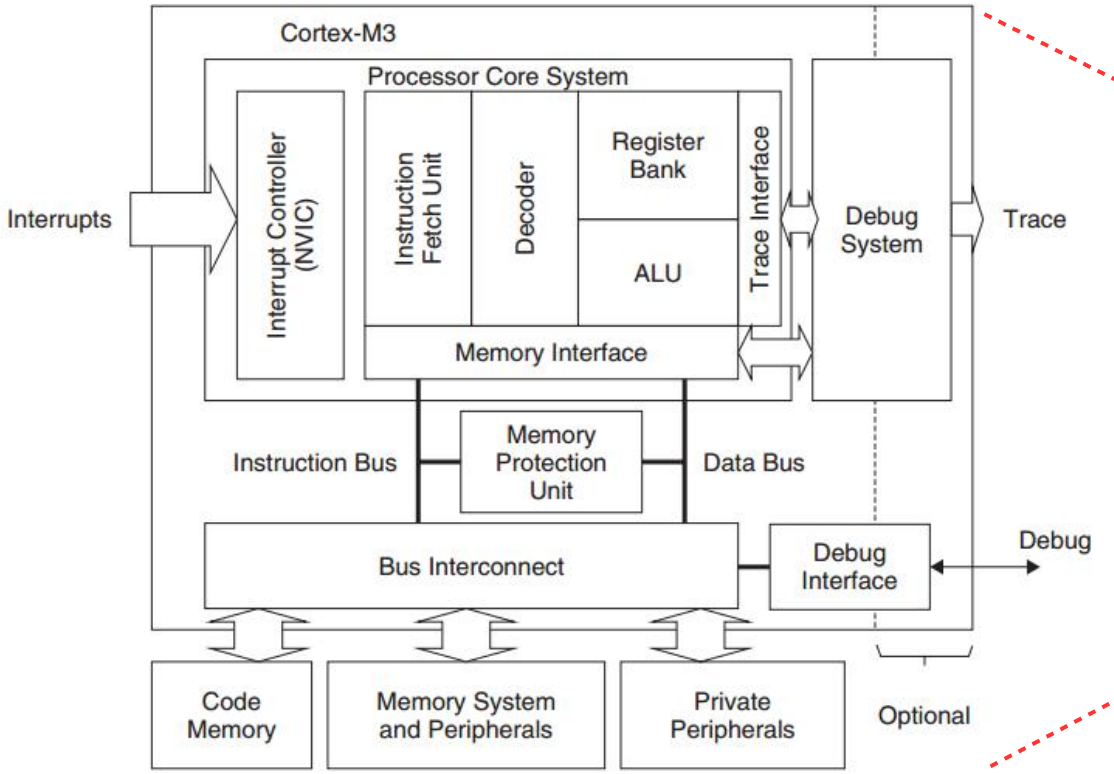
Architecture Harvard

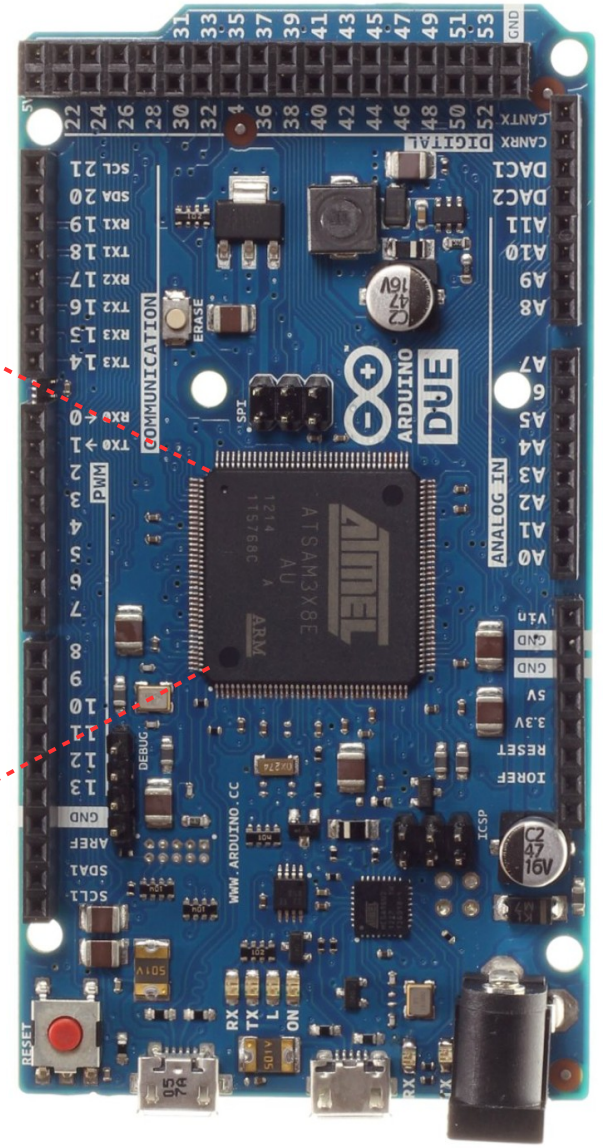
- En bref : Architecture Harvard = Architecture Von Neumann avec :
  - 1 mémoire pour les instructions ;
  - 1 mémoire pour les données.



Exemple d'architecture Harvard

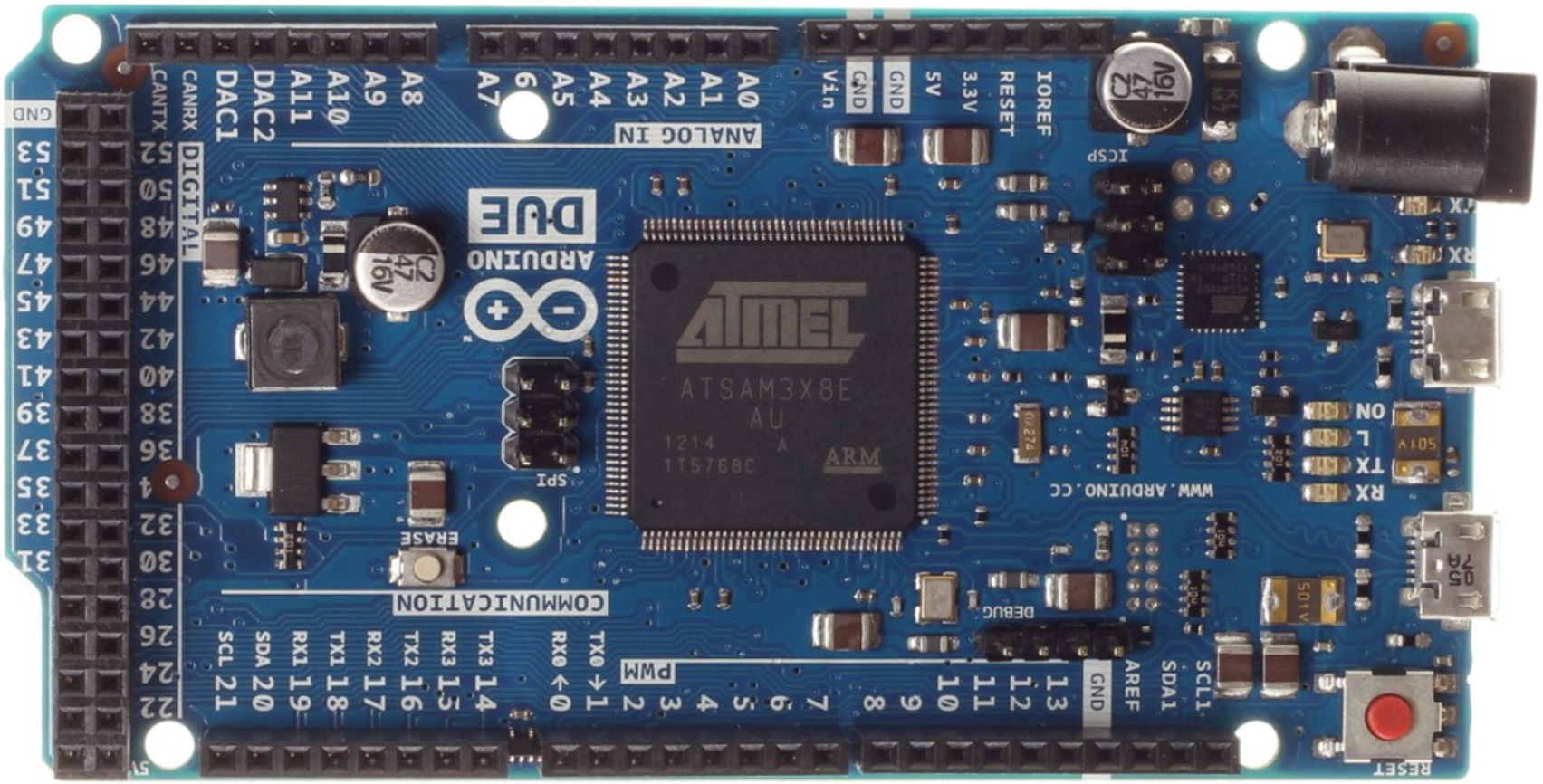
ARM Cortex-M3 → Arduino





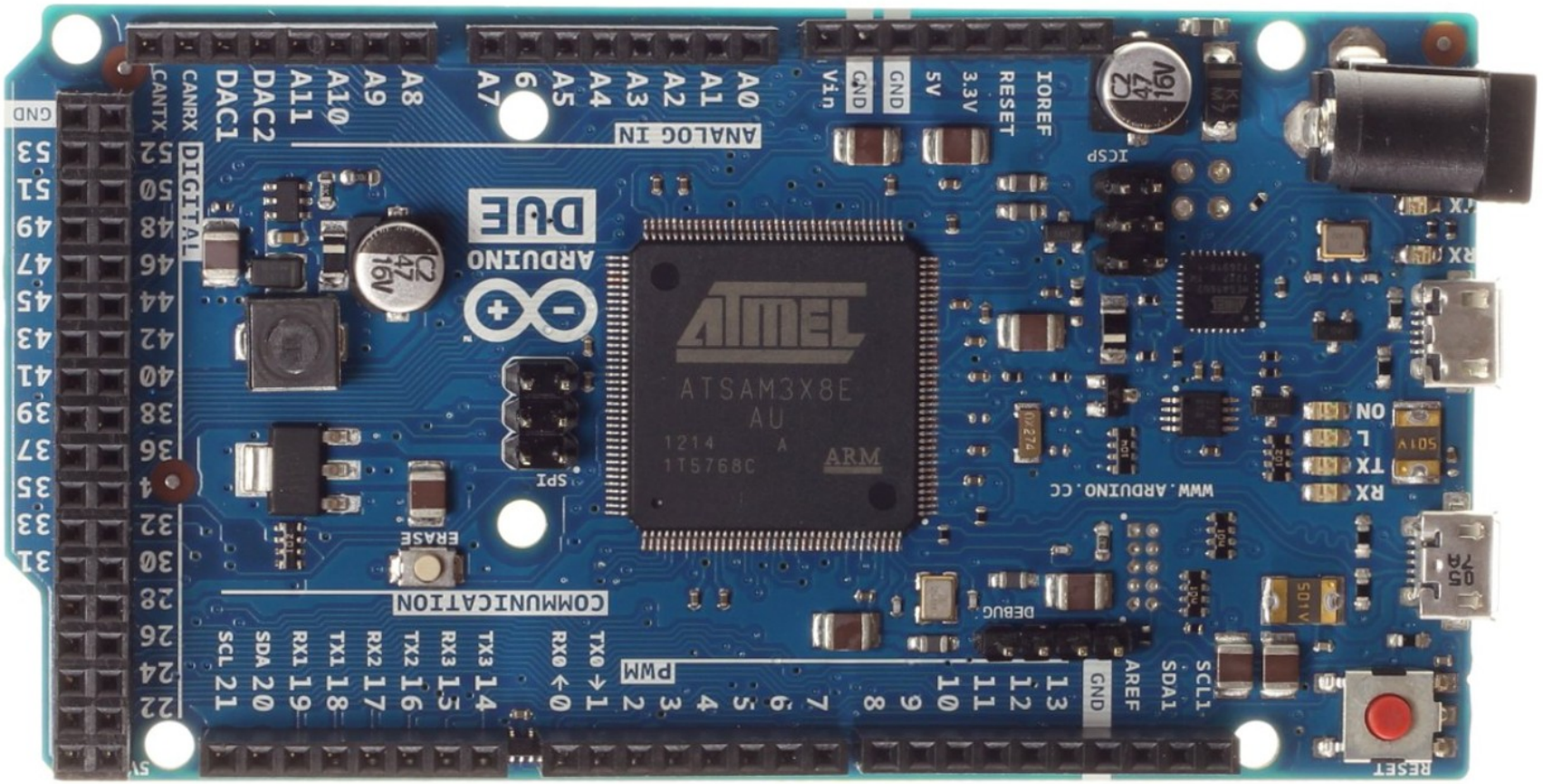
Si vous êtes observateur....

- A propos de ARM...



Si vous êtes observateur….

- A propos de ARM
  → ARM est écrit en petit devant ATMEL



Si vous êtes observateur….

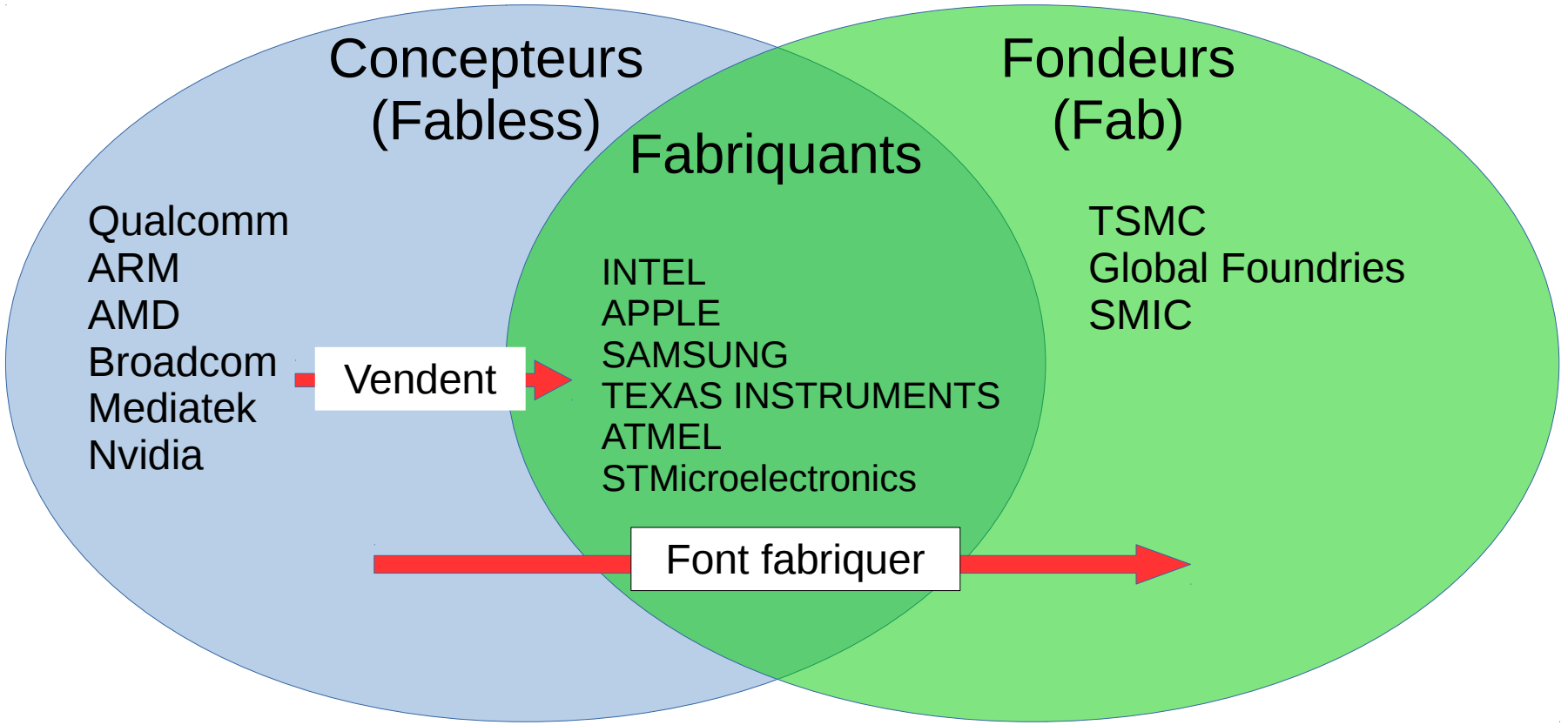
A propos de ARM

→ ARM est écrit en petit devant ATMEL

Pourquoi ? Réponse :

- ARM ne fabrique pas de processeur, il vendent les architectures.
- Les fabriquants de processeurs, récupèrent des architecture et fondent les composants.

Segmentation du marché des processeurs



Concepteurs (Fabless) : Ne font que la conception des circuits électroniques
Fondeurs (Fab) : Ne font que la fabrication des circuits électroniques
Fabriquants : Font les deux !

Limitation de l’architecture Harvard

Faible souplesse de fonctionnement :

- Découper physiquement programme et données fait qu’il n’est pas simple d’ajouter à la volée des programmes à l’ordinateur.

Cependant, cette architecture à tout à fait sa place dans des applications où la flexibilité n’est pas primordiale (systèmes embarqués).

- Solution pour la flexibilité :
  L’architecture Harvard modifiée.

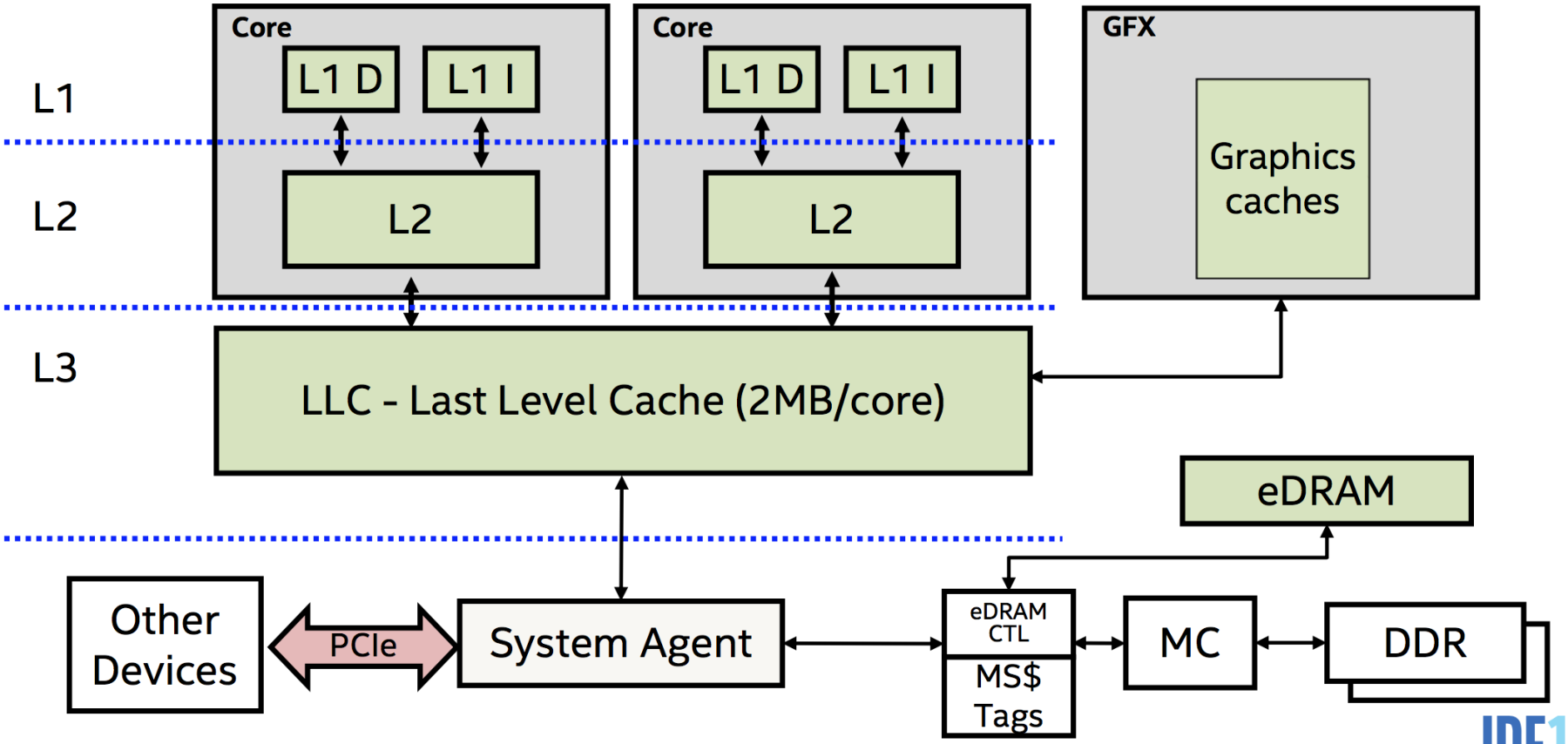
L’architecture Harvard modifiée

- Reprend **le principe de l’architecture Von Neumann** avec une mémoire principale contenant instructions + données.
- Deux mémoire rapides (appelées **caches**) sont implémentées dans le processeur **pour découper instructions et données**.

Exemple d'architecture Harvard modifiée

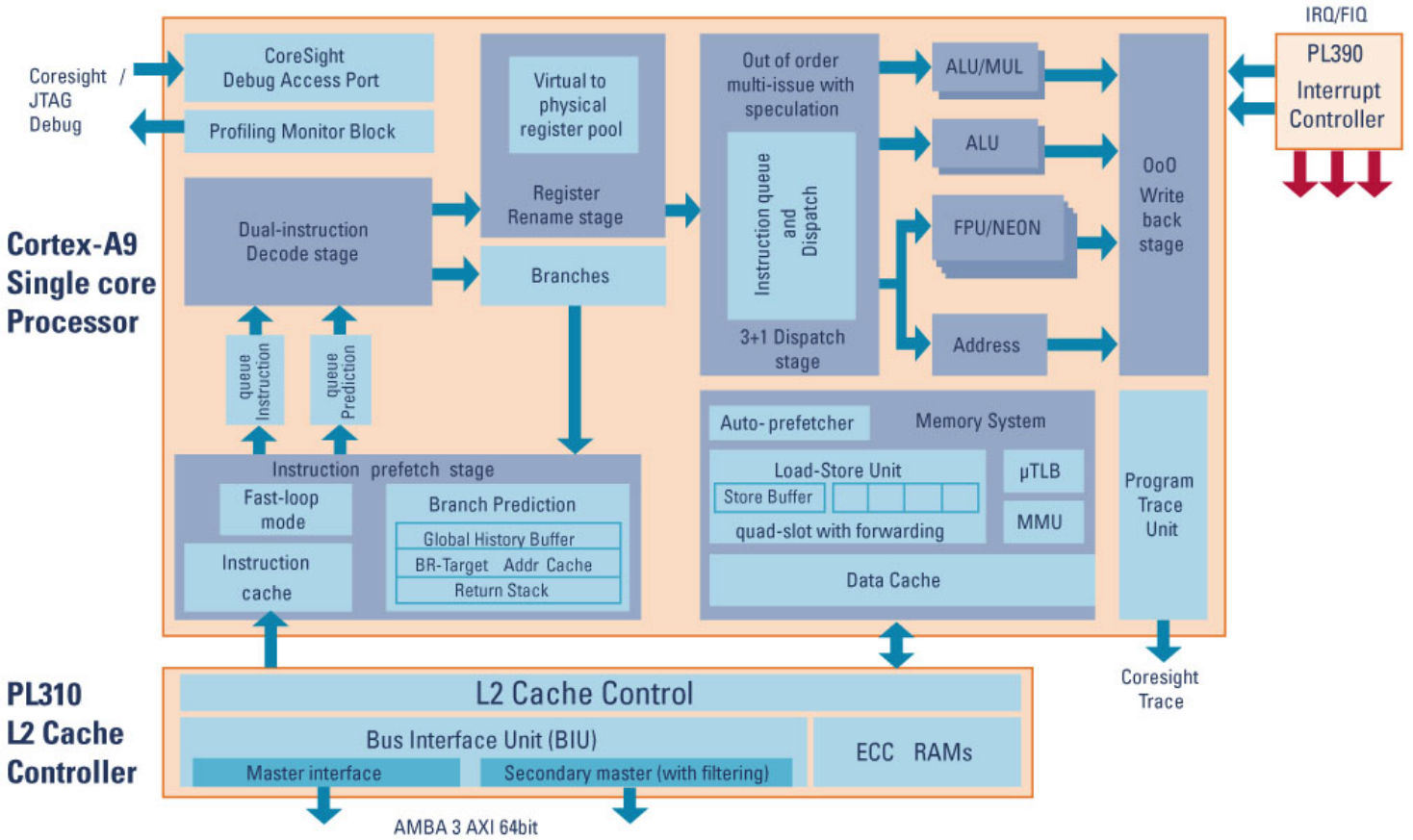
Intel Core – Skylake (6ème génération)

eDRAM Based Cache



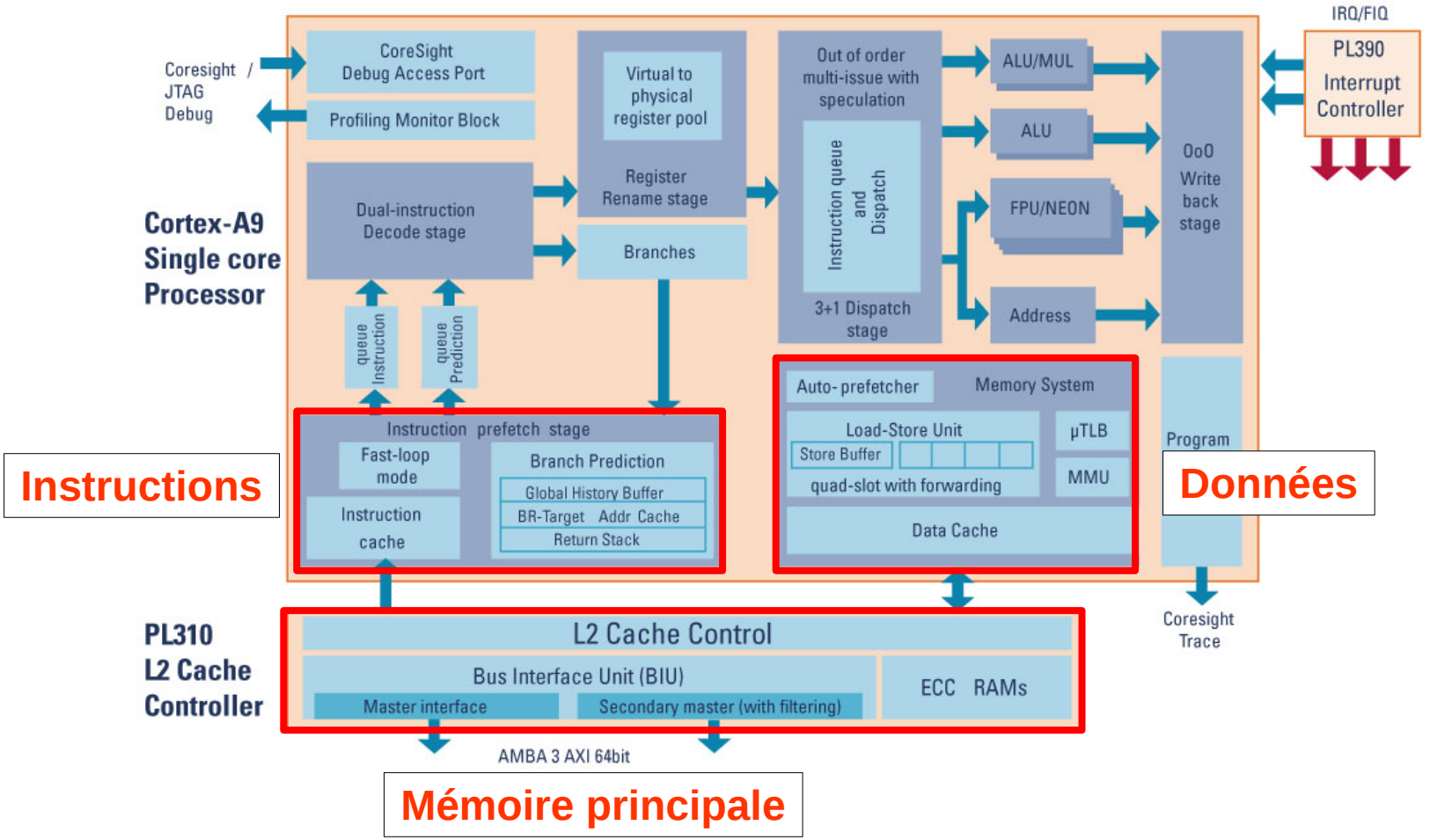
Exemple d'architecture Harvard modifiée

ARM Cortex-A9

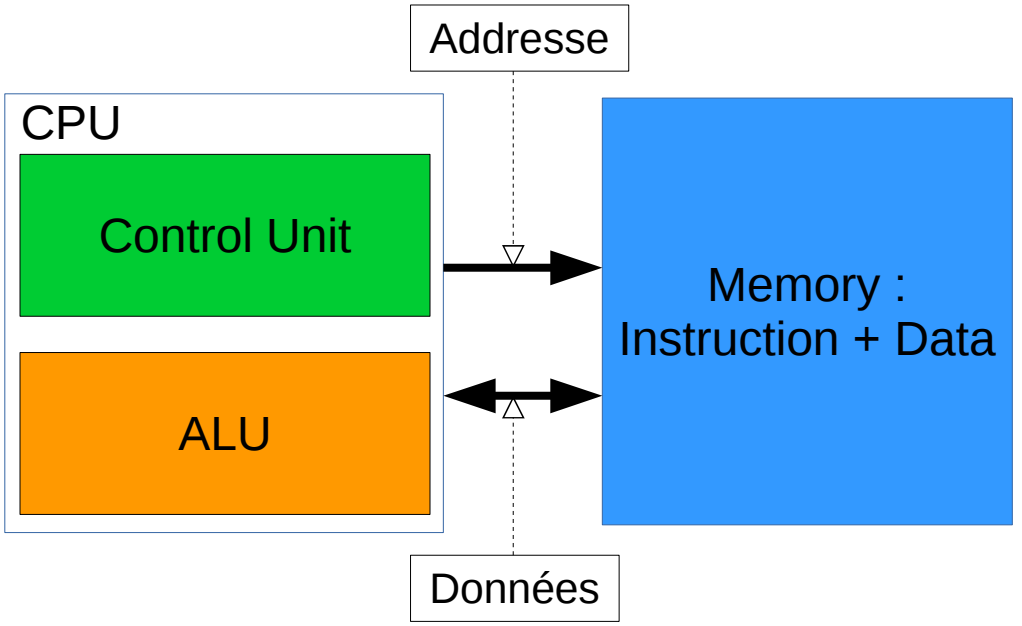


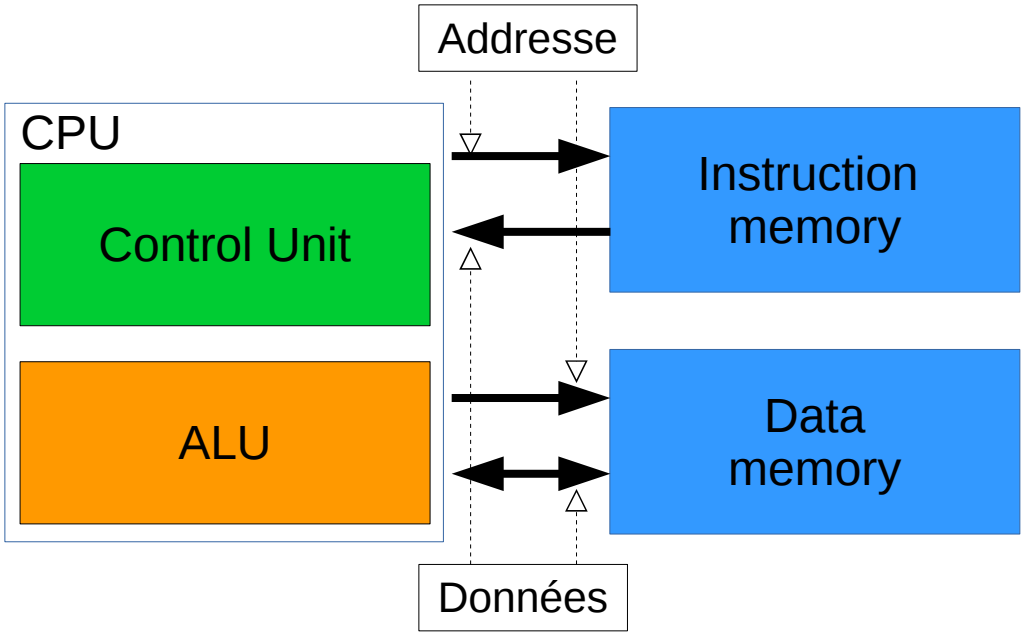
Exemple d'architecture Harvard modifiée

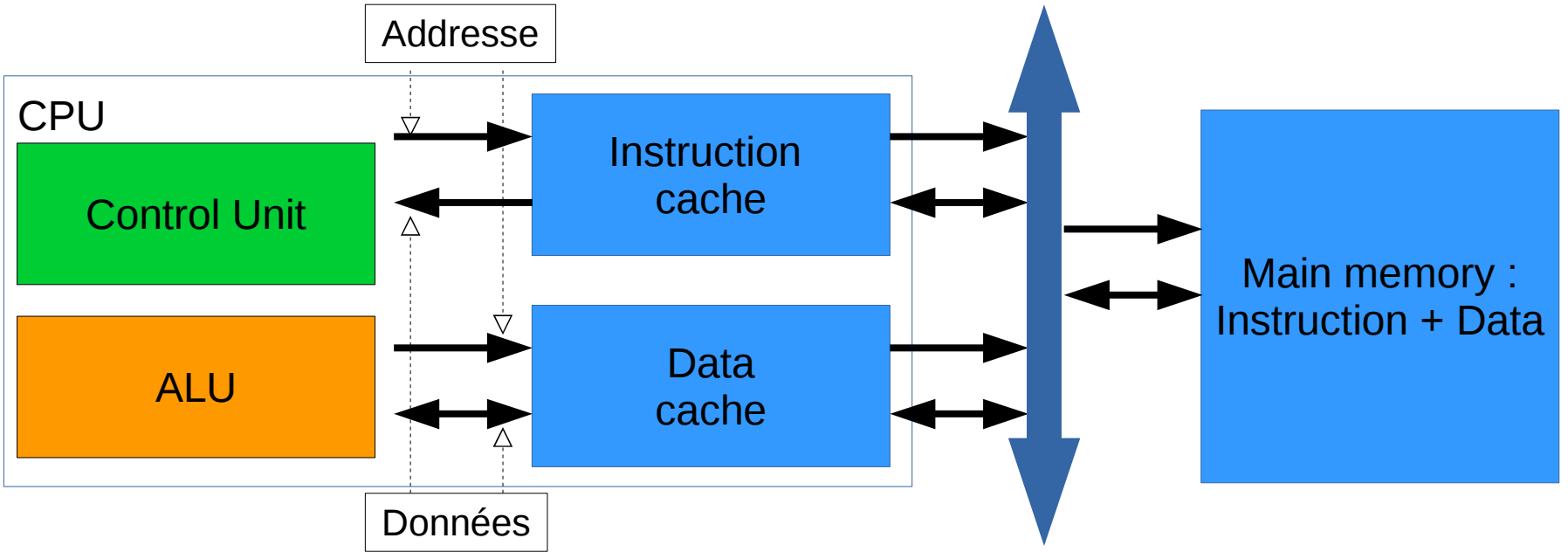
ARM Cortex-A9



Récapitulatif





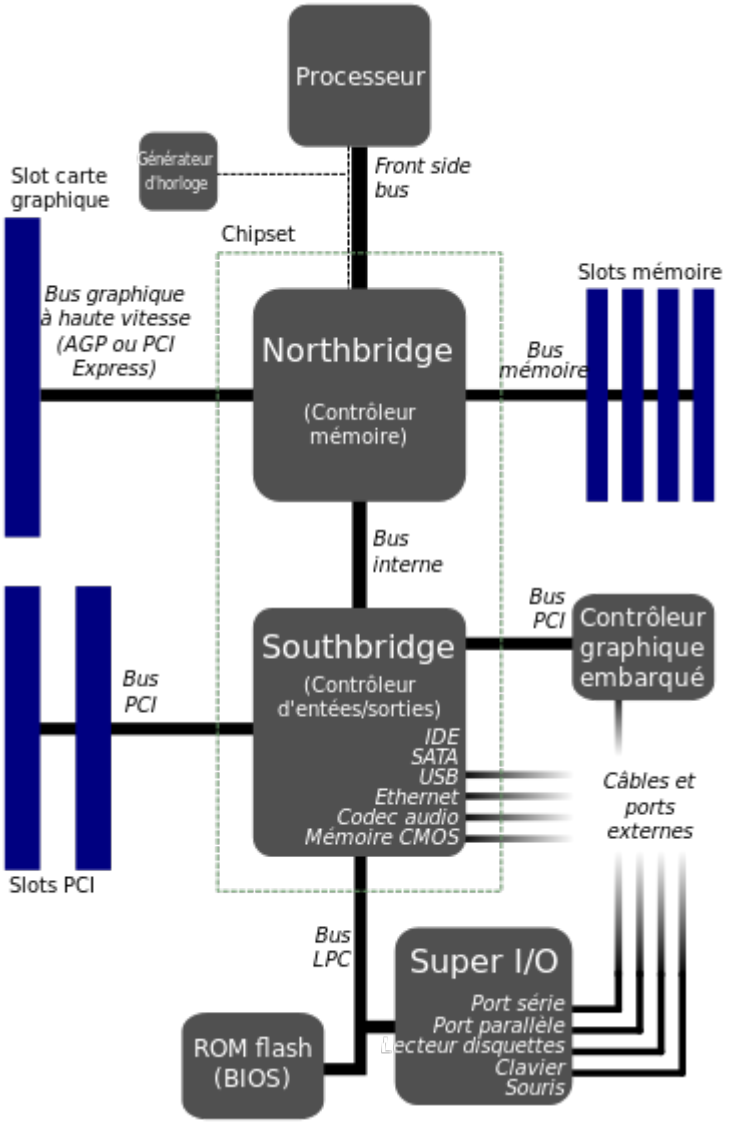


Architecture dans un ordinateur personnel

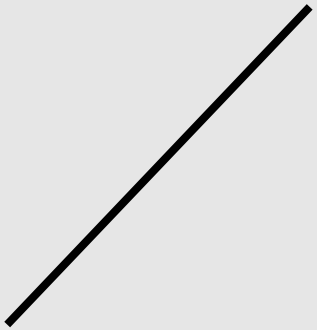
- Deux « ponts », le north bridge et le south bridge.
- **North bridge :**

  Composants fortement sollicités par le processeur (mémoire vive, Carte graphique).
- **South Bridge :**

  Les autres composants, ainsi que le BIOS.



Conclusion

	Von Neumann	Harvard	Harvard modifiée
Découpage Instructions / données	Instructions et données sur la même mémoire	Deux mémoires distinctes pour les instructions et les données	Les instructions et les données sur la même mémoire, mais au niveau du processeur, les instructions et les données sont découpées au niveau de mémoires rapides appelées caches.
Avantages comparatifs		On peut traiter en parallèle instructions et données. En pratique, les fabricants utilisent ce découpage pour implémenter deux technologies de mémoire différentes.	Flexibilité d'utilisation, les données pouvant être traitées comme des instructions.

Quelques informations complémentaires sur ARM

Trois gammes :

- **Cortex-M**, gamme micro-controlleurs : **(Architecture Harvard)**

  circuits intégrés possédant l'**essentiel** pour le fonctionnement d'un ordinateur. Bas coût, faible consommation, idéal pour des **systèmes embarqués**. (d'où son utilisation sur Arduino…)

- **Cortex-A**, gamme micro-processeur : **(architecture Harvard modifiée)**

  circuits intégrés possédant des **unités avancées** pour une utilisation sur des ordinateurs personnels (Principalement dans les smartphones pour également leur propriété de faible consommation énergétique).

- **Cortex-R**, gamme temps réel :

  gamme pour les **systèmes critiques**, à savoir où l'on doit s'assurer qu'une tâche sera terminée avant un moment fixé.

  Exemple : avionique, automobile,…

Quelques informations complémentaires sur Intel

(Plein) de gammes :

- **Intel Core i3, i5, i7, i9**, gamme grand public : **(Architecture Harvard modifiée)**

  processeurs possédant typiquement **entre 2 et 8 cœurs en parallèle**. Des Processeurs graphiques sont également implémentés pour une utilisation dans des ordinateurs grands public. Attention : Intel garde le même nom (i3,i5,..) pour chacune des générations de processeurs (actuelle la 8eme).

- **Intel Xeon**, gamme serveurs de calcul : **(architecture Harvard modifiée)**

  processeurs possédant un grand nombre de cœurs en parallèle, typiquement **entre 10 et 24 cœurs**.

- Intel Atom (mobile), Intel Celeron (bon marché)