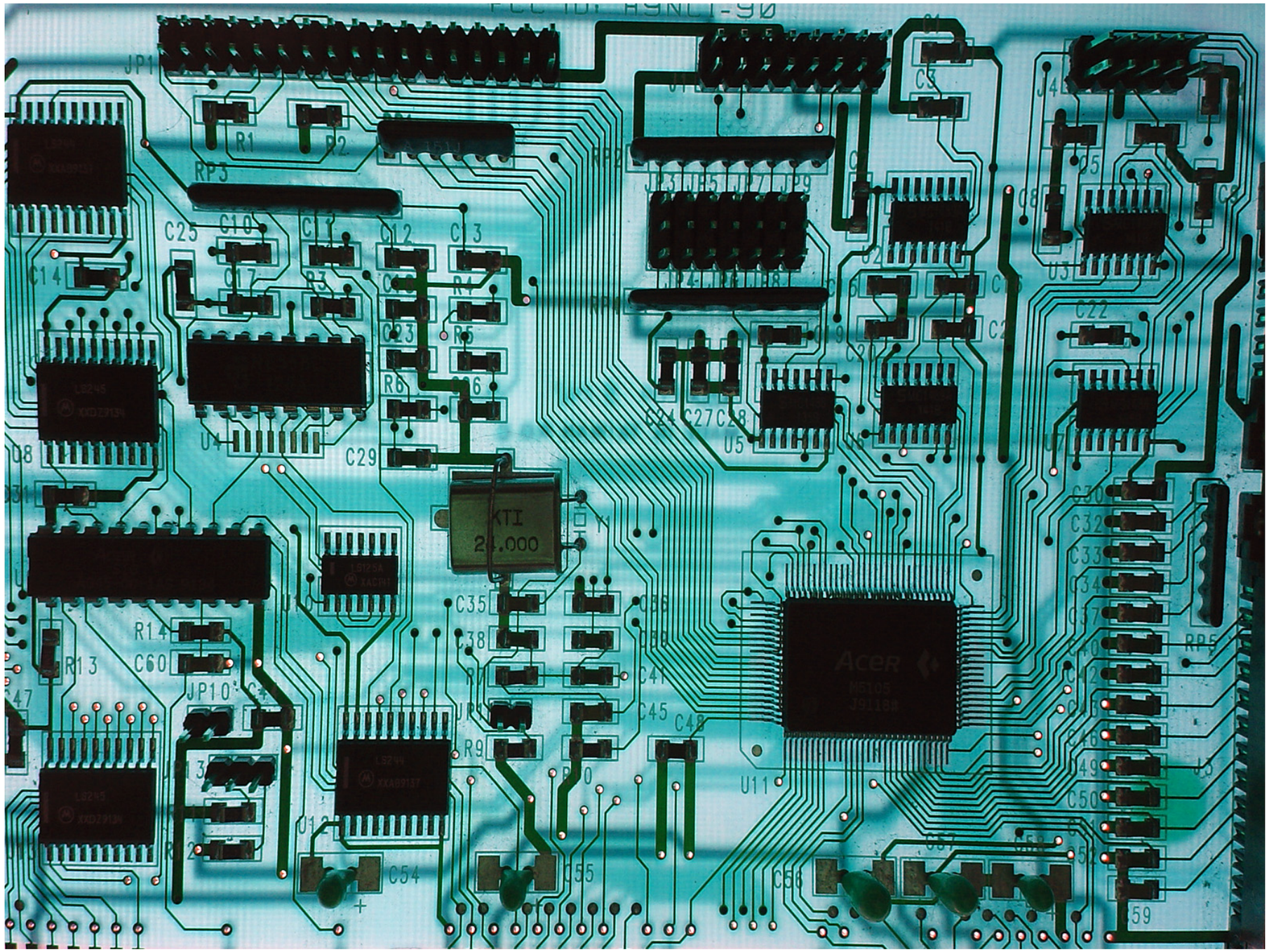
Mémoire et bus



GIF-1001: Ordinateurs: Structure et Applications

Jean-François Lalonde

Mémoire

- Un emplacement qui contient des données.
- Organisé par cellules individuelles.
- Chaque cellule possède:
  - (la même) taille: en nombre de bits.
  - une adresse unique: permet de savoir à quel endroit on veut stocker l’information.

Analogie #1 (mémoire): boîte aux lettres

- chaque boîte à lettres contient:
  - un nombre fixe de lettres (contenu)
  - un numéro (adresse)

Analogie #2 (mémoire): dossiers

- chaque dossier contient:
  - un nombre fixe de feuilles (contenu)
  - un numéro pour pouvoir le retrouver (adresse)

Mémoire = stockage de bits

Chaque case contient un bit (0 ou 1)

Une case ne peut jamais être «vide»!
Elle contient *toujours* 0 ou 1.

On regroupe les bits ensemble pour former un «mot».
Ici, les mots ont 8 bits (1 octet)

Pour identifier les mots, on leur associe une adresse, qui correspond au numéro de ligne dans le tableau.
Par exemple, ce mot aurait l'adresse 4
(on commence à 0).

0	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
0	1	0	0	1	0	0	1
0	1	0	0	1	1	0	1
0	1	1	1	1	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	0	1	0	0	0	1

Mémoir de bits

Ici, l'exemple utilise 8 bits d'adresse, et des mots de 8 bits. Cependant, différents systèmes peuvent avoir différentes valeurs!

On utilise un nombre (fini et prédéterminé) de bits pour représenter l'adresse.

Adresse (sur 8 bits)	Donnée (mot de 8 bits)							
	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
0x00	0	1	0	1	1	0	1	1
0x01	1	1	1	0	1	0	0	1
0x02	0	1	0	0	1	0	0	1
0x03	0	1	0	0	1	1	0	1
0x04	0	1	1	1	1	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0xFF	1	1	0	1	0	0	0	1

Mémoire

- On décrit une mémoire grâce à deux informations (indépendantes):
  - le **nombre d'adresses** possibles
    - ici: $2^8 = 256$ adresses
  - la **taille des mots** de la mémoire
    - ici: 8 bits (1 octet)

Adresses	Données							
	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
0x00	0	1	0	1	1	0	1	1
0x01	1	1	1	0	1	0	0	1
0x02	0	1	0	0	1	0	0	1
0x03	0	1	0	0	1	1	0	1
0x04	0	1	1	1	1	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0xFF	1	1	0	1	0	0	0	1

- La taille totale de la mémoire représente la quantité totale de bits pouvant être stockée dans la mémoire.
  - Elle est calculée de cette façon:

$$\text{taille mémoire} = \text{nombre d'adresses} \times \text{taille d'un mot}$$
$$= 2^8 \times 1 \text{ octet} = 256 \text{ octets}$$

Question 1

Rappel
taille mémoire = nombre d'adresses × taille d'un mot
1 kilo-octet = 2^{10} octets

Une mémoire stocke des mots de 8 bits (1 octet)
et possède 2^{16} adresses.
Quelle est la taille totale de la mémoire en kilo-octets (Ko)?

Question 1

Rappel
taille mémoire = nombre d'adresses × taille d'un mot
1 kilo-octet = 2^{10} octets

Une mémoire stocke des mots de 8 bits (1 octet)
et possède 2^{16} adresses.
Quelle est la taille totale de la mémoire en kilo-octets (Ko)?

Taille d'un mot: 1 octet
Nombre total de mots: 2^{16}
Taille mémoire = 1 x 2^{16} = 2^{16} octets = 2^{6} Ko = 64 Ko.

Question 2

Rappel
taille mémoire = nombre d'adresses × taille d'un mot
1 kilo-octet = 2^{10} octets

Une mémoire stocke des mots de 16 bits (2 octets)
et nécessite 8 bits pour les adresser.
Quelle est la taille totale de la mémoire en octets?

Question 2

Rappel
taille mémoire = nombre d'adresses × taille d'un mot
1 kilo-octet = 2^{10} octets

Une mémoire stocke des mots de 16 bits (2 octets)
et nécessite 8 bits pour les adresser.
Quelle est la taille totale de la mémoire en octets?

Taille d'un mot: 2 octets
Bits pour les adresses: 8 bits
Nombre d'adresses: $2^8 = 256$

Taille mémoire = 2 x 2^8 = 2^9 octets = 512 o.

Question 3

Rappel
taille mémoire = nombre d'adresses × taille d'un mot
1 kilo-octet = 2^{10} octets

Une mémoire possède une taille totale de 32 Mo
et peut stocker des mots de 32 bits.

Combien de bits a-t-on besoin pour représenter les adresses dans cette mémoire?
Quelles sont les adresses minimales et maximales de cette mémoire exprimées en hexadécimal?

Question 3

Rappel
taille mémoire = nombre d'adresses × taille d'un mot
1 kilo-octet = 2^{10} octets

Une mémoire possède une taille totale de 32 Mo
et peut stocker des mots de 32 bits.

Combien de bits a-t-on besoin pour représenter les adresses dans cette mémoire?
Quelles sont les adresses minimales et maximales de cette mémoire exprimées en hexadécimal?

Taille mémoire: 32Mo
Taille d'un mot: 32 bits = 4o (octets)
Nombre total de mots = 32 Mo / 4o = 32 x 2^{20} / 4 = 2^{25} / 2^{2} = 2^{23}.
Donc, nous avons besoin de $\log_2(2^{23})$ = 23 bits pour représenter les adresses.

Les adresses minimales et maximales sont:
minimale: 0b00000000000000000000000 (23 bits) = 0x000000
minimale: 0b11111111111111111111111 (23 bits) = 0x7FFFFF

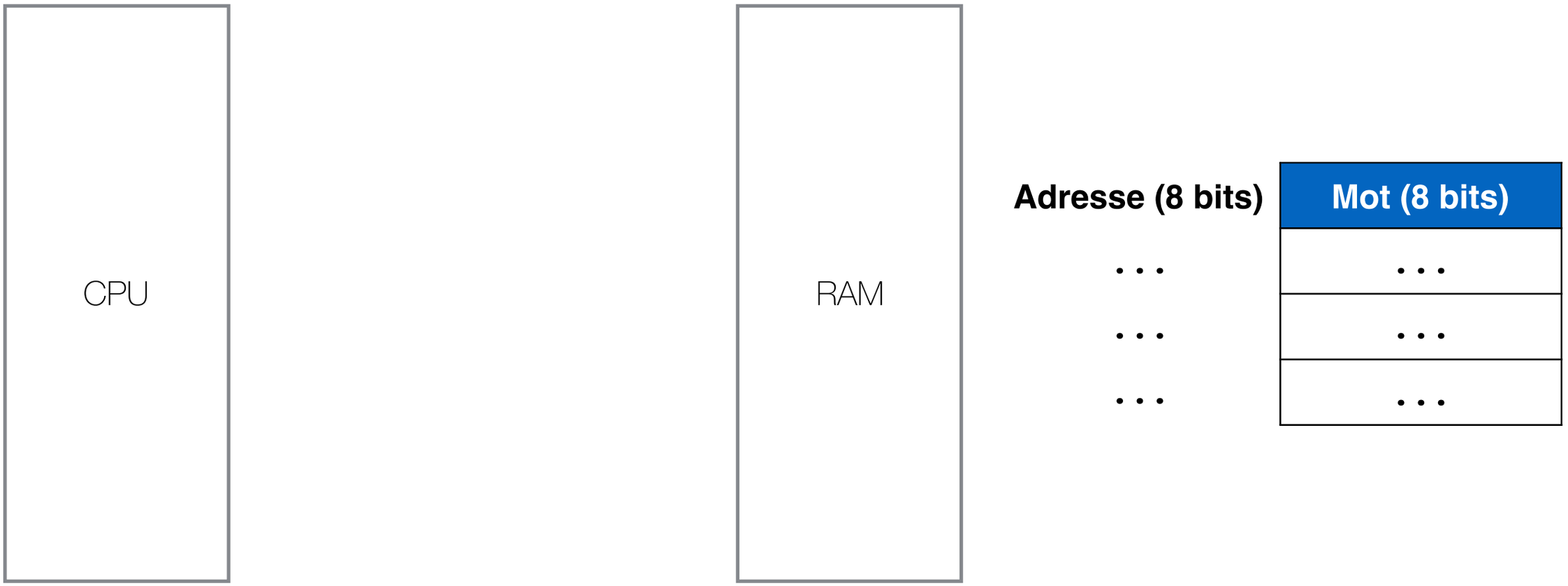
Types de mémoires

- Les mémoires peuvent être:
  - volatiles: perdent leur contenu lorsqu'elles perdent leur alimentation;
  - ou non-volatiles: conservent leur contenu même sans alimentation.
- Les mémoires volatiles peuvent être:
  - statiques: n'ont pas besoin d'être lues pour conserver leurs valeurs
  - dynamiques: nécessitent un rafraîchissement de leur données de façon périodique. Si les données d'une mémoire dynamique ne sont pas "lues" régulièrement, elles s'effacent.
- Les mémoires
  - ROM: ne peuvent pas être écrites (Read Only Memory)
  - RAM: peuvent être écrites (Random Access Memory);
- Les noms sont donnés aux mémoires en fonction de ces caractéristiques. Par exemple, SRAM est de la RAM Statique.

Types de mémoires

- Pour le moment, les deux types de mémoires qui nous intéressent sont:
  - ROM: ne peuvent **pas** être écrites (Read Only Memory)
    - doivent être écrites au préalable, une fois écrite on ne peut plus modifier leur contenu!
  - RAM: peuvent être écrites (Random Access Memory);

Construisons un ordinateur



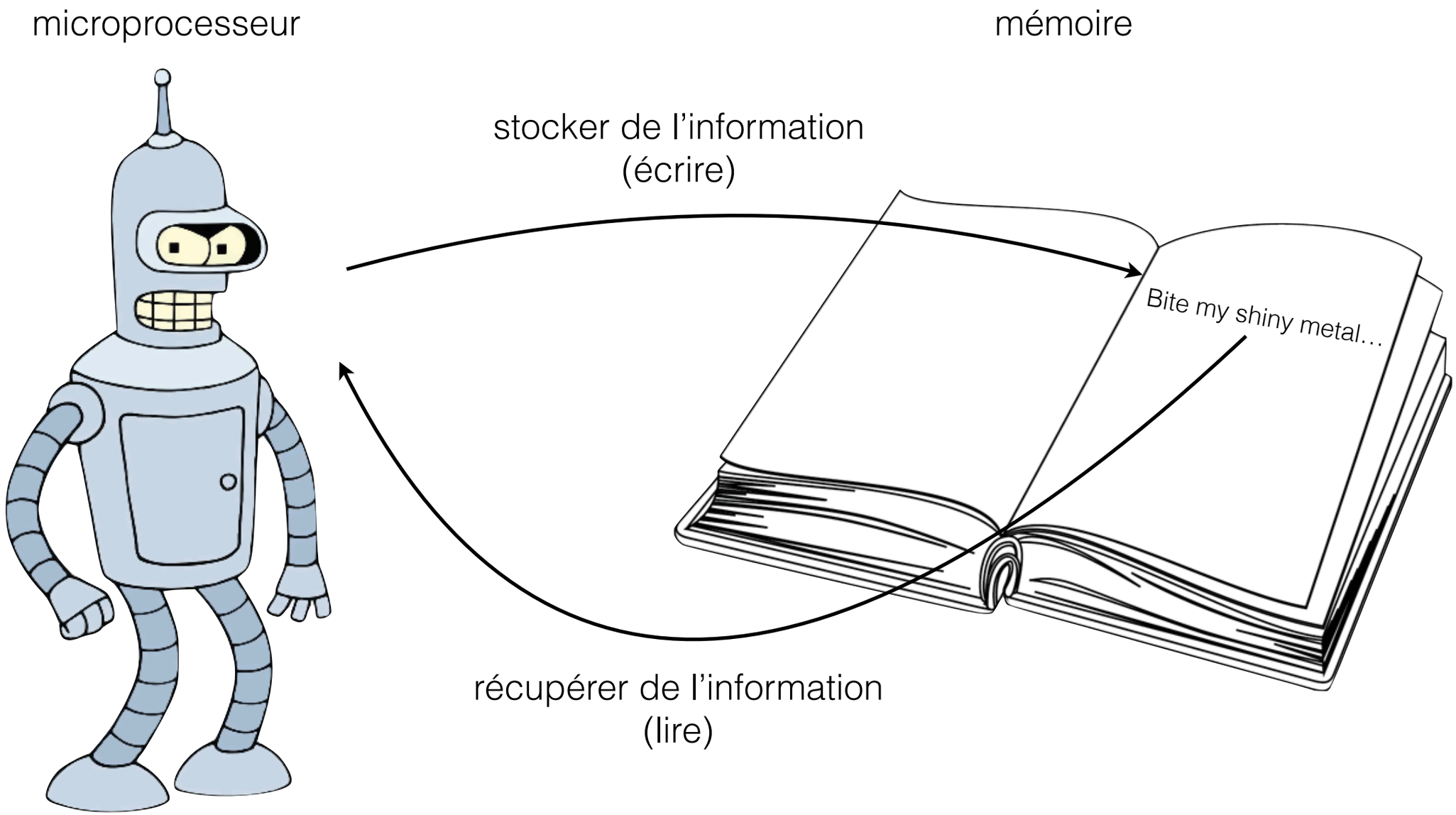
Nous avons nos deux composantes: un CPU et une RAM.

Comment faire pour que le CPU puisse accéder à la RAM?

À quoi sert la mémoire?

À stocker et récupérer de l'information.

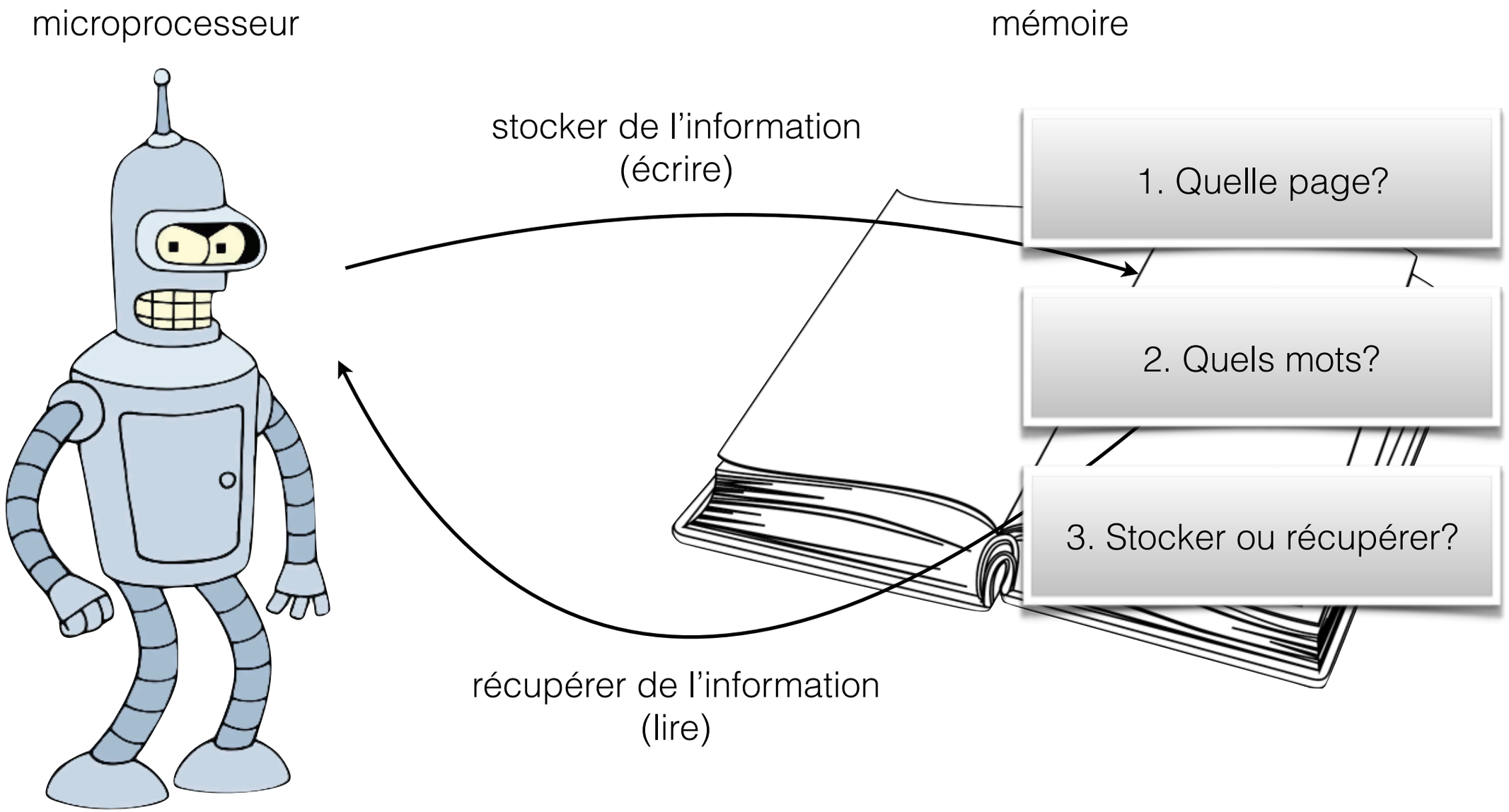
À se souvenir, pardi!



À quoi sert la mémoire?

À stocker et récupérer de l'information.

À se souvenir, pardi!

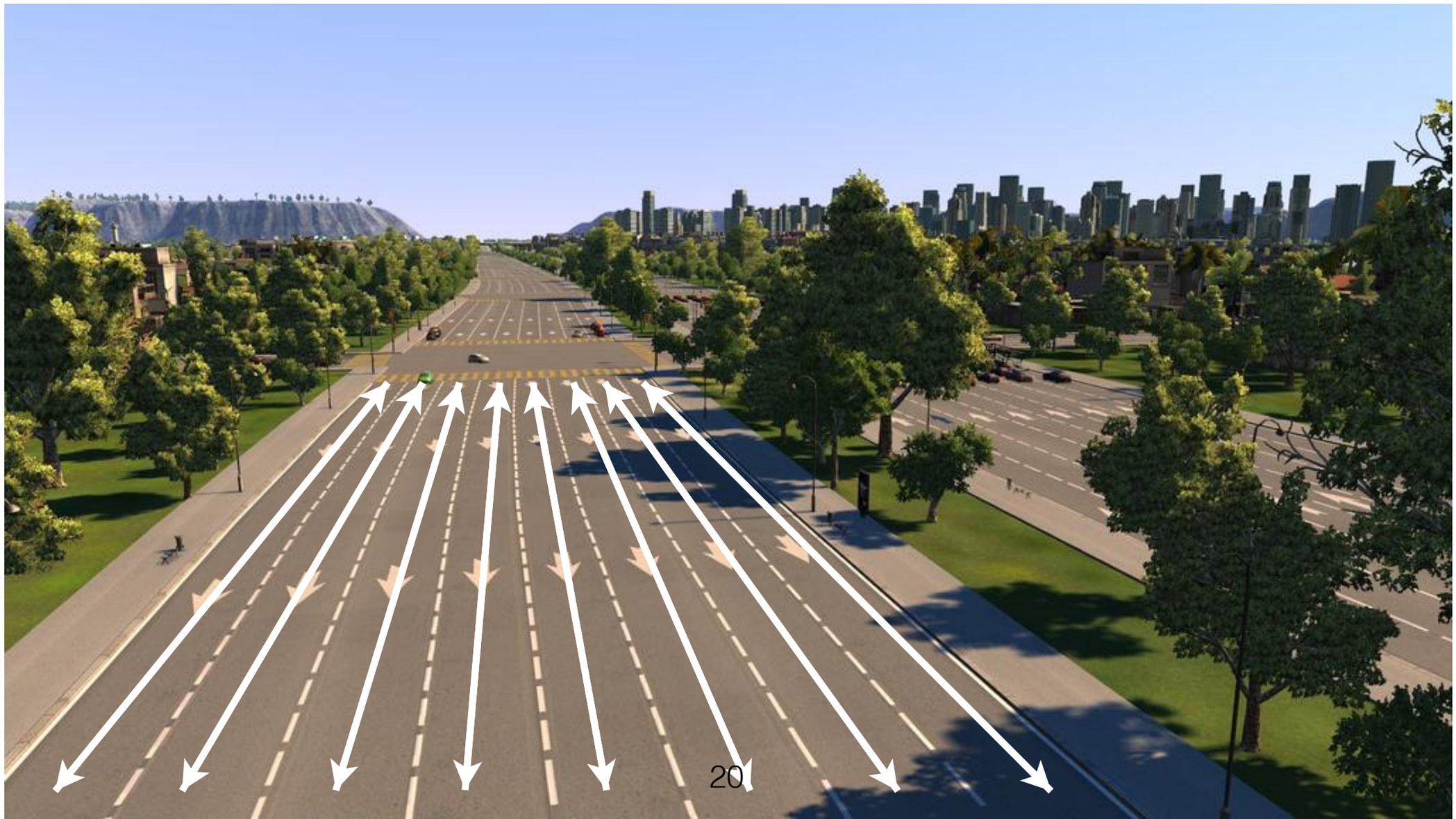


Bus

- Un bus est un groupe de lignes électriques qui relie le CPU aux autres composantes.
- Chaque ligne peut transférer un bit d’information à la fois.

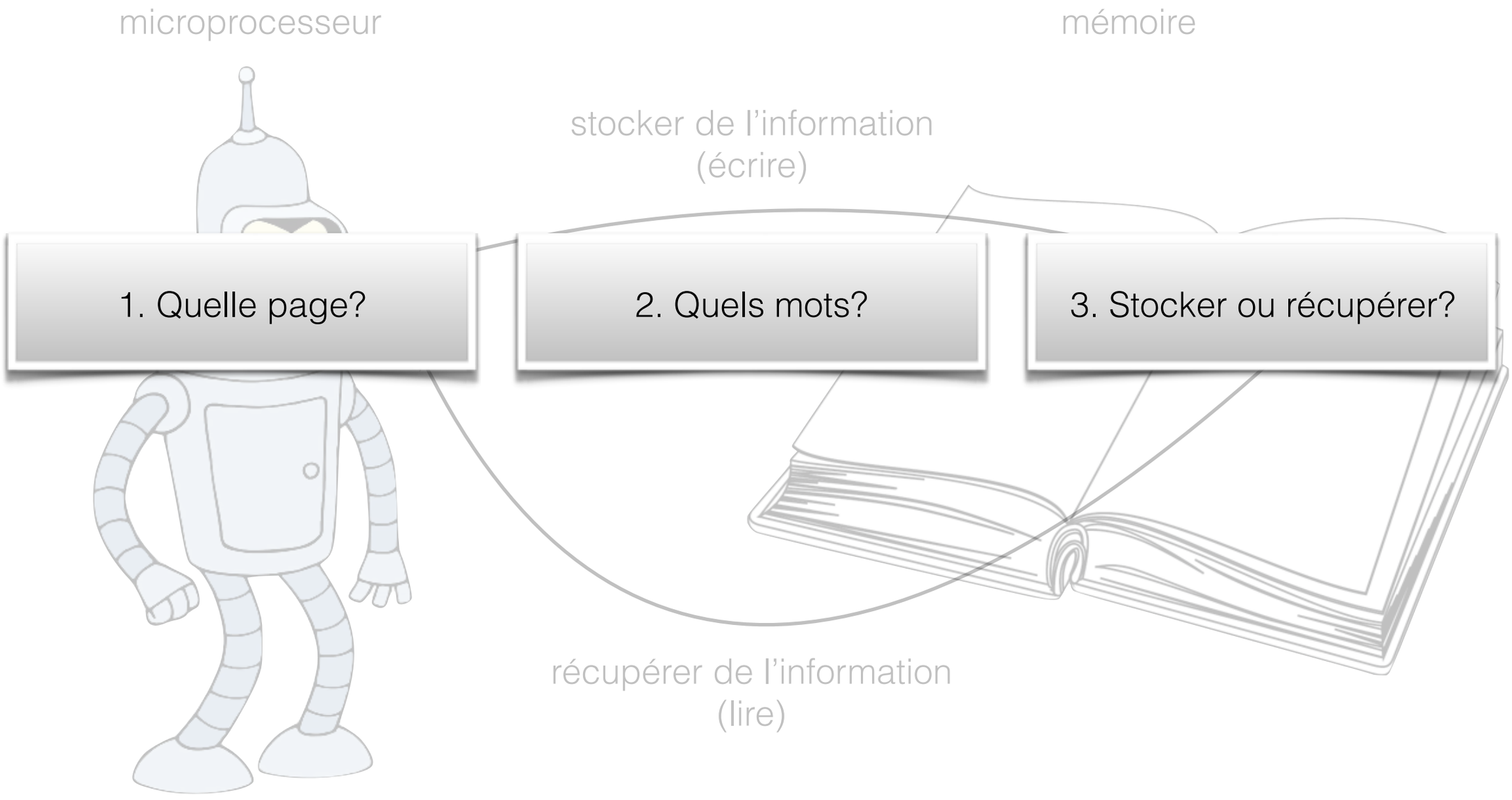
Analogie (bus): autoroutes

- Sauf que les voitures…
  - démarrent et arrivent à destination en même temps
  - ne peuvent circuler que dans un sens à la fois



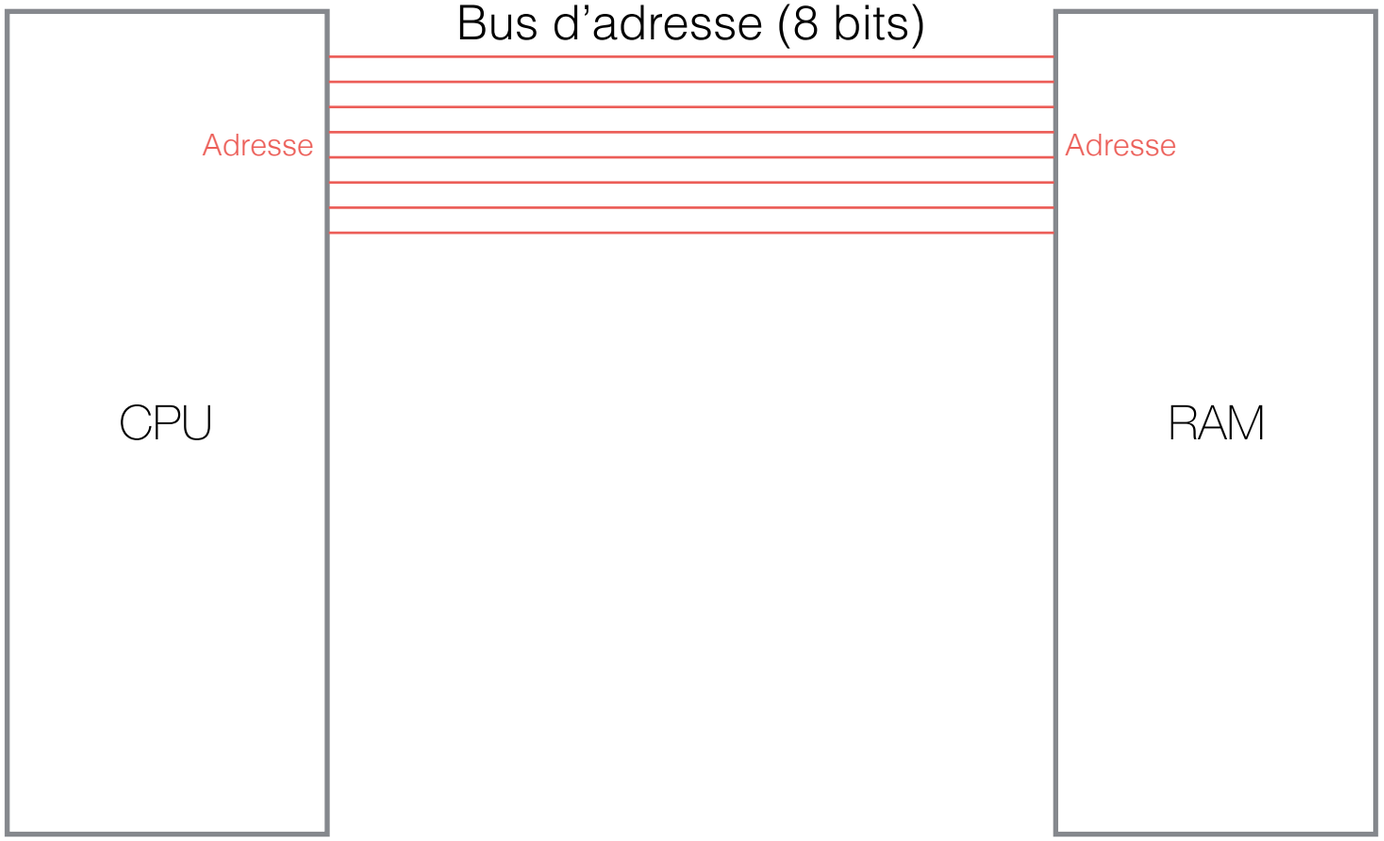
À quoi sert la mémoire?

À stocker et récupérer de l’information.



1. Quelle page?

Bus d’adresses

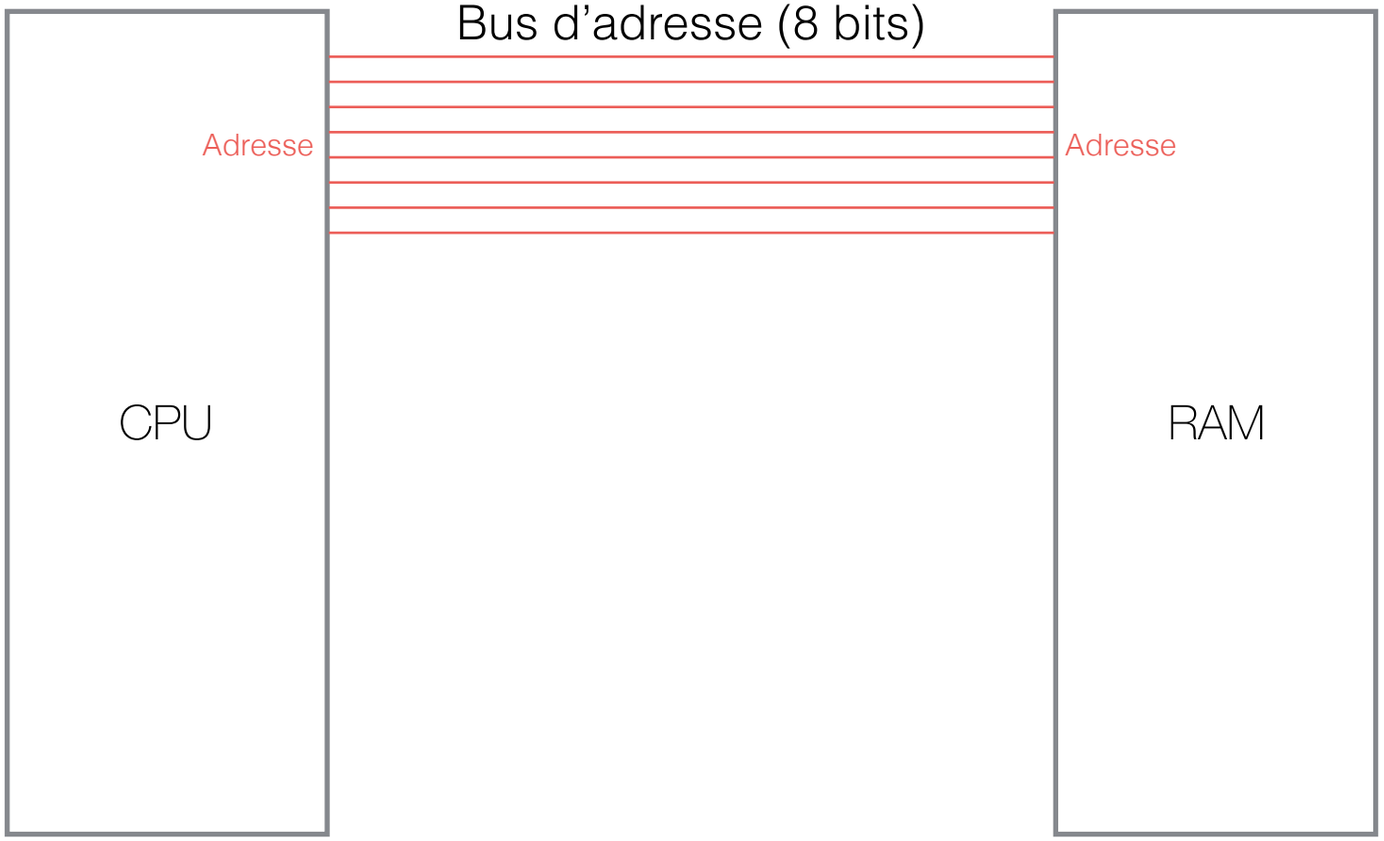


Adresse (8 bits)	Mot (8 bits)
...	...
...	...
...	...

- Le **bus d’adresse** indique l’emplacement de la mémoire (ou des périphériques) visé par la transaction sur le bus.
- C’est le microprocesseur qui le contrôle (place des adresses sur le bus).
- La taille du bus d’adresse (le nombre de lignes) détermine la quantité maximum de mémoire (ou d’entrées-sorties) que le CPU peut utiliser

1. Quelle page?

Bus d'adresses



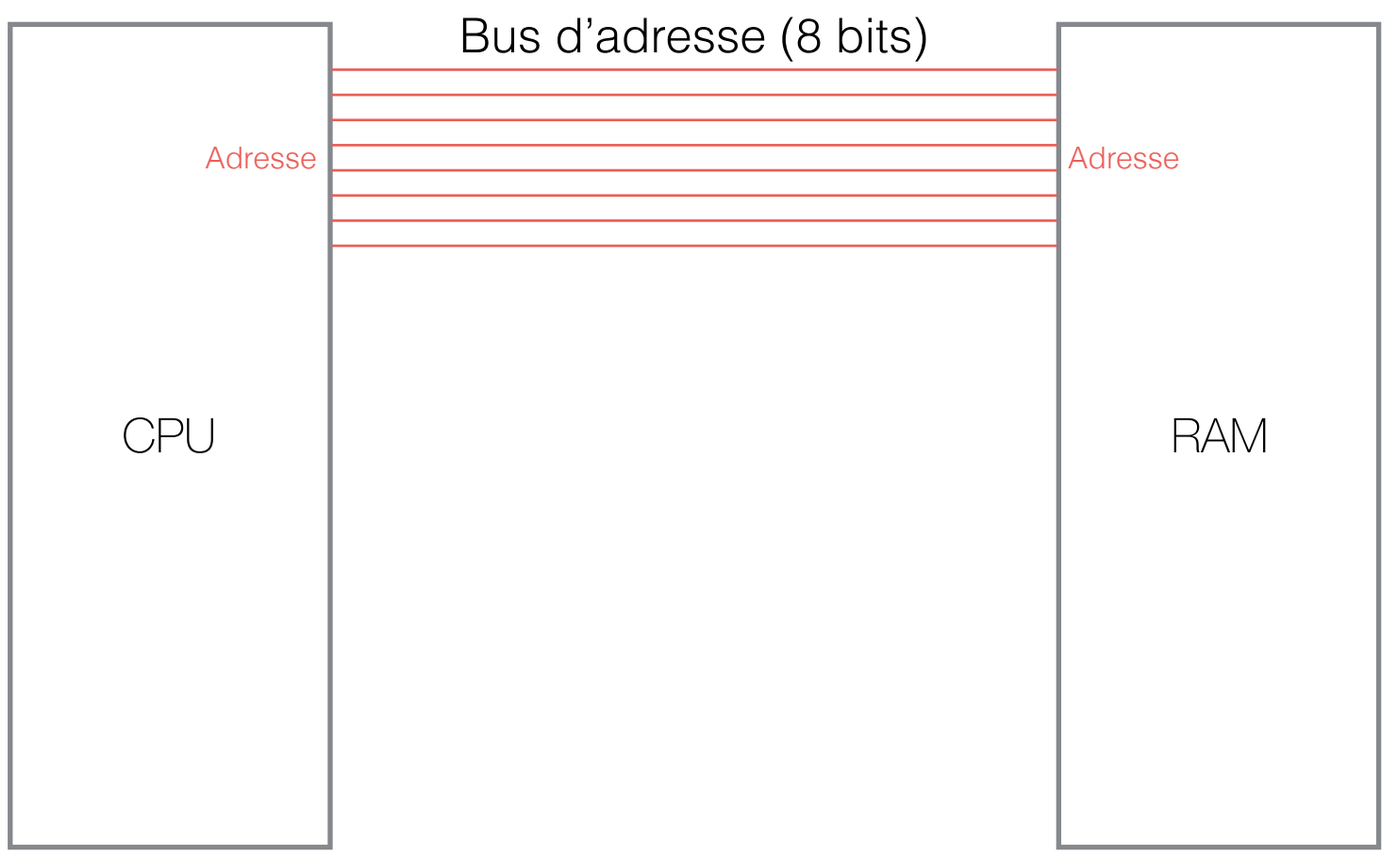
Adresse (8 bits)	Mot (8 bits)
...	...
...	...
...	...

- Le **bus d'adresse** indique l'emplacement de la mémoire (ou des périphéri
- C'est le m sur le bus).
- La taille d quantité maximum de mémoire (ou d'entrées-sorties) que le CPU peut utiliser

Combien d'adresses le CPU peut-il générer?

1. Quelle page?

Bus d'adresses



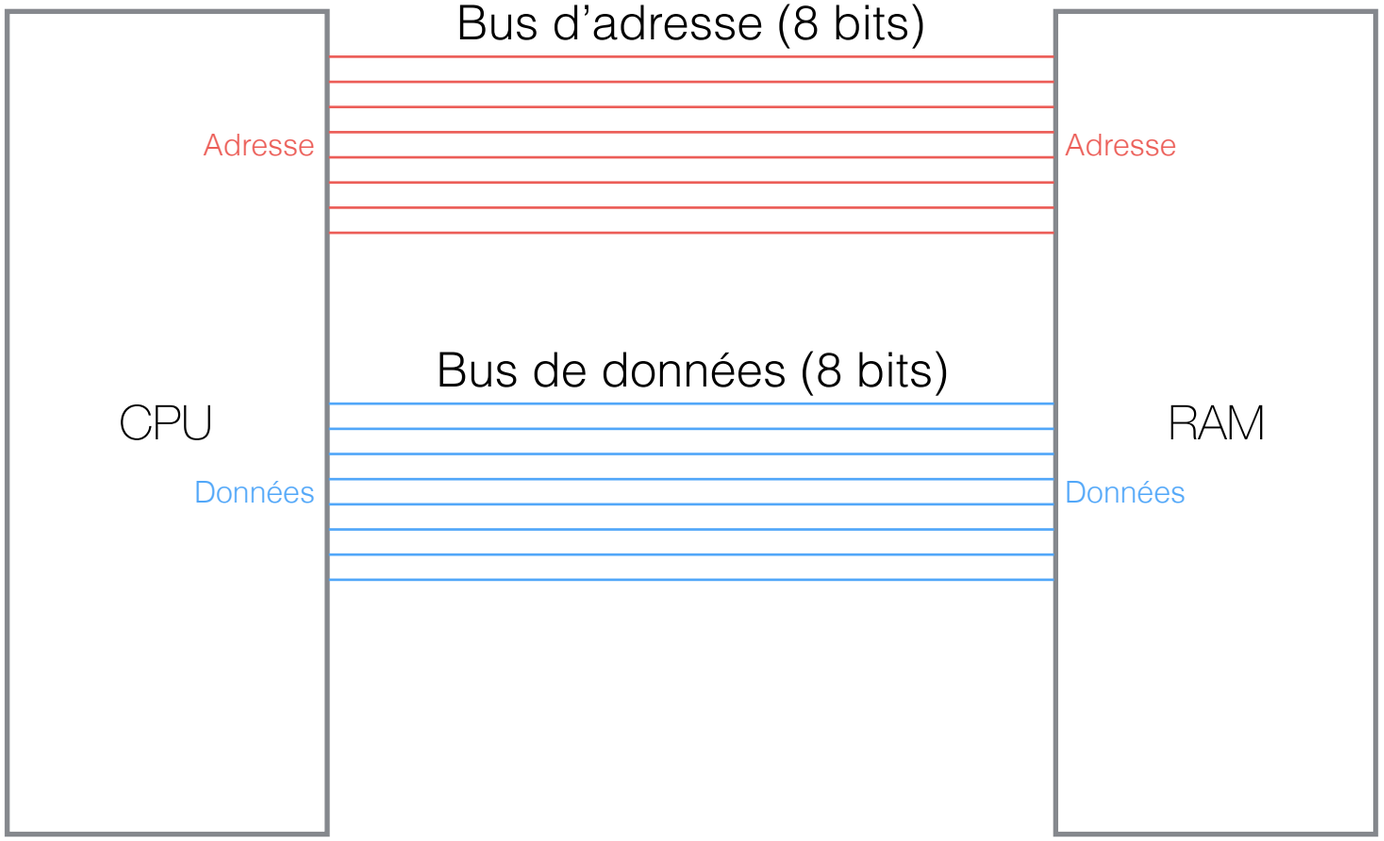
Adresse (8 bits)	Mot (8 bits)
...	...
...	...
...	...

- Le **bus d'adresse** indique l'emplacement de la mémoire (ou des périphéri
- C'est le m sur le bus).
- La taille d quantité maximum de mémoire (ou d'entrées-sorties) que le CPU peut utiliser

Combien d'adresses le CPU peut-il générer?

$2^8 = 256$ adresses

Bus de données

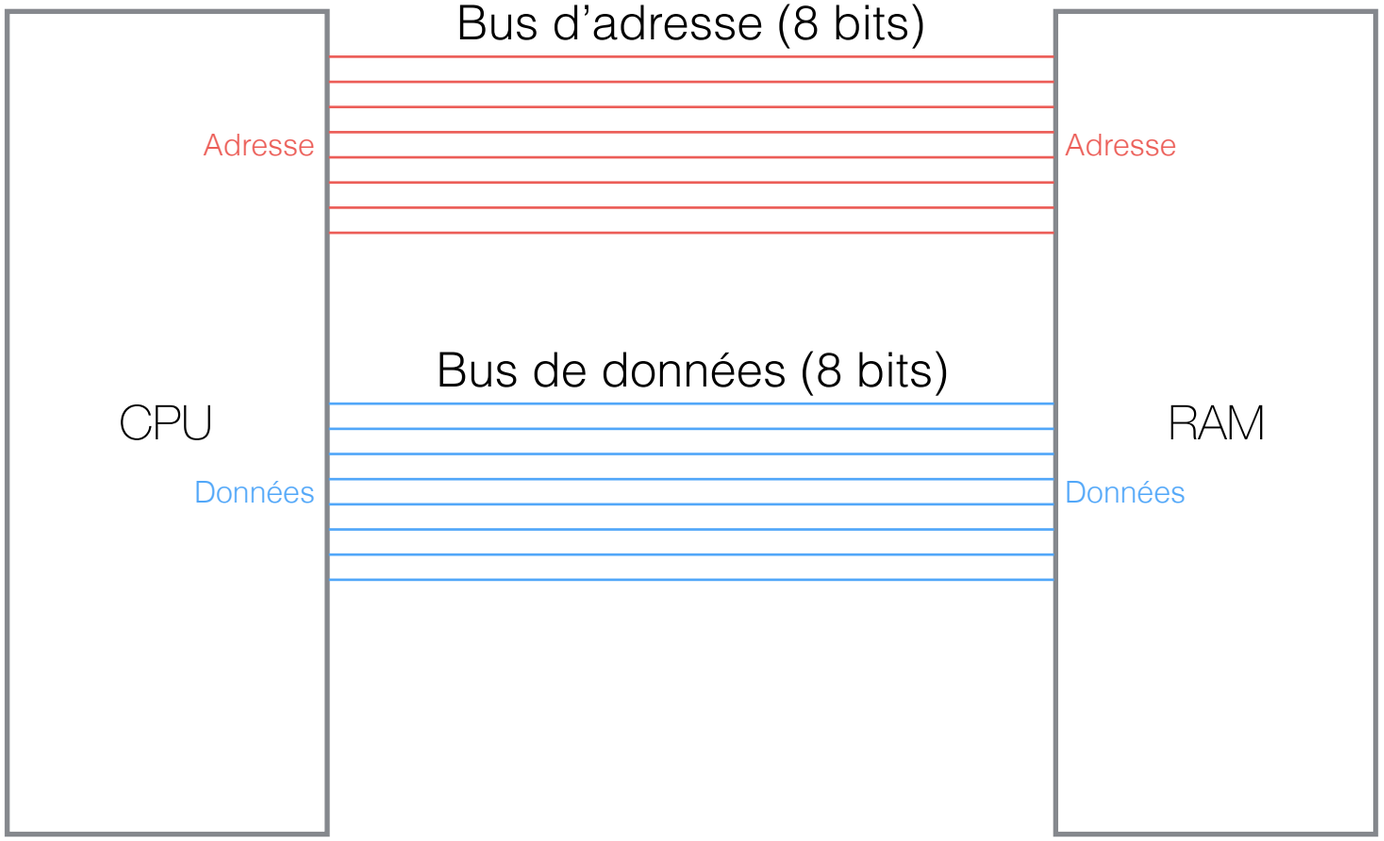


Adresse (8 bits)	Mot (8 bits)
...	...
...	...
...	...

- Le **bus de données** permet le transfert des données.
- Les données peuvent circuler dans les deux sens, mais elles ne circulent que dans un seul sens à la fois.
- La taille du bus de données (le nombre de lignes) détermine la grandeur maximale des mots pouvant être transférés d'un coup.

2. Quels mots?

Bus de données



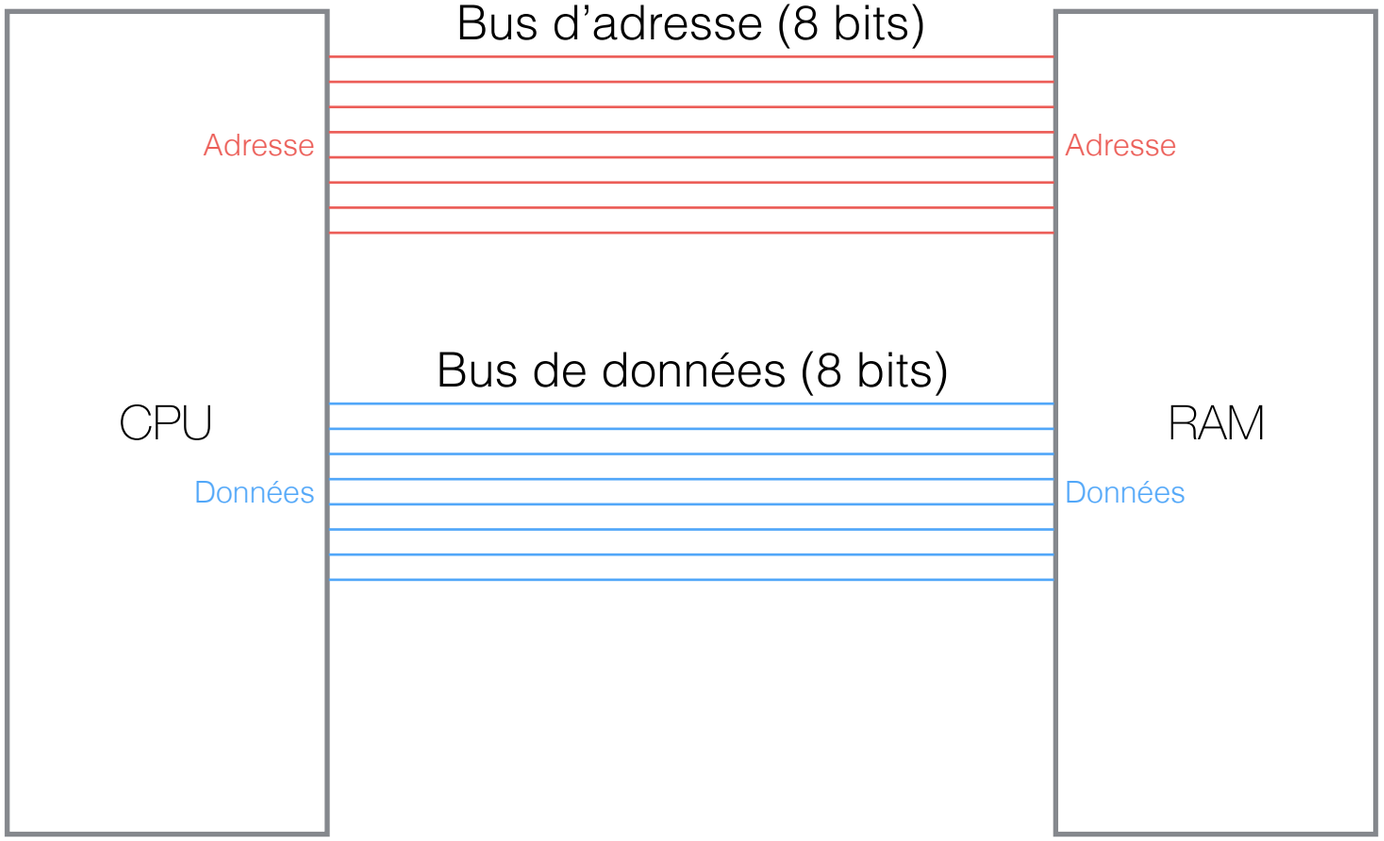
Adresse (8 bits)	Mot (8 bits)
...	...
...	...
...	...

- Le **bus de données** permet le transfert des données.
- Les don elles ne circulen
- La taille mine la grandeur maximale des mots pouvant être transférés d'un coup.

Que faire si je dois écrire 2 caractères ASCII (8 bits) en mémoire?

2. Quels mots?

Bus de données



Adresse (8 bits)	Mot (8 bits)
...	...
...	...
...	...

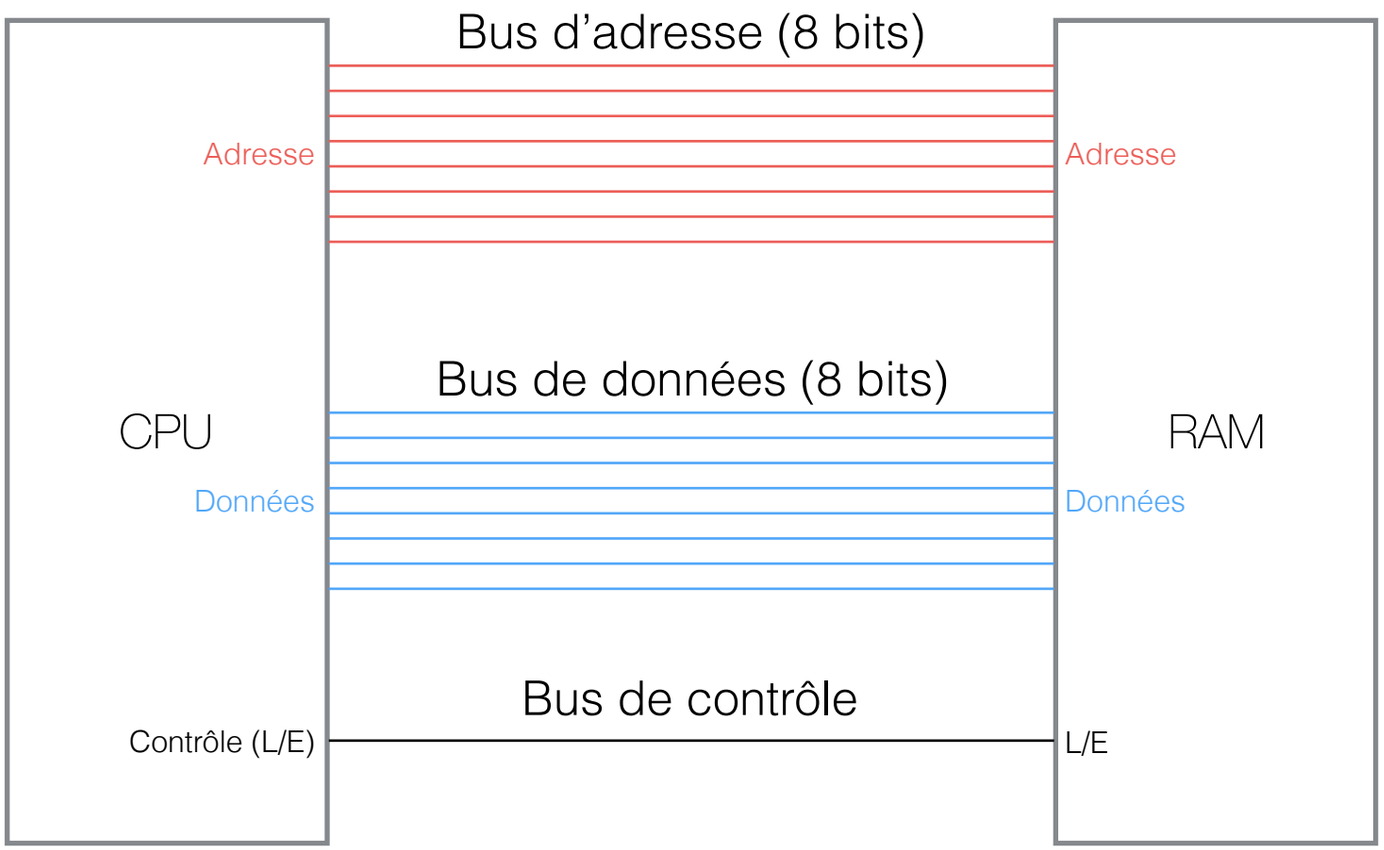
- Le **bus de données** permet le transfert des données.
- Les don elles ne circulen
- La taille mine la grandeur maximale des mots pouvant être transférés d'un coup.

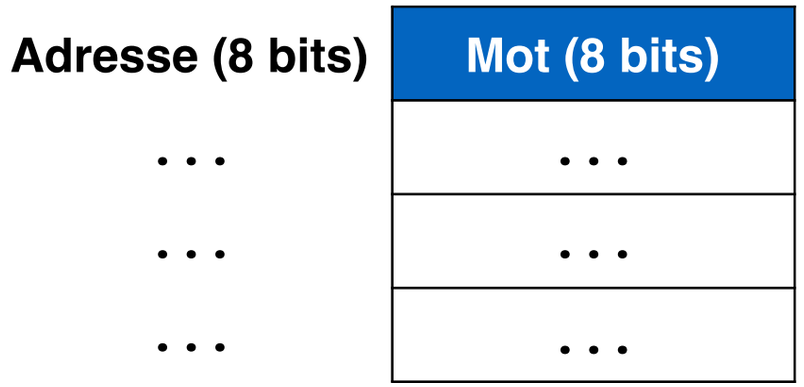
Que faire si je dois écrire 2 caractères ASCII (8 bits) en mémoire?

Chaque caractère nécessite 8 bits et le bus de données a 8 bits.
Il faudra donc effectuer 2 transferts
pour écrire les 2 caractères en mémoire.

3. Stocker ou récupérer?

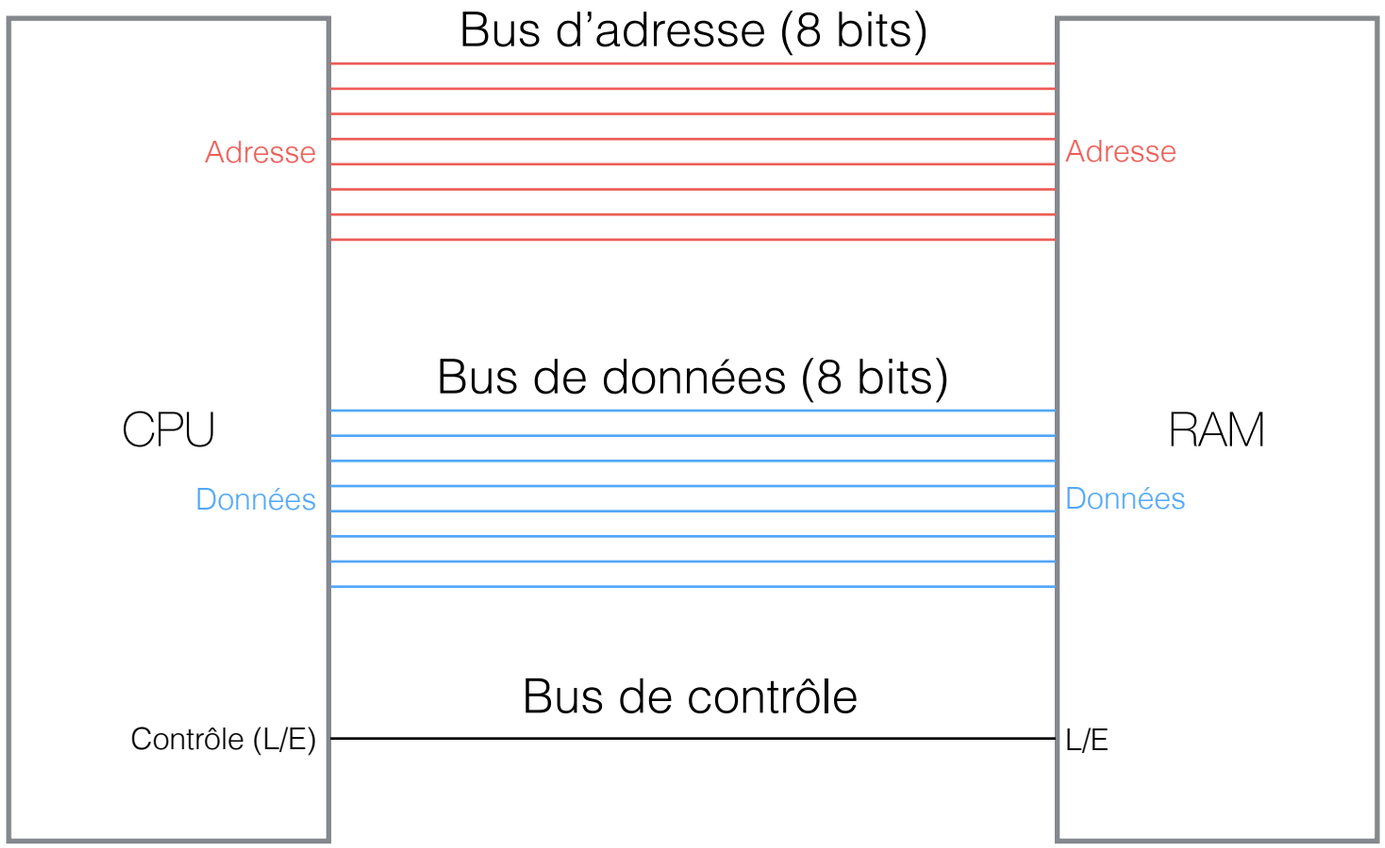
Bus de contrôle



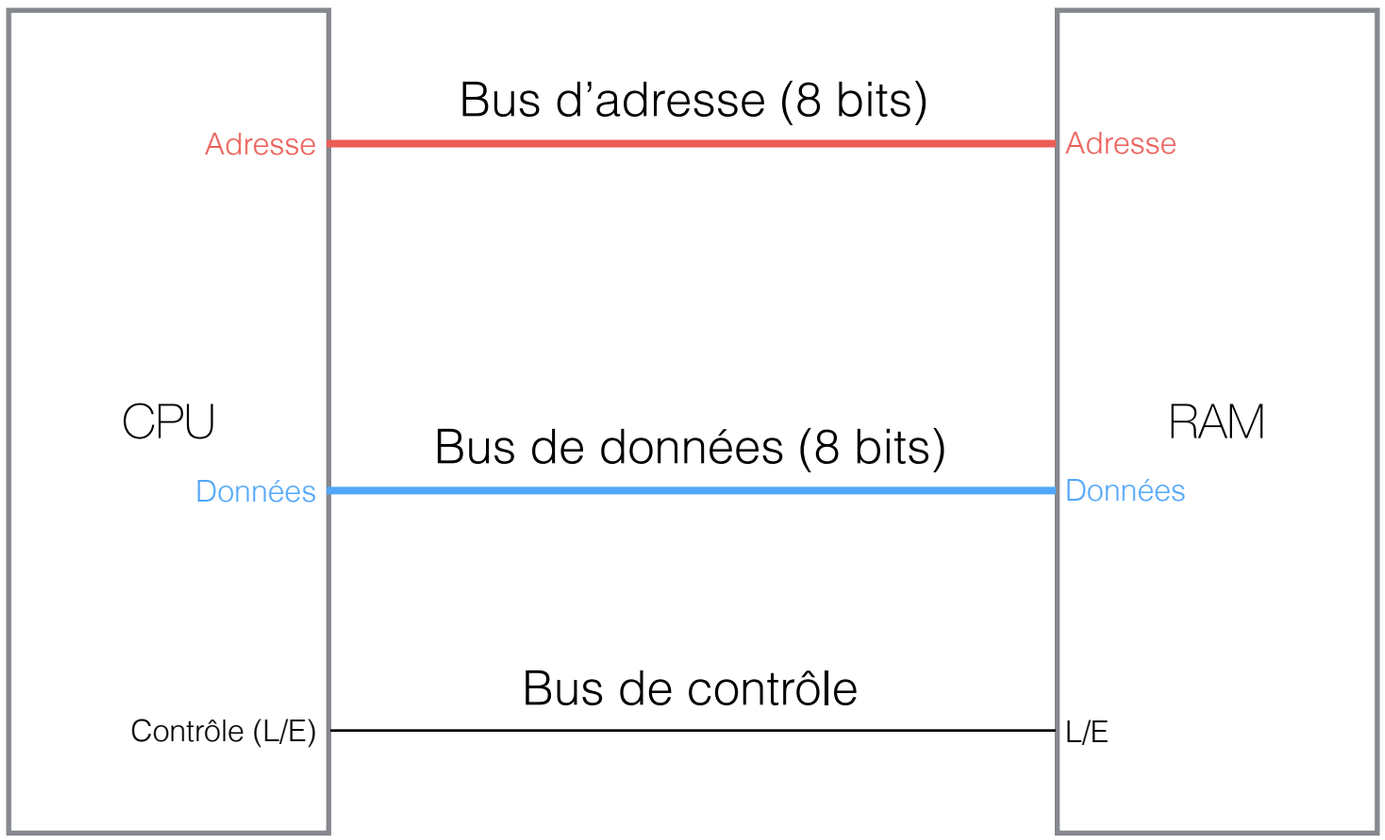


- Le **bus de contrôle** contrôle l'utilisation des bus de données et d'adresse.
- Il permet de gérer la direction des données sur le bus des données (lecture ou écriture).
- Le bus de contrôle a aussi une horloge, qui détermine la vitesse à laquelle les données peuvent être transférées et qui synchronise les opérations

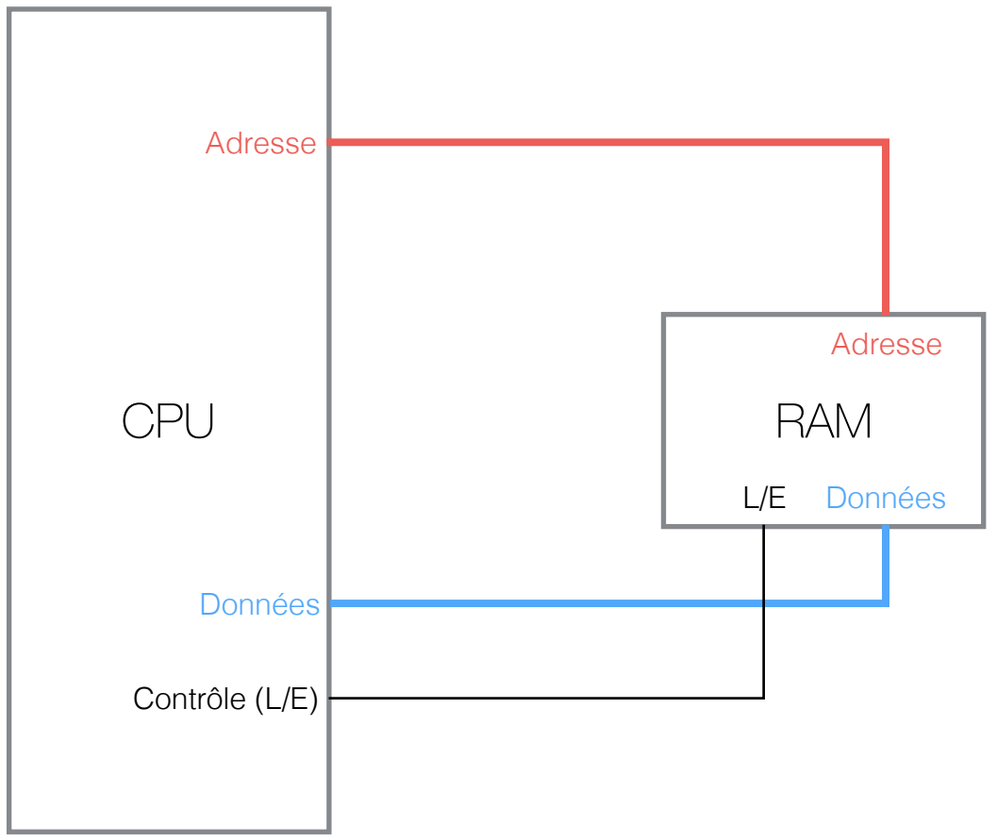
En résumé, 3 bus importants



En résumé, 3 bus importants



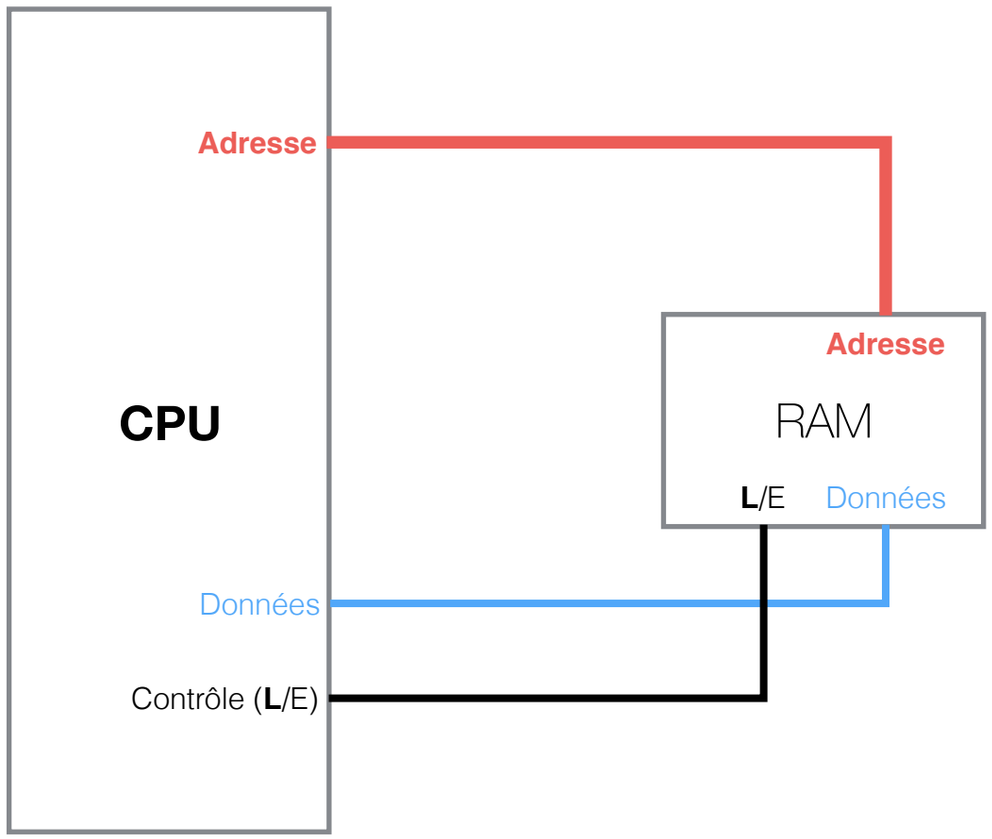
Lecture d’une donnée en mémoire



Adresse (8 bits)	Mot (8 bits)
0x40	0x12
0x41	...
0x42	...

- C’est le **CPU** qui initie toute action.
- Une lecture s’effectue en deux temps.

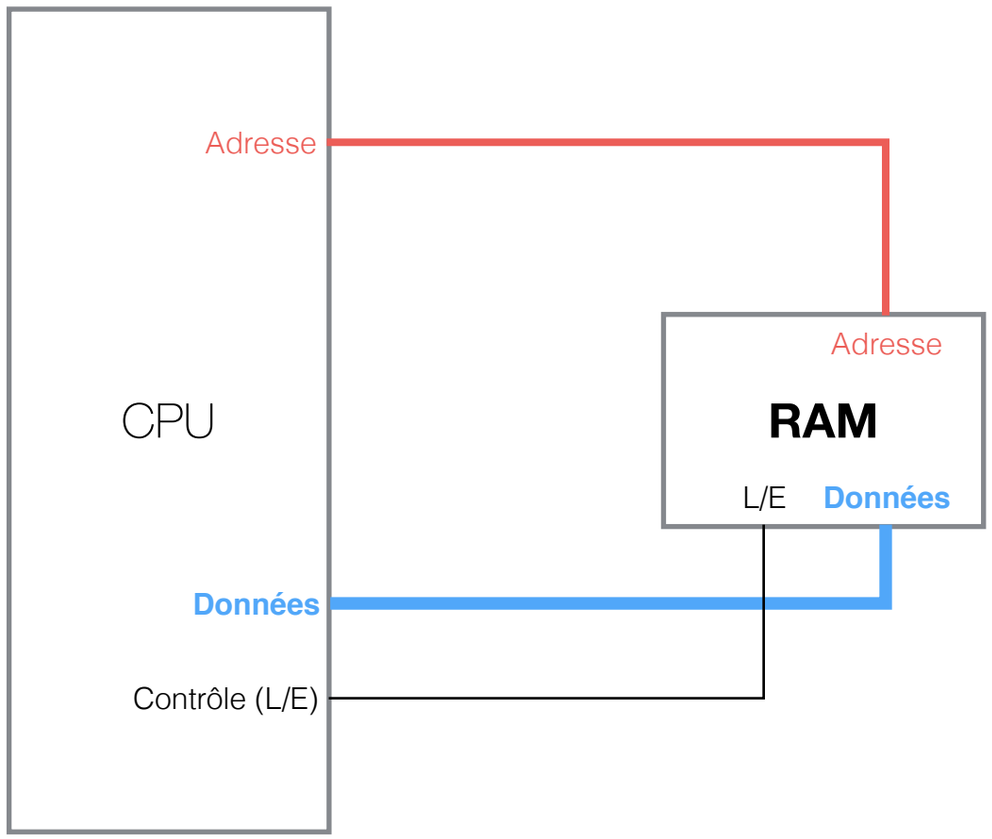
Lecture d’une donnée en mémoire (1/2)



Adresse (8 bits)	Mot (8 bits)
0x40	0x12
0x41	...
0x42	...

- Le CPU:
  - place l’adresse (ex: 0x40) sur le bus d’adresses;
  - active le bus de contrôle en lecture.

Lecture d’une donnée en mémoire (2/2)

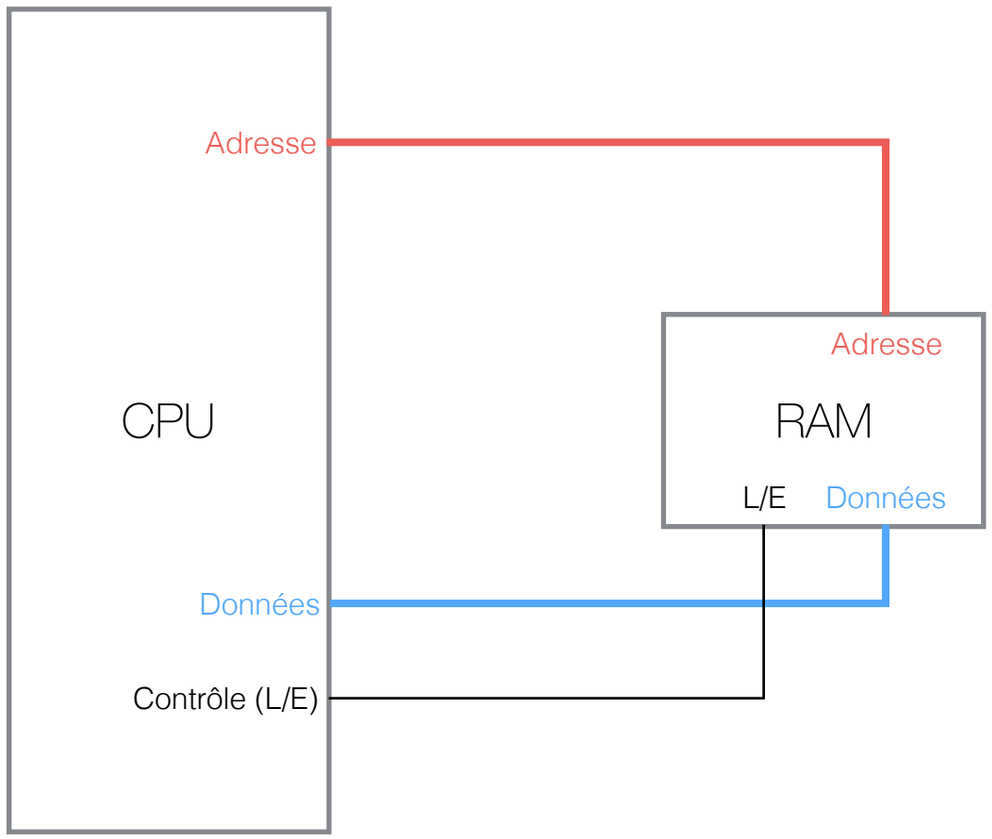


Adresse (8 bits)	Mot (8 bits)
0x40	0x12
0x41	...
0x42	...

- La mémoire:
  - place la donnée correspondante sur le bus de données;
- Le CPU:
  - récupère la donnée (ex: 0x12) sur le bus de données (et la place dans un registre).

Nous parlerons des registres très bientôt!

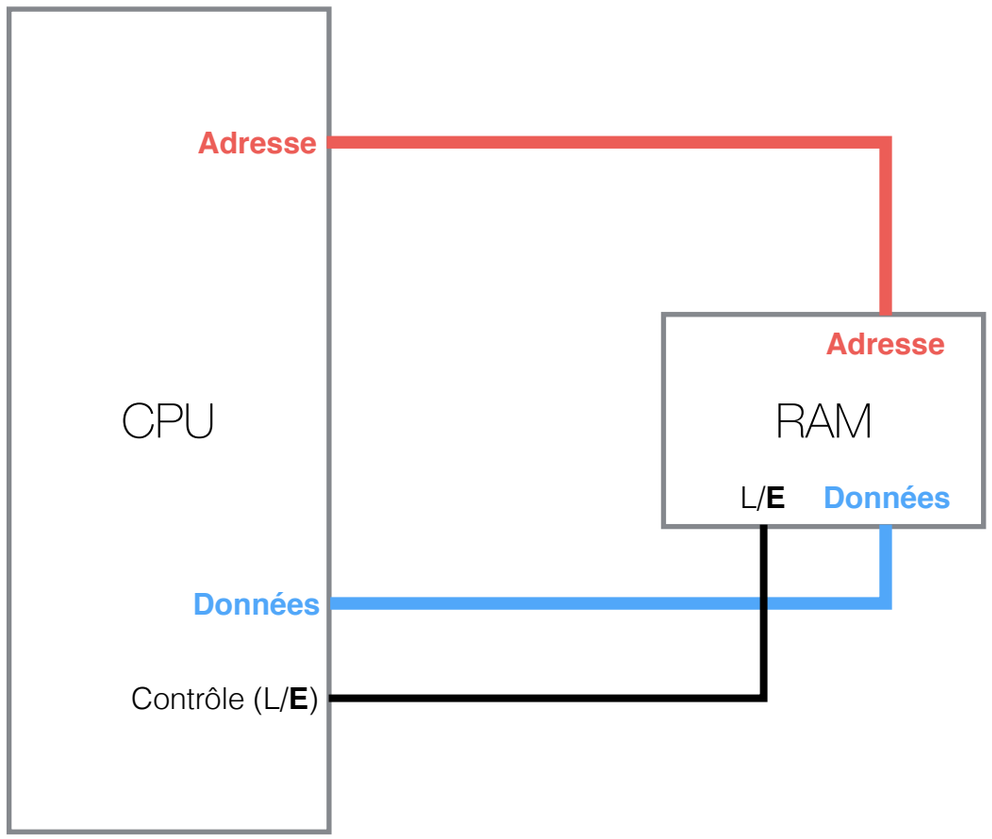
Écriture d’une donnée en mémoire



Adresse (8 bits)	Mot (8 bits)
0x40	0x12
0x41	...
0x42	...

- C’est le **CPU** qui initie toute action.
- Une écriture s’effectue en deux temps.

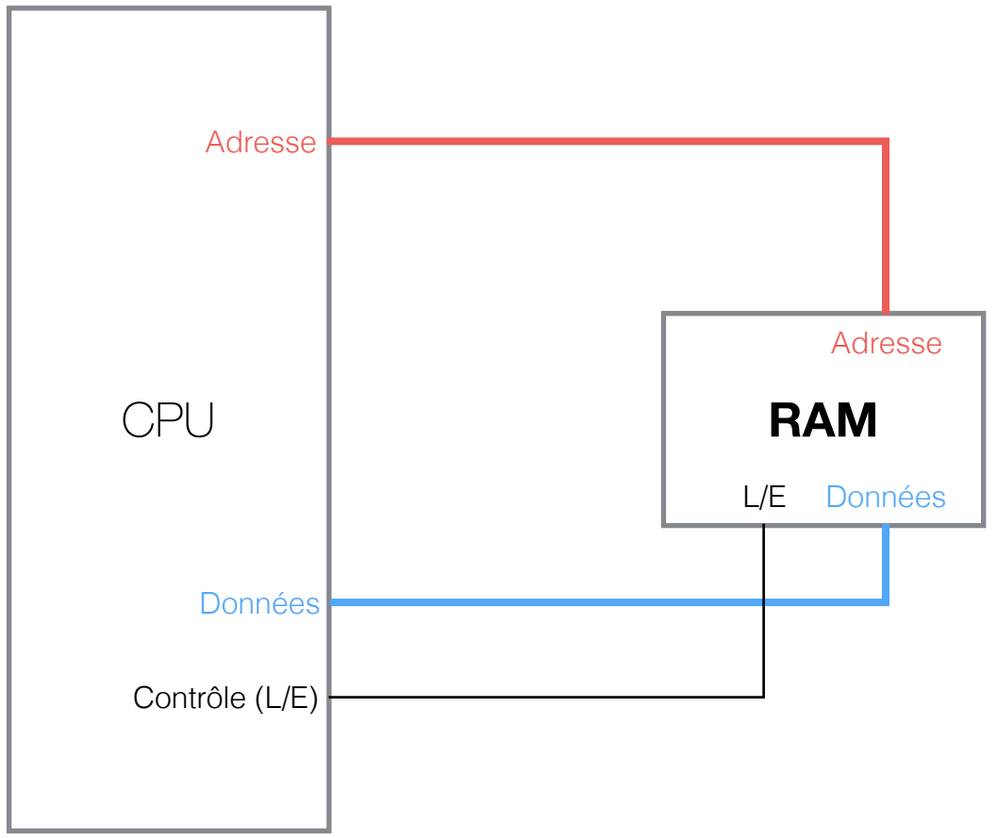
Écriture d’une donnée en mémoire (1/2)



Adresse (8 bits)	Mot (8 bits)
0x40	0x12
0x41	...
0x42	...

- Le CPU:
  - place l’adresse (ex: 0x40) sur le bus d’adresses;
  - la donnée à écrire (ex: 0xAB) sur le bus de données;
  - active le bus de contrôle en écriture.

Écriture d’une donnée en mémoire (2/2)



Adresse (8 bits)	Mot (8 bits)
0x40	**0xAB**
0x41	...
0x42	...

- La mémoire:
  - récupère la donnée (ex: 0xAB) sur le bus de données;
  - l’écrit à l’adresse disponible sur le bus d’adresse (ex: 0x40).