Introduction

G. Dewaele

Contact

guillaume.dewaele@sci-phy.org (ou ENT...)

De l'utilisation des heures de colles

TP en 1/3 de classe (deux semaines sur trois)

Colles à compter de courant octobre

(4 séquences réparties sur l'année, et TP+débriefing)

Déroulement de l'année

Introduction au C / programmation impérative

Introduction au Caml / programmation fonctionnelle

Représentation des données

Structures usuelles

(piles, files, dictionnaires, arbres...)

Stratégies de programmation

Graphes, algorithmique des textes, logique, bases de données

Qu'est-ce que l'informatique ?

Informatique : mot valise

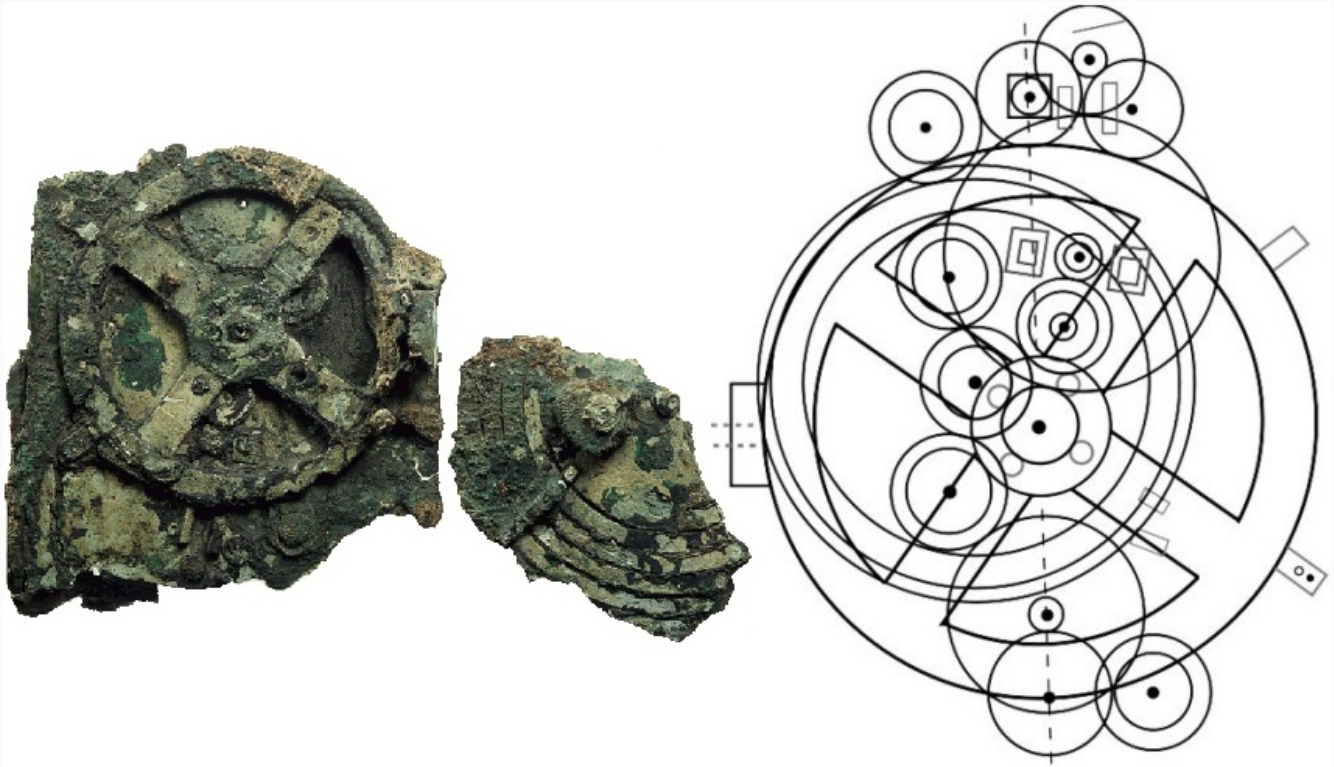
information + auto**matique**

→ science du traitement automatisé de l'information

Un *ordinateur* est un dispositif concrétisant cette notion

La machine d'Anticythère

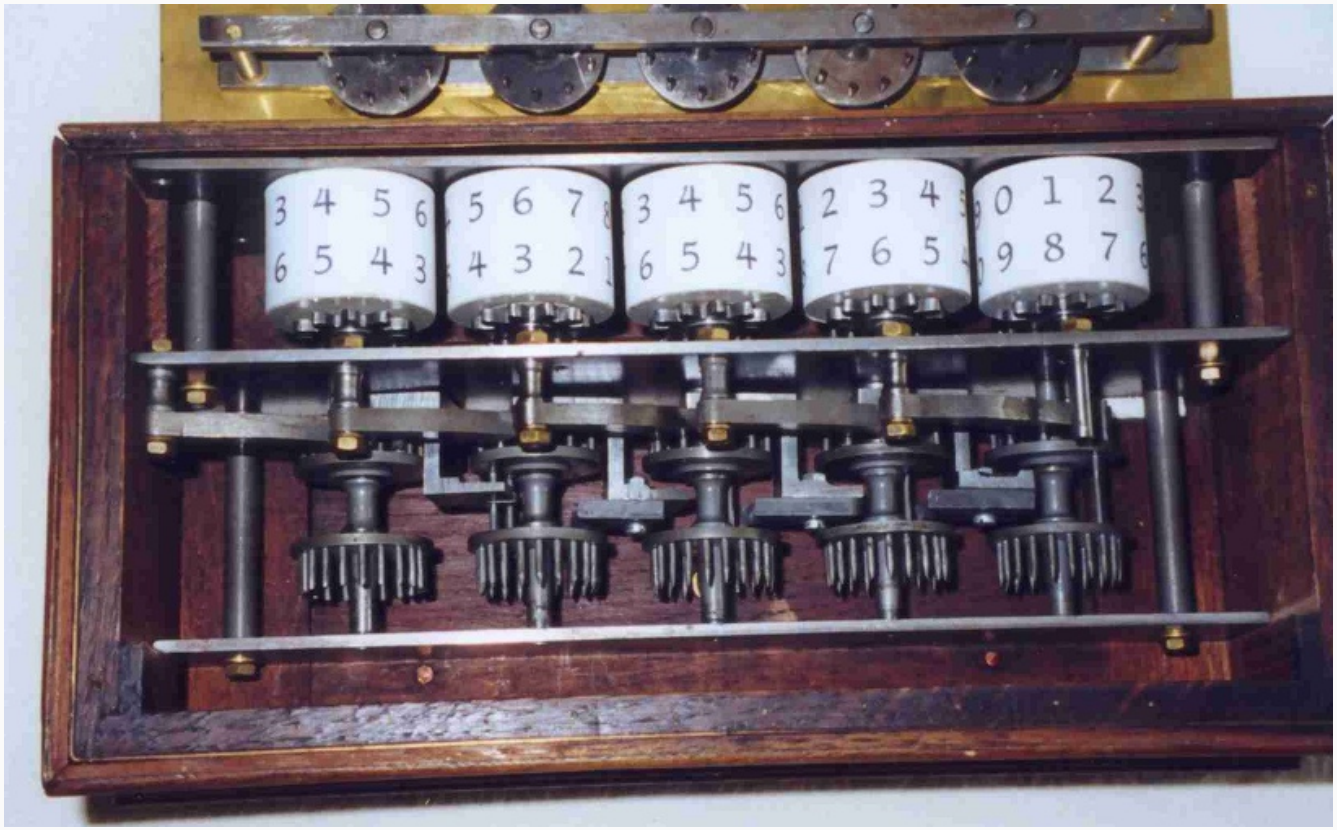
Nommée d'après le lieu de découverte, environ 2000 ans



Détermine la date d'événements astronomiques (éclipses, ...) grâce à un mécanisme contenant plus de 50 engrenages

La Pascaline

Machine à calculer mécanique (additions, soustractions)



Créée par Blaise Pascal vers 1640.

Les métiers à tisser Jacquard

Cartes perforées automatisant la production de motifs (1801)

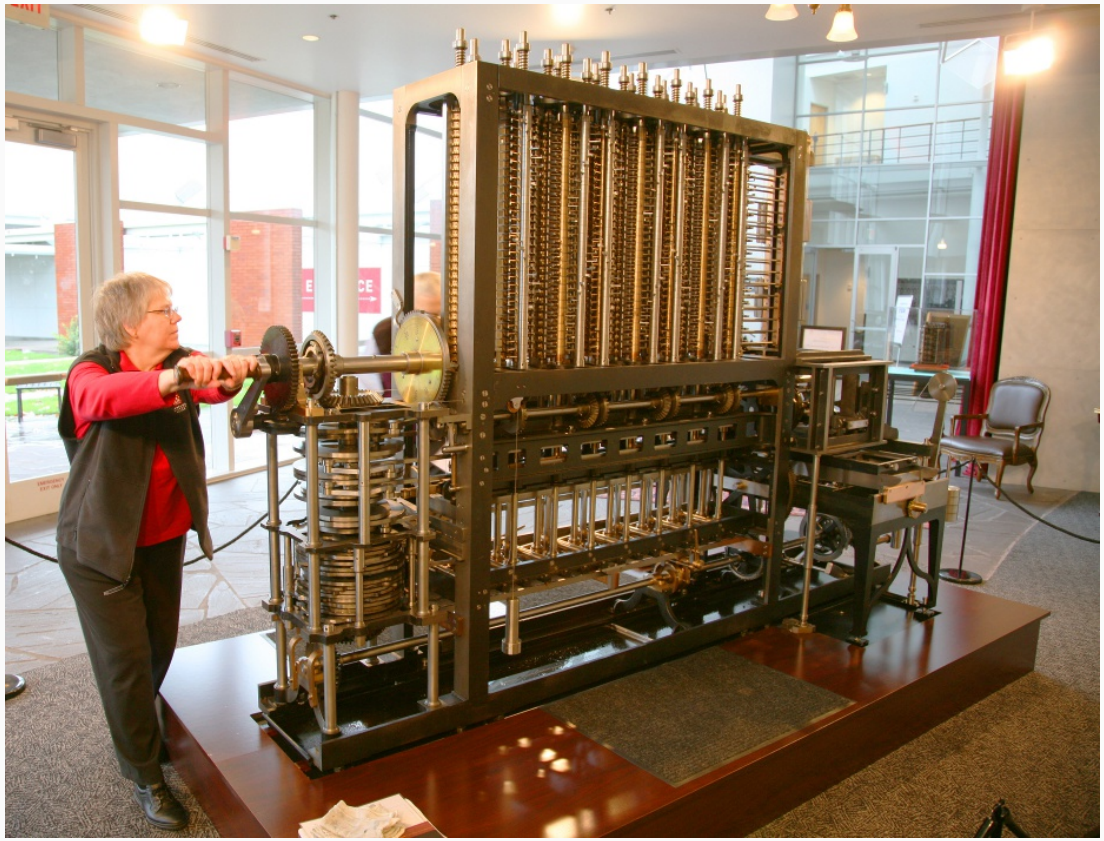
Premiers programmes !

La machine à différences de Charles Babbage

Au XIX^{e} siècle, grand besoin de tables de logarithmes

Problème : trop d'erreurs (de calcul et de copie)

le gouvernement anglais veut une machine pour les produire



La machine analytique de Babbage

C. Babbage préférerait une machine *programmable*

utilisant des cartes perforées

Il la conçoit et la décrit, mais ne peut achever sa construction

Ada Lovelace l'aide et conçoit les premiers programmes

La machine analytique de Babbage

Construit en partie vers 1900 par son fils cadet

Utilisée pour calculer des multiples de π (29 décimales)

Réalisée complètement à la fin du XX^{e} siècle

avec les techniques de l'époque

Elle fonctionne parfaitement

Fin de partie ?

En 1937, Kurt Pannke, fabricant de calculateurs mécaniques :

« on a à peu près tout inventé sur les calculateurs

il ne reste rien de fondamental à découvrir »

Les progrès dans les dix ans qui suivent sont immenses

Naissance de l'ordinateur

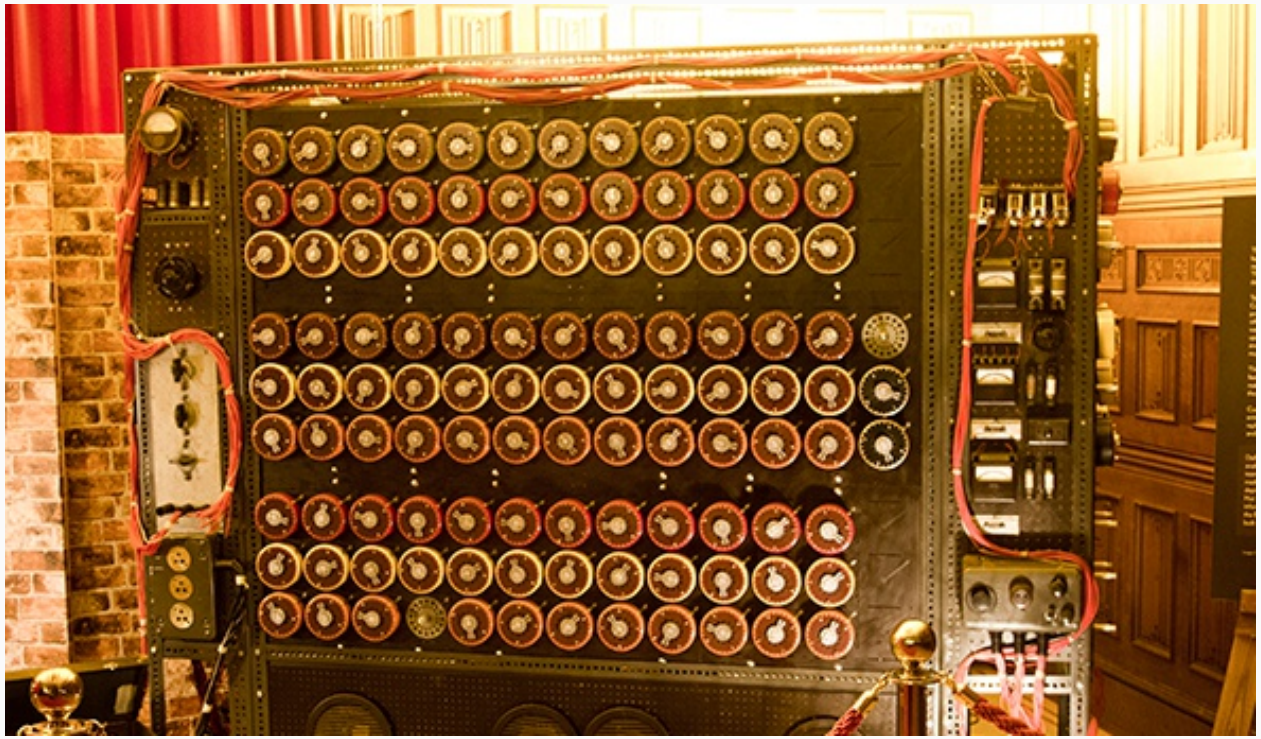
La seconde guerre mondiale accélère les choses !

Savoir calculer vite et automatiquement est un avantage :

- calculer une trajectoire prend des jours
- l'ingénierie (aéronautique...) nécessite beaucoup de calculs
- la cryptanalyse, impraticable à la main, enjeu stratégique

La bombe d'Alan Turing

Appareillage mécanique destiné à casser le cryptage *Enigma*



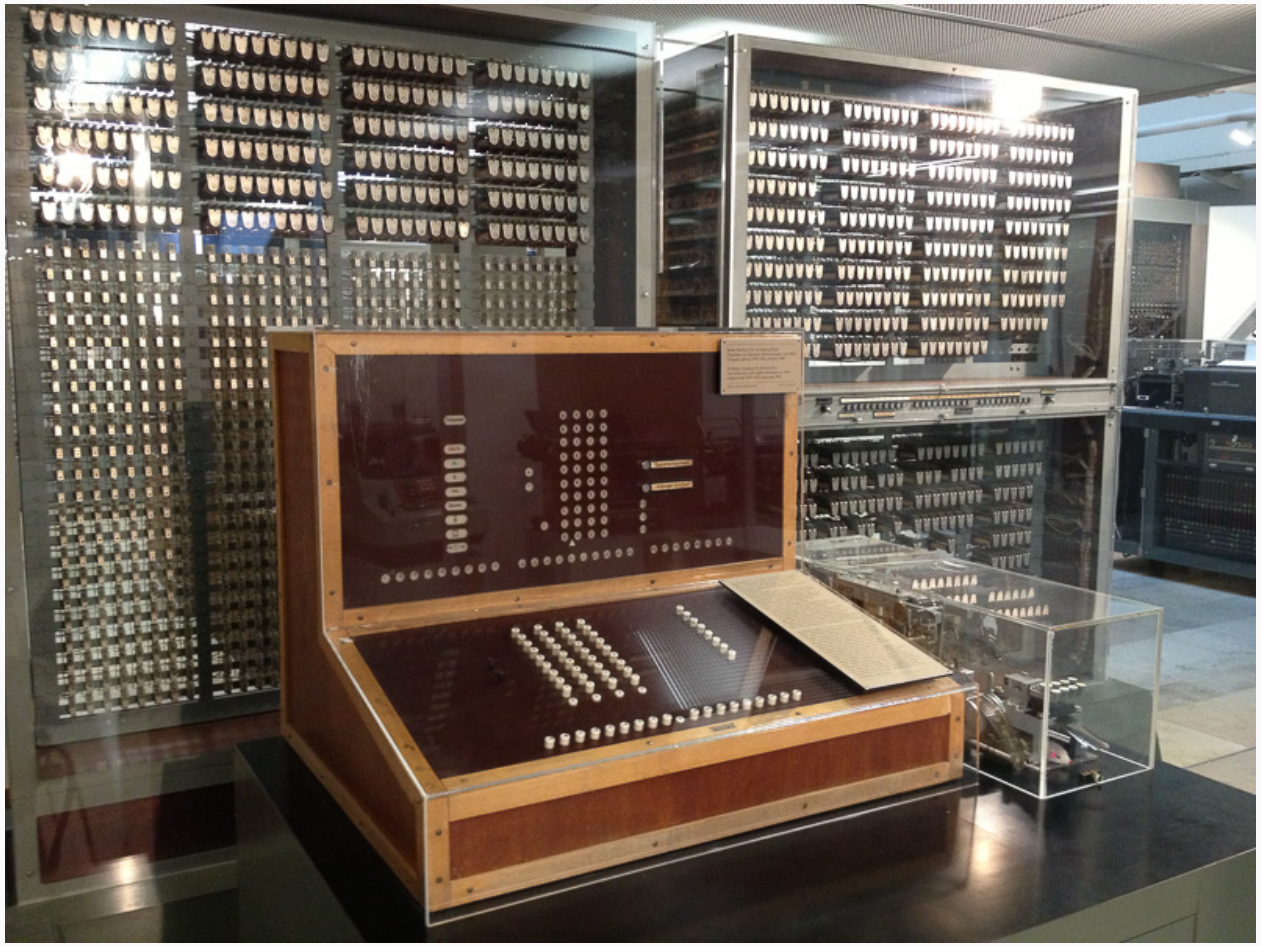
Essentiellement des dizaines de machines Enigma assemblées,

la bombe recherche la clé par force brute

Pas réellement un ordinateur encore...

Le Z3 de Konrad Zuse

Réalisé en 1941 par Konrad Zuse.



Détruit en 1943 dans un bombardement.

Le Z3 de Konrad Zuse

Beaucoup de concepts novateurs :

- complètement programmable (rubans perforés) ;
- calcule en binaire et en virgule flottante (22 bits) ;
- électrique (relais) plutôt que mécanique ;
- entrées par clavier et sorties par lampes ;
- mémoire séparée (64 valeurs de 22 bits).

Capable de réaliser cinq opérations ($+$, $-$, $\times$, $\div$, $\sqrt{}$)
à raison d'une opération par seconde environ
et peut effectuer des boucles (mais pas de tests)

Évolutions

Les variantes S1 et S2 sont employées dans l'industrie aéronautique.

Le Z4, mis au point difficilement en raison de la situation en Allemagne, est encore plus évolué (plusieurs dizaines d'opérations).

Les machines de Zune pèsent plus d'une tonne et consomment des milliers de watts.

Autres contributions de Konrad Zuse

Konrad Zuse propose également le *Plankalkül*

premier *langage de programmation de haut niveau*

avec typage, affectation, opérations conditionnelles, boucles, sous-programmes, opérations logiques, manipulation de listes, exceptions arithmétiques, etc.

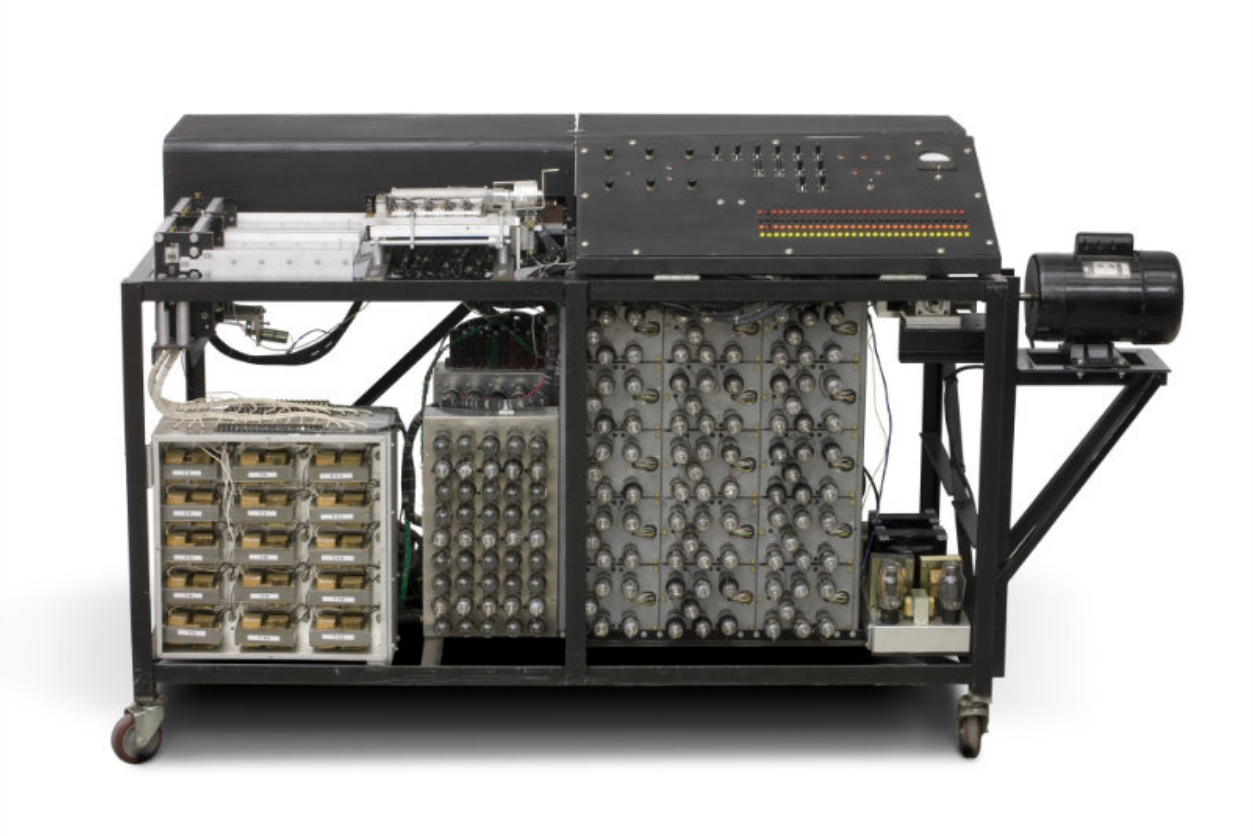
Il écrit quantité d'algorithmes (tris, recherches, etc.)

Il étudie la possibilité d'algorithmes pour les échecs et prédit que 50 ans plus tard, un ordinateur battra le champion du monde d'échecs.

L'Atanasoff-Berry Computer

Imaginé en 1937 par John Vincent Atanasoff et Clifford Berry.

Réalisé et testé en 1942.



Capable de calculer en binaire, mais non programmable.

Colossus de T. Flowers, M. Von Neumann et A. Turing

Destiné à casser le code de cryptage *Lorentz*.

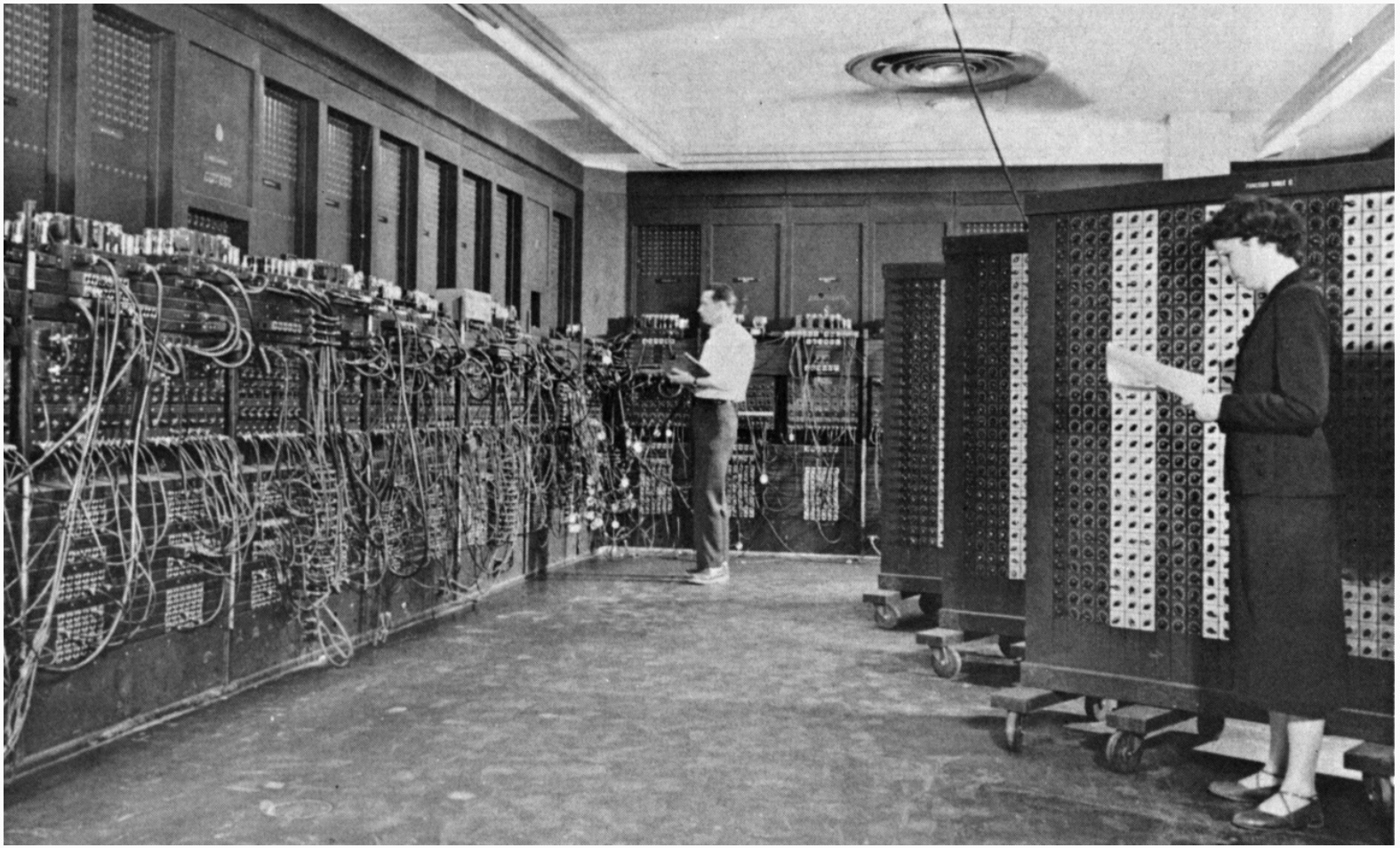
- entièrement électronique (1500 à 2400 tubes à vide)
- fonctionne en binaire
- totalement reprogrammable
- environ 5000 opérations par seconde

Inconnu à l'époque car top-secret, les dix exemplaires furent détruits entre 1945 et 1960 par l'armée britannique.

Quelques plans illégalement conservés ont récemment permis de l'étudier en partie.

Electronic Numerical Integrator Analyser and Computer

Projet lancé en 1943 par le physicien John William Mauchly.



Electronic Numerical Integrator Analyser and Computer

Construit initialement pour le calcul de tables de tir.

- complètement électronique (tubes à vide)
- travaille en décimal
- nécessite un recablage pour la programmation.

Peu innovant mais très rapide pour l'époque
(10^5 additions par seconde).

Beaucoup de pannes (lampes, premiers bugs) !

Fait la une de Time et Newsweek.

Développement de l'offre commerciale

- 1946 : Univac, premier ordinateur commercial
  (Utilisé pour le recensement aux USA)
- 1947 : William Shockley, John Bardeen et Walter Brattain
  (Bell Laboratories) inventent le transistor
- 1958 : Jack Kilby (Nobel 2000) et Robert Noyce
  introduisent le circuit intégré
- 1971 : Intel 4004, premier microprocesseur commercial
  (=ENIAC)
- 1974-1976 : premiers ordinateurs personnels
  (IBM 5100, TRS-80, Commodore PET, Apple-I)

Côté logiciel et interface

- 1964 : Douglas Engelbart présente le principe de la souris et de l'interface graphique
- 1969 : création de UNIX chez Bell Laboratories
- 1971 : Alan Shugart (IBM) crée la disquette
- 1978 : premier tableur (Visicalc)
- 1979 : premier traitement de texte (WordStar)
- 1981 : introduction de DOS (pour les IBM PC)
- 1984 : débuts de Mac OS
- 1991 : conception de Linux par Linus Torvalds

Apparition des réseaux

- 1973 : Robert Kahn and Vinton Cerf introduisent le TCP
- 1973 : Robert Metcalfe (Xerox) conçoit le réseau Ethernet
- 1989 : Tim Berners-Lee (CERN) développe l'HyperText Markup Language
- 1990 : premier serveur web, premier navigateur, premier site
- 1991 : le *world wide web* devient publiquement accessible

Machine de Max Von Neumann

C'est une des premières description des constituants d'un ordinateur.

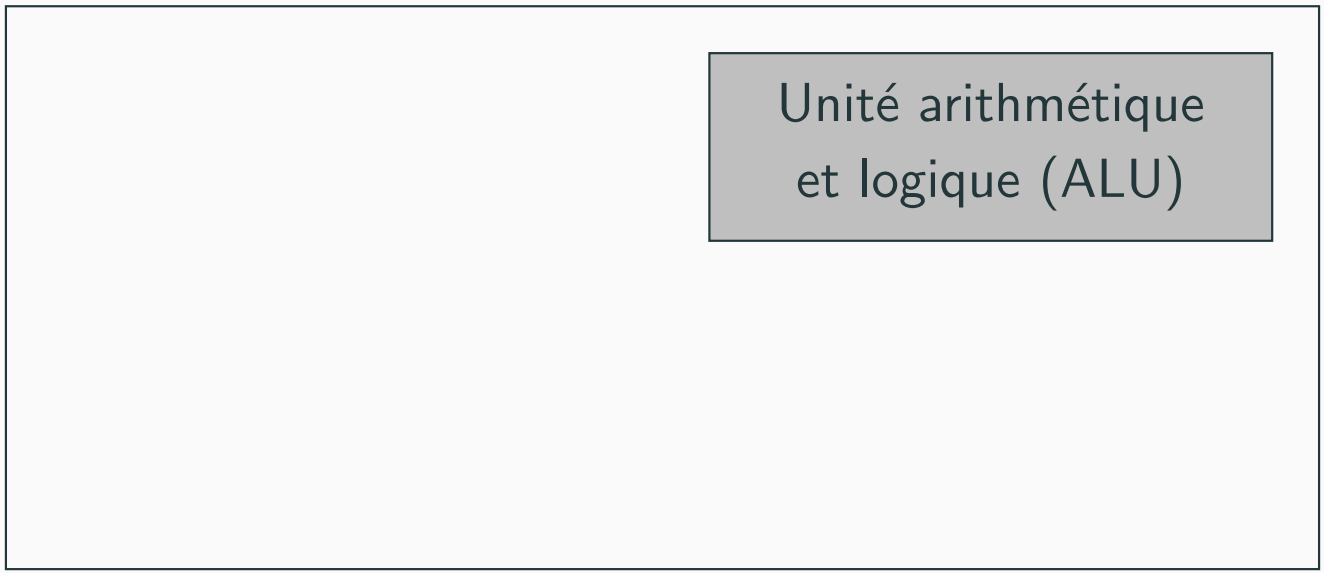
Des dires de Max Von Neumann, elle serait en partie inspirée d'idées d'Alan Turing.

La machine regroupe différents éléments.

Machine de Max Von Neumann

C'est une des premières description des constituants d'un ordinateur.

Des dires de Max Von Neumann, elle serait en partie inspirée d'idées d'Alan Turing.

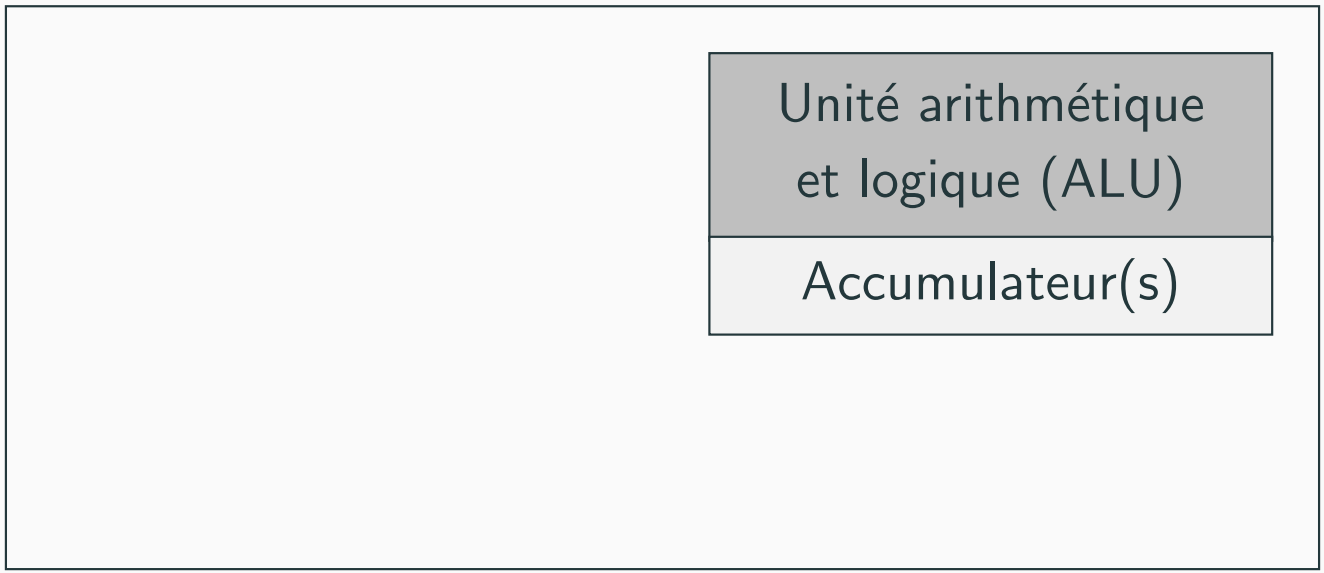


En premier lieu, une unité capable d'effectuer des calculs.

Machine de Max Von Neumann

C'est une des premières description des constituants d'un ordinateur.

Des dires de Max Von Neumann, elle serait en partie inspirée d'idées d'Alan Turing.

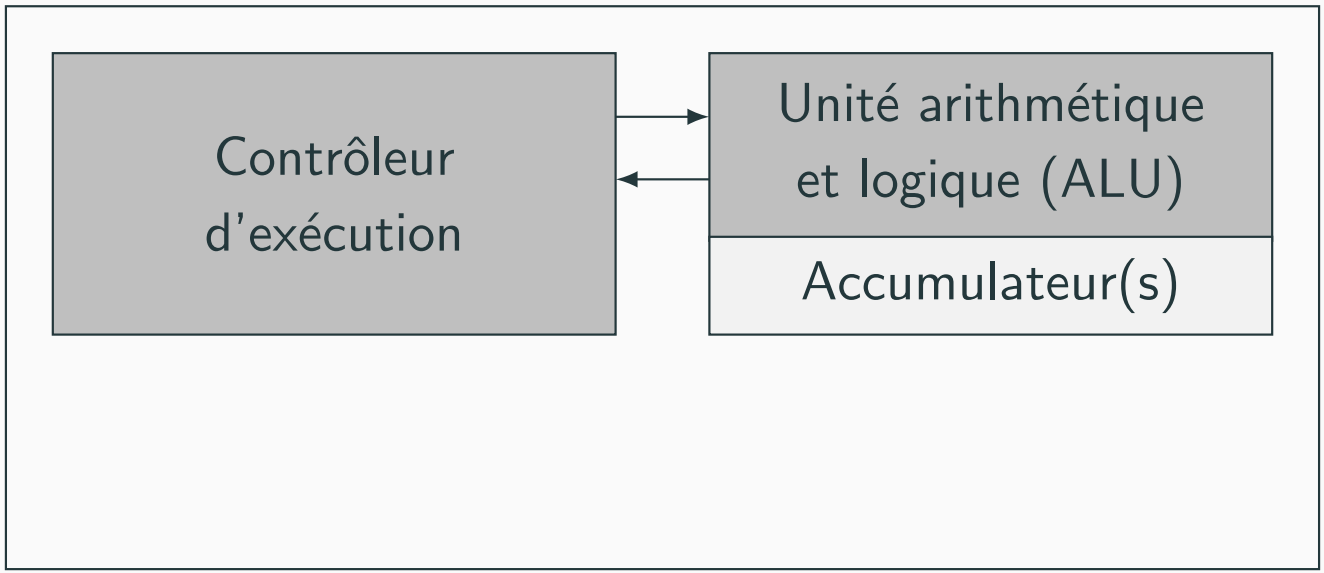


Des accumumateurs (registres) accueillent arguments et résultats.

Machine de Max Von Neumann

C'est une des premières description des constituants d'un ordinateur.

Des dires de Max Von Neumann, elle serait en partie inspirée d'idées d'Alan Turing.

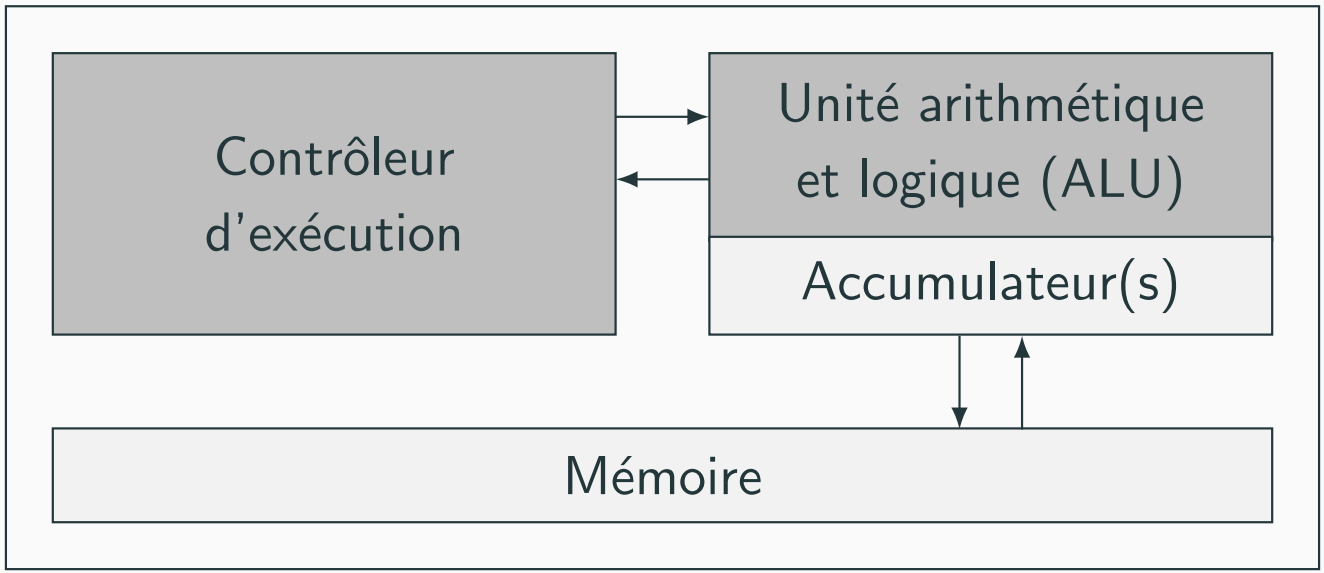


Une unité de contrôle détermine quelles opérations effectuer.

Machine de Max Von Neumann

C'est une des premières description des constituants d'un ordinateur.

Des dires de Max Von Neumann, elle serait en partie inspirée d'idées d'Alan Turing.

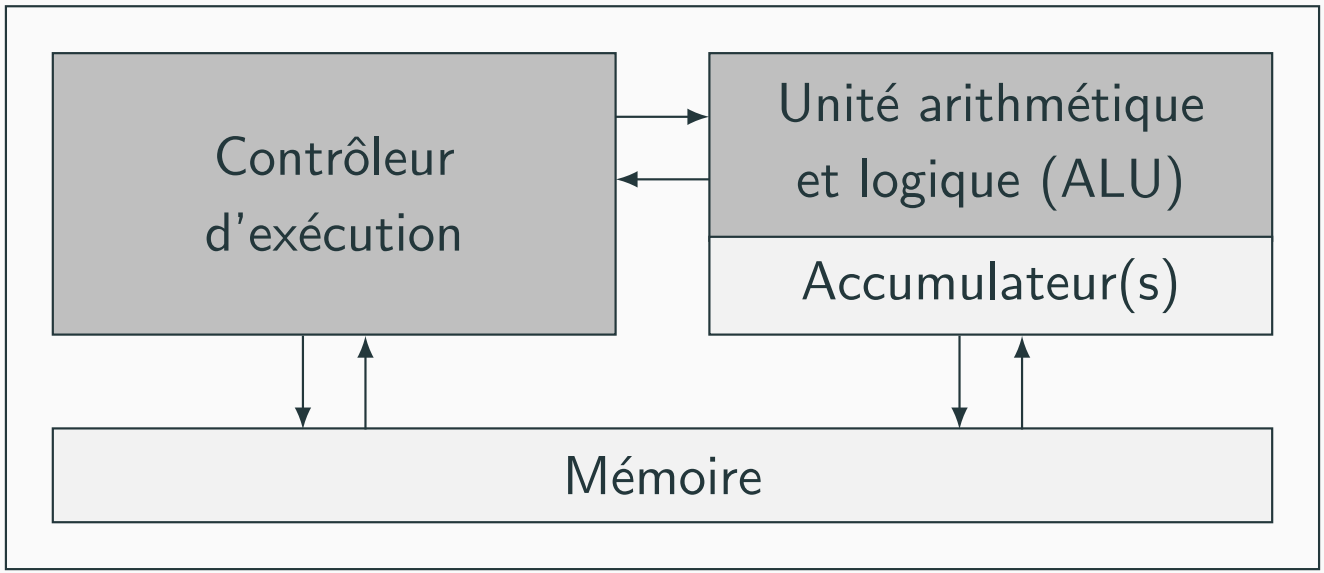


Une mémoire permet de manipuler davantage de données.

Machine de Max Von Neumann

C'est une des premières description des constituants d'un ordinateur.

Des dires de Max Von Neumann, elle serait en partie inspirée d'idées d'Alan Turing.

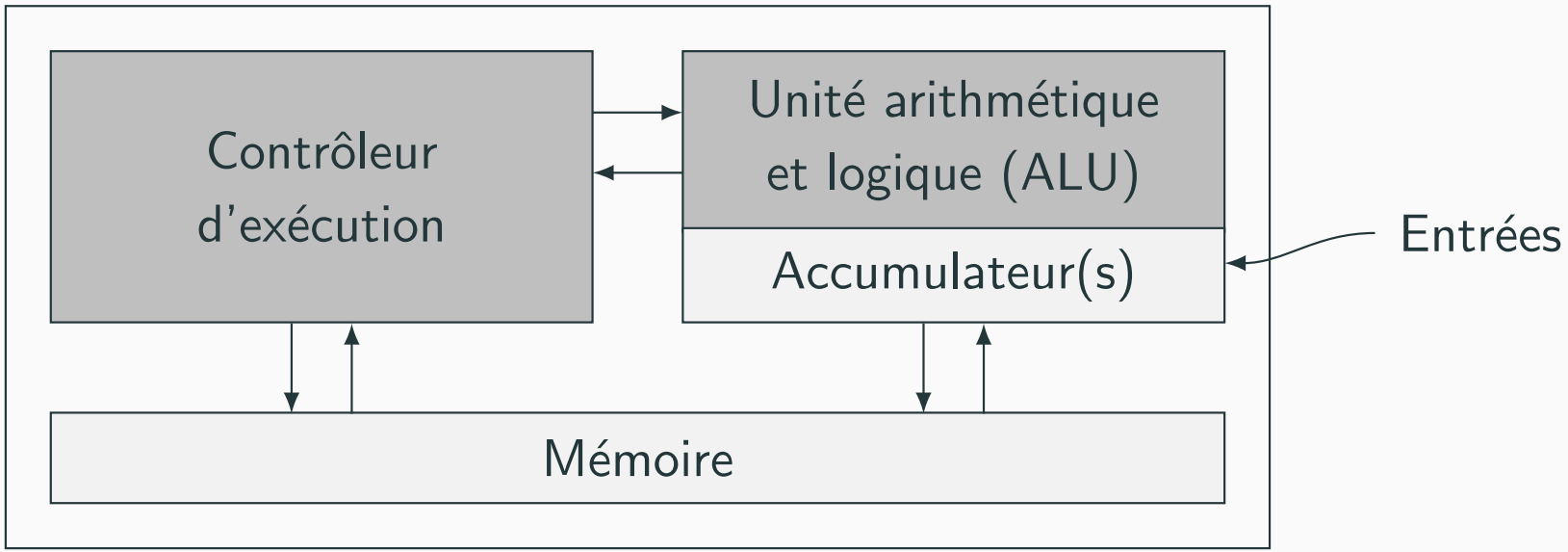


Le programme (modifiable) exécuté s'y trouve également.

Machine de Max Von Neumann

C'est une des premières description des constituants d'un ordinateur.

Des dires de Max Von Neumann, elle serait en partie inspirée d'idées d'Alan Turing.

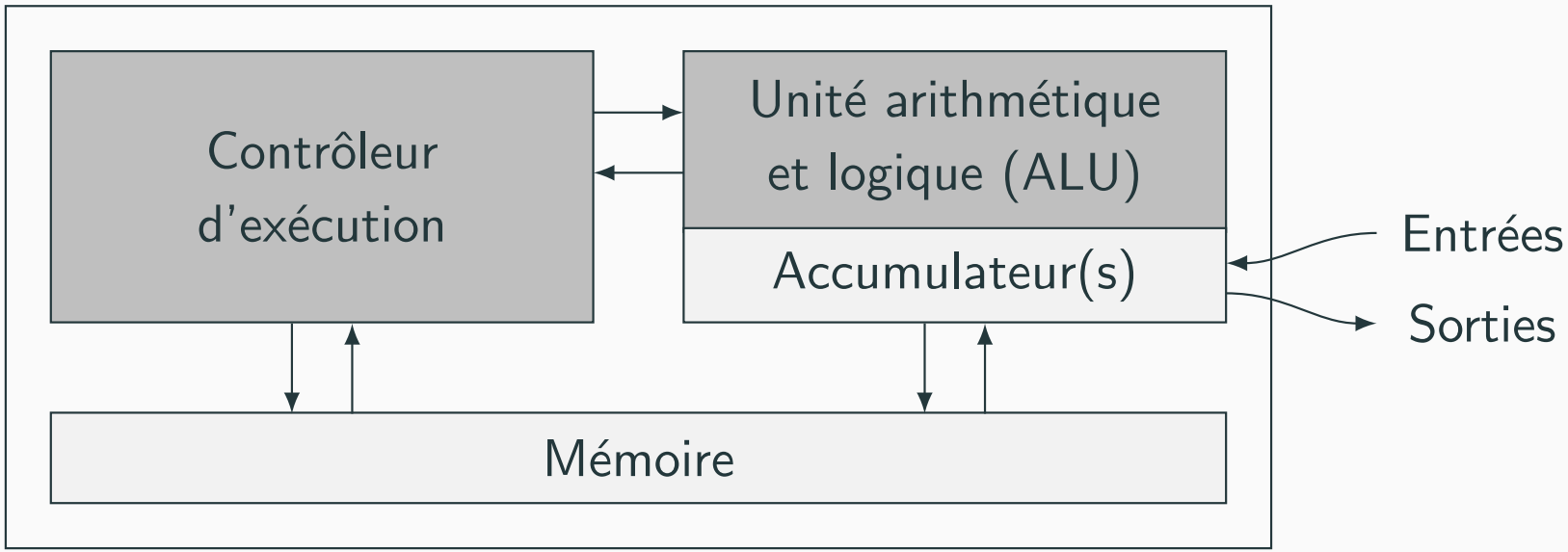


La machine peut accepter des informations extérieures.

Machine de Max Von Neumann

C'est une des premières description des constituants d'un ordinateur.

Des dires de Max Von Neumann, elle serait en partie inspirée d'idées d'Alan Turing.



Et également transmettre ses résultats.

Machine de Max Von Neumann

Quasi-identique au modèle proposé par Konrad Zuse avant guerre

Principale différence : le programme, en mémoire, modifiable

(K. Zuse était conscient de l'intérêt, mais ses machines ne disposaient pas assez de mémoire pour cela)

Cette possibilité n'est que rarement utilisée actuellement

(par les compilateurs, les interpréteurs... et les hackers !)

C'est toujours la structure des ordinateurs (et microcontrôleurs) actuels.

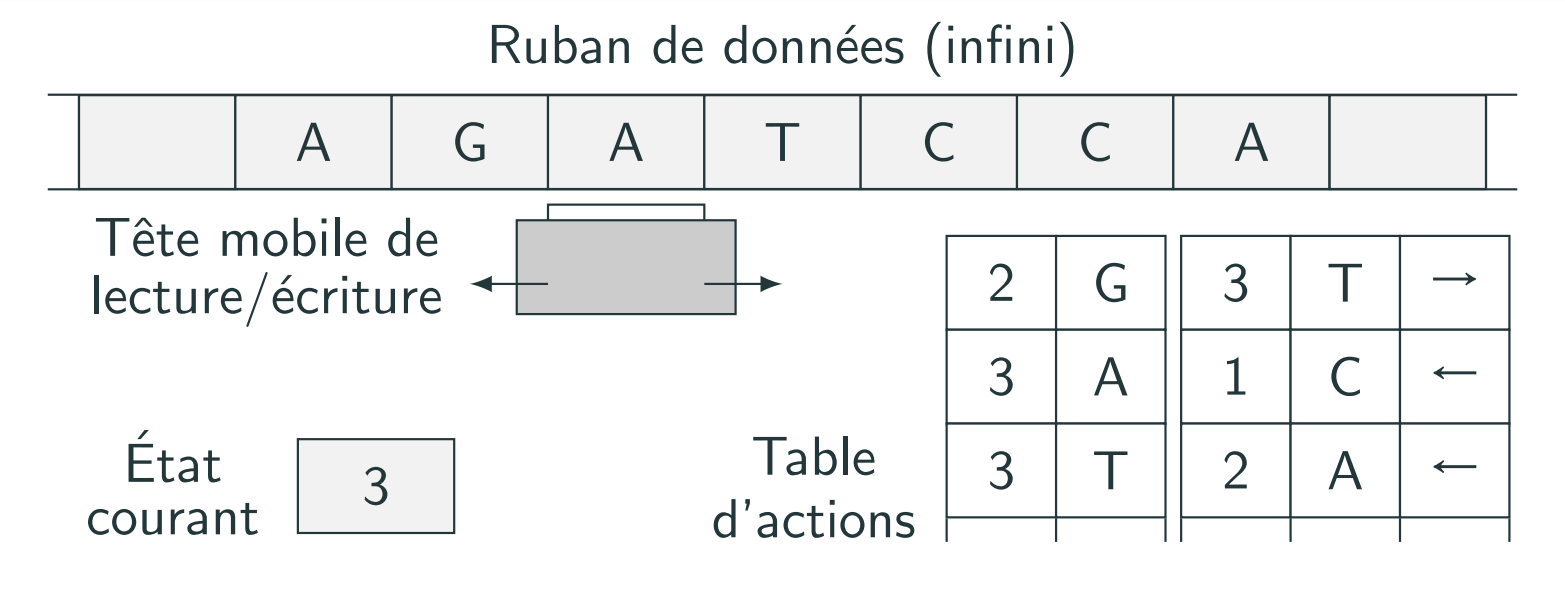
Machine de Turing

Modèle *abstrait* d'un ordinateur et sa mémoire

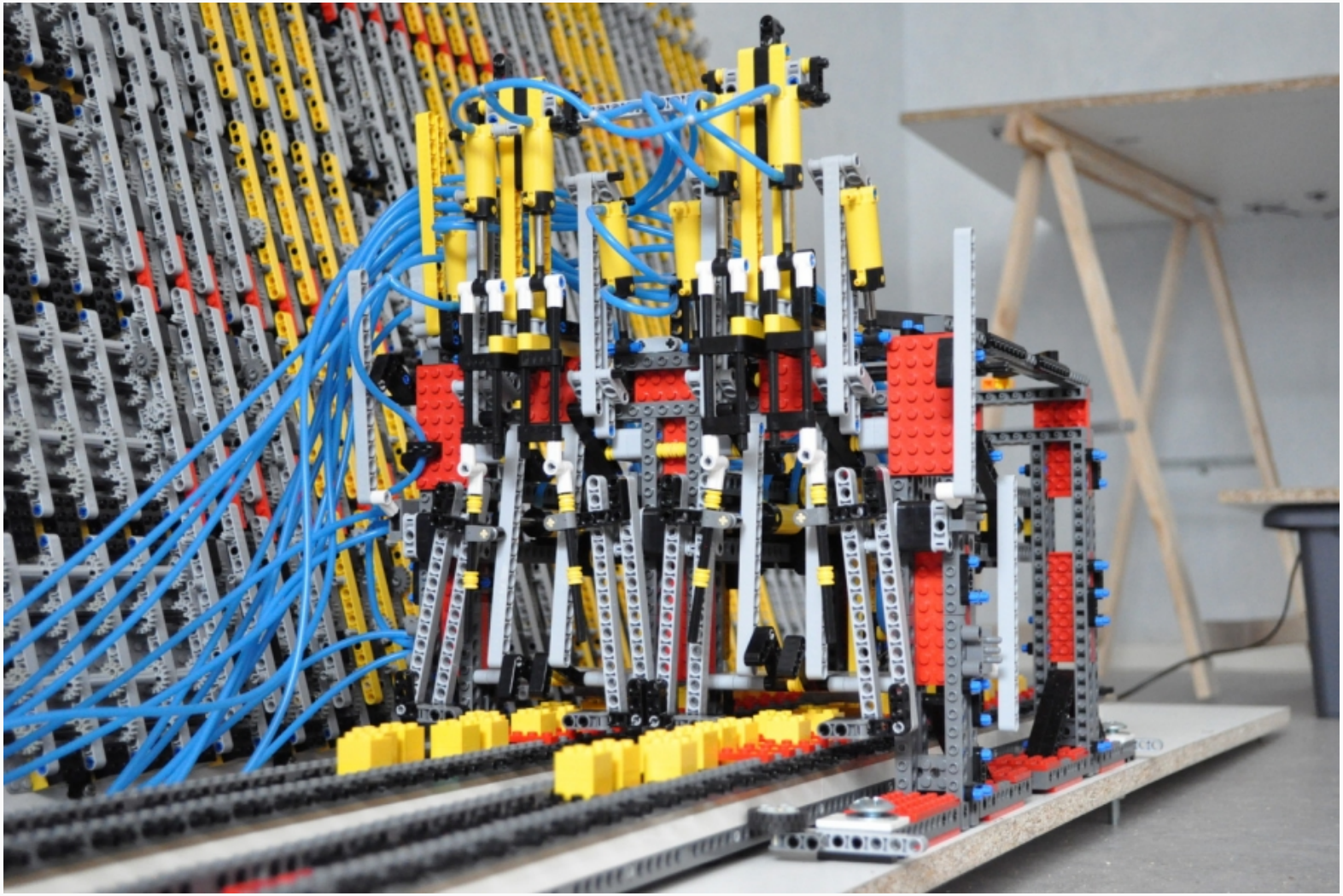
Introduit par Alan Turing en 1936

Utilisé pour étudier algorithmes, complexité, calculabilité

Toujours un fondement de l'informatique théorique



En Lego !



Réalisation ENS Lyon (projet Rubens)

Machine de Turing

On peut concevoir une machine de Turing pour appliquer un algorithme ou un mécanisme de calcul quelconque

A. Turing : on peut créer une machine de Turing *universelle*

capable de simuler le fonctionnement de n'importe quelle machine de Turing

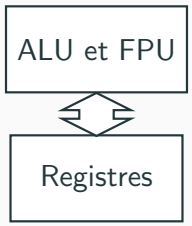
Dite *Turing-complète*, elle est en mesure de calculer tout ce qui est calculable, les problèmes qu'elles peut résoudre sont exactement les problèmes résolubles par un algorithme

En dehors de la condition de mémoire infinie, les ordinateurs (et la plupart des langages de programmation) sont généralement capables d'émuler une machine de Turing universelle, et donc de résoudre n'importe quel problème

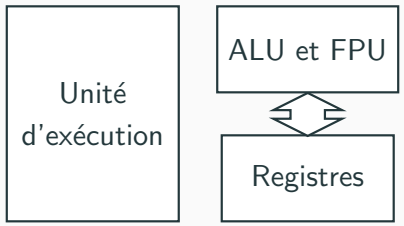
Structure (simplifiée) d'un ordinateur

Structure (simplifiée) d'un ordinateur

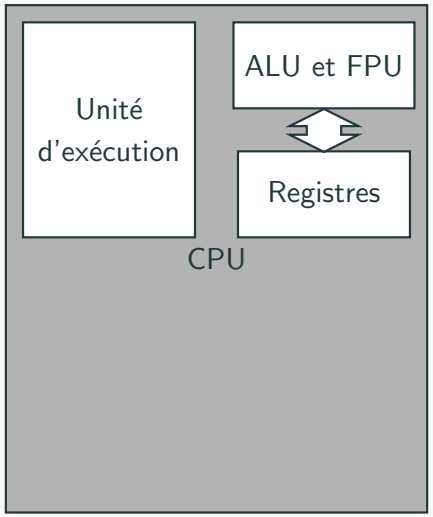
Structure (simplifiée) d'un ordinateur



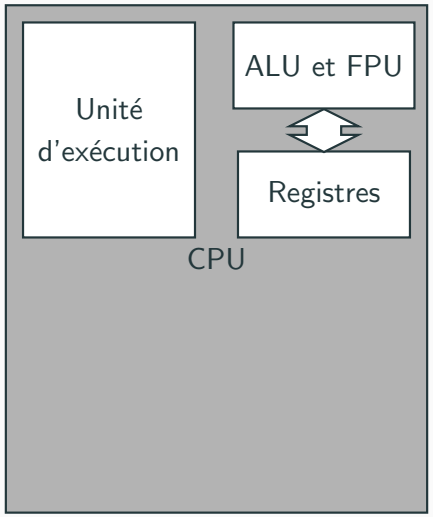
Structure (simplifiée) d'un ordinateur



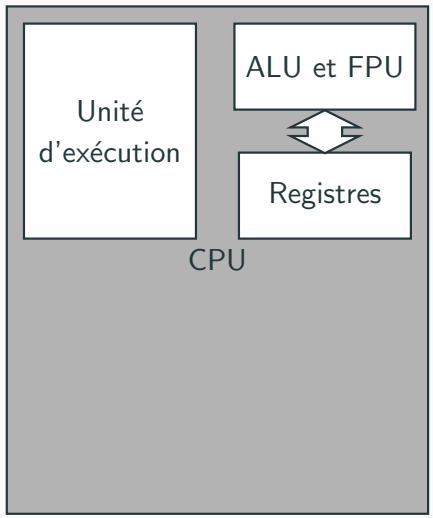
Structure (simplifiée) d'un ordinateur



Structure (simplifiée) d'un ordinateur



Structure (simplifiée) d'un ordinateur



RAM

Écran/moniteur

Réseau filaire

Antenne WiFi

Disque dur

Lecteur optique

Imprimante

Souris

Clavier

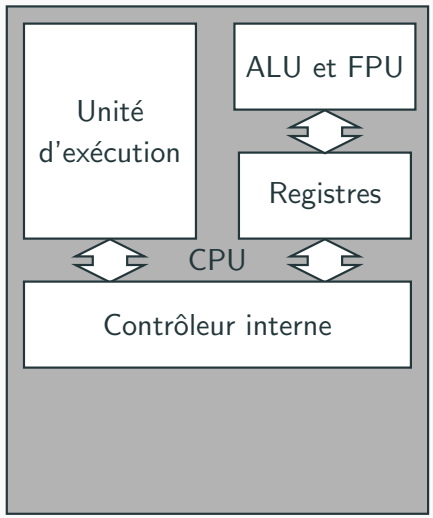
Clé USB

Webcam...

Casque

Micro

Structure (simplifiée) d'un ordinateur



RAM

Écran/moniteur

Réseau filaire

Antenne WiFi

Disque dur

Lecteur optique

Imprimante

Souris

Clavier

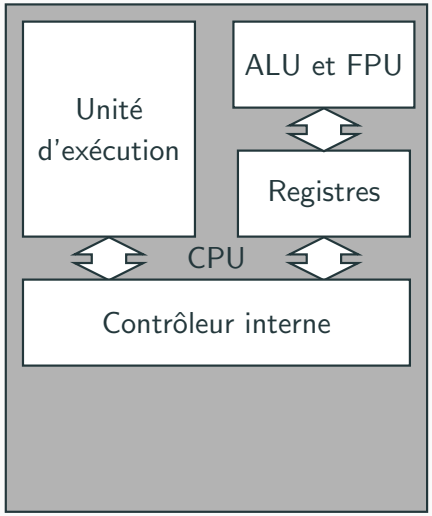
Clé USB

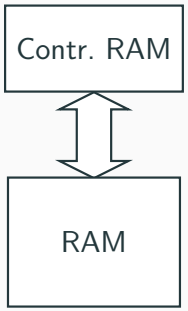
Webcam...

Casque

Micro

Structure (simplifiée) d'un ordinateur





Écran/moniteur

Réseau filaire

Antenne WiFi

Disque dur

Lecteur optique

Imprimante

Souris

Clavier

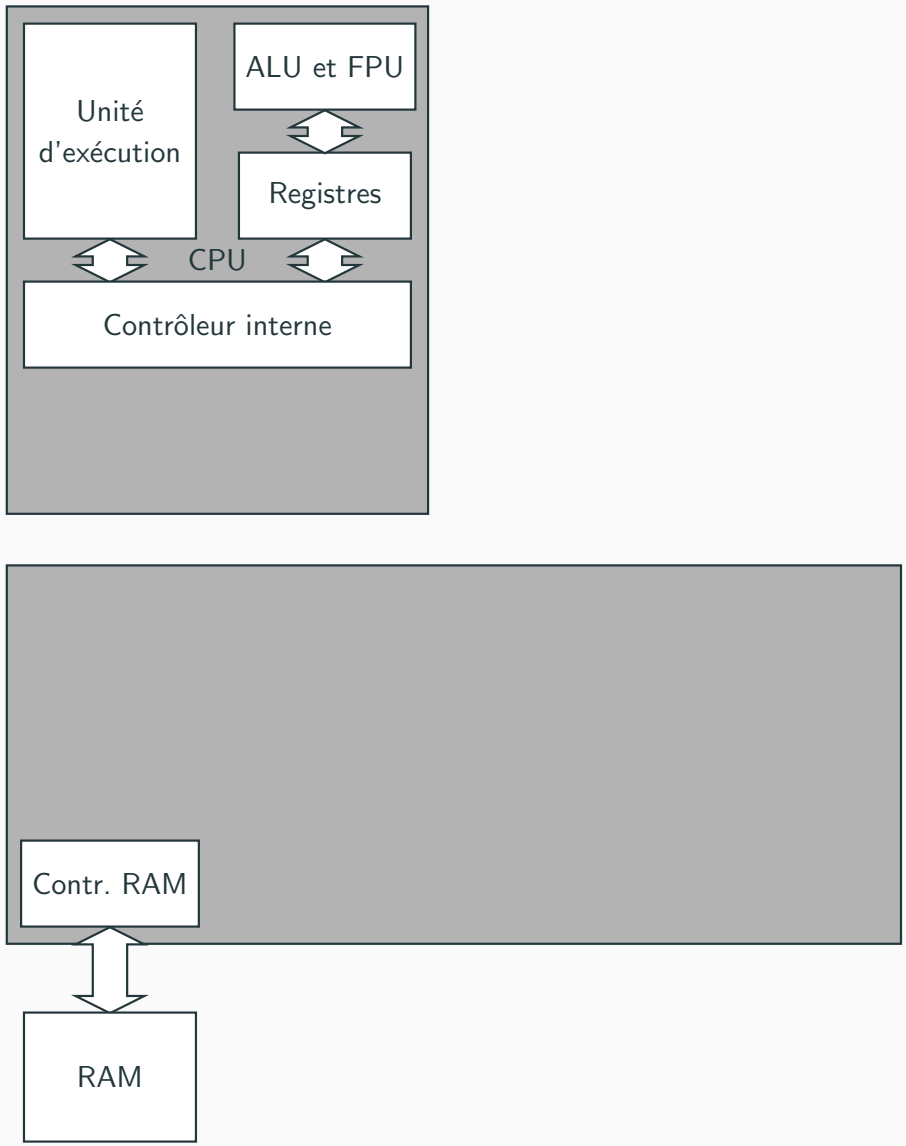
Clé USB

Webcam...

Casque

Micro

Structure (simplifiée) d'un ordinateur



Écran/moniteur

Réseau filaire

Antenne WiFi

Disque dur

Lecteur optique

Imprimante

Souris

Clavier

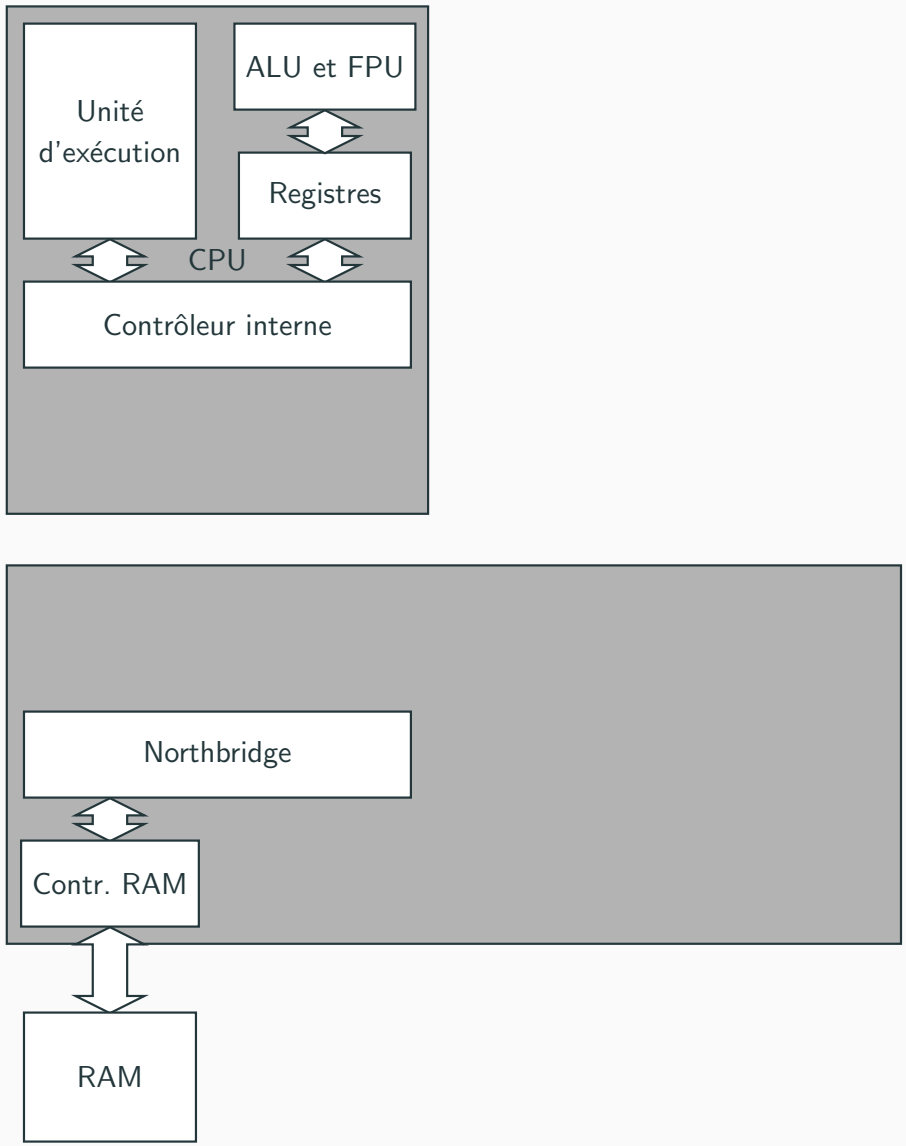
Clé USB

Webcam...

Casque

Micro

Structure (simplifiée) d'un ordinateur



Écran/moniteur

Réseau filaire

Antenne WiFi

Disque dur

Lecteur optique

Imprimante

Souris

Clavier

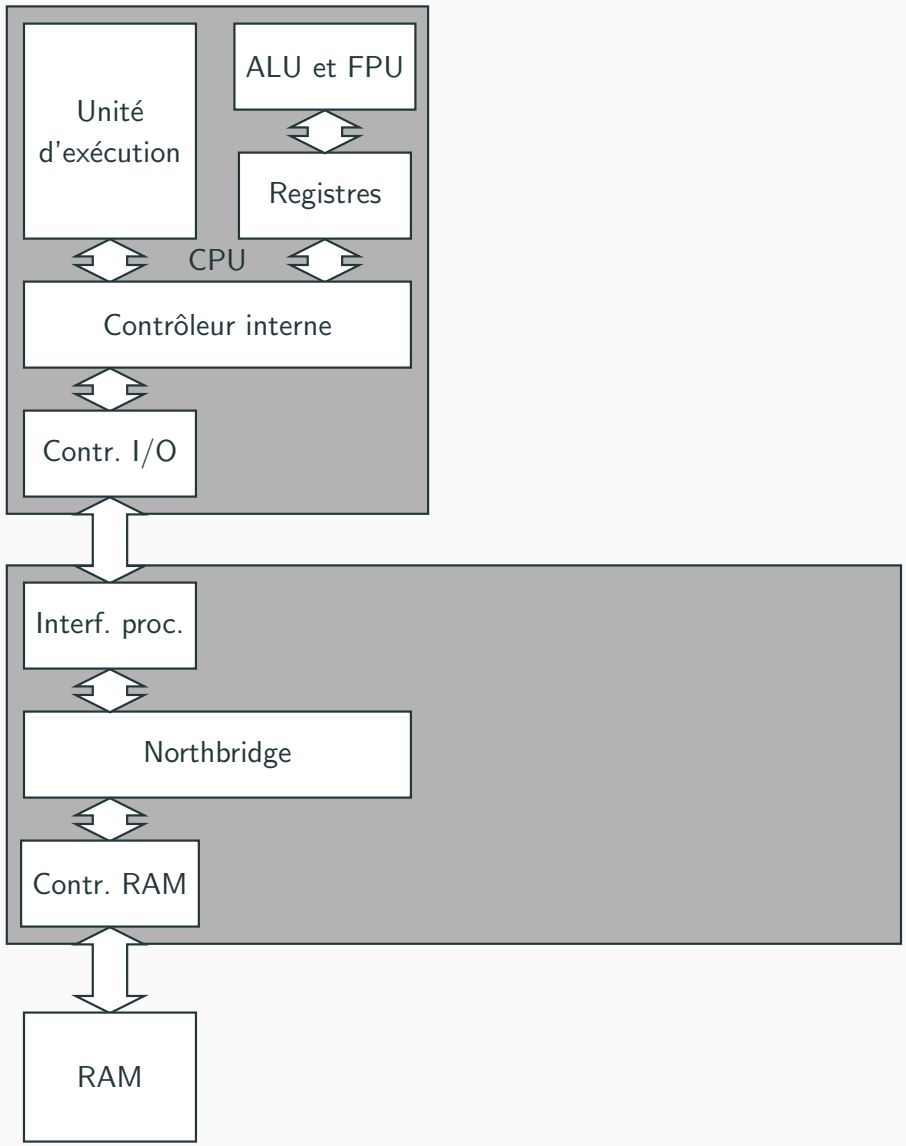
Clé USB

Webcam...

Casque

Micro

Structure (simplifiée) d'un ordinateur



Écran/moniteur

Réseau filaire

Antenne WiFi

Disque dur

Lecteur optique

Imprimante

Souris

Clavier

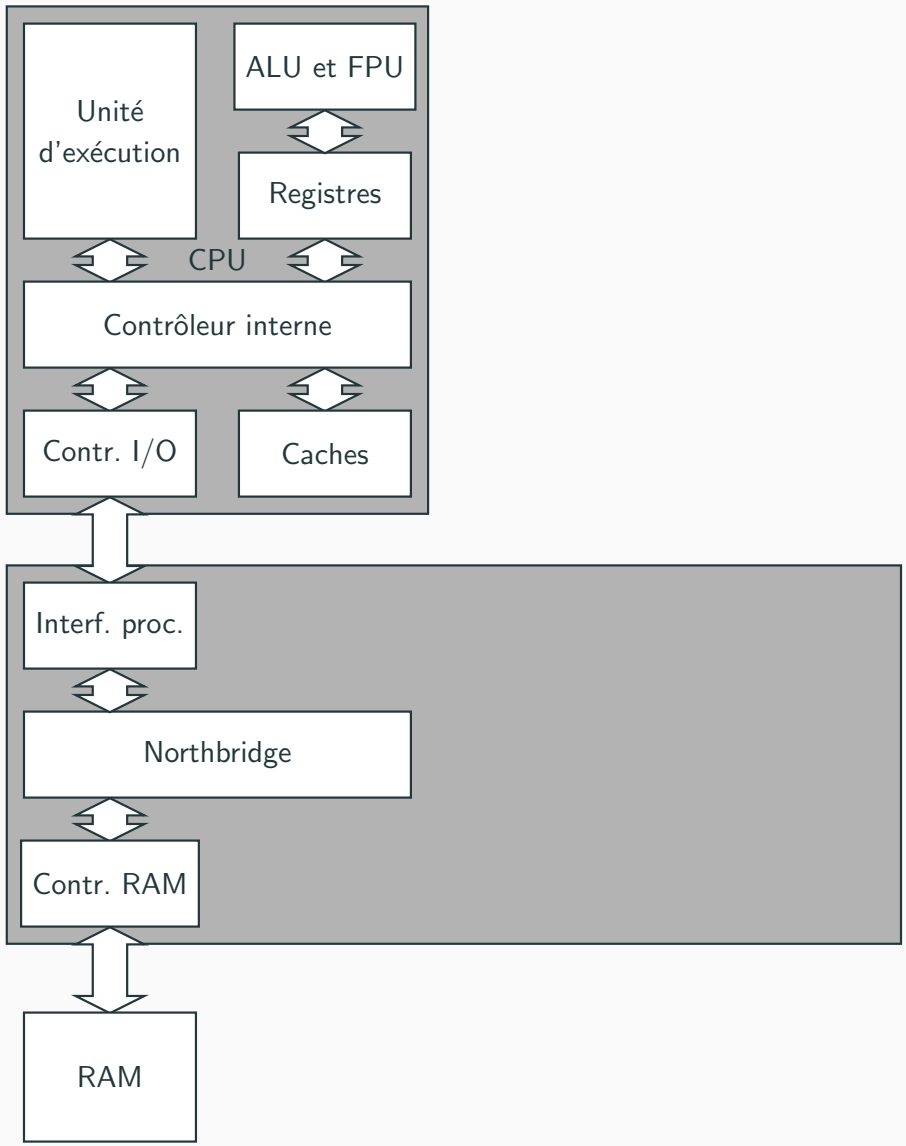
Clé USB

Webcam...

Casque

Micro

Structure (simplifiée) d'un ordinateur



Écran/moniteur

Réseau filaire

Antenne WiFi

Disque dur

Lecteur optique

Imprimante

Souris

Clavier

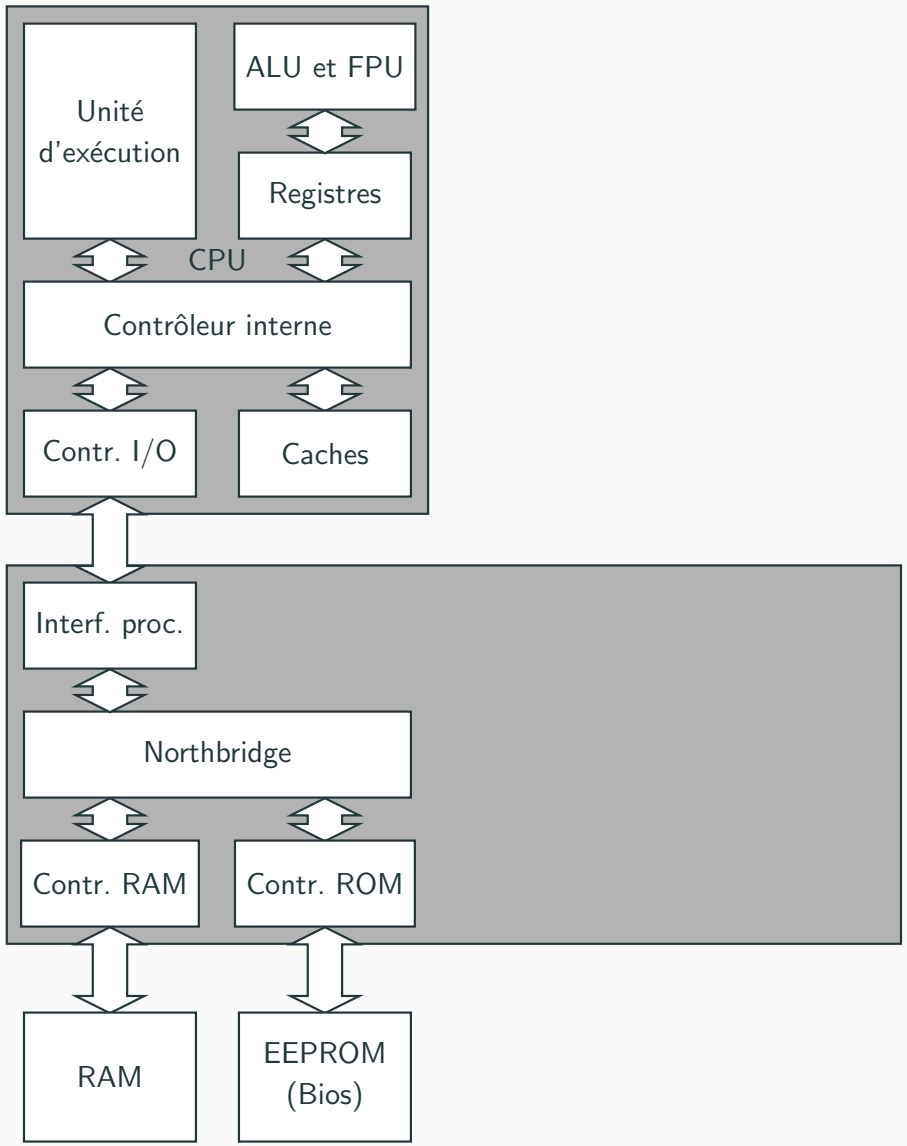
Clé USB

Webcam...

Casque

Micro

Structure (simplifiée) d'un ordinateur



Écran/moniteur

Réseau filaire

Antenne WiFi

Disque dur

Lecteur optique

Imprimante

Souris

Clavier

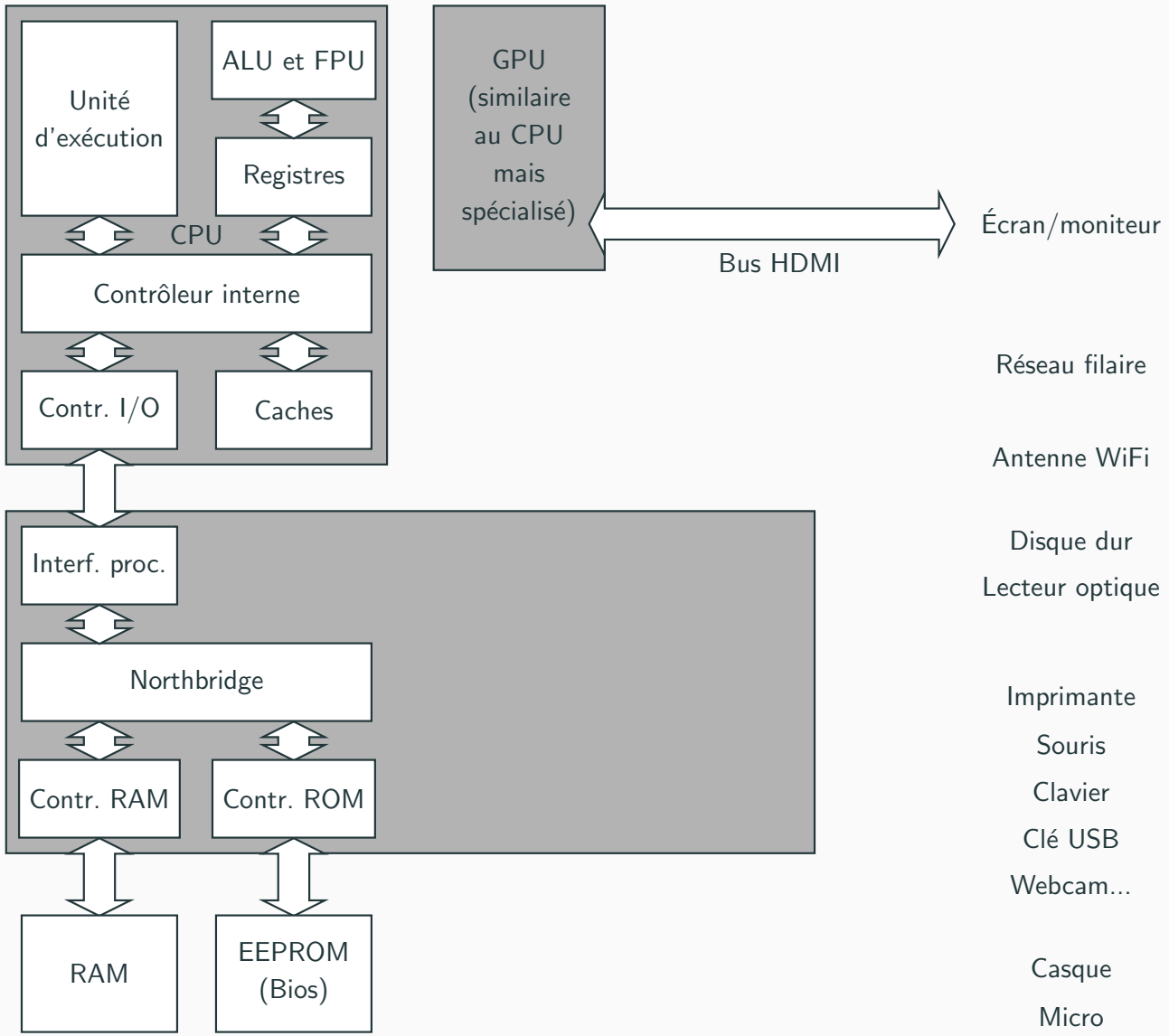
Clé USB

Webcam...

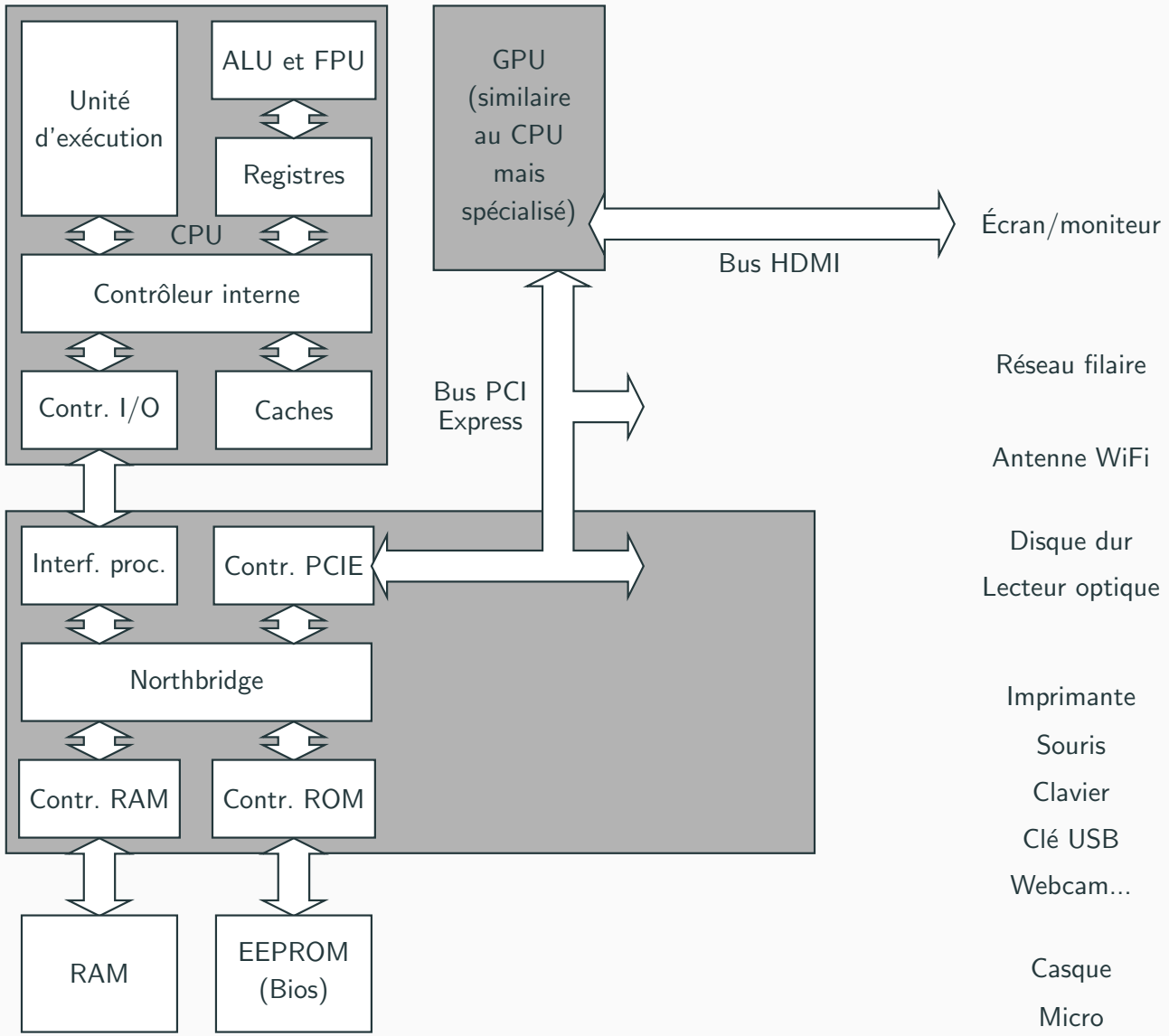
Casque

Micro

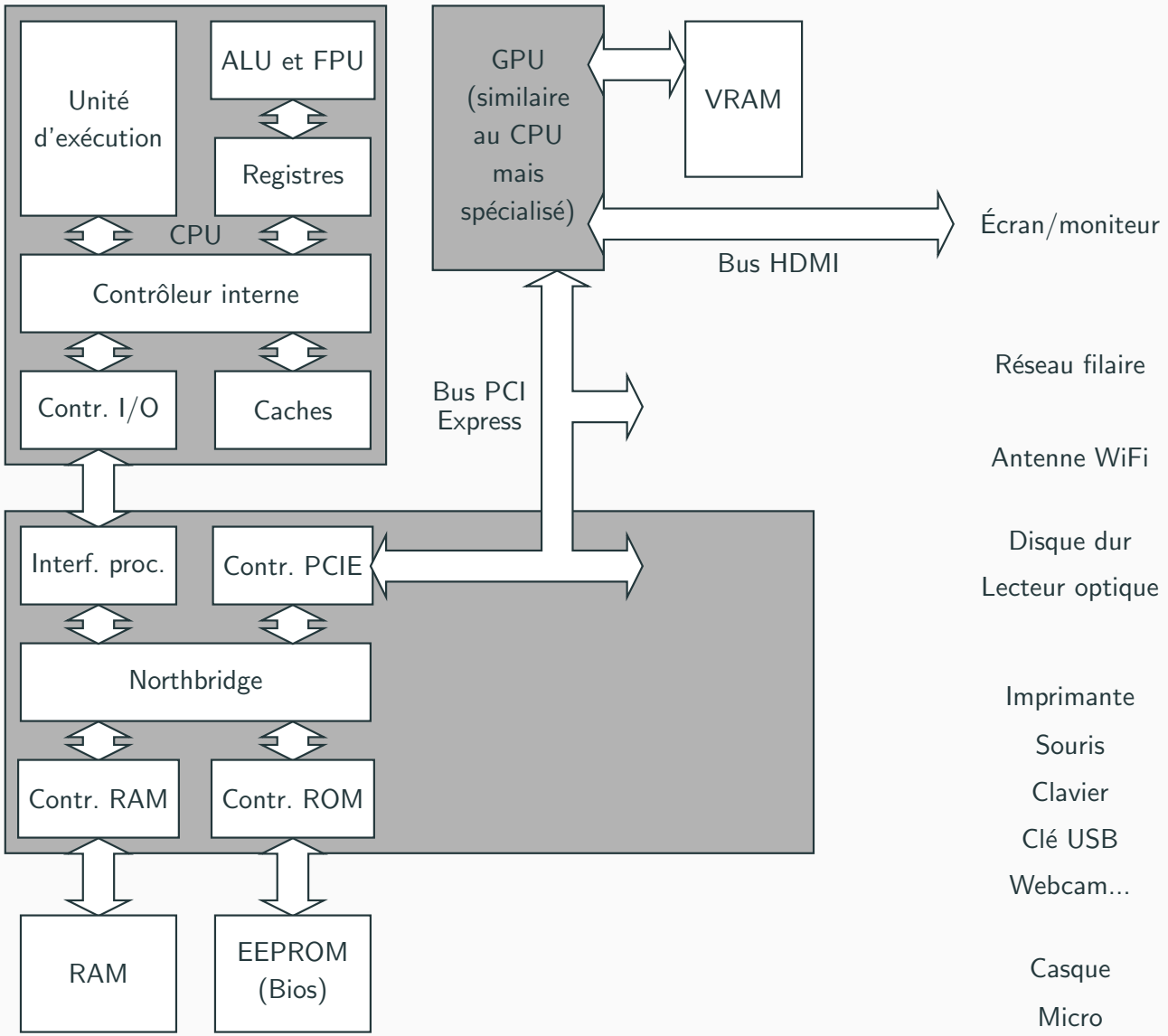
Structure (simplifiée) d'un ordinateur



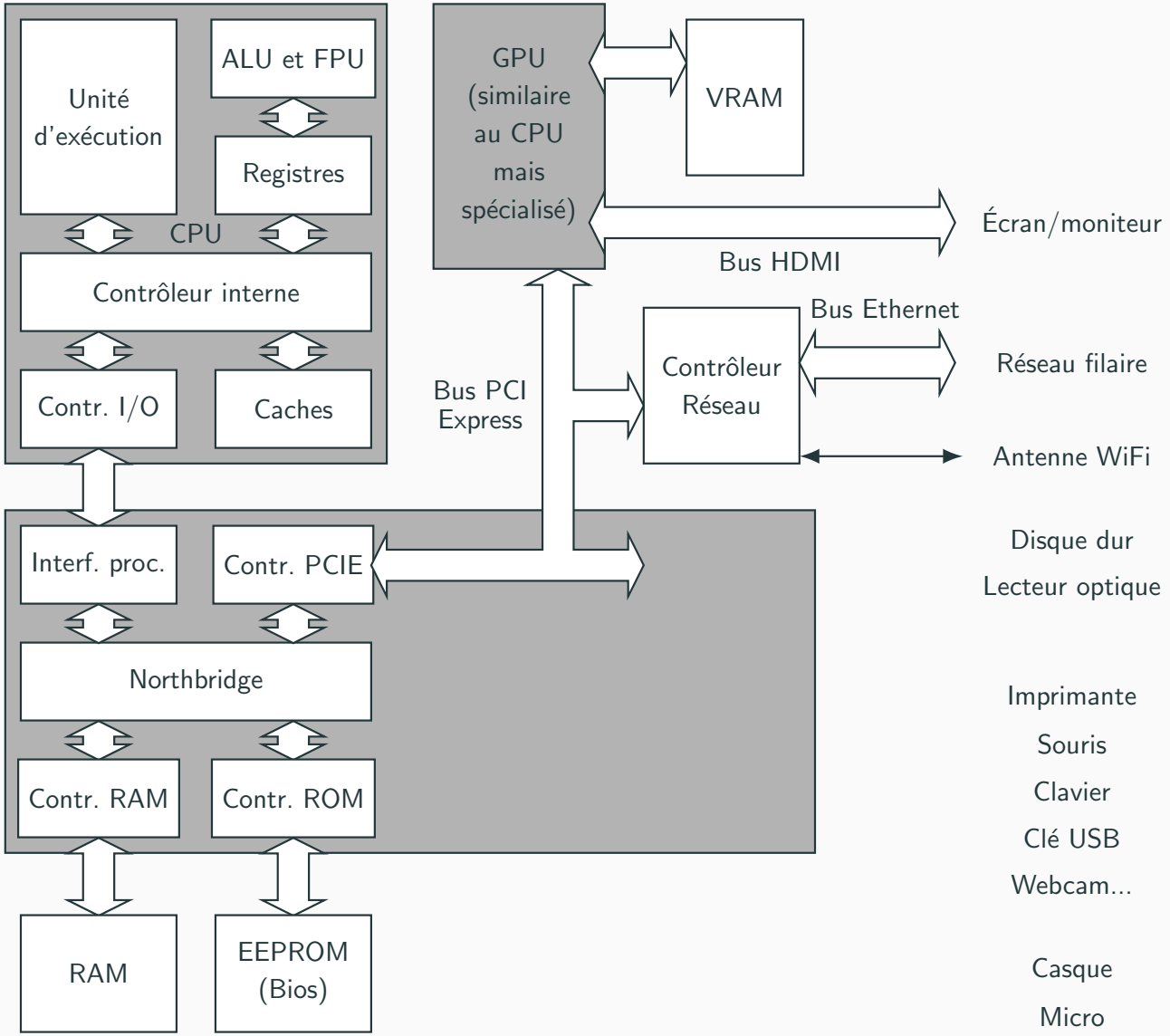
Structure (simplifiée) d'un ordinateur



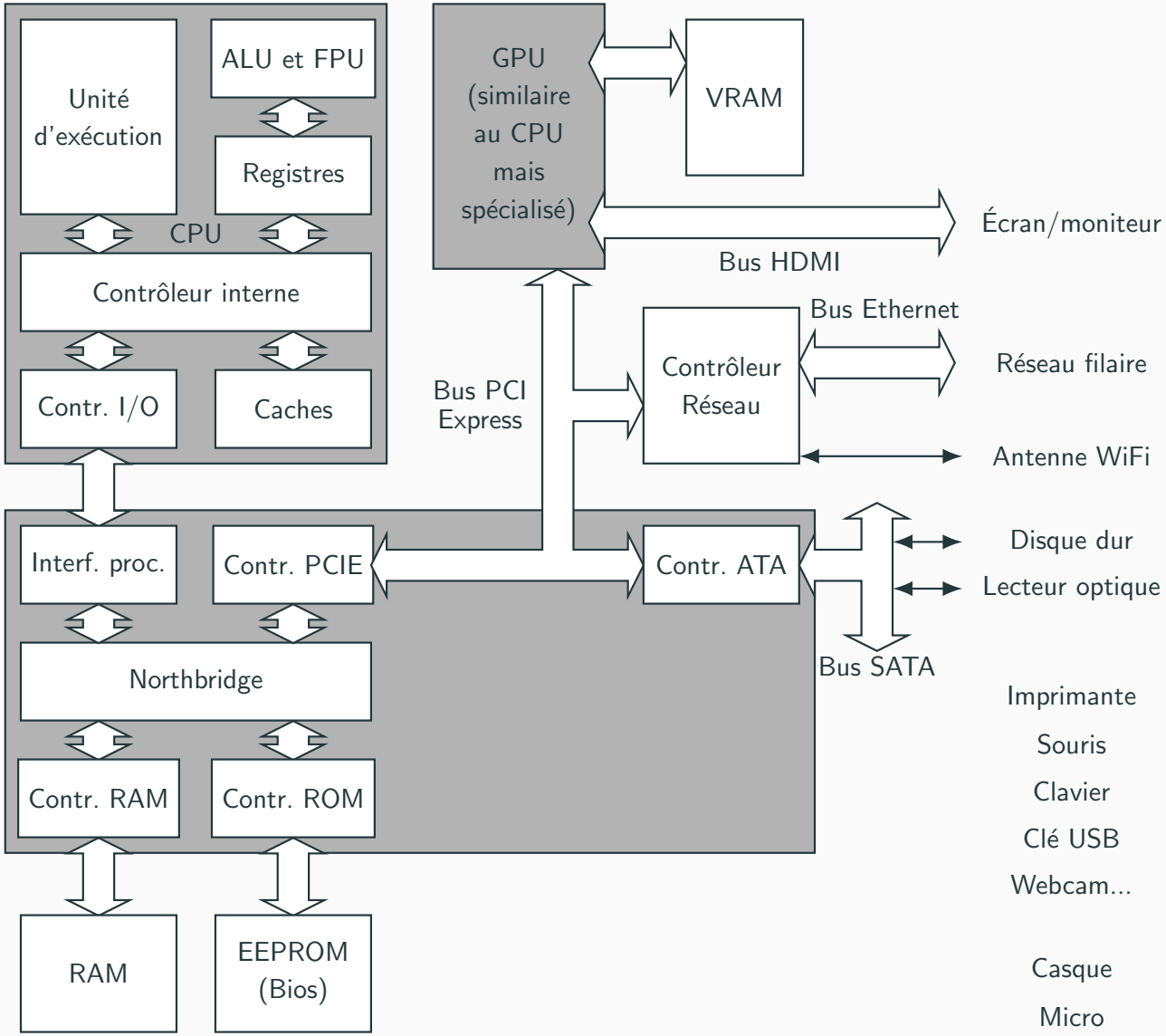
Structure (simplifiée) d'un ordinateur



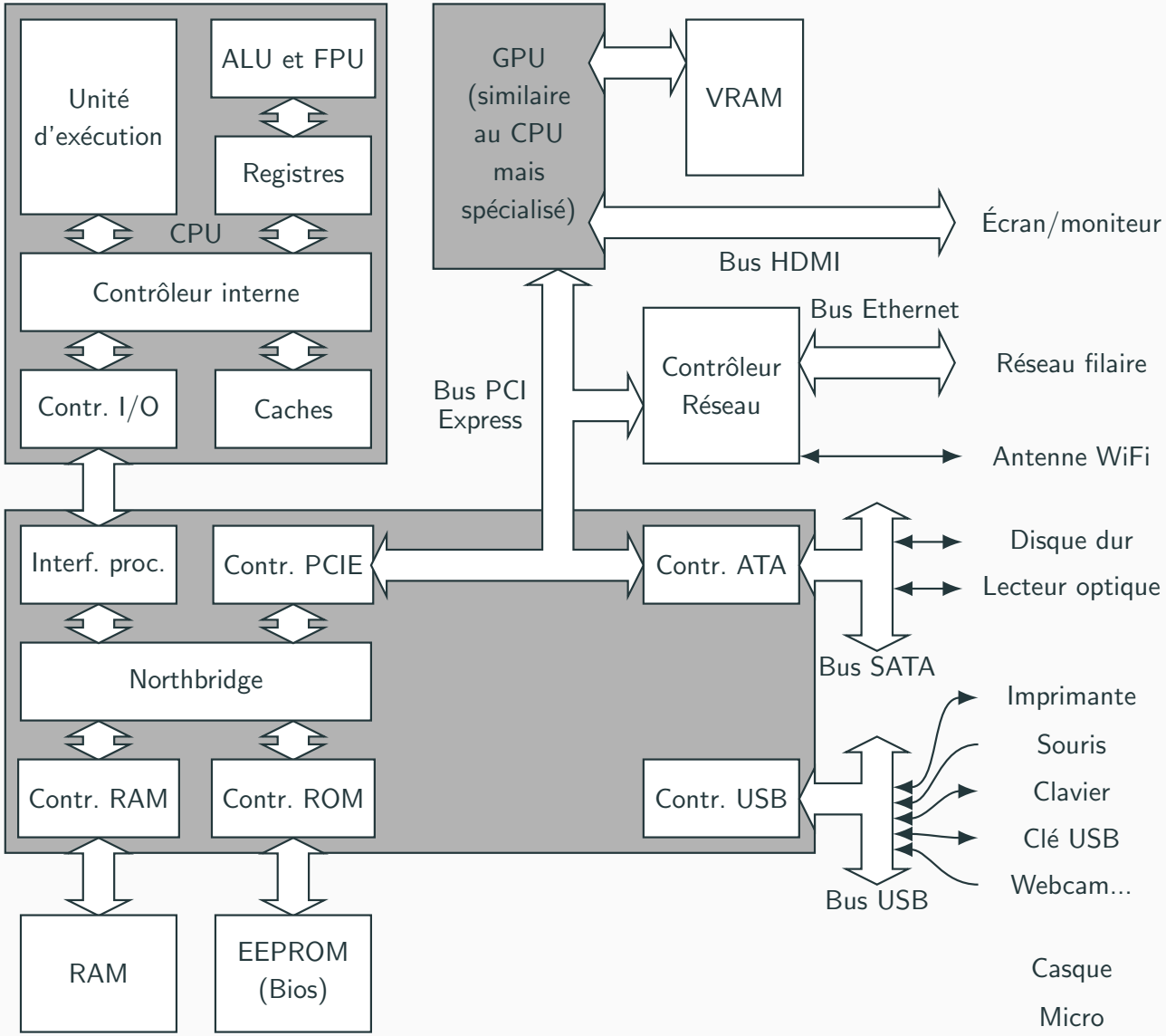
Structure (simplifiée) d'un ordinateur



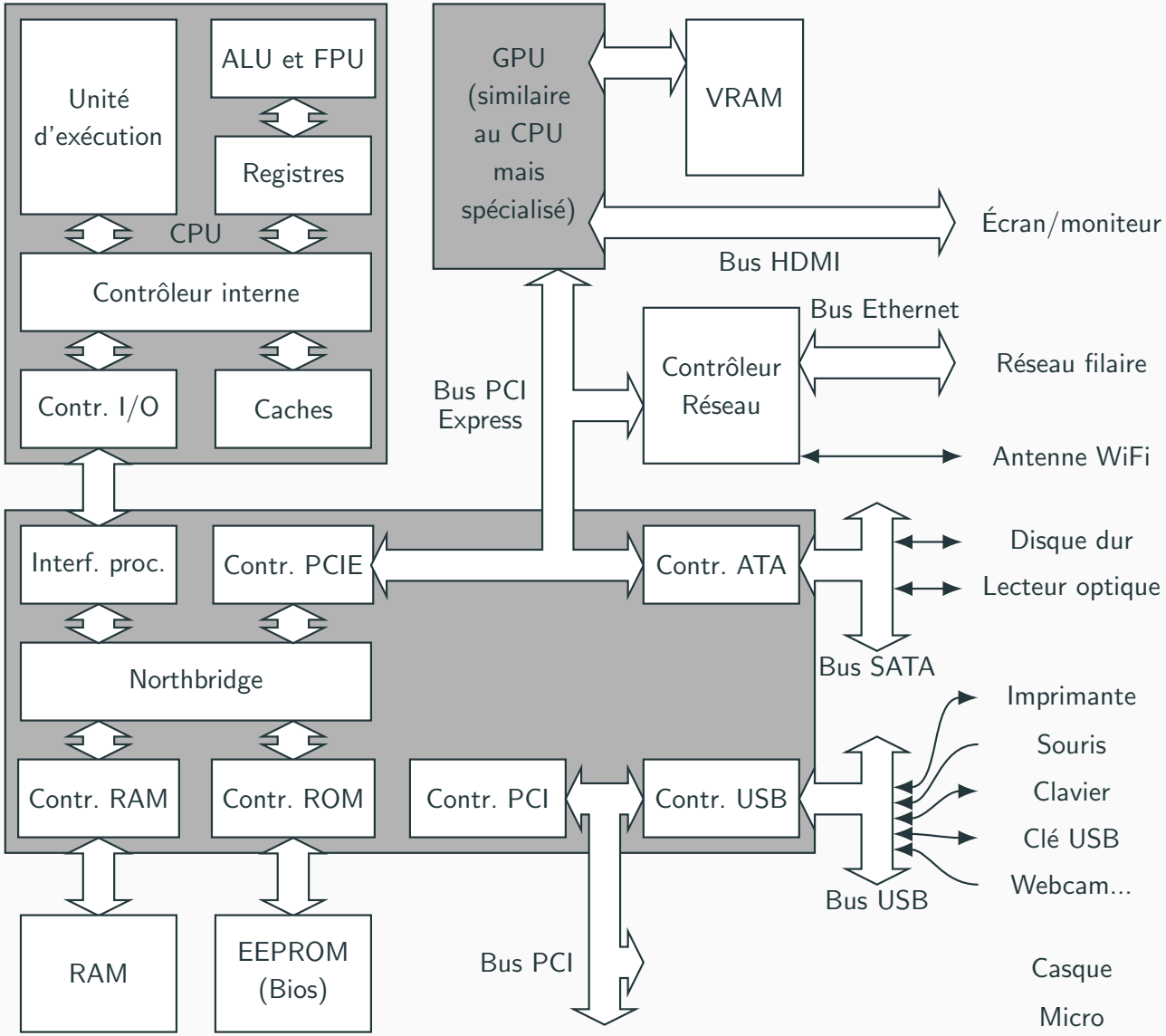
Structure (simplifiée) d'un ordinateur



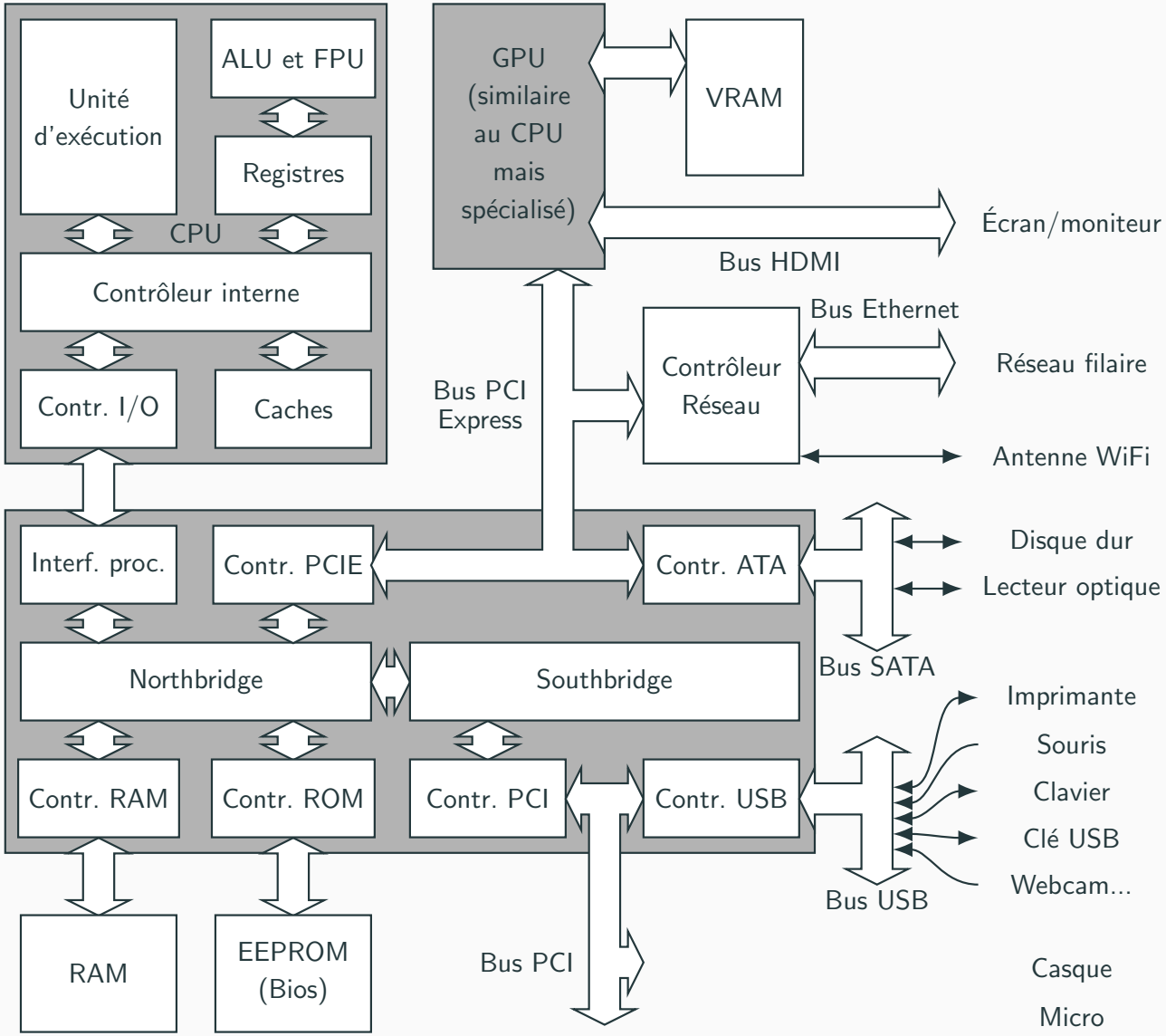
Structure (simplifiée) d'un ordinateur



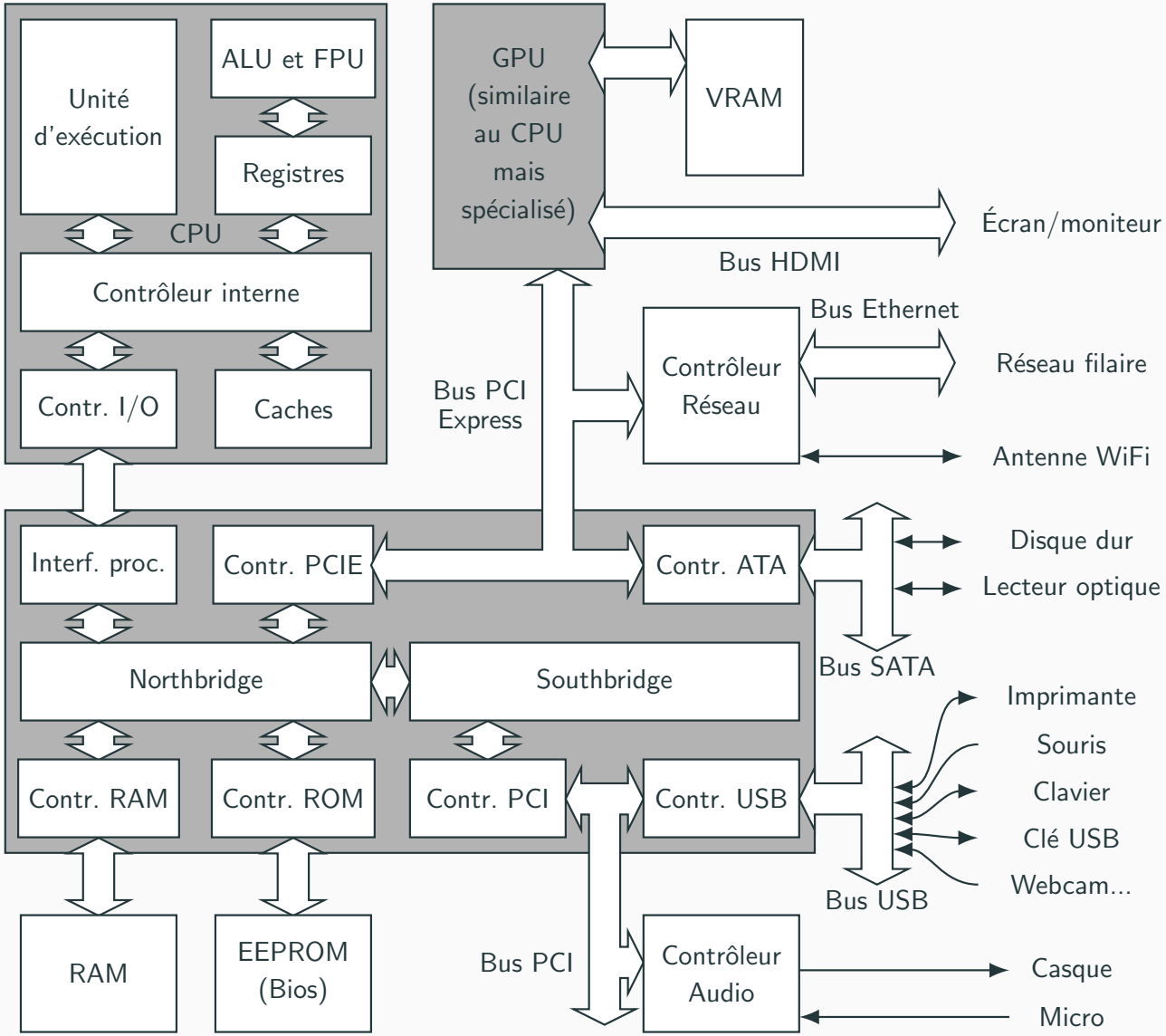
Structure (simplifiée) d'un ordinateur



Structure (simplifiée) d'un ordinateur



Structure (simplifiée) d'un ordinateur



Stocker les données

On dispose de nombreux endroits pour stocker des données :

Type de mémoire	Quantité	Temps d'accès
Registres	$\equiv 10 - 100$	< 1 ns
Cache	$\equiv 10^5$ octets	$\simeq 10$ ns
Mémoire vive	$\equiv 10^9$ octets	$\simeq 50$ ns
Disques durs	$\equiv 10^{12}$ octets	$\simeq 5$ ms
Réseau	$\equiv 10^{21} - 10^{24}$? octets	$\simeq 20$ ms

Les temps d'accès correspondent ici à un accès aléatoire, ils peuvent diminuer fortement en cas de transferts séquentiels de données.

Les types de mémoire

On peut distinguer différents types de mémoire, selon l'usage :

- la mémoire **vive**, accessible en lecture et en écriture, dans laquelle on range les données en cours de traitement et les programmes en cours d'exécution
- la mémoire **de masse**, qui sert d'espace de stockage à plus long terme
- la mémoire **morte** (ou ROM, *Read-Only Memory*), qui contient des données « immuables » comme la séquence d'amorçage de l'ordinateur

Les types de mémoire

On peut aussi considérer la pérennité du stockage et distinguer :

- la mémoire **volatile**, dont les données disparaissent lorsque l'alimentation cesse

  (certains technologies de mémoire nécessitent même un rafraîchissement périodique des données)

- la mémoire **rémanente**, dont les données sont préservées en cas de rupture de l'alimentation

Généralement, la mémoire vive est volatile, tandis que la mémoire de masse est rémanente

Organisation de la mémoire

La mémoire peut être vue comme un ensemble de « cellules » numérotées (leur « adresse ») pouvant contenir des « *bytes* » représentant valeurs numériques, des caractères, des instructions... :

adresse	mémoire
	⋮
49748	11100111
49749	00011011
49750	11111000
49751	01100101
49752	11111001
49753	11101110
49754	00101010
	⋮

Les types de mémoire

Les mémoires, enfin, diffèrent par la façon dont on accède au données :

- les mémoires **à accès direct** (ou RAM, *Random Access Memory*) permettent un accès direct à n'importe quelle case
- les mémoires **à accès séquentiel** (ou SAM, *Sequential Access Memory*) imposent que l'on accède aux cases dans l'ordre

La mémoire vive est virtuellement toujours à accès direct (même si avec certaines technologies des accès séquentiels peuvent être plus rapides), aussi le terme RAM désigne parfois abusivement la mémoire vive

Transmettre les données

Les données se déplacent d'un point à un autre de l'ordinateur à travers des **bus**.

Le principe est le même qu'une lettre. On doit avoir :

- une donnée à transmettre ;
- un destinataire

Composantes d'un bus informatique

Un bus informatique regroupe donc :

- un bus de **données**, qui transmet la donnée proprement dite
- un bus d'**adresse**, qui indique le destinataire de la donnée
- un bus de **contrôle**, qui joue le rôle de chef d'orchestre et donne notamment le « top » pour la transmission

Les différents types de bus informatique

On peut distinguer deux types de bus informatiques :

- les bus **parallèle**, où tout est transmis d'un seul coup (port parallèle, bus ISA, PCI, P-ATA...) ;
- les bus **série**, où les bits sont envoyés séquentiellement (port série, USB, Firewire, PCI-Express, S-ATA...).

Avantages des bus série

Les bus série nécessitent nettement moins de cables.

Paradoxalement ils permettent souvent des débits plus élevés.

(les différents cables d'un bus parallèle, proches, se perturbent mutuellement, limitant le débit).

La fonction factorielle

On souhaite qu'un ordinateur puisse calculer la fonction factorielle :

$$f : \begin{cases} \mathbb{N} \longmapsto \mathbb{N} \\ n \longmapsto n! = n \times (n-1) \times \ldots \times 1 \end{cases}$$

La fonction factorielle

On souhaite qu'un ordinateur puisse calculer la fonction factorielle :

$$f : \begin{cases} \mathbb{N} \longmapsto \mathbb{N} \\ n \longmapsto n! = n \times (n-1) \times \ldots \times 1 \end{cases}$$

Traduit dans le langage d'un processeur (i386) :

```
10111000000000010000000000000000000000010000011
11111001000000010111111000000111000011111101011111
11000001111111111110010011111010111111010011000011
```

La fonction factorielle

En binaire :

```
10111000000000010000000000000000000010000011
11111001000000010111111000000111000011111101011111
11000001111111111110010011110101111110100110000011
```

En écriture hexadécimale :

```
b8 01 00 00 00 83 f9 01 7e 07 0f af c1 ff c9 eb f4 c3
```

Un peu plus facile à recopier, mais pas beaucoup plus simple à écrire.

La fonction factorielle

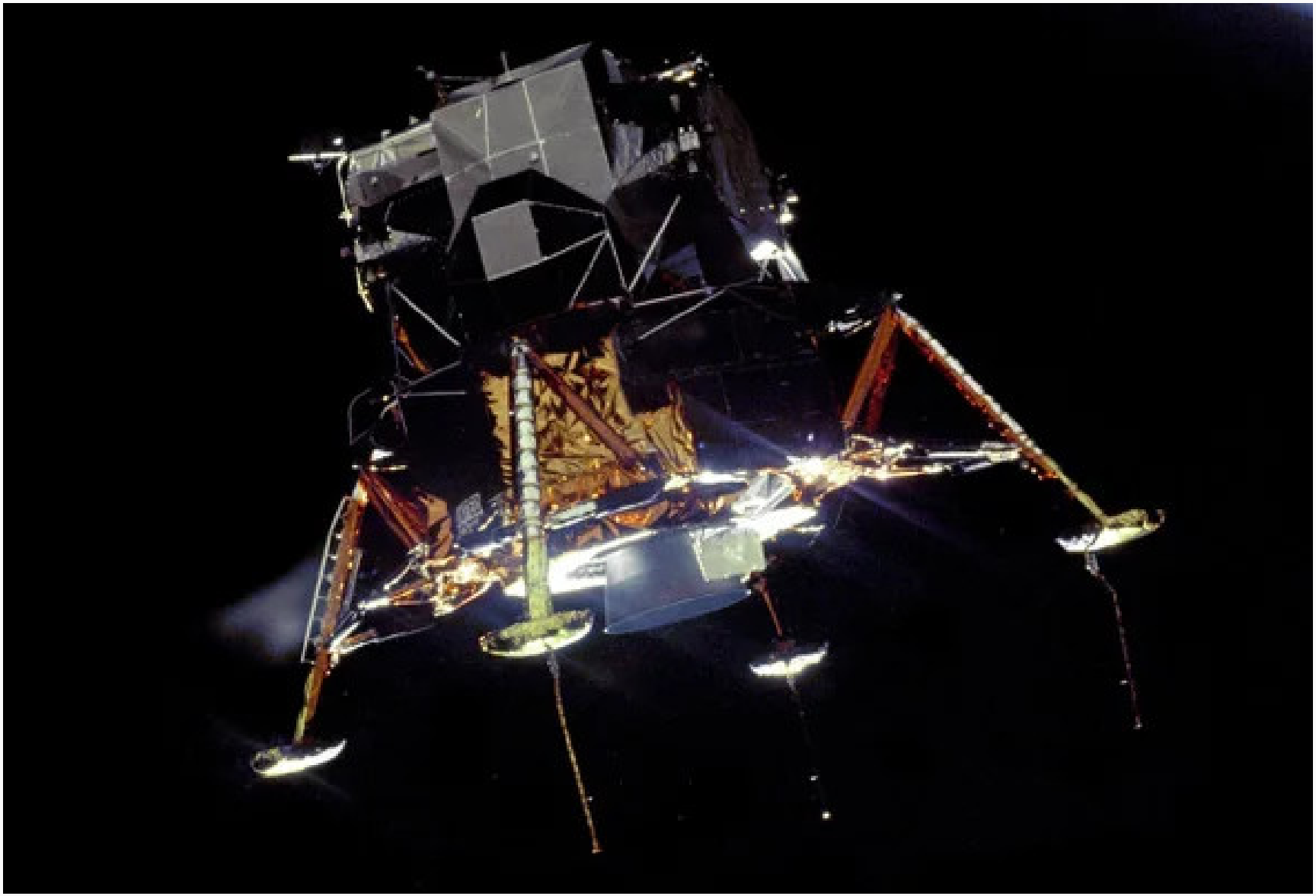
En écriture hexadécimale :

```
b8 01 00 00 00 83 f9 01 7e 07 0f af c1 ff c9 eb f4 c3
```

En langage d'assemblage :

```
        mov    $0x1,%eax
 loop:  cmp    $0x1,%ecx
        jle    end
        imul   %ecx,%eax
        dec    %ecx
        jmp    loop
  end:  ret
```

Un exemple réel

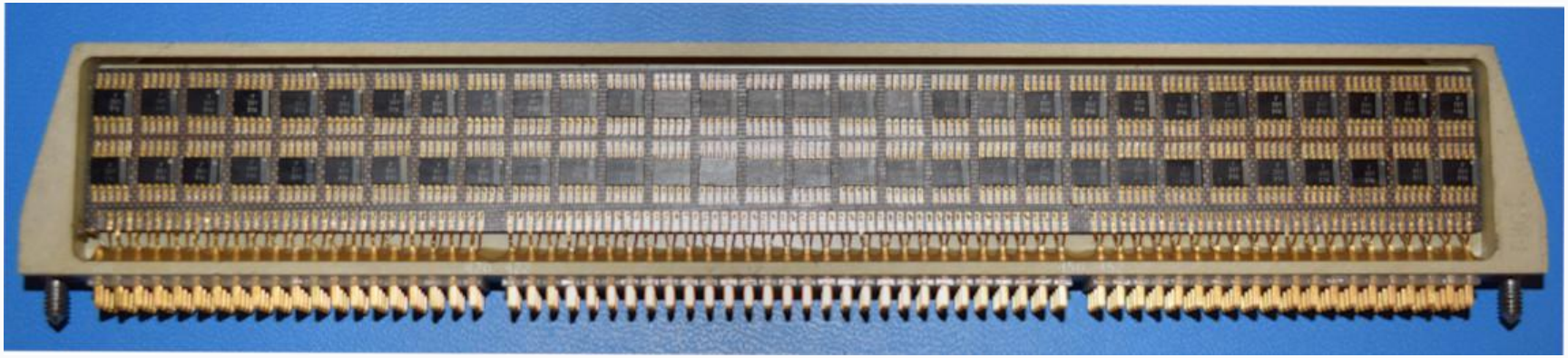


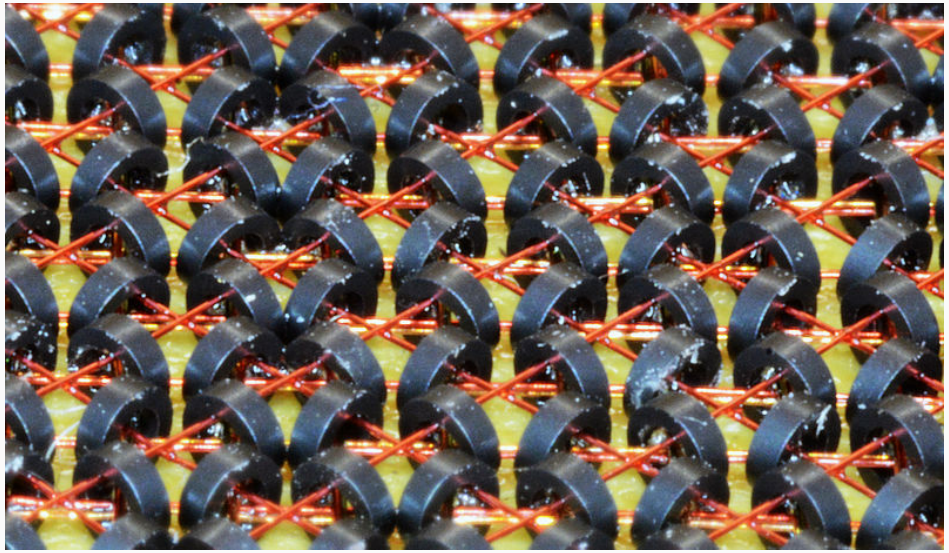
Un morceau de code commenté

```
P63SPOT3  CA      BIT6       # IS THE LR ANTENNA IN POSITION 1 YET
          EXTEND
          RAND    CHAN33
          EXTEND
          BZF     P63SPOT4   # BRANCH IF ANTENNA ALREADY IN POSITION 1

          CAF     CODE500    # ASTRONAUT:    PLEASE CRANK THE
          TC      BANKCALL   #               SILLY THING AROUND
          CADR    GOPERF1
          TCF     GOTOP00H   # TERMINATE
          TCF     P63SPOT3   # PROCEED       SEE IF HE'S LYING

P63SPOT4  TC      BANKCALL   # ENTER         INITIALIZE LANDING RADAR
          CADR    SETPOS1

          TC      POSTJUMP   # OFF TO SEE THE WIZARD ...
          CADR    BURNBABY
```

Margaret Hamilton et son équipe du MIT

Unité arithmétique et mémoire de l'AGC





Retour sur la factorielle

En langage d'assemblage :

```
      mov    $0x1,%eax   # Place 1 dans EAX
loop: cmp    $0x1,%ecx   # Compare ECX et 1
      jle    end         # Si ECX <= 1, saute à (end)
      imul   %ecx,%eax   # Multiplie EAX par ECX
      dec    %ecx        # Décrémente ECX
      jmp    loop        # Saute à (loop)
 end: ret                # Le résultat est dans EAX
```

Retour sur la factorielle

En langage d'assemblage :

```
      mov    $0x1,%eax    # Place 1 dans EAX
loop: cmp    $0x1,%ecx    # Compare ECX et 1
      jle    end          # Si ECX <= 1, saute à (end)
      imul   %ecx,%eax    # Multiplie EAX par ECX
      dec    %ecx         # Décrémente ECX
      jmp    loop         # Saute à (loop)
 end: ret                 # Le résultat est dans EAX
```

En langage C :


```c
int accum = 1;
while (n > 1) {
    accum = accum * n;
    n = n - 1;
}
return acc;
```

Premiers langages de programmation

- 1954 : FORTRAN (FORmula TRANslation)
- 1958 : LISP
- 1958 : ALGOL (ALGOrithmic Language)
- 1959 : COBOL (COmmon Business Oriented Language)

Premiers langages de programmation

- 1954 : FORTRAN (FORmula TRANslation)
- 1958 : LISP
- 1958 : ALGOL (ALGOrithmic Language)
- 1959 : COBOL (COmmon Business Oriented Language)

Le langage C est créé entre 1972 et 1973 par Denis Ritchie

(travaillant chez Bell Laboratories)

pour servir de base au système d'exploitation UNIX

Évolution et standardisation du langage C

- 1978 : ouvrage « *The C Programming Language* » par Brian Kernighan et Dennis Ritchie (K&R C)
- 1989 : normalisation par l'American National Standard Institute (ANSI C / C89)
- 1990 : adoption comme norme ISO (C90, identique)
- 1999 : révision C99
- 2011 : révision C11
- 2018 : révision mineure C17
- 2024 : révision mineure C23

Avantages et inconvénients du langage C

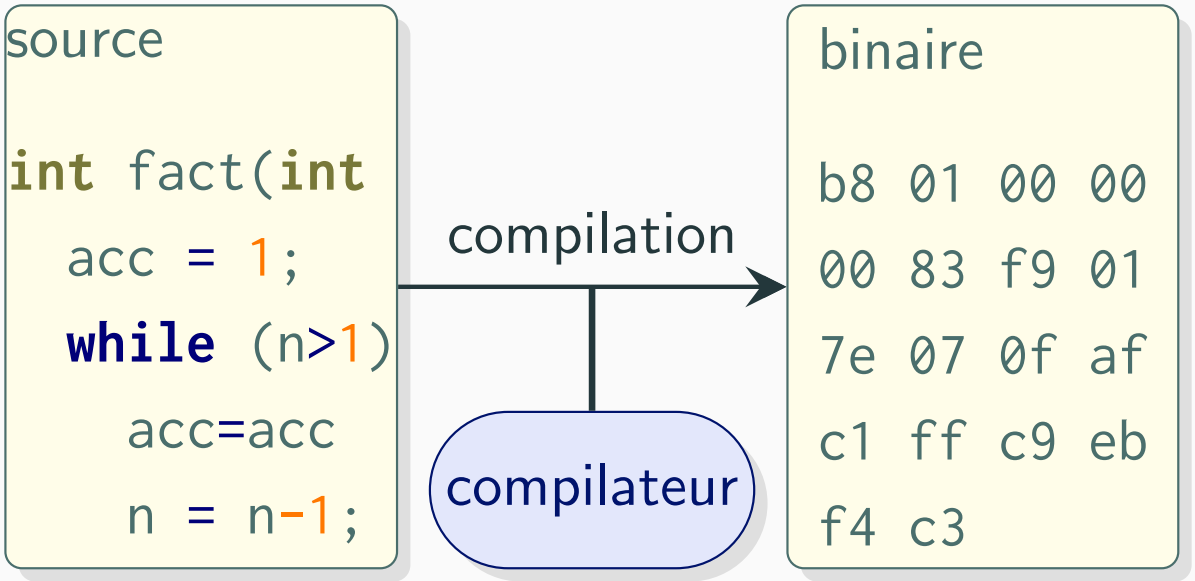
Le langage C est

- proche d'une machine « idéalisée »
- avec une syntaxe restreinte

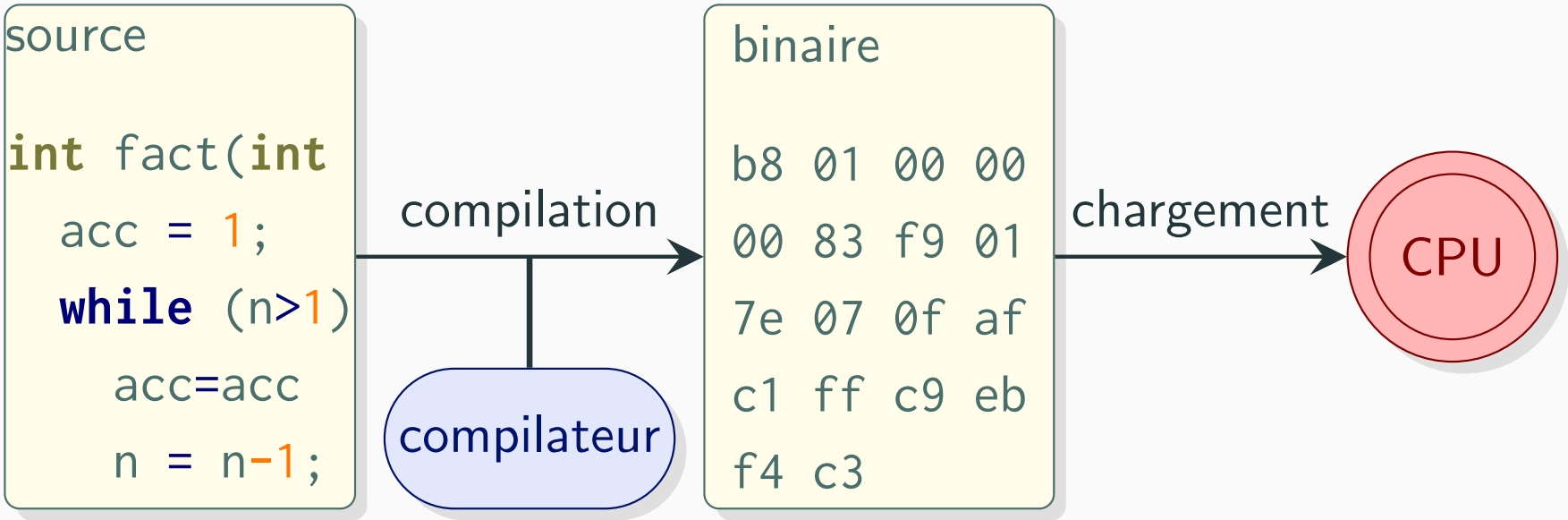
Il n'offre que très peu de protection

(et une expressivité limitée)

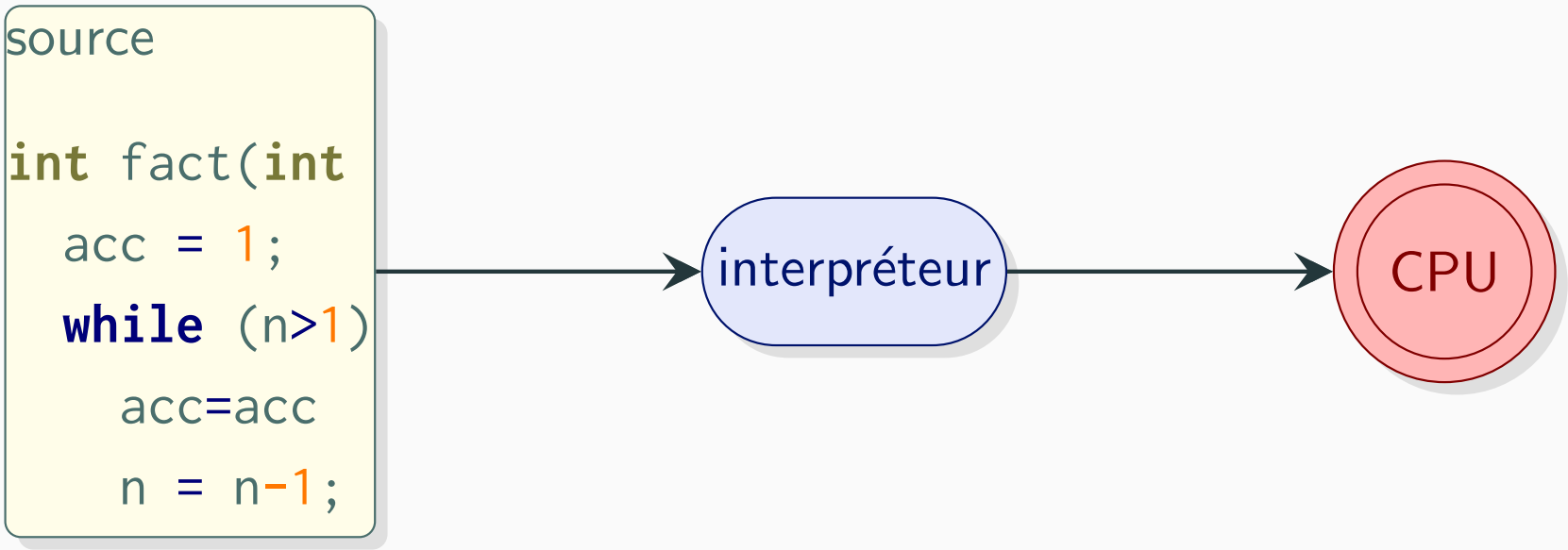
Compilation d'un programme



Compilation d'un programme



Interprétation d'un programme



Un premier programme


```c
#include <stdio.h>                                          // ①
#include <stdlib.h>

int main(void) {                                            // ②
    int cherry;                                             // ③
    int mango;

    cherry = 6;                                             // ④
    mango = cherry + 1;
    printf("Le produit de %d avec %d vaut %d\n",            // ⑤
           cherry, mango, cherry * mango);

    return EXIT_SUCCESS;                                    // ⑥
}
                                                            // ⑦
```

Oups

```
                                                    extern int
                                                       errno
                                                         ;char
                                                            grrr
                                 ;main(                       r,
  argv, argc )                 int    argc                      ,
   r         ;                char *argv[];{int                 P( );
#define x  int i,             j,cc[4];printf("      choo choo\n"    ) ;
x  ;if    (P(  !              i              )       |  cc[  !      j ]
&  P(j    )>2  ?              j              :       i  ){*  argv[i++ +!-i]
;              for    (i=                   0;;    i++                   );
_exit(argv[argc- 2    / cc[1*argc]|-1<<4 ]    ) ;printf("%d",P(""));}}
  P  (    a  )   char a   ;  {    a  ;   while(    a  >      "  B   "
  /* -    by E            ricM    arsh             all-      */);
```

Eric Marshall, soumission IOCCC, 1986

Oups

```c
#define _ -F<00||--F-OO--;
int F=00,OO=00;main(){F_OO();printf("%1.3f\n",4.*-F/OO/OO);}F_OO()
{
              _-_-_-_
          _-_-_-_-_-_-_-_-_
       _-_-_-_-_-_-_-_-_-_-_
     _-_-_-_-_-_-_-_-_-_-_-_-_
   _-_-_-_-_-_-_-_-_-_-_-_-_-_
  _-_-_-_-_-_-_-_-_-_-_-_-_-_-_
  _-_-_-_-_-_-_-_-_-_-_-_-_-_-_
 _-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_
 _-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_
 _-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_
 _-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_
  _-_-_-_-_-_-_-_-_-_-_-_-_-_-_
  _-_-_-_-_-_-_-_-_-_-_-_-_-_-_
   _-_-_-_-_-_-_-_-_-_-_-_-_-_
     _-_-_-_-_-_-_-_-_-_-_-_
        _-_-_-_-_-_-_-_-_
            _-_-_-_-_
}
```

Westley, IOCCC, 1988

Oups

```
                              #include/**/<time.h>
                              #include <ncurses.h>
                              # include <stdlib.h>
            /**               */#define O()for(y-=              !!\
          y;y<H&&           /*...Semi-Automatic.*/y<          p/W+2;\
        y++)for(x=p%   W,x-=!!/*..MineSweeper...*/x;x<W&&   x<p%W+2;x++)
      #define _(x,y)COLOR_##x,COLOR_##y /*  click / (R)estart / (Q)uit  */
    #define Y(n)attrset(COLOR_PAIR(n)),mvprintw(/* IOCCC2019 or IOCCC2020 */
   typedef int I;I*M,W,H,S,C,E,X,T,c,p,q,i,j,k;char G[]=" x",U[256];I F(I p){ I
      r=0,x,y=p/W,q;O()q=y*W+x,r+=M[q]^=p-q?(M[q]&16)<<8:0;return r;}I K(I p
         ,I f,I g){ I x=(g+     f/256)%16-(f+g/256)%16,y=p/W,c=0,n=g/4096
          ,m=x==n?0:x==g            /16%16-f/16%16-n?256:-1; if(m+1)O()if
        ((4368&M[n=y*W               +x])==4112){ M[c=1,n]=(M[n]&~16)|m; }
       return c;}void                D(){I p,k,o=0,n=C,m=0,q=0;if(LINES-1<H
      ||COLS/2<W)clear               (),Y(4)LINES/2,COLS/2-16,"Make the ter\
minal bigger!");else{for            (p=0;p<S;o+=k==3,Y(k)p/W+1,p%W*2,G),p++)G[1]=""
"_*!..12345678"[k=E?256&M[p     ]?n--,2:E-2||M[p]%2<1?M[p]&16?q=p,m++,3:4+F(p)%16:
1:3];k=T+time(0);T=o||T>=0||E-1?T:k;k=T<0?k:T;Y(7)0,0,"%03d%*s%03d",n>999?999:n,W*
2-6,"",k>999?999:k);Y(9)0,W-1,E>1?"X-(":E-1||o?":-)":"8-)");M[q]|=256*(n==m&&n); }
refresh();}short B[]={_(RED,BLACK),_(WHITE,BLUE),_(GREEN,RED),_(MAGENTA,YELLOW),_(
CYAN,RED)};I main(I A,char**V){MEVENT e;FILE*f;srand(time(0));initscr();for(start\
_color();X<12;X++){init_pair(X+1,B[X&&X<10?X-1:2],B[X?X<3?2:1:0]);}noecho();cbreak
();timeout(9);curs_set(0);keypad(stdscr,TRUE);for(mousemask(BUTTON1_CLICKED|BUTTO\
N1_RELEASED,0);;){S=A<2?f=0,W=COLS/2,H=LINES-1,C=W*H/5,0:fscanf(f=fopen(V[A-1],"r"
      ),"%d %d %d",&W,&H,&C)>3; ;S+=W*H;M=realloc(M,S*sizeof(I)*2);for(i=0
       ;i<S;i++)!f?M[i]=i,i&&(k=M[j=rand()%i],M[j]=M[i],M[i]=k):fscanf(f,
        "%d",M+i);if(f)fclose(f);T=E=X=0;for(clear();D(),c=getch(),c-'r'
         &&(c-KEY_RESIZE||E);){ if(c=='q'){ return(endwin(),0); }if(c==
       KEY_MOUSE&&getmouse(&e)==OK&&e.x/2<W&&e.y<=H){if(!e.y&&(W-2<e.x&&
     e.x<W+2)){break;}p=e.x/2+e.y*W-W;if(p>=0){if(!E){for(i=0;i<S;i++)M[S+M
   [i]]=i,M[i]=16+(M[i]<C);C-=M[p]&1;M[p]=16;E=1;T=-time(0);}if(E<2)M[p]&=(M[p]
    &257)==1?T+=time(0),E=2,273:257;}}for(p=0;p<S&&E==1;M[p++]&=273){}for(i=
     (X+S-1)%S;E==1&&i!=X;X=(X+1)%S){if(!(M[p=M[X+S]]&272)){if(K(p,c=F(p)
       ,0)){goto N;}   for(k=p/W-2,k=k<0?0:k;k<p/W+3&&k   <H;k++)for(j=
         p%W-2,j          =j<0?0:j;j<W&&j<p%W+3;)if          (!(M[q=
           k*W               +j++]&272)){ if(K(p,              c,F
                             (q))){ goto N; }F(q)
                             ; }F(p); }}N:; } } }
                             /*(c)Yusukse Endoh*/
```

Endoh, IOCCC, 2020

Oups

```
#include<stdio.h>
typedef unsigned int _;_ d,b,
#define i(I1,Il,lI)if(Il){lI;}else{I1;}
I[256],                          n,y,a,r,u,k,o
,L,l[                                  256],O,K[
/**/                                    #define\
q(g)                                        g char\
*C,                                           *Q,c[\
]=                                              "KfW"\
""                                                "Ww|"\
 /*                                                   'UU!\
  %                                                    NYA!\
  */                                                      "Z}"\
";fRo?JtJaV<x4@*?R?&JV1"                                  ".s"\
"{Fyj2_;khB1xQ5oxm~mS@B|(pa>oRU"                              "Ro"\
"nB}h@o?)d.X)NSTIUCz7@%",*s[]={c,"#en"                      "di"\
"f/*}||1;\n__DATA__\40*/\n\n#ifndef\40q\n#d"                    "ef"\
"ine\x20q\n#include<stdio.h>\ntypedef\40unsign"                    "e"\
"d\x20int\x20_;_\x20K[]={\n#include\40__FILE__\n#u"                  "n"\
"def q","0},L,O,l[256],I[256],n,y,a,r,u,k,o;"#g"char"              "*"\
"S,s[]=\"",c,c,"\";int main(){X();for(S=s+*K;*S>37;){for"            "(o"\
"=0;o<5;o++)r=r*85+(83+*S++)%89;r","^=*x();for(o=0;o<4;o++"           ")"\
"{s[O++]=r&255;r>>=8;}}return!fwrite(s,O-*S%5,1,stdout);}\n"           "#"\
"endif",c},S[256]="#ifdef/*\n'true'\40or\40q{\nexec\40head\40"          "-"\
"8\40$0\n};for(open$O,$0;<$O>;print\40if$f){$f|=/^$/;}q{*/q",/*          */z;
256];q(_*x(){if(!L--){y+=++a;for(o=0;o<256;y=l[o++]=I[255&(k>>10          )]+u
){n^=(o&1)?n>>(( o& 2)?16:6):n<<((o&2)?2:13);u=I[o];k=I[o]=I[255&         (u>>
2)]+(n+=I[(o+128)  &   255]) +y;}L=255;}return&l[L];}_*X(){for(O=0         ;256
>O;I[O++]=0);for(O   =     0;   sizeof(K)/sizeof( _)> O;O++)I[O&255]      ^=K[
O];for(n=y=a=L=O=0 ;O<1<<24;++   O)x( );r=O=0x0; return&O;})int/*^^*/      main
(int p,char**P){FILE* Z=fopen(p>    (+  1)?P[01   ]   :"/dev/urandom",      "rb"
);i(;,Z,O=fread(K,256  ,4,Z);/*P          */     fclose(Z))X();for(p=b=d      =O=
0;O<256;K[O++]=0)*K+=+  86;for(O               =1;12> O;K[O++]=*x());X();     for
(C=Q=S;r-8;){i(*C++=34,  (r-4&&r              -5)||C- S ,;)z=Q[p++];i(;,z     !=
32||r-3,i(i(C+=sprintf((    C),                "%uU"    ",",K[b++]);i(d=1;C    =S
;i(d=02,b-12,;),b%6,;),r-1                      ,i(b=   fread(c,1,4,stdin);i    (p
=O=0,b,for(d=O=0;O<04;O++)d                           +=(c[O]&255)<<(8*O);d    ^=
*x();for(p=5;p;c[--p]=O<32?O+                          95:O+6){O=d%85;d/=85;}   O=
5)i(d=0,b<4,c[O++]=b?b-1?b-2?36:      37:33:35          ;d=2)c[O]=0,r-4,i(i (d=  2
|d,C!=S+6,*C++=(*x()%34)+93;p--),r      -5,*s=          C;d|=2)  )),z ,i(*C++ =92
z-63||C [-1]-63||C>S+76,;)*C++=z))i(                    ;,d>1,d=  d-2  ;Q=s[r]  ;i(
,r<3||  r>5,d=1;i(;,r-1, *C=0)C=S)  i(;,           r-4, p=0)++   r)   i(*(C++ )=
34,r    <4||r>5||   C<S+    78,;)i       (*C++=0;d=1; C=S   ,r<3       ||       r>
5       ||C<S+     79,;      )i(;,d,                    puts             (
        S);         d=0      )}  return
                                    0;}
```

Omoikane, IOCCC, 2012

Oups

```
#include                                  <math.h>
#include                                <sys/time.h>
#include                                <X11/Xlib.h>
#include                               <X11/keysym.h>
                                       double L ,o ,P
                                      ,_=dt,T,Z,D=1,d,
                                      s[999],E,h= 8,I,
                                      J,K,w[999],M,m,O
                                     ,n[999],j=33e-3,i=
                                     1E3,r,t, u,v ,W,S=
                                     74.5,l=221,X=7.26,
                                     a,B,A=32.2,c, F,H;
                                     int N,q, C, y,p,U;
                                   Window z; char f[52]
                                 ; GC k; main(){ Display*e=
 XOpenDisplay( 0); z=RootWindow(e,0); for (XSetForeground(e,k=XCreateGC (e,z,0,0),BlackPixel(e,0))
; scanf("%lf%lf%lf",y +n,w+y, y+s)+1; y ++); XSelectInput(e,z= XCreateSimpleWindow(e,z,0,0,400,400,
0,0,WhitePixel(e,0) ),KeyPressMask); for(XMapWindow(e,z); ; T=sin(O)){ struct timeval G={ 0,dt*1e6}
; K= cos(j); N=1e4; M+= H*_; Z=D*K; F+=_*P; r=E*K; W=cos( O); m=K*W; H=K*T; O+=D*_*F/ K+d/K*E*_; B=
sin(j); a=B*T*D-E*W; XClearWindow(e,z); t=T*E+ D*B*W; j+=d*_*D-_*F*E; P=W*E*B-T*D; for (o+=(I=D*W+E
*T*B,E*d/K *B+v+B/K*F*D)*_; p<y; ){ T=p[s]+i; E=c-p[w]; D=n[p]-L; K=D*m-B*T-H*E; if(p [n]+w[ p]+p[s
]== 0|K <fabs(W=T*r-I*E +D*P) |fabs(D=t *D+Z *T-a *E)> K)N=1e4; else{ q=W/K *4E2+2e2; C= 2E2+4e2/ K
 *D; N-1E4&& XDrawLine(e ,z,k,N ,U,q,C); N=q; U=C; } ++p; } L+=_* (X*t +P*M+m*l); T=X*X+ l*l+M *M;
  XDrawString(e,z,k ,20,380,f,17); D=v/l*15; i+=(B *l-M*r -X*Z)*_; for(; XPending(e); u *=CS!=N){
                                  XEvent z; XNextEvent(e ,&z);
                                      ++*((N=XLookupKeysym
                                        (&z.xkey,0))-IT?
                                        N-LT? UP-N?& E:&
                                        J:& u: &h); --*(
                                        DN -N? N-DT ?N==
                                        RT?&u: & W:&h:&J
                                         ); } m=15*F/l;
                                         c+=(I=M/ l,l*H
                                         +I*M+a*X)*_; H
                                         =A*r+v*X-F*l+(
                                         E=.1+X*4.9/l,t
                                         =T*m/32-I*T/24
                                          )/S; K=F*M+(
                                          h* 1e4/l-(T+
                                          E*5*T*E)/3e2
                                          )/S-X*d-B*A;
                                          a=2.63 /l*d;
                                          X+=( d*l-T/S
                                           *(.19*E +a
                                           *.64+J/1e3
                                           )-M* v +A*
                                           Z)*_; l +=
                                           K *_; W=d;
                                           sprintf(f,
                                           "%5d  %3d"
                                           "%7d",p =l
                                          /1.7,(C=9E3+
                                 O*57.3)%0550,(int)i); d+=T*(.45-14/l*
                                X-a*130-J* .14)*_/125e2+F*_*v; P=(T*(47
                                *I-m* 52+E*94 *D-t*.38+u*.21*E) /1e2+W*
                                179*v)/2312; select(p=0,0,0,0,&G); v-=(
                                 W*F-T*(.63*m-I*.086+m*E*19-D*25-.11*u
                                  )/107e2)*_; D=cos(o); E=sin(o); } }
```

Banks, IOCCC, 1998

Oups

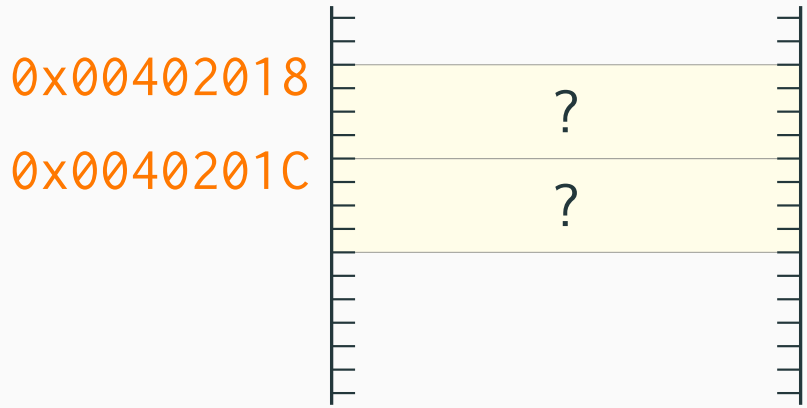

```c
#include <stdio.h>
#include <math.h>
#define clear 1;if(c>=11){c=0;sscanf(_,"%lf%c",&r,&c);while(*++_-c);}\
  else if(argc>=4&&!main(4-(*_++=='('),argv))_++;g:c+=
#define puts(d,e) return 0;}{double a;int b;char c=(argc<4?d)&15;\
  b=(*_%__LINE__+7)%9*(3*e>>c&1);c+=
#define I(d) (r);if(argc<4&&*#d==*_){a=r;r=usage?r*a:r+a;goto g;}c=c
#define return if(argc==2)printf("%f\n",r);return argc>=4+
#define usage main(4-__LINE__/26,argv)
#define calculator *_*(int)
#define l (r);r=--b?r:
#define _ argv[1]
#define x

double r;
int main(int argc,char** argv){
  if(argc<2){
    puts(
      usage: calculator 11/26+222/31
      +~~~~~~~~~~~~~~~~~~~~~~~~~calculator-\
      !                          7.584,367 )
      +~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~+
      ! clear ! 0 ||l   -x  l   tan  I (/) |
      +~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~+
      ! 1 | 2 | 3 ||l  1/x  l   cos  I (*) |
      +~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~+
      ! 4 | 5 | 6 ||l  exp  l  sqrt  I (+) |
      +~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~+
      ! 7 | 8 | 9 ||l  sin  l   log  I (-) |
      +~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(0
    );
  }
  return 0;
}
```

Hou, IOCCC, 2011

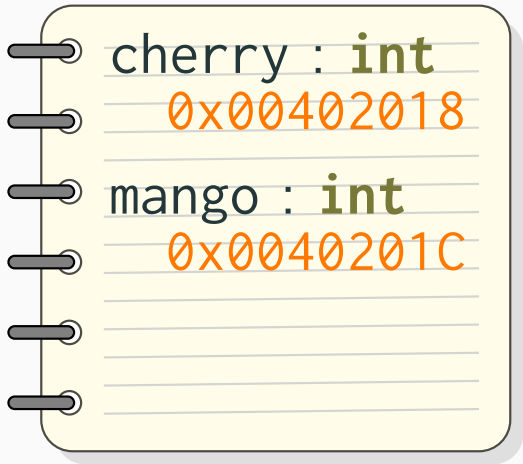
Déclarations

```c
int cherry;
```

- `int` : entiers
- `double` : flottants ($\simeq réels$)
- `bool` : booléens



Mémoire



Devenir des variables après compilation

```
c7 45 fc 06 00 00 00    movl   $0x6, -0x4(%rbp)
8b 45 fc                mov    -0x4(%rbp), %eax
83 c0 01                add    $0x1, %eax
89 45 f8                mov    %eax, -0x8(%rbp)
8b 45 fc                mov    -0x4(%rbp), %eax
0f af 45 f8             imul   -0x8(%rbp), %eax
```

Précédence des opérateurs

	opération	associativité
`+ - !` `(type)`	plus/moins/négation unaire conversion	⇐
`* / %`	multiplication/division	⇒
`+ -`	addition/soustraction	⇒
`< <= > >=`	comparaisons	⇒
`== !=`	égalité/non égalité	⇒
`&&`	■ et ■ logique	⇒
`||`	■ ou ■ logique	⇒
`= += ...`	affectation et affectations augmentées	⇐

Précédence des opérateurs

Exemple :

```c
bool b = n % 2 == 0 && n % 3 != 0;
```

Portée des variables


```c
#include <stdio.h>
                                    // ①
int mango = 1;

int main(void) {
    int cherry = 2;                 // ②
    ...                             // ③
    int apple = 3
    ...                             // ④
    if (...) {
        ...                         // ⑤
        int banana = 4;
        ...                         // ⑥
    }                               // ⑦
    ...
    return 0;
}                                   // ⑧
...                                 // ⑨
```

Occultation de variables


```
... {
    ...                          // ①
    int mango = 37;              // ②
    ...                          // ③
    ... {
        ...                      // ④
        int mango = 42           // ⑤
        ...                      // ⑥
    }                            // ⑦
    ...                          // ⑧
}                                // ⑨
```