

Architecture des ordinateurs

Samuele Giraudo

Université Paris-Est Marne-la-Vallée

LIGM, bureau 4B055

`samuele.giraudo@u-pem.fr`

`http://igm.univ-mlv.fr/~giraudo/`

L'informatique

Informatique : contraction d'« **in**formation » et d'« autom**at**ique »
~> Traitement automatique de l'information.

L'informatique

Informatique : contraction d'« **information** » et d'« **automatique** »
~> Traitement automatique de l'information.

Ordinateur : machine automatique de traitement de l'information obéissant à des programmes formés par des suites d'opérations arithmétiques et logiques.

L'informatique

Informatique : contraction d'« **information** » et d'« **automatique** »
↪ Traitement automatique de l'information.

Ordinateur : machine automatique de traitement de l'information obéissant à des programmes formés par des suites d'opérations arithmétiques et logiques.

On utilise des ordinateurs pour

- 1 **accélérer** les calculs ;
- 2 **traiter** de gros volumes de données.

Point de vue adopté

« L'informatique n'est pas plus la science des ordinateurs que l'astronomie n'est celle des télescopes. »

— M. R. Fellows, I. Parberry

Point de vue adopté

« L'informatique n'est pas plus la science des ordinateurs que l'astronomie n'est celle des télescopes. »

— M. R. Fellows, I. Parberry

L'informatique est un donc vaste domaine qui comprend, entre autres,

- l'algorithmique ;
- la combinatoire ;
- la calculabilité ;
- la cryptographie ;
- l'intelligence artificielle ;
- le traitement d'images ;
- les réseaux ;
- l'étude des machines.

Point de vue adopté

« L'informatique n'est pas plus la science des ordinateurs que l'astronomie n'est celle des télescopes. »

— M. R. Fellows, I. Parberry

L'informatique est un donc vaste domaine qui comprend, entre autres,

- l'algorithmique ;
- la combinatoire ;
- la calculabilité ;
- la cryptographie ;
- l'intelligence artificielle ;
- le traitement d'images ;
- les réseaux ;
- l'étude des machines.

Ce cours s'inscrit dans ce dernier point.

Objectifs du module et bibliographie

Trois axes principaux :

- 1 introduction au **codage des données** ;

Objectifs du module et bibliographie

Trois axes principaux :

- 1 introduction au **codage des données** ;
- 2 connaissance des principes de **fonctionnement d'un ordinateur** ;

Objectifs du module et bibliographie

Trois axes principaux :

- 1 introduction au **codage des données** ;
- 2 connaissance des principes de **fonctionnement d'un ordinateur** ;
- 3 bases de la **programmation en langage machine**.

Objectifs du module et bibliographie

Trois axes principaux :

- 1 introduction au **codage des données** ;
- 2 connaissance des principes de **fonctionnement d'un ordinateur** ;
- 3 bases de la **programmation en langage machine**.

Bibliographie (minimale) :

- P. Carter, *PC Assembly Language*, 2006,
<http://www.drpaulcarter.com/pcasm/> ;
- J. L. Hennessy, D. A. Patterson, *Architectures des ordinateurs : une approche quantitative*, 2003.

Plan du cours

1 Histoire

- Chronologies
- Mécanismes
- Lampes
- Transistors

2 Représentation

- Bits
- Entiers
- Réels

- Caractères

3 Programmation

- Assembleur
- Bases
- Sauts
- Fonctions

4 Optimisations

- Pipelines
- Mémoires

Plan

1 Histoire

- Chronologies
- Mécanismes
- Lampes
- Transistors

Plan

- 1 Histoire
 - Chronologies
 - Mécanismes
 - Lampes
 - Transistors

Domaines de progression

Les progrès en la matière du développement des ordinateurs sont la conséquence d'avancées de trois sortes :

Domaines de progression

Les progrès en la matière du développement des ordinateurs sont la conséquence d'avancées de trois sortes :

- 1 les **découvertes théoriques** (mathématiques, informatique théorique) ;

Domaines de progression

Les progrès en la matière du développement des ordinateurs sont la conséquence d'avancées de trois sortes :

- 1 les **découvertes théoriques** (mathématiques, informatique théorique) ;
- 2 les **avancées technologiques** (physique, électronique) ;

Domaines de progression

Les progrès en la matière du développement des ordinateurs sont la conséquence d'avancées de trois sortes :

- 1 les **découvertes théoriques** (mathématiques, informatique théorique) ;
- 2 les **avancées technologiques** (physique, électronique) ;
- 3 les **réalisations concrètes d'ordinateurs** (ingénierie).

Les avancées théoriques majeures

-350 : Aristote fonda les bases de la logique ;

Les avancées théoriques majeures

-350 : Aristote fonda les bases de la **logique** ;

1703 : G. W. Leibniz inventa le **système binaire** ;

Les avancées théoriques majeures

-350 : Aristote fonda les bases de la **logique** ;

1703 : G. W. Leibniz inventa le **système binaire** ;

1854 : G. Boole introduisit l'**algèbre de Boole** ;

Les avancées théoriques majeures

- 350 : Aristote fonda les bases de la **logique** ;
- 1703 : G. W. Leibniz inventa le **système binaire** ;
- 1854 : G. Boole introduisit l'**algèbre de Boole** ;
- 1936 : A. M. Turing introduisit la **machine de Turing** ;

Les avancées théoriques majeures

- 350 : Aristote fonda les bases de la **logique** ;
- 1703 : G. W. Leibniz inventa le **système binaire** ;
- 1854 : G. Boole introduisit l'**algèbre de Boole** ;
- 1936 : A. M. Turing introduisit la **machine de Turing** ;
- 1938 : C. Shannon expliqua comment **utiliser l'algèbre de Boole** dans des circuits ;

Les avancées théoriques majeures

- 350 : Aristote fonda les bases de la **logique** ;
- 1703 : G. W. Leibniz inventa le **système binaire** ;
- 1854 : G. Boole introduisit l'**algèbre de Boole** ;
- 1936 : A. M. Turing introduisit la **machine de Turing** ;
- 1938 : C. Shannon expliqua comment **utiliser l'algèbre de Boole** dans des circuits ;
- 1945 : J. von Neumann définit l'architecture des ordinateurs modernes : la **machine de von Neumann** ;

Les avancées théoriques majeures

- 350 : Aristote fonda les bases de la **logique** ;
- 1703 : G. W. Leibniz inventa le **système binaire** ;
- 1854 : G. Boole introduisit l'**algèbre de Boole** ;
- 1936 : A. M. Turing introduisit la **machine de Turing** ;
- 1938 : C. Shannon expliqua comment **utiliser l'algèbre de Boole** dans des circuits ;
- 1945 : J. von Neumann définit l'architecture des ordinateurs modernes : la **machine de von Neumann** ;
- 1948 : C. Shannon introduisit la **théorie de l'information**.

Les avancées technologiques majeures

1904 : J. A. Fleming inventa la diode à vide ;

Les avancées technologiques majeures

1904 : J. A. Fleming inventa la **diode à vide** ;

1948 : J. Bardeen, W. Shockley et W. Brattain inventèrent le **transistor** ;

Les avancées technologiques majeures

1904 : J. A. Fleming inventa la **diode à vide** ;

1948 : J. Bardeen, W. Shockley et W. Brattain inventèrent le **transistor** ;

1958 : J. Kilby construisit le premier **circuit intégré** ;

Les avancées technologiques majeures

1904 : J. A. Fleming inventa la **diode à vide** ;

1948 : J. Bardeen, W. Shockley et W. Brattain inventèrent le **transistor** ;

1958 : J. Kilby construisit le premier **circuit intégré** ;

1971 : la société Intel conçut le premier **microprocesseur**, l'INTEL 4004.

Les réalisations majeures

Antiquité : utilisation et conception d'abaques ;

Les réalisations majeures

Antiquité : utilisation et conception d'**abaques** ;

1623 : W. Schickard conçut les plans de la première machine à calculer, l'**horloge calculante** ;

Les réalisations majeures

Antiquité : utilisation et conception d'**abaques** ;

1623 : W. Schickard conçut les plans de la première machine à calculer, l'**horloge calculante** ;

1642 : B. Pascal construisit la **Pascaline** ;

Les réalisations majeures

Antiquité : utilisation et conception d'**abaques** ;

1623 : W. Schickard conçut les plans de la première machine à calculer, l'**horloge calculante** ;

1642 : B. Pascal construisit la **Pascaline** ;

1801 : J. M. Jacquard inventa le premier **métier à tisser programmable** ;

Les réalisations majeures

Antiquité : utilisation et conception d'**abaques** ;

1623 : W. Schickard conçut les plans de la première machine à calculer, l'**horloge calculante** ;

1642 : B. Pascal construisit la **Pascaline** ;

1801 : J. M. Jacquard inventa le premier **métier à tisser programmable** ;

1834 : C. Babbage proposa les plans de la **machine analytique** ;

Les réalisations majeures

Antiquité : utilisation et conception d'**abaques** ;

1623 : W. Schickard conçut les plans de la première machine à calculer, l'**horloge calculante** ;

1642 : B. Pascal construisit la **Pascaline** ;

1801 : J. M. Jacquard inventa le premier **métier à tisser programmable** ;

1834 : C. Babbage proposa les plans de la **machine analytique** ;

1887 : H. Hollerith construisit une **machine à cartes perforées** ;

Les réalisations majeures

Antiquité : utilisation et conception d'**abaques** ;

1623 : W. Schickard conçut les plans de la première machine à calculer, l'**horloge calculante** ;

1642 : B. Pascal construisit la **Pascaline** ;

1801 : J. M. Jacquard inventa le premier **métier à tisser programmable** ;

1834 : C. Babbage proposa les plans de la **machine analytique** ;

1887 : H. Hollerith construisit une **machine à cartes perforées** ;

1949 : M. V. Wilkes créa un ordinateur à architecture de von Neumann, l'**EDSAC**.

Les générations d'ordinateurs

Le dernier siècle peut se découper en **générations**, suivant le matériel utilisé. Des quatre principales avancées technologiques découlent les quatre générations suivantes.

Les générations d'ordinateurs

Le dernier siècle peut se découper en **générations**, suivant le matériel utilisé. Des quatre principales avancées technologiques découlent les quatre générations suivantes.

1^{re} génération : de 1936 à 1956, emploi de **tubes à vide**.

Les générations d'ordinateurs

Le dernier siècle peut se découper en **générations**, suivant le matériel utilisé. Des quatre principales avancées technologiques découlent les quatre générations suivantes.

1^{re} **génération** : de 1936 à 1956, emploi de **tubes à vide**.

2^e **génération** : de 1956 à 1963, emploi de **transistors**.

Les générations d'ordinateurs

Le dernier siècle peut se découper en **générations**, suivant le matériel utilisé. Des quatre principales avancées technologiques découlent les quatre générations suivantes.

1^{re} **génération** : de 1936 à 1956, emploi de **tubes à vide**.

2^e **génération** : de 1956 à 1963, emploi de **transistors**.

3^e **génération** : de 1963 à 1971, emploi de **circuits intégrés**.

Les générations d'ordinateurs

Le dernier siècle peut se découper en **générations**, suivant le matériel utilisé. Des quatre principales avancées technologiques découlent les quatre générations suivantes.

1^{re} **génération** : de 1936 à 1956, emploi de **tubes à vide**.

2^e **génération** : de 1956 à 1963, emploi de **transistors**.

3^e **génération** : de 1963 à 1971, emploi de **circuits intégrés**.

4^e **génération** : de 1971 à 2016 (et plus), emploi de **microprocesseurs**.

Les générations d'ordinateurs

Le dernier siècle peut se découper en **générations**, suivant le matériel utilisé. Des quatre principales avancées technologiques découlent les quatre générations suivantes.

1^{re} **génération** : de 1936 à 1956, emploi de **tubes à vide**.

2^e **génération** : de 1956 à 1963, emploi de **transistors**.

3^e **génération** : de 1963 à 1971, emploi de **circuits intégrés**.

4^e **génération** : de 1971 à 2016 (et plus), emploi de **microprocesseurs**.

Parcourons maintenant l'évolution des machines en les rangeant en trois catégories : les machines à mécanismes, les machines à lampes et les machines à transistors.

Plan

1 Histoire

- Chronologies
- Mécanismes
- Lampes
- Transistors

Le temps des machines à mécanismes

Cette période s'étend de l'antiquité et se termine dans les années 1930.

Le temps des machines à mécanismes

Cette période s'étend de l'antiquité et se termine dans les années 1930.

Souvent d'initiative isolées, les machines construites à cette époque possédaient néanmoins souvent quelques points communs :

- utilisation d'**engrenages** et de **courroies** ;
- nécessité d'une **source physique de mouvement** pour fonctionner ;
- machines construites pour un **objectif fixé *a priori*** ;
- espace de **stockage très faible** ou inexistant.

Le temps des machines à mécanismes

Cette période s'étend de l'antiquité et se termine dans les années 1930.

Souvent d'initiative isolées, les machines construites à cette époque possédaient néanmoins souvent quelques points communs :

- utilisation d'**engrenages** et de **courroies** ;
- nécessité d'une **source physique de mouvement** pour fonctionner ;
- machines construites pour un **objectif fixé *a priori*** ;
- espace de **stockage très faible** ou inexistant.

Ces machines faisaient appel à des connaissances théoriques assez rudimentaires, toute proportion gardée vis-à-vis de leur époque d'introduction.

Les abaqués

Abaque : instrument mécanique permettant de réaliser des calculs.

Exemples :



Cailloux



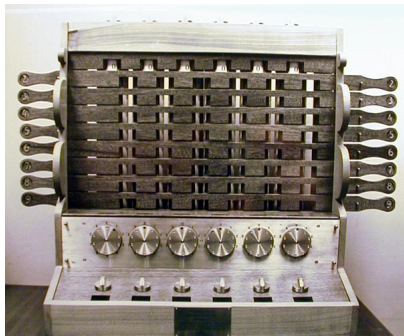
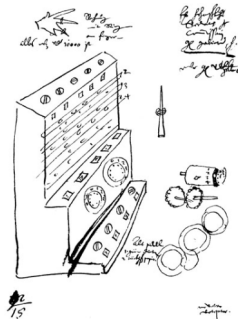
Bouliers



Bâtons de Napier
(J. Napier, 1617)

L'horloge calculante

En 1623, Schickard écrivit à Kepler pour lui présenter les plans de son « horloge calculante ».



Elle permettait de faire des calculs arithmétiques, utilisait des roues dentées et gérât le report de retenue. Elle ne fut construite qu'en 1960.

La Pascaline

En 1642, Pascal conçut la « **Pascaline** ».



Elle permettait de réaliser des additions et soustractions de nombres décimaux jusqu'à six chiffres.

La Pascaline

En 1642, Pascal conçut la « **Pascaline** ».



Elle permettait de réaliser des additions et soustractions de nombres décimaux jusqu'à six chiffres.

En 1671, Leibniz améliora la Pascaline de sorte à gérer multiplications et divisions.



La Pascaline

En 1642, Pascal conçu la « **Pascaline** ».



Elle permettait de réaliser des additions et soustractions de nombres décimaux jusqu'à six chiffres.

En 1671, Leibniz améliora la Pascaline de sorte à gérer multiplications et divisions.



Ces machines permettent de faire des calculs mais leur **tâche** est **fixée dès leur construction** : la notion de programme n'a pas encore été découverte.

Le métier à tisser programmable

En 1801, Jacquard proposa le premier métier à tisser programmable, le « **Métier Jacquard** ».



Il était piloté par des **cartes perforées**. Celles-ci dirigeaient le comportement de la machine pour tisser des motifs voulus.

Le métier à tisser programmable

En 1801, Jacquard proposa le premier métier à tisser programmable, le « **Métier Jacquard** ».

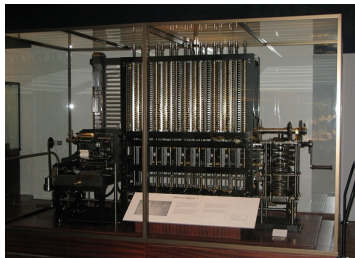
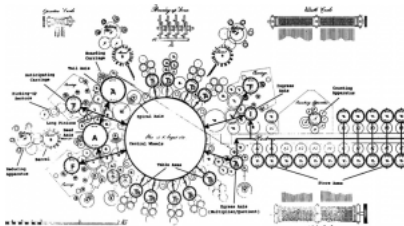


Il était piloté par des **cartes perforées**. Celles-ci dirigeaient le comportement de la machine pour tisser des motifs voulus.

Ces cartes jouaient le rôle de **programme** : une même machine pouvait exécuter des tâches différentes sans avoir à être matériellement modifiée.

La machine analytique

En 1834, Babbage proposa les plans d'une machine, la **machine analytique**.



Il ne put cependant pas la construire. Elle ne le fut finalement que dans les années 1990.

La machine analytique — quelques caractéristiques

La machine analytique est étonnamment moderne pour avoir été imaginée si tôt dans l'histoire.

La machine analytique — quelques caractéristiques

La machine analytique est étonnamment moderne pour avoir été imaginée si tôt dans l'histoire.

Elle comprend, de manière séparée et bien organisée,

- un lecteur de cartes perforées pour les instructions du programme ;
- un lecteur de cartes perforées pour les données passées en entrée ;
- une unité de calcul (le « moulin ») ;
- une mémoire (le « magasin »).

La machine à cartes perforées

En 1887, Hollerith construisit une **machine à carte perforées** pour faciliter le recensement de la population des États-Unis.

[illegible]

Les cartes perforées servaient à décrire les caractéristiques des individus en y percevant des trous.

La machine pouvait ainsi dénombrer les individus selon plusieurs critères.

Plan

1 Histoire

- Chronologies
- Mécanismes
- Lampes
- Transistors

Le temps des machines à lampes

Cette période débuta dans les années 1930 avec l'introduction de la machine (théorique) de Turing et se termina dans les années 1950.

Le temps des machines à lampes

Cette période débuta dans les années 1930 avec l'introduction de la machine (théorique) de Turing et se termina dans les années 1950.

D'un **point de vue matériel**, elle était basée sur

- les relais ;
- les tubes à vide ;
- les mémoires à tores de ferrite ;
- les bandes magnétiques.

Le temps des machines à lampes

Cette période débuta dans les années 1930 avec l'introduction de la machine (théorique) de Turing et se termina dans les années 1950.

D'un **point de vue matériel**, elle était basée sur

- les relais ;
- les tubes à vide ;
- les mémoires à tores de ferrite ;
- les bandes magnétiques.

D'un **point de vue théorique**, elle avait pour fondements le système binaire, la machine de Turing et l'architecture de von Neumann (pour la plupart).

Le temps des machines à lampes

Cette période débuta dans les années 1930 avec l'introduction de la machine (théorique) de Turing et se termina dans les années 1950.

D'un **point de vue matériel**, elle était basée sur

- les relais ;
- les tubes à vide ;
- les mémoires à tores de ferrite ;
- les bandes magnétiques.

D'un **point de vue théorique**, elle avait pour fondements le système binaire, la machine de Turing et l'architecture de von Neumann (pour la plupart).

Les machines de cette ère se programmaient en

- langage machine pour les toutes premières ;
- assembleur.

La diode à vide

En 1904, Fleming inventa la **diode à vide**.



Elle s'utilise principalement comme **interrupteur** ou comme **amplificateur** de signal électrique.

Ceci mena à l'invention des **tubes à vide**, composants électriques utilisés dans la conception des télévision, postes de radio et les premières machines électriques.



Ils sont encore utilisés aujourd'hui dans des domaines très précis : fours à micro-ondes, amplificateurs audio et radars entre autres.

Le système binaire, la logique et l'électronique

En 1703, Leibniz s'intéressa à la représentation des nombres en **binaire** et à leurs opérations.

Le système binaire, la logique et l'électronique

En 1703, Leibniz s'intéressa à la représentation des nombres en **binaire** et à leurs opérations.

Les bases de la logique étaient connues au temps d'Aristote mais il fallut attendre 1854 pour que Boole formalise la notion de **calcul booléen**.

Celui-ci est formé de deux valeurs, le **faux** (codé par le 0 binaire) et le **vrai** (codé par le 1 binaire) et par diverses opérations :

- le *ou* logique, noté $\vee$;
- le *ou exclusif*, noté $\oplus$;
- le *et* logique, noté $\wedge$;
- la *négation* logique, notée $\neg$.

Le système binaire, la logique et l'électronique

En 1703, Leibniz s'intéressa à la représentation des nombres en **binaire** et à leurs opérations.

Les bases de la logique étaient connues au temps d'Aristote mais il fallut attendre 1854 pour que Boole formalise la notion de **calcul booléen**.

Celui-ci est formé de deux valeurs, le **faux** (codé par le 0 binaire) et le **vrai** (codé par le 1 binaire) et par diverses opérations :

- le *ou* logique, noté $\vee$;
- le *ou exclusif*, noté $\oplus$;
- le *et* logique, noté $\wedge$;
- la *négation* logique, notée $\neg$.

Ces opérations donnèrent lieu aux **portes logiques** dans les machines et constituent les composants de base des unités chargées de réaliser des calculs arithmétique ou logiques.



ou



et



ou excl.



nég.

La machine de Turing — calculabilité

La **machine de Turing** est un concept mathématique qui fut découvert par Turing en 1936.

Son but est de fournir une abstraction des mécanismes de calcul.

La machine de Turing — calculabilité

La **machine de Turing** est un concept mathématique qui fut découvert par Turing en 1936.

Son but est de fournir une abstraction des mécanismes de calcul.

Elle pose la définition de ce qui est **calculable** :

« tout ce qui est effectivement *calculable*
est **calculable** par une machine de Turing ».

La machine de Turing — calculabilité

La **machine de Turing** est un concept mathématique qui fut découvert par Turing en 1936.

Son but est de fournir une abstraction des mécanismes de calcul.

Elle pose la définition de ce qui est **calculable** :

« tout ce qui est effectivement *calculable*
est **calculable** par une machine de Turing ».

Des deux occurrences du mot « calculable »,

- 1 la 1^{re} signifie ce que l'on peut calculer de manière intuitive (ce qu'un être humain peut calculer par un raisonnement) ;

La machine de Turing — calculabilité

La **machine de Turing** est un concept mathématique qui fut découvert par Turing en 1936.

Son but est de fournir une abstraction des mécanismes de calcul.

Elle pose la définition de ce qui est **calculable** :

« tout ce qui est effectivement *calculable*
est **calculable** par une machine de Turing ».

Des deux occurrences du mot « calculable »,

- 1 la 1^{re} signifie ce que l'on peut calculer de manière intuitive (ce qu'un être humain peut calculer par un raisonnement) ;
- 2 la 2^e signifie ce que la machine de Turing peut produire comme résultat à l'issue de l'exécution d'un programme.

La machine de Turing — fonctionnement et complétude

Une machine de Turing travaille sur un ruban (infini) contenant des cases qui peuvent être soit vides ($\cdot$), soit contenir la valeur 1, soit contenir la valeur 0.



Le ruban est la **mémoire** de la machine.

La machine de Turing — fonctionnement et complétude

Une machine de Turing travaille sur un ruban (infini) contenant des cases qui peuvent être soit vides ($\cdot$), soit contenir la valeur 1, soit contenir la valeur 0.



Le ruban est la **mémoire** de la machine.

Une tête de lecture/écriture vient produire un résultat sur le ruban en modifiant le contenu de la case pointée et en se déplaçant d'un pas à gauche ou à droite.



Cette tête de lecture est pilotée par un **programme**.

La machine de Turing — fonctionnement et complétude

Une machine de Turing travaille sur un ruban (infini) contenant des cases qui peuvent être soit vides ($\cdot$), soit contenir la valeur 1, soit contenir la valeur 0.



Le ruban est la **mémoire** de la machine.

Une tête de lecture/écriture vient produire un résultat sur le ruban en modifiant le contenu de la case pointée et en se déplaçant d'un pas à gauche ou à droite.



Cette tête de lecture est pilotée par un **programme**.

Un machine est **Turing-complète** si elle peut calculer tout ce que peut calculer une machine de Turing.

L'architecture de von Neumann — caractéristiques

L'**architecture de von Neumann** est un modèle organisationnel d'ordinateurs décrit par von Neumann en 1945.

L'architecture de von Neumann — caractéristiques

L'**architecture de von Neumann** est un modèle organisationnel d'ordinateurs décrit par von Neumann en 1945.

Quelques caractéristiques importantes :

- une machine de von Neumann possède diverses parties bien distinctes et dédiées à des tâches précises (mémoire, unité de calcul et unité de contrôle) ;

L'architecture de von Neumann — caractéristiques

L'**architecture de von Neumann** est un modèle organisationnel d'ordinateurs décrit par von Neumann en 1945.

Quelques caractéristiques importantes :

- une machine de von Neumann possède diverses parties bien distinctes et dédiées à des tâches précises (mémoire, unité de calcul et unité de contrôle) ;
- le programme à exécuter se situe dans la mémoire interne de la machine (un programme est une donnée comme une autre). Cette caractéristique se nomme **machine à programme enregistré** ;

L'architecture de von Neumann — caractéristiques

L'**architecture de von Neumann** est un modèle organisationnel d'ordinateurs décrit par von Neumann en 1945.

Quelques caractéristiques importantes :

- une machine de von Neumann possède diverses parties bien distinctes et dédiées à des tâches précises (mémoire, unité de calcul et unité de contrôle) ;
- le programme à exécuter se situe dans la mémoire interne de la machine (un programme est une donnée comme une autre). Cette caractéristique se nomme **machine à programme enregistré** ;
- elle est pourvue d'entrées et de sorties qui permettent la saisie et la lecture de données par un humain ou bien par une autre machine.

L'architecture de von Neumann — caractéristiques

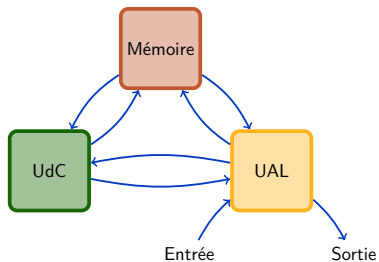
L'**architecture de von Neumann** est un modèle organisationnel d'ordinateurs décrit par von Neumann en 1945.

Quelques caractéristiques importantes :

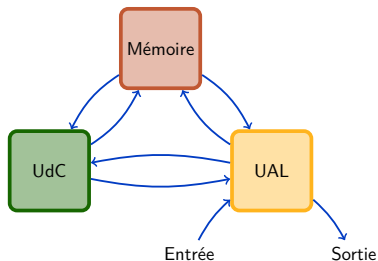
- une machine de von Neumann possède diverses parties bien distinctes et dédiées à des tâches précises (mémoire, unité de calcul et unité de contrôle) ;
- le programme à exécuter se situe dans la mémoire interne de la machine (un programme est une donnée comme une autre). Cette caractéristique se nomme **machine à programme enregistré** ;
- elle est pourvue d'entrées et de sorties qui permettent la saisie et la lecture de données par un humain ou bien par une autre machine.

Encore aujourd'hui la très grande majorité des ordinateurs sont des machines de von Neumann.

L'architecture de von Neumann — organisation

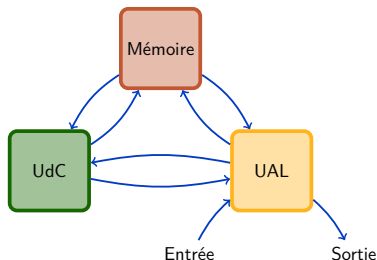


L'architecture de von Neumann — organisation



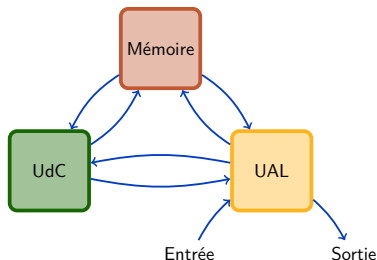
- La **mémoire** contient à la fois des données et le programme en cours d'exécution.

L'architecture de von Neumann — organisation



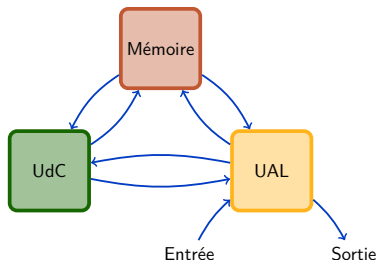
- La **mémoire** contient à la fois des données et le programme en cours d'exécution.
- L'**UdC**, Unité de Contrôle, permet de traiter les instructions contenues dans le programme. Elle est chargée du **séquençage** des opérations.

L'architecture de von Neumann — organisation



- La **mémoire** contient à la fois des données et le programme en cours d'exécution.
- L'**UdC**, Unité de Contrôle, permet de traiter les instructions contenues dans le programme. Elle est chargée du **séquençage** des opérations.
- L'**UAL**, Unité Arithmétique et Logique, permet de réaliser des instructions élémentaires : opérations arithmétiques ($+$, $-$, $\times$, $/$, $\dots$), opérations logiques ($\vee$, $\wedge$, $\oplus$, $\neg$, $\dots$), opérations de comparaison ($=$, $\neq$, $\leq$, $<$, $\dots$).

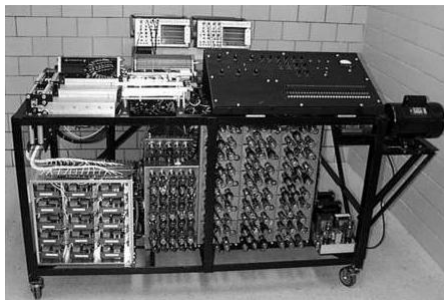
L'architecture de von Neumann — organisation



- La **mémoire** contient à la fois des données et le programme en cours d'exécution.
- L'**UdC**, Unité de Contrôle, permet de traiter les instructions contenues dans le programme. Elle est chargée du **séquençage** des opérations.
- L'**UAL**, Unité Arithmétique et Logique, permet de réaliser des instructions élémentaires : opérations arithmétiques ($+$, $-$, $\times$, $/$, $\dots$), opérations logiques ($\vee$, $\wedge$, $\oplus$, $\neg$, $\dots$), opérations de comparaison ($=$, $\neq$, $\leq$, $<$, $\dots$).
- L'**entrée** permet de recevoir des informations.
- La **sortie** permet d'envoyer des informations.

L'ABC

L'**Atanasoff-Berry Computer** (ABC) fut le premier ordinateur numérique électronique. Il fut construit en 1937 par Atanasoff et Berry.

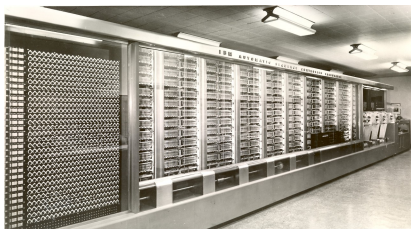


Il représentait les données en binaire et il adoptait une **séparation entre mémoire et unité de calcul**.

Il a été construit pour la résolution de systèmes d'équations linéaires (il pouvait manipuler des systèmes à vingt-neuf équations).

L'ASCC

L'**Automatic Sequence Controlled Calculator** (ASCC), également appelé Harvard Mark I, fut construit par IBM en 1944. Il fut le premier ordinateur a **exécution totalement automatique**.

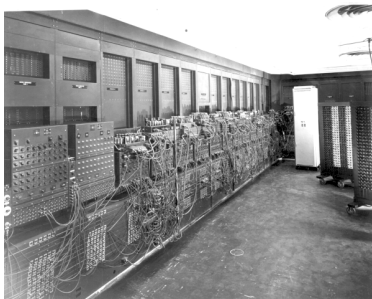


Il pouvait réaliser une multiplication de nombres de vingt-trois chiffres décimaux en six secondes, une division en quinze secondes et des calculs trigonométriques en une minute.

Il ne vérifie pas l'architecture de von Neumann car il fonctionne à cartes perforées (le programme n'est pas une donnée).

L'ENIAC

L'**E**lectronic **N**umerical **I**ntegrator **A**nalysér and **C**omputer (ENIAC) fut achevé en 1946. C'est le premier ordinateur électronique **Turing-complet**.

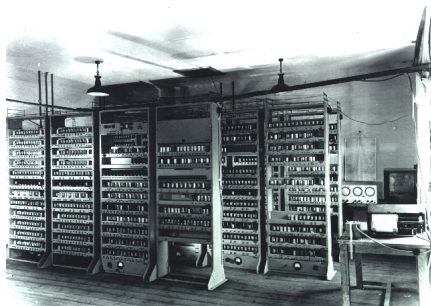


Il pouvait réaliser des multiplications de nombres de dix chiffres décimaux en trois millièmes de seconde.

Il contient 17468 tubes à vide et sa masse est de trente tonnes.

L'EDSAC

L'**Electronic Delay Storage Automatic Calculator** (EDSAC), descendant de l'ENIAC, fut construit en 1949. C'est une **machine de von Neumann**.



Sa mémoire utilisable est organisée en 1024 régions de 17 bits. Une instruction est représentée par un code 5 bits, suivi d'un bit de séparation, de 10 bits d'adresse et d'un bit de contrôle.

Plan

1 Histoire

- Chronologies
- Mécanismes
- Lampes
- Transistors

Le temps des machines à transistors

Cette période débuta dans les années 1950 et se prolonge encore aujourd'hui (en 2016).

Le temps des machines à transistors

Cette période débuta dans les années 1950 et se prolonge encore aujourd'hui (en 2016).

D'un **point de vue matériel**, elle se base sur

- les transistors ;
- les microprocesseurs ;
- les circuits intégrés ;
- les mémoires SRAM et flash.

Le temps des machines à transistors

Cette période débuta dans les années 1950 et se prolonge encore aujourd'hui (en 2016).

D'un **point de vue matériel**, elle se base sur

- les transistors ;
- les microprocesseurs ;
- les circuits intégrés ;
- les mémoires SRAM et flash.

D'un **point de vue théorique**, il y a assez peu de différences par rapport à l'ère précédente. Les machines sont de von Neumann et Turing-complètes. Les principales avancées théoriques sont de nature algorithmique où de nombreuses découvertes ont été réalisées.

Le temps des machines à transistors

Cette période débuta dans les années 1950 et se prolonge encore aujourd'hui (en 2016).

D'un **point de vue matériel**, elle se base sur

- les transistors ;
- les microprocesseurs ;
- les circuits intégrés ;
- les mémoires SRAM et flash.

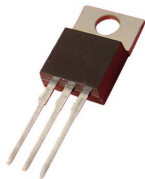
D'un **point de vue théorique**, il y a assez peu de différences par rapport à l'ère précédente. Les machines sont de von Neumann et Turing-complètes. Les principales avancées théoriques sont de nature algorithmique où de nombreuses découvertes ont été réalisées.

L'apparition des premiers langages de programmation est un signe annonciateur de cette nouvelle ère :

- le FORTRAN (Formula Translator) en 1957 ;
- le COBOL (Common Business Oriented Language) en 1959.

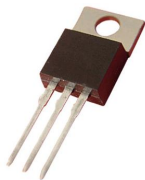
Le transistor

En 1948, Bardeen, Shockley et Brattain inventèrent le **transistor**.



Le transistor

En 1948, Bardeen, Shockley et Brattain inventèrent le **transistor**.

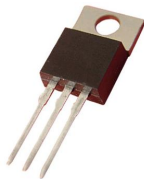


C'est une **version améliorée du tube à vide** :

- élément moins volumineux et plus solide ;
- fonctionne sur de faibles tensions ;
- pas de préchauffage requis.

Le transistor

En 1948, Bardeen, Shockley et Brattain inventèrent le **transistor**.



C'est une **version améliorée du tube à vide** :

- élément moins volumineux et plus solide ;
- fonctionne sur de faibles tensions ;
- pas de préchauffage requis.

Ils peuvent être miniaturisés au point de pouvoir être assemblés en très grand nombre (plusieurs milliards) dans un très petit volume.

L'IBM 1401

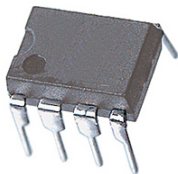
L'**IBM 1401** fut fabriqué de 1959 à 1965. Il fut l'une des machines à transistor les plus vendues de son époque.



Il pouvait réaliser 193000 additions de nombres de huit chiffres décimaux par seconde et disposait d'une mémoire d'environ 8 Kio.

Le circuit intégré

En 1958, Kilby inventa le **circuit intégré**.



C'est intuitivement un composant obtenu en connectant d'une certaine manière des transistors entre eux.

Son rôle est donc de regrouper dans un espace très réduit un très grand nombre de composants électroniques (transistors, portes logiques, etc.).

L'IBM 360

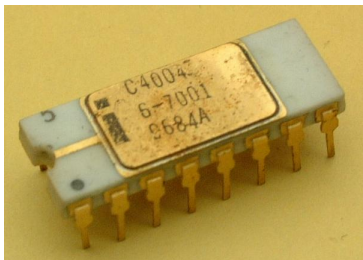
L'**IBM 360** fut commercialisé dès 1966 et fut l'une des premières machines à circuits intégrés les plus célèbres de son époque.



Il pouvait accueillir jusqu'à 8 Mio de mémoire.

Le microprocesseur

Le premier **microprocesseur** fut conçu en 1971. Il s'agit de l'INTEL 4004.



Il contenait 2300 transistors et sa fréquence était de 740 KHz.

Il fournissait une puissance équivalente à l'ENIAC.

Comparaison de quelques réalisations

Machine	Date	Fonctionnement	Base	T.-complet
Mach. analytique	1834	Mécanique	Décimal	Oui (!)
ABC	1937	Électronique	Binaire	Non
ASCC	1944	Électromécanique	Décimal	Non
ENIAC	1946	Électronique	Décimal	Oui
IBM 360	1966	Électronique	Binaire	Oui

Vers les ordinateurs d'aujourd'hui — matériel

Depuis l'invention du microprocesseur, les machines ont subi de nombreuses **évolutions matérielles** :

- **miniaturisation** de plus en plus poussée. Apparition des ordinateurs portables dans les années 1980, des PDA dans les années 1990, des smartphones et des tablettes dans les années 2010 ;

Vers les ordinateurs d'aujourd'hui — matériel

Depuis l'invention du microprocesseur, les machines ont subi de nombreuses **évolutions matérielles** :

- **miniaturisation** de plus en plus poussée. Apparition des ordinateurs portables dans les années 1980, des PDA dans les années 1990, des smartphones et des tablettes dans les années 2010 ;
- meilleure **gestion de l'énergie**. La consommation énergétique d'un composant est une problématique importante. Les batteries offrent des capacités (mesurées en ampères-heure A.h) de plus en plus grandes et les composants électroniques consomment de moins en moins ;

Vers les ordinateurs d'aujourd'hui — matériel

Depuis l'invention du microprocesseur, les machines ont subi de nombreuses **évolutions matérielles** :

- **miniaturisation** de plus en plus poussée. Apparition des ordinateurs portables dans les années 1980, des PDA dans les années 1990, des smartphones et des tablettes dans les années 2010 ;
- meilleure **gestion de l'énergie**. La consommation énergétique d'un composant est une problématique importante. Les batteries offrent des capacités (mesurées en ampères-heure A.h) de plus en plus grandes et les composants électroniques consomment de moins en moins ;
- plus grande **puissance de calcul** ;

Vers les ordinateurs d'aujourd'hui — matériel

Depuis l'invention du microprocesseur, les machines ont subi de nombreuses **évolutions matérielles** :

- **miniaturisation** de plus en plus poussée. Apparition des ordinateurs portables dans les années 1980, des PDA dans les années 1990, des smartphones et des tablettes dans les années 2010 ;
- meilleure **gestion de l'énergie**. La consommation énergétique d'un composant est une problématique importante. Les batteries offrent des capacités (mesurées en ampères-heure A.h) de plus en plus grandes et les composants électroniques consomment de moins en moins ;
- plus grande **puissance de calcul** ;
- augmentation des quantités de **mémoire**. Ceci concerne les mémoires de stockage ainsi que les mémoires de travail ;

Vers les ordinateurs d'aujourd'hui — matériel

Depuis l'invention du microprocesseur, les machines ont subi de nombreuses **évolutions matérielles** :

- **miniaturisation** de plus en plus poussée. Apparition des ordinateurs portables dans les années 1980, des PDA dans les années 1990, des smartphones et des tablettes dans les années 2010 ;
- meilleure **gestion de l'énergie**. La consommation énergétique d'un composant est une problématique importante. Les batteries offrent des capacités (mesurées en ampères-heure A.h) de plus en plus grandes et les composants électroniques consomment de moins en moins ;
- plus grande **puissance de calcul** ;
- augmentation des quantités de **mémoire**. Ceci concerne les mémoires de stockage ainsi que les mémoires de travail ;
- meilleure **fiabilité**.

Vers les ordinateurs d'aujourd'hui — théorie

En parallèle, de nouvelles **connaissances théoriques** viennent se joindre à ces avancées :

- nouveaux **algorithmes**. De nombreuses découvertes théoriques améliorent, p.ex., l'efficacité des algorithmes de recherche de données, des calculs de chemins dans les graphes et des calculs algébriques pour la cryptographie ;

Vers les ordinateurs d'aujourd'hui — théorie

En parallèle, de nouvelles **connaissances théoriques** viennent se joindre à ces avancées :

- nouveaux **algorithmes**. De nombreuses découvertes théoriques améliorent, p.ex., l'efficacité des algorithmes de recherche de données, des calculs de chemins dans les graphes et des calculs algébriques pour la cryptographie ;
- nouveaux **langages de programmation** et paradigmes de programmation. Les langages deviennent de plus en plus abstraits (éloignés des considérations matérielles), ce qui permet de programmer des choses de plus en plus complexes et des projets de plus en plus conséquents ;

Vers les ordinateurs d'aujourd'hui — théorie

En parallèle, de nouvelles **connaissances théoriques** viennent se joindre à ces avancées :

- nouveaux **algorithmes**. De nombreuses découvertes théoriques améliorent, p.ex., l'efficacité des algorithmes de recherche de données, des calculs de chemins dans les graphes et des calculs algébriques pour la cryptographie ;
- nouveaux **langages de programmation** et paradigmes de programmation. Les langages deviennent de plus en plus abstraits (éloignés des considérations matérielles), ce qui permet de programmer des choses de plus en plus complexes et des projets de plus en plus conséquents ;
- apparition du **calcul parallèle** ;

Vers les ordinateurs d'aujourd'hui — théorie

En parallèle, de nouvelles **connaissances théoriques** viennent se joindre à ces avancées :

- nouveaux **algorithmes**. De nombreuses découvertes théoriques améliorent, p.ex., l'efficacité des algorithmes de recherche de données, des calculs de chemins dans les graphes et des calculs algébriques pour la cryptographie ;
- nouveaux **langages de programmation** et paradigmes de programmation. Les langages deviennent de plus en plus abstraits (éloignés des considérations matérielles), ce qui permet de programmer des choses de plus en plus complexes et des projets de plus en plus conséquents ;
- apparition du **calcul parallèle** ;
- apparition des **réseaux** ;

Vers les ordinateurs d'aujourd'hui — théorie

En parallèle, de nouvelles **connaissances théoriques** viennent se joindre à ces avancées :

- nouveaux **algorithmes**. De nombreuses découvertes théoriques améliorent, p.ex., l'efficacité des algorithmes de recherche de données, des calculs de chemins dans les graphes et des calculs algébriques pour la cryptographie ;
- nouveaux **langages de programmation** et paradigmes de programmation. Les langages deviennent de plus en plus abstraits (éloignés des considérations matérielles), ce qui permet de programmer des choses de plus en plus complexes et des projets de plus en plus conséquents ;
- apparition du **calcul parallèle** ;
- apparition des **réseaux** ;
- apparition des **machines virtuelles**.

Plan

2 Représentation

- Bits
- Entiers
- Réels
- Caractères

Plan

2 Représentation

- Bits
- Entiers
- Réels
- Caractères

Pour qu'un ordinateur puisse traiter des données, il est nécessaire de les **représenter** de sorte qu'il puisse les « comprendre » et les manipuler.

Pour qu'un ordinateur puisse traiter des données, il est nécessaire de les **représenter** de sorte qu'il puisse les « comprendre » et les manipuler.

Il est facile de représenter électroniquement deux états :

- **l'état** 0, incarné par l'absence de courant électrique ;
- **l'état** 1, incarné par la présence de courant électrique.

Pour qu'un ordinateur puisse traiter des données, il est nécessaire de les **représenter** de sorte qu'il puisse les « comprendre » et les manipuler.

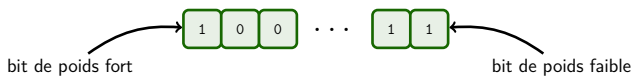
Il est facile de représenter électroniquement deux états :

- **l'état** 0, incarné par l'absence de courant électrique ;
- **l'état** 1, incarné par la présence de courant électrique.

On appelle **bit** l'unité de base d'information, à valeur dans l'ensemble $\{0, 1\}$, symbolisant l'état 0 ou l'état 1.

Suites de bits

Dans une suite de bits, on distingue deux bits particuliers :



Suites de bits

Dans une suite de bits, on distingue deux bits particuliers :



Chaque bit d'une suite de n bits est implicitement indicé de 0 pour le bit de poids faible à $n - 1$ pour le bit de poids fort :



Suites de bits

Dans une suite de bits, on distingue deux bits particuliers :



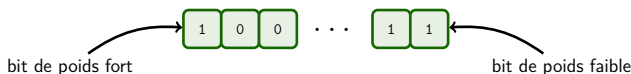
Chaque bit d'une suite de n bits est implicitement indicé de 0 pour le bit de poids faible à $n - 1$ pour le bit de poids fort :



Un bit d'indice plus grand (resp. petit) qu'un autre est dit de « poids plus fort » (resp. « poids plus faible »).

Suites de bits

Dans une suite de bits, on distingue deux bits particuliers :



Chaque bit d'une suite de n bits est implicitement indicé de 0 pour le bit de poids faible à $n - 1$ pour le bit de poids fort :



Un bit d'indice plus grand (resp. petit) qu'un autre est dit de « poids plus fort » (resp. « poids plus faible »).

Dans un ordinateur, les informations sont représentées par des **suites finies de bits**. On parle de **représentation sur n bits** lorsque l'on fixe le nombre de bits n pour représenter une donnée.

Représenter des informations

Pour qu'une suite de bits représente de l'information, il faut savoir comment l'**interpréter**. Une interprétation s'appelle un **codage**.

On placera, si possible pour chaque suite de symboles, le nom de son codage en indice.

Représenter des informations

Pour qu'une suite de bits représente de l'information, il faut savoir comment l'**interpréter**. Une interprétation s'appelle un **codage**.

On placera, si possible pour chaque suite de symboles, le nom de son codage en indice.

Les principales informations que l'on souhaite représenter sont

- les entiers (positifs ou négatifs) ;
- les nombres réels ;
- les caractères ;
- les textes ;
- les instructions (d'un programme).

Représenter des informations

Pour qu'une suite de bits représente de l'information, il faut savoir comment l'**interpréter**. Une interprétation s'appelle un **codage**.

On placera, si possible pour chaque suite de symboles, le nom de son codage en indice.

Les principales informations que l'on souhaite représenter sont

- les entiers (positifs ou négatifs) ;
- les nombres réels ;
- les caractères ;
- les textes ;
- les instructions (d'un programme).

Pour chacun de ces points, il existe beaucoup de codages différents.

Leur raison d'être est que, selon la situation, un codage peut s'avérer meilleur qu'un autre (plus simple, plus économique, plus efficace).

Types de données fondamentaux

Type	Taille
bit	1 bit
octet (byte)	8 bits
mot (word)	16 bits
double mot (dword)	32 bits
quadruple mot (qword)	64 bits

Types de données fondamentaux

Type	Taille
bit	1 bit
octet (byte)	8 bits
mot (word)	16 bits
double mot (dword)	32 bits
quadruple mot (qword)	64 bits
kibioctet (Kio)	$2^{10} = 1024$ octets
mébioctet (Mio)	$2^{20} = 1048576$ octets
gibioctet (Gio)	$2^{30} = 1073741824$ octets

Types de données fondamentaux

Type	Taille
bit	1 bit
octet (byte)	8 bits
mot (word)	16 bits
double mot (dword)	32 bits
quadruple mot (qword)	64 bits
kibioctet (Kio)	$2^{10} = 1024$ octets
mébioctet (Mio)	$2^{20} = 1048576$ octets
gibioctet (Gio)	$2^{30} = 1073741824$ octets
kilooctet (Ko)	$10^3 = 1000$ octets
mégaoctet (Mo)	$10^6 = 1000000$ octets
gigaoctet (Go)	$10^9 = 1000000000$ octets

Types de données fondamentaux

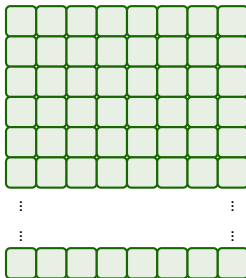
Type	Taille
bit	1 bit
octet (byte)	8 bits
mot (word)	16 bits
double mot (dword)	32 bits
quadruple mot (qword)	64 bits
kibioctet (Kio)	$2^{10} = 1024$ octets
mébioctet (Mio)	$2^{20} = 1048576$ octets
gibioctet (Gio)	$2^{30} = 1073741824$ octets
kilooctet (Ko)	$10^3 = 1000$ octets
mégaoctet (Mo)	$10^6 = 1000000$ octets
gigaoctet (Go)	$10^9 = 1000000000$ octets

On utilisera de préférence les unités Kio, Mio et Gio à la place de Ko, Mo et Go.

Mémoire d'une machine

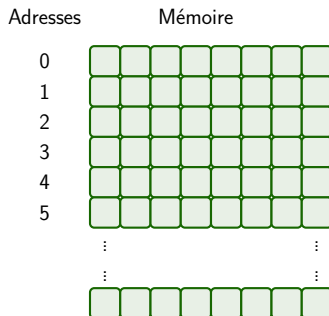
La **mémoire** d'un ordinateur est un tableau dont chaque case contient un octet.

Mémoire



Mémoire d'une machine

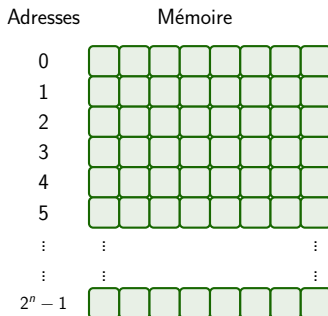
La **mémoire** d'un ordinateur est un tableau dont chaque case contient un octet.



Chaque octet de la mémoire est repéré par son **adresse**. C'est sa position dans le tableau.

Mémoire d'une machine

La **mémoire** d'un ordinateur est un tableau dont chaque case contient un octet.



Chaque octet de la mémoire est repéré par son **adresse**. C'est sa position dans le tableau.

Sur un **système n bits**, l'adressage va de 0 à $2^n - 1$ au maximum.

La structure octet par octet de la mémoire donne lieu au problème suivant : comment organiser en mémoire une suite de bits u constituée de plus de huit bits ?

Boutisme

La structure octet par octet de la mémoire donne lieu au problème suivant : comment organiser en mémoire une suite de bits u constituée de plus de huit bits ?

Il existe pour cela deux conventions : le **petit boutisme** (*little-endian*) et le **grand boutisme** (*big-endian*). Les architectures qui nous intéressent sont en **petit boutisme**.

Boutisme

La structure octet par octet de la mémoire donne lieu au problème suivant : comment organiser en mémoire une suite de bits u constituée de plus de huit bits ?

Il existe pour cela deux conventions : le **petit boutisme** (*little-endian*) et le **grand boutisme** (*big-endian*). Les architectures qui nous intéressent sont en **petit boutisme**.

L'organisation se fait en trois étapes :

- 1 si u n'est pas constitué d'un nombre de bits multiple de huit, on ajoute des 0 à sa **gauche** de sorte qu'il le devienne. On appelle u' cette nouvelle suite ;

Boutisme

La structure octet par octet de la mémoire donne lieu au problème suivant : comment organiser en mémoire une suite de bits u constituée de plus de huit bits ?

Il existe pour cela deux conventions : le **petit boutisme** (*little-endian*) et le **grand boutisme** (*big-endian*). Les architectures qui nous intéressent sont en **petit boutisme**.

L'organisation se fait en trois étapes :

- 1 si u n'est pas constitué d'un nombre de bits multiple de huit, on ajoute des 0 à sa **gauche** de sorte qu'il le devienne. On appelle u' cette nouvelle suite ;
- 2 on découpe u' en k morceaux de huit bits

$$u' = u'_{k-1} \dots u'_1 u'_0 ;$$

Boutisme

La structure octet par octet de la mémoire donne lieu au problème suivant : comment organiser en mémoire une suite de bits u constituée de plus de huit bits ?

Il existe pour cela deux conventions : le **petit boutisme** (*little-endian*) et le **grand boutisme** (*big-endian*). Les architectures qui nous intéressent sont en **petit boutisme**.

L'organisation se fait en trois étapes :

- 1 si u n'est pas constitué d'un nombre de bits multiple de huit, on ajoute des 0 à sa **gauche** de sorte qu'il le devienne. On appelle u' cette nouvelle suite ;
- 2 on découpe u' en k morceaux de huit bits

$$u' = u'_{k-1} \dots u'_1 u'_0 ;$$

- 3 en **petit** (resp. **grand**) boutisme, les morceaux sont placés, des petites aux grandes adresses, de u'_0 à u'_{k-1} (resp. u'_{k-1} à u'_0) dans la mémoire.

P.ex., pour

$u := 011110000000101110111,$

Boutisme

P.ex., pour

$u := 011110000000101110111,$

on a

$u' = 000011110000000101110111$

Boutisme

P.ex., pour

$$u := 011110000000101110111,$$

on a

$$u' = 000011110000000101110111$$

et

$$u'_2 = 00001111, \quad u'_1 = 00000001, \quad u'_0 = 01110111.$$

Boutisme

P.ex., pour

$$u := 011110000000101110111,$$

on a

$$u' = 000011110000000101110111$$

et

$$u'_2 = 00001111, \quad u'_1 = 00000001, \quad u'_0 = 01110111.$$

Selon les deux conventions, le placement de u à l'adresse adr en mémoire donne lieu à

Adresses	Mémoire							
	⋮							⋮
$\text{adr} + 0$	0	1	1	1	0	1	1	1
$\text{adr} + 1$	0	0	0	0	0	0	0	1
$\text{adr} + 2$	0	0	0	0	1	1	1	1
	⋮							⋮

en petit boutisme

Adresses	Mémoire							
	⋮							⋮
$\text{adr} + 0$	0	0	0	0	1	1	1	1
$\text{adr} + 1$	0	0	0	0	0	0	0	1
$\text{adr} + 2$	0	1	1	1	0	1	1	1
	⋮							⋮

en grand boutisme