Architecture des ordinateurs

Cours 1

17 septembre 2012

Avant de commencer

- contact : carine.pivoteau@univ-mlv.fr
- page web du cours :
  http://http://www-igm.univ-mlv.fr/~pivoteau/ARCHI/
  - planning, dates d'Exam
  - transparents de cours, sujets de TD
  - bibliographie et lien utiles
- 7 séances de cours (toutes les 2 semaines)
  - lundi, 10h30 - 12h30
- 14 séances de TD (dont 5 de TP)
  - mercredi après-midi
- contrôle des connaissances :
  - contrôle continu (interros, TPs)
  - 1 exam (à la moitié du cours)
  - 1 exam (mi-janvier)

But du cours

Culture	comprendre les grands principes de fonctionnement d'un ordinateur
Technique	manipuler des concepts basiques récurrents en informatique
Informatique	acquérir une connaissance "bas niveau" de la programmation

Introduction

E. Dijkstra (mathématicien et informaticien néerlandais) :
“ La science informatique n'est pas plus la science des ordinateurs que l'astronomie n'est celle des télescopes ”

Informatique : contraction d'information et automatique

▷ traitement automatique de l'information par des machines

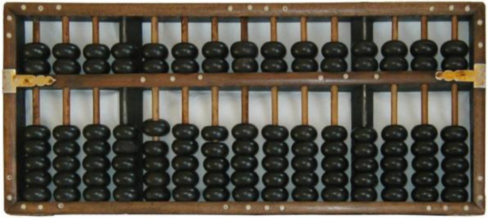
Architecture des ordinateurs : domaine de l'informatique centré sur les machines.

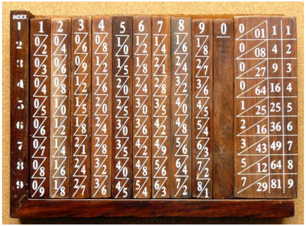
▷ point de vue à la fois matériel et logiciel

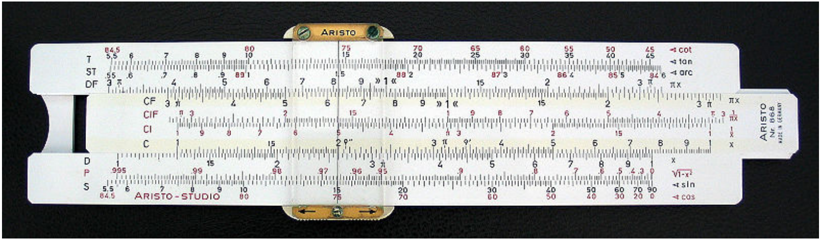
Et pour commencer, un peu d'histoire...

Préhistoire : les abaques Avant 1600

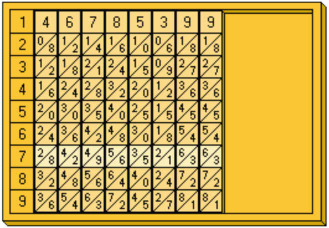
Abaque : instrument mécanique facilitant le calcul.
Exemple : boulier, bâtons de Neper, règle à calculer, ...

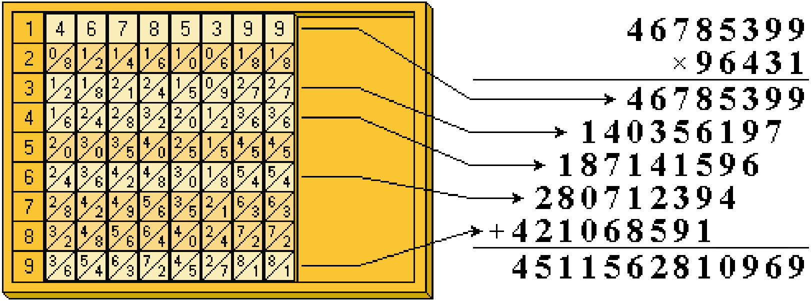






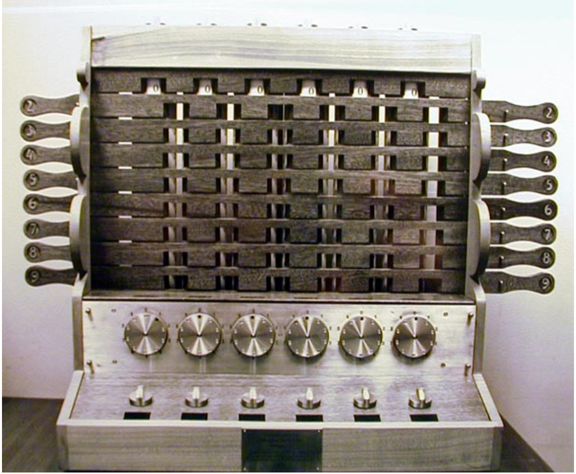
Multiplier avec des bâtons de Neper





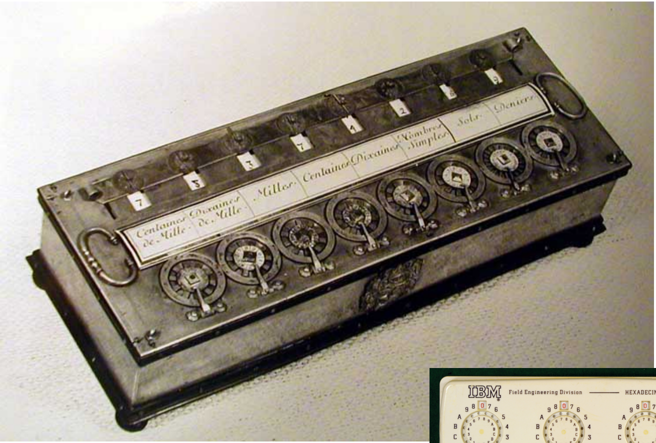
Préhistoire 1623

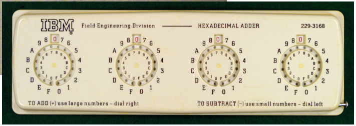
Wilhelm Schickard construit une machine à calculer mécanique en appliquant les idées de Neper (règle à calculer, 1614).



Préhistoire 1642

Pascal présente une machine qui additionne et soustrait les nombres de 6 chiffres en base 10 : la Pascaline.

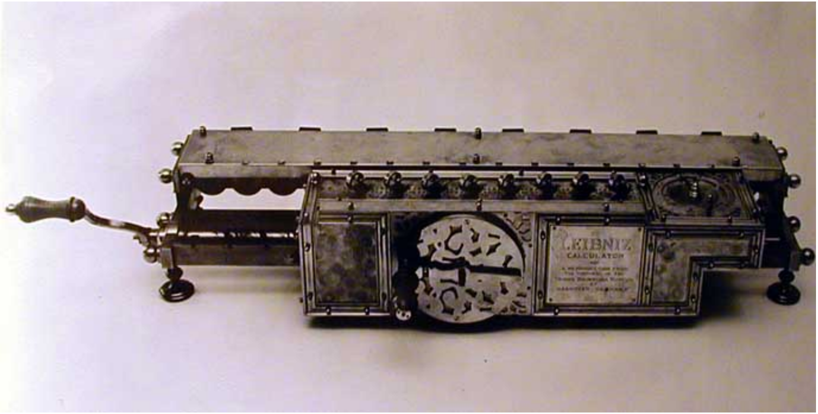




Multiplication : additions en chaîne.

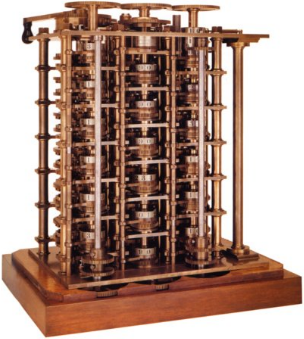
Préhistoire 1672

Leibnitz améliore la Pascaline : un chariot mobile permet de faire les multiplications et les divisions automatiquement.

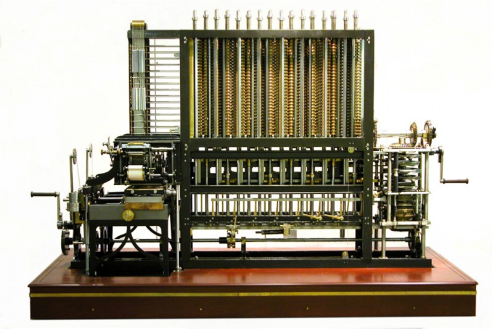


XIX^{e} siècle 1822

Charles Babbage conçoit la machine à différences pour calculer des tables numériques (pour la navigation).



Morceau de la première machine à différences



Machine à Différences n°2 1854, Babbage et Scheutz.

XIX^e siècle 1805

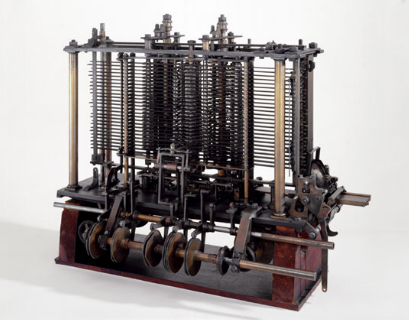
Joseph Jacquard (d'après des idées de Falcon en 1728) : cartes perforées pour métiers à tisser.

XIX^{e} siècle 1805

Joseph Jacquard (d'après des idées de Falcon en 1728) : cartes perforées pour métiers à tisser. C'est le premier programme !

XIX^{e} siècle 1833

Babbage conçoit ensuite une machine programmable capable de réaliser différentes opérations codées sur des *cartes perforées* : la Machine Analytique.



- un dispositif d'entrée et de sortie ;
- un organe de commande gérant le transfert des nombres et leur mise en ordre pour le traitement ;
- un magasin permettant de stocker les résultats intermédiaires ou finaux (mémoire) ;
- un moulin chargé d'exécuter les opérations sur les nombres ;
- un dispositif d'impression.

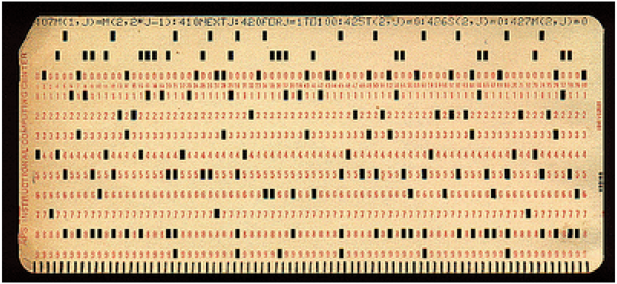
Programmes d'Ada Augusta. Irréalisable à l'époque.

XIX^{e}

1854 - Georges Boole - *Une étude des lois de la pensée*

Fondements mathématiques de la logique.
L'électronique actuelle repose sur l'algèbre de Boole.

1890 : Calculateur statistique d'Hermann Hollerith pour les recensements. Il fonde la *Tabulating Machine Company* en 1896 qui devient IBM en 1908.



XX^e années 30

1936 - Alan Turing - *Machines de Turing*

Ce que l'on peut calculer et ce que l'on ne peut pas.

1938 - Claude Shannon - *Théorie de l'information*

Tout peut être représenté par des 0 et des 1 : c'est la numérisation.

Première génération d'ordinateurs (1936-1956)

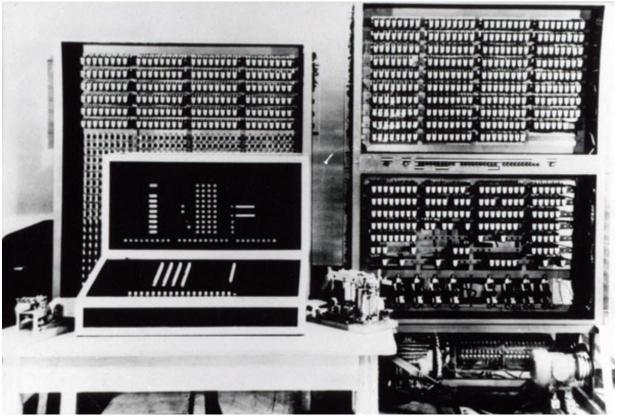
Des calculateurs programmables aux premiers ordinateurs.

Composants : relais, tubes à vides, résistances.

Logiciels : langage machine seulement.

1$^{\text{ère}}$ génération fin des années 30

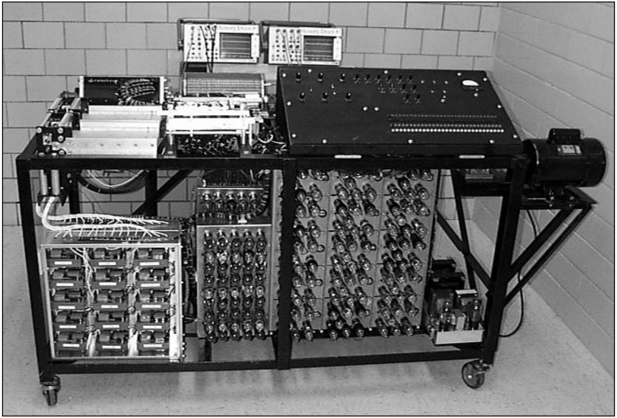
Indépendement : Konrad Zuse, John Atanasoff, Georges Stibitz travaillent à la conception de machines binaires.



Machine à calculer électromécaniques de Zuse (photo : Z3)

$1^{\text{ère}}$ génération fin des années 30

ABC (Atanasoff-Berry Computer) : officiellement, le premier ordinateur numérique électronique



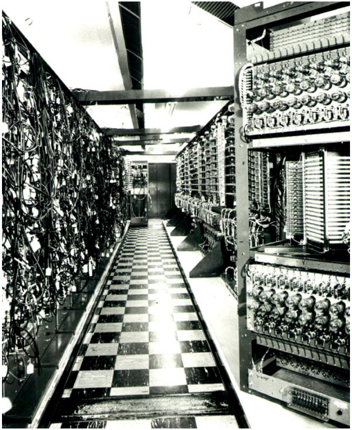
1ère génération 1944

Howard Aiken, machine électromécanique (Harvard Mark 1) :

- Multiplication de nombres de 23 chiffres en 6 secondes ;
- Addition en 3 dixièmes de seconde.

1$^{\text{ère}}$ génération durant la guerre

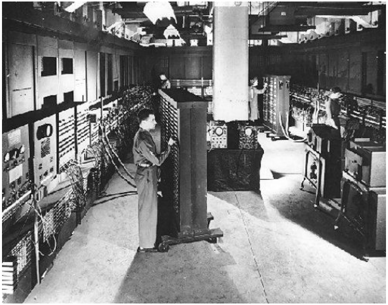
Le Colossus Mark II est utilisé pour la cryptanalyse. Il est constitué de 2 400 tubes à vide et réalise 5 000 opérations par seconde.

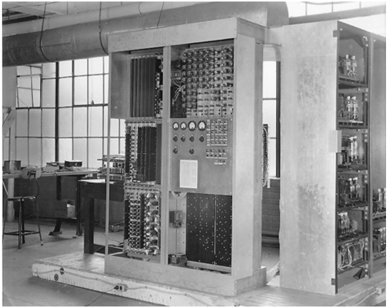


$1^{\text{ère}}$ génération 1945

John Eckert et John Mauchly construisent l'*ENIAC* (Electronic Numerical Integrator And Calculator) : 18 000 tubes, 30 tonnes. Multiplication de nombres de 10 chiffres en 3ms.

▽



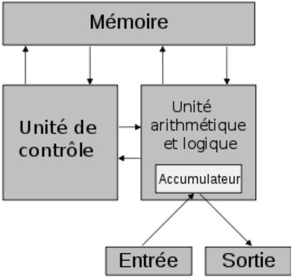


△

Von Neumann (avec Eckert et Mauchly) propose l'*EDVAC* (Electronic Discrete Variable Automatic Computer). Système binaire contrairement à l'ENIAC. Mis en service en 1951.

Architecture de von Neumann

- Machine *universelle* programmée
- Instructions numériques stockées en mémoire
- Instructions modifiables exécutées en séquence avec des ruptures de séquence (sauts).

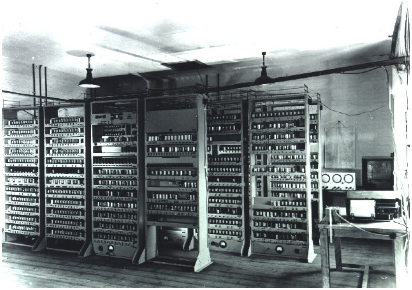


- **unité arithmétique et logique (UAL)** qui effectue les opérations de base,
- **unité de contrôle** qui est chargée du séquençage des opérations,
- **mémoire** qui contient à la fois les données et le programme,
- dispositifs d'**entrée-sortie** qui permettent de communiquer avec le monde extérieur.

Actuellement, la plupart des ordinateurs sont des machines de Von Neumann, seules les technologies ont changé.

1$^{\text{ère}}$ génération début des années 50

Maurice Wilkes construit l'*EDSAC* (Electronic Delay Storage Automatic Calculator). 5 000 opérations mathématiques dont 4 000 multiplications en 1 minute (en 1949).



Eckert et Mauchly construisent l'*UNIVAC* (UNIVersal Automatic Computer). 5 200 tubes à vide, 13 tonnes, 1905 opérations par seconde. Données stockées sur une bande magnétique (1951).

$1^{\text{ère}}$ génération 1953

IBM lance le 701 (19 exemplaires). Mémoire à tubes cathodiques.
16 000 additions ou 2 200 multiplications par seconde.

Peu de temps après, le 650 est lancé à 2 000 exemplaires.

1951 - M. V. Wilkes - *la microprogrammation*

L'unité centrale d'un ordinateur peut être contrôlée par un programme informatique spécialisé stocké en mémoire.

$1^{\text{ère}}$ génération synthèse

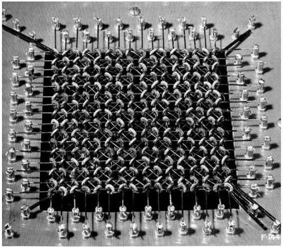
	composants	mémoire	opérations
Z3	2 600 relais	64×22 bits	4 add/s. 15 mul/min.
ABC	280 tubes	60×50 bits	30 add/s.
HM I	765 000 comp.	72×23 (déc.)	add : 3 ds. mul : 6 s.
CM II	2 400 tubes		
ENIAC	17 468 tubes	20 chiffres signés	5 000 add/s. 350 mul/s.
EDVAC	6 000 tubes	1000×44 bits	add : 0.8 ms. mul : 2.9 ms.
EDSAC	3 000 tubes		4000 mul/min.
UNIVAC	5 200 tubes	1000×11 déc.	add : 0.5 ms. mul : 0.21 ms.
IBM 650		60×10 déc. ferrite	add : 1.63 ms mul : 12.96 ms

Deuxième génération d'ordinateurs (1956-1963)

1948, John Bardeen, Walter Brattain et William Shockley découvrent *le transistor*.

Composants :	transistors, mémoire à tores de ferrite, imprimantes, bandes magnétiques.
Logiciels :	apparition des systèmes d'exploitation, langages évolués FORmula TRANSlator (1957) COmmon Business Oriented Language (1959)

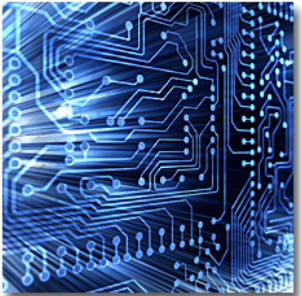
Apparition de l'industrie, IBM, DEC, HP, etc.



Troisième génération d'ordinateurs (1963-1971)

En 1958, Jack Kilby (Texas Inst.) crée le premier *circuit intégré*.

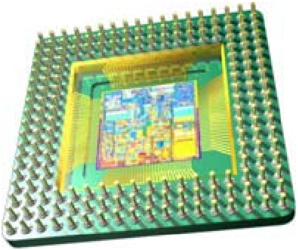
Composants :	circuits intégrés,
Machine :	faible consommation énergétique, fiable, encombrement réduit.
Évolution :	multiprocesseur, temps partagé, accès interactif, apparition des réseaux, premiers problèmes de compatibilité entre machines.



Quatrième génération d'ordinateurs (1971-)

Miniaturisation des circuits : l'ère de la *micro-informatique*

Composants :	Very Large Scale Integration premier microprocesseur INTEL 4004 (1971)
Logiciel :	traitement distribué, machine virtuelle réseau, base de données convivialité
Évolution :	parallélisme d'exécution ordinateurs personnels augmentation en puissance



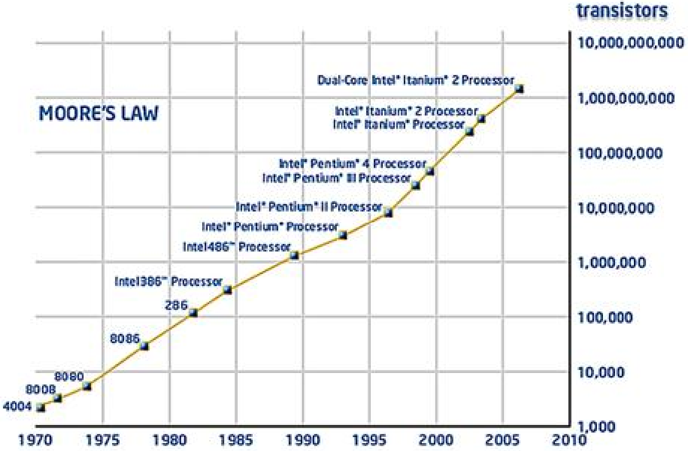
Résumé

avant 1600	les abaques...
XVII^{e} siècle	machines à calculer mécaniques
XIX^{e} siècle	machines à calculer programmables
1936-1956	1ère génération : des calculateurs programmables aux ordinateurs
1956-1963	2ème génération : apparition des transistors
1963-1971	3ème génération : apparition des circuit intégrés
depuis 1971	4ème génération : la micro-informatique

Conclusion

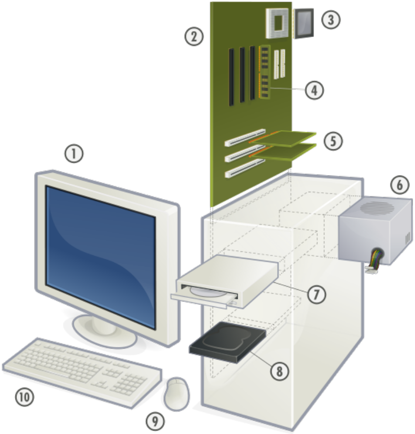
Innovations technologiques supportées par des bases théoriques.

Loi de Moore (1965) : le nombre de transistors sur une puce double tous les 2 ans.



Architecture en couches

L'ordinateur : point de vue externe



1. Écran
2. Carte mère
3. CPU (Microprocesseur)
4. Mémoire vive (RAM)
5. Cartes de périphériques
6. Alimentation
7. Lecteur de disques (ex. DVD)
8. Disque dur
9. Souris
10. Clavier

L'ordinateur : point de vue interne

- Machine électronique binaire
- Fonctionnement des composants de base : circuits électroniques
- Organisation et communication entre les composants
- Langage de la machine
- Système d'exploitation :
  - programme principal de l'ordinateur
  - exécution simultanée d'autres programmes
  - gestion des périphériques : entrées-sorties, stockage, ...

Machines multi-niveaux

5. Langages haut niveau

Compilation

4. Langage d'assemblage

Assembleur

3. Système d'exploitation

Appels système

2. Jeu d'instructions propre à chaque machine

Microprogrammes : micro-instructions binaires

1. Micro-architecture (UAL, opérations, registres, ...)

Assemblage physique des portes logiques

0. Circuits logiques

La couche circuits logiques

Circuit logique

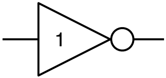
▷ Un circuit logique est un circuit dans lequel seules 2 valeurs logiques sont possibles : 0 ou 1.

▷ En pratique : circuit électrique (transistors) dans lequel une faible tension représente le signal 0 alors qu'une tension élevée correspond au signal 1.

▷ Composants de base : les portes logiques qui permettent de combiner ces signaux binaires.

Une porte logique est un composant qui reçoit en entrée une ou plusieurs valeurs binaires (souvent 2) et renvoie en sortie une unique valeur binaire.

porte NON



Si la valeur d'entrée est 1 alors la sortie vaut 0.
Si la valeur d'entrée est 0 alors la sortie vaut 1.

Fonctions logiques et tables de vérité

Fonction logique : fonction d'une ou plusieurs variables booléennes (qui peut prendre la valeur 0 ou 1) dans $\{0, 1\}$.

Exemples :

1. $f_0(a) = \begin{cases} 0 & \text{si } a = 1 \\ 1 & \text{si } a = 0 \end{cases}$
2. $f_1(a, b) = a$
3. $f_2(a, b, c) = \begin{cases} c & \text{si } a = 0 \\ 1 - c & \text{si } a = 1 \end{cases}$

Tables de vérité

fonction f_0

a	S
0	1
1	0

fonction f_1

a	b	S
0	0	0
0	1	0
1	0	1
1	1	1

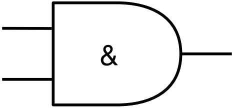
fonction f_2

a	b	c	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Voir l'algèbre booléenne (cours de Structures Discrètes)

Portes ET et OU

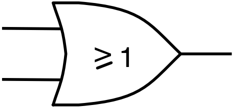
porte ET



a	b	S
0	0	0
0	1	0
1	0	0
1	1	1

$$S = f(a,b) = a \cdot b$$

porte OU

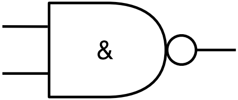


a	b	S
0	0	0
0	1	1
1	0	1
1	1	1

$$f(a,b) = a + b$$

Portes NON-ET et NON-OU

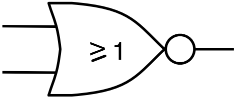
porte NON-ET



a	b	S
0	0	1
0	1	1
1	0	1
1	1	0

$$f(a,b) = \overline{a \cdot b}$$

porte NON-OU



a	b	S
0	0	1
0	1	0
1	0	0
1	1	0

$$f(a,b) = \overline{a + b}$$

Résumé

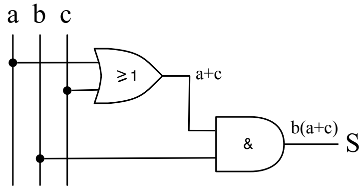
Les 16 fonctions booléennes de 2 variables ($\{0,1\}^2 \to \{0,1\}$)

00	01	10	11	$f(a,b)$
0	0	0	0	0
0	0	0	1	ab
0	0	1	0	$a\bar{b}$
0	0	1	1	a
0	1	0	0	$\bar{a}b$
0	1	0	1	b
0	1	1	0	$a \oplus b$
0	1	1	1	$a + b$

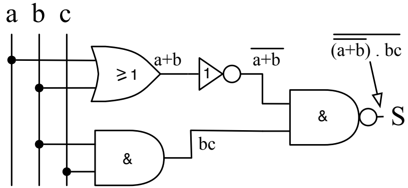
00	01	10	11	$f(a,b)$
1	0	0	0	$\overline{a+b}$
1	0	0	1	$\overline{a \oplus b}$
1	0	1	0	$\bar{b}$
1	0	1	1	$a + \bar{b}$
1	1	0	0	$\bar{a}$
1	1	0	1	$\bar{a} + b$
1	1	1	0	$\overline{ab}$
1	1	1	1	1

Avec 3 variables, combien de fonctions ?

Du circuit à la table de vérité



a	b	c	$a+c$	$b(a+c)$
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	1	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	1
1	1	1	1	1



a	b	c	$a+b$	$\overline{a+b}$	bc	S
0	0	0	0	1	0	1
0	0	1	0	1	0	1
0	1	0	1	0	0	1
0	1	1	1	0	1	1
1	0	0	1	0	0	1
1	0	1	1	0	0	1
1	1	0	1	0	0	1
1	1	1	1	0	1	1

Règles de calcul

Constantes	$a + 0 = a$	$a \cdot 0 = 0$
	$a + 1 = 1$	$a \cdot 1 = a$
Idempotence	$a + a = a$	$a \cdot a = a$
Complémentation	$a + \overline{a} = 1$	$a \cdot \overline{a} = 0$
Commutativité	$a + b = b + a$	$a \cdot b = b \cdot a$
Distributivité	$a + (bc) = (a + b)(a + c)$	
	$a(b + c) = (ab) + (ac)$	
Associativité	$a + (b + c) = (a + b) + c = a + b + c$	
	$a(bc) = (ab)c = abc$	
Lois de De Morgan	$\overline{ab} = \overline{a} + \overline{b}$	$\overline{a + b} = \overline{a}\,\overline{b}$
Autres relations	$\overline{\overline{a}} = a$	$(a + b)(a + \overline{b}) = a$

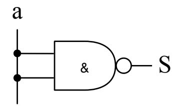
Complétude de la porte NON-ET

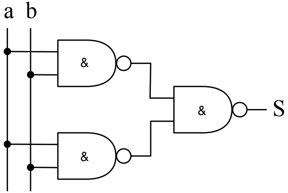
La porte NON-ET est complète : on peut réaliser n'importe quelle fonction booléenne avec seulement des portes NON-ET.

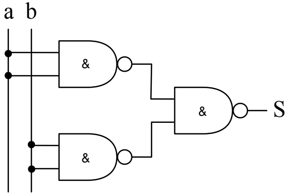
$$\bar{a} = \overline{a \cdot a}$$

$$\begin{aligned} a \cdot b &= \overline{\overline{a \cdot b}} \\ &= \overline{\overline{ab} \cdot \overline{ab}} \end{aligned}$$

$$\begin{aligned} a + b &= \overline{\overline{a + b}} \\ &= \overline{\bar{a} \cdot \bar{b}} \\ &= \overline{\overline{aa} \cdot \overline{bb}} \end{aligned}$$



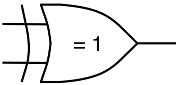




Complétude de NON-OU : en TD

Porte XOR

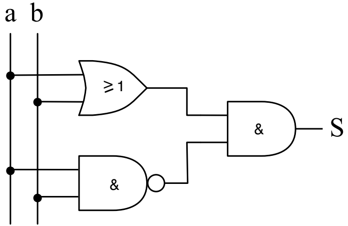
porte XOR

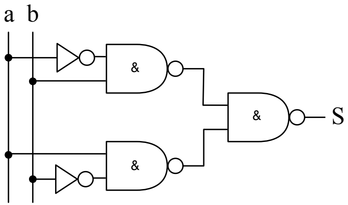


a	b	S
0	0	0
0	1	1
1	0	1
1	1	0

$$
\begin{aligned}
f(a,b) &= a \oplus b \\
&= (a+b)(\overline{ab}) \\
&= (a+b)(\bar a + \bar b) \\
&= a\bar a + a\bar b + b\bar a + b\bar b \\
&= a\bar b + b\bar a \\
&= \overline{\overline{a\bar b}} + \overline{\overline{b\bar a}} \;=\; \overline{\overline{a\bar b}\cdot\overline{b\bar a}}
\end{aligned}
$$

Réalisation à partir des portes précédentes.



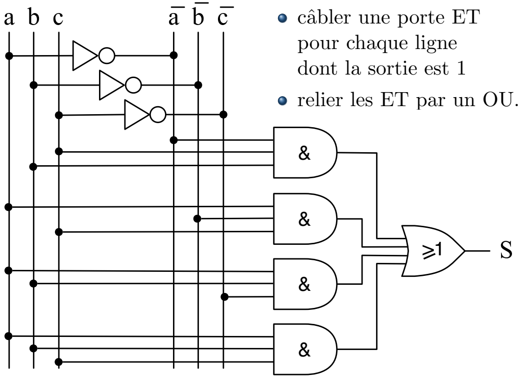


De la table de vérité au circuit

La fonction majoritaire : la valeur de sortie est celle qui apparaît majoritairement en entrée.

- câbler les entrées et leurs négations
- câbler une porte ET pour chaque ligne dont la sortie est 1
- relier les ET par un OU.

a	b	c	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



$$f(a,b,c) = \bar{a}bc + a\bar{b}c + ab\bar{c} + abc$$

Simplification

Simplifier = diminuer le nombre d'opérateurs = diminuer le nombre de portes logiques (et donc le coût).

exemple précédent :

$$\begin{aligned} f(a,b,c) &= \overline{\overline{a+b}\cdot bc} \\ &= \overline{\overline{a+b}} + \overline{bc} \\ &= a + \color{red}{b + \bar{b}} \color{black}{} + \bar{c} \\ &= a + 1 + \bar{c} = 1 \end{aligned}$$

fonction majoritaire :

$$\begin{aligned} f(a,b,c) &= \bar{a}bc + a\bar{b}c + ab\bar{c} + abc \\ &= \bar{a}bc + a\bar{b}c + ab\bar{c} + abc \color{red}{+ abc + abc} \\ &= (a+\bar{a})bc + (b+\bar{b})ac + (c+\bar{c})bc \\ &= ab + ac + bc \end{aligned}$$

$$\begin{aligned} &= (\bar{a}b + a\bar{b})c + ab(\bar{c}+c) \\ &= (a+b)(\bar{a}+\bar{b})c + ab \\ &= (ac+bc)\overline{ab} + ab \\ &= (ab+ac+bc)(\overline{ab}+ab) \end{aligned}$$

Méthode de Karnaugh

La fonction majoritaire

binaire classique

a	b	c	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

binaire réfléchi

a	b	c	S
0	0	0	0
0	0	1	0
0	1	1	1
0	1	0	0
1	1	0	1
1	1	1	1
1	0	1	1
1	0	0	0

a \ bc	00	01	11	10
0	0	0	1	0
1	0	1	1	1

a \ bc	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$$f(a,b,c) = ab + ac + bc$$

Méthode de Karnaugh (suite)

On veut le moins de porte ET et OU possible.

ab \ cd	00	01	11	10
00	1	1	1	1
01	0	1	1	0
11	0	0	1	0
10	1	0	1	1

ab \ cd	00	01	11	10
00	1	1	1	1
01	0	1	1	0
11	0	0	1	0
10	1	0	1	1

$$f(a,b,c,d) \;=\; \bar{a}\bar{b} + cd + a\bar{b}\bar{d} + \bar{a}b\bar{c}d \;=\; cd + \bar{a}d + \bar{b}\bar{d}$$

La première formule est juste, mais pas simplifiée au maximum !

Méthode de Karnaugh

Méthode graphique (visuelle) pour simplifier au maximum les formules et/ou les circuits.

1. Écrire la table de vérité sous la forme d'un code de Gray (ou binaire réfléchi) : ainsi, les valeurs des entrées ne diffèrent que d'un seul bit entre chaque ligne.
2. Compacter la table (ces 2 étapes peuvent être simultanées)
3. Entourer tous les 1 dans des rectangles :
   - les plus grands possibles,
   - tels que leur taille est une puissance de 2,
   - éventuellement sur les bords.
4. En déduire la formule et le circuit :
   - une somme (OU) des formules de chaque rectangle
   - le formule d'un rectangle est un produit (ET) :
     - des variables qui valent toujours 1 dans ce rectangle
     - des négations de celles qui valent toujours 0.
     - les autres variables n'apparaissent pas dans le ET.
5. !!! Il peut y avoir plusieurs simplifications maximales !