

Architecture des ordinateurs 1

Jean-Luc Scharbarg - ENSEEIHT - Dpt. SN

Octobre 2020

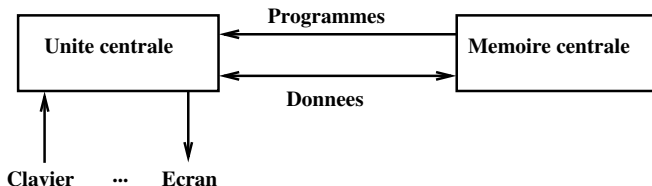
Organisation de l'enseignement

- Semestre 1 (de l'algorithme au circuit configurable)
 - ▶ 4 Cours, 3 TD, 4 TP, 1 examen écrit + 1 note de TP (présence + travail réalisé)
 - ▶ Introduction à la représentation des données
 - ▶ Circuits logiques
 - ★ Logique combinatoire
 - ★ Logique séquentielle
 - ★ Circuits séquentiels remarquables
 - ★ Construction de circuits séquentiels complexes
- Semestre 2 (de l'algorithme au processeur)
 - ▶ 3 Cours, 3 TD, 6 TP, 1 examen écrit + 1 note de TP (présence + travail réalisé)
 - ▶ Introduction à l'exécution d'un programme écrit dans un langage de haut niveau sur une architecture matérielle
 - ▶ Introduction à l'échange d'informations entre un processeur et son environnement
 - ▶ Introduction aux caractéristiques des processeurs actuels
 - ▶ Mise en œuvre d'un processeur simple

Un ordinateur, c'est quoi ?

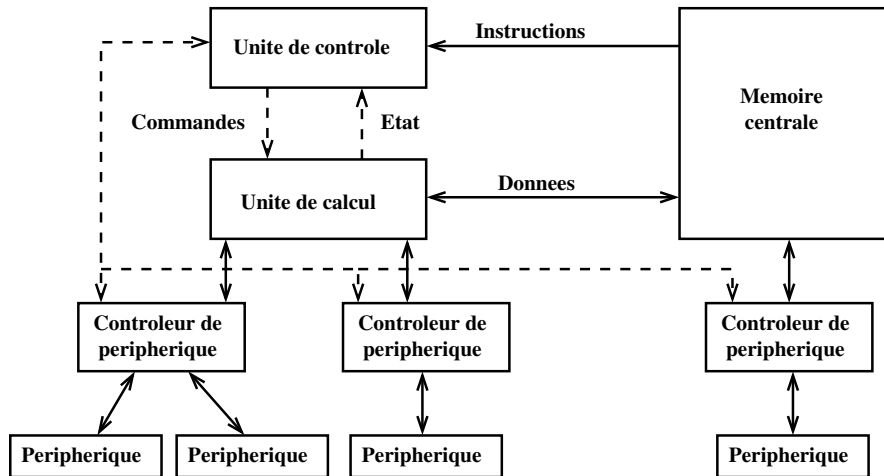
- Un ou plusieurs processeurs qui exécutent des programmes,
- Des moyens pour envoyer des ordres (clavier, souris, ...),
- Des moyens pour récupérer des résultats (écran, ...),
- Des moyens pour stocker de l'information (mémoire, disques, ...)
- Des moyens pour dialoguer avec d'autres dispositifs (interface réseau, ports, ...)

Organisation générale d'un ordinateur



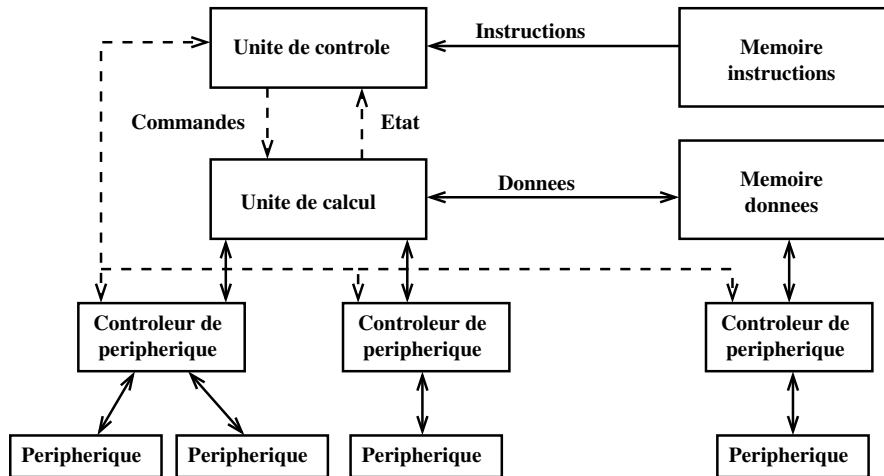
- Exécution de programmes par l'unité centrale
 - ▶ Lecture et écriture de données en mémoire centrale
 - ▶ Interactions avec l'extérieur
- Programmes et données stockés et véhiculés logiquement sous la forme de chiffres binaires ou bits (binary digits), physiquement sous la forme de signaux électroniques
- Développement d'un programme
 - ▶ Ecriture du programme dans un langage de "haut niveau" (e.g. Pascal)
 - ▶ Traduction du programme en langage machine (instructions plus rudimentaires)
 - ▶ Exécution du programme en langage machine

Modèle de Von Neuman



- Une mémoire commune aux données et aux programmes

Modèle de Harvard



- Deux mémoires séparées pour les données et les programmes

Représentation de l'information

- Calculateur : traitement automatisé des données \Rightarrow représentation de ces données en machine :
 - ▶ Nombres entiers signés ou non (toutes les variables entières)
 - ▶ Nombres réels (toutes les variables réelles)
 - ▶ Caractères (les variables caractères ou chaînes de caractères, le code source d'un programme, *ldots*)
 - ▶ Structures de données (assemblage des autres types de données)
 - ▶ Instructions (le code exécutable du programme)
- Représentation numérique en binaire (suites de 0 et de 1)
- Un chiffre binaire : un binary digit (bit)
- Dans cette partie du cours :
 - ▶ Représentation des entiers non signés en binaire pur
 - ▶ Représentation des entiers signés en complément à deux
 - ▶ Représentation des réels en virgule fixe
 - ▶ Représentation des réels en virgule flottante
 - ▶ Représentation des caractères

Entiers non signés en binaire pur

- Valeur de l'entier naturel en base deux sur n bits
 \Rightarrow représentation des entiers naturels α tels que $0 \leq \alpha \leq 2^n - 1$
- Le nombre de valeurs représentables est fini
- $A^n = (a_{n-1}^n, \dots, a_0^n)$: représentation de α : $\alpha = \sum_{i=0}^{n-1} (a_i^n \times 2^i)$
- Exemple :

$$\alpha_2 = 10110011$$

$$\alpha_{10} = 1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

$$\alpha_{10} = 128 + 32 + 16 + 2 + 1 = 179$$

Entiers non signés en binaire pur

- Passage décimal \rightarrow binaire pur :

- ▶ Divisions successives par 2

| | Quotient | Reste |
|------|----------|-------|
| 37/2 | 18 | 1 |
| 18/2 | 9 | 0 |
| 9/2 | 4 | 1 |
| 4/2 | 2 | 0 |
| 2/2 | 1 | 0 |
| 1/2 | 0 | 1 |

$$\Rightarrow 37_{10} = 100101_2$$

- ▶ Justification

$$\alpha = a_{n-1} \times 2^{n-1} + a_{n-2} \times 2^{n-2} + \dots + a_1 \times 2^1 + a_0 \times 2^0$$

$$\alpha = 2 \times (a_{n-1} \times 2^{n-2} + a_{n-2} \times 2^{n-3} + \dots + a_1 \times 2^0) + a_0$$

$$\alpha = 2 \times \textit{Quotient} + \textit{Reste}$$

Entiers non signés en binaire pur

- Passage binaire pur \Rightarrow décimal :

- ▶ schéma de Horner : $\alpha = \sum_{i=0}^{n-1} (a_i^n \times 2^i)$

- ★ $S_0 = 0$

- ★ $S_i = 2 \times S_{i-1} + a_{n-1}^n$ pour $1 \leq i \leq n$

- ★ S_n : valeur de α en décimal

- ▶ Exemple : $\alpha = 111001_2$

- ★ $S_0 = 0$

- ★ $S_1 = 2 \times S_0 + a_5 = 2 \times 0 + 1 = 1$

- ★ $S_2 = 2 \times S_1 + a_4 = 2 \times 1 + 1 = 3$

- ★ $S_3 = 2 \times S_2 + a_3 = 2 \times 3 + 1 = 7$

- ★ $S_4 = 2 \times S_3 + a_2 = 2 \times 7 + 0 = 14$

- ★ $S_5 = 2 \times S_4 + a_1 = 2 \times 14 + 0 = 28$

- ★ $S_6 = 2 \times S_5 + a_0 = 2 \times 28 + 1 = 57$

Entiers non signés en binaire pur

- Opérations arithmétiques similaires aux opérations arithmétiques en décimal

$$\begin{array}{r} 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1 \\ +\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1 \\ \hline 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0 \end{array}$$

$$\begin{array}{r} 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1 \\ -\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1 \\ \hline 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0 \end{array}$$

- Ecriture plus compacte :

- ▶ Représentation en octal (base huit \Rightarrow chiffres de 0 à 7)

Base 10 : 8607

Base 2 : 10 000 110 011 111

Base 8 : 2 0 6 3 7

- ▶ Représentation en hexadécimal (base 16 \Rightarrow chiffres 0, ..., 9, A, ..., F)

Base 10 : 8607

Base 2 : 10 0001 1001 1111

Base 16 : 2 1 9 F

Entiers signés

- Représentation d'un entier (positif, nul ou négatif) α par un nombre binaire sur n bits $A^n = (a_{n-1}^n, \dots, a_0^n) \in [0, 2^n - 1]$
- Solution la plus immédiate : un bit pour le signe et $n - 1$ bits pour la valeur absolue
 - ▶ a_{n-1}^n : bit de signe : 0 si $\alpha \geq 0$, 1 sinon
 - ▶ $(a_{n-2}^n, \dots, a_0^n)$: représentation de $|\alpha|$ en binaire pur
 - ▶ exemples sur 8 bits :
$$\alpha = 11 \quad \Rightarrow \quad A^8 = 00001011$$
$$\alpha = -13 \quad \Rightarrow \quad A^8 = 10001101$$
- ▶ Nombres représentables sur n bits : $-(2^{n-1} - 1) \leq \alpha \leq 2^{n-1} - 1$
- ▶ Deux représentations pour $\alpha = 0$: $00 \dots 0$ et $10 \dots 0$
- ▶ Intervalle des nombres représentables symétrique
- ▶ Passage d'une représentation sur n bits à une représentation sur r bits avec $r > n$: $A^n = (a_{n-1}^n, \dots, a_0^n) \Rightarrow A^r = (a_{n-1}^n, 0, \dots, 0, a_{n-2}^n, \dots, a_0^n)$
- ▶ Pas de bonnes propriétés arithmétiques \Rightarrow opérations de base complexes
- Représentation avec de bonnes propriétés arithmétiques : le complément à deux

Entiers signés en complément à deux

- Objectif : rendre le signe des opérandes transparents pour les opérations arithmétiques (addition, soustraction, ...)
- $A^n = (a_{n-1}^n, \dots, a_0^n)$: représentation en complément à deux sur n bits de l'entier signé α
 - $\alpha \geq 0 \Rightarrow A^n$: représentation de α en binaire pur
 - $\alpha < 0 \Rightarrow A^n$: représentation de $\alpha + 2^n$ en binaire pur
- Nombres signés représentables sur n bits : $-(2^{n-1}) \leq \alpha \leq 2^{n-1} - 1$
- Exemple pour $n = 3$: $-4 \leq \alpha \leq 3$

| | |
|---------------|-------------|
| $\alpha = -4$ | $A^n = 100$ |
| $\alpha = -3$ | $A^n = 101$ |
| $\alpha = -2$ | $A^n = 110$ |
| $\alpha = -1$ | $A^n = 111$ |

| | |
|--------------|-------------|
| $\alpha = 0$ | $A^n = 000$ |
| $\alpha = 1$ | $A^n = 001$ |
| $\alpha = 2$ | $A^n = 010$ |
| $\alpha = 3$ | $A^n = 011$ |

- Passage d'une représentation sur n bits à une représentation sur r bits avec $r > n$:
 $A^n = (a_{n-1}^n, \dots, a_0^n) \Rightarrow A^r = (a_{n-1}^n, \dots, a_{n-1}^n, a_{n-2}^n, \dots, a_0^n)$

Entiers signés en complément à deux

- Passage de la représentation de α à la représentation de $-\alpha$:
Compémentation bit à bit et ajout de 1 *modulo* 2^n

$$\begin{array}{rclclcl} \alpha & = & 6 & \Rightarrow & A^8 & = & 00000110 \\ & & & & & & 11111001 \\ & & & & & & + \quad 1 \\ -\alpha & = & -6 & \Rightarrow & M_A^8 & = & \hline & & & & & & 11111010 \end{array}$$

- Opérations arithmétiques : A^n : représentation de α , B^n : représentation de β
 - ▶ S^n : représentation de $\alpha + \beta$
si pas de débordement ($\alpha + \beta$ représentatble en complément à deux sur n bits) alors

$$S^n = (A^n + B^n) \bmod 2^n$$

- ▶ D^n : représentation de $\alpha - \beta = \alpha + (-\beta) \Rightarrow$ on se ramène à une addition

Représentation des nombres réels

- Représentation par une séquence de longueur finie de bits \Rightarrow nombre fini de valeurs représentables \Rightarrow pas de représentation des nombres réels au sens mathématique du terme
- Critères d'évaluation d'une représentation des nombres réels
 - ▶ Intervalle des nombres représentables
 - ▶ Précision (pourcentage d'erreur)
 - ▶ Complexité de mise en œuvre
- Deux grandes classes de représentations
 - ▶ Représentations en virgule fixe
 - ★ Les plus simples
 - ★ Pas très bonnes en terme de précision et d'intervalles de nombre représentables
 - ▶ Représentations en virgule flottante
 - ★ Plus complexes à mettre en œuvre
 - ★ bonnes en terme de précision et d'intervalle de nombres représentables
 - ★ Norme : format IEEE 754

Nombres réels en virgule fixe

- Utilisation de la représentation des nombres entiers, avec une virgule implicite toujours au même endroit dans la représentation du nombre
- Nombre réel α non signé : représentation $A^n = (a_{n-1}^n, \dots, a_0^n)$ sur n bits, dont d après la virgule

$$\alpha = \sum_{i=0}^{n-1} (a_i^n \times 2^{i-d})$$

- Exemple avec $n = 16$ et $d = 8$

$$\begin{aligned} A^n = 00100001 \ 10010000 &\Rightarrow \alpha = 2^5 + 2^0 + 2^{-1} + 2^{-4} \\ \alpha &= 32 + 1 + \frac{1}{2} + \frac{1}{16} \\ \alpha &= 33.5625 \end{aligned}$$

- Calcul de la représentation en virgule fixe

| | Quotient | Reste |
|-----|----------|-------|
| 6/2 | 3 | 0 |
| 3/2 | 1 | 1 |
| 1/2 | 0 | 1 |

| | Produit | Partie entière |
|------------------|---------|----------------|
| 0.375×2 | 0.75 | 0 |
| 0.75×2 | 1.5 | 1 |
| 0.5×2 | 1.0 | 1 |

$$\Rightarrow 6.375_{10} = 110.011_2$$

Nombres réels en virgule fixe

- Représentation non exacte de la plupart des nombres réels

Exemple avec $n = 16$ et $d = 8$: $11.8_{10} = 00001011.11001100_2$

- Opérations arithmétiques en virgule fixe similaires aux opérations arithmétiques sur les entiers non signés

$$\begin{array}{r|l} \alpha = 2.75 & 0 \ 0 \ 1 \ 0 \ . \ 1 \ 1 \ 0 \\ \beta = 3.625 & + \ 0 \ 0 \ 1 \ 1 \ . \ 1 \ 0 \ 1 \\ \hline 6.375 & 0 \ 1 \ 1 \ 0 \ . \ 0 \ 1 \ 1 \end{array}$$

- Nombres réels signés en virgule fixe : utilisation du complément à deux en faisant abstraction de la virgule

Exemple pour $n = 7$ et $d = 3$

$$\begin{array}{lcl} \alpha = 2.75 & \Rightarrow & \text{Représentation : } 0010.110 \\ - \alpha = -2.75 & \Rightarrow & \text{Représentation : } 1101.010 \end{array}$$

- Opérations arithmétiques comme sur les entiers signés

$$\begin{array}{r|l} \alpha = 2.75 & 0 \ 0 \ 1 \ 0 \ . \ 1 \ 1 \ 0 \\ \beta = -3.625 & + \ 1 \ 1 \ 0 \ 0 \ . \ 0 \ 1 \ 1 \\ \hline -0.875 & 1 \ 1 \ 1 \ 1 \ . \ 0 \ 0 \ 1 \end{array}$$

Nombres réels en virgule fixe

- Différence entre deux valeurs consécutives représentables exactement : 2^{-d} , constante sur tout l'intervalle de représentation

Exemple pour $n = 7$ et $d = 3$

$$0000.000 \Rightarrow \alpha = 0$$

$$0000.001 \Rightarrow \alpha = 2^{-3}$$

$$0000.010 \Rightarrow \alpha = 2 \times 2^{-3}$$

...

- Pourcentage d'erreur potentielle d'autant plus fort que la valeur absolue du nombre réel représenté est petite
- Intervalle des valeurs représentable relativement réduit :
 $[-2^{n-d-1}, 2^{n-d-1} - 2^{-d}]$
Pour $n = 32$ et $d = 16$: $[-2^{15}, 2^{15} - 2^{-16}]$
- Représentation en virgule fixe de moins en moins utilisée de nos jours

Nombres réels en virgule flottante : principe

- S'inspire de la notation scientifique des nombres réels :

$$\alpha = f \times 10^e \text{ (en base 10)}$$

- f : mantisse, e : exposant
- Notation non unique : $3.14 \times 10^0 = 0.314 \times 10^1 = 314 \times 10^{-2}$
- Virgule flottante : notation scientifique en base 2
 α représenté par $\mu_2 \times 2^{e_2}$
- Deux possibilités pour la normalisation de l'écriture :
 $\mu \in [0.5, 1[\Rightarrow \mu = 0.1 \dots$
 $\mu \in [1, 2[\Rightarrow \mu = 1. \dots$
- Représentation en machine avec trois champs

| | | |
|-----|-----|-----|
| S | E | M |
|-----|-----|-----|

S : signe du nombre sur 1 bit

E : exposant en excédent sur e bits

M : mantisse sur m bits

Nombres réels en virgule flottante : IEEE 754

- Standard défini en 1985 par l'IEEE (Institute of Electrical and Electronics Engineers)
- Utilisé par la plupart des processeurs actuels
- Deux formats principaux définis :
 - ▶ simple précision sur 32 bits : $e = 8$, $m = 23$
 - ▶ double précision sur 64 bits : $e = 11$, $m = 52$
- Nombres réels normalisés : $\alpha = 1.\mu' \times 2^\epsilon$

$$M = \mu' \text{ et } E = \epsilon + 2^{e-1} - 1$$

Exemple : $\alpha = -9.5 \Rightarrow \alpha = -1.0011_2 \times 2^{11}_2$

Simple précision : $S = 1$

$$E = 3 + 127 = 130_{10} = 10000010_2$$

$$M = 00110 \dots 0$$

Double précision : $S = 1$

$$E = 3 + 1023 = 1026_{10} = 10 \dots 010_2$$

$$M = 00110 \dots 0$$

Nombres réels en virgule flottante : IEEE 754

- Nombres réels dénormalisés : représentation des nombres réels à très petite valeur absolue
 - ▶ tous les bits du champ E à 0 $\Rightarrow \epsilon = -127$ (simple précision) ou $\epsilon = -1023$ (double précision)
 - ▶ la mantisse vaut $0.M$
 - ▶ Exemple en simple précision :
 $S = 0, E = 0 \dots 0, M = 010 \dots 0 \Rightarrow \alpha_{10} = 0.25 \times 2^{-127}$
- Représentation de 0 : $E = 0 \dots 0, M = 0 \dots 0, S$ indifférent
- Représentation de l'infini : $E = 1 \dots 1, M \neq 0 \dots 0$
- Représentation de NaN (Not a Number : indéfini) :
 $E = 1 \dots 1, M = 0 \dots 0$
- Intervalle des nombres représentables
 - ▶ Simple précision : de -10^{38} à 10^{38}
 - ▶ Double précision : de -10^{308} à 10^{308}
- Pourcentage d'erreur potentielle constant

Représentation des caractères

- Codage numérique d'un répertoire de caractères
- La longueur du code définit le nombre maximum de caractères du répertoire
- Le code ASCII (American Standard Code for Information Interchange)
 - ▶ Le standard sur les machines actuelles
 - ▶ Codage sur 7 bits \Rightarrow 128 caractères possibles
 - ▶ Code construit pour simplifier la manipulation des caractères
- Le code ISO-8859-1 (ASCII étendu)
 - ▶ Codage sur 8 bits \Rightarrow 256 caractères possibles
 - ▶ Langues latines (accents, ...)
- Les codes universels sur 2 ou 4 octets

Table ASCII (1)

| Hexa | Carac | Hexa | Carac | Hexa | Carac | Hexa | Carac |
|------|-------|------|-------|------|-------|------|-------|
| 0 | NUL | 10 | DLE | 20 | | 30 | 0 |
| 1 | SOH | 11 | DC1 | 21 | ! | 31 | 1 |
| 2 | STX | 12 | DC2 | 22 | " | 32 | 2 |
| 3 | ETX | 13 | DC3 | 23 | # | 33 | 3 |
| 4 | EOT | 14 | DC4 | 24 | \$ | 34 | 4 |
| 5 | ENQ | 15 | NAK | 25 | % | 35 | 5 |
| 6 | ACK | 16 | SYN | 26 | & | 36 | 6 |
| 7 | BEL | 17 | ETB | 27 | ' | 37 | 7 |
| 8 | BS | 18 | CAN | 28 | (| 38 | 8 |
| 9 | TAB | 19 | EM | 29 |) | 39 | 9 |
| A | LF | 1A | SUB | 2A | * | 3A | : |
| B | VT | 1B | ESC | 2B | + | 3B | ; |
| C | FF | 1C | FS | 2C | , | 3C | < |
| D | CR | 1D | GS | 2D | - | 3D | = |
| E | SO | 1E | RS | 2E | . | 3E | > |
| F | SI | 1F | US | 2F | / | 3F | ? |

Table ASCII (2)

| Hexa | Carac | Hexa | Carac | Hexa | Carac | Hexa | Carac |
|------|-------|------|-------|------|-------|------|-------|
| 40 | @ | 50 | P | 60 | ' | 70 | p |
| 41 | A | 51 | Q | 61 | a | 71 | q |
| 42 | B | 52 | R | 62 | b | 72 | r |
| 43 | C | 53 | S | 63 | c | 73 | s |
| 44 | D | 54 | T | 64 | d | 74 | t |
| 45 | E | 55 | U | 65 | e | 75 | u |
| 46 | F | 56 | V | 66 | f | 76 | v |
| 47 | G | 57 | W | 67 | g | 77 | w |
| 48 | H | 58 | X | 68 | h | 78 | x |
| 49 | I | 59 | Y | 69 | i | 79 | y |
| 4A | J | 5A | Z | 6A | j | 7A | z |
| 4B | K | 5B | [| 6B | k | 7B | { |
| 4C | L | 5C | \ | 6C | l | 7C | |
| 4D | M | 5D |] | 6D | m | 7D | } |
| 4E | N | 5E | ^ | 6E | n | 7E | ~ |
| 4F | O | 5F | _ | 6F | o | 7F | DEL |