

Architecture des ordinateurs

Introduction

Florent Devin

EISTI



Introduction

Déroulement

Déroulement

- 12 séances de cours
- 6 TD (papier) et 6 TP (VHDL)
- 1 examen (50 % évaluation)
- 1 projet (50 % évaluation)
 - 1 TP = 1 pièce du projet
 - 6H de projet encadrées
 - présence obligatoire (sinon 0 au livrable)

Bibliographie

Bibliographie

- Paolo ZANELLA et Yves LIGIER. *Architectures et technologie des ordinateurs - Cours et exercices résolus - 3^e édition* - Dunod
- Andrew TANENBAUM. *Architecture de l'ordinateur* - Inter Edition
- Ernest HIRSCH et Serge WENDLING. *Structure des ordinateurs - Concepts de base, machines conventionnelle et architectures parallèles* - Armand Colin
- F. Thomson LEIGHTON. *Introduction aux algorithmes et architectures parallèles* - Morgan Kaufmann

Objectifs

Objectifs

- Qu'est ce qu'un ordinateur ?
 - Comment fonctionne un ordinateur ?

Définition

Ordinateur : n. m. INFORM Machine capable d'effectuer *automatiquement* des *opérations arithmétiques* et *logiques* (à des fins scientifiques, administratives, comptables, ...) à partir de *programmes* définissant la *séquence* de ces opérations.

Objectifs

Différents composants

- Applications : *Software*
 - Winzip, LaTeX, Xemacs, WoF, Xblast, ...
- Systèmes d'exploitation : *Operating System*
 - Windows XP, Unix, Solaris, Linux, BeOS, MacOS, ...
- Matériel : *Hardware*
 - PC, PowerPC, Apple, Station UltraSparc, ...

Machine perçue par l'utilisateur

Ce que l'on perçoit

Interactions :

- entrée de commandes ;
- lancements d'application ;
- visualisation, ...

avec la machine par des périphériques :

- clavier ;
- souris ;
- écran ;
- imprimante ;
- disquette, ...

Machine invisible à l'utilisateur

Fonctionnement interne

L'unité centrale contient 3 unités fonctionnelles :

- l'automate ;
- la partie calcul ;
- la mémoire.

Action de l'utilisateur \Rightarrow séquence d'opérations faisant intervenir ces unités.

Historique

Historique

Préhistoire

- -500 : Apparition des premiers outils à calculer \Rightarrow boulier, abaque
- 1623 : Francis BACON invente le premier codage de l'alphabet
- 1632 : Invention de la règle à calcul (OUGHTRED)
- 1642 : PASCAL invente la pascaline
- LEIBNIZ (1646-1713) : envisage le raisonnement d'une machine
- 1728 : FALCON construit le métier à tisser utiliser les *cartes perforées*

Historique

Préhistoire

• 1833 : Machine de Babbage

- Principes du métier à tisser
- 4 opérations arithmétiques de bases
- Ensuite il imagine une machine analytique
 - Quatre parties : magasin (mémoire), moulin (unité de calcul), entrée (lecteur de cartes perforées), sortie (perforation)
 - Opérations arithmétiques, test et branchement conditionnel
 - ADA LOVELACE crée le premier programme informatique
 - Non construit faute de moyen.

Historique

Préhistoire

- 1840 : ADA LOVELACE définit le principe des itérations successives \Rightarrow *algorithme*
- **1854 : Boole publie un ouvrage sur la logique**
- 1904 : JOHN FLEMING Invention du tube à vide
- **1937 : Alan Turing publie des articles sur les fonctions calculables**
- **1938 : Shanon rapproche l'algèbre de Boole et les circuits électroniques**
- 1943 : Création du ASCC Mark I \Rightarrow Automatic Sequence-Controlled Calculator
- 1945 : Naissance du bogue (*Bug*)

Historique

Première génération

- **1946 : Création de l'ENIAC**
 - Electronic Numerical Integrator and Computer
 - Architecture de Von Neumann
- **1947 : Invention du transistor (Bell telecom)**

Historique

Deuxième génération

- **1956 : Premier ordinateur à transistor \Rightarrow TRADIC (Bell)**
- **1958 : Premier circuit intégré \Rightarrow Texas Instrument**
- 1960 : Premier jeu sur ordinateur \Rightarrow SpaceWar !
- 1964 : Langage de programmation BASIC
- 1965 : loi de MOORE
- **1968 : Invention de la souris \Rightarrow Stanford**
- **1969 : Systèmes d'exploitation**
 - MULTICS puis **UNIX** \Rightarrow Bell

Historique

Troisième génération

- 1971 : ARPANET (ancêtre d'internet)
- **1971 : Premier microprocesseur \Rightarrow Intel**
 - 4004 : 4 bits, 108 KHz, 2300 transistors en 10 microns
- 1972 : Intel sort le 8008 \Rightarrow 8 bits, 200 KHz, 3500 transistors
- 1972 : Bill Gates et Paul Allen fondent Traf-of-Data
- **1973 : Gary Kildall écrit le système d'exploitation CP/M**
- **1973 : Invention du C pour le développement d'UNIX**

Historique

Troisième génération

- 1974 : Le français François Moreno invente la carte à puce
- **1974 : Motorola commercialise son premier processeur \Rightarrow le 6800, 8 bits**
- 1974 : Intel sort le 8080 \Rightarrow 8 bits
- 1975 : Traf-of-data devient Micro-Soft
- **1976 : Steve Jobs et Steve Wozniak commercialisent l'Apple Computer, à base de MOS Tech 6502**
- **1976 : Zilog sort le Z80 \Rightarrow 8 bits, 2.5 MHz**

Historique

Troisième génération

- **1978 : Intel lance son 8086 \Rightarrow 16 bits, 4.7 Mhz, 29000 transistors à 3 microns**
- 1979 : Taito sort le jeu Space Invaders
- **1979 : Motorola commercialise le 68000 \Rightarrow 16/32 bits, 68000 transistors**
- 1980 : Sinclair sort le ZX80 à base de Z80
- 1980 : IBM sous traite le système d'exploitation de sa future machine, à base de 8086, à Micro-Soft
 - QDOS \Rightarrow 86-DOS \Rightarrow MS-DOS

Historique

Quatrième génération

- **1982 Intel commercialise le 80286 \Rightarrow 16 bits, 6 MHz, 134000 transistors**
- 1982 : Micro-Soft édite une version de MS-DOS pour compatibles...
- Sony et Phillips invente le CD-ROM
- 1984 Apple sort le MacIntosh avec une interface graphique conviviale !
- **1985 Intel commercialise le 80386 \Rightarrow 32 bits**
- 1986 : Premier ordinateur multi-processeur

Historique

Quatrième génération

- 1989 : TIM BERNERS-LEE : HTML
- 1990 : Pc en réseau, Premier CD-R, Windows 3.0
- 1991 : LINUS TORVALDS : Linux
- 1993 : Pentium
- 1995 : Windows 95 (1 million en 3 jours)
- 1996 : Javascript
- 1998 : Windows 98 - iMac
- 1999 : iBook
- 2000 : Pentium IV, Itanium, Wap

Codage de l'information

Principe de codage

Présentation

- plusieurs types de codage :
 - longueur fixe :
 - numéro de téléphone ;
 - numéro de sécurité sociale ;
 - code postal, ...
 - b étant la base du codage \Rightarrow représentation de b^n éléments
 - longueur variable :
 - alphabet morse ;
 - ADN, ... ;

Codage de l'information

Nécessité

- exemple : transmission d'un message
 - amplification directe : porte-voix
 - codage : courrier, signaux de fumée, signal électrique
- informatique \Rightarrow codage binaire










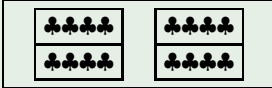
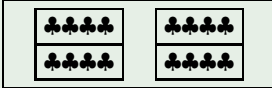

Système de numération

Base 4

- En base 10, on utilise les chiffres
 - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- En base 4, on utilise les chiffres
 - 0, 1, 2, 3

Système de numération

Base 4

	base 4	base 10		base 4	base 10
	0	0		11	5
	1	1		12	6
	2	2		13	7
	3	3		20	8
	10	4			
					
$3 + 1 * 4 + 2 * 4 * 4 = 3 + 4 + 32 = 39$				213	39

Système les plus utiles

Différents systèmes

- 10 : décimal
 - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- 2 : binaire
 - 0, 1
- 8 : octal
 - 0, 1, 2, 3, 4, 5, 6, 7
- 16 : hexadécimal
 - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Codage des nombres

Conversion

Principe de conversion

- Comment passer d'un système à un autre ?
 - Définition : utile pour les conversions *vers* le système décimal
 - $(N_1)_A \rightarrow (N_2)_{10}$
 - $(32)_o \rightarrow (3 * 8 + 2)_{10} \rightarrow (26)_{10}$
 - Division et multiplication successives : utile pour les conversions à partir de la base 10
 - $(N_1)_{10} \rightarrow (N_2)_B$
 - $(34,625)_{10} \rightarrow (?)_2$

Conversion

Exemple de conversion

$$(34, 625)_{10} \rightarrow (?)_2$$

Partie entière			Partie fractionnaire		
			↓		
$34/2 = 17$	r	0	$0.625 * 2 =$	1	.25
$17/2 = 8$	r	1	$0.25 * 2 =$	0	.5
$8/2 = 4$	r	0	$0.5 * 2 =$	1	
$4/2 = 2$	r	0			
$2/2 = 1$	r	0			
$1/2 = 0$	r	1			
↑					

$$(34, 625)_{10} \rightarrow (100010.101)_2$$

Conversion

Binaire \leftrightarrow hexadécimal

- binaire \rightarrow hexadécimal
 - groupe par 4 à partir du point (partie entière)
 - 1 groupe \Leftrightarrow 1 chiffre hexadécimal
- hexadécimal \rightarrow binaire
 - conversion de chaque chiffre en son équivalent binaire
- exemples
 - $0000_{(2)} \leftrightarrow 0_{(16)}$
 - $0101_{(2)} \leftrightarrow 5_{(16)}$
 - $1010_{(2)} \leftrightarrow A_{(16)}$

Module et signe

Présentation

- idée : ajouter un bit en tête pour indiquer le signe
 - on utilise le bit de poids fort pour représenter le signe
 - exemple :
 - $0011_{(2)} \leftrightarrow 3_{(10)}$
 - $1011_{(2)} \leftrightarrow -3_{(10)}$
 - avantage :
 - très facile à comprendre
 - inconvénients :
 - 2 façons de coder 0 ;
 - comparaison difficile ;
 - test du signe obligatoire avant addition.

Complément à 1

Complément à 1

- pour un nombre négatif, on prend la représentation de la partie entière, et on inverse tous les bits
- exemple :
 - $0011_{(2)} \leftrightarrow 3_{(10)}$
 - $1100_{(2)} \leftrightarrow -3_{(10)}$
- avantages :
 - comparaison aisée
 - facile à “câbler”
- inconvénients :
 - 2 façons de représenter 0

Complément à 2

Complément à 2

- Représentation utilisée par défaut
 - même méthode que le complément à 1, mais on ajoute 1 à la fin. (Complément à 1)+1 = complément à 2 \Leftrightarrow abus de langage
 - exemple :
 - $0011_{(2)} \leftrightarrow 3_{(10)}$
 - $1101_{(2)} \leftrightarrow -3_{(10)}$
 - avantages :
 - comparaison aisée
 - une seule façon de coder 0
 - permet les soustractions rapides
 - inconvénients :
 - difficile à appréhender

Exemple

Comparaison des différentes méthodes

	S	C1	C2
000	0	0	0
001	1	1	1
010	2	2	2
011	3	3	3
100	-0	-3	-4
101	-1	-2	-3
110	-2	-1	-2
111	-3	-0	-1

Norme IEEE 754

Présentation de la norme

- simple précision

1 bit	8 bits	23 bits
signe	exposant	mantisse

- double précision


1 bit	11 bits	52 bits
signe	exposant	mantisse

- $nombre = (-1)^{signe} * 1, mantisse * 2^{(exposant - biais)}$
- biais
 - simple précision : 127
 - double précision : 1023

Norme IEEE 754

Exemple : codage de $-118,625$ en IEEE 754 (32 bits)

- en binaire, ce nombre vaut $n_{(2)} = -1110110,101$
- à mettre sous la forme $(-1)^s * 1, M * 2^{e-b}$
- bit de signe : $s = 1$
- décalage de la virgule à gauche :
 $|n_{(2)}| = 1,110110101 * 2^6$
- mantisse : $M = 110110101000000000000000$
- exposant ($b = 127$) : $e = (6 + 127)_{(10)} = 10000101_{(2)}$
- code du nombre :

$s:31$ $exposant:30..23$ $mantisse:22..0$


Norme IEEE 754

Cas particuliers

signe	exposant	mantisse	valeur
-	0...0	0...0	± 0
-	1...1	0...0	$\pm \infty$
-	1...1	—	NaN
-	0...0	—	Nombre denormalisé

- nombre dénormalisé
 - $\text{nombre} = (-1)^{\text{signe}} * 0, \text{mantisse} * 2^{(\text{exposant} - \text{biais})}$

Correction de code

Correction d'erreur

Présentation

- Transformation d'un 0 en 1, ou inversement, fréquente en informatique
 - problème de transmission, défaillance d'un circuit, perturbation électromagnétique, ...
- Utilisation d'un code pour détecter les erreurs
 - bit de parité
 - ajout d'un bit pour vérifier le nombre de bits
 - 0110 \rightarrow 01100 (nb 1 pair, ajout de 0)
 - 1011 \rightarrow 10111 (nb 1 impair, ajout de 1)
 - Code de Hamming : 3 bits ajoutés à 4 bits de données
 - Code de Gray
 - VRC / LRC

Contrôle de parité

Présentation

- le plus simple
- Composé de $m + 1$ bits : m bits d'information + 1 bit de parité
- Somme des bits à 1
 - paire \Rightarrow parité paire
 - impaire \Rightarrow parité impaire
- Nombre d'erreurs impaire \Rightarrow détection possible
- Nombre d'erreurs paire \Rightarrow détection impossible
- Utilisé que dans des transmissions fiables \rightarrow l'intérieur d'un ordinateur

Double parité

Exemple

- Exemple : Double parité impaire

0	1	2	3	4	5	6	p	
0	1	1	1	0	0	1	0	Erreur
0	1	1	1	0	0	1	1	OK
0	1	1	0	1	1	0	1	OK
0	1	1	1	0	0	0	0	OK
1	1	1	1	0	0	1		
OK	OK	OK	Erreur	OK	OK	OK		

VRC/LRC

Contrôle de parité croisé VRC/LRC

- transmission du message HELLO en code ASCII
- parité paire VRC (*Vertical Redundancy Check*)
- parité paire VLC (*Longitudinal Redundancy Check*)

	Code ascii 7 bits	LRC
H	1001000	0
E	1000101	1
L	1001100	1
L	1001100	1
O	1001111	1
VRC	1000010	0

Code de Hamming & CRC

Présentation

- Code de Hamming
 - Ajout de k bits de test aux m bits de donnée
 - $2^k > k + m$
 - Permet la détection et la correction de 1 bit (dans sa version simple)
 - Voir Td sur les entrées/sorties
- CRC
 - Détection d'erreurs groupées
 - Méthode basé sur les polynômes

Code Gray

Code Gray ou binaire réfléchi ou REFLEX

- Code binaire classique :

0	0	0
1	0	1
2	1	0
3	1	1

Pb : inversion de 2 bits à la fois

- Code Gray :

0	0	0
1	0	1
2	1	1
3	1	0

1 seul bit change à la fois

Code Gray

Construction du code Gray

- pour 1 bit, on commence à zéro
- pour n bits ($n > 1$) :
 - on reprend le code sur $n - 1$ bits
 - on le reproduit en "miroir" pour la 2e moitié du code
 - on complète avec le bit de poids fort :
 - 0 pour la 1ère moitié du code
 - 1 pour la 2ème moitié

Code Gray

Code Gray à 3 bits

0	0	0	0
1	0	0	1
2	0	1	1
3	0	1	0
4	1	1	0
5	1	1	1
6	1	0	1
7	1	0	0

Code Gray

Code Gray à 3 bits

0	0	0	0
1	0	0	1
2	0	1	1
3	0	1	0
4	1	1	0
5	1	1	1
6	1	0	1
7	1	0	0

Diagram illustrating the 3-bit Gray code sequence (0 to 7) and its relationship to the 3-bit binary code (0 to 7). The 3-bit Gray code is shown in the first column, and the 3-bit binary code is shown in the second column. The 3-bit Gray code is derived from the 3-bit binary code by applying a mirror operation (miroir) to the first bit of the binary code. The diagram shows the 3-bit Gray code (0 to 7) and the 3-bit binary code (0 to 7). The 3-bit Gray code is derived from the 3-bit binary code by applying a mirror operation (miroir) to the first bit of the binary code. The diagram shows the 3-bit Gray code (0 to 7) and the 3-bit binary code (0 to 7). The 3-bit Gray code is derived from the 3-bit binary code by applying a mirror operation (miroir) to the first bit of the binary code.

Autres codages

Compression

- Codage de Huffman
- mp3
- uuencode, ...

Conversion

- wav
- bmp, ...