Architecture des Ordinateurs
et
Systèmes d'Exploitation

Partie 1 : Architecture des Ordinateurs
Introduction & Rappels

Fabrice BOISSIER & Elena KUSHNAREVA
2017/2018

fabrice.boissier@gmail.com
elena.kushnareva@malix.univ-paris1.fr

Introduction/Rappels

- Dans ce document :
  - Les nombres binaires sont précédés d'un caractère ' % '
  - Les nombres hexadécimaux sont précédés d'un ' $ '

Introduction/Rappels

- Compter en binaire :
  - Base 2
  - 2 symboles pour tout représenter (0 ou 1)

Binaire	0	1	10	11	100	101	110	111	1000	1001	…
Décimal	0	1	2	3	4	5	6	7	8	9	…

Introduction/Rappels

- Représentation des nombres entiers sur 8 bits :

% 0010 1010

0	0	1	0	1	0	1	0
128	64	32	16	8	4	2	1

(0 * 128) + (0 * 64) + (1 * 32) + (0 * 16) +
(1 * 8) + (0 * 4) + (1 * 2) + (0 * 1)

32 + 8 + 2 = 42

Introduction/Rappels

- Entiers non-signés :

  Chaque bit code une puissance de 2

8 bits = 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 (de 0 à 7)
% 0000 0000 = 0 % 1111 1111 = 255
(256 valeurs codées = 2^8 valeurs codées)

16 bits = 2^15 + … + 2^0 (de 0 à 15)
% 0000 0000 0000 0000 = 0 % 1111 1111 1111 1111 = 65.535
(65.536 valeurs codées = 2^16 valeurs codées)

32 bits = 2^31 + … + 2^0 (de 0 à 31)
% 0 = 0 $ FFFF FFFF = 4.294.967.295

Introduction/Rappels

- Entiers non-signés :

  Chaque bit code une puissance de 2

  (astuce visuelle :

  Nombres pairs finissent par ‘0’ à droite
  Nombres impairs finissent par ‘1’ à droite)

  Quelques valeurs notables :

  % 0000 0000 = 0 (Valeur minimale)

  % 1111 1111 = 255 (Valeur maximale)

Introduction/Rappels

- Entiers signés : 1 bit réservé au signe

nombre négatif = « complément à un du positif » + 1

(« complément à un » = inverser chaque bit)

42	-- sur 8 bits -->	% 0010 1010
complément à un de 42	-- sur 8 bits -->	% 1101 0101
« complément à un de 42 » + 1	-- sur 8 bits -->	% 1101 0110
-42	-- sur 8 bits -->	% 1101 0110

Introduction/Rappels

- Entiers signés : 1 bit réservé au signe
  nombre négatif = « complément à un du positif » + 1

  (astuce visuelle :

  Entiers positifs commencent par ‘0’ à gauche
  Entiers négatifs commencent par ‘1’ à gauche)

  42 -- sur 8 bits --> % 0010 1010
  -42 -- sur 8 bits --> % 1101 0110

Introduction/Rappels

- Entiers signés : 1 bit réservé au signe
  nombre négatif = « complément à un du positif » + 1

  Quelques valeurs notables :

  % 0000 0000 = 0
  % 1111 1111 = -1
  % 1000 0000 = -128 (valeur minimale)
  % 0111 1111 = 127 (valeur maximale)

Introduction/Rappels

- Entiers non-signés :
  Valeur minimale sur N bits = 0
  Valeur maximale sur N bits = 2^(n) – 1
  Nombre de valeurs codées sur N bits : 2^n

Exemple sur 8 bits :

0

2^8 – 1 = 255

2^8 = 256

- Entiers signés : 1 bit réservé au signe
  Valeur minimale sur N bits = -2^(n-1)
  Valeur maximale sur N bits = 2^(n-1) – 1
  Nombre de valeurs codées sur N bits : 2^n

Exemple sur 8 bits :

-2^(7) = -128

2^7 – 1 = 127

2^8 = 256

Convertir : binaire -> décimal

- Nombres Positifs :
  % 0110 1101 : multiplier par puissances de 2
  - 0 + 64 + 32 + 0 + 8 + 4 + 0 + 1 = 109

- Nombres Négatifs :
  % 1110 1110 :
  1. Faire complément à 1 : 0001 0001
  2. Ajouter +1 : 0001 0010
  3. Convertir : 2 + 16 = 18
  4. Résultat : -18

Convertir : décimal -> binaire

- Nombres Positifs :

  108 : Faire divisions par 2 successives, et conserver restes

  108 / 2
  0 54 / 2
  0 27 / 2
  1 13 / 2
  1 6 / 2
  0 3 / 2
  1 **1** => 110 1100 = % **0110 1100**

- Nombres Négatifs :

  -42 : Prendre positif, complément à 1, ajouter 1

42	=>	0010 1010		
complément à 1 de 42	=>	1101 0101		
(complément à 1 de 42) +1	=>	1101 0110	=>	% **1101 0110**

Introduction/Rappels

- Codage binaire (base 2) : de 0 à 1
  0, 1, 10, 11, 100, 101, 110, 111, 1000, …
- Codage octal (base 8) : de 0 à 7
  0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 12, …, 16, 17, 20, 21, …
- Codage décimal (base 10) : de 0 à 9
  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, …
- Codage hexadécimal (base 16) : de 0 à F
  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, 10, 11, …

Introduction/Rappels

- Exemple d'un entier dans plusieurs bases :

Décimal : 42

Binaire : % 101010 (1 * 2^5 + 1 * 2^3 + 1 * 2^2 = 32 + 8 + 2)

Octal : 52 (5 * 8 + 2 = 40 + 2)

Hexadécimal : $ 2A (2 * 16 + 10 = 32 + 10)

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Introduction/Rappels

- Hexadécimal : on convertit des paquets de 4 bits

% 0000 1111 => $ 0F => 15

% 1111 0000 => $ F0 => 240

(8 bits)

% 0000 0110 0110 0110 => $ 0666

(16 bits)

$ DEAD BEEF => % 1101 1110 1010 1101
% 1011 1110 1110 1111

(32 bits)

Introduction/Rappels

- Hexadécimal :

Nombre négatif si chiffre à gauche est supérieur à 7

$ 70	=>	% 0111 0000	=>	112
$ 80	=>	% 1000 0000	=>	128 ou -128
$ 42	=>	% 0100 0010	=>	66
$ AB	=>	% 1010 1011	=>	171 ou -85

Introduction/Rappels

- Incrémenter / Décrémenter :
  (Increase / Decrease)

  Augmenter de 1 / Réduire de 1

Incrémenter 42 => 42 + 1 = 43

Décrémenter 42 => 42 – 1 = 41

Introduction/Rappels

- Overflow : Dépassement de la valeur maximale

  Sur 8 bits, prenons 255 en non-signé : % 1111 1111
  Incrémentons-le ! (255 + 1 = 256) % 1111 1111 + 1
  % 0000 0000 => 255 + 1 = 0 (sur 8 bits)

  On a fait un « overflow » (on a dépassé la valeur maximale)

  Fonctionne sur les signés : sur 8 bits, 127 + 1 = -128
  (sur N bits, MAX + 1 = MIN)

Introduction/Rappels

- NaN : Not A Number

  Exception exprimant une valeur impossible à gérer.

  Sur les entiers, il s'agit souvent des divisions par 0. (ou de la racine carrée d'un nombre négatif)

  Sur les flottants, il s'agit des valeurs trop petites, ou du résultat d'opérations avec une valeur NaN.

Introduction/Rappels

- RAM : Random Access Memory
  Mémoire modifiable par de simples instructions

- ROM : Read-Only Memory
  Mémoire en lecture seule
  Inscriptible par une manipulation très précise

Introduction/Rappels

- Flottants (float ou double) : nombres à virgule

  Est stocké sur du binaire lui aussi…
  Mais dans des formats spécifiques : IEEE-754
  - Simple précision (32 bits)
  - Double précision (64 bits)
  - Quadruple précision (128 bits)

- Précision imparfaite : il est difficile de comparer deux flottants proches (il faut choisir un écart minimum)

Introduction/Rappels

- Flottants (float ou double) : nombres à virgule
- 3 parties : Signe, Exposant, Mantisse
  Signe : 1 bit
  Exposant : 8 bits (simple prec.) / 11 bits (double prec.) / …
  Mantisse : 23 bits (simple prec.) / 52 bits (double prec.) / …
- Exceptions détectables :
  - Zéro : Exposant à 0, Mantisse à 0 (+0 et -0)
  - Infini : Exposant au max (plein de 1), Mantisse à 0 (+∞ et -∞)
  - NaN : Exposant au max (plein de 1), Mantisse non nulle

  (dans les flottants IEEE-754, 1 divisé par 0 renvoie +∞, et non pas NaN)
  (mais 0 divisé par 0 renvoie un NaN)

Introduction/Rappels

- Flottants (float ou double) : nombres à virgule

- Exposant à 0, Mantisse non nulle : nombre dénormalisé
  - (32bits) Entre $1,4\times10^{-45}$ et $1,17549421\times10^{-38}$
  - Nombres très proches de 0

- Exposant non nul (mais pas au max) : nombre normalisé
  - (32bits) Entre $1,17549435\times10^{-38}$ et $3,40282346\times10^{38}$
  - Nombres allant de « proches de 0 » à « grands nombres »

- Codent des nombres plus grands et plus petits que les entiers

Introduction/Rappels

- Flottants (float ou double) : nombres à virgule

- Il est difficile de comparer deux flottants !
  - On peut comparer deux flottants « éloignés »
  - On ne peut pas faire d'égalité stricte entre deux flottants (on peut, mais c'est complexe)
  - Il vaut mieux observer l'écart entre deux flottants, et voir si celui-ci est suffisamment petit pour considérer les deux nombres comme égaux (ou très proches)

Introduction/Rappels

- Unités : bit(s) et octet(s)
- Le « bit » est l'unité la plus petite
  - Un « bit » est dans l'état « 0 » ou « 1 »
- 8 bits = 1 octet
  - 16 bits = 2 octets
  - 32 bits = 4 octets
  - 64 bits = 8 octets
  - …

Introduction/Rappels

- Préfixes binaires (CEI) :
  - Kibi (Ki) : 2^10 = 1 024
  - Mébi (Mi) : 2^20 = 1 048 576
  - Gibi (Gi) : 2^30
  - Tébi (Ti) : 2^40
  - Pébi (Pi) : 2^50
  - Exbi (Ei) : 2^60
  - Zébi (Zi) : 2^70
  - Yobi (Yi) : 2^80

(Erreur)

(2%)

(5%)

(7%)

(10%)

(13%)

(15%)

(18%)

(21%)

- : Préfixes décimaux (SI)
  - 1 000 = 10^3 : (k) Kilo
  - 1 000 000 = 10^6 : (M) Méga
  - 10^9 : (G) Giga
  - 10^12 : (T) Téra
  - 10^15 : (P) Péta
  - 10^18 : (E) Exa
  - 10^21 : (Z) Zetta
  - 10^24 : (Y) Yotta

Introduction/Rappels

- Dans la pratique…

On utilise la dénomination du SI, mais avec les mesures CEI…

1 Mo (1 Méga-octet) = 1024 Ko (Kilo-octet) = 1 048 576 octets

(Mais regardez bien vos disques durs, et les étiquettes dessus…)

Enfin rien n'est vraiment normalisé, en fait…

Introduction/Rappels

- Autres mesures :

- Mbps/kbps : Mega bits par seconde / Kilo bits par seconde

  8 Mbps = 1 Mo/s

  - Utilisé pour mesurer les débits de transmission (réseaux & interfaces physiques)

- Baud (Bd) :

  1 Mbauds

  - Mesure le débit de « symboles » transmis sur la porteuse du signal (réseaux & télécoms)

Introduction/Rappels

- ASCII :

  American Standard Code for Information Interchange

  – 128 caractères (dont 95 imprimables)

  – Encodage « de base » sur UNIX
    (et beaucoup d'autres)

  – Codage sur 7 bits (ignore le bit de poids de fort)
    char c <= parfait pour ASCII

Introduction/Rappels

Table ASCII

(code hexadécimal)

hexa	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VF	FF	CR	SO	SI
10	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
20	SP	!	"	#	$	%	&	‘	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Introduction/Rappels

- Quelques autres encodages existant :
  - Unicode, UTF-8, UTF-16, UTF-32

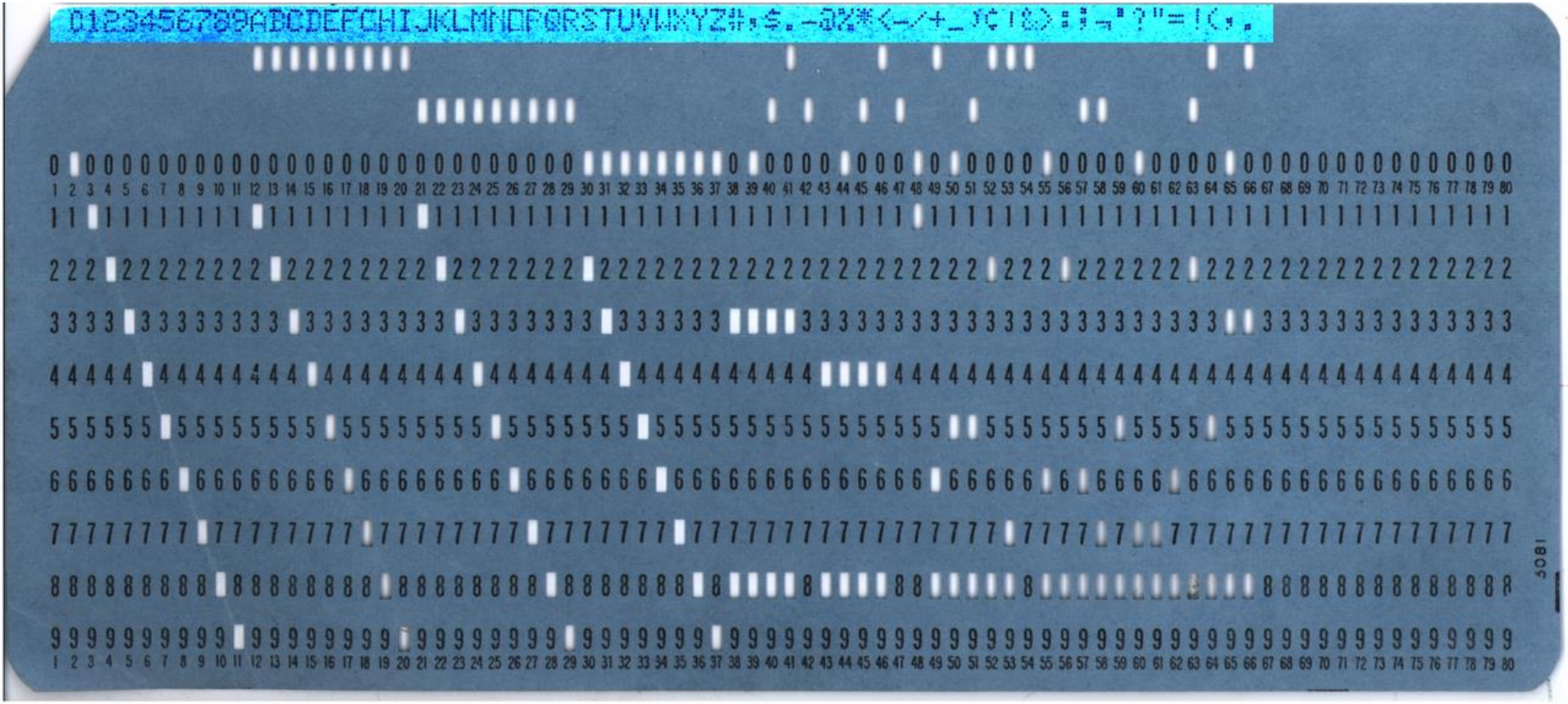
Unicode	UTF-8	UTF-16	UTF-32
char[4]	char	char[2]	char[4]
wchar_t[2]		wchar_t	wchar_t[2]

  - EBCDIC (et ses dizaines/centaines de variantes)
    - *Extended Binary Coded Decimal Interchange Code*
    - Hérité du codage BCD
    - 8 bits
    - Approximativement : un EBCDIC par pays…

Introduction/Rappels

EBCDIC

0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ#,$.-@%*<-/+_ ¢ &):;¬ ?"=|(,.



Introduction/Rappels