Architecture des Ordinateurs

2ème Année Licence Informatique

Représentation des Données

Travaux Dirigés #1

1. Contrôle préalable

Cette partie est conçue comme une vérification pour vous permettre de déterminer si vous comprenez les concepts déjà vus en 1ère année. Veuillez répondre par « Vrai » ou « Faux » aux questions suivantes et inclure une explication :

1.1. Selon le contexte, le même ensemble de bits peut représenter des choses différentes.
Vrai ! Le même ensemble de bits peut être interprété de différentes manières ! Cela peut représenter un nombre non signé ou signé ou même autre chose comme un caractère. Tout dépend de l'interprétation que nous donnons à cet ensemble.

1.2. Il est possible d'obtenir une erreur de « dépassement de capacité » quand on ajoute deux nombres de signes opposés dans la représentation en complément à deux.
Faux ! Les erreurs de « dépassement de capacité » se produisent uniquement lorsque le résultat correct de l'addition se situe en dehors de la plage de $[-2^{n-1}, 2^{n-1}-1]$, pour une représentation sur n bits. L'addition de nombres de signes opposés ne produira jamais de nombres en dehors de cette plage.

1.3. Si vous interprétez les nombres négatifs dans la représentation en complément à deux comme étant des nombres non signés, alors les valeurs de ces nombres seraient plus petites que les nombres positifs.
Faux ! Dans la représentation en complément à deux, le bit le plus significatif est toujours égale à 1 pour un nombre négatif. Cela signifie que tous les nombres négatifs du complément à deux seront plus grands que les nombres positifs dans une représentation non signée.

2. Entiers non signés

Soit une représentation non signée à n chiffres $d_{n-1}d_{n-2}\ldots d_0$ dans la base r. Alors la valeur de ce nombre est $\sum_{i=0}^{n-1} r^i d_i$. Cette notation mathématique signifie simplement que pour représenter un nombre donné dans une base r, nous utilisons les unités $r, r^2, \ldots$ au lieu des unités 10, 100, etc. Dans le cas des bases binaire, décimal et hexadécimal, r aura les valeurs 2, 10 et 16, respectivement.

2.1. Convertir les nombres suivants depuis leur base initiale vers les deux autres bases. C.-à-d. Si la base initiale est une représentation en binaire alors donnez les représentations équivalentes en décimale et en hexadécimale et ainsi de suite.

a) $(10010011)_2$ = $(147)_{10}$ = $(93)_{16}$

b) $(63)_{10}$ = $(0011\ 1111)_2$ = $(3F)_{16}$

c) $(00100100)_2$ = $(36)_{10}$ = $(24)_{16}$

d) $(0)_{10}$ = $(0)_2$ = $(0)_{16}$

e) $(39)_{10}$ = $(0010\ 0111)_2$ = $(27)_{16}$

f) $(437)_{10}$ = $(0001\ 1011\ 0101)_2$ = $(1B5)_{16}$

g) $(0123)_{16}$ = $(0000\ 0001\ 0010\ 0011)_2$ = $(291)_{10}$

2.2. Convertir les nombres suivants de la représentation hexadécimale vers la représentation équivalente en binaire

a) $(BAD)_{16}$ = $(1011\ 1010\ 1101)_2$

b) $(F00D)_{16}$ = $(1111\ 0000\ 0000\ 1101)_2$

c) $(FACE)_{16}$ = $(1111\ 1010\ 1100\ 1110)_2$

d) $(0FF)_{16}$ = $(0000\ 1111\ 1111)_2$

3. Entiers signés

En binaire, le schéma non signé n'est pas très bien adapté pour représenter, au même temps, des nombres positifs et négatifs. En ce sens, un certain nombre de schémas différents ont été inventés pour représenter les nombres signés, mais nous allons nous limiter au schéma de la représentation en complément à deux.

- Le bit le plus significatif dans une représentation en complément à deux représente un nombre négatif. Tous les autres bits sont positifs. Ainsi, la valeur d'un nombre à n chiffres en complément à deux peut être écrite comme : $-2^{n-1}d_{n-1} + \sum_{i=0}^{n-2} 2^i d_i$.
- Une astuce pour trouver le complément à deux d'un nombre après une inversion de signe : basculer tous les bits et ajouter 1.
- L'opération d'addition est effectuée exactement de la même manière qu'avec un nombre non signé.
- Le nombre zéro possède une seule représentation en complément à deux : $(000\ldots0)_2$.

Pour les questions (1 à 3), considérez **une représentation sur 8 bits** et donnez vos réponses pour les cas signés et non-signés. Si vous jugez que la question ne possède pas de réponse alors indiquez cela par un "N / A" (signification : Non Applicable).

3.1. Quel est le plus grand entier ? Quel sera le résultat si on ajoute la valeur 1 à ce nombre ?

non signé ? 255 , 0

signé ? 127 , -128

3.2. Donnez les représentations des nombres, $(0)_{10}$, $(1)_{10}$ et $(-1)_{10}$

non signé ? $(0000\ 0000)_2$, $(0000\ 0001)_2$, N / A

signé ? $(0000\ 0000)_2$, $(0000\ 0001)_2$, $(1111\ 1111)_2$

3.3. Donnez les représentations des nombres, $(33)_{10}$, et $(-33)_{10}$

non signé ? $(0010\ 0001)_2$, N / A

signé ? $(0010\ 0001)_2$, $(1101\ 1111)_2$

3.4. Quel est le plus petit nombre négatif représentable sur 8 bits ?

en binaire ? $(1000\ 0000)_2$

en décimal ? -128

3.5. Calculez $(33)_{10}$ et $(-33)_{10}$ en binaire sur 16 bits

a) $(33)_{10}$ = $(0000\ 0000\ 0010\ 0001)_2$

b) $(-33)_{10}$ = $(1111\ 1111\ 1101\ 1111)_2$

3.6. Comment passons-nous d'une représentation sur 8 bits à une représentation sur 16 bits ? Inversement, à quelle condition pouvons-nous passer d'une représentation sur 16 bits à une représentation sur 8 bits ?

Pour passer d'une représentation en complémentent à deux sur 8-bits à une représentation sur 16-bits, nous opérons une extension de signe. C.-à-d. le bit n°7 de l'octet est répliqué dans les positions 8 à 15 dans la représentation 16-bits.

Pour tronquer une représentation 16-bits à une représentation sur 8-bits, il est nécessaire que les bits 8 à 15 soient identiques au bit n°7.

4. Arithmétique en binaire

4.1. Toujours sur 8 bits et en complément à deux, calculez les additions suivantes en prenant soin de noter les retenues. Donnez la valeur décimale de chaque résultat :

a) 14 + 59 = (0100 1001)$_2$ = (73)$_{10}$, retenue = 0

b) 59 + 80 = (1000 1011)$_2$ = (-117)$_{10}$, retenue = 0

c) 59 + (-80) = (1110 1011)$_2$ = (-21)$_{10}$, retenue = 0

d) -59 + (-14) = (1011 0111)$_2$ = (-73)$_{10}$, retenue = 1

e) -59 + (-80) = (0111 0101)$_2$ = (117)$_{10}$, retenue = 1

f) 59 + (-59) = (0000 0000)$_2$ = (0)$_{10}$, retenue = 1

4.2. Certains des résultats obtenus à la question précédente ne sont pas conformes à l'attente (dépassement de capacité, ou « overflow » en anglais). Que s'est-il passé ? Quel test simple permet de détecter ces dépassements de capacité ?
Dans les opérations signées, un « overflow » se produit quand nous additionnons deux valeurs ayant le même signe mais le résultat ait un signe opposé. Pour détecter donc un « overflow », il faut comparer les bits de signes des opérandes avec celui du résultat.

La « retenue » indique un « overflow » pour les opérations NON signées, seulement.

5. Représentation des données

Quel est le plus petit nombre de bits nécessaires pour représenter les plages suivantes en utilisant n'importe quel schéma de représentation numérique (justifiez !).

a) 0 à 256
n bits peuvent être utilisés pour représenter tout au plus 2^n choses distinctes. Ainsi, 8 bits peuvent représenter $2^8 = 256$ nombres. Cependant, la plage 0 à 256 contient en fait 257 nombres, donc, nous avons besoin de 9 bits.

b) -7 à 56
Plage de 64 nombres qui peuvent être représentés avec 6 bits parce que $2^6 = 64$.

c) 64 à 127
Ici, 127-64+1 = 64 nombres à représenter au total, donc 6 bits.

d) -64 à -127
Même réponse, 6 bits pour 64 nombres distincts.

6. Code ASCII

L'**American Standard Code for Information Interchange** (ASCII) est une norme informatique de codage de caractères apparue dans les années 1960. Cette norme définit 128 codes à 7 bits pour représenter des caractères comprenant des codes de synchronisation (de 0 à 31), les chiffres de 0 à 9, les 26 lettres de l'alphabet latin en minuscules et en majuscules, et des symboles mathématiques et de ponctuation (Figure ci-dessous).

Bits		$b_6b_5b_4$							
		000	**001**	**010**	**011**	**100**	**101**	**110**	**111**
$b_3b_2b_1b_0$	**0000**	(NUL)	(DLE)	(SP)	0	@	P	`	p
	0001	(SOH)	(DC1)	!	1	A	Q	a	q
	0010	(STX)	(DC2)	"	2	B	R	b	r
	0011	(ETX)	(DC3)	#	3	C	S	c	s
	0100	(EOT)	(DC4)	$	4	D	T	d	t
	0101	(ENQ)	(NAK)	%	5	E	U	e	u
	0110	(ACK)	(SYN)	&	6	F	V	f	v
	0111	(BEL)	(ETB)	'	7	G	W	g	w
	1000	(BS)	(CAN)	(	8	H	X	h	x
	1001	(HT)	(EM)	)	9	I	Y	i	y
	1010	(LF)	(SUN)	*	:	J	Z	j	z
	1011	(VT)	(ESC)	+	;	K	[	k	{
	1100	(FF)	(FS)	,	<	L	\	l	|
	1101	(CR)	(GS)	-	=	M	]	m	}
	1110	(SOH)	(RS)	.	>	N	^	n	~
	1111	(SI)	(US)	/	?	O	_	o	(DEL)

Figure 1. Table ASCII

6.1. A partir du tableau, donnez le code en binaire sur 7 bits des lettres 'a' et 'A'

'a' : $(110\ 0001)_2$, 'A' : $(100\ 0001)_2$

6.2. Faites la même chose pour les paires de lettres ('b', 'B') et ('c', 'C'). Que remarquez-vous ?

'b' : $(110\ 0010)_2$, 'B' : $(100\ 0010)_2$, 'c' : $(110\ 0011)_2$, 'C' : $(100\ 0011)_2$

6.3. Si un caractère ASCII représentait un chiffre décimal, quelle valeur, en décimal, pourrait prendre le code de ce caractère ? Comment déduire la valeur décimale représentée par ce caractère ?

Les caractères décimaux '0' à '9' sont codés avec les valeurs $(011\ 0000)_2$ à $(011\ 1001)_2$ en binaire. En décimal, cela donne la plage des valeurs 48 à 57. Pour trouver la valeur décimale associée à un caractère décimal, une solution consiste à soustraire la valeur 48 au code ASCII de ce caractère.