

Codage des entiers naturels et relatifs

Année universitaire 2024–2025

1 Puissances de 2

1. Donner la valeur de 2^i pour $i \in [0, 16]$.
2. Donner la valeur de 2^i pour $i \in [-4, 0[$.
3. En constatant que $2^{10} \simeq 10^3$, donner le nombre de chiffre nécessaire pour écrire en base 10 les nombres de la forme 2^i pour $i \in \{20, 30, 32, 64, 128\}$.
4. Combien de bits faut-il pour écrire les nombres suivants : 1234 ? 12345678 ? 10^{12} ?

2 Unités de mesure de l'information

1. Rappeler l'origine des noms d'unité bit et octet.
2. Combien il y a-t-il de bits dans un octet ?
3. Pour chacun des symboles d'unités suivants, donner son nom complet, le nombre de bits correspondant, et préciser son cadre d'utilisation usuel : Mbit, MB, MiB.
4. Sur les machines modernes, la mémoire est organisée en pages de 4 KiB. Ces pages sont découpées en mots mémoire de 64 bit. Combien de mots il y a-t-il par page ? Combien d'octets ? Combien de bits ?

3 Notation positionnelle en base 2, 10 et 16

1. Compléter le tableau ci-dessous (note : en base 2, ajouter si nécessaire des 0 non significatifs à gauche pour toujours avoir 8 bits et présenter vos résultats en groupant les bits par 4.).

Base 2	Base 10	Base 16
0000 0010		
	10	
		10
0101 1010		
	42	
	100	
0111 1111		
		23
		AB
		FF

- Donner le codage en base 16 de $1101\,0010\,1111\,0011\,1010\,0111\,1100\,0010_2$
- Donner le codage en base 2 de $16\,1234_{16}$
- Donner le codage en base 2 de $DEADBEEF_{16}$

4 Arithmétique binaire sur 8 bits

On se place en base 2. Faire les calculs ci-dessous sans passer par la base 10. Vérifier ensuite les résultats en passant par la base 10.

- $0100\,0011 + 0111\,1001$
- $1010\,1010 + 1001\,0010$
- $0001\,0100 \times 0000\,1100$

5 Décodage et encodage en complément à 2

- Les nombres ci-dessous sont codés en complément à 2 sur 8 bit. Pour chacun, donner sa valeur en base 10.

- | | | |
|--------------|--------------|--------------|
| a) 0000 0000 | c) 1000 0000 | e) 1001 1010 |
| b) 0110 1001 | d) 1110 1001 | f) 1111 1111 |

- Les nombres ci-dessous sont en base 10. Pour chacun, donner son codage en complément à 2 sur 8 bits (à écrire en notation binaire et hexadécimale) et 16 bits (en notation hexadécimale uniquement).

- | | | |
|-------|--------|---------|
| a) 1 | c) 42 | e) 127 |
| b) -1 | d) -42 | f) -128 |

- Coder le nombre -4 en complément à 2 :

- | | |
|--|--|
| a) sur 4 bits (à écrire en notation binaire) | c) sur 16 bits (à écrire en notation hexadécimale) |
| b) sur 8 bits (à écrire en notation binaire et hexadécimale) | d) sur 64 bits (à écrire en notation hexadécimale) |

4. Les nombres ci-dessous sont en base 10. Pour chacun, donner son codage en complément à 2 sur 32 bits (à écrire en notation hexadécimale).

a) 17

b) -17

c) -103

d) -65 536

6 Bornes minimales et maximales

- Donner la valeur (en bases 2, 10 et 16) du plus grand et du plus petit nombre que l'on peut écrire en notation positionnelle en base 2 sur 8 bit, 16 bit, 32 bit et 64 bit.
- Même question mais en considérant cette fois un codage en complément à 2.

7 Arithmétique en complément à 2 : calcul de l'opposé

- Compléter le tableau ci-dessous.

x	x_{c2}	$(-x)_{c2}$
25		
42		
100		

- Vérifier que $x_{c2} + (-x)_{c2} = 0_{c2}$
- Que vaut $x_{c2} + \overline{x_{c2}}$? En déduire une technique simple pour calculer l'opposé d'un nombre dans le codage en complément à 2. Calculer les opposés de -1, -42 et -128.

8 Arithmétique en complément à 2 et bits de status

On considère un additionneur 32 bits qui renseigne à chaque calcul 3 bits de status (en plus des sorties classiques S et C) :

- N (négatif) est à 1 ssi, interprété en complément à 2, le résultat est négatif ;
 - Z (zéro) est à 1 ssi le résultat est nul ;
 - V (oVerflow) est à 1 ssi, interprété en complément à 2, le résultat du calcul a « débordé ».
- Expliquer comment calculer N , Z , et V à partir des entrées et sorties de l'additionneur. Donner le schéma des circuits correspondants.
 - Compléter le tableau ci-dessous (les opérandes sont données en notation hexadécimale, le résultat est à donner dans la même notation).

Opération	Résultat	N	Z	C	V
70000000 + 70000000					
90000000 + 90000000					
80000000 + 80000000					
00001234 - 00001000					
00000004 - 00000005					
C3314150 - 96694242					

9 Les types entiers en Go

Étudier le code des deux fonctions go ci-dessous et prédire les affichages.

```
func PrintInt() {
    fmt.Println("Différents affichages d'une même constante entière")

    a := 9760
    fmt.Printf("%T, %#v, %d, %c, %O, %#x, %b\n", a, a, a, a, a, a, a)
}

func CastInts() {
    var a uint32
    var b int64
    var c int32
    var d int64
    var e int64
    var f int8

    fmt.Println("Conversion entre types entiers")

    // conversions
    a = 123 << 25 | 255
    b = int64(a)
    c = int32(a)
    d = int64(a)
    e = int64(c)
    f = int8(a)

    /// affichages
    fmt.Printf("a : %#v -- %#x\n", a, a)
    fmt.Printf("b : %#v -- %#x\n", b, b)
    fmt.Printf("c : %#v -- %#x\n", c, c)
    fmt.Printf("d : %#v -- %#x\n", d, d)
    fmt.Printf("e : %#v -- %#x\n", e, e)
    fmt.Printf("f : %#v -- %#x\n", f, f)
}
```