Systèmes Digitaux

Cours 1

Mise en contexte

- De nos jours, le monde devient de plus en plus électronique
  - Ex: Les fours, les montres, les voitures, etc.
  - Il existe 2 types d'électronique : numérique et analogique
- L'électronique numérique prend de plus en plus d'importance.
  - La musique, la télévision, les films… sont devenus numériques

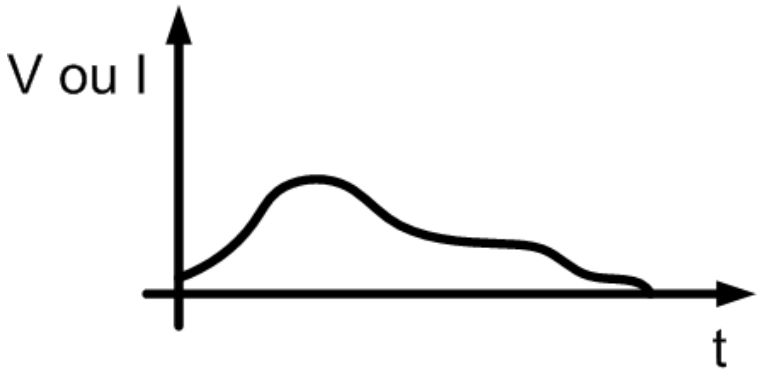
Pourquoi?

Mise en contexte

- Le numérique a plusieurs avantages:
  - Stockage de données compacte
  - Transmission plus rapide (fibre optique)
  - Plus robuste et fiable
  - Conception plus facile et rapide…
- Ce cours va se concentrer sur la base des systèmes numériques (digitaux)

Analogique et Numérique

- En électronique analogique, les voltages et courants
  - Peuvent avoir n'importe quelle valeur
  - Peuvent changer n'importe quand
  - Ce sont des valeurs <u>analogiques en temps continu</u>

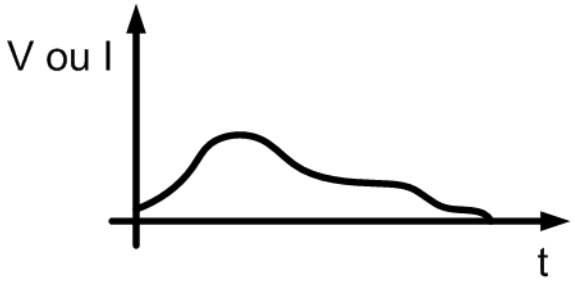


Analogique et Numérique

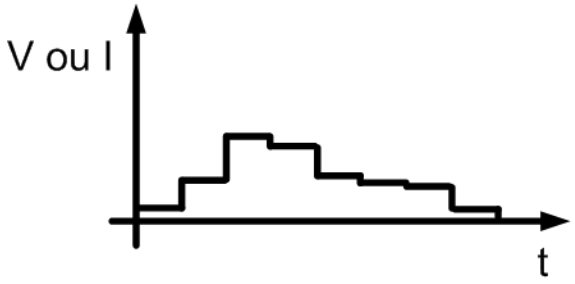
- En électronique analogique les valeurs exactes sont importantes:
  - On veut connaître la valeur “exacte” des tensions
  - On veut connaître la valeur “exacte” des courants
- En temps continu, on veut connaître les valeurs en tout temps
  - Au temps 0s
  - Au temps 0.001s
  - Au temps 0.002s (même entre ces instants)

Analogique et Numérique

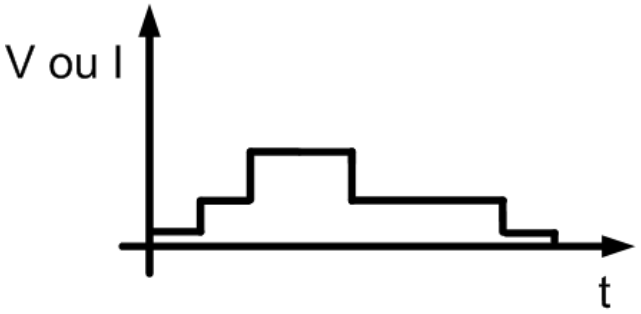
- Les systèmes digitaux ont 2 caractéristiques:
  - Temps discret: on ne regarde la valeur qu'une fois de temps en temps
  - Valeur quantifiée: on n'a pas besoin d'être infiniment précis (il y a des plages de valeurs)



Temps continu



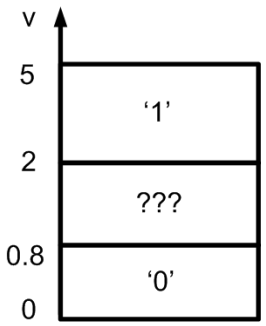
Temps discret



Numérique

Analogique et Numérique

- Dans un système digital, il y a un nombre bien determiné de plages de valeurs:
- Dans un cas simple, on a 2 niveaux. Ex:
  - 0v à 2.5v sera ‘0’
  - 2.5v à 5v sera ‘1’
- Il existe d’autres conventions:
  - Par exemple, en TTL, les ‘0’ et ‘1’ sont definis comme ceci



Analogique et Numérique

- On sait qu'il existe plusieurs conventions pour determiner '0' et '1'
  - Dans ce cours, on ne se préoccupe pas des détails
  - On va simplement dire '0' ou '1' sans savoir les valeurs exactes des tensions et des courants
- Chaque chiffre de '0' ou '1' est appelé un bit (Binary digIT)

Analogique et Numérique

- Avec un bit, on peut avoir ‘0’ ou ‘1’
- En collant 2 bits ensemble, on peut avoir “00”, “01”, “10” et “11”…
    - En ajoutant plus de bits, on a plus de valeurs possibles
- C’est la même chose avec les nombres décimales:
    - Un chiffre va de 0 à 9
    - Deux chiffres vont de 0 à 99…

Analogique et Numérique

À force de se servir des nombres binaires, on arrive souvent à se rappeler des correspondances jusqu'à 4 bits.

Ce serait un bon objectif d'apprentissage…

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

Analogique et Numérique

- Est-ce qu'il existe seulement les bits et les nombres décimales?
  - Non. Il existe une infinité de "bases"
- Une base définit le nombre de valeurs possibles
- Une base binaire dit qu'il y a 2 valeurs possibles (0 et 1)
- Une base décimale dit qu'il y en a 10…

Les bases

- Il y a une infinité de bases, mais certaines nous intéressent particulierement
  - Base 2: binaire
  - Base 10: décimale
  - Base 8: Octale
  - Base 16: Hexadécimale

Commecons par les décimales qu'on connait bien…

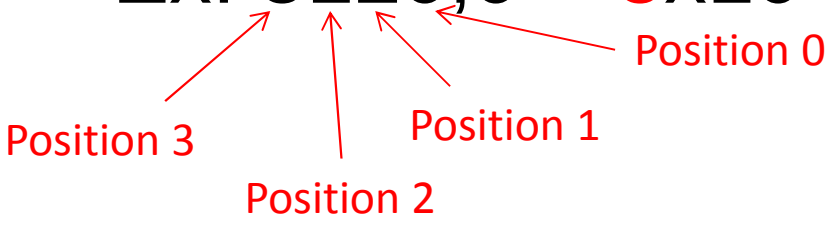
Décimal

- On est habitué à s'exprimer en nombres décimales: 132, 2, 21, 0.7, etc.
  - Chaque chiffre a une signification
  - On le prend souvent pour acquis…
- Dans 132, "1 est avant 3 qui est avant 2"
  - 1 représente les centaines (x 10^2)
  - 3 représente les dizaines (x 10^1)
  - Et 2 représente les unités (x 10^0)
  - Donc, $1x10^2 + 3x10^1 + 2x10^0 = 132$

On fait cette opération sans s'en rendre compte...

Décimal

- On peut formaliser le processus en faisant l'opération suivante:
    1) On repère la virgule délimitant les décimales
    2) L'élément à la gauche est la position 0
    3) Son poids est 10^0
    4) À sa gauche, c'est l'élément 1: son poids est 10^1
    5) À sa gauche, c'est l'élément 2: son poids est 10^2
    6) ...
- Ex: $5120{,}0 = 5\text{x}10^3+1\text{x}10^2+2\text{x}10^1+0\text{x}10^0$



Décimal

- On remarque que le chiffre est toujours multiplié par 10^x, où x est la position
  - 10^0, 10^1, 10^2, 10^3…
- Le 10 est là parce qu'on est en base 10
  - Si c'était en base 2, on aurait mis 2^x
- Par exemple:
  - En binaire, 1101 serait $1x2^3+1x2^2+0x2^1+1x2^0=13$

Binaire

- En décimale, il existe 10 valeurs de 0 a 9
  - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9...
  - Une fois arrive à 9, ça recommence a 0
  - Le chiffre à sa gauche augmente de 1: 10
  - Et le cycle continue: 10, 11, 12, 13, 14...
- En binaire, il existe 2 valeurs de 0 à 1
  - 0, 1
  - Une fois arrive à 1, ça recommence à 0
  - Le chiffre à sa gauche augmente de 1: 10
  - Et le cycle continue: 10, 11, 100, 101, 110,…

Binaire

- En binaire, ça commence à 0 et puis ca monte a 1... meme chose avec le decimal
  - Donc, 0 binaire correspond a 0 en decimale
  - Et 1 binaire correspond à 1 en décimale
- Par la suite, en binaire, ça devient 10:
  - Donc, 10 binaire représente 2 en decimale
  - Et 11 binaire représente 3 en decimale
  - Il est donc possible de trouver une correspondance entre binaire et décimale

Binaire

- Pour connaître la valeur d'un nombre binaire, on utilise la méthode précédente
- Pour trouver les valeurs décimales, nous avions procédé comme suit:
  - Ex: 132 = $1x10^2+3x10^1+2x10^0$
  - Rappel: 10 c'est la base décimale

On peut faire la même chose pour le binaire…

Binaire

- En binaire, on est en base 2:
  - On aura 2^x à la place de 10^x
- Donc, pour traduire 1101, il faudrait:
  - Identifier la virgule et savoir que les bits seront associés à 2^3, 2^2, 2^1 et 2^0…
  - 1101= $1x2^3 + 1x2^2 + 0x2^1 + 1x2^0$
  - Ça se traduit en 8+4+1=13
  - Donc, 1101 en binaire c'est 13 en décimale

Exemple (seul)

- Convertissez 10111 du binaire en décimale

Exemple (seul)

- 10111: il y a 5 chiffres a gauche de la virgule
  - De gauche à droite, il y aura: 2^0, 2^1, 2^2, 2^3 et 2^4
  - Donc $1x2^4+0x2^3+1x2^2+1x2^1+1x2^0$
- En faisant les calculs simples, on obtient:

$$16+4+2+1=23$$

Binaire

- Il existe un truc pour faciliter le travail
  - On peut voir que $2^0=1$, $2^1=2$, $2^2=4$, $2^3=8$, $2^4=16$, ...
  - Et on continue à doubler à chaque fois
- À la place d'écrire l'équation au long, cette méthode pourrait être plus rapide:
  - Survoler le nombre binaire et mettre 1, 2, 4, 8, 16... à côté de chaque chiffre correspondant
  - Si on voit '1', on ajoute la valeur à la somme
  - Si on voit '0', on passe au suivant sans rien faire

Illustrons ca avec un exemple...

Exemple

- Convertissez 11001 du binaire en décimale

Exemple

- On commence par mettre les valeurs:

16 8 4 2 1

1 1 0 0 1

- Il y a ‘1’ a côté de 1, 8 et 16..
  - On les additionne ensemble: 1+8+16=25
  - Donc, 11001 est égal a 25 en décimal

Octal et hexadecimal

- En plus du décimale et du binaire, on s'intéresse aussi à 2 autres bases..
  - Octal et hexadécimal
- Octal (base 8): valeurs allant de 0 à 7
  - 0,1,2,3,4,5,6,7,10,11,12,13,14,15,16,17,20…
- Hexadécimal (base 16): valeurs allant de 0 à F
  - 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F,10,11,12,13,14…
  - A est égal à 10, B est égal à 11…

On utilise la même technique que tantôt: $2x8^1+4x8^0=...$

Pourquoi octal et hexadecimal?

- On est habitué aux décimales depuis qu'on est jeune: ça justifie la base 10
- Les ordinateurs utilisent 0 et 1: ça justifie la base 2
- Pourquoi base 8 et base 16?

Pourquoi octal et hexadécimal?

- À l'aide de 1 bit, on va de 0 a 1
  - Base 2
- À l'aide de 2 bits, on va de 0 a 3
  - Base 4
- À l'aide de 3 bits, on va de 0 a 7
  - Base 8 (octal)
- À l'aide de 4 bits, on va de 0 a 15
  - Base 16 (hexadécimal)

Pourquoi octal et hexadécimal?

- Pensez aux nombres décimales qui vont de 0 a 9
  - On a besoin de 4 bits
  - Cependant, 4 bits peuvent representer jusqu’à 15
  - On “gaspille” donc les nombres 10 à 15…
- Les nombres hexadécimales sont “optimales”
  - Ils utilisent tous les bits à leurs pleins potentiels
  - Avec 3 bits, l’optimal, c’est la base 8 (octal)

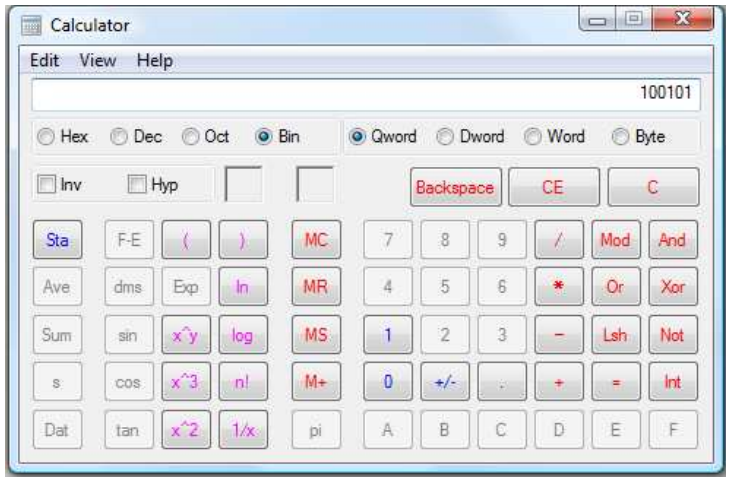
Choses pratiques a savoir

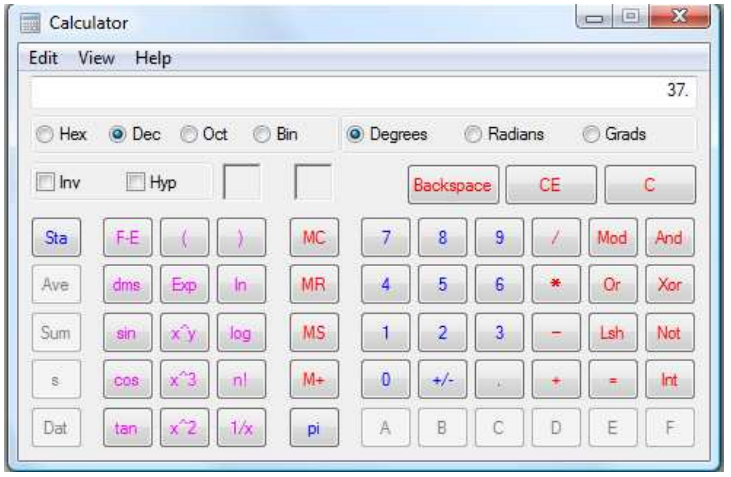
- Information sur le nombre de valeurs et la valeur maximale
- Pour un nombre de n bits…
  - Il y a 2^n valeurs possibles
  - La valeur maximale est 2^n-1
- Donc…
  - Avec 1 bit, il y a 2 valeurs allant de 0 a 1
  - Avec 2 bits, il y a 4 valeurs allant de 0 a 3
  - Avec 3 bits, il y a 8 valeurs allant de 0 a 7

Familiarisez-vous avec ca…

Choses pratiques a savoir

- La calculatrice peut nous aider:
  - On se met en binaire
  - On entre le chiffre qu'on veut convertir
  - On pèse sur décimal
  - La conversion est automatique





Note: Il faut etre en mode "scientific"

Exemple (seul)

- Convertissez 24 (octal) en décimal
- Et convertissez 7A (hexa) en décimal

Exemple (seul)

- On utilise la même méthode que tantôt

- En octal, on a 24:
  - $2x8^1+4x8^0=20$ en décimal
- En hexadécimal, on a 7A:
  - $7x16^1+Ax16^0$
  - Il faut juste se rappeler que A=10…
  - $7x16^1+10x16^0=122$

C’est une bonne idée de vérifier avec la calculatrice…

Différentes bases

- Il peut souvent y avoir confusion dans la notation:
  - Par exemple: “Convertissez 101 en décimale”
  - 101 est exprimé en quelle base?
- On voit donc la nécessite de SPÉCIFIER la base
- On met les valeurs entre parenthèses et un petit chiffre pour la base:
  - $(101)_2$, $(101)_8$, $(101)_{10}$ ou $(101)_{16}$…

Conversion inverse

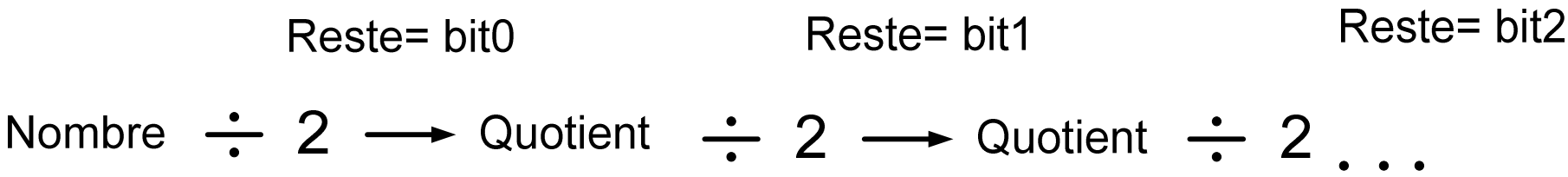
- On sait comment interpréter les valeurs en bases diverses
  - Ex: “$(1011)_2$ est égal a quoi en décimale?”
  - Trouver la virgule et identifier les poids
  - Multiplier par le poids et additionner…
- Sommes-nous capables de faire l’opération inverse?
  - Comment représenter $(72)_{10}$ en binaire?

Conversion inverse

- Dans un exemple précédent, on a pris $(11001)_2$ et on l'a convertit en $(25)_{10}$
  - On a fait $1x2^4+1x2^3+0x2^2+0x2^1+1x2^0$
  - On a utilisé la multiplication
- La question est "avec $(25)_{10}$, comment trouver $(11001)_2$?"
  - On utilise la division!

Division progressive

- L'approche est de diviser progressivement par la base:
  - On divise les quotients par la base jusqu'à ce que le quotient soit 0 (avec un reste)
  - Les valeurs de reste deviendront notre réponse
- Pour un nombre binaire, c'est la base 2:



Division progressive

- On pourrait résumer la technique de la façon suivante:
  1) Diviser par la base et le reste donne la valeur du bit
  2) Prendre le quotient et diviser par la base
  3) Examiner le reste (donne la valeur du bit)
  4) Repeter les etapes 2) et 3) jusqu'a ce qu'on ait fini

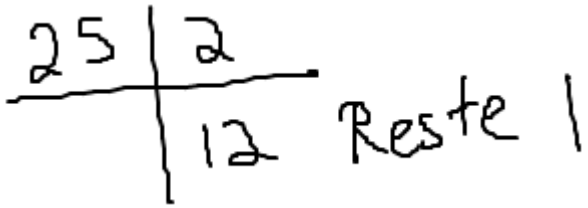
Important: Le dernier chiffre est le chiffre le plus significatif (on lit a « l'envers »)

Exemple

- Convertissez $(25)_{10}$ en binaire.

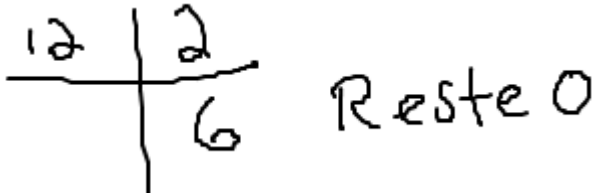
Exemple

- On commence par diviser 25 par 2:



Bit0='1'

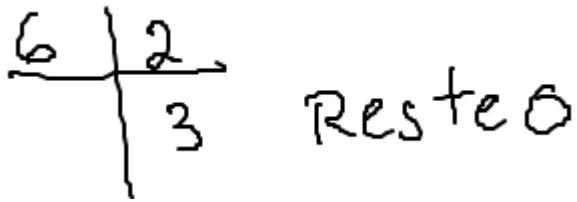
- On divise 12 par 2:



Bit1='0'

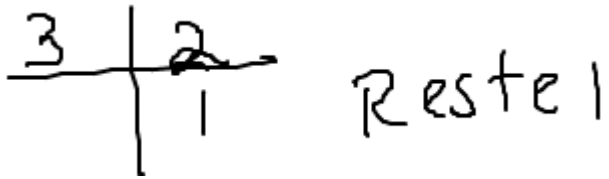
Exemple

- On divise 6 par 2:



Bit2='0'

- On divise 3 par 2:



Bit3='1'

- On divise 1 par 2:

Bit4='1'

- Le résultat est: 11001

Exemple (seul)

- Convertissez $(123)_{10}$ en hexadécimal (base 16)

Exemple (seul)

- On divise 123 par 16
  - Le quotient est 7 et le reste est 11 (ou B)
- On divise 7 par 16
  - Le quotient est 0 et le reste est 7

- La représentation hexadécimale est donc $(7B)_{16}$
  - Vérifiez à l'aide de votre calculatrice

Nombres fractionnaires

- Avec les nombres décimales, il existe des parties entières et fractionnaires
- On peut aussi utiliser des fractions en binaire

101,011

- Le poids à gauche de la virgule c'est 2^0
- Le poids à droite de la virgule c'est 2^{-1}

Nombres fractionnaires

- Donc, on pourrait identifier le poids de chaque terme et faire la somme

4 2 1 1/2 1/4 1/8

101,011

- Le résultat est 4+1+1/4+1/8=5.375

- Malheureusement, la calculatrice ne peut pas nous aider ici

Nombres fractionnaires

- On est capable de faire une conversion de binaire fractionnaire en décimale
  - Sommes-nous capable de faire la conversion inverse?
- On l'avait fait avec les nombres entiers:
  - Division progressive
  - Le faire avec les fractions n'est pas évident

Par exemple, comment convertir $(0.625)_{10}$ en binaire?

Nombres fractionnaires

- Une façon est de faire des multiplications successives par 2
  - Si le résultat est plus que 1, on note ‘1’
  - On soustrait 1 de ce résultat
  - Si le résultat était moins que 1, on note ‘0’ (on ne soustrait pas)
  - On multiplie par 2 et on répète…
- Les chiffres qu’on note nous donnent le nombre binaire après la virgule

Nombres fractionnaires

- Donc, pour 0.625, on le multiplie par 2
  - 0.625 x 2 = 1.25
  - Le premier bit est 1… On le note et on soustrait 1
  - On poursuit avec une multiplication par 2
  - 0.25 x 2 = 0.5
  - Le deuxième bit est 0… On le note
  - On le multiplie par 2
  - 0.5 x 2 = 1... Le 3e bit est 1 et on a fini…

$(0.101)_2=(0.625)_{10}$

Exemple (seul)

- Trouvez la valeur de 6.25 en binaire.

Exemple (seul)

- Les parties entières et fractionnaires se font à part (différentes methodes)
- Pour le 6:

Exemple (seul)

- Pour la partie fractionnaire, on prend 0.25:
  - On multiplie par 2: 0.5. L'entier est 0…
  - On multiple encore par 2: 1. L'entier est 1
  - On soustrait 1 et il ne reste plus rien
- Donc: $(0.25)_{10}=(0.01)_2$
- Résultat final: $(6.25)_{10} = (110.01)_2$

Nombres signés

- On sait comment représenter les chiffres en différentes bases
  - Binaire, octal, hexadecimal, etc.
- En binaire, on sait par exemple que $(3)_{10}$ c'est $(0011)_2$
  - Mais comment représenter les nombres négatifs?
- La norme est de mettre "-" devant le nombre
  - Comment exprimer '-' avec des 0 et 1??

Nombres signés

- Il existe 3 façons de représenter les nombres signés:
  - Signe-magnitude
  - Complément à 1
  - Complément à 2
- Dans les 3 cas, quand c'est positif, la représentation est la même
  - La même qu'en non-signé, en fait
  - C'est quand c'est négatif que ça diffère

Le bit le plus à gauche nous dit si c'est négatif ou pas

Nombres signés

- Le bit à gauche fonctionne de la façon suivante:
  - Si c’est ‘1’, le reste du chiffre est négatif
  - Si c’est ‘0’, le reste du chiffre est positif
- Selon la représentation choisie, il resterait à connaître l’amplitude

Nombres signés

- En “signe-magnitude”, les autres bits représentent l’amplitude
- Donc, pour représenter un nombre en signe-magnitude:
  - On exmprime l’amplitude du chiffre
  - Par la suite, on ajoute un bit à gauche (0 pour dire positif et 1 pour dire négatif)

Exemple

- Exprimez $(-3)_{10}$ et $(3)_{10}$ en utilisant la convention signe-magnitude avec 3 bits

Exemple

- Pour exprimer en signe-magnitude, on commence par exprimer l'amplitude: 11
- Par la suite, on ajoute un bit à gauche pour indiquer le signe:
  - $(3)_{10}=(011)_2$
  - $(-3)_{10}=(111)_2$

Mais $(111)_2$ pourrais aussi représenter le chiffre 7, non?

Nombres signés

- Il faudrait donc savoir 2 choses:
  - Si c'est un nombre SIGNÉ ou pas (est-ce qu'il est toujours positif ou peut-il etre négatif aussi?)
  - Le nombre de bits
- Un nombre qui peut seulement être positif est un nombre non-signé (*unsigned*)
  - Pour 4 bits, le nombre le plus petit est 0000 (0)
  - Et le nombre le plus gros est 1111 (15)

Nombres signés

- Pour un nombre signé en signe-magnitude
  - Un bit est le signe
  - Le reste est l'amplitude
- En 4 bits, min/max sont donc -/+ 2^3
  - Le plus petit chiffre est 1111 (-7)
  - Et le plus gros chiffre est 0111 (7)
- Note: 0000 et 1000 représentent 0 et -0...
  - On gaspille donc un des nombres parce qu'il existe 2 fois le nombre 0

On ne peut pas exprimer les valeurs plus élevées que 7 avec 4 bits en signe-magnitude

Exemple

- En utilisant la notation en signe-magnitude
  - Remplissez le tableau suivant en utilisant des nombres binaires de 3 bits

Décimales	**Signe-magnitude**
-3	
-2	
-1	
-0	
0	
1	
2	
3	

Exemple

- On commence par mettre les amplitudes
- Par la suite, on ajoute les signes

Décimales	Signe-magnitude
-3	111
-2	110
-1	101
-0	100
0	000
1	001
2	010
3	011

Exemple (seul)

- Pour des nombres “signe-magnitude”, quelles sont les valeurs décimales de
  - 2 bits: $(11)_2$
  - 3 bits: $(011)_2$
  - 3 bits: $(111)_2$

Exemple (seul)

- Si c’est en signe-magnitude, on aurait:
  - $(11)_2$: $(-1)_{10}$
  - $(011)_2$: $(3)_{10}$
  - $(111)_2$: $(-3)_{10}$

Complément à 1

- Il existe aussi d'autres façons d'exprimer les nombres négatifs
- Un approche s'appelle le complément à 1
  - De façon plus générale, on l'appelle le complément a (N-1) (N est la base)
  - En base 2, c'est le complement à 1
- La technique c'est de prendre le plus gros chiffre et de soustraire notre amplitude

Complément à 1

- Par exemple, avec 4 bits, le plus gros chiffre c'est $(1111)_2$
- Si je voulais représenter $(-6)_{10}$ en complement à 1:
  - Mon amplitude est 0110
  - Je prends 1111 et je soustrais ce 0110: 1001
  - Donc, en complement à 1, -6 est égal à $(1001)_2$
  - +6 est représente bien normalement: 0110

Exemple

- Représentez 4 et -4 en complément à 1 avec 4 bits

Exemple

- On commence par prendre l'amplitude qui est de 4: $(0100)_2$
- Si j'étais positif, je m'exprime encore de la même manière: $(4)_{10}=(0100)_2$
- Pour -4, je dois continuer à manipuler:
  - Le plus gros chiffre c'est $(1111)_2$
  - $1111-0100=(1011)_2$

Note: on obtiendrait le même résultat en inversant les bits: $0 \leftrightarrow 1$
$(4)_{10}=(0100)_2 \leftrightarrow (-4)_{10}=(1011)_2$

Exemple (seul)

- Trouvez le complement à 1 de 6 et -6 avec 5 bits

Exemple (seul)

- On commence avec l’amplitude de 6 et de -6 avec 5 bits: 00110
  - Le nombre le plus eleve avec 5 bits: 11111
  - On soustrait: 11111-00110=11001
- On utilisant le truc, on inverse les bits:
  - 6: 00110
  - -6: 11001

Redondance +0 et -0

- Avec notre truc, la conversion en complement à 1 devient plus simple:
  - Pour -9, on trouve 9: 01001 et on inverse: 10110
  - Pour -5, on trouve 5: 00101 et on inverse: 11010
- Pour 0, on trouve 0: 00000 et on inverse 11111
  - Encore ici, on se retrouve avec des cas redondants... Il y a 2 façons d’écrire 0!

Complement à (N-1)

- Le complément à 1 c’est pour les nombres binaires
- De façon générale, c’est le complément à N-1
    - Ici notre base était 2, donc N-1 est 1...
- Ça se fait aussi avec d’autres bases
    - En décimale, base 10, c’est le complément à 9
    - Allons voir un exemple...

Exemple

- Trouvez le complément à 9 de -32 en utilisant 4 chiffres:
  - On commence par prendre l'amplitude de -32: 32
  - Le chiffre le plus gros avec 4 chiffres est 9999
  - -32 nous donnerait 9999-0032=9967
  - Le complément à 9 de -32 est donc 9967...

Exemple (seul)

- En utilisant la notation complément à 1
  - Remplissez le tableau suivant en utilisant des nombres binaires de 3 bits

Décimales	**Complément à 1**
-3	
-2	
-1	
-0	
0	
1	
2	
3	

Exemple (seul)

- On commence par mettre les amplitudes
- Par la suite, on inverse (au besoin)

Décimales	Complément à 1
-3	011 → 100
-2	010 → 101
-1	001 → 110
-0	000 → 111
0	000 → 000
1	001 → 001
2	010 → 010
3	011 → 011

Complément à 2

- Le problème avec le complément à 1 et signe-magnitude:
  - Il existe +0 et -0 qui devraient être égaux
  - On “gaspille” 2 valeurs pour représenter 0
- Pour résoudre le problème, on présente le complément à 2
- Pour parler du complément à 2 on commence par le complément à 1...

Complément à 2

- On sait que 0000 représente +0 et 1111 represente -0
  1) Si j'ajoutait 1 au complément à 1, j'obtiendrais 10000
  2) Si je considère que 4 bits, j'obtiendrais 0000
- On appelle ça complément a 2…
  1) Faire le complément à 1
  2) Ajouter 1
  3) Négliger l'élément de gauche si ça dépasse le nombre de bits

Exemple

- Trouvez le complément à 2 de -5 avec 4 bits

  - On commence par convertir 5 en binaire: 0101
  - Par la suite, on inverse: 1010 (complément à 1)
  - Pour avoir le complément à 2, on ajoute 1:

  - Le résultat devient 1011

Complement à 2

- Il y a 2 avantages à utiliser le complément à 2:
  - +0 et -0 deviennent le même
  - Les manipulations se font sans complications (on va le voir bientôt)

Complément à N

- Complément à 2 c’est pour les nombres binaires
  - Il existe l’équivalent pour les autres bases aussi...
  - C’est le complément a N, ou N est la base
- En décimale, c’est le complément à 10
  - Ça se fait en trouvant le complément à 9 et en ajoutant 1

Exemple

- Trouvez la valeur de -4 en complément à 10 avec 4 chiffres:
    - L’amplitude de -4 c’est 4
    - Le plus gros chiffre en 4 bits c’est 9999
    - 9999-4=9995 (complément à 9)
    - Le complément à 10 sera donc 9996

Exemple

- Calculez 067-033 en utilisant le complément à 10:
  - On commence par calculer le complément à 9: 999-033=966
  - On additionne 1: 967 (complément à 10)
  - On additionne maintenant les 2 chiffres: 067+967
  - Le résultat est 1034, mais on ne garde que 3 chiffres: 34
  - Le fait d'avoir un nombre moins que 5 à gauche dit que c'est positif (034)

Exemple (seul)

- Calculez 067-098 en utilisant le complément à 10

Exemple (seul)

- On commence par calculer le complément à 9: 999-098=901
  - On additionne 1: 902 (complément à 10)
  - On additionne maintenant les 2 chiffres: 067+902
  - Le resultat est 969… il n'y a que 3 chiffres
  - Le fait de ne PAS avoir de retenue à gauche indique que c'est NÉGATIF…
  - On peut retrouver la "vraie" valeur en faisant la meme opération: 999-969+1=31… donc -31

Resume

- Des nombres peuvent être:
  - Non-signés: de 0 jusqu'au maximum
  - Signe-Magnitude: 1 bit de signe et le reste représente l'amplitude
  - Complément à 1: soustraire l'amplitude du nombre négatif de la valeur maximum
  - Complément à 2: Faire le complément à 1 et ajouter 1 à la solution

Signe

- Le complément à 2 est le plus efficace:
  - Il va de -8 a +7.. parce qu'il a un seul 0