Numération et arithmétique

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

ReDS
Reconfigurable & embedded Digital Systems

Etienne Messerli
Institut REDS, HEIG-VD

Le 21 février 2013

Contenu de la présentation

- Systèmes de numération, représentation en binaire
- Représentation des nombres négatifs
- Addition, soustraction, dépassement (C, Ovr)
- Multiplication de nombre entier (signé et non-signé)
  - ✓décomposition spatiale et temporelle
- Division de nombre entier *(ne sera pas traité)*
- Représentation des nombres flottants

Contenu : Systèmes de numération

- Introduction : systèmes de numération
- Bases 2, 10 et 16
- Décimal codé en binaire
- Changements de base
  - ✓De base 10 en base 2 à la main
  - ✓De base 2 en base 10 à la main
  - ✓De base 10 en base 2 à la machine
  - ✓De base 2 en base 10 à la machine

Systèmes de numération : définitions…

- Système de numération = langage composé de
  - ✓Une liste ordonnée de symboles (chiffres)
  - ✓Des règles pour créer des nombres avec ces symboles
  - ✓Des règles définissant un jeu d'opérations = arithmétique (addition, soustraction, multiplication …)
- Base = nombre de symboles différents
- Nombre = juxtaposition de chiffres

…systèmes de numération : définitions…

- Système de numération de position : chaque chiffre dans un nombre a un poids dépendant de sa position
- Pondération usuelle : puissances de la base
- Exemple : base 10
  - ✓ 1er chiffre à gauche de la virgule = unités (10^0)
  - ✓ 2ème chiffre à gauche de la virgule = dizaines (10^1)
  - ✓ 1er chiffre à droite de la virgule = dixièmes (10^{-1})
  - ✓ 2ème chiffre à gauche de la virgule = centièmes (10^{-2})
- Contre-exemples : système romain, code de Gray

…systèmes de numération : définitions…

- Dans une numération de position avec des puissances de la base,

  un nombre = une expression polynomiale
- Exemple en base 10 :
  - ✓ $1993 = 1 \cdot 10^3 + 9 \cdot 10^2 + 9 \cdot 10^1 + 3 \cdot 10^0$
- Exemple en base 2 :
  - ✓ $1001_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 9$

…systèmes de numération : définitions

- Convention : base explicitée si autre que dix ou si risque d'ambiguïté
- Exemples de notation :
  - ✓En base 2 : 1001_2, ou 0b1001, ou B"1001" (style VHDL)
  - ✓En base 16 (hexadécimal) : $A2E_{16}$, ou **0xA2E**, ou X"A2E" (style VHDL)
- Vocabulaire : un chiffre binaire est appelé *bit* (contraction de *binary digit*, signifie aussi *petit morceau*)

Bases 2, 10 et 16

- Bases utilisées dans les systèmes de traitement de l'information :
  - ✓Base 2 (binaire) : toute l'électronique numérique travaille en base 2 exclusivement
  - ✓Base 10 (décimal) : la plupart des humains (nos clients) comptent en base 10
  - ✓Base 16 (hexadécimal) : forme condensée du binaire, pour la communication entre humains ; utilise la suite ordonnée de symboles 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Décimal codé en binaire…

- Il est possible de réaliser des circuits électroniques faisant des calculs sur des nombres décimaux
- Astuce : chaque chiffre décimal est codé en binaire (BCD, binary coded decimal)
- En réalité le fonctionnement électronique reste binaire
- Les changements de base sont ainsi évités, mais les circuits sont plus complexes

…décimal codé en binaire…

- Exemple : 1994_{10} s'écrit 0001 1001 1001 0100 en BCD
- Addition 9 + 4 : 1001 + 0100 = 1 0011 en BCD, mais 1101 en binaire
- L'addition de 2 chiffres BCD peut être faite en binaire, mais doit être suivie d'une correction (+6) si le résultat est $\geq 10_{10}$, ceci pour chaque chiffre BCD!
- Les valeurs 1010, 1011, 1100, 1101, 1110 et 1111 ne sont pas utilisées (n'ont pas de sens)

…décimal codé en binaire

- Pour chaque chiffre décimal il faut 4 bits
- Pour un entier entre 0 et 9999_{10} il faut
  - ✓16 bits (4 x 4) en BCD
  - ✓mais seulement 14 bits en binaire pur
- En résumé :
  - ✓opérations sont plus complexes en BCD qu'en binaire pur
  - ✓stockage de données en BCD prend plus de place qu'en binaire pur
  - ✓en pratique pas de calcul en BCD, conversion en binaire

Changements de base

- Puisque le calcul en binaire est plus simple que le calcul en BCD, il faudra généralement passer par des changements de base
- De base 10 en base 2 pour les données introduites dans le système
- De base 2 en base 10 pour les résultats fournis par le système
- Méthodes différentes pour conversion «à la main» et conversion «à la machine»

D'une base à une autre, à la main…

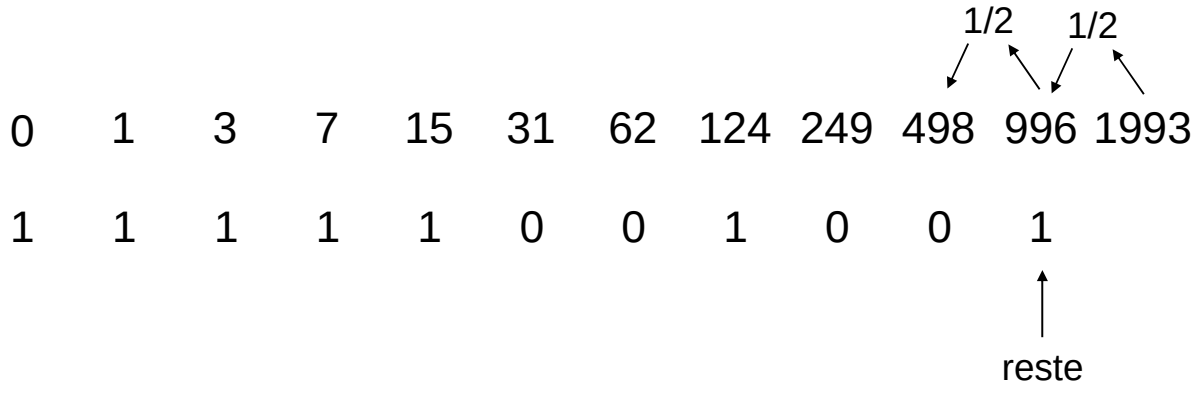
- Partie entière E (à m+1 chiffres) d'un nombre, sous forme polynomiale, en base b :
  - ✓$E_b = c_m \cdot b^m + c_{m-1} \cdot b^{m-1} + \ldots + c_1 \cdot b^1 + c_0 \cdot b^0$
  - ✓$E_b = (c_m \cdot b^{m-1} + c_{m-1} \cdot b^{m-2} + \ldots + c_1) \cdot b^1 + c_0$
- Le reste de la division E_b / b est c_0
- Les restes des divisions successives par la base b souhaitée donnent les chiffres du nombre en base b, à partir de la virgule

…d'une base à une autre, à la main…

- Partie fractionnaire F (à k chiffres) d'un nombre, sous forme polynomiale, en base b :
  - ✓$F_b = c_{-1} \cdot b^{-1} + c_{-2} \cdot b^{-2} + \ldots + c_{-k} \cdot b^{-k}$
  - ✓$F_b = (c_{-1} \cdot b^0 + c_{-2} \cdot b^{-1} + \ldots + c_{-k} \cdot b^{-k+1}) \cdot b^{-1}$
- La partie entière de la multiplication $F_b \cdot b$ est c_{-1}
- Les parties entières des multiplications successives par la base b souhaitée donnent les chiffres du nombre en base b, à partir de la virgule

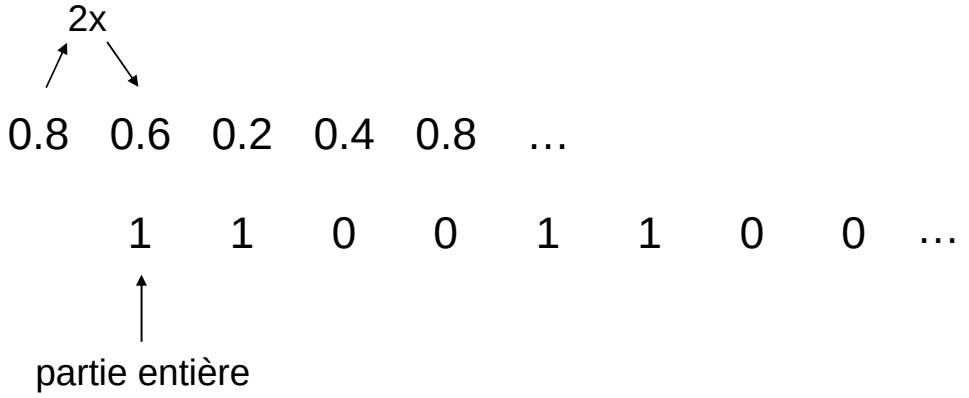
De base 10 en base 2 à la main…

- Les calculs seront faits en décimal
- Exemple : $1993.8_{10} = ?_2$
  - ✓Partie entière



…de base 10 en base 2 à la main

- ✓Partie fractionnaire



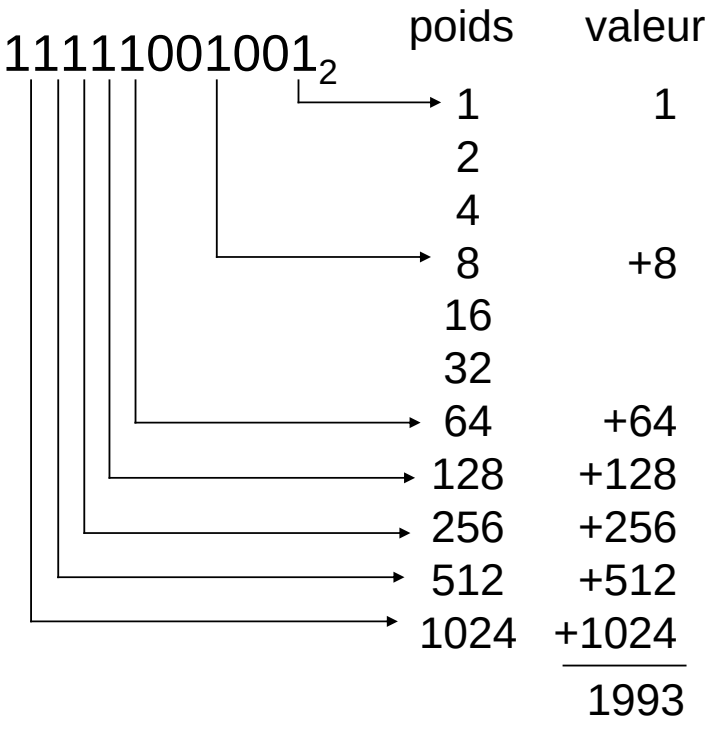
D'où :

$$1993.8_{10} = 1111100100,\ 1100\overline{1100}_2$$

De base 2 en base 10 à la main…

- A la main: simple application de la forme polynomiale avec calcule réalisé en base 10
  - ✓calculer en décimal la valeur du polynôme
  - ✓utilisation d'une table des puissances de 2 (les poids des chiffres dans un nombre binaire)
- Possible car nous calculons dans la base que nous avons apprise à l'école!

…de base 2 en base 10 à la main…



Conversion à la machine

- Calcul doit être fait en binaire!
- Représentation des nombres décimaux (base 10)
  $\Rightarrow$ utilisation du BCD
- Préférable de réaliser les conversion sous forme de calcul répétitif (boucle séquentielle ou structure modulaire)

De base 10 en base 2 à la machine…

- En partant de la forme polynomiale d'un entier et en mettant la base b en évidence toutes les fois que l'on peut, on obtient :
  - ✓ $E_b = (\ldots((c_m \cdot b) + c_{m-1}) \cdot b) + \ldots) \cdot b + c_1) \cdot b + c_0$
  - ✓ suite de multiplications et additions
- Exemple : 1993_{10} = 0001 1001 1001 0011 en BCD se traduit en binaire par le calcul ((0001•1010+1001)•1010+1001)•1010+0011

…de base 10 en base 2 à la machine

- Algorithme en pseudo Ada :

```
Result := 0;
for i in m downto 0 loop
  Result := Result * 1010₂ + Digit_BCD[i];
  -- le calcul est effectue en binaire
end loop;
```


De base 2 en base 10 à la machine

- Partie entière E (à m+1 chiffres) d'un nombre, sous forme polynomiale, en base 10 (soit 1010_2) :
  - ✓ $E_b = c_m \cdot (1010)^m + c_{m-1} \cdot (1010)^{m-1} + \ldots + c_1 \cdot (1010)^1 + c_0 \cdot (1010)^0$
- Méthode générale :
  - ✓ partie entière : restes des divisions par dix
  - ✓ pour la partie fractionnaire : parties entières résultant des multiplications par dix

Exercices…

1. Quelles multiplications sont les plus faciles à faire en décimal? Et en binaire?
2. Peut-on multiplier par dix en binaire à l'aide d'une simple addition?
3. Ecrivez un algorithme pseudo Ada pour convertir de BCD en binaire la partie fractionnaire d'un nombre

…exercices

4. Pourquoi ne fait-on pas la conversion à la machine de base 2 en base 10 selon une méthode similaire à celle utilisée, à la machine également, pour le passage de base 10 en base 2 ?
5. Etablissez un algorithme pour passer de l'hexadécimal au BCD, à la machine.

Contenu : représentation nombres négatifs

- Notations utilisées pour les nombres négatifs :
  - ✓Complément à 2^n et dépassement de capacité
  - ✓Signe-amplitude
  - ✓Excédent de $2^{n-1} – 1$
  - ✓Complément à 1

Nombres négatifs : complément à 2^n…

- Les instructions des ordinateurs traitent des nombres de taille fixe : 4, 8, 16, 32 ou 64 bits
- Avec un nombre composé de n bits, on ne peut représenter que 2^n valeurs entières différentes
- On souhaite disposer de valeurs positives et de valeurs négatives
- On souhaite pouvoir réaliser les 4 opérations arithmétiques (add, sub, mul, div) de la façon la plus simple possible

…nombres négatifs : complément à 2^n…

- Complément à 2^n => représentation naturelle

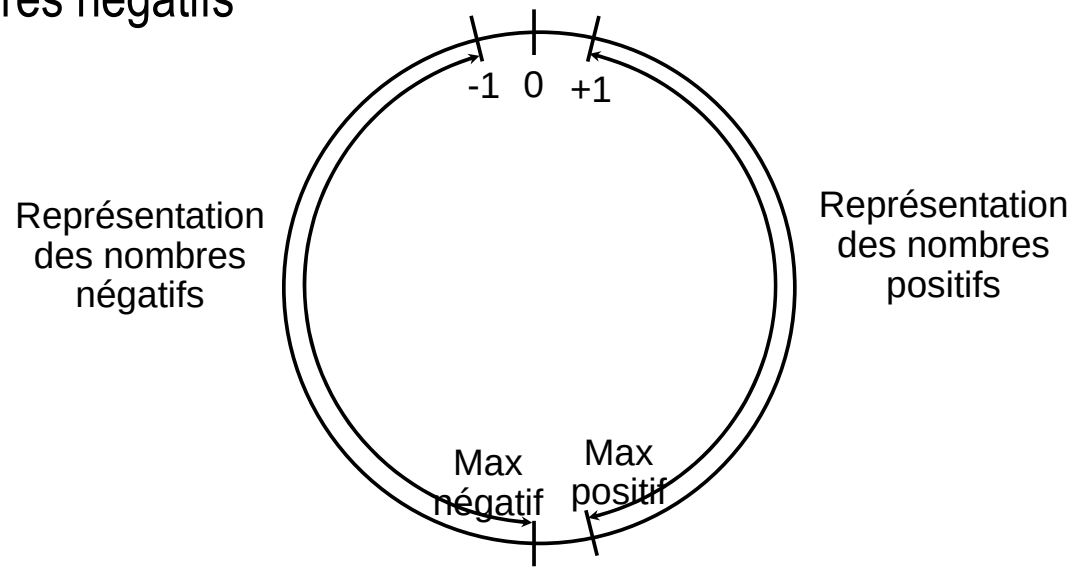
  3 - 4 = -1, soit 0011 - 0100 = 1111 !
- Un compteur-décompteur n bits en binaire pur
  - ✓compte en boucle : 0, 1, … 2^n – 1, 0, 1, …
  - ✓en effet : de 2^n – 1, il passe à 2^n , ce qu'il fait en mettant à 0 ses n bits et en générant un report sur le bit de poids directement supérieur (le bit de poids 2^n , qui n'est pas dans ce compteur n bits !)
  - ✓décompte en boucle : 2^n – 1, 2^n – 2,…1, 0, puis 2^n – 1 de nouveau (à 0 il génère un emprunt)

…nombres négatifs : complément à 2^n…

- Sur un compteur n bits, la valeur –1 est naturellement représentée par 2^n – 1, qui est la valeur obtenue en décomptant 1 fois depuis 0
- Avec cette représentation, en additionnant –1 et +1 on obtient 2^n :
  - ✓-1 est le complément à 2^n de +1
  - ✓-2 est le complément à 2^n de +2, etc
- D'où : «représentation en complément à 2^n»
- Autre terminologie souvent utilisée : ce nombre est (écrit) «en (notation) complément à 2»

…nombres négatifs : complément à 2^n…

- On utilise les valeurs de 1 à $2^{n-1}-1$ pour les nombres positifs, 0 pour le nombre nul, et 2^n-1 à 2^{n-1} pour les nombres négatifs



Exercices

1. Dans une représentation sur 4 bits :
   - ✓ Combien a-t-il de combinaisons possible ?
   - ✓ Combien a-t-il de nombre positif ?
   - ✓ Combien a-t-il de nombre négatif ?
   - ✓ Est-ce que toutes les combinaisons sont-elles utilisées ?
2. Répondre aux mêmes questions pour une représentation sur 8 bits.

…nombres négatifs : complément à 2^n…

- A=11110000 est-il plus grand ou plus petit que B=01110000 ?
- Si A et B sont des nombres sans signe, A>B
- Si A et B sont des nombres signés dans la représentation «en complément à 2^n», B>A (car B est positif et A est négatif)

…nombres négatifs : complément à 2^n…

- La représentation en complément à 2^n d'un nombre négatif –A, s'obtient en calculant le complément à 2^n de +A
- Le complément à 2^n d'un nombre s'obtient en inversant chacun de ses n bits, puis en ajoutant 1 au résultat
- Inverser un bit : mettre 0 à la place d'un 1 et 1 à la place d'un 0

…nombres négatifs : complément à 2^n

- Avantages de la notation «complément à 2»
  - ✓cohérence avec le fonctionnement d'un compteur
  - ✓représentation "naturelle" des nombres négatifs
  - ✓on peut faire une soustraction en utilisant un circuit d'addition et un circuit calculant le complément à 2
  - ✓une seule représentation de la valeur zéro

Nombres négatifs : autres notations…

- Signe-amplitude : le bit de gauche est utilisé pour le signe (0 pour positif, 1 pour négatif), les autres pour la valeur absolue
  - ✓utilisée pour la mantisse des nombres en virgule flottante (notation scientifique)
- Excédent de $2^{n-1} – 1$: on l'obtient en ajoutant une constante (biais = $2^{n-1} – 1$) à nos valeurs
  - ✓utilisée pour l'exposant des nombres en virgule flottante (notation scientifique)

…nombres négatifs : autres notations …

- Complément à 1 : en fait, c'est le complément à $2^n - 1$
  - ✓avantage : facile à calculer (inverser tous les bits)
  - ✓inconvénient : 2 représentations du zéro
  - ✓pas utilisé actuellement
- Formule pour calculer le complément à 1 :
  - ✓$C_1(A) = 2^n - 1 - A = \text{not}(A)$ (inversion bit à bit)

…nombres négatifs : relation entre C_1 et C_2

- Complément à 1 :
  - ✓$C_1(A) = 2^n - 1 - A$
- Complément à 2 :
  - ✓$C_2(A) = 2^n - A = 2^n - 1 + 1 - A = C_1(A) + 1$
    $= \text{not}(A) + 1$

Exercices …

1. En binaire, sur 8 bits, écrivez les nombres suivants
   - ✓ sans signe : $+128_{10}$
   - ✓ en notation «complément à 2» : -128_{10}
   - ✓ en notation «complément à 2» : $+128_{10}$
   - ✓ le complément à 2 de : $+128_{10}$
   - ✓ en notation «signe-amplitude» : -127_{10}
   - ✓ en notation «signe-amplitude» : $+128_{10}$
   - ✓ en notation «excédent de 127» : $+128_{10}$
   - ✓ en notation «excédent de 127» : -128_{10}

… exercices

2. Comment se justifie la recette de cuisine pour calculer le complément à 2 d'un nombre «inverser tous les bits puis ajouter 1» ? En examinant les chiffres 1 à 1 depuis la droite, trouvez une autre recette.
3. Extension de nombres signés : comment étendre sur 2n bits un nombre de n bits, signé, en notation «complément à 2»?

Etendre un nombre en complément à 2

- Pour étendre, par exemple de 8 à 16 bits, un nombre sans signe, il suffit de le compléter avec des 0 sur sa gauche
  - ✓Exemple : le nombre sans signe 1001 1100 étendu sur 16 bits devient 0000 0000 1001 1100
- Qu'en est-il pour un nombre signé, en notation «complément à 2»?
- Si le nombre est positif, on le complète avec des 0, comme un nombre sans signe

Etendre un nombre en complément à 2

- Si le nombre est négatif, en passant de 8 à 16 bits on passe de «complément à 2^8» à «complément à 2^{16}» !
- Il faudra lui ajouter $2^{16} - 2^8 = 1111111100000000$
- Pour étendre de k bits un nombre signé, en notation «complément à 2», il faut le compléter sur sa gauche avec k copies du bit de signe
  - ✓Exemple :
    le nombre signé 1001 1100 étendu sur 16 bits devient .....

Contenu : addition/ soustraction et dépassement

- Addition et soustraction
  - ✓nombre entiers non signés et signés
- Dépassement de capacité
  - ✓carry, borrow et overflow

Addition…

- Pour effectuer une addition en binaire, on peut procéder comme en décimal : chiffre après chiffre
- Ainsi, l'addition de 2 nombres de n bits est une opération décomposable en n additions de 3 nombres de 1 bit
- Pourquoi 3 ?
- Commençons par le bit de poids faible (2^0)

…addition…

- L'addition chiffre à chiffre est plus simple en binaire qu'en décimal :
  - ✓0+0 = 0
  - ✓0+1 = 1+0 = 1
  - ✓1+1 = 1 0
    - → bit de même rang du résultat
    - → reporté sur le rang suivant
- Pour le bit de rang 1, il faut tenir compte, en plus des deux bits, du report en provenance du rang 0

…addition : cas général …

- A(i) = chiffre de rang i du nombre A
- B(i) = chiffre de rang i du nombre B
- C(i) = report sur le rang i (vient du rang i-1)
- S(i) = résultat de rang (i)

  = 1 lorsque { (seul A(i)=1) ou (seul B(i)=1) ou (seul C(i)=1) ou [(A(i) et B(i) et C(i))=1] }

  sinon 0
- C(i+1) = report sur le rang i+1 (généré par le rang i)

  = 1 lorsque { [(A(i) et B(i)) ou (A(i) et C(i)) ou (B(i) et C(i))] =1 } sinon 0

…addition : cas général …

- Table de vérité additionneur 1 bit :

C(i)	B(i)	A(i)	C(i+1)	S(i)
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

- Equations logiques

C(i+1) = C(i) . B(i) # C(i) . A(i) #
B(i) . A(i)

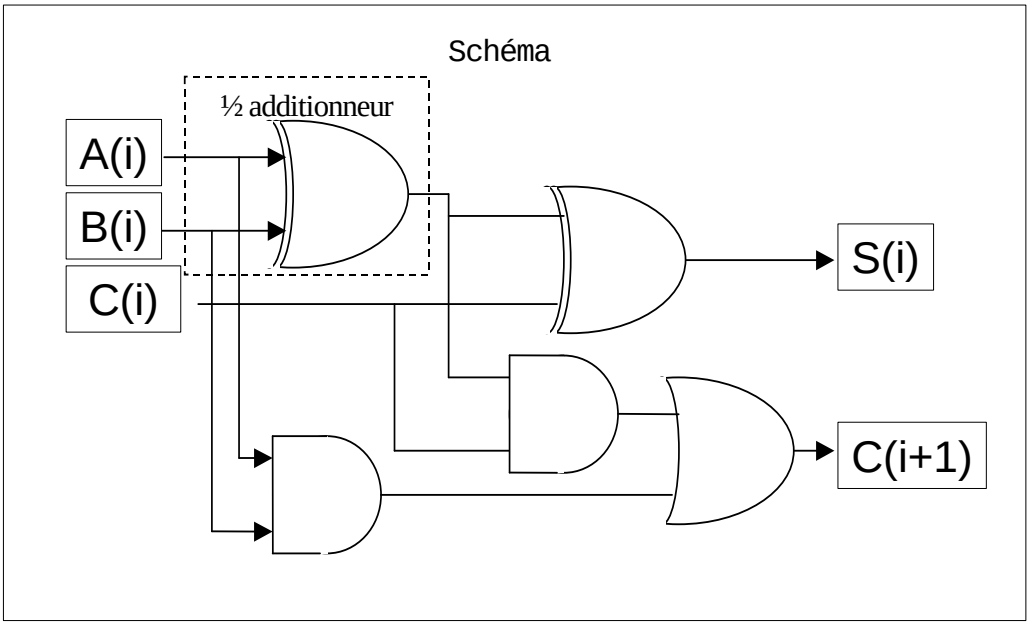
ou

C(i+1) = C(i) . (B(i) xor A(i)) #
B(i) . A(i)

S(i) = C(i) xor B(i) xor A(i)

…addition : cas général

- Schéma (1ère porte ou-exclusif appelée ½ additionneur)



Additionneurs

- Décomposition additionneur:
  Voir présentation Yann Thoma
  - ✓fichier "YTA_additionneurs.pdf"

Exercices

- Dessinez un schéma bloc d'un additionneur 4 bits en utilisant des additionneurs 1 bit

Soustraction…

- On peut faire la même démarche que pour l'addition
- Autre approche : au lieu de soustraire B de A, on peut calculer A plus le complément à 2 de B, sur n bits (rangs 0 à n-1)

$$A + (2^n - B) = 2^n + (A - B)$$

- Le bit de poids 2^n se trouve au rang n, donc en dehors de notre représentation
- Remarque :
  - ✓Dépassement de capacité n'est pas indiqué par le Carry

…soustraction

- Pour additionner et soustraire, au lieu d'un additionneur et d'un soustracteur, il suffit d'un additionneur et d'un inverseur bit à bit commandable
- Pour additionner et soustraire:
  - ✓Addition

$$A + B$$

  - ✓Soustraction: nous utiliserons le C2, soit:

$$A - B = A + C2(B) = A + C1(B) + 1$$
$$= A + \text{not } B + 1$$

Opérations et dépassement

Détection des cas de dépassement:

- Représentation des nombres non signés:
  - ✓Addition => dépassement indiqué par "Carry"
  - ✓Soustraction => dépassement indiqué par "Borrow"

    Borrow = Emprunt = not Carry
- Représentation des nombres signés en C2:
  - ✓Addition => dépassement indiqué par "Overflow"
  - ✓Soustraction => dépassement indiqué par "Overflow"

    Overflow = Cn xor Cn-1

Dépassement de capacité: résumé

- Carry
  - ✓ dépassement de capacité pour les additions de nombres non signés
  - ✓ généralement abrégé: C
- Borrow
  - ✓ dépassement de capacité pour les additions de nombres non signés
  - ✓ Borrow = not Carry
- Overflow
  - ✓ Dépassement de capacité pour les additions et soustraction de nombres signés
  - ✓ généralement abrégé: V

http://en.wikipedia.org/wiki/Carry_flag
http://en.wikipedia.org/wiki/Overflow_flag

Exercices …

1. Dans la soustraction par addition du complément à 2, que représente le bit de rang n-1 (MSB) du résultat ? (suggestion : faites plusieurs essais de soustraction avec diverses valeurs, sur 4 bits)
2. Décrivez le fonctionnement d'un «inverseur commandable» et dessinez son schéma logique
3. Comment faire le +1 requis par le calcul du complément à 2 utilisé dans une soustraction?
   - Réaliser un soustracteur 4 bits à l'aide d'un additionneur 4 bits (utiliser le symbole du ADD4 exe 6), sans gestion du dépassement de capacité

… exercices : dépassement capacité …

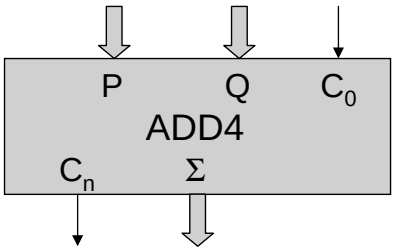
4. Les nombres 8 bits ci-dessous sont en notation «complément à 2». Faites les calculs indiqués et notez les reports
   - ✓ 10001101 + 10010010
   - ✓ 00000100 + 11111111
   - ✓ 11111111 + 00000001
   - ✓ 11111100 + 00000010
   - ✓ 00100010 + 01010001
   - ✓ 01111000 + 00001000
5. Certains résultats auraient besoin de plus de 8 bits : ils dépassent la capacité de la représentation. Comment peut-on détecter ce dépassement ?

… exercices : dépassement capacité …

6. Déterminer la valeur du résultat corrigé dans les cas suivants:
   - En représentation non signée
     - Dépassement lors d'une addition
     - Dépassement lors d'une soustraction
   - En représentation signée en C2
     - Dépassement lors d'une addition
     - Dépassement lors d'une soustraction

… exercices …

7. Vous disposez de modules d'addition de deux nombres de 4 bits

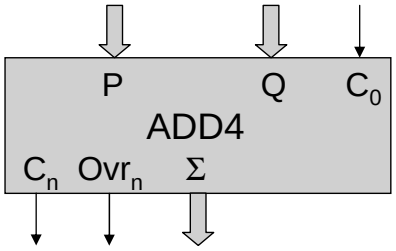


   Donnez le schéma pour additionner deux nombres de 8 bits

8. Proposez un schéma pour réaliser l'addition et la soustraction de deux nombres de 4 bits avec des additionneurs 1 bits et de la logique
   Les nombres sont en notation «complément à 2»

… exercices

9. Proposez un schéma pour réaliser l'addition et la soustraction de deux nombres de 8 bits en notation «complément à 2» avec des additionneurs 4 bits et de la logique
   Voici le symbole CEI pour les additionneurs 4 bits



Contenu de la présentation

- Multiplication d'entiers binaires sans signe
  - ✓Décomposition en opérations simples
  - ✓Algorithmes
  - ✓Décomposition combinatoire & séquentielle
  - ✓Réalisation combinatoire
  - ✓Réalisations séquentielles
- Multiplication d'entiers binaires signés (notation en complément à 2)

Multiplication à la main : exemple

```
   1 2 5
 x 1 3 9
 ---------
 1 1 2 5   ← 9x5  + 9x2x10 + 9x1x100
 3 7 5 0   ← 10x (3x5 +3x2x10 + 3x1x100)
1 2 5 0 0  ← 100x (1x5  +1x2x10 + 1x1x100)
 ---------
1 7 3 7 5
```

- La multiplication de 2 nombres est réalisée à l'aide d'une suite d'opérations plus simples

Multiplication en décimal : décomposition

- En décimal, la multiplication peut être décomposée en
  - ✓des multiplications Mcande x chiffre du Mcateur
    - →des multiplications chiffre x chiffre (génèrent un report)
    - →des décalages (multiplications par la base, tiennent compte du poids des chiffres du Mcande)
    - →des additions (avec reports)
  - ✓des décalages (multiplications par la base, tiennent compte du poids du chiffre du Mcateur)
  - ✓des additions (avec reports)

Multiplication en binaire…

- En binaire, la multiplication est **plus simple**. Elle peut être décomposée en
  - ✓des multiplications Mcande x chiffre du Mcateur
    - →des multiplications chiffre x chiffre (ne génèrent pas de report)
    - →des décalages (multiplications par la base, tiennent compte du poids des chiffres du Mcande)
  - ✓des décalages (multiplications par la base, tiennent compte du poids du chiffre du Mcateur)
  - ✓des additions (avec reports)

…multiplication en binaire…

- En binaire, la multiplication d'un chiffre par un chiffre se résume au livret suivant

  0 x 0 = 0

  0 x 1 = 0

  1 x 0 = 0

  1 x 1 = 1 il n'y a pas de report !

- Ce livret correspond à la fonction logique ET !

…multiplication en binaire…

```
      1 0 1 0
    x 1 0 0 1
    ---------
      1 0 1 0  ← 0x1 + 1x10x1 + 0x100x1 + 1x1000x1
    0 0 0 0 0  ← 10x (1010 x 0)
  0 0 0 0 0 0  ← 100x (1010 x 0)
1 0 1 0 0 0 0  ← 1000x (1010 x 1)
-------------
1 0 1 1 0 1 0
```

- La multiplication de 2 nombres est réalisée à l'aide d'une suite d'opérations plus simples

…multiplication en binaire

- Les décalages sont utilisés pour tenir compte du poids du bit concerné du multiplicateur
  - ✓par exemple : le résultat du produit du multiplicande par le bit de poids 2^3 du multiplicateur est décalé 3 rangs sur la gauche, ce qui revient bien à le multiplier par 2^3
- Les résultats du produit du multiplicande par chaque chiffre du multiplicateur (correctement décalés) sont additionnés

1er algorithme de multiplication

- La multiplication à la main peut être traduite en l'algorithme pseudo-Ada ci-dessous:

```
Resultat:= 0;
i:=0;
while i < Nb_Bits_Mcateur loop
  if Mcateur(i)=1 then --bit de rang i
    Resultat:= Resultat + Mcande;
  end if;
  Mcande:= Mcande*2;
  i:=i+1;
end loop;
```

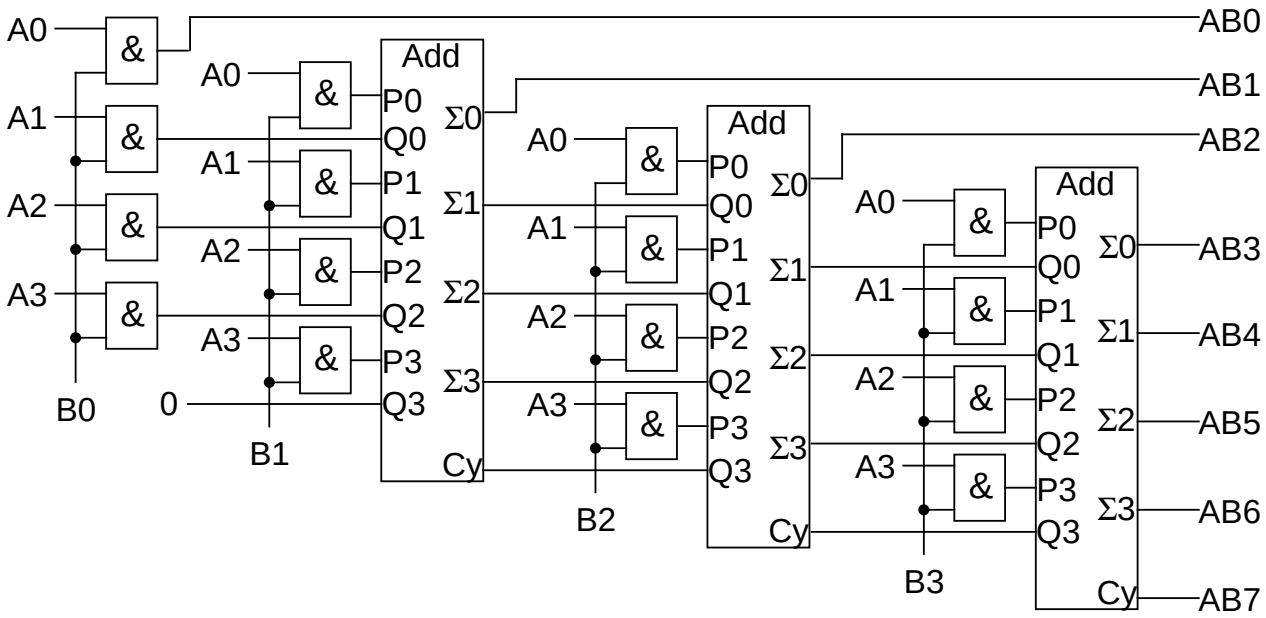

Décomposition combinatoire & séquentielle

- Toute fonction combinatoire peut-être décomposée dans l'espace,
  on parle d'une décomposition spatiale
  - ✓dans le cas de la multiplication, il s'agit de multiplicateur 1 bit (porte ET) et d'additionneur
- La décomposition peut aussi être dans le temps,
  on parle d'une décomposition séquentielle
  - ✓Il s'agit dès lors de minimiser le matériel
  - ✓Il faut mémoriser les résultats intermédiaires

Multiplication combinatoire (spatiale)

- L'algorithme ci-dessus peut être implémenté sous forme d'un circuit combinatoire
  - ✓Chaque opération requiert un exemplaire de l'opérateur concerné (ET, addition)
  - ✓Les décalages s'obtiennent par câblage
  - ✓La succession des opérations est obtenue par le passage des données à travers les divers opérateurs, dans l'ordre où ils sont câblés
- La boucle de l'algorithme est déroulée dans l'espace
  - ✓décomposition spatiale (combinatoire)

Exemple : multiplication 4 bits x 4 bits



Multiplication séquentielle…

- Il est également possible de n'utiliser
  - ✓qu'une seule rangée de fonctions ET pour calculer le produit du Mcande par un bit du Mcateur
  - ✓qu'un seul additionneur pour additionner les résultats partiels successifs
- mais il faut alors répéter 4 fois la séquence
  - ✓multiplier Mcande par 1 bit Mcateur et décaler
  - ✓additionner au résultat partiel précédent
  - ✓mémoriser le nouveau résultat partiel
  - ✓passer au bit suivant du Mcateur

…multiplication séquentielle…

- Dans le cas général de la multiplication de deux nombre de n bits donné par l'algorithme de la page 62, nous avons besoin de :
  - ✓ET 2n bits x 1 bit
  - ✓Additionneur 2n bits (résultat sur 2x n bits)
  - ✓Registre parallèle/série 2n bits pour le Mcande
  - ✓Registre parallèle/parallèle 2n bits pour le résultat
  - ✓Registre parallèle/série n bits pour le Mcateur (ou mux pour sélectionner les bits un à un)

…multiplication séquentielle…

- Optimisation :
  - ✓Lors de la décomposition spatiale de la page 65, nous pouvons constater que :
    - →Additionneur sur n bits
    - →Un bit du résultat à chaque pas
    - →Report de l'additionneur utilisé pour le prochain pas
  - ✓Nous pouvons aussi constater que :
    - →A chaque étape du calcul séquentiel, il y a un bit de plus pour le résultat et un de moins pour le multiplicateur
    - →Disposer d'un registre parallèle/série de 2n bits pour le résultat et le multiplicateur

…multiplication séquentielle…

- Finalement la multiplication de deux nombres de n bits nécessite :
  - ✓ET n bits x 1 bit
  - ✓Additionneur n bits
  - ✓Une bascule pour mémoriser le report
  - ✓Registre parallèle/série 2n bits pour le résultat/Mcateur (ou deux registres n bits liées pour le décalage)
  - ✓Registre parallèle/parallèle n bits pour le Mcande

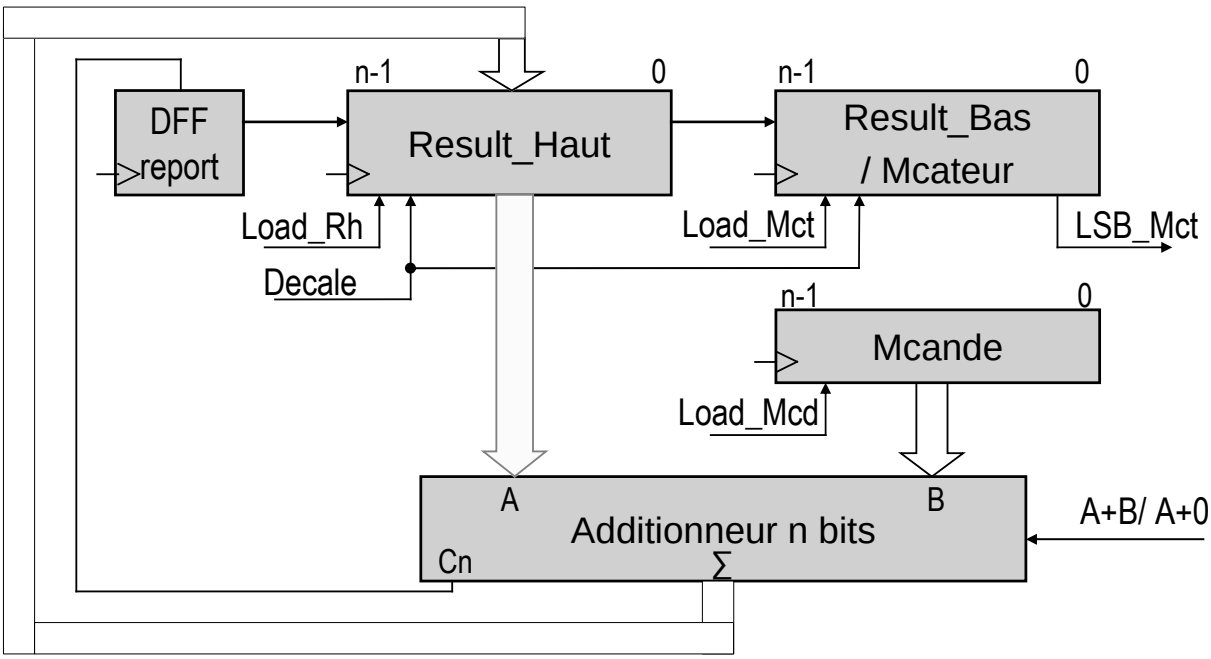
…multiplication séquentielle…

- 2ème algorithme pour la multiplication (optimisé):

```
Result_Haut:= 0;
for i in 1 to n loop
  if Bit_Poids_Faible(Mcateur) = 1 then
    Result_Haut:= Result_Haut + Mcande;
    --report est mémorisé dans un flip-flop
  else
    Result_Haut:= Result_Haut + 0;
  end if;
  Decale_a_Droite(Report, Result_Haut, Mcateur);
  --report precedent va dans MSB de Result_Haut
end loop;
```


…multiplication séquentielle…

- Schéma bloc de l'UT pour la multiplication séquentielle



Exemple avec "Numlab"

- Etudier la multiplication spatiale (sans signe)
  - ✓1001 x 0101; 1100 x 0110; 1101 x 1011
- Etudier la multiplication séquentielle (sans signe)
  - ✓1001 x 0101; 1100 x 0110; 1101 x 1011

Exercices

1. Nous souhaitons réaliser une système qui calcul le nombre total d'heures travaillées durant une semaine. Nous disposons du nombre d'heures travaillées durant le jour en cours (NHr) et du nombre de jours travaillés depuis le début de la semaine (NJr).
   - ✓ Vous devez concevoir et réaliser le système calculant le nombre total d'heures travaillées durant la semaine.
   - ✓ Vous utiliserez le principe de la décomposition spatiale pour la multiplication
   - ✓ Le nombre d'heures travaillées par jour est de 8.

Exercices

2. Nous souhaitons réaliser une système qui calcul le nombre total d'heures (TotHrs) travaillées durant le mois. Nous disposons du nombre de semaine travaillées (NbrSem) et du nombre de jour de la semaine en cours (NbrJr).
   - ✓ Vous devez concevoir et réaliser le système calculant le nombre total d'heures travaillées durant le mois.
   - ✓ Vous utiliserez le principe de la décomposition spatiale pour la multiplication
   - ✓ Le nombre d'heures travaillées par semaine est de 40, et le nombre d'heures travaillées par jour est de 8.

Exercices

3. Réaliser un circuit pour convertir en binaire un nombre BCD de 2 digits.
   - ✓ Vous utiliserez le principe de la décomposition spatiale pour la multiplication
4. autres exercices à prévoir.

Multiplication d'entier signé …

- Utilisation algorithme nombres non signés

Nombres non signés

```
  11
x 13
  33
 11_
 143
```

OK <=>

```
    1011
x   1101
    1011
 0 000
 10 11
 101 1 _
1000 1111
```

Nombres signés

```
1011 = -5
1101 = -3

   -5
x  -3
   15
```

Faux => -113

… multiplication d'entier signé …

- Nombre signé : notation en complément à 2
  - $A = C_2(A) = 2^n - A$
- Multiplication avec nombre signé
  - ✓utiliser l'algorithme pour nombre non signé et traiter le signe séparément
  - ✓adapter l'algorithme pour pouvoir l'utiliser avec des nombres signés

… multiplication d'entier signé …

- Multiplication avec nombre signé :

  cas B * A avec A < 0 d'où

  $B * A = B * C_2(|A|) = B * (2^n - |A|) = \underbrace{B * 2^n}_{\text{terme en trop!}} + B * (-|A|)$

  d'où :

  $B * (-|A|) = B * A - B * 2^n$ le second terme est la correction

- Calcul intermédiaire :
  - ✓nombres signés => extension du signe !

… multiplication d'entier signé …

- Correction de l'algorithme

```
          1011 = -5
        x 1101 = -3
      1111 1011
    + 0000 000
    + 1110 11
    + 1101 1  _
      1011 1111
    - 1011 0000
      0000 1111 = + 15
```

Résultats partiels signés : extension du signe

correction –(multiplicande * 2^n)

```
  -5
x -3
 15
```

< = >

OK

… multiplication d'entier signé …

- Optimisation de l'algorithme
  - ✓Il y a une correction seulement si multiplicateur négatif, donc le MSB est à '1'

    dans ce cas il y a l'opération

    $+ B * 2^{n-1}$

    qui est suivie par la correction

    $- B * 2^{n}$
  - ✓il est possible de grouper ces deux opérations, soit :

    $+ B * 2^{n-1} - B * 2^{n} = B * 2^{n-1} – 2 * B * 2^{n-1} = - B * 2^{n-1}$

… multiplication d'entier signé …

- Algorithme de Robertson

```
Result_Haut:= 0;
for i in 1 to n-1 loop   -- n = nombre de bits
  if Bit_Poids_Faible(Mcateur) = 1 then
    Result_Haut:= Result_Haut + Mcande;
  else
    Result_Haut:= Result_Haut + 0;
  end if;
  Decale_a_Droite(Ovr xor MSBit, Result_Haut, Mcateur);
end loop;
if Bit_Poids_Faible(Mcateur)= 1 then
  Result_Haut:= Result_Haut – Mcande; --cor. anticipée
else
  Result_Haut:= Result_Haut - 0;
end if;
Decale_a_Droite(Ovr xor MSBit, Result_Haut, Mcateur);
```

… multiplication d'entier signé …

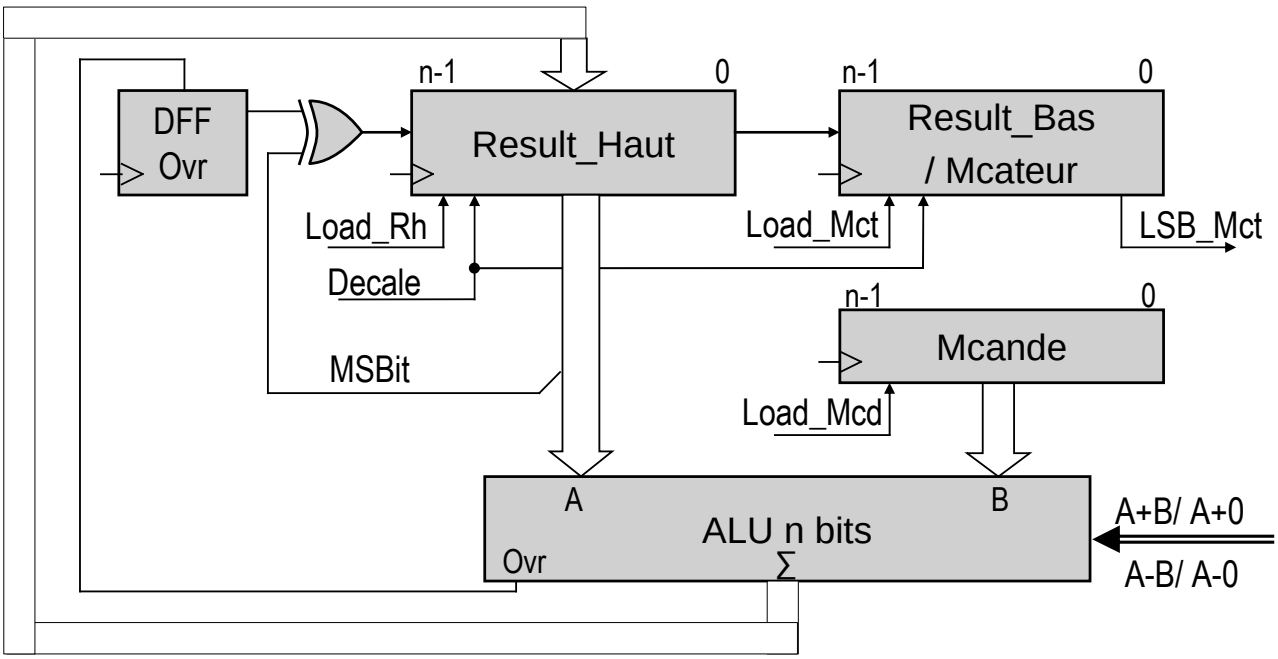
- Exemple avec algorithme de Robertson

```
        1011 = -5
      x 1101 = -3
    1111 1011
+ 0000 000
+ 1110 11
- 1101 1   _correction  anticipée –(multiplicande * 2^(n-1))
    0000 1111 = + 15
```

```
   -5
x  -3
   15
```


… multiplication d'entier signé …

- Schéma bloc de l'UT pour la multiplication d'entier signé



Exemple avec "Numlab"

- Etudier la multiplication séquentielle selon Robertson (nombres signés)
  - ✓ -5 x -3 (1011 x 1101); -5 x 3 (1011 x 0011); 5 x -3 (0101 x 1101)
  - ✓ -7 x -3 (1001 x 1101), -1 x -1 (1111 x 1111); 7 x 7 (0111 x 0111)

Dia laissé vide volontairement

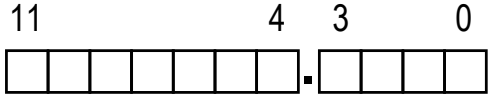
Contenu de la présentation

- Virgule fixe et virgule flottante
  Dynamique et résolution
- Formats standard en virgule flottante
- Opérations sur les nombres en virgule flottante
- Bits de gardes et arrondi

Virgule fixe

- Compatible avec les circuits vus précédemment
- Place fixe pour la virgule
- Dynamique dépend nombre de bits, si N bits :

  dynamique = 2^N-1
- Représentation de nombres grands et petits:
  - 80 = 1010000 nécessite 7 bits à gauche de la virgule
  - 0,0625 = 0.0001 nécessite 4 bits à droite de la virgule
  - Format commun : nécessite 12 bits



Virgule flottante

- Meilleure dynamique
  - ✓dépend de la taille de l'exposant
  - ✓par contre diminution de la précision
- Notation scientifique en décimal:
  - ✓signe, mantisse, exposant; p. ex. + 4.5 * 10^{-12}
- Notation scientifique en binaire: F = (S,E,M)
  - ✓ S: Signe = signe du nombre
  - ✓ M: Mantisse = chiffres significatifs
  - ✓ E: Exposant = facteur d'échelle, signé
    E est en notation biaisée

Virgule fixe: exemples

- 16 bits, virgule fixe:
  permet 65'536 combinaisons différentes
  - ✓0, 1,..., 65534, 65535;
  - ✓-32768,...,-1,0-1,...,32767
    valeur max : 32767 = 3.2767 * 10^4
  - ✓-16384,...,-0.5,0,0.5,...,16383.5
  - ✓...

Virgule flottante: exemples

- 16 bits, virgule flottante, avec
  - ✓signe (1 bit)
  - ✓mantisse sur 10 bits,
  - ✓exposant sur 5 bits:

    →$-1023 * 2^{16}$,...,$-1 * 2^{-15}$,0 , $1 * 2^{-15}$,..., $1023 * 2^{16}$

    valeur max : $1023 * 2^{16} = 67'043'328 = 6.7 * 10^7$

16 bits : toujours 65'536 combinaisons différentes !

Dynamique, résolution …

- Dynamique :
  - ✓rapport entre la plus grande valeur et la plus petite
- Résolution:
  - ✓le plus petit intervalle observable

    →NB: virgule fixe: résolution = constant
    erreur relative = f(valeur)

    → virgule flottante: résolution = f(valeur)
    erreur relative = constant si normalisé

Dynamique, résolution …

- Résolution:
  - ✓virgule fixe:

	Max	Min
→16 bits : max 65535, min 1	65235	1
→erreur absolue ½ LSB	0,5	0,5
→erreur relative en %	0,0015%	50%

  - ✓Virgule flottante

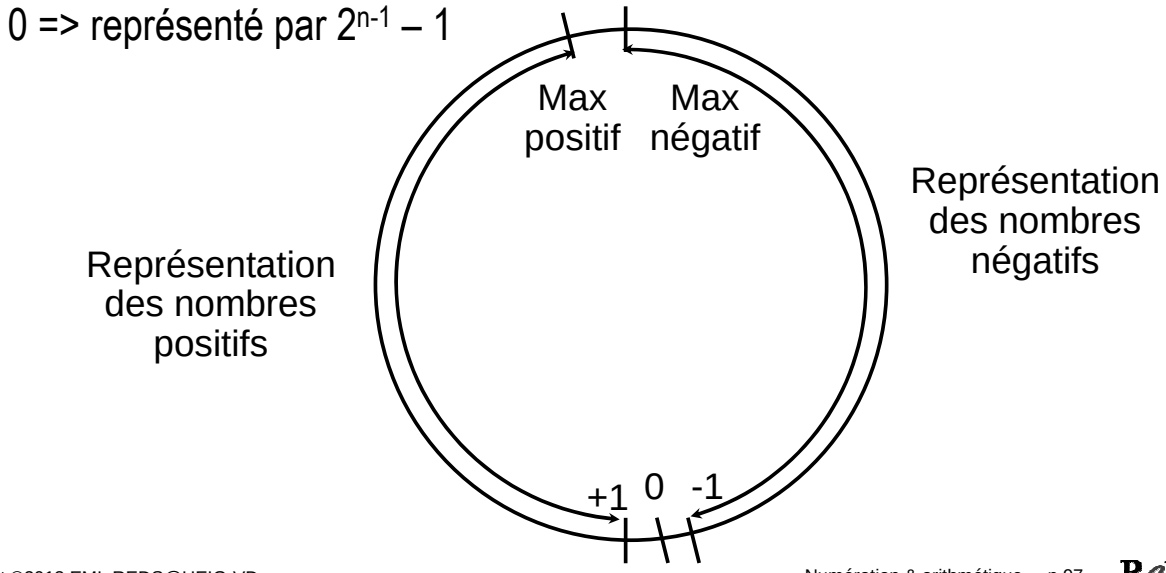
	Max	Min
→16 bits, dont 10 bits pour la mantisse (cas normalisé)	1.023	1.023
→erreur absolue ½ LSB	0.0005	0.0005
→erreur relative	0,05%	0,05%

Formats IEEE

- Simple précision:
  - ✓ 32 bits : exposant de 8 bits, mantisse de 23 bits
- Double précision
  - ✓ 64 bits : exposant de 11 bits, mantisse de 52 bits
- Quadruple précision
  - ✓ 128 bits : exposant de 15 bits, mantisse de 112 bits
- Voir site Wikipedia
  - ✓ http://fr.wikipedia.org/wiki/IEEE_754

Notation en excédent …

- Excédent de $2^{n-1} – 1$: on l'obtient en ajoutant une constante (biais = $2^{n-1} – 1$) à nos valeurs



… notation en excédent

- $N_{excédent} = N_{10} + 2^{n-1} – 1$
- nombre positif > nombre négatif
- calcul compliqué: 0 + 0 ≠ 0 !
- MSB à '1' indique un nombre positif
- Notation en excédent sur 8 bits : $N_{excédent} = N_{10} + 127$
  - ✓min = "0000 0000" => ……..
  - ✓zéro = "0111 1111" => 0
  - ✓max = "1111 1111" => ……..

Normalisez... Dénormalisez...

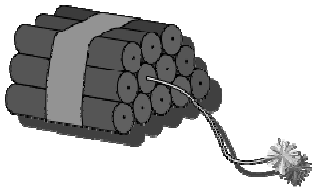
- Slogans de partis politiques?
- Non, ici c'est une façon d'accorder le format de la mantisse aux besoins du calculateur:

  ✓mantisse normalisée: partie entière = 1;

  →par exemple: 1.00110110

  ✓mantisse dénormalisée: partie entière = 0;

  →par exemple: 0.01001101



Exemple: normalisation du nombre 12

$1100 = 1.1*2^3$

Format standard ...

- Signe 1 bit '0' = positif +, '1' = négatif –

  la mantisse représente une valeur absolue
- Mantisse fractionnaire normalisée

  1._ _ _ _ _ _ _ _ _ _

  partie entière toujours à '1', souvent pas représentée, il alors nommé "hidden bit" (bit caché)
- Exposant en notation biaisée

  représentation signée

  exposant négatif < exposant positif

  facilite les comparaisons

… format standard: cas spéciaux …

- Représentation du zéro
  - ✓pas de représentation normalisée
  - ✓convention: exposant et mantisse sont nuls
    - →bits de l'exposant: 0, bits de la mantisse: 0, signe quelconque
- Représentation de l'infini
  - ✓Indication d'un dépassement de capacité
  - ✓convention: exposant tous les bits à '1', mantisse nulle
    - →bits de l'exposant 1, bits de la mantisse 0
    - →signe = 0 : + infini, signe = 1 : - infini,

… format standard: cas spéciaux

- Représentations de messages d'erreurs
  - ✓But: propager les erreurs des calculs
  - ✓Il ne s'agit pas de nombres!
  - ✓On les appelle des NaN (Not a Nnumber)
  - ✓Convention: exposant tous les bits à '1', mantisse /= nullle
    - →bits de l'exposant 1, signe quelconque
    - →bits de la mantisse : ne doivent pas être tous nul
      indique le numéro de l'erreur

Format standard, simple précision

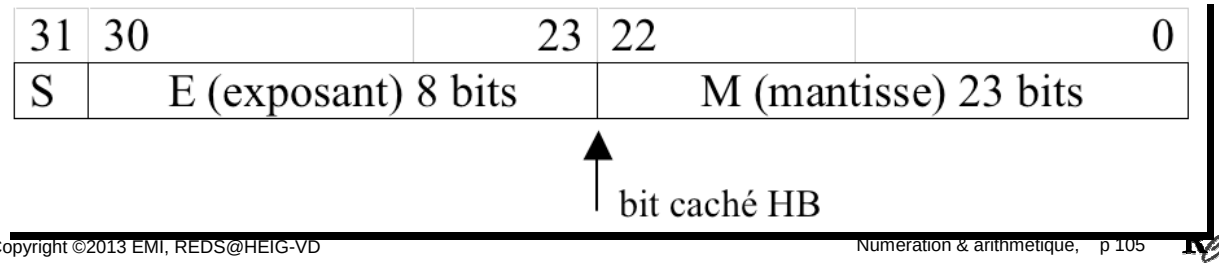
- Format sur 32 bits
- Signe 1 bit '0' = positif +, '1' = négatif –
- Mantisse normalisée sur 23 bits avec bit caché
  - ✓ la mantisse est donc sur 24 bits
  - ✓ exception : dénormalisé lorsque exposant est nul (E = 0)
- Exposant en notation biaisée sur 8 bits
  - ✓ excédent = 2^{n-1} -1, excédent pour 8 bits = 127
  - ✓ E = e + 127

Exercices : représentation simple précision

1. Donner la représentation de +∞
2. Donner la représentation de zéro
3. Donner la valeur maximum et minimum représentable en simple précision

Simple précision

forme de F	interprétation	
$-126 \leq e < 127$	nb. normalisé	$(-1)^S (1 + mf) 2^e$
$e = -127, mf \neq 0$	nb. dénormalisé	$(-1)^S (mf) 2^{-126}$
$e = -127, mf = 0$	zéros	$(-1)^S 0$
$e = +128, mf = 0$	infinis	$(-1)^S \infty$
$e = +128, mf \neq 0$	pas des nombres	NaN



Formats IEEE: récapitulatif

formats IEEE nb. de bits	simple précision 32	simple précision étendu 44	double précision 64	double précision étendu 80	quadruple précision 128
nb. bits E	8	11	11	15	15
notation E	excédent 127	excédent 1'023	excédent 1'023	excédent 16'383	excédent 16'383
val. min. max. e	-126, +127	-1'022, +1'023	-1'022, +1'023	-16'382, +16'383	-16'382, +16'383
nb. bits M	23	32	52	64	112
bit caché	oui	non	oui	non	Non
val. max. F	$(2 - 2^{-23}) \cdot 2^{127}$	$(2 - 2^{-31}) \cdot 2^{1'023}$	$(2 - 2^{-52}) \cdot 2^{1'023}$	$(2 - 2^{-63}) \cdot 2^{16'383}$	$(2 - 2^{-111}) \cdot 2^{16'383}$
val. min. $F \neq 0$	2^{-149}	$2^{-1'053}$	$2^{-1'074}$	$2^{-16'445}$	$2^{-16'493}$

Bits de garde et arrondi

- bits de garde utilisés **seulement** pendant le calcul
- **arrondi** nécessaire pour revenir au format standard
  - ✓ par troncature: arrondi vers zéro
  - ✓ vers +
  - ✓ vers –
  - ✓ au plus près

Opérations en virgule flottante (1)

Opération d'addition avec des nombres flottants

- Soit les nombres flottants A et B

$$A = m_A \cdot 2^{e_A} \quad \text{et} \quad B = m_B \cdot 2^{e_B}$$

- nous souhaitons calculer :

$$C = A+B = m_A \cdot 2^{e_A} + m_B \cdot 2^{e_B}$$

- que vaut: m_C et e_C ?

Opérations en virgule flottante (2)

Addition et soustraction:

- Les 2 nombres doivent avoir le même facteur d'échelle
  - ✓il faut donc aligner les exposants avant de réaliser un calcul sur les mantisses
  - ✓Au besoin, dénormaliser le plus petit

Pour les 4 opérations:

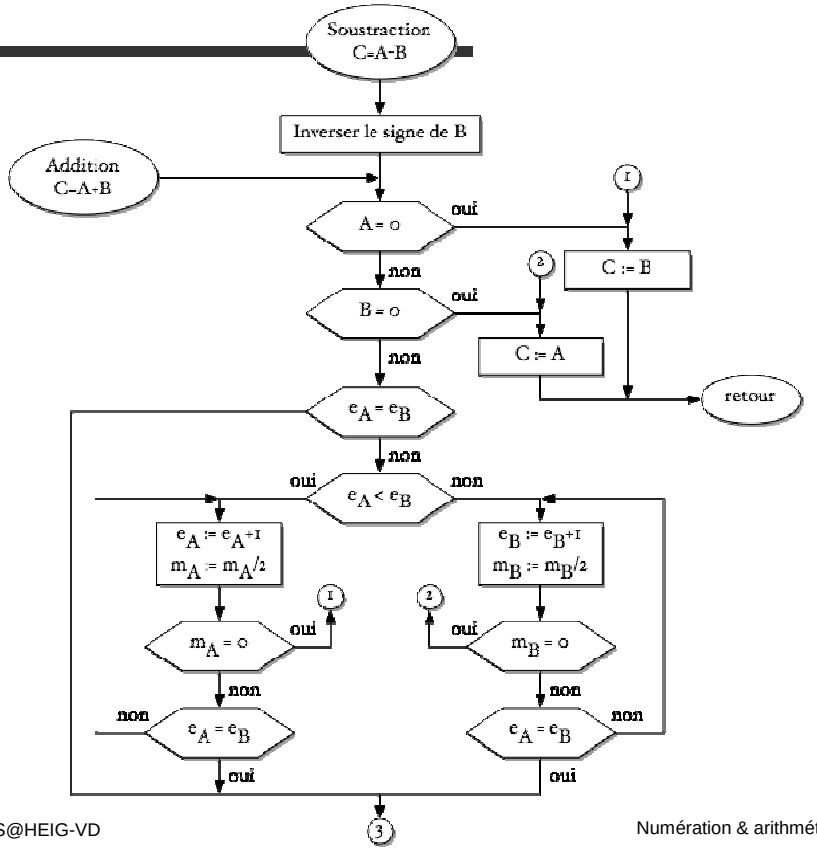
- au besoin, terminer par une normalisation

Opérations en virgule flottante (3)

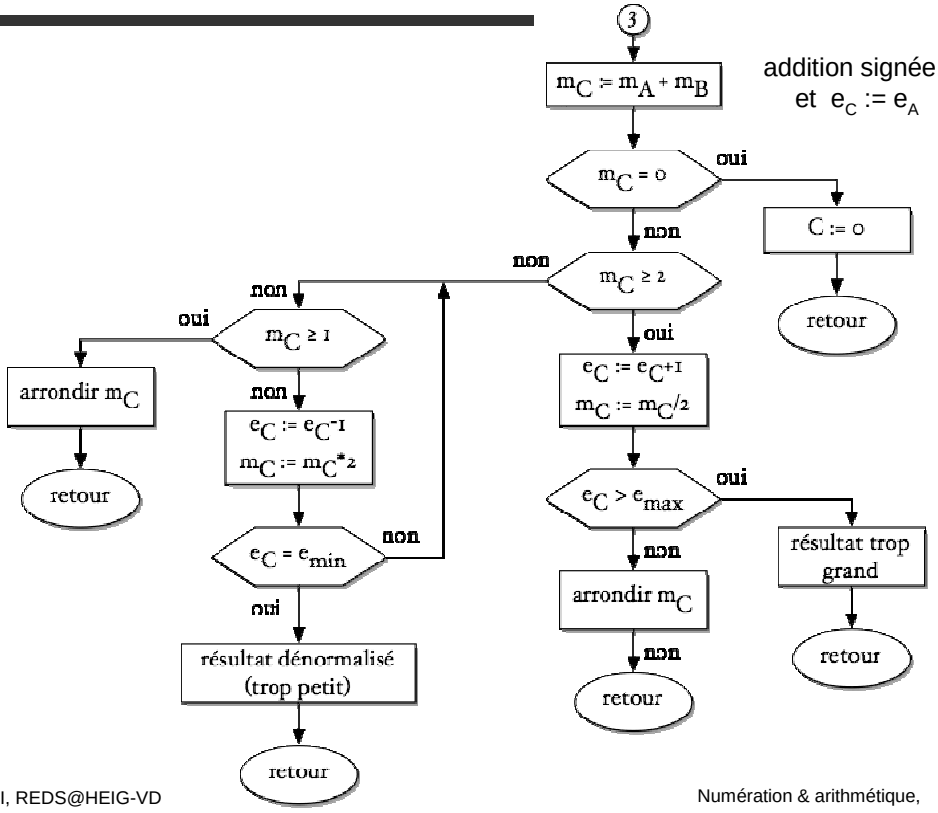
Soit pour chaque opération:

- $A+B = (m_A \cdot 2^{e_A - e_B} + m_B) \cdot 2^{e_B}$ si $e_A <= e_B$
- $A+B = (m_B \cdot 2^{e_B - e_A} + m_A) \cdot 2^{e_A}$ si $e_B <= e_A$
- $A-B = (m_A \cdot 2^{e_A - e_B} - m_B) \cdot 2^{e_B}$ si $e_A <= e_B$
- $A-B = (m_B \cdot 2^{e_B - e_A} - m_A) \cdot 2^{e_A}$ si $e_B <= e_A$
- $A \cdot B = (m_A \cdot m_B) \cdot 2^{e_A + e_B}$
- $A/B = (m_A / m_B) \cdot 2^{e_A - e_B}$

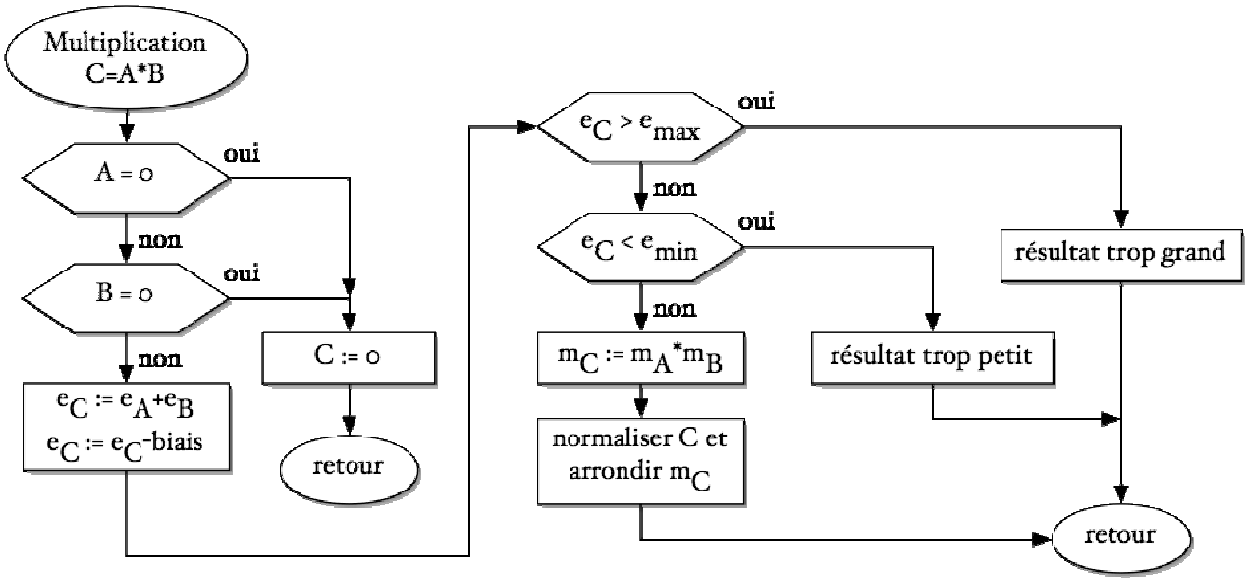
Addition / soustraction (1/2)



Addition / soustraction (2/2)



Multiplication



Division