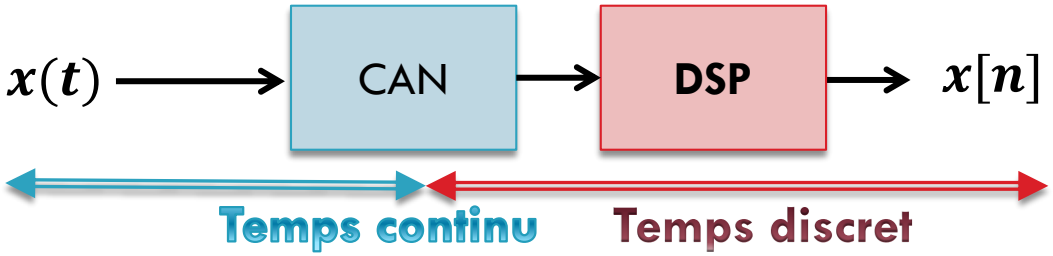
CHAPITRE 2

Virgule fixe Vs. Virgule flottante

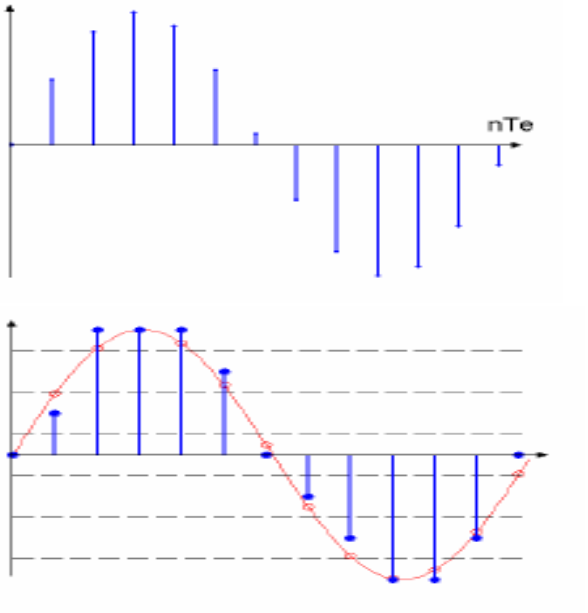
Introduction



- La conversion A-N permet la numérisation ou la discrétisation d'un signal analogique
- La discrétisation est effectuée en temps et en amplitude
- Le signal analogique doit avoir une bande limitée
  - Filtrage analogique anti-repliement

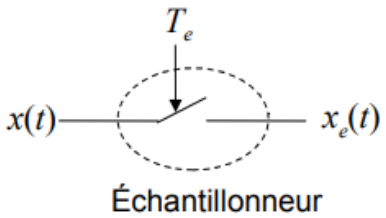
Conversion Analogique-Numérique

- La conversion analogique-numérique passe par deux étapes fondamentales:
  - Echantillonnage
    - Discrétisation du temps
    - Réversible
  - Quantification
    - Discrétisation des amplitudes du signal
    - Non-réversible



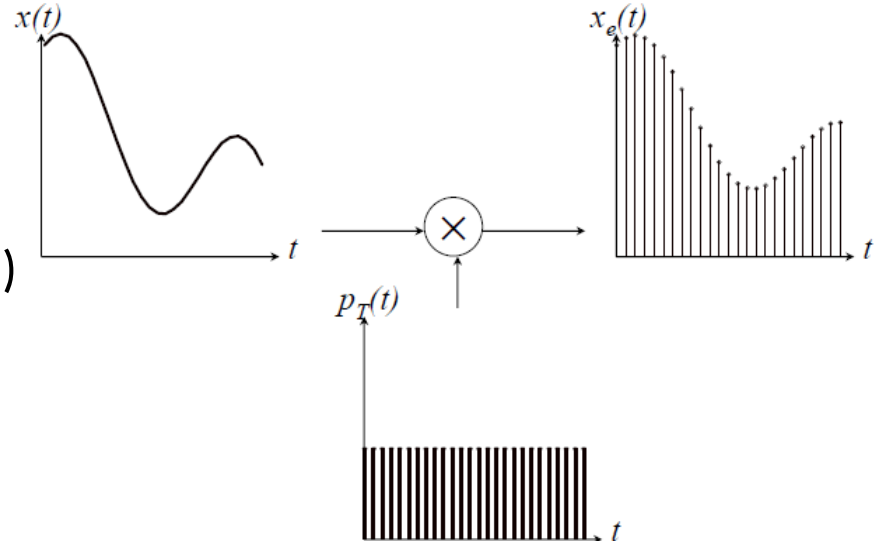
L'échantillonnage

- **Définition:**
  - L'échantillonnage consiste à prélever d'un signal $x(t)$ des échantillons à un intervalle régulier nT_e
  - T_e : Période d'échantillonnage où $F_e = \frac{1}{T_e}$ échantillons / sec (Hz)
- **Modélisation physique:**
  - Un échantillonneur est vu comme un interrupteur actionné à un intervalle régulier T_e



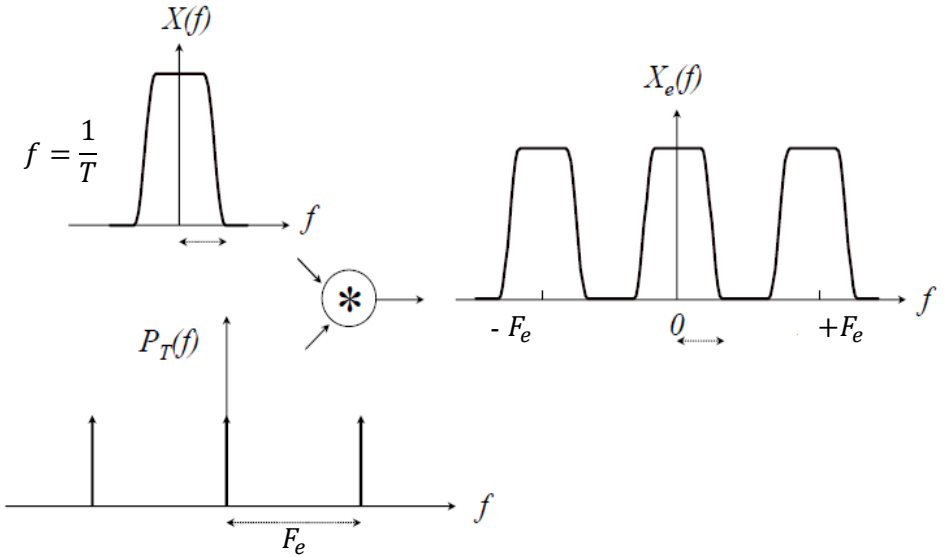
L'échantillonnage

- **Modélisation mathématique**
  - Domaine temporel:
    - $P_{T_e}(t) = \sum \delta(t - n T_e)$
    - $x_e(t) = \sum x(nT_e) \times \delta(t - n T_e)$
    - $\boldsymbol{x_e(t) = x(t) \times \sum \delta(t - n T_e)}$



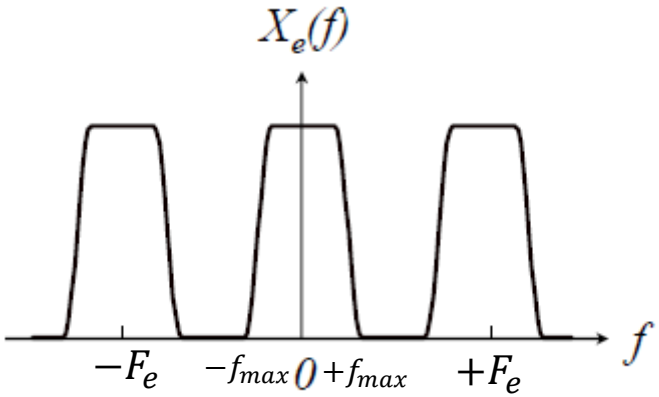
L'échantillonnage

- **Modélisation mathématique**
  - Domaine fréquentiel:
    - $P_{F_e}(f) = F_e \sum \delta(f - n F_e)$
    - $X_e(f) = X(f) * Pgn_{F_e}(f)$
    - $\boldsymbol{X_e(f) = F_e \sum X(f - n F_e)}$

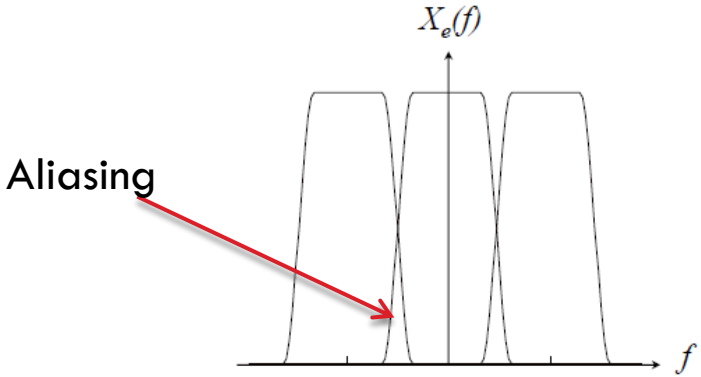


L'échantillonnage

- **Théorème de l'échantillonnage ou théorème de Shannon**
  - Repliement du spectre



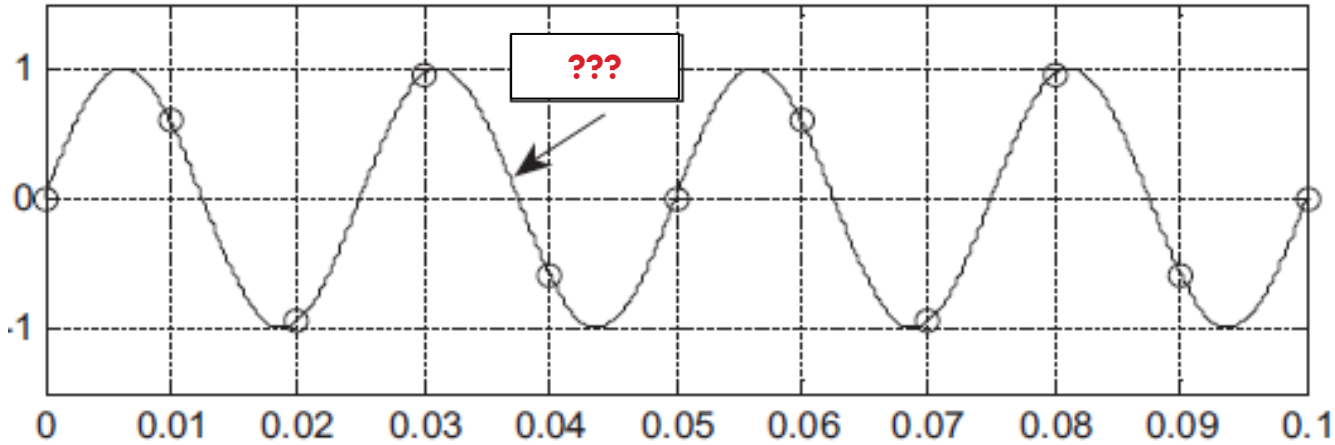
$F_e \geq 2\, f_{max}$



$F_e < 2\, f_{max}$

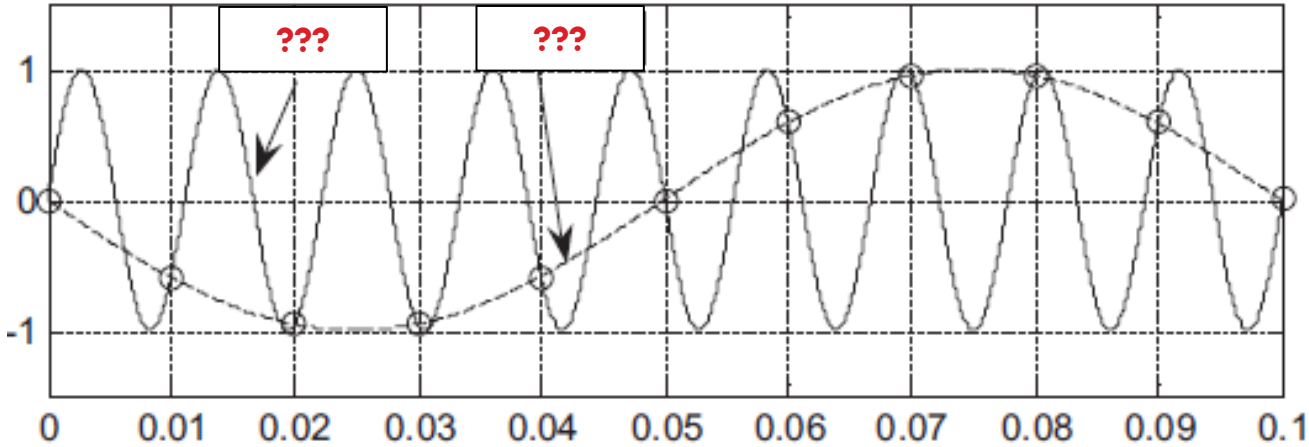
L'échantillonnage

- **Théorème de l'échantillonnage ou théorème de Shannon**
  - Exemple dans le domaine temporel:
    - Déterminez F_e et f_{max} dans les tracés suivants :



L'échantillonnage

- **Théorème de l'échantillonnage ou théorème de Shannon**
  - Exemple dans le domaine temporel:
    - Déterminez F_e et f_{max} dans les tracés suivants :

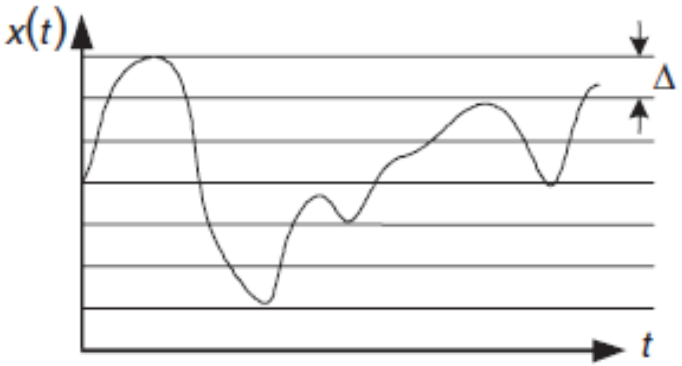


La quantification

- **Définition:**
  - Un processus irréversible permettant de faire correspondre aux amplitudes continues du signal échantillonné des amplitudes discrètes identiques et/ou approximatifs.
- **Types de quantifications:**
  - Quantification uniforme (linéaire)
  - Quantification non-uniforme
  - Quantification logarithmique

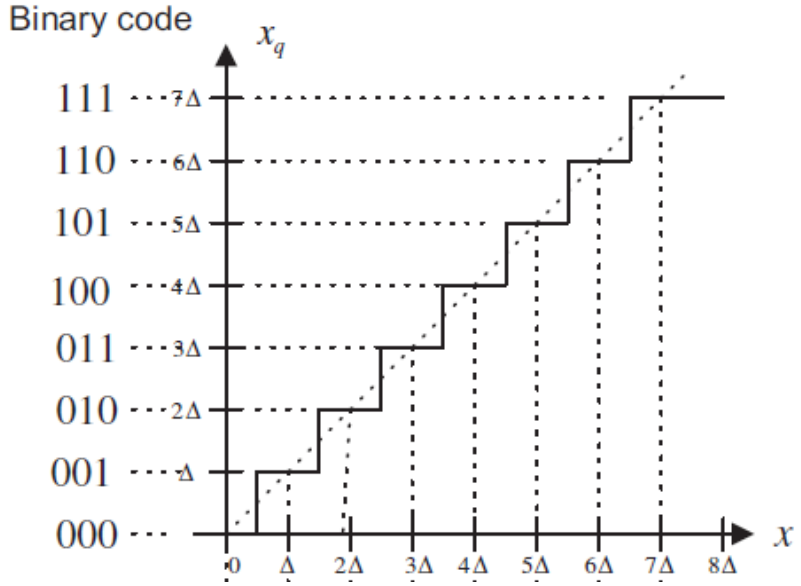
La quantification

- **Pas de quantification:**
  - La différence entre 2 niveaux adjacents en quantification est noté Δ
  - **Cas de la quantification uniforme :**
    - Pas constant
    - $\Delta = \frac{V_{ref+} - V_{ref-}}{2^N}$ où: N est la résolution CAN en bits
    - Valide pour un quantificateur unipolaire ou bipolaire



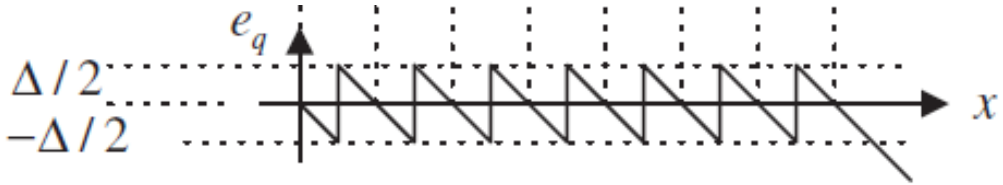
La quantification

- **Pas de quantification:**
  - Ainsi le pas de quantification pour un échantillon donné est défini par:
    - $X_q = V_{ref-} + i \,.\, \Delta$
    - où L'indice $i = round\left(\frac{X - V_{ref-}}{\Delta}\right)$
      - $i = 0, 1, \ldots, 2^N - 1$



La quantification

- **Erreur (bruit) de quantification:**
  - Lors de la reconstruction du signal (CNA), l'approximation des valeurs implique une erreur de quantification $\boldsymbol{e_q}$ uniformément distribuée sur l'intervalle $[-\frac{\Delta}{2}, \frac{\Delta}{2}]$
    - $\boldsymbol{e_q} = X_q - X$, $avec\ \boldsymbol{e_q} \in [-\frac{\Delta}{2}, \frac{\Delta}{2}]$
    - Espérance: $E(\boldsymbol{e_q}) = 0$

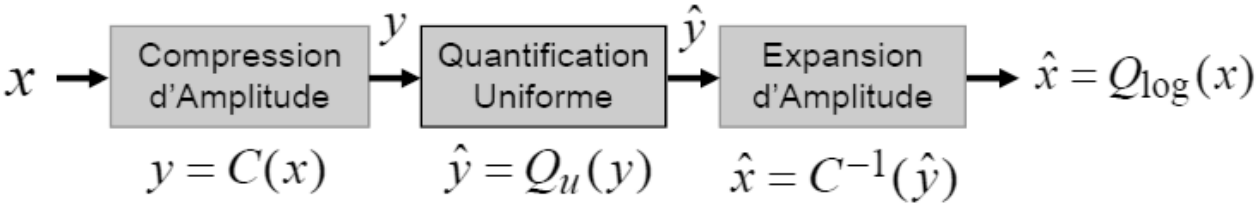


La quantification

- **Rapport Signal sur Bruit de quantification (SQNR):**
  - En pratique, le bruit de quantification dépend du pas de quantification :
    - $E\left(e_q^2\right) = \frac{\Delta^2}{12}$; puissance du bruit de quantification
    - $E\left(x^2\right) = x_{rms}^2$; puissance du signal
  - Le SQNR en dB est donné par:
    - $SQNR_{dB} = 10,79 + 20\, log_{10}\left(\frac{x_{rms}}{\Delta}\right)$
    - Dépend du pas de quantification et de la dynamique du signal

La quantification

- **Quantification logarithmique:**
  - Utilise la compression/expansion d'amplitude avec $SQNR_{dB}$ constant (erreur constante)
  - Dans le cas des signaux de la parole, deux lois sont utilisées:
    - Loi μ: aux États-Unis et Loi A: Reste du monde
  - Les circuits qui implémentent cela s'appellent des **compandeurs**



$$Q_{\log}(x) = C^{-1}\left(Q_u\left(C(x)\right)\right)$$

Entiers non signés

- Les entiers non signés représentent le zéro et les entiers positifs

- Format binaire sur N bits:
  - $X = \sum_{i=0}^{N-1} b_i \, . \, 2^i \; o\grave{u} \;\; b_i = 0 \; ou \; 1$

- Dynamique:
  - 0 à 2^N-1

Entiers non signés

- **Opérations arithmétiques binaires:**
  - Addition
    - 1 + 1 = 0 avec un report 1 (Carry en anglais)
  - Soustraction
    - 0 – 1 = 1 avec une retenue 1
  - Multiplication / Division
  - Complément à 1 (C1)
    - C1(1010) = 0101

Entiers signés

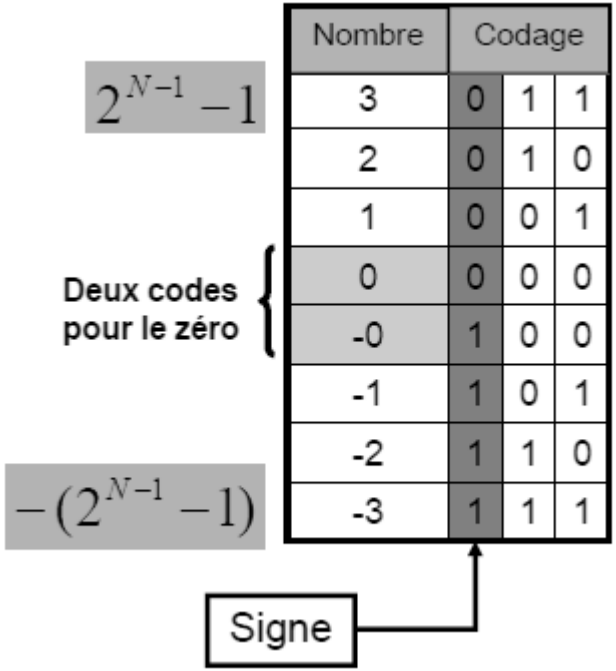
- **Formats de représentation des entiers signés:**
  - Format signe / valeur absolue
  - Format complément à 1 (C1)
  - Format complément à 2 (C2)

- **Ces formats ont les caractéristiques suivantes:**
  - Le bit de poids fort (MSB) est un bit de signe
  - Les N-1 bits restants sont en fonction du format

Entiers signés

- **Format Signe / Valeur absolue :**

  - Le bit de signe = 0 si l'entier est positif
  - Le bit de signe = 1 si l'entier est négatif
  - Dynamique:
    - $-(2^{N-1}-1) \rightarrow +(2^{N-1}-1)$



Entiers signés

- **Format C1:**
  - Le bit de signe = 0 si l'entier est positif
  - Le bit de signe = 1 si l'entier est négatif
  - Le reste des bits:
    - Positif → Format non signé
    - Négatif → C1 du format non signé
  - Dynamique:
    - $-(2^{N-1}-1) \rightarrow (2^{N-1}-1)$

$2^{N-1}-1$

$-(2^{N-1}-1)$

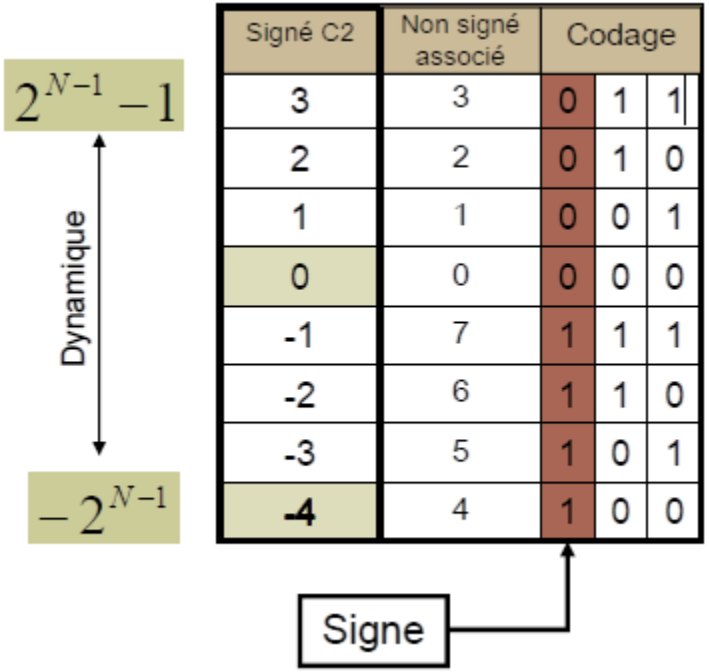
Deux codes pour le zéro

Signé C1	Non signé associé	Codage		
3	3	0	1	1
2	2	0	1	0
1	1	0	0	1
0	0	0	0	0
-0	7	1	1	1
-1	6	1	1	0
-2	5	1	0	1
-3	4	1	0	0

Signe

Entiers signés

- **Format C2:**
  - Le bit de signe = 0 si l'entier est positif
  - Le bit de signe = 1 si l'entier est négatif
  - Le reste des bits:
    - Positif → Format non signé
    - Négatif → C2 de son opposé
  - Dynamique:
    - $-2^{N-1} \rightarrow (2^{N-1}-1)$
  - Example sur le simulateur

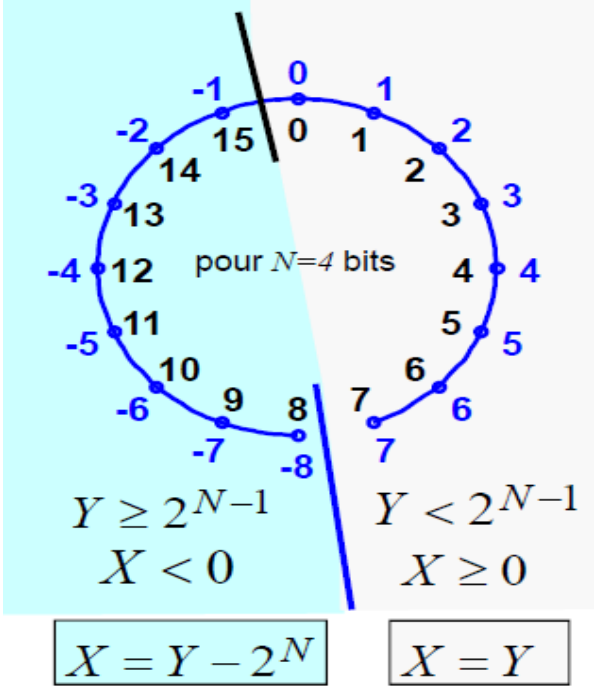


Signé C2	Non signé associé	Codage		
3	3	0	1	1
2	2	0	1	0
1	1	0	0	1
0	0	0	0	0
-1	7	1	1	1
-2	6	1	1	0
-3	5	1	0	1
-4	4	1	0	0

Entiers signés

- **Comparaison entier C2 et entier non signés**

  - X est un entier signé C2
  - Y est un entier non signé
  - Exemple avec N = 4 bits:



Entiers signés

- **Addition et soustraction en C2:**

  - L'opération de soustraction au format C2 est une **addition**
  - Exemple avec N = 4 bits:
    - 2 - 4 = 2 + C2(4) = 0010 + C2(0100)

      = 0010 +1100

      = 1110

      = - C2(1110) = - 2

Entiers signés

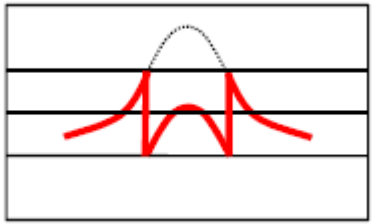
- **Extension du signe :**
  - Lors d'une opération arithmétique en C2, les deux nombres doivent avoir le même nombre de bits N
  - Une extension de bits doit être effectuée dans le cas de l'addition d'un entier C2 de N bits avec un autre de M bits où N < M, soit une extension du bit de signe de M-N bits
    - Exemple de l'extension de bit de signe de l'entier -5:
      - Pour N = 4: 1011
      - Pour M = 8: 1111 1011

Entiers signés

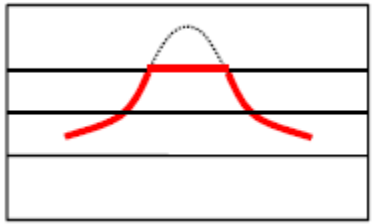
- **Dépassement de capacité:**
  - Un dépassement de capacité (overflow) est détecté s'il y a un changement de signe dans le résultat d'une addition de deux entiers C2 de même signe
  - Exemple 4 + 5 = 9 sur 4 bits:
    - 0100
    + 0101
    1 001 = - C2(1 001) = - 7 **FAUX !** ➔ OVERFLOW (**OV**)
  - Il faut noter que:
    - Les dépassement **intermédiaires** peuvent être tolérés si le résultat final peut tenir sur N-bits

Entiers signés

- **Gestion des dépassements :**
  - Pour éviter les OV lors d'une addition de M entiers C2 de N bits il faut:
    - **N+ log2(M) bits , soit au minium N+1 bits pour M = 2**
  - Modes de gestion :
    - **Mode sans saturation:**
      - Exemple pour N = 3 bits:
        - A = 3 + 1 ➔ OV !
        - A = **-4**



    - **Mode avec saturation**
      - Exemple pour N = 3 bits:
        - A = 3 + 1 ➔ OV !
        - **A = 3**



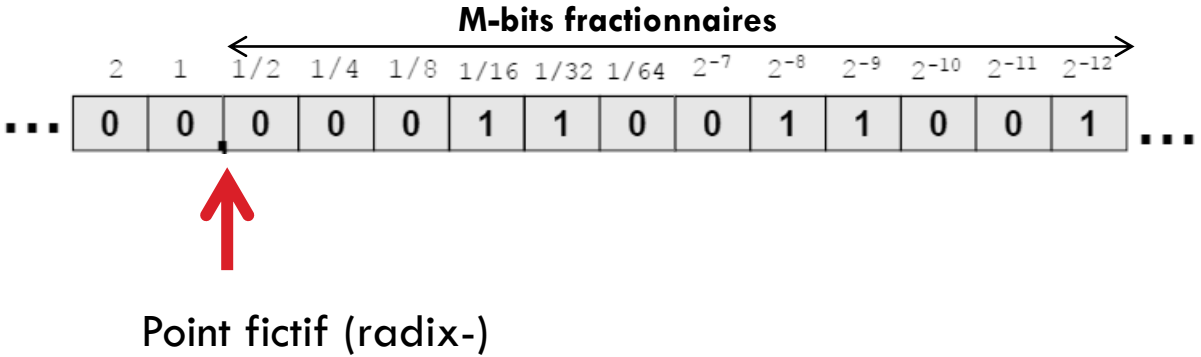
Format $Q_{N,M}$

- **Motivations:**

  - Contraintes de calcul :
    - Grande Précision ≠ Dynamique Large ?
  - Contraintes d'implémentation:
    - DSP à virgule fixe
  - Contraintes de dépassement de capacité (Multiplication / Addition)

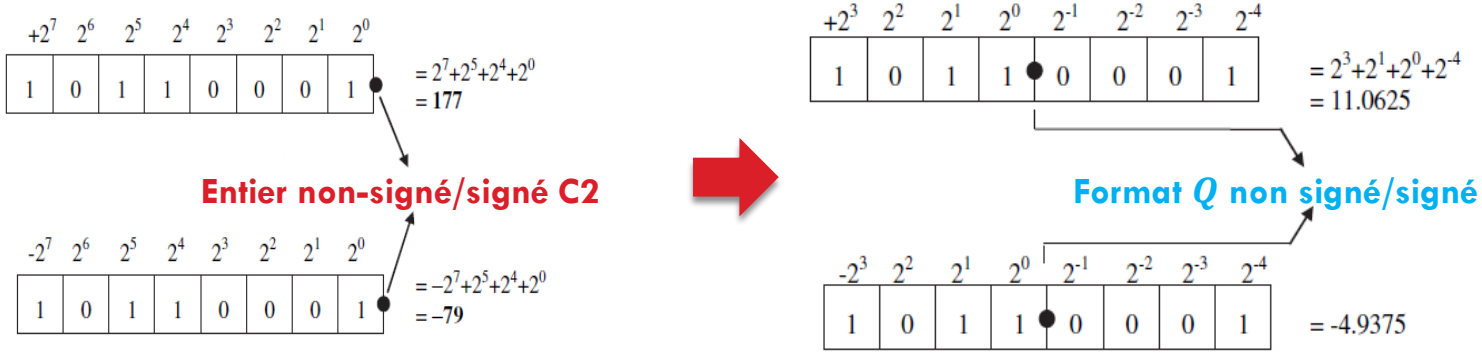
Format $Q_{N,M}$

- **Définition:**
  - Permet de représenter un nombre réel x sur N bits en utilisant une représentation fractionnaire en fonction d'un point fictif (radix-) considéré comme séparateur entre la partie fractionnaire (M-bits) et la partie entière (N-bits)



Format $Q_{N,M}$

- **Exemple :**
  - Supposons le cas du format Q 4,4



  - 0.5 au format Q2.3

Format $Q_{N,M}$

- **Précision et dynamique:**

  - La précision dans un format Q dépend du nombre des bits fractionnaires M :
    - $\Delta = 2^{-M}$

  - La dynamique du format Q:
    - $-2^{N-1} \rightarrow (2^{N-1} - \Delta)$
    - Q16,0 c'est le format entier (integer) : 0x8000 → 0x7FFF

Format $Q_{N,M}$

- **Opération arithmétiques binaires :**

  - **Addition:**

    - Si A → $Q_{N,M}$ et Si B → $Q_{N,M}$ ⇒ A + B → $Q_{N,M}$
    - Si A → $Q_{N,M}$ et Si B → $Q_{N',M'}$ ⇒ **Opération non possible**

Format $Q_{N,M}$

- **Opération arithmétiques binaires :**
  - **Multiplication**
    - Pas de problème de dépassement en multiplication format Q
    - Si A → $Q_{N,M}$ et Si B → $Q_{N',M'}$ ⇒ A x B → $Q_{N+N',M+M'}$
    - **Exemples:**

      1 - Unsigned x Unsigned

      Q3.3 a = 101.001

      b = 100.010

      a*b = 10101.110010

Format $Q_{N,M}$

- **2. Signed x Unsigned**

Q3.3
a = 101.001 et b=100.010
a*b = -C2(110011.110010)
= –(001100.001110)
= – 12.21875

1							1	0	1	0	0	1		-23
2		×					1	0	0	0	1	0		34
3			0	0	0	0	0	0	0	0	0	0	0	0	
4			1	1	1	1	1	1	0	1	0	0	1		
5			0	0	0	0	0	0	0	0	0	0			
6			0	0	0	0	0	0	0	0	0				
7			0	0	0	0	0	0	0	0					
8		+	1	1	0	1	0	0	1						
9		1	1	1	0	0	1	1	1	1	0	0	1	0	-782

Format $Q_{N,M}$

- **2. Unsigned x Signed**

Q2.3
a = 01.001 et b=10.010
a*b = 0001.100010
= -0001.111110
= -1.96875

1							0	1	0	0	1	9
2		×					1	0	0	1	0	-14
3							0	0	0	0	0	
4						0	1	0	0	1		
5					0	0	0	0	0			
6				0	0	0	0	0				
7		+	1	1	0	1	1	1				
8			1	1	1	0	0	0	0	0	1	0	-126

Format $Q_{N,M}$

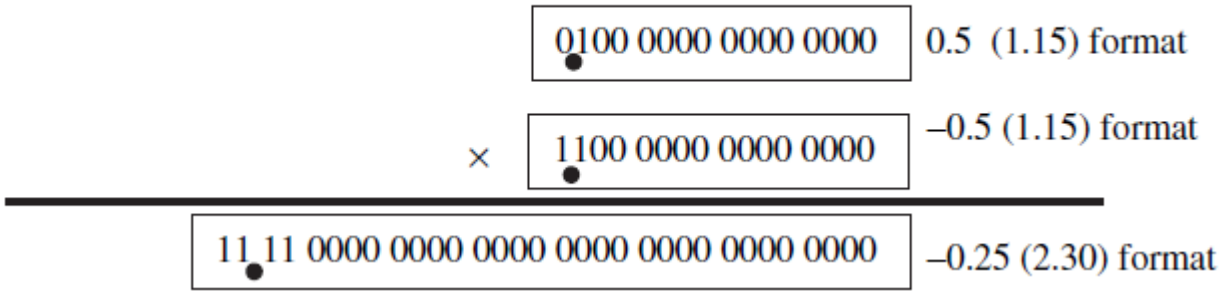
- **2. Signed x Signed**

Q2.3
a = 11.001 et b=10.010
a*b = 0001.100010
= 1.53125

1							1	1	0	0	1	-7
2		×					1	0	0	1	0	-14
3			0	0	0	0	0	0	0	0	0	0	
4			1	1	1	1	1	1	0	0	1		
5			0	0	0	0	0	0	0	0			
6			0	0	0	0	0	0	0				
7		+	0	0	0	1	1	1					
8			0	0	0	1	1	0	0	0	1	0	98

Format $Q_{N,M}$

- **Opération arithmétiques binaires :**
  - **Multiplication**
    - Pas de problème de dépassement en multiplication format Q
    - Si A → $Q_{N,M}$ et Si B → $Q_{N',M'}$ ⇒ A x B → $Q_{N+N',M+M'}$
    - **Exemple:**



Format $Q_{N,M}$

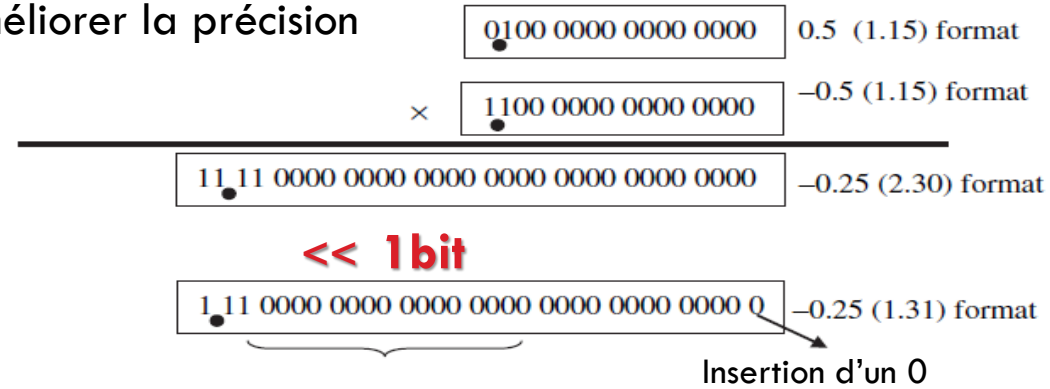
- **Opération arithmétiques binaires:**

  - **Décalage**

    - Si A → $Q_{N,M}$ ⇒ A << n → $Q_{N,M+n}$ (insertion de zéros)
    - Si A → $Q_{N,M}$ ⇒ A >> n → $Q_{N,M-n}$

Format $Q_{N,M}$

- **Troncation :**
  - Exemple dans le cas d'une multiplication **d'entiers signés C2**
  - Implique une **erreur de précision** due à la troncation
  - L'arrondi peut améliorer la précision



- **Résultat converti au format (Q1.15) après un décalage de 1 bit et une troncation sur la partie fractionnaire**

Application sur le simulateur

Format $Q_{N,M}$

- **Gestion des dépassements dans le format Q:**
  - Ce problème existe toujours dans l'opération d'addition au format Q
  - Solutions possibles au problème:
    - Utilisation d'un format Q différent au détriment d'une perte en précision
    - Utilisation d'une arithmétique à saturation (flags)
    - Une mise à l'échelle du signal à traiter (perte de 6dB du SQNR)
    - Utilisation des bits de gardes dans les accumulateurs
    - Un mix entre toutes les solutions proposées

Représentation en virgule flottante

- **Introduction:**

  - Un nombre réel s'écrit de différentes manières :

    - $(+25.375)_{10} = +2,5375 . 10^{1} = +0,25375 . 10^{2} = +0,025375 . 10^{3}$

    - $(+11001.011)_{2} = +1100.1011 \times 2^{1} = +110.01011 \times 2^{2}$

Représentation en virgule flottante

- **Normalisation :**
  - Pour éviter différentes représentations du même nombre, la mantisse est normalisée sous la forme générale **1.F** :
    - $\pm \mathbf{1}.bbb \ldots bb \times 2^{\pm e}$
    - e : Exposant
  - **Example:**
    - $+1100.1011 \times 2^{1} = +\mathbf{1}.1001011 \times 2^{4}$

Représentation en virgule flottante

- **Représentation normalisée en virgule flottante :**
  - Le codage d'un nombre réel sur N bits en virgule flottante **1.F**:
    - $x = (-1)^S \ \ 1,\mathrm{M} \,.\, 2^E$
      - **S: Bit de signe (1 bit)**
      - **M (Mantisse): réel positif sur M-bits où $1 \leq 1,M < 2$**
      - E (Exposant): Entier sur **E**-bits

Représentation en virgule flottante

- **Représentation non-normalisée en virgule flottante :**
  - Impossibilité de représenter la valeur 0,0 dans une représentation normalisée, car le nombre à gauche de la mantisse est toujours égale à 1
  - Pour résoudre ce problème, on passe à une représentation non-normalisée :
    - Mettre les bits de l'exposant à 0
    - Mettre la mantisse sous la forme:
      - $0 \leq 0,M < 1$

Représentation en virgule flottante

- **Représentation non-normalisée en virgule flottante :**
  - On aura donc deux valeurs pour le zéro :
    - + 0,0 et – 0,0

  - Les nombres non normalisés différents de 0.0 représentent de très petits nombres proches de la valeur 0.0

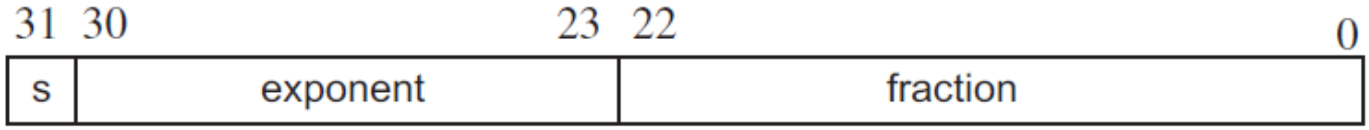
Représentation en virgule flottante

- **Valeurs particulières:**
  - Infinies :
    - $+\infty$ et $-\infty$

  - NaN (Not a Number) :
    - Signe indifférent

Représentation en virgule flottante

- **Standard IEEE 754**
  - **Format SP (Simple Précision 32bits) :**



$$x = (-1)^s \times (1.F) \times 2^{E-127}$$

- S: Bit de signe
- F: Bits Fractions (ou M: Mantisse) 23 bits
- E: bits Exposant 8 bits avec un décalage (biais) maximum de: $2^{8-1} - 1 = 127$

Exemple:

0 10000001 10100000000000000000000

Représentation en virgule flottante

- **Standard IEEE 754**
  - **Format SP (Simple Précision 32bits) :**
    - Valeur normalisée maximale :
      - 0 111…0 1111….1 $= 1.111\ldots1 \times 2^{254-127}$
      - $3.40282346 \times 10^{38}$
    - Valeur normalisée minimale:
      - 0 000…1 0000….0 $= 1.000\ldots0 \times 2^{1-127}$
      - 1.175494×10^{-38}

Représentation en virgule flottante

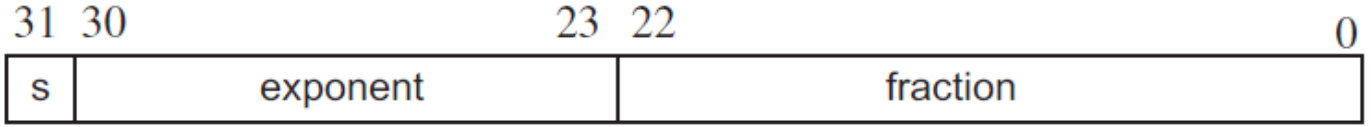
- **Standard IEEE 754**
  - **Format SP (Simple Précision 32bits) :**
    - Valeur non-normalisée maximale:
      - 0 000… 1111….1 $= 0.111 \ldots 1 \times 2^{-126}$
    - Valeur non-normalisée minimale :
      - 0 000… 0000….1 $= 0.000 \ldots 1 \times 2^{-126}$

Représentation en virgule flottante

- **Standard IEEE 754**
  - **Format SP (Simple Précision 32bits) :**
    - Valeurs particulières:
      - $Infini$:
        - 0 111…1 000….0 $= +\infty$
        - 1 111…1 000….0 $= -\infty$
      - $NaN\ (Not\ a\ number)$
        - Signe indifférent
        - Exposant = 111…1
        - **Mantisse ≠ 0**

Représentation en virgule flottante

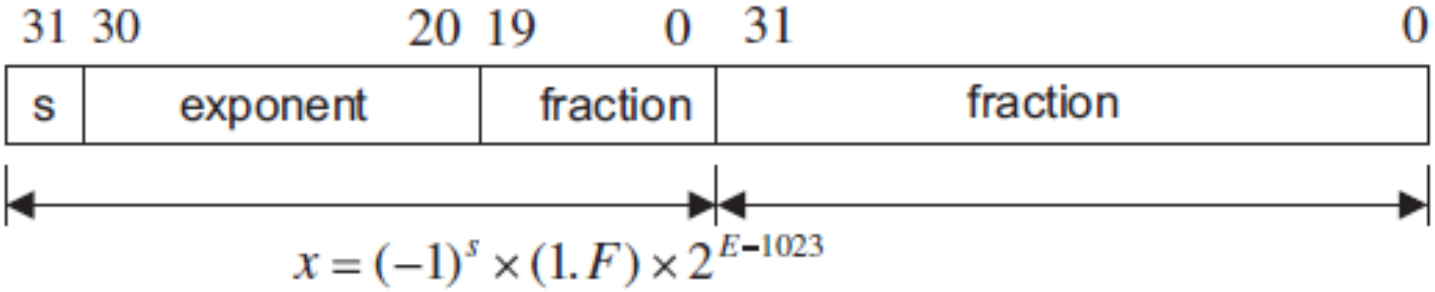
- **Dynamique vs. Précision**
  - Augmentation du nombre de bits exposant → Dynamique s'étend
  - Fixation du nombre de valeurs représentables → Précision diminue
  - Pour augmenter la précision et la dynamique en parallèle il faut augmenter le nombre de bits total N



$$x = (-1)^s \times (1.F) \times 2^{E-127}$$

Représentation en virgule flottante

- **Standard IEEE 754**
  - **Format DP (Double Précision 64bits) :**



- S: Bit de signe
- F: Bits Fractions (Mantisse) 52 bits
- E: Bits Exposant 11 bits avec un décalage (biais) maximum de: $2^{11-1} - 1 = 1023$

Représentation en virgule flottante

- **Standard IEEE 754**
  - **Format DP (Double Précision 64bits) :**
    - Valeur non-normalisée maximale:
      - 0 000… 1111….1 $= 0.111 \ldots 1 \times 2^{-1022}$
    - Valeur non-normalisée minimale :
      - 0 000… 0000….1 $= 0.000 \ldots 1 \times 2^{-1022}$

Représentation en virgule flottante

- **Standard IEEE 754**

Exemples SP :

85,125 ----- 01000010101010010000000000000000

-4.5 ------ 11000000100000011001100110011001

0.1 ------- 00111101110011001100110011001100

Avantages et inconvénients

- **Virgule Fixe**
  - **Dynamique limitée:** position de la virgule fixé au début
  - Pour une taille fixe d'un mot, 32bits par exemple :
    - Si on augmente le nombre de bits N on augmente la dynamique mais on réduit la précision
    - Si on augmente le nombre de bits M on augmente la précision mais on réduit la dynamique

- **Virgule Flottante**
  - **Dynamique plus large:** position de la virgule variable(flottante) en fonction du besoin (bits exposant)
  - Pour une taille fixe d'un mot, 32bits par exemple :
    - La dynamique dépend des bits de l'exposant
    - La précision dépend des bits de la mantisse
    - La précision ne change pas si on change la dynamique (Précision constante)

Avantages et inconvénients

- **XILINX White Paper: Floating vs Fixed Point:**

https://www.xilinx.com/support/documentation/white_papers/wp491-floating-to-fixed-point.pdf

	DSP48E2		LUT	
	Resource Count	**Device Utilization**	**Resource Count**	**Device Utilization**
Single-Precision Floating Point	4,230	62%	231,060	20%
Fixed Point	850	12%	19,730	2%

Avantages et inconvénients

- **XILINX White Paper: Floating vs Fixed Point:**

https://www.xilinx.com/support/documentation/white_papers/wp491-floating-to-fixed-point.pdf