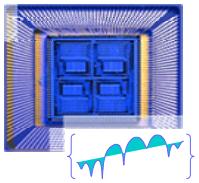
Implantation d'algorithmes de traitement du signal sur les architectures virgule fixe



Daniel Ménard, Olivier Sentieys
IRISA/ENSSAT Lannion
Université de Rennes I
sentieys@irisa.fr
http://www.irisa.fr/R2D2

IRISA

Plan

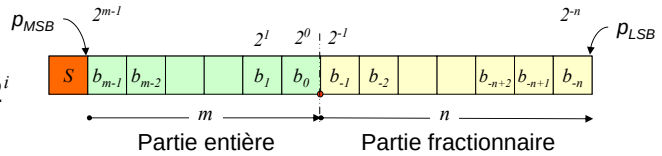
- 1. Arithmétique virgule fixe
- 2. Détermination du domaine de définition
- 3. Codage des données

IRISA

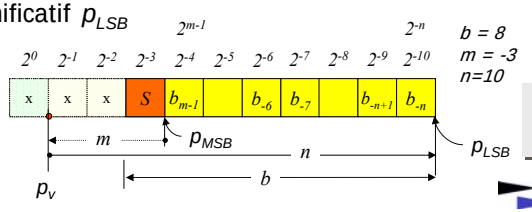
Codage en virgule fixe complément à 2

- **Définition :**

$$x = -2^m S + \sum_{i=-n}^{m-1} b_i 2^i$$



- ***m*** : distance (en nombre de bits) entre la position du bit le plus significatif p_{MSB} et la position de la virgule p_V
- ***n*** : distance entre la position de la virgule p_V et la position du bit le moins significatif p_{LSB}



b = 8
m = -3
n=10

ex: 0.09472 (8,-3,10)
01100001
0.0625+0.03125+2^{-10}

$b = m + n + 1$ bits
format: *(b,m,n)*

IRISA

Codage en virgule fixe complément à 2

- Domaine de définition du codage : $\left[-2^m, 2^m - 2^{-n}\right]$
- Pas de quantification

$$q = 2^{-n}$$

- Exemple de codage
  - (6,3,2), Q26

0111.11	7.75
0111.10	7.5
0000.11	0.75
0000.10	0.5
0000.01	0.25
0000.00	0
1111.11	-0.25
1111.10	-0.5
1111.01	-0.75
1000.01	-7.75
1000.00	-8

-0.5 = -8+7. 5

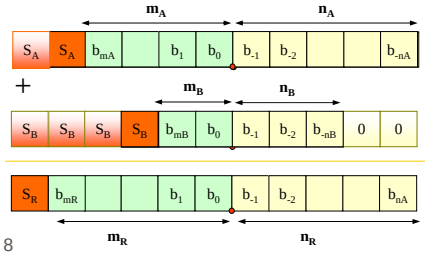
-7.75 = -8+0.25

IRISA

Règles de l'arithmétique virgule fixe

- Addition: $r=a+b$
  - Règle : le format des opérandes a et b doit être identique
  - Étapes
    - Choix d'un format commun (b_c,m_c,n_c)
    - Alignement de la virgule
    - Extension des bits des opérandes *a* et *b*

$$\begin{aligned} m_c &= \max(m_A,m_B) \\ n_c &= \max(n_A,n_B) \\ b_c &= m_C+n_C+1 \end{aligned}$$



$$\begin{aligned} m_R &= \begin{cases} m_C+1 & \text{si } a+b \notin D_C \\ m_C & \text{si } a+b \in D_C \end{cases} \\ n_R &= n_C \\ b_R &= m_R+n_R+1 \end{aligned}$$

IRISA

Exemple de calcul

- Addition: $r=a+b$
  - Débordement possible si $sign(a) = sign(b) = s_{ab}$
    - Débordement présent si $sign(r) \neq s_{ab}$

Extension du bit de signe

```
 1 1 1            1 1 1            1 1 1
 0110  (6)        0011  (3)        00011  (3)
 1011  (-5)       0111  (7)   ⇨    00111  (7)
 ----             ----             -----
 0001  (1)        1010  (-6)       01010  (10)
```

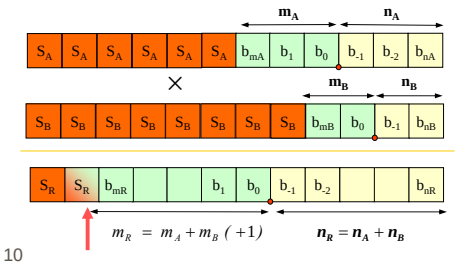
S'il n'y a pas de débordement la dernière retenue peut être ignorée

Nécessité d'un bit supplémentaire pour coder le résultat

IRISA

Règles de l'arithmétique virgule fixe

- Multiplication: $r = a \times b$
  - Règle : la représentation de a et b doit être identique
  - Étapes
    - Extension des bits de signe des opérandes *a* et *b*
    - Doublement du bit de signe du résultat
      - Le bit redondant peut être intégré à la partie entière du résultat



$$\begin{aligned} n_R &= n_A+n_B \\ m_R &= m_A+m_B+1 \\ b_R &= b_A+b_B \end{aligned}$$

IRISA

Exemple de calcul

- Multiplication: $r = a \times b$
  - Extension du signe des opérandes

```
      1100  (-4)
      0110  ( 6)
    -------
   1111100.
   111100. .
  1 1 1
  ---------
  11101000
```

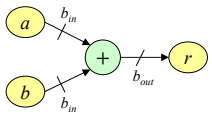
-128+64+32+8 = -24

```
    111110  (-2)
    111100  (-4)
    ------
    1110 . .
    110 . . .
    10 . . . .
   1
   ---------
   001000  (8)
```


IRISA

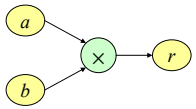
Résumé des règles

- Addition
  - Choix du format commun



si $b_{out} = b_{in}$ alors $m_c = \max(m_A, m_B, m_R)$
si $b_{out} > b_{in}$ alors $m_c = \max(m_A, m_B)$

- Multiplication



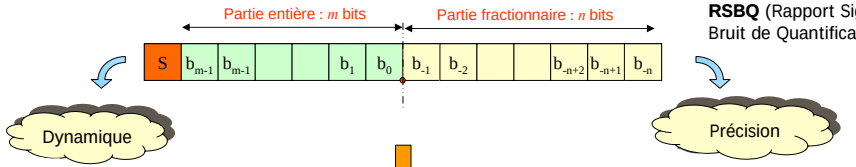
$n_R = n_A + n_B$
$m_R = m_A + m_B + 1$
$b_R = b_A + b_B$

Doublement du bit de signe

IRISA

Codage en virgule fixe : objectifs

- Objectifs pour le codage en virgule fixe
  - Garantir l'absence de débordements
    - Connaissance du domaine de définition des données
  - Maximiser la précision



RSBQ (Rapport Signal à Bruit de Quantification)

Dynamique → Garantire l'absence de débordement

Respecter les règles de l'arithmétique virgule fixe

Précision → Maximiser la précision des calculs

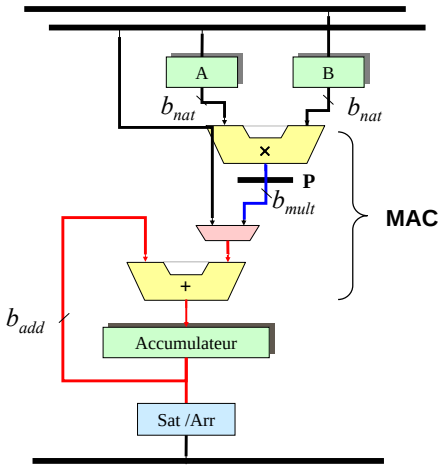
IRISA

Eléments de l'unité de calcul

Opérateurs	N_{bits} Entrées	N_{bits} Sortie
Multiplieur	b_{nat}	$2\ b_{nat}$
Additionneur/ UAL	$2\ b_{nat}$ $2\ b_{nat} + b_g$	$2\ b_{nat}$ $2\ b_{nat} + b_g$
Saturation/Arrondi	$2\ b_{nat}$ $2\ b_{nat} + b_g$	b_{nat} b_{nat}

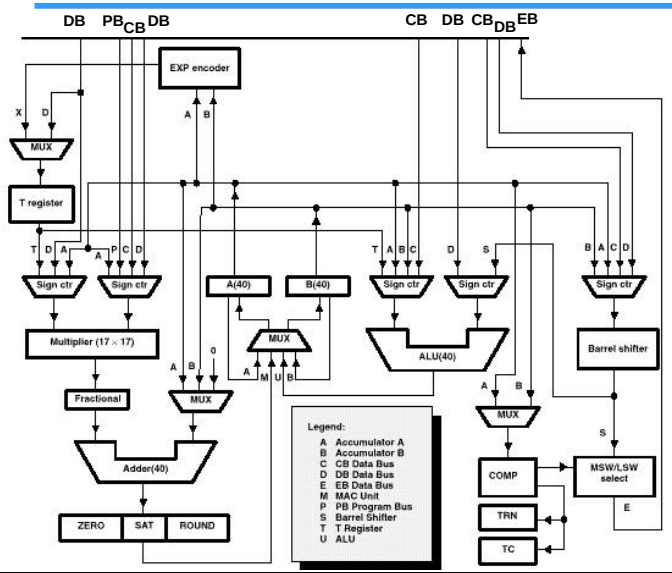
Registre	N_{bits}
Opérande source	b_{nat}
Accumulateur	$2\ b_{nat}$ $2\ b_{nat} + b_g$

Interconnexions registres - opérateurs spécialisées
⇨ Structure hétérogène
⇨ Bonnes performances en terme de consommation et de surface



IRISA

Exemple : TMS320C54x



1 multiplieur 16*16 bits
  Op source 1 : registre T
  Op source 2 : mémoire
  Op destination :

1 additionneur 40 bits
1 ALU (40 bits)

2 registres accumulation 40 bits

1 registre à décalage en barillet

1 unité dédiée Viterbi

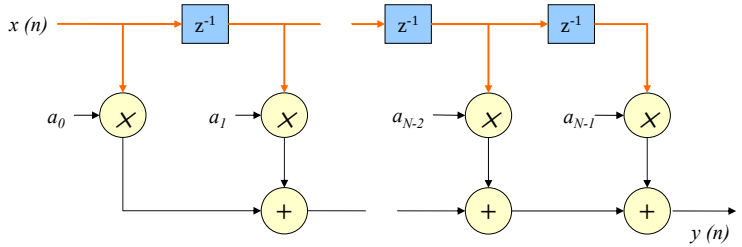
IRISA

Codage en virgule fixe : étapes

1. Détermination de la dynamique des données : $D(x_i)$
   - Méthodes statistiques ou analytiques
2. Détermination de la position de la virgule : m_{x_i}
3. Détermination de la largeur des données : b_{x_i}
   - Opérateurs SIMD, précision multiple
4. Evaluation de la précision des calculs
   - Méthodes par simulation ou analytiques
   - Calcul du Rapport Signal à Bruit de Quantification

Fil rouge : filtre FIR

$$y(n) = \sum_{i=0}^{N-1} a_i \times x(n-i)$$



Fil rouge : filtre FIR

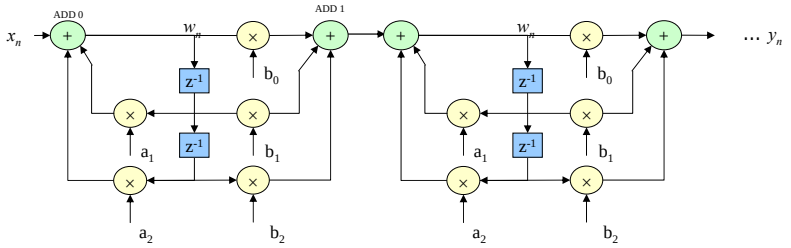
```
float x[N] = {0.123, -0.569,...} /* Signal d'entrée du filtre */
float h[N] = {-0.0295, 0.0364,.., -0.0295 }; /* Coefficients du filtre */
int main() {
  float x[N], y[M], acc;
  int i,j;

  for(i=0; i<N; i++){x[i] = 0;}    /* Initialisation variables internes */

  for(j=0;  j<M; j++) {            /* Filtrage du vecteur d'entrée input */
        x[0] = Input[j];
        acc = x[0]*h[0] ;
        for(i=N-1; i>0; i--)
        {
          acc = acc + x[i]*h[i]; /* Calcul d'une cellule du filtre */
          x[i] = x[i-1];         /* Vieillissement des variables internes */
        }
        y[j] = acc;
   }
}
```


Fil rouge 2 : filtre récursif (IIR)

- Cascade de cellules d'ordre 2



$$w_i(n) = x(n) - \sum_{j=1}^{2} a_{j,i} w_i(n-j)$$

$$y(n) = \sum_{j=0}^{2} b_{j,i} w_i(n-j)$$

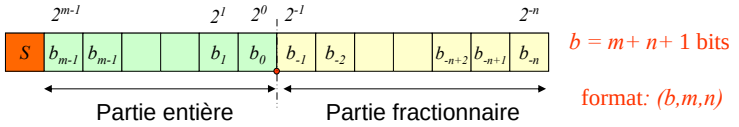
Plan

- 1. Arithmétique virgule fixe
- 2. Détermination du domaine de définition
- 3. Codage des données

IRISA

Détermination du domaine de définition

- Format des données: *(b,m,n)*



- But: *déterminer le nombre de bits minimal pour représenter la partie entière* (m_{xi})

$$m_{x_i} = \lceil \log_2(\max(|x_i|)) \rceil$$

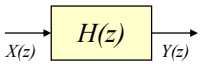
IRISA

Méthodes pour déterminer $D(x_i)$

- Méthodes statistiques
  - Simulation de l'algorithme en virgule flottante
  - Détermination de la dynamique à partir des paramètres statistiques du signal
    - ⇨ Estimation précise mais dépendante du signal d'entrée
- Méthodes analytiques
  - Détermination de l'expression de la dynamique
  - Utilisation de normes (L_1, ...) ou de l'arithmétique d'intervalle
    - ⇨Estimation conservatrice mais absence de débordement garantie

IRISA

Méthodes analytiques



- Normes dans le cas des systèmes linéaires
  - Norme L1

    $y_{\max 1} = \max_n(|x(n)|) \cdot \sum_{m=-\infty}^{\infty} |h(m)|$

    ➢ Méthode garantissant l'absence de débordement
  - Norme Chebychev

    $y_{\max 2} = \max_n(|x(n)|) \max_\omega(|H(\omega)|)$

    ➢ Méthode garantissant l'absence de débordement pour un signal d'entrée à bande étroite ($x(n)=cos(wt)$)
  - Norme L2

    $y_{\max 3} = \max_n(|x(n)|) \cdot \sqrt{\sum_{m=-\infty}^{\infty} |h(m)|^2}$

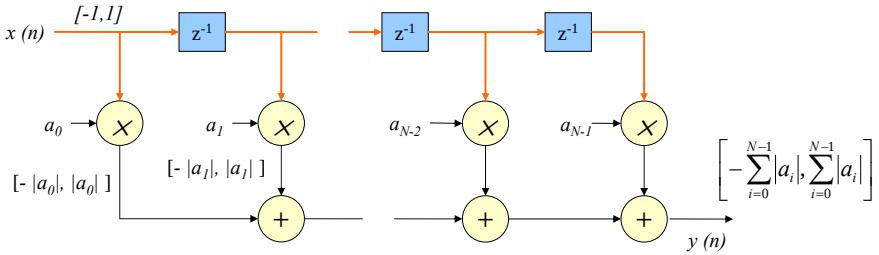
    ➢ Méthode limitant la probabilité de débordement

$$y_{\max 3} \leq y_{\max 2} \leq y_{\max 1}$$

IRISA

Filtre FIR

- Propagation de la dynamique (intervalles) des entrées au sein du GFS représentant l'application



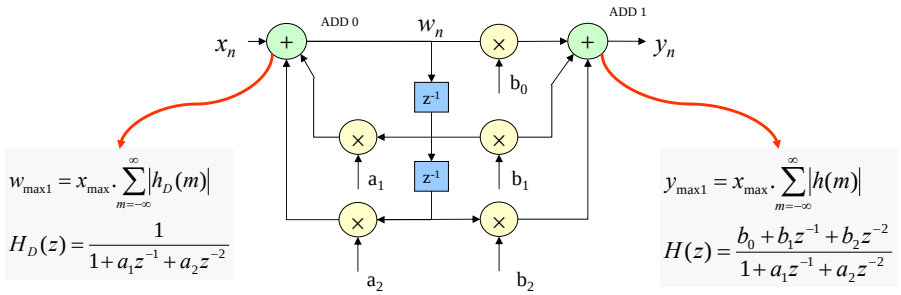
- Norme L1

$$\max_n\left(\left|y(n)\right|\right) = \max_n\left(\left|x(n)\right|\right).\sum_{m=-\infty}^{\infty}\left|h(m)\right| = \sum_{m=0}^{N-1}\left|a_i\right| = 1.65$$

IRISA

Dynamique des données (IIR)

- Sources de débordements : additionneurs
  - IIR : 2 sources ADD0 et ADD1



IRISA

Dynamique des données (IIR)

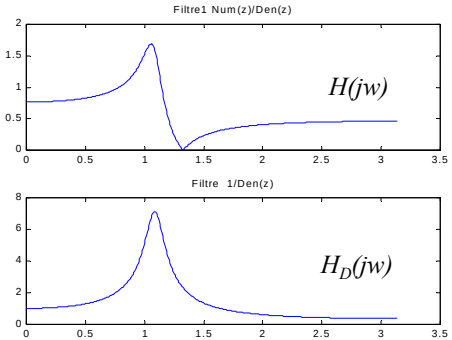
- Exemple

$$x \in [-1,1]$$

$$H(z) = \frac{0.5 - 0.2428z^{-1} + 0.5}{1 - 0.85z^{-1} + 0.8417z^{-2}}$$

$$x_{\max}\sum_{m=-\infty}^{\infty}\left|h(m)\right| = 2.5645 \longrightarrow m_y = 2$$

$$x_{\max}\sum_{m=-\infty}^{\infty}\left|h_D(m)\right| = 9 \longrightarrow m_w = 4 \rightarrow k_1 = 4$$



Fonction de transfert des filtres H(z) et $H_D(z)$

Codage des coefficients

```
a1 = -27951
a2 =  27581
b2 =  16384
b0 =  16384
b1 = -7956
```


IRISA

Dynamique des données (IIR)

- Comparaison des différentes méthodes

Méthodes	$G_{1.max}$	N_1	$G_{2.max}$	N_2
Méthodes analytiques				
Norme L_1	2,56	2	9	4
Norme Chebychev	1.687	1	7,13	3
Méthodes statistiques				
Chirp	1,66	1	7.12	3
Bruit blanc gaussien	0.74	0	2.34	2
Bruit blanc uniforme	1.4	1	4.87	3

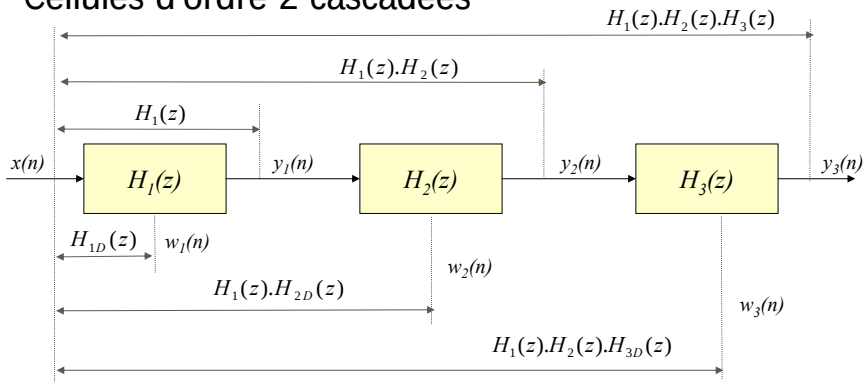
$$G_{1\max} = \frac{y_{\max}}{x_{\max}} \qquad N_1 = \lceil \log_2(G_{1\max}) \rceil$$

$$G_{2\max} = \frac{w_{\max}}{x_{\max}} \qquad N_2 = \lceil \log_2(G_{2\max}) \rceil$$

IRISA

Dynamique des données (IIR)

- Cellules d'ordre 2 cascadées



$$H_{iD}(z) = \frac{1}{1 + a_{1i}z^{-1} + a_{2i}z^{-2}} \qquad H_i(z) = \frac{b_{0i} + b_{1i}z^{-1} + b_{2i}z^{-2}}{1 + a_{1i}z^{-1} + a_{2i}z^{-2}}$$

IRISA

Plan

IRISA

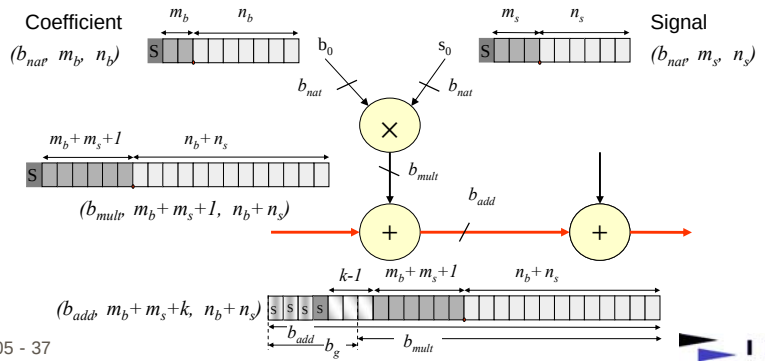
Contraintes du codage en virgule fixe

- Respect des règles de l'arithmétique virgule fixe
  - Alignement de la virgule des opérandes sources d'une addition ou d'une soustraction
    - Format commun pour les opérandes sources
- Prévenir les débordements
  - Débordement possible lors d'une addition ou d'une soustraction
    - Nécessité d 'introduire des bits supplémentaires
      - Recadrage des données : adapter le format des données à la dynamique des données

IRISA

Introduction des bits supplémentaires

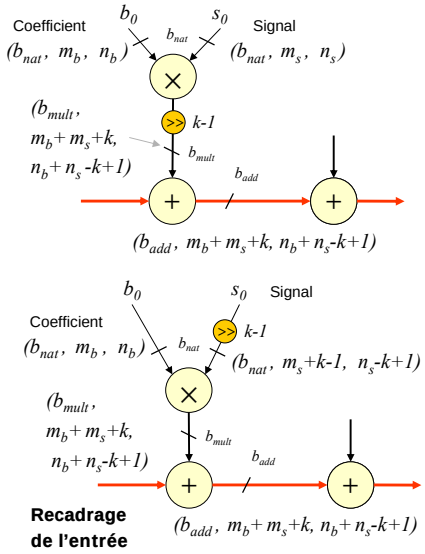
- Utilisation des bits de garde de l'accumulateur (b_g bits)
  - $b_{add} = b_{mult} + b_g$
  - Pas de débordement si le nombre de bits supplémentaires k respecte $k \leq b_g$



IRISA

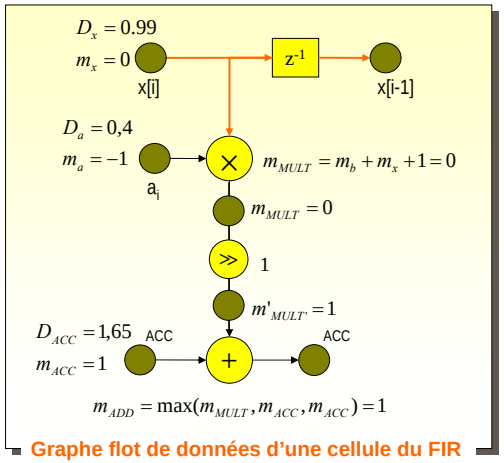
Introduction des bits supplémentaires

- Recadrage de la sortie du multiplieur $(b_{add} = b_{mult})$
  - Recadrage interne
- Recadrage avant la multiplication $(b_{add} = b_{mult})$
  - Recadrage de l'entrée
  - Recadrage des coefficients
    - réalisé lors du codage de ces coefficients



Filtre FIR

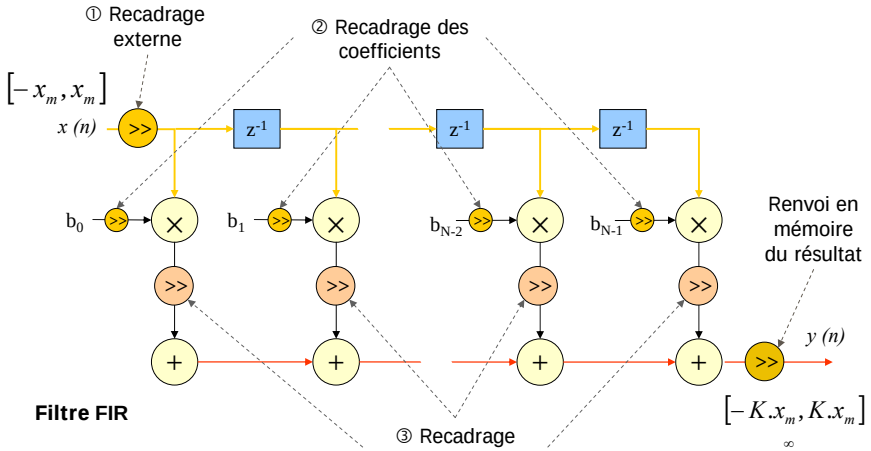
- Détermination de la position de la virgule pour les données et les opérations
- Insertion des opérations de recadrage pour aligner la position de la virgule



Graphe flot de données d'une cellule du FIR

IRISA

Recadrage des données dans un FIR



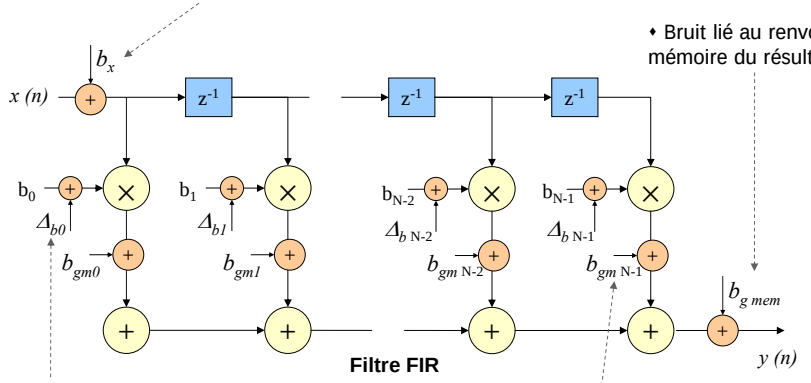
$$[-K.x_m, K.x_m]$$

$$K = \sum_{m=-\infty}^{\infty} |b_m|$$

$$k = \lceil \log_2(K) \rceil$$

Sources de bruit dans un FIR

- Bruit de quantification associé à l'entrée
- Bruit lié au recadrage externe ①

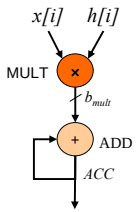


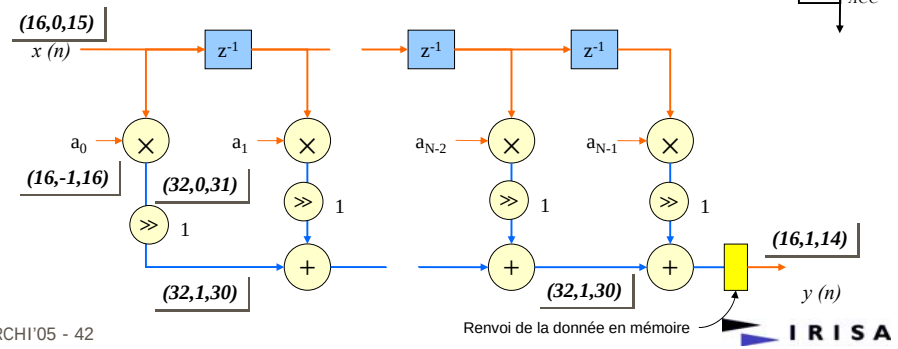
- Bruit lié au renvoi en mémoire du résultat
- Biais lié au codage des coefficients ②
- Bruit lié au recadrage interne ③

IRISA

Filtre FIR

- Architecture du processeur cible
  - Données en mémoire sur 16 bits
  - Multiplication 16 bits × 16 bits ⇨ 32 bits
  - Addition 32 bits + 32 bits ⇨ 32 bits





IRISA

Filtre FIR : code C virgule fixe

```
int x[M] = {-809, -6180, -1570, ...}        /* Signal x, codage (16,0,15)*/
int h[N] = {-1933, 2386, 3631,};            /* Coefficients (16,-1,16)  */

int main() {
 int x[N]; y[M];
 long  acc;
 int i,j;

  for(i=0; i<N; i++){x[i] = 0;}    /* Initialisation des variables internes du filtre */

  for(j=0;  j<M; j++) {            /* Filtrage du vecteur d'entree input */
    x[0] = Input[j];
    acc = (long)(x[0]*h[0])>>1;

    for(i=N-1; i>0; i--) {
       acc = acc + ((long)(x[i]*h[i])>>1); /* Calcul d'une cellule du filtre */
        x[i] = x[i-1];                     /* Vieillissement des variables internes */
    }
         y[j] = (int)(acc>>16);
  }
}
```

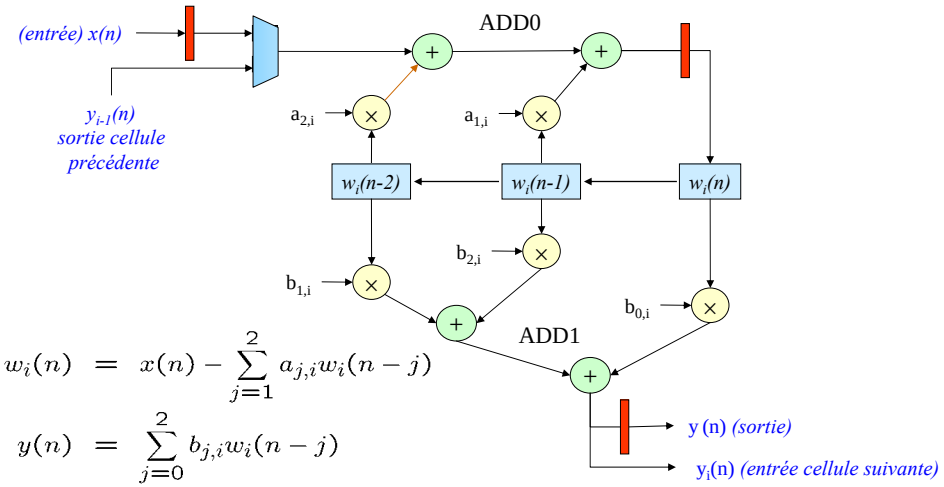
Le signal d'entrée et les coefficients sont spécifiés au niveau du code C en entiers sur 16 bits (absence de type virgule fixe en C) :
- l'entier représentant x (16,0,15) est obtenu en multipliant x par 2^{15}
- l'entier représentant h (16,-1,16) est obtenu en multipliant h par 2^{16}

Recadrage de la sortie de la multiplication : changement de format : (32,0,31) ⇨ (32,1,30)

Réduction de la largeur de la variable 32 bits ⇨ 16 bits
Récupération des 16 bits les plus significatifs de la donnée *(l 'opération de cast sur acc permet de récupérer les bits de poids faible uniquement)*

IRISA

Filtre IIR d'ordre 2 *(cellule i)*



$$w_i(n) = x(n) - \sum_{j=1}^{2} a_{j,i} w_i(n-j)$$

$$y(n) = \sum_{j=0}^{2} b_{j,i} w_i(n-j)$$

IRISA

Dynamique des données (IIR)

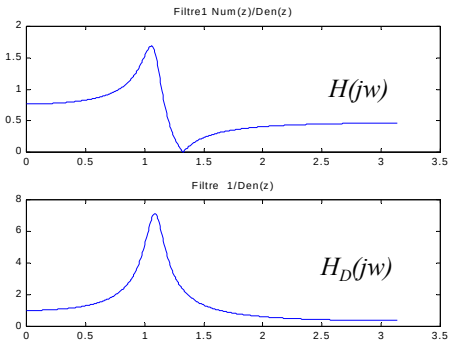
- Exemple

$$x \in [-1,1]$$

$$H(z) = \frac{0.5 - 0.2428 z^{-1} + 0.5}{1 - 0.85 z^{-1} + 0.8417 z^{-2}}$$

$$x_{\max} \sum_{m=-\infty}^{\infty} |h(m)| = 2.5645 \longrightarrow m_y = 2$$

$$x_{\max} \sum_{m=-\infty}^{\infty} |h_D(m)| = 9 \longrightarrow m_w = 4 \rightarrow k_1 = 4$$



Fonction de transfert des filtres H(z) et H_D(z)

Codage des coefficients

```
a1 = -27951
a2 =  27581
b2 =  16384
b0 =  16384
b1 = -7956
```

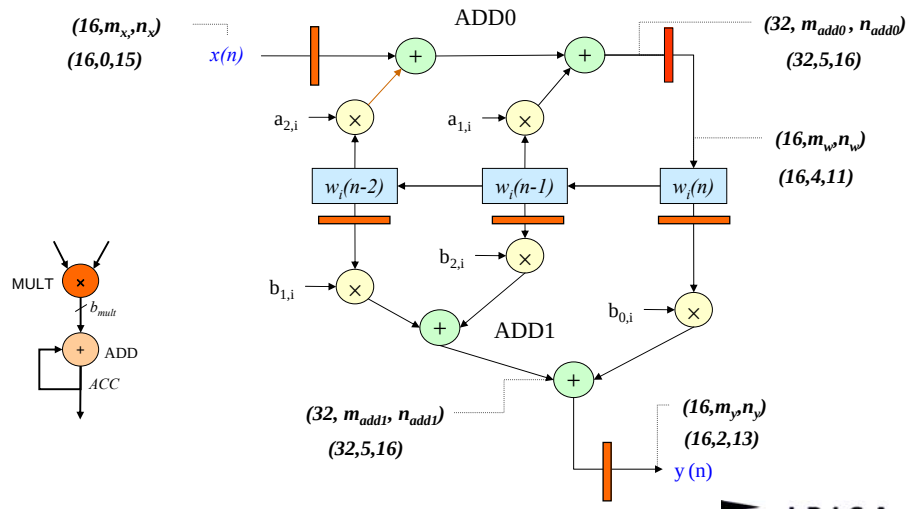

IRISA

Position de la virgule (IIR)

$$m_x = \lceil \log_2 (\max_n (|x(n)|)) \rceil = 0$$
$$m_{w_i} = \lceil \log_2 (\max_n (|w_i(n)|)) \rceil = 4$$
$$m_{y_i} = \lceil \log_2 (\max_n (|y_i|(n))) \rceil = 2$$
$$m_{a_{ij}} = \lceil \log_2 (\max_j (|a_i|)) \rceil = 0$$
$$m_{b_{ij}} = \lceil \log_2 (\max_j (|b_{ij}|)) \rceil = 0$$
$$m_{ADD0} = MAX(m_x, m_{a_i} + m_{w_i} + 1, m_{w_i}) = 5$$
$$m_{ADD1} = MAX(m_{b_i} + m_{w_i} + 1, m_{y_i}) = 5$$

IRISA

Codage des données (IIR)



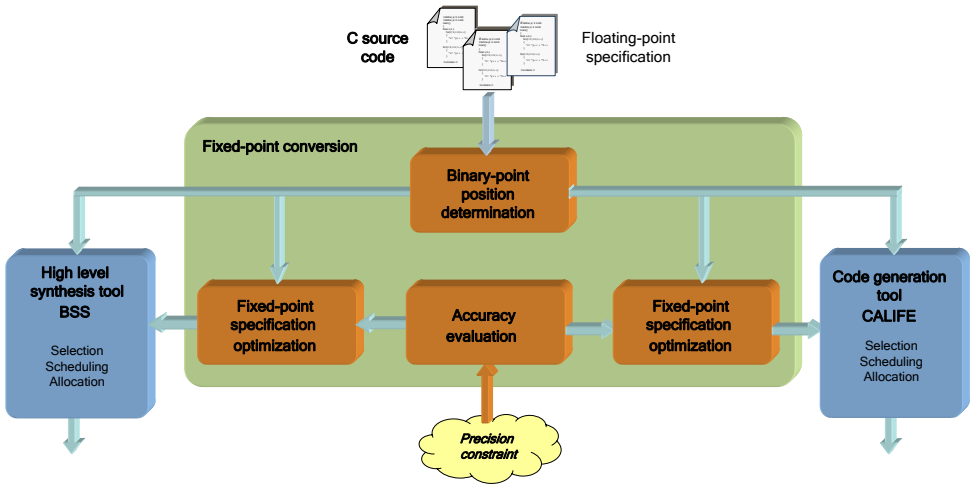
IRISA

Conclusion

- Différentes étapes du flot de codage
  - Détermination des fonctions de transfert intermédiaires
  - Calcul de leur réponse impulsionnelle
  - Détermination du domaine de définition des données
  - Analyse du graphe flot de données du programme
  - Détermination du système d'équations régissant le format des données
  - Résolution du système d'équations
  - Obtention des recadrages et du format des coefficients

IRISA

Conversion flottant-fixe (IRISA)



[Ménard02, Ménard04]

IRISA