



**IMT Atlantique**

Bretagne-Pays de la Loire

École Mines-Télécom

# ECG Filter

Davi SPERANDIO AGATTI

Marouen AIDOU DI

# SOMMAIRE

1. L'unité opérative
2. L'unité de contrôle



**IMT Atlantique**  
Bretagne-Pays de la Loire  
École Mines-Télécom

# L'unité opérative



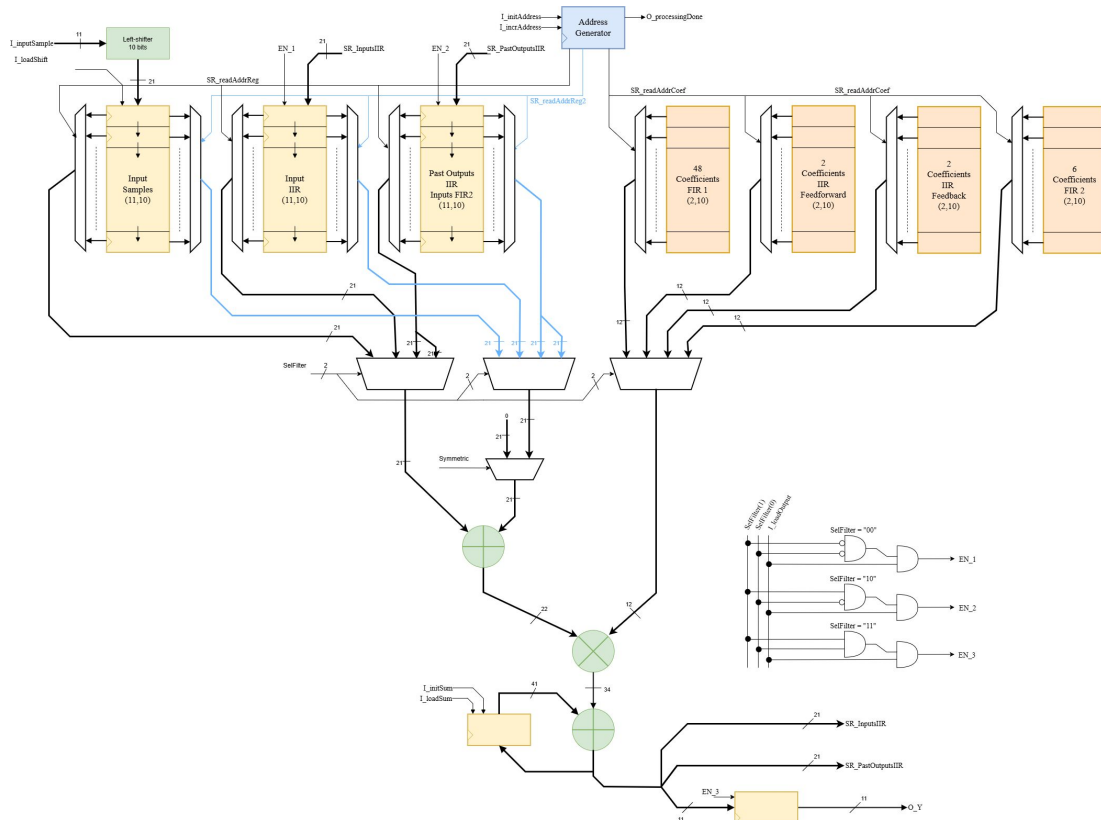
**IMT Atlantique**  
Bretagne-Pays de la Loire  
École Mines-Télécom

# L'unité opérative

## 1.1 L'architecture

4

- L'architecture développée a été conçue pour optimiser le traitement numérique d'un signal EMG. Trois filtres ont été implémentés pour filtrer ce signal, à savoir :
  - Un filtre FIR d'ordre 128 pour filtrer les fréquences inférieures à 5 Hz ;
  - Un filtre Notch de Pei-Tseng (IIR) à 50 Hz ;
  - Un filtre de Parks-McClellan pour filtrer les fréquences supérieures à 50 Hz.



# L'unité opérative

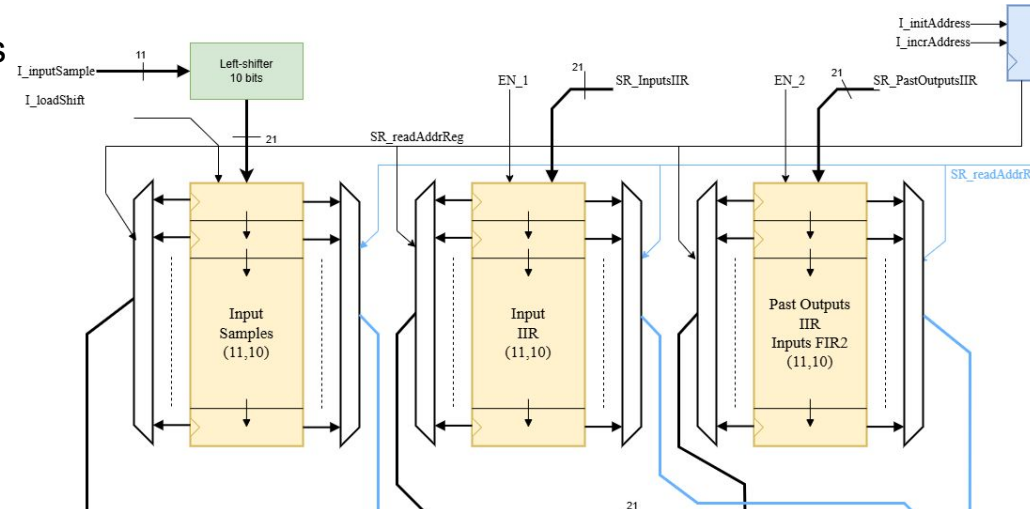
## 1.1 L'architecture

5

Le système a été conçu pour fonctionner avec 11 bits en entrée et 11 bits en sortie, dont 1 bit pour le signe et 10 bits pour représenter des valeurs entières. Il a été décidé de travailler avec une précision en virgule flottante de 10 bits (ce choix provenant des orientations initiales du projet).

L'architecture est composée de trois banques de registres, toutes fonctionnant sur 21 bits – 11 pour l'entrée et 10 pour la virgule flottante – afin d'accumuler les échantillons tout au long du processus de filtrage :

- Une banque qui stocke les valeurs d'entrée, utilisées par le premier filtre ;



- Une banque qui stocke la valeur de sortie après le traitement par le premier filtre. Cette banque est utilisée pour l'étape feedforward du filtre IIR ;
- Une banque qui stocke les valeurs de sortie du feedforward. Cette banque est utilisée à la fois pour les étapes de feedback du IIR et pour le traitement par le troisième filtre.

The diagram illustrates the FIR2 architecture, which consists of three main processing blocks: 'Input Samples (11,10)', 'Input IIR (11,10)', and 'Past Outputs IIR Inputs FIR2 (11,10)'. A 'Left-shifter 10 bits' block is connected to the 'Input Samples' block. The architecture is controlled by 'SR\_readAddrReg' (21 bits) and 'SR\_readAddr' (21 bits). It also features 'EN\_1' and 'EN\_2' control signals, and 'SR\_InputsIIR' and 'SR\_PastOutputsIIR' data paths. The output is 'I\_incrAddress'.

# L'unité opérative

## 1.2 Les optimizations

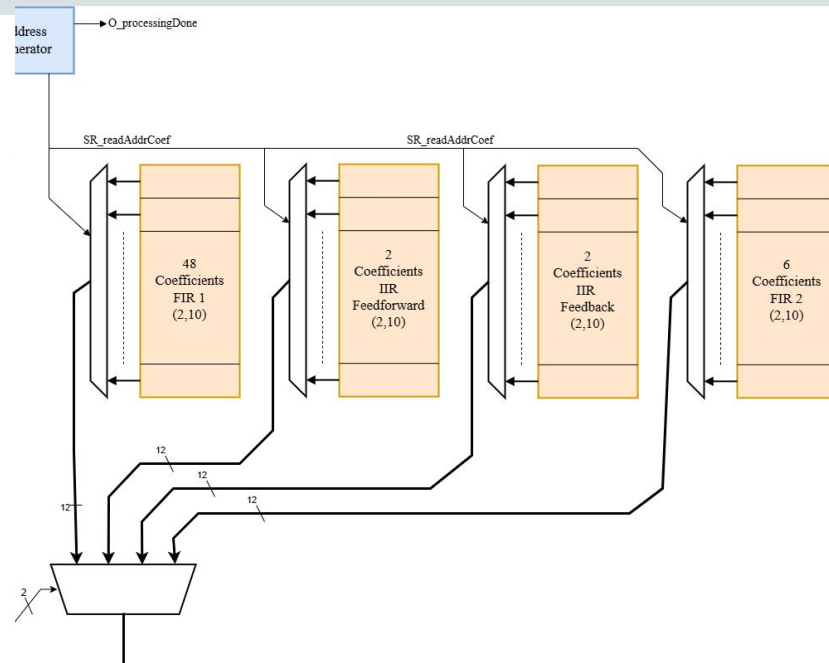
7

Il existe quatre banques de registres pour stocker les coefficients de chacun des filtres : une pour le premier filtre FIR, deux pour le filtre IIR, et une pour le second filtre FIR. Une précision de 10 bits a été utilisée pour les valeurs en virgule flottante, avec 1 bit supplémentaire pour le signe et 1 bit pour représenter la partie entière.

En observant certaines caractéristiques des coefficients de ces filtres, il a été possible d'appliquer quelques optimisations :

- Le premier FIR possède 129 coefficients. Cependant, comme on utilise 10 bits pour la virgule flottante, après ajustement de la quantification, il a été constaté que 34 d'entre eux devenaient nuls. Ainsi, il a été possible de réduire la taille de la banque de registres de 129 à 95 coefficients, optimisant ainsi l'espace de stockage.

- Il a également été observé que les filtres FIR utilisés, ainsi que l'étape feedforward du filtre IIR, possédaient des coefficients symétriques et en nombre impair. Par conséquent, il a été possible de réduire la taille des trois banques de registres de coefficients à  $(N + 1)/2$ , où N représente le nombre original de coefficients.



# L'unité opérative

## 1.2 Les optimizations

8

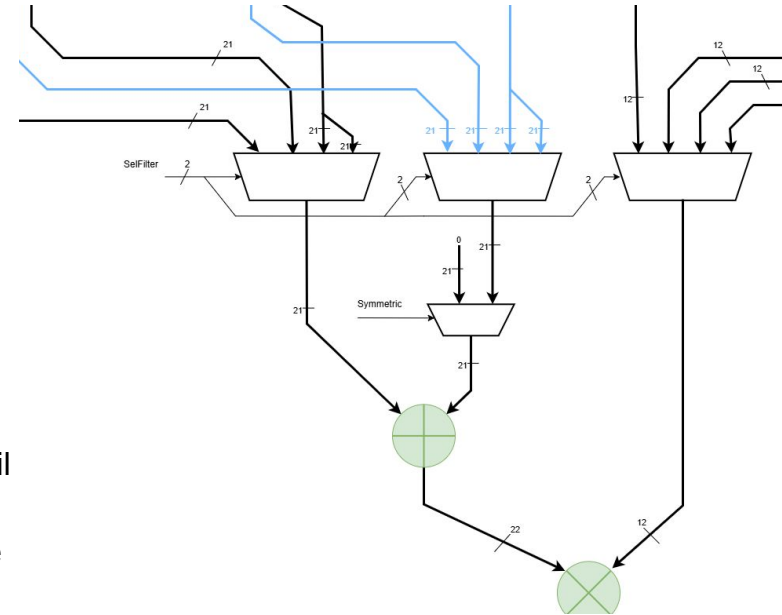
Pour les filtres symétriques:

$$y[n] = h[0].x[n] + h[1].x[n-1] + \dots + h[n-1].x[1] + h[n].x[0]$$

$$y[n] = h[0].x[n] + h[1].x[n-1] + \dots + h[1].x[1] + h[0].x[0]$$

$$y[n] = h[0].(x[n] + x[0]) + h[1].(x[n-1] + x[1]) + \dots + h[(n+1)/2].x[(n+1)/2]$$

On remarque qu'il est possible de réduire le nombre de multiplications effectuées (opérations plus coûteuses à exécuter) en factorisant les échantillons associés aux mêmes coefficients. La structure ci-contre a été implémentée précisément dans ce but. Si le filtre n'est pas symétrique (ou s'il s'agit simplement de la dernière valeur dans le cas des filtres de longueur impaire), l'échantillon est additionné à zéro puis multiplié normalement par le coefficient correspondant. Comme nous additionnons deux valeurs de 21 bits, il est nécessaire que le résultat de cette somme soit sur 22 bits.



- Chaque banque de coefficients est associée à deux multiplexeurs permettant d'accéder à des adresses différentes : l'un commence à la première adresse et incrémente (lignes noires sur la branche gauche de l'image), et l'autre commence à la dernière adresse et décrémente (lignes bleues).

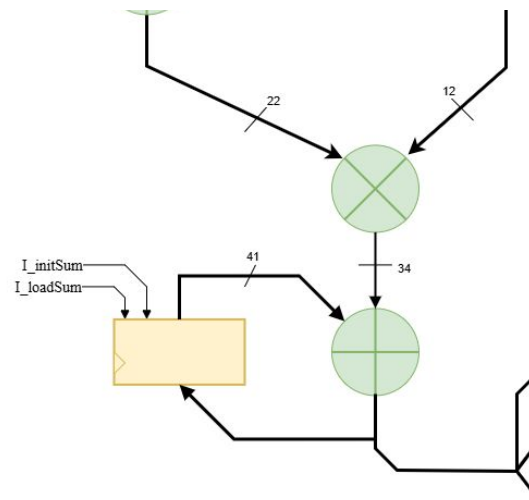


# L'unité opérative

## 1.1 L'architecture

Pour la structure du multiplicateur-accumulateur :

- Le multiplicateur multiplie toujours un échantillon, venant de la gauche, par un coefficient. Le choix des valeurs à utiliser dépend des signaux **SelfFilter** et **Symmetric**. Sa sortie est sur 34 bits ( $22 + 12$ ).
- L'accumulateur additionne les valeurs si **LoadSum** est actif, et il est remis à zéro si **initSum** est actif. Comme le premier filtre doit effectuer 95 opérations, il nécessite 7 bits supplémentaires, soit un total de 41 bits.



# L'unité opérative

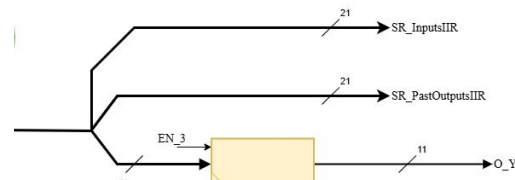
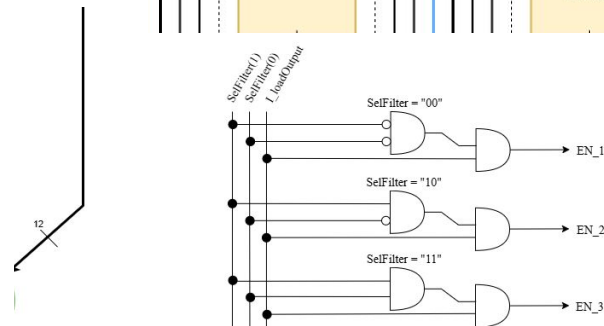
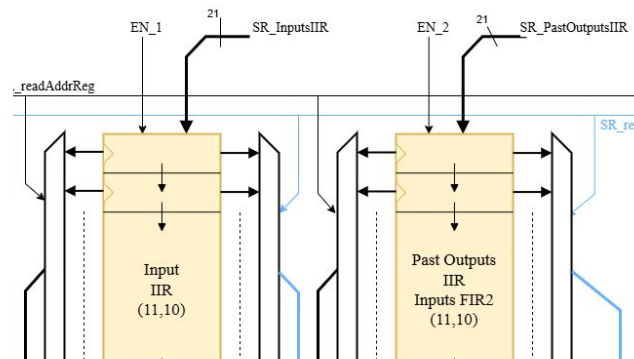
## 1.1 L'architecture

10

La valeur obtenue après le traitement par le premier filtre est enregistrée, sur 21 bits (bits 30 à 10 du registre accumulateur), dans la banque de registres **Input IIR**, tandis que la valeur obtenue après le traitement du second filtre (parties *feedback* et *feedforward*) est enregistrée dans la banque de registres **Past Outputs IIR / Inputs FIR 2**.

La sortie finale est obtenue, sur 11 bits (bits 30 à 20), après le traitement par le troisième filtre.

Comme les valeurs des échantillons et des coefficients possèdent toutes deux une précision de 10 bits pour la représentation en virgule flottante, les 20 bits les moins significatifs du registre accumulateur correspondent donc à la partie en virgule flottante.



- **Entrée**

- **Clock:** L'horloge du système.
- **Reset:** Signal activé manuellement pour réinitialiser toutes les variables/structures utilisées dans l'architecture.
- **InputSample:** Valeur de l'échantillon d'entrée sur 8 bits.
- **Symmetric:** Indique si le filtre est symétrique ou non. S'il l'est, on additionne avec la valeur symétrique positionnelle dans la banque de registres ; sinon, on utilise uniquement la valeur elle-même pour la multiplication.
- **LoadShift:** Responsable de sauvegarder la valeur de l'échantillon décalé dans le premier registre de la banque de registres, tout en décalant toutes les autres valeurs déjà présentes dans cette banque.
- **SelfFilter:** Sélectionne quel ensemble d'opérandes sera utilisé dans le multiplicateur-accumulateur. Définit également quelle est l'adresse maximale que le générateur d'adresses devra atteindre dans les banques de registres. Ce nombre dépend de l'ordre du filtre, comme expliqué dans le transparent .....
  - SelfFilter = "00" → Échantillons d'entrée et coefficients du filtre FIR 1, maxAdress = 47, lastAdress = 94 ;
  - SelfFilter = "01" → Sortie du FIR 1 et coefficients *feedforward*, maxAdress = 1, lastAdress = 2 ;
  - SelfFilter = "10" → Sorties passées du filtre IIR et coefficients *feedback*, maxAdress = 1, lastAdress = 1 ;
  - SelfFilter = "11" → Sorties passées du filtre IIR et coefficients du filtre FIR 2, maxAdress = 5, lastAdress = 10 ;

- **Entrée**

- **InitAdress:** Réinitialise le compteur d'adresse.
- **IncAdress:** Incrémente l'adresse à chaque signal d'horloge.
- **InitSum:** Réinitialise la valeur de l'accumulateur.
- **LoadSum:** Ajoute la valeur présente à la sortie du multiplicateur à la valeur stockée dans le registre de l'accumulateur.
- **LoadY:** Charge la valeur actuelle (sur 8 bits) présente dans le registre d'accumulation de la somme vers le registre de sortie, générant ainsi une valeur de sortie pour chaque échantillon traité. Il est également utilisé pour sauvegarder cette même valeur (sur 18 bits, en prenant en compte les 10 bits de la partie en virgule flottante), tout en décalant l'ensemble de la banque de registres des sorties passées.

- **Sortie**

- **ProcesingDone:** Signal émis par l'unité opérative indiquant que le compteur d'adresses a atteint la dernière adresse (maxAdress).
- **Y:** Sortie après le traitement de l'échantillon d'entrée.

# L'unité opérative

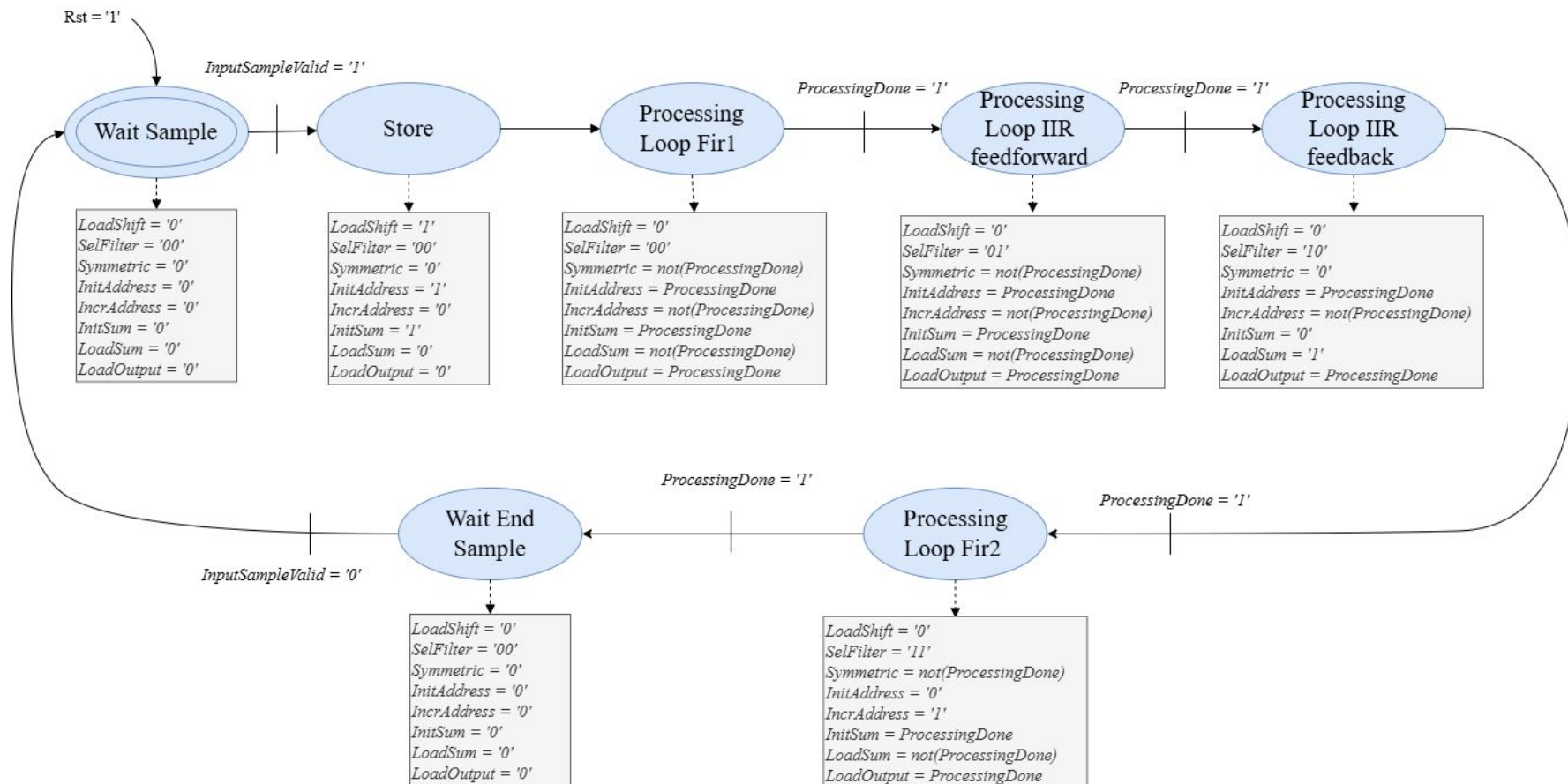


**IMT Atlantique**  
Bretagne-Pays de la Loire  
École Mines-Télécom

# L'unité de contrôle

## 2.1 La machine d'état

14



- **Wait Sample** : État d'attente d'un nouvel échantillon.
- **Store** : État qui enregistre le nouvel échantillon dans la banque de registres d'échantillons, tout en décalant les valeurs déjà présentes.
- **Processing Loop FIR 1** : Multiplie et additionne toutes les valeurs contenues dans la première banque de registres avec les coefficients du filtre FIR 1.
- **Processing Loop IIR feedforward** : Multiplie et additionne toutes les valeurs contenues dans la deuxième banque de registres avec les coefficients *feedforward*.
- **Processing Loop IIR feedback** : Multiplie et additionne toutes les valeurs contenues dans la troisième banque de registres avec les coefficients *feedback*.
- **Processing Loop FIR 2** : Multiplie et additionne toutes les valeurs contenues dans la troisième banque de registres avec les coefficients du filtre FIR 2. À la fin du traitement par le filtre, la valeur de sortie de l'unité opérative devient disponible
- **Wait End Sample** : État d'attente de fin d'échantillonnage.

- **Entrée**

- ***InputSampleValid*** : Signal d'entrée indiquant que l'échantillon d'entrée peut être lu et sauvegardé.
- ***ProcesingDone***: Déjà décrit.

- **Sortie**

- ***LoadShift, SelfFilter, Symmetric, InitAdress, IncAdress, InitSum, LoadSum, LoadOutput*** : Déjà décrit.



# MERCI



**IMT Atlantique**  
Bretagne-Pays de la Loire  
École Mines-Télécom