# Une architecture programmable de traitement des impulsions zéro-temps mort pour l'instrumentation nucléaire

Yoann Moline

ÉCOLE DOCTORALE
PRES I Bourgogne I Franche-Comté
Environnements - Santé

UNIVERSITÉ DE BOURGOGNE

**UNIVERSITE DE BOURGOGNE**

Ecole doctorale E2S

**THESE DE DOCTORAT**

par

**Yoann Moline**

pour obtenir le grade de

**Docteur de l'Université de Bourgogne**

Mention : Informatique

# Une architecture programmable de traitement des impulsions zéro-temps mort pour l'instrumentation nucléaire

**Thèse soutenue le 16 décembre 2015**

devant le jury composé de :

**M. Gwenolé CORRE**, Docteur, CEA Saclay / Encadrant

**M. Philippe COUSSY**, Professeur, Université de Bretagne Sud / Président du jury

**M. Bertrand GRANADO**, Professeur, Université Paris 6 / Rapporteur

**M. Paolo MUTTI**, Docteur, Institut Laue-Langevin / Examinateur

**M. Michel PAINDAVOINE**, Professeur, Université de Bourgogne / Directeur de thèse

**M. Mathieu THEVENIN**, Docteur, CEA Saclay / Co-Encadrant

**M. Marc WINTER**, Directeur de recherches, Institut Pluridisciplinaire Hubert Curien / Rapporteur

*À Dany et Marthe*

*« All fixed set patterns are incapable of adaptability or pliability. The truth is outside of all fixed patterns. »*

- Lee Jun Fan

# Remerciements

Je ne peux commencer ces remerciements sans exprimer ma profonde gratitude envers l'ensemble des membres de mon jury de thèse. Je remercie Bertrand Granado ainsi que Marc Winter pour avoir acceptés d'évaluer ce travail en qualité de rapporteurs ainsi que Paolo Mutti d'avoir accepté de juger ce travail et pour l'intérêt qu'il a montré envers celui-ci et enfin Philippe Coussy pour avoir accepté de présider mon jury de thèse et plus personnellement pour m'avoir donné goût à l'électronique numérique il y a quelques années de cela.

Je me permets de remercier encore une fois Philippe Coussy mais aussi Michael Hübner pour avoir suivi mes travaux au fil de ces trois années en tant que comité de thèse. L'ensemble de leurs conseils et commentaires sont en grande partie responsables de l'avancement de ces travaux.

Je tiens à manifester toute ma reconnaissance à mon directeur de thèse, Michel Paindavoine, d'avoir accepté ma candidature pour la réalisation de ces travaux. Je lui suis également reconnaissant pour le temps qu'il m'a accordé, ses qualités pédagogiques et scientifiques ainsi que sa franchise et sa sympathie.

J'adresse de chaleureux remerciements à mon encadrant de thèse, Mathieu Thevenin. N'ayant pas un caractère facile, je ne peux que m'incliner devant la manière avec laquelle il a su adapter son approche pédagogique face à « mon cas ». En effet, au bout de trois années de débats et séances de *brainstorming* aussi bien scientifiques que philosophiques (sur lesquels je dois admettre avoir eu souvent tort), j'ai finalement pu observer chez moi l'évolution apportée par ses enseignements tant sur le plan scientifique que personnel. Pour tout cela, merci encore.

Enfin, j'associe des remerciements particuliers à mon co-encadrant de thèse Gwenolé Corre qui, par son attitude et son esprit réfléchi-et posé, a évité à plusieurs reprises que je n'implose, jouant ainsi son rôle de co-encadrant à la perfection tant la balance méritait parfois d'être équilibrée sur le plan caractériel. De plus, son rôle de soutien (voire de béquille dans certains jours) et le fait qu'il m'ait ouvert la porte de l'instrumentation nucléaire font que je lui suis encore une fois extrêmement reconnaissant.

J'exprime également ma reconnaissance envers les trois chefs successifs du Laboratoire Capteurs et Architectures Électroniques (LCAE), Stéphane Normand, Medhi Gmar et Alexandre Bounouh pour leurs conseils avisés ainsi que leur disponibilité.

Mes remerciements vont également à Nathalie Feiguel, Florence Chedaute et Caroline Gauthier qui portent l'attention de mères avec nous et nos « soucis » administratifs, mais le font avec patience et sourire.

Les mots me manquent en parlant de mon « co-bureau », Thomas Peyret. Il a toujours été à mes côtés, toujours disponible pour m'écouter, trouvant les mots qu'il faut quand ça n'allait pas tout en m'accordant toute l'aide possible. Nos discussions sur ce qui est important dans la vie, pour la science et éventuellement pour dominer le monde ont participé au bon déroulement de ces travaux tant elles m'ont apporté du recul sur moi-même et donc sur le fruit de ceux-ci. Malgré des tempéraments opposés, c'est celui de nous deux qui a fait le plus d'efforts pour s'adapter à l'autre, notamment lorsque j'accusais l'alignement des planètes pour une défaite à Magic (même si cela

arrivait peu souvent), et que, sans laisser paraître un seul soupir, il me disait que je n'avais pas eu de chance. Merci encore.

Je tiens également à remercier Vincent Schoepff d'avoir été l'oreille attentive qu'il me fallait lors de (trop) nombreuses pauses cigarettes (on y revient toujours). A chaque défaite, lorsque je voulais tout plaquer, à chaque victoire, lorsque je pensais avoir révolutionné l'instrumentation nucléaire, il m'a écouté et supporté. Finalement, grâce à son franc-parler il a à chaque fois su me ramener à la raison. Pour toutes ces raisons, merci.

J'adresse également des remerciements particuliers à celui qui a joué le rôle du petit frère durant ces trois années, le bien nommé Jordan « Kevin » Gameiro, qui, malgré son jeune âge est rempli de sagesse. Jordan ne dit jamais non, sera toujours là en cas de problème et accepte même de faire des détours de plusieurs dizaines de kilomètres sur un caprice. Le tout, emballé dans une belle carrosserie rouge. Que demande le peuple ?

Bien sûr, atteindre ces objectifs n'aurait pas été possible sans l'ensemble des membres du personnel du LCAE à qui je tiens également à exprimer ma sincère reconnaissance pour leur disponibilité et tout ce que j'ai pu apprendre d'eux et de leurs compétences respectives. Matthieu H., Romain, Hassen, Karim, Frédéric C., Frédéric L., Mounir, Romuald, Adrien, Vladimir, Jean-Michel, Mathieu T., Vesna, Amélie, Christelle et Hamid, à tous un grand merci. Je remercie également ceux qui qui n'étaient plus là lors de mon départ tels que Guillaume, Fabien, Marion et Jennifer avec une attention particulière à Licinio qui a grandement participé à mon devenir de « thésard ».

Je remercie également mes partenaires d'affliction de tous les instants pour ainsi dire, les autres doctorants que j'ai eu la chance de côtoyer, mes aînés qui ont eu la chance de me bizuter, et mes cadets que j'ai tenté de bizuter à mon tour. Merci à Maugan, Pauline, Hermine, Emmanuel, Jonathan, Eva, Isabelle, Camille, Héléna et Thomas B.

Je désire aussi remercier Christophe Bobin et Cheick Thiam du *Laboratoire National Henri Becquerel (LNHB) pour leur contribution et la richesse des informations et des données qu'ils m'ont transmis* ainsi que Mathieu A., Thierry, Juan, Bénédicte du Laboratoire de Modélisation et Simulation des Systèmes (LM2S) que j'ai eu longtemps l'occasion de côtoyer au 516 notamment lors de très studieuses pauses café / mots fléchés.

Pour finir, je remercie mon stagiaire Jeremy Gosset pour sa participation à ces travaux et qui, je l'espère, a pu tirer des bénéfices de son stage sous mon encadrement et celui de Mathieu Thevenin.

Je ne saurai oublier d'adresser un grand merci, à ma famille, en particulier Bernard, Martine, Sébastien et Monique, ainsi qu'à ma compagne Solenn et ses parents Jean-Charles et Yveline pour leur soutien moral, leurs conseils, leurs encouragements et pour tous les efforts et les sacrifices qu'ils n'ont cessé de faire pendant toute la période de réalisation de cette thèse.

# Sommaire

# Liste des figures

# Liste des tableaux

# Introduction

## Contexte et problématiques

Le domaine de l'instrumentation nucléaire couvre un large éventail d'applications allant du contrôle-commande de cœurs de réacteur, la métrologie, la sécurité (portiques de détection dans les ports, aéroports et aux frontières), la lutte contre la non-prolifération en passant par le médical (scintigraphie, tomographie etc.). Pour chacune de ces applications il existe de nombreux détecteurs et méthodes spécifiques pour lesquels sont développées et redéveloppées des chaînes de mesures dédiées ce qui est coûteux. Une chaîne de mesure de la radioactivité se divise en deux parties, comme illustré en Figure 1. La première sert à convertir l'énergie issue d'un radio-émetteur et déposée dans le détecteur en un signal électrique utilisable ; ce signal est ensuite mis en forme avant d'être numérisé. La deuxième partie concerne l'ensemble des traitements numériques qui permettent d'obtenir de ce signal des données exploitables.

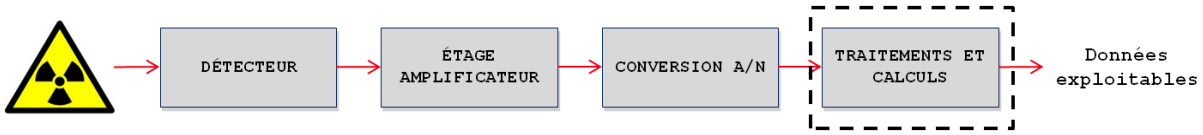



**Figure 1 : Chaîne de mesure de radioactivité traditionnelle.**

De nouveaux algorithmes sont constamment proposés grâce aux avancées dans le domaine du traitement numérique du signal. Cependant, ils ne sont pas encore mis en œuvre dans les systèmes de mesure actuels qui doivent faire face à deux problématiques majeures. La première est la caractéristique du signal lui-même, qui est constitué d'impulsions dont les amplitudes, les durées, et les dates d'arrivée sont aléatoires puisqu'il est admis qu'elles sont issues d'un processus poissonnien homogène. Si ces impulsions sont de durées variables et que la possibilité existe qu'une impulsion arrive directement après la fin de la première alors les traitements doivent être capables de travailler en flux continu sur les échantillons. Dans le cas contraire la solution traditionnelle implique de paralyser les chaînes de mesure après l'arrivée d'une impulsion. Dans ce cas de figure, aucune nouvelle impulsion ne peut être traitée par le système durant cette paralysie. Cette seconde problématique est appelée temps mort. Ce phénomène est dû soit aux limitations de l'électronique de traitement du signal en termes de performance de calcul soit par le traitement lui-même. Les applications du domaine médical ou liées à la sécurité doivent limiter le temps mort afin d'exploiter un maximum d'informations contenues dans le signal. Les chaînes de mesures utilisent actuellement des composants reconfigurables de type FPGA (Field Programmable Gate Array) qui leur permettent de travailler en flux sur les échantillons. Cette solution est limitante pour deux raisons. La première est que le concepteur doit se contraindre à utiliser des algorithmes permettant naturellement de traiter les échantillons à la volée (filtres, discriminateur, calcul d'aire etc.). La seconde est d'ordre pratique puisque l'implémentation d'algorithmes dédiés sur les technologies reconfigurables est une tâche complexe et coûteuse en temps. C'est pourquoi une grande partie des algorithmes sont exécutés hors ligne après l'acquisition des données.

Une architecture numérique et temps-réel de traitement des impulsions pouvant être programmée dans un langage de haut niveau tel que le C ou le C++ permettrait d'apporter la flexibilité nécessaire pour supporter la variété d'algorithmes associés aux applications du domaine mais aussi pour accélérer la mise en œuvre et le prototypage des mesures en laboratoire par des non-électroniciens. Toutefois, les performances de solutions programmables actuelles et la manière dont il est possible de leur acheminer des données ne permettent pas un fonctionnement en flux et en temps-réel sans avoir à gérer la problématique de temps mort. Cette complexité augmente sensiblement avec le nombre de voies d'acquisition nécessaires à certaines applications. L'architecture doit donc être en mesure de passer à l'échelle et de garantir son fonctionnement aussi bien pour une seule voie de mesure que pour traiter des données provenant de centaines de voies.

L'objectif de cette thèse est donc d'offrir une architecture de traitement du signal couvrant l'ensemble des applications du domaine de l'instrumentation nucléaire tout en étant capable de répondre au besoin de programmabilité, de multivoie et capable d'être zéro-temps mort sans avoir à dimensionner une architecture de calcul au pire cas, ce qui se révèlerait coûteux.

## Démarche

La démarche de cette étude est organisée comme suit :

Le Chapitre 1 détaille plus particulièrement le contexte et les problématiques de cette thèse afin d'analyser les limitations intrinsèques d'une chaîne d'instrumentation nucléaire qui obligent les concepteurs à réaliser des architectures dédiées. Cette étude passe par la caractérisation des besoins en capacité de calcul d'une architecture multi-applicative zéro-temps mort et donc par l'analyse des applications, des algorithmes et des détecteurs. Ce chapitre est complété par l'état de l'art des architectures du domaine les plus à même de répondre aux problématiques posées.

Le Chapitre 2 présente le modèle d'exécution d'une architecture générique pour les applications de l'instrumentation nucléaire. Ce modèle, montre analytiquement qu'il est possible de répondre aux contraintes de temps mort tout en étant programmable et multivoie. Il se base sur l'hypothèse qu'une séparation et une extraction précise des impulsions du reste du signal est possible.

Le Chapitre 3 présente un extracteur d'impulsions que nous proposons et le compare aux autres discriminateurs de la littérature. Il détaille comment cet extracteur est capable de s'adapter dynamiquement aux différentes tailles et forme d'impulsions afin d'être utilisé avec n'importe quel détecteur mais surtout pour n'importe quelle application. Ce dernier permet de tirer pleinement parti du modèle d'exécution présenté en 3.

Le Chapitre 4 présente l'architecture programmable permettant le traitement des impulsions extraites. Une proposition d'implémentation y est présentée et a permis la réalisation d'un simulateur ayant pour fonction la validation du modèle d'exécution. Finalement, une première preuve de concept basée sur le modèle d'exécution et fabriquée à partir de composant programmable du marché est ensuite présentée.

Enfin, le dernier chapitre conclut cette thèse et présente ses différentes perspectives.

# Chapitre 1 : Analyse de la problématique & état de l'art

Ce chapitre définit le contexte et les problématiques dans lesquels s'inscrit cette thèse. Dans un premier temps, les concepts de base de l'instrumentation nucléaire sont présentés. Le signal à traiter est décrit et les chaînes de mesures classiques sont introduites au travers des différentes applications du domaine. Les problématiques inhérentes à la mesure en instrumentation nucléaire sont détaillées (1.1). La deuxième partie du chapitre présente la démarche qui a permis la caractérisation des besoins en capacité de calcul d'une architecture programmable pour atteindre le zéro-temps mort. Pour cela, les applications (1.2), les algorithmes (1.3) et les détecteurs (1.4) traditionnellement utilisés dans une chaîne de mesure sont analysés. Enfin, cette étude est mise en perspective par une analyse des composants programmables du marché, ce qui a permis de faire apparaître les limitations de ceux-ci dans le cadre des prérequis de l'instrumentation nucléaire (1.5). Un état de l'art des chaînes de traitement numérique des impulsions pour l'instrumentation nucléaire est analysé afin de mettre en évidence les solutions mises en œuvre pour faire face à ces limitations (1.6).

## 1.1 Concepts de l'instrumentation nucléaire

### 1.1.1 Mesure de la radioactivité

La **radioactivité**, de manière générale, est le plus souvent abordée sous l'angle de ses risques. Elle l'est plus rarement sous l'angle de ses nombreuses applications ou sous l'angle du phénomène physique lui-même. Il est donc utile, dans un premier temps, de comprendre le phénomène physique de la radioactivité, avant d'aborder ses multiples applications contemporaines. La radioactivité est un phénomène physique naturel au cours duquel un noyau atomique émet des particules ou ondes électromagnétiques que l'on appelle rayonnement. C'est dans le contexte de la mesure de ces rayonnements que cette thèse se situe.

La détection de rayonnement est principalement utilisée dans la recherche fondamentale, pour la connaissance de l'univers et de ses lois ; elle permet également de contrôler le pilotage des réacteurs nucléaires, de mesurer l'irradiation des personnes potentiellement exposées, de contribuer aux nouvelles techniques de médecine, de sécuriser et contrôler les transports et la conformité des procédés industriels, de lutter contre le terrorisme et la prolifération nucléaire. La détection de ces rayonnements est donc importante à la fois pour se protéger, surveiller et mettre à profit cette formidable source d'énergie qu'est l'atome. Il est possible de distinguer deux grands types de méthode pour la mesure des rayonnements ionisants et pour comprendre ses applications. Ces deux types de mesure sont dits « mesures actives et passives » (Lyoussi 2010).

#### 1.1.1.1 Mesures passives

Les mesures passives sont utilisées lorsque les rayonnements émis spontanément par l'élément radioactif ont une intensité et un parcours moyen dans la matière suffisants pour être détectés et donc permettre la caractérisation de l'émetteur et la catégorisation de l'objet mesuré. La mise en œuvre de ces mesures nécessite de disposer d'une chaîne de détection et d'acquisition spécifique au type de rayonnements émis. Les deux grandes applications de la mesure passive sont la

spectrométrie gamma et le comptage (neutronique ou gamma). Un exemple de cas applicatif est la mesure passive utilisée dans les portiques de détection de radioactivité. Ces portiques, installés à l'entrée des centres de traitement des déchets par exemple, comme illustré en Figure 1-1, ont principalement pour rôle d'alerter l'exploitant de la présence de radionucléides dans le chargement d'un véhicule afin d'assurer la sécurité de son personnel. L'objectif de ce contrôle est par exemple la détection de sources radioactives ponctuelles (par exemple des sources de Cobalt 60 ($^{60}$Co) ou de Césium 137 ($^{137}$Cs)) qui sont dangereuses du point de vue de la radioprotection du personnel.

**Figure 1-1 : Un portique de détection pour véhicules routiers.**

#### 1.1.1.2 Mesures actives

Si, *a contrario*, la mesure des rayonnements émis spontanément par l'objet ne permet pas de le caractériser, le recours aux techniques actives s'avère alors nécessaire. Cela peut dépendre soit de la nature du rayonnement émis, soit de son intensité mais aussi de la présence ou non de rayonnements parasites interférant avec les signaux utiles. Comme leur nom l'indique, ces techniques nécessitent l'utilisation de sources externes génératrices de particules. Ces particules sont alors dites interrogatrices. Un exemple de cas applicatif de la mesure active est la caractérisation de matériaux dont la composition est inconnue et qui ne contiennent que des éléments n'émettant pas spontanément des radiations. C'est également une méthode employée pour la caractérisation de déchets radioactifs, illustrée en Figure 1-2, afin de permettre la séparation des matières valorisables de celles qui ne le sont pas. Les principales applications de la mesure active sont la mesure neutronique active et la mesure par photofission induite.


**Figure 1-2 : Fûts de déchets radioactifs.**

### 1.1.2 Définition du signal

L'objectif de cette thèse est d'aboutir à une architecture permettant de couvrir un maximum de cas applicatifs. La chaîne de détection et de mesure du signal est identique pour les mesures non destructives passives et actives. Ce qui influence la chaîne de détection est le mode du signal.

#### 1.1.2.1 Principes de base de la détection

Le rayonnement est une forme d'énergie. Cette énergie peut se déposer en totalité ou en partie dans un milieu approprié et produire ainsi un effet appelé interaction rayonnement-matière. Les interactions ont lieu avec ce qu'on appelle un milieu détecteur. Ces interactions génèrent directement ou indirectement des charges électriques lesquelles, une fois collectées, sont (pré)amplifiées et converties en signaux électriques. Cette opération est rendue possible grâce à la polarisation électrique du détecteur conduisant à l'établissement d'un champ électrique responsable du mouvement des charges produites et de leur collection.

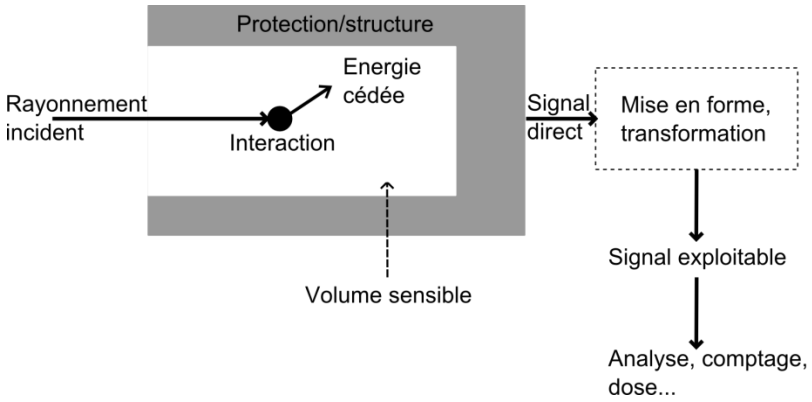



**Figure 1-3 : Représentation schématique d'une chaîne de détection de rayonnements.**

D'une manière générale, la détection et la mesure de rayonnements constituent un processus à plusieurs étapes comme le montre la Figure 1-3. Il s'agit, dans un premier temps, de faire interagir le rayonnement incident utile avec le milieu détecteur. Ces interactions sont ensuite converties en impulsions électriques qui sont traitées électroniquement et acheminées vers une unité d'acquisition et d'analyse. On obtient ainsi un premier résultat appelé grandeur brute ou grandeur mesurée. C'est cette donnée qui va ensuite parcourir une série de traitements afin d'être analysée pour accéder à ce qu'on appelle la grandeur recherchée. Toutes ces étapes permettent d'obtenir l'activité ou de caractériser la source au moyen de traitements appropriés prenant notamment en compte la sensibilité de détection, la distance source-détecteur, le bruit de fond etc... Dans la partie suivante, les différents modes de lecture possibles du signal sont présentés. Toutefois, quel que soit le mode de fonctionnement d'un détecteur et donc le principe sur lequel s'appuie la détection des rayonnements, il est constitué des mêmes éléments :

- un capteur au niveau duquel le rayonnement interagit avec la matière ;
- un système d'amplification qui met en forme et amplifie le signal produit par la sonde ;
- un système de traitement du signal (analogique et/ou numérique) ;
- un système d'affichage ou de post-analyse.

#### 1.1.2.2 Principes et techniques de la mesure de radioactivité

Selon l'intensité du flux (quantité de rayonnement), trois modes de fonctionnement sont considérés pour le traitement. Le mode le plus couramment utilisé est le mode impulsion (ou moins couramment appelé régime de comptage) à basse puissance. Cependant, dans cette partie nous nous intéresserons également au mode courant à haute puissance ainsi qu'au mode fluctuation dans une moindre mesure, car il est désormais utilisé sur de nouveaux réacteurs expérimentaux ainsi que sur certains réacteurs embarqués comme dans les sous-marins à propulsion nucléaire. La Figure 1-4 représente les signaux correspondant à ces différents régimes en fonction de l'augmentation du flux de particules détectées. La Figure 1-4 (C) montre les variations de courant correspondant aux interactions individuelles. Sur la Figure 1-4 (B), celles-ci se confondent et forment des empilements à tel point qu'il n'est plus possible de les distinguer. Enfin, la Figure 1-4 (A) présente un mode particulier de mesure spécifique à certains détecteurs gazeux qui permettent d'extraire de l'information de flux de manière indirecte.

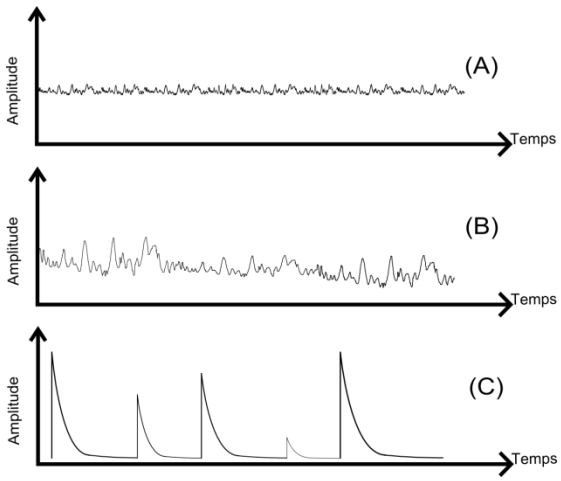



**Figure 1-4 : Modes du signal avec (A) le mode courant, (B) le mode fluctuation, (C) le mode impulsion.**

A présent, les fondamentaux des techniques de mesures qui y sont associées sont analysés afin de voir si ces éléments permettent d'envisager la création d'une architecture commune à tous les modes de fonctionnement et applications associées.

### 1.1.3 Principes et techniques de la mesure de radioactivité

#### 1.1.3.1 Mode courant et fluctuation

Le mode courant, présenté en Figure 1-4 (A), consiste en la mesure directe du courant mis en circulation dans le circuit détecteur. Le mode courant s'utilise pour une mesure moyenne de flux de particules. Lorsque le détecteur est exposé en permanence à un flux intense de rayonnement, ce dernier génère une succession continue de signaux qu'il n'est plus possible de mesurer individuellement et que l'on mesure alors sous forme d'un courant par une électronique appropriée. Le courant est donc composé de deux parties, une partie continue et une partie qui varie autour de cette première. L'unité peut être un débit de fluence exprimé en nombre $n$ de particules par seconde et par cm². L'information obtenue reste globale et ne renseigne pas sur les caractéristiques de chaque particule. Ce mode est principalement utilisé dans le contrôle commande des réacteurs pour mesurer le flux neutronique. La chaîne de mesure est constituée du détecteur polarisé par une alimentation haute tension et d'un dispositif, équivalent à un ampèremètre, permettant de mesurer la moyenne du courant. Le courant étant, dans ce mode, proportionnel au nombre de particules ayant interagi avec le

milieu détecteur. A l'aide du premier théorème de Campbell, on démontre cette relation de proportionnalité par la formule présentée dans l'équation 1-1 .

Avec :

$\bar{I}$ le courant continu

$K_i$ la sensibilité du détecteur en courant

$\phi$ le flux neutronique en n. $s^{-1}$. $cm^{-2}$

$$\bar{\bar{I}} = K_i \cdot \phi \qquad \textbf{1-1}$$

Il n'y a actuellement aucun enjeu (défini) à rendre programmable une chaîne de mesure travaillant en mode courant. Il n'est donc pas nécessaire de poursuivre dans le détail l'étude des chaînes de mesures qui s'appliquent à ce mode puisque la problématique se situe principalement sur la partie analogique de la chaîne.

Le mode fluctuation ou mode Campbell est finalement un mode particulier du mode courant qui utilise les fluctuations statistiques du courant pour remonter à l'intensité du rayonnement mesuré. Il trouve son utilité lorsque le taux de comptage est trop élevé pour tenter de discriminer les impulsions qui s'empilent. Le signal de sortie peut alors être assimilé à un bruit dont la variance est proportionnelle au flux. Ce sont donc les caractéristiques statistiques qui sont étudiées (moyenne et variance). La chaîne de mesure est principalement constituée du détecteur polarisé par une alimentation haute tension, d'un CAN, et d'un traitement numérique qui permet l'extraction des caractéristiques statistiques. D'une façon simplifiée, à l'aide du deuxième théorème de Campbell, et toujours dans le cadre applicatif du contrôle commande de réacteur, on démontre que la variance est proportionnelle au flux neutronique incident comme le montre l'équation 1-2.

Avec :

$var(\bar{I})$ la variance du courant continu moyen ;

$K_f$ la sensibilité du détecteur en courant ;

$\phi$ le flux neutronique en n. $s^{-1}$. $cm^{-2}$ ;

$Cst$ une constante liée à la réponde impulsionnelle du détecteur en $s^{-1}$ ;

$$\overline{\overline{var(\bar{I})}} = K_i \cdot Cst \cdot \phi \qquad \textbf{1-2}$$

Contrairement au mode courant, il a été démontré par (Fanet 2002) qu'il y a un réel intérêt au traitement numérique de ce signal car « *Les moments statistiques d'ordre 1 et 2 épuisent l'information que l'on peut extraire d'un signal gaussien puisque tous les autres moments s'en déduisent. Dans le cas des flux poissonniens généralement considérés dans les mesures, cette propriété ne s'applique pas et l'estimation des moments d'ordres supérieurs permet d'extraire une information supplémentaire. Quand plusieurs flux poissonniens sont simultanés, l'analyse des statistiques d'ordres supérieurs permet*

*alors d'estimer séparément chaque composante* ». Il s'avère donc que le traitement numérique est de manière générale plus adapté au calcul de variance mais il l'est encore plus lorsqu'il s'agit d'estimer des moments statistiques d'ordre supérieur à 2. Cette technique est déjà aboutie (Barbot et al. 2015; Thevenin et al. 2014) et ne définit pas d'enjeux à rendre programmable une chaîne de mesure travaillant en mode fluctuation.

#### 1.1.3.2 Mode impulsion

Le mode impulsion s'utilise pour la détection individuelle de particules. Les mesures en impulsions sont de très loin les plus répandues. Ce mode est donc l'objet principal des travaux de cette thèse. Les impulsions à traiter en électronique nucléaire ont une occurrence aléatoire suivant un processus de distribution de poisson homogène. Le phénomène de la radioactivité étant statistique, les impulsions ne se suivent donc pas avec une fréquence donnée. Les impulsions sont rapides (durée totale pouvant varier de la nanoseconde à plusieurs microsecondes) et peuvent dans le pire cas être très proches ; les systèmes électroniques utilisés doivent donc avoir une large bande passante. La forme de l'impulsion de tension couramment rencontrée est donnée sur la Figure 1-5.

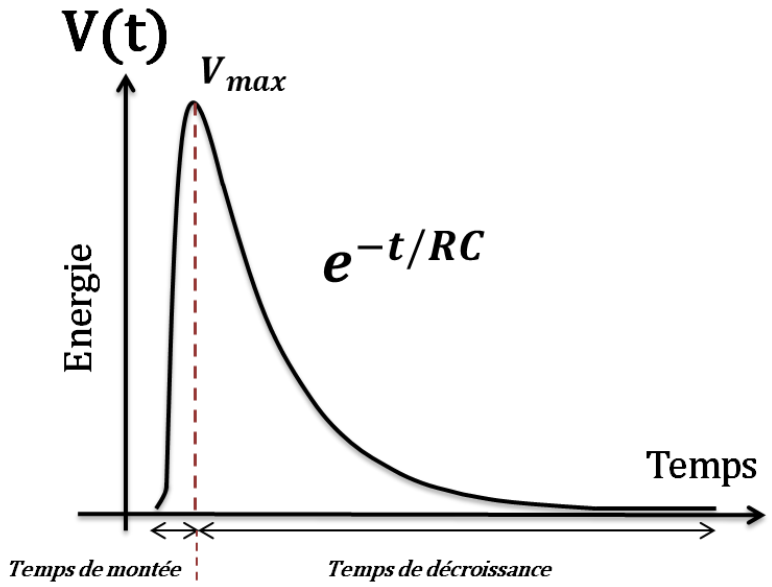



**Figure 1-5 : Impulsion de tension typiquement rencontrée dans les mesures de radioactivité.**

La charge d'une impulsion est proportionnelle à l'énergie perdue lors de l'interaction qui l'a générée. Les impulsions sont mesurées à l'aide des variations de tension du détecteur aux bornes d'une résistance associée à une capacité qui caractérise à la fois le détecteur lui-même et la chaîne électronique associée (câbles, préamplificateur, amplificateur, etc.). Si on modélise l'arrivée d'un signal de charge totale $Q$ et de durée $Tc$ en provenance du détecteur, deux scénarios aux limites sont possibles (Knoll 2010).

La constante de temps $RC$ est très petite devant $Tc$, le circuit électronique est suffisamment rapide pour transmettre l'impulsion initiale au facteur $R$ près. Ce mode de fonctionnement est donc très intéressant pour les mesures à haut débit. Il est cependant complexe à mettre en œuvre car la mesure de l'amplitude maximum de chaque impulsion est difficile et, de plus, la sensibilité du système

est trop faible pour détecter les fluctuations réelles des signaux. On ne peut donc pas, ou très difficilement, caractériser l'énergie dans ce mode.

La constante de temps $RC$ est grande devant $Tc$ ; la charge $Q$ s'écoule alors très lentement à travers la résistance $R$ et reste momentanément intégrée dans C, jusqu'à atteindre un maximum $Vmax = Q/C$. Une fois atteint, la charge s'évacue selon une fonction exponentielle décroissante. Le temps de montée est une caractéristique propre au détecteur alors que le temps de descente dépend uniquement de la constante $RC$ du circuit électronique. Ce mode de fonctionnement est donc adapté à des applications de caractérisation car le pic d'amplitude maximum de la tension est plus facile à mesurer car proportionnelle à la charge Q qui elle-même est proportionnelle à l'énergie du rayonnement interagissant avec le milieu détecteur. Cependant, pour des signaux présentant des pics rapprochés, des phénomènes tels que les empilements (dûs au temps de décroissance des signaux) apparaissent comme décrit précédemment.

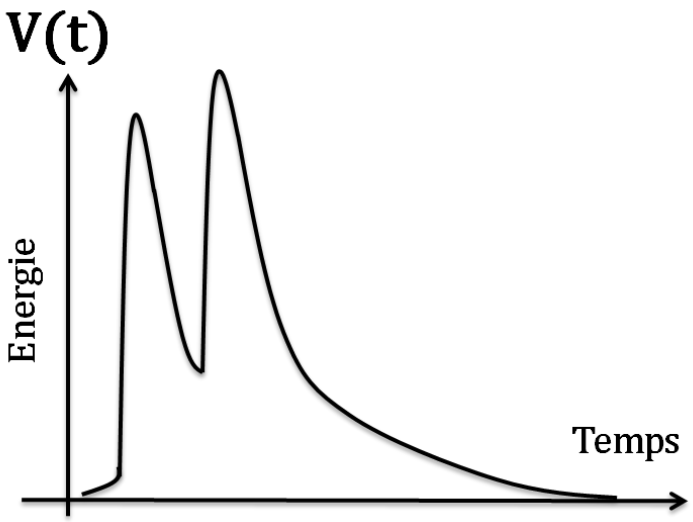



**Figure 1-6 : Exemple d'empilement, ici on peut supposer que seules deux impulsions sont empilées.**

La Figure 1-6 présente un cas typique d'empilement. Dans cette illustration, deux impulsions semblent être empilées. En effet, le phénomène d'empilement « cache » l'arrivée de certains rayonnements. Par conséquent le nombre apparent de rayonnements incidents peut être plus petit que le nombre réel. Cela se traduit par une sous-estimation de l'activité de la source, si on la mesure naïvement. D'une manière générale, un système de mise en forme (filtres de types passe-haut) est utilisé pour se prémunir des empilements avant l'acquisition des valeurs d'intérêt du signal. Si des empilements persistent, l'approche traditionnelle consiste à les rejeter à l'aide d'un système de détection des empilements adapté (Basilio Simoes & Cardoso 1994). Cependant, il est admis que plus le signal original est déformé, plus il est difficile de corréler les informations acquises à la source du signal. C'est pourquoi de récentes applications tendent à traiter les empilements de manière différente que les impulsions individuelles. De cette façon, la globalité du signal n'est pas affectée et les empilements bénéficient d'un régime particulier pour en extraire des informations corrélables au reste du signal (Trigano & Dautremer 2005).

En pratique, on recherche une constante de temps RC qui soit le meilleur compromis entre caractérisation d'énergie fine et haut taux de comptage tout en tenant compte du risque d'empilement. Quel que soit le scénario, ce sont des impulsions, qui une fois mises en forme pour être distinguables entre-elles, sont numériquement analysées et traitées. Ces impulsions sont de tailles, de formes et de durées variables. Le traitement numérique des impulsions est appelé Digital Pulse Processing (DPP) et les unités de calcul conçues pour les traiter sont appelées Digital Pulse Processor.

#### 1.1.3.3 Chaîne de mesure classique du mode impulsion

Cette sous-partie présente les différentes étapes de traitements appliqués au signal pour une chaîne de mesure classique. Cette chaîne de mesure a pour objectif de compter chaque impulsion et d'en extraire la caractéristique énergétique (ici l'amplitude).

**Alimentation haute tension :**

L'alimentation de haute tension doit, dans la très grande majorité des cas, remplir les conditions suivantes (Lyoussi 2010; Fanet 2002) :

- être réglable pour les tensions imposées par les détecteurs ;
- pouvoir supporter sans chute de tension le courant débité par le détecteur ;
- être stabilisée à hauteur de l'exigence des détecteurs ;
- ne pas présenter de dérive au cours du temps ;
- avoir un bruit très faible.

**Préamplificateur :**

Le préamplificateur est présent dans la majorité des chaînes électroniques. Ces principales fonctions sont :

- récupérer le maximum de signal ;
- minimiser les effets capacitifs ;
- adapter l'impédance élevée du détecteur avec la basse impédance du câble coaxial de transport du signal puis de l'électronique de traitement ;
- effectuer une première mise en forme du signal (production de l'impulsion à traiter) ;

Il existe trois types de préamplificateurs : en tension, en charge, en courant. Ils sont utilisés en fonction de l'éloignement de l'électronique de traitement. Par exemple, le préamplificateur de courant permet un éloignement élevé de l'électronique de traitement car il adapte l'entrée de celle-ci avec la basse impédance des câbles coaxiaux.

Le préamplificateur est inutile lorsqu'on utilise des détecteurs délivrant des impulsions suffisamment grandes pour être traitées. Par exemple, c'est le cas des compteurs Geiger-Müller, qui délivrent des impulsions de plusieurs volts. A l'heure actuelle, l'évolution des outils numériques et des convertisseurs analogiques numériques permet de se passer du reste des éléments de la chaîne et de numériser directement en sortie du préamplificateur. Ces éléments sont toujours utilisés notamment dans le monde industriel et constituent un bon point de départ pour comprendre le fonctionnement global d'une chaîne d'instrumentation nucléaire.

**Amplificateur :**

Placé à la suite du préamplificateur, l'amplificateur a pour fonction de multiplier dans un rapport donné ajustable (le gain) l'amplitude du signal qu'il reçoit. Le gain doit être linéaire sur la totalité de la dynamique des signaux d'entrée. L'amplificateur contribue aussi à la mise en forme finale du signal en vue de son analyse ou de son traitement. Il permet des réglages tels que la restauration de ligne de base, la compensation de pôle zéro. Il permet également le rejet des empilements, en liaison avec l'analyseur multicanaux.

Toutes les fonctions réalisées par l'amplificateur peuvent actuellement être réalisées par des traitements numériques (Warburton et al. 2000a).

**Discriminateur :**

Le discriminateur, ou déclencheur, produit un signal logique si une impulsion dépasse une valeur de seuil réglable. Plus couramment appelé *trigger*, son rôle peut être multiple. Son rôle principal est de détecter la présence d'impulsions, c'est à dire éliminer la contribution du bruit de fond aux signaux utiles (les impulsions). Il est fondé sur le principe du monostable (bascule temporisée). Un discriminateur plus complexe peut également signaler la présence d'empilement ou appliquer un délai entre la détection de deux événements. Il permet également la construction de spectres d'amplitudes d'impulsions en mode « intégral », voire de discriminer deux types de rayonnements dans certains cas. A l'aide du discriminateur seul, il est possible de comprendre le principe d'une chaîne de comptage ; mais il fait également apparaître la difficulté de réagir face au bruit (faux positifs) ou à l'empilement (réduction de la statistique réelle). C'est pourquoi, d'autres systèmes de discrimination ont fait leur apparition.

**Analyseur monocanal :**

L'analyseur monocanal, ou discriminateur différentiel, est composé de deux discriminateurs associés à une fonction d'anti-coïncidences qui permettent de ne prendre en compte que les impulsions dont les amplitudes sont comprises entre les deux seuils de discrimination. La Figure 1-7 illustre ce principe. Dans cet exemple, seule l'impulsion 2 sera traitée. Dans ce cas de figure, l'analyseur engendre une impulsion logique lorsque l'amplitude de l'impulsion analogique à l'entrée est comprise dans la fenêtre. L'ensemble des valeurs comprises entre ces deux seuils est proportionnel à l'énergie de l'impulsion entre ces deux seuils et permet de reconstituer un spectre proportionnel aux énergies cédées dans le détecteur (Lyoussi 2010).

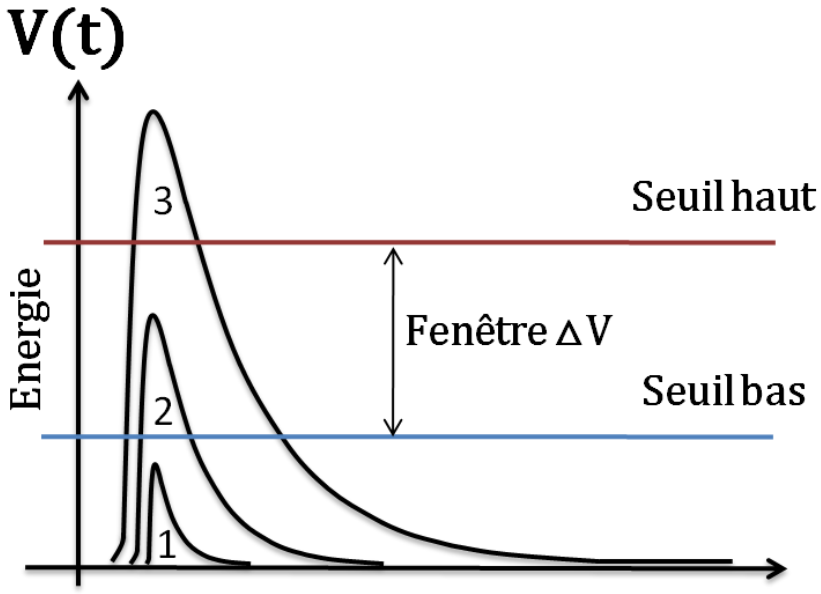



**Figure 1-7 : Principe de fonctionnement de l'analyseur monocanal.**

Avec un seuil haut réglé au maximum, l'analyseur monocanal peut également servir de discriminateur pour procéder à des opérations de comptage filtrant par définition le bruit ou les impulsions se situant sous le seuil bas.

**Analyseur multicanaux :**

Lorsqu'on cherche à mesurer l'énergie des impulsions pour que le sélecteur monocanal soit efficace, la largeur du canal (fenêtre de mesure) doit être faible, sinon il n'est pas possible de restituer correctement la forme des pics. Il est conventionnel d'utiliser alors des analyseurs multicanaux. Les analyseurs multicanaux peuvent être considérés comme une somme de nombreux analyseurs monocanaux contigus. A chaque fenêtre de discrimination est associée une valeur en énergie. Plus le nombre de discriminateurs est grand, plus le nombre d'énergies discriminables est grand. Dans le principe, il s'agit d'un convertisseur analogique-numérique (CAN). Il est cependant équipé d'une mémoire. Cette mémoire est divisée en segments qui sont aussi appelés canaux. A chaque canal est associée une valeur en énergie propre à la fenêtre de discrimination correspondante. L'affichage de l'ensemble de ces canaux est appelé spectre. Le spectre est donc un histogramme et le principe est ainsi corrélé à l'application de spectrométrie expliquée en section 1.2.

Le spectre résultant de cette analyse est représentatif de l'ensemble des énergies détectées. La Figure 1-8 représente un schéma simplifié d'un spectre où les sommes des énergies présentes dans un même canal font apparaître les caractéristiques de l'énergie mesurée.

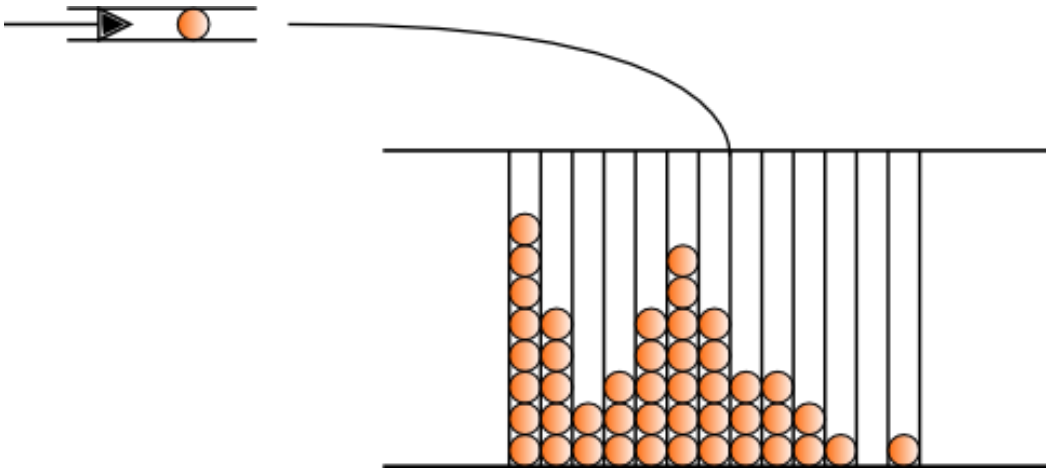


**Figure 1-8 : Principe schématique simplifié de la construction d'un spectre en énergie.**

### 1.1.4 Le temps mort

Si le dépôt d'énergie résultant de l'interaction d'un rayonnement dans un détecteur peut être considéré comme instantané (d'une picoseconde à quelques nanosecondes), l'information finale délivrée par le détecteur met en général beaucoup plus de temps à survenir. Si le détecteur délivre des impulsions électriques, ce signal peut durer de quelques ns (scintillateurs plastiques) à quelques ms (détecteurs gazeux). Cela dépend du type de détecteur et de sa géométrie. La durée des impulsions produites par les interactions est alors non-nulle et variable. Si l'on considère que le signal est numérisé directement en sortie d'un premier étage de pré-amplification, à cette durée devient alors associé un nombre d'échantillons proportionnel à la taille de l'impulsion.

Une fois le signal numérisé, si la durée des traitements associés à une impulsion est supérieure à la durée de l'impulsion elle-même, alors il s'en suit une période pendant laquelle le détecteur est « occupé » et il est donc tout ou partiellement indisponible pour le traitement complet d'une impulsion suivante. Il s'ensuit des pertes d'informations d'autant plus importantes que l'intensité (ou fréquence d'arrivée des particules sur le détecteur) des particules incidentes est élevée. Ces pertes d'informations doivent être corrigées pour remonter à l'irradiation réelle.

Pour caractériser cette période où le détecteur est « occupé », on utilise la notion de **temps mort**.

Définition : *le temps mort est le plus petit intervalle de temps entre deux informations pour que chacune d'entre elles soit prise en compte par le système ; Il s'agit du temps d'aveuglement du système de mesure, le temps où il ne peut traiter une information supplémentaire tant qu'il termine de traiter l'information en cours.*

Ces pertes affectent les taux de comptages observés et distordent la distribution temporelle des évènements. Pour éviter de larges effets de temps mort, il faut tendre vers un taux de comptage plus faible ou une réduction de temps mort de la chaîne de mesure. Il y a donc un compromis entre haut et bas taux de comptage. Chaque élément du système dispose de son propre temps mort. Par exemple, dans le cadre d'un CAN, le temps mort est la durée séparant deux échantillons. Si l'objectif est de réaliser l'ensemble des traitements en numérique, alors cela oblige à être capable de travailler en flux sur les échantillons numérisés sous peine de temps mort.

Une chaîne de détection se distingue par sa capacité à gérer le temps mort. On distingue deux types de mode de gestion du temps mort. On parle ici soit de détecteur à temps de résolution extensible, fixe, ou non-paralysable, soit de détecteur non-extensible, reconductible, paralysable ou cumulatif.

- Le temps de résolution de type fixe : pendant la durée $\tau$, le détecteur n'est pas affecté par toute interaction consécutive à celle qui engendre la formation du signal. Le détecteur a donc une réponse en fonction du taux d'informations qui tend vers une saturation. On peut chiffrer le taux moyen d'informations recueillies, *m*, en fonction du taux moyen d'interactions dans le détecteur, *n,* grâce à l'équation 1-3. La Figure 1-10 illustre le cas d'un détecteur paralysable.

$$m = \frac{n}{1 + n\tau} \qquad \textbf{1-3}$$

- Le temps de résolution de type reconductible : la durée d'occupation $\tau$ du détecteur est reconduite de $\tau$, ce dernier restant sensible à toute interaction consécutive à celle qui vient d'engendrer la formation du signal. Le détecteur a donc une réponse en fonction du taux d'informations qui passe par un maximum puis tend à s'annuler ; le détecteur est dit paralysable. Un même taux d'informations peut correspondre à deux taux d'interactions distincts. Le taux moyen d'informations recueillies, $m$, en fonction du taux moyen d'interactions dans le détecteur, *n,* est présenté par l'équation 1-4.

$$m = n * e^{-n\tau} \qquad \textbf{1-4}$$

La Figure 1-10 illustre le comportement d'un détecteur paralysable pour la gestion du temps mort. Dans cet exemple, le temps mort est reconduit à chaque impulsion détectée, et finalement seule l'impulsion 5 est traitée.

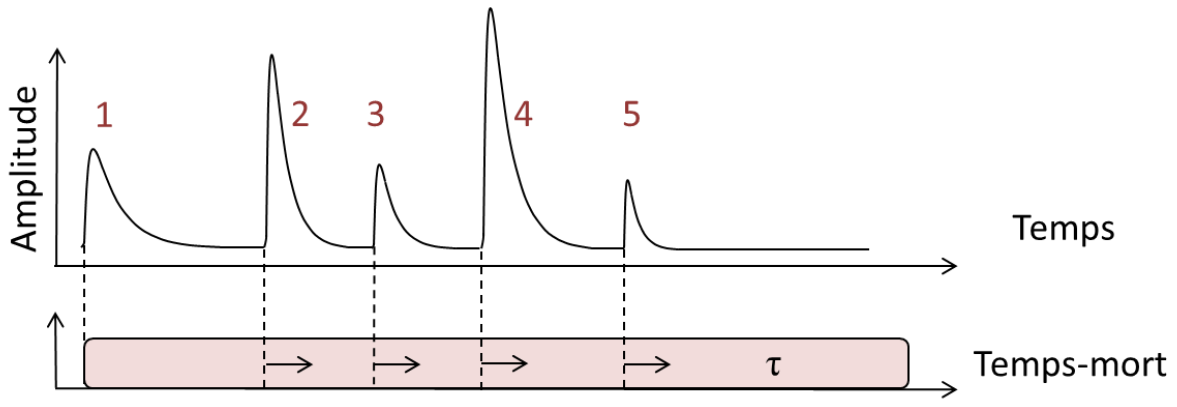



**Figure 1-9 : Illustration de la problématique de temps mort paralysable. Dans cet exemple, seule l'impulsion 5 est traitée.**

La Figure 1-10 illustre le comportement d'un détecteur non-paralysable pour la gestion du temps mort. Dans cet exemple, seules deux impulsions sur cinq sont traitées car le système ne se réenclenche pas avant que l'impulsion en cours soit totalement traitée.

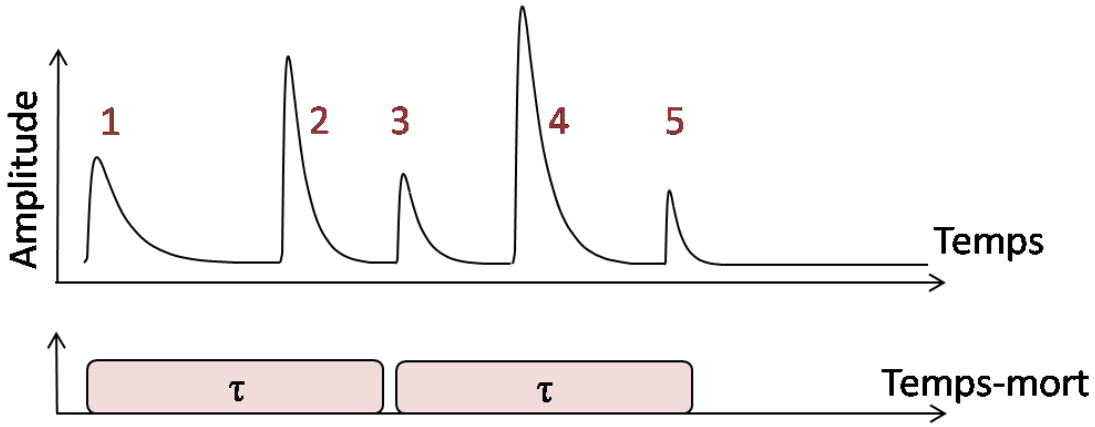



**Figure 1-10 : Illustration de la problématique de temps mort non-paralysable. Dans cet exemple, seules les impulsions 1 et 3 sont traitées.**

Il est acquis qu'un système paralysable ne permet pas d'obtenir un taux d'impulsions traitées supérieur à un système non-paralysable. De plus, à haut taux de comptage, le système paralysable se dégrade en terme de nombre d'impulsions traitées fonction du nombre d'impulsions détectées (Knoll 2010). Enfin, il est possible de connaître l'impact général du temps mort du système et donc, dans le cadre d'un système non-paralysable de remonter statistiquement au taux d'informations réelles détectées à partir du taux d'informations traitées.

Le temps mort est une notion qui soulève un grand nombre d'ambiguïtés quant à sa définition. En effet, si l'on parle de temps mort au niveau de la partie physique du détecteur, si les impulsions s'empilent jusqu'à saturer le système de détection, alors aucune impulsion n'est visible. On peut donc parler de temps mort si l'on se réfère à la définition stricte. C'est pourquoi, dans la suite du manuscrit nous parlerons uniquement de temps mort électronique puisque l'ensemble de nos travaux se placent après conversion analogique-numérique.

### 1.1.5 Conclusion

Cette section a mis en évidence la nature du signal à traiter par une architecture unifiée (multi-applicative) pour l'instrumentation nucléaire. Si les chaînes actuelles numérisent les signaux directement en sortie de préamplificateur pour réaliser tous les traitements en numérique (Warburton et al. 2000a), alors notre architecture doit travailler sur un signal échantillonné composé d'impulsions de durées aléatoires et aux dates d'arrivées non-déterministes. La problématique majeure mise en évidence est le temps mort. En général la perte d'impulsion due au temps mort n'est pas considérée comme problématique pour des mesures longues en laboratoire où la nature aléatoire du signal garantit d'avoir une statistique proche de l'activité de la source même en perdant des impulsions. Cependant, dans le cadre d'applications sur le terrain où la durée des mesures est sous contrainte (portiques par exemple), perdre trop d'impulsions à cause du temps mort ne permet pas d'aboutir à une certitude de mesure suffisamment élevée pour prendre une décision (seuil d'alarme par exemple). De plus, pour des traitements compliqués et longs, le risque de temps mort augmente. Cela signifie rater beaucoup d'impulsions et donc rendre inutilisable la mesure pour des applications temps réel. La gestion ou la suppression du temps mort (être capable de travailler en flux) est donc une condition *sine qua non* pour une architecture se voulant multi-applicative. Cependant, être programmable signifie ne pas connaître à l'avance la durée d'exécution du programme sur un échantillon. C'est pourquoi, pour

ne pas avoir de temps mort où le contrôler totalement, les chaînes d'instrumentation nucléaire développent des chaînes d'acquisition dédiées. L'état de l'art présenté en partie 1.6 affirme cette idée.

## 1.2 Applications du mode impulsion

Cette partie s'intéresse à l'étude des applications du mode impulsion. Les grandes familles d'application sont analysées au cas par cas afin de faire apparaître les contraintes qu'elles imposent à une architecture multi-applicatives de traitements des impulsions.

### 1.2.1 Comptage et spectrométrie

Dans un premier temps, deux types d'applications peuvent être identifiés : le premier, est l'ensemble des méthodes permettant d'estimer le nombre d'impulsions pendant un temps donné. On parle ici de comptage. Le second est l'ensemble des techniques permettant de mesurer une caractéristique particulière pour chaque impulsion, la charge de l'impulsion étant proportionnelle à l'énergie déposée dans le détecteur. Ces deux mesures, corrélées, sont appelées mesures spectrométriques. Nous l'avons vu lors de l'analyse du mode impulsion d'une chaîne de détection, le comptage et la spectrométrie peuvent être réalisés par l'intermédiaire de discriminateurs, analyseurs mono et multicanaux.

#### 1.2.1.1 Le comptage

Le comptage consiste à mesurer le nombre de particules pénétrant dans le détecteur par unité de temps. La mesure est donc binaire ; on parle de « coups », l'unité utilisée étant cps ou nombre de coups par seconde (counts per second). Pour une interaction (donc une impulsion) détectée, un seul coup doit être compté. La Figure 1-11, présente une chaîne de mesure en mode comptage et peut être décomposée de la manière suivante :

- en sortie de préamplificateur, le signal est dans un premier temps mis en forme de manière à maximiser le rapport signal-à-bruit et corriger éventuellement la forme des impulsions, la ligne de base, ou atténuer le bruit ;

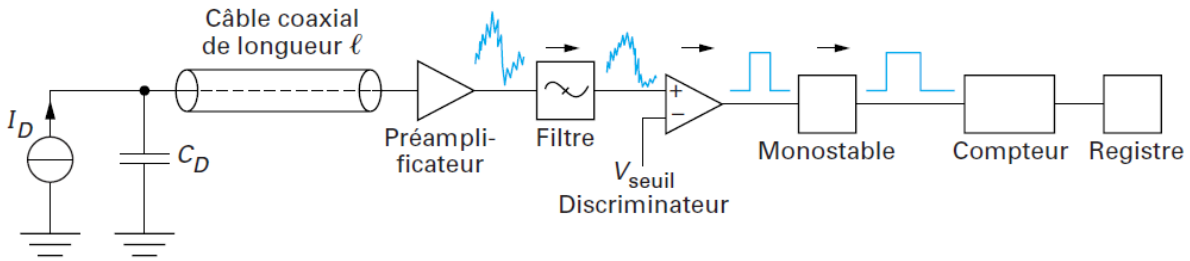



**Figure 1-11 : Synoptique d'une chaîne de mesure en mode comptage**

- le discriminateur, présenté en 1.1, est généralement un comparateur à seuil, il transforme l'impulsion électrique analogique en une impulsion logique, indiquant la présence d'un signal ;

- les impulsions logiques sont ensuite calibrées par un monostable puis comptées. Le résultat de la mesure est mis en mémoire dans un registre accessible par le logiciel de comptage ;
- le registre doit donc être lu au moins aussi vite que le taux de comptage entrant si l'on ne souhaite pas perdre d'information ;

Le registre de comptage est alors lu et mis à jour selon une période définie à l'avance permettant de représenter l'activité de la source mesurée en coups par seconde.

#### 1.2.1.2 La spectrométrie

La spectrométrie consiste à mesurer l'énergie déposée dans le détecteur pour chaque interaction. L'objectif principal d'une chaîne de spectrométrie est, pour chaque impulsion, d'extraire l'information en énergie.

Cela peut s'effectuer de deux façons :

- la plus classique est d'extraire l'amplitude (l'énergie du pic) des impulsions (spectre en énergie) que l'on peut convertir, par le coefficient adéquat (obtenu par calibration de la chaîne), en énergie perdue par le rayonnement. Il s'agit de la fonction étudiée par l'analyseur multicanal ;

- la seconde méthode consiste à calculer l'aire des impulsions. De manière analogique, on parlera d'intégrateur actif de charge. De manière numérique, plusieurs solutions sont possibles pour calculer l'aire, de la somme des échantillons à la somme d'aires de rectangles ou de trapèzes ;

Ces deux méthodes permettent une grande variété de chaînes de mesures dont les différences se situent principalement au niveau des étapes de mises en forme du signal dont les objectifs sont multiples : correction de la ligne de base, augmentation du rapport signal à bruit/amplification, lissage, rejection d'empilements, fonctions pondérées (filtre trapézoïdal par exemple). Ces algorithmes sont expliqués plus en détail dans la partie 1.3.

La difficulté de ce type de mesure est de s'assurer que le détecteur et l'électronique peuvent revenir à la notion d'énergie par la simple analyse des impulsions électriques par application d'une fonction réciproque. Du point de vue numérique un spectre est un tableau dans lequel chaque case représente un canal en énergie exprimée en électron-volt (eV). Dans le cadre d'un spectre en amplitude, le nombre de canaux du spectre est donc directement corrélé à la résolution de codage du CAN. Un spectre issu d'une grandeur analogique codée sur 16 bits n'excédera pas, en théorie, 65536 canaux. A l'instar de l'analyseur multicanal, pour chaque valeur d'énergie extraite d'une impulsion, le numéro de canal correspondant est incrémenté après un nombre d'acquisitions suffisant pour faire apparaître des canaux qui se distinguent en termes de nombre d'incréments (coups), comme chaque radionucléide dispose d'une signature spectrale unique, il est possible d'identifier le ou les radioéléments à l'origine de l'activité mesurée grâce au spectre. Un spectre en énergie est calibré en fonction d'une source étalon dont l'activité est connue. De cette façon, il est possible de savoir à quelle énergie correspond un canal. Le $^{60}$Co est par exemple une source étalon typique pour la spectrométrie gamma.

La Figure 1-12 représente un spectre en énergie mesuré à l'aide d'une source de $^{60}$Co. Le $^{60}$Co se caractérise par l'émission caractéristique de deux rayonnements gamma de haute intensité (haute probabilité), respectivement à 1,1732 et 1,3325 MeV. La prédominance de deux raies à ces énergies (ou canaux correspondants) sur le spectre, nous permet alors de dire qu'il s'agit d'un spectre de $^{60}$Co.

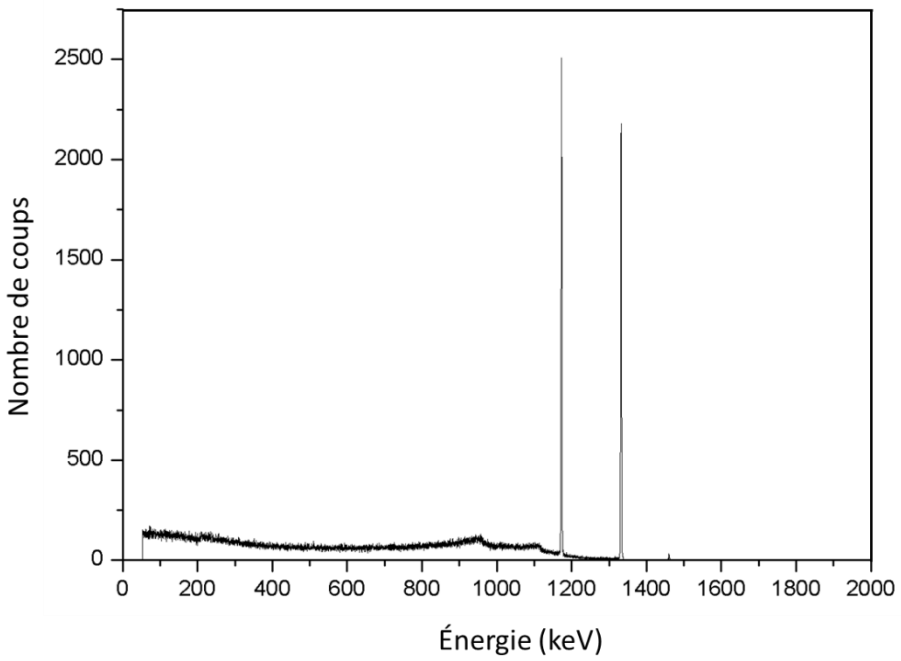



**Figure 1-12 : Exemple de spectre en énergie mesuré à l'aide d'une source de 60Co.**

La construction du spectre est généralement réalisée par un ordinateur distant et l'analyse de celui-ci est réalisée soit par un expert (un physicien) soit par un logiciel. Afin de rappeler les enjeux de cette thèse, nous précisons qu'elle ne se positionne pas sur les problématiques d'analyse de spectre mais uniquement sur les problématiques de mesure.

**Problématique d'empilements :**

La problématique d'empilement est relative à tout type de mesure du mode impulsion. Nous choisissons de la présenter dans le cadre de l'application de spectrométrie car il est aisé de comprendre son impact, en particulier sur un spectre en énergie.

D'une manière générale, l'empilement d'impulsions a deux conséquences (TRIGANO, 2006) :

- l'empilement se traduit concrètement par un déplacement erroné d'une partie du spectre vers les fortes énergies. Puisque les énergies s'accumulent, la mesure d'amplitude ou d'aire est alors biaisée. Cela peut gêner considérablement l'identification des radionucléides ;
- le phénomène d'empilements « cache » l'arrivée de certains rayonnements. Par conséquent, le nombre apparent de rayonnements incidents est plus petit que le nombre réel. Cela se traduit donc par une sous-estimation de l'activité de la source, si on la mesure naïvement.

**Problématique de temps mort :**

Nous profitons également de la présentation de la spectrométrie pour offrir une vue de l'impact du temps mort. L'influence du temps mort est également de deux ordres :

- dans le cadre du monde non-paralysable, si les traitements d'une impulsion s'arrêtent au moment où une impulsion est déjà au-dessus d'un seuil de déclenchement, alors il se peut que le système traite des impulsions tronquées. Cela signifie que des énergies biaisées apparaîtront dans le spectre, voire à l'apparition de pics ne correspondant à aucun élément physique. Cette problématique est donc induite directement par l'étage de discrimination ;
- plus il y a de temps mort, moins il y a d'impulsions traitées vis-à-vis du taux d'impulsion réellement détecté. Cela signifie que des informations sont perdues ; il en résulte le besoin d'attendre un plus grand nombre d'impulsions avant de pouvoir analyser un spectre qui se « construit » plus lentement.

### 1.2.2 Discrimination de forme d'impulsion

Selon le détecteur, les interactions de rayonnement de natures différentes peuvent aboutir à la génération d'impulsions de formes différentes. A l'inverse de la spectrométrie qui s'intéresse à la quantité d'énergie déposée dans le détecteur et l'impulsion, ici la forme de l'impulsion, croissance ou décroissance, est analysée. La discrimination de forme d'impulsion (Pulse Shape Discrimination (PSD)) rassemble tous les procédés permettant l'identification d'impulsions issues d'interactions de natures différentes. Une application de type PSD est la discrimination neutron-gamma (n-$\gamma$). Dans cette partie, nous choisissons de nous focaliser sur cette application importante du mode impulsion. En effet, s'il existe des détecteurs qui discriminent naturellement (de par leur composition chimique) des rayonnements neutron des rayonnements gamma (détecteur à Hélium 3 ($^3$He)), dans d'autres détecteurs cette différence est mince ou inexistante. A cause de la pénurie d'$^3$He, l'un des enjeux majeurs de ces dernières années concerne la discrimination neutron-gamma à l'aide de scintillateurs organiques, et plus précisément les scintillateurs plastiques (Blanc et al. 2014; Bertrand et al. 2015), principalement pour des questions de coûts et de pérennité qui ne sont pas abordés dans cette thèse.

Les applications de discrimination n-$\gamma$ concernent l'ensemble des méthodes qui permettent de mettre en évidence la différence entre deux impulsions de nature différente à l'aide de moyens électroniques et algorithmiques. Par exemple, dans le cadre des scintillateurs organiques, la différence de forme d'une impulsion neutron et d'une impulsion gamma s'observe en analysant les décroissances de celles-ci. La décroissance d'une impulsion neutron est plus longue, à amplitude équivalente, que la décroissance d'une impulsion gamma. Cette différence peut s'observer en Figure 1-13.

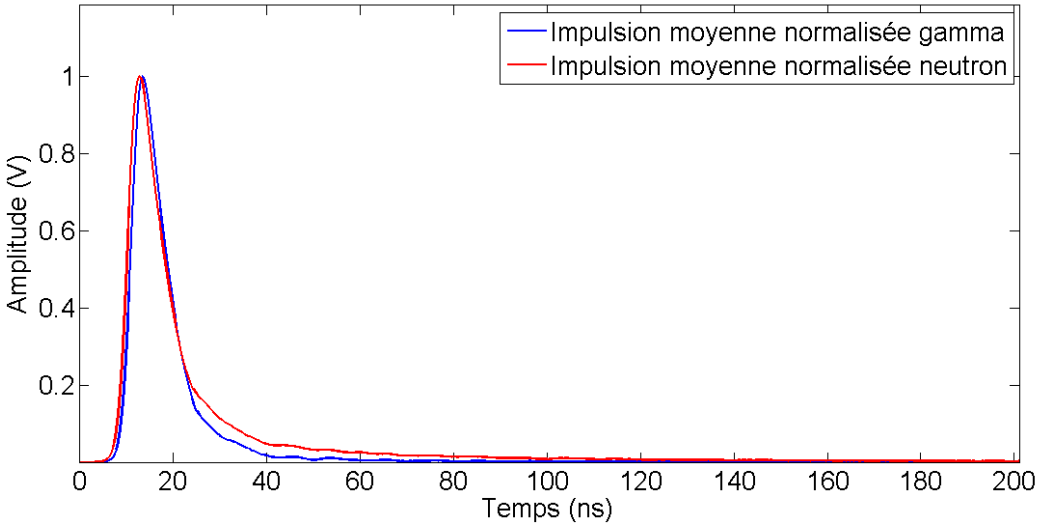



**Figure 1-13 : Moyennes normalisées des impulsions neutron et gamma obtenues à l'aide d'une source $^{252}$Cf sur scintillateur BC-501A.**

Dans la pratique, deux méthodes sont principalement utilisées. Le passage par zéro (*Zero Crossing* (Z/C)) (Corre et al. 2009) et la comparaison de charge par intégration du signal (Zaitseva et al. 2011). La première consiste à intégrer le signal puis à différencier le signal intégré, c'est-à-dire la charge. Le moment où le signal différencié (donc la dérivée de la charge) passe par la ligne de base (ou le zéro) dépend du type de particule. La seconde méthode consiste à extraire le rapport de la charge lente par rapport à la charge totale de l'impulsion. La charge lente est notée $Q_{tail}$, et correspond à l'intégration du signal dans la fin de sa décroissance, et la charge totale $Q_{tot}$, correspond à l'intégralité de la charge déposée. La Figure 1-14 représente la chaîne d'acquisition permettant de faire cette discrimination.

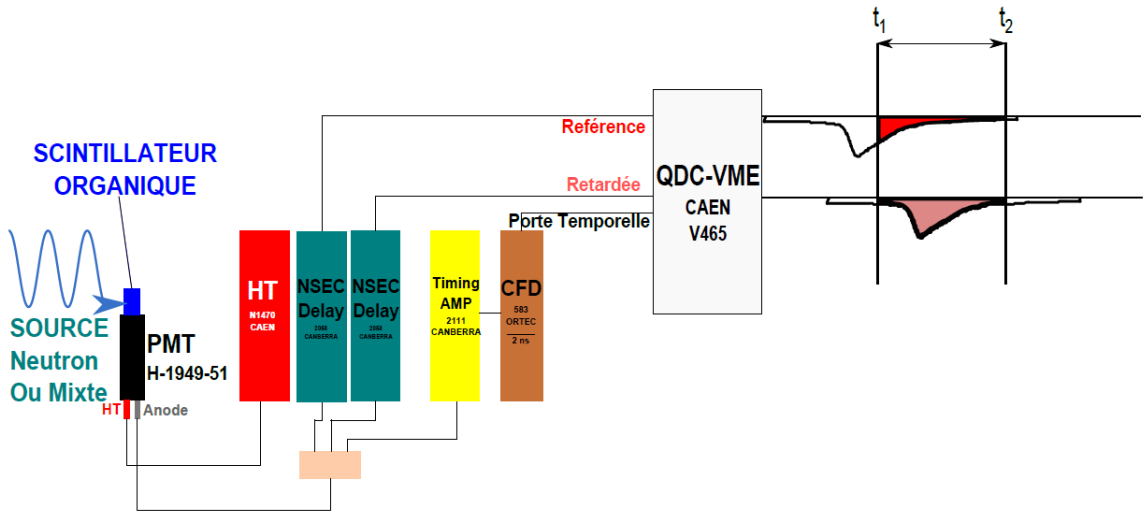



**Figure 1-14 : Dispositif expérimental d'intégration de charge pour l'évaluation des efficacités de discrimination n-γ des scintillateurs organiques.**

Le principe est assez simple : l'arrivée de l'impulsion déclenche, par l'intermédiaire du Discriminateur à Fraction Constante (CFD, détaillé en 1.3), la création de deux portes temporelles. La première porte répond directement au signal du CFD pour déclencher l'ouverture de la porte de $Q_{tot}$. La deuxième, dont le départ est décalé par l'intermédiaire d'un délai configurable, sera déclenchée pour ouvrir la porte de $Q_{tail}$. Ainsi, deux parties différentes du signal sont converties en tension et numérisées par l'intermédiaire d'un convertisseur numérique de charge (Charge-Digital Convertor (QDC)). Après avoir intégré de manière numérique la charge totale des deux signaux, un ordinateur distant est utilisé pour tracer une courbe en fonction de $Q_{tot}$ et $Q_{tail}$ pour mettre en évidence la discrimination neutron gamma. La Figure 1-15 représente cette fonction. Elle exprime le rapport $Q_{tot}/Q_{tail}$ en fonction de l'énergie des impulsions traitées (appelée spectre de discrimination).

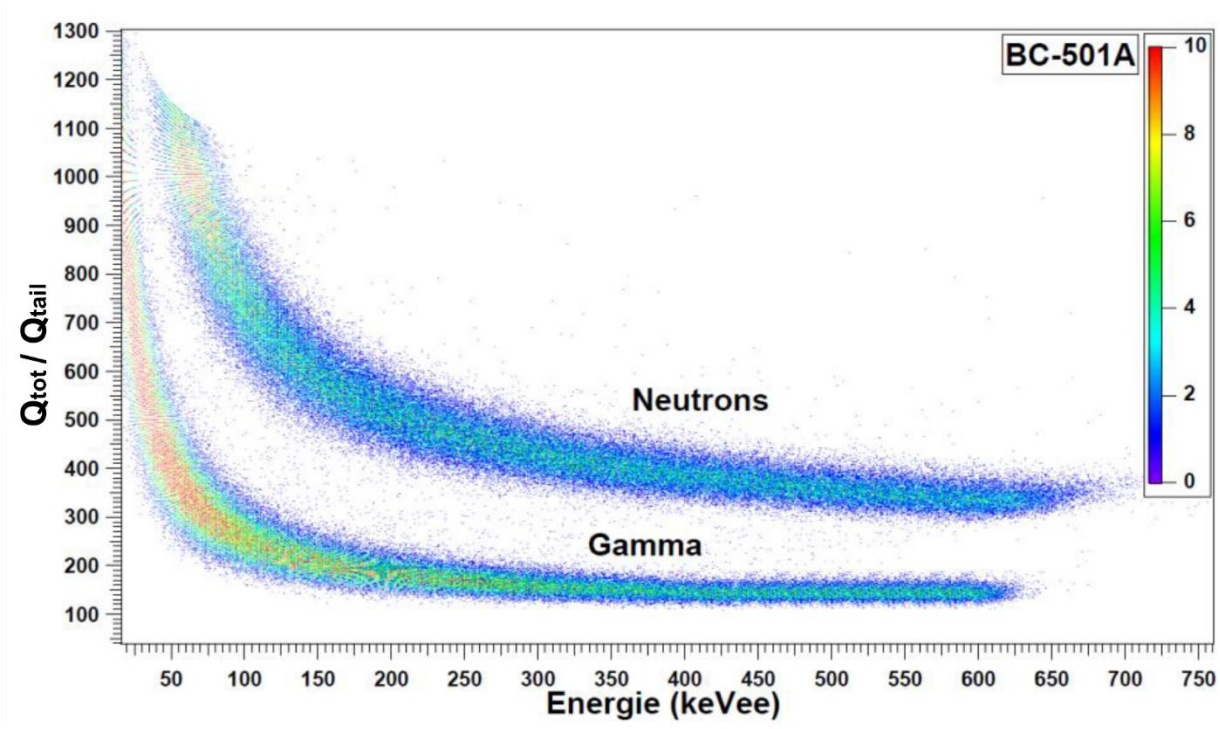



**Figure 1-15 : Spectre de discrimination n-γ par comparaison de charge d'un détecteur BC-501A pour une source d'Américium Béryllium 241 ($^{241}$AmBe).**

**Remarque :**

Si les deux méthodes sont corrélables, la deuxième a donné lieu à l'apparition de nombreux algorithmes de discrimination neutron-gamma dans la littérature. Ils seront présentés dans la partie 1.3. Si ces algorithmes sont nombreux, c'est qu'ils sont principalement implémentés hors-ligne. Or, l'application finale nécessite de pouvoir discriminer en ligne les neutrons des gammas et donc que ces algorithmes soient implémentés en ligne eux aussi. Ce point nous permet de mettre plus en avant l'importance d'une chaîne de mesure programmable.

### 1.2.3 Mesure de temps de vol et coïncidence

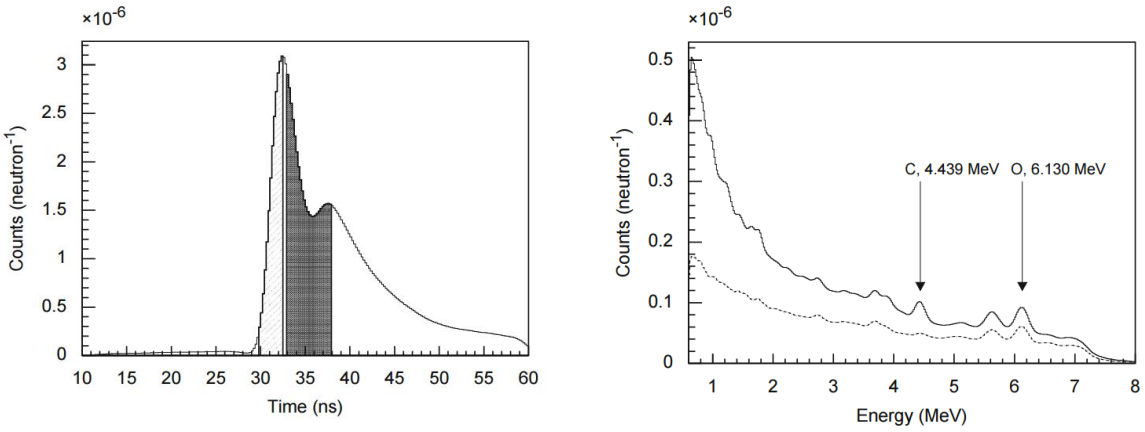
Deux cas applicatifs, la mesure de temps de vol et la coïncidence, mettent en évidence la notion d'acquisition multivoies et de datation des impulsions.

Le premier cas applicatif présenté est la coïncidence. La coïncidence peut être réalisée dans le cadre d'une mesure active ou passive. Certains radioéléments peuvent être caractérisés par des émissions dites simultanées. Par exemple, si un radioélément émet plusieurs neutrons en même temps suite à une émission spontanée (mesure passive) ou lors d'une réaction engendrée par l'utilisateur (mesure active), alors, il est possible en utilisant plusieurs détecteurs, de corréler temporellement la détection d'événements simultanés sur ces détecteurs.

Le second cas applicatif est la mesure de temps de vol. Ici, l'objectif est de calculer le temps qu'un signal émis par un individu (rayon interrogateur) met pour atteindre un détecteur. GPS, télémétrie, radar, tous ces systèmes utilisent ce principe. De manière générale, il est nécessaire de connaître la vitesse du rayonnement émis. Ainsi, cela permet de connaître la distance entre le détecteur/émetteur et cette surface. De ce fait, il est possible d'avoir une notion de localisation de l'élément radioactif (un exemple d'application est la Tomographie par Emission de Positons (TEP)). Ce principe appliqué à la mesure de rayonnement est plus complexe pour trois raisons. Premièrement, les interactions entre des rayonnements ionisants et la matière ne provoquent que très rarement un phénomène équivalent aux « rebonds » d'ondes sonores par exemple. De plus, il n'est pas possible de choisir le top départ d'une particule interrogatrice (sans utiliser une méthode dite destructive) et il n'est pas possible de moduler les caractéristiques du signal émis.

Le cas applicatif présenté dans ce paragraphe est celui de la mesure de coïncidence. La coïncidence est, nous le rappelons, le moyen d'associer les signaux de différents détecteurs correspondant à un même événement. Le principe est le suivant : nous souhaitons associer les événements qui arrivent sur des détecteurs à une valeur de temps. On appellera ce principe la datation.

La méthode la plus simple consiste à utiliser un convertisseur temps amplitude (TAC). Ce module utilise deux entrées recevant des signaux logiques indiquant les dates de détection de signaux pour chaque détecteur. A la réception d'un signal du détecteur 1, il déclenche une minuterie d'une certaine durée (de l'ordre de dizaines de nanosecondes), et s'il reçoit un signal du second détecteur pendant cette durée, il émet une impulsion d'amplitude proportionnelle à la durée écoulée. Ainsi, on peut corréler les informations mesurées par une méthode de spectrométrie standard (corrélation énergie/nombre de coups) avec ces nouvelles valeurs de temps de coïncidence. Ces trois informations nous permettent d'obtenir un spectre « en trois dimensions ». Cela signifie que l'on peut extraire un spectre en énergie pour chaque durée de coïncidence détectée. La Figure 1-16 représente un spectre de temps de vol et le spectre en énergie associé à la région sélectionnée sur le spectre de temps de vol (Carasco et al. 2010).




(a) (b)

**Figure 1-16 : Spectre de temps de vol (a) et spectre en énergie correspondant à la région sélectionnée en grise (b).**

### 1.2.4 Conclusion sur la présentation des cas applicatifs

Dans cette partie, les principaux cas applicatifs du mode impulsion, le comptage, la spectrométrie, la discrimination, la coïncidence temporelle et le temps de vol ont été présentés. Cette étude permet d'aboutir à une première spécification d'une architecture numérique multi-applicative de traitement des impulsions. En se basant sur les problématiques inhérentes à ces différentes applications, les postulats suivants peuvent être formulés :

- le comptage est réalisé dans n'importe quelle application ;
- les applications de spectrométrie imposent de pouvoir extraire une information en énergie pour chaque impulsion. Une mise en forme du signal est parfois nécessaire pour faciliter la récupération de ces informations ou pour se rapprocher le plus possible du rapport de proportionnalité énergie de l'impulsion/énergie déposée dans le détecteur ;
- la discrimination utilise les même bases que la spectrométrie mais introduit la notion de fenêtrage pour sélectionner les régions d'intérêt (temps de montée, temps de descente, charge totale) ;
- la coïncidence utilise également les mêmes bases que la spectrométrie mais y ajoute la notion de datation ainsi que la nécessité de gérer simultanément plusieurs voies de mesure.

De plus, cette étude met en évidence des problématiques similaires à chaque application :

- la gestion de l'empilement est commune à tous les cadres applicatifs si l'on souhaite ne pas biaiser la mesure ;
- la gestion du temps mort est primordiale dans tous les cas applicatifs si l'on souhaite ne pas perdre trop d'information et si l'application est contrainte en durée de mesure.

Enfin, on remarque que certains des prérequis applicatifs peuvent se compléter :

- le fenêtrage utilisé dans la discrimination neutron-gamma n'empêche pas une analyse de type spectrométrie ;

- la discrimination neutron-gamma peut se combiner avec une application de coïncidence neutronique pour supprimer les incidences gamma et inversement dans le cadre de la spectrométrie gamma.
- la gestion de plusieurs voies de mesure en parallèle permet l'acquisition d'un échantillon statistique plus grand dans le cadre de la spectrométrie, ce qui se conclut par la construction plus rapide d'un spectre.

En résumé, d'un point de vue strictement architectural, une chaîne de mesure multi-applicative doit répondre aux caractéristiques suivantes :

- posséder un dispositif de discrimination pouvant extraire les impulsions entièrement afin de les rendre accessibles à tout type de mesure ;
- gérer plusieurs voies de mesures en parallèle ;
- faciliter l'intégration de différents algorithmes de traitement ;
- gérer voire supprimer le temps mort ;
- être suffisamment flexible pour s'adapter aux prérequis de performances de calculs de l'ensemble détecteur + traitement.

L'ensemble des applications décrites utilise des algorithmes pouvant être regroupés en trois catégories. L'étude de ces catégories d'algorithmes est nécessaire afin de caractériser les architectures de traitement qui doivent être mises en œuvre.

## 1.3 Algorithmes du mode impulsion

Cette partie a pour objectif l'analyse des différents algorithmes associés aux applications du mode impulsion. Cette analyse doit permettre de comprendre quels types d'algorithmes sont utilisés, et quels sont leurs impacts sur le dimensionnement d'une architecture. L'objectif étant d'aboutir à une architecture programmable, nous avons choisi d'implémenter (en Matlab et C) et de tester un jeu d'algorithme représentatif du mode. Les algorithmes ont été testés et validés à l'aide de jeux de données réelles. Ils peuvent être classés selon trois catégories pour lesquelles nous détaillerons un ou plusieurs algorithmes :

- les algorithmes de discrimination/déclenchement (ou *trigger) ;*
- les algorithmes de mise en forme du signal ;
- les algorithmes permettant l'extraction de l'information d'intérêt dans chaque impulsion ; dans cette catégorie se trouvent les algorithmes de PSD puisque leur objectif est bien d'extraire une caractéristique (dire si l'impulsion est issue d'une interaction neutron ou gamma par exemple) ;

### 1.3.1 Algorithmes de discrimination

Les algorithmes de déclenchement ont pour objectif de sélectionner une région d'intérêt du signal, en l'occurrence les impulsions. On distingue deux grands types de déclencheurs : les déclencheurs à seuils et les déclencheurs à passage par zéro pour la détection des impulsions (détection de l'arrivée d'une impulsion). A ces déclencheurs peuvent s'ajouter des fenêtres d'observation, qui servent à déterminer l'instant d'arrêt de la sélection de l'impulsion (comme

présenté dans l'application de discrimination neutron-gamma). A titre d'exemple nous choisissons de présenter un CFD (Fallu-Labruyere et al. 2007).

Le CFD est principalement utilisé pour détecter (et dater) équitablement les impulsions. En effet, pour des instants d'arrivée équivalents, l'amplitude ou la forme des impulsions peut varier. Cela signifie que l'utilisation d'un seuil de détection traditionnel (seuil simple) ne permet pas des mesures de temps précises. Cette problématique est illustrée en Figure 1-17. Dans cette figure, si $t$ représente l'instant d'arrivée considéré pour une impulsion, alors l'utilisation d'un seuil en énergie (A) ne permet pas de détecter les deux impulsions représentées au même instant. L'instant d'arrivée est donc dépendant de l'énergie de l'impulsion (B).

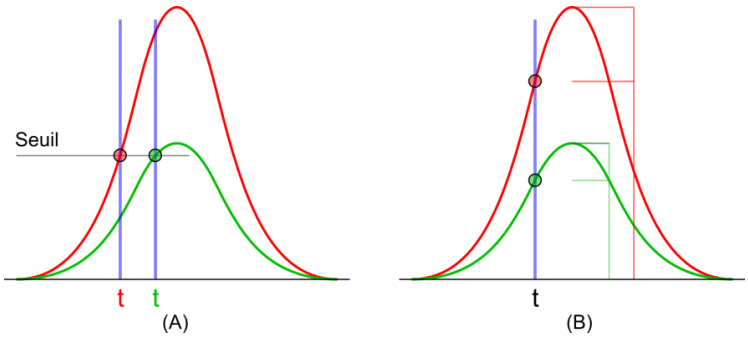



**Figure 1-17 : Problématique de détection équivalente d'impulsion de tailles différentes.**

Le principe du CFD est le suivant : le signal d'origine est retardé par un délai D et une copie de celui-ci est inversée et multipliée par un coefficient $C$, avec $0 < C < 1$. Les deux signaux sont ensuite additionnés. Ce processus, lorsqu'il est optimisé, transforme l'impulsion unipolaire en une impulsion bipolaire. L'impulsion bipolaire coupe l'axe du temps à une fraction constante de la hauteur de l'impulsion initiale. Le temps de passage est généralement calculé par *zero-crossing* si le signal est normalisé autour de 0 mais une interpolation linéaire doit être réalisée s'il a eu lieu entre deux échantillons. Cette méthode peut être réalisée à l'aide de l'équation 1-5. Avec $x(t)$ le signal d'entrée, $D$ le délai et $C$ le coefficient de fraction et $y(t)$ le signal résultant.

$$y(t) = x(t - D) - C * x(t) \qquad \textbf{1-5}$$

La Figure 1-18 illustre la transformation d'une impulsion idéale par CFD.

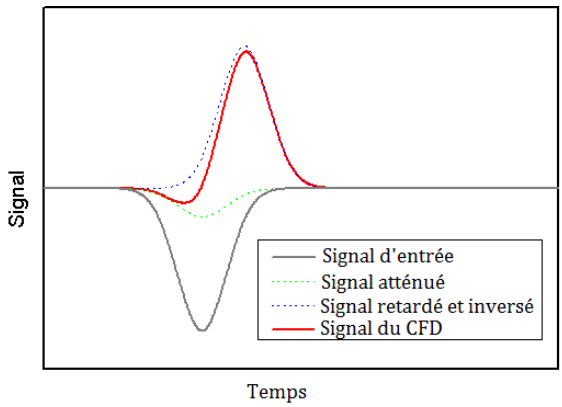



**Figure 1-18 : Exemple d'une discrimination à fraction constante**

On retiendra des algorithmes de déclenchement qu'ils sont sensibles au bruit et aux variations de ligne de base. Les déclencheurs décrits dans la littérature sont généralement dédiés à des applications spécifiques, c'est-à-dire qu'ils répondent à des paramètres statiques définis à l'avance par l'utilisateur pour un signal donné. Cela a pour conséquence d'aboutir à des algorithmes de déclenchement qui tronquent les impulsions, si seule l'acquisition de l'amplitude est visée, ou qui ne sont pas capables de s'adapter à la variabilité de taille des impulsions (problématique dans le cadre de la discrimination n-$\gamma$). Ces problématiques sont détaillées dans le chapitre 4 de cette thèse. La réjection d'empilement, qui est en général réalisée à cette étape, ne sera pas détaillée.

### 1.3.2 Algorithmes de mise en forme du signal

Les algorithmes de mise en forme consistent en la modification du signal afin d'améliorer le résultat d'algorithmes d'extraction de régions d'intérêt. Ils peuvent être réalisés avant ou après l'étape de discrimination du signal. Les algorithmes de mise en forme sont généralement des filtres dont l'objectif est d'augmenter le rapport signal à bruit mais surtout de rendre le signal exploitable en éliminant les variations de ligne de base, le bruit voire même en modifiant radicalement la forme des impulsions. Si le filtrage (anti-bruit, passe-haut et passe-bas) est connu dans la théorie du traitement du signal, d'autres algorithmes sont particulièrement dédiés au traitement d'un signal du mode impulsion. Dans cette partie, nous présentons deux algorithmes. Un algorithme de correction de ligne de base (BaseLine Restauration (BLR)) dont l'objectif est de supprimer la composante continue mais surtout de lutter contre les dérives de celle-ci. Le second algorithme présenté est un filtre trapézoïdal qui est un filtre pondéré dont l'objectif est de transformer les impulsions en trapèze de manière à faciliter l'extraction de l'amplitude en ajustant la taille (et donc le nombre d'échantillons) du plateau du trapèze.

#### 1.3.2.1 Correction de la ligne de base

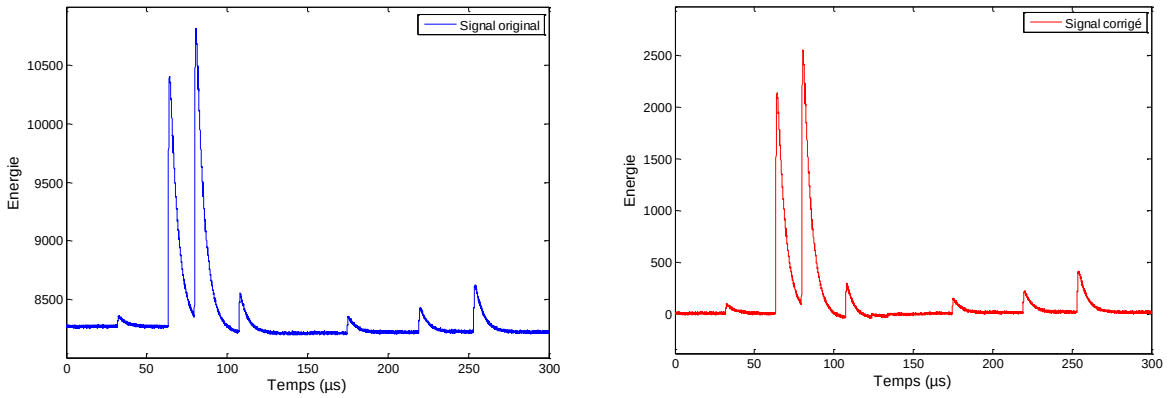
La correction de ligne de base permet de pallier les variations de la composante continue générées par un couplage AC. Le principe est de garder la composante continue du signal autour de zéro. Le BLR proposé est une version similaire à (Faisal et al. 2013) et se décompose en deux états :

Premier état :

Dans cet état, si aucune impulsion n'est détectée au travers d'un premier seuil, une valeur de référence est calculée par la moyenne des n échantillons précédents. Cette valeur est ensuite soustraite aux données d'entrées.

Deuxième état :

Il se produit quand une impulsion est détectée (au-dessus d'un seuil fixe paramétré). La ligne de base n'est pas recalculée dans cet état. La valeur de référence reste la dernière calculée lors de l'état 1. Cette valeur est ensuite soustraite aux données d'entrées. L'algorithme revient au premier état lorsque la valeur des données d'entrées devient inférieure à la référence calculée dans l'état 1 (seuil dynamique).




(a) (b)

**Figure 1-19: Illustration d'une correction de ligne de base avec (a) une fraction du signal original et (b) le signal corrigé**

La Figure 1-19 illustre le résultat de cet algorithme. On constate que la ligne de base qui dérive dans (a) est repositionnée dans (b) autour de 0 sans variation apparente. Cette correction facilite l'extraction de la valeur d'amplitude sans normalisation. Cependant, comme cela est montré dans (Moline et al. 2015), ce traitement affecte grandement la forme des impulsions et donc la résolution finale d'un spectre en énergie. En effet, une partie des impulsions se situant au-dessous de la moyenne est tronquée, ce qui raccourcit les impulsions se situant au-dessus de cette moyenne. Cela est relatif au seuil de déclenchement qui est fixé de manière à ne jamais déclencher sur le bruit et qui est fixe.

### 1.3.2.2 Filtre trapézoïdal

Le filtre trapézoïdal est un filtre numérique invariant dans le temps (Réponse Impulsionnelle Finie (FIR)) permettant la transformation des impulsions vers une forme trapézoïdale/triangulaire. L'algorithme est conçu pour convoluer un signal à décroissance exponentielle avec des fonctions rectangulaires et rampes tronquées (Jordanov et al. 1994).

Il existe trois paramètres de mise en forme de l'algorithme $k$, $l$ et $m$. $k$ représente le temps de montée de l'impulsion trapézoïdale et $l$ est la somme du temps de montée et du sommet plat. La largeur de l'impulsion trapézoïdale est donnée par $k + l$. $m$ est la constante de temps (résolution temporelle $\tau$) du signal d'entrée et est relative à la forme de l'impulsion trapézoïdale. Lorsque $m$ est égale à la constante de temps du signal d'entrée, une impulsion trapézoïdale symétrique peut être obtenue (Zhou et al. 2015).

Le filtre doit donc générer un signal de sortie avec un sommet plat d'une durée choisie pour obtenir le moins de variations possibles (le haut du trapèze doit être aussi plat que possible). Enfin, la montée et la descente du signal de sortie sont choisies arbitrairement pour répondre à d'autres besoins, comme par exemple la diminution des empilements.

L'équation 1-6, définit la fonction du filtre trapézoïdal étudié.

$$s(n) = \sum_{i=0}^{n} \sum_{j=0}^{i} d^{k,l}(j) + d^{k,l}(i)M \qquad \textbf{1-6}$$

Avec $d^{k,l}(j)$ définie dans l'équation 1-7 pour $v(j)$ le signal original, et $M$ définie dans l'équation 1-8 pour $\tau$ la constante de décroissance moyenne des impulsions et $T_{clk}$ la période entre deux échantillons.

$$d^{k,l}(j) = v(j) - v(j-k) - v(j-l) + v(j-k-l) \quad \textbf{1-7}$$

$$M = \frac{1}{e^{T_{clk}/\tau} - 1} \quad \textbf{1-8}$$

La Figure 1-20 illustre le résultat du filtre paramétré pour l'obtention d'un sommet plat le plus large possible et la Figure 1-21 illustre le résultat du filtre paramétré pour réduire la largeur des impulsions afin de diminuer le risque de traiter des empilements.

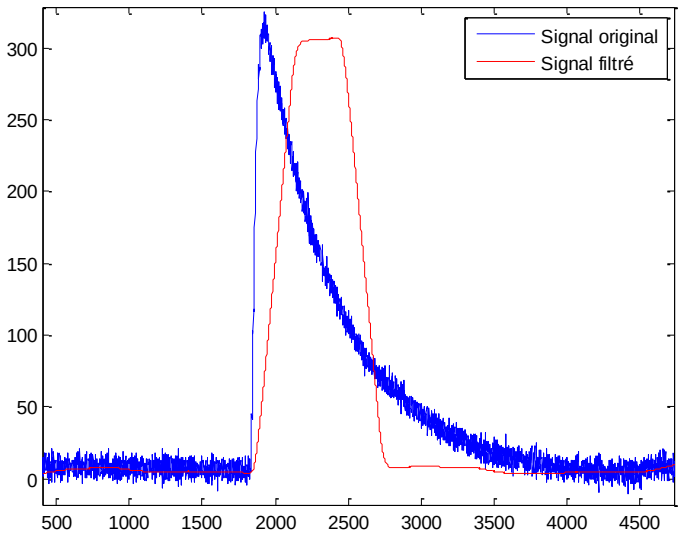



**Figure 1-20 : Résultat du filtre trapézoïdal configuré pour avoir un sommet plat le plus large possible.**

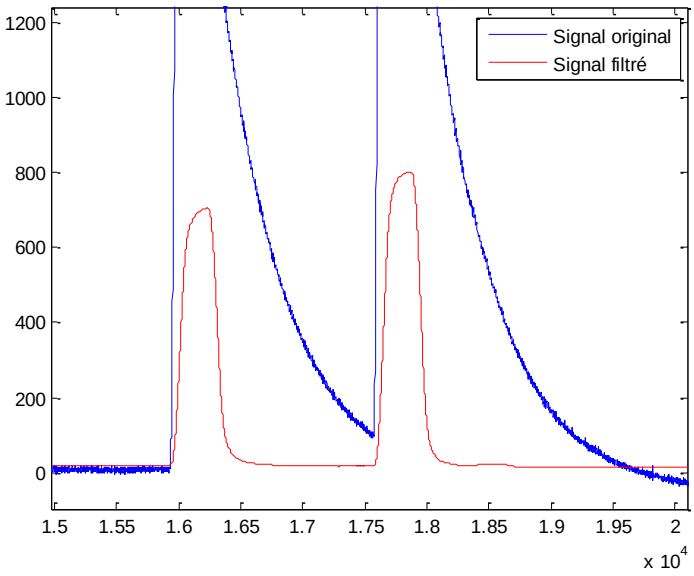



**Figure 1-21 : Résultat du filtre trapézoïdal configurer pour diminuer le risque d'empilements.**

La Figure 1-20 et la Figure 1-21 illustrent également la notion de compromis apportée par le filtre. Car s'il est possible de réduire suffisamment la durée des impulsions pour diminuer le risque d'empilements, cela ne permet pas de configurer idéalement le filtre pour obtenir un sommet plat suffisamment large. Durant les essais, on constate que les filtres à paramètres fixes comme le filtre trapézoïdal ne permettent pas de faire face à la grande dispersion des tailles d'impulsions, puisqu'ils s'ajustent sur une impulsion moyenne.

On distingue cependant deux avantages. Le premier permet de faire face au déficit balistique. Ce déficit peut engendrer des formes différentes pour deux impulsions de même charge. Ces différences de forme peuvent s'expliquer par le fait que les charges d'ionisation sont créées dans des zones différentes du détecteur. Le second permet d'être plus précis sur la recherche de l'amplitude maximum d'une impulsion. En effet, le plateau du trapèze (si les réglages sont bons) offre une meilleure précision temporelle sur la recherche de l'amplitude maximale. C'est encore plus le cas sur un signal numérisé, où la valeur de l'échantillon maximum dépend de la période d'échantillonnage.

Si l'utilisation de filtres de mise en forme permet d'améliorer la qualité de l'extraction d'énergie d'une impulsion, elle détériore cependant une partie du signal original (les impulsions qui s'éloignent trop de l'impulsion type ayant servie au paramétrage) biaisant ainsi la mesure. L'utilisation d'un algorithme de filtrage ajustant dynamiquement ses paramètres en fonction du signal semble être une solution intéressante. Cependant la littérature de l'instrumentation nucléaire ne propose pas ce type d'algorithme.

### 1.3.3 Algorithmes d'extraction de l'information d'intérêt

Les algorithmes d'extraction de l'information d'intérêt, comme leur nom l'indique, permettent d'obtenir selon les applications un résultat sur l'énergie de l'impulsion ou sur sa forme. Nous passons rapidement sur les algorithmes permettant l'extraction de l'énergie car ils sont relativement peu complexes. Ils se limitent à la recherche d'un maximum sur l'ensemble des valeurs d'une impulsion ou à un calcul d'aire sur cet ensemble. La complexité se situe dans les algorithmes de discrimination de forme des impulsions (PSD), notamment pour la discrimination n-$\gamma$.

Il existe de nombreux algorithmes de PSD (Siddavatam 2014). Nous ne présentons ici que les plus utilisés dans les instruments de mesure électroniques en ligne (c'est-à-dire en temps réel). Dans le domaine temporel nous avons abordé la comparaison de charge dans la section 1.2.2. Nous choisissons donc de présenter un algorithme appelé analyse de gradient (« Pulse Gradient Analysis », PGA). Il s'agit d'une méthode de discrimination non-linéaire fondée sur la comparaison de l'amplitude du maximum de l'impulsion à l'amplitude correspondant à un temps donné. La pente entre l'amplitude du maximum et l'amplitude de discrimination est ensuite calculée (Liu et al. 2010). L'équation 1-9 représente le calcul du gradient avec : $p$ le gradient dans le domaine temporel, $y_p$ l'amplitude de crête, $y_d$ l'amplitude de la discrimination et $t_p$ , $t_d$ respectivement les dates de ces amplitudes.

$$p = \frac{y_p - y_d}{t_p - t_d} \qquad \textbf{1-9}$$

Le temps dit « de discrimination » est choisi en fonction des caractéristiques du scintillateur ainsi que du photomultiplicateur permettant ainsi de séparer les événements de type neutron de ceux de type photon en fonction de leur constante de décroissance. La Figure 1-22 illustre ce principe.

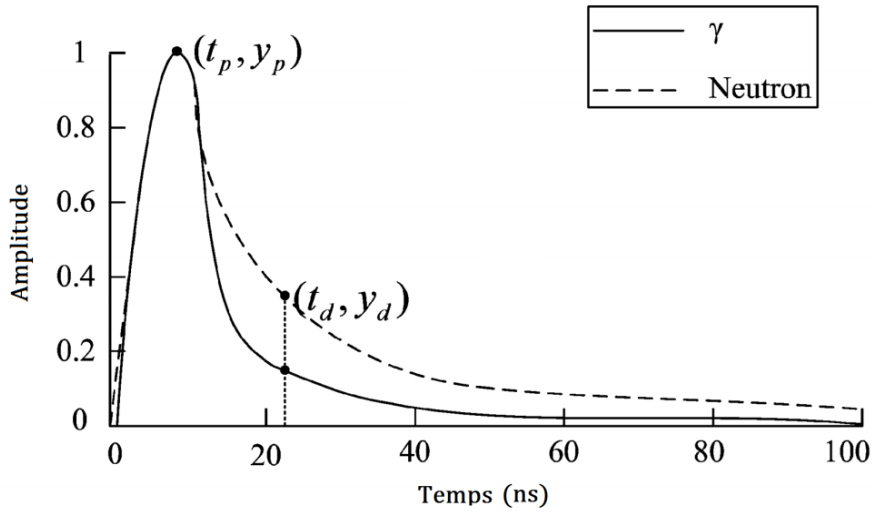



**Figure 1-22 : PGA appliquée à des impulsions neutron et gamma.**

La comparaison de charge ou l'analyse du gradient sont des techniques relativement simples à implémenter en ligne. D'autres algorithmes de PSD plus complexes existent mais ne sont jamais implémentés en ligne. On peut citer des algorithmes travaillant dans le domaine de Fourier (Liu et al. 2013), à base d'ondelettes (Yousefi et al. 2008) ou même à base de réseau de neurones (Xiaohui et al. 2013).

### 1.3.4 Conclusion

L'analyse des algorithmes utilisés a fait apparaitre que la majorité de ceux utilisés est conçue et optimisée pour être implémentée en ligne, de par leur nature à pouvoir travailler échantillon par échantillon sur un flux de données. Cependant, les algorithmes les plus complexes ou nécessitant d'interrompre le flux (mémoriser l'impulsion avant de la traiter) ne sont jamais implémentés en ligne. C'est notamment le cas des algorithmes de PSD les plus complexes. Deux raisons peuvent être mises en avant pour expliquer cette non-utilisation. La première est l'absence de chaîne de mesure programmable permettant d'implémenter et tester en ligne de nouveaux algorithmes. La seconde est la complexité de ces algorithmes qui demandent des prérequis en performances plus élevées pour ne pas souffrir de temps mort. En effet, s'il existe dans la littérature des algorithmes permettant l'analyse des impulsions dans le domaine de Fourier ou des ondelettes, ces algorithmes ont la particularité de devoir mémoriser l'impulsion totalement (attendre le dernier échantillon) avant de la transformer. Cette mémorisation, s'il elle n'est pas gérée, implique au minimum un temps mort équivalent à la durée de l'impulsion. Ces deux arguments sont corrélables compte-tenu des différences de performances entre des composants programmables et des circuits dédiés sur ASIC ou FPGA par exemple. Nous verrons dans la partie 1.5 que ce sont en partie les caractéristiques du signal qui imposent le choix de ces derniers. En conclusion, si les algorithmes réellement implémentés en ligne sont peu complexes afin de pouvoir travailler en flux, alors ce sont les détecteurs et leurs signaux qui imposent le dimensionnement de l'architecture de calcul et qui doivent donc être analysés.

## 1.4 Analyse des détecteurs

Cette partie analyse les différents types de détecteurs de l'instrumentation nucléaire afin de définir les paramètres essentiels permettant de caractériser les qualités d'un détecteur. Cette étude nous permettra de comprendre leur impact sur le dimensionnement d'une architecture multi-applicative et multi-détecteur. La plupart des détecteurs sont associables au mode impulsion. Cette partie s'intéresse également à la compréhension de l'impact du choix du détecteur sur la construction de la chaîne d'acquisition.

Les détecteurs se différencient par leur capacité à répondre à différentes natures ou vitesses de particules ainsi qu'à l'intensité de l'interaction. Si l'on cherche en particulier à connaître l'énergie déposée dans le détecteur à chaque interaction et à dénombrer les interactions qui y ont eu lieu. Le détecteur sera alors choisi en fonction de sa capacité à offrir le meilleur de ces deux aspects pour une application donnée. Malgré de grandes différences entre chaque détecteur utilisé dans le domaine du nucléaire, on peut leur attribuer des caractéristiques communes que sont :

- la résolution en énergie ;
- l'efficacité de détection ;
- le temps de résolution ou la résolution temporelle.

### 1.4.1 Résolution en énergie

Grâce à l'étude des applications, il a été constaté qu'un détecteur doit permettre la mesure de la distribution en énergie des radiations (spectrométrie, discrimination). Une propriété directement reliée à cette distribution est appelée résolution en énergie. La résolution en énergie est la capacité d'un détecteur à avoir une réponse la plus précise possible pour une énergie donnée. Dans le cadre de la spectrométrie cela se vérifie par la distribution du nombre d'énergies mesurées autour d'un même canal. Par exemple, dans le cadre d'une étude sur une source radioactive mono-énergétique, la Figure 1-23 illustre deux distributions en énergie différentes.

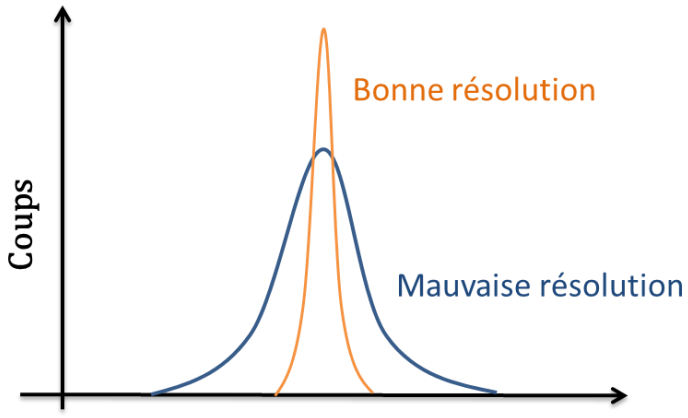



**Figure 1-23 : Exemple d'une bonne et mauvaise résolution en énergie.**

Si un nombre d'évènements envoyés au détecteur est le même, il devrait, en théorie, avoir dans un spectre des pics identiques, c'est-à-dire de même surface. Cependant, bien que chacune de ces distributions soit centrées sur la même valeur, la largeur de la courbe bleue est deux fois plus

grande que celle de la courbe orange. Cette largeur reflète qu'un grand nombre de plus grandes différences (amplitude, charge) ont été enregistrées pour chaque impulsion du détecteur, bien qu'à chaque fois la même énergie fut déposée. Si l'on veut mesurer des spectres en énergie, il est essentiel que ce signal soit, à des fluctuations statistiques près, proportionnel à l'énergie cédée par le rayonnement dans le détecteur ; cette proportionnalité (ou linéarité) devant exister sur une gamme d'énergies suffisamment étendue.

La capacité d'un détecteur à avoir une réponse la plus précise possible, et donc une « bonne » résolution en énergie est un avantage très important puisqu'il permet, pour deux énergies très proches, de les distinguer.

L'analyse de la résolution en énergie peut être réalisée grâce au calcul de largeur à mi-hauteur. La largeur à la mi-hauteur (*Full Width at Half Maximum* FWHM) représente la dispersion des informations délivrées par le détecteur soumis à une même sollicitation. La manière de chiffrer la résolution relative $R$ en énergie d'un détecteur est donc donnée en fonction du FWHM et de la position du pic $H_0$ à l'aide de l'équation 1-10.

$$R = \frac{FWHM}{H_0} \qquad \textbf{1-10}$$

La définition de la résolution en fait une grandeur sans dimension puisqu'il s'agit d'un ratio. Donc plus petit est le ratio meilleure est la résolution relative.

**Remarque :**

Si la résolution énergétique est importante pour tous les détecteurs, certains détecteurs se caractérisent également par leurs qualités naturelles de discrimination. D'une manière similaire à l'analyse de la résolution énergétique par FWHM, la qualité de la discrimination peut se calculer à l'aide d'un paramètre appelé facteur de mérite. Plus il est élevé, meilleure est la discrimination. Avec $|D_n - D_\gamma|$ la distance entre les deux pics et $W_n$ $W_\gamma$, la FWHM respective des deux pics, $M$, le facteur de mérite est obtenu à l'aide de l'équation 1-11.

$$M = \frac{|D_n - D_\gamma|}{W_n - W_\gamma} \qquad \textbf{1-11}$$

### 1.4.2 Efficacité de détection

De manière générale, la géométrie (forme, volume) d'un détecteur ne permet pas de mesurer la totalité des rayonnements émis. En effet, de manière théorique, lorsque l'on détecte des particules chargées, compte-tenu de leur parcours, il semble possible de mettre en œuvre des géométries de détection dans lesquelles pratiquement 100% des rayonnements incidents interagissent dans le détecteur (très gros volume et surface). Cependant, les impératifs expérimentaux et la nature des rayonnements à mesurer nous éloignent bien souvent de cette situation idéale. On parle alors de la notion d'efficacité de détection.

On distingue deux types d'efficacité pour les détecteurs : l'efficacité absolue $Eff_{abs}$ et l'efficacité de détection intrinsèque $Eff_{int}$.

- L’efficacité absolue (ou rendement), aussi appelée efficacité totale, est définie comme étant la fraction d’évènements enregistrés par rapport au nombre d’évènements émis par la source.
- L’efficacité intrinsèque prend en compte uniquement les signaux ayant traversé le détecteur. Il s’agit donc de la fraction d’évènements enregistrés par rapport au nombre d’évènements pénétrant dans le détecteur.

L’efficacité de détection n'impacte donc pas le dimensionnement d'une architecture multi-applicative mais met en évidence un certain besoin de flexibilité. Si pour une même expérience (même source à identifier par exemple) il est possible d’utiliser différents volumes de détection, donc avoir différentes efficacités, l'architecture aura alors un nombre d'impulsions à traiter qui va varier. Souvent, une électronique est dédiée à un détecteur, donc changer le volume de détection sans s'assurer de l'impact de l'augmentation du nombre d'impulsions sur le système (si le système accepte du temps mort) n'est pas une solution pertinente pour observer les changements d'efficacité de détection.

### 1.4.3 Résolution temporelle

La résolution temporelle d'un détecteur a été introduite dans la section 1.1. En effet, dans la présentation du temps mort nous avons introduit la notion d'impulsions de tailles variables. La durée des impulsions, caractéristique au détecteur, est appelée résolution temporelle.

Si l'on soustrait la notion de préamplification, alors chaque détecteur possède une résolution temporelle qui lui est propre. Il est possible de parler de temps mort « physique » dans le cadre de cette durée non nulle. Cependant, si l'on se réfère au cadre applicatif de la thèse, nous préférons parler d'empilement, qui est la conséquence directe de cette durée. Plus un détecteur met de temps pour « créer » une impulsion, plus la probabilité d'avoir des empilements, et donc de sortir du mode impulsion augmente.

### 1.4.4 Conclusion

Ce rappel sur les détecteurs nous a permis de mettre en évidence les caractéristiques physiques qui peuvent impacter directement les traitements et le dimensionnement d'une architecture de traitement des impulsions. Ces caractéristiques sont résumées au travers du Tableau 1-1 pour une liste non-exhaustive mais représentative des détecteurs associés au mode impulsion.

Le Tableau 1-1 présente trois caractéristiques pour quatre classes de détecteur. A chaque classe de détecteur est associée une résolution temporelle qui varie en fonction de l'état de l'art de la classe de détecteur associée. La durée des impulsions peut varier de quelque ns à plusieurs dizaines de µs. Cette donnée est extrapolée vers le taux d'impulsion maximum atteignable avant de sortir du mode impulsion. Il n'y a pas de limite mathématique entre ces deux modes, c'est pourquoi nous choisissons arbitrairement que le pire cas est un signal composé d'un train d'impulsions qui se suivent sans aucun délai entre elles. Ce cas est à la limite du mode empilement et permet de fixer une borne haute en termes de taux d'impulsion entrant. La dernière colonne présente la résolution en énergie des détecteurs et sur quelle plage énergétique le détecteur peut fonctionner. Les résolutions affichées sont les meilleurs possibles, sans tenir compte de la non-linéarité de celles-ci sur les plages en énergie présentées. Cette résolution varie d'une précision de la centaine eV à quelques dizaines de keV.

Les différences entre les détecteurs pour les trois caractéristiques présentées mettent en évidence la notion de compromis dans le choix du détecteur pour une application donnée. En effet, si l'on soustrait la nature du détecteur qui, en général, est conçu pour une application qui lui est dédiée, on constate que meilleure est la résolution énergétique d'un détecteur plus faible est sa résolution temporelle. Il n'existe donc pas de détecteur parfait pour toute la gamme d'application de l'instrumentation nucléaire. Un détecteur avec une résolution temporelle poussée sera plus utile pour une application de comptage alors qu'à résolution énergétique élevée il sera plus utile pour une application de spectrométrie.

| Classe de détecteur | Résolution temporelle | Taux d'impulsion maximum toléré par le mode impulsion | Résolution énergétique (non-linéarité comprise) |
|---|---|---|---|
| **Gazeux** | $1 - 100\ \mu s$ | $10^3 - 10^5\ i/s$ | $10\ eV - 7\ MeV$ $\pm 2 - 50\ keV$ |
| **Semi-conducteur** | $10\ ns - 100\ \mu s$ | $10^3 - 10^7\ i/s$ | $3\ keV - 10\ MeV$ $\pm 100 - 300\ eV$ |
| **Scintillateur organique** | $2 - 10\ ns$ | $10^7 - 10^8\ i/s$ | $0 - 4\ MeV$ $\pm 1 - 40\ keV$ |
| **Scintillateur inorganique** | $0{,}3 - 1{,}1\ \mu s$ | $10^5 - 10^6\ i/s$ | $0 - 4\ MeV$ $\pm 1 - 40\ keV$ |

**Tableau 1-1 : Caractéristiques physiques du signal imposé par les détecteurs de l'instrumentation nucléaire**

**Références associées au Tableau 1-1 :**

- Gaz : Chambre d'ionisation / Compteur proportionnel / Compteur Geiger-Müller (Julin 2011; Knoll 2010)
- Semi-conducteurs : Germanium Ge [HP] / Silicium Si (Canberra 2012; Knoll 2010; Mihailescu et al. 2007)
- Scintillateurs organiques : Liquide Toluène–PPO / Plastique Polystyrène / Plastique Benzène (Moszynski et al. 1992; Moszynski et al. 2008; Knoll 2010; Lynch 1968)
- Scintillateurs inorganiques : Na I (TI) / Cs I (TI) (Knoll 2010; Moszynski et al. 2008)

## 1.5 Analyse des besoins en capacité de calcul de l'architecture

Cette partie synthétise les analyses des applications et des détecteurs pour obtenir les pré-requis en performance imposés à une architecture de traitement des impulsions se voulant multi-applicative et donc multi-détecteur. Dans un premier temps, sont présentées les caractéristiques du signal obtenu par l'analyse des détecteurs. La complexité des traitements obtenue par l'implémentation des algorithmes est ensuite étudiée. Enfin, l'ensemble de ces informations est corrélé.

### 1.5.1 Prérequis en performance imposés par les détecteurs

Les prérequis en performance imposés par les détecteurs ont été obtenus de la manière suivante :

- la résolution temporelle permet l'obtention d'une fréquence d'échantillonnage minimum ;
- la résolution en énergie permet l'obtention de la précision de codage nécessaire ;
- la corrélation de ces deux résolutions permet l'obtention de la bande-passante ;

Pour rappel, nous nous plaçons toujours dans le cadre de traitements d'impulsions numériques. Ce qui signifie que de la résolution temporelle va dépendre la fréquence d'échantillonnage requise par le CAN. Dans le cadre de l'instrumentation nucléaire, le théorème de Nyquist-Shannon n'est pas toujours appliqué. En effet, si un nombre d'échantillons par impulsion plus petit que les limites imposées par Nyquist-Shannon est extrait, il est compensé par l'acquisition d'un grand nombre d'impulsions. Cela se comprend pour des applications de comptage par exemple. Dans cette analyse nous choisissons d'imposer un minimum de dix échantillons par impulsion afin de calculer les fréquences d'échantillonnage associées aux détecteurs. Ce nombre est obtenu à l'aide de (Belli et al. 2013; Flaska et al. 2013) dans lesquels l'analyse est réalisée dans le cadre d'applications de discrimination neutron-gamma où la forme et l'énergie des impulsions sont importantes. Nous choisissons d'appliquer cette règle à l'ensemble des détecteurs et sans tenir compte de l'application utilisée afin d'uniformiser la métrique.

La précision de codage est calculée à partir de la résolution en énergie et de la plage énergétique de fonctionnement des détecteurs. Si cette résolution binaire définit le nombre de canaux nécessaires, alors il s'agit du ratio entre la plage énergétique et la meilleure résolution possible. Le Tableau 1-2, traduit donc le Tableau 1-1 en transformant les caractéristiques physiques/analogiques du signal en caractéristiques numériques.

| Classe de détecteur | Résolution de CAN requise | Fréquence d'éch. min (10 éch./impulsion) | Bande-passante |
|---|---|---|---|
| **Gazeux** | $8 - 13\ bits$ | $0{,}1 - 10\ Mhz$ | $1 - 130\ Mbits/s$ |
| **Semi-conducteur** | $14 - 18\ bits$ | $0.1\ Mhz - 1\ Ghz$ | $2\ Mbits\ /s - 18\ Gbits/s$ |
| **Scintillateur organique** | $7 - 14\ bits$ | $1 - 5\ Ghz$ | $7 - 70\ Gbits/s$ |
| **Scintillateur inorganique** | $7 - 14\ bits$ | $100 - 300\ Mhz$ | $1 - 4\ Gbits/s$ |

**Tableau 1-2 : Étude comparative des performances imposées par les détecteurs de l'instrumentation nucléaire.**

Le Tableau 1-2 fait apparaître une grande diversité en termes de prérequis de dimensionnement. Il permet de connaître les caractéristiques des CAN (au pire cas) à associer aux détecteurs. Cela permet de connaître raisonnablement les bornes imposées à une architecture multi-détecteur. Dans la mesure où notre proposition doit être en mesure d'exploiter la plus grande gamme de détecteurs possible, nous considérerons le pire cas (en rouge dans le Tableau 1-2). Il s'agit des scintillateurs organiques qui imposent, par voie de mesure, une fréquence d'échantillonnage d'entrée de 5 Ghz et une résolution de 14 bits, soit une bande passante approximative de 70 Gbits/s. Il s'agit du pire cas, mais l'état de l'art des CAN nous permet d'être optimistes quant à la possibilité de numériser à cette fréquence pour des résolutions atteignant 14 bits sans avoir recours à des CAN à réseaux de condensateurs commutés qui imposent du temps mort (l'impulsion est échantillonnée/mémorisée puis numérisée). Cette thèse se place uniquement dans la partie traitement numérique des impulsions. Il faut donc retenir que la flexibilité imposée à une architecture numérique multi-détecteur, nécessite qu'elle doive également fonctionner sur des bandes passantes moins étendues. Si l'obtention du

système sans temps mort est un prérequis, alors, compte tenu du pire cas, une architecture dédiée sera souvent surdimensionnée par rapport à la grande variété de détecteurs disponibles. A des débits d'échantillons aussi élevés, faire en sorte que les traitements travaillent en flux continu devient plus complexe. Cela est souligné par la corrélation entre ces résultats et l'analyse de la complexité des algorithmes.

### 1.5.2 Prérequis des besoins en ressources de calcul imposés par les traitements

Dans le modèle non-paralysable tel que décrit précédemment, le temps mort de traitement $\tau$ est directement lié à l'application exécutée par l'unité de traitement. Afin d'estimer si une ressource de calcul du commerce permet l'exécution des algorithmes d'une chaîne d'acquisition, il convient de déterminer leurs besoins en ressources de calcul pour atteindre le zéro-temps mort tout en étant en flux. Cela signifie, par exemple, pour un CAN à 100 MHz, qu'il faudra réaliser les traitements en 10 ns par échantillon quel que soit le nombre d'opérations. Associer les résultats d'analyse des détecteurs (bande passante des CAN) à la complexité en opérations par seconde imposée par les traitements permet ensuite de connaître le besoin en capacité de calcul. Cette sous-partie présente de manière comparative le nombre d'opérations par échantillon pour les algorithmes étudiés en prenant comme référence l'algorithme du domaine le plus simple qui consiste en un comparateur à seuil. Pour chaque classe d'algorithme étudiée, le nombre d'opérations flottantes ou fixes par échantillon retenu est présenté. Le résultat de cette conversion est présenté dans le Tableau 1-3 dans lequel aucune opération n'est différenciée (additions, multiplications etc. sont confondues), puisque la cible architecturale n'est pas identifiée. Nous rappelons que seuls les traitements actuellement réalisés en ligne sur des systèmes actuels sont présentés.

| **Algorithmes** | | **Opérations/Echantillon** |
|---|---|---|
| **Discrimination** | Comparateur à seuil | 1 |
| | Discriminateur à fraction constante | 5 |
| | Trigger de Schmitt | 4 |
| | Réjection d'empilement (dérivée) | 8 |
| **Mise en forme** | DC block | 6 |
| | Déconvolution à fenêtre glissante | 26 ($*$ 1) |
| | Filtre trapézoïdale | 18 |
| | Filtre semi-gaussien | 40 ($*$ 2) |
| | Restaurateur de ligne de base | 36 |
| | Filtre passe-haut | 8 |
| | Filtre médian | 14 ($*$ 3) |
| | Filtre à moyenne mobile exponentielle | 6 |
| | Filtre à moyenne glissante | 8 |
| **Estimation de l'énergie** | Intégrale de la charge | 2 |
| | Temps au-dessus du seuil | 2 |
| | Amplitude max | 3 |
| | Charge totale/charge décalée | 8 |

**Tableau 1-3 : Représentation de différents algorithmes du mode impulsion en termes de nombre d'opérations par échantillon requis pour leur exécution (la taille des fenêtres utilisées dans les algorithmes *1, *2 et *3 est adaptée pour des impulsions de 10 échantillons minimum).**

Le Tableau 1-3 met en évidence deux caractéristiques principales des algorithmes du mode impulsions. Premièrement, les algorithmes sont peu complexes ; seuls les algorithmes dépendant de la taille des impulsions atteignent des besoins en capacité de calcul élevés (jusqu'à 40 Op/éch pour le filtre semi-gaussien pour une impulsion de dix échantillons seulement). Mais cette complexité est très variable, allant d'une simple comparaison à de la déconvolution sur plusieurs échantillons. On constate aussi que les plus hautes complexités se situent dans la catégorie mise en forme du signal. Dans cet exemple, le rapport du nombre d'opérations à exécuter pour un traitement va donc de 1 à 40.

La partie suivante s'intéresse à borner un système multi-applicatif en termes de capacité de calcul en corrélant les informations extraites des algorithmes et des détecteurs pour des cas d'applications classiques.

### 1.5.3 Corrélation des prérequis en besoins de ressources de calcul imposés par les traitements et les détecteurs

Cette partie vise donc à observer les capacités de calcul théorique requises pour implémenter en ligne les traitements étudiés sur des composants programmables en fonction du signal fourni par les détecteurs.

Comme chaque association de traitement dépend d'une application particulière, nous décidons d'appliquer les résultats de caractérisation à un modèle analytique de chaîne typique de spectrométrie et de discrimination n-$\gamma$. Ces modèles se composent d'un détecteur, d'un étage de préamplification d'un CAN et d'un processeur. Les traitements doivent être réalisés en flux sur les échantillons de manière à ne pas générer de temps mort. Pour cette analyse, les traitements sont exécutés sur un processeur généraliste. Ce processeur est directement relié en sortie d'un front end composé d'un dispositif de détection et de préamplification et d'un CAN. Le processeur exécute un programme de spectrométrie ou de discrimination neutron-gamma de manière séquentielle et doit permettre de travailler en flot de données afin d'éviter le temps mort. Cela implique que ses performances sont dimensionnées par le signal lui-même, le débit d'arrivée des échantillons et la complexité de l'application de spectrométrie exécutée. Le système analytique utilisé est illustré en Figure 1-24.

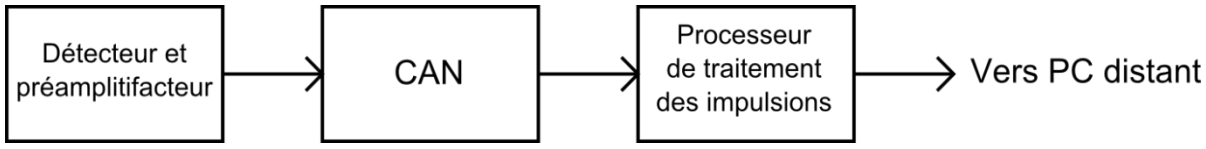



**Figure 1-24 : Schéma bloc du système programmable de traitement des impulsions analysé**

La métrique choisie est le nombre d'opérations par échantillon que l'on ramènera au nombre d'opérations par seconde (op/s) grâce au taux d'impulsions détectées. Pour simplifier le modèle tout en étant indépendant de la cible architecturale, aucune différence ne sera faite entre les instructions utilisées pour les opérations.

Pour le modèle de spectrométrie nous choisissons d'utiliser un signal en sortie d'un dispositif composé d'un détecteur cristal à scintillation de type CsI(TI) (Moszynski et al. 2008) suivi d'un photomultiplicateur (temps de résolution total de $1\mu s$) et d'un ADC cadencé à $100\ Mhz$ pour une résolution de $16\ bits$. La chaîne de traitement est composée d'un trigger de Schmitt, d'un filtre trapézoïdal et d'une recherche d'amplitude maximum. Le débit d'arrivée des impulsions est de

$10^5\ cps$. Pour le modèle de discrimination, nous choisissons un détecteur liquide à scintillation de type Toluène–PPO (Lynch 1968) suivi d'un photomultiplicateur (résolution temporelle totale $5\ ns$) et d'un ADC cadencé à $2\ Ghz$ pour une résolution de $12\ bits$. Le débit d'arrivée des impulsions est de $10^7\ cps$. La chaîne de traitement est composée d'un CFD et du calcul charge totale/charge décalé. Les Figure 1-25 et Figure 1-26 montrent les capacités de calcul qui doivent être atteintes par le processeur pour respecter le zéro-temps mort de ces deux modèles.

La Figure 1-25 et la Figure 1-26 montrent que pour ces deux cas applicatifs concrets, il y a une grande variabilité en termes de dimensionnements possibles d'un processeur pouvant être capable de travailler en flux sur les échantillons pour être zéro-temps mort. Par exemple, pour des traitements peu coûteux, un processeur de DPP affecté à une chaîne de discrimination n-γ peut atteindre une capacité de calcul de $28\ Gop/s$ alors qu'un processeur affecté à de la spectrométrie et avec des traitements plus complexes devra « seulement » avoir une capacité de calcul de $2.5\ Gop/s$. Ces chiffres prennent en compte que le processeur a des traitements variables à cause de l'étage de discrimination. En termes de traitement, ils correspondent au pire cas que le processeur doit exécuter pour s'assurer d'être zéro-temps mort.

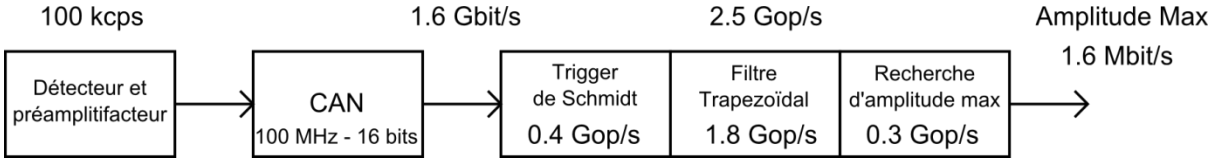



**Figure 1-25 : Schéma bloc du système programmable de spectrométrie analysé**

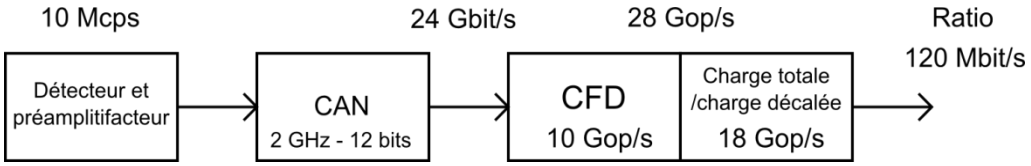



**Figure 1-26 : Schéma bloc du système programmable de discrimination n-γ analysé**

Ce n'est donc pas la complexité des traitements qui va dimensionner le choix du processeur mais bien le débit d'échantillons. De plus, si l'accès au programmable a pour objectif d'étendre les applications traditionnelles vers des applications jusque-là exécutées hors ligne (Fourier, ondelette) ces besoins en capacité de calculs peuvent augmenter considérablement.

Cette première analyse fait apparaître des besoins élevés en termes de capacité de calcul pour une seule voie de mesure mais pose surtout le problème du débit de données entrant. En effet, si des processeurs actuels sont capables d'atteindre des capacités de calcul de l'ordre de plusieurs dizaines de Gop/s, ce qui serait suffisant pour un grand nombre d'applications du mode impulsion, l'accès aux données, lui, est limitant. Dans la littérature, pour l'accès aux données, on distingue deux types d'architectures. Le premier concerne les processeurs généralistes qui disposent d'une hiérarchie mémoire de plusieurs niveaux de cache et d'une RAM auxquelles le processeur peut accéder de différentes manières. Afin de rendre possible l'exécution des traitements durant la lecture des données, le processeur utilise un accès direct à la mémoire (Direct Memory Acces (DMA)) qui permet de récupérer les données d'entrées directement en mémoire sans intervention du processeur.

Cependant, ce mode de fonctionnement se fait par interruption et par paquet de données. De ce fait, garantir la lecture en flux des échantillons aux débits affichés par le système de discrimination n-$\gamma$ analysé semble difficile.

Il existe d'autres architectures dédiées aux traitements en flux (Thevenin et al. 2014), mais le design est contraint directement par l'application ; elles sont donc dédiées. De plus, ce type d'architectures ne garantit pas la flexibilité recherchée et offerte par les processeurs généralistes qui ne requièrent aucune compétence spécifique de compilation ou de parallélisme par l'utilisateur. Si l'utilisation de processeur de traitement vidéo semble dimensionnée pour travailler à ce niveau de complexité, leurs architectures sont cependant adaptées pour des signaux 2D (images) et leurs capacités de calcul sont directement corrélables au haut taux de parallélisme qu'ils peuvent atteindre. En revanche, le débit de données en sortie des traitements est directement relié au nombre d'impulsions entrantes et l'étude des applications montre que le nombre de données à extraire par impulsion ne dépasse pas la résolution en énergie maximum du CAN ainsi que l'éventuel ajout d'une date par impulsion. Par exemple, dans les deux cas d'application étudiés le débit ne dépasse pas 120 Mbit/s.

### 1.5.4 Conclusion

Corrélées à l'étude des détecteurs, les contraintes applicatives laissent apparaître un besoin important mais surtout variable en termes de capacité de calcul pour ne pas avoir de temps mort. Ces besoins ne permettent pas l'utilisation classique de composants programmables du marché en tenant compte de la difficulté d'acheminer un tel débit d'informations à ces processeurs. De plus, si l'on souhaite être programmable à des débits de données très élevés et compte-tenu des possibilités algorithmiques présentées, on s'aperçoit qu'une solution pour la gestion du temps mort doit être trouvée. La solution ne peut donc pas se limiter à un simple choix de processeur suffisamment performant et une mise en œuvre d'une solution pour l'acheminement des données. Une telle solution serait surdimensionnée et ne serait pas suffisamment flexible pour s'adapter à la grande variabilité d'applications et de détecteurs et surtout au nombre de voies de mesures.

Des solutions visant à répondre aux problèmes de dimensionnement, de gestion de temps mort, et de traitement de données pour l'instrumentation nucléaire doivent donc être trouvées. La section 1.6 décrit les principaux travaux de la littérature adressant ces problématiques. Les solutions qui sont généralement retenues pour répondre à ces contraintes sont analysées et font l'objet d'une discussion permettant de proposer un nouveau modèle d'exécution intégrant l'ensemble des contraintes définies tout au long de ce chapitre.

## 1.6 Etat de l'art des chaînes de traitement numérique des impulsions

Dans cette partie, l'analyse de l'état de l'art des chaînes de mesures du mode impulsion est effectuée. Compte-tenu du contexte applicatif de cette thèse, les architectures sont nombreuses, mais en général dédiées à une application. Au premier regard, on constate que la majorité des architectures est dédiée à un détecteur et à une application donnés. Donc, d'une manière générale la gestion du temps mort est rigide et est traitée soit par l'utilisation de composants suffisamment performants pour l'application et le signal, soit par l'adaptation des traitements de manière à réduire leur durée d'exécution pour ne pas avoir de temps mort.

Cet état de l'art présente les architectures qui sont, soit programmables, soit qui se montrent suffisamment flexibles pour répondre aux besoins applicatifs du mode impulsion. Ces besoins ont été décrits dans les parties précédentes et peuvent être résumés par trois critères : la programmabilité, la gestion du multivoie ainsi qu'une gestion avancée du temps mort par l'utilisateur lorsque les traitements sont modifiés. L'utilisation de ces critères nous a permis de trier et d'extraire les architectures proposant des solutions ré-exploitables pour l'architecture souhaitée.

Si les circuits analogiques sont traditionnellement utilisés pour répondre aux contraintes du temps mort, ils sont petit à petit remplacés par des circuits numériques. L'état de l'art des architectures de traitement numérique des impulsions est présenté par ordre chronologique puisque cela fait apparaître les changements d'approches utilisées en fonction de l'évolution des composants électroniques du marché.

Les travaux (J. M. R. Cardoso et al. 1999) présentent une architecture dédiée à la spectrométrie qui propose d'associer un processeur du traitement du signal (DSP) avec une hiérarchie mémoire à deux niveaux. Une vue globale de l'architecture est présentée en Figure 1-27. La hiérarchie mémoire est présentée en Figure 1-28. Le premier niveau mémorise le signal en sortie de CAN puis transmet les données dans un second niveau mémoire lié au DSP. Le DSP repère directement les impulsions en mémoire grâce à leur position déterminée par un système de déclenchement analogique dédié. Le DSP dispose d'un DMA qui permet de s'affranchir du non-déterminisme de l'arrivée des impulsions tout en réduisant le temps mort du système. De plus, la flexibilité est apportée par le DSP qui est programmable.

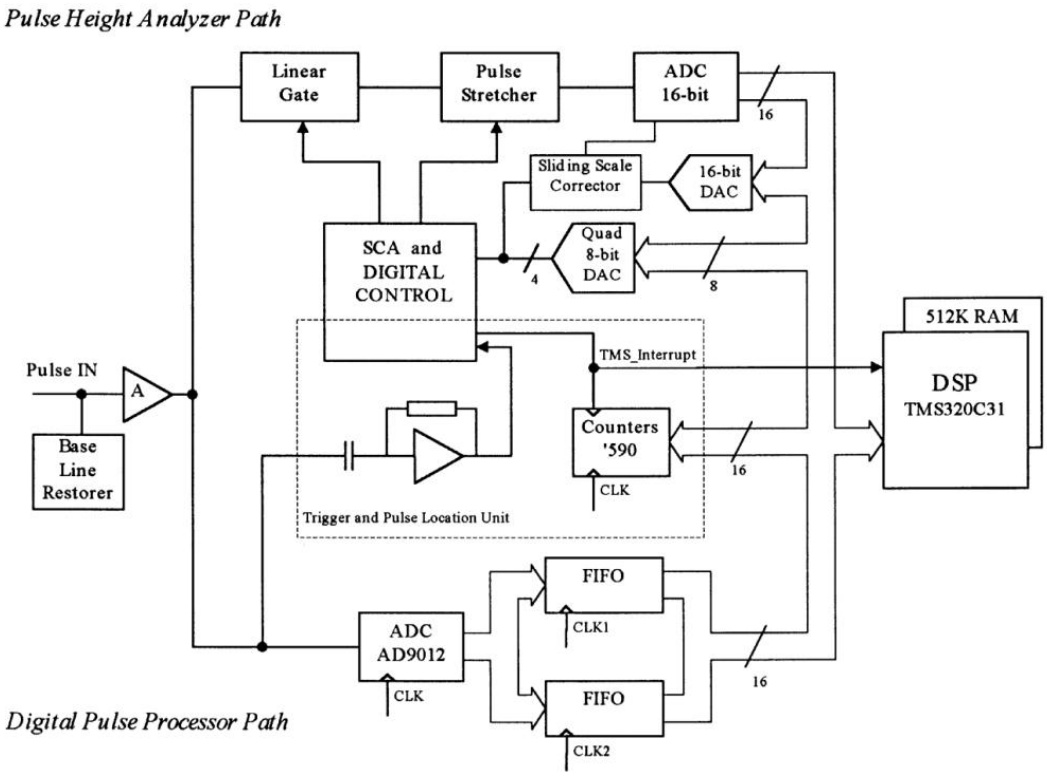



**Figure 1-27 : Vue globale de l'architecture hybride de traitement des impulsions de Cardoso.**

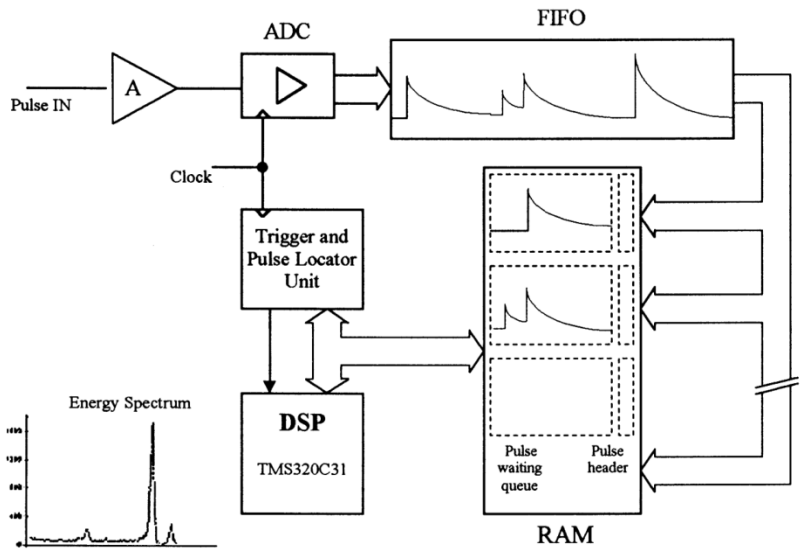



**Figure 1-28 : Gestion de la mémoire du DPP de Cardoso.**

Cette architecture a été améliorée dans (Cardoso et al. 2004a) grâce à l'ajout de plusieurs tuiles de calcul appelées Slave Unit (SU) contenant chacune le second niveau de mémorisation et un DSP. Une vue globale de l'architecture est présentée en Figure 1-29. Un DSP maître s'occupe de la communication avec le déclencheur pour le repérage des impulsions dans le premier niveau de mémoire. Celui-ci gère ensuite la distribution des portions du signal contenant des impulsions sur un bus partagé entre N SU. Si un nombre suffisant de SUs est présent, le temps mort tend vers zéro puisque toutes les impulsions peuvent être théoriquement traitées. Si cette approche permet de répondre à la problématique d'acheminement des données vers un processeur dont la durée des traitements est non-déterministe, elle n'est pas en mesure de gérer plusieurs voies de mesure ni de les corréler entre elles.

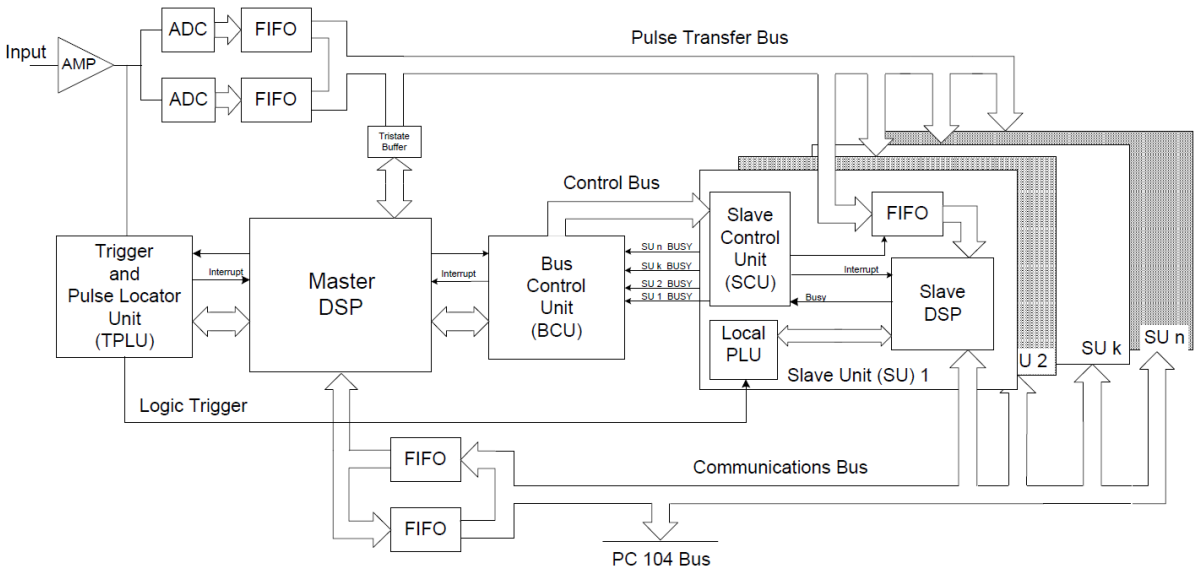



**Figure 1-29 : Vue globale de l'architecture multiprocesseur de traitement des impulsions de Cardoso**

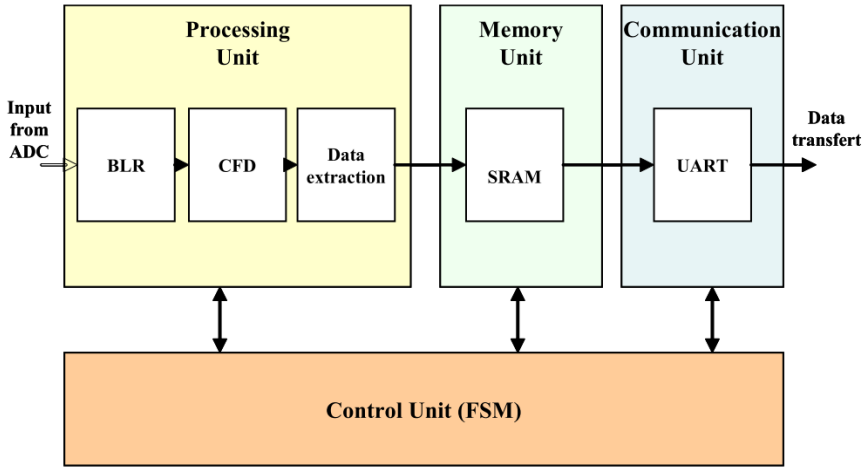



**Figure 1-30 : Schéma bloc de la configuration du FPGA de la carte PING.**

Des travaux plus récents (Normand et al. 2009; Normand et al. 2012) présentent une architecture reconfigurable sur FPGA. Un schéma bloc du *firmware* implémenté sur celui-ci est présenté en Figure 1-30 . Cette architecture est capable de réaliser une partie des mesures se trouvant dans le cycle de combustible du nucléaire comme le comptage neutronique et la spectrométrie. Une voie de mesure est associée à une carte de traitement, avec la possibilité de travailler individuellement puis de corréler les résultats fournis par un ensemble de cartes. Cette architecture introduit la notion de macro-pipeline en séparant les étages de traitement, réduisant ainsi le temps mort à la latence de l'étage le plus lent. Cependant, le *firmware* doit être repensé et mis à jour en cas de modification du cadre applicatif. Cette modification peut être longue, complexe et coûteuse puisqu'elle nécessite des experts en traitement du signal et en VHDL/Verilog. Les utilisateurs finaux ne peuvent donc pas intervenir directement sur les traitements. Cette contrainte est allégée par la plate-forme proposée en (CAEN 2011) qui combine un dispositif de mesures et un ensemble de *firmwares* prédéfinis permettant d'exécuter de nombreuses applications. Elle répond ainsi partiellement au besoin de flexibilité. L'architecture de (Schiffer et al. 2011; Faisal et al. 2013) répond mieux au besoin de flexibilité applicative en offrant une plateforme dotée d'un FPGA capable de réaliser du comptage, de la spectrométrie gamma, de la discrimination neutron-$\gamma$ et de la coïncidence. Un schéma bloc du *firmware* implémenté sur le FPGA est présenté en Figure 1-31. La carte dispose de quatre voies de mesure. A chaque voie est associée une tuile de calcul. Les traitements associés à chaque tuile sont implémentés indépendamment de manière à pouvoir travailler en parallèle sur le signal pour diminuer le temps mort. De plus, une interface avec un PC facilite la reconfiguration de certains éléments. L'architecture offre le choix entre deux types de déclencheurs permettant de s'adapter à différents types de signaux. Si ces travaux se rapprochent le plus d'une plateforme multi-applicative, ils n'en restent pas moins dépendants du *firmware* du FPGA. De plus, seules quatre voies de mesures sont gérées, puisque les tuiles sont conçues pour être implémentées au sein d'un même FPGA, ce qui, d'une manière générale, limite le nombre de voies à la capacité du FPGA.

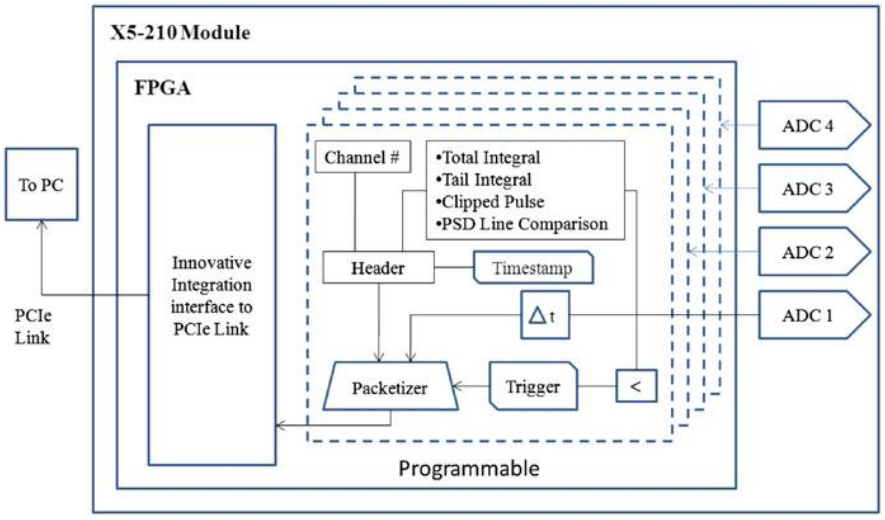



**Figure 1-31 : Architecture du FPGA du module X5-210.**

Les travaux de (Lee et al. 2012) présentent également une plateforme dotée d'un FPGA. Un schéma bloc du *firmware* implémenté sur celui-ci est présenté en Figure 1-32. Sur cette figure, on constate que le modèle d'exécution est différent puisqu'il sépare les impulsions du reste du signal en amont de tout autre traitement en se basant sur un fenêtrage temporel des impulsions. La taille des fenêtres, qui est reconfigurable, est prédéfinie pour une application donnée en fonction des caractéristiques des détecteurs. Grâce à cette méthode, la suite des traitements intégrés à la puce peut se concentrer uniquement sur les impulsions, diminuant ainsi drastiquement le temps mort. Ensuite, les impulsions sont distribuées sur deux étages de traitement fonctionnant en parallèle. Nous comprenons alors que ce modèle se rapproche de celui de Cardoso à la différence qu'il est possible d'utiliser un déclencheur plus complexe afin de simplifier le découpage du signal avant distribution. De plus, les traitements sont capables de travailler en flux sur les impulsions à l'origine d'un déclenchement. De cette manière, la hiérarchie mémoire proposée par Simoes n'est pas nécessaire. Cependant, cette solution s'éloigne de la flexibilité proposée dans (Faisal et al. 2013), de la réduction du temps mort apportée par (Cardoso et al. 2004a) et n'est ni multivoie ni programmable.

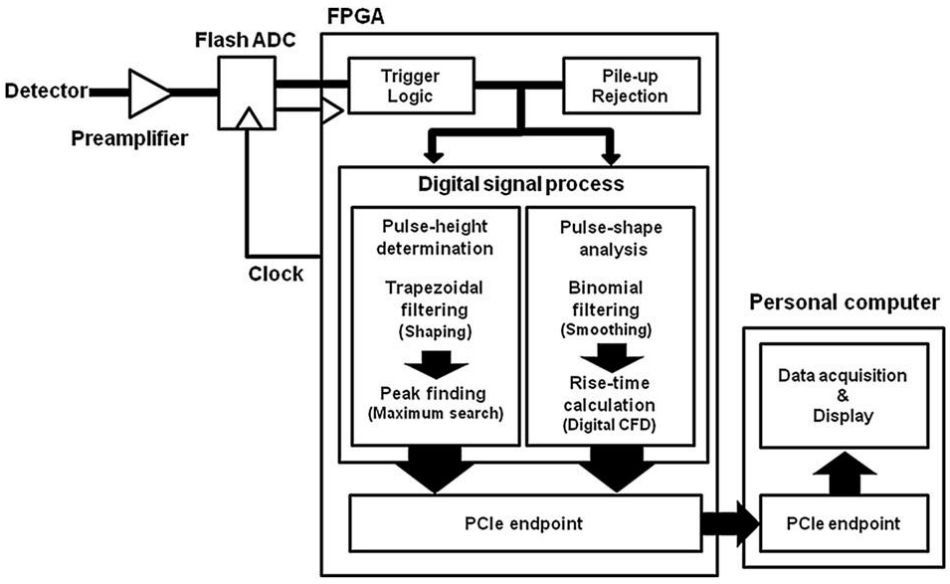



**Figure 1-32 : Schéma bloc de la configuration du FPGA de traitement numérique des impulsions de Lee.**

Le Tableau 1-4 résume l'analyse de ces travaux en fonction des critères définis au début de la section. Nous pouvons constater que, pris individuellement, les travaux présents dans la littérature ne permettent pas de répondre conjointement aux besoins de suppression du temps mort, de flexibilité ou de multivoie. Toutefois, nous retiendrons l'intérêt de constituer un pipeline afin de limiter le temps mort, ainsi que la possibilité d'exploiter une hiérarchie mémoire associée à différents éléments de calcul. Les approches proposant de séparer les impulsions du reste du signal permettent d'exploiter au mieux la capacité de calcul. Enfin, aucune des approches proposées n'est suffisamment flexible, excepté dans (Cardoso et al. 2004a), pour être en mesure de gérer une grande variété d'applications.

| Architecture | Spectrométrie | Discrimination n-γ | Coïncidence | Programmable | Multivoie | Gestion temps mort |
|---|---|---|---|---|---|---|
| **Simoes** | oui | non | non | oui | non | non |
| **Cardoso** | oui | non | non | oui | non | oui |
| **Normand** | oui | non | oui | non | oui | non |
| **CAEN** | oui | oui | non | non | non | non |
| **Schiffer** | oui | oui | oui | non | oui | non |
| **Lee** | oui | non | non | non | non | non |

**Tableau 1-4 : Tableau récapitulatif des architectures de l'état de l'art analysé vis-à-vis des contraintes imposées à une architecture multi-applicative.**

## 1.7 Conclusion

Ce chapitre a montré qu'une architecture de traitement du signal couvrant les applications du domaine de l'instrumentation nucléaire devrait être capable de répondre à trois problématiques. La première concerne la flexibilité applicative de l'architecture qui implique d'être programmable. La deuxième concerne la gestion de plusieurs voies de mesures. La troisième concerne le temps mort. L'étude des applications et des détecteurs fait apparaître des besoins en capacité de calcul élevé dans le cadre de traitements en flux. Les travaux de l'état de l'art ont montré que des architectures dédiées sont généralement mises au point pour répondre à ces problématiques. Cependant, d'autres approches modifient le modèle d'exécution par une gestion différente du flux de données permettant soit l'utilisation de composants programmables, soit la réduction des besoins en capacité de calcul de l'architecture.

Les travaux de cette thèse reposent sur la mise en place d'un modèle d'exécution pour le traitement d'impulsions. Ce modèle d'exécution doit permettre de répondre aux problématiques de programmabilité, temps mort et multivoie. Notre modèle d'exécution se base sur un module spécifique d'extraction des impulsions et sur l'utilisation d'une architecture multiprocesseur distribuée programmable. L'extracteur s'adapte aux différentes tailles et formes d'impulsions contrairement aux approches des littératures qui sont dédiées à un détecteur. Il permet de segmenter le flux avant les traitements. Les impulsions ainsi extraites sont ensuite distribuées sur une architecture de calcul spécifique sur laquelle la durée d'exécution des traitements n'est plus directement contrainte par le temps mort mais par le nombre de processeurs.

Le chapitre suivant détaille notre proposition de modèle d'exécution pour répondre aux problématiques d'une architecture multi-applicative de traitement d'impulsions. Les Chapitre 3 et Chapitre 4 décrivent chacun une partie précise de la proposition globale.

# Chapitre 2 : Modèle d'exécution de l'architecture

## 2.1 Introduction

Ce chapitre présente le modèle d'exécution d'une architecture numérique de traitement des impulsions multi-applicative répondant aux besoins définis dans le Chapitre 1. Pour rappel, ces besoins sont la programmabilité, la flexibilité en termes de dimensionnement et nombre de voies de mesure ainsi que la gestion du temps mort. Le chapitre précédent a montré que l'état de l'art offre uniquement des solutions qui répondent partiellement à ces besoins. Elles passent par la modification du modèle d'exécution classique qui consiste à traiter en flux les échantillons du signal par des traitements dédiés sur une architecture également dédiée. Pourtant, il y a deux manières d'optimiser un système de calcul. La première consiste à optimiser la vitesse d'exécution des traitements, soit par l'utilisation de composants dédiés, soit par des modifications de l'algorithmie (parallélisme). La seconde concerne la gestion du flux de données (distribution, mémoire, réseaux etc.), appelée ici modèle d'exécution. L'état de l'art montre qu'il est possible d'aboutir à un modèle d'exécution qui allège les contraintes de performances ou qui permet de rendre l'architecture flexible à ces besoins. Mais, comme le montre (Cardoso et al. 2004a), la création d'un modèle d'exécution adapté aux contraintes du signal de l'instrumentation nucléaire est une bonne solution et n'empêche pas l'optimisation des traitements ultérieurs. C'est pourquoi dans ce chapitre nous proposons un modèle d'exécution adapté d'une part aux caractéristiques non déterministes du signal et d'autre part au caractère individuel des impulsions constituant le signal.

Ce chapitre présente le modèle d'exécution que nous proposons. Il est composé de quatre étapes. Pour chacune des étapes, l'analyse porte sur les performances de calcul et la gestion du temps mort.

## 2.2 Définition du modèle d'analyse

Le modèle d'exécution proposé est analysé de deux façons. La première concerne directement les prérequis en bande passante et performance de calcul imposés pour atteindre le zéro-temps mort. La seconde s'intéresse uniquement à l'impact du modèle sur la gestion du temps mort. Pour cela, nous proposons de commencer par la définition d'un modèle dit primaire, se basant sur une chaîne standard de spectrométrie travaillant en flux, et de le faire évoluer avec notre proposition.

### 2.2.1 Modèle d'analyse des prérequis en performances

Pour cette analyse, nous utilisons une application type de spectrométrie que nous exécutons sur un processeur appelé « processeur X ». Ce processeur est directement relié en sortie d'un *front end* composé d'un dispositif de détection et de pré-amplification et d'un CAN. Le processeur exécute un programme de spectrométrie de manière séquentielle et doit permettre de travailler en flot de données (ou flux) afin d'éviter le temps mort. Cela implique que ses performances sont dimensionnées par le signal lui-même, le débit d'arrivée des échantillons et la complexité de l'application de spectrométrie exécutée. Le système analytique utilisé est illustré en Figure 2-1.

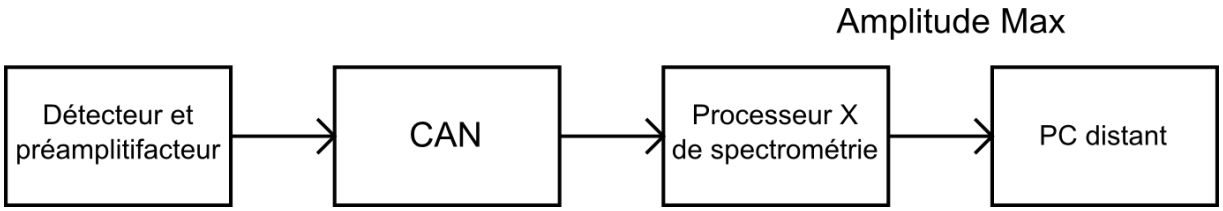



**Figure 2-1 : Schéma bloc du système analysé pour l'analyse des performances.**

Comme l'illustre la Figure 2-1, le processeur envoie le résultat de ses traitements vers un PC distant dont l'objectif est la construction et la post-analyse du spectre. Dans ce chapitre, nous analysons seulement la partie traitement numérique du signal de la chaîne. Nous considérons que la gestion du flux en sortie du processeur est, sans intervention du modèle d'exécution, une problématique inhérente au détecteur et au taux d'impulsion qu'il peut traiter. Cependant, si une impulsion est un ensemble d'échantillons qui varie de la dizaine à plusieurs milliers d'échantillons (voir Chapitre 1), alors, le débit de données de sortie sera toujours inférieur au débit d'entrée si l'information extraite ne dépasse pas la taille de la résolution binaire maximum du CAN.

#### 2.2.1.1 Application analysée

L'application traditionnelle de spectrométrie choisie pour être exécutée séquentiellement sur le processeur X est composée de quatre algorithmes. Le premier est un trigger de Schmitt (Pasquali et al. 2007), suivi d'un dispositif de rejet d'empilement basé sur une dérivée (CAEN 2011). Les impulsions à l'origine d'un déclenchement sont ensuite mise en forme à l'aide d'un filtre trapézoïdal (Radeka 1972) pour ensuite en extraire l'amplitude maximum de chacune d'elles. Dans le Chapitre 1, nous avons montré que ces algorithmes ont été développés et testés afin d'obtenir un indicateur de complexité qui est le nombre d'opérations par échantillon que l'on ramènera au nombre d'opérations par seconde. Pour simplifier le modèle tout en étant indépendant de la cible architecturale, aucune différence ne sera faite entre les instructions utilisées pour les opérations. Cette méthode permet de comparer les modèles entre eux. Pour chaque algorithme étudié, le nombre d'opérations par échantillon retenu est le suivant :

- trigger de Schmitt : 4 opérations/échantillon ;
- réjection d'empilement : 8 opérations/échantillon ;
- filtre trapézoïdal : 18 opérations/échantillon ;
- recherche d'amplitude maximum : 3 opérations/échantillon ;

#### 2.2.1.2 Signal analysé

Pour le modèle, nous choisissons d'utiliser un signal en sortie d'un dispositif composé d'un détecteur CsI(Tl) (Moszynski et al. 2008) suivi d'un photomultiplicateur et d'un ADC cadencé à 1 GHz pour une résolution de 16 bits (Texas Instruments 2015). Les impulsions numérisées ont une durée moyenne de 1 $\mu$s et le débit d'arrivée des impulsions (taux de comptage) est de $10^5$ impulsions par seconde. Ce taux d'impulsion peut être exprimé en coups par seconde (cps).

#### 2.2.1.3 Modèle de départ

En prenant en compte les caractéristiques définies dans les deux sous-parties précédentes, le système de spectrométrie retenu pour respecter le zéro-temps mort peut être modélisé par la Figure 2-2. Compte tenu des caractéristiques imposées par le modèle primaire et le signal d'entrée, le processeur X doit être capable de supporter un débit d'échantillon de 16 Gbit/s et d'atteindre 33 Gop/s pour travailler en flux sur les échantillons et donc de ne pas avoir de temps mort. Il s'agit du cas le plus défavorable. En effet, si l'on considère que le premier étage de traitement est conditionnel (le *trigger*), les traitements suivants ne seront exécutés qu'en présence de données utiles. Cependant, pour pouvoir travailler en flux, il est nécessaire de prendre en compte le cas le plus défavorable. Dans le cas contraire, le processeur ne garantit pas de traiter tous les échantillons d'une impulsion.

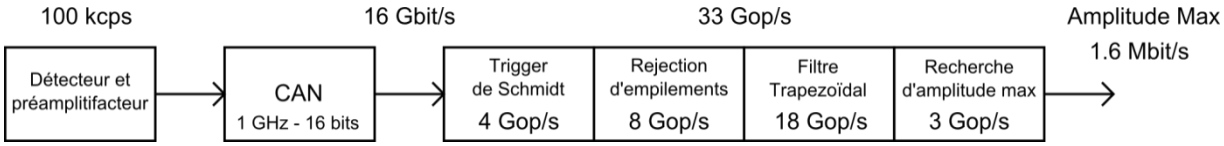



**Figure 2-2 : Analyse des performances requises d'un système sans temps mort pour le modèle d'exécution primaire.**

A partir de l'étude des applications, nous considérons que la caractéristique extraite d'une impulsion dans le dernier étage de traitement ne peut dépasser la résolution intrinsèque du détecteur et donc la résolution binaire choisie pour le CAN. C'est pourquoi nous ne faisons pas apparaître, pour le moment, la communication vers le PC distant dans cette analyse. Les parties suivantes présentent donc des solutions qui tendent à réduire ces prérequis en modifiant le modèle d'exécution.

### 2.2.2 Modèle d'analyse du temps mort

Le modèle initial décrivant le temps mort d'un dispositif de traitement des impulsions est le modèle non-paralysable décrit dans (Knoll 2010) et introduit dans le Chapitre 1. Ce modèle impose que chaque impulsion détectée par le déclencheur soit suivie d'une période de temps mort $\tau$ connue et/ou fixée à l'avance. Durant cette période, aucune autre impulsion ne peut être traitée et toutes les impulsions parvenant sur le dispositif sont donc perdues. $\tau$ est calculé en fonction de la durée maximum des traitements sur une taille d'impulsion moyenne afin de tenir les contraintes du temps réel. Si, par exemple, les traitements sont capables de travailler en flux sur l'impulsion détectée, alors la durée $\tau$ est définie en fonction de la taille moyenne des impulsions de manière à ne pas réenclencher le déclencheur avant la fin d'une impulsion. Les traitements sont donc dirigés par un délai défini à l'avance mais pas par les impulsions, ce qui empêche la saturation du système. Il est admis (Knoll 2010) que ne pas faire saturer le système offre de meilleures performances en termes de nombre d'impulsions traitées qu'un système paralysable pour de forts taux de comptage. La Figure 2-3 illustre temporellement le temps mort fixe d'un système non-paralysable pour un signal idéal. La figure montre que si la durée des traitements comprise dans $\tau$ est définie à deux fois la durée moyenne d'une impulsion, les impulsions numéro 2, 4 et 5 ne sont pas traitées et donc perdues.

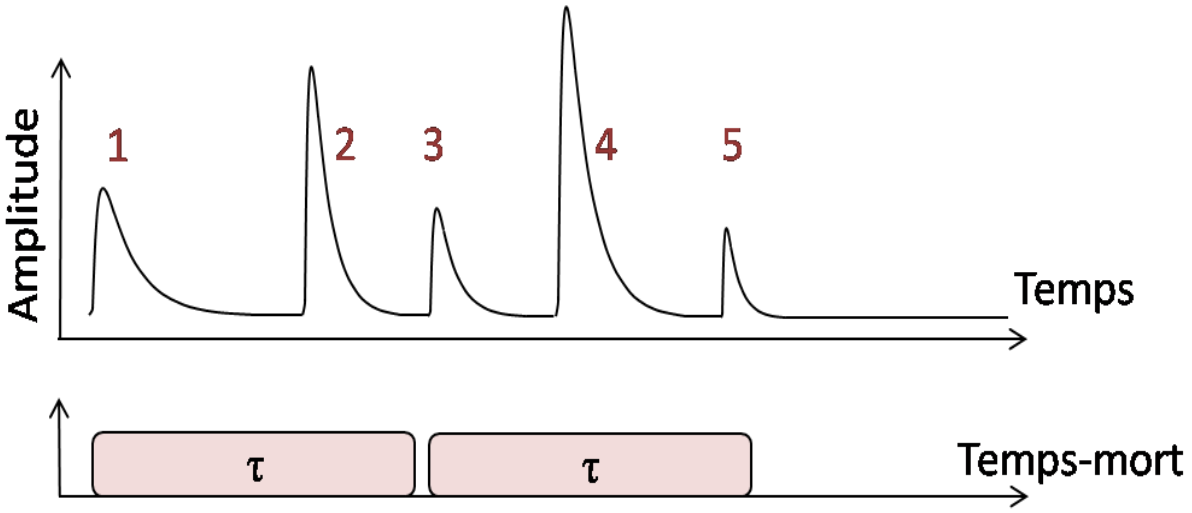



**Figure 2-3 : Illustration du temps mort fixe d'un modèle non-paralysable.**

L'impact du temps mort sur le nombre d'impulsions traitées peut être formalisé de la manière suivante. Pour un taux d'impulsions détectées $n$, le taux d'impulsions réellement traitées $m_0$ est donné par l'équation 1-3.

$$m_0 = \frac{n}{1 + n\tau} \qquad \textbf{2-1}$$

Les parties suivantes présentent des solutions pour améliorer le taux d'impulsions réellement traitées. Ces solutions passent par la modification du modèle d'exécution classique afin d'offrir une gestion du temps mort permettant de le supprimer totalement et donc d'aboutir à un nombre d'impulsions traitées égal au nombre d'impulsions détectées.

## 2.3 Séparation des impulsions, modèle dirigé par les impulsions

La première étape de notre modèle est l'extraction des impulsions en amont de tout traitement. Le principe est de séparer le signal dit « utile », les impulsions, du reste du signal. De cette manière, si l'extraction est un processus bien distinct des traitements, ceux-ci ne travailleront que sur les impulsions. Cette méthode est proposée dans (Lee et al. 2012) et utilisée dans un dispositif de spectrométrie où les impulsions sont extraites puis traitées au sein d'un même FPGA mais utilisant des processus bien distincts. Ce principe a été initialement introduit dans (Simoes et al. 1996), cependant dans ce cas ce sont des portions de signal de tailles fixes qui sont extraites et non uniquement les impulsions.

La séparation des impulsions en amont de tout traitement ajoute la notion de mémorisation. En effet, dès la détection d'une impulsion par l'étage d'extraction, il est possible de mémoriser les échantillons de celle-ci afin de bénéficier du délai potentiel entre deux impulsions, puisque leurs dates d'arrivées sont aléatoires. Cette approche permet une meilleure gestion du temps mort que le modèle primaire non-paralysable (Cardoso et al. 2004b).

### 2.3.1 Définition de l'étage d'extraction des impulsions

L'étage d'extraction des impulsions a pour objectif de réduire le débit moyen d'échantillons affectés aux traitements. Si ceux-ci ne voient et ne traitent que les impulsions, le besoin en capacité de calcul associée n'est plus régi par le débit d'échantillons mais par le débit d'impulsions entrant. Dans ce cas, la quantité générale d'échantillons à traiter diminue pourvu que le débit d'impulsions respecte la borne du mode impulsions (moins de signal utile que de signal total).

La définition de notre étage d'extraction des impulsions ne doit pas être amalgamée avec les approches traditionnelles de discrimination. L'étage proposé est un déclencheur « parfait » car il extrait tout le signal utile. Par comparaison, si le modèle de Cardoso mémorise périodiquement des portions du signal. Ces portions sont de tailles fixes et extraites périodiquement et peuvent contenir plusieurs impulsions, ou à l'inverse n'en contenir aucune. Le cas échéant, un déclencheur analogique renseigne la position des impulsions dans une portion du signal préalablement mémorisée à un processeur de traitement du signal. C'est pourquoi, avant de parler d'une architecture de traitement d'impulsions nous souhaitons nous assurer d'avoir un system de déclencheur qui respecte cette nomination en extrayant uniquement les impulsions. Par exemple, le déclencheur de (Lee et al. 2012) est conçu pour des applications de spectrométrie et n'est pas adapté à l'extraction totale des impulsions en fonction de leur taille, ce qui le rend inadapté pour des applications de discrimination de forme et de coïncidence.

Afin de s'affranchir du type de signal (taille des impulsions, forme, etc.) le dispositif d'extraction doit être capable d'extraire les impulsions entières en fonction de leur taille, et non uniquement une partie de celles-ci (comme le ferait un trigger de Schmitt par exemple). Ceci implique que ce dispositif d'extraction de l'impulsion doit être implémenté sur des composants suffisamment flexibles (donc numériques) comme dans (Lee et al. 2012). Cependant, la problématique d'utiliser un composant programmable pour travailler en flux sur des débits de données variables reste inchangée. C'est pourquoi le dispositif d'extraction proposé rejoint la partie dite dédiée du *front-end* d'acquisition car il doit respecter la contrainte temps réel du débit d'arrivée des échantillons et être capable d'absorber des flux de données pouvant dépasser la dizaine de Go/s. La littérature montre que l'utilisation de FPGA ou d'ASIC est particulièrement adaptée pour répondre à ces contraintes (Normand et al. 2009; Faisal et al. 2013; Lee et al. 2012). L'étage d'extraction des impulsions doit donc répondre aux exigences suivantes :

- s'adapter à toutes les tailles d'impulsions ;
- s'adapter au bruit du signal ;
- permettre la datation des impulsions;
- identifier les empilements (ne pas systématiquement les rejeter);
- ne jamais tronquer les impulsions (empilées ou non) ;

Cet étage d'extraction d'impulsions produit donc des paquets contenant les impulsions individuelles associées à leurs dates d'arrivée. Le signal à traiter passe donc de l'état flux à l'état « flux borné ». Cela permet à chaque application en aval de travailler sur des tableaux de données de tailles variables simplifiant l'implémentation des algorithmes associés. Le nombre de paquet à distribuer varie avec le nombre d'impulsions. Le passage d'un modèle de temps mort fixe vers un modèle de temps mort variable est alors possible, comme l'illustre la Figure 2-4. Dans cette figure, la durée des traitements est arbitrairement réglée à deux fois la durée d'une impulsion. De cette manière,

seulement les impulsions 2 et 4 ne sont pas traitées et sont donc perdues contrairement à l'exemple présenté en Figure 2-3.

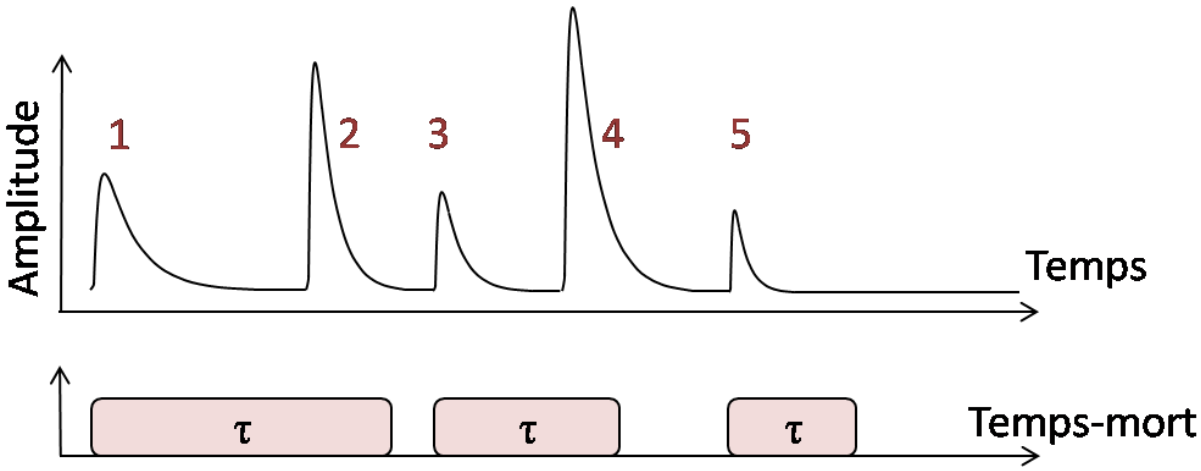



**Figure 2-4 : Illustration du temps mort variable d'un modèle non-paralysable.**

### 2.3.2 Analyse des besoins en ressources de calcul

#### 2.3.2.1 Evolution du modèle primaire vers le modèle dirigé par les impulsions

Le modèle primaire est donc réévalué vers un modèle où les traitements sont désormais dirigés par les impulsions et non plus dépendants du temps mort moyen. Il est présenté en Figure 2-5. Les prérequis de chaque algorithme sont désormais exprimés en opérations par impulsion (op/impulsion).

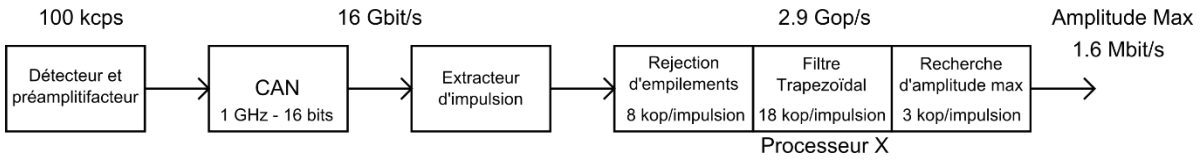



**Figure 2-5 : Analyse des besoins en ressources de calcul requis d'un système sans temps mort pour le modèle d'exécution dirigé par les impulsions**

Si l'on réévalue les besoins en ressources de calcul du modèle en fonction du taux d'impulsions entrant, on constate une diminution des performances requises. En effet, dans le cadre du modèle étudié, la capacité de calcul requise a diminué d'une décade en passant de 33 Gop/s à 2.9 Gop/s pour un taux de comptage de $10^5$ cps. Cependant, il s'agit d'une moyenne corrélée au taux de comptage. Si l'on considère que le cas le plus défavorable est un grand nombre d'impulsions qui se suivent, cela ramène la problématique à l'état précédent, le modèle primaire. Cependant, la séparation des traitements nous permet d'introduire la notion de mémoire.

#### 2.3.2.2 Ajout d'une étape de mémorisation

De manière à diminuer l'impact de l'arrivée aléatoire des impulsions, il est possible d'ajouter une mémoire-tampon dédiées aux impulsions avant les traitements (Simoes et al. 1996). L'ajout de cette mémoire-tampon au modèle est présenté en Figure 2-6.

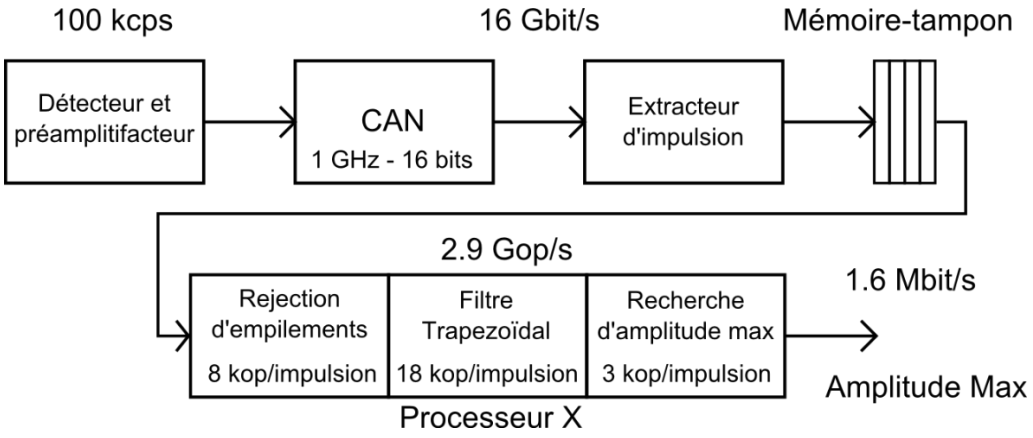



**Figure 2-6 : Système sans temps mort pour le modèle d'exécution dirigé par les impulsions incluant une étape de mémorisation.**

Si la profondeur de la mémoire permet au moins de contenir une impulsion, alors le modèle dirigé par les impulsions permet de réduire les performances requises (Cardoso et al. 2004b). En résumé, cette approche permet aux traitements de travailler de deux manières. La première, dite « flux borné », autorise les traitements, si la capacité de calcul le permet, à travailler en flux sur les échantillons d'une impulsion dès le déclenchement. La seconde, dite « traitement par paquet », autorise l'attente de la mémorisation complète ou partielle des échantillons d'une impulsion avant de les traiter. Par exemple, la majorité des filtres peuvent traiter les échantillons d'une impulsion en flux alors qu'une FFT a besoin de la mémorisation complète de celle-ci. La seconde approche implique donc un risque de temps mort si une autre impulsion se présente avant la fin du traitement de la précédente. Pour illustrer la transformation du flux, la Figure 2-7 représente le flux par l'écoulement de l'eau en sortie d'un robinet.

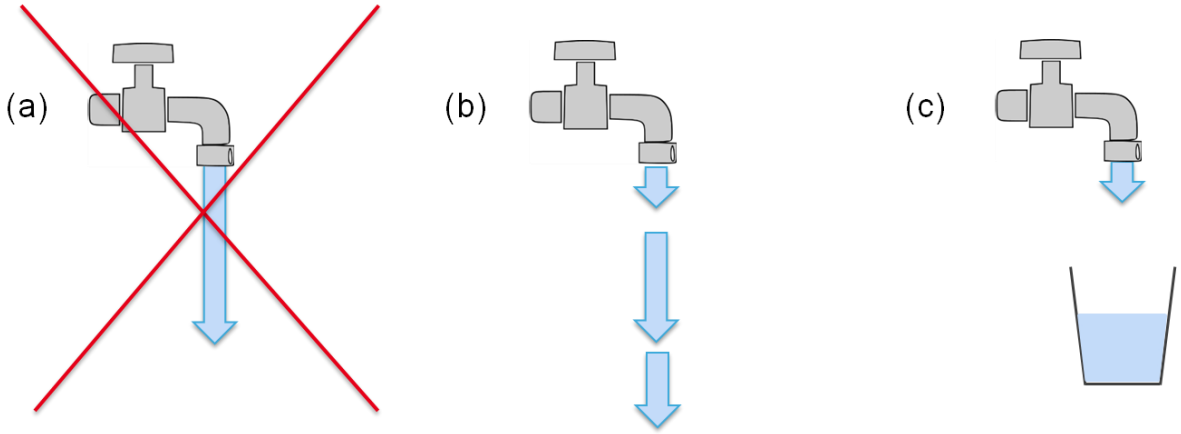



**Figure 2-7 : Illustration de la transformation du flux (a) vers des flux bornés (b) ou des paquets de données (c).**

Le flux borné réduit donc la quantité d'échantillons à traiter mais oblige les traitements à pouvoir travailler en flux. Ces flux sont intermittents et de taille variable. Le traitement par paquet impose par définition un risque de temps mort, mais a l'avantage de pouvoir transmettre aux traitements des informations telles que le nombre d'échantillons contenus dans le paquet.

Cette étape de mémorisation est introduite par l'intermédiaire du modèle *1-pulse stack* dans (Simoes et al. 1996) et est étendue vers le modèle *n-pulse stack* présenté dans (Cardoso et al. 2004b). Le premier modèle décrit n'autorise la mémorisation que d'une impulsion, tandis que dans le second cas la profondeur de mémorisation est variable. Ces modèles sont utilisés par leurs auteurs pour améliorer le modèle non-paralysable présenté en 2.2.2.

La partie précédente met cependant en évidence les différences entre l'extracteur d'impulsion de notre modèle d'exécution et le discriminateur de Simoes. Le système dit *1-pulse stack* entre en opposition avec le système décrit par Cardoso puisqu'il ne mémorise pas uniquement des impulsions. En effet, le déclencheur analogique utilisé n'extrait pas les impulsions. Son objectif est d'indiquer au processeur la position d'une impulsion (sa profondeur) dans une FIFO. Ce qui signifie que dans l'étage de mémorisation se trouvera à la fois du signal utile (des impulsions) et du signal non utile (bruit seul). Cette problématique s'étend alors au modèle *n-pulse stack.* Nous proposons donc d'utiliser le modèle *1-pulse stack* dans la suite de l'analyse tout en respectant la contrainte de ne mémoriser que des impulsions grâce à la définition de notre extracteur.

Nous décidons cependant, pour le moment, de ne pas étendre, en termes de capacité de mémorisation, notre modèle d'exécution jusqu'au modèle *n-pulse stack*. En effet, il paraît déjà difficile dans le système de Cardoso de dimensionner la mémoire pour qu'elle puisse, soit contenir la plus longue impulsion physiquement réalisable par le détecteur, soit rejeter un paquet d'empilements s'il dépasse la taille maximum de la mémoire-tampon tout en prenant en compte la probabilité qu'une impulsion puisse débuter à n'importe quelle profondeur de la FIFO. Dans le cadre de la mémorisation de plusieurs impulsions, la problématique est encore plus grande. De par l'aspect non déterministe de la taille des impulsions, une fois la première mémorisée, il n'est pas possible de s'assurer que la suivante respecte les limites du dimensionnement. De plus, il n'est pas précisé quelle action est effectuée sur la mémoire si celle-ci est pleine. C'est pourquoi nous choisissons pour le moment de nous focaliser sur l'utilisation du modèle *1-pulse stack*, qui entre en accord avec la définition de notre modèle, puisqu'il traite et mémorise effectivement des impulsions uniquement. Nous décidons d'y ajouter la contrainte suivante : la taille de la mémoire-tampon doit au moins contenir la plus grande impulsion physiquement productible par le détecteur.

En conclusion, et contrairement au modèle de Cardoso, seules les impulsions (qui constituent le signal utile) sont mémorisées dans notre proposition. De plus, l'utilisation d'un extracteur numérique permet facilement de signaler la fin d'une impulsion dans la mémoire afin que le processeur n'ait pas à la lire intégralement. Un empilement peut être également signalé par notre extracteur d'impulsion. Si la mémoire utilisée est une FIFO, alors le processeur peut, s'il en est capable, lire les échantillons à la volée. Cependant, il est obligatoire dans un système où le programme n'est pas connu à l'avance, de prendre le cas le plus défavorable, qui correspond ici à la nécessité de mémoriser entièrement une impulsion avant de la traiter (FFT).

#### 2.3.2.3 Impact sur le temps mort

Si l'on distingue deux processus différents pour la mémorisation et le traitement d'une impulsion, et si l'on admet que le système n'accepte pas d'autres impulsions alors que la mémoire n'est pas vide, alors, en prenant en compte ces éléments, l'impact du temps mort est diminué et le taux d'impulsions traitées augmente significativement (Simoes et al. 1996). L'avantage du modèle dirigé par les impulsions et de l'ajout d'une mémoire-tampon est représenté en Figure 2-8. Dans cette figure, si

les impulsions en sortie d'extracteur sont mémorisées à la volée, que les traitements (de durée arbitraire : deux fois la durée de l'impulsion) ne démarrent qu'une fois l'impulsion entière mémorisée (pire cas) et qu'une seule impulsion à la fois est acceptée en mémoire, alors seule l'impulsion 3 est perdue. Cette figure illustre que la mémoire permet la « dé-*randomisation* » des impulsions puisque tous les traitements s'enchaînent sans délai.

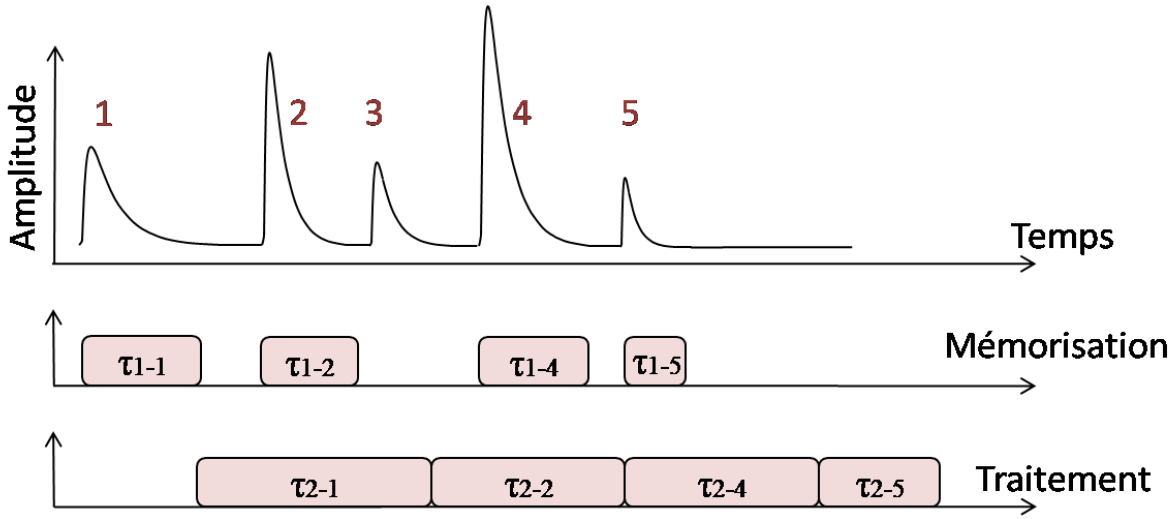



**Figure 2-8 : Illustration de la gestion du temps mort pour le modèle *1-pulse stack* dirigé par les impulsions.**

L'avantage de cette solution peut être démontré grâce à la réévaluation du modèle « non-paralysable » présenté en équation 1-3.

Avec $\tau_1$, la période de mémorisation, durant laquelle le système est aveugle à toute autre impulsion, et $\tau_2$, le temps de traitement d'une impulsion par le processeur durant laquelle le système de mémorisation peut accepter une nouvelle impulsion. Cette durée $\tau_2$ prend en compte l'émission du résultat.

Avec $P_t(x)$, est donnée la probabilité que $x$ événements aléatoires se produisent dans l'intervalle de temps $t$ avec un taux d'impulsions détectées $n$, est donnée par la distribution de Poisson présentée en équation 2-2.

$$P_t(x) = (nt)^x \cdot \frac{e^{-nt}}{x!} \qquad \textbf{2-2}$$

Pour faciliter la démonstration, l'équation 1-3 est réécrite sous la forme de l'équation 2-3.

$$m_0 = n - m_0 n\tau \qquad \textbf{2-3}$$

De même, $n\tau$ peut être remplacé par l'expression présentée en équation 2-4.

$$n\tau = \sum_{x=1}^{\infty} x \cdot P_t(x) \qquad \textbf{2-4}$$

Alors l'équation 2-3 peut être reformulée par l'équation 2-5. Pour $m_0 P_t(x)$, la probabilité de perdre $x$ impulsions par seconde.

$$m_0 = n - m_0 \sum_{x=1}^{\infty} x \cdot P_t(x) \qquad \textbf{2-5}$$

Par conséquent, le taux d'impulsions réellement traitées est présenté en équation 2-6. Avec $m(x-1)P_{\tau_2}(x)$, le taux avec lequel $(x-1)$ impulsions sont perdues si une impulsion est déjà en mémoire.

$$m = n - m \sum_{x=1}^{\infty} x \cdot P_{\tau_1}(x) - m \sum_{x=2}^{\infty} x \cdot P_{\tau_2}(x) \qquad \textbf{2-6}$$

Finalement, en réarrangeant l'équation 2-6 et en substituant l'équation de Poisson, le taux d'impulsions réellement traitées du modèle *1-pulse stack* peut être représenté par l'équation 2-7.

$$m = \frac{n}{e^{-n\tau_2} + n(\tau_1 + \tau_2)} \qquad \textbf{2-7}$$

Il devient donc possible de comparer les tendances de ce modèle avec le modèle non paralysable de Knoll. Pour cela, on utilisera les caractéristiques de notre modèle primaire pour une durée de mémorisation proportionnelle à la taille d'une impulsion et une durée d'exécution dix fois supérieure à la taille d'une impulsion. Pour $\tau_1 = 1\,\mu s$, $\tau_2 = 10\,\mu s$ et $n = 10^5 cps$, la Figure 2-9 compare les différents modèles.

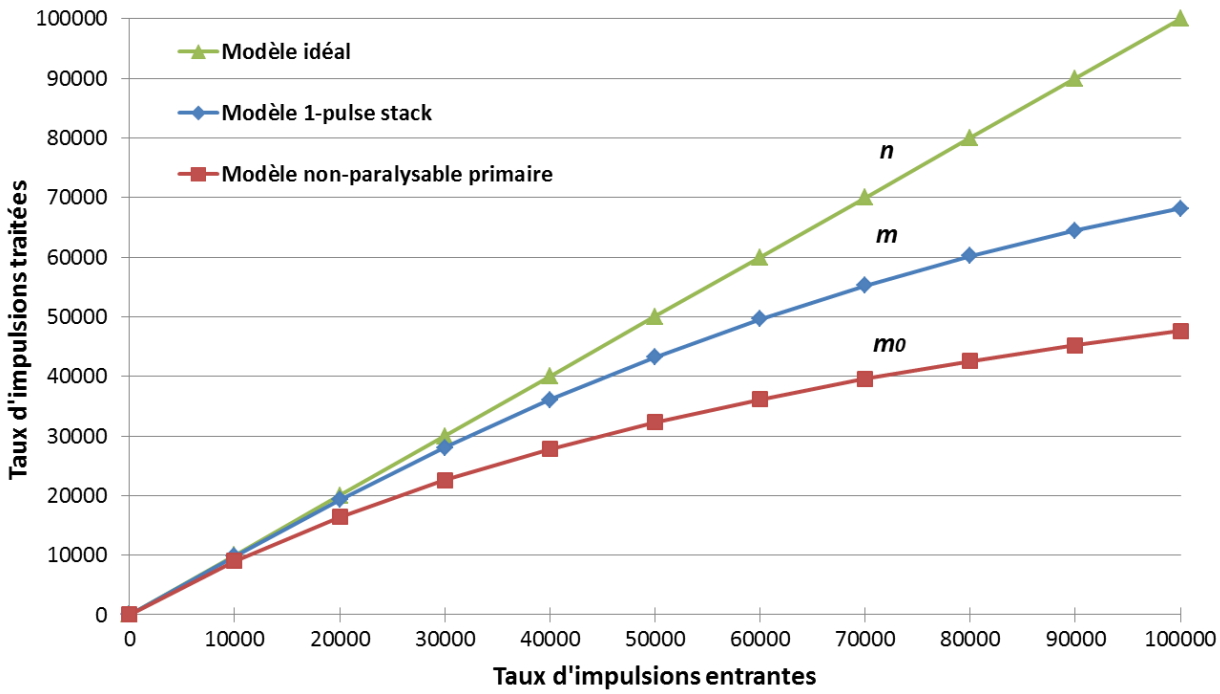



**Figure 2-9 : Taux d'impulsions traitées pour *n*, le modèle idéal, *m* le modèle non-paralysable et $m_0$ le modèle 1-*pulse stack*.**

Les résultats présentés en Figure 2-9 montrent une augmentation du taux d'impulsions traitées par le modèle dirigé par les impulsions avec mémoire face au modèle non paralysable conventionnel. Les deux courbes croissent de manière asymptotique en fonction du taux d'impulsions entrantes.

### 2.3.3 Conclusion

En conclusion, la première étape de notre proposition de modèle d'exécution consiste en la séparation des impulsions du reste du signal à l'aide d'un extracteur d'impulsions numérique capable de s'affranchir de la taille des impulsions. A cette étape, s'ajoute un étage de mémorisation permettant de faire face à l'arrivée aléatoire des impulsions. Grâce à cette première étape, le temps mort moyen est réduit.

Si l'ajout de la mémoire est essentiel pour permettre au processeur de travailler à sa propre fréquence, les traitements sont désormais dirigés par les impulsions et ne travaillent plus sur la totalité des échantillons, ce qui réduit le prérequis en performance du composant programmable utilisé. La mémoire n'empêche pas le processeur de travailler en flux sur les échantillons si son traitement le permet. De plus, la durée des traitements varie désormais avec la taille des impulsions. Pour finir, on peut désormais parler de gestion de flux borné ou de paquets par le processeur puisqu'il ne travaille que sur des paquets d'échantillons de tailles variables. De cette manière, si l'implémentation mémoire le permet (une FIFO asynchrone) le processeur n'a pas de contrainte de synchronisation avec l'extracteur d'impulsion.

Le modèle dirigé par les impulsions permet de changer de traitement et même de changer de processeur tout en garantissant le bon fonctionnement du système puisqu'ils deviennent asynchrones au signal d'entrée. Cette approche permet de garantir que le processeur démarre ses traitements et ne travaille que lorsqu'il y a du signal utile, permettant ainsi d'envisager une analyse de consommation générale du système.

## 2.4 Macro-pipeline des traitements

La seconde étape de notre modèle consiste à faire parcourir les impulsions extraites dans un macro-pipeline de traitement. En effet, les traitements d'une chaîne de mesure sont généralement composés d'une succession d'algorithmes. Ils forment naturellement un macro-pipeline logiciel tel que dans (Normand et al. 2009).

### 2.4.1 Définition du macro-pipeline des traitements

Si l'idée d'un modèle dirigé par les impulsions était de se servir de la nature aléatoire des impulsions et surtout de l'indépendance des traitements vis à vis de chaque impulsion, alors cette approche peut se répéter autant de fois qu'il y a de traitements. Notre modèle de macro-pipeline se base sur la granularité d'un processeur par traitement. A chaque étage de pipeline, on y retrouve le modèle dirigé par les impulsions. Pour passer d'un étage de traitement à un autre, l'impulsion passe par une étape de mémorisation afin de permettre à chaque processeur de la ligne de travailler à son rythme.

### 2.4.2 Modèle d'analyse des prérequis en performances

#### 2.4.2.1 Evolution du modèle dirigé par les impulsions vers le modèle macro-pipeliné dirigé par les impulsions

Le modèle dirigé par les impulsions est donc réévalué. Comme l'illustre la Figure 2-10, notre approche consiste à assigner un algorithme par processeur afin de réduire la latence globale à la pire

latence d'exécution d'un des processeurs du pipeline. Dans la Figure 2-10, la capacité de calcul maximum est alors ramenée à celle de l'algorithme « le plus gourmand », dans ce cas de figure, le filtre trapézoïdal. Le besoin de calcul maximum d'un processeur s'en trouve donc réduit.

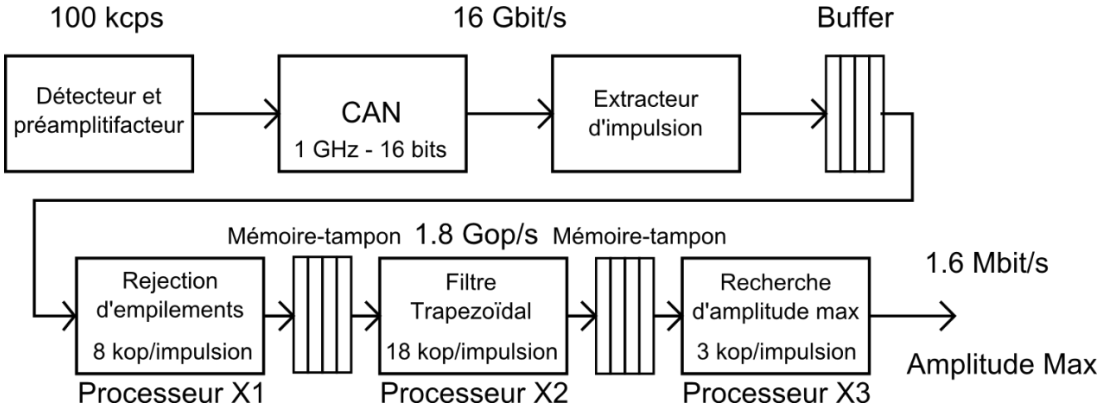



**Figure 2-10 : Système sans temps mort pour le modèle d'exécution dirigé par les impulsions incluant un macro-pipeline de processeur.**

Comme chaque traitement est exécuté localement et de manière indépendante sur un processeur, il est raisonnable de penser que cette approche limite les ressources requises, notamment en termes de mémoire partagée par rapport à un système multiprocesseur traditionnel. Cette approche est également entièrement compatible avec notre modèle dirigé par les impulsions, puisque chaque processeur distribue des paquets de taille variable dès que possible sur le premier processeur disponible, quelle que soit la durée du traitement. Les différents étages de processeur peuvent donc être asynchrones entre eux. Ce qui signifie qu'ils peuvent être choisis de manière à être dédiés à un traitement en particulier, rendant ainsi le modèle complètement hétérogène d'un point de vue calcul. Tel que proposé, le modèle autorise toujours la distribution du résultat à la volée dans l'étage de mémoire suivant (traitement en flux borné). Ce qui est possible, par exemple, pour un étage de filtrage.

#### 2.4.2.2 Impact sur le temps mort

L'impact sur le temps mort est difficile à évaluer analytiquement, puisqu'il dépendra principalement des traitements. C'est à dire que plus les traitements individuels sont gourmands en termes de ressources de calcul par échantillon et plus les étages sont nombreux, alors plus grand est le gain en termes de temps mort.

Cependant, le cas le plus défavorable du modèle dirigé par les impulsions est toujours présent à chaque étage du pipeline. En effet, il est possible que chaque traitement attende la totalité d'une impulsion avant de démarrer. Cela peut être envisageable pour un filtre anti-causal suivi d'une transformée de Fourrier par exemple.

La modification du modèle de temps mort de notre système est donc un ajustement de $\tau_2$. Si $\tau_2$ devient la somme des durées des traitements et que $\tau_1$reste toujours le cas le plus défavorable d'une mémorisation complète d'une impulsion avant traitement, alors la durée de $\tau_2$ devient la durée d'exécution la plus longue parmi les traitements du macro-pipeline appelée $\tau_{2max}$ comme illustré en 2-8.

$$m = \frac{n}{e^{-n\tau_{2max}} + n(\tau_1 + \tau_{2max})} \qquad \textbf{2-8}$$

### 2.4.3 Conclusion

La seconde étape de notre modèle d'exécution a consisté en la construction d'un macro-pipeline de traitement parcouru par les impulsions extraites. Cette approche nous a permis de séparer les traitements dirigés par les impulsions au moyen d'un macro-pipeline de traitements dirigés par les impulsions. Ce modèle a l'avantage de réduire l'ampleur des ressources de calcul nécessaire pour atteindre le zéro-temps mort s'il y a plus d'un traitement. Plus il y a de traitements de complexité équivalente, plus le modèle est avantageux. Ce système améliore la gestion du temps mort et, dans le cadre d'algorithmes pouvant travailler sur le flux d'échantillons d'une impulsion, permet plus facilement d'envisager le zéro-temps mort.

## 2.5 Distribution des impulsions

La troisième étape de notre modèle consiste à distribuer les impulsions en sortie de l'extracteur d'impulsions sur différents étages de traitements macro-pipelinés. L'idée principale est que, si l'on souhaite atteindre des débits d'impulsions traitées plus élevés, notre modèle d'exécution doit pouvoir surmonter le goulot d'étranglement des traitements ne pouvant pas travailler en flux sur les échantillons. Même si l'utilisation des deux précédentes étapes permet de réduire drastiquement le besoin en ressources de calcul, si l'un des traitements présents sur le pipeline ne peut pas assumer de travailler en flux, alors, avec l'augmentation du débit d'impulsions entrant, le taux d'impulsions traité tend à saturer. L'une des solutions possibles est donc d'augmenter le nombre de processeurs disponibles pour le traitement des impulsions.

### 2.5.1 Définition de la distribution des impulsions

Augmenter le nombre de processeurs est une solution classique des problématiques de performances des calculs numériques si ceux-ci peuvent être parallélisés. Ici, la solution ne consiste pas en la décomposition des traitements sur différents processeurs, mais en la multiplication des traitements sur différents processeurs. En effet, les impulsions sont individuelles. Cela signifie que les traitements peuvent les aborder dans le désordre, et sans contraintes de synchronisation entre eux. Même si dans certaines applications, la date d'arrivée des impulsions a un impact sur la suite des traitements (coïncidence), les impulsions peuvent toujours être traitées dans le désordre pourvu qu'elles soient datées à l'étage d'extracteur d'impulsions. Une fois datées et extraites, les impulsions pourront être distribuées sur différents étages de traitements. La définition d'un tel système se rapproche, selon la taxonomie de Flynn, d'un modèle Multiple Instruction Multiple Data (MIMD) à mémoire distribuée puisque chaque CPU à son propre système d'exploitation et sa propre mémoire (Flynn 1972). Cependant, en intégrant la notion de macro-pipeline, avec la granularité d'un traitement/programme par processeur avec un minimum de deux programmes, le modèle doit être étendu vers une extension du MIMD moins commune appelée Multiple Programme Multiple Data (MPMD) (George et al. 2011). Connaître la catégorie de notre modèle d'exécution permettra, lors de la conception de l'architecture associée, d'exploiter les solutions existantes de la littérature. En résumé, dans ce type d'architecture, il n'y a aucune dépendance entre les processeurs, ni même entre les traitements qui les composent. Ils ne partagent ni mémoire ni impulsions.

Pour la suite de l'étude nous appellerons ligne de traitements, l'ensemble des processeurs macro-pipelinés sur lequel il est possible de distribuer une impulsion. Le système doit donc permettre de choisir sur quelle ligne de traitement envoyer les impulsions. Une ligne de traitements est la combinaison des deux modèles précédents. Nous appelons étage les différentes étapes de traitements

qui composent le macro-pipeline. Si chaque ligne doit être identique en termes d'étage de traitements, les étages ne sont pas nécessairement homogènes entre eux. Il est en effet possible sur un modèle MPMD d'utiliser des processeurs différents (adaptés au traitement) à chaque étage. Cependant, pour faciliter la suite de l'explication, nous considérerons que les étages sont homogènes.

### 2.5.2 Analyse des performances

#### 2.5.2.1 Evolution du modèle macro-pipeliné vers le modèle distribué

Cette sous-section s'intéresse donc à observer de quelle manière la distribution des impulsions va permettre de réduire les prérequis en performances des processeurs de notre modèle. Pour un même débit d'entrée $n$ de $10^5$ cps et un choix arbitraire de trois étages de traitements, le modèle réévalué est présenté en Figure 2-11. Dans ce modèle, les impulsions sont distribuées ligne par ligne dans un ordre numérique prédéfini (X1-1, X2-1 puis X3-1) donc sans ordonnancement.

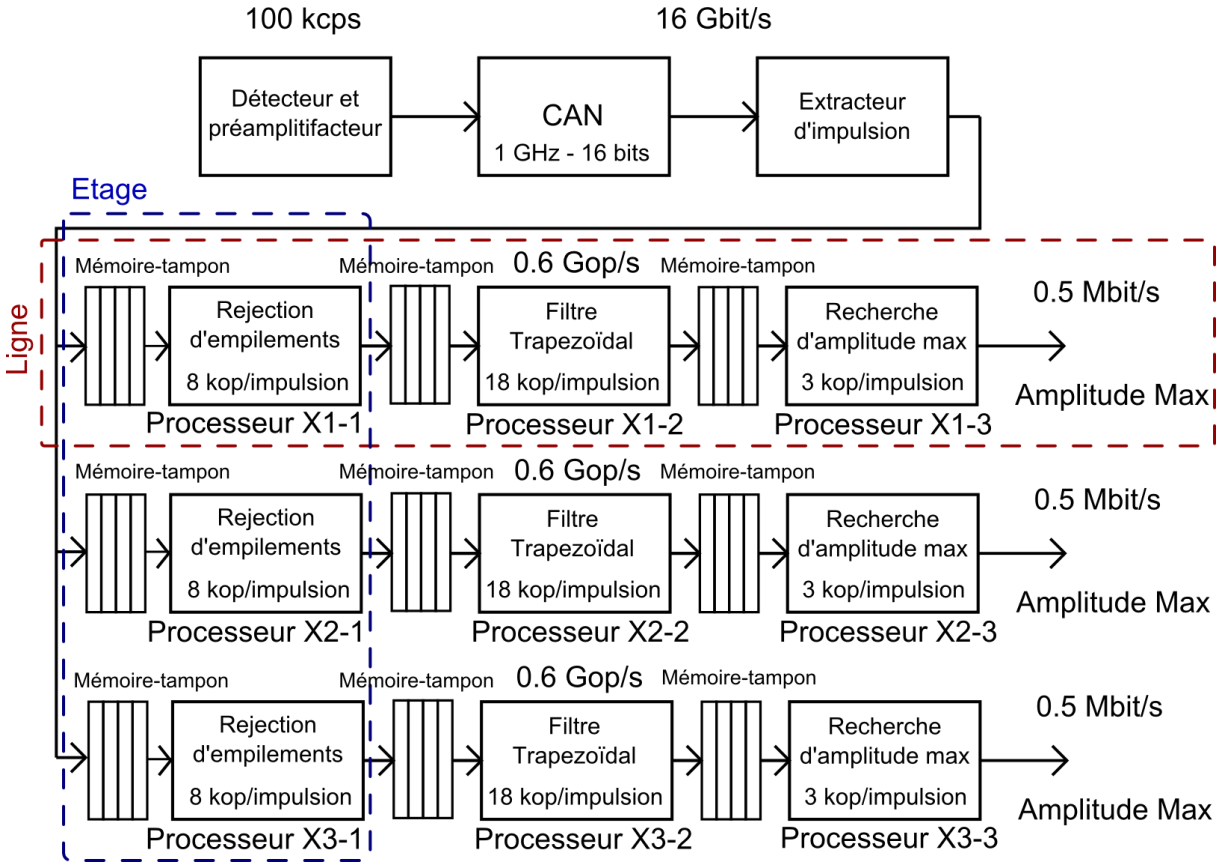



**Figure 2-11 : Système sans temps mort pour le modèle d'exécution distribué. En bleu, un étage de traitements, en rouge, une ligne de traitements.**

La Figure 2-11 met en évidence le gain en termes de ressources de calcul nécessaires pour être zéro-temps mort. Sur chaque étage, le nombre d'opérations requises est divisible par le nombre d'étages présents. En effet, le nombre d'impulsions à traiter par ligne diminue en fonction du nombre de lignes de traitements. Dans cet exemple, le pire cas, en termes de dimensionnement, devient 0.6 Gop/s par ligne de traitements. Cela diminue les performances de calcul requises par ligne de traitements, mais aussi le risque d'être confronté au pire cas (train d'impulsions à traiter).

**Conclusion intermédiaire :**

Le modèle distribué fait apparaitre la flexibilité nécessaire en termes de dimensionnement, puisqu'en fonction du détecteur et des traitements le nombre de lignes de traitements requis pour être zéro-temps mort peut varier. De même, les processeurs présents par étage de traitements peuvent être hétérogènes et être choisis en fonction du traitement à effectuer. De par sa nature globalement asynchrone, un tel système pourrait être suffisamment modulaire pour être dimensionné en fonction des capacités de calcul imposées par une application. Ce qui, en dépit d'une problématique de coût (en termes de prix ou de surface), rend réaliste l'utilisation de composants programmables, puisque le manque de performances peut être compensé par le nombre de processeurs. Dans ce modèle, aucun ordonnancement n'est pour le moment nécessaire, il n'y a donc aucun délai de prise de décision pour la distribution des impulsions, ce qui n'implique pas de temps mort. Il est cependant possible d'ajouter un étage d'ordonnancement pour distribuer les impulsions en fonction de différentes contraintes, comme la disponibilité des processeurs. Cet aspect sera traité dans le Chapitre 4.

#### 2.5.2.2 Impact sur le temps mort

Le temps mort des modèles précédemment analysés (modèle non paralysable, dirigé par les impulsions, avec mémoire et macro-pipeliné) a été obtenu analytiquement. Toutefois, le temps mort d'architectures numériques plus complexes, contenant un plus haut degré de parallélisme, est extrêmement difficile à évaluer à l'aide d'une approche probabiliste classique. Selon (Cardoso et al. 2004b), « la meilleure approche pour analyser ce type d'architecture complexe est d'utiliser un outil de simulation ». Cet outil est développé et présenté dans le Chapitre 4. La partie suivante s'intéresse donc plus à une vue théorique du modèle et met en évidence les enjeux architecturaux nécessaires pour la réalisation de l'architecture et du simulateur associé.

Si l'analyse du temps mort est devenue trop complexe à interpréter par le calcul, nous pouvons établir le constat suivant : en distribuant les impulsions, s'il y a suffisamment d'étages de traitements pour traiter les impulsions alors il y a zéro-temps mort. Ce constat est possible si les contraintes suivantes sont respectées :

- il n'y a pas de délai d'ordonnancement, ou ce délai peut être comblé avant l'arrivée d'une impulsion (pré-ordonnancement) ;
- le délai de transfert et de mémorisation $\tau_1$ n'augmente pas avec l'augmentation du nombre d'étages de manière à conserver $\tau_1 \leq \tau_2$.

En respectant ces contraintes, il est possible d'atteindre le zéro-temps mort. Cette méthode est illustrée en Figure 2-12. La Figure 2-12 montre que pour une distribution des impulsions dans un ordre prédéfini, c'est-à-dire un étage de traitements après l'autre (sans ordonnancement), toutes les impulsions sont traitées.

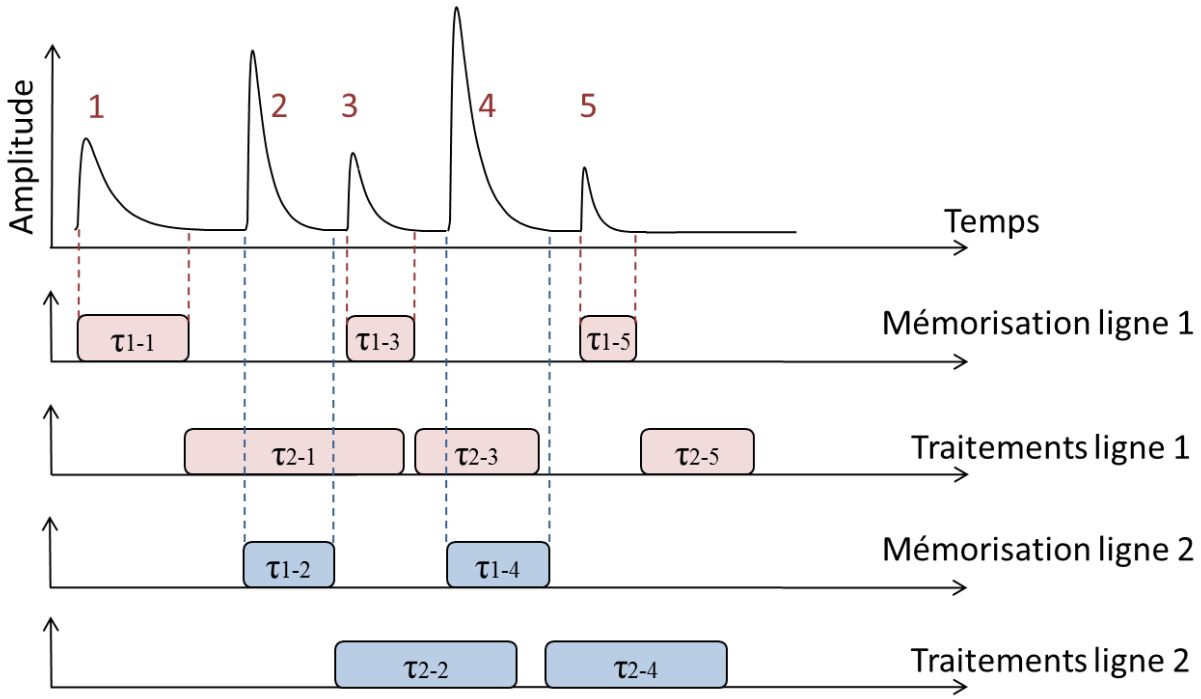



**Figure 2-12 : Illustration de la gestion du temps mort pour le modèle distribué et dirigé par les impulsions. La distribution un étage après l'autre. Dans cet exemple, toutes les impulsions sont traitées.**

Cette première illustration indique qu'il est en principe possible d'atteindre le zéro-temps mort. On constate que les ressources sont utilisées équitablement mais on ne fait pas apparaître analytiquement l'impact résultant de l'ajout d'un étage de traitements. Dans l'analyse du modèle précédent, pour le même signal idéal analysé, seule l'impulsion 3 est perdue, ce qui laisse envisager qu'une méthode de distribution prédéfinie va tendre vers un surcoût de l'architecture en termes de dimensionnement. De plus, cette méthode présente le risque de distribuer des impulsions sur des lignes n'étant pas en état de les accepter (le premier étage de mémoire étant déjà occupé par une impulsion par exemple).

Nous proposons donc de réaliser la même analyse pour une distribution gérée par un ordonnanceur. L'ordonnancement est de type première ligne de traitement disponible, première servie. Comme pour le modèle dirigé par les impulsions, si durant $\tau_2$ (le temps de traitement d'une impulsion par le premier processeur d'un des étages de traitements), plus d'une impulsion est mémorisée au-delà de la première, elles sont perdues. Cette analyse est illustrée en Figure 2-13, et confirme que l'étage deux n'est en réalité nécessaire que pour le traitement d'une impulsion. Ceci permet d'envisager la création d'un simulateur qui permettrait d'éviter le surcoût d'un dimensionnement purement analytique en permettant de faire apparaître le taux d'utilisation de chaque ligne de traitements.

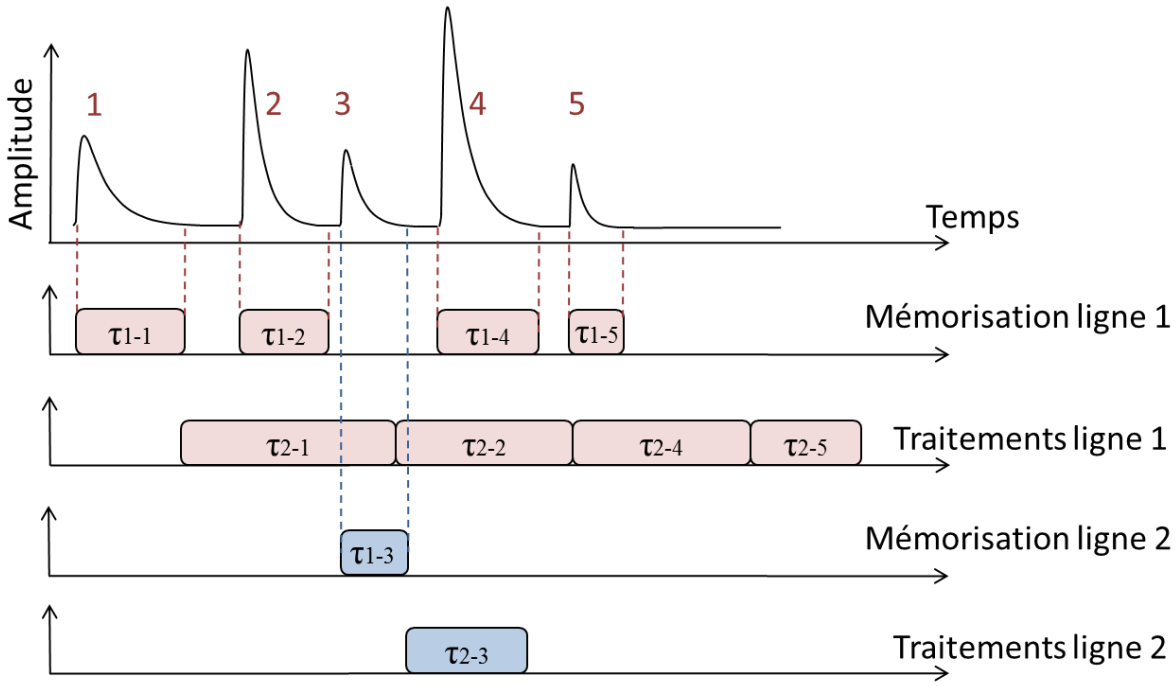



**Figure 2-13 : Illustration de la gestion du temps mort pour le modèle distribué et dirigé par les impulsions. La distribution se fait au premier étage disponible pour traiter une impulsion. Dans cet exemple, toutes les impulsions sont traitées.**

Si, analytiquement, on souhaite se rapprocher du cas réel pour le calcul du temps mort d'un système distribué, nous proposons par approximation l'utilisation de l'équation 2-7. Avec $L$, le nombre de lignes de traitements.

$$m = \sum_{i=1}^{L} \frac{\frac{n}{i}}{e^{-\frac{n}{i}\tau_2} + \frac{n}{i}(\tau_1 + \tau_2)} \qquad \textbf{2-9}$$

La Figure 2-14 compare les modèles précédents avec le modèle distribué pour $L = 2$, $L = 5$ et $L = 10$. La Figure 2-14 fait nettement apparaître le gain apporté par la distribution des impulsions par rapport aux modèles précédents. L'ajout de lignes de traitements augmente le taux d'impulsions traitées et fait tendre le système vers le zéro-temps mort pour les taux d'impulsions entrantes les plus faibles. Cependant, si le gain est important entre l'ajout d'une deuxième ligne de traitements comparativement au modèle dirigé par les impulsions (77 % d'impulsions traitées en plus), ce gain est réduit avec l'ajout d'autres lignes de traitements, si bien qu'entre le modèle à cinq lignes et dix lignes la différence est très faible (moins de 10 % d'impulsions supplémentaires traitées). Il est donc possible de dimensionner un système se rapprochant le plus du zéro-temps mort, mais aux risques d'un surcoût élevé en termes de processeurs ajoutés. Le modèle offre donc au concepteur un compromis possible entre performance et coût qu'il sera possible d'évaluer à l'aide d'un simulateur apte à modéliser ce parallélisme pour visualiser le temps mort.

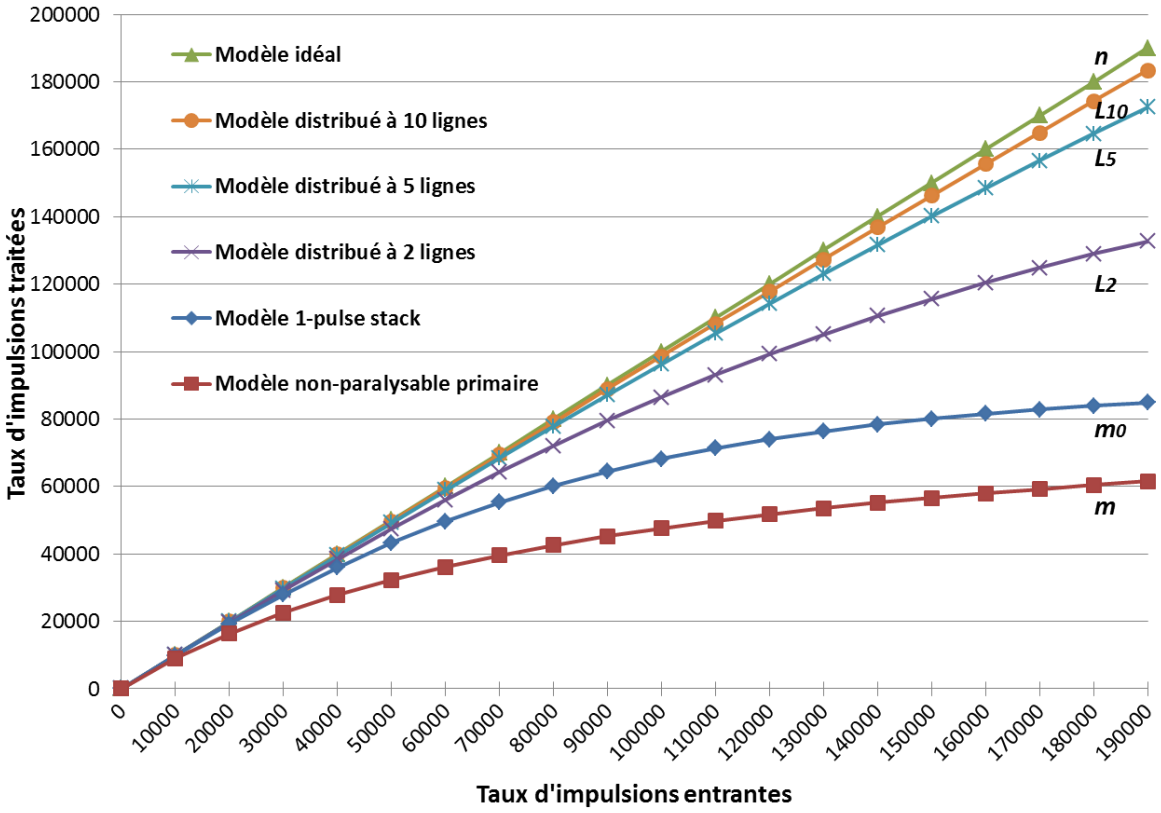



**Figure 2-14 : Taux d'impulsions traitées pour n, le modèle idéal, m le modèle non paralysable, m0 le modèle 1-pulse stack, L2 le modèle distribué à 2 lignes de traitements, L5 le modèle distribué à 5 lignes de traitements et L10 le modèle distribué à 10 lignes de traitements.**

### 2.5.3 Conclusion

La troisième étape du modèle d'exécution proposé distribue les impulsions en sortie de l'extracteur sur différentes lignes de traitements. Chaque ligne est composée d'un macro-pipeline de traitements dirigés par les impulsions. Cette étape permet d'augmenter significativement le taux d'impulsions traitées en fonction du nombre de lignes allouées pour les traitements. Le gain diminue avec l'augmentation du nombre de lignes pour des taux de comptage identiques. Ce modèle introduit la notion de réseaux de distribution et d'ordonnancement qui doivent être confrontés à la problématique du passage à l'échelle. Une distribution prédéfinie admet cependant de distribuer les impulsions une voie de mesure après l'autre, donc sans ajouter un temps mort dû à la prise de décision. Les limites du modèle analytique du temps mort ont été atteintes par l'ajout du parallélisme, faisant apparaître le besoin d'un simulateur dédié à ce modèle d'exécution et apte à faire apparaître le gain réel en termes de temps mort. Si ce modèle permet théoriquement d'atteindre le zéro-temps mort, la forte granularité de l'utilisation d'un composant programmable par algorithme fait apparaître la problématique du passage à l'échelle, notamment dans le cadre d'applications multivoies.

### 2.5.4 Discussion

Dans cette étape, le nombre de voies de traitement est augmenté. Cela implique qu'il sera nécessaire de multiplexer les données en sortie de chaque voie. Le volume de données est plus faible en sortie qu'en entrée d'une chaîne de mesure, cependant ces données arrivent en sortie de chaque ligne de traitements de manière aléatoire. Cela implique une gestion de la concurrence entre ces différentes lignes. Heureusement, pour ce type de problématique des solutions existent déjà, elles se

présentent par exemple sous la forme d'arbres de multiplexeurs (Chao 2002), de réseaux partagés (*Time Division Multiple Access*), ou d'architecture à accès mémoire non uniforme (NUMA).

## 2.6 Partage des ressources entre voies d'acquisition

Augmenter le nombre de voies implique que plusieurs extracteurs d'impulsions doivent distribuer leurs impulsions sur les lignes de traitements disponibles. De ce fait, plus d'impulsions doivent être traitées, augmentant ainsi les performances de calcul requises ou le nombre de lignes de traitements requis pour atteindre le zéro-temps mort. Avec le modèle d'exécution actuel, il suffirait d'allouer à chaque voie de mesure un modèle distribué. Cependant, chaque voie est indépendante et reçoit des évènements de manière aléatoire, de par la nature isotrope de l'émission des rayonnements. Cela signifie théoriquement que les extracteurs d'impulsions associés à chaque voie ne devront pas, en général, distribuer leurs impulsions aux mêmes instants. Cela signifie qu'ils n'ont pas forcément besoin d'accéder simultanément à leurs lignes de traitements pour affecter leurs impulsions. Par conséquent, partager les lignes de traitements entre les voies d'acquisition peut permettre d'augmenter le nombre d'impulsions traitées en réduisant le délai pour trouver une ligne disponible. L'affectation est régie par un ordonnanceur.

### 2.6.1 Définition du partage de ressources

Le partage des lignes de traitements entre voies d'acquisition redéfinit entièrement les contraintes de notre modèle. En effet, la gestion de plusieurs voies d'entrée en parallèle rend obligatoire l'utilisation d'un module permettant de gérer l'affectation des impulsions sur les lignes de traitements (appelé « *manager* » dans un modèle MPMD). Cet ordonnanceur doit pouvoir gérer l'arrivée aléatoire des impulsions, mais aussi la disponibilité des ressources. Statistiquement, deux impulsions peuvent être détectées par le modèle d'extraction aux mêmes instants. Il en est de même pour les premiers étages de traitements de chaque ligne qui peuvent être disponibles (plus d'impulsion dans la mémoire) exactement au même instant. De plus, la nature asynchrone de ces modules rend difficile la gestion des accès concurrentiels. Enfin, s'il y a un ordonnanceur cela veut dire qu'il peut y avoir un délai d'ordonnancement, ce délai va donc se rajouter au délai de mémorisation et de traitement. Il en est de même pour les réseaux d'interconnexions qui doivent pouvoir supporter l'envoi simultané de plusieurs impulsions sans congestionner et donc ajouter une latence supplémentaire. Ces problématiques feront parties intégrantes des choix architecturaux présentés dans le Chapitre 4. Seul le cas de fonctionnement idéal (sans latence) sera abordé ici.

L'ordonnancement peut être défini comme celui illustré en Figure 2-13. Chaque ligne de traitements possède plusieurs étages de pipeline. Le premier étage est celui qui communique avec l'ordonnanceur qui choisit vers où seront distribuées les nouvelles impulsions. Cet étage renseigne simplement son état : occupé ou libre. Libre signifie qu'il n'y a plus d'impulsion dans l'étage de mémoire, occupé, l'inverse. L'ordonnanceur a donc connaissance des états de tous les premiers étages de chaque ligne de traitements. Dès qu'une impulsion est détectée par un module d'extraction, l'ordonnanceur doit déjà avoir choisi vers quelle ligne libre elle sera distribuée. Dans le cas contraire, l'impulsion est perdue. L'affectation peut se réaliser par l'intermédiaire de la combinaison d'un ordonnancement de type première ligne libre-première servie. De cette manière, dès qu'une impulsion est détectée, elle sera directement distribuée vers une ligne de traitements libre sans délai.

En tenant compte de la nature non déterministe des traitements en termes de durée, on s'aperçoit que ce processus d'ordonnancement et d'affectation doit être réalisé à chaque étage de

pipeline de chaque ligne. Notre modèle se distingue donc à présent d'une architecture MPMD, puisque que cette architecture implique un « *manager* » pour l'ensemble des processeurs. Dans notre modèle, un ordonnanceur est affecté à chaque paire d'étages. Par conséquent, chaque étage peut être hétérogène en termes de nombre d'éléments programmables. Ce qui veut dire qu'à chaque étage, le duo élément programmable et mémoire renseigne son état à l'ordonnanceur précédent pour déverser son impulsion/résultat à l'étage suivant. Une illustration du modèle pour trois voies de mesure et des étages homogènes est présentée en Figure 2-15.

### 2.6.2 Analyse des performances

#### 2.6.2.1 Evolution du modèle distribué vers le modèle partagé

Si la théorie du partage de ressources semble adaptée à la nature non déterministe du signal, cela ne peut se visualiser par notre méthode de capacité de calcul requise pour atteindre le zéro-temps mort. Comme l'illustre la Figure 2-15, si statistiquement le taux d'impulsion par voie de mesure ne change pas, alors pour chaque ligne il est possible d'appliquer le calcul de performances du modèle à mémoire dirigé par les impulsions. Dans ce cas de figure, la capacité de calcul maximum requise pour atteindre le zéro-temps mort reste de 1.8 Gop/s, Ce qui correspond à trois fois la capacité requise pour une voie. Cette analyse ne permet donc pas d'observer un gain. Ce gain sera visible dans l'analyse du temps mort de ce modèle.

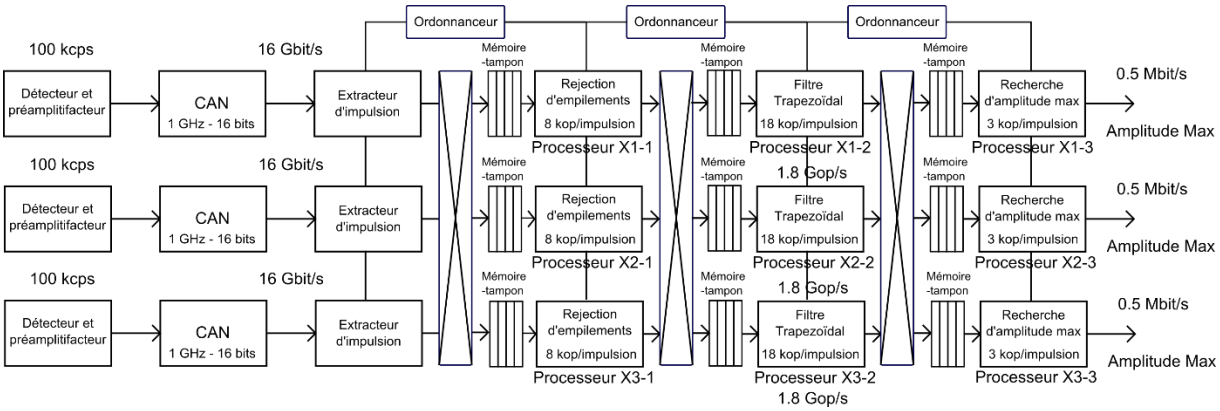



**Figure 2-15 : Système sans temps mort pour le modèle d'exécution partagé.**

#### 2.6.2.2 Impact sur le temps mort

Le modèle analytique étant trop complexe dans ce cas de figure, nous préférons choisir d'illustrer les avantages théoriques du modèle partagé pas l'utilisation d'exemples concrets.

Dans l'exemple illustré en Figure 2-13, deux lignes de traitements étaient requises pour traiter les impulsions issues d'une voie d'acquisition. Nous conservons pour cet exemple le signal idéal présenté en Figure 2-13. Nous décidons d'ajouter une voie de mesure et d'observer les différences en termes de temps mort obtenues par le partage. Le signal issu de la seconde voie de mesure est présenté en Figure 2-16.

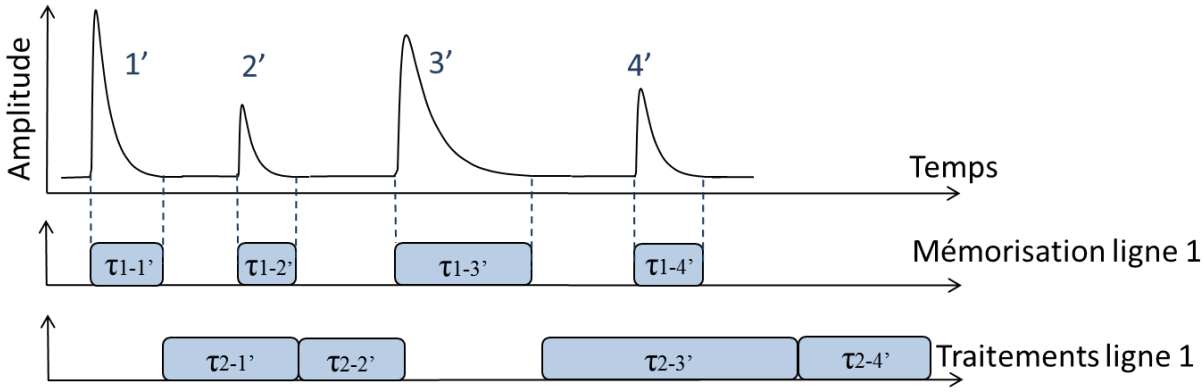



**Figure 2-16 : Seconde illustration de la gestion du temps mort pour le modèle distribué et dirigé par les impulsions. Dans cet exemple, toutes les impulsions sont traitées.**

Si, pour la première voie de mesure (Figure 2-13), il faut deux lignes de traitements pour atteindre le zéro-temps mort et si, pour la seconde voie de mesure, il faut une seule ligne de traitements pour atteindre le zéro-temps mort, alors le modèle distribué nécessitera trois lignes de traitements pour traiter toutes les impulsions. Nous décidons donc d'utiliser ces deux signaux pour le modèle partagé présenté en Figure 2-17.

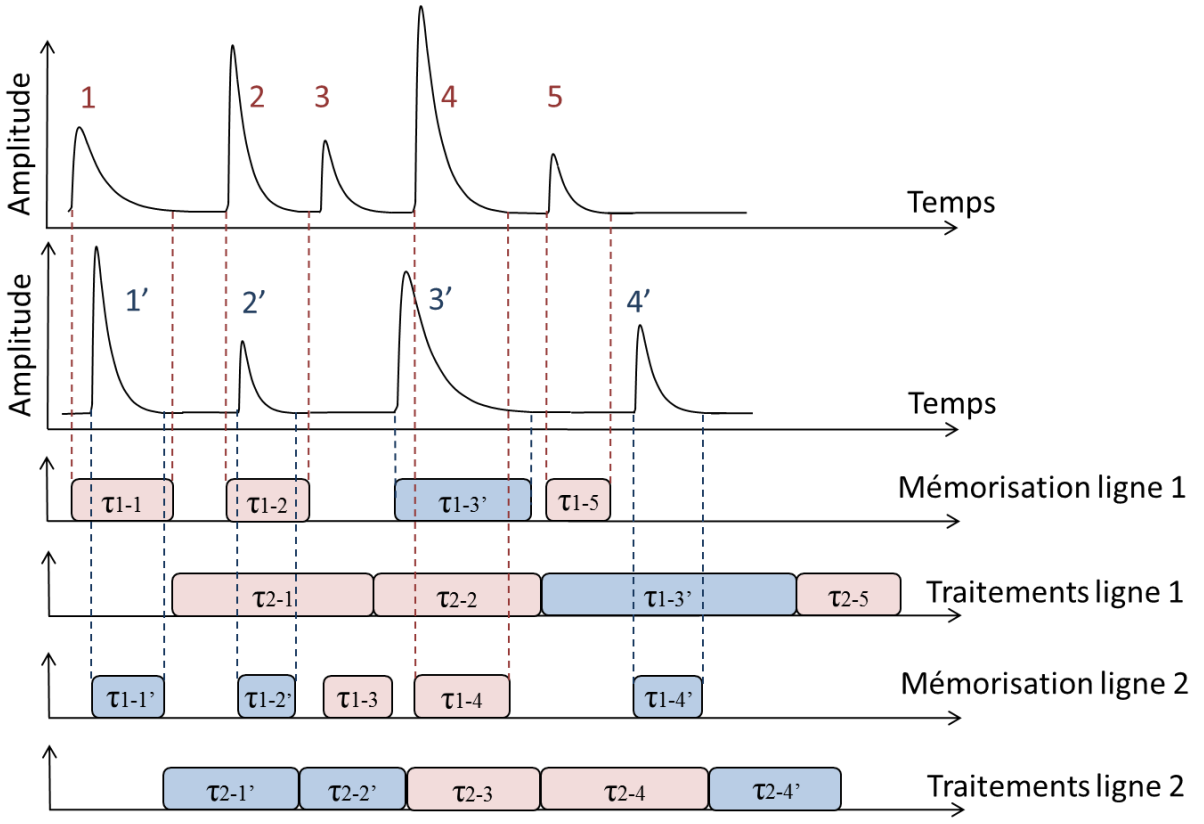



**Figure 2-17 : Illustration de la gestion du temps mort pour le modèle partagé. La distribution se fait à la première ligne de traitements disponible pour traiter une impulsion. Dans cet exemple, toutes les impulsions sont traitées.**

Dans la Figure 2-17, toutes les impulsions sont traitées avec seulement deux lignes de traitements. Cela nous permet d'être optimistes quant à l'utilisation du modèle partagé pour faciliter

le passage à l'échelle. Cependant, pour appuyer cette théorie, il devient critique de développer un simulateur. Dans un premier temps, nous choisissons donc d'avoir une vue empirique du gain, en termes de nombre de ressources, apportée par cette dernière étape du modèle d'exécution en fonction du nombre de voies de mesure pour le cas d'étude utilisé tout au long de ce chapitre.

#### 2.6.2.3 Validation du passage à l'échelle du modèle dirigé par impulsions à ressources de calcul partagées

Dans les parties précédentes la difficulté a été énoncé d'établir un modèle analytique permettant d'évaluer notre modèle d'exécution actuel. Cependant, pour valider la théorie et montrer que le partage de ressources entre voies de mesures offre un gain en termes de passage à l'échelle, il est possible de s'appuyer sur les résultats d'une simulation.

Nous avons choisi de réaliser cette simulation sous Matlab pour avoir un premier aperçu empirique du gain par le partage de ressources. Dans un premier temps pour un même taux de comptage, puis pour des taux de comptage variables. Cette approche est comparable au simulateur énoncé dans (Cardoso et al. 2004b). Toutefois, dans ce cas, nous ne cherchons pas à dimensionner une architecture pour une application spécifique. En effet, nous souhaitons observer, pour notre modèle à ressources partagées, l'augmentation du nombre de lignes de traitements en fonction, d'une part de l'augmentation du nombre de voies et, d'autre part, du débit d'impulsions entrant tout en respectant la contrainte d'un système zéro-temps mort.

**Paramètres de simulation :**

Le simulateur permet d'intervenir sur les paramètres suivants :

- la fréquence *f* d'échantillonnage des impulsions ;
- la durée $\tau_2$ d'exécution des algorithmes de traitements d'impulsions présents sur les unités de calcul ;
- un nombre d'échantillons par impulsion aléatoire et compris entre deux valeurs réalistes ;
- la durée de l'expérimentation ;
- le nombre de détecteurs ;
- le débit *i* d'arrivée des impulsions aléatoirement réparties sur une période de temps donnée ;
- la taille des mémoires-tampon allouées à chaque unité de calcul.

Les hypothèses suivantes ont été considérées :

- le temps d'accès aux ressources ne croît pas avec l'augmentation du nombre de celles-ci ;
- l'ordonnancement se fait en un cycle ;
- il n'y a qu'une seule unité de calcul par ligne, le macro-pipeline n'intervenant pas significativement dans la simulation ;
- L'algorithme présent sur les unités de calcul permet de lire en flot de données la mémoire-tampon ;
- afin de respecter le mode impulsions, la somme des durées des impulsions est inférieure à la durée considérée pour la simulation ;
- les détecteurs sont répartis uniformément autour de la source comme dans (Bazzacco 2004). De ce fait, la probabilité qu'un détecteur interagisse avec un rayonnement radioactif est la même pour tous les détecteurs.

La distribution des impulsions sur les lignes de traitements est réalisée à l'aide d'un algorithme premier-étage-disponible premier-étage-servie appliqué à la somme des échantillons présents dans les mémoires allouées à chaque unité de calcul et du nombre de cycles restant pour que l'unité de calcul termine son traitement sur l'impulsion en cours.

Les paramètres choisis sont ceux du cas applicatif utilisé pour ce chapitre. C'est-à-dire $f = 1Ghz$, $\tau_2 = 10\mu s$, $i = 10^5 cps$, la durée moyenne des impulsions est de $1\ \mu s$.

**Résultats de simulation :**

La Figure 2-18 et la Figure 2-19 présentent l'évolution du nombre de ressources de calcul nécessaire par détecteur pour réaliser les traitements d'une durée $\tau_2$ sans temps mort en fonction du nombre de détecteurs. Le débit d'impulsions *i* entrantes par voie de mesure est fixe dans la Figure 2-18 et varie dans la Figure 2-19. La Figure 2-18 montre que le besoin en nombre de ressources de calcul total ramené au nombre de détecteurs ne croît pas de manière proportionnelle au nombre de détecteurs. Au contraire, la courbe s'apparente à une exponentielle décroissante. La présence de bruit est due au non déterminisme de l'arrivée et de la taille des impulsions, puisqu'à chaque boucle de simulation le tirage aléatoire des dates d'arrivées des impulsions est ré-effectué. Ce bruit permet de visualiser la possibilité d'un signal dans le cas le plus défavorable (train d'impulsions). En effet, on constate que pour certains tirages associés à un nombre de voies de mesures, le nombre de lignes de traitements peut être supérieur à celui des tirages précédents avec moins de voies de mesure. *In fine*, nous avons en perspective de borner le pire cas par un calcul de variance sur ce bruit. La Figure 2-19 confirme cette tendance quel que soit le débit d'impulsion. En effet, si le débit ne dépasse pas les limites du mode impulsions, il influence la limite haute du nombre de ressources par détecteur en suivant une tendance qui s'apparente à une croissance linéaire, confirmant ainsi que le nombre de ressources ne divergera pas. Cette étude permet de confirmer la pertinence de l'utilisation du modèle à ressources partagées pour les systèmes multivoies. De plus, on s'aperçoit que plus le taux d'impulsions entrantes est élevé, plus le nombre de lignes de traitements décroît rapidement en fonction de l'augmentation du nombre de voies de mesure.

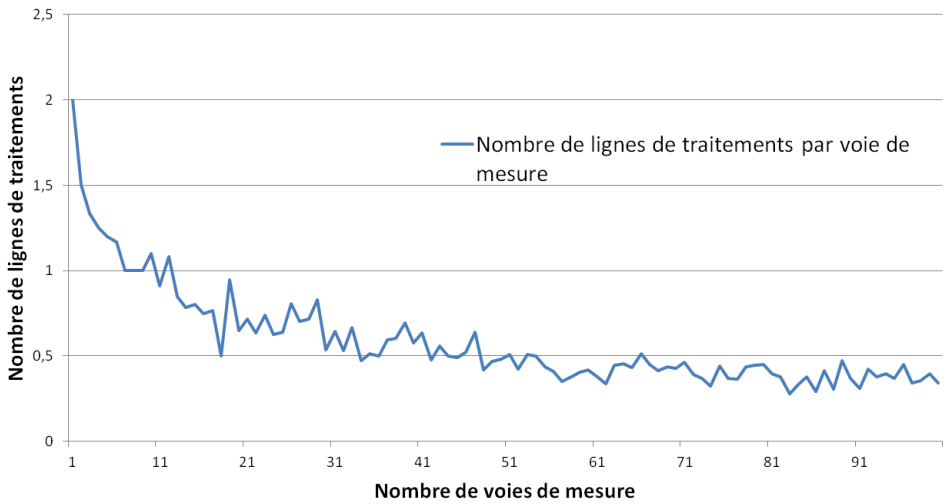



**Figure 2-18 : Evolution du nombre de ressources nécessaires au zéro-temps mort par détecteur en fonction du nombre de détecteurs pour un taux de comptage donné ($10^5$ cps).**

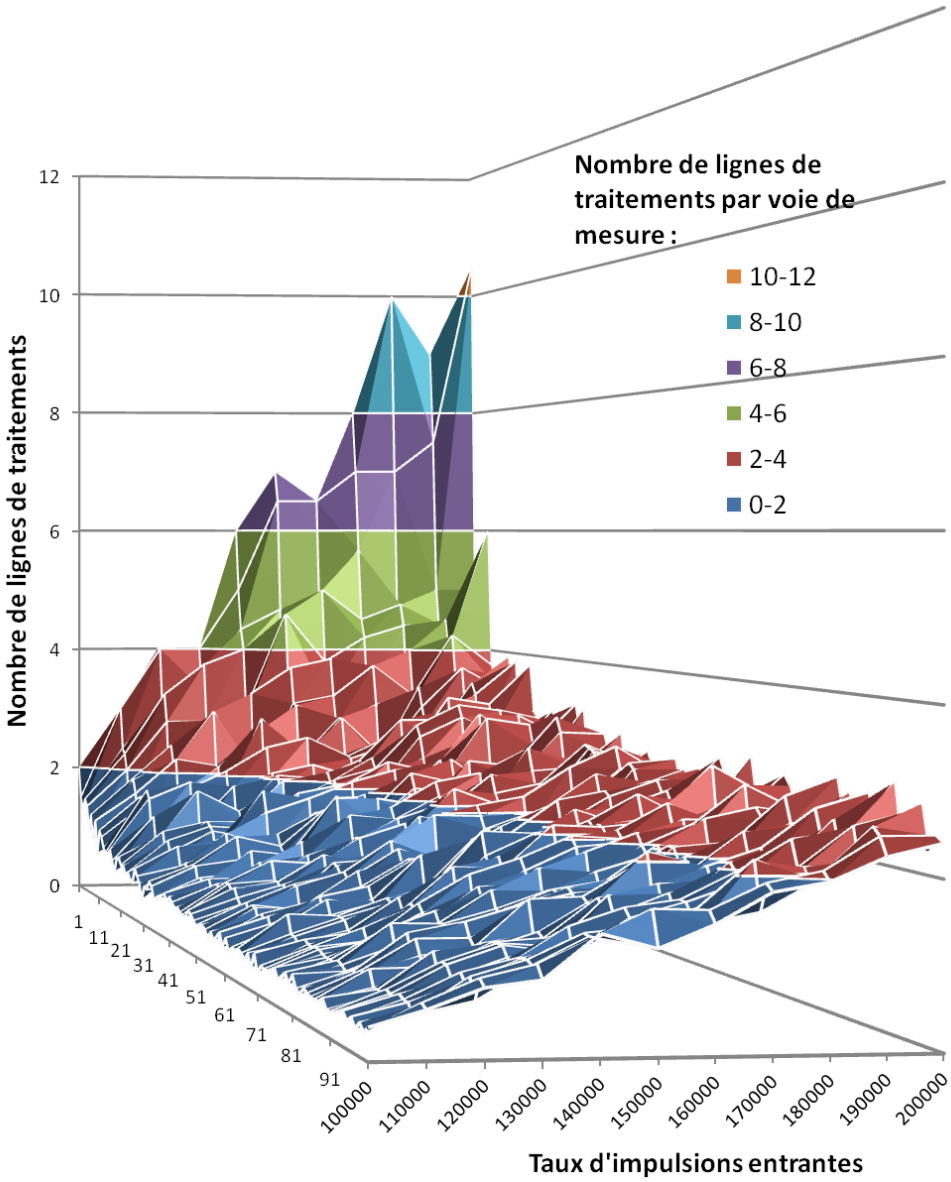



**Figure 2-19 : Evolution du nombre de ressources nécessaires au zéro-temps mort par détecteur en fonction du nombre de détecteurs et du nombre d'évènements à traiter sur une période donnée.**

### 2.6.3 Conclusion

La quatrième étape du modèle d'exécution proposé partage les lignes de traitements allouées entre différents extracteurs d'impulsions associés aux voies de mesure. A chaque étage de pipeline, les processeurs de chaque ligne de traitement deviennent partagés et disponibles pour recevoir les impulsions des étages précédents. L'information à traiter circule d'étage en étage mais aussi de ligne en ligne vers la première ressource disponible. Ce modèle offre une alternative intéressante au modèle distribué dans le cadre d'applications multivoies en réduisant théoriquement le nombre de ressources qu'il est nécessaire d'allouer à un modèle purement distribué pour un signal donné et un objectif de taux d'impulsions traitées qui peut tendre vers le zéro-temps mort. Si le modèle partagé se base sur le modèle distribué, il est donc également nécessaire de trouver le moyen de l'évaluer. La solution est donc de concevoir un simulateur permettant de modéliser ce modèle d'exécution.

## 2.7 Conclusion

Dans ce chapitre, un modèle d'architecture innovant, car dirigé par les impulsions, est proposé. Ce modèle offre une solution à la problématique de temps mort et au contexte multi-applicatif en séparant des impulsions du reste du signal. Les impulsions sont extraites puis distribuées en fonction de la disponibilité des ressources au sein d'unités de calcul programmables partagées. Ces unités de calcul fonctionnent de manière autonome, ce qui leur permet de s'adapter à la variabilité des données et des traitements. En s'affranchissant du non déterminisme du signal, cette nouvelle approche permet désormais d'envisager l'utilisation, pour le domaine de l'instrumentation nucléaire, de solutions couramment utilisées dans d'autres domaines, telles que les modèles multiprocesseurs ou la conception asynchrone. La conception de ce modèle d'exécution a mis en évidence plusieurs besoins. Le premier est la conception d'un module d'extraction des impulsions qui réponde aux critères imposés par notre modèle d'exécution. Le deuxième est la conception d'une architecture à base de composants programmables, de réseaux d'interconnexions et d'ordonnanceurs permettant le traitement des impulsions extraites en respectant les contraintes du modèle d'exécution. Le dernier est la conception d'un simulateur permettant d'analyser le temps mort pour les modèles distribués et partagés.

# Chapitre 3 : Extraction dynamique des impulsions

Ce chapitre vise à définir la première étape de notre modèle d'exécution, c'est-à-dire l'extraction des impulsions. En effet, comme cela est présenté dans le Chapitre 1, différentes caractéristiques doivent être extraites d'une impulsion en fonction de l'application. De plus, comme cela est présenté dans le Chapitre 2, une séparation précise des impulsions individuelles du reste du signal est un prérequis de notre modèle d'exécution. Pour cela, un déclencheur dynamique d'impulsion a été développé. Il a pour but d'être générique, ce qui signifie être capable de s'adapter aux variabilités de taille des impulsions et de bruits et donc être indépendant de la nature du détecteur. Dans un premier temps, l'état de l'art des méthodes de déclenchement est analysé (3.1). La méthode puis le modèle mathématique de notre déclencheur sont ensuite présentés (3.2). Enfin, une implémentation du déclencheur est proposée puis comparée aux autres méthodes de déclenchement de l'état de l'art (3.3). Cet algorithme a donné lieu à un brevet (Moline et al. 2014) et un article dans un journal à comité de lecture (Moline et al. 2015).

## 3.1 Introduction aux méthodes de déclenchement

Les approches traditionnelles de déclenchement reposent sur l'utilisation de discriminateurs à seuil, pour déclencher et stopper le traitement des impulsions. Si l'utilisation d'un seuil permet d'extraire les impulsions en fonction de leur taille, cette méthode est sensible au bruit. Une solution alternative est d'utiliser une constante de délai que nous appellerons par la suite fenêtre temporelle. Cette fenêtre est utilisée pour stopper et/ou ne pas réenclencher les traitements durant cette période. Le seuil et la fenêtre sont dimensionnés pour une durée d'impulsion et un niveau de bruit type. Toutefois, les caractéristiques des impulsions pouvant varier, cette approche se révèle être trop rigide. Cette partie consacrée aux méthodes traditionnelles de déclenchement s'ouvre sur l'exposé du principe de seuillage (3.1.1), pour continuer avec le principe de fenêtrage (3.1.2), avant que les deux traitements majeurs associés à l'étape de déclenchement : mise en forme du signal et gestion des empilements, ne soient présentées (3.1.3).

### 3.1.1 Principe du seuillage

Le seuillage traditionnel est basé sur la comparaison entre l'énergie du signal et un discriminateur statique afin de détecter le front montant d'une impulsion (Knoll 2010). Lorsque les impulsions représentatives des informations à extraire franchissent la valeur du seuil de détection, les traitements s'exécutent pour extraire l'information voulue (amplitude maximum, comptage). Cette approche oblige donc l'utilisateur à connaître à l'avance la totalité ou une partie de l'information qu'il souhaite extraire. A titre d'exemple, si l'utilisateur cherche à détecter une source caractérisable par ses hautes énergies, il peut se permettre de placer le seuil assez haut. Cette méthode n'est pas donc pas compatible avec des applications où l'on ne connaît pas la ou les sources à caractériser. Le seuil doit donc toujours être proche du bruit pour être capable de détecter toutes les énergies. *A contrario*, l'utilisateur doit, selon le système (en l'absence de filtre antibruit parfait), se prémunir au maximum de détecter du bruit (fausse alarme), l'obligeant, par méconnaissance de l'évolution du bruit, à fixer un seuil de détection suffisamment haut. De ce fait, une partie des informations (basses énergies) peuvent être perdues et les autres sont inexactes comme le présente la Figure 3-1.

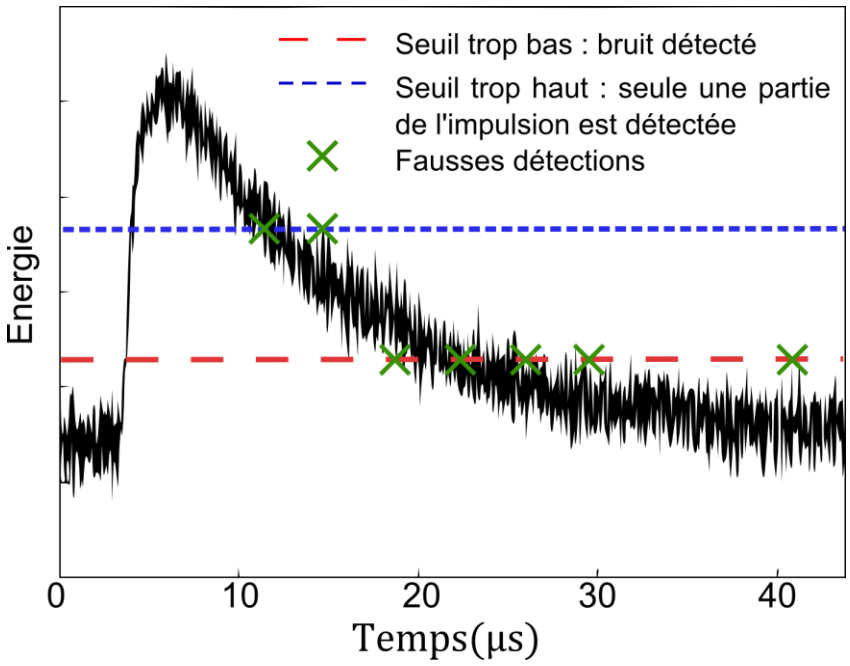



**Figure 3-1 : Problématique du déclenchement sur seuil statique.**

La Figure 3-1 fait apparaître les limites de l'utilisation d'une méthode de discrimination à seuil. Cette méthode, si elle est utilisée seule, sans fenêtrage ou modification du signal original, est sensible au bruit, aux variations de la ligne de base, aux empilements et surtout aux possibilités de déclencher sur la même impulsion. De plus, on constate que si cette méthode est en partie adaptée pour récupérer l'amplitude maximum d'une impulsion, elle ne se prête pas à l'extraction de l'impulsion entière pour des analyses de forme.

Pour faire face au bruit, il est cependant possible d'estimer précisément ce seuil par une mesure de bruit hors présence d'impulsions, par exemple, à l'aide d'une transformée de Fourrier (J. M. . Cardoso et al. 1999). Cette solution n'est pas satisfaisante compte tenu de l'évolution du signal et/ou du bruit dans le temps si le seuil n'est jamais réévalué. De plus, il n'est pas possible par cette méthode de réaliser le calibrage sur le terrain en présence d'éléments générant des impulsions (fond propre).

Une autre approche consiste en l'utilisation d'un second seuil de niveau d'énergie différent du premier (Craig 1993; Pasquali et al. 2007; Faisal et al. 2013).Il s'agit de l'application type d'un trigger de Schmitt dans le domaine numérique. Un seuil de détection « haut » est fixé pour la détection de l'arrivée d'une impulsion, et un seuil « bas » est fixé pour la fin de l'impulsion. L'utilisation du second seuil permet d'éviter les fausses alarmes que le bruit engendre lorsque le signal se situe autour du seuil. Les deux seuils sont fixés manuellement suite à une observation préalable du signal par l'utilisateur. Cette méthode est illustrée en Figure 3-2.

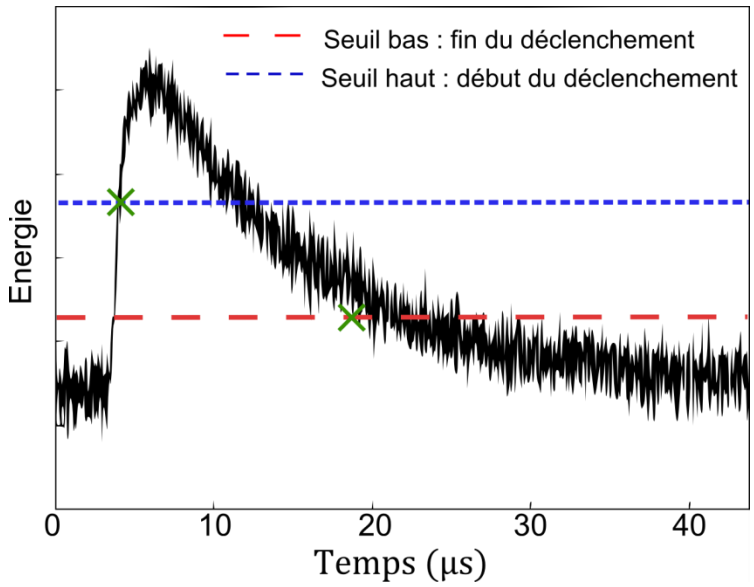



**Figure 3-2 : Méthode de déclenchement par trigger de Schmitt.**

La Figure 3-2 montre que si l'écart entre les deux seuils est supérieur à l'écart-type du bruit, alors les risques de faux déclenchements diminuent. L'utilisation de deux seuils en énergie permet nativement d'extraire les impulsions quelle que soit leur durée et est plus performante qu'une méthode à seuil simple pour une détection d'amplitude maximum. Cependant, cette méthode ne permet toujours pas l'extraction précise de l'impulsion entière, puisqu'une partie de celle-ci est encore tronquée, le paramétrage du seuil haut devant faire face au bruit.

Pour éviter de tronquer les impulsions, il est donc nécessaire d'ajouter un ou plusieurs délais temporels au processus de déclenchement. Il s'agit du principe de fenêtrage temporel.

### 3.1.2 Principe du fenêtrage

La solution actuelle pour isoler les impulsions du reste du signal est donc l'utilisation d'un fenêtrage temporel, c'est-à-dire qu'en lieu et place d'un seuil de détection de fin d'impulsion, dont les défauts ont déjà été cités, on ouvre une fenêtre à partir de la première détection de l'impulsion afin de réussir à avoir la totalité de la forme de l'impulsion (Esmaeili-Sani et al. 2011; Lee et al. 2012; Chen et al. 2013; Kim et al. 2009). C'est ce type de mise en œuvre qui est généralement utilisé pour les applications de discrimination neutron-gamma (CAEN 2010; Voltz et al. 1966; Voltz & Laustriat 1969). Cette méthode oblige l'utilisateur à régler ce paramètre manuellement et à lui donner une taille suffisamment grande pour être capable d'enregistrer les plus grandes impulsions. De ce fait, en présence de petites impulsions beaucoup d'informations stockées sont inutiles, diminuant la qualité des résultats comme illustré en Figure 3-3. De plus, si plusieurs impulsions, empilées ou non, se trouvent dans cette fenêtre, elles sont considérées et traitées comme une seule impulsion, faussant ainsi les résultats.

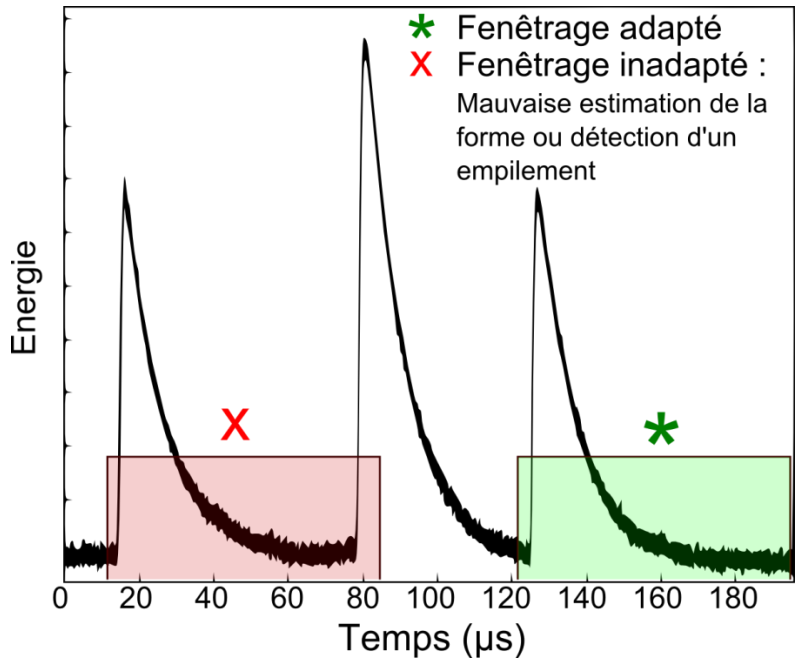



**Figure 3-3 : Problématique de la méthode de fenêtrage statique : cas d'une fenêtre trop grande.**

*A contrario*, l'utilisation d'une fenêtre de taille trop petite résulte en la troncature des impulsions les plus grandes ou des impulsions empilées comme illustré en Figure 3-4.

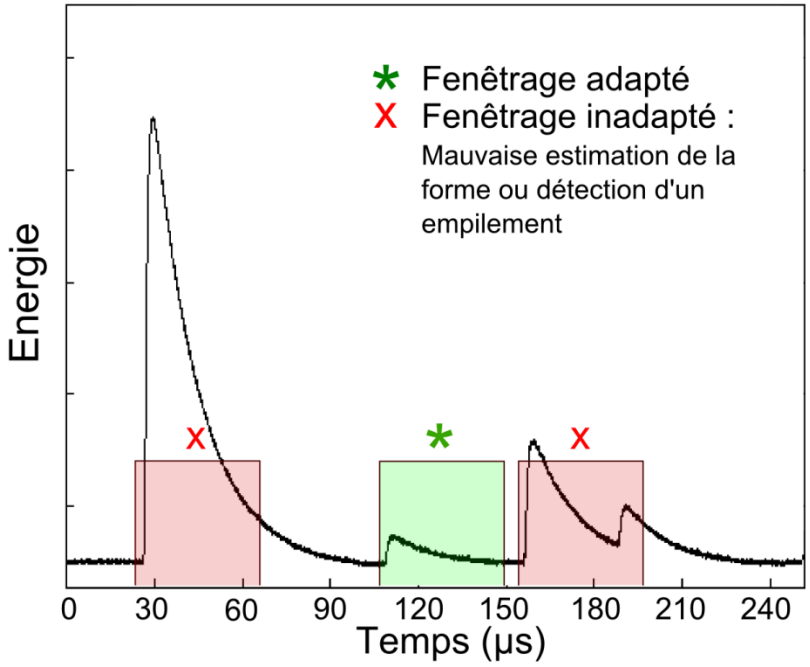



**Figure 3-4 : Problématique de la méthode de fenêtrage statique : cas d'une fenêtre trop petite.**

Le fenêtrage est donc une méthode paramétrable délicate à mettre en œuvre à cause de la variabilité de la taille des impulsions.

### 3.1.3 Mise en forme du signal et gestion des empilements

La ou les mise(s) en forme du signal permettent de s'affranchir au maximum de l'évolution du signal, de sa ligne de base (Knoll 2010; Esmaeili-Sani et al. 2011) et du bruit électronique (Whitford 2005; CAEN 2011) au cours du temps. Ce procédé augmente ainsi les performances du dispositif de déclenchement, mais engendre une déformation partielle (restauration de ligne de base (BLR)) ou totale (filtre lisseur ou anti-empilement) du signal original.

La gestion traditionnelle des empilements suit cette même logique (Warburton et al. 2000b; Radeka 1974). Un filtre passe-haut est utilisé pour réduire la durée des impulsions et ainsi augmenter la probabilité de déclencher sur des impulsions individuelles. C'est pourquoi, dans le cadre de l'utilisation d'une voie secondaire d'acquisition ayant pour objectif la conservation du signal original, les empilements, même s'ils sont détectés, sont rejetés. D'une manière générale, les empilements, même s'ils sont détectés, sont rejetés.

### 3.1.4 Méthodes à voie multiples

D'autres travaux (Warburton et al. 2000b; CAEN 2010) montrent qu'il n'est pas nécessaire de déformer le signal original pour achever une extraction des impulsions effectives. En effet, ils montrent qu'il est possible d'utiliser une voie d'acquisition secondaire (appelée voie rapide) sur laquelle des traitements de mises en forme seront effectués mais utilisés pour déclencher l'extraction du signal sur la voie primaire. De cette manière, les traitements nécessaires au déclenchement de l'extraction d'un point de vue temporel n'affectent pas le signal original. De plus, l'utilisation d'une ligne à retard permet également de connaître le futur/passé du signal et ainsi améliorer l'extraction des impulsions. La Figure 3-5 illustre ce principe de déclenchement. Le signal original est filtré sur une seconde voie de manière à faciliter la détection de l'impulsion. Puis, afin de pouvoir extraire également la période entre le début du déclenchement et le début réel du front montant de l'impulsion, un délai connu est appliqué sur une troisième voie et est pris en compte dans la durée du fenêtrage afin d'extraire la totalité de l'impulsion.

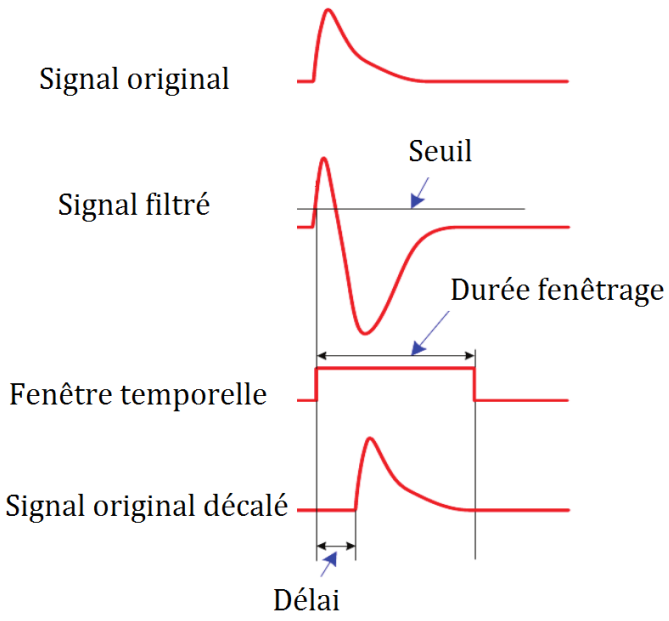



**Figure 3-5 : Méthode de déclenchement à voies multiples.**

La méthode à voies multiples permet de pallier les défauts apportés par la mise en forme du signal original mais reste sensible aux problématiques de fenêtrage statique citées précédemment.

### 3.1.5 Conclusion intermédiaire

Aucune des méthodes présentées dans la littérature n'est parfaite car adaptée en général à des besoins applicatifs spécifiques. C'est pourquoi, dans ce chapitre, nous proposons une méthode de déclenchement adaptée aux contraintes de seuillage et de fenêtrage citées.

## 3.2 Proposition d'un extracteur dynamique d'impulsion

Cette partie concerne la proposition d'un extracteur d'impulsion s'affranchissant des problématiques traditionnelles de déclenchement. L'algorithme peut être décomposé en deux parties. La première concerne l'estimation dynamique d'un seuil de déclenchement immune au bruit. La seconde concerne la définition d'une fenêtre temporelle qui s'ajuste dynamiquement en fonction des impulsions, permettant leur extraction en fonction de leurs durées respectives.

### 3.2.1 Evaluation dynamique du seuil de détection

L'estimation dynamique du seuil de détection vise à définir un seuil au plus proche du bruit afin de permettre la détection des impulsions de faible énergie tout en évitant les fausses alarmes.

Deux types de bruit sont traditionnellement rencontrés dans la mesure de radioactivité. Le premier est le bruit thermique qui est considéré comme un bruit blanc gaussien. Le second est le bruit de grenaille qui souvent est négligeable en comparaison du bruit thermique (Texas Instruments 2008). La prédominance du bruit gaussien permet de modéliser le bruit par une loi normale de distribution pour l'estimation du seuil. Cette approximation permet alors l'utilisation de la mesure de dispersion par le calcul de l'écart-type (Morsy & Von Ramm 1999). L'estimation de l'amplitude du seuil est déterminée par l'équation 3-1, avec $\mu_t$ la valeur moyenne du signal et $\sigma_t$ son écart-type. $C$ est une variable d'ajustement utilisée pour contrôler la valeur du seuil selon une loi normale de distribution.

$$seuil = \mu_t + C \cdot \sigma_t \qquad \textbf{3-1}$$

Il existe plusieurs méthodes d'estimation de l'écart-type dans la littérature (Pastor & Socheleau 2012; Jiang et al. 2013). Pour notre proposition, nous utilisons la formule conventionnelle définie dans l'équation 3-2, qui consiste en la racine carrée de la variance. Avec $s(t)$ une portion du signal sans présence d'impulsion et $E(s(t))$ l'espérance mathématique de $s(t)$.

$$\sigma_t = \sqrt{E(s(t)^2) - E(s(t))^2} \qquad \textbf{3-2}$$

Le signal d'entrée utilisé pour le déclenchement est la dérivée du signal original $\frac{ds(t)}{dt}$, ce qui est nécessaire pour l'étape de fenêtrage proposée ultérieurement (3.2.2). Elle permet d'augmenter la probabilité de détection des empilements et de soustraire la partie continue $\mu_t$ du signal. De ce fait, les équations 3-1 et 3-2 peuvent être reformulées par 3-3 et 3-4.

$$seuil' = C \cdot \sigma_t' \qquad \textbf{3-3}$$

$$\sigma_t{}' = \sqrt{E\left(\left(\frac{ds(t)}{dt}\right)^2\right) - E\left(\frac{ds(t)}{dt}\right)^2} \qquad \textbf{3-4}$$

Avec $C$, la variable d'ajustement d'une loi normale de distribution, l'estimation du seuil permet l'utilisation, pour notre proposition, d'un intervalle de confiance respectant la règle du *68-95-99,7*. Par exemple, si $C = 1$, 68 % de l'amplitude du bruit varie entre $-\sigma_t{}'$ et$\sigma_t{}'$, alors qu'avec $C = 3$, 99,7 % des valeurs d'amplitudes de bruit varie entre $-3\sigma_t{}'$ et $3\sigma_t$. Ce qui signifie que, statistiquement, seul 0,27 % du bruit sera à l'origine de fausses alarmes. La Figure 3-6 illustre ce principe.

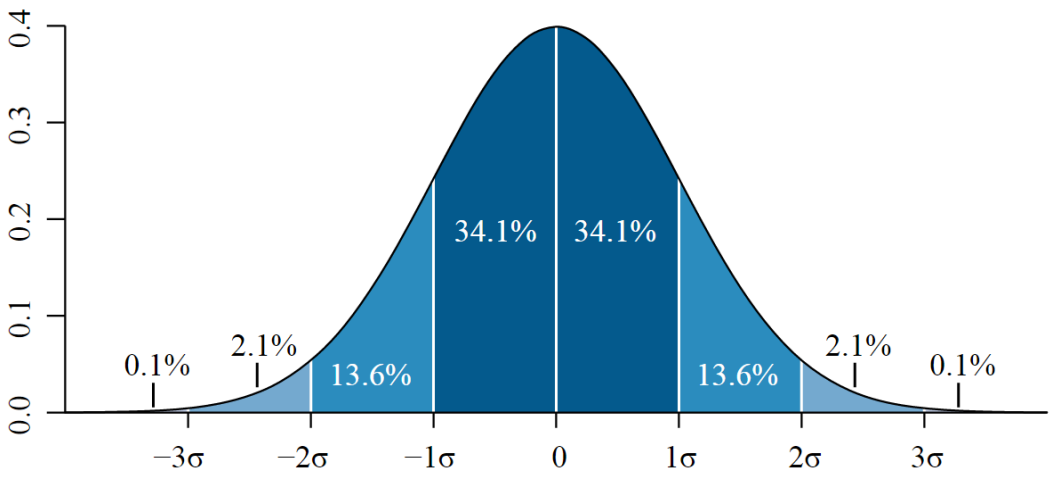



**Figure 3-6 : Représentation graphique d'une loi normale. Chaque bande colorée a la largeur d'un écart-type (*source : Wikipédia*).**

### 3.2.2 Fenêtrage adaptatif et détection des empilements

La seconde partie de notre approche consiste en l'extraction dynamique des impulsions individuelles et du *tag* des empilements en fonction de leur durée afin de n'obtenir que la partie utile du signal. Cette méthode est basée sur l'estimation du seuil obtenue par 3-3. Dans l'objectif d'adapter dynamiquement la taille de la fenêtre temporelle, nous utilisons une décroissance exponentielle comme approximation de la décroissance d'une impulsion. De précédents travaux montrent que cette approximation correspond bien aux caractéristiques réelles des impulsions (Knoll 2010). De ce fait, il est possible de calculer une constante de temps nommée $\tau$ en utilisant les caractéristiques mathématiques d'une décroissance exponentielle comme illustré dans la Figure 3-7.

L'amplitude d'une décroissance exponentielle atteint 37 % ($\exp^{-1}$) de l'amplitude du pic après une durée $\tau$, et plus de 99 % de l'énergie de l'impulsion est contenue sur $5\tau$. D'autres approximations d'une impulsion peuvent être utilisées.

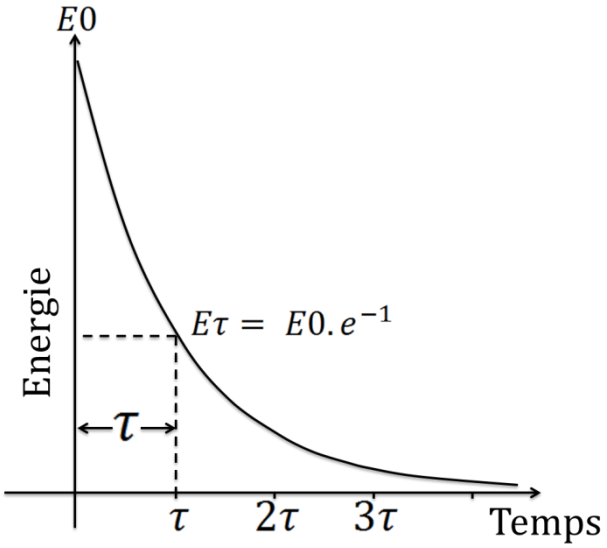



**Figure 3-7 : Caractéristiques mathématiques d'une décroissance exponentielle.**

La durée de notre fenêtrage dépend donc de l'amplitude de l'impulsion. Il est possible de connaître l'amplitude maximum d'une impulsion sans avoir à attendre d'avoir observé l'intégralité de celle-ci grâce à l'utilisation de la dérivée. Une fois que la dérivée devient négative cela signifie que l'amplitude maximum de l'impulsion est atteinte. Cette caractéristique permet de calculer $\tau$ et donc de prédire la fin de la portion la plus large de l'impulsion comme illustré en Figure 3-8.

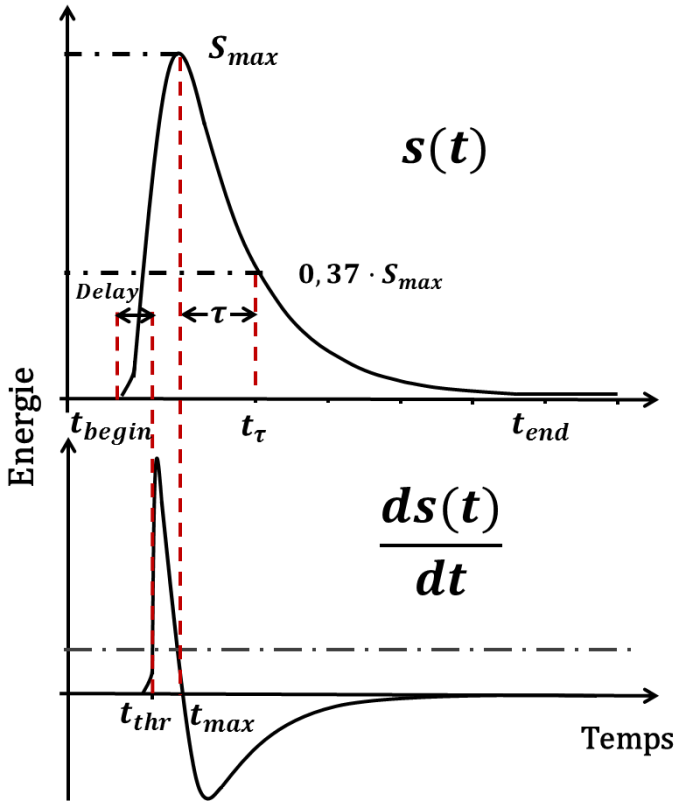



**Figure 3-8 : Méthode de fenêtrage basée sur la caractéristique exponentielle d'une impulsion et sur l'utilisation de la dérivée.**

Pour exploiter temporellement cette approche, nous définissons deux fonctions 3-5 et 3-6 retournant une date pour une amplitude donnée :

$$g\big(s(t)\big) = \begin{cases} t, quand\ \dfrac{ds(t)}{dt} < 0 \\ 0, \quad sinon \end{cases} \qquad \textbf{3-5}$$

$$h\left(\frac{ds(t)}{dt}\right) = \begin{cases} t, quand\ \dfrac{ds(t)}{dt} \geq 0\ et\ \dfrac{d^2s(t)}{dt^2} > 0 \\ 0, \quad sinon \end{cases} \qquad \textbf{3-6}$$

Comme illustré dans la Figure 3-8, nous considérons également le jeu d'événements temporels suivant :

- $Delay$, la constante de préhistoire ;
- $t_{Thr}[p] = h(Thr), \forall p\ \epsilon\ \mathbb{N}, p \geq 0$ , pour désigner les instants de déclenchement par comparaison du seuil et de la dérivé du signal $\frac{ds(t)}{dt}$ ;
- $t_{Begin}[p] = t_{Thr}[p] - delay, \forall p\ \epsilon\ \mathbb{N}, p \geq 0$, pour désigner les instants où les impulsions de $s(t)$ détectées débutent ;
- $t_{Max}[p] = h(0), \forall p\ \epsilon\ \mathbb{N},\ p \geq 0$, pour désigner les instants des amplitudes maximum des impulsions de $s(t)$ ;
- $t_{\tau}[p] =\ g(s(t_{Max}[p]).e^{-1}), \forall p\ \epsilon\ \mathbb{N},\ p \geq 1$, pour désigner les instants où les amplitudes maximum $e^{-1}$ décroissent sur $s(t)$ ;
- $t_{End}[p] = t_{Max}[p] + 5(t_{\tau}[p] -\ t_{Max}[p]),\ \forall p\ \epsilon\ \mathbb{N},\ p \geq 1$, pour désigner les instants où les impulsions de $s(t)$ détectées se terminent.

Enfin, il est possible d'extraire les impulsions du reste du signal grâce à la fonction porte $B_{Pulses}(t)$ présentée dans 3-7 avec $H(t)$ la fonction de Heaviside.

$$B_{Pulses}(t) = \bigcup_{i=1}^{p-1} \big[H\big(t - t_{Begin}[i]\big) - H(t - t_{End}[i])\big] \qquad \textbf{3-7}$$

Finalement, le signal ne contenant que les impulsions $s_{Pulses}(t)$ est défini par l'équation 3-8. Un exemple illustrant le procédé est présenté en Figure 3-9.

$$s_{Pulses}(t) =\ B_{Pulses}(\mathrm{t}) \cdot s(t) \qquad \textbf{3-8}$$

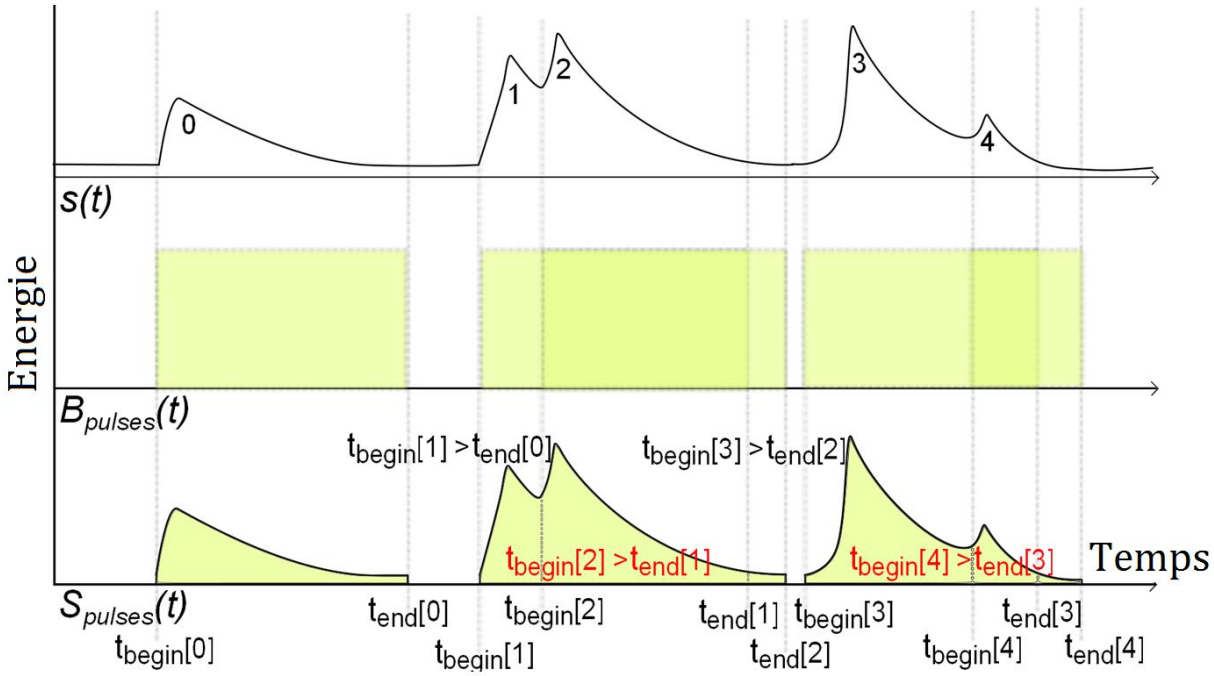



**Figure 3-9 : Représentation des fonctions $s(t)$, $B_{Pulses}(t)$ et $s_{Pulses}(t)$ pour des cas de détection d'impulsions simples et empilées.**

Comme présenté en Figure 3-9, la façon la plus simple de *tagger*, rejeter ou extraire uniquement les empilements pour les impulsions $p$ et $p+1$ est de vérifier la condition définie par l'équation 3-9.

$$f(p) = \begin{cases} 1, & t_{End}[p] > t_{Begin}[p+1] \\ 0, & t_{End}[p] \leq t_{Begin}[p+1] \end{cases} \quad \textbf{3-9}$$

La fonction porte $B_{Pileup}(t)$ présentée dans l'équation 3-10 permet l'extraction des empilements.

$$B_{Pileup}(t) = \bigcup_{i=1}^{p-1} \left[H\left(t - t_{Begin}[i]\right) - H(t - t_{End}[i+1])\right] \cdot f(i) \quad \textbf{3-10}$$

Finalement, il devient possible d'obtenir le signal avec uniquement les empilements $s_{Pileup}(t)$ ou avec uniquement les impulsions individuelles $s_{SinglePulses}(t)$ comme respectivement représenté par les équations 3-11et 3-12.

$$s_{Pileup}(t) = B_{Pileup}(\mathrm{t}) \cdot s(t) \quad \textbf{3-11}$$

$$s_{SinglePulses}(t) = (B_{Pulses}(t) - B_{Pileup}(\mathrm{t})) \cdot s(t) \quad \textbf{3-12}$$

## 3.3 Simulations et résultats

### 3.3.1 Implémentation

Notre proposition d'extracteur d'impulsions transmet les impulsions individuelles vers des traitements numériques des impulsions. La Figure 3-10 présente le schéma bloc de cette approche. Comme présenté en 3.2, l'implémentation proposée de notre système de déclenchement peut être décomposée en deux parties décrivant respectivement l'implémentation numérique de l'estimateur de seuil (3.3.1.1) et de l'extracteur dynamique d'impulsions (3.3.1.2).

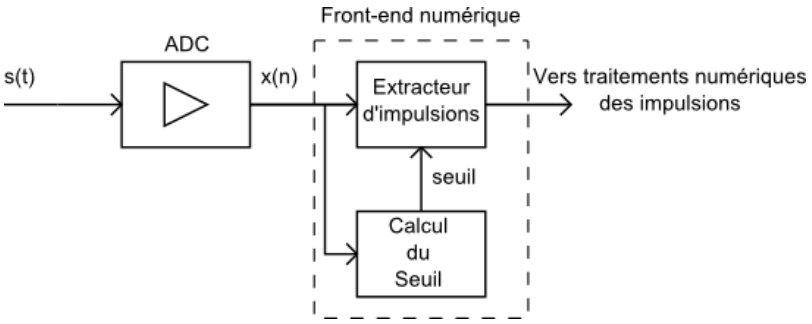



**Figure 3-10 : Schéma bloc du système.**

#### 3.3.1.1 Implémentation de l'estimateur de seuil

Le seuil de détection de la dérivée du signal est estimé par calcul de l'écart-type du signal en l'absence d'impulsions (e.g. uniquement le bruit) comme présenté dans l'équation 3.3. Dans le cadre d'une implémentation numérique, il est nécessaire d'appliquer ce traitement sur une portion du signal uniquement, et de répéter cette action de manière cyclique. Pour éliminer la présence éventuelle d'impulsions, une version lissée du signal est soustraite au signal original. Cependant, les contraintes d'implémentation réelles ne permettent pas l'utilisation d'un filtre idéal anti-bruit. C'est pourquoi ce filtre peut être approximé par l'utilisation d'un filtre à Réponse Impulsionnelle Infinie (IIR) conventionnel. Ce filtre doit être défini avec attention pour éliminer un maximum de bruit. Dans notre implémentation nous utilisons, à titre d'exemple, un filtre de type Moyenne Glissante Exponentielle (EMA), qui offre de bons résultats s'il est appliqué au signal utilisé pour les tests, et décrit en 3.3.2.1.

Notre proposition d'implémentation numérique dépend des paramètres suivants :

- $x(n)$, le signal discret d'entrée ;
- $n$, le numéro de l'échantillon ;
- $N$, le nombre d'échantillons utilisé pour le calcul de la dérivée ;
- $M$, le nombre d'échantillons utilisé pour le calcul de l'écart-type ;
- $X$, le paramètre d'évaluation du seuil ;
- $\alpha$, la constante de lissage du filtre EMA ;
- $\sigma$, l'écart-type de toutes les observations.

Pour illustrer l'implémentation, nous choisissons comme exemple une portion de signal de taille $M$illustrée en Figure 3-11.

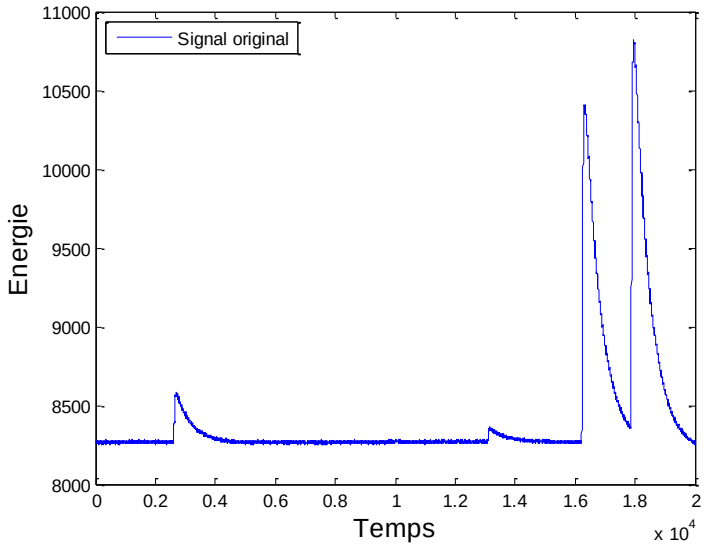



**Figure 3-11 : Exemple d'une portion du signal utilisée pour le calcul du seuil.**

La première étape consiste donc en la soustraction du signal original, sur une voie secondaire, avec le signal original lissé. Le lissage du signal original par l'utilisation du filtre EMA est présenté dans l'équation 3-13 et est illustré en Figure 3-12 pour la portion du signal choisie en exemple.

$$y(n) = \alpha.x(n-1) + (1-\alpha)\cdot y(n-1) \qquad \textbf{3-13}$$

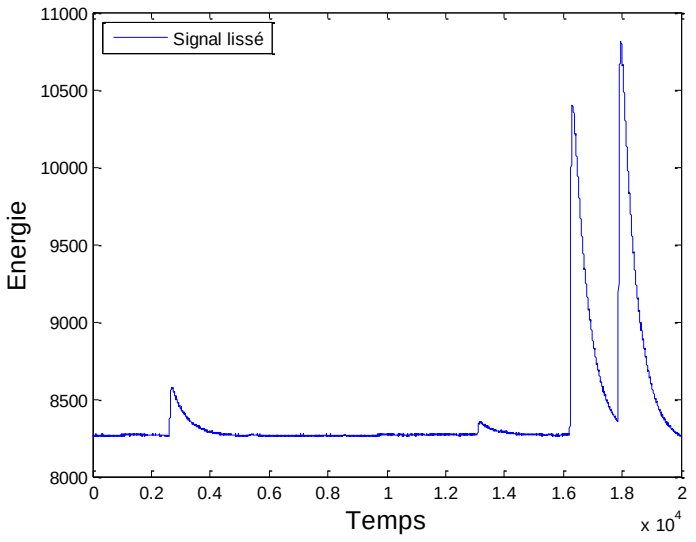



**Figure 3-12 : Portion du signal original lissé par filtrage EMA.**

La différence entre le signal original et le signal lissé est présentée dans l'équation 3-14 et illustrée en Figure 3-13.

$$y_{diff}(n) = x(n) - y(n) \qquad \textbf{3-14}$$

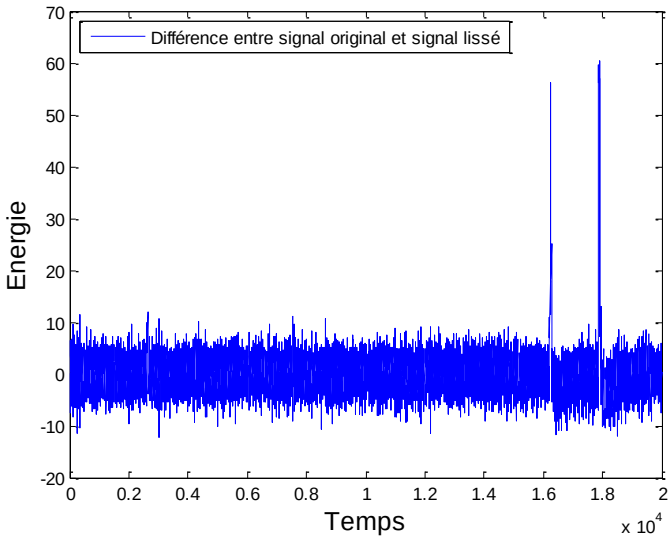



**Figure 3-13 : Résultat de la différence entre la portion du signal original et le signal lissé.**

Le filtre n'étant pas idéal, du signal résiduel issu des impulsions les plus larges est encore présent, même si son impact est amoindri pour un calcul d'écart-type sur cette portion. De plus, une partie du bruit à été dégradée durant la soustraction. L'étape suivante consiste à dériver le signal résultant de cette soustraction. La dérivée est la voie utilisée pour le déclenchement du début de notre méthode d'extraction comme présenté dans le chapitre précédent. La dérivée du signal résultant est calculée dans 3-15 et illustrée en Figure 3-14. $N$ doit être suffisamment petit pour permettre une bonne approximation de la pente de la tangente. Dans notre implémentation, $N$ correspond au nombre d'échantillons de la moyenne des fronts montants des impulsions.

$$y'_{diff}(n) = \frac{(y_{diff}(n) - y_{diff}(n-N))}{N} \qquad \textbf{3-15}$$

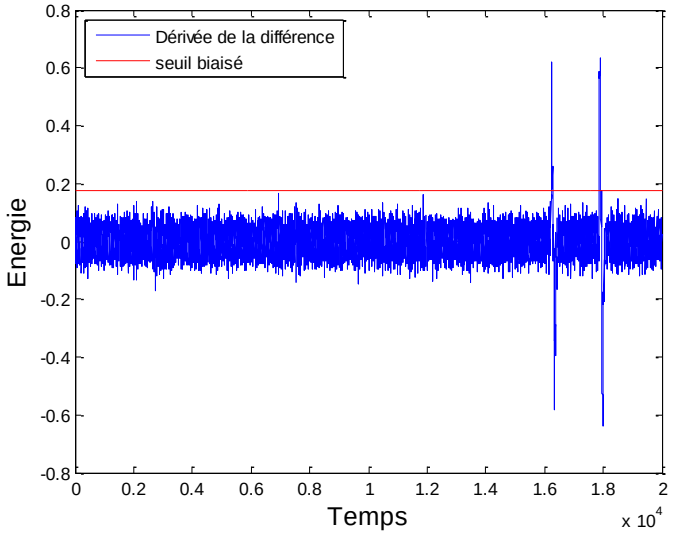



**Figure 3-14 : Dérivée du résultat de la soustraction sur la portion du signal. Un premier seuil dit « biaisé » est alors calculé.**

Un premier calcul de seuil dit « biaisé » est évalué par l'utilisation de l'écart-type sur la dérivée du signal obtenu par la différence entre le signal original et sa version lissée. Le calcul de ce seuil est alors effectué sur le nombre d'observations $M$ et est présenté dans les équations 3-16 et 3-17. Le résultat de ce seuil pour la portion du signal exemple est illustré en Figure 3-14.

$$\sigma_1 = \sqrt{\left(\frac{1}{M}\sum_{n=1}^{M} y'_{diff}(n)^2\right)} \qquad \textbf{3-16}$$

$$seuilBiaisé = X \cdot \sigma_1 \qquad \textbf{3-17}$$

Ce seuil biaisé est ensuite utilisé pour éliminer, par comparaison, les impulsions présentes sur la dérivée du signal original. Dans un premier temps, la dérivée du signal original est calculée dans 3-18 et illustrée en Figure 3-17.

$$y'(n) = \frac{(y(n) - y(n-N))}{N} \qquad \textbf{3-18}$$

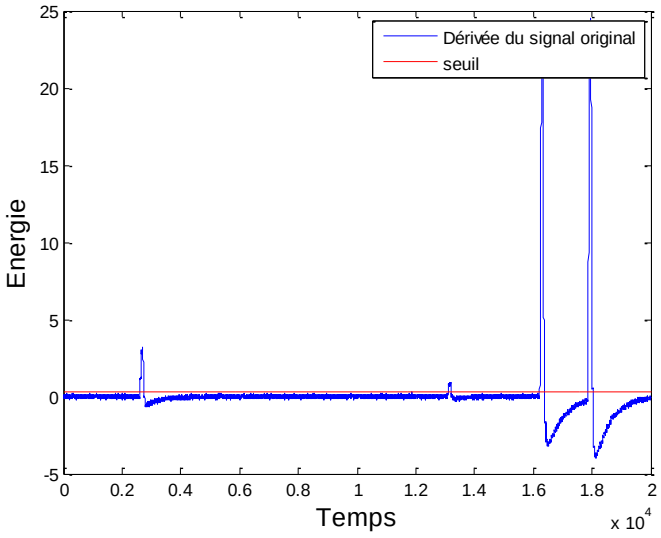



**Figure 3-15 : Dérivée de la portion du signal original.**

Ensuite, le seuil biaisé est utilisé pour ne pas prendre en compte les impulsions lors du calcul de l'écart-type sur le signal original dérivé. La Figure 3-16 illustre la portion du signal original dérivée débarrassée de ses impulsions. Le signal obtenu contient donc uniquement le bruit utilisé pour l'estimation du seuil qui sera exploité par notre dispositif de fenêtrage. La valeur du seuil de détection est calculée grâce à l'écart-type des valeurs absolues de la dérivée sous la valeur du seuil biaisé comme présenté dans 3-19 et 3-20. Ce seuil est donc réévalué tous les $M$ échantillons. Le résultat du seuil final apparaît sur la Figure 3-15.

$$\sigma_2 = \sqrt{\left(\frac{1}{M}\sum_{n=1}^{M} y'(n)^2\right)}, \forall |y'(n)| < seuilBiaisé \quad \textbf{3-19}$$

$$seuil = X \cdot \sigma_2 \quad \textbf{3-20}$$

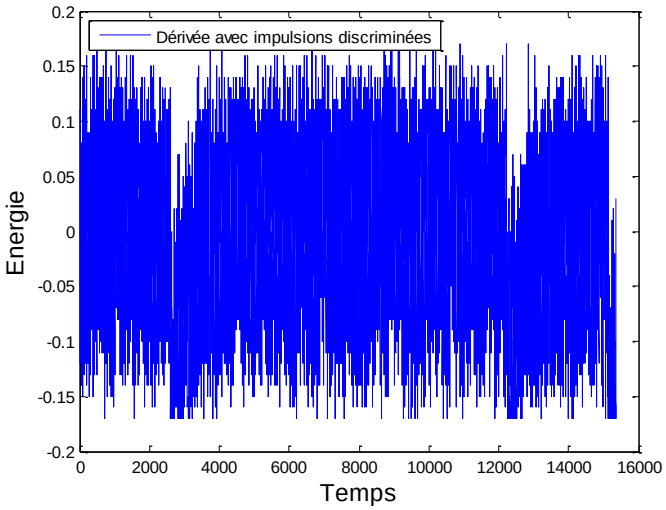



**Figure 3-16 : Portion du signal original dérivé avec impulsions discriminées.**

L'ensemble de ces étapes est ensuite réitéré tout les $M$échantillons. Il s'agit d'un processus qui fonctionne en parallèle de l'extracteur. Il est donc nécessaire de calculer le seuil une première fois avant de démarrer le processeur d'extraction. A ce stade, la valeur de la variable $Thr$ du modèle mathématique présenté dans le chapitre précédent est déterminée.

### 3.3.1.2 Implémentation de l'extracteur dynamique d'impulsions

Le fenêtrage dynamique permettant l'extraction des impulsions peut être représenté par une machine à états finie. La machine est représentée graphiquement dans la Figure 3-17. Dans cette partie, les états font références à la Figure 3-17. Une fois le début de l'acquisition lancé (état 1), la première étape consiste à calculer la dérivée du signal acquis, tandis que les différents échantillons du signal original numérisé sont mémorisés. Cette dernière opération est réalisée à l'aide d'une mémoire-tampon circulaire de taille programmable (état 2). La dérivée numérique est ensuite comparée au seuil de détection dynamique calculé dans 3-20. Si le seuil est dépassé (état 3), indiquant la détection d'une impulsion, la valeur se situant en queue de la mémoire-tampon est mémorisée (état 4). Suite à la détection de l'impulsion, une opération de remplissage d'un tableau mémorisant les différents échantillons du signal acquis est réalisée (état 5) ; en prenant, par exemple, les premières valeurs enregistrées dans la mémoire tampon qui sont antérieures au déclenchement du seuil, puis pour valeurs suivantes celles correspondant à l'impulsion. Parallèlement, il est procédé au suivi de la dérivée pour localiser le pic de l'impulsion lorsque la dérivée change de signe (état 6). Une fois le pic atteint, le calcul de l'amplitude théorique de $\tau$ est réalisé. Ainsi, il est possible de calculer la durée $\tau$ jusqu'à l'atteinte de cette amplitude (état 7).

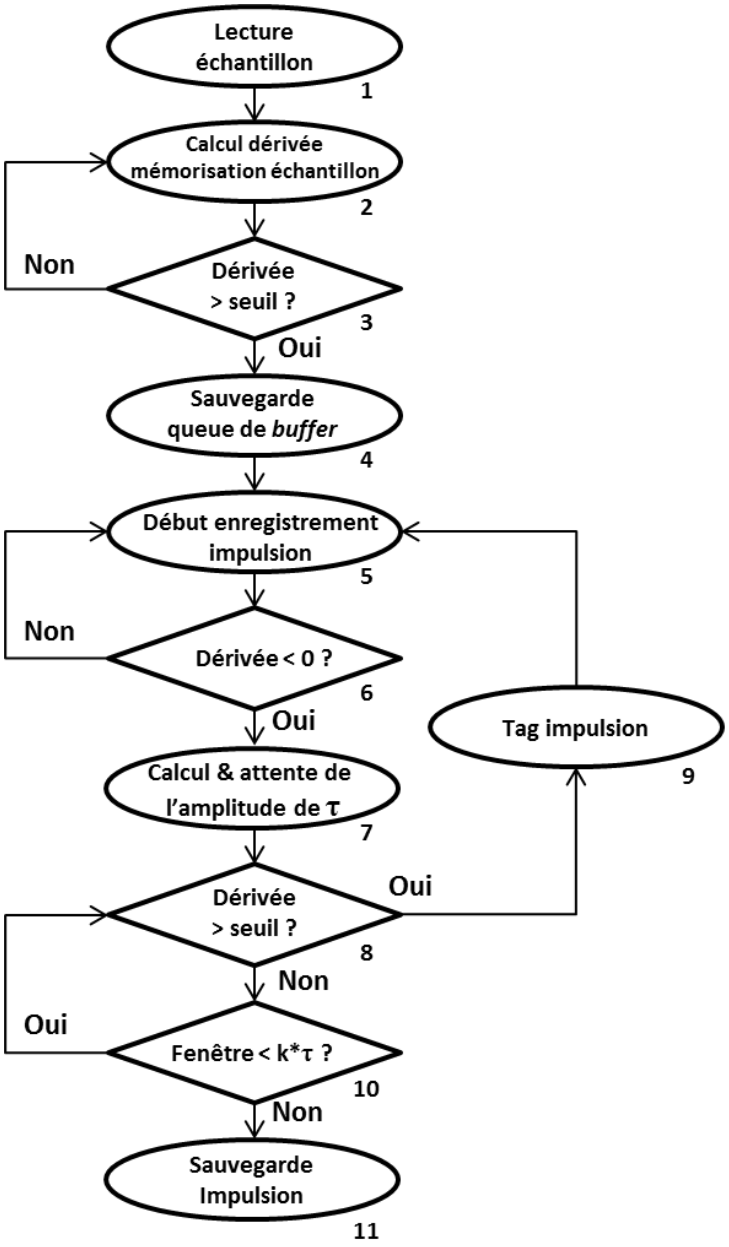



**Figure 3-17 : Machine à état de l'algorithme de fenêtrage dynamique.**

L'amplitude du pic de l'impulsion peut être déterminée en comparant l'amplitude de l'échantillon du signal acquis correspondant à la valeur de queue de la mémoire-tampon, référence de la ligne de base. L'opération de remplissage du tableau continue alors jusqu'à atteindre la durée calculée par un multiple $k$ de $\tau$ (état 10). $k = 5$ dans notre implémentation, ce qui correspond à la totalité théorique d'une impulsion comme définie dans 3.2.2. Durant la période d'attente de $\tau$, puis de $5\tau$, le suivi de la dérivée est toujours réalisé pour procéder à la détection des empilements (état 8). Lorsque la dérivée passe au-dessus du seuil, une opération de marquage indique que le tableau mémorisant l'impulsion contient un empilement. Les différentes étapes sont en outre réitérées jusqu'au calcul d'un nouveau $\tau$. Cependant, l'amplitude théorique est calculée à partir de la première valeur de la queue de la mémoire-tampon précédemment enregistrée. De cette manière, nous assurons une fenêtre suffisamment large pour mémoriser l'empilement en entier. Une fois la condition d'attente de la durée $5\tau$ remplie, l'impulsion ou l'empilement tagué est enregistré.

#### 3.3.1.3 Variantes d'implémentation possibles

L'approche proposée a été implémentée et testée avec certaines variantes présentées ci-dessous. En tenant compte de la présence du bruit, et du risque d'empilement croissant si la fenêtre est trop grande, il est possible d'ajuster l'arrêt du fenêtrage à $k = 3$. En effet, entre $3\tau$ et $5\tau$, le signal utile peut être confondu avec le bruit. Il est également possible d'inclure le seuil calculé sur l'écart-type comme condition d'arrêt supplémentaire afin de s'assurer que la dérivée du signal n'est plus distinguable du bruit en accord avec la probabilité associée à une loi normale de distribution.

### 3.3.2 Protocole expérimental

#### 3.3.2.1 Description du signal d'entrée

Les données utilisées pour les tests ont été obtenues avec une source d'Am-241 et un détecteur de type cristal-puits NaI(Tl). Le détecteur 4πγ NaI(Tl) puits utilisé pour les mesures présente les caractéristiques géométriques suivantes : 152 mm de diamètre et 127 mm de hauteur de cylindre, avec un trou coaxial de 21 mm de diamètre et 47,5 mm de hauteur. Le détecteur à scintillation est couplé à un photomultiplicateur type EMI 9791 ; le système est logé dans un blindage composé d'une couche de 1 cm d'épaisseur interne de Cu et une couche extérieure de 15 cm d'épaisseur de Pb afin de réduire le bruit de fond (de l'ordre de 16-20 $s^{-1}$). Les sources mesurées ont été préparées par évaporation de fractions aliquotes de la solution radioactive en utilisant la technique du pycnomètre (masse de 10 à 30 mg) ; la gouttelette radioactive a été déposée au centre du support de source (feuille de Mylar : 18 µm d'épaisseur, 18 mm de diamètre). Afin d'éviter une perte du matériel radioactif ou une contamination potentielle, le dépôt séché a été recouvert d'un ruban adhésif (45 µm d'épaisseur). L'échantillon a ensuite été placé au fond du puits de détection pour la mesure. La figure 10 montre le détecteur NaI(Tl) puits.

**Figure 3-18 : Détecteur cristal NaI(Tl), muni d'un puits au fond duquel l'échantillon à mesurer est placé (4π sr) *(source : LNHB)*.**

Le signal sorti du préamplificateur a été directement numérisé à 125 MHz avec une résolution de 16 bits par échantillon, puis directement mémorisé. La Figure 3-19 présente une fraction du signal utilisé pour les tests.

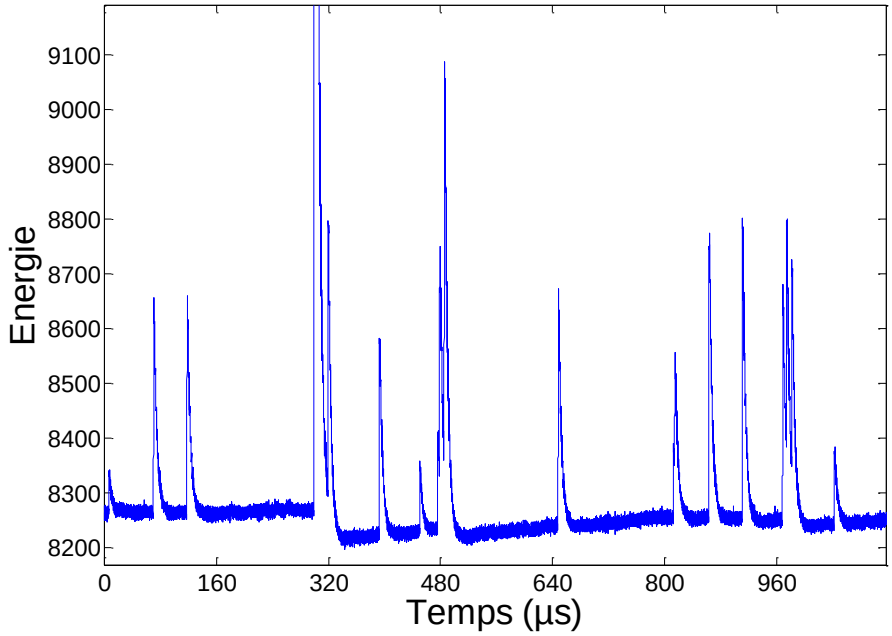



**Figure 3-19 : Partie du signal utilisé pour les tests qui montre une grande variabilité de taille des impulsions, des empilements, et la déformation de la ligne de base.**

Pour tester notre algorithme, le logiciel MATLAB a été choisi. De plus, nous avons choisi de travailler sur un signal d'une durée de 2 secondes, i.e. 512 Mo de données, ce qui représente plus de 22000 impulsions. En effet, ici, l'objectif n'est pas de tester une application complète. Plus la durée de la mesure est élevée, plus la probabilité de faux positifs augmente, ce qui empêcherait l'observation des différences entre les méthodes de déclenchement étudiées. Ce temps de mesure court est nécessaire pour tester les méthodes en profondeur, ce qui signifie la nécessité d'une bonne qualité de déclenchement, de fenêtrage et de gestion des empilements. En outre, les mesures courtes sont représentatives des applications de surveillance de rayonnement au moyen de dispositifs portables. Ces mesures obligent le système détection à rapidement obtenir des résultats permettant d'identifier une source ou d'atteindre un seuil d'alarme. Donc, sur une courte période, être capable de « voir » toutes les informations acquises par le détecteur est primordial. L'Am-241 émet principalement deux types de gamma, respectivement à 17,045 keV et 59,5409 keV, avec une probabilité d'émission de 37,66 % et 35,92 %. Ces énergies sont basses, et peuvent donc rapidement discriminer les méthodes de déclenchements inadaptées pour des mesures proches du bruit. En outre, le taux de comptage est suffisamment élevé pour observer de nombreux empilements, ce qui permet aux méthodes intégrant une gestion des empilements d'obtenir de manière théorique de meilleurs résultats. Enfin, le signal montre une grande variabilité en termes de durée des impulsions (entre 500 et 6000 échantillons), ce qui est nécessaire pour évaluer la qualité des différents fenêtrages testés, ainsi que des variations de ligne de base importantes.

#### 3.3.2.2 Critères de comparaison

Le critère utilisé est la résolution avec une largeur à mi-hauteur (FWHM). Nous avons décidé de comparer séparément deux types de spectres en énergie. Le premier est constitué à partir de l'amplitude des impulsions, le second à partir de l'aire de celles-ci. Ceci est réalisé pour éviter la différence de largeur de canal relative en termes d'énergie. « *Count* » représente la surface nette du

pic avec les incertitudes associées. La meilleure valeur possible est le « *count* » le plus élevé en corrélation avec la plus grande « contribution », qui représente la surface nette du pic par rapport au nombre total de coups avec les incertitudes associées. « *Error* » est la comparaison entre « contribution » et la probabilité d'émission. Ce critère reflète l'instabilité des différents algorithmes par rapport à leurs performances de résolution. Enfin, le meilleur résultat possible est une bonne résolution en corrélation avec une erreur faible.

Pour évaluer la qualité de notre méthode de déclenchement, nous avons donc choisi d'utiliser deux types de spectres d'énergie. Le premier, nous l'avons dit, est un spectre en énergie fonction des amplitudes des impulsions. Il est utilisé pour discriminer les algorithmes en termes de mauvais déclenchement, mise en forme d'impulsions, BLR, et gestion des empilements. Le second consiste à mesurer la surface de chaque impulsion détectée. En effet, l'aire d'une impulsion peut également être utilisée pour la mesure de l'énergie (charge totale). Ainsi, l'aire est représentative de la qualité du fenêtrage. Meilleure est la résolution obtenue, meilleure est la méthode de déclenchement pour l'analyse de forme des impulsions, comme pour la discrimination neutron-gamma. Pour chaque méthode dépendante de paramètres, nous avons testé différentes valeurs de paramètres afin d'obtenir la meilleure résolution possible. De plus, nous ajoutons qu'aucune amélioration n'a été apportée aux algorithmes mis en œuvre pour améliorer l'estimation de l'énergie, afin de comparer équitablement les différentes méthodes.

#### 3.3.2.3 Méthodes évaluées

Les méthodes de déclenchement choisies pour notre étude comparative sont les suivantes :

- méthode 1 : simple seuil (paramétré) pour le début et de la fin du déclenchement (Knoll 2010) ;
- méthode 2 : seuil double (trigger de Schmitt) pour le début et la fin du déclenchement, avec deux ensembles de seuil comparés, Méthode 2 (b) pour des seuils plus bas (Craig 1993; Pasquali et al. 2007; Elhanany et al. 1999) ;
- méthode 3 : BLR, simple seuil pour le début de déclenchement et fenêtre de taille fixe pour l'arrêt (Faisal et al. 2013) ;
- méthode 4 : BLR, Discriminateur à fraction constante (CFD) pour le début de déclenchement, et fenêtre de taille fixe pour l'arrêt. CFD pour la réjection d'empilement (Kim et al. 2009) ;
- méthode 5 : BLR, CFD pour le début de déclanchement, fenêtre de taille fixe pour l'arrêt et utilisation d'une voie rapide (dérivée) pour la rejection d'empilement (CAEN 2011);
- méthode 6 : BLR, voie rapide (dérivée) et seuil calculé pour le début de déclenchement, la fenêtre de taille fixe pour l'arrêt et voie rapide (dérivée) pour la rejection d'empilement (CAEN 2010) ;
- méthode 7 : La méthode proposée, voie rapide (dérivée) avec seuil dynamique et automatique pour le début de déclenchement, fenêtre adaptative pour l'arrêt et voie rapide (dérivée) de rejet d'empilement.

### 3.3.3 Résultats

Le Tableau 3-1 et le Tableau 3-2 présentent les résultats de l'étude des amplitudes et des aires des impulsions acquises selon les différentes méthodes testées. De prime abord, l'exhaustivité de notre méthode nous permet d'obtenir la meilleure résolution sur les deux types de spectres. En outre, l'erreur associée montre que le procédé est stable comparativement à la probabilité d'émission. Pour ces raisons, nous pouvons évaluer la capacité de notre méthode à ne pas détecter de faux positifs, malgré l'utilisation d'un seuil « proche du niveau du bruit ». Si l'on insiste sur le spectre obtenu à partir de l'aire des impulsions, on constate que la différence de résolution entre notre déclencheur et les méthodes de la littérature montrent que notre approche est en mesure d'estimer la longueur d'une impulsion avec précision.

La suite de l'étude des résultats s'intéresse aux autres méthodes dans l'ordre. Pour la méthode 1, la résolution du pic à 17,05 keV est trop pauvre pour être calculée. En effet, sans ajouter de fenêtrage, ou un deuxième seuil, si le seuil unique est trop proche du bruit, le taux de faux positifs explose. Par conséquent, la contribution du bruit ajoutée à la contribution présente dans la partie Compton du spectre est fusionnée avec le premier photo-pic. Cependant, la méthode 1 fournit une résolution acceptable du pic à 59,54 keV malgré son incapacité à gérer les distorsions de ligne de base ou les empilements. La Méthode 2 montre une plus grande aisance à éviter la détection de faux positifs par l'utilisation d'un second seuil, mais la résolution du premier pic reste faible. Cependant, en raison d'une bonne erreur et une résolution acceptable, cette méthode semble garantir un traitement équitable entre les performances et la complexité de mise en œuvre. Avec un seuil bas proche bruit, la méthode 2 (b) montre une meilleure capacité à détecter le premier pic, mais montre une mauvaise stabilité en raison du taux d'erreur pour le second pic. La méthode 3 montre qu'une BLR combinée avec un seuil simple permet l'amélioration de la résolution du premier pic. Cependant, de par son traitement, de nombreuses impulsions sont rehaussées ou tronquées, ce qui augmente le taux d'erreur sur le spectre en aire. Nous pouvons également voir que la BLR a une mauvaise influence sur les résolutions obtenues pour les basses énergies, comme la différence entre les méthodes 6 et 7 l'illustre. Pour le spectre en amplitude obtenu avec les méthodes 4 et 5, on constate que la résolution ne peut pas être prise en compte à cause du nombre inhabituel de coups sous l'aire de ceux-ci. Ces résultats montrent que le seuil simple et le CFD ont des difficultés de déclenchement à basse énergie. Cependant, il est entendu que le CFD a l'avantage de pouvoir dater précisément une impulsion, un critère qui ne fait pas partie de la présente évaluation. La méthode 6 semble être la méthode de déclenchement la plus polyvalente de la littérature, que ce soit pour le spectre en amplitude ou en aire. Cependant, comme avec les méthodes 4 et 5, elle dépend d'une BLR pour son seuil de déclenchement, ce qui, on l'a vu, et cela est confirmé, est préjudiciable pour la résolution. Ainsi, on peut conclure qu'une stratégie de fenêtrage statique, tout en restant acceptable, donne des résultats plus médiocres qu'un fenêtrage dynamique, comme le montrent les résolutions obtenues par notre méthode pour le spectre en aire.

| Energie | Méthode | Résolution (%) | Coups | Contribution (%) | Erreur (%) |
|---|---|---|---|---|---|
| 17.05 $keV$ | 1 | $NA$ | $NA$ | $NA$ | $NA$ |
| | 2 | 650 | 3962±87 | 12.13±0.29 | 25.53 |
| | 2 ($b$) | 84.85 | 7824±74 | 18.60±0.23 | 19.06 |
| | 3 | 200 | 7514±99 | 21.14±0.21 | 16.52 |
| | 4 | 404.55 | 15268±89 | 76.05±0.11 | 38.39 |
| | 5 | 395.83 | 15205±84 | 81.93±0.11 | 44.27 |
| | 6 | 171.42 | 4283±67 | 20.26±0.22 | 17.40 |
| | 7 | 52.63 | 5952±58 | 26.50±0.19 | 11.16 |
| 59.54 $keV$ | 1 | 23.19 | 12963±50 | 13.21±0.27 | 22.71 |
| | 2 | 22.90 | 11968±67 | 36.65±0.17 | 0.73 |
| | 2 ($b$) | 19.70 | 11836±63 | 28.14±0.19 | 7.78 |
| | 3 | 34.41 | 11110±89 | 31.26±0.18 | 4.66 |
| | 4 | 37.50 | 9713±64 | 48.38±0.14 | 12.46 |
| | 5 | 34.48 | 9249±75 | 49.84±0.14 | 13.92 |
| | 6 | 33.33 | 8713±70 | 41.21±0.16 | 5.29 |
| | 7 | 17.54 | 8060±57 | 35.88±0.17 | 0.04 |

**Tableau 3-1 : Résolution des pics de l'Am-241 pour le spectre en aire (en %).**

| Energie | Méthode | Résolution (%) | Coups | Contribution (%) | Erreur (%) |
|---|---|---|---|---|---|
| 17.05 $keV$ | 1 | $NA$ | $NA$ | $NA$ | $NA$ |
| | 2 | 340 | 5985±84 | 18.05±0.24 | 19.61 |
| | 2 ($b$) | 100 | 8280±63 | 18.55±0.23 | 19.11 |
| | 3 | 70.65 | 6605±89 | 18.59±0.23 | 19.07 |
| | 4 | 16.89 | 531±74 | 2.57±0.64 | 35.09 |
| | 5 | 17.99 | 562±63 | 2.84±0.60 | 34.82 |
| | 6 | 40.00 | 2763±54 | 13.07±0.28 | 24.59 |
| | 7 | 33.33 | 4061±57 | 18.08±0.24 | 19.58 |
| 59.54 $keV$ | 1 | 19.80 | 10281±32 | 7.65±0.36 | 28.27 |
| | 2 | 22.44 | 10513±55 | 31.71±018 | 4.21 |
| | 2 ($b$) | 18.84 | 10457±52 | 23.42±0.21 | 12.50 |
| | 3 | 18.57 | 10969±71 | 30.87±0.18 | 5.05 |
| | 4 | 12.67 | 6364±55 | 30.83±0.18 | 5.09 |
| | 5 | 14.02 | 6599±52 | 33.29±0.17 | 2.63 |
| | 6 | 14.02 | 6141±45 | 29.05±0.19 | 6.87 |
| | 7 | 10.84 | 5723±39 | 25.48±0.20 | 10.44 |

**Tableau 3-2 : Résolution des pics de l'Am-241 pour le spectre en amplitude (en %).**

A titre d'exemple, nous choisissons d'illustrer le résultat du processus d'extraction à l'aide de la Figure 3-20. Cette figure permet de visualiser à quelles problématiques doit faire face le déclencheur (empilement, ligne de base) et quelle est sa réponse. Toutes les impulsions extraites sur le signal choisi pour les tests ont été vérifiées manuellement afin de valider le marquage des impulsions empilées ou non. Le taux de réussite pour le signal testé est de 100 %, ce qui veut dire que la totalité des impulsions extraites sont reconnues comme individuelles ou empilées.

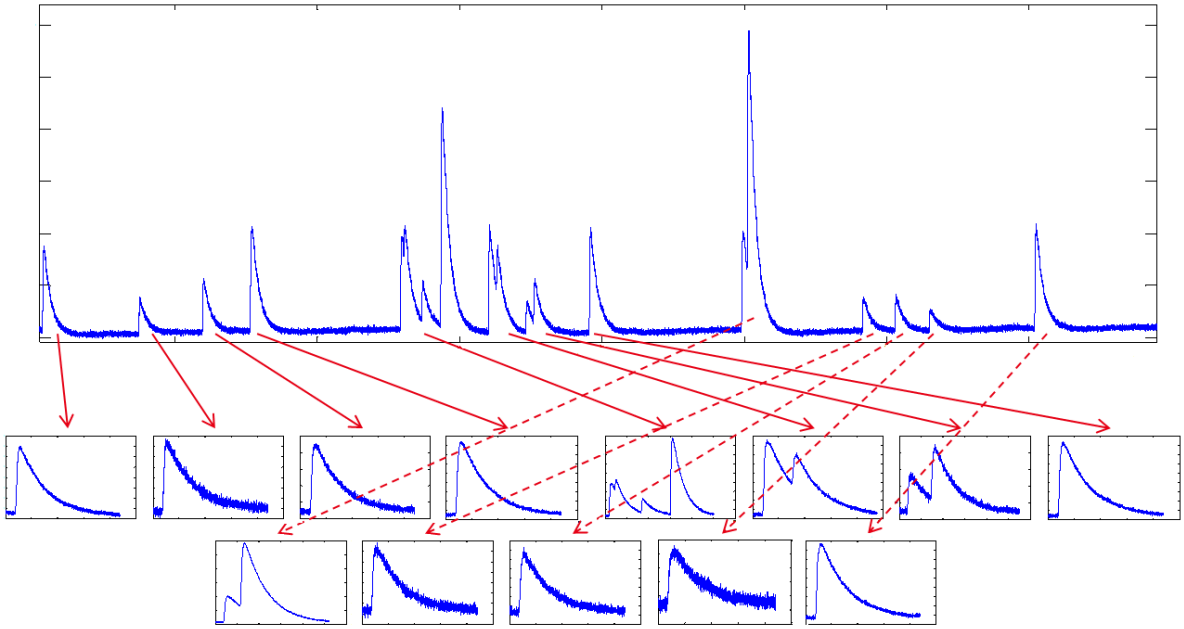


**Figure 3-20 : Illustration du comportement de notre méthode sur une portion du signal traité.**

## 3.4 Conclusion

La première étape de notre modèle d'exécution est d'extraire des impulsions du reste du signal. Ceci est habituellement réalisé par l'utilisation d'un seuil fixe et d'une fenêtre de temps prédéfinie. Comme la longueur des impulsions et l'intensité du bruit varient en fonction des conditions expérimentales et des détecteurs utilisés, une telle étape d'extraction ne peut être trivialement réalisée par les approches traditionnelles, qui sont dédiées, comme le montre les résultats du présent chapitre. Ce chapitre présente la faisabilité d'un dispositif numérique et générique d'extraction précis des impulsions basé sur un seuil dynamique et une méthode de fenêtrage s'adaptant à la taille des impulsions. La comparaison entre la méthode proposée et les méthodes de la littérature a montré la supériorité de notre approche pour une application traditionnelle de spectrométrie, en particulier dans le cas de mesures de courte durée. Notre méthodologie a obtenu de meilleurs résultats en termes de résolution pour deux types de mesures, amplitude et aire. L'étude du spectre obtenu à partir de l'aire des impulsions montre que notre méthode s'adapte également aux mesures d'analyse de forme d'impulsion. Avec ces résultats, la première étape de notre modèle d'exécution est validée. L'étape suivante consiste donc en la conception de l'architecture du modèle d'exécution proposé dans cette thèse.

Des travaux en cours se concentrent sur une implémentation VHDL de cette méthode afin qu'elle soit mise en œuvre sur des chaînes d'instrumentation. Ils ne sont pas abordés dans ce mémoire de thèse.

# Chapitre 4 : Proposition d'architecture basée sur le modèle d'exécution

Ce chapitre présente un modèle d'architecture électronique basé sur le modèle d'exécution décrit dans le Chapitre 2. Pour rappel, il admet qu'une fois les impulsions extraites, il est possible de les distribuer individuellement sur des ensembles processeurs et mémoires regroupés en étages de macro-pipelines. Le déclencheur permettant l'extraction des impulsions du reste du signal étant développé dans le Chapitre 3, l'étape suivante consiste donc à modéliser l'architecture de traitement des impulsions associées. Ce modèle est conçu aussi bien pour être implémenté sur SoC qu'avec des composants sur étagère. Conformément au modèle d'exécution, l'architecture visée est multivoie, dirigée par les impulsions et capable d'atteindre le zéro-temps mort.

Dans ce chapitre, une première proposition de cette architecture est tout d'abord décrite. Puis, un simulateur développé en SystemC modélisant ses caractéristiques est proposé. Il doit permettre de valider les objectifs de passage à l'échelle et d'atteinte du zéro-temps mort qui n'ont pas pu être démontrés analytiquement dans le Chapitre 2. A l'aide de jeux de données réels, il est possible de simuler le comportement de l'architecture vis-à-vis du temps mort et du passage à l'échelle tout en étant modulable dans le choix des composants architecturaux testés. Enfin, une première proposition d'implémentation à l'aide de cartes de développement programmables du marché est détaillée. Le modèle d'architecture et le simulateur ont donné lieu à deux articles, l'un dans une conférence internationale et l'autre dans une conférence francophone.

## 4.1 Présentation du modèle de l'architecture

Cette section présente l'architecture et une proposition d'implémentation. Cette architecture se compose des éléments définis par le modèle d'exécution. Un sous-ensemble du modèle d'exécution est illustré en Figure 4-1.

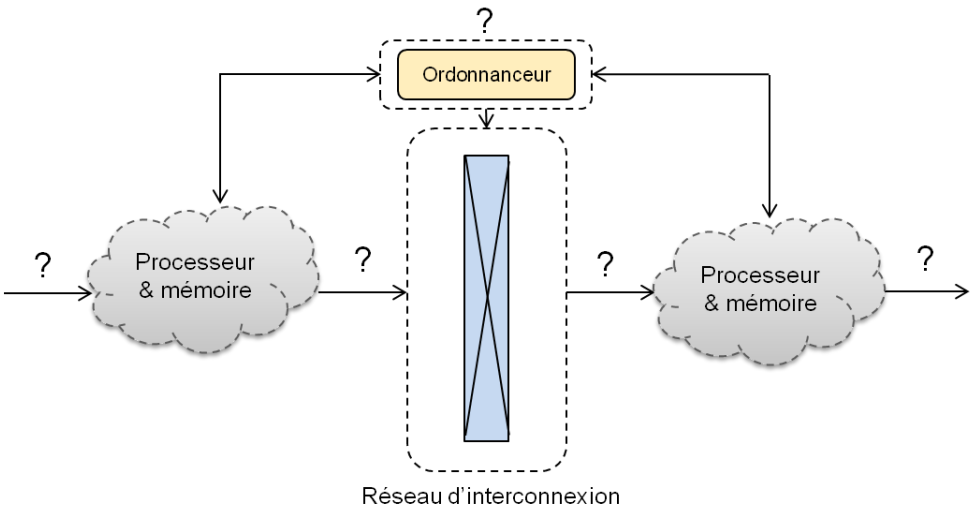



**Figure 4-1 : Schéma délimitant les quatre zones de conception de l'architecture : processeur, mémoire, ordonnancement et réseau d'interconnexion.**

Comme l'illustre la Figure 4-1, le modèle d'exécution se compose de processeurs génériques, de mémoires qui ont pour rôle d'acheminer les données aux processeurs, d'ordonnanceurs et de réseaux d'interconnexion. Ces quatre éléments peuvent être définis individuellement s'ils respectent les contraintes du modèle. La suite de la section décrit la conception de ceux-ci.

### 4.1.1 Processeur générique

Comme l'architecture doit être programmable, il est prévu d'utiliser un processeur générique. Pour respecter le modèle d'exécution, il doit répondre aux contraintes suivantes :

- il doit pouvoir au moins supporter l'implémentation « entière » d'un algorithme parmi les blocs applicatifs prédéfinis (correction, mise en forme, récupération de l'information d'intérêt) ;
- la gestion des flux de données, des mémoires tampons ou de synchronisation, du réseau et de l'ordonnancement doit être transparente pour l'utilisateur ;
- il doit être modulaire et facilement intégrable. Il doit être possible de modifier le type de processeur sans modifier l'architecture dans sa globalité afin de s'adapter à la grande variabilité de dimensionnements possibles imposée par les détecteurs ;
- il doit pouvoir signaler à l'ordonnanceur son état, disponible pour recevoir des données ou non.

Comme le programmeur ne doit pas gérer lui-même les aspects contrôle, il convient de régler la question du démarrage des traitements. Le modèle étant dirigé par les impulsions, les traitements démarrent immédiatement lors de l'arrivée d'une impulsion. Pour cela, une première solution est d'utiliser les interruptions que tous les processeurs implémentent, associées à un modèle de programmation basé sur des fonctions utilisateur. Ces fonctions offrent à l'utilisateur la possibilité par exemple de traiter soit une impulsion (ou un paquet de données) entière, soit un signal en flux borné échantillon par échantillon. De telles fonctions peuvent être représentées en C, comme en Figure 4-2.

```
int* process_packet(int * packetIn)
{
        int * result;
        /*
                Code utilisateur de traitement de pulse
        */
        return result;
}

int* process_sample(int * sampleIn)
{
        int * result ;
        /*
                Code utilisateur de traitement d'un échantillon
        */
        return result;
}
```

**Figure 4-2 : Prototypes de fonctions utilisateurs du modèle de programmation proposé.**

Grâce à celles-ci, l'utilisateur écrit le traitement qui sera réveillé à chaque appel de la fonction par le gestionnaire d'interruption. L'utilisation des interruptions soulève le problème de leur gestion pour l'acheminement des données au processeur et l'émission des résultats à l'étage de pipeline suivant.

### 4.1.2 Alimentation du processeur en données

Dans le modèle d'exécution, la mémoire tampon a pour rôle d'alimenter en données le processeur. Son implémentation doit respecter les contraintes suivantes :

- le contrôle et la mémorisation des données entrantes et sortantes du processeur doit être adaptatif ou paramétrable car les contraintes sur la nature du flux/type de données imposées par l'application présente dans le processeur ne sont pas connues à l'avance ;
- une impulsion entière doit pouvoir être mémorisée. En effet, certains traitements attendent l'intégralité d'une impulsion avant de la traiter (transformée de Fourier) ;
- la mémoire doit pouvoir gérer des communications asynchrones car les processeurs de chaque étage de pipelines peuvent être hétérogènes. Il peut donc y avoir des différences d'horloge entre processeurs se communiquant des données.

Comme le modèle proposé est indépendant du choix du processeur, la gestion du flux de données entrant doit être indépendante d'un mode de fonctionnement où le processeur chargerait dans sa mémoire de travail (RAM) l'ensemble des données entrantes dans la mémoire avant de les traiter. Autrement-dit, les données doivent être prêtes lorsque l'interruption est levée et que le processeur démarre son traitement. La nature des flux à traiter implique différents cas possibles que l'ensemble processeur et mémoire doivent être en mesure de gérer :

- entrée flux borné, traitement puis sortie flux borné : il s'agit d'un cas où les échantillons d'une impulsion ou d'un paquet entrant dans la mémoire sont traités à la volée et la distribution des résultats est réalisable au rythme d'arrivée des échantillons entrants. Le résultat du traitement se présentent également sous la forme d'une impulsion ou d'un paquet dont les caractéristiques ont pu changer (nombre d'impulsions, valeur des données). Un exemple peut être un filtre FIR ;
- entrée flux bornée, traitement puis sortie paquet/impulsion(s) : il s'agit d'un cas où les échantillons d'une impulsion ou d'un paquet entrant dans la mémoire sont traités à la volée mais la distribution des résultats attend la fin de l'absorption des données entrantes et la fin du traitement. Le résultat du traitement se présente également sous la forme d'une impulsion ou d'un paquet dont les caractéristiques ont pu changer. Un exemple peut être un correcteur d'empilement ou simplement une recherche de caractéristique(s) sur une impulsion ;
- entrée paquet/impulsion(s), traitement puis sortie flux borné : dans ce cas, la totalité des échantillons d'une impulsion ou d'un paquet entrant doit être mémorisée avant le traitement mais la distribution des résultats peut se faire à la volée. Le résultat du traitement se présente également sous la forme d'une impulsion ou d'un paquet dont les caractéristiques ont pu changer. Un exemple peut être un filtre trapézoïdal qui doit attendre d'avoir les caractéristiques complètes de l'impulsion à filtrer avant de commencer son traitement (durée, amplitude) ;

- entrée paquet/impulsion(s), traitement puis sortie paquet/impulsion(s) : il s'agit d'un cas où la totalité des échantillons d'une impulsion ou d'un paquet entrant doit être mémorisée avant le traitement, la distribution des résultats implique l'attente de la fin des données entrantes et la fin du traitement. Le résultat du traitement se présente également sous la forme d'une impulsion ou d'un paquet dont les caractéristiques ont pu changer. Un exemple peut être une transformée de Fourier.

Les débits des données émises et lues dépendent donc directement des traitements qui peuvent être en flux borné ou en paquet. L'utilisation de mémoires spécifiques pour l'acheminement des données permet de contourner cette problématique. Une mémoire d'entrée sert à la réception des données d'entrée et à leur distribution au processeur. Une mémoire de sortie sert à l'émission des résultats. La première permet la mémorisation complète d'un paquet de données avant traitement. La seconde permet l'émission en flux borné des résultats sans *a priori* sur la vitesse d'absorption de ces résultats à l'étage suivant. Les interruptions permettant le déclenchement des traitements se basent donc sur l'état de ces mémoires de communications.

#### 4.1.2.1 Modèle initial d'acheminement des données

Deux FIFOs à deux horloges (lecture/écriture) sont utilisées afin de permettre l'acheminement des données entre deux domaines d'horloge. La Figure 4-3 illustre leur adjonction au processeur générique décrit précédemment.

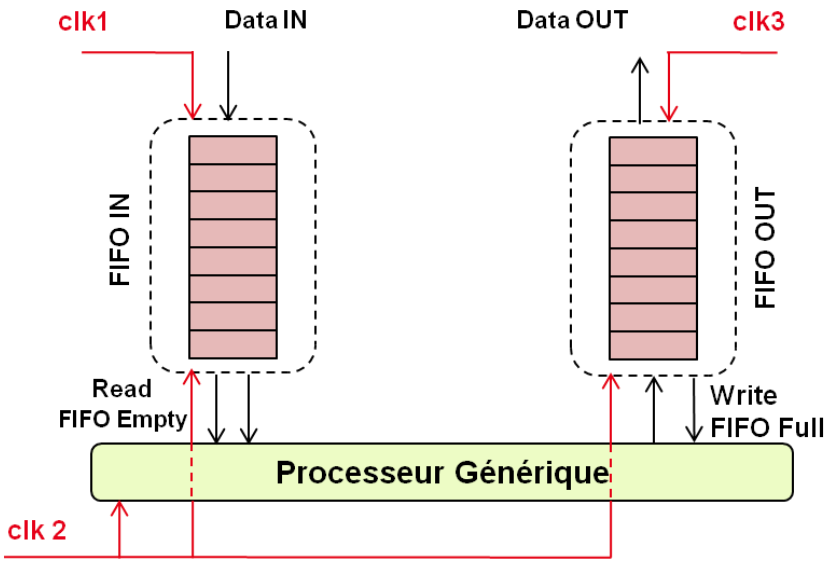



**Figure 4-3 : Modèle d'architecture du processeur générique, des mémoires d'entrée et de sorties utilisées pour l'acheminement des données.**

La FIFO d'entrée, appelée *FIFO IN*, utilise une horloge synchrone à celle du processeur pour la lecture des données ; cela permet par exemple l'utilisation d'un mode DMA d'Accès en Mémoire Direct. La FIFO de sortie, appelée *FIFO OUT* utilise une horloge synchrone à celle du processeur pour l'écriture des données. La taille de *FIFO IN* du premier étage de pipeline doit être dimensionnée au pire cas, c'est-à-dire qu'elle doit pouvoir contenir la longueur maximum d'une impulsion ou d'un empilement toléré par l'extracteur d'impulsion. Cela permet d'éviter les débordements de mémoire.

Toutes les autres FIFO doivent être paramétrées à cette taille ou à une taille adaptée au type de résultat fourni par les traitements. L'interruption qui permet au processeur de déclencher son traitement est levée dès que FIFO IN n'est plus vide. Cette information est obtenue par le signal *FIFO EMPTY.*

Il existe de nombreuses implémentations de FIFOs qui adaptent deux domaines d'horloge (Zhang et al. 2011; Wang et al. 2004). D'une manière générale, l'émetteur écrit les données dans une mémoire RAM double ports à sa fréquence de fonctionnement et le récepteur lit les données à sa propre vitesse de fonctionnement. La synchronisation entre les domaines de fonctionnement n'est nécessaire que pour gérer les états pleins et vides de la FIFO. L'utilisation de ce type de mémoire de communication permet de travailler avec n'importe quel type de flux et n'admet aucun non-déterminisme dans les délais de communication, permettant de garantir la gestion du temps mort. La Figure 4-4 illustre schématiquement l'architecture d'une FIFO à deux horloges.

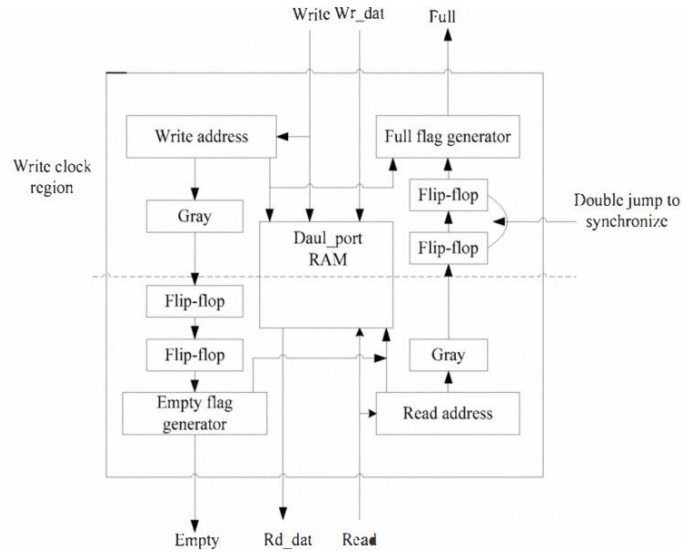



**Figure 4-4 : Schéma simplifié d'une FIFO à deux horloges (Wang et al. 2004).**

**Problématiques d'une implémentation FIFO pour la mémoire :**

L'utilisation d'un modèle FIFO pour l'implémentation des mémoires d'acheminement des impulsions possède quelques inconvénients. En effet, la taille des impulsions étant variable, il devient difficile de garantir l'intégrité des données si plus d'une impulsion/paquet sont présents dans ces mémoires. Deux exemples sont décrits.

Mémoriser plus d'une impulsion dans la mémoire d'entrée sans avoir connaissance du débit d'absorption par le traitement de ces données est risqué. La transmission des données peut s'arrêter en cas de FIFO pleine. Des données sont alors perdues et le traitement des données se fera sur des fragments d'impulsions. Dans ce cas de figure, si la totalité des processeurs du système fonctionnent dans un mode de flux borné vers flux borné, alors le système peut se retrouver paralysé alors que d'autres processeurs sont peut-être disponibles.

Dans le cas de la FIFO de sortie, les problématiques sont similaires. Si par exemple la mémoire est dimensionnée pour deux fois la taille maximum d'une impulsion admise par l'extracteur d'impulsions, alors plus de deux impulsions peuvent se trouver en mémoire au même instant puisque leurs tailles varient. La Figure 4-5 illustre un cas typique pouvant être rencontré dans ce cadre. Deux

impulsions déjà traitées sont en attente d'émission ; l'une d'elle est invalidée à la fin du traitement mais comme celui-ci est configuré pour émettre ses résultats en flux borné, l'impulsion attend que le contrôleur lise cette information au travers d'une métadonnée. La FIFO n'étant pas pleine, le traitement commence l'émission de nouveaux résultats mais se heurte à un *FIFO Full*. Il n'est alors plus possible de traiter les données en flux borné puisqu'il est impossible de supprimer la deuxième impulsion avant l'émission de la première.

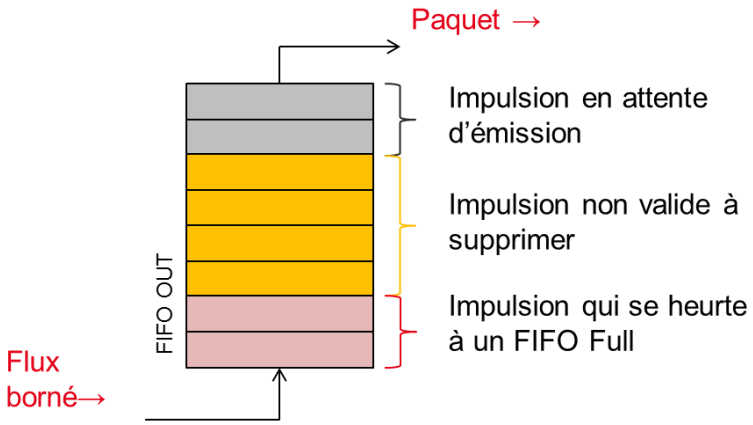



**Figure 4-5 : Exemple de problème rencontré pour la gestion de plusieurs impulsions dans une FIFO.**

Pour répondre à ce problème, la règle suivante est définie :

**Règle 1** : un processeur peut traiter une nouvelle impulsion si et seulement si sa FIFO d'entrée est vide.

Cette approche possède l'inconvénient de sur-dimensionner le nombre de processeurs nécessaires au zéro-temps mort mais garantit l'intégrité des données en sortie d'extracteur d'impulsion.

**Gestion de la mémoire pleine :**

Le modèle n'admet pas le passage à l'état *FIFO Full* des FIFOs puisqu'elles sont dimensionnées/bornées par les limites d'acceptabilité du déclencheur ou par le type de résultat fourni par un traitement. En effet, l'état *FIFO Full* est atteint lors de la limite max imposée par le traitement plus un échantillon. Le mode de fonctionnement défini au Chapitre 2 qui précise qu'une seule impulsion est traitée à la fois quel que soit le type de flot de donnée ne permet pas de remplir totalement les FIFOs. Cependant, compte-tenu de la nature programmable de l'étage de traitement, la possibilité d'une FIFO pleine doit être gérée et renseignée à l'utilisateur. Sachant qu'une *FIFO IN* ne peut être de taille supérieure à la taille d'une *FIFO OUT* de l'étage précédent, alors deux règles sont définies :

**Règle 2 :** lorsque l'une des *FIFO IN* est pleine, elle est automatiquement vidée et l'ensemble des échantillons en cours de distribution dans celle-ci est supprimé jusqu'à la fin de la transmission.

**Règle 3 :** Lorsque l'une des *FIFO OUT* est pleine, elle est automatiquement vidée et l'ensemble des échantillons en cours de distribution par le traitement dans celle-ci est supprimé jusqu'à la fin du traitement.

#### 4.1.2.2 Contrôle des données

Pour orchestrer l'arrivée et le départ des données en fonction des ordonnanceurs, il convient de disposer d'un dispositif de contrôle. Un premier dispositif, appelé contrôle des données d'entrée, a pour fonction de réaliser le transfert des données avec un étage précédent afin de remplir *FIFO IN*. Le second, appelé contrôle des données de sortie, réalise le transfert des données avec un étage suivant afin de vider *FIFO OUT*. La Figure 4-3 illustre l'ajout de ces dispositifs au modèle d'architecture processeur et mémoires. La disponibilité du processeur pour recevoir une donnée ou un ensemble de donnée est régit par l'état de *FIFO IN* et donc par le traitement. L'état est renseigné à l'ordonnanceur connectant ce contrôleur à l'étage précèdent à l'aide du signal *Busy*, tandis que la présence d'une donnée disponible à l'envoi est renseignée à l'ordonnanceur connectant ce contrôleur à l'étage suivant à l'aide du signal *Data Ready*.

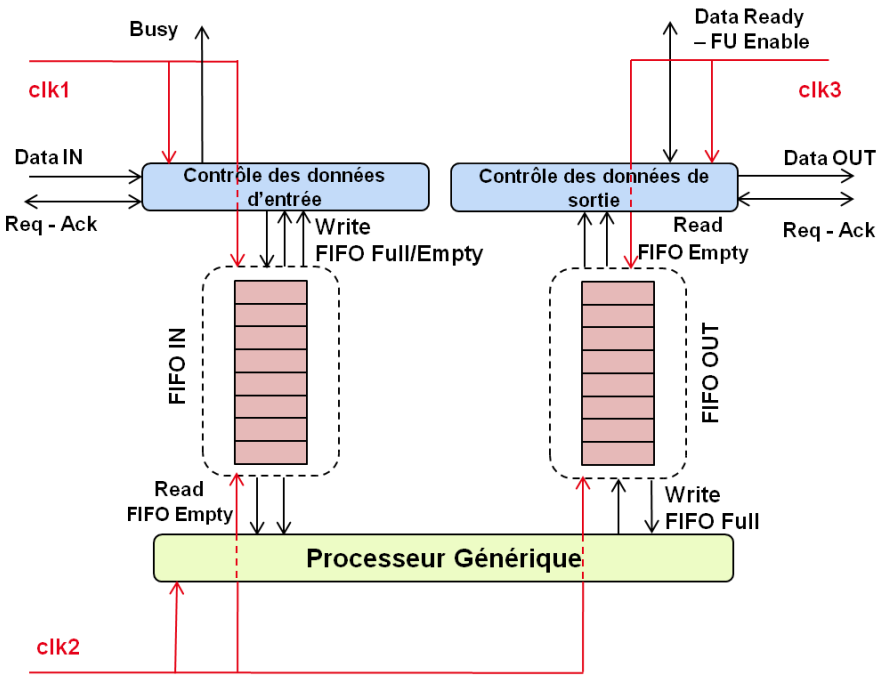



**Figure 4-6 : Modèle d'architecture incluant le processeur générique, les mémoires d'entrée et de sortie et les contrôles des données d'entrée et de sortie.**

Les contrôles des données d'entrée d'un étage N sont cadencés par une horloge commune aux contrôles des données de sortie de l'étage N-1. Les contrôles de données de sortie d'un étage N sont cadencés par une horloge commune aux contrôles des données de sortie de l'étage N+1. Cette approche, réalisable sur SoC permet d'éviter de propager une horloge commune à chaque étage augmentant les capacités de passage à l'échelle du système. Cependant, le modèle est également conçu pour admettre des communications entre étages de processeurs asynchrones. Pour cela, une couche de communication supplémentaire peut être ajoutée afin de permettre le dialogue inter processeurs et ordonnanceurs (*handshaking* par exemple). Cette surcouche ne change pas le mode de fonctionnement du contrôle des données.

**Problématiques restantes :**

Contrairement aux contrôleurs, le processeur possède sa propre mémoire de travail. Cela signifie que dans tous les cas, l'émission du/des résultat(s) peut se faire à la volée. En effet, à l'aide de sa mémoire, le processeur peut décider d'attendre ou non la fin des traitements avant le remplissage de résultat dans FIFO OUT. Cependant, un algorithme pouvant prendre la décision d'invalider le(s) résultat(s) (détecteur d'empilements) doit le faire avant l'émission des résultats dans *FIFO OUT*. Sinon, le contrôleur risque de lire et d'envoyer des données invalides. Cette approche limite, pour ce type d'algorithme, la possibilité de travailler en flux borné et d'émettre les résultats en *FIFO OUT* avant de prendre la décision de les invalider, augmentant ainsi le temps mort.

Une seule impulsion peut être traitée à la fois par un processeur afin d'éviter la gestion d'impulsion tronquée. Si une impulsion est présente, le processeur est considéré comme occupé et ne peut pas être choisi par l'ordonnanceur pour recevoir une impulsion. Cependant, ce mode de fonctionnement est limité par la volonté de pouvoir aussi bien traiter en flux borné les données (acquérir un échantillon, le traiter, et l'émettre dans la foulée), ou dans un mode de paquets (attendre la fin du traitement avant l'envoi des données). En effet, le processeur ne communique pas son état au contrôleur et peut lire les données plus vite qu'elles ne sont remplies dans la mémoire d'entrée. Ce cas génère des états transitoires où la mémoire d'entrée est vide entre chaque lecture, rendant le processeur disponible à la réception d'une impulsion avant la fin des traitements. De plus, ces états transitoires ne permettent pas au processeur de savoir quand s'arrête le segment de données à traiter. C'est pourquoi, le processeur et les contrôleurs admettent l'ajout de métadonnées. Ces métadonnées permettant, sans une communication directe entre le processeur et le contrôle des données, de stabiliser les états de contrôle et de disponibilité d'un processeur. Elles permettent également de résoudre la problématique d'invalider les données après remplissage de *FIFO OUT*, ce qui permet à un traitement de ce type de travailler en flux borné.

#### 4.1.2.3 Métadonnées

Le modèle proposé permet l'ajout de différents types de métadonnées aux données du signal en fonction des applications. Elles sont ajoutées par l'utilisateur. L'étude des différents modes de fonctionnement et des applications nous permet de distinguer deux types de métadonnées : les métadonnées servant au contrôle du flux de données, et les métadonnées servant à ajouter de l'information au résultat. Dans un premier temps, nous présentons celles concernant le contrôle du flux de données et la gestion des données invalides :

- ***End Packet*** signale la fin d'un paquet de données. Si *End Packet* = 1, alors les contrôleurs débutent le transfert des données soit dans *FIFO IN* soit vers une processeur de l'étage suivant. Si *End packet* = 0, alors les contrôleurs transmettent les données jusqu'à lire *End packet* = 1. Le deuxième cas est utilisé pour éviter qu'un processeur passe à l'état disponible si *FIFO IN* se vide plus vite qu'elle ne se remplit. Ces *états* transitoires entre chaque lecture rendent le processeur disponible pour l'ordonnanceur ;
- ***Flush*** signal au processeur de l'étage suivant que le segment de donnée doit être supprimé. Il s'agit du cas particulier d'un algorithme pouvant prendre la décision d'invalider le(s) résultat(s) après l'émission des résultats dans *FIFO OUT*. Si pour une nouvelle série de données lues par

le processeur la première indique *Flush* = 1 quand *End Packet* = 0, alors l'ensemble des données jusqu'à *End Packet = 1* doit être lu avant traitement. *Si flush* = 1 quand *end packet* = 1, alors le processeur ne traite pas et supprime l'ensemble des données lues. Ce mode de fonctionnement est nécessaire dans un modèle où aucune communication directe n'est possible entre traitement et contrôle. Le traitement ne peut pas décider seul de « reset » *FIFO OUT* puisqu'il ne connaît pas l'état du contrôleur et donc la possibilité d'un transfert déjà débuté.

Dans un second temps, nous présentons les métadonnées permettant l'ajout d'informations aux échantillons individuels ou aux paquets :

- ***Pileup*** signale que le paquet/segment de données contient un empilement. C'est un cas particulier réservé au déclencheur où à un étage spécifique de détection d'empilement ;
- ***Timestamp*** signale que la donnée est la référence temporelle de l'échantillon. C'est également un cas particulier réservé au déclencheur si sa fonction permet la datation de l'impulsion extraite ;
- l'utilisateur peut également ajouter ses propres métadonnées.

En résumé, l'ajout de métadonnées, corrélé à l'état *Busy* d'un processeur, permet d'utiliser un seul et unique mode de contrôle de données pour gérer l'ensemble des types de flots de données ; cela a été décrit en 4.1.2.1 et rend le code utilisateur indépendant du contrôle des données, permettant plus de flexibilité.

#### 4.1.2.4 Exemple d'utilisation des métadonnées par le processeur

L'ensemble des éléments décris précédemment permet de définir le comportement du processeur face aux interruptions et aux métadonnées pour le démarrage et la fin des traitements. La seule contrainte imposée à l'utilisateur programmant le processeur est l'ajout de métadonnées *End Packet* et *Flush.* Si cette contrainte est respectée, alors la description de l'exécution du traitement du processeur est facilitée et peut être définie grâce à une machine à état finie présentée en Figure 4-7.

L'exécution des traitements du processeur commence lorsque *FIFO IN* n'est plus vide. Le processeur récupère alors une première donnée afin de vérifier si le paquet à venir peut être supprimé. Si pour la première donnée lue, la métadonnée associée *Flush* = 1, alors le processeur lit entièrement la FIFO jusqu'à *End Packet*. Dès lors, si *Flush* = 1, il supprime totalement les données avant tout traitement. Dans le cas contraire si *Flush = 0* dans la première donnée lue, le processeur peut lire, traiter et émettre le(s) résultat(s) dans *FIFO OUT* à la volée jusqu'à lire *End Packet*.

En conclusion, le processeur générique défini permet de répondre aux contraintes de flexibilité imposées à l'architecture. Seules les lectures de *FIFO IN* et l'écriture dans *FIFO OUT* relient le processeur au reste de l'architecture. Grâce aux FIFOs et aux métadonnées, la durée du traitement du processeur peut varier sans corrompre les données à cause du temps mort. Cette approche permet un changement aisé de processeur et facilite le dimensionnement de l'architecture en fonction des détecteurs ou des traitements.

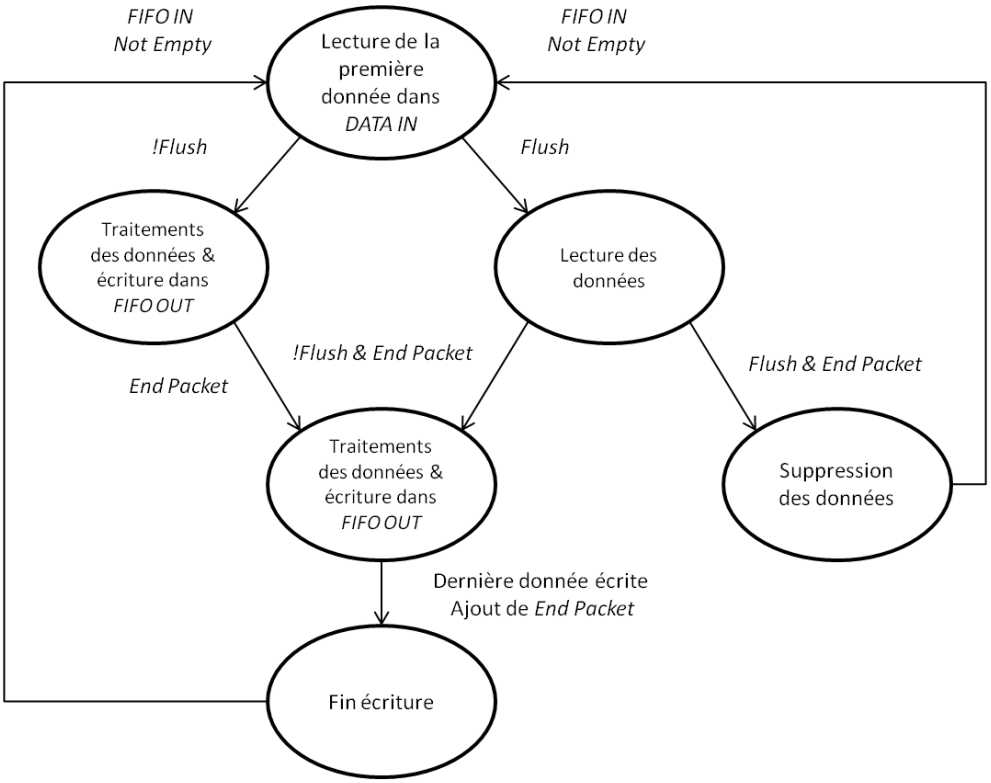



**Figure 4-7 : Machine à état du processeur générique.**

#### 4.1.2.5 Définition du comportement du contrôle des données

L'utilisation d'un contrôle des données pour l'entrée l'acheminement des données vers un processeur et pour la sortie des données depuis un processeur permet, grâce à l'utilisation de FIFOs à deux horloges et à l'ajout de métadonnées, de simplifier la gestion des différents types de flot de données. Le fonctionnement des deux contrôleurs de données peut simplement être exprimé par deux machines à état finies représentées en Figure 4-8 et Figure 4-9.

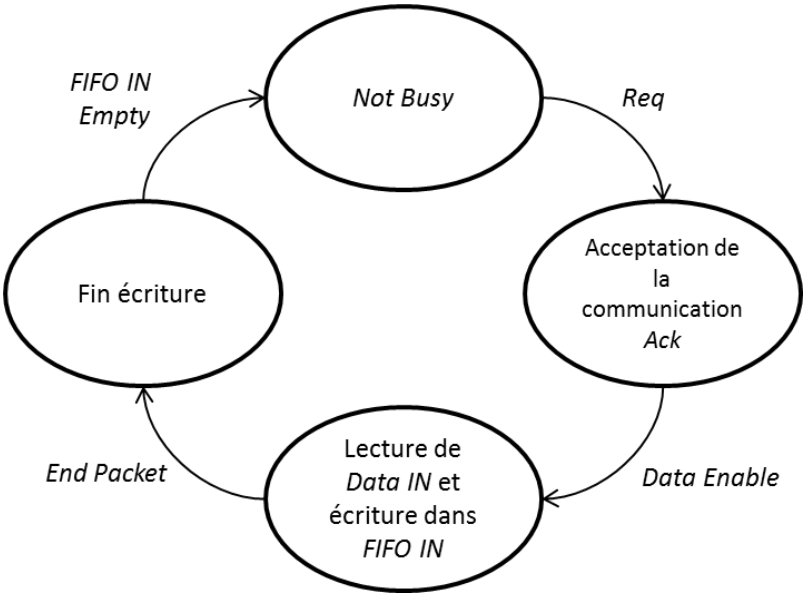



**Figure 4-8 : Machine à état du contrôle des données d'entrée.**

A l'état disponible (*FIFO IN* vide), le contrôleur d'entrée attend une demande de communication de la part d'un processeur de l'étage précédent. Dès acquittement de cette requête, il débute la transmission jusqu'à *End Packet* (si communication synchrone). Chaque donnée reçue est écrite dans *FIFO IN*.

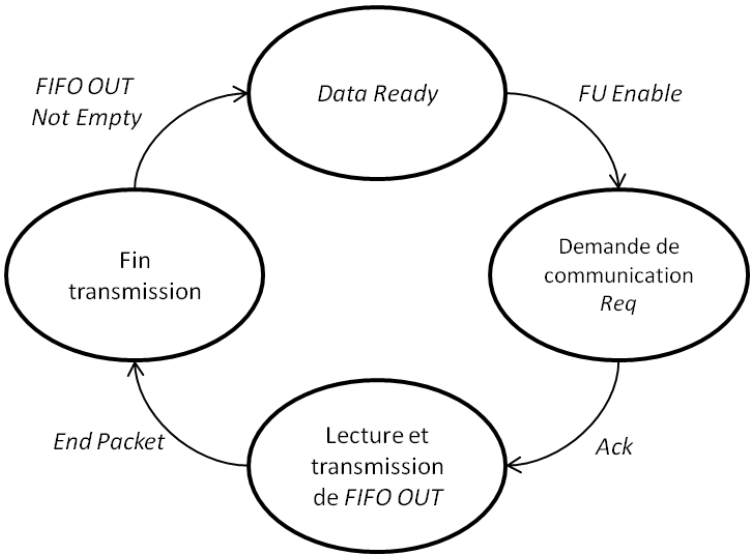



**Figure 4-9 : Machine à état du contrôle des données de sortie.**

Dès que *FIFO OUT* n'est plus vide, le contrôle des données de sortie signale à l'ordonnanceur la présence de données à émettre. L'ordonnanceur, au travers du signal *FU Enable*, signale au dispositif de contrôle qu'un processeur est prête à l'étage suivant. Le routage est effectué au même cycle que l'émission de *FU Enable*. Le contrôleur de sortie lit et transmet les données de *FIFO OUT* jusqu'à *End Packet*.

#### 4.1.2.6 L'unité fonctionnelle

Les composants que sont le processeur, les mémoires d'entrée et de sortie ainsi que le dispositif de contrôle des flux décrits dans cette section, composent une unité de calcul programmable indépendante appelée Unité Fonctionnelle (FU). La Figure 4-10 illustre le schéma de conception d'une FU permettant de répondre à l'ensemble des contraintes définies par le modèle d'exécution.

Pour synthétiser l'assemblage des différents composants présentés dans cette section, le fonctionnement général d'une FU est décrit. Son rôle se résume séquentiellement à la réception, au traitement et à l'émission de données. Chacune de ces étapes possède son propre débit de données. Les données reçues arrivent au débit d'émission imposé par le traitement d'un FU de l'étage précédent. Les segments de données sont de tailles variables. Un segment est limité par la nature impulsionnelle des données d'entrées. L'acceptation et le transfert des données reçues dans une mémoire dite « d'entrée » est régie par la partie contrôle des données d'entrée. Les données sont reçues au travers de l'entrée *Data IN* et directement dirigée dans la mémoire d'entrée *FIFO IN* qui est une FIFO à deux horloges. Dès la présence de données dans la *FIFO IN* (détectée par le processeur à l'aide du signal *FIFO Empty*), une interruption est levée permettant de lancer la partie traitement utilisateur. Les traitements exécutés par le processeur ne sont pas contraints par la nature du débit de données entrant dans la mémoire. L'exécution peut nécessiter une lecture préalable de l'ensemble des données présentes en mémoire les données à la volée. Une fois le résultat acquis, les données sont émises par

le processeur dans une mémoire dite de « sortie » *FIFO OUT* qui est également une FIFO à deux horloges. Dès la présence de données dans cette mémoire (détectée par le contrôle à l'aide du signal *FIFO Empty*), le contrôleur des données de sortie peut lire et envoyer les données sur l'étage suivant.

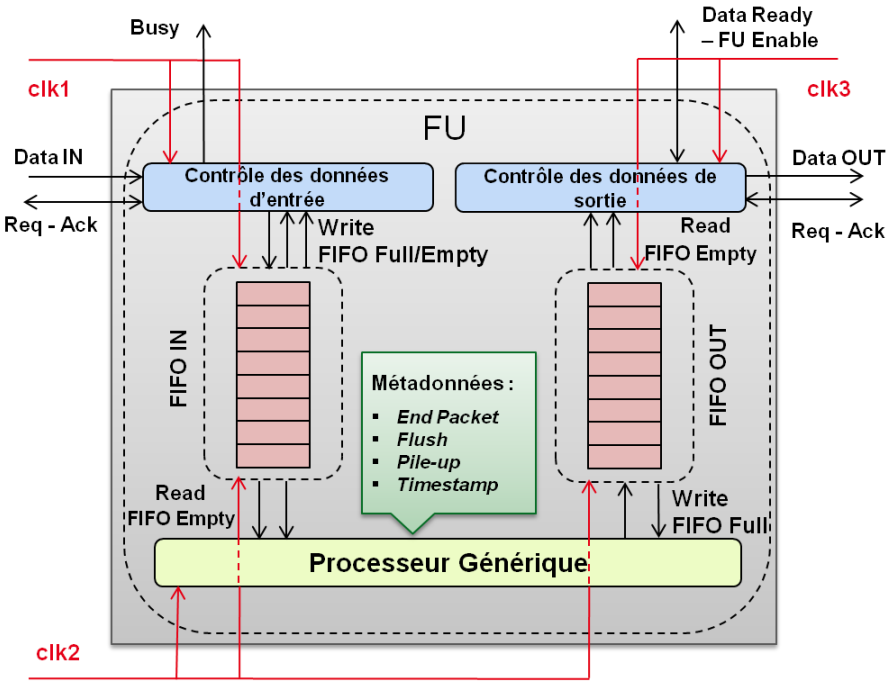



**Figure 4-10 : Modèle de FU programmable pour le traitement des impulsions.**

### 4.1.3 Réseau d'interconnexion

Un réseau d'interconnexion relie chaque étage de FU entre eux. Le choix du réseau est dans un premier temps contraint par le modèle mathématique de notre modèle d'exécution ; c'est-à-dire qu'en plus de trouver une solution pleinement connectée entre chaque étage de FU, idéalement, le nombre de données en circulation dans le réseau ne doit pas engendrer d'encombrement pour ne pas impacter sur les délais de communication.

Dans notre modèle, un commutateur parfait (*e.g crossbar*) est utilisé pour répondre à ces contraintes. Les commandes de routages sont reçues par l'ordonnanceur dans la limite d'une commande de routage par cycle. La complexité d'un *crossbar* augmente avec le carré du nombre d'éléments connectés. Il est donc évident que ce genre d'interconnexion limite le nombre d'éléments interconnectés si la surface est une contrainte forte. D'autres solutions existent pour limiter l'empreinte du *crossbar* augmentant ainsi le passage à l'échelle de l'architecture (Roca et al. 2012). Elles ne sont pour le moment pas analysées dans cette version de l'architecture mais seront prises en compte dans les travaux à venir.

Le *crossbar* admet, entre deux étages N et N+1, autant d'entrée que de FU à l'étage N et autant de sortie que de FU à l'étage N+1. La création de liens de routage consiste en la réception de duos d'adresses de FU. Pour la création de liens, le *crossbar* nécessite un arbitrage intégrant les règles de routage. Dans notre architecture, il s'agit du rôle de l'ordonnanceur. Le fonctionnement du *crossbar* consiste donc à attendre un duo d'adresses à la fois et de créer les liens entre les FUs de deux étages adjacents.

### 4.1.4 Ordonnancement/distribution

L'ordonnancement est une fonction qui permet aux différents éléments de contrôle des FUs de gérer la réception et l'émission de données. Les sous-parties précédentes montrent la nécessité d'utiliser un ordonnanceur entre deux étages pour permettre à la fois de connaître la disponibilité des FUs, la présence de données prêtes à l'envoi dans des FUs et le routage des du réseau d'interconnexion *crossbar*. Le rôle de l'ordonnanceur consiste à définir la cible et l'ordre de distribution des données/impulsions entre ces étages.

Comme un ordonnanceur traditionnel, son rôle est de distribuer des impulsions (assimilables à des tâches) entre plusieurs FUs. Cependant, la notion d'impulsion n'admet pas de préemption. Une impulsion ne pourra jamais être partagée entre plusieurs FUs. Cet ordonnancement est soumis à des incertitudes. En effet, la taille des impulsions/paquets à ordonnancer n'est pas connue *a priori*. L'ordonnanceur dirige et autorise la distribution de paquets de tailles variables sans connaître la durée d'exécution des traitements des FUs destinatrices. Cette information ne peut pas servir à adapter l'algorithme d'ordonnancement. Cependant, cela simplifie également le rôle de l'ordonnanceur. En effet s'il ne différencie pas les FUs destinatrices, il connaît la disponibilité d'une FU et la choisir immédiatement si une donnée est disponible, prête à l'envoi. Pour réaliser cela, nous favorisons une approche purement réactive. Cette approche, qualifiée aussi de « dynamique », est souvent utilisée dans des environnements logistiques fortement perturbés. Les décisions sont prises dynamiquement et en temps réel, en privilégiant la rapidité des décisions sur leur qualité. Une des méthodes les plus utilisées pour l'ordonnancement réactif est la gestion de files d'attentes par règle de priorité (Pinedo 2008).

Sur la base de cette idée, notre ordonnanceur utilise un algorithme de type FCFS (First come, first served) que nous appelons ici première-FU-disponible, première-FU-servie. L'ordonnanceur scrute cycliquement les disponibilités des FUs à un étage N+1 et les demandes de FUs de l'étage N. Il tient à jour une liste (qui peut donc être implémentée par une FIFO) de disponibilités. Dès qu'il reçoit une demande de la part d'une FU à l'étage N, il lui affecte la première FU de sa liste, qui est donc automatiquement disponible.

Un ordonnanceur entre les étages N et N+1 a autant d'entrées que de FUs à l'étage N et de sorties que de FUs à l'étage N+1. Cela signifie que des cas particuliers d'ordonnancement peuvent se produire si le nombre de FUs par étage est hétérogène. Par exemple, un ordonnanceur peut régir la distribution entre X entrées et 1 sortie. Dans ce cas, la scrutation des FUs de l'étage N se fait à tour de rôle (O(n)) pour éviter les accès concurrentiels. A la manière d'une règle de type PCO (Prefered Customer Order), la priorité est donnée aux FUs dans un ordre numérique prédéfini. Dès qu'une liaison N et N+1 est possible, l'ordonnanceur envoie l'adresse de la FU source et l'adresse de la FU cible au crossbar.

D'autres algorithmes d'ordonnancement doivent être étudiés, comme l'ordonnancement prioritaire, ou totalement aléatoire. Toutefois, compte-tenu de l'aspect non-déterministe des traitements et des dates d'arrivée des données à distribuer, aucun d'eux n'est un choix « évident ». Ils seront explorés dans les travaux à venir.

### 4.1.5 Une FU pour le contrôle de l'extracteur d'impulsion

L'étage d'extraction des impulsions nécessite de dialoguer avec l'ordonnanceur pour signaler l'arrivée d'une impulsion. Cependant, contrairement au processeur, celui-ci travaille en flux, ce qui signifie qu'il ne peut pas attendre la décision d'un ordonnancement sous peine de perdre des échantillons. C'est pourquoi, nous choisissons d'encapsuler le déclencheur dans une FU spécifique. Pour rappel, cette FU faisant la liaison entre flux et flux borné, elle doit être dédiée au détecteur. La FU permettant l'encapsulation de l'extracteur est présentée en Figure 4-11.

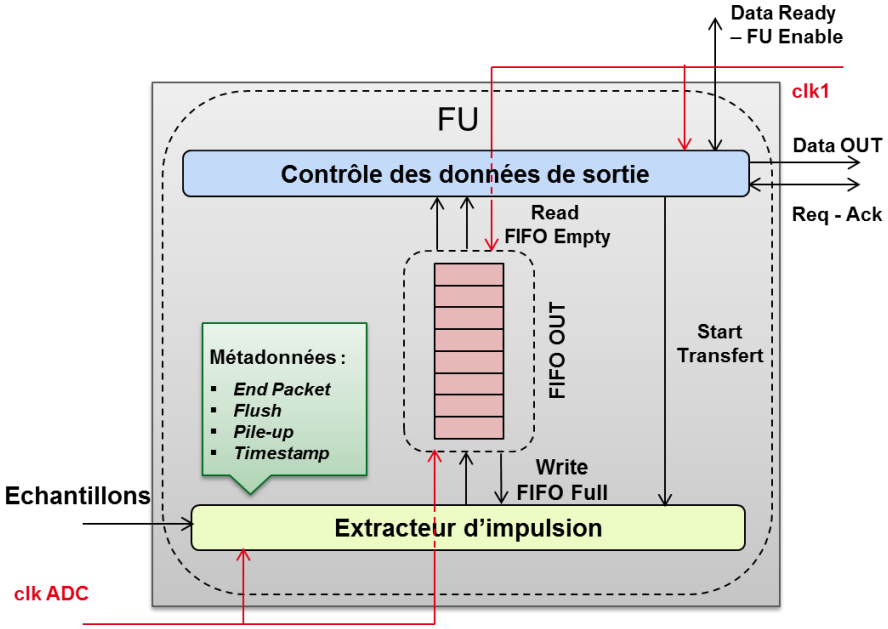



**Figure 4-11 : Modèle de FU encapsulant l'extracteur d'impulsion.**

Pour qu'une FU puisse fonctionner dans un mode de flux vers flux borné, une approche consiste à rejeter l'impulsion si aucune FU n'est choisie par l'ordonnanceur dès la détection d'une impulsion (déclenchement). De cette manière, l'extracteur termine son travail et attend l'arrivée d'une nouvelle impulsion avant de redemander la distribution. Le problème de cette approche est qu'elle ne permet pas de gérer plus d'un seul extracteur d'impulsion. Dans le modèle d'exécution multivoie, l'architecture intègre plusieurs extracteurs d'impulsions. Comme ils sont implémentés dans des FUs indépendantes, chacun d'eux peut détecter des impulsions arrivant au même instant. Cette probabilité existe pour des dates d'arrivée d'impulsions aléatoires. Si une règle de type PCO est conservée, alors un extracteur peut devoir attendre son tour avant la distribution. Cette problématique oblige à ajouter une mémorisation à l'étage d'extraction. C'est pourquoi nous décidons de conserver partiellement le format d'une FU pour l'encapsulation de l'extracteur afin qu'il bénéficie de *FIFO OUT* et des fonctions de contrôle d'une FU.

L'extracteur travaillant en flux vers flux borné, seul le contrôle des données de sortie et *FIFO OUT* sont conservés. Cette FU doit être implémentée de telle manière à ce que l'horloge du contrôle permette l'émission des données de sortie au moins aussi vite que le débit d'échantillon en sortie du CAN. Dans le cas contraire, les données peuvent être corrompues par une FIFO qui se remplit plus vite qu'elle ne se vide. Ces contraintes nécessitent donc un design dédié de l'extracteur au couple front-end d'acquisition analogique et ADC.

L'approche consistant à perdre l'impulsion détectée si aucune FU n'est disponible devient :

**<u>Règle 4 :</u> Pour une FU allouée à un extracteur d'impulsions,** si une impulsion est détectée et que l'impulsion précédente n'a pas encore été envoyée, alors l'impulsion est perdue.

Dès lors, le déclencheur attend la fin de sa durée de déclenchement. De cette manière, une impulsion peut commencer à être mémorisée dans FIFO OUT avant son émission. Le contrôle doit donc communiquer au déclencheur si une transmission est en cours. De cette manière, l'extracteur peut commencer à mémoriser l'impulsion détectée durant le transfert de la précédente. Le début du transfert est renseigné au travers d'une communication directe entre le contrôle et l'extracteur implémentée de manière à respecter les contraintes de l'asynchrone (double registres, FIFO asynchrone). Pour cela, l'extracteur doit également permettre l'ajout des métadonnées *End Packet*. La métadonnée *Flush* peut également être utile pour un déclencheur permettant un contrôle des impulsions.

### 4.1.6 Conclusion sur la conception de l'architecture

Cette partie a présenté un premier modèle d'architecture adapté aux contraintes imposées par le modèle d'exécution tout en étant programmable et flexible en termes de dimensionnement. Pour cela, une architecture composée d'étages de FUs macro-pipelinées a été proposée. Les FUs sont conçues pour répondre aux différentes contraintes imposées par les traitements sur le flot de données et pour permettre une modification aisée de l'élément de calcul programmable. Un réseau d'interconnexion et un ordonnanceur sont proposés pour répondre au besoin de distribution des données entre les étages tout en respectant les contraintes de temps mort. Ce modèle d'architecture peut donc permettre d'atteindre le zéro-temps mort si les performances ou le nombre des FUs sont suffisants pour traiter l'ensemble des impulsions extraites. La Figure 4-12 représente, à titre d'exemple, une architecture composée de trois étages de trois FUs.

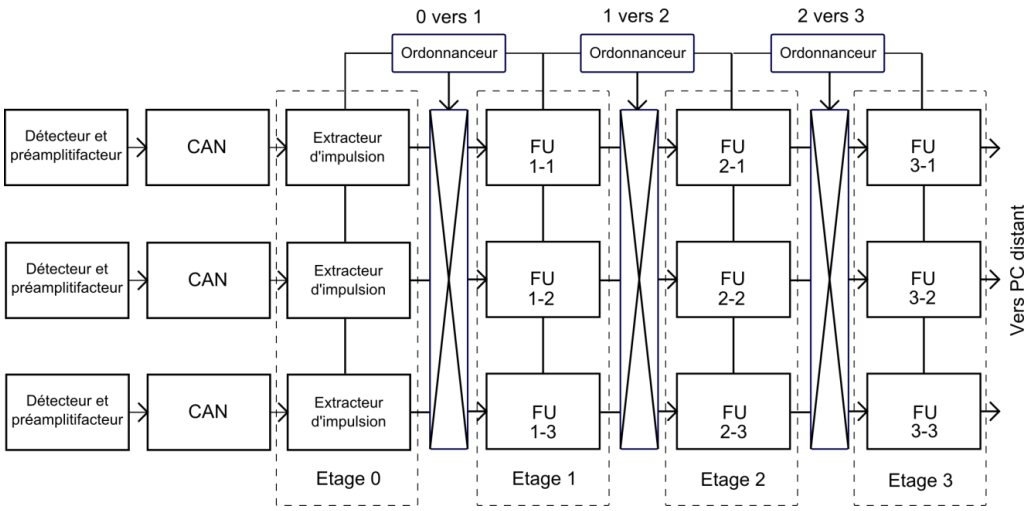



**Figure 4-12 : Schéma bloc d'une architecture à trois étages de trois FUs.**

Afin de valider l'atteinte du zéro-temps mort ainsi que le gain, en termes de FU, apporté par le partage des FUs entre chaque étage, il est nécessaire de réaliser une architecture basée sur l'architecture décrite dans cette section.

## 4.2 Simulation de l'architecture et introduction à l'expérimentation

Cette partie s'intéresse à l'implémentation du simulateur de l'architecture. Ce simulateur a pour objectif, dans un premier temps, de valider le modèle d'exécution sur les points qui n'ont pas pu l'être par l'utilisation du modèle analytique, à savoir l'atteinte du zéro-temps mort et le gain apporté par le partage de ressource.

### 4.2.1 Description de l'étalon

Il n'existe aucun étalon spécifique comparant des architectures de traitement d'impulsions entre elles dans la littérature. D'une manière générale, seuls les résultats des traitements et donc les algorithmes (résolution, figure de mérite) sont testés. Le taux de comptage est également utilisé comme indicateur de performance mais uniquement pour des applications de comptage qui ne se soucient pas de l'intégrité même d'une impulsion.

Cependant, les travaux présentés dans (Cardoso et al. 2004b) utilisent un simulateur dédié pour évaluer l'évolution du temps mort en fonction d'une application donnée et du nombre de tuiles de calcul utilisées. En partant de cette idée, nous proposons un étalon basé sur le nombre d'impulsions perdues par un système de traitement d'impulsions pour un signal donné. Cela peut être par exemple à cause du manque de ressources (mémoire, capacité de calcul etc.). Cela rend donc possible une comparaison équitable entre architectures de traitement des impulsions à condition que le même jeu de données et extracteur d'impulsions soient utilisés. Cet étalon impose les critères suivant :

- comme aucune condition expérimentale dans l'instrumentation nucléaire n'est reproductible (à cause de l'environnement, bruit de fond, de la variabilité de l'activité de la source etc.), il est nécessaire, à l'instar des étalons pour processeurs, d'utiliser un même jeu de données d'entrée pour chaque test. Il est possible de travailler sur des jeux de données synthétiques ou des jeux de données réelles préalablement enregistrées ;
- comme on ne souhaite pas évaluer l'étage de déclenchement, il convient d'imposer le même type de *trigger*/extracteur d'impulsion utilisant des paramètres identiques fixés à l'avance. De cette manière, pour chaque test, autant d'impulsions sont détectées et potentiellement traitées. Le nombre d'impulsions extraites fait donc figure de référence.

Si l'ensemble de ces critères est respecté, alors cet étalon nous permet de tester notre modèle d'exécution et l'architecture qui en découle. Cela permettra aussi de proposer aux architectures de la littérature un moyen équitable de se comparer.

### 4.2.2 Description de l'implémentation du simulateur en SystemC

La conception de notre architecture est réalisée de manière à ne pas être figée par la cible d'implémentation. Lors d'une exploration architecturale, le concepteur de systèmes intégrés cherche en général la meilleure solution de déploiement d'une application logicielle sur une architecture matérielle paramétrable. Pour réaliser à la fois la validation des résultats du modèle d'exécution et la modification des paramètres de l'architecture jusqu'à l'obtention d'une solution satisfaisant les contraintes, nous choisissons de modéliser notre simulateur en C++ grâce à l'utilisation du langage SystemC (Müller et al. 2003). SystemC est un langage de description de matériel. Ce langage se présente sous la forme d'une bibliothèque de classes C++ open-source. La représentation en SystemC, utilisée lors de nos expérimentations, est décrite dans la sous-section suivante.

Il existe dans la littérature des simulateurs permettant l'exploration d'architectures multiprocesseurs asymétriques comme par exemple SESAM (Ventroux et al. 2010). Nous avons cependant choisi de réaliser notre propre simulateur afin de garantir une gestion fine du temps mort.

#### 4.2.2.1 Description du niveau d'abstraction utilisé en SystemC

L'architecture de notre système est composée de plusieurs composants matériels. Les principaux composants sont les FUs comprenant les processeurs, des mémoires, des contrôleurs, mais aussi des réseaux d'interconnexions et des ordonnanceurs. Ces composants sont interconnectés par des signaux. Le comportement de chaque composant matériel est décrit, dans le simulateur, sous la forme de modules comprenant un à plusieurs processus. Les composants de nos simulateurs sont représentés par des modules C++ héritant de la classe *sc_module*. Ces modules utilisent le type *sc_port* pour la communication entre module et le type *sc_signal* pour modéliser les registres. Les registres sont des variables C++, membres du module et de type *sc_signal*. Pour obtenir un modèle générique, le simulateur adopte une approche « bottom-up » en modélisant finement les petits blocs numériques (FIFO, contrôleurs, ordonnanceur et crossbar). Ces briques de base sont ensuite associées pour réaliser les modèles plus complexes comme les FUs. Le processeur est un cas particulier du simulateur. En effet, son choix n'étant pas figé, nous n'avons pas modélisé un type de processeur en particulier. Dans ce simulateur, son rôle est l'exécution d'un des traitements préalablement développés (Chapitre 1), et l'ajout de métadonnées. Finalement, le seul paramètre intéressant pour l'étude du temps mort est la durée d'exécution du traitement pour un échantillon. Enfin, la charge des algorithmes est traduite par le nombre de cycles processeur requis pour traiter un échantillon d'une impulsion. Cette durée peut être corrélée à la durée réelle d'exécution du traitement sur un processeur grâce aux estimations en besoins en ressources de calculs présentés dans le chapitre 2. A tous les composants de notre architecture décrits jusqu'alors, nous ajoutons la description d'un module CAN permettant l'émulation d'un détecteur jusqu'à la conversion analogique-numérique du signal.

La description de notre modèle se situe entre deux niveaux d'abstraction : le premier est TF (Time Functional). La modélisation de la fonctionnalité et des interfaces de communication du système intègre des notions de durée (temps d'exécution des processus) en vue de l'estimation des performances temporelles. Le second est dit CABA (Cycle Accurate Bit Accurate). La modélisation des fonctions des composants et des interfaces de communication de notre système est définie au cycle d'horloge près mais admet l'utilisation de Template pour le format des données.

### 4.2.3 Définition du protocole expérimental

Cette sous-section présente les données d'entrées, ainsi que les différents paramètres utilisés par notre simulateur.

#### 4.2.3.1 Données d'entrées

Les données utilisées pour les tests ont été obtenues à l'aide d'une source Am-241 et d'un détecteur 4πγ NaI(TI)-puits. La conversion analogique numérique est résolue sur 14-bit et cadencée à 125 MHz. Le signal est ensuite mémorisé dans des fichiers correspondant à deux secondes de mesure. Ils contiennent statistiquement plus de 22 000 impulsions au regard de l'activité de la source radioactive. Chaque fichier est alors associé à une voie de mesure modélisée par un CAN en SystemC.

#### 4.2.3.2 Paramètres de simulation

Le simulateur admet les paramètres suivants :

Les paramètres généraux sont :

- le nombre de FUs par étage ;
- le nombre d'étages ;
- la durée de simulation.

Pour les FUs, les paramètres sont homogènes pour un étage :

- l'horloge du contrôle des données d'entrée ;
- l'horloge du contrôle des données de sortie ;
- la taille de *FIFO IN* ;
- la taille de *FIFO OUT* ;
- le traitement du processeur ;
- l'horloge du processeur.

Pour le CAN, les paramètres sont :

- l'horloge du CAN ;
- le fichier de données d'entrée ;
- le format des données d'entrée.

Pour l'ordonnanceur, les paramètres sont :

- sa position dans le macro-pipeline (entre quels étages)
- le nombre de voies d'entrée et de sortie est automatiquement paramétré pour correspondre au nombre de FUs à l'étage d'entrée et à l'étage de sortie ;
- l'horloge de l'ordonnanceur.

Pour le réseau d'interconnexion *crossbar*, les paramètres sont :

- sa position dans le macro-pipeline (entre quels étages)
- le nombre de voies d'entrée et de sortie est automatiquement paramétré pour correspondre au nombre de FUs à l'étage d'entrée et à l'étage de sortie ;
- l'horloge du *crossbar*.

Durant la simulation, des sondes sont utilisées dans chaque FU pour connaitre le nombre de fois où une donnée est perdue par manque de ressources de calcul (FU). La trace temporelle des *sc_signal* est récupérée et sauvegardée au format *vcd* pour être lue avec le logiciel ModelSim afin de valider le comportement de chaque composant à l'instar d'un design VHDL. Un code « maître » s'occupe de l'assemblage des différents composants mais aussi de la récupération des données des sondes et du résultat final des traitements. Les FUs de l'étage 0 seront toujours des FUs encapsulant un extracteur d'impulsion.

### 4.2.4 Résultats de caractérisation

Dans un premier temps nous avons testé unes à unes les différentes voies de mesures utilisées. Pour ce premier test, tout le système est synchrone à l'horloge des données d'acquisition du CAN. Les algorithmes choisis sont capables de travailler sur des entrées en flux borné. De cette manière, il n'y a pas de temps mort. Cinq voies de mesures sont testées et sont nommées A, B, C, D, E. D'autres mesures seront réalisées pour la suite des travaux et pour étendre l'exploration architecturale vers un plus grand nombre de voies d'entrée. Pour chaque voie, l'architecture est configurée avec les paramètres suivants :

- nombre de FUs à l'étage 0 : 1 ;
- nombre de FUs à l'étage 1 : 1 ;
- algorithme des FUs de l'étage 0 : extracteur d'impulsion ;
- algorithme des FUs de l'étage 1 : recherche d'amplitude maximum ;
- période de l'horloge générale : 1 ns (la valeur importe peu si tous les composants ont la même horloge que le CAN);
- profondeur de FIFO OUT 0 : 8500 ;
- profondeur de FIFO IN 1 : 8500 ;
- profondeur de FIFO OUT 1 : 1 (le résultat est l'amplitude maximum);

| Voie de mesure | Nombre d'impulsions maximum traitées |
|---|---|
| $A$ | 22857 |
| $B$ | 22976 |
| $C$ | 22949 |
| $D$ | 22993 |
| $E$ | 22851 |

**Tableau 4-1 : Nombre maximum d'impulsions traitées par voie de mesure**

Pour chaque voie, le nombre d'impulsions détectées et traitées est présenté dans le Tableau 4-1. Il correspond bien au taux de comptage de référence de l'ensemble source/détecteur utilisé pour les mesures. Grâce à ces premiers résultats, nous validons le fonctionnement structurel des composants utilisés mais obtenons aussi les références en termes de nombre d'impulsions traitées nécessaires à l'exécution des instances qui permettront de valider le modèle d'exécution.

#### 4.2.4.1 Distribution des impulsions

Afin de valider la possibilité d'atteindre le zéro-temps mort s'il y a suffisamment de FUs pour traiter les impulsions extraites, l'évolution du nombre de FUs requis pour atteindre le zéro-temps mort est simulée. Pour cela, nous nous proposons d'exécuter plusieurs instances de l'architecture pour une seule voie d'acquisition. Plusieurs exécutions du modèle sont réalisées en utilisant le même jeu de données, mais pour différents nombres de FUs et durées d'exécution du programme. Nous choisissons donc d'observer le nombre d'impulsions perdues pour chaque instance et de réaliser autant d'instances qu'il faut pour atteindre zéro impulsion perdue.

Dans la sous-section 5.2.4, nous avons constaté que les voies de mesures sont identiques en termes de nombre d'impulsions maximum traitées. C'est pourquoi, nous présentons les résultats de la distribution d'impulsions uniquement pour la voie de mesure A.

Le simulateur de l'architecture est configuré avec les paramètres suivants :

- Nombre de FUs à l'étage 0 : 1 ;
- Nombre de FUs à l'étage 1 : 1 à M ;
- Algorithme des FUs de l'étage 0 : extracteur d'impulsion ;
- Algorithme des FUs de l'étage 1 : recherche d'amplitude maximum ;
- Période de l'horloge du CAN : 1 ns ;
- Période de l'horloge des contrôles de données : 1 ns ;
- Période de l'horloge de l'ordonnanceur et du *crossbar* : 1 ns ;
- Période de l'horloge des processeurs de l'étage 1 : 5-15 ns ;
- Profondeur de FIFO OUT 0 : 8500 ;
- Profondeur de FIFO IN 1-M : 8500 ;
- Profondeur de FIFO OUT 1-M : 1 (le résultat est l'amplitude maximum);

Ces paramètres montrent comment nous avons influencé la durée d'exécution des traitements par la modification de l'horloge des processeurs. Par exemple, une horloge d'une période de 15 ns, vis-à-vis d'un CAN et des contrôles cadencés à 1 ns, signifie que le processeur traitera un échantillon en 15 cycles pour un débit d'échantillons d'un cycle par échantillon. Cela correspond à la durée $\tau_2$ du modèle d'exécution exprimée dans le Chapitre 2.

Le nombre d'impulsions perdues par manque de ressources est présenté en Figure 4-13. Trois périodes d'horloge du processeur ont été testées. Cela peut donc correspondre soit à des traitements différents, soit à des différences de performance du processeur. Ces résultats montrent que la distribution des impulsions sur les FUs permet d'atteindre le zéro-temps mort, confirmant ainsi la théorie. Pour une période d'horloge de 15 ns du processeur de l'étage un, six FUs sont nécessaires pour atteindre le zéro-temps mort. Pour 10 ns, quatre FUs sont nécessaires et pour 5 ns, trois FUs. Dans notre exemple, aucune impulsion n'est perdue avec six FUs, quelle que soit la configuration du simulateur. De plus, la figure montre que le gain obtenu par l'ajout de FUs décroit exponentiellement. Par exemple, pour une durée d'exécution de 15 ns, seules 17 impulsions sont « gagnées » par l'ajout d'une sixième FU vis-à-vis de l'instance utilisant cinq FU. Cela équivaut à un gain de 0.07% par rapport au nombre maximum d'impulsions détectées. Cela signifie que dans certain cas, l'architecture est surdimensionnée pour un gain très faible si l'objectif est d'atteindre le zéro-temps mort. Heureusement, l'architecture est conçue pour permettre à l'utilisateur de trouver un compromis puisqu'il est possible d'ajouter où d'enlever des FUs. Ces résultats montrent qu'il est donc possible de dimensionner une architecture qui se « rapproche » du zéro-temps mort si nombre de FU utilisés est sous contraintes.

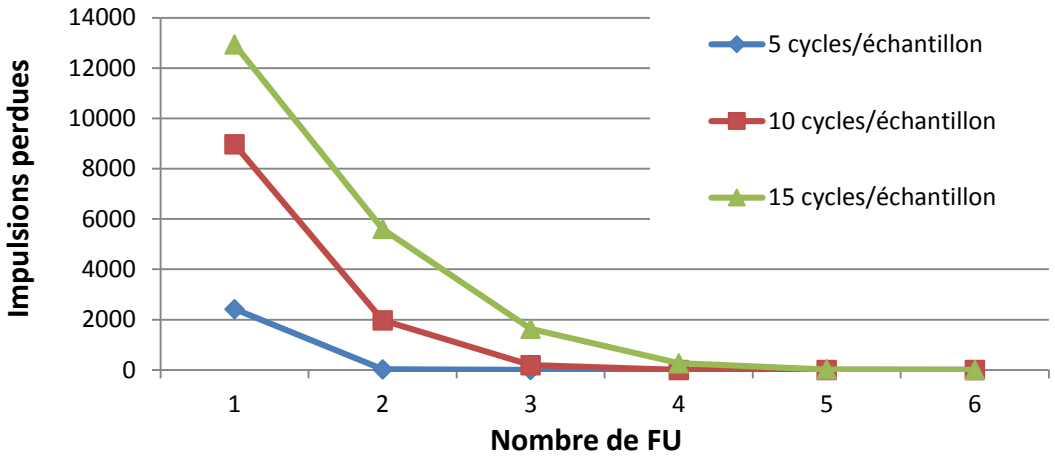



**Figure 4-13 : Résultats de simulation pour une voie d'acquisition relatifs au nombre de FUs nécessaires au zéro-temps mort.**

#### 4.2.4.2 Partage des ressources entre les voies de mesure

Le second rôle du simulateur est de nous permettre d'évaluer le gain, en termes de nombre de FUs, obtenu grâce à notre proposition de partage de FUs entre voies d'acquisition. Pour cela, nous proposons d'exécuter une simulation sur une instance de l'architecture comprenant de deux à cinq voies de mesure acquérant simultanément du signal.

Le simulateur de l'architecture est configuré avec les paramètres suivants :

- nombre de FUs à l'étage 0 : 1 à M ;
- nombre de FUs à l'étage 1 : 1 à M ;
- algorithme des FUs de l'étage 0 : extracteur d'impulsion ;
- algorithme des FUs de l'étage 1 : recherche d'amplitude maximum ;
- période de l'horloge du CAN : 1 ns ;
- période de l'horloge des contrôles de données : 1 ns ;
- période de l'horloge de l'ordonnanceur et du *crossbar* : 1 ns ;
- période de l'horloge des processeurs de l'étage 1 : 1-15 ns ;
- profondeur de FIFO OUT 0-M : 8500 ;
- profondeur de FIFO IN 1-M : 8500 ;
- profondeur de FIFO OUT 1-M : 1 (le résultat est l'amplitude maximum).

Pour comparer le gain entre le modèle simplement distribué et le modèle partagé, le nombre d'impulsions perdues est enregistré pour la voie A uniquement (voie B désactivée), pour la voie B uniquement (voie A désactivée), pour les deux voies sans partage de FUs et pour les deux voies avec le partage de FUs. Ces étapes sont réalisées jusqu'à la comparaison de cinq voies de mesure en parallèle. Dans un premier temps, nous présentons le nombre de FUs requis pour atteindre le zéro-temps mort pour chaque voie dans le Tableau 4-2.

| Protocole d'acquisition | Nombre de FUs requis | | | |
|---|---|---|---|---|
| | 15 $cycles/éch$ | 10 $cycles/éch$ | 5 $cycles/éch$ | 2 $cycles/éch$ |
| Voie A | 6 | 4 | 3 | 2 |
| Voie B | 6 | 4 | 3 | 2 |
| Voie C | 6 | 4 | 3 | 2 |
| Voie D | 6 | 4 | 3 | 2 |
| Voie E | 6 | 4 | 3 | 2 |

**Tableau 4-2 : Nombre de FUs nécessaires pour atteindre le zéro-temps mort par voie de mesure et pour des durées d'exécution différentes.**

Ensuite, la somme du nombre de FUs requis pour chaque association comparée est effectuée pour avoir le nombre total de FUs utilisées sans partage de FUs entre les voies de mesure. Le résultat de cette somme est présenté en Tableau 4-3. Nous pouvons voir qu'il faut par exemple trente FUs pour traiter l'ensemble des impulsions du système pour cinq voies de mesure pour une durée d'exécution de 15 ns. Maintenant, nous souhaitons observer si ce nombre diminue en partageant les FUs entre voies d'acquisition. Le Tableau 4-4 présente les résultats obtenus avec le partage de ressources pour les cinq configurations présentées, avec en gras, le gain en termes de nombre de FUs économisées.

| Protocole d'acquisition | Nombre de FUs requis | | | |
|---|---|---|---|---|
| | 15 $cycles/éch$ | 10 $cycles/éch$ | 5 $cycles/éch$ | 2 $cycles/éch$ |
| Voies A&B | 12 | 8 | 6 | 4 |
| Voies A&B&C | 18 | 12 | 9 | 6 |
| Voies A&B&C&D | 24 | 16 | 12 | 8 |
| Voies A&B&C&D&E | 30 | 20 | 15 | 10 |

**Tableau 4-3 : Nombre de FUs nécessaires pour atteindre le zéro-temps mort pour plusieurs voies de mesure sans partage des FUs entre voies de mesure.**

| Protocole d'acquisition FUs partagées | Nombre de FUs requis | | | |
|---|---|---|---|---|
| | 15 $cycles/éch$ | 10 $cycles/éch$ | 5 $cycles/éch$ | 2 $cycles/éch$ |
| Voies A&B | 9 **(25%)** | 6 **(25%)** | 4 **(33%)** | 2 **(50%)** |
| Voies A&B&C | 13 **(28%)** | 8 **(33%)** | 6 **(33%)** | 3 **(50%)** |
| Voies A&B&C&D | 17 **(29%)** | 11 **(35%)** | 8 **(40%)** | 4 **(50%)** |
| Voies A&B&C&D&E | 20 **(33%)** | 13 **(35%)** | 9 **(40%)** | 5 **(50%)** |

**Tableau 4-4 : Simulation du gain en termes de passage à l'échelle offert par le partage de ressources.**

Pour chaque configuration testée, un nombre significatif de FUs est économisé lorsque le partage des ressources est activé. Cette économie est présentée sous forme de gain relatif au nombre de FUs gagnées par rapport au nombre de FUs initialement nécessaires. Ce gain est de 50% pour les durées d'exécution des programmes les plus faibles. Le gain décroit plus lentement que la durée d'exécution du programme n'augmente. Sa décroissance tend à se stabiliser vers les durées d'exécution les plus élevées. Cela confirme qu'utiliser la caractéristique d'arrivée aléatoire des

impulsions sur chaque voie pour partager les FUs permet un meilleur passage à l'échelle de l'architecture pour les applications multivoies. De plus, on constate une légère augmentation du gain pour les durées d'exécution les plus élevées avec l'augmentation du nombre de voies d'acquisition. Ces résultats sont en accord avec les simulations Matlab du modèle analytique présenté en section 3.6. En conclusion, plus le nombre de voies de mesure est grand, plus il est avantageux de partager les FUs entre ces voies. Ce résultat permet de répondre partiellement à la problématique du passage à l'échelle de l'architecture.

#### 4.2.4.3 Occupation des FUs

L'état *busy* d'une FU étant connu durant la simulation, nous décidons d'analyser leur taux d'occupation. En effet, les résultats sur la distribution des impulsions présentés en 5.2.4.1 montrent qu'un compromis peut être trouvé pour se rapprocher du zéro-temps mort sans sur-dimensionner notre architecture. Cela est confirmé par l'analyse du temps d'occupation d'une FU durant une acquisition comme présenté dans le Tableau 4-5. A titre d'exemple, la sixième FU n'est occupée que 0.34% du temps, son utilité peut donc être remise en question dans certaines applications si le zéro-temps mort n'est pas une contrainte forte. Ces résultats sont principalement dus à l'utilisation d'un ordonnancement semi-prioritaire qui, en cas de disponibilités équivalentes entre au moins deux FUs à un instant donnée donnera d'abord la priorité à la première par ordre numérique. Nous constatons donc que le mode d'ordonnancement impacte grandement la répartition de l'occupation. Ce type d'étude peut en outre permettre d'ouvrir la voie à une exploration pour, par exemple, optimiser la consommation électrique globale de l'architecture ainsi que sa température de fonctionnement.

| **Nombre de FUs** | **Taux d'occupation d'une FUs durant la mesure (en %)** | | | | | |
|---|---|---|---|---|---|---|
| | **FU 1** | **FU 2** | **FU 3** | **FU 4** | **FU 5** | **FU 6** |
| **1** | 97.71 | $N.A$ | $N.A$ | $N.A$ | $N.A$ | $N.A$ |
| **2** | 88.43 | 82.79 | $N.A$ | $N.A$ | $N.A$ | $N.A$ |
| **3** | 81.44 | 72.78 | 57.84 | $N.A$ | $N.A$ | $N.A$ |
| **4** | 75.42 | 68.73 | 52.28 | 29.11 | $N.A$ | $N.A$ |
| **5** | 75.20 | 67.09 | 49.90 | 27.19 | 7.63 | $N.A$ |
| **6** | 75.20 | 67.09 | 49.96 | 26.79 | 7.63 | 0.34 |

**Tableau 4-5 : Simulation du temps d'occupation des FUs pour 15 cycles/échantillon.**

### 4.2.5 Conclusion

Cette section a présenté le simulateur utilisé pour valider le modèle d'exécution pour les critères du zéro-temps mort et la possibilité d'un gain apporté par le partage de ressources en termes de passage à l'échelle. Le simulateur développé en SystemC montre des résultats préliminaires prometteurs pour ces deux critères. D'autres résultats sont en cours de réalisation, notamment pour l'étude de l'impact d'un réseau d'interconnexion admettant de la congestion mais ayant une meilleure capacité de passage à l'échelle que le *crossbar* sur le dimensionnement d'une architecture zéro-temps mort. L'étude d'autres ordonnanceurs, par exemple aléatoires, est en cours. D'autres travaux en cours ont pour objectif d'utiliser ce simulateur comme outil de dimensionnement de l'architecture. A relativement court terme, pour des traitements, des types de processeurs et des détecteurs donnés, il sera possible de définir le nombre de FUs nécessaires pour atteindre le zéro-temps mort ou s'en approcher.

## 4.3 Implémentation d'un démonstrateur

La section précédente présente une première version de l'architecture répondant aux contraintes de programmabilité et de la possibilité d'atteindre une capacité de traitement sans temps-mort. Cette section s'intéresse à la conception d'un démonstrateur basée sur cette architecture. Ce démonstrateur s'inscrit dans le cadre d'un projet plus important. Ce projet à pour objectif la création d'une plateforme de mesure et de traitement intégrant plusieurs voies d'acquisition de signaux issues du monde physique et utilisant le réseau Gigabit Ethernet pour communiquer avec une matrice de processeurs (ARM) comme illustré en Figure 4-14. Tel que présenté, si pour chaque élément d'acquisition est associé une FU encapsulant un extracteur d'impulsion, alors notre modèle d'exécution et la première définition de l'architecture associée semblent correspondre aux attentes de ce projet.

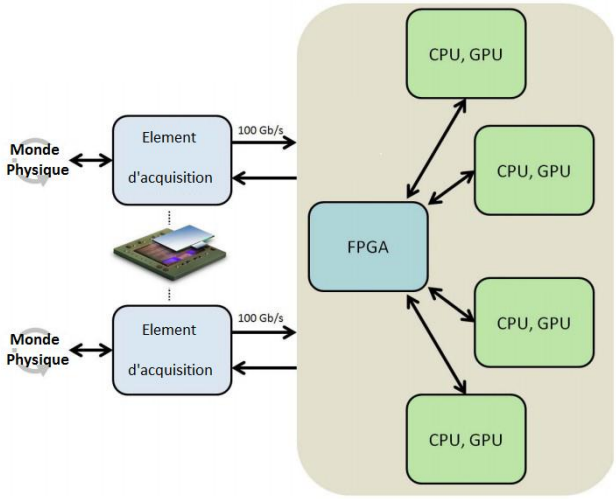



**Figure 4-14 : Illustration du projet motivant l'implémentation d'un démonstrateur.**

Afin de préparer une maquette en vue d'une intégration future sur SoC à base de composants ARM et de réseaux sur puce 100Gb Ethernet, nous avons utilisé des cartes de développement de circuits logiques programmables de type ZedBoard. Ces cartes de développement exploitent le SoC Zynq-7000 de Xilinx. Ce dernier se compose d'une partie programmable (processeur dual-core Cortex-A9), permettant une certaine flexibilité de développement et d'une partie reconfigurable (FPGA). Afin de simplifier le démonstrateur et d'homogénéiser les moyens de communication inter-FU, nous considèrerons qu'une carte fait office de FU. Nous utiliserons également un module permettant d'exploiter jusqu'à quatre ports physiques Ethernet par carte (FU), appelés Ethernet FMC (Opsero 2014). Ce dernier permet de bénéficier d'un port Ethernet en entrée et d'un port Ethernet en sortie afin de constituer la topologie en étages présentée dans ce manuscrit.

Comme l'illustre la Figure 4-15 qui représente un étage complet, un switch Ethernet est utilisé pour l'interconnexion des ZedBoards. Cette figure met également en évidence que l'implémentation proposée de notre architecture nécessite d'être adaptée. En effet, la fonction d'ordonnancement ne peut plus être facilement centralisée. C'est pourquoi, dans un premier temps, la conception et la mise en œuvre d'un ordonnanceur distribué sur chaque ZedBoard sont étudiées. De même, le protocole Ethernet introduit une étape mémorisation/empaquetage des données avant envoi qui introduit des délais et donc du temps mort. Les problématiques de communication et d'ordonnancement sont étudiées et des premiers résultats sont ensuite présentés.

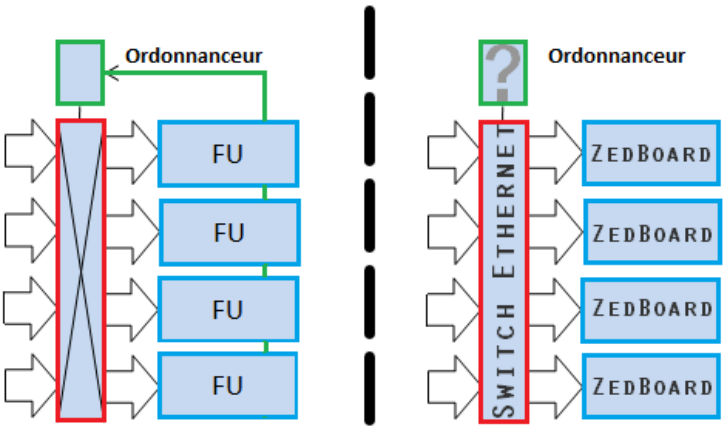



**Figure 4-15 : Opposition entre le modèle de l'architecture et un modèle à base de composants sur étagère.**

### 4.3.1 Problématiques liées à l'utilisation du protocole Ethernet

Le Protocol Ethernet est optimisé pour la transmission de paquets. Cela nécessite de mémoriser préalablement les données dans une mémoire-tampon. En effet, l'émission dans un mode flux borné en utilisant le protocole Ethernet engendre un encombrement du réseau par l'émission de trames de contrôle à chaque envoi d'un échantillon. Dans l'architecture présentée dans ce chapitre, le mode d'émission par paquet admet déjà un temps mort de mémorisation avant envoi. Dans ce cas de figure, la solution est de restreindre les traitements présents sur une FU à l'utilisation du mode paquet pour l'émission des résultats et donc à l'utilisation du prototype de fonction associé.

Avec l'utilisation du protocole Ethernet, des délais de communication variables vont apparaître. Heureusement, en exploitant l'architecture décrite dans ce chapitre, ces délais vont simplement s'ajouter au temps mort induit par le programme de traitement, ce qui nécessitera par contre plus de ressources/ZedBoards pour atteindre le zéro-temps mort, c'est-à-dire ne jamais perdre d'impulsions. Le temps mort d'une FU peut alors s'exprimé par $\tau = \tau_1 + \tau_2 + \tau_3$, avec $\tau_1$ la durée de réception et de mémorisation des données, $\tau_2$ la durée du traitement et $\tau_3$ la durée de transfert des données. Afin de minimiser le temps mort de communication et ainsi perdre un minimum de données. Nous choisissons d'employer le protocole UDP.

Les données n'étant plus mémorisées dans une FIFO avant l'envoi, une optimisation possible est de se servir des métadonnées *End Packet* pour dimensionner la mémoire-tampon afin qu'elle puisse mémoriser plusieurs impulsions. Un dimensionnement basé sur l'utilisation de la variance du taux d'impulsion, du débit moyen d'émission de ces impulsions et du nombre de ZedBoard présents sur le réseau est en cours d'étude.

### 4.3.2 Ordonnancement distribué

L'ordonnanceur actuel est directement connecté à chaque FU des deux étages qu'il relie. Les composants sur étagère choisis pour la mise en œuvre de la preuve de concept rendent difficile l'exploitation de cette approche. C'est pourquoi une solution plus facile à mettre en œuvre, limitant cette quantité de connexions physiques, est envisagée. Il s'agit d'un ordonnancement distribué sur chaque FU : celles-ci communiqueront directement entre elles leur état, disponible ou occupée, et adapteront leur comportement en conséquence. Pour cela, deux cas de figure demandant exigeant

toutes deux autant de stratégies d'ordonnancement se distinguent. Pour N, un étage du pipeline et M le nombre de FUs présentes à cet étage :

- N(1) vers N+1(M) : dans ce cas de figure, il y a concurrence entre les FUs de l'étage N+1 aptes à recevoir les données de la FU de l'étage N. Pour cela, chaque FU réceptrice diffuse sa disponibilité sur le réseau par une trame destinée à la FU émettrice. Cette dernière utilise l'ordonnancement première-FU-disponible, première-FU-servie sans ajout de la règle PCO puisqu'il n'y a qu'une seule entrée et donc pas de concurrence. Dès la présence de données à émettre, la première entrée de la liste est lue et choisie comme adresse d'émission. Sur le principe d'une FIFO, cette adresse est donc effacée, ce qui correspond à une adresse considérée comme occupée, jusqu'à ce que la FU à cette adresse manifeste à nouveau sa disponibilité. le protocole UDP se trouve particulièrement efficace pour gérer ce genre de situation. En effet, celui-ci permet d'ignorer les paquets ne pouvant être emmagasiné dans la mémoire-tampon de réception, empêchant ainsi ces derniers d'écraser les précédents.

- N(M) vers N+1(1) ou vers N+1(M) : dans ce cas de figure, l'arrivée non déterministe de données à émettre sur plusieurs FUs rend probable l'émission simultanée de données. Si le protocole Ethernet gère nativement la présence de plusieurs données sur le même réseau, cela pose le problème de choisir la FU réceptrice. En effet, des FUs de l'étage N+1 peuvent diffuser leur disponibilité sur le réseau à tout instant. De ce fait, les tables de disponibilités des FUs de l'étage N peuvent être les mêmes. Aussi, si des données sont prêtes à être émises au même instant sur les FUs de l'étage N, sans ajout de contraintes, les ordonnanceurs choisiront la même adresse de destination et des données seront perdues. Pour ce cas de figure, nous proposons que les FU prêtes à l'envoi communiquent cet état aux FUs du même étage. En cas d'égalité, la règle PCO est appliquée.

### 4.3.3 Résultats pour une voie

Les premiers tests ont été réalisés pour une voie de mesure en entrée. Cette voie de mesure est simulée grâce à un PC. Pour cela, un ensemble de données réelles préalablement mesurées et numérisées est utilisé. Les impulsions de ce signal sont ensuite extraites avec l'extracteur d'impulsions proposé au Chapitre 3. Les intervalles de temps entre l'envoi de chaque impulsion sont contrôlés selon un processus poissonnien homogène fonction d'un taux d'impulsion paramétrable. Il est ainsi possible de faire varier le taux d'impulsion, d'en observer l'effet sur le comportement de la maquette. Cette dernière se divise en deux programmes : un serveur (étage N) présent sur l'ordinateur, et des clients (étage N+1) sur chaque ZedBoard.

Le serveur se compose de trois processus légers :

- le premier processus traite en continu les trames que le PC reçoit depuis l'étage N+1, afin de mettre à jour la liste de clients disponibles. Pour cela, il lit le contenu de la mémoire-tampon de réception et vérifie qu'il s'agit d'une trame indiquant la disponibilité d'un client. Si ce n'est pas le cas, il l'ignore. Si c'est le cas, il ajoute à la fin de la liste de disponibilité l'adresse du client qui a envoyé cette trame ;

- le second processus simule l'arrivée des impulsions sur l'ordinateur. Il s'agit d'une boucle qui va lire les impulsions préalablement extraites. Dès la lecture du fichier, un délai fonction de la taille de l'impulsion est utilisé pour simuler le comportement du *trigger* et la durée théorique de mémorisation de l'impulsion. Après cette attente, une vérification de la liste de disponibilités est effectuée. Entre chaque impulsion lue, un délai préalablement obtenu par calcul (pour correspondre à un taux d'impulsion donné) est utilisé pour simuler le processus poissonnien. Si aucun client n'est disponible, l'impulsion est perdue. Si un client est disponible, on récupère son adresse et on la supprime de la liste. Puis, ce processus signale à un troisième processus différentes informations qui peuvent être ajoutées comme métadonnées (adresse du client, numéro de l'impulsion, taille etc.) ;

- le troisième et dernier processus est utilisé pour construire le paquet de données. Il prend les paramètres reçus du deuxième et s'en sert pour envoyer construire une trame contenant les métadonnées de l'impulsion et l'impulsion elle-même.

Le rôle du client consiste à recevoir toutes les trames qui lui sont destinées. Une fois le contenu d'une trame mémorisé, il diffuse sur le réseau une trame pour indiquer qu'il est disponible. Il analyse ensuite le contenu de celle-ci pour dissocier les données des métadonnées afin d'envoyer les données au traitement en fonction des métadonnées lues. Une fois le traitement terminé, le client mémorise à nouveau la mémoire-tampon de réception avant de signaler sa disponibilité.

Pour le moment, la maquette a été testée avec trois FUs afin de visualiser le gain en termes d'impulsions traitées par l'ajout de FUs. Une photo de la maquette est présentée en Figure 4-16. Pour réaliser ces tests, les données d'entrées ont été paramétrées pour qu'elles correspondent à un signal avec un taux d'impulsion d'environ 11000 impulsions par seconde pour se rapprocher du signal réel utilisé dans la section 4.2. Une application de spectrométrie est testée : le traitement des impulsions (recherche d'un maximum et incrémentation de l'histogramme). Le Tableau 4-6 présente les résultats obtenus pour une seconde de mesure et pour cinq tentatives.

| **Nombre de FUs** | **1** | **2** | | **3** | | |
|---|---|---|---|---|---|---|
| **Tentative 1** | 2200 | 2157 | 2155 | 2128 | 2115 | 2115 |
| **Tentative 2** | 2214 | 2187 | 2169 | 2079 | 2081 | 2089 |
| **Tentative 3** | 2209 | 2158 | 2131 | 2035 | 2026 | 2028 |
| **Tentative 4** | 2223 | 2195 | 2184 | 2088 | 2081 | 2074 |
| **Tentative 5** | 2220 | 2099 | 2092 | 2144 | 2130 | 2120 |
| **Nombre total moyen d'impulsions traitées** | 2213 | 4305 | | 6266 | | |

**Tableau 4-6 : Résultats d'implémentation de la preuve de concept pour une voie de mesure et trois FUs.**

Les résultats montrent qu'il est possible de traiter environ 20 % des impulsions avec une seule FU. Ce nombre passe ensuite à 40 % pour deux FUs, puis 60 % pour trois FUs, ce qui permet de confirmer les résultats obtenus avec le simulateur SystemC. Le nombre d'impulsions traitées par FU

diminue légèrement avec l'ajout de FUs, ce qui permet de confirmer que le gain tend vers un seuil avant de se rapprocher du zéro-temps mort comme illustré en Figure 4-13.

### 4.3.4 Travaux en cours

Les travaux présents s'intéressent à l'interfaçage de la partie acquisition d'une carte PING (Normand et al. 2009) avec une Zedboard dédiée à l'extracteur d'impulsion afin de réaliser des mesures en temps réel et de tester ainsi différentes applications pour différentes configurations de ZedBoard. La mise en œuvre d'Ethernet FMC, le module mentionné dans la présentation des composants utilisés, est actuellement en cours afin de permettre un macro-pipeline de ZedBoard. Enfin, l'étude de l'impact de l'ordonnancement distribué choisi sur la congestion du réseau Ethernet est en cours de réalisation. Si le nombre de FUs augmente, l'utilisation de trames pour renseigner l'état des FUs risque d'encombrer le réseau, ce qui doit être confronté au passage à l'échelle du système.

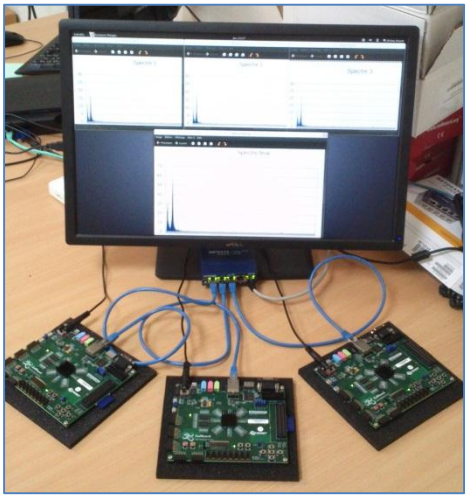


**Figure 4-16 : Maquette de l'architecture équipée de trois ZedBoards.**

## 4.4 Conclusion

Dans ce chapitre, une architecture multivoie de traitement des impulsions pouvant atteindre le zéro-temps mort est proposée. Cette architecture est programmable et particulièrement adaptée au traitement numérique des impulsions pour l'instrumentation nucléaire. La séparation des impulsions et leur distribution sur différents étages pipelinés de FUs programmables résout le problème du temps mort et de la programmabilité. Les FUs sont conçues pour permettre de modifier facilement l'élément de calcul programmable afin de rendre l'architecture plus flexible en termes de dimensionnement. Un simulateur développé en SystemC est utilisé pour valider les résultats théoriques issus du modèle d'exécution. Ce simulateur montre des résultats prometteurs dans la possibilité d'atteindre le zéro-temps mort par l'ajout de FUs. Ces résultats montrent également que le passage à l'échelle pour les applications multivoies est grandement amélioré par le partage des FUs entre voies de mesure. Une maquette basée sur l'utilisation de composants sur étagère est également proposée. Elle montre qu'il est possible d'exploiter l'architecture définie sans avoir à fondre un SoC. Les résultats préliminaires obtenus avec cette maquette permettent d'être optimiste puisqu'ils se rapprochent des résultats obtenus par simulation.

# Conclusion et perspectives

## Conclusion des travaux

Dans l'instrumentation nucléaire, le signal mesuré est composé d'impulsions dont la date d'arrivée est aléatoire et dont l'amplitude et la durée sont non-déterministes. Cette caractéristique impose aux architectures actuelles de travailler en flux de données. En effet, si le débit d'impulsion est trop élevé, les besoins en temps réel impliquent de paralyser l'acquisition du signal durant le traitement d'une impulsion. Durant ce délai, appelé temps mort, des impulsions peuvent être perdues. Cette contrainte conduit les architectures actuelles à utiliser des solutions dédiées à base de FPGA. La grande variété d'applications nécessite de disposer d'une architecture flexible pouvant être aisément reprogrammée. De plus, l'exigence de mesures en temps réel impose de disposer d'une grande capacité de calcul et de bande passante pour la mise en forme et l'extraction des caractéristiques des impulsions. Enfin, ces architectures doivent être capables de passer à l'échelle pour supporter des applications nécessitant parfois un très grand nombre de voies de mesure.

L'objectif des travaux de cette thèse était d'aboutir à une architecture de traitement du signal couvrant l'ensemble des applications du domaine de l'instrumentation nucléaire tout en étant capable de répondre aux besoins de programmabilité, de multivoie et d'être zéro-temps mort sans avoir à dimensionner une architecture de calcul au pire cas, ce qui se révélerait coûteux.

La démarche entreprise au cours de ce travail était organisée comme suit.

Tout d'abord, nous avons étudié les limitations intrinsèques des chaînes d'instrumentation nucléaire traditionnelles qui obligent les concepteurs à réaliser des architectures dédiées. Cette étude a montré qu'une architecture de traitement du signal couvrant les applications du domaine de l'instrumentation nucléaire devrait être capable de répondre à trois problématiques : la programmabilité, la gestion de plusieurs voies de mesures et la gestion du temps mort. Pour comprendre quelles seraient les caractéristiques d'une architecture respectant ces critères, nous avons caractérisé les besoins en capacité de calcul par l'analyse des applications, des algorithmes et des détecteurs. Cette caractérisation a fait apparaître des besoins en capacité de calcul élevés sous peine de temps mort. Cependant, l'analyse de l'état de l'art des architectures de traitement des impulsions montre qu'il est possible de modifier le modèle d'exécution traditionnel de gestion en flux pour atténuer ces besoins.

Ensuite, nous avons défini et étudié un modèle d'exécution adapté aux besoins d'une architecture générique pour les applications de l'instrumentation nucléaire. Ce modèle se divise en quatre étapes. Dans un premier temps, les impulsions sont extraites afin que les traitements ne travaillent plus sur un flux mais sur des flux bornés ne contenant que des données utiles. Ces impulsions sont ensuite distribuées en fonction de la disponibilité des ressources au sein d'unités de calcul programmables partagées. Ces unités de calcul fonctionnent de manière autonome, ce qui leur permet de s'adapter à la variabilité des données et des traitements. Désormais, la perte d'impulsions n'est plus due à la durée d'exécution des traitements s'il y a suffisamment d'unités de calculs pour les traiter.

C'est pourquoi, les travaux suivants ont consisté à développer un extracteur d'impulsions. Ils montrent qu'un tel extracteur est capable de s'adapter dynamiquement aux différentes tailles et formes d'impulsions afin d'être utilisé avec n'importe quel détecteur mais surtout pour n'importe quelle application. Il a ensuite été comparé avec les autres méthodes de déclenchement de la littérature et a montré de meilleurs résultats pour une application traditionnelle de spectrométrie, en particulier dans le cas d'une mesure de courte durée pour deux types de mesures : amplitude et aire.

La dernière partie des travaux présentés dans ce manuscrit définit l'architecture programmable basée sur le modèle d'exécution permettant le traitement des impulsions extraites ainsi qu'en une première proposition d'implémentation afin de valider le modèle d'exécution. Il y est décrit dans un premier temps la conception d'une unité fonctionnelle (FU). Cette FU combine à la fois un processeur pour les traitements, des mémoires pour l'acheminement des données et un système de contrôle des transferts des données. Elle a été conçue pour permettre de modifier facilement l'élément de calcul programmable afin de rendre l'architecture plus flexible en termes de dimensionnement. Ensuite, les composants tels que le réseau d'interconnexion et l'ordonnanceur permettant de connecter ces FUs ont été décrits. Enfin, une solution permettant l'encapsulation de l'extracteur d'impulsion proposé dans une FU est détaillée. L'ensemble de ces éléments permet d'aboutir à une architecture qui correspond au modèle d'exécution, permettant donc l'atteinte du zéro-temps mort et la gestion du multivoie tout en étant programmable. Pour vérifier cette attente, un simulateur en SystemC a été développé. Les résultats obtenus par le simulateur ont montré qu'il est possible d'atteindre le zéro-temps mort si un nombre suffisant de FUs est présent dans notre architecture. Ils ont également montré des résultats prometteurs en termes de passage à l'échelle pour les applications multivoies grâce au partage des FUs entre voies de mesure. Finalement, la mise en œuvre d'une première preuve de concept basée sur le modèle d'exécution et à partir de composants sur étagères a été présentée ; les résultats préliminaires obtenus se rapprochent des résultats obtenus par simulation.

Cette thèse formule une démarche complète ayant permis, uniquement à partir de la volonté de rendre programmables les chaînes de mesures de l'instrumentation nucléaire, la conception d'une architecture programmable, zéro-temps mort et multivoie. La solution proposée inclut la définition :

- d'un modèle d'exécution adaptable à l'ensemble des chaînes de mesure du mode impulsion ;
- d'un déclencheur générique adapté à n'importe quel détecteur délivrant des impulsions ;
- d'une architecture numérique avec la description de son fonctionnement global et une première proposition d'implémentation ;
- d'un simulateur permettant de tester différentes configurations de l'architecture et différents programmes ;
- d'une première preuve de concept utilisant des composants du marché.

Ces travaux ouvrent la voie à des applications innovantes de traitement numérique des impulsions jusqu'alors limitées aux traitements hors-lignes à cause du temps mort et de l'utilisation de composants dédiés. Ils ont été sélectionnés dans le cadre d'un projet qui pourrait nous permettre de réaliser un SoC.

## Perspectives

Les travaux présentés dans cette thèse ont permis de lever les verrous qui freinent l'utilisation de composants programmables dans les architectures électroniques de traitement des impulsions pour l'instrumentation nucléaire : la gestion du temps mort, le besoin de flexibilité en termes de dimensionnement dans un environnement multi-applicatif et multi-détecteur et la gestion du multivoie. Cette section présente les différentes voies ouvertes par les travaux de cette thèse. Elle commence par présenter les perspectives d'exploration de l'architecture. Les prochains travaux effectués sur le simulateur développé sont ensuite décrits. Enfin, ceux concernant la preuve de concept et son utilisation sont détaillées.

**Exploration architecturale**

Le modèle d'architecture présenté utilise un *crossbar* pour l'implémentation du réseau d'interconnexion. Si la surface est une contrainte forte, il faut limiter le nombre d'éléments interconnectés par le *crossbar*. C'est pourquoi, il est nécessaire d'évaluer à l'aide du simulateur d'autres solutions existantes pour accroître la capacité de passage à l'échelle de l'architecture. Cependant, ces solutions peuvent limiter le nombre d'interconnexions, impliquant que plus de FUs seront nécessaires pour s'assurer de traiter toute les impulsions, ou ne garantissent plus le support de plusieurs communications simultanées sans ajouter de congestion au réseau. Un compromis doit donc être trouvé afin de pouvoir adapter cette architecture aux applications les plus gourmandes en termes de FUs et utilisant un grand nombre de voies de mesure.

L'ordonnancement actuel est dirigé par l'état de disponibilité des FUs émettrices et réceptrices. Il ne prend en considération ni la taille des impulsions (qui est non-déterministe), ni la durée d'exécution des traitements (qui est non-déterministe également mais qui dépend de la taille des impulsions en grande partie). C'est pourquoi, d'autres types d'ordonnancement doivent être étudiés, comme des ordonnancements aléatoires ou basés sur des *a priori* sur la durée d'exécution des traitements en fonction de la taille des impulsions qu'il distribue.

**Simulateur**

Le simulateur présenté, s'il permet d'analyser le comportement de l'architecture pour différentes configurations et différents composants, doit permettre *in fine* le dimensionnement de l'architecture. En effet, s'il est possible de calculer la durée d'exécution d'un algorithme en fonction du processeur utilisé pour son exécution, il devient alors possible de prédire le nombre de FUs nécessaire à l'atteinte (ou se rapprochant) du zéro-temps mort pour une application donnée. En fonction d'un signal préalablement mémorisé ou synthétique et d'un ou plusieurs traitements, l'utilisateur final doit obtenir un nombre de FUs fonction du taux d'impulsion par seconde traitées. En fonction des résultats obtenus, l'utilisateur s'approchera des dimensions de l'architecture souhaitées. Un grand nombre d'algorithmes étant déjà préalablement développés en C, l'étape suivante consistera à créer un émulateur permettant de connaître en moyenne la durée d'exécution en fonction d'un processeur donné.

**Preuve de concept**

Pour le moment, la maquette actuelle de l'architecture n'a été testée qu'avec une seule voie d'entrée (rôle attribué à un PC). D'autres configurations de cette maquette sont envisagées. Une configuration avec plusieurs PC en entrée sera réalisée, puis, à l'aide du module EthernetFMC, plusieurs

étages de ZedBoard seront ajoutés. En parallèle, il sera étudié l'impact sur la congestion du réseau par l'utilisation de l'ordonnancement distribué. Ensuite, nous envisageons de modifier la partie acquisition d'une carte PING (Normand et al. 2009) avec une Zedboard qui jouera le rôle d'une FU dédiée à l'extracteur d'impulsion. Ce système sera alors testé dans des conditions réelles de mesure afin de le comparer aux architectures existantes du laboratoire puis de la littérature pour des applications traditionnelles. Enfin, à l'aide de cette plateforme, nous envisageons le test d'applications encore jamais exploitées en ligne ; notamment des applications de discrimination neutron-gamma à base de transformées de Fourier ou de transformées en ondelettes.

# Bibliographie personnelle

## Brevets :

- Y. Moline, M. Thevenin, G. Corre, T. Peyret, “Procédé et système d’extraction dynamique d'impulsions dans un signal temporel bruité” Brevet numéro : S 55 544 TM, 2014.

- Y. Moline, M. Thevenin, G. Corre, “Système, procédé et programme d'ordinateur pour la numérisation d'impulsions rapides sans temps mort” Brevet numéro : S 58 541 TM, 2015.

## Revues avec comité de lecture :

- Y. Moline, M. Thevenin, G. Corre, M. Paindavoine. “Auto-Adaptive Trigger and Pulse Extraction for Digital Processing in Nuclear Instrumentation” in *IEEE Transactions on Nuclear Science*, 2015. DOI: 10.1109/TNS.2015.2411857.
- J. Dumazert, R. Coulon, V. Kondrasovs, K. Boudergui, Y. Moline, G. Sannié, J. Gameiro, S. Normand, L. Méchin, “A robust hypothesis test for the sensitive detection of constant speed radiation moving sources”, in *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 2015, DOI:10.1016/j.nima.2015.06.016.

## Conférence internationale avec acte :

- Y. Moline, M. Thevenin, G. Corre, M. Paindavoine. “A programmable and zero dead-time architecture for Nuclear Instrumentation” in *Advancements in Nuclear Instrumentation Measurement Methods and their Applications* (*ANIMMA 2015*), 2015.

## Conférences internationales francophones avec actes et colloques :

- Y. Moline, M. Thevenin, G. Corre, M. Paindavoine. “Vers une architecture électronique unifiée et zéro-temps mort pour l'instrumentation nucléaire” in *Conférence en Parallélisme, Architecture et Système (ComPAS’2014)*, 2014.

- Y. Moline, M. Thevenin, G. Corre, M. Paindavoine “Vers une architecture électronique unifiée et zéro-temps mort pour l'instrumentation nucléaire” in *Groupement de Recherche en « System-On-Chip » et « System-In-Package » (GDR SOC-SIP 2014)*, 2014.

- Y. Moline, M. Thevenin, G. Corre, M. Paindavoine “Une architecture programmable de traitement des impulsions pour l’instrumentation nucléaire” in *Groupement de Recherche en Traitement du Signal et des Images (GRETSI 2015)*, 2015.

# Bibliographie

Barbot, L. et al., 2015. On Line Neutron Flux Mapping in Fuel Coolant Channels of a Research Reactor. *IEEE Transactions on Nuclear Science*, 62(2), pp.415–419. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7058454.

Basilio Simoes, J. & Cardoso, J.M.R., 1994. A PC104 Multiprocessor DSP System For Radiation Spectroscopy Applications. Available at: http://people.na.infn.it/~barbarin/Biblioteca.

Bazzacco, D., 2004. The Advanced Gamma Ray Tracking Array AGATA. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 746, pp.248–254. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0375947404009625 [Accessed January 21, 2014].

Belli, F. et al., 2013. A study on the pulse height resolution of organic scintillator digitized pulses. *Fusion Engineering and Design*, 88(6-8), pp.1271–1275. Available at: http://dx.doi.org/10.1016/j.fusengdes.2012.12.030.

Bertrand, G.H.V. et al., 2015. Pulse shape discrimination between (fast or thermal) neutrons and gamma rays with plastic scintillators: State of the art. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 776, pp.114–128. Available at: http://linkinghub.elsevier.com/retrieve/pii/S016890021401465X.

Blanc, P. et al., 2014. Neutron/gamma pulse shape discrimination in plastic scintillators: Preparation and characterization of various compositions. *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 750, pp.1–11. Available at: http://dx.doi.org/10.1016/j.nima.2014.02.053.

CAEN, 2010. Application Note AN2503 Charge Integration: Analog Vs. Digital. , pp.1–8.

CAEN, 2011. WP2081 Digital Pulse Processing in Nuclear Physics. , (August).

Canberra, 2012. High-purity Germanium (HPGe) Detectors. *Germanium-Det-SS-C39606*. Available at: http://www.canberra.com/products/detectors/germanium-detectors.asp.

Carasco, C. et al., 2010. Material characterization in cemented radioactive waste with the associated particle technique. *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 619(1-3), pp.432–435.

Cardoso, J.M.., Basilio Simoes, J. & Correia, C.M.B.., 2004a. A High Performance Reconfigurable Hardware Platform for Digital Pulse Processing. *Nuclear Science, IEEE transactions on*, 51(3), pp.921–925. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1311992 [Accessed December 19, 2013].

Cardoso, J.M.., Basilio Simoes, J. & Correia, C.M.B.., 1999. A mixed Analog-digital Pulse Spectrometer. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 422(1-3), pp.400–404. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0168900298009917.

Cardoso, J.M.., Basilio Simoes, J. & Correia, C.M.B.., 2004b. Dead-time Analysis of Digital Spectrometers. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 522(3), pp.487–494. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0168900203033151 [Accessed January 14, 2014].

Cardoso, J.M.R., Basilio Simoes, J. & Correia, C.M.B.A., 1999. A mixed Analog-digital Pulse Spectrometer. *Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip.*, 422(1-3), pp.400–404. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0168900298009917.

Chao, Y.J., 2002. Procede et systeme d'arbre binaire de programmation par multiplexage. Available at: http://www.google.com/patents/CA2450008A1?cl=fr.

Chen, L. et al., 2013. Digital Beta Counting and Pulse-shape Analysis for High-precision Nuclear Beta Decay Half-life Measurements: Tested on. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 728, pp.81–91. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0168900213008644 [Accessed December 16, 2013].

Corre, G., Kondrasovs, V. & Normand, S., 2009. Method Capable Of Discriminating Between A Gamma Component And A Neutron Component In An Electronic Signal.

Craig, E.C., 1993. *Electronics via Waveform Analysis*, Cambridge City. Available at: http://www.springerlink.com/index/Q2NR3L3J01712744.pdf [Accessed December 2, 2013].

Elhanany, I. et al., 1999. A Novel Architecture for Digital Pulse Height Analysis with Application to Radiation Spectroscopy. In *7th Mediterranean Conference on Control and Automation*. pp. 2143–2151. Available at: http://med.ee.nd.edu/MED7-1999/med99/papers/MED057.pdf [Accessed December 2, 2013].

Esmaeili-Sani, V. et al., 2011. Triangle bipolar pulse shaping and pileup correction based on DSP. *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 665, pp.11–14. Available at: http://linkinghub.elsevier.com/retrieve/pii/S016890021102081X [Accessed December 16, 2013].

Faisal, M. et al., 2013. A Data Processing System for Real-Time Pulse Processing and Timing Enhancement for Nuclear Particle Detection Systems. *Nuclear Science, IEEE Transactions on*, 60(2), pp.619–623. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6490451.

Fallu-Labruyere, A. et al., 2007. Time Resolution Studies Using Digital Constant Fraction Discrimination. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 579(1), pp.247–251. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0168900207006213 [Accessed March 13, 2014].

Fanet, H., 2002. *Electronique Associee aux Detecteurs de Rayonnements* Génie Nucl., Paris: Techniques de l'ingénieur.

Flaska, M. et al., 2013. Influence of sampling properties of fast-waveform digitizers on neutron-gamma-ray, pulse-shape discrimination for organic scintillation detectors. *Nuclear Instruments*

*and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 729(July 2015), pp.456–462.

Flynn, M.J., 1972. Some Computer Organizations and Their Effectiveness. *IEEE Transactions on Computers*, C-21(9).

George et al., 2011. *Distributed Systems: Concepts and Design* 5th Editio., Boston: Addison-Wesley.

Jiang, X., Kirubarajan, T. & Zeng, W.-J., 2013. Robust Sparse Channel Estimation and Equalization in Impulsive Noise using Linear Programming. *Signal Processing*, 93(5), pp.1095–1105. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0165168412004392 [Accessed December 16, 2013].

Jordanov, V.T. et al., 1994. Digital techniques for real-time pulse shaping in radiation measurements. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 353(1-3), pp.261–264. Available at: http://www.sciencedirect.com/science/article/pii/0168900294916527.

Julin, J., 2011. *Development of a High Energy Resolution Gas Ionization Detector for a Recoil Spectrometer*. University of Jyväskylä.

Kim, H.J. et al., 2009. Development of Multifunctional Pulse Processing Device for Sensor Signal Acquisition. *IEEE Transactions on Nuclear Science*, 56(3), pp.1184–1187. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5075994.

Knoll, G.F., 2010. *Radiation Detection and Measurement* 4th ed., Wiley, John & Sons.

Lee, P., Lee, C. & Lee, J., 2012. Development of FPGA-based Digital Signal Processing System for Radiation Spectroscopy. *Radiation Measurements*, 48, pp.12–17. Available at: http://dx.doi.org/10.1016/j.radmeas.2012.11.018 [Accessed December 19, 2013].

Liu, G. et al., 2013. A Comparison of Different Discrimination Parameters for the DFT-Based PSD Method in Fast Scintillators. *Radiation Measurements*, 58, pp.12–17. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1350448713003077 [Accessed May 6, 2014].

Liu, G. et al., 2010. A digital method for the discrimination of neutrons and gamma rays with organic scintillation detectors using frequency gradient analysis. *IEEE Transactions on Nuclear Science*, 57(3 PART 3), pp.1682–1691.

Lynch, F.J., 1968. New Liquid Scintillators with Higher Speed and Efficiency. *IEEE Transactions on Nuclear Science*, 15(3), pp.102–106.

Lyoussi, A., 2010. *Détection de rayonnements et instrumentation nucléaire*, EDP Sciences.

Mihailescu, L.C., Borcea, C. & Plompen, A.J.M., 2007. Data Acquisition With a Fast Digitizer for Large Volume HPGe Detectors. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 578(1), pp.298–305. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0168900207009825 [Accessed January 31, 2014].

Moline, Y. et al., 2015. Auto-Adaptive Trigger and Pulse Extraction for Digital Processing in Nuclear Instrumentation. *IEEE Transactions on Nuclear Science*, 62(2), pp.480–486. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7072573.

Moline, Y. et al., 2014. Procédé et système d'extraction dynamique d'impulsions dans un signal temporel bruité. *Patent number FR14 50568*.

Morsy, A.A. & Von Ramm, O.T., 1999. FLASH Correlation: a New Method for 3-D Ultrasound Tissue Motion Tracking and Blood Velocity Estimation. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 46(3), pp.728–36. Available at: http://www.ncbi.nlm.nih.gov/pubmed/18238473.

Moszynski, M. et al., 2008. Energy Resolution of Scintillation Detectors—New Observations. *Nuclear Science, IEEE Transactions on*, 55(3), pp.1062–1068. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4545171.

Moszynski, M., Bizard, G. & Costa, G., 1992. Study of n-γ Discrimination by Digital Charge Comparison Method for a Large Volume Liquid Scintillator. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 317(1–2), pp.262–272. Available at: http://www.sciencedirect.com/science/article/pii/016890029290617D [Accessed January 21, 2014].

Müller, W., Rosenstiel, W. & Ruf, J., 2003. *SystemC: Methodologies and Applications*, Springer Science & Business Media.

Normand, S. et al., 2009. PING : A New Approach For Nuclear Fuel Cycle Instrumentation. *1st International Conference on Advancements in Nuclear Instrumentation, Measurement Methods and their Applications*, pp.1–4. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5503789.

Normand, S. et al., 2012. PING for Nuclear Measurements: First Results. *Nuclear Science, IEEE Transactions on*, 59(4), pp.1232–1236. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6196192.

Opsero, 2014. Ethernet FMC. Available at: http://ethernetfmc.com/.

Pasquali, G. et al., 2007. A DSP Equipped Digitizer for Online Analysis of Nuclear Detector Signals. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 570(1), pp.126–132. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0168900206017669 [Accessed December 16, 2013].

Pastor, D. & Socheleau, F.-X., 2012. Robust Estimation of Noise Standard Deviation in Presence of Signals With Unknown Distributions and Occurrences. *IEEE Transactions on Signal Processing*, 60(4), pp.1545–1555. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6132435.

Pinedo, M.L., 2008. *Scheduling: Theory, Algorithms, and Systems* 3rd ed., Springer Publishing Company, Incorporated.

Radeka, V., 1974. Signal, Noise and Resolution in Position-Sensitive Detectors. *Nuclear Science, IEEE Transactions on*, 21(1).

Radeka, V., 1972. Trapezoidal Filtering of Signals From Large Germanium Detectors at High Rates. *Nuclear Instruments and Methods*, 99, pp.525–539. Available at: http://www.sciencedirect.com/science/article/pii/0029554X72906660 [Accessed March 13, 2014].

Roca, A. et al., 2012. Enabling high-performance crossbars through a floorplan-aware design. In *Proceedings of the International Conference on Parallel Processing*. pp. 269–278.

Schiffer, R.T. et al., 2011. A Scalable FPGA-based Digitizing Platform for Radiation Data Acquisition. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 652(1), pp.491–493. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0168900210022783 [Accessed November 27, 2013].

Siddavatam, A.P.I., 2014. Methods of Pulse Shape Discrimination (PSD). In *International Technological Conference (I-TechCON)*. Vivekanand Education Sociey's Institute of Technology (VESIT): International Journal of Application or Innovation in Engineering and Management (IJAIEM), pp. 281–285. Available at: http://www.vesit.edu/TechCON-14/I-TechCON/Track3/315.pdf [Accessed March 21, 2014].

Simoes, P., Martins, J.C. & Correia, C.M.B.A., 1996. A new digital signal processing technique for applications in nuclear spectroscopy. *IEEE Transactions on Nuclear Science*, 43(3), pp.1804–1809. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=507226.

Texas Instruments, 2015. ADS54J60 Dual-Channel, 16-Bit, 1.0-GSPS Analog-to-Digital Converter. *SBAS706A*.

Texas Instruments, 2008. Op Amp Noise Theory and Applications SLOA082.

Thevenin, M. et al., 2014. Digital Real-Time Multiple Channel Multiple Mode Neutron Flux Estimation on FPGA-based Device. In *Fourteenth International Symposium on Reactor Dosimetry*. Aix-en-Provence.

Trigano, T. & Dautremer, T., 2005. Pile-up Correction Algorithms for Nuclear Spectrometry. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on*. pp. 441–444. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1416040 [Accessed January 21, 2014].

Ventroux, N. et al., 2010. SESAM: An MPSoC simulation environment for dynamic application processing. In *Proceedings - 10th IEEE International Conference on Computer and Information Technology, CIT-2010, 7th IEEE International Conference on Embedded Software and Systems, ICESS-2010, ScalCom-2010*. pp. 1880–1886.

Voltz, R. et al., 1966. Influence of the Nature of Ionizing Particles on the Specific Luminescence of Organic Scintillators. *The Journal of Chemical Physics*, 45(9), pp.3306 – 3311. Available at: http://link.aip.org/link/?JCPSA6/45/3306/1 [Accessed November 27, 2013].

Voltz, R. & Laustriat, G., 1969. Radioluminescence des milieux organiques. *Actions chimiques et Biologiques des radiations*, 29, pp.159 – 166.

Wang, X.W.X., Ahonen, T. & Nurmi, J., 2004. A synthesizable RTL design of asynchronous FIFO. *2004 International Symposium on System-on-Chip, 2004. Proceedings.*

Warburton, W. et al., 2000a. Digital Pulse Processing: New Possibilities in Nuclear Spectroscopy. *Appl. Radiat. Isot.*, 53(4-5), pp.913–920. Available at: http://www.ncbi.nlm.nih.gov/pubmed/11003541.

Warburton, W. et al., 2000b. Digital Pulse Processing: New Possibilities in Nuclear Spectroscopy. *Applied radiation and isotopes : including data, instrumentation and methods for use in agriculture, industry and medicine*, 53(4-5), pp.913–20. Available at: http://www.ncbi.nlm.nih.gov/pubmed/11003541.

Whitford, C.H., 2005. Implementation of Optimal Filters for Pulse Height Measurement from Non-linear Detectors with Non-stationary Noise. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 555(1-2), pp.255–259. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0168900205018048 [Accessed December 16, 2013].

Xiaohui, C. et al., 2013. Analysis of Three Digital n/γ Discrimination Algorithms for Liquid Scintillation Neutron Spectrometry. *Radiat. Meas.*, 49, pp.13–18. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1350448712003599.

Yousefi, S., Lucchese, L. & Aspinall, M.D., 2008. A novel wavelet-based method for neutron/gamma discrimination in liquid scintillators. In *IEEE Nuclear Science Symposium Conference Record*. pp. 2387–2391. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4774836.

Zaitseva, N. et al., 2011. Pulse shape discrimination in impure and mixed single-crystal organic scintillators. *IEEE Transactions on Nuclear Science*, 58(6 PART 2), pp.3411–3420.

Zhang, Y. et al., 2011. Asynchronous FIFO implementation using FPGA. In *ICEOE 2011 - 2011 International Conference on Electronics and Optoelectronics, Proceedings*.

Zhou, J.-B. et al., 2015. Trapezoidal pulse shaping for pile-up pulse identification in X-ray spectrometry. *Chinese Physics C*, 39(6), p.068201. Available at: http://stacks.iop.org/1674-1137/39/i=6/a=068201?key=crossref.574e082367154d9034dd22cee4ddb578.

# Une architecture programmable de traitement des impulsions zéro-temps mort pour l'instrumentation nucléaire


## Résumé

Le domaine de l'instrumentation nucléaire couvre un large spectre d'applications qui évoluent de façon permanente grâce à l'avancée des technologies du numérique et du traitement du signal. Malgré des exigences communes, des chaînes de mesure dédiées sont généralement mises au point pour répondre aux problèmes spécifiques d'un marché de niche. Cette tâche est complexe et nécessite une expertise dans différents domaines tels que la physique, la chimie, les statistiques, la micro-électronique, l'informatique, etc. La mesure de radioactivité se caractérise par l'analyse d'un signal spécifique constitué d'impulsions d'amplitudes et de durées aléatoires dont les occurrences sont issues d'un processus poissonnien homogène. Ce signal influence la conception des architectures actuelles qui s'oriente vers des solutions de traitements en flux de données sous contrainte de temps mort, ce qui ne permet pas l'utilisation de composants programmables du marché.

Ce manuscrit propose un modèle d'exécution qui permet l'utilisation de composants programmables pour les chaînes d'instrumentation. L'architecture qui en découle doit pouvoir s'adapter à l'ensemble des applications traditionnelles du domaine. Pour cela, il convient de prendre en compte les contraintes liées à l'instrumentation nucléaire : temps réel, multivoie, gestion du temps mort et programmabilité. Cette architecture est composée d'extracteurs d'impulsions placés en sortie des convertisseurs analogiques numériques. Ils sont capables d'extraire dynamiquement les impulsions en fonction de leurs tailles pour n'importe quel type de détecteur délivrant des impulsions. Ces impulsions sont ensuite distribuées sur un ensemble d'unités fonctionnelles (FUs) programmables indépendantes pilotées par les impulsions. Ces FUs sont capables de gérer l'arrivée d'évènements non déterministes et des durées d'exécution de programme variables et indéterminées à l'avance. Dans un premier temps, le modèle d'exécution est analysé analytiquement. Ce travail a ensuite permis de proposer une architecture qui a été modélisée et validée en SystemC au cycle d'horloge près. Les résultats obtenus sont prometteurs en termes de passage à l'échelle tout en maintenant le zéro-temps mort quelle que soit la durée d'exécution du programme. Cela valide le modèle d'exécution et permet de proposer une première implémentation de l'architecture. Ces travaux ouvrent la voie à des applications innovantes de traitement numérique des impulsions jusqu'alors exécutées hors-ligne.

Ces travaux de thèse ont fait l'objet de deux brevets, d'un article de journal et de quatre conférences dont une à l'international.



## Mots clefs

Instrumentation nucléaire, traitement numérique des impulsions (DPP), traitement numérique du signal (TNS), architecture électronique distribuée.

# A Programmable Zero Dead Time Digital Pulse Processing Architecture for Nuclear Instrumentation


## Abstract

The field of nuclear instrumentation covers a wide range of applications. These applications are constantly evolving thanks to the advances in the domain of the digital signal processing. Up to now, instrumentation chains are full-custom and application-specific designs which require expertise in different fields such as physics, chemistry, statistics, microelectronics, computers engineering, etc. The electric signal obtained by the radioactivity measurement made of random amplitude and durations pulse. Their occurrences follow the uniformly distributed Poisson process. This signal requires current architectures to work in dataflow mode (sample per sample) to avoid dead time. This mode makes it impossible the use of programmable component off the shelf.

The work presented in this manuscript is an execution model that enables the uses of programmable component in nuclear instrumentation chains. The architecture derived from the execution must be flexible enough to execute the traditional applications of nuclear instrumentation and fit with specific constraints: real-time, multi-channel, dead time management and programmability. The proposed architecture is composed of pulse extractors placed at the output of the Analog Digital Convertors (ADC). They are capable of dynamically extracting the pulses according to their size for any type of detector that delivering pulses. Then, pulses are distributed on a set of programmable and independent Functional Units (FU). These FUs are able to handle the arrival of non-deterministic events and variable program execution times and indeterminate in advance thanks to proposed « pulse driven » execution model. The propositions are first analyzed analytically. The proposed architecture is then modeled and validated in a cycle-accurate SystemC. The results are promising in terms of scalability while maintaining zero dead time whatever the program duration. This validates execution model and allows us to offer a first implementation of the architecture. This work paves the way for innovative applications of Digital Pulse Processing (DPP) which were previously performed offline.

This work led to the publication of two patents, a journal article one international conference article and three national communications.



## Keywords

Nuclear instrumentation, Digital Pulse Processing (DPP), Digital Signal Processing (DSP), Distributed computing.