aux modulations QAM, on peut utiliser des modulations QAM d'une constellation de grande taille.

- les techniques de multiplexage: plusieurs techniques cherchent à optimiser la bande passante, nous citons en particulier, les techniques basées sur l'utilisation conjointe de plusieurs porteuses, et celles basées sur l'étalement du spectre. Dans le premier cas, les données sont transmises sur plusieurs porteuses orthogonales: *OFDM* (*Orthogonal Frequency Division Multiplexing*). L'OFDM avec cette orthogonalité peut maximiser l'utilisation du spectre (nombre élevé de bit par Hertz). Dans le second cas, on utilise des codes d'étalement pour étaler le signal sur une large bande passante. Ce signal étalé devient sous le niveau de bruit. Cette technique est utilisée dans les communications 3G, qui l'implémentent au travers du *WCDMA* (*Wideband Code Division Multiple Access*).

- au niveau du récepteur, les fonctions de démodulation et de décodage sont les fonctions inverses respectives des fonctions de modulation et de codage situées du côté émetteur.

5. Les plateformes de la radio reconfigurable

Nous étudions dans ce paragraphe les circuits utilisés en bande de base pour le SDR: les circuits spécialisés de type Application-Specific Integrated Circuit (ASIC) et dérivés, les processeurs de type General Purpose Preprocessor (GPP), les FPGA et les DSP (Digital Signal Processor).

5.1. Composants matériels

Les composants matériels sont utilisés pour leurs performances très élevées. Ils permettent de résoudre de fortes contraintes temps réel. On peut distinguer deux types: les ASIC et les FPGA. Les ASIC sont des composants spécialisés développés pour une application spécifique. Les FPGA, sont des circuits intégrés qui peuvent être reconfigurés après sa fabrication.

5.1.1. Les ASIC (Application-Specific Integrated Circuit)

Les ASIC (circuit intégré propre à une application, en français), ce sont des circuits intégrés spécialisés, que sont développés pour un seul client et selon les spécifications de ce dernier. En général, un ASIC regroupe un grand nombre de fonctionnalités uniques ou sur mesure. Les ASIC ne sont pas reconfigurables (reprogrammables), c'est-à-dire ils sont figés (fixés) pour un nombre de fonctionnalités spécifiques. En raison de leurs caractéristiques spéciales, ils sont utilisés couramment dans la SDR, bien qu'ils soient des composants non modifiables contredisant ainsi le principe de base des SDR ; c'est-à-dire la reconfigurabilité et la flexibilité. En effet, les ASIC, présentent une puissance de calcul très élevée, une faible consommation d'énergie, une haute intégration et une petite taille. Ils montrent aussi un coût de fabrication faible pour des quantités élevées. C'est pour ces

raisons que les portables représentent un domaine d'application typique de ces composants.

Le manque de la flexibilité et le cycle de développement très long des ASIC constituent leur principal inconvénient [47]. À noter qu'il existe également des ASSP (*Application Specific Standard Product*), circuits type ASIC légèrement programmables, regroupant un grand nombre de fonctionnalités pour satisfaire à une application généralement standardisée. En général, les ASIC et les ASSP sont basés sur les mêmes processus de conception et technologies de fabrication. La différence est que les ASIC sont destinés à une compagnie spécifique, tandis que les ASSP sont vendus à de multiples clients. Un ASSP pour GSM issu d'un fabricant unique est utilisé comme circuit de base par différents fabricants de téléphones portables qui se distinguent sur d'autres aspects tels que le logiciel, l'écran, le boîtier, la batterie, etc. La programmation d'un ASIC s'effectue avec un langage de bas niveau, et les tests de validation sont intensifs. Une erreur qui apparait après sa production peut provoquer la nécessité de repasser par la phase de développement et coûte par conséquent très cher [47].

5.1.2. Les FPGA (Field-Programmable Gate Array)

Les FPGA (Les réseaux de portes programmables *in situ*, en français), est un ensemble de blocs logiques élémentaires que l'utilisateur peut interconnecter pour réaliser les fonctions logiques de son choix. Un FPGA est composés d'un nombre élevé de blocs logiques configurables ou CLB (*Configurable Logic Block*). Un CLB est constitué au minimum d'une table de correspondance (LUT – *Look-Up Table*) et d'une bascule. Une LUT est considérée comme la partie fondamentale du FPGA, elle agit en tant que générateur de fonction et sert à implémenter des équations logiques (généralement de 4 à 6 entrées et une sortie), une petite mémoire, un multiplexeur ou un registre à décalage. L'architecture d'un CLB se diffère en fonction du FPGA et de la technologie utilisée. Sur une famille de FPGA Xilinx Virtex 4 par exemple, chaque CLB dispose de deux LUT à quatre entrées (LUT4), alors que pour les Virtex 5, et 6, chaque CLB dispose de quatre LUT à six entrées (LUT6), soit la possibilité d'utiliser six variables d'entrée pour chaque fonction au lieu de quatre, et deux fois plus de fonctions combinatoires, alors que dans le Virtex 7 chaque CLB dispose de quatre LUT, qui peuvent être configurés soit comme une LUT à 6 entrées avec une sortie, ou comme deux LUT à 5 entrées avec des sorties séparées. Les CLB sont interconnectés par un système matriciel configurable de liaisons haute vitesse, et sont reliés à des blocs d'E/S programmables ou (IOB – *Input Output Blocks*). Les blocs d'entrées/sortie sont utilisés pour relier le FPGA à des signaux externes, et ainsi à des mémoires externes rapides dans le cas où la mémoire interne est insuffisante. Deux types de mémoire interne existent dans le FPGA; de petites mémoires rapides double accès (pour les tampons et les tables de coefficient), et de grandes mémoires simple accès (pour le stockage de grandes tables de données). Les blocs dédiés

dans le FPGA, permettent de gérer l'horloge, l'aident à la programmation, à réaliser facilement de nombreuses opérations tels qu'une PLL et les traitements numériques des signaux (blocs DSP). En effet, la présence des multiplicateurs, des accumulateurs et des mémoires facilitent la réalisation de la convolution, la corrélation et le filtrage. La figure II.8 présente un exemple d'architecture de FPGA.

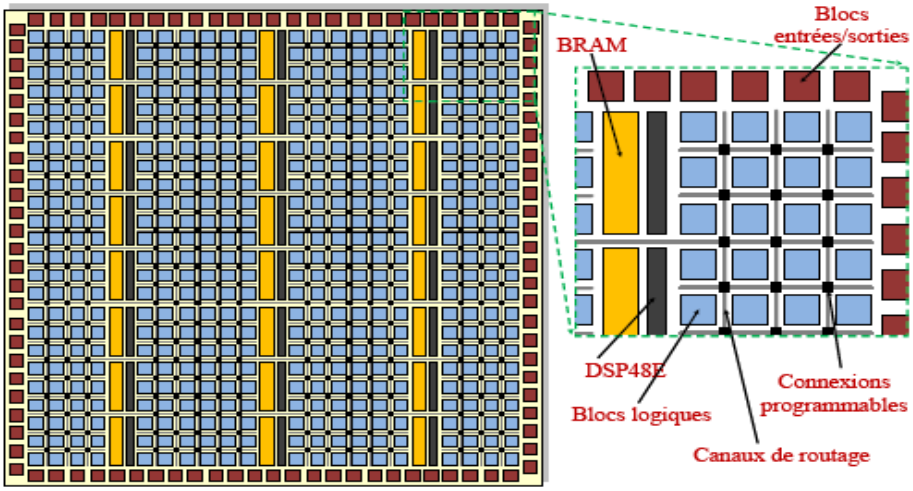

Figure II.8. Exemple d'architecture de FPGA (Architecture, Interconnexions et CLB).

L'aspect le plus séduisant des FPGA est leur flexibilité et leur reconfigurabilité, ce qui respecte pleinement le principe de base des SDR. En plus, ils présentent une consommation d'énergie faible et une puissance de calcul grande par rapport aux processeurs DSP et GPP. Ceci revient à un grand nombre d'unités de traitements et de mémoires tampons (*buffers*). Par exemple, la famille des FPGA Virtex-7 de Xilinx [48], contient entre environ 45 000 et 300 000 slices (CLB), chaque slice contient quatre LUT et huit bascules. Nous avons aussi comme exemple le FPGA Stratix V de Altera, qui est un FPGA de haute performance [49]. Cette énorme ressource permet d'implémenter plusieurs fonctions et même des standards et normes. Par exemple L. Bisdounis et al [50] ont implémenté sur un FPGA la partie bande de base et la couche MAC du standards HIPERLAN/2 et IEEE 802.11a.

La concurrence entre les principales firmes des FPGA notamment Xilinx et Altera contribuent à une amélioration permanente des performances de ces composants. Pour cette raison, la performance actuelle des FPGAs est un peu approchée de celle des ASIC. En plus, la tendance pour les circuits FPGA est la possibilité de réaliser des systèmes sur puce (SoC) en utilisant des "composants virtuels" et de concevoir ainsi des blocs de propriété intellectuelle IP qui sont par exemple des fonctions VHDL/Verilog génériques réutilisables, on parlera alors de « System On Programmable Chip» (SoPC) . Les systèmes de type SoPC peuvent intégrer sur le même FPGA un ou plusieurs processeurs

softcore ou hardcore avec ses périphériques et une mémoire interne. Cette mémoire est utilisée pour sauvegarder les données (Data) ou le code exécutable correspondant à l'application logicielle déployée sur le ou les processeurs. Parmi les processeurs embarqués intégrables dans un système SoPC, on peut citer les solutions softcore propriétaires: MicroBlaze et nios développées par Xilinx et Altera respectivement. D'autres processeurs performants (hardcore) sont issus du monde opensource tels que: processeurs Leon 2/3, plasma, ARM,...

Les FPGA se programment grâce à leurs LUT et leur réseau d'interconnexion en utilisant le plus souvent un langage de description de matériel tel que le VHDL (acronyme de VHSIC-HDL: Very High Speed Integrated Circuit Hardware Description Language) ou Verilog. L'outil de développement (ISE par exemple [51]) transforme cette description en un fichier de configuration du FPGA en plusieurs étapes (figure II.9): Lors de l'étape de synthèse, la description HDL du circuit est transformée en un assemblage (Netlist) de primitives de base (portes logiques, bascules, etc.).

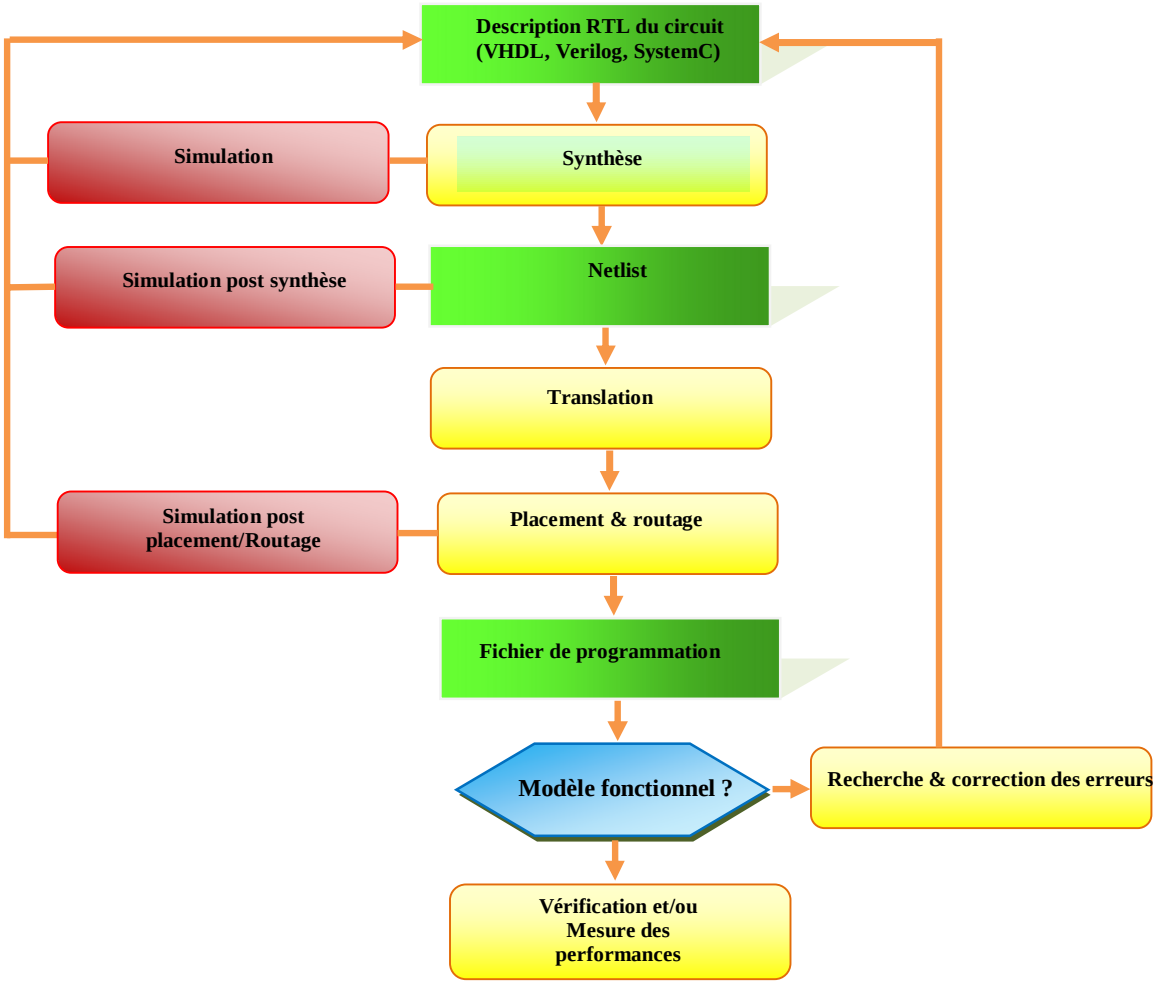

Figure II.9. Cycle de programmation d'un FPGA.

A ce point, il est possible d'effectuer une étape de simulation post-synthèse, l'intérêt de cette étape étant de vérifier les propriétés temporelles du circuit. Il vient ensuite l'étape de placement qui place les éléments de la Netlist (correspondant maintenant à des cellules) physiquement sur le circuit FPGA. Une fois l'emplacement physique de chaque cellule fixé, le routeur effectue le calcul des interconnexions entre les cellules physiques en respectant la connectivité des cellules comme indiqué dans la Netlist donnée en entrée. Une fois la phase de placement et routage terminée, le fichier de configuration (bitstream) du FPGA peut être généré. Le bitstream est une image binaire du circuit FPGA, il y détermine pour chaque élément (cellule logique, table de correspondance LUT, élément de routage, cellule mémoire) l'état logique. L'exécution proprement dite du circuit est possible après que le fichier ait été chargé sur le circuit FPGA (on parle ici de la reconfiguration du FPGA soit pour placer une nouvelle application, soit pour gérer l'évolution d'une application). Il est alors possible de vérifier les fonctionnalités du modèle et d'effectuer des mesures de performances.

Pendant l'étape de reconfiguration, un FPGA doit être dans un état inactif, puis se reconfigurer, enfin être remis en état opérationnel, bien que d'autres séries des FPGA supportent un mode de configuration partielle, ne nécessitant pas le composant dans un état inactif [52].

5.2. Composants logiciels

Ils offrent par rapport aux composants matériels une flexibilité très élevée. On peut distinguer les DSP et les GPP. La programmation s'effectue le plus souvent en langage de haut niveau, ce qui simplifie le processus de développement.

5.2.1. Les DSP

Les processeurs de traitement numérique du signal ou DSP (*Digital Signal Processor*) sont des microprocesseurs optimisés et adaptés pour exécuter des applications de traitement numérique du signal (filtrage, convolution, etc.). Ils sont classés donc comme des processeurs spécifiques. Les DSP occupent de nombreux secteurs de l'électronique, car leur architecture interne est dimensionnée pour le calcul intensif, ce qui permet d'exécuter efficacement des opérations de multiplications/accumulations (MAC), avec des vitesses de traitement plus rapides.

Les processeurs DSP ont des architectures capables de réaliser des opérations sur des nombres à virgule fixe, et à virgule flottante. En effet, les unités arithmétiques et logiques ne sont pas construites de la même manière. Le calcul flottant requiert une architecture plus complexe qui conduit à une fréquence de fonctionnement généralement plus faible. Par conséquent, les flottants sont utilisés seulement lorsqu'une grande précision et une grande dynamique sont requises sur les valeurs traitées [47]. L'architecture peut ainsi inclure des interfaces série, PCI, DMA (Direct Memory Access) et une mémoire.

À l'origine, les DSP étaient composés d'un seul cœur qui exécute des opérations MAC aussi vite que possible. Les DSP modernes peuvent utiliser plusieurs cœurs de calcul, en plus la possibilité d'embarquer des coprocesseurs dédiés (par exemple coprocesseurs de décodage Viterbi). Le DSP de Texas Instruments TMS320C6416 par exemple est un DSP à deux cœurs de calcul (A et B) qui ont chacun quatre unités (L, S, M et D). Huit instructions peuvent donc être exécutées en un cycle (voir datasheet de TMS320C6416). Le DSP d'Analog Devices « Blackfin ADSP-BF561 » est aussi un DSP à double cœur optimisé pour le traitement du signal de haute performance, il peut effectuer jusqu'à 2,4 Giga-MAC (milliards de multiplication/accumulation) par seconde avec une fréquence d'horloge maximale de 600 MHz [53] [54].

Ces DSP récents comportent trois techniques qui permettent d'augmenter leurs performances:

- l'architecture VLIW (Very Long Instruction Word) lit des instructions de 256 bits pour fournir à chaque cycle d'horloge jusqu'à huit instructions de 32 bits aux huit unités fonctionnelles;
- les unités de calcul 32 bits du TMS320C6416 par exemple, permettent d'effectuer une opération 32 bits, deux opérations 16 bits ou quatre opérations 8 bits, grâce aux instructions SIMD (Single Instruction Multiple Data). Les unités fonctionnelles peuvent donc être exploitées de manière intensive en utilisant toute la largeur disponible;
- les transferts entre les périphériques et la mémoire et également entre la mémoire interne et la mémoire externe sont réalisés par le DMA (Direct Memory Access), qui permet de décharger le cœur de calcul, et de paralléliser l'accès mémoire et les traitements.

Les DSP peuvent être programmés avec un langage de haut niveau, tel que C ou C++, et sont accompagnés d'outils de développement performants. Ces outils de développement, les compilateurs et les simulateurs, sont capables d'optimiser fortement un programme et d'informer le développeur sur le résultat en ajoutant des commentaires dans le programme (nombre de cycles par exemple). Ils permettent aussi d'obtenir des informations sur la charge de calcul, le comportement des mémoires caches, et le temps d'exécution [47]. D'après Jean-Philippe Delahaye [55], les DSP modernes de haute performance avec leurs souplesses et leurs productions en masse, constituent un choix plausible pour le SDR.

5.2.2. Les processeurs à usage général (GPP)

Les processeurs à usage général ou GPP (*General Purpose Processor*) sont les processeurs qui se trouvent dans la plupart des ordinateurs. Ils sont classés comme des processeurs généralistes [47]. Les architectures des GPP les plus répandues sont les CISC (Complex Instruction Set Computer) comme le processeur Pentium d'Intel ou AMD

Athlon, et les RISC (Reduced Instruction Set Computer), comme les Sparc, MIPS, ARM et PowerPc. Ces instructions effectuent des opérations telles que la multiplication, l'addition ou le stockage, mais ne sont toutefois pas optimisées pour un usage particulier.

Comme pour les DSP, la plupart des GPP sont accompagnés de nombreux outils de développement puissants (compilateurs, debuggers) [55].

Plusieurs avantages des GPP les placent parmi les composants plausibles pour le SDR, tels que leur grande flexibilité, leur simplicité de configuration et surtout la possibilité de réutilisation des programmes pour d'autres applications. Cependant, les inconvénients des GPP, sont décrits par le manque d'une puissance de calcul suffisante pour gérer le flux du signal radio de haut débit et l'inadaptation au temps réel pour les applications complexes [47]. Par conséquent, il est préférable d'utiliser les GPP lorsque le signal est en bas débit, c'est à dire dans les opérations finales d'un récepteur ou les opérations initiales de l'émetteur. Les GPP présentent d'autres inconvénients tels que la grande consommation, la dissipation d'énergie élevée, l'encombrement et le prix.

Toutefois, les GPP subissent une grande évolution au niveau des performances. En effet, les nouveaux processeurs disposent d'unités de traitement vectoriel (SIMD « Single Instruction Multiple Data ») haute vitesse. Ils sont ainsi capables d'optimiser l'exécution d'un programme à la volée pour améliorer les performances des applications qui ne sont pas optimisées à la base, pour un type de processeur particulier. On peut résumer les techniques utilisées par les GPP modernes comme suit:

- **Pipeline d'exécution:** elle permet de découper les opérations en plusieurs étapes afin d'augmenter la fréquence des processeurs, chaque étape étant plus courte. Le processeur peut alors contenir plusieurs instructions, chacune à une étape différente.
- **Super-pipeline:** elle permet aux nouvelles versions des GPP d'effectuer plus d'un milliard d'opérations mathématiques par seconde.
- **Super-scalaires:** plusieurs unités de calcul dédiées sont embarquées (calcul arithmétique et logique, calcul flottant, accès mémoire, branchement). Cela permet d'exécuter plusieurs instructions par cycle grâce au fonctionnement en parallèle de ces unités.
- **Unité de ré-ordonnancement des instructions et des accès mémoire:** Les dépendances de données entre les instructions séquentielles d'un programme sont analysées, puis les instructions sont parallélisées sur les différentes unités de calcul, et réordonnées si besoin. De même que pour les instructions, les accès aux données en mémoire sont réordonnés.
- **Prédiction de branchement:** Un branchement conditionnel dans un programme est très préjudiciable. En effet, la condition doit être calculée avant d'effectuer le

branchement, induisant une perte de cycles. Pour les limiter, le branchement est prédit. Une analyse statistique en temps réel permet de prédire un branchement et de garder les unités de calcul actives.

- **Mémoire cache:** La mémoire cache est utilisée afin de limiter le nombre d'accès à la mémoire centrale du système et de minimiser, par conséquent, l'utilisation du bus et de la consommation d'énergie du système.

Le microprocesseur Core i7 multi-coeur, multi-thread (multi-tache) d'Intel, est un processeur pouvant s'avérer utile dans les applications SDR.

5.2.3. L'accélération matérielle: GPU (Graphics Processing Unit).

L'accélération matérielle est un circuit intégré dédié qui effectue une fonction spécifique (initialement effectuée par le processeur CPU) de façon plus efficace. L'accélération matérielle est implémentée soit sur la carte mère, soit sur une carte fille, soit intégrée dans le CPU. Il existe différents types d'accélérateurs matériels pour différents usages, les plus reconnus pour les utilisateurs des PC, sont les cartes d'accélération audio (cartes son) et les cartes d'accélération vidéo. Les premières sont utilisées pour la numérisation et le traitement du signal audio et les deuxièmes sont utilisées pour le traitement du signal vidéo, décompression MPEG, etc. La carte d'accélération vidéo embarque un ou plusieurs processeurs graphiques, ou GPU (*Graphics Processing Unit*). Généralement, les GPU s'interfacent avec la carte mère d'un ordinateur via un port PCI-express. De plus, les GPU partagent la mémoire vive avec la carte mère, bien qu'il existe des GPU qui disposent de leur propre mémoire vive.

Le GPU se compose d'une multitude (plusieurs centaines, voir milliers) de processeurs indépendants capables d'exécuter simultanément plusieurs calculs (figure II.10).

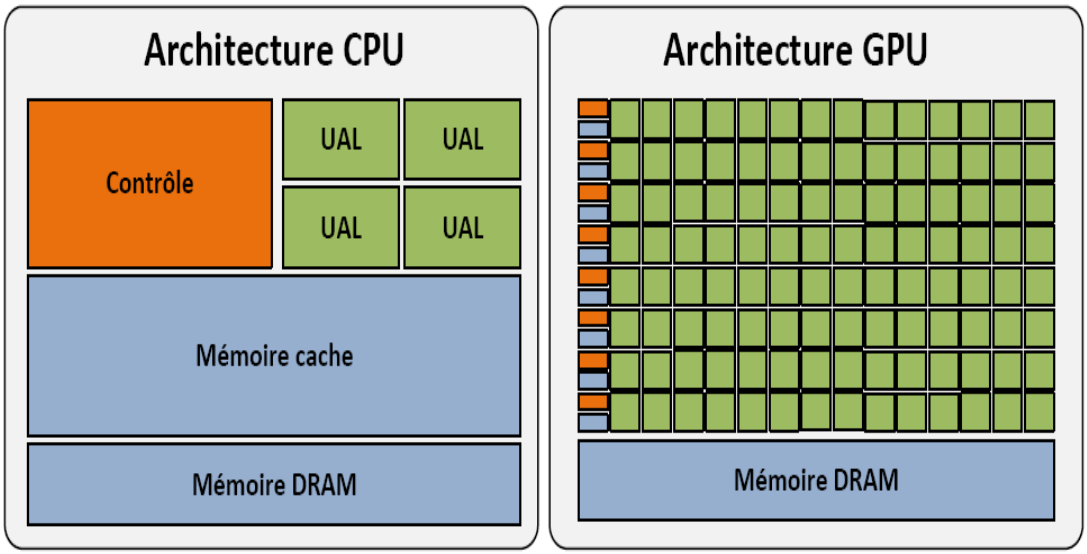

FIGURE II.10. Architectures CPU et GPU.

C’est pour cela que les GPU sont parfaitement adaptés aux traitements parallèles avec une performance très supérieure par rapport aux processeurs GPP. Un GPU offre une puissance de calcul de six fois supérieure à un coût relativement réduit que celle des GPP [47]. Cela les rend très utilisables à la radio logicielle restreinte [56].

Il a été montré que l’utilisation d’un GPU peut améliorer les performances et l’exécution dans une application SDR et ceci beaucoup mieux que l’utilisation d’un CPU [56] [57]. De même, J. KIM et al [58] réalisent un terminal SDR pour le WiMAX à l'aide d'un ordinateur équipé à la fois d'un PC et d'un GPU. En comparant les performances d’une carte graphique GeForce 9800GTX de Nvidia et d’un DSP TMS320C6416 de Texas Instruments, ils ont trouvé que le décodage d'un code convolutif avec l'algorithme de Viterbi par la carte graphique est 90 fois plus rapide que le DSP. Ainsi, le débit de décodage de Viterbi implémenté sur le GPU est de 181.6 Mb/s comparé à 3.07 Mb/s pour le DSP [58]. Il faut noter que Nvidia fournit également un kit de développement basé sur le langage C (CUDA: Compute Unified Device Architecture) qui permet le débogage et la simulation sur CPU.

5.3. Analyse des plateformes de la radio reconfigurable.

La radio flexible est clairement dépendante de la capacité d’un système à modifier son mode de fonctionnement. Les composants logiciels présentent plus de flexibilité par rapport aux composants matériels, il semble donc que les cibles logicielles sont les plus adaptés pour le SDR. En effet, un processeur comme les General Purpose Processor (GPP), les GPU et les DSP (Digital Signal Processor) sont virtuellement capables d’exécuter n’importe quelle opération, par changement du programme qui la décrit. Cependant, il est difficile d’ignorer les aspects de performances et de consommation dans ces composants. Dans les systèmes embarqués, cette contrainte de consommation est, et restera forte, en effet elle est supérieure à celle d’un ASIC et FPGA, surtout pour les traitements intensifs de données. En plus, les performances de ces processeurs sont souvent insuffisantes face aux besoins de traitement du signal des applications récentes. Bien qu’il existe des processeurs spécialisés de traitement du signal comme les DSP, ils ne peuvent pas prendre en charge l’ensemble des traitements d’une chaîne radio [55].

Dans une application réelle de la radio logicielle avec les cibles logicielles telles que les GPP, GPU et DSP, la modification d’une application est faite par un simple téléchargement d’un nouveau programme (code) applicatif. Cependant, cela pose le problème de la limitation de mémoire disponible pour le stockage des données et des fichiers de configuration.

D'autres problèmes résident dans le risque sur l'intégrité de l’équipement lors de la phase de téléchargement et sont multiples et bien connus du monde de l’informatique tels que:

• bogues logiciels,

• transmission de virus informatiques par téléchargement,
• failles logicielles.

Par conséquent, il n'est pas toujours nécessaire d'utiliser une cible logicielle afin d'avoir la flexibilité pour le SDR. Les cibles matérielles comme les ASIC sont des implantations figées et spécifiques à l'application développée, qui montrent des performances extraordinaires avec une consommation réduite. Cependant, leur point faible est le manque de la flexibilité, qui est une caractéristique indispensable dans la définition de la radio reconfigurable. Pour réaliser un compromis entre la flexibilité et la performance, il existe des solutions reconfigurables comme les circuits de type FPGA. Les FPGA montrent une puissance de calcul élevée par rapport aux DSP ou aux GPP, cette performance résulte du fait que les FPGA autorisent des traitements quasiment en parallèle, c'est à dire qu'après un ou quelques cycles d'horloge, leur valeur de sortie est disponible en fonction de la valeur d'entrée. En revanche, les DSP et les GPP travaillent essentiellement en mode série. Ainsi les FPGA sont flexibles par rapport aux ASIC et les performances des FPGA modernes ont presque atteint le niveau de performance des ASIC. Ils peuvent donc remplir les compromis flexibilité/performance et constituer un choix indispensable pour le SDR. Il existe d'autres techniques et des solutions prometteuses pour avoir la flexibilité et la performance attendue par la radio logicielle (SDR) que nous verrons dans les sections suivantes.

6. Solutions hybrides (hétérogènes)

Comme toute évolution technologique, son utilisation (GPP, DSP/GPU, ASIC, FPGA) présente des avantages et des inconvénients. L'une des solutions envisagées pour compenser les inconvénients est l'utilisation de la plateforme hybride. La radio logicielle flexible nécessite l'utilisation des processeurs généralistes pour obtenir une flexibilité maximale du système qui nécessite alors un coût élevé en consommation. Dans les systèmes embarqués, cette contrainte de consommation est forte. Or les ASIC, sur les quels sont basés les systèmes actuels, souffrent d'un manque de flexibilité à cause de sa paramétrisation réduite. Les systèmes à base de FPGA offrent des performances intéressantes en plus de la flexibilité. C'est pour cette raison que les systèmes de transmissions dédiés aux applications de la radio logicielle possèdent généralement une architecture hétérogène, car ces plateformes sont constituées de divers éléments ayant beaucoup d'avantages complémentaires (figure II.11).

Il existe quelques travaux sur les plateformes hétérogènes, nous citons à titre d'exemple la plateforme Kansas University Agile Radio (KUAR) [59] qui est une plateforme radio logicielle disposant d'un processeur Pentium M de fréquence de 1.4 GHz avec une mémoire DDR2 SDRAM de 1GB et un FPGA Virtex2 de Xilinx [59].

L'Institut national japonais de la technologie de l'information et des communications (NTIC) a construit une plate-forme de radio logicielle restreinte (NTIC plateforme SDR) pour les réseaux mobiles de prochaines générations. La plate-forme dispose de deux processeurs embarqués, quatre FPGA Virtex 2 de Xilinx avec module RF qui pourrait soutenir 1,9 GHz à 2,4 GHz et de 5,0 GHz à 5,3 GHz. Le traitement du signal a été partagé entre le CPU et le FPGA. À cette fin, un certain nombre de normes commerciales ont été mises en œuvre dans cette plate-forme, par exemple 802.11a/b/g, la radiodiffusion numérique terrestre, le WCDMA et le système de communication OFDM.

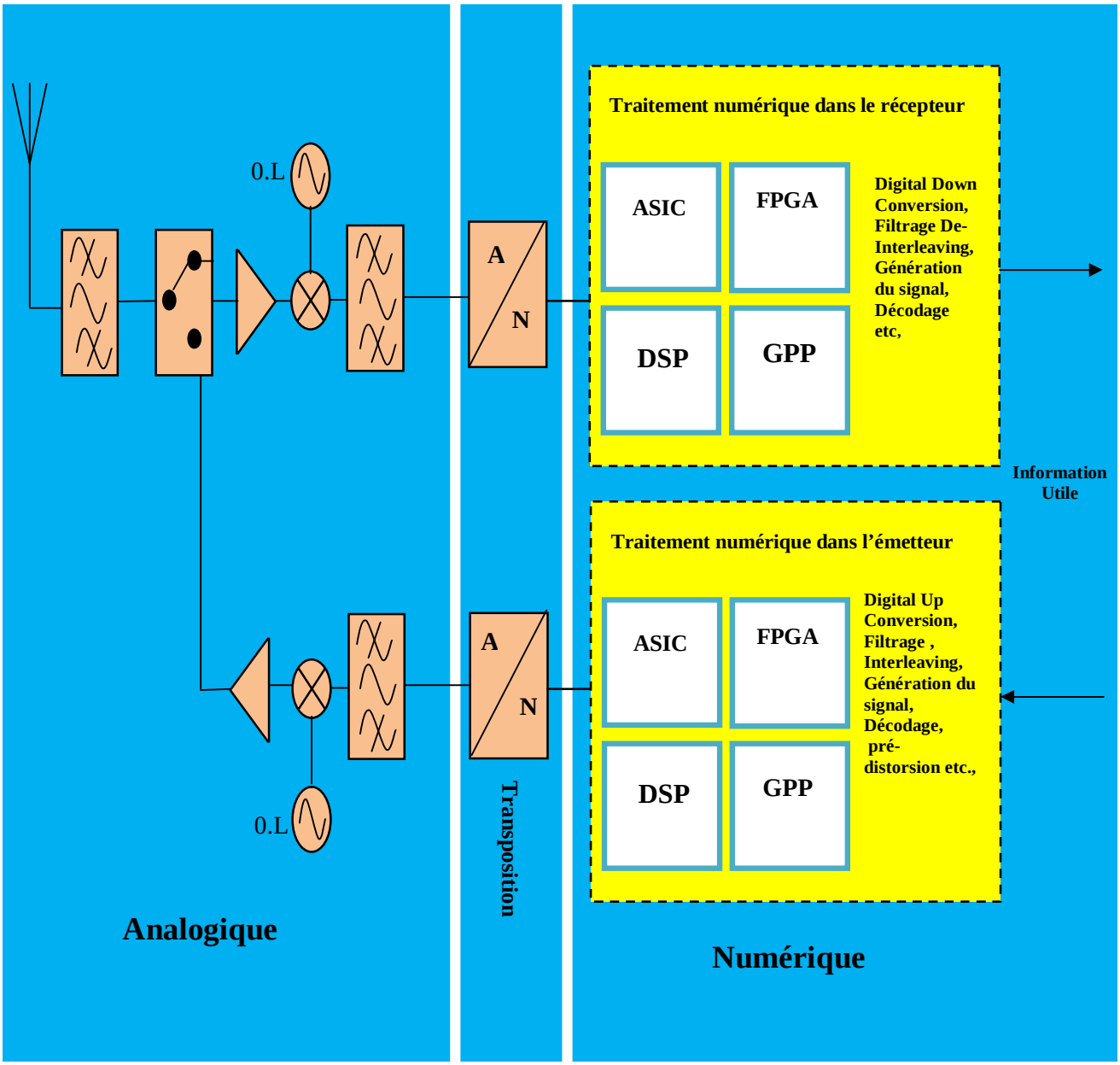

Figure. II.11. Structure possible d'une radio logicielle.

D’autres plateformes sont à base d’un FPGA et d’un DSP où le FPGA assure la conversion entre le haut débit des CAN/CNA et un débit plus faible, permettant ainsi au DSP de réaliser l’intégralité des traitements.

Le problème des plateformes hétérogènes réside dans la difficulté à les utiliser (difficulté pour gérer cette intégration). En effet, l’intégration de processeurs ayant chacun leur propre interface d’accès à la mémoire, leur propre jeu d’instruction, leur propre ordre dans lequel les octets sont organisés en mémoire ou dans une communication ("boutisme" ou endianness), leur vitesses de traitement, leur méthode d’acheminement des données dans l’architecture, présente de grandes difficultés et même un défi. Certaines approches utilisent un GPP pour gérer cette intégration, où le GPP joue le rôle d’un contrôleur qui donne les ordres et les données aux autres processeurs. De plus, il n’existe pas un environnement de travail unique permettant de concevoir de tels systèmes, donc malheureusement, le concepteur doit recourir à de nombreux logiciels et langages de programmation lui permettant de programmer les différents composants.

7. L’approche de paramétrisation

Cette approche est beaucoup plus générale, pourtant c’est une technique plus efficace pour l’implantation des terminaux multistandards [60]. Cette approche vise à trouver des points ou caractéristiques communs entre les traitements de différents standards dans un premier temps, puis d’en proposer des architectures communes (génériques) et flexibles. Ces architectures de traitements communs restent figées après la conception, mais elles peuvent prendre en charge un ensemble de fonctionnalités grâce à un simple jeu de paramètres, d’où le nom de paramétrisation [5] [60]. L’opposé d’une approche de type Velcro permet de diminuer le nombre d’éléments à implémenter et constitue une méthode de reconfiguration en elle-même [6] [7] [61]. Selon la méthodologie de paramétrisation, les aspects communs des différents standards deviennent un élément de traitement commun qui pourrait être installé dans le dispositif et exécuté par un simple appel.

La paramétrisation se décline selon deux approches: l'approche théorique et l'approche pragmatique. **L'approche théorique** consiste à lister de façon hiérarchique tous les appels de fonctions possibles dans un terminal (Par exemple, la modulation OFDM fait appel à la FFT, qui elle même fait appel à l'opérateur papillon et ce dernier faisant appel aux opérateurs arithmétiques classiques). La démodulation WCDMA comme un autre exemple, fait appel à un récepteur Rake, qui lui même fait appel à l’unité de désétalement et cet dernier faisant appel aux opérateurs arithmétiques classiques. Ensuite, l’ensemble des fonctions réalisées par un terminal multi-standards est représenté par un diagramme dont chaque sommet représente un élément de traitement (Processing Element- PE) fonctionnel qui occupe un niveau de granularité donné. Cette représentation en graphe permet de sélectionner un chemin optimal selon un processus d’optimisation basé sur des

coûts (consommation, surface, délais…). Ce chemin optimal permettant de privilégier l'appel à certains opérateurs, qui deviennent alors communs. Le processus d'optimisation est basé sur une fonction de coût/performance, qui calcule le choix de chaque jeu d'opérateurs. Cette fonction de coût lorsqu'elle est optimisée permettra de sélectionner le jeu d'opérateurs ayant le coût minimal. La fonction de coût qui a été proposée par S.T. Gul [62] prend comme paramètre: le coût de fabrication (Building Cost, BC), le coût de calcul (Computational Cost, CC), et le nombre d'appels (Number of Calls, NoC). En général, par cette approche on peut définir de nouveaux opérateurs à partir d'une combinaison d'opérateurs déjà présents dans le graphe considéré. Par conséquent, cette approche nécessite une deuxième approche qui permet de définir les opérateurs communs. Cette approche dite **l'approche pragmatique**, que nous privilégiions dans cette thèse. Elle consiste dans un premier temps à identifier dans la littérature les traitements similaires qui peuvent partager des ressources communes (tant au niveau algorithmique qu'architectural), puis dans un second temps à réaliser un opérateur générique qui devra alors être reconfigurable par changement de paramètres. C'est pour cela que cette approche arbore un côté plus pratique dans l'application de la technique des opérateurs communs (section 7.2) [60][62]. Mais il faut noter que les deux approches convergent vers le même objectif i.e l'identification des meilleurs points communs entre les différentes normes, ce qui conduit à une conception multi-standard reconfigurable flexible.

Deux approches peuvent être distinguées en paramétrisation: l'approche fonction commune et l'approche opérateur commun.

7.1. Fonctions communes (FC)

Le niveau de ce type de paramétrisation est de granularité élevé, puisque la paramétrisation se fait au niveau fonction. Ici, on cherche à définir des fonctions paramétrables communes à plusieurs standards, où tous les composants dédiés à la même fonctionnalité ont été regroupées dans le même FC. Par exemple, on dédie à la fonction commune « codage canal », les différents types de « codage canal » utilisés pour les différents standards. Une liste de paramètres est associée à la fonction et détermine son mode de fonctionnement.

On peut prendre l'exemple d'Anne Wiesler et F. Jondral [6] [7] [63], qui sont considérés comme pionniers dans ce domaine. En effet, dans leurs travaux, ils ont proposé des structures paramétrables pour les fonctions de modulation et d'égalisation communes aux standards GSM, DECT (Digital Enhanced Cordless Telecommunications), UMTS UTRA/FDD et IS-136. Cet exemple de fonction commune est illustré dans la figure II.12. A.-R. Rhiemeier [64] qui est dans la même équipe de recherche, a proposé une fonction

commune de codage canal. Nous avons aussi l'architecture nommée VITURBO, qui est une structure unifiée de turbo décodage et de Viterbi pour les systèmes 3G [65].

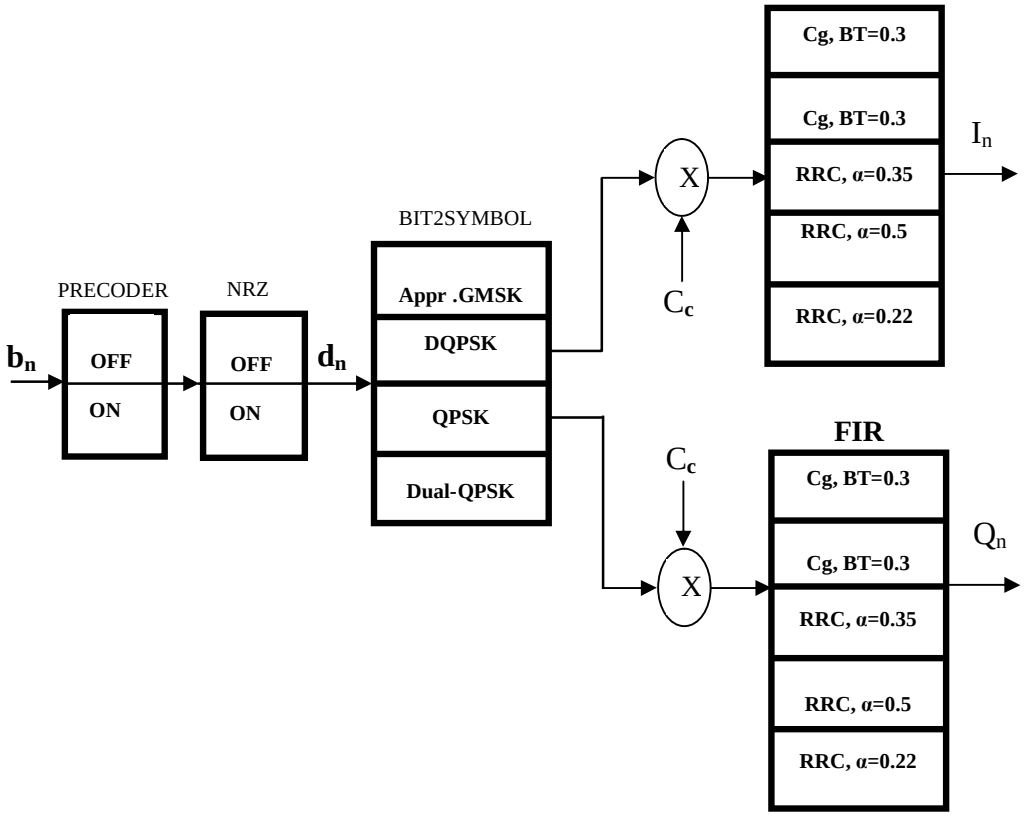

Figure. II.12. Fonction de modulation paramétrable [7].

Jean-Philippe Delahaye a proposé sur la base de cette technique une chaîne de transmission multistandards pour le GSM, UTRA/FDD et 802.11g [55]. Il a étudié chaque chaîne de traitement des 3 standards, puis il a remarqué que ces chaînes présentent un certain nombre de fonctions similaires appelées fonctions multi-standards, et fonctions spécifiques à chaque standard (figure II.13). Enfin, il a proposé d'ajouter les parties spécifiques à la partie fonctions multi-standards afin d'obtenir une chaîne multi-standards unifiée [55] (figure II.14).

L'approche « fonction commune » présente l'avantage d'un temps de changement de modes négligeable, et ainsi par un nombre limité de paramètres, on peut décrire plusieurs modes de fonctionnement.

Cependant, cette approche souffre d'un problème d'évolution et d'un problème d'efficacité énergétique. En effet, tous les exemples présentés ci-dessus, nous permettent de conclure que cette technique montre une forte dépendance des normes choisies, avec un faible partage de ressources. La technique « fonction commune » (FC) consiste en

l'agrégation des composants nécessaires dans une seule fonction, ce qui signifie que l'ajout d'une nouvelle norme implique l'ajout des composants distincts de chaque fonction dans les FC associées. Donc cette structure ne peut pas s'adapter facilement aux nouveaux besoins des standards [62][64]. Selon L. Alaus et al [67] cette technique est considérée comme une technique fixe.

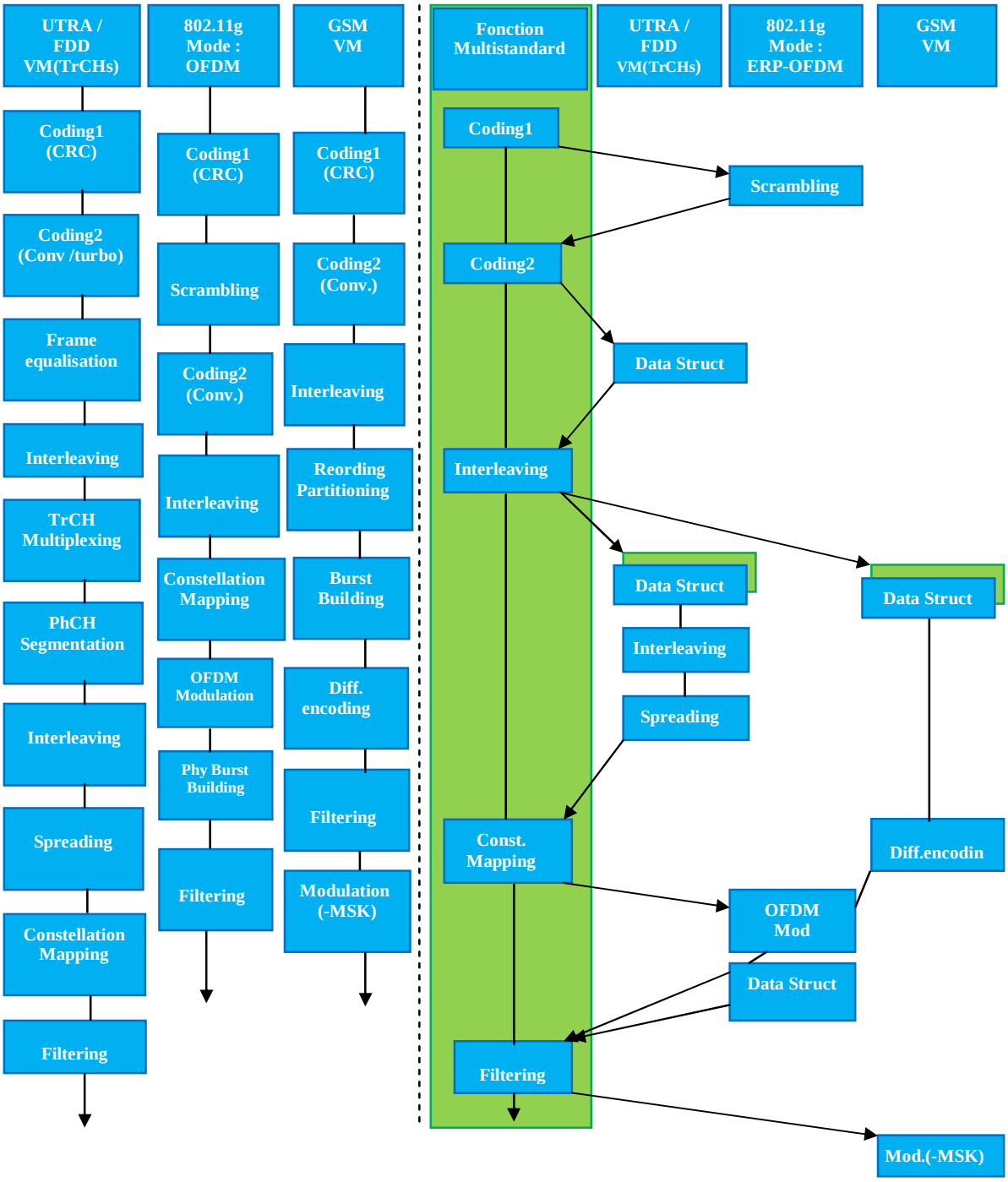

Figure. II.13. Vers une chaîne de transmission multi-standard unifiée [55].

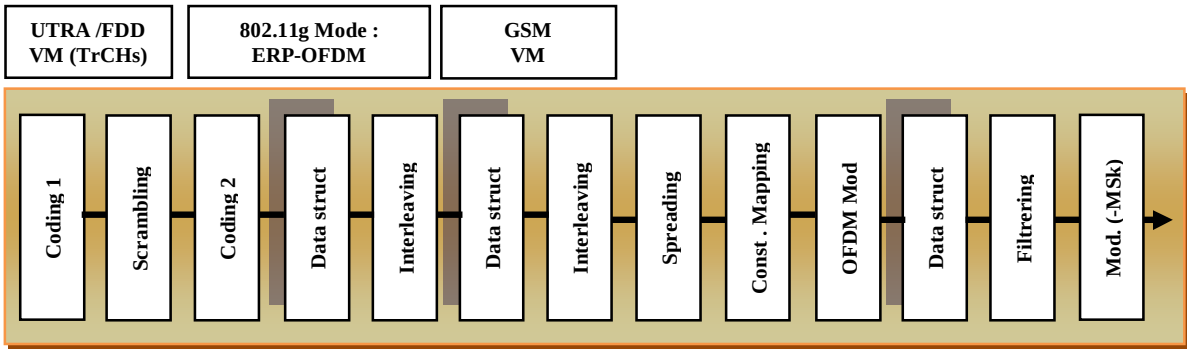

Figure. II.14. Chaîne de transmission multi-standard unifiée [55].

De plus, au moment de l'exécution, tous les modules de traitement quelque soit leur état actif (les modules requis par les paramètres sélectionnés) ou inactif (les modules non requis par les paramètres sélectionnés) consomment de l'énergie, ceci montre alors la perte de l'efficacité énergétique. Bien qu'il existe des mécanismes pour couper l'alimentation sur des fonctions entières, il est plus compliqué de couper l'alimentation sur des sous-blocs d'une même fonction [18]. Le coût en surface, qui peut aussi être significatif [18] [64], s'ajoute aux deux problèmes rencontrés précédemment. Les limites de la technique « fonction commune » ouvrent la voie à l'émergence d'une autre approche définie comme indépendante de la norme qui les utilise.

7.2. Opérateurs communs (OC)

La technique de l'opérateur commun est une deuxième approche de la paramétrisation située à un niveau de granularité intermédiaire entre fonction commune et opérateur arithmétique classique. Ce niveau de granularité est considéré comme idéal, car il fournit le meilleur compromis entre performances et complexité [62].
Cette technique consiste à faire une étude bibliographique sur les différentes architectures de transmission/réception (codage de canal, décodage, modulation, etc.…) pour chaque standard afin d'identifier les points de similarité entre les différentes fonctions et architectures étudiées [2][4][66] [68]. Les points de similarité identifiés nous permettent ensuite de proposer des éléments communs (opérateurs ou structures de calcul) de granularité intermédiaire pour les appliquer à un grand nombre de fonctions.

Les opérateurs considérés ont pour objectif d'être communs à différents standards et aussi à différentes fonctions d'un même standard. Ils sont capables de passer d'un mode de fonctionnement à un autre c'est-à-dire reconfigurable par simple téléchargement de paramètres. Ces architectures génériques sont paramétrables en cours d'exécution et peuvent aussi être implantées sur n'importe quelle cible, logicielle ou matérielle [18] [68]. Les opérateurs peuvent être adaptables et évolutifs à n'importe quel standard, si les opérateurs communs sont définis comme indépendants des fonctions et par conséquent des normes qui les utilisent. En d'autres termes, un OC peut être défini par un opérateur, qui peut être réutilisé par chaque fonction ou bien le traitement exige la fonctionnalité de

l'opérateur, indépendamment du contexte d'application. Dans ce cas, la réutilisation des ces opérateurs génériques devient très optimale. En essence, un CO est utilisé pour effectuer des opérations sans savoir leur application [62]. Selon L. Alaus et al [67] la technique de l'opérateur commun est considérée comme une technique ouverte (open technique). En plus de l'aspect évolutif, cette technique aspire à être un design optimisé permettant d'obtenir un gain en complexité vis-à-vis de la méthode Velcro Classique [2] [4] [69].

Palicot et al [68] introduisent cette notion d'opérateur commun en radio logicielle par l'exemple de l'opérateur FFT. Ils se sont basés sur l'approche pragmatique, ils constatent que plusieurs traitements intéressant dans la chaine de transmission/réception tels que la (dé)modulation OFDM, l'égalisation, le filtrage, etc., peuvent être effectués avec l'opérateur FFT. Cette étude présentée peut être considérée comme pionnière sur le sujet des opérateurs communs [68]. Dans la littérature [8][70], les auteurs développent quatre opérateurs communs à base des registres LSFR: (1) Reconfigurable Fibonacci LFSR (RF-LFSR), (2) Reconfigurable Galois LFSR (RG-LFSR), (3) Reconfigurable LFSR (R-LFSR) et (4) Extended Reconfigurable LFSR (ER-LFSR) qui peuvent être utilisés pour plusieurs fonctions tel que le codage canal et filtrage. A.Al Ghouwayel [69] propose une paramétrisation de l'opérateur FFT dans le champ de Galois pour la réalisation d'un encodeur Reed-Salomon. Cet encodeur est réalisé dans le domaine fréquentiel pour bénéficier de l'architecture matérielle performante du papillon FFT. L'opérateur proposé par A.Al Ghouwayel [69] est appelé double mode FFT (DMFFT), car ce dernier est capable de fonctionner sur deux domaines différents (champ complexe (C) et le champ de Galois (GF)). Finalement, M. Naoues et al. [3] [71] ont proposé des cellules de traitement matériel reconfigurables pour les algorithmes FFT et Viterbi, capables de prendre en compte leurs différences fonctionnelles et de s'y adapter à l'utilisation.

La technique des Opérateurs Communs est une approche ambitieuse dans le design de terminaux multistandards mais nécessite une gestion complexe (scheduling). Ce point faible de la méthode affecte non seulement la capacité du terminal à gérer les opérateurs mais aussi la complexité résultante de leurs implémentations en introduisant le besoin des structures de contrôle [72].

8. Notre solution préconisée dans le travail de thèse

Si on fait une analyse des plateformes de la radio reconfigurable et des solutions existantes pour avoir la flexibilité, la performance, l'efficacité énergétique et l'efficacité en surface attendue par la radio logicielle (SDR), nous pouvons tirer essentiellement la remarque suivante: la plate-forme hybride offre un bon compromis en termes de flexibilité et de complexité, mais la configuration d'une telle plate-forme revient à traiter les problèmes de synchronisation et de répartition des tâches entre les différentes

composantes. De plus, la condition d'avoir un environnement unique de travail se pose également. Pourtant, on peut remarquer que la technologie FPGA laisse la place à une solution intermédiaire, où on allie flexibilité de la programmation et puissance de calcul des architectures spécialisées. Sachant que, le développement des FPGA de haute performance et de capacités de calculs élevées est en pleine évolution offrant ainsi la possibilité de concevoir des System On Programmable Chip (SoPC) intégrant des processeurs embarqués, de mémoire de large capacité, des accélérateurs dédiés, et les différents types de flexibilité logicielle et matérielle.

En effet, selon M. Cummings [73] le compromis performance/flexibilité apporté par les FPGA en fait un bon candidat pour les applications radio logicielle par rapport aux ASIC et DSP. Aussi Selon Laurent Alhaus [5] les FPGA sont en fait un bon candidat pour la réalisation d'une plateforme hybride pour la Radio Logicielle, puisque les FPGA ressemblent de plus en plus à des systèmes sur puce, et présentent l'ensemble des niveaux de granularité: **(1) petit grain:** des systèmes de LUTs, **(2) grain moyen:** Multiplieurs à X-bit, additionneurs, et un accumulateur à K-bit, et **(3) gros grain:** DSPs, processeurs PowerPC. C'est pour cette raison que nous nous sommes particulièrement intéressés à ce type de circuits et à leur mise en œuvre dans les applications « radio logicielle ».

De plus, la paramétrisation et la technique d'opérateurs communs sont des techniques clés dans la radio logicielle [68][69]. Selon Malek NAOUES [18] la technique d'opérateurs communs se situe parmi les techniques de reconfiguration actuelles et vise une implémentation multistandards plus efficace. J.P. Delahaye [55], a fait dans sa thèse une comparaison de différentes approches de réalisation de traitements multi-standards (tableau II.2.).

Tableau II.2. Comparatif des approches de conception des multi-traitements [55].

	Fonction Fixe (Velcro)	Fonction reconfigurable	Paramétrisation Operateur commun	Paramétrisation Fonction commune
Performance	+++	++ (+)	+	++ (+)
Surface	--	++	++	-
Consommation Energétique	--	++	++	-
Temps de changement de contexte	+++	–	+	+++
Potentiel de réutilisabilité	--	++	+++	+
Complexité de conception	+++	++	--	+

À partir de cette comparaison, on peut remarquer que les deux approches: la reconfiguration et la Paramétrisation avec la technique d'opérateurs communs, présentent le meilleurs choix pour la radio logicielle. Cependant, le temps de changement dans l'approche reconfigurable représente son grand point faible. En effet, le temps de changement de modes doit être négligeable dans la radio logicielle, puisque les couches physiques travaillent en temps réel. Par contre, la technique de paramétrisation avec l'approche opérateur commun apparaît comme une technique très efficace pour le SDR et constitue elle même une méthodologie de conception de la Plateforme Hybride (hétérogène). De plus, la Paramétrisation avec la technique d'opérateurs communs peut constituer une méthode de reconfiguration en elle-même ou une technique complémentaire aux méthodes existantes telle que la Reconfiguration Partielle d'un FPGA.

Prenons l'exemple de [5] qui suppose qu'une technique de paramétrisation définisse une entité commune (COM1) entre deux standards S1 et S2. Supposons toujours que le paramètre α appliqué à COM1 permet d'exécuter une opération de S1 et que le paramètre β appliqué à COM1 permet d'exécuter S2. Alors, passer de S1 à S2 ne nécessitera plus de reconfigurer le terminal, de charger un code ou d'envoyer un bitstream mais juste de fournir en plus des données le paramètre α ou β selon le cas. Maintenant, supposons l'existence d'une deuxième entité commune COM2, suivant le même principe. En combinant Paramétrisation et reconfiguration partielle, nous pouvons sur la même sous partie du FPGA, faciliter la reconfiguration partielle en autorisant à reconfigurer seulement deux fois (soit COM1, soit COM2) au lieu de passer par quatre reconfigurations partielles.

Cette technique vise à ajouter aussi de la flexibilité pour les composants qui ne sont pas flexibles tel que les ASICs ou encore d'améliorer plus cette flexibilité pour les FPGA. Ceci nous permet d'utiliser des composants matériels de haute performance présentant une grande efficacité énergétique. Cette technique est surtout performante dans un FPGA pouvant lier la flexibilité logicielle et matérielle aussi bien à la conception que durant l'utilisation. C'est pour cette raison que l'approche opérateur commun a été privilégiée dans notre travail.

9. Conclusion

Dans ce chapitre nous avons présenté le besoin des terminaux reconfigurables pour la radio logicielle restreinte et la radio intelligente. Nous avons également donné le détail des différents éléments constitutifs d'une radio logicielle restreinte, en commençant par l'antenne agile jusqu'à, la partie de bande de base. Cette description détaillée des différents éléments de SDR, est renforcée par des exemples de quelques travaux de recherches récentes dans ce domaine.

En termes de reconfiguration du terminal multistandard, nous avons remarqué que celui-ci ne peut se réaliser exclusivement sur des cibles logicielles mais demande l'utilisation des cibles matérielles. La plateforme hétérogène est donc une solution prometteuse. Cependant il présente ainsi quelques interrogations à éclaircir.

Dans ce contexte, la technique d'opérateur commun définit des éléments qui peuvent être "reconfigurés" en temps réel en ne modifiant que des simples paramètres. Cette technique constitue elle-même une méthode de reconfiguration et une méthodologie de conception de la plateforme hybride. En plus si elle est combinée avec la technique de reconfiguration partielle, nous pouvons sur la même sous partie du FPGA faciliter ce type de reconfiguration.

Dans le chapitre qui suit, nous proposons un opérateur générique et versatile pour trois fonctions : l'algorithme CORDIC, FFT-SDF et l'unité de désétalement. Il peut être considéré comme un opérateur générique (commun) utilisé dans un grand nombre d'algorithmes classiques de traitement du signal. En plus dans le chapitre 4 nous proposons des architectures universelles pour la modulation et la démodulation.