# TABLE DES MATIÈRES

# LISTE DES FIGURES

# LISTE DES TABLEAUX

# LISTE DES ACRONYMES

**AM**: Amplitude Modulation

**CAN**: Analog to Digital conversion

**CAP**: Contention Access Period

**CDMA**: Code Division Multiple Access

**CFP**: Contention Free Period

**CSMA-CA**: Carrier Sense Multiple Access with Collision Avoidance

**DSP**: Digital Signal Processing

**DSSS**: Direct Sequence Spread Spectrum

**FFD**: Full Function Device

**FPGA**: Field-programmable gate array

**GMSK**: Field-programmable gate array

**GTS**: Guaranteed Time Slots

**ISM**: Industrial, Scientific and Medical

**LNA**: Low noise Amplifier

**LR-WPAN**: Low-Rate Wireless Personal Area Networks

**MAC**: Media Access Control

**MIMO**: Multiple-Input Multiple-Output

**O-QPSK**: Quadrature phase shift keying

**PA**: Power Amplifier

**PAN**: Personal Area Network

**RF**: Radio Frequency

**RFD**: Reduce Function Device

**SDR**: Software Defined Radio

**SNR**: Signal-to-noise Ratio

**USRP**: Universal Software Radio Peripheral

**WLAN**: Wireless local Area Network

**WSN**: Wireless Sensor Networks

CHAPITRE 1

# INTRODUCTION GÉNÉRALE



## 1.1. Contexte

Les communications sans fil représentent l'un des domaines les plus dynamiques dans le monde des télécommunications d'aujourd'hui. Bien qu'il ait été un sujet d'étude depuis les années 1960, les dix dernières années ont vu une forte augmentation des activités de recherche dans ce domaine. De nombreuses technologies évoluent donc en permanence et changent graduellement le monde des télécommunications. Dans notre travail, on s'est intéressé aux réseaux de capteurs où chaque entité réseau est alimentée à l'aide d'une batterie, ce qui permet une autonomie de trois à quatre années. Ce type de réseau permet de surveiller les conditions physiques ou environnementales, telles que le son, la vibration, la pression, le mouvement ou les polluants des zones rurales et transmette leurs données via le réseau à un emplacement principal. Ceci devrait beaucoup aider les intervenants en mesures d'urgence comme les militaires et les pompiers pour contrôler et surveiller les zones éloignées comme les forêts, par exemple pour éviter la propagation des incendies de forêts ou bien réagir au moment propice.

Dans le cadre de la maîtrise en ingénierie des télécommunications à l'université du Québec en Abitibi-Témiscamingue, on a eu l'opportunité de travailler sur un projet de recherche qui a comme but de relayer et faire passer le trafic entre deux types de réseaux hétérogènes (réseaux de capteurs et réseaux de stations de bases ) en

se basant sur les nouvelles technologies de radio télécommunications. Pour bien comprendre le but du projet, une mise en œuvre de ce concept permettra la programmation et l'utilisation du protocole de radiocommunications ZigBee sur un matériel compatible permettant le relais ou la réception du signal. Quant à la partie traitement de ce signal, elle se fait de manière logicielle. Cette particularité permet d'en faire un équipement reprogrammable et donc adaptable à n'importe quelle technologie de transmission sans fil, le but final étant de réaliser un analyseur de protocole ZigBee, c'est à dire capter le signal ZigBee et le relier de façon à pouvoir l'analyser au niveau du coordonateur.

### 1.2. Plan du mémoire

Ce mémoire est composé de six chapitres. Un ensemble de recherches a eu lieu pour préciser le contexte du projet et la problématique dans le premier chapitre. Le deuxième chapitre présente l'importance des réseaux de capteurs et les différences applications, topologies, standards et architectures avec lesquels ils peuvent fonctionner. Le troisième chapitre présente l'utilité et le rôle principal que la radio logicielle joue dans l'interconnexion des radios à bases des ondes radios. Par la suite, on présente la solution Gnu Radio comme outil logiciel pour réaliser les expérimentations. La dernière partie de ce chapitre présente le projet UCLA ZigBee, qui se présente sous la forme d'une libraire qu'on implémente sur GnuRadio et qui contient des blocs responsables de la transmission/ la réception et du traitement de signal. Le quatrième chapitre donne les détails des équipements matériels utilisés, soit en ce qui concerne l'USRP ou le capteur ZigBee. Pour résoudre la problématique de ce travail, on a aussi présenté un schéma des blocs responsables du relais. Le chapitre cinq présente l'environnement de tests qui a été préparé pour les expérimentations. La partie suivante de ce chapitre contient les résultats des tests réalisés pour le cas de la communication entre deux stations de bases (USRP), ou

bien pour le cas de la communication des stations de bases avec les capteurs ZigBee, en incluant les mesures qui ont été prises pour le calcul du relais pour différent cas.
Le dernier chapitre donne une conclusion générale, et les étapes de travail de chaque chapitre sont résumées en quelques lignes. On dresse finalement les perspectives proposées pour les projets du futur.

### 1.3. Problématique à l'étude

Selon la figure 1.1 ci-dessous, la problématique principale de ce travail est de pouvoir capter le trafic ZigBee à travers les stations de bases et de traiter l'information obtenue du réseau de capteurs et réagir selon les conditions, ce qui va permettre une surveillance des zones éloignées. L'objectif final est le relais car la distance maximale entre un capteur et un autre ne dépasse pas 100 mètres si la fréquence utilisée est 2.4 GHz. Alors, il faut pouvoir envoyer le signal jusqu'au coordinateur qui s'occupe du traitement de données et qui se trouve loin (plusieurs kilomètres) dans les es zones rurales, là où se trouve le réseau de capteurs. Le travail demandé est la connexion du réseau de capteurs avec un autre de station de base. Pour cette approche, on doit utiliser des capteurs à faible consommation et à faible coût, et c'est dans cette optique que le standard ZigBee a été introduit car ce protocole permet une durée de vie des capteurs sur une plusieurs années, leur alimentation étant toutefois sur batterie. Selon l'état de l'art, on a déduit que la résolution du problème du relais du signal entre deux réseaux hétérogènes demande qu'on travaille avec la solution radio-logicielle qui permet d'avoir un récepteur ou un transmetteur radio. La tâche principale à faire est la programmation logicielle des blocs responsable de traitement de signal et, dans une moindre mesure, sa réalisation avec du matériel commercial.

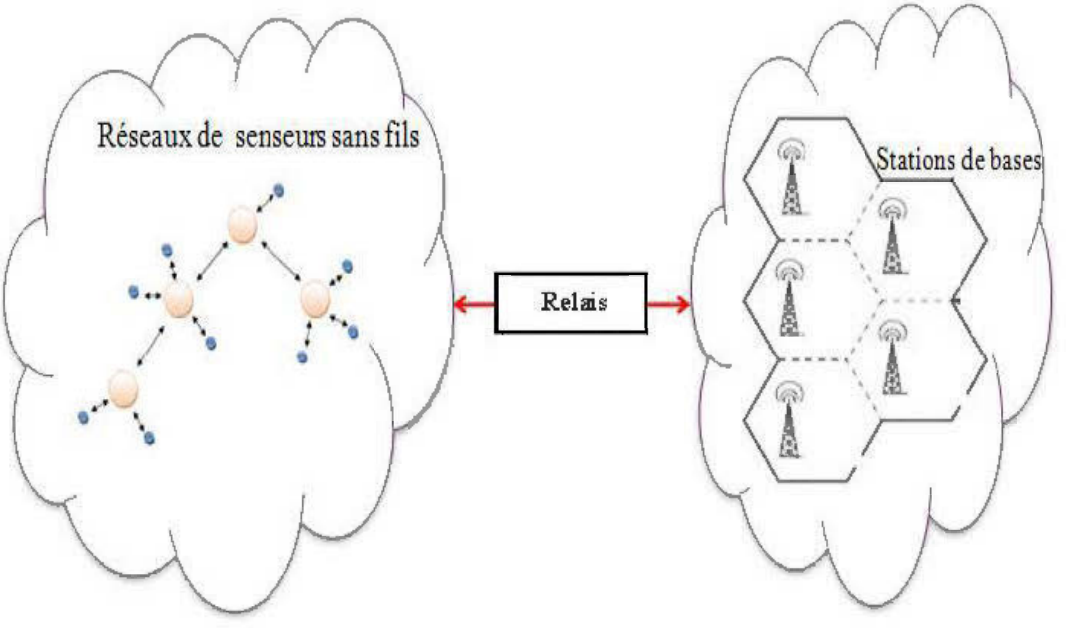



**Figure 1. 1:** L'interconnexion de deux réseaux hétérogènes.

# CHAPITRE 2

# LES RÉSEAUX DE CAPTEURS

## 2.1. Introduction

Actuellement, dans les domaines de télécommunications, de nombreuses applications de connexions sans fils, aussi bien dans les domaines industriels que personnels, requièrent des transmissions à faibles débits, organisées sous forme de réseaux. Après une recherche sur l'état de l'art, notre travail se base sur les réseaux de capteurs sans fils nommé sous WSN (Wireless Sensor Networks) [1]. Ce type de réseau se compose d'un grand nombre de nœuds qui sont des micro-capteurs capables de récolter et de transmettre des données environnementales d'une manière autonome. La position de ces nœuds n'est pas obligatoirement prédéterminée. Ils peuvent être aléatoirement dispersés dans une zone géographique, appelée « champ de captage » correspondant au terrain d'intérêt pour le phénomène capté.

## 2.2. Fonctionnalités des réseaux de capteurs

Dans les réseaux de capteurs, toute configuration manuelle ou toute administration devient inutile ce qui, dans certaines applications, est très appréciable. Individuellement, chaque nœud possède une vue locale qui permet de communiquer avec les nœuds proches [2]. En général, la notion d'auto-organisation. Permet de résoudre une tâche complexe en agissant de manière coordonnée tout en étant adaptable, robuste et de dimensions ajustables.

### 2.2.1. Adaptable

Un réseau de capteurs, dans un lieu de géographique où on est obligé de changer l'emplacement des capteurs de temps à autre dépendant des besoins ou des nécessités doit bénéficier d'une adaptabilité acceptable pour les applications pressenties. La fonction adaptable, dans un réseau de capteurs, permet une facilité pour le nœud de s'adapter au changement de situations, c'est-à-dire qu'il est dynamiquement configurable

### 2.2.2. Robuste

Il se peut qu'un nœud dans un réseau de capteurs installé dans une zone éloigné perde son efficacité ou tombe en panne, pour diverses raisons. Alors la fonction robuste permet au réseau de continuer à fonctionner, car dans chaque groupe de nœuds, un de ceux-ci est le principal et il s'occupera du nœud de la communication avec les autres groupes du réseau.

### 2.2.3. Dimension ajustable

Les réseaux de capteurs sont connus par leur une densité très importante, allant parfois jusqu'à des milliers d'unités. La fonction dimension ajustable permet au réseau de capteurs de s'adapter aux nouveaux nœuds installés grâce à une facilité de déploiement et une configuration automatique. Cela permet un fonctionnement constant de tout le réseau

## 2.3. Application et services

Depuis les années 1990, le monde des technologies sans fils a connu beaucoup de progrès (diminution de la taille et des coûts des micro-capteurs, élargissement de

la gamme des types de capteurs disponibles, évolution des supports de communication sans fil), de même que le champ d'application des réseaux de capteurs qui est devenu assez élargi. Les ingrédients pour le développement des applications concrètes sont les facultés réelles de détection et d'actionnement qui peuvent équiper un nœud capteur qui devient alors plus doué et très précis. Voici quelques exemples d'applications des réseaux de capteurs :

### 2.3.1. Application militaires

Tel qu'on le retrouve pour plusieurs technologies, le domaine militaire a été un moteur principal pour le développement des réseaux de senseurs. Le déploiement facile, le coût réduit, la notion "d'auto-organisation" et la tolérance aux pannes sont des caractéristiques qui permettent à ce type de réseaux de pouvoir être considéré comme un outil appréciable dans ce domaine. Comme exemple d'application dans ce domaine, on peut penser à un réseau de capteurs déployé dans un endroit géographique difficile d'accès afin de surveiller toutes les activités et les mouvements des forces ennemies, ou d'analyser le terrain avant d'y envoyer des troupes. Dans une autre type d'utilisation, un peut considérer un capteur servant à détecter le mouvement des soldats et, grâce ce dispositif, les intervenant peuvent connaître l'état actuel d'un combattant, si il est entrain de dormir, de courir, de marcher ou si il est mort grâce à la détection de la température du corps.

### 2.3.2. Applications médicales et vétérinaires

La surveillance des fonctions vitales d'un organisme vivant pourrait, à l'avenir, être plus facile grâce à des micro-capteurs avalés ou implantés sous la peau. D'ambitieuses applications biomédicales sont à l'étude comme la surveillance de la glycémie, la surveillance des organes vitaux ou la détection précoce de cancers. Les

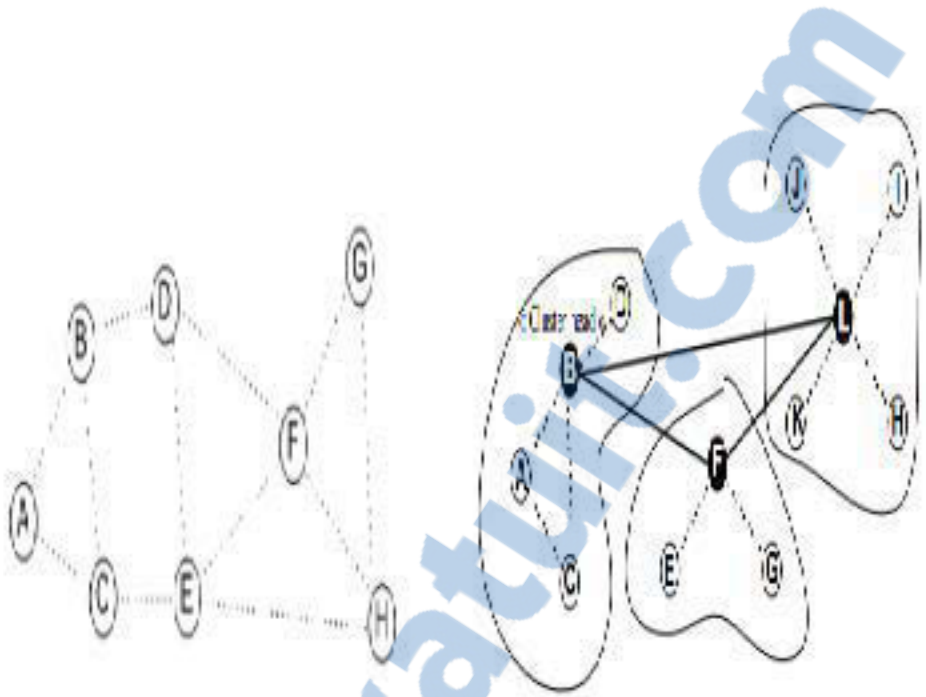
réseaux de capteurs permettraient, théoriquement, une surveillance permanente des patients et offriraient la possibilité de collecter des informations physiologiques de meilleure qualité, facilitant ainsi le diagnostic de certaines maladies.

### 2.3.3. Application en sécurité

Les mouvements des bâtiments, dûs au vieillissement ou aux tremblements de terre, pourraient être détectés avec des capteurs intégrés dans les murs ou dans le béton, sans alimentation électrique ou autres connexions filaires. Un réseau de capteurs de mouvements peut constituer un système d'alarme distribué servant à détecter les intrusions sur un large secteur. La déconnexion du système ne serait plus aussi simple puisqu'il n'existe pas de point critique. La protection des barrages pourrait être accomplie en y introduisant des capteurs dans le but de vérifier le niveau de l'eau afin d'éviter les inondations. La surveillance des voies ferrées pour prévenir les accidents de trains avec des animaux ou des êtres humains. Ce type d'application veillera sur la sécurité de l'être humain.

## 2.4. Architecture et déploiement

Dans ce type de réseau, il n'y a pas d'infrastructure fixe, comme c'est par exemple le cas pour les stations de base des réseaux cellulaires. Ici, les nœuds sont des micro-capteurs capables de récolter et de transmettre des données entre les entités réseaux, Ainsi, chaque nœud achemine l'information vers un autre nœud dans le réseau, tel qu'illustré à la figure2.1.a. Ce type de topologie se nomme la topologie plate. Il est également possible d'utiliser une topologie hiérarchique (Figure 2.1.b) où des groupes de nœuds élisent un maître, le (cluster Head), qui est le responsable de la communication avec les maîtres des autres groupes.




A: Topologie Plate

B: Topologie Hiérarchique

**Figure 2. 1:** Topologie de réseaux de capteurs sans fils [2]

### 2.4.1. L'infrastructure

L'infrastructure d'un réseau de capteurs se base sur les entités réseaux. Plus précisément elle influence, par les caractéristiques physiques des différents capteurs (la capacité de capture et de calcul, la taille de la mémoire, les caractéristiques de la batterie, et le mode de transmission...) et la stratégie de déploiement (la densité des capteurs, leur localisation, et leur mobilité).

### 2.4.2. Les protocoles du réseau

Ils sont responsables de maintenir des chemins de communication entre les capteurs et les observateurs. Le cluster Head se trouve dans chaque groupe; il est responsable de la gestion et de la configuration des entités qui se trouve dans sa cellule. En plus, il s'occupe du routage ce qui lui permet de lier la communication entre différents nœuds à base des Protocoles réseaux.

## 2.5. Standards

Nous avons déjà souligné, dans l'introduction aux réseaux de capteurs, que la contrainte principale dans ce type de réseaux est la consommation d'énergie, ce qui prouve que les autres standards classiques sans fils tel que Bluetooth, WLAN, etc.…ne peuvent pas être adaptés pour ce type de réseaux. Les seuls standards qui sont convenables pour l'utilisation de la communication sans fils dans les réseaux de capteurs sont le standard IEEE 802.15.4 et ZigBee.

### 2.5.1. Le standard IEEE 802.15.4

Cette norme définit le protocole et l'interconnexion des appareils par communication radio à titre de réseau personnel (PAN) [3]. La norme utilise la détection de la porteuse à accès multiple avec un milieu permettant l'évitement des collisions dues aux mécanismes d'accès et soutient les topologies étoiles ainsi que le point à point. L''accès des médias se fait en utilisant la structure de la supertrame, les créneaux horaires peuvent être alloués par le coordinateur du PAN pour les appareils avec des données en temps critiques aux réseaux plus performants est fournie par un PAN coordinateur.

#### 2.5.1.1. Les topologies réseaux du standard IEEE 802.15.4

Selon les exigences de l'application, le LR-WPAN peut fonctionner dans l'une des deux topologies suivantes: la topologie étoile ou la topologie point à point. Les deux sont présentées à la figure 2.2 ci-dessous. Dans la topologie en étoile la communication est établie entre les périphériques et un contrôleur central unique et le coordonnateur est appelé un PAN. Un dispositif de ce genre a une certaine application qui lui est associée soit au point d'initiation ou le point de la communication terminale du réseau. Le coordonnateur de PAN est le contrôleur principal et tous les dispositifs fonctionnant sur un réseau de topologie ont une unique adresse de 64 bits étendus. Cette adresse peut être utilisée pour la communication directe au sein du PAN. Le coordinateur de PAN peut fonctionner sur le secteur, tandis que les dispositifs seront très probablement alimentés par une pile.

La topologie point à point a aussi un coordonnateur de PAN, mais il diffère de la topologie en étoile dans le sens où tout appareil peut communiquer avec n'importe quel autre appareil aussi longtemps qu'ils sont à portée les uns des autres. La topologie point à point permet des formations réseaux plus complexes à mettre en œuvre, telles que la topologie de réseau maillé. Cette dernière permet la fonction de l'auto-organisation et les sauts multiples pour router les messages à partir de n'importe quel appareil à un autre appareil sur le réseau. Ces fonctions peuvent être ajoutées à la couche réseau, grâce à l'alliance ZigBee [3].

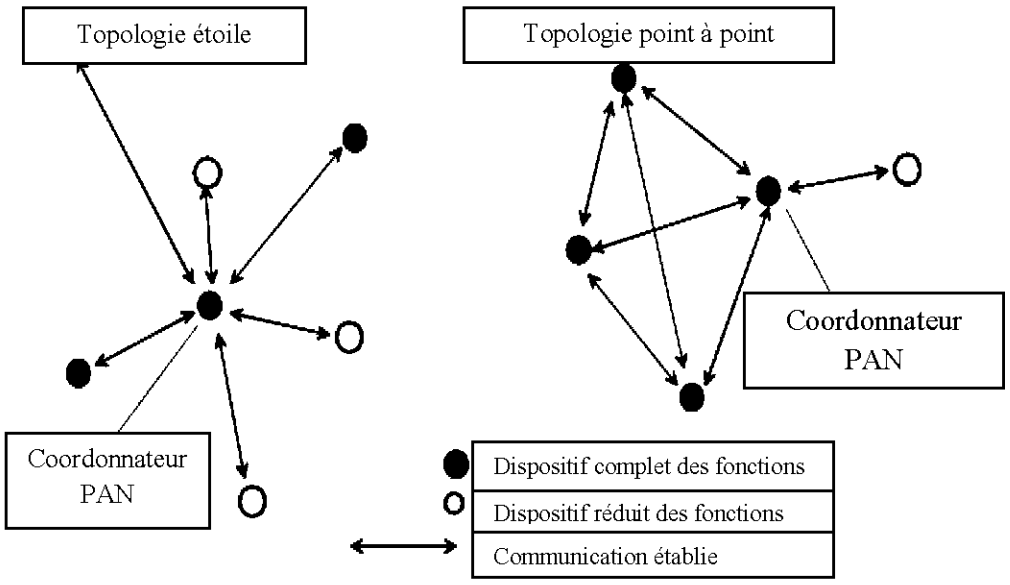



**Figure 2. 2:** Topologies étoile et point à point [3].

Chaque PAN indépendant sélectionnera un identifiant unique. Cet identifiant PAN permettra la communication entre les appareils au sein d'un réseau en utilisant des adresses courtes et des transmissions entre les dispositifs à travers des réseaux indépendants.

### 2.5.1.2. Les caractéristiques du standard IEEE 802.15.4

La couche physique est simple.

- La couche MAC est également simple, l'accès des nœuds au canal étant géré à l'aide du protocole CSMA-CA qui se base sur l'écoute avant d'utiliser le canal pour ne pas avoir des problèmes de collisions

- Le débit binaire est faible, varie de 20 à 250 kbits/s selon la bande de fréquence utilisée.
- La portée se situe entre 10 et 100 m, dans les bandes de fréquence ISM 800/900 MHz et 2.4 GHz.
- Le protocole, défini pour une durée de vie de batterie maximale, permet que chaque capteur fonctionne sur une alimentation par pile pendant une durée minimum de 10 ans.



### 2.5.1.3. La Couche MAC du standard IEEE 802.15.4

La couche liaison de données MAC gère les accès au médium radio et résout les problèmes d'accès concurrents au médium. Le standard 802.15.4 propose deux types d'accès [3] : un mode non coordonné qui est le CSMA/CA et un mode coordonné qui est le mode temps garantie ou GTS (guaranteed time slot) qui est disponible uniquement dans une topologie étoile, où le coordinateur envoie périodiquement des trames balises (beacon) pour synchroniser et préparer les nœuds du réseau pour l'envoi ou la réception. Toute entité du réseau qui écoute cette balise peut ainsi se synchroniser et se servir de ce coordonnateur comme d'un relais. Ce type de mode permet d'avoir les meilleures performances sur le plan énergétique car, une fois l'information transmise au relais, le nœud transmetteur peut somnoler; de même, les messages étant stockés dans la mémoire du relais, le nœud destinataire se prépare à choisir l'instant où il va demander le rapatriement des données pour prolonger sa durée de somnolence. L'espace temporel entre deux trames balises s'appelle une supertrame.

La supertrame comprend une portion active ou les noeuds peuvent émettre et recevoir et une autre portion inactive où les noeuds sont en somnolence. La portion active est divisée en 16 intervalles (slots) temporels qui ont la même durée, le beacon occupe toujours la première slot qui est le 0 et permet la synchronisation et la

préparation de tous les nœuds à portée radio. Les noeuds du réseau se réveillent juste avant la slot 0 et se mettent à l'écoute pour se préparer à l'échange. À la réception du beacon, ils prennent connaissance de la structure de la supertrame qui débute et préparent les autres nœuds à échanger les données qui sont en attente s'ils n'ont ni de données à émettre ni à recevoir, tous les entités du réseau peuvent somnoler jusqu'au beacon suivant; sinon, ils se mettent en veille dès que le cursus (émission/réception) est terminé. La figure 2.3 représente la structure d'une supertrame IEEE 802.15.4.

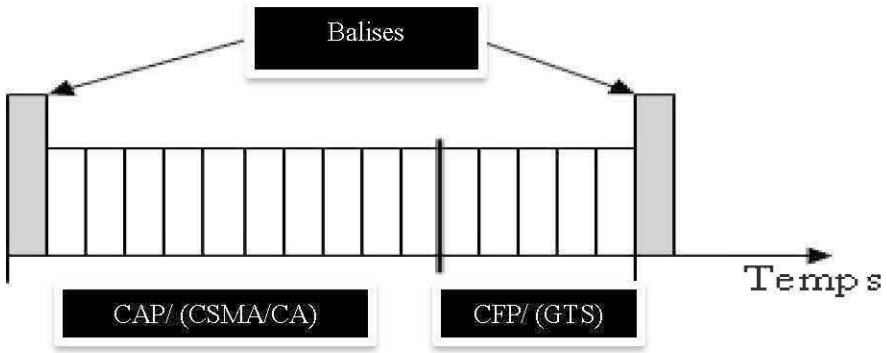



**Figure 2. 3:** Structure d'une supertrame IEEE 802.15.4

Le mode coordonné de 802.15.4, propose deux méthodes d'accès au niveau de la supertrame :

**Avec contention** : Dans ce mode, les accès au médium se font de façon classique par le best-effort, selon le protocole CSMA/CA qui se base sur l'écoute du canal. Ce mode d'accès au canal est toujours possible et, systématiquement, on dédie une partie de la supertrame pour ce mode.

**Sans contention :** Cette partie de la supertrame est appelée CAP (Contention Access Period) où les accès au médium sont maîtrisés par le coordonnateur qui gère le groupe et l'interconnecte avec les autres groupes. Ce mode peut être utilisé par les noeuds qui font la demande pour accéder au canal, et, si la capacité du réseau le permet, le coordonnateur pourra allouer une ou plusieurs slots à un noeud en particulier. On parle ici de slots temporelles dédiées ou GTS (Guaranteed Time Slots). Les GTS occupent les dernières slots de la supertrame. Cette partie de la supertrame s'appelle la période d'accès sans contention ou CFP, pour Contention Free Period. Ce mode sans contention rend la réservation de la bande passante possible et il permet d'offrir des garanties temporelles. Le début de la supertrame, via la CAP, reste toujours en accès libre par le protocole CSMA/CA pour permettre l'accès aux transports ne nécessitant pas ou peu de garantie. Les demandes de GTS ainsi que les demandes d'association au réseau ne peuvent se faire que dans la CAP. C'est pour cette raison qu'on doit limiter la taille du CFP.

### 2.5.1.4. La Couche Physique du standard IEEE 802.15.4

Le standard IEEE 802.15.4 spécifie la couche physique (PHY) dans les bandes ISM 868 MHz, 915 MHz et 2.4 GHz avec des débits respectifs de 20, 50 et 250kbits/s [2]. Le nombre de canaux disponible étant plus important dans la bande 2.4 GHz, comme c'est montré dans la figure ci-dessous c'est elle qui est la plus utilisée.

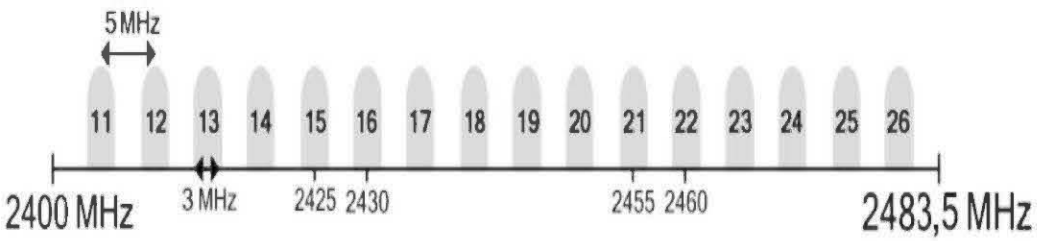



**Figure 2. 4:** Répartition des canaux 802.15.4 dans la bande 2.4 GHz [2].

Dans la bande disponible entre 2.405 GHz et 2.4835 GHz, le standard 802.15.4 défini donc 16 canaux avec un espacement inter-canal de 5MHz. Ils sont numérotés de 11 à 26, les dix premiers correspondants aux canaux disponibles dans la bande de fréquence 868 et 915 MHz.

Enfin, précisons que ce standard fonctionne en mode half-duplex, pour lequel il ne peut y avoir simultanément réception et transmission de données, contrairement à ce qui est réalisé dans le mode full-duplex.

#### 2.5.1.4.1. La modulation du standard IEEE 802.15.4

La chaîne d'émission du standard IEEE 802.15.4 utilise le codage des bits par étalement spectral qui est le DSSS (Direct Sequence Spread Spectrum). Cette technique d'étalement permet de faire fonctionner la transmission avec le rapport signal à bruit (SNR) plus important que ceux des systèmes à bande étroite, d'où l'intérêt d'une bande passante plus importante. La technique utilisée par le standard 802.15.4 consiste à faire correspondre à chaque groupe de 4 bits (appelés symboles) un code d'étalement de 32 bits appelés chip dans le cadre d'un codage par étalement de spectre qui est fixé par le standard. Le débit brut dans l'air est 250 kbits/s, le chip obtenu est de 2 Mchips/s.

Les chips sont modulées sur la fréquence porteuse à l'aide d'une modulation O-QPSK (Offset Quaternary Phase Shift Keying) où les chips paires sont envoyés sur la voie I et les chips impaires sur la voie Q, chaque chip étant répété deux fois sur chaque voie. Un exemple de cette modulation est présenté sur la figure 2.5. Il montre les signaux I et Q après modulation pour une trame de chips composée de 110110001 ainsi que le diagramme de phase correspondant.

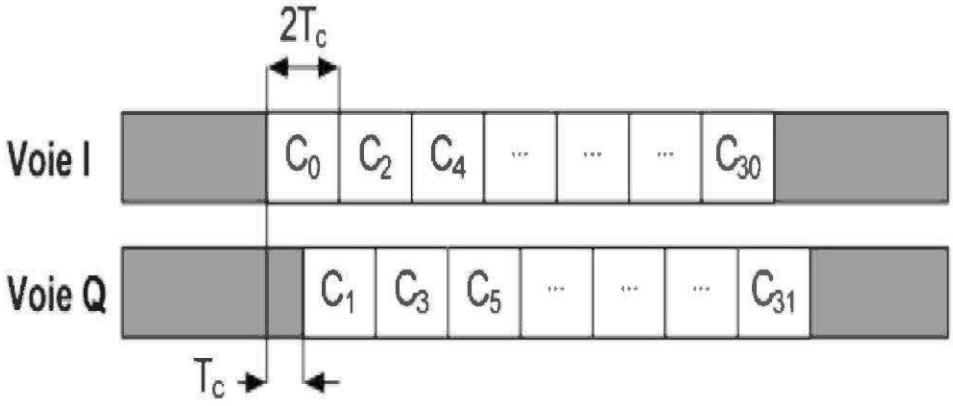



**Figure 2. 5:** Répartition des chips sur les voies I et Q [4].

La modulation O-QPSK est une modulation à enveloppe constante permettant de faciliter la conception de la chaîne de réception. En effet, en l'absence de modulation d'amplitude, le signal est très résistant au bruit et il est en mesure de supporter des phénomènes de compression tant que ces phénomènes n'arrivent pas à introduire des erreurs dans la modulation de phase. La figure 2.6 donne l'allure du spectre pour une modulation O-QPSK.

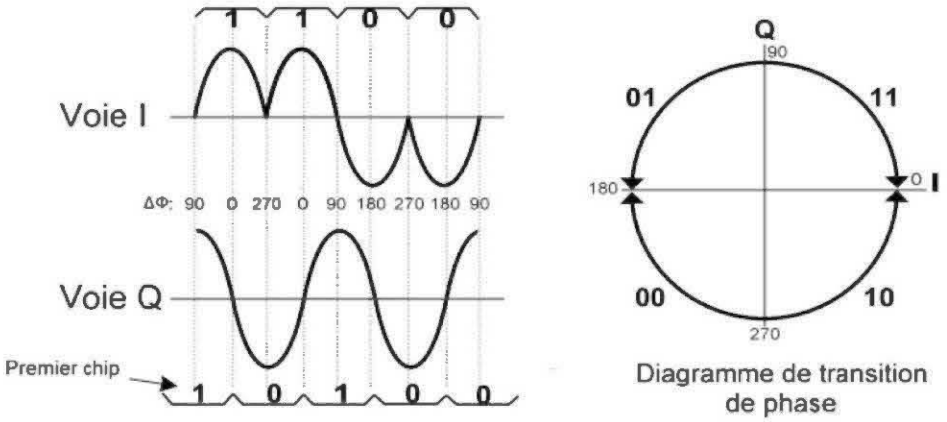



**Figure 2. 6:** Exemple de la modulation O-QPSK pour la trame de chips 1101100001 [4].

Le tableau ci-dessous nous présente les principales caractéristiques de la couche physique 802.15.4, ce qui nous précise le débit, la bande totale et surtout la bande du canal et la modulation qui font la différence entre le standard 802.15.4 et les autres standards.

| **Paramètres** | **Valeurs** |
|---|---|
| Débit binaire | 250 kbits/s |
| Bande totale | 83.5 MHz |
| Nombre de canaux | 16 |
| Bande da canal | 3 MHz |
| Espace inter-canal | 5 MHz |
| Modulation | O-QPSK |

**Tableau 2.1 :** Résumé de la couche physique du standard IEEE 802.15.4 [2].

### 2.5.1.5. Comparaison du standard IEEE 802.15.4 avec les autres standards

Le tableau 2.2 ci-dessous présente une comparaison générale entre les différents standards sans fils existants, et montre les raisons pour lesquelles notre choix s'est porté pour le standard 802.15.4. Selon des études [4] ce standard est le plus pour les réseaux de capteurs.

| IEEE | 802.15.4 | 802.15.1 | 801.11a/b/n |
|---|---|---|---|
| **Besoin mémoire** | 4-32 kb | 250 kb+ | 1 Mb+ |
| **Autonomie avec pile** | Des années | Des jours | Des heures |
| **Nombre de nœuds** | 64 000 + | 7 | 32 |
| **Débit binaire** | 250 kbits/s | 1 Mb/s | 11-54-108-302 Mb/s |
| **Portée** | 100 m | 10 m | 300 m |
| **La bande du canal** | 3 Mhz | 10 Mhz | 22 Mhz |

**Tableau 2.2 :** Comparaison avec d'autres standards [2].

Selon le la comparaison avec les autres technologies sans fils citées au tableau, on résume les principaux points forts pour le 802.15.4 :

- ✓ Il n'occupe pas beaucoup d'espace mémoire pour fonctionner.
- ✓ La durée de vie de la batterie est assez importante par rapport aux autres standards, elle peut durer une dizaine d'années.
- ✓ Par rapport aux autres standards, le débit est faible et la bande du canal est de 3 MHz, ce qui ne demande pas beaucoup d'énergie pour la communication ou bien le traitement de données.
- ✓ La densité des nœuds dans un réseau peut dépasser 64 000, ce qui fait une énorme différence par rapport aux autres technologies.

### 2.5.2. Le standard ZigBee

La norme ZigBee a été développé par l'Alliance ZigBee [5] qui regroupe des centaines d'entreprises membres de l'industrie des semi-conducteurs, des développeurs d'entreprises membres de l'industrie des semi-conducteurs, des développeurs de logiciels et des installateurs [6,7]. L'alliance ZigBee a été formée en 2002 en tant qu'organisation à but non lucratif ouverte à tous ceux qui veulent y adhérer. Le ZigBee standard a adopté IEEE 802.15.4 pour la couche physique (PHY) et la couche liaison moyen, Access Control (MAC) des protocoles.

La norme ZigBee est spécifiquement développée pour répondre aux besoins de la mise en œuvre à très faible coût et à faible débit de données des réseaux sans fil à ultra-basse consommation. Elle permet de réduire le coût de la mise en œuvre par la simplification des protocoles de communication et la réduction du débit de données. Les exigences minimales qui spécifient le ZigBee et IEEE 802.15.4 sont relativement détendues par rapport à d'autres standards, ce qui réduit la complexité et le coût des émetteurs-récepteurs de type ZigBee.

La figure 2.7 ci-dessous montre la différence entre l'alliance ZigBee et le standard IEEE 802.15.4 Ce dernier présente une couche physique de radiocommunications que c'est l'alliance ZigBee lui permet d'ajouter à une couche réseau qui est responsable du routage des données dans un réseau de capteurs [9]. La figure ci-dessous nous montre les différents couches de l'alliance ZigBee et le standard 802.15.4.

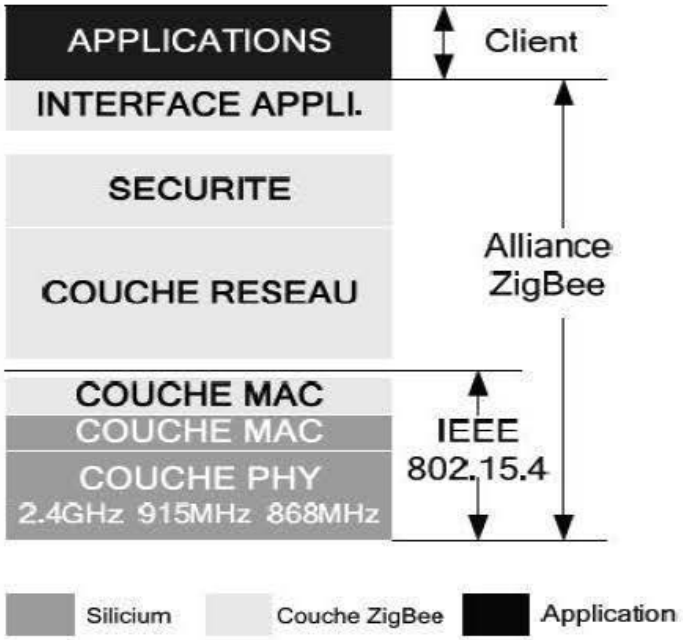



**Figure 2. 7:** Différence entre l'alliance ZigBee et le standard IEEE 802.15.4 [2]

### 2.5.2.1. La topologie réseau du standard ZigBee

Le standard 802.15.4, avec l'alliance ZigBee permet la création des réseaux maillés (mesh) grâce à un routage automatique. C'est la couche réseau ZigBee qui est responsable du routage des données dans un réseau de capteurs.

La figure 2.8 illustre l'architecture de la topologie des réseaux maillés avec un coordonnateur ZigBee. Celui-ci est responsable du traitement des données et aussi du routage et l'interconnexion avec les autres groupes. Il est branché à une alimentation directe. On a quelque chose d'identique pour le routeur ZigBee FFD (Full function device) qui permet aussi le routage sauf qu'il interconnecte les nœuds qui appartiennent à son groupe seulement. La dernière entité réseau dans ce type de topologie est l'entité réduite RFD (Reduce function device); elle ne permet pas le routage et son rôle est seulement de capter le signal et de l'envoyer directement au

coordonnateur le plus proche, ce qui lui permet une consommation d'énergie réduite significativement car elle est alimentée à base d'une batterie.

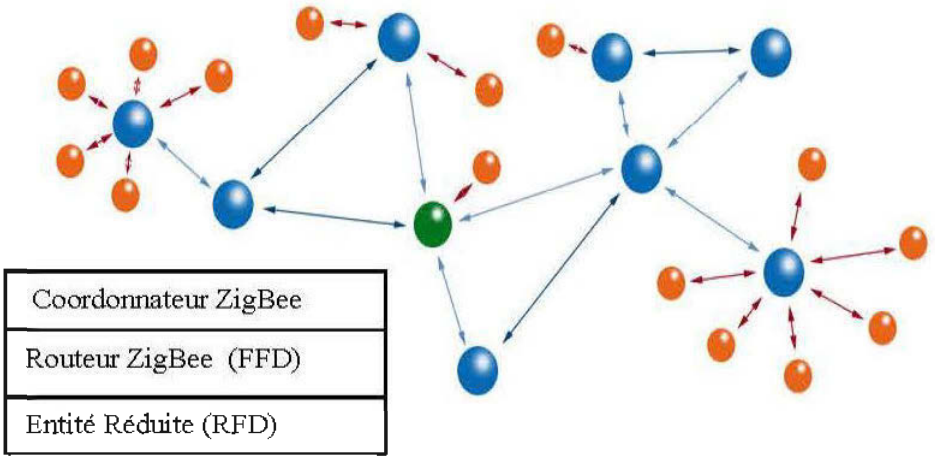



**Figure 2. 8:** Topologie Maillée

### 2.5.2.2. Le standard ZigBee par rapport aux autres standards de communication

La Figure 2.9 nous présente la position de divers standards sur une échelle de portée et de débit par rapport au standard ZigBee. On remarque que le WIFI, autorise un débit de données de 11 à 55 Mbps. Ce débit de données élevé contribue à des applications de transmission vidéo sans fil à partir d'un appareil à un téléviseur à proximité. Le Bluetooth, avec un débit de données de 1 à 3 Mbps, est utilisé dans la transmission de la voix de haute qualité dans les casques sans fil. ZigBee, avec un débit de données maximum de 250kbps, est classé comme un LR-WPAN. Ce standard envoi des données format texte ce qui lui permet d'avoir un faible débit, et c'est ce qui lui permet aussi de faire le processus transmission / réception avec une faible consommation d'énergie.

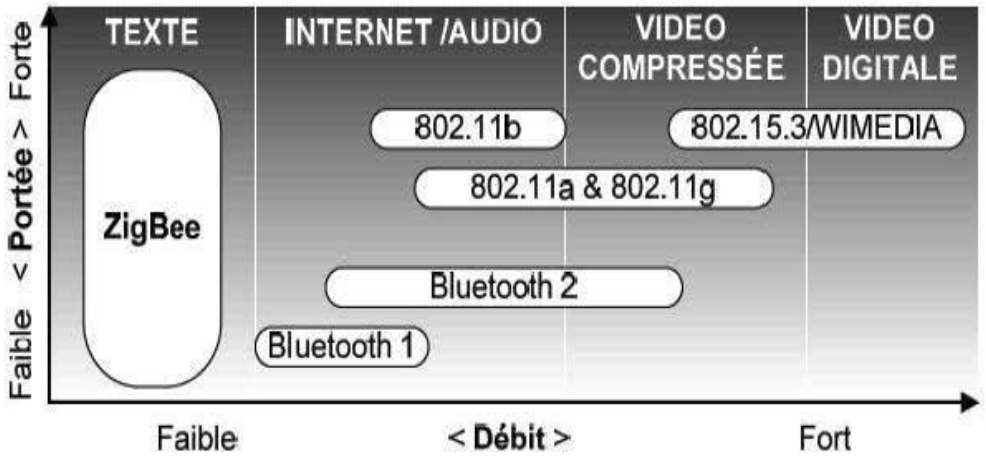



**Figure 2. 9:** Standard ZigBee par rapport aux autres standards sans fils [2]

## 2.6. Conclusion

Les réseaux de capteurs sans fil conçus pour être utilisé en sécurité possèdent plusieurs caractéristiques différentes par rapport aux autres réseaux sans fil traditionnels. Car ils ont été conçus spécifiquement pour la surveillance et le contrôle des zones rurales. Les études et les recherches [2, 4, 5, 6, 7, 8, 9] ont démontré que les réseaux de capteurs permettent une communication des entités réseaux à faible débit et faible coût. Les standards convenables pour ce genre de réseaux sont IEEE 802.15.4 et ZigBee.

Dans ce chapitre, on s'est intéressé à définir le standard IEEE 802.15.4 et aussi les raisons pour lesquelles il a été choisi pour les réseaux de capteurs en détaillant ces caractéristiques de la couche physique et de liaison. La dernière partie du chapitre a présenté les caractéristiques du standard ZigBee en incluant pour comparaison avec les autres technologies sans fils.

# CHAPITRE 3

# LA RADIO LOGICIELLE

## 3.1. Introduction

La technologie Radio Définie par Logiciel (SDR -Software Defined Radio) est une technologie qui a évoluée rapidement et qui reçoit la reconnaissance énorme. Elle suscite un intérêt très répandu dans l'industrie des télécommunications.

Au cours des dernières années, les systèmes radio analogiques ont été remplacés par des systèmes de radio numériques pour les applications radio différentes dans les espaces militaires, civils et commerciales. De plus, des modules matériels programmables sont de plus en plus utilisés dans les systèmes radio numériques à différents niveaux fonctionnels. La technologie SDR vise à tirer parti de ces modules matériels programmables pour construire une architecture ouverte du logiciel système basé sur la radio.

## 3.2. La différence entre la radio logicielle et la radio traditionnelle

La différence entre ces deux types de radio différents, c'est qu'il y'a plus de deux décennies, la logique de conception des architectures radios ne permettait pas de mettre l'accent sur la flexibilité des architectures développées. Les couches matérielles et logicielles des radios traditionnelles n'étaient pas conçues pour être facilement réutilisables et, également, n'étaient pas conçues pour être évolutives au fil du temps. En général, la radio était composée des chaines de modulation et

démodulation, une pour chaque standard. Des recherches et études [10] ont aidé à trouver une nouvelle technologie de transmission plus efficace et flexible. Cette dernière est la radio logicielle ou (Software Defined Radio). En résumé, le principe du SDR est de réaliser, de façon logicielle, le plus grand nombre possible de fonctionnalités de traitement de signal, notamment la modulation et la démodulation des signaux radios à travers une programmation logicielle.

### 3.2.1. Utilisation de la radio logicielle

On a pris comme exemple l'utilisation de la radio logicielle par le service militaire, mais il n'est pas le seul service ayant besoin de cette solution. De nombreux organismes, ont répondu à diverses catastrophes naturelles et artificielles dans le monde et les communications entre les différents groupes ont souvent été entravées par le fait que des différents systèmes de communication fonctionnent rarement l'un avec l'autre car il y'avait un manque de flexibilité dans les anciennes solutions de communications.

La SDR offre une solution idéale à ces dilemmes. Ainsi, une station de base centralisée déployée pourrait être utilisée pour recevoir les transmissions d'une seule agence et les rediffuser sur les fréquences des autres organismes d'intervention [11]. Si le système est reconfigurable, que les agences de nouvelles arrivent ou partent, le SDR peut être rapidement modifié pour accueillir les services requis. Lorsque la catastrophe est terminée, le système peut facilement être rangé et redéployé à une date ultérieure en cas de besoin.

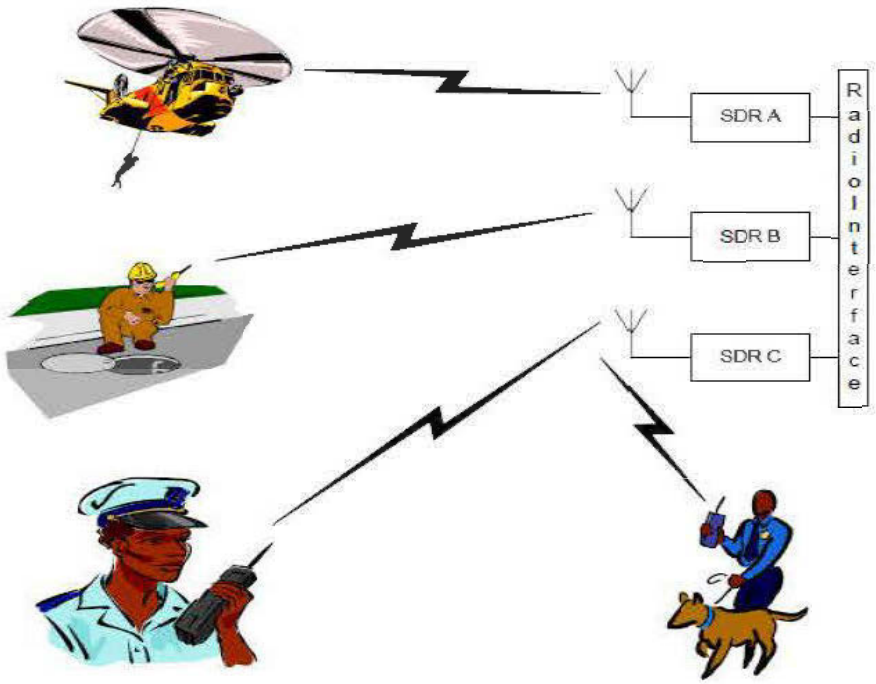



**Figure 3. 1:** utilisation du SDR [11]

### 3.2.2. Définition de la radio logicielle

La radio logicielle ou Software Defined Radio est une technologie de communication radio qui est basée sur une partie logicielle qu'on peut programmer pour définir des protocoles de communication sans fil au lieu d'implémentations câblées. De plus elle nous permet de faire des mises à jour automatiques de la bande de fréquence, du protocole d'interface avec le milieu hertzien et des fonctionnalités qui peuvent être mises à niveau avec le logiciel de téléchargement au lieu d'un remplacement complet du matériel. On notera que le SDR offre une solution efficace et sûre au problème de la construction multimode, multi-bandes et multi appareils de communication sans fil.

### 3.2.3. L'architecture de la radio logicielle

En général, les fonctionnalités des architectures radio sont déterminées principalement par le matériel et sont accompagnées d'une configuration par logiciel. Le matériel se compose d'amplificateurs, de filtres, de mélangeurs et d'oscillateurs et le logiciel est utilisé pour contrôler l'interface avec le réseau, l'adressage et le contrôle d'erreur. Puisque le matériel domine la conception, la mise à niveau d'une conception radio à travers le logiciel signifie essentiellement l'abandon complet de l'ancienne conception.

La solution radio logicielle, grâce à sa flexibilité en programmation, permet de fonctionner avec différentes formes d'ondes et les protocoles à travers le chargement dynamique de nouvelles formes d'ondes et de protocoles. Ces formes d'ondes et ces protocoles peuvent contenir un certain nombre de parties différentes, y compris des techniques de modulation, de sécurité et les caractéristiques de performance définies dans le logiciel dans le cadre de l'onde elle-même.

La figure 3.2 présente l'architecture d'une Radio Logicielle. On remarque que la conversion analogique numérique (A/N) s'effectue directement après l'antenne, le filtrage, l'étage d'amplification à faible bruit (LNA, Low noise Amplifier) et l'amplification de puissance (PA, Power Amplifier).

L'architecture qui est entièrement reprogrammable car le DSP qui fait le traitement de signal est directement relié au convertisseur délivrant le signal RF numérisé.

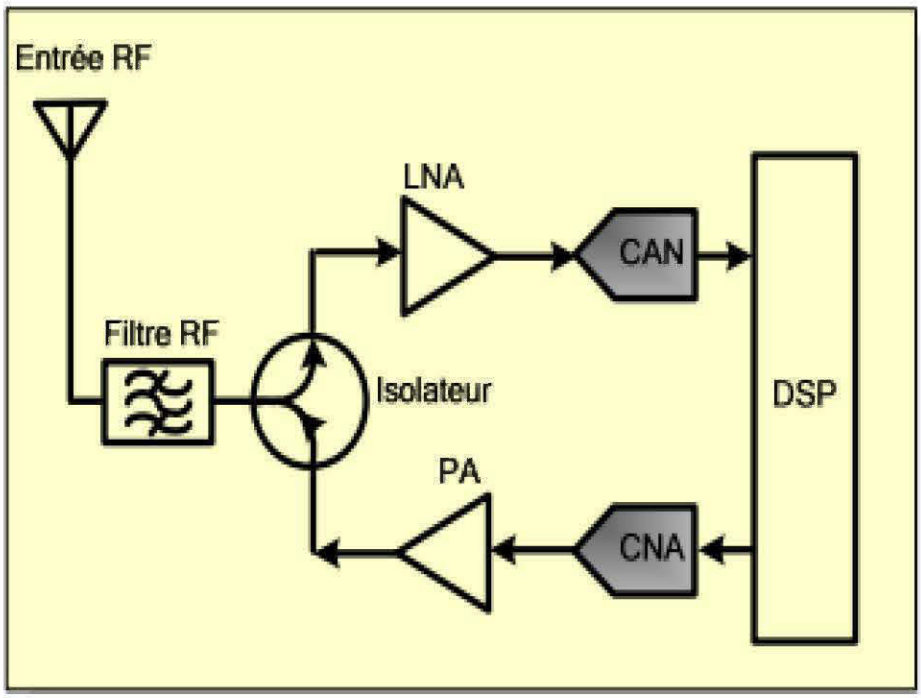



**Figure 3. 2:** L'architecture de la radio logicielle [12]

### 3.2.4. Avantages

Parmi les avantages de la radio logicielle, elle offre une solution efficace et sûre au problème de la communication multi-mode, multi-bandes, multi-porteuses de systèmes de communication sans fil. On donne ci-dessous une explication plus détaillée de chaque avantage.

#### 3.2.4.1. Multi-bande

Les architectures radio traditionnelles fonctionnent sur une seule bande ou sur une plage de fréquences fixe. Il existe de nombreuses applications où les fréquences multiples d'opérations sont souhaitées. Il s'agit notamment des communications

cellulaires, celles utilisées par les organismes gouvernementaux et non gouvernementaux, la norme est d'utiliser de multiples émissions radios, chacune conçue pour fonctionner dans une bande spécifiée. Un radio multi-bande a la capacité de fonctionner dans deux ou plusieurs bandes de manière séquentielle ou simultanée dans le cas d'une station de base où elles peuvent être être combinées à partir de liaison sur des bandes différentes.



### 3.2.4.2. Multi- porteuse

Un support multi-ou multi-canal radio a la capacité de fonctionner en même temps sur plus d'une fréquence à la fois. Cela peut être dans la même bande ou, dans le cas d'une radio-multi bande, dans deux bandes différentes en même temps. Le principe est de transmettre des données numériques en parallèle modulées sur un grand nombre de porteuses à bas débit. Dans un système conventionnel de transmission de données en série, les symboles sont transmis séquentiellement : le spectre de chaque donnée est autorisé à occuper toute la bande passante disponible.

### 3.2.4.3. Multi-mode

Multi-mode implique une capacité de traiter plusieurs types de normes. Des exemples de normes comme le AM, le FM, le GMSK ou le CDMA, mais est limité à chacune d'elles. Une SDR a la capacité de travailler avec de nombreuses normes différentes et être en reprogrammation permanente. Par conséquent, un meilleur terme que multi-mode, ce qui implique un nombre discret de modes, peut-être le mode variable, ce qui implique un mode continu changeant de fonctionnement continuellement. Ces modes peuvent être séquentiels ou simultanés dans le cas d'une radio multi-porteuse.

### 3.2.5. Contraintes

La radio logicielle se démarque lorsqu'il s'agit des contraintes de flexibilité, compacité et coût. En effet, le coté programmation de la radio logicielle est très flexible et fait en sorte que celle-ci est potentiellement plus sensible aux attaques au niveau réseau, ce qui peut causer des problèmes dans ses blocs programmables. De plus, la mutualisation des ressources matérielles entre les différents standard de communications sans fil implantés permet d'une part, de réduire le nombre de composants de circuits embarqués dans une radio et, d'autre part, de réduire le coût de fabrication d'une radio logicielle dont la partie matérielle est générique.

## 3.3. La solution logicielle GNU Radio

La GNU Radio est un ensemble de ressources disponibles à tous (toolkit open-source) créé par Blossom en 1998 [13, 14]. Couplé avec l'équipement matériel USRP, cette ressource permet une plate-forme complète pour la construction d'un Software Defined Radio.

Les systèmes d'exploitation recommandés pour la construction du GNU Radio est le Linux, mais il peut également être construit sur MS Windows en utilisant l'un des environnements « Linux-like » tels que Cygwin ou Min GW / MSYS, ainsi que sur MAC OS et NetBSD. La plupart des applications GNU Radio sont écrites en Python, alors que C++ est utilisé pour la mise en œuvre des blocs de traitement du signal.

Les commandes Python utilisées pour contrôler l'ensemble des USRPs sont définis par les paramètres du logiciel, tels que la puissance d'émission, le gain, la fréquence, l'antenne sélectionnée, etc.., dont certains peuvent être modifiés alors que l'application est exécutée.

La solution GNU Radio est construite principalement sur les blocs de traitement du signal. Les blocs sont structurés de manière à avoir un certain nombre de ports d'entrée et de sortie, composés de petits composants de traitement du signal. Lorsque les blocs sont connectés de façon appropriée, dans ce cas on peut dire que le graphe de flux est fait. Les blocs GNU Radio peuvent être classés en tant que sources et filtres.

Comme le montre la figure 3.3, un certain nombre de blocs est responsable du traitement de signal pour différentes techniques de modulation et de démodulation, du filtrage, des indicateurs de signal etc... Ils sont tous intégrés au sein de GNU Radio qui aura besoin d'un USRP pour recevoir u transmettre des ondes radios réelles.

La solution logicielle GNU Radio est organisée selon une structure à deux niveaux. Elle permet des performances qui sont cruciales pour avoir des blocs de traitement du signal mis en œuvre en C++, tandis que l'organisation de niveau supérieur, le raccordement et le collage des blocs de signaux ensemble se fait en utilisant le langage de programmation Python.

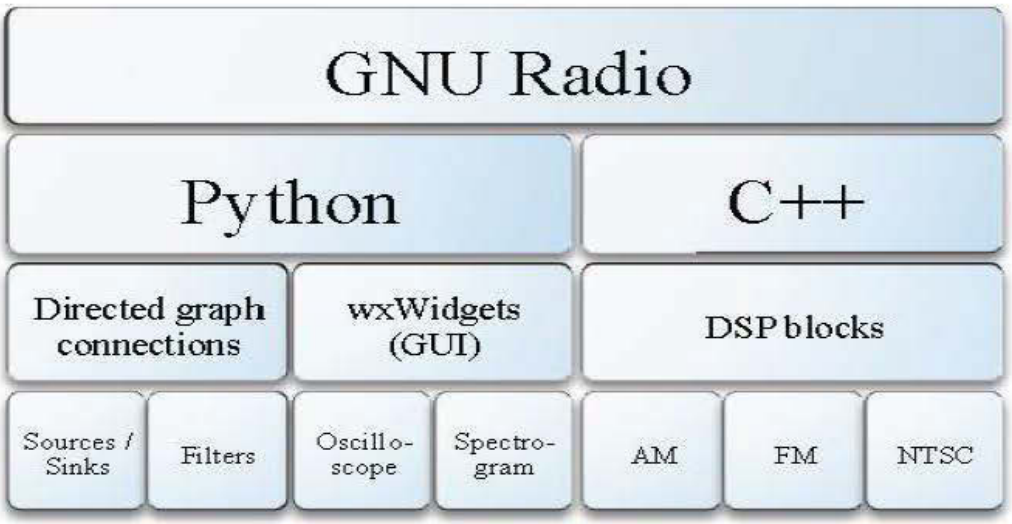



**Figure 3. 3:** Présente les blocs de GNU Radio [13]

### 3.4. Python

Python est un langage interprété, interactif, orienté objet (POO). C'est un langage script connu pour sa simplicité, facile à utiliser avec la syntaxe appropriée. C'est un langage écrit en C ; il peut donc fonctionner sur Windows, Unix / Linux, Mac OS X, Java, etc… en raison de sa nature orientée objet. Les programmes écrits en Python peuvent se spécialiser dans des classes écrites dans d'autres langages de programmation orientés objet, tels que le C + +. Cette fonctionnalité est souvent utilisée dans GNU Radio comme l'illustre la figure 3.3 où les blocs de traitement du signal sont écrits en C + + et Python et ils sont utilisés pour "coller" l'ensemble et contrôler le flux de données numériques.

### 3.5. Projet UCLA ZigBee

Pendant nos expérimentations et tests, afin de pouvoir capter les paquets ZigBee avec l'USRP, nous avons tout d'abord regardé les solutions qu'offraient GNU Radio. Nous nous sommes vite aperçus que les blocs GNU Radio destinés au 802.15.4 étaient inexistants.

Après un bon nombre de recherches, nous avons trouvé une nouvelle librairie développée pour GNU Radio. Cette librairie se nomme UCLA ZigBee qui a été spécialement développée dans le but d'interagir avec les puces radios CC1000 et CC2420 utilisant le standard IEEE 802.15.14 [14]. Le composant CC2420 est justement présent dans le matériel que nous avions préparé pour notre environnement de test. Les applications offertes et les tests dont dispose cette librairie sont "open source", on peut donc modifier et ajouter des codes ou des blocs pour résoudre d'autres problématiques liées à ce sujet. Il y a un ensemble des scripts programmés avec le langage Python dans cette librairie, leur rôle étant d'effectuer des tests de réception et d'émission de paquets entre les équipements matériels à base du standard

IEEE 802.15.4. Chaque bloc est responsable soit de la chaine transmission, réception, ou bien d'une opération de traitement de signal comme la modulation, démodulation, filtrage etc.....

### 3.6. Conclusion

Le développement de services de télécommunications a été à l'origine de l'apparition du concept de radio logicielle. En première partie, on a expliqué son importance par rapport à l'ancienne technologie de transmission qu'est la radio traditionnelle. On a aussi présenté le besoin de son utilisation ainsi que ses avantages, ses contraintes et, par la suite, la plateforme GNU Radio sur laquelle notre travail se base en détaillant son rôle ainsi le langage Python, avec lequel les blocs ont été programmés. La dernière partie de ce chapitre s'est intéressée au projet UCLA ZigBee qui est parmi les rares travaux qui sont fait pour l'interconnexion de capteurs à base du standard 802.15.4 et GNU Radio.

# CHAPITRE 4

# RELAIS DE RÉSEAU DE CAPTEURS À BASE DE LA SOLUTION RADIO LOGICIELLE

## 4.1. Introduction

Après une revue d'ensemble de l'état de l'art, ce chapitre présente toutes les solutions requises pour la préparation de l'environnement où on va faire les expérimentations et tests pour résoudre la problématique. En premier lieu, on rappelle la problématique, et par la suite, on présente l'architecture de la solution et le matériel avec lequel on réalisera le relais qui est l'USRP ainsi qu'une explication du relais. La dernière partie consiste à détailler les étapes de l'implémentation qui ont eu lieu pendant les expérimentations et les tests.

## 4.2. L 'architecture de la solution proposée

L'objectif principal dans ce travail est de pouvoir relayer deux types de réseaux hétérogènes, dans notre cas un réseau de capteurs avec un réseau de stations de base. Les recherches ont montré que c'est possible en utilisant une radio logicielle. Le but, comme le montre la figure 4.1 ci-dessous, est de pourvoir capturer le signal et de le relayer pour le traitement au niveau du coordinateur. Les différents utilisateurs ou intervenants peuvent réagir ou prendre les décisions dans les moments convenables, surtout dans le cas des zones rurales, là où il est toujours difficile de savoir ce qui se passe. Dans notre environnement de tests, pour les stations de bases, on utilise le matériel USRP, pour le capteur et un capteur ZigBee pour le

coordonnateur ce rôle étant confié à un ordinateur .La partie qui se suit définit et détaille les matériaux utilisés dans notre architecture.

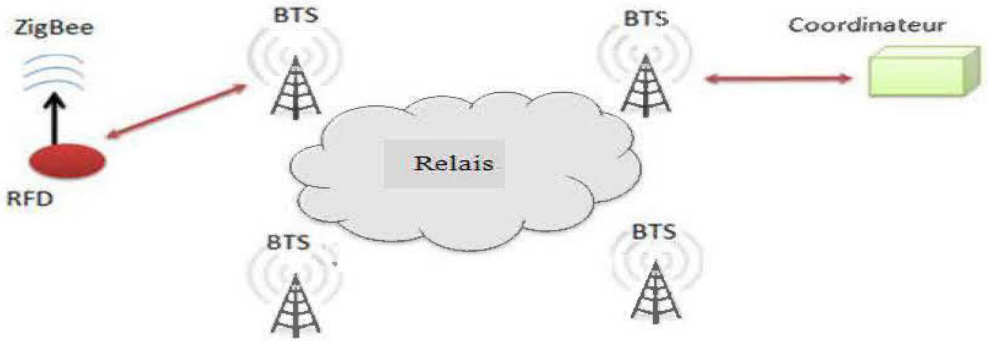



**Figure 4. 1:** Problématique

## 4.3. L'USRP

L'USRP (Universal Software Radio Peripheral) est un équipement développé par la compagnie Ettus Research [16]. Ce matériel comporte un boîtier contenant une carte mère munie d'un DSP sur laquelle on raccorde des cartes filles émettrices et réceptrices [17].L'USRP est un matériel à faible coût, qui permet de concevoir rapidement et mettre en œuvre un traitement de signal puissant, flexible adaptés aux systèmes radio logiciels.

On peut mettre sur l'USRP des cartes filles couvrant une large gamme de fréquences, ce qui permet d'obtenir un logiciel radio qui peut être mis en marche rapidement. Il suffit de télécharger GNU Radio avec une ouverture complète radio logicielle source et un package de traitement du signal, et l'USRP est prêt à utiliser. Une fois qu'on installe le logiciel et que l'on a branché à un ordinateur hôte, il est prêt à transmettre et à recevoir une variété pratiquement illimitée des signaux.

La vraie valeur de l'USRP est qu'elle permet aux concepteurs de créer, avec un petit budget et avec un minimum d'effort, une combinaison puissante de ce matériel souple et de le faire fonctionner avec des logiciels open-source qui permet d'avoir une plate-forme pour le développement de la radio logicielle souhaitée.


**Figure 4.2:** Vue externe du boitier USRP [16]

### 4.3.1. Les caractéristiques de l'USRP

Comme le montre la figure 4.3, la carte circuit contient quatre convertisseurs Analogique/Numérique avec une bande de signal de 64MHz -12 bits et quatre autres convertisseurs Numérique/Analogique avec une bande de signal de 128MHz - 14 bits. Il y a un support pour les cartes filles qu'on utilise pour la transmission et la réception RF qui couvrent tout le spectre radio de 0 Hz à 2.9 GHz. Les opérations à haut débit du signal en bande de base s'effectuent par le FPGA grâce au bus USB qui relie l'USRP à l'ordinateur et qui atteint un débit de 480 Mbits/s avec une largeur de bande de signal qui peut aller jusqu'à16MHz.

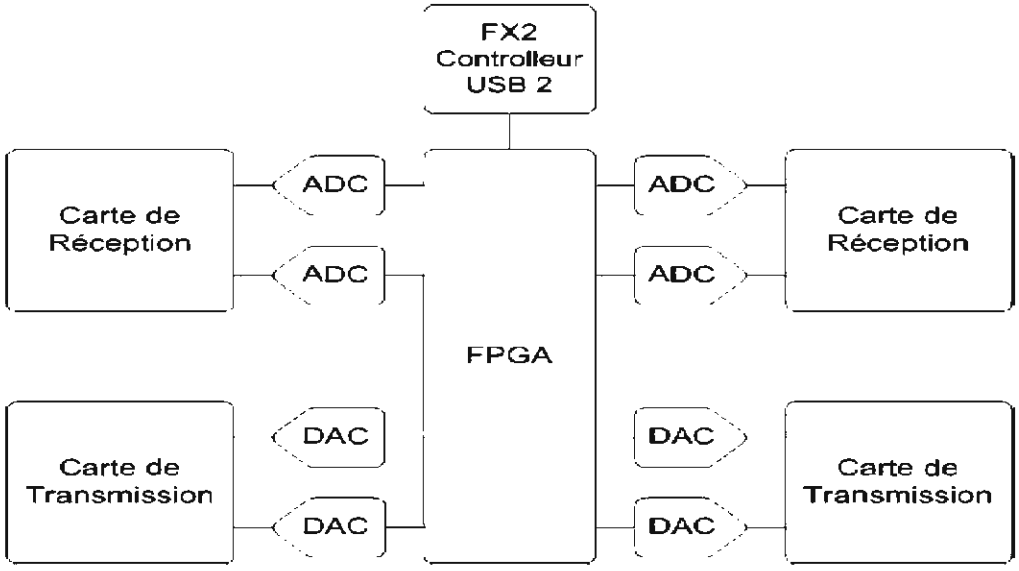



**Figure 4.3:** Schéma de l'architecture interne de l'USRP [17]

#### 4.3.1.1. Les cartes filles RF utilisées dans l'USRP

Dans le cadre de notre de projet, nous avons choisi les cartes filles les mieux adaptées. Dans notre cas, nous avons préconisé la carte RFX2400 qui permet l'émission et la réception [17] et sa bande de fréquence d'utilisation va de 2.3 à 2.9 GHz. Cette configuration nous offre donc la possibilité de travailler sur des liaisons 802.15.4 et peut aussi servir pour tout autre standard utilisant cette bande de fréquence.

Les cartes filles montées sur l'USRP fournissent une entrée RF (RF front-end) intégrée. Une grande variété des cartes filles disponibles permet d'utiliser des fréquences différentes pour une large gamme d'applications, selon le type de la communication sans fil. L'USRP accueille jusqu'à deux RF cartes filles en transmission et jusqu'à deux autres en réception pour tout type de signal RF. On donne ci-dessous quelques types de cartes filles utilisées sur l'USRP.

- ✓ RFX400: utilise les fréquences entre 400-500 MHz en transmission et en réception
- ✓ RFX900: utilise les fréquences entre 750-1050 MHz en transmission et en réception
- ✓ RFX1200: utilise les fréquences entre1150-1450 MHz en transmission et en réception
- ✓ RFX1800: utilise les fréquences entre 1.5-2.1 GHz en transmission et en réception
- ✓ RFX2400: utilise les fréquences entre 2.3-2.9 GHz en transmission et en réception

La figure 4.4 nous montre une image des cartes filles RFX 2400 qu'on a utilisée dans notre travail pratique, elle peut être utilisée pour tout le standard qui utilise la bande de fréquence 2.4GHz.

Sur le même USRP, on peut utiliser juste une seule carte pour faire une transmission/réception à la fois où bien on peut utiliser deux cartes une pour la transmission et une autre pour la réception selon notre environnement de test et selon notre besoin matériel pour le travail pratique souhaité.

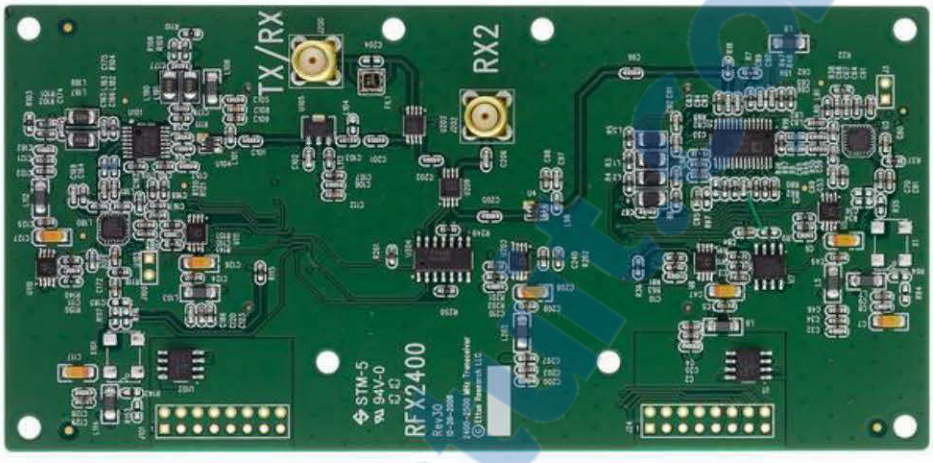



**Figure 4.4 :** Carte fille RFX 2400 [19]

### 4.4. Le capteur utilisé dans l'architecture de test

Dans la solution retenue pour préparer l'environnement de test, on n'était pas obligé de travailler avec un capteur précis. Nous allons donc travailler sur le relais entre les réseaux de capteurs et les stations de bases que sont les USRP.

L'essentiel est d'avoir un capteur qui fonctionne dans la bande 2.4 GHz et sur le standard Zigbee/IEEE 802.15.4. On a alors utilisé le capteur Silicon qui existait déjà parmi le matériel disponible au LRTCS.

La figure 4.5 nous montre un capteur Silicon qui est doté d'un microcontrôleur C8051F121 et d'une antenne de transmission Chipcon CC2420. Il s'agit d'une véritable mono-puce de 2,4 GHz fonctionnant sur IEEE 802.15.4 conforme à l'émetteur-récepteur RF à faible puissance conçu pour les applications sans fil et à

basse tension. Le microcontrôleur comprend un processeur 8051 à 100 MIPS, une mémoire flash de 128 octets et une mémoire vive de 8 koctets.

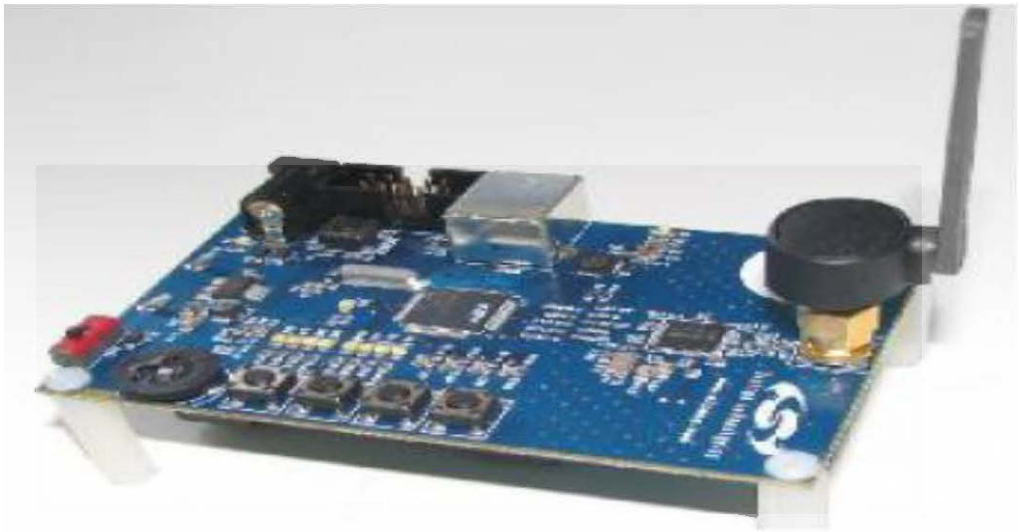


**Figure 4. 5:** Capteur Silicon 2.4 GHz [21]

### 4.5. Schéma- bloc proposé pour la réalisation du relais

Comme on a mentionné au préalable que la problématique majeure de notre projet est de relayer le signal entre un réseau de stations de bases et un autre de capteurs. En se basant sur le projet Ucla ZigBee, comme il 'est open source, on a pu modifier et ajouter de nouveaux blocs de programmes afin de résoudre la problématique.

La figure 4.6 ci-dessous nous montre le schéma-bloc proposé dans notre architecture.

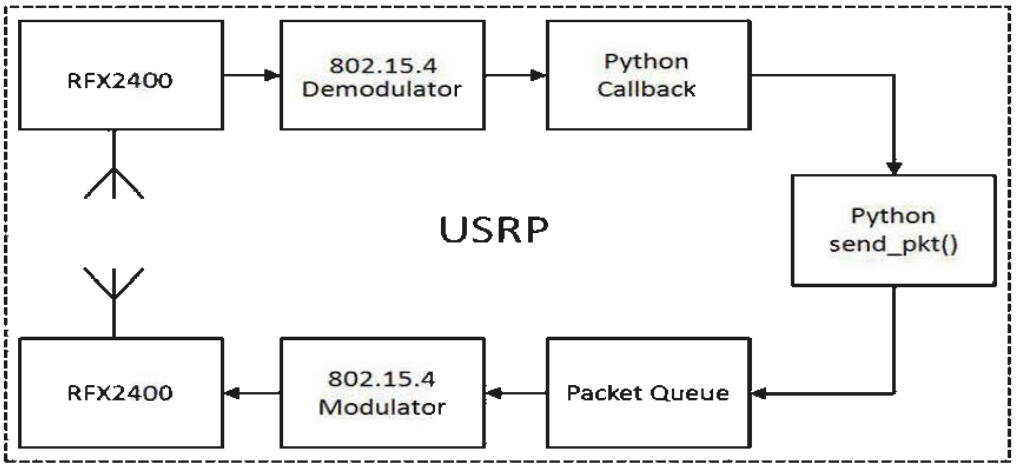



**Figure 4.6 :** Schéma des blocs qui sont responsables de la communication des réseaux decapteurs au niveau de l'USRP

Selon le schéma, les données reçues par la carte fille RFX 2400 sont acheminées vers le bloc démodulateur 802.15.4 qui est implémenté en Python et C + +. Le bloc démodulateur 802.15.4 est constitué d'un démodulateur FM, corrélateur et une file d'attente de paquets. Ces deux derniers détectent la séquence des données reçues, la bande de la synchronisation et le préambule. Les paquets avec les informations d'en-tête sont ensuite assemblés dans une file de messages. Une fonction de rappel Python (Python Callback) détecte les informations de l'en-tête de chaque paquet et transmet les données utiles au bloc send_pkt () où la charge utile est recueillie. Après, les données utiles sont envoyées au Packet Queue et placées dans une file d'attente de messages. L'en-tête 802.15.4 MAC et le pied de page sont ajoutés au paquet et suivi par l'ajout de l'en-tête de la couche de synchronisation physique. Chaque image est passée au bloc modulateur 802.15.4 où la charge utile est convertie à partir d'une séquence d'octets à puce. Les puces sont converties en symboles, ensuite modulées par la technique O-QPSK avant de les envoyer à la carte fille RFX 2400 qui se charge de transmettre le signal à l'antenne.

## 4.6. Conclusion

Dans ce chapitre, on a présenté les propositions pour l'architecture et le matériel utilisé dans l'environnement de test. La deuxième partie de ce chapitre a présenté les schéma-blocs proposés pour résoudre le problème du relais entre deux types différents de réseau à base de la radio logicielle. Le chapitre suivant nous présentera l'environnement ainsi les résultats et les mesures des expérimentations réalisées.

# CHAPITRE 5

# EXPÉRIMENTATION ET RÉSULTATS

## 5.1. Introduction

Cette partie du mémoire présente l'environnement expérimental, les tests réalisés et les résultats qui démontrent que le défi de la problématique a été relevé. Avant chaque test on donne une explication de la procédure utilisée et du pourquoi de chaque étape, afin de bien démontrer que la communication entre les capteurs à base d'un signal ZigBee à travers la radio logicielle est faisable et que nous l'avons réalisé. Pour bien réussir les tests, les notions élaborées aux chapitres précédents ont significativement aidé à éviter beaucoup d'obstacles et ont permis de choisir des équipements matériels et logiciels d'architecture simple et d'une flexibilité importante, ce qui a aidé à réaliser les expérimentations requises pour obtenir des résultats probants.

## 5.2. L'environnement de test

La figure 5.1 illustre l'environnement de test utilisé pour la résolution du relais des réseaux hétérogènes basée sur une radio logicielle. Tel que l'on peut voir sur cette figure, on a un ordinateur portable qui est branché à deux stations USRP qui vont jouer le rôle des stations de bases. Le système d'exploitation utilisé est l'Unix (Ubunto) sur lequel on a implémenté la solution radio logicielle GNU Radio. Par la suite, on utilise le projet Ucla Zigbee qui contient des librairies écrites en C/C++ sous forme de blocs responsables de la communication des réseaux de capteurs à

base de la radio logicielle. Notre contribution a permis d'apporter des modifications et de créer des blocs en programmation C/C++ pour pouvoir répondre à la problématique selon nos objectifs.

De l'autre côté, il y'a un ordinateur de bureau avec un système d'exploitation Windows XP. Le besoin de celui-ci était uniquement pour pouvoir utiliser l'interface graphique des deux capteurs à l'étude et vérifier qu'ils échangent vraiment les messages entre eux.

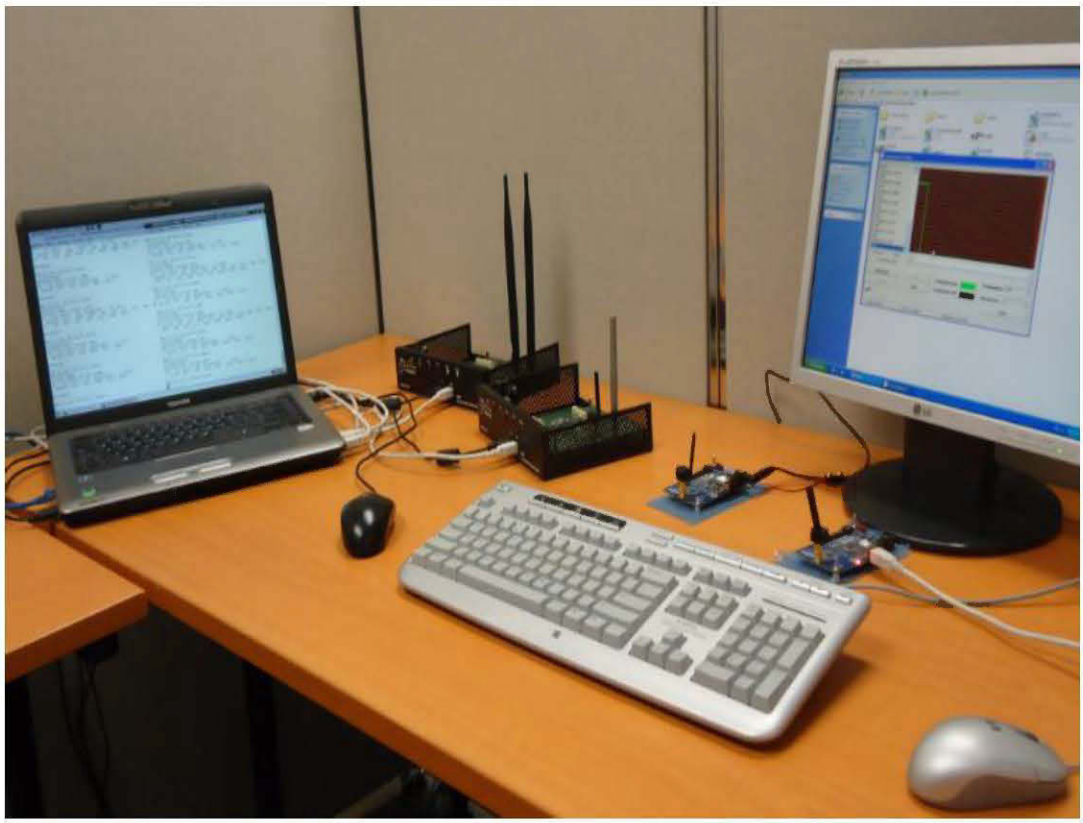


**Figure 5. 1 :** Environnement de test

### 5.3. Les tests et les expérimentations

Dans cette partie, qui est l'une des parties les plus importantes de ce travail, on donne les schémas des tests réalisés et les résultats très significatifs obtenus. On présentera également en détails les deux tests qui ont été faits et qui démontrent que nous avons obtenu une solution pour le problème à l'étude.

#### 5.3.1. Test réalisé entre deux USRPs

Ici, on a testé la communication sans fil à base de signal ZigBee entre deux USRPs, ce qui a nous a permis de s'assurer que le relais entre les stations de bases fonctionne bien, Le scénario de test est donné à la figure 5.2 .

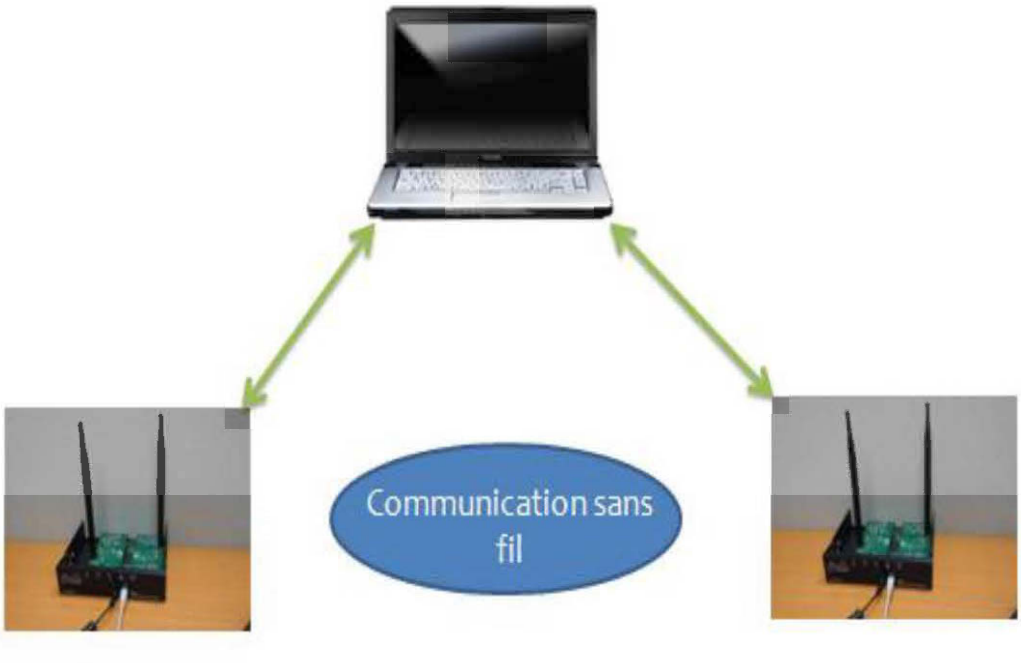



**Figure 5. 2 :** Communication sans fils à base du signal ZigBee entre deux USRPs

#### 5.3.1.1. Données reliées au niveau de l'USRP

On montre ci-dessous que le signal basé sur ZigBee et qui relie, les données à travers l'USRP. À l'exécution, on précise le choix de l'USRP, la carte fille qu'on va choisir selon la technologie et puis on précise la fréquence d'envoie qui est 450 MHz et la fréquence de réception qui est 2.4 GHz.

Il y a une grande différence entre les deux fréquences de réception et d'émission car le but était de prouver que l'USRP est capable de faire ce genre de relais. Le débit des données, qu'on a précisé est de 500 kbits, et cette partie de test nous a permis de s'assurer que le relais au niveau de l'USRP a été bien réussi.

Après avoir précisé les étapes à faire pour réussir le relais des données au niveau de l'URSP .Dans ce test, la majorité du travail s'est produit du côté de la programmation par le langage Python (un langage de programmation basé sur les langages C et C++). Toutes les modifications concernant la bande de fréquence, le gain de l'antenne et le débit de données se sont fait dans le fichier CC2420_relaytest.py, Et leur exécution pendant le test se fait au niveau du terminal Ubunto par des commandes.

Les trois premières lignes de la figure présente la partie exécution, et les autres lignes montrent la partie du signal capturé dans l'USRP pour ce qui a permis la réalisation du relais. Pour cela on a dans les résultats obtenus toutes les informations sur les paquets retransmis comme la confirmation de la réception, numéro de paquets, séquence des paquets et l'affichage des données des paquets qui sont en hexadécimal.

```
karim@karim-laptop:~/Téléchargements/ucla_zigbee_phy/src/examples$
./cc2420_relaytest.py -w 0 -r 500000 -R B -T A -t 450000000 -c
2400000000
Using TX d'board A: Flex 400 Tx MIMO B
cordic_freq = 2.4G
data_rate = 500k
samples_per_symbol =  2
usrp_decim =  64
fs =  1M
Using RX d'board B: Flex 2400 Rx MIMO B
>>> gr_fir_fff: using SSE
802_15_4_pkt: waiting for packet
received packet
checksum: 16629, received: 16629
ok =  True  pktno =  229  len(payload) =   22  1/1
  payload: ['0x1', '0x0', '0xe5', '0xff', '0xff', '0xff', '0xff',
'0x10', '0x0', '0x10', '0x0', '0x1', '0x80', '0x80', '0xff',
'0xff', '0x10', '0x0', '0x20', '0x0', '0xf5', '0x40']
Retransmit
 ------------------------
802_15_4_pkt: waiting for packet
received packet
checksum: 16629, received: 16629
ok =  True  pktno =  229  len(payload) =   22  2/2
  payload: ['0x1', '0x0', '
0xe5', '0xff', '0xff', '0xff', '0xff', '0x10', '0x0', '0x10',
'0x0', '0x1', '0x80', '0x80', '0xff', '0xff', '0x10', '0x0',
'0x20', '0x0', '0xf5', '0x40']
Retransmit
 ------------------------
802_15_4_pkt: waiting for packet
received packet
```

**Figure 5. 3 :** Données reliées au niveau de l'USRP

### 5.3.1.2. Données reçues au niveau de l'USRP

La figure ci-dessous nous montre la réception du signal basé sur ZigBee au niveau de l'USRP. On a utilisé la même fréquence pour la réception et l'émission du signal. Le débit des données qu'on a précisé est le même du signal reçu, soit 500 kbits et cette partie de test nous a permis de se rassurer que la réception des données au niveau de l'USRP a été bien faite.

Des modifications ont eu lieu concernant la bande de fréquence, le gain de l'antenne et le débit de données se sont fait dans le fichier CC2420_rxtest.py, Et leur exécution pendant le test se fait par des commandes au niveau du terminal Ubunto qui est dédié à la réception du signal.

Les trois premières lignes présentent la partie exécution pour recevoir le signal, et les autres lignes montrent la partie du signal reçu dans l'USRP. Pour cela on a dans les résultats obtenus toutes les informations sur les paquets reçus comme la confirmation de la réception, numéro de paquets, séquence des paquets et l'affichage des données des paquets qui sont en hexadécimal

```
karim@karim-
laptop:~/Téléchargements/ucla_zigbee_phy/src/examples$ sudo
./cc2420_rxtest.py -g 40 -R B -r 500000 -c 450000000
cordic_freq = 450M
data_rate =  500k
samples_per_symbol =  2
usrp_decim =  64
fs =  1M
Using RX d'board B: Flex 400 Rx MIMO B
freq=450000000.0
>>> gr_fir_fff: using SSE
802_15_4_pkt: waiting for packet
uOuOreceived packet
```

```
 checksum: 16629, received: 16629
 ok =  True  pktno =  256  len(payload) =   22  1/1
   payload: ['0x1', '0x0', '0xe5', '0xff', '0xff', '0xff',
'0xff', '0x10', '0x0', '0x10', '0x0', '0x1', '0x80', '0x80',
'0xff', '0xff', '0x10', '0x0', '0x20', '0x0', '0xf5', '0x40']
  ------------------------
 802_15_4_pkt: waiting for packet
 uOreceived packet
 checksum: 16629, received: 16629
 ok =  True  pktno =  256  len(payload) =   22  2/2
 payload: ['0x1', '0x0', '0xe5', '0xff', '0xff', '0xff',
'0xff', '0x10', '0x0', '0x10', '0x0', '0x1', '0x80', '0x80',
'0xff', '0xff', '0x10', '0x0', '0x20', '0x0', '0xf5', '0x40']
  ------------------------
 802_15_4_pkt: waiting for packet
 received packet
 checksum: 16629, received: 16629
 ok =  True  pktno =  256  len(payload) =   22  3/3
   payload: ['0x1', '0x0', '0xe5', '0xff', '0xff', '0xff',
'0xff', '0x10', '0x0', '0x10', '0x0', '0x1', '0x80', '0x80',
'0xff', '0xff', '0x10', '0x0', '0x20', '0x0', '0xf5', '0x40']
  ------------------------
 802_15_4_pkt: waiting for packet
 received packet
 checksum: 16629, received: 16629
 ok =  True  pktno =  256  len(payload) =   22  4/4
```

**Figure 5. 4 :** Données reçues au niveau de l'USRP

### 5.3.2. Test réalisé entre l'USRP et un réseau de capteurs ZigBee

Dans cette partie, on fait le relais des données entre les senseurs et le coordinateur à travers les USRP, ce qui répond au problème fondamental du projet, soit la connexion entre deux réseaux hétérogènes à base du signal ZigBee. La figure 5.5 nous montre le scénario de test :

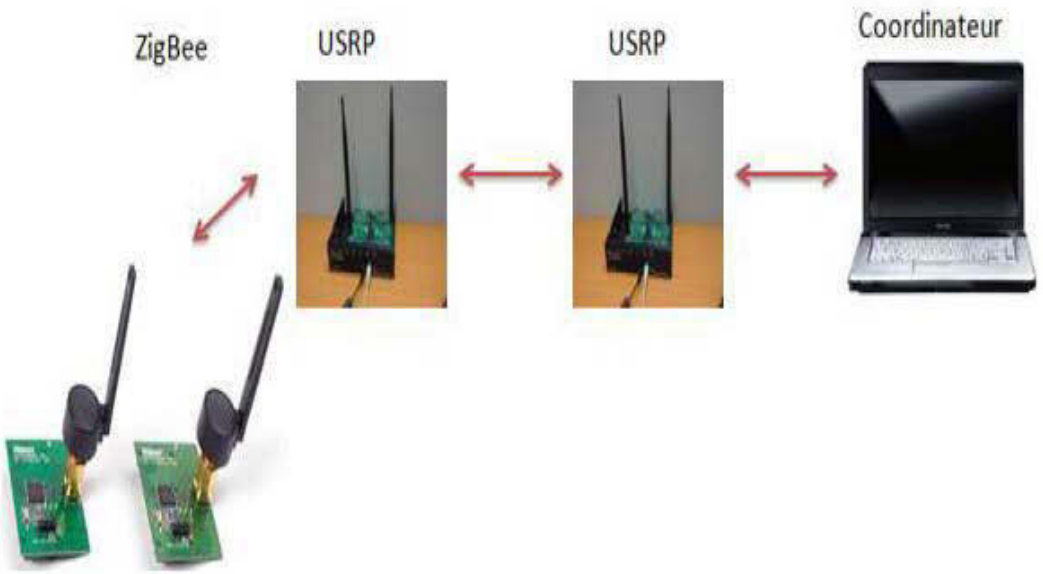



**Figure 5. 5 :** Relais entre un réseau d'USRPs et un réseau de capteurs

La figure 5.6 ci-dessous nous montre les messages envoyés entre deux capteurs ZigBee, le type de données envoyées, dans ce cas la détection de la température. On remarque sur cette figure qu'il y'a une ligne de couleur verte qui indique l'échange entre ces deux capteurs, avec la date du côté gauche et l'heure de l'échange des messages.

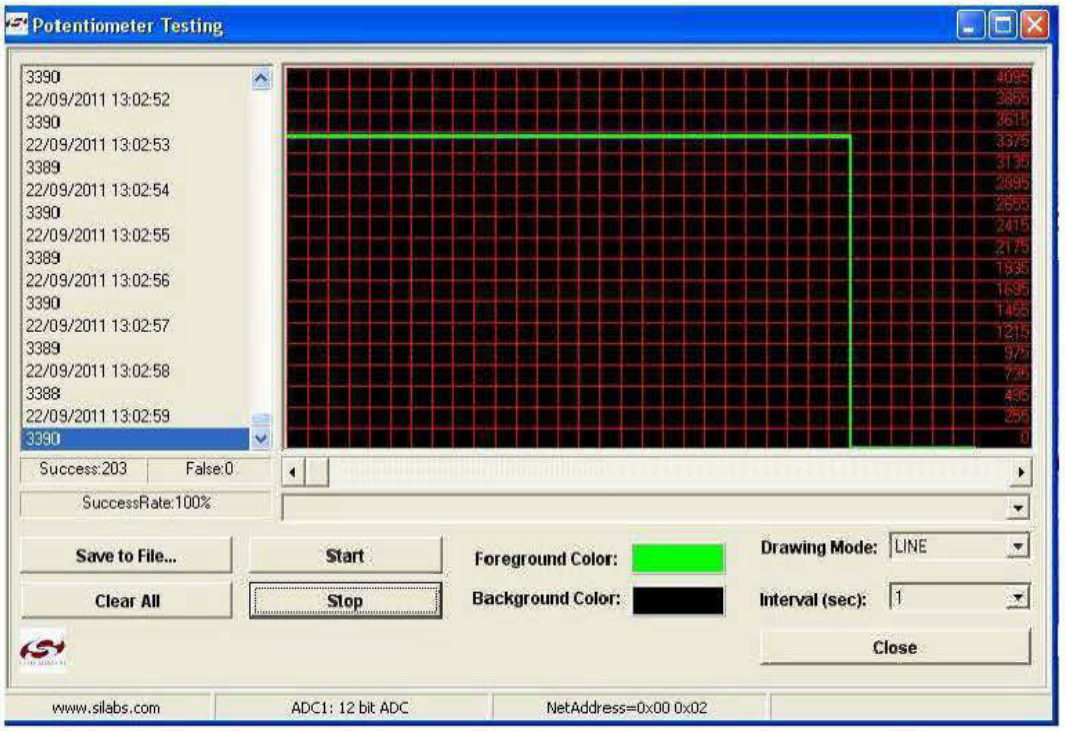



**Figure 5. 6 :** Communication entre les capteurs sur la bande de fréquence 2.4 GHz

### 5.3.2.1. Réception du signal entre les deux capteurs reliés aux autres USRPs

On explique ici que le signal envoyé par le capteur a bel et bien été reçu par l'USRP qui est lui-même relié à une autre USRP. Pour cela, on a adapté la fréquence et le débit de données envoyées en ce qui est compatible avec le standard ZigBee. Le débit utilisé est de 250 kb/s et on a travaillé avec les fréquences de la bande 2.4 GHz.

Ensuite, d'après les commandes sur le terminal, on a donné un gain aux antennes MIMO qui les rend capables d'émettre et recevoir et ce afin de bien réaliser le test et d'avoir une communication réussie.

Comme dans les tests qui précèdent celui-là, la majorité du travail s'est produit du côté de la programmation par le langage Python. Sauf qu'ici on a deux types de matériels différents, l'USRP et le capteur. Alors l'USRP son rôle était de recevoir le signal qui provient du capteur et le relayer. Pour résoudre cette problématique, le code du fichier CC2420_relaytest.py a subis plusieurs modifications et aussi plusieurs tests ont eu lieu pour que l'USRP soit capable de relayer n'importe quel signal de type Standard IEEE 802.15.4.

La figure suivante nous présente dans les trois premières lignes la partie exécution du relais, et les autres lignes montrent la partie du signal capturé par l'USRP ce qui va permettre le relais. Pour cela on a dans les résultats obtenus toutes les informations sur les paquets retransmis comme la confirmation de la réception, numéro de paquets, séquence des paquets et l'affichage des données des paquets qui **sont** en hexadécimal.

```
karim@karim-laptop:~/Téléchargements/ucla_zigbee_phy/src/examples$
./cc2420_relaytest.py -w 0 -R A -T A -t 2430000000 -c 2450000000 -r
250000
Using TX d'board A: Flex 2400 Tx MIMO B
cordic_freq = 2.45G
data_rate =  250k
samples_per_symbol =  2
usrp_decim =  16
fs =  500k
Using RX d'board A: Flex 2400 Rx MIMO B
>>> gr_fir_fff: using SSE
802_15_4_pkt: waiting for packet
```

```
received packet
checksum: 20755, received: 20755
ok =  True  pktno =   72  len(payload) =   24  1/1
  payload: ['0x41', '0x88', '0x48', '0x0', '0x0', '0xff', '0xff',
'0x0', '0x0', '0x40', '0xf0', '0x0', '0x0', '0xff', '0x0', '0x0',
'0x4', '0x0', '0xf', '0x0', '0x1', '0x1', '0x13', '0x51']

Retransmit
 ------------------------
802_15_4_pkt: waiting for packet
received packet
checksum: 50471, received: 50471
ok =  True  pktno =   67  len(payload) =    5  2/2
  payload: ['0x2', '0x0', '0x43', '0x27', '0xc5']
Retransmit
 ------------------------
802_15_4_pkt: waiting for packet
received packet
checksum: 45430, received: 45430
ok =  True  pktno =   74  len(payload) =   18  3/3
  payload: ['0x61', '0x88', '0x4a', '0x0', '0x0', '0x4', '0x0',
'0x0', '0x0', '0x28', '0xf2', '0x0', '0x0', '0x4', '0x81', '0x0',
'0x76', '0xb1']

Retransmit
 ------------------------
802_15_4_pkt: waiting for packet
```

**Figure 5. 7 :** Réception du signal entre les deux capteurs reliés aux autres USRPs

La figure ci-dessous montre que le signal reçu à l'USRP, est relié au coordonnateur pour faire le traitement. Pour cela, on a adapté la fréquence et le débit de données envoyées en conformité avec le standard ZigBee et, afin d'avoir une bonne adaptation, soit un débit de 250 kb/s et une fréquence dans la bande 2.4 GHz. On l'a adapté cette fréquence à celle de l'émetteur pour bien recevoir les données et, ensuite d'après les commandes sur le terminal, et d'une façon similaire aux autres tests, on a donné un gain aux antennes MIMO qui les rend capable d'émettre et de recevoir afin de bien réalisé le test et d'avoir une communication couronnée de succès. Les premières lignes du résultat présentent la partie exécution de la réception, et les autres lignes montrent la partie du signal reçu par l'USRP. Pour cela on a dans les résultats obtenus, toutes les informations sur les paquets reçus comme la confirmation de la réception, numéro de paquets, séquence des paquets et l'affichage des données des paquets qui sont en hexadécimal.

```
karim@karim-laptop:~/Téléchargements/ucla_zigbee_phy/src/examples$
./cc2420_rxtest2.py  -g 40 -r 250000 -c 2430000000  -R A
cordic_freq = 2.43G
cordic_freq = 2.43G
data_rate =  250k
samples_per_symbol =  2
usrp_decim =  16
fs =  500k
Using RX d'board A: Flex 2400 Rx MIMO B
freq=2430000000.0
>>> gr_fir_fff: using SSE
802_15_4_pkt: waiting for packet
received packet
802_15_4_pkt: waiting for packet
received packet
checksum: 32786, received: 32786
ok =  True  pktno =  256  len(payload) =   24  1/1
  payload: ['0x1', '0x0', '0x48', '0x0', '0x0', '0xff', '0xff',
```

```
'0x0', '0x0', '0x40', '0xf0', '0x0', '0x0', '0xff', '0x0', '0x0',
'0x4', '0x0', '0xf', '0x0', '0x1', '0x1', '0x12', '0x80']
 ------------------------
802_15_4_pkt: waiting for packet
received packet
checksum: 626, received: 626
ok =  True  pktno =  256  len(payload) =   18  2/2
  payload: ['0x1', '0x0', '0x4a', '0x0', '0x0', '0x4', '0x0',
'0x0', '0x0', '0x28', '0xf2', '0x0', '0x0', '0x4', '0x81', '0x0',
'0x72', '0x2']
 ------------------------
802_15_4_pkt: waiting for packet
received packet
checksum: 7628, received: 7628
ok =  True  pktno =  256  len(payload) =    7  3/3
  payload: ['0x1', '0x0', '0x45', '0x11', '0xa0', '0xcc', '0x1d']
 ------------------------
802_15_4_pkt: waiting for packet
received packet
```

**Figure 5. 8 :** Signal relié au coordonnateur pour faire le traitement

### 5.4. Les mesures prises pour le traitement du délai des relais sur l'USRP

Des mesures ont été effectuées au niveau de l'USRP, car c'est à ce niveau du matériel que se fait le relais. Pour le type de mesures que nous avons effectuées, le calcul du temps de réception et de l'émission présente la performance du relais. Le calcul des délais pour le relais prend en compte la durée des traitements des blocs utilisés sur GNU Radio et qui sont responsables de la réception et la détection des données plus les opérations traitement de signal comme la modulation, démodulation

etc…. . Les mesures ont été faites sur une séquence de messages. On a remarqué que le temps de traitement n'est pas fixe, mais il faut aussi noter qu'il n'est pas prédéfini par le standard. On a noté le temps de délai qui a été pris pour le traitement des messages reçus et on a estimé un temps moyen de traitement du relais pour chaque type de mesures.

On a comparé nos mesures actuelles avec celles que l'on retrouve dans la littérature publique [28]. On n'a pas trouvé de différences notables mais, ce qui est sûr, c'est qu'on a affirmé qu'il n'est pas possible d'avoir des valeurs fixes et exactes. Une explication pour cela est que dans l'environnement de tests, il y beaucoup de composants matériels qui influent sur les mesures, comme par exemple le câble USB qui relie l'USRP avec l'ordinateur. Les performances de ce dernier compte beaucoup aussi lorsqu'il y a d'autres processus qui s'exécutent en même temps.

### 5.4.1. Délais de traitement au niveau de l'USRP


**Figure 5. 9** : Délai de traitement au niveau de l'USRP

La figure 5.9 illustre le délai de cursus de traitement de signal au niveau de l'USRP pendant la réception et l'émission qui formule le relai. Le signal est de type ZigBee à une fréquence de 2.4 GHz, les messages envoyés ont une taille de données de 22 octets et un débit de transmission de 250 kbit/s.

D'après ces résultats, on remarque que le délai n'est pas fixe et il varie entre 6.1 et 6.5 ms.

La figure 5.10 nous montre le délai de cursus de traitement de signal au niveau de l'USRP pendant la réception et l'émission Le signal est à nouveau de type ZigBee à la fréquence de 2.4 GHz et les messages envoyés ont à nouveau une taille de données de 22 octets mais cette fois, le débit de transmission de 25 kbit/s.

D'après nos résultats, on remarque cette fois que le délai varie entre 16.5 ms et 21 ms.

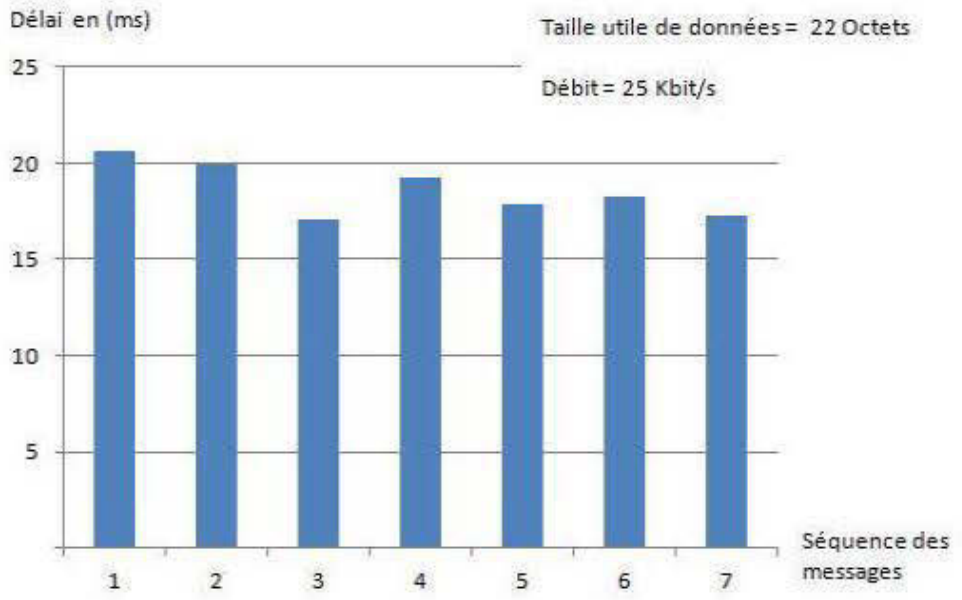



**Figure 5. 10 :** Délai de traitement au niveau de l'USRP avec un débit de 25 Kbit/s et une taille de messages de 22 octets

Finalement, la figure 5.11 illustre le délai de cursus du traitement de signal au niveau de l'USRP pendant la réception et l'émission, ce qui présente le relai.

Le signal est de type ZigBee à la fréquence de 2.4 GHz, les messages envoyés ont une taille de données de 100 octets et un débit de transmission de 250 kbit/s, D'après les résultats de délais obtenus, on remarque que le délai varie entre 10.5 et 19 ms

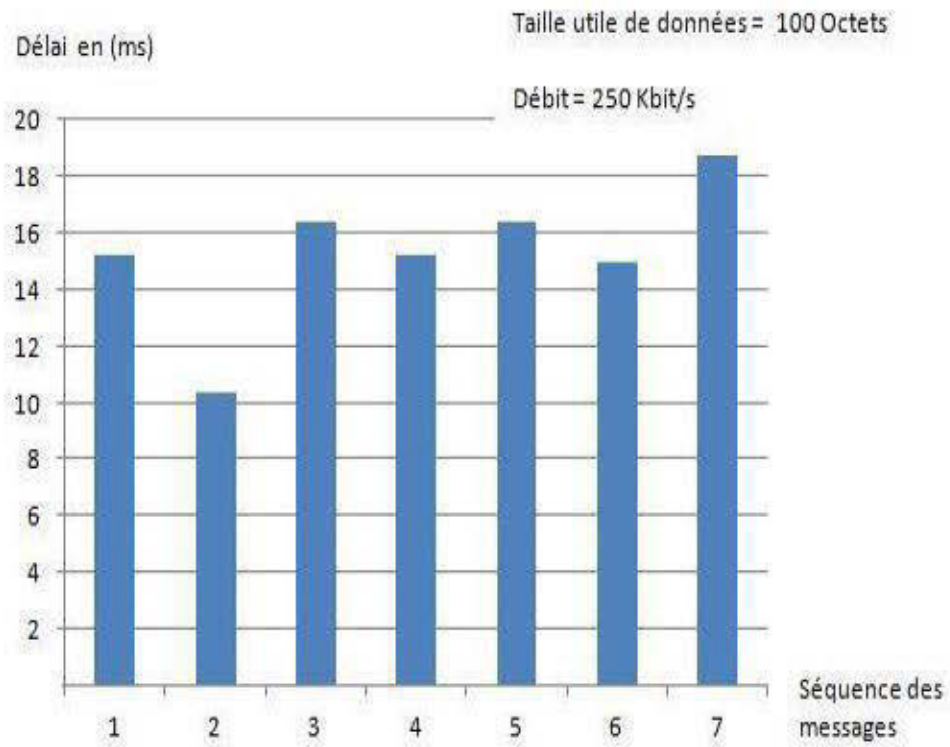



**Figure 5. 11 :** Délai de traitement au niveau de l'USRP avec un débit de 250 Kbit/s et une taille de messages de 100 octets

### 5.4.2. Analyse des délais moyens pour toutes les mesures

La figure 5.12 ci-dessous nous donne une comparaison des moyennes des séquences des messages de tous les délais mesurés au niveau de traitement de signal sur l'USRP.

On a remarqué que lorsque l'on garde la même taille de message de 22 octets et que l'on modifie le débit de 250 kbit/s à 25 kbit/s, le cursus de traitement de signal prend plus de temps et quand on fait le contraire, i.e. que l'on garde le débit supérieur de 250 kbit/s et on augmente la taille de message à 100 octets, on en déduit clairement que le délai de traitement est influencé par la taille et le débit de données.

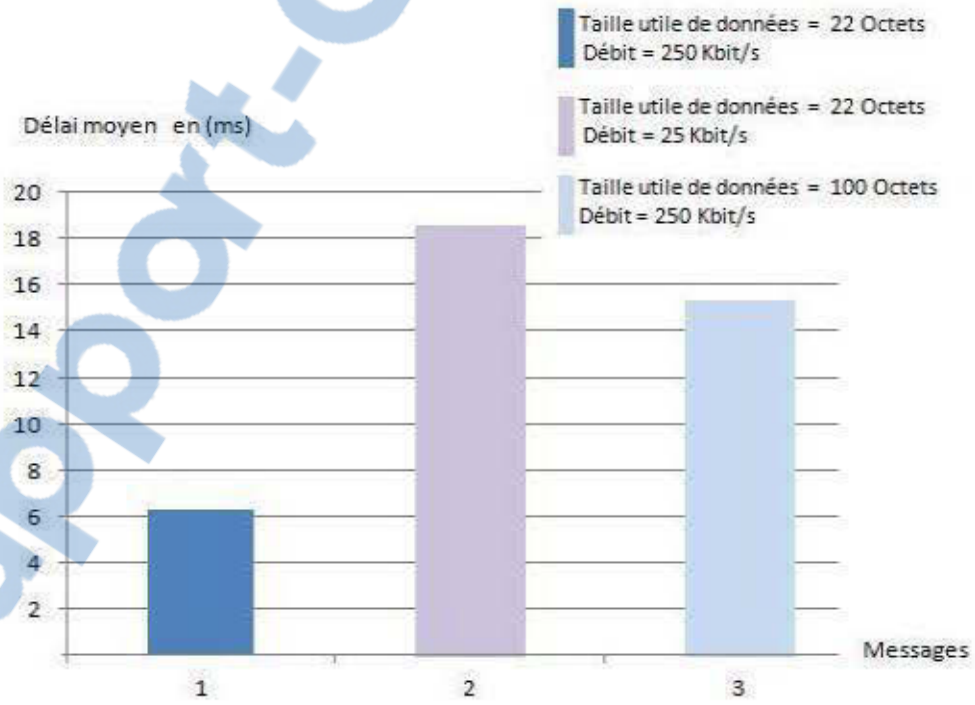



**Figure 5. 12 :** Comparaison des moyennes de tous les délais

# CHAPITRE 6



# CONCLUSION

Dans ce projet, nous avons résolu le problème de la communication entre les réseaux de capteurs et les réseaux de stations de bases grâce à la radio logicielle, la solution GNU Radio et l'USRP. Ces solutions logicielles et matérielles nous ont permis de faire des tests selon nos objectifs et de montrer que les résultats obtenus étaient très acceptables. Alors pour répondre à la problématique, un ensemble de tests a eu lieu dans l'environnement de test qu'on a préparé. Le premier test était le relais du signal Zigbee entre deux stations USRP, la première avait comme rôle de retransmettre le signal et la deuxième son rôle était de recevoir, parmis les démarches prisent dans la pratique on a adapté le signal de la réception à celui de l'émission ce qui a permis la réussite du test. Le deuxième test c'est ce qui a répondu à la problématique, le but c'était de capturer le signal des deux capteurs qui communique entre eux et le relayer à travers des stations USRPs, pour réussir cela, après l'état de l'art, on a travaillé avec la fréquence et le débit de données qui sont définis par le standard 802.15.4. Le travail s'est basé sur la programmation et la modification des blocs responsables du relais et de la réception du signal.

La première partie de la thèse a présenté un état de l'art général sur les réseaux de capteurs, en déterminant les topologies, les architectures et les standards utilisés dans ce type de réseaux ce qui permet d'avoir une communication sans fil à faible coût, à faible débit et efficace. En deuxième partie nous avons présenté l'utilité et les avantages de la radio logicielle sur laquelle notre travail le travail est basé. Subséquemment, on a donné les définitions et les caractéristiques des équipements matériaux utilisés, pour la préparation de l'environnement de test et on a ensuite

proposé un schéma- bloc programmé avec le langage Python pour faire fonctionner le relais. La dernière partie du mémoire a présenté les différents résultats obtenus dans nos expérimentations de transmission du signal ZigBee entre les stations de bases (USRP), ou bien entre les stations de bases et les capteurs ZigBee utilisés. Un ensemble de mesure a été fait au niveau de la station de base, en calculant le temps total de l'émission et la réception ce qui présente le relais.

Pour calculer le temps de traitement de signal pendant le relais au niveau de l'USRP, on a fait des mesures en modifiants le débit et la taille de données. Lorsque l'on garde la même taille de message de 22 octets et que l'on modifie le débit de 250 kbit/s à 25 kbit/s, le cursus de traitement de signal prend plus de temps et quand on fait le contraire, i.e. que l'on garde le débit supérieur de 250 kbit/s et on augmente la taille de message à 100 octets, on en déduit clairement que le délai de traitement est influencé par la taille et le débit de données. Les résultats des mesures des délais, donnent une idée sur le temps traitement du cursus émission/ réception car il y'a des mesures fixes définies par le standard, elle change selon les performances des équipements matériels (USRP, Capteur, Ordinateur) à un autre, et d'une version de solution logicielle (GNU Radio, Ubunto) à une autre.

D'après les objectifs qui ont été atteint dans ce projet, il faut préciser que ce qui a été réalisé a pris beaucoup de temps et d'essais, car le travail a été fait sur une plateforme existante (problèmes de compatibilité des versions logicielles, méthode et fonctionnement des équipements matériels etc….).Également, les modifications aux codes ont pris beaucoup de temps car quand quelque chose ne marche pas, on est obligé de vérifier et chercher d'autres solutions pour réussir les expérimentations. Ce n'est pas comme dans le cas des simulations sur des logiciels, là où on a une erreur le logiciel nous donne son type et dans quelle ligne de code on doit corriger.

Selon ce qui a été réalisé jusqu'à présent et pour améliorer davantage le fonctionnement du relais, il faut penser à approfondir les recherches. Ce qu'on peut

proposer comme perspectives, pourquoi ne pas penser à refaire ce travail en utilisant le matériel USRP 2. Il est plus développé et il permettait de faire les expérimentations en exécutant toute la pile ZigBee, ce qui veut dire qu'on ne va pas s'arrêter au niveau de la couche physique et MAC, mais que l'on peut arriver jusqu'à la couche réseau et applications.

# BIBLIOGRAPHIE

[1] Karch, H., A. Willig, A., Wireless Sensor Network, University of Potsdam Germany, Wiley, April, 2005, (First Edition) Chap 1.p. 15-30.

[2] Camus, M., Architecture de réception RF très faible coût et très faible puissance. Application aux réseaux de capteurs et au standard ZigBee, Thèse, Université Paul Sabbatier, (Toulouse II ) Fevrier 2008.

[3] Bossche V. D. , " Une méthode totalement déterministe pour un réseau personnel sans fil ", Thèse, Université de Toulouse II (Le Mirail), (EDSys) 2007

[4] IEEE computer society The, Institute of Electrical and Electronics Engineers, "Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), The Institute of Electrical and Electronics Engineers, October 2003, Chapter 6.

[5] ZigBee Alliance, ZigBee Specification, 2005, Http://www.nd.edu/~mhaenggi/ee67011/zigbee.pdf

[6] ZigBee Alliance. Site web, Http://www.zigbee.org

[7] IEEE Computer Society, 802.11 IEEE Standard for Information technology

[8] IEEE Computer Society, 802.15.4 IEEE Standard for Information technology, 2003

[9] Farahani, S. , ZigBee Wireless Networks and Transceivers, Elsevier's Science & Technology Rights Department in Oxford, 2008, Chap 1. P.1-22.

[10] Grand, M., Conception d'un crypto-système reconfigurable pour la radio logicielle sécurisée, Thése, École doctorale des sciences physiques de l'ingénieur (Université de bordeaux 1), Spécialité Électronique, Décembre 2011.

[11] Software Defined Radio, http://www.converter-radio.com/SDR.pdf

[12] Jouida, N. , Modulateur Σ Δ Complexe Passe-Bande a Temps-Continu pour la Reception Multistandard, Thèse, École Supérieure des Communications, Tunis, février 2010.

[13] Gnu radio sur www.gnuradio.org

[14] Verduin, A., GNU Radio, Wireless protocol analysis approach, University Van Amsterdam, (Final Version), October 2008, Chapter 1, p 7-15.

[15] Thomas Schmid, UCLA ZigBee PHY project website, Https://www.cgran.org/wiki/UCLAZigBee

[16] Usrp_documentation: www.ettus.com/

[17] Balister, P., Reed, J., USRP Hardware and Software Description, Bradley Dept. of Electrical & Computer Engineering Virginia Polytechnic Institute & StateUniversity June 30, 2006.

[18] RFX/FLEX 400 daughterboard datasheet, Http://www.ettus.com/downloads/transceiver_dbrds_v3b.pdf

[19] RFX2400 daughterboard datasheet, Http://www.ettus.com/downloads/er_ds_transceiver_dbrds_v5b.pdf

[20] RFX 2400, block diagram, Http://gnuradio.org/redmine/projects/gnuradio/wiki/UsrpRfxDiagrams

[21] Le, H.C., Optimisation d'accès au médium et stockage de données distribuées dans les Réseaux de capteurs, Thèse, U.F.R des sciences et techniques, Université de Franche-Comté, 2007.

[22] Wang, H., Tao, T., Song, J., "Analysis of Common Structure and Software Download for Software Defined Radio" , Beijing Univ. of Posts & Telecom. PCN & CAD Center,2000 pp. 1113-1116.

[23] Shirato, Y., Shiba , H., Uehara, K., Shono , T., Katsuhiko, K., Umehira, M. "IEEE 802.11 Wireless LAN Implemented on Software Defined Radio With Hybrid Programmable Architecture", IEEE Transactions On Wireless Communications, vol.4, no. 5, SEPTEMBER 2005.

[24] Ronga, L.S., Vettori, L., Del ,R. E., Presti, L.L., Falletti,E.,Pini, M., "Software Defined Radio Terminal for Assisted, Localization in Emergency Situations", Dipartimento di Elettronica e Telecomunicazioni, Università di Firenze, Dipartimento di Elettronica, Politecnico di Torino, Istituto Superiore Mario Boella (ISMB), Torino, Italy 2009.

[25] IEEE 802.15.4-2006 standard, http://standards.ieee.org/getieee802/802.15.html

[26] Sanka, S., Konchady, G., "Communication between wireless sensor devices and GNU Radio", Departement of electrical and computer engineering, Cleveland State University, May,2009

[27] Li, C., Raghunathan, A., Jha, N.K., "An Architecture for Secure Software Defined Radio" Princeton University, Purdue University, 2009

[28] Schmid, T., Sekkat, O., Srivastara, M.B., University of California "Study of Network Performance Impact of Increased Latency in Software Defined Radios", Network and Embedded Systems Laboratory, University of California, Los Angeles, September 2007

[29] Mark W. Ch., "A Software Defined By Radio", Harris Corporation, RF Communications Division, Rochester, New York 2007

[30] Revés, X. Maojevic, V., Gelonch, A., Ferrus, R., "The Cost of an Abstraction layer on FPGA Devices for software radio applications", Universitat Politècnica de Catalunya, Barcelona, Spain, 2004

# ANNEXES

## Annexe A

### Carte RFX2400 2.3-2.9 GHz Rx/Tx

Le RFX2400 est une carte de haute performance, capable d'émettre et de recevoir duplex intégral de signaux conçu spécifiquement pour fonctionner dans la bande 2,4 GHz. Le RFX2400 fournit une puissance de sortie typique de 50 mW, et la figure de bruit de 8 dB. La carte fille utilise un filtre SAW en série avec le port TX / RX pour fournir une sélectivité supérieure et de la performance parasite entre 2,4 et 2,483 GHz. Le port RX2 donne accès à la gamme de fréquences de la carte fille, 2.3 à 2.9 GHz. Le RFX2400 utilise une entrée indépendante de LO pour la chaîne de transmission et la chaîne de réception.

Dans le schéma, G=gain et N=noise (bruit) figure en (dB) à 2.4 GHz

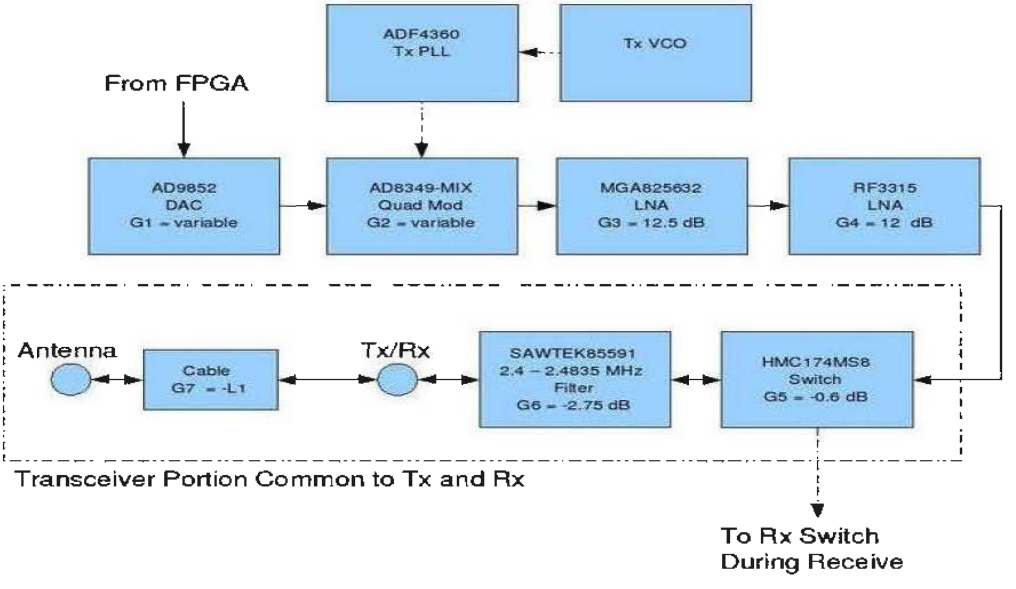



**Figure 5. 13 :** Schéma principal du trajet d'un signal émis dans RFX 2400

Schéma principal du trajet d'un signal reçu dans RFX 2400 :

Dans la figure, G=gain et N=noise (bruit) figure en (dB) à 2.4 GHz

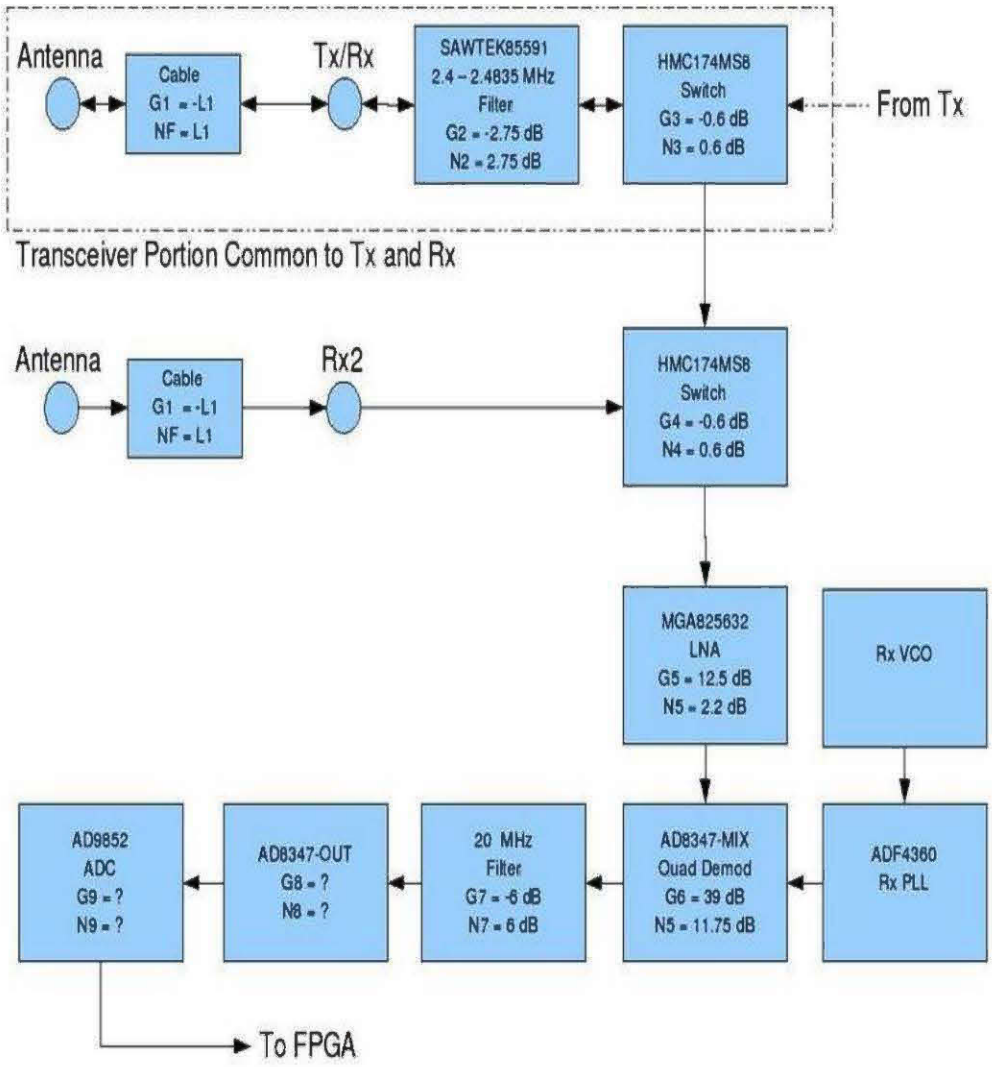



**Figure 5. 14 :** Schéma principal du trajet d'un signal reçu dans RFX 2400

**Annexe B**

**Antenne**

Antenne VERT 2450

Comprend un VERT2450 Dual Band 2,4 à 2,48 GHz et 4.9 à 5.9 GHz, antenne omnidirectionnelle verticale, à Gain 3 dBi.

**Figure 5. 15 :** Antenne VERT 2450

**Annexe C**

Cette partie de l'annexe nous présente les composants et les circuits qui font partie du capteur Silicon. Les données ci-dessous sont prises de la fiche technique du matériel.

**Capteur Silicon C8051F120/1/2/3/4/5/6/7**

Analog Peripherals

- SAR ADC
• 12-Bit (C8051F120/1/4/5)
• 10-Bit (C8051F122/3/6/7)
• ± 1 LSB INL
• Programmable Throughput up to 100 ksps
• Up to 8 External Inputs; Programmable as Single-Ended or Differential
• Programmable Amplifier Gain: 16, 8, 4, 2, 1, 0.5
• Data-Dependent Windowed Interrupt Generator
• Built-in Temperature Sensor
- 8-bit ADC
• Programmable Throughput up to 500 ksps
• 8 External Inputs (Single-Ended or Differential)
• Programmable Amplifier Gain: 4, 2, 1, 0.5
- Two 12-bit DACs
• Can Synchronize Outputs to Timers for Jitter-Free Waveform Generation
- Two Analog Comparators
- Voltage Reference
- VDD Monitor/Brown-Out Detector

ON-CHIP JTAG Debug & Boundary Scan

- On-Chip Debug Circuitry Facilitates Full- Speed, NonIntrusive In-Circuit/In-System
Debugging
- Provides Breakpoints, Single-Stepping, Watchpoints,
Stack Monitor; Inspect/Modify Memory and Registers
- Superior Performance to Emulation Systems Using ICEChips, Target Pods, and Sockets
- IEEE1149.1 Compliant Boundary Scan
- Complete Development Kit

High Speed 8051 µC Core

- Pipelined Instruction Architecture; Executes 70% of
Instruction Set in 1 or 2 System Clocks
- Up to 100 MIPS (C8051F120/1/2/3) or 50 MIPS
(C8051F124/5/6/7) Throughput using Integrated PLL
- 2-cycle 16 x 16 MAC Engine (C8051F120/1/2/3)
- Flexible Interrupt Sources

Memory

- 8448 Bytes Internal Data RAM (8k + 256)
- 128k Bytes Banked FLASH; In-System programmable in
1024-byte Sectors
- External 64k Byte Data Memory Interface (programmable multiplexed or non-multiplexed modes)

Digital Peripherals

- 4 Byte-Wide Port I/O (C8051F121/3/5/7); 5V tolerant
- Programmable 16-bit Counter/Timer Array with
- 5 General Purpose 16-bit Counter/Timers

Clock Sources

- Internal Precision Oscillator: 24.5 MHz
- Flexible PLL technology
- External Oscillator: Crystal, RC, C, or Clock

Power Supplies

- Supply Range: 2.7-3.6V (50 MIPS) 3.0-3.6V (100 MIPS)
- Power Saving Sleep and Shutdown Modes

  100-PIN TQFP OR 64-PIN TQFP PACKAGING
- Temperature Range: -40°C to +85°C

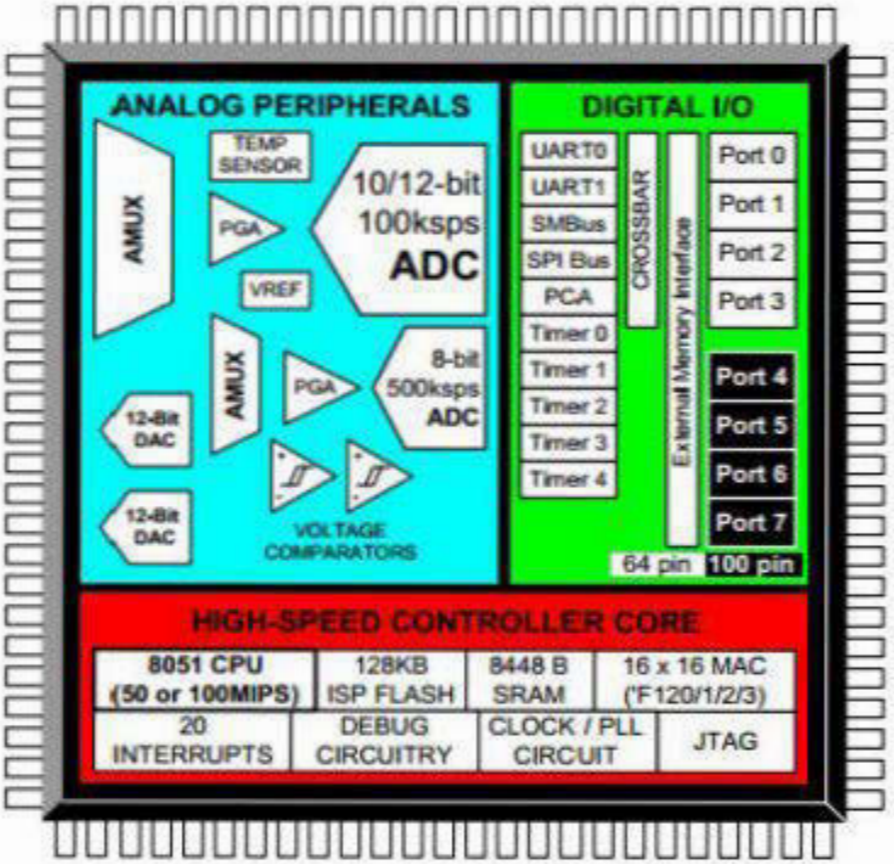



**Figure 5.16 :** Schéma de bloc de C8051F120/1/2/3/4/5/6/7