PONTAGNIER Etienne

VUONG Dan

PG208 – Projet Thématique

-

Communication sans fils

# I/. Introduction

## 1. Objectifs

L’objectif de ce projet était de mettre en place une communication sans fils entre deux radios. Les deux radios utilisées pour ce projet sont des radios de type USRP. Ces radios utilisent une carte électronique USRP qui permet une connexion entre celles-ci et un ordinateur pour pouvoir interagir avec un logiciel de radio. Le but étant d’envoyé des données d’un ordinateur à l’autre que ce soit un fichier texte, une image ou une vidéo.

Les radios USRP n’ont pas besoin d’être programmées, elles permettent d’être utilisée avec le logiciel GNU radio qui permet d’envoyer et de recevoir des données. Cependant il faut au niveau de l’émission il faut au préalable mettre en forme les données à envoyer et au niveau de la réception il faut traiter les données reçues afin de les rendre compréhensible.

Ce rapport décrit le fonctionnement de la chaine de communication, les étapes qui la compose ainsi que les différentes fonctions et algorithmes utilisées. Le code définissant le fonctionnement des radios USRP ne sera pas explicité en détail.

## 2. Cahier des charges

Le procédé de codage utilisés pour les signaux envoyés sera le procédé OFDM. La chaine de communication ainsi réalisé devra permettre l’envoi ainsi que la réception de données tel qu’un fichier texte, une image ou une vidéo. Le procédé OFDM devra permettre :

-de limiter le décalage temporel ;

-de limiter le décalage en fréquence ;

-de limiter le bruit introduit par le canal ;

-de limiter l’action du coefficient de canal.

Pour le début du projet la chaine de communication sera modélisée sous Matlab afin de faciliter le développement de la chaine. Par la suite le traitement des données sera codé en C++ afin de pouvoir être utilisé avec les radios USRP.

# II/. Description de l’architecture et fonctionnement.

## 1. Fonctionnement de la chaine de communication

Pour que les radios puissent envoyer efficacement les données et pour éviter que des erreurs, du bruit, soient introduit dans le canal de transmission ces données doivent d’abord être traités. Les données seront transmises par modulation de phase :

FIGURE 1 CHAINE EMISSION

- Bloc « Association Bits/Symbole : La première étape consiste à mettre en forme les données binaires pour la modulation de phase pour ce projet c’est la méthode BPSK qui sera utilisée. Les valeurs binaires du message à envoyer prendront des valeurs entre 1 et -1 et ces données seront structurées sous forme d’un vecteur ss. Ce vecteur ss sera donc composé uniquement de 1 et -1.

-Bloc « Série/parallèle » : Une fois le vecteur ss formé, il faut le diviser en plusieurs sous vecteurs d’une taille défini au préalable. En effet le procédé OFDM est une modulation multi porteuse, c’est-à-dire que les données sont transmises en les modulant sur plusieurs porteuses en même temps. Donc les données du vecteur ss sont groupées par paquet de n éléments.

- Bloc « IFFT » : Une fois le signal ss multiplexé en plusieurs sous vecteurs, il faut ensuite les moduler. On utilise donc la IFFT sur les différents vecteurs parallèles pour les moduler.

-Bloc « Ajout préfixes cycliques » : L’envoi des données dans le canal au moment de la transmission peut provoquer un phénomène d’écho, les signaux envoyés peuvent être retardés par l’environnement et il peut donc y avoir une interférence entre deux trames émises successivement. Pour contrer cette interférence on insère un préfixe cyclique avant chaque trame qui est une copie de la fin de cette trame.

- Bloc « Parallèle/Série » : Maintenant que la donnée à transmettre est formée il faut la remettre sous forme d’un seul vecteur pour pouvoir la transmettre. Le vecteur est composé à ce moment-là de complexe double.

C’est ensuite la partie radio logiciel qui prend le relai. Elle se charge de d’envoyer la donnée mise en forme précédemment. D’autre part une deuxième radio identique réglée en mode « réception » se charge de recevoir le signal. Le signal composé de complexe double va donc ensuite être traitée afin de reconstituer le signal d’origine et ceci de la même manière que pour chaine de transmission :

FIGURE 2 CHAINE RECEPTION

-Bloc « Série/Parallèle » : Afin de reconstituer le signal original il faut donc rediviser le vecteur en sous vecteurs pour pouvoir les traiter en parallèle.

-Bloc « Corrélation et détection des préfixes » : Maintenant qu’on a récupéré et mis en parallèle le signal, on va chercher à détecter les préfixes cycliques insérer précédemment afin de les supprimer. Pour cela on utilisera la technique présentée par Schmidl et Cox qui permet de déterminer l’emplacement des préfixes cycliques dans le signal. Cette technique consiste à calculer le coefficient de corrélation pour déterminer la ressemblance de différents buffers du signal, si deux buffers sont ressemblants se sont surement des préfixes cycliques.

-Bloc « calcul de l’offset et moyennage » : En plus de l’interférence entre trames il peut aussi y avoir un offset du signal lors de la réception. Lors de la réception de celui-ci il faut donc déterminer la durée de l’offset afin de le supprimer. Pour cela on réalisera une moyenne dans le temps afin d’obtenir la durée de l’offset et une fois cette durée obtenue on pourra corriger la position des préfixes cycliques obtenues précédemment.

-Bloc « suppression préfixes cycliques » : Une fois la position des préfixes cycliques corrigés il suffit maintenant de tronquer le signal aux bons endroits afin de les supprimer.

-Bloc « FFT » : Les données dans la chaine de transmission ont été modulées par une IFFT, pour reconstruire le signal d’origine il faut dont réaliser une FFT (Opération inverse de l’IFFT). Le signal obtenu est donc un vecteur composé de double.

-Bloc « décision » : Cette étape permet de réduire le bruit qui a pu être introduit dans le canal. Il s’agit d’associer chaque valeur du vecteur précèdent à sa valeur symbole (1 ou -1).

-Bloc « symbole/Bits » : Pour reconstituer le signal d’origine il suffit maintenant d’associer chaque symbole à sa valeur binaire : si la valeur symbole est 1 alors sa valeur binaire est 1 et si la valeur symbole est -1 alors sa valeur binaire est 0.

## 2. Format des données utilisées

Pour le format des données à envoyés on choisira :

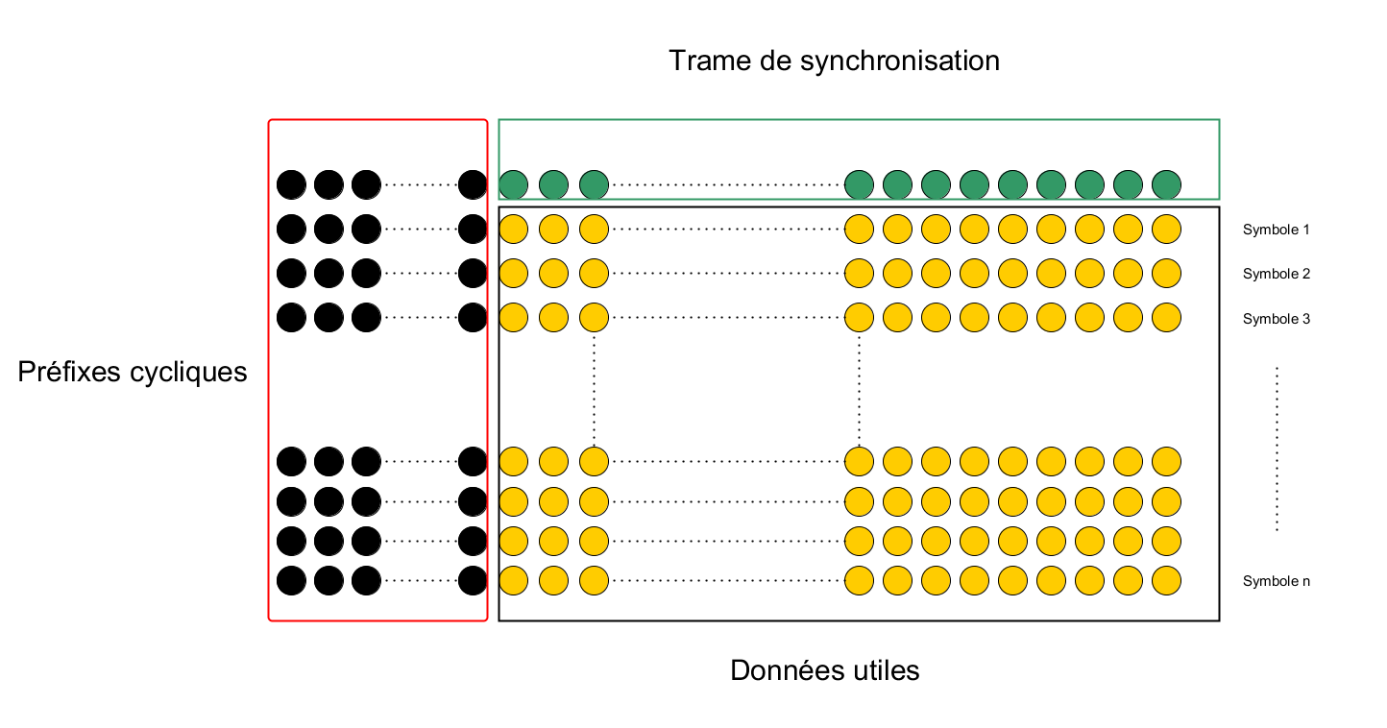
-n = 128 : Le nombre de sous porteuses pour nos symboles OFDM ;

-l = 16 : La taille des préfixes cycliques ;

-N= 20 : Le nombre de trames pour le signal envoyé ;

-ISR = 10 : 1 trame de synchronisation pour 10 trames de données ;

-Nsync = N/ISR : Le nombre de trames de synchronisation ;

 -Ns = n\*N : Nombre totale de données à envoyer ;

*Figure 3. Format des données envoyées*

Une trame OFDM contient donc :

. Pour une trame de synchronisation :

-16 bits de préfixes cycliques ;

-128 bits de données connues.

. Pour une trame de données :

-16 bits de préfixes cycliques ;

-128 bits de données utiles à envoyer

# III/. Description des blocs de traitement des données

## 1. Chaine de transmission

Avant de travailler sur les radios il a d’abord fallu tester la chaine de communication sur Matlab. Pour cela on a donc généré un vecteur binaire aléatoire de Ns= 2560 éléments soit 20 trames de 128 éléments.

### Association Bits/Symbole

Le vecteur x[] généré grâce à la fonction rand(2, 1, Ns) de Matlab nous permet d’avoir une vecteur ss[] de Ns entiers compris entre 0 et 1 de manière aléatoire. Pour associer les bits à leurs symboles équivalent il suffit sous Matlab de multiplier le vecteur par 2 et soustraire 1 à chaque élément, de cette façon les 0 binaire deviennent des -1 et les 1 restent des 1.

### b. Série/parallèle et IFFT

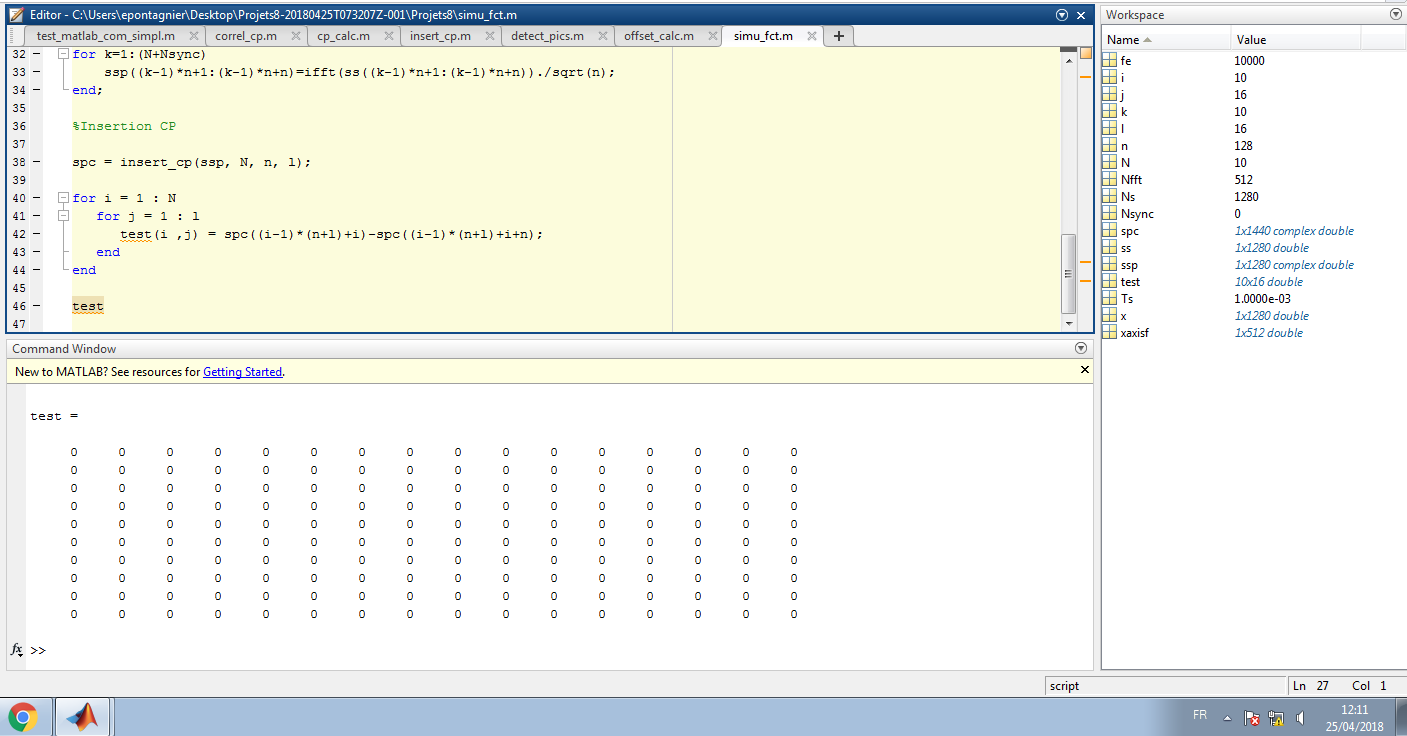
Là où la mise en parallèle des vecteurs aurait dû se faire avant l’IFFT les outils de Matlab permettent de faire ces deux étapes en simultanées. On réalise ensuite une IFFT sur des sous vecteurs de ss[] de 128 éléments qui forment donc chacun une trame du signal. Le logiciel Matlab intègre déjà une fonction ifft() qui retourne un l’ifft d’un vecteur rentré en argument. Ces trames sont ensuite concaténées dans un seul vecteur ssp[] qui contient donc à la suite les 20 trames de 128 éléments.

### c. Génération et insertion des trames de synchronisation

Pour limiter les problèmes de décalage de phase et aussi corriger l’effet du canal sur le signal il faut insérer des trames pilotes dans le signal d’origine. Ces trames pilotes sont connues et permettent donc de déterminer le coefficient de canal qui perturbe le signal d’origine, en effet il suffit de comparer la trame pilote envoyé et la trame pilote reçu qui a été altéré par le canal lors de la transmission. Pour le prototype sur matlab la trame pilote sera générée aléatoirement avec la fonction randi() de matlab. On choisira d’insérer une trame pilote toutes les 10 trames OFDM.

### d. Insertion de préfixes cyclique

Pour éviter les phénomènes d’interférences entre symbole il faut insérer des préfixes cycliques avant chaque trame du signal, et comme préciser précédemment ces préfixes cycliques sont la copie des l=16 derniers éléments de chaque trame. Pour cela, sur Matlab on parcourt le vecteur ssp[] puis on stocke les l=16 derniers éléments de chaque trames pour les inclure entre celles-ci. Pour vérifier si l’algorithme à bien recopié les bons éléments on soustrait les 16 derniers éléments de chaque trame aux 16 premiers.



On remarque bien que le résultat de cette soustraction est 0 ce qui indique bien les préfixes cycliques sont bien les copies des 16 derniers éléments de chaque trame.

Pour faciliter le prototypage de la chaine de communication on stockera une copie des ces préfixes cycliques dans un vecteur ainsi que leurs positions dans le signal.

