الجـــــــــــــــــــــمـهـوريـــــة الجـــــــــــــزائـريـة الديمـــــــقـراطيـة الشـــــــــعبيـة

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التـــــــــــــــــــــــــــــعـليـم العـــــــالي و البـــــــــــحث العـــــلـمـــي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جـــــــــــــــامعة أبي بكـــر بـلـقـايد– تـلمســـــــــــــــان –

Université Aboubakr Belkaïd– Tlemcen –
Faculté de TECHNOLOGIE

MEMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER**

En : Télécommunications

Spécialité : Systèmes de Télécommunications

Par : GÂAD Mohammed et RAHMI Bachir

Sujet

Conception et Réalisation d'un émetteur OFDM à base d'une carte RASPBERRY PI

Soutenu publiquement, le **25 /06/ 2018** , devant le jury composé de :

Mr F.T.BENDIMERAD	Professeur	Univ. Tlemcen	Président
Mr F.DERRAZ	Maître de Conférences	Univ. Tlemcen	Encadreurs
Mr M.BOUSAHLA	Maître de Conférences	Univ. Tlemcen	
Mr S.KAMECHE	Maître de Conférences	Univ. Tlemcen	Examinateur

Année universitaire: 2017/2018

À ***Nos parents***

& Nos Familles

Conception et Réalisation d'un émetteur OFDM à base d'une carte RASPBERRY PI

RÉSUMÉ

On observe de nos jours un intérêt considérable pour de nouvelles technologies sans fil capables de réaliser des hauts débits. La technique du multiplexage par division de fréquences orthogonales (OFDM) est l'une de ces techniques qui ont pris un grand succès.

Dans un système OFDM, la bande de fréquence est divisée en des multiples sous-porteuses orthogonales. L'information à transmettre est répartie en petits blocs de données qui sont chacun affectés à ces sous-porteuses. L'avantage de la technique OFDM et de pouvoir récupérer l'information transmise même si plusieurs échos ont affecté la transmission radioélectrique.

L'objet de ce projet est l'étude, la conception et réalisation d'un émetteur OFDM à base d'une carte Raspberry Pi. Nous avons commencé par l'installation du support package du Raspberry Pi pour Simulink, puis nous avons fait la conception et la simulation de l'émetteur OFDM sous Simulink. La simulation nous a permis de vérifier le bon fonctionnement de notre émetteur OFDM et d'étudier ses performances en fonction de la variation de plusieurs paramètres.

La dernière étape de notre réalisation est l'implémentation de l'émetteur OFDM sur la carte Raspberry Pi.

Mots clé:

La technique du multiplexage par division de fréquences orthogonales (OFDM), Raspberry Pi, conception, réalisation, canal à trajets multiples.

ABSTRACT

There is considerable interest nowadays in new wireless technologies capable of achieving high speeds. Orthogonal frequency division multiplexing (OFDM) is one of the most successful of these techniques.

In an OFDM system, the frequency band is divided into multiple orthogonal subcarriers. The information to be transmitted is divided into small data blocks, each of which is assigned to these sub-carriers. The advantage of OFDM technology is that the information transmitted can be recovered even if several echoes have affected the radio transmission.

The purpose of this project is to study, design and manufacture an OFDM transmitter based on a RaspberryPi card. We started by installing the support package for the Raspberry Pi for Simulink, then we designed and simulated the OFDM transmitter under Simulink. The simulation allowed us to verify the correct operation of our OFDM transmitter and to study its performance according to the variation of several parameters.

The last step of our realization is the implementation of the OFDM transmitter on the Raspberry Pi card.

Keywords:

Orthogonal frequency division multiplexing (OFDM) technique, Raspberry Pi, design, realization, multipath channel.

REMERCIEMENTS

Nous remercions tout d'abord DIEU pour tous les bénédictions qui nous ont donné durant toutes ces années la santé et le courage poursuivre notre étude.

Nous exprimons du profonde du cœur les plus sincères remerciements à nos encadreurs, monsieur ***M. BOUSAHLA*** *MCB et monsieur* ***F. DERRAZ*** *MCB à l'université de Tlemcen, d'avoir dirigé notre travail avec la confiance qu'ils nous ont accordé, et le temps qu'ils nous ont donné, et en outre, les conseils et les idées qui nous ont facilité de nombreuses difficulté tout au long de notre travail. Nous avons appris d'eux comment développer nos compétences scientifiques et pratiques, en plus de la persévérance et du sérieux qui nous avons tenus à l'écart des difficultés. A travers sa longue expérience combinant l'ingénierie, la recherche scientifique, l'enseignement et finalement la direction.*

Nous remercions aussi tous les membres du jury d'avoir accepté notre participation à cette soutenance, vous nous avez donné l'honneur d'évaluer et d'examiner notre travail pendant tout le temps que vous avez passé à lire notre mémoire.

Nous remercions également Mr F.T. BENDIMERED, professeur à l'université de Tlemcen pour avoir accepté de présider le jury.

Nous adressons nos remerciements à Mr S. KAMECHE, Maitre de conférence à l'université de Tlemcen pour l'intérêt qu'il a bien voulu porter à ce travail en acceptant de faire partie du jury.

Nos remerciements vont également à tous les enseignants de l'université et également tout le personnel du Laboratoire de Télécommunications pour leurs gentillesses ainsi qu'à la promotion MASTER Systèmes de télécommunication, Réseaux et Télécommunication.

Enfin, nous exprimons nos remerciements à tous ceux qui ont contribué de près ou de loin à l'élaboration de ce travail.

DÉDICACES

En premier en remercie dieux pour tout.

*À **Nos parents***

Que nous aimons tant et qui nous ont soutenu ce long chemin d'étude

Nous les remercions de leur confiance et de leur amour.

Nous espérons que le fruit de nos efforts leurs apporteront fierté.

*À **Nos sœurs et nos frères***

Pour leurs encouragements continus et pour toute nos familles.

À la promo systèmes de Télécommunication ST 2017/2018 de l'université de Tlemcen

Nous vous souhaitons une vie pleine de réussite, santé et de bonheur.

À nos enseignants dont la liste s'allonge d'une année à un autre

À toutes les personnes qui ont participé à l'élaboration de ce travail à tous ceux que nous avons oublié de citer.

GAAD Mohammed et RAHMI Bachir

TABLE DES MATIÈRES

LISTE DES FIGURES

ABRÉVIATION

ACI	Adjacent Channel Interference.
AGC	Automatic Gain Control.
ALSA	Advanced Linux Sound Architecture.
BBtoIF	Base Band to Intermedia Frequency.
BNC-RCA	Bayonet Neill Concelman- Root Cause Analysis.
CPU	Central processing Unit.
CSI	Camera Serial Interface.
DC	Direct Courent.
DFT	Discret Fourier Transform.
DSI	Display Serial Interface.
FDM	Frequency Division Multiplexing.
FFT	Fast Fourier Transform.
FI	Frequency Intermediar.
GPIO	General Purpose Input/Output.
GPU	Graphics Processing Unit.
HDMI	High Definition Multimedia Interface.
IDFT	Inverse Discret Fourier Transform .
IEP	Interférences Entre Porteuses.
IES	Interférences Entre Symboles.
IFFT	Inverse Fast Fourier Transform.
IIS	Interférences Inter Symboles.
IOT	Internet Of Things.
IP	Internet Protocol.
MAQ	Modulation d'Amplitude en Quadrature.
MRF	Multiplexage par Répartition en Fréquence.
OFDM	Orthogonal Frequency Division Multiplixing.
OL	Oscillateur Local.
PAM	Pulse Amplitude Modulation.
PC	Préfixe Cyclique.
PLL	Phase Loop Locking.
PSK	Phase Shift Keying.
QAM	Quadratur Amplitude Modulation.
QPSK	Quadratur Phase Shift Keying.

RAM	Random Acces Memory.
RCA	Root Cause Analysis.
RF	Ratio Frequency.
SC	Suffixe Cyclique.
SD	Secure Digital.
SNR	Signal to Noise Ratio.
SOC	System On Chip.
USB	Universal Serial Bus.
USB-LAN	Universal Serial Bus-Local Area Network.
VC	Virtuel Carriers.
WIFI	Wireless Fidelity.

INTRODUCTION GÉNÉRALE

Les systèmes des télécommunications numériques ont récemment subi de grands développements et ceci grâce aux progrès réalisés en microélectronique qui ont permis l'implémentation matérielle d'algorithmes complexes de traitement numérique des signaux. On observe une croissance constante des débits de transmission ainsi qu'un besoin de se libérer des câbles afin de permettre aux usagers de se déplacer tout en maintenant une qualité acceptable des communications. La qualité des communications sans fil dépend principalement du canal. En général, le canal sans fil est un canal à trajets multiples qui varie dans le temps.

Dans ce type de système de communication il faut résoudre efficacement l'effet des canaux à trajets multiples. La technique OFDM (Orthogonal Frequency Division Multiplexing) a été développé et a apporté une solution à ce problème.

Les propriétés les plus importants de cette technique sont la lutte contre le délai d'étalement généré par un canal à trajets multiples et l'obtention d'un débit élevé.

OFDM (Orthogonal Frequency Division Multiplexing) est un système de communication numérique multi-porteuse. Il répartit l'information à transmettre sur un grand nombre de sous-porteuses à faible débit de données pour construire un système de communication composite à haut débit de données. L'orthogonalité donne aux porteuses une raison valable d'être rapprochées, voire superposées, sans interférence entre porteuses. Le faible débit de données de chaque porteuse implique de longues périodes de symboles, ce qui diminue considérablement les interférences inter-symboles.

Le travail présenté dans ce projet a pour but la conception et la réalisation d'un émetteur OFDM à base d'une carte Raspberry Pi.

Ce mémoire comporte trois chapitres suivis d'une conclusion générale:

Le premier chapitre est consacré à la présentation de la technique OFDM. Il est composé de deux parties, la première partie présente le canal à trajets multiples, l'effet des trajets multiples et la sélectivité du canal. Dans la deuxième partie de ce chapitre nous définissons la technique OFDM et nous donnons le principe de cette technique.

Le deuxième chapitre contient une description générale et simplifiée de la carte Raspberry Pi. Nous présentons aussi dans ce chapitre, toutes les étapes du processus d'installation du package de la carte Raspberry Pi et le test de connectivité de cette carte avec l'ordinateur.

Le troisième chapitre présente l'émetteur OFDM que nous avons conçu. Nous donnons le schéma Simulink de cet émetteur et la définition de chaque sous bloc. Ensuite nous présentons les résultats de simulation sous Simulink. Ainsi, nous présentons l'implémentation de l'émetteur OFDM sur la carte Raspberry Pi.

Enfin, nous terminons notre mémoire avec une conclusion générale.

CHAPITRE I

Modulation Multi porteuses (OFDM)

1. Introduction:

Un des problèmes majeurs en télécommunications est d'adapter l'information à transmettre au canal de propagation. Pour des canaux sélectifs en fréquence, une technique est l'utilisation de modulations multi porteuses.

Dans ce chapitre nous présentons la technique OFDM. La première partie de ce chapitre est consacrée à la présentation du canal à trajets multiples, l'effet des trajets multiples et l'effet doppler. Après nous introduisons la sélectivité de canal.

Dans la deuxième partie nous introduisons le principe de base de l'OFDM. Nous exposons l'orthogonalité fréquentiel. Ensuite nous parlons sur la conversion série/parallèle. Puis nous présentons la modulation de chaque sous-porteuse. Et après l'utilisation d'IFFT pour la génération de signal OFDM. Nous expliquons l'intervalle de garde, la bande de garde et ses influences sur la modulation. Ainsi, nous exposons la modulation RF du signal OFDM. En fin nous introduisons aussi le fenêtrage de signal OFDM.

2. Canal à trajets multiples :

Les communications numériques sans fil en milieu urbain ainsi que dans les édifices sont sujettes aux effets des trajets multiples. En effet, le signal mesuré à l'entrée du récepteur sans fil représente la somme des signaux qui ont traversé différents trajets durant leur propagation et qui arrivent au récepteur à un instant donné. Ce signal reçu est distorsionné par rapport au signal transmis et subit des variations d'amplitude importantes et aléatoires.

Dans un canal multi-trajets, le signal émis sous forme d'onde conduit à plusieurs phénomènes physiques (figure 1.1):

- La réflexion du signal sur un obstacle.
- La réfraction du signal lorsque celui-ci traverse un milieu d'indice différent de celui d'où il provient.
- La diffraction due à un obstacle.

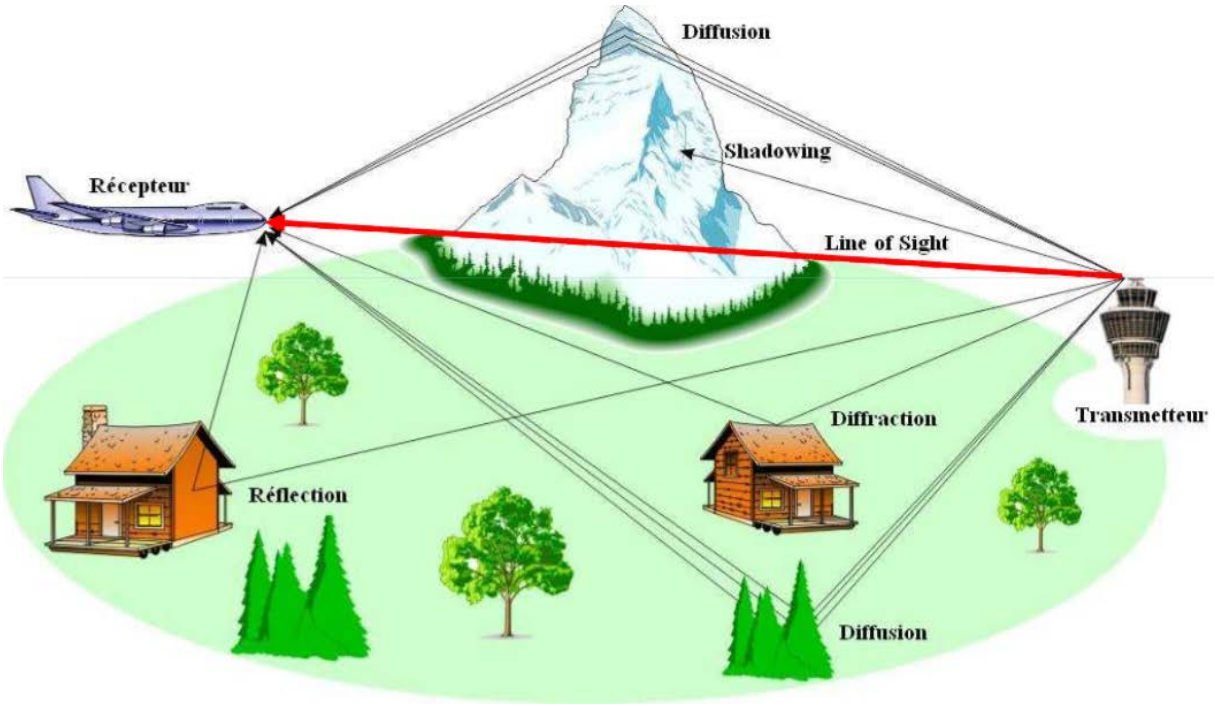


Figure 1.1 : Propagation par trajets multiples

Tous ces phénomènes physiques entraînent une série d'échos de provenances diverses et impossible de prévoir, (propagation par trajets multiples due à la présence d'obstacles) pouvant engendrer des évanouissements (fadings) qui sont des « trous de transmission » résultant de l'annulation du signal à un instant et une fréquence donnée, et ces échos d'amplitudes variables introduisent de retard variables. Par conséquent, lors d'une réception fixe par un mobil, la probabilité de recevoir uniquement une onde directe provenant d'un émetteur est très faible. En effet, en plus du signal reçu, une multitude de signaux d'échos atténues et retardés s'ajouter [1].

2.1 Effet des trajets multiples :

Dans un système de communication, pour qu'un signal arrive à sa destination, plusieurs trajets sont possibles. A la réception un symbole véhiculé car le signal peut être affecté par un autre symbole en retard. Si on considère que le signal partout le symbole n de deux trajets différentes et pour un retard connu entre les deux trajets. Alors, à la réception, les données sont démodulées en examinons toutes les informations reçues par rapport à ce symbole n.

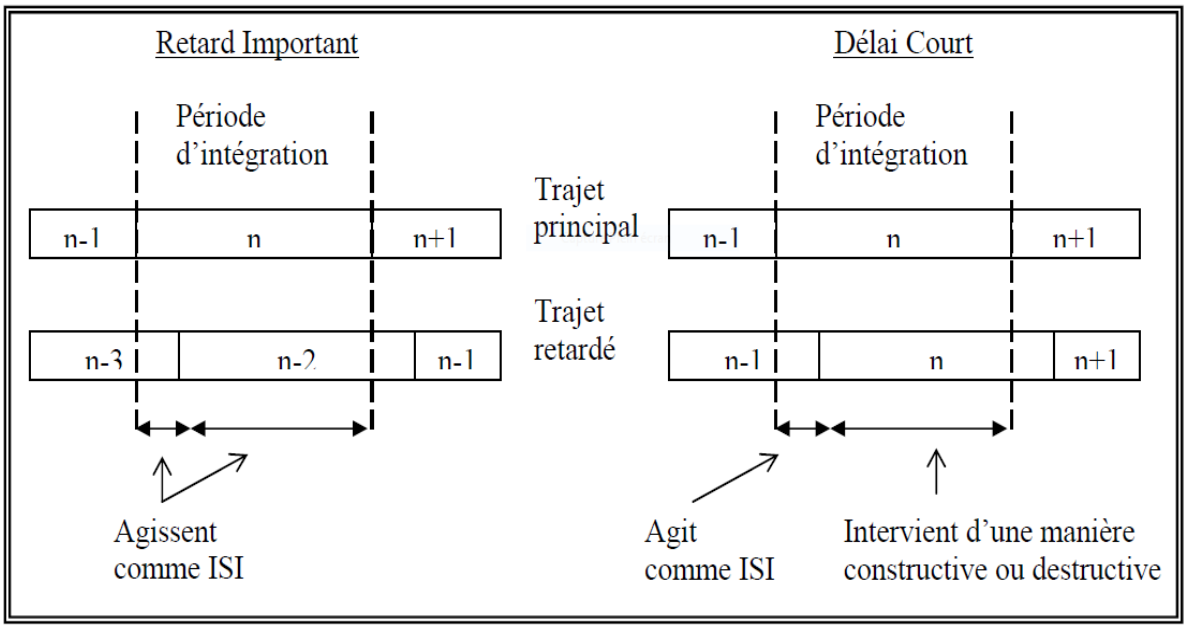


Figure 1.2 : Interférence inter symbole (ISI)

Lorsque le retard relatif est supérieur à une période de symbole (voir figure 1.2 à gauche), le signal provenant du second trajet agit uniquement comme un brouillage, puisqu'il n'achemine que des informations appartenant à un ou plusieurs symbole(s) précédent(s).

Un tel brouillage inter symbole implique que le signal retardé ne peut avoir qu'un niveau très faible car ce dernier a subi trop d'atténuation (le niveau exact dépendant de la constellation utilisée et de la perte de marge de bruit acceptable).

Lorsque le retard relatif est inférieur à une période de symbole (voir figure1.2 à droite), seule une partie du signal transmis sur ce trajet agit comme un brouillage, puisqu'elle n'achemine que des informations appartenant au symbole précédent. Le reste achemine des informations du symbole utile, mais peut s'ajouter de manière constructive ou destructive aux informations du trajet principal [1].

2.2 Effet Doppler :

L'étalement Doppler est causé par le mouvement relatif de l'émetteur et du récepteur. Par exemple, dans un environnement urbain au centre-ville, les véhicules sont toujours en mouvement, les piétons changent continuellement d'emplacement, de sorte que leurs mouvements affectent le moyen de transmission. Un Doppler élevé peut être ressenti lorsqu'un utilisateur se trouve dans une voiture en mouvement rapide ou dans un train rapide, parce que le mouvement relatif sera plus élevé lorsque l'émetteur ou le récepteur se déplace très rapidement. Ce mouvement relatif de l'émetteur et du récepteur modifie le signal reçu par rapport au signal transmis à l'origine. Lorsqu'ils se rapprochent l'un de l'autre, la fréquence du signal reçu est plus élevée que la source et lorsqu'ils s'éloignent l'un de l'autre, la fréquence reçue diminue. Lorsque la vitesse relative est plus élevée, le décalage Doppler peut être très élevé et, par

conséquent, le récepteur peut devenir incapable de détecter la fréquence du signal transmis. Même à un mouvement relatif plus faible lorsque le décalage Doppler est généralement très faible [2].

3. La sélectivité :

La fonction de transfert d'un canal résultant d'une propagation à trajets multiples présente une réponse fréquentielle qui n'est pas plate, mais comporte des creux et des bosses dus aux échos et réflexions entre l'émetteur et le récepteur. Un très grand débit impose une grande bande passante, et si cette bande couvre une partie du spectre comportant des creux, il y a perte totale de l'information pour la fréquence correspondante. Le canal est dit alors sélectif en fréquence.

Un canal est dit sélectif en fréquence lorsqu'il ne se comporte pas identiquement suivant la fréquence du signal. Certaines fréquences seront transmises plus rapidement que d'autres, ou encore seront atténuées plus que d'autres, le signal sera alors déformé lors de la transmission.

L'évanouissement sélectif en fréquence se produit lorsque le canal introduit une dispersion temporelle et que l'étalement du délai est plus grand que la période du symbole. L'évanouissement sélectif en fréquence est difficile à compenser parce que les caractéristiques d'évanouissement sont aléatoires et parfois difficilement prévisibles [2].
Ce phénomène de sélectivité en fréquence est aggravé par la présence de trajets multiples pour un même signal transmis.

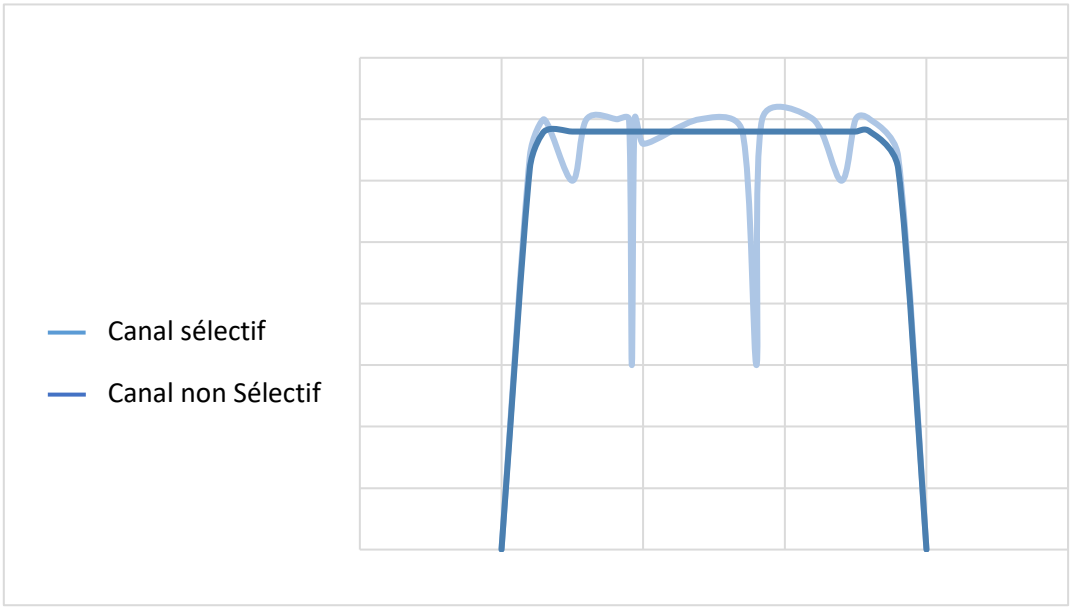


Figure 1.3: L'effet de canal sélectif

Pour remédier à ce désagrément, l'idée est de répartir l'information sur un grand nombre de porteuses, créant ainsi des sous-canaux très étroits pour lesquels la réponse fréquentielle du canal peut être

considérée comme constante. Ainsi, pour ces canaux, le canal est non sélectif en fréquence, et s'il y a un creux, il n'affectera que certaines fréquences, qui pourront être récupérées grâce à un codage convolutif.

4. OFDM (Orthogonal Frequency Division Multiplexing):

Pour répondre à un niveau non négligeable des signaux retardés, il faut réduire le débit de symboles pour que la gamme des retards (entre le premier trajet reçu et le dernier) ne représente qu'une partie minime de la période de symbole. Les informations susceptibles d'être acheminées par une porteuse unique sont dès lors limitées en cas de trajets multiples. Si une porteuse ne peut transporter le débit de symboles nécessaire, on arrive tout naturellement à diviser ce débit de données élevé en plusieurs flux parallèles de débit moins élevé, acheminés chacun par sa propre porteuse. Leur nombre peut être élevé. Il s'agit d'une forme de FDM (Multiplex par répartition en fréquence), première étape vers l'OFDM. Donc l'OFDM a été réservé pour une forme particulière de FDM.

L'OFDM (Orthogonal Frequency Division Multiplexing) est un procédé de codage des signaux numériques par répartition en fréquences orthogonales sous forme de multiples sous-porteuses. Donc au lieu d'utiliser une seule porteuse, modulée à haut débit, le système OFDM permet de diviser un signal numérique en un grand nombre de sous-porteuses, modulées individuellement a faible débit. Le faible débit de chaque porteuse => T_u (durée d'un symbole) long => diminue grandement l'interférence inter symbole (IIS) [3]. Cette technique permet de lutter contre les canaux sélectifs en fréquence en permettant une égalisation de faible complexité. Ces canaux particulièrement évidents en présence de trajets multiples et sont d'autant plus pénalisant que le débit de transmission est élevé. C'est la raison pour laquelle on trouve cette technique largement adoptée dans la plupart des applications à très haut débit.

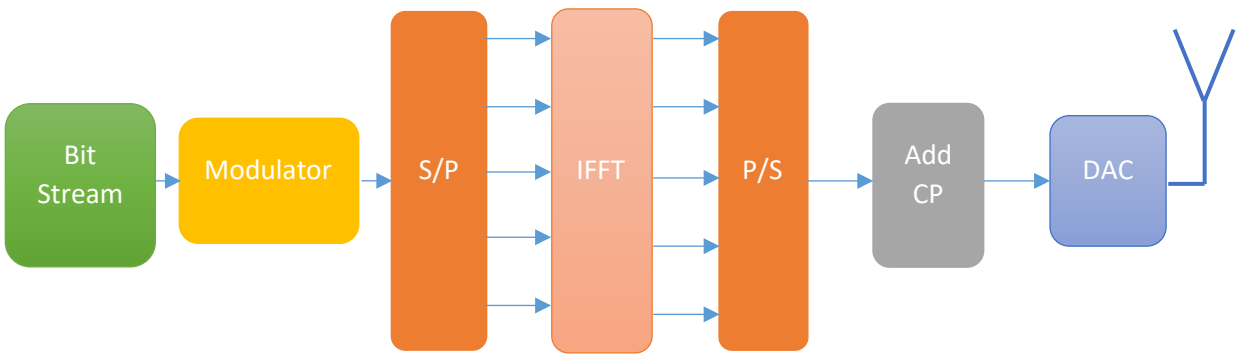


Figure 1.4: Bloc d'émission OFDM

5. L'orthogonalité :

Les signaux sont orthogonaux s'ils sont mutuellement indépendants les uns des autres. L'orthogonalité est une propriété qui permet la transmission parfaite de plusieurs signaux d'information sur un canal commun et leur détection sans interférence. La perte d'orthogonalité entraîne le flou entre ces signaux d'information et la dégradation du canal de communication. Dans le domaine des fréquences, la plupart des systèmes FDM (Frequency Division Multiplexing) sont orthogonaux, car chacun des signaux émis est bien espacé en fréquence pour éviter les interférences. Les sous-porteuses d'un système OFDM sont espacées le plus proche possible les unes des autres, mais l'orthogonalité est maintenue [4].

Cet espacement étroit de sous-porteuses donne une naissance d'interférence inter porteuses (IIP). Les porteuses doivent respecter une contrainte d'orthogonalité. La condition d'orthogonalité permet d'avoir un recouvrement entre les spectres des différentes sous-porteuses, et d'éviter les interférences entre les sous-porteuses.

Deux signaux limités dans le temps $(o \leq t \leq T_u)$ sont orthogonaux s'ils satisfont les conditions de l'équation (1.2). Si l'on spécifie un espacement rigoureusement régulier de $f_u = 1/T_u$ entre les porteuses [5], où T_u est la période (utile ou active) du symbole pendant laquelle le récepteur intègre le signal démodulé, les porteuses forment alors un ensemble orthogonal.

La $k^{ème}$ porteuse (en bande de base) peut s'écrire sous la forme:

$$S_K(t) = e^{jkw_u t} \qquad (1.1)$$

Où, $w_u = 2\pi/T_u$, et où les porteuses doivent satisfaire la condition d'orthogonalité:

$$\int_{\tau}^{\tau+Tu} S_k(t) * S_i(t)dt = \begin{cases} T_u, & k = i \\ 0, & k \neq i \end{cases} \qquad (1.2)$$

C'est la procédure de démodulation d'une porteuse qui consiste à la multiplier par une porteuse de même fréquence puis on intègre le résultat. Toutes les autres porteuses donneront des battements qui se situent à des multiples entiers de w_u. Tous ces battements (brouilleurs) ont un nombre entier de cycles pendant la période d'intégration T_u. Leur intégrale est donc nulle [6].

On peut démoduler séparément toutes les porteuses sans aucune diaphonie mutuelle, simplement en choisissant leur espacement. Ainsi, on ne gaspille pas de spectre et les porteuses sont situées de manière à occuper la même largeur de spectre qu'une seule porteuse modulée avec toutes les données.

- **L'orthogonalité fréquentiel:**

On peut percevoir la notion d'orthogonalité du signal OFDM dans le domaine fréquentiel. Chaque sous-porteuse OFDM a une réponse en fréquence sinus cardinal, sin (x)/x. La forme sinus cardinal a un lobe principal étroit, avec de nombreux lobes latéraux.

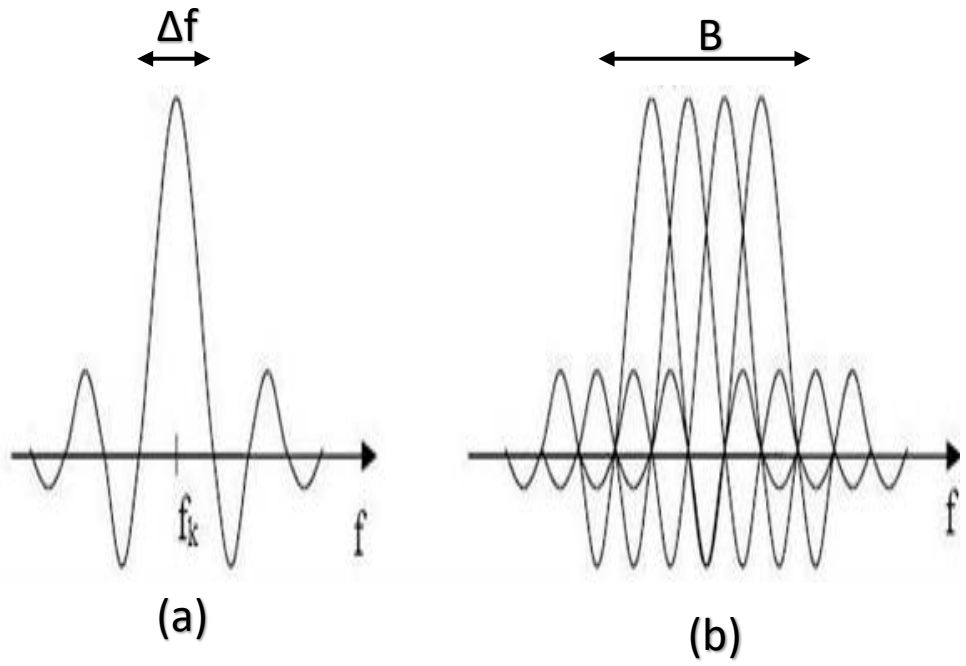


Figure 1.5: (a) spectre d'une sous porteuse (b) spectre d'un signal OFDM

L'orthogonalité dans le domaine fréquentiel est réalisée lorsque le maximum de chaque sous-porteuse correspond à un "zéro" des autres. Cette condition permet ainsi d'avoir une occupation spectrale idéale et d'éviter les interférences entre sous-porteuses [7]. Le spectre total du signal OFDM est la somme des spectres individuels des différentes porteuses.

La figure 1.5(b) montre qu'alors, la bande en fréquence est occupée de façon optimum, puisque le spectre est presque plat dans cette bande. La bande occupée est à peu près :

$$B = \frac{N}{T_u}$$

Chaque sous-porteuse occupe à peu près $\frac{1}{T_u}$.

6. Conversion série/parallèle :

Les données à transmettre prennent généralement la forme d'un flux de données en série. Par conséquent, un étage de conversion S/P est nécessaire pour convertir le flux de bits série d'entrée en données parallèle à transmettre dans chaque symbole OFDM, comme le montre la figure 1.6 [4].

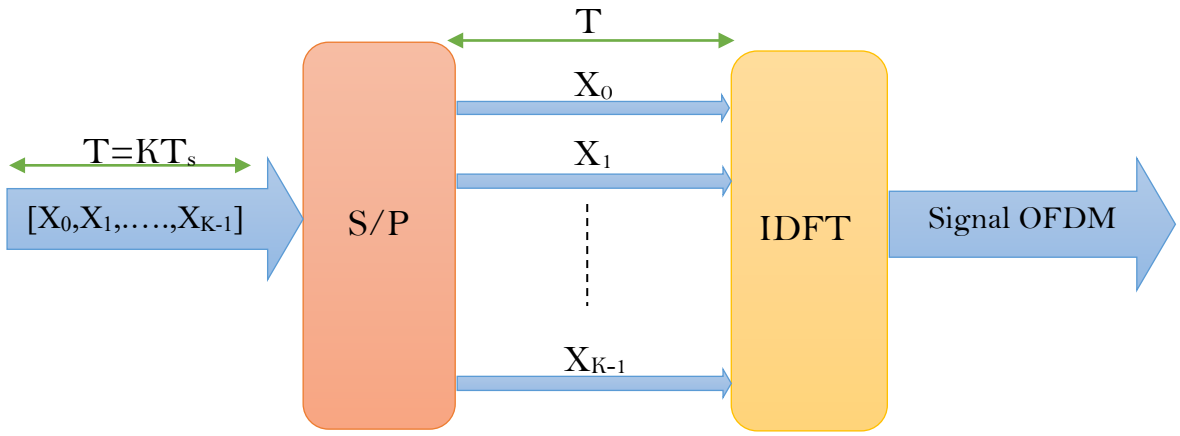


Figure 1.6: Génération d'un signal OFDM par conversion S/P

Les données attribuées à chaque symbole dépendent du schéma de modulation utilisé et du nombre de sous-porteuses.

7. Modulation des sous-porteuses :

Dans chaque symbole OFDM, chaque sous-porteuse est modulée (multipliée) par un nombre complexe pris dans un ensemble de la constellation (QPSK, 16-QAM, 64-QAM). On peut augmenter le débit binaire en augmentant le nombre d'états dans une constellation. Mais plus la constellation comporte d'états, plus chaque porteuse peut acheminer de bits pendant un symbole, et ainsi les points de la constellation sont rapprochés. Par conséquence si on suppose une puissance transmise constante la sensibilité au bruit augmente [8]. Un symbole très bruité et donc éloigné de son emplacement d'origine, peut être confondu avec le symbole adjacent.

Parmi les modulations utilisées dans un système OFDM, la plus robuste est la modulation QPSK (Quaternary Phase Shift Keying) ou modulation à déplacement de phase à 4 états), qui code 2 bits par symbole. Les deux porteuses, appelées **I** (réel) et **Q** (imaginaire), sont déphasées de 90°. On peut résumer cela de la manière suivante:

$$S(t) = I(t)cos\ (\omega t)\ +\ Q(t)sin\ (\omega t) \qquad (1.3)$$

Les 4 états (modulation QPSK) de la porteuse sont représentés sur un diagramme appelé constellation des états (voir la Figure 1.7). Sur la figure on voit que *I(t)* valant $+X$ pour un bit à 0 et $-X$ pour un bit à 1 et Q (t) valant $+Y$ pour un bit à 0 et $-Y$ pour un bit à 1. Un symbole transmis contient deux bits d'informations.

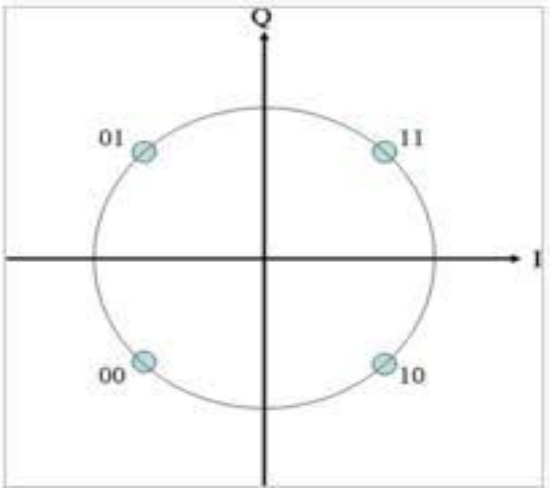


Figure 1.7: Constellation de modulation QPSK

On voit que les 4 états de la porteuse sont bien définis lors de la modulation. Au niveau de réception le signal reçu est souvent faible et fortement bruité, et la constellation des états en sortie du démodulateur est nettement mauvaise. Les constellations sont devenues 4 nuages de points (voire la figure 1.8) à cause de bruit lors le passage dans un canal de transmission. Le processus de la démodulation s'effectuera grâce à un module de décision qui estimera le symbole le plus probable même si les points deviennent des tâches, à cause du bruit. Donc il suffit de savoir dans quel quadrant se situe le point pour retrouver l'information.

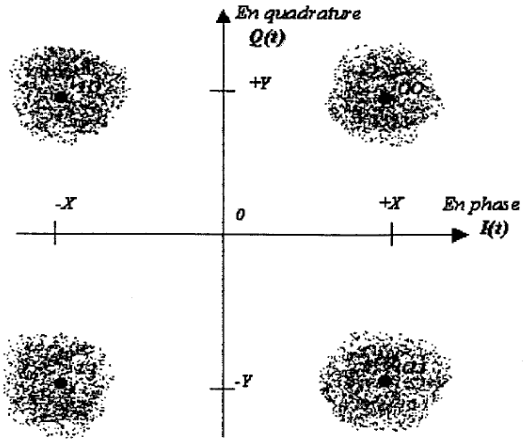


Figure 1.8: Constellation au niveau de réception

Si on parle d'une modulation à efficacité maximal, c'est-à-dire transportant un maximum de bit par symbole. On utilise une modulation QAM (Quadrature Amplitude Modulation) composée de deux modulations d'amplitude à porteuse supprimée en quadrature comme pour le QPSK.

Le fait que l'on augmente le rendement du code en augmentant le nombre de bits codés par symbole, donc en augmentant le débit binaire [9].
Si chaque axe code deux bits, on transporte alors 4 bits par symbole et l'on définit une modulation 16 QAM (représentée sur la Figure 1.9) :

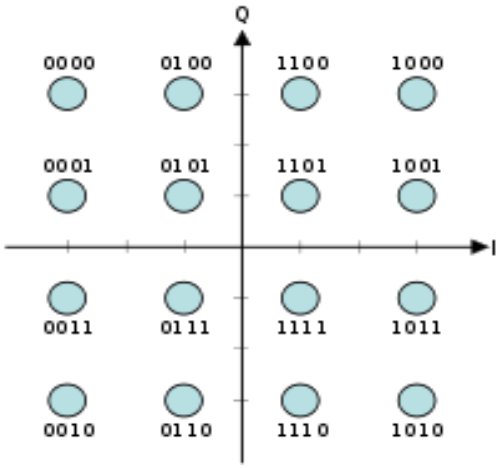


Figure 1.9: Constellation de la modulation 16QAM

On remarque à la figure que, dans chaque quadrant de la constellation, les deux bits de poids fort *I – Q,* qu'ont une valeur constante sur le quadrant et que cela s'apparente alors à une modulation QPSK.
A la réception, il suffit de détecter le quadrant dans lequel se trouve le symbole transmis pour retrouver la valeur de *I -Q.*

Dans les systèmes de modulation à constellation rectangulaire comme QPSK, 16-QAM ou 64-QAM, on utilise un codage de Gray. Dans ce code, le passage d'un nombre au suivant nécessite le changement d'un seul bit.

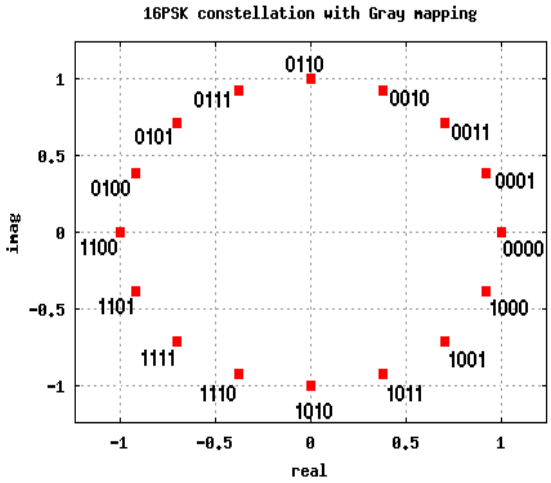


Figure 1.10: Codage de Gray de la constellation 16QAM

8. IFFT et FFT :

Dans la pratique, les systèmes OFDM sont mis en œuvre en utilisant une combinaison de blocs de transformée de Fourier rapide (FFT) et de transformée de Fourier rapide inverse (IFFT) qui sont des versions mathématiquement équivalentes de la DFT et de l'IDFT, respectivement, mais plus efficaces à mettre en œuvre. Un système OFDM traite les symboles sources (par exemple, les symboles QPSK ou MAQ qui seraient présents dans un système à porteuse unique) à l'émetteur comme s'ils étaient dans le domaine fréquentiel. Ces symboles sont utilisés comme entrées d'un bloc IFFT qui amène le signal dans le domaine temporel. L'IFFT prend en compte N symboles à un moment où N est le nombre de sous-porteuses dans le système. Chacun de ces N symboles d'entrée a une période de symbole de T secondes. Rappelons que les fonctions de base d'une IFFT sont N sinusoïdes orthogonales. Ces sinusoïdes ont chacune une fréquence différente et la fréquence la plus basse est DC. Chaque symbole d'entrée agit comme un poids complexe pour la fonction de base sinusoïdale correspondante. Comme les symboles d'entrée sont complexes, la valeur du symbole détermine à la fois l'amplitude et la phase de la sinusoïde pour cette sous-porteuse. La sortie IFFT est la somme de toutes les N sinusoïdes. Ainsi, le bloc IFFT fournit un moyen simple de moduler les données sur N sous-porteuses orthogonales. Le bloc de N échantillons de sortie de l'IFFT constitue un seul symbole OFDM. La longueur du symbole OFDM est NT où T est la période du symbole d'entrée IFFT [10].

Après quelques traitements supplémentaires, le signal dans le domaine temporel qui résulte de l'IFFT est transmis à travers le canal. Au niveau du récepteur, un FFT est utilisé pour traiter les données reçues et l'amener dans le domaine fréquentiel. Idéalement, la sortie FFT sera les symboles originaux qui ont été envoyés à l'IFFT à l'émetteur. Lorsqu'ils sont tracés dans le plan complexe, les échantillons de sortie FFT forment une constellation, comme la MAQ. Cependant, il n'y a pas de constellation pour le signal dans le domaine temporel. Ainsi, tout traitement du récepteur qui utilise le concept de constellation

(comme le découpage de symboles) doit avoir lieu dans le domaine fréquentiel. Le schéma fonctionnel de la figure 1.11 illustre le passage du domaine fréquentiel au domaine temporel dans un système OFDM.

Figure 1.11: Schéma fonctionnel d'un système OFDM simple

9. Modulation RF :

En général, les modulateurs et les démodulateurs réels du system OFDM fonctionnent soit en bande de base, soit à une fréquence intermédiaire (FI). Nous devons émettre notre signal à une fréquence radioélectrique (RF) prédéterminée. Dans la pratique on doit déplacer le signal modulé vers le haut pour le porter à cette fréquence (RF) à l'émission. A la réception, on descend de la RF à la FI ou à la bande de base. Ceci peut être mis en application en utilisant des techniques analogiques ou on utilise un convertisseur numérique. Cette opération nécessite des oscillateurs à déplacement de fréquence (OL), dont la fréquence est ultérieurement stabilisée grâce à une boucle à verrouillage de phase (PLL).

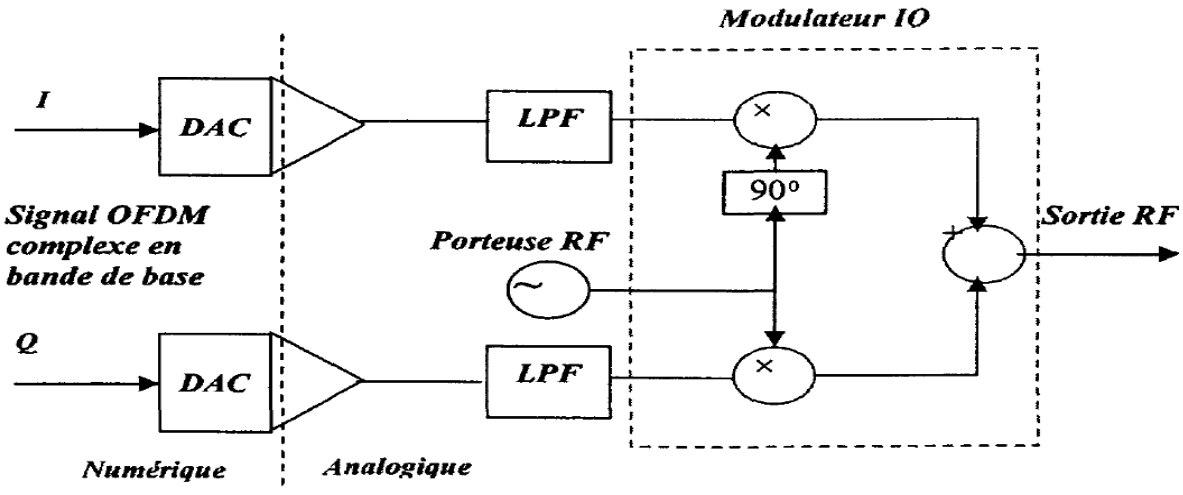


Figure 1.12: Modulation RF du signal complexe OFDM (en bande de base) avec des techniques analogiques

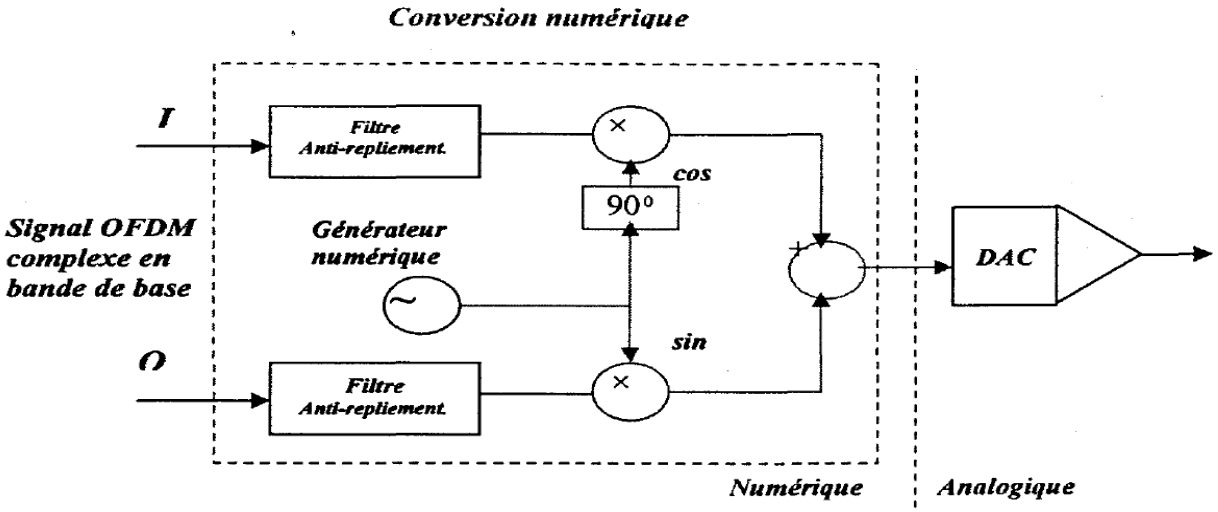


Figure 1.13: Modulation RF du signal complexe OFDM (en bande de base) avec des techniques numériques

Les synthétiseurs (générateurs de signaux RF) permettent la modulation avec des signaux I/Q pour obtenir des modulations QPSK, OQPSK, 8PSK, QAM... afin de répondre aux besoins croissants d'informations transmises (débit binaire) dans une bande passante donnée. L'amplification, quel que soit son type, n'est jamais parfaitement linéaire. Il est donc impératif de filtrer après amplification [11].

10. Intervalle de garde :

Une même suite de symbole arrivant à un récepteur par deux chemins différents se présente comme une même information arrivant à deux instants différents, elles vont donc s'additionner provoquant ainsi les deux types de défauts suivants :

- L'interférence intra symbole: Addition d'un symbole avec lui-même légèrement déphasé.
- L'interférence inter symbole: addition d'un symbole avec le suivant plus le précédant légèrement déphasé.

Une des raisons importantes pour lesquelles on utilise le même terme OFDM est sa robustesse face aux problèmes de transmission dans un environnement à trajets multiples. On rajoute un intervalle de garde à chaque symbole OFDM pour éliminer les interférences inter symboles (IIS). Il faut que la durée de cet intervalle doit- être supérieur à la durée de retard maximal causé par les phénomènes de propagation à trajets multiples et qu'un symbole ne puisse pas interférer avec le prochain symbole [11].

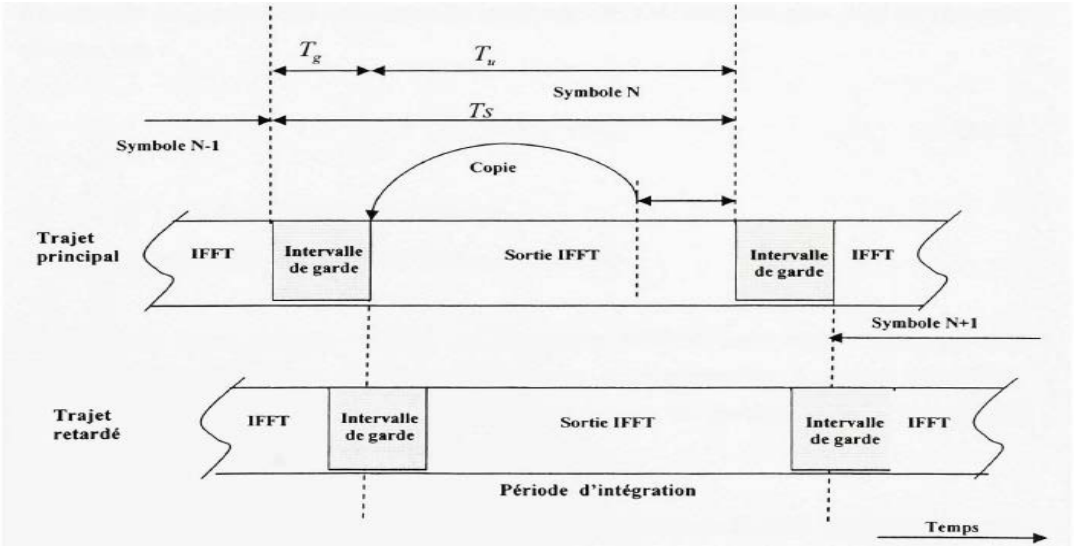


Figure 1.14: Intervalle de garde par prolongation cyclique

Plusieurs techniques différentes existent pour l'introduction des intervalles de gardes dans OFDM. Il s'agit soit de remplir l'espace de garde entre deux symboles consécutif par des zéros, soit d'introduire une extension cyclique du symbole OFDM [12].

L'extension cyclique peut être introduite de deux façons différentes. L'une nommée préfixe cyclique et l'autre suffixe cyclique.

10.1 Préfixe cyclique :

Le principe du préfixe cyclique est de copier les derniers échantillons d'un symbole et les placer à son début. En utilisant cette technique, on garde une continuité dans le symbole.
Le préfixe cyclique est une bonne méthode pour combattre les interférences entre symboles (IES) et entre porteuses (IEP). Grâce à cette extension, la période du symbole est plus longue. Précisons que bien que la période soit plus longue, Cela n'a aucun incident sur le spectre fréquentiel du signal. Aussi longtemps que le bon nombre d'échantillons est pris en réception n'importe où dans le symbole rallongé. L'orthogonalité est maintenue et des interférences éliminées.

$$T_s = T_u + T_g$$

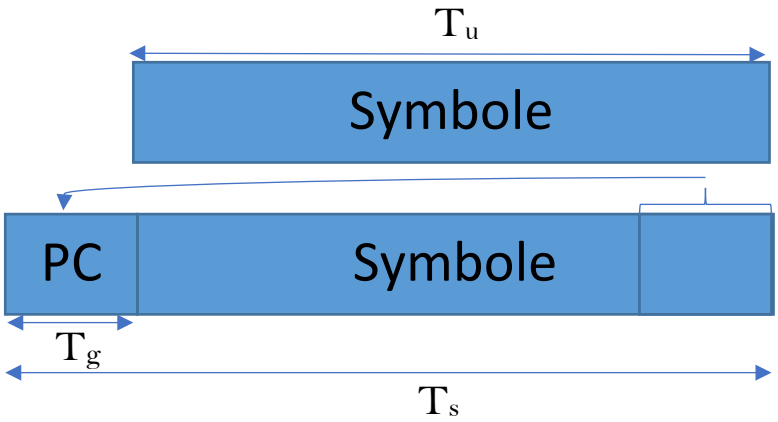


Figure 1.15: Insertion de préfixe cyclique

Il est important de savoir que la longueur de préfixe doit être défini en fonction des délais maximums de retard prévus dans le canal multi trajet. En effet, l'idée n'est pas qu'on résolve uniquement le problème d'interférence en utilisant le préfixe, mais aussi d'effectuer une utilisation la plus optimale possible de la bande passante. L'information répétée dans le préfixe crée un gaspillage de bande passante qu'il est important d'amoindrir au maximum [12].

10.2 Suffixe cyclique :

Le suffixe cyclique est aussi une extension cyclique du symbole OFDM. C'est le même principe de rajout que précédemment sauf que dans ce cas, le suffixe est intégré à la fin du symbole OFDM. Il est utilisé pour éviter l'interférence entre le flot montant et celui descendant dans une communication. Pour choisir sa longueur, on calcule la différence de temps de réception entre le flot montant et descendant, tandis que comme vu précédemment, le préfixe cyclique lui est choisi de façon à couvrir le temps de dispersion du canal [12].

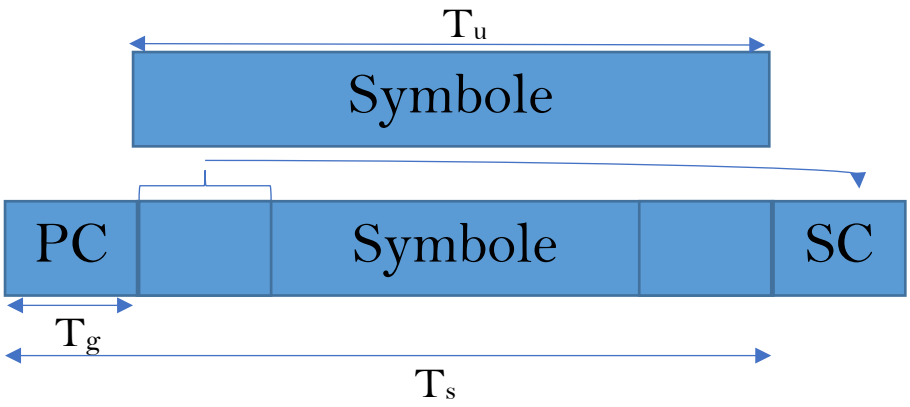


Figure 1.16: L'intérêt de l'intégration de suffixe et préfixe

11. Bande de garde :

Le spectre du signal sur chaque sous-porteuse est une fonction sinus cardinal avec une bande de $\frac{2}{T_s}$. La densité de puissance spectrale d'un signal OFDM est alors la somme de plusieurs fonctions sinus cardinal décalées par conséquence, la puissance hors bande devient significative. Cette puissance provoque l'apparition de l'interférence entre canal adjacents(ACI). Cela nécessite d'utiliser une bande de garde pour réduire l'effet de l'ACI dans un système OFDM. Deux techniques peuvent être utilisées pour réduire la puissance hors bande [13].

Un fenêtrage dans le domaine temporel du symbole OFDM par une fonction de type cosinus surélevé (RC: Raised Casinos).

Une autre mesure est l'utilisation des porteuses virtuelles (VC: Virtuel Carriers) qui sont des sous porteuses non utilisées placées aux deux bouts de la bande de transmission. Cependant, l'efficacité spectrale sera réduite de $N_u/2$ fois à cause des sous porteuses non utilisées, (N_u est le nombre de sous porteuses utilisées pour la transmission). Les deux techniques RC et VC peuvent être combinées pour combattre l'interférence entre canal (ACI).

12. Fenêtrage de signal OFDM :

Le signal OFDM est la somme de sous-porteuses modulées, pondérées par une fenêtre rectangulaire qui définit la durée de chaque symbole OFDM. La FFT de cette fenêtre possède un lobe principal et des lobes secondaires qui sont en dessous de celle de son lobe principal. La largeur du lobe central détermine la résolution spectrale de la fenêtre, donc sa capacité de se distinguer deux fréquences proches l'une de l'autre. L'amplitude des lobes latéraux détermine l'étalement spectral de la fenêtre. Ces lobes secondaires augmentent la largeur de bande du signal OFDM, dégradant l'efficacité spectrale [11].

Pour rendre le niveau des lobes secondaires aux limites acceptables, on utilise un filtrage du signal OFDM ou à utiliser sciemment du signal OFDM par une fenêtre de pondération dont la largeur du lobe principal et la position des lobes secondaires sont les caractéristiques fréquentielles. Les fenêtres utilisées le plus couramment sont :

- **Hanning :**

$$W(k) = 0.50 - 0.5\,cos\,(2\pi k/N) \qquad \text{pour} \quad n=0\ldots N\text{-}1$$

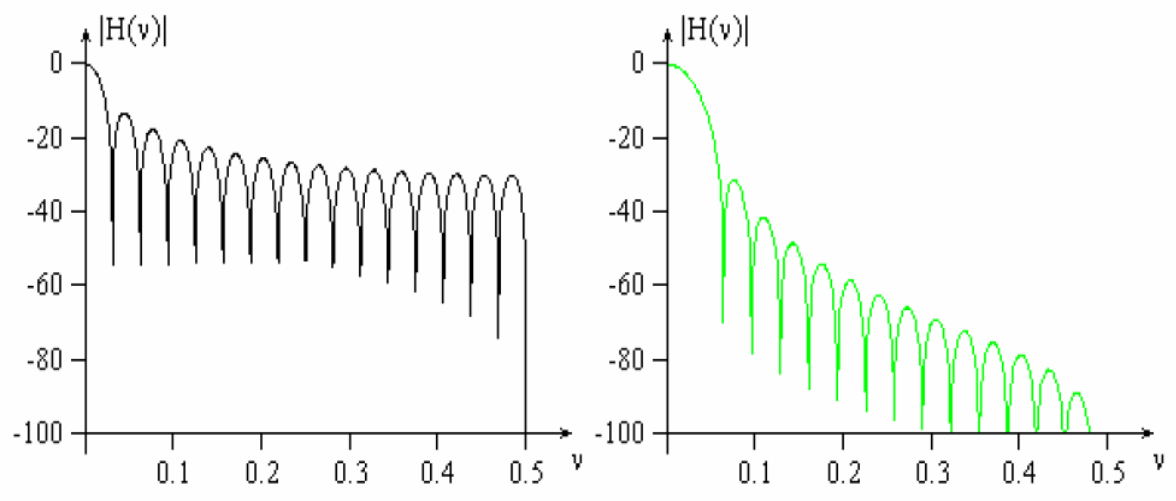


Figure 1.17: Fonctions de transfert en dB correspondant à la fenêtre naturelle (en noir à gauche) et à la fenêtre de Hanning (en vert à droite)

- **Hamming :**

$$W(k) = 0.54 - 0.46\cos(2\pi k/N) \qquad \textit{pour} \quad n=0...N-1$$

- **Blackman :**

$$W(k) = 0.42 - 0.50\cos(2\pi k/N) + 0.08\cos(2\pi k/N) \qquad \textit{pour} \quad n=0...N-1$$

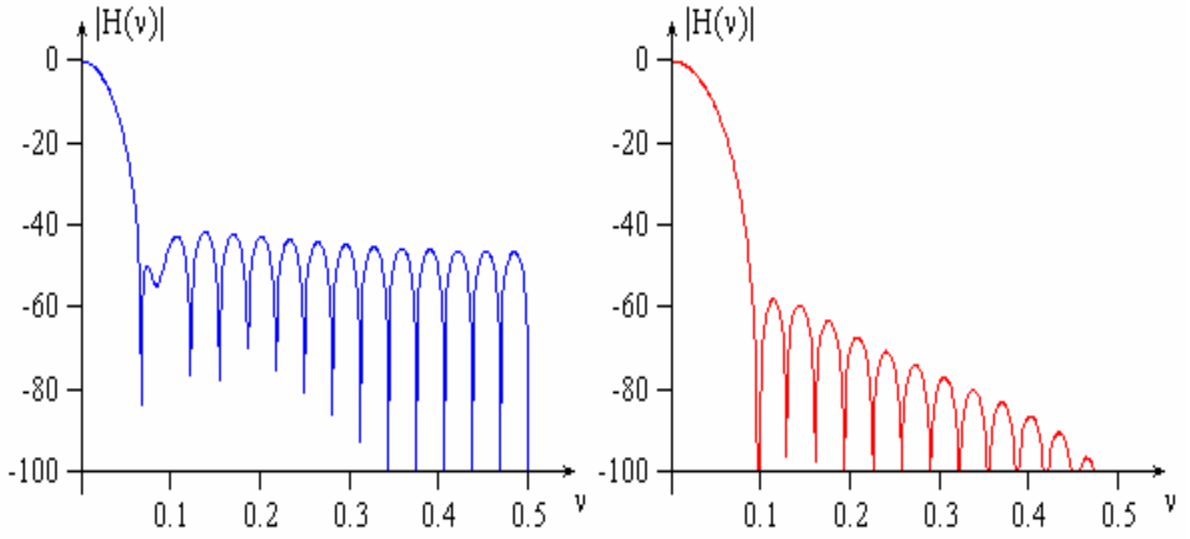


Figure 1.18: Fonctions de transfert en dB correspondant à la fenêtre Hamming (en bleu à gauche) et à la fenêtre de Blackman (en rouge à droite)

Les fenêtres permettent de réduire la hauteur et l'énergie présente dans les lobes secondaires. La largeur du lobe principal et le niveau des lobes secondaires pour ces différentes fenêtres sont données au Tableau (I).

Tableau (I)

Propriétés des fenêtres de pondération usuelles

Type	Largeur du lobe principal (en F)	Niveau des lobes secondaires	Pente
Hanning	1.5/N	-30dB	-18dB
Hamming	2/N	-40dB	-6dB
Blackman	2.75/N	-60dB	-18dB

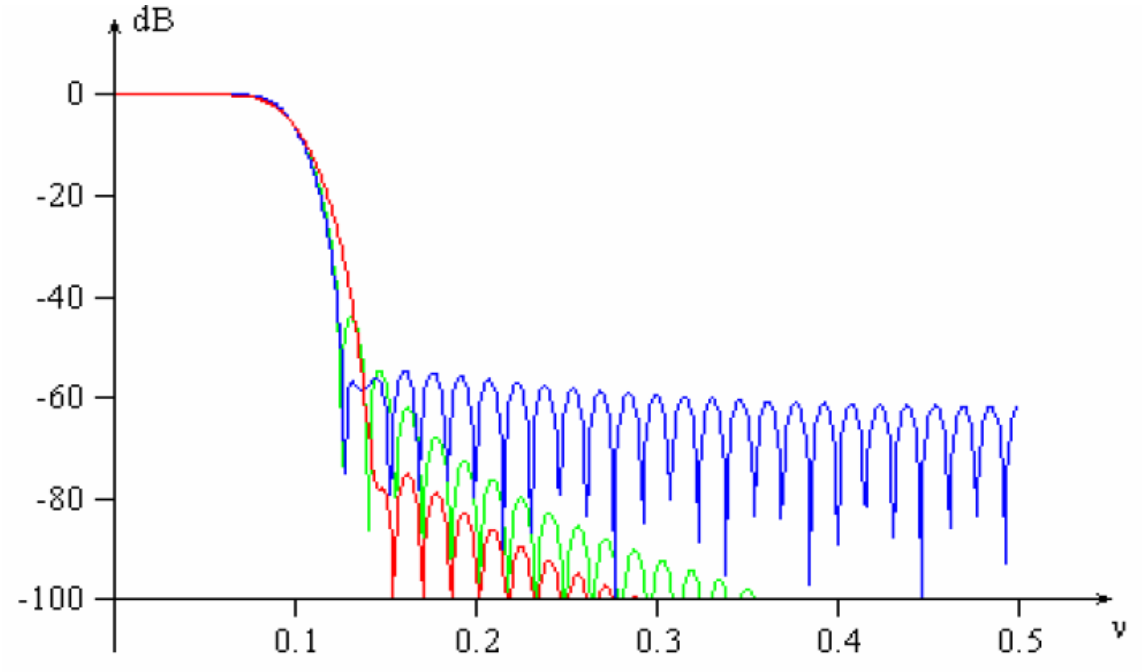


Figure 1.19: Fonction de transfert d'un filtre passe-bas numérique pondéré par Hanning (vert), Hamming (bleu) et Blackman (rouge)

13. Conclusion :

Dans ce chapitre, après avoir abordé les canaux à trajets multiples et les perturbations causées par ces trajets multiples, nous avons présenté la technique OFDM qui apporte une solution à ce problème crucial que rencontre la transmission à débit élevé et ceci en divisant la bande de transmission en N sous bandes orthogonales.

CHAPITRE II

Programmation de Raspberry Pi à l'aide de Simulink

1. Introduction :

Le mini-ordinateur Raspberry Pi est polyvalent et peut être utilisé pour la réalisation de nombreux projets.

Dans ce chapitre nous commençons par présenter le Raspberry Pi et ses différentes versions existantes. Ces informations sont nécessaires afin de nous permettre d'utiliser le Raspberry Pi dans notre projet. Puis nous présentons toutes les étapes nécessaires à l'installation de support package du Simulink pour le Raspberry Pi.

2. Description de Raspberry Pi :

Figure 2.1 : Carte Raspberry Pi

RASPBERRY Pi est un ordinateur intégré, fabriqué à partir d'une seule puce électronique contenant les composants traditionnels de l'ordinateur, un processeur à un seul cœur de 700MHZ et un GPU double cœur de 250MHZ capable de lire des films haute définition, avec une RAM qui peut aller jusqu'à 512MHZ [14].

En plus des sorties de contrôle numériques qui peuvent contrôler diverses composants électriques et électroniques tels que les microcontrôleurs, toutes ces capacités sont sur une micro puce, connue sous le nom de SOC (Système On Chip). Ce petit ordinateur fonctionne sur des systèmes Linux open source.

Il a une taille de 8.6 cm x 4.5 cm et pèse environ 45 grammes, ce qui en fait l'un des panneaux d'ordinateur les plus légers dans la planète.

On peut utiliser RASPBERRY Pi comme n'importe quel ordinateur traditionnel pour surfer sur internet et envoyer des e-mails et même éditer des fichiers via le bureau libre office… etc. On peut par exemple faire [15]:

- La conception de système de contrôle de maison intelligente « Smart Home Auto motion ».
- La fabrication de robots de sous-marines et de drones « ROV and UAV Robots ».
- Des applications de surveillance telles que des caméras de travail pour diffuser des vidéos et des images à distances « Caméra Streamers ».
- Surveillance de l'environnement tel que le système de surveillance de la température et de l'humidité à distance « Remote Monitor ».
- Smart TV.
- Différents serveurs Linux tel que Linux : serveur http, FTP, MSQL, SSH, etc…
- Supercalculateurs « Super computers ».

L'utilisation de RASPBERRY Pi donne de meilleures opportunités que de voir seulement les résultats de simulation. De plus, il encourage la pensée créative et l'expérimentation de nouvelles applications. Cependant, le dispositif d'application comme RASPBERRY Pi a des capacités limitées et donc le logiciel doit être développé en gardant à l'esprit les limitations.

3. L'historique de Raspberry Pi :

RASPBERRY Pi, tel qu'illustré à la figure 2.1, est un ordinateur de bord unique ayant un format aussi petite qu'une carte de crédit de la fondation RASPBERRY Pi. Une idée de production RASPBERRY Pi a commencé en 2006 avec la prise de conscience de l'absence de la jeune génération des connaissances sur le fonctionnement de l'ordinateur. Un groupe d'universitaires et d'ingénieurs de l'Université d'Ottawa. L'Université de Cambridge a donc décidé de mettre au point un très petit ordinateur qui permet d'effectuer les tâches suivantes que tout le monde peut se permettre d'acheter pour créer un environnement d'apprentissage dans la programmation. Le RASPBERRY Pi projet est devenu prometteur avec l'apparition de bon marché et puissant des processeurs mobiles avec de nombreuses fonctionnalités avancées permettant un développement possible de RASPBERRY Pi qui s'est poursuivi sous la fondation RASPBERRY Pi spécialement créée à cet effet avec le premier produit lancé en 2012 [14].

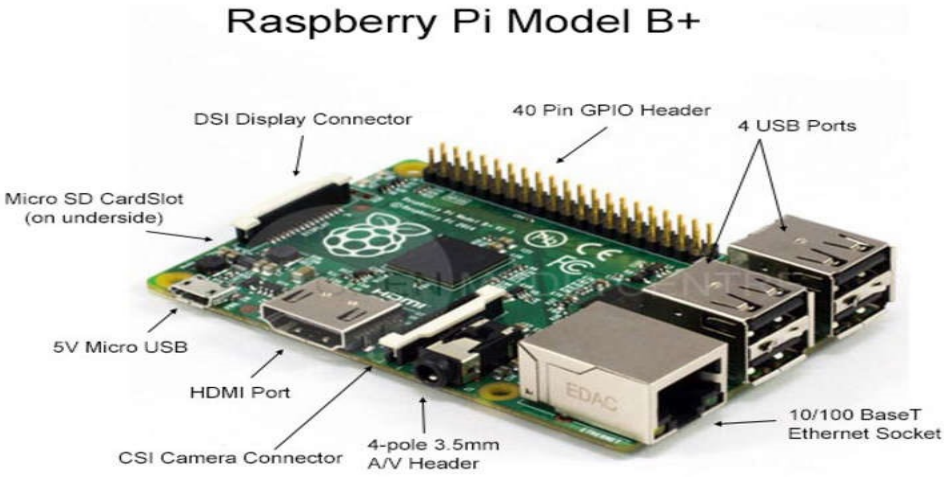


Figure 2.2 : Les portes de Raspberry Pi

Le développement de mini-ordinateur RASPBERRY Pi est un processus continu. Jusqu'au printemps 2015, la fondation RASPBERRY Pi a publié cinq modèles en deux générations d'ordinateurs. La première génération se compose de quatre modèles, comme le montre la figure 2.2. D'abord est le modèle B suivi respectivement du modèle A, du modèle B+ et du modèle A+. Ces deux derniers modèles sont des versions mises à jour de leurs versions précédentes pour rendre l'ordinateur plus efficace et plus pratique pour les utilisateurs, en particulier en ayant une consommation d'énergie plus faible et plus de ports USB (Universal Serial Bus). La deuxième génération se compose d'un seul modèle appelé Raspberry Pi 2 modèle B dont la spécification est basée sur Raspberry Pi modèle B+ mais avec un CPU plus rapide et plus de mémoire [14].

RASPBERRY Pi modèle B, comme le montre la figure 2.(b), a été publié au début de 2012 avec la spécification de 256 Mo de RAM, deux ports USB et un port Ethernet. Plus tard dans la même année, dans une nouvelle version, la quantité de RAM a été augmentée à 512 Mo, suivie de la sortie de la carte à spécifications inférieures, modèle A, comme le montre la Figure 2.3(a). Le modèle A est sorti avec la même quantité de RAM que l'ancien modèle B à 256 Mo, mais avec un port USB et pas de port Ethernet. Ces ordinateurs de première génération utilisent Broadcom SoC, BCM2835, qui intègre un processeur ARM1176JZF-S à cœur unique de 700 MHz, un GPU VideoCore IV et une variété de périphériques. Alors que le modèle B peut être utilisé dans n'importe quelle application, le modèle A, moins cher, est utile dans des applications spécifiques qui nécessitent un poids léger et une faible consommation d'énergie, comme la robotique ou tout service de médias portables.

En 2014, une version améliorée des deux modèles précédents a été lancée, sous les noms B+ et A+, comme le montrent les figures 2.3(c) et 2.3(d). Bien que les deux nouveaux modèles aient le même

processeur, GPU et RAM que leurs versions précédentes, plusieurs mises à jour et de nouvelles fonctionnalités ont été ajoutées, telles qu'une prise mémoire micro SD remplaçant une fente pour carte SD, une version améliorée de GPIO de 26 à 40 broches, un son à faible bruit et une consommation d'énergie inférieure de 0,5 à 1 W en ayant un régulateur à découpage au lieu d'un régulateur linéaire. De plus, le modèle A+ est d'environ 2 cm plus court que son prédécesseur, tandis que le modèle B+ fournit quatre ports USB au lieu de deux dans le modèle B.

Au début de l'année 2015, une nouvelle génération d'ordinateur RASPBERRY Pi appelée RASPBERRY Pi modèle B a été lancée. Le modèle représente une amélioration significative des capacités de RASPBERRY Pi et a été accueilli avec enthousiasme par la communauté. RASPBERRY Pi modèle B utilise un BCM2836 beaucoup plus puissant qui offre 900 MHz quadri-cœur ARM Cortex-A7 CPU, 1 Go de RAM et les mêmes spécifications de processeur graphique qu'auparavant, ce qui lui permet de fonctionner environ six fois plus vite que ses prédécesseurs à un prix inchangé. De plus, les supports de la version Windows 10 Internet of Things (IoT) est disponible pour RASPBERRY Pi modèle B gratuitement offrant un grand potentiel d'utilisation future.

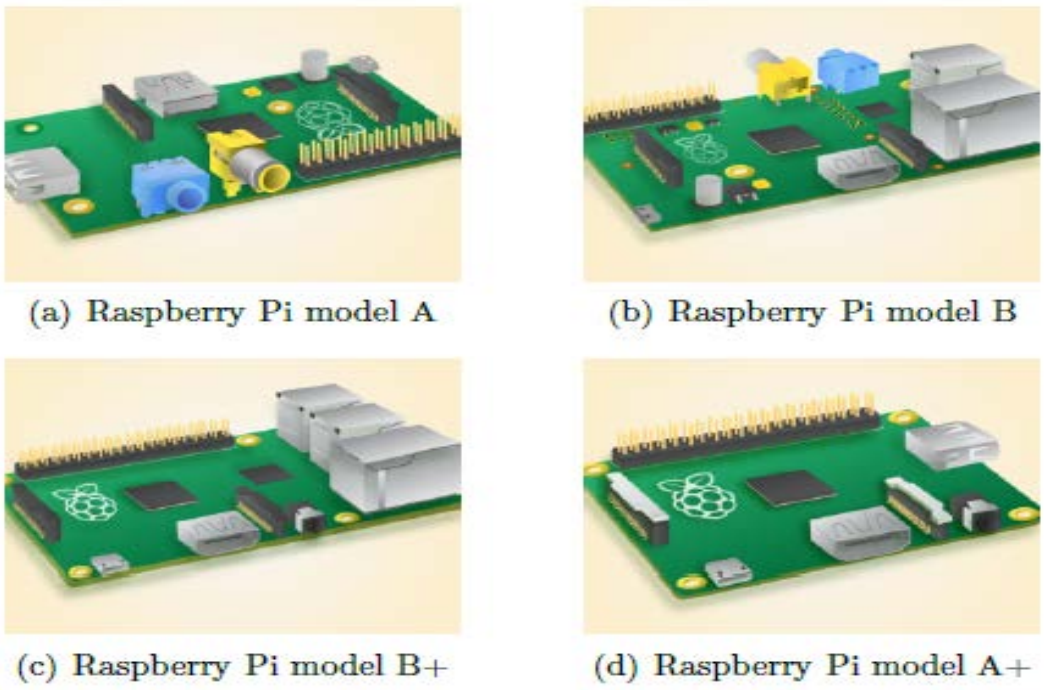


Figure 2.3 : Les modèles de Raspberry Pi

Tableau (II) Les principales caractéristiques des modèles RASPBERRY Pi

Raspberry Pi	Generation 1				Generation 2
Specification	**Model A**	**Model B**	**Model A+**	**Model B+**	**Model B**
Power	300mA	700mA	200mA	600mA	900mA
Ethernet Port	No	Yes	No	Yes	Yes
USB Port	1	2	1	4	4
GPIO	26	26	40	40	40
SD Carte Slot	SD	SD	MicroSD	MicroSD	MicroSD
SoC	BCM2835	BCM2835	BCM2835	BCM2835	BCM2836
CPU	700MHz ARM11 Single-core	700MHz ARM11 Single-core	700MHz ARM11 Single-core	700MHz ARM11 Single-core	900MHz ARM Cortex-A7 Quad-core
RAM	256MB	512MB	256MB	512MB	1GB

Et chaque nouveau modèle de Raspberry pi s'améliore et s'enrichie, et ce qui nous donne un grand nombre de choix, pour nos applications ou nos projets.

4. Composants de base :

4.1 Ports USB :

Le RASPBERRY Pi possède des ports USB permettent de connecter des claviers, des souris, des dongles WiFi et des clés USB. Comme les ports ne fournissent pas beaucoup d'énergie, si on veut ajouter un hub USB au Raspberry Pi, on doit trouver un qui vient avec une alimentation externe.

Figure 2.4 : Ports USB

4.2 Port Ethernet :

Ce port sert à connecter le Raspberry pi à l'Internet via un câble Ethernet, il est semblable à celui qu'on trouve à l'arrière d'un routeur. Cette méthode est plus facile à mettre en place que le WiFi et peut fournir un accès Internet plus rapide, mais on est limité par la longueur du câble.

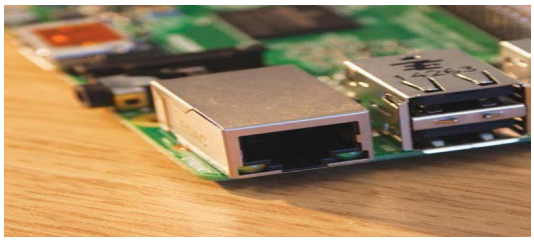


Figure 2.5 : Port Ethernet

4.3 Connecteur audio:

C'est une sortie audio via une prise casque de 3,5 mm qui prend également en charge une sortie vidéo analogique pour les modèles A+, B+ et RASPBERRY Pi modèle B. Pour les modèles précédents, A et B, le connecteur n'est que pour une sortie audio.

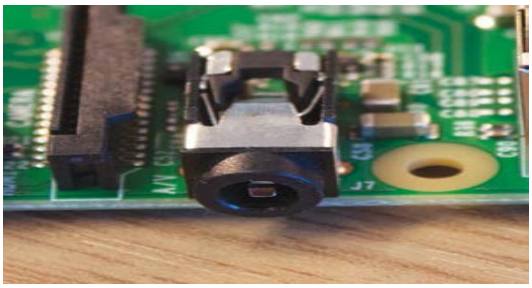


Figure 2.6 : Connecteur audio

4.4 GPIO :

C'est un ensemble de broches de connecteurs d'entrée et de sortie universelles ont diveres fonctions, elles permettent de connecter au RASPBERRY Pi avec un circuit électronique. Vous pouvez alors programmer le Pi pour contrôler le circuit et faire des choses importantes. Il y a 26 épingles dans les modèles A et B et 40 épingles dans les modèles A+, B+ et Raspberry Pi 2 modèle B.

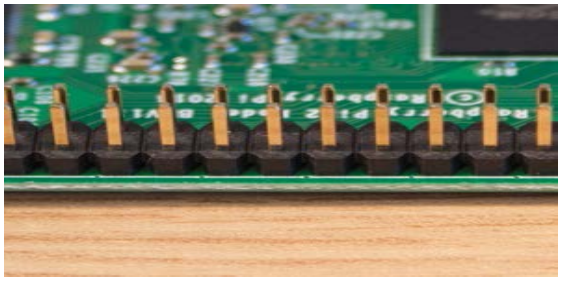


Figure 2.7 : GPIO

4.5 Fente pour carte MicroSD :

Cette tente est utilisée pour contenir une petite carte SD utilisée comme disque dur du RASPBERRY Pi dont laquelle le système d'exploitation sera installé.

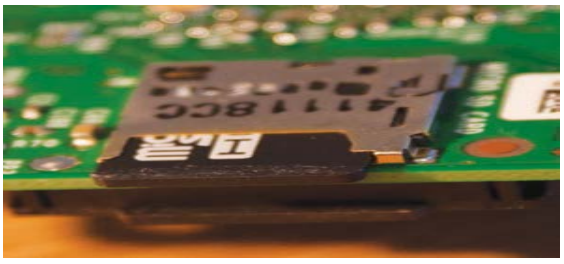


Figure 2.8 : Fente pour carte MicroSD

4.6 Port HDMI :

C'est un port HDMI similaire à celui qui se trouve à l'arrière de la plupart des téléviseurs et moniteurs d'ordinateur modernes. On utilise un câble HDMI standard pour connecter le RASPBERRY Pi un écran, pour voir (et entendre) tout ce qu'il fait et pour configurer le Pi.

Figure 2.9 : Port HDMI

4.7 Puissance :

C'est un port de charge qu'on peut trouver dans un smartphone. Ce port micro-USB signifie qu'on peut alimenter le Pi avec le bon type de chargeur de téléphone portable ou directement à partir d'un PC - cependant, il est préférable d'utiliser l'alimentation officielle RASPBERRY Pi pour s'assurer que le Pi reçoit suffisamment d'énergie.

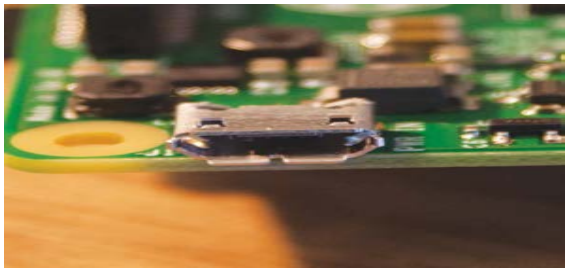


Figure 2.10 : Port d'alimentation

4.8 Le connecteur d'interface série de caméra (CSI) :

C'est un connecteur pour un module de caméra, comme illustré à la figure 2.5(a). L'appareil caméra est spécialement conçu pour RASPBERRY Pi et peut être connecté avec un câble ruban fournissant un support à la fois pour la photographie et la vidéo HD.

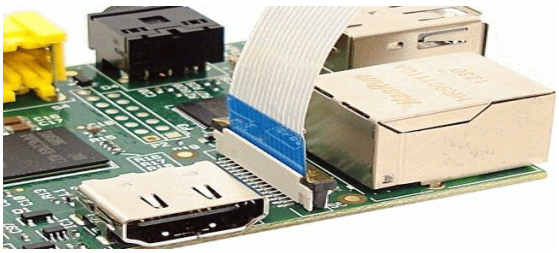


Figure 2.11 : Connecteur (CSI)

4.9 Port DSI display :

Il est utilisé pour connecter des écrans tactiles tels que les smartphones et les tablettes [15].

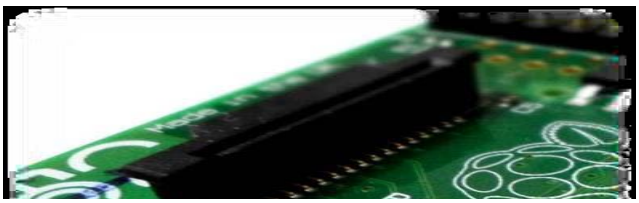


Figure 2.12 : Port DSI

5. Programmation de Raspberry Pi à l'aide de Simulink :

Il est possible d'utiliser MATLAB pour communiquer avec les cartes RASPBERRY Pi et ses périphériques en utilisant MATLAB Support Package for RASPBERRY Pi. Ainsi on peut programmer ces cartes pour exécuter nos algorithmes à l'aide de Simulink Support Package for RASPBERRY Pi Hardware. Le package de support génère du code à partir de notre modèle Simulink en un clic d'un bouton qui s'exécute ensuite sur la carte RASPBERRY Pi [16].

Et pour effectuer ses opérations on passe par plusieurs étapes :

5.1 Configuration de Package de support Simulink pour le Raspberry Pi:

Pour obtenir le support package, premièrement on ouvre MATLAB et on clique sur le bouton Add-Ons qui situe en haut à droite et on choisit « Get hardware Support Packages ».

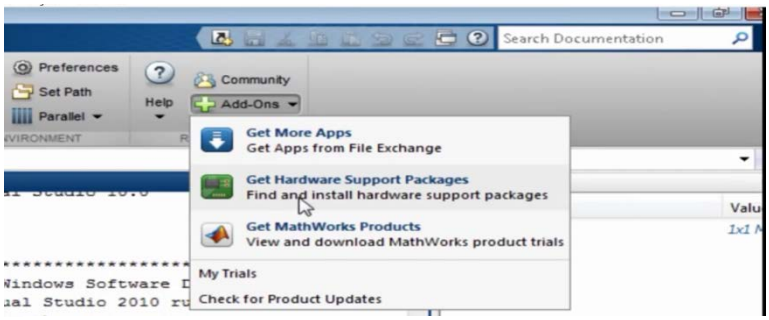


Figure 2.13 : Etape a

5.2 L'installation du Support Package :

Pour installer le support package de RASPBERRY Pi, soit on l'installe directement à partir d'internet en choisissant « Install From Internet » (Figure 2.14) et il sera télécharger et installé, soit on l'installe à partir du dossier en choisissant « Install From Folder » (Figure 2.15) si on l'a déjà téléchargé. En choisissant par exemple l'installation directement à partir de l'internet alors on a choisi « Install From Internet ». En suivant les étapes suivantes :

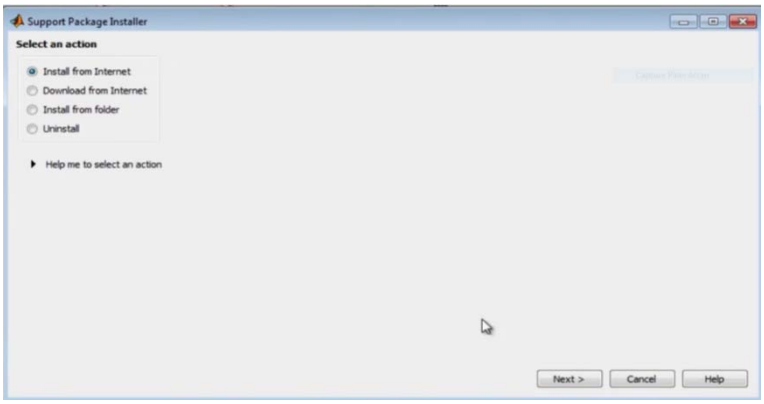


Figure 2.14 : Etape b

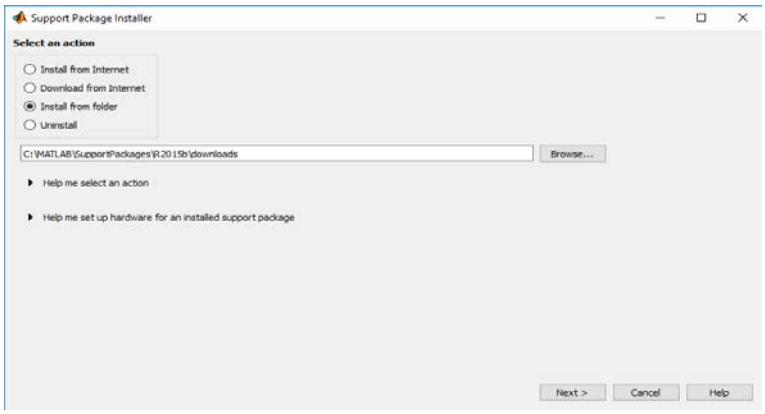


Figure 2.15 : Etape c

En cliquant sur « Next » on obtient une liste de supports packages dans cette liste. On choisit RASPBERRY Pi pour installer les deux paquets en même temps (Figure 2.16).

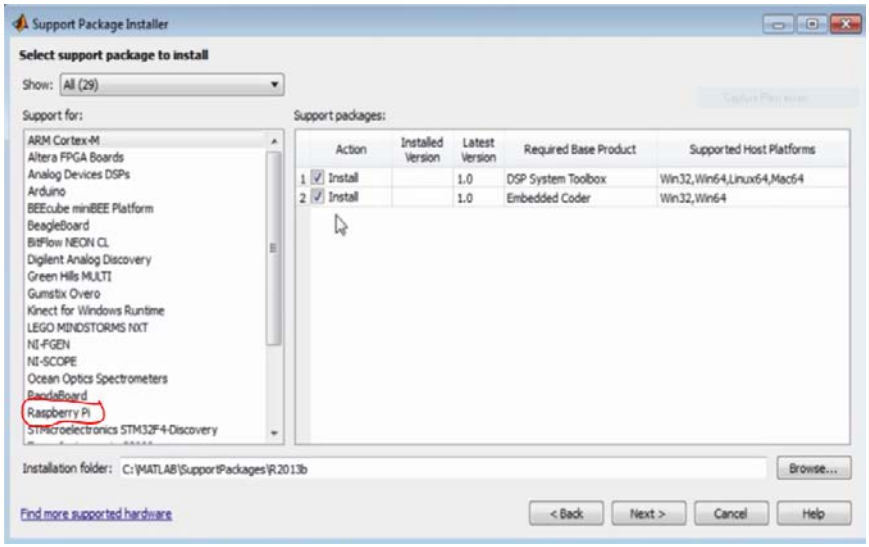


Figure 2.16 : Etape d

En cliquant sur « Next », une fenêtre s'affiche pour entrer l'adresse E-mail et le mot de passe du compte Matlab (voir la figure 2.17), si l'utilisateur n'a pas le compte, il a la possibilité de la créer pendant le processus d'installation, après avoir cliqué sur Login l'installation commence.

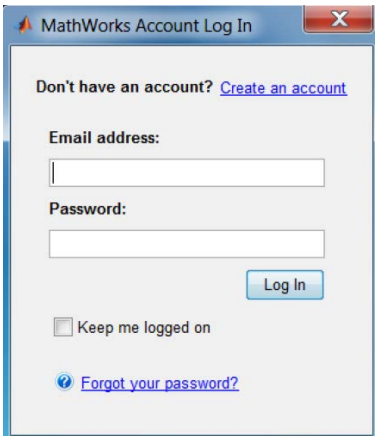


Figure 2.17 : Etape e

Une fois l'installation de package terminer le programme commence à installer les pilotes ou les mises à jour « firmware ».

Le processus de mise à jour du firmware pour le matériel RASPBERRY Pi consiste à copier une image de firmware rasbian wheezy linux dans une carte mémoire et à démarrer le RASPBERRY Pi avec cette carte mémoire. L'étape obligatoire dans ce processus est de choisir le model de RASPBERRY Pi (Figure 2.18).

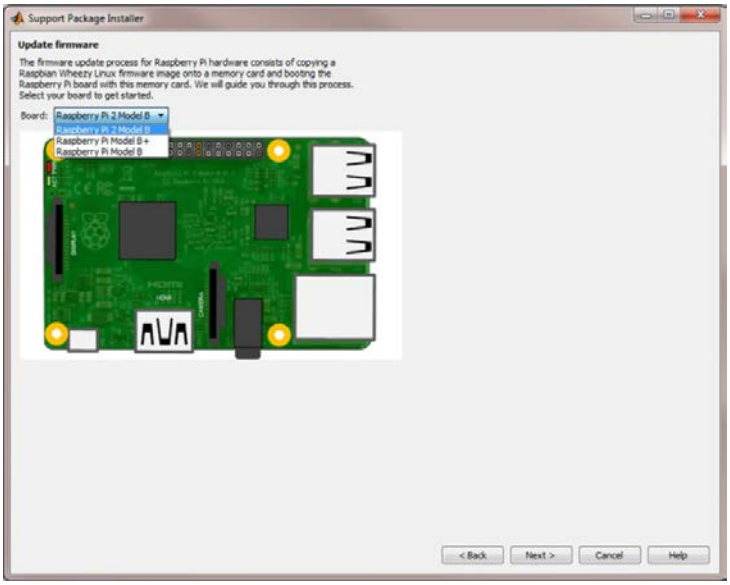


Figure 2.18 : Etape f

En cliquant sur « Next » une fenêtre s'affiche où on doit sélectionner « **Direct connection to host computer »** (Figure 2.19)**.**

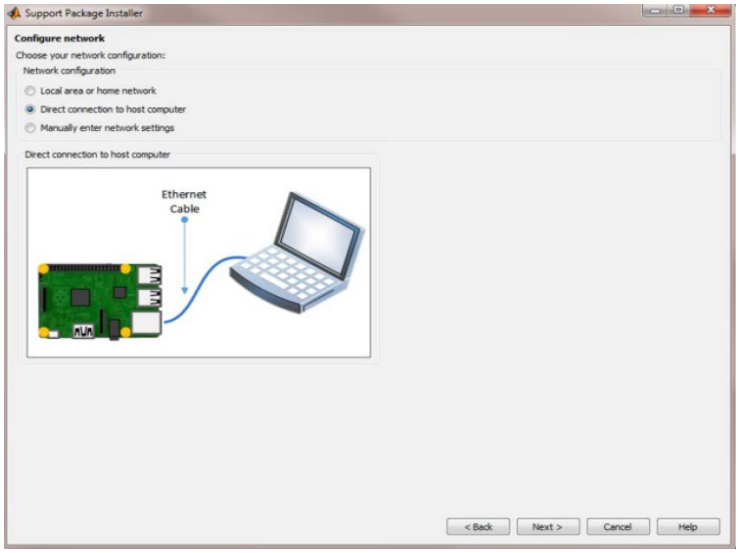


Figure 2.19 : Etape g

Une autre fenêtre s'affiche après avoir cliqué sur « Next » où on doit Sélectionner un lecteur de cartes MicroSD détectées par MATLAB et qui apparaîtront sous forme de liste (Figure 2.20). Si la carte mémoire MicroSD n'est pas détectée par MATLAB, mais est détectée par le système d'exploitation, il faut fermer MATLAB et le redémarrer en tant qu'administrateur. Pour continuer le processus, la commande « targetupdater » peut être utilisée dans MATLAB.

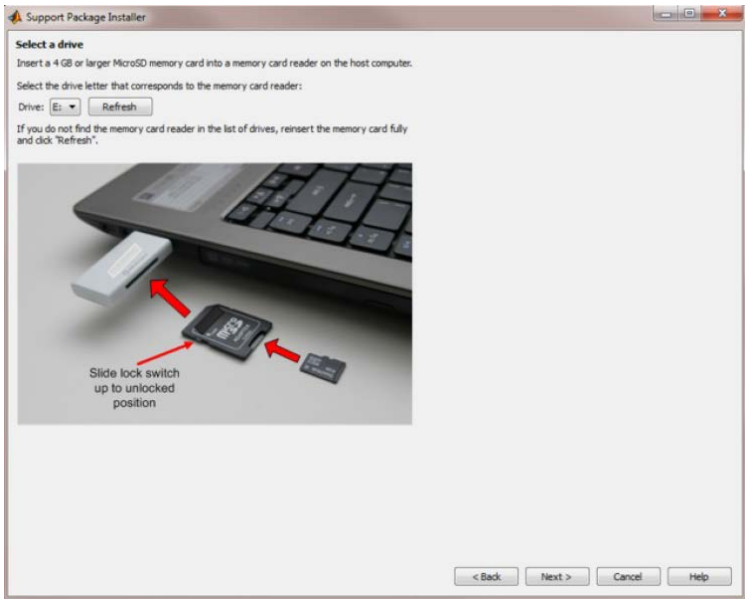


Figure 2.20 : Etape h

En cliquant sur « Next », une fenêtre s'affiche qui nous permet de sélectionner l'option d'écriture pour effacer les éléments existants sur la carte mémoire et faire flasher le dernier firmware requis par le Support Package (Figure 2.21).

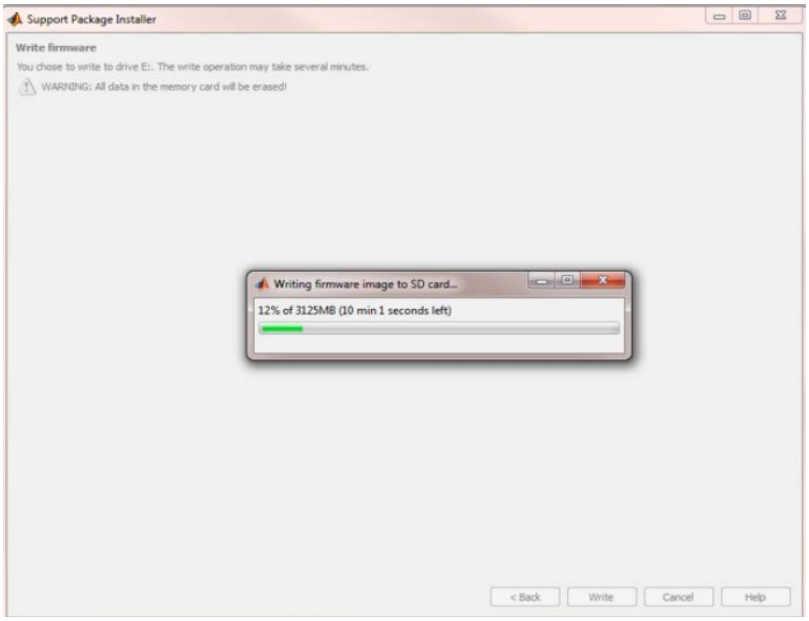


Figure 2.21 : Etape i

En fin, en cliquant sur « Next » le programme termine l'installation complet de support package pour Raspberry pi. Donc dans ce cas on peut implémenter un schéma Simulink dans la RASPBERRY [16].

6. Tester la RASPBERRY PI + Simulink :

6.1 Vérification de l'installation du Support Package :

Le support Package de Simulink pour le matériel Raspberry Pi doit se trouver dans la bibliothèque Simulink.

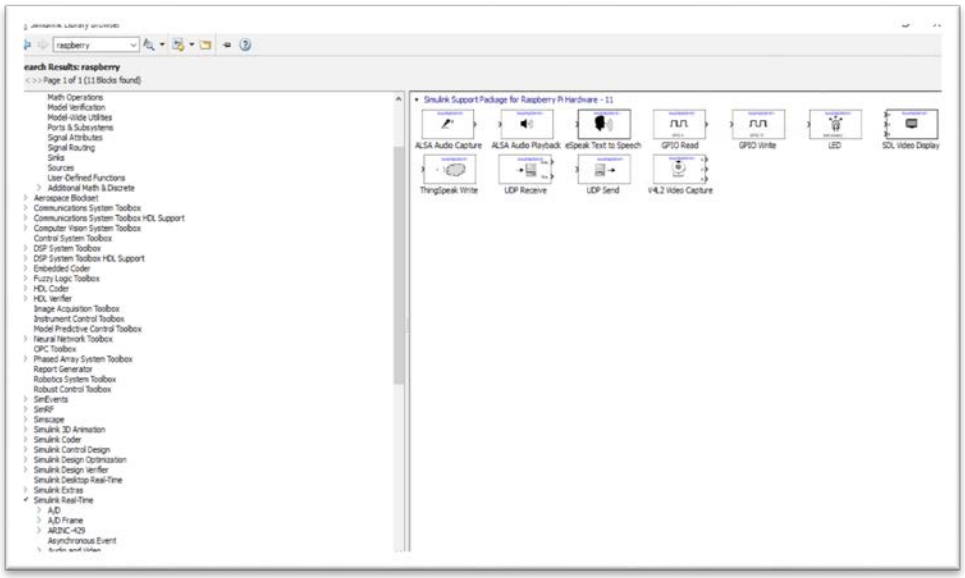


Figure 2.22 : Vérification de blocs Raspberry en Simulink

On peut aussi taper directement en ligne de commande MATLAB « raspberrypilib » pour vérifier la bibliothèque Simulink [17].

6.2 Test de la connectivité (PC à Raspberry Pi) :

Si on ne sait pas le HOSTNAME de Raspberry on tape sur la ligne commande "raspberrypi", on obtient les informations nécessaires de Raspberry pi. La figure 2.23 suivant illustre cette étape :

Figure 2.23 : Vérification de paramétres de Raspberry

Ensuite on tape en ligne de commande MATLAB "! ping + l'adresse IP" de raspberry, par exemple le cas où l'adresse IP de raspberry pi est 169.254.0.2 et voilà le résultat de test :

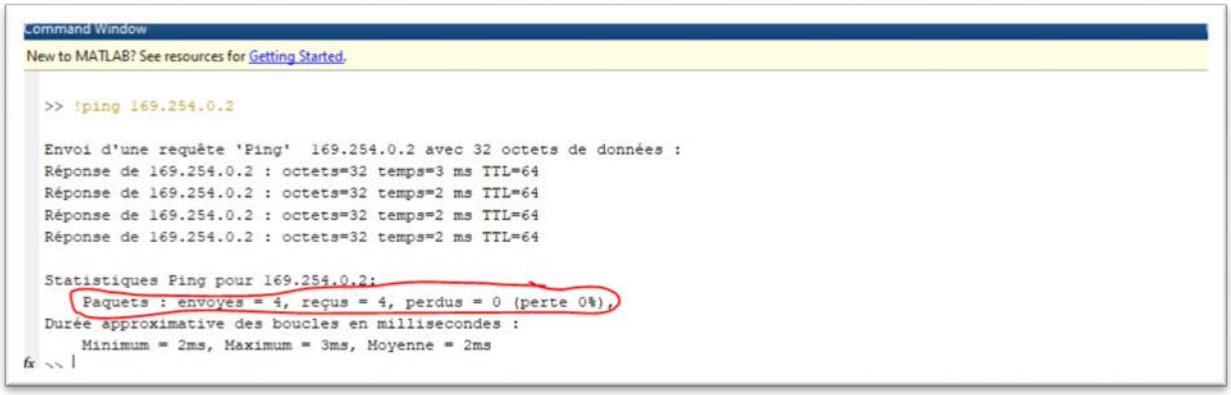


Figure 2.24 : Test la connectivité

Si on obtient le résultat encadré en rouge *« Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%) »*, on assure que notre Raspberry est bien connecté avec notre PC.

7. Conclusion :

Dans ce chapitre nous avons présenté la carte Raspberry Pi et ses différentes versions existantes. Nous avons présenté aussi toutes les étapes nécessaires à l'installation du support Package de Simulink pour le Raspberry Pi.

CHAPITRE III

Conception et réalisation de l'émetteur OFDM

1. Introduction:

La modulation OFDM est un schéma de modulation numérique qui divise la transmission entre N sous-porteuses différentes à des fréquences adjacentes, généralement appelées sous-porteuses, qui sont transmises simultanément.

Dans ce chapitre nous allons présenter l'émetteur OFDM que nous avons conçu. Une description de chaque sous bloc est donnée. Nous terminons ce chapitre par les résultats de simulation sous Simulink et l'implémentation de cet émetteur sur la carte Raspberry Pi.

2. Conception de l'émetteur OFDM sous Simulink:

Le schéma bloc de l'émetteur OFDM est représenté sur la figure 3.1. Il est constitué de sept blocs : modulation en bande de base, OFDM, Upsampling, Base de Bande vers la Fréquence Intermédiaire (BBtoIF), Contrôle Automatique de Gain (AGC), Control LED, Raspberry Pi output.

La figure 3.2 représente le schéma de l'émetteur OFDM sous Simulink

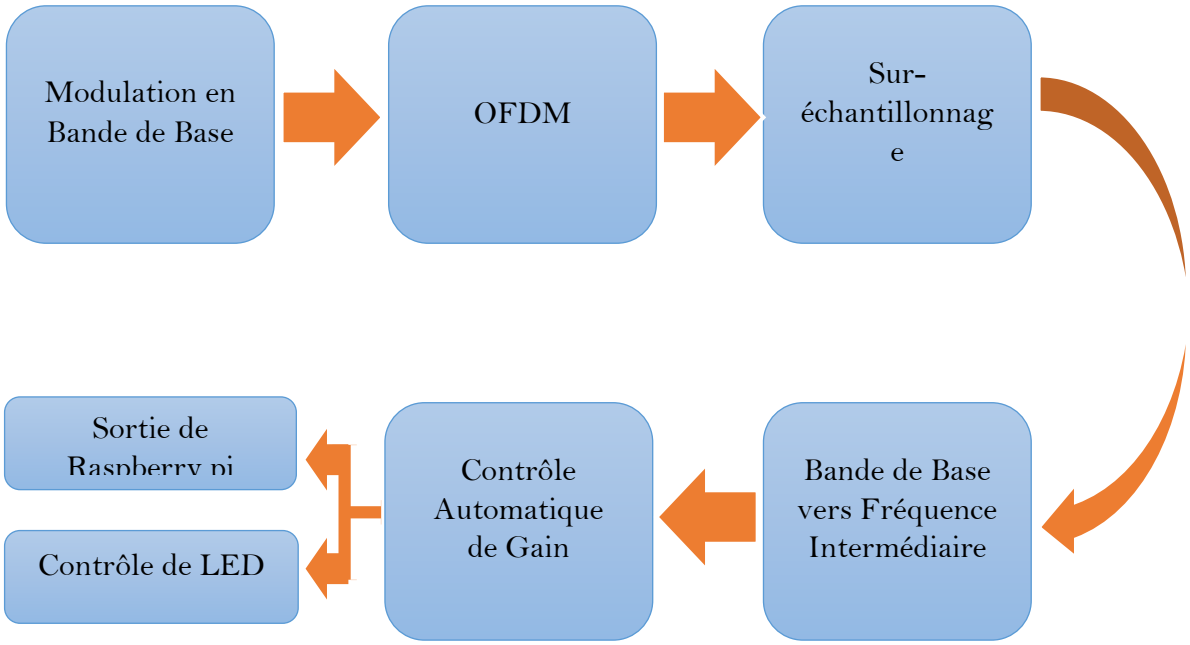


Figure 3.1 : schéma bloc de l'émetteur OFDM

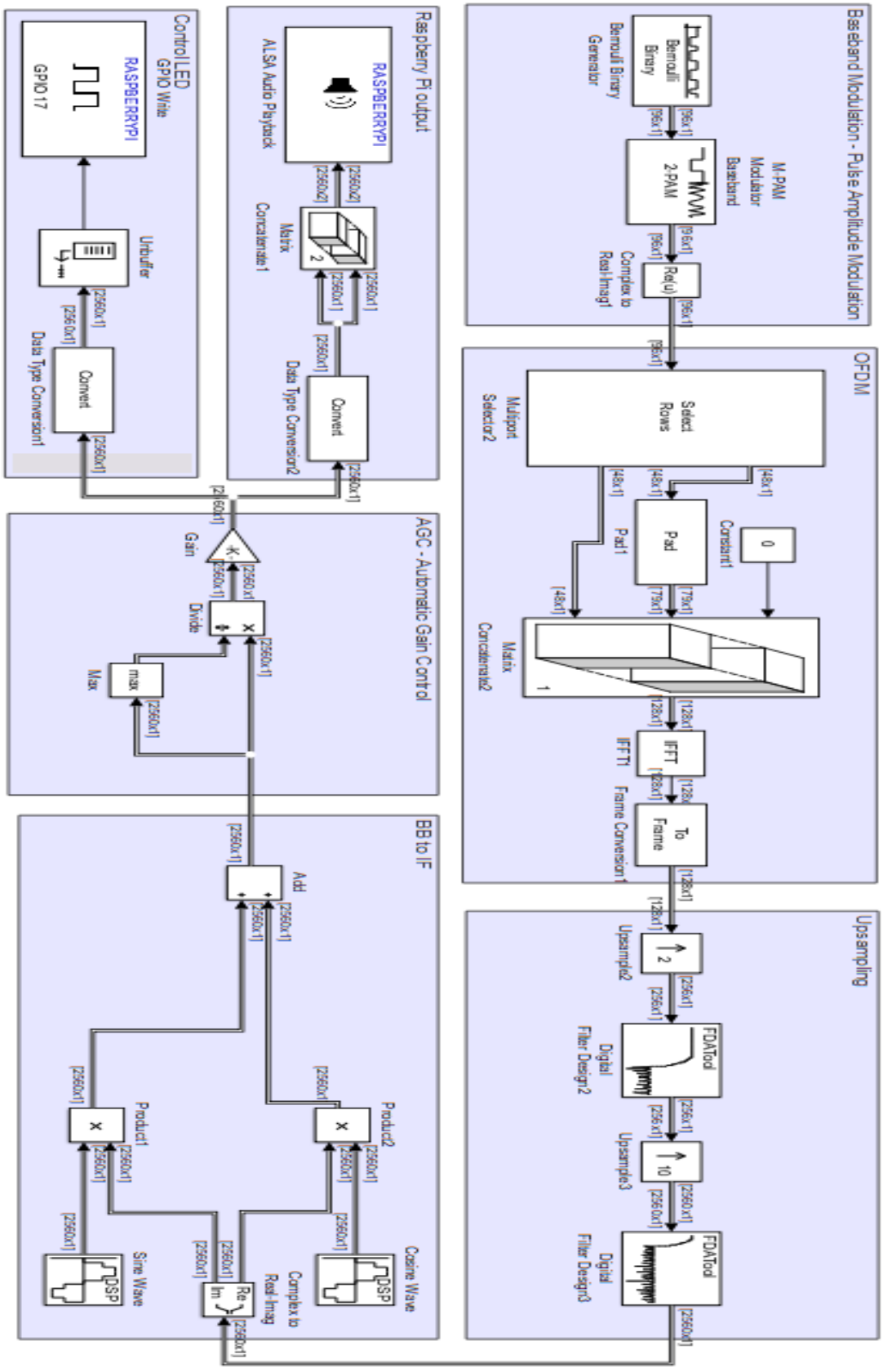


Figure 3.2 : Schéma de l'émetteur OFDM sous simulink

2.1 Modulation en bande de base :

Le macro-bloc Base Bande Modulation est une modulation en bande de base qui génère des symboles réels $a_i \in \{-1, 1\}$, à partir des bits produits par le bloc Bernoulli Binary Generator. Ces symboles seront utilisés pour moduler les sous-porteuses du signal OFDM. Comme la modulation est binaire, il y a deux symboles donc la modulation adoptée pour les sous-porteuses est 2-PAM (Figure 3.3).

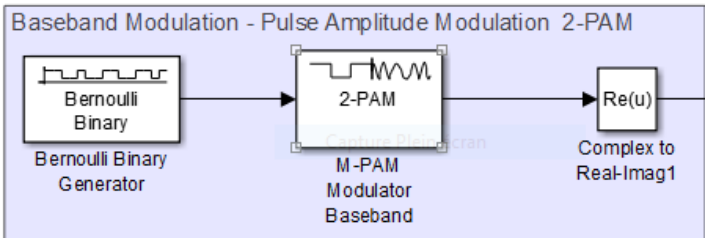


Figure 3.3 : Le bloc Base band Modulation

2.1.1 Générateur binaire de Bernoulli :

Le bloc générateur Bernoulli Binary Generator est une source d'information binaire, comme le montre la fenêtre de configuration illustrée dans la figure 3.4. Les bits sont générés à un débit d'un bit toutes les secondes et sont groupés, avant de les passer au bloc suivant, dans des trames de longueur égale à l'échantillon par trame.

Dans ce cas particulier, la valeur assignée au paramètre temps d'échantillonnage doit remplir la condition [18]:

$$Sample\ Time = \frac{Upsampling}{b_{symbole} * Audio\ Sampling\ Frequency} * \frac{N}{N_u} \qquad (3.1)$$

$\frac{1}{Sample\ Time}$ =>représente la fréquence d'échantillonnage d'entrée.

Tell que :

- Upsampling : représente le facteur de sur-échantillonnage introduit par le macro-bloc d'échantillonnage Upsampling.
- Audio Sampling Fréquency: c'est la fréquence d'échantillonnage audio indiquée dans le bloc lecture audio de l'ALSA (ALSA Audio Playback).

- $b_{symbole} = \log_2 M$ est le nombre de bits utilisés pour générer des symboles de modulation pour chaque sous-porteuse. M indiquant le nombre de symboles de la modulation adoptée (Dans le 2-PAM, M=2).
- N est le nombre total de sous-porteuses utilisées (N correspondant à la taille IFFT).
- N_u est le nombre de sous-porteuses effectivement utilisées pour transmettre des données.

Dans notre cas, nous avons fixé la fréquence d'échantillonnage audio à 48000, le facteur de sur-échantillonnage à 20, le nombre de sous porteuses N à 128 et N_u à 96. Nous avons utilisé une modulation binaire M=2 donc $b_{symbole} = \log_2 M = 1$, il en résulte :

$$Sample\ time = \frac{20}{48000} * \frac{128}{96}$$

Le paramètre échantillon par trame (Samples per Frame) doit se voir attribuer une valeur correspondant au nombre N_u de sous-porteuses effectivement utilisées pour la transmission des données. Donc on a pris une valeur de 96.

La sortie du macro-bloc de modulation en bande de base est donc une séquence de trames, l'une après l'autre, contenant les 96 symboles qui modulent les 96 sous-porteuses utiles, comme indiqué ci-dessous :

$$\underbrace{\{\text{Symbole\#1, Symbole\#2, } \dots \text{ Symbole\#96}\}}_{\text{Trame}} \qquad (3.2)$$

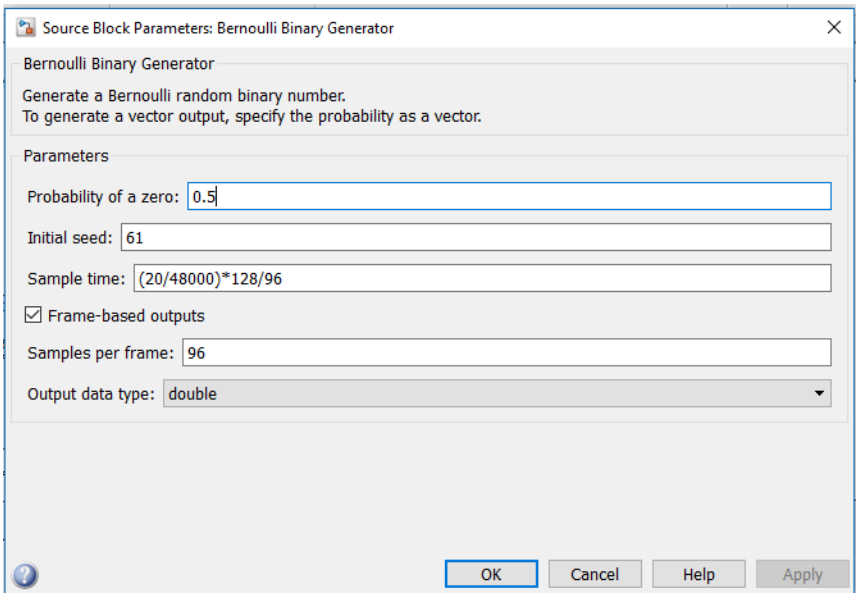


Figure 3.4 : Fenêtre de configuration du bloc Bernoulli Binary Generator

2.1.2 Modulateur M-PAM Bande de base:

Le bloc modulateur en bande de base M-PAM convertit l'entrée bits ∈ {0, 1} en 2 symboles PAM ∈ {-1, 1}. La configuration de ce bloc est illustrée sur la figure 3.5.

Malgré le fait que les symboles 2-PAM sont des quantités purement réelles, le bloc M-PAM Modulator Baseband génère des symboles complexes avec une partie imaginaire nulle (es. ⁀…, 1 + i0, -1 + i0, …⁀).

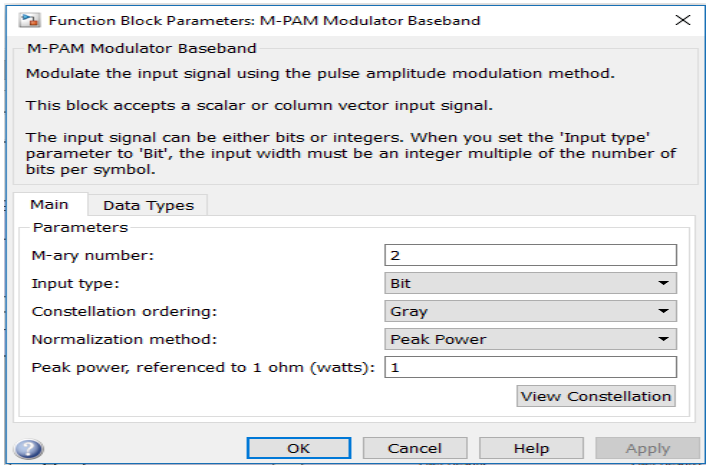


Figure 3.5 : Fenêtre de configuration du bloc M-PAM

Un bloc Complex to Real-Imag est donc nécessaire pour supprimer les parties imaginaires.

2.1.3 Le bloc Complex to Real-Imag:

Le bloc M-PAM Modulator Baseband est suivi du bloc Complex to Real-Imag, dans le but de convertir les symboles complexes dans le format réel, en supprimant les parties imaginaires (Figure 3.6).

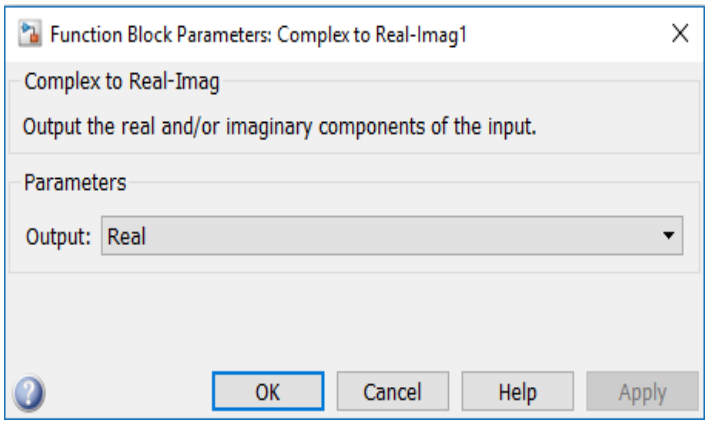


Figure 3.6 : Fenêtre de configuration du bloc Complex to Real-Imag

2.2 Le bloc OFDM :

A partir des symboles de modulation a_i à son entrée, le macro-bloc OFDM (Figure 3.7) génère le signal en bande de base correspondant à une modulation OFDM avec N = 128 sous-porteuses, dont N_u = 96 sont effectivement utilisées pour transmettre des symboles de modulation (sous-porteuses utiles) alors que les autres N_z = 32 ont une amplitude nulle (31 sous-porteuses virtuelles + 1 sous-porteuse DC qui correspond à la fréquence zéro)

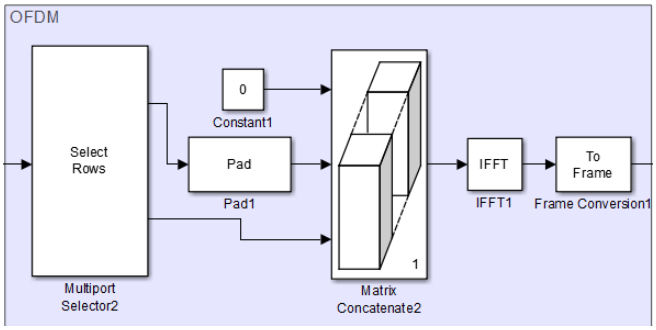


Figure 3.7 : Bloc OFDM

Dans ce cas, les N = 128 sous-porteuses sont utilisées comme suit :

- 16 sous-porteuses virtuelles (de 1 à 16) d'amplitude nulle.
- 48 sous-porteuses de données (de 17 à 64) avec modulation 2-PAM.
- 1 sous-porteuse DC (numéro 65) avec amplitude nulle (correspondant à la fréquence zéro).
- 48 sous-porteuses de données (de 66 à 113) avec modulation 2-PAM.
- 15 sous-porteuses virtuelles (de 114 à 128) d'amplitude nulle.

La figure 3.8 montre le spectre en bande de base du signal généré par le macro-bloc OFDM, ainsi qu'un numéro de référence indiquant la position de chaque sous-porteuse dans la séquence des sous-porteuses.

Désignant par F_s la fréquence d'échantillonnage du signal généré par le macro-bloc OFDM et par Δf l'intervalle entre deux sous-porteuses consécutives, la composante spectrale contenue dans la bande de Nyquist $\left[\frac{-F_s}{2}, \frac{F_s}{2}\right]$ est représentée par une ligne continue, tandis que les répétitions périodiques avec période $F_s = N\,\Delta f$, causées par le caractère temporel discret du signal, sont représentées par une ligne en pointillée. Le spectre a été enrichi par l'indication du numéro de symbole associé à chaque sous-porteuse. Une amplitude nulle est attribuée aux sous-porteuses restantes. (DC et virtuelle) (Figure 3.8).

La génération de ce signal est la tâche assignée au bloc IFFT, qui reçoit les N symboles associés à toutes les N sous-porteuses (96 utiles, 31 virtuelles et 1 DC) et produit le signal OFDM en bande de base.

Le bloc IFFT, cependant, a besoin de recevoir les symboles associés aux sous-porteuses avec un ordre particulier : le premier symbole doit être celui associé à la sous-porteuse avec la fréquence 0, le second symbole celui associé à la sous-porteuse avec la fréquence Δf et ainsi de suite, jusqu'au dernier symbole, qui doit être celui associé à la sous-porteuse avec la fréquence (N-1) Δf. L'ordre des N = 128 symboles à l'entrée du bloc IFFT est illustré à la figure 3.8 :

$$[0; \text{symbol\#49}, \ldots, \text{symbol\#96}, \underbrace{0, \ldots, 0}_{31\ \text{Zéros}}, \text{symbol\#1}, \ldots, \text{symbol\#48}] \quad (3.3)$$

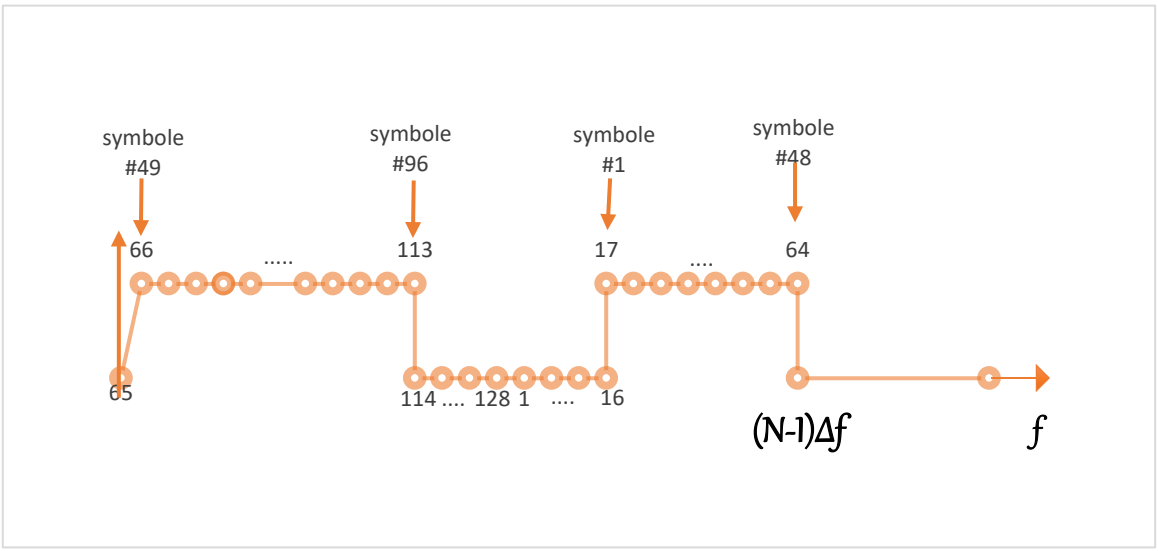


Figure 3.8 : Spectre de signal avec le numéro associé à chaque sous-porteuse

La première étape pour passer de la trame de 96 symboles dans l'expression (3.2) généré par le macro-bloc de modulation en bande de base, à une trame de 128 symboles (composé de 96 symboles + 32 zéros) comme montré dans l'expression (3.3), est effectuée par le bloc Multiport Selector2. Ce bloc divise le cadre de 96 symboles dans l'expression (3.2) à son entrée en deux cadres de 48 symboles chacun. En particulier, le cadre qu'il génère au premier port de sortie contient les symboles occupant les positions 49 à 96 dans le cadre original, tandis que le cadre au deuxième port de sortie contient les symboles occupant les positions 1 à 48 dans le cadre original. La figure 3.9 montre les paramètres de configuration de ce bloc.

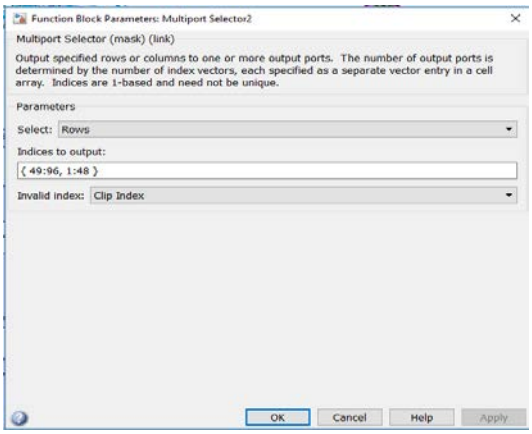


Figure 3.9 : Fenêtre de configuration du Multiport Selector

L'étape suivante consiste à construire l'expression (3.3) en insérant dans la bonne position les 32 zéros correspondant à la sous-porteuse DC et aux 31 sous-porteuses virtuelles. Le bloc Pad1 a pour tâche de créer un vecteur de 79 éléments, dont les 48 premiers correspondent aux symboles de 49 à 96 (c'est-à-dire la première sortie du **Multiport Selector2**), et les 31 valeurs de zéros qui représentent les 31 sous-porteuses virtuelles. La configuration pour ce bloc est illustrée sur la figure 3.10.

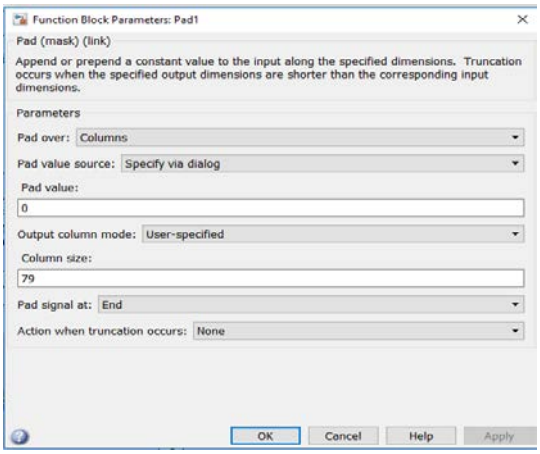


Figure 3.10 : configuration du bloc PAD

Pour arriver éventuellement à la trame donnée par l'expression (3.3), le bloc Matrix Concatenate est utilisé, il regroupe en concaténant dans une seule trame ses 3 entrées, correspondant à :

- La constante 0, correspondant à l'amplitude des sous-porteuses DC.
- Le cadre avec les symboles 49 à 96 suivis de 31 zéros, générés par le bloc Pad1.
- Le cadre avec les symboles 1 à 48, fournis par la deuxième sortie du bloc Multiport Selector2.

Les réglages du bloc Matrix Concatenate sont illustrés à la figure 3.11.

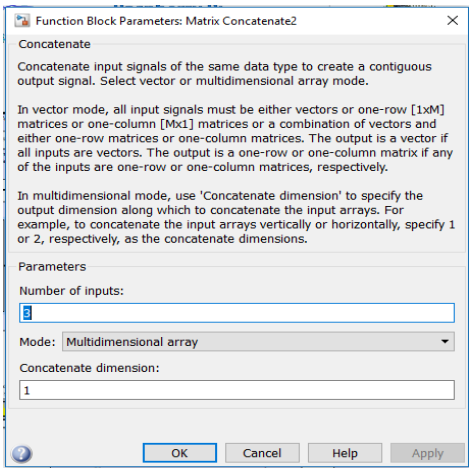


Figure 3.11 : Fenêtre de configuration du bloc Matrix Concatenate

A partir de la trame (3.3), il est enfin possible de générer le signal OFDM en bande de base en utilisant simplement le bloc IFFT (transformée de Fourier discrète inverse), qui associe les symboles d'entrée dans le domaine temporel aux sous-porteuses orthogonales correspondantes dans le domaine fréquentiel. La configuration de ce bloc est illustrée à la figure 3.12.

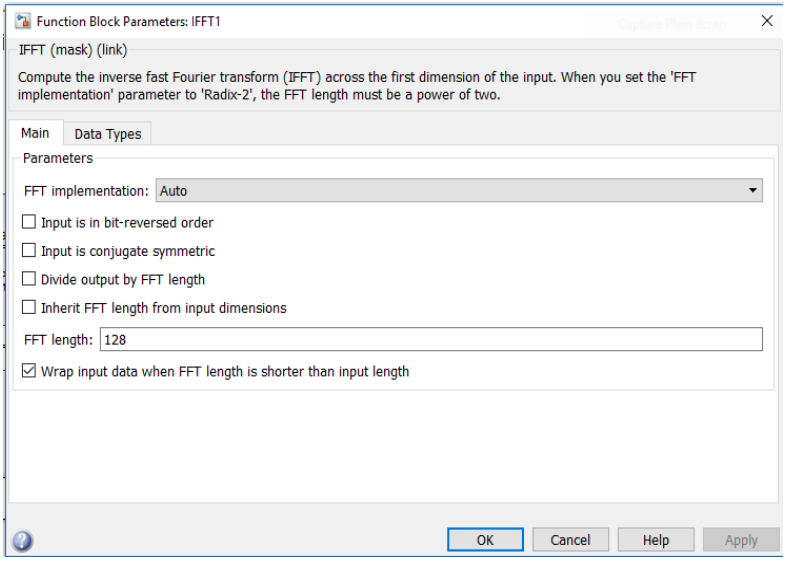


Figure 3.12 : Fenêtre de configuration du bloc IFFT

Afin de maintenir l'élaboration en mode frame based, un bloc de conversion de trame est inséré à la suite du bloc IFFT. La fenêtre de configuration correspondante est illustrée à la figure 3.13.

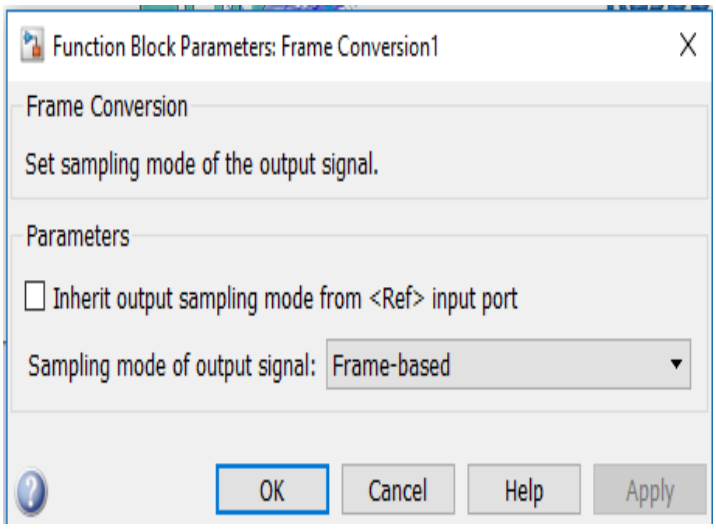


Figure 3.13 : Fenêtre de configuration du bloc frame conversion

2.3 Sur-échantillonnage (Upsampling) :

Le signal en bande de base généré par le macro-bloc OFDM a une fréquence d'échantillonnage de $F_s = \frac{48000}{20} = 2400 \frac{samples}{s}$.

Le taux d'échantillonnage $\frac{1}{sample\ time} = \frac{48000}{20} * \frac{96}{128} = 1800 \frac{samples}{s}$ choisis dans le bloc Générateur de Bernoulli a été, en fait, augmenté d'un facteur de $\frac{128}{96}$ par le bloc IFFT, qui produit une trame de 128 éléments pour chaque trame de 96 éléments reçus en entrée.

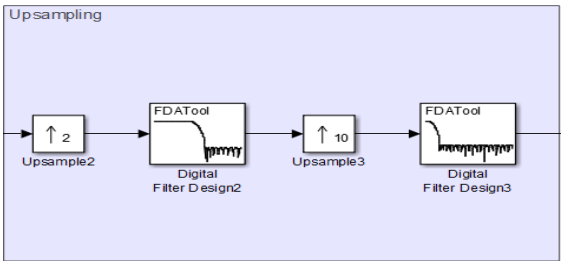


Figure 3.14 : Bloc Upsampling

Pour moduler le signal OFDM en bande de base, la fréquence d'échantillonnage doit être augmentée, il convient donc, de l'augmenter d'un facteur de 20, en l'amenant à la valeur de 48000 $\frac{samples}{s}$ requis par le CAD de Raspberry Pi.

L'opération de sur échantillonnage est réalisée par la séquence de blocs inclus dans le macro-bloc Upsampling, qui effectue un sur échantillonnage par un facteur de 2 d'abord et ensuite par un facteur de

10. La procédure en deux étapes est pratique, comparée à un sur échantillonnage en une seule étape par un facteur de 20, parce qu'elle réduit la charge de calcul globale [18].

Le premier étage du macro-bloc Upsampling (Figure 3.14) introduit un 0 entre un échantillon et l'autre du signal d'entrée. Cette opération augmente la fréquence d'échantillonnage d'un facteur 2, la faisant passer de 2400 $\frac{samples}{s}$ à 4800$\frac{samples}{s}$, sans modifier le spectre du signal. Le filtre numérique « Filter Design 2 » a donc pour tâche de supprimer une répétition périodique sur deux dans le spectre du signal, comme le montre la figure 3.15.

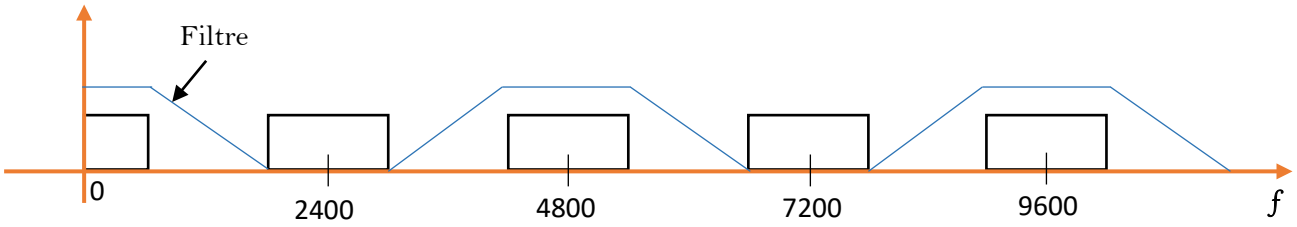
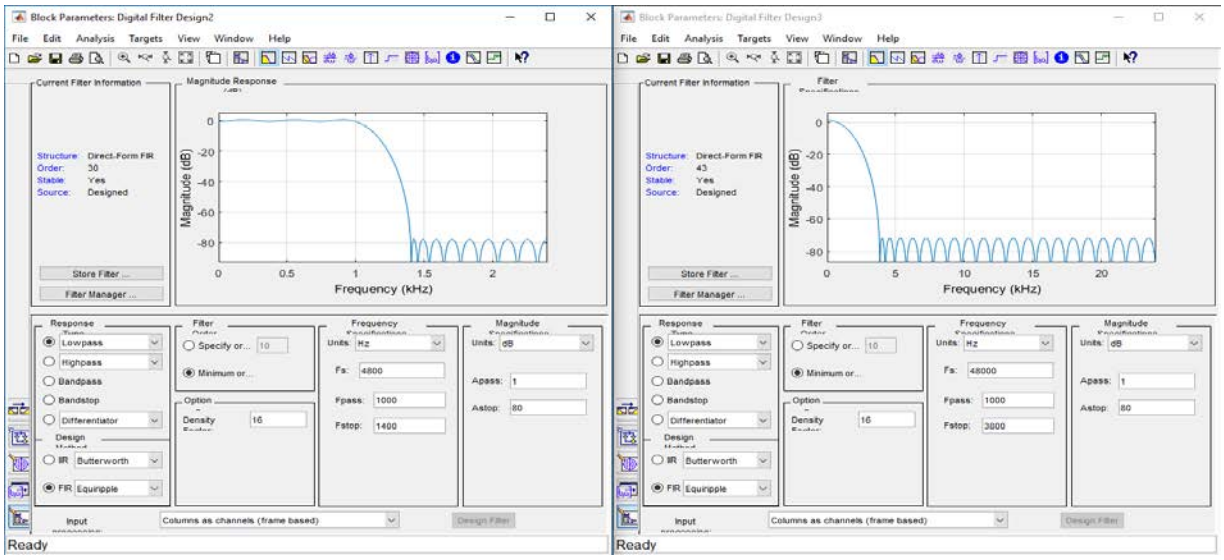


Figure 3.15 : Filtrage après sur échantillonnage par un facteur 2

Le spectre du signal à la sortie de filtre montre des répétitions périodiques avec une période égale à 4800 Hz, car il est approprié pour un signal avec une fréquence d'échantillonnage de 4800$\frac{samples}{s}$.

Un deuxième bloc de sur échantillonnage est inséré, il effectue un sur échantillonnage par un facteur 10 suivi d'un filtre, selon le même principe : l'étage de sur échantillonnage insère neuf 0s après chaque échantillon de signal d'entrée, augmentant la fréquence d'échantillonnage d'un facteur 10, et le filtre supprime 9 répétitions spectrales périodiques sur 10, de sorte que la première répétition périodique du spectre est centrée sur 48000 $\frac{samples}{s}$.

Afin de concevoir correctement les filtres, nous devons régler F_s à la nouvelle fréquence d'échantillonnage 48000 et assigner à F_{pass} 1000, la limite supérieure de la bande à préserver et assigner à F_{stop} la valeur de l'ancienne fréquence d'échantillonnage moins la bande à préserver. La figure 3.16 montre les réglages des deux filtres.



(a) Filtre avec F_s=4800Hz (b) Filtre avec F_s=48000

Figure 3.16 : réglage des filtres

2.4 Bande de base vers la fréquence intermédiaire (BBtoIF):

Le macro-bloc BBtoIF module le signal, le déplaçant de la bande de base à la fréquence intermédiaire. Cette opération est réalisée au moyen d'un modulateur en quadrature avec des porteuses générées en interne à une fréquence $F_{IF} = 15$ kHz.

Le macro-bloc BBtoIF est un modulateur classique en quadrature, avec deux chemins séparés pour les composantes réelles (en phase) et imaginaires (en quadrature) du signal. Chaque composante est déplacée par un mélangeur (représenté par le bloc Produit) piloté par un cosinus ou un support sinusoïdal, généré par les blocs Cosine Wave et Sine Wave. Les deux signaux modulés sont ensuite additionnés et injectés à l'étage suivant comme le montre la figure 3.17.

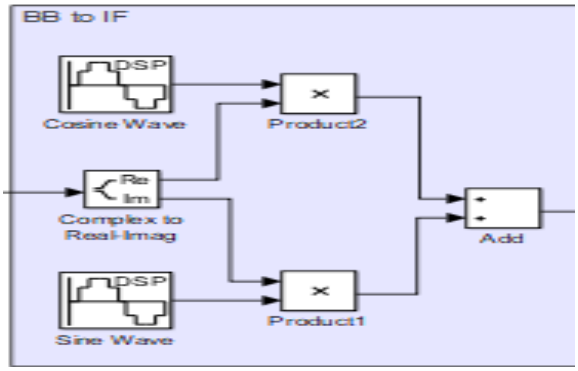
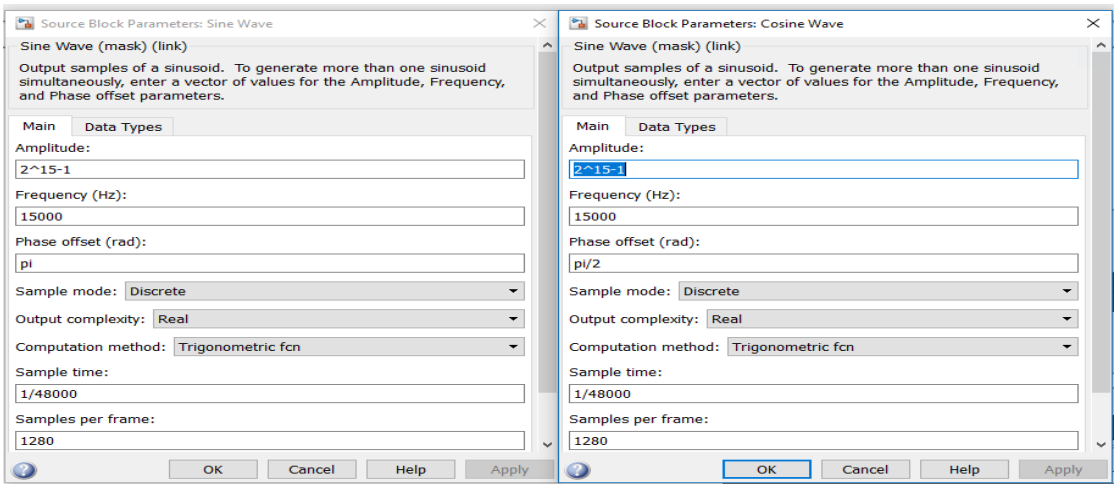


Figure 3.17 : Bloc BB to IF

Nous utilisons conventionnellement cos ($2\pi f_{IF}$ t) comme porteuse pour le signal en phase et -sin ($2\pi f_{IF}$ t) comme porteuse pour le signal en quadrature.

La figure 3.18 montre les réglages des blocs de production des porteuses. En particulier, le signal -sin ($2\pi f$ t) est généré en introduisant un décalage de phase π au signal sin ($2\pi f$ t) qui serait généré par défaut, tandis que le cos ($2\pi f$ t) est généré en introduisant un décalage de phase $\pi/2$.



(a)Réglage pour la génération –sin(2πft) (b) Réglage pour la génération cos(2πft)

Figure 3.18 : Fenêtre de configuration pour le bloc cosine wave et le bloc sine wave

2.5 Contrôle Automatique de Gain (AGC) :

Le macro-bloc de contrôle automatique du gain adapte la plage dynamique du signal à l'intensité du signal aux exigences du port de sortie du Raspberry Pi [18]. La structure du macro-bloc, illustrée à la figure 3.19, est basée sur les blocs élémentaires Max, Divide et Gain.

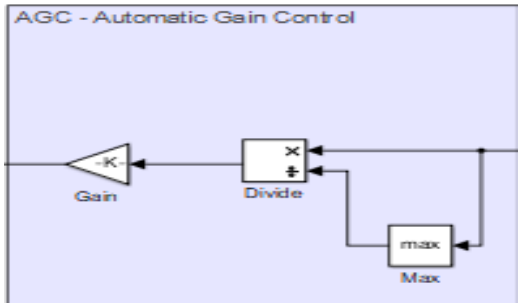
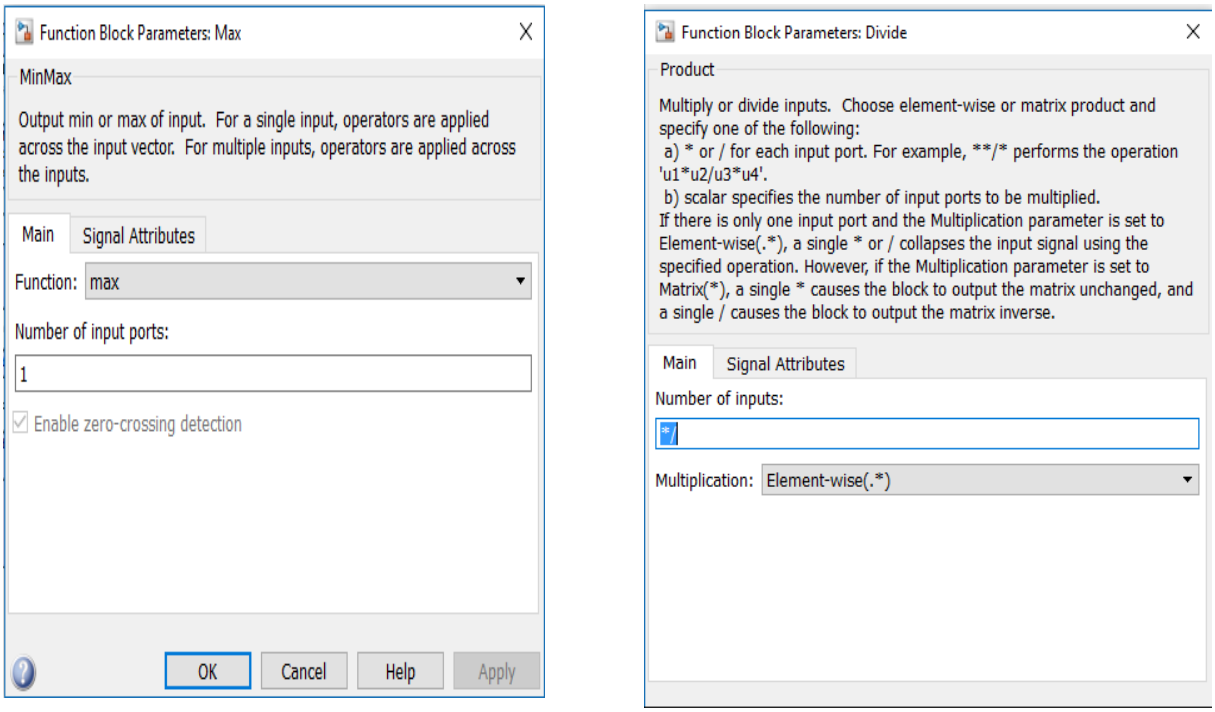


Figure 3.19 : Bloc AGC

Pour chaque trame d'entrée, le macro-bloc de contrôle automatique de gain effectue une normalisation, en divisant tous les échantillons de chaque trame par leur valeur maximale, suivit d'une

multiplication par K = 2^{14} -1. Les étapes de configuration sont montrées sur les figures 3.20(a) et (b) et 3.21.

En conséquence de cette opération, la valeur la plus élevée dans chaque trame est égale à 2^{14}-1, correspondant à la valeur la plus élevée requise par le port de sortie du Raspberry Pi égale à 2^{15} -1 [18]. Cette dernière valeur peut causer un dysfonctionnement dans certaine configuration. Ce problème est résolu en choisissant une valeur de 2^{14}-1.



(a) Max (b) Division

Figure 3.20 : Blocs Max et Divide

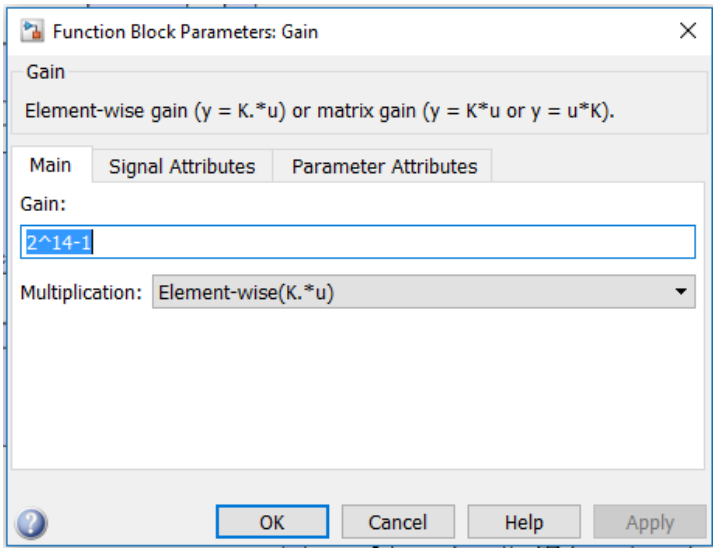


Figure 3.21 : Bloc Gain

2.6 Raspberry Pi output :

Le macro-bloc de sortie Raspberry Pi représente le port de sortie du signal. Ce macro-bloc se compose de : Data Type Conversion, Matrix Concatenate et ALSA Audio Playback (Figure 3.22).

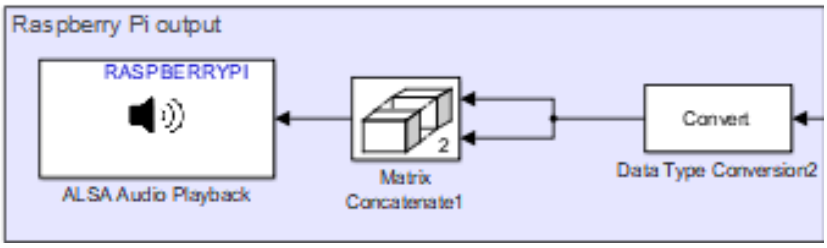


Figure 3.22 : Bloc Raspberry pi Output

2.6.1 Data Type Conversion:

Le bloc Data Type Conversion convertit un signal d'entrée de n'importe quel type de données Simulink au type de données spécifié dans sa fenêtre de configuration.

Pour notre projet, en particulier, le bloc ALSA Audio Playback nécessite des données d'entrée au format int16 (entier signé 16 bits). La conversion du type de données réelles, fournies par le bloc Sine Wave, en données int16 est effectuée par le bloc Data Type Conversion1 (dans le macro-bloc de sortie Raspberry Pi) en réglant le mode Inherit : Inherit via back propagation dans le champ du type de données de sortie, comme indiqué dans la figure 3.23. De cette façon, le bloc Data Type Conversion1 adapte automatiquement le type de données de sortie aux exigences des étapes en aval.

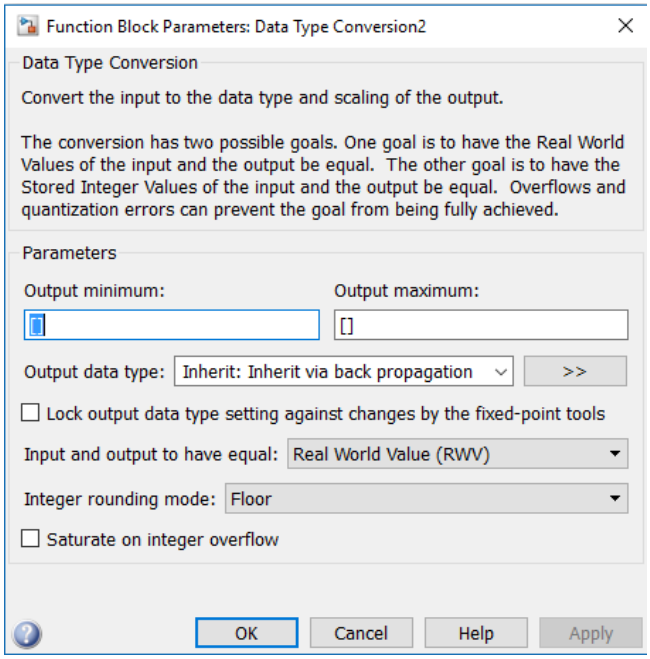


Figure 3.23 : Fenêtre de configuration du bloc Data Type conversion

2.6.2 Matrice Concaténation :

Le bloc Matrix Concatenate permet de concaténer plusieurs signaux en un seul signal de sortie. Le signal de sortie peut être un vecteur ou un tableau multidimensionnel.

Dans notre projet, ce bloc est utilisé en mode multidimensionnel, afin de produire un signal de sortie à deux canaux (c'est-à-dire un signal stéréo), à partir des deux signaux mono à son entrée.

Cette opération est nécessaire car le bloc ALSA Audio Playback, qui suit le bloc Matrix Concatenate, nécessite un signal d'entrée stéréo. C'est la raison pour laquelle le nombre de champ (Number of input) d'entrée de la fenêtre de configuration du bloc doit être réglé sur 2, de même pour le champ dimension de concaténation (Concatenate Dimension) (voir Figure 3.24).

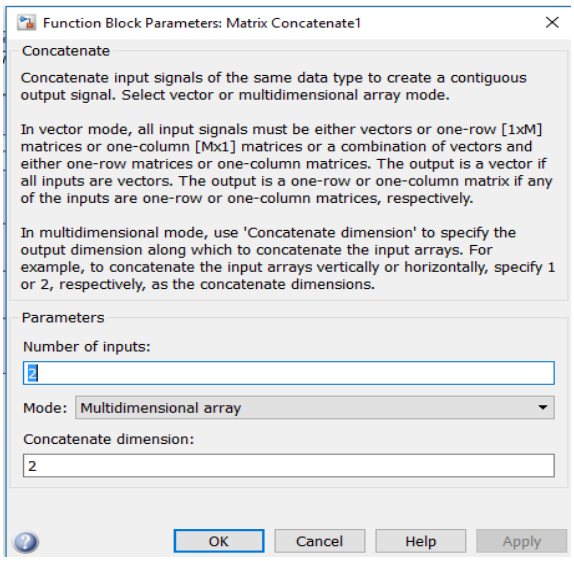


Figure 3.24 : Fenêtre de configuration du bloc Matrix Concatenate

2.6.3 ALSA Audio Playback :

Le bloc ALSA Audio Playback représente la sortie analogique de la Raspberry Pi. Sa tâche est d'effectuer la conversion numérique-analogique du signal et de l'envoyer à la carte son pour la lecture. Ce bloc repose sur le pilote audio ALSA du Raspberry Pi, qui gère tous les périphériques audio connectés au Raspberry Pi.

Le signal d'entrée doit avoir une dimension de [Nx2], où N est le nombre d'échantillons dans chaque trame et 2 est le nombre de canaux audio (dans ce cas particulier N=2048). Chaque échantillon doit être représenté dans le format int16.

Comme montré sur la fenêtre de configuration illustrée à la figure 3.25, le taux d'échantillonnage est sélectionné dans la fréquence d'échantillonnage du champ audio. Nous pouvons choisir une valeur parmi plusieurs valeurs proposées mais la valeur maximale est de 48000 Hz.

L'identificateur de la carte son doit être saisi dans le champ Device Name : cet identificateur est défini par la syntaxe 'plughw: card, device', où les deux paramètres carte et périphérique peuvent être facilement obtenus en utilisant la commande aplay -l dans un shell Linux. Le résultat est une liste à l'écran de tous les appareils connectés au Raspberry Pi et de leur carte de paramètres correspondante.

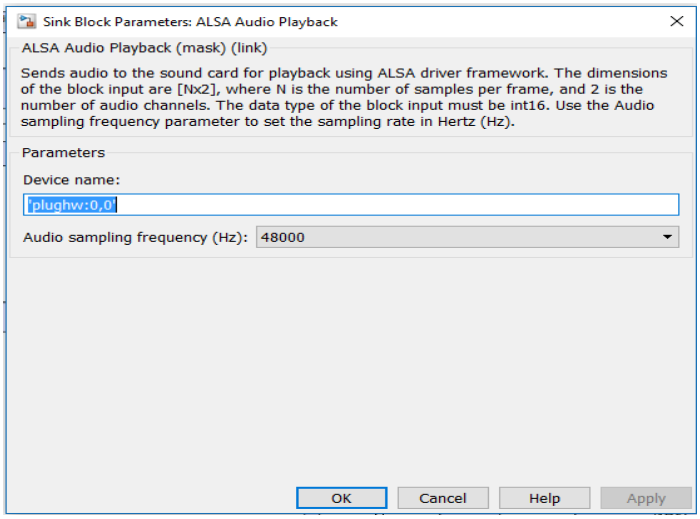


Figure 3.25 : Fenêtre de configuration du bloc ALSA Adio Playback

2.7 Control LED:

Le macro-bloc de contrôle LED représente le port de sortie du signal pour vérifier le fonctionnement. Ce macro-bloc se compose de : Data Type Conversion1, Unbuffer et GPIO Write (Voir la Figure 3.26). Le seul but du macro-bloc LED de contrôle est d'allumer et d'éteindre la LED pendant l'exécution du projet, en contrôlant visuellement le projet en cours d'exécution.

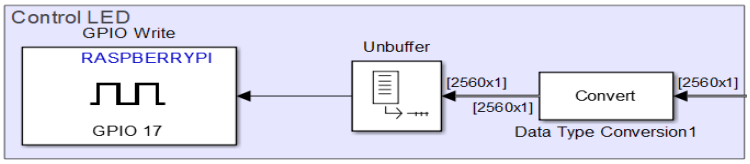


Figure 3.26 : Bloc Control LED

2.7.1 Data Type Conversion1:

Le bloc Conversion de type de données a été décrit à la section 2.6.1. Il convertit un signal d'entrée de n'importe quel type de données Simulink au type de données spécifié dans sa fenêtre de configuration. Ce bloc adopte les mêmes configurations que celles de la figure 3.23.

2.7.2 Unbuffer:

Le bloc Unbuffer convertit une trame en série échantillons avec un débit élevé. En d'autres termes, les entrées ne sont pas mises en mémoire tampon par ligne, de sorte que chaque ligne de matrice devient un échantillon temporel indépendant dans la sortie. La vitesse à laquelle le bloc reçoit les entrées est généralement inférieure à la vitesse à laquelle le bloc produit les sorties.

Cette opération est nécessaire car le bloc GPIO Write qui suit le bloc Unbuffer, nécessite un signal d'échantillon scalaire à une fréquence plus élevée. La fenêtre de configuration de ce bloc est représentée sur la figure 3.27.

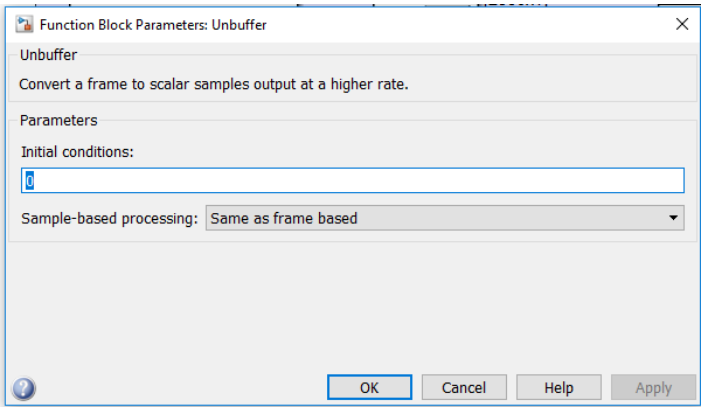


Figure 3.27 :Bloc Unbeffer

2.7.3 GPIO Write:

Le bloc GPIO Write est formé de PINs, il permet de connecter le Raspberry pi avec d'autre composants. Sur la fenêtre de configuration nous pouvons choisir la carte et le numéro de GPIO, comme illustrée à la figure 3.28.

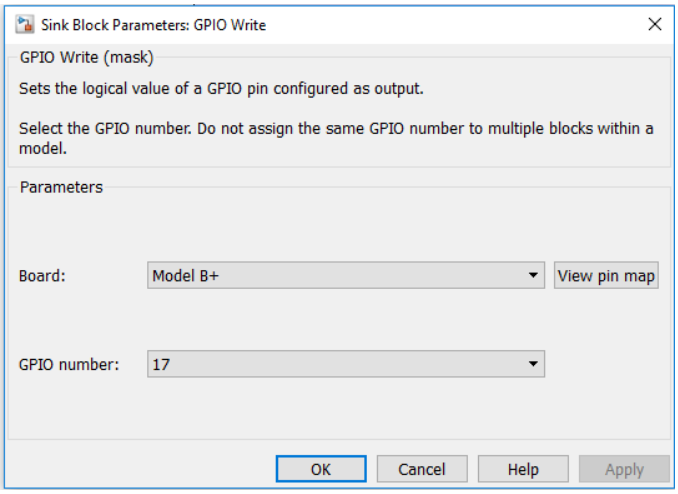


Figure 3.28 : Bloc GPIO Write

Pour trouver le numéro de PIN sur le Raspberry pi, nous devons afficher la carte des PINs « **View pin map** » comme illustrée à la figure 3.28.

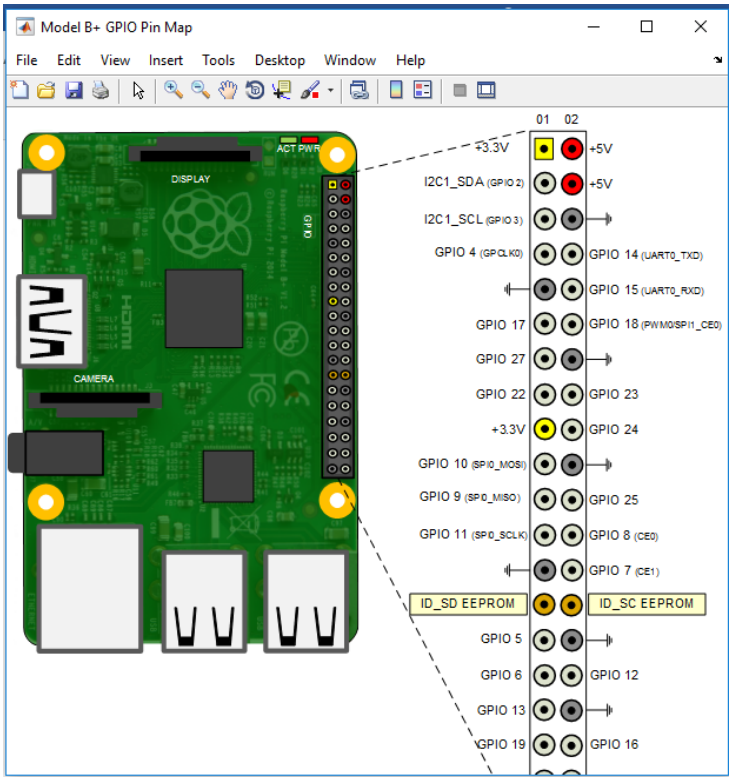


Figure 3.29 : Carte des broches

3. Simulation de l'émetteur OFDM sous Simulink:

Après avoir défini les différents blocs de notre émetteur OFDM, nous vérifions son fonctionnement sous Simulink en choisissant le mode normal (voir la figure 3.30).

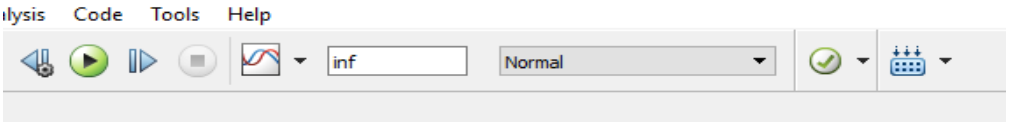


Figure 3.30 : L'exécution en mode simulation

3.1 Résultats de simulation après le bloc d'IFFT :

Pour visualiser le spectre on insère un analyseur de spectre après le bloc complet de l'OFDM. Les résultats suivants montre ce spectre pour différentes fréquences:

- **Pour une fréquence d'échantillonnage de 48000 Hz :**

A partir de cette fréquence on obtient le résultat suivant :

$$\frac{48000}{20} * \frac{96}{128} = 1800$$

1800 : représente la bande de signal OFDM théorique et c'est la fréquence d'échantillonnage choisie dans le bloc générateur de Bernoulli.

48000: La fréquence d'échantillonnage audio c'est la fréquence d'échantillonnage indiquée dans le bloc Lecture audio de l'ALSA on a plusieurs choix de cette fréquence d'échantillonnage: (44100, 32000, 22050, 16000, 8000, 4000) Hz.

20 : le facteur de sur échantillonnage.

96 : le nombre de sous-porteuses effectivement utilisées pour transmettre des données.

128 : la taille d'IFFT qui représente le nombre des sous-porteuses totale utilisées.

La figure ci-dessous représente le signal obtenu à la sortie de bloc OFDM:

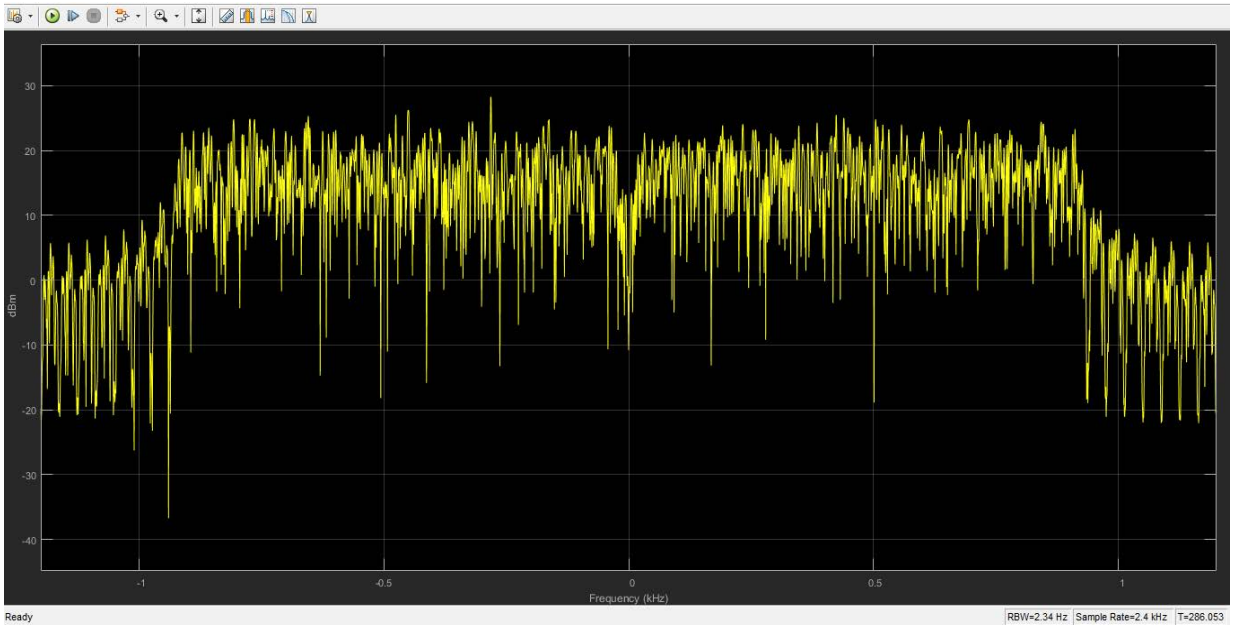


Figure 3.31 : le spectre OFDM pour une fréquence d'échantillonnage de 48000 Hz

Remarque :

- On note que le signal produit à la sortie de bloc IFFT a une bande de 1.8 KHz similaire à la valeur choisie au départ.
- Chaque pic représente une sous-porteuse.
- Le pic à faible amplitude représente la sous-porteuse centrale.

- **Pour une fréquence d'échantillonnage de 44100 Hz :**

La bande de signal OFDM est : 1653.75 Hz.

La figure ci-dessous représente le résultat de simulation.

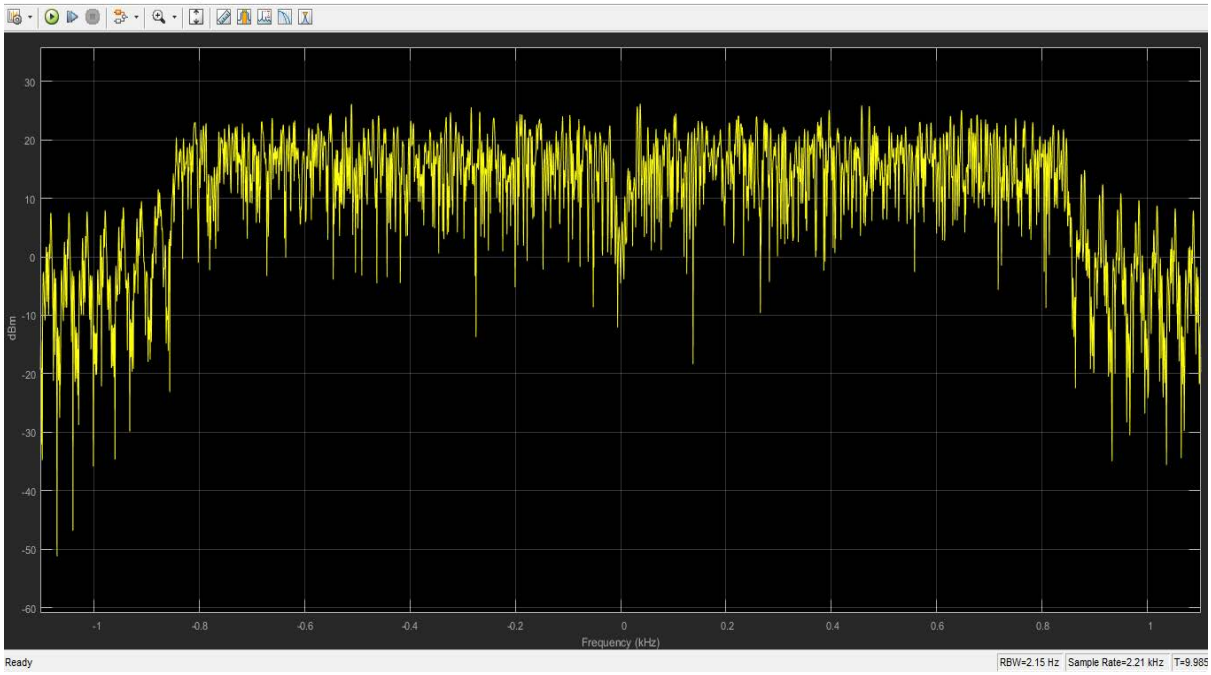


Figure 3.32 : le spectre OFDM pour une fréquence d'échantillonnage de 44100 Hz

Pour la fréquence d'échantillonnage de 32000 :

La bande de signal OFDM est égale à 1200 Hz.

La figure ci-dessous représente le résultat de simulation.

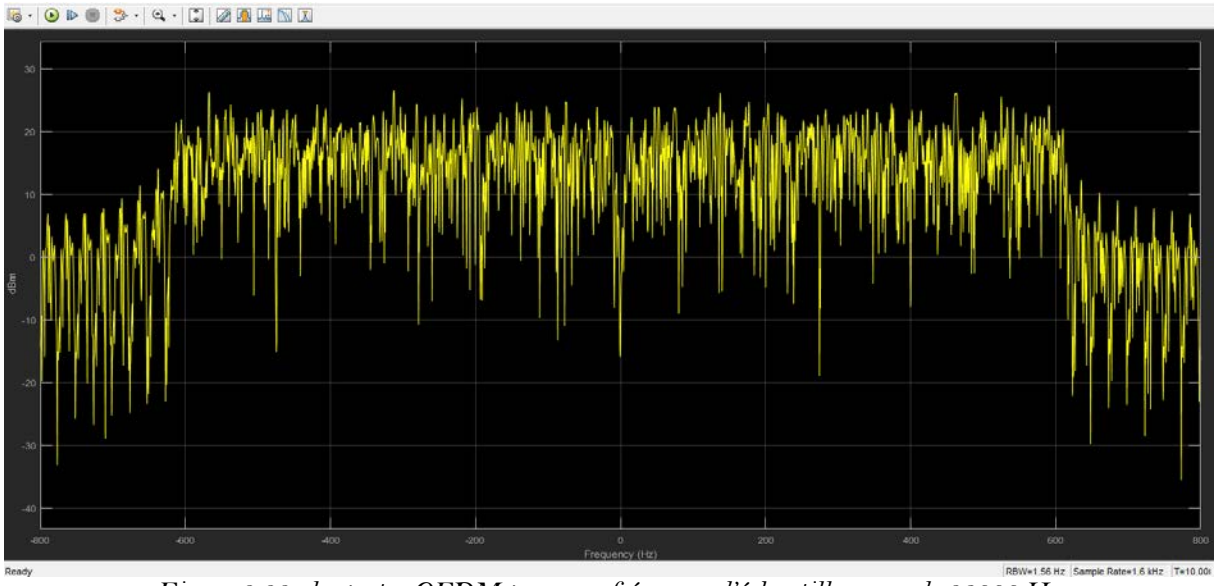


Figure 3.33 : le spectre OFDM pour une fréquence d'échantillonnage de 32000 Hz

A partir de ces figures on observe que plus on diminue la fréquence d'échantillonnage, la bande de signal OFDM diminue, et si on augmente la fréquence d'échantillonnage la bande de signal augmente.

3.2 Le spectre de signal OFDM après le bloc BBtoIF :

Dans ce cas, on va fixer la fréquence d'échantillonnage à la valeur 48 KHz et on change la fréquence intermédiaire « IF »de bloc BBtoIF.

Première cas : F_{IF}=15 KHz.

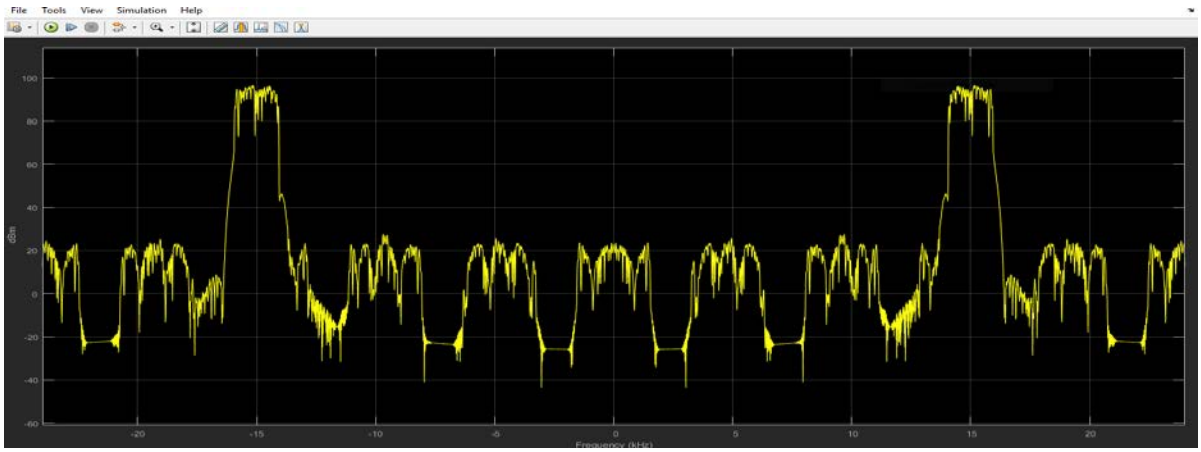


Figure 3.34 : Le spectre OFDM après déplacement de 15000 Hz

On remarque que le signal est centré à la valeur choisie.

Deuxième cas : F_{IF}=20 KHz.

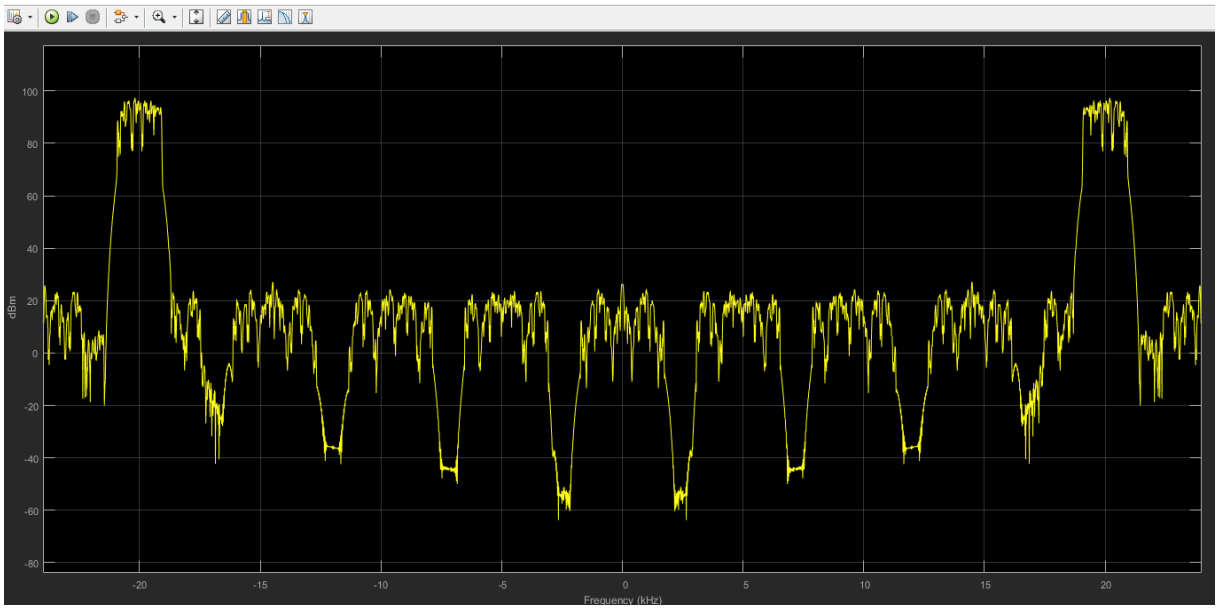


Figure 3.35 :Le spectre OFDM après déplacement de 20000 Hz

Dans ce cas aussi on voit que le signal est centré à la valeur choisie.

Troisième cas : F_{IF}=48 KHz.

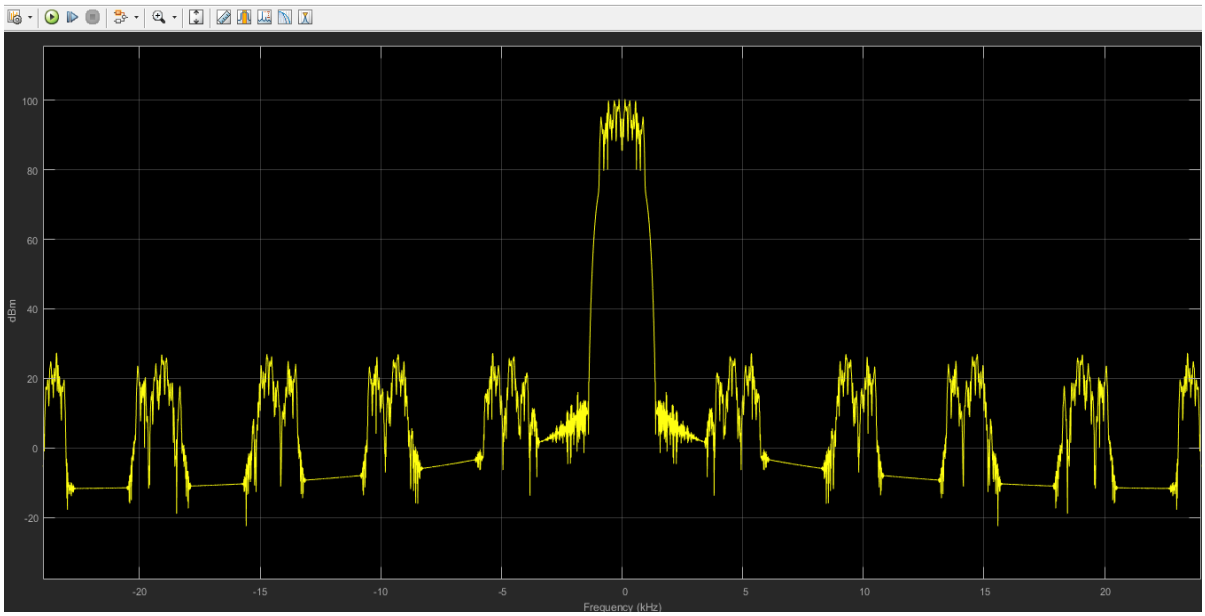


Figure 3.36 : Le spectre OFDM après déplacement de 48000 Hz

Dans ce cas le signal n'est pas centré à la valeur désirée.

3.3 Interprétation des résultats :

Dans le premier cas et le deuxième cas le signal est centré à la fréquence choisie car dans les deux cas nous avons respectés le critère de Nyquist, mais dans le troisième cas le signal n'est pas centré à la fréquence désiré car nous n'avons pas respecté le critère de Nyquist.

La valeur maximale de la fréquence centrale est comme suit : $\frac{F_{S2}}{2} - \frac{F_{S1}}{2}$ tel que :

F_{S2} : La fréquence d'échantillonnage de sortie ALSA.

F_{S1} : La fréquence d'échantillonnage d'entrée de Générateur de Bernoulli.

4. Implémentation de l'émetteur OFDM sur la carte Raspberry Pi :

Pour la réalisation de notre émetteur nous avons utilisé le matériel suivant:

- Raspberry pi B (a)
- Câble micro-USB (b)
- Carte mémoire micro SD(c)
- Adaptateur USB-LAN (d)
- Câble réseau(e)
- Carte audio externe (f)
- Câble jack 3.5mm-RCA (g)
- Adaptateur BNC-RCA (h)

Les images des différents composants sont données dans le tableau suivant:

(a)

(b)

(c)

(d)

(e)

(f)

(g)

(h)

Table (III) Images des différents composants

4.1 Exécution du projet sur Raspberry:

Une fois que le projet a été réalisé et testé sous Simulink et pour que le projet soit implémenté sur la carte Raspberry, il est essentiel de fournir à Simulink les informations nécessaires pour identifier le Raspberry Pi [19].

En Simulink, nous sélectionnons "tools/run on target hardware/options", comme illustré à la figure 3.37, afin de sélectionner le matériel cible. Dans ce cas, le choix est évidemment Raspberry Pi.

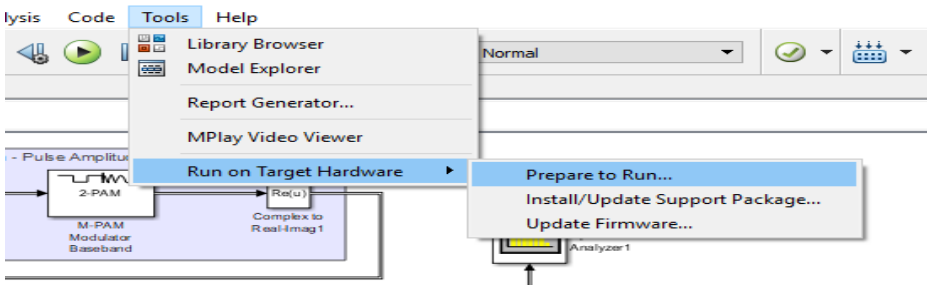


Figure 3.37 : Téléversement du fichier exécutable

Dans la fenêtre suivante (Figure 3.38), nous devons entrer les paramètres concernant le nom d'hôte, le nom d'utilisateur, le mot de passe, le répertoire Build (le fichier où on va exécuter le projet).

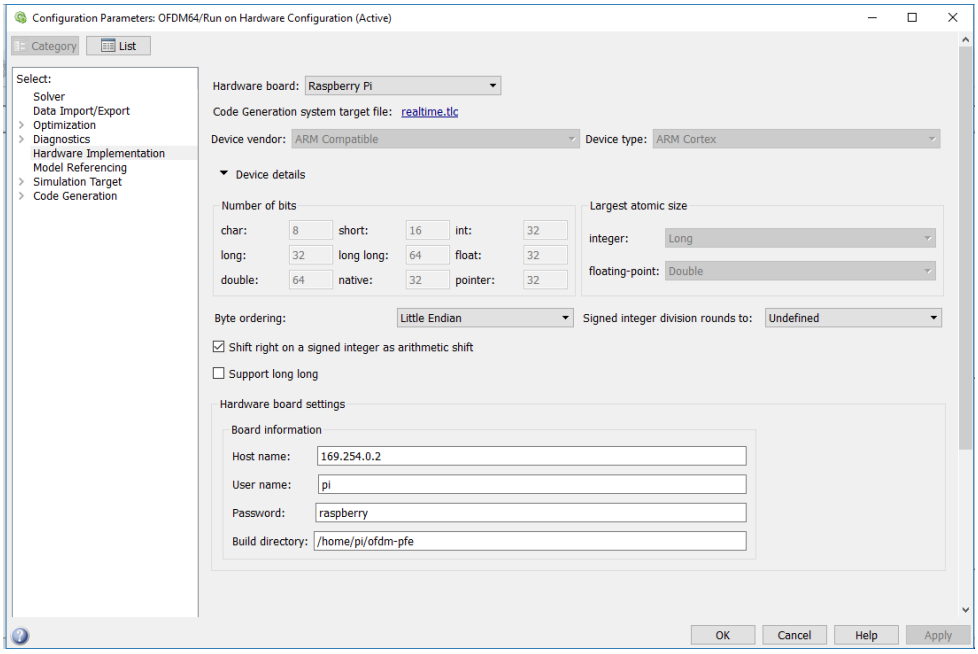


Figure 3.38 : Réglage de paramètres de Raspberry pi

Une fois que Simulink a été correctement configuré pour interagir avec le Raspberry Pi, il est possible de compiler le projet, de générer l'exécutable et d'effectuer l'étape Déployer vers Hard-Ware

« **Deploy to Hard-ware »** (Figure 3.39). Pour mener à bien cette opération, nous devons avoir les droits d'écriture sur le dossier courant de Matlab [18] [19].

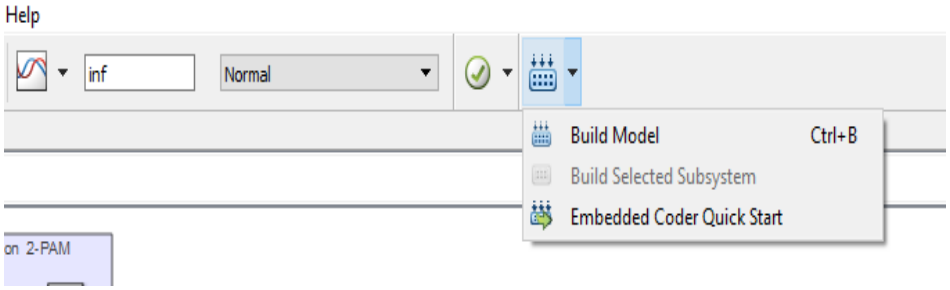


Figure 3.39 : L'étape de générer l'exécutable en Raspberry

Après cette étape le Raspberry pi devient un émetteur OFDM.

Tout projet Simulink déjà chargé sur la carte Raspberry, par le biais de la procédure Deploy to Hardware décrite précédemment, peut être exécuté indépendamment de Simulink. L'exécutable est sauvegardé dans le répertoire Build Directory de la carte mémoire micro SD. En général, il y a deux modes différents pour lancer l'exécution d'un projet sur un Raspberry Pi sous Matlab:

1. Lancement de l'exécution dans Simulink.

2. Lancement de l'exécution avec les commandes Matlab.

4.1.1 Lancement de l'exécution sous Simulink :

Simulink offre une possibilité d'exécuter un projet sur un Raspberry Pi en mode Externe. Pour exécuter le projet sur Raspberry pi, nous choisissons le mode EXTERNAL (Figure 3.40).

Contrairement au cas précédent (**Deploy to Hard-ware**), le mode externe maintient active la connexion entre Simulink et le Raspberry Pi, ce qui permet d'arrêter l'exécution avec le bouton Stop, comme le montre la figure 3.41 et la figure 3.42. Dans ce cas, l'exécutable et les auxiliaires nécessaires à une exécution ultérieure en dehors de Simulink ne seront pas sauvegardés sur la carte mémoire du Raspberry Pi.

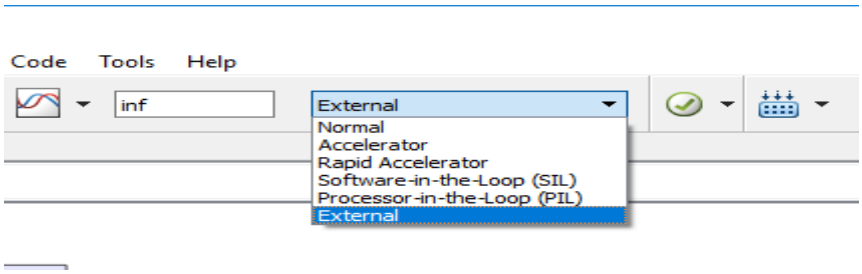


Figure 3.40 : L'exécution en mode simulink

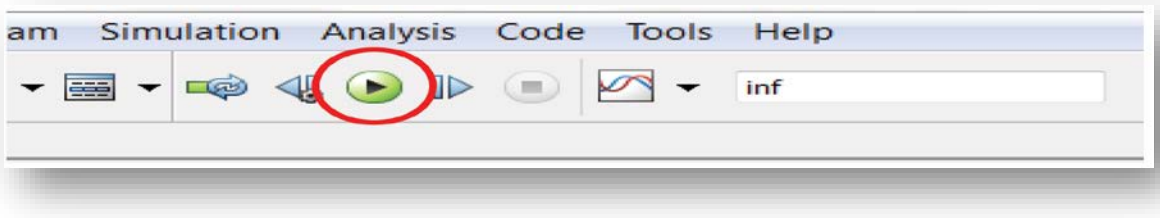


Figure 3.41 : lancer l'exécution en mode externe

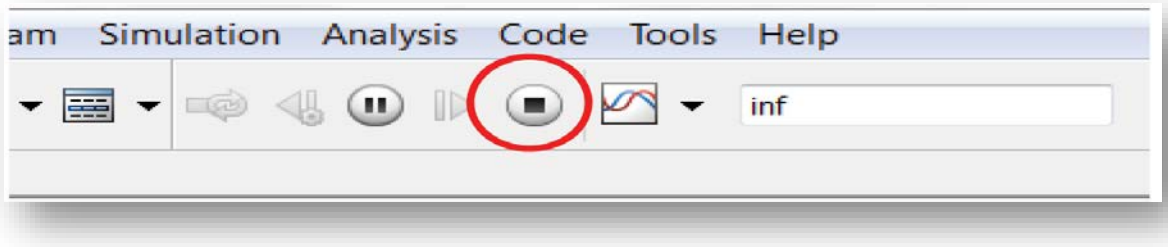


Figure 3.42 : Arrêter l'exécution

Ce mode est particulièrement approprié pendant la phase de conception du projet. En effet, pendant l'exécution en Mode Externe, il est possible de changer les paramètres du projet (par exemple, l'amplitude du sinus) et d'observer leurs effets en temps réel. Cependant, l'exécution en mode externe implique une plus grande charge de calcul pour le Raspberry Pi par rapport à l'exécution activée par la procédure Deploy to Hardware. Ceci est dû à l'échange continu d'informations entre le matériel et Simulink pendant l'exécution en mode externe.

4.1.2 Lancement de l'exécution avec les commandes Matlab :

Pour lancer l'exécution d'un projet à l'aide des commandes Matlab, il est nécessaire de créer l'objet (**h =raspberrypi**) comme indiqué dans la liste 3.1, et d'invoquer l'exécution de la commande, en entrant comme paramètre d'entrée le nom du projet à lancer (la liste 3.2) [14].

```
>> h=raspberrypi('169.254.0.2');
```

Liste 3.1: Commande Matlab pour créer un objet raspberry.

```
>> run(h,'ofdm64')
```

Liste 3.2: Commande Matlab pour exécuter un projet sur un Raspberry Pi.

Si l'exécutable n'est pas sauvegardé dans le BuildDir par défaut, il est nécessaire d'exécuter la commande étendue h=raspberrypi(...) comme indiqué dans la liste 3.3, en spécifiant le nouveau BuildDir.

```
>> h=raspberrypi('169.254.0.2','pi','raspberry','ofdm-pfe')
```

Liste 3.3: Commande Matlab pour créer un objet raspberry.

```
>> stop(h,'ofdm64')
```

Liste 3.4: Commande Matlab pour arrêter l'exécution

5. Utilisation d'un RASPBERRY PI autant qu'émetteur OFDM :

L'interconnexion des différents dispositifs avec notre ordinateur est clairement représentée à la figure 3.43).

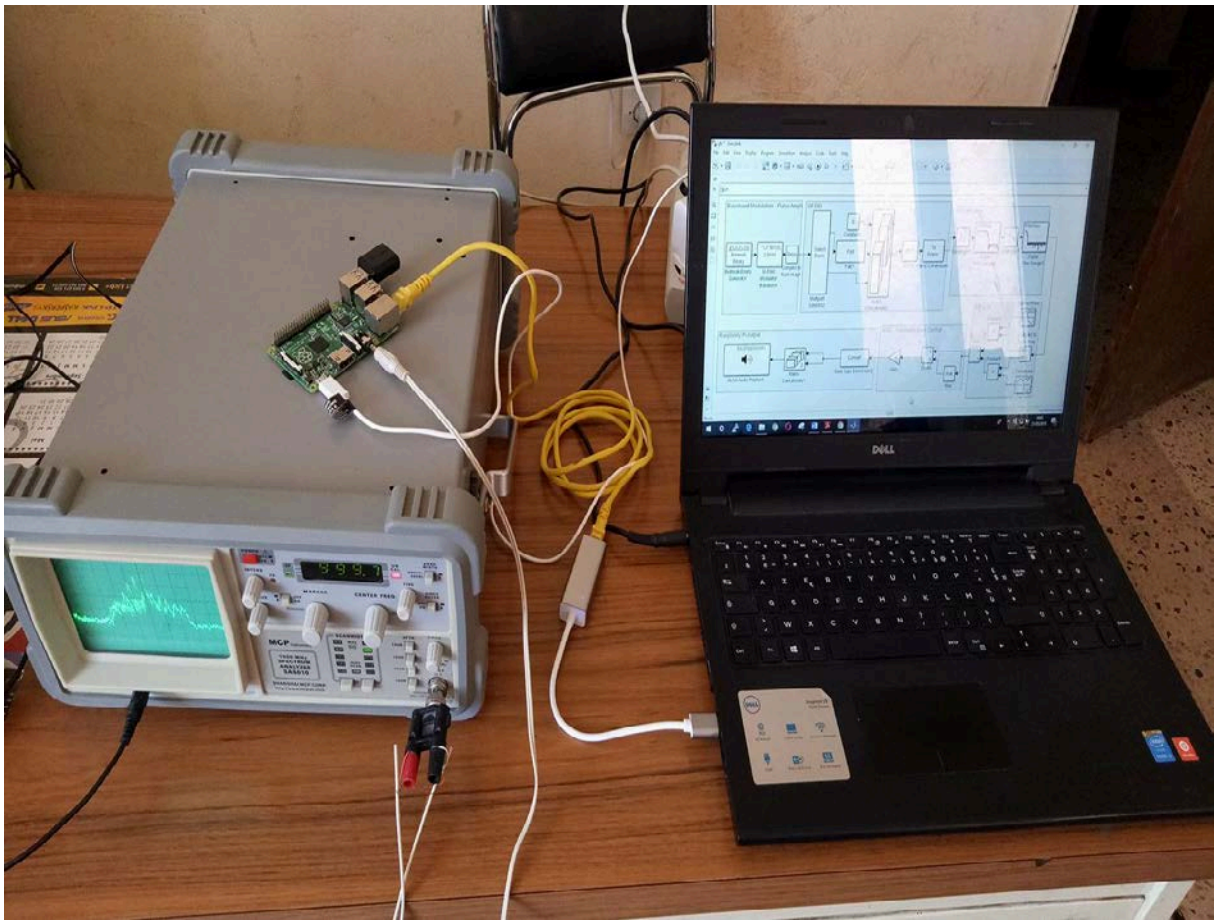


Figure 3.43 : L'interconnexion des différents dispositifs

Après la génération de l'exécutable et la réalisation de l'étape « **Deploy to Hard-ware »** qui est décrite à la section 4.1, le raspberry pi devient un émetteur OFDM. Nous avons connecté notre ordinateur avec le raspberry pi à l'aide d'un câble Ethernet et la sortie ALSA de Raspberry avec un Analyseur de spectre à l'aide d'un câble jack 3.5mm-RCA pour observer le spectre du signal OFDM, mais dommage nous n'avons pas eu le résultat car nous avons utilisé un Analyseur de spectre qui mesure de 150 KHz jusqu'à 1050 MHz et la fréquence maximale de la sortie ALSA est de l'ordre de 48 KHz. L'analyseur disponible est que nous avons utilisé nous n'a pas permis de voir notre signal à cause de sa bande passante.

Pour vérifier le bon fonctionnement du notre émetteur OFDM, une LED a été bronchée entre la PIN 17 et la PIN GND. A chaque fois que l'émetteur est met en service, la LED clignote.

6. Conclusion:

Dans ce chapitre nous avons présenté l'émetteur OFDM que nous avons conçu et implémenté sur la carte Raspberry Pi.

Nous avons présenté le schéma Simulink de cet émetteur et la description de chaque sous bloc. Nous avons aussi présenté les résultats de simulation sous Simulink et l'implémentation de cet émetteur sur la carte Raspberry Pi.

CONCLUSION GÉNÉRALE

Les Télécommunications font partie des technologies qui ont révolutionné notre mode de vie.

L'augmentation des besoins en débit se heurte à la nature des canaux qui sont de type multi-trajets dans plusieurs applications.

L'influence de ces canaux sur le signal augmente avec le débit de transmission. La technique de modulation multiporteuses (OFDM) apporte une solution à ce problème.

Dans le premier chapitre, nous avons présenté les problèmes que rencontre la transmission à débit élevé, et qui sont le problème de trajets multiples et de sélectivité de canal. Ensuit nous avons présenté la technique OFDM qui apporte une solution à ces problèmes.

Dans le second chapitre, nous nous sommes intéressés à la carte Raspberry Pi. Nous avons donc commencé par définir cette carte et donner tous les modèles existants. Ensuite, nous avons présenté tous les étapes nécessaires du processus d'installation de package de Raspberry Pi.

Dans le troisième chapitre, nous avons présenté l'émetteur OFDM que nous avons conçu. Nous avons donc commencé par donner le schéma Simulink de cet émetteur et une définition de chaque sous bloc. Ensuite, nous avons présenté les résultats de simulation sous Simulink.

Enfin, et après vérifié le bon fonctionnement de l'émetteur OFDM sous Simulink en mode Normal et External nous avons implémenté notre émetteur OFDM sur la carte Raspberry Pi.

BIBLIOGRAPHIE

[1] Jawad Nakad, « Allocations de ressources radio dans un réseau local sans fil (WLAN) de type OFDM », Soutenance le Lundi 22-Dec-2003.

[2] Muhammad Imadur Rahman, Suvra Sekhar Das, Frank H.P. Fitzek, « OFDM Based WLAN Systems », Center for TeleInFrastruktur (CTiF), Aalborg University
Neils Jernes Vej 12, 9220 Aalborg Øst, Denmark, 18 February 2005.

[3] Paul Guanming Lin, « OFDM SIMULATION in MATLAB », A Senior Project
Presented to the Faculty of California Polytechnic State University San Luis Obispo, June 2010.

[4] Emad S. Hassan, «Multi-CarrierCommunicationSystems with Examplesin MATLAB », by Taylor & Francis Group, Boca Raton London New York, LLCCRC Press is an imprint of Taylor & Francis Group, an Informa businessNo, 2016.

[5] Understanding an OFDM transmission:
http://www.dsplog.com/2008/02/03understanding-an-ofdm-transmission/.

[6] Minimum frequency spacing for having orthogonal sinusoidals
http://www.dsplog.com/2007/12/31/minimum-frequency-spacing-for-having-orthogonal-sinusoidals/

[7] Virginie Dégardin.Analyse de la faisabilité d'une transmission de données haut Débit sur le réseau électrique basse tension. Thèse de doctorat en électronique Université des sciences et technologies de Lille Décembre2002.

[8] GAL YNA PISKONOVA, « TRANSMISSION OFDM POUR LA TÉLÉPHONIE CELLULAIRE », MONTRÉAL, 19 DÉCEMBRE 2003.

[9] M.C.D.Maddocks, B.Sc(Eng.), C.Eng., M.I.E.E., « AN INTRODUCTION TO DIGITAL MODULATIONAND OFDM TECHNIQUES », Research Departement, Engineering Division THE BRITISH BROADCASTING CORPORATION BBC RD 1993/10.

[10] Louis Litwin and Michael Pugel, « The principles of OFDM », January 2001.

[11] ABDELALI EL KHETTABI, « CONCEPTION DU SYSTEME DE TRANSMISSION OFDM CODE POUR LES APPLICATIONS À HAUT DÉBIT », MONTREAL, LE 1 FEVRIER 2008.

[12] Yong Soo Cho : Chung-Ang University, Republic of Korea. Jaekwon Kim : Yonsei University, Republic of Korea. Won Young Yang : Chung-Ang University, Republic of Korea. Chung G. Kang : Korea University, Republic of Korea, « MIMO-OFDM WIRELESS COMMUNICATIONS WITH MATLAB », IEEE PRESS, John Wiley & Sons (Asia) Pte Ltd, 2 Clementi Loop, # 02-01, Singapore 129809, 2010.

[13] Uma Shanker Jha Ramjee Prasad, « OFDM Towards Fixed and Mobile Broadband Wireless Access », Library of Congress Cataloging-in-Publication Data, British Library Cataloguing in Publication Data, ISBN-13: 978-1-58053-641-7, 2007.

[14] SIRIPHAT POMYEN, « SIGNAL AND IMAGE PROCESSING WITH MATLAB ON RASPBERRY PI PLATFORM », Examiner and topic approved by the Faculty Council of the Faculty of Computing and Electrical Engineering on 4 February 2015.

[15] Russell Barnes, David Whale, Laura Clay, Phil King, Lorna Lynch, « The Official Raspberry Pi Projects Book », 2014

[16] Anuja Apte, « Set up and Blink - MATLAB and Simulink with Raspberry Pi », adafruit lesrning system, Last updated on 2015-04-16 05:50:07 PM EDT.

[17] Technical Marketing Brian.McKay@mathworks.com.

[18] Gianni Pasolini, Flavio Zabini Wilab, University of Bologna Bologna (Italy), Alessandro Bazzi IEIIT-CNR Bologna (Italy), Stefano Olivieri The MatWorks Inc. Turin (Italy), « A Software Defined Radio Platform with Raspberry Pi and Simulink », 24th European Signal Processing Conference (EUSIPCO), 2016.

[19] Simulink and Raspberry Pi Workshop Manual, « A brief workshop on Simulink Support for Project Based Learning with Raspberry Pi », Mathwork, Inc . 04/14/2015.