TP4 : QPSK - Partie RF, filtre d'émission

Introduction

Nous consacrons deux séances à la QPSK. La première traite du modulateur, de la partie RF du démodulateur, de l'implémentation d'un filtre de mise en forme pour réduire la bande côté émetteur.
Dans la seconde, on abordera la partie bande de base du récepteur, et en particulier l'élimination du bruit et la récupération du timing symbole.

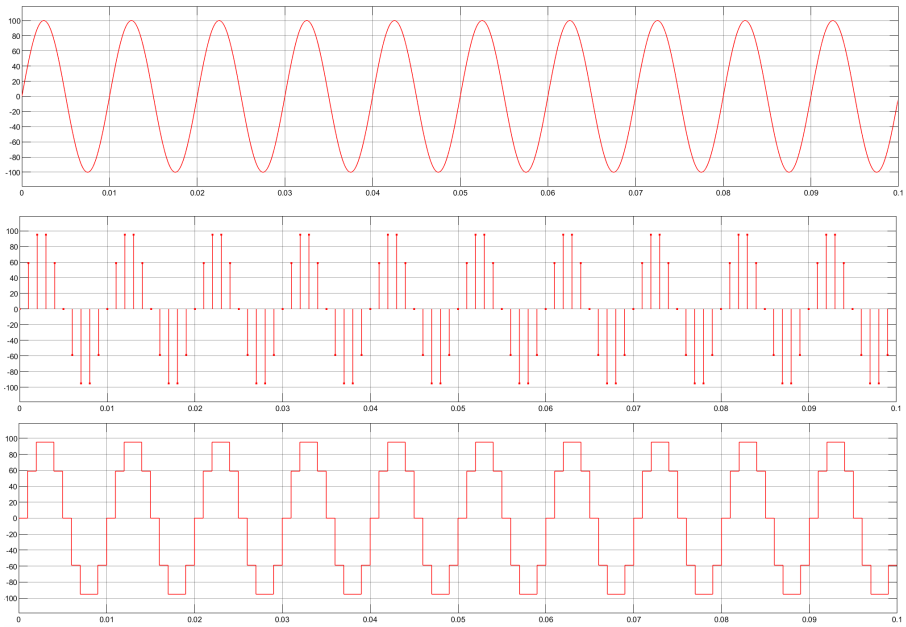
Préliminaire

On considère un signal à temps continu, $s(t)$ (déterministe, à puissance moyenne finie). Sa transformée de Fourier, $S(f)$, occupe une bande $[-B, B]$.

On échantillonne $s(t)$ avec ue fréquence f_e ; on obtient un signal $s_{samp}(t)$.

1. Quelle contrainte la fréquence f_e doit-elle respecter ?
2. Tracer l'allure du spectre du signal échantillonné (pas de calcul demandé).

A présent, on cherche à reconstituer $s(t)$ à partir de $s_{samp}(t)$; appelons le signal obtenu $s_{ana}(t)$. Technologiquement, le passage d'un signal numérique (représenté par notre signal échantillonné) à un signal analogique se fait par un DAC. Celui-ci *maintient constant* chaque échantillon pendant un temps T_e, et on obtient une allure "en marche d'escaliers" caractéristique.
Ci-dessous, on a représenté ces trois signaux, $s(t)$, $s_{samp}(t)$ et $s_{ana}(t)$ dans le cas simple où $s(t)$ est une sinusoïde :



La question est : quel est le spectre de $s_{ana}(t)$?

Le modèle mathématique du DAC fait appel à la fonction rectangle ; en effet, à partir d'un échantillon de $s_{samp}(t)$, on peut obtenir le motif de $s_{ana}(t)$ par convolution avec la fonction qui vaut 1 sur $[0, Te]$ et est nulle ailleurs (que je note $rect_{T_e}(t)$:

$$\underbrace{s(kT_e)\delta(t-kT_e)}_{\text{un échantillon de } s_{samp}(t)} * rect_{T_e}(t) = \underbrace{s(kT_e)\ rect(t-kT_e)}_{\text{maintien constant pendant } T_e}$$

Et donc, si l'on considère tous les échantillons :

$$\underbrace{\sum_k s(kT_e)\delta(t-kT_e)}_{s_{samp}(t)} * rect_{T_e}(t) = \underbrace{\sum_k s(kT_e)\ rect(t-kT_e)}_{s_{ana}(t)}$$

A partir de là, on obtient le spectre de $s_{ana}(t)$ par *multiplication* du spectre de $s_{samp}(t)$ (connu) avec la transformée de Fourier de la fonction rectangle, un sinus cardinal qui s'annule tous les f_e.

3. Tracer l'allure du spectre de $s_{ana}(t)$ (pas de calcul demandé).

Pour vérifier tout cela ouvrir le modèle simulink `sampling_sin`. Double-cliquer sur la source et noter la fréquence de la sinusoïde ainsi que sa fréquence d'échantillonnage. Lancer la simulation, observer le signal temporel (scope) et le spectre sur l'analyseur.

A présent, on aimerait que notre sinusoïde soit semblable à la sinusoïde $s(t)$ tracée ci-dessus, donc plus lisse. Pour cela, on doit faire suivre le DAC d'un filtre appelé *filtre de lissage* ou *filtre de restitution*.

4. Quelles doivent être les caractéristiques de ce filtre ? (passe-bas, passe-haut, etc ? fréquence de coupure ?)

5. Ajouter un bloc `Analog Filter Design`, le configurer (par exemple de type `Butterworth`), observer le résultat sur le scope et sur l'analyseur.

QPSK : rappels

Principe de base

En QPSK, les bits sont regroupés deux par deux ; chaque couple de bits détermine *la phase à l'origine des temps* du signal modulé. La forme d'onde obtenue pour un groupe de deux bits donné est appelée *symbole*.

Sur un temps symbole T_s, le signal modulé a pour expression :

$$s(t) = A\cos(\omega_c\ t + \varphi)$$

où φ est la phase à l'origine des temps, constante sur un temps symbole, et vaut $\pi/4$, ou 3 $\pi/4$, ou $5\pi/4$, ou 7 $\pi/4$.

Le signal $s(t)$ peut aussi être exprimé de la façon suivante, toujours sur le temps symbole :

$$s(t) = I(t)\cos(\omega_c\ t) - Q(t)\sin(\omega_c\ t) \qquad (1)$$

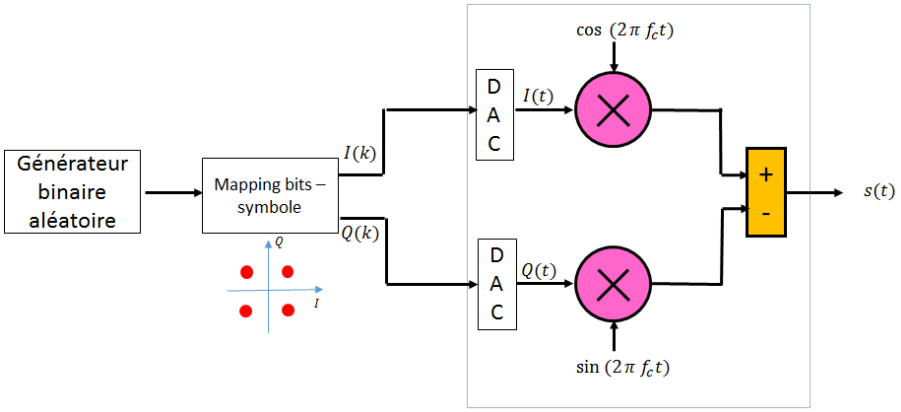
avec

$$I(t) = A\cos(\varphi)$$
$$Q(t) = A\sin(\varphi)$$

$I(t)$ et $Q(t)$ sont des signaux *continus en temps* et, ici, *constants sur le temps symbole considéré* (car φ est constante).

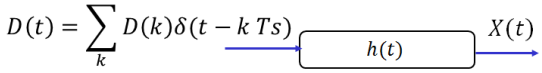
Émetteur

De l'expression (1), on peut tirer un schéma de modulateur :

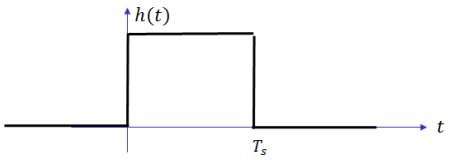


Sur ce schéma, il est très important de distinguer la partie *Bande de base*, à gauche avant la transposition en fréquence, c'est-à-dire la partie numérique et le DAC ; de la partie *RF*, dans laquelle principalement se fait la transposition vers la fréquence centrale du canal, par multiplication par la porteuse, et l'amplification (non représentée). Cette partie traite de signaux à des fréquences élevées (plusieurs GHz), hors de portée des sytèmes numériques (imaginez que pour représenter une sinusoïde à 2 Ghz avec 10 points par période, il faudrait 20 Gech/s...)
La partie encadrée du schéma ci-dessus se situera dans l'USRP.

Noter qu'un *modèle à temps continu* très souvent utilisé pour modéliser le DAC (en particulier pour appliquer la formule de Bennett) est :



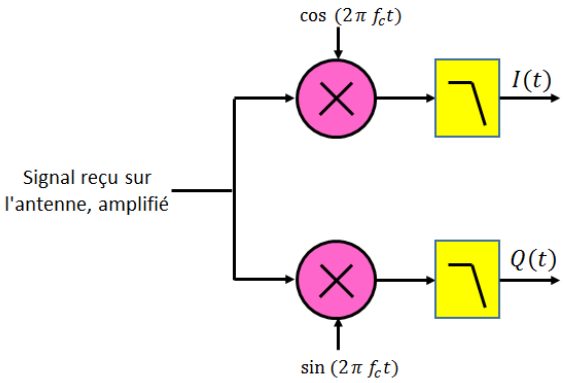
où $D(k) = I(k) + jQ(k)$, $X(t) = I(t) + jQ(t)$ et $h(t)$ est un filtre de réponse impulsionnelle rectangle :



Ce modèle fait qu'on parle couramment de *filtre rectangle* pour le schéma du modulateur de la page 1. Retenir que "filtre rectangle" signifie "$I(t)$ et $Q(t)$ constants sur le temps symbole".

Récepteur

En réception, on utilise le schéma suivant pour récupérer $I(t)$ et $Q(t)$:



Noter que la partie bande de base n'apparaît pas sur ce schéma. Nous verrons dans le prochain TP que si la récupération des bits à partir $I(t)$ et $Q(t)$ est simple dans une première approche, elle pose en réalité de vrais difficultés à cause des imperfections des oscillateurs et horloges.

1 Émetteur avec filtre rectangle

Nous allons commencer par simuler l'émetteur dans son intégralité, bande de base + RF. On rappelle pour une transmission réelle, le DAC et la partie RF se trouveront dans l'USRP.

1. Ouvrir le fichier `simu_QPSK_1`. Faire `CTRLE+D` pour faire afficher les couleurs. Le modèle est déjà dessiné : vérifier qu'il correspond au schéma donné ci-dessus.

2. On commence par s'occuper de la partie bande de base à gauche. Double cliquer sur le bloc `Bernoulli Binary`, qui est le générateur de bits. Les bits sortent deux par deux. Le `sample time` renseigné ici est le temps bit ; quelle est sa relation avec le temps symbole ? Faire `CRTLE+J` pour vérifier que cette partie du schéma est cadencée par la fréquence symbole.

3. Double-cliquer sur le bloc de Mapping, puis sur `View constellation`. On voit que chaque nombre complexe $I(k)+jQ(k)$ est en fait associé à un *entier* de 1 à 4. Chercher le paramètre à modifier pour que ce soit un couple de bits plutôt qu'un entier (ne pas passer à la suite tant que ce n'est pas fait), puis relever la constellation.

4. Observer les bits, $I(k)$ et $Q(k)$ sur le scope de gauche, et interpréter, à l'aide de la constellation.

5. On passe à présent à la partie RF (encadrée). Quels blocs génèrent $\cos(\omega_c t)$ et $\sin(\omega_c t)$? Que vaut la fréquence porteuse ? Qu'est-ce qui permet d'avoir un cosinus ou un sinus ?

6. Observer le signal modulé dans le scope de droite, mettre en évidence les changements de phase à l'origine des temps.

7. Décommenter le bloc `Spectrum analyser` à la sortie (clic droit > `Uncomment`). Faire passer le temps de simulation à `inf`, et quand le spectre s'affiche, zoomer sur la bande intéressante (autour de la fréquence porteuse bien sûr) en utilisant , pour garder la bande $[0.6, 1.4]$ kHz environ.
   Relever la largeur du lobe principal ; la comparer au débit de symboles. Relever la DSP à la fréquence porteuse, en dBm/Hz, puis en mW/Hz.

   Comparer ces deux valeurs (largeur du lobe principal, valeur en 0) à ce que donne la formule théorique (Bennett) :
   $$S_{XX}(f) = \frac{T_S}{2} sinc^2\left[\pi(f+f_c)T_s\right] + \frac{T_S}{2} sinc^2\left[\pi(f-f_c)T_s\right]$$

   Rappels sur l'analyse spectrale : le traitement réalisé par le bloc `Spectrum analyser sans moyennage` calcule en fait la DSP d'un signal *déterministe*, ou alors *d'une seule réalisation* d'un processus aléatoire. Or la DSP d'une seule réalisation, aléatoire par définition, est aléatoire (ie en chaque f, on obtient une valeur aléatoire).
   Pour un processus aléatoire, il faut normalement prendre *l'espérance* des DSP de plusieurs réalisations ; donc en pratique moyenner celles-ci [1]. C'est pourquoi on a réglé le paramètre `Averages` à 100. Pour vous en convaincre, faire varier ce paramètre entre 1 et 500 par exemple, puis le remettre à 100.
   Le nombre de points sur lequel on calcule la FFT, N, définit la *résolution*, c'est-à-dire l'écart fréquentiel entre deux points. Il est réglé par la `Window length`, avec `NFFT` sur `Auto`. Diminuer ce paramètre `Window Length` à 4096 ; la résolution vaut alors
   $$\frac{f_e}{4096}$$
   Ici f_e est fixé par le bloc `Zero-order Hold` situé juste avant l'analyseur et vaut 64 kHz ; soit une résolution de 16 Hz environ. Vous pouvez vérifier que dans un intervalle 100 Hz, il y a un peu plus de 6 points.

2 Le récepteur, partie RF

1. Ouvrir le fichier `simu_QPSK_2`. La partie RF du récepteur est ébauchée. (Pour nous, en pratique, cette partie sera dans l'USRP.)

1. Le résultat de ce moyennage est toujours aléatoire, mais la variance autour de la valeur théorique peut être aussi faible que l'on veut si on moyenne suffisamment de DSP de réalisations – c'est la loi des grands nombres.

2. Connecter les blocs et configurer les deux filtres passe-bas. Par défaut, ceux-ci sont mal configurés et ça ne fonctionnera pas si vous ne changez pas le paramètre `Passband edge frequency` ; attention, il faut entrer `2*pi*fp` où `fp` est la fréquence extrême de la bande passante. L'objectif est d'obtenir sur le scope un `I_demod` identique au `I` de l'émetteur (aux artefacts liés à la réponse du filtre près).

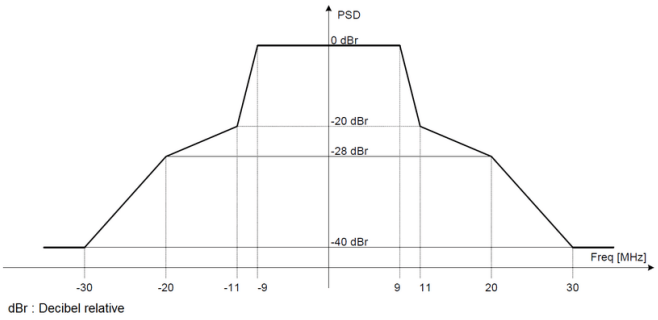
   Indication : nous avons déjà parlé de ce filtre analogique dans le TP1 lors du passage en fréquence intermédiaire (son rôle est identique ici). Vous pouvez vous aider de l'analyseur de spectre qui suit en le connectant avant le filtre.

3 Le filtre d'émission (ou de mise en forme)

Revenons à l'émetteur. Le problème du filtre rectangle, ce sont les lobes secondaires du spectre. Ces lobes ne sont pas vraiment utiles car la majorité de la puissance se situe dans le lobe principal ; pour autant, on ne peut pas émettre un autre signal là où se trouvent les lobes secondaires, car ce dernier serait perturbé.

Il faut comprendre que le spectre électromagnétique est complètement occupé. C'est une ressource rare ! L'État peut, pour le déploiement des réseaux mobiles par exemple, libérer des bandes ; le prix que payent les opérateurs pour avoir le droit de les exploiter est une bonne indication de leur valeur. En 4G, dans la bande des 700 MHz, les opérateurs ont payé près de 1 Milliard d'euros pour 10 MHz de bande.
(`www.lesechos.fr/2015/11/les-encheres-pour-les-frequences-4g-sont-bel-et-bien-terminees-281970`)
Tout ça pour dire qu'on ne peut se permettre de gâcher des morceaux de spectre à cause de lobes secondaires. Tous les standards, qu'il s'agisse de réseaux mobiles, Wifi, ADSL, etc, spécifient un *spectal mask* qui donne l'atténuation minimum du signal par rapport à son maximum (on note l'attébuation en dBr, ce qui signifie dB relativement au maximum), à certaines distances de la fréquence porteuse du canal.

A titre d'exemple, le masque spectral du 802.11 a (Wifi), où le canal a une largeur de 20 MHz :

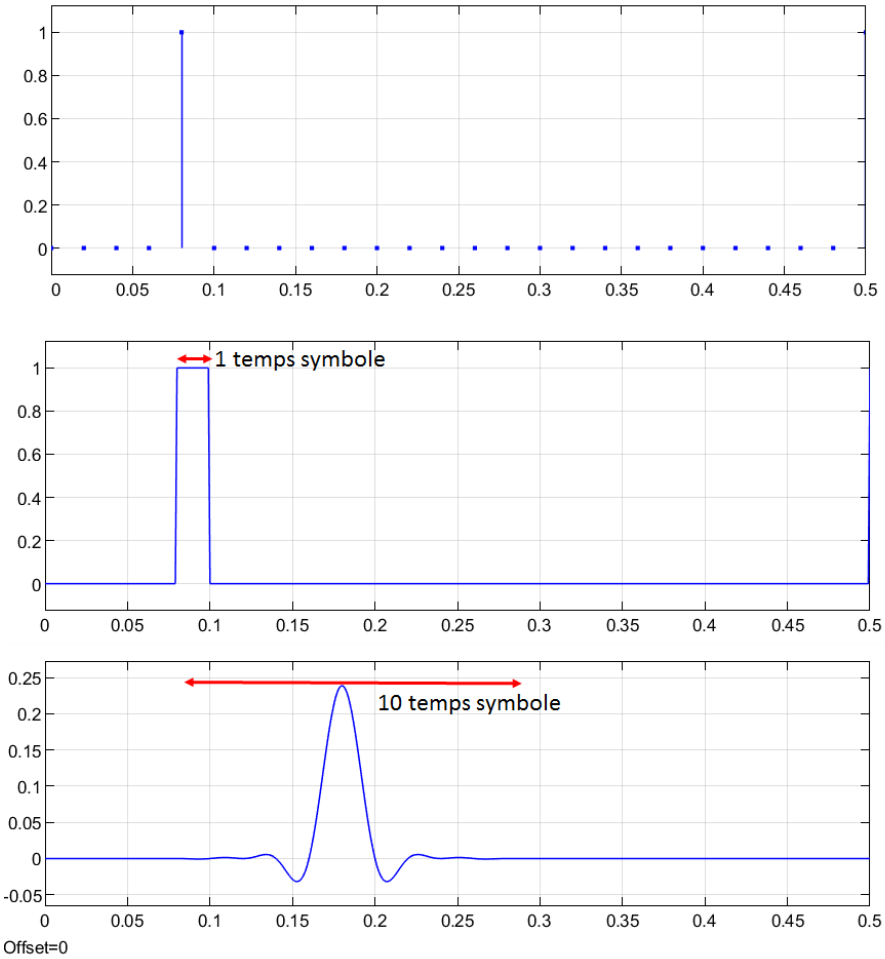


On voit qu'à 9 MHz du centre (sur le bord droit du canal), le signal n'a pas besoin d'être atténué ; en revanche, à 11 MHz (sur le bord gauche du canal voisin), il doit être atténué de 20 dB (par rapport au maximum) ; et à 20 MHz (centre du canal voisin) de 30 dB.

Pour éviter les lobes secondaires, on utilise un *filtre de mise en forme*, qui réduira la bande occupée au strict minimum.
Le filtre en cosinus surélevé (raised cosine filter) est très souvent utilisé car, en plus de réduire la bande, il permet d'éviter l'Interférence Entre Symboles (IES ou ISI en anglais). Nous reviendrons sur cet aspect lors du TP5.
Le facteur de roll-off, souvent noté α, est un paramètre du filtre en cosinus surélevé qui détermine

dans le domaine fréquentiel la raideur de coupure (plus le roll-off est faible, plus la pente est raide) ; et dans le domaine temporel, le temps de montée (plus le roll off est faible, plus le temps de montée est important – plus le signal est déformé).

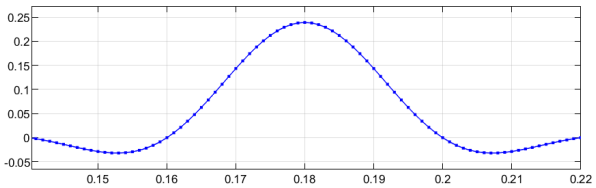
Pour implanter ce filtre, on cherche à générer en bande de base un signal discret ayant l'allure désirée (donc si l'on utilise une USRP, avec Simulink, avant envoi vers l'USRP).



On a représenté ci-dessus, avec toujours 50 symboles/s, soit un temps symbole de 0.02 s :

— un échantillon du signal discret $I(k)$ ou $Q(k)$ ($I(k)$ et $Q(k)$ contiennent un seul échantillon par temps symbole, défini par le couple de bits) ;
— la sortie du DAC si on lui envoie directement cet échantillon (filtre rectangle) ;
— la réponse pour un filtre en cos surélevé.

Si un signal discret doit représenter cette sortie, il doit nécessairement comporter *plusieurs échantillons par temps symbole*. Ci-dessous, un zoom sur 4 temps symboles, en mettant en évidence les échantillons (ici on a 10 échantillons par temps symbole) :



Noter que si l'on avait un seul échantillon par temps symbole, comme ci-dessus le temps symbole vaut 0.02, ce serait aux instants 0.14, 0.16, 0.18, 0.20, 0.22. On aurait des échantillons nuls sauf en 0.18, donc pas du tout l'allure de la réponse souhaitée ! Le rapport entre les sample rates à la sortie et à l'entrée du filtre est appelé *facteur de sur-échantillonnage* ou *OverSampling Ratio* (OSR).

Pour générer le signal discret qu'on enverra à l'USRP, deux étapes :
- une étape d'`Upsample` qui consiste à insérer des 0 entre les échantillons de $I(k)$ et $Q(k)$ (on insère OSR-1 échantillons nuls) ;
- et une étape de filtrage numérique par un fitre FIR, dont les coefficients sont obtenus en échantillonnant la réponse impulsionnelle théorique.

1. Ouvrir de nouveau `simu_QPSK_1`, l'enregistrer sous un autre nom, par exemple, `simu_QPSK_3`. Dans la partie bande de base, avant le DAC, ajouter un bloc `Upsample`. Choisir un OSR compris entre 4 et 10 (soit 4 à 10 échantillons par temps symbole). Dans `Rate options`, choisir `Allow multirate processing`. Que vaut f_e en sortie de ce bloc ? (Pour vérifier votre réponse : `CTRLE+D, CTRLE+J`)

2. Ajouter derrière un bloc `Discrete FIR filter`. Entrer pour les coefficients du filtre la commande
   `rcosdesign(`α`, Span, OSR, Shape)`, où α est le facteur de roll-off ; `OSR` le facteur de sur-échantillonnage ; `Shape` une chaîne de caractères parmi '`normal`' et '`sqrt`'. Pour nous, pour le moment, c'est '`normal`' qu'il faut choisir. (Nous y reviendrons dans le TP5).

   Noter que la réponse impulsionnelle théorique du filtre a une durée infinie ; elle doit être *tronquée* à un certain nombre de temps symbole (ci-dessus par exemple, elle est tronquée pour ne durer que 10 temps symbole). Le paramètre `Span` est le nombre de temps-symbole sur lequel s'étale la réponse impulsionnelle tronquée (forcément pair, typiquement 6 à 12).

3. Avant de tester le filtre, on peut observer sa réponse théorique sous Matlab. Pour cela, taper dans la fenêtre de commandes Matlab (en remplaçant les paramètres par les valeurs numériques choisies) :
   `my_filter=rcosdesign(`α`, Span, OSR, Shape)`
   `fvtool(my_filter)`

   Vous devez voir apparaître la réponse fréquentielle. Pour accéder à la réponse impulsionnelle, cliquer sur 
   Sur combien d'échantillons s'étale cette réponse ? Pouvait-on déduire cela des paramètres du filtre ? Au bout de combien d'échantillons la réponse à une impulsion (un Dirac) atteint-elle son maximum ? Quel est le retard occasionné par le filtre, en temps symbole ?

4. Simuler. Observer seulement l'oscilloscope. Vérifier, en zoomant si nécessaire, que vous obtenez l'allure caractéristique en marches d'escalier, avec le temps T_e prévu. Observer l'analyseur. Vérifier que vous obtenez les répliques du spectre autour de f_e, $2f_e$, etc.

5. Ajouter un bloc `Analog Filter Design` comme dans la partie préliminaires, et observer le résultat.

6. A présent, on suppose qu'on utilise des canaux de largeur 100 kHz. le cahier des charges est le suivant :
   - à 50 kHz du centre, le signal doit être atténué de 20 dB minimum par rapport au centre ;

— à 100 kHz, le signal doit être atténué de 40 dB minimum.

Représenter le masque spectral correspondant.

7. Le bloc `spectrum analyser` permet d'effectuer un certain nombre de mesures ; en particulier, on peut obtenir l'`Occupied BandWidth` (OBW), qui est une façon classique de mesurer l'encombrement spectral. Il s'agit de la bande dans laquelle se trouve un certain pourcentage de la puissance totale du signal, en général 98 ou 99 %. Pour la mesurer, aller dans `Tools > Measurements > Channel Measurements`. Dans `Span`, mettre la bande dans laquelle sera calculée 100% de la puissance : pour nous, mettre la largeur du canal, 100 Hz ; dans `CF` (Carrier Frequency), entrer la fréquence centrale (ici 1 kHz). Prendre comme pourcentage 99.5% ; utiliser une taille de fenêtre de 4000 (pour avoir un nombre entier de symboles – spectre moins déformé), en laissant `NFFT` sur `Auto` ; et une fenêtre rectangulaire. Mesurer l'OBW pour un roll-off $\alpha = 0.2$ puis 0.5 et 0.8. Conclure sur l'influence du roll-off.

8. On peut aussi spécifier un masque spectral et vérifier que le signal que nous émettons est conforme. Pour cela, on définit une matrice contenant, sur chaque ligne, les coordonnées d'un sommet de notre masque. Par exemple, les trois premières lignes seront `[0.9e3 -40; 0.95e3 -20; 0.95e3 0]` pour (90 kHz, -40 dBr), (95 kHz, -20 dBr) et (95 kHz, 0 dBr).
   Dans la fenêtre de commande Matlab, mettre votre masque dans ue variable que vous appellerez `mask`.
   Puis, revenir à l'analyseur, désactiver la mesure de l'OBW, et faire à la place `Tools > Spectral Mask`. Dans `Masks`, choisir `Upper`, et dans `Upper limits` : écrire `mask` (votre variable), dans `Reference level` : `Spectrum peak`. Le nombre de points sur lequel on calcule la FFT influence le calcul ; mettre 40 000 au lieu de 4000 (ne laissant `NFFT` sur `Auto`). Essayer plusieurs valeurs de roll-off.

   Il va de soi qu'avec le filtre rectangle, le signal ne passerait pas le test !