La radio logicielle

par l'exemple SDR-1000 et filtrage numérique

F4DAN, Christophe BOURGUIGNAT

La radio logicielle (SDR : Software Defined Radio), parfois appelée "radio reconfigurable" ou "radio intelligente" est aujourd'hui un des sujets chauds parmi les multiples activités radioamateur.
Le nombre d'utilisateurs de cette nouvelle technologie est sans cesse grandissant, et le terrain d'expérimentation est vaste. Ce succès est illustré par les nombreuses réalisations et projets qui fleurissent sur le Net et dans les magazines spécialisés.

Rappelons le principe de base de la radio logicielle : construire son transceiver par un assemblage de code informatique, et non plus de composants électroniques !

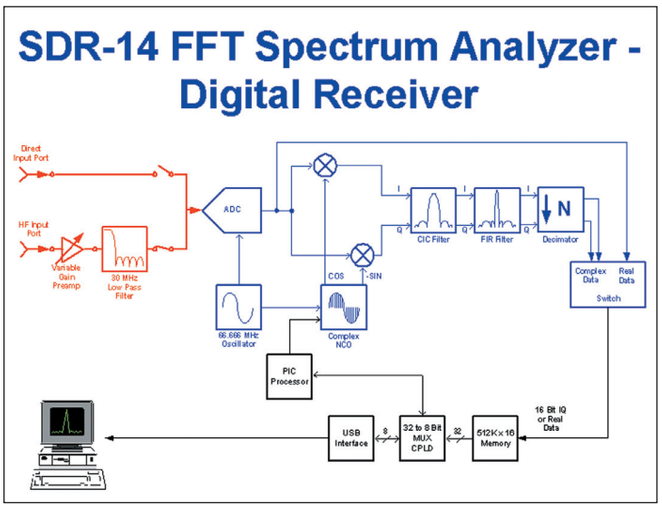
Avouez que l'idée a de quoi séduire.

Une technologie arrivée à maturité ?

La radio logicielle est explorée par les OM depuis maintenant plus de 10 ans, certains travaux remontant au moins à 1994 [KC1HR]. Pour l'anecdote, la réalisation de KC1HR utilise alors un convertisseur analogique numérique avec 6 bits de résolution, soit une " misérable " (terme de KC1HR…) dynamique de 35 dB. Mais ce qui est important pour lui est de montrer la faisabilité d'un tel concept.

Depuis, de nombreuses réalisations ont été décrites.
Beaucoup d'entre elles utilisent un procédé de numérisation simple, bon marché, et efficace : une carte-son de PC, permettant de numériser la sortie BF d'un récepteur ([LINPSK] par exemple) ou une FI analogique centrée sur quelques kHz ([SDR-1000] par exemple).
D'autres n'utilisent pas de carte son, mais un convertisseur analogique-numérique dédié, agissant au niveau d'une FI ([DSP-10] ou [F1OAT] par exemple).
Malgré tout, ces réalisations continuent à mettre en œuvre, au moins en début de chaîne de réception, un changement de fréquence analogique traditionnel (mélangeur + oscillateur local). Le pas du "tout numérique", de la numérisation en sortie d'antenne, est en effet longtemps resté difficile à franchir (au moins dans le domaine amateur). La raison principale provenait des limitations des Convertisseurs Analogiques Numériques : ils manquaient de dynamique, étaient bruyants, et onéreux ([KD70]).

Avec les progrès des composants, le rêve devient maintenant peu à peu réalité :
quelques réalisations permettent aujourd'hui une numérisation directe des signaux HF, sans passer par une FI. Citons l'USRP de gnuradio [GNURADIO], et le récepteur SDR-14 [RFSPACE]. A titre illustratif, le schéma bloc de ce dernier est reproduit ci-dessous.



La radio logicielle par l'exemple

Les avantages de la radio logicielle (facilité de reconfiguration, nouveaux modes numériques, performances accrues, etc. …) ont déjà été souvent exposés, tout comme ses principes de base (conversion analogique / numérique, FFT, filtrage numérique, etc. …). Ces thèmes peuvent faire l'objet à eux seuls d'articles complets et fouillés.

Aujourd'hui, c'est sous un angle nouveau et original nous semble-t-il, que nous allons aborder la radio logicielle. Nous avons choisi de reproduire des extraits de codes informatiques issus de différents projets SDR, et de les commenter de façon la plus didactique possible, comme s'il s'agissait de schémas électroniques… Pourquoi utilise-t-on cette variable ? A quoi sert cette boucle ?

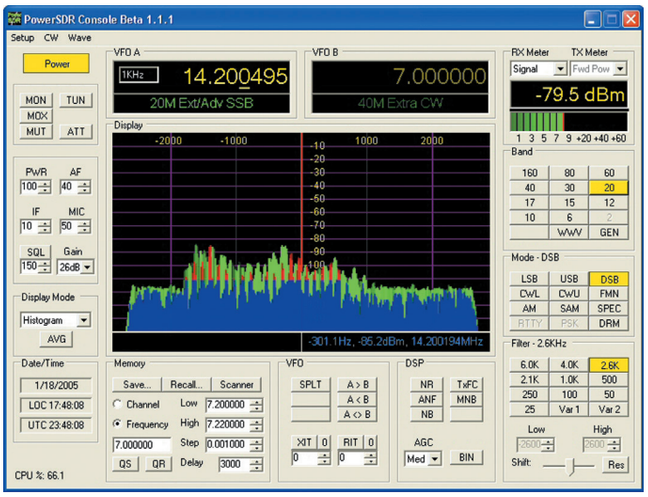
Pourquoi faire de cette façon plutôt que de telle autre ? Voilà autant de questions auxquelles nous répondrons au cours de nos descriptions. En cours d'exposé, les rapprochements avec la théorie seront faits dès que nécessaire.

Les extraits retenus ont été soigneusement choisis pour leur caractère illustratif, et leur niveau de complexité modéré. Les explications ont été rédigées de façon à ce qu'elles puissent être compréhensibles même si le lecteur ne possède pas de connaissance particulière en programmation.
A travers ces explications, nous souhaitons permettre au lecteur de comprendre un peu mieux ce qui se passe quand il utilise son logiciel radio préféré. Il s'agit également de dresser un panorama des différentes techniques utilisées en SDR, et pourquoi pas parfois de comparer ou critiquer (de manière constructive !!) différentes implémentations. Enfin, nous espérons encourager l'expérimentation dans ce domaine.

Entrons dans le vif du sujet : notre premier code informatique

Le code informatique que nous allons étudier est extrait du projet SDR-1000 [SDR-1000].
Il s'agit d'une routine de génération d'un filtre numérique passe-bas.

Le filtrage numérique est le premier avantage de la radio logicielle auquel on pense. C'est en effet l'un des traitements les plus puissants, et c'est donc tout naturellement par celui-ci que nous commencerons notre série d'articles.



Passons donc en revue chacune des lignes de ce code :

■ Ligne 1 : Définition de la fonction et de ses paramètres
Cette ligne définit notre fonction de création du filtre passe bas numérique. Son nom est : *newFIR_Lowpass_REAL*.
Elle prend 3 paramètres :

● *Cutoff* : c'est la fréquence de coupure (cutoff en anglais) du filtre passe bas que l'on veut créer

● *Sr* : c'est la fréquence d'échantillonnage (Sampling Rate) du signal numérique qui entrera dans le filtre après sa création. Par exemple : 44 100 Hz pour un signal numérisé par une carte son

● *Size* : c'est la taille (en nombre de coefficients) du filtre que l'on veut créer. Plus un filtre a un nombre de coefficient important, et plus ce filtre aura de bonnes performances en réjection hors-bande et en raideur de la coupure

■ Lignes 2 et 3 : On sort de la fonction si ses paramètres d'appel sont invalides.
Lors du futur appel à la fonction de création de filtre, les 3 paramètres qui viennent d'être décrits ne pourront pas prendre n'importe quelles valeurs :

● *Cutoff* ne doit pas être négatif. En effet une fréquence de coupure négative n'aurait ici aucun sens

● Plus intéressant : la fréquence de coupure ne doit pas être supérieure à la moitié de la fréquence d'échantillonnage.
Cela résulte des fameuses conditions de Shannon et de Nyquist, qui précisent qu'en traitement numérique du signal, la fréquence d'échantillonnage doit être au moins deux fois plus grande que la plus grande fréquence à traiter.
Après avoir vérifié que les paramètres ne sont pas invalides, la fonction poursuit donc son travail.

■ Lignes 4, 5 et 6 : Quelques variables utilisées pour les calculs qui vont suivre
Ces trois lignes définissent 6 variables qui seront ensuite utilisées dans les calculs. Voici les plus importantes :

● *p* : il s'agit de la variable qui contiendra, à la fin des calculs, l'ensemble des informations caractérisant le filtre : son type (passe bas en l'occurrence), sa taille (*size* en l'occurrence), ses coefficients, etc. …

● *h* : il s'agit de la variable qui contiendra, à la fin des calculs, les coefficients du filtre. Il est d'usage dans les ouvrages théoriques de donner le nom " h " aux coefficients d'un filtre

● *w* : il s'agit d'une variable, utilisée en milieu de traitement, qui contiendra des coefficients de " pondération " ou de " fenêtrage ". Nous reviendrons sur ces coefficients un peu plus tard. " w " est l'initiale de Window (fenêtre en anglais).

● *Fc* : cette variable est initialisée à une valeur égale au rapport entre *cutoff* et *sr*. En effet, pour créer un filtre numérique, seul compte le rapport entre ces deux quantités, et non pas leurs valeurs respectives.
Par exemple, un filtre numérique passe-bas de fréquence de coupure *cutoff* = 1000 Hz traitant un flux échantillonné à 8000 Hz aura les mêmes coefficients qu'un filtre de fréquence de coupure *cutoff* = 2000 Hz traitant un flux échantillonné à 16000 Hz. *Fc* est parfois appelé " fréquence réduite "

■ Lignes 8 et 9 : Le filtre doit avoir un nombre de coefficients impair !
Le type de filtre construit ici doit posséder un nombre impair de coefficients. C'est ce que vérifient ces 2 lignes.
Si le paramètre *size* passé lors de l'appel à la fonction est impair, tant mieux, on ne fait rien. S'il est pair, on l'augmente d'un pour le rendre impair, et on poursuit !

Comme le nombre de coefficients est maintenant assurément impair, l'un d'entre eux se trouve exactement au milieu des autres (par exemple s'ils sont numérotés de 1 à 5, le coefficient du milieu est le troisième).

L'indice du milieu est stocké dans la variable *midpoint*. Un sort particulier sera en effet réservé au coefficient du milieu du filtre, comme nous allons le voir plus bas.

■ Lignes 10 à 13 : Réservation de l'espace mémoire
Ces quelques lignes de code permettent de réserver l'espace mémoire nécessaire au stockage des coefficients qui vont être calculés par la suite.

■ Lignes 15 à 20 : Enfin le calcul des coefficients du filtre !!
Nous avons ici une boucle allant de 1 à *size* (ligne 15), c'est à dire parcourant tous les coefficients de notre filtre. Cette boucle permet de calculer chaque coefficient.
La valeur de chaque coefficient *h* d'indice *j* est calculée ligne 18. Il s'agit d'une formule classique de calcul de filtre. Sans trop entrer dans la théorie, remarquons tout de même les points suivants :

● Comme leur nom l'indique, twopi et onepi ont pour valeur respectivement 2.π et π.
Ces valeurs sont définies dans une autre partie du code, qui n'a pas été reproduite ici pour alléger le texte

● Chaque coefficient est le produit d'un terme principal *sin ((twopi*(i-midpoint)*fc)/(onepi*(i-midpoint)))*, pondéré par une valeur : *w[j]*. Le terme principal correspond à la valeur des coefficients d'un filtre passe-bas parfait de coupure infiniment raide, et ayant un nombre de coefficients infini. Comme en pratique notre nombre de coefficients doit être fini, la pondération (ou fenêtrage) par *w* permet de calculer notre filtre " fini " à partir des coefficients du filtre " parfait ". Pour les initiés, le fenêtrage utilisé ici est du type Blackman-Harris, mais il en existe d'autres.

● Pour *fc* donné, la valeur du terme principal ne dépend que de la position de son indice par rapport à l'indice du milieu (i.e. de *i–midpoint*). Cette propriété de symétrie est illustrée sur le diagramme (voir page suivante) .
On peut montrer que cette propriété permet d'avoir un filtre à phase linéaire, c'est à dire ne déformant pas trop les signaux (notion de temps de propagation de groupe, utile pour certains types de modulation)

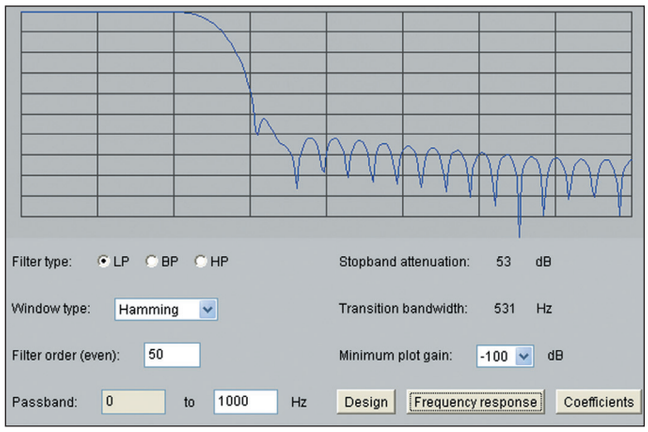
■ Enfin, ligne 20, nous constatons que le coefficient se situant au milieu du filtre a droit à un sort particulier, et n'est pas calculé avec la même formule que les autres (ligne 18). Cela provient du fait que la formule ligne 18 n'est pas calculable pour l'indice du milieu, sous peine de division par " zéro " (la division par zéro est impossible : essayez avec une calculatrice pour vous en persuader si nécessaire !).
Par " extension " de la formule générale de la ligne 18 (on parle aussi de " continuité " …), on peut montrer que la valeur à donner à l'indice du milieu est précisément 2.fc.
Tous nos coefficients sont ainsi calculés !!!!

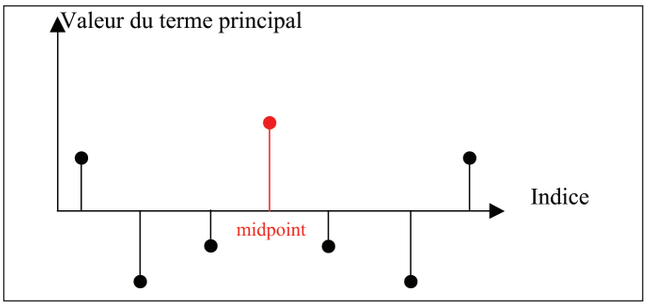
■ Lignes 23 à 25 :
La ligne 23 permet de libérer l'espace mémoire utilisé temporairement par les coefficients de fenêtrage, et dont on n'a maintenant plus besoin.
On n'oublie pas, ligne 24, de préciser que le filtre nouvellement créé est de type passe-bas (lowpass en anglais). Les lecteurs intéressés pourront trouver, dans le reste du code du projet PowerSDR, d'autres types de filtre : passe-haut, passe-bande, etc. …

Et on retourne, ligne 25, notre filtre à l'utilisateur qui en aura demandé la création, et qui s'en servira sans doute à bon escient !

Pour aller plus loin…

Pour conclure, et pour les lecteurs intéressés par un approfondissement de la théorie, nous citerons la référence [DSPGUIDE], qui contient des informations complémentaire sur le type de filtre que nous venons d'étudier (filtres à réponse impulsionnelle finie obtenus à partir du fenêtrage de réponses impulsionnelles idéales infinies).
Enfin, pour la pratique, nous citerons la référence [DSPTUTOR], qui contient une applet Java permettant d'obtenir les coefficients d'un filtre et sa réponse fréquentielle, à partir de l'entrée de ses paramètres. La copie d'écran ci-contre reproduit l'applet Java en question.

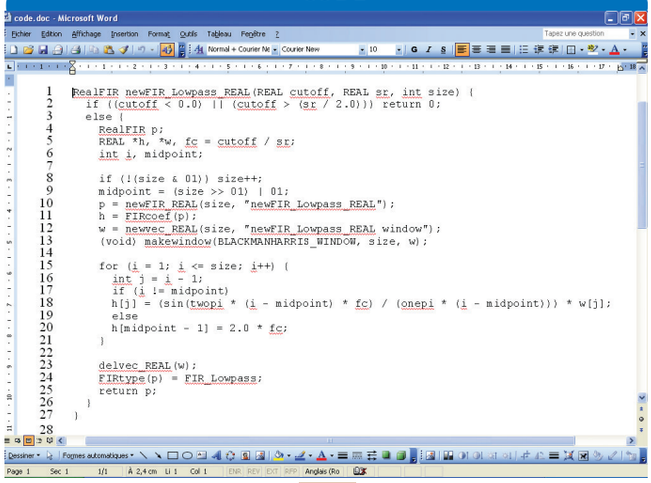




Références
■ [DSP-10] www.proaxis.com/~boblark/dsp10.htm
■ [DSPGUIDE] www.dspguide.com, chapitre 16
■ [DSPTUTOR] www.dsptutor.freeuk.com/FIRFilterDesign/FIRFilterDesign.html
■ [F1OAT] www.teaser.fr/~frible/
■ [GNURADIO] www.gnuradio.com
■ [KC1HR] "A simple SSB Receiver Using a Digital Down Converter", P. Anderson, QEX, mars 1994
■ [KD7O] "A high-performance Digital-Transceiver Design", J. Scarlett, QEX juillet/août 2002
■ [LINPSK] linpsk.sourceforge.net/
■ [RFSPACE] www.rfspace.com
■ [SDR-1000] www.flex-radio.com. L'extrait est issu du fichier filter.c du logiciel " PowerSDR " dont les sources sont téléchargeables sur ce site

D'autres références sont disponibles sur le site Internet de l'auteur : http://f4dan.free.fr.
Merci à Rémy F6ABJ, pour sa contribution et ses commentaires éclairés.

CODE



une seconde d'arc

La société Heidenhain présente des capteurs rotatifs incrémentaux dont l'électronique de traitement du signal offre une résolution de 27 bits soit 134 millions de positions par tour en valeur absolue, même lors de grandes vitesses de rotation, c'est-à-dire plus précis qu' une seconde d'arc.

quartz horloger

Plusieurs constructeurs comme le Suisse Micro Crystal, le Japonais Epson et l'Américain Fox Electronics proposent un quartz horloger dans un boîtier d'une épaisseur inférieure au millimètre. Bien souvent il est encapsulé dans un boîtier CMS.

canon électrique

Un projectile, actuellement en développement, pourrait bien remplacer l'arme traditionnelle. En exploitant l'énergie électrique, le canon dit électrique ou électromagnétique a l'avantage de ne plus être limité par la vitesse de combustion de la poudre, et permet de dépasser la vitesse de propulsion actuelle, qui est d'environ 1800 m/s. En effet, dans le cas du canon électrique, la vitesse de propulsion est proportionnelle à l'intensité du champ électromagnétique généré. Lors d'une expérimentation sous vide, une équipe russe a atteint les 10 000m/s. Avec une intensité de l'ordre du million d'ampères, les militaires tablent sur une vitesse de propulsion de 2400m/s. Quand on sait que l'énergie de perforation du projectile est proportionnelle au carré de sa vitesse, on imagine les capacités de destruction de ces armes. Pour le système d'alimentation, des bancs de condensateurs sont utilisés pour libérer des intensités de l'ordre du million d'ampères dans des intervalles très courts. Ils occupent actuellement une salle entière.

diode laser

Sanyo vient de compléter sa gamme de diodes laser bleu/UV avec un modèle émettant une puissance optique de 50 mW en continu (100 mW en régime impulsionnel) à 405 nm. Compatible avec l'enregistrement de données sur disques "Blue-ray" à double couche dont la capacité de stockage peut atteindre 54Go, cette diode laser, encapsulée dans un boitier de 5,6 mm, est la plus puissante actuellement dans sa catégorie.