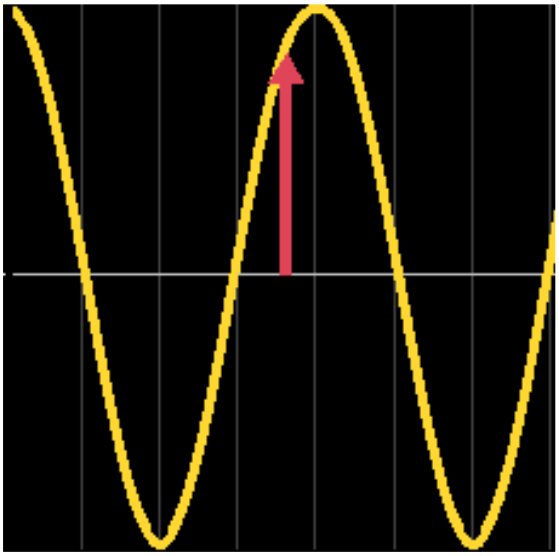
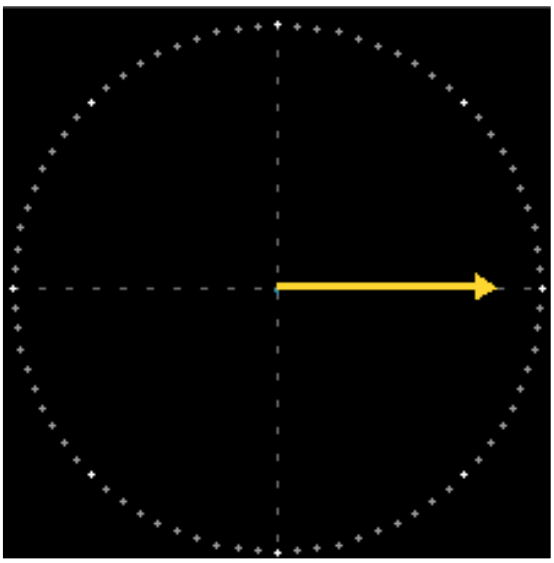
Il Ruolo della S

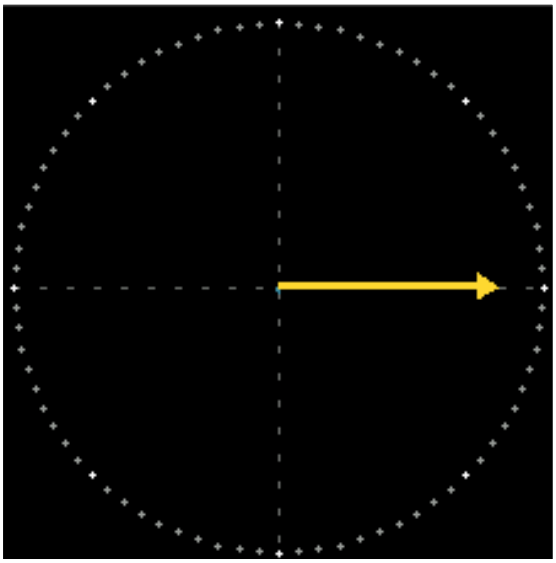
Alberto di Bene I2PHD

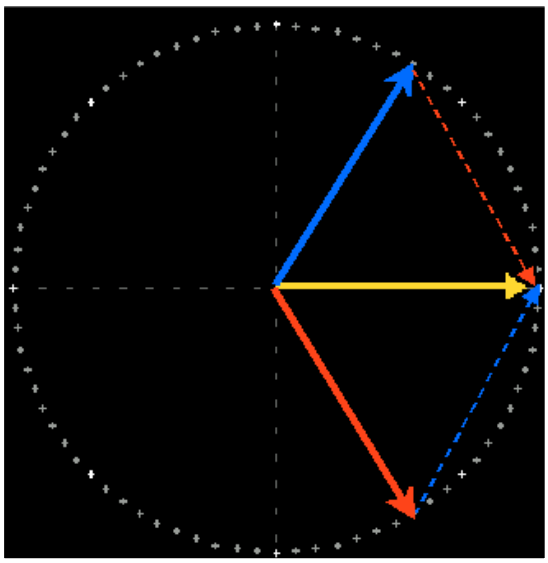
Ovviamente la S e' la la iniziale della sigla SDR (Software Defined Radio), e con questa frase si intende concentrare la attenzione sui compiti assegnati al software in una applicazione di questo tipo. L'hardware di una SDR generalmente produce una coppia di segnali, detti I e Q, che il software usa per ricavarne la modulazione originariamente impressa, non importa se AM, FM, SBB od altro.
Per capire bene perche' si usino due segnali in quadratura di fase, facciamo un rapido excursus sul processo di mixing, quindi di conversione in banda (quasi) base del segnale ricevuto.
Il segnale di partenza e' (piu' o meno) una sinusoide, quindi avente esclusivamente una componente reale, come tutti i segnali attinenti alla realtà fisica. Pero' nulla vieta di darne una descrizione matematica di questo tipo :





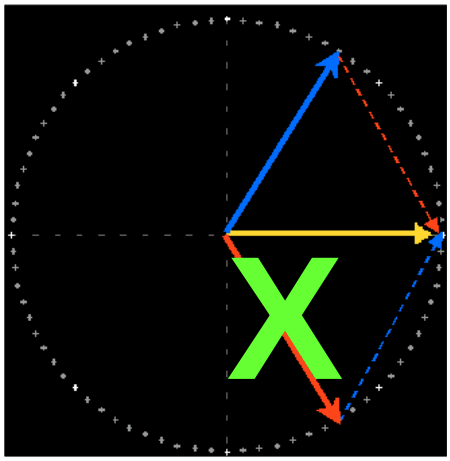
L'ampiezza istantanea del segnale e' rappresentata da un fasore che scorre avanti e indietro sull'asse reale, dal massimo positivo a zero al massimo negativo e viceversa. A sua volta questo fasore reale lo si puo' pensare come somma vettoriale di due fasori complessi, ruotanti in ciascuno in senso opposto all'altro



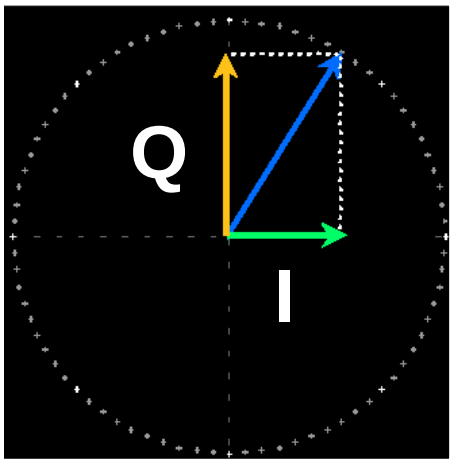


Suggestivamente di potrebbe dire che un segnale reale e' composto da una componente cha avanza nel tempo, e da una retrograda nel tempo… comunque sia, questa componente negativa e' quella che provoca alcuni fastidi nei processi di conversione, e precisamente la comparsa delle frequenze immagine. Infatti un mixer altro non e' che un moltiplicatore, e se facciamo la moltiplicazione di due segnali reali, quindi di quattro fasori complessi, ricordandoci che nel campo complesso la moltiplicazione ha come risultato un numero il cui modulo e' il prodotto dei moduli dei fattori, e la cui anomalia (angolo) e' la somma delle anomalie dei fattori, si vede che il prodotto dei quattro fasori ha come risultato una coppia (positiva e negativa) di fasori corrispondenti alla somma delle frequenze (abbiamo sommato le anomalie, e quindi la loro variazione nel tempo…), ed un'altra coppia di fasori che corrispondono alla parte positiva e negativa di un segnale di frequenza pari alla differenza delle frequenze (abbiamo sommato una anomalia che si incrementa nel tempo, con una che invece si decrementa…) .

Balza immediatamente alla mente che il problema della generazione dell'immagine (sia essa lo somma o la differenza) si potrebbe risolvere eliminando uno dei due fasori che descrivono il segnale. A questo punto il risultato di un processo di mixing sarebbe solo il segnale desiderato (somma o differenza).



Se elimino ad esempio il fasore retrogrado nel tempo evito il problema della frequenza immagine, pero' il fasore rimanente non e' piu' la rappresentazione di un solo segnale reale, come lo era prima… ora questo fasore rappresenta una coppia di segnali, uno reale e l'altro sull'asse immaginario, quindi con sfasamento temporale di 90 gradi rispetto al primo.



Convenzionalmente questi due segnali vengono detti I (In fase) e Q (in Quadratura di fase). Tirando le fila di tutti questi discorsi, siamo arrivati alla conclusione che per avere un mixer che non produca una frequenza immagine, i segnali non basta piu' che siano rappresentati da una singola componente reale, ma devono essere descritti da due componenti ortogonali, uguali in modulo, ma sfalsate di 90 gradi una rispetto all'altra.

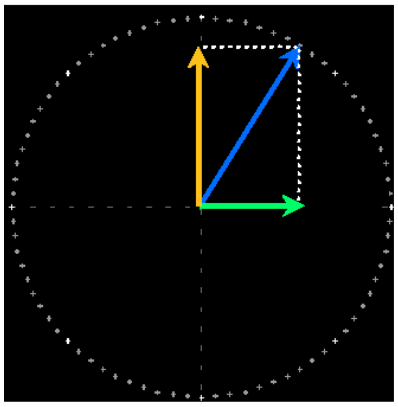
Ovviamente anche il risultato della conversione, che ora sara' scevro dalla sua immagine, sara' un segnale cosiddetto analitico, cioe' descritto anche lui da una coppia I/Q. Ed e' questa coppia che il software deve essere in grado di gestire.

Il processo di demodulazione

La coppia I e Q descrive compiutamente il segnale risultato della conversione, compresa una eventuale modulazione ad esso impressa. Vediamo come si opera per ricavare il segnale modulante partendo da I e Q

AM

La modulazione di ampiezza consiste nella variazione nel tempo del modulo del fasore ruotante descritto da I e Q. Per ricavare il segnale modulante si applica semplicemente il teorema di Pitagora…

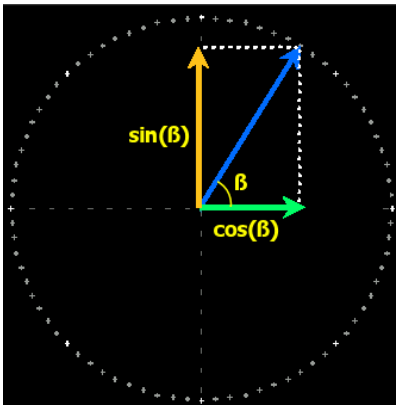


$$A = (I^2 + Q^2)^{0.5}$$

Ovviamente sia A che I e Q sono funzioni del tempo. Il risultato dell'operazione e' un valore costante, che e' l'ampiezza della portante, usando termini radio, piu' uno scostamento variabile che e' il segnale modulante. Basta quindi applicare un filtro passa alto per eliminare il termine costante, ed il risultato e' il segnale modulante desiderato.

FM

La modulazione di frequenza, vista nell'ottica in cui ci siamo messi finora, e' una perturbazione della velocita' di rotazione del fasore, che quindi percorrera' angoli diversi nelle stesse unita' di tempo, in funzione del segnale modulante.



$$ß = \arctan(Q/I)$$

Anche qui **ß**, I e Q sono funzione del tempo. **ß** rappresenta la anomalia istantanea, che si incrementa nel tempo con velocita' pari alla frequenza della portante, velocita' che pero' e' perturbata dal segnale modulante. Posso calcolare **ß** con l'uso dell'arcotangente, pero' poi devo fare una derivata prima rispetto al tempo per ricavare la frequenza, e dalle variazioni di questa il segnale modulante. Anche qui si impone l'uso di un filtro passa alto per eliminare il termine costante.

SSB

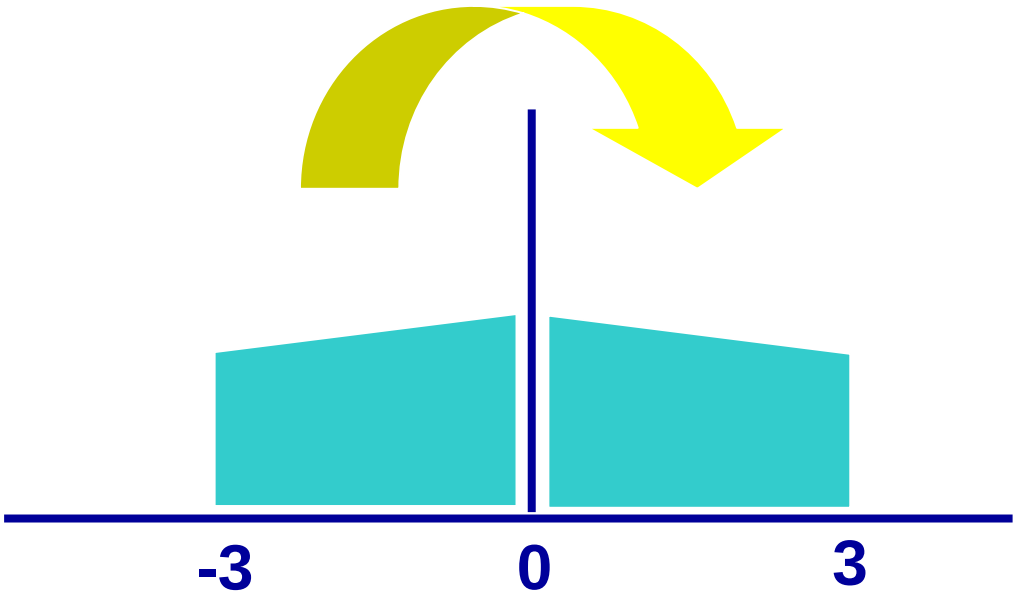
Parlare di modulazione e demodulazione nel caso di un segnale SSB e' improprio, in quanto un segnale SSB altro non e' che il segnale fonico di bassa frequenza semplicemente traslato nello spettro sino alla frequenza RF di trasmissione.
Quindi per riottenere il segnale originario occorre solo ritraslare in banda base il segmento RF opportuno. Questo non presenta difficoltà, perche' all'interno del software si possono usare esattamente le stesse tecniche di conversione in quadratura di fase viste in precedenza, ed anzi, con il vantaggio che operando ora nel campo numerico non ci sono le imprecisioni e le tolleranze dei componenti analogici che possono far si' che, ad esempio, l'angolo tra I e Q non sia di 90 gradi esatto, oppure che le loro ampiezze siano leggermente diverse.

Quindi quello che otteniamo dalla conversione operata nel software (che tra parentesi fa uso di un NCO, ovvero di un Numerically Controlled Oscillator) e' un segnale in banda base, pero' non in formato reale, bensi' analitico, cioe' descritto dagli ormai nostri amici fasori I e Q.

Bisogna trasformarlo in un segnale reale da inviare al DAC, che non capisce i numeri complessi…

Una tecnica molto usata e' quella di usare un trasformatore di Hilbert, che, detto con altri termini, altro non e' che un filtro all-pass, cioe' che lascia passare tutte le frequenze senza variazioni di ampiezza, e che pero' opera uno sfasamento di 90 gradi costanti. Se consideriamo che Q e I hanno una differenza di fase di +90 oppure di -90 gradi a seconda che stiamo considerando la banda superiore oppure quella inferiore rispetto alla frequenza zero, ecco che se ritardiamo Q di 90 gradi e poi in alternativa andiamo a sommarlo oppure a sottrarlo ad I, avremo eliminato la parte superiore oppure quella inferiore dello spettro, ovvero la frequenza immagine.

Questa, come detto, e' una tecnica molto usata, e funziona anche… pero' in Winrad ho preferito usare un metodo concettualmente equivalente, ma diverso, e, a mio parere, forse piu' elegante. Io chiamo questo metodo "specchiamento coniugato in frequenza".
Tutti coloro che in vita loro hanno calcolato una trasformata di Fourier di un segnale sanno che il risultato e' composto da due meta', simmetriche una rispetto all'altra, con la seconda parte uguale al complesso coniugato della prima. E alla luce di quanto visto finora, cioe' della descrizione di un segnale con due fasori controrotanti, questo risulta anche ovvio. Ogni meta' e' la descrizione nel dominio della frequenza di ciascuno dei due fasori. Se ritrasliamo in banda base il segnale SSB, spostandone i coefficienti di Fourier a partire da frequenza zero, se ci azzardassimo a fare una trasformata inversa complessa, otterremmo un segnale nel dominio del tempo di carattere complesso, inadatto per essere inviato ad un DAC. Se la trasformata invece fosse reale, prenderemmo anche la parte di spettro nel campo negativo, mescolando in modo irreversibile la banda voluta e quella "pseudo" immagine, che conterra' informazione non desiderata oppure noise.
Per evitare questo, e' sufficiente sintetizzare uno spettro in frequenza cosi' come se lo aspetta la IFT di tipo reale. Cioe' prendiamo il segnale voluto, lo ribaltiamo rispetto alla frequenza zero e ne calcoliamo il complesso coniugato. Se facciamo ora una IFT reale, ecco che otteniamo un bel segnale reale corrispondente all'andamento nel tempo dello spettro dell'informazione desiderata.



Architettura di massima di Winrad

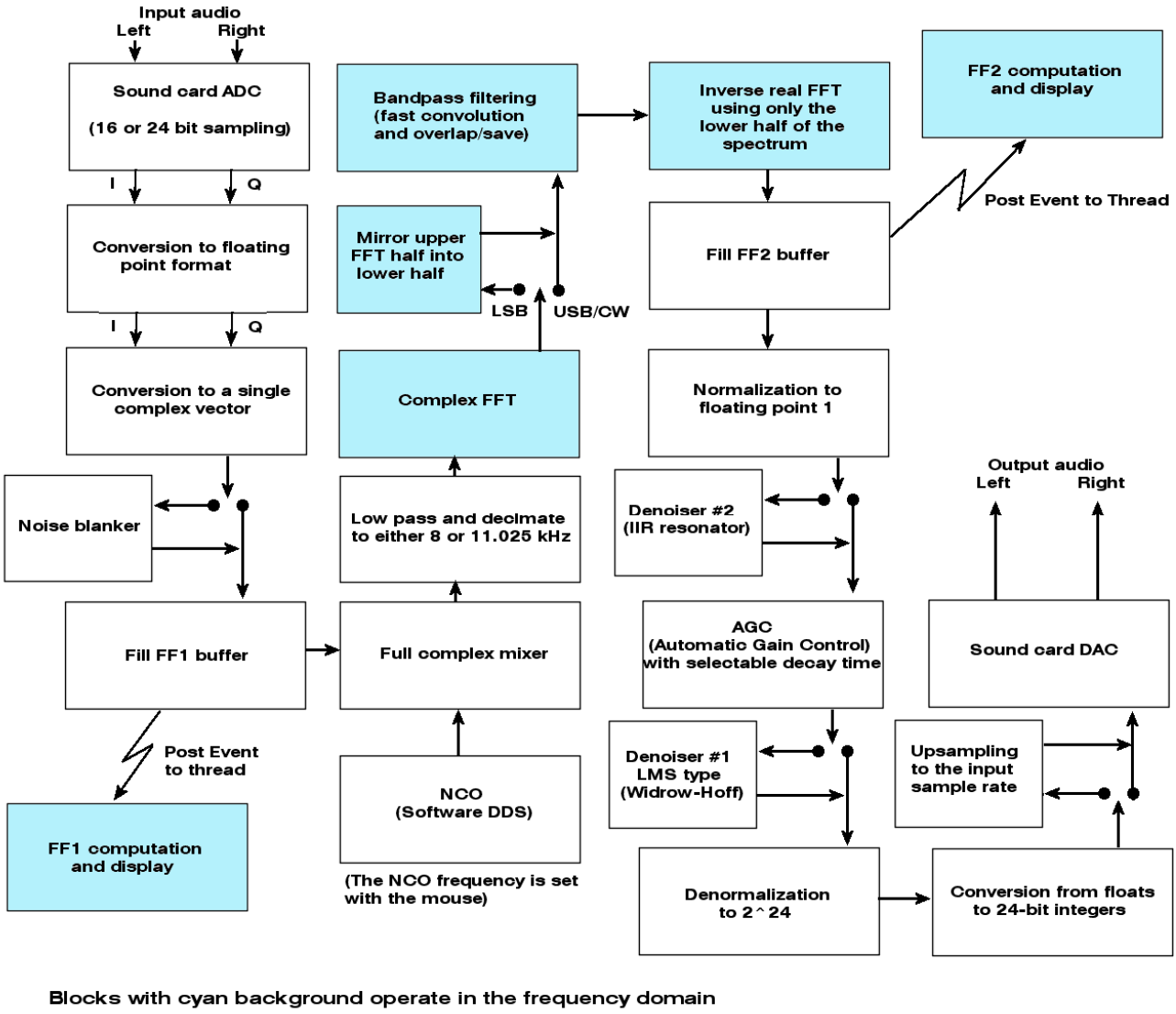


Figura 1 - Architettura di massima

Nella figura 1 e' rappresentata la architettura del thread di processing audio, che e' separato dal thread di gestione interfaccia utente e dai due threads di calcolo e display delle due FFT (Fast Fourier Transform) con relativi waterfall. Nel seguito daro' qualche breve nota sul funzionamento dei blocchi di questo thread, gli altri threads non sono particolarmente interessanti da un punto di vista concettuale.

La digitalizzazione del segnale viene effettuata dalla scheda audio, con la scelta del subsystem da usare, WMME a 16 bit oppure ASIO a 24 bit, sempreche' la scheda audio presente nel PC abbia installati dei drivers ASIO (non tutte ne sono fornite). Subito dopo la digitalizzazione si converte immediatamente ad un formato floating point, che sara' usato nel resto del thread, fino al momento di restituire il segnale alla scheda audio per la riproduzione.

I segnali I e Q rispettivamente presenti sul canale sinistro e destro vengono conglobati in un unico vettore complesso. A questo punto si effettua opzionalmente una fase di noise blanking, in cui si individuano e si eliminano gli spikes nel segnale che con grossa probabilita' rappresentano solo dei disturbi. La soglia di intervento del noise blanker e' regolabile, e questo blanker da solo puo' giustificare il perche' questo programma sia dichiarato utile per il traffico EME. Il segnale cosi' ripulito viene accodato in buffer per il calcolo e il display della prima FFT nel pannello superiore dell'interfaccia utente. Questo permette non solo di visualizzare lo spettro del segnale in arrivo, ma

anche di fare un point-and-click per impostare l'oscillatore locale software (NCO) di conversione in banda base

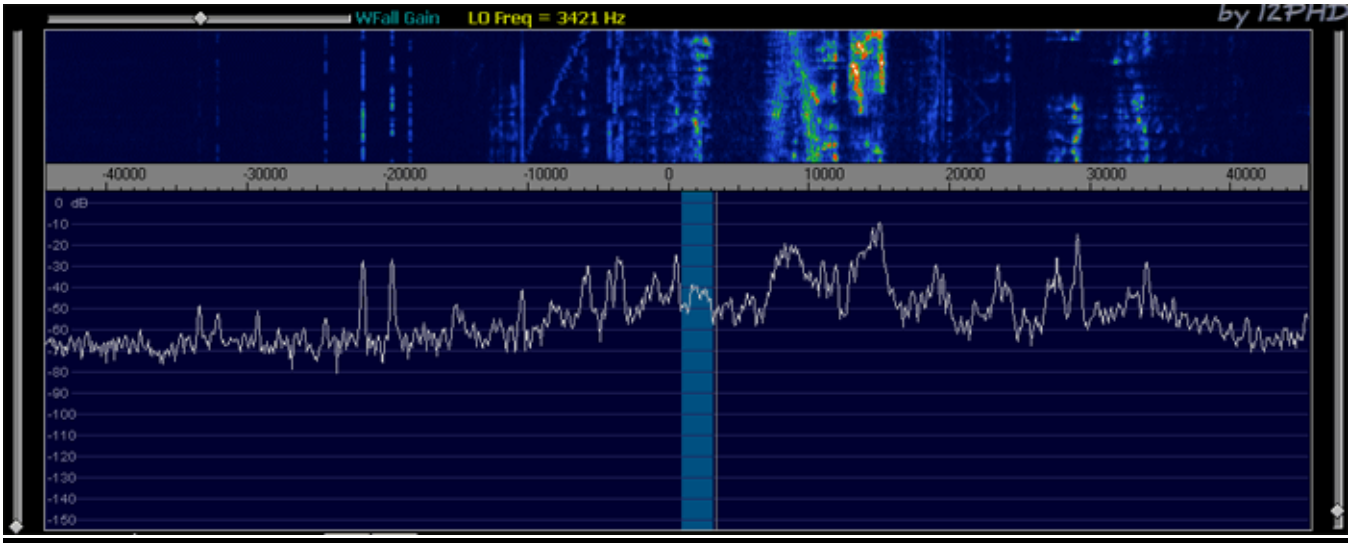


Figura 2 - Il display della FF1

Nella figura 2 e' possibile vedere una fetta di 96 kHz della banda dei 40 metri ricevuta con un ricevitore a conversione diretta con mixer di Tayloe. E' stato sintonizzato un QSO e la finestrella azzurro chiaro rappresenta la banda passante correntemente impostata, circa 2.4 Khz. Il modo di ricezione e' LSB, come evidenziato dal fatto che la finestrella azzurra e' al di sotto della riga rossa (nella figura sembra bianca...) che indica la frequenza virtuale della portante soppressa. La figura, per esigenze di riduzione tipografica, non e' chiarissima, ma il vero display lo e'.

La finestrella azzurra e' spostabile con il mouse, permettendo la ricezione del segmento voluto della banda dei 40 metri, sia in SSB che in CW.

Proseguendo con la analisi del percorso del segnale nel thread di processing audio, siamo ora arrivati al full complex mixer. Il nome "full complex" indica che accetta un segnale complesso sia nella porta RF che nella porta LO. Il risultato e' che si ha una conversione senza produzione di frequenza immagine, in quanto entrambi i segnali sono di tipo analitico, quindi senza componente specchiata sull'asse delle frequenze. Come detto precedentemente, la frequenza dell'oscillatore locale viene impostata con il mouse, in funzione del segmento che si vuole ricevere. Il segnale LO e' generato da un NCO (Numerically Controlled Oscillator), cioe' dall'equivalente software di un DDS. Visto che qui non si hanno i limiti imposti dall'hardware, questo NCO ha una sine table piuttosto lunga, esattamente 262144 entries, in modo da minimizzare i prodotti spurii.

Il risultato della conversione e' il segnale desiderato in banda audio. Quindi puo' essere conveniente, nel caso la campionatura iniziale fatta con la scheda audio fosse piuttosto alta, fare una decimazione in modo da portare il sampling rate ad un valore inferiore, compatibile con la banda audio da riprodurre. Per minimizzare gli errori di arrotondamento, e grazie all'osservazione che tutte le frequenze di campionamento delle schede audio sono multipli di 8 oppure 11.025 kHz, questi due valori sono stati scelti come valori target per il downsampling (decimazione), in funzione del valore iniziale di campionamento. Ovviamente la decimazione e' preceduta da un filtro FIR passabasso per rispettare Nyquist, che si arrabbia se un segnale campionato a frequenza F contiene delle componenti spettrali di frequenza superiore a F/2 (nota per i puristi : so benissimo che questa affermazione non e' esattissima, e che esistono le Nyquist zones, ma qui non stiamo facendo un corso di Signal Processing...).

Arrivati a questo punto abbiamo il segnale desiderato, che, nel caso della SSB, ancora contiene la banda superiore e quella inferiore rispetto alla frequenza zero, che e' quella di sintonia. Cosi' come gia' descritto nella prima parte di questo documento non useremo il metodo a sfasamento di Hilbert, ma lo specchiamento coniugato.

Quello che facciamo e' calcolare una FFT complessa del segnale. Il risultato di questa FFT e' uno spettro che, a differenza di quello ottenuto da una FFT di un segnale reale, non e' coniugato simmetrico nella sua seconda meta' rispetto alla prima meta'. Invece quello che abbiamo e' che la prima meta' e' lo spettro della parte positiva delle frequenze (ricordiamoci che ora stiamo lavorando a IF zero), mentre la seconda meta' e' lo spettro della parte negativa delle frequenze, e cioe', in altre parole, della banda LSB rispetto al carrier virtuale a zero Hz. Capito questo, il resto viene di conseguenza. Se vogliamo la LSB, altro non facciamo che prendere la seconda meta' dello spettro FFT, ne calcoliamo il coniugato simmetrico, lo specchiamo, e rimpiazziamo la prima meta' dello spettro con il risultato. Se invece vogliamo la USB, semplicemente saltiamo questa ultima operazione. Ora compiamo un piccolo imbroglio. Facciamo finta che lo spettro cosi' ottenuto sia il risultato di una FFT reale, non complessa, e ne calcoliamo la FFT inversa reale. Per magia, otteniamo un segnale audio reale, che e' quello che volevamo. Abbiamo quindi demodulato la USB oppure la LSB senza scomodare Hilbert. Provare per credere.

Nella descrizione del paragrafo precedente e' stato anticipato un passaggio, e cioe' il filtraggo passabanda. Visto che abbiamo il segnale nel dominio della frequenza, viene spontanea l'idea di usare il metodo della fast convolution, suggerito, come detto, anche da considerazioni di perfomance. In pratica si calcola il kernel del FIR che rappresenta il filtro voluto, lo si smussa agli estremi con una finestra adeguata (in Winrad e' stata usata una finestra di Blackman ottimizzata), se ne calcola la FFT, si moltiplica questa FFT per la FFT del segnale, e poi si applica l'algoritmo overlap/save per trasformare la convoluzione circolare in una lineare. Nel caso specifico il kernel del FIR viene calcolato on-the-fly in funzione dell'impostazione fatta dall'utente dei limiti del passabanda, simulando un numero di prese (taps) del FIR molto alto, e cioe' 1537. Questo numero sarebbe assolutamente inaccettabile per una implementazione del filtro nel dominio del tempo, ci metterebbe una eternita' come tempo di esecuzione. Usando invece la fast convolution il tempo derivante dalla lunghezza lo si paga una sola volta, cioe' al momento del calcolo del kernel. Questo numero equivalente di prese cosi' alto permette di avere una reiezione fuori banda maggiore di 160 dB, assolutamente sufficiente per ogni esigenza.

Abbiamo quindi riottenuto un segnale audio nel dominio del tempo, di formato reale. Potremmo gia' inviarlo alla scheda audio per la riproduzione. Ma abbiamo ancora qualcosetta da fare. Innanzi tutto prepariamo un buffer per seconda FFT, quella che usiamo per il display nel pannello inferiore. Ogni volta che il buffer e' pieno, segnaliamo un Event al thread della FF2. che asincronicamente fara' il suo lavoro. Una piccola nota : sia il thread della FF1 che quello della FF2 hanno una priorita' di esecuzione inferiore al thread audio. In questa applicazione, ascoltare e' piu' importante che vedere!

Quello che dobbiamo ancora fare e' l'applicazione di una funzione di denoising al segnale, selezionabile tra due tipi diversi, ed un AGC in modo da mantenere il livello di ascolto relativamente costante. Infatti l'uso principale di questo programma, come gia' detto, e' dopo un ricevitore a conversione diretta che normalmente non ha AGC a livello hardware.

Per motivi che qui sarebbe un po' lungo spiegare, il segnale viene normalizzato al valore floating point 1. Cioe' un segnale che corrisponde alla saturazione del DAC della scheda audio viene scalato al livello 1. A questo punto si applica, opzionalmente mediante una scelta dell'utente, il denoiser di tipo 2, che e' una combinazione di una amplificazione non lineare con un risuonatore IIR, e che e' inseribile solo se il modo di ricezione e' il CW. Vediamo di spiegare meglio. Si analizza il segnale audio e si ricava il valore medio del noise, facendolo arbitrariamente corrispondere al quartile inferiore delle magnitudini misurate nello spettro. Si misura inoltre il valore di picco delle magnitudini dell'intervallo di tempo considerato. Abbiamo ora i dati per calcolare un coefficiente di amplificazione del segnale che dipende dalla distanza di ogni singola magnitudine dal valore del noise. In pratica, il picco avra' coefficiente di amplificazione 1, e questo coefficiente sara' tanto piu' basso quanto piu' lontano dal picco sara' il segnale, fino ad arrivare a zero per il livello stimato del noise. L'effetto netto e' quello di lasciare inalterati i picchi, che corrispondono al segnale utile, e di abbattere il noise di fondo. E' ovvio che questo metodo funziona bene solo per rumore pseudo gaussiano, non certo per rumore impulsivo. Pero' all'inizio del percorso del segnale all'interno del programma abbiamo gia' incontrato il noise blanker che ha gia' ripulito il segnale da tutti gli impulsi spurii. Questa descritta e' la prima meta' del denoiser di tipo 2. La seconda meta' si basa sul fatto che il pannello della FF2 permette all'utente di specificare qual e' il pitch CW gradito all'utente stesso. Con questa informazione calcoliamo i coefficienti di un filtro IIR a due zeri e due poli. I due zeri vengono semplicemente piazzati nell'origine, mentre i due poli (uno il coniugato simmetrico dell'altro) vengono posti appena leggermente (per evitare auto-oscillazioni) all'interno del circolo

unitario sul piano di Argand-Gauss della trasformata Z. Se proprio vogliamo fare un paragone con il campo analogico, possiamo dire che questo filtro si comporta come un risuonatore con un Q variable. La frequenza di accordo del risuonatore e' quella impostata mediante la scelta del CW pitch, mentre il fattore Q lo si puo' scegliere facendo comparire, mediante il tasto destro del mouse sul bottone corrispondente, uno slider di regolazione. L'effetto auditivo e' esattamente quello che si avrebbe con un circuito risuonatore hardware. Se si sceglie un Q troppo alto (che in questo caso significa che i due poli si avvicinano molto al circolo unitario) il filtro e' piu' selettivo, con pero' il pericolo di ringing, che puo' arrivare fino al punto di rendere i punti e linee del CW indistinguibili tra di loro, perche' la stessa frequenza di manipolazione CW e' al di fuori della banda passante del filtro !

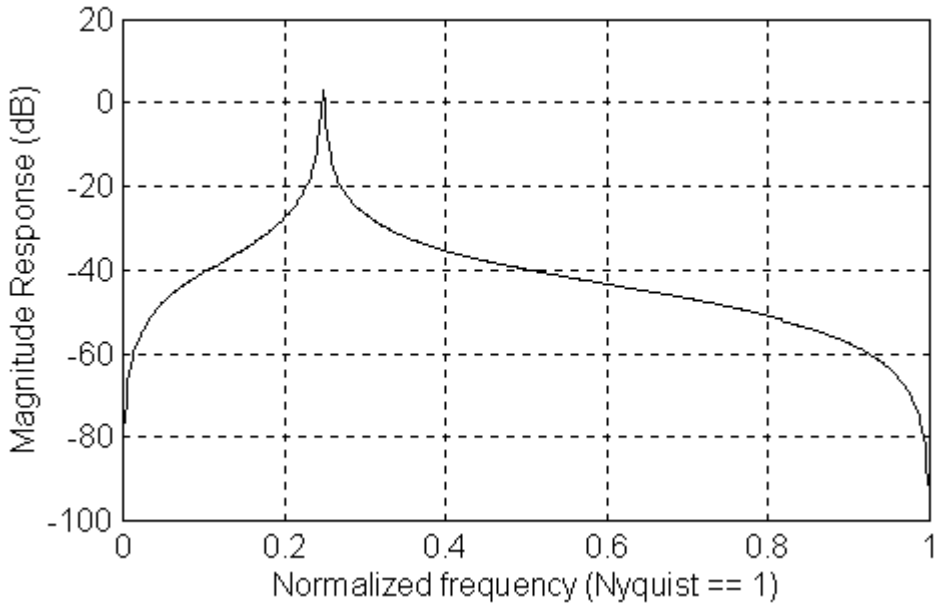


Figura 3 - Risposta in frequenza del risuonatore IIR

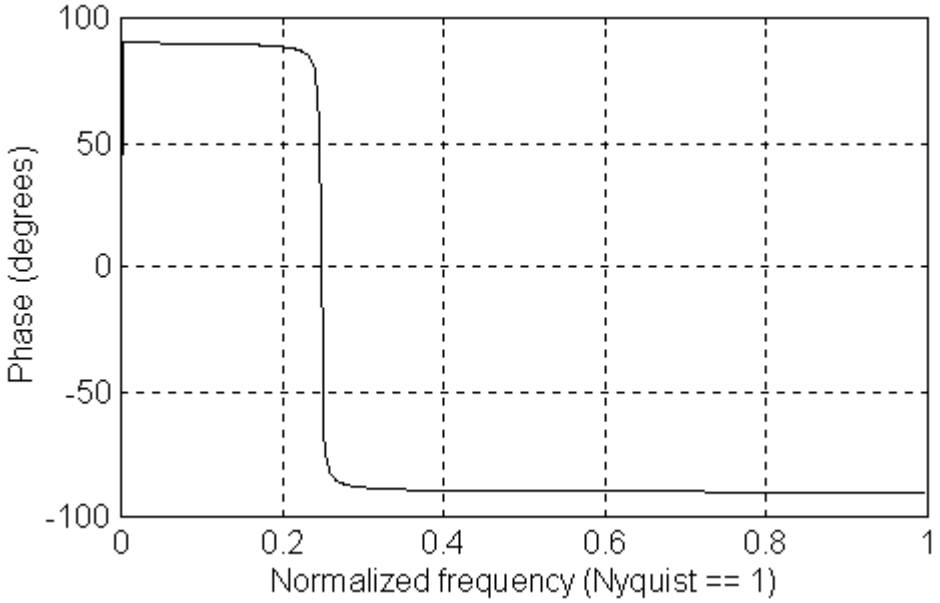


Figura 4 - Risposta in fase del risuonatore IIR

La figura 4 mostra la risposta in frequenza e la figura 5 quella in fase del risuonatore, simulate mediante Matlab, per un fattore Q medio. Aumentando questo fattore, il risultato e' che il picco diviene piu' stretto, piu' aguzzo.

Dopo il denoiser di tipo 2 incontriamo la routine AGC, che si preoccupa di rendere piu' o meno costante il livello del segnale. Il tempo di attacco e' bassissimo (sto addirittura valutando se porlo a zero), mentre il tempo di rilascio e' attualmente selezionabile tra due valori, che nel pannello

principale di Winrad sono indicati con Fast e Slow. Solo l'esperienza di uso dira' se e' necessario aggiungere altre velocita' di rilascio e quali.

Dopo l'AGC troviamo il denoiser di tipo 1, che altro non e' che l'arcinoto algoritmo LMS, con aggiustamento dinamico dei coefficienti h secondo Widrow-Hoff. In pratica l'algoritmo privilegia i segnali che hanno una correlazione temporale alta, mentre attenua gli altri. In termini tecnici si tratta di un FIR di lunghezza 64 prese, con una linea di ritardo (quella che determina la correlazione temporale) di lunghezza variabile a seconda che il modo di ricezione sia il CW oppure la SSB. Infatti nel caso della voce l'intervallo di correlazione e' parecchio inferiore a quello del CW.

Bene, il segnale e' arrivato quasi al termine del suo lungo viaggio all'interno del programma. Non resta che denormalizzarlo, cioe' lo riportiamo ad un fondo scala di 2^24, e poi lo convertiamo da formato floating point a formato intero. Siamo pronti a mandarlo alla scheda audio. Pero' c'e' un ultimo passo da fare. Il programma consente di usare schede audio separate per input e per output. Se la scheda output e' diversa da quella input, viene impostata per una frequenza di campionamento di playback pari a 8 oppure 11.025 kHz a seconda di come avevamo fatto la decimazione. Se invece usiamo la stessa scheda, quasi tutte le schede, eccetto quelle di fascia alta, professionale, non consentono frequenze di campionamento diverse tra ingresso e uscita. Se usiamo Windows XP ci viene incontro lo stesso sistema operativo, che virtualizzando la scheda audio, rende questo problema trasparente. Pero' il programma deve funzionare anche con Windows 98, ed inoltre il ricampionamento fatto da Windows XP a volte lascia qualcosa a desiderare. Quindi l'unica soluzione e' che il programma stesso effettui un up-sampling (quando necessario) per far si' che la frequenza di campionamento in uscita sia la stessa impostata in ingresso.

Ora abbiamo veramente finito. Il segnale viene inviato contemporaneamente al canale destro e sinistro della scheda audio per la riproduzione.

Come anticipato all'inizio, l'uso di Winrad rende possibile l'ascolto e la decodifica di segnali CW in condizioni di noise ambientale man-made tali da rendere problematico addirittura il riconoscimento della presenza di un segnale CW con i filtri esclusi. Questa e' una situazione ben nota a chi fa traffico EME !

Buon ascolto con Winrad !

Alberto di Bene, I2PHD

Il programma e' liberamente disponibile sul sito :

http://www.weaksignals.com