di Mauro Olivero Pistoletto IK1IMG

Interfaccia Echolink di terza generazione

Una scheda "intelligente" in grado di prendere "decisioni"

1ª parte

Introduzione

Come gli appassionati d'Echolink sanno, le possibilità di accedere al sistema sono di due tipi: via PC oppure via radio attraverso un link o un ripetitore; nel secondo caso è necessario eseguire un collegamento fisico tra PC e radio. Tale operazione permette di far transitare i segnali BF fra PC e RTX (e viceversa) oltre a fornire alla radio il comando del PTT (trasmissione radio in corso) e ricevere eventualmente quello di COR (ricezione radio in corso). Anche qui vi sono due strade: collegamento diretto o tramite interfaccia separatrice. La prima opzione è molto rischiosa perché può portare a malfunzionamenti (a causa di deleteri anelli di massa) oppure a danneggiamenti del PC o viceversa (in caso di differenze di potenziale). La cosa non è da prendere sottogamba, qualche anno fa anch'io ne sono stato vittima. Avevo il PC collegato tramite porta seriale ad una scheda RF ed è bruciato l'integrato PLL pilotato direttamente da tale linea. Quindi non si tratta di leggende come mi è già capitato di udire in radio: i componenti vanno "arrosto" davvero. Ecco quindi la necessità di una scheda separatrice in modo che vi sia il libero scambio dei segnali senza collegamento fisico. Vediamo in fig. 1 i blocchi essenziali di tale interfaccia che potremmo definire di "**prima generazione**".

Interfaccia di prima generazione

I segnali da separare sono quattro: due di BF (RX BF e TX BF da e per la sound blaster) e due di controllo (PTT e COR da e per la porta seriale). Per quelli del primo gruppo si adottano generalmente semplici trasformatori separatori di segnale con rapporto 1:1 impedenza 600 Ω. Nel secondo caso, invece, il lavoro è svolto da fotoaccoppiatori e da un annesso circuito che adatta i livelli a quelli richiesti dalla porta RS232 (ed oggi è spesso seguito anche da un adattatore seriale-USB per i PC sprovvisti di questa porta). Le linee del segnale hanno generalmente dei trimmer per la regolazione delle ampiezze (V1 e V2) e la BF proveniente dall'RTX fluisce su una resistenza di carico (RC). Un jumper (JPTX) permette di chiudere a massa, tramite una resistenza, il circuito BF del microfono al fine di comandare la trasmissione sugli apparati che richiedono questa modalità. Si noti che la parte sinistra "RTX" (gialla) è **fisicamente** separata da quella destra "PC" (azzurra) sia nella parte segnali sia in quella masse (a tale scopo queste ultime sono state disegnate diversamente per evidenziare tale caratteristica). Questa interfaccia, non avendo connotazioni particolari, può essere utilizzata anche per altri scopi (Packet, PSK31 ecc.). Esistono varianti totalmente optoisolate (linee di controllo e BF) che hanno però il non indifferente svantaggio di richiedere una doppia alimentazione senza fornire alcun che di migliorativo. Chi usa il sistema Echolink sa anche che la gestione radio avviene tramite bitoni DTMF e che la loro corretta decodifica è essenziale. Quella software del programma fa quello che può tanto è vero che sono nate da subito delle interfacce che trasferiscono in hardware tale necessità. Questo fa anche risparmiare al PC risorse non essendo impegnato in quest'operazione. Ecco quindi la nascita d'interfacce che definirei di "**seconda generazione**".

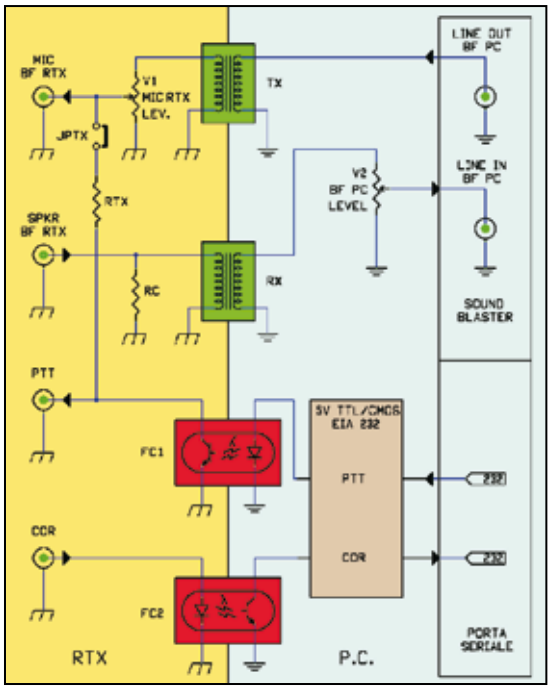

Fig. 1

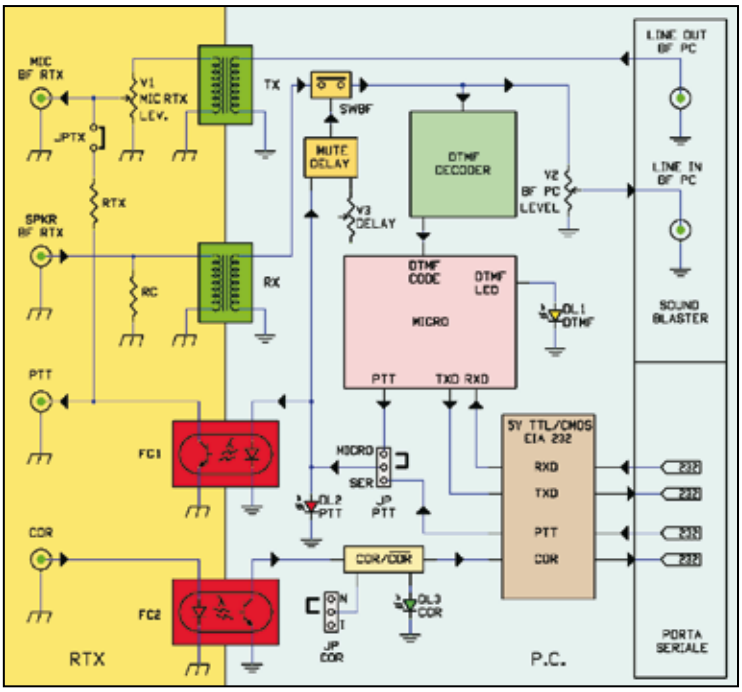

Fig. 2

Interfaccia di seconda generazione

Osserviamo la fig. 2. Oltre ai blocchi già esaminati in precedenza, in essa si vede chiaramente la nascita del circuito di decodifica DTMF, compito affidato usualmente ad un integrato tipo MT 8870 oppure equivalente. Tale componente è un DTMF decoder che fornisce un codice binario a 4 bit in base al bitono ricevuto e, tramite un'apposita linea, informa della ricezione in corso. Compare anche un microcontrollore, di solito un PIC 16F84A oppure un 16F628A, che assolve esclusivamente l'incombenza di trasferire al PC, attraverso la porta seriale, il codice in questione. La comunicazione tra PC ed interfaccia avviene in ASCII, per questo il micro effettua prima dell'invio la necessaria conversione dalla differente codifica del decoder DTMF. Nel caso del 16F84A deve essere implementato anche una parte di programma per emulare il protocollo seriale poiché non dispone in hardware di questa possibilità che invece è prevista nel 16F628A. Compaiono anche alcuni led di segnalazione: DL1 che indica la ricezione in corso di un bitono DTMF, DL2 del PTT radio, DL3 del COR indicante la ricezione radio in corso. Il pilotaggio di detti led è eseguito, di norma, direttamente dal circuito del quale forniscono l'informazione con l'eccezione del led DTMF che può anche essere comandato dal microcontrollore (anche se questo in realtà replica solo quanto fornito da un'apposita linea del decoder). Alcuni jumper permettono di modificare manualmente l'hardware per selezionare le funzioni normali o negate della linea COR proveniente dalla radio e della linea PTT. Echolink consente di azionare il PTT o direttamente tramite una linea dedicata della seriale, oppure per mezzo di uno specifico comando ASCII. Sempre in hardware è realizzato un timer che ritarda l'inserzione della BF radio in ricezione dopo la caduta del PTT al fine di sopprimere un'eventuale coda di ponte ripetitore. L'interfaccia in questione, pur essendo dotata di microcontrollore, è quindi "**stupida**", perché non prende assolutamente decisioni di alcunché e si limita ad eseguire passivamente gli ordini che il programma Echolink gli invia tramite la porta seriale. Tutte le interfacce specificatamente dedicate ad Echolink, sia presentate in internet per l'autocostruzione sia in commercio già pronte all'uso, sono di questo tipo**.**

Interfaccia di terza generazione

Lo schema a blocchi è visibile in fig. 3. La principale caratteristica è che essa è di tipo "**intelligente**", quindi in grado di prendere anche "**decisioni"**. Questa particolarità ha richiesto la progettazione di un hardware che comunque, grazie al software, **non è maggiormente complesso** di quello della seconda generazione. Il problema principale consisteva nel fatto che, essendo parecchie le funzioni effettuate via software, la modifica dei parametri richiedeva un organo d'introduzione degli stessi nel micro ed uno di visualizzazione. La prima esigenza è stata risolta inviando via radio i comandi con codici DTMF. Per la seconda, invece, la scelta è caduta su Echolink stesso, in altre parole utilizzando la finestrella nel quale compaiono normalmente i caratteri corrispondenti alle note DTMF ricevute dall'RTX (zona conosciuta come **annunciators**). Co-

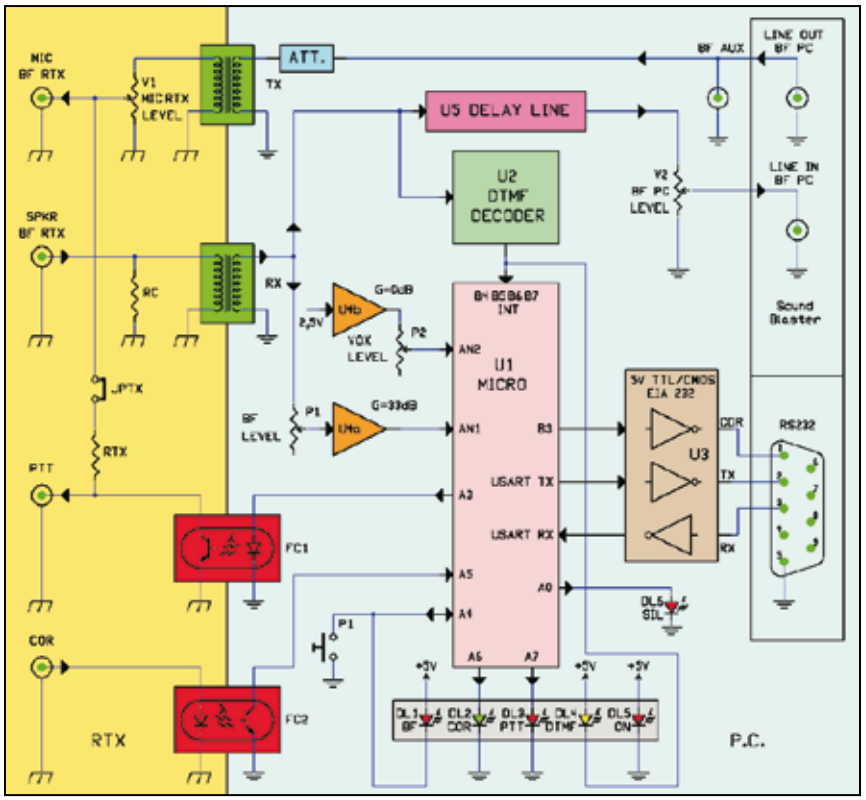

Fig. 3

me microcontrollore, valutate le varie necessità, è stato scelto un modello con un alto rapporto qualità/prezzo, cioè il PIC 16F628A. La principale differenza con i fratelli maggiori è l'assenza del converter A/D. Poiché nella mia interfaccia il microcontrollore esegue processi di trattamenti/decisioni sul segnale audio, ho dovuto sfruttare i 2 comparatori interni di cui invece dispone questo modello, riuscendo ad aggirare l'ostacolo con risultati equivalenti.

Il progetto

Veniamo adesso al progetto vero e proprio. La parte hardware ingloba alcune sezioni delle generazioni precedenti che quindi tralascio perché già trattate. La prima differenza consiste nell'**assenza** dei jumper delle funzioni COR e PTT normale/negato: tale selezione si eseguirà in software. I led COR e PTT, dovendo assolvere a molteplici funzioni, **non** sono più pilotati dai circuiti più disparati ma dipendono direttamente dal microcontrollore. Questa interfaccia lavora colloquiando con il PC **esclusivamente in ASCII** perciò la linea dedicata del comando PTT è stata tolta. Il PIC utilizzato, come detto, dispone di **porta seriale hardware** che è stata sfruttata per questa funzione. Le velocità supportate da Echolink sono 2,4 e 9,6 Kbs e quest'interfaccia è stata sperimentata **con entrambe** fornendo risultati assolutamente identici. Non vi sono significative differenze in termini di prestazioni perché il tempo impiegato per l'invio dei dati è trascurabile rispetto a quello d'inattività per questo la scelta più logica è ricaduta sui 2,4 Kbs che, almeno sulla carta, consentono meno probabilità d'errore nel trasferimento. In internet girano **software** per interfacce di seconda generazione che utilizzano il PIC16F628A e che sono esclusivamente di derivazione PIC16F84A quindi **simulando** solo la porta seriale lasciando inutilizzata quella hardware. E' bene sottolineare che, per i computer che non dispongono più della porta seriale, è sufficiente **un apposito adattatore seriale-USB**: io ho testato quest'interfaccia anche con uno di questi dispositivi. Come accennato, il micro dispone di due comparatori interni per questo è possibile acquisire anche segnali analogici che saranno in seguito trattati per via digitale. Grazie a tale opportunità questa interfaccia sostituisce con elevata affidabilità tutta la funzione VOX d'Echolink consentendo anche l'accensione del led COR indicante una ricezione in corso. Il sistema in uso è misto hardware-software in modo di sfruttare i vantaggi di entrambi. A differenza del programma Echolink, il funzionamento può avvenire in **tre modi: VOX, COR, MIX**: prescindendo da quanto selezionato sull'interfaccia, nel programma Echolink si dovrà **sempre abilitare il COR**.

Funzionamento VOX

Al microcontrollore sono applicati due distinti segnali: uno è la BF proveniente dalla radio opportunamente dosata dal potenziometro PT1 e amplificata dall'operazionale U4a, l'altro è la tensione continua proveniente dal potenziometro PT2 che regola la soglia d'intervento. Il giusto dosaggio di queste due regolazioni, per niente critiche, permettono di far funzionare il VOX perfettamente. Al fine di monitorare la BF è stato implementato un apposito led (DL1 BF) che lampeggia in conformità al segnale.

Un VOX efficiente è caratterizzato da parametri specifici (alcuni presenti anche in Echolink con nomi diversi): un tempo *variabile* di caduta del VOX al **cessare** del segnale BF (F-VOX), un tempo *fisso* di **verifica** del segnale BF (V-VOX), un tempo *variabile* d'insensibilità all'**attacco** BF (A-VOX), un tempo *variabile* d'insensibilità alla BF **dopo la caduta** del VOX (I-VOX). Il tempo V-VOX è impiegato nel riconoscimento della BF, quello A-VOX per evitare falsi agganci sui colpi di portante, quello C-VOX è il clas-

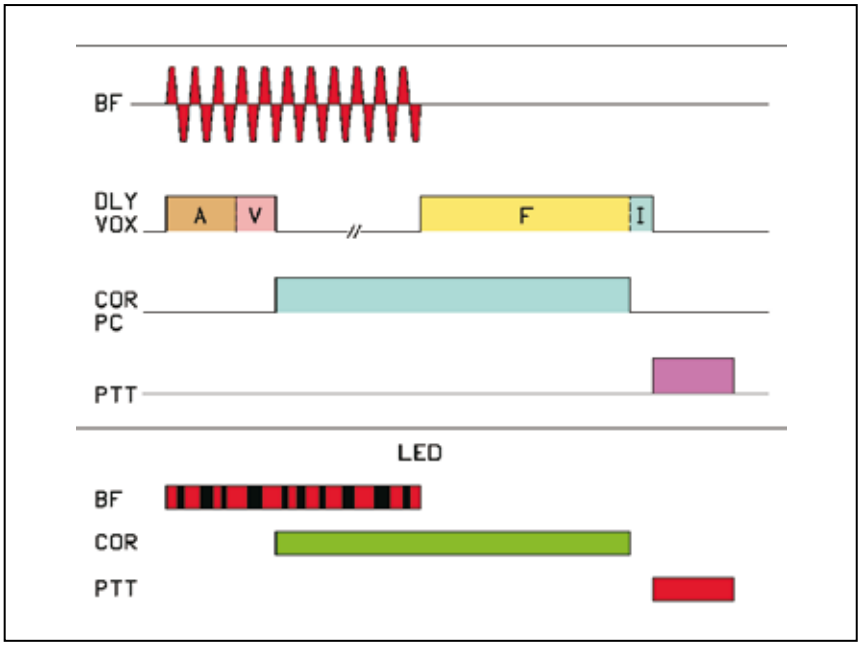

Fig. 4

sico tempo che determina la caduta del VOX al termine della BF ed infine I-VOX impedisce il riaggancio durante i rumori generati dallo squelch. Tutti questi parametri sono controllati dal micro attraverso i **timer hardware interni** con la precisione nell'ordine dei **microsecondi** e sono tutti modificabili (di quest'operazione si parlerà in seguito).

In fig. 4 sono raffigurati questi tempi (DLY VOX) in funzione della BF e l'effetto che hanno sul COR, sul PTT e sui led: il led BF lampeggia in funzione della bassa frequenza proveniente dalla radio invece il led del COR si accende fisso per tutta la durata della trasmissione verso internet. Ai più arguti non sarà sfuggito il grafico del PTT con relativo led ovvero sintomo di una trasmissione radio: trattasi di un'altra novità introdotta da questa scheda. Quando si aggancia un ponte, e cessa il segnale che lo impegna, al passaggio in ricezione si assiste alla **coda** del ripetitore fornendo quel **senso di sicurezza** nell'aver impegnato il sistema. Nel link, invece, al termine della trasmissione non avviene **nulla** lasciando nell'incertezza sia dal punto di vista "tecnico" che "psicologico" l'operatore.

Questa funzione (selezionabile da software) permette di **simulare ad un link la coda** come per un ripetitore. Tale funzione, essendo legata all'interfaccia, è attiva a prescindere dallo stato del collegamento Echolink. Qualora si operi con il VOX la risposta si attiverà al cessare del tempo I-VOX, nel caso si operi con il COR radio essa avverrà immediatamente al cessare di quest'ultimo. Ciò significa che nel primo caso basta una qualsiasi BF per ottenere una risposta (escluse note DTMF), nel secondo un colpo di portante, analogamente a quello che avviene in un ripetitore; anche a questo tempo è possibile variare la durata. Al fine di far funzionare il tutto al meglio è stato necessario introdurre una linea di ritardo sul cammino della BF tra la ricezione radio ed il PC.

Infatti, come per tutti i VOX, il riconoscimento avviene quando è già presente il segnale e ad esso vanno sommati altri ritardi dovuti al sistema. Il risultato è un taglio più o meno pronunciato della prima parte del parlato. Con l'uso classico del programma Echolink questo non avviene perché lo stesso ha già all'interno tale linea (virtuale), la stessa usata per sopprimere la BF dei bitoni DTMF verso internet: nel nostro caso va quindi aggiunta esternamente. Anni fa la cosa era improponibile perché si sarebbe dovuto assemblare un'infinità di componenti; adesso, invece, esiste un integrato specifico nato per il surround (dal costo di pochi euro) che ingloba tutto il necessario. Il fatto poi che è ideato per questo scopo fa sì che si garantiscano delle specifiche audio (come ad esempio banda passante e distorsione) **ben superiori** a quello che richiede questa specifica applicazione. Anche se nella mia realizzazione il ritardo di riconoscimento è appena avvertibile, ne esiste un altro creato appositamente per uno specifico scopo che obbliga l'utilizzo di un simile congegno. Il ritardo A-VOX, infatti, **impedisce** che sia attivata la comunicazione via internet quando s'interroga il sistema con i bitoni DTMF.

Una funzione simile è presente in Echolink ma sopprime solo la BF dei codici, **attivando comunque la trasmissione del PC**. Il programma per quest'interfaccia **riconosce immediatamente** quando si tratta di bitono DTMF invece di semplice BF e **non attiva** la procedura d'invio al computer. Il risultato finale è evidente in una conferenza. Se si pensa che i sistemi collegati sono spesso decine e altrettanto numerose possono essere le interrogazioni orarie è presto fatto il calcolo delle trasmissioni inutili con conseguente scocciatura nei confronti di chi è in ascolto. Inoltre il ritardo A-VOX **aiuta** a discriminare una portante **muta** da quella **modulata**.

(Continua)