francopic

DECODER DTMF CON UN SOLO PIC

13 December 2016

Introduzione

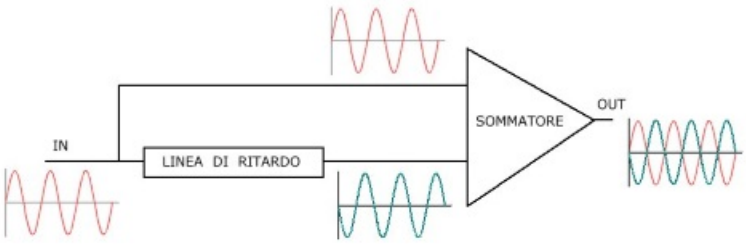
Ciao a tutti.

Eccomi ancora una volta a parlarvi del pic 16F1705 e delle sue straordinarie capacità, questa volta nelle vesti di decodificatore di toni dtmf. Desidero proporvelo, poiché i nostri progetti, senza l'ausilio di decoder dedicati del tipo MT8870, potranno essere più snelli e più semplici. Oltretutto vista l'enorme flessibilità del pic in questione, potremo se lo desideriamo, impostare delle frequenze fuori standard, al fine di ottenere dei toni dtmf personalizzati, naturalmente tutto ciò in abbinamento all'encoder dtmf che vi ho già proposto in passato.

Il raggiungimento di questo risultato è stato possibile grazie al concentrato di tecnologia presente all'interno del pic 16F1705. In particolare mi riferisco al blocco DAC, ADC ed alla RAM interna, non di grandi dimensioni è vero, ma sufficiente al nostro scopo. La tecnica da me utilizzata per la decodifica dei segnali, è nota come "Autocorrelazione dei segnali periodici", di cui per la verità non sapevo assolutamente nulla, e che avrei continuato ad ignorare se non fosse stato per il mio amico "Daniels118", che mi ha parlato per la prima volta di questa tecnica e che ringrazio per i suggerimenti.

Autocorrelazione

In estrema sintesi, l'autocorrelazione consiste nel sommare ad un segnale periodico quanto si voglia complesso, se stesso, però ritardato di un certo tempo "t". Accade che, fra tutti i segnali presenti all'ingresso del sommatore, uno sarà annullato, cioè quello il cui ritardo "t" corrisponde al suo semiperiodo. Solo in questo caso infatti i due segnali diretto e ritardato, si presenteranno all'ingresso del sommatore in opposizione di fase, cioè sfasati tra loro di 180°, per cui si elideranno a vicenda e non saranno più presenti all'uscita del sommatore, mentre tutti gli altri saranno presenti. Col disegnino che segue spero di essere più esplicito.



Filtro notch digitale

In poche parole, con tale tecnica è possibile implementare nel pic un filtro notch, cioè un filtro escludi banda, ossia un filtro che non si lascia attraversare dal segnale sinusoidale per il quale è stato progettato. Ciò accade comunemente con l'elettronica analogica, ne abbiamo visti in tutte le salse, dai semplici circuiti LC, ai più sofisticati circuiti con amplificatori operazionali.

Implementare un filtro notch digitale è abbastanza semplice, occorre solo creare una linea di ritardo pari a mezza lunghezza d'onda del segnale ricercato, ed eseguire la somma del segnale diretto e del segnale ritardato. La linea di ritardo ovviamente è costituita da un certo numero di locazioni di memoria RAM, dove si va a scrivere il segnale campionato presente all'ingresso del ADC, e dopo un certo tempo "t" programmabile, si va a rileggerlo per sommarlo al segnale diretto. Accade che, la frequenza il cui semiperiodo è impostato sulla linea di ritardo sarà annullata, azzerata; così come descritto pocanzi.

Decodifica

Immaginiamo adesso di avere due filtri notch digitali in serie, e di poter modificare ed impostare a nostro piacimento, la loro frequenza di funzionamento. Immaginiamo inoltre, di avere all'ingresso di tali filtri, la coppia di toni dtmf relativa al numero "1", ossia le due frequenze di 1209 Hz e 697 Hz. Vedi tabella.

FREQ. IN Hz.		FREQ. ALTE			
		1209	1336	1477	1633
FREQ. BASSE	697	1	2	3	A
	770	4	5	6	B
	852	7	8	9	C
	941	*	0	#	D

Tabella toni

Come avrete certamente intuito, se i due filtri sono programmati per funzionare uno a 1209 Hz e l'altro a 697 Hz, all'uscita del secondo filtro non troveremo alcun segnale, troveremo cioè zero. Infatti, il primo filtro annullerà il tono a 1209 Hz lasciando passare l'altro, che sarà annullato a sua volta dal secondo filtro, all'uscita del quale non ci sarà più niente.

Supponiamo adesso di non sapere quale sia il tono dtmf all'ingresso dei filtri, per scoprirlo basterà impostare sui filtri notch, una dopo l'altra a rotazione, tutte le sedici coppie di frequenze possibili a noi note, vedi tabella. Ebbene, solo in un caso l'uscita dei filtri sarà zero, e quando ciò accadrà avremo anche trovato le due frequenze che compongono il tono dtmf, e di conseguenza il codice dtmf associato ad esse.

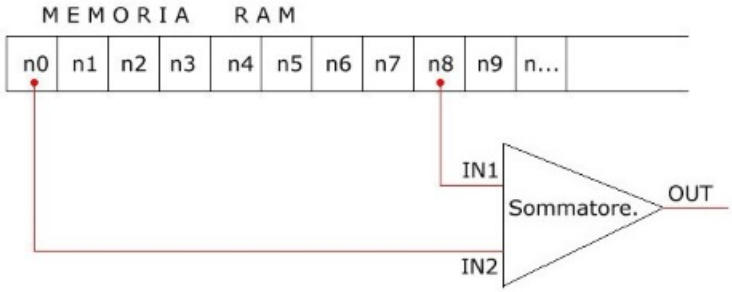
Per tutte le altre coppie, il segnale d'uscita dei filtri sarà diverso da zero, infatti può accadere che un filtro blocchi e non lasci passare una delle due frequenze, ma l'altra passerà, e se non sarà bloccata dal successivo filtro ce la ritroveremo all'uscita, e di conseguenza sapremo che non è quello il tono dtmf che stiamo cercando.

Funzionamento

Per cominciare il segnale d'ingresso viene digitalizzato con una frequenza di campionamento di circa 29,4 Khz, e se il segnale campionato supera il valore prefissato come tensione di soglia, viene memorizzato sulla RAM del pic. La soglia altro non è che una sorta di circuito squelc, indispensabile per evitare di memorizzare il fruscio di fondo del ricevitore.

Questa operazione viene ripetuta fino a memorizzare 240 campioni, per un tempo totale di 8 mS circa, campioni più che sufficienti per consentire le successive operazioni, che naturalmente richiederanno un certo tempo di elaborazione. Questo significa come vedremo più avanti, che già un segnale dtmf dalla durata di appena 25 mS sarà riconosciuto e decodificato correttamente.

A seguire sul segnale così memorizzato, eseguiremo il primo filtraggio, e sarà quindi riletto dalla RAM da due punti differenti. Il primo punto è quello iniziale che chiamiamo segnale diretto, il secondo punto che si trova più avanti che chiamiamo segnale ritardato, dista dal precedente esattamente di un certo numero di locazioni, per un tempo complessivo pari a mezz'onda. I due segnali quindi sono sommati tra loro ed il risultato salvato su un secondo blocco di RAM.

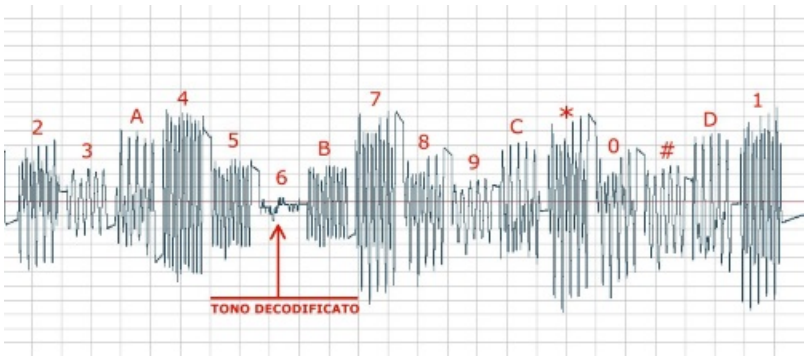


Tali operazioni naturalmente vanno eseguite dalla prima all'ultima cella di memoria, per tutte le 240 locazioni e sempre con lo stesso ritardo; quindi con le coppie n0+n8; n1+n9; n2+n10 e cosi via. Adesso occorre ripetere la stessa procedura adottata per il primo blocco di RAM, anche sul secondo blocco. Supponendo che i ritardi impostati siano quelli giusti, una frequenza sarà annullata dal primo filtro e l'altra dal secondo, ed all'uscita del secondo blocco ci ritroveremo zero.

Se così sarà, vuol dire che quelle sono le due frequenze che componevano il tono dtmf che abbiamo quindi individuato. Se così invece non sarà, ripeteremo per sedici volte le operazioni di filtraggio, cambiando di volta in volta i ritardi introdotti in modo appropriato, fino a quando almeno per una di tali operazioni il risultato sarà zero, e quando ciò accadrà sapremo anche che quello è il tono riconosciuto e decodificato.

Questa tecnica non è tra le più sofisticate, ma il software di volta in volta, adegua automaticamente i valori delle tensioni di soglia, all'ampiezza del segnale in ingresso all'ADC, ed alle ampiezze in uscita dai due filtri. Tali adeguamenti ricavati dal valore di picco di tutti i campioni memorizzati, consentono una corretta decodifica dei toni dtmf nel cento per cento dei casi, e per una vasta gamma di valori di tensione in ingresso.

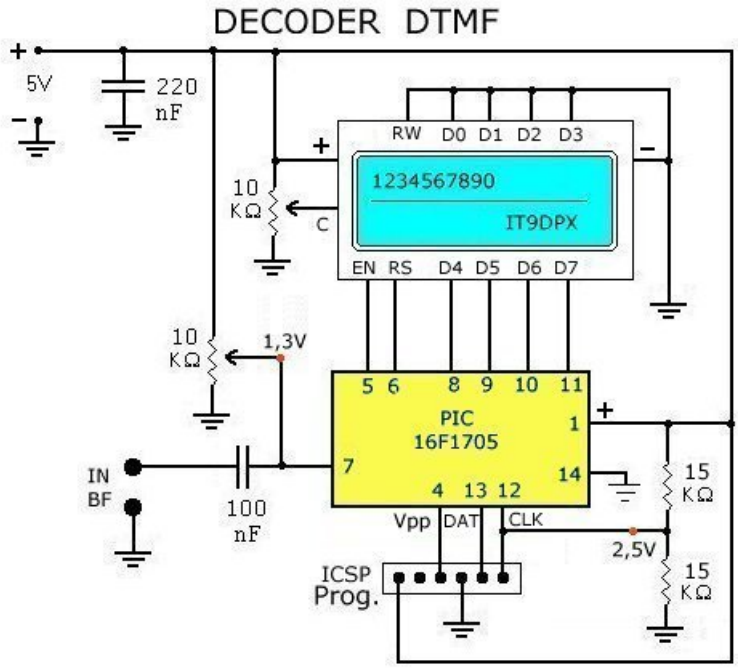
Per la precisione, ci tengo a specificare che nella realtà, dato che si lavora sulle ampiezze dei segnali, all'uscita dei filtri non sempre otterremo proprio zero, ma sicuramente il segnale decodificato avrà la minore ampiezza rispetto a tutti gli altri quindici dtmf. Vedi figura che segue.



Segnale in uscita dai due filtri notch.

Essendo la frequenza di clock del pic pari a 32 MHz, le operazioni di riconoscimento del dtmf sono eseguite in pochissimi mS, ed effettuando delle prove con un generatore di segnali dtmf esterno, ho potuto verificare che i toni sono decodificati correttamente, anche quando la loro durata è pari a 25 ms con una pausa di intertono di 50 mS. Ciò significa che in un secondo, il circuito che vi propongo può decodificare correttamente una raffica di 13 DTMF circa.

Lo schema elettrico come visibile dalla figura che segue, è veramente ridotto all'osso. Troviamo il solo pic, che oltre ad occuparsi della decodifica dei toni deve pilotare anche il display, e se aggiungiamo altro software, sarà possibile realizzare ad esempio una chiave dtmf senza l'ausilio di decodificatori dedicati, o tutto ciò che la nostra fantasia può immaginare.



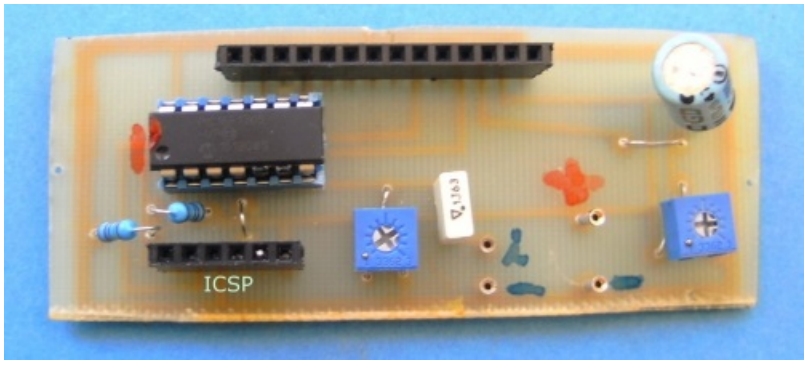
Schema elettrico decoder

Il primo potenziometro collegato al display serve a regolarne il contrasto. Il secondo, bisogna regolarlo fino a leggere sul suo pin centrale evidenziato dal pallino rosso, una tensione di circa 1,3V, questo sarà il valor medio di tutti i campioni memorizzati. Al centro tra le due resistenze da 15K, deve essere presente una tensione di 2,5V, che sarà la tensione di riferimento massima per il convertitore ADC del pic.

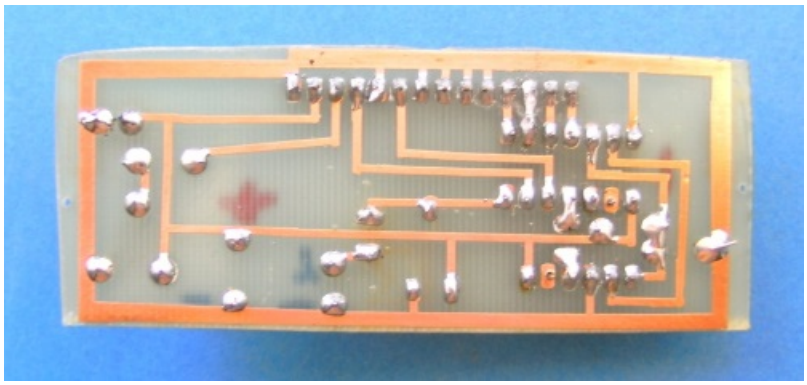
Nel circuito non ci sono altri punti di taratura, Per le mie prove ho sempre prelevato il segnale dalla presa dell'altoparlante esterno del mio ricevitore VHF, con un livello del segnale di circa 1,5V; anche se ho verificato che regolando il volume, per livelli di tensione compresi tra 0.6 e 2 Vpp, la decodifica è sempre stata perfetta.

A seguire troverete il **File.asm**, che ho corredato di varie note e che vi invito ad approfondire, perché permette d'implementare funzioni che altrimenti con altri linguaggi non sarebbero assolutamente possibili.

Display lcd



Componenti



Piste

Buon lavoro e buon divertimento.

Saluti....it9dpx

Francesco Mira. #135

```
;*************************
; DTMF DECODER   02 07 2016
; REGISTRO SU RAM DEL PIC PER 8MS CIRCA
; RILEGGO SEGNALE SENZA INTERRUPT ELIMINO TONO BASSO E RISCRIVO
; RILEGGO SEGNALE SENZA INTERRUPT  ELIMINO TONO ALTO E RISCRIVO
; RILEGGO SEGNALE PRIVO DI TONI
; provo con tute le combinazioni di ritardi
; buon compromesso per tutti i toni
; ESEGUO (A+B)/2
; A FINE ELABORAZIONE INVIO NUMERO DI TONO AD RC0 RA2 RA4 RA5
; FUNZIONA BENE ENTRO UN'AMPIA ESCURSIONE DI VPP D'INGRESSO
; impulso out di interrupt se tono è valido
; una sola passata di tutte le combinazioni
; 32 ms per la decodifica 24 ms per l'interrupt
; DECODIFICA CONTINUAMENTE
; controllo dello squelc
; INVIO DATI DECODIFICATI AL DISPLAY CON LATC,1

; RC0 = OUT D6
; RC1 = N.C.
; RC2 = OUT D4
; RC3 = ING. B.F.
; RC4 = OUT RS
; RC5 = OUT CK (EN)
; RA0 = ICSP
```

```
; RA1 = V.REF ADC  ICSP
; RA2 = OUT D7
; RA3 = ICSP
; RA4 = OUT D5
; RA5 = N.C.
;*************************

   PROCESSOR       16F1704
   RADIX           DEC
   INCLUDE         "P16F1704.INC"

   ERRORLEVEL      -302
   ERRORLEVEL      -305

 CBLOCK 70H  ;   ORG 0170H  ;  130H
INC_PASSO
TEMP_PASSO
PASSO_A
PASSO_B
RIT        : 2
PASSATA
CONTATORE
VALORE_VPP
VALORE_MIN
VALORE_MAX
LIVEL_TONO
NUMER_TONO
REGTEMP
CONT16CHR
 ENDC

    __CONFIG  H'8007', H'3FA4'
    __CONFIG  H'8008', H'1FFF'

#define     RS    PORTC,4
#define     CK    PORTC,5

       ORG 00H
  GOTO   VIA

;----INTERRUPT---------
    ORG 04
;  NOP
```

```
 ; NOP
;  NOP
  NOP

  MOVLB   0
  MOVLW   124
  MOVWF   TMR0

  BTFSC   PASSATA,5
  GOTO    PROSSIMO
  CALL    SQUELC
  GOTO    FINE_INT
PROSSIMO
  BTFSC   PASSATA,0
  GOTO    PROSSIMO_1
  CALL    REGISTRA_BUFFER_1
  GOTO    FINE_INT
PROSSIMO_1
  BTFSC   PASSATA,6
  GOTO    PROSSIMO_2
  CALL    RILEGGI
  GOTO    FINE_INT
PROSSIMO_2
  BTFSC   PASSATA,7
  GOTO    PROSSIMO_3
  CALL    FINE_TONO
  GOTO    FINE_INT
PROSSIMO_3
  CLRF    PASSATA
FINE_INT
  BCF     INTCON,2
  RETFIE
;----FINE INTERRUPT-------

VIA
  MOVLB  1
  MOVLW    B'00000011'
  MOVWF    TRISA

  MOVLW    B'00001000'
  MOVWF    TRISC

  MOVLB  3
```

```
  MOVLW    B'00001000'
  MOVWF    ANSELC
  CLRF   ANSELA

  MOVLB  0

  MOVLW    135
  MOVWF    TMR0

  MOVLW  B'00000000'
  MOVWF   PORTC

  MOVLW  B'00000000'
  MOVWF   LATA

  MOVLB    2

  CLRF   CM1CON0
  CLRF   CM2CON0

;     MOVLW    B'10000000'
;     MOVWF    DAC1CON0

  MOVLB    10

;       MOVLW    B'10010010'
  CLRF    OPA1CON
  CLRF    OPA2CON

  MOVLB    30
  CLRF     CLC1CON
  CLRF     CLC2CON
  CLRF     CLC3CON

  MOVLB    1

  MOVLW    B'00011101'
  MOVWF    ADCON0

  MOVLW    B'01100010'
  MOVWF    ADCON1
```

```
  MOVLW    B'00000000'
  MOVWF    ADCON2

  MOVLW    B'11110000'    ;26mS
  MOVWF    OSCCON

  CLRF   OPTION_REG   ;,7 ;pull-up
;--------------------
;  MOVLB  3
;  MOVLW  B'00001000'  ;MOVLW  B'00010000'
;  MOVWF  ANSELC    ;MOVWF  ANSELA

;----------------
  CLRF   INC_PASSO
  CLRF   PASSATA

  CALL   INIZ_LCD

  MOVLB    1
  MOVLW  B'10100000'
  MOVWF  INTCON

;-----------------------

ATTESA
;  goto   prova
  NOP
  NOP
  NOP
  NOP
  NOP
  GOTO    ATTESA


REGISTRA_BUFFER_1
;----------------------------------
;-----scrivo un byte nella ram------

   ;inizializzo ram
   BTFSC   PASSATA,3
   GOTO    FINIZIALIZ_FINE

   CALL    INIZIALIZZO_RAM_1
```

```
   MOVLW  .2
   MOVWF  RIT+1
   MOVLW  .224
   MOVWF  RIT+0

FINIZIALIZ_FINE
   ;avvio nuova ADC
   MOVLB  1
   BSF    ADCON0,ADGO

   ;ADC terminata ?
   BTFSC   ADCON0,ADGO
   GOTO    $-1
   MOVFW   ADRESH

   ;scrivo dato
   ;  MOVLW   .33 ;TOGLIERE DOPO
   MOVWF   INDF0
   MOVLB    .31
   INCF    FSR0L_SHAD
   BTFSC   STATUS,Z
   INCF    FSR0H_SHAD

   DECFSZ  RIT+0
   RETURN
   DECFSZ  RIT+1
   RETURN

  ;fine  memoria
   BSF   PASSATA,0  ;fine_registrazione
   BCF   PASSATA,3  ;bit inizializzazione

   MOVLW   .255 ;prepara alla decodifica
   MOVWF   LIVEL_TONO
   RETURN
;----fine scrittura------
;------------------------

SPOSTA_BUFFER_2A1
   ;inizializzo ram
   CALL    INIZIALIZZO_RAM_1
   CALL    INIZIALIZZO_RAM_2A
INIZ_FATTA
```

```
   MOVFW   INDF1
   MOVWF   INDF0
   MOVLB    .31
    INCF    FSR1L
    BTFSC   STATUS,Z
    INCF    FSR1H
   INCF    FSR0L
   BTFSC   STATUS,Z
   INCF    FSR0H
     DECFSZ  CONTATORE
   GOTO    INIZ_FATTA
   BCF   PASSATA,3  ;bit inizializzazione
   RETURN

;---------------------------
INIZIALIZZO_RAM_1
   MOVLW   .240
   MOVWF   CONTATORE ;fine memoria
;inizio indirizzamento indicizzato
   MOVLB    .31
   MOVLW   0X00
   MOVWF    FSR0L
   MOVWF    FSR0L_SHAD
   MOVLW   0X20
   MOVWF    FSR0H_SHAD
   MOVWF    FSR0H
   BSF      PASSATA,3
   RETURN
;------------------------
;---------------------------
INIZIALIZZO_RAM_2
   MOVLW   .240
   MOVWF   CONTATORE ;fine memoria
INIZIALIZZO_RAM_2A
;inizio indirizzamento indicizzato
   MOVLB    .31
   MOVLW   0XF0
   MOVWF    FSR1L
   MOVWF    FSR1L_SHAD
   MOVLW   0X20
   MOVWF    FSR1H_SHAD
   MOVWF    FSR1H
   BSF      PASSATA,3
```

```
   RETURN
;-------------------------

;-------------------------
INIZIALIZZO_RAM_3

   MOVLB    .31
   MOVFW   PASSO_A   ;MOVLW    .10  ;valore del salto
   MOVWF    FSR1L
   MOVWF    FSR1L_SHAD
   MOVLW    0X20
   MOVWF    FSR1H_SHAD
   MOVWF    FSR1H
   RETURN
;-------------------------
;-------------------------
INIZIALIZZO_RAM_4

   MOVLB    .31
   MOVFW   PASSO_B  ;MOVLW    .18  ;valore del salto
   MOVWF    FSR1L
   MOVWF    FSR1L_SHAD
   MOVLW    0X20
   MOVWF    FSR1H_SHAD
   MOVWF    FSR1H
   RETURN
;-------------------------

PROVA_SALTI
   INCF   INC_PASSO
   MOVFW  INC_PASSO
   ANDLW  .3
   CALL   SALTO_A
     MOVWF   PASSO_A
   RRF    INC_PASSO,W
   MOVWF  TEMP_PASSO
   RRF    TEMP_PASSO,W
   ANDLW  .3
   CALL   SALTO_B
     MOVWF  PASSO_B
   RETURN

SALTO_A
```

```
   ADDWF  PCL
   RETLW  .12
   RETLW  .11
   RETLW  .10
   RETLW  .9

SALTO_B
   ADDWF  PCL
   RETLW  .21
   RETLW  .19
   RETLW  .17
   RETLW  .16

;------------------------
TABELLA
   CLRF   PCLATH
   ADDWF  PCL
   RETLW "1"
   RETLW "2"
   RETLW "3"
   RETLW "A"
   RETLW "4"
   RETLW "5"
   RETLW "6"
   RETLW "B"
   RETLW "7"
   RETLW "8"
   RETLW "9"
   RETLW "C"
   RETLW "*"
   RETLW "0"
   RETLW "#"
   RETLW "D"
;---------------

ELIMINA_TONO_BASSO
   ;inizializzo ram
   CALL    INIZIALIZZO_RAM_1
   CALL    INIZIALIZZO_RAM_4
   GOTO    NO_INIZIALIZZ

ELIMINA_TONO_ALTO
   ;inizializzo ram
```

```
   CALL    INIZIALIZZO_RAM_1
   CALL    INIZIALIZZO_RAM_3

NO_INIZIALIZZ
   MOVFW   INDF1 ; leggo dato da ram con salto
   ADDWF   INDF0 ; sommo dato alla ram
   RRF     INDF0 ; diviso due annullo tono

   DECFSZ  CONTATORE
   GOTO    INCREMENTA_MEM
   RETURN

INCREMENTA_MEM
   MOVLB    .31

   INCF    FSR1L
   BTFSC   STATUS,Z
   INCF    FSR1H

   INCF    FSR0L
   BTFSC   STATUS,Z
   INCF    FSR0H
   GOTO   NO_INIZIALIZZ  ;RETURN

;---fine lettura 2 byte-------
;-----------------------------

;//////////////////////////
;//////////////////////////
RILEGGI
;-----leggo tono alto dalla ram------

   ;inizializzo ram
   BTFSC   PASSATA,3
     GOTO    NO_INIZIALIZ
   CALL  SPOSTA_BUFFER_2A1
   CALL  ELIMINA_TONO_BASSO
   CALL  ELIMINA_TONO_ALTO

   CALL    INIZIALIZZO_RAM_1
   MOVLW   .200
   MOVWF   CONTATORE ;fine memoria
```

```
NO_INIZIALIZ
   MOVFW   INDF0 ; leggo dato da ram
   MOVLB   2
   MOVWF   DAC1CON1 ;invio all'adc

;---salvo apiezza tono decodificato---
TROVA_TONO
   BTFSC   PASSATA,4
     GOTO    DECRE_AMP
   BSF     PASSATA,4
   MOVWF   VALORE_MAX ;1^ dato di riferimento
   MOVWF   VALORE_MIN ;1^ dato di riferimento
   GOTO    INCRE_FSR

DECRE_AMP
   MOVFW   INDF0 ; leggo dato da ram
   SUBWF   VALORE_MIN,W
   BTFSS   STATUS,C ; è> o è< ?
     GOTO    INCRE_AMP ;è>
   MOVFW   INDF0 ; leggo dato da ram
   MOVWF   VALORE_MIN ;è<
     GOTO    INCRE_AMP

INCRE_AMP
   MOVFW   INDF0 ; leggo dato da ram
   SUBWF   VALORE_MAX,W
   BTFSC   STATUS,C ; è> o è< ?
     GOTO    INCRE_FSR  ;è<
   MOVFW   INDF0 ; leggo dato da ram
   MOVWF   VALORE_MAX ;è>
     GOTO    INCRE_FSR
INCRE_FSR
 ;--fine controllo ampiezza----

   DECFSZ  CONTATORE
   GOTO    INCREMENTA_2

   BCF     PASSATA,4 ;nuovo tono
     MOVFW   VALORE_MIN ;calcolo Vpp
     SUBWF   VALORE_MAX,W
     MOVWF   VALORE_VPP ;salvo Vpp

   SUBWF   LIVEL_TONO,W
```

```
   BTFSS   STATUS,C
     GOTO    AMP_MAGG
   MOVFW   VALORE_VPP
   MOVWF   LIVEL_TONO ;salvo la minore Vpp
   MOVFW   INC_PASSO
   ANDLW  .15
   MOVWF   NUMER_TONO ;salvo num. del tono
AMP_MAGG

   CALL  PROVA_SALTI
   MOVFW  INC_PASSO
   ANDLW  .15  ;passaggio per lo zero
   BTFSS  STATUS,Z
     GOTO    ANCORA_TONO

;---controllo tono valido---
   MOVFW   LIVEL_TONO
   SUBLW   .16
   BTFSS   STATUS,C
     GOTO    NO_TONO_VALIDO
;   MOVLB  0
;   MOVLW  B'00000000'
;   MOVWF   LATA
   CALL   TX_LED  ;CALL    TX_TONO
 ;  BSF    PORTC,1 ;invio impulso
 ;  CALL   RITARDO
 ;  BCF    PORTC,1 ;invio impulso
   BSF    PASSATA,6
NO_TONO_VALIDO
   BCF   PASSATA,0  ;nuova_registrazione
   BCF   PASSATA,3  ;nuova_registrazione
   RETURN
   GOTO   $-1  ;fine ricerca tono
;---fine controllo tono valido---

   MOVLW   .255 ;prepara alla decodifica
   MOVWF   LIVEL_TONO

ANCORA_TONO
   BCF   PASSATA,3  ;bit inizializzazione
   GOTO    RILEGGI

INCREMENTA_2
```

```
   MOVLB    .31
   INCF    FSR0L
   BTFSC   STATUS,Z
   INCF    FSR0H
   GOTO    RILEGGI
;^^^^^^^^^^^^^^^^^^^^^^^^
;^^^^^^^^^^^^^^^^^^^^^^^^

RITARDO
  movlw    .127
  movwf    RIT+0
  movlw    .255
  movwf    RIT+1
  decfsz   RIT+1
  goto     $-1
  decfsz   RIT+0
  goto     $-3
  RETURN
; fine pausa

;***************************

TX_LED
   MOVLB  2
   MOVFW   NUMER_TONO
   ANDLW  .15
   CALL    TABELLA
   call    TXDATO
     INCF    CONT16CHR
     BTFSS   CONT16CHR,4  ; 16 CARATTERI INVIATI?
   RETURN
     MOVLW   80H
     CALL    TXCMD
     CLRF    CONT16CHR
     RETURN
;---------------------------------
;***controllo livello di squelc***
SQUELC
;avvio nuova ADC
   MOVLB  1
   BSF    ADCON0,ADGO
;ADC terminata ?
   BTFSC   ADCON0,ADGO
```

```
   GOTO    $-1
   MOVFW   ADRESH
SALVA_RIF_AMP
   BTFSC   PASSATA,4
     GOTO    MINORE_AMP
   BSF     PASSATA,4
   MOVWF   VALORE_MAX ;1^ dato di riferimento
   MOVWF   VALORE_MIN ;1^ dato di riferimento
   MOVLW   .50  ;tempo d'ascolto
   MOVWF   CONTATORE
   MOVLW   .16  ;livello di squelc
   MOVWF   LIVEL_TONO
   GOTO    FINE_CONTROLLO
MINORE_AMP
   MOVFW   ADRESH ; leggo dato da adc
   SUBWF   VALORE_MIN,W
   BTFSS   STATUS,C ; è> o è< ?
     GOTO    MAGGIORE_AMP ;è>
   MOVFW   ADRESH ; leggo dato da adc
   MOVWF   VALORE_MIN ;è<
     GOTO    MAGGIORE_AMP
MAGGIORE_AMP
   MOVFW   ADRESH ; leggo dato da adc
   SUBWF   VALORE_MAX,W
   BTFSC   STATUS,C ; è> o è< ?
     GOTO    FINE_CONTROLLO  ;è<
   MOVFW   ADRESH ; leggo dato da adc
   MOVWF   VALORE_MAX ;è>
     GOTO    FINE_CONTROLLO
FINE_CONTROLLO
   DECFSZ   CONTATORE
   RETURN
  BCF     PASSATA,4 ;prepara per nuovo controllo
   MOVFW   VALORE_MIN ;calcolo Vpp
   SUBWF   VALORE_MAX,W ;nuova Vpp
   SUBWF   LIVEL_TONO,W ;livello di squelc
   BTFSS   STATUS,C
     GOTO    SEGNALE_PRESENTE
   RETURN   ;assenza di segnale
SEGNALE_PRESENTE
   BSF    PASSATA,5  ;presenza di segnale
   RETURN
;***fine controllo livello di squelc***
```

```
;---------------------------------
;***controllo livello di squelc***
FINE_TONO
;avvio nuova ADC
   MOVLB  1
   BSF    ADCON0,ADGO
;ADC terminata ?
   BTFSC   ADCON0,ADGO
   GOTO    $-1
   MOVFW   ADRESH
SALVA_RIF_AMP1
   BTFSC   PASSATA,1
     GOTO    MINORE_AMP1
   BSF     PASSATA,1
   MOVWF   VALORE_MAX ;1^ dato di riferimento
   MOVWF   VALORE_MIN ;1^ dato di riferimento
   MOVLW   .50  ;tempo d'ascolto
   MOVWF   CONTATORE
   MOVLW   .16  ;livello di squelc
   MOVWF   LIVEL_TONO
   GOTO    FINE_CONTROLLO1
MINORE_AMP1
   MOVFW   ADRESH ; leggo dato da adc
   SUBWF   VALORE_MIN,W
   BTFSS   STATUS,C ; è> o è< ?
     GOTO    MAGGIORE_AMP1 ;è>
   MOVFW   ADRESH ; leggo dato da adc
   MOVWF   VALORE_MIN ;è<
     GOTO    MAGGIORE_AMP1
MAGGIORE_AMP1
   MOVFW   ADRESH ; leggo dato da adc
   SUBWF   VALORE_MAX,W
   BTFSC   STATUS,C ; è> o è< ?
     GOTO    FINE_CONTROLLO1  ;è<
   MOVFW   ADRESH ; leggo dato da adc
   MOVWF   VALORE_MAX ;è>
     GOTO    FINE_CONTROLLO1
FINE_CONTROLLO1
   DECFSZ   CONTATORE
   RETURN
```

```
  BCF     PASSATA,1 ;prepara per nuovo controllo
   MOVFW   VALORE_MIN ;calcolo Vpp
   SUBWF   VALORE_MAX,W ;nuova Vpp
   SUBWF   LIVEL_TONO,W ;livello di squelc
   BTFSC   STATUS,C
     GOTO    SEGNALE_ASSENTE
   RETURN   ;presenza di segnale
SEGNALE_ASSENTE
   BSF    PASSATA,7  ;assenza di segnale
   RETURN
;***fine controllo livello di squelc***


;------------------------------
;......INIZ_LCD.................
INIZ_LCD

  MOVLB  2
          bcf     RS
          bcf     CK
       movlw   .250    ;Wait 30 ms
       call    msDelay

;       MOVLW   30H   ;Set LCD command mode

       BCF    LATA,2 ;D7 Send a reset sequence to LCD
       BCF    LATC,0 ;D6 Send a reset sequence to LCD
       BSF    LATC,1 ;D5 Send a reset sequence to LCD
       BSF    LATC,2 ;D4 Send a reset sequence to LCD

       CALL    CARICA
       CALL    CARICA
       CALL    CARICA

;       MOVLW   20H       ;Set LCD command mode

       BCF    LATA,2 ;D7 Set LCD command mode
       BCF    LATC,0 ;D6 Set LCD command mode
       BSF    LATC,1 ;D5 Set LCD command mode
       BCF    LATC,2 ;D4 Set LCD command mode

       CALL    CARICA
```

```
        movlw   28H  ;Set 4 bit TX, 2 RIGHE
        call    TXCMD
        movlw   06H  ;Entry mode set, increment, no shift
        call    TXCMD

        movlw   0FH  ;Display ON, Curson ON, Blink ON
        call    TXCMD

        movlw   01H ;Clear display
        call    TXCMD

        movlw   .17   ;Wait 2 ms
        call    msDelay

;.........................
ripetizione
 ;  BSF  LATC,1

        MOVLW   H'CA' ;DD RAM 18° CIFRA
        CALL    TXCMD

        MOVLW   "I"
        call    TXDATO
        MOVLW   "T"
        call    TXDATO
        MOVLW   "9"
        call    TXDATO
        MOVLW   "D"
        call    TXDATO
        MOVLW   "P"
        call    TXDATO
        MOVLW   "X"
        call    TXDATO

        MOVLW   H'80' ;DD RAM 1° CIFRA
        CALL    TXCMD
;   BCF  LATC,1

ATTENDI
        CALL    PAUSA
 ;        goto   ripetizione
        return
```

```
;.............................
TXNUM
   ADDLW   30H
TXDATO
          bsf     RS
          call    TXBYTE
          return
TXCMD
          bcf     RS
          call    TXBYTE
          return
TXBYTE
          movwf   REGTEMP ;Save value to send
  ;tx 4 bits alti
       BCF    LATA,2 ;D7
       BCF    LATC,0 ;D6
       BCF    LATC,1 ;D5
       BCF    LATC,2 ;D4

          BTFSC   REGTEMP,7
          BSF    LATA,2 ;D7
       BTFSC   REGTEMP,6
       BSF    LATC,0 ;D6
          BTFSC   REGTEMP,5
          BSF    LATC,1 ;D5
       BTFSC   REGTEMP,4
       BSF    LATC,2 ;D4

        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
```

```
        nop
        bsf       CK           ; clock fase 2
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        bcf       CK

;tx 4 bits bassi
     BCF    LATA,2 ;D7
     BCF    LATC,0 ;D6
     BCF    LATC,1 ;D5
     BCF    LATC,2 ;D4

        BTFSC   REGTEMP,3
        BSF    LATA,2 ;D7
     BTFSC   REGTEMP,2
     BSF    LATC,0 ;D6
        BTFSC   REGTEMP,1
        BSF    LATC,1 ;D5
     BTFSC   REGTEMP,0
     BSF    LATC,2 ;D4

        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
```

```
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        bsf       CK           ; clock fase 2
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        bcf       CK

   movlw   .16         ;Wait 2ms
   call    msDelay
      RETURN

CARICA
   movlw   .10         ;Wait 1ms
   call    msDelay
   bsf     CK  ;Enables LCD
   movlw   .10         ;Wait 1ms
   call    msDelay
   bcf     CK  ;Disables LCD
   movlw   .10         ;Wait 1ms
   call    msDelay
   return

;------------------------
```

```
msDelay
          movwf   RIT+1
          clrf    RIT+0
RITLOOP
          nop
          decfsz  RIT+0,F
          goto    RITLOOP
          nop
          decfsz  RIT+1,F
          goto    RITLOOP
          return
;------------------------------
PAUSA
       movlw   .250    ;Wait 250 ms
       call    msDelay
       movlw   .250    ;Wait 250 ms
       call    msDelay
    RETURN

;------ 2° RIGA ------
       MOVLW   H'80' ;DD RAM 18° CIFRA
       CALL    TXCMD

;------------------------------



   NOP
   NOP
   RETURN
;-----------------------


   END
```

Estratto da "http://www.electroyou.it/mediawiki/index.php?title=UsersPages:Francopic:decoder-dtmf-con-un-solo-pic"