```
        page    132,79,1,1
        opt     rc
        title   'LMS AutoNotcher/Denoiser'

;*************************************************************
;* QRMQRN.ASM -- LMS Automatic Notch and Noise Filter         *
;*                                                           *
;* Provides Automatic Notch Filter (QRM reductor) and        *
;* Noise Filter (QRN reductor) based on the LMS algorithm.   *
;* When compiled with notch=0 noise filter code is activated.*
;* When compiled with notch=1 automatic notch filter code is *
;* activated.                                                *
;*                                                           *
;* This implementation is for Alef Null DSP CARD 4 with      *
;* Leonid monitor.                                           *
;*                                                           *
;* Algoriths are based on the book and article               *
;*      Candy, J., V.:                                       *
;*      "Signal Processing, The Modern Approach",            *
;*      McGraw-Hill, 1988                                    *
;*                                                           *
;*      Reyer, S., Hershberger, D.:                          *
;*      "Using The LMS Algorithm For QRM and QRN Reduction", *
;*      QEX, September 1992                                  *
;*                                                           *
;* Copyright (C) 1993 by Alef Null. All rights reserved.     *
;* Author(s): Jarkko Vuori, OH2LNS                           *
;* Modification(s):                                          *
;*************************************************************


        nolist
        include 'leonid.asm'                    ; DSP CARD 4 monitor equates
        list

notch   equ     0
lmslen  equ     24                              ; LMS filter lenght
buflen  equ      8                              ; lenght of sample buffer

        if notch
; Notcher constants
dlen    equ     63                              ; delay line lenght
beta    equ     0.125                           ; adaptation coefficient
decay   equ     0.99915                         ; coefficient decay value
        else
; Denoiser constants
dlen    equ     1                               ; delay line lenght
beta    equ     0.02                            ; 0.1875 adaptation coefficient
decay   equ     0.98047                         ; coefficient decay value
        endif


        org     p:user_code

; initialize address registers
        move                #<buffer+2,r7       ; interrupt handler pointer
        move                #buflen*4-1,m7

        move                #<lmscoef,r6        ; LMS filter coeff pointer

        move                #<iircoef,r5        ; IIR filter coeff pointer
        move                #4-1,m5

        move                #<buffer,r2         ; sample buffer read pointer
        move                #4-2,n2
        move                #buflen*4-1,m2
```

```
        move                #<dline,r1              ; delay line sample pointer
        move                #dlen+lmslen-1,m1

; initialize codec
; fs = 16 kHz, line input, headphones and line output, no gain and attenuation
        ctrlcd  1,r2,buflen,MIC,0.0,0.0,LINEO|HEADP,0.0,0.0
        opencd  16,NOHPF

; wait for one sample
loop    waitblk r2,buflen,1
        move                x:(r2)+,x0              ; read sample from the left channel

; highpass filter the input signal (with one biquad IIR section)
        move                #0.5*g,x1               ; scale input signal
        mpy     x0,x1,a     #<iirs,r0

        ori     #$0b,mr                             ; set left shift scaling mode
        move                x:(r0)+,x0  y:(r5)+,y0  ; s1, a1
        mac     -x0,y0,a    x:(r0),x1   y:(r5)+,y0  ; s2, a2
        macr    -x1,y0,a    x0,x:(r0)-  y:(r5)+,y0  ; new s2, get b1
        mac     x0,y0,a     a,x:(r0)    y:(r5)+,y0  ; new s1, get b2
        macr    x1,y0,a
        andi    #$f4,mr                             ; restore scaling mode

        move    a,x0                                ; back scaling
        move                #>@cvi(1.0/g+.5),x1
        mpy     x0,x1,a
        asr     a                                   ; adjust binary point
        move                a0,x:(r1)+
        move    a0,x1

; Wiener filter convolution part
        clr     a           x:(r1)+,x0  y:(r6)+,y0
        rep     #lmslen-1
        mac     x0,y0,a     x:(r1)+,x0  y:(r6)+,y0
        macr    x0,y0,a     x:(r1)-,x0  y:(r6)-,y0

        if !notch
; store FIR output as a program output if denoiser code
        move                a,y:(r2)+               ; left channel
        move                a,y:(r2)+n2             ; right channel
        endif

; calculate error (e = D - Y) to x1
        neg     a           #decay,y1
        add     x1,a        #beta,x0
        move    a,x1

        if notch
; store error as a program output if notcher code
        move                x1,y:(r2)+              ; left channel
        move                x1,y:(r2)+n2            ; right channel
        endif

; Wiener filter adaptation part
        mpyr    x0,x1,a     r6,r4                   ; x1 = beta * e
        move    a,x1

        move                x:(r1)-,x0  y:(r6)-,y0  ; get x(0), c(0)
        mpy     y0,y1,a
        macr    x0,x1,a     x:(r1)-,x0  y:(r6)-,y0  ; c(0) = decay * c(0) + e * x(0)
        do      #lmslen-1,adaloop
        mpy     y0,y1,a                 a,y:(r4)-   ; save new c
        macr    x0,x1,a     x:(r1)-,x0  y:(r6)-,y0  ; c(n) = decay * c(n) + e * x(n)
adaloop
        move                            a,y:(r4)-   ; save c(N)
        move                x:(r1)+,x0  y:(r6)+,y0
```

```
        move                x:(r1)+,x0  y:(r6)+,y0

; and do this forever
        jmp     <loop


        org     x:user_data

iirs    ds      2                                   ; highpass IIR states

buffer  dsm     buflen*4                            ; input sample buffer

dline   dsm     dlen+lmslen                         ; delay line and LMS states


        org     y:user_data

; 0.5 dB/40 dB elliptic IIR HPF
; pass 300 Hz, rej 35 Hz
g       equ     0.007                               ; biquad filter scaler
iircoef dc      -1.880563819/2.0,0.8990682309/2.0   ; biquad filter coeffs, a1, a2
        dc      -1.999905221/2.0,1.0/2.0            ;                       b1, b2

        dsm     buflen*4                            ; output sample buffer

lmscoef dsm     lmslen                              ; LMS filter coefficients

        end
```
♂