RMV ELECTRONICS INC.

Application note

Application note # 21	
Date: March 1995	
Description: ACTIVE FILTER ANALYSIS USING AN ITC232-A	**Status:** Final

Sooner or later everyone involved in electronic design comes across a project that uses RC active filters in the audio frequency range. There are many books and articles describing how to design and calculate these filters but unless you work in a very sophisticated environment you will have to trust your calculations and hope that the filter comes as close as possible to its theoretical performance. In some cases you might inject a signal and "peak" the filter to the desired frequency. This, however does not tell you anything about the quality of the filter. An error in the calculations, a faulty component or even just bad luck in terms of component tolerances and your filter can end up not working to specifications and worse, you do not know it!

The performance of a filter is evaluated from its curve of response, by injecting several different frequencies into the filter and measuring the output. The voltages are then transformed into dB of attenuation and the values are then plotted against the corresponding frequency. This is time consuming since in order to have a realistic curve you need to obtain many points. Also needed is a precise signal generator or a frequency counter and a detection circuit.

OPERATIONAL DESCRIPTION

Please refer to the schematic diagram and the functional software listing. Also refer to the appropriate manufacturers' data sheets and the ITC232-A user's manual for further detail.

The ITC232-A chip allows for an easy and quick implementation of the circuitry necessary to test an audio filter. Using an I/O-232 board, which includes an ITC232-A, a MAX232 serial driver and a MC145041 analog/digital converter, it took the author only a few minutes to bread board this application. Should you prefer to breadboard the whole circuit, use the circuit depicted in Figure 1. The ITC232-A pulse width modulation (PWM) pin is used to generate a signal of known frequency (determined by the command "Wf" where [f] is the frequency in Hz (10-10,000)). This signal is injected into the filter. After being amplified and rectified, the voltage emerging from the filter is read using one of the A/D converter channels. The program discussed in this application note shows how to:

(1) Obtain the readings between 100 and 10,000 Hz with a point every 20 Hz,
(2) Save the values on disk in a format compatible with any spreadsheet
(3) Display the voltage output in a very elementary way (the attenuation curve in dB can be done using a spreadsheet program as discussed below).

Figure 2 shows the conditioning circuit used to inject the signal and read the filter's output. R7 sets the level of the input signal whereas R4 controls the gain of the system. The D.C. voltage on pin 1 of LM324 should be ~1.2 V. To operate the system, connect the PWM signal to R6, C5 to the filter input, the filter output to C1 and pin 1 on the LM324 to channel 0 on the analog/digital converter on the I/O-232 board. Adjust the 10K trimmer on the I/O-232 board until you get a voltage of ~3.5V on the TP1 pin. This voltage is the +VREF applied to the ADC and sets the upper limit of your readings. If you get values of 255 on the readings, the reference voltage may be too low for the A/D converter. Increase the voltage on TP1 until no more 255 values are found at any frequency. The regulation of R, R and the trim pot are not critical. Different settings should be experimented with to obtain the best results.

Breadboard any of the filters shown in Figures 3a, 3b or 3c. Apply power to the system and load the QBasic software program listed. At the beginning of the program, SCREEN 9 selects a VGA screen output. Change this to the appropriate number if you are using another CRT device (Help in QBASIC lists all the possibilities). Choose the serial port to which the I/O-232 board is connected and reset the board. Your computer screen should show the following 2 options:

1. TUNE allows you to generate a given frequency on the PWM pin and shows the values obtained by the A/D converter. This is useful when you want to "tune" a filter to a given frequency.

2. ANALYZE SPECTRUM to obtain the response curve which is what we wish to do. Input a file name to save the data collected or press Enter to select the default (this will erase any previous "FILTER.PRN" file). Choose 100 Hz as the lowest frequency and 10,000 Hz as the highest frequency. Press Enter and the spectrum analysis will be plotted on the screen. On completion, the program saves the file on disk and then maximizes the graph to the screen limits (you might need to change some of the program parameters in order to fit it to screens other than a VGA). Note that this curve is a voltage curve.

Including the graph routines in the QBASIC program would make it too long to be listed here. Instead the data is inputed into a spreadsheet and its graphics capabilities are used to show and print the curves. Lotus 1-2-3® or Quattro®can be used for this purpose as follows: Import the *.PRN file generated by the Qbasic program. There should be 2 columns of values displayed. On the left are the frequency values increasing at 20 Hz intervals. On the right column the values from the A/D converter are displayed. The first value corresponds to the background. The measurement is taken with the PWM off (the first value on the frequency column is 0). The system response is not totally flat and if more precise readings are needed they can be calculated by subtracting the background curve obtained by feeding the PWM signal directly into C1. Import that set of values into the spreadsheet and use them to correct for non-linearity. (We observed a difference of only 9 units (147 units at 100 Hz and 156 units at 10,000 Hz) which represents a very small drift.)

You can now generate a third column which represents the original values minus the background (from the background curve or the first value in the set). This is the actual response of the filter.

The representation below shows the equations used in the spreadsheet. They must be copied down to the last frequency value on the spreadsheet.

```
        A          B          C                 D                     E

        Freq     A/D val    Val-Band         Attenuation(db)         Max

1       0          144                                          @MAX(C3..C498)[76]
2
3       100        219      +B3-$B$1[75]      20*@LOG(C3/$E$3)[-0.11505]
4       120 ...
```

To represent the curves, go into Graph, select type XY, assign the frequency column to the X axis and the attenuation column to the Y axis (A in Lotus, 1st series in Quattro). "View" will show the graph. You can input names for the axes, legends, etc. before printing the plot.

Figures 4, 5 and 6 show the curves obtained with the filter circuits shown in Figures a, b and c respectively. Interestingly enough, we found some undesired peaks on the band pass filter at low frequencies which are however not seen on the other filters.

There are other curves of response you can measure with this set-up. For example you can feed the signal into your stereo and analyze its curve of response. In this regard, a very interesting experiment is to feed the signal from a microphone into the conditioning signal circuitry and place the microphone in front of the stereo loud speakers. You might be surprised at what you find. Besides some less than ideal response from the microphone, you will find that the speakers curve of response falls far behind that of the amplifier (unfortunately there is no easy way to trace the curve of response).

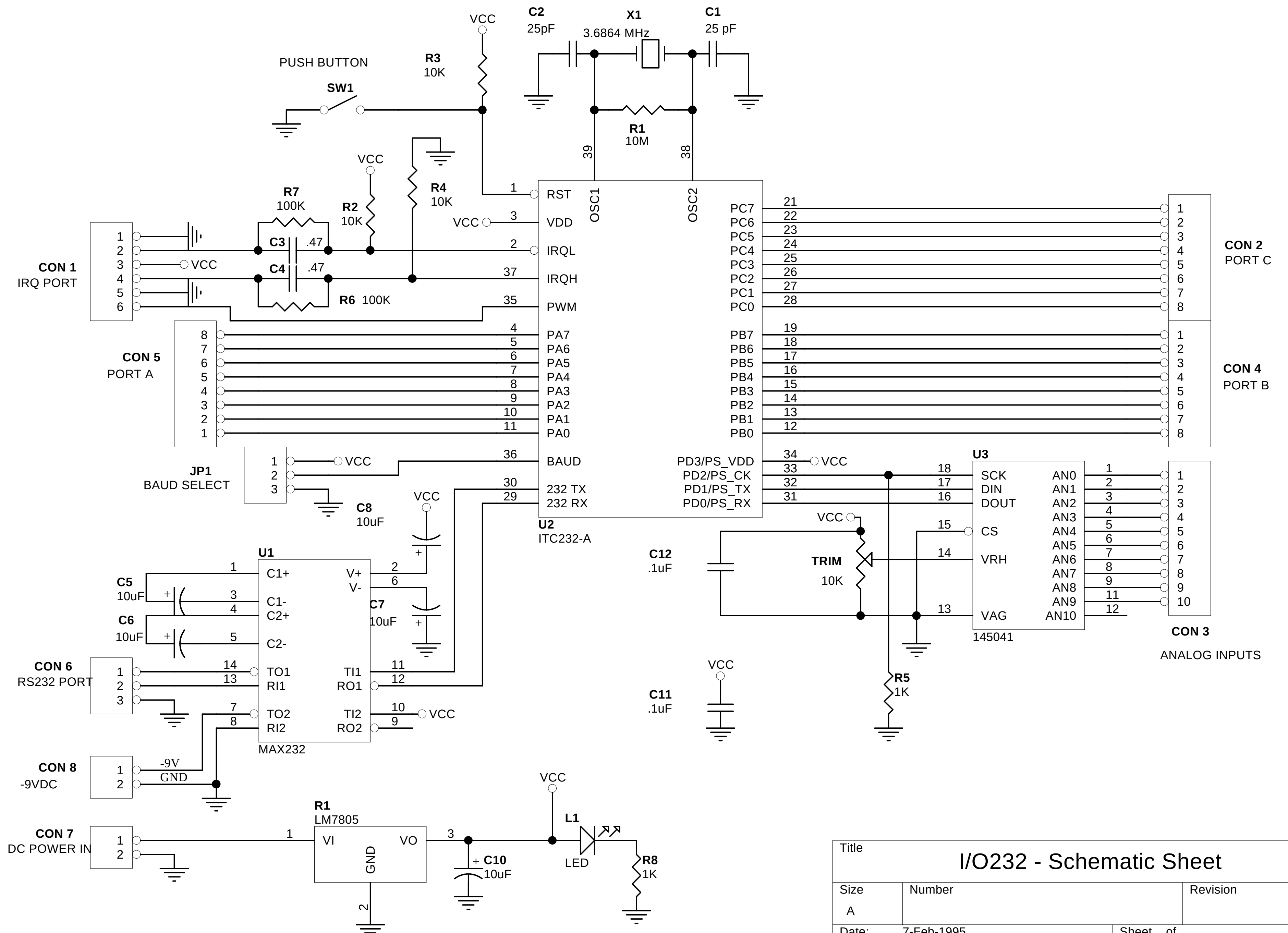
The two major drawbacks of this application are that we are using a square wave rather than a fine wave to inject into the filter and that the highest frequency available is 10KHz. On the other hand, for most practical filter applications, this circuit will prove to be very useful. If you assemble the 3 circuits in Figure 3 separately, you will be able to combine them. For example, placing a low pass filter before a high pass filter should yield the curve of response of a bandpass filter. Designing the low pass filter a frequency of 6 KHz and the high pass filter for 2KHz will yield a rather wide and flat bandpass filter. You can also try cascading to band pass sections and compare the curve thus obtained to that of only one stage.

PROGRAM LISTING (QBasic)

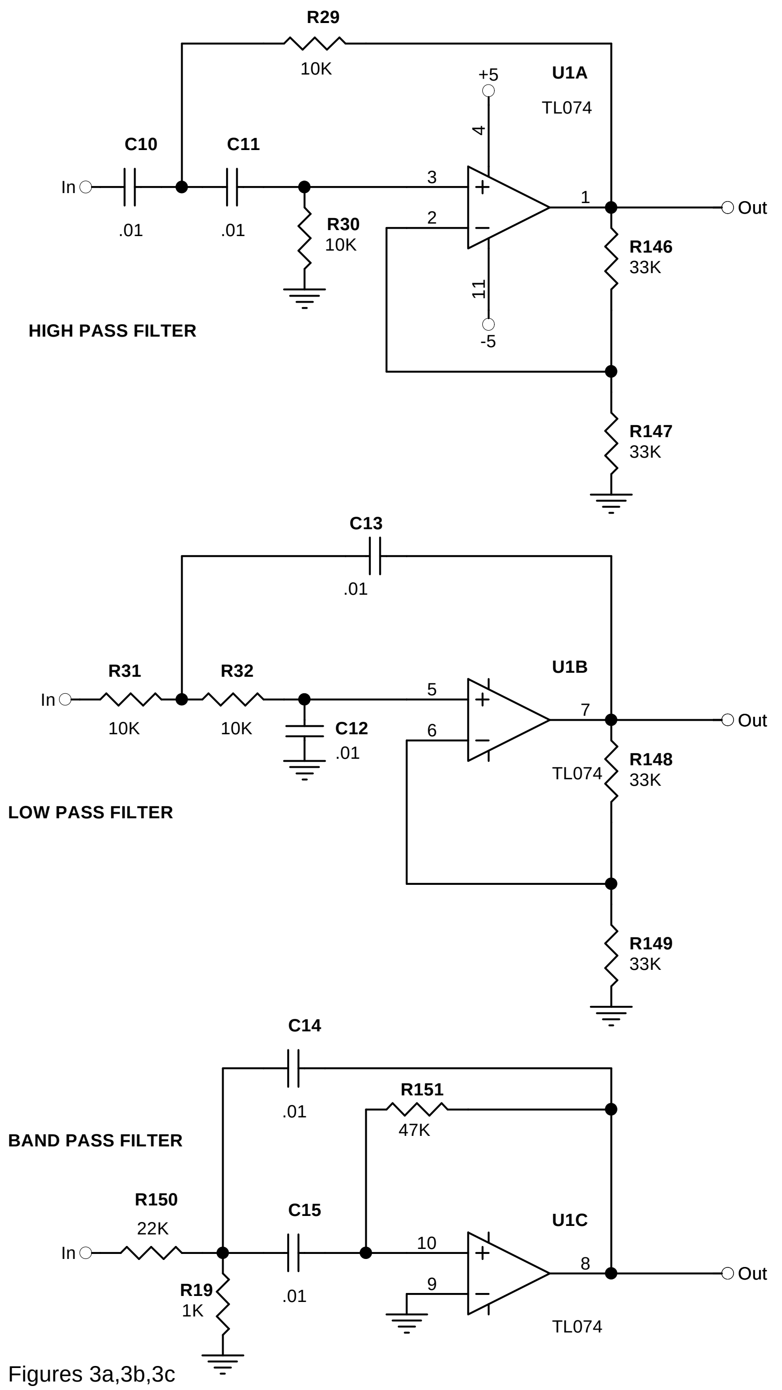
```
SCREEN 9
CLEAR
TRUE = 1: FALSE = 0
```

```
datain = FALSE
DIM d(3, 600)

REM Open COM port
CLS : LOCATE 2, 25: PRINT "FILTER SPECTRUM ANALISIS PROGRAM"
LOCATE 4, 30: PRINT "RMV ELECTRONICS INC."
LOCATE 23, 10: PRINT "THE 1st VALUE ON THE DATA FILE IS THE BACKGROUND READING"

COMport: LOCATE 10, 30: INPUT "Serial port 1 or 2 : ", c
IF c <> 1 AND c <> 2 THEN GOTO COMport
c$ = "COM" + RIGHT$(STR$(c), 1) + ":9600,N,8,1,CD0,CS0,DS0,OP0,RS,TB2048,RB2048"

OPEN c$ FOR RANDOM AS #1


LOCATE 14, 20: PRINT "PLEASE RESET THE ITC232-A or ESC TO ABORT"
RES:
IF INKEY$ = CHR$(27) THEN END
IF LOC(1) = 0 THEN GOTO RES

GOSUB READSERIAL
W$ = "CRAP": GOSUB WRITESERIAL
W$ = "PCSA128": GOSUB WRITESERIAL
W$ = "PWS0": GOSUB WRITESERIAL
W$ = "WL": GOSUB WRITESERIAL
FOR n1 = 1 TO 1000: NEXT n1
W$ = "PRS": GOSUB WRITESERIAL: GOSUB WRITESERIAL
bkgnd = v


menu1: CLS : REM Main Menu
LOCATE 2, 25: PRINT "FILTER SPECTRUM ANALISIS PROGRAM"
LOCATE 4, 30: PRINT "RMV ELECTRONICS INC."
LOCATE 23, 10: PRINT "THE 1st VALUE ON THE DATA FILE IS THE BACKGROUND READING"
LOCATE 10, 30: PRINT "0. Exit": LOCATE 12, 30: PRINT "1. Tune filter": LOCATE 14, 30: PRINT "2. Analyze spectrum"
IF datain = TRUE THEN LOCATE 16, 30: PRINT "3. Replot last curve"
menu2: LOCATE 18, 30: INPUT menu
IF menu = 0 THEN END
IF menu = 1 THEN GOSUB TUNEFILTER: GOTO menu1
IF menu = 2 THEN datain = TRUE: GOSUB SPECTRUM: GOTO menu1
IF menu = 3 AND datain = TRUE THEN GOSUB SPECTRUM: GOTO menu1
GOTO menu2

REM                     Subroutines

TUNEFILTER:
CLS : INPUT "Center frequency: ", cf
PRINT : PRINT "PRESS A KEY TO EXIT"
W$ = "W" + STR$(cf): GOSUB WRITESERIAL
TF1: W$ = "PRS": GOSUB WRITESERIAL
LOCATE 10, 10: PRINT SPACE$(4): LOCATE 10, 10: PRINT v;
IF INKEY$ = "" THEN GOTO TF1:
RETURN

SPECTRUM:
CLS : CLOSE #2
IF menu = 3 THEN GOTO SPECT1
INPUT "File name for spreadsheet file ( [Enter] = FILTER.PRN ) : ", FileName$
IF FileName$ = "" THEN FileName$ = "Filter.prn"

INPUT "Lowest f :", f1$
IF f1$ = "" THEN f1 = 100: f2 = 10000:  ELSE f1 = VAL(f1$): INPUT "Highest f :", f2

IF f1 < 100 THEN GOTO SPECTRUM
IF f2 > 10000 THEN GOTO SPECTRUM
IF f2 <= f1 THEN GOTO SPECTRUM
f1 = INT(f1 / 20) * 20: f2 = INT(f2 / 20) * 20
OPEN FileName$ FOR OUTPUT AS #2: PRINT #2, 0; " "; bkgnd
S = 2: IF f2 - f1 > 5500 THEN S = 1
W$ = "PRS": GOSUB WRITESERIAL: 'Clear last reading
SPECT1: CLS : LOCATE 24, 1: PRINT "Hz"; : LOCATE 1, 1: PRINT "V out"
LINE (10, 30)-(10, 300)
W$ = "W100": GOSUB WRITESERIAL
W$ = "PRS"
FOR n1 = 1 TO 10: GOSUB WRITESERIAL: FOR n2 = 1 TO 100: NEXT n2: NEXT n1
p = 10: max = 0

FOR n1 = f1 TO f2 STEP 20
IF INKEY$ <> "" THEN RETURN
p = p + 1
IF menu = 3 AND datain = TRUE THEN GOTO SPECT2

PRINT #2, n1; " ";
W$ = "W" + STR$(n1): GOSUB WRITESERIAL: IF p = 11 THEN GOSUB WRITESERIAL: 'Otherwise 1st value is historical
t = TIMER + .1
tl1: IF t > TIMER THEN GOTO tl1

W$ = "PRS": GOSUB WRITESERIAL
d(1, p) = p: d(2, p) = v - bkgnd: IF max < d(2, p) THEN max = d(2, p)
PRINT #2, v

SPECT2: IF p > 11 THEN LINE (lp * S, 300 - lv)-(p * S, 300 - d(2, p))
```

```
lp = p: lv = d(2, p)
IF n1 / 100 = INT(n1 / 100) THEN LINE (p * S, 305)-(p * S, 300)
IF n1 / 500 = INT(n1 / 500) THEN LINE (p * S, 310)-(p * S, 300): LINE (p * S, 300)-(p * S, 30), 6
IF n1 / 1000 = INT(n1 / 1000) THEN LINE (p * S, 315)-(p * S, 300): IF INT(p / (8 / S)) - 1 > 0 THEN LOCATE 24, INT(p / (8 / S)) - 1:
PRINT n1;
NEXT n1

LINE (p * S, 300)-(p * S, 30)
LINE (0, 300)-(p * S, 300)
LINE (10, 30)-(p * S, 30)

FOR l1 = 45 TO 285 STEP 15
LINE (5, l1)-(10, l1): LINE (10, l1)-(p * S, l1), 6
IF l1 / 75 = INT(l1 / 75) THEN LINE (0, l1)-(10, l1)
NEXT l1
W$ = "WL": GOSUB WRITESERIAL

CLOSE #2
IF menu = 3 THEN GOTO SPECT3
max = 0
FOR n2 = 1 TO p
'IF d(2, n2) > 0 THEN d(2, n2) = 20 * LOG(d(2, n2)):
IF max < d(2, n2) THEN max = d(2, n2)
NEXT n2

FOR n2 = 1 TO p
d(2, n2) = d(2, n2) * (270 / max)
NEXT n2
menu = 3: GOTO SPECTRUM

SPECT3: LOCATE 1, 60: PRINT "KEY to continue"
SP1: IF INKEY$ = "" THEN GOTO SP1
RETURN

REM  Writing to serial port
WRITESERIAL:
PRINT #1, W$
GOSUB READSERIAL
RETURN

REM  Reading serial port
READSERIAL:
S$ = ""
IF LOC(1) = 0 THEN GOTO READSERIAL

REM Get received string into S$
Lp1:
c$ = INPUT$(1, #1)
S$ = S$ + c$
IF c$ <> ">" THEN GOTO Lp1

REM decode string (V$) and value (V)
VALIDERROR = TRUE
ERRORCODE$ = ""
v$ = ""

FOR H = 1 TO LEN(S$)
IF MID$(S$, H, 1) = CHR$(7) THEN VALIDERROR = FALSE
IF MID$(S$, H, 1) = "?" THEN ERRORCODE$ = MID$(S$, H + 1, 1)
NEXT H

IF (VALIDERROR = TRUE AND ERRORCODE$ <> "") THEN GOSUB ERRORSUB: RETURN
v$ = ""
FOR n = 1 TO LEN(S$)
x$ = MID$(S$, n, 1)
IF x$ <> CHR$(13) THEN IF x$ >= "0" AND x$ <= "9" THEN v$ = v$ + x$
v = VAL(v$)
NEXT n
RETURN

ERRORSUB:
PRINT
PRINT "Error #"; ERRORCODE$
RETURN
```



Title	I/O232 - Schematic Sheet		
Size A	Number		Revision
Date:	7-Feb-1995	Sheet of	
File:	C:\RMV\ITC232A\IO232.SCH	Drawn By:	



Figures 3a,3b,3c
Filter Circuits

Signal conditioning (Amplifier and detector)

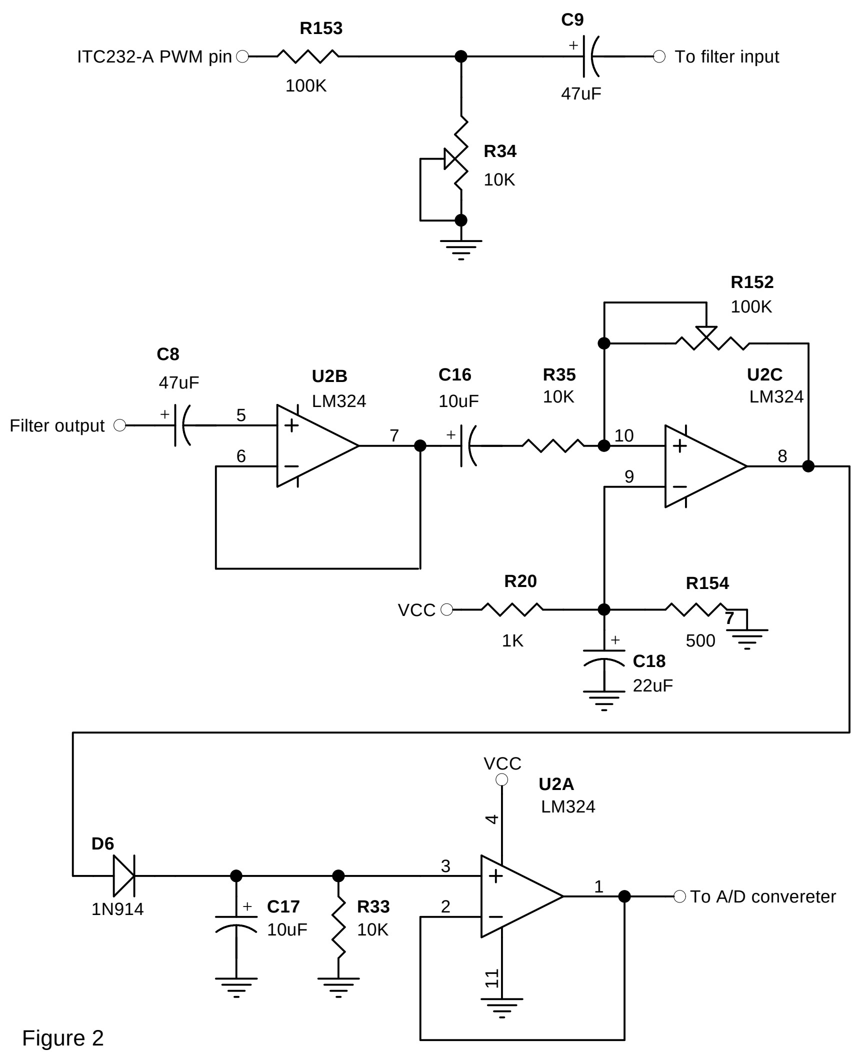


Figure 2
Conditioning Circuit

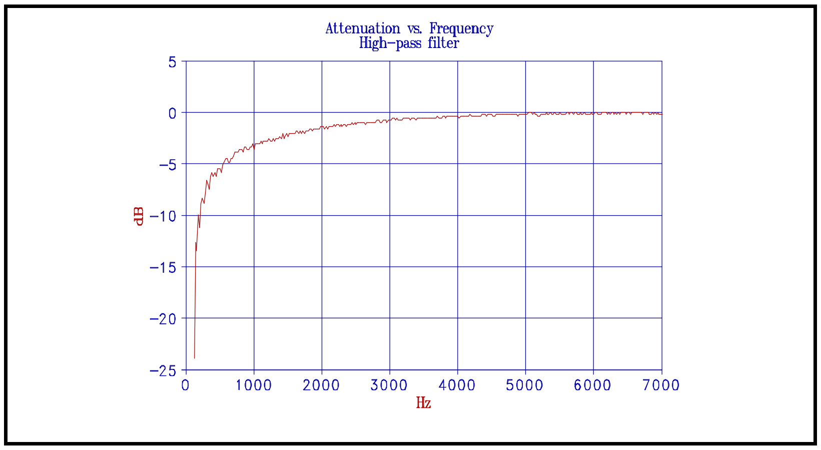


Figure 4
High pass filter response curve

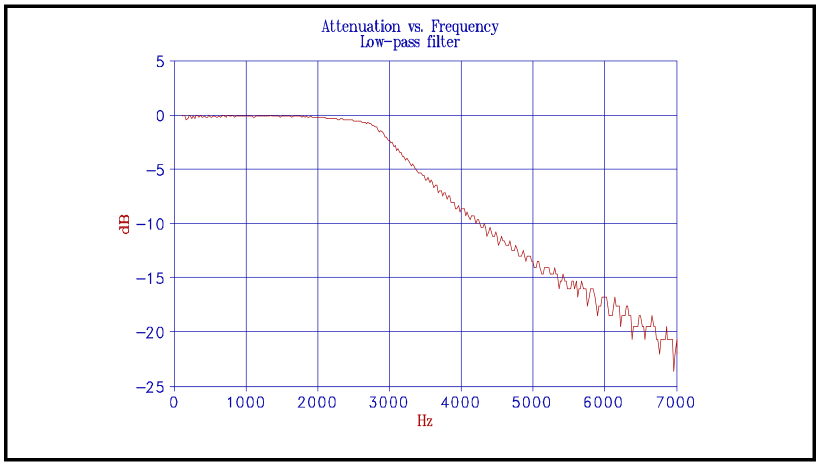


Figure 5
Low pass filter response curve

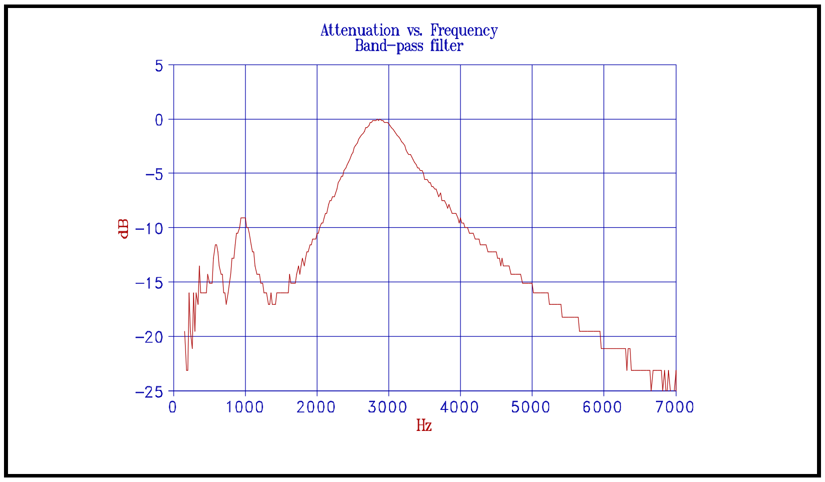


Figure 6
Band pass filter response curve