SEPTEMBER 1993

EVERYDAY WITH PRACTICAL ELECTRONICS

INCORPORATING ELECTRONICS MONTHLY

FULLY S.O.R. £1.80

LIGHTWORK
Check trailer lights without assistance

SOUND ACTIVATED TRIGGER
Tell your camera to take your picture

RAPID NICAD CHARGER
A 15 minute charge

CONSUMER ELECTRONICS SHOW-CHICAGO
The latest high tech. products reviewed by Barry Fox

PLUS: Circuit Surgery; Amateur Radio; Interface; New Technology; etc.

09
9 770262 361034

THE No. 1 INDEPENDENT MAGAZINE for ELECTRONICS, TECHNOLOGY and COMPUTER PROJECTS

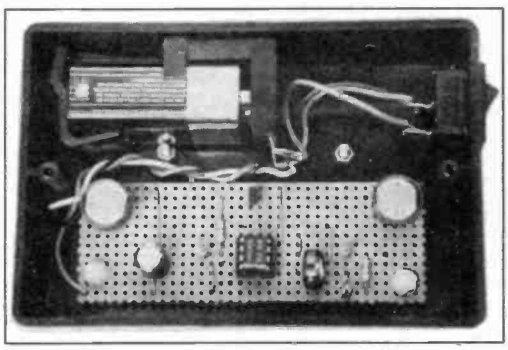
EVERYDAY WITH PRACTICAL
ELECTRONICS
INCORPORATING ELECTRONICS MONTHLY

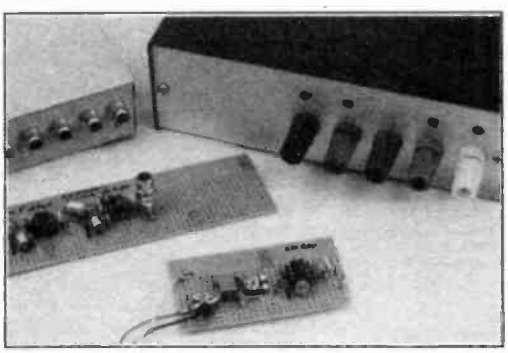
VOL. 22 No. 9 SEPTEMBER 1993

The No. 1 Independent Magazine for Electronics, Technology and Computer Projects

ISSN 0262 3617

PROJECTS . . . THEORY . . . NEWS . . .
COMMENT . . . POPULAR FEATURES . . .



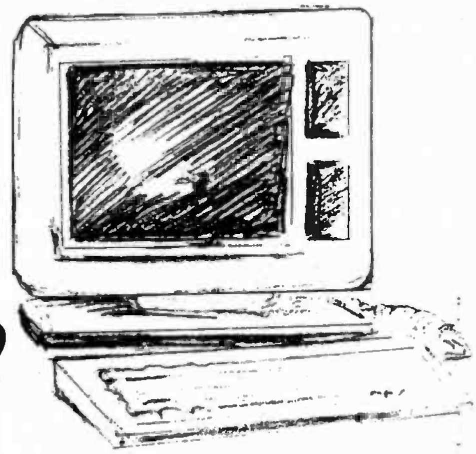


Our October '93 issue will be published on Friday, 3 September 1993. *See page 635 for details.*

Projects

Series

Features



AMSTRAD PCW 8-CHANNEL A/D CONVERTER

JASON SHARPE

PART TWO

Using the ADC for monitoring/data logging and signal sampling. Plus construction of buffer/filter and pre-amplifier boards.

FOLLOWING on from last month's constructional project, this month we set out some further programming information and outline some possible applications for the Amstrad PCW 8-Channel A/D Converter. Included are a simple add-on Buffer Board, an Active Filter Board and a Pre-amplifier Board.

There are many uses for this analogue to digital converter unit. The first part of this article describes how to use the ADC for monitoring/data logging with simple "sensors". The second half describes its use for sampling signals at higher sampling rates and some elementary signal processing.

SENSORS

Some very simple sensors which can be connected to the 8-Channel ADC are shown in Fig. 1. Potentiometers can easily be connected to the ADC inputs and these can be used as input devices i.e. hand controls. For instance, analogue joysticks, which consist of two potentiometers (X and Y), can be connected to two channels and could for example be used to control a cursor. Or the potentiometer maybe connected to a stepper motor shaft or other device, to provide position information.

The light sensor, Fig. 1b, could be used to monitor light levels during the day. If a temperature sensor was connected the temperatures during the day could be logged.

Alternatively, the ADC could be used to replace a voltmeter on an existing project, as long as the voltage range is between 0V and 5V.

SIMPLE DATA LOGGER PROGRAM

A data logger program which reads all *Eight* channels at set intervals (set by the user) is set out in Listing. 1. The period

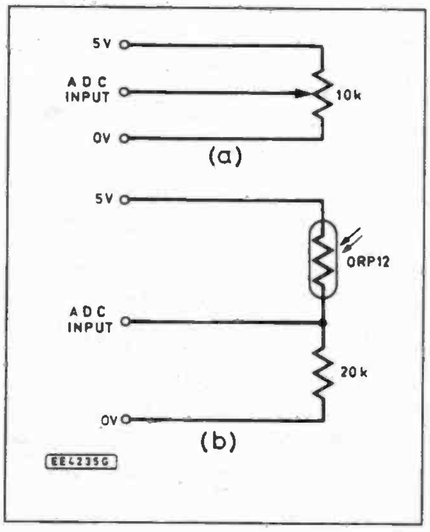


Fig. 1a. Connecting a potentiometer to the ADC. Analogue joysticks can be fitted, they consist of two 'pots' (for X and Y movement), and thus require two channels. (b) Light-level sensor. With a 20k resistor and an ORP12, the output voltage is nearly full range–from 0V to 5V

Listing 1. ADC Data Logger.

```
1 REM ***********************************************************
2 REM *                8-CHANNEL DATA LOGGER PROGRAM            *
3 REM *                  By  J.M.Sharpe (C) 1992                *
4 REM ***********************************************************
5 :
100 ma=&HFBF7:sa=ma+1:REM                 Address of internal clock
110 OPTION BASE 0:DIM d(7):V=5/255:c$=CHR$(27)+"H"+CHR$(27)+"E":
120 PRINT c$"8-Channel Data Logger Program"
125 PRINT "-----------------------------":PRINT
130 PRINT "TIME BETWEEN READINGS (Minutes[59 max],Seconds[59 max])";
135 INPUT "",M%,S%
140 m%=ABS(m%):s%=ABS(s%)
150 IF m%>59 OR s%>59 THEN PRINT "INVALID DELAY !":GOTO 130
160 PRINT c$:PRINT "PRESS ANY KEY TO EXIT PROGRAM"
165 REM *****************Wait for end of time delay****************
170 PRINT:POKE ma,0:POKE sa,0
180 cm=VAL(HEX$(PEEK(ma))):cs=VAL(HEX$(PEEK(sa)))
190 IF INKEY$<>"" THEN END
200 IF cm<>m% THEN 180 ELSE IF cs<>s% THEN 180
205 REM ***********Read port values into array d(0..7)*************
210 d(0)=INP(176):REM                  Start conversion on channel 0
220 FOR Chan=1 TO 7:d(Chan-1)=INP(176+Chan)*V:NEXT: d(7)=INP(183)*V
225 REM ********************Print results***************************
230 PRINT "Channel   Voltage"
240 FOR Chan=0 TO 7
250 PRINT TAB(3);Chan;TAB(11);LEFT$(STR$(d(Chan))+".000",6)
260 NEXT: GOTO 170
```

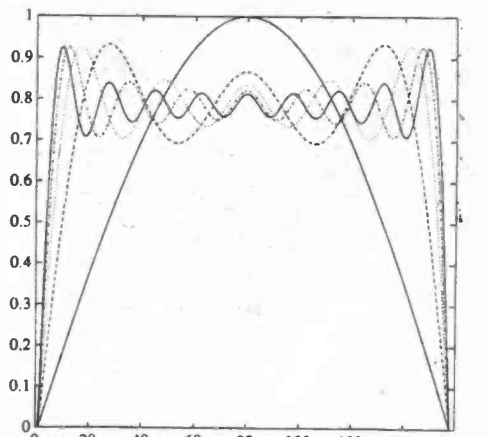


Fig. 2. The building of a square wave: Peaks at the edge are called "Gibbs' effect".

between readings is timed by using the internal clock. At present the data is simply displayed on the screen, but this could be changed so that the data can be stored in an array, or maybe dumped to the printer or a file, depending on the amount of information you wish to store.

The value read from the ADC will be between 0 and 255, this can be converted into the value of the input voltage by multiplying the value by (5 ÷ 255).

SAMPLING HIGHER FREQUENCY SIGNALS

To sample continuously changing signals (such as audio signals) ADCs are often used. This may be done for storage (e.g. converting sound to digital data for storage on CD or DAT), signal processing or signal analysis.

This all seems very straight forward, just keep reading in samples and store them in memory and process them as required. Unfortunately it is not quite this simple!

This is a very large and complicated subject, what follows is just a brief introduction. In 1807 Fourier presented his theory to the French Academy in Paris. Basically Fouriers' theory stated that an "arbitrary" single-valued real function (or signal) *can be represented by an infinite series of pure sine and cosine functions* (subject to certain conditions).

An example of this is shown in Fig. 2, a square wave can be built up of odd harmonics of sine waves, as the number of sine terms used increases the more the signal looks like a square wave. Large "peaks" start to build up at the edge of the square wave, this is called "Gibbs effect", but we shall not worry about this.

You can examine the "frequency components" (the frequencies of the sinusoidal components which make up the signal) of signals using a Spectrum Analyser – if you are lucky enough to have access to one of these, as they are rather expensive.

So what has this got to do with ADC's? Well they take samples of signals at discrete moments in time. Fig. 3 shows what happens to a signal which is sampled too slowly, the signal reconstructed from the sampled data is a lower frequency than the original signal. This is called "aliasing".

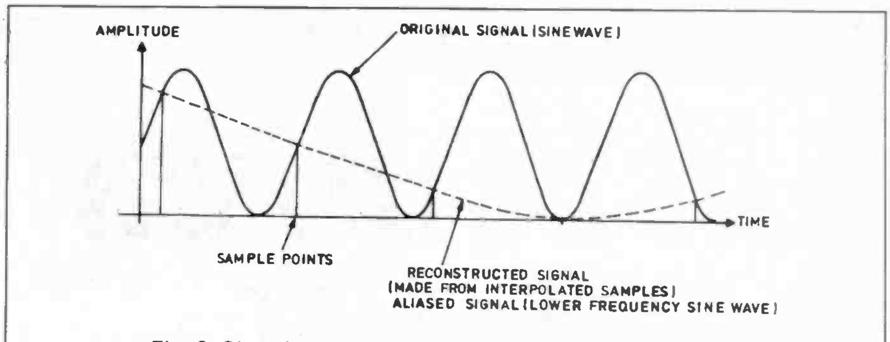


Fig. 3. Showing the effect of under sampling – "aliasing".

All frequency components of the signal which are greater than twice the sampling frequency are "aliased" to lower frequencies, which causes the sampled signal to be distorted (once this has happened there is no way of getting back the original signal). To prevent aliasing the sampling frequency should be more than twice that of the highest frequency component of the signal being sampled (this is called Nyquists' Theorem).

A low-pass filter is normally used to remove the high frequency components of the input signal. Sampling at half the highest frequency requires a perfect low-pass filter (which do not exist!), so in practice higher sampling rates are normally used.

BUFFER/FILTER

The circuit diagram of a buffer and 4th order low-pass filter is shown in Fig. 4. The buffer is there to provide a high input impedance, and also to shift the "bias" on the input signal if required. The filter cutoff frequency is about 16kHz.

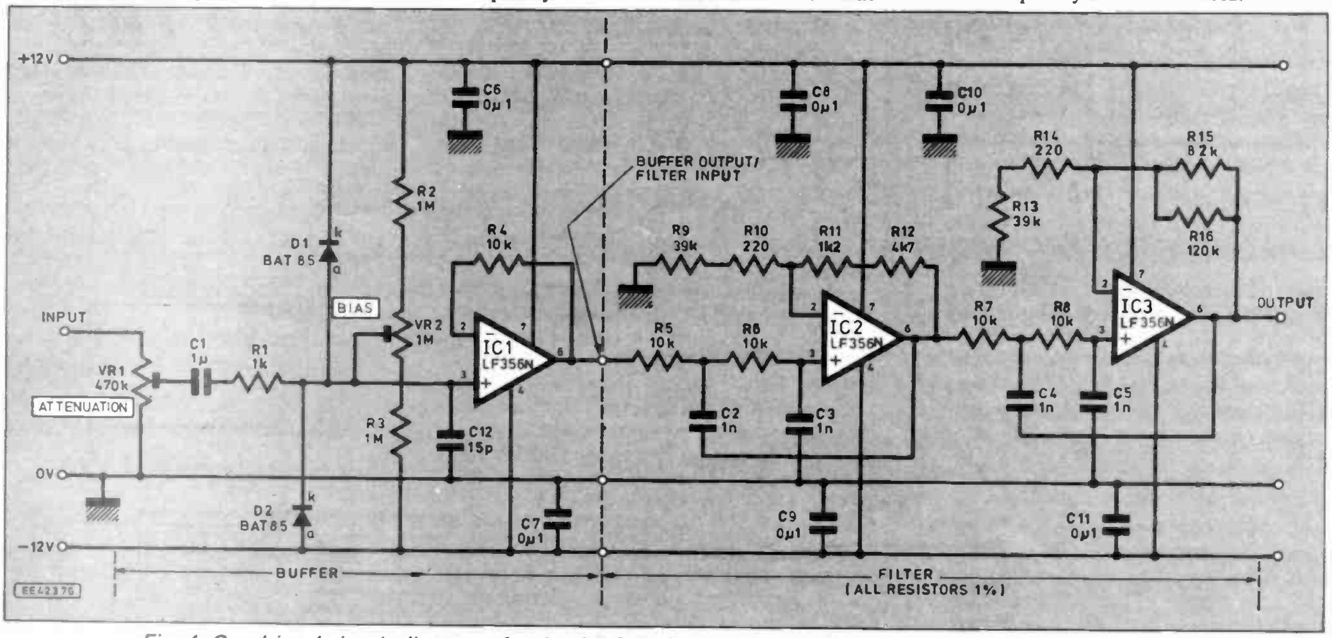
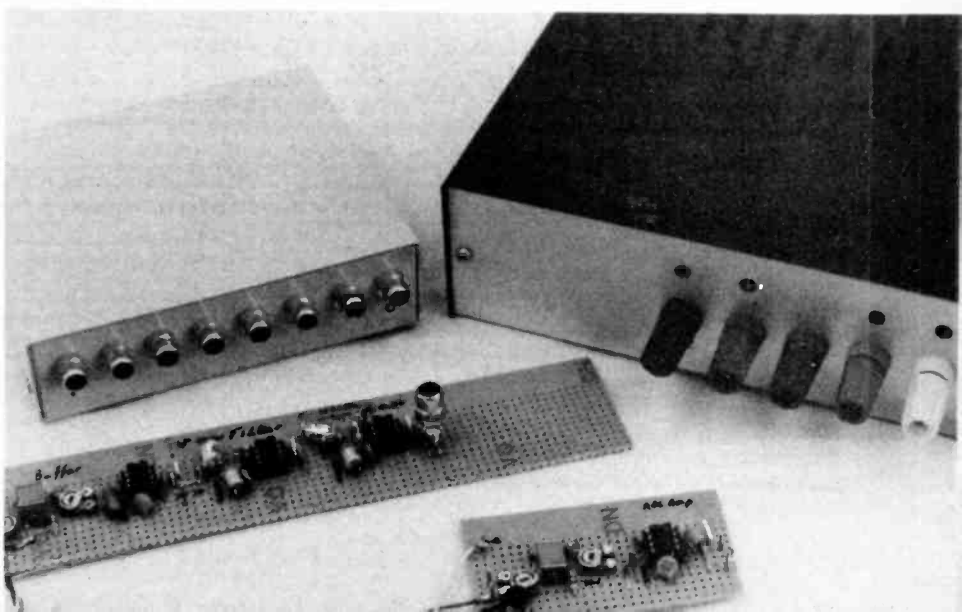


Fig. 4. Combined circuit diagrams for the ADC Buffer and Filter (4th order, cutoff frequency 16kHz).

Buffer

The gain of the Buffer is one, i.e. the output signal is of the same amplitude as the input signal. Preset potentiometer VR1 can be used to attenuate the input signal if it is larger than required. The filter unit has a gain of 2·6 (8·3dB), so if using the buffer with the filter VR1 should be adjusted so

Completed ADC, Linear Power Supply, Buffer/Filter board and Pre-amplifier board (foreground).



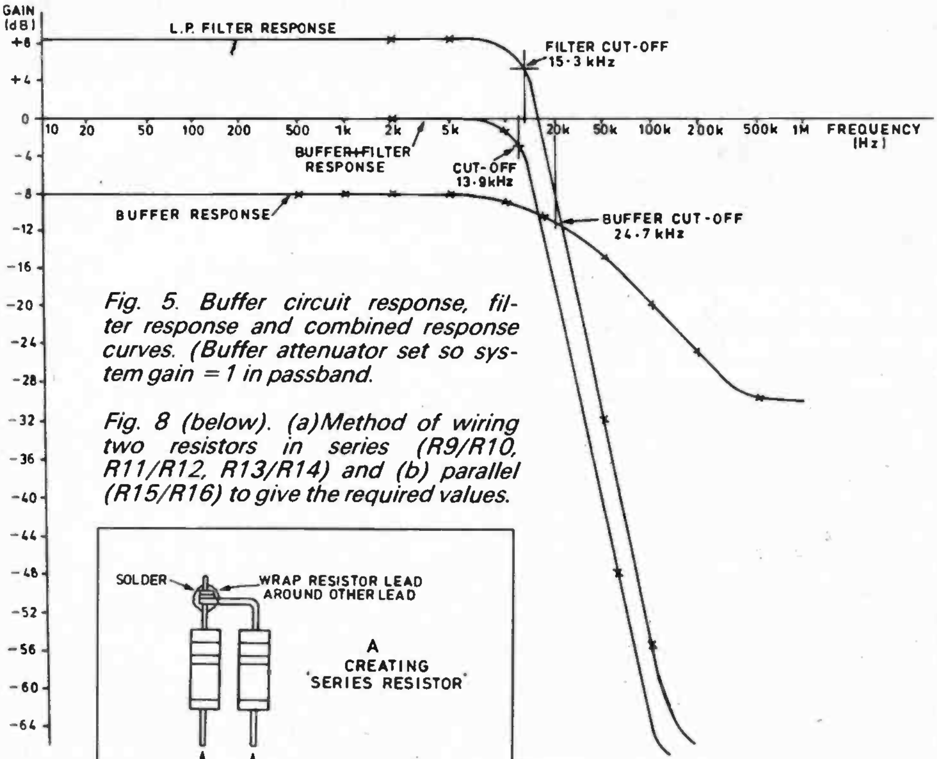
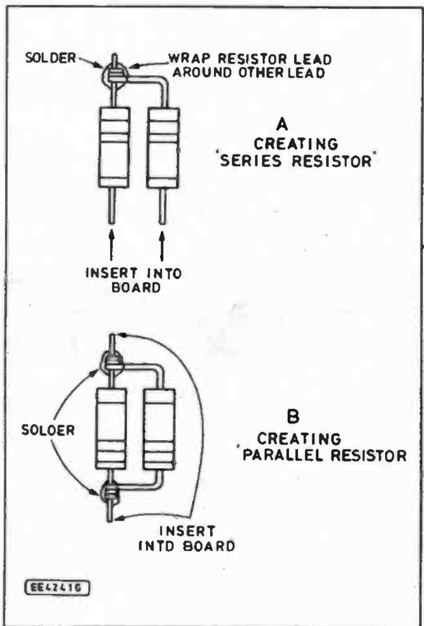


Fig. 5. Buffer circuit response, filter response and combined response curves. (Buffer attenuator set so system gain = 1 in passband.

Fig. 8 (below). (a)Method of wiring two resistors in series (R9/R10, R11/R12, R13/R14) and (b) parallel (R15/R16) to give the required values.



the combined filter and buffer has a gain of one.

Capacitor C1 removes any d.c. bias from the signal. The voltage from the slider of preset VR2 is added onto the signal. This is useful if, for example, the input signal is a sinewave which is oscillating between ±2·5V. If the buffer output is set to 2·5V (with no input signal), then when the input signal is applied, the output would be a sine wave oscillating between 0V and 5V (which can be inputted into the ADC).

Capacitor C12 filters off high frequencies. The diodes D1 and D2 are to protect the input from voltages outside the supply rails (±12V). Note that because of C1 this buffer will start to attenuate signals below ≃10Hz.

Filter

The Filter is an active 4th order low-pass Butterworth filter made of two cascaded 2nd order low-pass filters. The cutoff frequency of the amplifier (when the gain has fallen to 0·707 of the original value) is around 16kHz. The exact cutoff frequency will depend on the tolerance of the components used.

The actual cutoff frequency of the prototype was 15·3kHz which is within 5 per cent of the expected value. The frequency response of the prototype buffer, filter and their combined response is shown in Fig. 5.

With a fourth order filter the gain starts to fall at 24dB/octave (i.e. the gain is reduced to 0·0631 of its last value every time the frequency doubles) after the cutoff frequency.

One "unit" of voltage to the ADC is (5 volts/255) ≃ 20mV, so an input voltage of 1mV or 10mV would still give the value of 0. The highest sampling rate possible on the PCW is 200kHz, this means that all frequencies above 100kHz (Nyquist) should be reduced to a negligible value, i.e. below 20mV.

If a 100kHz sine wave with an amplitude of five volts is introduced to the input, it would have to be reduced to 0·004 of its original value to be negligible. A fourth order filter with a cutoff frequency of 16kHz achieves this as its gain has fallen to less than 0·001 by 100kHz.

CONSTRUCTION

The Buffer and Filter circuits can be easily constructed on stripboard. The Buffer component layout and breaks required in the underside copper tracks is shown in Fig. 6 and board details for the Filter in Fig. 7.

The component leads, jumpers etc. should be kept as short as possible to help prevent "stray" pickup. Some of the

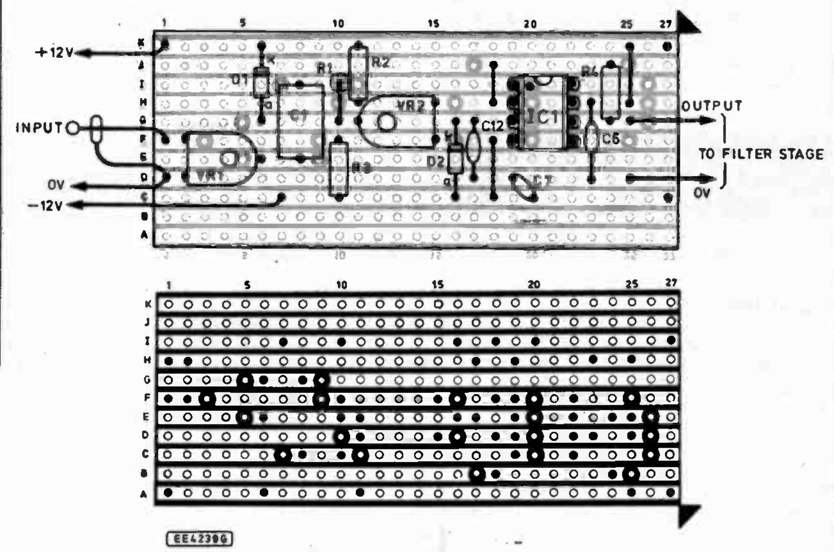


Fig. 6. Stripboard component layout and details of breaks required in the underside copper tracks of the Buffer board.

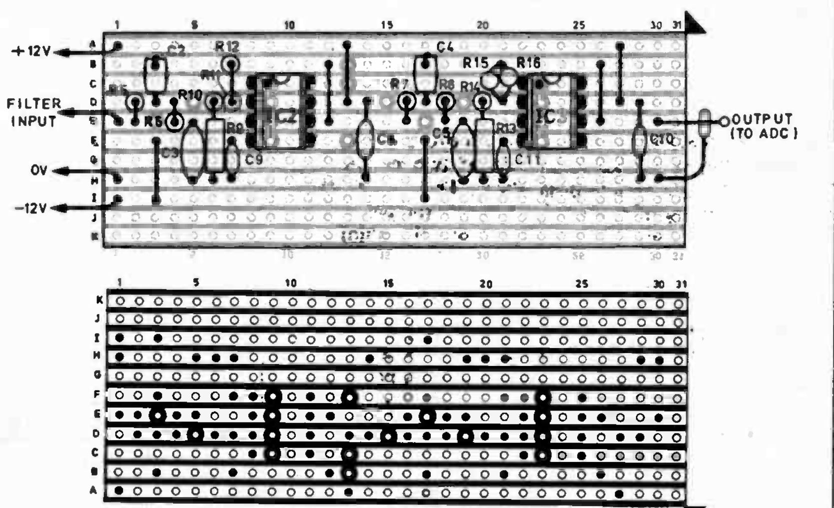
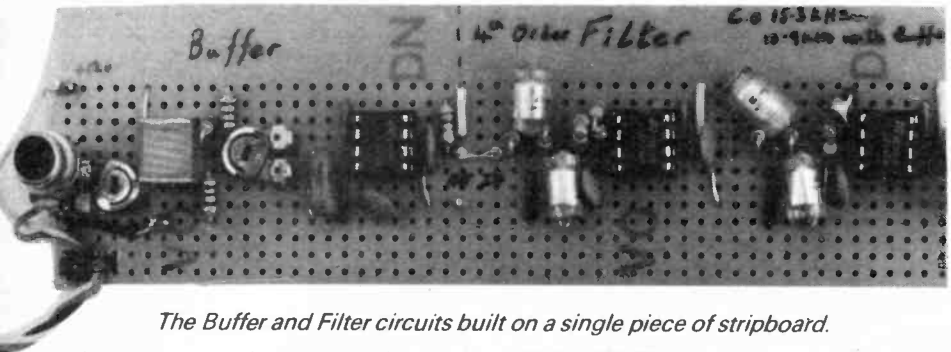


Fig. 7. Filter board component layout and details of underside breaks in the copper strips. In the prototype the Buffer and Filter circuits were combined on a single board – see photograph.



The Buffer and Filter circuits built on a single piece of stripboard.

resistor values required for the filter are non standard and so are made of series/parallel combinations, Fig. 8 shows how these can be made up to fit the layout.

SETTING UP

To set the gain of the whole system to "one", attach a 1kHz sinewave generator to the input of the buffer (connect the buffer output to the filter input if you have not already done so) and the filter output to an oscillloscope. Adjust VR1 until the input and output signals are the same size.

If you do not have access to an oscilloscope, set the d.c. output voltage to 0V by adjusting VR2, and use a 100Hz signal and an a.c. voltmeter. Adjust VR1 until the input and output voltages are equal.

The most useful value to set the bias to is probably 2·5V. To do this adjust preset VR2 until the (d.c.) output voltage of the unit is 2·5V, with the input unconnected.

IN USE

The Filter and Buffer Unit was designed for sampling audio signals, although it can be used for other purposes. The buffer input can be connected directly to the "Ear" or "Ext.Spk." output of most cassette players.

Set the bias to 2·5V as described above, if you have a 'scope connect it to the filter output and adjust the volume on the cassette player so that the output always remains within 0V to 5V. When this is done connect the output to the ADC. Otherwise start with the volume at minimum, and use the program described later, increasing the volume until it is at a reasonable level.

PRE-AMPLIFIER

If you wish to digitise smaller signals Fig. 9 shows a circuit diagram for a Single I.C. Pre-amplifier. The purpose of potentiometers VR1 and VR2 are the same as in

COMPONENTS

PRE-AMPLIFIER

Resistors

R1	1k
R2, R3	1M (2 off)
R4	10k
R5	47k

All 0·25W 5% carbon film

Potentiometer

VR1, VR2	470k enclosed carbon preset, lin. (2 off)

Capacitors

C1	1µ polyester layer
C2, C3	0µ1 ceramic (2 off)
C4	22p polystyrene

Semiconductors

D1, D2	BAT85 Schottky diode (2 off)
IC1	LF356N f.e.t.-input wideband op.amp

Miscellaneous

Stripboard 0·1in. matrix, size 10 strips x 30 holes; case to choice (optional); 8-pin d.i.l. socket; conectors; multistrand connecting wire; solder pins; solder, etc.

Approx cost guidance only **£5**

COMPONENTS

BUFFER/FILTER

See SHOP TALK Page

Resistors

R1	1k
R2, R3	1M (2 off)
R4 to R8	10k (5 off)
R9, R13	39k (2 off)
R10, R14	220 (2 off)
R11	1k2
R12	4k7
R15	82k
R16	120k

All 0·6W 1% metal film

Potentiometer

VR1	470k min. enclosed carbon preset, lin.
VR2	1M min. enclosed carbon preset, lin.

Capacitors

C1	1µ polyester layer
C2 to C5	1n polystyrene (±5% or better – 4 off)
C6 to C11	0µ1 ceramic (6 off)
C12	15p polystyrene

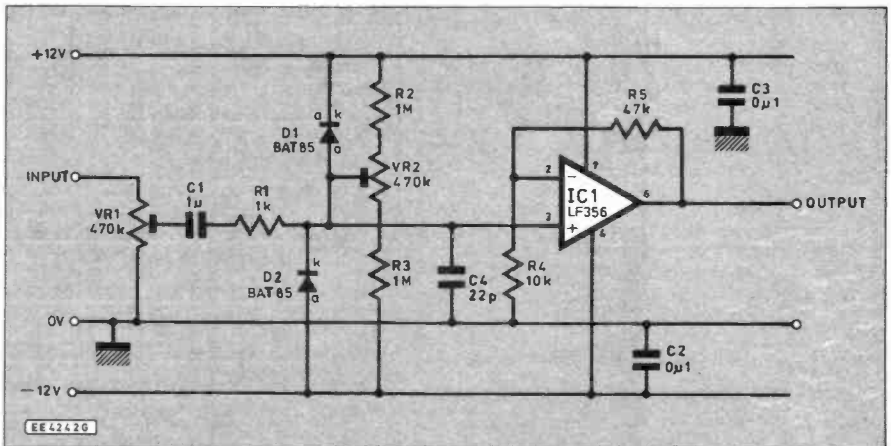
Semiconductors

D1, D2	BAT85 Schottky diode (2 off)
IC1, IC2, IC3	LF356N f.e.t.-input wideband op.amp (3 off)

Miscellaneous

Stripboard 0·1in. matrix, size 11 strips x 27 holes, and 11 strips x 31 holes; case to choice (optional); conectors; multistrand connecting wire; 8-pin d.i.l. socket (3 off); solder pins; solder, etc.

Approx cost guidance only **£8**



Circuit diagram for a simple Single I.C. Pre-amplifier (non-inverting).

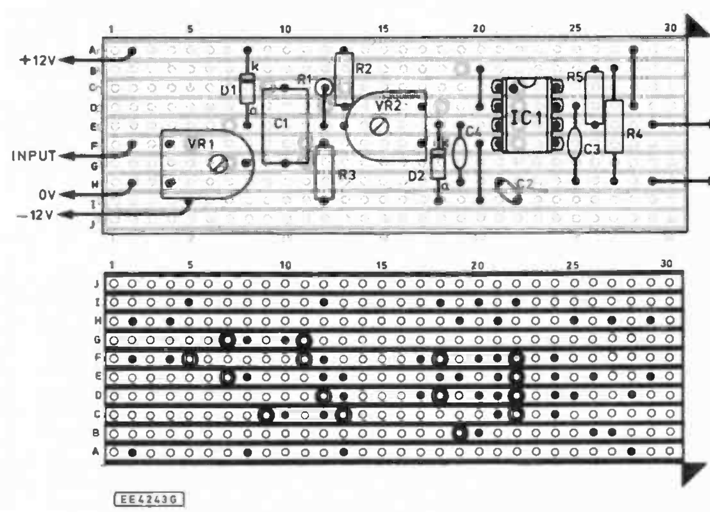


Fig. 10. Pre-Amplifier stripboard component layout and details of breaks required in the underside copper tracks.

the buffer circuit. The amplifier has a maximum gain of six. This circuit *cannot* be used to amplify d.c. signals due to capacitor C1.

The stripboard component layout is shown in Fig. 10. The construction details are the same as for the buffer.

DIGITAL SIGNAL PROCESSING

Volumes can be (and have been) written on the subject of digital signal processing so this is a very brief introduction.

Signals are digitised by taking discrete samples of a continous signal. Let the value of the first sample be (taken at time 0) x[0], the second taken at time T (the sampling period) be, x[1], ... at time nT be, x[n].

You now have an "array" of sampled values. These values can be processed in various ways. These digitised signals can be low, high, band pass (etc) filtered by using software routines. These processed signals can then be outputted to a DAC.

Software filters are often used as they are far more versatile than analogue filters. There are three basic building blocks that are used to make up software filters, these are shown in Fig. 11.

The time delay block delays a signal by 1 unit of time, if x[n] is input at time 1 then at time 2 the output will be x[n]. When inputting the array of sampled values the output is x[n − 1] when the input is x[n]. The multplier and summation blocks multiply or sum the inputs and output the result.

The PCW is not really fast enough for signal processing, as this is normally required to be done in "real time". But we have included a simple low-pass filter routine in the 'scope program described below.

The basic principle is shown in Fig. 12a. A sample, x[n], is feed into the filter, this is then added to the last value input into the filter, x[n − 1]. The result is then multiplied by 0·5 (output = 0·5 × (x[n] + x[n − 1])).

The output is the average value of the current sample and the last sample. This is called a two term moving averager and has a low-pass filter effect. Fig. 12b shows the effect it has on a signal, note that the amplitued of the "spike" is reduced more than the rest of the low frequency signal.

The filter implimented in the 'scope program uses four terms instead of two, as this has a more noticable effect. Fig. 13 shows a setup you can use to test the effect of the filter on sinewaves of different frequencies.

SCOPE PROGRAM

A simple Storage 'Scope program is shown in Listing. 2 and Fig. 14 shows some screen dumps from the program. The grid drawn on the screen is 500 ± 4µS per division horizontally and 1V per division vertically. Most of the program is written in machine code for speed. Some of these machine code routines can be called from basic.

Init: Sets up the screen. Must be called before other routines are used.

Scope: This is the main routine. When invoked a grid is drawn on the screen, it then takes 720 samples (one every 6µS) from the ADC and plots them on the screen. By default it does this 20 times, erasing the old line each time, and then returns to basic.

The number of scans can be altered by POKEing 'NoScans' with a value from 1 to 255 (0 will result in 256 scans). This routine calls the "Sample" routine to read in the data – see below.

Sample: Reads 720 samples (6µS period) from channel 0 of the ADC. The data can be accessed by PEEKing locations 'ADC data' to 'ADCdata' + 719.

Display1: This does the same as Display2 but erases the last plot displayed.

Display2: Plots the data stored by 'Sample' on the screen.

Grid: Plots a grid on the screen.

Hline(x1%,x2%,y%): High speed horizontal line drawing routine.

Vline(x%,y1%,y2%): High speed vertical line drawing routine. The X coordinates are in the range 0..719, and Y coordinates in the range 0..255.

MACHINE CODE PROGRAMS

To get the best performance from the ADC machine code subroutines can be written to be called from basic, or a stand alone program could be written.

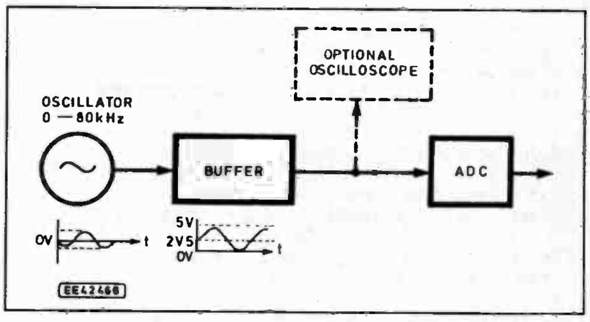


Fig. 13 (above right). Testing the effect of the software low bass filter.

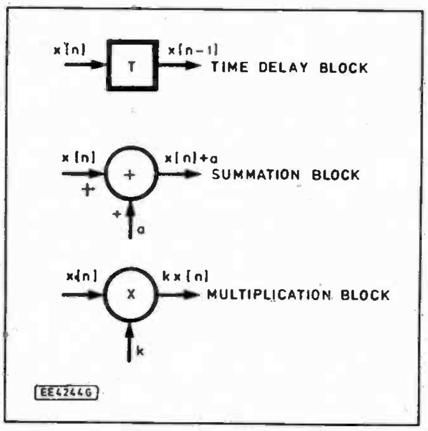


Fig. 11 (left). The three basic building blocks that are used to make up software filters.

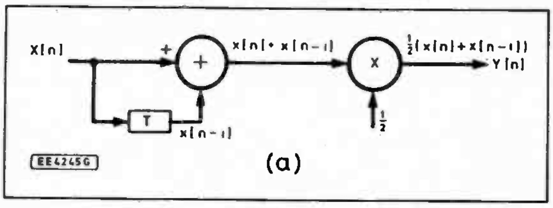
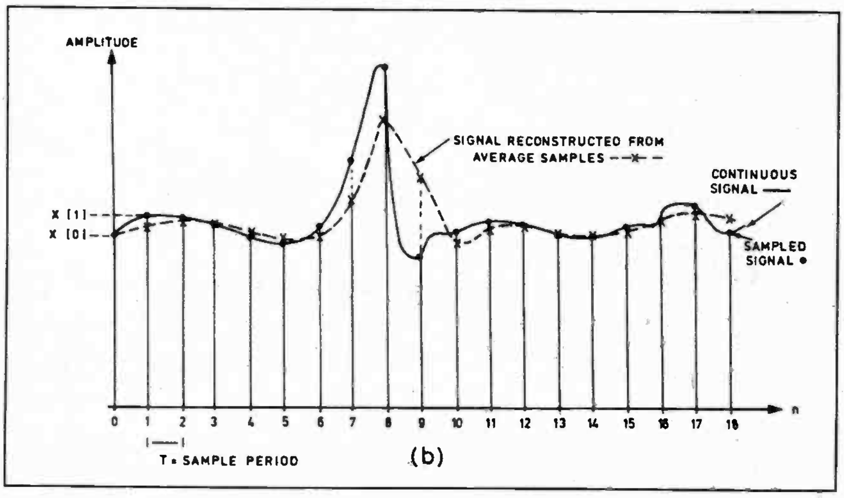


Fig. 12a (left). Simple low pass filter (two term moving averager).

Fig. 12b (below). Graph showing effect of two term moving averager.



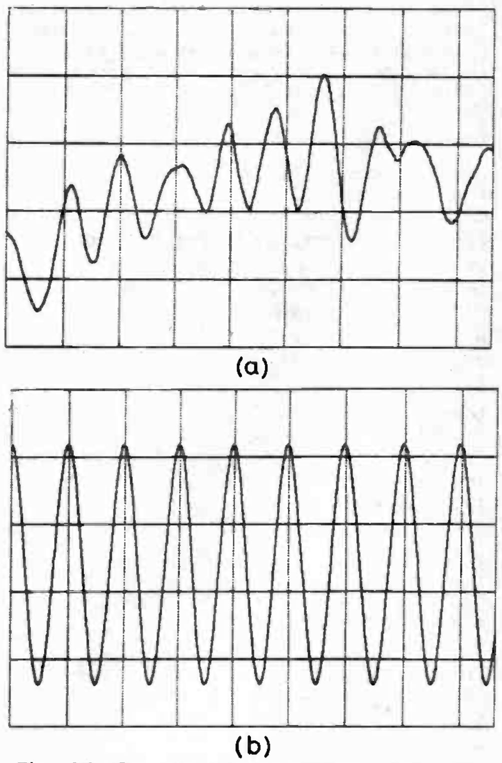


Fig. 14. Sample screen displays using the 'Scope program (0·5mS division "horizontal" and 1·0V division "vertical").
(a) screen dump from some sampled music.
(b) screen dump of sampled 2kHz sinewave.

Listing 2: Storage 'Scope Program

```
1 REM ****************************************************************
2 REM *      Fast 'Scope' Program for use with 8 channel ADC          *
3 REM *                      By J.M.Sharpe  (C) 1992                  *
4 REM ****************************************************************
5 :
10 MEMORY &HCFFF :CLEAR:GOSUB 40000
99  REM **************************PROGRAM*****************************
100 CALL Init :REM                                   Initialise screen
110 POKE NoScans,5 :CALL Scope :REM                         Do 5 scans
120 FOR n=ADCdata TO ADCdata+720 :REM            Low Pass Filter data
130  POKE n,(PEEK(n)+PEEK(n+1)+PEEK(n+2)+PEEK(n+3))/4
140 NEXT n
150 PRINT "Press a key to plot filtered signal";
160 WHILE INKEY$="":WEND :CALL Display1:REM       Plot filtered value
170 PRINT CHR$(27)"HPress a key to continue                  "
180 WHILE INKEY$="":WEND :CALL INIT :REM        Re-Initialise screen
210 CALL Grid      :REM                                     Draw Grid
220 CALL Sample    :REM                                  Take samples
230 CALL display2 :REM                                        Display
240 REM                           Loop forever, Sampling when key pressed
250 CALL sample:CALL display1:WHILE INKEY$="":WEND:GOTO 240
39997 :
39998 END
39999 REM ***********************SET UP MCODE*************************
40000 ad=&HE000     :NoScans=&HE017 :ADCdata=&HE4D8
40010 Scope=&HE000 :Grid=&HE061    :Init=&HE053     :Sample=&HE06A
40020 Hline=&HE0A0 :Vline=&HE08F   :Display1=&HE070 :Display2=&HE079
40025 PRINT CHR$(27)"E"CHR$(27)"HSETTING UP";
40030 FOR n=1 TO 75:PRINT "*";
40040 READ c$,s:ck=0
40050 FOR x=1 TO 32 STEP 2
40060 c=VAL("&H"+MID$(c$,x,2)):POKE ad,c:ck=ck+c:ad=ad+1
40070 NEXT x:IF ck<>s THEN PRINT"ERROR IN DATA - LINE:";n:STOP
40080 NEXT n
40090 RETURN
40200 DATA "CD25E0CD5AE4CDB4E021DAE711DBE701",2548
40210 DATA "00033600EDB00614C5CD04E1CD14E1C1",1770
40220 DATA "10F6C340E0F3DD223EE0DDE1FDE5ED73",2809
40230 DATA "2FEB312DEB3E81D3F13CD3F2DDE90000",2221
40240 DATA "ED7B2FEBDD2A3EE03E85D3F13CD3F2FD",2604
40250 DATA "E1FBC9CD5AFCC200CD25E0CD5AE4C340",2666
40260 DATA "E0CD25E0CDB4E0C340E0F3CD04E1FBC9",2911
40270 DATA "CD25E0CD14E1C340E0CD25E021DAE711",2364
40280 DATA "DBE70100033600EDB0CD14E1C340E00A",1864
40290 DATA "471A4F7E23666FCD25E0CDB7E1C340E0",2112
40300 DATA "0A4F7E23666FEB7E23666FCD25E0CD7A",1865
40310 DATA "E3C340E03EAA0601CDE5E23EAA0601CD",2053
40320 DATA "1BE40E000606C521000011CF02CD7AE3",1291
40330 DATA "C13E33814F10EF210000060AFD21A5E1",1494
40340 DATA "C506FF0E00CDB7E1FD6E00FD6601FD23",2092
40350 DATA "FD23C110EB3EFF0600CDE5E23EFF0600",2038
40360 DATA "CD1BE4C90EB006D221D7E4EDB200EDB2",2373
40370 DATA "00EDB2C9FD21D8E4DD21DAE721000006",2088
40380 DATA "00C5DDE5E5DD4E00DD4601CDB7E1E1E5",2534
40390 DATA "FD4E00FD4601CDB7E1E1DDE1C1FD7E00",2511
40400 DATA "DD7700DD2323FD2310D7C5DDE5E5DD4E",2325
40410 DATA "00DD4601CDB7E1E1E5FD4E00FD4601CD",2219
40420 DATA "B7E1E1DDE1C1FD7E00DD7700DD2323FD",2535
40430 DATA "2310D706CFC5DDE5E5DD4E00DD4601CD",2151
40440 DATA "B7E1E1E5FD4E00FD4601CDB7E1E1DDE1",2801
40450 DATA "C1FD7E00DD7700DD2323FD2310D7FD7E",2101
40460 DATA "00DD7700C95300A700FA004D01A101F4",1525
40470 DATA "0147029B02CF023EFF90473EFF91B8D2",1828
40480 DATA "C5E14F7841903C575DCB3CCB1DCB3CCB",2031
40490 DATA "1DCB3CCB1D653E07A0CB38CB38CB3868",1735
40500 DATA "E5CD42E406004F09E5697982DAF8E1B9",2283
40510 DATA "CAF8E1FE09DA9FE23E07A33C47AF6737",2237
40520 DATA "1F10FD5F7AD608855729290115E209E3",1525
40530 DATA "DDE17BDDE9AE77237BAE77237BAE7723",2253
40540 DATA "7BAE77237BAE77237BAE77237BAE7723",1804
40550 DATA "7BAE773E0742A257CB38CB38CB38CA69",1884
40560 DATA "E2E12CE5CD42E47BAE77237BAE77237B",2248
40570 DATA "AE77237BAE77237BAE77237BAE77237B",1804
40580 DATA "AE77237BAE777B10D8E12CCD42E41415",1908
40590 DATA "C87BAE7715C8237BAE7715C8237BAE77",1960
40600 DATA "15C8237BAE7715C8237BAE7715C8237B",1723
40610 DATA "AE7715C8237BAE7715C8237BAE77C93E",1900
40620 DATA "07A33C47AF67371F10FD5F29092901B8",1305
40630 DATA "E209E3DDE1C1DDE9AE7715C8237BAE77",2520
40640 DATA "15C8237BAE7715C8237BAE7715C8237B",1723
40650 DATA "AE7715C8237BAE7715C8237BAE7715C8",1858
40660 DATA "237BAE77C9DDE5FDE5D9E5D5D9C54F21",2769
40670 DATA "79E37EA9714F06082149E2DD2116E2FD",1936
40680 DATA "2173E2110400D911060021B9E2D9CB21",1532
40690 DATA "D22DE33E77AE773E77DDAE00DD77003E",1934
40700 DATA "77FDAE00FD7700D93E77AE77D919DD19",2097
40710 DATA "D919FD19D910D7C13EAE0520043EB618",1706
40720 DATA "050520023EA606082148E2DD2115E2FD",1371
40730 DATA "2172E2110400D911060021B8E2D977DD",1634
40740 DATA "7700FD7700D97719FD19D919DD1910EE",1872
40750 DATA "D9D1E1D9FDE1DDE1C9FF3EFF914FE5AF",3193
40760 DATA "ED52E1F287E3EBAFED5223EBE5CB3CCB",2842
40770 DATA "1DCB3CCB1DCB3CCB1D653E07A1CB39CB",1813
40780 DATA "39CB3969CD42E406004F09C13E07A1F5",1683
40790 DATA "1514C2BEE383DABEE3FE08DA07E4F1CA",2576
40800 DATA "DBE3474F3EFFCB3F10FCE6FFAE77EB3E",2522
40810 DATA "08914FAF47ED42EB0E08094B43CB3ACB",1653
40820 DATA "18CB3ACB18CB3ACB180504CAF8E31108",1711
40830 DATA "003EFFAE771910F93E07A1C847AF371F",1662
40840 DATA "10FCE6FFAE77C943371F10FCC10504CA",2072
40850 DATA "16E4CB3F10FCE6FFAE77C932CBE332F2",2535
40860 DATA "E33203E43217E43EAE0520043EB61805",1359
40870 DATA "0520023EA632CCE332F3E33204E43218",1624
40880 DATA "E4C9F5D5E52600291195E4195E2356E1",2054
40890 DATA "6C260029292919D1F1C9F5C5D5E51100",1846
40900 DATA "002195E4D5E5CD7AE4E173237223D13E",2202
40910 DATA "08835F30EFE1D1C1F1C926006B291100",1793
40920 DATA "B6195E23567BE6076F7B175F7A17577B",1489
40930 DATA "E6F0B55FC9990000000000000000000000",1100
40940 DATA "00000000000000000000000000000000",0
40950 DATA "00000000000000000000000000000000",0
```

References to assembly language commands below assume you are using an assembler which uses Zilog Z80 mnemonics. Assemblers of this type are widely available in the Public Domain, and from other suppliers. MAC supplied with the PCW uses 8080 mnemonics. The hex values for the commands are given in most Z80 books, which can be directly entered into SID.

TIMINGS

The main reason for using machine code is speed. The amount of time instructions take to execute are listed in most Z80 books. The PCW inserts a "wait state", a delay of one clock cycle (0.25μS), for every memory access. So the timings given need 0·25μS adding for each memory access, e.g.

Mnemonic	Hex	μS@ 4Mhz	μS on PCW	Notes
INC r	3C	1·00	1·25	
NOP	00	1·00	1·25	
LD r, (HL)	7E	1.75	2·25	
INIR	EDB2	5·25	6·00	B≠0
INIR	EDB2	4·00	4·75	B=0
INI	EDA2	4·00	4·75	
IN A, (n)	DB n	2·75	3·25	

Note that LD r, (HL) has 0·5μS added to the timing, 0.25μS to fetch the opcode and the other 0.25μS to fetch the contents of memory location HL.

The fastest way to input a large amount of data into memory is to use a long list of INI's, this reads a value from the port held in register C, stores the value in memory location HL, and then decrements B and increments HL. The fastest way to read in two channels is to use INI's interleaved with EXX instructions. EXX switches to other register set, this takes 1·25μS. In this way two "arrays" in memory can be filled with data, e.g.

```
LD HL,400  ;START OF FIRST ARRAY
LD C,176   ;ADC CHANNEL 0
EXX        ;OTHER REGISTER SET
LD HL,800  ;START OF FIRST ARRAY
LD C,177   ;ADC CHANNEL 1
INI        ;START CONVERSION ON
            CHANNEL 1
EXX
INI        ;GET RESULT,STORE,
            AND START CONVER-
            SION ON CHANNEL 0.
EXX
INI        ;GET RESULT,STORE,
            AND START CONVER-
            SION ON CHANNEL 1
... etc ...
```

This is fast but uses up a large amount of memory. The INIR instruction is useful for sampling one channel (this is used in the 'scope program). This is similar to the LDIR instruction but copies the values from a port (in reg. C) instead of from memory. A maximum of 256 values can be read at once (set B=0). To get more than this NOP's can be inserted between INIR instructions to equalise the timing, e.g.

```
LD C,176   ;CHANNEL 0
LD B,0     ;256 INPUTS
INIR       ;READ 256 VALUES
            WITH 6μS PERIOD
NOP        ;1·25μS DELAY
INIR       ;READ ANOTHER 256
            VALUES
... etc ...
```

INTERRUPTS

Note that all of the above timing assume that interrupts have been turned off (using DI). The PCW is interrupted 50 times per second, leaving these switched on will really mess up the timings.

Next month: Linear Power Supply for the 8-Channel ADC.