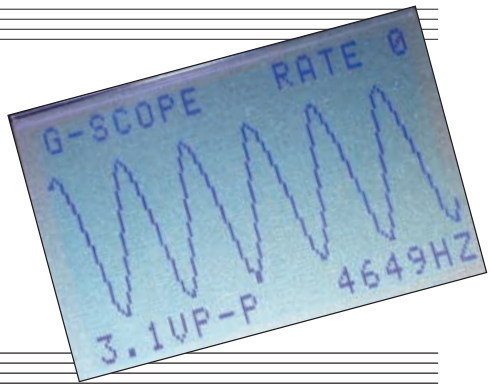
Constructional Project

PIC GRAPHICS L.C.D. SCOPE



JOHN BECKER

Long awaited, a PIC and graphics l.c.d. design for monitoring audio frequency signals has arrived!

OUR February '01 issue of *EPE* contained a supplement in which the author's researches into *Using Graphics L.C.D.s* were published. In this demonstration (nay, *semi-tutorial*) article, various programming routines were illustrated in conjunction with a specially designed printed circuit board.

Demos 11 and 12, many of you will recall, showed the results of experiments with creating waveform displays on the screen. As the text said, these were created preparatory to designing the *PIC Graphics L.C.D. Scope* (*G-Scope*) described here.

No doubt *EPE* will publish other graphics l.c.d. designs in the future, the "dam having been broken", so to speak. That is, the mysteries of using such devices have been revealed (and well hidden they were previously!). The *G-Scope*, though, is such an obvious application for them, that its design is inevitably the first to appear.

MULTI-SCOPING

G-Scope is, in fact, another addition to the widening family of oscilloscope-type constructional projects published in *EPE* over the last few years.

The *EPE Virtual Scope* (*V-Scope*) of Jan-Feb '98 was the most sophisticated of this family, interfacing a complex dual-channel hardware unit to a PC-compatible computer, with a frequency maximum in excess of 10MHz.

Micro-PICscope (M-Scope) followed in April '00, in which a stand-alone unit used an ordinary alphanumerics l.c.d. to display single waveforms on eight of its character cells. It was intended principally as a visual signal tracer, catering for frequencies in the audio range, up to around 15kHz or so.

October '00 saw the publishing of the *PIC Dual-Chan Virtual Scope* (*PIC V-Scope*) which used a PIC-controlled hardware unit to interface to a PC. It was a considerably easier unit to build than the original *V-Scope* and used a cut-down version of the same PC software. The frequency range was nominally audio, although this extended well above and below the human hearing range.

G-SCOPE

The *G-Scope* described here is a self-contained single channel unit, also catering nominally for the audio range. Like *M-Scope*, it is a stand-alone design intended for visually monitoring signals, but having a greater resolution of the signal amplitude display. Whereas *M-Scope* used a display area of 8 × 40 pixels, *G-Scope*'s graphics screen has a pixel density of 64 × 128 (vertical × horizontal).

Like *M-Scope*, it also displays frequency and signal amplitude factors as alphanumeric text lines.

G-Scope also provides sync (waveform synchronisation stability) on/off selection, frequency/voltage monitoring on/off and a choice of three sampling rates. The lowest sampling rate allows sub-Hertz signals to be slowly traced on screen while they occur.

The signal source can be a.c. or d.c. and waveforms up to 5V peak-to-peak can be input without external attenuation. A simple pre-amp stage can be switched to provide ×1 or ×10 amplification. The circuit does not permit negative d.c. voltages to be input.

CIRCUIT DETAILS

The *G-Scope* circuit diagram in Fig.1 is closely similar to that for *M-Scope*. One principal difference is the l.c.d. type used, in this case a Powertip PG12684 graphics display (X2) – i.e. the same device discussed when examining the use of graphics displays in the Feb '01 Special Supplement.

The second significant difference is that the display requires a negative voltage to control the screen contrast. This is provided by the voltage inverter IC4. It is powered at +5V, as set by the positive voltage regulator IC3, and outputs –5V from pin 5. Capacitor C8 sets the frequency at which the inverter operates and C9 smooths the output voltage.

Preset potentiometer VR1 is then used as a variable resistor to set the current flowing through the l.c.d.'s pin 4, so controlling the screen contrast.

A further difference is that the display is controlled by PORTD of the PIC16F877 microcontroller (IC2), instead of the previous PORTB. This now allows PORTB to be used for the mode switches (S4 to S6), taking advantage of this port's internal pull-up resistors in order to use two pushbutton switches instead of three s.p.d.t. toggles.

The signal to be monitored is input via socket SK1 to the gain-selecting switch S2. At this point, the signal routing is switchable via resistors R1 or R2. The gain is determined by the value of the selected input resistor in relation to that of the feedback resistor (R3) in the inverting op.amp circuit around IC1a.

Following R1/R2, switch S3 selects for d.c. or a.c. coupling, the latter routing being through capacitor C4. Resistors R4 and R5 provide mid-rail bias (+2·5V) to the op.amp's non-inverting input (pin 3).

From IC1a, the signal is fed to IC2 pin RA0, which is configured as an analogue-to-digital converter (ADC) input.

The PIC is operated at 5MHz, the maximum in keeping with the highest sampling rate that the PIC's ADC can handle.

As usual with this author's designs, the PIC can be programmed on-board, using a programmer such as *PIC Toolkit Mk2* (May-June '99). The programming connections are via terminal pin block TB2. Diode D1 and resistor R6 prevent the programming voltages from disturbing the rest of the circuit.

Resistor R7 holds l.c.d. pin $\overline{CE}$ (chip enable) high to prevent random display detail being created on screen while the PIC is being programmed.

Pre-programmed PICs are available should you not have a PIC programmer. This month's *Shoptalk* page gives details of this, and of obtaining the software (free) via the *EPE* web site, or from the *EPE* Editorial office on 3·5-inch disk (for which a nominal handling charge applies).

The unit may be powered by a 9V battery, or from an existing 7V to 12V d.c. mains operated power supply (e.g. mains adaptor). Current consumption is only a few milliamps. Note that 9V PP3 batteries are typically rated at between 100mA/hr (NiCad) and 500mA/hr (alkaline) and are not suited to long term powering of the unit.

CONSTRUCTION

Component positioning and track layout details for the *G-Scope* printed circuit board are shown in Fig.2. This board is available from the *EPE PCB Service*, code 300.

COMPONENTS

See SHOP TALK page

Resistors

R1, R4, R5, R7	10k (4 off)
R2, R3	100k (2 off)
R6	1k

All 0·25W 5% or better.

Potentiometer

VR1	22k (or 25k) min. round preset, horiz.

Capacitors

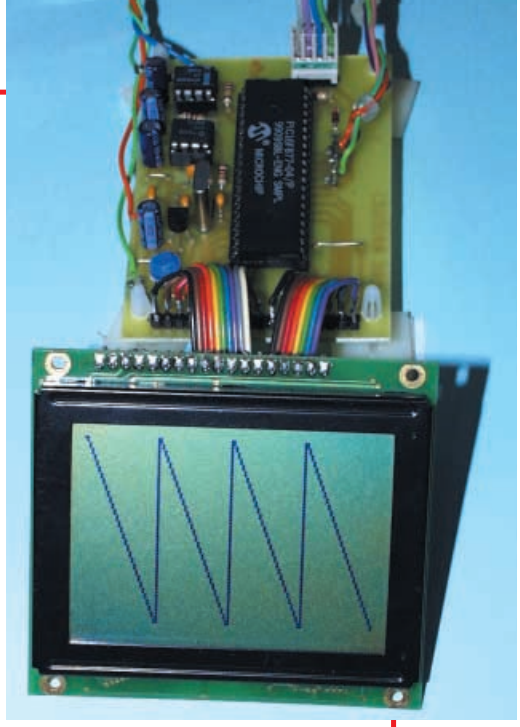
C1, C4, C5, C8, C9	22μ radial elect. 10V (5 off)
C2, C3	100n ceramic disc, 5mm pitch (2 off)
C6, C7	10p ceramic disc, 5mm pitch (2 off)

Semiconductors

D1	1N4148 signal diode
IC1	MAX492 dual op.amp (see text)
IC2	PIC16F877-20P (20MHz version), pre-programmed (see text)
IC3	78L05 +5V 100mA voltage regulator
IC4	7660 negative voltage converter

Miscellaneous

X1	5MHz crystal
X2	PG12864 graphics l.c.d. module (see text)
S1 to S3, S5	s.p.d.t. min. toggle switch (4 off)
S4, S6	min. s.p. push-to-make switch(2 off)



Printed circuit board, available from the *EPE PCB Service*, code 300; plastic case 190mm x 110mm x 60mm (see text); p.c.b. supports, self-adhesive (4 off); 1mm pin-header terminal strips; 80pin d.i.l. socket (2 off); 40-pin d.i.l. socket; mounting nuts and bolts to suit l.c.d.; connecting wire; solder, etc.

Approx. Cost Guidance Only **£55** ***excluding case.***

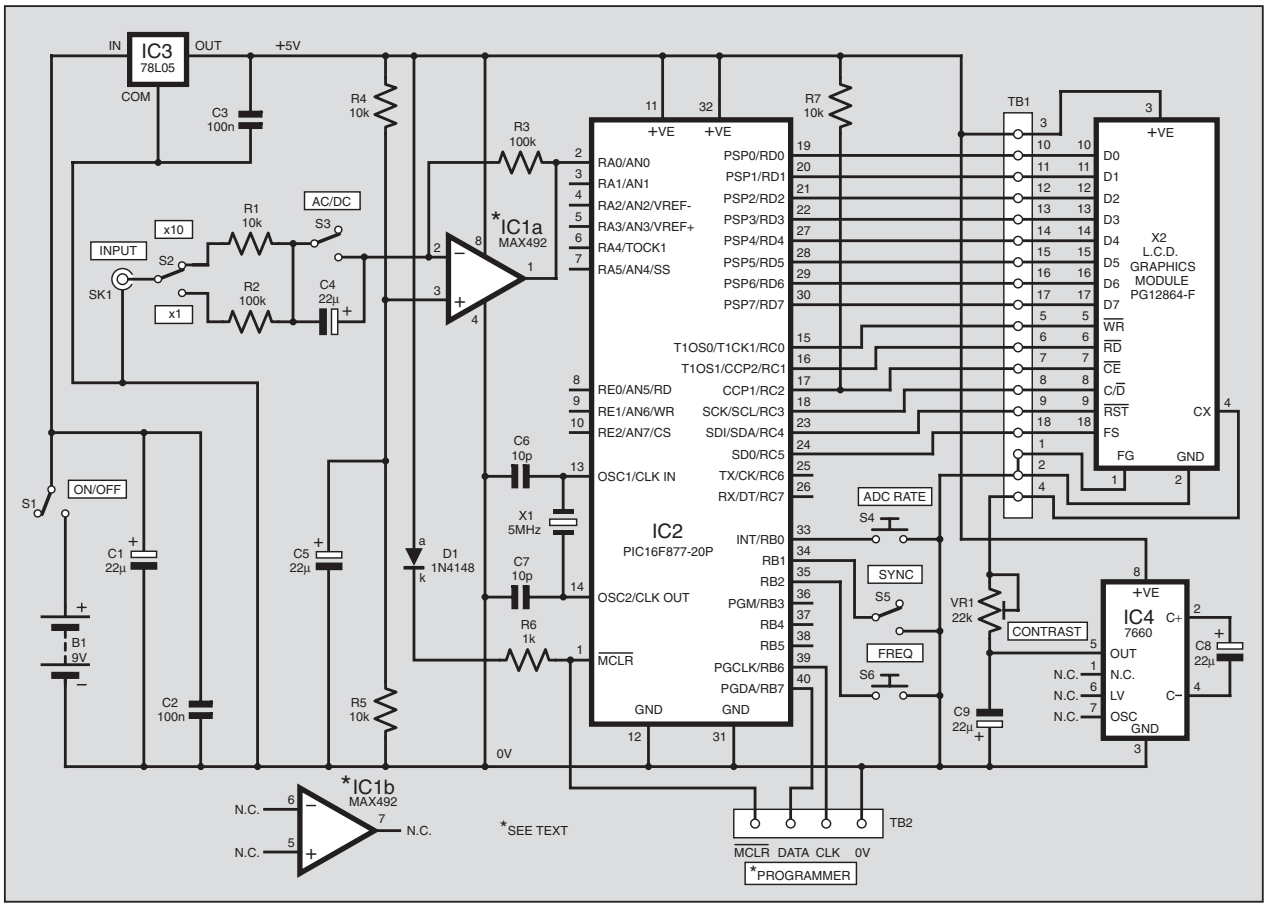


Fig.1. Complete circuit diagram for the PIC Graphics L.C.D. Scope.

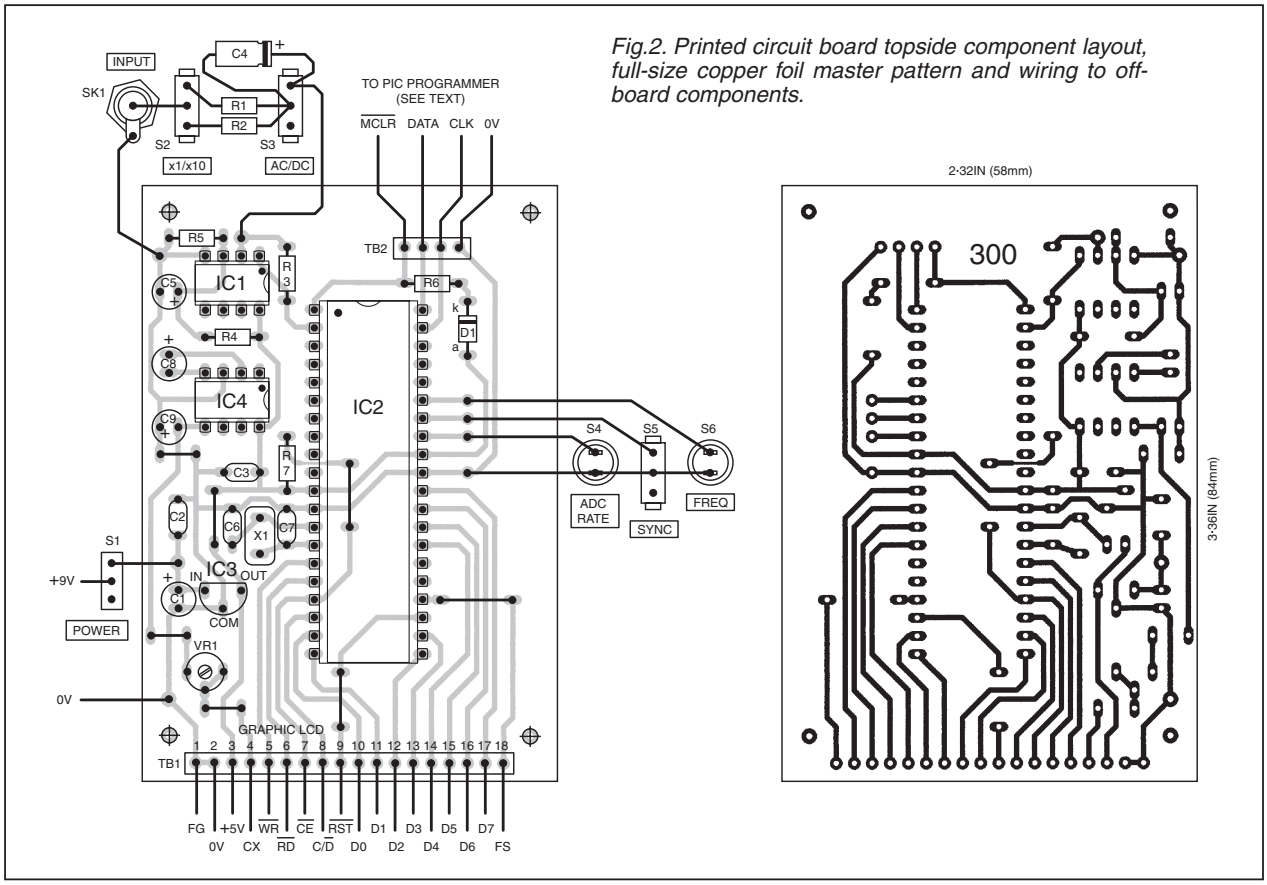


Fig.2. Printed circuit board topside component layout, full-size copper foil master pattern and wiring to off-board components.

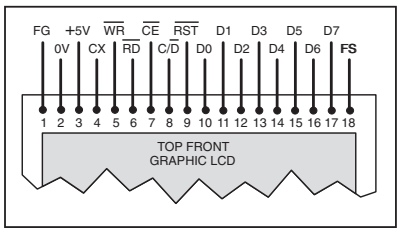


Fig.3. Pinout details from the l.c.d. to the circuit board.

Assemble in order of component size, starting with link wires, d.i.l. sockets (IC1, IC2 and IC4), and then upwards in ascending order. The d.i.l. i.c.s should not be inserted until after the board assembly and voltage output from IC3 have been fully checked.

Note that resistors R1, R2 and capacitor C4 are hard-wired between switches S2 and S3, which are mounted on the front panel, along with S4 to S6.

The p.c.b. pinout connections for the l.c.d. are in the same order as those on the l.c.d. module itself (see Fig.3).

Those of you who purchased a graphics l.c.d. in connection with the "Using them" article will be able to interchange it between the two units if you used a connector then.

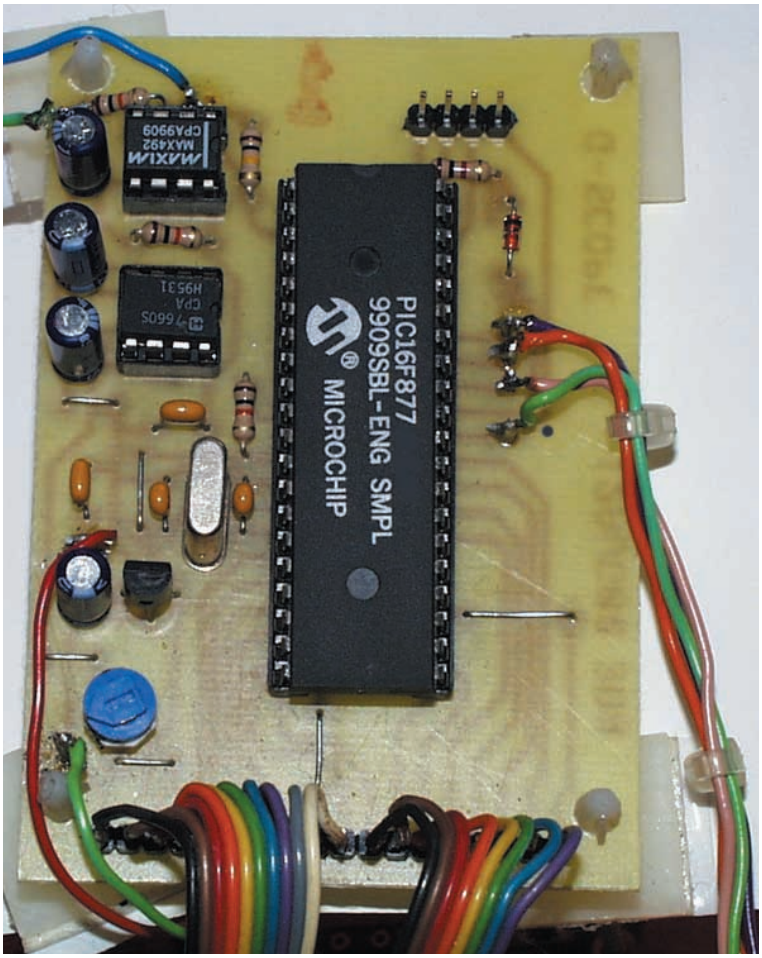
Do not connect the l.c.d. until you know that +5V and –5V are being correctly delivered by IC3 and IC4, respectively.

ENCLOSURE

The author used a suitable graphics l.c.d. viewing aperture in a case. He had, however, come very close to rebelling against this and almost used a case having a see-through lid, mounting the display immediately below it!

For ease of screen viewing the l.c.d. should face upwards in the case. This will allow the maximum amount of light to fall on its transreflective face. Viewing from the side could cause difficulties in a poorly lit workshop. It would be feasible, though, for a back-lit version (somewhat more expensive) to be used sideways in a different style of case.

The author cannot advise on back-lit types, other than to comment that many use internal l.e.d.s as the illumination source and thus probably require a fairly robust power source. Their data sheets should be consulted on this point.



Component layout on the completed prototype circuit board.

GREAT EXPECTATIONS

Having cracked the code structure for graphics l.c.d.s, the author had great expectations of not having to do much further programming work in respect of this *G-Scope*. He had, after all, already written the software for the seemingly similar *M-Scope*. It was, then, just a matter of a few changes in order to suit the needs of *G-Scope*.

But, *the best laid progs o' mice an' men gang aft a-gley*, almost said a certain Scot a wee two and half centuries ago!

Well, er, yes. While some routines *were* almost transported to *G-Scope* as library items, from both *M-Scope* and the L.C.D. Demo progs, the integration was considerably more complex than had first been anticipated.

The principle area of complexity was with the considerably greater quantity of data to be processed by *G-Scope* in comparison to *M-Scope*. The latter only needs 64 samples to be acquired and stored for intermediate processing. *G-Scope* needs 128.

In addition, *M-Scope* has only eight vertical screen positions to be filled or cleared. *G-Scope*, though, has 64 – a significant difference that required an investigative interruption of program development.

DATA BANKS

In normal use, the PIC16F877 has 96 bytes available for data storage. Not enough in which to store and process the sampled 128 data bytes for output to the l.c.d., let alone allow for the many other bytes needed for a variety of essential processes.

It was necessary, therefore, to bring the PIC16F877's additional banks of memory into play. Doing so required research into this aspect of the said PIC (and its other family members of the PIC16F87x series). This resulted in the *PIC16F87x Extended Memory Use* article which will be published next month, and to which you are referred for more information on this useful feature. Such research created quite a detour in the process of *G-Scope* completion.

Basically, members of the PIC16F87x family have four banks (pages) of data memory available, with a total capacity of between 192 and 368 bytes, depending on the device type. Any of this memory can be used in any program, but you have to keep a few wits about you in order to keep the banks correctly allocated, especially as some bytes have intentional joint-access between the banks.

To sum up the memory allocation for *G-Scope*:

- generally accessed variables are held in Bank 0
- the 128-byte ADC "recording memory" is split as 64 bytes in each of Bank 0 and Bank 1
- Bank 2 is used for the data compilation sent to the l.c.d. as graphics (waveform) drawing information
- Bank 3 is allocated for the variables used in decimalisation (from binary) of frequency and amplitude values prior to their screen display as text characters.

This leaves some data memory unused, but insufficient for two-channel's worth of signals to be processed, consequently *G-Scope* has had to be designed as a single-channel unit.

The software source code listing is "commented" with brief notes on the memory and bank use. For a fuller understanding of multiple bank use, though, see next month's article on the subject.

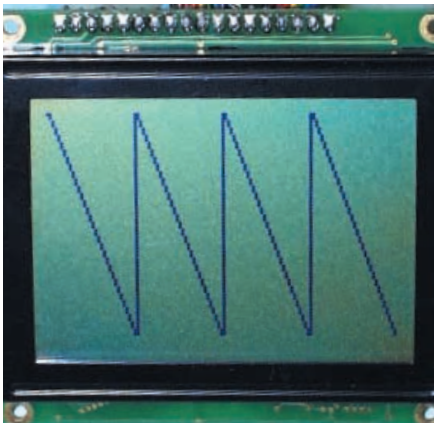
PROGRAM BASICS

In common with the *Using Graphics L.C.D.s* demo software, *G-Scope* uses PIC PORTD for l.c.d. data input/output, and PORTC for its command control.

Following the usual basic initialisation of program variables, an l.c.d. setup routine is called. In this, eight subroutines are called which, to all intents and purposes, are direct copies of those discussed in the L.C.D.s article. A ninth routine (also of the same origin) outputs tabled text data to the screen.

The body of the program starts at label MAIN. Here the choice of whether to monitor with or without synchronisation is checked, according to the status of switch S5.

If sync is needed, three subroutines (commencing with WAITS1) examine the input signal and wait for it to doubly cross a "trigger window" before progressing.



Graphics display module.

SAMPLES

On acquisition of the sync-trigger flag, or switched command to bypass sync, at label SAMPLE1, 128 bytes of input signal data are sampled, converted from analogue to digital, and stored in the memory bytes allocated, 64 in Bank 0 and 64 in Bank 1. This process is completed as rapidly as the PIC's internal ADC allows (also see later).

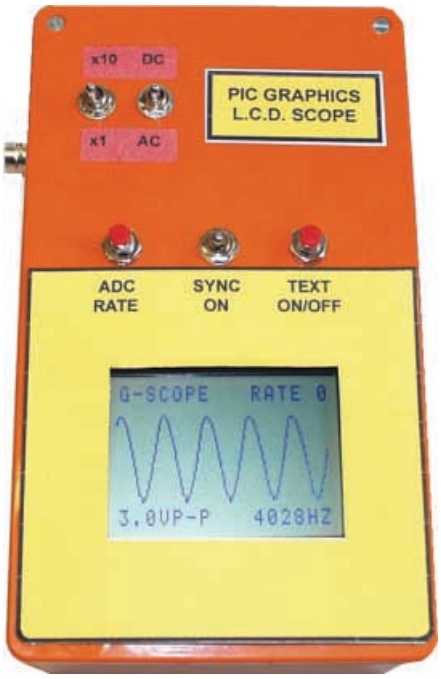
Although the PIC's ADC offers 10-bit sampling, only the upper eight sample bits are used, the lower two being ignored.

The 8-bit stored data is subsequently divided by four to limit the maximum value to 64, this figure being the same number of vertical pixels on the l.c.d. screen.

Each resulting data byte value now represents the actual line on which the data is to be shown as part of the overall signal trace. Its horizontal position is determined by which sample number it is in the 128-sample batch.

COORDINATES

Drawing the screen line at the correct horizontal/vertical address, though, is complicated by having to clear any previous data from that same region. Failing to do so would cause the screen to become rapidly filled with a congestion of lines.



The clearing process is aided by keeping track of each sample's value in relation that of the previous one. Suppose, for instance, that the first sample at the start of the screen trace has a value of 32. The time axis (Y) for that sample is at screen column zero (Y = 0). A sample value of 32 requires that pixel 32, counting upwards from the bottom of the screen, should be seen as active, i.e. at location X32/Y0.

However, data from the previous batch of samples is displayed somewhere in the Y0 column. The software cannot be told where it is since there is insufficient memory available in order to record the coordinates for 128 previous samples.

Because of this, all pixels of column Y0 have to be cleared in order to ensure that the previous data is removed. Only then can the pixel for the new value be activated at X32/Y0.

Sample two, now let's suppose, has a value of 40. Its coordinates are thus X40/Y1 (next step along the time axis).

We know that a "real" oscilloscope draws a constant trace between each position up and along its cathode ray tube screen. It is hence necessary to try to simulate a similar situation on the l.c.d. screen. Consequently a series of pixels between X32/Y0 and X40/Y1 has to be activated, and again the previous data cleared from that column.

It is wasteful of processing time to clear a full column on each step along the time axis. It is better to clear only those pixels above and below those that need to be active.

The software thus clears the lowest value pixels in the column, sets those required, and then clears those above. Before this happens, though, the software has to compare the preceding and current values and ascertain which is the lowest.

Then the software uses the following three sub-routines:

1. Clear pixels X0/Y1 to XL/Y1
2. Set pixels (XL + 1)/Y1 to XH/Y1
3. Clear pixels (XH + 1)/Y1 to X63/Y1

where XL is the lowest value, XH is the highest, and X63 is the top of the l.c.d. screen.

Obviously, various programming intercepts have to be included to cater for such situations as XL = XH, XL = 0, XH = 63, etc.

In order to use the same routines for each of the 128 time axis columns, prior to entering the controlling loop the program sets XL = XH = the value of the first sample, which is destined for column Y0.

Whilst it is easy to write to individual pixels on the l.c.d. graphics screen, it is far quicker to compile the data for eight columns as a series of 64 8-bit bytes, and then write the 64 bytes to the screen. After which the next 8-column batch can be assembled similarly. The process is so fast that the eye does not notice that it is a stepped assembly and display taking place.

As said earlier, the raw sampled data is held in data memory Bank 0 and Bank 1, while the screen data is assembled in Bank 2.

AMPLITUDE ASSESSMENT

Having displayed a full 128-byte batch of data on screen, frequency and minimum-maximum amplitude values are assessed.

Amplitude min-max values are easily ascertained. First, two temporary variables, MIN and MAX are set so that MIN is greater than or equal to the *highest* value expected, and MAX is set to be less than or equal to the *minimum* value expected. In this instance they are set for MIN = 255 and MAX = 0.

It is then a matter of repeatedly checking each data value against both MIN and MAX. If the sample is less than MIN then MIN is set to now equal the sample. If the sample is greater than MAX, then MAX is set to equal the sample. This checking occurs for all 128 data byte values.

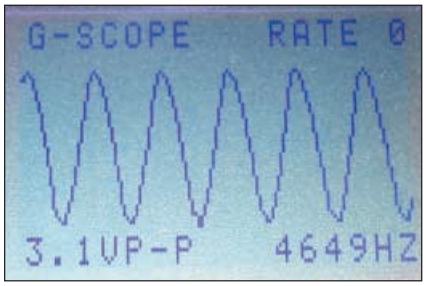
At the end of the process, MIN is subtracted from MAX and the result converted from its binary value to a 3-byte BCD (binary coded decimal) format. This is then output to the l.c.d.'s text screen as a 3-digit number with a decimal point inserted between the lefthand and middle digits, referencing the reading to the scale of the PIC's ADC.

An ADC value of 255 actually represents the supply line voltage at which the PIC's ADC is referenced, i.e. nominally 5V. The routines prior to decimalisation double the ADC values so that a MAX - MIN result of 5V (255) is represented as 510, and displayed as 5·10V. No attempt has been made to exactly "tune" the displayed value to the "real" value. The displayed value, therefore, should only be treated as a guide to actual min-max voltages.

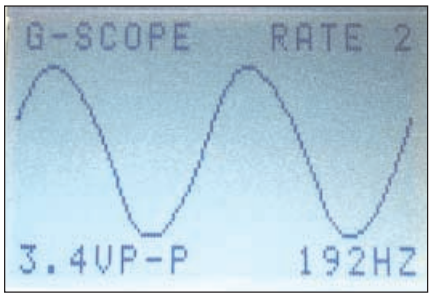
It should also be noted that the PIC does not monitor which gain setting has been selected via switch S2. The voltage reading simply represents that arriving at the PIC's RA0 pin.

FREQUENCY CALCULATION

Frequency calculation uses the same technique as in *M-Scope*. During development of the latter, the author fine-trimmed some counter reference values set into the program. In operation, the number of times that a signal value crosses a trigger level is counted during the period that a counter



Examples of waveforms sampled at different rates, showing the peak-to-peak voltage and the frequency. An example of a display without the text captions is shown earlier.



holding the preset value counts down to zero. The trigger-crossing count represents the frequency of the input signal in Hertz.

The technique is remarkably accurate, but is subject to slight deviation from correct values for systems not operating at *exactly* the same rate as the author's.

Should readers wish to correct for their controlling crystal's actual oscillation rate, the preset values can be corrected within the software. It is necessary, though, to have a signal generator and frequency counter so that the signal frequencies can be compared with those shown on the *G-Scope* screen. It is also necessary to have a suitable PIC programmer (such as *Toolkit*) to allow the source code to be recompiled and downloaded to the PIC.

The routines to be amended start at label GETFREQ0, with sufficient notations in the source code to clarify the appropriate ones. There are three involved, catering for each of the ADC sampling rates (more on which in a moment).

The author was interested to find, however, that the factors originally ascertained with *M-Scope* still applied to *G-Scope*, even though the 5MHz crystal was physically another component, bought at a different time.

The *M-Scope* and *G-Scope* test models achieved the following maximum frequency input values while still maintaining good accuracy:

Rate	Sig-Gen	Display
0	17007Hz	16984Hz
1	17007Hz	16998Hz
2	5827Hz	5812Hz

MONITORING ON/OFF

Using the frequency and voltage monitoring routines adds to the time taken to process each sampled signal batch. Even though the process is still quite fast, it was felt worthwhile to allow it to be bypassed. Switch S6 controls this function, toggling between the on and off states.

When the functions are turned off, the screen is cleared of the related text "messages".

When monitoring is in use, decimalisation of the voltage and frequency binary values is performed by a routine which is based in Bank 3. After conversion, leading zeros are blanked for the frequency reading, but not for the voltage reading.

ADC RATES

In common with *M-Scope*, three ADC sampling rates can be selected by switch S4. The rates are stepped through cyclically at each switch press. The screen displays the rate selected by its allocated number, between 0 and 2, but not in terms of specific time values.

The rates are set according to the value by which the PIC's master clock oscillator is divided within the ADCON0 prescaler. Bits 7 and 6 of ADCON0 control the division rate and the displayed numerical value represents the value set into those bits (see the PIC16F87x data sheet, Table 11-1).

Rate 0 (bits 7 and 6 = binary 00 = 0) is the fastest sampling rate for a conversion time of 400ns when using a 5MHz crystal oscillator. The data sheet refers to this rate as 2Tosc. Theoretically, the rate is faster than the data sheet recommends, but the author has frequently run the PIC's ADC at this rate in other designs without experiencing problems.

Rate 1 (bits 7 and 6 = binary 01 = 1) sets the 8Tosc rate, in which the conversion time is 1.6μs at 5MHz. Rate 2 (bits 7 and 6 = binary 10 = 2) sets the rate at 32Tosc with a conversion time of 6·4μs at 5MHz.

The data sheet shows a fourth rate selected with bits 7 and 6 = binary 11 = 3, but this rate (conversion period 2μs to 6μs) is reserved for when the PIC is run under RC (resistor-capacitor oscillator) mode. It is not suitable for implementing with *G-Scope*.

Rate 0 is the one required for sampling signals having higher frequency rates. Rate 2 is well suited to sampling sub-Hertz frequencies.

PROBES

It is not essential that a proper oscilloscope probe is used with G-Scope, although using one will help to keep the monitored signal free of external interference, and provide a convenient probed or clipped connection to the monitored circuit.

In many situations, though, using your multimeter's leads will provide an adequate coupling solution, and more cheaply. If you choose this option, socket SK1 can be replaced by two sockets to suit your meter leads. One should be the signal input socket, and the other for the 0V (GND) connection that is required between the unit and the circuit being monitored.

SOFTWARE

If programming your own PIC, it must be initialised with the settings stated at the head of the source code.

The source code (.ASM) is written in TASM, for which the assembled file is in .OBJ format, such as required by *Toolkit Mk2*. For those who work in MPASM, the .ASM file can be translated to that dialect using *Toolkit*'s software, even if you do not have the *Toolkit* hardware. □