Constructional Project

PIC WORLD CLOCK

JOHN BECKER

Graphically displays calendar, clock and global time-zone data.

RETURNING from holiday, Editor Mike commented to the author that he had seen an interesting world clock display at his hotel. It consisted of a world map across which was a series of light emitting diodes whose brilliance portrayed local time-zone daylight conditions. Could the author design one?

As with so many questions these days, it seemed that the Internet could well provide an answer. The first thing to ascertain was what such a clock might actually look like in detail. Search engine **www.google.com** was opened and told to search on various combinations of words such as *world, time-zone* and *clock*, amongst others.

World clock produced an astonishing number of web sites, but none that showed the display looked for. However, one of the sites revealed the screen dump shown in Fig.1. This set the author along a completely different thinking path.

In *EPE* Feb '01, his article *Using Graphics L.C.D.s* had been presented. Could this l.c.d. (liquid crystal display) be used to portray a world map? Following a letter about bitmaps and l.c.d.s from Javier Fernandez published in *Readout* Nov '01, the author knew that, in principle, it was possible to produce a screen dump of any image and process it for loading into a PIC microprocessor for output to a graphics l.c.d.

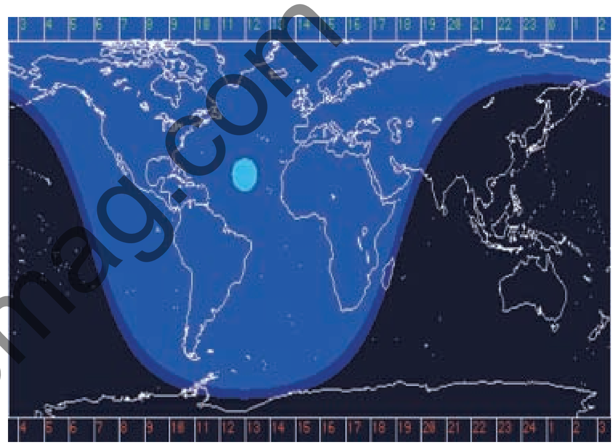


Fig.1. Real-time world clock as displayed by ***www.world-clock.org****.*

CLOCK PERFORMANCE

Before discussing *how* this was finally achieved, it is pertinent to say now that the end result is a PIC16F877-based circuit whose graphics l.c.d. shows the following:

- Simplified World map
- Current UK clock and calendar data
- Clock data for any other time-zone, adjustable via switches
- Flashing marker for sun's current highest position, i.e. true noon at that longitude (angle in relation to 0°, GMT, Greenwich Mean Time, London).
- Marker's position vertically (latitude) varies with the weeks and months throughout the year, spanning the Tropics of Capricorn and Cancer.
- Multi-paged text display of 150 major cities and their time-zone displacements in relation to GMT (e.g. New York –5 hours, Sydney +11 hours)
- Additional city time-zones can be readily added by those readers who have *PIC Toolkit Mk2 or Mk3 (TK3 V1.2 or higher).*
- Accuracy of clock time-keeping adjustable via switches.
- Principal clock and calendar data stored in the PIC's non-volatile EEPROM (electrically erasable programmable read-only memory) for recall in the event of power failure.
- Runs from a mains powered 9V battery adaptor, plus standby battery back-up.

BITMAP CONCEPT

When the Print Screen button of a PC's keyboard is pressed, the image on-screen at that moment is copied into the Windows

Clipboard. This can then be pasted into the PC's Paint software via the path:

Start – Programs – Accessories – Paint – Edit – Paste – File – Save As

In his *Readout* letter, Javier referred to a web site that supposedly described how this could be done. Regrettably, it turned out that this site, and Javier's own, were no longer accessible when tried by the author. There were, though, enough clue's in Javier's letter for the author to experiment and find an alternative way.

It eventually turned out to be fairly simple. The saved screen dump shown in Fig.1 was first converted to a black and white image, and then inverted to show black detail on a white background (Fig.2).

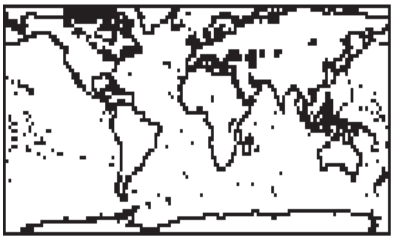


Fig.2. The image in Fig.1 was inverted to produce a black-on-white outline.

The graphics l.c.d. screen is 128 pixels wide by 64 pixels high. The screen dump image at this stage was too large and needed to be reduced in detail to fit the l.c.d. screen.

Next the image was "cleaned-up" by carefully using a combination of bit deleting and line clearing through Paint's toolbox. The image was then cut, the screen cleared and the image re-pasted to the very top left of the blank screen. At various stages during this process, the image was repeatedly re-saved in case of imminent errors.

Through Paint's Stretch/Skew option, the image was reduced in size to exactly 128 × 64 pixels. This image, of course, was of extremely low resolution and needed further cleaning-up to remove individual unwanted pixels (goodbye Hawaii!) which interfered with the main image required.

Inevitably, parts of Malaysia, the Mediterranean and North America had to be accepted as ill-defined (no offence, folks!). Ultimately, it turned out that the image also had to be rotated by 180° and flipped left to right (Fig.3).

Fig.3. Final "doctored" image, flipped left to right.

(While doing the map changes, there was a certain feeling of kinship with Slartibartfast, who designed Norway in *The Hitch-Hiker's Guide*!)

Using QuickBasic (QB), it was then established which aspects of the saved file data were Paint format commands and which were image data. A program was written which split the required data from the rest, converted it from binary to decimal values, to which a prefix of several spaces and the PIC command RETLW were added. The author had ensured that exactly 128 × 64 pixel values were processed, resulting in 1024 commands.

The file was saved with a .INC extension so that the PIC could import it as an Include file.

It has to be said that the process was not actually as straightforward as suggested by the foregoing. There were many stages of experimentation with this hitherto unknown technique before a satisfactory result was achieved.

PIC JUMP CAPACITY

A parallel problem to be solved was how this data could be used with a PIC16F877. The author already knew that the PCLATH command could allow table data to be stored in PIC program memory beyond the basic limitation of the first 256 bytes.

The problem was, he had never used it before. Much PIC experimentation ensued (Microchip data can be very short in adequate detail on occasions!). Eventually, using various PCLATH values, it was found that not only could data tables be accessed in separate 256 blocks beyond program address 255, but that the data could be loaded as a single table containing almost as many RETLW data commands as there were program memory locations still available.

It turned out that not all jumps were accessible, however. The first address for any table has to be (at least) ADDWF PCL,F, which removes this location from the table's use. Additionally, only 254 locations in each subsequent block of 256 commands could be accessed. Trying to access the 255th always took the program counter PCL into the "unknown" (as far as it was concerned) with a resulting "hang-up" of the program (as when tables in the normal address block 0 to 255 are too long).

With the map data it did not matter if the 255th data byte was not used, since it was known to be a screen border character which could be sent to the l.c.d. separately.

It did matter, though, with the table of city time-zone factors (see later). These are also held as consecutive RETLW data values and without formatting spaces, to conserve space. Each final byte of each block could not be ignored as with the map. The solution was to insert an additional data byte at every multiple of 256 bytes. An asterisk was used, but it could be any character.

This table is also stored as an Include file.

PCLATH USE

Associated with using PCLATH for extended table jumps was the need to also use this command to access PIC addresses $0800 (decimal 2048) upwards. This was

LISTING 1. Map display routine (TASM dialect)

```
MAP:    clrf PCLATH         ; reset PCLATH to zero
        clrf ADRLSB         ; * set l.c.d. graphics column
        movlw 0             ; * set l.c.d. graphics line number to zero
        call GLINE          ; * set line length
        call SCREENADR      ; * set l.c.d. screen address to these values
        movlw AWRON         ; * load W with Auto Write On value
        call SENDCMD        ; * send this command value to l.c.d.
        clrf LOOPE          ; clear table address counter
MAP2:   movf LOOPE,W        ; load table address into W
        bsf PCLATH,3        ; set PCLATH for program memory block $0800
        call WCLOCK         ; call Map table commencing at $0800, return with
                            ; graphics display data
        bcf PCLATH,3        ; clear PCLATH from block $0800 to block $0000
                            ; but leave bits 0-2 unchanged
        call OUTDATA        ; * send table data to l.c.d. via prog in block $0000
        incf LOOPE,F        ; increment table counter
        incf LOOPE,W        ; inc counter again, but only into W
        btfss STATUS,Z      ; is it zero i.e. is table counter = 255
                            ; i.e. has end of 256 table block been reached?
        goto MAP2           ; not yet, so repeat data get and send
        movf PCLATH,W       ; yes, so is PCLATH now pointing to sub-page 3?
        andlw %00000111     ; limit check to within value of 7
        xorlw 3             ; do = 3 check
        movlw 1             ; preset W with value of 1 (for RHS border)
        btfsc STATUS,Z      ; is Status bit Zero = 1 (equality)?
        movlw 0             ; no, border not needed as map display ended
        call OUTDATA        ; * send appropriate border value to l.c.d.
        clrf LOOPE          ; clear loop counter to zero
        incf PCLATH,F       ; inc PCLATH for next table block of 256
        movf PCLATH,W       ; is table block count now = 4?
        andlw %00000111     ; limit check to within value of 7
        xorlw 4             ; do = 4 check
        btfss STATUS,Z      ; is status bit Zero = 1 (equality)?
        goto MAP2           ; not yet, repeat for next block
        clrf PCLATH         ; yes, fully clear PCLATH back to zero
        movlw AWROFF        ; * load W with Auto Write Off value
        call SENDCMD        ; * send this command value to l.c.d.
        return              ; return to main program
```

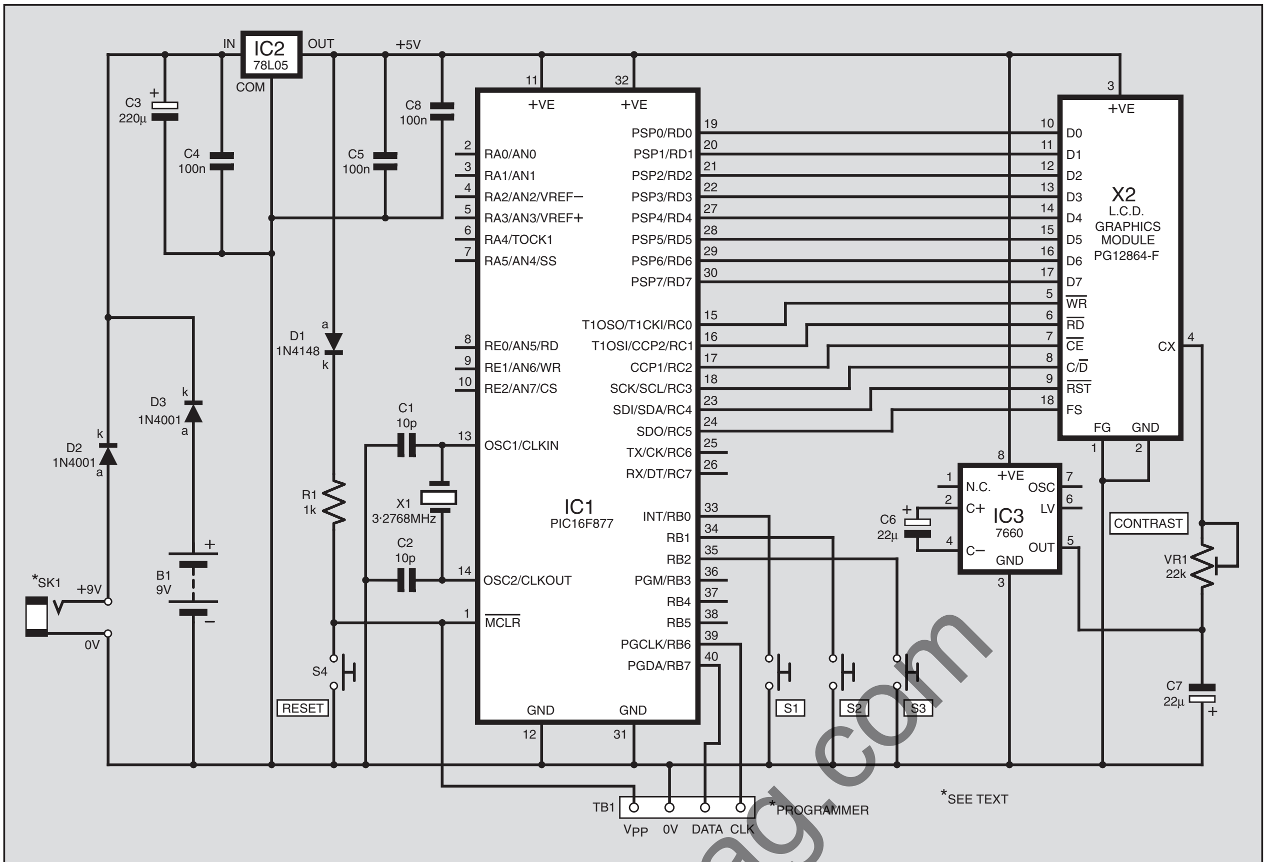


Fig.3. Complete circuit diagram for the PIC World Clock.

the author's first foray into that region, and also needed research.

Some readers may not appreciate that large-capacity PICs have this limitation. Program memory addresses are split into blocks of 2048, each of which requires PCLATH to be set accordingly, in addition to setting it for individual blocks of 256 within the main block when tables are embedded there. The 2048 limitation does not affect the PIC16F84, of course, since this only has 1024 memory locations anyway, but it can affect the PIC16F873 to 877 when the extra program memory capacity is required.

At the time the author was working on this PIC World Clock, John Waller's article *Using the PIC's PCLATH Command* had not yet been discussed with him, let alone published (as it was last month, July '02). Readers now have access to John's information so PCLATH will not be discussed further in detail here.

It is, however, appropriate to show an extract of the program listing associated with displaying the map. See Listing 1.

The commands and sub-routines called and marked by an asterisk are those which were discussed in detail in the *Using Graphics L.C.D.* article previously mentioned. All the graphics l.c.d. routines used are those discussed there, and used here as "library" routines.

CIRCUIT DIAGRAM

Whilst the software is the longest that author has written for a PIC (around 5500 commands), the circuit is one of the simplest. Its schematic diagram is shown in Fig.3.

The PIC16F877 is notated as IC1. It simply runs its program and controls the data output to the graphics l.c.d. X2. Additionally, it responds in various ways to switches S1 to S4 being pressed, more on this later.

If considering using a different l.c.d. to the Powertip PG12864 recommended, ensure that it is based on a Toshiba T6963C controller.

The PIC is powered at +5V, as supplied by regulator IC2, which may be powered at between about 7V and 12V d.c. Capacitors C3 to C5, plus C8, simply help to maintain power line stability.

As discussed in *Using Graphics L.C.D.s*, the Powertip graphics l.c.d. module requires a split supply of +5V, 0V and –5V. The latter is generated by the d.c.-to-d.c. voltage converter IC3. This produces a –5V d.c. output when powered from a +5V supply.

It is a switched-mode device (frequently seen in *EPE* designs) whose oscillation frequency is set by capacitor C6. The output voltage is smoothed by C7. The l.c.d. screen contrast is determined by the current flowing from its pin 4 (CX) into the negative line, and is controllable by preset VR1.

The PIC is operated at 3·2768MHz, as set by crystal X1 in conjunction with capacitors C1 and C2.

It can be programmed *in situ* from a PIC programmer such as *Toolkit Mk2 or Mk3.* The World Clock software and preprogrammed PICs are available as stated later.

Diode D1 and resistor R1 allow the PIC to be correctly controlled when being programmed. They also provide bias to Reset switch S4, whose function is described later. Do not omit these two components even if you do not intend to program the PIC yourself.

Terminal block TB1 provides access to the PIC's programming pins.

A "belt and braces" option is provide for power input. Surprisingly, the circuit draws around 18mA, much of which is demanded by the l.c.d. Even making use of the PIC's SLEEP mode with interrupts did little to reduce the overall consumption. Consequently, continuous operation of the clock from a 9V PP3 battery is unrealistic.

Instead, the unit should normally be powered from a battery adaptor having an output of around 9V d.c. A PP3 battery can be used as a back-up supply in the event of a mains power failure, and a battery holder is included on the printed circuit board (p.c.b.) for this purpose. Diodes D2 and D3 prevent the battery and adaptor supply from mutual interference, allowing the battery to take over if the mains supply fails.

CONSTRUCTION

The p.c.b., component and track layout details are shown in Fig.4. This board is available from the *EPE PCB Service*, code 363.

It has also been designed as a general purpose board for use in other simple PIC16F877/graphics l.c.d. applications. Consequently, additional holes are provided to allow access to the otherwise unused PIC port pins. They should be ignored in this application.

Commence construction by soldering in the several link wires, noting that a few are positioned below IC1 and IC3. Dual-in-line (d.i.l.) sockets should be used for both these i.c.s. Do not insert the i.c.s themselves until the first stage of power checking has been performed. The same caution applies to the l.c.d. as well.

COMPONENTS

Resistor
R1 1k 0·25W 5% carbon film

Potentiometer
VR1 22k min. round preset

See SHOP TALK page

Capacitors
C1, C2 10p ceramic, 0.2in pitch (2 off)
C3 220μ radial elect. 16V
C4, C5, C8 100n ceramic, 0·2in pitch (3 off)
C6, C7 22μ radial elect. 16V

Semiconductors
D1 1N4148 signal diode
D2, D3 1N4001 rectifier diode (2 off)
IC1 PIC16F877 microcontroller, preprogrammed (see text)
IC2 78L05 +5V 100mA voltage regulator
IC3 7660 d.c.-to-d.c. voltage converter

Miscellaneous
S1 to S4 min. s.p. push-to-make switch 0·2in x 0·25in pitch (4 off) (S1 to S3 may be panel mounting types, see text)
SK1 power socket for 9V battery adaptor
X1 3·2768MHz crystal
X2 PG12864-F graphics l.c.d. module, T6963-based

Printed circuit board, available from the *EPE PCB Service*, code 363; 8-pin d.i.l. socket; 40-pin d.i.l. socket; PP3 battery holder, p.c.b. mounting; 9V PP3 battery; 9V battery adaptor, mains powered; plastic case, 190mm x 110mm x 60mm (see text); 18-way ribbon cable or pin-header connector pair, p.c.b. mounting (see text); mounting bolts to suit; 1mm terminal pins; solder.

***Approx. Cost Guidance Only* £60**
excluding case & PSU

Assemble the other few components in any order you prefer. Leave the battery holder until last.

Thoroughly check your board for poor soldering and other errors, and then connect the 9V mains adaptor. Check that regulator IC2 outputs +5V, within a few per cent. If not, switch off and remedy the cause of malfunction.

When testing, if the unit does not behave as expected, and when inserting or extracting the i.c.s, always disconnect the power.

When satisfied with the +5V output, plug in d.c. converter IC3 and check that it outputs –5V, again within a few per cent.

If this is satisfactory as well, connect the l.c.d. to its designated p.c.b. pins, which are in exactly the same order as on the l.c.d. itself. Ribbon cable was directly soldered to terminal pins on the prototype, but a p.c.b. mounting 0·1 inch pitch 18-way pin-header strip with connector could be used if preferred.

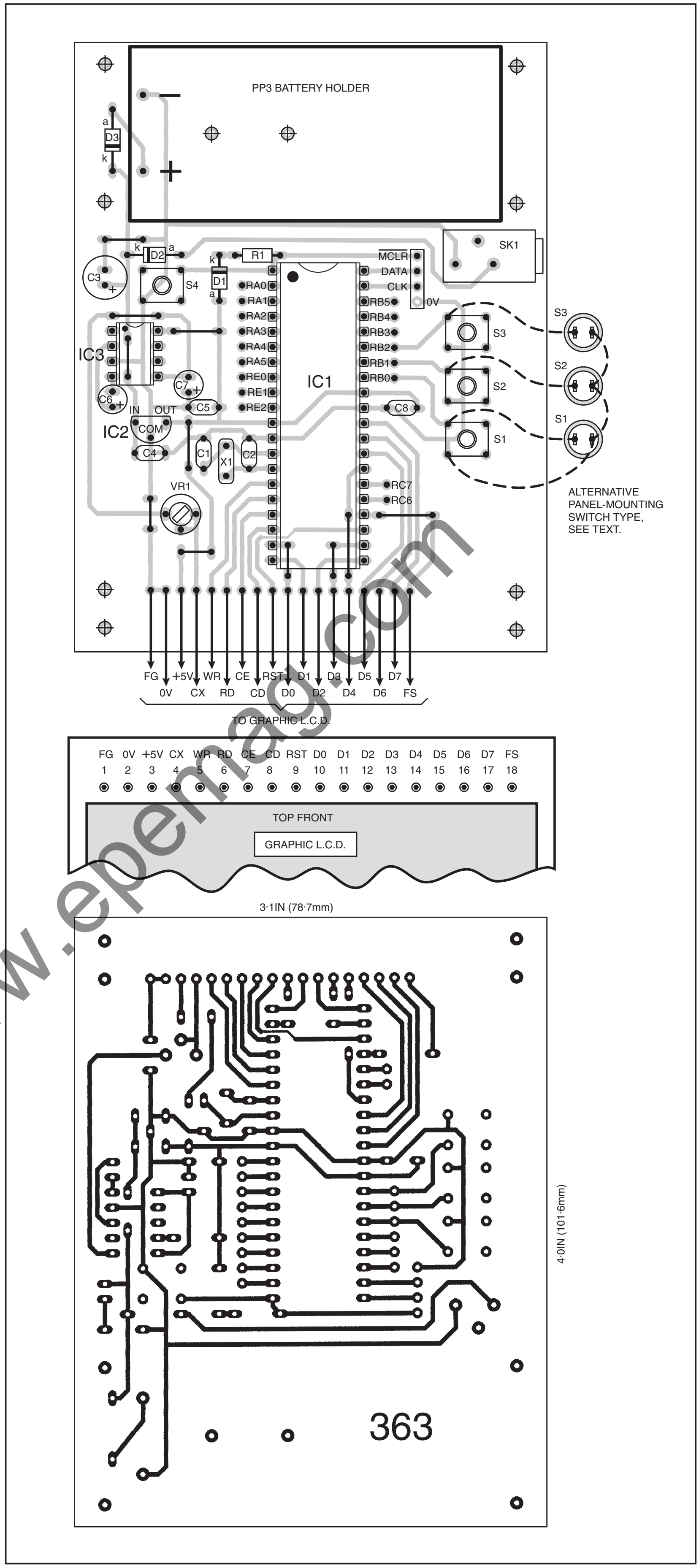
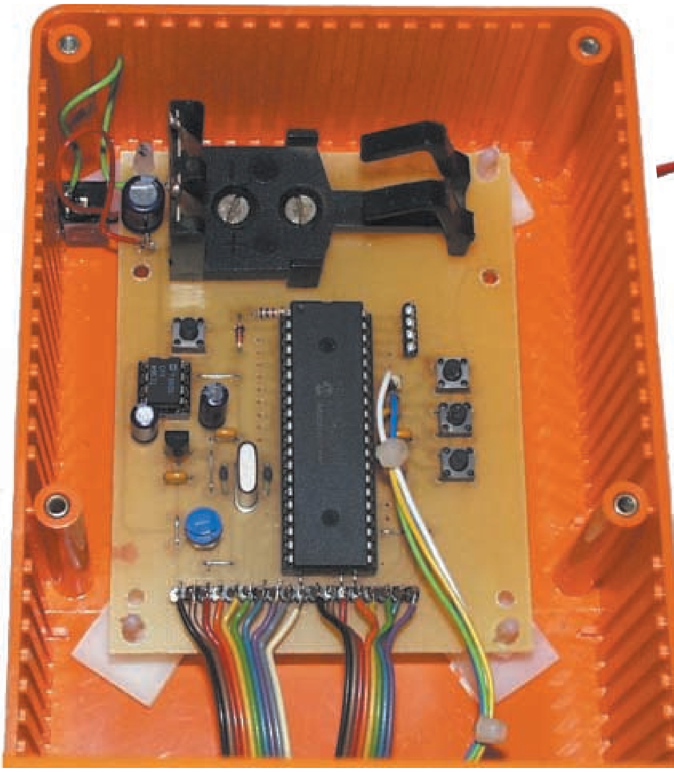


Fig.4. Printed circuit board topside component layout, full-size copper foil master pattern and wiring to the l.c.d. and off-board switches (see text).



Component layout on the completed prototype circuit board.

On powering up, adjust preset VR1 until a change in l.c.d. screen colour is observed. Adjust VR1 until the background shows a very light shade of blue (might be grey with some makes of l.c.d.).

Next the PIC can be inserted and, if it has not been preprogrammed, it can be programmed now, using a suitable programmer, as stated above.

ENCLOSURE

It was originally conceived that the PIC World Clock could look more interesting if not enclosed in a box. Consequently the p.c.b. was designed to be bolted behind the l.c.d., and for the "sandwich" to be bolted to a perspex sheet mounted in a low-cost picture frame. Access to the switches is then from behind the assembly.

Ultimately, however, the author used the same box as previously used in another design. As a result switches S1 to S3 and SK1 were mounted in the case (although S1 to S3 are still to be seen in the photograph of the p.c.b.). Do not mount S4 in the box leave it on the p.c.b.

SOFTWARE CHECKS

With the unit running under PIC control, the first action that will be seen is that the screen should show a display similar to that in photo below left. If necessary, adjust VR1 until the contrast is suitable.

Example screen in adjustment mode.

A rudimentary map of the world will be seen, with three lines of text superimposed on a partly blanked area.

Under the first 0 of the time (tens of hours) a cursor cell should be seen flashing. Pressing and releasing switch S1 steps this value on a cycle from 0 to 2 and back to 0. Set your current tens of hours now.

Pressing switch S2 steps the cursor to the units of hours. Using S1, these cycle through 0 to 9 followed by a rollover to 0 when the tens of hours show 0 or 1. If the tens show a value of 2, the rollover of the units is after 3 (24 hours clock).

Switch S2 progressively steps through the minutes digits, selectable by S1 again, with a rollover limitation of 59. To suit the program's correct use of calender factors, the next steps are to the tens and units of years, with a rollover after 99. It is worth noting that Microchip only guarantee retention of a PIC's program contents for 40 years, so a year value of 99 is grossly over-optimistic!

Next the month can be set. This cycles from JAN to DEC followed by rollover. Tens and units of days in the month are next, with rollover limits set by the conventional number of days in any named month, with automatic allowance for leap years if the month is FEB.

The named day of the week follows, MON to SUN, with a 7-day rollover.

CLOCK ACCURACY

The next press of switch S2 sets the program into clock accuracy adjustment mode. This is prefixed by the symbols +/– followed (when first used) by +0000, the cursor flashing on the forward-slash symbol. Switch S1 continues to step count values upwards, but S2 now causes a downwards count. The full range is –9999 to +9999.

Whilst the PIC is crystal controlled, when used in clock-type applications there is normally an inherent slippage of accuracy over time, partly due to the crystal frequency not being at an exact value, due to normal component tolerance factors. An additional slippage can occur because of very slight inconsistencies in the rate at which the PIC's internal timing counter is accessed.

In a long-term clock design, such as this is intended to be (40 years, anyway!), it is desirable that the clock rate can be adjusted in the light of experience, as with many types of normal clock.

To cope with this, the software has been written so that the amount by which the clock registers are incremented is adjustable according to externally set values.

In simple PIC clock designs, the TMR0 counter is set so that it rolls over at, say, every 1/25th of second. Counting 25 of these rollovers then equals a one-second time lapse.

In this design, though, TMR0 is set to roll over once every 1/50th of a second. On each occasion, a 3-byte counter has a preset value added to it. This includes an adjustment factor as set by the user. With no adjustment factor set, this counter rolls over once for every two TMR0 rollovers (i.e. at every 1/25th of a second).

When first used, the software sets the 3-byte counter to a decimal value held as MSB = 128, NMSB = 0, LSB = 0. Two additions of this value cause the MSB to rollover and set a separate register flag. Only if this flag = 1 is another counter incremented, whereupon the flag is reset.

Only when this counter has incremented 25 times is the seconds register incremented. In fact, for programming ease, the intermediate counter is preloaded with a value of 25, and then decremented down to zero, at which point it is reset to 25 and the seconds counter incremented.

It will be seen that if the 3-byte counter is preset with a value greater than 128, 0, 0, then its MSB rollover with be faster than just described. Similarly, the rollover will be slower if the preset value is lower.

Each unit of change, set via S1 or S2, in the adjustment count value shown on screen, represents one second of change every 4,194,304 seconds. There are approximately one million seconds in 11.5 days, so the potential for clock setting accuracy is good.

It is worth understanding, though, that a crystal's frequency can drift fractionally with temperature and age.

This technique was first used in the author's *Canute Tide Predictor* of June '00 and has proved remarkably accurate. Implementing any adjustment should only be carried out after several days of observation to determine how much the clock has drifted over that time, and then to apply an adjustment calculated in relation to the above four million ratio.

At this stage of use, the adjustment factor should left at zero.

GLOBAL DISPLAY

Having adjusted the clock and calendar values, press switch S3. This first stores the values to the PIC's EEPROM, where they remain even after power loss, to be recalled when power is restored. The screen is then cleared of the data setting display, to reveal the world map as below:

Typical World Clock display when in normal running mode.

To the left of the map are shown the calendar values, which will be kept updated for as long as the clock is powered. At the bottom left of the screen the current hours and minutes time is shown, plus a seconds counter. To the right is shown another hours and minutes display, prefixed by the letter L, meaning Local. Currently it will be showing the same time as the first clock.

One of the functions of the World Clock is to allow a principal time to be shown for the UK and for a secondary time to be shown in relation to any time-zone across the globe, i.e. Local time in that zone.

In the middle of the screen is a vertical dotted line passing through what would be seen as roughly London on a better definition display. Pressing switch S1 shifts this line to the east in large steps, of *about* 1 hour 30 minutes. The time-step values are automatically added to the UK time and displayed as the Local time for the longitude indicated by the line.

Having reached the eastern map edge, the line then reappears from the west. Crossing the full map represents 24 hours.

Pressing switch S2 moves the line in smaller increments, 128 steps across the screen. This allows the line to be more precisely set in relation to the map. The line can only be moved in an easterly direction. Its position is never stored to EEPROM and returns to its default position (UK) should power be lost and then restored (or the Reset switch, S4, pressed – see later).

SOLAR AND NATIONAL TIME-ZONES

Greenwich, UK, is regarded as having a longitude of zero. Because the Earth is roughly a globe, it is said to have a circumference of 360°. Consequently, for a 15° shift westwards of the sun from a noon position above Greenwich at 0° means a one hour time zone shift (24/(360/15)). Solar noon is now at this 15° position, and the time at Greenwich has become 1.00pm.

However, what about the time actually experienced by the positions under the solar noon position? What about *EPE* in Dorset, 2° west of Greenwich? Do we experience noon at 2° = 8 minutes later than Greenwich? No, of course not, when it's noon at Greenwich it's noon at *EPE!* (Although some of us might feel/wish it were actually 5pm . . .)

Interestingly, it was only with the increasing use of the railways, that, in 1884 at an international conference of 27 nations in Washington DC, national and international time-zones became rationalised, with Greenwich as the 0° meridian. Prior to that local noons were at different instants to each other. Improved transport systems though, required consistent timekeeping.

Generally speaking, time-zones change in steps of 30 minutes, although there are occasional differences, where a 15-minute step might occur.

As a result, France, for instance, is in a time-zone one hour ahead of the UK (+1 hour) even though much of it is due south of the UK. New York is five hours behind Greenwich (–5 hours), but because of the scale of the USA, Chicago for example has a time-zone displacement of –6 hours from the UK. Curiously, despite its size, China has only one time-zone.

TIMELY DILEMMA

These facts presented the author with an initial dilemma. Should the dotted time-zone line have its clock display incremented to suit solar time, or local time? And, if the latter, then *which* national time zone, since some countries along that line could have adopted different displacements in relation to Greenwich.

It was decided that the Local clock should only be incremented in steps of 30 minutes and in relation to solar noon. Thus most of France will appear to be in the same solar time-zone as the UK. Intelligent assessment of the line's position in relation to the displayed Local time must be used!

Only if a much larger display were to be used could there be any chance of tailoring regional time-zones to the geographic location under the line. In other words, a PIC and simple l.c.d. are not up to that degree of definition! If you need more accuracy, use a computer and browse appropriate sites via the Internet (such as listed later)!

To comply with the 30-minute stepped update, a look-up table is used, which allocates whether a value of 30 or 0 minutes is added to the Local time display at each increment of the line.

In early stages of program development, the Local time was in fact incremented according to actual solar time. This required a value of 11 minutes 15 seconds to be added for each of the 128 increments.

SOLAR SEASON

Whilst the sun appears to move round the Earth in a westerly direction, the Earth rotates eastwards to greet the rising sun, of course. Surprisingly, doing a "straw poll" recently, the author found that the correct answer was not always given. But even the illustrious SF author Arthur C. Clarke is said to have wrongly stated the Earth's direction of rotation in one of his books!

Because of the Earth's tilt on its axis, the sun's overhead position changes in latitude throughout the year as the Earth travels around sun. An indication of this has been added to the display. A flashing 4-pixel vertical line travels across the map indicating where solar noon is occurring.

The flashing noon line also changes position vertically throughout the year, traversing between the Tropic of Capricorn (northern hemisphere) and the Tropic of Cancer (southern). This is calculated in relation to the stated month and its numbered day, and makes use of more look-up tables. The position is only an approximation – don't navigate by it (hey, who's moved America?)!

CITY TIME-ZONES

Having discovered the benefits of using PCLATH with a large-capacity PIC, it became obvious that lots more table data could be added to the software. This resulted in City time-zone displacements being downloaded from the web, at **www.timeanddate.com**, whose information is presented in real time in relation to local time of the user, as follows, for example:

Addis Ababa	Tue 5:34 PM	Hanoi	Tue 9:34 PM
Adelaide	Wed 12:04 AM	Harare	Tue 4:34 PM
Aden	Tue 5:34 PM	Havana *	Tue 10:34 AM
Aklavik *	Tue 8:34 AM	Helsinki *	Tue 5:34 PM
Algiers	Tue 3:34 PM	Hong Kong	Tue 10:34 PM
Amman *	Tue 5:34 PM	Honolulu	Tue 4:34 AM
Amsterdam *	Tue 4:34 PM	Houston *	Tue 9:34 AM
Anadyr *	Wed 3:34 AM	Indianapolis	Tue 9:34 AM
Anchorage *	Tue 6:34 AM	Islamabad *	Tue 8:34 PM
Ankara *	Tue 5:34 PM	Istanbul *	Tue 5:34 PM
Antananarivo	Tue 5:34 PM	Jakarta	Tue 9:34 PM
Asuncion	Tue 10:34 AM	Jerusalem *	Tue 5:34 PM
Athens *	Tue 5:34 PM	Johannesburg	Tue 4:34 PM
Atlanta *	Tue 10:34 AM	Kabul	Tue 7:04 PM
Baghdad *	Tue 6:34 PM	Kamchatka *	Wed 3:34 AM
Bangkok	Tue 9:34 PM	Kathmandu	Tue 8:19 PM
Barcelona *	Tue 4:34 PM	Khartoum	Tue 5:34 PM

Example display of international time at ***www.timeanddate.com****.*

Using QB, these were analysed, formatted into a look-up table, and imported to the PIC as another .INC file.

The following is an extract, in relation to New York, showing a –5 hours difference from UK time. Note the "&" end of name marker and the "*" 256-jump padder referred to earlier:

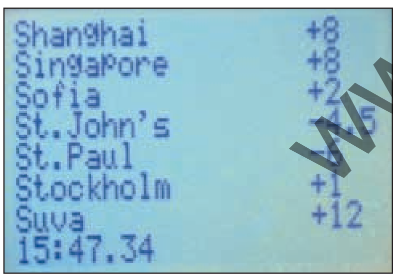
```
retlw 'N'
retlw 'e'
retlw 'w'
retlw ' '
retlw 'Y'
retlw '*'          ;256 rollover 3
retlw 'o'
retlw 'r'
retlw 'k'
retlw '-'
retlw '5'
retlw '&'
```

As supplied with the software, there are 136 city names, ranging from Adis Ababa to Zurich. They are called to screen by first pressing switch S3. This clears the map, and sets the screen area to 20 × 8 text character cells, instead of the previous 16 × 8. This width is not suitable for map graphics display since bits 7 and 6 of each screen data byte are ignored by the l.c.d.

The width is well-suited to text-only displays, however, allowing 20 characters per line instead the normal 16.

On entry to the City Time-zone display, the first seven cities are named, with their time-zone displacement from Greenwich shown to their right. On the eighth line at bottom left a continuation of the UK real-time clock count is shown:

Example of time-zone displacement data screen.



Another example of time-zone displacement data screen.

Pressing switch S1 or S2 "turns the page" to the next seven cities. The clock count continues as before. There are 19 pages that can be stepped through, with just three cities on the final page.

Pressing S1 or S2 when the final page is displayed rolls the display back to page 1. At any page, switch S3 may pressed to make a return to the map display. Re-entering the cities mode always starts the display at page 1.

The author briefly considered having a facility to automatically show the current time for each city, but decided that readers are perfectly capable of doing the mental maths and add the displacements to the UK time shown!

ADDING CITIES

Readers who have the *Toolkit TK3 V1.2* (or MPASM) software update can add their own cities to the TIME-ZONE.INC file, and then re-assemble and send to the PIC. Note that only the revised TK3 software (TK3 V1.2 or later) can handle PIC addresses from $0800 onwards.

There are over 3000 program memory locations that could be filled. However, if the TIME-ZONE table extends into the next 2048 block, changes to the PCLATH control (using PCLATH bits 4 and 3) would need to be made.

As things stand, 550 additional characters could be added to the .INC file without crossing the next 2K boundary, after $1FFF.

To add names, TK3's Include File Edit/View Facility can be used. First use its DIR button to select file TIME-ZONE.INC. Open the file via the Edit Incl button.

Split the city name into individual letters and enter them in order at an alphabetically suitable place within the file, in the same way that the other letters are treated. Follow the name with the time displacement value, and use the "&" symbol to indicate the end of that city's data. Do not use space (" ") characters unless they are part of the name.

City names can also be deleted.

Be aware that adding or deleting RETLW commands will affect the 256-jump allocation beyond that point. The author used DOS EDIT to correct for this as it has a good line counter. At each line count multiple of 256 (i.e. 256, 512, 768, 1024, 1280, etc) insert the RETLW '*' separator, as done in the original file, removing the author's inserts as necessary.

Failure to do so will not "crash" the program, but will cause one or more asterisks to be seen and with consequential non-display of some characters.

The final entry of RETLW '#' must be retained. Removing it could cause the PIC program to "crash" on the final page of city data display.

Re-save the file once corrected, reassemble from ASM to HEX and program the PIC with the new HEX code.

ZONE HOME ET!

It is perfectly feasible for non-UK readers to set their own time zone values into the clock in place of UK time. However, this will have two side-effects.

Firstly, the solar noon flashing line will continue to think that the time is still related to GMT. No facility to change this has been included.

Secondly, the city time-zone displacement text values will become invalid. This can be amended if *Toolkit TK3 V1.2* or MPASM is used.

Accessing the web site stated later, view the times quoted for the various cities in relation to the zone from which you have entered the site. Calculate and note the displacement. Amend the TIME-ZONE.INC file so that the new values replace the GMT ones. Then resave and proceed as described in the previous section.

GENERAL USE

There are three situations in which the calendar and clock data are stored to the PIC's data EEPROM: following a program reset after a total power failure restoration or by deliberate intent; at each midnight rollover; and when the Cities text display is entered.

To have updated the EEPROM on a more frequent regular timed basis would undesirably use up its theoretical life expectancy, which value could not be found in Microchip's data for the PIC16F877, but is believed to be about 10,000 write cycles. In fact, the author believes that over the years of using the same PICs over and over in different applications, the write cycle count has probably been well exceeded on several of them, without failure.

Should a Reset occur, the currently stored data will be recalled and displayed on-screen as first described earlier. In the event of a short halt in running, the time and calendar data will need little adjustment.

To allow the clock to be adjusted, a fourth switch has been included, Reset switch S4. This is connected to the PIC's MCLR line and physically resets the PIC so that it starts running the program from the beginning.

Any time or calendar values can now be adjusted, pressing the switches as before, ignoring any values that do not need adjustment. At any stage, if the remaining values do not need to be changed, press switch S3 to jump straight into map display mode.

Before using Reset switch S4, first press S3 to enter the City text display, which will cause the current time to be saved, for immediate recall following S4 being pressed.

TIMELY END

Apart from describing a novel time-zone clock-calendar, it is hoped that this article has provided you with further thoughts about using PCLATH, accessing very long look-up tables, crossing Page boundaries, and using screen dumps to obtain data for loading into a graphics l.c.d. via a PIC microcontroller.

May the sun always cross your zone!

RELATED WEB SITES

www.timeanddate.com/worldclock/. World Clock Time-zones – current times for global cities.

www.world-clock.org. Visual Map of the World's Time – Imagery by Matthew Kaufman, the site which inspired the l.c.d. map.

www.google.com. Excellent search engine – says it contains 1,960,000 sites related to searching on the command World Clock. Both the above sites are on the first page of its display.

www.greenwichmeantime.com/info/time-zone/htm. History of time-zones, plus related matters.

SOFTWARE

The software for the PIC World Clock is available on 3·5-inch disk (for which a small handling charge applies) from the *EPE* Editorial office. It is also available for free download from the *EPE* ftp site, which is most easily accessed via the click-link option at the top of the screen page when you enter the main web site at **www.epemag.wimborne.co.uk**.

On entry to the ftp site take the path PUB – PICS – WORLDCLOCK, downloading all files within the latter folder.

For information about obtaining components and preprogrammed PICs for this project, read the *Shoptalk* page in this issue. □

EVERYDAY PRACTICAL ELECTRONICS

BACK ISSUES

THE No.1 MAGAZINE FOR ELECTRONICS TECHNOLOGY & COMPUTER PROJECTS

We can supply back issues of *EPE* by post, most issues from the past three years are available. An *EPE* index for the last five years is also available – see order form. Alternatively, indexes are published in the December issue for that year. Where we are unable to provide a back issue a photocopy of any *one article* (or *one part* of a series) can be purchased for the same price. Issues from Jan. 2002 onwards are also available to download from **www.epemag.com**.

DID YOU MISS THESE?

APRIL '01

PROJECTS ● Wave Sound Effect ● Intruder Alarm Control Panel–Part 1 ● Sound Trigger ● EPE Snug-Bug Pet Heating Control Centre.
FEATURES ● The Schmitt Trigger–Part 6 ● Practically Speaking ● Ingenuity Unlimited ● Circuit Surgery ● Net Work – The Internet Page ● *FREE* supplement – An End To All Disease.

MAY '01 Photostats only

PROJECTS ● Camcorder Mixer ● PIC Graphics L.C.D. Scope ● D.C. Motor Controller ● Intruder Alarm Control Panel–Part 2.
FEATURES ● The Schmitt Trigger–Part 7 ● Interface ● Circuit Surgery ● Ingenuity Unlimited ● New Technology Update ● Net Work – The Internet Page.

JUNE '01

PROJECTS ● Hosepipe Controller ● In-Circuit Ohmmeter ● Dummy PIR Detector ● Magfield Monitor.
FEATURES ● Controlling Jodrell Bank ● PIC1687x Extended Memory Use ● Practically Speaking ● Ingenuity Unlimited ● New Technology Update ● Circuit Surgery ● Net Work – The Internet Page.

JULY '01

PROJECTS ● Stereo/Surround Sound Amplifier ● PIC to Printer Interface ● Perpetual Projects 1– Solar-Powered Power Supply and Voltage Regulator ● MSF Signal Repeater and Indicator.
FEATURES ● The World of PLCs ● Ingenuity Unlimited ● Circuit Surgery ● New Technology Update ● Net Work – The Internet Page.

AUG '01

PROJECTS ● Digitimer ● Lead-Acid Battery Charger ● Compact Shortwave Loop Aerial ● Perpetual Projects 2 – L.E.D. Flasher – Double Door-Buzzer.
FEATURES ● Controlling Power Generation ● Ingenuity Unlimited ● Interface ● Circuit Surgery ● New Technology Update ● Net Work – The Internet Page.

SEPT '01

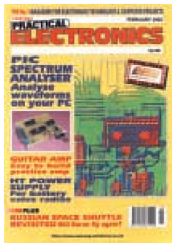
PROJECTS ● Water Monitor ● L.E.D. Super Torches ● Synchronous Clock Driver ● Perpetual Projects 3 – Loop Burglar Alarm – Touch-Switch Door-Light – Solar-Powered Rain Alarm.
FEATURES ● Controlling Flight ● Ingenuity Unlimited ● Practically Speaking ● Circuit Surgery ● New Technology Update ● Net Work – The Internet Page.

OCT '01 Photostats only

PROJECTS ● PIC Toolkit Mk3 ● Camcorder Power Supply ● 2-Valve SW Receiver ● Perpetual Projects 4 – Gate Sentinel – Bird Scarer – In-Out Register.
FEATURES ● Traffic Control ● Ingenuity Unlimited ● New Technology Update ● Circuit Surgery ● Interface ● Net Work – The Internet Page ● *Free* 2 CD-ROMs – Microchip 2001 Tech Library.

NOV '01

PROJECTS ● Capacitance Meter ● Pitch Switch ● Lights Needed Alert ● Teach-In 2002 Power Supply.
FEATURES ● Teach-In 2002 – Part 1 ● Practically Speaking ● Circuit Surgery ● New Technology Update ● Ingenuity Unlimited ● Net Work – The Internet Page ● *Free* 16-page Supplement – PIC Toolkit TK3 For Windows



DEC '01

PROJECTS ● Ghost Buster ● PIC Polywhatsit ● Twinkling Lights ● Mains Failure Alarm.
FEATURES ● Teach-In 2002 – Part 2 ● Marconi – The Father of Radio ● Interface ● Ingenuity Unlimited ● Circuit Surgery ● New Technology Update ● Net Work – The Internet Page ● 2001 Annual Index.

JAN '02

PROJECTS ● PIC Magick Musick ● Time Delay Touch Switch ● Versatile Bench Power Supply ● Forever Flasher.
FEATURES ● Teach-In 2002 – Part 3 ● Practically Speaking ● Ingenuity Unlimited ● New Technology Update ● Circuit Surgery ● Net Work – The Internet Page.

FEB '02

PROJECTS ● PIC Spectrum Analyser ● Guitar Practice Amp ● HT Power Supply ● Versatile Current Monitor.
FEATURES ● Teach-In 2002 – Part 4 ● Ingenuity Unlimited ● Russian Space Shuttle Revisited ● Circuit Surgery ● Interface ● New Technology Update ● Net Work – The Internet Page.

MAR '02

PROJECTS ● MK484 Shortwave Radio ● PIC Virus Zapper ● RH Meter ● PIC Mini-Enigma.
FEATURES ● Teach-In 2002 – Part 5 ● Ingenuity Unlimited ● Programming PIC Interrupts–1 ● Circuit Surgery ● Practically Speaking ● New Technology Update ● Net Work – The Internet Page.

APR '02

PROJECTS ● Electric Guitar Tuner ● PIC Controlled Intruder Alarm ● Solar Charge and Go ● Manual Stepper Motor Controller.
FEATURES ● Teach-In 2002 – Part 6 ● Interface ● Programming PIC Interrupts–2 ● Circuit Surgery ● Ingenuity Unlimited ● New Technology Update ● Net Work – The Internet Page ● *FREE* Giant Op.Amp Data Chart.

MAY '02

PROJECTS ● PIC Big-Digit Display ● Simple Audio Circuits – 1 ● Freezer Alarm ● Washing Ready Indicator.
FEATURES ● Teach-In 2002 – Part 7 ● Ingenuity Unlimited ● Practically Speaking ● New Technology Update ● Circuit Surgery ● Net Work – The Internet Page.

JUNE '02

PROJECTS ● Biopic Heartbeat Monitor ● Frequency Standard Generator ● Simple Audio Circuits – 2 ● World Lamp.
FEATURES ● Teach-In 2002 – Part 8 ● Interface ● New Technology Update ● Circuit Surgery ● Ingenuity Unlimited ● Net Work – The Internet Page.

JULY '02

PROJECTS ● EPE StyloPIC ● Infra-Red Autoswitch ● Simple Audio Circuits – 3 ● Rotary Combination Lock.
FEATURES ● Teach-In 2002 – Part 9 ● Practically Speaking ● Using The PIC's PCLATH Command ● Ingenuity Unlimited ● Circuit Surgery ● New Technology Update ● Net Work–The Internet Page.

BACK ISSUES ONLY £3.30 each inc. UK p&p.
Overseas prices £3.80 each surface mail, £5.25 each airmail.

We can also supply issues from earlier years: 1998 (**except** Jan. to May, July, Nov., Dec.), 1999, 2000 (except Feb., July), 2001 (except May, Oct.). Where we do not have an issue a photocopy of any ***one article*** or ***one part*** of a series can be provided at the same price.

ORDER FORM – BACK ISSUES – PHOTOCOPIES– INDEXES

☐ Send back issues dated . . . .

☐ Send photocopies of (article title and issue date) . . . .

☐ Send copies of last five years indexes (£3.30 for five inc. p&p – Overseas £3.80 surface, £5.25 airmail)

Name . . . .

Address . . . .

. . . . Tel: . . . .

☐ I enclose cheque/P.O./bank draft to the value of £ . . . .

☐ Please charge my Visa/Mastercard/Amex/Diners Club/Switch £ . . . . Switch Issue No. . . . .

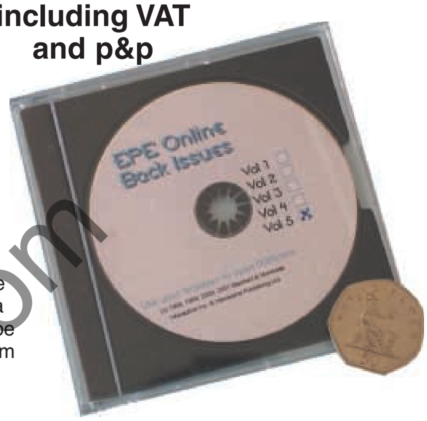
Card No. . . . . Card Expiry Date . . . .

SEND TO: **Everyday Practical Electronics, Wimborne Publishing Ltd., 408 Wimborne Road East, Ferndown, Dorset BH22 9ND.**
Tel: 01202 873872. Fax: 01202 874562.
E-mail: orders@epemag.wimborne.co.uk **On-line Shop:** www.epemag.wimborne.co.uk/shopdoor.htm
Payments must be in £ sterling – cheque or bank draft drawn on a UK bank. Normally supplied within seven days of receipt of order.
Send a copy of this form, or order by letter if you do not wish to cut your issue.
M08/02

STORE YOUR BACK ISSUES ON MINI CD-ROMS

VOL 6 NOW AVAILABLE

ONLY
£12.45 each
including VAT and p&p



A great way to buy *EPE* Back Issues – our mini CD-ROMs contain back issues from our *EPE Online* website plus bonus articles, all the relevant PIC software and web links. Note: no free gifts are included. All this for just £12.45 each including postage and packing.

VOL 1 CONTENTS

BACK ISSUES – November 1998 to June 1999 (all the projects, features, news, IUs etc. from all eight issues). Note: No advertisements are included. **PIC PROJECT CODES** – All the available codes for the PIC based projects published in these issues.

VOL 2 CONTENTS

BACK ISSUES – July 1999 to December 1999 (all the projects, features, news, IUs, etc. from all six issues). Note: No advertisements are included. **PIC PROJECT CODES –** All the available codes for the PIC-based projects published in these issues.

VOL 3 CONTENTS

BACK ISSUES – January 2000 to June 2000 (all the projects, features, news, IUs, etc. from all six issues). **PIC PROJECT CODES** – All the available codes for the PIC-based projects published in these issues.

VOL 4 CONTENTS

BACK ISSUES – July 2000 to Dec. 2000 (all the projects, features, news, IUs etc. from all six issues). **PROJECT CODES** – All the available codes for the programmable projects in these issues.

VOL 5 CONTENTS

BACK ISSUES – January 2001 to June 2001 (all the projects, features, news, IUs etc. from all six issues). **PROJECT CODES** – All the available codes for the programmable projects in these issues, including those for *Interface*.

VOL 6 CONTENTS

BACK ISSUES – July 2001 to December 2001 (all the projects, features, news, IUs etc. from all six issues). **PROJECT CODES** – All the available codes for the programmable projects in these issues, including those for *Interface*.

EXTRA ARTICLES – ON ALL VOLUMES

BASIC SOLDERING GUIDE – Alan Winstanley's internationally acclaimed fully illustrated guide. **UNDERSTANDING PASSIVE COMPONENTS** – Introduction to the basic principles of passive components. **HOW TO USE INTELLIGENT L.C.Ds**, By Julyan Ilett – An utterly practical guide to interfacing and programming intelligent liquid crystal display modules. **PhyzzyB COMPUTERS BONUS ARTICLE 1** – Signed and Unsigned Binary Numbers. By Clive "Max" Maxfield and Alvin Brown. **PhyzzyB COMPUTERS BONUS ARTICLE 2** – Creating an Event Counter. By Clive "Max" Maxfield and Alvin Brown. **INTERGRAPH COMPUTER SYSTEMS 3D GRAPHICS** – A chapter from Intergraph's book that explains computer graphics technology in an interesting and understandable way with full colour graphics.

EXTRA ARTICLE ON VOL 1 & 2

THE LIFE & WORKS OF KONRAD ZUSE – a brilliant pioneer in the evolution of computers. A bonus article on his life and work written by his eldest son, including many previously unpublished photographs.

NOTE: These mini CD-ROMs are suitable for use on any PC with a CD-ROM drive. They require Adobe Acrobat Reader (available free from the Internet – **www.adobe.com/acrobat**)

Order on-line from
www.epemag.wimborne.co.uk/shopdoor.htm
or www.epemag.com (USA $ prices)
or by phone, Fax, E-mail or Post

BACK ISSUES MINI CD-ROM ORDER FORM

Please send me .... (quantity) BACK ISSUES CD-ROM **VOL 1**
Please send me .... (quantity) BACK ISSUES CD-ROM **VOL 2**
Please send me .... (quantity) BACK ISSUES CD-ROM **VOL 3**
Please send me .... (quantity) BACK ISSUES CD-ROM **VOL 4**
Please send me .... (quantity) BACK ISSUES CD-ROM **VOL 5**
Please send me .... (quantity) BACK ISSUES CD-ROM **VOL 6**

Price £12.45 each – includes postage to anywhere in the world.

Name . . . .

Address . . . .

. . . .

. . . . Post Code . . . .

☐ I enclose cheque/P.O./bank draft to the value of £ . . . .

☐ Please charge my Visa/Mastercard/Amex/Diners Club/Switch

£ . . . .

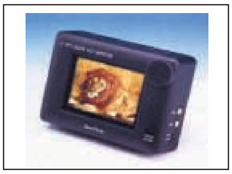
Card No. . . . .

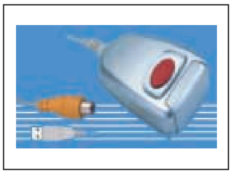
Expiry Date . . . . Switch Issue No. . . . .

SEND TO: **Everyday Practical Electronics,**
Wimborne Publishing Ltd.,
408 Wimborne Road East, Ferndown, Dorset BH22 9ND.
Tel: 01202 873872. Fax: 01202 874562.
E-mail: orders@epemag.wimborne.co.uk

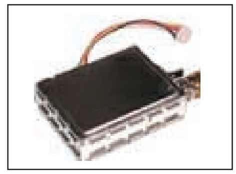
Payments must be by card or in £ Sterling – cheque or bank draft drawn on a UK bank.
Normally supplied within seven days of receipt of order.
Send a copy of this form, or order by letter if you do not wish to cut your issue.

Video Surveillance





• C-MOS B/W Camera 15mm/15mm	£29.00
• C-MOS Colour Camera 15mm/15mm	£65.00
• PCB B/W Camera 32mm/32mm	£24.00
• PCB Colour Camera w/Audio 32mm/32mm	£65.00
• 23cm (1·3GHz) Video/Audio Transmitter	£35.00
• 13cm (2·4GHz) Video/Audio Transmitter	£35.00
• 1·2 Watt 2·4GHz Video/Audio Transmitter	£120.00
• 4" TFT Boxed Colour Monitor w/Audio	£110.00
• Video to VGA Converter	£65.00
• VGA to Video Converter	£90.00
• External USB Video Capture Box	£55.00


All prices exclude VAT.

Many more products on our website:

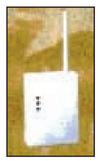
WWW.BITZTECHNOLOGY.COM

Tel: 01753 522 902 Fax: 01753 571 657

Wireless Remote Controls

RC-11
3 channels RC-11 Key fob Control Transmitter with Rolling Code technology.

RC-11 Price: £14.99



UC-216
3 channels receiver with 2 relays output working from 9V to l2vdc UC-216 with learning mode and rolling code technology.

UC-216	9-l2Vdc	£29.00
UC-222	240 Vac	£29.00

Also	Wireless pager alarm PG-4W	£125.00
	Wireless PIR alarm JA-60S	£49.00
	Wireless door magnet alarm	£39.00
	Wireless smoke alarm	£49.00
	Wireless glass break alarm	£49.00
	Wireless alarm control panel	£99.00
	Wireless alarm control panel with digital /voice dialler	£149.00

Distributors Welcome.
All prices exclude VAT and carriage. All transmitters are licence exempt working on 433.92Mhz. All transmitters work with rolling code technology.

Bitz Technology Ltd
sales@bitztechnology.com
+44 (0) 1753 522 902

EVOLUTIONARY ELECTRONICS

CHRIS MACLEOD AND GRANT MAXWELL

A revolution in evolution, and anyone can experiment with it

WE hear about advances in electronics all the time – smaller circuits, faster chips, new devices and technologies. But there's another revolution happening, one that in a few years may change electronics forever and perhaps even lead to the development of the first truly intelligent machines. This new revolution is called Evolutionary Electronics.

One of the most interesting and unusual attributes of this revolution is its accessibility to hobbyists. The answers to the big questions aren't clear yet and the rewards for getting it right are immense. The experiments don't need million dollar machines or laboratories, just access to some good computing equipment and a degree of ingenuity.

A CASE FOR EVOLUTION

The idea is simple. Suppose that we want to make a machine so complex that we don't know how to design it. A good example would be the human brain – the most complex structure in the known universe. Where would we start? Well, we could look to nature; after all she has made incredibly complex machines – just look at us! But she's done this not with conscious design, but through the power of evolution by natural selection.

We all know what evolution is; it's a simple and elegant concept. If you take a population of animals which have random genes and leave them in a particular environment, those with good traits will survive and those that are not as fit will die. The animals which die may have problems like not being fast enough to outrun a predator or not tall enough to reach food.

The better-suited members of the population survive to breed and to mix and pass on their good traits to the next generation. In this way the population gets more suited for its environment and perhaps over many generations evolves into new species.

So, why not do the same with circuits? Set them up randomly, test how good they are (their fitness), and allow the best ones to survive and mix their traits (to breed!). Well, this can be done and it has been done with some very interesting results, as we will see. There are several ways of doing it, but the best known and most popular is called the Genetic Algorithm.

GENETIC ALGORITHM

The Genetic Algorithm, often called simply the GA, works like this. We code the system we want to evolve, in this case our circuit, as a string of numbers (we'll come back to this shortly). We then set up a random population of these strings, usually between fifteen and fifty strings. We test them all to see how well the circuits they represent work (of course, right at the beginning, none of them will work very well). We then make another, new population, out of the old one, by copying across the best strings. The better the fitness of a string in the old population, the more chance it has of appearing in the new one.

Having generated this new population out of the best members of the old one, we allow the strings to "breed" by swapping some of their numbers as shown in Fig.1.

23 42 12 | 89 02 10
65 04 19 | 05 98 34
Members of the population. The line represents the "Crossover point"

23 42 12 05 98 34
65 04 19 89 02 10
This is the pair after they have bred. Notice that they swap some of their information generating two new strings.

Fig.1. Strings breeding.

This generates new strings. The idea is that some of these new strings will have the good traits from both parents and so be better than either.

The final part of the algorithm is called "Mutation" and is designed to add some variation into the population by introducing some new numbers to it. It simply involves choosing a few numbers from the strings and changing them by adding a random element. The algorithm is then repeated and after a few generations the circuits become fit enough to fulfil their functions (that's the theory anyway). You can use either real numbers as shown above or binary numbers.

CHOOSING GA VALUES

This sort of technique is particularly useful for designing circuits like filters. All you need is a software simulator. You can use the GA strings to generate netlists, and off you go. For example, take the sort of circuit shown in Fig.2.

You can set up the string as shown, fill it with random numbers, generate a population and watch it evolve. In this case, the fitness of the circuit is how close the simulated response is to the desired response as designed by you.

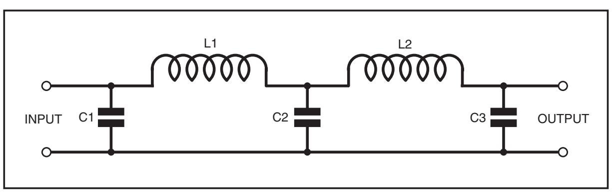


Fig.2. A typical filter circuit. In this case each string could be: L1, L2, C1, C2, C3.

CHOOSING GA CONNECTIONS

You can also get the GA to choose the wiring of the circuit (and even the components if you want). This time, rather than component values, the GA chooses which components are connected to which others. One way of doing this is shown in Fig.3.

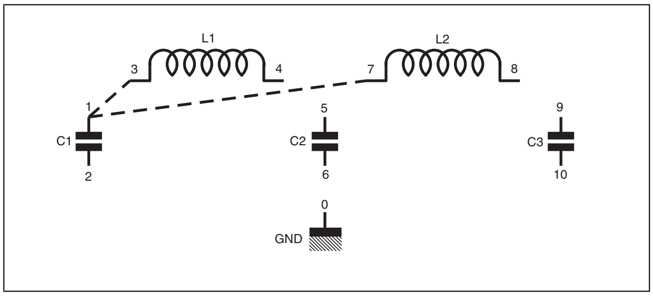


Fig.3. Letting a GA choose wiring topology.

Each wire in the circuit is given a node number. In this case a 11-bit number can encode the connections to a particular node. For example, if node one is connected to nodes three and seven as shown, the code would be 00010001000 the position of the "1"s being three and seven (the first node is zero). The total string length would then be eleven times eleven (one connection code for each node of the circuit) or 121 bits.

You can think up other schemes easily; for example, the algorithm can, if you code it to, choose both the wiring and the component values at the same time.

EVOLUTIONARY ALGORITHMS

Although Evolutionary Algorithms like the GA are useful for choosing components in complex circuits where tradeoffs have to be made, like filters, their real promise lies in Artificial Intelligence. The two tenets of Evolutionary Connectionism (using Artificial Evolution to make networks of components in an attempt to create AI) can be stated as questions and answers:

Q. *Is it possible to build a machine which is intelligent?*

A. *Yes, the brain is simply a machine and if nature can do it, eventually so can we.*

Q. *Is it possible to make a machine like this even if we don't understand how it works?*

A. *Yes, nature used evolution to build it and again so can we.*

Genetic Algorithms and their kin are being used right now to create Artificial Neural Networks (known as ANNs). These are networks of small processors, modelled on brain cells that can learn from experience, just like a real brain. Although these experiments have been quite successful in some respects there are many problems left to solve. After all, we haven't succeeded in making a brain yet.

PROBLEMS

Although there have been some huge projects to try and produce large intelligent systems using Artificial Evolution (like that by Hugo de Garis, which produced circuits with literally millions of elements in them) none have really succeeded yet. That's not to say that there haven't been some interesting results (Adrian Thompson for example has succeeded in evolving large numbers of digital gates into a circuit which did some interesting and unusual things).

So where is Artificial Evolution failing? The answer probably lies in the difference between it and "real" – biological – evolution. At first they may seem similar, but in reality there are some critical differences.

In biology, the wiring pattern of the processors (brain cells) is not coded directly into the DNA itself, in the way illustrated above in electronic circuits. In fact, the DNA in your cells doesn't hold enough information to directly wire-up even a small part of your brain. No, its action is different – and more subtle; what DNA actually does is produce Proteins.

PROTEINS

The Proteins are the universal machines of biology – your body is made up from them. They can react chemically, form rigid structures or perform a multitude of other tasks and, critically, they can self-organize like automata or a jigsaw puzzle into a greater and more complex whole.

So the system has two components: a code and the self-organizing machines which the code specifies. Because the machines can self-assemble, they can produce more complex structures than the code. Another important point is that because all these proteins are made up of only 20 different structural units, they can mutate into different forms easily and so evolution becomes possible. And, unlike the GA, whole sections of real DNA can be deleted, pasted into the wrong area or even reversed, by copying incorrectly – so adding more variation to the whole system.

FURTHER TRICKS

There are further tricks too. Proteins can actually lock to the parent DNA and stop it producing more of the protein (or a different protein). So parts of the code can be switched on or off. Released proteins set up "gradients", which in turn inhibit or excite other proteins in the organism building up patterns of material. In this way smaller and smaller details can be built as one protein triggers another; these symmetrical patterns of structure, reproducing at different scales, are sometimes called "fractal".

The result of all this activity is that the physical structure produced is not homogenous but modular, with delineated identifiable regions that perform specific tasks.

MODULAR SYSTEMS

The fact that the genes in your body produce a modular system is important. There is conclusive evidence to suggest that the brain is modular – for example, if you damage one part of it, you usually wipe out a very particular function.

Electronic systems too are modular; after all, you don't start designing a radio system out of one enormous mass of components; you start by designing oscillators, amplifiers, mixers and suchlike separately. The reasons for brain modularity are complex and not all of them are clearly understood as yet.

So, can we conceive of a system which can evolve modularity in this way? The answer is obviously yes, but the biological system is so complex that it may be almost impossible to reproduce accurately and the organizational element of processors is several levels removed from the protein level anyway; so we must turn to other methods capable of evolving modularity.

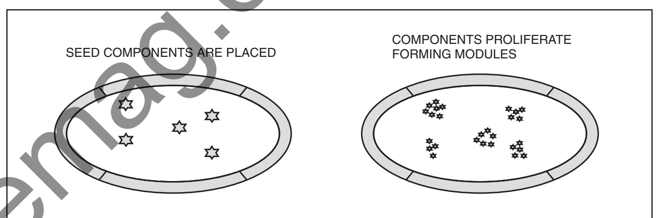


Fig.4. A biological approach to Evolutionary Electronics.

SOLUTIONS?

There are several different ways to introduce modularity into artificial evolution.

Firstly, we could try and code it into the string of the GA itself. This could perhaps be achieved by splitting the GA string into substrings, each of which represents a module. A similar proposal is to have a local string assigned to different parts of the network controlling how it evolves. This is representative of how the genes switch each other on or off as described above.

Another possibility is to try and model biology, not at the DNA level, but at the cellular level, by mimicking the way tissue is placed in the developing organism. This process is shown in Fig.4. The components first migrate to their places in the structure, then proliferate (become more numerous), then finally the wiring is set up locally (by an evolutionary algorithm).

From a pragmatic point of view, why not simply add modules to the circuit, as shown in Fig.5a and allow them to be wired locally by a Genetic Algorithm as in Fig.5b or Fig.5c? These are only some of the possibilities; others include using fractals, automata or special treelike rules to complete the circuit. Whatever way is chosen, it is likely that not only the circuit will have to grow, but also the system which it is controlling at the same time. This is

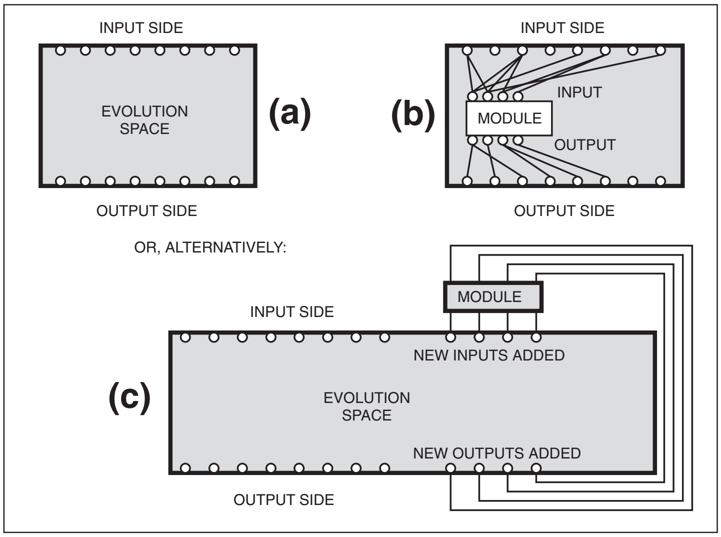


Fig.5. A more pragmatic approach.

because the brain of an organism did not evolve in isolation, but as part of the animal as a whole.

The exciting thing about Evolutionary Electronics is that we don't know the answers yet and the experimentation lies within the ability of the ambitious amateur. Not only that, but this area could hold great rewards for the future of electronics.

ACKNOWLEDGEMENT

The authors would like to thank Ann Barbara Reddipogu, Sethuraman Muthuraman, Niccolo Capanni and David McMinn for their contribution to this article. ☐

FURTHER READING

Simple Evolution: R.L. Haupt and S.E. Haupt, *Practical Genetic Algorithms*, Wiley, 1998.

Modular Evolution: C. MacLeod et al, Evolution and Devolved Action, in appendix B of: D. McMinn, *Using Evolutionary Artificial Neural Networks to Design Hierarchical Animate Nervous Systems,* PhD Thesis, The Robert Gordon University, 2002, available on request from the authors: email **chris.macleod@rgu.a.uk** or **g.m.maxwell@rgu.ac.uki**.