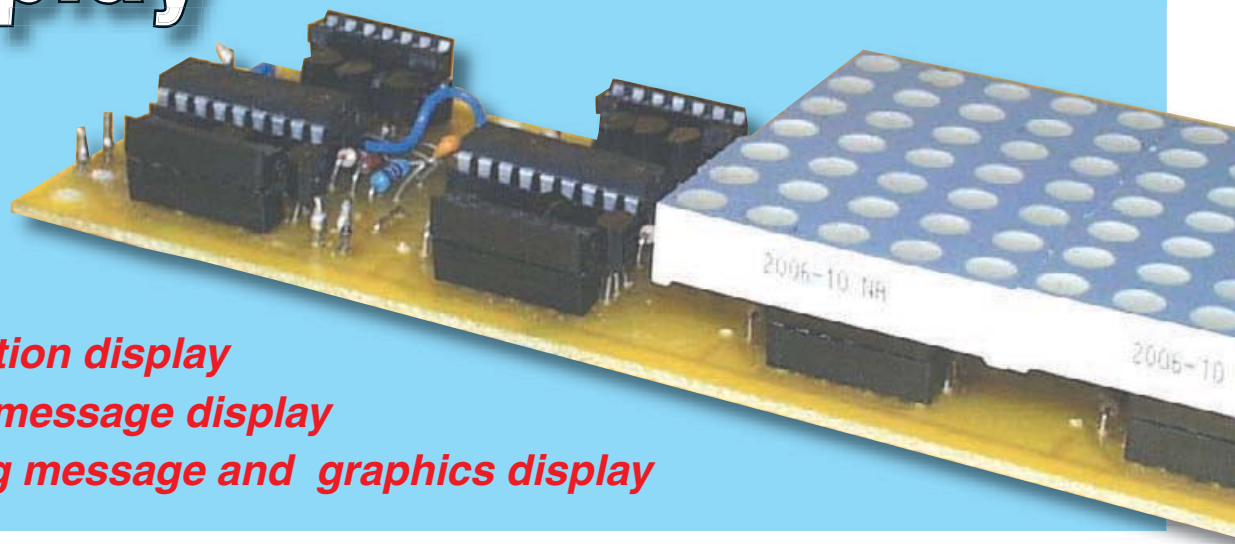


Giant LED Message Display



- * **Animation display**
- * **Static message display**
- * **Moving message and graphics display**

Proclaim your message loud and clear in lights!

By John Becker

THIS Giant LED Message Display unit provides the following functions:

- Moving message and graphics display
- Static message display
- Animation display

It is in modular form and can be built pretty much to what ever size and format one prefers, as discussed later. It is PIC microcontroller controlled, programmable for its displays type and content from a PC computer.

Once the system has had the basic information sent to it, the system can be disconnected from the PC. The data is retained indefinitely even after switch-off, and can be recalled after switch-on again by means of three function switches on a master control box.

The PC provides the basic data via a serial connection (it is not designed for USB use) to a master PIC, which in

turn stores the data, and passes on the information to a series of slave PICs, each of which controls an LED display module comprising a 5×7 matrix of LEDs. Each LED matrix module is 52mm high and 38mm wide. Its format is suited to displaying alphanumeric messages, graphics or animation displays, or a mixture all three.

Display formats

There can be up to 128 such display modules, each controlled by its own slave PIC. The display printed circuit boards (PCBs) are also modular and each holds four slave PICs and four LED modules. They have been designed so that they butt up against each other, allowing an overall display to be arranged linearly, or as a square or rectangle.

Each finished display PCB measures 152mm (6 inches) long, and so a fully linear arrangement of 128

LED modules would be 4.86m (16 feet) long! A more conventional rectangular arrangement of, say, 24 LED displays wide and five displays high, would produce a display measuring 91cm wide \times 26cm high. The choice of arrangement is up to the user.

Previous message display

The last time *EPE* published an LED Matrix Message Display was in May-June 1994, by Julian Ilett and Brett Gossage. This comprised 448 separate LEDs arranged on a single PCB in a 64 columns by seven rows format, measuring about 10.2cm \times 58.4cm. The display was controlled by a Z80A microprocessor and peripheral devices.

Changes in technology now allow the use of PIC microcontrollers and LED display modules (saving an awful lot of LED soldering!). The use of PICs and LED

modules also makes the overall control somewhat easier, as well as allowing for various modular arrangements.

Whereas the previous design used a mixture of in-built messages and a simple keypad plus a separate PC interface for creating data, PICs allow for easy interfacing to a PC and its serial communications options, RS232 in this case, for the creation and transmission of data.

Slave controllers

As with any LED, the current flowing through it must be limited. The maximum current allowed for the chosen LEDs depends not only on their characteristics, but also on those of the devices controlling them. The initial factor considered here is what current the PIC's control pins can source or sink. The maximum limit for the current sunk by any pin of a typical

anode control lines must use a current-boosting interface to provide the needed current.

Referring to the Slave Control circuit diagram in Fig.2, transistors are used as the interface, one for each column, TR1 to TR5. These are used as emitter followers, their collectors jointly connected to a common power supply line. Current can flow through any transistor from its collector (c) to its emitter (e) when a control voltage and current are applied to its base (b), as provided by PIC pins RA0 to RA3, plus RA6, provided there is a path through the LEDs.

This path is completed when the cathode of any LED is connected to a current sink pin of the PIC, via a current-limiting resistor. PIC pins RB0, RB3 to RB7 and RA7 control these cathode paths. The choice of PIC pins has been dictated by the convenience of routing connections on the PCB.

The current flowing into each cathode controlling PIC pin is limited by resistors R1 to R7. Their values have been selected so that the total current flow cannot exceed the PIC's limits in the event of all LEDs in a column being turned on simultaneously. Normally, the cathode-controlling PIC pins are held high (logic 1) when the LEDs are to be turned off. To turn on an LED to which anode current is provided, the respective PIC pin is taken low (logic 0).

Rate of change

The rate at which the columns and rows are activated is sufficiently fast that the eye does not perceive a noticeable flicker in the LED intensities. It is immaterial in which order the anodes

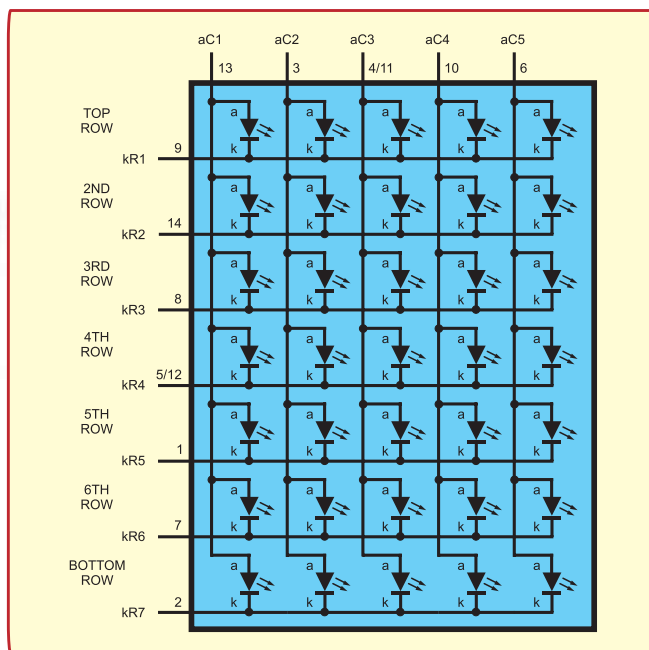
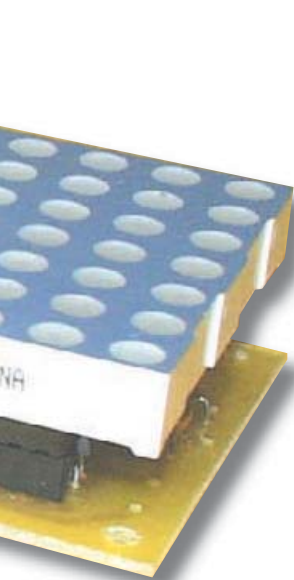


Fig.1. Internal structure of a 7 x 5 matrixed LED module

LED matrix modules

Before discussing any part of the control circuits, it is worthwhile examining first what it is that has to be controlled – the matrixed LED modules. As explained earlier, the type using 35 LEDs has been chosen. They are arranged in what is known as the *common anode* format, as shown in Fig.1.

The LEDs are internally connected so that there are five columns of them, each column having its own common connection to the anodes of the LEDs within it, aC1 to aC5 in Fig.1. The cathodes of each LED are connected so that seven rows are formed, which are each brought out as common cathode connections, kR1 to kR7. Any cathode-controlling code from the controller is applied to the cathodes of each LED in that row, but the LED only becomes active (lit) if the anode connection in the column to which it is connected is powered. Thus a single control byte of seven bits is required for the cathodes, and a single byte of five bits is required by the anodes.

PIC is 25mA, and that provided as a source is 20mA.

Clearly, the anode current required by the LEDs must be greater than 20mA in order to provide adequate brilliance when seven of them in a column are all turned on. Consequently, the PIC's

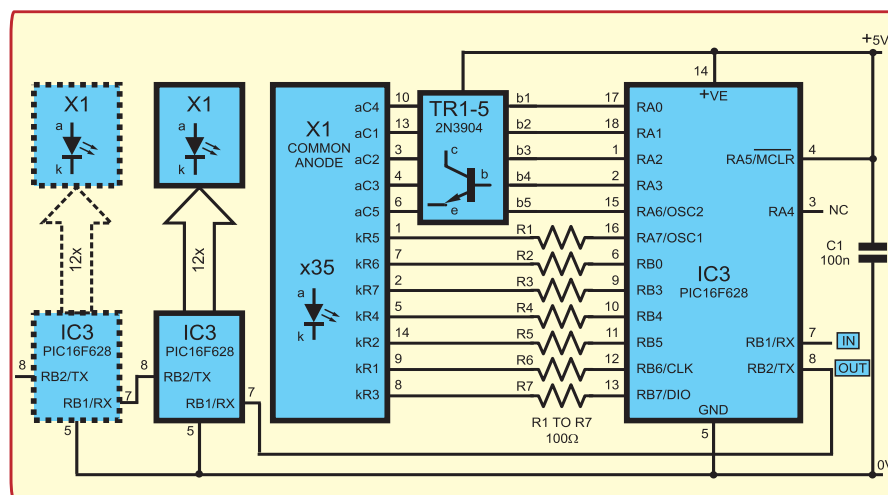


Fig.2. Circuit diagram for the slave control. Note TR1 to TR5's collectors (c) go to the +V line; the bases (b) to the PIC and emitters to display module

are turned on, provided the applied cathode turn-on code is synchronised to change appropriately for each active column.

Each LED matrixed module and its controlling PIC are connected identically, and indeed these PICs are identically programmed. The PICs control their displays either from data held within their memory, or from data continually supplied by the master PIC, at a rate set by the user.

Each slave PIC operates at a clock frequency of approximately 4MHz, as generated internally according to the configuration code with which they have been programmed. They do not need an external clock control source.

There is inevitably a minor clock rate difference between each Slave PIC. This is normally of no consequence, but a synchronisation of the software routines within the PICs can be activated for animation display routines that need it. More on this later.

Master control

The circuit diagram for the Master Control is shown in Fig.3.

There are two devices in this circuit, the master control PIC, IC1, and an RS232 interface, IC2, for connecting to a PC. The RS232 device has featured in many *EPE* PIC projects and will not be discussed further here. Suffice to say that it allows the PIC and PC to 'talk to each other' bi-directionally, via connector SK1.

The PIC takes the user-generated message and animation data from the PC, using handshakes while doing so, at a baud rate of 9600. Moving message and graphics data is stored in its internal memory for use on request. Static and animation data is sent on to the slaves. In the case of static messages, all PICs receive the data serially, but only store it for future use if the data is not a blank space in the text, allowing animation or static data to be shown in that space if preferred.

The animation data is intended for use by specified PICs and displays in the chain of slaves. PIC address data is embedded in the code and only the PIC whose address corresponds will store that data for future use, it otherwise passes it to the next PIC, which does likewise, and so on until the required destination PIC is reached. There is no handshaking between PICs during

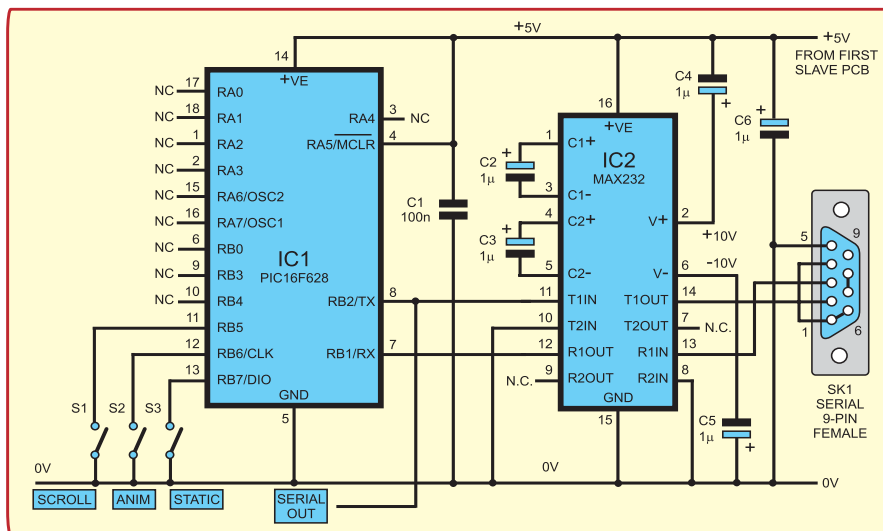


Fig.3. Circuit diagram for the Master Controller. The serial out is routed to the first slave circuit at IC3 pin 7

this process. The same, or different animation data may be sent to other PICs by their address codes, as told to the PC by the user.

Character map

Regarding the message text data, the codes sent to the master PIC by the PC are the ASCII codes for those characters, alphabetic, numeric and punctuation data etc. A table showing the data which can be displayed is shown in Fig.4.

Information about the columns and rows in which the data is used is held by the master PIC as a lookup table. In all cases, when a particular character is being processed by the master, it goes

to the corresponding section of the table and allocates the specified row and column data as 8-bit byte values. There are five display columns used for each character, and thus five bytes are produced, plus an additional zero value byte to provide a space between each character/letter in moving message mode.

The font used in the displays is basically a copy of what would be seen on a normal alphanumeric LCD display. No provision to change it has been made. The data is held in file LCDMTX1.INC should you wish to examine it. Moving animation is dealt with similarly, except that the data has already been coded within the PC since look-up tables are not appropriate to what is essentially random data, as the graphics data depends entirely on what the user wants.

When running a moving message, sending it on to the slave chain, the master sends the five display bytes to the slaves with a user-selectable delay between each byte. On receiving each byte the slave shows that byte in the righthand column of the display. On receipt of the next byte, it transfers each of the column bytes to the next one to the left, and then stores the new byte in place of the previous one.

In doing so, the previous left-most byte of the first display is passed to the next PIC and its display. The process continues for each of the stored characters and their bytes, and then repeats. The result is a very effective repeating moving message display, traversing the display area right to left at the user's chosen rate.

	0	@	P	'	p
!	1	A	Q	a	q
"	2	B	R	b	r
#	3	C	S	c	s
\$	4	D	T	d	t
%	5	E	U	e	u
&	6	F	V	f	v
'	7	G	W	g	w
(8	H	X	h	x
)	9	I	Y	i	y
*	:	J	Z	j	z
+	;	K	[k	{
,	<	L	\	l	
-	=	M]	m	}
.	>	N	^	n	£
/	?	O	_	o	

Fig.4. Character library available for use

Power supply

The current capacity of the power source needed depends on how many display matrix modules are used. With all LEDs turned on, the displays each consume about 150mA. The voltage range of the power source is from 7V to about 9V DC. The 7V lower limit allows for the 2V 'headroom' needed by the 5V regulators used in the design. 9V still prevents the regulators from getting hot and thus needing a heatsink.

Although the transistors, TR1 to TR5, could handle a collector voltage of greater than 5V, there would be the danger of exceeding their 200mA limit. For the same reason, resistors R1 to R7 should not have values less than 100Ω. This value also prevents the current flowing into the PIC via the cathode lines from exceeding the PIC's limits.

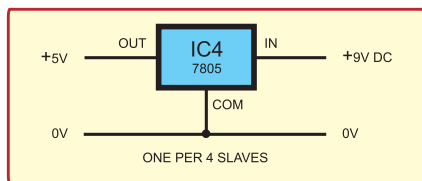


Fig.5. The +5V regulator circuit diagram. One per four slaves is required. It also powers the master circuit. The regulator appears on the first slave section on the PCB – see Fig.7

It is recommended that high-brightness LED modules are used to give good visibility in daylight without the need for high currents.

Each slave PCB, with its four PICs and displays, has its own on-board 5V regulator, shown in Fig.5 as IC4. The power supply should be connected to each regulator on the slave boards by wires radiating out from the power

supply, in 'star' fashion. It would be unwise to serially connect the PCB supply lines to the PSU, as this could cause undesirable current and voltage spikes in the lines as the LEDs are turned on and off.

It would, in fact, be feasible to use just one regulator per two slave PCBs. The

total maximum consumption for each board is about 600mA, with all LEDs turned on, consequently a 7805 regulator rated at 1.5A (rather than 1A) will cope well with the maximum possible current. It is highly unlikely anyway that all LEDs will be on together, unless you deliberately create that situation.

Parts List – Giant LED Message Display

Master Control

- 1 PC board, code 594 (Master), available from the *EPE PCB Service*, size 45.7mm x 35.6mm
- 1 plastic case, size and type to choice – see text
- 3 SPST toggle switches (S1 to S3)
- 1 9-pin D-type serial connector, female, plus cable – see text (SK1)
- 1 16-pin DIL socket
- 1 18-pin DIL socket
- 11 PC solder stakes

Semiconductors

- 1 PIC16F628 microcontroller, pre-programmed – see text (IC1)
- 1 MAX232 RS232 serial interface (IC2)

Capacitors

- 1 100n min ceramic (C1)
- 5 1μF radial elect. 16V (C2 to C6)

Multistrand connecting wire; mounting hardware to suit – see text; solder, etc

Slave Display

(one set needed for each display module, unless indicated)

- 1 PC board, code 595 (Slave) – four displays, available from the *EPE PCB Service*, size 150mm x 51mm
- 2 14-pin DIL socket – see text
- 1 18-pin DIL socket
- 1 2-inch, high-brightness, 35-LED matrix module – see text (X1)

Semiconductors

- 5 2N3904 NPN transistors (TR1 to TR5)
- 1 PIC16F628 microcontroller, pre-programmed – see text (IC3)
- 1 7805 +5V 1.5A voltage regulator – only one required for each group of four displays, see text (IC4)

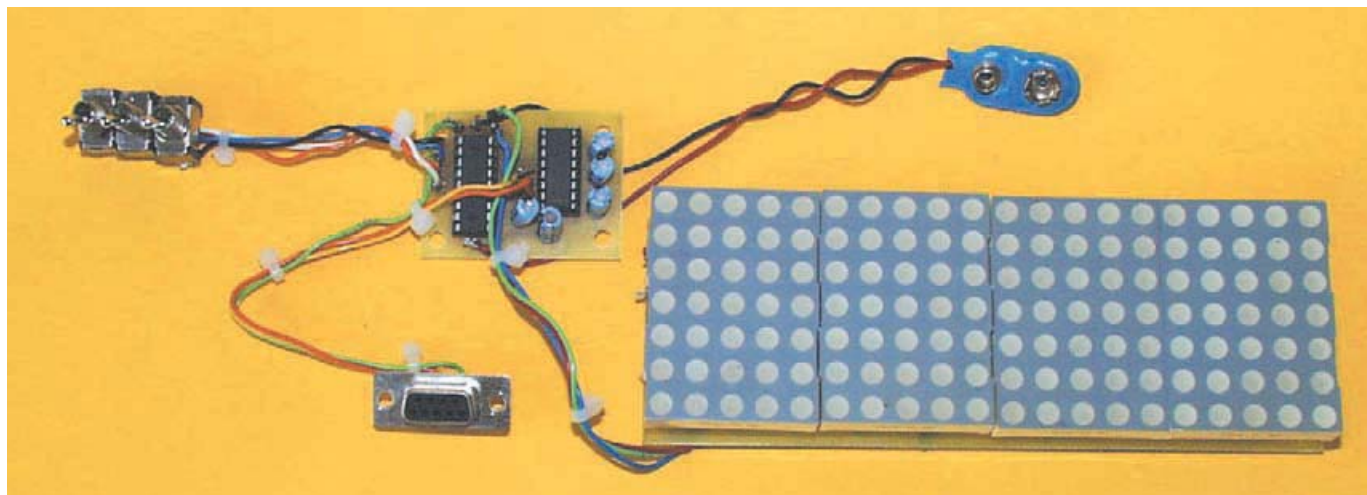
Capacitors

- 1 100n min ceramic (C1)

Resistors

- 7 100Ω 0.25W 5% (R1 to R7)

Multistrand connecting wire; mounting hardware to suit – see text; PC solder stakes; solder, etc



Construction

The Master control unit has its own PCB. The Slaves have one PCB for each group of four PICs and their displays. The assembly details for the boards are shown in Fig.6 and Fig.7 respectively, along with their track layouts. These boards are available from the *EPE PCB Service*, codes 594 (Master) and 595 (Slave). As implied, you only need one master board, plus one slave board for each group of four displays.

There are no special concerns for the master board, but do take the usual care over component soldering, and observing the correct polarity of the components. Use DIL (dual-in-line) sockets for the two ICs, and note that they are static-sensitive. You should discharge static electricity from your body by touching the bare metal of any earthed item of equipment before handling any of the DIL ICs in this design.

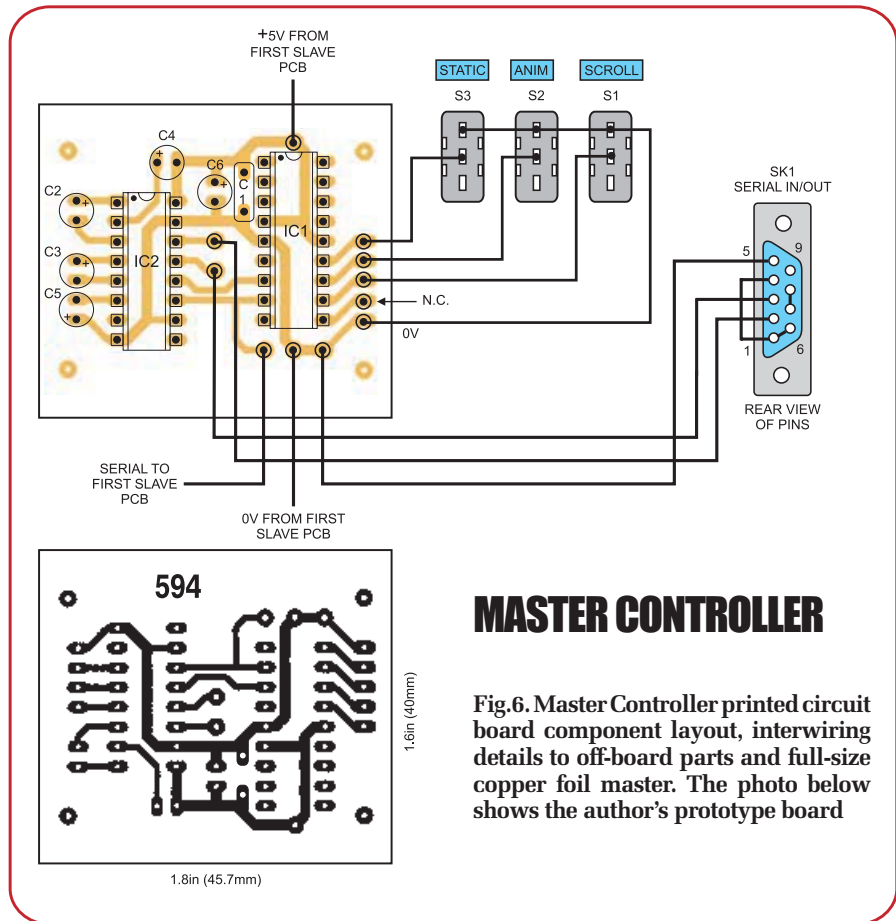
Display Slave PCB

With the slave board, all components are positioned below where the displays are mounted, and there are a number of link wires needed. Connect these first. Again use sockets for the PICs. Assemble each board in order of ascending component size. The pinouts for the transistors and regulator are shown inset. Note that the regulators are mounted on the *rear* of the PCBs because of their size.

The LED modules first need two 7-pin SIL sockets (cut lengthwise from a 14-pin DIL socket), and then a further 7-pin SIL socket plugged into each. This allows clearance of the components by the displays. Be very careful when inserting the displays that you do not bend their pins. Note that the displays are shaped at their sides, with their orientation as shown in Fig.7. The pin notations for the displays are given in Fig.8.

The board has been designed to be slightly smaller than the display area of the LED modules when mounted, to allow close mounting of the boards in the final display format that you choose. You will need to choose a material on which to mount the display boards, and suitably enclose them if preferred.

The master board can be mounted in any small plastic box of choice and adequate size. Holes must be drilled for the switches, serial connector SK1, and the connecting wires to the slaves. Note that this board takes its 5V power supply from the first slave board. The wires from the PSU to the



slaves must be routed according to the needs of your layout, providing adequate securing, and also preventing the wires from becoming a trip hazard to other people.

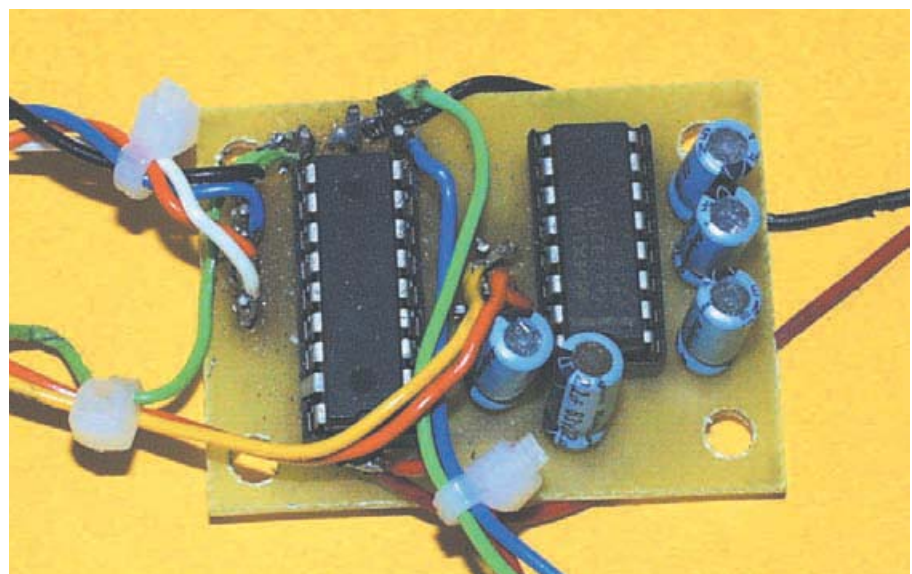
First tests

Software and pre-programmed PICs for this design are available as stated later.

On completion of the PCB assemblies, and without DIL ICs or LED modules

inserted, connect the boards to the power supply. After switching on, check that +5V is present at the output of each regulator, IC4, and at the +VE connections of the sockets for the PICs and IC2, with the common (black lead) of the meter on the relevant 0V (GND) point.

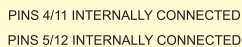
If any corrections are required to your assembly, or when inserting or extracting PICs etc, make sure the PSU is switched off, and allow a few



TOP VIEW IC4 IN e TOP VIEW TR1-5
COM b
OUT c

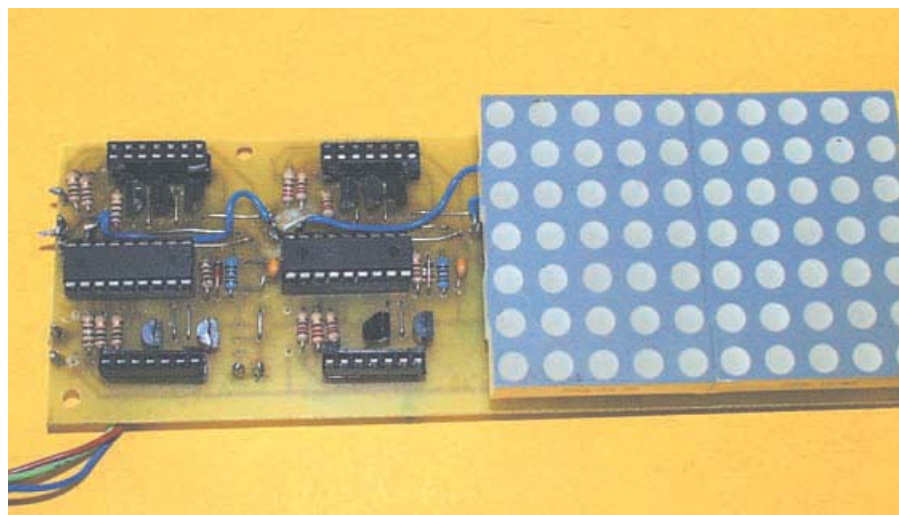


NOTE POSITION OF
LED MATRIX MODULE
INDENTS/ OUTDENTS



On power-up, the master PIC reads the status of the switches and reacts accordingly, in this case, as S1 is switched on, by starting the flow of the moving message with moving graphics

as programmed into the PIC's memory. Progressively, the moving message and graphics will flow from the master PIC and along through the slaves, being displayed as it progresses.



When satisfied with the first test, insert the pre-programmed PICs, RS232 chip IC2, and the displays, ensuring their correct orientation. Switch on S1 (Moving Message) and connect the power. Double-check the +5V power lines again.

The message is MIND THE CAR, followed by a simple caricature of a car, as copied from the header of the original 1994 article. The message and graphics are repeated indefinitely, at a rate built into the master PIC's program at this stage.

Switch off S1 to stop the message and clear the displays. Switch on S1 again to restart the message. No other switches will work at this time since no static messages or animation data have been sent from the PC yet.

Next check

Now check the power supply to the RS232 chip and its voltage outputs. There should be +5V on its pin 16, 0V on pins 15, 10 and 8. +10V should be present on pin 2 and -10V on pin 6, being generated internally by the device itself.

You can now connect the unit via a suitable standard modem serial lead (non-crossover) to whichever PC COM socket you prefer (as said earlier, the unit is not designed to run under USB).

Having obtained and copied the PC software into whatever hard-drive you prefer, including a partitioned drive, into a folder named as you wish. Double-click on the PICMessage.EXE file to launch it, during which a few working files are created. You will be told if the PC encounters any problems at this time.

Moving message screen

On completion of the setup, the blank Moving Message screen will be shown, similar to that in Fig.9, but without the graphics or text data.

At the top is a series of small boxes into which you put the graphics data by clicking on them. There are basically 40 boxes horizontally by seven boxes high, representing the LEDs in eight LED matrix modules.

On entry to the screen, all boxes are a uniform grey colour. Left-clicking the mouse on any box changes its colour to an orange shade. Clicking the mouse again on that box returns it to the grey colour. It would have been preferable visually if Visual Basic (VB6), the program in which the software is written, had allowed circular boxes to be clicked to change their colour, but it does not, so it had to be square ones, which do allow the facility.

Below the squares are three bargraphs. These allow you to choose the points between which data is to be saved, and to indicate where further boxes may be inserted if you wish, losing the equivalent

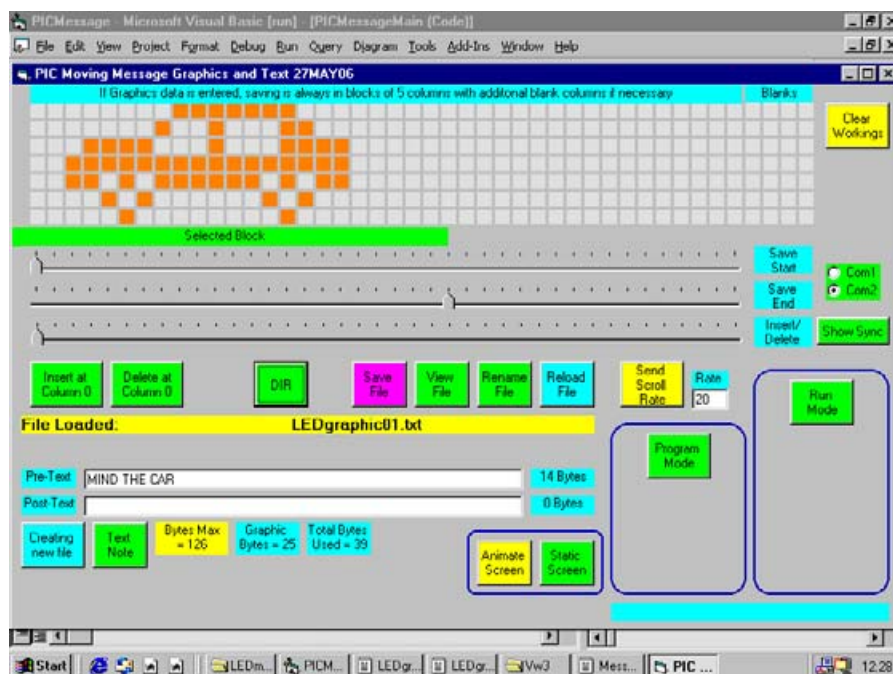


Fig.9. Main moving message screen, with author's test data

number of boxes at the right. The buttons to the left below the bargraphs allow insertion or deletion of graphics data in the first column position.

The chosen columns of graphics data are further confirmed by a green marker below the boxes, expanding or contracting as appropriate. Graphics data is saved in groups of five columns, corresponding to those in the displays themselves. At times, this means that the Blanks area at the right of the main boxes will be used to complete a group of five columns.

Also below the bargraphs are several other control boxes. More on these shortly. Below those is a horizontal box in which the name of any file loaded from the directory is displayed.

Below that are two boxes into which text can be entered. The upper box holds the message, if any, you would like to precede the moving graphic. The second holds the message, if any, you would like to follow the message. Either box can contain blank spaces (space bar) if you wish to do so.

Message length

There is a maximum total message length, including any graphics data, that can be handled by the master PIC. The limit is 126 bytes that it can store.

Alphanumeric and punctuation symbols, which are accessed through the master's font file mentioned earlier, take one byte each. Each column of graphics data takes another byte. The total number of bytes used for any

message being built up is displayed in three boxes below the text boxes.

The remaining two bytes of the total 128 bytes that the master PIC's data memory can hold, are automatically taken up by the values for the total message length, and for the rate at which it is to be scrolled across the display area, as you set into the Rate box towards the right. The rate is automatically sent when the full data is sent, but it may also be sent when the message is running, immediately changing the rate of scroll. Click Send Scroll Rate to do so.

To save a newly created message file, click on the DIR button to open the Directory screen (Fig.10) which has an area near bottom in which you enter the file name you want to use, then click the Make File option. This creates the file by that name. It cannot overwrite previous files if file names conflict. You will be told of such conflict and another name can be chosen.

Having created a file, it is automatically opened in Notepad to confirm that it has been created. Close this file in Notepad's usual way.

Full details of the Directory screen's options can be read through its Notes file. The details will not be discussed now. There are several options available. Click again on the DIR button, select the file from the file names offered by double clicking on its name. This closes the DIR screen and opens the file into the LED Message screen routines.

You can then save your new message into that file, by clicking the Save File button on the message screen.

You can be reminded of the process needed to create a new file by clicking the screen's Creating New File button.

Sending graphics and moving messages

To load and send a file's contents to the master PIC, ensure that switches S1 to S3 on the unit are switched off, then click DIR and select the file you want. Its contents will be loaded to the screen's graphics and text areas.

Plug in the serial cable to the COM port connector on the PC that you wish to use. Then on the program's main (moving message) screen click either the COM1 or COM2 "radio" button towards the top right, as suited to the COM port connector chosen. This choice is also stored to disk for future recall.

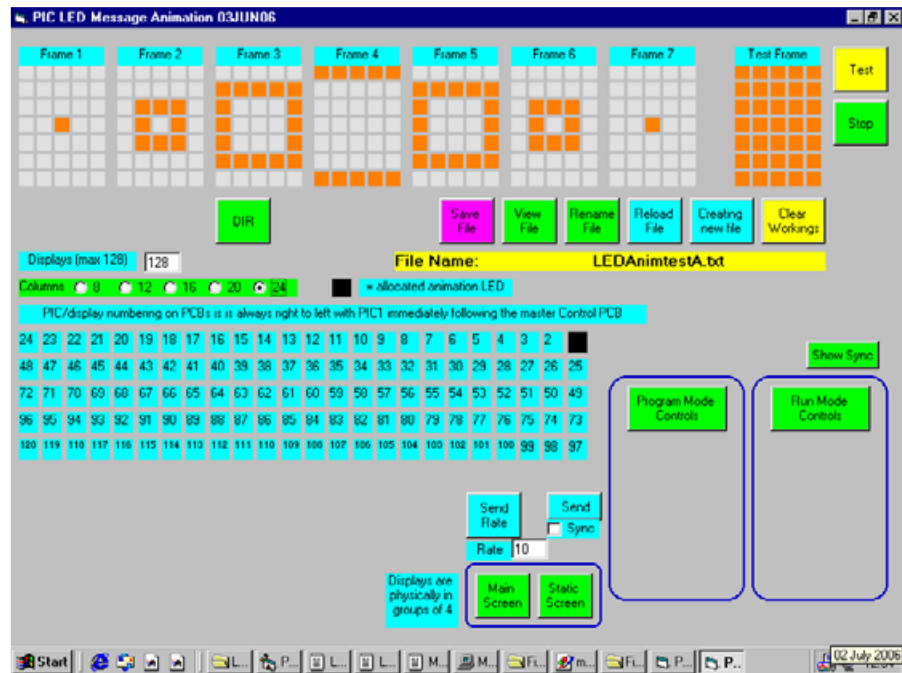


Fig.12. Animation screen, also with author's test data

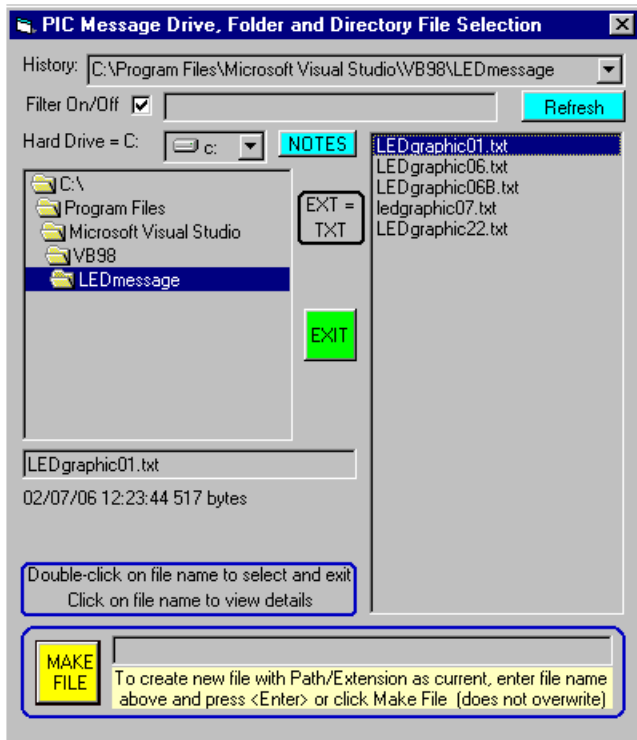


Fig.10. The directory screen

Now, with the PIC unit's power switched on and with the serial cable connected, first click on the Program Mode button towards the bottom right. This action alerts the master PIC to the fact that new data is about to come in.

It also causes four other buttons to be displayed (Fig.11). The only one that concerns you right now is the Send Moving button. The other three buttons

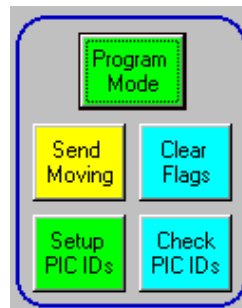


Fig.11. Program mode zone on moving message screen.

are discussed later. Clicking this button causes the loaded data to be sent to the master PIC, which stores it.

Switch on S1 and the displays will start to show the message and any graphics data you have created. As before, S1 allows the message to be started or stopped.

Data facts

It is worth noting that sending a new message to the master PIC automatically overwrites the existing data. This means that the original moving car message will be overwritten the first time you use this option. Should

you wish to reinstate the original programmed message, select file LEDgraphic01.txt via DIR and send that to the PIC.

Any file you have created can be amended, reloading it first and then re-saving. If you wish to rename a file, use the Rename button, which provides the facilities. The View File and Reload File buttons cause those options to occur. In the case of the View File button, the file is opened through Notepad, through which you could amend it if you prefer, although understanding the graphics information will not be obvious at first glance.

If viewing a file in this way, note that the file length and scroll speed values are quoted at the top. It is probably unwise to amend the values here.

When creating graphics data, the Clear Workings button can be used to clear the screen of any data entered.

Ignore the Run Mode button for the moment, it will be discussed later.

Animation screen

Near the bottom right of the main (moving mode) screen are two buttons, Animate Screen and Static Screen. Click the Animate Screen button to display the screen through which animation data is created and sent to the PICs. The term *animation* in this instance means display data in a given LED display module which changes at a user set rate. It never changes LED module position.

An example of animation is available via the DIR button. Click and select file LEDanimTestA.txt. When this is loaded, its data is shown in the seven groups of boxes at the screen's top (Fig.12). Each represents one 'frame' of animation data. Click the Test button to see the animation displayed on the PC screen in the far right display boxes. Click Stop to stop the display.

A similar action will take place on a selected LED module when data has been sent to its controlling PIC, at a rate of change as set via the Rate box near the screen's bottom, a value of 10 at the current time, but may be changed within its screen box by clicking the mouse on it and using the keyboard numeric keys.

Display module selection

The PIC to which the data is sent is selected via the blue boxes. These are numbered to correspond with the slave PICs in the display chain (which, as described later, will have their identities sent to them by the PC and master PIC.

The PIC and its display to which data is sent is indicated by the black square in amongst the blues, PIC 1 at present. Clicking any of the other blue blocks will cause that box to change colour to black, and return the previous black box to blue.

Display quantity selection

First, though, several pre-conditioning actions must take place. The PC must be told how many PICs there are in your chain. Cosmetically, and of no importance to the PC, the boxes can be re-arranged to the display format you have chosen, but only in a regular format of rows and columns.

In the Displays quantity box toward the upper left of the screen, enter the quantity of the displays you are using, from four to 128 is the range, in multiples of four. Then click one of the 'radio' options to select the number of displays in each column of your display format. The choices are 8, 12, 16, 20 or 24 columns.

The boxes will be re-arranged on screen to suit the selection, limited to the physical possibility of that arrangement, for example, selecting 12 displays and eight columns does not work, nor does eight displays and 12 columns! The screen will do what it can to meet your needs, blanking areas if the format is inappropriate.

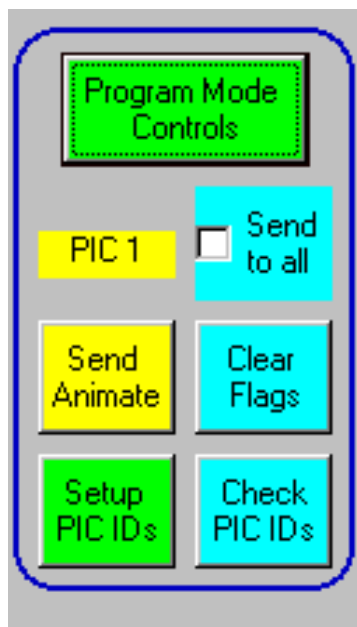


Fig.13. Program mode zone on animation screen

Having chosen a suitable display quantity and format, select the display you want to animate, the first display is suggested right now.

Slave PIC IDs

Next the slave PICs must be told in what order number they are. All are programmed identically by the same software and without identity. With power to the units switched on, click this screen's Program Mode Controls button to reveal several other buttons (Fig.13).

Click the Setup PIC IDs button. This sends a Setup command to the master PIC which initiates the process. It sends a Setup command to the first slave PIC, along with the ID value of 1. The first Slave PIC in the chain accepts this ID of 1 as being its own, and stores it in memory. It then sends the Setup command to the next PIC, incrementing the ID value to 2, and sending to the next slave PIC, which accepts it as its own ID and stores it. So the process ripples along the whole chain of slave PICs, for as many as there are, up to 128.

From then on all PICs now know their identities. And at the moment that they receive them, they output that value to the displays as a binary value affecting all five columns, which can be examined if you wish. At any time you may examine the IDs held by the slave PICs by clicking the Check

PIC IDs button. (For reasons which cannot be determined, occasionally the PICs lose or have their identities corrupted, which then have to be re-sent by the PC. So it's worth checking them sometimes when using the Animation mode to send Animation data to selected slaves. Animation is the only mode in which IDs are used.)

With the IDs sent and the animation data loaded in the PC, ensure switches S1 to S3 are switched off, then click the Send Animate button to send the animation to the selected PIC. On receipt of the data, that PIC then shows the animation in action on its display, as you saw in the PC's test routine. The PC screen confirms which slave it has been sent to via the display box above the button.

The same, or different, animation data may be sent to another PIC in the same way. It may also be sent to all slave PICs identically by first clicking the Send To All box to reveal a tick mark, and then click the Send Animate button. All slave PICs will receive the data and start to display it. The tick mark can be turned off by clicking its box again.

If data has been sent to all PICs identically, the small display box shows a value of 255.

Your own animation sequence can be created in a similar way to creating graphics, clicking on the PC's display boxes to create the frames sequence you want. Larger animation spanning two or more displays can be created by allocating different data to adjacent PICs, vertically or horizontally. You are always limited to the use of seven frames.

Non-moving symbols can also be created similarly, allocating identical data to each of the seven frames.

Data is saved to a file in the same manner as with moving messages, first creating a named blank file, loading it, and saving the data into that named file.

Synchronisation

If the animation data has been sent to all slaves. It will be seen that eventually they go out of synchronisation with each other, as their respective clock rates are not precisely matched, as explained earlier.

Synchronisation can be achieved by clicking the Sync box to reveal a tick prior to sending the data to the slaves. Sync can also be turned on or

off by clicking the Sync box to hide or reveal the tick, on an alternating cycle. Sync can be changed while running by clicking the Send button above the tick box.

The status of the sync flag can be checked by clicking Check Sync on the main screen. The display is binary 8 for sync off, and binary 9 for sync on.

The rate at which the animation takes place can be changed while the displays are running by amending its box value and then clicking the Send Rate button.

Static message

Click now on the Static Screen button at the bottom of the Animation screen to enter the Static Screen, shown in Fig.14.

Again, there is the selection of displays quantity and arrangements available, but this will have been set to the same format as used for animation when you entered the Static screen. Similarly, such change as you make here will be repeated if you go back to the Animation screen.

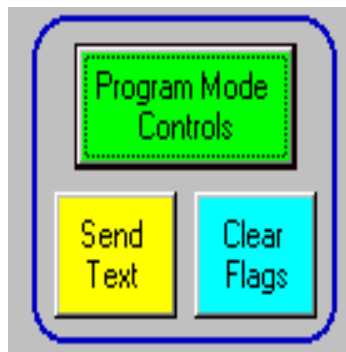


Fig.15. Program mode zone on static message screen

To create a static message, enter the message into the text box provided. Create a named file, load and save into it as before. Reloading that file, the message will be displayed back in the text box and in the blue boxes where it can be examined. The boxes are not numbered on screen as you can arrange the text as you want it and sent to as many displays as implied by its length.

To send the loaded message to the slaves, ensure the unit's switches S1 to S3 are switched off, and click the Program Mode Controls button (if it was not last clicked in a previous screen), to reveal just two buttons now (Fig.15). Click the Send Text button to send it, whereupon the slaves will display it.

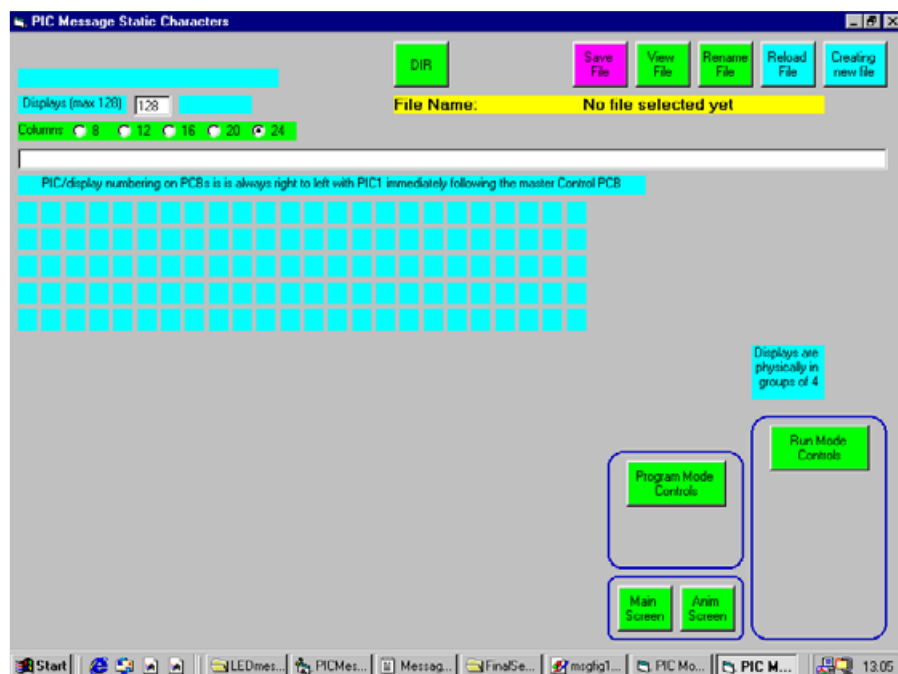


Fig.14. Static message screen, without additional data

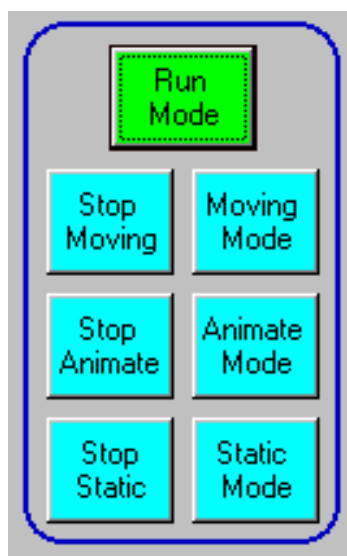


Fig.16. Run mode zone – common to all screens

Static points

A few points to note about static messages. First, they are normally shown in place of animation or moving data. The exception to this is in respect of blank spaces in a static message. The slaves do not regard the blanks as having data, consequently, animation can be sent to the slaves so that it occurs in those blank displays.

Secondly, the act of sending static data to the slaves overwrites the existing moving message in the master PIC's memory. However, that moving

message data can be re-sent to the master following the sending of the static data. This restriction is imposed by the master using the same program routines to send static data to the slaves as it does to store and send moving messages.

Mode selection

There is a choice of methods by which to select moving or static messages, or animation displays. Normally, with the PC unplugged from the unit, the selection is made via switches S1 to S3, each respectively turning its selected mode on or off, S1 for moving message, S2 for animation and S3 for static message.

The same selection can be made from the PC. Click the Run Mode button on any of the three mode screens. This reveals six buttons (Fig.16) while hiding the buttons in the Program Mode area. The function of the buttons is marked and is the same as that for the switches.

Note that there is no communication between the PC and master PIC in this context, and so the PC does not know the setting of any switch. All this means is that the PC's buttons may not respond as expected until the other associated button has been clicked. The PC's control then overrides the switches. The switches may still be used after the PC has selected the mode.

All three modes may be run concurrently, the appropriate display appearing on a particular LED module. For example, if animation data has been sent to some displays, but not others, those others will show the moving message, it 'passing behind' the animation displays unseen. Similarly, if a static message has been sent with spaces in it, those spaces will show the moving message as it scrolls along, again 'passing behind' the used static displays. A mixture of all three modes is possible.

When static and animation data are sent to the slaves, a 'marker flag' is internally set within the PIC's memory. When the static or animation buttons or switches are used, only those PICs whose data type is appropriate to the button or switch will show that data, or terminate the mode if already active.

The control codes are 0 for normal moving message only, 1 for animation and 2 for static. There is one only flag in each PIC and so only one mode code is ever active in it.

When static or animation data is sent to the slaves, the flag is always changed to the current code type if the destination PICs are intended to accept the latest data.

In connection with this, the flags may be totally cleared in all slaves by clicking the Clear Flags button in the Program Mode box of any PC display.

Message over and out

That then is the overall nature of this message display unit. Over to you now to enjoy making use of it and its facilities, let us know how you find it.

Finally, a warning. If you think about making changes to the software, beware that it is highly complex, probably the most complex project software on which the author has worked. So many factors are inter-linked, more so than may appear at first sight. You make changes entirely at your own risk. Make sure you have a backup copy before you start!

Note also that Joe Farr's Serial Interface software, available from our Down-

Resources

Software, including source code, is available for free download via the EPE website at www.epemag.co.uk. Pre-programmed PICs are available from Magenta Electronics Ltd, contact details as in their advert in this issue, prices on request. Ask about quantity discounts.

There are a few other example test data files that can be examined through the Directory mode of the various screens.

loads site, must be installed if you wish to run the PC program under VB6.

Hopefully, you will not wish to make changes and enjoy using the design at it is.

And just in case you're thinking of such – it is illegal in the UK to have an illuminated message in your car's rear window. So resist the temptation to tell that other driver to stop climbing up your exhaust pipe! *EPE*

PCB-POOL®
SERVICING YOUR COMPLETE PROTOTYPE NEEDS

Price Example: 16 Sq-Inches (double sided pth)

2 Days: €123 (incl vat)
8 Days: €31 (incl vat)

Standard PCB-Pool Service
SIMPLY SEND YOUR FILES AND ORDER ONLINE!

New Service: **WATCH "ur" PCB®**

Save vital time on design errors in advance of receiving your Prototype. View high resolution photographic images of your PCB during each production stage. Be one step ahead, use our realtime PCB monitoring service.

WWW.PCB-POOL.COM

Free Phone: 0800 389 8560
sales@beta-layout.com

Logos: TARGET, PROTEUS, Protel, Electronics WORKBENCH, orcad, Sprint Layout, Easy-PC, GraphiCad, R

Text on PCB: **INDUSTRY QUALITY LEAD FREE**, **ROHS / WEEE conform**

Text at top right: **DOWNLOAD OUR FREE PCB SOFTWARE**
www.free-pcb-software.com