Eprom programmer/copier

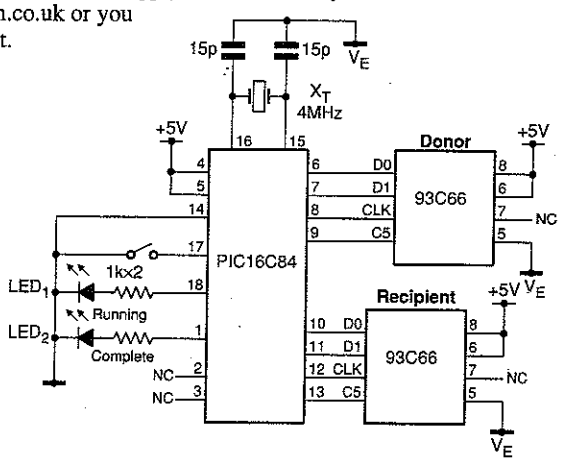
Having previously used a pc-based method of programming eeproms, I produced this device to free the pc for other work.

A donor microwire memory device is read by the circuit one byte at a time, the information then being programmed into a recipient, which is automatically erased when the process starts; all locations are written. Furthermore, instead of the manual input needed by the pc programmer, this only needs one four-second switch operation. You do, of course, have to make one donor using the pc or some other method; donors can be kept to guard against hiccups of the hard disk.

Pressing the switch starts the programming operation, which halts on completion; led 1 flashes while programming is in operation and led 2 glows on completion; opening the switch primes the *PIC16C84* for another run, but switch the power off when changing the recipient chip.

Opcode is shown here, but I can supply a 3.5in disk; my e-mail address is brian@briano.demon.co.uk or you can use ordinary post.

Brian Olliver
12, Fountain Drive
St Georges
Telford TF2 9DP



Instead of tying up a pc, use this programmer with donor and recipient devices to program eeproms.

```
:100000004A2800288610861106110000086158614DD
:1000100000000061500000000061100348612861349
:1000200006130000861786160000061700000000061
:1000300006130034A1406150000061C8A100611E7
:100040000348D01083091000310 8D0D1A200D1021
:100050008A180D14910B252800348A1E3128861623
:1000600032288612061700000613003412089100 89
:100070008A12901B8A162D20900D910B382800347F
:1000800012089100 0A10901B0A142D20900D910B5C
:1000900042280034013065008501113066008601 78
:1000A000051850288514E2002309200003090007 0
:1000B00036200830920 0C03090003620861300 00B1
:1000C000B52000308C0002200230920080309000 79
:1000D000402008309200 0C089000402021200D089C
:1000E000970021200D089800861183286E30960015
:1000F000000000000000000000000000960B7828BF
:10010000950B762800340E2002309200403090008B
:10011000362008309200 0C089000362017089000 16
:10012000362018089000362086130A309500762075
:10013000 8C0F6328A728851005156430950076205C
:10014000051164309500762005185028 9C280E2053
:10015000023092000030900036200830920000 30CB
:10016000900036208613000 9B280E20023092005B
:10017000003090003620083092008030900036200 9
:0C018000861300003C309500762008003B
:00000001FF
```

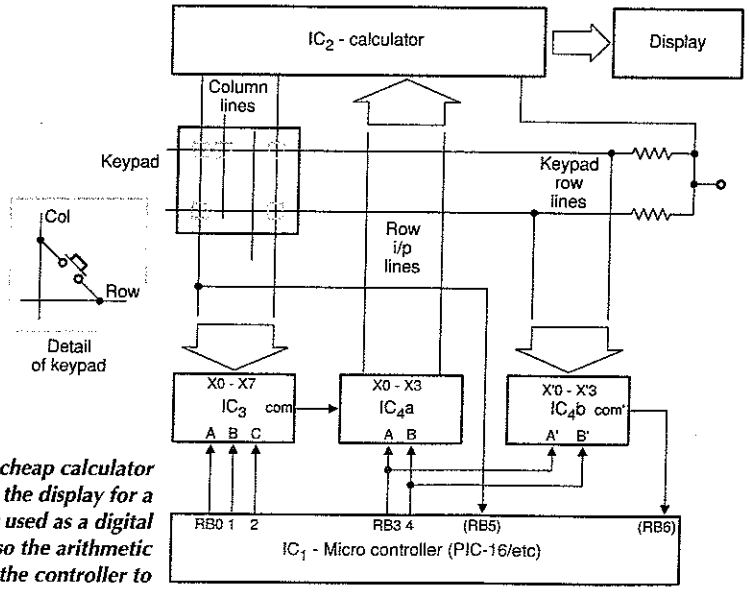
Calculator chip as maths coprocessor and display

Using a cheap microcontroller can make digital instruments cost-effective in small numbers, but the cost of displays and their demands on the processor pose a problem.

One solution to both problems is to use the ic and display from a cheap calculator as a combined coprocessor and display controller, all calculation and display functions then being carried out by this purpose-designed processor; the main microcontroller is then left to pursue its own affairs.

Calculator keypads are normally multiplexed and may be emulated from the microcontroller by monitoring column outputs and pulling the appropriate row input line high or low, depending on the calculator used.

The diagram shows an alternative: two cmos analogue multiplexers, $IC_{3,4}$, will simulate the keypad directly, simplifying the software interface and reducing the number of port lines used. Only five lines, $RB_{0\text{-}4}$, are needed for calculators using the 6 by 4 or 7 by 4 matrix. Since these lines are only used when writing to the coprocessor, they can be multiplexed for other purposes – possibly using the enable pin on IC_3 to disable display writes. An unused column address in IC_3 is otherwise used as a No-op command and each digit or operand entry is achieved by writing one five-byte to the display controller.



Innards of a cheap calculator provide not only the display for a microcontroller used as a digital instrument, but also the arithmetic functions to free the controller to

As you can see from the diagram, the calculator keypad can be kept as a free control panel by using the spare 4:1 multiplexer in IC_4 and the extra port lines $RB_{5,6}$. Selecting No-op on $RB_{0\text{-}2}$ and the row address on $RB_{3,4}$ allows RB_6 to be monitored for a pulse. If columns rise in sequence, the active key column can be determined from the time delay from the first column at RB_5, or a priority encoder could be used to indicate a keystroke directly.

A variety of arithmetic functions is possible with eight-digit precision, even from cheap calculators; not only the basic functions, but with a little ingenuity, accumulation, average, delta and many others.

You can measure the mains frequency accurately without long gate times by using a *PIC16X* in timer mode and the calculator to find the reciprocal. Functions not normally found in commercial counter-timers are percentage or frequency change; these can be performed easily using this method.

Anthony New *Bristol*