ACORN COMPUTER
Acorn Computers Limited, 4a Market Hill, Cambridge CB2 3NJ, England. Telephone 0223 312772. Telex 81383 Peatam G

ACORN TECHNICAL MANUAL

Prom Programmer Board.....200.007

Issue 2 Jan 1981

INTRODUCTION

The Acorn PROM Programmer Board connects to the standard Acorn Computer Bus and may be used to program bipolar PROMs of the 74LS571 type, or Ultra Violet Erasable PROMs types 2758, 2716, 2516 and 2532. These are the 5 volt only 1K, 2K and 4K byte devices.

Suplied with the board is a listing of a 1.5 K byte assembler program allowing device type selection and:-

Verification that a PROM is blank

Copying of a PROMs contents into RAM

Verification of a PROMs contents against RAM

Blowing of a PROM to match the contents of RAM

A cyclic redundancy check is performed on PROMs giving a unique signature identifying different PROM data. The program requires that an Acorn Operating System is present.

The interface is via an INS 8255 on the board which provides 24 lines for control of address, data and enable lines to the PROM, and automatic switching of the programming power rail to the PROM.

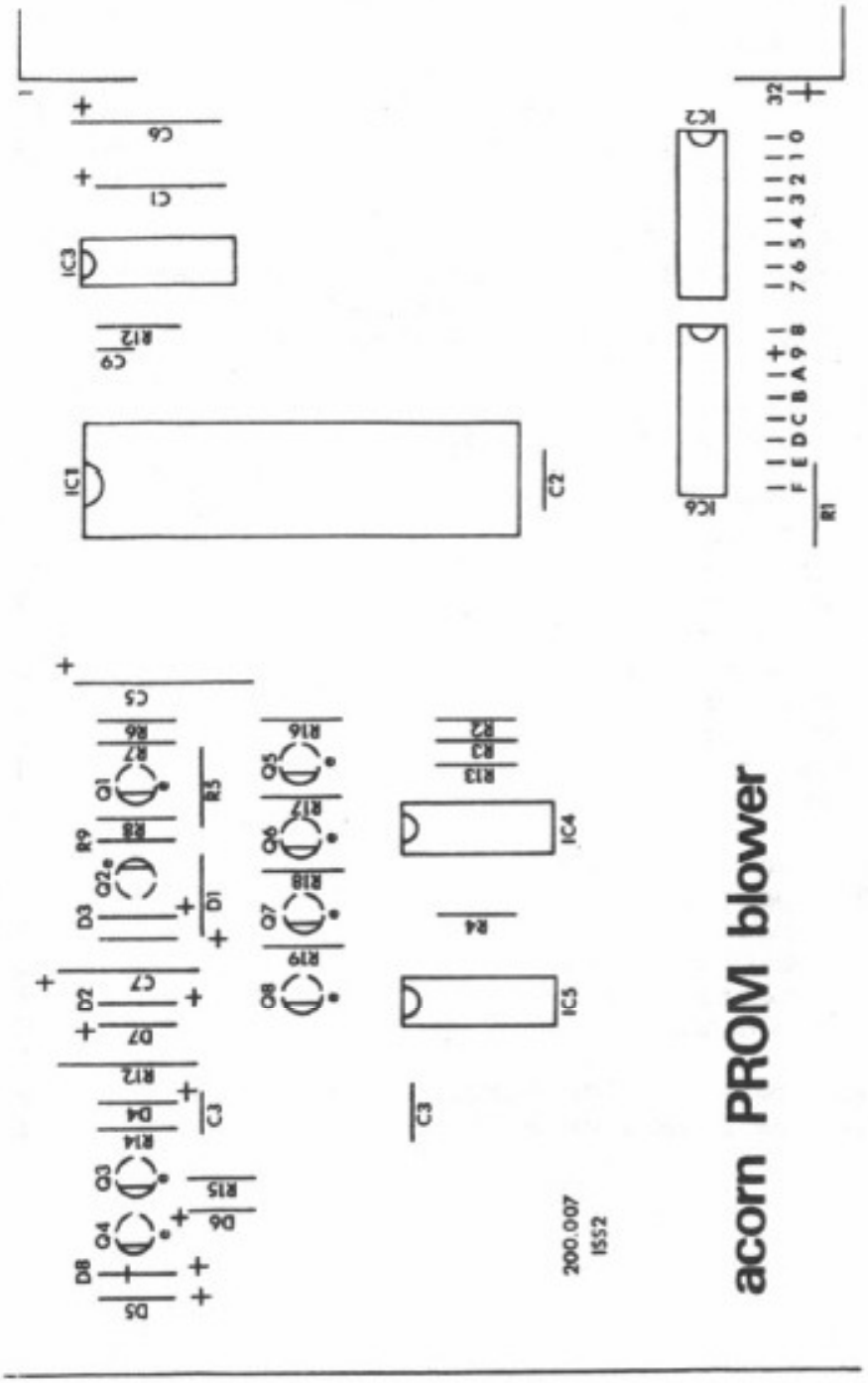
The programming board is longer than the usual Acorn Eurocard so that the two Low Insertion Force PROM sockets are accessible from the front of the system. If desired this extra piece of board can be removed and connections to sockets on a remote panel can be made.

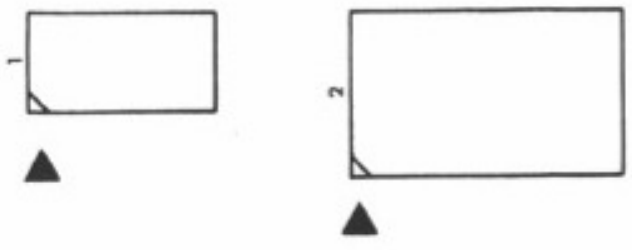
The board requires a +12 volt supply for programming of 74LS571 devices, and a +26 volt supply for programming EPROMs.

PARTS LIST FOR ACORN PROM PROGRAMMER

PCB	Acorn Computers Ltd. pt no 200.007		
IC1	INS 8255 P.I.A		and 40 pin socket
IC2	74LS138 decoder		and 16 pin socket
IC3	74LS00 Quad nand gate		and 14 pin socket
IC4	74C906 Hex CMOS inverter		and 14 pin socket
IC5	CD4016B Transmission gate		and 14 pin socket
IC6	74LS138 Decoder		and 16 pin socket
Q 1	BC237,BC239,BC107,etc.	N.P.N transistor	
Q 2	BC307,BC329,BCY70,etc.	P.N.P transistor	
Q 3	BC237,BC239,BC107,etc.	N.P.N transistor	
Q 4	2N3053	N.P.N transistor	
Q 5-8	BC237,BC239,BC107,etc.	N.P.N transistor	
D1-6	IN4002 diode		6 off
D7	BZX85 5.6V 1W zenner diode		1 off
R1-2	4K7 resistor		2 off
R3-5	10K resistor		3 off
R6	4K7 resistor		1 off
R7-8	10K resistor		2 off
R9	100K resistor		1 off
R12	47R resistor 1 watt		1 off
R13-14	10K resistor		2 off
R15	4K7 resistor		1 off
R16-19	4K7 resistor		4 off
R20-21	1K resistor		2 off
C1	10 uF capacitor 10V		1 off
C2-3	47 nF capacitor		2 off
C5	10 uF capacitor 30V		1 off
C6	22 uF capacitor 16V		1 off
C7	10 uF capacitor 10V		1 off
C8-9	330pF capacitor		2 off
SK1	16 Pin Low Insertion Force Socket		1 off
SK2	24 Pin Low Insertion Force Socket		1 off

CIRCUIT BOARD LAYOUT

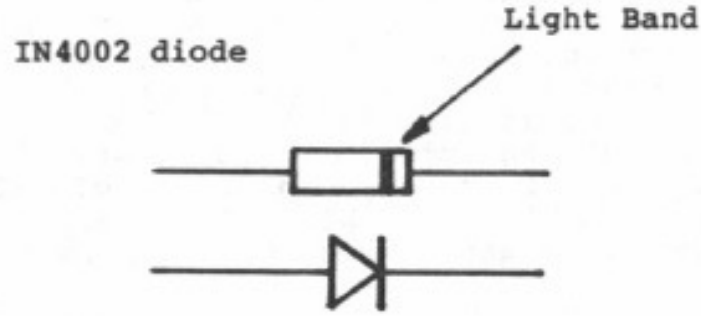




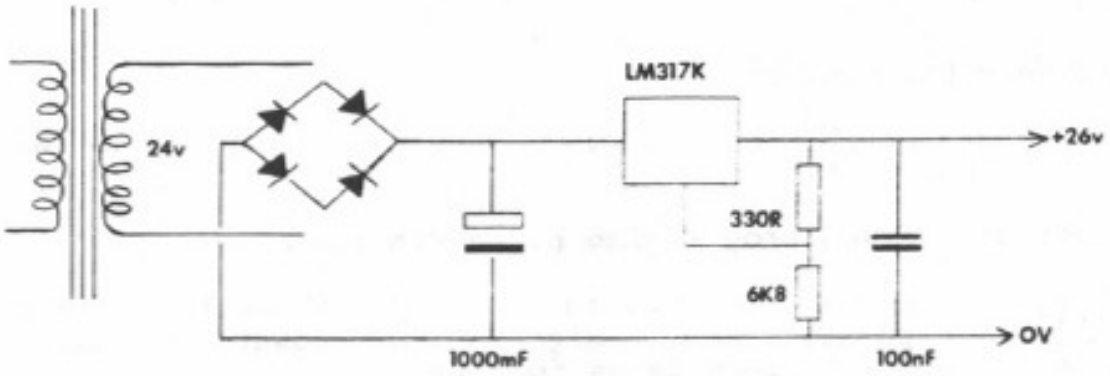
CONSTRUCTION NOTES

Sockets are supplied for all IC's and should be used. The system technical manual contains notes on soldering technique and component identification. The transistors and diodes supplied are shown below:-

transistors viewed from below
ie. lead side.



A +12 volt power supply is available in systems with disk drives. To provide the +26 volt supply the circuit shown below can be constructed.-



CIRCUIT DESCRIPTION

To use this board as supplied with the Acorn system it should be connected to the Acorn bus for all pins on side A, and extra power should be supplied to the board on side B. To blow fuseable link PROMs of the 74LS571 type a supply of +12 volts at 300mA is required on pin B1 of the edge connector. To blow 5V EPROMs a supply of +26 volts at 100 mA is required on pin B19 of the edge connector. When programming EPROMs without the 12V supply IC5 should be removed.

As supplied the PROM programmer is memory mapped into page 19 (hex), by using the links associated with IC's 2 and 6 this may changed. When changing the location of the board in memory reference should be made to the memory map of the system (see system manual) to ensure that no conflict occurs. The position of the board should not be changed if the programmer is to be driven by the PROMER software without changing this software. The links allow the PROM programmer to be mapped into any of the upper 8 pages in any of the lower 8 blocks in the memory map (see figure 1). The ports of 8255 are then enabled between locations 80 and CF of this page. These ports are accesed by the PROM programmer software at the following locations:-

Port A	1980
Port B	1981
Port C	1982
Control register	1983

The ports are used to control the lines to the PROMs as follows

PORT A Data bus control.

The lines of port A are used to read data from or write data to a PROM. When programming 74LS571s only the lower four bits of this port are used.

PORT B Address bus control

The lines of port B are used to control the address lines A0 to A7 going to the PROM.

PORT C Address bus and programming pulse control

The lines of port C are used to control address lines A8 to A11, the programming pulses, enable signals, and power supplies to the PROM. The allocation of lines is as follows:-

Bit 0,1 Address lines A8 and A9

Bit 2 AR signal on 2758 (held high), A10 on all other EPROMS.

Bit 3 A11 on 2532. Program pulse on all other EPROMS pulsed high to program location specified by address lines with data from data lines. Chip enable/program pulse on 74LS571.

Bit 4 Program pulse on 2532 pulsed low to program. Chip select or output enable on other EPROMs.

Bit 5 Read/write control for 74LS571. This signal controls the CD4016 bilateral switch to isolate the port A inputs from the 12V programming signals on the data lines when programming 74LS571s. This signal is low to isolate the port, and high to read the chip as normal.

Bit 6 Power control for 74LS571 programming. When this signal is low the 74LS571 receives 5V on its power rail, via D5 from a 5.6V source regulated by D7 and R12 from the 12V source. When this signal is high the chip to be programmed receives a supply of 10.2V switched by Q3 and Q4 from the 12V supply, and 10.2V can be switched to a data line by Q5 to Q8 as required for programming.

Bit 7 Power control for EPROM programming. When this signal is low the EPROM receives 5V on pin 21 via D2 from a 5.6V source regulated by D7 and R20 from the 26V source. When the signal is high pin 21 of the EPROM is held at 25V switched by Q2 and Q1 from the 26V source, as required for programming.

SOFTWARE DESCRIPTION

The software to drive the PROM blower is listed here and may be typed in or suplied on ROM, cassette tape or disk, it occupies memory between addresses #2800 and #2D00, and should be entered at #2840. If you have the PROMER in ROM or RAM you should enter it be typing

```
GO 2840
```

using DOS or COS you can type

```
RUN "PROMER"
```

The PROMER program wil return the prompt

```
Which PROM ?
```

to which you should reply with the identification number of the type of PROM you wish to blow, the types recognised are as follows:-

2532	4K uvEPROM
2516	2K uvEPROM
2716	2K uvEPROM
2758	1K uvEPROM
571L	74LS571 1/2K, 4 bit fusable link PROM (This will blow the lower half of the bytes in memory into the PROM)
571H	74LS571 1/2K, 4 bit fusable link PROM (This will blow the upper half of the bytes in memory into the PROM)

Alternatively the PROMER will at this point accept any Operating System comand preceeded by a star '*'. This allows files to be save or loaded under control of the DOS or COS. The reply to the prompt is terminated by a cariage return, any response that is not understood will cause the prompt

```
Which PROM ?
```

to appear.

When the type of PROM to be blown has been identified the prompt

```
BLOW : READ : VERIFY : CLEAR
Enter initial letter ?
```

will appear, and the option may be selected terminated by a carriage return.

In response to the comands BLOW, READ and VERIFY the prompt

```
From ?
```

will appear to which the reply should be a hexadecimal number which is the base address of the area of memory to be used.

The options then operate in the following manner.

CLEAR

This option is used to confirm that a PROM is blank, a report of blank or not blank will be output for each 1K of PROM.

READ

This option is used to read the contents of a PROM into memory. The contents are read into memory starting at the address specified in response to the

From ?

prompt. A Cyclic Redundancy Check sum is formed from the data to provide a unique signature for EPROMs only and this is printed out.

VERIFY

This option is used to verify the contents of a ROM against an area of RAM or ROM in memory starting at the specified base address. A report of FAIL or OK is output for each 1K of the PROM, the CRC for EPROMs only is also created and printed out.

BLOW

This option is used to blow the contents of a section of memory in the system (ROM or RAM) into the specified PROM starting at the specified base address. This is done in three steps, the CLEAR option is automaticaly called to output a report on the state of the PROM. When this has been done the prompt

Blow ?

will appear to which the responce should be Y to blow the PROM, any other responce terminates the BLOW option. The reply

Blowing

will be output and the PROM will be blown an idication of how much of a PROM has been programed is provided as a hexadecimal digit is output for each 1/4K that has been blown. When the blowing is complete the verify option is called to confirm that the PROM has been correctly blown and print out the CRC for EPROMs only.

When an option has been completed the prompt

Repeat ?

will appear a responce of Y will repeat the previous action, any other reply will cause a return to the prompt

Which PROM ?

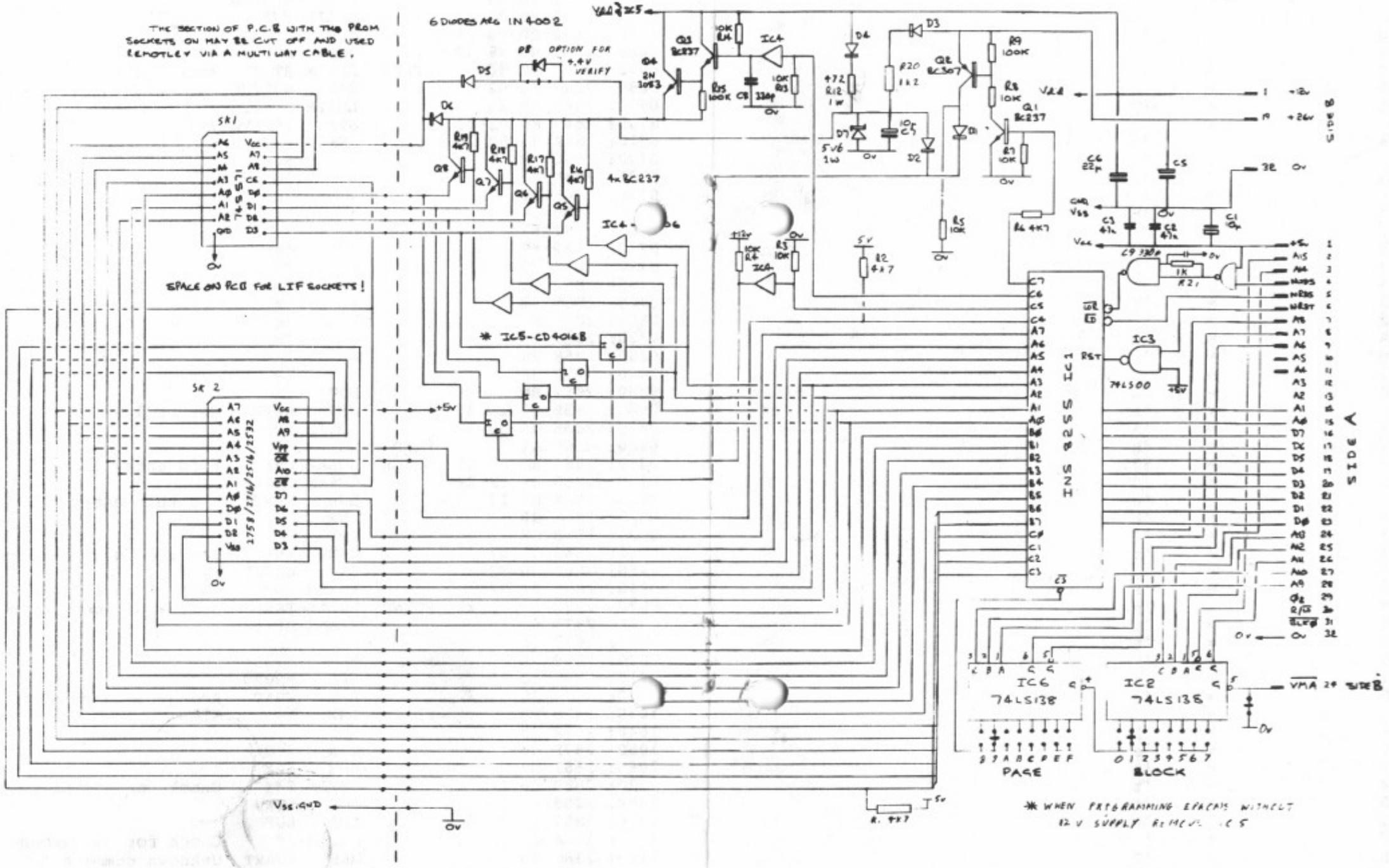
Removing and inserting PROMS

Removing PROMS when the programming voltage is applied may cause damage and they should not be changed while an option is running. It is safe to change PROMS when the program is waiting for any reply.

PROGRAM LISTING - part 1

```
0010: 2800                PROMER ORG   $2800
0020: 2800                BFLIP  *     $0080   Zero page locations used
0030: 2800                CK     *     BFLIP  +01 Location #80
0040: 2800                HL     *     CK
0050: 2800                VE     *     CK     +01
0060: 2800                BL     *     VE     +01
0070: 2800                HOWVEC *     BL     +01
0080: 2800                TOBLOW *     HOWVEC +02
0090: 2800                TEMP   *     TOBLOW +02
0100: 2800                REGMAP *     TEMP   +02
0110: 2800                JOB    *     REGMAP +01
0120: 2800                FROM   *     JOB    +01
0130: 2800                POINT  *     FROM   +02
0140: 2800                MASK   *     POINT
0150: 2800                TRIES  *     MASK   +01
0160: 2800                EXTRA  *     TRIES  +01
0170: 2800                COUNT  *     POINT  +02
0180: 2800                CURENT *     COUNT  +01
0190: 2800                NUMBER *     CURENT +01
0200: 2800                CRC    *     NUMBER +01 To location #94
0210: 2800                BUFF   *     $0100  Input buffer
0220: 2800                KB     *     $0E21
0230: 2800                BLOWER *     $1980  Location of prom blower
0240: 2800                PORTA  *     BLOWER    data
0250: 2800                PORTB  *     PORTA  +01   address low
0260: 2800                PORTC  *     PORTA  +02   address high/control
0270: 2800                CWR    *     PORTA  +03 Control word register
0280: 2800                ESC    *     $1B    Escape character
0290: 2800                BRKVEC *     $0202  Used to trap errors
0300: 2800                OSWRCH *     $FFF4  OS calls
0310: 2800                OSASCI *     $FFE9
0320: 2800                OSRDCH *     $FFE3
0330: 2800                OSCRLF *     $FFED
0340: 2800                OSECHO *     $FFE6
0350: 2800                OSCLI  *     $FFF7
0360: 2800 68             VSTRNG PLA          Display string
0370: 2801 85 88                 STA   TEMP
0380: 2803 68                    PLA
0390: 2804 85 89                 STA   TEMP   +01
0400: 2806 A0 00          VSTRLP LDYIM $00
0410: 2808 E6 88                 INC   TEMP
0420: 280A D0 02                 BNE   INCTPC
0430: 280C E6 89                 INC   TEMP   +01
0440: 280E B1 88          INCTPC LDAIY TEMP
0450: 2810 C9 EA                 CMPIM $EA
0460: 2812 F0 06                 BEQ   VSTRNX
0470: 2814 20 E9 FF              JSR   OSASCI
0480: 2817 4C 06 28              JMP   VSTRLP
0490: 281A 6C 88 00       VSTRNX JMI   TEMP
0500: 281D A2 00          BUFFIN LDXIM $00    Input buffer routine
0510: 281F 20 E6 FF       ANSWER JSR   OSECHO
0520: 2822 9D 00 01              STAAX BUFF
0530: 2825 C9 7F                 CMPIM $7F    Check for delete
0540: 2827 D0 05                 BNE   ENTERD
0550: 2829 CA             BUFFA  DEX
0560: 282A 30 F1                 BMI   BUFFIN
0570: 282C 10 F1                 BPL   ANSWER
```

```
0580: 282E C9 0D     ENTERD CMPIM $0D    Terminated by CR
0590: 2830 D0 01            BNE   NOTDUN
0600: 2832 60               RTS
0610: 2833 E8        NOTDUN INX
0620: 2834 E0 28            CPXIM $28    Limit buffer length
0630: 2836 D0 E7            BNE   ANSWER
0640: 2838 A9 7F            LDAIM $7F
0650: 283A 20 F4 FF         JSR   OSWRCH
0660: 283D 4C 29 28         JMP   BUFFA
0670: 2840 A9 40     START  LDAIM START  Entry point
0680: 2842 8D 02 02         STA   BRKVEC Initialise for errors
0690: 2845 A9 28            LDAIM START  /
0700: 2847 8D 03 02         STA   BRKVEC +01
0710: 284A 20 00 28         JSR   VSTRNG
0720: 284D 0D               =     $0D
0730: 284E 57               =     'W
0740: 284F 68               =     'h
0750: 2850 69               =     'i
0760: 2851 63               =     'c
0770: 2852 68               =     'h
0780: 2853 20               =     '
0790: 2854 70               =     'p
0800: 2855 72               =     'r
0810: 2856 6F               =     'o
0820: 2857 6D               =     'm
0830: 2858 20               =     '
0840: 2859 3F               =     '?
0850: 285A EA               NOP
0860: 285B 20 1D 28         JSR   BUFFIN Get reply
0870: 285E A0 00            LDYIM $00
0880: 2860 A2 00     FIND   LDXIM $00
0890: 2862 BD 00 01  WHICH  LDAAX BUFF   Find entry in table
0900: 2865 D9 19 29         CMPAY TABLE
0910: 2868 D0 13            BNE   NEXT   Not this entry
0920: 286A E8               INX
0930: 286B C8               INY
0940: 286C E0 04            CPXIM $04    For all four characters
0950: 286E D0 F2            BNE   WHICH
0960: 2870 A2 07            LDXIM $07    Good we have found it
0970: 2872 B9 19 29  FOUND  LDAAY TABLE  So put parameters in pg zro
0980: 2875 95 80            STAAX BFLIP
0990: 2877 C8               INY
1000: 2878 CA               DEX
1010: 2879 10 F7            BPL   FOUND
1020: 287B 30 1C            BMI   WHAT   And go to next step
1030: 287D E8        NEXT   INX          Step to next entry
1040: 287E C8               INY
1050: 287F E0 0C            CPXIM $0C
1060: 2881 D0 FA            BNE   NEXT
1070: 2883 C0 44            CPYIM $44    Check for end of table
1080: 2885 90 D9            BCC   FIND
1090: 2887 AD 00 01         LDA   BUFF
1100: 288A C9 2A            CMPIM '*     Check for OS command
1110: 288C D0 B2            BNE   START  Unknown command
1120: 288E A9 20            LDAIM '
1130: 2890 8D 00 01         STA   BUFF
```



ISSUE	B	1	DRN C&T	TITLE PROM & EPROM PROGRAMMER.	REF 200.007/C
DATE	24-07-79	11-06-80	DATE 24-07-79		ACORN COMPUTERS LTD

```
1140: 2893 20 F7 FF          JSR   OSCLI  And hand to  OS
1150: 2896 4C 40 28          JMP   START
1160: 2899 20 00 28  WHAT    JSR   VSTRNG
1170: 289C 42                =     'B
1180: 289D 4C                =     'L
1190: 289E 4F                =     'O
1200: 289F 57                =     'W
1210: 28A0 20                =     '
1220: 28A1 3A                =     ':
1230: 28A2 20                =     '
1240: 28A3 52                =     'R
1250: 28A4 45                =     'E
1260: 28A5 41                =     'A
1270: 28A6 44                =     'D
1280: 28A7 20                =     '
1290: 28A8 3A                =     ':
1300: 28A9 20                =     '
1310: 28AA 56                =     'V
1320: 28AB 45                =     'E
1330: 28AC 52                =     'R
1340: 28AD 49                =     'I
1350: 28AE 46                =     'F
1360: 28AF 59                =     'Y
1370: 28B0 20                =     '
1380: 28B1 3A                =     ':
1390: 28B2 20                =     '
1400: 28B3 43                =     'C
1410: 28B4 4C                =     'L
1420: 28B5 45                =     'E
1430: 28B6 41                =     'A
1440: 28B7 52                =     'R
1450: 28B8 0D                =     $0D
1460: 28B9 45                =     'E
1470: 28BA 6E                =     'n
1480: 28BB 74                =     't
1490: 28BC 65                =     'e
1500: 28BD 72                =     'r
1510: 28BE 20                =     '
1520: 28BF 69                =     'i
1530: 28C0 6E                =     'n
1540: 28C1 69                =     'i
1550: 28C2 74                =     't
1560: 28C3 69                =     'i
1570: 28C4 61                =     'a
1580: 28C5 6C                =     'l
1590: 28C6 20                =     '
1600: 28C7 6C                =     'l
1610: 28C8 65                =     'e
1620: 28C9 74                =     't
1630: 28CA 74                =     't
1640: 28CB 65                =     'e
1650: 28CC 72                =     'r
1660: 28CD 20                =     '
1670: 28CE 3F                =     '?
1680: 28CF EA                NOP
1690: 28D0 20 1D 28          JSR   BUFFIN Get reply
```

```
1700: 28D3 AD 00 01          LDA    BUFF
1710: 28D6 85 8B             STA    JOB
1720: 28D8 C9 43             CMPIM  'C
1730: 28DA F0 3A             BEQ    NONUM
1740: 28DC 20 00 28  WHERE   JSR    VSTRNG Get number if needed
1750: 28DF 46                =      'F
1760: 28E0 72                =      'r
1770: 28E1 6F                =      'o
1780: 28E2 6D                =      'm
1790: 28E3 20                =      '
1800: 28E4 3F                =      '?
1810: 28E5 EA                NOP
1820: 28E6 20 1D 28          JSR    BUFFIN Get reply
1830: 28E9 A2 00             LDXIM  $00    And translate ASCII to hex
1840: 28EB 86 8C             STX    FROM
1850: 28ED 86 8D              STX    FROM   +01
1860: 28EF CA                DEX
1870: 28F0 E8        NUMLOP  INX
1880: 28F1 BD 00 01          LDAAX  BUFF
1890: 28F4 C9 30             CMPIM  '0
1900: 28F6 90 1E             BCC    NONUM
1910: 28F8 C9 3A             CMPIM  $3A
1920: 28FA 90 08             BCC    NUMCON
1930: 28FC E9 07             SBCIM  $07
1940: 28FE 90 16             BCC    NONUM
1950: 2900 C9 40             CMPIM  $40
1960: 2902 B0 12             BCS    NONUM
1970: 2904 29 0F     NUMCON  ANDIM  $0F
1980: 2906 0A                ASLA
1990: 2907 0A                ASLA
2000: 2908 0A                ASLA
2010: 2909 0A                ASLA
2020: 290A A0 04             LDYIM  $04
2030: 290C 0A        NUMCIN  ASLA
2040: 290D 26 8C             ROL    FROM
2050: 290F 26 8D             ROL    FROM   +01
2060: 2911 88                DEY
2070: 2912 D0 F8             BNE    NUMCIN
2080: 2914 F0 DA             BEQ    NUMLOP
2090: 2916 6C 86 00  NONUM   JMI    TOBLOW
2100: 2919 35        TABLE   =      '5     Table of PROM types
2110: 291A 37                =      '7
2120: 291B 31                =      '1
2130: 291C 48                =      'H
2140: 291D 2A                =      FUSE   What to blow
2150: 291E A9                =      FUSE   /
2160: 291F 00                =      $00    Blank entries for 74LS571
2170: 2920 00                =      $00
2180: 2921 00                =      $00
2190: 2922 00                =      $00
2200: 2923 00                =      $00
2210: 2924 00                =      $00
2220: 2925 35                =      '5
2230: 2926 37                =      '7
2240: 2927 31                =      '1
2250: 2928 4C                =      'L
```

```
2260: 2929 2A            =    FUSE
2270: 292A A9            =    FUSE   /
2280: 292B 00            =    $00
2290: 292C 00            =    $00
2300: 292D 00            =    $00
2310: 292E 00            =    $00
2320: 292F FF            =    $FF
2330: 2930 00            =    $00
2340: 2931 32            =    '2
2350: 2932 37            =    '7
2360: 2933 31            =    '1
2370: 2934 36            =    '6
2380: 2935 29            =    EPROM  What to blow
2390: 2936 61            =    EPROM  /
2400: 2937 29            =    DOTWO  How to blow it-no of K bytes
2410: 2938 C7            =    DOTWO  /
2420: 2939 90            =    $90    Control word for BLOW
2430: 293A 80            =    $80    Control word for VERIFY
2440: 293B 00            =    $00    Control word for READ
2450: 293C 07            =    $07    Control word for prog pulse
2460: 293D 32            =    '2
2470: 293E 37            =    '7
2480: 293F 35            =    '5
2490: 2940 38            =    '8
2500: 2941 29            =    EPROM
2510: 2942 61            =    EPROM  /
2520: 2943 29            =    CORE
2530: 2944 CC            =    CORE   /
2540: 2945 90            =    $90
2550: 2946 80            =    $80
2560: 2947 00            =    $00
2570: 2948 07            =    $07
2580: 2949 32            =    '2
2590: 294A 35            =    '5
2600: 294B 33            =    '3
2601: 294C 32            =    '2
2602: 294D 29            =    EPROM
2603: 294E 61            =    EPROM  /
2604: 294F 29            =    DOFOUR
2605: 2950 C2            =    DOFOUR /
2606: 2951 90            =    $90
2607: 2952 00            =    $00
2608: 2953 00            =    $00
2690: 2954 08            =    $08
2700: 2955 32            =    '2
2710: 2956 35            =    '5
2720: 2957 31            =    '1
2730: 2958 36            =    '6
2740: 2959 29            =    EPROM
2750: 295A 61            =    EPROM  /
2760: 295B 29            =    DOTWO
2770: 295C C7            =    DOTWO  /
2780: 295D 90            =    $90
2790: 295E 80            =    $80
2800: 295F 00            =    $00
2810: 2960 07            =    $07
```

PROGRAM LISTING-part 2

```
0010:                        **************
0020:                        *   EPROM    *
0030:                        **************
0040: 2961 A5 8B     EPROM   LDA   JOB
0050: 2963 C9 42             CMPIM 'B
0060: 2965 D0 20             BNE   ANOTBL
0070: 2967 20 A5 29          JSR   ACHECK Blow prom so check for blank
0080: 296A 20 FE 2B          JSR   BLOWQR And Query for continue
0090:                  ****  BLOW PROM  ****
0100: 296D A5 83             LDA   BL     Load control word
0110: 296F A0 00             LDYIM $00    And control flags
0120: 2971 20 AF 29          JSR   STARTR Initialise pointers
0130: 2974 20 AC 29          JSR   HOW    And do it
0140:                   Now verify blowing
0150:                      ****  VERIFY PROM  ****
0160:                      as for check prom
0170: 2977 A5 82     ANOTC   LDA   VE     Load control word
0180: 2979 A0 80             LDYIM $80    And cotrol flags
0190: 297B 20 AF 29          JSR   STARTR Initialise pointers
0200: 297E 20 AC 29          JSR   HOW    And do it
0210: 2981 20 42 2C          JSR   CRCOUT Print out CRC check
0220: 2984 4C 1D 2C          JMP   REPEAT And exit
0230:                      ****  READ PROM  ****
0240:                      as for check prom
0250: 2987 C9 52     ANOTBL  CMPIM 'R
0260: 2989 D0 10             BNE   ANOTRD
0270: 298B A5 81             LDA   CK     Load control word
0280: 298D A0 40             LDYIM $40    And control flags
0290: 298F 20 AF 29          JSR   STARTR Initialise pointers
0300: 2992 20 AC 29          JSR   HOW    And do it
0310: 2995 20 42 2C          JSR   CRCOUT Print out CRC check
0320: 2998 4C 1D 2C          JMP   REPEAT And exit
0330:                    ****  CHECK FOR BLANK  ****
0340: 299B C9 43     ANOTRD  CMPIM 'C
0350: 299D D0 D8             BNE   ANOTC
0360: 299F 20 A5 29          JSR   ACHECK Do check for blank
0370: 29A2 4C 1D 2C          JMP   REPEAT And exit
0380: 29A5 A5 81     ACHECK  LDA   CK     Load control word
0390: 29A7 A0 C0             LDYIM $C0    And control flags
0400: 29A9 20 AF 29          JSR   STARTR Initialise pointers
0410: 29AC 6C 84 00  HOW     JMI   HOWVEC Jump depends on size of PROM
0420: 29AF A6 8C     STARTR  LDX   FROM   This initialises pointers
0430: 29B1 86 8E             STX   POINT
0440: 29B3 A6 8D             LDX   FROM   +01
0450: 29B5 86 8F             STX   POINT  +01
0460: 29B7 84 91             STY   CURENT
0470: 29B9 A2 00             LDXIM $00
0480: 29BB 86 92             STX   NUMBER
0490: 29BD 86 93             STX   CRC
0500: 29BF 86 94             STX   CRC    +01
0510: 29C1 60                RTS
0520: 29C2 20 C7 29  DOFOUR  JSR   DOTWO  4K bytes to blow
0530: 29C5 49 0C             EORIM $0C
0540: 29C7 20 CC 29  DOTWO   JSR   CORE   2K bytes to blow
0550: 29CA 09 04     DOONE   ORAIM $04
0560: 29CC 85 8A     CORE    STA   REGMAP
```

```
0570: 29CE 48              PHA
0580: 29CF A9 80           LDAIM $80
0590: 29D1 A4 91           LDY   CURENT Now check control flags
0600: 29D3 F0 02           BEQ   WCWR     blow so all output
0610: 29D5 09 10           ORAIM $10      set data to input
0620: 29D7 8D 83 19 WCWR   STA   CWR
0630: 29DA A9 00           LDAIM $00
0640: 29DC 85 90           STA   COUNT
0650: 29DE 48              PHA            Clear error flag
0660: 29DF A8              TAY
0670: 29E0 8C 81 19 CORELP STY   PORTB
0680: 29E3 AD 21 0E        LDA   KB       Check for esc key
0690: 29E6 C9 1B           CMPIM ESC
0700: 29E8 D0 03           BNE   NOESC
0710: 29EA 4C 1D 2C        JMP   REPEAT
0720: 29ED A5 90     NOESC LDA   COUNT  Set upper address lines
0730: 29EF 05 8A           ORA   REGMAP
0740: 29F1 8D 82 19        STA   PORTC
0750: 29F4 A9 FF           LDAIM $FF
0760: 29F6 24 91           BIT   CURENT Check what to do
0770: 29F8 D0 2C           BNE   READ
0780: 29FA B1 8E     BLOW  LDAIY POINT  Blowing so load data
0790: 29FC 8D 80 19        STA   PORTA  Send to port
0800: 29FF 20 95 2A        JSR   FLIPBT And apply program pulse
0810: 2A02 C8        GOON  INY          Next location
0820: 2A03 D0 DB           BNE   CORELP
0830: 2A05 E6 8F           INC   POINT  +01 Adjust pointers
0840: 2A07 E6 90           INC   COUNT
0850: 2A09 A5 91           LDA   CURENT
0860: 2A0B D0 07           BNE   NOTELL
0870: 2A0D A5 92           LDA   NUMBER Print out number if blowing
0880: 2A0F 20 5C 2C        JSR   DIGIT
0890: 2A12 E6 92           INC   NUMBER
0900: 2A14 A5 90     NOTELL LDA   COUNT
0910: 2A16 C9 04           CMPIM $04
0920: 2A18 D0 C6           BNE   CORELP Do 4 quarter K bytes
0930: 2A1A 24 91           BIT   CURENT
0940: 2A1C 30 47           BMI   NOTBLW If blowing no testing is done
0950: 2A1E A9 90           LDAIM $90
0960: 2A20 8D 83 19        STA   CWR     switch things off
0970: 2A23 68              PLA          Ignore flags
0980: 2A24 68              PLA          Reload regmap
0990: 2A25 60              RTS
1000: 2A26 30 22     READ  BMI   VERIFY Still checking what to do
1010: 2A28 AD 80 19        LDA   PORTA  Reading so get data
1020: 2A2B 91 8E           STAIY POINT  Store it
1030: 2A2D 20 33 2A        JSR   PUTCRC Increment CRC
1040: 2A30 4C 02 2A        JMP   GOON   And continue
1050: 2A33 A2 08     PUTCRC LDXIM $08    This calculates the CRC
1060: 2A35 48              PHA
1070: 2A36 4A        CRCLP LSRA
1080: 2A37 26 93           ROL   CRC
1090: 2A39 26 94           ROL   CRC    +01
1100: 2A3B 90 08           BCC   NOC
1110: 2A3D 48              PHA
1120: 2A3E A5 93           LDA   CRC
```

```
1130: 2A40 49 2D          EORIM $2D
1140: 2A42 85 93          STA   CRC
1150: 2A44 68             PLA
1160: 2A45 CA       NOC   DEX
1170: 2A46 D0 EE          BNE   CRCLP
1180: 2A48 68             PLA
1190: 2A49 60             RTS   End    of CRC calculation
1200: 2A4A 70 10    VERIFY BVS  CHECK  Check what we are doing
1210: 2A4C AD 80 19       LDA   PORTA  Verify so get data
1220: 2A4F 20 33 2A       JSR   PUTCRC Update CRC
1230: 2A52 D1 8E          CMPIY POINT  And verify data
1240: 2A54 F0 AC          BEQ   GOON   Continue if OK
1250: 2A56 68       FLAG  PLA          Otherwise flag error
1260: 2A57 09 FF          ORAIM $FF
1270: 2A59 48             PHA
1280: 2A5A D0 A6          BNE   GOON   And then continue
1290: 2A5C AD 80 19 CHECK LDA   PORTA  Load data
1300: 2A5F C9 FF          CMPIM $FF    Check for blank location
1310: 2A61 F0 9F          BEQ   GOON   Continue if blank
1320: 2A63 D0 F1          BNE   FLAG   Otherwise flag not blank
1330: 2A65 70 17    NOTBLW BVS  NOTVER Now check, see what to print
1340: 2A67 68             PLA          Verifying so get flag
1350: 2A68 D0 09    FAILQ BNE   NOGOOD And print OK/FAIL
1360: 2A6A 20 00 28       JSR   VSTRNG
1370: 2A6D 0D             =     $0D
1380: 2A6E 4F             =     'O
1390: 2A6F 4B             =     'K
1400: 2A70 EA             NOP
1410: 2A71 68             PLA          Reload regmap
1420: 2A72 60             RTS
1430: 2A73 20 00 28 NOGOOD JSR  VSTRNG
1440: 2A76 0D             =     $0D
1450: 2A77 46             =     'F
1460: 2A78 41             =     'A
1470: 2A79 49             =     'I
1480: 2A7A 4C             =     'L
1490: 2A7B EA             NOP
1500: 2A7C 68             PLA          Reload regmap
1510: 2A7D 60             RTS
1520: 2A7E 68       NOTVER PLA         Check for blank so get flag
1530: 2A7F F0 08    FCHKX BEQ   BLANK  And print Blank/Not blank
1540: 2A81 20 00 28       JSR   VSTRNG
1550: 2A84 4E             =     'N
1560: 2A85 6F             =     'o
1570: 2A86 74             =     't
1580: 2A87 20             =     '
1590: 2A88 EA             NOP
1600: 2A89 20 00 28 BLANK JSR   VSTRNG
1610: 2A8C 42             =     'B
1620: 2A8D 6C             =     'l
1630: 2A8E 61             =     'a
1640: 2A8F 6E             =     'n
1650: 2A90 6B             =     'k
1660: 2A91 0D             =     $0D
1670: 2A92 EA             NOP
1680: 2A93 68             PLA          Reload regmap
```

```
1690: 2A94 60              RTS
1700: 2A95 A5 80     FLIPBT LDA   BFLIP  Programming pulse
1710: 2A97 8D 83 19         STA   CWR    Flip bit defined in BFLIP
1720: 2A9A A2 19            LDXIM $19    Now wait 50mS
1730: 2A9C C6 88     DLYLP  DEC   TEMP
1740: 2A9E D0 FC            BNE   DLYLP
1750: 2AA0 CA               DEX
1760: 2AA1 D0 F9            BNE   DLYLP
1770: 2AA3 49 01            EORIM $01
1780: 2AA5 8D 83 19         STA   CWR    return bit to previous level
1790: 2AA8 60               RTS
1800:                  ************
1810:                  *   FUSE   *
1820:                  ************
1830: 2AA9 A5 8B     FUSE   LDA   JOB
1840: 2AAB C9 42            CMPIM 'B
1850: 2AAD D0 0F            BNE   FNOTBL
1860:                   **** BLOW 74LS571  ****
1870: 2AAF 20 D2 2A         JSR   FCHECK Check for blank
1880: 2AB2 20 FE 2B         JSR   BLOWQR Query for continue
1890: 2AB5 20 FC 2A         JSR   FPROG  Blow PROM, fall into verify
1900:                     **** Verify 74LS571 ****
1910: 2AB8 20 83 2B  FNOTC  JSR   FVERIF Verify PROM
1920: 2ABB 4C 1D 2C         JMP   REPEAT And exit
1930: 2ABE C9 52     FNOTBL CMPIM 'R     Check what to do
1940: 2AC0 D0 06            BNE   FNOTRD
1950:                  **** Read 74LS571  ****
1960: 2AC2 20 BD 2B         JSR   FREAD
1970: 2AC5 4C 1D 2C         JMP   REPEAT And exit
1980: 2AC8 C9 43     FNOTRD CMPIM 'C
1990: 2ACA D0 EC            BNE   FNOTC
2000:                   **** Check for clear 74LS571  ****
2010: 2ACC 20 D2 2A         JSR   FCHECK
2020: 2ACF 4C 1D 2C         JMP   REPEAT
2030: 2AD2 A2 01     FCHECK LDXIM $01
2040: 2AD4 A0 00            LDYIM $00
2050: 2AD6 84 92            STY   NUMBER Clear flag location
2060: 2AD8 A9 90            LDAIM $90    Program port for reading
2070: 2ADA 8D 83 19         STA   CWR
2080: 2ADD A9 30            LDAIM $30    Value of port C-lower half
2090: 2ADF 8D 82 19  HALFK  STA   PORTC
2100: 2AE2 8C 81 19  QUARK  STY   PORTB  Set up address
2110: 2AE5 AD 80 19         LDA   PORTA  Load data
2120: 2AE8 29 0F            ANDIM $0F    Mask relavent bits
2130: 2AEA 05 92            ORA   NUMBER
2140: 2AEC 85 92            STA   NUMBER And save in flag location
2150: 2AEE C8               INY
2160: 2AEF D0 F1            BNE   QUARK  For 256 bytes
2170: 2AF1 A9 31            LDAIM $31    Value of port C-upper half
2180: 2AF3 CA               DEX
2190: 2AF4 F0 E9            BEQ   HALFK  And do other half
2200: 2AF6 A5 92            LDA   NUMBER Put flag on stack
2210: 2AF8 48               PHA
2220: 2AF9 4C 7F 2A         JMP   FCHKX  And output report
2230: 2AFC A2 00     FPROG  LDXIM $00
2240: 2AFE A0 00            LDYIM $00
```

```
2250: 2B00 A9 08     HALFB  LDAIM $08    Start on most significant bit
2260: 2B02 85 8E            STA   MASK
2270: 2B04 A9 80     FORBIT LDAIM $80    No of tries at blow bad locs
2280: 2B06 85 8F            STA   TRIES
2290: 2B08 A9 10            LDAIM $10    No of pulses after blow loc
2300: 2B0A 85 90            STA   EXTRA
2310: 2B0C 20 21 2B         JSR   FCORE  Go and blow bit
2320: 2B0F 46 8E            LSR   MASK   Step to next bit in byte
2330: 2B11 90 F1            BCC   FORBIT for four bits
2340: 2B13 C8               INY
2350: 2B14 D0 EA            BNE   HALFB  For 256 bytes
2360: 2B16 8A               TXA
2370: 2B17 D0 05            BNE   FPROGX
2380: 2B19 E6 8D            INC   FROM   +01 Increment pointer
2390: 2B1B E8               INX
2400: 2B1C D0 E2            BNE   HALFB  And do other half of PROM
2410: 2B1E C6 8D     FPROGX DEC   FROM   +01 Reset pointer
2420: 2B20 60               RTS          And exit
2430: 2B21 A9 80     FCORE  LDAIM $80    Set up port for programming
2440: 2B23 8D 83 19         STA   CWR
2450: 2B26 8C 81 19         STY   PORTB  Set up address
2460: 2B29 8A               TXA          Get last address bit
2470: 2B2A 49 18            EORIM $18    Add control bits
2480: 2B2C 8D 82 19         STA   PORTC
2490: 2B2F B1 8C            LDAIY FROM   Load data to blow
2500: 2B31 24 81            BIT   HL     Check-blowing up or lo half
2510: 2B33 30 04            BMI   LOW
2520: 2B35 6A               RORA         Shift upper half down
2530: 2B36 6A               RORA
2540: 2B37 6A               RORA
2550: 2B38 6A               RORA
2560: 2B39 25 8E     LOW    AND   MASK   Mask of current bit
2570: 2B3B F0 45            BEQ   FCORX  Nothing to program so exit
2580: 2B3D 8D 80 19         STA   PORTA  Set bit to program
2590: 2B40 A9 0D            LDAIM $0D    Flip power on
2600: 2B42 8D 83 19         STA   CWR
2610: 2B45 EA               NOP          Delay for power to settle
2620: 2B46 EA               NOP
2630: 2B47 EA               NOP
2640: 2B48 EA               NOP
2650: 2B49 EA               NOP
2660: 2B4A A9 06            LDAIM $06    Flip enable on
2670: 2B4C 8D 83 19         STA   CWR
2680: 2B4F EA               NOP          Delay for program pulse width
2690: 2B50 EA               NOP
2700: 2B51 A9 07            LDAIM $07        enable off
2710: 2B53 8D 83 19         STA   CWR
2720: 2B56 A9 0C            LDAIM $0C
2730: 2B58 8D 83 19         STA   CWR          power off
2740: 2B5B A9 90            LDAIM $90    Set port A ip to verify bit
2750: 2B5D 8D 83 19         STA   CWR
2760: 2B60 8C 81 19         STY   PORTB  Set address again
2770: 2B63 8A               TXA          Get upper address bit
2780: 2B64 09 30            ORAIM $30    Set other control bits
2790: 2B66 8D 82 19         STA   PORTC
2800: 2B69 AD 80 19         LDA   PORTA  Get bit
```

```
2810: 2B6C 25 8E          AND   MASK
2820: 2B6E D0 0E          BNE   BLOWN  Good the bit is blown
2830: 2B70 C6 8F          DEC   TRIES  Try to blow again
2840: 2B72 D0 AD          BNE   FCORE
2850: 2B74 98             TYA
2860: 2B75 48             PHA
2870: 2B76 20 00 28       JSR   VSTRNG Print a . for each failed bit
2880: 2B79 51             =     'Q
2890: 2B7A EA             NOP
2900: 2B7B 68             PLA
2910: 2B7C A8             TAY
2920: 2B7D 60             RTS          And give up
2930: 2B7E C6 90    BLOWN DEC   EXTRA  16 extra blows for luck
2940: 2B80 D0 9F          BNE   FCORE
2950: 2B82 60       FCORX RTS          And return
2960: 2B83 A2 01    FVERIF LDXIM $01
2970: 2B85 A0 00          LDYIM $00
2980: 2B87 84 92          STY   NUMBER Clear flag location
2990: 2B89 A9 90          LDAIM $90    Value of ctrl word for read
3000: 2B8B 8D 83 19       STA   CWR
3010: 2B8E A9 30          LDAIM $30    Value of Port C for lo half
3020: 2B90 8D 82 19 ALLVE STA   PORTC
3030: 2B93 8C 81 19 HALFC STY   PORTB  Set up address
3040: 2B96 B1 8C          LDAIY FROM   Get data from memory
3050: 2B98 24 81          BIT   HL     Check for upper or lower half
3060: 2B9A 30 04          BMI   LOWC
3070: 2B9C 4A             LSRA         Shift upper half down
3080: 2B9D 4A             LSRA
3090: 2B9E 4A             LSRA
3100: 2B9F 4A             LSRA
3110: 2BA0 4D 80 19 LOWC  EOR   PORTA  Match against data in PROM
3120: 2BA3 29 0F          ANDIM $0F    Mask bits of interest
3130: 2BA5 05 92          ORA   NUMBER And store in flag location
3140: 2BA7 85 92          STA   NUMBER
3150: 2BA9 C8       NOFLG INY          Do for 256 bytes
3160: 2BAA D0 E7          BNE   HALFC
3170: 2BAC CA             DEX
3180: 2BAD D0 06          BNE   DUN
3190: 2BAF E6 8D          INC   FROM   +01 Increment pointer
3200: 2BB1 A9 31          LDAIM $31    Value of port C for up half
3210: 2BB3 D0 DB          BNE   ALLVE  And do upper half
3220: 2BB5 C6 8D    DUN   DEC   FROM   +01 Reset pointer
3230: 2BB7 A5 92          LDA   NUMBER Put flag on stack
3240: 2BB9 48             PHA
3250: 2BBA 4C 68 2A       JMP   FAILQ  And print out result
3260: 2BBD A0 00    FREAD LDYIM $00
3270: 2BBF A2 01          LDXIM $01
3280: 2BC1 A9 90          LDAIM $90    Value of ctrl word for read
3290: 2BC3 8D 83 19       STA   CWR
3300: 2BC6 A9 30          LDAIM $30    Value of port C for lo half
3310: 2BC8 8D 82 19 HALFRD STA  PORTC
3320: 2BCB 8C 81 19 FREDLP STY  PORTB  Set up address
3330: 2BCE B1 8C          LDAIY FROM   Get data
3340: 2BD0 24 81          BIT   HL     Check for uper or lower half
3350: 2BD2 30 0E          BMI   LOWRF
3360: 2BD4 29 0F          ANDIM $0F    Upper-mask and save lo half
```

```
3370: 2BD6 85 92           STA   NUMBER    of current data
3380: 2BD8 AD 80 19        LDA   PORTA  Load half byte from PROM
3390: 2BDB 0A              ASLA         Shift it up
3400: 2BDC 0A              ASLA
3410: 2BDD 0A              ASLA
3420: 2BDE 0A              ASLA
3430: 2BDF 4C EB 2B        JMP   POSIT  Now assemble byte
3440: 2BE2 29 F0    LOWRF  ANDIM $F0    Save up half of current data
3450: 2BE4 85 92           STA   NUMBER
3460: 2BE6 AD 80 19        LDA   PORTA  Load half byte from PROM
3470: 2BE9 29 0F           ANDIM $0F    Mask of bits
3480: 2BEB 05 92    POSIT  ORA   NUMBER Assemble new data
3490: 2BED 91 8C           STAIY FROM   And save it
3500: 2BEF C8              INY
3510: 2BF0 D0 D9           BNE   FREDLP Do for 256 bytes
3520: 2BF2 CA              DEX
3530: 2BF3 D0 06           BNE   DUNFRD
3540: 2BF5 E6 8D           INC   FROM   +01 Increment pointer
3550: 2BF7 A9 31           LDAIM $31    Value of port C for up half
3560: 2BF9 D0 CD           BNE   HALFRD And do upper half
3570: 2BFB C6 8D    DUNFRD DEC   FROM   +01 Restore pointer
3580: 2BFD 60              RTS          And return
3590: 2BFE 20 00 28 BLOWQR JSR   VSTRNG Send prompt
3600: 2C01 42              =     'B
3610: 2C02 6C              =     'l
3620: 2C03 6F              =     'o
3630: 2C04 77              =     'w
3640: 2C05 20              =     ' 
3650: 2C06 3F              =     '?
3660: 2C07 EA              NOP
3670: 2C08 20 E6 FF        JSR   OSECHO Get answer
3680: 2C0B C9 59           CMPIM 'Y
3690: 2C0D D0 0E           BNE   REPEAT Not yes so escape
3700: 2C0F 20 00 28        JSR   VSTRNG
3710: 2C12 0D              =     $0D
3720: 2C13 42              =     'B
3730: 2C14 6C              =     'l
3740: 2C15 6F              =     'o
3750: 2C16 77              =     'w
3760: 2C17 69              =     'i
3770: 2C18 6E              =     'n
3780: 2C19 67              =     'g
3790: 2C1A 0D              =     $0D
3800: 2C1B EA              NOP
3810: 2C1C 60              RTS
3820: 2C1D A2 FF    REPEAT LDXIM $FF    Reset stack
3830: 2C1F 9A              TXS
3840: 2C20 20 00 28        JSR   VSTRNG And check for repeat
3850: 2C23 0D              =     $0D
3860: 2C24 52              =     'R
3870: 2C25 65              =     'e
3880: 2C26 70              =     'p
3890: 2C27 65              =     'e
3900: 2C28 61              =     'a
3910: 2C29 74              =     't
```

```
3920: 2C2A 20                =       '
3930: 2C2B 3F                =       '?
3940: 2C2C EA                NOP
3950: 2C2D A9 9B             LDAIM $9B     Release all control lines
3960: 2C2F 8D 83 19          STA   CWR
3970: 2C32 20 E6 FF          JSR   OSECHO Get reply
3980: 2C35 C9 59             CMPIM 'Y
3990: 2C37 D0 06             BNE   END
4000: 2C39 20 ED FF          JSR   OSCRLF new line
4010: 2C3C 6C 86 00          JMI   TOBLOW And do it again
4020: 2C3F 4C 40 28  END     JMP   START
4030: 2C42 20 00 28  CRCOUT  JSR   VSTRNG Print out CRC header
4040: 2C45 0D                =       $0D
4050: 2C46 43                =       'C
4060: 2C47 52                =       'R
4070: 2C48 43                =       'C
4080: 2C49 3A                =       ':
4090: 2C4A 20                =       '
4100: 2C4B EA                NOP
4110: 2C4C A5 94             LDA   CRC     +01 First byte
4120: 2C4E 20 53 2C          JSR   BYTOUT
4130: 2C51 A5 93             LDA   CRC     And second byte
4140: 2C53 48        BYTOUT  PHA           Print out byte in hex
4150: 2C54 4A                LSRA
4160: 2C55 4A                LSRA
4170: 2C56 4A                LSRA
4180: 2C57 4A                LSRA
4190: 2C58 20 5C 2C          JSR   DIGIT
4200: 2C5B 68                PLA
4210: 2C5C 29 0F     DIGIT   ANDIM $0F     Print out half byte
4220: 2C5E C9 0A             CMPIM $0A
4230: 2C60 90 02             BCC   ITDIGT
4240: 2C62 69 06             ADCIM $06
4250: 2C64 69 30     ITDIGT  ADCIM $30
4260: 2C66 4C F4 FF          JMP   OSWRCH
```