ace inc.

IPSO FACTO

DEVOTED ENTIRELY TO MICROCOMPUTER TECHNOLOGY

FEBRUARY 1986 ISSUE 46

INDEX

	PAGE
RADIO SHACK GRAPHICS TABLET by Richard Cox	4
PROGRAM LOGGER by Michael Smith	7
ABANDON THE 1802? by Richard m. Cox	13

DEPARTMENTS

VITAL INFORMATION	2
EDITORIAL	3
CLUB COMMUNIQUE	31
B.I.C.H. SHEET	16

ace inc. publication

IPSO FACTO is published by THE ASSOCIATION OF COMPUTER-CHIP EXPERIMENTERS (ace), A NON-PROFIT EDUCATION organization. Information in IPSO FACTO is beleived to be accurate and reliable. However, no responsibility is assumed by IPSO FACTO or THE ASSOCIATION OF COMPUTER-CHIP EXPERIMENTERS for its use; nor for any infringements of patents or rights of third parties which may result from its use.

Treasurer: Ken Bevis (416)-277-2495
Secretery: Michael Smith
Directors: Johhn Moris Fred Pluthero
Ken Bevis Mike Franklin
Membership: Fred Pluthero

Newsletter:

Chief Editor and Production Manager:
Fred Pluthero
Editors: Tony Hill
Publication: Dennis Mildon

Product Mailing:

Publications: Ed Leslie (416)-528-3222
Boards and Software: Michael Smith

Development:

Hardware: Ken Bevis Tony Hill
Mike Franklin

Software: Michael Smith

Club Mailing Address ACE IMK.
P.O. BOX 6464 STATION F
HAMILTON, Ontario
Canada
L9C 7C7

ARTICAL SUBMISSIONS:

The content of IPSO FACTO is voluntarily submitted by club members. While ACE assumes no responsibility for errors nor for infringement upon copyright, the Editors verify articale content as much as possible. ACE can use articals, both hardware and software, of any level, relating to any micro computer components, peripherals, and products. Please specify the equipment and/or software to which the articale applies. Typed articles are preferred and are printed first. Please send origanals, not phtocopy matcaterial. We will return photocopies of original if required.

PUBLICATION POLICY:

The newsletter staff assume no respors-ibility for articale errors nor for infringement upon copyright. The content of all articles will be verified, as much as possible, and limitations listed (i.e. Make of equipment, or auther of software). The newsletter will be published every other month, commencing in January. Delays may be incurred as result of loss of staff, postal disruption, lack of articles, etc. We apologize for such inconvenience, hoeever, thy are generally caused by factors beyond the control of the Club.

ITS BARGAIN TIME
SEE PRODUCT PAGE
FOR DETAILES

NOTE

NEW ADDRESS

ACE INC.
P.O. BOX 6464
STATION "F"
HAMILTON ONT.
CANADA L9C 7C7

MEMBERSHIP POLICY:

A membership is contracted on the basis of a Club year - January 1 to December 31
Each member is enttitled to, among other privileges of membership, all issues of IPSO FACTO published during the club year.

EDITORIAL

I would like to wish everyone a belated seasonal greeting. MAY 1986 bring you and yours health, happiness, and prosperity.

When I started publishing the bulletin, I had very little good material. That is why your bulletins for 1985 have been slow in comeing. I have enough material for issue #47. Issue #48 will be our swan song unless something miraculous happens. Lack of support by all but a small percentage of our members has made it impossible to continue producing a bulletin of quality.

What I propose we do, is return to a small local club, and do what our name implies experiment with new chips and circuits. This is the only way we can continue sharing information. The members will have to attend meetings to share or get information. Richard Cox, in his letter has expressed this view.

The problem is that a few people shared their knowledge, and experience, while the majority waited for someone els to contribute.

The next issue will have a cross compiller written in basic on a TRS 80. The author has tried to leave out all the instructions used only on the TRS 80. This will give you a means to program your 1802 system from a host computer.

If you have any comments please write me. Note our new addres.

...Fred...Cluthe.....

PS- NOTE OUR NEW ADDRESS

ACE INC.
P.O. BOX 6464 STATION "F"
HAMILTON ONTARIO CANADA
L9C 7C7

Also there are some members who never received the ipso, the reason for this was the membership list was not cept up to date due to our membership cherman moving. All ipso facto's are being printed so you will receive all issues owing you.

RADIO SHACK GRAPHICS TABLET

by Richard Cox

How many of you have envied the computers which have a graphics tablet as an input device? You know, the little pen which is moved about on a pad, which the computer screen senses. These little devices can be a lot of fun as well as useful. Well, up until now, the cost was prohibitive, several hundred dollars at least, and home construction difficult, even if you did know how they worked. The technologies differ, but the magic behind them is not too difficult when analyzed.

Well, it seems our old friend Radio Shack developed one for their color computers, and sells them for about $350.00, a little much for a $50.00 computer, eh? When the recent sale flyer came, offering this device for less than $100, it was too much to resist. Despite admonishments from the salesman about the compatibility with other computers, and that the warranty would be void if I connected it to some other computer, I had the guts spread all over my workbench five minutes after I got it home. No schematic was included, so I got out my voltmeter and IC data books and traced the thing out. Here are the results.

The "pen" is actually a ball point pen refill in a slim metal tube, attached by a two conductor shielded cable to the electronics box, which has a 40 pin 0.1" spacing edge connector on one end. The tablet is similarly connected by a 6 conductor shielded cable to the electronics box. There are 18 ICs in the box, including 2 quad Op Amps, a quad comparator, several CMOS and LSTTL devices, and an assortment of resistors and capacitors. A look at the service manual for the Color Computer confirmed the necessary interface which consists of 16 address lines, 8 data lines, a read signal, a clock at 895 KHZ, and power requirements of +5V, +12V, and -12V. Shouldn't be t. hard to figure out! As it turns out, it is incredibly easy. The device is simply read as three consecutive memory locations. One location gives the "X" or horizontal position as 0 to 255, the next gives the "Y" or vertical location as 0 to 191, and the third has 4 bits of status information such as pen proximity, pen down (there is a switch which activates when the pen is pressed down on the surface), and whether the pen is within the margins of the tablet, which extend about 1 inch in from the outline of the paper, except on the bottom, which has no margin defined.

It works by outputing a 110 KHZ signal from the pen tip, and detecting that signal in the tablet by reading a voltage across a series resistor divider which has its nodes connected to PC traces which extend across the vertical and horizontal planes in the tablet, one on each side of a large PC board in the plastic case. Four potentiometers in the tablet set the size and position of the area detected.

Hooking this device up to your 1802 system is simple. Plug an appropriate connector onto the edge connector, adding inverters in series with some of the address lines if necessary, and connecting the data lines directly to your data bus. Divide your 1.79 MHZ system clock by two for the clock input or add a 3.58 MHZ oscillator and divide by 4, connect your MRD-N through a inverter to the READ signal, and apply the proper supply voltages. The connector orientation and numbering is shown below.

```
1------ top side -------39
2----- bottom side -----40
```

Looking into connector with the open portion of the slot in the case down. (warranty label up)

HOOKUP

PIN	SIGNAL
1	-12V
2	+12V
6	CLOCK .89MHZ
9	+5V
10	D0
11	D1
12	D2
13	D3
14	D4
15	D5
16	D6
17	D7
18	READ-POS
19	A0
20	A1
21	ADDRESS-NEG
22	ADDRESS-NEG
23	ADDRESS-NEG
24	ADDRESS-POS
25	ADDRESS-POS
26	ADDRESS-NEG
27	ADDRESS-POS
28	ADDRESS-POS
29	ADDRESS-POS
30	ADDRESS-POS
31	ADDRESS-POS
34	GROUND
37	ADDRESS-POS
38	ADDRESS-POS
39	ADDRESS-POS

Observe that no address line designations were given for 14 of the 16 lines. Connect them so that all inputs marked ADDRESS-POS are at a "1" level, and all inputs marked ADDRESS-NEG are at a "0" level to enable the device. If desired, inverters can be added if you need more -NEG inputs. For example, to address the device at locations E8F0-E8F2, the address lines must look like this:

A15														A0
1	1	1	0	1	0	0	0	1	1	1	1	0	0	A1 A0

Since 6 NEG inputs are needed, add two inverters to any two of the POS inputs, and consider those now NEG inputs. Connect address lines A2, A3, A8, A9, A10, and A12 to these Neg inputs, and the remaining lines to the POS inputs, except A0 and A1, which connect directly.

The status register 4 bits are interpreted as follows. Bit 0 true indicates that the pen switch is closed, that is, pushed down onto a surface.
Bit 1 indicates that the pen is within 1" of the surface of the tablet.
Bit 2 indicates that the pen is within about 1" of the left or right margin.
Bit 3 indicates that the pen is within 1" of the top margin. Note that the position registers will wrap around when outside the margin, which is about 1" from the left, right, and top edges.

One important point. When the X position is read, all of the registers

will latch the current reading. When reading, then, read first the X, then the Y and status registers.

Software to run this thing is another matter. Many of the graphics boards, notably the ACE system, have the same 256 X 192 resolution of the tablet. This should allow the user to directly translate tablet output to screen position. Perhaps someone interested will develop some graphics programs to use this interesting device, and share it with everyone via IPSO FACTO.

PROGRAM LOGGER

By Michael Smith, 1 Cranleigh Court, Islington, Ontario, M9A 3Y2

Being an ardent machine language software writer, I have developed a large toolbox of routines that make my job of writing a program easier. However, accessing these utilities requires the constant use of SYMON to jump around in memory, and pieces of paper with addresses of routines on them, which tend to get lost extremely easily. This program was conceived while working on a disk analysis program in which such a problem arouse.

This program makes use of the fact that SYMON uses a vector table to store its I/O routine calls and long branches. If one goes into this table and changes the addresses, one can have another routine analyze whatever you type in. Thereby, a specific control code can easily be trapped, and another program executed whenever this is done. Program Logger traps the ASCII code 1F (control underline). Once the input stream has been interrupted, Program Logger asks you for your selection. At this point you can do various things.

If you hit return in response to its querry, the program that was being run previously is resumed; you continue typing as if the program had never been interrupted. If you hit escape in answer to its query, the program will print the list of programs that have been logged in its file space. If there are no programs logged, then you will be told.

However, the real power of the Program Logger is in the number keys. If you type a number, it will execute the appropiate program from its list of logged programs. This means that you can switch programs with two keystrokes from any place in a program, as long as it uses the SYMON vector table. The program logger can store up to nine programs in its file space. The programs are run with register three as the program counter.

To add a program to the list already logged, one uses the "a" command. You will be asked to supply a name (of up to 16 characters in length) and an address from which the program can be entered with PC=3. Finally, by typing "e" you can erase the entire selection of logged programs.

Program Logger is a very useful utility which is simple to use and is quite transparent to the user. It resides from F800 to FA00 in memory. It stores its data on page FA00. From FA00 to FA8F are the names of the programs that are logged. Starting at FA90 are the addresses of the start of the program. The number of logged programs is stored in address FAA3.

Once this program has been loaded into memory, it still has to be installed to trap the input of control underline. You can execute the installation routine with PC=0 from address F9F7 or with PC=3 from address F991. The installation routine changes SYMON's vector page to modify the input routine and the SYMON cold start address. Normally, whenever you reset SYMON from either the end of a routine or by hitting control-x, SYMON starts running at C018; and the first thing it does is to rewrite its vector table at FE00. This, of course, will turn off the Program Logger. Hence, one has to change the cold start address to avoid rewriting the vector table. This address is C04A on my system. Furthermore, one can no longer execute SYMON from address C000, for this will once more rewrite the vector table. SYMON has to be entered from FE00 to preserve the Program Logger.

The listing of the program should be detailed enough to explain its workings.

```
STRT F800  D4 CF00          CALL $CF00        CALL ASCII INPUT ROUTINE
     F803  FF 1F             SMI $1F          TRAP CONTROL UNDERLINE
     F805  3A 43             BNZ $43          GO BACK TO PROGRAM IF NOT
                                              CONTROL UNDERLINE
     F807  D4 FE0C           CALL $FE0C       PRINT A LINE FEED
     F80A  D4 C187           CALL $C187       PRINT FOLLOWING ASCII DATA
                                              END PRINT ON MSB=1
                                              NOPSs HAVE BEEN PROVIDED
                                              IN CASE YOUR ROUTINE ENDS
                                              ON A NULL (00).
     F80D  53 45 4C 45 43 54 49 4F 4E 3F A0   ASCII DATA: SELECTION?
     F818  C4                NOP              SPARE BYTE
     F819  F8 FA             LDI $FA          LOAD HIGH ADDRESS OF
     F818  B8                PHI 8            WORK TABLE INTO R8
     F81C  D4 CF00           CALL $CF00       CALL ASCII INPUT ROUTINE
     F81F  FF 0D             SMI $0D          CHECK FOR CARRIAGE RETURN
     F821  32 45             BZ  $45          GO TO EXIT IF TRUE
     F823  C4                NOP              SPARE BYTE
     F824  FF 0E             SMI $0E          CHECK FOR ESCAPE
     F826  32 48             BZ  $48          GO LIST THE FILES IF TRUE
     F828  FF 16             SMI $16          CHECK IF NOT NUMBER OR
     F82A  3B 1C             BNF $1C          ALPHABETIC, IF SO, GO GET
                                              ANOTHER COMMAND.
     F82C  FF 09             SMI $09          CHECK IF NUMBER
     F82E  3B 9E             BNF $9E          IF TRUE, GO RUN PROGRAM
     F830  9F                GHI F            GET ENTERED CHARACTER
     F831  F9 20             ORI $20          CONVERT TO LOWER CASE
     F833  FF 61             SMI $61          CHECK FOR "a"
     F835  32 0C             BZ  $0C          GO ADD A FILE
     F837  FF 04             SMI $04          CHECK FOR "e"
     F839  C2 F966           LBZ $F966        GO DELETE ALL FILES
     F83C  F8 07             LDI $07          LOAD ASCII BELL CHARACTER
     F83E  D4 $FE06          CALL $FE06       OOPS, USER MADE A MISTAKE
     F841  30 1C             BR  $1C          GIVE HIM ANOTHER CHANCE
     F843  9F                GHI F            GET ENTERED CHARACTER
     F844  05                RETN             RETURN TO CALLING PROGRAM
     F845  C0 F900           LBR $F900        RESUME PREVIOUS PROGRAM
LIST F848  F8 A3             LDI $A3          POINT TO NUMBER OF ENTRIES
     F94A  A8                PLO 8            REGISTER 8 IS POINTER TO
                                              WORK PAGE
     F848  08                LDN 8            LOAD NUMBER OF ENTRIES
     F84C  3A 70             BNZ $70          MAKE SURE THERE IS
                                              SOMETHING TO LIST
     F84E  D4 FE0C           CALL $FE0C       PRINT CARRIAGE RETURN
     F851  D4 FE0C           CALL $FE0C       PRINT CARRIAGE RETURN
     F854  D4 C187           CALL $C187       CALL ASCII OUTPUT
                                              SEE PREVIOUS NOTE
     F857  4E 4F 20 4C 4F 47 47 45 44 20      ASCII DATA: NO LOGGED
     F861  50 52 4F 47 52 41 4D 53 AE                     PROGRAMS.
     F86A  C4                NOP              SPARE BYTE
     F868  D4 FE0C           CALL $FE0C       PRINT CARRIAGE RETURN
     F86E  30 0A             BR  $0A          GET ANOTHER COMMAND
     F870  D4 FE0C           CALL $FE0C       PRINT CARRIAGE RETURN
```

```
     F873  D4 FE0C          CALL $FE0C       PRINT CARRIAGE RETURN
     F876  08               LDN 8            LOAD NUMBER OF ENTRIES
     F877  AA               PLO A            PUT NUMBER INTO COUNTER
     F878  F8 31            LDI $31          LOAD ASCII "1"
     F87A  BA               PHI A            STORE IN RA.1
     F87B  F8 00            LDI $00          POINT TO BOTTOM OF ASCII
                                             NAME FILES
     F87D  A8               PLO 8            R8 IS FILE POINTER
     F87E  9A               GHI A            GET ASCII NUMBER
     F87F  D4 FE06          CALL $FE06       CALL ASCII OUTPUT, PRINT
                                             THE NUMBER OF THE FILE
     F882  D4 C187          CALL $C187       CALL STRING OUTPUT ROUTINE
     F885  2E A0                             ASCII DATA:
     F887  C4               NOP              SPARE BYTE
     F888  48               LDA 8            LOAD BYTE TO PRINT
     F889  D4 FE06          CALL $FE06       PRINT THE CHARACTER
     F88C  88               GLO 8            CHECK IF THIS FILE
     F88D  FA 0F            ANI $0F          NAME IS FINISHED (16 CHAR)
     F88F  3A 88            BNZ $88          CONTINUE UNTIL ALL DONE
     F891  D4 FE06          CALL $FE06       CALL CARRIAGE RETURN
     F894  2A               DEC A            DECREMENT COUNTER
     F895  8A               GLO A            FETCH COUNTER, CHECK
     F896  32 0A            BZ  $0A          IF WE ARE DONE
     F898  9A               GHI A            FETCH ASCII NUMBER
     F899  FC 01            ADI $01          ADD ONE TO ASCII NUMBER
     F89B  BA               PHI A            PUT PACK INTO REGISTER
     F89C  30 7E            BR  $7E          PRINT ANOTHER NAME
NUM  F89E  9F               GHI F            GET ENTERED NUMBER
     F89F  FF 30            SMI $30          CONVERT ASCII DIGIT
                                             INTO HEX NUMBER
     F8A1  52               STR 2            STORE TEMPORARILY
     F8A2  F8 A3            LDI $A3          POINT TO NUMBER OF ENTRIES
     F8A4  A8               PLO 8            R8 IS POINTER
     F8A5  08               LDN 8            LOAD NUMBER OF ENTRIES
     F8A6  F7               SM               CHECK IF YOU WANT TO
                                             ACCESS AN EXISTING FILE
     F8A7  33 CA            BDF $CA          IF HE DID, GO FETCH THE
                                             ADDRESS OF THE ROUTINE
     F8A9  D4 FE0C          CALL $FE0C       PRINT A CARRIAGE RETURN
     F8AC  D4 C187          CALL $C187       CALL ASCII STRING OUTPUT
     F8AF  53 45 4C 45 43 54 49 4F 4E 20     ASCII DATA: SELECTION
     F8B9  4E 4F 54 20 41 56 41 49 4C 41                 NOT AVAILABLE.
     F8C3  42 4C 45 AE
     F8C7  C4               NOP              SPARE BYTE
     F8C8  30 07            BR $07           GIVE HIM ANOTHER CHANCE
     F8CA  9F               GHI F            WE'VE GOT THIS FAR, ASSUME
                                             ITS LEGAL, THEREFORE GET
                                             ENTERED NUMBER.
     F8CB  FF 31            SMI $31          CONVERT TO HEX DIGIT -1
     F8CD  FE               SHL              MULTIPLY BY TWO, FOR
                                             ADDRESS IS TWO BYTES LONG
     F8CE  FC 90            ADI $90          ADD OFFSET OF 90 TO POINT
                                             TO ADDRESS TABLE
```

```
        F8D0  A8                LO 8              R8 IS POINTER
        F8D1  93                GHI 3             WE HAVE TO CHANGE PC
        F8D2  BA                PHI A             SO PREPARE REGISTER A
        F8D3  F8 D7             LDI $D7
        F8D5  AA                PLO A             LOW BYTE
        F8D6  DA                SEP A             JUST CHANGED PC
        F8D7  48                LDA 8             LOAD HI BYTE OF ADDRESS
        F8D8  B3                PHI 3             STORE IN REGISTER 3
        F8D9  08                LDN 8             LOAD LO BYTE OF ADDRESS
        F8DA  A3                PLO 3             STORE IN REGISTER 3
        F8DB  D3                SEP 3             GO EXECUTE THE DESIRED
                                                  PROGRAM WITH PC=3
ADD     F8DC  F8 A3             LDI $A3           POINT TO NUMBER ENTRIES
        F8DE  A8                PLO 8             REGISTER 8 IS POINTER
        F8DF  08                LDN 8             LOAD NUMBER OF ENTRIES
        F8E0  FF 09             SMI $09           MAXIMUM OF 9 ENTRIES
        F8E2  CA F905           LBNZ $F905        LESS THAN 9 ENTRIES, ALLOW
                                                  USER TO ADD ANOTHER ONE
        F8E5  D4 FE0C           CALL $FE0C        PRINT CARRIAGE RETURN
        F8E8  D4 C187           CALL $C187        CALL ASCII STRING OUTPUT
        F8EB  50 52 4F 47 52 41 4D 20 54 41 42    ASCII DATA: PROGRAM TABLE
        F8F6  4C 45 20 49 53 20 46 55 4C 4C AE              IS FULL.
        F901  C4                NOP               SPARE BYTE
        F902  C0 F807           LBR $F807         GET ANOTHER COMMAND
        F905  08                LDN 8             LOAD NUMBER OF ENTRIES
        F906  FC 01             ADI $01           ADD 1 TO NUMBER OF ENTRIES
        F908  58                STR 8             STORE NUMBER OF ENTRIES
        F909  FF 01             SMI $01           NUMBER OF ENTRIES-1
        F90B  FE FE FE FE       SHL SHL SHL SHL   SHIFT TO HIGH NIBBLE
                                                  (MULTIPLY BY 16)
                                                  TO POINT TO NAME FIELD
        F90F  F9 0F             ORI $0F           POINT TO END OF FIELD
        F911  A8                PLO 8             REGISTER 8 IS POINTER
        F912  F8 20             LDI $20           LOAD ASCII SPACE
        F914  58                STR 8             GO ABOUT ERASING FIELD
        F915  88                GLO 8             LOW POINTER ADDRESS
        F916  FA 0F             ANI $0F           ONLY WANT LOWER NIBBLE
        F918  32 1D             BZ $1D            ARE WE AT THE START
                                                  OF THE NAME FIELD?
        F91A  28                DEC 8             NO, SO MOVE ONE LOWER
        F91B  30 12             BR  $12           CONTINUE ERASING FIELD
        F91D  D4 FE0C           CALL $FE0C        PRINT A CARRIAGE RETURN
        F920  D4 C187           CALL $C187        CALL STRING OUTPUT ROUTINE
        F923  50 52 4F 47 52 41 4D 20             ASCII DATA: PROGRAM
        F92B  4E 41 4D 45 3F A0                           NAME?
        F931  C4                NOP               SPARE BYTE
        F932  D4 CF00           CALL $CF00        CALL INPUT ROUTINE
        F935  FF 08             SMI $08           CHECK FOR BACKSPACE
        F937  3A 45             BNZ $45           CONTINUE IF NOT
        F939  88                GLO 8             GET LOW BYTE OF POINTER
        F93A  FA 0F             ANI $0F           ONLY WANT LOWER NIBBLE
        F93C  32 32             BZ  $32           MAKE SURE POINTER DOESN'T
                                                  GO INTO PREVIOUS FIELD
```

```
        F93E  29                DEC 8           DECREMENT POINTER
        F93F  30 32             BR  $32         GET ANOTHER CHARACTER
        F941  C4                NOP             SPARE BYTE
        F942  C4                NOP             SPARE BYTE
        F943  C4                NOP             SPARE BYTE
        F944  C4                NOP             SPARE BYTE
        F945  FF 05             SMI $05         CHECK FOR CARRIAGE RETURN
        F947  32 51             BZ  $51         LEAVE INPUT IF RETURN
        F949  9F                GHI F           GET CHARACTER
        F94A  58                STR 8           STORE CHARACTER IN FIELD
        F94B  18                INC 8           MOVE POINTER ONE OVER
        F94C  88                GLO 8           FETCH POINTER
        F94D  FA 0F             ANI $0F         ONLY WANT LOWER NIBBLE
        F94F  3A 32             BNZ $32         CONTINUE INPUT UNTIL 16
                                                CHARACTERS HAVE BEEN
                                                ENTERED
        F951  D4 C100           CALL $C100      CALL SYMON'S ROUTINE TO
                                                FETCH START ADDRESS
        F954  F8 A3             LDI $A3         POINT TO NUMBER OF ENTRIES
        F956  A8                PLO 8           REGISTER 8 IS POINTER
        F957  08                LDN 8           LOAD NUMBER OF ENTRIES
        F958  FF 01             SMI $01         NEED ENTRIES-1
        F95A  FE                SHL             MULTIPLY BY 2
        F95B  FC 90             ADI $90         ADD OFFSET TO ACCESS
                                                ADDRESS TABLE
        F95D  A8                PLO 8           REGISTER 8 IS POINTER
        F95E  9D                GHI D           GET HI ADDRESS OF PROGRAM
                                                FROM SYMON ROUTINE
        F95F  58                STR 8           STORE IN ADDRESS TABLE
        F960  18                INC 8           POINT TO LOW BYTE IN TABLE
        F961  8D                GLO D           GET LOW ADDRESS OF PROGRAM
        F962  58                STR 8           STORE LOW ADDRESS IN TABLE
        F963  C0 F807           LBR $F807       GO GET ANOTHER INPUT
DEL     F966  D4 FE0C           CALL $FE0C      PRINT CARRIAGE RETURN
        F969  D4 C187           CALL $C187      CALL STRING OUTPUT ROUTINE
        F96C  43 4F 4E 46 49 52 4D 3F 20
                                ASCII DATA: CONFIRM?
        F975  28 59 2F 41 4E 59 29 A0
                                            (Y/ANY)
        F97D  C4                NOP             SPARE BYTE
        F97E  D4 CF00           CALL $CF00      CALL ASCII INPUT ROUTINE
        F981  F9 20             ORI $20         CONVERT TO LOWER CASE
        F983  FF 79             SMI $79         CHECK FOR "y"
        F985  CA F807           LBNZ $F807      RETURN IF NOT
        F988  F8 A3             LDI $A3         POINT TO NUMBER OF ENTRIES
        F98A  A8                PLO 8           DON'T ERASE FILES, JUST
        F98B  F8 00             LDI $00         SET NUMBER OF ENTRIES
        F98D  58                STR 8           TO ZERO
        F98E  C0 F807           LBR $F807       GET ANOTHER INPUT.
INST    F991  F8 FE             LDI $FE         POINT TO VECTOR TABLE
PC=3    F993  B8                PHI 8
        F994  F8 02             LDI $02         POINT TO COLD START VECTOR
        F996  A8                PLO 8
        F997  F8 4A             LDI $4A         CHANGE VECTOR TO "C0 C04A"
        F999  58                STR 8
```

```
        F99A  18              INC 8           POINT TO INPUT VECTOR
        F99B  18              INC 8
        F99C  F8 F8           LDI $F8         CHANGE INPUT VECTOR TO
                                              POINT TO PROGRAM LOGGER
        F99E  58              STR 8           CHANGE HI BYTE
        F99F  F8 00           LDI $00
        F9A1  18              INC 8           POINT TO LOW BYTE
        F9A2  58              STR 8           CHANGE LOW BYTE
        F9A3  F8 FA           LDI $FA         POINT TO WORK PAGE
        F9A5  B8              PHI 8
        F9A6  F8 A3           LDI $A3         POINT TO NUMBER OF ENTRIES
        F9A8  A8              PLO 8
        F9A9  F8 00           LDI $00         RESET NUMBER OF ENTRIES
        F9AB  58              STR 8           TO ZERO
        F9AC  04 FE0C         CALL $FE0C      PRINT CARRIABLE RETURN
        F9AF  04 C187         VSLL $C187      CALL STRING OUTPUT ROUTINE
        F9B2  50 52 4F 47 52 41 4D 20 4C 4F 47 47   ASCII DATA: PROGRAM LOGGER
        F9BE  45 52 20 49 4E 53 54 41 4C 4C 45 44                 INSTALLED.
        F9CA  AE
        F9CB  C4              NOP             SPARE BYTE
        F9CC  C0 FE00         LBR $FE00       ENTER SYMON
        F9CF  C4              NOP             SPARE BYTE
        F9D0  04 FE0C         CALL $FE0C      PRINT CARRIAGE RETURN
        F9D3  04 C187         CALL $C187      CALL STRING OUTPUT ROUTINE
        F9D6  52 45 53 55 4D 49 4E 47 20 50 52 45   ASCII DATA: RESUMING
        F9E2  56 49 4F 55 53 20 50 52 4F 47 52 41                 PREVIOUS
        F9EE  4D AE                               PROGRAM.
        F9F0  C4              NOP             SPARE BYTE
        F9F1  04 FE0C         CALL $FE0C      PRINT CARRIAGE RETURN
        F9F4  C0 CF00         LBR $CF00       LONG BRANCH TO ASCII
                                              INPUT ROUTINE
INST    F9F7  90              GHI 0           GOING TO CHANGE PC
PC=0    F9F8  B3              PHI 3           SO PREPARE REGISTER 3
        F9F9  F8 91           LDI $91         START PROGRAM AT F991
        F9FB  A3              PLO 3
        F9FC  D3              SEP 3           GO TO INSTALL PROGRAM
```

ABANDON THE 1802?

Richard M. Cox
2670 Calle Abedul
Thousand Oaks, Ca. 91361

I have been observing the decline of the 1802, mostly via IPSO FACTO for the past couple of years. I have some observations and comments on the reason for this discouraging trend. The development of mass market computers with advanced capabilities such as color graphics, sound, tons of cheap game and business software has obviously drawn many people away from the "hacker" to the "user" category. And why not? Why spend hundreds of hours building and programming a computer, with no chance of having anything near the capabilities of, say, a Commodore 64? Even if it were possible to build the hardware capabilities equal to such a machine (and it is), even a group as large as ACE used to be cannot possibly develop the software base of these machines. There is simply no market for 1802 software, and after all, money is what software development is all about.

I got started with the 1802 for industrial use in about 1978, when the 1802 was the only CMOS processor around. My company bought an ELF II from good old Netronics, and I learned about hardware and software on it. I then decided it was time to build a home computer based on the 1802, that could use the Netronics software with cassette interface. Over the past six years or so, I have built my machine into what it is today, a system with 56K of RAM, a Netronics Smart terminal, RCA compatible card cage and bus, two 5-1/4 disk drives, and my disk operating system, ELFDOS, which has been described in IPSO FACTO. Hundreds, perhaps thousands of hours were spend building the hardware, writing software, which, hopefully others could use. I joined the ELF of the Valley club early on, when 20 to 30 members, all of whom had 1802 systems, gave up one Saturday morning a month to share ideas, help each other, and have a good time. We still meet each month, but we are lucky if 10 people show, and only 4 or 5 still have 1802 systems, while others bring TRS-80s and Macintoshes. All of the remaining 1802 users are using different hardware, including John Deering's ACE system, but all are using my disk controller and ELFDOS software, which includes an Editor, Assembler, many utilities, FORTH, and BASIC. It is gratifying to see this small band hang on to their systems, and enjoy some of the fruits of my labor. All have contributed in some way, to the software base to add printer buffers, emulators similar to Cuddly's TRACE, a master disk directory database, etc.

My introduction of ELFDOS via IPSO FACTO, along with an offer to sell the controller and software at very reasonable prices, met with disappointing response. Only 3 people bought the system, and with a little help from me, brought the system up on their varied hardware.

The ability to simply pop in a diskette, and type a program name to bring it into memory and run it, almost makes one think that they have a "real" computer. Since all computers are alike in the dark, that is, they can all be programmed to do anything any other one can do, means that all that is required is a coordinated effort in software development.

I still work with computers in advanced systems using 16 bit processors, megabytes of RAM, co-processing, and all the goodies. Have I been tempted to abandon the 1802 and build or buy a new system? Sure I have, many times. My son has a Commodore 64, and if I ever have the urge to blast some aliens or explore a cave, there it is. I had an opportunity recently to buy an IBM PC compatible system at a good price. But why? So I can create

a database of my wife's recipes or balance my checkbook? Big deal! If I ever build another machine, it will be something I can explore with, create something new with. In the words of my friend and fellow 1802er, Frank Lane "Why do I need another computer, I don't fully understand this one yet?"

I have nothing against those who wish to advance their knowledge by exploring new hardware and software, but let's not forget our roots. After all, ACE still stands for the Association of Computer-Chip Experimenters. So let's experiment. Anyone ever hear of a Z8000? It is a 16 bitter with an architecture similar to the 1802, i.e. lots of general purpose registers. How about the BYTE articles on the HD64180, a super CMOS Z80. I am really tempted by that one for a neat portable, with expansion bus for my own creations, and CP/M compatibility.

If we must change to survive, let's remember why we all got into this crazy hobby in the first place. How about a member poll on which way to go?

#15

CLUB COMMUNIQUE

NAME:_____ DATE:_____

PRODUCT ORDER	QTY.	UNIT PRICE	TOTAL
CPU BOARD	____	~~50.00~~ 30.00	____.__
BACKPLANE & I/O BOARD VER.2	____	~~40.00~~ 25.00	____.__
FRONT PANEL WITH EPROM PROG., CLOCK	____	~~35.00~~ 25.00	____.__
VDU BOARD VER.2 "SPECIAL"	____	~~25.00~~ 25.00	____.__
EPROM/RAM (BYTEWIDE & EPROM) "SPECIAL"	____	~~25.00~~ 25.00	____.__
80 X 25 VIDEO BOARD	____	~~50.00~~ 25.00	____.__
DISK CONTROLLER BOARD VER.2	____	~~50.00~~ 25.00	____.__

BOARDS AVAILABLE WILL BE SHIPPED. EXCESS PAYMENT WILL BE RETURNED.
WITH THE PURCHASE OF A COMPLETE SET OF BOARDS WE WILL GIVE A 30% DISCOUNT
TOTAL PRICE WITH DISCOUNT=137.50 + 5.00 SHIPPING & HANDLING

SOFTWARE SUPPLIED IN NETRONICS CASSETTE FORMAT

Fig. FORTH - 6K - 0000H	____	$10.00	____.__
Tiny PILOT - 2K - 0000H	____	10.00	____.__
SYMON - 2K - 0000H	____	10.00	____.__
SYDOS - 4K - D000H	____	10.00	____.__
CHIP 8AE - 1.5K-1000H	____	10.00	____.__
SPACE WAR - disk - 32k & tape - 32k	____	15.00	____.__
COSMIC CONQUEST - tape - 32k	____	10.00	____.__

PRICE NOTE

Prices listed are in local funds. Americans and overseas pay in U.S. funds Canadians in Canadian funds. Overseas orders: for all items add $10.00 for air mail postage. Please use mony orders or bank draft for promp shipment. Personal cheques require up to six weeks for bank clearence prior to shipping orders.

SALES POLICY

ALL OF THE ABOVE PRODUCTS WORK IN A ACE configured microcomputer. We will endever to assist in custom applications, but assume no liability for such use. Orders will be shipped as promptly as payment is guaranteed.

16

B.I.C.H. SHEET

BITCHES, INFORMATION, COMMENTS, HELP

29 August 1985

Association of Computer-Chip Experimenters Inc.
P.O. Box 581
Burlington, Ontario
Canada L7R 3Y5

Attn: Development, Hardware

About two and a half years ago I bought the original disc drive controller board kit, designed for a certain CDC 8" disc drive (I have misplaced my instructions, and don't recall the exact model number).

I had no luck procuring the recommended CDC drive, but the kit instructions said it would "probably" work with a certain Shugart drive. A few months ago I had the opportunity to get some of the Shugart drives in question (model no. 800-2). Unfortunately, no data was provided, and Shugart Associates ~~de~~ capute.

Would any of you happen to have information on the pin-outs for the Shugart drive? Or have any ideas where I could find them? Or how I could best figure them out? I need to know some of the jumper connections too.

Thanks in advance for any help you might be able to give me.

Sincerely yours,

Bart Mathias

Bart Mathias
4894-3 Kilauea Ave.
Honolulu HI 96816
USA

I would like some information on the "Extended ELF Basic" which was mentioned in I.F. #32 page 15.

Sincerely,

E. L. Smothers
5022 Judy Lynn
Memphis, Tenn. 38118
USA