Ipso Facto

ISSUE 28 MARCH 1982

A PUBLICATION OF THE ASSOCIATION OF THE COMPUTER-CHIP EXPERIMENTERS (ACE) 1981

1981/82 EXECUTIVE OF THE ASSOCIATION OF COMPUTER-CHIP EXPERIMENTERS

President:	John Norris	416-239-8567	Vice-President:	Ken Bevis	416-277-2495
Treasurer:	Mike Franklin	416-878-0740	Secretary:	Tony Hill	416-523-7368
Directors:	Bernie Murphy Fred Pluthero	416-845-1630 416-389-4070			
Newsletter:			Membership:	Bob Silcox Earle Laycock	416-681-284B
Production Manager:	Mike Franklin	416-878-0740	Program Convener:	Bernie Murphy Bert Dekat	
Editors:	Fred Feaver Steve Carter Bob Siddall Tony Hill		Tutorial/Seminars:	Ken Bevis Fred Feaver	
Advertizing:	Fred Pluthero	416-389-4070	Draughtsman:	John Myszkowski	
Publication:	Dennis Mildon John Hanson				
Hardware & R. and D.	Ken Bevis Don McKenzie Fred Pluthero Dave Belgrave	416-277-2495	Software:	Wayne Bowdish	416-388-7116
			Product Mailing:	Ed Leslie	416-528-3222

CLUB MAILING ADDRESS:

A.C.E.
c/o Bernie Murphy
102 McCraney Street East
Oakville, Ontario
Canada
L6H 1H6
Phone: 416-845-1630

CLUB MEETINGS:

Meetings are held on the second Tuesday of each Month, September through June at 7:30 in Room B123, Sheridan College, 1430 Trafalgar Road, Oakville, Ontario. A one hour tutorial proceeds each meeting. The college is located approximately 1.0 km north of the QEW, on the west side. All members and interested visitors are welcome.

ARTICLE SUBMISSIONS:

The majority of the content of Ipso Facto is voluntarily submitted by club members. While we assume no responsibility for errors nor for infringement upon copyright, the Editorial staff verify article content as much as possible. We can always use articles, both hardware and software, of any level or type relating directly to the 1802 or to micro computer components, periferals, products, etc. Please specify the equipment or support software upon which the article content applies. Articles which are typed are prefered, and usually printed first, while handwritten articles require some work. Please, please send original, not photocopy material. We will return photocopies of original material if requested. Photocopies usually will not reproduce clearly.

ADVERTISING POLICY

ACE will accept advertising for commercial products for publication in Ipso Facto at the rate of $25 per quarter page per issue with the advertiser submitting camera-ready copy. All advertisements must be pre-paid.

PUBLICATION POLICY

The newsletter staff assume no responsibility for article errors nor for infringement upon copyright. The content of all articles will be verified, as much as possible and limitations listed (ie Netronics Basic only, Quest Monitor required, requires 16K at 0000-3FFF etc.). The newsletter staff will attempt to publish Ipso Facto by the first week of: Issue 25 - Oct 81, 26 - Dec 81, 27 - Feb 82, 28 - Apr 82, 29 - Jun B2, and 30 - Aug 82. Delays may be incurred as a result of loss of staff, postal disruptions, lack of articles, etc. We apologize for such inconvenience, however they are generally caused by factors beyond the control of the club.

MEMBERSHIP POLICY

A membership is contracted on the basis of a club year - September through the following August. Each member is entitled to, among other privileges of membership, all 6 issues of Ipso Facto published during the club year.

EDITOR'S CORNER

ACE Convertion

ACE proposes to host a weekend seminar/convention in early August this year, tentatively at a Hamilton location. The club has been successful in lining up two commercial 1802 users as speakers and hope to add RCA and several other "applications" users. Plan to set aside the first weekend in August. Hamilton is an hour drive from Buffalo, and within reach of most of our members. We are attempting to arrange low cost accommodation.

New ACE Products - Dynamic Memory

At last, the 64k dynamic board is here, in stock. For less than $125.00, you can add 64k of 4116 dynamic memory to your ACE buss. The 6 X 9.5 inch board (standard ACE) is all CMOS, with gold edge connectors and extensive documentation. The board produces 4k block shadow decoding to disable RAM for EPROM, ROM or memory map space. Or board provision for adaptation to the new 64k X 1 chips. CHEAP MEMORY IS HERE!

The board schematic is reproduced on page 4.

Proposed ACE Micro Board

The new ACE 1802/6 micro board is well advanced and should be available by the summer.

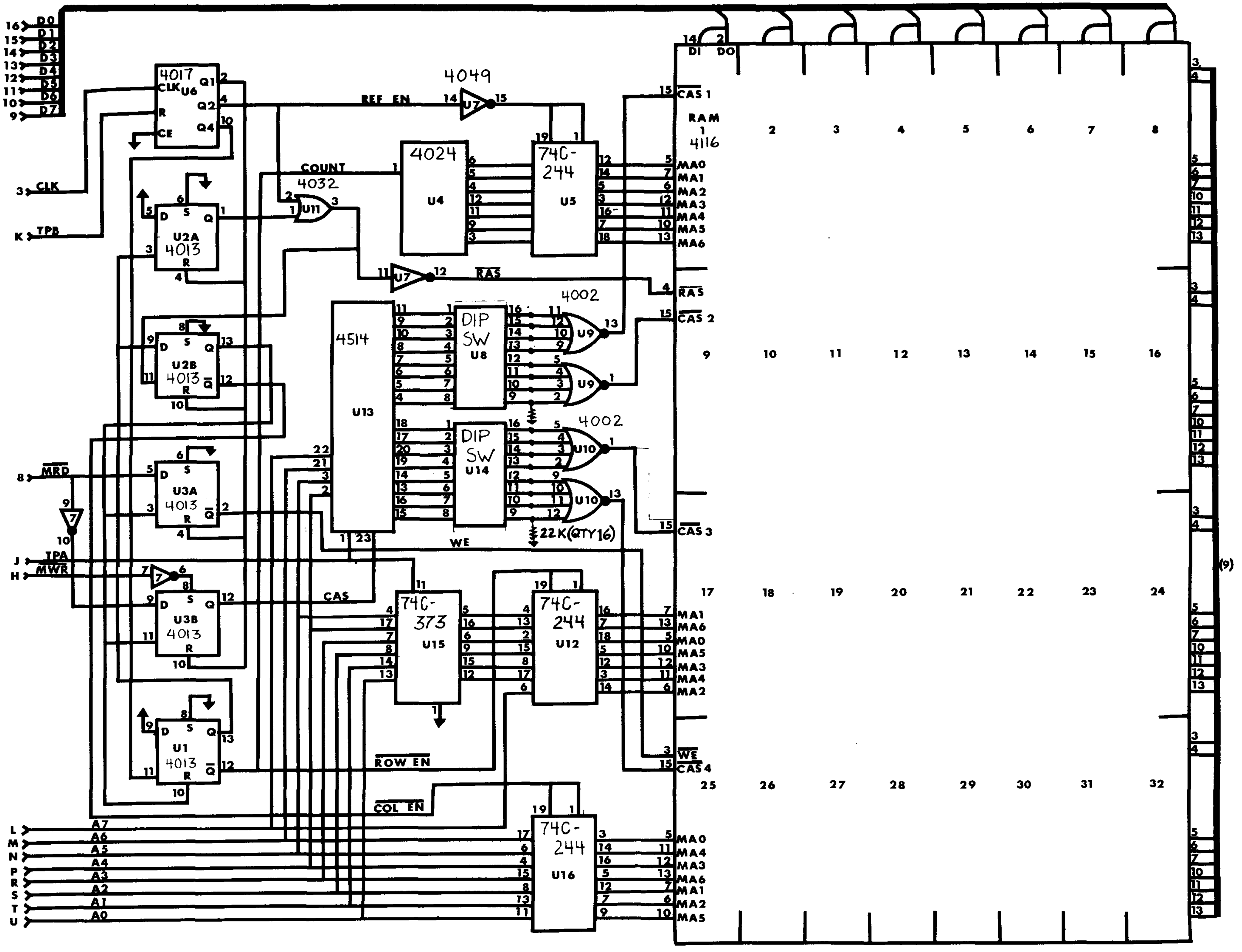
New Backplane and I/O Board

The club has produced version 2 of its 44 pin backplane and added cassette, parallel, serial (RS232C and TTL) input/output devices and a clock crystal to the board, doing away with the cassette relay controller and 16K block decoder. The 7" X 13" board is now in stock and available for $40.00. A separate I/O section is nearing completion which may be added to the existing backplane to convert it to the same features. This board will be available for $20.00 in April, 1982.

Netronics Adapter Board Bugs

A design error has been discovered on the NAB board. When LOAD and $\overline{\text{CLEAR}}$ were brought from the NETRONICS buss to the ACE buss, the connecting traces were reversed. Netronics pin 10 should connect to ACE pin 3 and Netronics pin 14 should connect to ACE 4. The correction is most easily be made near the angled Netronics edge connector, utilizing the plate through holes.

The RS232C-Db25 connector is not standard pin out, and care should be exercised connecting terminal and modem devices. Check the device connector pin out and adjust it on the NAB accordingly.


D0
D1
D2
D3
D4
D5
D6
D7
16
15
14
13
12
11
10
9
3 CLK
K TPB
8 MRD
J TPA
H MWR
L A7
M A6
N A5
P A4
R A3
S A2
T A1
U A0
4017
U6
CLK
R
CE
Q1
Q2
Q4
4049
U7
REF EN
COUNT
4032
U11
4024
U4
74C-244
U5
U2A
4013
U2B
4013
U3A
4013
U3B
4013
U1
4013
D
S
Q
R
RAS
4514
U13
DIP SW
U8
DIP SW
U14
4002
U9
4002
U10
22K(QTY16)
WE
CAS
74C-373
U15
74C-244
U12
ROW EN
COL EN
74C-244
U16
DI
DO
CAS 1
CAS 2
CAS 3
CAS 4
RAM 1 4116
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
MA0
MA1
MA2
MA3
MA4
MA5
MA6
WE
(9)

MEMBERS CORNER

FOR SALE

M. Sachse, 36 Jean Dr., N. Attleboro, MA. 02760
- complete ELFII-G.B., 1X4K RAM. Video Display, Cherry Keyboard, Video 100 (10mHZ) Monitor, 32K dynamic (wire wrapped) - $330.00 U.S.

D. Schuler, 3032 Avon Rd., Bethlehem, PA. 18017, 215-865-1188
- 2 - 8086 CPU @$45 or trade for 32 - 2114LA's.

A. Boisvent, 4830 des Pervenches, Orsainville, P.Q. G1G 1R7
- used 8" disk @$3.00, several 5 ft. by 20 wire ribbon cable @$2.00.

J. Briante, 18 Allison Pl., Guelph, Ont. N1H 6X7, 519-578-5369
- complete ELF II - G.B., 2X4K RAM, cassette controller, documentation, working $250.00 CDN.
- 1 - COPE 1030 Selectric Terminal - RS-232 - IBM Correspondence APL code, manuals - $400.00 CDN.

T. Crawford, 50 Brentwood Dr., Stoney Creek, Ont. L8G 2W8, 416-662-3603
- Intel single board computer SBC 80/10A including documentation - $300.00.
- ASR 33 teletype (incl. paper tape reader and punch) RS-232 - documentation - $300.00.

M. Olah, 324 Grant Avenue, Cuyahoga Falls, Ohio 44221, 216-928-4160
- complete ELF II, G.B., 3X4K RAM, 2X16k RAM, NOM board, terminal with Sony monitor - consider offers - need cash for tuition!!!
- Quest Super ELF, super expansion board, 4K ROM, SSM Video, keyboard - $175.00 U.S.

D. Thornton, 1403 Mormac Rd., Richmond, VA. 23229
- 2 - 2708, 1-C8702A-4, 1-C1702A-2, 1-MM5736N, 1-MK5012P, 1-CT5005, 1-MM5311N, 1-AY-5-1013A - FREE!!! for postage costs.
- 1 - paper tape punch - not much info. - from check writer machine - write for details.

F. Shinyei, 10545 129 St., Edmonton, Alta. T5N 1W9
- RCA boards - VP 595 - $25.00 - VP550-$45.00, VP576-$17.00 - VP700-$35.00 - VP710-$7.50
- MM57109 math board (home made) - $35.00.
- Cybernex 6XY16 video board - $110.00.
- Bell 103 modem - $25.00
- SWTPC 64 X 16, K5B keyboard - $30.00

ERRATA

- the following corrections have been reported for recently published articles. My thanks to the contributors for the corrections, and apologies to the authors for errors created by the Editoral staff.

1802 Serial I/O Board (I.F. #25, p. 28)
- by Tom Crawford

p28 - as published, page 34 should follow paragraph 4 on page 28.

p36 - table 1 - 600 baud - count value (LO) should be BA not BR

p36 - last line should be (#FF00 to FF0F)

p41 - PIN 21 of 1802 Edge commentor is I/O select
PIN 43 of 1802 Edge commentor is -5V regulated

p43 - serial I/O edge connector (top of page refer to page 41 for pin numbers.

U9 - $\overline{CS}$ is pin 21 not 1, 2; $\overline{WR}$ is PIN 23
U13 - gate out of U10 - pin are 12, 11 and 13.

Kaleidoscope (I.F. #26, p. 14)
- D. Ruske, Callauauga Rd., Waupun Wi., USA, 53963

There were a few typos in the screen layout data: 0247 should be 0E, 024A should be EE, 02BA should be 8D, an 02BB should be 75. Also, if your using an ELF II change 0040 from 61 to 69. Thanks to V. Cayer for an excellent program.

Kingdom (I.F. #25, p. 18)
- L. A. Hart

Mr. Hart found a couple of errors in his Kingdom program as published in Ipso Facto #25

line	should be;
500	G=G+L/4/(L*9/P+P*9/L)*M*(RND(Ø,4)-Z)-(A+P+S)/19*F
1310	1FR<1PR"?";

ANOTHER BOOT CIRCUIT - IF 21

Circuit one contains one error. The line going to pin 15 of IC B should go to pin 14. With that change the circuit works fine.

The boot circuit shown in figure two also contains one error. It doesn't work! Furthermore, it can't be made to work (or at least not easily). My apologies to Mike Franklin and anyone else who wasted their time on it. The person responsible for it has been sacked.

......Tony Hill

From Tony Hill's Note Book
- by Tony Hill

NOTES ON NIES MONITOR - VERSION 2

The following are some random notes on Steve Nies monitor, collected after some months of use. They may be of some help to ACE members using his software. Most of them were only possible because I have a dissassembled listing (mostly due to the efforts of Wayne Bowdish) and used it to explain the funny things that happened at times.

Note 1 - Keyboard Input

Since bringing up "The Monitor" I have noticed how poor my keyboard is. Or so I thought. It kept missing characters that I was sure I had typed. The reason? Well it seems that the input routine in "The Monitor" tests the keyboard flag for input and then issues an INP command no matter what the status of the flag. This had the effect of resetting my keyboard, a most nasty habit if you are in the middle of a keystroke. Change the byte at F 61 from 64 to 65 to fix this problem.

Note 2 - Using the ACE VDU

While "The Monitor" is written to work with a 6847 VDU type display, some changes are necessary to make it work with the ACE VDU board. The required changes set the ACE board into alpha-numeric mode, and mask off bit 6 of displayed characters to eliminate block graphic characters. The changes are really patches to the existing software, and are implemented at the price of losing the bell routine.

ADDRESS	DATA
I 20	30 A9
P 7F	C0 I B3
I A8	D5 F8 FF B9 F8 02 59 90
I B0	B9 30 22 CB S 6A 9F FA
I B8	BF C0 P 83

The byte at I AA is the page address of the VDU control register.

Note 3 - Adding Disk Commands

I have a listing of a 1/2 K program to allow the ACE disk board to be run from "The Monitor". When run ,it automatically patches a set of new commands into the command table , allowing such functions as multiple sector reads and writes, seek to track, set sector number, read controller status and master reset drive. Anyone with an ACE controller board and "The Monitor" (or anyone otherwise interested) should drop me a line for a copy.

ACE STANDARD DISK FORMAT

At a meeting of the unofficial ACE standards group on Dec. 29, 1981, a standard format for ACE disks was documented. This standard was set to ensure that work now being done by various people on disk operating systems would result in compatable disks. The standard format should apply to most disks, either floppy (5- 1/4" or 8") or hard.

In establishing this standard, ACE realizes that it will most likely not match any 1802 disk operating system now in use. However, it is necessary to pick something as a standard and so every effort has been made to establish a good general one. The ACE standard is designed to allow room for growth as disk operating systems become more complex, while being usable by the most simple disk operating system.

Initial use of this format will probably be mainly for 8" drives, so a few words about physical standards for 8" floppy disks are necessary. In general, other disks will have different physical formats but the general ACE structure for their layout should apply. ACE has elected to adopt the "standard" IBM soft sector format for their 8" drives. IBM format for single density floppy disks specifies 77 tracks divided up into 26 sectors each. Each sector consists of 128 data bytes plus assorted header information.

In the ACE standard, Track 0 will be used for "boot up" code, bad sector lock-out tables and reserved space for future expansions. Sectors 01 to 08 inclusive are available for user "boot up" code, which will either be part of the operating system or user supplied. The rest of the track is reserved for sector lock out tables if required. Track 1 will contain the disk index of files, formatted as per the ACE standard. The rest of the tracks will contain sequential files, with each file using as many sectors and tracks as necessary. No assumption of file format is made, beyond the fact that the file is continuous over sequential sectors and tracks.

The index of files, more commonly called the directory, consists of a contigious set of 32 byte entries, one per disk file. Each entry contains all information necessary to find, load and if possible run that file as well as various pieces of "book keeping" data. Index entry format is summarized in table 1. Simple systems may not maintain all fields in the entry, but by standardizing the format, any systems should be able to read any other system's index. A suggested minimum set of entries to be maintained is the filename, load address, relative block number and number of sectors used entries.

There is no correspondance required between the order of entries in the directory and the order of files on the disk. However, the first index entry should be a record pointing to the index, which is a continous file itself. This standard for the first entry has been adopted to allow the index to be treated as a file by a disk operating system. Treating the index as a file simplifies the job of index display and data manipulation.

TABLE ONE - INDEX RECORD LAYOUT

BYTE	MEANING	BYTE	MEANING
00	Record Type	10,11	Load Address
01-08	File Name	12,13	Run Address
09-0B	Extension	14-16	Relative Sector Number
0C	Year/Month	17-19	Sectors Allocated
0D	Day	1A-1C	Sectors Used
0E	Version	1D-1F	Relative Sector Number of Extension
0F	Spare (Reserved)		

NOTES

1) File name and extension are usually in ASCII. All other entries are in Hex.
2) Record type is 00 for deleted entries, 01 for standard file records, 02 for extension records and FF for the end-of-directory record.
3) Year/Month is stored as year in the high nibble and month in the low.
4) Relative block number is the number of sectors per track times the track number plus the required sector on that track.
5) Extension records are included for future use if larger directory records are required. Their format is not defined at this time, only the the possibility that they may exist.
6) Year, month and day refer to the date the file was most recently accessed. However the first directory entry will have the creation date of the directory in this space.
7) The spare reserved byte (0F) will be used to designate the operating system the disk was created by. Designation codes will be assigned by ACE in a manner yet to be determined.
8) Relative sector numbers are equal to the physical sector number added to the product of the track number and the number of sectors per track.

FORTH

At the request of the editor, I am hereby submitting a status report on the use of FORTH for club 1802 systems. I apologize in advance for anybody whose name I forget to credit (or blame) .

First of all, the club executive now has a working FORTH system running and more Toronto/Hamilton area members are in the process of bringing it up. If you don't know what FORTH is, this is not the place for me to tell you about it. I would suggest that you beg borrow or steal a copy of the August 1980 issue of BYTE as a good starting point.

Anyways, a little about the history of the current FORTH version is in order. The copy we brought up (after months of work) was sent to us by Ken Mantei of Cal State College in San Bernardino. From what I can tell from his notes, it is the same as the version distributed as the fig-FORTH standard (although I hope that version has the ?STACK code done correctly). The majority of the original programming work was done originally by Gary Bradshaw, with later refinements by Gordon Fleming, Richard Cox and Ken Mantei.

The current status of FORTH as a club project is that we are not sure how to distribute it. Ken Mantei has written a good set of installation instructions, and copies of the source code are available from the FORTH interest group. Also available and recommended is the fig-FORTH installation guide. However, as the many months I spent working on bringing up our version have shown, typing in a 6K program by hand is no thrill. So we need a distribution media of some type, either floppy , cassette or three 2K EPROMs. If you are interested, drop me a line stating your preference for media type, and maybe by next issue we will have ordering information.

An EPROM Programmer for Single +5v Supply EPROMS
- by M. Franklin

In Ipso Facto 21, I presented modification to the Netronics full Math Package board to relocate the Eproms to addresses other than 0000H. For the past 3 months, I have been utilizing the board to house by Monitor and DOS at C000-DFFF. The board has been further modified to burn single +5v supply Eproms (2716's). The following article, schematic and program form the bases for this capability. While the circuits were implimented upon the former Math Board, the circuit and program could be implemented by other means.

Theory of Operation

Single +5v eproms require a +5v pulse to be supplied to pin 18 for 50 ms. when valid address and data are available on the appropriate busses. Pin 21, the program pin, is held to +5v during the programming operation, and dropped to +5v for read operations.

Addresses are supplied sequentally from a 4040 11 bit counter, which is incremented by a port enable pulse following each data byte transfer. The page count is shown by the page count led, which is on for odd number pages.

Data is supplied to this EPROM from an 1852 port which is enabled by OUT PORT 4 enable pulse, which also simultaneoulsy displays the data on the ELF II system data HEX LEDS. Once the data is available to the EPROM, the service request pin ($\overline{SR}$) goes HI and trips the 4538 timer to generate a 51 ms pulse on the EPROM. The duration of the time pulse is determined by the R-C network on pin 14/15 and is quite critical, since manufacturers specified a 50 ms. pulse.

Use precision components, a 5.11 Kohm, 1% resistor and a 10 mf capacitor will produce 51.1 ms. pulse.

Use a good scope to check the length of the pulse and proper sequencing of the two port enable pulses.

Operation

The program first of all determines the length and location of the source to be transfered to the EPROM. If the source is not located at the beginning of the EPROM (000H) the program increments the counter and decrements the address count to 000H to correctly position the address count for the transfer. In order to achieve reliable results, it is good practice to locate the source data at this correct relative address to a 4K boundry (ie. to burn C478 - C500H locate at 1478-1500H and start the program count at 1000H.)

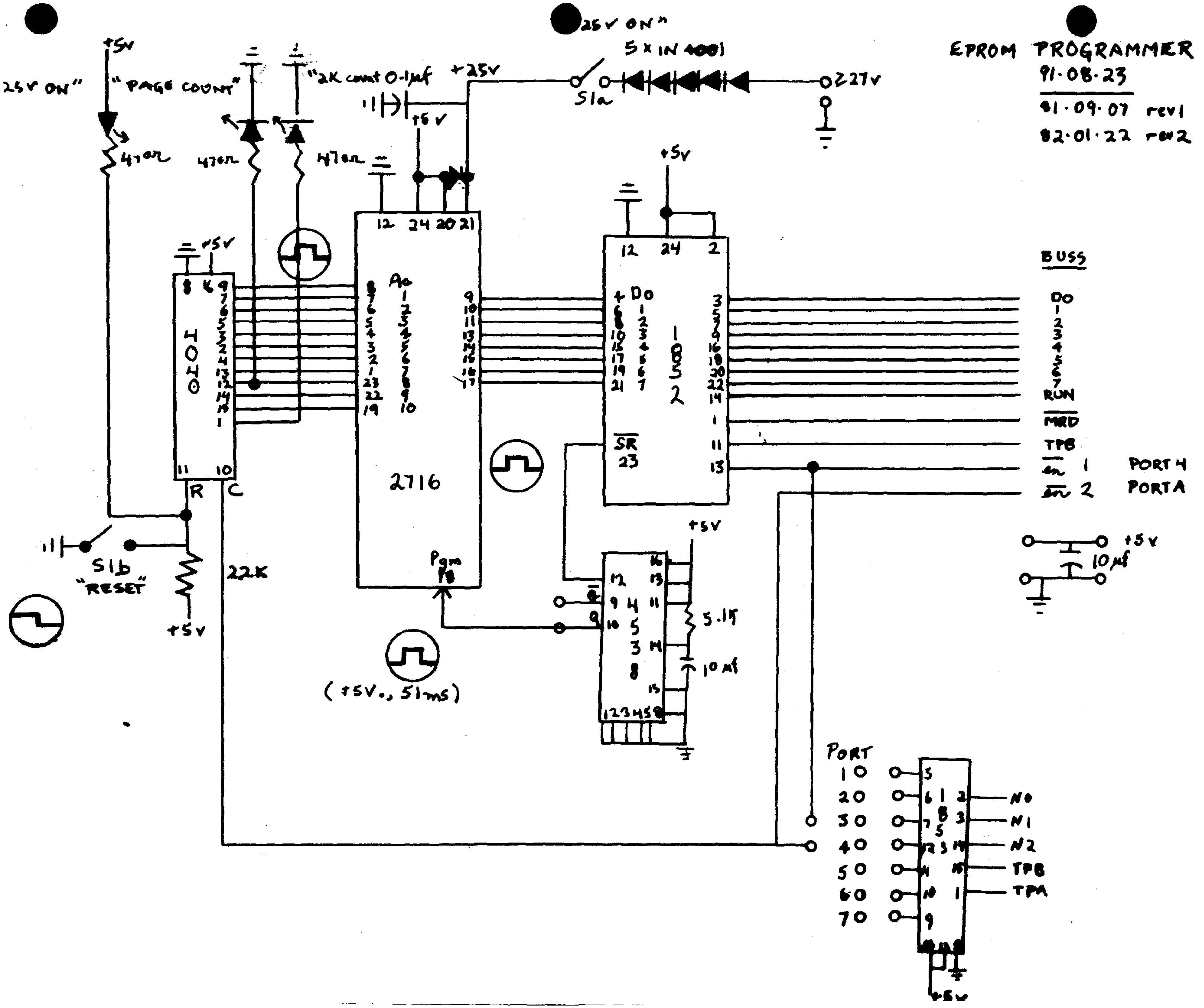
The appropriate values are loaded into R6 and R7 and the EPROM size or program length into R4. Switch on the 25V switch, (S1) which also resets the counter and lights the 25V on LED. Hit the run switch and watch the HEX LEDS display the data. Run time for 2K is 115 seconds with the values specified.

EPROM PROGRAMER - 2716
MEF - 81-09-25

```
0000   F808     LDI #08     INITIALIZE
0002   B4       PHI R4
0003   F800     LDI #00     R4 = Program Burn Length +1
0005   A4       PLO R4
0006   F800     LDI #00     R6 = Source of Data Address
0008   B6       PHI R6           Note = Locate in 2K block in
0009   F800     LDI #00                 correct relative location
000B   A6       PLO R6
000C   96       GHI R6      R7 = 0000H - Offset of Source Relative
000D   FA0F     ANI #0F          to beginning of Eprom at 0000.
000F   B7       PHI R7
0010   86       GLO R6
0011   A7       PLO R7

0012   97       GHI R7      SET COUNTER
0013   3A18     BNZ #18
0015   87       GLO R7      If R7 ≠ 0000
0016   321D     BZ  #1D     DEC R7, inc counter until R7 = 0000
0018   E1       SEX R1         counter set at correct start address
0019   6A       INP P2
001A   27       DEC R7
001B   3012     BR  #12

001D   E6       SEX R6      BURN EPROM
001E   64       OUT P4
001F   F80A     LDI #0A     Output Data via Port 4 from M(R6)
0021   B5       PHI R5      (Displayed on LEDs)
0022   F89F     LDI #9F
0024   A5       PLO R5      Delay 55 MS for Program Pulse.
0025   25       DEC R5      Count in R5
0026   95       GHI R5
0027   3A25     BNZ #25     DEC count, Test if R4 = 0000
0029   24       DEC R4      if so, quit.
002A   94       GHI R4      if not, inc. counter
002B   3A30     BNZ #30     loop till done.
002D   84       GLO R4
002E   3234     BZ  #34
0030   E1       SEX R1
0031   6A       INP P2
0032   301D     BR  #1D

0034   7B       SEQ         'Q' on, quit!
0035   3034     BR  #34

DONE!                       Run Time - 135 sec. (2K)
```

Addendae

For those who do not have the ports decoded, such as on the Netronics Giant Board, the schematic includes a port decoder circuit using a 1853 as an alternative. Port assignments may be changed by altering the enable wiring and changing the appropriate program addressing.



EPROM PROGRAMMER
91.08.23
91.09.07 rev1
82.01.22 rev2

MODS TO NETRONICS FULL BASIC PART II
- by M. Franklin, 690 Laurier Avenue, Milton, Ontario, L9T 4R5

In IPSO FACTO 17, p. 29 and Defacto III - 95, I wrote about modification to the cassette version of Netronics Full Basis Level III. While my version worked, apparently it didn't for others. I recently had the opportunity to convert an EPROM/ROM version and found the problem - the error messages were incorrect.

The following changes work on both cassette and EPROM versions. Note - you cannot EPROM the cassette version since the RAM start and cassette load addresses will be incorrect.

1. CHANGE TO 600 BAUD

```
M (1469) - FF 08 52 12
M (1332) 00
```

NOTE: will not work with Netronics VID board which is programmed for 300 baud only.

2. Fixed line length (64 characters) and abbreviated sign on

```
M(147E)1A
M(1480)F8 15 BE D7 36 F8 37(18 for 32 characters)5D E3
M(1489)65 2F D4 13 BB D4 14 97 D8 20 09 C0 00 27
```

Change M (1497-AC) and M (14E0-1519) to C4

```
M (151A)-4E 45 54 52 4F 4E 49 43 53 20 42 41 53 49 43 - "NETRONICS
         0A 0D 52 45 41 44 59 0A 0D 00                    BASIC READY"
```

3. SHORTENED COMMAND INPUT
   - the following changes allow programmer to abbreviate command with a "." (period) i.e. PRINT becomes P.; LOAD = LO.;LIST LI.;etc.

```
M (02B9) = 4E
M (01B7) = C9, C4

M (02 C1) = 4E FF 40 33 C1 1E 30 B3 1A 0A FF 2E
            C2 03 E0
M (03E0) = 4E FF 40 33 E0 2E 1A
```

4. ADD MONITOR JUMP TO COMMAND TABLE
   - with shorter commands PR is no longer needed.
   - move M (02E2-03DE) to (02E3 - 03DF) - shift 1 byte.
   - insert M (02E2) - 4D 4F 4E F0 00.
     "MON F0 00" or appropriate address
     (change M (0000) to C4 C4 C4.

5. LIST/RELIST CORRECTION
   - to correct the programs problems with line numbers ending in 0D (13), change the routine as follows:

```
M (0414) - D8 5C 6B 1D 4B 5D 1D 0B 5D D4 01 2B
           FF 01 32 42 FF 01 32 30

M (0428) - to C4 in EPROM version
           to 1B in cassette version
```

NEW HIGH SPEED CMOS SERIES

- ref: Electronics December 1, 1981, p137-140

Several manufacturers have begun distribution of a 74HC series of CMOS chips offering superior speed to 74C and 4000B Series chips, yet retaining the low power and large fan out capacity of CMOS.

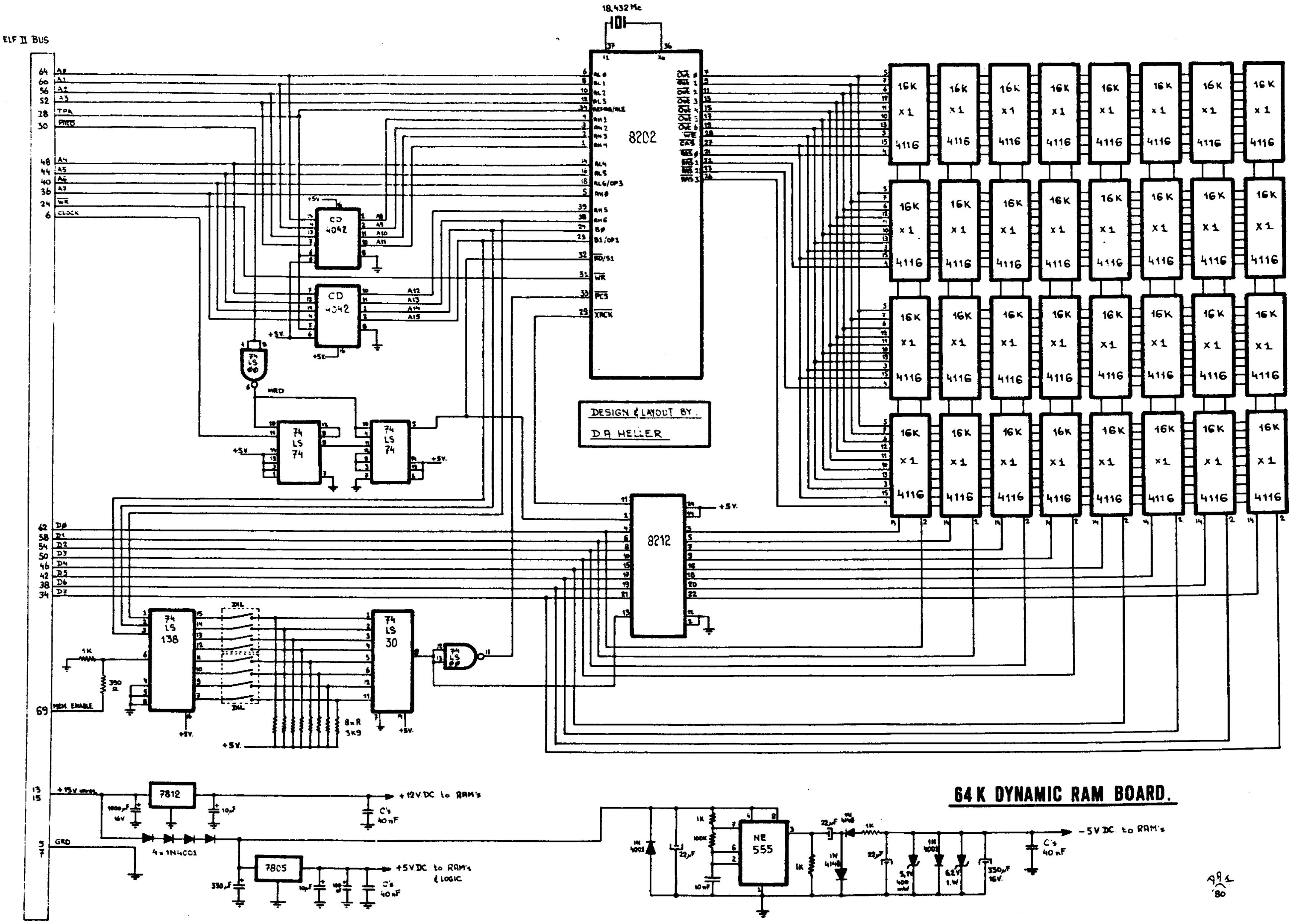
From a club point of view, the introduction of CMOS octal Uini and bi directional buss buffers is welcomed (74HC 243 and HC245). The chips are pin for pin compatible with standard 74 series chips. Approximately 100 devices will be available from National Semiconductor Corps, Mitel and Motorola.

74HC, 74C and 4000 B chips are input and output compatible as long as they share a common power supply.

64k Dynamic Board
- by D. Heller, V.H. Goedhartin, 317 1181 VN Amstelveen, the Netherlands

I BOUGHT MY ELF II ABOUT 3 YEARS AGO, AND FOR MEM. EXPANSION ONLY
4K STATIC BOARDS ARE AVAILABLE, WHICH I FIND TOO LITTLE (MAX. OF 16K
PLACE FOR THE ELF II)
ALSO, I DON'T LIKE THE BIG POWER SUPPLY NEEDED THEREFORE!
THAT'S THE REASON WHY I HAVE DEVELOPED A 64K DYNAMIC RAM BOARD.
ENCLOSED I SEND YOU THE SCHEME FOR IT.
FOR THIS PROJECT I HAVE USED INTELS POWERFULL 8202 DYNAMIC RAM CONTROLLER.
THE WHOLE SYSTEM WORKS WHIT EVERY 1802 CLOKFREQUENCY, AND ONLY NEEDS
A POS. 15 VOLT UNSTABILIZED POWER.
THE MAX. CURRENT IS UNDER THE 1 AMPERE FOR THE WHOLE SYSTEM.
(ELF II + 64K DYN. RAM BOARD + GIANTBOARD)
THERE ARE MANY 1802 USERS IN THE NETHERLANDS WHICH HAVE THIS BOARD.
MY OWN ELF II WITH DYN. RAM BOARD IS CONTINUOUSLY POWERED ON AND IS
LOADED WITH SEVERAL PROGRAMS IN QUEST-BASIC, AND THIS SYSTEM HAS
BEEN WORKING ABOUT 2 YEARS WITHOUT ANY PROBLEMS.
I HAVE MY PROTOTYPE BOARD WIRE-WRAPPED AND THIS WORKS WELL, BUT MANY
1802 USERS ASKED ME FOR A PRINT, SO I HAVE DEVELOPED A P.C. BOARD.
(DOUBLED SIDED PLATED THROUGH AND GOLDPLATED CONNECTORFINGERS.)
THIS PRINT COMPLETLY ASSEMBLED AND TESTED WITH 16K RAM IS AVAILABLE FOR
THE 1802 USERS.
MORE MEM. NEEDED? SIMPLY BUY 24 X 4116'S AND PLACE THEM IN SOCKETS ON
BOARD AND YOU HAVE 64K RAM AVAILABLE!
MEM. ON BOARD IS BANK SELECTABLE IN BLOCKS OF 8K.
IF YOU HAVE PROM'S OR EPROM'S IN YOUR SYSTEM, (NETRONICS MONITOR PROM ON
GIANTBOARD) AND YOU HAVE ADDRESSDECODED MEM. ENABLE SIGNALS, SIMPLY
CONNECT IT TO THE DYNAMIC RAM BOARD AND YOU HAVE NO BUSPROBLEMS
BETWEEN THE DYNAMIC RAM DATA AND PROM AND/OR EPROM DATA.
IF THERE ARE PEOPLE INTERRESTED: A P.C. BOARD COST $50 AND A COMPLETLY
ASSEMBLED AND TESTED BOARD (ALL IC'S ON SOCKED'S) WITH 16K RAM INSERTED
WITH MANUAL IS AVAILABLE FOR $250.
THEN I HAVE ONE QUESTION: I LIKE TO PLAY CHESS, BUT I HAVE NEVER HEARD
ABOUT PROGRAM'S FOR THIS GAME FOR 1802 MICRO'S.
ARE THERE CHESS PROGRAM'S AVAILABLE?

DOUWE HELLER
V.H.GOEDHARTLN.317
1181 KN AMSTELVEEN
THE NETHERLANDS



64 K DYNAMIC RAM BOARD.

Tiny Pilot Terminal Tester Program
- by T. Jones, Enterprise, Alba U.S.A. 36330

```
PILOT1.2

EDIT
#WI 25
T:TERMINAL TEST
T:HOW MANY CHARACTERS/LINE?
T:
A:#C
%0 T:ENTER TEST NUMBER.
T:  1.SLIDING ASCII PATTERN.
T:  2.VERTICAL EDGE SHADING .
T:  3. HORIZ. EDGE SHADING.
T:  4.HAMMER ALIGNMENT.
T:  5.DOT MATRIX MISSING WIRE
T:  6.RANDON ASCII.
T:  7.ASR 33/35 PRINT HEAD.
T:  8.ASR DASH-POT TEST.
T:  9.BAUDOT CHAR. SET.
T: 10.ECHO INPUT LINE.
T: 11.LINE FEED.
T: 12.FORM FEED.
T: 13.ALL TESTS FROM 1-6.
T:
A:#Q
X:Q>13
JY:500
T:
T:HOW MANY TEST CYCLES?
A:#L
%200 U:Q
C:L=L-1
X:L=0
JN:200
T:
T:END OF TEST
T:ANOTHER TEST?
A:
M:YES,Y
JY:0
E:
    %1   T:
         C:P=32
         C:E=127
    %104 C:N=C
    %103 K:P
         C:P=P+1
         C:N=N-1
DONE?    X:P>E
         RY:
         X:N=0
         JN:103
         T:
         J:104
    %2   C:B=69
```

```
      %100 C:A=C
      %105 K:B
           C:A=A-1
           X:A=0
           JN:105
           T:
           R:
      %3   C:B=77
           J:100
      %4   C:B=45
           J:100
      %5   C:B=47
           J:100
      %6   C:A=C
      %106 Z:95
           K:Z+32
           C:A=A-1
           X:A=0
           JN:106
           R:
      %7   C:U=85
           C:S=42
           C:A=C
      %107 K:U
           K:S
           C:A=A-2
           X:A=0
           JN:107
           R:
      %8   C:R=30
           C:T=R
CR         K:14
      %125 K:32
           C:T=T-1
           X:T=0
           JN:125
           C:T=R
           K:14
      %130 K:63
           C:T=T-1
           X:T=0
           JN:130
LF         K:10
           R:
      %9   T:
           T:ABCDEFGHIJKLMNOPQRSTUVWXYZ
           T:-?  3!8 8'().,9014 57:216"
           R:
      %10  A:$D
           T:$D
           R:
      %11  C:N=0
      %119 T:#N
           C:N=N+1
           X:N=10
           JN:119
           R:
      %12  C:N=15
           T:FORM FEED
           K:12
      %112 K:0
```

```
     C:N=N-1
     X:N=0
     JN:112
     R:
%13  C:G=1
%115 U:G
     C:G=G+1
     X:G>6
     JN:115
     R:
%500 T:
     T:TRY AGAIN!!
     J:0
```

```
#M
LLL
```

```
TERMINAL TEST
HOW MANY CHARACTERS/LINE?

?72
ENTER TEST NUMBER.
  1.SLIDING ASCII PATTERN.
  2.VERTICAL EDGE SHADING .
  3. HORIZ. EDGE SHADING.
  4.HAMMER ALIGNMENT.
  5.DOT MATRIX MISSING WIRE
  6.RANDON ASCII.
  7.ASR 33/35 PRINT HEAD.
  8.ASR DASH-POT TEST.
  9.BAUDOT CHAR. SET.
 10.ECHO INPUT LINE.
 11.LINE FEED.
 12.FORM FEED.
 13.ALL TESTS FROM 1-6.

?13

HOW MANY TEST CYCLES?
?2

 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefg
hijklmnopqrstuvwxyz{|}~
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
------------------------------------------------------------------------
////////////////////////////////////////////////////////////////////////
FlYQS)()=[8Q@~;VK4fjUIshn]Y4IO^Z5J3ceL6Lxq"%G-;WOA >]X()<v+6jreiTFlYQ$)()=[
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefg
hijklmnopqrstuvwxyz{|}~
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
------------------------------------------------------------------------
////////////////////////////////////////////////////////////////////////
8Q@~;VK4fjUIshn]Y4IO^Z5J3ceL6Lxq"%G-;WOA >]X()<v+6jreiTFlYQ$)()=[8Q@~;VK4f
END OF TEST
ANOTHER TEST?
?YES
ENTER TEST NUMBER.

?7

HOW MANY TEST CYCLES?
?1
U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*U*
END OF TEST
ANOTHER TEST?
?Y
ENTER TEST NUMBER.
?9

HOW MANY TEST CYCLES?
?1

ABCDEFGHIJKLMNOPQRSTUVWXYZ
-?  3!8 8'().,9014 57
END OF TEST
```

Kaleidoscope and Life Program for the 1802/1861
- by J. Munck, 20228 Clark St., Woodlands, CA., U.S.A. 91367

This program generates a kaleidoscopic pattern of pixels on a(one)page 1861 display. The image is symmetrical about the center. The X and Y values for the plotter are generated by a pseudo-random number generator of only 12 bytes. The PNG, by Lester Hands, (from a Questdata newsletter)spews out a string of over 32000 15 bit numbers that are non-repetitive.

A register, F, holds the entire number. The register is split in half to form 8 bits for each of the x and y components. The random values are further manipulated to form:
(x,y),(y,x),(-x,y),(y,-x),(x,-y ,(-y,x),(-x,-y),(-y,-x).
Eight sets of coordinates are then presented to the plotter(0056). There is no significance to eight sets. Any more only seemed to make the display too busy. After plotting the eight dots the PNG is aroused again and eight more pairs are generated.

The basic method of addressing the screen uses the formula:
(X+32/8)-(Y+16)(8)+Screen size + Screen offset.
This determines the byte location. The pixel(bit) location is determined from the original X coordinate using a table look-up.

The speed is fast, and a pleasing display is formed especially when the display is square. This can be done from the hex keypad, using a value of OF or O7. The plot routine was adjusted to blank out existing bits when the MSB of the number in RF.1=1.
The display then twinkles enough to make either the VIP or the Studio II patterns proud.

When the input switch (EF4) is depressed all motion ceases, and when released life begins in an 48 * 30 universe.

The program that generates life was also obtained from a Questdata article (vol 2, #4). The life pgm is essentially that published by Ray Tully with two exceptions. I did not use the suggested input routine to generate the initial life patterns, but instead chose to use my random pattern as the starting point.

In addition, a frequently called sub-routine of six bytes(test, inc) (D3 ,3B 50,1B,30 50) was unraveled to substitute in-line code to see if life could be speeded up some. This proved to be the case, as approximately 0.5 seconds was shaved off the 1.8 seconds.

The choice to use the random pattern resulted ,of course, in an unpredictable input to life. As life evolved the symmetry was retained. One might imagine the display to be snowflakes,. starbursts, cellular division, or to the jaded purist, only an 1861 turning bits on and off.

To freeze the life forms, press the input switch. To exit life release the switch and the program will clear the screen and begin some more kaleidoscope plotting.(For this, try an input of OF).
One byte, in the program, may prove to be of interest to the compulsive tweaker: (OOF5), if increased from 03 to 04 or 05 allows more neighbors, resulting in a super-nova. Execute at 0000.

```
;         KALEIDOSCOPIC LIFE  - 1861 GRAPHICS PROGRAM    1-30-82

;                    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
                     ...............................................

;         0000      90 B3 B4 B8 BC F8 03 B9 F8 09 A6 A9 F8 D4 AC F8
;         0010      E4 AD F8 FF A2 F8 DC A1 F8 11 A7 F8 56 A4 F8 7F
;         0020      A3 F8 01 B1 B2 BD A5 F8 02 B5 B6 B7 BF 5D D3 XX
;         0030      XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
;         0040      XX XX XX XX XX XX XX XX XX XX XX XX XX 80 40 20
;         0050      10 08 04 02 01 D3 A8 FC 20 F6 F6 F6 F9 F8 52 8A
;         0060      FC 10 FE FE FE F5 AA 88 FA 07 FC 4D A8 9F FE 08
;         0070      33 77 52 0A F1 30 7C FB FF 52 0A F2 5A 30 55 EA
;         0080      F8 03 BA F8 FF AA F8 00 73 9A FB 01 3A 86 1A E2
;         0090      69 9F FE 52 FE F3 FE 8F 7E AF 9F 7E BF 6C 9F F2
;         00A0      BB FD 00 BE 8F F2 AB AA FD 00 AE 9B D4 9B AA 8B
;         00B0      D4 8B AA 9E D4 9E AA 8B D4 8E AA 9B D4 9B AA 8E
;         00C0      D4 8E AA 9E D4 9E AA 8E D4 3F 91 37 CB F8 00 AB
;         00D0      C0 01 00 D3 3F DD 37 D6 F8 00 A0 B0 D0 8B FB 02
;         00E0      32 FC 8B FF 02 33 F3 F8 00 F6 09 7E 59 7A F8 00
;         00F0      AB 30 D3 8B 7D 03 3B E7 F8 FF 30 E9 31 F8 30 E7
                     ...............................................
;         0100      05 FE 3B 05 1B FE 3B 09 1B 07 FE 3B 0E 1B FE 3B
;         0110      12 1B 25 05 F6 3B 18 1B 15 26 06 F6 3B 1F 1B 16
;         0120      27 07 F6 3B 26 1B 17 06 FE 3B 2C 7B FE 3B 30 1B
;         0130      DC F8 06 A8 05 AA 06 BA 07 BB 8A FE 3B 3F 1B FE
;         0140      3B 43 1B FE 3B 47 1B 9B FE 3B 4C 1B FE 3B 50 1B
;         0150      FE 3B 54 1B 9A FE 3B 59 1B FE 3B 5D 7B FE 3B 61
;         0160      1B DC 8A FE AA 9A FE BA 9B FE BB 28 88 3A 3A 05
;         0170      F6 3B 74 1B F6 3B 78 1B 07 F6 3B 7D 1B F6 3B 81
;         0180      1B 06 F6 3B 86 7B F6 3B 8A 1B 15 16 17 05 FE 3B
;         0190      92 1B 06 FE 3B 97 1B 07 FE 3B 9C 1B DC 19 86 FE
;         01A0      FE FE FE FE FB E0 3A 00 15 15 16 16 17 17 19 19
;         01B0      86 FB F9 3A 00 F8 01 A5 F8 09 A6 A9 F8 11 A7 0D
;         01C0      FB 02 3A CF F8 03 B5 B6 B7 5D F8 02 B9 30 00 F8
;         01D0      02 B5 B6 B7 5D F8 03 B9 30 00 72 70 22 78 22 52
;         01E0      C4 C4 C4 F8 02 B0 F8 00 A0 80 E2 E2 20 A0 E2 20
;         01F0      A0 E2 20 A0 3C E9 30 DA XX XX XX XX XX SS SS SS
;                    -----------------------------------------------
```

The program requires 1 K of memory. Program 2 pages, display 2p. LOCATION 0090: NETRONICS=69, QUEST=61. INPUT ON EF4 + HEX KEYPAD. Please pardon the absence of a listing. Watch those eights and Bees.

SS=STACK, XX=DONT CARE.

END

Telephone Dialing Program

by: (Author unknown - please write us for credit)

****
*EDITOR'S NOTE- This article is presented for interest only and *
* it's application may in fact be illegal in your area. Check with *
* your telephone company before connecting to their system !! *
****

The software/hardware presented here enables your 1802 computer to dial a telephone number for you. This was created because of a television show called "Trivia" on our local cable TV system. The moderator asks a series of questions and people phone in and try to answer them. The callers with the correct answers win prizes. This type of operation jams up the phone system, so that many repeated tries are required before getting a connection to the TV station. With only rotary dial phones in our house this becomes a pain in the index finger for a one hour show. My wife got after me about an automatic dialer. This will dial up to a fourteen digit number (3 digits for a dial access code, 3 digits for an area code and 7 digits for the phone number) or more if required for some reason.

The dialer program is designed for my particular application - dialing a repetitive phone number. This program may also function as a subroutine of a larger communications program, with the addition of features such as ring/dial-tone/busy-signal detect, auto hang-up, redial and electronic directory.

The program, as written, utilizes an IN switch to initiate, a relay to pulse the red wire of the telephone line, and two hex digits to indicate the dialed digits. Specific hardware assignments may be changed to suit your system with the appropriate software changes. A drawing of the hardware required to implement the system is shown in figure 1.

The program is organized as a series of two nested counting loops and one subroutine timing loop. The inside loop is for counting the number of dialing pulses for each digit and the outside loop counts the number of digits to be dialed. The subroutine timing loop is for delays in the dialing operation, the pulse width, interpulse spacing and the interdigit spacing.

The number of digits to be dialed is stored as a hex digit at M(34) and the digits to be dialed are stored starting at M(35). One digit is stored in each byte , with 01 to 09 HEX representing the digits 1 - 9 and 0A HEX representing 0. For example-

1-203-555-1212 = 0B,01,02,0A,03,05,05,05,01,02,01,02

starting at memory address 34 HEX.

The program is entered at M(0000) with R(0) or R(3) as the program counter. After removing the telephone handset and getting a dial tone, press the IN key to initiate dialing. The phone number is dialed once and the program returns to wait for the IN key to be pressed again for a redial. The digits are displayed in the HEX LED display when they are dialed.

The range of delay times shown in Figure 2 are what worked with my telephone company's equipment, and may need to be adjusted to fit your companies requirements. This is , by the way, a replacement for a rotary dial phone, not for dual-tone frequency dialing. Remember also to check the telephone companies regulations regarding user connected equipment (and any required inspection and approval) before use.

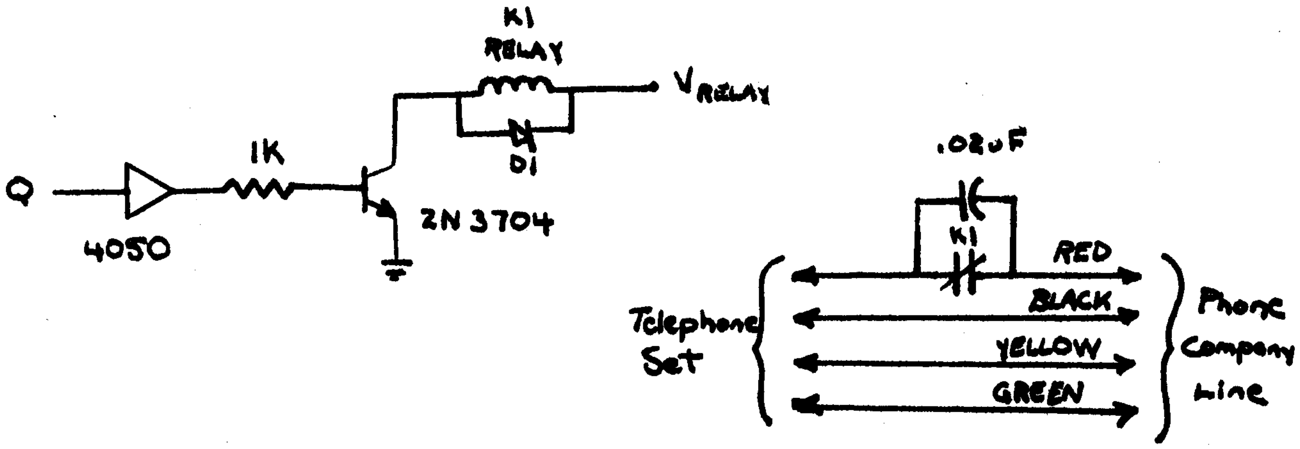


FIGURE 1 - HARDWARE REQUIREMENTS

R(2)	Stack Pointer
R(3)	Program Counter
R(9)	Number of Digit Counter
R(A)	Dialed Digits Counter
R(B)	Subroutine Program Counter
R(C)	Delay Loop Counter

Pulse Time = (SDLY) = 10 (slow) to 04 (fast)
Digit Time = (LDLY) = 80 (slow) to 10 (fast)

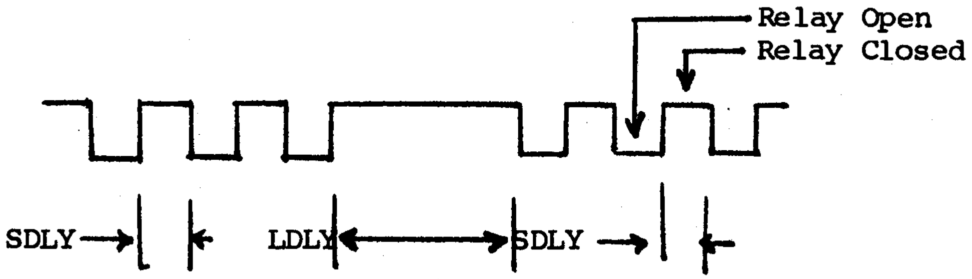


FIGURE TWO - REGISTER USAGE AND TIME CONSTANTS

```
                            ;*******************************************
                            ;*                                         *
                            ;*    TELEPHONE DIALING PROGRAM            *
                            ;*                                         *
                            ;*******************************************

            SDLY:           .EQL    #10     ; SHORT DELAY CONSTANT
            LDLY:           .EQL    #80     ; LONG DELAY CONSTANT

0000  90                    GHI     R0      ; INITIALIZE REGISTERS
0001  B2                    PHI     R2      ; DATA POINTER
0002  B3                    PHI     R3      ; PROGRAM COUNTER
0003  BB                    PHI     RB      ; SUBROUTINE PROGRAM COUNTER
0004  E2                    SEX     R2      ; DATA POINTER
0005  F8 09                 LDI     MAIN    ; INITIALIZE PC
0007  A3                    PLO     R3      ;
0008  D3                    SEP     R3      ; JUMP TO START

0009  F8 29          MAIN:  LDI     DELAY   ; SUBROUTINE PROGRAM COUNTER
000B  AB                    PLO     RB      ;
000C  3F 0C         CHECK:  BN4     @       ; WAIT FOR INPUT KEY
000E  F8 34                 LDI     COUNT   ; SET X -> # OF DIGITS
0010  A2                    PLO     R2      ;
0011  72                    LDXA            ; GET DIGIT COUNT AND
0012  A9                    PLO     R9      ;     LEAVE R(9) AT FIRST DIGIT
0013  F0              REP:  LDX             ; LOAD DIGIT INTO R(A)
0014  AA                    PLO     RA      ;
0015  64                    OUT     4       ; DISPLAY IT
0016  7B            AGAIN:  SEQ             ; OPEN RELAY
0017  DB                    SEP     RB      ; CALL DELAY SUBROUTINE
0018  10                    .BYTE   SDLY    ; DELAY LENGTH DATA
0019  7A                    REQ             ; CLOSE RELAY
001A  DB                    SEP     RB      ; CALL DELAY SUBROUTINE
001B  10                    .BYTE   SDLY    ; DELAY LENGTH DATA
001C  2A                    DEC     RA      ; DECREMENT DIGIT
001D  8A                    GLO     RA      ; GET DIGIT
001E  3A 16                 BNZ     AGAIN   ; DONE YET?
0020  DB                    SEP     RB      ; CALL DELAY SUBROUTINE
0021  80                    .BYTE   LDLY    ; DELAY LENGTH DATA
0022  29                    DEC     R9      ; DECREMENT # OF DIGITS
0023  89                    GLO     R9      ; GET NUMBER OF DIGITS LEFT
0024  3A 13                 BNZ     REP     ; REPEAT IF NOT DONE
0026  30 0C                 BR      CHECK   ; IF DONE, WAIT FOR IN SWITCH AGAIN
                            ;
                            ;  DELAY SUBROUTINE
                            ;
0028  D3            DERET:  SEP     R3      ; RETURN TO MAIN PROGRAM
0029  43            DELAY:  LDA     R3      ; GET DELAY CONSTANT
002A  BC                    PHI     RC      ; PUT INTO R(C).1
002B  F8 FF                 LDI     #FF     ;
002D  AC                    PLO     RC      ;
002E  2C             REDE:  DEC     RC      ; LOOP UNTIL R(C)=00FF
002F  9C                    GHI     RC      ;
0030  3A 2E                 BNZ     REDE    ;
0032  30 28                 BR      DERET   ; RETURN WHEN DONE
                            ;
                            ;  TELEPHONE NUMBER DATA TABLE
                            ;
0034  00                    COUNT:      .BYTE   #00     ; # OF DIGITS TO DIAL
0035  =    00 0E            DIGITS:     .BLOCK  14      ; DIGITS TO BE DIALED
```

NEW MNEMONICS FOR THE NETRONICS ASSEMBLER

- by David W. Schuler, 3032 Avon Road, Bethlehem, Pa. 18017,
U.S.A.

The idea of a CALL or RETURN instruction in the Netronics assembler is only a dream to some people. With a simple patch, it is possible to add both a CALL and RETURN instruction to eliminate the sequence:

```
SEP CALL; ,A(SUBRTN)      / CALL=R4
                or
SEP RETURN                / RETURN=R5
```

Having to type both of these instructions is a waste of time and is very prone to errors.

DATA TABLE

The Netronics assembler has its data table loaded from HØ9ØØ to HØB12. (See figure 1 for an ASCII listing) Upon disassembling this table, each entry has the form:

```
[mnemonic] [cr] [opcode] [extra_byte_count]
```

for example, the DEC instruction takes the form

```
DEC [cr] 2Ø Ø1    /in hex: #44 45 43 ØD 2Ø Ø1
```

The key to each entry is the EXTRA_BYTE_COUNT byte. This byte tells the assembler how many bytes to add to the object file.

ØØ	- no bytes	- ex.	SEQ
Ø1	- low order only	- ex.	DEC
Ø2	- one byte	- ex.	BR
Ø6	- two bytes	- ex.	LBR

CHANGES

In order to add a CALL and RETURN instruction, some space must be freed-up in the data table. Fortunately, RCA has more than one mnemonic for a few instructions. The space is gained by deleting the entries for BL, BM, and NBR. These instructions are still accessed by using BNF, BNF, and SKP respectively. The table is then compressed to take out the spaces that were created when the routines were deleted. At the new end, the CALL and RETURN entries are added, which will again fill up the table completely. (Listing 2 shows a complete listing of the final table (HØ9ØØ to HØB12)).

NEW FORM

The new form for the CALL and RETURN instructions is:

```
CALL SUBRTN ..which will assemble as D4 __ __
          and
RETURN ..which will assemble as D5
```

No punctuation is required with this form, which reduces the effects of MURPHY'S LAW since there are fewer characters to type. In a long source file, these changes can take off a substantial number of bytes. Each new CALL instruction takes 9 fewer source bytes and each RETURN takes 4 fewer source bytes.

CONCLUSION

The above changes are only applicable to people who are using SCRT (R4=CALL, R5=RETURN), but the theory behind the changes can be used in almost any program for frequently used sequences. If anyone has any questions or comments on the above changes or the concept, please write to me at the above address and I will try to help you out.

Listing 1
(Original version)

```
MONITOR V4.1
*ASCII DUMP
START:0900
STOP: 0B12
INC
0906 DEC
090C GLO
0912 GHI
0918 PLO
091E PHI
0924 LDA
092A STR
0930 LDX
0936 ORI
093C AND
0942 XOR
0948 ADD
094E SDI
0954 SHR
095A SMI
0960 LDI
0966 OR
096B ANI
0971 XRI
0977 ADI
097D SD
0982 SM
0987 BR
098C BZ
0991 BDF
0997 BPZ
099D BGE
09A3 B1
09A8 B2
09AD B3
09B2 B4
09B7 SKP
09BD BNZ
09C3 BNF
09C9 BM
09CE BL
09D3 BN1
09D9 BN2
09DF BN3
09E5 BN4
09EB IDL
09F1 SEP
09F7 SEX
09FD RET
0A03 DIS
0A09 SAV
0A0F OUT
0A15 INP
0A1B LDN
0A21 BQ
0A26 BNQ
0A2C IRX
0A32 LDXA
0A39 STXD
0A40 ADC
0A46 SDB
0A4C SHRC
0A53 RSHR
0A5A SMB
0A60 MARK
0A67 REQ
0A6D SEQ
0A73 ADCI
0A7A SDBI
0A81 SHLC
0A88 RSHL
0A8F LSDF
0A96 SHL
0A9C SMBI
0AA3 LBR
0AA9 LBQ
0AAF LBZ
0AB5 LBDF
0ABC NOP
0AC2 LSNQ
0AC9 LSNZ
0AD0 LSNF
0AD7 NLBR
0ADE LSKP
0AE5 LBNQ
0AEC LBNZ
0AF3 LBNF
0AFA LSIE
0B01 LSQ
0B07 LSZ
0B0D NBR
```

```
MONITOR V4.1
*HEX DUMP
START:0900
STOP: 0B12
0900 494E 430D 1001 4445 430D 2001 474C 4F0D;
0910 8001 4748 490D 9001 504C 4F0D A001 5048;
0920 490D B001 4CA4 410D 4001 5354 520D 5001;
0930 4CA4 580D F000 4F52 490D F903 414E 440D;
0940 F200 584F 520D F300 4144 440D F400 5344;
0950 490D FD03 5348 520D F600 534D 490D FF03;
0960 4CA4 490D F803 4F52 0DF1 0041 4E49 0DFA;
0970 0358 5249 0DFB 0341 4449 0DFC 0353 440D;
0980 F500 534D 0DF7 0042 520D 3002 425A 0D32;
0990 0242 4446 0D33 0242 505A 0D33 0242 4745;
09A0 0D33 0242 310D 3402 4232 0D35 0242 330D;
09B0 3602 4234 0D37 0253 4B50 0D38 0042 4E5A;
09C0 0D3A 0242 4E46 0D3B 0242 4D0D 3B02 424C;
09D0 0D3B 0242 4E31 0D3C 0242 4E32 0D3D 0242;
09E0 4E33 0D3E 0242 4E34 0D3F 0249 444C 0D00;
09F0 0053 4550 0DD0 0153 4558 0DE0 0152 4554;
0A00 0D70 0044 4953 0D71 0053 4156 0D78 004F;
0A10 5554 0D60 0449 4E50 0D60 054C 444E 0D00;
0A20 0142 510D 3102 424E 510D 3902 4952 580D;
0A30 6000 4C44 5841 0D72 0053 5458 440D 7300;
0A40 4144 430D 7400 5344 420D 7500 5348 5243;
0A50 0D76 0052 5348 520D 7600 534D 420D 7700;
0A60 4D41 524B 0D79 0052 4551 0D7A 0053 4551;
0A70 0D7B 0041 4443 490D 7C03 5344 4249 0D7D;
0A80 0353 484C 430D 7E00 5253 484C 0D7E 004C;
0A90 5344 460D CF00 5348 400D FE00 534D 4249;
0AA0 0D7F 034C 4252 0DC0 064C 4251 0DC1 064C;
0AB0 425A 0DC2 064C 4244 460D C306 4E4F 500D;
0AC0 C400 4C53 4E51 0DC5 004C 534E 5A0D C600;
0AD0 4C53 4E46 0DC7 004E 4C42 520D C806 4C53;
0AE0 4B50 0DC8 004C 424E 510D C906 4C42 4E5A;
0AF0 0DCA 064C 424E 460D CB06 4C53 4945 0DCC;
0B00 004C 5351 0DCD 004C 535A 0DCE 004E 4252;
0B10 0D38 02;
*
```

Listing 1

```
*ASCII DUMP
START:Ø9ØØ
STOP: ØB12
Ø9ØØ INC
Ø9Ø6 DEC
Ø9ØC GLO
Ø912 GHI
Ø918 PLO
Ø91E PHI
Ø924 LDA
Ø92A STR
Ø93Ø LDX
Ø936 ORI
Ø93C AND
Ø942 XOR
Ø948 ADD
Ø94E SDI
Ø954 SHR
Ø95A SMI
Ø96Ø LDI
Ø966 OR
Ø96B ANI
Ø971 XRI
Ø977 ADI
Ø97D SD
Ø982 SM
Ø987 BR
Ø98C BZ
Ø991 BDF
Ø997 BPZ
Ø99D BGE
Ø9A3 B1
Ø9A8 B2
Ø9AD B3
Ø9B2 B4
Ø9B7 SKP
Ø9BD BNZ
Ø9C3 BNF
Ø9C9 BN1
Ø9CF BN2
Ø9D5 BN3
Ø9DB BN4
Ø9E1 IDL
Ø9E7 SEP
Ø9ED SEX
Ø9F3 RET
Ø9F9 DIS
Ø9FF SAV
ØAØ5 OUT
ØAØB INP
ØA11 LDN
ØA17 BQ
ØA1C BNQ
ØA22 IRX
ØA28 LDXA
ØA2F STXD
ØA36 ADC
ØA3C SDB
ØA42 SHRC
ØA49 RSHR
ØA5Ø SMB
ØA56 MARK
ØA5D REQ
ØA63 SEQ
ØA69 ADCI
ØA7Ø SDBI
ØA77 SHLC
ØA7E RSHL
ØA85 LSDF
ØA8C SHL
ØA92 SMBI
ØA99 LBR
ØA9F LBQ
ØAA5 LBZ
ØAAB LBDF
ØAB2 NOP
ØAB8 LSNQ
ØABF LSNZ
ØAC6 LSNF
ØACD NLBR
ØAD4 LSKP
ØADB LBNQ
ØAE2 LBNZ
ØAE9 LBNF
ØAFØ LSIE
ØAF7 LSQ
ØAFD LSZ
ØBØ3 CALL
ØBØA RETURN
```

```
MONITOR V4.1
*HEX DUMP
START:Ø9ØØ
STOP: ØB12
Ø9ØØ 494E 43ØD 1ØØ1 4445 43ØD 2ØØ1 474C 4FØD;
Ø91Ø 8ØØ1 4748 49ØD 9ØØ1 5Ø4C 4FØD AØØ1 5Ø48;
Ø92Ø 49ØD BØØ1 4C44 41ØD 4ØØ1 5354 52ØD 5ØØ1;
Ø93Ø 4C44 58ØD FØØØ 4F52 49ØD F9Ø3 414E 44ØD;
Ø94Ø F2ØØ 584F 52ØD F3ØØ 4144 44ØD F4ØØ 5344;
Ø95Ø 49ØD FDØ3 5348 52ØD F6ØØ 534D 49ØD FFØ3;
Ø96Ø 4C44 49ØD F8Ø3 4F52 ØDF1 ØØ41 4E49 ØDFA;
Ø97Ø Ø358 5249 ØDFB Ø341 4449 ØDFC Ø353 44ØD;
Ø98Ø F5ØØ 534D ØDF7 ØØ42 52ØD 3ØØ2 425A ØD32;
Ø99Ø Ø242 4446 ØD33 Ø242 5Ø5A ØD33 Ø242 4745;
Ø9AØ ØD33 Ø242 31ØD 34Ø2 4232 ØD35 Ø242 33ØD;
Ø9BØ 36Ø2 4234 ØD37 Ø253 4B5Ø ØD38 ØØ42 4E5A;
Ø9CØ ØD3A Ø242 4E46 ØD3B Ø242 4E31 ØD3C Ø242;
Ø9DØ 4E32 ØD3D Ø242 4E33 ØD3E Ø242 4E34 ØD3F;
Ø9EØ Ø249 444C ØDØØ ØØ53 455Ø ØDDØ Ø153 4558;
Ø9FØ ØDEØ Ø152 4554 ØD7Ø ØØ44 4953 ØD71 ØØ53;
ØAØØ 4156 ØD78 ØØ4F 5554 ØD6Ø Ø449 4E5Ø ØD6Ø;
ØA1Ø Ø54C 444E ØDØØ Ø142 51ØD 31Ø2 424E 51ØD;
ØA2Ø 39Ø2 4952 58ØD 6ØØØ 4C44 5841 ØD72 ØØ53;
ØA3Ø 5458 44ØD 73ØØ 4144 43ØD 74ØØ 5344 42ØD;
ØA4Ø 75ØØ 5348 5243 ØD76 ØØ52 5348 52ØD 76ØØ;
ØA5Ø 534D 42ØD 77ØØ 4D41 524B ØD79 ØØ52 4551;
ØA6Ø ØD7A ØØ53 4551 ØD7B ØØ41 4443 49ØD 7CØ3;
ØA7Ø 5344 4249 ØD7D Ø353 484C 43ØD 7EØØ 5253;
ØA8Ø 484C ØD7E ØØ4C 5344 46ØD CFØØ 5348 4CØD;
ØA9Ø FEØØ 534D 4249 ØD7F Ø34C 4252 ØDCØ Ø64C;
ØAAØ 4251 ØDC1 Ø64C 425A ØDC2 Ø64C 4244 46ØD;
ØABØ C3Ø6 4E4F 5ØØD C4ØØ 4C53 4E51 ØDC5 ØØ4C;
ØACØ 534E 5AØD C6ØØ 4C53 4E46 ØDC7 ØØ4E 4C42;
ØADØ 52ØD C8Ø6 4C53 4B5Ø ØDC8 ØØ4C 424E 51ØD;
ØAEØ C9Ø6 4C42 4E5A ØDCA Ø64C 424E 46ØD CBØ6;
ØAFØ 4C53 4945 ØDCC ØØ4C 5351 ØDCD ØØ4C 535A;
ØBØØ ØDCE ØØ43 414C 4CØD D4Ø6 5245 5455 524E;
ØB1Ø ØDD5 ØØ;
*
```

Listing 2

Listing 2 (modified version)

FULL BASIC Number Cruncher sans BASIC

-by Tony Setaro, 145 Shunpike Rd., Cromwell, Connecticut, USA, 06416

The purpose of this article is to present information to assist in the independant use of the 57109 Number Oriented Processor that is an integral part of the NETRONICS FULL BASIC package.

Background

Like many others, I find FULL BASIC to be extremely limited except for its excellent math capability made possible by the 57109. It naturally follows that the use of the 57109, independant of BASIC should be helpful.

The following narrative,subroutines and examples should assist anyone to make use of this tool in their programming. It is assumed that anyone interested in this subject has the National Semiconductor Data Sheet on the 57109 that was included in the NETRONICS' package. Except for the interfacing, the Data Sheet contains all information needed to use the 57109.

INPUT

All numbers and commands are entered from memory to the math processor (MP) via an OUTPUT 5 (65) instruction. Do this by setting RX to the address of the byte to be input and issue the 65 instruction or set X to the PC and issue a 65 instruction immediately followed by the byte to be input.

The MP requires all numbers and commands to be 2 digits. Each of the 2 digits is in octal notation and only contains 3 bits. However, these 2 octal digits are moved from memory into the MP as a single hexadecimal byte where the 2 high order (leftmost) bits are insignificant. eg., to input the number 1 into the MP,"01"octal must be fed in as 6 bits- 000 001. This must represented in memory as hex "01" or 0000 0001.Straight forward so far,but to input the command "Master Clear", 57 octal must be fed in as- 101 111. This is represented as 0010 1111 or 2F.

Exhibit A is a list of all numbers and commands in hex.

The MP must be synchronized with the 1802 by a delay between each input byte. This delay is accomplished by issuing INPUT 2 instructions until the proper response is received. See the subroutine "I/P Delay".

OUTPUT

Results can be obtained from the MP and put into memory by issuing 2 OUT instructions (16). The first of these instructions is followed by an input delay and the second is followed by an output delay. The output delay is a loop that continues until EF2 = 1. When EF2=1, the MP is ready to output results which can be put into memory by issuing an INPUT 2(6A) instruction either 10 or 12 times depending on whether the MP is in Floating Point (FP) or Scientific Notation (SN) mode respectively. Each 6A instruction must be followed by an output delay.

Each 6A instruction places a byte in memory. The high order nibble of each byte is a 9 which is meaningless and can be eliminated. If the MP is in FP mode, the first output byte identifies the sign (90=+,98=-) and the second byte identifies the position of of the decimal point in the mantissa. This requires manipulation as a 9B indicates the decimal should be placed immediately after the most significant mantissa digit and decrements until 94 indicates the decimal is after the least significant digit. The 8 mantissa digits follow from most to least significant.
If the MP is in SN mode, the first output byte is the most significant exponent digit and the second byte is the least significant exponent digit. The third output byte contains the sign of both the exponent and the mantissa (the 8 and the 0 bits). The fourth byte is the decimal point location which is always 9B and is not needed. The 8 mantissa digits follow.

SUBROUTINES

To illustrate the above, two subroutines are included in exhibits B and C. Both routines utilize a standard call and return technique (SCRT) that always resets X to 2 on a D4 or D5.

The input subroutine is called by D4 XX00 and is followed by a string of numbers or commands in hex notation as shown in exhibit A. The string is terminated by an FE. If an FF byte is encountered in the string, the next two bytes must contain an address where from 1 to 8 numerical bytes are stored followed by either a plus sign (2B) or a minus sign (2D). The string continues after the 2 byte address until an FE or another FF is encountered.

The output subroutine is called by D4 XX3B and is followed by three bytes. The first identifies the number of decimal places that are wanted in the final result (from 00 to 08) and the next two are the address of the final 9 byte answer (8 digits plus a sign). This output routine is used for floating point only and assumes that the MP has not been toggled into SN mode.

The output subroutine first dumps 10 bytes into memory at XXF0-XXF9 while changing the high order nibbles from 9 to 3. Eight bytes of memory at the designated location are zeroed and the sign is stored. The number of decimal places wanted in the final answer is calculated and the result in ASCII is moved to the final location.

SCIENTIFIC NOTATION

It is important to realize that all processing takes place internally in scientific notation mode. The TOGGLE command (22) changes mode from FP to SN and vice-versa. The MCLR command (2F), in addition to resetting all MP registers, places the MP in in FP mode. EE (0B) may be issued with the MP in either mode and does not TOGGLE.

The TOGGLE command only affects the size and format of the results from the MP.

The full results of the MP when in SN are 12 bytes, not 10 as in the attached subroutines.

EXAMPLES

Assuming that you entered the code from exhibit B and C into page XX of memory, enter following into some other page:

```
00   D4XX00          Call Input
03   2F              Master Clear
04   01 02 03 39     123+
08   02 02 3B        22X
0B   FE              Terminate Instruction
0C   D4XX3B          Call Output
0F   00              No decimal consideration
10   YYYY            Address of final answer
12   3012
```

Execute and YYYY should equal 30 30 30 30 32 37 30 36 2B
or 123+ 22X = 2706+

Now enter

```
00   D4XX002F        Call Input and MCLR
04   FFYYYY          Enter numbers at YYYY until a 2B or 2D
07   2134            Enter and √ Commands
09   01020A020539    12.25+
0F   FE              Terminate instruction
10   D4XX3B          Call Output
13   02              Allow 2 decimal places in final answer
14   YYYY            Address of final answer
16   3016
```

Execute and YYYY should equal 30303030363432362B or 2706√ 12.25+ = 64.26+

HEX	Numbers	HEX	Commands	HEX	Commands
00	0	21	Enter	38	Y^x
01	1	39	+	37	1/x
02	2	3A	-	34	√
03	3	3B	X	33	Square
04	4	3C	÷	32	10X
05	5	2F	MCLR	31	EX
06	6	2B	ECLR	35	LN
07	7	16	OUT	36	LOG
08	8	27	SF1	24	SIN
09	9	28	PF1	25	COS
0A	.	29	SF2	26	TAN
0B	EE	2A	PF2	2D	DTR
0C	CS	22	TOG	2C	RTD
0D	π	23	ROLL	1C	MS
		2E	POP	1D	MR
3F	NOP	30	XEY	1E	LSH
		1B	XEM	1F	RSH

2 WORD COMMANDS

20	39	M+
20	3A	M-
20	3B	MX
20	3C	M÷

57109 Commands in HEX

EXHIBIT A

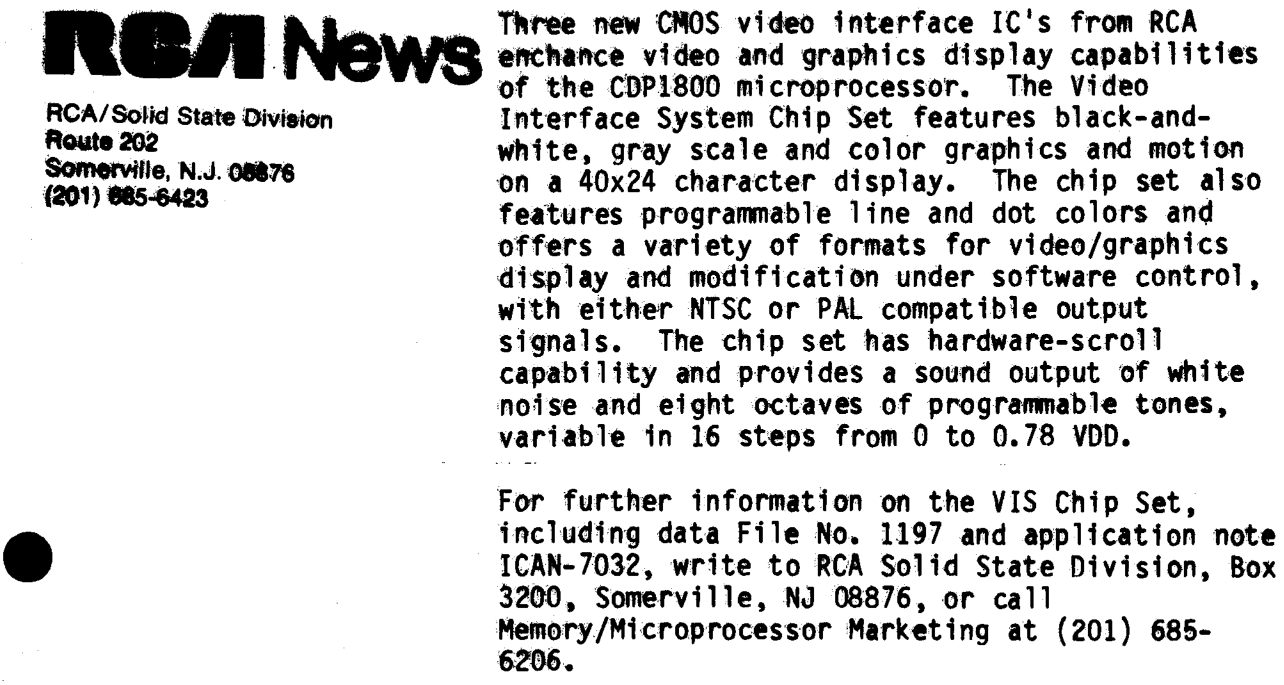
SUBROUTINE : INPUT TO MATH PROCESSOR   CALL: D4 XX00   ssssss...FE Terminates
FF followed by 2 byte addres

```
00   E6              Start     SEX 6      Set X to Link
01   06FBFE321A                Test for FE- Yes, RETURN
06   FB013210                  Test for FE- Yes address follows
0A   65D4XX1C                  Input String- Call IP Delay
0E   3000                      Branch to Start
10   1646BA46AA      Addr      Input from memory- load RA from link
15   D4XX22                    Call Inpmem
18   3000                      Branch to Start
1A   16D5            Ret       Main Return
1C   6A7E7E          IP Delay  Input Delay Subroutine
1F   3B1C
21   D5
22   EA0A            Inpmem    Input memory subroutine
24   FB2B3221                  If + Return
28   FB063293                  If - Change sign then Return
2C   0AFA0F5A                  Zero high nibble
30   65D4XX1C                  Input byte from memory- Call IP Delay
34   3022
36   0000000000                No meaning
```

EXHIBIT B

```
      SUBROUTINE : OUTPUT FROM MATH PROCESSOR    CALL: D4XX3B BB AAAA
                                                 BB= No. of decimal places  AAAA= Address

3B  F80AAC                         RC= No. of characters from MP
3E  93BDF8F0AD                     RD= Address of scratch work area
43  46AB46BA46AA                   Link load RB.0 and RA
49  E36516D4XX1C                   Issue OUT command- Call IP Delay
4F  E36516                         Issue second OUT command
52  3D52                 OP Delay  Output delay - Loop until OP ready
54  ED6A                           OP ready- put byte in memory
56  F0FA0FF9305D                   - There is time for this processing
5C  1D2C                           - before going to
5E  8C3A52                         - OP Delay
61  F808AD                         All MP results are npw in work area
64  F8305A                         Zero area before storing final answer
67  2D1A8D3A64
6C  F8F0AD                         Point RX to sign
6F  F0FB30328F                       is it +?
74  F82D5A2A             Sign      Store - sign
78  1DF83C                         Point RX to decimal point
7B  F7AC                           Calc no. in integer
7D  8BE252                         Calc total no. in answer-
80  8CF4AC                           store in RC.0
83  FCF1AD
86  EA0D73               Move      Move answer from work area to
89  2C2D                             final location
8B  8C3A87
8E  D5                   Ret       Main Return
8F  F82B                 +         Store + sign
91  3076
93  E3650C               CS        Change sign- from Input subroutine
96  D4XX1C                         Call IP Delay
99  D5                             Return
```

EXHIBIT C



RCA News

RCA/Solid State Division
Route 202
Somerville, N.J. 08876
(201) 685-6423

Three new CMOS video interface IC's from RCA enchance video and graphics display capabilities of the CDP1800 microprocessor. The Video Interface System Chip Set features black-and-white, gray scale and color graphics and motion on a 40x24 character display. The chip set also features programmable line and dot colors and offers a variety of formats for video/graphics display and modification under software control, with either NTSC or PAL compatible output signals. The chip set has hardware-scroll capability and provides a sound output of white noise and eight octaves of programmable tones, variable in 16 steps from 0 to 0.78 VDD.

For further information on the VIS Chip Set, including data File No. 1197 and application note ICAN-7032, write to RCA Solid State Division, Box 3200, Somerville, NJ 08876, or call Memory/Microprocessor Marketing at (201) 685-6206.

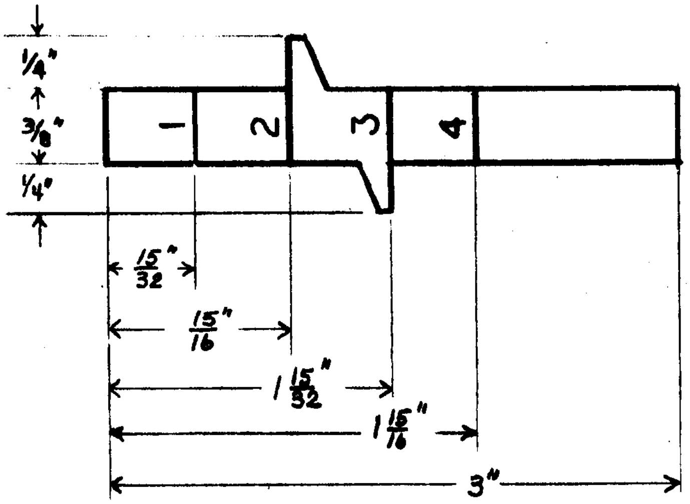
SELECTRIC TYPEWRITER TOOLS

- by R. N. Thornton, 1403 Mormac Road, Richmond, Va. 23229

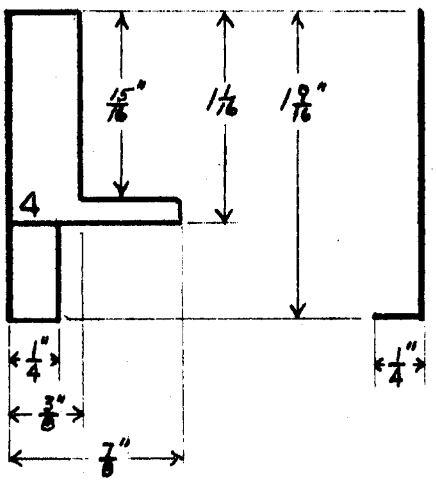
I recently purchased a used Selectric terminal from Worldwide Electronics. Since my budget was limited, I ordered the $200 "D" unit, which is as-is, and guaranteed not to work. The printer arrived in good shape, and with an excellent maintenance manual. After a lot of reading, cleaning, oiling, greasing, and adjusting, it works. This article was typed on the machine. During my work and reading, there were frequent references to two tools which are required, but which are not generally available. One is a Hooverometer, the other a Hand Cycle tool. After asking around, I was finally able to locate and borrow a Hooverometer, and attempted to make one, but was unsuccessful. After reading over the manual, it was apparent that a couple of simple gauges could be made of sheet metal to doo the required adjustments. These two gauges are shown as tools #1 and #2. The Hand Cycle tool allows you to turn the operation shaft to operate the machine manually during adjustments. This is merely a small disk with a threaded rod which screws into the end of the operation shaft on the right side of the typewriter. The trouble is that the tool has a left-hand thread. To solve the problem, I filed flats on opposite sides of the end of the operation shaft and made tool #3, which is placed on the end of the filed shaft and allows you to turn it (counter-clockwise only, please). There are also a set of drawings which show how to use the homemade gauges. The escapement rack position adjustment is a little different than that shown in my maintenance manual, and was taken from an IBM Selectric manual. Both methods result in the same distance, but the IBM method was easier without the Hooverometer.

If you should decide to buy one of these units, ask for a correspondence code machine if they have it, as the type balls for BCD and correspondence machines are different. Balls for the correspondence code machines can be obtained from local stationary stores, but BCD balls are hard to find. I have had to use this printer with the computer keyboard and a translation program only, as I havent been able to get a BCD ball. The correspondence code balls work fine, but the keyboard doesn't match the characters printed.

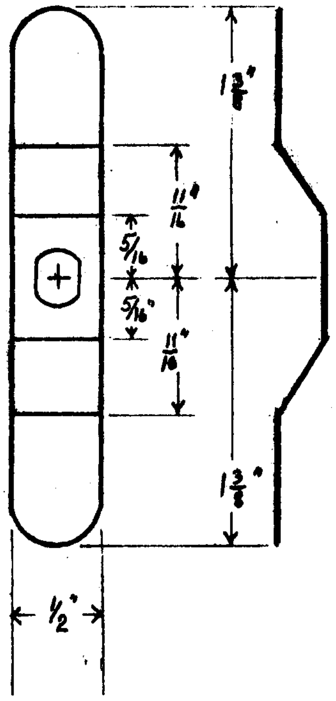
Since I wanted only a printer, and don't like serial interfaces, I removed the transmit coding devices and the interface board, and replaced it with a parallel interface of my own design. I'll be glad to share experiences or offer any help I can to others who may purchase a Selectric. I love it!



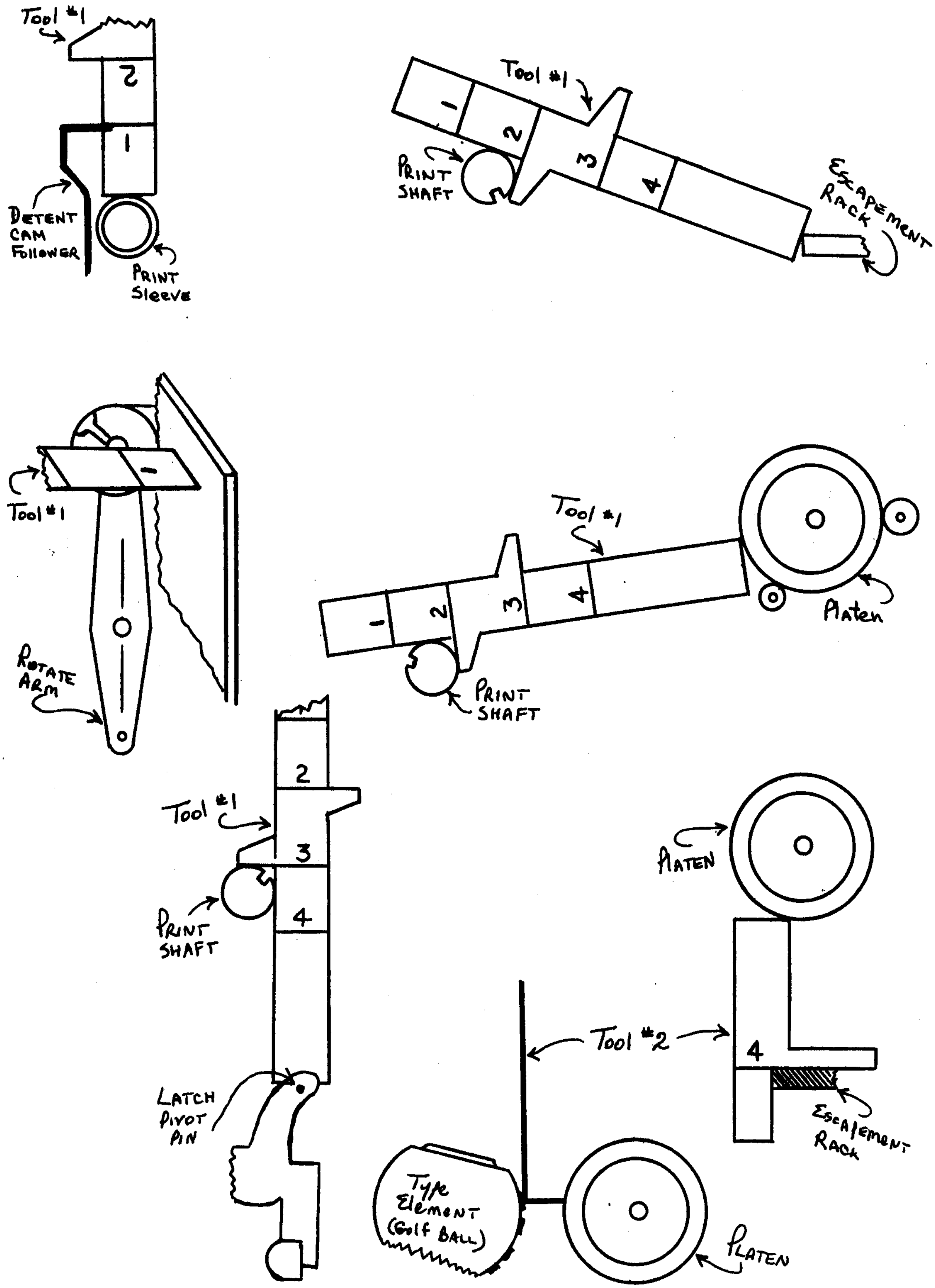
TOOL *1



TOOL *2



TOOL *3


Tool #1
2
1
Detent Cam Follower
Print Sleeve
Tool #1
1
2
3
4
Print Shaft
Escapement Rack
Tool #1
1
Rotate Arm
Tool #1
1
2
3
4
Print Shaft
Platen
2
Tool #1
3
4
Print Shaft
Latch Pivot Pin
Platen
Tool #2
4
Escapement Rack
Type Element (Golf Ball)
Platen

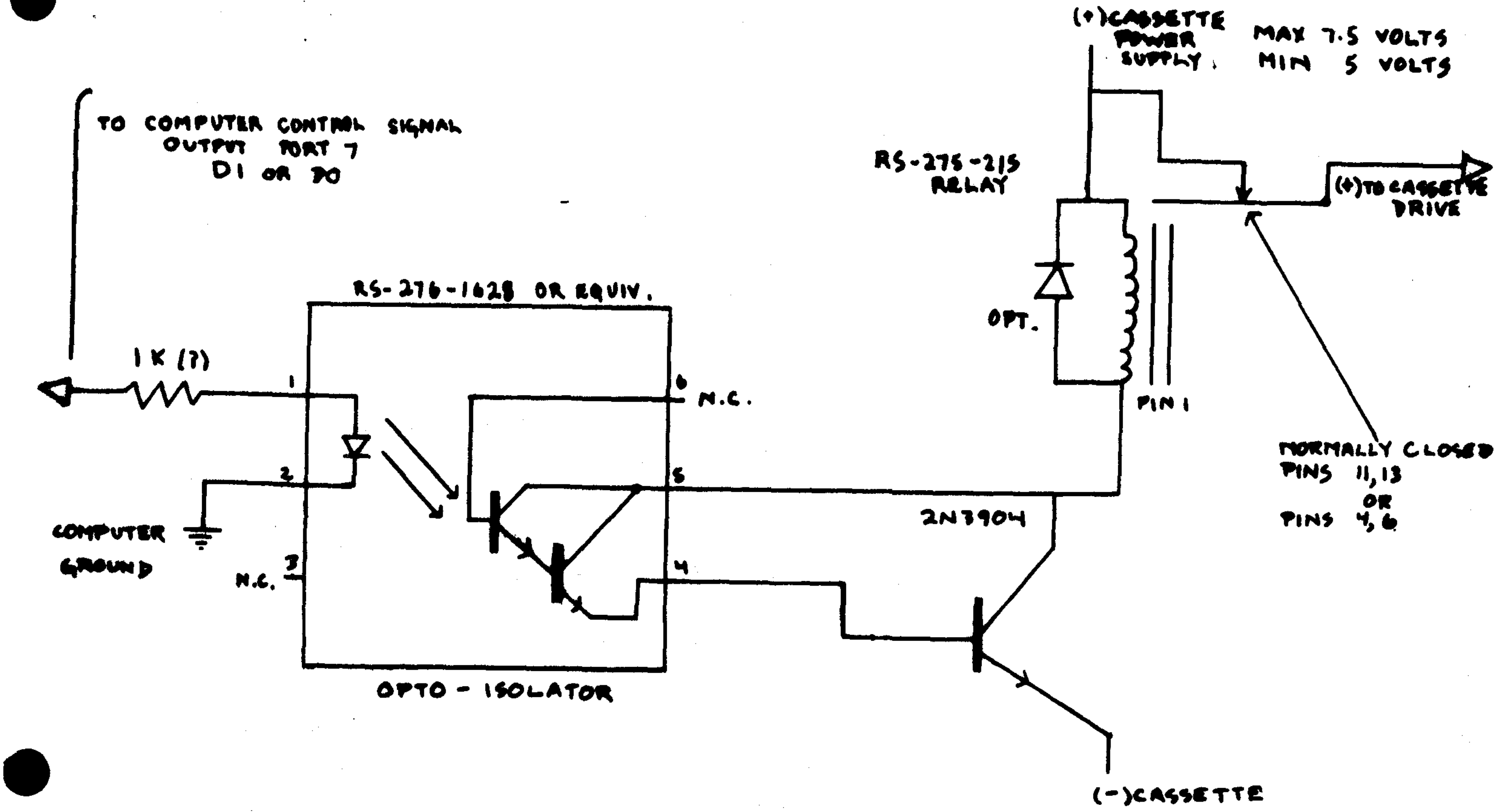
ANOTHER TAPE CONTROLLER

- by Karl W. Schultz, 4069 Forest Ave., Western Springs, Il 60558

Like David W. Schuler (IPSO FACTO #19, P25), I was faced with the problem of putting together a tape controller circuit at low cost. I built a circuit very much like David's, but the sudden switching of the relays caused a loss of power elsewhere in my computer because of the sudden current drain and lack of an adequate power supply. This power problem caused loss of data in my 8K of RAM. Not wanting to sink money into a larger power supply at this time, I went a different route.

I decided to utilize the cassette tape recorder's power supply for the power needed to drive the relay. Since the normally closed contacts on the relay are used, the cassette drive is enabled when the relay coil is de-energized. When the relay coil is energized, the cassette drive contacts are opened, stopping the cassette. Thus current flows either through the relay coil or the cassette drive, but never both (for very long, anyway).

Note the use of the opto-isolator. I used it to keep the cassette drive as isolated as possible from the computer because I was having ground loop problems with the audio portions of the circuitry. So I used both poles of the relay switch to switch both power leads to the cassette on and off. I have had good results from the circuit below. (Same circuit for each drive). I put everything inside the battery compartment of each deck so only two wires come from the computer to each deck (not counting audio wires).



TOTAL CURRENT DRAIN WITH RELAY ON IS ABOUT 150 MA (DEPENDS ON TRANSISTORS AND ISOLATOR)

AN INTERFACE FOR THE EPSON MX-80 PRINTER

- by Harley Shanko

As one adds peripherals to a homebrew microprocessor, unless much advanced planning is done, certain interfaces present problems. This was observed regarding my own system when an Epson MX-80 printer was purchased and interfaced.

Figure 1 illustrates the present printer hardware interface; my orginal (SwTPc PR-40) printer interface was usable with a few mods. Because the MX-80 requires a delayed stobe, relative to the data transitions, the one-shot was added. The negative transition of the complement OUT 3 signal stobes data into the "port" and the positive edge triggers the one-shot.

The MX-80 inputs require about 1.5 ma. to sink them to ground. The 74LS123 and the 74LS05 open collector drivers with 4.7 K pull-ups are more than adequate for this purpose. My cable length is about 6 feet of 40 conductor ribbon cable, with 36 lines interfaced to a 57-30360 Amphenol (Centronics compatible) connector at the printer end.

Listing 1 is my present routine which drives the 6847 VDG and the printer, if enabled. From the keyboard routine I use Q=1 for printer ON, Q=0 for OFF; thus the switch at address 8A9A. If the printer is enabled, the following EF2 tests for whether the printer is ON and not BUSY. Listing 2 has been utilized for driving the printer in the graphics mode. This routine is called via a BASIC USR and dumps 256 bytes to the printer; it then returns to BASIC to provide formatting (spacing to move the graphics from the left margin) and "CRLF" via a PRINT statement, in addition to counting out the number of lines to be printed.

Listing 1

```
8A90 FA . ANI   7F          Mask for 7 bit ASCII
8A92 AF / PLO   F           Save for monitor
8A93 BF ? PHI   F             and T BASIC
8A94 D4 T CALL  8B12        VDG scroll driver
8A97 8F . GLO   F           Get byte
8A98 22 " DEC   2             and store
8A99 52 R STR   2             on stack
8A9A 39 9 BNQ   A0          Test if printer ON (Q=1)
8A9C 3D = BN2   9C          If so, also check if busy
8A9E 63 . OUT   3           If not, send to printer
8A9F 38 8 SKP                 and skip 'inc'
8AA0 12 . INC   2           Non-print exit, account for prior dec
8AA1 D5 U *RET              Exit to calling routine
```

Listing 2*

```
2FE0 F8 . LDI   00          Set count for 256 bytes
2FE2 A0   PLO   0
2FE3 E8 . SEX   8           Use R8 as pointer*
2FE4 63 . OUT   3           Output to printer
2FE5 E2 . SEX   2           Restore x for exit
2FE6 3D = BN2   E6          Printer BUSY?
2FE8 20   DEC   0           Check if
2FE9 80 . GLO   0             count is
2FEA 3A : BNZ   E3            done
2FEC D5 U *RET              If so, return to caller
```

*This was written for Tiny BASIC USR use, thus R8 contains address of first byte to be printed on this pass

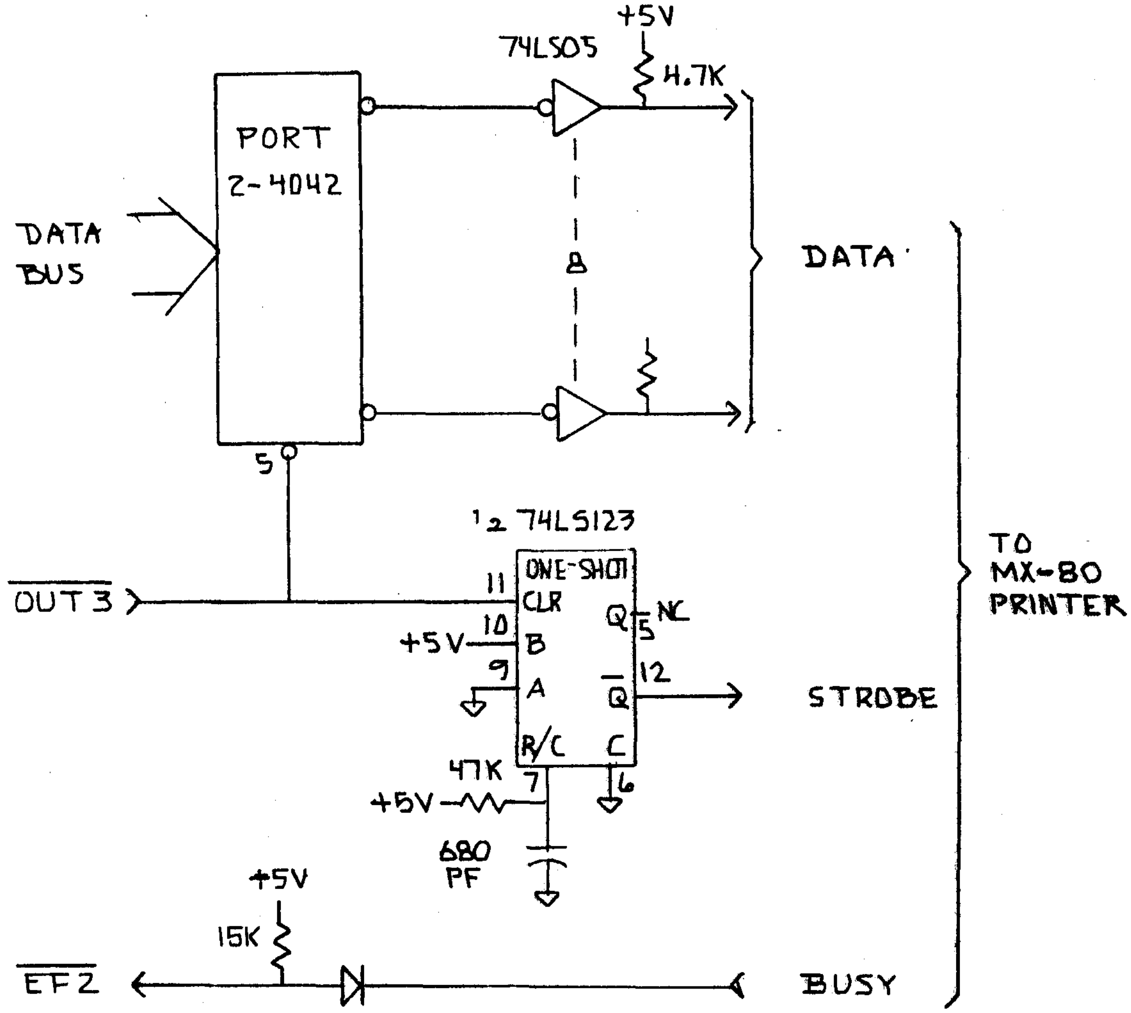
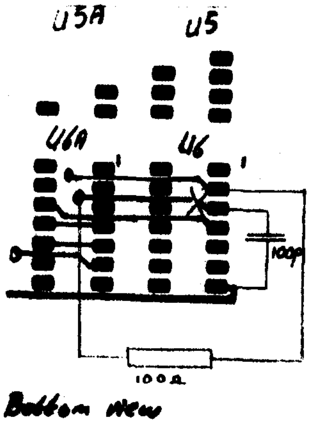


FIGURE 1 PRINTER INTERFACE

QUEST DYNAMIC BOARD FIX

- by Lourens Blok and Cors Bouwhuis Bentrotstraat 28, 7531 AB Enschede, the Netherlands

A problem exists with the wait-state interface between the dynamic board and the ELF II. To correct the problem, delay TPA by placing a resistor capacitor delay in the U6-U6A circuit. Cut the trace from pin 3 of U6, under the board, and connect an 100 ohm 1/4 watt resistor between pin 2 of U6 and the plate through hole for pin 3 U6 (under U6A), and connect a 100 pf capacitor between pins 3 and 7 of U6. Corrections are illustrated on the drawing below:

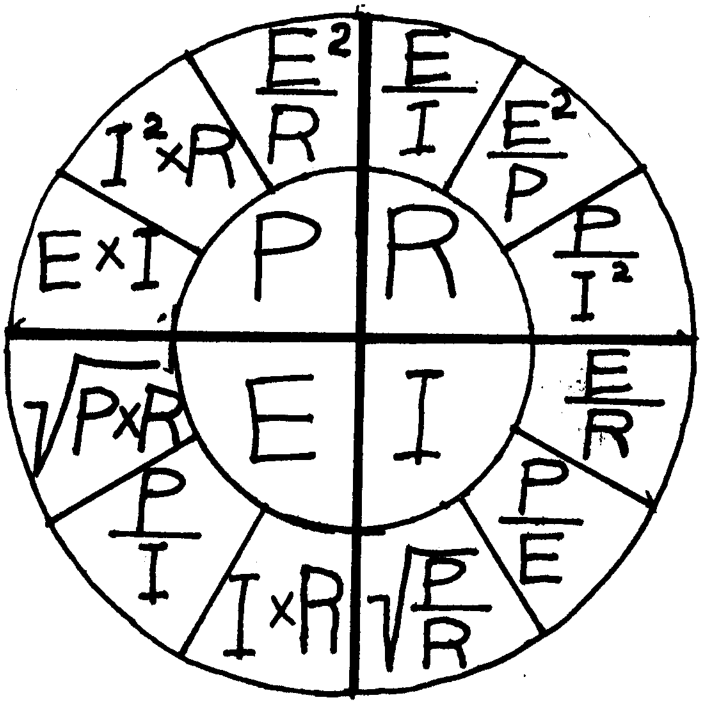


OHM'S LAW
- by H. Hallaska, 212 N. 70th St., Milwakee, WI. U.S.A. 53213

Here is a program that may be useful to members who are electricians, C.E.T. or students thereof.

The program is written in netronics full basis level III. I started in tiny basic, but saw the obvious pitfalls of integer math and the intracacies with working square root subroutines into it, tricking "tiny" into printing decimals and the like, ergo full basis.

The following figure shows OHM'S Law relationships to voltage, current, resistance, and power:



As can be seen from the above figure, there are 12 possible computations, 3 for each of the values of voltage, current, resistance, and power; and as in any math problem, 2 values have to be known if we are going to have something to compute.

When the program is run, the computer asks for 4 inputs; voltage, current, resistance, and power. Enter the 2 known quantities. Enter 0 for the unknown quantities. Enter only volts, amps, etc. never millivolts, milliamps, etc. For example 100 milliamps is entered as .1 amp. Answers are also in volts, amps, etc. Convert them back by moving the decimal.

This program will also serve as a sample program for those who are learning programming in RPN (Reverse Polish Notation) which is the maths syntax in netronics full basic. The program will work with other basics, if converted to algebraic syntax. For instance line 110 could read "110 if E>0 if I>0 PR"R"=;R; (E/I);" OHMS".

```
10 PR"     OHM'S LAW"
15 PR
20 LET E=0
25 LET I=0
30 LET R=0
35 LET P=0
37 LET S=0
38 LET M=0
39 LET D=0
40 PR"VOLTAGE";
45 INPUT E
50 PR "CURRENT";
55 INPUT I
60 PR"RESISTANCE";
65 INPUT R
70 PR"POWER":
75 INPUT P
80 PR
85 PR
90  IF E>0 IF I>0 PR "E =";E;"VOLTS"
100 IF E>0 IF I>0 PR"I=";I;"AMPS"
110 IF E>0 IF I>0 PR"R=":E#I/;" OHMS"
120 IF E>0 IF I>0 PR"P=";E#I*;" WATTS"
130 IF E>0 IF R>0 PR"E=";E;" VOLTS"
140 IF E>0 IF R>0 PR"R=";R;" OHMS"
150 IF E>0 IF R>0 PR"I="E#R/;" AMPS"
160 IF E>0 IF R>0 PR"P=";E#SQRE#R/;" WATTS"
170 IF I>0 IF R>0 PR"I=";I;" AMPS"
180 IF I>0 IF R>0 PR"R=";R;" OHMS"
190 IF I>0 IF R>0 PR"E=";I#R*;" VOLTS"
200 IF I>0 IF R>0 PR"P=";I#SQRE#R*:" WATTS"
210 IF E>0 IF P>0 PR"E=";E;" VOLTS"
220 IF E>0 IF P>0 PR"P=;P;" WATTS"
230 IF E>0 IF P>0 PR"I=";P#E/;" AMPS"
240 IF E>0 IF P>0 PR"R=";E#SQRE#P/;" OHMS"
250 IF I>0 IF P>0 PR"I=";I;" AMPS"
260 IF I>0 IF P>0 PR"P=";P;" WATTS"
270 IF I>0 IF P>0 PR"E=";P#I/;" VOLTS"
280 IF I>0 IF P>0 LET S=I#SQRE
285 IF I>0 IF P>0 PR"R=";P#S/;" OHMS"
290 IF R>0 IF P>0 PR"R=";R;OHMS"
300 IF R>0 IF P>0 PR"P=";" WATTS"
305 IF R>0 IF P>0 LET M=P#R*
310 IF R>0 IF P>0 PR"E=";M#SQRT;" VOLTS"
315 IF R>0 IF P>0 LET D=P#R/
320 IF R>0 IF P>0 PR"I=";D#SQRT;" AMPS"
330 PR
340 PR
500 END
```

1802 Quickies

- by L.A. Hart, Technical Micro Systems Inc., P.O. Box 7227, Ann Arbour, Michigan, U.S.A. 48107

Memory-Mapping the 1851

There's a timing problem in the 1851. If you must memory-map it, you must invert $\overline{\text{MRD}}$ and $\overline{\text{MWR}}$ and interchange them, and delay TPA and CS to the 1851 by two gate delays (use two 4069 inverters, for example).

1854 UARTs Can Argue With Each Other

If two 1854 UARTs are used to talk to each ohter at high baud rates using the DA and CTS handshake lines, there will be a timing problem. The receiving 1854 asserts DA before it finishes receiving the character. If this signal reaches the transmitting 1854's CTS input, it will abort the character in the middle of the stop bit, fouling up reception. A solution is to add about two baud-rate-clocks of delay to the CTS input (both stages of a 4013). Side note: the 1854 is not very good at receiving distorted or badly-timed characters. The improved 1854A is better, but still not as good as most other (NMOS) UARTs.

New 1802 Software Source

Technical Micro systems Inc., P.O. Box 7227, Ann Arbor, MI, USA 48107. They've been selling boards, parts, and systems for a few years now, and have recently added 8TH, a programming system based on the FORTH language, for the 1802. 8TH comes with full source code in a 4K ROM, and includes a monitor, assembler, editor, and compiler, all merged into a single package. Write for details.

Hex Keypad for the ELF (PART II)

- by A. Boisvert, Quebec City, P.Q.

Program used with the hex keyboard to enter consecutive byte in ELF memory.

ADD.	CODE.	Comments
00	90BF	Set start address
02	F810AF	(0010) in reg. F
05	3710	Go to X'10' if enter pressed with RUN
07	3E07	Wait for Data Ready on EF3
09	EF	Set reg. X to F
0A	6C64	Read on display data just entered
0C	7B7A	Set a beep tone
0C	3007	Go wait for the next byte
0E	3007	Go wait for the next byte

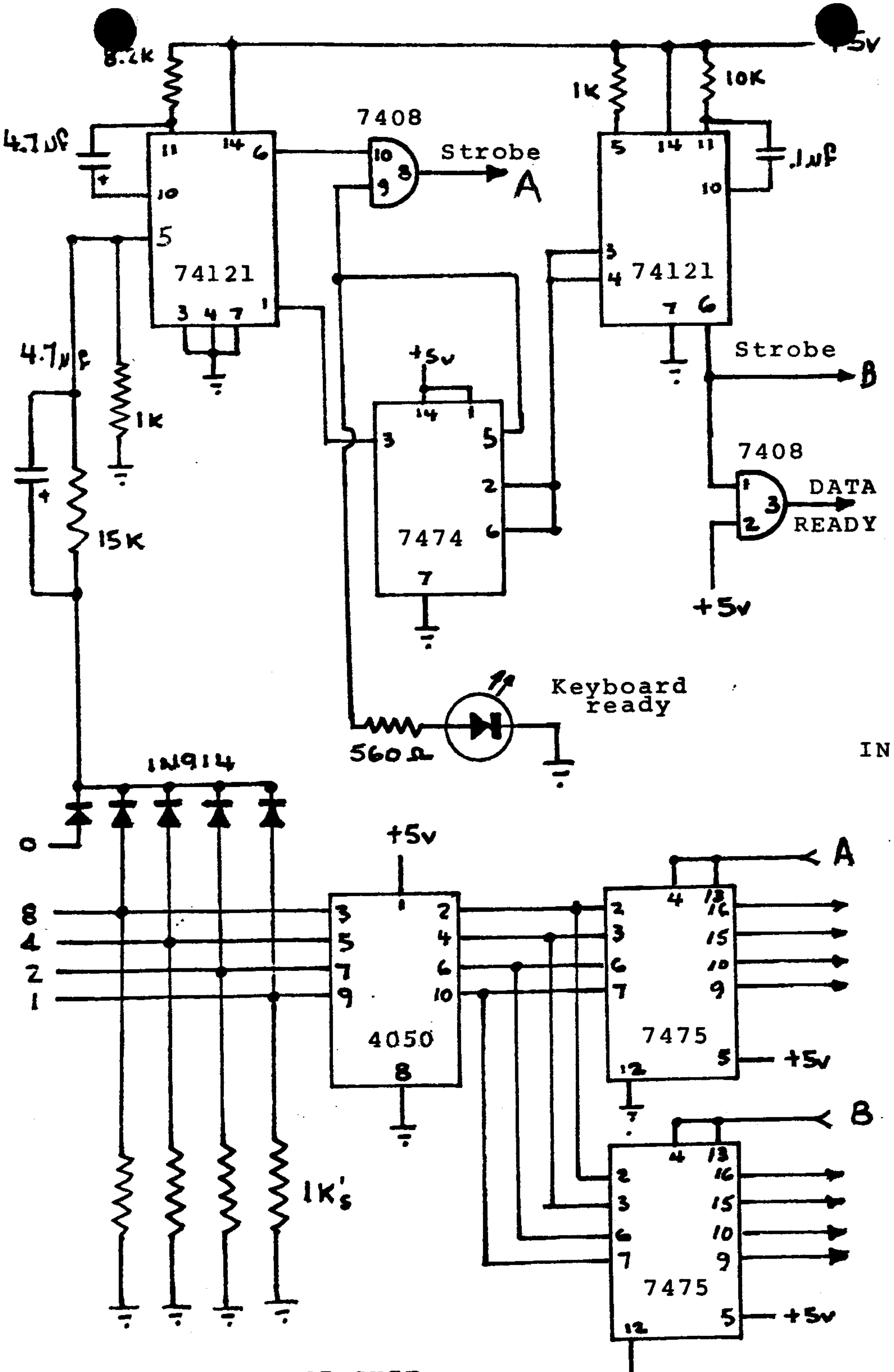
NOTES

Address X'0003' - start address of data (or prog.)
X"0006" - branch to add 10 after the data or program is loaded.
X"0007" - data ready pulse test. Use the proper instruction if you are using another EF line.

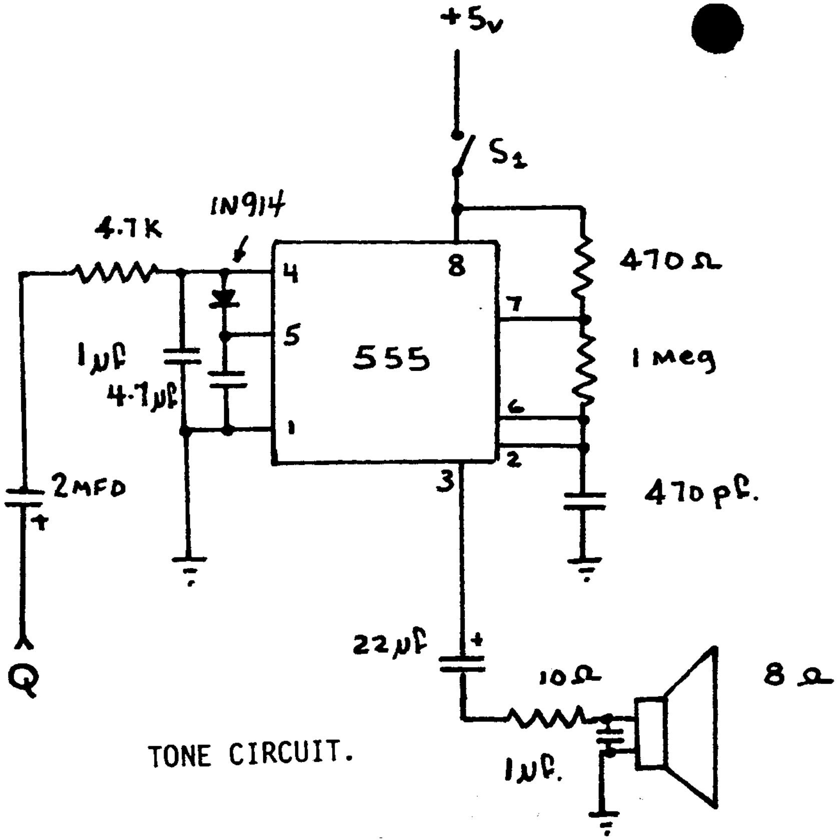
HOW TO USE THE PROGRAM

Just enter the byte on the hex keyboard (high digit first). The data will enter in memory and will be displayed automatically. The memory address will also increment and a tone will sound if the tone circuit is turned on.

To branch to your program, turn RUN to off and while holding the ENTER switch set RUN to on. The ELF will branch to the address set at location X"0006".

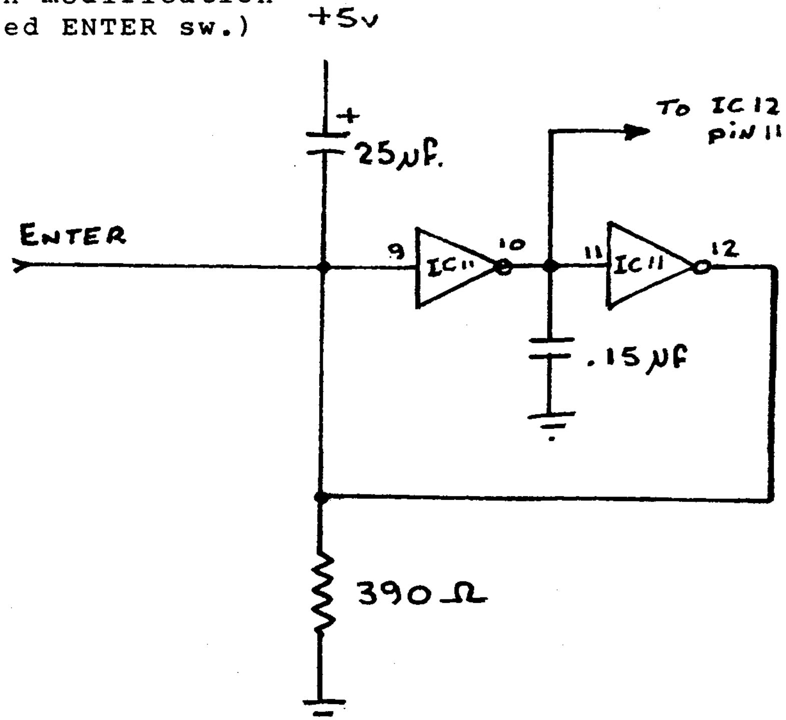


HEX KEYBOARD CIRCUIT.



TONE CIRCUIT.

IN switch modification
(Called ENTER sw.)



USER SURVEY

In a forthcoming issue, I plan to compare the most commonly used monitors. If you use one of the following, please write the Editor, ACE, and give me your opinion of its strengths, weaknesses, general utility and also ideas for improvements. As you may know, the club has been working on a "better universal" monitor for some time, but it is hard to improve on the existing monitors, and even harder to be universal, accommodating all types of I/O. Your opinions will be helpful.

1) Steve Nies "The Monitor"
2) T. Crawford "RCA Bug"
3) R. Cox "Monitor"
4) Quest "Super Monitor"

CLUB COMMUNIQUE

NAME ____________________ DATE ______________

PRODUCT ORDER	QUANTITY	UNIT PRICE *	TOTAL
Hardware			
1. Backplane Ver. 1		$25.00	
2. Kluge (wire wrap) Board		25.00	
3. Eprom (2708) Board		25.00	
4. VDU (6847/2114) Board		40.00	
5. Netronics - Ace Adapter Boardrd		25.00	
6. Netronics - Quest Adapter Board		20.00	
7. 8" Disk Controller Board		40.00	
8. 64k Dynamic (4116) Board		50.00	
9. DMA Adapter Board (ELF II)		3.00	
10. EPROM (27/6/32) Board		40.00	
11. Backplane and I/O board Ver. 2		40.00	
12. I/O Adapter for Backplane Ver. 1		20.00	
Software			
1.		$	
2.			
3.			
4.			
Back Issues			
1. "Defacto" Year 1-3 (Edited)		$18.00	
2. Year 4 Reprint		8.00	
Membership			
Current year - Sept 81 to Aug 22, includes 6 issues of Ipso Facto		$18.00	

VOTE FOR BEST ARTICLE

Issue _____ Article ______________________________________

PRICE NOTE

Prices listed are in local funds. Americans pay in US funds. Canadians in Canadian funds. Overseas in US Funds. Overseas orders, for all items other than membership, add $2.00 US for air mail rather than parcel post. Please use money orders or bank draft for prompt shipment. Personal cheques require up to 6 weeks for bank clearance.

SALES POLICY

We guarantee that all our products work in an A.C.E. configuration micro computer. We will endeavour to assist in custom applications but assume no liability for such use. Orders will be shipped as promptly as payment is guaranteed.

NAME: ______________________________

MAILING ADDRESS: ______________________________

____

____

____

PHONE NO.: ______________________________

Note: Ensure mailing address is correct, complete and printed.
Please ensure payment is enclosed.

ASSOCIATION OF COMPUTER-CHIP EXPERIMENTERS
C/O M. E. FRANKLIN
690 LAURIER, AVENUE
MILTON, ONTARIO
L9T 4R5
