IPSO FACTO

Issue #11
April, 1979

(The Newsletter of the Association of Computer Experimenters)

TABLE OF CONTENTS PAGE

Editor: Bernie Murphy
Invaluable Assistants: Wayne Bowdish, Tom Crawford, Vic Sydiuk, Diane York, and all contributors to this issue.

Send Newsletter correspondence to: Bernie Murphy
102 McCrany Street,
Oakville, Ontario
Canada L6H 1H6

ACE EXECUTIVE COMMITTEE

In accordance with the Constitution, the 1978-79 Executive Committee approved at the Annual General Meeting is:

President	KEN BEVIS	220 Cherry Post Drive, Mississauga, Ontario, L5A 1H9 (277-2495)
Past President	TOM CRAWFORD	50 Brentwood Drive, Stoney Creek, Ontario, L8G 2W8 (662-3603)
Secretary/ Treasurer	GEORGE YORK	60 Chester Road, Stoney Creek, Ontario, L8E 1Y2 (664-5264)
Newsletter Editor	BERNIE MURPHY	102 McCrany Street, Oakville, Ontario, L6H 1H6 (845-1630)
Program Co-ordinator	BERT DEKAT	P.O. Box 137, Lynden, Ontario LOR 1TO (647-3931)
Training Co-ordinator	ROD DORE	660 Oxford Road, Unit 32, Burlington, Ontario, L7N 3M1 (681-2456)
Hardware Co-ordinator	FRED FEAVER	105 Townsend Avenue, Burlington, Ontario, L7T 1Y8 (637-2513)
Membership Co-ordinator	WAYNE BOWDISH (Temporary)	149 East 33rd Street, Hamilton, Ontario, L8V 3T5 (388-7116)
Newsletter Publishing Committee	DENNIE MILDON	44 Wildewood Avenue, Hamilton, Ontario, L8T 1X3 (385-0798)
	JOHN HANSON	955 Harvey Place, Burlington, Ontario, L7T 3E9 (637-1076)

ACE CLUB MEETING SCHEDULE

Unless notice to the contrary, the following is the meeting schedule until the end of May. The April 10 meeting will be a tour of the Hamilton Spectator newspaper facility. The May 22 meeting will be a tour of the Colortron film processing plant in Stoney Creek. The May 8 meeting will be the Annual General Meeting that includes elections of the new club executive.

DATE	TUTORIAL/MEETING	TUTORIAL
Apr 10	8:00 (TOUR)	
24		7:30
May 8	7:00/8:00(ANNUAL GENERAL MEETING)	
22	7:30 (TOUR)	

ALSO--Unless notice to the contrary, all meetings will be held in the Stelco Wilcox St. Auditorium.

EDITOR'S REMARKS April 7, 1979.

CASSETTE INTERFACE UPDATE

The club has decided to produce a "Kan$as City" cassette interface kit. We will assemble 16 kits (complete with PC board) that interface to the TEC-1802 bus. All artwork and diagrams will be found in the next issue of the newsletter. I felt it would be better to have local users build the first versions in case problems do occur. First owners of the Kit, can publish their experiences in getting their cassette interface going.

DUE TO LACK OF INTEREST ACE WILL NO LONGER EXIST!

Now that I have your attention, I have a serious topic to discuss. Up to the time of writing this diatribe (April 7, 1979) we have not received ANY nominations for the 1979-1980 Executive. If there are no nominations by the time of the ANNUAL GENERAL MEETING (Tuesday, May 8, 1979), the current Executive may have no choice but to fold the club! COME ON MEMBER --- DON'T LET THE OTHER GUY DO IT!

The current Executive has served their turn. It's time for new blood to guide and run the club.

If you want a club (and newsletter!) next term, send your nominations to TOM CRAWFORD, 50 Brentwood Drive, Stoney Creek, Ontario, L8G 2W8 (662-3603).

ASCII TO HEX CONVERTER

Richard Mack
1748 Fenwick Dr.
Santa Rosa, Calif. 95401

Here is a solution to change ASCII to Hex for your members that have ASCII Keyboards and basic. I have converted my system and it has worked fine without any problems. My system consists of:

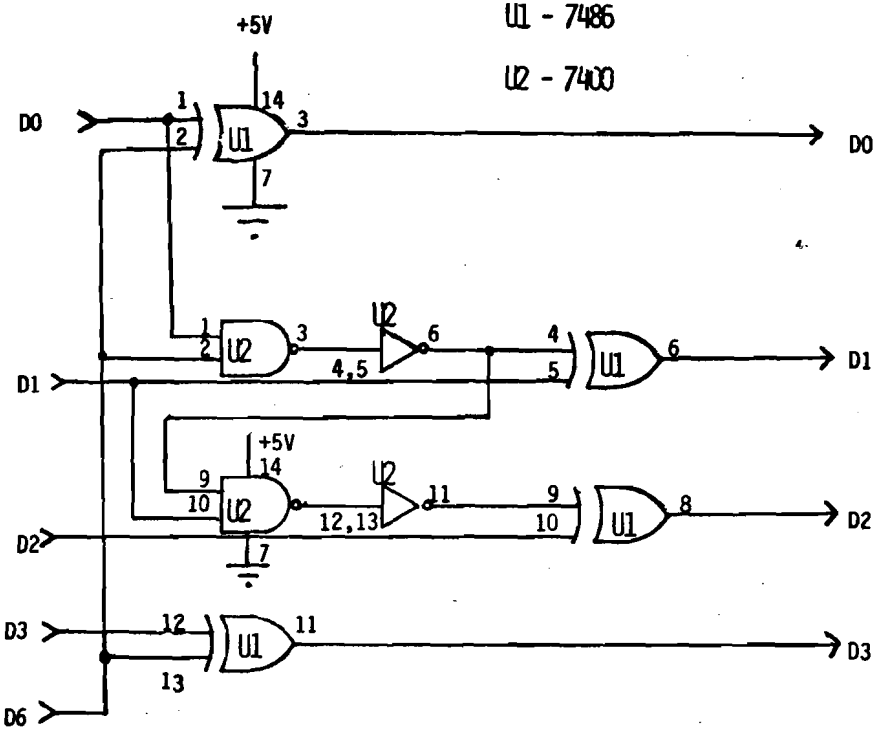
1. Netronics Elf II
2. Big Board
3. 8K of memory
4. ASCII Keyboard
5. TINY BASIC
6. One model 26 Baudot Teletype
7. UART and modem (not yet running)
8. Light pen (from Netronics)

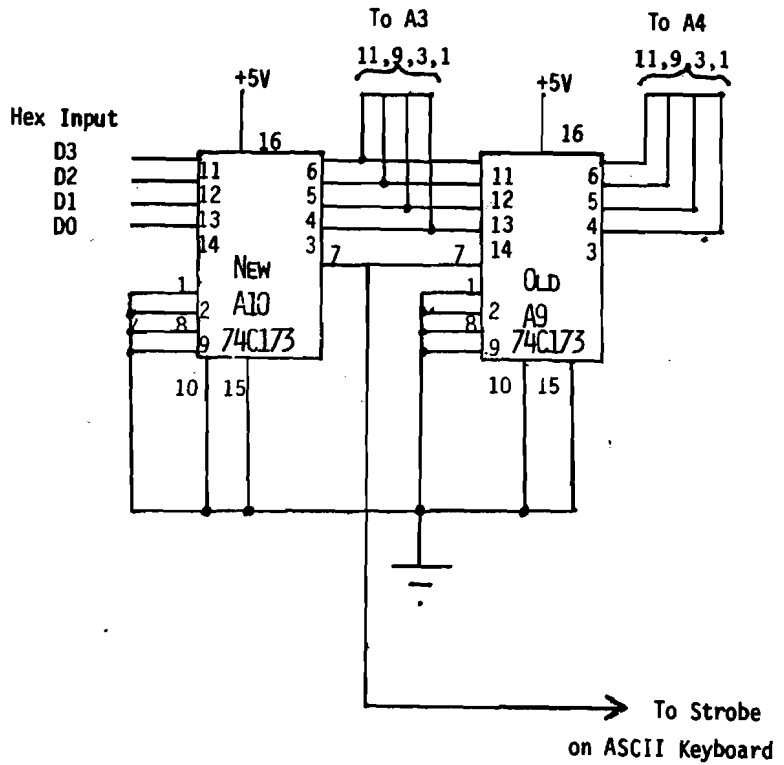
Here is what you must do to convert your Keyboard. First take out the A10 IC 74C922 and save it. Then solder two wires to the input switch on the main board and bring them up to your Keyboard or panel near the keyboard. At this point, you could solder these wires to an unused Key or add another. Be careful, I tried to use an unused Key on my Keyboard and it did not work. Next, wire up the conversion circuit on a wire wrap or printed board. If you want, you can add the new A10 74C173 on the same board. Wire the new A10 into A9 as shown and you now have two latchs. I wired my 74C173 to a wire wrap socket I installed in A3(4016) on the main board as the A9 is connected there also.

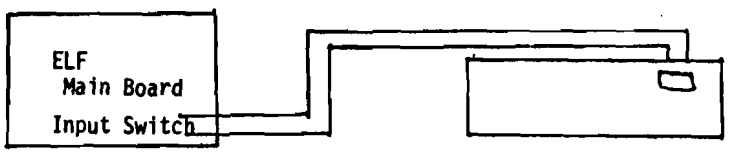
Now for the test! After power has been turned on, you enter your monitor C0, press input switch, F0, load switch, turn on Run switch and enter 04,00,00,0F,FF, the same way you entered C0,F0,00. When basic has been loaded, turn off Run and on again. Now you can use the ASCII Keyboard with return switch. If you return to monitor you use the input switch. The reason for this is the monitor looks at EF4 and basic looks at EF3. The signals are present at both, but only the port open to it works.

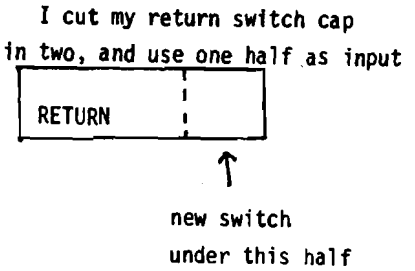
My thanks to Edward Copes who wrote the article on the Hex conversion in Kilobaud, June 1978. I added the latch.

I also typed in the TIC-TAC-TOE game for the light pen found in Popular Electronics, Nov. 1978. I changed one location 0454 byte 3F to 3E and now I use the light pen to start scanning instead of the input switch. Hope you enjoy the conversion.


U1 - 7486
U2 - 7400
+5V
D0
D1
D2
D3
D6
U1
U2
4,5
12,13


Hex Input
D3
D2
D1
D0
+5V
To A3
11,9,3,1
To A4
11,9,3,1
NEW
A10
74C173
OLD
A9
74C173
To Strobe
on ASCII Keyboard


ELF
Main Board
Input Switch


I cut my return switch cap
in two, and use one half as input
RETURN
new switch
under this half

PACKET RADIO (USING AN 1802)

Ken Smith VE3HWB
Glen Simpson VE3DSP

Packet radio is the transmission of digital information, usually ASCII messages, in block or "packet" form. Because of the block structure of the data, transmissions are usually short and many users can share a common channel. Transmissions are in half duplex mode. Although a user will usually wait for a clear channel, "collisions" can occur and the message must be sent again.

A header is used to give origin, destination, message #, message length, etc. Error checking codes are used. If an error is detected, the receiver will send a negative acknowledge (or acknowledge or no errors) and the sender will repeat the message until it is received without errors.

We have designed and are testing our own transmission system. It is similar to many packet systems, although there is no common standard in the industry. It is a basic system but was designed to be flexible and was not over-simplified to have severe limitations.

Modulation Method:

Frequency shift keying (FSK) is used, due to it's popularity and is a good balance between hardware complexity and performance in the presence of noise and interference. Since the transmission would be at VHF frequencies, a great simplification would be made if existing or ready available equipment could be used for transmitting and receiving. By using frequency modulation and restricting the bandwidth to about 12 KHz, common VHF transceivers could be used. The FM detector in the receiver becomes the FSK demodulator.

Given that the bandwidth is to be restricted to 12 KHz, the maximum easily achievable standard baud rate is 4800 baud. In order to maximize flatness of in-baud spectrum, a modulation index of 0.3875 is used. At 4800 baud, the frequency shift is set to 3720 Hz. In order to minimize sideband spillover, low-pass filtering of the baseband signal is done.

Summary of transmission characterictics

Baud rate: 4800 baud
Framing: 8 bit data, Asynchronous - 1 start, 1 stop bit
Modulation: FSK, 3720 Hz. carrier shift
Polarity: Mark = Lo frequency Space = High frequency

Before filtering and transmission, the serial data is converted to a differential return-to-zero form, as seen in fig. 2. The resulting waveform now has a constant average DC value of zero, allowing it to be AC coupled. It is now possible to modulate a frequency synthesizer, which do not respond to modulating signals below 50 Hz.

The receiver circuit in Fig. 3 is designed so that modulation level and transmitter centre frequency is not critical. The comparators determine whether a "0" or "1" is sent. The decision threshold is set to ½ the peak value (+ or -) from zero. The timing and clocking is best determined from when the signal reaches it peaks (zero slope).

Protocol:

Fig. 4 shows the data format. Because of the AC coupled nature of the system, the frames should be continuous, without idle periods. (i.e. 1 stop bit). The 512 null characters allows the transmitter and receiver to "get going". Call letters are used for source and destination. For error checking, the so-called "cyclic redundancy code" or CRC is used. This is a form of a checksum, but much more complicated. The probability of errors going undetected (from two or more errors "cancelling out") is at least 10000 times less than a simple checksum. CRC's are used on almost all commercial tape and disc drives.

HEADER FORMAT

FRAME		HEX CODE	
-512to-1	NUL	FF	Null characters
0	SYN	10	Sync character (after nulls)
1	SOH	01	Start of header
2-11	DESTINATION	10 ASCII Char	Station call letters
12-15	"b DE b"	20444520	"From"
16-25	SOURCE	10 ASCII Char	Station call letters (if less than 10 chars required, fill in remainder with spaces)
26,27	PACKET #		4 digit BCD code in 2 bytes
28	PACKET TYPE	00 - BINARY DATA IN TEXT	
		01 - ASCII CHARS IN TEXT	
		06 - ACK (acknowledge receipt of message)	
		15 - NAK (Negative)	
29,30	LENGTH of Text	(Straight binary count) 2 bytes	
31,32	CRC	2 bytes	Error check over frames 2-30 inclusive
		TEXT	
	STX	02	Start of text
	TEXT		# bytes = LENGTH in header
	ETX	03	End of text
	CRC	2 bytes	-over text, not including STX and ETX

NOTE: for PACKET TYPE ACK or NAK, only header is sent.

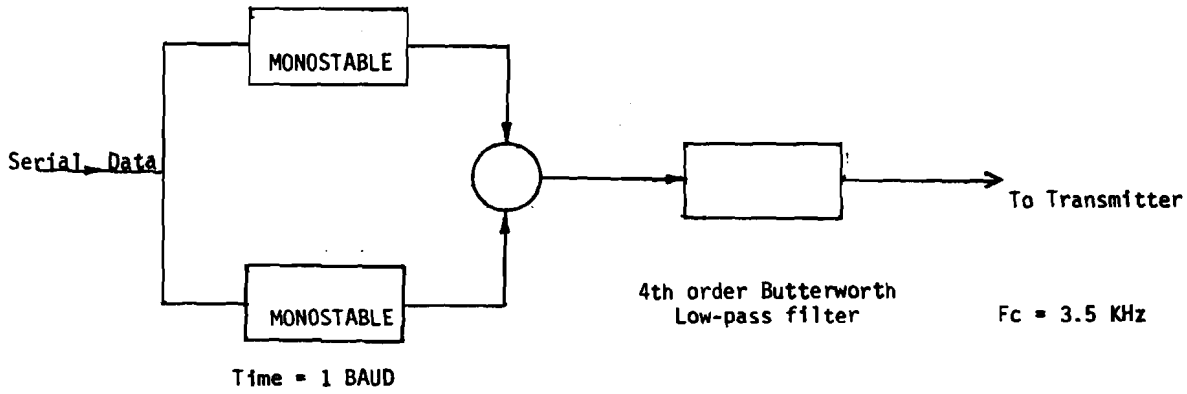


Figure 1

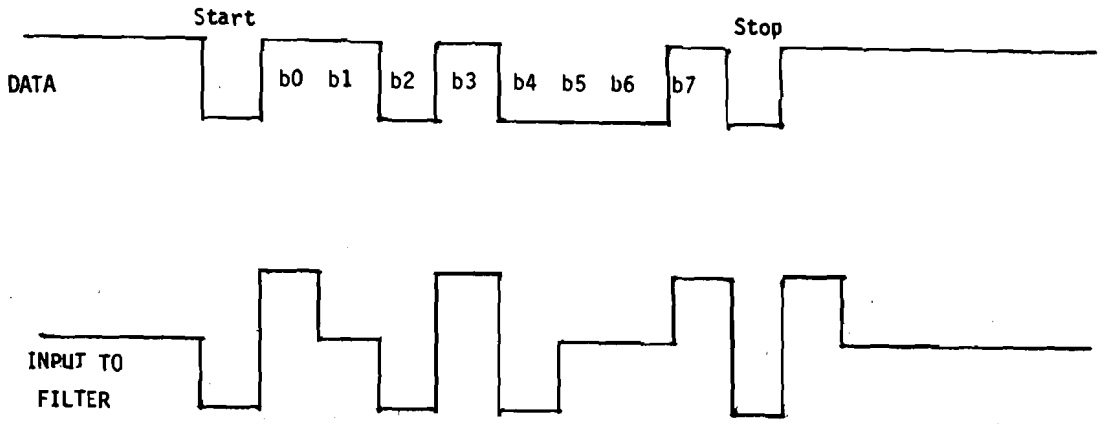


Figure 2

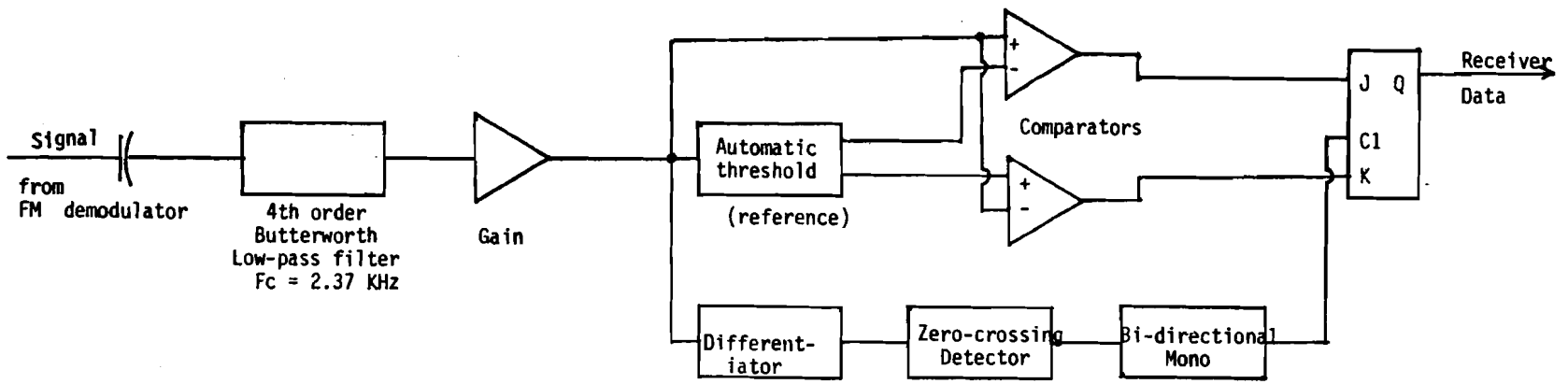


Figure 3

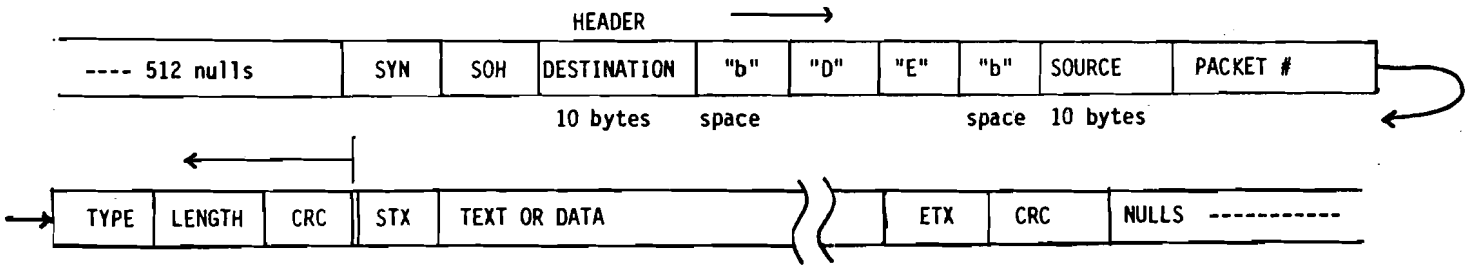


Figure 4

A 2708 EPROM PROGRAMMER BOARD

Malcolm Coyne
104-115 Cherryhill Blvd.
London, Ontario

Earlier on this year as part of a project at school I found that it would be advantageous to have an EPROM programmer at home. It was necessary that this programmer be simple, cheap, CMOS compatible and small. After a search through current literature it became evident that no one had a circuit that fitted my needs, so I set about designing my own using the best features of the programmer circuits I could find. The resulting circuit is shown in Figure 1.

CIRCUIT DESCRIPTION:

The circuit function is quite simple. The AND gates of U1 decode the N lines of the 1802 and provide clock pulses for the latches. U3 and U4 are decoded for the 62 output instructions and latch the data to be programmed into the 2708. U5 latches the four least significant bits of the data bus in response to a 61 instruction and control the various functions of the programmer. U6 is used to generate the addresses into which the data is to be programmed. Finally U2 in conjunction with Q1 is used to switch the 25 to 27 V supply for programming pulses.

CIRCUIT OPERATION:

It is appropriate at this point to go into the requirements of the 2708 for programming. The most important thing about the 2708 is that the data bytes must be programmed in sequence from the lowest to highest address, not randomly. The second important fact to remember is that each programming pulse should be approximately 25 to 27 volts and 1 millisecond in duration and the total number of pulses delivered to any location should be approximately 100. This means that the addresses for the complete 1 K must be stepped through 100 times.

With the above considerations in mind the circuit's operation can now be explained. Upon start of programming it is necessary to reset the counter U6 and make sure the programming pulser is off. This is done by outputing 02 to the control port and then outputing 00. Next the data to be programmed into the first address of the 2708 is output to the data port. A 1 MS pulse is then applied to the program input of the EPROM by first outputing 08 to the control port, delaying 1 MS and then outputing 00. This completes the first programming cycle of the first address. The address counter is then incremented to the second address by outputing a 01 then a 00 to the control port. The data to be placed at this address is then loaded into the data port and

a programming pulse applied. This process is repeated until the address counter (U6) overflows after 1 K has been programmed. The overflow is detected by the flag line EF 1. The entire process is now repeated and after 100 repeats the EPROM is programmed and ready to use.

CONSTRUCTION:

Layout and assembly are not overly critical although the lines associated with the 25 volt supply should not be placed too close to other lines in order to minimize possible noise. I used a Radio Shack 'multi-purpose edge-card board' to mount the circuit and the 25 to 27 volt supply may most cheaply be supplied by three 9 volt batteries.

SOFTWARE:

The only program I have written to run the EPROM programmer board is an integral part of a monitor I wrote for my system. I have included a copy of this routine but it would require changes to operate with a different system.

PROM PROGRAMMER ROUTINE

```
00  B0   E3       SEX       Zero Programmer Board
    B1   6100     OUT 1
    B3   6200     OUT 2
    B5   7A       REQ
    B6   D4       SEP       Call input routine to get
    B7   0080                start address
    B9   7B       SEQ       Prompt Ready to start
    BA   3FBA     BN4       Start by depressing & releasing I
    BC   37BC     B4
    BE   7A       REQ
    BF   F864AE   LDI PLO   Loop constant 100(10)
    C2   8FAC     GLO PLO   Set start address
    C4   9FBC     GHI PHI
    C6   E3       SEX
    C7   6204     OUT 2     Reset address counter
    C9   6200     OUT 2
    CB   EC       SEX
    CC   61       OUT 1     Output Byte to be programmed
    CD   E3       SEX
    CE   6201     OUT 2     Pulse programmer 1 MS
    D0   F814AD   LDI PLO
    D3   2D       DEC
    D4   8D       GLO
    D5   3AD3     BNZ
    D7   6208     OUT 2     Increment address counter
    D9   6200     OUT 2
    DB   3CCB     BNI       If end of K jump
    DD   2E       DEC
```

PROM PROGRAMMER ROUTINE (CONT'D)

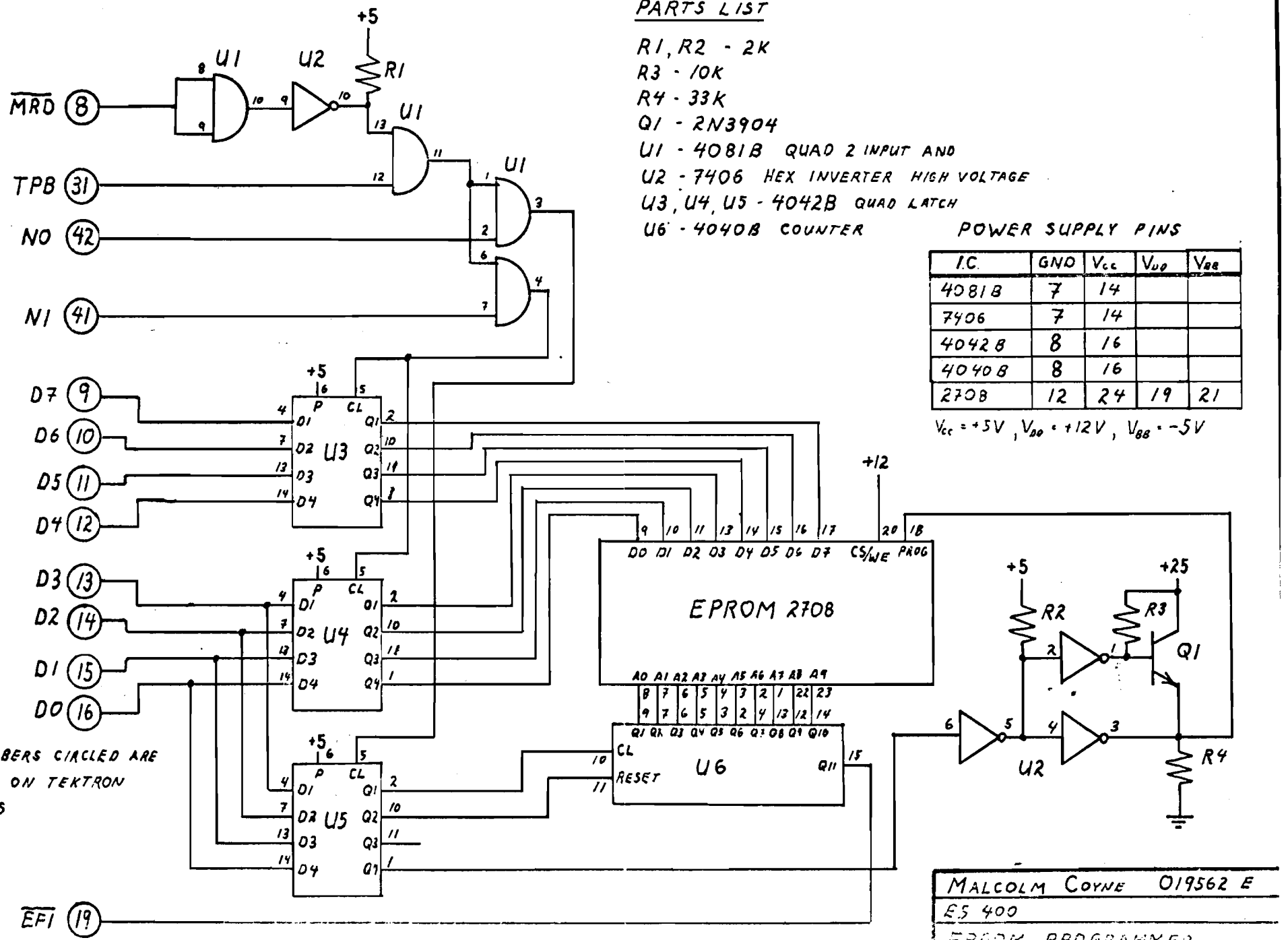
```
DE    8E          GLO
DF    3AC2        BNZ         If not end of loop jump
E1    7B          SEQ
E2    3000        BR          Jump to monitor
E4
```

NOTES: 1) Program counter is R(3)

2) Standard call and return technique as outlined in RCA manual is used to call subroutine at address 00 B6. This routine inputs the start address of the 1 K block to be programmed and stores it in R(F).

3) Address 00 D1 is timing constant to give 1 MS pulse.
With system clock 1 MHz time constant 14
1.79 MHz time contant 25
2 MHz time constant 29

4) At completion of programming routine jumps to start of current page.


PARTS LIST
R1, R2 - 2K
R3 - 10K
R4 - 33K
Q1 - 2N3904
U1 - 4081B QUAD 2 INPUT AND
U2 - 7406 HEX INVERTER HIGH VOLTAGE
U3, U4, U5 - 4042B QUAD LATCH
U6 - 4040B COUNTER
POWER SUPPLY PINS
I.C. | GND | Vcc | VDD | VBB
4081B | 7 | 14 | |
7406 | 7 | 14 | |
4042B | 8 | 16 | |
4040B | 8 | 16 | |
2708 | 12 | 24 | 19 | 21
Vcc = +5V, VDD = +12V, VBB = -5V
MRD 8
TPB 31
NO 42
NI 41
EFI 19
D7 9
D6 10
D5 11
D4 12
D3 13
D2 14
D1 15
D0 16
U1
U2
R1
U3
U4
U5
U6
EPROM 2708
D0 D1 D2 D3 D4 D5 D6 D7 CS/WE PROG
A0 A1 A2 A3 A4 A5 A6 A7 A8 A9
Q1 Q2 Q3 Q4 Q5 Q6 Q7 Q8 Q9 Q10
CL
RESET
Q11
+5
+12
+25
R2
R3
Q1
R4
NUMBERS CIRCLED ARE PINS ON TEKTRON BUSS
MALCOLM COYNE 019562 E
ES 400
EPROM PROGRAMMER

DOUBLE-BUFFER SPEEDUP HARDWARE FOR 64X128 GRAPHICS WITH THE COSMAC 1802 AND THE 1861 VIDEO CHIP

Ben Hutchinson
30 Partridge Road
Lexington, Mass
USA 02173

The accompanying diagrams and tables describe a "ping-pong" or double-buffer memory system which can be added on to an essentially unmodified 1802-1861 combination to double the available graphics resolution in both axes, with square picture elements. The add-on buffer is usable either with a VIP and the CHIP-10 interpreter, or with an ELF or other COSMAC system with machine language programs.

The complete system uses 18 ordinary integrated circuits, 14 and 16 pin DIP'S. Two are CMOS hex buffers, two are high-speed 256X1 TTL RAMS, and the rest are common 7400 TTL. LS TTL could be used instead, and the package count could be reduced slightly by using data selector or ROM logic for control. The packaging consisted of wire-wrap sockets fastened with 2-56 screws to an OK Machine & Tool HB-1 Hobby Board; this board fits the 44-pin Expansion Interface socket on the VIP (left rear). If your VIP has no sockets, the OK CON-1 connector, designed for wire wrap, will fit with only slight dogleg bending of the wire wrap pins. Tie the bus strips on the card into interleaved grids, one for +5 volts and one for ground, with frequent cross-connections; distribute at least 5 or 6 0.01 uf disc ceramic bypass capacitors around the board, with short leads, between +5 and ground buses. Provide more DC power; the board takes about 300 MA with standard TTL. I use the stock VIP power supply just for the buffer board; the VIP; itself (with 4K memory) now runs from a separate 1-amp supply.

Functionally, the buffer is simple: a pair of 128-bit memories (256-bit stock RAMS used inefficiently), with address counters and control. One of the two memory systems (called "Left" and "Right" on the schematics) accepts and stores two successive 64-bit lines of video output data from the 1861 at its standard 1.76 MHz rate. Meanwhile, the other memory system is putting out high-resolution twice, on two successive TV scan lines. The stored video came, of course, from the 1861 on the previous two scan lines; the twin memory systems swap roles every two scan lines. The role swapping is controlled by the 7474, which functions as a 2-bit counter clocked by the horizontal sync pulses. The Q and $\overline{Q}$ outputs of this counter select either the 1.76 MHz or the 3.52 MHz clock for each of the 8-bit address counters (74163's), and switch the memory chips to either READ or WRITE mode as appropriate. The memories are Fairchild 93421's, very fast (35 nsec) TTL units with tri-state outputs.

The basic concept and operation of the buffer are, as described above, pretty straightforward; the device data sheets will clear up any subtle details. The control of the address counters is, unfortunately, much more complicated and hard to follow. Study the two tables which describe the sequence of counter states, STOP states, and control signals.

The basic requirement is that the counters start at exactly the right time, i.e., the beginning of each scan line, in the proper initial state (0 or 64), and stop counting and hold exactly 64 counts later (when writing) or 128 counts later (when reading out) as the video scan line ends. A pulse signal can easily be derived from COSMAC state code signals SC0 and SC1 gated with timing pulse TPB; the first occurrance of this pulse coincides with the beginning of the actual video output (left edge of the display window). This signal (called S2•TPB on the drawings) thus serves nicely to start the counters; however, there is no easy way to derive a corresponding signal at exactly the right time to stop the counters. S2, the DMA state, ends 1/8 line too soon. The solution chosen is to have the counters stop themselves when they reach the proper state; as noted in the table, this happens whenever the MSB=1 in state 128 or 192. The MSB is fed back through an inverter to the counter-enable input to do this.

When in the read (output) mode, the counter reaches the proper stop state (128) naturally, after counting up from 0 to 127 in one scan line. When in the write (input) mode, the counter must stop after state 63. When a logic gate array senses state 63, the top two data-input bits of the counter are made = to 1, and the $\overline{\text{LOAD}}$ input is made active (=0), causing the counter to go to state 192 on the next clock pulse and stop there.

The S2•TPB signal actually occurs 8 times during each scan line, but only the first occurrence is wanted. Furthermore, the counter must be either set to 0 (if it's stopped in state 128) or to 64 (if it's stopped in state 192). These requirements are met by gating S2•TPB so that it only causes counter reset or jam set to 64 when the counter is already stopped at either 128 or 192 respectively.

The INTERRUPT signal from the 1861 occurs at the beginning of the video frame, and sets both address counters and the ping-pong counter to the proper initial states to start everything off on the proper foot. The first (top) line seen on the screen is actually the last line from the previous frame, sometimes with some displacement. This effective loss of one line out of 64 seemed to me not worth the trouble to fix.

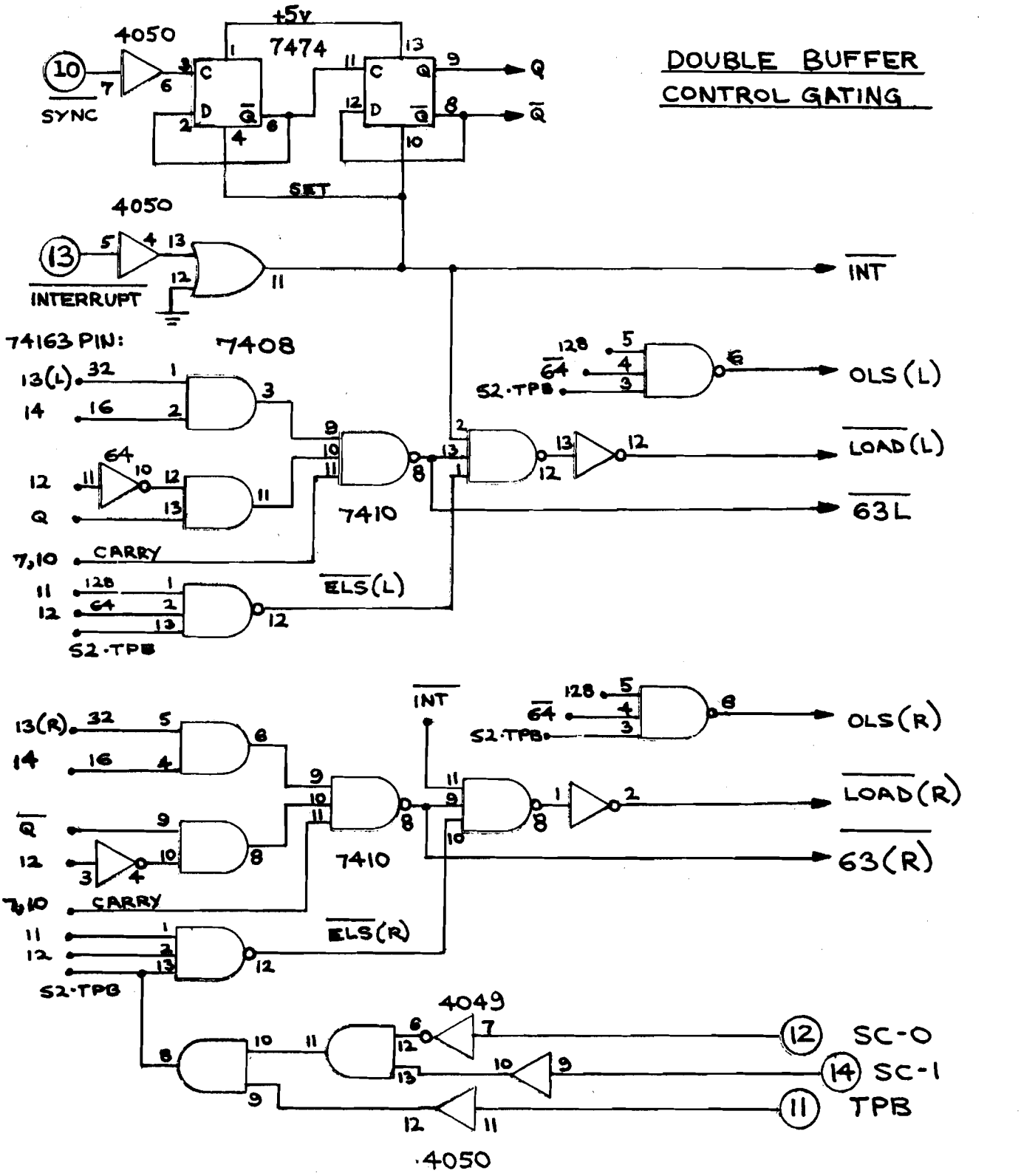
The only actual mod needed to the VIP hardware is to lift the 200 ohm video output summing resistor from ground so that the 1861 output signals will have enough voltage swing to drive the CMOS buffer gates. Standard 32X64 operation can be had either via the switch with the board plugged in, or via the normal VIP video output by re-grounding the 200 ohm resistor, with or without the card plugged in.

As indicated briefly in the accompanying waveform sketch, the TPB signal does not occur sharply coincident with the 1.76 MHz clock edges, as drawn so prettily on all the RCA data sheets for the 1861 and 1802. It happens delayed by at least half the period of the fast clock; furthermore, if you

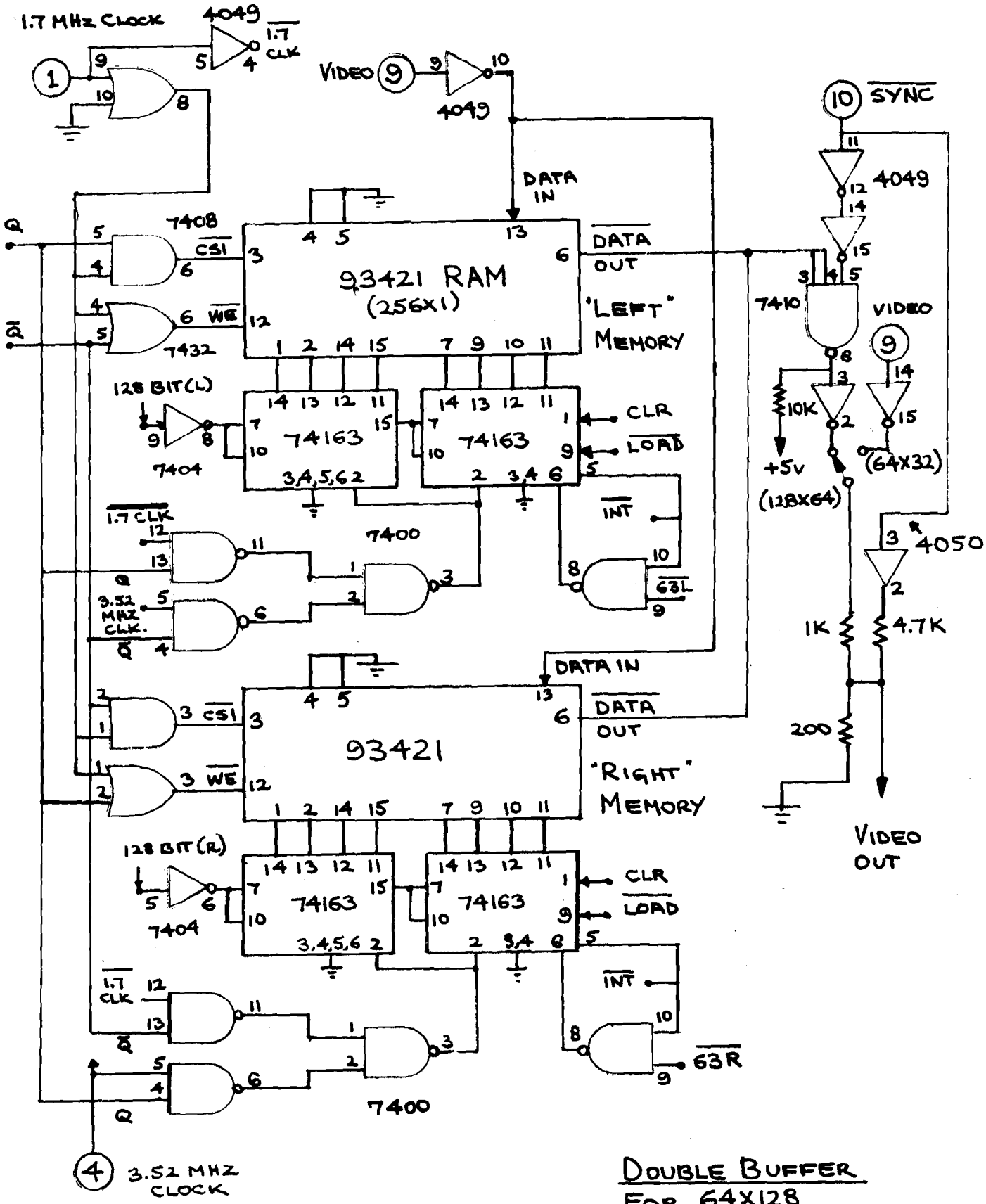
pore over the fine print of the data sheet, such a delay is apparently quite legal! The solution was to invert the 1.76 MHz clock fed (via gating) to the 74163 counters (using a 4049 section, shown at the very top of the drawing). Without this, the high-resolution picture is split by a dark verticle line, with the halves left/right swapped.

COUNTER CONTROL SIGNALS FOR 74163 ADDRESS COUNTERS

SYMBOL	DERIVATION	FUNCTION
$\overline{\text{LOAD}}$	$\overline{\text{ELS}}+63+\overline{\text{INT}}$	3 FUNCTIONS: 1. Just before frame start, when $\overline{\text{INT}}$=0, loads both counters to stop state (128), so both start at 00 on first occurrence of S2•TPB (start of first video line). 2. When and only when counter is in write mode (Q=1 for "left", $\overline{Q}$=1 for "right"), loads counter with stop state (192) following state 63. This ends 1st write line. 3. Only in write mode, loads counter with state 64 on first occurrence of S2•TPB when in state 192 (stopped). This starts 2nd write line.
(PINS 9, BOTH PKG)		
$\overline{\text{OLS}}$	$\overline{128 \cdot 64 \cdot S2 \cdot TPB}$	Sets counter to 0 on first occurrence of S2•TPB when in state 128. (PIN 1 is synchronous clear). This starts 1st write line & all read lines.
(ODD LINE START) PINS 1, BOTH PKG)		
ENABLE	$\overline{128}$	Stops counter whenever the most significant bit, with valve 128, is 1. This normally means 128 or 192. -Any state 128, entered at random or at power-up, will stop count to wait for ELS or OLS.
(PINS 7&10, LS PKG)		
$\overline{\text{ELS}}$	$\overline{128 \cdot 64 \cdot S2 \cdot TPB}$	See 3 under $\overline{\text{LOAD}}$ above.
(EVEN-LINE START) (TO PINS 9 THRU $\overline{\text{LOAD}}$ GATING)		


DOUBLE BUFFER
CONTROL GATING
+5v
4050
7474
SYNC
SET
INTERRUPT
INT
74163 PIN:
7408
7410
OLS(L)
LOAD(L)
63L
CARRY
ELS(L)
S2·TPB
OLS(R)
LOAD(R)
63(R)
ELS(R)
4049
SC-0
SC-1
TPB

17

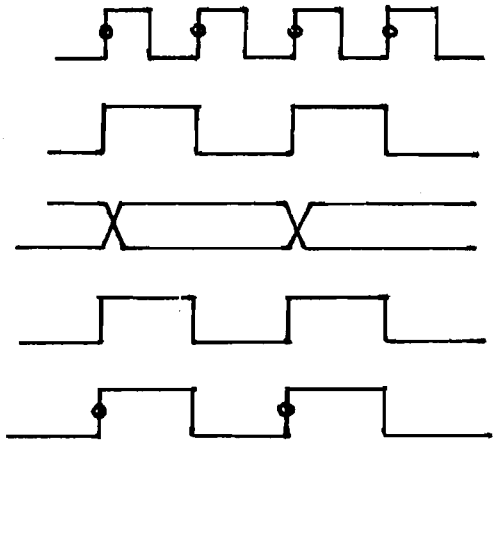


DOUBLE BUFFER
FOR 64X128
GRAPHICS
1861 · 1802

LINE NUMBER	Q	$\overline{Q}$	"LEFT" MEMORY & ADDRESS COUNTER FCN	$\overline{CS1}$	$\overline{WE}$	COUNTER (163) CLOCK	COUNTER STATES	"RIGHT" MEMORY & ADDRESS COUNTER FCN	CS1	$\overline{WE}$	COUNTER (163) CLOCK	COUNTER STATES
1	1	0	STORE 1861 OUTPUT	(1.7MHZ CLK)		1.76MHZ	0 TO 63	SUPPLY OUTPUT VIDEO	0	1	3.52MHZ	0 TO 127
2	1	0	(WRITE)	(1.7MHZ CLK)		1.76MHZ	64 TO 127	(READ)	0	1	3.52MHZ	0 TO 127
3	0	1	SUPPLY OUTPUT VIDEO	0	1	3.52MHZ	0 TO 127	STORE 1861 OUTPUT	(1.7MHZ.CLK)		1.76MHZ	0 TO 63
4	0	1	(READ)	0	1	3.52MHZ	0 TO 127	(WRITE)	(1.7MHZ.CLK)		1.76MHZ	64 TO 127

COUNTER STOP STATE	WHEN ENTERED	FOLLOWS STATE:	FOLLOWED BY STATE:	HOW RESTART?
192	END OF ODD LINES IN WRITE MODE ONLY	63	64	S2·TPB·(STATE 192)=ELS=1 MAKES $\overline{LOAD}$ = 0 (PIN 9)
128	END OF EVEN LINES IN WRITE MODE; END OF ALL LINES IN READ MODE	127	0	S2·TPB·($\overline{64}$·128)=OLS=1; $\overline{OLS}$ MAKES $\overline{CLR}$ = 0 (PIN 1)

Clock Phasing: (taking 1861 sheet literally; ie., shift video out on low - 2 - high Trans. of $\overline{CLK}$)

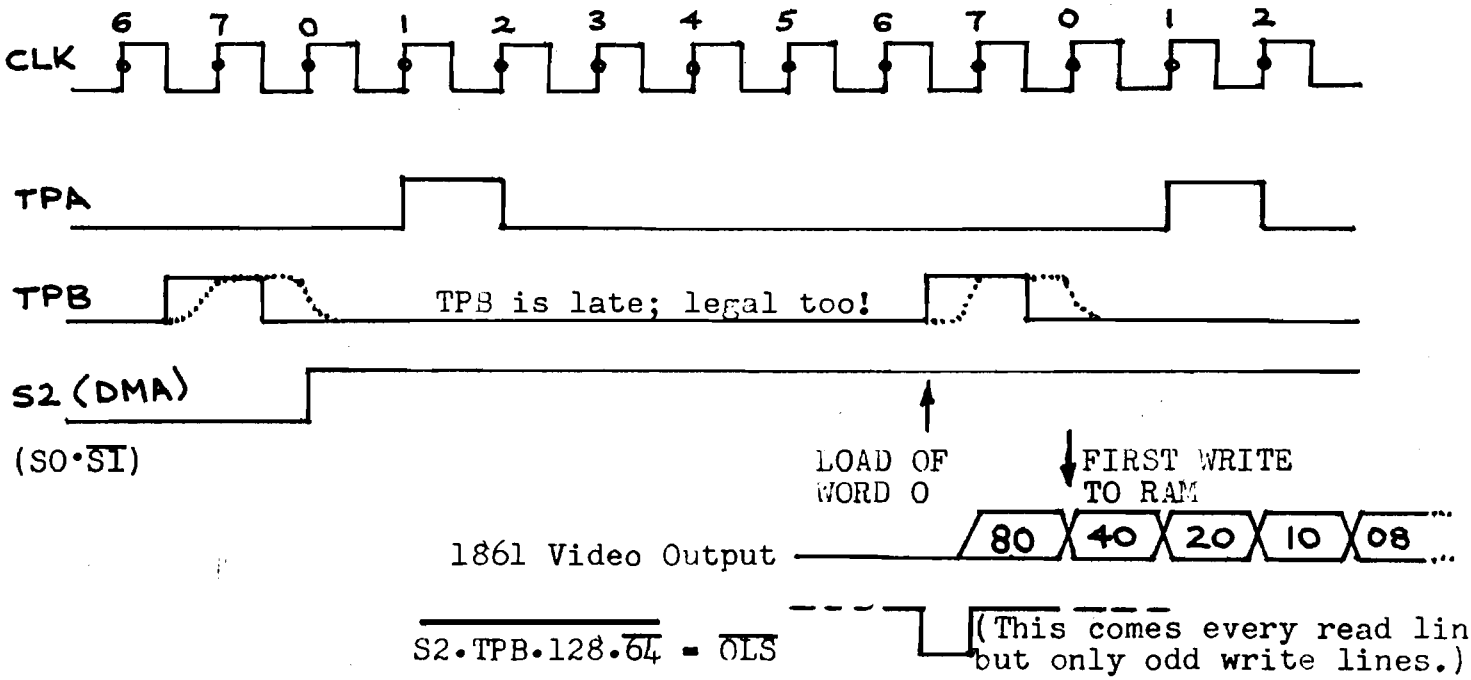


Clearly it looks like the 93421 $\overline{WE}$ and the 74168 clock should have the same polarity.

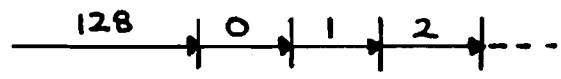
Best interpretation of data sheet is that this is also same as 1861 pin, ie., $\overline{CLK}$. If not, one inverter fixes it.

SEE BELOW!

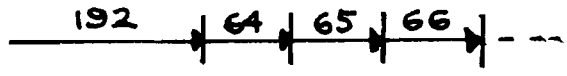
START OF LINE TIMING



Counter state, read mode and odd line write mode.



Counter state, even-line write mode.



ANSWER: Invert the clocks to the 74163 counters.

Event Timer

Malcolm Coyne
104-115 Cherryhill Blvd.
London, Ontario.

The fact that virtually all commonly used 1802 instructions execute in the same time facilitates the implementation of exact time delays using software. However, exact delays are not as easy to perform in software if the program must service variable length routines during the delay or if two or more different delays are to be timed simultaneously. It was this problem that gave rise to the simple event timer circuit.

This circuit allows the processor to load a time constant into the timer and then go and do any necessary housekeeping. The timer counts down to zero and then sends out an end-of-the-count signal. In my application--the control of stepping motors--I needed time intervals of 1/3 to 1/800 of a second and that is the reason for the ÷ 256 prescaler on TPB (0.22 MHz).

Circuit Operation:

U1 and U2 are loaded with the time constant from the data bus in response to the Out 6 instruction which also resets the counter/divider U3. U3 divides the incoming clocking pulses, in this case TPB, by the amount chosen by jumper J1. The divided clock is used to decrement the down counters U2 and U1. At a zero count U1 generates a borrow signal which is used to disable the clocking pulses and signal the processor. Clock remains disabled until the next time constant is loaded. Divider begins counting first clock pulse after loading.

Software:

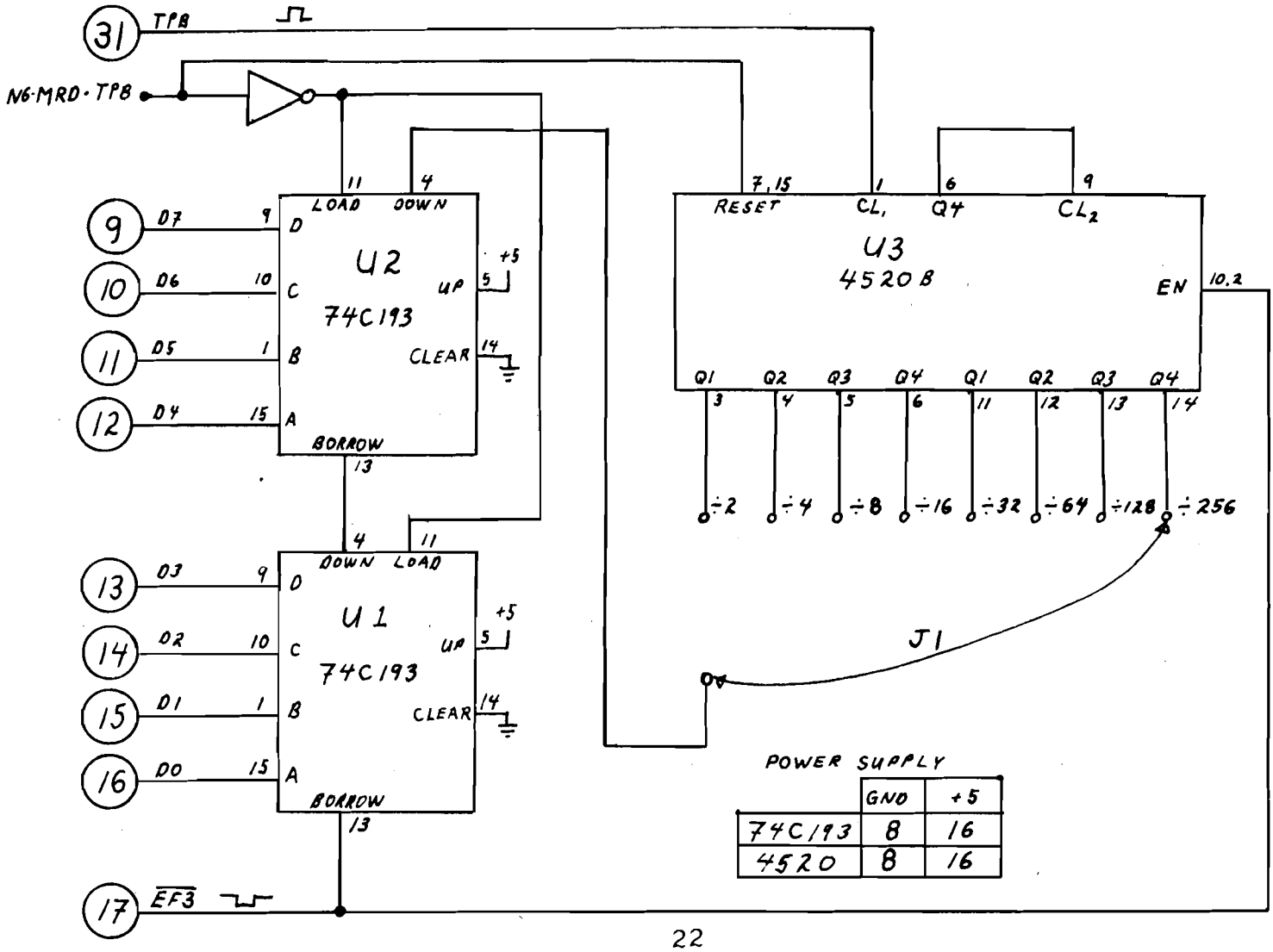
This will depend on the application, however, if a flag line is used to signal end of count, then any other processing should take less time than the minimum counter delay if the processor is to be able to determine the exact time of change of state of the flag. (i.e. processor should have adequate time after loading time constant to do all processing required of it and be waiting for the flag line to change.)

Sample Timer Program

```
00  7A          SEQ          -Turn Q off
01  F8 03 AA    LDI PLO      -Set constant to time 3
04  E3           SEX           timer delays.
05  66 FF        OUT 6        -Load t.c. in timer
07  2A           DEC
08  8A           GLO
09  32 0F        BRZ
0B  3E 0B        BN3          -Wait for end of timer
0D  30 05        BR            delay
0F  C5           LSNQ         -If Q is off skip
10  30 00        BR
12  7B           REQ          -Turn Q on
13  30 01        BR
```

NOTE: Program counter R(3)

This program on my system blinks Q at about 1 Hz with circuit as shown and system clock of 1.79 MHz.



CHESS TUTOR

Rev. Cyprian Rosen
St. Francis Seminary
Lafayette, N.J. 07848

The enclosed program, called "Chess Tutor" is not a chess program. It merely permits playing over previously recorded games or studying openings or end games. It is modelled on Tom Orr's "Attention Chess Buffs" in the December, '78 Kilobaud. Tom's program was written in Basic. Rather than try to adapt it to Tiny, I wrote it up in machine language. It will run in less than one page of memory.

As in Tom's program, the moves are entered in International Correspondence Chess Notation (cf. diagram). This is the only standard notation compatible with a hex keyboard. It is similar to algebraic notation, in that the squares are numbered from white's side and the moves are indicated in terms of the "from" square and the "to" square; e.g., white's P-K4 would be written 52 54, while black's P-K4 = 57 55. Castling is indicated by the King's move, e.g., 0-0 = 51 71; black's 0-0-0 = 58 38.

I am presently working on a graphics display for the program. I have the initial position (which takes 2 pages). It's not fancy, but you can't do very much with only 64 bits across. I'll send it in if I ever get it finished (spare time is hard to find.)

The program can easily be relocated. I start at location 0010, because a reset will then put me back into my monitor. This allows me to repeat the game, change sides with the computer, or imput a new game.

The program is pretty straight forward. The byte at location 0016 determines who moves first: #20 means the computer is white and moves first; #32 means the player is white. The Q light comes on whenever it is the player's turn to move.

The last "move" is always in the form #00XX. The computer keeps checking for #00 (end of game). It will then display #A0 if white has the advantage or has won; #0A if the advantage or win is black's; #AA if the game is even or a draw.

One important note: if the error display comes on after the 1st byte of the player's move, the wrong piece has been moved; if it comes on after the 2nd byte, the right piece has been moved, but to the wrong square. In either case, the player must re-enter the 1st byte.

While the program was originally designed for studying openings and end games, it lends itself to playing through complete games. I've included one game which, while not one of the greatest, is one that I have always enjoyed playing over. The player is white, so M(0016) = #32.

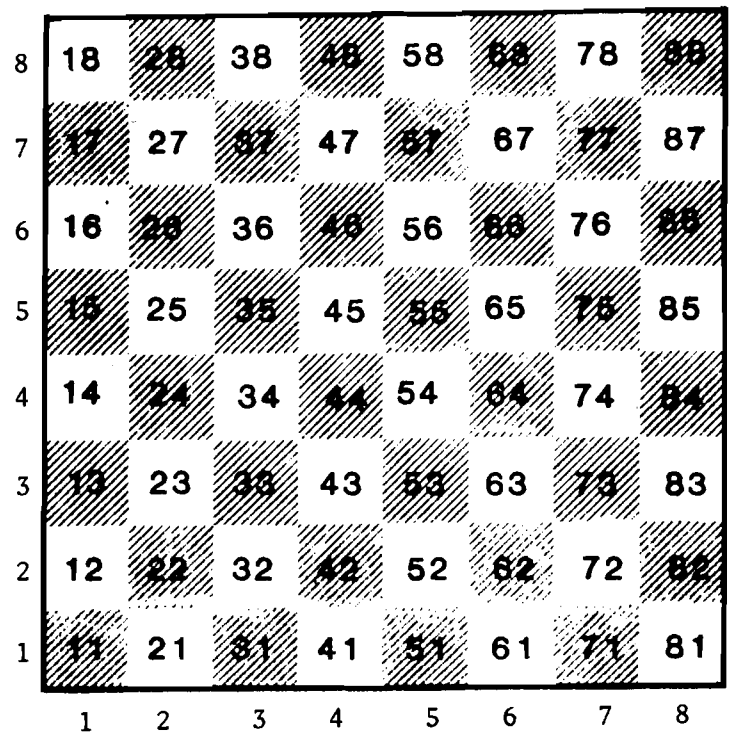
```
0010   F8 00          LDI #00      ..Zero high registers.
0012   B2             PHI R2
0013   B3             PHI R3
0014   B4             PHI R4
0015   F8 XX          LDI XX       ..#20 if Computer white
                                   ..#32 if Player white.
0017   A3             PLO R3       ..R3=Program Counter.
0018   F8 6E          LDI STACK
001A   A2             PLO R2       ..R2=Stack Pointer.
001B   F8 70          LDI MSTOR    ..Data begins at M(0070).
001D   A4             PLO R4       ..R4=Move Pointer.
001E   E2             SEX R2       ..R2=RX.
001F   D3             SEP R3       ..Begin play.
0020   3F 20    CPMOV: BN4 *        ..Wait for "IN" pressed.
0022   44             LDA R4       ..Get 1st byte of CPMOV.
0023   32 5B          BZ  ADVAN    ..Test for end. If D=0
                                   ..Go to ADVAN.
0025   37 25          B4  *        ..Else, wait for "IN" released.
0027   52             STR R2       ..Put it in Stack.
0028   64             OUT 4        ..Display 1st byte.
0029   22             DEC R2       ..Restore Stack Pointer.
002A   3F 2A          BN4 *        ..Wait.
002C   44             LDA R4       ..Get 2nd byte of CPMOV.
002D   52             STR R2       ..Get ready to display it.
002E   37 2E          B4  *        ..Wait.
0030   64             OUT 4        ..Display 2nd byte.
0031   22             DEC R2       ..Restore Stack Pointer.
0032   7B      PLMOV: SEQ          ..Q on=Player's Move.
0033   3F 33          BN4 *        ..Wait.
0035   44             LDA R4       ..Get 1st byte of PLMOV.
0036   32 5B          BZ  ADVAN    ..Test for End.
0038   24             DEC R4       ..Reset Move Pointer.
0039   6C             INP 4        ..Enter 1st byte of Player's
                                   ..Move.
003A   44             LDA R4       ..Put correct byte into D.
003B   F3             XOR          ..Compare them.
003C   3A 51          BNZ ERR1     ..If D≠0, go display EE.
003E   37 3E          B4  *        ..Else, wait.
0040   64             OUT 4        ..Display 1st byte.
0041   22             DEC R2       ..Restore Stack Pointer.
0042   3F 42          BN4 *        ..Wait.
0044   6C             INP 4        ..Enter 2nd byte of Player's
                                   ..Move.
0045   44             LDA R4       ..Put correct byte into D.
0046   F3             XOR          ..Compare them.
0047   3A 50          BNZ ERR2     ..If not correct, show EE.
0049   37 49          B4  *        ..Else, wait.
004B   64             OUT 4        ..Display 2nd byte.
004C   22             DEC R2       ..Restore Stack Pointer.
004D   7A             REQ          ..Q off=Computer's Move.
004E   30 20          BR  CPMOV    ..Next move.
```

```
0050   24        ERR2:   DEC R4          ..Reset Move Pointer.
0051   37 51     ERR1:   B4  *           ..Wait.
0053   F8 EE             LDI #EE        ..Get ready to show error.
0055   52                STR R2
0056   64                OUT 4          ..Display EE.
0057   22                DEC R2         ..Restore Stack Pointer.
0058   24                DEC R4         ..Restore Move Pointer.
0059   30 33             BR  PLMOV+1    ..Try again.
005B   37 5B     ADVAN:  B4  *          ..Wait.
005D   44                LDA R4         ..Get ready to display advantage
005E   52                STR R2         ..#A0=White adv. or White wins
                                        ..#0A=Black adv. or Black wins
                                        ..#AA=even game or draw.
005F   64                OUT 4          ..Display advantage.
0060   00        END:    IDL

006E   XX        STACK:

0070   XX        MSTOR:                 ..Enter data here.
```

The following "Center Counter Game" is taken from A. Horowitz, Golden Treasury of Chess, p. 196. The player is white; therefore, M(0016)=#32.

```
         W          B          W          B
0070   52 54      47 45      54 45      78 66
0078   21 33      66 45      61 34      45 26
0080   34 23      28 36      71 63      57 55
0088   42 43      38 74      82 83      74 85
0090   63 55      85 41      23 67      58 57
0098   31 75      57 46      33 54      46 55
00A0   62 64      55 44      11 41      44 53
00A8   51 71      36 44      41 51      44 52
00B0   51 52      53 52      67 85      52 53
00B8   61 63      53 44      85 67      26 45
00C0   32 33      45 33      22 33      00 A0
```

International
Correspondence Chess Notation



HORSE RACE PROGRAM

Guy R. Gilbert
304 Vassel
Drummondville, Que J2b 5H3

I am enclosing a horse race program that uses some of Ed McCormick's techniques (Tic-Tac-Toe, Pop. El. Nov '78) and requires 1K of memory.

The program starts at 0003 because the first three bytes are required for the Elf monitor. Two memory pages are used for the graphic (it gives a nicer picture). Two, three or four horses may run depending on the input at 002B. The input key is depressed repeatedly (009E) to advance the horses randomly until there is a winner. The winning horse's number flashes shortly then the horses come back to the starting gate. Memory required can be reduced by about one half if only one page is used for the display, only two horses run and the winning horse's number doesn't flash.

HORSE RACE PROGRAM

ADDRESS	OBJ CODE	COMMENTS
0003	F800BF	Initialization:
0006	F80AAFDF	RF: main
000A	F801B1B2	R1: interrupt
000E	B3B4B5BD	R2: stack
0012	F802B7B8	R3: delay sub.
0016	F803B9BA	R4: print sub.
001A	F8C5A1F8FBA2	R5: characters add.
0020	F8BAA3F87CA4	
0026	F8FFAD	
0029	ED69	Start TV
002B	3F2B372D	Enter number of horses
002F	6C642D	and display
0032	F849A7	
0035	F8A9A8	Starting addresses
0038	F809A9	of horses
003B	F869AA	
003E	F802B6	
0041	F800A6AC	
0045	8C5696	
0048	FB03324F	Clear screen
004C	163045	
004F	86FBFF3257	
0054	163045	
0057	0DFF02327F	
005C	FF013271	
0060	F803B6F868A6	Print starting positions
0066	F8A4A5D4	
006A	F869A6F8B2A5D4	according to
0071	F808A6F89DA5D4	
0078	F809A6F8B2A5D4	number of horses
007F	F802B6F8A8A6	
0085	F896A5D4	in race
0089	F8A9A6F8B2A5D4	
0090	F848A6F88FA5D4	
0097	F849A6F8B2A5D4	
009E	0DAC3FA6	Press input to
00A2	37A230AC	determine which
00A6	2C8C3AA0	horse will advance
00AA	309E	
00AC	FB0432BC	
00B0	8CFB0332CA	Which horse advances?
00B5	8CFB0232D8	
00BA	30EC	
00BC	9AB688AA6	
00C0	F8ABA5D4	Advance horse no.4
00C4	1A8AA6D4	
00C8	30F2	
00CA	99B689A6	
00CE	F8ABA5D4	Advance horse no.3
00D2	1989A6D4	
00D6	30F2	

ADDRESS	OBJ CODE	COMMENTS
00D8	98B688A6	
00DC	F8ABA5D4	Advance horse no.2
00E0	1888A6D4	
00E4	30F2	
00E6	97B687A6	
00EA	F8ABA5D4	Advance horse no.1
00EE	1787A6D4	
00F2	F804AB	
00F5	F802B6F857A6	
00FB	06CA0117	
00FF	F8B7A6	
0102	063A2D	Any winners?
0105	F803B6F817A6	
010B	063A43	
010E	F877A6	
0111	063A59	
0114	C0009E	
0117	F848A6	
011A	F8ABA5D4D3	
011F	F848A6	Flash no.1
0122	F88FA5D4D3	
0127	2B8B3A17	
012B	306D	
012D	F8A8A6	
0130	F8ABA5D4D3	
0135	F8A8A6	Flash no.2
0138	F896A5D4D3	
013D	2B8B3A2D	
0141	306D	
0143	F808A6	
0146	F8ABA5D4D3	
014B	F808A6	Flash no.3
014E	F89DA5D4D3	
0153	2B8B3A43	
0157	306D	
0159	F868A6	
015C	F8ABA5D4D3	
0161	F868A6	Flash no.4
0164	F8A4A5D4D3	
0169	2B8B3A59	
016D	C00032	Back to start
0170	0000000000	
0175	000000000000	
017B	DFE5F807AC	
0180	F0568CFF01AC	Print character
0186	327B1586	subroutine
018A	FC08A63080	
018F	08130808080831C	Characters: 1
0196	1C220204081033E	2
019D	1C22020C02221C	3
01A4	060A123E020202	4
01AB	00000000000000	blank
01B2	0303FC3C7C4281	horse
01B9	DFF823BE	Delay
01BD	2E9E3ABD30B9	subroutine

ADDRESS	OBJ CODE	COMMENTS
01C3	7270C42278	
01C8	2252F802B0	
01CD	F800A0C4C4	Interrupt routine
01D2	E230E220A0	
01D7	E23CDB80	
01DB	E220A0E2	
01DF	34DA30C3	
01E3	END	

MORSE CODE DECODER PROGRAM AVAILABLE

Cecil M. Bouie
P.O. 802
York, Penna. 17405

I have a Morse code decoder program for the ELF II, and for 1802 based computers that can run the same programs as the ELF II. The signals to be decoded can be taken from the speaker leads or the headphone jack. The hardware required to run this program is: an 1861 based video display; 1K of programmable RAM; and a very simple interface circuit. An object listing, along with a description of the interface can be obtained by sending $2.00 and a self addressed, stamped 41/8"X91/2" or larger envelope to: Cecil M. Bouie, P.O. 802, York, Penna. USA 17405.

FOR SALE

NRI Master Colour TV Servicing Course #4, copyright 1974 and 1975, including experimenter's chasis and possibly other items if desired.

100 5 inch reels of ¼inch acetate recording tape, used and in boxes.

Hewlett Packard dual beam oscilliscope model 132A with manual.

I will accept cash offers or trade of microcomputer equipment, preferably usable with RCA VIP. Eugene Fleming, 1327 Prairie Road, Colorado Springs, Colo., 80909.

FOR SALE: The following items are assembled and tested and in perfect working order. Elf II with several manuals and many newsletters, etc., $85; Netronics Giant Board $25; Netronics Kludge Board $10; 8 volt 4 amp Power Supply $15. Jim LaVeck, Route 1, Box 180, Dexter, New York, 13634, USA, 315-639-6383.

JOHN RUSTENBURG BROCK UNIV.

STEPPER MOTOR PGM FOR RCA 1802

USING THE TEC 1802 OR ANY 1802 MP WITH 256 BYTES RAM

Enter a four digit decimal number via an ASCII keyboard under program control and the stepper motors start to scream at a 2KHz pulse rate , providing more than 1/2 inch per second travel on a position control at better than 100 pounds thrust.

This best describes the stepper motor control which can provide a displacement of ten inches with a resolution of .001 inches and reproduce it using only 256 bytes of RAM with more than 20 bytes to spare. Thats enough to make my CPU want to crawl out of its socket.

The program is such that after initialization any entry up to 9.999 inches will result in the position of the carriage to match the entry and automatically moves in forward and reverse in relation to the last entry .It is allways a absolute distance to the reference.ie enter 2.477 and thats the distance from the ref.
enter 7.791 and thats the distance from the ref.
enter 0.009 and thats the distance from the ref.

Also to enter displacements of less than 1 inch simply depress the space bar instead of zeros ,it does the same thing .

Our carriage has two steppers one for each end and are driven in step. The shaft of the motor is directly coupled to the lead screw which has 20 threads / INCH . To drive two motors in this way you can use the same translator but must isolate the motors, this means let Q5 Q6 Q7 Q8 drive independent sets of output transistors for both motors; a parallel system.

If you have your own translator drive assy. then use only the gating switch portion of the ckt.(up to pins 4 ,5 of the 74192 chip) they will then go to the CW and CCW terminals and the strobe out DB7 may be disreguarded if you need the holding torque and dont mind the heat. (translator and power supply get hot)

Use buffers to interface translator and keyboard to processor.

The high data bit DB7 is used to stobe power to translator drive just prior to the Q out pulsing.

Data bit DB4 is used to gate the forward / reverse input to the translator. For up count DB4 is 0 , for down count DB4 is 1

The Q output is used to strobe the clock input on the translator Memory address DA can be changed to scale the distance moved for a given input (inches , mm etc.)

Stepping motors used were Superior Electric M062-F009
1.7 V 4.7 A per winding 65 OZ. IN. torque

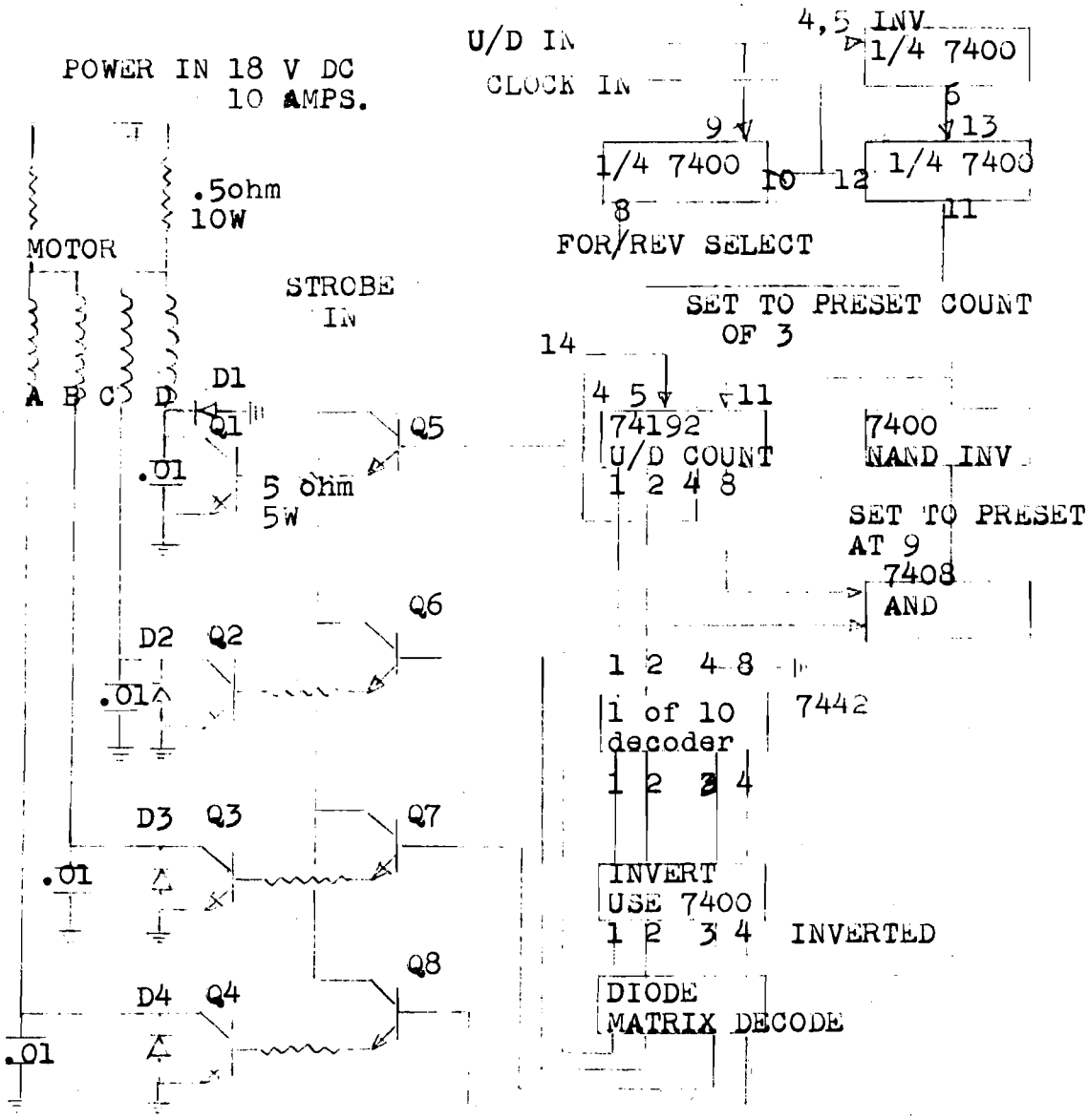
```
PGM TO READ ASCII KEY BRD

00   F81A      SUBR PTR
02   A3        PLO R3
03   F8FF      RAM STR @FF MSB
05   A4        PLO R4 READOUT
06   A5        PLO R5 STR
07   D3        SUBRT
08   F8FE      RAM STR @ FE
0A   A4        PLO R4 READOUT
0B   A5        PLO R5 STR
0C   D3        SUBRT
0D   F8FD      RAM STR @ FD
0F   A4        PLO R4 READOUT
10   A5        PLO R5 STR
11   D3        SUBR
12   F8FC      RAM STR @ FC LSB
14   A4        PLO R4 READOUT
15   A5        PLO R5 STR
16   D3        SUBR
17   303C      BRANCH TO MULT
19   D0        INITIALIZE SUBR PTR
          AND RETURN TO PGM KEYBRD
1A   E5        SEX R5  I/O
1B   351F      BR IF EF2 LOW
1D   301B      RET IF EF2 HIGH
1F   6C        READ
20   FB0F      XRI   XOR WITH OF
          TO INVERT ASCII DATA
22   FA0F      ANI   MASK OFF
          HIGH 4 BITS OF KEYBRD
          ENTRY
24   54        STR VIA R4
25   64     OUTPUT KEYBRD ENTRY
26   3D19      BRANCH IF EF2 HIGH
28   3026      RET IF EF2 LOW
PGM TO GET DATA@ FC FD FE FF AND
                    MULTIPLY

3C   F8F0      LDI F0
3E   A5        PLO R5 STR FOR DISPLY
         OF MULT PROD LOW ORDER
3F   F869      SUBR ENTRY
41   A3        PLO R3   PTR
42   F800      LDI 00 IN R6 LO HI
44   A6B6
46   F8FC      LDI FC DATA LSB
48   AA        PLO RA
49   F801      DECIMAL 1 IN R4 LO
4B   A4
4C   F800      LDI 00 IN R4 HI
4E   B4   NEED NOT REINITIALIZE R4
4F   D3        SUBR
50   1A        INC RA TO NEXT DATA
51   F80A      LDI DECIMAL 10
53   A4        PLO R4
54   D3        SUBR
55   1A   INC RA TO NEXT DATA
56   F864      LDI  DECIMAL 100
58   A4        PLO R4
59   D3        SURT
5A   1A        INC RA NEXT DATA
5B   F803      LDI DECIMAL 1000
5D   B4        PHI R4
5E   F8E8      LDI E8
60   A4        PLO R4
61   D3        SUBR
62   86        GLO R6
63   55        STR VIA R5
64   E5        SEX I/O
TEST FOR PRODUCT OF LOW ORDER
65   64        OUTPUT
66   307F      BR TO SUBTRACT RT
68   D0        INIT SUBR PTR
SUBR MULTIPLY 4 DIGITS
69   EA        SEX  ALU COMP
6A   86        GLO R6 PREVIOUS
               RESULT
6B   F4        ADD TO D REG
6C   A6        PLO R6 FINAL
               RESULT
6D   3B76      BR IF DF 0
6F   96        GHI R6
70   FC01      ADD IMMEA 01
72   B6        PHI R6
73   F800      PUT 00 IN D REG
75   F6        SHR  DF TO 0
76   24        DEC R4 MULT CONST
77   94        GHI R4
78   3A69      BNZ  TO EA
7A   84        GLO R4
7B   3A69      BNZ  TO EA
7D   3068      BR RET TO MULT

DECIMAL PRODUCT OF KEYBRD
ENTRY IS IN R6 LOW AND HIGH

7F   F8F2      DATA STR R8 @ F2
81   A8        PLO R8
82   86        GLO R6 TO SUBTRACT
               FROM R7
83   58        STR VIA R8
84   E8        SEX  ALU COMP
85   87        GLO R7 DATA IN
               REFERENCE POSN
86   F5        SD  SUBTRACT R6 R7
87   AA        PLO RA
88   96        GHI R6 TO SUBTRACT
               FROM R7
89   58        STR VIA R8
8A   E8        SEX  ALU COMP
8B   97        GHI R7 DATAIN REF
```

1802 STEPPER PGM

```
8C   75     SBD   SUBTRACT WITH
            BORROW
8D   BA     PHI RA  R6-R7 IS IN
            RA
8E   3B97   BR IF DF 0 ie LESS
90   F880   LDI 80 COUNT DOWN
92   55     STR VIA R5
93   E5     SEX I/O
94   64     OUTPUT 80 FOR COUNT
            DOWN ON STEPPER
95   30AD   BR
97   F890   OUTPUT 90 ON DATA
            BUSS TO COUNT UP
99   55     STR VIA R5
9A   E5     SEX  I/O
9B   64     OUTPUT FOR UP CONTR.
9C   9A     GHI RA
9D   FBFF   XRI TO FIND POSITIVE
DISPLACEMENT OF NEGATIVE RESULT
9F   BA     PHI RA TO STORE
A0   8A     GLO RA
A1   FBFF   XRI FF TO FIND DISP
A3   AA     PLO RA
A4   FC01   ADD IMMEAD 01
A6   AA     PLO RA
A7   3BAD   BR IF DF IS ZERO
A9   9A     GHI RA
AA   FC00   ADD IMMEAD 00 TEMP
AC   BA     PHI RA
AD   86A7   GLO R6 PUT IN R7 LOW
AF   96B7   GHI R6 PUT IN R7 HI
TO INITIALIZE AND PROVIDE REF
B1   F8D2   SUBR ENTRY Q OUT
B3   A3     PLO R3
B4   9A     GHI RA
B5   32BF   BR IF ZERO
B7   7B     SET Q
B8   C4C4   NO OP  DELAY
B9   D3     SUBR
BA   2A     DEC RA
BC   9A     GHI RA
BD   3AB7   BNZ
BF   8A     GLO RA
C0   32CA   BR IF ZERO
C2   7B     SET Q
C3   C4C4   NO OP  DELAY
C5   D3     SUBR
C6   2A     DEC RA
C7   8A     GLO RA
C8   3AC2   BNZ
CA   F800   RESET STROBE OUT
CC   55     STR IN R5
CD   E5     SET X FOR I/O
CE   64     OUTPUT
CF   3000   BR TO KEYBRD PGM

Q OUTPUT ROUTINE

D1   D0     SEP SUBR RET
D2   F816   LDI 16  DELAY MUST
BE SYMMETRICAL TO MA DF ENTRY
D4   A9     PLO R9
D5   29     DEC R9
D6   89     GLO R9
D7   3AD5   BNZ
D9   7A     REQ  RESET Q
DA   F813   TOTAL 20 COUNTS Q
            PER RA DECREMENT
DC   A4     PLO R4
DD   24     DEC R4
DE   7B     SET Q
DF   F816   LDI 16 DELAY
E1   A929   PLO DEC R9
E3   89     GLO R9
E4   3AE2   BNZ
E6   7A     REQ
E7   84     GLO R4
E8   3ADD   BNZ
EA   30D1   BR TO SET SUBR

STACK    F0  I/O  DATA
         F2  R8  ALU CALC

         FC  KEYBRD DATA
         FD
         FE
         FF       MSB
```

STEPPING MOTOR DRIVE AND TRANSLATOR FULL STEP

D1 TO D4 1N4002
Q1 TO Q4 2N4913 Q5 TO Q8 2N6043 DARLINGTON

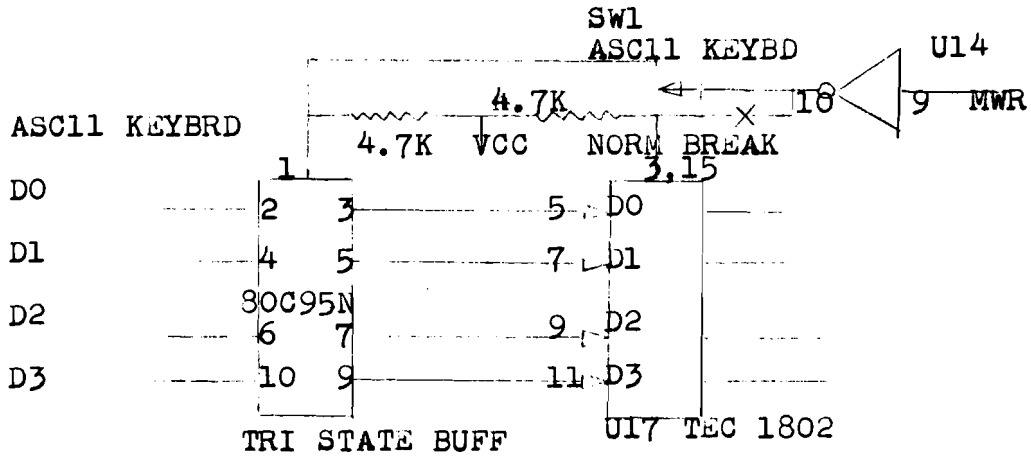


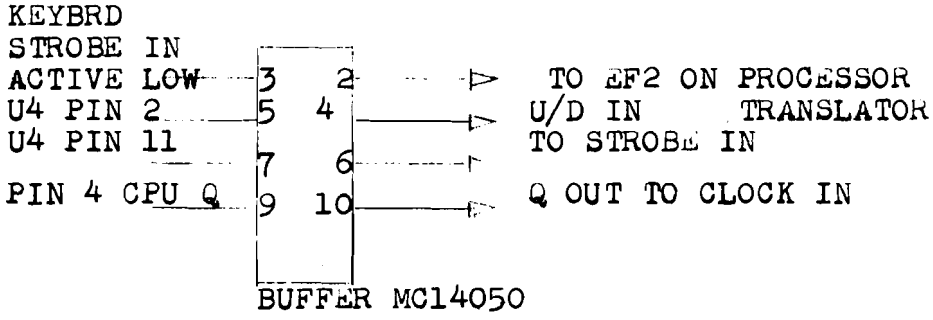
DECODE MATRIX TRUTH TABLE

	MOTOR WINDING A	B	C	D		
U/D COUNT 1	1	0	0	1	FOR	REV
2	0	1	0	1	↓	↑
3	0	1	1	0		
4	1	0	1	0		
1	1	0	0	1		

PIN 1,15 HIGH 9,10 LOW ON 74192
CLOCK IN TO Q OUT ON MP U/D IN TO DB4 STROBE IN TO DB7

ASC11 KEYBOARD INTERFACE





MC14050 PIN 8 GND PIN 1 VCC 80C95N PIN 8 GND PIN 16 VCC
PIN 15 GND

THE 4.7K RESISTORS TO THE 80C95 AND U17 HOLD EACH DEVICE IN HIGH Z STATE WHEN SW1 DOES NOT ADDRESS IT.
WITH SW1 IN KEYBRD POSN CONTROL FUNCTIONS BUT NOT DATA ENTRIES ARE ACTIVE.

U4 pin 2 U4 pin 11 ARE DB4 AND DB7 OF TEC 1802 OUT 4

To start program after data is entered put SW1 to KEYBD press RS and RUN enter 0000 to initialize R6 . Then press WAIT,RS, RUN and again enter 0000 to initialize R7 ,this is the reference posn. and wont change until the system is shut down. Now turn on the power to driver /translator ckt. and your running.

It may also be usefull to set up the ASC11 keyboard to enter data at any location in memory,edit,increment memory and output data under program control. If anyone is interested I will write the program

John Rustenburg
RR-2 Creek Rd.
Dunnville Ont. N1A 2W2

COMPUTER HOBBYISTS MUMBLE TO THEMSELVES A LOT

Mike Franklin
24 Duby Rd.
Acton, Ont. L7J 2P1

Once upon a time, a large number of people blew a lot of hard earned money on their own microcomputer, in the belief that men not machines made war, and that talking to an ELF was more rewarding than talking to one's wife. My wife, and possibly your spouse, thinks anyone who buys their own computer to be crazy, especially since that computer seemingly can only turn on a light bulb, a very small red light bulb at that, and can communicate in a language of only 91 two digit words, and, well, a new car would have been nice and it actually does something-useful.

I resent that, my computer does something useful, a lot of useful things, such as obey me - which is more than my kids sometimes do; it keeps me sane - except when I can't figure out why it is doing what it isn't supposed to do. It can add, subtract, and multiply, and sort of divide, (as long as the number isn't too big, or too small, and is an even number); it can balance my cheque book as long as the balance isn't over $327.67, which is usually isn't so it doesn't matter. Cec Williamson taught it to play Christmas carols, it plays as well as I sing, and I like it anyway. Ed McCormick taught it TIC-TAC-TOE and I can usually beat it, or turn it off if I can't, (which I can't do to my friends when they beat me). Tom Pittman taught it psuedo-English, and it communicates better than my neighbour's kids do; and it can spell, do arithmetic, work tirelessly, repetetively, and predictably, which is more than my employees do.

Now an enterprising chap in California has taught my ELF to be a VIP and now I have the programming capability to use the 1861 for what it was intended - graphics manipulation and display. As a bonus, for $5.00, RCA will sell me a book of 20 games with which to entertain my friends. Yes, my dear wife, my ELF is almost as good as the $69.00 Studio II, but it costs more.

TO VIP AN ELF, GAMES AND VIDEO MANIPULATION FROM RCA

program changes by Lynn Clock, article by Mike Franklin

The VIP software consists of an operating system which provides MEMORY READ/WRITE, CASSETTE I/O, VIDEO DISPLAY, and CHARACTER sub routines, and a HEX KEYPAD DECODING SCAN, as well as system initialization. The CHIP 8 interpreter is a system providing graphics manipulation sub routines. CHIP 8 utilizes the operating READ/WRITE subroutines. The $5.00 RCA manual #300 provides detailed instruction on the use of the two systems, and a listing of 20 pages. The games basically follow the theme of intercepting a program controlled object by a keypad controlled object.

To use the RCA programs, the OPERATING SYSTEM must be modified to use the hardware interfaces of your equipment. The following table summarizes the differences between the various 1802 systems:

FUNCTION	VIP	TEC	NETRONICS	QUEST
video chip on	69	69	69	61
video chip off	61	61	61	62
video status	EF1	EF1	EF1	EF1
HEX display	--	64	64	64
HEX pad input	62	6C	6C	6C
In switch status	EF3	EF4	EF4	EF4

The op system must further be modified to interface the hardware HEX keypad rather than utilize the VIP software scanning technique. The Cassette I/O was deleted since our own system's monitor provided these features.

PROGRAM CHANGES

Tables 1 and 2 list the operating and CHIP 8 program changes for the Netronics ELF II. To modify them for a Quest or TEC 1802, simply substitute a 61 for the 69 at M(0C70). This version utilizes 4K of memory, as identified on table 3. The programs may be compacted to a minimum of 2K by changing the high byte addresses underlined in tables 1 and 2 to the appropriate pages.

Suggested formatting is to zero your entire memory with the following program: 90,B7,F8,0B,A7,F8,nn,57,17,30,07, where nn is the value to be stored in memory, 00 in this case, and load the two programs at their respective addresses. Tape the entire program, 0000-0DFF.

OPERATION

Memory location 0002 must contain a 00 for operating system use or a 10 for CHIP 8 use. ELF II users can use C0 F0 00 C0 0C 00 C0 0C 10 and jump to the appropriate start address each time, 03 for operating system or 06 for CHIP 8.

Upon use of the operating system, the TV will display a 84 byte stack at the bottom of the screen, which is page 0B. Push 0, IN, and an address (ie. 0,IN,2,IN,0,IN) and the address and its content will be displayed below the stack. Push 0, IN to WRITE memory, or A,IN to READ memory. NOTE The IN button must be pushed EVERY time a byte is input onto the bus. Game commands in CHIP 8 also require use of the IN button, sometimes continuously, as in game 2.

VERIFICATION

Lynn Clock developed the above modification for his home-brew ELF. I have applied them to my Netronics ELF II with complete success. Eugene Tekatch has promised to try them on his system, and will report on his success later.

My friends and I have enjoyed the games developed by RCA for their system. While we have pirated their programs in a

sense, the development of a third language for 1802 users should help improve the popularity and use of our favourite micro. My thanks to Lynn for sharing his program modifications, and to RCA for the development of a fun language for my ELF.

Table 1
OPERATING SYSTEM CHANGES

```
0C00 F8 0C B2 F8 06 A2 E2 D2
0C08 F8 FF A1 F8 0B B1 30 28
0C10 F8 0C B0 F8 16 A0 D0 30
0C18 93 00 00 00 00 00 00 00
0C20 00 00 00 00 00 00 00 00
0C50 D3 90 B2 BB BD F8 0D B1
0C68 A5 F8 BA A7 F8 95 AC E2
0C78 D7 A6 D4 DC BE 32 8A FB
0C80 0A 32 85 30 83 DC 16 D4
0C88 30 85 D7 D7 D7 56 D4 16
0C90 30 8A 00 F8 0B BB F8 0A
0C98 B2 B6 F8 CF A2 F8 0D B1
0CA0 F8 46 A1 F8 00 B4 F8 1B
0CA8 A4 C0 00 14 00 E6 06 BF
0CB0 9C BE F8 CE AE 2A 1A F8
0CB8 00 5A 0E F5 3B C5 56 0A
0CC0 FC 01 5A 30 BA 4E F6 3B
0CC8 B6 9F 56 2A 2A D3 64 0A
0CD0 01 3F E7 F8 FF AF BF AE
0CD8 F8 0F BE EF 6C FA 0F EE
0CE0 F3 FA 0F 3A E7 15 15 D3
0CE8 3F E5 F8 FF AF BF AE F8
0CF0 0F BE EF 6C FA 0F EE F3
0CF8 FA 0F 3A E5 D3 00 00 00
0D90 35 90 30 82 D3 F8 FF AF
0D98 BF EF 3F 9A 37 9C 6C FA
0DA0 0F 52 E2 30 94 00 00 00
0DA8 00 00 00 00 00 00 00 00
0DB0 00 00 00 00 00 00 00 00
```

Table 2
CHIP-8 INTERPRETER CHANGES

```
0000 C0 0C ** 00 00 00 00 00
0008 00 00 00 00 00 00 00 00
0010 00 00 00 00 F8 01 B5 F8
0108 56 D4 F8 0D BC F8 95 AC
0128 D4 F8 0D BA 06 FA 0F AA
0130 0A AA D4 F8 0C BC F8 AD
0138 AC DC 30 53 00 F8 0C BC
0140 F8 D1 AC DC 30 A0 F8 0C
0148 BC F8 E8 AC DC 30 A3 00
0150 00 00 2A D4 00 22 86 52
0198 84 30 F2 5E 45 A3 30 3D
01A0 D4 30 46 D4 F8 F0 A7 E7
01F0 BA D4 F8 0F BE F8 FF AE
01F8 06 30 93 00 00 E0 00 4B
```

Table 3

CHIP-8 Memory Map

Location	Use
0000	CHIP-8 LANGUAGE INTERPRETER
01FF	
0200	User programs using CHIP-8 instruction set (2192 bytes available in 4096-byte system
0AA0	CHIP-8 stack (48 bytes max. for up to 12 levels of sub-routine nesting)
0ACF	
0AD0	Reserved for CHIP-8 INTERPRETER work area
0AEF	
0AF0	V0
0AF1	V1
0AF2	V2
0AF3	V3
0AF4	V4
0AF5	V5
0AF6	V6
0AF7	V7
0AF8	V8
0AF9	V9
0AFA	VA
0AFB	VB
0AFC	VC
0AFD	VD
0AFE	VE
0AFF	VF
0B00	256-byte RAM area for display refresh
0BFF	0BAC-0BFF - 84 bytes of op system stack

Register Use for CHIP-8 Interpreter

R0 - DMA pointer (page 0B for display refresh)
R1 - INTERRUPT routine program counter
R2 - Stack pointer
R3 - INTERPRETER subroutine program counter
R4 - CALL subroutine program counter
R5 - CHIP-8 instruction program counter
R6 - VX pointer (R6.1 must not be changed)
R7 - VY pointer (available for machine-language subroutines)
R8 - Timers (R8.1 - timer, R8.0 - tone duration)
R9 - Random number (+1 in INTERRUPT routine)
RA - 1 pointer
RB - Display page pointer (RB.1 - 0B)
RC - Available
RD - Available
RE - Temporary storage address pointer
RF - Available
0BB0-0BBF - Register high address storage
0BC0-0BCF - Register low address storage

KILOBAUD 1802 ARTICLES

M. Skodny
80 Weir St South
Hamilton, Ont. L8K 3A6

KILOBAUD (MICROCOMPUTING since January) published a series of articles on 1802 based systems:

- THE AMAZING 1802: D/A and A/D applications
  COSMAC can function as a scan generator, a digital voltmeter, or a function generator. August 1978

- INTERFACING THE ELF II
  HEX display for 1802, and a simple operating system (0000-003D). December 1978

- ONWARD WITH THE COSMAC ELF!
  Memory expansion, HEX display, multipage system. Operating system for 65K of memory. February 1979

- DOTS - PART I
  Software character generation in 1802 systems. (RCA 1861 Video Chip). February 1979

- PROGRAMMING THE 1802
  Basic techniques, how to input and output data, add, subtract and multiply. March 1979

- DOTS - PART 2 Will be published shortly.

TINY BASIC NOTES

Jim R. Smith
4629 North Shore Drive
Wichita Falls, Tex. 76310

The following applies to an ELF type computer (inputing from Hexpad to input port 4 and outputing to Hex readouts from output port 4) which is running TINY at 0100-08FF.

I have found three ways to lose track of "TINY'S program end plus stack reserve."

One way is to give the command "CLEAR". The second is to enter TINY'S cold start. Either of these will write 00 at 0900 and 0901. It doesn't "CLEAR" the memory of the program you wished you'd saved. It merely writes 0000 at the start. Writing in a low line number at 0900 and 0901 (like 00 01) will recover your "LOST" program---but what happens at 0024 and 0025 (TINY'S PGM end plus stack reserve)? well, it has reset itself and now reads 09 21.

The third way to lose the information is to record just the program space from 0900 to the end of the program area. Later on, when this program is "TACKED ON" to TINY already in memory, it isn't likely that the bytes at 0024 and 0025 will correspond to the "program end plus stack reserve" of that particular program anymore.

Who cares ? well - if all you want to do is run the program or just list it, it doesn't matter.

The rub comes when you want to change it. Then, the information at 0024 and 0025 becomes very important.

If you lost the information using method one or method two, or by method three by failing to write the information on the tape label of the "PROGRAM ONLY" you recorded---what then ? list it, write it down in longhand, cold start TINY (or warm start plus "CLEAR") and then type it back in ? not me!

There is a way to find it.

Apparently, "LINE NUMBER ZERO" is the key. The following program searches from 0900 on to find the first two consecutive bytes that are 00, adds an appropriate number to that address and outputs the address to the hex readouts that you'll need at 0024 and 0025.

It is not page conscious. It can be run up high in memory past the program, or you can run it on page 01, write the result down, reload TINY'S page 01 - change 0024-0025 to the correct values, enter the warm start and you're back in business.

When the program is run - the led comes on to indicate it has found two consecutive 00's in memory and it displays the HI byte you'll need at 0024. Push the input switch and it displays the LOW byte you need at 0025.

The program uses memory spaces 00FE and 00FF for its "WORKSPACE".

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

XX00 F8 09 BF F8 00 AF 4F FB 00 32 0D 30 06 0F FB 00

XX10 32 14 30 06 7B EE F8 FE AE F8 00 BE F6 8F 7C 20

XX20 AF 3B 27 9F FC 01 BF 9F 5E 64 3F 2A 37 2C 8F 5E

XX30 64 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

TEC-1802 EDITOR CORRECTIONS AND ENHANCEMENTS

Fred Feaver
105 Townsend Ave.
Burlington, Ont. L7T 1Y8

Some users of the TEC-1802 Editor program using the TEC-MB1 may have found that when they wanted to read the contents of MA0x69 (where X is page 1,2,3) they read "69". They may also have found that the low order address used in Edit, advance count or retrace could would always appear in location 69 of the page on which they were working.

This bothered me considerably and accompanied by the fact that I had had considerable trouble with my MB1 almost caused me

to give up using it.

I contacted Dan Carrigan of Tekatch Equipment Company and learned that he knew of the trouble and had a simple software correction. It is given below: See corrected Editor Program by Bernie Murphy IPSO FACTO #4, page 3) for master program and change it as follows:

Delete 0025 EE (as written) and retard all memory locations one position eg. MA0025 will now be DD

0026	"	"	"	6C
0027	"	"	"	BE
0028	"	"	"	DD
0029	"	"	"	6C
002A	"	"	"	AE
002B	"	"	"	EE (moved from 0025)

That is the total change and it corrected the fault.

TWO IMPROVEMENTS: The first recommended by B. Murphy and the second by Dan Carrigan are:

(1) Change MA 004E to B0
0051 to A0
0052 to D0 (This mod sets the P.C. to 0 for the target program)

(2) When using the editor program to enter new data there is no indication of where in memory the data is being deposited. Dan's suggestion is to make a simple software addition (given below) that, used with an 04 instruction, causes the next low order address to appear in the display.
See IPSO FACTO #4, page 32 and change MA006C to 3A74:
ADD the following steps

MA00 74	FC01	Add 01 to D
76	3A00	Branch to start if D is not 0
78	8E	Get last entry in RE.0
79	EC	RC = address register
7A	5C	Store last entry in RC
7B	64	Display "NEXT" address
7C	2C	DEC RC
7D	302E	GO TO 2E

To use this addition, "initialize" editor for EDIT--enter an address which you know you have used for data and enter 04 followed by input. The next address will appear in data display.

This is not a cure-all--You can find the last entry by successively entering "LATER" addresses until you find the address at which you stopped entering data or you can start at some address where the data is known and step ahead with an 01 input--but this suggestion does indilate the address, not the contents in an address. You can also step through your program using an 01 "INPUT" and without having to remember how many "STEPS" you have taken, you can enter 04 "INPUT" and find the next address to your present position.

LUBRICATING THE FLYING WOMBAT

INO ITALL
ANYWHERE, ONTARIO, FOX LCX

Load this program into your ELF or TEC-1802 and see what happens! You might build a "Flying Wombat" to test it and make programming fun. Try writing the program anyway.

The first step is to load the grease gun with a special grease call GOOSE #2. Second, you have a choice between one of two possible decisions. You must either press the blidget or turn the wingding. If you decide to turn the wingding, you must press the blidget next. On the other hand, if the blidget is pressed first, it will not be necessary to turn the wingding at all. Third, you must now insert the special grease into the brocket. If you have trouble inserting the grease you must press the blidget as in step two until the brocket is loaded.

The fourth step must check the wombat's trunk, vexit and reset the quack. These steps are in three parts and follow the third step. They are as follows:

A - press the quack reset
B - reset the vexit
C - close the trunk

A only outputs to the fifth step.
B outputs to fifth step and A.
C outputs to fifth step and A.
There is no relationship between B and C. When A and B and C are satisfied, the fifth step seals the aspt of the flying wombat.

MORE ABOUT HARDWARE BASICS

Fred Feaver
105 Townsend Ave
Burlington, Ont L7T 1Y8

An excellent article on "HARDWARE BASICS" will be found in November 1978 Issue of Radio-Electronics starting on page 45 under "Digital Reference Charts". This article goes quickly through the Base 10 and Base 2 numbering systems, Binary Words, Addition, Subtraction, Multiplication and Division. Boolean Algebra Rules, Law of Products, Law of Union, Law of Tautology, Law of Complements, Law of Double Negation, Flip-Flops, Nor Gate, Truth Table and Boolean Algebra as applied to all the basic gates (AND, OR, etc) are also presented.

The sheets are marked for cutting out of the magazine to permit their assembly into a private file.

ADDING A MATH FUNCTION TO YOUR 1802

Fred Feaver
105 Townsend Ave
Burlington, Ont L7T 1Y8

An excellent article on adding calculator functions to an 1802 will be found in Dec 1978, Jan 1979 issues of Radio Electronics. The MM57109, Number Oriented Microprocessor (Number Cruncher) put out by National Semiconductors, has all the functions of a microcomputer in a dedicated CPU and ROM. It eliminates the number crunching software usually required in a micro.

ADDING A MATH FUNCTION TO YOUR 1802 (CONT'D)

Features of the MM57109 are: RPN (reverse Polish Notation- the same system used in Hewlett Packard calculators), 1 to 8 digit mantissa, 2 digit exponent, four register stack, one memory location, trigonometric functions, logarithmic functions, e^x, y^x, π etc.

Error conditions are specified and if an error occurs an ECLR (Error Flag Clear) instruction must be executed.

The details shown include a 22 pin dual edge connector card which could make its use simple with the TEC-1802.

Material can be obtained from Questar Engineering Co., 50 S. MacDonald St., Mesa, Az. 85202. All as described in part 1.

Part 2 gives the detailed PC board layouts and discusses the programming.

Readout is not described but would be through a video terminal or repeated use of the data LEDS.

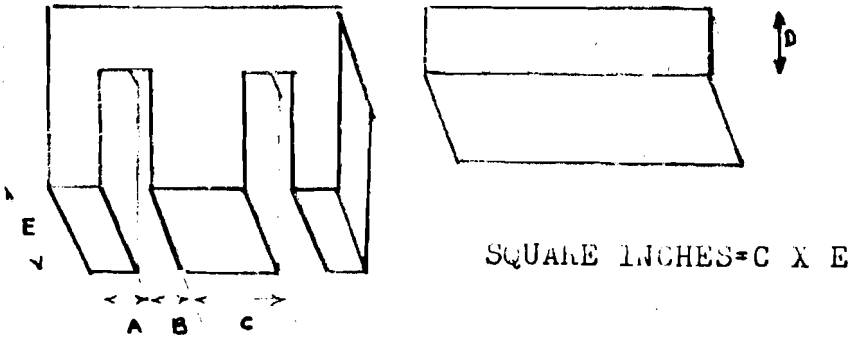
MAKE YOUR OWN POWER TRANSFORMERS

Ken Bevis
220 Cherry Post Drive
Mississauga, Ont L5A 1H9

I believe most experimenters have had some ideas at some time of building a power supply, but on finding the price of the power transformer, have abandoned the project.

There is no great mystery to power transformers; a few simple instructions will help you to make or to modify a transformer to suit the need.

1. THE CORE: consists of E/I shaped silicon steel in a laminated form, referred to as iron.



A = B = D
A + B = C
E = STACK HEIGHT

SQUARE INCHES=C X E

Core dimensions may vary but the relationships remain the same. Core dimensions are useful to know because a couple of measurements on the outside will tell you the complete structure. The important dimensions for our use are "C" and "E", for they tell the wattage capacity and determine the number of turns required to give efficient operation.

2. CONDUCTOR: various gauges are available from any electrical supply house (small amounts may be bought from an electric

motor rewinding shop or salvaged from a transformer). To select the proper size it is necessary to know the required amperage. Allow 500 circular mills (CM) per ampere in a confined space such as transformers. This data can be obtained from the copper wire table. Guage #23 will handle 1 amp. Further study of this table shows that an increase of 3 guage numbers doubles the circuler mill area. Thus, #23 carries 1 AMP, #20 carries 2 amps or #26 carries ½ an amp. If you require 4 amps, select guage #17.

3. INSULATION: normal magnet wire has insulation adequate for 200 volts, however some protection is required from the core and between windings. Plain cardboard will serve if the commercial product is not available. Spagetti sleeving is very useful for leads entering and leaving the winding, the sleeving can be slit about 1 inch from the end and caught under a few turns of the winding to provide some strain relief. As an alternative to the long spagetti, a piece of standard wire could be wrapped with the last 10 turns and the winding end then soldered to this lead.

4. PUTTING IT ALL TOGETHER: The core size is not a linear function of power handling capacity.
   1 square inch of iron will handle 40 watts
   2 square inches of iron will handle 100 watts
   3 square inches of iron will handle 250 watts

Winding turns are also based on the square inches of iron; use 5.5 turns for 1 square inch. If you double the core size then only ½ the turns are used, etc.

EXAMPLE A transformer with 110 volt primary and 7.6 volts secondary at 10 amperes.

(a) The core size = 7.6 x 10 = 76 WATTS: so lets assume 1½ square inches of iron.

(b) Secondary turns = $\frac{5.5}{1.5} \times 7.6 = 27.86$

this must be rounded out to 28 turns for 10 amperes the copper wire table says #13 or to make it easier use 2 #16 in parallel. (5 Amperes each)

(c) Primary turns are a direct ratio of the secondary using this equation:

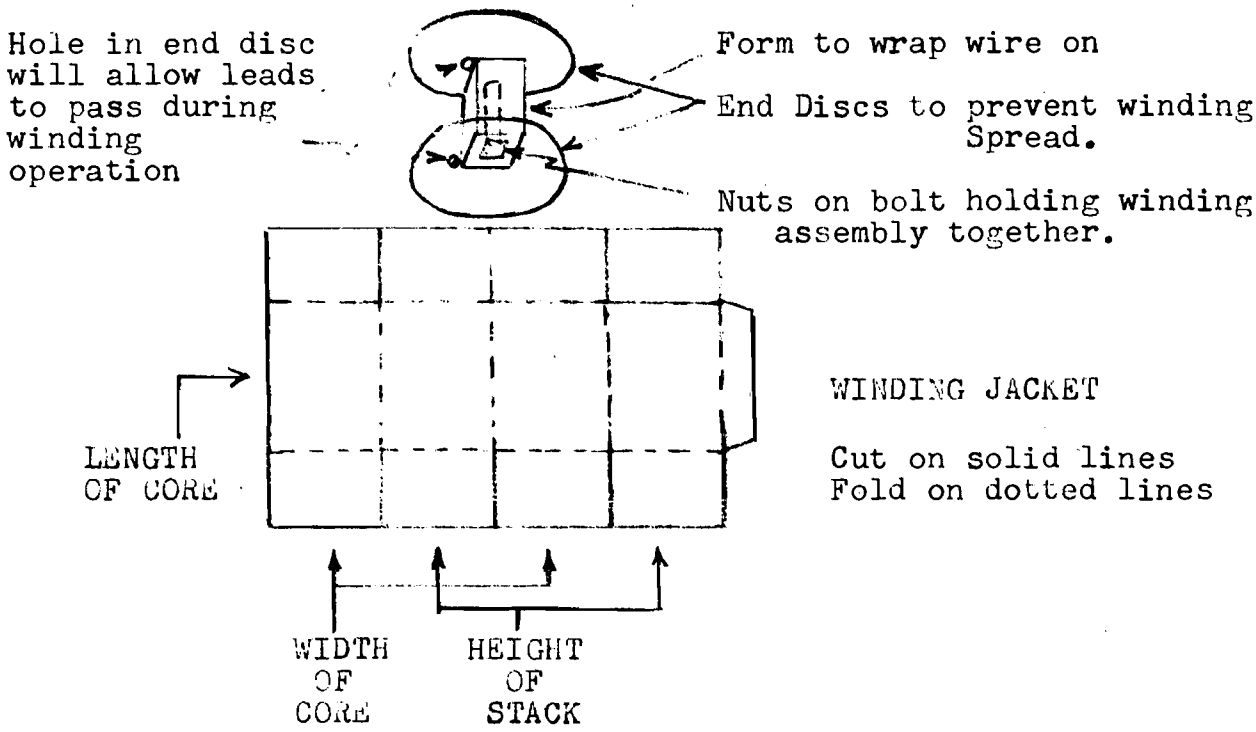
$$\frac{28 \text{ SEC TURNS}}{7.6 \text{ SEC VOLTS}} \times 110 \text{ PRI VOLTS} = 405.26$$

or 406 primary turns. Since this is a 76 watt transformer, the full load will be $\frac{76}{110} = .69$ amperes, the chart says #24

summary: transformer to deliver 7.6 volts at 10 amperes requires 1½ square inches of iron, with a primary of 406 turns of #24 wire and a secondary of 28

turns of #13 wire or equivalent.

(d) The winding should be wound using a piece of hard wood cut to the shape of the core but 1/16 inch larger where it slides over the core and 1/16 inch shorter where it fits between the core. A winding jacket should be cut from insulation (cardboard).



Wrap winding jacket around form and hold all tabs in place with small pieces of masking tape. Punch small holes with a nail to allow the lead wire to pass through the jacket. After the primary is wrapped, wrap a piece of brown paper to separate it from the secondary and then proceed. When all the windings are in place fold the sides of the jacket over all the way around and hold in place with masking tape.

When the masterpiece is dismounted and slid off the form, you then assemble the core laminations from each end, if any space is left, slide in a wedge to prevent the core from vibrating. Make sure that the two windings are properly identified before plugging into 110 volts or you might damage a good effort!

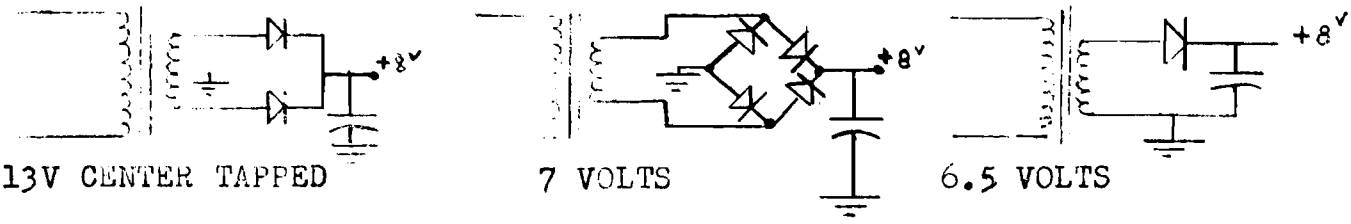
ADAPTING A TRANSFORMER TO YOUR REQUIREMENTS

Many times you may have a transformer in the junk box or the surplus store that can be adapted to a particular requirement.

EXAMPLE: A requirement for 5 volts regulated DC volts at 4 amperes.

ADAPTING A TRANSFORMER TO YOUR REQUIREMENTS (CONT'D)

You should start with a minimum of 8 volts DC and regulate down to 5 volts. More than 9 volts will generate to much unwanted heat. There are three methods to get 8V DC, don't forget an AC meter normally measures RMS.



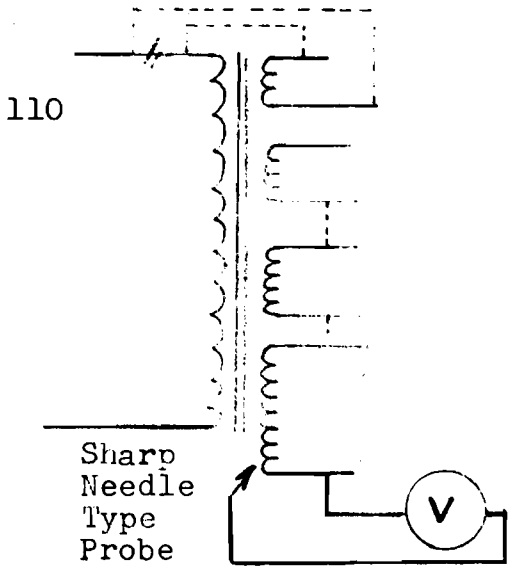
13V CENTER TAPPED — 7 VOLTS — 6.5 VOLTS

A B C

The output from the rectifiers accross a capacitor is equal to 1.414 times the RMS volts but the forward drop through silicon diodes must be taken into account on the lower voltage supplies. An extra .5 volt must be allowed in figure B as compared to A & C.

Power up your transformer and do a general voltage check to determine actual voltage. Also, check the guage of wire and refer to the wire table to check the circular mill area (see make your transformers) to determine the current available. Check to see if two windings can be joined in series to get your desired voltage.

If only a small change is required (2-3%) and an extra 5 or 6 volt secondary winding is available, it could be put in series with the primary.



Adding a small secondary in series to raise or lower volts on other windings. This winding should not exceed 6 volts!

Series Windings

If voltage exceeds the requirement having followed all above suggestions, TRY

with 1 end of meter connected to transformer winding, probe any visable turns on end of winding.

A further step would be to retain the existing 110V primary and install a new secondary that develops the required voltage and current. Some experimentation may be necessary to determine the actual turns per volt. If a 5 or 6 volt secondary exists and the turns are counted, then the "turns per volt" can be calculated.

ANNOTATED BIBLIOGRAPHY OF ARTICLES PERTAINING TO 1802 USES

Eugene Fleming
1327 Prairie Rd
Colorado Springs
Colo 80909

"Infinite UC 1800 Microcomputer", Radio Electronics, Aug. 1977, p.27
An "Equipment Report" describing both a wired and kit version of a trainer based on the CDP1802 microprocessor, manufactured by Infinite, Inc. Note: Nothing has been heard of this unit since the report.
Note: Radio Electronics examined for articles thru Sept. 1978

Wesibecker, Joseph A.; "COSMAC VIP, the RCA Fun Machine"; Byte Aug. 1977, p. 30
A detailed description of the RCA VIP system, including a good discussion of the CDP1802 architecture. Color photographs include a monitor and cassette recorder.
Note: Byte issues after this date not available for examination, but all prior to it were. This is the only article found.

Hutchinson, Thomas E.; "The Cosmac Connection"; 73 Amateur Radio
Part 1 in Jan. 1979, p. 102
Part 2 in Feb. 1979, p. 106
This pair of articles contains schematics, flowcharts and machine language programs for use of a machine using the 1802 as a high grade electronic keyer. Author used the ELF, but implies the information is applicable to other 1802 machines.

Hutchinson, Thomas E.; "Modify Your COSMAC ELF"; Kilobaud, Nov. 1979, p. 108
Describes addition of a hexadecimal keyboard to the basic ELF. Some support circuitry and beeper are included.

Feldman, Michael A. and Faye A. Dion; "Computerized Climate Control"; Kilobaud Microcomputing, Feb. 1979, p. 38
Includes schematics and program for controling thermostat in a seven day cycle using the ELF. With only some program modifications, seem to have potential for many other types of control applications.

Duntemann, Jeff; "Onward with the COSMAC ELF"; Kilobaud Microcomputing, Feb. 1979, p. 66
Some schematics and program ideas for Elf memory expansion.

Pittman, Tom; "DOTS"; Kilobaud Microcomputing, Feb. 1979, p. 84
Detailed discussion of software character generation with some program segments and flowcharts.

Cotter, Robert J.; "Programming the 1802" Kilobaud Microcomputing March 1979, p. 122
A beginners machine language tutorial at last!

QST and Ham Radio have been carefully examined in all issues to March 1979. No mention of the CDP 1802 or related equipment have been found in them.

A LETTER FROM THE MEMBERSHIP CO-ORDINATOR
MARCH 30, 1979

Wayne Bowdish
149 East 33rd St.
Hamilton, Ont.
L8V 1X3

Over the last year and a half of club existance the questions of "what does the membership want, or need?" and "how should ACE best supply these needs?" have arisen many times. Often, it has been claimed that ACE is not fulfilling the needs of the membership. I would like to make a few comments on the above claim.

Our newsletter consists of material supplied by members. Therefore, the contents of the newsletter reflects the interests of the active members. Active is used here to define members who are actually doing things with their micros, and who are willing to tell others about their experiences. What about the rest of the membership? The question "are we interested in these members and their contribution...or not?" has been put forth. The answer is yes, we are interested in these members but they don't contribute (other than their dues). The fact that a person does not contribute doesn't mean that there is no interest. I subscribe to several magazines which I have never, and may never, contribute to. I maintain my subscription because others contribute articles which I find interesting.

If members would like to see a different club or newsletter bias then they must contribute, either by supplying articles or by investing their time in the club executive. In regards to the newsletter, the policy of the present editor and his predecessor, has been to print any usable articles (ie. ones which can be read by our typist Diane York). We are not yet large enough, or rich enough to hire people to write articles. For those who have, as yet, not acquired the abilities to develop their own projects, all I can suggest is self study. There are numerous books on basic hardware, just go to your library. If you are weak in software, study the 1802 Users Manual and the simple programs in IPSO FACTO. A good book is Tom Pittmans "A SHORT COURSE IN PROGRAMMING" distributed by NETRONICS RESEARCH AND DEVELOPMENT LTD. Ask around, or send a letter of contract to IPSO FACTO. Maybe somebody in your area will be willing to help you out.

Any worthwhile endeavour requires work. In the beginning it may seem that the effort is excessive, but in the end, I'm sure that you will succeed and find micro computing an interesting and fulfilling hobby.

Before I close off this letter I'd like to comment on George Yorks comments at the October meeting about the loss of a large number of our previous years members. As of October, almost 240 members had not rejoined. Why? Well by the end of March 80 of these people, or about one third, had rejoined. We got many letters of the general form "I forgot about it" or "I didn't know it was time to sign up again." The remaining 158 people have not rejoined for several reasons. Probably some still do not realize that they have to send their dues for the 78/79 fiscal year. Others have written and said "loved my 1802 but now I'm going to move up to a Super Duper XYZ MARK III." Still others have stated that they thought micro computing would be a good hobby but, after their initial exposure, they decided that it wasn't quite what they expected. I've seen this happen to many people in many different areas. They think that photography or the stock market or whatever might be fun so they invest some time and money in the new hobby. After a period of time they lose interest and move on to another hobby. This is only natural and should be expected. Still other members took a course (and this is especially true of TEC1802 users) which was paid for by their employer. These people needed to get "up to speed" on micro computers. These people are now using micros in their work (probably not 1802's).

The above reasons, I believe, explain the majority of the "lost" memberships. What about the rest? To date I have only received one or two letters from people saying "your newsletter is no good so I'm not going to sign up for another year". I can only guess at the proportion of the 158 that this letter represents but I believe that it is a small part. What can we do for these few? I don't have any answers. All I can do is refer them back to my comments at the beginning of this letter.

In summary, I believe that we have an extremely good club and newsletter (the letters from members every month seems to prove this) and that we are supplying a useful function to our membership. Certainly the number of "lost?" members cannot be used as an indication of how well, or poorly, we are performing.

LETTERS TO THE EDITOR

Dear Mr. Murphy:

I recently received my tenth precious issue of IPSO FACTO and feel compelled to express my opinion in regards to a somewhat negatively critical opinion by Mr. Skodny.

Looking back through a few of the early issues I am reminded that this Newsletter was established on the basis of membership input. If the "unwashed brethern" have been forgotten, it has been by themselves. I should make it clear that I include myself in the ranks of the "unwashed brethern". I have been sitting back, collecting this great wealth of information for a skimpy ten dollars a year because I'm sure sometime in the future that the articles which are way over my head right now will be very helpful. It is my opinion that anyone who failed to renew their membership probably wasn't very interested in the first place.

As a newcomer to the world of microprocessors and Electronics I have neither a hardware nor software background and I depend on well documented articles such as those donated by the more experienced members of ACE to show me the way. Since all of the information I have used to get my micro to the state it's in today has been published somewhere before, I have nothing to offer in the line of a major breakthrough. I would, however, like to briefly describe what I have, with the thought in mind that maybe others have gone the same route as me and ran into a snag that they couldn't work out. I would be more than happy to help if they want to contact me. I know I found myself scratching my head many times.

I first got involved by building the ELF from Popular Electronics. It was (and still is) a perf-board and wiring pencil job. I used Popular Electronics articles to upgrade it to include 1K memory, Hex keypad, 1861 video, one page of battery protected RAM, Q-line speaker, and cassette interface. I bought the cassette tape machine that Radio Shack uses with their system. It's not very expensive and works fine. I modified my Zenith F4030W colour TV to have direct video. By using the manual for my set and an article on direct video in Jan 78 Electronics Today I managed to pull that little project off without a hitch.

Programs like Game of Life, and Mouse Trap from IPSO FACTO and Tic-Tac-Toe from Popular Electronics, to mention a few of the more interesting programs that I have been able to get running, were very useful to me by illustrating how programs are arranged to obtain a particular end result.

My ultimate goal is a complete personal computer system and to that end my most immediate plans include building a good power supply, adding more memory, and a TVT.

At present, the road ahead is very unclear and I for one, am counting on IPSO FACTO to keep up the good work. Sincerely,
Dave Robinson, 6097 Karen Ave., Newfane, N.Y. 14108, (716)778-5017

Dear Bernie,

Well today, just some small items:

1. There's always a bug (at least one). My issue on some thoughts on the call"... IPSO FACTO #9, has lost one vital instruction: INC RPC just in front of the last BR SEXIT, in case anyone wants to use it.
2. I'm beginning to fear that my SUPER ELF system has fallen into the ATLANTIC OCEAN. I have not yet received it, Quest has sent it, well it should be insured (Was that the right word?). If it really is lost, it will take quite some time to find out that it really is lost and then some time to get another. It's partly my own fault, I decided rather to wait a little longer with surface mail and use the "saved" money to get a little extra hardware.
3. FORTH. I've seen an advertisement from Diversified Technology, P.O. Box 465, 112 E. State St., Ridgeland, Mississippi 39157, phone (601)856-4121-556-4107 concerning systems with micro FORTH. The uP is called SCP1802, probably from Solid State Scientific. It seems rather expensive for personal use.
4. I've found out about a Swedish 1802 Club and have one issue of their newsletter. It's written in Swedish and seems hooked on the VIP. As our club seems to become the 1802 Club, I think you should send them an IPSO FACTO. Address: COSMAC Digital Group, Box 300, 76100 Norrtalje, Sweden.
5. How long can you all go on with the enormeous amount of work for the Club and the newsletter? It would be a pity, if you decided you can't go on anymore just for the fun of it. I would think it would be fair enough, if RCA, Quest, Netronics etc. would begin to contribute some. We shouldn't get dependent on them though. I fear (well not really) sooner or later the club has to put one or two of you on a payroll. What's your own opinion? All the rest of us can just hope you're able to go on.
6. Bernie once had an article about interrupts, enabling and disabling. It turned out, he had overlooked something, so everything works just as it should. I must admit I have this feeling that something should have been different about interrupts, enabling and disabling and the use of the T-register. All other instructions are quite obvious, but each time I want to use the MARK instruction, I have to look it up, just to see what it does exactly, and always I have to add another instruction either before or after just to make use of it. I have tried to figure out how to do a software interrupt. Here are some solutions, none really satisfies me.

(a) add a little hardware and then use: SEQ, REQ or INPx or OUTx, i.e. generate an external interrupt.
(b) use IDL and then when you get tired of waiting generate an external interrupt (with your switches)
(c)
```
MARK     that's a rather long sequence just to save
INC R2   the T register for the interrupt action.
DIS #21  Now, writing it down, I'm beginning to doubt
         it will work. Maybe it should rather be:
MARK, DIS #21, SEX R2, RET
```

Anyone with a really good solution?

I do miss the following 1 byte instructions: STX and DRX with obvious meaning instead of LBQ and LBNQ, LSQ and LSNQ; SAV, MARK, RET, DIS could then be C. instructions; IRX, IDL, STX, DRX could be 7. instructions. So much for wishful thinking. Honestly, I rather like the 1802 instruction set. I think in general its simple and easy to use compared to the 8080 "gang". The logic of its instruction set and register use is clearly one of the reasons why I choose the 1802. Another is the 1861, still another is the following: The 1802 is the most 16 bit word uP of all 8 bit uP I know of.

Extra: Unfortunately my typewriter has broken down and I've noticed you had retyped my last letter anyway.

7. IPSO FACTO KC standard

I'm feeling a bit guilty for responding so late to your requests for feedback. My problem: When I'm gonna start saving my file I'll not know how long its gonna be. What am I to do? I'll write a file header specifying an unknown number of fixed-length blocks, start every block with an extra byte or two telling this is a data-block, fill up the last block and end with a file trailer, telling how many blocks there should have been. Next problem: When I start reading the next data-block, the recorder will notoriously be placed right in the middle of a data-block. (As usual I was just a little late stopping it, so I had to rewind just a little.) I'm not even certain it's the next block. Each block (data, header or trailer) will have to start with a certain character sequence (Suggestion: at least 3 SYNC followed by SOH for header, and STX for data, EOT for trailer). A data-block contains a data block number for sequencing of blocks of a file. It sounds quite silly using control characters from the BSC (block synchronous) transmission protocols for asynchronous transmissions. Do I have a point? Does anyone know of a simple standard solving my problems? I don't feel capable of making a detailed suggestion at present.

Finally, IPSO FACTO is the best. Keep it going!

Volker Raab, Ramtenvy 30, DK8581 Nimtofte, Denmark.

Dear Editor:

My story starts by getting a year subscription to Byte, buying an ELF II from a friend 5 months later, and then reading all issues of your newsletter he happened to have on hand. The system consists of the ELF II, Giant Board, 4K memory, and an old Craig tape deck that has so much wow and flutter that the program tapes I have sound better to listen to than the Beatles!. (But it still works!!) I won't even mention my "terminal", but it's satisfactory.

I got "Life" going on Elf with little to no problems. (I use the on board monitor for clearing and storing patterns.) However, I will mention that the notes in the program in the first 30 lines, the ones concerning changing pages to make it work on non VIP systems, refused to go (Don't yell, I know it was me not the machine!) but when run as listed it worked fine.

My main problems are the fact that I'm in the Air Force stationed in Germany (220V 50HZ) causing problems with my video, and the complaint that the basic Elf looks like a Robby the Robot reject. How about an expando-cabinet! (Netronics listen up!)

Write if your interested in whats happening on this side of the "pond" with the 1802! (Please do tell us of any interesting news! Ed.)
Sincerely, Tom Creviston, Box 4029 52nd AMS, APO N.Y. 09123.

Dear Mr. York:

I have a Quest Super Elf with the 4K expansion board option. As to the software that came with the unit, I received a source listing of TINY BASIC. I decided instead that the TINY BASIC on ROM would cut down on storage required to operate since I only have 4K to work with.

My electronic knowledge is limited and I had someone else put the unit together seeing that it was too complicated. I received very little documentation and that is why I hope to gain a little insight as to the programming techniques required for the 1802 which is discussed in your newsletter.
Sincerely, Dennis Battocchio, 1315 Virginia Ave., Windsor, Ontario, N8S 2Z3

LETTERS OF CONTACT

My main interest in microprocessors is their use as a system's controlling element. My main interest is in hardware design and the low level language programming necessary to integrate the hardware devices into a system. (BASIC is too slow and cumbersome on a small system). I presently use my system to control a homemade step motor driven X-Y plotter table to which I am currently adding a low resolution image sensor to allow the table to be used to digitize patterns.

I would appreciate hearing from anyone whose interests fall into the same area of control as mine. Yours truly, Malcolm Coyne, 104-115 Cherryhill Blvd., London, Ontario

I would like to hear any comments on interfacing the TVT-6 to the 1802. I am currently trying to interface to the Netronics ELF II. With the notes in the #3 issue of IPSO FACTO, I believe that it makes enough sense (I'm new at this) but anything is helpful.

I haven't seen any article that just came out and said, "hey, this is exactly the way it's done"; so if I come up with one that works, I plan on submitting it as an article to IPSO FACTO (unless you beat me). Thank you, SSGT Larry L Garrison, FR 519 58 6481, PSC Box 2713, APO New York, NY 09125, USA

CALL FOR NOMINATIONS FOR 1979-1980 ACE EXECUTIVE

The election of a new set of executive members for the club will take place at the Annual General Meeting (Tuesday, May 8, 1979). The continued existence and future success of the club depends on getting a full slate of executive members. The old saying "Many hands make light work" still holds. There were a number of positions that were not filled this year. Wayne Bowdish was "railroaded" to serve as membership co-ordinator because there were no nominations or volunteers for the position. He is doing a good job but, if this type of situation continues, the club will "die on the vine". OKAY, MEMBERS IT'S UP TO YOU. Send your nominations to TOM CRAWFORD, 50 Brentwood Drive, Stoney Creek, Ontario, L8G 2W8 (662-3603).

MANUALS REQUIRED

Fred Feaver
105 Townsend Ave
Burlington, Ontario L7T1Y8

I have acquired 2 video terminals which are inoperative now but should be repairable if the proper service data can be acquired. I require the schematic diagrams and technical manuals (I have the operator's manual) for the CONRAC 401 MARK II COMPUTER TERMINAL and the Technical manual for the DATAPOINT 3300 terminal.

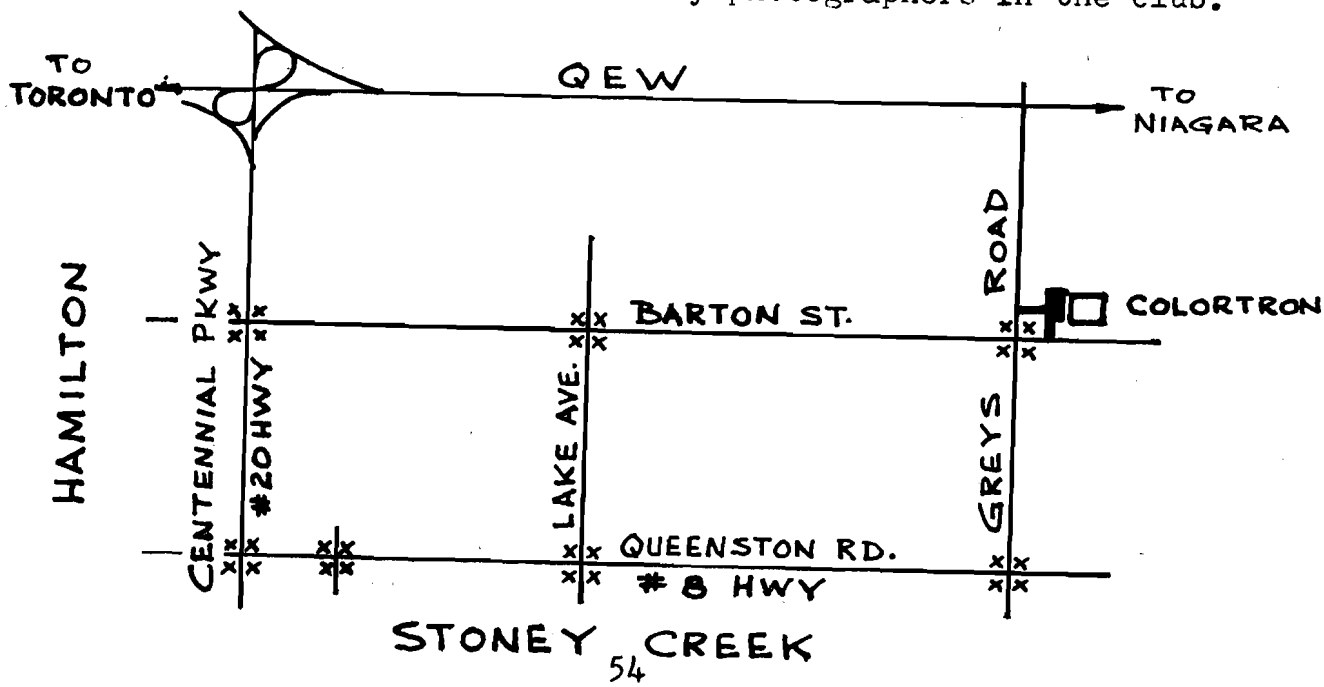
I will pay a nominal fee for a good copy of the above information or will copy and return promptly.

COLORTRON TOUR

May 22, 1979, 7:30 P.M.
Colortron, 340 Grays Rd., Stoney Creek

Colortron is an excellent example of a modern film processing facility. Among the features is a Kodak computer controlled colour print filter/exposure system and a film development process using a microprocessor to control chemical quality.

This should be an interesting tour for computer/microprocessor enthusiasts and also any photographers in the club.



THE ASSOCIATION OF COMPUTER EXPERIMENTERS MINUTES OF CLUB MEETING 79-3 HELD AT STELCO WILCOX ST. AUDITORIUM 13 MARCH, 1979, 8:00 P.M.

79-3-1 The meeting was preceded by a tutorial.

79-3-2 Motion to adopt Minutes 79-1 and 79-2 as included in Newsletter Issue #9 and Issue #10.
Proposed - John Morris
Seconded - Ken Bevis
Carried

79-3-3 George York reported a current bank balance of $2945.45 and a paid membership of 455.

79-3-4 George York reported on the T-shirt/Logo investigation. T-shirt prices ranged from $2.50 to $4.00. T-shirts would involve a large amount of handling and logistics problems -- eg. size, colour, mailing, customs, finances, etc. An alternative suggestion is to have the Logo produced in an iron-on transfer. The cost would be $0.20 per transfer. At that rate the club could afford to include a free transfer for each member. After discussion it was decided to have a dark blue iron-on transfer.
Motion to approve the purchase and distribution of iron-on transfers rather than T-shirts.
Proposed - George York
Seconded - Mike Franklin
Carried unanimously.

79-3-5 The tutorial group has completed design/development of a cassette interface. The club will provide interim financing to enable a set of 16 cassette interface kits to be put together. The kit cost will be less than $30.00. The total cost per kit will cover all costs incurred.
Motion that the cassette interface be offered as a complete kit including the P.C. board.
Proposed - Glen Simpson
Seconded - Bill Reed
Carried

79-3-6 The meeting concluded with a flea market/discussion period. About 30 people attended the meeting.

NOTICE OF CLUB MEMBERSHIP RENEWAL & NEW CLUB DUES ASSESSMENT

The club dues assessment for the 1978-79 fiscal year is $10.00. Make cheques payable to the Association of Computer Experimenters.

If you know of anyone who would be interested in joining our club, why not give him/her the membership application. We are constantly looking for new members with new and interesting ideas.

MEMBERSHIP APPLICATION FOR THE ASSOCIATION OF COMPUTER EXPERIMENTERS

FIRST NAME | LAST NAME

FIRST LINE OF ADDRESS

SECOND LINE OF ADDRESS

CITY & PROVINCE | POSTAL CODE

Association of Computer Experimenters

MEMBERSHIP RENEWAL FORM

ATTACH MAILING LABEL HERE

FIRST NAME | LAST NAME

FIRST LINE OF ADDRESS

SECOND LINE OF ADDRESS

CITY & PROVINCE | POSTAL CODE