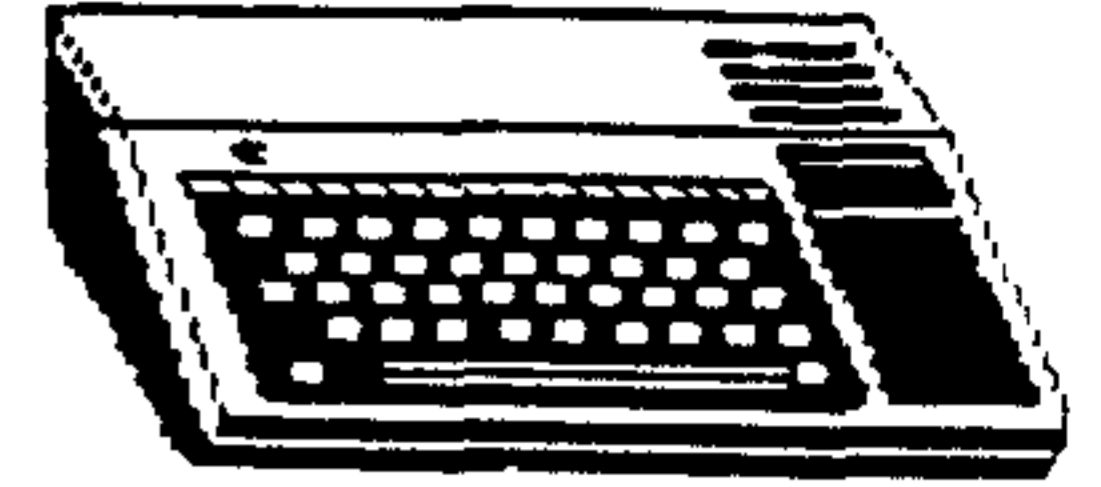


CENTRAL OHIO

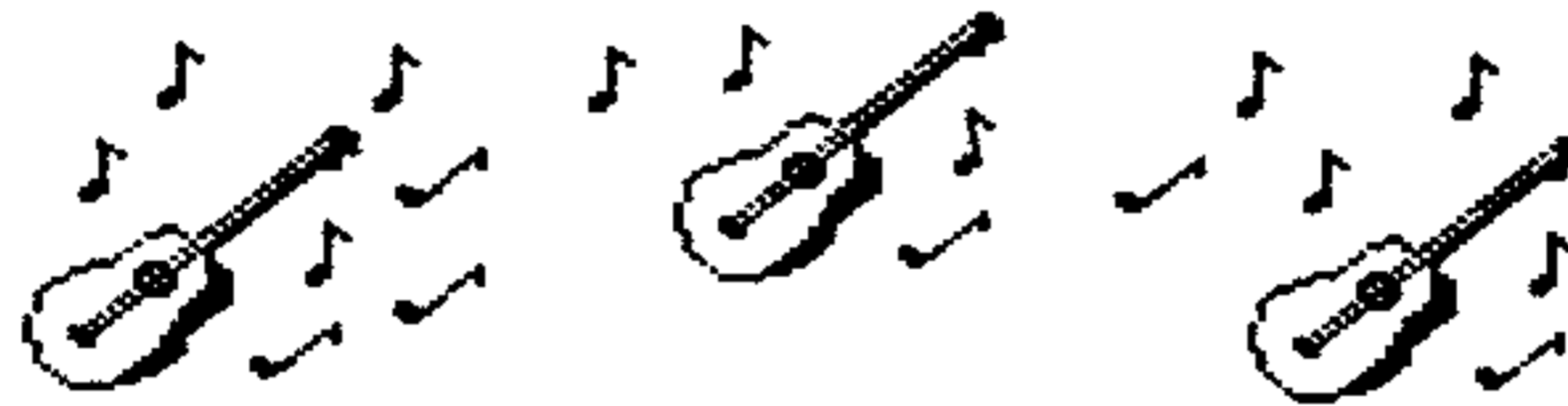


Spirit of 99

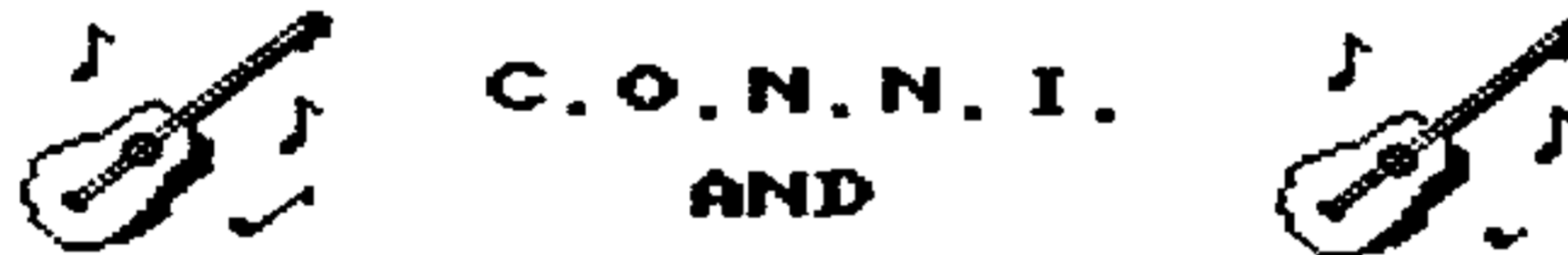
NINETY NINERS INC.

THE OFFICIAL NEWSLETTER OF THE CENTRAL OHIO NINETY-NINERS INC.

PUBLISHED MONTHLY IN COLUMBUS OHIO



HAPPY
BIRTHDAY



C.O.N.N.I.
AND
SPIRIT OF 99
ARE 10 YEARS OLD
THIS MONTH



COPYRIGHT C 1985
 Central Ohio Ninety-
 Niners Incorporated
 (C.O.N.N.I.). Colum-
 bus Ohio 43212, USA.
 All rights reserved.
 Spirit of 99 is pub-
 lished for Central
 Ohio Ninety Niners
 Inc. by C.O.N.N.I.
 members and is the
 official newsletter
 of C.O.N.N.I. User
 Group.

Editorial, address
 is:
 181 HEISCHMAN AVE
 WORTHINGTON, OH 43085
 Subscription rate
 (USA) \$20.00 /1 year
 \$25.00 /1 year out-
 side continental U.S.
 Third class postage
 paid at Columbus, OH
 CHANGE OF ADDRESS:

Send both OLD and
 NEW address to add-
 ress listed above.
 We assume no respon-
 sibility for manus-
 cripts, programs (tape
 or disk) not accomp-
 anied by return pos-
 tage. Letters to the
 Editor are property
 of Spirit of 99. If
 published, we reserve
 the right to edit at
 our discretion.

OPINIONS EXPRESSED
 HEREIN ARE THE AUTH-
 ORS AND ARE BASED ON
 VALID DOCUMENTABLE
 RESEARCH, THEY DO NOT
 NECESSARILY REFLECT
 THE OPINIONS OF THE
 PUBLISHER.

We will not know-
 ingly publish copy-
 right material with-

out the permission
 of the author and
 credit due.

All programs pub-
 lished herein are of
 public domain unless
 otherwise noted.

Other non-profit
 user groups may use
 material from this
 newsletter only if
 source and credit is
 given.

Central Ohio Nine-
 ty Niners Inc. is a
 non-profit organiza-
 tion comprised of ME
 MBERS who own or use
 the TI99/4A computer
 and it's related pro-
 ducts and have paid
 a yearly membership
 fee of \$30 and whose
 main objective is
 the exchange of Edu-

cational and Scient-
 ific information for
 the purpose of comp-
 uter literacy.

C.O.N.N.I. meetings
 are held the 3rd sat-
 -urday of each month
 at Chemical
 Abstract, 2540
 Olentangy River Road
 Columbus, OH. Meet-
 ing time is 8:30 AM
 til 2:30PM, Meetings
 are open to the pub-
 lic. Membership dues
 (\$20.00) are payable
 yearly to C.O.N.N.I.
 and cover the immed-
 iate family of the
 member. Please send
 check to our member-
 ship registrar and
 join C.O.N.N.I.
 Please address it to:
 John L. Parkins

2215 Bayfield Drive
 Columbus, OH 43229

INDEX

	PAGE
ABOUT THE DOM.....	13
ASSEMBLY LESSON 6.....	10
CONNI IS TEN.....	04
CONNI CALENDAR.....	03
CONNI MINUTES.....	02
DISK DRIVES.....	14
FROM THE TEACHERS DESK..	06
JUST A LITTLE BIT HELPS.	06
MUSIC - MUSIC.....	09
RIBBONS AND INK.....	07
STOR MOR-REVIEW.....	07
TI WRITER FONT MAKER....	09
WHAT IS AN IC.....	08
WORD PROCESSING WITH....	
MULTIPLAN.....	05

C.O.N.N.I. MINUTES

MINUTES OF C.O.N.N.I. MEETING Saturday 19 Sept 1992

President John Parkins conducted the meeting and read letters from various out-of-town members who had sent various amounts to renew their membership due to our recent changes in dues. It was decided to pro-rate such memberships.

Upon request of Dick Beery, it was agreed to allocate funds from the Clearing House BBS to mail out copies of Steve Burns' new Simulator tutorial program to all Clearing House members. A motion was made and passed to transfer funds to/from Clearing House and general accounts as needed.

A special thanks is due to Steve Burns for all his work in writing his Simulator, which is intended to teach those not familiar with BBS operations how to use the Spirit of 99 BBS and Clearing House BBS.

Bud Wright reported that he had uploaded his assembly music, ported over from IBM, with a request for a donation to C.O.N.N.I., to Delphi. There have been 400 downloads, out of which we received one donation, from Walter Ward. The members voted to thank Walter for his donation.

Chuck Grimes announced the contents of the Disk of the Month.

Jean Hall announced the results of her committee study to determine which user groups we would continue to send newsletters to, and read the list of 14 groups. After discussion, it was decided to drop F06 from the list.

Jim Peterson offered to resume mailing a flyer, together with his Tigercub Software orders, to promote C.O.N.N.I. out-of-town membership and D.O.M., and offered to include a promotion for the Clearing House BBS if problems with giving members access are resolved.

Next month's meeting will be our 10th anniversary meeting, for which we will return to the Martin Janis Senior Center since Chemical Abstract will not be available. Dick Beery offered to form a committee to provide refreshments. Plans were made for equipment to be brought, previous members to be invited, etc.



After the meeting, Jim Peterson demonstrated some of his music which he has converted from XBasic to SNF for use with Midi Master 99.

Respectfully submitted
 Co-secretary Jim Peterson

C.O.N.N.I. CALENDAR

October 1992

SUN MON TUE WED THU FRI SAT

				1	2	3
4	5 COLLIER'S DAY 	6	7	8	9	10
11		13	14	15	16	CONNI MEETING
18	19	20	21	22	23	24
25	26	27	CONNI MEETING	TRICK OR TREET	30	

SATURDAY MEETING 17 OCT 1992

Janis Center -- Columbus

This is a special meeting in celebration of our TENTH year of existence. Bring your friends and previous members that you know. Give them a call! We would like to see a big turnout for this event. 7:30 AM will be allocated for various setups of equipment. 8:30 AM Refreshments will be available.

Demos will be presented:
 Quadrophonic sounds - Chuck Grimes
 Caves of Grog - Dave Truesdale
 4A Flyer - Dave Truesdale
 Digitized pictures - Bud Wright

Other demonstrations may also be presented

WEDNESDAY MEETING -- 28 Oct 1992

McDONALD'S -- Cleveland and Main -- Westerville

7:30PM MEETING TIME

Lets have a better turnout than we have for the past two months.

 * C.O.N.N.I. IS TEN!!! *

C.O.N.N.I. IS TEN!!!
 by Dick Beery

I wonder if our founders, when they gathered in October of 1982 in Pat Saturn's living room on Grandview avenue could even have imagined the many changes that would occur during the ensuing ten years? They certainly did not realize that within a year Texas Instruments would stop marketing and supporting their home computer lines. Nor could they have foretold the tremendous support and expansion for the 99/4A that have occurred within this time period. When I joined, in June or July of 1983, most people had cassette systems, myself included. The lines at the cassette library each meeting were long, and you had to choose three times as many as you could get (2) and hope that maybe one of those would still be available by the time you got to the head of the line.

The many software offerings, including those for eighty columns, are well known, as is the fact that your 4A can now sport a hard drive and several floppies.

The Geneve (Myarc 9640), slow to leave the launching pad, offered great promise, largely though not totally fulfilled. It seems to have new life being breathed into it at this instant. Bud Mills promises soon to offer for the 4A expanded memory and other capabilities. Other hardware and software offering hope for things we never dreamed of in

the beginning keep appearing. I plan to stick around and see what they can do--it's amazing the strides that have been, and are being, made.

In C.O.N.N.I. itself, we expanded the disk library (free); added a Disk of the Month (pay); raffles; cartridge and publications libraries; opened and operated a BBS (Spirit of '99), which now includes also the National Clearinghouse BBS; offered, withdrew and again offered, the opportunity for those outside our immediate area to subscribe to the newsletter and D.O.M.; attached a special section of files from Genie to the library; and made several changes of meeting location --Electrical Workers' hall, Martin Janis Center, and now Chemical Abstracts. During the presidency of Irwin Hott, we added a second monthly meeting, on Wednesday evening, to accommodate those who for various reasons were unable to attend on Saturday.

We have exhibited at some mall shows and multicomputer shows; taken sale disks and information to the Chicago, Milwaukee and Lima fairs on numerous occasions; placed notices in national and local publications (Micro Center and the Columbus Dispatch, plus Computer Shopper and another). Our members have given numerous demos at the aforementioned fairs; Irwin Hott even went to New York state to demo the use of the 4A by the blind. Jim Peterson's excellent programming and worldwide renown have shed glory on the group in

general. Bud Wright owns and operates TIABS, a computer BBS on the 4A that he wrote himself. He and Karl Romstedt have provided us with many outstanding programs (and how-to classes) in assembly. Pat Saturn in the earlier years provided classes in Forth. Art Morgan gave classes in the uses of Multiplan. Jean Hall taught wordprocessing and Ken Marshall, TI-Artist. Harold Timmons has brightened our lives with his many transcriptions of music for the computer. Many others have offered their services in many ways: teaching classes, serving as officers or on committees, and just generally being helpful. C.O.N.N.I. has the reputation of being extraordinarily helpful to the beginner, as well as the more advanced user. Let me now apologize to those whose contributions I may seem to slight. I got this assignment with a short deadline, and have done my best to remember all.

Where will the group be two, three or five years from now? No one knows. The erosion of members to other computers has been great, especially as the prices for these others have been drastically reduced. This is true in 4A groups all over the country. We will just have to wait and see. I personally enjoy using more than one brand of computer and hope others will find the same to be true for them. Maybe it doesn't have to be "either/or". Maybe it can be "and". I hope so.

END

MUSIC - MUSIC

Here is another one of those musical tunes that plays on and on until you stop it. You may enjoy it.

10 CALL CLEAR

20 REM MUSICAL DOODLES

30 REM TI BASIC

40 RANDOMIZE

50 DIM N(30)

60 F=220

70 FOR J=0 TO 36

80 X=X+1+(X=12)*12

90 IF (X=2)+(X=5)+(X=7)+(X=1

0)+(X=12) THEN 120

100 Y=Y+1

110 N(Y)=INT(F*1.059463094^J

)

120 NEXT J

130 K=8

140 K=K-INT(5*RND+1)+INT(5*R

ND+1)+(K>21)*2-(K<1)*2

150 IF (K<1)+(K>21) THEN 140

160 CALL SOUND(-999,N(K),0,N

(K)*2,0,N(K)*3.75,30,-4,5)

170 GOTO 140

(Thanks to San Antonio Area 99ers)

Word processing with Multiplan? Why not? Multiplan has many advantages over Ti-Writer and the Editor/Assembler Editor. For instance, Multiplan will allow you to format your document in a columnar layout and print it in condensed text, providing for a larger amount of text on a given page. In addition, Multiplan will center your text where desired, and allow for the movement of blocks of text in a more flexible format.

Using Multiplan as a word processor does have its drawbacks. Among these are the lack of a global editor, editing of text is a bit more difficult (you can't simply type over your text), and fast typist will have to learn to slow down a little due to the programs relatively slow processing speed.

Despite these drawbacks, however, for many applications Multiplan may be the easiest way to solve the problem at hand.

I don't propose to go into a full tutorial on the use of Multiplan, for that I would refer you to the Multiplan Manual. I realize that many people find this a formidable document, but for use as a text processor, only a general knowledge of the use of Multiplan is necessary. Therefore, in this discussion, I will merely cover what I have found to be the easiest steps to follow in setting up and using the worksheet.

Starting with an empty worksheet, your first step should be to select the OPT or OPTIONS command and turn off the recal option. Since you will be doing no mathematical calculations, this will eliminate the considerable delay incurred as the program searches for mathematical cells.

Next, select the FORMAT option, then DEFAULT on the sub-menu, and finally WIDTH on the next menu, and set the default width at 30 columns. I realize that it is possible to set the width up to 32 columns, but by setting it at 30 we will later be able to widen it to 32 to allow for a buffer between columns

of text.

The next setup step that is advisable is to again select the FORMAT, DEFAULT option, but this time select the CELLS option on the third menu. In the alignment column select L for left. Remember, when Multiplan is displaying the ALPHA/VALUE prompt, hitting a number as the first character in a line will select the VALUE option rather than the ALPHA. Therefore, if the first character in a line is a numeric one, you must first hit enter twice to specifically select the ALPHA command. In case you forget, however, and the only characters entered on that line are numeric ones, this will prevent them from being right justified or otherwise skewed.

The final setup step I use is to select the WINDOW option and place a border around the one open window. You may then use this border as a line length guide while typing. You may type up to but not including the column containing the right border without having the end of your text cut off.

You are now ready to begin entering your text. Start at row one, column one, and enter one line after the other in column one. I prefer to enter all of my text in column one and format it later, since this makes it somewhat easier to move data about. Another advantage is that you don't have to worry about keeping track of where you are located on the page.

Once you have finished entering your text, you are ready to format the data into columns. Since the maximum column width on the TI printer is 132, we will divide the text into 4 equal columns of 32 characters each and have a 2 column border on the left and right margins.

Assuming we're working with one page as an example, there are two ways you can format the text. One would be to simply divide it into 54 rows per column (assuming your page length is 66), and leave whatever may be left over in the fourth column. You may also decide that you would like the columns to be of even length, in which case you would simply divide the total number of rows by four, and make each column that length.

For example, let's assume the total number of rows, when the document is

formatted in one column is 200. 200 divided by 4 equals 50. We would therefore make each column 50 lines long.

To do this, we would copy from row 51 to 100, and place the copy in row 1, column 2. Next we would copy from row 101 to 150, and place the copy in row 1, column 3, and finally, we copy from row 151 to 200 and place the copy in row 1, column 4.

You now have the entire document in rows 1 through 50 and columns 1 through 4, but you still have copies of columns 2 through 4 below row 50 in column 1. To get rid of these use the delete command. Now change the default width to 32 to provide space spaces between columns.

You are now ready to print the file. To do this, first, save the file to disk. Next, exit Multiplan and select TI BASIC, then enter the following commands:

```
OPEN #1:"PIQ.CR" (USE DOUBLE QUOTES)
PRINT #1:CHR$(15)
BYE
```

If your printer is not connected to the parallel I/O interface, you will have to supply the proper file name. This procedure sets up the TI printer to print in condensed text.

Next re-enter Multiplan and select PRINT, OPTIONS. Enter your printer name in the setup field and return to the PRINT menu. Now, select MARGINS and set the left margin to 2 and change the print width to 132.

All that needs to be done now is select the PRINTER command and your document should come out in 4 even columns.

I'll admit that this procedure sounds a bit tedious, but it is the most flexible means I know of to format text into columnar form. I have made several attempts to devise a program to translate a TI-Writer file into a Multiplan file using the symbolic link file format, but so far all of my attempts have proved to be fruitless. I'm still working on it, so if I have any success I'll let you know.

Pete Phillips

END

JUST A LITTLE BIT HELPS

by Jim Peterson

I'm not going to say that everybody should learn to program. Most folks can get along just fine by using the programs written by others.

But, every once in a while you need to do something that would be so easy for the computer to do, and so hard to do without a computer - but no one has written a program to do it. That is when just a little bit of programming knowledge can be a lifesaver!

In a previous article, I described my genealogy program, and the index for it I had created in Funnelweb by setting the tabs at 1 for the person's index number, 5 for the first name, 36 for the surname, 56 for the paragraph number, 61 for the father's index number and 66 for the mother's, and 71 for the spouse.

I have been busily adding names to that index until I am now up to well over 900, and I realize that I have goofed!

I set that second tab to allow space at left for a 1- to 3-digit index number and a space, but I will soon be going past 999. I need an extra space there. I allowed more than enough spaces for the names, so I can take some of those away. While I'm at it, I would like to allow space for second and third spouse numbers at the right, and I want to keep the total spaces well short of 80 so I can print the index in two columns of elite condensed.

Am I going to retype 900+ lines, and undoubtedly make some errors while doing

so? No way! Never do anything that you can make the computer do for you.

All this requires is knowing how to open one file to read from and another to write to, read a record from the one file, manipulate it a little, and write it to the other file.

The file part is extremely simple, just OPEN #1:"DSK1.OLDFILE",INPUT and OPEN #2:"DSK2.NEWFILE",OUTPUT - or whatever drives and filenames I choose. The computer takes it for granted that I'm using DVBO files, so I don't have to tell it. Even the INPUT and OUTPUT could be omitted, but it's best not to. If you don't tell it otherwise, the computer will allow you to accidentally write to the file you meant to read from, and that is disastrous!

The reading and writing is equally simple - 110 LINPUT #1:M% :: PRINT #2:M% :: IF EOF(1)<>1 THEN 110 ELSE CLOSE #1 :: CLOSE #2

Always use LINPUT when reading a text file, because INPUT will only read the record up to the first comma it finds.

EOF is set to zero until the last record in the file has been read, when it is set to 1, so IF EOF(1)<>1 means "if the end of the file opened as #1 has not been reached" - in which case, go back to the same line number and read another record, but if you have read everything, close the files - and especially close the one you have written to, or you will have wasted your time.

But, I wanted to change the lengths of the fields in each record before I move it to the new file. For that, I need to know only how to use two commands - SEG\$ and &.

SEG\$ extracts a part of a string - a string, in this case, is the record I get with that LINPUT, which is one line of up to 80 characters that I keyed in with Funnelweb. SEG\$(M%,1,4) will give me 4 characters starting with the first character of M%; SEG\$(M%,5,25) will give me 25 characters starting with the 5th. And & will put those two together into one string. I want to keep that first 4-character field but expand it to 5 characters, so SEG\$(M%,1,4)&" ". The next two fields, starting at 5 and 36 and consisting of 31 and 20 characters, I want to cut to 20 and 16 characters, so - SEG\$(M%,1,4)&" "&SEG\$(M%,5,20)&SEG\$(M%,36,16). And I go on to add a space to each of the next three fields, and the whole program looks like this -

```
100 OPEN #1:"DSK1.INDEX1",IN
PUT :: OPEN #2:"DSK1.NEWFILE
",OUTPUT
110 LINPUT #1:M% :: PRINT #2
:SEG$(M%,1,4)&" "&SEG$(M%,5,
20)&SEG$(M%,36,16)&SEG$(M%,5
6,5)&" "&SEG$(M%,61,5)&" "&S
EG$(M%,66,5)&" "&SEG$(M%,71,
3)
120 IF EOF(1)<>1 THEN 110 EL
SE CLOSE #1 :: CLOSE #2
```

And in a few minutes, that program does the job, saving many hours of work!

Attention, newsletter editors - if you print this through the Formatter, PLEASE transliterate the & !!

END

FROM THE TEACHERS DESK

by Dave Howell

(Thanks to ERIE 99'ERS GROUP)

ELECTRONIC CATALOG FOR COLLEGES???

Memphis State University (MSU) is using computer technology to produce an innovative guide to the school. It comes on a disk and it's called the MSU Electronic Viewbook with some useful and entertaining features built in. The disk contains information about admissions, financial aid, academic programs, scholarships and other aspects of life at MSU. It is being distributed to guidance counselors in and out of state. The

product even includes an option to print out applications and provides a calculator that lets students estimate the cost of going to the school.

The reason for this move? The cost savings are great. Computer disk technology can distribute vast amounts of information for a production cost of only pennies per disk. A general guidance packet of printed material cost about \$15 to produce and mail compared to a little more than \$1 for a disk.

Because students have such a brief attention span, a computer program with good graphics and entertaining activities is important. Another option on the program even allows students to

print out banners and other materials with MSU's name on them.

EDUCATION SOFTWARE AT HOME-NOT VERY POPULAR!!???

According to a study released by the Software Publishers association (SPA), "Education and recreational entertainment users average less than five hours per week using their [personal computer]." Most startling, however is the revelation that much of the software is not exactly bought and paid for! SEE TEACHERS PAGE 15

by Jim Peterson

In a recent article regarding the NX1020 printer, I mentioned that Midwest Micro was selling ribbons for that printer at \$3.98 each in lots of six. I ordered six of them.

When I put in the first one and tried to print something, the chatter clutch on the printer made a terrible racket. The spindle of the cartridge was stuck tight, would not budge.

I put in another one, and tried again. It printed beautifully for several pages, then I noticed that it had stopped putting anything on the paper. The spindle was turning but the ribbon was not moving.

I called Midwest Micro and they, a bit reluctantly, gave me a return authori-

zation for all six ribbons.

In the meantime, I had called V-Tech Inc. and talked to a very knowledgeable man about reinking ribbons. He sent me a 2 oz. bottle of V17R roller ink, which he recommended, for \$2.90 plus \$1.80 shipping. That should be enough to last longer than my printer.

He also sent an instruction sheet which states that original ribbons and good quality replacement ribbons have a woven seamless ribbon loop, but that replacement cartridges from MEI, Fullmark and Midwest Micro have a poor spliced loop (Midwest Micro calls it "electronically welded") and are not worth reinking.

V-Tech also sells a very wide variety of ribbon cartridges at very reasonable prices, as well as reinking machines, etc. They also have replacement ribbons;

a woven loop ribbon for the Star NX-1020 is only \$1.50, a 42 ft. welded loop 4-color ribbon for the NX-1020R is only \$3.50.

I found it very easy to re-ink an NX-1020 cartridge; just use a knife blade to pry it open, and apply ink to the foam roller. However, there is a strip of tin in there which can fall out, so keep the cartridge flat on the table as you are opening it.

You must saturate the foam roller with ink - it soaks in very slowly - and you must run several pages of printing through before the ribbon will be inked and printing evenly.

The address of V-Tech Inc. is 2223 Rebecca, Hatfield PA 19440. The phone number is (215) 822-2989.

END

STOR MOR - REVIEW BY JIM PETERSON

Quite a few years ago, shortly after Texas Instruments abandoned ship, I invested in a P-box with a 32k card.

I had been working on a program that generated a lot of strings internally, and used up more memory than I had available. Now that I had those monstrous megabytes of extra 32k available, I loaded that program, ran it - and got a MEMORY FULL error!

I was immediately on the phone to the Texas Instruments technical people, and learned the sad news - even with the extra 32k available, in Extended Basic the TI-99/4A can only store string data in the console's memory.

To me, that has always been one of the two weaknesses of our favorite computer - the other being that 28-column screen that makes it look like a child's toy.

If you do not have the 32k card, programs are loaded into VDP RAM in the console. Any strings that you load into the program from a cassette or disk file or from DATA statements in the program, or that the program execution generates, must also fit into that memory.

If you do have the 32k, the program is loaded into the 24k of it called "high memory"; the other 8k of "low memory" is reserved for assembly routines. Strings are still stored in the 12k of VDP RAM or "stack" in the console. If you are not using strings, the VDP RAM is largely unused; if you do not have

links to assembly, the low memory is unused; and since most programs are far less than 24k in size, much of the high memory is unused. The TI-99/4A has plenty of memory - it just isn't distributed efficiently. Even a 640k PC has only 64k available for Basic programs. I have never run short of memory when writing a TI program, but I have several times been frustrated by lack of string storage memory.

Finally, Bruce Harrison has done something about this problem, and he has done his usual thorough job. His program checks to see how much high memory remains unused and then pokes strings into it, just as you would load an array. For instance, in Extended Basic after opening a file you might execute
 FOR J=1 TO 10 :: LINPUT #1:M\$(J):: NEXT J.
 With Bruce's routine you would use
 FOR J=1 TO 10 :: LINPUT #1:M\$:: CALL LINK("PUTHI",M\$,J):: NEXT J.
 In XBasic you could then write PRINT M\$(8); with STOR MOR you write
 CALL LINK("GETHI",M\$,8):: PRINT M\$.

Before performing those links, you must first tell the computer how many strings to accept, by doing a CALL LINK K("SETHI",X), where X is the number. To find out how many bytes are available, you can CALL LINK("AVHI",X). Repeating the call to SETHI will wipe out everything you have stored so you can start over.

Just in case you do write such an immense XBasic program that it doesn't

leave space in high memory for strings, Bruce has also provided a routine to store strings in low memory. It takes up only 1000 bytes, leaving about 6000 available for storage. It works in the same way except that you link to SETLO, PUTLO, GETLO and AVLO. You can't use both high and low memory for storage in one program - but you still have that 12k available in VDP RAM.

You can also store numeric data, by converting it to strings. There are still other features, including some unique error trapping methods, and routines to save even more memory by preloading the assembly. The instruction file explains everything clearly. This program has everything I could have thought of asking for, plus things I would never have thought of.

The disk contains source code, object code and demo programs for both the high memory and low memory storage programs as well both preload programs. It also contains the instruction file, a program to print the instructions, and Tod Kaplan's ALSAVE so that you can imbed the assembly into your program for instant loading.

And for all that, Bruce is asking the princely sum of \$6, which includes S&H. Don't you wish we could buy TI programs at PC prices? The address is Harrison Software, 5705 40th Place, Hyattsville MD 20781.

END

WHAT IS AN IC?

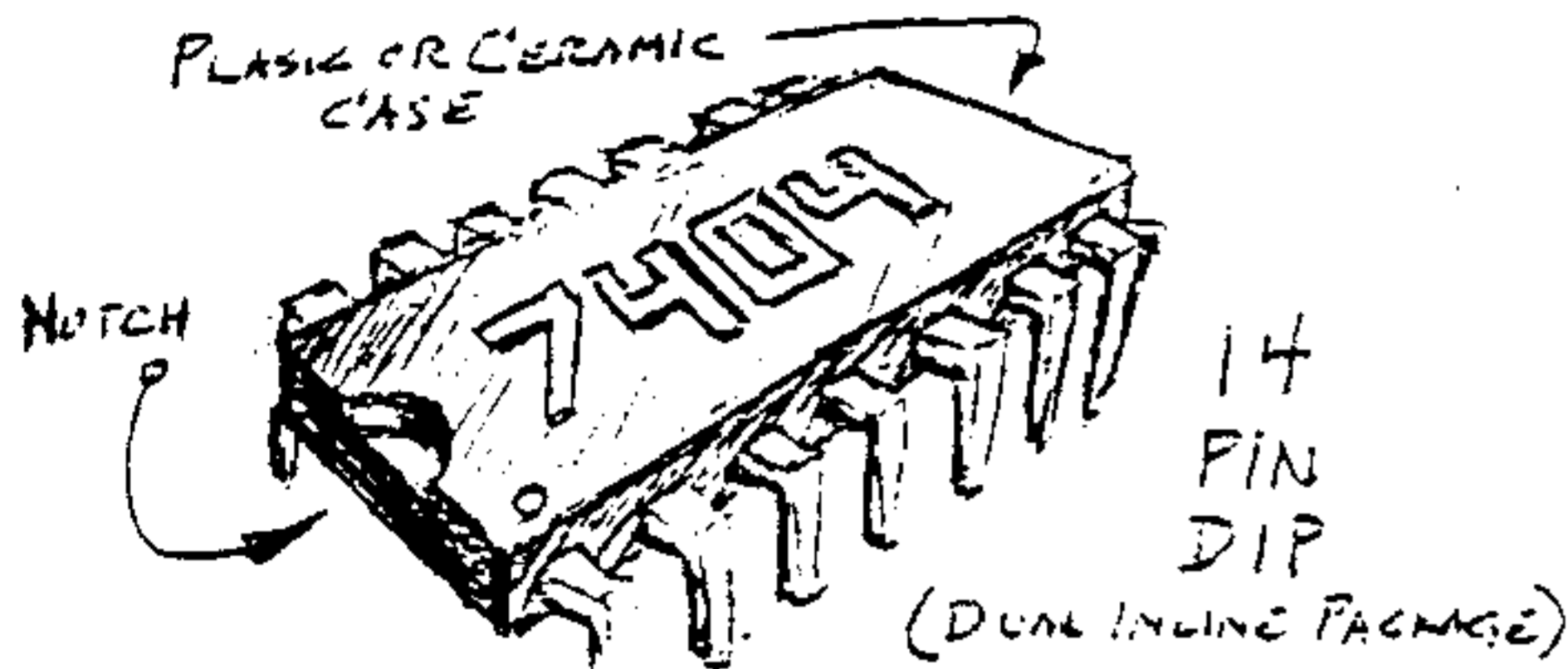
by

Bob Webb

(POMONA VALLEY 99ers)

What is an IC, and why should I care? In truth you don't need to know. However, if you are at all curious, please read.

Inside your Computer Console there are many Black IC Chips. They all look the same except for their size. You may have noticed some have 14 legs and others have 28. If you look close, all of these rectangular bug like objects have a notch at one end. This notch is usually shaped like a half moon or a semi circle.



Well we all seem to know what they look like. However, look at those numbers and letters. What do they mean?

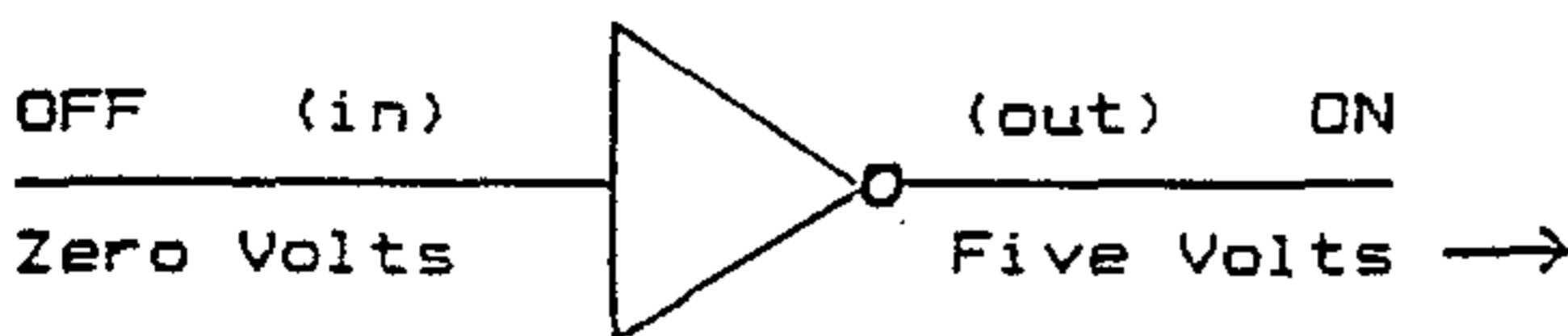
Lets begin with those letters and numbers. We can all guess that they are some kind of identification.

There are thousands of kinds of IC Chips. One "family" of IC Chips is the 7400 group. This group of Chips always start with the number 74, and the next 2 numbers is the identifier. For example the Number 7404 is known as a HEX INVERTER. The 7408 is known as a QUAD 2-INPUT AND GATE.

I can hear you from here. You are thinking, "YIKES, WHATS THAT?". Don't worry about that right now. That is just their names.

Now I want to add to their names. A 74C04 is a HEX INVERTER also but the "C" in the middle means it is made from a different material. The Chip looks identical to the 7404 Chip. But, due to its different chemical makeup, it can be used in equipment that operates in an electrically noisy type environment. In reality, it is an improved version of the 7404. You could also buy a 74C08. With this little bit of information you can identify some of the Chips in your Console. The "C" stands for CMOS, or Complementary Metal-Oxide Semiconductor. There are many other types of 7400 Chips. 74LS04, 74HCT04, 74HC04, and 74VHC04 just to name a few. These Chips are all the same except for their chemical makeup. They all have their own special applications. For Example, the 7400 Family Chips is composed of mostly DIGITAL LOGIC TYPES. Simply stated this means they only have use in digital computing or digital switching machines.

Computers operate on BINARY LOGIC. Either on or off. Their purpose in life is to turn things on or off.

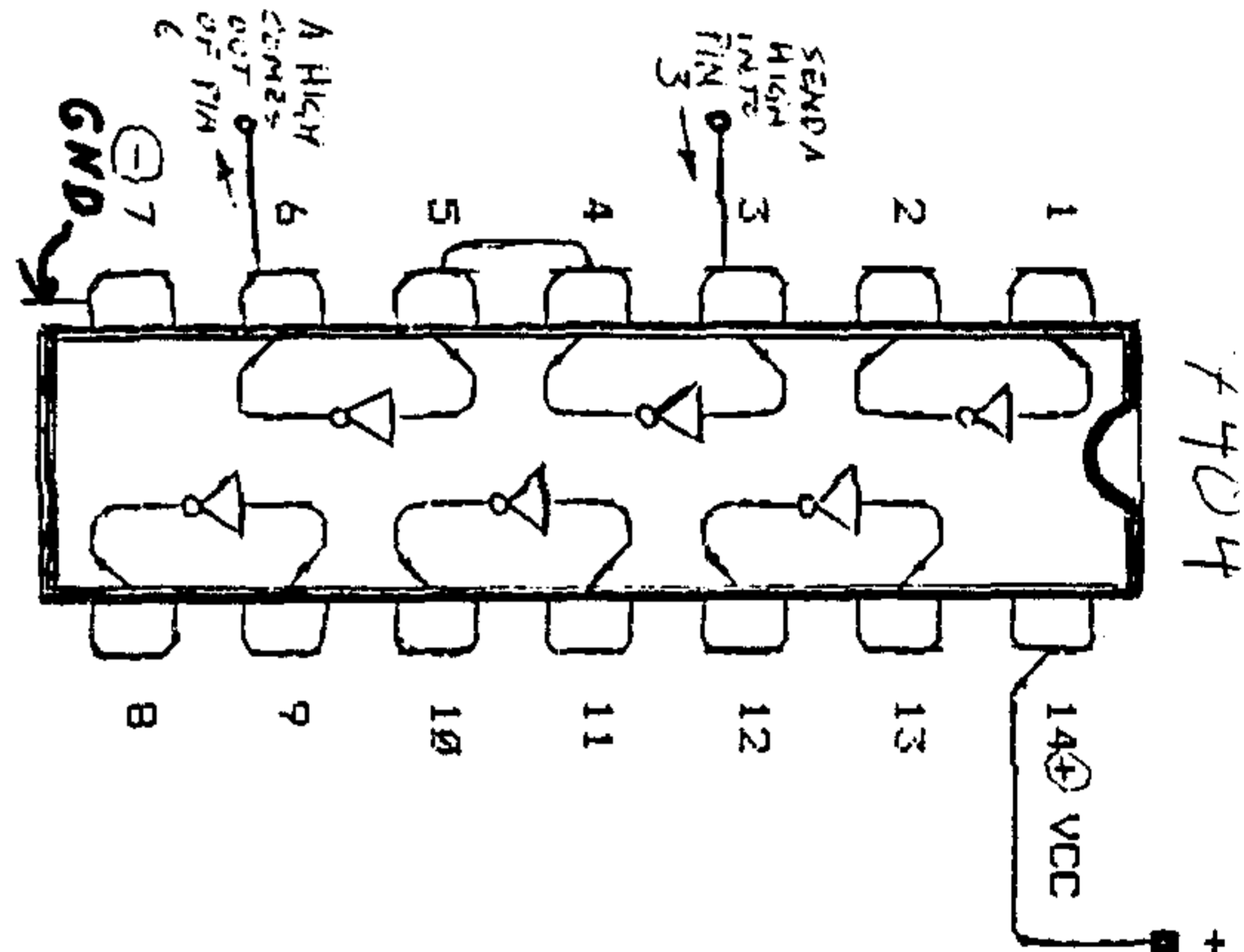


The symbol above is known as an INVERTER. It is one of many symbols used in Digital Logic. Lets say your computer has a single wire coming out of it. That wire is off. There is no elec-

tricity coming out of it. Digitaly speaking it is "off", or Zero volts. The 7404 HEX INVERTER is at the end of that wire. The "off" signal goes in to the INVERTER and the wire coming out is "ON" or 5 volts. If the wire from the computer suddenly goes on the IC will INVERT that signal and the wire exciting the 7404 will go off. How does the IC get the 5 volts? Most IC Chips must have power to operate. Just like your Stereo, you must have it hooked up to the electrical power source in order for it to operate. this brings us to all of those metal, bug like legs. One Leg is Positive hook up wire and the other is ground.

Just like your Car Stereo, the Chip operates on DC current. So, you must have a positive terminal and a negative terminal. The "Family" of 7400 IC's uses this Standard no matter how many legs the Chip has:

The number one leg is always the Top Left Leg. The Notch at the top of the Chip tells you which end is the Top of the Chip.



Also note that some Chips have a Dot next to the Number One Pin, or Leg.

The Numbering Scheme is ALWAYS the same no matter how many Pin. Number 1 is at the Top Left and the rest go down the left side and wrap up the right side from the bottom. The Numbering Scheme never changes, however the power pin locations can vary from one type to another. MOST OF THE TIME THE BOTTOM LEFT PIN IS GROUND PIN AND THE TOP RIGHT PIN IS THE POSITIVE DC VOLTAGE PIN!

Now that we have power to the Chip we can send High's and Low's through one of the INVERTER's. This particular IC has six INVERTER's built into it. Send a Low, or Zero Volt signal to pin number 3 and out of pin number 4 will come a High or 5 Volts.

ON, OFF, ON, OFF, ON, OFF. This is not a Glamor Chip. It is a bread and butter work horse chip. Most of the time these types of chips are used as BUFFERS. A Computer has a wire going inside from an outside peripheral like a disk drive. If the disk drive fails and sends a dangerous amount of High Voltage down that line it could destroy the CPU or other related Chips inside the Computer. So, the Engineers put INVERTER Chips between the Computer and Peripheral to protect each Unit. Hence the name BUFFER. Notice if you pass a High into Pin 3 and connect pin 4 and 5 together you will get a High back on pin 6! There is no change to the Signal High or Low but you have made a very good

BUFFER! The IC will get FRIED before the CPU, stopping the damage. Chops can vary a great deal. But, the Pin Sequence is always the same. Some Chips are Analog chips. They operate over a range of voltages. The Digital type deals with Zero and Five Volts, or Zero and Fifteen Volts.. High and Low, or ON and OFF. Some Chips are nothing more than a pack of resistors.

They are all built to withstand harsh environments of Heat and Vibration, or allowed to vary in Voltage even if they work next

to a big AC motor. Some Chips are used to clean up a signal after it has traveled a long distance over a wire.

Some Chips are Extremely complex inside, like out TMS 9900 Microprocessor. It has thousands of INVERTER's, TRANSISTORS, AND LOGIC GATEDS built into it. Not all Chips are the same, even though they look like it from outside

END

```

TI-WRITER FONT MAKER          zKzclearzzPzprintzzMzmirrorz
  by James Stringfellow       " : "zzzzzzzzzzzzzzzzzzzzzzzzzzzzzz
    Paris, France              zzz" :: M$,M2$=""
    (Thanks to TISHUE NL)      150 DISPLAY AT(1,7):"zTIWRIT
                                ERzFONTsz":Z$:Z$ :: DISPLAY
                                AT(18,1):Z$:Z$:Z$ :: R=4 ::
                                C=9 :: K=75 :: CALL SPRITE(#
                                1,128,16,25,65)
                                160 REM
                                170 CALL KEY(3,K,S):: Y=(K=6
                                9 OR K=82 OR K=87)-(K=88 OR
                                K=90 OR K=67):: X=(K=83 OR K
                                =87 OR K=90)-(K=68 OR K=82 O
                                R K=67) :: IF K=81 THEN 190
                                :: IF K=80 THEN 200 : IF K=7
                                5 THEN 120
                                180 IF K=64 THEN 370 :: IF K
                                =77 THEN 330 :: R=R+Y :: C=C
                                +X :: R=R-(R<4)+(R>17):: C=C
                                -(C<9)+(C>24):: CALL HCHAR(R
                                , C,CH):: CALL LOCATE(#1,R#8
                                -7,C#8-7):: GOTO 170
                                190 CH=CH+1+(CH=129)#2 :: CA
                                LL HCHAR(R,C,CH):: FOR I=1 T
                                O 30 :: NEXT I :: GOTO 70
                                200 GOSUB 350 :: FOR C=9 TO
                                24 :: X=64 :: FOR R=4 TO 10
                                :: CALL GCHAR(R,C,6):: F6=12
                                9 THEN A=A+X
                                210 X=X/2 :: NEXT R :: FOR J
                                =1 TO LEN(STR$(A)):: CALL VC
                                HAR(J,C,ASC(SEG$(A),J,1))::
                                NEXT J :: M2$=M2 $&CHR$(A):
                                : A=0 :: NEXT C
                                220 FOR C=9 TO 24 :: X=64 ::
                                FOR R=11 TO 17 :: CALL GCHA
                                R(R,C,6):: IF 6=129 THEN A=A
                                +X
                                230 X=X/2 :: NEXT R :: FOR J
                                =1 TO LEN(STR$(A)):: CALL VC
                                HAR(17+J,C,ASC(SEG$(STR$(A),
                                J,1))): NEXT J :: M$= M$&CH
                                AR$(A):: A=0 :: NEXT C :: CA
                                LL DELSPRITE(ALL):: CALL MAG
                                NIFY(1)
                                235 CALL SPRITE(#1,128,16,18
                                5,233):: DISPLAY AT(24,6):"z
                                zzzzzzDoublezDensityzN" :: A
                                CCEPT AT(24,28)VALIDATE ("YN
                                ")SIZE(-1):Q$ :: K=75 :: IF
                                Q$="Y" THEN K=K+1
                                240 OPEN #1:"RS232.BA=4800.D
                                A=8.PA=N"
                                250 PRINT #1:CHR$(27)&"A"&CH
                                R$(7)
                                260 PRINT #1:CHR$(27)&CHR$(K
                                )&CHR$(16)&CHR$(0)&M2$&CHR$(
                                13)
                                270 PRINT #1:CHR$(27)&CHR$(K
                                )&CHR$(16)&CHR$(0)&M$ :: CLO
                                SE #1
                                280 DISPLAY AT(24,6)SIZE(23)
                                : "Printzagainzyeszorznozn" :
                                : ACCEPT AT(24,28)VALIDATE("
                                YN")SIZE(-1):Q$ :: IF
                                Q$="Y" THEN 235
                                290 DISPLAY AT(24,6):"zzzzzz
                                zzzzSavezyesznozn" :: ACCEPT
                                AT(24,28)VALIDATE("YN")SIZE(
                                -1):Q$ :: IF Q$="N" THEN 140
                                300 CALL LOCATE(#1,185,145):
                                : DISPLAY AT(24,2):"zzzFilen
                                amezDSK" :: ACCEPT AT(24,17)
                                SIZE(-12):F$ :: IF F$="" THE
                                N 140 :: OPEN #2:"DSK"&F$,FI
                                XED 80
                                310 PRINT #2:CHR$(27)&"K"&CH
                                R$(16)&CHR$(0)&M2$
                                320 PRINT #2:CHR$(27)&"K"&CH
                                R$(16)&CHR$(0)&M$ :: CLOSE #
                                2 :: GOTO 140
                                330 FOR R=4 TO 17 :: X=64 ::
                                FOR C=9 TO 24 :: CALL GCHAR
                                (R,C,CH):: IF CH=128 THEN CA
                                LL HCHAR(R,C,129,1)ELSE IF C
                                H=129 THEN CALL HCHAR(R,C,12
                                8,1)
                                340 NEXT C :: NEXT R :: CH=C
                                H+1+(CH=129)#2 :: GOTO 140
                                350 DATA 80,76,69,65,83,69,3
                                2,87,65,73,84
                                360 CALL DELSPRITE(#1):: CAL
                                L MAGNIFY(2):: FOR I=2 TO 22
                                STEP 2 :: READ Y :: CALL SPR
                                ITE(#I,Y,16,I#8-7,I+I#8-7+16
                                ):: NEXT I :: RESTORE :: RET
                                URN
                                370 END

```

Thanks!!!! Walter

Thanks!!!! Walter

C.O.N.M.I. User Group members and Bud Wright extends a hearty thanks to Walter Hard for the contribution to our group for Bud Wright's music programs that he received.

EASY ASSEMBLY LANGUAGE

LESSON 6

by Bob Webb

Hello again. This will be the last of my beginners series of lessons. I am studying BIT-MAP mode right now and hope to show how it works soon. BIT-MAP mode is something you have seen in graphic drawing programs. Each pixel on the screen can be turned on or off, and each row of eight pixels can have two colors. This is not as good as a different color for every pixel but it can be used in stunning ways. More on that later. If you have any questions, or want to enlighten me on any points, please write. Also, if you would like the full set of lessons, please include \$10.00 with your request to cover postage and copying. The post office charges \$.95 just for postage! Jeepers! Send your request for the series to:

Bob Webb
P O Box 3023
Arcadia CA 91007

LESSON NUMBER SIX

Well, we want to write our first letter of instructions to 9900 MAN. We will use the EDITOR/ASSEMBLER MODULE to do so. 9900 MAN relies upon the ASSEMBLER program to translate our ENGLISH LANGUAGE (ASCII DV/80 FILE) LETTER OF INSTRUCTIONS into BINARY MACHINE CODE INSTRUCTIONS. Like BASIC and EXTENDED BASIC the way you write the instructions requires a certain format.

The ASSEMBLY LANGUAGE FORMAT is simple. There are 4 vertical columns of information used. Only the center 2 are needed. The other 2 are optional. Here is a sample program lifted from the EDITOR/ASSEMBLER manual. Pages 342-344.

1 To 41 characters
|<----->|
| 1 4|
|<=>| |<=>| |
123456 1234 123456789012345-to-B901

0001 DEF BUBBLE
0002 REF VMBW,VMBR,VSBW
0003 BBLE DATA >3C7E,>CFDF,>FFFF,>7E3C
0004 COLOR DATA >F333
0005 BBL BYTE >A0
0006 SPACE BYTE >A8
0007 LOC DATA >01DA,>020D,>0271,>02A5
0008 DATA >02D6,>02E1,>0000
0009 MYREG BSS >20
0010 DATA >20
0011 BUBBLE LWPI MYREG ENTRY POINT
0012 LI R0,>394
0013 LI R1,COLOR
0014 LI R2,2
0015 BLWP @VMBW

0016 LI R0,>D00
0017 LI R1,BBLE
0018 LI R2,B
0019 BLWP @VMBW
0020 CLR R0
0021 LOOP1 MOV @SPACE,R1
0022 BLWP @VSBW
0023 INC R0
0024 CI R0,>300
0025 JNE LOOP1
0026 MOV @BBL,R1
0027 LI R2,LOC
0028 LOOP2 MOV \$R2+,R0
0029 MOV R0,R0
0030 JEB SCROLL
0031 BLWP @VSBW
0032 JMP LOOP2
0033 VDPBF1 BSS >20
0034 VDPBF2 BSS >20
0035 SCROLL CLR >20
0036 LI R1,VDPBF1
0037 LI R1,>20
0038 BLWP @VMBR
0039 LI R0,>20
0040 LI R1,VDPBF2
0041 LI R2,>20
0042 LOOP3 BLWP @VMBR
0043 S @LINE,R0
0044 BLWP @VMBW
0045 AI R0,>40
0046 CI R0,>300
0047 JI LOOP3
0048 LI R0,>2E0
0049 LI R1,>VDPBF1
0050 BLWP @VMBW
0051 JMP SCROLL

0052 END

I HAVE ADDED THE LINES TO HELP YOU SEE THE STRUCTURE.

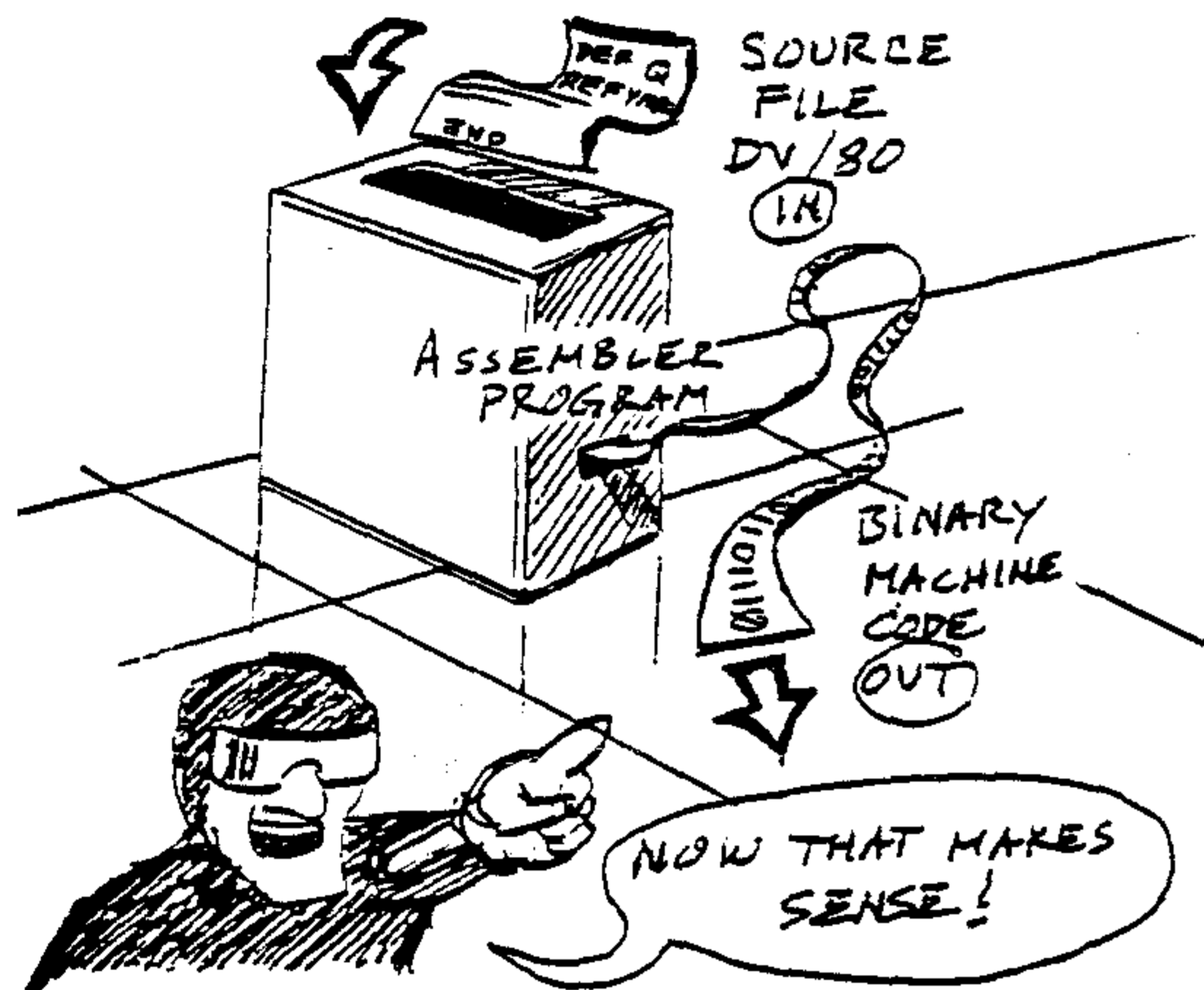


I have only shown 3 vertical columns of data and one column of line numbers. The line numbers are there so I can refer to them. Also to show what it would look like on your computer screen. The fourth column of information would be to the far right of the other three. This column is for comments. Much like a REM statement in BASIC. I have chosen this program from the EDITOR/ASSEMBLER manual because it is simple and gives a great deal of satisfaction when running. I could have written my own program for this lesson. But each of you probably have the manual and this introduction will show you how it is laid out. I have changed the program from the book in only two ways. I have deleted the comments and split the DATA statement at line number 0007. This was done to make it fit on the first page. You will note that when I did this I was not following the manual's strict layout with comments following the asterisks. And, that because this was done, the line numbers would not be the same as the manual's. Line numbers are only used to debug programs. They are not referred to in any

way in any assembly program. When you get around to assembling your program if there are any errors the screen will show you what line number is the problem.

The assembler will give you an idea of what the problem is by saying something like SYNTAX ERROR 0003. (0003 being the line number). But in no way do you refer to a line number in the program itself. This brings me to our first column. Instead of using line numbers for a GOTO type of instruction, ASSEMBLY uses a LABEL. A LABEL is a point to start at or go to.

In assembly JMP is used instead of GOTO. In our example there is an instruction at line number 0032 that says JMP LOOP2. At line number 0028 in the first column is LOOP2. If we were to write this JMP instruction in BASIC and used the line numbers it would be GOTO 28. So JMP is just like GOTO in BASIC but instead of line numbers we use LABELS.



With this example we have introduced the functions of column 2 and 3. The JMP instruction is in the OPCODE column. OPCODE is a fancy name for instruction. This column is the work horse of assembly. All instructions reside in this column. No data or labels are ever put here.

You will also note that there is a space between the LABEL column and the OPCODE column. This space is needed to tell the assembler that the label is finished and the opcode is next. The assembler expects up to six characters in the LABEL column, followed by at least one space, and up to four characters in the OPCODE column. You can have more than one space between columns. The third column is the OPERAND column. This is where you tell the assembler what you want added, moved, changed or read. The instruction is JMP and the OPERAND is LOOP2. The manual calls these columns, in order, LABEL FIELD, OPERATION FIELD, OPERAND FIELD, and COMMENT FIELD.

The Source Statement Format part of the manual goes from page 46 to 48. Now that you have a Global view of the layout we can discuss some of the details needed to make your program work. The ASSEMBLER program needs to know the name of your program. It always needs to see a DEF statement at the top of your list of instructions. The ASSEMBLER takes the name in the OPERAND FIELD

and places it in LOW MEMORY in an area called the REF/DEF table. This is a VECTOR table. An area in memory 9900 MAN knows to go to to find the starting point of your program. It is very important to note line number 0009. The LABEL at 0011 is BUBBLE. Also please note on line number 0001 the Instruction DEF BUBBLE. The ASSEMBLER reads DEF BUBBLE and places the name in the DEF-REF VECTOR TABLE in LOW MEMORY and then looks for that label in the LABEL column. It then knows that line number 0011 is the ENTRY POINT into the program. Or, rather, the first instruction to be executed.

If you will recall from one of the earlier lessons the instruction LWPI. This means to LOAD WORKSPACE POINTER IMMEDIATE. 9900 MAN will take the address following this instruction and place it in his WP digital readout on the Dash Panel in his 9900 MAIL VAN.

After the needed DEF instruction to the ASSEMBLER comes the REF Instruction. In Assembly it is mandatory that you tell the computer that you are going to use one of its, built in, SUBROUTINES in advance of using it. The ASSEMBLER does not like surprises.

An analogy to EXTENDED BASIC would be, you must tell the computer in the first lines of instructions the Commands you are going to use. You would be forced to say, in this program I am going to use the PRINT statement and the DISPLAY AT statement. If you do not put that in the first line an error will be generated and you will not be able to use them. This is how you must treat the built in subroutines in Assembly. What kind of subroutines are there? In our program the VMBW, VMBR and VSNW subroutines are used. So they must be declared in the first instructions. Once again this REF OPCODE is only for use by the ASSEMBLER program. It reads REF VMBW. And knows that anywhere that term VMBW shows up in our text it is to equate that term with the ENTRY POINT into the subroutine program. This subroutine is an assembly language program burned into a GROM chip inside the EDITOR/ASSEMBLER module. The VMBW program, or subroutine, is declared to the ASSEMBLER and it then links your program to the VMBW program. When you use the BLWP or "BULLWHIP" command you are actually leaving your program for a while and running this small subroutine and then branching back to where you left off. Just like the GDSUB command in BASIC. There are many BRANCH routines in Assembly. BLWP stands for BRANCH AND LINK WORKSPACE POINTER IMMEDIATE. The is used if you are branching out from your program, accessing the Mail Boxes at TEX'S GROM RANCH, or sending or receiving Mail to the VDP RAM area. You don't have to declare BLWP because that is an OPCODE. But you must declare the subroutines built in to ROM or GROM. These subroutines make communicating with TEX at GROM RANCH or sending data to the screen in VDP an easy task instead of a long drawn out process. VMBW stands for VIDEO DISPLAY PROCESSOR RAM MULTIPLE BYTE WRITE. What a mouthful!

All that means us that it helps you write multiple bytes of data anywhere in VP RAM. Here is how it works.

Lets say you want to write two bytes of data to an area in VDP RAM. It is not important what this area is used for. Just know that it is something to do with putting an image on the screen of your monitor.

We can use the example starting at line 0011 or, to speak like an Assembly Programmer, at the LABEL "BUBBLE".

After we set up our Workspace Register Scratch Pad area with the LWPI instruction we can see the first line of information needed

NEXT PAGE

for the VMBW program. LI R0,>394 (>394=VDP ADDRESS) LOAD IMMEDIATE >394 into register zero. Register Zero is 9900 MAN's Scratch Pad Ram space. It is a place he holds information while he manipulates it. We don't know where that WORKSPACE AREA is. Because earlier we told the assembler to place it anywhere it likes with the command line number 0009, MYREG BSS >20. BSS is used to tell the assembler to set aside an area of memory X number of bytes and give that area a name. In this case the name is MYREG. Standing for MY REGISTER. You could name it anything you like. Pick a name that will help you remember it. The BSS Instruction set aside >20 bytes of memory for the SCRATCH PAD RAM. You could make it an exact address if you like. LI R1,COLOR LOAD IMMEDIATE >F333 into Register 1. Earlier the LABEL COLOR WAS GIVEN TO THE NUMBER >F333. Now anytime you use the name COLOR anywhere in the program the the ASSEMBLER knows that you mean >F333. In this case it means we make the color of the objects, that will be defined later on in the program, with a foreground color of >F(HEX for WHITE). And a background color of >3 (HEX for LIGHT GREEN). Then foreground >3 and background >3. LI R2,2 LOAD IMMEDIATE into Register 2 the DECIMAL NUMBER 2. Since there is no greater than symbol ahead of the number it knows that the number is a DECIMAL. BLWP @VMBW "BULLWHIP" to the VMBW subprogram. 9900 man then branches to the subprogram and begins to execute those instructions. The BULLWHIP command does what is called a CONTEXTSWITCH. When it branches to the subprogram it branches to another set of WORKSPACE REGISTERS. These registers are dedicated to the subprogram in ROM and GROM exclusively. You better not use these other SCRATCH PAD RAM area's for your programs or you will crash. This context switch takes place so that your registers will be left alone while the subprogram is running. It uses its own space and not yours. No other computer in the world does this that I know of. This is one of the great features of our machines. The subprogram runs and takes the data you put in your own Registers 0,1 && 2. It then does all of the work loading those 2 bytes of data into VDP RAM. Change the number from 2 to 500 and you can see the power of this little subroutine. The subroutine branches back into the program where it left off (line number 0015). And continues to the next Instruction.

The last but not least OPCODE is the END command. If you type the LABEL BUBBLE in the OPERAND column the program will start automatically. You won't have to type the word BUBBLE to start the program. That's it! Now you know it all. HA HA. Study your manual and Good Luck! Thanks for your letters of support and may GOD BLESS US ALL! BYE FOR NOW!

```

* This program run's right.
* It had a bug in the scroll
* routine. The Editor/Assembler
* Manual is wrong. I fixed the
* bug. It was in the SCROLL loop.
* This program is on pages 342
* to 344. The bug is on the 2nd
* line on page 344. The Manual
* adds when it should subtract.
*
* Assemble with the R option
*

```

```
DEF BUBBLE
```

Entry

```
REF VMBW,VMBR,VSBW
```

```

*
BBLE DATA >3C7E,>CFDF, FFFF, 7E3C
COLOR DATA >F333
BBL BYTE >A0
SPACE BYTE >AB
LOC DATA >01DA,020D,>0271,>02A5
DATA >02D6,>02E1,>0000
MYREG BSS >20

```

```
LINE DATA >20
```

```

*
*Set up colors
* BUBBLE is the entry point into the program.
*

```

```
BUBBLE
```

```

LWPI MYREG
LI R0,>394
LI R1,COLOR
LI R2,2
BLWP @VMBW

```

color table 20 and 21
load colors >F3 and >33
2 bytes to load
move to VDP RAM

WHITE
And
GREEN

```

*
* Set up character definition
*

```

```

LI R0,>D00
LI R1,BBLE
LI R2,8
BLWP @VMBW

```

character >A0 location
definition of bubble character
8 bytes to move

```
* Clear screen
```

```

CLR R0
LOOP1 MOVB @SPACE,R1
BLWP @VSBW
INC R0
CI R0,>300
JNE LOOP1

```

start at VDP RAM >0000
move space character
move one space at a time
points to next location on screen
end on >20 = 768th char pos
jump if not equal to loop1

First character
position
screen

```
* Place bubbles on the screen
```

```

MOV B @BBL,R1
LI R2,LOC
LOOP2 MOV #R2+,R0
MOV R0,R0
JEG SCROLL
BLWP @VSBW
JMP LOOP2

```

load character code for bubble
load pointer to address for bubble
load real address
check if finished loading
finished. Start scrolling
write bubble on the screen

```
* Scroll screen
```

```

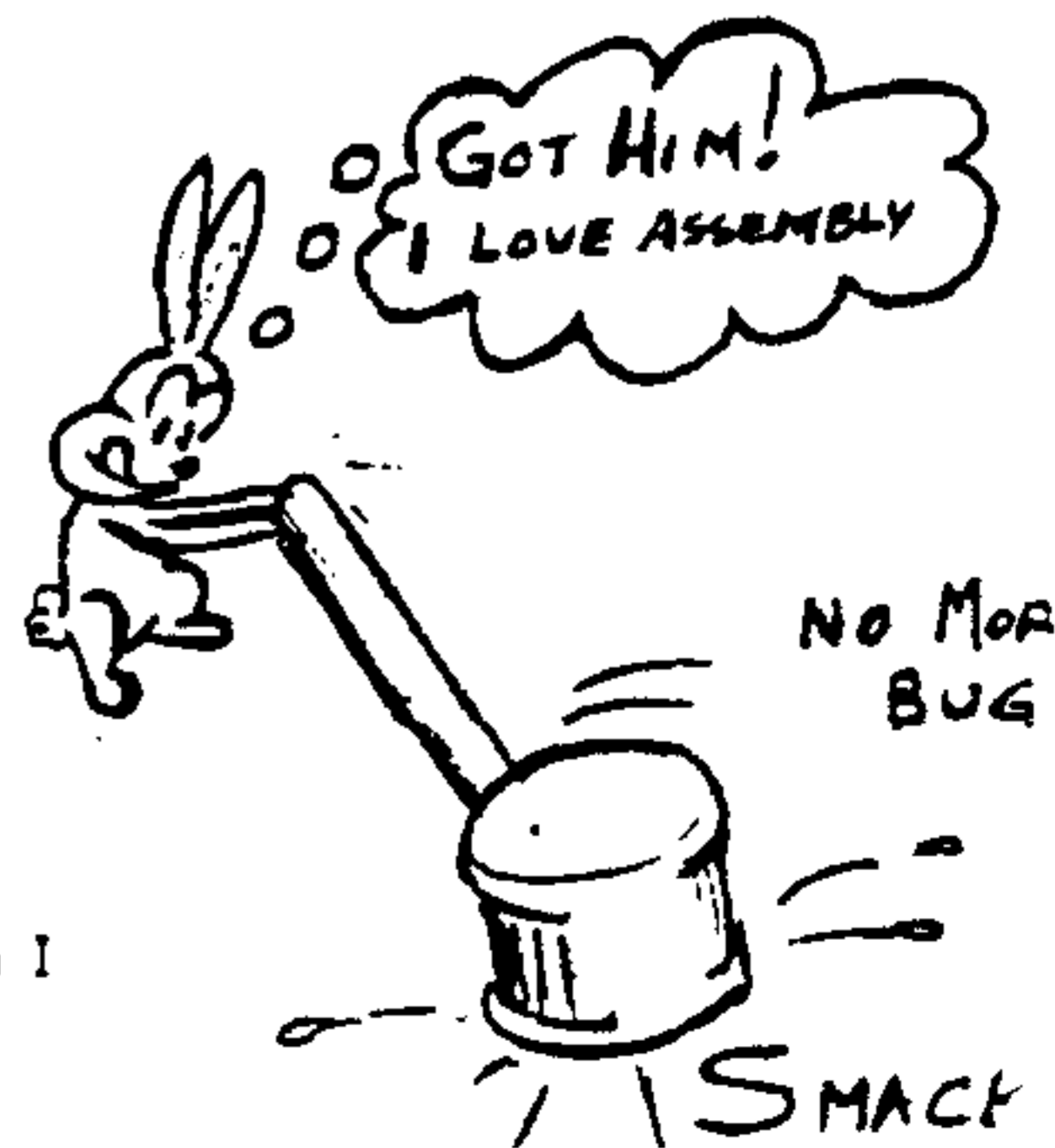
VDPBF1 BSS >20
VDPBF2 BSS >20

```

```
* Fixed SCROLL routine
```

```
* Pages 342 to 344 in the  
* Editor/Assembler Manual
```

```
* At the top of the program I
```



```

$ added this...
$
$ LINE DATA >20
$
$ And in the SCROLL routine I
$ added this....
$
$ S @LINE,R)
$
$ The Editor/Assembler Manual
$ shows an AI (ADD IMMEDIATE)
$ instruction when they should
$ have subtracted. (Top of page
$ 344).
$

```

```

SCROLL
CLR R0          VDP address >0000 =1st screen position
LI R1,VDPBF1   CPU buffer address
LI R2,>20       number of bytes to move
BLWP @VMBR     move >20 from VDP RAM
$
$                               (32 chars=1 line)
LI R0,>20       VDP address >20

```

ABOUT THE DOM - - -

ABOUT THE D.D.M. ...

Well, we're finally up to the August D.D.M. And it's only the October newsletter!

This time a single disk. Plus two programs of interest to you 9640 users.

On side A is 9640 chess. Can be played only on a 9640 or a /4A with a game device such as Gram-Kracker or P-Gram.

Following this is TC-MISC^, a collection of goodies from Jim Peterson, the Tigercub. Archived, and in Extended Basic. "Buckets" has you try to distribute 8 gallons of liquid evenly between two unlike containers. "Form+1/2" lets you fine tune a text file to a greater degree than is possible with the TI-Writer Formatter. Hyphenate, add carriage returns, suppress linefeeds, change margins, strip blanks, right-justify, etc. Docs on this side also. Then a long-division cryptogram where letters have been substituted for numbers. It offers hints. Then "N" "Psycho" and "Ultimate", all involving tricks with numbers. Finally the well-known game "Reverso". A lot of fun here. And it isn't all easy.

Next on side A is a phone list-type database which lets you keep track of all those people you want to call, but not necessarily on a daily basis. Good!

Way back on D.D.M. #32, which you can still order from Chuck, was a good version of Tetris. Unfortunately, the documentation was not in English. Here on side A are the docs in English, finally available.

And last on side A is the READ--THIS file that gives detailed information on all the files and programs. Print it out using Funn-

```

LI R1,VDPBF2   CPU buffer address
LI R2,>20       number of bytes to move
LOOP3 BLWP @VMBR   copy the line
S @LINE<R0     move to lower VDP memory
BLWP @VMBW     write back to the lower line
AI R0,>40       read next line
CI R0,>300      check if end of screen
JL LOOP3       if not, copy more (char position 768)
$
LI R0,>2E0      write the last line
LI R1,VDPNF1   CPU buffer where the first line is
BLWP @VMBW     move CPU to VDP
$
JMP SCROLL     keep scrolling

```

END BUBBLE

(Remove this if you do not want the program to Auto Start. If you remove it you must then type "BUBBLE" to start it).

END

elweb or whatever. Remember my suggestion in an earlier column: print in Condensed or Elite Condensed. You can get the whole file on one sheet of paper by cutting out the excess and taping it to page one.

Side B:

HORSERACE^ A 1992 fairware game (\$5). Purely chance. In Basic, but runs in XB also. Simple graphics. Blind betting. Slow action, but it IS fun!

MICRO^ 39 files, unpacks to 554 sectors, so us a DSSD or higher disk to unpack. Space prohibits revealing all, but here are a few: Dialtone--produces a tone sounding like a dialtone; Disassa--XB program to disassemble Assembly object code files, so you can "see how they did it"; Fibonacci--program generates series of F. numbers--each is the sum of the two preceding numbers. Has serious applications but is fun to watch; Grammar by Regena helps get you ready for entrance exams, etc. Generates a set of two-four sentences, only one of which is correct. Beef up your English grammar painlessly. You can add more data to expand the options.

There are LOTS more programs in this archive, all as good or better than the ones described. Enjoy!

Last, for the Geneve, a Y-modea file for the modea section of GEN-TRI. Won't work on the 4A.

Well, that's it for another month! Having any problems with this or another D.D.M.? Write to us at the Heischman Ave. address and we will explain in the next possible column. See you next time!

END

DISK DRIVES

By Jim Ness, LAUG

It's funny (at least to me), but there are lots of people who seem to know lots of stuff about their computers, and all those tiny chips, and how the bits and bytes are handled. And there seems to be next to nobody that knows anything about disk drives, and how they work. Sensing this huge gap in man's knowledge, I decided to figure out what makes them tick.

The great thing about disk drives is that they can find files buried randomly within a huge field of data, and they do it pretty fast. Actually, they can do it so fast because it's not all random.

The mechanical concept is not all that complicated. A small motor spins at 300 rpm (at least in this country with its 60 hz power supply), and there is a tiny stepping motor attached to a read/write head. A stepping motor is a common item in indexing applications, where you want a motor to move a precise distance and stop on a dime. The read/write head is just a smaller version of what you have on a cassette recorder.

The stepping motor "steps" the head from track to track on a diskette. The tracks are concentric circles, not a long spiral as you would have on an album.

All of this is ultimately controlled by the disk software provided with your computer. Usually this is located in ROM within the machine. In most machines, the ROM is only sophisticated enough to load in the official Disk Operating System (DOS) which is located on the disk in the drive when the machine is turned on. The DOS contains all the file handling software, copying software, etc, and because it is on disk, it can be easily modified and/or updated as time goes by.

Our friends at TI decided to put the whole thing in ROM, which has a few bad side-effects. First, it makes it hard to update and improve the software, which is located in the Disk Controller

Card. Second, although the machine is a 64k machine, just like all the others, TI has set aside so much memory for special purposes, that there is only 32k left to play with. They set aside 8k for cartridges, 8k for disk drives, 8k for RS232/PID, 8k for the Operating System (can't complain about that one), and 8k for various interfaces (speech, sound, VDP). OK, those are all good applications to have, but if you don't use them, you still can't use that memory for other things.

Anyway, all of the controlling software for the TI is located in the ROM card, as I said. This software tells the step motor when to step to the next track, when to return to the beginning, etc.

There is no STANDARD for how a computer keeps track of data. In the case of the TI/4A there is a directory of existing files, and a map of where they are located, at the beginning of each disk. These files are not necessarily all in complete groups. If you delete a 12 sector file from a disk, there is a 12 sector gap recorded in the map. Then if you add a 20 sector file, the software will put the first 12 sectors in the gap, and put the rest in the first available spot. When you ask for a file that is broken up this way, you can hear the disk head scooting along to read each individual segment.

Because the disk drives themselves are pretty STANDARD, there are a few things that don't change. For instance, there are 48 tracks per inch in most 5-1/4 systems. (There is a new 96TPI system around, not TI compatible). Most systems use only 35 or 40 of the available 48 tracks. There are either 9 or 18 sectors per track (single or double density). Each sector holds 256 bytes of data. The standard design allows 250,000 bits per second to be written.

The following is a complete, and to the best of my knowledge, accurate description of the Disk Directory format and file storage allocation used by the TI 99/4A computer.

SECTOR 0 Volume Information Block
Address Contents

0000-0009 Disk name, up to 10 char.

000A-000B Number/sectors on disk
>0188=360, >02D0=720,
>05A0=1440

000C Sectors/track 09sd,12dd

000-000f 'DSK' 445348

0010 >50 Protected, >20 Not P

0011 Tracks/side 28=40, 23=35

0012-0013 Sides/Density 0101 SS/SD
0201 DS/SD 0102=SS/DD
0202=DS/DD

0038-end Sector allocation bit map

This is a sector-by-sector bit map of sector use; 1=sector used, 0=sector available. The first byte is for sectors 0 through 7, the second for sectors 8 through F, and so on.

Within each byte, the bits correspond to the sectors from RIGHT to LEFT. For example, if byte >0038 contained >CF00 then the byte equals 1100 1111. This means that sectors 0 through 3 are used, sectors 4 and 5 unused and sectors 6 and 7 used.

Information for sector 188 starts at >00B5. Therefore, if your disk is SS/SD, all addresses from >00B5 to end should be FFFF if it was formatted by DISK MANAGER and has not been tampered with.

SECTOR 1-Directory Link

Each 16-bit word lists the sector number of the File Descriptor Record (FDR) for an allocated file, in alphabetical order of the file names. The list is terminated by a word containing >0000; therefore, the maximum number of files per disk is 127 [(256/2)-1]. Any addresses past >0000 will not catalog, but will still be accessible. If the first address is >0000, move all addresses four digits to the left, (eliminating this false address), then the disk will catalog. If the alphabetical order is corrupted (by a system crash during name change, for instance), the binary search method used to locate files will be affected and files may become unavailable.

END

Sector >2 to >21-FDRRecords

0000-0009 File name, up to 10 char.

000C Filetype: >01=Program (memory image) <00=Dis/Fix >02=Int/Fix >80=Dis/Var >82=Int/Var File deletion protection invoked by DM2 will be shown by >08 added to the above.

000D (MAXRECSIZE) Records/sector

000E-000F #/sectors to file DM2 will list one more.

0010 For memory-image program files and variable-length data files; this contains

TEACHERS FROM PAGE 6

"While outright purchase is the most often-mentioned method for users to obtain software, a high portion of users also copy software from friends, at work or school," according to the study. Forty percent of entertainment software users, 20 percent of education software users and 19 percent of personal productivity software users said that they copied software from friends, work or school. Shame on you!

SENIORS "MORE COMPETENT" WITH COMPUTERS

Here's one for us old guys! According to a study by McMaster University researchers in Hamilton, Ontario, older adults were rated as "significantly more competent" in the use of computers even though younger adults believe that older adults are less likely to succeed in a computer course and less likely to complete the course than younger adults.

A "PC" THAT YOU WEAR?

NEC is considering a series of computers that can be worn. How about a PC around your neck that not only frees your hands but comes in designer colors as well?

Called "The Wearable Data Terminal," NEC projects its use by inventory man-

the number of bytes used in the last disk sector. This is used to determine end-of-file.

0011 MAXRECSIZE of data file. >50=80, >FE=254, etc.

0012-0013 File record count, but with the second byte being the high order of the value.

001C Block Link (see note)

NOTE on file storage:

Files are placed on the disk in first-come/first-served manner. The first file written will start at sector >0022, and each subsequent file will be placed after it. If the first file is deleted, a newer file will be written in the space it occupied. If this space isn't big enough, the file will be 'fractured', and the remainder will be

agers - worn over the shoulder with a forearm bar code reader. Another version envisioned by NEC is "The Lapbody Computer" to be used by writers who don't have access to a desk. Instead, it hangs from the shoulder and has a keyboard and LCD screen that folds out.

Then there's the computer for emergency medical technicians. This concept would allow them to check patients' vital signs and injuries with a handheld trackball containing an 8-millimeter video camera and sensors. The patient's medical history would be displayed in 2-D glasses and a built-in comlink would transmit data and video to hospital physicians for consultation.

Admittedly, all this is conjecture at this point. Except to wait 10 or 15 years for devices like these. However, NEC has more immediate plans for these devices: A hands-free telephone for skiers by 1993. Walkie-talkies are already popular on Japan's ski slopes - but other people can listen in.

A wearable, wireless CD player can be expected in the next few years.

"CHANNEL ONE" GOOD OR BAD?

CHANNEL ONE, that 10-minute controversial news program, installed in the Erie Public schools last fall, has not received a spectacular reception. Although locally yet, a study conducted elsewhere claims the program beamed to

placed in the next available block of sectors. The block link map keeps track of this fracturing. Each block link is 3 bytes long. The value of the 2nd digit of the second byte, followed by the 2 digits of the first byte is the address of the first sector of this extent. The value of the 3rd byte followed by the 1st digit of the 2nd byte is the number of additional sectors within this extent.

Sectors >2 through >21 are reserved for File Descriptor Records and are allocated for file data ONLY if no other available sectors exist. If more than 32 files are stored on a disk, additional FDR's will be allocated as needed, one sector at a time, from the general available sector pool.

END

high schools does not improve students' knowledge of current events greatly. This study was conducted by the University of Michigan's Institute for Social Research and by Interwest Applied Research of Beaverton, Oregon, who were commissioned by Whittle Communications Inc., owner of Channel One.

The study went on to state that even the small observed differences between viewers and non-viewers are "potentially important." and the advantage of viewing Channel One may be greater at a different point in history. Most students felt they were learning important things from viewing both about life and about national and world news, but its effect on the measured current events knowledge of the average viewer was quite small."

According to the study, Channel One viewers on average knew more about current events than non-viewers, "but the advantage was very small - 3.3 percent on a 30-item test. Students of high academic ability did better - they had a 6-point advantage."

The study found "small positive advantages of 2 to 10 percent" for questions asking where (on an outline map) important events were taking place, and for questions dealing with "non-saturation events," such as news from Poland, the Soviet Union, and South Africa.

**MEETING DATES
FOR
1992**

3RD SATURDAY

17 OCT 1992 *
21 NOV 1992 *
19 DEC 1992
19 DEC 1992

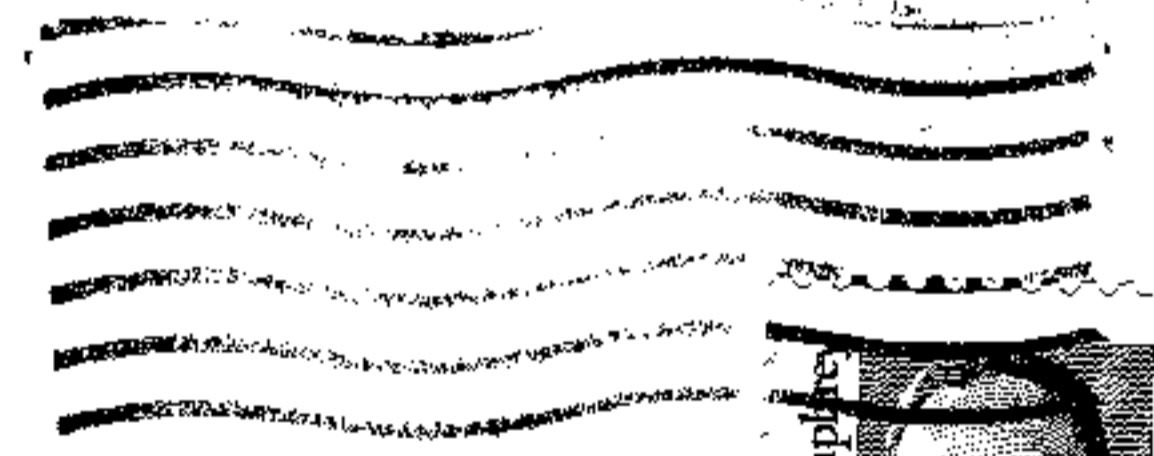
* Meeting will be held at the Janis Center due to OSU football games.

4TH WEDNESDAY

28 OCT 1992
24 NOV 1992
22 DEC 1992

C.O.N.N.I. BOARD MEMBERS

Pres. - John Parkins	614/891-4965
Treas - Everett Wade	614/262-6346
Sec/Sat - Jim Peterson	614/235-3545
Sec/Wed - Dick Beery	614/459-3597
Membership - John Parkins	614/891-4965
Librarian - Chuck Grimes	614/268-8821
Disk - Dick Beery	614/459-3597
Cassette - Everett Wade	614/262-6346
Cartridge - Ken Marshall	614/876-1670
NL Exchange - Jean Hall	614/885-4223
TIABS BBS	614/851-0708
Vice Pres. - Chuck Grimes	614/268-8821
Spirit of 99 BBS	614/263-3412
Irwin Hott	614/263-5319
Dick Beery	614/459-3597
Co-Editors/Spirit of 99 Newsletter	
Jean Hall	614/885-4223
Bob DeVilbiss	614/891-0566



C.O.N.N.I.
181 HEISCHMAN AVE
WORTHINGTON, OH 43085

**TIME SENSITIVE MATERIAL
POSTMASTER - PLEASE DELIVER PROMPTLY**

EX 12/92LD

