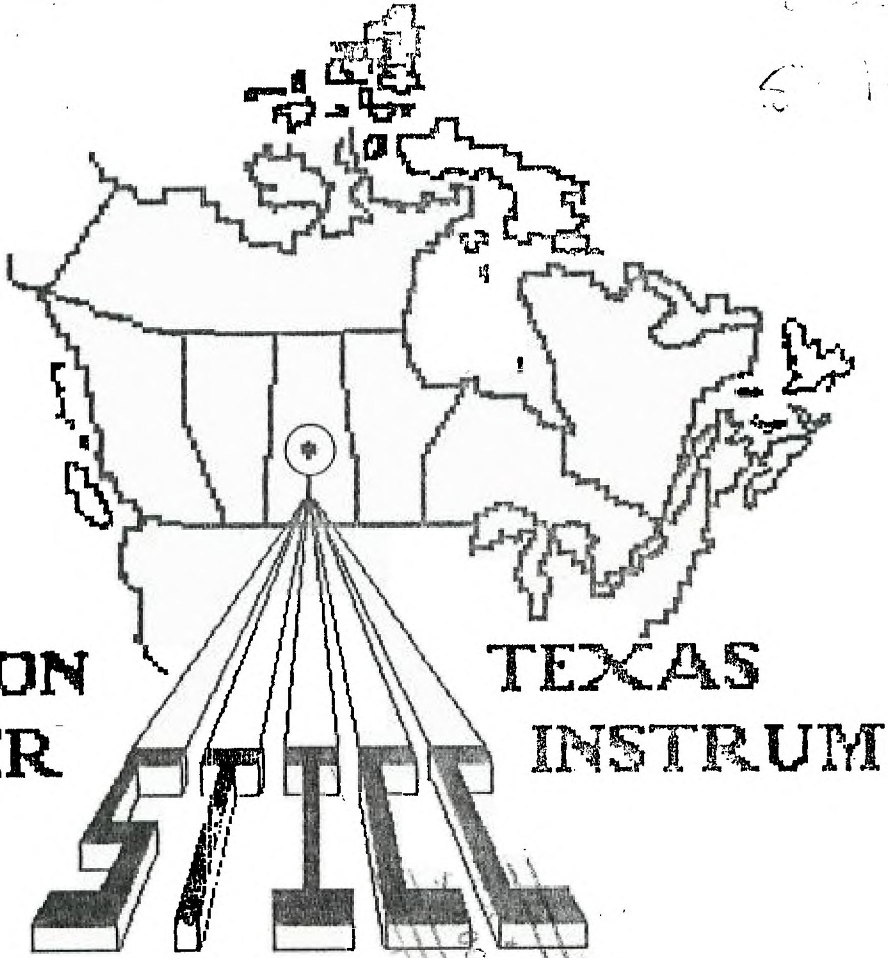


SASKATOON COMPUTER CLUB

TEXAS INSTRUMENTS

****

CLUB NEWSLETTER MARCH, 1987

****

FIRST CLASS MAIL

FROM: STICC
P.O.Box 7925
Saskatoon
Sask.,Canada
S7K 4R6

Edmonton 99'ers
P.O.Box 11983
Edmonton, Alberta
T5J 3L1

SASKATOON TEXAS INSTRUMENTS COMPUTER CLUB

We discuss and review new products for the TI99/4A while providing technical support for any problems that a member may have. We also support a software library and have regular contacts with other groups in Canada and the United States. Our membership fees are very reasonable: $10.00/single or $12.00/family. If you would like to become a member, or require more information, contact any member of the executive.

1987 EXECUTIVE COUNCIL

PRESIDENT:	Harry Caruk	384-6321
VICE PRESIDENT AND SECRETARY:	Steve Zabarylo	931-7812
TREASURER AND LIBRARIAN:	Mike Trimble	242-6215
EDITOR:	John Thomson	244-1394

MAILING ADDRESS

P.O.BOX 7925, Saskatoon, Sask., Canada S7K 4R6

STICC DISCLAIMER

We are a small, nonprofit club here in Saskatoon. We never have, or ever will, consider our newsletter a professional medium in its field. Therefore, with this consideration, the views presented in the articles, by various authors, may not be the views of the newsletter committee, the STICC executive, or its members. On this basis, this club cannot be held responsible for errors, omissions, views, or copy infringements, as presented in the articles.

MEETINGS

General meetings are normally held at 7 PM on the first monday of each month at the Saskatoon Region Community College, 145-1st Avenue North. If the first Monday of the month is a holiday (civic or statutory), the meeting will be held on the following Monday. The meeting room number will be posted on a sign at the main entrance. (Usually room 221 or 144)

```
*****************************************
*                                       *
*  SCHEDULED MEETINGS :   APRIL  6, 1987 *
*                                       *
*                         MAY  4, 1987  *
*                                       *
*****************************************
```

EDITOR'S NOTES

New Hardware: The new IBM compatible add-on for the TI99/4A marketed by Triton appears to be a turkey. Essentially it is a modest IBM clone that allows use of the TI99/4A as a keyboard. Since the TI's keyboard is not one of its strong points and is somewhat unsuitable for IBM software, this TI connection makes no sense to me. In any event, Triton are not selling it in Canada since it does not meet some Canadian standard. (probably RFI, radio frequency interference)

Another approach is being pursued by Mechatronics of West Germany. They have breadboarded a PEB card to give IBM compatibility that will utilize more of the existing TI hardware. There is no word yet when or if it will go into production.

New Software: Asgard Software, Box 10306, Rockville, MD 20850, has a product called "Font Writer" for US$25. This software (1) allows creating or editing existing TI Artist or CSGD fonts (2) has a formatter that accepts both TI Writer formatting commands and new graphics commands to allow mixing text with graphics (3) has an organization tool to assist in maintaining graphics files.

Another software package now available for mixing text and graphics ("desktop publishing") is "The Printer's Apprentice" by McCann Software, Box 34160, Omaha, NE 68134 for US$22.50. Reports indicate that this is a comprehensive and complex package of software that takes time and experience to master, but it is very powerful.

At our next meeting, April 6, Francis Gaston will demonstrate the latest version of TI Artist (Ver 2.01) as well as new companion programs, such as a slide presentation. He will also provide information on a proposed group purchase plan. A printer will be available for some "hard copies". See you there.

###

FOR SALE:

(1) Full height disk drive	$ 50
(2) Editor Assembler	$ 30
(3) Ask about inexpensive daisy-wheel printers	

Cliff Peters 382-9580

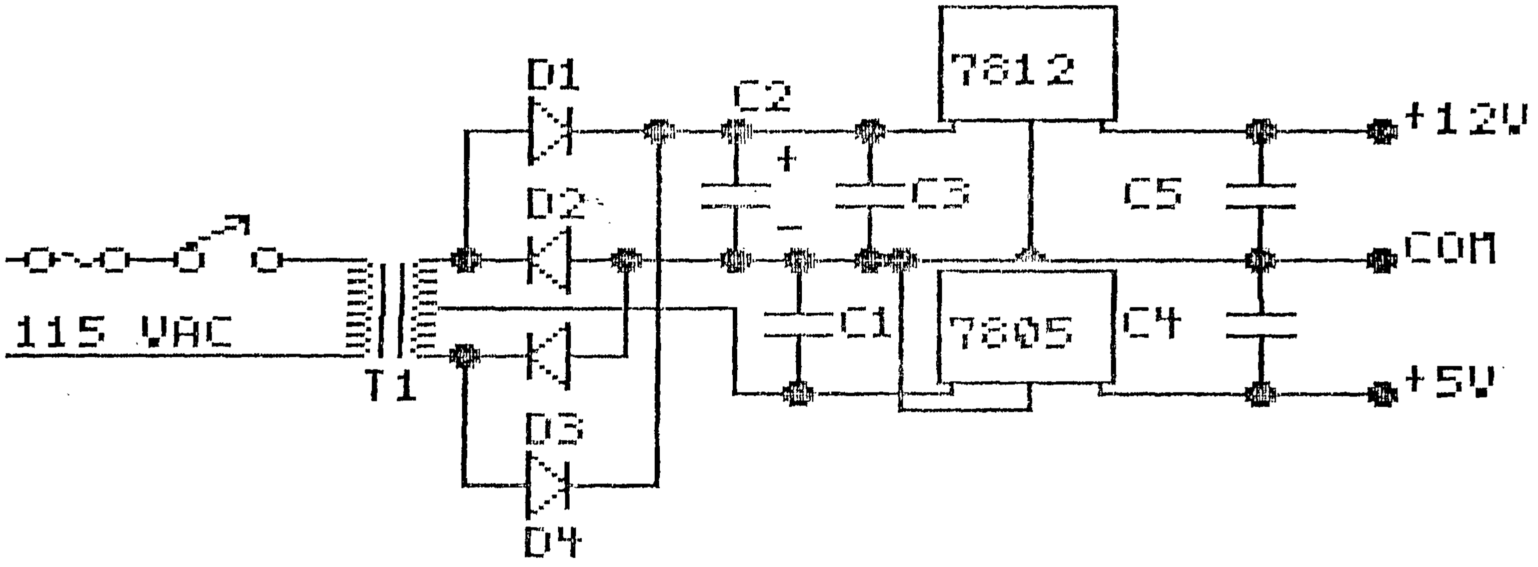
###

FOR SALE:

(1) TI Peripheral Expansion Box with Flex Cable Interface	$ 175
- with TI disk drive controller	$ 200
(2) TI disk drive, single sided,used infrequently	$ 75
(3) TI Extended Basic, with manual	$ 50
(4) TI Editor/Assembler, with manual	$ 30
(5) TI 99/4A Computer, console modified with connector for external keyboard, PLUS: external TI keyboard in a box with 8 ft cable	$ 100
(6) Book: Best of 99'er	$ 8

See John Thomson at the next STICC meeting, or call 244-1394
###

External Disk Drive Power Supply



NOTE: REGULATORS 7805 & 7812
MUST BE MOUNTED ON
SUITABLE HEAT SINK!

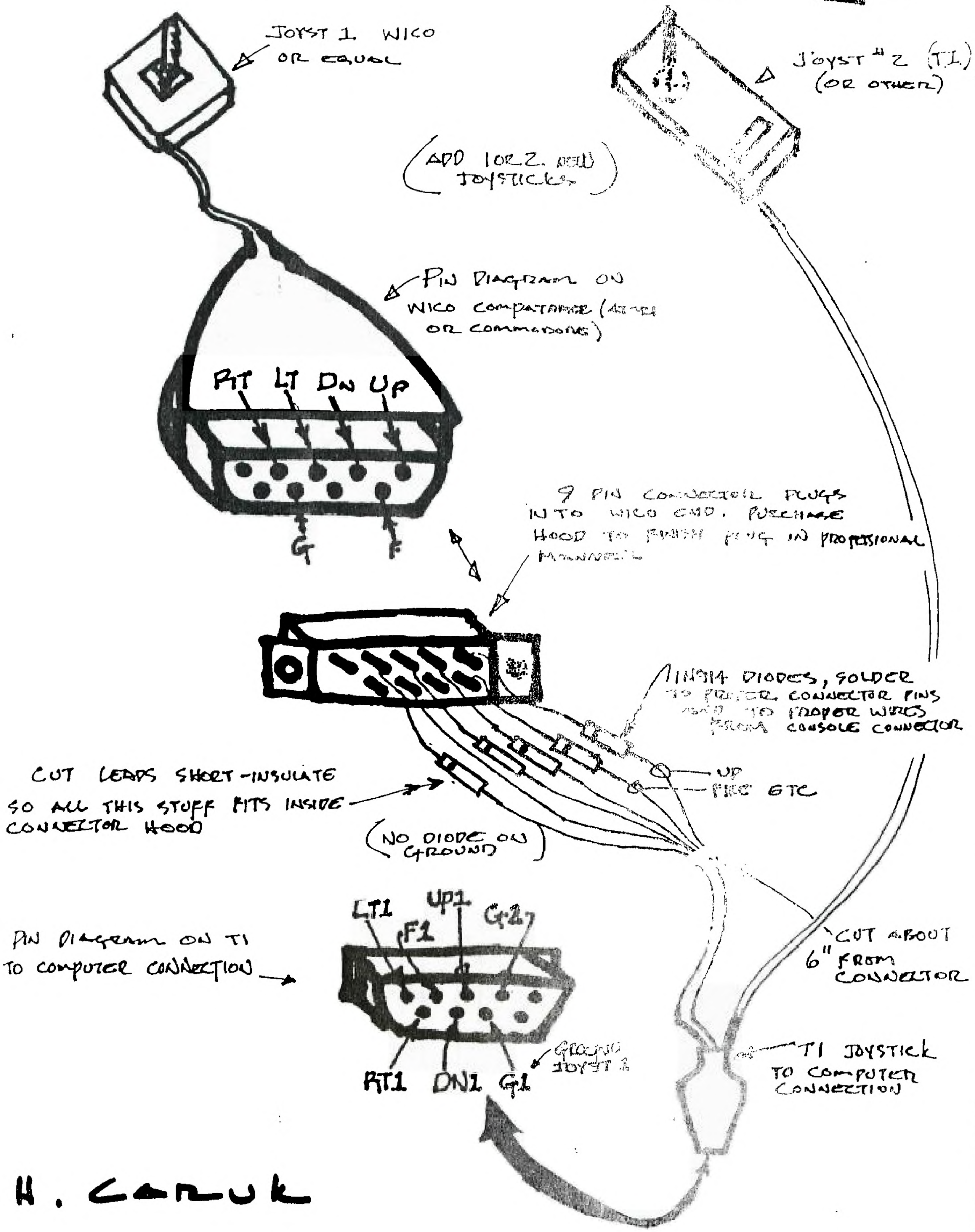
	QTY	DESCRIP.	PART NO.	COST		1987 PRICES/CANADA RADIO SHACK CATALOG
T1	1	TRNSFRMR 16VCt@2A	2731515	$6.99		$ 12.95
C1,C2,C3	3	CPCTR 2200UF@35V	2721020	$2.49	EACH	$ 3.79
C4,C5	2	CPCTR 100UF@35V	2721016	$0.79	EACH	$ 1.29
D1-4	4	3 AMP BARREL DIODE 1N5402	2761143	$0.89	FOR TWO	$ 1.59
7812	1	FIXED REG 12V	2761170	$1.59		$ 2.89
7805	1	FIXED REG 5V	2761171	$1.59		$ 2.89
			TOTALS	14.34		$ 25.40

Artwork By

Paul Desner

FEB 1987 WINNIPEG UG'S NEWSLETTER

JOYSTICK [illegible] 2([illegible])



(ADD 1 OR 2 NEW JOYSTICKS)

PIN DIAGRAM ON WICO COMPATABLE (ATARI OR COMMADORE)

9 PIN CONNECTOR PLUGS INTO WICO END. PURCHASE HOOD TO FINISH PLUG IN PROFESSIONAL MANNER

1N914 DIODES, SOLDER TO PROPER CONNECTOR PINS AND TO PROPER WIRES FROM CONSOLE CONNECTOR

CUT LEADS SHORT-INSULATE SO ALL THIS STUFF FITS INSIDE CONNECTOR HOOD

(NO DIODE ON GROUND)

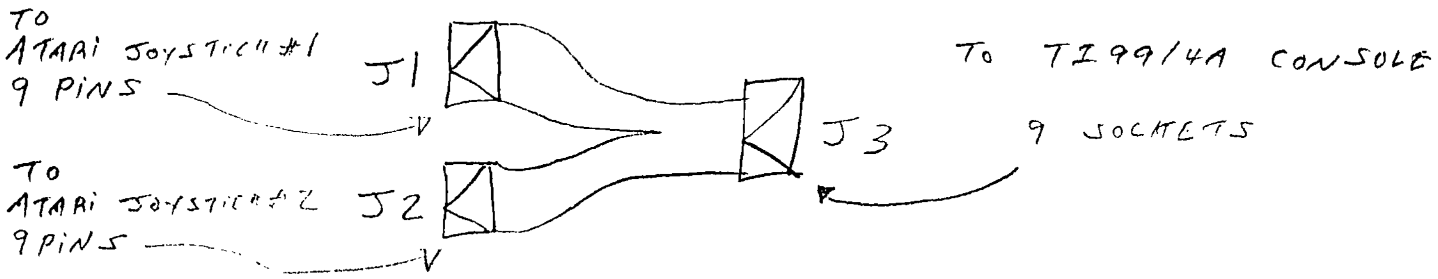
PIN DIAGRAM ON TI TO COMPUTER CONNECTION

CUT ABOUT 6" FROM CONNECTOR

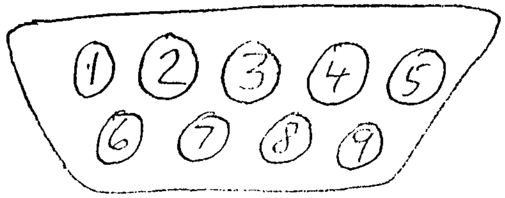
TI JOYSTICK TO COMPUTER CONNECTION

H. CARUK

ADAPTER CABLE: TWO ATARI JOYSTICKS TO TI99/4A

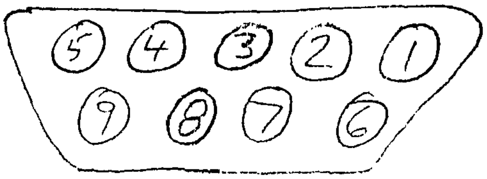


J1 & J2 PIN VIEW



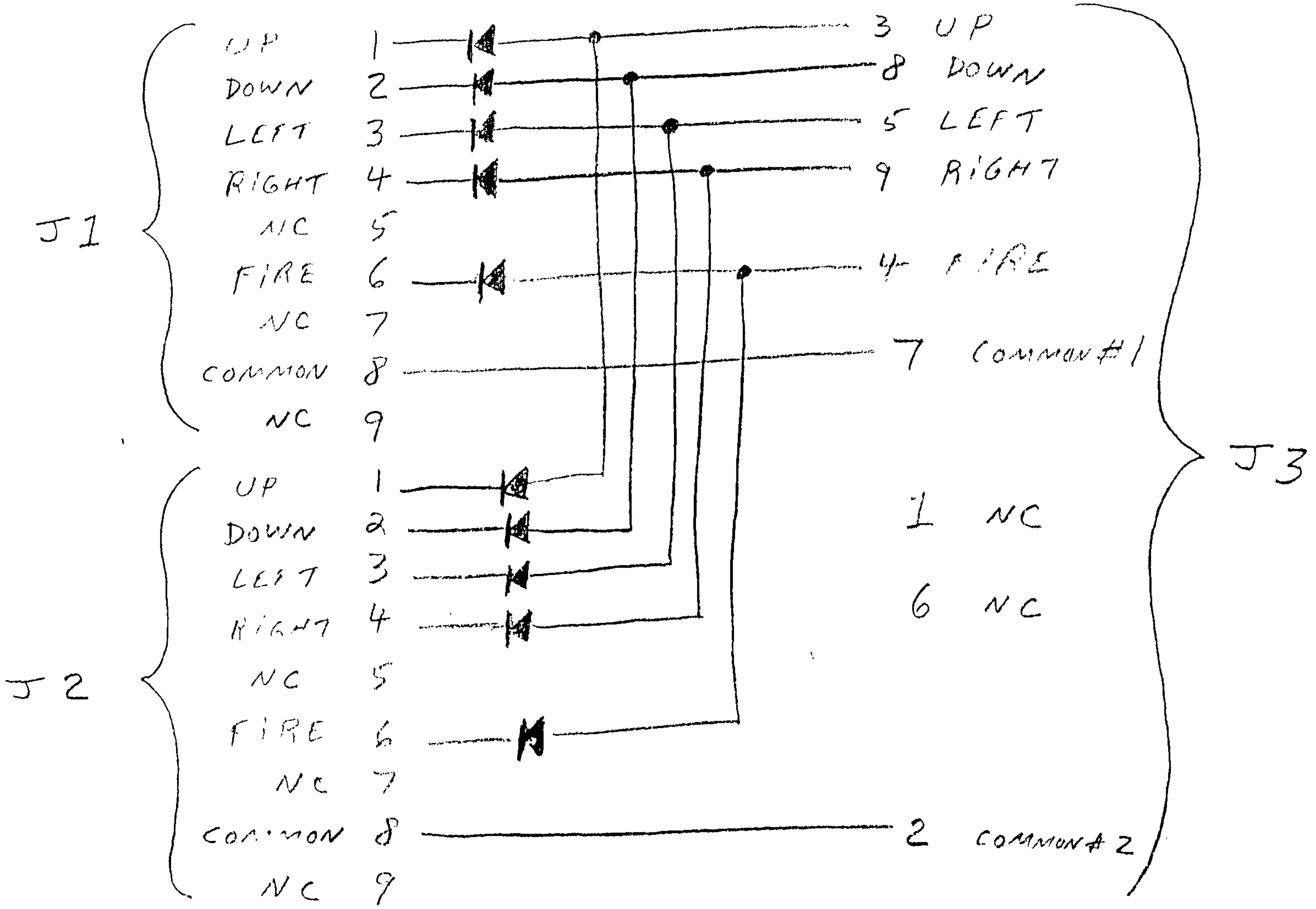
J3 REAR VIEW

J3 SOCKET VIEW



J1 & J2 REAR VIEW

SCHEMATIC:



NC = NO CONNECTION

CALL LOAD, PEEK AND POKEV ADDRESSES

SPEECH CHIP LOCATION	CALL LOAD(-27648,A,B,C)
VARY KEYBOARD RESPONSE	CALL LOAD(-31572,0-255)
LAST SOUND ON	CALL LOAD(-31700,A)
SOUND CHIP ON	CALL LOAD(-31721,A)
QUIT TO TITLE SCREEN	CALL LOAD(-31730,33)
NO AUTO SPRITE MOTION RO SOUND	CALL LOAD(-31740,192)
NORMAL OPERATION	CALL LOAD(-31740,224)
MAGNIFIED SPRITES	CALL LOAD(-31740,225)
DOUBLE SIZE SPRITES	CALL LOAD(-31740,226)
MAGNIFIED DOUBLE SPRITES	CALL LOAD(-31740,227)
MULTICOLOR MODE (48 X 64 SQUARES)	CALL LOAD(-31740,232)
SOUND REGISTERS 0-255	CALL LOAD(-31740,A,B)
SAME AS CALL SOUND A OR -A = A NUMBER	CALL LOAD(-31744,A)
FREEZE SCREEN THEN BLANK IT OUT (RESTORE BY PRESSING FCTN-.)	CALL LOAD(-31745,0)
FASTER CURSER & DURATION OF TONE A=2-7	CALL LOAD(-31748,A)
TURN OFF CURSER & DURATION OF TONE	CALL LOAD(-31748,0)
SET NORMAL CURSER & DURATION OF TONE	CALL LOAD(-31748,1)
BLANKS SCREEN AT NEXT KEYSTROKE	CALL LOAD(-31788,160)
DISABLE SPRITE MOTION AND AUTOMATIC SOUND	CALL LOAD(-31788,192)
NORMAL SPRITE MOTION AND SOUND	CALL LOAD(-31788,224)
MAGIFIED SPRITES	CALL LOAD(-31788,225)
DOUBLE SIZED SPRITES	CALL LOAD(-31788,226)
DOUBLE SIZE AND MANIFIED SPRITES	CALL LOAD(-31788,227)
PLACES YOU INTO MULTI-COLOR MODE	CALL LOAD(-31788,232)
COUNTDOWN TIMER FOR CALL SOUND FOR A=1 TO 255	CALL LOAD(-31794,A)
QUIT TO TITLE SCREEN	CAL). LOAD(-31803,35)
BYE CALL INIT :: CALL PEEK(2,A,B) ::	CALL LOAD(-31804,A,B)
CURSOR FLASH RATE (0 TO 255)	CALL LOAD(-31804,A)
RESTART PROGRAM FROM MODULE	CALL LOAD(-31804,160)
ALL BIT OFF-ENABLES AUTO SPRITE MOTION, AUTO SOUND PROCESSING AND QUIT KEY	CALL LOAD(-31806,0)
BIT 3 ON DISABLE THE QUIT KEY (FCTN=)	CALL LOAD(-31806,16)
BIT 2 ON DISABLE AUTO SOUND PROCESSING	CALL LOAD(-31806,32)
CONTINUOUS SOUND	CALL LOAD(-31806,-32)
BIT 2 ON AND BIT 3 ON	CALL LOAD(-31806,48)
BIT 1 ON DISABLE SPRITE MOTION	CALL LOAD(-31806,64)
BIT 1 ON AND BIT 3 ON	CALL LOAD(-31806,80)
BIT 1 ON AND BIT 2 ON BIT 4-7 NOT USED	CALL LOAD(-31806,96)
BIT 0 ON DISABLE AUTO SPRITE MOTION, AUTO SOUND AND THE QUIT KEY	CALL LOAD(-31806,128)
TURNS ALL THE (-31806'S) OFF	CALL LOAD(-31806,129)
DOUBLE RANDOM NUMBERS (0 TO 255) NEED RANDOMIZE	CALL LOAD(-31808,A,B)
IF A=170 THAN CALL INIT HAS BEEN EXECUTED	CALL LOAD(-31848,A)
WILL GO TO CONSOLE BASIC AFTER "NEW" COMMAND	CALL LOAD(-31860,4)
AUTOMATIC RUN OF "DSK1.LOAD"	CALL LOAD(-31860,8)
RESTARTS XB AND RUNS DSK1.LOAD	CALL LOAD(-31862,128)
END OF CPU PROGRAM ADDRESS (A6+B)OR(A6+8)	CALL LOAD(-31866,A,B)
ADD 8K TO EXPANSION SIZE	CALL LOAD(-31866,33,0)
REDUCE ACCESS TO 32K (1-159) FIRST BASIC TOKEN	CALL LOAD(-31866,A)
SAME AS ON BREAK GOTO (SYS LOCKUP)	CALL LOAD(-31868,0)
EXPANSION MEMORY OFF	CALL LOAD(-31868,0,0)
EXPANSION MEMORY ON	CALL LOAD(-31868,255,231)
SCREEN COLUMN TO START AT WITH A "PRINT"(A=3-32)	CALL LOAD(-31873,A)
VDP STATUS REGISTER A=128 =60Hz VDP INTERRURPT	
GENERATES LINES ON SCREEN	CALL LOAD(-32114,119)
GENERATES RANDOM CHARACTERS ON SCREEN	CALL LOAD(-32116,2)
TO GO FROM XBASIC TO BASIC	CALL LOAD(-32116,4)
UNPROTECTED EXTENDED BASIC PROGRAM	CALL LOAD(-32187,0)
PROTECT EXTENDED BASIC PROGRAM	CALL LOAD(-32187,1)
SET "ON WARNING NEXT" COMMAND	CALL LOAD(-32187,2)
SET "ON WARNING STOP" COMMAND	CALL LOAD(-32187,4)
SET LINE NUMBER ZERO	CALL LOAD(-32187,9)
SET "UNTRACE" COMMAND	CALL LOAD(-32187,14)
SET "UNTRACE" COMMAND AND "NUM" COMMAND	CALL LOAD(-32187,15)
SET "TRACE" COMMAND	CALL LOAD(-32187,16)
SET "ON BREAK NEXT" COMMAND	CALL LOAD(-32187,64)
PROTECT EXT. BASIC PROGRAM	CALL LOAD(-32187,128)
CHANGE COLOR AND RECEIVE SYNTAX ERROR	CALL LOAD(-32188,1)
CHANGE COLOR AND RECEIVE BREAKPOINT	CALL LOAD(-32188,1271
RESET TO TITLE SCREEN	CALL LOAD(-32630,128)
UNPROTECT TEXT. BASIC PROGRAM	CALL LOAD(-32699,01
SET "ON WARNING NEXT" COMMAND	CALL LOAD(-32699,2)
SET "ON WARNING STOP" COMMAND	CALL LOAD(-32699,4)
SAME AS CALL UNTRACE	CALL LOAD(-32699,14)
SAME AS CALL UNTRACE THEN NUM	CALL LOAD(-32699,15)
SAME AS CALL TRACE	CALL LOAD(-32699,16)
SET "ON BREAK NEXT" COMMAND	CALL LOAD(-32699,64)
SET "UNTRACE" COMMAND	CALL LOAD(-32699,120)
PROTECT EXT. BASIC PROGRAM	CALL LOAD(-32699,128)
CLEAR SCREEN FOR INSTANT	CALL LOAD(-32700,0)
RUN "DSK1.LOAD"	CALL LOAD(-32729,0)
RESET TO TITLE SCREEN	CALL LOAD(-32730,32)
RESET TO TITLE SCREEN	CALL LOAD(-32961,51)
SET "ON BREAK GOTO" LOCKS SYSTEM	CALL LOAD(-32961,149)
TO SAVE A PROGRAM LINE ERASED BY FCTN ERASE	FCTN P THEN ENTER
IF A<>70 OR A<>121 THEN DO A CALL INIT	CALL PEEK(8192,A)
A=FIRST FREE ADDRESS IN LOW MEMORY	CALL PEEK(8194,A)
A=LAST FREE ADDRESS IN LOW MEMORY	CALL PEEK(8196,A)
IF THE SPEECH SYNTHESIZER IS ON A=96 OFF A=0	CALL PEEK(-28672,A)
DOUBLE RANDOM NUMBER GENERTOR 0-255	CALL PEEK(-31808,A,B)
IF A=231 THEN 32K IS PRESENT	CALL PEEK(-31863,A)
A256+B-41023 = FREE PROGRAM SPACE IN HIGH MEMORY	CALL PEEK(-31866,A,B)
SPRITE COINCIDENCE, HIT=32	CALL PEEK(-31877,HIT)
HIGHEST # SPRITE IN AUTO-MOTION	CALL PEEK(-31878,)
VDP INTERRUPT TIMER	CALL PEEK(-31879,TIME)
RANDOM NUMBER GENERTOR 0-99	CALL PEEK(-31880,NUM)
POINTER TO THE ENDING ADDRESS OF THE NUMBER LINE TABLES	CALL PEEK(-31950,A,B)
IF A=55 THEN MEMORY IS OFF	CALL PEEK(-31952,A)
TO RECOVERY A PROGRAM ERASED BY FCTN QUIT	CALL PEEK(-31952,A,B,C,D)
:: PRINT A;B;C;D NOW PRESS QUIT CALL INIT	CALL LOAD(-31952,A B,C,D)
IS THE CURRENT LINE BEING REFEHENCED IN THE TABLE	CALL PEEK(-31954,A,B)
A256+0-2487 MEMORY SIZE	CALL PEEK(-31974,A,B)
USE POKEV(784,A)(WHERE A IS 16 TO 31) CHANGES BACKGROUND COLOR OF CURSOR	CALL POKEV(784,A)
WHITE EDGES	CALL POKEV(30945,0)
DISABLE DISK DRIVES, SAME AS CALL FILES(0) BYE TO RESET	CALL POKEV(-31888,63,255)
TEXT MODE(-32270,0,"",-39945,0 40 COLUMNS -32768,0 TO RESET)	CALL POKEV(-32272,0)

FROM NORTHERN NJ UG

```
FOR MULTICOLOR MODE                                  CALL POKEV(-32280.0)
CLEAR SCREEN                                         CALL POKEV(-32352.0)
TURN ON THE BIT MAP MODE                             CALL POKEV(-32766.0)
TURN ON NORMAL MODE                                  CALL POKEV(-32768.0)
BLANK SCREEN,KEY RESTORES                            CALL POKEV(-32352.107)
TEXT MODE ON                                 CALL POKEV(-32272.0,"",-30945.0)
WITH EXTENDED BASIC THE CONTROL KEY AND ANY OTHER KEY = SEE LIST
1= TO       2= STEP     3= ,        4= ;        5= :            6= )
7= (        8= OPTION   9= OPEN     0= THEN     /= AND
== CALL     A= ELSE     B= ::       C= !        D= IF           E= GO
F= GOTO     G= GOSUB    H= RETURN   I= DEF      J= DIM          K= END
L= FOR      M= LET      N= BREAK    O= UNBREAK P= TRACE         Q= UNTRACE
R= INPUT    S= DATA     T= RESTORE U= RANDOMISE V= NEXT         W= READ
X= STOP     Y= DELETE   Z= REM      ;=PRINT       .= ON
WITH EXTENDED BASIC THE FCTN KEY AND ANY OTHER KEY =  SEE LIST
0= XOR      Q= /\(shift 6)          /= OR         H= <          J= >
K= +        L= ~        ;= NOT      B= =          N= /          M= *
,= &       V=  (blank space)
```

Here is a neat little program to change the cursor - taken from the Hunter Valley Newsletter.

```
1 CALL INITAL :: CALL LOAD(8196.63.248)  :: CALL LOAD
(16376,67,85.82.83.79.82.48.8) :: CALL LOAD
(12288,98,82,74,70.127.8,40.56)
2 CALL LOAD(12296,2,0.3.240.2.1.48.0.2.2.0.8.4.32.32.36.4.91) :: CALL
LINK("CURSOR")
```

Line 1 CALL LOAD(12288) is developed by the follow in table.
Determine the Hex value and convert it to Decimal value.

```
                                    _HEX_        _DECIMAL_
_______________
|_|X|X|_|_|_|X|_|                    62            98
|_|X|_|X|_|_|X|_|                    52            82
|_|X|_|_|X|_|X|_|                    4A            74
|_|X|_|_|_|X|X|_|                    46            70
|_|X|X|X|X|X|X|X|                    7F           127
|_|_|_|_|X|_|_|_|                    08             8
|_|_|X|_|X|_|_|_|                    28            40
|_|_|X|X|X|_|_|_|                    38            56
```

The decimal values are used for Extended Basic program.
To write this as an Assembly Program you used the Hex values. The source code is as follow:

```
       DEF  CURSOR    Program name
VMBW   EQU  >2024     XB equate for VMBW
       AORG >3000     AORG the program to address >3000 the upper end
*                     of low memory expansion
       DATA >6252,>4A46,>7F08,>2838 data for new cursor shape
CURSOR LI   R0,>03F0 start of programm and loads the address of
*                     character 30 (the cursor) in the pattern
*                     descripter table into R0.
       LI   R1,>3000 loads the starting address of where the new
*                     character shape data is stored
       LI   R2,>0008 loads the number of bytes to read
       BLWP @VMBW     branch to VMBW routing and do it
       RT             return
       END
```

Assemble the source code and type in Extended Basic the following program.

```
CALL INIT :: CALL LOAD("DSKX.FILENAME") :: CALL LINK("CURSOR")
```

```
    A=32 = SPRITE COINCIDENCE
    A=64 = 5 SPRITES ON A LINE                     CALL LOAD(-31877.A)
STOP SPRITE MOTION SLOWER THEN -31806              CALL LOAD(-31878.0)
START SPRITE MOTION (A=HIGHEST # SPRITE)           CALL LOAD(-31878.A)
FOR A=0 TO 255. TIMER FOR VDP INTERUPTS
    EVERY 1/60 OF A SECOND                         CALL LOAD(-31879.A)
FOR A=1 TO 99, RANDOM NUMBER, NEEDS
    "RANDOMIZE"                                    CALL LOAD(-31880.A)
SELECTS KEYBOARD MODE AS IN "CALL
    KEY(K,...); FOR A=0 TO 5                       CALL LOAD(-31884.A)
SAME AS BYE                                        CALL LOAD(-31885.255)
VDP STATUS REGISTER
  A=32=SPRITE COINCIDENCE  A=64= 5 SPRITES ON SAME LINE
      BIT 0=THE 60Hz INTERRUPT
      BIT 1=(64) IF MORE THAN 4 SPRITES IN A ROW
      BIT 2=(32) IF THERE IS A SPRITE COINCIDENCE
      BIT 3-7=CONTAINS THE HEX # OF THE 5TH SPRITE IN A ROW
             (BIT 1 MUST BE ON)                    CALL LOAD(-31887,A)
TO TURN DISK DRIVE ON      BYE OR CALL LOAD(-31888.55.215)  THEN EDIT
CALL FILES(2)                                      CALL LOAD(-31888.57,221)
CALL FILES(1)                                      CALL LOAD(-31888.59.227)
SAME AS CALL FILES(0)              CALL LOAD(-31888.63.255)  THEN EDIT
UNPROTECTED EXTENDED BASIC PROGRAM                 CALL LOAD(-31931,0)
SET "ON WARNING NEXT" COMMAND                      CALL LOAD(-31931,2)
SET "ON WARNING STOP" COMMAND                      CALL LOAD(-31931,4)
SET "UNTRACE" COMMAND                              CALL LOAD(-31931.14)
SET "UNTRACE" COMMAND AND "NUM" COMMAND            CALL LOAD(-31931.15)
SET "TRACE" COMMAND                                CALL LOAD(-31931.16)
SET COMMAND "ON BREAK NEXT"                        CALL LOAD(-31931.64)
RESULTS IN XB *DONE*                               CALL LOAD(-31931.120)
PROTECT EXT. BASIC PROGRAMS                        CALL LOAD(-31931.128)
REMOVE READY STOPS EXECUTION                       CALL LOAD(-31932.0)
RETURN TO TITLE SCREEN                             CALL LOAD(-31961.51)
SAME AS RUN"DSK1.LOAD"  CALL LOAD(-32729.0) OR CALL LOAD(-31961.149)
RETURN TO TITLE SCREEN                             CALL LOAD(-31962.32)
GOES DIRECTLY INTO BASIC                           CALL LOAD(-31962.33.111)
DELETES PROGRAM FROM MEMORY "NEW"                  CALL LOAD(-31962.99)
RESTARTS XB AND RUN"DSK1.LOAD"                     CALL LOAD(-31962.99.114)
EXECUTES "RUN" COMMAND                             CALL LOAD(-31962.100.155)
EXECUTES "NEW" COMMAND                             CALL LOAD(-31962.100.124)
EXECUTES "CON" COMMAND - FROM COMMAND MODE ONLYCALL LOAD(-31962.100.126)
EXECUTES "LIST" COMMAND-FROM COMMAND MODE ONLY CALL LOAD(-31962.100.128)
EXECUTES "BYE" COMMAND -CLOSES ALL OPEN FILES  CALL LOAD(-31962.100.130)
EXECUTE DEFAULT "NUM" COMMAND -WHEN RUNNING PROGRAM ENDS....
 LINE 100 WILL CONTAIN GARBAGE SO JUST PLACE
   REM THERE.                                      CALL LOAD(-31962.100.132)
EXECUTES "RES" COMMAND                             CALL LOAD(-31962.100.136)
EXECUTES "LIST" COMMAND-FROM COMMAND MODE ONLY CALL LOAD(-31962.101.190)
    SAME AS -31962,100,128
GENERATES COLORFUL TITLE SCREEN                    CALL LOAD(-31962.160.000)
EXECUTES "RUN" COMMAND WITHOUT PRE-SCAN (FASTER THAN HAVING
   A "RUN" COMMAND IN YOUR PROGRAM TO
   RESTART IT.)                                    CALL LOAD(-31962.160.04)
RESTART EXTENDED BASIC                             CALL LOAD(-31962,255)
LAST LINE ADDRESS                                  CALL LOAD(-31982.1)
SEARCHES DISK1 FOR 777                             CALL LOAD(-32112.8)
GENERATES RANDOM GARAGE ON SCREEN                  CALL LOAD(-32114,2)
GENERATES A TOTALLY "WILD" SCREEN                  CALL LOAD(-32114,13)
```

FROM NORTHERN NJ UG

A GRAPHIC DILEMMA... SOLVED!

I recently purchased a new set of fonts and clips from Asgard Software called Companion III. This package of new artwork was,of course, intended for use with Asgard's fantastic Graphx. I, like most people, however, split my time between Graphx and TI-Artist because both have features that the other lacks. I really need BOTH to turn out good graphic work for the Interface. Generally, this is no problem because Artist allows me to convert pictures from Graphx and Alphabet fonts from CSGD. Unfortunately there is no way to convert a font from Graphx to Artist, and I prefer Artist for font work (usually). I presented the problem to Bob Coffey and, after much contemplation and a sleepless night, he got back to me with the solution. I felt that many people may be wishing for the answer to this transfer problem and asked Bob to document it. It's not as hard as it sounds so give the following a try.

-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-

TI-Artist Font Format

by Robert Coffey Jr.

When you load a TI-Artist font style into the TI-Writer editor, you will see a basic pattern which is very similar to the format of an Artist Instance!

```
INSTANCE:

A,B
0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0

FONT:

n
A,B,E
0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0
```

A and B refer to the (8*8) blocks that will define the Instance or Font letter. A will be how many blocks across, and B will be how many blocks down!

n equals the character being defined in the font.(A;B;C etc)

E is equal to the numbers of pixels wide the character is plus a pixel of space between the font letters.

A*B will equal how many lines it will take to define the character or instance. Each definition line (the lines that show as "0"s here) contains 8 numbers, ranging from 0-255.

Artist Font Design

For the sake of time and space we will assume that you have already either drawn a full font as an Artist picture or have converted a font picture from Graphx to Artist.

(1) When you set your letters in the picture, leave 5-7 pixels on the bottom and right side of your letters! You may seperate letters with the Move feature of Enhancement if you didn't leave enough space between them.

(2) After you've done this, find your largest and widest character. Write down the pixel height and width. Also write down the WIDTH plus *one*, (1), of ALL the characters. The extra pixel is for space between the characters. If you don't have eagle eyes, use the Zoom feature to count them.

(3) Save all your characters in 1 picture if possible, else give them sequential names like PIC1, PIC2.

(4) Copy your reference pictures to a clean disk! Things will get VERY MESSY if you don't!

(5) Go into Enhancement and press [S] for the slides section, next press 7 to save a Instance.

(6) Use a file name that fits whatever character type you may be working on, such as : a number 1,2,3... UPPERcase UA,UB,UC... lowercase LA,LB,LC... SYMBOLS !,/,#,$...
Once the file is named you will be returned to your picture.

(7) now use the joystick and get to the upper left-hand corner of your character. When you start to increase the size of the Instance box, the top line of the box should cross over the top-most pixel(s) of your character, same for the left-most pixel(s). Remember, whatever is UNDER the box line will be included!

(8) The box will increase by 8 pixels at a time. Increase the box so that it covers the entire character. (if you find that the box also covers a part of another character, then go back and move it so it doesn't.) Use the least amount of space as possible to cover the letter!

(9) When you have covered the entire letter, hit the fire button, the letter will automatically be saved under the entered name.

(10) REPEAT STEPS 6-9 until you have saved all the characters that you want to use in your font style!

****

From West N.Y. 99ers VIA NEW JUG NORTH

ASSEMBLING THE FONT

(1) You will now start to assemble your font file. Use the EDIT SECTION OF TI-WRITER. Place the character of whatever one you are going to load in first, (probably "A") on line 0001, note the line you are at.

(2) Hit Function 9 (BACK), so that you now are back in the command mode of the editor.

(3) Type the LF command (Load File), then using this format below load the first character Instance:

```
              [xxx DSKx.A_I]
                 :       :
                 :       Instance file
       line number
   you put the font char on
```

This will load the character instance into memory after line xxx. Using the LF command this way will allow you to load D/V 80 files (Instances!) without disturbing the data that you have in the editor already!

(4) After you have loaded in the Instance at the end of your file (line xxx), you will need to add a number to the line that has only 2 numbers on it! It will look something like this:

2,3

After the second number in that line type a comma and the WIDTH number that you wrote down for the character you are working on. Say the number is 9, it will look like this:

2,3,9

(5) Go back to step (2) until you have finished loading and altering all your characters for your font style.

WARNING*WARNING: If you are doing it right, you are working WITHOUT carriage returns. YOU DON'T WANT THEM! So for Pete's sake and your own.. DON'T EVER HIT REFORMAT!

NOTE: It is a good idea to ALWAYS include a space character in every font you do. To do this put a blank line at the end of your assembled file (don't erase this one!) and load any character Instance after that blank line. Take the width of your widest character and add it to the 2 number line(like we did before). Now change those other lines that loaded in to all 0's. Keep the same number of numbers, but change them to 0's.

(6) You will then go through your file and make sure that there are NO BLANK LINES (except the space character), or C/Rs at the ends of lines. Also, double check that those lines that had 2 numbers now have 3!

(7) Now that you've double checked everything, hit Function 9 (BACK). Type the PF command. (Print File) Type the filename that you would like to call your font. Use this format:

DSKx.nnnnnn_F

(Remember, you do not want to SAVE FILE, you want to PRINT FILE to disk.)
You have now created your very own font! Now go into the Enhancement part of TI-Artist and load your font, and see how good it looks, you may need to alter some of your characters. If everything doesn't look satisfactory, then continue to step 8.

(8) Go into the Enhancement section of TI-Artist, and load your font. Get all your characters onto the screen.
There should be 1 pixel spacing between your characters. If your characters touch the one on the left may have been defined incorrectly, when you saved it as an Instance.

(9) Go into the [S] Slides section and re-save that character, making sure that the left side of the box goes over the left-most pixel of the character.

(10) If your characters are not level, then you may need to see which characters are too high. Write down all the characters that need to be lowered.

(11) Go back into Enhancement, and re-save those characters as Instances, and make sure that you start 1 (or more) pixel higher than last time when you re-save it! Keep doing this until you've corrected all the faulty characters.

(12) After you have re-saved all the characters that were not right, go back to the Editor in TI-Writer. Load your Font file and Scan through the file until you find the character that you want to correct.

(13) Delete the definition lines below the 3 number line. Note the line that the 3 number line is at! Load your saved instance using the same format as before: xxxx DSKn.xxxxxxx_I

(14) Delete the line that has been loaded that only has 2 numbers on it.

You have now (hopefully!) corrected that character, if not, do it again. Lower case characters and symbols can cause you problems on centering, etc., so a little experience may be necessary to get things right, but a little common sense will prevail.
Learning to do this could open the door to a lot of possibilities... who says a font has to look like an alphabet?!

FROM West NY 99ers - VIA NEWSUG NORTH