40 COLUMN UTILITIES FOR XB

VERSION 2.5 FEB 1992

Provides additional commands to allow the use of a 40 column screen in addition to the normal screen of Extended Basic.

A graphics screen dump that functions in 32 or 40 columns is also provided.

40_UTIL VERSION 2.3
REGISTRATION FORM

User's Name: _____ Date: _____

Address: _____

_____

_____

Number of copies to register: _____

Amount enclosed: $_____

Where did you obtain your copy of 40_UTIL?:

_____

Please note that these utilities are not intended to be 'free'. They are not public domain either. The closest name I know for this distribution method is 'Fairware'. This means that you get to try out the routines, and if you find them useful you should send me something in return. You are of course asked to make copies for your friends, etc., to help distribute the utilities. I am suggesting an amount of $5.00. I think that it's a fair amount, I'm not looking to get rich, you know! Always pass along the 40_UTIL disks as you received them, without modifications.

Please send this registration form and payment to:

BRAD SNYDER

WALNUTPORT PA 18088
USA

If you would like me to provide the disk and mailer, I ask that you pre-pay for the utilities by sending me $5.00 + $1.50 for mailer and postage.

You are allowed to use the utilities in programs that you are distributing, all I ask is that you register, give me a plug in your docs, and don't provide documentation for my utilities with your program.

INTRODUCTION

First of all, I would like to thank Barry Boone for writing Systex, a program that converts Object Code into an Extended Basic program. This allows for much quicker loading of my utilities. I would also like to thank Miller's Graphics for publishing their great DSRLNK & GPLLNK. It was necessary to have these routines to add the graphics screen dump to my utilities. MG's routines are the most efficient in terms of memory usage, and I want to leave as much of the low memory available as possible for user assembly routines.

The 40 column utilities differ from other similar packages in that you may have two screens in memory at one time. One standard graphics and one 40 column text. You may switch between screens with simple commands, and the switch appears instantaneous. It doesn't matter if you have a 40 column text screen displayed while you are drawing a graphics screen, or vice versa. You just have to issue the proper command when you want to show the screen. Also, ALL of Extended Basic's commands are still usable, with the small limitation that you don't use MAGNIFY or SCREEN while the 40 column screen is displayed.

History:

version 1.0 - worked with the E/A cart. in Basic.
2.0 - first XB version, two screen capability.
2.1 - the REPS parameter in HORZ, VERT, and GSTRN is now optional. 3 string manipulation commands are added.
2.2 - a positive size in ACCEPT now clears the input field. New keys to terminate ACCEPT have been added. The KEYVALUE and OFFSET parameters in ACCEPT are optional.
2.3 - the screen dump is added. also made DSRLNK and GPLLNK available to user assembly routines. XBEEP, XHONK and SCRL32 removed. SCROLL works in 4 directions.
2.4 - LOWCAS now load a CHARA1 type file. Note: this version never distributed.
2.5 - Word Wrap added to ACCEPT, SETUP will tell you if the VDP is already initialized, CLOCK routine is now built in. HOTKEYs added for the screen dumps.

Requirements:

Extended Basic
32K memory expansion
disk system - has been tested with TI, CorComp and Myarc disk controllers, and the Myarc Hard and Floppy Disk Controller.

The utilities and your program can be saved to cassette, but to run them again, you will need at least XB and

the 32K. The disk controller wouldn't be necessary.

The utilities are NOT guaranteed to be compatible with the Geneve. Especially with the HOTKEYs active.

Once again, If you are going to distribute a program that uses the utilities, please don't document their use with your documentation. People may assume that since they have paid for your program, they may use the utilities as if they have paid for them.

INDEX

40 COLUMN ROUTINES

CHAPTER ONE

LOADING INSTRUCTIONS

LOADING INSTRUCTIONS:

The quickest way to load the utilities is to 'RUN' the '40_XB' program. After it runs, type in NEW and press enter. The 40_XB program loads the utilities and then does 'SETUP'.

An alternate file called '40_UTIL:XB' is also provided. To use type in CALL INIT::CALL LOAD("DSK1.40_UTIL:XB") and then press enter. After a long delay while the utilities are loading, type in CALL LINK("SETUP") and enter, then NEW and press enter. I don't recommend loading in this manner because of the lengthy loading delay, but I am providing this file for maximum flexibility to the user.

No matter which way you choose to load the utilities, either NEW must be typed in, or another program RUN. I recommend copying 40_XB to another disk, renaming it LOAD, and adding a line to it such as 100 RUN "DSK1.YOURPGM". This way your program will autoload, and the utilities will already be loaded. This is the method that the DEMO program uses to load.

Check out the files LOAD and ELIZA for examples of loading the utilities.

It is required that you type NEW or run another program after SETUP because SETUP changes pointers to make room for the additional screen, and XB doesn't update it's pointers until you type in NEW, or it does NEW itself before it runs another program.

CALL FILES:

The CALL FILES command may be used before or after SETUP has been used. If you write a program that needs more files than 3, or if you need more stack space free by setting less than 3 files, you will need to use CALLS FILES. The best way is to type in CALLS FILES(?) at the XB prompt, then type in NEW, then RUN "DSK1.LOAD" if you have set up the load program as I have recommended.

40 COLUMN ROUTINES

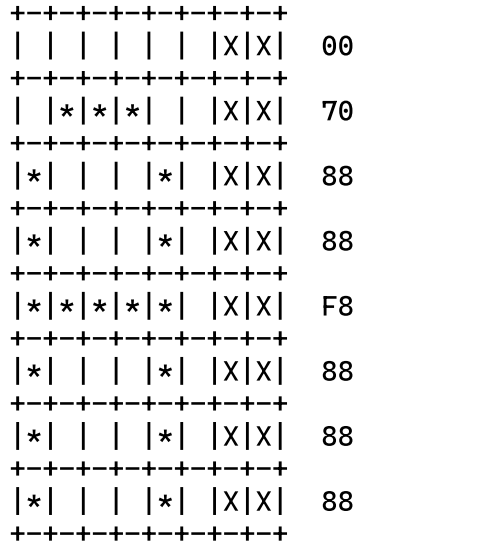
CHAPTER TWO

40 COLUMN COMMANDS

INTRODUCTION TO THE 40 COLUMN COMMANDS

These commands are not intended to improve on TI's screen commands, but to give you 40 column text capability. All of TI's commands are available, but will go to the 32 column graphics screen, not the 40 column screen. The commands in these utilities will go only to the 40 column text screen. You may have a graphics screen displayed while you are printing a 40 column text screen. Only the graphics screen will be shown on the monitor. When you are ready to display the text screen use the command CALL LINK("TEXT",16,5), this command would also set the foreground color to 16 (white) and the background color to 5 (dark blue).

You may use graphics with the 40 column screen but you must remember that only 6 columns of the pattern definition will be used. The standard CALL CHAR(.....) command is used to redefine characters. The same characters are used by both screens.



The columns with 'X' in them are ignored by the 40 column screen. When you make up the pattern definition, you must make up the numbers as if the columns are there, as shown in the example at the left.

COMPATIBILITY with versions earlier than 2.2: The SIZE parameter of the ACCEPT command may cause a problem. Before this version, only a positive value was allowed and the input field was never blanked. Now a positive size will blank the field, and a negative won't. If you don't want to make too many changes to an existing program, but you want to use the new version of 40_UTIL, change the SIZE in accept to -SIZE.

XBEEP, XHONK, and SCRL32 have been removed to save memory. They were only two sound and a scroll of the graphics screen. To replace XHONK use CALL SOUND(167,218,0), and for XBEEP use CALL SOUND(167,1398,4). These call sounds will sound just like XB's beeps and honks. To replace SCRL32, simply use PRINT. I hope that the removal of these commands is not an inconvenience for anyone.

In the descriptions of the commands that follow, in the syntax the brackets ([]) mean that the parameter is optional.

ACCEPT / ACBEEP COMMANDS

Allows input from the 40 column text screen.

SYNTAX:

CALL LINK("ACCEPT",ROW,COL,SIZE,CK$,RET$[,K[,OFFSET]])

ROW is the starting row of the input field and can range from 1 to 24. Can be constant or a variable

COL is the starting column of the input field and can range from 1 to 40. Can be constant of a variable

SIZE is the size of the input field. Can range from 1 to 255. If it is a positive number the input field will be blanked prior to accepting input. If it is negative, whatever is in the field will still be there. This works just like XB's ACCEPT AT. Can be a constant or a variable.

CK$ is a string of characters used to validate the keypresses during input. If CK$ is null ("") then any key will be accepted. For example if CK$="0123456789", then only numeric digits will be accepted. CK$ may be a string variable or a constant (string entered directly into the command).

RET$ is the string that the user entered at the keyboard. Of course it has to be a variable.

K if used, is the value of the key that the user pressed to terminate the ACCEPT command. For example if ENTER was pressed, K will equal 13 after ACCEPT is processed. The keys that can be pressed to terminate ACCEPT are:

key pressed	K=
FCTN E (up arrow)	11
FCTN X (dn arrow)	10
ENTER	13
FCTN 4 (clear)	2
FCTN 6 (proceed)	12
FCTN 9 (back)	15
FCTN 8 (redo)	6
FCTN 7 (aid)	1

K must be a variable.

OFFSET if used, is both an input and return from ACCEPT. Upon calling ACCEPT, it specifies the cursor position from the start of the input field. On return, it is the offset

from the start of the field that the cursor is currently at.

```
100 CALL LINK("DISP",1,1,"DSK3.F    displays DSK3.FILE on
    ILE"):: OFFSET=5                screen. Sets OFFSET to 5

110 CALL LINK("ACCEPT",1,1,15,""    Get input starting at
    ,RET$,K,OFFSET)                 D in DSK, but the cursor
                                    will be on the F in FILE
```

as if you had already used the arrow key to get there. OFFSET is the offset of the cursor position, so it gets added to the start of the input field to get the cursor position. OFFSET must be a variable. And if you use OFFSET, you must use K.

The editing keys in ACCEPT and ACBEEP are:

```
          FCTN 1 (delete)  - deletes one character at the
                             cursor.
          FCTN 2 (insert)  - inserts a space at the cursor.
          FCTN 3 (erase)   - erases the input field.
          FCTN 5 (begin)   - moves the cursor to the
                             beginning of the field.
          FCTN S (lt arrow)- moves cursor 1 space to left.
          FCTN D (rt arrow)- moves cursor 1 space to right.
```

When a key is held for more than a second, the key will start repeating like XB. But the longer that you hold the key down, the faster the repeat will be.

The ACBEEP command functions exactly the same as ACCEPT, but with a beep when it is called.

CLS

Blanks the 40 column screen.

SYNTAX:

CALL LINK("CLS"[,CHAR])

CHAR is the ascii value of the character to be used to fill the screen. Range is from 30 to 143. Constant or variable.

If CHAR is not used, CLS will clear the 40 column screen by writing a space character to all 960 locations. If CHAR is used, that character will be written to all screen locations.

DISP

Used to DISPLAY AT on the 40 column screen.

SYNTAX:

CALL LINK("DISP",ROW,COL,MSG$)

ROW is the starting row position on the 40 column screen. Range is from 1 to 24. Can be a variable or a constant.

COL is the starting column position. Range is from 1 to 40. Can be a variable or constant.

MSG$ is the string to be displayed. Can be variable or constant.

Displays a string on the 40 column screen. If the string goes past the right edge of the screen, printing continues on the next screen line. If the next line would be off the bottom of the screen, the screen will be scrolled up one line and printing will continue on the last line.

NOTE: this command does not use the limits set by the SCINIT command, in other words it doesn't do windows. It wraps around the full 40 column screen, and the whole screen is scrolled if needed.

GSTRN

Reads characters from the 40 column screen into a string.

SYNTAX:

CALL LINK("GSTRN",ROW,COL,RET$[,REPS])

ROW is the starting row. Range from 1 to 24. Variable or constant.

COL is the starting column. Range if from 1 to 40. Can be a variable or constant.

RET$ is the string that was read from the screen. Must be a variable.

REPS if used, is the number of characters to read from the screen. Range is from 1 to 255. Can be a variable or back to the top of the screen. Will wrap around from the right edge to the left.

This command will read the number of characters specified from the screen, either until all characters specified are read, or the end of the screen is reached. Will not wrap back to the top of the screen. Will wrap around from the right edge to the left.

HORZ

Places a character on the 40 column screen and optionally repeats it horizontally.

SYNTAX:

CALL LINK("HORZ",ROW,COL,CHAR[,REPS])

ROW is the starting row. Range 1 thru 24. Variable or constant.

COL is the starting column. Range 1 thru 40. Variable or constant.

CHAR is the ascii value of the character to place on the screen. Range 30 thru 143. Variable or constant.

REPS, if used is the number of times to repeat the character. Range is 1 to 960. Can be a variable or a constant. If not used, only one character will be displayed.

Wraps around the screen the same as HCHAR. Only difference is the limit of 960 reps. Why would you want to do more than a full screen anyway?

NORM

Displays the graphics mode screen.

SYNTAX:

CALL LINK("NORM")

Returns you to the normal screen mode of TI X-basic. Use CALL SCREEN and CALL MAGNIFY when in this mode.

The MAGNIFY that you set for sprites will be retained when switching between text and graphics screens, but you will have the same screen color when you switch back to NORM as you had when in TEXT mode.

PRINT

Prints to the window defined with the SCINIT command.

SYNTAX:

CALL LINK("PRINT",MSG$)

MSG$ is the string to be displayed. Variable or constant.

The window is first scrolled up, and then the string is printed to the bottom line of the window. If the string is longer than the line, the window will be scrolled again, and printing will continue on the bottom line again. This will continue until the entire string has been printed.

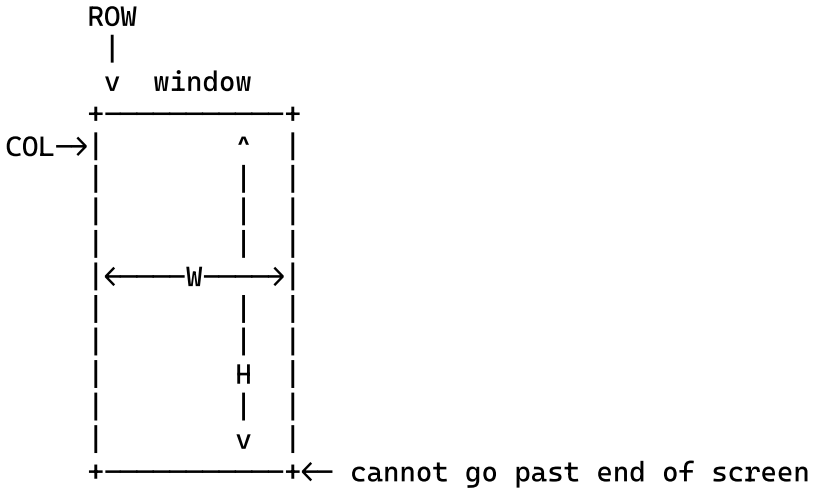
PRINT can be used to print a string sideways on the screen by defining a window only one column wide, then using print. The one column wide window will be scrolled once for each character in the string, and the string will be read on the screen from top to bottom.

SCINIT

Sets up a window on the screen.

SYNTAX:

CALL LINK("SCINIT",ROW,COL,W,H)



ROW is the starting row of the window. Range is from 1 to 23. Can be a variable or constant.

COL is the starting column. Range is from 1 to 40. Variable or constant.

W is the width of the window. Range is 1 to 40. Variable or constant.

H is the heigth of the window. Range 2 to 24. Variable or constant.

This routine sets the size and location of the print window on the 40 column screen. Only PRINT and SCROLL use the window.

The end of the window cannot go past the end of the screen (lower right hand corner) or an error is returned.

If your program is going to use PRINT or SCROLL you must use SCINIT at least once before printing to the 40 column screen. Even if you don't want a window, but want to use the whole screen, you should set the window to the full screen by using CALL LINK("SCINIT",1,1,40,24).

SCROLL

Scrolls the window defined with SCINIT.

SYNTAX:

CALL LINK("SCROLL"[,DIR[,REPS]])

DIR controls the direction of the scroll. Range is 1 to 4. Can be variable or constant.

1 - scroll up (same as CALL LINK("SCROLL")
2 - scroll down
3 - scroll right
4 - scroll left

REPS specifies the number of lines to scroll. The range is 1 to 40. Can be a variable or constant. You must use DIR to use REPS.

Moves everything in the window one column or row in the direction specified. The row or column scrolled out of the window is lost. The scrolling will continue for the number of REPS specified.

SETUP

Gets the VDP memory ready for the 40 column screen and for screen dumps.

SYNTAX:

CALL LINK("SETUP")

This command will reserve space in the VDP (console) memory for the additional screen image table. It does something similar to CALL FILES, but doesn't change the number of files available to you, only the amount of 'stack' memory available. You can check the amount of stack memory with XB's SIZE command.

You must enter NEW, or have a load program RUN another program after using this command, as described in the

loading instructions.

Setup can be called more than once, it knows not to change memory pointers after the first time.

Setup also installs the external links to GPLLNK and DSRLNK at >A000. The routines are for use from assembly language.

TEXT

Displays the text mode screen, and sets the colors.

SYNTAX:

CALL LINK("TEXT",F,B)

F is the foreground color. Range 1 to 16. Variable or constant.

B is the background color. Range 1 to 16. Variable or constant.

The numbers for the colors are the same as in XB graphics mode.

When in the text mode there is only one foreground color and the background color is the screen color. The 40 column mode does not use the color sets. This is a function of the 9918A video processor in our consoles, not my utilities.

VERT

Places a character on the 40 column screen and optionally repeats if vertically.

SYNTAX:

CALL LINK("VERT",ROW,COL,CHAR[,REPS])

ROW is the starting row. Range 1 to 24. Variable or constant.

COL is the starting column. Range 1 to 40. Variable or constant.

CHAR is the ascii value of the character to place on the screen. Range is 30 to 143. Variable or constant.

REPS, if used, is the number of times to repeat the character. Range is 1 to 960. Can be a variable or constant. If not used, only one character will be displayed.

Wraps around the screen the same as VCHAR. The only difference is the limit of 960 repetitions.

40 COLUMN ROUTINES

CHAPTER THREE

OTHER UTILITIES

CAPS

Raises an ascii string to upper case.

SYNTAX:

CALL LINK("CAPS",OP$)

OP$ is the string of characters that you want converted to upper case. Must be a variable because it is modified.

If there are any lower case letters in OP$ they will be replaced with capital letters.

DUMP - Epson compatible only.

Does a graphics screen dump of the 32 column or 40 column screen. Screen to dump doesn't have to be the one displayed.

SYNTAX:

CALL LINK("DUMP",DEV$[,SIZE[,DENS[,SCRN[,IND]]]])

DEV$ is your printer port name. For example: "PIO.CR". The CR must be used in whatever device name you use. Can be a variable or constant.

SIZE, if used specifies single size or double size printout. 0=single size, 1=double size. Can be a variable or constant. If not specified, single size will be used. If double size is selected, the IND option will be ignored because the printout will fill the width of the paper.

DENS, if used, specifies the density that you want to print in. 0=single density, 1=double density, 2=quad density. Can be variable or constant. Will not change the size of the printout, but will affect the darkness. If not specified, the printout will be in single density. To use DENS, you must use SIZE.

SCRN, if used, selects the screen to dump to the printer. 0=the 32 column screen, 1=the 40 column screen. Can be a variable or constant. Again, the screen being dumped does NOT have to be the one currently displayed. If not specified the 40 column screen will be dumped. If using SCRN, you must use DENS and SIZE.

IND, if used, lets you shift the printout towards the right side of the paper. The value is the number of spaces to print before each line of the screen. The range is 0 to 40. Can be variable or constant. If not specified, there is no shift and the printing starts at the left edge of the paper.

If you use IND, you must use DENS, SCRN, and SIZE also.

Any graphics that you have placed on the screen will be on the printout.

CALL LINK("DUMP","PIO.CR") will dump the 40 column screen in single density.

CALL LINK("DUMP","PIO.CR",1,1) will dump the 40 column screen in double density, and double size.

CALL LINK("DUMP","PIO.CR",0,2,0) will dump the 32 column screen in quad density.

CALL LINK("DUMP","PIO.CR",0,2,1,20) will dump the 40 column screen in quad density, centered on the paper.

LOWC

Converts an ascii string to lower case.

SYNTAX:

CALL LINK("LOWC",OP$)

OP$ is the string of characters to convert to lower case. Must be a variable because it is modified.

If there are any capital letters in OP$, they will be replaced with lower case letters.

LOWCAS

Loads an alternate lower case character set.

SYNTAX:

CALL LINK("LOWCAS")

Overwrites the character definitions in characters 97 thru 122.

REMOVE

Removes selected characters from a string.

SYNTAX:

CALL LINK("REMOVE",OP$,CON$)

OP$ is the string that you want to remove certain character from. Must be a variable.

CON$ is a string of characters that you want removed from OP$. Can be a variable or constant.

If OP$="T*E?S*T?I*N?G", after doing CALL LINK ("REMOVE",OP$,"*?"), OP$ will equal "TESTING".

APPENDIX

APPENDIX A - FILES ON THE DISTRIBUTION DISKS

ON THE MAIN DISK (OR SIDE 1 IF A FLIPPY):

```
  -README1   - docs for use of this disk
  40_SUBS    - merge file to make command use easier
  40_UTIL:XB - 40 column utilities in CALL LOAD format
  40_XB      - 40 column utilities in XB format
  DEMO       - demonstration program
  DOC1       - instruction file used by demo
  DOC10      -      "        "    "   "   "
  DOC11      -      "        "    "   "   "
  DOC12      -      "        "    "   "   "
  DOC13      -      "        "    "   "   "
  DOC14      -      "        "    "   "   "
  DOC2       -      "        "    "   "   "
  DOC3       -      "        "    "   "   "
  DOC4       -      "        "    "   "   "
  DOC5       -      "        "    "   "   "
  DOC6       -      "        "    "   "   "
  DOC7       -      "        "    "   "   "
  DOC8       -      "        "    "   "   "
  DOC9       -      "        "    "   "   "
  ELIZA      - 40 column routines for ELIZA
  ELIZA2     - the main Eliza program
  LDINS      - loading instructions, used by demo
  LOAD       - auto-loader for demo program
  MEMO       - another sample program

  JIFFYDOCS  - docs for jiffy clock routines
  JIFFYDOCS* -  "    "    "     "      "

  JIFFY      - jiffy clock for standard XB
  JIFFY40    - jiffy clock for 40 column screen
```

ON THE DOCUMENT DISK (OR SIDE 2 IF A FLIPPY):

```
  -README2   - docs for use of this disk
  DOCS       - documentation to be printed by DEMO

  HELP1      - docs for newsletter Editor and Formatter
  HELP1*     -  "    "      "         "     "      "
  HELP2      -  "    "      "         "     "      "
  HELP2*     -  "    "      "         "     "      "
  HELP3      -  "    "      "         "     "      "
  HELP3*     -  "    "      "         "     "      "
  HELP4      -  "    "      "         "     "      "
  HELP4*     -  "    "      "         "     "      "

  LOADNEF    - loader for Newsletter Editor and Formatter
  NEF        - the Generic Newsletter Editor and Formatter
               by Bill Gaskill
```

APPENDIX B - DEMO, ELIZA, MEMO

To load the DEMO program just select Extended Basic with the program disk in drive one. An alternate way to run DEMO is to type in at the XB command line: RUN "DSK1.DEMO" . If you haven't already loaded the 40 column routines, type in: RUN "DSK1.LOAD" . DEMO will auto load. DEMO will show how many of the commands work, and let you read instructions for some of the new commands. For a complete description of the commands, please refer to this manual.

ELIZA is my interpretation of the program ELIZA from 'BASIC COMPUTER GAMES'. To run it just type in: RUN "DSK1.ELIZA" . A special version of the 40 column utilities will load, and then the actual ELIZA program will load. The 40 column utilities used by ELIZA include an additional command to manipulate XB's data pointer, and the DUMP routine is missing to save space. To use ELIZA, just follow the prompts.

MEMO shows a simple full screen editor created with the ACCEPT command. The 40 column routines must be loaded before you RUN memo. The arrow keys move the cursor around the screen. ENTER puts you at the beginning of the next line. Press PROCEED to print the screen. The screen is printed from string arrays, not via the DUMP command.

NEF is the Generic Newsletter Editor and Formatter by Bill Gaskill. To run it, type in 'RUN "DSK1.LOADNEF"'. The loader for NEF contains the 40Col routines plus the 40 column jiffy clock routine. After you have NEF running, press 'L' and type in 'HELP1' to read the first docs file. If you choose to print out the help files, NEF will print them in double columns for you.

THE 'MEMO' PROGRAM

To run, with the disk in drive #1: first type in 'RUN "DSK1.40_XB"' and press ENTER. When you get the cursor back, enter 'RUN "DSK1.MEMO"'.

Now that memo is running, notice that you can type on the 40 column screen. You have full cursor movement control. As an example of having two screens in memory at the same time, press FCTN 9 to see a menu in the normal graphics mode. Note: the options on the menu are not implemented, it's just a sample. Press any key to return to the editor screen, right where you left off! All in an XB program yet! The menu screen was built in the beginning of the program, so when you press FCTN 9, it just seems to appear. It is not redrawn, because both screens stay intact unless you change them. Press FCTN 6 if you want to print the screen.

Explanation of the program:

100 - setup an array to hold the screen, not really needed

for this simple demo, I could have read the screen with the GSTRN command to print it.

110 - sets the starting row and column to 1, and the offset to zero so the cursor will appear on column 1 and row 1.

120 - SCINIT sets the window to the whole screen. (not needed here because ACCEPT does not use the window, that is planned for the future version). CLS clears the 40 column screen, even though it isn't displayed yet. TEXT displays the blank 40 column screen and sets the colors to 16 foreground, 5 background.

130 - using standard XB commands, clears the 32 column screen and displays the menu. But you won't see this yet, because the 40 column screen is displayed now.

140 - sets the colors used for the standard graphics screen. (they can by different)

150 - The heart of this program. Accepts one 40 column line. Saves it in the array S$(ROW).

160 - if ENTER was pressed, and ROW<24 adds one to the ROW, and resets OFFSET to the beginning of the line.

170 - if the DOWN ARROW was pressed, does the same as ENTER, but doesn't reset OFFSET, so that the cursor is in the same column.

180 - if the UP ARROW was pressed, and the ROW<1 subtracts one from the ROW.

190 - if BACK was pressed, returns to the NORMal screen, and waits for a keypress. The 40 column screen is not lost, but is not displayed. You now would see the menu we printed earlier. After you press a key, you are returned to the 40 column screen.

200 - if you get here, enter or the up or down arrows weren't pressed. If anything but FCTN 6 was pressed, ignore it and return to the editor.

210 - FCTN 6 was pressed, so print out the contents of S$(). Program ends here.

220 - the last three lines are just a subprogram to wait for a keypress.

APPENDIX C - THE 40_SUBS MERGE FILE

This file is to be merged into your XB program to make using the 40 column commands easier, and take up less memory. If you need to save more memory, give the subprograms shorter names.

In most cases, the syntax of the subprograms is similar to the 40 column utilities syntax. XB subprograms can't have optional parameters like the 40 column commands have though. So you will have to decide if you want to pass all of the parameters to the subprogram, or you may choose to set some in the subprogram itself.

For example, as I have the subprogram VERT written, you have to specify REPS even though if you were using the CALL LINK, REPS would be optional. So instead of CALL LINK("VERT",ROW,COL,CHAR) with the subprogram you would use CALL VERT(ROW,COL,CHAR,1). In ACCEPT, I haven't used the optional parameters KEYVAL or OFFSET. You will have to look at the subprograms and modify them to suit your purposes. In the subprogram DUMP I let the subprogram do all of the parameter setups for CALL LINK("DUMP"....). Nothing is passed to the subprogram.

The subprogram PNTSCN is not one of the 40 column utilities, but shows you how to do a TEXT screen dump of the 40 column screen. Don't confuse this with the graphics screen dump command, which will print out any graphics you may have printed on the screen. The syntax to use PNTSCN is: CALL PNTSCN("device"). You should not use a .CR in the device name.

CURRENT SUBROUTINES IN THE 40_SUBS FILE:

CALL SCINIT(R,C,W,H)	CALL VERT(R,C,CH,RP)
CALL TEXT(F,B)	CALL HORZ(R,C,CH,RP)
CALL NORM	CALL GSTRN(R,C,A$,RP)
CALL CLS	CALL LOWCAS
CALL DISP(R,C,A$)	CALL CAPS(A$)
CALL ACCEPT(R,C,S,C$,R$)	CALL LOWC(A$)
CALL ACBEEP(R,C,S,C$,R$)	CALL REMOVE(OP$,CON$)
CALL PRINT(A$)	CALL DUMP
CALL SCROLL(REPS)	CALL PNTSCN(DEV$)

APPENDIX D - MEMORY USEAGE BY 40 COLUMN ROUTINES

CPU USEAGE:

The utilities load into the lower 8K of the expansion memory. This area is not used by Extended Basic. 2490 bytes are left free for use by other assembly programs. Either way you load the utilities (40_UTIL:XB or 40_XB) you will have the 2490 bytes free.

If you are loading your own assembly with the 40 column routines, the 40_XB file MUST be loaded before your routines. 40_UTIL:XB may be loaded before or after your routines. Either way, after loading all assembly routines you should use a utility such as SYSTEX or ALSAVE to save the combined assembly, for quicker loading.

VDP USEAGE:

After you CALL LINK("SETUP"), VDP RAM is used as follows:

VDP MEMORY	BYTES	USE
>37DD - >37FF	35	not used by 40 col utilities
>3800 - >3BBF	960	40 column screen image table
>3BC0 - >3BD3	20	validation that SETUP is done
>3BD4 - >3C05	50	PAB area for DUMP
>3C06 - >3C55	80	buffer area for DUMP
>3C56 - >3DEE	409	not used by 40 col utilities

The areas marked 'not used' may be freely used by your programs. The areas marked for the DUMP routine may also be used, but if DUMP is used, it will overwrite whatever you have stored there. Of course you may write routines that manipulate the 40 column screen image table.

Whatever you do, DON'T play with the validation area, or you will guarantee problems with disk access at some time!

APPENDIX E - GPLLNK AND DSRLNK

As assembly programmers who link to XB know, TI didn't see fit to give us a GPLLNK and DSRLNK to use from the XB environment. However I needed a DSRLNK to write a screen dump routine. So I have used MG's DSRLNK and GPLLNK from THE SMART PROGRAMMER, July 1986. Since it would be a waste to include these in my code and let other programmers provide their own, I have modified SETUP to install a link to the GPLLNK and DSRLNK in my code. You must do SETUP from XB before the GPLLNK and DSRLNK are available to your assembly routines.

To use these routines in your code just equate GPLLNK and DSRLNK to the following addresses. The addresses won't change no matter where in memory the utilities load at. Future versions of the utilities will also use the same addresses.

```
GPLLNK EQU  >A000
DSRLNK EQU  >A004
```

After that you can use GPLLNK and DSRLNK as you normally would.

```
       BLWP @DSRLNK
       DATA 8

       BLWP @GPLLNK
       DATA >20
```

Or whatever.

If you are going to use the GPLLNK and DSRLNK you must not change the memory area >A000 thru >A007 in high memory. This area of memory does survive the loading of another Extended Basic program, so you only need to do SETUP once.

The DSRLNK returns any error codes in the caller's R0.

APPENDIX F - QUICK REFERENCE

```
CALL LINK("ACCEPT",ROW,COL,SIZE,CK$,RET$[,K[,OFFSET]])
     -accepts input from 40 column screen

CALL LINK("ACBEEP",ROW,COL,SIZE,CK$,RET$[,K[,OFFSET]])
     -same as ACCEPT but with a beep when called

CALL LINK("CLS"[,CHAR])
     -clears the 40 column screen

CALL LINK("DISP",ROW,COL,MSG$)
     -display at row,col on the 40 column screen

CALL LINK("GSTRN",ROW,COL,RET$[,REPS])
     -reads a string of characters from 40 col screen

CALL LINK("HORZ",ROW,COL,CHAR[,REPS])
     -HCHAR to 40 column screen

CALL LINK("NORM")
     -display graphics mode screen

CALL LINK("PRINT",MSG$)
     -prints to window on 40 column screen

CALL LINK("SCINIT",ROW,COL,WIDTH,HEIGTH)
     -sets location and dimensions of print window

CALL LINK("SCROLL"[,DIR[,REPS]])
     -scrolls window on 40 column screen

CALL LINK("SETUP")
     -gets VDP memory ready for 40 column screen

CALL LINK("TEXT",FOR,BACK)
     -displays the 40 column screen, sets colors

CALL LINK("VERT",ROW,COL,CHAR[,REPS])
     -VCHAR to 40 column screen

CALL LINK("CAPS",OP$)
     -changes OP$ to all capital letters

CALL LINK("DUMP",DEV$[,SIZE[,DENS[,SCREEN[,INDENT]]]])
     -does a graphics screen dump of 32 or 40 column screen

CALL LINK("LOWC",OP$)
     -changes OP$ to all lower case letters

CALL LINK("LOWCAS")
     -loads an optional lower case letters

CALL LINK("REMOVE",OP$,CON$)
     -removes the characters in CON$ from OP$
```

APPENDIX G - LEHIGH 99'ER COMPUTER GROUP INFO

If you're looking for a computer group that caters to the TI-99/4A, in Eastern Pennsylvania, please feel free to contact:

THE LEHIGH 99'ER COMPUTER GROUP

ALLENTOWN, PA 18102

We meet on the third Monday of every month at the Sacred Heart Hospital in Allentown.

Our dues are $12.00 per year.

You don't have to be a member to stop in and visit us at our meeting. Anyone is welcome.

Also, for TI and Geneve support, call the First Floor BBS at (215) 760-0527, 24hrs, 3/12/2400 baud. Running on a TI. After 1/1/94 the area code will change to (610).