T40XB

Forty Column Utilities for XB v2.8 G.E.M. on the TI-99/4A Home Computer

by Harry Wilhelm

06/03/20

INTRODUCTION

T40XB is a collection of assembly language subroutines that give the Extended BASIC programmer easy access to the 40 column text mode built into the TMS 9918A VDP. This mode is not normally available in Extended BASIC. No knowledge of assembly language is required to use T40XB. Programs are written completely in Extended BASIC, so they are both easy to write and easy to understand.

T40XB lets a running program select from two independent screens. The default screen is G32, which is the same 32 column graphics mode normally used by Extended BASIC. The usual XB graphics statements are used to access this screen. T40 is the 40 column text screen which provides 24 rows of 40 columns. For the T40 screen there are assembly equivalents that replace PRINT, CLEAR, COLOR, INPUT, CHAR, HCHAR, VCHAR plus routines that scroll the screen and invert text on the screen. You can toggle between the G32 and T40 screens as desired, while preserving the text and graphics on each screen. You can define an area of the screen as a window where text can be input or printed, word wrap can be used when printing, and entire fonts can be loaded from or saved to disk.

EQUIPMENT REQUIRED

T40XB has been tested with XB 2.8 G.E.M. on the Classic99 emulator. It should work using the TI-99/4A console, XB 2.8 G.E.M. cartridge, and a 32K memory expansion. A disk drive system is helpful. It is compatible with the CF7 expansion.

DIFFERENCES FROM EXTENDED BASIC

The T40 text mode has certain limitations compared to the normal G32 graphics mode. Sprites cannot be displayed. You can only use two colors. In the T40 screen characters from 0 to 223 can be defined. Characters 0 to 31 can be used for custom graphics. Characters 32 to 127 are used for the main font. By default, the characters from 127 to 223 are inverse video copies of the main front characters from 32 to 127. If desired, these 96 characters can contain a second font.

When a program starts running in T40XB it defaults to the G32 mode. The program can then select the T40 mode when desired. The G32 screen is unmodified but hidden while the T40 screen is displayed. When you break a program by pressing <Fctn 4> the display will automatically revert to the standard 32 column editing screen. After breaking a program, you can type CON to continue. If you were using the T40 screen, T40XB will display that screen just as it was when the program was interrupted. You can toggle between the G32 screen and the T40 screen within a running program.

The memory available for a program is 24488 bytes which is the normal size for XB. The stack space is reduced from 11840 bytes to 9098 bytes (with the default 3 disk files). The stack is primarily used to contain string data and subprogram names and there should be enough room for most programming needs.

Disk file space is allocated in the usual manner using CALL FILES(n)

USING T40XB

T40XB contains 31 assembly language subroutines.

Except for the commands T40ON and T40OFF, all these subroutines should be called from within a running Extended BASIC program. No error message results when the subroutines are called from the immediate mode, but nothing useful will result. They only have an effect on the T40 screen. You can use the T40 routines in the G32 mode or XB graphics instructions (CALL HCHAR, CALL VCHAR, etc.) in the T40 mode but the results will only appear when you change the screen mode.

COMMANDS

CALL LINK("T40XB")

Turns the T40XB interrupt routine on When you load T40XB this command is done by default in line 10 of the loader. You would use this in the immediate mode to reactivate T40XB if you turned it off with CALL LINK("OFF"), described below. (Immediate mode)

CALL LINK("OFF")

Turns off the interrupt routine for T40XB. This turns off the interrupt routine without having to return to the master title screen. T40XB is still loaded can can be reactivated with CALL LINK("T40XB") (Immediate mode)

THE SUBROUTINES

The subroutines are described below. The first line of each description shows the correct syntax to use when calling the subroutine. Most of the subroutines require additional information after the name of the subroutine. This information is supplied in the form of a parameter list. Be careful to include these parameters in the order described and be sure strings or numbers as required. Sometimes there are optional parameters. These optional parameters are shown enclosed in brackets. The purpose of each of the parameters in the list is fully described. Numbers and strings can be constants, variables, or elements of an array.

CALL LINK("G32")

Selects G32, the normal XB screen. The screen color will be set to cyan. This has no effect if you are already in the G32 mode..

CALL LINK("T40")

Selects the T40 mode which is the 40 column text mode.. This has no effect if you are already in the T40 mode. When a program first calls this, the T40 screen is cleared, a character set with lower case descenders is loaded from ASCII 32 to 127. An inverted copy of this font is loaded to characters from 128 to 223. The default colors are a white foreground on a dark blue background.

CALL LINK("COLOR”,foreground-color,background-color)

This is used to change the text colors in the T40 mode. The color codes are the same as in Extended BASIC

CALL LINK("CHAR”,character-code,pattern-identifier[,...])

This is the T40 equivalent of CALL CHAR in XB. It is used to change the patterns of the characters used in the T40 mode.. *Character-code* specifies the ASCII code of the character you wish to define. It must be a value from 0 to 223. *Pattern-identifier* can be a hexadecimal string of up to 255 bytes. (This is much longer than XB allows.) 16 bytes are used to define each character. If the length of *pattern identifier* is not a multiple of 16, then zeros are added to the final character definition to make it 16 bytes long. Characters from 0 to 31 are available for custom screen graphics. Characters from 32 to 127 are used for the main font. Characters from 128 to 223 are used for the second font which is usually an inverted copy of the main font. Add 96 to the main font ASCII to get the code for the inverted characters. If the ASCII is from 32 to 127, CHAR will define the character, then redefine the inverted character(s). Up to 8 character codes/pattern identifiers can be used in one CALL LINK(“CHAR”)

```
     CALL LINK(“CHAR”,65,”0123456789ABCDEFF”) defines
     character  65 to be “0123456789ABCDEF” and character 66
     to be “F000000000000000”. Characters 161 and 162 have
inverted definitions.

     CALL LINK(“CHAR”,65,”1234”,80,”5678”) defines character
     65 to be “1234000000000000”, 161 to have an inverted  copy of
ASC 65, character 80 to be “5678000000000000”,  and character 176
to be an inverted copy of ASC 80.
```

CALL LINK("INVERT”,row,column,length[,...])

INVERT is used to toggle text to inverse video or back to normal. *Row* and *column* determine the first character you want to invert and *length* is the number of characters to invert. *Row* is a value from 1 to 24. *Column* is a value from 1 to 40. *Length* is a value from 1 to 920. INVERT will stop at the lower right of the screen even if you provide a combination of *row, column* and *length* that would go off the bottom of the screen. A character in the area being inverted with an ASCII of less than 128 will have 96 will be added. A character in the area being inverted with an ASCII greater than 127 will have 96 subtracted. Because characters 128 to 223 are set to inverse video when T40 is called, the hiliting takes place automatically. If you want to restore text back to normal, do a second CALL LINK(“HILITE”,row,column,length). If you are using two different fonts, INVERT can also be used to change text from one font to the other.

CALL LINK(“SCROLL”[,repeats])

SCROLL will scroll the window up one line and fill the bottom line with spaces. Include an optional number from 1 to 24 to scroll that number of times.

CALL LINK(“SCROLI”[,repeats])

Identical to SCROLL except the spaces are in inverse video.

CALL LINK(“CLS”)

Fills the entire T40 screen with spaces. (ASCII 32)

CALL LINK(“CLSI)

Fills the entire T40 screen with inverted spaces. (ASCII 128)

CALL LINK(“CLW”)

Fills the T40 window with spaces. (ASCII 32)

CALL LINK(“CLWI)

Fills the T40 window with inverted spaces. (ASCII 128)

CALL LINK(“HCHAR”,row,col,character-code[repeats,.....])

HCHAR is the T40 equivalent of HCHAR in XB. *Row* is from 1 to 24; *col* is from 1 to 40; *character-code* is from 32 to 223. As in XB you can add an optional number to repeat. You can do up to 4 HCHARs per call by adding additional *row,col,character-code,repeats.* If you want to use this feature you must include the optional repeat so that 4 values are passed per HCHAR.

```
CALL LINK(“HCHAR”,2,2,42,10,4,4,65) will display 10
asterisks (ascii 42) starting at row 2, column 2. It
then displays a single “A” (ascii 65) at row 4, column 4
```

CALL LINK(“VCHAR”,row,col,character-code[repeats,.....])

Identical to HCHAR above except it displays characters vertically like VCHAR in XB.

CALL LINK(“CAT”)

This subprogram is used to catalog a disk.

PRINTING TO AND INPUTTING FROM THE SCREEN

CALL LINK(“WINDOW”,row1,col1,row2,col2) or **CALL LINK(“WINDOW”)**

WINDOW lets you select an area on the screen to use for inputting and printing text. Row1,column1 is the upper left corner of the window. Row2,column2 is the lower right corner of the window. At startup, default is to use the entire screen as the window. If no arguments are passed then the entire screen is selected as if you had done CALL LINK(“WINDOW”,1,1,24,40) When printing to or inputting from a window, the row and column are relative to the upper left corner of the window.

```
CALL LINK(“WINDOW”,4,4,11,11)::CALL LINK(“PRINT”,2,3,”Hello”)
!selects an 8x8 window starting at row=4,col=4 and prints A$
at row=2,col=3 of the window; or 5,6 on the screen
```

CALL LINK(“WRAP”)

This is used to turn on word wrap when printing. (default on T40 startup)

CALL LINK(“NOWRAP”)

This is used to turn off word wrap when printing.

The four subroutines described below, CALL LINK(“PRINT”), “PRINTI”), (“INPUT”) and (“INPUTI”) all print or input within the boundaries of the window.

CALL LINK(“PRINT”,row,col,string or number[,length,.....])

PRINT prints a string or number on the T40 screen.. If using the full screen , *row* is a number from 1 to 25 and *col* is a number from 1 to 40. If you are using a window, *row* must be a number from 1 to the height of window+1 and *col* is a number from 1 to the width of window. The optional length must be a value from 0 to 255. Printing starts at row and col; remember that these are relative to the upper left boundary of the window.

If no length is specified, or if the length is zero, then the entire string or number is printed, with scrolling as necessary. Then the remainder of the last line is cleared.

You can print one row below the bottom of the window. In that case the window scrolls once and printing commences normally on the bottom line of the window.

Including the optional length forces PRINT to use the specified number of characters. If the string is longer than that, only the specified number of characters are printed. If the string is shorter than the optional length then spaces are used for the additional characters. Bear in mind that word wrap removes and inserts spaces and those changes become part of the character count. Once the specified number of characters have been printed, the remainder of the line will *not* be cleared unless a scroll has happened.

You can do up to 4 prints per call, but you must include the optional length

```
CALL LINK(“HCHAR”,1,1,42,1920)::CALL
LINK(“PRINT”,2,2,”Hello”,10,4,4,”Hello World”,
10,6,1,3.14159265)

!The HCHAR fills the screen with asterisks;

CALL LINK(“PRINT”) does the following:
At r2,c2 it prints “Hello” followed by 5 spaces;
At r4,c4 it prints “Hello Worl” (the first 10
characters of  “Hello World”);
At r6,c1 it prints Pi with the rest of the line filled
with spaces because there was no optional length.
```

CALL LINK(“PRINTI”,row,col,string or number[,length,.....])

Identical to CALL LINK(“PRINT”) except it prints using the second font which is usually inverse video.

CALL LINK(“INPUT”,row,col,string-or-numeric-variable[,length,prompt-string])

INPUT is used to input a string or numeric variable. If using the full screen, *Row* is a number from 1 to 25; c*ol* is a number from 1 to 40. If using a window, row is a number from 1 to the height of window+1; col is a number from 1 to width of window. Remember that *row* and *column* are relative to the upper left corner of the window.

The optional length can be a value from -255 to 255, but the number of characters must fit within the window. If the specified length will not fit then a BAD VALUE error message will be issued. If the length is positive then the specified number of characters will be licleared and the cursor will flash at the first position. If the length is negative then no characters are cleared and the cursor flashes at the first position. This lets you leave text on the screen as a suggested prompt. If no length is provided, or if the length is 0, the line will be cleared to the right until it reaches the right hand column of the window.

After the length, you can provide an optional prompt string. The cursor flashes on the first character of the prompt. You should use a string even if you are inputting a number. For example, use “3.14159” for PI. The optional prompt must be proceeded by the optional length.

The row can be 1 lower than the bottom of the window. In that case, one or more scrolls will happen until enough characters are available to accommodate the string.

For some applications you will not know in advance how many characters will be entered. In this case, use a row that is one below the bottom of the window, and do not specify a length, or specify a length of zero. In this case only, the window will scroll once, the bottom line will be cleared, and you can start entering characters. If you reach the right hand column, the window will scroll and you get a fresh line to continue with. (this is similar to the editing mode in the TI) You cannot scroll past the top of the window. If your window is a size that lets you enter more than 255 characters, then only the first 255 characters will be returned for the string.

```
CALL LINK(“PRINT”,2,2,”Hello World”)::CALL
LINK(“INPUT”,2,2,A$,-15) will input the string variable
A$ at r2,c2 using “Hello World” as a prompt.

CALL LINK(“INPUT”,4,4,X,20,”3.14159265”) will input the
variable X at r4,c4 using the value of Pi as a prompt.
```

CALL LINK(“INPUTI”,row,col,string-or-numeric-variable[,length,prompt-string])

Identical to CALL LINK(“INPUT”) except it inputs text using the second font, which is usually inverse video.

FONTS

T40XB now has the ability to load and save entire fonts. If you are using XB28GEM, there are 60 fonts available in the cartridge rom. If you do not have XB28GEM, the same 60 fonts can be loaded from disk. They are included in the folder 60FONTS; the names of the files are FONT1 to FONT60. The file for a font is 768 bytes long and it contains the patterns for characters 32 to 127, with the cursor saved as ascii 127. You can load a font into three different locations: the normal G32 font, the standard T40 font (ascii 32 to 127), and the inverted T40 font (ascii 128 to 223). Usually the inverted font is an inverted copy of the main T40 font, but if you load a second font to this location you can simultaneously have two different fonts on the screen.

To use fonts built into XB28 G.E.M.:

CALL LINK("FONTA",N)

This is used by XB28GEM to load one of the 60 stored fonts from the cartridge rom. The font number N must be a number from 1 to 60. This will load a font used by T40 to characters 32 to 127. It also loads an inverted copy of the same font to characters 128 to 223. (XB28GEM only)

CALL LINK("FONTI",N)

This is used by XB28GEM to load a font from one of the 60 stored fonts from the cartridge rom. The font number N must be a number from 1 to 60. This will load a font used by T40 *only* to the patterns used by the inverted font, characters 128 to 223. (XB28GEM only)

To use disk based fonts available to all Extended BASIC versions:

CALL LINK("LFONT","DSKn,filename")

This loads a font used by the G32 mode from disk.

CALL LINK("SFONT","DSKn,filename")

This saves a font used by the G32 mode to disk. Characters patterns from 32 to 127 are saved in the font, with the cursor saved as ASC 127. You would use this if you want to save a custom font created in the G32 graphics mode.

CALL LINK("LFONTA","DSKn,filename")

This loads a font used by the T40 mode from disk The font is loaded to characters 32 to 127. This also loads an inverted copy of the same font to characters 128 to 223

CALL LINK("SFONTA","DSKn,filename")

This saves a font used by the T40 move to disk. Characters from 32 to 127 are saved in the font, with the cursor saved as ASC 127. You would use this if you want to save a custom font created in the T40 text mode.

CALL LINK("LFONTI","DSKn,filename")

This loads a font used by the T40 mode from disk. The font is only loaded to the inverted font characters (ascii 128 to 223)

16K VDP RAM MEMORY MAP WITH T40XB

0 – 767 >0000->02FF	Screen Image Table	VDP address is given by (Row-1)*32+Column-1
768 – 879 >0300->036F	Sprite Attribute Table	Room for 28 sprites – each sprite needs 4 bytes vertical position-1,horizontal position, char #+96,color-1(+128 for early clock)
1008 – 1919 >03F0->077F	Pattern Descriptor Table for G32	8 bytes per character – char 30 starts at 1008
1920 – 2031 >0780->07EF	Sprite Motion Table	4 bytes per sprite. Vertical velocity, horizontal velocity; sys use; sys use
2048 – 2079 >0800->081F	Color Table for G32	1 byte per character set – char set 0 = 2063 (foreground-1)*16+background-1
2304 – 2559 >0900 - >09FF	Character definitions for ASC 0 to 31	
2560 – 3327 >0A00 - >0CFF	Pattern descriptor table for characters 32 to 127 (Normal font)	
3328 – 4095 >0D00 - >0FFF	Pattern descriptor table for characters 128 to 223 (Inverse or second font)	
4096 – 5055 >1000 - >13BF	Screen Image Table for T40 mode	960 bytes long
5056 - 14295 >13C0 - >37D7	Value stack	Used by XB for strings, etc. CALL LINK(T40XB,n) will reserve N bytes starting at 5056
14296 – 16383 >37D8 - >3FFF	Disk Buffering Area for CALL FILES(3)	