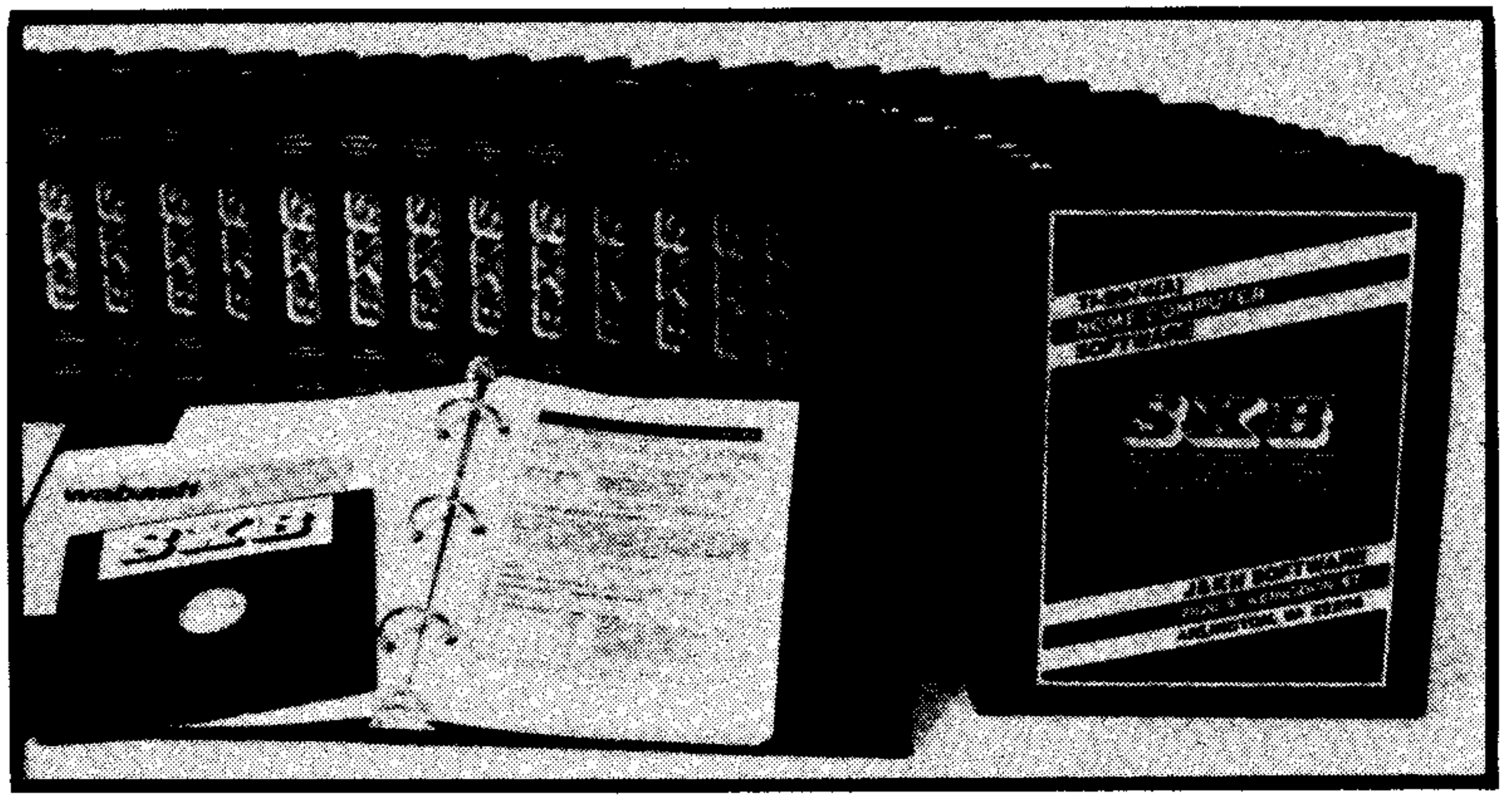
# SUPER EXTENDED BASIC™



The February, 1985, issue of MICROpendium (pages 20-23) contained a review of SUPER EXTENDED BASIC™ by Chris Bobbitt. Following the review was a response from the author, James Hollender. The two articles have been merged below for the sake of clarity. Mr. Hollender's response is shown in *italicized* print. The articles are reprinted with permission of MICROpendium.

## REPORT CARD

**Performance: ..... A**
**Ease of Use: ..... A**
**Documentation: ..... B**
**Value: ..... A**

**Final Grade: ..... A**

**List Price: $99.95 (diskette)**

**Manufacturer: J&KH Software, 2820 S. Abingdon St., Arlington, VA 22206, (703) 820-4131**

**Requirements: Extended BASIC, 32K memory expansion, disk drive and controller**

I bet everyone who has the memory expansion card or peripheral remembers the day when he or she first plugged it in.

Recall your surprise when you discovered that the "32K" that you bought gave you only 24K through Extended BASIC. Oh, of course, you have the 12K of "stack" memory, but that's only what's left of the original 16K that came in the console.

Where is the missing 8K? Well, if you have the Editor/Assembler manual, or a knowledgeable friend, you may well have discovered that you got everything that you paid for even though the "missing" 8K appears to be rather useless, separated as it is from the other 24K. (The 24K segment is called "high memory" and the 8K segment is called "low memory.")

This "hidden" or unused 8K has a lot of assembly language programmers pretty excited. Where else should you put assembly language subroutines for Extended BASIC without taking up program memory? Whether TI did this by design or not, this has turned into a real blessing. On practically every other computer on the market assembly language routines take up memory that would otherwise hold part of a program. With our machine,

the assembly language programmer gets 8K of otherwise empty space with his name on it, so to speak. James Hollender, of J&KH Software, must have realized this potential before he began work on his Super Extended Basic package.

**Performance:** Super Extended Basic is a collection of over 100 assembly language subroutines that can be called from Extended BASIC like a command or a program statement. These subroutines are "programming aids" in that they are to be used by the Extended BASIC programmer to write better programs. Included are routines for graphics, data base handling, string and string array handling, and mathematics. A lot of these routines could be duplicated in Extended BASIC, but you couldn't get them to work nearly as fast, or in so little space with such ease of use. Also, you would have to actually write them, something J&KH Software has already done for you.

All of these subroutines are crammed into that little 8K segment called low memory. Several of these subroutines — mostly the ones that involve lots of data, such as the array handling and data base routines — require the use of one or more 256-byte blocks of high memory. Unless you are writing a really large program, you should never even miss the memory.

The routines take about 30 seconds to load from disk. The program includes a loader program, which I assume you have to run before you load your program, or actually merge into your own program. Each of these subroutines is accessed, like other assembly language subroutines, with the CALL LINK command in the following format:

```
CALL LINK("Subroutine-Name"[,Var1,Var2,...])
```

*You are correct that it takes approximately 30 seconds to load SXB™ (as we like to call it) — but once the subroutines are loaded, you can remove the SXB disk and begin loading and running you own programs. Under normal conditions SXB will remain loaded and ready to use until the machine is turned off or a CALL LINK("QUIT") is invoked. You can even do CALL FILES(1) followed by NEW and SXB will still be there!*

The subroutine name can be up to six letters long. Each of the variables contains data which the subroutine is to act upon. These variables are collectively called the "argument list." The variables can be arrays, so even though the argument list can't have more then 16 variables in it, with arrays (both string and numeric) you have effectively as many variables as can be stored in the computer.

Collectively, all the routines are divided into six categories: Data Base, String Array, String, Integer mathematics, Garphics or VDP (Video Display Processor), and Miscellaneous. Eleven routines are devoted exclusively to data base handling. I believe it is possible to write an entire database management program comparable to the Personal Record Keeping cartridge, out of just these routines. Included are routines that let you set up the data base so it accepts only data of a certain length, or of a certain type, and moving, changing and sorting the data in the database. Practically all the work has already been done for the programmer. He just has to figure out the order in which to put the routines.

There are ten routines devoted exclusively to handling string arrays. A number of these should have been commands in Extended BASIC, like the routine which tells the programmer how many elements of an array contain data, or the routine which copies every element of one array to another array. There are routines for encrypting the data in an array so no one can read it, reversing the encryption, deleting every item in a string array at once, and finding the particular string item with the longest length in an array.

The next routines should really be counted along with the string array routines but the manual makes a distinction, so I will describe them separately. These are the eight string routines. A number of these routines also should have been included by TI as Extended BASIC commands, among them the routines which fixes a string to a particular length by either adding spaces or deleting characters, and the routine which swaps the values of two strings. There are also routines which delete the specified number of characters off the end of a string,

convert a binary string to a hexadecimal string and vice versa, and for compacting and uncompacting a string to save memory.

*The reason for the distinction between STRING and STRING ARRAY subroutines is because only STRING ARRAY subroutines act on entire arrays with a single LINK.*

The Integer routines, to me, seem the least useful. Other than the fact that they allow you to cram four numbers into the space usually reserved for one (as long as they are between -32,768 and 32,767) they have little use in most programs. Unless a program is a spreadsheet (with no decimals), or something that doesn't require much in the way of computing power, I don't have any notion as to how useful these would be. However, the great majority of routines are integer mathematics routines, about 70 altogether. There are routines that allow you to perform the basic operations (add, subtract, multiply and divide) on these integers in any way to obtain any conceivable answer.

*With respect to the INTEGER subroutines it should be pointed out that their sole purpose is to free up PROGRAM SPACE (part of the 24K). Generally they are not any faster. If you have a lot of small numbers to keep track of, this will free up to 75 percent of your numeric array space — which amounts to six bytes per number. Our new master catalog program, the Multi-Disk Informer™ (a sizeable upgrade from Super Cataloger™), can only keep track of as much information as it does with the use of these INTEGER subroutines.*

*True, the INTEGER subroutines cannot deal with decimals — but they can handle numbers in the range -32,768 to +32,767, substantial enough for most counting functions.*

*False, there are not 70 INTEGER subroutines — only 44. Most of these are taken up with eight each for adding, subtracting, multiplying and dividing. Please note that although there is a substantial number, they occupy only a very small portion of the total storage space for SXB. The reason for the large number is to provide as much flexibility as possible for integration with numbers in Extended BASIC. This way you don't have to exclusively use SXB INTEGERS but can have the best of both worlds.*

**I believe the graphics (or VDP routines, as they are called) are about the most useful in the whole package. These 28 or so routines are collectively worth the $99.95 price tag. All of these routines operate within a user-defined "window" on the screen. The window can be of any size, up to 24 rows by 32 columns. Any of the graphics routines executed take place within the window. This allows you to isolate the graphics portion of a program from the text. The window is defined with the subroutine called VMWNDW, by indicating the coordinates of the upper left and lower right corners of the window within the argument list.**

Some of these routines let the programmer clear the window instantly without disturbing anything around the window, read in all the characters in a window into string variables (sort of like a CALL GCHAR statement), and allow the user to type anything, anyplace in the window using a full-screen editor (the cursor can be moved up, down, left and right). Other routines let the programmer place or get strings in the window in any one of eight directions, redefine up to 31 characters instantly, or, in reverse, get the character patterns of up to 31 characters at once, change the colors of all the sprites and all the character sets at once to the desired color for each, redefine all the lower-case letters to typewriter-style, and redefine some unused characters as lines for creating boxes and objects on the screen. All in all, my only complaint is with the routine that lets you type in anything you want anyplace in the window — the keys are not auto-repeating.

*Concerning the VMTYPE command which allows freeform typing within the active window, yes, we do not have the auto-repeat feature. It's not that we forgot about it — it just took up too much space. So to compensate we included some even better features, like editing (delete/insert), erase and move to: home (upper left), left, center, right and left on next line (like a carriage return).*

The last category of routines, Miscellaneous, is well named since the routines do not fit into any other category. The first routine I find absolutely useless, but interesting. This routine, named BANNER, turns a string into a banner where each regular-sized character is represented by a character eight asterisks high and eight asterisks wide. The routine called KEY1 is very useful, since it is basically a CALL KEY statement that responds only to the keys specified, and then returns a different integer (appropriate for use in an ON GOTO statement) for each key pressed. There are two routines that allow the programmer to transfer data between two programs without using a data file, and a routine called USRSUB, which allows the programmer to include one of his own assembly language routines. The last routine in the package allows the program to quit back to the master title screen from a program. (There is a CALL LOAD address for this one, by the way.)

*The BANNER subroutine does have some uses. If you ever want to print banners on your output listings and didn't have this subroutine, it would take up a considerable amount of program space. This subroutine helps make the TI-99/4A's output look like it came from a mainframe computer. Also, we used BANNER in the Multi-Disk Informer. It makes looking for a particular version of the report a lot easier. Note also that combining BANNER with VMWNDW and VMREAD in a loop can provide an easy method of producing a giant-sized screen dump!*

**All the routines seem to work well, and most of them are very useful. With this package the user can create programs that will rival regular assembly language programs.** Unfortunately, we will not see a lot of software using this because J&KH Software, like most of the companies that create assembly language routine packages, requires a royalty for each program that is sold that uses the routines from SXB.

*No, J&KH Software does not demand a royalty for each program that is sold which requires SXB. Actually, just the opposite is true. We will pay a royalty to anyone writing programs which use the SXB subroutines as long as they have the programs published by us (assuming the programs meet our high standards of quality). Anyone is allowed to publish programs using SXB. What we do not allow is inclusion of the SXB subroutines with the published program. SXB must be purchased separately just as TI Extended BASIC must be purchased separately from programs which are written in TI Extended BASIC. We anticipate publishing some SXB programs within the next few months. At that time we will be making available a special version of SXB which will only include the subroutines and instructions on how to load them. This will provide an avenue for the person who wants to use programs written by others that use the SXB subroutines, but does not want to spend the extra money for all the documentation and other authoring aids. This special version of SXB, to be called SXB RUN-TIME OPTION™, will be priced at $54.95. Persons wishing to upgrade to the full version at a later date will be able to do so for an additional $49.95 (only with proof of purchase, of course).*

**Ease of Use: All in all, the package is very easy to use. It is logically designed, with no exceptions. All the CALL LINK commands follow a particular pattern. Routines in the first five categories of this package have, as the first two letters of their name, a prefix to indicate the category that they belong in. This is useful when writing and debugging programs. The package functions well, and because the subroutines take the place of ones the programmer would have had to write in Extended BASIC, they don't have to be debugged (saving a lot of time).**

The manual is also very easy to use. Each category of routines is covered in its own section, and each section is a different color from the others. Finding a particular routine is very simple. Included in the back of the manual is an extensive index for looking up by page and section number each routine, as well as definitions of important terms. There is no table of contents, but because each section is color-coded, a table of contents is not needed.

The manual is tough reading, though. More on that later.

*The reason for the absence of a table of contents is that the user is allowed to rearrange the color-coded sections of the manual to his or her own liking and the index still remains valid. This is mentioned in the introduction to the manual.*

**Documentation: The manual comes in an attractive, professional-looking, black three-ring binder. Each page is about the size of a half page of regular paper. The binder is not cheap or shoddy, and the pages snap in and out well. It is about the best looking documentation I have every seen accompanying a third party product for the TI-99/4A.**

Using the manual is a different story. It is difficult to read, up there with the Editor/Assembler manual. Even the examples are none too clear. It seems as if every word in the manual is polysyllabic. The explanations aren't very well written, and may be confusing to both beginning and advanced programmers alike. The best way to discover how a routine works is to judge it by its name, which usually indicates what it does, and by trying it in a short program. Then read the short summary in the manual.

*The manual itself was designed more as a reference manual than a tutorial. It is definitely something that you would not want to read from beginning to end with the exception of the introduction. When you need to use a particular subroutine, that is when you want to read its particular page or pages. We have also included on the SXB disk several programs as examples. The one that has been found to be the most useful by our customers is the SXB-TEST which also verifies that SXB is working correctly on your own particular computer (a few people have had problems with bad memory — portions of which are being flexed for the first time). This program includes most of the examples used in the manual so you can see just how the CALL LINK statements interface with the TI Extended BASIC coding.*

**Value: This is one of the most valuable programming tools for the Extended BASIC programmer. Granted, the programmer gets few of the enhanced graphics routines found in other packages, but in their place he gets routines that do more "meat and potatoes" chores, such as string handling and sorting. Like Extended BASIC itself, the SXB package is not specific to one task. It is a COMPLETE enhancement package in that it makes every programming task simpler, not just those involving graphics or data handling alone. It would be great if many of these routines could have been included in the Extended BASIC cartridge itself. That enough is indicative of how valuable I think this package is.**

*One thing not mentioned is the fact that SXB is the only software for the TI-99/4A which has an option for extending the product. I am referring to the SXBrief™ Newsletter, a monthly publication solely about SXB. It consists of four pages to add to the three-ring binder each month. In fact, the first six issues (January — June, 1984) are included with the full SXB package. A subscription to SXBrief is available to registered purchasers of the full SXB package at a cost of $10 per year (plus $5 for overseas delivery). Subscriptions always start with the July, 1984, issue so you won't miss a single issue no matter when your subscription starts. J&KH Software is committed to publishing the newsletter through at least June of 1986. One of the things included in almost every issue is a new USRSUB which can be invoked in your own programs with the CALL LOAD instruction. The USRSUBs published during 1984 included VMFILL, SADEL, DBFND2, GRAPH, SAINS, VMSCUP, VMSCDN, SMCNTR, VMRECT and DBMARK.*

---

MICROpendium P.O. Box 1343 Round Rock Texas 78680

— 12 Issue Subscription Fee —

| | |
|---|---|
| Domestic 3rd Class* | $15.00 |
| Domestic 1st Class* | $18.50 |
| Canadian Delivery | U.S. $18.50 |
| Foreign Delivery (surface) | U.S. $21.50 |
| Foreign Delivery (air mail) | U.S. $28.50 |

*Texas residents add 5.125% sales tax.

---

# Super Extended Basic

AUTHORIZED DEALER