



HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER USERS GROUP
 USERS GROUP HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER
 GROUP HOOSIER USERS HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER
 HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER
 USERS GROUP HOOSIER HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER
 GROUP HOOSIER USERS HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER
 HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER
 USERS GROUP HOOSIER HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER
 GROUP HOOSIER USERS HOOSIER USERS GROUP HOOSIER USERS GROUP HOOSIER
 HOOSIER USERS

THE HUGGERS
HOOSIER USERS GROUP
 People Helping People

Jan 1989

THE HUGGERS NEWLETTER

Volume 7, Number 1

HOOSIER USER GROUP
 BULLETIN BOARD (317) 782-9942
 by William M. Lucid

HUG has TEXTLINK by the Ottawa user group. I purchased this bbs for \$40.00 at the Chicago Faire. This bbs has an all assembly code, xmodem upload and xmodem download, plus a large message base that tracks the messages the user has read.

When calling (317)-782-9942 the user of the bbs needs to use a terminal emulator program such as FAST-TERM or TELCO, the bbs will not respond to the Terminal emulator II.

Commands available are displayed on three menu screens which are the [Main Menu], [General Mail], and [File transfer]. The package supports control s, control q, control p. Control s suspends scrolling. Control q resumes scrolling. Control p aborts last command. Pressing [ENTER] will redisplay the current MENU. An asterisk will LOGOFF user. There is a prompt before LOGOFF asking user to confirm, this gives user a change to re-enter the menu.

Leaving a message on the bbs is done by selecting [E]xpedit from the [General Mail] menu. There is a submenu for the message editor. Exiting from the EDITOR is done placing a period on a blank line.

The system configuration four disk drives (two of which are off-line), a Myarc ramdisk, MBP Clock card providing time in 24 hour format along with date stamping, and a Editor/Assembly cartridge.

```

+-----+
| Main Menu |
+-----+
[B]ulletins           [C]hat with Sysop
[E]xchanges/Sales    [F]ile transfer
[H]elp on line       [I]nfomart
[I]ast 10 users      [M]essage base
[O]ther BBS systems [U]ser list
[Y]our parameters   [?]This menu
[*]System utilities [*]Logoff
Choice [BCEFHILMOUY?*]
  
```

16:13:59 Command ->

```

+-----+
| General Mail |
+-----+
[E]xpedit mail       [K]ill mail
[M]ain menu          [P]review mail
[R]ead new mail      [S]elective read
[*]Logoff
I have 028 active messages
Numbered from 0001 to 0030
Last message you read 0029
Choice [EKMPRS*?] Cmd ->
  
```

```

+-----+
| File transfer |
+-----+
XMODEM only.
[0] DOWNLOAD        [5] DOWNLOAD
[1] DOWNLOAD        [6] DOWNLOAD
[2] DOWNLOAD        [7] DOWNLOAD
[3] DOWNLOAD        [8] DOWNLOAD
[4] DOWNLOAD        [9] DOWNLOAD

[D] Download        [U] Upload
[M] Main Menu       [*] Logoff
Choice [0123456789DMU?*]->
  
```

"A Poor Man's Loader"

Written by:

Paul E. Scheideattle

From what I've seen over the years, most people I know have had a tendency to rely on extendedbasic loaders written by others. Unfortunately with little understanding of what goes on inside the program... So when they have a special problem, such as mixing xbasic programs with alot of files and/or programs with there own loaders, and programs that are made-up of multiple programs... with 1 running the other. They're usually forced to keep them on separate diskettes. Even though I do use the special type loaders that read the disk directory and then display the contents on the screen for your selection (I even wrote one of these myself 'SUPERLOADER'). I still find myself constantly writing a LOAD program for disks where the only thing I want on the screen is the Name of the program and a way to run it with a single key stroke! So I thought I'd sit down and write myself a generalized loader that can be modified easily and shared with others with the same kind of problems that I have! Listed here and hopefully explained as best as possible is a simple loader that can be modified for most disks holding xbasic and even option #5 E/A type programs. Note that the option #5 programs themselves must have xbasic loaders!

One of the first things to do when combining these files on to one diskette is checking to make sure that there will be no duplication of filenames! You may have 5 or 6 programs that you would like to put on the same disk but... they all have a load program. No problem! Just change the name of the LOAD program to something else! I would suggest using a name that fits in with that set of files and yet you will still recognize it as the load program, for example changing the name of say TI Artist's loader to "RUNARTIST" and so on. Next you would install the loader listed below and make a few minor changes that I'll describe. At that point you've combined the programs and saved yourself disk space at the same time!

Now to the program! In lines 100 and 110 we start with a simple remark statement. What the program is and who wrote it! Normally I would have also at this point have included version numbers, date, and possibly running information such as REQ'D: Xbasic or REQ'D: Xbasic/32k and so on!

```
100 ! Simple Loader Call Keypress Demonstration
110 ! By Paul E. Scheideattle
```

Line 120 clears the screen and sets the screen color to light green. Usually pleasant to the eyes... but any color here would have done as long as it wasn't black.

```
120 CALL CLEAR :: CALL SCREEN(4)
```

This segment of the program is the main menu. Line 130 is just to help divide up the program into readable segments. While line 140 is a work-horse line! Here we do what is called a 'RESTORE', this causes all data in the program data statements to be restored so that it can be read from the 'TOP' (meaning the 1st DATA statement in the program). Data in this case is read from 'TOP TO BOTTOM' - 'LEFT TO RIGHT'. Next the 'READ B' Reads the number of Menu items to place on the screen from line 160. Now with this information we can use a loop 'FOR A=1 TO B' to read the program names ('A\$') and display them on the screen. Here in the DISPLAY AT routine has been added a simple way of giving you/and displaying a keystroke char 'CHR\$(A+64)' the char you'd get if you add 1+64 would be 'A', and the CHR\$ command converts a ASC char number to a string. The _ is used here to add strings together! So when we read the first 'A\$', which would be "EXIT PROGRAM" and add the 'A' and the string " - " to it we get "A - EXIT PROGRAM" as the string to be displayed on the screen at 'A' (row) height and in column #4. From here we keep going until the value of 'A' is greater than the value of 'B', and then leave the loop! Also inside the loop we setup a string 'B\$' to store the keystrokes that can be used to make selections and is used in line 200.

```
130 ! MENU
140 RESTORE :: READ B :: FOR A=1 TO B :: READ A$ ::
    DISPLAY AT T(A,4):CHR$(A +64) - "A$ :: B$=B$
    CHR$(A+64):: NEXT A
150 ! SAVE # PROGRAMS (+) for EXIT) IN THIS DATA
    STATEMENT!
160 DATA 5
170 ! PLACE NAMES OF PROGRAMS HERE!
180 DATA EXIT PROGRAM,BATTLE SHIP,LEMONDROP TREE,MANHOLE,
    STAR GALLERY
```

Now we proceed on to line 190 to display a question on the bottom of the screen! Pretty straight forward. Line 200 is special CALL statement which is defined as a SUBPROGRAM at the end of the program. I'll explain the values used in the statement, but will discuss the subprogram later! The values are 24 = the row and 23 = the column that I want display a character on, B\$ = the chars that are acceptable, 66 = ASC value of the "B" the first runnable program on the disk (a default value), and RK = the return ASC key that is pressed! Line 220 again a simple statement 'ON RK GOTO', based on the value of 'RK' it will goto to the line specified by the number. For example lets say RK=3, it would go to the third line number listed. Here is where we take action.... and 'Run or Exit' the program in lines 220 thru 280. Line 230 shows a "QUIT MESSAGE" and ask if you are sure? Again we use the CALL KEYPRESS routine. Lines 250 thru 280 are used to place the 'RUN' command with the disk number and program name!

```
190 DISPLAY AT(24,1):"    YOUR CHOICE:"
200 CALL KEYPRESS(24,23,B$,66,RK)
210 ! ADD LINE # OF NEXT PROGRAM TO BE RUN IN LINE 220
```

```

      SUCH AS 281,282,283 ETC!
220 ON RK=64 GOTO 230,250,260,270,280
230 DISPLAY AT(24,1):" QUIT ARE YOU SURE: (Y/N)" :: CALL
      KEYPRESS(24,29,"Yn",89,RK) :: IF RK=89 OR RK=121
      THEN CALL CLEAR :: END ELSE 190
240 ! PLACE RUN STATEMENTS HERE CONSECUTIVELY!
250 RUN "BSKI.BATTLE"
260 RUN "BSKI.LEMONDROP"
270 RUN "BSKI.MANHOLE"
280 RUN "BSKI.STARGAL"
290 END

```

Now finally we get to the SUB PROGRAM 'KEYPRESS': To start with line 300. All the letters in between the() have a purpose and lets look at them! Bye-the-way they are called 'parameters'. The parameters become necessary when you want to pass information back and forth between the main program and a Subprogram! So here I wanted a single call key statement that could not only check the keyboard for a response but display a character or a default character and check for an acceptable responses! To get a char on the screen and make it flash I had to rotate between a space char and a default one. So it was necessary to pass this information to the routine as well as pass the key value back to the program. So... ROW = screen row, COL = screen column, CHK\$ = acceptable string info, DC = default character, and RK = return key value.

```
300 SUB KEYPRESS(ROW,COL,CHK$,DC,RK)
```

Line 310. Here we check the keyboard to see if a key has been touched, display the 'SPACE' char. and if no key is pressed then display the default char. and repeat until a key is pressed.

```
310 CALL KEY(3,RK,S) :: CALL HCHAR(ROW,COL,32):: IF S=0
      THEN CALL HCHAR(ROW,COL,DC):: GOTO 310
```

Line 320. using the 'POS' function here we can check key response against the CHK\$ (with the >ENTER< key value added so that if you just press enter... it will respond as if you pressed a correct key). This is done by checking to see if one string (in this case the key pressed 'RK' converted to a char string) is found in the CHK\$ starting at the first location in the string. If there is no match then 'I' is set to '0' and you are returned to the call key routine. If 'I' is any other char it is checked to see if it was the enter key (13) if so it then changes RK (both key pressed/and return key) value to equal the DC value... So that on the following line 330 the correct character may be displayed provided it is a char between 32 and 127! Finally line 340 relinquishes control and returns you to the main program.

```
320 I=POS(CHK$(CHR$(13),CHR$(RK)),1) :: IF I=0 THEN 310
      ELSE IF RK=13 THEN RK=DC
330 IF RK>31 AND RK<128 THEN CALL HCHAR(ROW,COL,RK)
340 SUBEND
```

EXTENDED BASIC - STILL A GOOD CHOICE!

The case for the BASIC Language!
By ART BYERS

Part One: Introduction.

There are new guys in the 99/4A neighborhood. Among them are such stars as FORTRAN, FORTH, PILOT and SMALL C. They have lots of adherents who talk about "Like Basic" (FORTRAN), "Freedom and Exceptionally flexible" (FORTH), "Simplicity" (PILOT), and "Speed and structure" ('c'). They are Compiled languages which means they certainly run much faster than old friend XBasic. SOOOoooo? Why bother with Extended Basic at all? Why not go with the New? The Better? The Faster?

One of the great things about our beloved 99/4A is that even with its limited memory, it CAN support FORTH and C and PILOT. I consider any of the computer languages that will accomplish what is needed to be fine! For me, however, Extended Basic still remains the EASIEST and BEST, most especially when coupled with Assembly Language subroutines that speed up often used important areas.

Let me try to lead you through a discussion of the pros and cons of Extended Basic without "putting down", in the slightest ANY other language for the 99/4A (including Pascal -However Pascal requires a special PEB card and those are hard to find and some early versions have bugs).

Extended Basic has many advantages from a programmer's viewpoint, not the least of which is that it is an interpreted language with a plethora of error debugging routines built in. One of the real swift pains in the neck of a compiled language is that if it is compiled containing errors or bugs, these are extremely difficult to find. This does not mean they cannot be found or that good programmers cannot produce error free compiled code. It is just that debugging, adding to, subtracting from, changing code, etc. is much easier with XB. It is a shame that TI chose to make XB a "double" interpreted language by

writing it in GPL, TI's "secret" proprietary language, also interpreted, (which to the best of my knowledge TI has NEVER released and should they have chosen to take legal action, they could make trouble for those who have violated their rights by selling GPL programs, books explaining GPL, etc. and etc.). It would have been better if the interpreter had been written in Assembly a la MYARC's XB. The added speed of MYARC's XB is a big improvement over TI's XBasic. However, The whole subject of execution speed will be covered in more detail in part 3 of this series. It deserves separate discussion because this area is what is most often raised in any and all debates on the merits of TI XB.

One of the biggest advantages of XB is its EASE OF USE AND UNDERSTANDING. BASIC itself was written just for that purpose. BASIC is supplied with such popular computers as Apple, Atari, Commodore, and IBM. This ease of use was most important in bringing better understanding of computers and use of computer languages to large numbers of Americans. For no other reason, the Basic language continues to survive.

As far as the 99/4A goes, another advantage is that the language itself resides outside the RAM areas. It is in ROM and GROM. The cover of the XB manual states that the module contains "32k bytes of preprogrammed memory". Most of the RAM is free. Additionally, XB accesses, again with simplicity, clarity and ease, the built in ROM routines such as Device Service - printers, cassette, disk drives -, screen access and display, setting up of buffers, graphics and sprites, mathematics, etc. Many of the "new" languages save RAM memory by also accessing these same ROM routines, running at the same speed for all!

Now lets talk about available memory. Because support for Forth and 'c', for examples, must be loaded into the main 32k memory area, they do not have as much memory available as some programmers feel is absolutely

necessary. This problem has been solved by using virtual memory - that is disk storage of Forth screens (blocks) or C support routines. XB support resides in console ROM and the module itself, the full 24k upper RAM is available for programs and the 8k low memory for Assembly support routines, and most of VDF RAM for string storage etc. For example, I recently purchased a Disassembler which was written in Forth. The author plainly stated that because of the memory used by Forth itself plus the program, it was not feasible to disassemble programs from RAM. It did its disassembly right off the disk!. Since Basic resides in ROM, a disassembler written for E/A or MM modules can be written in plain old BASIC, and can disassemble programs that use the 24k upper and 8k lower memory, because it resides in VDF RAM, and not overwrite the program.

Some last points! let us look at what we have to work with. We have a machine designed as a HOME computer. For almost every purpose or use at home, memory and speed available through XB are more than sufficient. We are not tracking satellites, doing high order lengthy math, searching a database the size of the national Social Security register. We have a hundred or so names on our phone list. We do not require massive spread sheets. For our normal practical purposes XB and the 99/4A can suit our needs. In fact I may be accused of HERESY, but I did almost everything with only the XB module and cassette - NO memory expansion or disk!!!

What is more, when I need a special program written to fill a personal need, I write it, debug it and am using it in a matter of a few minutes to at most an hour. This is possible because the most frequently used XB GOSUB routines and CALL SUBs are saved on disk as MERGE files ready to be placed into a program, easily and quickly. Many programmers overlook this useful feature of XB.

The following articles in the series offer concrete evidence to backup the

ideas expounded above. They are NOT a tutorial in Basic programming. Rather, they will place a point of view before you as food for thought. It, hopefully, will lead to a return to some good Basic programming.

XXXXXXXXXXEoFXXXXXXXXXX
 XXXXXXXXXXXXXXXXXXXXXXX

OFFICERS

PRESIDENT....JOHN POWELL 786-3270
 VICE-PRES.....CARL CLARK 1-398-6226
 LIBRARIAN..BRYANT PEDIGO 255-7381

FOR SALE
 MBP CLOCK A/D CARD

\$45.00

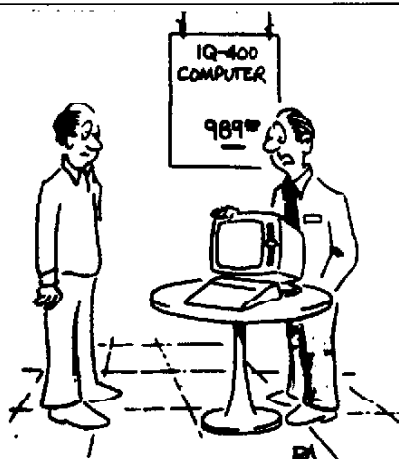
Call Dan Eicher Sundays After
 5:00 P.M.
 317-241-9942

 * SOUTH SIDERS MEETING *
 * SECOND-----THURSDAY *
 * AFTER THE MEETING *
 * MONTHLY *
 * CALL 786-3270 *
 * FOR LOCATION *
 * *****

MEETING DATES FOR THE COMING YEAR

January 8
 February 12
 March 19
 April 9
 May 21
 June 11

St Ann School will be closing at the end of the school year. Watch the NEWS LETTER for further meeting dates



INSURANCE SALES

"Let's get one thing straight--it's smarter than you and it knows it!"

DISCLAIMER

This newsletter is brought to you through the efforts of the officers and members of the HOOSIER USERS GROUP. Every member is encouraged to submit articles.

If you have an article you would like to share with the other members mail it to:

John Powell
 327 W. Southern Ave.
 Indianapolis, IN 46225

Opinions expressed are those of the author and not necessarily those of the HOOSIER USERS GROUP.

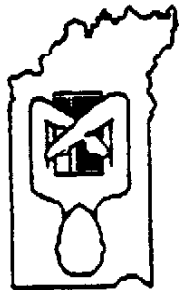
 * PASTE UP PARTY *
 * HELP GET THE NEWSLETTER *
 * OUT *
 * LAST FRIDAY OF JANUARY *
 * AT BOB STAHLHUT'S HOUSE *
 * CALL 856-4962 *
 * *****

MONTHLY MEETING LOCATION

ST. ANN'S SCHOOL
 2839 S. McCLURE
 INDIANAPOLIS, IN
 MEETINGS OPEN AT
 2:00 PM
 JANUARY 8 1989

TIME DATED
JANUARY 8 1989
MATERIAL

HOOSIER USERS GROUP
P.O. Box 2222
Indianapolis, IN 46206-2222
Forwarding and Address
Correction Requested



APPLICATION FOR MEMBERSHIP

Below you will find an application for membership to the Hoosier Users Group. Active membership entitles you to the Newsletter, up and download on the HUGbbs, attendance and voting rights at regular club meetings, access to the HUGger Library of Programs, special club activities and special guest speakers for one year. Subscribing members will receive the **NEWSLETTER** only.

Make check or money order payable to **Hoosier Users Group**. Send completed application to:

HOOSIER USERS GROUP
P.O. Box 2222
Indianapolis, IN 46206-2222

(Cut on dotted line)

Check One:

Active Member

New: \$20 _____
Renewal: 15 _____

Subscribing Member

New: \$10 _____
Renewal: 7.50 _____

Amount Enclosed: \$ _____

_____ D _____
S _____ O _____

Name: _____ Today's Date: _____

Address: _____ Apt. # _____

City: _____ State: _____ Zip: _____

Phone: (____) _____-_____

Interests/Comments: _____

Dear Former Hugger;

These are exciting times for those who own a 99/4(a). In fact ~~some~~ many new products have come out that we would like for you to come to a meeting and see what is available.

Here is a sample of what is now currently available:

PRESS - This is an all new wordprocessor that has the look and feel of a professional product for the IBM called Wordstar. This program includes text size limited only by your available disk memory, a 100,000 word spell checker. This program also automatically configures itself to what ever hardware you may have available. O' yes menu driven with windows, of course.

Telco - This is the final word on terminal emulators for the 4(a). It provides you with 4 types of transfers including - x-modem, y-modem or Kermit. This program is user configurable and menu driven with pop-up windows.

PC-Transferr-

If you use an IBM PC at work this is the program for you. This program will allow you to transfer text files from a PC formatted disk to a TI formatted disk and vice/verse. What ran out of room on your MS-DOS disk? This program will even format an IBM floppy for you!!

DSKu - This program combines a disk manager and disk sector editor all in one package. Some of the features are-

Graphics Applications-

Here are some of the many graphics programs that are now available that makes full use of the bit-map mode on you 99/4(a).

TI-ARTIST -

MC-FLEX - This program will allow you to display drawings created with a MacIntosh. Other utilities programs will also let you display programs in RLE and GPL format.

The Printers Apprentice-

This program will allow you to create signs and banners like your friends with PRINTSHOP do.

Not only has software that pushes your TI to limit just came out, but hardware to further expande the TI's limit.

Here is an example-

IBM Keyboard

This package is sold by RAVE. All you have to do is plug the interface card into your console and you have a full professional keyboard!

80-COLUMNES-

At long last you can see multiplan and TI-writers in 80-columns. Not only does this card give you 80 columns, but you also get much better graphics! This card gives you graphics capabilities that rival those found on the Atari St or the Commodore AMIGA.

1 MEGABYTE battery backed up ramdisk-

Ever get tired of doing the disk shuffle? Ever envy your friends that have a harddisk? Well with this card you can have up to 1 Meg. worth of programs available to you upon power up! Not only that but the time it takes to load files is less than if you were using a harddrive!

HardDrive controller card-

available this card is for you. This controller will control up to 4 floppies and 1 harddrive also for the fastest backup possible you are given the ability to backup your harddrive to streamer tape. O' yes time and date stamping is included with this package!

P-Gram-

This is a very new card that fits right into your P-box. This card will allow you to load, save and modify cartridges. No more annoying lock-ups do to poor cartridge port connections! This device opens up all kinds of modifications to programs that were once locked in silicon! Also a real time clock feature is an option on this card!

Gramulator-

This device offers the same abilities as the above mentioned card. But in addition you can modify the operating system of the TI to suit your needs and preferences.

There are many more hardware upgrades and modifications to mention. Much much more software to cover. We would like for you to come to the Feb 12 meeting, we will have a LOCAL dealer there with most of the products on display and ready for purchase.

Hope to see you then!