YESTERDAY'S NEWS

VOLUME 4 NUMBER 8 Established 2016 AUGUST 2019

30 Years Ago...

Historical Information taken from Bill Gaskills TIMELINE

AUGUST 1989:

MICROpendium announces a subscription rate increase to take effect Oct. 15th.

Glenn BernaseK launches Gee*Bee Basics, a custom programming firm for TI-99/4A omputer users.

Memex memory expansion unit for the Myarc Geneve is released by Bud Mills Services.

Asgard Software releases Legends II and MDOS Conversion Notes.

McCann Software releases The Printer's Apprentice for the Myarc Geneve 9640.

Giant Art Posters by Paul Coleman is released by Comprodine.

Designer Labels v2.4 is released by Texaments.

GenProg program development software for the Geneve is released by Paul Charlton ,through Jeff Guide's DisK Only Software company.

Ron Prewitt of Tacoma, Washington releases Columntext v4.2.

Production begins on the Zenoboard protyping board created by Eric Zeno.

Hardmaster, a hard disK editor for the Myarc HFDC controller hard disK, debuts from Asgard Software. Program author is Australian Colin Christensen.

Dr. Guy Steffen-Romano, the original custodian of the IUG's user-written software library, dies on August 15th. He was 57 years old.

Barry Boone releases EXEC utility for the Myarc Geneve.

J. Peter Hoddie (James Peter Hoddie), formerly of the Boston Computer Society, owner of Genial Computerware, and assembly language programming wizard on a TI-99 /4A, hires on at Apple Computer.

INSIDE INFORMATION

VOID

HOME COMPUTER COMPENDIUM
Vol 1, No 1
February 1984

REPORT CARD	
PERFORMANCE	A
EASE OF USE	A
DOCUMENTATION	B
VALUE	B
FINAL GRADE	B+

Review by John Kolean

I found Void to be an imaginative and highly challenging game, one that I would recommend to anyone who thinKs he's a joystick jocKey. This game has excellent graphics and action and is a real test of hand-eye coordination and problem-solving abilities. With 20 screens, it is also highly addictive. I stayed up several nights trying to advance just one more screen, and every one that I managed to reach was different from those that preceded it.

PERFORMANCE: Void is an arcade-type game that requires quicK reflexes and quicK thinKing to win. You actually can win this game by finishing the twentieth screen. The farthest I got while reviewing Void is the sixteenth screen.

I found every screen to be well done, not only from a graphics standpoint but also in terms of how imaginative each is, despite operating under the same basic requirements. Sound effects are well done, too.

Starting with nine lives (none are added for the rest of the game), you must cause a man-liKe figure to jump or run over obstacles while avoiding everything from spider-liKe critters and moving walls to a low-flying moon that you may, if your timing is excellent, jump over. Beginning at

See "VOID", page 2

TI CLASSROOM

TIPS FROM THE TIGERCUB

NUMBER 03

By Jim Peterson

I'm just a one-man User's Group pretending to be a business - not a business pretending to be a User's Group!

Here's a lifesaver that was passed on to me verbally so I don't Know who to credit for discovering it... It's 2 A.M., You just got you got the last bug out of your new program, you sleepily put a new cassette in the recorder, type OLD CS1, hit ENTER and ...ooooh!, you meant to type SAVE CS1!! But all is not lost - just type Shift E, hit ENTER, get an IO error message, and start over.

```
100 CALL CLEAR
110 RANDOMIZE
120 DATA TIGERCUB SOFTWARE,P
RESENTS,THE,CHAMELEON,SCREEN
BORDER,AND,WIPE, by Jim Pete
rson," ","        TOUCH ANY
KEY"
130 REM - M$ COMPOSED OF PAI
RS OF HEX CODES WHICH ARE MI
RROR IMAGES OF EACH OTHER
140 M$="1800665AC342DB667E18
8100995AC3A5E78142BD24DB6600
8142992400 7E5AC3A53C241800FF
DB5AFF7EFF0099188100660018"
150 RESTORE 120
160 REM - PRINTS TEXT FOR DE
MONSTRATION
170 FOR P=1 TO 10
180 READ A$
190 REM - TAB TO CENTER TEXT
200 PRINT TAB(15-LEN(A$)/2);
A$:" "
210 NEXT P
220 GOSUB 300
230 REM - PAUSE, WAIT FOR AN
Y KEY
240 CALL KEY(0,K,ST)
250 IF ST=0 THEN 240
260 GOSUB 440
270 GOTO 150
280 REM - SUBROUTINE TO PICK
PATTERN AND COLORS, DRAW BOR
DER
290 REM - RANDOMLY SELECT AN
Y STRING OF 8 SYMMETRICAL PA
IRS FOR HEX CODE, DEFINE CHA
RACTER
300 CALL CHAR(128,SEG$(M$,IN
T(43*RND+1)*2-1,16))
310 REM - RANDOMLY SELECT FO
REGROUND AND BACKGROUND COLO
RS BETWEEN 3 AND 16
320 X=INT(14*RND+3)
330 Y=INT(14*RND+3)
340 REM - IF BACKGROUND IS S
AME COLOR AS FOREGROUND, PIC
K ANOTHER
350 IF Y=X THEN 330
360 REM - DRAW THE BORDER
370 CALL COLOR(13,X,Y)
380 CALL HCHAR(1,2,128,31)
390 CALL HCHAR(24,2,128,31)
400 CALL VCHAR(1,2,128,24)
410 CALL VCHAR(1,31,128,24)
420 RETURN
430 REM - SUBROUTINE FOR ALT
ERNATING WIPES. VALUE OF T A
LTERNATES BETWEEN 1 AND 2
440 T=T+1-ABS(T=2)*2
450 ON T GOTO 470,510
460 REM - LEFT TO RIGHT WIPE
470 CALL VCHAR(1,3,128,768)
480 CALL CLEAR
490 GOTO 530
500 REM - TOP TO BOTTOM WIPE
510 CALL HCHAR(1,1,128,768)
520 CALL CLEAR
530 RETURN
```

Of course, that routine can be varied in many ways. For example, try changing line 300 CALL CHAR(128,"FF"&SEG$(M$,INT(43*RND+1)*2-1,12)&"FF"). The basic algorithm of line 140 (which can be any combination of the mirrow-image pairs) and lines 300-350, has endless uses for putting colorful graphics on your screen. For instance -

```
100 REM - TIGERCUB RANDOM BA
RS, by Jim Peterson
110 CALL CLEAR
120 RANDOMIZE
130 M$="0018243C425A667E8199
00A5BDC3DBE7FFFFE7DBC3BDA500
817E665A423C2418 0018243C425A
667E8199A5BDC3DBE7FFFFE7DB"
140 FOR CH=40 TO 142 STEP 8
150 CALL CHAR(CH,SEG$(M$,INT
(43*RND+1)*2-1,16))
160 X=INT(14*RND+3)
170 Y=INT(14*RND+3)
180 IF Y=X THEN 170
190 CALL COLOR(CH/8-3,X,Y)
200 CALL HCHAR(23*RND+1,31*R
ND+1,CH,10*RND+1)
210 CALL VCHAR(23*RND+1,31*R
ND+1,CH,10*RND+1)
220 Z=INT(10*RND)
230 IF Z<>0 THEN 250
240 CALL CLEAR
250 IF Z <>1 THEN 270
260 CALL SCREEN(INT(15*RND+2
))
270 NEXT CH
280 GOTO 140
```

Hey! I just thought of something else to try that Chameleon Screen Border and Wipe. Try changing -
250 IF ST=0 THEN 220

The previous routine generated random redefined characters which have 2-way symmetry, left and right. The following routine generates characters which have 4-way symmetry and are even more interesting, although the routine is a bit slower.

```
100 CALL CLEAR
110 RANDOMIZE
120 DIM A$(16)
130 DATA 00,18,24,3C,42,5A,6
6,7E,81,99,A5,BD,C3,DB,E7,FF
140 FOR J=1 TO 16
150 READ A$(J)
160 NEXT J
170 FOR CH=40 TO 152 STEP 8
180 FOR L=1 TO 4
190 X=INT(16*RND+1)
200 B$=B$&A$(X)
210 C$=A$(X)&C$
220 NEXT L
230 CALL CHAR(CH,B$&C$)
240 B$=NUL$
250 C$=NUL$
260 NEXT CH
270 FOR S=2 TO 16
280 Y=INT(15*RND+2)
290 Z=INT(15*RND+2)
300 IF Z=Y THEN 290
310 CALL COLOR(S,Y,Z)
320 NEXT S
```

That's the routine. Now, to try it out...

```
330 T=T+1
340 IF T>1 THEN 170
350 CH=40
360 TX=0
370 FOR X=1 TO 12
380 CALL HCHAR(X,1+X,CH,29-X
-TX)
390 CALL HCHAR(25-X,1+X,CH,2
9-X-TX)
400 CALL VCHAR(X,1+X,CH,25-X
-TX)
410 CALL VCHAR(X,31-X,CH,25-
X-TX)
420 CH=CH+8
430 TX=TX+1
440 NEXT X
450 GOTO 170
```

For a different effect, try changing...

```
180 FOR L=1 TO 3
190 X=INT(8*RND+1)
230 CALL CHAR(CH,"00"&B$&C$&
"00")
```

.....Experiment!

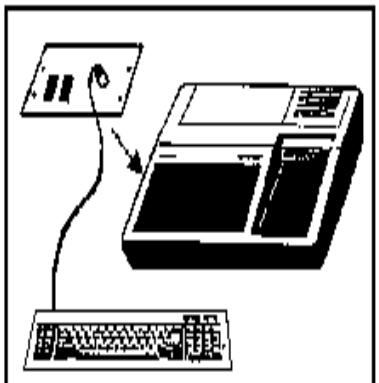
A tip for beginning programmers: Don't use character sets 15 and 16 (ASCII codes 144-159) unless you really need to. And if you use multiple colons :: as print separators, put a space between them : : : Then, when you get to Extended Basic, your program

will run without modification in Extended Basic, and usually faster and better.

OUT OF MEMORY...so that's all for now. You won't hurt my feelings if you mention Tigercub Software to your friends. I have some bargain programs they might like. Just tell them I'd like a dollar for my catalog to cover bankruptcy court fees.

Will there be a tips from the Tigercub #4? It's up to you!!

HAPPY HACKIN'
Jim Peterson



RAVE KEYBOARD INTERFACE CARD

YN

"VOID" continues...

one end of the screen, you must propel your man to the opposite end either to obtain a Key by which the next screen can be reached or to slip through an exit that allows you to descend to the next level. Each level has its own colorful screen.

The difficulty of each screen depends on a number of factors which are impossible to describe briefly. Suffice it to say that the uniqueness of each screen is such that each is a challenge unto itself.

Oh, yes, if you jump too far, your man will fall off the screen. If the man is overtaken by one of the critters or runs into certain stationary obstacles, he will also fall off the screen.

The man will run left or right depending on the direction you push the joystick. Pressing the fire button and using, the joystick simultaneously permits the man to jump. You must combine jumping with running to outwit some of the faster moving monsters you encounter.

Keyboard input is easy. The 1 and 2 Keys control direction while the 0 Key is used to make the man jump. This is preferable to the use of the unwieldy arrow Keys as found in many games. I found more success using the Keyboard than I did using joysticks.

There is no scoring in this game per se. You measure your progress by the number of screens you manage to cover before losing your ninth life. The number of lives you have left is displayed every time a life is lost.

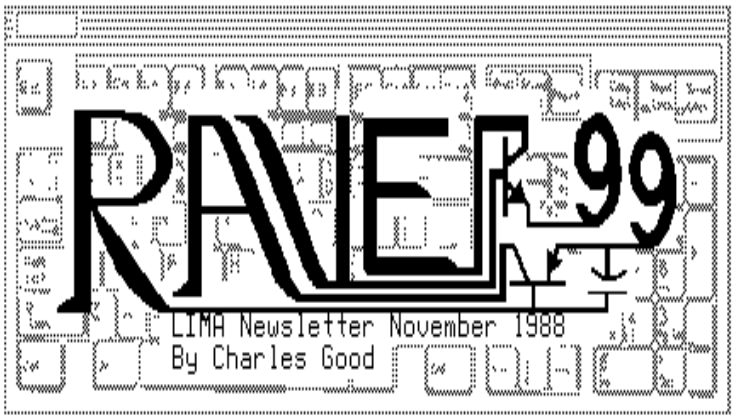
EASE OF USE: It seems that using a joystick in any fast-moving game programmed in TI Extended BASIC is a mixed blessing. I found the TI joysticks to be the least useful of those that I tried. At certain points, the program did not seem to react to the joystick command fast enough to avoid being overtaken by an approaching critter.

The fact that figuring out some of the screens is not a piece of cake simply adds to the challenge of the game.

DOCUMENTATION: The four-page manual that comes with Void actually devotes only one page to a description of the game and how to play it. I would like to have seen more detail in describing the various levels, though not a whole lot about how to play them. That would be like giving away the plot of a mystery to someone who has yet to read it.

VALUE: This game is fun to play and, until you manage to reach the twentieth level, a challenge. The graphics are superb, the screens are imaginative and the action is as fast as I, for one, can handle. It seems to be a bit on the high-priced end of the Extended BASIC scale, though the only thing that keeps the value from being rated A is the fact that purchasers must include $1.50 for postage and handling.

YN



In an article I wrote last year about expanding 99/4A systems cheaply, I stated, "Because of cost, I can't recommend the fancy (RAVE) Keyboard..." Well, guess what? I bought one anyway! If you already have two double sided drives, a good printer, lots of Horizon RamdisK capacity, and are still looking for additional ways to expand your 99/4A system then the RAVE Keyboard deserves serious consideration. In terms of cost/benefit the question of whether the additional features of the RAVE Keyboard justify the price of $199.95 (Why not be honest, RAVE, and make the cost an even $200?) only you can decide.

INSTALLATION: You have to remove the existing Keyboard from your console and install a circuit board in its place. Very detailed directions and illustrations for this procedure are in the RAVE docs. The process does not require any soldering, and took me about 25 minutes.

Basically all you do is unscrew the console covering, unplug the internal components including the original Keyboard, plug the circuit board into the motherboard in place of the Keyboard, and screw everything back together again. You plug the Keyboard into a connector on the circuit board. The Keyboard attaches to the console with a very sturdy coiled cord that can extend about 5 feet. It can easily be unplugged if necessary.

The general appearance of the console after the RAVE circuit board is attached is rather ugly. The circuit board is recessed about 1 inch below the top of the console and has lots of chips and a few wires exposed to view. You can, if you wish, spend $12 and purchase from RAVE an optional plastic cover that mounts flush with the top of the console. This seems rather expensive to me for just a piece of plastic. I have read in the newsletters of some users who have made their own home made covers. I choose not to use a cover. In my system the console is out of sight, so the ugly doesn't show. Also, I suspect that without a cover cooling is improved. Air can get in to the power supply from the side now in addition to the normal "in the bottom and out the top" convection cooling air. I realize that this means dust can get into the console, but I don't think dust by itself is much of a problem to electronics.

Although the RAVE Keyboard replaces the console Keyboard, you still need access to the console and can't get away with hiding the console in some difficult to get at out of the way place. You have to get at the module port, and you have to use the console's on/off switch for correct system power up and power down. For example, my GRAM KRACKER, sometimes loses part of its memory if I turn off the PE box without first turning off the console, which means I can't get away with using a switched plug box to turn on and off my entire system all at once with just one on/off switch. This GK memory loss can occur even after I first move the GK's NORMAL/GK OFF switch to the GK OFF position, which I always do. Use of the RAVE Keyboard will significantly increase the footprint (required flat surface area) of your system, and potential purchasers should keep this in mind before purchase.

RAVE Keyboard owners can purchase an optional wiring harness that allows the Keyboard's HELP button to act as a reset and load interrupt switch. Installation of this option requires some soldering. It is not necessary to solder or desolder chips, so the work probably isn't very delicate. Since I already have a reset switch in my system, I did not choose this RAVE Keyboard option.

THE KEYBOARD:
The current version of the RAVE Keyboard has 105 Keys and is quite different from the original 101 Key RAVE Keyboard that was reviewed in the December 86 issue of Micropendium. Almost all operational details of the new Keyboard are different from those described in the Micropendium review, so prospective purchasers will get a much better description of the current product by reading this review rather than referring back to the review published in Micropendium. The current model 99/105 Keyboard is probably better than the original, and the cost is higher.

What can you do on the 99/4A with 105 Keys? Except for the load-interrupt/reset option mentioned above, and the rather unimportant ENHANCE Key (described below), there is nothing you can do with RAVE's 105 Keys that can't be done with the 99/4A's 48 Keys. The RAVE Keyboard does make things easier to do, in some cases much easier. Whether this extra ease is worth $200 only the reader can decide.

Keyboard feel is softer than that of the Keys usually found on black and silver consoles. The feel resembles that of the Keys on the newer gray consoles or the exact replacement 99/4A Keyboards now available at Radio Shack (See a review of these exact replacement Keyboards in the October 88 BB&P.) There is a numeric Keypad on the right side which includes a decimal, all digits, *, -, and + but does not include an = Key or a separate RETURN Key. The following Keys are enlarged on the RAVE Keyboard for easy touch typing: ESCAPE, CONTROL, SHIFT, BACK SPACE (left arrow), ENHANCE, down arrow, RETURN (same as ENTER on the original Keyboard), DELETE, and ZERO (on the numeric Keypad).

There are 24 numbered function Keys that do with one Key press the same thing as FCTN/(top Key row) and CTRL/(top Key row) on the original Keyboard. This still leaves extra function Keys for more stuff such as F21 used for BEGINNING OF LINE (same as CTRL/V on original Keyboard) in TI-Writer. Some of these function Keys have additional labels that seem to describe word processing functions. However, these additional labels on the numbered function Keys do not correspond to any 99/4A software that I know of, and I find these function Key labels potentially confusing. For example the F4 Key (same as FCTN/4 on the original Keyboard) is labeled "PRINT". In TI-Writer this Key rolls the display down one screen. In Multiplan this Key moves the cursor back one character. Neither of these actions is a "PRINT". Users should ignore all the function Key labels.

There are also specific labeled Keys that perform the named function in both TI-Writer and Multiplan, and sometimes in BASIC. Unlike the weird labels on the numbered function Keys, when you press these dedicated labeled Keys the action you get is exactly what the labeled name suggests. The dedicated Keys often duplicate some of the numbered function Keys. This means that of the 105 Keys on the Keyboard, more than one Key will often do exactly the same thing. Labeled Keys include SCRL/BREAK (scrolls right in TI-Writer as does F5 ... BreaKs a program in BASIC as does F4), TAB (same as F7 in TI-Writer ... same as F12 or CTRL/2 in Multiplan), HOME,

BACK SPACE, INS (insert, same as F2), DELETE (same as F1), HELP (same as F7), and ESCAPE (command/escape in TI-Writer, same as F9). I really like these plainly labeled dedicated keys.

The four separate cursor movement keys are a real blessing any time full screen cursor movement is allowed, such as in T.I. Writer.

The ALPHA LOCK does not affect joystick operation in either position. In the locked (down) position the quote is automatically selected when the "/' key is pressed. This is very useful when typing in or writing BASIC programs. One aspect of the ALPHA LOCK key I don't like is that when the ALPHA LOCK key is locked down to select all upper case letters and you press SHIFT, you get a small case letter. I find this confusing. I am used to the old keyboard and to typewriters where SHIFT gives you a capital letter irrespective of the position of the ALPHA LOCK. There are no little lights on the ALPHA LOCK, or on any other special key, to tell you that the special key is activated. It is easy to forget that ALPHA LOCK is activated (down).

CONTROL and FCTN keys are provided and can be used in exactly the same way as the original keyboard, but this is seldom needed because of the numbered function keys. The FCTN key is a small key labeled "ALT" (alternate) on the keyboard, rather than "FCTN", and its use is almost never needed. Regular keys, which sometimes need to be SHIFTed, are provided for quote, question mark, back slash, underline, etc. I like this.

FAST TERM users sometimes have to press three keys at the same time on the original keyboard (FCTN/SHIFT/P, T, or X). On the RAVE keyboard, these are reduced to two key presses (CONTROL/F2, F3, or F4).

The current keyboard has two modes of operation, not the rather confusing four modes described in the 1985 Micropendium review of the older model RAVE keyboard. The latest TENEX catalog shows a picture of the new keyboard, but states "four distinct modes of operation." I suspect TENEX bases this statement on the now outdated Micropendium review. The two modes are "MULTIPLAN/CLONE" mode 1 (SHIFT LOCK key up), and "TI-WRITER" mode 2 (SHIFT LOCK key down). The quoted names are from the docs. I have no idea what "CLONE" means. I prefer to call mode 1 "MULTIPLAN/EVERYTHING ELSE" because this is the mode the docs say should be used with any software that has a prompt strip. This includes either BASIC.

The TI 99/4A console can recognize two key codes that cannot be created with the original keyboard. The RAVE keyboard can generate these "missing keys" with its ENHANCE key. One code is ENHANCE and the other is ENHANCE/SHIFT. Only ENHANCE is recognized in BASIC, and can be detected by a -1 in the "return variable" of CALL KEY. The docs state that, "you may be confident that these keys are truly unused key codes that have never been used in a program up to now." So what good are the ENHANCE key codes if only those with a RAVE keyboard can use them? Not much. The only use I can think of for these key codes is security. You could, for example, require the use of an ENHANCE key code in your personal checkbook program so that only users of a RAVE keyboard can read and alter your check records. Jim Peterson (Tigercub Software) has several methods to hide code in an XBASIC program so that the code is not obvious when LISTing the program.

COMPATIBILITY:
The Rave 99/105 keyboard is compatible with all software I have tried EXCEPT Gram Kracker Extended Basic (also known as GK UTILITY I). This "adds features to regular extended basic" software allows you to move the cursor up and down rows when editing a program line or in an INPUT statement from within a program with FCTN/SHIFT/E or X. You can also move instantly to the beginning or end of a program or INPUT line with FCTN/SHIFT/S or E with GK extended basic. These keystrokes don't do this with the RAVE keyboard. I have discovered that when using the RAVE keyboard with GK extended basic CONTROL/F1 will move the cursor to the end of a program or INPUT line, and CONTROL/F4 moves the cursor down one row. I have not discovered the secret of moving the cursor up one row, or instantly to the beginning of the line. While not many T.I. users use GK extended basic, there are many who have purchased the SUPER EXTENDED BASIC (version 120) module from TRITON or TEX COMP. I understand that this module is an expanded version of GK extended basic. This probably means that SUPER EXTENDED BASIC module users will have similar problems with the RAVE keyboard.

FINAL COMMENTS:
I recommend to the manufacturer three changes to the 99/105 keyboard. First I would get rid of the meaningless labels on the numbered function keys. This requires a little paint on the existing keys, or different plastic key tops. Second, the incompatibility problem described above should be corrected. Finally, I would like to have a keyboard buffer, a small amount of RAM that remembers the previous 10 or so key codes. This would allow you to avoid the occasional dropped letter that occurs at word wrap in the TI-Writer/FUNNELWEB editor. Since the RAVE keyboard already includes a separate circuit board, it seems to me that it wouldn't be too difficult to design such a keyboard buffer for the circuit board.

I have gotten quite used to the RAVE keyboard. It indeed is easier to use and much nicer than the original keyboard, and I am glad to have it. From strictly a cost/benefit basis, the $200 cost of the RAVE probably doesn't justify the features gained. However, to many users such as myself, the 99/4A is a hobby. Such users sometimes crave the very best for their computer systems irrespective of cost.YN

MYARC EXTENDED BASIC II

By Stephen Shaw - October 1986

This new version of Extended Basic is a far more than a TI-compatible : it is largely compatible with TI ExBas but has a number of enhancements which make it a worthy successor, a definite must for anyone who programs in Basic - and it could even win converts from other languages.

One major drawback: none of it is in GPL. In other words, it is not possible to fit even the TI-compatible bits into a module. With the extensions... therefore you MUST have at least a 128K ram card to run it. At present it will run with the Myarc and Foundation cards only, and a new EPROM for the cards is required - included in the price for a Myarc card and at very low cost for a Foundation card (Foundation have now ceased trading by the way).

So here is what you need in order to use Myarc ExBas: at least 128K ram, preferably 512K, and at least one disk drive, with disk operating system.

TI BASIC: Myarc Extended Basic allows you to run 99% of programs written in TI BASIC. You may need to use CALL FILES(1) to load them from disk - but a clever bit of sidestepping allows the Myarc XB to load a longer TIB file than TI's XB can.

TI EXTENDED BASIC: One major problem was found with running commercial programs: some commercial producers not only supply programs in PROTECTED format, but the program also checks the CPU RAM flag to see if protection has been removed, and if it has, it locks out the console (there are a range of CALL PEEKS that can do this). Bad news: in an effort to increase the protection of PROTECTION, Myarc have moved the flag - so these programs take a look, fail to find the marker, and crash!

A minor but annoying bug resides in the tail rem (!): if your XB program has a tail rem following a FOR...TO command, you will be thrown a SYNTAX ERROR. The cure is either to remove the tail rem, or insert a separator:

```
FOR X=Y TO Z ! HOHO ...=syntax error
FOR X=Y TO Z    .......=FINE
FOR X=Y TO Z :: ! HOHO.=also FINE
```

Possibly associated with this, the prescan check of for-next loops does not function - and this can make debugging very interesting! Remember to check that every FOR has its NEXT yourself if you have problems! For existing TI XB programs this change to pre-scan makes no difference.

And of course, although it is untidy, I have no doubt that some XB authors will be able to make good use of this prescan check NOT being there, making their programs really obscure with several NEXT's for each FOR, dotted around sub-routines!

ACCEPT AT and DISPLAY AT also differ from TI XB, in an undocumented manner which could actually be helpful!
For instance:
ACCEPT AT(11,12)SIZE(32):A$
does what?
In TI XB it blanks from column 11 to the end of line 12, and accepts A$ from column 11 to column 28. In Myarc XB however, it blanks from column 11, line 12, to column 11 of line 13, and the ACCEPT cursor appears on LINE 13!!!

This can cause problems! Fortunately, you are unlikely to come across this problem in many TI XB programs!

DISPLAY AT(11,12)SIZE(32):A$ makes the same deletion, spreading over two lines, but at least the display starts in the right place! Again you are unlikely to have problems with this.

Interesting... ACCEPT AT(24,12)SIZE(32):A$ will place the cursor on ROW 1!!

OK ... now to the useful bit. Lets be really odd and try:
ACCEPT AT(12,2)SIZE(255):A$
NOW... Beginning at row 12, column 2, count 255 screen characters... including the edge characters. THEN the cursor appears! BUT those 255 screen characters may not be blanks! They seem to be taken from the input buffer but I could be wrong. The result is not useful!

Now...ACCEPT AT(12,2)SIZE(-255):A$. Nothing is blanked, but the input field for A$ measures 255 chars from (12,2), including any positions outside current margins (e.g. edge characters). You can therefore type in quite a lot of text, and have it accepts with one key push into one string. This could be very useful. You can of course use HCHAR first to blank out that section of screen. As this is undocumented it may not stay through to any subsequent versions, but I hope it does, as it answers a question asked often in Micropendium & other TI-related mags and newsletters...

BLACK MARK: I may be the only person to OPEN files with the APPEND declaration, but Myarc XB fails to support it. It will open the file without an error message, but then refuses to write to it! Bye Bye APPEND.

IMAGE / PRINT USING: TI XB places asterisks (***) if you have more characters or numbers than you can fit into an IMAGE. Myarc XB does not. For strings, it appears to print a blank line. For numbers however, it prints a code which is related to the number of excess numbers, and also the actual numbers involved. The moral is to always use enough hashes (###) in your IMAGE.

DIM... apart from not allowing you to DIM an array with integer variables(see below) Myarc XB has an interesting aberration involving both string and numeric arrays:
In both XBs the following is legal:

```
100 FOR T=1 TO 6
110 A$(T)="TEST"
120 NEXT T
```

HOWEVER, only in TI XB is the following allowed:

```
100 FOR T=1 TO 6
110 A$(T+0)="TEST"
120 NEXT T
```

Myarc XB returns SYNTAX ERROR, but then again, as you are all good neat programmers, you won't have this problem because you will never see it, having got into the habit of writing your program like this:

```
100 DIM A$(6)
110 FOR T=1 TO 6
120 A$(T+0)="TEST"
130 NEXT T
```

with which Myarc XB is very happy - moral: always DIM your arrays! - it would appear that Myarc XB will only set the array default size when the first instance is a simple number - if evaluation is required, it can't handle it. The cure is so simple - just dimension the array first.

The manual I have bears the following claim: "VASTLY IMPROVED ERROR HANDLING SUPPORT" - so of course I have had a close look at this area, and find the claim entirely false (sorry). In most cases, the same error messages are given as in TI XB, but in several instances, quite different messages appear (e.g. SYNTAX ERROR instead of BAD VALUE). In one case, an abbreviated error message is reported - e.g. only SYNTAX ERROR instead of SYNTAX ERROR : RECURSIVE SUBPROGRAM CALL.

In checking error messages out I came across an undocumented TI XB error message- Myarc have also not documented it but worse, have omitted the error trap entirely. Try this one:

```
 100 DEF T(N)=T(N)-1
 110 PRINT T(N)
```

Short program isn't it! In TI XB I received the message:
UDF REFS ITSELF (almost an X-cert message eh!)
BUT in Myarc XB, these two lines crash the system!!!

And also an undocumented error message in Myarc XB, where a standard error message should be:
INVALID ERROR NUMBER IN NNNN
That's a good one - this was produced by the following code:

```
 100 GOTO 120
 110 SUB TRAP
 120 PRINT "ERROR MESSAGE="
 130 SUBEND
```

And in checking out this area of errors, I found something very interesting - not just a difference between the two XBs, but an unreported feature of TI XB. Read the manual on SUB PROGRAMS and it tells you that SUBEND can ONLY be followed by: Another subprogram, REM and END.
Try this:

```
 100 CALL HEH
 110 SUB HEH
 120 SUBEND
 130 PRINT "PROGRAM END"
```

Does it work?
No it doesn't, you get an error message in TI XB, whereas Myarc XB prints "READY".
SO TRY THIS ONE in TI XB:

```
 100 CALL HUM
 110 SUB HUM
 120 SUBEND
 130 !@P-
 140 PRINT "PROGRAM END"
```

and now TI XB works just like the Myarc XB. The word SUB appears to act in the same manner as STOP, and forces the READY message.
Now try this one - in Myarc XB the !@P- is not required:

```
 100 PRINT "START"
 110 CALL MIDDLE
 120 PRINT "RETURNED FROM CALL"
 130 GOTO 180
 140 SUB MIDDLE
 150 PRINT "IN CALL"
 160 SUBEND
 170 !@P-
 180 PRINT "THIS IS AFTER SUBEND"
```

NOW... I am not advocating that you should place your SUB PROGRAMS in the MIDDLE of your programs - let's accept they are better at the end - BUT - they really do not HAVE to be at the end!!!

That covers normal TI XB, now onto the enhancements. The Myarc manual referred to is the one I have : it may be subject to revision!

The manual suggests that:
PROTECTED programs will auto-run and erase if CLEARed or at termination. wrong, They act just like TI XB programs!
DEF can be used with more than one parameter - wrong, it works exactly like TI XB.
RUN and OLD will load a program LISTed in DV80 format- wrong.
DEFINT can be used in subprograms: wrong, it cannot- you CAN use the format suggested but the results are not those desired. Don't try it!
DEFINT can be used for numeric arrays- wrong. Most unfortunate as this is the area that using INT can really save memory!

The original manual suggests a command to force a garbage collection, which can be useful to prevent irritating

problems with music and sprites when you least want them, but the command has not been implemented.

Now we move onto the GOOD bits of the ExBas side of things... and some of these are a lot better than some the ads mention...

First you have commands strangely similar to UNIX commands... PWD and CHDIR ... used in COMMAND mode, these:
CHDIR: sets the name of a default i/o device. If you do not use the command, the default device is DSK1. To use it you type in: CHDIR RD. or similar.
PWD: prints the current default device name.

What use are they? Suppose you are working on a program called PROGRAM, which is on a disk in drive 1 and the default device is DSK1. To save PROGRAM to DSK1 you only type: SAVE PROGRAM and to reload it you only type OLD PROGRAM

The default device name is inserted between OLD, RUN, and SAVE and the file name! Neat.

RUN when used in a PROGRAM uses a QUOTED string, and you can now have in your program:
100 INPUT "PROGRAM NAME?":A$
110 RUN "DSK1."&A$
or similar - something TI wouldn't let us do.

Also, in your program you can use a line like this one:
RUN "DSK2.PARTTWO",CONTINUE
Know what this does? It retains all the variable values from program one for use in program two - including string variables!!! Think about it...

In COMMAND mode, RUN seems to use an unquoted string- quotation marks are not obligatory. They are also optional with OLD and SAVE.
thus RUN DSK1.LOAD and RUN "DSK1.LOAD" are equivalent.

PRINT and DELETE still require quotation marks.

You can also use RUN or OLD to load and run a machine code program which is on disk in memory image format. This may not always work though - see the section on Machine code below!

GRAPHICS are a real delight, and screen displays are set up SO QUICKLY! You can use the 32 column screen, the 40 column screen, and bit map mode, all very easy to use with simple EX BAS CALL commands.

The 32 and 40 column screens use identical commands, except for colour: in 40 column mode, text colour is set by using two parameters with CALL SCREEN!
In BIT MAP mode you have the extra CALLS:
CIRCLE, RECTANGLE, DRAW, DRAWTO, FILL, POINT, WRITE, DCOLOR
where WRITE is used to place text onto the bit map screen, using the 8x8 pixel character size. CALL HCHAR and CALL VCHAR are also available in bit map mode.

In the 32 and 40 column text modes, you have the ability to "window" by setting margins for screen top, bottom, left and right. I recommend you do not LIST a program in a window two characters wide....! The margins can be changed several times, and HCHAR and VCHAR let you place characters outside the margins anyway..

In default, the margins are set for two unused columns on either side of the screen, allowing 28 and 36 printable columns respectively. Unlike standard TI XB, PRINT does NOT scroll characters in the margins (e.g. outside the window).

The reason you can run TI Basic programs without worrying about character sets 15 and 16 is that Myarc XB actually gives you 256 definable characters in 32 character sets - and 32 sprites for good measure. That is enough for most people! (In bit map mode Chars 216 to 255 are not available).

CALL CHARSET restores characters 32 to 95 only, maintaining compatibility, but CALL GRAPHICS(N) restores all predefined characters, as well as restoring default colours and clearing the screen....

VARIABLES can now use lowercase, so that you can use A,a,b,B=2 and have FOUR variables set. CALL color is accepted and remains unchanged when you LIST but has exactly the same effect as CALL COLOR.

If you use Myarc XB to write a program and use lower case variables, the program WILL run in TI XB - but if you edit a line with a lower case variable, TI XB will re-crunch the line and change the lower case to upper case!

Here is one Myarc have kept quiet about.... MERGE format is a remarkably USEFUL thing that TI gave to us. BUT it was so slow... a long program could easily take an hour or so to load! MYARC have performed a miracle, and MERGE now saves and loads almost in a twinkling (especially if you use a RAM disk!). I was astonished at the speed up. Amazed even! Thanks Myarc!

INTEGERS... so often we hear that using integer math makes a CONSIDERABLE difference in speed. Maybe, but not what Myarc has done! You have the ability to define simple variables to be integers. If you do this, any decimal values are ROUNDED not decimal-stripped. The increase in speed? In a simple benchmark I clocked a 20% increase in speed. Handy but nothing to rave over in ads, when there are so many other improvements! The other change is that your integer variable eats up only two bytes instead of 8.

Nice new command: VALHEX. Ever wondered what >A000 was? Then use this command, as follows: PRINT VALHEX("A000") and there you are. Unfortunately the reverse is not available!

You can now use a variety of keys to terminate a input, as well as ENTER and FCTN E and X. Whenever you do enter data, a predefined variable called TERMCHAR is set and can be referenced in your program at any time until the NEXT input! Then your program can branch depending on whether ENTER or AID or BEGIN!!!!..... catch the drift? Unfortunately REDO has not been implemented. Pity.

The various ram card commands can be used in your EXBas program now, such as CALL PART, EMDK, VOL, and RDDIR.

Extra commands include PEEKV and POKEV.

TECHNICAL STUFF

Myarc ExBas comes in the form of a module, a disk , an eprom and a manual. Not bad for the price. Good value!

The MODULE contains ram (>6000 to >8000) into which the memory management routines go. These are needed as there is a lot of paging going on!

The 128K is divided into FOUR pages (of 32K each if your calculator is off...).

Looking at the 24K area >A000 to >FFE8 first...
  PAGE 1= EX BAS PROGRAM
  PAGE 2= VARIABLES
  PAGE 3= STRINGS
  PAGE 4= Basic Interpreter

So altogether you have 72K for program and variables.

In the 8K area >2000 to >4000, you have:
 PAGE 1: For your machine code routines
 PAGE 2: basic interpreter again, and a small area of what
         Myarc call RAM DISK.
 PAGE 3: routines for VDP and Speech, and i/o buffers
 PAGE 4: MORE basic interpreter and the value stack.

As you can appreciate, a lot of code has to be loaded from the disk, and life is great deal simpler if the first thing you do is copy the disk to ramdisk- to do this however you must have the 512K card! Otherwise, every time you QUIT you must reload the whole system from floppy.

If you have copied the disk to ramdisk, the module will load direct from the ramdisk, whatever it is called. The search appears to be: Is the file in RAMDISK? If yes then load, if no then what's on DISK 1.... and so on.

Once the system is loaded from any drive, it takes a look at DSK1 for a file called LOAD. You can bypass this (undocumented) by holding down FCTN and 4 (CLEAR) until the READY message appears.

The system disk has a nice graphics demo by Chris Faherty.

MACHINE CODE at last...

Firstly, just as with TI XB, your m/c routines have a little less than 8K to fit into. You can enlarge the area slightly by typing in: CALL INIT :: CALL LOAD(8194,32,130)

This is about the only time you MUST use call init! If you use CALL LOAD to load a machine code program, it checks to see if CALL INIT has been used and if not, does one for you.

Naturally if you wish to move the m/c boundary back a little, you must first use call load, or else when you load your file, the system will see no call init has been done, do one, and reset the boundary!

Now then...

1. Programs in machine code written for TI XB loading and running will not function. The TI XB equates are wrong! If you have any XB loading programs they will not operate correctly (e.g. the hidden code LOAD program with FUNLWRITER).

2. In general, machine code programs written for editor/assembler will work with the exceptions found below.

DF80 files must be under 8K to load in directly. Longer files can however be loaded using the UTILITY option of FUNLWRITER. (the error message MEMORY OVERFLOW tells you to try Funlwriter!).

MEMORY IMAGE files can usually be loaded using OLD or RUN.

You will find that some machine code programs - in either format - make assumptions regarding the operating environment which do not apply. Two areas of difficulty:

1. GRAPHICS: If the screen is blank or the graphics are not what they should be, try loading the program with the FUNLWRITER Utility option- this restores the environment. I have only found CUBIT to fail, and even then the failure was minor, the program could be used.

2. SPRITES? The question mark indicates I have not traced the problem area, and have not cured it. Although many

programs operate quite happily when loaded with Myarc ExBas (with or without Funlwriter) there are a few which have disconcerting side effects, apparently connected with auto-sprite motion.

In the SSI Frogger-look-alike program for instance, the lorries and logs don't move, which limits the Frogs route somewhat! In other programs in which a missile is fired, you fire the missile, it appears on screen - and then not only does it not move, but you have no further access to the program! presumably due to a loop checking the progress of your missile. This is NOT a console lock out as such, as other elements of the program continue to run quite happily!

3. FORTH: Do not use the original TI loader for Ed/AS, it will not function. Instead use the later modification for mini/memory (the Universal loader).

4. FUNLWRITER. [reminder: Funlwriter is a fairware program: have you sent a donation to the author yet? DO IT!]. The LOAD program with Funlwriter will not work. You must use the UTIL1 file. RUN UTIL1 will function, but as you have a Myarc ram card, Funlwriter does not remember which disk it is on....and will access DSK1 for all menu options. In order to use Funlwriter AND have it remember its work disk you must do the following:

a. Insert FUNLWRITER disk into DRIVE N.
b. Type RUN MGR3.
c. Disk stops, screen is blank.
d. Press 1 then 4 then N.
e. Funlwriter loads, remembers its source drive number, and resets various things to allow you access to otherwise difficult programs such as DM1000, RLE-MAX and others, which would have blank screens otherwise!

5. DM1000, RLE-MAX, any program with odd graphics or no graphics - load with Funlwriter as No.4 above.

6. NEW HORIZON UTILITIES: If you have the source code for the several XB utilities which the New Horizon group have given us, you need to change any EQUates for: PAD, GPLWS, SOUND, VDPRD, VDPSTA, VDPWD, VDPWA, SPCHRD, SPCHWT, GRMRD, GRMRA, GRMWA, SCAN, XMLLNK, KSCAN, VSBW, VSBR, VMBW, VMBR, VWTR, DSRLNK, LOADER, NUMASG, NUMREF, STRASG, and STRREF to external REFerences.
For example, change:

```
  NUMASG EQU >2008
  STRREF EQU >2014
    to:
        REF NUMASG,STRREF
```

Then reassemble. ... in case the normal labels are not used, just watch out for any EQUates to >20NN. You will then need to look up the name!

7. INTERRUPT ROUTINES - as are used in graphics dumps such as DUMP for instance, or clock routines - do not seem to be available. There is only one vector available for interrupt routines and it looks as though Myarc XB is using it.

8. CRU ADDRESS COLLISIONS: The Myarc ram card (needed for Myarc XB) has its own DSR in EPROM and some programs may disable it! - for instance, the disk copying program MASS COPY must be amended to run with the Myarc Ram Card with either version of Extended Basic!

9. The Myarc ExBas does a great deal of memory paging, and it is possible that a machine code program could interfere with this - may be the sprite problem listed above?

--- the Scratch Pad Ram (256 bytes) is used differently to TI XB but no data has been published as yet. Any machine code program using this area MAY have problems.

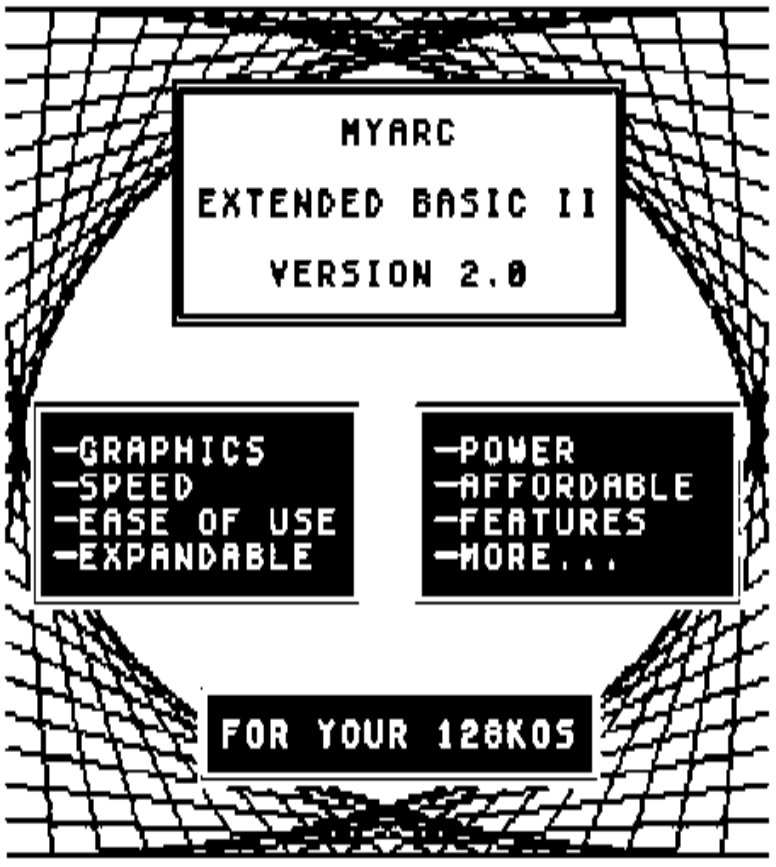
Cassette handling has not been included - possibly as a result of differing VDP mapping?

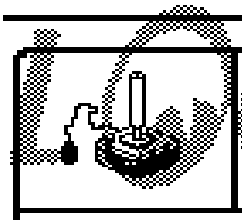
CONCLUSION:

I will retain my TI ExBas module to run the odd bits of code that I do not have source code for but which require the TI XB environment. Otherwise, I can strongly recommend a Myarc ExBas purchase. Only snag is you do need the RAM card... but that in itself is WELL worth having!

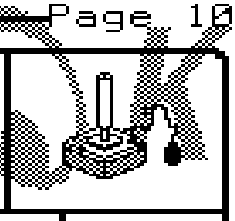
Rediscover Extended Basic. Treat yourself!

(c)October 1986 Stephen Shaw





INTERNATIONAL FUN & GAMES



GAME TITLE	SCORE	JOYSTICK JOCKEY	TI CLUB	DATE
BACKSTEINE	155900	STEVEN JAKABFY	OSHTI UG	09/95
BIGFOOT	290500	DAVID HANDLE	OZARK 99	01/95
BLASTO	44880	MIKE CENDROWSKI	W/PENN 99	11/94
BREAKTHROUGH	1850	RAY FRANTZ	VAST	11/93
BURGER BUILDR	1000000	ELEANOR ZIC	W/PENN 99	03/94
BURGERTIME	82600	MICKEY CENDROWSKI	W/PENN 99	09/85
CAR WARS	6050	JIM WAYNE	VAST	11/93
CENTIPEDE	301930	MICKEY CENDROWSKI	W/PENN 99	01/87
COLORS	1000000	HARRY HOFFMAN	CLEVELAND	03/95
COMBAT	750	AIRSHACK	VAST	02/19
DIG DUG	262460	FRANK ZIC	W/PENN 99	03/94
ENTRAPMENT	3668	FRANK ZIC	W/PENN 99	11/93
HOPPER	4031826	TOM BEERSMAN	OZARK 99	06/94
HUSTLE	WON 52	ELEANOR ZIC	W/PENN 99	03/94
JAWBREAKER	15025	JIM WAYNE	VAST	11/93
JUMPY	131900	ELEANOR ZIC	W/PENN 99	03/94
MICRO PINBALL	1776500	NORM ROKKE	W/PENN 99	05/87
MIONITE MASON	27100	FRANK ZIC	W/PENN 99	11/93
MOON PATROL	73150	MIKE SEALY	W/PENN 99	03/94
MUNCHMAN	202170	PAUL BROCK SR.	W/PENN 99	09/87
PACMAN	153000	GARY TAYLOR	W/PENN 99	09/87
PARSEC	47300	MICKEY CENDROWSKI	W/PENN 99	09/87
PKR SOLITAIRE	3790	JACKIE REMENSKI	VAST	11/93
POLE POSITION	57700	MICKEY CENDROWSKI	W/PENN 99	12/94
SUPER YAHTZEE	615	JACKIE REES	VAST	11/93
THE ATTACK	31800	JIM WAYNE	VAST	11/93
TI INVADERS	15930	PAUL BROCK SR.	W/PENN 99	09/87
TI TRIS	2208	FRANK ZIC	W/PENN 99	11/93
TOMBSTNE CITY	154400	DANNY MCGUIRE	OZARK 99	11/94
TRN SOLITAIRE	351	CAROL HOFFMAN	CLEVELAND	03/95
TREASURE ISLE	37800	MIKE CENDROWSKI	W/PENN 99	10/94
TRIS (ASGARD)	8393	MICKEY CENDROWSKI	W/PENN 99	12/94
YOUR GAME	0000000	YOUR NAME	GROUP?	00/00
YOUR GAME	0000000	YOUR HANDLE	STATE?	00/00
YOUR GAME	0000000	YOUR NAME	COUNTRY?	00/00
YOUR GAME	0000000	YOUR HANDLE	GROUP?	00/00
YOUR GAME	0000000	YOUR NAME	STATE?	00/00
YOUR GAME	0000000	YOUR HANDLE	COUNTRY?	00/00
YOUR GAME	0000000	YOUR NAME	GROUP?	00/00
YOUR GAME	0000000	YOUR HANDLE	STATE?	00/00
YOUR GAME	0000000	YOUR NAME	COUNTRY?	00/00
YOUR GAME	0000000	YOUR HANDLE	GROUP?	00/00
YOUR GAME	0000000	YOUR NAME	STATE?	00/00
YOUR GAME	0000000	YOUR HANDLE	COUNTRY?	00/00
YOUR GAME	0000000	YOUR NAME	GROUP?	00/00
YOUR GAME	0000000	YOUR HANDLE	STATE?	00/00

BOLD LINES INDICATE NEW HIGH SCORE OR GAME SUBMITTED

Please submit all scores to SPARKDUMMER via a private message on the ATARIAGE TI-99/4A forum.

Yesterday's News Information

Yesterday's News is a labor of love offered as a source of pleasure & information for users of the TI-99/4A and Myarc 9640 computers.

TI-99/4A HARDWARE

TI99/4A COMPUTER
MODIFIED PEB
WHT SCSI AND SCSI2SD
MYARC DSQD FDC
MYARC 512K MEMORY
HORIZON 1.5 MEG HRD
TI RS232
CORCOMP TRIPLE TECH
1 360K 5.25 DRIVE
1 360K 3.50 DRIVE
1 720K 5.25 DRIVE
1 720K 3.50 DRIVE

TI-99/4A SOFTWARE

PAGEPRO 99
PAGEPRO COMPOSER
PAGEPRO FX
PAGEPRO HEADLINER
PAGEPRO GOFER
PAGEPRO FLIPPER
PAGEPRO ROTATION
PIXPRO
PICASSO PUBLISHER
BIG TYPE
TI ARTIST PLUS
GIF MANIA

PC HARDWARE

COMPAQ ARMADA 7800
COMPAQ ARMADASTATION
SAMSUNG SYNCMASTER

PC SOFTWARE

DEAD WINDOWS 98SE
FILECAP
PRN2PBNS
IRFANVIEW
ADOBE DISTILLER
ADOBE ACROBAT

Yesterday's News is composed entirely using a TI-99/4A computer system. It consists of 11 PagePro pages which are "printed" via RS232 to PC to be published as a PDF file.