# NIBBLES & BITS

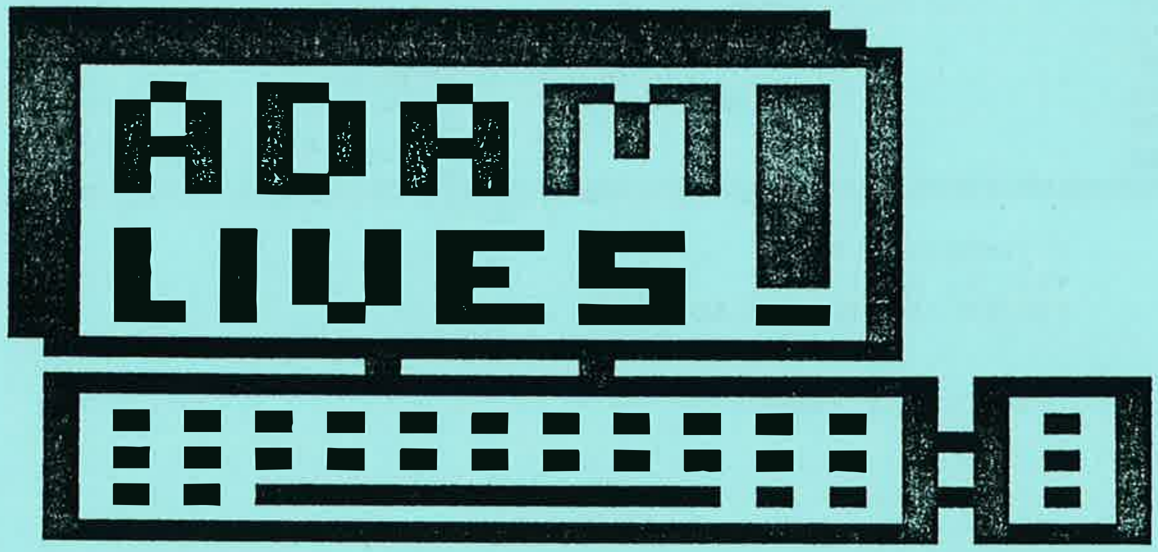
THE COMPREHENSIVE MONTHLY NEWSLETTER FOR ADAM USERS

Published by DIGITAL EXPRESS, INC.
1203 Northwoods Drive, Kings Mtn, NC 28086
(704) 739-9106

September 1986
vol: 1, nmb: 3
SINGLE ISSUE: $3.50

INSIDE THIS ISSUE:

This issue includes 14 SmartBASIC program LISTs, 5 tables (charts), and 7 assembly language lists.



## PUBLIC NOTICE

NIBBLES & BITS is published monthly by DIGITAL EXPRESS, INC. Individual issues may be purchased for the current issue or a back issue (premier issue was July, 1986) for $3.50. The standard subscription rate for one year (12 issues) is $18.00 in the USA, its possessions, and Canada and $24.00 in other foreign countries. The standard subscription rate for six months is $12.00 in the USA, its possessions, and Canada and $16.00 in other foreign countries.

Telephone calls are welcomed. However, specific technical questions regarding ADAM should be mailed.

We welcome contributions of original reviews, programs, articles, questions, and comments. Please include your subscription ID number from your mailing label on all written correspondence to us.

Your subscription ID number is on the first line of your mailing label (affixed to the newsletter). It is a 10 digit code. The first four digits are the month and year of the final issue in your current subscription. Following the ID number is a brief message. If this is your final issue, the message will read "FINAL ISSUE!!!". If this is your penultimate issue, the message will read "TIME TO RENEW". Otherwise the message will apprise you of the exact number of issues remaining in your subscription (excluding the current issue). Please verify this information each month.

To insure that you don't miss an issue, please renew early and let us know promptly of any address change.

## THE N&B STAFF

EDITOR-IN-CHIEF:
Vernon L. Whitman, Sr. (Luke)
TECHNICAL CONSULTANT:
Dr. Solomon Swift
DESIGN EDITOR:
Tim Whetstine
CONTRIBUTING WRITERS:
Jeff Smithers
Janet Weston
Ted Johnson

## DISCLAIMER

## EDITOR'S NOTE

Last month (August) we received 1793 pieces of mail (orders, subscriptions, questions, comments, and programs). WE LOVE IT. *Keep it coming!*

About one third of the mail included compliments on the newsletter. We're glad you like us. Even still, we try to make each issue a little better than the previous one.

Because we do get so much mail, it takes a little longer than we'd like to answer your questions. And (sometimes) it takes longer than we prefer to process product orders and subscriptions. We've added three new employees who do nothing but process mail. This is helping. Thank you for your patience.

Luke

Vernon L. Whitman, Sr.
EDITOR-IN-CHIEF

## N&B NEWS

→ We would like to publicly thank **Lyle Marschand** of **NIAD** for writing a nice review of both our newsletter and Intel-LOAD (rated A) in the last issue (July) of his monthly ADAM newsletter. We received 57 new subscribers from his review. If you don't already belong to the NIAD users group, we highly recommend that you consider their fine publication. Every issue is packed with ADAM info; they have a discount buying service; and, they have huge public domain libraries (probably more than a thousand individual programs).

→ As promised, we've added several new products to our own discount buying service. We plan to increase the selection every month. Also, please note that items have more than one price listed. We offer a special discount to subscribers (this is the price you've seen listed in the two previous issues). We also list the price for non-subscribers.

→ DEI has several software projects underway. Most of these are written in machine code. Our technical consultant has flowcharted a new version of BASIC, compatible with SmartBASIC but much shorter. If enough of you tell us that you'd like to get this (no obligation), he'll start encoding it soon. He estimates a three month effort.

→ Despite all the mail, less than five percent of you have mailed in SWIFT POLL ballots. We have concluded that this paucity is due to the fact that we've required that you cut out the ballot. Beginning with this issue, you can duplicate it, if you prefer. As an added incentive, when we tabulate the results (every three months) we'll put all the ballots in a box to randomly select a subscriber's name. This lucky person will receive a cash prize of $25.00. If you send in all three ballots (they don't have to be identical) within the valid dates, you have three chances of winning. We'll publish the results of the first poll (ending 9-30-86) in next month's issue. So . . . get your ballots in.

→ Have you noticed? Once again, we've changed the print format of the newsletter. We now print 52 characters per line. Let us know what you think about the improvement.

→ Marathon Computer Press will give a 30% discount on their own software to NIBBLES & BITS subscribers. We'll include a control number for ordering from them with the discount in next month's issue. Their current programs include: Codevisor 4.1, The Investment Analyst, The Spanish Vocabularian, and CopyWriter 1.0.

→ We've sold so many Intel-BEST 3.3's that we're starting an Intel-BEST public domain library. Please send in your programs that utilize the features of this package.

## ENTERING PROGRAMS

We usually include several BASIC programs in every issue. We try to keep them short so that they are easy to enter. However, keying in a program can be frustrating.

Here are a few tips which may facilitate the endeavor.

1. If you're a beginner, be EXTREMELY CAREFUL entering programs from the HACKER'S DELIGHT section. All of these programs include machine language. Even one incorrect keypress could lock up ADAM or, worse yet, destroy a datapak or disk.

2. Always SAVE the program before RUNning it. This way if the program does crash, your efforts won't be lost.

3. Print a hardcopy LIST and compare it with the newsletter LISTing.

4. Ninty-nine percent of the time, operational problems are caused by simple typos.

5. Pay particular attention to the numbers one and zero. Don't confuse them with the letter keys (lower case "L" and the upper case "O").

## OUR ERRATA

We had a few typos in the last issue. On page 9, the right column, "CALL 11060 = TEXT" should have read "CALL 11060 = NORMAL". And, "CALL 6169 = RUN" should have read "CALL 6159 = RUN".

On page 14, the right column, the last paragraph, two references were made to bit #2. This should have been bit #4.

The VOLUME NAME CHANGER program on page 19 should have included the following line.

```
150 POKE 16149,255: POKE 16150,255
```

## ADAM NEWS

→ Family Computing Magazine has plans to start a new series of articles entitled "Out Of Production Computers". Among others, information on our ADAM will be featured. The series is scheduled to begin with their November issue.

→ Despite Coleco's claim that they are "out of the electronics industry", they are now marketing two new electronic hand-held educational toys. By the way, Coleco is an acronym for COnnecticut LEather COmpany.

→ A new ADAM supporter, UNITED SOFTWARE, has just released two packages. ULTRA UTILITIES includes several media control utilities. CHARACTER USER is a utility for designing your own font sets. Reviews are upcoming.

→ Marathon Computer Press will soon release a new utility package. When completed, we'll review it. We'll have a review of their THE SPANISH VOCABULARIAN next month -- it's a very nice program.

→ Reedy Software is marketing a new package, MageQuest. It's a text adventure game using the hand-held game controllers. See our review in this issue of their ENTERTAINMENT PACK I.

→ The Hinkle's have finished and are shipping "The Hacker's Guide To ADAM: VOLUME II". It's even better than their first two books. We'll have a detailed review next month.

→ CompuServe is now offering FREE access time for file uploads to data libraries.

→ JJ's Gourmet Hardware and Software Emporium is no longer operating as an individual corporate entity. Orphanware is handling all of their business. Orphanware is an ADAM hardware developer. They have produced a 64K expander, 256K expander, Centronics parallel interface, and an 80 column interface.

■■■■■■■

## EXPANDING YOUR SYSTEM

### BINARY CONVERTERS

If you write or use BASIC programs, you'll most likely need to get one of the popular binary BASIC converters.

These machine coded utilities take a standard BASIC program (filetype A) and convert it to a binary BASIC program (filetype H). This doesn't alter the program's execution. As soon as you enter LIST or RUN the program is converted back to standard BASIC.

What makes these converters *sine qua non* is that you can BSAVE the binary program to load up to TWELVE times faster than normal. Indeed, a binary BASIC program BSAVEd on a datapak can be retrieved almost TWICE as fast as the same program SAVEd on disk!!!

The operation of the converter is rather simple. However, to understand its function, you need to realize the shortcoming of BASIC's standard SAVE command.

Coleco wanted SmartBASIC programs to be completely compatible with SmartWriter. However, SmartWriter was developed by a different company and almost a year earlier than SmartBASIC.

So when SmartBASIC was encoded, it was necessary to SAVE programs in their text format (just like a LIST). Most BASIC's SAVE a program in a cross between machine code and text format called tokenized format. All this means is that numbers are used to represent BASIC words. For example, in tokenized format GOTO is a 2, GOSUB is a 3, and INPUT is a 4.

In fact, BASIC actually uses this tokenized format to execute programs. The text format is only used for LISTs and SAVEs. The binary converters just append a table of pointers to the tokenized code along with a routine that puts the pointers back in their appropriate addresses when you BRUN the program.

Have you ever noticed the brief delay when BASIC LOADs a program? During this pause, BASIC is converting SmartWriter compatible text format to BASIC executable tokenized format. Binary converters simply store your program in tokenized code.

Thus, to be precise, a binary converter doesn't convert anything. Rather, it is a machine code program for bypassing text format in the datapak (or disk) storage process.

## ADAM USERS FORUM

The following questions and comments were culled from recently received mail. Generally, both the reader's input and our response are excerpted from the actual correspondence.

### COLOR BLEEDING

Here is a short program (we have it LISTed below) that has a unique twist to it. The program colors the HGR screen from the top left corner down to the bottom right corner. Then it recolors the screen in each of the other 14 colors.

The twist is that when it colors the screen the first time, there is a perfectly straight, diagonal line that moves across the screen. However, when it colors the screen with the other colors, the program draws steps instead of a straight line. Why does the program behave this way?

David H. Bastock
59 West Lovell Avenue
Akron, OH 44310

IN RESPONSE: This program is a classic example of SmartBASIC V1.0's HGR bug, commonly called color bleeding. Hi-res graphics uses two VRAM tables to define the screen. One table determines which pixels are set and the other determines the color of set pixels (screen dots). The problem is that color can only be set for every eight pixels. On a horizontal line, dots numbered 0 - 7, 8 - 15, 16 - 23, etc. use the same color. You'll notice that each step is eight pixels long.

```
 5 REM color-bleed demo
10 HGR: FOR c = 1 TO 15
20 HCOLOR = c
30 FOR b = 0 TO 191
40 HPLOT 0,b TO b,0: NEXT b
50 FOR a = 0 TO 64
60 HPLOT a,191 TO a+191,0: NEXT a
70 FOR b = 0 TO 191
80 HPLOT 62+b,191 TO 255,b: NEXT b
90 NEXT c
```

## POWER SUPPLY OVERSEAS

I will be moving to Isreal soon. I would like information about using ADAM's power supply in that country.

Dan Rabin
Moshar Hayogev
D.N. Megido 19232
Israel

**IN RESPONSE:** In the USA we normally use 110 volts at 60 cycles. In most foreign countries 220 volts at 50 cycles is standard. You can get a voltage step-down transformer in most electronics stores in the US. You'll probably have to get the cycle step-up converter in the other country.

## WHITE GHOSTS ON SCREEN

Do you know of a fix for ghosts (white distortions) on the computer's TV screen?

Bruce Danyi
171 North Yarrow
Oregon, OH 43616

**IN RESPONSE:** Standard TV's are designed to display pictures rather than characters. Unless your TV has a very high screen resolution, these ghosts appear. If fine tuning doesn't work and all cable connections are correct, you might want to get a monitor. Also, if you change channels with a rotary dial, you might try setting the dial between two channels -- this helps on many TVs.

## Intel-BEST 3.3 QUESTIONS

I have recently purchased Intel-BEST. I would like to know the address where the ASCII value for "I", the INVERSE shortcut, is located. Also, the PR#2 command works with the Orphanware card in slot #2 with my Radio Shack CGP-220 color printer.

David Kennedy
Bldg 844-6-J
Governor's Island, NY 10004

**IN RESPONSE:** Intel-BEST's PR#2 command will work with any parallel interface that accesses port #64. At the first printing of the instruction manual we were not aware that Orphanware's interface (in slot #2) accessed this port. Here are the addresses for the ASCII codes for each of the command shortcuts.

| | |
|---|---|
| 791: | F (for FLASH) |
| 796: | H (for HOME) |
| 801: | I (for INVERSE) |
| 806: | N (for NORMAL) |
| 816: | T (for TEXT) |

## CURSOR CONTROL

In your premeir issue, you mentioned cursor editing features. I think you'll find that you don't need to use CONTROL with the up/down arrow keys to move the cursor around on the screen during line editing with SmartBASIC. I think this implementation of cursor control makes SmartBASIC's line editor one of the slickest on the market.

George A. Havach
550 - 27th Street #202
San Francisco, CA 94131

**IN RESPONSE:** Yes you are correct about not having to use CONTROL with the up/down arrow keys during line editing. This was an oversight on our part. Thank you.

■■■■■■■

DID YOU KNOW?

In 1974 a federal judge, Earl R. Larson, ruled that John Vincent Atanasoff was the true inventor of the concepts required for a working electronic digital computer. John Atanasoff was born in Hamilton, NY on 4 October, 1903. Atanasoff began work on a digital calculator in the early 1930's. In 1942, as a professor in mathematics and physics at Iowa State College, he completed a working model of the Atanasoff-Berry computer (ABC) -- Clifford Berry was a graduate student at the university. The ABC was a serial, binary, electromechanical machine which employed various techniques that Atanasoff had invented.

# BIT BY BIT

## USEFUL DEFINITIONS

Applications software:
A broad term which describes programs or groups of programs that perform specific functions, such as spreadsheet calculations, word processing, etc.

Compiler:
A program that translates the programming instructions of a high-level language (one that uses words) into machine code.

Firmware:
Applications programs which are stored on ROM chips. SmartWriter is an example of firmware.

Font:
The design and size of a particular typeface. Font is commonly used as a synonym for character.

Interpreter:
A program that translates each statement of a high-level language into machine code as each command is entered on the keyboard or executed from memory. SmartBASIC is an interpreted language.

Machine language (or code):
A language consisting of 1's and 0's that the computer understands directly. Machine code is commonly represented in either the decimal or hexadecimal system.

Operating system:
A program or set of machine code routines, generally provided by the manufacturer as part of the computer system, which controls physical operations, such as screen printing and keyboard input.

Sprites:
Graphics characters designed especially for their smooth animation capabilities. ADAM's 16K video chip supports a set a 32 sprites.

Utility programs:
Software that performs frequently required tasks, such as sorting data and copying media.

## THE INPUT COMMAND

The INPUT command allows the user of your program to enter data. The command then assigns that information to the variable name which you specify.

When you use the INPUT command, the typed data is entered when [RETURN] is pressed. If a word is entered when the variable name doesn't end with a dollar sign ($, indicating a string), an error message will be printed.

As with the PRINT command, INPUT may include a string constant. Let's take a look at how to use INPUT.

```
10 PRINT " What's your name?"
20 INPUT name$
30 PRINT " HELLO, "; name$; "!!!"
```

or (in standard shortcut fashion),

```
10 INPUT " What's your name? "; name$
20 PRINT " HELLO, "; name$; "!!!"
```

or (without user response),

```
10 LET name$ = "Bill"
20 PRINT " HELLO, "; name$; "!!!"
```

There are several points worth noting here. First, each of these simple programs accomplishes the same task. The difference is how they go about it.

In the first program, line #10 displays the question on the screen. Then line #20 stops the program and waits for the user to enter something. [RETURN] must be pressed in order to let ADAM know that the entry is finished. Line #30 uses semicolons to separate the string constants from the variable.

The second program uses the string constant question in the INPUT parameter. This saves memory.

The third program doesn't allow the user an entry opportunity. It assumes that his name is Bill.

You can also allow the user to enter more than one piece of information. To do so, separate each variable name in the command line with a comma (also called a delimiter). If you employ this feature, the user must also separate his entries with commas. Here's an example.

```
10 INPUT " Enter three numbers? "; a,b,c
20 PRINT " Their sum is: "; a+b+c
```

# BYTE-SIZED BASIC

## SIMPLE PROGRAMMING TRICKS (part 2)

This month we have four more tricks. Each of these short programs is designed specifically to illustrate a somewhat esoteric BASIC algorithm.

### Randomizing DATA:

There are many situations in which you'll find it necessary to randomize your DATA variables. A good example is randomizing multiple choices for a quiz. We'll show you precisely how to do that next month. But for now, let's take a look at how data can be randomized without duplication of variables. The first program on the top of page 9 shows you how.

The program assigns the nine string variables to the dimensioned variable name "begin". The program uses a simple technique to achieve true randomization (to keep down the LIST length) -- you may prefer to employ the more advanced randomization technique revealed last month. Once the RNG is seeded, the program begins generating random numbers between one and nine, inclusive.

Here's where the logic gets a little tricky. The alogorithm initializes each of the dummy$(x) variables to equal the begin$(x) variables. When the first random number is selected, the program sets the rand$(x) variable corresponding to that number to equal the dummy$(x) variable for that number. Then that dummy$(x) variable is set equal to "used".

This way, when the algorithm generates a new random number, it can check to see if that string has already been randomized to rand$(x).

### Playing with strings:

The second program on page 9 demonstrates some of the features of SmartBASIC's string functions. If you haven't used these functions before, you might want to play around with the values in this program.

### Number crunching:

The next three programs on page 9 illustrate number crunching routines. Unless you are experienced with mathematical reasoning, the logic might elude you at first glance.

The first of these shows you how to let ADAM calculate factorials. The exclamation point is the mathematical symbol for factorial. A factorial is the product of a series of consecutive positive integers from one to a given number.

The next program shows you how to use BASIC to find the factors of a given number. Factors are the integer divisors of a number for which the quotient is also an integer.

The final program on page 9 shows you an easy way to find prime numbers -- integers for which the only factors are one and the given number. Line #120 counts from two to half the given number. This is to exclude one and the number itself. Then if the algorithm finds a factor, it goes on to check the next number. If no factors are found, it prints the prime number.

## POKES TO PLAY WITH (part 3)

### The HOME command:

11090 = execution start
16954 = ASCII of clear character
16993 = number of lines to clear
16994 = right margin
16995 = top margin
16996 = left margin

If you CALL 11090, you'll execute the HOME command.

If you POKE a value other than 32 into 16954, you can change the character that HOME clears the screen to.

In GR or HGR mode, the HOME clear character has to have 128 added to it. For example, if you want the HOME clear font to be the letter "A", in either graphics mode, you'll need to POKE a 193 (65+128) into 16954. This same principle applies to the cursor font (address 16953) mentioned in the August issue. These two modes limit the number of fonts to 96 instead of the 256 for TEXT mode. Rather than displaying characters 32 through 127, fonts 160 through 255 are used.

- continued on page 10 -

```
  10 REM demonstration of how to
  20 REM randomize DATA in BASIC
 100 DIM begin$(9),rand$(9),dummy$(9): TEXT
 110 FOR x = 1 TO 9: READ begin$(x): dummy$(x) = begin$(x): NEXT
 120 DATA ace,bat,cat,doe,eat,fit,got,hat,ink
 130 INPUT " enter random number: ";r1: r2 = RND(-(ABS(r1)))
 140 FOR x = 1 TO 9
 150 r3 = INT(RND(1)*9)+1
 160 IF dummy$(r3) = "used" GOTO 150
 170 rand$(x) = dummy$(r3): dummy$(r3) = "used": NEXT
1000 HOME: VTAB 6: HTAB 12: INVERSE: PRINT "begin";: NORMAL: PRINT SPC(1);
1010 INVERSE: PRINT "random": NORMAL
1020 FOR x = 1 TO 9: HTAB 9: PRINT x;" ! ";begin$(x);" ! ";rand$(x): NEXT
```

```
  10 REM string manipulation demo
 100 TEXT: PRINT " Please enter a word for me"
 110 PRINT " (5 to 10 characters long)?"
 120 INPUT " ";word$
 200 lw = LEN(word$): PRINT: PRINT
 210 PRINT " The first letter is:  ";LEFT$(word$,1): PRINT
 220 PRINT " The last letter is:  ";RIGHT$(word$,1): PRINT
 300 PRINT: PRINT " FORWARD:": PRINT
 310 PRINT " ";word$: PRINT: PRINT
 400 PRINT " BACKWARD:": PRINT
 410 FOR x = lw TO 1 STEP -1
 420 bk$ = bk$+MID$(word$,x,1): NEXT
 430 PRINT " ";bk$
```

```
 .10 REM simple factorial program
 100 TEXT: PRINT " Give me a number (1 - 33),"
 110 PRINT " and I'll instantly tell you"
 120 INPUT " its factorial value: ";nu: PRINT
 200 fv = 1: FOR x = 1 TO nu: fv = fv*x: NEXT
 210 PRINT " ";nu;"! = ";fv
```

```
  10 REM factoring demo
 100 TEXT: PRINT " Enter a number to factor?"
 110 INPUT " ";nu: PRINT: PRINT: PRINT " ";
 200 FOR x = 1 TO nu: y = nu/x
 210 IF y = 1 THEN  PRINT y: END
 220 IF y = INT(y) THEN  PRINT y;", ";
 230 NEXT x
```

```
  10 REM prime number finder
 100 PRINT " prime numbers are: ": PRINT: PRINT
 110 FOR z = 1 TO 100
 120 FOR x = 2 TO z/2: y = z/x
 130 IF y = INT(y) THEN  NEXT z: END
 140 NEXT x: PRINT " ";z: NEXT z: END
```

For example, here's how to change the cursor font and HOME clear character in GR mode.

```
GR     [RETURN]
POKE 16953, 193   [RETURN]
POKE 16954, 223   [RETURN]
HOME   [RETURN]
```

When you're done . . .

```
POKE 16953, 95   [RETURN]
POKE 16954, 32   [RETURN]
TEXT   [RETURN]
```

Here's one way to change the text window in either graphics mode to eight lines instead of SmartBASIC's default four lines.

```
GR (or HGR)
POKE 16958, 16
POKE 16993, 8
POKE 16995, 16
HOME
```

The CATALOG command:

ADAM uses special machine language codes in order to recognize each storage drive. Here are the drive codes.

| | |
|---|---|
| tape one: | 8 |
| tape two: | 24 |
| disk one: | 4 |
| disk two: | 5 |

When you use BASIC's ",d#" suffix with the file handling commands (SAVE, LOAD, CATALOG, etc.), ADAM translates that suffix to the corresponding drive code.

The value is stored in address 16821. You can change drives directly by POKEing the appropriate code into that address.

Here's an alternative to using the actual CATALOG command.

```
POKE 16821, (drive code value)
CALL 21144
CALL 21154
```

You can change the blank spaces printed with the CATALOG command with the following addresses. Note that each address normally contains a 32.

21409
21420
21373

Address 21405 contains the ASCII value of the character that will designate a LOCKed file. It's default value is 42, ie, the asterisk.

Addresses 21454 through 21476 contain the ASCII codes for the "Volume:" and "Blocks Free" messages. The first value for each string is its letter length.

The RECOVER command:

The first time you store a file with a name that already exists in the catalog, BASIC automatically makes a change. The previous filename's filetype is changed to lowercase (A to a or H to h). Then the latest filename is given the normal uppercase filetype.

You can RENAME the newest identical filename and RECOVER the previous filename's filetype. However, SmartBASIC has a minor bug with this command. It won't RECOVER an "h" filetype.

Here are the addresses that control RECOVER's filetype changes.

| | |
|---|---|
| 20590, | 97 (a) |
| 20595, | 65 (A) |
| 20614, | 104 (h) |
| 20619, | 104 (h) |

If you change the value in address 20619 to a 72 (H), you can RECOVER "h" filetypes.

## <u>HOW TO SAVE A GR SCREEN</u>
(part 2)

As promised, here it is -- BlockPAINT!! With this gem of a program you can SAVE, LOAD, and edit your GR screens. The program includes lots of nice features, all of which are designed to make the program very easy to use. The LIST covers the next three pages.

```
  10 REM BlockPAINT
  20 REM Version 1.7
  30 REM COPYRIGHT (c) 1986
  40 REM by DIGITAL EXPRESS, INC.
  50 REM written by Solomon Swift
  60 REM This program is not for public domain!
  70 REM You may, however, enter it, save it, and use it yourself
 100 LOMEM :32000: POKE 16149,255: POKE 16150,255
 110 DATA 62,17,17,0,16,33,0,0,205,38,253,201
 120 FOR x = 27600 TO 27611: READ ml: POKE x,ml: NEXT
 130 DATA enter draw mode,change colors,enter erase mode
 140 DATA load picture,save picture,display catalog,exit program
 150 FOR x = 1 TO 7: READ m1$(x): NEXT
 160 DATA change pen drawing color,change all of one color
 170 DATA review the color values,abort color change option
 180 FOR x = 1 TO 4: READ m2$(x): NEXT
 190 DATA black,lght grn,med grn,drk blue,lght blue,drk red
 200 DATA aqua,med red,lght red,drk yelw,lght yelw,drk grn
 210 DATA magenta,gray
 220 DIM cr$(14): FOR x = 1 TO 14: READ cr$(x): NEXT
 300 POKE 18607,4: POKE 18633,17: POKE 18711,23: GR
 310 hp = 19: vp = 16: co = 8: ud$ = "down"
 320 FOR x = 0 TO 15: POKE 18781+x,x: NEXT
 330 DIM grs(39,31)
 500 GOSUB 8000: cs = SCRN(hp,vp)
 520 FOR x = 1 TO 7: PRINT " ";x;" = ";m1$(x): NEXT
 530 GET key$: k% = VAL(key$)
 540 IF k% < 1 OR k% > 7 GOTO 530
 550 IF k% = 1 THEN  mode$ = "[draw]"
 560 IF k% = 3 THEN  mode$ = "[erase]"
 570 ON k% GOTO 1000,2000,5000,3000,4000,6000,7000
1000 IF mode$ = "[draw]" THEN  cv = co
1002 IF mode$ = "[erase]" THEN  cv = 1
1004 COLOR  = cv: POKE 64885,0
1010 HOME: PRINT " pen status: ";ud$
1020 PRINT " pen color:  ";cr$(cv)
1030 PRINT " mode type:  ";mode$
1040 PRINT: PRINT " d = put the pen DOWN"
1050 PRINT " m = show the MENU"
1060 PRINT " u = lift the pen UP"
1070 IF cv = 3 OR cv = 7 OR cv = 9 THEN  fz = 1: GOTO 1100
1080 IF cv = 10 OR cv = 11 OR cv = 14 THEN  fz = 1: GOTO 1100
1090 fz = 15
1100 k% = PEEK(64885): IF k% <> 0 GOTO 1170
1110 COLOR  = fz: PLOT hp,vp
1120 COLOR  = cv: PLOT hp,vp
1130 FOR de = 1 TO 5: NEXT de: GOTO 1100
1170 key$ = CHR$(k%)
1180 IF ud$ = "down" THEN  COLOR  = cv
1190 IF ud$ = "up" THEN  COLOR  = cs
1200 PLOT hp,vp
1210 IF key$ = "m" OR key$ = "M" GOTO 500
1220 IF key$ = "d" OR key$ = "D" THEN  ud$ = "down": GOTO 1000
1230 IF key$ = "u" OR key$ = "U" THEN  ud$ = "up": GOTO 1000
1240 IF k% >= 160 AND k% <= 163 GOTO 1300
1250 POKE 64885,0: GOTO 1100
```

```
1300 IF k% = 160 THEN  vp = vp-1
1310 IF k% = 161 THEN  hp = hp+1
1320 IF k% = 162 THEN  vp = vp+1
1330 IF k% = 163 THEN  hp = hp-1
1400 IF vp < 0 THEN  vp = 31
1410 IF vp > 31 THEN  vp = 0
1420 IF hp < 0 THEN  hp = 39
1430 IF hp > 39 THEN  hp = 0
1440 cs = SCRN(hp,vp)
1450 POKE 64885,0: GOTO 1100
2000 HOME: FOR x = 1 TO 4: PRINT " ";x;" = ";m2$(x): NEXT
2010 GET key$: k% = VAL(key$)
2020 IF k% < 1 OR k% > 4 GOTO 2010
2030 ON k% GOTO 2100,2300,2200,500
2100 HOME: PRINT " Current color = ";co;
2110 PRINT " (";cr$(co);")": PRINT
2120 PRINT " Please enter your new pen"
2130 INPUT " color (1 - 14): ";co$
2140 co = VAL(co$): IF co < 1 OR co > 14 GOTO 2120
2150 GOTO 500
2200 HOME: FOR x = 1 TO 7: PRINT " ";x;" = ";cr$(x): NEXT
2210 FOR x = 8 TO 9: VTAB 9+x: HTAB 18: PRINT x;" = ";cr$(x): NEXT
2220 FOR x = 10 TO 14: VTAB 9+x: HTAB 17: PRINT x;" = ";cr$(x): NEXT
2230 PRINT " press [RETURN] to continue";
2240 GET key$: GOTO 500
2300 HOME: PRINT " Change which color (1 - 14):"
2310 INPUT " ";c1$
2320 c1 = VAL(c1$): IF c1 < 1 OR c1 > 14 GOTO 2310
2330 PRINT: PRINT " Change it to which other "
2340 INPUT " color (1 - 14): ";c2$
2350 c2 = VAL(c2$): IF c2 < 1 OR c2 > 14 GOTO 2340
2360 HOME: PRINT " one moment . . .": PRINT: PRINT " changing";
2370 PRINT " ";cr$(c1);" to ";cr$(c2)
2380 COLOR  = c2: FOR y = 0 TO 31: FOR x = 0 TO 39
2390 IF SCRN(x,y) = c1 THEN  PLOT x,y
2400 NEXT: NEXT: GOTO 500
3000 HOME: PRINT " Which filename to load?": PRINT
3002 PRINT " REMEMBER:  the filename must"
3004 PRINT " end with .GRP": PRINT: PRINT
3010 INPUT " ?";fa$: fa% = LEN(fa$)
3020 IF fa% < 1 OR fa% > 10 GOTO 3000
3030 IF RIGHT$(fa$,4) <> ".GRP" GOTO 3000
3200 HOME: PRINT " loading ";fa$;" . . ."
3210 PRINT CHR$(4);"bload ";fa$
3300 HOME: PRINT " sorting data . . ."
3310 xy = 0: FOR y = 0 TO 31: FOR x = 0 TO 39
3320 COLOR  = PEEK(xy+30000): PLOT x,y: xy = xy+1
3330 NEXT x: NEXT y
3340 GOTO 500
```

(continued from previous page)

```
4000 HOME: PRINT " What filename to save?"
4010 PRINT " (limit 10 characters)": PRINT
4020 PRINT " (the name must end with .GRP,"
4030 PRINT " eg, pict1.GRP or blocks.GRP)": PRINT
4040 INPUT " ?";fi$: fi% = LEN(fi$)
4050 IF fi% < 1 OR fi% > 10 GOTO 4000
4060 IF RIGHT$(fi$,4) <> ".GRP" GOTO 4000
4300 HOME: PRINT " sorting data . . ."
4310 xy = 0: FOR y = 0 TO 31: FOR x = 0 TO 39
4320 POKE xy+30000,SCRN(x,y): xy = xy+1
4330 NEXT x: NEXT y
4340 HOME: PRINT " saving ";fi$;" . . ."
4350 PRINT CHR$(4);"bsave #,A30000,L1280"
4360 PRINT CHR$(4);"rename #,";fi$
4370 HOME: PRINT " The file is stored.": PRINT: PRINT
4380 PRINT " press [RETURN] to continue"
4390 GET key$: GOTO 500
5000 HOME: PRINT " Which erase option?": PRINT
5010 PRINT " 1 = erase individual blocks"
5020 PRINT " 2 = clear the entire window"
5030 PRINT " 3 = abort this erase option"
5040 GET key$: k% = VAL(key$)
5045 IF k% = 27 THEN  END
5050 ON k% GOTO 1000,5060,500
5060 CALL 27600: GOTO 500
5100 IF k% = 3 GOTO 500
5200 IF k% = 2 THEN  CALL 27600: GOTO 500
5300 co = 1: GOTO 1000
6000 HOME: PRINT " one moment please": PRINT
6010 PRINT " saving screen in RAM . . ."
6100 FOR x = 0 TO 39: FOR y = 0 TO 31
6110 grs(x,y) = SCRN(x,y): NEXT: NEXT
6200 TEXT: PRINT CHR$(4);"catalog"
6210 PRINT: PRINT " press [RETURN] to continue": GET key$
6220 GR: GOSUB 8000
6230 PRINT: PRINT " restoring graphics screen"
6300 FOR x = 0 TO 39: FOR y = 0 TO 31: COLOR  = grs(x,y)
6310 PLOT x,y: NEXT: NEXT: GOTO 500
7000 TEXT: PRINT " program terminated.": END
8000 POKE 16958,16: POKE 16993,8: POKE 16995,16: HOME: RETURN
```

BlockPAINT is a rather long program to type from a LIST. If you take the time, though, you'll have many hours of enjoyment with this one. And, this issue also includes two other programs that allow you to PRINT BlockPAINT files on the ADAM printer or a Panasonic dot matrix printer.

This is an actual LIST of the program. It works. If you encounter execution problems, check your own LIST. Have fun. Let's examine how the program works.

Line numbers 100 through 330 initialize the assorted variables. Line #110 is particularly important. If you enter any of the numbers in this DATA statement incorrectly, ADAM could crash when the program is RUN.

Line numbers 500 through 570 control the first menu of options. When you select an option number, the program will branch one of the seven supporting modules (sub-programs).

Line numbers 1000 through 1450 control the drawing on the GR screen. You can put the pen (graphics window cursor) DOWN to draw blocks or lift it UP to move the pen without drawing. The location of the pen will flash. If the pen color is light, the pen will flash between that color and black. If the pen color is dark, the pen will flash between that color and white.

You move the pen with the arrow keys. The routine that controls the movement is in line numbers 1300 through 1450. You might find it beneficial to allow for diagonal pen movements by adding corresponding lines here for ASCII values 168 through 171.

Line numbers 2000 through 2400 control the "change colors" option. When you're finished with these options, you'll be returned to the first (primary) menu.

You should note that line #320 corrected the COLOR and SCRN color tables to ADAM's master color code values. Option three in the "change colors" menu will let you review the color values. The program won't allow you to draw in black or white because these colors are used to control the pen cursor.

Line numbers 3000 through 3340 constitute the "load picture" module. And, line numbers 4000 through 4390 constitute the "save picture" module.

With the erase option (line numbers 5000 through 5300), you can erase the entire graphics window. Or, you can erase individual blocks. Here, the pen color is set to black.

Line numbers 6000 through 6310 control the "display catalog" option. Here, the GR screen is saved in RAM before executing the TEXT command. This is the technique mentioned last month.

BlockPAINT is designed to be educational from a programmer's point of view, as well as, entertaining from the user's perspective. We hope that you enjoy it.

## HOW TO PRINT A GR SCREEN

The program on page 15 allows you to print the GR screens, that you save with BlockPAINT, on the ADAM printer. You have a variety of print options. If you select double width and double length, the picture will fill an entire sheet of standard size typing paper. With the single width option, you can set the left margin to your choice.

The printed characters are ASCII characters (32 - 45). Black is printed as a blank space.

■■■■■■■

**DID YOU KNOW?**

In 1948, John Bardeen, Walter H. Brattain, and William Shockley (American physicists) invented the transistor. On 3 October, 1950 their patent was finally issued. In 1956, all three were awarded the Nobel Prize in physics for their invention.

In the late nineteen fifties, the transistor was first used in the manufacture of computers, replacing the troublesome vacuum tubes. This development marked the beginning of the second generation of computer circuitry.

Incidentally, in recent years William Shockley has gained some degree of notoriety for his beliefs that poverty proliferates lower intelligence. Many have interpreted this as a racist slur.

```
  10 REM GR screen dump to ADAM printer
  20 REM designed for use with BlockPAINT files
 100 LOMEM :32000: GR
 110 HOME: PRINT " Load which BlockPAINT file? "
 120 PRINT " (filename must end with .GRP)"
 130 INPUT " ? ";fa$: fa% = LEN(fa$)
 140 IF fa% < 1 OR fa% > 10 GOTO 110
 150 IF RIGHT$(fa$,4) <> ".GRP" GOTO 110
 160 HOME: PRINT " loading ";fa$;" . . ."
 170 PRINT CHR$(4);"bload ";fa$
 300 xy = 0: FOR y = 0 TO 31: FOR x = 0 TO 39
 310 COLOR  = PEEK(xy+30000): PLOT x,y: xy = xy+1
 320 NEXT x: NEXT y
 500 HOME: PRINT " 1 = dump graphics to printer"
 510 PRINT " 2 = load another file"
 530 PRINT " 3 = exit program"
 540 GET key$: k% = VAL(key$)
 550 IF k% < 1 OR k% > 3 GOTO 540
 560 IF k% = 2 GOTO 110
 570 IF k% = 3 GOTO 5000
 600 HOME: PRINT " 1 = print single width"
 610 PRINT " 2 = print double width"
 620 GET key$: k% = VAL(key$): IF k% < 1 OR k% > 2 GOTO 620
 630 lm$ = "": IF k% = 2 THEN  sp$ = " ": GOTO 800
 700 HOME: PRINT " Enter left margin in inches:"
 710 INPUT " (0 to 4): ";lm: IF lm < 0 OR lm > 4 GOTO 710
 720 FOR x = 0 TO lm*10-1: lm$ = lm$+" ": NEXT
 800 HOME: PRINT " 1 = single length": PRINT " 2 = double length"
 810 GET key$: le% = VAL(key$): IF le% < 1 OR le% > 2 GOTO 810
 900 HOME: PRINT " press [RETURN] to print"
 910 PRINT " press [ESCAPE] to abort"
 920 GET key$: IF key$ = CHR$(13) GOTO 1000
 930 IF key$ = CHR$(27) GOTO 500
 940 GOTO 920
1000 PR #1: xy = 0: FOR y = 0 TO 31: PRINT lm$;: FOR x = 0 TO 39
1010 co = PEEK(xy+30000): xy = xy+1
1020 PRINT CHR$(31+co);sp$;: NEXT x
1030 IF le% = 1 THEN  PRINT CHR$(13);: NEXT y: GOTO 1050
1040 IF le% = 2 THEN  PRINT CHR$(13): NEXT y
1050 PR #0: GOTO 500
5000 TEXT: PRINT " program terminated.": END
```

# HACKER'S DELIGHT

## HACKER'S NOTES

→ Since we're starting the workshop on CP/M 2.2 with this issue, we'll refer to that operating system as FDOS (Full Disk OS) and ADAM's standard operating system as EOS (Elementary OS).

→ When using the EOS routines (write block, init medium, read VRAM, etc.), you need to set up certain registers first (as explained in the previous issues). However, the sequence of this set up is inconsequential. For example, when using the "read block from storage medium" routine discussed last month, you could set up HL first, then DE, then BC and the accumulator (register A) and then CALL the routine.

→ Many of the EOS routines change the values in some registers. If you CALL the routine from another machine language module, it's a good precaution to PUSH all the register pairs (onto the stack) first. Then, POP them back off the stack (in reverse order) when the EOS routine is finished.

→ BASIC programming is much more efficient when you create several subordinate routines first. Then, depending on the program, you can merge particular routines into one large group and use them as subroutines that are branched to, from the main BASIC program or from individual modules.

→ In machine language, creating subordinate routines is not merely an efficient facility; it is a MANDATORY requirement. For this reason, we've been showing you how to use some of EOS's routines (after all, an operating system is simply a collection of routines for controlling peripherals).

→ BASIC permits slipshod programming techniques; machine language does not. Machine code programming is much more precise and requires structured techniques and an in-depth understanding of the system.

→ VRAM is an indirect 16K RAM chip; it is not directly accessible from BASIC. You can access this Texas Instruments chip (the TMS9918A) in one of two ways. You can use the EOS VRAM routines (listed in the July issue) or you can transfer data through port numbers 190 and 191.

→ Most changes to VRAM from BASIC are only temporary. The TEXT, HOME, GR, and HGR commands alter the contents of the chip.

→ In BASIC there are usually several different ways of accomplishing any given task. In machine code, moreover, the possibilities are virtually endless. More explicitly, when we demonstrate a particular Z80 and/or EOS algorithm, you can be certain that many alternatives exist.

## ACCESSING THE Z80 PORTS

Input and output (I/O) transfers on a microcomputer are accomplished through special memory locations called I/O ports. For the most part, you don't need to concern yourself with ADAM's various ports, if you use the EOS routines.

However, certain situations warrant that you create routines for accessing a particular port. For example, you might want to access the 64K expansion RAM. There is a special port (127) which allows you to "bank switch" 32K sections of this RAM in or out of the main 64K RAM.

Another example is using the sound chip. The EOS includes five routines for sound chip access, but they are extremely complicated. It's much easier to transfer data directly to the chip.

This month we'll take a look at accessing a parallel printer. As a demonstration, a program is included that will print a hardcopy of BlockPaint files on a Panasonic KX series dot-matrix printer. Using this particualar screen dump program with another printer will require minor modifications. We've included printer control codes that are common to most newer dot-matrix printers so that conversion will be easy.

Most peripherals are MUCH slower than the computer itself. This is particularly true of a printer. For this reason, data transfer routines must include a technique to slow down the computer. There are three general methods of accomplishing this: looping, polling, and interrupting.

Looping is the simlist method. Here, a part of the transfer routine keeps repeating an IN status check of the device. When the device is ready, the byte is sent. This is the method we've used.

Polling is typically used for sending data to each of several devices. Hardware interrupts allow the CPU to continue other operations until the device is ready.

## USING A PARALLEL PRINTER

The program on page 18 (continued on page 19) allows you to print a hardcopy of your BlockPAINT files on a parallel printer using bit image graphics. The print routine can be used on any parallel printer. However this particular program is designed for the Panasonic KX series (1080, 1090, 1091, 1092). With a few changes, you can adapt the program to another printer.

You have a variety of options to choose from. First, the program will load the BlockPAINT file that you select. Then the file will be displayed on the GR screen. You can choose the dot density (vertical lines per inch). You can select the horizontal line width and the left margin.

All black portions of the screen will be printed as empty space on paper. Any other color will be shaded. You could modify the program to print a different bit configuration for each individual color. It is currently set for all bits to be set, ie, 255 -- line #1130.

Line numbers 140 and 150 reset the printer, just as if you had turned it off and then back on. Line numbers 160 and 170 set the vertical spacing so that there will be no blank space between each row of the picture.

Line number 1110 is the set up for each individual row. Each row ends with a CHR$(10) and a CHR$(3), ie, a line feed and an end of ASCII message to the print routine.

The graphics on the top of page 2 of this issue were printed with this program. The three GR graphics programs in this issue can offer many hours of fun for the entire family.

Most of our subscribers who do have dot-matrix printers, mentioned on their application that they use Panasonic printers. If you have a problem adapting the program to a different parallel printer, just write to us. We don't know the specifics of every printer on the market, but our technical consultant should be able to help.

## THE PRINTER ROUTINE

The printer routine used by this program is detailed in the assembly language list on the top of page 21. This routine is designed for use as an OS routine.

To use the routine, you must first set up the HL register pair with the starting RAM address of the ASCII code to be transferred. Then, you call (or jump) to the print routine. This set up is detailed in asmb #11.

Getting back to the print routine itself (asmb #10), Line #1 loads the ASCII value pointed to by the HL register pair into the accumulator. It then checks to see if the code is a 3. If it is, it returns either to BASIC (if jumped to) or the previous routine (if called).

The next section of the algorithm checks for ready to print status. First it PUSHes the AF register pair, to save the original value in the accumulator. When the printer is ready, it POPs back the previous accumulator value. This loop will continue until the status bit indicates readiness.

Next, the routine sends the current accumulator value to the printer. Then it increments HL (for the next character). Then the loop goes back to the beginning of the algorithm.

## Z80 EOS INPUT

The EOS includes three routines for keyboard input. For general applications, you'll only need to employ the "read keyboard" routine.

| | | |
|---|---|---|
| 64620 read keyboard | (108,252) | (FC6C) |
| 64587 end read keyboard | ( 75,252) | (FC4B) |
| 64680 start read keyboard | (168,252) | (FCA8) |

With the read keyboard routine, the carry flag is set if a key is pressed. If not, the routine will RETurn. For this reason, you need to include your EOS input in a loop that checks to see if a key has been pressed.

- continued on page 23 -

```
  10 REM GR screen dump to Panasonic KX series printer
  20 REM designed for use with BlockPAINT files
 100 LOMEM :35000: GR
 110 DATA 33,0,123,205,135,122,201
 120 DATA 126,254,3,200,245,219,64,203,71,40,250,241,211,64,35,24,239
 130 FOR x = 31360 TO 31383: READ ml: POKE x,ml: NEXT
 140 DATA 27,64,10,3: REM reset printer
 150 FOR x = 31488 TO 31491: READ ml: POKE x,ml: NEXT: CALL 31360
 160 DATA 27,65,7,10,3: REM set vertical line spacing
 170 FOR x = 31488 TO 31492: READ ml: POKE x,ml: NEXT: CALL 31360
 500 HOME: PRINT " Load which BlockPAINT file?"
 510 PRINT " (filename must end with .GRP)"
 520 INPUT " ? ";fa$: fa% = LEN(fa$)
 530 IF fa% < 1 OR fa% > 10 GOTO 500
 540 IF RIGHT$(fa$,4) <> ".GRP" GOTO 500
 550 HOME: PRINT " loading ";fa$;" . . ."
 560 PRINT CHR$(4);"bload ";fa$
 570 xy = 0: FOR y = 0 TO 31: FOR x = 0 TO 39
 580 COLOR  = PEEK(xy+30000): PLOT x,y: xy = xy+1
 590 NEXT x: NEXT y
 700 HOME: PRINT " 1 = dump screen to printer"
 710 PRINT " 2 = load another file": PRINT " 3 = exit program"
 720 GET key$: k% = VAL(key$): IF k% < 1 OR k% > 3 GOTO 720
 730 IF k% = 2 GOTO 500
 740 IF k% = 3 GOTO 5000
 750 HOME: PRINT " How many characters per inch"
 755 INPUT " (80 or 120)? ";cpi
 760 IF cpi <> 80 AND cpi <> 120 GOTO 750
 762 IF cpi = 80 THEN  md = 4
 764 IF cpi = 120 THEN  md = 1
 800 HOME: PRINT " Enter the line width in"
 810 INPUT " inches (0.5 to 8): ";lw: PRINT
 820 IF lw < .5 OR lw > 8 GOTO 800
 830 size = (cpi/40)*lw
 840 IF size = INT(size) GOTO 870
 850 IF cpi = 120 THEN  size = INT(size)+1
 860 IF cpi = 80 THEN  size = INT(size)+.5
 870 IF 8-lw = 0 GOTO 950
 900 HOME: PRINT " Enter the left margin in"
 910 PRINT " inches (0 to ";8-lw;"):";
 920 INPUT " ";lm
 930 IF lm < 0 OR lm > 8-lw GOTO 900
 950 HOME: PRINT " press [RETURN] to print"
 960 PRINT " press [ESCAPE] to abort"
 970 GET key$: IF key$ = CHR$(13) GOTO 995
 980 IF key$ = CHR$(27) GOTO 700
 990 GOTO 970
 995 HOME: PRINT " printing graphics screen . ."
1000 POKE 31488,27: POKE 31489,108: POKE 31490,lm*10
1010 POKE 31491,10: POKE 31492,3
1020 CALL 31360: REM set left margin
1100 td = lw*cpi: hi% = td/256: lo% = td-(256*hi%)
1110 mode$ = CHR$(27)+CHR$(42)+CHR$(md)+CHR$(lo%)+CHR$(hi%)
1120 FOR x = 1 TO size: empty$ = empty$+CHR$(0)
1130 fill$ = fill$+CHR$(255): NEXT
```

```
1200 xy = 0: FOR y = 0 TO 31: begin = 31488
1210 FOR z = 1 TO LEN(mode$): POKE begin,ASC(MID$(mode$,z,1))
1220 begin = begin+1: NEXT
1300 FOR x = 0 TO 39: co = PEEK(xy+30000): xy = xy+1
1310 block$ = fill$: IF co = 1 THEN  block$ = empty$
1320 FOR z = 1 TO size: POKE begin,ASC(MID$(block$,z,1))
1330 begin = begin+1: NEXT
1400 NEXT x: POKE begin,10: POKE begin+1,3
1410 CALL 31360: NEXT y: GOTO 700
5000 POKE 31488,27: POKE 31489,64: POKE 31490,10
5010 POKE 31491,3: CALL 31360
5020 TEXT: PRINT " program terminated.": END
```

```
 10 REM Z80 get special character
100 LOMEM :30000
110 DATA 205,108,252,48,251,254,27,32,247,201
120 FOR x = 27600 TO 27609: READ ml: POKE x,ml: NEXT
130 PRINT " press [ESCAPE] . . ."
140 CALL 27600: LIST
```

```
 10 REM Z80 get character with bracket test (65-127)
100 LOMEM :30000
110 DATA 205,108,252,48,251,203,127,32,247,254,65,56,243,201
120 FOR x = 27600 TO 27613: READ ml: POKE x,ml: NEXT
130 PRINT " press any key in the ASCII"
140 PRINT " range of 65 through 127 . . ."
150 CALL 27600: LIST
```

```
 10 REM Z80 get 3 characters
100 LOMEM :30000
110 DATA 6,3,205,108,252,48,251,16,249,201
120 FOR x = 27600 TO 27609: READ ml: POKE x,ml: NEXT
130 PRINT " press any three keys . . ."
140 CALL 27600: LIST
```

```
 100 LOMEM :30000
 110 re$ = CHR$(13): es$ = CHR$(27): bk$ = CHR$(8)
 200 DATA 62,23,17,8,0,33,0,21,205,38,253,201
 210 FOR x = 27600 TO 27611: READ ml: POKE x,ml: NEXT
 220 POKE 18607,4: POKE 18633,17: POKE 18711,23
 300 GR: GOSUB 2000: GOSUB 3000
 310 PRINT " [RETURN] = change font color"
 320 PRINT " [ESCAPE] = exit this program"
 330 VTAB 24: HTAB 1: GET key$
 340 IF key$ = es$ GOTO 1000
 350 IF key$ = re$ THEN  HOME: GOTO 500
 360 GOTO 330
 500 INPUT " font foreground (0-15): ";ff
 510 IF ff < 0 OR ff > 15 GOTO 500
 520 INPUT " font background (0-15): ";fb
 530 IF fb < 0 OR fb > 15 GOTO 520
 540 PRINT " which font (just press key)?";
 550 GET key$: IF key$ < " " OR key$ > CHR$(127) GOTO 540
 600 key% = ASC(key$): disp = (key%-32)*8: start = disp+5376
 610 hi% = start/256: lo% = start-(256*hi%)
 620 POKE 27606,lo%: POKE 27607,hi%
 630 POKE 27601,(ff*16)+fb: CALL 27600
 640 HOME: GOTO 310
 999 END
1000 TEXT: PRINT " end of program.": END
2000 POKE 16958,16: POKE 16993,8: POKE 16995,16: HOME: PRINT " ";
2010 FOR x = 32 TO 126 STEP 24: FOR y = 0 TO 23
2020 PRINT CHR$(x+y);: NEXT y
2030 PRINT re$;" ";: NEXT x: PRINT: RETURN
3000 POKE 16958,21: POKE 16993,3: POKE 16995,21: HOME: RETURN
```

```
 10 REM instant graphics window color
100 GR: LOMEM :30000
110 DATA 62,17,17,0,20,33,0,0,205,38,253,201
120 FOR x = 27600 TO 27611: READ ml: POKE x,ml: NEXT
200 HOME: PRINT " What color code value for"
210 PRINT " the graphics window (0-15)?"
220 INPUT " ";co%
230 IF co% < 0 OR co% > 15 GOTO 200
300 cp% = co%*16+co%: POKE 27601,cp%: CALL 27600
400 HOME: INPUT " again ('y' or 'n')? ";ag$
410 IF LEFT$(ag$,1) = "y" OR LEFT$(ag$,1) = "Y" GOTO 200
420 TEXT: PRINT " program terminated."
```

## TITLE (asmb#10):
## New OS Routine For Parallel Printer

| Line#: | Label: | Decimal value: | Op-code: | Comment: |
|---|---|---|---|---|
| 1 | GETVAL | 126, | LD A, (HL) | ; load ASCII to accumulator |
| 2 | ENDCHK | 254, 3, | CP 03H | ; check if end of ASCII string |
| 3 | | 200, | RET Z | ; if end, RETurn to BASIC |
| 4 | STATUS | 245, | PUSH AF | ; save accumulator on stack |
| 5 | | 219, 64, | IN A, (40H) | ; get status byte |
| 6 | | 203, 71, | BIT 0, A | ; check status bit |
| 7 | | 40, 250, | JR Z, FAH | ; if not ready, goto STATUS + 1 |
| 8 | | 241, | POP AF | ; retrieve original accumulator |
| 9 | OUTVAL | 211, 64, | OUT (40H), A | ; send ASCII byte to printer |
| 10 | REPEAT | 35, | INC HL | ; set up for next ASCII value |
| 11 | | 24, 239 | JR EFH | ; go back to GETVAL |

## TITLE (asmb#11):
## Set Up For Parallel Printer Routine

| Line#: | Label: | Decimal value: | Op-code: | Comment: |
|---|---|---|---|---|
| 1 | SETUP | 33, 0, 123, | LD HL, 7B00H | ; load location of ASCII start |
| 2 | | 205, 135, 122, | CALL 7A87H | ; exec parallel printer routine |
| 3 | | 201 | RET | ; RETurn to BASIC |

## TITLE (asmb#12):
## Get Special Character

| Line#: | Label: | Decimal value: | Op-code: | Comment: |
|---|---|---|---|---|
| 1 | GETKEY | 205, 108, 252, | CALL FC6CH | ; call OS read keyboard |
| 2 | | 48, 251, | JR NC, FBH | ; if no keypress, goto GETKEY |
| 3 | CHKKEY | 254, 27, | CP 1BH | ; check for [ESCAPE] |
| 4 | | 32, 247, | JP NZ, F7H | ; if not, goto GETKEY |
| 5 | | 201 | RET | ; RETurn to BASIC |

## TITLE (asmb#13):
## Get Character With Bracket Test

| Line#: | Label: | Decimal value: | Op-code: | Comment: |
|---|---|---|---|---|
| 1 | GETKEY | 205, 108, 252, | CALL FC6CH | ; call OS read keyboard |
| 2 | | 48, 251, | JR NC, FBH | ; if no keypress, goto GETKEY |
| 3 | BRACK | 203, 127, | BIT 7, A | ; check if >= 128 |
| 4 | | 32, 247, | JR NZ, F7H | ; if so, goto GETKEY |
| 5 | | 254, 65, | CP 41H | ; check if < 65 |
| 6 | | 56, 243, | JP C, F3H | ; if so, goto GETKEY |
| 7 | | 201 | RET | ; if in range, RETurn to BASIC |

## TITLE (asmb#14): Get Several Characters

| Line#: | Label: | Decimal value: | Op-code: | Comment: |
|---|---|---|---|---|
| 1 | SETUP | 6, 3, | LD B, 03H | ; set up count limit |
| 2 | GETKEY | 205, 108, 252, | CALL FC6CH | ; call OS read keyboard |
| 3 | | 48, 251, | JR NC, FBH | ; if no keypress, goto GETKEY |
| 4 | CHKCNT | 16, 249, | DJNZ F9H | ; if count not 0, goto GETKEY |
| 5 | | 201 | RET | ; RETurn to BASIC |

## TITLE (asmb#15): Text Font Color In Graphics Mode

| Line#: | Label: | Decimal value: | Op-code: | Comment: |
|---|---|---|---|---|
| 1 | SETUP | 62, 23, | LD A, 17H | ; load color code value |
| 2 | | 17, 8, 0, | LD DE, 0008H | ; load # of sets * 8 |
| 3 | | 33, 0, 21, | LD HL, 1500H | ; load VRAM start address |
| 4 | VRAMBY | 205, 38, 203, | CALL FD26H | ; call OS write byte to VRAM |
| 5 | | 201 | RET | ; RETurn to BASIC |

## TITLE (asmb#16): Instant Graphics Window Color

| Line#: | Label: | Decimal value: | Op-code: | Comment: |
|---|---|---|---|---|
| 1 | SETUP | 62, 17, | LD A, 11H | ; load color code value |
| 2 | | 17, 0, 20, | LD DE, 1400H | ; load byte count |
| 3 | | 33, 0, 0, | LD HL, 0000H | ; load VRAM start address |
| 4 | VRAMBY | 205, 38, 203, | CALL FD26H | ; call OS write byte to VRAM |
| 5 | | 201 | RET | ; RETurn to BASIC |

## HACKER'S CONTEST #3

The NIBBLES & BITS Hacker's Contest is a monthly competition. The winner of each contest is randomly selected from the correct responses postmarked within the specified dates. No individual shall be named the winner in three consecutive contests. The winner of each contest shall be awarded ten dollars and a free three month extension to his/her NIBBLES & BITS subscription term. Decisions of the judges are final.

Responses for this month's contest will be considered valid if, and only if, they are postmarked after August 31, 1986 and prior to October 1, 1986. The winner shall be announced in the November issue of NIBBLES & BITS.

Write a SmartBASIC program (it may include machine code in DATA statements), which will save a TEXT screen on data pack or disk. Good luck . . .

The ASCII value of the pressed key is returned in the accumulator. If you don't use a loop to check for the keypress, the routine will load the accumulator with a two.

The three programs on the bottom of page 19 demonstrate usage of the read keyboard routine. The assembly language lists (asmb#12 through asmb#14) correspond respectively to these 3 BASIC programs.

Asmb#12 (on page 21) shows you how to screen input for a special key. If that key ([ESCAPE] in our example) is not pressed, the routine will continue repeating itself.

Asmb#13 extends the acceptable input range over the previous algorithm. To winnow out unwanted keypresses in this manner is referred to as "bracket testing". The allowable range here is 64 through 127, inclusive.

Asmb#14 loops within itself until any three keys are pressed. The DJNZ (16) statement in line #4 is a convenient code. It decrements the value in register B and if it is not a zero yet, it executes a relative jump in the number of bytes specified by the signed displacement.

## A LOOK AT VRAM

The VRAM table arrangement in HGR and GR modes is completely different from that of the TEXT mode. Let's take a look at these tables.

GRAPHICS WINDOW COLOR:
0 - 5119 ( 0, 0 - 255, 19)

TEXT FONT COLOR:
5376 - 5887 ( 0, 21 - 255, 22)

TEXT SCREEN IMAGE:
6656 - 6911 ( 0, 26 - 255, 26)

GRAPHICS WINDOW DEFINITION:
8192 - 13311 ( 0, 32 - 255, 51)

TEXT FONT DEFINITION:
13568 - 14079 ( 0, 53 - 255, 54)

Colors in the graphics window are determined with this formula (gv is the color code value):

(gv * 16) + gv

This table is designed and used in exactly the same manner both by the GR and HGR modes.

The BASIC program on the bottom of page 20 shows you how to instantly change the graphics window color (in either graphics mode). Asmb#16 details the assembly language of the routine.

The text font color in the graphics modes is determined in the same manner as in TEXT mode, ie, (nl * 16) + ns. One difference here is that only 96 fonts are used. Fonts 160 through 255 are implemented in lieu of fonts 32 through 127.

Another difference with graphics mode text fonts is that you can set the color for each individual font. In fact, you can set the color for each of the eight rows of the bit mapped character definition for each of the 96 fonts.

The challenge presented by the first HACKER'S CONTEST (July, 1986) was to set a different color for each of three fonts in either graphics mode. We did not have a contest winner (too bad).

The BASIC program LISTed on the top of page 20 shows you how to change the color of individual fonts in GR or HGR mode. Asmb#15 details the assembly language of the routine.

The routine is currently set to change only one font color. However, by altering the values in the DE and HL register pairs you can change the color of groups of fonts. The following table shows you some substitution values that you might want to experiment with. By using these values, you can create pseudo-inverse characters. For example, you can change all upper case fonts to one color and change the lower case to another color.

ASCII range: 160 - 191
LD DE: 17, 0, 1
LD HL: 33, 0,21

ASCII range: 160 - 223
LD DE: 17, 0, 2
LD HL: 33, 0,21

ASCII range: 160 - 255
LD DE: 17, 0, 3
LD HL: 33, 0,21

ASCII range: 192 - 223
LD DE: 17, 0, 1
LD HL: 33, 0,22

ASCII range: 192 - 255
LD DE: 17, 0, 2
LD HL: 33, 0,22

ASCII range: 224 - 255
LD DE: 17, 0, 1
LD HL: 33, 0,23

In BASIC TEXT mode (if you'll recall from the July issue) two screen image tables are required. This is technique is implemented solely for the purpose of using the FLASH command. The two tables alternate on the screen with the frequency of the FLASH command. Since the two tables are identical except for FLASHing characters, the screen image appears steady. FLASH simply stores the NORMAL characters in the first table and the INVERSE characters in the corresponding screen location in the second table.

The graphics modes use only one text screen image table. There are not enough fonts to support INVERSE characters. And, thus, no need to enable FLASH.

The first 128 bytes of the text screen image table in the graphics modes is used for the top 4 lines in an 8-line text window. The next 128 bytes are for the lower 4 lines in the text window.

The graphics window character definition is different for GR and HGR modes. In HGR mode a set bit is indicated with a one and a reset bit is indicated with a zero. In GR mode the bytes are set up differently in order to create the 6-by-4 pixel vertical blocks. The byte pattern of 252, 240, 192 is repeated for the entire GR graphics window character definition. This information is taken from addresses 18564 through 18587 in the interpreter's area of RAM. You can POKE new values into this RAM data table to achieve different sized blocks.

## HOW TO INIT PROTECT YOUR MEDIA

In the August issue we mentioned that the BASIC OS command INIT searches for the BASICPGM (SmartBASIC) file before initializing a medium's directory. You won't see this file listed with the BASIC CATALOG command because the programmers used a trick to conceal it. We'll discuss this trick next month.

The following INIT protection will ONLY work with the BASIC INIT command! Most machine language INITs do not check for the BASICPGM file.

BASIC allows you to create files that use any of four filetypes, ie, A, a, H, h. Coleco's BASICPGM file has a different filetype. It is an ASCII two, ie, the frowning face font.

This means that the BASICPGM file that we create must also have a filetype of ASCII two. A few POKEs will take care of this for us. Here are the locations where BASIC gets some of its filetypes (with their default values).

DELETE:
20435: 65 (A)
20451: 72 (H)

SAVE:
23925: 65 (A)

LOAD and RUN:
23522: 65 (A)

Here's a simple way to INIT protect your media:

1. Enter this program line:
   10 REM dummy file

2. Enter this (in immediate mode):
   POKE 23925, 2

3. Enter this (remember to press [RETURN]):
   SAVE BASICPGM

4. Enter this:
   POKE 23925, 65

Now enter CATALOG to verify that the filetype is a frowning face. If so, try to INIT it. Now do this to all your important BASIC media and you're safe from INIT.

Later, you may want to enable INIT with a particular medium. To do so, just substitute DELETE and address 20435 in the procedure above.

## GETTING INTO CP/M 2.2

This month we barely have room to establish a few CP/M preliminaries. Next month, we'll go into a little more detail.

When first exposed to CP/M, most of us are somewhat bewildered. Hopefully, the articles to follow in this workshop will augment your understanding of this powerful system.

Your very first step should be to read the CP/M manual. Naturally, some of the information in the book will not be entirely lucid in the beginning. You'll probably want to get at least one of the many CP/M self-tutorials. CP/M is not as popular as it was a few years ago, but there are still hundreds of books and thousands of public domain programs available.

One of the first concepts that you have to acclimate yourself to is CP/M's drive conventions. The drive that CP/M is booted (first loaded) from is ALWAYS drive A. Drive B is the corresponding SIMILAR device. If tape one is drive A, then tape two is drive B. If disk one is drive A, then disk two is drive B. The remaining storage drives (C and D) are assigned to the corresponding DIFFERENT devices. If tape one is drive A, then disk one is drive C. If disk one is drive A, then tape one is drive C.

Once you understand the drive conventions, the next step should be to make a back up of your CP/M 2.2 medium. THIS IS CRITICAL. Some of the CP/M commands can destroy all or part of a medium with just a couple of misguided keypresses.

The manual describes how to FORMAT an ADAM EOS medium to CP/M. Then, you're instructed to use the BACKUP command to copy the entire CP/M medium. If you're new to CP/M, even this procedure can be dangerous. It's much easier and faster to simply use one of the ubiquitous EOS copy utilities. This way you'll copy the special format and backup CP/M at the same time. And, you'll be using a program that you're familiar with to do it. Be sure to copy all the medium's blocks.

In SmartBASIC, the prompt is a single right bracket. With CP/M, the prompt is two characters. The first character is the current drive label and second character is a greater than symbol. If you're using drive A, the prompt will be "A>". If you're using drive B, the prompt will be "B>".

## LOCAL ADAM USERS GROUPS

### MICHIGAN

ADAM Network
P.O. Box 85
East Detroit, MI 48021

### MINNESOTA

Bill Rahn
12426 - 15th Street South
Afton, MN 55001

Downtown Minneapolis AUG
Thomas C. Gilmore
1424 West 33rd Street
Minneapolis, MN 55408

### NEBRASKA

Omaha ADAM Users Club
Norman Castro
809 West 33rd Avenue
Bellevue, NE 68005

### NEVADA

Al Roginski
4327 Thorndale Place
Las Vegas, NV 89103

### NEW YORK

Metro ADAM Users Group
Russell Williams
414 West 149th Street
New York, NY 10031

Genesee Valley ADAM Users
Donald K. Zimmerman
5132 Jordon Road
Silver Springs, NY 14550

## ADAM PRODUCT REVIEWS

PRODUCT: MultiWRITE
MANUFACTURER: Strategic Software
MEDIA TYPE: DDP
GRAPHICS RATING: Not Applicable
INSTRUCTIONS: 85
VALUE FOR MONEY: 95
RECOMMENDATION: highly recommended (particularly for those with a monitor)
PRICE: $36.00 (from Alpha-1)
RATED BY: Robert A. Hopstetter
22 Lehman Street
Lebanon, PA 17042
(717) 273-3646

This is an excellent word processing program, which shows what determination and hard work can provide. I was quite dissatisfied with Strategic's initial software, such as, the super-slow SmartSPELLER and the restrictive PowerPRINT; but, I've seen a different product in MultiWRITE. It's quick, easy to use, and is equipped with some of the formatting directions that SmartWRITER is lacking. The 64-column display screen is nice; but without a monitor's definition, some of the letters are hard to read. Probably, once you get used to the letters and the commands, the program will replace using SmartWRITER.

Some of the features are: variable left and right margins, variable line spacing and hard page breaks, automatic centering, tab setting, word searching, and more. All the features are useful and easy to learn.

Some of the drawbacks are: (1) getting accustomed to working with the spacing of the 80-column page, and (2) adjusting to the smaller letters. But, overall, the price is very reasonable for the enhanced word processing features that are now available.

Strategic Software has shown that their products will keep improving as time passes and more information is available to the third party market. This program and PaintMASTER are two good items.

ASIDE: Strategic Software might improve it marketability of this program by offering current PowerPRINT owners a trade-in reduction for returning their PowerPRINT tape when ordering the MultiWRITE tape; since MultiWRITE includes all the formatting commands as are included with PowerPRINT. Food for thought, Strategic.

PRODUCT: ENTERTAINMENT PACK 1
MANUFACTURER: REEDY SOFTWARE
MEDIA TYPE: DDP/DISK
GRAPHICS RATING: 99
INSTRUCTIONS: 95
VALUE FOR MONEY: 95
RECOMMENDATION: highly recommended
PRICE: $18.95/$16.95
RATED BY: staff

Animation games written in BASIC are usually not very impressive. Reedy's ENTERTAINMENT PACK 1 is quite different from the norm.

This is a collection of three computer classics. All programs are fast loading. The package features smoothly animated sprites, color screens, other graphics, and uses the sound chip very nicely. And the three programs are controlled from a central menu. With out a doubt, the ENTERTAINMENT PACK 1 is Reedy Software's magnum opus . . . to date. Let's encourage them to continue developing software of comparable quality.

BLOCKADE is a breakout type game (break the brick wall with a bouncing ball). You have four skill levels to choose from -- one or two players. And, a pause feature is included.

CONNECT 4 is their ADAM version of the classic board game. Opponents compete to line up four blocks first. You can play against ADAM or a friend.

SLIDE PUZZLE is a computer version of the hand-hand mechanical puzzels in which you rearrage pieces within a frame trying to get them in correct sequence. This is a rather challenging game.

PRODUCT: THE DESKMASTER SERIES
MANUFACTURER: Nickelodian Graphics
MEDIA TYPE: DDP
GRAPHICS RATING: 93
INSTRUCTIONS: 96
VALUE FOR MONEY: 92
RECOMMENDATION: recommended
PRICE: $29.95
RATED BY: staff

For a BASIC program, this is a very nice combination of three compatible filing systems. The package uses SmartKEY referencing in menus and employs color screens. Advanced features include search capabilities and hardcopy print control.

B-DESK is formatted primarily for alphabetical files. M-DESK is formatted primarily for financial files. F-DESK allows you to customize the categories (fields). Saved files are compatible with each DESK.

## BULLETIN BOARD

ADAM SOFTWARE
**REEDY SOFTWARE**
10085 60th Street
Alto, MI 49302

Hacker's Guide: Volume 2
**Ben Hinkle**
117 Northview Road
Ithaca, NY 14850

44 cents in stamps for catalog
**NICKELODIAN GRAPHICS**
5640 West Brown
Glendale, AZ 85302

Two New Software Items
**UNITED SOFTWARE**
P.O. Box 311
Mundelein, IL 60060

Hardware for ADAM
**ORPHANWARE**
P.O. Box 324
Canal Fulton, OH 44614

ADAM Hardware
**ADAMLAND**
795 Garfield
Lander, WY 82520

Looking for Local ADAMites
**Harold L. Shaw**
350 Broken Arrow Court
Indianapolis, IN 46234

Interested in Contacting ADAMites
**Robert A. Hopstetter**
22 Lehman Street
Lebanon, PA 17042

## PRODUCT LIST

### DEI SOFTWARE

■ Intel-BEST 3.3
dynamic enhancement to SmartBASIC -- makes over 3 dozen changes

$24.95 STANDARD PRICE (datapak only)
$18.95 SUBSCRIBER DISCOUNT PRICE (datapak only)

■ Intel-LOAD
converts BASIC programs to LOAD up to 12 times faster -- stays in RAM plus onscreen help

$15.95 STANDARD PRICE (datapak only)
$11.95 SUBSCRIBER DISCOUNT PRICE (datapak only)

### DEI HARDWARE SUPPLIES

■ DEI blank datapaks
DEI offers a lifetime warranty on its storage media -- see the SPECIAL NOTICE for details

$3.50 (each) or $29.95 (for ten) STANDARD PRICE
$2.75 (each) or $24.95 (for ten) SUBSCRIBER PRICE

■ DEI blank disks
Single-sided, double-density, reliable quality

$1.25 (each) or ($11.95 for 10) STANDARD PRICE
~~[illegible]~~ $1.19 (each) or $9.95 (for 10) SUBSCRIBER PRICE

■ DEI ADAM printer ribbons
just like the ones the came with your ADAM

$5.50 (each) or $15.50 (for 3) STANDARD PRICE
$4.95 (each) or $13.45 (for 3) SUBSCRIBER PRICE

## DEI PAPER SUPPLIES

■■■ adhesive labels
white, tractor-feed, fan-fold, 3 1/2 x 15/16, single column

$2.95 (for 500) STANDARD PRICE
$2.25 (for 500) SUBSCRIBER DISCOUNT PRICE

$5.50 (for 1000) STANDARD PRICE
$3.95 (for 1000) SUBSCRIBER PRICE

■■■ blank white paper
tractor-feed, fan-fold, 9 1/2 x 11, 20# wt., clean edge, 250 sheets

$5.95 STANDARD PRICE
$5.45 SUBSCRIBER DISCOUNT PRICE

## DEI EZ-REFERENCE GUIDES

EZ #101
approximately 700 NUMERIC Z80 instructions: decimal, hex, op codes, operands, 9 full-size pages (FREE shipping)

$2.50 (each) STANDARD PRICE
1.95 (each) SUBSCRIBER DISCOUNT PRICE

EZ #102
approximately 700 ALPHABETIC Z80 instructions: decimal, hex, op codes, operands, 9 full-size pages (FREE shipping)

$2.50 (each) STANDARD PRICE
1.95 (each) SUBSCRIBER DISCOUNT PRICE

## DATA DOCTOR SOFTWARE

■■■ SmartBEST V1.0
enhances SmartBASIC

$18.95 STANDARD PRICE
$16.95 SUBSCRIBER DISCOUNT PRICE

■■■ SmartTRIX I
a set of 10 programmer utilities (including two extremely nice sprite design programs) and a 68 page manual

$34.95 STANDARD PRICE
$29.95 SUBSCRIBER DISCOUNT PRICE

■■■ STRATEGY STRAIN I
a set of 9 computer classics selected for their intellectual challenge (graphics, sound, SmartKEYS)

$24.95 STANDARD PRICE
$18.95 SUBSCRIBER DISCOUNT PRICE

■■■ QUIKFAX QUEST I
three academic quizes (U.S. capitals, world capitals, elements of chemistry)

$24.95 STANDARD PRICE
$19.95 SUBSCRIBER DISCOUNT PRICE

## COLECO PRODUCTS

■■■ SmartLOGO
Coleco's version of the popular language

$34.95 STANDARD PRICE
$27.95 SUBSCRIBER DISCOUNT PRICE

■■■ CP/M 2.2
version of the popular operating system configured for ADAM

$34.95 STANDARD PRICE
$27.95 SUBSCRIBER DISCOUNT PRICE

■■■ UNLESS OTHERWISE NOTED, ALL SOFTWARE IS ON DATAPAK.

■■■ THE COLECO PRODUCTS ARE IN VERY LIMITED QUANTITIES. WHEN YOU ORDER COLECO SOFTWARE, PLEASE INDICATE ON THE ORDER FORM IF YOU'LL ACCEPT BACKORDER STATUS OF A TEMPORARILY OUT-OF-STOCK ITEM.

SPECIAL NOTICE: All DEI datapaks are warrantied to be free from defects in material and workmanship. If the storage medium proves defective, return it to DEI for repair or a replacement (at DEI's discretion).

■■■ The prices listed above are effective 9-1-86 thru 9-30-86

## PRODUCT ORDER FORM:

YOUR NAME: ______________________________

ADDRESS: ________________________________

CITY: _______________ STATE: __ ZIP: _______

PHONE NUMBER: ___________________________

SUBSCRIPTION ID NUMBER: _________________

| < ITEM/QUANTITY/MEDIA > | < PRICE > |
|---|---|
| <______________________________> | $<___.__> |
| <______________________________> | $<___.__> |
| <______________________________> | $<___.__> |
| <______________________________> | $<___.__> |
| <______________________________> | $<___.__> |

| | | |
|---|---|---|
| SUBTOTAL: | $____.__ | |
| SHIPPING: | ____.__ | (inside contiguous USA: $2.50; elsewhere: $4.00) |
| TAX: | ____.__ | (NC residents only -- 4.5%) |
| OTHER: | ____.__ | |
| OTHER: | ____.__ | (subscription/renewal) |
| TOTAL: | $____.__ | |

To order: complete this form, and send check or money order (US FUNDS) to:

DIGITAL EXPRESS, INC.
1203 Northwoods Drive
Kings Mtn., NC 28086

## SOFTWARE EXCHANGE

Our first three public domain libraries are completed. Each BASIC PD library contains over 70K of programs. Each library includes two instruction files which can be read or printed from SmartWriter. All BASIC programs are speed-RUN. Most of the programs are controlled from a "ramdisk" written primarily in machine code.

Each library is available on datapak for $7.95.

To get a free copy of a specific library: (1) contribute an original program, (2) send a signed statement that the program is not copyrighted, (3) send the program on a data pack, (4) request the specific library that you want in return, and (5) include return postage and a mailer or $2.50 for shipping. "FAMILY COMPUTING" programs are not accepted.

The first three BASIC volumes are: B-1.0, B-2.0, and B-3.0. Below is list of the programs on the B-1.0 volume.

```
VOLUME TITLE:  N&B B-1.0              FREE BLOCKS:  170

BOOT        :S  1   DIRECTORY  :S  1   HELLO      :A  1   ml.obj      :H  3
GoHACKER    :H  2   HackerDISK :H  6   LibertyBel :H  2   BarGraph    :H  2
TicTacToe   :H  5   HelixDance :H  2   EZtuneEDIT :H  4   PEEKreview  :H  1
INPUTcntl   :H  3   ScrnCOLORS :H  5   ASCIIdemo  :H  1   SmKEYdemo   :H  2
FullCirc    :H  1   EmptyCirc  :H  1   HxDcChart  :H  1   3Dart#1     :H  1
MlBkCOL     :H  1   MlTEXTcol  :H  1   MlINVblox  :H  1   DAC-Calc    :H  9
MetriCON    :H  9   BASICPGM   :M  1   UtCopy.BIG :H  6   CatPRT.BIG  :H  3
READ-1.WPR  :H  5   READ-2.WPR :H  4
```

## SWIFT POLL BALLOT

As a NIBBLES & BITS subscriber, you are invited to submit one SWIFT POLL ballot per month. This ballot may be cut out or duplicated and must be postmarked prior to Oct. 1, 1986 in order to be verified as valid. The results for the current poll shall be tallied and made public in the October issue of NIBBLES & BITS. Please list your top ten software preferences for the month of September, 1986 in the order that you like them (best first, next best second, etc.).

**YOUR NAME:**______________________ **SUBSCRIPTION ID NUMBER:**______________________

1. ______________________ 6. ______________________
2. ______________________ 7. ______________________
3. ______________________ 8. ______________________
4. ______________________ 9. ______________________
5. ______________________ 10. ______________________