

December 1986
 vol: 1, nmb: 6
 SINGLE ISSUE: \$3.50

THE N&B STAFF	2
PUBLIC NOTICE	2
DISCLAIMER.	2
EDITOR'S NOTE	3
N&B NEWS.	3
ADAM NEWS	4
EXPANDING YOUR SYSTEM	4
ADAM USERS FORUM.	5
BIT BY BIT	
The GET Command.	6
Studying a BASIC Program	6
BYTE-SIZED BASIC	
POKES to Play With (part 6).	8
SmartBASIC 2.0 Features.	9
Using the Game Controllers in BASIC.	9
BASIC ANIMATION.	9
HACKER'S DELIGHT	
Hacker Notes	11
The Fundamentals of Z80 Programming.	11
Assembly Language Mnemonics.	12
Transferring Data.	12
ADAM's Operating System.	13
Transferring SmartBASIC.	16
32 AND 40 Column TEXT Modes.	16
GETTING INTO CP/M 2.2	
THE BUILT-IN COMMANDS (part 3)	22
ADAM PRODUCT REVIEWS.	23
ADAM ACCESS	25
LOCAL ADAM USERS GROUPS	27
BULLETIN BOARD.	27
PRODUCT LIST.	28
PRODUCT ORDER FORM.	30
SOFTWARE EXCHANGE	31
HACKER'S CONTEST.	21

This issue includes 8 SmartBASIC program LISTs and 2 assembly language lists.

COLECOVISION, ADAM, SmartBASIC, and SmartWriter are registered trademarks of Coleco Industries, Inc.
 CP/M 2.2 is a registered trademark of Digital Research, Incorporated.

NIBBLES & BITS is printed in the USA. Copyright (c) 1986 by DIGITAL EXPRESS, INC. All rights reserved.



THE N&B STAFF

FOUNDER:

Vernon L. Whitman, Sr.

EDITOR-IN-CHIEF:

Dr. Solomon Swift

DESIGN EDITOR:

Tim Whetstine

TECHNICAL DIRECTOR:

Chris Davidson

CIRCULATION DIRECTOR:

Tony Michaels

CONTRIBUTING EDITORS:

Janet Weston

Ted Johnson

Cindy Harrington

PUBLIC NOTICE

NIBBLES & BITS is published monthly by DIGITAL EXPRESS, INC. Individual issues may be purchased for the current month or for 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, and \$16.00 in other foreign countries.

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. Please include your subscription ID number on address change notifications.

DISCLAIMER

The editor(s) and publisher have exercised due care in the preparation of this newsletter. Neither the N&B staff, nor DEI, nor any contributors (of any capacity) make any warranty either expressed or implied with regard to the information contained herein either by interpretation, use, or misuse. Reviews and opinions submitted by the readership at large do not necessarily reflect the opinions of the editor or staff. DEI has no affiliation with Coleco Industries, Inc. Unless otherwise stated, all correspondence shall be considered as "open to public review".

EDITOR'S NOTE

N&B NEWS


As the new year approaches, I'm reminded of the day that Coleco announced it was to discontinue production of the ADAM computer. It was January 2, 1985. As news of this effort to boost stock value circulated, it seemed that ADAM's doom was certain.

In retrospect, we can readily conclude that this action by Coleco has, in fact, been beneficial to ADAM. The prospects for our orphaned computer are improving at an almost exponential rate. This fact alone prompts a vital question. What is the motivation for these advances?

Sure, ADAM is a sound investment. Yes, its a great value. Indeed, it is a superior 64K computer. And, profit is certainly a consideration for third party developers.

The simple fact, though, is that ADAM users, themselves, give impetus to the growth of ADAM products. Each purchase that you make has a threefold benefit. The immediate advantage is that you improve your system. One long range effect is that your purchase encourages that particular company to continue supporting ADAM with new products. Another long range effect is that in the light of the success of existing companies, new developers go into business for themselves.

In short, you play the vital role in perpetuating ADAM. The fact that there is so much software and hardware available for ADAM today, two years after Coleco dropped it, is the direct result of support from ADAM users all around the world. At the current rate of growth, we should expect very favorable developments in 1987.



Dr. Solomon Swift
EDITOR-IN-CHIEF

We now offer the programs from issues of NIBBLES & BITS on data pack or disk. Every three months we compile the programs onto one medium. We have the programs from the July thru the September issue and those from the October thru the December issue ready. You have two ways to get these. You can send a blank disk or data pack along with a return mailer with sufficient postage to get one FREE. Or, you can send \$4.95 to cover the costs involved.

Many of you have asked how to transfer BASIC to a medium without erasing any programs already on the medium. We have a program in the HACKER'S DELIGHT department this month that will do this for you.

We already have two winners in the Jeopardy promotion.

The current SWIFT POLL is drawing to a close. Be sure to get your ballots in soon.

Notice our new low price for disks in the PRODUCT LIST.

The set of pinball games in our public domain library is our own creation. To use them, you must also have the public domain volume 'Pinball Construction with Hardhat Mac'.

A great many of you have gotten ShowOFF I. Please let us know what you think about this software. We'd really like read your comments and any suggestions for improvement.

In the HACKER'S DELIGHT department we're starting a thorough disassembly of the EOS with this issue. Our study of the EOS has been aided by Darrell Sage's (Expandable Computer News) published research.

We now stock Coleco/LORAN data packs. And, E&T SOFTWARE is supplying us with their own very high quality data packs. We are so impressed with these that we now stock them vice our own formatted data packs.

Remember the annual N&B subscription will be \$22.00 effective 1/1/87. This is to maintain FIRST CLASS delivery throughout the USA.

□ We are offering a special discount on two DATA DOCTOR packages through 1/31/87. Until then you can get QUIKFAX QUEST or STRATEGY STRAIN on data pack or disk for only \$9.95 each.

□ We have a winner of HACKER'S CONTEST #4. This ADAM hacker received a \$10.00 prize and a three month extension to his NIBBLES & BITS subscription. The winning hacker is:

Michael Bogrees of Englewood, OH

ADAM NEWS

□ MMSG, who developed BACKUP+ 3.0, will soon release a new package. EASY COME - EASY GO will help you answer a variety of financial questions. We'll have a review in a future issue.

□ Marathon Computer Press normally offers a 30% discount to our subscribers. But, for a limited time, they're offering a super discount exclusively to NIBBLES & BITS subscribers. See the ADAM ACCESS department this month for details.

□ KAUG (from Kansas), mentioned last month, has announced that they are discontinuing their newsletter. They have generously offered to give DIGITAL EXPRESS all their public domain volumes.

□ If you're a subscriber to the ADAMland newsletter, and haven't heard yet, they have suffered substantial data loss due to lightning damage. They ask that their subscribers contact them in writing as soon as possible.

□ When playing Jeopardy, the first player responds by tapping the space bar. The second player responds by pressing a trigger on the #1 game controller. The third player responds by pressing a trigger on the #2 game controller.

□ Generally the most frequent break down with ADAM occurs with the printer. This is most unfortunate because the printer houses ADAM's power supply. American Design Components has a viable solution or preventive measure. They offer the ADAM printer, less the top cover plate, for \$69.50 plus shipping. They also have disk drive power supplies for \$14.95 plus shipping.

EXPANDING YOUR SYSTEM

PRINTER ALTERNATIVES

(part 2)

As we discussed last month, most dot matrix printers connect to a computer via the Centronics parallel interface. Both Eve Electronics and Orphanware offer this interface for ADAM. The product quality from each of these companies is very good. But, there are some differences worth mentioning. The following is not a review; it is merely a comparison of the two products.

Eve calls its parallel interface the SP-1P. Orphanware's is the PIA2. The SP-1P usually sells for about \$80.00. The PIA2 usually sells for about \$50.00.

The SP-1P comes with the cable to connect to a printer. The PIA2 does not; you must purchase the cable separately. The price for one of these varies from seven to about fifteen dollars.

Both interfaces come with software that allows you to use the dot matrix printer from SmartBASIC, SmartWriter, SmartFiler, and CP/M. The software that comes with the SP-1P includes some customized functions which permit you to use a few of the particular printer's built-in word processing features. The SP-1P also includes a program that creates a PR#2 command. This allows you to direct printing to either the ADAM or the dot matrix printer from within SmartBASIC.

The SP-1P plugs into the bus extender on the right side of the memory console. The PIA2 plugs into the second slot under memory console cover.

Eve's software modifies the disk manager program that comes with the ADAM disk drive. The Orphanware software does not require that you have a disk drive.

Both interfaces access ADAM through port number 64. This means that they can use the same software. This fact will probably lead to the development of a wide range of printer software.

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.

SmartWriter Tips

A few words concerning SmartWriter ...

ADAM's 1 1/2 line spacing bug can be circumvented by using the Sub/Superscript feature. For example, first type some text and press [return] to end the line. Now press Sub/Superscript then press SUBSCRIPT and continue. Remember that you must complete each line within margins with DONE before going to the next line. Repeat the above procedure to maintain correct line spacing until the end of the paragraph. To start the next paragraph, press Sub/Superscript once more and hit the space bar until the end of the margin is reached. Now, hit DONE and begin the next paragraph in normal fashion.

BOLD TYPING parts of text can be achieved as follows. First print the text (tractor feed will produce better results). Now return the sheet of paper to the beginning of the text (use the knurled knob on the printer). Press HI-LITE and mark all the text to be typed as boldface. Here's the trick. If you need to skip lines between bold typing, you must tell ADAM by HI-LITING at least one space per line to be skipped. Press PRINT and then press PRINT HI-LITE. Good luck!

David Giampietro
2017 Glen Una
San Jose, CA 95125

A CompuServe Alternative

Most ADAM owners that have modems have used the FAMILY COMPUTING FORUM on CompuServe at one time or another. But, the online costs of 'PLink' are almost half the non-prime time rates for CompuServe. PLink even has what they call a HAPPY HOUR when rates are discounted -- at one time they were as low as \$1.00 per online hour.

Users do not even have to have a charge card to use PLink; they can prepay a dollar amount and use the system till that amount is used up. PLink can be reached through both TIMNET and TELENET. And, if you would like more information about PLink, you may call them at 1-800-524-0100.

David E. Carmichael
1325 North Meridian, Apt. #201
Wichita, KS 67203-4637

N&B TIPS

NOTE: Along with a complimentary letter, Mr. Farmer sent several suggestions. The following is an excerpt from this list.

- a list of PEEKs and POKEs cross referenced to Apple's BASIC.
- a program to format data packs such as disks are by Disk Manager.
- a classified section for readers to buy, sell, and communicate.
- a tips from readers section.
- change the Swift Poll Ballot to indicate new items we'd like to see developed.

Hugh Farmer
Covington ADAM Users
10204 South Dinah Circle
Covington, GA 30209

IN RESPONSE: We selected these items from his list for the benefit of all readers. We welcome your suggestions.

We are currently compiling a list of APPLE to ADAM peeks. We would greatly appreciate tips from our readers.

A device may be available in the not too distant future that will convert cassettes to DDPs in 6 minutes. One of the prototypes actually fits inside the ADAM.

The BULLETIN BOARD section may be used for general communication from our readers. The ADAM USERS FORUM is for articles, tips, and questions submitted by our readers. Finally, we would like to hear from any of our readers regarding suggestions for new software to develop for ADAM.

BIT BY BIT

The GET Command

The GET command is used to accept a single keystroke. Like the INPUT command, GET is used to get a response from the program's user. Unlike the INPUT command, the character struck is not printed on the screen. And, with GET the response is accepted as soon as the key is struck. [RETURN] does not necessarily have to be pressed with GET.

The format for GET is:

GET variable

When the program encounters a GET command, it waits until a key is pressed. Then, the value of that key is assigned to the variable. Consider this example:

10 GET key\$

Suppose that the 'H' key is pressed. Then, key\$ = "H" -- or CHR\$(72).

Studying a BASIC Program

The program on the next page is a simple arithmetic quiz. It consists of three parts or modules.

The first part presents the user's choices. A selection of options such as this is typically called a menu. Line numbers 100 through 190 constitute the menu.

Line numbers 1000 through 1130 ask and evaluate the addition question. Line numbers 2000 through 2130 ask and evaluate the subtraction question.

The program uses three variables to keep track of the score. These are:

count = number of questions asked
right = number answered correctly
perc = percentage correct

Line numbers 110 through 160 display the menu screen. Below the three choices, the current quiz status is revealed.

Line number 170 accepts the user's selection. If the first option is chosen, the program branches to line number 1000. If the second option is chosen, the program branches to line number 2000. If any other key is pressed, line number 190 is executed -- this ends the program.

Line number 1000 selects the two numbers to be added. Line # 1010 increments the variable that keeps track of the number of questions asked.

The PRINT command in line number 1020 without a parameter (words to print) simply prints a blank line. Line number 1040 allows the user to enter an answer. If the answer is incorrect, the program branches to line number 1100.

Line number 1070 causes a programmed delay that lasts for two seconds. This gives the user time to read the 'CORRECT!!!' message. Line number 1080 increments the variable that keeps up with the number answered correctly. Line number 1090 calculates the current quiz score and then branches program execution back to the menu.

Line numbers 1100 through 1130 handle the incorrect answer response. Line number 1110 displays the correct answer. The delay caused by line # 1120 lasts four seconds. Line # 1130 branches back to line number 1090. This calculates the current score and goes back to the menu.

The subtraction module is almost identical to the addition module. The exception is the range of the two random numbers. The 'x' variable will have a value between 50 and 99. The 'y' variable will have a value between 0 and 49. This way the correct answer is always positive.



```
100 count = 0: perc = 100: right = 0
110 TEXT: PRINT " - simple math quiz -": VTAB 4
120 PRINT " Which one do you want?": PRINT
130 PRINT " 1 = addition": PRINT " 2 = subtraction"
140 PRINT " 3 = end the quiz"
150 VTAB 16: PRINT "number asked: ";count
160 VTAB 18: PRINT "percentage correct: ";perc
170 GET choice$: IF choice$ = "1" GOTO 1000
180 IF choice$ = "2" GOTO 2000
190 TEXT: PRINT " end of program": END
1000 x = INT(RND(1)*100): y = INT(RND(1)*100): HOME
1010 count = count+1
1020 PRINT: PRINT " how much is ";
1030 PRINT x;" + ";y;"? ";
1040 INPUT " ";ans
1050 IF ans <> x+y GOTO 1100
1060 VTAB 10: PRINT " CORRECT!!!"
1070 FOR delay = 1 TO 1500: NEXT delay
1080 right = right+1
1090 perc = INT(right/count*100): GOTO 110
1100 VTAB 10: PRINT " INCORRECT!!!": PRINT
1110 PRINT " The answer is ";x+y;"."
1120 FOR delay = 1 TO 3000: NEXT delay
1130 GOTO 1090
2000 x = INT(RND(1)*50)+50: y = INT(RND(1)*50): HOME
2010 count = count+1
2020 PRINT: PRINT " how much is ";
2030 PRINT x;" - ";y;"? ";
2040 INPUT " ";ans
2050 IF ans <> x-y GOTO 2100
2060 VTAB 10: PRINT " CORRECT!!!"
2070 FOR delay = 1 TO 1500: NEXT delay
2080 right = right+1
2090 perc = INT(right/count*100): GOTO 110
2100 VTAB 10: PRINT " INCORRECT!!!": PRINT
2110 PRINT " The answer is ";x-y;"."
2120 FOR delay = 1 TO 3000: NEXT delay
2130 GOTO 2090
```



BYTE-SIZED BASIC

POKEs to Play With

(part 6)

Using spaces in filenames:

The BASIC interpreter checks each letter of every filename that you enter. First it checks for upper case letters, lower case letters, and numbers. If the character doesn't match one of these, it then checks it against a table of 10 ASCII values. The table occupies addresses 23231 through 23240. You can change the value at any one of these addresses to a 32 (ASCII for a blank space).

For example, the value at address 23240 is a 95 (an underscore). If you POKE a 32 into address 23240, you can now use a blank space in your filenames. However, you can no longer use an underscore.

You should keep two precautions in mind though, when employing this technique. First, only use spaces between letters. If you use spaces at the end of a filename, you may have trouble later determining the exact number of spaces used. Also, you must POKE a 32 into the appropriate address each time BASIC is booted in order to load filenames that include blank spaces.

Changing BASIC's Drive Suffix:

With BASIC's media commands (CATALOG, SAVE, LOAD, etc.) you can follow the command with a drive suffix in order to change drives. As mentioned in previous issues, the BASIC interpreter converts this suffix to a certain drive code and stores that value at address 16041. The codes are:

```
d1 = POKE 16041, 0
d2 = POKE 16041, 24
d5 = POKE 16021, 4
d6 = POKE 16021, 5
```

As you work with the drive suffixes, you may wonder why 'd3' and 'd4' were skipped. One of the prototypes of ADAM was expandable to four tape drives. These other two drive suffixes were intended to access the third and fourth tape drives.

The table that converts BASIC's drive suffixes to ADAM drive codes occupies addresses 23273 to 23278. Here's how to change 'd3' and 'd4' to designate the first and second disk drives.

```
POKE 23275, 4
POKE 23276, 5
```

This change is simply a convenience. You can still use 'd5' and 'd6' for disk drives.

True BASIC to SmartWriter Compatibility:

Most of us have, at one time or another, used SmartWriter to print a hardcopy of a BASIC program LIST. Have you ever been disappointed with this hardcopy because blank spaces aren't used to format the LIST properly? There's a simple trick that can fix this annoyance.

We mentioned last month that you can POKE a zero into address 16148 to see a LIST exactly as SmartWriter displays it. Then you can POKE a 32 back into the address to restore the LIST function.

SmartBASIC's SAVE command actually does this during the process of writing a program to tape or disk. The routine that performs this function occupies addresses 24099 through 24125. This technique is employed to conserve storage space.

Now, suppose you want to SAVE a program that you'll later print with SmartWriter. Here's how to SAVE it so that SmartWriter will display it exactly as the BASIC LIST command does.

```
POKE 24100, 0
POKE 24101, 0
POKE 24102, 0SAVE filename
```

The program may take two or three more blocks on your disk or data pack than it would have otherwise. But, the formatted SmartWriter hardcopy should more than compensate for that extra storage space.

If you want to restore the SAVE function to its default values, here's all you need to do:

```
POKE 24100, 50
POKE 24101, 20
POKE 24102, 63
```


TEXT Margin POKEs

The following four addresses control the margins for the TEXT command.

17198 (number of rows)
 17199 (number of columns)
 17201 (top margin)
 17202 (left margin)

On some older televisions the character at the left margin is not displayed. Here's how to overcome this limitation.

POKE 17199, 29 (default value is 30)
 POKE 17202, 2 (default value is 1)

SmartBASIC 2.0 Features

(part 2)

Correcting the HGR and GR Color Tables:

In the July issue we explained how to correct the HGR and GR color tables in SmartBASIC V1.0. With this technique, you only have to use one color table -- ADAM's video chip master color code. Here's how to do the same thing with SmartBASIC 2.0.

```
10 FOR x = 0 TO 15
20 POKE 25360 + x, x
30 POKE 25370 + x, x
40 NEXT x
```

Changing The 2.0 Prompt:

Address 1121 contains the number of fonts to be displayed. Address 1122 contains the ASCII value of the first font. Address 1123 contains the ASCII value of the second font.

Miscellaneous Features:

SmartBASIC 2.0 uses an entirely different algorithm to display the cursor. As such, there is no address to POKE the ASCII value of the cursor.

You can still change the ASCII value of the HOME command's blank space. Address 16939 controls this function.

With version 1.0 the default LOMEM setting is 27407. With version 2.0 the default LOMEM setting is 26960. Addresses 1594 and 1595 are the high and low order byte pointers to 2.0's LOMEM value.

Using the Game Controllers in BASIC

"PROGRAMMING WITH ADAM", the SmartBASIC guide that came with your ADAM, only gives a very brief explanation of the PDL function. The program on the next page (page 10) shows you how to use the various PDL functions in SmartBASIC.

To get the value of a particular function you must set a variable to equal the function. This automatically causes BASIC to read the game controller. If you want to wait until a specific game controller function is used, you need to put the read function in a loop. Consider this example:

```
10 IF PDL(9) = 1 THEN GOTO 30
20 GOTO 10
30 REM continue your program
```

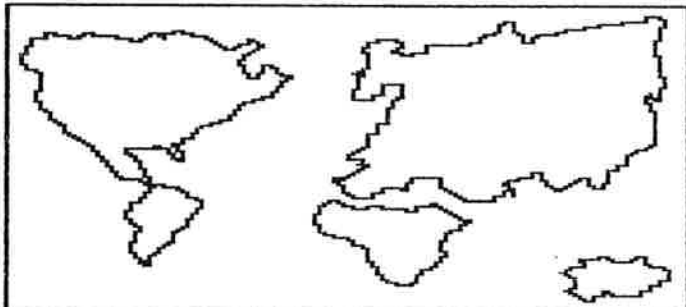
With this simple illustration the program waits until the right trigger on the front game controller is pressed. You could easily substitute some other function in line number 10.

The joystick recognizes nine positions. The list on page A-66 only lists four of them. Here are the nine values that can be returned when you read the joystick with a PDL(5) (for the front controller) or a PDL(4) (for the rear controller).

```
0 = center position
1 = UP only
2 = RIGHT only
3 = UP + RIGHT
4 = DOWN only
6 = DOWN + RIGHT
8 = LEFT only
9 = UP + LEFT
12 = DOWN + LEFT
```

```
50 REM PDL test
100 POKE 16953,0: POKE 16149,255: POKE 16150,255: POKE 64885,0
110 TEXT: HTAB 2: INVERSE
120 PRINT " game controller test ": NORMAL
130 VTAB 4: PRINT " vertical value:": PRINT " horizontal value:"
140 VTAB 7: PRINT " joystick direction:"
150 VTAB 9: PRINT " left trigger:": PRINT " right trigger:"
160 VTAB 12: PRINT " keypad value:"
170 VTAB 20: PRINT " press any keyboard to exit..."
200 ud = PDL(1): fr = PDL(3): dr = PDL(5)
210 ft = PDL(7): rt = PDL(9): kp = PDL(13): kb = PEEK(64885)
220 IF kb <> 0 GOTO 2000
300 IF ft = 0 THEN VTAB 9: HTAB 22: PRINT "off"
310 IF ft = 1 THEN VTAB 9: HTAB 22: PRINT "on"
400 IF rt = 0 THEN VTAB 10: HTAB 22: PRINT "off"
410 IF rt = 1 THEN VTAB 10: HTAB 22: PRINT "on"
500 VTAB 4: HTAB 22: PRINT ud
600 VTAB 5: HTAB 22: PRINT fr
700 VTAB 12: HTAB 22: IF kp = 15 THEN PRINT "nothing": GOTO 800
710 IF kp = 10 THEN PRINT "*": GOTO 800
720 IF kp = 11 THEN PRINT "#": GOTO 800
730 PRINT kp
800 VTAB 7: HTAB 22: IF dr = 0 THEN PRINT "center"
810 IF dr = 1 THEN PRINT "up"
820 IF dr = 2 THEN PRINT "right"
830 IF dr = 3 THEN PRINT "up+right"
840 IF dr = 4 THEN PRINT "down"
850 IF dr = 6 THEN PRINT "down+right"
860 IF dr = 8 THEN PRINT "left"
870 IF dr = 9 THEN PRINT "up+left"
880 IF dr = 12 THEN PRINT "down+left"
1000 GOTO 200
2000 POKE 16953,95: VTAB 22: END
```

```
ADAM      ADAM      ADAM      ADAM      ADAM
ADAM
ADAM
ADAM
ADAM
ADAM
ADAM
ADAM
ADAM
ADAM
ADAM
ADAM      ADAM      ADAM      ADAM      ADAM
```



HACKER'S DELIGHT

Hacker Notes

This month we're taking a look at three different aspects of machine code programming. We're progressing through the SmartBASIC enhancements by revealing a program that will allow you to use the video chip's 40 column mode as an alternative to the standard 32 column TEXT mode.

And, we're starting a detailed disassembly of ADAM's EOS. We'll break down each of the 101 main EOS routines. We'll explain what registers to set up before calling a routine, explain what happens within the routine, and reveal the values returned in the registers. We'll try to include an example program with each routine's explanation. This month we're beginning the series of EOS articles with a list of the EOS main routines.

Many of you have written to ask how we come up with our Z80 programs, enhancements, etc. The main question seems to be 'how do you know which numbers perform what function?'. We mentioned in the premier issue (July, 1986) that you need a complete list of the Z80's codes in the beginning. Many books on Z80 programming include one of these as an appendix. Also, we have the alphabetic and numeric lists (EZ #101 and EZ #102) in our product list for \$1.95 each (including postage or shipping).

The Z80 code lists, however, are essentially for reference. The alphabetic list is useful for encoding Z80 programs. And, the numeric list is useful for decoding Z80 routines.

This month we're starting a detailed explanation of the Z80 functions. You should note that the Z80 codes are the same for any computer that uses this particular chip. The EOS, on the other hand, is designed specifically for ADAM.

The Fundamentals of Z80 programming

Over the past few months you have no doubt garnered some facts regarding Z80 programming even though we didn't go into a lot of detail on how to develop routines or algorithms. Indeed, it is not necessary to understand the role of each Z80 instruction in order to begin programming.

Let's recapitulate from the premier issue. The Z80 includes eight general purpose registers: A, F, B, C, D, E, H, and L. These 8-bit registers can be combined in pairs to form four 16-bit registers: AF, BC, DE, and HL.

The Z80 also includes a 16-bit stack pointer, the SP register. It includes a 16-bit program counter, the PC register. It also has two 16-bit index registers, IX and IY. And, it includes the 8-bit interrupt register (I) and the 8-bit refresh register (R).

The A register is commonly referred to as the accumulator. The F register is usually called the flags register. The accumulator and the HL register pair are more versatile than the others. That is, a great number of instructions are centered around these two.

For the sake of classification the Z80 operation codes (op codes) are generally grouped into five categories. These are:

- 1 - transfer data
- 2 - process data
- 3 - test datum and branch program execution
- 4 - input/output data
- 5 - Z80 controls

These categories are numbered by frequency of use in general applications. Data transfers are typically used more often than the others.

As you may already be aware, the BASIC interpreter is merely a collection of centrally organized machine code routines. Each BASIC command simply calls a specific routine. BASIC is often described as 'sloppy' by machine code programmers because it doesn't require that you know specifically where data is stored in RAM.

With machine code you MUST know exactly where every piece of information is located. This requires a comparably more intense effort at program organization than BASIC does. Generally a machine code program consists of five major components. These are:

- 1 - the program routines
- 2 - the operating system
- 3 - the stack
- 4 - the program's data tables
- 5 - the user/input data tables

Assembly Language Mnemonics

It is generally easier to think of machine code in terms of mnemonics rather than as a collection of numbers. Below is a small list of assembly language mnemonics and their corresponding functions.

ADC and **ADD** describe addition functions.

BIT operation codes test a specified bit of a particular register's contents or the value at a certain RAM address.

CALL performs the same function as its BASIC equivalent. It executes a machine code subroutine.

CP stands for **COMPARE**. Here, a specified register or value is compared with the current value in the accumulator.

DEC decrements the value in a specified register or address by one.

INC increments the value in a specified register or address by one.

JP stands for **Jump**. The BASIC equivalent is **GOTO**.

JR stands for **Jump Relative**. This causes the program to jump forward or backward a specified number of bytes.

LD stands for **Load**. This function is used to transfer data.

POP retrieves the value of a register pair from the stack.

PUSH puts the value of a register pair onto the stack.

RES stands for **RESet**. It is used to reset (make logical zero) the value of a specified bit of a particular byte.

RET stands for **RETurn**. This function is used to continue normal program execution after a **CALL**.

SET is used to set (make logical one) the value of a specified bit of a particular byte.

Transferring Data

Data transfers can be made between two registers or between a register and a RAM address. There are essentially three classes of data transfers between a register and RAM: single byte transfers (an 8-bit value), double byte transfers (a 16-bit value), and multi-byte transfers.

There are single codes for almost every conceivable transfer between registers. The transfers between registers and RAM are a little more complex. Suppose you want to **POKE** an 18 into address 65535. There are two typical ways of accomplishing this.

```
LD  A, $12
LD  ($FFFF), A
```

or,

```
LD  HL, $FFFF
LD  (HL), $12
```

The machine code equivalents are:

```
62, 18
50, 255, 255
```

and,

```
33, 255, 255
54, 18
```

Of course, the machine code equivalent of the BASIC **POKE** command is a very simple algorithm. But, large, powerful algorithms are constructed from several simple routines, such as this.

You should note the use of parentheses. This denotes indirect addressing in the mnemonics. This technique allows you to use a register pair as an address. For example, **LD (HL), \$12** means load the hex value 12 into the address pointed to by the HL register pair. Thus, it assumes that a 16-bit value has been assigned to the HL pair to designate a RAM address in high and low order bytes.

Next month we'll delve a little more deeply into this vital aspect of machine code programming.

ADAM's Operating System

As we've discussed in earlier issues, the EOS (Elementary Operating System) is a collection of machine code routines. These routines are used for reading from and writing to storage media, reading the keyboard, reading the game controllers, accessing the video chip, etc.

One popular question among beginning hackers is 'how does the EOS get into RAM?'. The EOS is on a ROM chip. When you pull the computer reset switch, it is automatically bank switched (an advanced data transfer technique) into its standard RAM location.

Because ADAM is a game system converted to a personal computer, it is a little more complicated than most home computers. ADAM includes the 16K video chip, the TMS9918A. It includes the sound chip, the SN76489A. And it includes several ROM chips: the game operating system, the EOS, cartridge ROM, and SmartWriter ROM.

The CPU or central processing unit is the brain of any computer. ADAM's principal CPU is the Z80A. The 'A' suffix means that it uses a 4 MHz clock. The standard Z80 chip uses a 2.5 MHz clock; thus, ADAM's chip is almost twice as fast.

In addition to the Z80A, ADAM uses five more microprocessors. Each of these is a 6801. One of these is assigned for both tape drives. One is assigned for both disk drives. One is assigned for the ADAM printer. And, one is assigned to read the keyboard. The final 6801 acts as a liaison between the Z80A and the other four 6801's.

The EOS has routines to access each of these chips. It stores information for each of the 6801's in areas of the EOS called device communication blocks. This makes the EOS rather difficult to study. But, with the various microprocessors ADAM can do several things at one time. For example, ADAM can write data to the disk, read data from a data pack, accept keyboard input, and print a page on the printer ALL AT THE SAME TIME. This process is called multi-tasking.

The EOS is segregated into seven distinct parts. The function of each of these areas are as follows.

54272 - 57343
read/write storage media RAM

57344 - 64511
EOS routines

64512 - 64559
EOS data table

64560 - 64862
EOS jump table

64863 - 65215
central EOS RAM

65216 - 65219
processor control data

65220 - 65535
device communication blocks

The EOS routines are generally accessed via the jump table. This table consists of 101 3-byte jump vectors. The format is: 195, low order byte, high order byte. Jump tables are a common technique used with machine code programs. The idea is that even though the routine's address may be changed by a programmer, it can still be called at the same address in the jump table.

We've distinguished the jump table address from the actual (or absolute) address for each main EOS routine in the table that occupies the next two pages (pages 14 and 15). This makes it easier to study the EOS. Next month we'll disassemble the reset system function.

You can print your own hardcopy of the EOS jump table with the BASIC program that starts on page 17 (it continues to the top of page 18). The absolute address (the address that the jump table vector jumps to) is where the actual function is stored in RAM.

Some of the routine descriptions are very similar. However, there are subtle differences within the code for these routines. Also, the routine descriptions that begin with 'start' make use of ADAM's multi-tasking ability.

ADAM's EOS Jump Table

- page one -

<u>EOS FUNCTION</u>	<u>JUMP TABLE ADDRESS</u>	<u>ABSOLUTE ADDRESS</u>
1: reset system	64560 (40,252)	63538 (50,248)
2: display character (no execute)	64563 (51,252)	63015 (39,246)
3: initialize display character	64566 (54,252)	62940 (220,245)
4: display character (with execute)	64569 (57,252)	62986 (10,246)
5: programmed delay	64572 (60,252)	63839 (95,249)
6: check printer status	64575 (63,252)	62904 (184,245)
7: check printer status	64578 (66,252)	62044 (124,245)
8: check if read device block done	64581 (69,252)	64226 (226,250)
9: check if read keyboard done	64584 (72,252)	64421 (165,251)
10: get keyboard status	64587 (75,252)	62688 (224,244)
11: check status after write device	64590 (78,252)	64283 (27,251)
12: check write status of printer	64593 (81,252)	64481 (225,251)
13: find device control block	64596 (84,252)	62534 (70,244)
14: find device control block	64599 (87,252)	62534 (70,244)
15: find processor control block	64602 (90,252)	64076 (76,250)
16: reset all devices	64605 (93,252)	63734 (246,248)
17: reset ADAMNet	64608 (96,252)	63819 (75,249)
18: send string to printer	64611 (99,252)	62741 (21,245)
19: send character to printer	64614 (102,252)	62716 (252,244)
20: read block from device	64617 (105,252)	64158 (158,250)
21: read keyboard for character	64620 (108,252)	62650 (186,244)
22: read keyboard return code	64623 (111,252)	64123 (123,250)
23: read printer return code	64626 (114,252)	64127 (127,250)
24: read device return code	64629 (117,252)	64135 (135,250)
25: read tape return code	64632 (120,252)	64131 (131,250)
26: relocate processor control block	64635 (123,252)	64047 (47,250)
27: specified status request	64638 (126,252)	62579 (115,244)
28: keyboard status request	64641 (129,252)	62667 (203,244)
29: printer status request	64644 (132,252)	62930 (210,245)
30: tape status request	64647 (135,252)	62935 (215,245)
31: scan active devices	64650 (138,252)	63947 (203,249)
32: initialize input/output processor	64653 (141,252)	63778 (34,249)
33: reset specified device	64656 (144,252)	64093 (93,250)
34: reset keyboard	64659 (147,252)	64081 (81,250)
35: reset printer	64662 (150,252)	64085 (85,250)
36: reset tape	64665 (153,252)	64089 (89,250)
37: start print string	64668 (156,252)	62040 (128,245)
38: start print character	64671 (159,252)	62029 (109,245)
39: start read device block	64674 (162,252)	64198 (198,250)
40: start read device byte	64677 (165,252)	64390 (134,251)
41: start keyboard read	64680 (168,252)	62672 (208,244)
42: start write device block	64683 (171,252)	64255 (255,250)
43: start write device byte	64686 (174,252)	64450 (194,251)
44: synchronize clocks	64689 (177,252)	63856 (112,249)
45: write device block	64692 (180,252)	64178 (178,250)
46: write device byte	64695 (183,252)	64373 (117,251)
47: initialize file manager	64698 (186,252)	61162 (234,238)
48: initialize device directory	64701 (189,252)	62243 (35,243)
49: open file	64704 (192,252)	59904 (0,234)
50: close file	64707 (195,252)	60164 (4,235)
51: reset file	64710 (198,252)	60268 (108,235)

ADAM's EOS Jump Table

- page two -

<u>EOS FUNCTION</u>	<u>JUMP TABLE ADDRESS</u>	<u>ABSOLUTE ADDRESS</u>
52: create directory entry	64713 (201,252)	59024 (144,230)
53: find directory entry	64716 (204,252)	58907 (27,230)
54: alter directory entry	64719 (207,252)	58961 (81,230)
55: read data from file	64722 (210,252)	60439 (23,236)
56: write data to file	64725 (213,252)	60815 (143,237)
57: set data	64728 (216,252)	61125 (197,238)
58: get date	64731 (219,252)	61140 (212,238)
59: rename file	64734 (222,252)	61711 (15,241)
60: delete file	64737 (225,252)	61774 (70,241)
61: read device dependent status	64740 (228,252)	62600 (136,244)
62: jump to word processor	64743 (231,252)	64148 (148,250)
63: execute Z80 return code	64746 (234,252)	64157 (157,250)
64: trim file size	64749 (237,252)	62017 (65,242)
65: check file control block	64752 (240,252)	61577 (137,240)
66: read device block	64755 (243,252)	61819 (123,241)
67: write device block	64758 (246,252)	61926 (230,241)
68: check open mode	64761 (249,252)	61657 (217,240)
69: search for directory entry	64764 (252,252)	61195 (11,239)
70: locate directory entry	64767 (255,252)	58904 (24,230)
71: not implemented	64770 (2,253)	62530 (66,244)
72: not implemented	64773 (5,253)	62530 (66,244)
73: not implemented	64776 (8,253)	62530 (66,244)
74: not implemented	64779 (11,253)	62530 (66,244)
75: not implemented	64782 (14,253)	62530 (66,244)
76: get in/out ports	64785 (17,253)	57745 (145,225)
77: bank switch memory	64788 (20,253)	57733 (133,225)
78: copy ASCII within VRAM	64791 (23,253)	57683 (83,225)
79: write table to VRAM	64794 (26,253)	57344 (0,224)
80: read table from VRAM	64797 (29,253)	57370 (26,224)
81: send value to VDP register	64800 (32,253)	57396 (52,224)
82: read value from VDP register	64803 (35,253)	57423 (79,224)
83: fill VRAM (repeat character)	64806 (38,253)	57433 (89,224)
84: initialize VRAM table (auto calc)	64809 (41,253)	57446 (102,224)
85: write table to VRAM	64812 (44,253)	57545 (201,224)
86: read table from VRAM	64815 (47,253)	57551 (207,224)
87: calculate pattern offset	64818 (50,253)	57610 (10,225)
88: calculate pattern position	64821 (53,253)	57641 (41,225)
89: transfer ASCII from ROM to VRAM	64824 (56,253)	57673 (73,225)
90: write sprite attribute table	64827 (59,253)	57797 (197,225)
91: read game controllers	64830 (62,253)	57939 (83,226)
92: update spinner	64833 (65,253)	58020 (164,226)
93: decrement low nibble	64836 (68,253)	58197 (85,227)
94: decrement high nibble	64839 (71,253)	58207 (95,227)
95: transpose high and low nibble	64842 (74,253)	58217 (105,227)
96: add accumulator to HL pair	64845 (77,253)	58228 (116,227)
97: initialize sound table	64848 (80,253)	58283 (171,227)
98: turn all sound off	64851 (83,253)	58321 (209,227)
99: start tune by number	64854 (86,253)	58343 (231,227)
100: send note to sound chip	64857 (89,253)	58374 (6,228)
101: end tune	64860 (92,253)	58552 (184,228)

Transferring SmartBASIC

Have you ever wanted to transfer SmartBASIC to a medium without destroying the files already on the medium? Several of our readers have asked how to do this.

The program that starts on page 19 (continues to the top of page 20) will accomplish this for you. It takes SmartBASIC from a source medium and puts it on a destination medium without disturbing the files already on the destination medium.

For disk users this is merely a convenience. But data pack users will also find it to be a great time saver. When you store programs on a SmartBASIC backup, it takes quite some time to get to the directory because SmartBASIC occupies the first 28 blocks of user file space. If you store the files first and then transfer SmartBASIC to the medium, directory access time is much faster (on data pack).

Our program consists of several modules. They are:

1. make certain SmartBASIC is on the source
2. make certain BASIC is not on the dest.
3. create directory entry on destination
4. transfer block zero from source to dest.
5. update destination directory
6. transfer BASIC by blocks

Line numbers 420 through 460 ascertain that the BASICPGM file is on the source medium. Line numbers 470 through 472 make certain that the file is 28K in length. This is to prevent an error in the event that you've INIT protected the medium as described in our September issue.

Line numbers 474 through 500 ascertain that the destination medium doesn't already contain the BASICPGM file. Line numbers 600 through 640 transfer block zero, the bootstrap routine, from the source to the destination.

Line numbers 650 through 820 create the directory entry on the destination medium. If there isn't enough room on the destination to store the BASICPGM file, an error message will be displayed.

Line numbers 830 through 860 transfer the first 14K of the BASICPGM file. Line numbers 865 through 880 transfer the last 14K of the BASICPGM file.

Five EOS routines are used by this program. Most of the media access routines return any error code in the accumulator. Also, the 'Z' flag can be used to detect errors with these routines.

We've covered the block read and write routines in previous issues. The Find Directory Entry routine is described in asmb#27 (at the top of page 21). This routine transfers the file's 26-byte directory entry to the designated buffer.

The Alter Directory Entry routine is described by asmb#28. This routine is intended for use in conjunction with the previous routine. The setup is exactly the same.

32 AND 40 Column TEXT Modes

In the August issue we LISTed a program that revealed the video chip's 40 column capability. We mentioned that with a few POKEs you could correct the margins so that this mode would be a viable alternative to the standard 32 column mode.

In "The Hacker's Guide to ADAM: Volume 2" Ben Hinkle reveals these POKEs. Expanding on his research we've created two TEXT modes for SmartBASIC.

The program at the bottom of page 18 is the SmartBASIC V1.0 version. The Intel-BEST 3.3 version is LISTed on the bottom of page 20.

The TEXT command will display the standard 32 column mode. The new 'T' command will create the new 40 column text mode. Thus, you can switch between the two modes in a similar manner to changing between GR and HGR modes.

We have employed an unused area within the BASIC interpreter to store our routines that control these new features. Addresses 11651 through 11775 were allocated for two commands that were never implemented, STORE and RECALL. Our routines fit nicely into this gap.


```
50 REM prints EOS jump table
100 tot = 100: DIM func$(tot)
110 FOR x = 1 TO 30: sp$ = sp$+CHR$(32): NEXT
200 TEXT: PRINT " one moment please ..."
210 FOR x = 0 TO tot: READ func$(x): NEXT
220 HOME: INVERSE: PRINT " EOS jump table ": NORMAL: VTAB 4
230 PRINT " Which option?": PRINT: PRINT
240 PRINT " 1 = print page one": PRINT " 2 = print page two"
250 PRINT " 3 = exit program"
260 GET key$: k% = VAL(key$): IF k% < 1 OR k% > 3 GOTO 260
270 ON k% GOTO 1000,2000,280
280 TEXT: PRINT " program terminated": END
1000 GOSUB 30000: PR #1: PRINT
1010 PRINT "ADAM's EOS Jump Table";SPC(20);"page one": GOSUB 30100
1020 FOR x = 0 TO 50: y = x*3+64560: GOTO 30200
2000 GOSUB 30000: PR #1: PRINT
2010 PRINT "ADAM's EOS Jump Table";SPC(20);"page two": GOSUB 30100
2020 FOR x = 51 TO tot: y = x*3+64560: GOTO 30200
20000 DATA reset system,display character (no execute)
20005 DATA initialize display character,display character (with execute)
20010 DATA programmed delay,check printer status
20015 DATA check printer status,check if read device block done
20020 DATA check if read keyboard done,get keyboard status
20025 DATA check status after write device,check write status of printer
20030 DATA find device control block,find device control block
20035 DATA find processor control block,reset all devices
20040 DATA reset ADAMNet,send string to printer
20045 DATA send character to printer,read block from device
20050 DATA read keyboard for character,read keyboard return code
20055 DATA read printer return code,read device return code
20060 DATA read tape return code,relocate processor control block
20065 DATA specified status request,keyboard status request
20070 DATA printer status request,tape status request
20075 DATA scan active devices,initialize input/output processor
20080 DATA reset specified device,reset keyboard
20085 DATA reset printer,reset tape
20090 DATA start print string,start print character
20095 DATA start read device block,start read device byte
20100 DATA start keyboard read,start write device block
20105 DATA start write device byte,synchronize clocks
20110 DATA write device block,write device byte
20115 DATA initialize file manager,initialize device directory
20120 DATA open file,close file,reset file,create directory entry
20130 DATA find directory entry,alter directory entry
20135 DATA read data from file,write data to file
20200 DATA set data,get date,rename file,delete file
20210 DATA read device dependent status,jump to word processor
20215 DATA execute Z80 return code,trim file size
20220 DATA check file control block,read device block
20225 DATA write device block,check open mode
20230 DATA search for directory entry,locate directory entry
20235 DATA not implemented,not implemented
20240 DATA not implemented,not implemented
20245 DATA not implemented,get in/out ports
20250 DATA bank switch memory,copy ASCII within VRAM
20255 DATA write table to VRAM,read table from VRAM
```

```

20260 DATA send value to VDP register,read value from VDP register
20265 DATA fill VRAM (repeat character),initialize VRAM table (auto calc)
20270 DATA write table to VRAM,read table from VRAM
20275 DATA calculate pattern offset,calculate pattern position
20280 DATA transfer ASCII from ROM to VRAM,write sprite attribute table
20285 DATA read game controllers,update spinner
20290 DATA decrement low nibble,decrement high nibble
20295 DATA transpose high and low nibble,add accumulator to HL pair
20300 DATA initialize sound table,turn all sound off
20305 DATA start tune by number,send note to sound chip
20310 DATA end tune
30000 HOME: PRINT " insert paper in printer": PRINT
30010 PRINT " press [RETURN] to print ..."
30020 GET go$: ON go$ (<> CHR$(13)) GOTO 200: RETURN
30100 PRINT: PRINT SPC(5);"EOS FUNCTION";
30110 PRINT SPC(24);"JUMP TABLE ADDRESS";
30120 PRINT SPC(4);"ABSOLUTE ADDRESS": PRINT: RETURN
30200 ja% = LEN(func$(x)): jb% = 36-ja%
30210 hi% = y/256: lo% = y-hi%*256: hi$ = STR$(hi%): lo$ = STR$(lo%)
30220 IF LEN(lo$) = 3 GOTO 30240
30230 lo$ = LEFT$(sp$,3-LEN(lo$))+lo$
30240 p1 = PEEK(y+1): p2 = PEEK(y+2): p3 = p1+p2*256
30250 p2$ = STR$(p2): p1$ = STR$(p1)
30260 IF LEN(p1$) = 3 GOTO 30280
30270 p1$ = LEFT$(sp$,3-LEN(p1$))+p1$
30280 nu$ = STR$(x+1): PRINT SPC(3-LEN(nu$));nu$;": ";
30290 PRINT func$(x);SPC(jb%);STR$(y);SPC(1);
30300 PRINT "(";lo$;",";hi$;")";SPC(7);
30310 PRINT STR$(p3);SPC(1); "(";p1$;",";p2$;")"
30320 NEXT x: PR #0: GOTO 220

```

```

1 LOMEM :28000
50 REM *** for SmartBASIC V1.0 ONLY !!! ***
100 REM *** PatchWORK ***
110 REM >>> simple BASIC enhancements and fixes
6999 REM TEXT=32 column::T=40 column
7000 DATA 17985,18036,18098,18162,18174,18188
7005 DATA 18210,18234,18401,18410,18427
7010 FOR x = 1 TO 11: READ adr: POKE adr,131: POKE adr+1,45: NEXT
7020 POKE 17166,192: POKE 17177,192: POKE 17059,23: POKE 17115,27
7030 DATA 41,197,229,41,41,193,50,112,66,183,32,1,9,193,201
7040 FOR x = 0 TO 14: READ m1: POKE x+11692,m1: NEXT
7050 DATA 245,229,62,30,50,47,67,62,224,50,63,67,62,32,50,68,70
7060 DATA 62,41,50,96,71,33,41,41,34,97,71,225,241,195,57,43
7070 FOR x = 0 TO 32: READ m1: POKE x+11707,m1: NEXT
7080 DATA 245,229,62,39,50,47,67,62,240,50,63,67,62,40,50,68,70
7090 DATA 62,205,50,96,71,33,172,45,34,97,71,225,241,195,57,43
7100 FOR x = 0 TO 32: READ m1: POKE x+11740,m1: NEXT
7500 POKE 6421+2*41,187: POKE 6422+2*41,45: REM 187+(45*256)=11707
7510 POKE 813,208: POKE 816,84
7520 POKE 6421+2*66,220: POKE 6422+2*66,45: REM 220+(45*256)=11740

```

```
50 LOMEM :45000: POKE 16149,255: POKE 16150,255
100 TEXT: PRINT " BASIC file copier"
110 VTAB 4: PRINT " This program will transfer "
120 PRINT " SmartBASIC from a SOURCE"
130 PRINT " medium to a DESTINATION"
140 PRINT " medium without disturbing any"
150 PRINT " files on the destination"
160 PRINT " medium."
170 PRINT: PRINT " Use with extreme care!!!"
200 VTAB 16: PRINT " Which option?": PRINT
210 PRINT " 1 = copy SmartBASIC V1.0"
220 PRINT " 2 = exit program"
230 GET key$: key% = VAL(key$)
240 IF key% = 1 GOTO 260
250 TEXT: PRINT "program terminated.": END
260 DATA tape one,tape two,disk one,disk two
270 FOR x = 1 TO 4: READ md$(x): NEXT
280 k$ = "BASICPGM"+CHR$(2)+CHR$(3)
285 FOR x = 1 TO LEN(k$): POKE x+29695,ASC(MID$(k$,x,1)): NEXT
290 DATA 62,0,17,0,116,33,160,253,205,204,252,50,255,255,201
295 FOR x = 29706 TO 29720: READ m1: POKE x,m1: NEXT
300 HOME: PRINT " Which SOURCE drive?"
305 GOSUB 900: ds% = dn%: d1 = key%
310 VTAB 10: PRINT " Which DESTINATION drive?": PRINT: GOSUB 900
320 dd% = dn%: d2 = key%: HOME
322 IF dd% <> ds% GOTO 330
324 TEXT: PRINT " you must use different"
326 PRINT " drives for the source and the"
328 PRINT " destination!!!": END
330 PRINT " insert medium which now"
340 PRINT " contains SmartBASIC into"
350 PRINT " ";md$(d1): VTAB 8
360 PRINT " insert medium that you want"
370 PRINT " to transfer SmartBASIC to"
380 PRINT " in ";md$(d2)
390 VTAB 16: PRINT " press [return] to continue..."
400 GET key$: ON key$ = CHR$(13) GOTO 420: GOTO 250
420 HOME: PRINT " one moment please ..."
430 POKE 29707,ds%: CALL 29706
440 IF NOT PEEK(65535) GOTO 470
450 TEXT: PRINT " SmartBASIC not found on"
460 PRINT " ";md$(d1);"!!!": END
470 IF PEEK(64947) = 28 GOTO 474
472 TEXT: PRINT " BASICPGM file not 28KB on": GOTO 460
474 s1 = PEEK(64941): POKE 29707,dd%: CALL 29706
480 IF PEEK(65535) GOTO 600
490 TEXT: PRINT " ";md$(d2);" already"
500 PRINT " contains the BASICPGM file!!!": END
600 HOME: PRINT " transferring SmartBASIC ..."
610 DATA 62,0,1,0,0,17,0,0,33,0,100,205,243,252,201
620 FOR x = 29721 TO 29735: READ m1: POKE x,m1: NEXT
630 POKE 29722,ds%: CALL 29721
640 POKE 29722,dd%: POKE 29733,246: CALL 29721
650 DATA 62,0,33,0,116,17,0,100,1,0,0,205,201,252,50,255,255,201
```

```

660 FOR x = 29736 TO 29753: READ m1: POKE x,m1: NEXT
670 POKE 29737,dd%: CALL 29736
680 IF NOT PEEK(65535) GOTO 800
690 HOME: PRINT " can NOT store BASICPGM file"
700 PRINT " on ";md$(d2);"!!!": END
800 POKE 29707,dd%: CALL 29706: s2 = PEEK(64941)
810 POKE 64947,28: POKE 64950,4
820 POKE 29715,207: CALL 29706
830 FOR x = 0 TO 13: POKE 29722,ds%: POKE 29727,s1+x
840 POKE 29731,117+x*4: POKE 29733,243: CALL 29721: NEXT
850 FOR x = 0 TO 13: POKE 29722,dd%: POKE 29727,s2+x
860 POKE 29731,117+x*4: POKE 29733,246: CALL 29721: NEXT
865 FOR x = 14 TO 27: POKE 29722,ds%: POKE 29727,s1+x
870 POKE 29731,117+(x-14)*4: POKE 29733,243: CALL 29721: NEXT
875 FOR x = 14 TO 27: POKE 29722,dd%: POKE 29727,s2+x
880 POKE 29731,117+(x-14)*4: POKE 29733,246: CALL 29721: NEXT
890 TEXT: PRINT " SmartBASIC transferred.": LOMEM :27600: END
900 PRINT: PRINT: FOR x = 1 TO 4
910 PRINT " ";x;" = ";md$(x): NEXT
920 GET key$: key% = VAL(key$): PRINT key%
930 IF key% = 1 THEN dn% = 8: RETURN
940 IF key% = 2 THEN dn% = 24: RETURN
950 IF key% = 3 THEN dn% = 4: RETURN
960 IF key% = 4 THEN dn% = 5: RETURN
970 POP: GOTO 250

```

```

1 LOMEM :28000
50 REM *** for Intel-BEST 3.3 ONLY !!! ***
60 REM >>> execute Intel-BEST first <<<
100 REM *** PatchWORK 3.3 ***
110 REM >>> simple BASIC enhancements and fixes
6999 REM TEXT=32 column:::T=40 column
7000 DATA 17985,18036,18098,18162,18174,18188
7005 DATA 18210,18234,18401,18410,18427
7010 FOR x = 1 TO 11: READ adr: POKE adr,131: POKE adr+1,45: NEXT
7020 POKE 17166,192: POKE 17177,192: POKE 17059,23: POKE 17115,27
7030 DATA 41,197,229,41,41,193,58,112,66,183,32,1,9,193,201
7040 FOR x = 0 TO 14: READ m1: POKE x+11692,m1: NEXT
7050 DATA 245,229,62,30,50,47,67,62,224,50,63,67,62,32,50,68,70
7060 DATA 62,41,50,96,71,33,41,41,34,97,71,225,241,195,57,43
7070 FOR x = 0 TO 32: READ m1: POKE x+11707,m1: NEXT
7080 DATA 245,229,62,39,50,47,67,62,240,50,63,67,62,40,50,68,70
7090 DATA 62,205,50,96,71,33,172,45,34,97,71,225,241,195,57,43
7100 FOR x = 0 TO 32: READ m1: POKE x+11740,m1: NEXT
7500 POKE 65378+2*41,187: POKE 65379+2*41,45: REM 187+(45*256)=11707
7510 POKE 812,66
7520 POKE 65378+2*66,220: POKE 65379+2*66,45: REM 220+(45*256)=11740

```

TITLE (asmb#27):
Find Directory Entry

<u>Line#:</u>	<u>Label:</u>	<u>Decimal value:</u>	<u>Op Code:</u>	<u>Comment:</u>
1	SETUP	62, nn,	LD A, nn	; load drive code
2		17, 0, 116,	LD DE, \$7400	; load address of filename
3		33, 160, 253,	LD HL, \$FCA0	; load address to put file entry
4		205, 204, 252,	CALL \$FCCC	; execute EOS function
5	ERROR	50, 255, 255,	LD (\$FFFF), A	; store error code at 65535
6	DONE	201	RET	; RETURN from routine

TITLE (asmb#28):
Alter Directory Entry

<u>Line#:</u>	<u>Label:</u>	<u>Decimal value:</u>	<u>Op Code:</u>	<u>Comment:</u>
1	SETUP	62, nn,	LD A, nn	; load drive code
2		17, 0, 116,	LD DE, \$7400	; load address of filename
3		33, 160, 253,	LD HL, \$FCA0	; load address of new file data
4		205, 207, 252,	CALL \$FCCF	; execute EOS function
5	ERROR	50, 255, 255,	LD (\$FFFF), A	; store error code at 65535
6	DONE	201	RET	; RETURN from routine

HACKER'S CONTEST #5

The NIBBLES & BITS Hacker's Contest is a bi-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 contest will be considered valid if, and only if, they are postmarked after November 30, 1986 and prior to February 1, 1987. The winner shall be announced in the February issue of NIBBLES & BITS.

Write a SmartBASIC program (it may include machine code in DATA statements) which will transfer SmartWriter to a self-booting disk or data pack.

- continued from page 16 -

Addresses 11651 through 11691 are reserved for a screen line buffer. The buffer must be 41 bytes in length for the CNTL-P function.

Addresses 11692 through 11706 are used to extend the print screen line routine for the 'T' (40 column) mode. Addresses 11707 through 11739 set up the various values for the TEXT (32 column) command. Addresses 11740 through 1172 set up the various values for the 'T' (40 column) command.

You can CALL 11707 to execute the 32 column mode. And, you can CALL 11740 to execute the 40 column mode. Line numbers 7500 through 7520 change the execution vectors for each command. The 'T' command replaces the '&', ampersand, command. This was a little known shortcut for the REM command.

Eleven addresses in SmartBASIC point to the screen line buffer. Previously this was at address 16960; we changed it to address 11651, 131 and 45 in low and high order bytes. Line numbers 7000 through 7010 take care of these changes.



DID YOU KNOW?

*** On 4 November 1952 the UNIVAC I correctly predicted election results.

*** On 18 November 1963 the first computer designed auto part was manufactured.

*** On 29 November 1972 the Atari game 'Pong' first appeared in arcades.

*** On 10 December 1815 Augusta Ada Byron was born. She programmed Charles Babbage's analytical engine in the 1830s. The Ada computer language, developed and used by the Department of Defense, is named after her.

*** On 26 December 1791 Charles Babbage was born.

*** On 28 December 1903 John von Neumann was born. He developed the 'stored program concept' used by all computers today.

GETTING INTO CP/M 2.2

The Built-in Commands

(part 3)

The TYPE Command:

The TYPE command can be used to display any ASCII file on the screen. The format is:

TYPE A:filename

The drive suffix is optional. Also, most text (or ASCII) filenames end with one of three extensions: DOC, TXT, or PRN.

You can also get a hardcopy of a text file on the ADAM printer. To do so, just press CNTL-P before entering the TYPE command. You can turn the printer off by pressing CNTL-P a second time.

The ERA Command:

The ERA command is used to erase files from the directory. It works almost the same as SmartBASIC's DELETE command. The format is:

ERA A:filename

ERA can be a dangerous command. There is a command in the public domain which will unerase (UNERA) a filename, but you should still use ERA with extreme care.

The SAVE Command:

The SAVE Command is used to store information from the transient program area (TPA) or user RAM to disk or data pack. This built-in command is rather complex. We'll go into more detail on it in a later issue.

ADAM PRODUCT REVIEWS

PRODUCT:	MultiWRITE
MANUFACTURER:	Strategic Software
MEDIA TYPE:	DDP
GRAPHICS/SOUND/DESIGN:	93/no sound/93
INSTRUCTIONS:	50
USEFULLNESS vs PRICE:	93
RECOMMENDATION:	highly recommended
PRICE:	\$36.00
RATED BY:	Jim Guenzel
	12221 Spring Shadow Court Maryland Heights, MO 63043

MultiWRITE is a complete word processing package for the ADAM. The package allows one to compose, edit, print and even convert SmartWriter files to MultiWRITE format for use by MultiWRITE.

The main feature of the program is a full 64 column display on a standard TV screen. This feature works! (Of course a sharp image on the TV or monitor you use will make the text more readable. But I get along just fine with a 7 year old 19 inch G.E. color TV.) The 64 column display is achieved by a smaller font set and by using the HGR2 mode during actual text entry and edit functions. During text entry and edit the screen will display 64 columns by 21 lines clude: underline, right and left justify, search, move, copy, quick to the top of the file (but no quick to the end of the file). Across the top of the screen is a 'ruler' with each tab marked by a '!'. Default settings are every 5 spaces and can be changed with 'CONTROL T'.

MultiWRITE is part BASIC programs and part machine language programs. The DDP includes multiwrite (the main program, 10K), PRINTFILE (the print program, 4K), convert (a program to convert SmartWriter files to MultiWrite format, 1K), and M.CODE (5K of machine code -- this includes the new fonts and routines which make the text entry and edit parts of the program work).

Documented text entry and edit functions include: underline, right and left justify, search, move, copy, quick to the top of the file (but no quick to the end of the file), center, forced page breaks, the ability to change line spacing and margins at different places in the text, and an insert feature. Just remember to do a 'CONTROL F' after insert procedures if the new line goes past the right margin or you will find that on printing, the inserted material will not print. A blank line will be there instead.

MultiWRITE loads from BASIC. You enter 'run multiwrite' or you can copy all the program to a disk or DDP with BASIC on it and then rename multiwrite to HELLO and multiwrite will auto-load when you boot BASIC. The program is menu driven and 'user friendly'. Using the program is as easy as using SmartWriter but without the SmartKEYs. Sometimes you will hit a special function key (like 'TAB') and nothing will happen. Press 'SHIFT LOCK' if this happens. Some functions will not work in the 'UPPER CASE' mode.

A nice feature during entry and edit functions is cursor control: the cursor can be moved all around the screen with the arrow keys. The cursor does 'lag' quite a bit and it will keep moving after you release the key. One way to make it stop where you want it to is to press the opposite arrow key for each place past the intended stopping point that the cursor moves while the cursor is still moving.

When you exit from or quit MultiWRITE or the printer program, the program calls a jump to SmartWRITER. This can be changed. Just change line 2000 in PRINTFILE to HOME:END. And, do the same thing to line number 9900 in the multiwrite program.

Merging files is not an option. MultiWRITE files are saved as 'H' binary files and I have not had time to write a program to merge them yet. SmartWriter will not load binary files. There is no super or subscript function but because of the ability to set 1/2 line spacing, you can achieve super or subscript.

The Operation Manual is only 10 pages long. No technical information is given. The information given is very sparse but it is adequate.

I enjoy using MultiWRITE (I write a number of business reports weekly) mainly because of the 64 column display and the simple edit features. I have four word processor programs for ADAM -- two of them are in CP/M. The CP/M programs are fast and almost 100% bug free and have more features than I can remember (thus I have to keep a manual next to me all the time). MultiWRITE has enough features to be useful and the 'CONTROL' key sequences are easy to remember. MultiWRITE is a great value as it stands. With a few added features and some technical information MultiWRITE would be a 'must buy' for ADAM owners.

PRODUCT:	EBU Package
MANUFACTURER:	Extended Software Company
MEDIA TYPE:	DDP/disk
GRAPHICS/SOUND/DESIGN:	88
INSTRUCTIONS:	88
USEFULLNESS vs PRICE	87
RECOMMENDATION:	recommended
PRICE:	\$21.95
RATED BY:	staff

The Extended BASIC Utilities (EBU) package modifies SmartBASIC to include four additional functions. One function allows you to set TEXT mode screen colors. Another function allows you to play music and create tunes in BASIC. And, it includes a hex to decimal calculator.

The most unique function of the enhancement is the renumber function. It can renumber a 25K program in about 5 seconds.

Of the four or five popular SmartBASIC enhancement packages, EBU has the least features per dollar. But, if you like to renumber your programs, this feature alone could make the package a worthwhile purchase.

ADAMtm SW&HW

by



Marathon Computer Press

Take your pick of up to (two) MCP programs
at a special Discount price!

DISCOUNT SOFTWARE FOR NIBBLES AND BITS SUBSCRIBERS

CODEVISOR 4.1

DISK OR DDP . . . ONLY \$12.50

THE INVESTMENT ANALYST

DISK OR DDP . . . ONLY \$10.00

CopyWriter 1.0 & DriveWriter 1.0

DISK OR DDP . . . ONLY \$9.50

THE SPANISH VOCABULARIAN

DISK OR DDP . . . ONLY \$10.50

THE MEGAUTIL DEVELOPMENT PACKAGE V-1

DISK OR DDP . . . ONLY \$20.00

THESE PRICES ARE BELOW NORMAL USER GROUP PRICES ! !

YOU MAY NOT USE THIS DISCOUNT OFFER WITH ANY OTHER

DISCOUNTS THAT YOU MAY QUALIFY FOR....

SEE NEXT PAGE FOR DETAILS AND INFORMATION ON HOW
TO ORDER SW AT THESE ROCK BOTTOM PRICES !

ORDERING INFORMATION

To receive the discounts mentioned on the previous page, all that is required is that you include your Nibbles & Bits subscription ID number found on your N&B mailing label, and include the following special offer control number.

(NIBB1286AD1)

The discounts offered here are lower than the normal 30% off retail price that we give to Nibbles & Bits subscribers. If you would like to order more than two of our software packages, send for a FREE catalog and simply deduct 30% off of the advertised price. THIS OFFER EXPIRES 1 FEB 1987.

BRIEF PRODUCT DESCRIPTIONS

CODEVISOR 4.1 - A dedicated data security program that allows you to LOCK your DDP or disk to prevent it's access on the ADAM(tm), by normal means. Works with SmartBASIC(tm), SmartLOGO(tm), ADAMCalc(tm), & SmartWriter(tm) files.

THE INVESTMENT ANALYST - An integrated package of programs to aid the Stock Market investor in making informed decisions, and as a utility for following individual stock performance and portfolio growth. Not list protected.

THE SPANISH VOCABULARIAN - A CAI (Computer Aided Instruction) program designed to help the user in mastering the Spanish Vocabulary. Allows you to create tests, take tests, review files, transfer files, print study sheets, and more... One of our most successful programs. It is provided with a 1600 word Database & an in-depth User's Manual. It was reviewed in the October issue of Nibbles & Bits.

CopyWriter 1.0 & DriveWriter 1.0 - An affordable copy utility that packs a powerful punch. A generous 20K copy buffer, Default Drive fixer for SmartBASIC(tm), entertaining graphics screen, SmartKeys, and more... One of the best buys in the ADAM(tm) market today!

THE MEGAUTIL DEVELOPMENT PACKAGE - A collection of utility programs designed to aid the applications programmer in cutting down development time. Includes a Block editor (DEC, BIN, ASCII, HEX), a Directory Expander, a user configured Security program, a copy utility, and a PD module library that can be linked to your own applications with the BASIC MERGE utility. Over 200K of programs and DOC files in this massive package! Even includes a PD Sprite Editor program.

The above items marked with a (tm) are trademarks of Coleco Ind., Inc. MCP is not affiliated with Coleco.

You may call our 24Hr. Hotline for a catalog at (804)460-5227. If you are placing an order, enclose a Check or M.O. for the indicated amount. We pay shipping to all locations within the 50 states. If you are located outside of the United States, please include an additional \$3.00 for shipping. All prices are in U.S. dollar equivalents. ONLY 2 PROGRAMS ALLOWED WITH THIS OFFER!! BEAT THE DEADLINE DATE.....

MARATHON COMPUTER PRESS
P.O. BOX 68503
VIRGINIA BEACH VIRGINIA, 23455-9433

**LOCAL ADAM
USERS GROUPS**

**BULLETIN
BOARD**

CANADA

Derek Townsend
Box 820
Claresholm, Alberta
CANADA T0L 0T0

Winnipeg ADAM Users Group
729 Government Avenue
Winnipeg, Manitoba
CANADA R2K 1X5

Robert Dunstan
95 Harland Crescent
Ajax, Ontario
CANADA L1S 1K2

Metro-Toronto ADAM Users Group
P.O. Box 123
260 Adelaide Street East
Toronto, Ontario
CANADA M5A 1N0

J.A. Girard
1420 Avenue Langevin Sud
Alma, Quebec
CANADA G8B 6B1

Mike Laurier
7350 Roi Rene
Anjou, Quebec
CANADA H1K 3G6

First Canadian ADAM Users Group
P.O. Box 547 Victoria Station
Westmount, Quebec
CANADA H3Z 2Y6

G. Hibbert
P.O. Box 10
Mistatim, Saskatchewan
CANADA S0E 1B0

PRINTING SERVICES - NEWSLETTERS, ETC.
Ted Gocal, Gannon University
121 West 7th Street
Erie, PA 16541
CompuServe ID# 75226,226

ADAM Software
Extended Software Company
11987 Cedar creek Drive
Cincinnati, OH 45240

ADAM newsletter
AUGment
P.O. Box P-C
Lynbrook, NY 11563

ADAM newsletter
Expandable Computer News
Route 2, Box 211 Scrivner Road
Russellville, MO 65074

ADAM Software
D.L. DECKER ENTERPRISES
RD #2, Box 15
Spring Mills, PA 16875-9720

ADAM Software
Practical Programs
Box 244
Kalamazoo, MI 49005-0244

Dust covers for ADAM
J CHECK SOFTWARE
P.O. Box 345
Millry, AL 36550

PRODUCT LIST**DEI HARDWARE SUPPLIES****DEI SOFTWARE****■■■ Intel-BEST 3.3**

dynamic enhancement to SmartBASIC -- makes over 3 dozen changes

\$24.95 STANDARD PRICE

\$18.95 SUBSCRIBER DISCOUNT PRICE

■■■ Intel-LOAD V1.0

converts BASIC 1.0 programs to LOAD up to 12 times faster -- stays in RAM plus onscreen help

\$15.95 STANDARD PRICE

\$11.95 SUBSCRIBER DISCOUNT PRICE

■■■ ShowOFF I

powerful graphics design package -- includes a variety of print options

\$29.95 STANDARD PRICE

\$24.95 SUBSCRIBER DISCOUNT PRICE

■■■ Intel-LOAD V2.0

converts BASIC 2.0 programs to LOAD up to 12 times faster -- stays in RAM plus onscreen help

\$15.95 STANDARD PRICE

\$11.95 SUBSCRIBER DISCOUNT PRICE

■■■ N&B issue programs #1

all the programs from issues 7/86 - 9/86

\$9.95 STANDARD PRICE

\$4.95 SUBSCRIBER DISCOUNT PRICE

■■■ N&B issue programs #2

all the programs from issues 10/86 - 12/86

\$9.95 STANDARD PRICE

\$4.95 SUBSCRIBER DISCOUNT PRICE

■■■ DEI blank disks

Single-sided, double-density, with envelope

\$1.99 (each) or \$19.95 (for 10) STANDARD PRICE

\$1.79 (each) or \$17.95 (for 10) SUBSCRIBER PRICE

■■■ DEI ADAM printer ribbons

just like the ones that came with your ADAM

\$5.50 (each) or \$15.50 (for 3) STANDARD PRICE

\$4.95 (each) or \$13.45 (for 3) SUBSCRIBER PRICE

■■■ Coleco/LDRAN data packs

manufactured by Loranger

\$4.95 (each) or \$39.95 (for 10) STANDARD PRICE

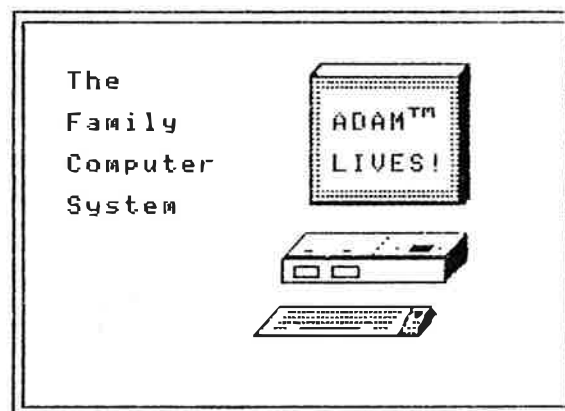
\$3.95 (each) or \$33.95 (for 10) SUBSCRIBER PRICE

■■■ Plain Label data packs

manufactured by E & T Software

\$3.95 (each) or \$39.95 (for 10) STANDARD PRICE

\$3.25 (each) or \$27.95 (for 10) SUBSCRIBER PRICE



DEI PAPER SUPPLIES

■■■ multi-purpose 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
the popular SmartBASIC enhancement

\$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 quizzes (U.S. capitals, world
capitals, elements of chemistry)

\$24.95 STANDARD PRICE
\$19.95 SUBSCRIBER DISCOUNT PRICE

COLECO PRODUCTS

(limited quantities)

■■■ SmartLDGO (DATAPAK ONLY)
Coleco's version of the popular language

\$34.95 STANDARD PRICE
\$27.95 SUBSCRIBER DISCOUNT PRICE

■■■ CP/M 2.2 (DATAPAK ONLY)
version of the popular operating system configured
for ADAM

\$34.95 STANDARD PRICE
\$27.95 SUBSCRIBER DISCOUNT PRICE

■■■ SmartFILER (DISK ONLY)
Coleco's popular general purpose database

\$19.95 STANDARD PRICE
\$14.95 SUBSCRIBER DISCOUNT PRICE

■■■ DISK VERSIONS NOW AVAILABLE. Unless otherwise
noted, all software is available on disk or data
pack.

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

■■■■■ The prices listed above are effective
12/15/86 through 1/31/87

PRODUCT ORDER FORM

Your name _____ Address _____ City _____ State _____ ZIP _____ Phone _____ ID Number _____
--

PRODUCT	QNTY	MEDIA	PRICE

Subtotal: _____
 Shipping: _____ (inside contiguous USA: \$2.50; elsewhere: \$4.00)
 Tax: _____ (WV residents only: 5%)
 Other: _____
 Other: _____ (subscription/renewal)
 Total: _____

- Thank you for your support!!! -

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

DIGITAL EXPRESS, INC.
 Route one, Box 29 - G
 Oak Hill, WV 25901

If you're a new subscriber, please answer these questions:

1. How long have you used an ADAM™ computer?
2. What topics would you like to see discussed in NIBBLES & BITS?
3. Would you like to contribute articles, reviews or programs?
4. Briefly, describe your system . . .

SOFTWARE EXCHANGE

We now have 5 BASIC public domain volumes and 6 Coleco public domain volumes. Each BASIC volume (requires SmartBASIC) includes two instruction files which can be read or printed from SmartWriter. All BASIC programs are speed-RUN. Most of the BASIC programs (except the UTILities volume) are controlled from a RAMDISK written primarily in machine code.

To get a FREE copy of a specific BASIC-volume: (1) contribute an original program, (2) send a signed statement that the program is not copyrighted, (3) send the program on DDP or disk, (4) request the specific volume that you want in return, and (5) include return postage and a mailer or \$2.50 for shipping. Below is the directory for the N&Bgames02 volume.

The BASIC volumes are N&Bgames01, N&Bgames02, N&Butil01, N&Bmath01, N&Bgraph01. The Coleco volumes are Jeopardy, Troll's Tale, SubRoc, Pinball Construction with Hardhat Mac, Pinball Games Set#01, and ADAMLink II. Each volume may be purchased on DDP or disk for ONLY \$5.95.

VOLUME TITLE: N&Bgames02		FREE BLOCKS: 3					
BOOT	:S 1	DIRECTORY	:S 1	HELLO	:A 1	GoHACKER	:H 2
nl.obj	:H 3	HackerDISK	:H 6	Duch	:H 4	BASICPGM	:M 1
FootBall	:H 8	Kutenhannen	:H 8	Committed	:H 12	StarShip	:H 14
scramble	:H 4	HangmanII	:H 4	CivilWar	:H 13	AceyDucey	:H 3
Baccarat	:H 5	Defuse	:H 3	KingQueen	:H 9	Elimin.BIG	:H 12
StrWar.BIG	:H 12	Poker.BIG	:H 16	READ-1.WPR	:H 6	READ-2.WPR	:H 8

SWIFT POLL BALLOT

As a NIBBLES & BITS subscriber, you are invited to submit the following SWIFT POLL ballot. You may submit no more than THREE ballots during the tally period ending December 31, 1986. Valid entries must include your subscription ID number and may be duplicated, if you prefer.

To complete, just list your favorite software title in the categories of your choice. You may list different favorites on each ballot. The results of this particular tally period will be published in the January issue.

YOUR NAME: _____ SUBSCRIPTION ID NUMBER: _____

Your favorite COLECO title: _____

Your favorite Public Domain title: _____

Copyrighted 3rd party titles:

Your favorite media/copy utility: _____

Your favorite game (cart, disk, or DDP): _____

Your favorite BASIC enhancement: _____

Your favorite tutorial book: _____

Your favorite CP/M software: _____

Your favorite educational title: _____

Your favorite miscellaneous utility: _____

Your favorite miscellaneous title: _____

DIGITAL EXPRESS
Route one, box 29-6
Oak Hill, WV 25901

12/88 Issue of NIBBLE & BITS