contact 4

the user group newsletter for dec. 1978 apple computer inc.®

SERIAL INTERFACE CARDS ARE HERE!

Have you ever wanted to connect a fast, letter-quality printer or a serial interface plotter to your Apple? If you have, the high-speed Serial Interface Card is for you.

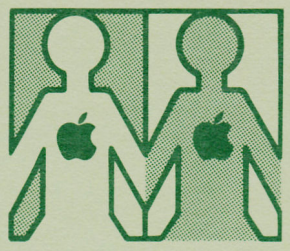
This card, which was first announced in our mid-summer catalog, allows the Apple to communicate with almost any serial interface (RS-232) device; and it offers the following benefits:

- Programmable speed: 75-19, 200 baud
- BASIC control of serial printers, with line feed insertion and automatic formatting of listings
- All control software is on-board ROM – no programs to write
- Switch-selectable preset conditions for speed, line length, auto line feed, and carriage return delay

This new serial card handles many applications that previously required either modifications of the existing Communication Card, or creation of a whole new hardware design. It is the interface to use for any application involving block data movement, hard copy printers, or data transfer speeds other than 110/300 baud.

With the availability of this card, the older Communication Card should be reserved for those applications requiring Apple to act like a terminal to other computers. That card still offers the advantage of full-duplex (simultaneous, bi-directional) communication while the Serial Card does not. (Although it is bi-directional, it cannot transmit and receive concurrently.) But because of its lack of automatic line feed, the Communication Card is not really suited for control of most printers. The Serial Card, on the other hand, not only inserts the line feed characters that most printers need, but it insures that BASIC key words are not broken into separate lines during a listing.

The new card should be available from your Apple dealer as you read this. It is part number A2B0005, with a retail price of $195.



LOCAL USER GROUPS

. . . Here come nine more

Here come nine new APPLE user groups, with two more waiting in the wings:

CALIFORNIA –

Computerland of Los Altos
Sarkis Kouzoujian
4546 El Camino Real
Los Altos, CA 94022
(415) 941-8154

NORTH ORANGE COUNTY COMPUTER CLUB
David Smith
607 North Twilight
Placentia, CA 92670
(714) 993-9939

APPLE-BIZ
Melvin Wong
301 Balboa
San Francisco, CA 94118
(415) 221-8500

APPLE USERS GROUP
Recreational Computer Center
Mark Wozniak
1324 S. Mary
Sunnyvale, CA 94087
(408) 735-7480

DELAWARE –

Computerland of Newark
James H. Higgins
Astro Shopping Center
Kirkwood Highway
Newark, DE 19711
(302) 738-9656

MINNESOTA –

MINI'APP'LES
Dan Buchler
13516 Grand Avenue South
Burnsville, MN 55337
(612) 890-5051

NEW YORK –

NYC USERS GROUP
c/o Computer/Mart of New York
Neil Shapiro
118 Madison Avenue
New York, NY 10016
(212) 686-7923

NORTH CAROLINA –

APPLE CORE
COMPUTER CLUB
Alex Popper
3915 E. Independence Blvd.
Charlotte, NC 28205
(704) 523-5107

OKLAHOMA –

APPLE II USERS GROUP
c/o The Tulsa Computer Society
Jerry Henshaw, President
P. O. Box 1133
Tulsa, OK 74101
(918) 836-7364

OREGON –

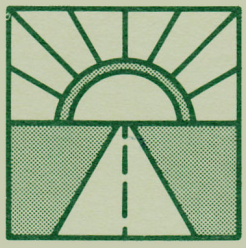
APPLE PORTLAND PROGRAM
LIBRARY EXCHANGE
Ken Hoggatt, President
9195 S.W. Elrose Court
Tigard, OR 97223
(503) 639-5505 (residence)
(503) 644-0161, x6136 (work)

And here are two people interested in forming APPLE user groups in their areas:

Larry Bugbee
2874 Ithaca
BOISE, IDAHO 83705
(208) 362-9132 (residence)
(208) 384-6100 (work)

Charles Kollett
32 N. Brewster Lane
BELLPORT, LI, NY 11713
(516) 286-0198

Finally, we'd like to remind you that if you'd like to form or join an APPLE user's group in your area, call or visit your local APPLE dealer – he'll be glad to help. And if you know of a group that's not yet been listed in this newsletter, tell us about it. Direct your note to Phil Roybal, marketing manager.



LOOKING AHEAD

. . . Some new things from us, for you

We've selected a modem that we think does a great job with APPLE. It's made by Novation, and includes an acoustic coupler. The unit operates in both the full- and half-duplex modes, and runs at both 110 and 300 baud. With our Communications Card and all necessary cables, our Model IIA retails for $495 as APPLE part no. A2M0017; without the card and cables, it costs $315, and carries part no. A2M0021.

And, too, we're making ready the second entry in our Dow-Jones series. Called PORTFOLIO EVALUATOR, the program lets APPLE keep track of your portfolio. You enter the stock names, number of shares, date purchased, and purchase price. APPLE tells you the current price, current portfolio value, long- and short-term capital gains, and the rate of return. The new program works in conjunction with our existing D-J service (STOCK QUOTE REPORTER). PORTFOLIO EVALUATOR will be available in the latter part of this last quarter of 1978.

Last but not least, we now have a new manual out for all you Applesoft users. It is the Applesoft II BASIC Programming Reference Manual: 170 pages of just about everything anyone every wanted to know about the language. From syntax to memory maps to the token structure of the language itself – it's all there! And it's printed in the same 6" x 8.5" spiral-bound format as our popular BASIC Programming Manual. This manual is now available from your Apple dealer at an introductory price of $7.50.

EDITORIAL

by Phil Roybal, Marketing Mgr.

. . . Please don't send money!

In the good old days of personal computing (about 2½ years ago), most business was handled through mail order because there were many small manufacturers, and not enough sales outlets to handle the products. Since then the market has increased dramatically, and manufacturers have searched for a way to service their customer base more efficiently. The solution we chose was to sell through a nationwide network of retail dealers. This gives our products wide exposure and insures that you have a local store to demonstrate, sell, and service the product.

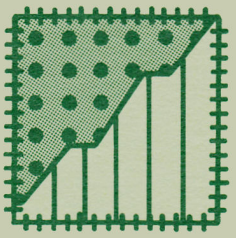
To maintain this dealer network, APPLE sends all retail business through it, and provides a profit margin that suports the field efforts. We divide the country into regions, and set up distributors to create and service the dealer network in each area. We sell to the distributor, the distributor sells to the dealer, and the dealer sells to the public. To keep our sales network healthy, we use it rather than compete with it. This is why we discourage individual orders by mail.

The heavy demand for Apple products has made it difficult to meet the immediate needs of this distributor/dealer network. Many of you have called the factory to inquire about the delivery status

of your order. Unfortunately, we have no visibility into the order level of individual dealers or what position they have in the delivery queue with their distributor. Consequently, we are unable to provide delivery information on your order. To get a delivery quote, ask your dealer where you are on his backorder list, and ask him to contact the distributor for an estimated delivery commitment. We schedule regular shipments to these distributors, and barring any unforseen difficulties, he can provide the information necessary to estimate your delivery date.

For those of you that ordered products in June, July, and August, the demand was very high and did not allow time for our supplies to fill our raw material requirements. Delivery of the orders placed in August, September, and October was delayed by our effort to fill the summer months' needs first. Delivery commitments on new orders are about ten weeks after receipt of order (ARO) presently.

As we start the last quarter of 1978, our business plans are directed at increasing our capacity to the point where we will bring our backlog down to four weeks ARO. Whether we are able to meet our goal will depend upon business conditions, the capacity of our suppliers, and acts of God. However, as you wait for your Disk, Applesoft ROM Card or Apple II System, it should please you to know that the reason you are waiting is because they are fine products in heavy demand. When you get them, you know you will be getting the best!



PATCHES AND PROGRESS

. . . Nobody is perfect

CONTACT 3's ASCII table

In the last issue of CONTACT (No. 3) we presented a table of decimal-number and APPLE-keyboard equivalents to ASCII characters. In this issue of CONTACT we'd like to correct that table.

First, decimal codes 153 and 154 are generated by Y^C and Z^C, respectively.

Second, it seems that we showed the APPLE keyboard as having a lower-case alphabet – which, of course, it hasn't. So. . . in the table's right-most column for ASCII characters a through z and decimal numbers 225 through 252, please delete both the "sa." ("same") near the top of the APPLE KYBD column and the long, downward-pointing arrow below it.

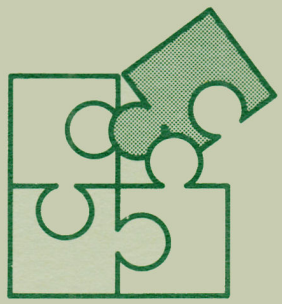
HIRES demo tape typo

Somehow we managed to ship a number of HIRES GRAPHICS demonstration tapes that carry a typographical error on the cassette labels. The error is in the brief loading instructions printed on the label: *800.FFFR. This should read: *C00.FFER. Note that the numeral "8" should be the letter "C". Programs on the tape will run correctly when you follow the proper loading instructions.

Making life easier in APPLESOFT II

If you use APPLESOFT II and enter a statement such as IF X = A THEN PRINT T, APPLE won't do what you want it to do. That's because APPLESOFT II's parsing action causes the statement to be read as if you entered IF X = AT HEN PRINT T; a syntax error will appear and you won't know why.

To prevent this, simply enclose the A in parentheses, which stops APPLESOFT II's parsing of the letter combination. Thus, enter the statement as IF X = (A) THEN PRINT T, and all will be right with the world.



BITS AND PIECES

. . . APPLES and PIAs

Do APPLEs work with Parallel Interface Adapters (PIAs)?

(A final word from EDN magazine)

Not too long back, Boston-based EDN Magazine – a publication widely read in the electronics industry – somehow became convinced that APPLE would not work satisfactorily with peripheral interface adapters. But now . . .

Apple II – no PIA problem

Our August 20 issue reported that Apple Computer, Inc. has supplied us with a working single-PIA interface for a keyboard. We have since checked and rechecked the hardware and software designs described in the article on pgs. 101-110 of our March 20 issue and have been unable to duplicate the Apple II timing problem "uncovered" in our May 20 issue

(pgs. 105-115). This development leads us to two important conclusions:

1. *PIA-based interfaces will work with the Apple II without any special design considerations. Delays or other fixes are not needed. Although the μC uses the θ_0 clock instead of the θ_2 clock, this provision should not prevent PIA operation.*
2. *The source of Indecomp's interface difficulties, while not pinpointed, must lie in human error on the part of EDN's staff.*

(Partially reprinted by permission from EDN Magazine, 9/20/78, pg. 17.)

The colon as a listing formatter for Applesoft

The basic description of the colon's action is that it can be used to separate BASIC statements. But with the colon you can structure your listing in any way you desire, whether you need vertical spacing or horizontal spacing – or both, as in tabulating data into blocks of rows and columns.

To do it, enter the line number, then the colon. If there is no information following the colon on the line, then the display will step vertically. If information does follow the colon, then that information will be spaced to the right of the colon, allowing the interior code of FOR . . . NEXT loops, etc., to be neatly indented.

You can see how the colon is used for this purpose in this issues's *HOW TO* section. Note the neat appearance of the listing.

Apple and Education

Apple Computer, Inc., announces the appointment of Roger Cutler as Education Specialist for the company. Roger is organizing an educators' user group. If you would like to be on his mailing list, please write him at Apple. Our plans include an educational software bank and advice on writing computer assisted instruction programs. Watch CONTACT for further information about educational applications of the Apple II.

Back issues of CONTACT

CONTACT 1, 2, and 3 are out of print, and there are no back issues left. For those of you who missed them, we will be printing The Best of CONTACT, a summary of the most useful information from the 1978 issues. That document should appear on your dealer's shelves in January, 1979.

Missing Here . . .

Our disk examples squeezed out the promised list of PEEKS, POKES, and CALLs from this issue. Look for them in CONTACT 5.

HOW TO . . . Now if only I knew how the disk worked

The final Disk Operating System manual is in the works, and it's going to contain a lot of information. But like all good things, it will take some time. Meanwhile, we have produced some examples that show how to handle the most asked-about situations.

DATA FORMAT

The first thing to understand about the disk is its information format. Data can be written as fixed-length records or random-length records. All data is written in ASCII. Fixed-length records are written when you OPEN the file with an "L" parameter. OPEN DAN, L40 will create a file named DAN, whose records are all 40 bytes long. If you only put 20 bytes of information into each, you will waste 20 bytes per record of disk space.

Random-length records are actually one byte each, but are grouped together into blocks (logical records) which can be up to 32K bytes long. Each logical record ends with a carriage return. Note that this means the number "1" will require 2 bytes (number followed by a RETURN), and the number 10 will require 3 bytes. If you forget this and later replace "1" with "10", you will destroy part of the following record (poor programming practice).

USING RANDOM-LENGTH RECORDS

Example number one illustrates the writing of random-length records. We simply open the file, then start printing. Each PRINT statement creates one logical record, ending with a carriage return. Thus, to put items into separate records, we must print them with separate PRINT statements. Each record is just long enough to hold the data you put in it.

Now run this program, using strings of less than 20 characters. It produces files called FILE and FILE.PTR, which are used in the following examples.

EXAMPLE 1

```
5  REM  THIS PROGRAM SHOWS HOW TO
     WRITE RANDOM-LENGTH RECORDS
10   HOME : DIM A$(20),B$(20),
     A(20):N = 1
20   LET D$ = "": REM  CTRL D
30   PRINT "ENTER 'END' TO QUIT":
     PRINT "ENTER STRING #";N;:
     INPUT A$(N): IF
     A$(N) = "END" THEN 60
40   PRINT "ENTER ANOTHER STRING";:
     INPUT B$(N): PRINT "ENTER A
     NUMBER ";: INPUT A(N)
50   PRINT :N = N + 1: GOTO 30
60   PRINT D$"OPEN FILE"
70   PRINT D$"WRITE FILE"
80   FOR X = 1 TO N - 1
90   PRINT A$(X): PRINT B$(X):
     PRINT A(X): NEXT
100   PRINT D$"CLOSE  FILE"
110   PRINT D$"OPEN FILE.PTR"
120   PRINT D$"WRITE FILE.PTR"
130   PRINT N - 1
140   PRINT D$"CLOSE FILE.PTR"
150   END

]
```

Example two reads the files created by example one. It first reads FILE.PTR, which tells it

how long the data file will be. Then (in lines 90-100) it reads in that many records from FILE. This is a sequential read, where each INPUT statement brings in the next record.

Lines 210-220 are random access reads (this has nothing to do with random record length), where we read a particular record from the middle of the file. To do random reads, we specify the desired record, *minus 1*. Note, however, that the computer doesn't know how long each logical record is. (They're random length, remember?) Therefore, it uses the *physical* record size, which is one byte long. (All records are physically one byte long, unless specified otherwise in an OPEN statement.) This means that if we say "READ FILE, R3," the next INPUT statement will start reading at the fourth *character* of the file. Try it and see.

EXAMPLE 2

```
5  REM  THIS PROGRAM SHOWS HOW TO
   READ BACK RANDOM-LENGTH RECORDS
10  LET D$ = "": REM  CTRL D
20  HOME : DIM C$(20),D$(20),C(20)
30  PRINT D$"OPEN FILE.PTR"
40  PRINT D$"READ FILE.PTR"
50  INPUT PTR
60  PRINT D$"CLOSE FILE.PTR"
70  PRINT D$"OPEN FILE"
80  PRINT D$"READ FILE"
90  FOR X = 1 TO PTR
100  INPUT C$(X): INPUT D$(X):
     INPUT C(X): NEXT
210  PRINT D$"READ FILE,R3"
220  INPUT C$(1),D$(1),C(1)
230  PRINT D$"CLOSE"
235  PRINT : PRINT : PRINT
240  PRINT C$(1),D$(1),C(1)
1000  END

]

]RUN RNDREADER
OPEN FILE.PTR
READ FILE.PTR
?3
CLOSE FILE.PTR
OPEN FILE
READ FILE
?THIS IS
?AN EXAMPLE
?1
?OF RANDOM
?RECORD
?2
?USAGE
?
?3
READ FILE,R3
?S IS
??AN EXAMPLE
??1
CLOSE


S IS  AN EXAMPLE     1
```

USING FIXED-LENGTH RECORDS

Fixed-length records are all of the same size and format. They offer the advantage of ease of use, since they always present information the same way. On the other hand, they are inflexible. You must know when you start a file how big the largest record will be. Then, any smaller records will waste disk space (since all records are as long as the longest one).

Example three shows the use of fixed-length records in both sequential and random access. Note that when writing into such a file (lines 120-150), we must specify each record number in a WRITE statement. Once the file is written, we can replace a random record with another that contains more characters (lines 160-180), so long as the new data does not exceed our 30-byte record length.

After a change is made, we can scan the whole file (lines 200-260) to find the change, read it, and do something about it.

EXAMPLE 3

```
10  REM   THIS PROGRAM PROVIDES AN
    EXAMPLE OF RANDOM RECORD ACCESS
40 :
60  REM  INITIALIZATION
70 :  DIM A$(40):D$ =  CHR$ (4)
80 :  PRINT D$;"NOMON C"
90  REM  CREATE FILE OF FIXED
    LENGTH, 30-BYTE RECORDS, EACH
    CONTAINING SIMILAR
100  REM  ASCII STRINGS.
110 :  PRINT D$;"OPEN TEST2,L30"
120 :  FOR I = 0 TO 5
130 ::  PRINT D$;"WRITE TEST2,R";I
140 ::  PRINT "NAME  ADDRESS ";I
150 :  NEXT I
160  REM  NOW CHANGE ONE RECORD....
170 :  PRINT D$;"WRITE TEST2,R3"
180 :  PRINT "APPLE DOS  VER 3.1"
200  REM  NOW READ THE FILE,
     LOOKING FOR THE CHANGE.
     PRINT SOMETHING WHEN IT
210  REM  IS FOUND
220 :  FOR J = 0 TO 5
230 ::  PRINT D$;"READ TEST2,R";J
240 ::  INPUT A$
250 ::  IF  LEFT$ (A$,5) = "APPLE"
     THEN  PRINT "THIS RECORD WAS
     CHANGED"
260 :  NEXT J
270  REM  NOW CLOSE THE FILE, SO
     YOU WON'T GET AN 'OUT OF DATA'
     ERROR WHEN
       THE
280  REM  PROGRAM TERMINATES.
290 :  PRINT D$;"CLOSE"
300  END
```

```
]RUN
NOMON C
NAME  ADDRESS 0
NAME  ADDRESS 1
NAME  ADDRESS 2
NAME  ADDRESS 3
NAME  ADDRESS 4
NAME  ADDRESS 5
APPLE DOS  VER 3.1
?NAME  ADDRESS 0
?NAME  ADDRESS 1
?NAME  ADDRESS 2
?APPLE DOS  VER 3.1
THIS RECORD WAS CHANGED
?NAME  ADDRESS 4
?NAME  ADDRESS 5
```

EXECUTE FILES

The last example concerns itself with EXEC files. Unlike other files that contain programs or data, EXEC files contain *commands*, just *exactly* as they would be typed on the keyboard by a computer operator. Thus, commands in this file are not proceeded by a CTRL/D, since that's not what you would type on the keyboard. An EXEC file is created by a program made up for the purpose. It simply opens a file, and then PRINTs each command into it, just as you would have typed it. Example 4 shows a typical EXEC-builder program, which creates a file called COMMANDS. While most operations are straightforward, putting quotes into the file (for string printing, etc.) is tricky. Lines 70-90 show how to do it.

After this program has run, you will have an EXEC file on your disk. If you then say EXEC COMMANDS, it will take control of the system, do the specified operations, and return command to the keyboard when finished.

EXAMPLE 4

```
10  REM  THIS PROGRAM SHOWS HOW
    TO BUILD AN EXEC FILE
20  LET D$ =  CHR$ (4): REM  CTRL D
30  PRINT D$"OPEN COMMANDS"
35  PRINT "FP"
40  PRINT D$"WRITE COMMANDS"
50  PRINT "FOR I=0 TO 10:PRINT I:
    NEXT I"
60  PRINT "RUN RNDREADER"
70  DIM Z$(100):Z$ = "THIS IS THE
    END OF THE EXEC FILE."
80 Q$ =  CHR$ (34): REM  WAY TO
    INSERT QUOTES
90  PRINT "PRINT";Q$;Z$;Q$
100  PRINT D$"CLOSE"
110  END
```

APPENDING FILES

Example 5 illustrates the use of the APPEND command. This command will open a file without

"rewinding" it back to the beginning. Thus, it allows you to build onto an existing file. The first item added to a file after it is appended will appear immediately following the last item of old data in the file. Try the following example to see how it works.

EXAMPLE 5

```
]LIST

100  REM   APPEND FILE

110  HOME :D$ = "": REM  CTRL D
120  INPUT "ENTER A STRING: ";A$
130  IF A$ = "" THEN 200
140  PRINT D$"APPEND TEST"
150  PRINT D$"WRITE TEST"
160  PRINT A$: PRINT D$"CLOSE"
170  GOTO 120
180 :
190 :
200  REM   GET IT BACK OUT

210  ONERR  GOTO 250
220  PRINT D$"OPEN TEST"
230  PRINT D$"READ TEST"
240  INPUT A$: GOTO 240
250  POKE 216,0: REM RESET FLAG
260  PRINT D$"CLOSE"
270 :
280 :
300  REM   THIS PROGRAM DEMON-
310  REM   STRATES THE USE OF
320  REM   THE 'APPEND' COMMAND
330 :
340  REM   IT WILL CONTINUE TO
350  REM   APPEND THE 'TEST'
360  REM   FILE UNTIL A NULL
370  REM   STRING IS ENTERED.
380 :
390  REM   AT THAT POINT IT WILL
400  REM   READ THE RECORDS BACK
410  REM   FROM THE FILE.

]
```

DOS ERROR CODES

The Disk Operating System handles its error codes in such a way that the Applesoft "ONERR GOTO" statement is able to trap them. Once an error has been trapped, conditional branching can be done depending on the nature of the error.

The following is a list of current DOS error codes:

CODE	ERROR
4	WRITE PROTECT
5	END OF DATA
6	FILE NOT FOUND
7	VOLUME MISMATCH
8	DISK I/O
9	DISK FULL
10	FILE LOCKED
11	COMMAND SYNTAX
12	NO FILE BUFFS
13	NOT BASIC PROGRAM
14	PROGRAM TOO LARGE
15	NOT BINARY FILE

The statement "X = PEEK (222)" sets the value of the variable "X" equal to the code of the error encountered. (Decimal location 222 is also the spot in which Applesoft places it's own set of error codes).

Another note is in order here. As part of any err-handling routine, your program must clear the err flag (set in memory by the error) in order for any other error messages to appear normally. If this is not done, any error, even a syntax error in command mode, will jump to your program's onerr "destination!" To clear the errflag, use the statement "POKE 216, 0."

The statement "Y = Peek (218) + Peek (219) *256" returns "Y" as equal to the line number in which an error occured if an "Onerr/Goto" statement has been previously executed.

GETTING COMMAS INTO APPLESOFT

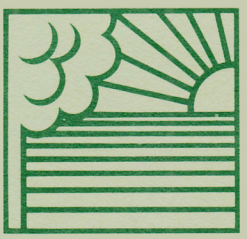
Several users have called us regarding the Applesoft comma problem. To those of you who haven't run across it, what happens is this: Your program stops for an input statement and you type in an answer (with an imbedded comma) and hit return. Then your Apple brilliantly types back "?EXTRA IGNORED." And it's done just that! The reason is that Applesoft uses the comma as a delimiter for input statements (so you may use "INPUT X, Y, Z"). The same thing occurs with a colon in an input response.

The program shown in Example 6 shows how to make it work.

EXAMPLE 6

```
\
]LIST

90 : REM  FIRST WE 'GET' A STRING


100  HOME :D$ =  CHR$ (4): PRINT
     "ENTER A STRING (INCUDE COMM
     AS) : ";
110  GET A$: IF A$ =  CHR$ (13) THEN
     130
120  LET B$ = B$ + A$: PRINT A$;:
      GOTO 110
125  REM

NOW WE WRITE IT TO DISK


130  PRINT : PRINT D$"OPEN FILE
140  PRINT D$"WRITE FILE"
150  PRINT B$ +  CHR$ (13)
160  PRINT D$"CLOSE FILE"
165  REM

AND GET IT BACK OUT!!


170  PRINT D$"OPEN FILE"
180  PRINT D$"READ FILE"
190  GET A$: IF A$ =  CHR$ (13) THEN
     210
200  LET C$ = C$ + A$: GOTO 190
210  PRINT C$
220  PRINT D$"CLOSE FILE"

]
```

Another point is that input from a disk file converts any lower case which you have laboriously constructed to upper case as it goes through the monitor routines. Using the above "GET" routine also works in this case. The only thing to watch for in your program is that any "GET" command *must* be followed by a "Print" or the DOS will work as if your program was in the "Trace" mode (i.e., not at all!)



OUTSIDE THE ORCHARD

. . . Some new things from others, for you

Software

Yet another source of software intended specifically for APPLE has come to our attention. The company's current list

apple computer inc.
10260 Bandley Drive
Cupertino, California 95014
(408) 996-1010

THIRD CLASS
U.S. Postage Paid
Permit No.
3440
San Francisco, CA

– a varied one that comprises games, music, financial aids, logic development, and more – totals sixteen programs, each on cassette, at prices ranging from $5 to $49, with about half pegged at the $10 mark. Rainbow Computing, Inc., 17023 White Oak Ave., Granada Hills, CA 91344. (213) 360-2171.

Joystick unit

According to its manufacturer, this new full-range joystick has been designed to plug directly into APPLE's game I/O connector. With each joystick are two switches and two trim-pots for adjusting to any APPLE II and any application. Completely assembled, the single joystick sells for $39.95; a double joystick is available at $79.95. Quantity discounts are available, and delivery is 30 days ARO. Microproducts Company, 1128 19th St., Santa Monica, CA 90403. (213) 393-8371.

Interface board

A newly announced peripheral interface board contains two 2716 PROMs, 20-mA and RS232 interfaces, a real-time clock, and a parallel output port. The board also has two 16-pin DIP sockets, which may be used to connect the board to an external source. Contact the manufacturer for pricing information on the complete board. A bare board – without components, but with assembly instructions, diagrams, and programs – is available for $29. Delivery, 3-4 weeks ARO. Peripheral Interface, 173-1128 McKercher Dr., Saskatoon, Saskatchewan, Canada S7H 4Y7.

Analog input card

The AI-02 is a single-card, 16-channel, analog data acquisition system for APPLE II. Each channel is individually addressable through software. The analog-to-digital conversion takes 70μs (8-bit resolution), after which an interrupt or a completion flag is activated and the converted value can be acquired by APPLE. Contact the manufacturer for more information on the AI-02, as well as for information on other of the firm's products for APPLE, such as a video input interface, and inventory and remote data entry software packages. Interactive Structures, Inc., Suite 204, Science Center, 3401 Market St., Philadelphia, PA 19104. (215) 381-8296.

Clock Interface (correction)

System Design Engineering (mentioned in CONTACT 3 as a source for a real-time clock) has moved. You can reach them at 25131½ Cypress, Lomita, CA 90717. (213) 370-1245 (11 - 8 p.m.); (213) 530-9891 (after 8 p.m.)

Dust cover

Protect your investment from dust, coffee spills, and idle fingers with a heavy duty beige vinyl cover, custom fitted to your Apple II Computer. For only $6.95 (includes shipping) you can help to insure a long and useful life for your investment.

TO ORDER, send check or money order to:

Henwood Enterprises, Inc.
1833 E. Crabtree Dr.
Arlington Heights, IL 60004

OR CALL TOLL FREE 800-323-7360 and use your Master Charge, VISA, or American Express credit card.