Open-Apple™

Releasing the power to everyone.

August 1985
Vol. 1, No. 7

AppleWorks Pie

AppleWorks makes plain and clear the computer's potential as a personal tool. And it elucidates the potential hidden within the "aged technology" of the Apple II. *AppleWorks* is also today's best-selling Apple II program. It sells so well, in fact, that it has become as important a characteristic of the Apple II as DOS, Applesoft, or the Monitor. It doesn't need to be italicized anymore.

I suspect that AppleWorks has snuck up on many of you old-timers; I suspect many of you aren't ready to accept it as one of the defining characteristics of the Apple II. But look at today's first-time Apple buyers. They are investing their learning time in AppleWorks rather than in Applesoft and DOS. Look at user group newsletters—there has been a subtle shift in their content during the last 18 months away from material on Basic programming and toward material on AppleWorks programming.

Releasing the full potential of the Apple II in 1985 requires expertise in both areas. Just as Applesoft allows you to do things that you'd never accomplish in assembly language, AppleWorks allows you to do things that you'd never accomplish in Basic—writing the program would simply take too long. On the other hand, just as assembly language routines can add speed and flexiblity to Applesoft programs, Applesoft routines can add speed and flexibility to the manipulation of data stored in AppleWorks files. This month we're going to start examining AppleWorks in detail.

AppleWorks on the II-Plus. Before all you II-Plus and Franklin users get mad and fire off angry letters to me, here's some good news for you. If you already have an 80-column card and 64K, you can get a program that will modify AppleWorks to run on your machine for as little as $19.95. The program is called *Plus-Works* and it's available from Norwich Data Services, P.O. Box 356, East Norwich, N.Y. 11732 (516-922-9584).

The program comes in three versions. The basic $19.95 version allows a 10K desktop. It provides for using escape as the open-apple key, for using control keys for the up and down arrows and delete key, and works with Videx-compatible 80-column cards (essentially all major cards except the Smarterm, Wizard, and Super-R-Term).

The other two versions are *Plus-Works XM* and *Plus-Works XM-P.* These versions cost $49.95 and provide all the features of the basic version, plus have the ability to increase the AppleWorks desktop size with standard-peripheral-slot memory boards (Legend and Saturn cards and the equivalent—but not Applied Engineering's RAMworks, which is an *auxiliary* slot memory board). Even IIe owners may be interested in *Plus-Works* if they already have one of these boards. Unlike the RAMworks expansion software, *Plus-Works* puts the entire available memory into the desktop—there is no loss due to overhead. Like RAMworks, (and unlike a similar program Videx is selling) *Plus-Works* modifies an image of AppleWorks, which can then be copied onto backup floppies or to a hard disk. However, *Plus-Works* does not yet provide any of the extra features of the RAMworks expander, such as file segmentation or increasing the maximum number of data base records.

The difference between the XM and XM-P versions is that the latter can also use the 64K of memory on a PCPI CP/M card. Franklin 1200 computers came with these cards as standard equipment, so this version was developed primarily for Franklins, but it works on Apples with the PCPI card as well.

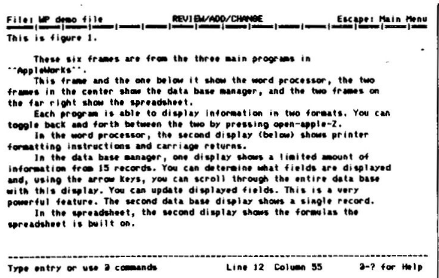
I have not tested *Plus-Works* myself, but was made aware of it by a call from an enthusiastic ***Open-Apple*** subscriber who has. I've talked to the people at Norwich Data Systems on the phone, and I think they potentially have a hot product here. It makes AppleWorks available on the entire Apple II family.

At the top of the next page is a set of six AppleWorks screen images. The two on the left are from the word processor, the two in the center from the data base program, and the two on the right from the spreadsheet. These are the three major programs within AppleWorks.

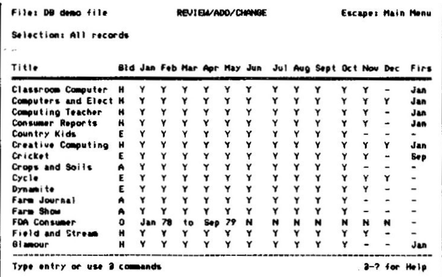
The word processor. The AppleWorks word processor provides standard full-screen editing. You move the cursor around the screen with the Apple's arrow keys. All of AppleWorks, including the word processor, supports two cursors. One, the "insert cursor," always pushes any existing text ahead of it as new characters are entered. It looks like a blinking underline. The second cursor is called the "overstrike cursor." It is a blinking box. With it, whatever you type replaces the character at that position. However, existing carriage returns are not overwritten but pushed to the right, so even this cursor will insert text in the right situation. At any point within AppleWorks you can switch between the two cursors by pressing open-apple-E.

Print formatting—adjusting margins, choosing type size and style, choosing line spacing and justification—is done throughout AppleWorks with the open-apple-O command. In the word processor, this command overlays a menu on the lower half of the screen, which shows all the

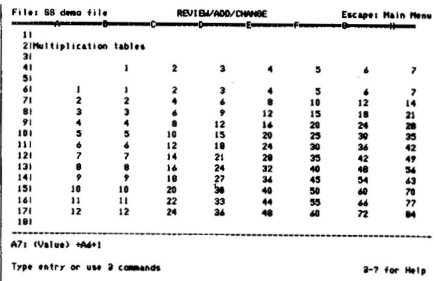
"OF COURSE IT'S PORTABLE SIR, LOOK HERE'S THE HANDLE."



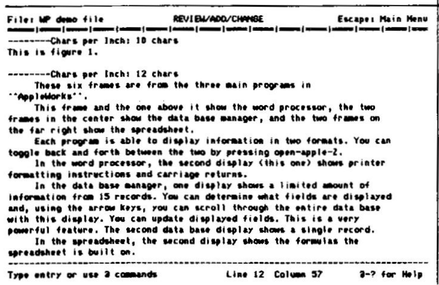
```
File: WP demo file           REVIEW/ADD/CHANGE           Escape: Main Menu
This is figure 1.

     These six frames are from the three main programs in
``AppleWorks``.
     This frame and the one below it show the word processor, the two
frames in the center show the data base manager, and the two frames on
the far right show the spreadsheet.
     Each program is able to display information in two formats. You can
toggle back and forth between the two by pressing open-apple-Z.
     In the word processor, the second display (below) shows printer
formatting instructions and carriage returns.
     In the data base manager, one display shows a limited amount of
information from 15 records. You can determine what fields are displayed
and, using the arrow keys, you can scroll through the entire data base
with this display. You can update displayed fields. This is a very
powerful feature. The second data base display shows a single record.
     In the spreadsheet, the second display shows the formulas the
spreadsheet is built on.

--------------------------------------------------------------------------
Type entry or use ∂ commands         Line 12  Column 55      ∂-? for Help
```



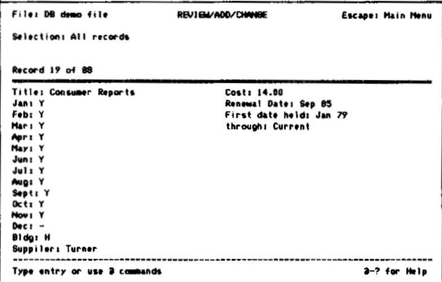
```
File: DB demo file           REVIEW/ADD/CHANGE           Escape: Main Menu

Selection: All records

Title               Bld Jan Feb Mar Apr May Jun  Jul Aug Sept Oct Nov Dec  Firs
-------------------------------------------------------------------------------
Classroom Computer  H   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   Y   -    Jan
Computers and Elect H   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   Y   Y    Jan
Computing Teacher   H   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   Y   -    Jan
Consumer Reports    H   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   Y   -    Jan
Country Kids        E   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   -   -    -
Creative Computing  H   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   Y   Y    Jan
Cricket             E   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   Y   -    Sep
Crops and Soils     A   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   -   -    -
Cycle               E   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   Y   Y    -
Dynamite            E   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   Y   -    -
Farm Journal        A   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   -   -    -
Farm Show           A   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   -   -    -
FDA Consumer        O   Jan 78  to  Sep 79  N    N   N   N    N   N   N    -
Field and Stream    H   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   Y   -    -
Glamour             H   Y   Y   Y   Y   Y   Y    Y   Y   Y    Y   -   -    Jan
-------------------------------------------------------------------------------
Type entry or use ∂ commands                                    ∂-? for Help
```



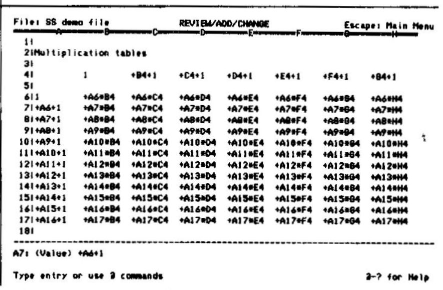
```
File: SS demo file           REVIEW/ADD/CHANGE            Escape: Main Menu
--------A--------B--------C--------D--------E--------F--------G--------H----
  1|
  2|Multiplication tables
  3|
  4|               1        2        3        4        5        6        7
  5|
  6|       1       1        2        3        4        5        6        7
  7|       2       2        4        6        8       10       12       14
  8|       3       3        6        9       12       15       18       21
  9|       4       4        8       12       16       20       24       28
 10|       5       5       10       15       20       25       30       35
 11|       6       6       12       18       24       30       36       42
 12|       7       7       14       21       28       35       42       49
 13|       8       8       16       24       32       40       48       56
 14|       9       9       18       27       36       45       54       63
 15|      10      10       20       30       40       50       60       70
 16|      11      11       22       33       44       55       66       77
 17|      12      12       24       36       48       60       72       84
 18|
----------------------------------------------------------------------------
A7: (Value) +A6+1

Type entry or use ∂ commands                                   ∂-? for Help
```



```
File: WP demo file           REVIEW/ADD/CHANGE           Escape: Main Menu
--------Chars per Inch: 10 chars
This is figure 1.

--------Chars per Inch: 12 chars
     These six frames are from the three main programs in
``AppleWorks``.
     This frame and the one above it show the word processor, the two
frames in the center show the data base manager, and the two frames on
the far right show the spreadsheet.
     Each program is able to display information in two formats. You can
toggle back and forth between the two by pressing open-apple-Z.
     In the word processor, the second display (this one) shows printer
formatting instructions and carriage returns.
     In the data base manager, one display shows a limited amount of
information from 15 records. You can determine what fields are displayed
and, using the arrow keys, you can scroll through the entire data base
with this display. You can update displayed fields. This is a very
powerful feature. The second data base display shows a single record.
     In the spreadsheet, the second display shows the formulas the
spreadsheet is built on.
--------------------------------------------------------------------------
Type entry or use ∂ commands         Line 12  Column 57      ∂-? for Help
```



```
File: DB demo file           REVIEW/ADD/CHANGE           Escape: Main Menu

Selection: All records


Record 19 of 88
-------------------------------------------------------------------------
Title: Consumer Reports                   Cost: 14.00
Jan: Y                                    Renewal Date: Sep 85
Feb: Y                                    First date held: Jan 79
Mar: Y                                    through: Current
Apr: Y
May: Y
Jun: Y
Jul: Y
Aug: Y
Sept: Y
Oct: Y
Nov: Y
Dec: -
Bldg: H
Suppiler: Turner
-------------------------------------------------------------------------
Type entry or use ∂ commands                                 ∂-? for Help
```



```
File: SS demo file           REVIEW/ADD/CHANGE            Escape: Main Menu
--------A--------B--------C--------D--------E--------F--------G--------H----
  1|
  2|Multiplication tables
  3|
  4|          1        +B4+1    +C4+1    +D4+1    +E4+1    +F4+1    +G4+1
  5|
  6|1         +A6*B4   +A6*C4   +A6*D4   +A6*E4   +A6*F4   +A6*G4   +A6*H4
  7|+A6+1     +A7*B4   +A7*C4   +A7*D4   +A7*E4   +A7*F4   +A7*G4   +A7*H4
  8|+A7+1     +A8*B4   +A8*C4   +A8*D4   +A8*E4   +A8*F4   +A8*G4   +A8*H4
  9|+A8+1     +A9*B4   +A9*C4   +A9*D4   +A9*E4   +A9*F4   +A9*G4   +A9*H4
 10|+A9+1     +A10*B4  +A10*C4  +A10*D4  +A10*E4  +A10*F4  +A10*G4  +A10*H4
 11|+A10+1    +A11*B4  +A11*C4  +A11*D4  +A11*E4  +A11*F4  +A11*G4  +A11*H4
 12|+A11+1    +A12*B4  +A12*C4  +A12*D4  +A12*E4  +A12*F4  +A12*G4  +A12*H4
 13|+A12+1    +A13*B4  +A13*C4  +A13*D4  +A13*E4  +A13*F4  +A13*G4  +A13*H4
 14|+A13+1    +A14*B4  +A14*C4  +A14*D4  +A14*E4  +A14*F4  +A14*G4  +A14*H4
 15|+A14+1    +A15*B4  +A15*C4  +A15*D4  +A15*E4  +A15*F4  +A15*G4  +A15*H4
 16|+A15+1    +A16*B4  +A16*C4  +A16*D4  +A16*E4  +A16*F4  +A16*G4  +A16*H4
 17|+A16+1    +A17*B4  +A17*C4  +A17*D4  +A17*E4  +A17*F4  +A17*G4  +A17*H4
 18|
----------------------------------------------------------------------------
A7: (Value) +A6+1

Type entry or use ∂ commands                                   ∂-? for Help
```

formatting options. **Boldface** and *underlining* can be turned on and off either by means of this menu, or directly with control-B and control-L. (For example, the first use of control-B within a paragraph turns bold on, the second turns it off. AppleWorks automatically turns off bold and underlining at the end of each paragraph.) The only other control-code supported by AppleWorks is control-Y, which deletes all material from the cursor position to the end of the line.

Some print formatting instructions, such as margin adjustments and centering, affect the screen display as well as printed output. One nice feature is that the open-apple-K command will calculate and display the position of all page breaks (in the spreadsheet open-apple-K forces recalculation).

Each of the three AppleWorks programs has two display formats. The standard formats are shown in the three images on the top of this page. The secondary formats are shown in the three images just under the top three. When using AppleWorks, you can toggle or switch between the two available display formats by pressing open-apple-Z(oom).

In the word processor, the second display shows embedded formatting information. At the top of this page, the second word processor display shows formatting information relating to the number of characters per inch. AppleWorks supports the multiple character widths available on today's dot matrix printers, including both double-wide and proportional characters.

The data base program. The data base program in AppleWorks, like the other AppleWorks modules, insists on having the entire data file being manipulated available in memory. This is not unusual in programs for the Apple II, but many CP/M users find it odd indeed. They are used to programs such as *Wordstar* and *dBase II* that work with disk-based files. These programs load only the actual part of the file being manipulated into memory. CP/M fans complain a lot about programs such as AppleWorks that can't deal with large, disk-based files. The difference, of course, is that disk-based programs must access the disk constantly. They tend to operate with a deliberate slowness, while programs that deal with memory-resident data are lightning fast. Both have their place in the world. Speed and memory-resident files, used by such early Apple II best-sellers as *VisiCalc* and *Apple Writer*, are the Apple II tradition.

By limiting itself to memory-resident data, the AppleWorks data base program is able to provide some outstanding features not usually found in data base programs. One is the standard data base display, shown in the middle of the top row above. This is called the multiple-record layout. It allows you to see 80 columns worth of information per record from the fields you select. Since each record is displayed on one line, you can see the same fields from 15 records at once on the screen. The arrow keys allow you to scroll forward and backward through the file. As in all AppleWorks programs, the open-apple key in conjunction with the number keys will move the display to any part of the file (open-apple-1 takes you to the beginning, open-apple-9 to the end, other digits to points in between).

Pressing open-apple-Z in the data base switches to the single-record layout. Whatever record the cursor was on in the multiple-record layout is shown in full in the single-record layout. A single record can hold no more information than will fit on this page. The information can be divided into as many as 30 categories or fields. No single field will hold more information than will fit on a single line—76 characters if you use a one-character category name. As supplied by the factory, AppleWorks allows 1,350 records per file, but Applied Engineering's RAMworks software increases this to over 4,000.

The spreadsheet. The AppleWorks spreadsheet has several impressive features. The first is its size—999 rows by 127 columns. An undocumented feature of the spreadsheet, however, is that only 60 of the cells in a single row can hold formulas. Usually you can get around this limitation, however, once you understand what's going on, by continuing a series of formulas down one row. All 999 cells in a single column, on the other hand, can hold formulas; all 127 cells in a single row can hold values or labels. How many cells you can actually fill, of course, depends on how much memory you have. Apple's documentation says a 128K computer can handle a spreadsheet with about 6,000 filled cells.

Another interesting feature of the spreadsheet is its ability to sort a block of rows according to the data shown in one of the spreadsheet's columns. For example, if you used the AppleWorks spreadsheet to create a grade book and entered student's names in the first column and their scores on the first test in column two, you could easily rank the students by order of test score—from lowest to highest or from highest to lowest. After that, you could put them back in alphabetical order by sorting the same rows on the data in the name column.

In the spreadsheet, zooming with open-apple-Z switches between standard spreadsheet format and a format that causes the formulas within the cells to appear. Only the portion of the formula that fits within the cell width will show. If you print from this display, your hard copy will also show formulas rather than the resulting data. This can be a very useful feature.

None of the three AppleWorks programs is the most powerful in its class. If you have heavy-duty word processing to do, *Apple Writer* is better. *Supercalc 3a*, which can generate graphs and use textual values, among other features not found in AppleWorks, is a more powerful spreadsheet (but why is its maximum size limited to 254 rows by 63 columns?). Disk-based data base programs such as *General Manager* and *DB Master* are more suitable for data bases with large records than the data base program in AppleWorks.

AppleWorks more than makes up for its relative lack of power on a one-on-one basis, however, with its tight integration. All three AppleWorks programs use similar, if not the same, commands. But more importantly, the three programs are closely knit by the AppleWorks Desktop.

The AppleWorks Desktop. The desktop metaphor is something AppleWorks shares with the Macintosh. Having worked with both, I find the Macintosh desktop to be far more glamorous. However, the AppleWorks Desktop is far more useful. It's fast—both in execution speed and in the speed at which you can give commands. Yet it's easy—the Desktop is entirely menu-driven.

Like many other Apple II programs, the AppleWorks Desktop uses what Apple calls the "magic menu" interface. At the bottom of the next page is a set of Desktop menu screens. The screen spown in the upper center is the Desktop's main menu. When you see this menu live on-screen, one of the six menu items is highlighted with an inverse bar. You can move the bar by either pressing the number of a different item or by pressing the up or down arrow keys. When you press Return, the highlighted choice is executed.

The "magic menu" interface works beautifully in AppleWorks. Unfortunately, this interface is out of favor at Apple Inc. Nowadays Apple encourages software developers to use a mouse/pull-down menu interface based on MouseText characters. This system is similar to that found on the Macintosh. Both interfaces have a place in the world, but more people than the computer magazines and marketers seem to realize prefer magic menus (including me). Unlike a mouse-based interface, magic menus don't require a flat empty space near the computer for the mouse to frolic in and they don't require removing one's hands from the keyboard every time a command must be given. Let's hope the strength of AppleWorks is enough to keep the magic menu interface alive and healthy.

The first three choices in the Desktop's main menu add files to the Desktop, allow you to work with a file already on the Desktop, and save files on the Desktop to disk. Unlike the Macintosh desktop, which is really just a fancy disk directory, you can actually load files onto the AppleWorks Desktop. Up to 12 AppleWorks files can be on the Desktop at once.

If you catalog a data disk holding AppleWorks data files from Basic.system (AppleWorks is ProDOS based), you'll find the files have unique file types. AppleWorks word processor files are identified with the three letters AWP, data base files with ADB, and spreadsheet files with ASP. Thus, when you load an AppleWorks file onto the Desktop, AppleWorks knows what kind of file it is and executes the appropriate program when you decide to "work with" that file. There is no way within the context of AppleWorks to "run" the word processor or to "run" the spreadsheet. What you do is add a word processor file or a spreadsheet file to the Desktop and begin to work with it.

The second screen in the center below shows what happens when you choose the second item on the main menu, "Work with one of the files on the Desktop." A *Desktop Index* appears on the screen. It, too, holds a magic menu bar. You use the bar to select which of the files currently on the desktop you want to work with next.

An important feature of AppleWorks is that open-apple-Q(uick change) can be used to call up the Desktop Index from *anywhere* within AppleWorks (whether you are deep within desktop menus or in any one of the three main programs). Open-apple-Q allows you to immediately choose a new file to work with. This means you can have several word processor or spreadsheet files on the desktop at once and quickly switch between them. Or you can switch between a data base or spreadsheet file holding your data and a word processor file holding a report you are writing. Or you can instantly switch between the budget you are working on in a spreadsheet file to your personal phone directory in a data base file or to your personal calendar in a word processor file. This is where AppleWorks gives you real power compared to stand-alone programs. (Open-apple-Q is also a handy command to use when you simply want to see how much space is left on the desktop—the amount appears in the lower right corner of the screen whenever the Desktop Index is present, as well as at some other times.)

The clipboard. AppleWorks provides a "clipboard" you can use to move data between any two files *of the same type.* Spreadsheet rows, including formulas, can be moved from one spreadsheet file to another. Text can be quickly cut from one word processor file and moved to another. Data base records can be moved from one file into any other (if the two files have different numbers of fields some data may be lost, but as much of the data as can be moved will be moved). You access the clipboard with the open-apple-C(opy) or open-apple-M(ove) commands. (Copy creates a duplicate of the selected material and moves it to or from the clipboard. Move deletes the selected material from the source while moving it.)

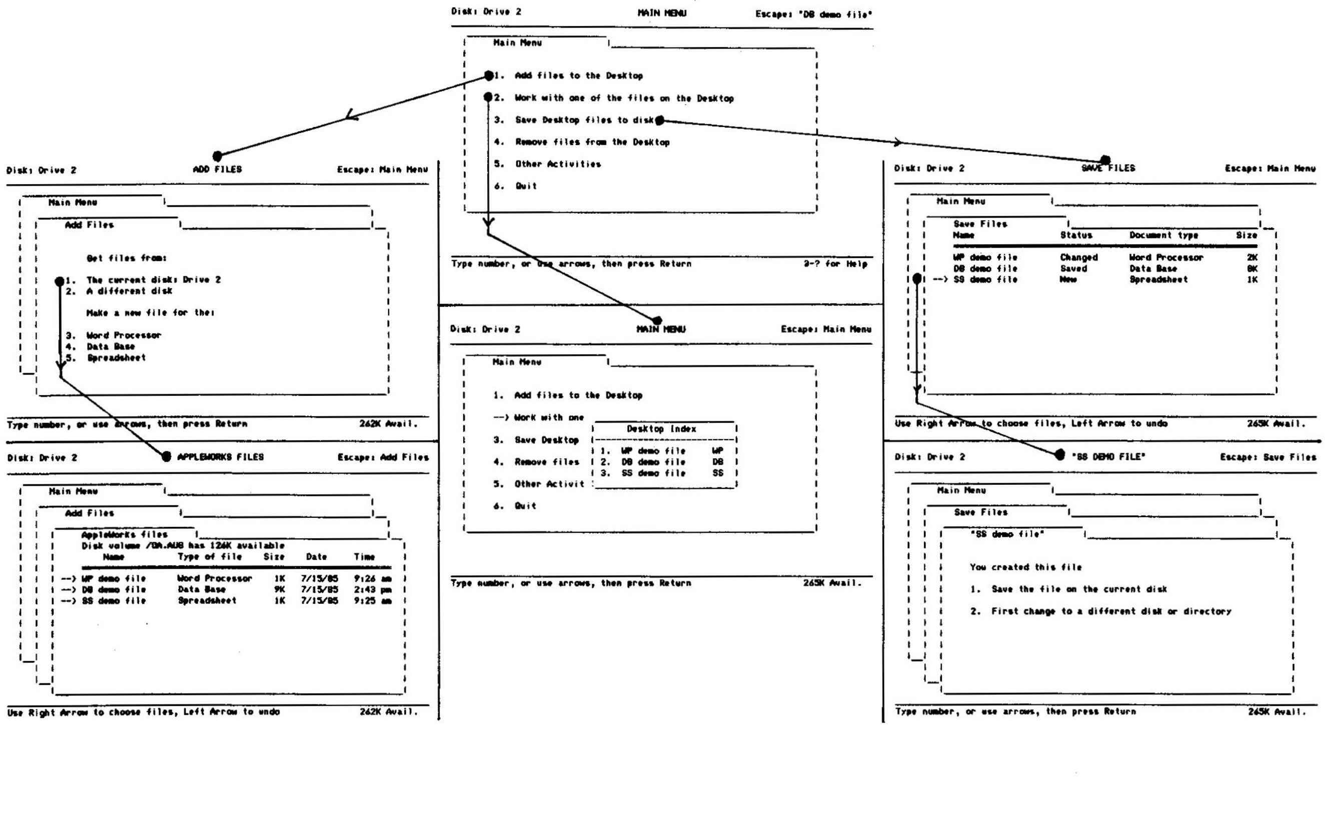
In addition to using the clipboard for moving data between files of the same type, data base and spreadsheet reports can be printed to the clipboard. They appear there just as they would if they had been printed on your printer. Once these reports are on the clipboard, they are word processor material, so you can move them into word processor files for futher editing or to combine them with other information.

Placing something on the clipboard deletes what was already there. There is no other way the clipboard can be cleared except by quitting AppleWorks. This is basically good, but can cause problems when a large amount of no-longer-needed material is on the clipboard and taking up precious Desktop space. There is no way to see what's on the clipboard except by copying it into a file of the appropriate type. We'll examine some of the tricks you can do with the clipboard further in future issues.

Adding files to the Desktop. The two screen images on the left side of this page show the menus for adding files to the Desktop. AppleWorks mercifully allows floppy drives to be specified by drive number rather than by ProDOS volume name. However, it also allows files to be specified with a ProDOS prefix, which works great for users with high-capacity disk drives. You can specify a prefix or a different drive by choosing item 2, "Get files from a different disk" on the Add Files menu.

When you choose to Add Files from the current disk, a catalog of that disk appears on your screen showing all of the AppleWorks files available. This is shown in the lower-left screen image on this page. Note that only AppleWorks files appear in the catalog. There are ways to load other types of files (for example, standard ASCII, DIF, and VisiCalc-format files) into AppleWorks, but this isn't it. We'll get to that in a moment.

Once the disk catalog is on your screen, you can scroll through it with the up and down arrows. You can select and deselect files for loading with the



```
Disk: Drive 2                    MAIN MENU              Escape: "DB demo file"

     Main Menu

     1.  Add files to the Desktop
     2.  Work with one of the files on the Desktop
     3.  Save Desktop files to disk
     4.  Remove files from the Desktop
     5.  Other Activities
     6.  Quit

Type number, or use arrows, then press Return                  ⓐ-? for Help
```

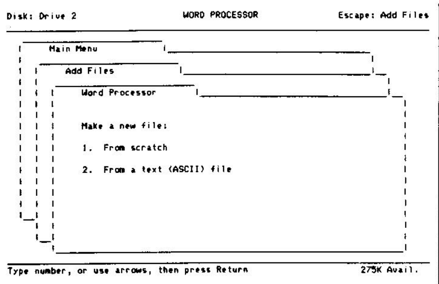
```
Disk: Drive 2                    ADD FILES                Escape: Main Menu

     Main Menu
       Add Files

          Get files from:

          1.  The current disk: Drive 2
          2.  A different disk

          Make a new file for the:

          3.  Word Processor
          4.  Data Base
          5.  Spreadsheet

Type number, or use arrows, then press Return                  262K Avail.
```

```
Disk: Drive 2                 APPLEWORKS FILES            Escape: Add Files

     Main Menu
       Add Files
         AppleWorks files
         Disk volume /OA.AUG has 124K available
              Name          Type of file     Size    Date      Time
         -------------------------------------------------------------
     --> WP demo file       Word Processor    1K    7/15/85   9:26 am
     --> DB demo file       Data Base         9K    7/15/85   2:43 pm
     --> SS demo file       Spreadsheet       1K    7/15/85   9:25 am

Use Right Arrow to choose files, Left Arrow to undo            262K Avail.
```

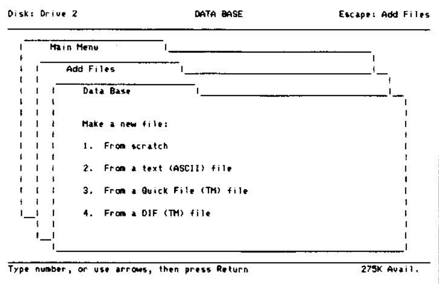
```
Disk: Drive 2                    MAIN MENU                Escape: Main Menu

     Main Menu

     1.  Add files to the Desktop

     --> Work with one    |      Desktop Index       |
                          |--------------------------|
     3.  Save Desktop     | 1.  WP demo file     WP  |
                          | 2.  DB demo file     DB  |
     4.  Remove files     | 3.  SS demo file     SS  |
                          |__________________________|
     5.  Other Activit

     6.  Quit

Type number, or use arrows, then press Return                  265K Avail.
```

```
Disk: Drive 2                    SAVE FILES               Escape: Main Menu

     Main Menu
       Save Files
       Name               Status       Document type        Size
       ----------------------------------------------------------
       WP demo file       Changed      Word Processor        2K
       DB demo file       Saved        Data Base             8K
   --> SS demo file       New          Spreadsheet           1K

Use Right Arrow to choose files, Left Arrow to undo            265K Avail.
```

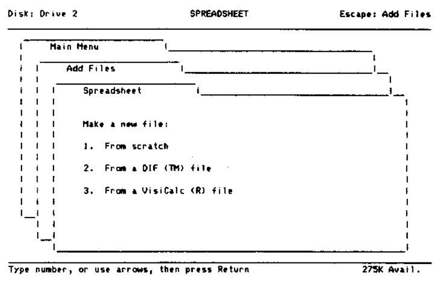
```
Disk: Drive 2                  "SS DEMO FILE"            Escape: Save Files

     Main Menu
       Save Files
         "SS demo file"

         You created this file

         1.  Save the file on the current disk

         2.  First change to a different disk or directory

Type number, or use arrows, then press Return                  265K Avail.
```



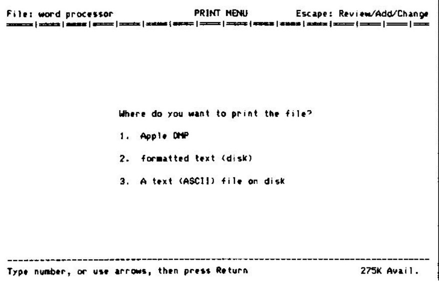
```
Disk: Drive 2              WORD PROCESSOR            Escape: Add Files

    Main Menu
       Add Files
          Word Processor

          Make a new file:

          1.  From scratch

          2.  From a text (ASCII) file

Type number, or use arrows, then press Return            275K Avail.
```



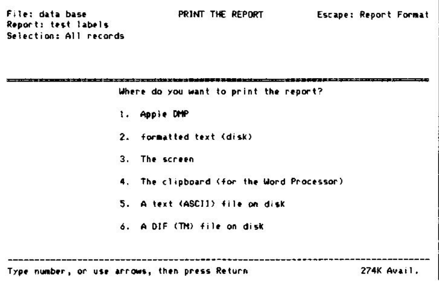
```
Disk: Drive 2                DATA BASE               Escape: Add Files

    Main Menu
       Add Files
          Data Base

          Make a new file:

          1.  From scratch

          2.  From a text (ASCII) file

          3.  From a Quick File (TM) file

          4.  From a DIF (TM) file

Type number, or use arrows, then press Return            275K Avail.
```



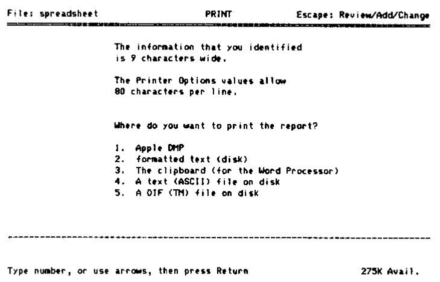
```
Disk: Drive 2               SPREADSHEET              Escape: Add Files

    Main Menu
       Add Files
          Spreadsheet

          Make a new file:

          1.  From scratch

          2.  From a DIF (TM) file

          3.  From a VisiCalc (R) file

Type number, or use arrows, then press Return            275K Avail.
```



```
File: word processor         PRINT MENU     Escape: Review/Add/Change




          Where do you want to print the file?

          1.  Apple DMP

          2.  formatted text (disk)

          3.  A text (ASCII) file on disk



Type number, or use arrows, then press Return            275K Avail.
```



```
File: data base           PRINT THE REPORT     Escape: Report Format
Report: test labels
Selection: All records


              Where do you want to print the report?

              1.  Apple DMP

              2.  formatted text (disk)

              3.  The screen

              4.  The clipboard (for the Word Processor)

              5.  A text (ASCII) file on disk

              6.  A DIF (TM) file on disk

Type number, or use arrows, then press Return            274K Avail.
```



```
File: spreadsheet              PRINT        Escape: Review/Add/Change

              The information that you identified
              is 9 characters wide.

              The Printer Options values allow
              80 characters per line.

              Where do you want to print the report?

              1.  Apple DMP
              2.  formatted text (disk)
              3.  The clipboard (for the Word Processor)
              4.  A text (ASCII) file on disk
              5.  A DIF (TM) file on disk

Type number, or use arrows, then press Return            275K Avail.
```

right and left arrows. If you select several files, AppleWorks will load them all to the desktop at once.

If the file you want to work with doesn't exist yet, or if it's in a non-AppleWorks file, rather than choosing to "Get files from the current disk" on the Add Files menu, you should select "Make a new file."

The three screen images across the top of this page show the next menu that will appear, depending on whether you choose to make a new file for the word processor, the data base, or the spreadsheet.

If you are making a new AppleWorks file for the word processor, you can either make it from scratch (that is, you'll start with a new, empty file), or you can make it from an existing text file on disk, as shown in the upper-left screen image.

Data base files can be made from scratch or can come from text files, Quick File files, or DIF files. (Text files for the data base should have a Return after each field or category. Each record must have the same number of fields. Records need no separators—AppleWorks figures it is at the end of a record when the proper number of fields has been read. You have to specify how many this will be before loading the file.)

Spreadsheet files can be made from scratch or can come from VisiCalc-format files or from DIF files. (For a complete description of VisiCalc-format files, see the January 1984 **DOStalk** in *Softalk*, page 237. For more information on DIF files, see the February 1984 **DOStalk** in *Softalk*, page 65.)

Be forewarned, however, that if you decide to make a new AppleWorks file from some other type of file there are two problems. First of all, the old file must be on a ProDOS disk. If it's on a DOS 3.3 disk, you'll have to convert it to ProDOS first. Even then, AppleWorks won't allow you to select the file from a catalog as before. Instead you'll have to type in the file's complete ProDOS pathname. If you don't know its name, you have to escape to the Main Menu, choose "Other Activities", choose "List all files on the current disk drive," memorize or write down the filename, and follow your breadcrumbs back again.

This is quite painful, especially if birds eat your breadcrumbs. A definite improvement for AppleWorks would be to allow any kind of file to be selected from a menu. A further improvement would be to show ProDOS subdirectories in these menus, and to list their contents when they are selected. This would greatly ease the process of selecting files by pathname.

Saving files. The two right-most screens on the previous page show the menus you get when you select "Save Desktop files to disk" on the Main Menu. The first screen that appears allows you to select the files currently on the desktop that you want to save. All files on the desktop, along with their status (new/unchanged/changed/saved), type, and size will be displayed. You select the files you want to save as before, and press Return.

Another menu then appears that allows you to save the files on the current disk, or to first switch to a different disk. AppleWorks does not delete original files until a save has been successfully completed. This means the biggest file you can resave on a floppy is half the capacity of the floppy. (If you have a bigger file, save it on a different disk or delete the original before attempting to resave it.)

You cannot *lock* or *unlock* files from within AppleWorks, though this may be a good idea for special empty templates of various types you don't want spoiled by mistake. You *can* lock AppleWorks files from Basic or with the Filer or IIc Utilities. This prevents AppleWorks from overwriting the file, even if you mistakenly tell it to.

When you use the "Save desktop files to disk" option from the main menu, the only types of file that information can be stored in are the three AppleWorks file types. However, just as AppleWorks can load information from other file types, it can also save information into other types of files. The only problem is that how you do it isn't at all obvious.

The bottom three screen images on this page show the menus you use for saving files in non-AppleWorks formats. To get to these menus you must be "working with" the file you want to save. Then choose open-apple-P(rint).

Before we can say much intelligent about these menus, you have to know that AppleWorks lets you define up to three printers for its use. One of these three can be a "custom" printer, the other two have to appear on a menu inside AppleWorks that has various Apple, Epson, and Qume printers listed.

In the screen images shown, two printers have been defined. One is an Apple Dot Matrix Printer, the second isn't really a printer at all, but a print-formatted text file. If you haven't used up your custom printer definition already, you can have one of these by defining one and selecting "Print onto disk" when AppleWorks asks which slot to access the printer with.

Defining such a printer gives you an extra way to save files to disk.

Important: unless you have done this (and named such a custom printer "formatted text"), the second option in the three screen images shown here will not appear in your own print menus.

In the word processor print menu, note that files can be saved into text files in two different ways. The formatted text file will contain all the control codes that would normally be sent to your printer—such as the control codes for underline and boldface. More importantly, a formatted file will have Returns at the end of each line. The "text file on disk" choice, on the other hand, will eliminate all control codes except for Returns you have entered yourself. Normally this would be only at the end of paragraphs, not at the end of lines. The formatted file is more appropriate for preparing files for transfer by modem. The unformatted file is more appropriate for moving files to other programs such as *Apple Writer*. You do have a choice.

The data base print menu shows you can print reports to the screen and clipboard as well as to formatted, unformatted, and DIF files. The standard "text file on disk" choice gives you a file with a Return after each field. This kind of file is not suitable for entry into a word processor. Use the formatted file option for this, or better yet, print the report to the clipboard. The "text file on disk" choice, however, *is* suitable for dumping information into a file that can be accessed with an Applesoft program. Make up a report format that simply contains all the fields in your data base. You can dump the whole data base or selected (and sorted) portions of it into a standard sequential text file for further processing with Applesoft. The ***Open-Apple*** subscription system is based on this trick.

The spreadsheet print menu shows you can print spreadsheet data to the clipboard, or to formatted, unformatted, or DIF files. Again, just as with data base files, the "text file on disk" option will put a Return after each *cell.* This is unhandy if your intention is to use the file with a word processor (print to a formatted text file or the clipboard instead), but works quite well if your intention is to access the information from an Applesoft program.

Help! I'm rapidly running out of room here, and we've talked about nothing but AppleWorks so far this month. (Note to new subscribers—this isn't typical.) However, there are a couple of sources of AppleWorks help I'd like to mention.

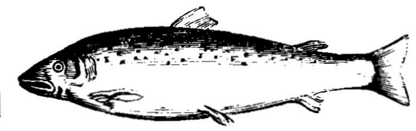
Robert Ericson (P.O. Box 16064, Rumford, RI 02916) has prepared a set of materials he calls *Notes for AppleWorks,* which he sells for $10. These printed notes consist of the equivalent of more than 50 single-spaced AppleWorks pages and are chock full of tips and tricks for using AppleWorks.

The AppleWorks Users Group (c/o Jim Willis, 1300 Hinton St., West Monroe, La., 71291) collects and publishes notes, templates, reviews, and similar material about AppleWorks. The group had eight disk sides of information in May and probably has more by now. You can purchase this information from the group for $4 per double-sided disk, or you can send in a disk *and return postage* and get their stuff at no additional cost. You are encouraged, of course, to put your own ideas, findings, and other material on the disk you send in so your stuff can be added to the club library.

The possibilities here are endless. Rather than simply having a library stuffed full of Basic programs, as most users groups do, this library can have reports and reviews in word processor files; useful, searchable data in data base files; and programs in spreadsheet files. One of the things on the first disk, for example, is a data base file on the group's own membership. Other possiblities would include a data base of software for the Apple, of AppleWorks-related magazine articles, of Apple users groups, and so on. (The sample data base file shown on page 58 comes from one of these disks.)

There's lots more to talk about when it comes to AppleWorks. More next month.

Miscellanea



Apparently the only program ever written that used MouseText's running man was our Humantext Demo in the April issue (page 27). Apple has announced that those characters will not appear in future versions of MouseText and now warns software developers not to use them. Apple hasn't revealed what kind of character will replace the running man.

The MouseText character set was designed by Bruce Tognazzini, an Apple II wizard who has been around since the early days of Apple. He recently wrote a very funny letter to *Call -A.P.P.L.E.* (July 1985, page 41), in which he explains why the MouseText characters ended up where they did (a hardware problem, he says). I'll reprint the letter if I can get the required permissions.

The internal structure of AppleWorks files is completely documented in a paper available for the asking from Debra Hara, Mail Stop-22D, Apple Computer, 20525 Mariani Ave, Cupertino, CA 95014. I wish it used the Apple-compatible dollar sign for hexadecimal numbers. Instead it uses the CP/M—IBM method of following hex numbers with an "H". I keep thinking H is the last digit, but can never find that key on my hexadecimal calculator.

The AppleTalk network's Apple II documentation is available in a preliminary form, though the combination hasn't yet been officially announced. This stuff is expensive, but necessary if you're interested in low-cost networks. First you need *Inside AppleTalk,* which is $75 plus your state's sale's tax from Apple Computer, 467 Saratoga Ave, Suite 621, San Jose, CA 95129. Then you'll need *AppleTalk Developer's Notes for the Apple IIe,* which is $25 plus your state's sales tax from Apple Software Group, Mail Stop 4-T, 20525 Mariani Ave, Cupertino, CA 95014.

The old reliable Disk II has been discontinued by Apple, though equivalent units are still available from other vendors. The Disk II has been replaced by a new half-height single drive called the UniDisk. The UniDisk is aesthetically nicer, but the catch is that the controller card that the UniDisk requires is not compatible with older Disk II controllers. Either controller can accomodate two drives, but Apple says the new UniDisk and old Disk II can't share a common controller card. Of course, Apple also says the Apple IIc can't use either of these drives either, but companies are selling adaptor plugs that allow a Disk II to work with a IIc. (For example, WGE International, Rt 202N, Peterborough NH 03458, 800-227-1560, advertises a Disk II/Apple IIc adaptor plug for $19.95) I don't know what the obstacles are in this case, but **Open-Apple** will pay hard cash for a publishable article on the Disk II/IIc connection and why or why not the Disk II and UniDisk can't share a controller card. Inquire within.

This month's cartoon is by Rich Tennant. Rich used to be a mainframe programmer; he says the experience left him with a gusher of malice that will last the rest of his life. He is coauthor of the *I Hate Computers Book* (Hayden Book Co., ISBN 0-8104-8000-X, $4.95). In a related development, the August ***inCider*** (page 36) includes a worthwhile article by Jack McCornack that shows how to hook a IIc to your car's dashboard or any other 12 volt power source with a few dollars worth of parts. You'll still need a monitor, but if 40-column mode is adequate, McCornack says 5-inch battery-powered televisions are available for about one-fifth the cost of Apple's flat panel display.

Reviewer's Corner

Hackers, Heros of the Computer Revolution. by Steven Levy.
Fire in the Valley, The Making of the Personal Computer. By Paul Freiberger and Michael Swaine.
The Little Kingdom, The Private Story of Apple Computer. By Michael Moritz.

When the president of the Apple user group here in Kansas City goes on vacation, she takes her Apple IIc with her. She drives around the country calling up local bulletin boards to add to her collection. After last year's trip, she suggested that the software, tourism, and hostelry industries get together and "put a sticker on all motel office windows depicting a computer with either an M for modular or an A for acoustic (and for the one motel owner I ran into, a computer with a slashed red circle around it.)"

It's possible, I'm told, to go on vacation and leave the world of computers totally behind, but I have a hard time imagining it. Instead of lugging a computer with you, though, how about a book? Here are three of them—none need electrical outlets—that will keep you from straying too far from the warmth generated by ROMs and RAMs.

All three are historical looks at personal computing with emphasis on the Apple II.

Hackers has the broadest scope. It begins in the late 1950s with the adventures of some students at the Massachusetts Institute of Technology who used one of the world's first transistor-run computers as if it had been an Apple. It ends with an extensive look at the people behind the large, Apple II-centered software company, Sierra On-Line. In between we find Wozniak and the Homebrew Computer Club. *Hackers* focuses on the hardware and software wizards of the computer age and what they stood for.

Fire in the Valley concentrates on the forces that gave us the personal computer. Most of these forces seemed to converge just south of San Francisco, in the Silicon Valley, during the late 1970s. Since Apple is one of the major figures in the history of personal computing, it has a major role in the book. What *Fire in the Valley* did for me was bring together hundreds of loose ends and fragments of things I'd read about but never understood. Things like what's an Altair, who is Dr. Dobbs, and why is CP/M.

The Little Kingdom is an historical look at Apple itself. Here's more than you'd ever want to know about Jobs and Wozniak. But it also introduces other important Apple-related people you've never heard of, such as Rod Holt, Chris Espinosa, Randy Wigginton, Alan Baum, and many others. This book would be better if it gave more of the technological history of Apple, but the stories of stock options and of programmers "calling one of their new supervisors a Software Nazi because he was steadfastly opposed to revealing details about the internal mechanisms of the machine" are very interesting, indeed.

Hackers, by Steven Levy (Anchor Press/Doubleday, Garden City, N.Y.; ISBN 0-385-19195-2). $17.95
Fire in the Valley, by Paul Freiberger and Michael Swaine (Osborne/McGraw-Hill, 2600 Tenth St., Berkeley, Calif. 94710; ISBN 0-88134-121-5).
The Little Kingdom, by Michael Moritz (William Morrow & Co, Inc., 105 Madison Ave, New York, N.Y. 10016; ISBN 0-688-03973-1). $16.95

Ask (or tell) Uncle DOS

Avoiding hi-res memory

I have written a few Basic programs for my elementary school. The better I get at writing these programs (offering more user options, combining tutorial and drill, etc.), the larger the program gets so that it runs into the graphics area. I've broken the last program I wrote into discrete parts and access each part with PRINT D$;"RUN" but this constant waiting for access to the next part is boring and tiresome for the kids. Any suggestions?

Doris M. Kneppel
Kinnelon, N.J.

There are several ways around the problem you are experiencing. They fall into two categories—loading your whole program into the computer but avoiding the graphics area or continuing to use smaller program parts but speeding up the transfer from one part to another.

The easiest way to speed transfer is to use a high-speed DOS, such as **DiversiDOS** *or* **ProntoDOS**, *which are DOS 3.3 work-a-likes, or Apple's newer* **ProDOS**, *which is similar but different. If your school has Apple IIcs or 128K Apple IIes, you could also use the extra available memory as a RAM disk to speed things up. With DOS 3.3 this requires a special software package such as Beagle Bros'* **DiskQuik**. *A RAM disk is installed automatically on 128K machines if you are using ProDOS with Applesoft.*

The other possibility—avoiding the graphics area entirely—can be accomplished fairly easily by simply tricking DOS into loading your program above the graphics area. There are 22,016 bytes of space available above hi-res page one. Below it, where Applesoft programs and variables normally live, there are only 6,144 bytes available. So just this simple trick will allow your program to be three-and-a-half times longer.

What you have to do is poke a zero at the program's new starting location, reset the "start of program" pointers (see figure 1 on page 4 of the January **Open-Apple**), *and immediately RUN a new program. For example, if your program is called BIG PICTURE, write a separate startup program that does this:*

```
10 REM moves program start to $4000
20 START=16384
30 POKE START,0
40 POKE 104,START/256
50 POKE 103,START-(PEEK(104)*256)
60 PRINT CHR$(4);"RUN BIG PICTURE"
```

If you want to put your program above both hi-res pages, change the value of START in line 20 to 24576. That will give you 13,824 bytes of program space—still more than double what you had before. The new program location will continue to be used if you RUN more programs. To reset the location back to normal under DOS 3.3, you can use the FP command. Under ProDOS, rerun Basic.system.

A problem with this trick is that it leaves the memory below the graphics area unused. Another solution to your problem is to split your program into two pieces; one below the graphics area, one above. The Beagle Bros disk **Silicon Salad** *includes a program that will do this; so does* **CALL -A.P.P.L.E. in Depth #1, All About Applesoft** *(page 81) and the June and October 1980 issues of* **Call -A.P.P.L.E.**.

No thank you

I wrote a letter to *Softalk's* **If-Then-Maybe** column six weeks before they folded. Admittedly, it was a trick question, but I think they took it pretty hard.

Brad Walters
Boston, Mass.

Pleeese don't send that question to me.

HCOLOR, block reading

Michael Ching states (June issue, page 47) that on power-up, the value of HCOLOR defaults to white on a II-Plus, but black on a IIe. This may be generally true, but not universally. It appears Applesoft itself does not initialize HCOLOR, so the "default value" is whatever happens to come up in the RAM chips. I was once involved with a commercial product that contained a dismembered II-Plus and did a lot of graphics. Dozens of units were installed with no apparent graphics problem. One day we were surprised to hear from the factory that a particular unit would not display any graphics in program X until you had first used program Y or Z. Investigation finally revealed that program X never set HCOLOR and unwittingly depended on having the II-Plus come up using white; our "defective unit" was an oddball II-Plus that came up using black.

A comment by Ken Kashmarek (July, page 56) makes me think the following suggestion could save a few people possible problems. Kashmarek is doing machine-language routines to access ProFile blocks directly by setting up a command table at $42-$47, then calling $CsXX, where s is the slot number and XX is the value at $CsFF. This may work fine for the ProFile but not for other hard disks; I would recommend instead that the ProDOS disk vector table at $BF10-$BF2F be inspected to find the address of the disk driver routine. I know of at least one hard disk system (Genie) in which the routine at the $CsXX address is a set-up routine; before performing the operation requested in the command table it checks all ProDOS volumes on all drives connected to the card and sets them up appropriately in the disk device tables at $BF10 and up; in particular it modifies the stored vector that led to its being invoked so that the next call to access the volume will go through a normal access routine. In this case repeated calling to $CsXX to access disk blocks wouldn't be particularly harmful, just slow; but conceivably other hard disk systems might tolerate it less benignly. In short, the vector $CsXX only shows where the interface card wants to be called the first time ProDOS uses it; if calling it makes it change its address in the disk vector table, it is good manners to call the latter address.

Robin Ault
Newtonville, Mass.

Thanks for pointing this out. Kashmarek actually wrote me that he accessed the ProFile with JSRs to $C5EA; I took the liberty of changing it to $CsXX so that the information would be useful to people with other types of hard drives. As you point out, however, $CsXX is not necessarily the best entry point, just the first one. For the record, the table you refer to at $BF10 holds 16 two-byte disk device driver entry points in the following order: slots 0 to 7, drive 1; slots 0 through 7, drive 2. The slot 0 entry points are reserved. /RAM appears as slot 3, drive 2. Slot/drive combinations with no device point to an error handler. The ProDOS Technical Reference Manual *talks about this stuff in considerably less detail on pages 92 and 110.*

Integer Basic

How can you run Integer Basic programs under ProDOS?

Robert C Platt
Amarillo, Texas

Compatibility of future machines from Apple is essential, but must they run APPLEVISION as you insist in the July issue (page 50)? For most of us, Integer Basic has been dead for so long that it doesn't even smell bad anymore.

William Cutrer
Dallas, Texas

To run an Integer Basic program under ProDOS you would need a system program somewhat like Applesoft's Basic.system. "Integer.system", or whatever you might call it, however, would have to hold the entire Integer Basic interpreter as well as the DOS related stuff of Basic.system. I don't think developing such a thing would be a massive undertaking as long as Apple would license you to steal code from Integer Basic and Basic.system.

I don't think there would be much remuneration in it, but writing "Integer.system" would make a nice hobby for some assembly language hacker. To my knowledge it hasn't been done yet.

Insisting that future Apple IIs can run APPLEVISION simply guarantees that a number of things stay the same with Apple graphics, sound, disk, and memory configurations. Though it hurts to say such a thing about the first computer language I ever learned, I agree that Integer Basic is as dead as Latin.

ProDOS drive size

The ProDOS read/write block routines require a unit number in the format DSSS0000, where D is the drive number and SSS is the slot. Is it logical to assume the drive number of a hard drive is 1?

I write a computer newsletter for my employer, a county schools office. I'd like to occasionally pass on information mentioned in **Open-Apple**. Do you have any objections as long as I give **Open-Apple** as the source?

Bob Crocker
Yuba City, Calif.

If a hard drive supports just one volume, the read/write block routines treat that volume as drive 1 (D=0). If the drive has two volumes, the second is treated as drive 2 (D=1).

ProDOS allows a maximum of 14 volumes to be accessible, or "on-line" at one time. Each of these can have up to 65,536 blocks (33,554,432 bytes) of storage space, for a total of about 4.7 billion bytes. (A shelf of **Open-Apples** *with that much data would be over 12 feet long.) By using "removable media", such as a floppy disk or a more automated type of device, any amount of information could be handled by ProDOS.*

Theoretically, all 14 volumes could be on one physical device attached to a card in a single slot, however, the ProDOS kernel would pretend they were 14 separate devices in separate slots.

Here are the "unit" or "device" numbers that **DSSS0000** *translates into:*

```
ProDOS unit or device numbers

  slot:   1    2    3    4    5    6    7
drive 1  $10  $20  $30  $40  $50  $60  $70
drive 2  $90  $A0  $B0  $C0  $D0  $E0  $F0
```

In some situations, ProDOS uses the lower four bits to designate the type of device. For example, in the "device list" (DEVLST) ProDOS keeps in the system global page at $BF32-3F, 0 indicates a Disk II-like floppy, 4 an Apple ProFile, and F indicates /RAM. The Sider shows up as a 5.

Ideas expressed in **Open-Apple** *are open game for anyone, just as they are in any written material. If you cite* **Open-Apple** *as the source of such ideas, however, your newsletters will fold easier and your stamps will stick better.*

The exact wording of the ideas expressed in **Open-Apple** *is copyrighted. The legal doctrine of "fair use" says you can use short portions of my wording if you indicate that the material is a direct quote and you cite the source. I do this to other people all the time; look around* **Open-Apple** *for examples. I will also allow Apple user groups and significant others to quote whole articles from* **Open-Apple** *on a word-for-word basis from time to time, but this is more complicated and requires a specific written request.*

Truly mixed

Is there any way to change the number of text lines on a mixed text and graphics page? Can I use a screen that is half and half?

Duncan Orr
Heber City, Utah

Indeed, you can, using assembly language on a IIe or IIc. It can also be done on a II-Plus, but you have to solder a 7-1/2 inch piece of wire between a couple of pins. The details of all this would fill up a whole issue of **Open-Apple**. *Instead, I'll refer you to the bible on this technique, Don Lancaster's* **Enhancing Your Apple II, Second Edition** *(Howard W. Sams & Co., 4300 West 62nd St, Indianapolis, IN 46268, $15.95). If you have an Apple IIe or IIc, start on page 232, where notes on using the techniques with these machines are found, then go back and examine chapters 4, 5, and 6.*

Most people don't believe it, but with Lancaster's technique you can mix any or all Apple II graphics modes on one screen—anyplace, even left-right splits. The half-text/half-graphics you're interested in is just the beginning of the possibilities.

Gorilla-Grappler cross

Can you tell me how to convince my Apple II-Plus to have my Grappler-Plus tell my Gorilla printer to print graphics? I've tried the control-I, G command, but it doesn't work.

Paul Low
Riverdale, Md.

If you don't get anything at all on your Gorilla when you print the control-I, G, you're probably not doing it right. The following program has an example of how to do it.

If you get garbage on your Gorilla in response to control-I, G, you need to reset the dip switches that tell your Grappler-Plus what kind of printer you have. The Grappler manual doesn't specifically say it supports the Gorilla, but the Gorilla probably prints graphics in response to the same commands as one of the supported printers. If you know something about this, it may help to review the Grappler-Plus manual.

Otherwise, proceed like this. Put the following startup program on a disk that has a printable graphic on it:

```
10 HGR
20 PRINT CHR$(4);"BLOAD GRAPHIC, A$2000"
30 PRINT CHR$(4);"PR#1"
40 PRINT CHR$(9);"G" : REM control-I, G
50 PRINT CHR$(4);"PR#0"
60 TEXT : END
```

Turn off your computer. Remove the lid and find the dip switches on your Grappler. There are four of them, so they can be set in 16 possible combinations. Start trying the combinations one-by-one until you find the one that works. Set the switches in your first configuration and turn the computer on to start up the above program. Check what appears on your printer. If it's a graphic, you're done. If not, turn the computer off, try another combination, and repeat. This method can be used to discover the right combination for any interface card and printer—if there is one.

The hard disk life

I lost 16,000 sectors on a Sider when I foolishly reset when printing from *Apple Writer* (ProDOS version). It bad blocked them and I couldn't work around it—had to reformat and lost many files I should have backed up! Any hints?

Donald Beaty
San Mateo, Calif.

Boy that's scary. I can think of no theoretical reason why pressing reset while printing from **Apple Writer** *should harm the Sider, but I'm not willing to try it on my system to make sure.*

Reset trap bug

There is a bug in the Reset trap demonstration program in the February issue (page 16). It prevents a disk from booting correctly if the patch is made a permanent part of DOS 3.3. Add this line to the program to get rid of the problem:

```
57 POKE 40382,21
```

BSAVE, T, and CREATE

Your July issue (page 51) correctly states that BSAVE, when used with the Type parameter, works only with pre-existing files. However, there seems to be no reason for this and I consider this inability a bug in Basic.system.

I discovered the bug while implementing FIG-FORTH in ProDOS. I wanted to save data blocks using as little code as possible, however I wanted the data blocks to use standard text files so they could be edited with word processors. I recognized that BSAVE could solve my space problem, but to use text files I would have to CREATE new files first, thus decreasing speed and increasing code length once again.

Therefore I looked for the BSAVE code to fix the bug and, with the help of the disassembly in the supplement to *Beneath Apple ProDOS*, found I could do it. I was able to accomplish this because the command parser puts the file type code in VTYPE ($BE6A) and then BSAVE and CREATE both place it there again, wasting code that I put to better use. Here are my patches:

```
bload  relocated          disassembly
 adr      adr            and comments

Basic.system 1.1 existing BSAVE code

$37F5   $ADF5   A9 06      LDA #$06  assume BIN
$37F7   $ADF7   8D 6A BE   STA $BE6A put at VTYPE
$37FA   $ADFA   8D B8 BE   STA $BEB8 put at setinfo
$37FD   $ADFD   AD 56 BE   LDA $BE56 type given?
$3800   $AE00   29 04      AND #$04
$3802   $AE02   D0 0E      BNE $AE12 yes--error
$3804   $AE04   20 46 AD   JSR $AD46 goto CREATE

$3741   $AD41   A9 0F      LDA #$0F  assume SYS
$3743   $AD43   8D 6A BE   STA $BE6A put at VTYPE
        $AD46   ------->             CREATE file

modifications for BSAVING new files of any type

$37F5   $ADF5   AE 6A BE   LDX $BE6A get parsed type
$37F8   $ADF8   AD 56 BE   LDA $BE56 type given?
$37FB   $ADFB   29 04      AND #$04
$37FD   $ADFD   D0 02      BNE $AE01 yes, branch
$37FF   $ADFF   A2 06      LDX #$06  no, use BIN
$3801   $AE01   8E B8 BE   STX $BEB8 put at setinfo
$3804   $AE04   20 43 AD   JSR $AD43 goto CREATE-3

$3741   $AD41   A2 0F      LDX #$0F  assume SYS
$3743   $AD43   8E 6A BE   STX $BE6A put at VTYPE
        $AD46   ------->             CREATE file
```

Mark Jackson
Chicago, Ill.

Your modifications look very good, particularly since they require no additional space. In testing them, however, I noticed that the record length for text files created this way is set wrong. Directory entries under ProDOS include a field called AUX_TYPE that is used for different things depending on the type of file. Files of types BIN, BAS, SYS, and VAR use it to store the default loading address of the file. This is the number that BSAVE puts in the directory with your technique.

TXT files, however, store the file's record length in AUX_TYPE. Thus, TXT files created with BSAVE have very strange record lengths. I suspect Basic.system prevents BSAVE from creating new non-binary files to avoid problems with AUX_TYPE. However, your modifications can be valuable if used with an understanding of the potential problems.

Incidentally, a subscriber recently tipped me off to the most interesting use of AUX_TYPE to date. AppleWorks files use AUX_TYPE as a bit mask that remembers which characters of a file's name are upper case and which are lower case. When you catalog an AppleWorks file with Basic.system, it will always appear in capitals, with periods instead of spaces, just like all other ProDOS filenames. From inside AppleWorks, however, file names can be upper and lower case and can include spaces as well as periods. The 16-bit AUX_TYPE value tells AppleWorks how to display the 16 character filename. Lower-case periods are displayed as spaces.

Nibble bytes tounge

Shocked by your expose (June, page 45) of our alleged hoarding of available exclamation points, I asked our controller to do an internal audit of our stocks! Much to my surprise, it divulged that there was a large crate of said points sequestered in a back stockroom, apparently an impulsive acquisition by a former sales manager! Appalled, I contacted the MPA, the ABA and the SMPA, who will independently oversee a lottery for point distribution to needy publishers, based on their immediate exclamatory needs! Hopefully, we will see a resultant softening of the point market, although this may in turn affect the availability of hypens—and ellipses...

David Szetela
Managing Editor, *Nibble*

PEEK(978) problems

This 57-year old has been struggling through **Open-Apple** and, believe it or not, some of it seems to rub in. Fantastic stuff.

The program for testing the effects of the Garbageman in the January issue (page 4) gave me a problem. Line 85 indicates that if PEEK(978)=190 then ProDOS is active, but that's also what you get when using DOS 3.3 and GPLE.DM (the version of GPLE that works with DOS 3.3 moved to the language card).

I've noticed you are also using PEEK(978) in other programs, and I thought I'd warn you you may have a problem with GPLE users. I have never learned to type in code without using GPLE.

Will Thompson
Buffalo Grove, Ill.

*Things transpire within **Open-Apple** at different levels. I try to fix it so that anybody interested in the Apple II can pick up some worthwhile stuff—but less experienced users will undergo more puzzlement than others. But remember—the more you read, the more you understand; the more you understand, the more you understand.*

PEEK(978) serves two different functions. In some programs, such as the one you mention in the January issue, it is used to determine whether DOS 3.3 or ProDOS is active. In others, it is used to confirm that DOS 3.3 is at its standard 48K location.

PEEK(978) usually works for these tests because standard Apple operating system protocol calls for a DOS warmstart "vector" to be stored in bytes 976-978 ($3D0-3D2). Byte 978 ends up holding the "page" or high-byte address of a routine inside DOS.

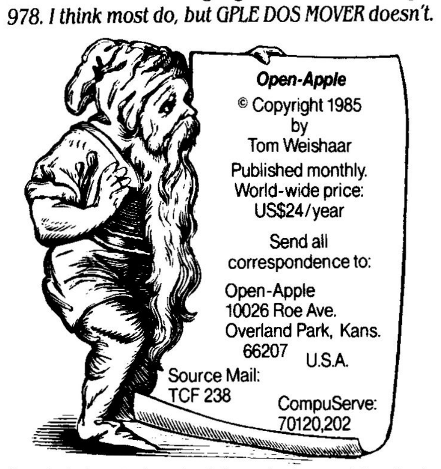
Here's what the value inside byte 978 usually indicates:

Situation indicated by value in byte 978 ($3D2)

<157	DOS 3.3 active, computer has < 48K
157	DOS 3.3 active at standard 48K position
190	ProDOS active
191	DOS 3.3 active on language card

However, as you point out, not all programs that move DOS 3.3 to the language card leave 191 in byte 978. I think most do, but GPLE DOS MOVER doesn't. Consequently, the DOS 3.3 vs ProDOS test fails. You GPLE.DM addicts may find a different test more appropriate—check byte 977 instead. ProDOS stores a zero here; DOS 3.3, on the other hand, keeps something greater than zero here in all situations I know of.

When programs check to see whether DOS has been moved, however, only the test of byte 978 will work. You will have to discern yourself which programs need to have 978 changed to 977 and which should be left at 978. All programs that poke changes into DOS 3.3 should check 978 to make sure DOS is where it's supposed to be. If it isn't, the pokes go into thin air and have no effect, or worse, have a drastic effect.



Open-Apple
© Copyright 1985
by
Tom Weishaar
Published monthly.
World-wide price:
US$24/year
Send all
correspondence to:
Open-Apple
10026 Roe Ave.
Overland Park, Kans.
66207 U.S.A.
Source Mail:
TCF 238
CompuServe:
70120,202

Converting CONVERT for MouseText

I have an enhanced IIe and also a IIc. Using either of these systems with the ProDOS CONVERT utility causes the inverse bar that highlights filenames to show the names in MouseText. I finally got tired of this and went looking for a solution to the problem.

The following Applesoft routine fixes the problem.

```
10 PRINT CHR$(4);"UNLOCK CONVERT"
20 PRINT CHR$(4);"BLOAD CONVERT,TSYS,A$2000"
30 POKE 26523,14
40 POKE 26711,63
50 PRINT CHR$(4);"BSAVE CONVERT,TSYS,A$2000,L20481"
60 PRINT CHR$(4);"LOCK CONVERT"
```

For the technically curious, line 30 fixes an initialization problem. The author of the code enabled the alternate character set instead of disabling it. Without line 40, the highlighted filename flashes. If one likes that feature, don't bother with line 40.

Cecil Fretwell
Waterloo, Iowa

Blazing the upgrade trail

I read with great interest your comments in the July **Open-Apple** (page 49) on compatibility between future members of the Apple II product family and existing products. Having spent a great deal on hardware and software for my system at home—so much, in fact, that the majority of Apple II series owners would probably consider it impossible, and a smaller, but still significant amount of my employer's money on a system at work, I have given the subject a great deal of thought.

One of the first things Apple had better realize is that customers will make buying decisions based on what they *perceive* as best for them. While it may hurt our pride to admit this, decisions on the purchase of personal computers, peripherals, accessories, and software are rarely rational. What is being purchased is sizzle. It is only after the purchase that one learns whether the sizzle was accompanied by steak.

If potential customers perceive that Apple's product offerings are intended to address the company's needs rather than customers' needs, the products will not be well received. In the case of future products in the II family, the primary customer base has to be the millions of owners of II series computers. If new products sell well to this audience, the continued vitality of the line is virtually assured, and it will sell to a wider audience.

There is little doubt that 3-1/2 inch disks will ultimately displace 5-1/4 inch disks in the personal computer marketplace. There is also little doubt that Apple would like to take advantage of the economies of scale associated with the use of a single type of disk drive in both of its product lines. There is undoubtedly also a line of reasoning that says that if new computers (e.g. those based on the 65816 microprocessor) capable of running new and more powerful software are compatible only with a new diskette standard, users of the existing products will upgrade. My response to this is that we will upgrade to Apple products *if and only if* the upgrade path offered us allows us to preserve most of our existing investment.

In my own case, I do not have room for a second machine and all the slots in my II-Plus are filled. (Actually, several cards are not in slots of their own, e.g. the Novation Apple CAT II 212 upgrade card is mounted atop the power supply, Applied Engineering Z-80-Plus and Timemaster II H.O. cards share slot 7 thanks to a Legend Industries "Slot 8", and a Videx Enhancer II replaces the original keyboard encoder.) When software starts to become available, I would be very interested in acquiring a 65816 co-processor, a hard disk and 3-1/2 inch drives, but I want these to be in a form that does not cause me to choose between my present equipment and a new set.

My proposal is that these items be offered in a form similar to the ill-fated Rana 8086/2 co-processor, which was supposed to provide MS-DOS compatibility for the II-Plus and IIe. That product apparently failed due to a combination of poor execution at the detail level and under-capitalization of its manufacturer. (At last report, Rana was attempting to reorganize under Chapter 11.)

I propose a unit that mounts atop an Apple II-Plus or IIe in the same manner as Apple's Duo-Disk unit. It would interface to the Apple bus via a card that would plug into slot 6 in place of the Disk II controller. The unit would contain the co-processor, its ROM and RAM (probably up to 1 megabyte) and sockets for chips that could be inserted to add hard disk, floppy disk, and RGB video interfaces. It would be capable of containing (at the user's option) one or two 3-1/2 inch floppy drives and a hard disk of at least 10 megabyte capacity. There would be connectors at the rear for a mouse, an external keyboard, and two Disk II drives. Switches or jumpers would determine whether on boot the system would search the 5-1/4 inch, 3-1/2 inch, or hard drive first. There would be both input and output connectors for an NTSC composite video signal. A video softswitch would allow switching between composite video passed through from the Apple II or video generated by the 65816. In all probability, at least part of the power supply would have to be external, as in the IIc, due to dimensional constraints.

If all this sounds suspiciously like a description of a self-contained microcomputer with limited expansion capability, it is! The important thing about it is that it would be designed to work with an Apple II. Those who wanted to use it as a stand-alone unit could purchase a separate keyboard and an expansion chassis to contain peripheral cards. These could be Apple II peripheral cards, or borrowing (and improving on, one would hope) an idea from the IBM PC/AT they could be cards with two sets of edge fingers. The first set would contain the same signals as the existing bus, while the second would contain additional signals. On the motherboard, these could be inserted into a single connector of stepped height, with the existing bus on the upper level. This would permit use of existing peripheral cards, while also allowing for more advanced equipment.

My goal for the selling price of the basic unit with no disk drives or optional chips and 256K of RAM on board would be $695.

Dan Strassberg
Arlington, Mass.