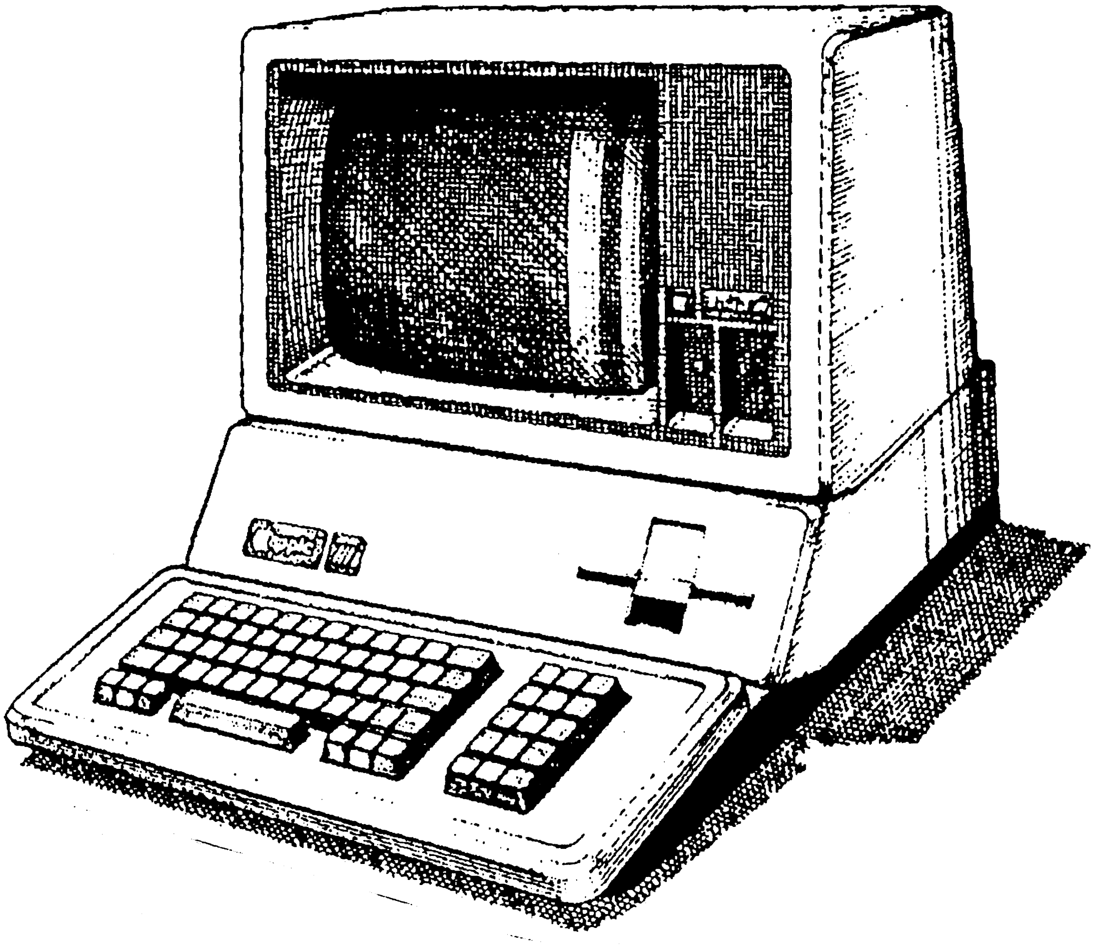
Apple /// Computer Information



DOCUMENT NAME

APPLE /// BYTE REVIEW (SEPT. 1982)

#

12

Ex Libris David T. Craig

"_36.PICT" 713 KB 2001-08-22 dpi: 600h x 600v pix: 4391h x 5965v

The Apple III and Its New Profile

An in-depth look at the "new" Apple III microcomputer and its Profile hard disk.

Robin Moore
Warner Hill Rd. RFD #5
Derry, NH 03038

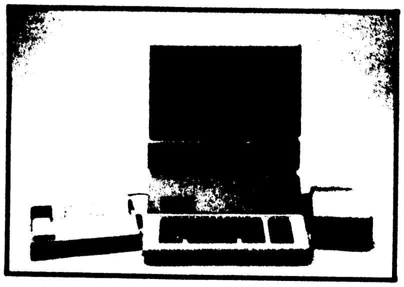


Photo 1: *A view of the Apple III, the Profile hard-disk drive, and the Monitor III showing a sample of Visicalc III on the screen.*

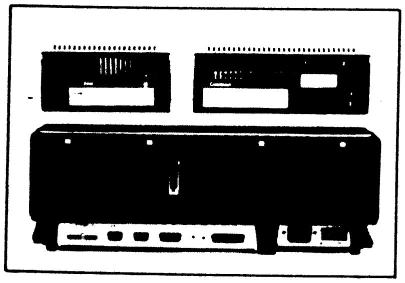


Photo 2: *A rear view of the Apple III and Profile showing the Silentype and game paddle ports A and B, along with the video, audio, RS-232C, and floppy-disk connectors. The peripheral card visible is the Profile interface card.*

In 1980 when the Apple III was first released, there were problems. Deliveries were delayed, and when the machines finally arrived, they often didn't work. The integrated circuits tended to wander out of their sockets. Little software except Visicalc was available, and the much-promoted real-time clock/calendar didn't work well. The Apple III was, on the whole, unreliable. It was a bad start.

Now, in 1982, the problems are gone. The sockets have been changed and the software bugs fixed. The Apple III has been rereleased with revised software, Pascal, and a brand-new peripheral—the Profile, a 5-megabyte hard-disk drive. The new Apple III is an impressive machine and certainly a contender for the title of Best Personal Computer in the less than $10,000 class.

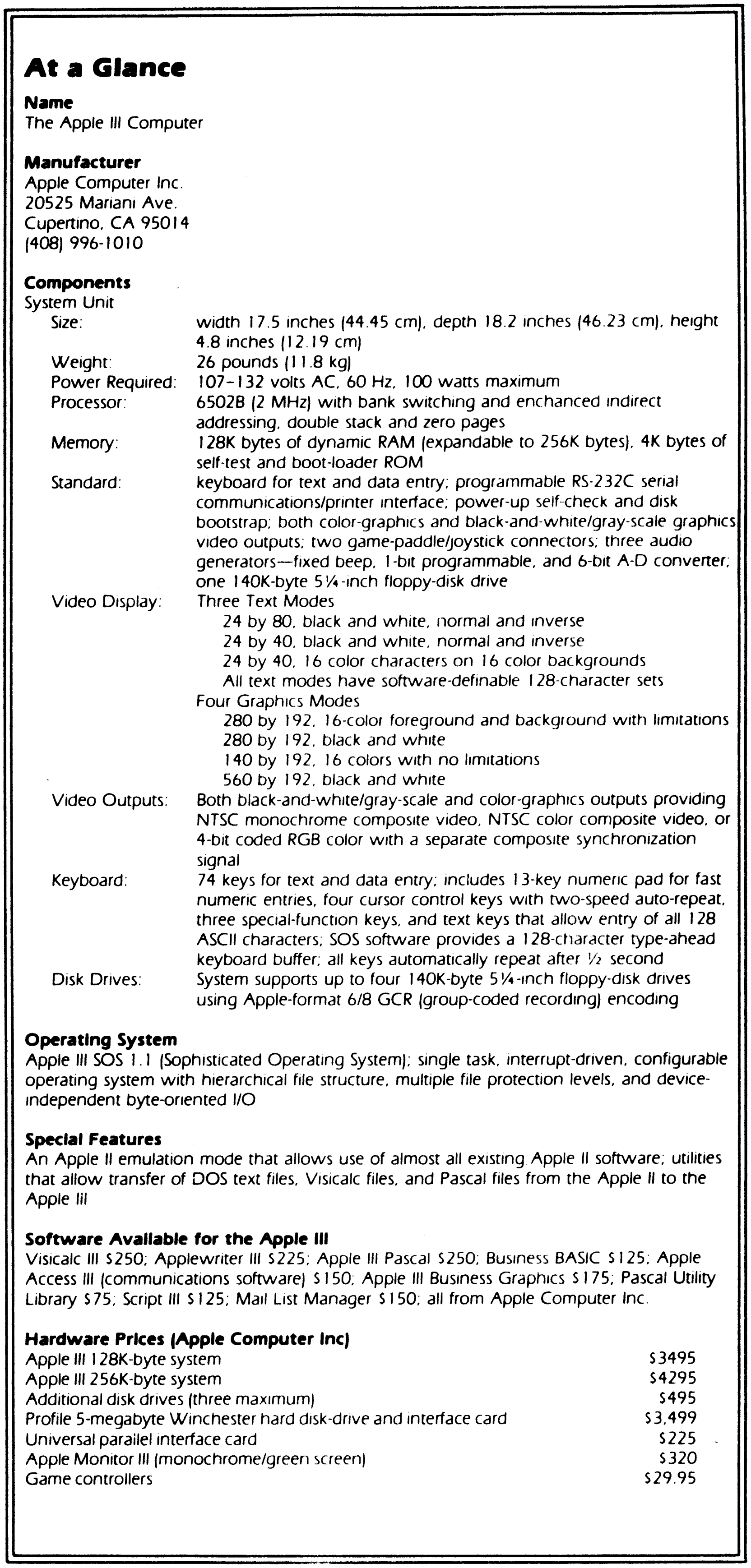
System Overview

Let's take a closer look. The Apple III is a single unit that includes the central processing unit, keyboard, memory, floppy-disk drive, and video output (see photos 1 and 2). It has been designed to meet the needs of the professional or small-business user. Instead of offering an initial low-cost unit requiring a number of additions, Apple Computer Inc. has included the most common system expansions as standard in the Apple III. These include an enhanced keyboard, a 24-row by 80-column display, an integral disk drive, 128K bytes of memory, a programmable 128-character set, improved high-resolution graphics, and an Apple II emulation program (see the At a Glance box for additional features and details).

In addition, several peripherals are available for the Apple III. The most impressive of these is the Profile, Apple's new 5-megabyte hard-disk drive. (The Profile will be described in detail later in this article.) Other options from Apple Computer in-

About the Author

Robin Moore is manager of microprocessor development for A. B. Dick Co. and maintains a strong interest in FORTH, graphics, and computer music. He is also librarian for the Southern New Hampshire Apple Core.



At a Glance

Name
The Apple III Computer

Manufacturer
Apple Computer Inc.
20525 Mariani Ave.
Cupertino, CA 95014
(408) 996-1010

Components
System Unit

Size: width 17.5 inches (44.45 cm), depth 18.2 inches (46.23 cm), height 4.8 inches (12.19 cm)

Weight: 26 pounds (11.8 kg)

Power Required: 107-132 volts AC, 60 Hz, 100 watts maximum

Processor: 6502B (2 MHz) with bank switching and enchanced indirect addressing, double stack and zero pages

Memory: 128K bytes of dynamic RAM (expandable to 256K bytes), 4K bytes of self-test and boot-loader ROM

Standard: keyboard for text and data entry; programmable RS-232C serial communications/printer interface; power-up self-check and disk bootstrap; both color-graphics and black-and-white/gray-scale graphics video outputs; two game-paddle/joystick connectors; three audio generators—fixed beep, 1-bit programmable, and 6-bit A-D converter; one 140K-byte 5¼-inch floppy-disk drive

Video Display: Three Text Modes
- 24 by 80, black and white, normal and inverse
- 24 by 40, black and white, normal and inverse
- 24 by 40, 16 color characters on 16 color backgrounds
- All text modes have software-definable 128-character sets

Four Graphics Modes
- 280 by 192, 16-color foreground and background with limitations
- 280 by 192, black and white
- 140 by 192, 16 colors with no limitations
- 560 by 192, black and white

Video Outputs: Both black-and-white/gray-scale and color-graphics outputs providing NTSC monochrome composite video, NTSC color composite video, or 4-bit coded RGB color with a separate composite synchronization signal

Keyboard: 74 keys for text and data entry; includes 13-key numeric pad for fast numeric entries, four cursor control keys with two-speed auto-repeat, three special-function keys, and text keys that allow entry of all 128 ASCII characters; SOS software provides a 128-character type-ahead keyboard buffer; all keys automatically repeat after ½ second

Disk Drives: System supports up to four 140K-byte 5¼-inch floppy-disk drives using Apple-format 6/8 GCR (group-coded recording) encoding

Operating System
Apple III SOS 1.1 (Sophisticated Operating System); single task, interrupt-driven, configurable operating system with hierarchical file structure, multiple file protection levels, and device-independent byte-oriented I/O

Special Features
An Apple II emulation mode that allows use of almost all existing Apple II software; utilities that allow transfer of DOS text files, Visicalc files, and Pascal files from the Apple II to the Apple III

Software Available for the Apple III
Visicalc III $250; Applewriter III $225; Apple III Pascal $250; Business BASIC $125; Apple Access III (communications software) $150; Apple III Business Graphics $175; Pascal Utility Library $75; Script III $125; Mail List Manager $150; all from Apple Computer Inc.

Hardware Prices (Apple Computer Inc)

Apple III 128K-byte system	$3495
Apple III 256K-byte system	$4295
Additional disk drives (three maximum)	$495
Profile 5-megabyte Winchester hard disk-drive and interface card	$3,499
Universal parallel interface card	$225
Apple Monitor III (monochrome/green screen)	$320
Game controllers	$29.95

"_38.PICT" 571 KB 2001-08-22 dpi: 600h x 600v pix: 2912h x 5953v

Apple III (list prices)	
128K-byte system unit with integral 140K-byte 5¼-inch floppy-disk drive, Apple SOS operating system software, both color-graphics and black-and-white/gray-scale video outputs, RS-232C serial interface, game control port, and Silentype printer interface	$3495
additional floppy-disk drive (three maximum)	$495
Apple Business BASIC software	$125
total	$4115
IBM Personal Computer (suggested retail prices)	
48K-byte system unit, disk-adapter card, one 160K-byte floppy-disk drive, DOS software, Disk BASIC	$2235
16K bytes of added memory and game adapter card	$145
additional floppy-disk drive (one maximum)	$570
serial RS-232C interface card	$150
additional 64K-byte memory card	$540
color-graphics video adapter card	$300
Microsoft extended BASIC software	$40
total	$3980

Table 1: *Price comparison of comparable versions of the Apple III and the IBM Personal Computer. Both systems include 128K bytes of memory, two floppy-disk drives, color-graphics video output, serial RS-232C interfaces for Qume (or equivalent) letter-quality printers, and game-paddle adapters. The system chosen is one that might be purchased by people who wish to combine business and personal applications. Note that in this configuration the IBM has used up all its expansion slots, while the Apple III still has all four of its slots left for further expansion.*

clude the Silentype thermal printer, additional floppy-disk drives, the monochrome green-screen Monitor III, a universal parallel I/O (input/output) interface card, and game controllers.

Many of the existing Apple II interface cards will work in an Apple III while in the Apple II emulation mode. However, use of Apple II cards in an Apple III will probably make it exceed FCC (Federal Communications Commission) radio-frequency radiation limits and may cause interference on nearby television sets or radios. In addition, Apple II cards are not compatible with Apple III software unless special *device-driver* routines are written, and Apple provides virtually no information on how to write them.

Apple Computer currently provides a variety of software packages for the Apple III in addition to Business BASIC and Apple Pascal. There are also various hardware and software products available for the Apple III from other vendors and the number of these will increase as the Apple III user community grows.

The only software built into the Apple III is a 4K-byte ROM (read-only memory) that holds power-up self-test and disk bootstrap routines. All other software is loaded from disk. Although this means that languages use up some of the available RAM (random-access read/write memory), it also allows easy software upgrades and fixes that would be more difficult if the software were permanently in ROM.

System Pricing

The approach to Apple III pricing is almost directly opposed to the pricing strategy used for the Apple II and the IBM Personal Computer. Because Apple chose to include a large number of standard features, the Apple III has a relatively high initial cost ($3495); however, it can expand to 256K bytes of memory, four floppy-disk drives, and a letter-quality Qume (or equivalent) printer without using *any* of the expansion slots. A fully usable system can be configured by adding just a video monitor and an inexpensive serial printer.

Table 1 shows a price comparison of the Apple III and the IBM Personal Computer. Both systems are configured with 128K bytes of memory, two floppy-disk drives, a serial RS-232C printer interface, color-graphics video outputs, and game controllers. The IBM system costs slightly less but uses all of its expansion slots, while the Apple III still has its four slots available for future growth.

The Apple III User

A look at the documentation and software supplied with the system will quickly reveal that the Apple III is targeted for professional and small-business users. Clear tutorials and example programs on disk demonstrate most system functions and features. There is even a two-disk program to lead you through the keyboard and display functions step by step.

The Apple III is not designed for the home hobbyist. Much of the technical information included with the Apple II is absent in the Apple III package. There is no discussion of bus structure, I/O addressing, memory usage, or screen-memory mapping. There are no listings published for any of the system software, either in the Apple III ROMs or on disk. Apple does not even tell you about the monitor program included in the ROMs (which is accessible by holding the Control and Open-Apple keys while pressing Reset).

All this technical information is unimportant to business users. They are more interested in *using* the Apple

Circle 214 on inquiry card. →

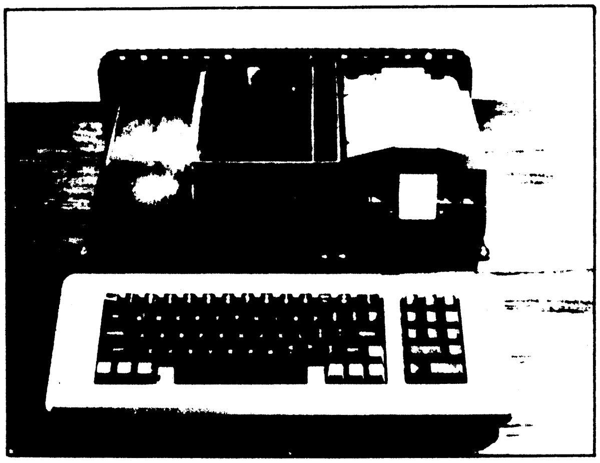


Photo 3: *The Apple III with its main cover removed. The power supply is housed in the enclosure visible to the left, I/O card slots are in the center, and the disk drive is on the right. The entire Apple III is built around a single thin-wall aluminum casting that provides both support and shielding.*

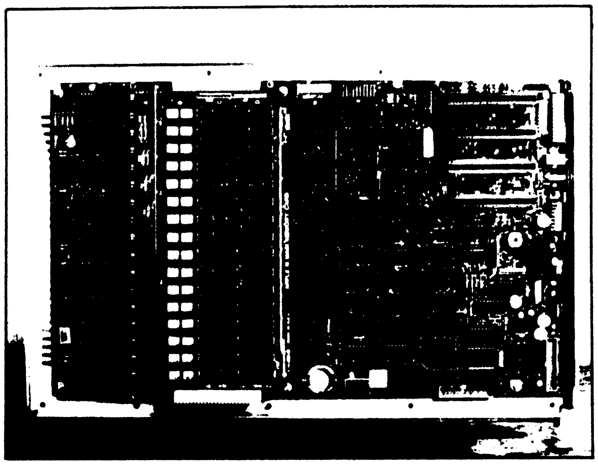


Photo 4: *The Apple III main PC board. The piggy-back-mounted board to the left of center is the removable main memory board. Using this board, the Apple II can be expanded to its full memory capacity without using up any of its I/O expansion slots.*

III than in dissecting it, and will, in most cases, use commerical software. The Apple III is admirably designed to serve their needs. For hobbyists there are better choices, namely, the Apple II.

Inside the Enclosure

The Apple III is a fine example of a quality product designed for high-volume production. The entire unit is built around a single thin-wall aluminum casting that provides support and shielding as well as heat dissipation so that no cooling fan is required. The expansion card guides are molded into the casting, and fully enclosed boxes are built in for both the main printed-circuit (PC) card and the switching power supply (see photo 3).

All of the circuitry, except memory, is on one main PC board (see photo 4). The system memory board mounts piggy-back style onto the main board and avoids taking an expansion slot. In fact, the Apple III can be expanded to its full 256K-byte memory capacity in the same fashion, leaving all slots free.

The Apple III central processing unit is based on a 6502B microprocessor with custom external circuitry that provides a number of enhancements to the normal 6502 instruction set. These enhancements include expanded addressing range, alternate stack and zero pages, and improved indirect addressing that is supported by a separate pointer page.

Although the technical information provided by Apple is somewhat vague, apparently the 6502B is run at 2 MHz during the video blanking intervals and at 1 MHz while the beam is writing information onto your monitor screen. This provides an average speed of about 1.4 MHz, but the screen can be turned off temporarily during program execution to allow the processor to run at its full 2-MHz speed, if desired.

The Apple III can be configured to 256K bytes without using a single expansion slot.

While a normal 6502B can address a maximum of 64K bytes of memory, the Apple III uses bank switching to expand this range to a theoretical maximum of 512K bytes.

Up to fifteen 32K-byte blocks of memory can be switched to occupy the range of addresses between 2000 and 9FFF hexadecimal. This switching is handled automatically by the operating system and is totally "transparent"; that is, the switching executes in the background without affecting any task you may be performing in the visible foreground. It should be noted that, to date, Apple Computer has not announced any Apple III memory expansion beyond 256K bytes. Perhaps this will be a future option.

The main PC board also includes the disk controller, serial interface, video generation circuitry, and the expansion card slots. The expansion-bus connections in the Apple III are essentially the same as those in the Apple II, although DMA (direct memory access) is handled somwhat differently. The Apple III *Owners Manual* provides no information about the expansion bus. Hopefully, this type of information will be available in the future. There are few competing systems that do not make this sort of information available to the public.

The Keyboard

Experienced typists should find the Apple III keyboard easy to use (see photo 5). Unlike the Apple II, this keyboard has a typewriter layout so that touch-typists should feel comfor-

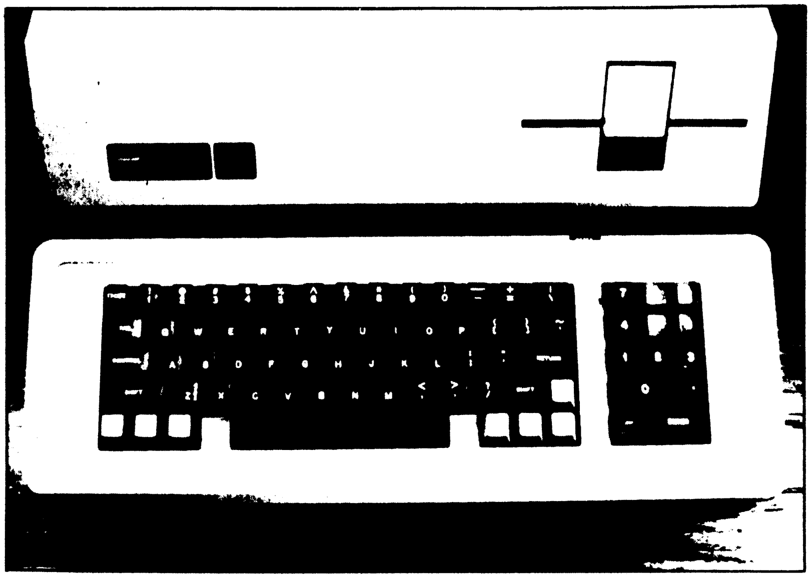


Photo 5: *The Apple III keyboard. Although it looks separate, it is actually part of the Apple III main enclosure.*

table with the key placement. The layout of the numeric keypad on the right, which resembles that of a calculator, allows easy entry of numeric data. The Apple III can also generate all 128 ASCII (American Standard Code for Information Interchange) codes without extra hardware.

In addition to the normal Shift, Control, and Caps-Lock modifier keys, the Apple III includes special Open-Apple and Close-Apple keys that you can define for special functions. All keys automatically repeat when pressed for more than ½ second, and the four cursor-movement keys each provide a 2-speed repeat—pressing gently repeats at 11 Hz, while pressing firmly repeats at 33 Hz.

Apple's SOS 1.1 operating system provides a 128-character type-ahead buffer so that keystrokes won't be lost if you continue to type while the system is busy. This buffer may be emptied, or flushed, if the program running needs to wait for a particular keystroke.

One of the biggest complaints about the original Apple II concerned the close proximity of the Reset key to the rest of the keyboard. In the Apple III the Reset key has been positioned at the rear edge of the keyboard enclosure, thus avoiding the accidental resets encountered in early Apple IIs. Simultaneously pressing Control and Reset simulates a power-up and reboots the system from the main disk drive.

In addition to the normal keyboard functions, a number of special control features are built into the Apple III keyboard. Pressing the Control key and one of the keys on the numeric pad will allow you to turn the video on and off, flush the type-ahead buffer, suspend screen output so that the processor can run at maximum speed, display control characters, or turn off the screen until the program requests an input.

In general, I found the keyboard versatile and pleasant to use. (Although the keyboard is actually part of the main enclosure, it is styled to appear as a separate unit. A convenient recess at the top can support a book or a pencil.) My only problem was that the very light touch required to avoid automatic key repeat sometimes caused me to produce extra characters. You have to break the habit of letting your hands rest on the keyboard while thinking about what to type next.

Mode	Format	Colors
0	24 by 40	black and white
1	24 by 40	16 foreground and 16 background colors
2	24 by 80	black and white

Table 2: *Apple III text display modes, screen formats, and color capabilities.*

Color	Color Value	ASCII Character	Gray Level
black	0	0	black
magenta	1	1	
dark blue	2	2	
lavender	3	3	
dark green	4	4	dark gray
gray	5	5	
medium blue	6	6	
light blue	7	7	
brown	8	8	medium gray
orange	9	9	
gray 2	10	:	
pink	11	;	
green	12	<	light gray
yellow	13	=	
aqua	14	>	
white	15	?	white

Table 3: *Table of graphics colors or gray levels produced by the GRAFIX driver routine. After opening the routine as an output device, colors may be selected by printing a CHR$(9) followed by an ASCII character. The color values shown are extracted from the lower four bits of the ASCII code transmitted. Higher-level graphics functions are provided by the BGRAF invocable module.*

Display Modes

The Apple III offers several text and graphics display modes. Either type of display is available in black and white or color, and both offer various formats and resolutions.

The normal text display is black and white, with a 24-row by 80-column format and a maximum of 1920 displayed characters. Alternate modes include 24 by 40 black and white and 24 by 40 color. In all three text modes the characters are normally displayed as a 5- by 7-dot matrix within a 7- by 8-dot character cell. However, all 128 characters are user-programmable and may be defined to be 7 dots wide by 8 dots high so that adjacent characters will touch in all directions if desired. (See table 2 for available text display modes.)

In the 40-column color-text mode, you can display 16 colors of characters on 16 colors of background. In combination with the user-definable character set, you can produce some surprisingly good color-graphics displays. For example, Apple's well-known "running-horse" demonstration program (shown in photo 6) is produced in color-text mode. The color values shown in table 3, although specified for graphics, can also be used for color text.

With four graphics modes, the Apple III's capabilities are significant-

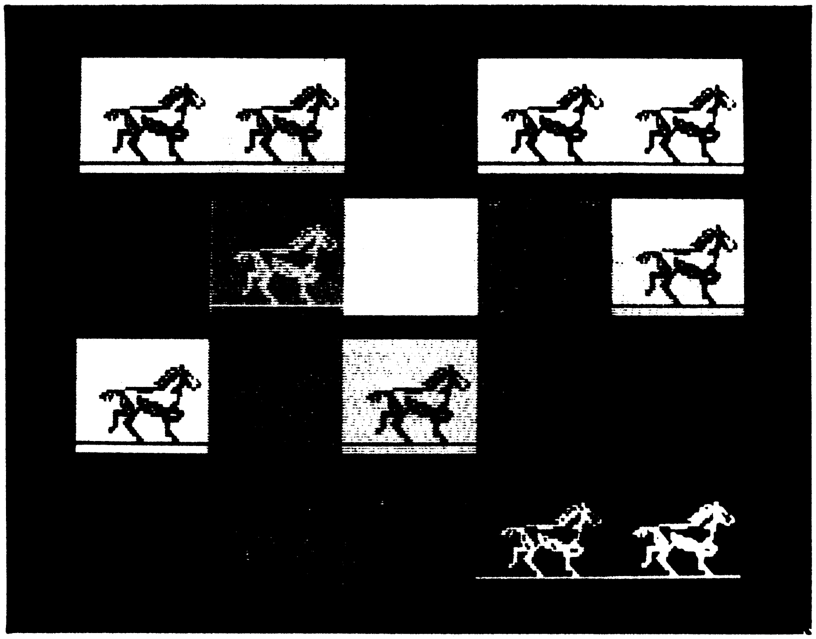


Photo 6: *The well-known "running horse" demonstration. This display was generated using the 24-row by 80-column color-text display mode using the Apple III's programmable character set to produce the special shapes required.*

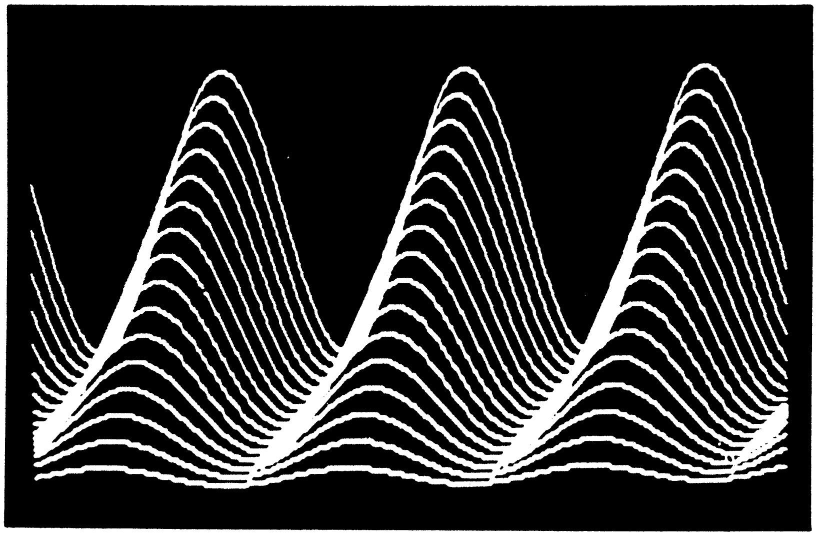


Photo 7: *An example of the 560- by 192-pixel graphics display mode. Although this mode doesn't offer color, it is ideal for displays that require fine detail.*

ly better than those of the Apple II (table 4 shows the available modes). The highest resolution offered is 560 by 192 pixels, black and white. This mode is useful for scientific or technical displays that require maximum resolution, as shown in photo 7. There is also a 280 by 192 black-and-white display mode.

The highest-resolution color display available is 280 by 192 pixels. Using this mode you can display up to 16 colors with some limitations. In each 7-dot-wide section of a given vertical coordinate, only two colors can be displayed. Bits that are turned on will display the specified foreground color, while bits that are turned off display the background color for that section. This is usually noticed only when lines of different colors cross. The limited color mode is useful for many applications where 16 colors are required but where maximum resolution is needed (an example is shown in photo 8).

The most colorful graphics mode is the 140- by 192-pixel 16-color mode. With no limitations on color placement, it is capable of producing very impressive displays (see photo 9). One of the more interesting techniques in this mode mixes various colors of dots to produce a variety of in-between shades of color. Using this technique, it is possible to produce several hundred colors on an Apple III.

Although the resolution is effectively reduced in the shaded areas, this method is typically used for filling in areas of pictures rather than for outlines, which are normally drawn in solid color. A talented artist with a digitizing tablet and the appropriate software can produce results like those shown in photo 10.

Apple SOS

Apple's SOS (Sophisticated Operating System) 1.1 is one of the more powerful operating systems available for an 8-bit microcomputer and offers features usually found only on larger machines. SOS supports multiple nested directories, handles interrupt-driven and DMA I/O, and manages the Apple III memory and hardware environment.

A unique feature of SOS is that there is no user interface. All communications with SOS are handled by the resident language (BASIC or Pascal for now) in a fashion compatible with the language syntax. For example, with Business BASIC you display a disk directory by typing CATALOG (or CAT), but in Pascal you would press F to enter the filter and then press E to get an extended directory. Rumor has it that Apple is working on a separate SOS user-interface package. This would allow access to SOS without requiring that a language be loaded into the system.

All Apple III I/O is handled by SOS through device drivers. Each

Graphics Mode		Graphics	Colors	Memory
Main Screen	Alternate Screen	Resolution	Available	Used
0	4	280h by 192v	black and white	8K
1	5	280h by 192v	16 colors with limitations	16K
2	6	560h by 192v	black and white	16K
3	7	140h by 192v	16 colors, no limitations	16K

Table 4: *The Apple III graphics modes, resolution, available colors, and graphics screen memory requirements. Each main mode allows two separate screen buffers so that one screen may be updated while the other screen is displayed. When the black-and-white gray-scale video output is used, the 16 colors are output as 16 gray levels from black to white.*

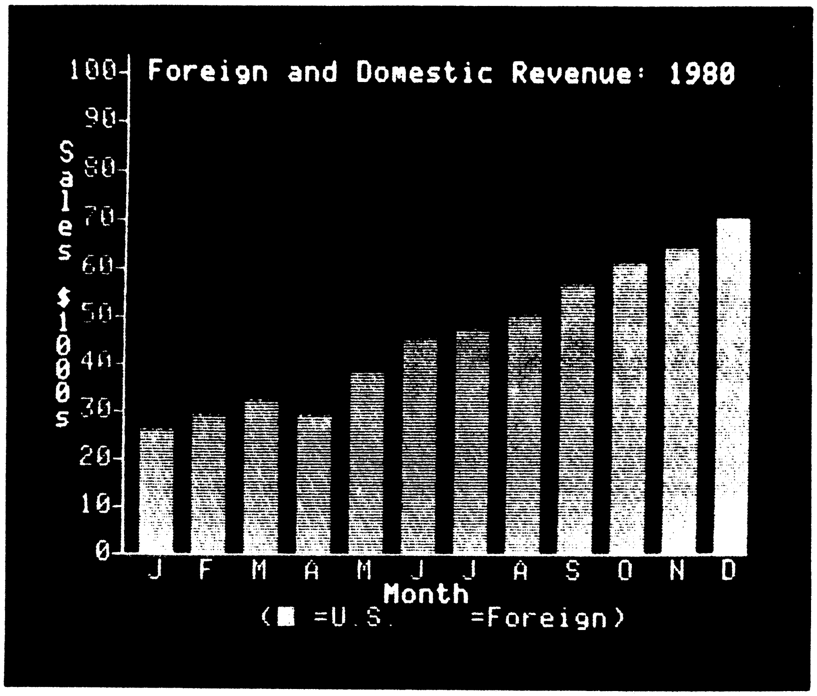


Photo 8: *An example of the Apple III's 280- by 192-pixel limited 16-color mode. While there are some limitations on the combinations of colors that can be displayed next to each other, this mode offers the highest color resolution and is useful in many applications.*

device driver is a group of routines designed to communicate with a particular hardware device and provide a uniform interface to SOS. For example, in a minimal Apple III system, you need the device driver .CONSOLE to handle the keyboard and text display, as well as .FMTD1 to handle the system floppy disk. Some of the other drivers included with the system are .AUDIO, .RS232, .PRINTER, and .GRAFIX. Even though the RS-232C interface and the graphics display hardware are included in the Apple III, they are considered optional I/O devices for programming purposes.

The System Configuration Program (SCP) provides a variety of tools that allow you to modify and reconfigure the system device drivers. Once the device drivers are specified, the SCP can regenerate a version of the system that meets your particular requirements. You can also use the SCP to specify whether a driver will be active or inactive. When the system is booted up, only the active drivers in the SOS.DRIVERS file will be loaded and require memory space.

From the programmer's point of view, device drivers are treated as files and can be used from either BASIC or Pascal. With Business BASIC they may be opened, accessed, and closed like any other file. (You can pass commands and data to an opened driver simply by using the PRINT# statement.) For example, the following Business BASIC lines would list the current program on the Silentype printer if the .SILENTYPE driver were installed:

```
10 OPEN#1, ".SILENTYPE"
20 OUTPUT#1
30 LIST
40 CLOSE#1
```

SOS allows the disk drives to be accessed either by their *device name* (e.g.,.D1) or by the *volume name* of the disk currently in the drive (e.g., MYDISK). Suppose that line 10 from the previous example were changed to read:

```
10 OPEN#1, "MYDISK/LISTFILE"
```

This would cause the program listing to be sent to a file called LISTFILE on a disk called MYDISK.

Unlike most systems which provide a single disk directory, SOS treats a directory like any other file. You can create and maintain directories easily with the same commands (LOCK, UNLOCK, RENAME, DELETE, etc.) that are used to maintain other files. You can assign any type of file to a directory, and any given directory may be a file assigned to another, higher-level directory.

The key to dealing with these nested levels of directories is the SOS *pathname*. Using device and file names separated by slashes, you can tell SOS what path to follow through various levels of directories. For example, the pathname /MYDISK/RECORDS/CHECKS/JAN.81/ would search the system for a disk volume

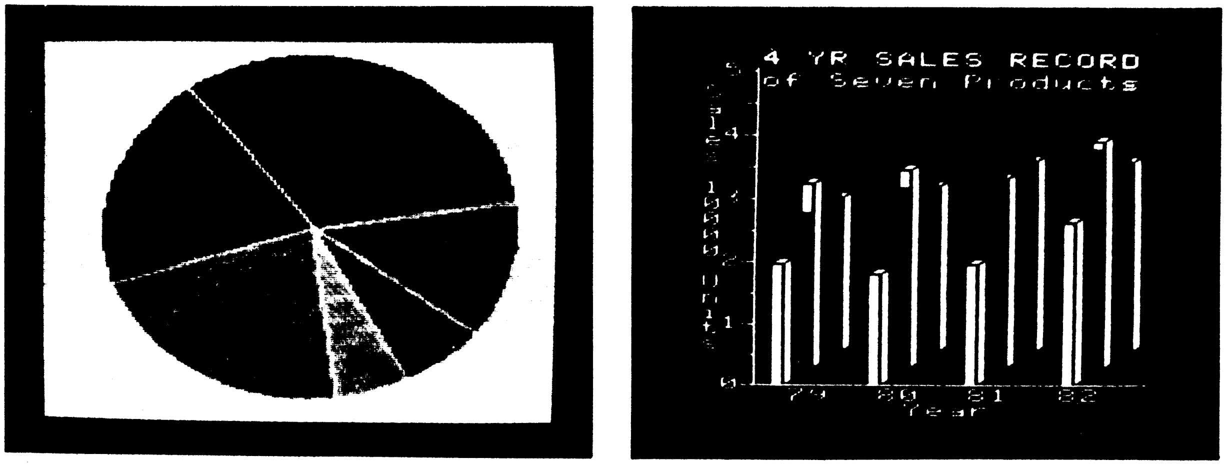


Photo 9: *Two examples of the 140- by 192-pixel full 16-color mode.*

Photo 10: *A talented artist with a digitizing tablet and the appropriate software can produce results like this by using blended colors in the Apple III's 140- by 192-pixel color mode.*

named MYDISK, locate the directory RECORDS (which itself would contain the subdirectory CHECKS), and then locate the file JAN.81. The pathname specifies the sequence of directories to follow when accessing a given file. As a convenience, SOS provides a pathname prefix facility. By using PREFIX$ in the previous example, we could have set the pathname prefix to /MYDISK/RECORDS/ and then simply referred to CHECKS/JAN.81.

File types supported by SOS include DATA, which holds raw binary data; PASTXT (a Pascal text file); PASCODE (a machine language or Pascal program file); BASIC program files; ASCII files of unformatted text; PASDTA (Pascal data files); CAT or directory files; FONT files for the programmable character generator; and FOTO files, which store graphics screen images.

Business BASIC

Although it is fairly conventional, Apple's Business BASIC provides a combination of advanced and unique features that makes it an easier language to use than Applesoft BASIC. With Business BASIC you should be able to write shorter programs with fewer errors. (See tables 5a-5e for a summary of the language.)

Business BASIC supports both TEXT and DATA files. The commands PRINT# and INPUT# are used to access text files while READ# and WRITE# allow you to store or read any type of data in a DATA file. All files may be sequential or random access (with the record size defined when the file is created). You can also use the word CREATE to make new files and directories. Directory entries may be examined by reading sequential text records from a directory file.

The language also provides formatted I/O. To output data to either the screen or a file, you can specify the format with an IMAGE statement or within the PRINT USING statement. The Apple III's output formats are very flexible. Numbers may be printed in fixed-point, floating-point, scientific, or engineering formats. You can also align the right or left edges of the output to a particular column or center the output if you wish.

Four main data types are available in Business BASIC. You can use integers ranging from −32,768 to +32,767, real numbers with 6-digit precision, long-integers with 64-bit binary precision, or strings that can vary from 0 to 255 characters. Arrays without dimensional limits can be created out of all four data types. To convert between the various data types, Business BASIC provides the numeric functions CONV, CONV%, CONV&, and CONV$, all of which will accept arguments of any type and will produce real, integer, long-integer, and string results, respectively.

An interesting feature of Business BASIC is its use of *reserved variables* to access and control certain system functions (see table 5f for a summary). Reserved variable names are used to hold error codes, the file record numbers, or the code for the last key pressed. Others may be used to hold or control the cursor position on the screen, set the listing FOR. . .NEXT loop indent level, control the listing line length, or set the SOS pathname prefix.

One of Business BASIC's most powerful features is its ability to use *invocable modules*. An invocable module is a file of external procedures and functions, written in assembly language or Pascal,* that can act as an extension to the BASIC language once invoked (loaded into the system). The modules provide features that are sometimes necessary but were not built into the Business BASIC language. The modules include VOLUMES.INV, which is used to show which volumes and devices are present in the system; READCRT.INV, which is used to read characters from the video display; DOWNLOAD.INV, which is used to load special text fonts into the Apple III's character generator; and RENUMBER.INV, which provides a variety of functions including program renumber, append, and merge. Another more significant module is BGRAF.INV which provides all the graphics procedures and functions used by Business BASIC.

Once a module has been invoked, the external procedures and functions provided in that file are accessed by using the BASIC commands PERFORM and EXFN. For example, the line

```
PERFORM PENCOLOR(%BLUE)
```

would execute the procedure to set

Command	Description
CATALOG	lists a disk directory
CHAIN	executes a program from disk leaving variables intact
CLEAR	clears program variables
CONT	continues interrupted program
CREATE	creates a new file or directory on disk
DEL	deletes a specified range of BASIC lines
DELETE	deletes a file from disk
HOME	clears current text window and places cursor in upper left-hand corner
INVERSE	sets further text output to inverse video characters
LIST	lists BASIC lines
LOAD	loads a BASIC program
LOCK	protects a file from alterations
NEW	clears a program and variables from memory
NORMAL	sets further text output to noninverse video
NOTRACE	turns off trace option
UNLOCK	removes protection from a disk file
RENAME	changes name of file on disk
RUN	loads and runs programs from disk or runs current program
SAVE	saves current program on disk
TEXT	sets screen to text mode with full-screen window
TRACE	turns on trace option

Table 5a: *A summary of Business BASIC commands.*

Statement	Description
CLOSE	closes all open files
CLOSE#	closes a particular file
DATA	standard DATA statements
DEFD FN	user-defined function
DIM	dimensions arrays
END	ends program
FOR. . .NEXT	standard FOR loop
GET	reads a single character from the keyboard or an EXEC text file
GOSUB	executes a subroutine
GOTO	continues execution at a specified line
IF. . .GOTO. . .ELSE	modified IF statement
IF. . .THEN. . .ELSE	standard IF statement
IMAGE	defines a PRINT USING format
INPUT	reads data from the keyboard
INPUT#	reads text from a disk file or other open device
INVOKE	loads an external file module of assembly-language routines
ON EOF#	sets up end-of-file error trap
OFF EOF#	turns off end-of-file error trap
ON ERR	sets up general error trapping
OFF ERR	turns off general error trapping
ON KBD	sets up keyboard interrupt handling
OFF KBD	turns off keyboard interrupt handling
ON GOSUB	standard computed GOSUB statement
ON GOTO	standard computed GOTO statement
OPEN# . . . AS	opens a file as INPUT, OUTPUT, or EXTENSION
OUTPUT#	sends subsequent output to file
PERFORM	executes a previously invoked routine
POP	removes one level of subroutine nesting
PRINT	prints to current output device or file
PRINT USING	prints using a given format

Table 5b: *A summary of Business BASIC statements.*

Table 5b continued on page 114

*Invocable modules must be written in 6502 assembly language, not Pascal.

Table 5b continued:

Statement	Description
PRINT#	prints to a particular output device or file
PRINT# USING	prints to a particular file or device using a given format
READ	reads information from DATA statements
READ#	reads information from a data file
REM	standard remark statement
RESTORE	resets read pointer to start of DATA list
RESUME	returns from on ON ERR statement
RETURN	returns from a subroutine, ON KBD or ON EOF routine
SCALE	adjusts PRINT USING decimal-point position
SPC	used in PRINT statements to output numbers of blanks
STOP	stops program execution
SWAP	swaps the values of two given variables
TAB	used in PRINT statements to position the cursor to a particular column
WINDOW	sets the text/scroll window size and position
WRITE#	writes information to a data file

Function	Description
ABS	absolute value
ASC	converts ASCII character to its numeric value
ATAN	arc tangent
BUTTON	paddle-button state
CHR$	converts number to equivalent ASCII character
CONV	evaluates expression—returns real number value
CONV$	evaluates expression—returns string value
CONV&	evaluates expression—returns long-integer value
CONV%	evaluates expression—returns integer value
COS	cosine
EXFN	executes an invoked external function that returns a real number value
EXFN%	executes an invoked external function that returns an integer value
EXP	exponential, base *e*
HEX$	returns a string that represents the hexadecimal value of the expression
INSTR	searches a string for a substring and returns location of occurrence
INT	largest integer less than or equal to argument
LEFT$	takes substring starting with first character
LEN	length of a string
LOG	natural logarithm
MID$	extracts a substring from a given string
PDL	returns a game-paddle position
REC	returns current file record number
RIGHT$	takes substring ending with last character
RND	random number
SGN	sign of argument
SIN	sine
SQR	square root
STR$	converts a number to a string
SUB$	inserts a substring into a given string
TAN	tangent
TEN	converts last four characters of a string from a hexadecimal text image to a decimal value
TYP	returns the data type of a file record
VAL	converts a string to a numeric value

Table 5c: *A summary of Business BASIC functions.*

the graphic drawing color to blue, provided that the variable BLUE has previously been defined properly.

While external procedures may be passed only integer values, external functions can return either integer or floating-point numbers. The reserved word EXFN% is used to call functions that return integers and EXFN accesses functions that return real values.

BASIC Graphics

Although you could use graphics from BASIC by simply opening the .GRAFIX driver and sending characters directly to it, the BGRAF.INV module provides a much cleaner and more powerful interface. It essentially adds a number of graphics commands to the Business BASIC language. (A similar library unit is included with Apple III Pascal.) The .GRAFIX driver must still be present and opened because you need a controller for the graphics hardware, but all graphics operations are performed by the external procedures and functions provided by BGRAF. The following two lines provide all the setup required:

```
100 OPEN#1, ".GRAFIX"
110 INVOKE "BGRAF.INV"
```

BGRAF provides all of the standard graphics operations. You can set PENCOLOR and the background FILLCOLOR, plot dots at absolute or relative positions with DOTAT and DOTREL, draw lines to absolute or relative points with LINETO and LINEREL, and position the graphics cursor with MOVETO and MOVEREL. BGRAF supports a graphics VIEWPORT that allows you to limit graphics drawing to a particular area of the display screen.

Text may be displayed with graphics by simply sending it to the opened .GRAFIX driver with a PRINT# statement. NEWFONT lets you redefine the graphics text font by specifying character form, height, and width. The SYSFONT command switches you back to the current text-mode display font.

Predefined images stored in integer arrays may be displayed with DRAWIMAGE. A given array may hold a

Operators	Type
+ − * / DIV MOD	arithmetic
AND OR = < > >< <> <= =< >= =>	logical
NOT	unary logical
+	string concatenation

Table 5d: *A summary of Business BASIC data operators. DIV and MOD apply only to the long-integer data type.*

Data Type	Type Name	Range
16-bit integer	integer	− 32768 to 32767
64-bit integer	long-integer	± 9223372036854775807 ($\pm 2^{63} - 1$)
32-bit floating point	real	$\pm 10^{38}$ with 6 digit precision
character strings	string	0 − 255 characters
arrays	(all types)	no dimensional limits

Table 5e: *A summary of Business BASIC data types and ranges.*

Variable	Description
EOF	holds reference number of file causing an EOF error
ERR	holds error type code of most recent error
FRE	holds amount of remaining bytes of memory available
HPOS	holds/controls cursor horizontal position
INDENT	holds/controls number of spaces to indent FOR . . . NEXT loops in listings
KBD	holds the ASCII value of the last key pressed
OUTREC	holds/controls the maximum line length output by the LIST command
PREFIX$	holds/sets current SOS pathname prefix
VPOS	holds/controls current cursor vertical position

Table 5f: *A summary of Business BASIC reserved system variables.*

Procedure	Description
DOTAT	plots a single dot at a given position
DOTREL	plots a dot relative to current position
DRAWIMAGE	draws a rectangular bit-map image at current position
FILLCOLOR	sets background color
FILLPORT	fills current VIEWPORT with FILLCOLOR
GLOAD	loads and displays a FOTO file from disk
GRAFIXMODE	specifies graphics mode and buffer choice
GRAFIXON	switches display to current graphics mode and buffer
GSAVE	saves current graphics display as a FOTO file on disk
INITGRAFIX	sets full-screen VIEWPORT, places cursor at upper left-hand corner and sets normal color and transfer tables
LINEREL	draws a line relative to current position
LINETO	draws a line from current to an absolute position
MOVEREL	positions cursor relative to current position
MOVETO	positions cursor at an absolute position
NEWFONT	used to specify a new graphics character font
PENCOLOR	sets current PLOT and DRAW color
RELEASE	frees highest graphics buffer memory
SETCTAB	sets a color-table entry
SYSFONT	causes normal system character set to be used as graphics character font
VIEWPORT	defines graphics-drawing window size and position
XFEROPTION	defines the logical operation that places dots on the screen
XLOC	returns graphics-cursor *x* position
XYCOLOR	returns color of dot on screen at current position
YLOC	returns graphics-cursor *y* position

Table 5g: *A summary of Business BASIC graphics procedures.*

number of images that can be selected with the DRAWIMAGE arguments.

One of the most interesting features of BGRAF is its control of color. By using two controllable processes—the color table and the transfer option—you can modify the effects of plotting and filling operations.

With 256 entries, the color table specifies which color results from plotting a dot of a given "source color" on top of a dot of a given "screen color." The color table is initialized to simply display the source color regardless of the existing color of the specified dot position. However, by altering the mapping conditions in the color table you can establish a color precedence. This precedence allows lines to appear to pass under or over existing images, or it can produce a number of other interesting effects.

To alter a color-table entry, you use the enternal function SETCTAB. The form of the statement would be:

```
SETCTAB (%SOURCECOLOR,
   %SCREENCOLOR,
   %RESULTCOLOR)
```

The following example would alter the color table so that when an orange dot was printed onto a blue background, the result would be green:

```
SETCTAB (%9, %6, %12)
```

Table 3 shows a summary of the graphics colors and their color values.

The black-and-white equivalent of the color table is the *transfer option*, which describes the logical operation used to place dots on the screen. De-.

pending upon the option specified, a dot (or its inverse) may replace existing data, overlay it, invert it, or erase it with new data. The XFEROPTION procedure and an argument specify the transfer mode. The transfer option may also be used with color data, but predicting the results is difficult.

Although circle drawing and turtle graphics are not supported, BGRAF is still a very nice package of routines that should allow you to produce a wide variety of color graphics. (See table 5g for a summary.)

Business BASIC Performance

Although Business BASIC is much more powerful than the Apple II's Applesoft BASIC, it is not much faster. Tests with the series of sixteen benchmark programs shown in listing 1 indicated that while Business BASIC is faster than Applesoft in some areas, it is slower in others. The net result should be a slight to medium speed improvement, depending upon the program being run.

The best test in the series was probably the Sieve of Eratosthenes prime-number program used by Jim Gilbreath (see "A High-Level Language Benchmark," September 1981 BYTE, page 180). Although this program is more representative of average program execution than any of the other benchmarks, it uses only addition and subtraction and does not have a wide variety of BASIC statements. In this test, the Apple III proved to be slightly faster than the Apple II but slower than the IBM Personal Computer or the 4-MHz Z80.

The execution speed advantage of the 6502B is largely cancelled out by the complexity of Business BASIC.

From the results of this limited set of benchmarks, it seems that the execution speed advantage of the Apple III's 6502B is largely cancelled out by the increased complexity of Business BASIC. However, I suspect that in larger programs Business BASIC will turn out to be a good deal faster than Applesoft. The combination of its powerful built-in features and invocable modules will eliminate the code required in Applesoft to accomplish the same functions. Also, if the benchmark programs had included the appropriate code to turn off the video screen during time-critical calculations, an additional 30 percent speed increase could have been gained by allowing the 6502B to run at 2 MHz. This would have placed the Apple III ahead of the IBM and Z80 computers in many tests.

Although benchmarks always have some validity, they may or may not be significant in a given application. It is best to approach the results with caution—the programmer frequently

Listing 1: *Execution benchmark programs. See table 6 for a summary of their results.*

Listing 1a: *tests a null loop.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
320     NEXT I
```

Listing 1b: *tests REM execution time.*

```
100   FOR I=1 TO 5000
120     REM
140     REM
160     REM
180     REM
200     REM
210     REM
240     REM
260     REM
280     REM
300     REM
320     NEXT I
```

Listing 1c: *tests the IF. . .THEN statement.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
120     IF A<B THEN 320
320     NEXT I
```

Listing 1d: *tests addition.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
120     C=A+B
320     NEXT I
```

Listing 1e: *tests multiplication.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
120     C=A*B
320     NEXT I
```

Listing 1f: *tests division.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
120     C=A/B
320     NEXT I
```

Listing 1g: *tests exponentiation.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
120     C=A^B
320     NEXT I
```

Listing 1 continued on page 124

Listing 1 continued:

Listing 1h: *tests transcendental functions.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
120     C=SIN(A)
320     NEXT I
```

Listing 1i: *tests the LOG function.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
120     C=LOG(B)
320     NEXT I
```

Listing 1j: *tests the ON. . .GOTO statement.*

```
80   M=2
100   FOR I=1 TO 5000
120     ON M GOTO 80,320,100
320     NEXT I
```

Listing 1k: *tests the GOSUB/RETURN statement.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
120     GOSUB 1000
320     NEXT I
1000   RETURN
```

Listing 1l: *tests the INT (integer) function.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
120     C=INT(A)
320     NEXT I
```

Listing 1m: *tests the MID$ function.*

```
80   A$="abcdefghijklm"
100   FOR I=1 TO 5000
120     B$=MID$(A$,6,6)
320     NEXT I
410   PRINT""
420   END
```

Listing 1n: *tests random number speed.*

```
60   A=2.71828
80   B=3.14159
100   FOR I=1 TO 5000
120     C=RND(1)
320     NEXT I
```

Listing 1o: *tests the CHR$ function.*

```
80   A$="abcdefghijklm"
100   FOR I=1 TO 5000
120     C$=CHR$(50)
320     NEXT I
```

Listing 1 continued on page 126

Listing 1 continued:

Listing 1p: *Jim Gilbreath's Sieve of Eratosthenes prime-number program.*

```
1    SIZE=7000
2    DIM FLAGS(7001)
3    PRINT"only 1 iteration"
5    COUNT=0
6    FOR I=1 TO SIZE
7      FLAGS(I)=1
8      NEXT I
9    FOR I=0 TO SIZE
10       IF FLAGS(I)=0 THEN 18
11      PRIME=I+I+3
12      K=I+PRIME
13       IF K>SIZE THEN 17
14      FLAGS(K)=0
15      K=K+PRIME
16      GOTO 13
17      COUNT=COUNT+1
18      NEXT I
19    PRINT COUNT,"primes";""
```

Listing 2: *Disk-access benchmark programs. Listings 2a and 2b are write and read tests for the Apple III. Similar programs were used for the Apple II and the IBM Personal Computer.*

(2a)

```
40    A$="12345678123456781234567812345678"
60    B$=A$+A$+A$+A$
80    NR=500
100    OPEN#1,"TEST"
140    FOR I=1 TO NR
160       INPUT#1;B$
200      NEXT I
220    CLOSE#1
240    PRINT"DONE"
```

(2b)

```
40    A$="12345678123456781234567812345678"
60    B$=A$+A$+A$+A$
80    NR=500
100    OPEN#1,"TEST"
140    FOR I=1 TO NR
160       PRINT#1;B$
200      NEXT I
220    CLOSE#1
240    PRINT"DONE"
```

Listing #	Benchmark	Apple III Business BASIC	Apple II Applesoft BASIC	IBM Advanced BASIC	4-MHz Z80 MBASIC 4.51
1a	empty loop	8.9	6.7	6.43	5.81
1b	10 REMs	19.2	19.5	21.0	15.8
1c	IF. . .THEN	22.9	19.8	17.6	14.9
1d	addition	19.5	17.5	18.2	16.3
1e	multiplication	25.0	27.3	19.6	19.9
1f	division	27.6	28.8	23.8	24.9
1g	exponentiation	184.5	249.1	84.8	121.1
1h	sine(x)	98.0	193.1	73.9	63.1
1i	log(x)	87.1	113.6	49.4	55.4
1j	ON. . .GOTO	18.6	17.5	17.3	12.9
1k	GOSUB/RETURN	16.4	13.6	12.4	9.4
1l	INT(x)	20.0	19.3	18.1	15.5
1m	MID$	37.3	32.5	23.0	18.6
1n	RND(x)	90.5	33.1	18.4	19.7
1o	CHR$	26.8	23.5	16.2	13.4
1p	prime numbers	222.4	224.4	190.0	151.0

Table 6: *Table of execution times (in seconds) for a series of benchmark tests run on Apple III Business BASIC, Apple II Applesoft BASIC, IBM Personal Computer Advanced BASIC, and a 4-MHz Z80 computer running Microsoft's MBASIC 4.51. The results shown may or may not be indicative of performance in a particular application; they should be interpreted with caution. The results for the IBM Personal Computer and the Z80 microcomputer were taken from Gregg Williams' "A Closer Look at the IBM Personal Computer" (January 1982 BYTE, page 54). See listing 1 for the benchmark programs used.*

makes more difference than the machine. (The benchmark results are summarized in table 6.)

Apple II Emulation

The Apple III's ability to emulate an Apple II is an extremely useful feature that allows access to the tremendous volume of Apple II software. Virtually all Apple II DOS 3.3 programs in either Applesoft or Integer BASIC can be run on the Apple III without change—the few exceptions are those programs that require a RAM card or language system to operate. Also, some of the Apple II arcade games use their own routines to read the game paddles rather than calling the routines in the Apple II's monitor ROM. These programs will run but will not operate correctly.

To use the Apple II emulation mode, you must boot a special emulation disk and select either Applesoft or Integer BASIC as the available language. Since the Language Card is not emulated, only one language at a time can be resident. The Apple III serial port can be configured to emulate either an Apple II serial card or a communications card. The data rates and carriage-return handling can also be specified. Once the emulation parameters are specified or the defaults accepted, you can boot a normal Apple II DOS 3.3 disk and start running.

The emulation mode has a few minor weak points. If you have an Apple III Silentype printer, it will not be accessible in emulation mode unless you install an Apple II Silen-

At a Glance

Name
Profile Winchester-technology disk drive

Manufacturer
Apple Computer Inc.
20525 Mariani Ave.
Cupertino, CA 95014
(408) 996-1010

Price
$3499

Storage Capacity
5 megabytes (equivalent to about 35 normal Apple 5¼-inch floppy disks)

Size
Height 4.39 inches (11.5 cm), width 17.28 inches (43.89 cm), Depth 8.81 inches (22.38 cm)

Weight
11 pounds (5 kg)

Power Required
110 volts AC, (U.S.), 35 watts

Hardware Required
Apple III computer

Software Required
Apple SOS 1.1

Organization
Four data surfaces, 153 tracks per surface, 16 sectors per track, 512 bytes per sector, 2448 sectors per surface, 9792 sectors per drive

Specifications
Data transfer rate: 5 megabits per second; average seek time: 95 milliseconds; rotational speed: 3600 revolutions per minute; ready to operate: 60 seconds

Interface
Interface card occupies one Apple III expansion slot; one drive per interface card, up to four drives per system

Special features
Power-up self-test and disk scan; automatic bad-sector relocation; error checking and limited error correction

type interface card, which may violate FCC radio-frequency radiation limits. Nor can you access the Profile hard-disk drive—Apple II and Apple III files won't mix on the same disk. Also, the RGB (red-green-blue) video outputs will not provide color signals while emulating Apple II graphics, but the composite video outputs will work normally.

The Profile

The Profile hard-disk drive is the newest component of the Apple III family and a worthy occupant of an expansion slot. With a 5-megabyte capacity, integral Z8-based controller, and built-in power supply, the Profile is a self-contained intelligent subsystem with its own self-test, error checking, and bad-sector relocation facilities.

When powered up, the Profile's controller waits for the disk to come up to speed and does a data integrity check by stepping from track to track to verify that all disk sectors read correctly. If a bad sector is found, either during this process or during normal activity, the Profile attempts to correct the data errors and then relocates as much data as possible to an alternate good sector.

The key component in the Profile is the ST-506, a 5¼-inch hard-disk drive manufactured by Seagate Technology Inc. The ST-506 uses the sealed disk environment and low-altitude (10-microinch) flying heads that characterize all Winchester-technology disk drives (see photo 11). Because a number of vendors produce drives that are plug-compatible with the ST-506, Apple should have no trouble producing Profiles even if Seagate's supplies get short.

During operation the disk drive is relatively quiet, emitting a soft tone as it steps from track to track. Between accesses you can hear the main drive motor, but the sound should not be obtrusive or even audible in most office environments.

The Profile is styled to match the rest of the Apple III system and may be positioned on top of or adjacent to the computer.

I found the Profile a pleasure to use. Its capacity is equivalent to that of about 35 normal Apple floppy disks, and its data throughput is about 10 times faster. Viewing its capacity in other terms, the Profile can hold over 1200 pages of typed text or more than 300 high-resolution graphics pictures occupying 16K bytes apiece.

The Profile's performance is excellent. In the disk-access benchmark programs shown in listing 2, the Profile effectively tripled the program speed when compared to an Apple or IBM floppy disk. Considering that a significant proportion of the program execution time is used to execute the BASIC program statements, the actual increase in disk-access speed would seem to be even higher. (The results of the disk-access benchmarks are summarized in table 7.)

The weakest point of the Profile and other similar products is data backup. If a hard disk fails, you can lose a great deal of important data. The only solution is to periodically back up the most critical files onto floppy disks or onto a second Profile hard-disk unit. (Apple Computer will happily allow you to connect up to four Profiles to your Apple III, at a total cost of $13,996 in addition to the cost of the Apple III.) However, chances are very slim that the entire Profile would be wiped out if a critical component failed. After repair, it should be possible to recover virtually all the original data in most cases.

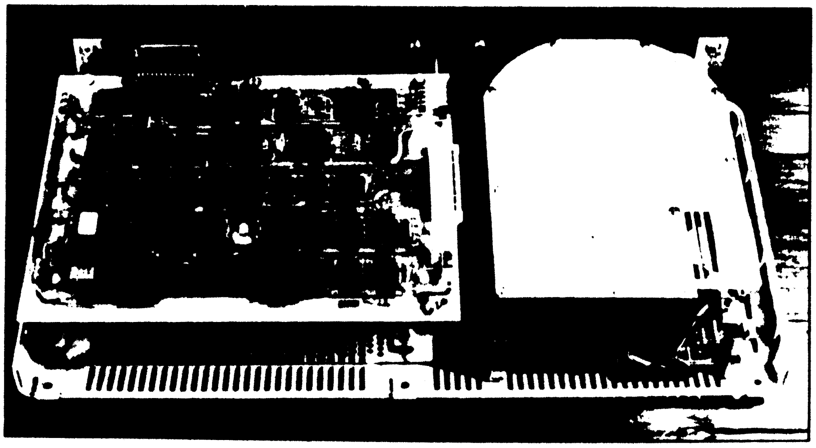


Photo 11: *The Profile with its top cover removed. The intelligent controller is shown on the left with the switching power supply beneath it. The HDA (hard-disk assembly) with its sealed internal environment is mounted on the right.*

	Apple III Profile	Apple III Floppy Disk	Apple II Floppy Disk	IBM Floppy Disk
Write	13.2	37.3	234	32
Read	10.2	33.2	273	22.9

Table 7: *A summary of disk-access-time benchmarks comparing the performance of the Apple III Profile hard-disk drive and the Apple III, the Apple II, and the IBM Personal Computer floppy-disk drives. The table shows the times (in seconds) taken to read and write 500 disk records.*

At $3499, the Profile isn't inexpensive—none of the available hard-disk subsystems are—but it provides a truly significant extension to the capabilities of the Apple III system.

Documentation

Apple Computer's documentation has always been excellent, and the manuals provided with the Apple III are no exception. All the manuals are in the familiar 6- by 8½-inch (12.8- by 21.6-cm) format, and a new flap has been added to the back cover so that the manual title is visible while the book is on the shelf. The manuals are all clearly written with numerous charts, tables, and screen photos to illustrate points described in the text.

With a Business BASIC system, you receive four manuals: the *Owner's Guide* and *Standard Device Drivers* deal with Apple III features and SOS, while volumes one and two of *Apple Business BASIC* provide a comprehensive description of the language.

The *Owner's Guide* explains how to set up the Apple III system and describes various aspects of SOS and the Apple III hardware. There are sections about system installation and start-up, the operating system, the System Configuration Program, and the machine itself. Appendixes explain error messages, describe proper disk care and handling, give I/O port specifications, and tell you how to use the Apple II Emulator. The information is presented in a clear, easy-to-read style and should be sufficient to get any novice started.

Standard Device Drivers provides complete specifications and descriptions of the operation of all of the standard I/O device-driver routines. After a short section that explains what device drivers are, the manual describes the System Configuration Program. Separate sections describe each individual driver in detail. The appendixes contain quick references for all the drivers, an explanation of the system error messages, and a description of the console data formats.

With a BASIC system, you'll get *Business BASIC* volumes 1 and 2. Although the manuals were not designed to teach BASIC, the 335 pages contain all the information required to learn Apple's version of that language. Volume 1 is primarily a tutorial section; it gives clear explanations of all of the BASIC statements and provides numerous examples. After a short introduction and a description of the BASIC editor, different sections describe BASIC I/O, control of program execution, and file I/O. The manual also explains invocable modules and shows you how to use external procedures and functions.

Business BASIC volume 2 is primarily a quick reference quide that will be of most use to people who have some familiarity with the Business BASIC language. Within the BASIC reference section, each language statement and function is described and shown in an example along with descriptions of any error messages that might be produced when it is used. Separate appendixes describe error messages and their causes, explain variable memory usage, tell how to program for maximum speed, and give syntactic definitions of the Business BASIC language. The Graphics invocable module (BGRAF.INV) is described in a 57-page section that gives detailed examples of plotting and drawing, saving pictures on disk, creating graphics text fonts, and setting up your own color and transfer tables.

If you purchase Apple III Pascal, you'll get an additional four manuals that describe the Pascal system, utility programs, and the Pascal language. One distinct benefit of Apple III Pascal is that the description of the Pascal assembler provides details about the 6502 enhanced features that are not found in any of the other manuals. Unfortunately, even though the BASIC invocable modules are written in Pascal, the manuals do not tell you how to write them. This may not be important to small-business users; nevertheless, the information should be available.

Summary

It is impossible to do the Apple III justice in one article. The machine is very flexible and has a mix of features and capabilities that are unmatched in any of its competitors. Some points, however, deserve special mention.

First, SOS is a unique and powerful operating system; it provides a variety of features that, as far as I know, are not available on any other 8-bit machine.

Business BASIC is also very powerful and includes options not found in most versions of the language. The use of invocable modules allows the user to maximize available memory space by adding only the capabilities needed. Its I/O-formatting and file-handling capabilities are extremely versatile and, for most business data-handling applications, will allow programs to be shorter and easier to debug.

As for hardware, although some people might argue that Apple should have chosen a more advanced microprocessor than the 6502B for the Apple III, I think the company made the right choice. Without the 6502B it would have been difficult, if not impossible, to transfer files and programs from the Apple II to the III, and Apple II emulation would not have been possible. Admittedly, it was a conservative choice—more powerful processors are available—but actual processor performance is much less important than software availability. Apple's choice clearly maximizes the usability of the system.

The Profile hard-disk drive is a significant enhancement to the Apple III. Its speed and high capacity will eliminate 99 percent of the disk swapping required when using only floppy disks, and the SOS nested directory structures will keep it well organized.

Finally, one of the strongest points in favor of the Apple III is Apple Computer Inc. When early Apple III users had problems with the first machines, Apple simply replaced the entire computer immediately—as many as two or three times in some cases. This unqualified backing of its products shows a commitment to customer satisfaction unequaled in the industry. ■