Apple *III*

SOS

Reference Manual, Volume 1

Contents

Volume 1: How SOS Works

2 Programs and Memory 9

Volume 2: The SOS Calls

Figures and Tables

Volume 1: How SOS Works

5 File Organization on Block Devices 75

6 Events and Resources 103

7 Interpreters and Modules 117

8 Making SOS Calls 147

Volume 2: The SOS Calls

Preface ix

10 Device Calls and Errors 57

E Data Formats of Assembly-Language Code Files 131

Preface

For your convenience and ease of reference, this manual is divided into two volumes. Volume 1: How SOS Works describes the operating system of the Apple III. Volume 2: The SOS Calls defines the individual SOS calls. Notice that the sequence of chapter numbers in Volume 1 continues unchanged into Volume 2.

Scope of this Manual

This manual describes SOS (pronounced "sauce"), the Sophisticated Operating System of the Apple III. With the information in this manual you'll be able to write assembly-language programs that use the full power of the Apple III.

However, this manual is not a course in assembly-language programming. It assumes that you can program in assembly language and know the architecture of the 65Ø2 microprocessor upon which the Apple III is based; it will explain how the architecture of the Apple III processor goes beyond that of the standard 65Ø2. If you need more information on 65Ø2 assembly-language programming, refer to one of the books listed in the bibliography of this manual.

The companion volume to this manual, the *Apple III SOS Device Driver Writer's Guide*, contains the information you may need about the interface hardware of the Apple III, and tells how to create device drivers to use that hardware. If you wish to create custom interface software or hardware for the Apple III, read the present manual before turning to the *Apple III SOS Device Driver Writer's Guide.*

Using this Manual

Before you begin with this manual, you should prepare yourself by reading the following:

- *the Apple III Owner's Guide* introduces you to some of the fundamental features of the Apple III—features that you will be exploring more deeply in this manual;
- *the Apple III Standard Device Drivers Manual* describes the workings of the Apple III's video screen, keyboard, graphics, and communications interfaces;
- *the Apple III Pascal Program Preparation Tools* manual explains the use of the Apple III Pascal Assembler, which is the only assembler that works with SOS.

You should also finish reading this preface, to learn about the notation and examples used in this manual.

About the Examples

Included in this manual are many sample programs and code fragments. These are intended as demonstrations *only*. In order to illustrate their concepts as well as possible, they are written to be clear and concise, without necessarily being efficient or comprehensive.

Notation and Symbols

Some special symbols and numeric notations are used throughout this manual.

Numeric Notation

We assume that you are familiar with the hexadecimal (hex) numbering system. All hexadecimal numbers in the text and tables of this manual are preceded by a dollar sign ($). Any number in the text, a table, or illustration that is not preceded by a dollar sign is a decimal number.

Program listings from the Apple III Pascal Assembler, however, do not prefix hex numbers with dollar signs. In such listings, you can distinguish decimal numbers from hex by the fact that decimal numbers end with a decimal point (.). You can distinguish hex numbers from labels by the fact that hex numbers always begin with a digit from Ø to 9, and labels always begin with a letter.

Type	Notation in Text	Notation in Listings
Decimal	255	255.
Hexadecimal	$3A5	3A5
Hexadecimal	$BAD1	ØBAD1
Label	BAD1	BAD1

Table 0-1. Numeric Notation

Additional notations are introduced in Chapter 1.

Special Symbols

Four special symbols are used in this manual to emphasize information about helpful or unusual features of the system.

This symbol precedes a paragraph that contains especially useful information.

Watch out! This symbol precedes a paragraph that warns you to be careful.

Stop! This symbol precedes a paragraph warning you that you are about to destroy data or harm hardware.

This symbol precedes a paragraph that is specific to versions 1.1, 1.2, and 1.3 of SOS. Note especially that, although the symbol indicates version 1.2, it is also applicable to versions 1.1 and 1.3.

1

The Abstract Machine

1.1 About Operating Systems

An *operating system* is the traffic controller of a computer system. A well-designed operating system increases the power and usefulness of a computer in three important ways. First, an operating system establishes an *abstract machine* that is defined by its concepts and models, rather than by the physical attributes of particular hardware. Second, it acts as a *resource manager*, to ease the programming task. Finally, it provides a *common foundation for software*.

If you are an experienced programmer of small computers, such as the Apple II, but you have never written large programs for a machine with an operating system, you should pay particular attention to this section.

1.1.1 An Abstract Machine

The low-level programming language of a computer is determined not only by its central processor, but by its operating system as well. The operating system is thus an essential part of the programming environment: knowing how it works lets you write programs that use the full power of the machine.

Most importantly, the combination of hardware and operating system software creates an abstract machine that is neither the hardware nor the operating system, but a synthesis of both. This is the machine you program.

The major advantage of the abstract-machine concept is that a program written for the abstract machine is not bound by the current configuration of the hardware. The operating system can compensate for expansions, enhancements, or changes in hardware, making these changes invisible to the programs. Thus programs properly written for an abstract machine need not be modified to respond to changes or improvements in the hardware.

1.1.2 A Resource Manager

An operating system also controls the flow of information into, out of, and within the computer. It provides standard ways to store and retrieve