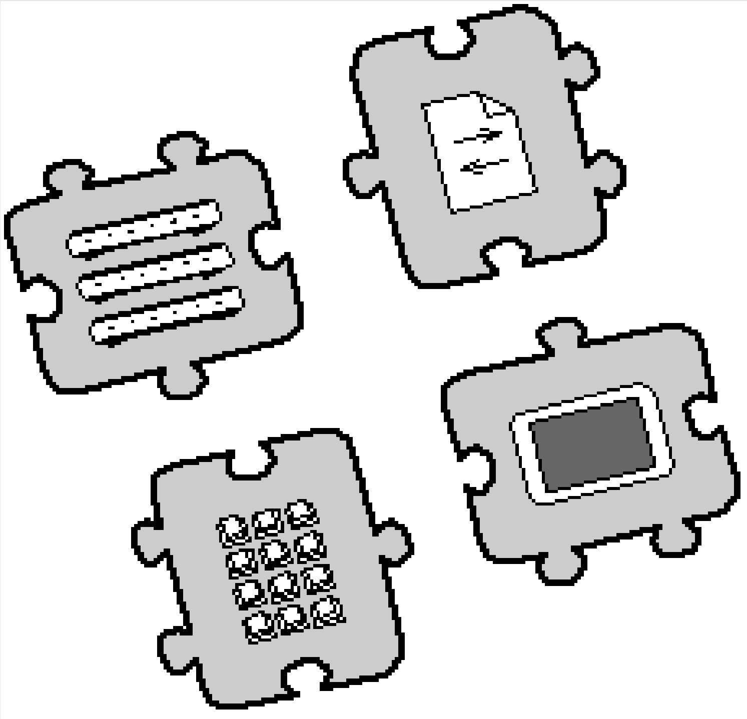
ModemWorks

Communications Toolbox



Morgan Davis Group

Copyright

http://www.morgandavis.net

Fourth printing — May 1992 — U.S.A.

Printed on recycled paper.

Contents

Chapter One: Getting Started

Chapter Two: Using ModemWorks

Chapter Three: ModemWorks Modules

Chapter Four: ModemWorks Commands

Introduction

What is ModemWorks?

The only *communications toolbox* for Applesoft, ModemWorks lets you develop high performance data communications software in BASIC. It provides basic communication services, such as transparent input/output, flow control, inactivity timing, cancel key and carrier loss management. It also includes extended services, like full screen terminal emulation and file transfer protocols. Although ModemWorks comes with everything needed to create communications programs on your computer, its modular design offers “plug and play” expandability, allowing developers to add additional features. Since ModemWorks runs under the Object Module Manager, it is compatible with a myriad of powerful tools and utilities available for Applesoft.

Throughout this manual the following symbols are used to denote keys on your keyboard:

◁reset	Reset
option	Option or solid-apple
command	Command or open-apple
control	Control
esc	Escape
return	Return
delete	Delete
↑	Up arrow or control-K
↓	Down arrow or control-J
←	Left arrow or control-H
→	Right arrow or control-U
tab	Tab or control-I
shift	Shift

Hyphenated key references, such as command-esc, tell you to press and hold the first key while typing the second.

Credits

ModemWorks, spanning nearly a decade of development and refinement, would not be in your hands now if it were not for key support I have received. In particular, I extend my thanks to Bill Blue and Joe Holt, both who had an early influence on my work.

Special thanks to Andy Nicholas (famous for ShrinkIt and the Apple IIGS Finder 6.0) for contributing ZMODEM modules. Andy spent a lot of free time on these between software engineering deadlines at Apple Computer.

The ModemWorks beta testing batallion deserves mention (in no particular order): Jon Thomason, Daniel Davidson, Don Elton, Mike Ungerman, Jeff Jungblut, Dean Fick, Shawn Goodin, Gary Snow, and Larry Colton. *For all you do, disk bugs for you!*

Great appreciation is reserved for Tim Swihart (Apple Computer). Tim goes out of his way to support Apple II developers, and has been especially valuable to us.

My perpetual gratitude goes to the many supportive customers and friends who have made it possible for me to create Apple II software. And, as always, I am grateful for the love and patience of family as I go off to hack everyday.

To you all, I express my most sincere thanks.

—Morgan Davis

Getting Started

This chapter shows you how to set up your computer to begin using and exploring ModemWorks. You'll learn about cables, modem settings, and everything needed to allow ModemWorks to run on your system.

What You Should Know

Since ModemWorks is a BASIC programmer's utility, a working knowledge of Applesoft and ProDOS BASIC (BASIC.System) is essential.

ModemWorks relies on the Object Module Manager (OMM) to efficiently manage its modular components such as console, serial, and modem drivers. For more details on using the OMM with your ModemWorks programs, see the OMM manual.

In order for ModemWorks to work correctly with your computer, you may have to make adjustments to your hardware. You should be comfortable working with your computer's peripherals, installing interface cards, setting DIP switches, and if you have an Apple IIGS, making adjustments in the Control Panel.

Hardware Supported

ModemWorks runs under ProDOS-8 on the Apple IIe and Apple IIGS, and is compatible with:

◊ Apple IIGS modem and printer ports
◊ Super Serial Card (and clones)
◊ Internal Hayes-compatible modem cards
◊ External modems
◊ Serial and Parallel printer interfaces
◊ AppleTalk networks

Since ModemWorks is based on a modular architecture, support for additional computers and hardware may become available as new modules are developed.

Internal Modem Cards

Internal modems mimic the combination of a Super Serial Card and a Hayes-compatible modem. ModemWorks should be treated as if just such a combination exists in the computer with an internal modem card. Read the sections describing the settings for the Super Serial Card and external modems.

Super Serial Card

To use ModemWorks with a Super Serial Card, the cable shown on the opposite page should be used with your modem. This wiring scheme provides hardware handshaking flow control (required by today's faster modems), data carrier detect, and support for DTR disconnections.

The special cable allows ModemWorks to detect the actual carrier signal on the phone line. Without it, ModemWorks could not know when the modem has disconnected with a remote modem. The diagram shows that DCD is tied to the DSR line. This is required by the Super Serial Card— it cannot communicate with the modem unless it believes that a carrier is present. The actual DCD signal from the modem is mapped to the DSR line at the serial port. ModemWorks reads this line for carrier status.

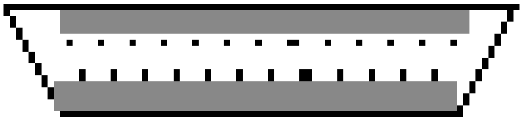
The Super Serial Card's jumper block must point to **Modem**, and the significant DIP switches should be set as follows:

◊ Communications mode (not Printer mode)
◊ Interrupts are on
◊ RS-232C signals are on

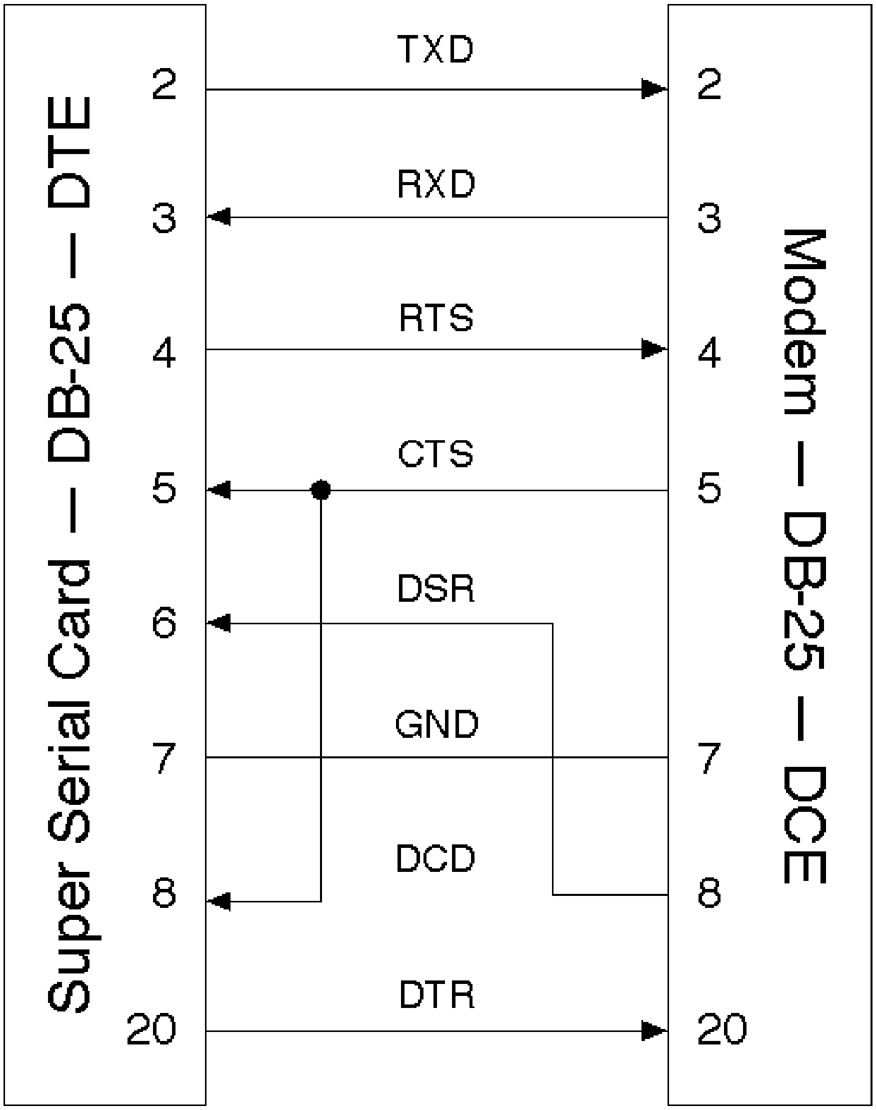
You may want to set the default operating speed of the card to the highest speed that your modem supports.

NOTE: To use a Super Serial Card in an Apple IIGS, make sure to visit the Control Panel. Set the slot containing the serial card to ***Your Card****.*

NOTE: The overhead in managing AppleTalk networking causes the Super Serial Card to lose data, especially at high speeds. It is advised that you use the built-in Modem port for communications on the Apple IIGS If you must communicate using the Super Serial Card, turn AppleTalk off.

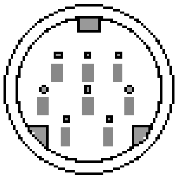


DB-25 Connector

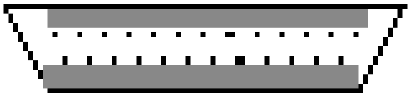


Apple IIGS Modem or Printer Port

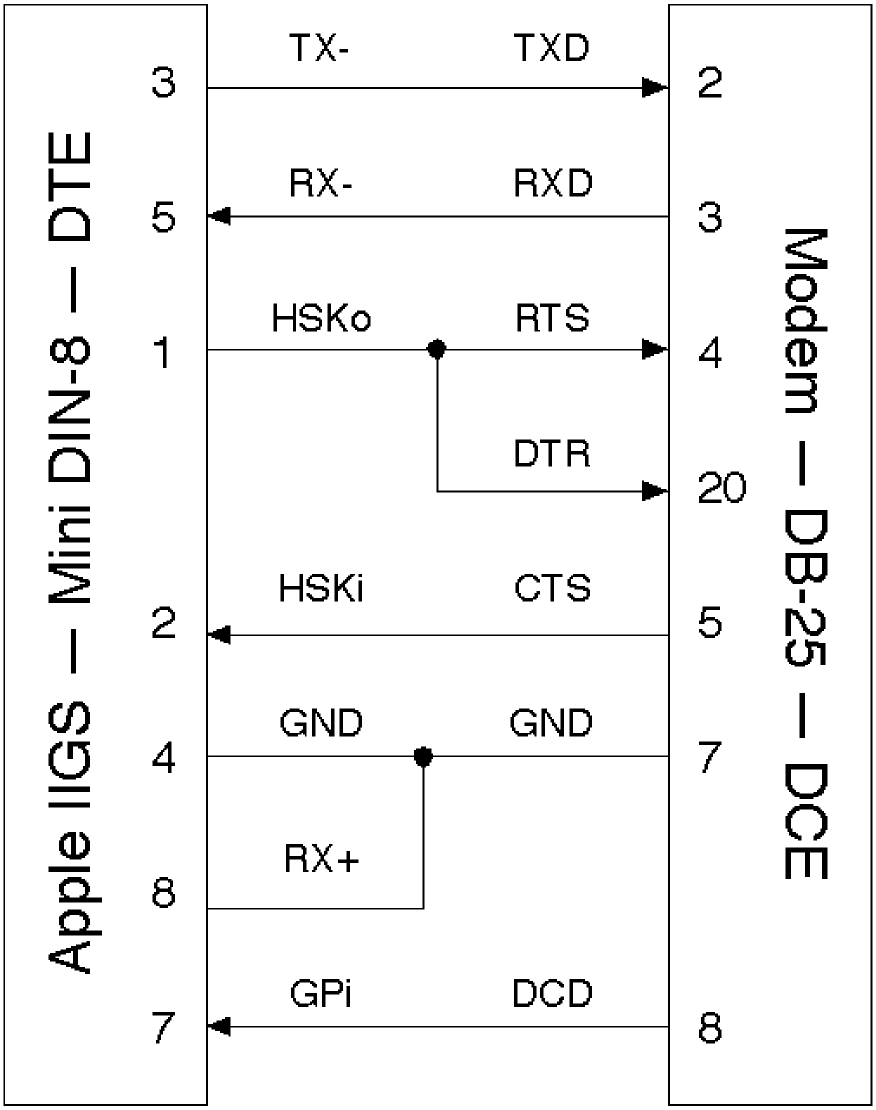
To use ModemWorks with the Apple IIGS built-in Modem or Printer ports, the following cable is required. This design is standard among high-end communications products for the Apple IIGS and Macintosh (which has a similar port design).



Mini DIN-8 Connector



DB-25 Connector



This cable provides true data carrier detection (DCD). It also supports hardware handshaking by connecting the Apple IIGS's two handshaking lines (HSKi and HSKo) to the modem's CTS and RTS lines. If hardware handshaking is not used, this wiring optionally supports DTR for disconnecting. The DTR wiring option is not required if the cable is to be used with only hardware handshaking.

WARNING: Since this cable allows the Apple IIGS's output handshake line to control two different modem signals, problems may occur if the modem is configured to respond to both RTS and DTR signals from the computer. With this cable, the modem must be configured to recognize only RTS or only DTR—not both at the same time. If hardware handshaking is used, the modem should be configured to ignore DTR.

The transmit-data and receive-data lines of the Apple IIGS serial interface conform to the EIA standard RS-422, which differs from the more commonly used RS-232-C standard. An RS-422 signal is less prone to noise and interference, and degrades less over distance, than an RS-232-C signal. By grounding the positive side of each RS-422 receiver (RX+) and leaving the positive side of each transmitter (TX+) disconnected, as illustrated in the diagram, the cable essentially converts to EIA standard RS-423, and can be used to communicate with most RS-232-C devices over distances up to fifty feet.

Visit the Classic Desk Accessory menu by pressing command-control-esc. Open the Control Panel, and select Modem Port or Printer Port depending on the port your modem is connected to. Make sure the following items are set as indicated:

```
√ Device Connected: Modem
  DCD Handshake: No
√ DSR/DTR Handshake: Yes
```

ModemWorks can control the other settings.

NOTE: If you do not use the recommended cable, as shown on the opposite page, set the ***DSR/DTR Handshake*** *item to* ***No****.*

External Modems

For proper operation with ModemWorks, your external modem should be configured to these settings (if supported):

◊ Send result codes
◊ Report the actual carrier status (support DCD)
◊ Enable command recognition

If you are using a hardware handshaking cable:

◊ Enable bidirectional RTS/CTS flow control
◊ Ignore Data Terminal Ready (DTR)—*Apple IIGS only*

Otherwise, if your are *not* using a hardware handshaking cable:

◊ Disable bidirectional RTS/CTS flow control
◊ Enable Data Terminal Ready (DTR) for disconnecting

Newer modems use terminal mode commands to set these options, to be saved in the modem's built-in memory. Older modems use DIP switches for these configurations. See your modem's manual for details.

High Speed Modems

ModemWorks provides support for high speed modems rated at 9600 bps or faster, or 2400 bps with data compression. For maximum performance, ModemWorks prefers that your computer and modem operate at the same speed all the time with a hardware handshaking cable. This is done by "locking the port" at the highest speed your computer and modem can handle. Although, your modem might be advertised as a 9600 bps modem, it might be able to communicate with your computer's serial port at 19,200 bps.

The Apple IIGS has a maximum port rate of 19,200 bps with standard software. The Apple Super Serial Card tops out at 9600 bps, but can be used at 19,200 bps on accellerated systems.

Custom Modem Control

As shipped, ModemWorks includes files containing information on nearly 50 modems. Called *modem capability files,* or *modemcaps,* they describe each modem's tolerances and features. You will find them in the **Modemcaps** directory on your ModemWorks disk. (Read the **Contents** file in this directory for a description of each modemcap file.)

A modemcap is preset to take full advantage of a modem's features. That is, the modem will operate at a fixed speed, at the highest rate supported, through the use of a hardware handshaking cable.

NOTE: Depending on your computer, serial interface, and cable, your system may not be able to accomodate the full abilities of your modem. For example, a stock Apple IIe may not be able to keep up with the data flow at 19,200 bps. You may have to make adjustments to your modem's modemcap file to provide compatible operation with your system.

ModemWorks comes with a modemcap file editor, called **MC**, located in the **Samples** directory. **MC** lets you change existing modemcap files as well as create new ones to the specifications required by your system, or to your own personal preferences.

Using ModemWorks

This chapter introduces you to the sample programs that come with ModemWorks. Before you can begin using them, you need to tell ModemWorks about your computer, modem type, and settings they require. This is done by using the Configure program provided on your ModemWorks disk. Configure stores your settings so that future sessions with ModemWorks won't require you to go through the configuration steps.

Starting Up

To begin your first session with ModemWorks, follow these steps:

1. **Insert a backup copy of ModemWorks into your disk drive.**

2. **Restart your computer, or use the Finder or a program selector to run the Startup program on the ModemWorks disk.**

   The ModemWorks Main Menu appears.

3. **Choose the first item, *Read Important Notes*.**

   This displays important information about ModemWorks that you won't find in this manual. Read it now to learn about changes, additions, last minute notes, and other late-breaking news.

4. **Choose *Configure ModemWorks***

   The ModemWorks configuration program appears. Follow the on-screen instructions for setting things up. You'll select your type of computer, the kind of communications port to use, your external modem and slot, type of printer port and slot, and kind of terminal emulation you want to use as a default.

5. **Choose *Install ModemWorks***

   The Install program loads in the various modules that make up ModemWorks, and then attempts to initialize your modem. You'll want to make sure your modem is connected, configured properly, and turned on. If the installation is successful, the *Applesoft Program Launcher* is activated, allowing you to select a sample ModemWorks program to run.

NOTE: If the installation is not successful, and everything is configured and connected correctly, the problem might be due to the default modemcap settings for the modem you've selected. As noted in the previous chapter, modemcap files are set up for optimum performance. This might be incompatible with your hardware, especially if you're using a Super Serial Card and a stock Apple IIe. You may need to customize the modemcap settings to agree with your hardware, such as reducing the modem's highest speed setting.

Although the installation may have failed, ModemWorks is resident in memory, so you can use MC, the modemcap editor. From the Main Menu, choose the ***Quit*** *option. This activates the* Applesoft Program Launcher. *Open the Samples directory. Then open the MC program. See the* ***Modemcap Editor*** *section in this chapter for details on using MC.*

Applesoft Program Launcher

The *Applesoft Program Launcher* looks and works surprisingly like the Apple IIGS standard file selection dialog box. It allows you to run any BASIC program, ProDOS-8 system application, or GS/OS 16-bit system application. It can also display the contents of text files. If you've never used an Apple IIGS or Macintosh, don't worry. Working with the Launcher is easy.

The Launcher presents a dialog box with a scrolling list of filenames and four buttons marked Disk, Open, Close, and Quit.

Use the ↑ and ↓ arrow keys to move through the list of files.

Once you have selected the desired file, press return (Open). Opening a directory (DIR) file displays a new list of files.

To back up to the previous directory, press esc (Close).

To switch to a different disk, press tab (Disk).

To quit, press command-Period (Quit).

Sample Programs

To demonstrate ModemWorks' features, these sample programs are included in the Samples directory:

IT. InteleTerm is a full-featured terminal program that employs all of ModemWorks abilities to provide terminal emulation and file transfer protocols. InteleTerm incorporates a powerful scripting language. A script is a file containing commands to perform sophisticated tasks for you automatically, such as logging into a host computer, downloading files, and then logging out. See the IT.doc file for instructions on using InteleTerm.

Compile. A companion to InteleTerm, Compile reads in script source files and generates output compatible with InteleTerm's scripting feature. The IT.doc file describes the scripting language, and Script.src is a sample script for your perusal.

IO.Changer. This program shows how to invoke the various input/output settings in ModemWorks, including display widths.

MC. This is the Modemcap Editor. It is described in detail in the next section.

Mini.BBS. Bulletin board systems are ModemWorks' specialty. This program lays the groundwork for a larger BBS with all the options. It shows how to trap carrier loss, print text files, redirect input and output, handle password input, trap cancel keys, and other BBS-related tasks.

TC. The Termcap editor creates or modifies terminal capability files. Termcap files are loaded into ModemWorks' Console module to provide full screen terminal emulation.

Time.Out. A BBS must be able to disconnect a caller who has been inactive for a certain amount of time. The Time.Out program shows how to trap for inactivity using ModemWorks.

Trace.Demo. This program demonstrates a valuable feature of ModemWorks: the ability to protect your computer in the event that your program ends unexpectedly. Should an unauthorized BBS caller enter Applesoft BASIC or the machine language monitor program, ModemWorks can hangup, remove ModemWorks, and reboot the computer.

Modemcap Editing

As described earlier, the Modemcap Editor (MC) is used to create or change modemcap files—the templates that describe the capabilities of a modem.

When MC is first run, it displays information for an old fashioned 1200 bps modem with default settings. A pointer (>) indicates the current item. Use the ↑ and ↓ arrow keys to move from item to item. A row of "buttons" on the bottom of the display control modemcap file management functions.

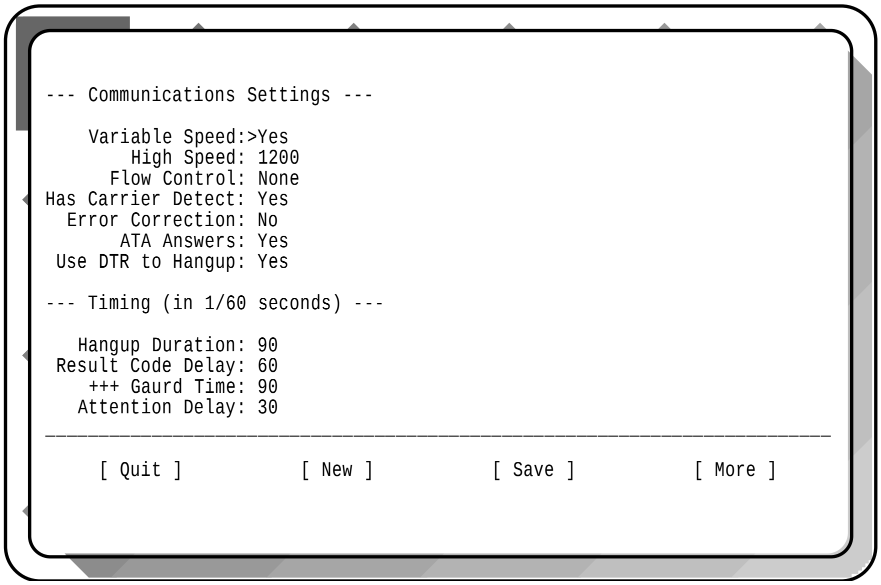
[Quit] Quits MC. If changes were made to the current modemcap, you're given the chance to save them.

[New] Work with a different modemcap file. This prompts you for a modemcap file name. If the file you enter exists, it is loaded in. If the file does not exist, a new modemcap file is created with the name specified.

[Save] Saves changes to the modemcap file. You have the option of changing the filename before the modemcap is saved.

[More] Switches to the alternate screen of modemcap information. The primary screen includes communications settings and timing information, while the alternate screen displays the modem's commands.

Pressing the first letter of a button's name selects that button. For example, press **Q** to quit. In addition to the **More** button, the tab key toggles between the two modemcap information screens.



```
--- Communications Settings ---

    Variable Speed:>Yes
        High Speed: 1200
      Flow Control: None
Has Carrier Detect: Yes
  Error Correction: No
       ATA Answers: Yes
 Use DTR to Hangup: Yes

--- Timing (in 1/60 seconds) ---

   Hangup Duration: 90
 Result Code Delay: 60
    +++ Gaurd Time: 90
   Attention Delay: 30
__________________________________________________________________________

     [ Quit ]          [ New ]          [ Save ]          [ More ]
```

MC's primary display screen.

Items on the primary screen are changed by using the ← and → arrow keys to scroll through a list of acceptable settings:

Variable Speed. Older 300, 1200, and 2400 bps modems typically require software to adjust the speed of the computer's serial port to match the modem's connection rate. Failing to match the port and modem speeds results in garbled characters. For modems like these, this setting should be **Yes** (vary the port speed according to the connection). Newer modems, such as high-speed and data-compression types are normally set to operate with the computer at a fixed, high speed, such as 19,200 bps. For newer modems, this setting should be **No** (do not vary the port speed).

High Speed. This number indicates the highest speed at which your modem and computer can communicate. Note that some modems can operate at 19,200 bps even though their throughput rating is lower.

Flow Control. This item should be set to **RTS/CTS** if a hardware handshaking cable is used. Otherwise, this item should be set to **None**.

Has Carrier Detect. If a properly wired cable is being used, one which supports Data Carrier Detect (DCD) reporting, set this item to **Yes.** ModemWorks cannot work as designed unless it can detect the carrier signal on the phone line.

Error Correction. Set to **Yes** if your modem includes commands that enable or disable error correction.

ATA Answers. Set to **Yes**, unless you have a very old modem which cannot answer a ringing phone line with ATA. Examples: Apple Modem 300, USRobotics Password and Auto-Dial 212.

Use DTR to Hangup. If your modem is being used on an Apple IIGS with a cable that supports hardware flow control, set this to **No**. Normally, this item should be set to **Yes.**

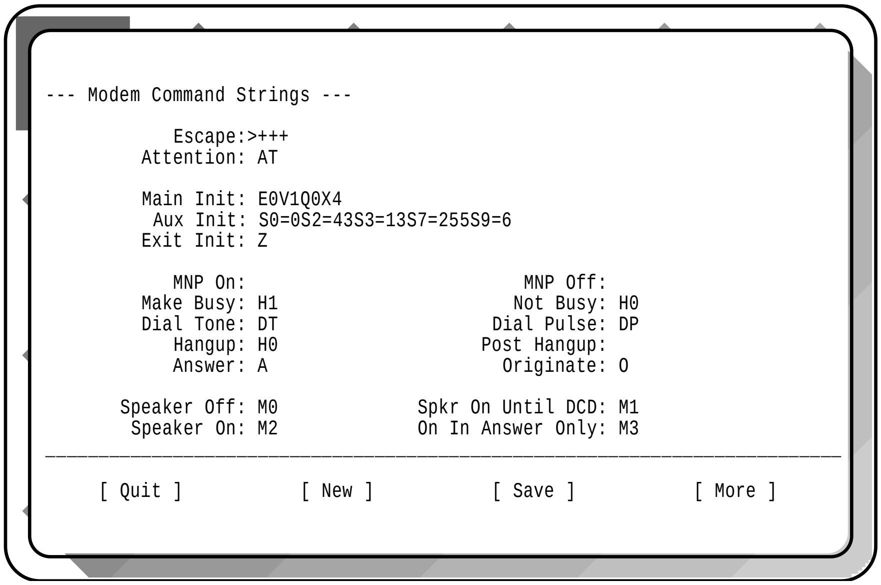
Hangup Duration. The time that DTR is held low to force the modem to disconnect. This setting is used only if **Use DTR to Hangup** is set to **Yes**.

Result Code Delay. The amount of time that the Modem module waits for a response from the modem after sending a command.

+++ Gaurd Time. The delay before and after the escape characters are sent (+++) to disconnect or to enter the modem's command mode.

Attention Delay. The delay before a new command is sent, as some modems cannot handle a command immediately following the response from a prior command.

Items on the alternate screen are changed by pressing return, entering new characters, and pressing return again when done.



```
--- Modem Command Strings ---

            Escape:>+++
         Attention: AT

         Main Init: E0V1Q0X4
          Aux Init: S0=0S2=43S3=13S7=255S9=6
         Exit Init: Z

            MNP On:                          MNP Off:
         Make Busy: H1                      Not Busy: H0
         Dial Tone: DT                    Dial Pulse: DP
            Hangup: H0                   Post Hangup:
            Answer: A                      Originate: O

       Speaker Off: M0             Spkr On Until DCD: M1
        Speaker On: M2             On In Answer Only: M3
___________________________________________________________________________

     [ Quit ]           [ New ]           [ Save ]           [ More ]
```

MC's alternate display screen.

The items in the alternate display screen should be set as appropriate for your modem. They must agree with any settings made in the primary screen. For example, if the **Error Correction** item is **Yes**, then the **MNP On** and **MNP Off** items must contain the modem's commands for adjusting error correction. Likewise, if **Flow Control** is **RTS/CTS**, then the **Main Init** item should include the command that enables RTS/CTS hardware flow control.

Main Init. ModemWorks requires that your modem respond to commands with verbose result codes. Thus, this item should include commands such as Q0 (not quiet) and V1 (verbose). Echo during command input is not required, so E0 (no echo) is advised. If your modem can report BUSY signals, be sure to include X4, or the appropriate command to enable BUSY signal reporting. With many modems, you may only need to include &F to have the factory settings enabled, plus any additional commands required.

Aux Init. This item is typically used for setting the modem's S registers. Most factory defaults are fine, except for S7 which controls when the modem gives up on a connection attempt (usually after 30 seconds). Since ModemWorks can control this timing itself, S7 should be disabled by setting it to 255.

Exit Init. When ModemWorks and its modules are removed, this command is sent to your modem.

MNP On and **MNP Off**. These commands switch your modem’s error correction mode on and off.

Make Busy and **Not Busy**. These two commands allow your modem to make your phone line busy.

Dial Tone and **Dial Pulse**. These commands are used when dialing a number with either Touch-Tone™ or pulse dialing.

Hangup and **Post Hangup**. These commands are sent before and after a disconnection request.

Answer and **Originate**. The Answer command is used when answering a ringing line to establish a new connection. The originate command is used by the dialing end to resume a connection.

Speaker. Four commands control various speaker attributes.

Disk Contents

Now that you've had a chance to get acquainted with some of the ModemWorks sample programs, you may want to explore the rest of the ModemWorks disk. Here's a road map:

ModemWorks

- System Software—ProDOS, BASIC.System
- MW.Notes—Important news and information
- MW.Protocols—Information on file protocols
- OMM.Loader—Object Module Manager
- Startup—ModemWorks Main Menu

Modemcaps

- Modem capability templates
- Contents—Describes the modemcaps files

Modules

- ModemWorks and AmperWorks modules
- Contents—Describes the modules

MW.Utils

- Configure—Disk configuration program
- Install—ModemWorks installation program
- Launcher—Program selector

Samples

- Sample programs

Sources

- Source code for select programs

Termcaps

- Terminal capability templates
- Contents—Describes the termcap files

ModemWorks Modules

This chapter describes the kinds of modules that comprise ModemWorks. It shows how to load and install ModemWorks for use by your own programs, efficiently use modular extensions, such as file transfer protocols, and remove ModemWorks and its modules when your program is done.

Modular Anatomy

It is said that the whole is greater than the sum of its parts. That is certainly true with ModemWorks. It takes a number of integrated modules to make ModemWorks functional. Each piece is responsible for certain related tasks, and if any piece is missing, things fail to work properly.

ModemWorks' various kinds of modules are:

ModemWorks The main interface between an Applesoft program and the modules that perform various communications tasks. It manages character input and output and other I/O related duties, as well as servicing ampersand commands and carrying them out with the help of other modules.

Time Tool Software timing and clock driver (e.g. Time and TimeGS). Used by every ModemWorks module, the Time Tool provides accurate timing needed in a communications environment. It also is the interface to a clock device.

Port Tool Low-level communications I/O (e.g. SerialGS). This module is the interface between other software and a high-speed communications port.

Modem Tool Modem device driver. This module communicates with a modem device connected to a port. It calls upon the Port Tool.

Printer Tool Printer device driver (e.g. Printer and ParallelPrinter). This tool handles output to a printer through its own port driver.

Console Tool Keyboard and screen driver. Handles input and output with the computer's keyboard and screen. This module is also responsible for terminal emulation processing.

Send Tool File transfer protocol module, for sending files (e.g. SX and SZ).

Receive Tool File transfer protocol module, for receiving files (e.g. RX and RZ).

Terminal Tool Terminal mode module. This module provides basic terminal mode functions. When used with a Console Tool, full screen terminal emulation is available.

Module ID Numbers

Each kind of module is identified by a unique number. For example, all Time Tool modules (i.e., Time and TimeGS) have an ID of 29812 ($7474). Here is a list of the standard module types and their ID numbers in hexadecimal and decimal formats:

Kind	Hex	Dec	Name
AmperWorks	$7761	30561	AmperWorks
Console Tool	$7463	29795	Console
Printer Tool	$746C	29804	Printer, ParallelPrinter
Modem Tool	$746D	29821	Modem
ModemWorks	$776D	30573	ModemWorks
Port Tool	$7470	29808	Serial, SerialDL, SerialGS
Receive Tool	$7472	29810	RX, RZ
Send Tool	$7473	29811	SX, SZ
String Storage	$5D73	23923	Store, Store512, StoreGS
Terminal Tool	$6D74	28020	Terminal
Time Tool	$7474	29812	Time, TimeGS

The Object Module Manager can reference modules by ID number as well as by an index number. See the Object Module Manager manual for more details.

Sample Installation

ModemWorks is based on the Object Module Manager (OMM) environment. To use ModemWorks and its modules, you should be familiar with the OMM's module installation and handling commands. Refer to the Object Module Manager manual for more details.

A typical installation for a bulletin board system running on an Apple IIGS might begin like this:

```
10 PRINT CHR$(4) "-OMM.Loader"
20 & LOAD GET "ModemWorks"
30 & LOAD GET "TimeGS"
40 & LOAD GET "SerialGS"
50 & LOAD GET "Console"
60 & LOAD GET "Modem"
```

Line 10 installs the Object Module Manager, allowing Lines 20 through 70 to load a basic set of ModemWorks modules. This set is all that is needed to begin a typical communications session. Other modules such as Send and Receive tools can be loaded by the BBS only when they're needed.

After the modules are loaded, the next step is to install a modemcap template:

```
70 PRINT CHR$(4) "BLOAD USR.HST, A768, TØ"
80 & MSET (768)
```

Line 70 loads a modemcap template into memory at location 768. In Line 80, the &MSET command instructs the Modem module to copy the template information from location 768 into its own storage area.

Next, the communications port and modem are initialized:

```
90 PRINT "Initializing modem...";
100 & SLOT (2), OK
110 IF NOT OK THEN PRINT "failed!" : END
120 PRINT "OK"
```

Line 100 tells ModemWorks to initialize the Port Tool (SerialGS) in slot 2. Since a Modem Tool is loaded, the modem is also initialized. If it is successful, a non-zero value is returned in the OK variable.

Finally, the ModemWorks input/output (I/O) system is activated:

```
130 & SCRN (2)
```

Line 130 initializes the Console module for an 80-column display. The I/O system, an integral part of ModemWorks, is responsible for transparent character I/O, carrier loss and cancel key trapping, flow control, software crash protection, and inactivity timing.

Efficient Module Usage

Always use the fewest modules needed. This way, you give maximum workspace to BASIC programs. For example, to transfer a file, load in a transfer module. After the transfer is completed, remove the module to reclaim memory. Example:

```
10 & LOAD GET "SX"
20 & LOAD PEEK 0, LAST
30 & SND ("Test.File"), RESULT
40 & LOAD FRE LAST
```

Line 10 loads the SX module. Line 20 asks the OMM to return the number of modules loaded and places it into the LAST variable. Line 30 sends the Test.File file. And, Line 40 removes SX, the last module loaded.

This example is not bullet-proof. If an error occurs in Line 30, like Test.File not being found, program flow is diverted to your error handling routine, and the SX module remains in memory. If your program attempts the transfer again, re-running Line 10 would give an error since the SX module is already loaded. There are two solutions to this problem.

First, use the non-error generating form of &LOAD GET:

```
10 & LOAD GET "SX", ERR
```

With this form, an error code is returned in the ERR variable. You can check its value for 19, a duplicate file error, indicating that a copy of the module is already in memory. (See Appendix B, **Error Codes.**) One would usually ignore error 19.

The second solution is to blindly use &LOAD FRE with a Send Tool's ID number, 29811 ($7473), prior to using &LOAD GET. There are no ill effects if the OMM is told to remove a non-existent module, so doing this is perfectly safe:

```
5 & LOAD FRE 29811
10 & LOAD GET "SX"
```

This would always load a fresh copy of a module into memory.

Shutting Down

To remove ModemWorks, use the OMM's &LOAD FRE command. You can use it to remove only certain modules (e.g. everything except AmperWorks), or you can use it to remove everything, including the OMM.

A typical shutdown procedure looks like this:

```
10 PRINT CHR$(4) "CLOSE"
20 & LOAD FRE
30 PRINT CHR$(4) "BYE"
```

This process involves closing any open files, removing the Object Module Manager and any modules loaded, and then quits BASIC.System.

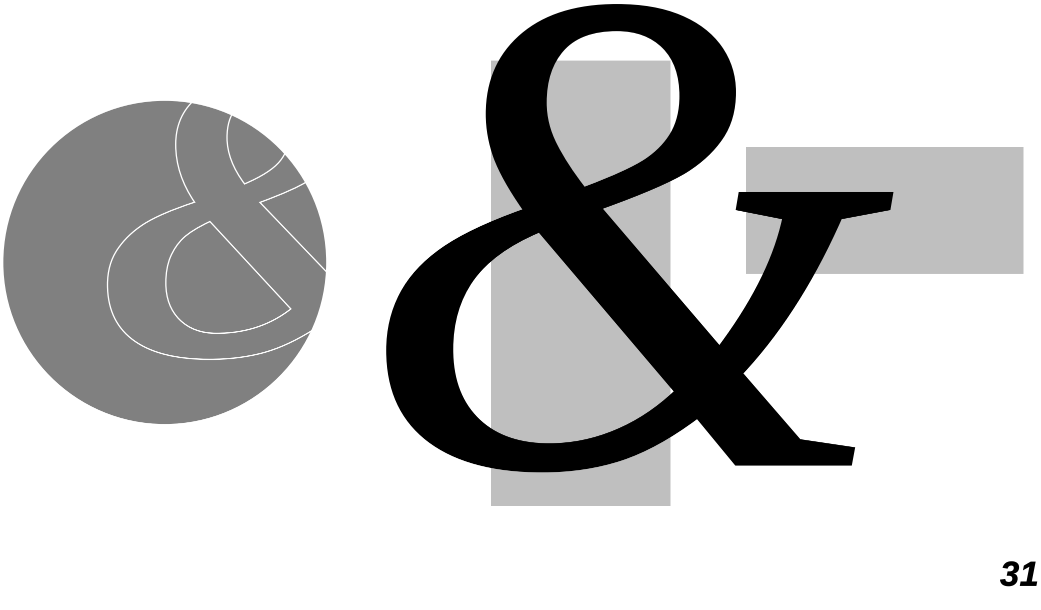
WARNING: Failing to shutdown ModemWorks, especially Serial or SerialDL, can cause your system to crash, because it installs interrupt handlers with ProDOS-8. They must be removed before using BASIC.System's BYE command.

Removing ModemWorks while your modem is connected will keep the connection intact.

ModemWorks Commands

This chapter describes the ModemWorks commands. Abbreviations are used for different types of arguments that these commands require. Refer to Appendix E, **BASIC Syntax**, for an explanation of the abbreviations.

Most ModemWorks commands are serviced by the ModemWorks module. A few commands are serviced by other modules. For example, the &TSET command, which sets up terminal emulation for a particular terminal, is handled by the Console module. In addition, the ModemWorks module relies on the functions of other modules. In particular, ModemWorks' &SND and &RCV commands, which initiate protocol transfers, require a transfer module such as SX or RX to be resident in memory.



Command Summary

ModemWorks consists of the following commands:

BEEP	Set pitch and duration of the beep
BREAK	Send a break
CALL	Dial a phone number
CHK	Toggle carrier checking
CLEAR	Flush keyboard and modem buffers
FAST	Set the Apple IIGS to fast speed
FN	Return ModemWorks function results
GOTO	Move the cursor on the local console
HANGUP	Disconnect modem, drop carrier
IN#	Set the input switch
INT	Interrupt key management
IOCTL	Perform a terminal emulation function
MSET	Set modem attributes
NULLS	Nulls after a carriage return
ON HANGUP	Set carrier loss / inactivity handler lines
ON INT	Set interrupt key handler line
PAGE	Manage screen paging
PICKUP	Lift the phone and send a carrier
PR#	Set the output switch
PUT	Send a string of characters to the modem
RCV	Receive a file with protocol
SCRN	Select the type of output display
SLOT	Set the communications slot
SLOW	Set the Apple IIGS to normal speed
SND	Send a file with protocol
SPEED=	Set the modem and/or port speed
TAB	Set tab character attributes
TERM	Enter terminal mode
TIMER	Inactivity timer management
TRACE	Set crash tracing protection
TSET	Set terminal emulation
USR	Perform modem and port procedures
WAIT	Wait (delay loop)
WAIT FOR	Wait for a pattern of text from port
WAIT ... CALL	Wait for a phone call
WAIT ... CARRIER	Wait for a modem connection

BEEP

& BEEP [(*numexp1*, *numexp2*)]

Sets the pitch (*numexp1*) and duration (*numexp2*) of the speaker's beep. A pitch of zero disables speaker output, which is handy if you can't turn it off altogether. Values for both the pitch and duration are from 0 to 255.

With no arguments, &BEEP simply beeps the speaker. It does not send a Control-G to the modem, however. Use

```
      PRINT CHR$(7);
```

to send a beep to both the local and remote computers.

Sample Program

```
10 DUR = 10
20 FOR PITCH = 1 TO 50
30  & BEEP (PITCH, DUR): REM Set up the beep
40  & BEEP: REM Now beep the speaker
50 NEXT
60 & BEEP (21,65) : REM Normal settings
```

BREAK

& BREAK

Sends a break signal. A *break* is a special attention-getting tone sent by the modem, usually interpreted as a *stop*. When a host computer receives a break, it usually stops what it is doing to return to the command level.

CALL

& CALL *strexp*

Dials the phone number in *strexp*. All characters in *strexp* are sent, so a modem may interpret some of them as special dialing commands. See your modem's instruction manual for more details.

Sample Program

```
10 BBS$ = "1 (619) 670-5379"
20 PRINT "Dialing: " BBS$;
30 & CALL BBS$
```

ModemWorks uses Touch-Tone™ dialing unless the phone number begins with the letter **P**.

If *strexp* begins with the capital letters **AT**, ModemWorks sends only the **AT** and all characters following to the modem. This allows you to issue a modem command before dialing. Example:

```
      & CALL "AT\N0DT670-5379"
```

This issues \N0 before dialing with DT.

Also See

PICKUP, HANGUP

CHK

```
& CHK ON
& CHK STOP
```

With a properly wired cable, ModemWorks can detect the presence or absence of a carrier signal—the tones that indicate that two modems are connected over a phone line. A BBS must know if the carrier signal is lost so it can accept the next caller.

&CHK switches carrier detection ON or STOPs it. If carrier checking is on and the connection is lost, the program flow is diverted to the line number specified by &ON HANGUP GOTO. This happens only if ModemWorks detects the loss of a carrier which once used to be present. So, you can use your programs from the local console; ModemWorks won’t “log you out”.

CAUTION: Unpredictable results occur when a program activates carrier checking, then runs new other programs that do not specify a new carrier loss handler with &ON HANGUP GOTO. Set up carrier loss handling before printing anything or getting any input. This ensures that your programs can handle carrier loss properly.

Sample Program

```
10 & ON HANGUP GOTO 40
20 & CHK ON
30 INPUT A$: PRINT "Still on-line": GOTO 30
40 PRINT "Lost Carrier"
```

Sample Run

The program loops in Line 30. When the carrier is lost, execution branches to Line 40 as specified in Line 10.

Also See

FN, ON HANGUP GOTO

CLEAR

```
& CLEAR
```

Clears the keyboard and modem input buffer of any characters currently waiting to be processed.

NOTE: It is a good idea to flush the input buffer before asking the user for a response which might result in some serious disaster. For example, if your program asks the user if he wants to delete a file, use &CLEAR before your input statement.

Also See

FN

FAST

& FAST

Sets the Apple IIGS to run at its fastest speed.

Also See
SLOW

FN

& FN ***numexp, numvar***

Requests information from various ModemWorks functions. A numeric expression is used to identify the type of information requested, and ModemWorks returns it in *numvar*.

FN	Description	Returns
0	**Modem Online**	0 = Offline, 1 = Online
1	**Ring**	0 = No ring, 1 = Ringing
2	**Modem Speed**	0=300, 1=1200, 2=2400, 3=9600, 4=19200
3	**Mode**	0 = Answer, 1 = Originate, 2 = Quiet
4	**<reserved>**	
5	**Port Input**	0 = Nothing, >127 = Character
6	**Modem Type**	0 = None, 1 = Internal, 2 = External
7	**Port Byte**	0-255 = Raw ASCII value from &FN 5
8	**Port Speed**	1-15 = Speed index (See SPEED=)
9	**Modem Speed**	1-15 = Speed index (See SPEED=)
10	**Scan Input**	0 = Nothing, 1-127 = Port character, 128-255 = Keyboard character
11	**Port DCD**	0 = No carrier, 1 = Carrier present

GOTO

& GOTO ***numexp1, numexp2***

Moves the cursor (the print position) to coordinates you specify, like HTAB and VTAB combined. *numexp1* specifies a column, 0 through 79. *numexp2* specifies a row, 0 through 23. The upper left corner coordinates of the screen are 0,0 not 1,1. This makes it easy to use memory locations 36 and 37 which contain the cursor’s horizontal and vertical coordinates.

HANGUP

```
& HANGUP
```

Disconnects with a remote computer, placing the phone switch on-hook. (This does not trigger &ON HANGUP GOTO.)

Also See
PICKUP

IN#

```
& IN# numexp
```

Selects input from the modem and/or keyboard during standard input. *numexp* is a number from 0 through 3, and has these effects:

numexp	Console Input	Modem Input
0	No	Yes
1	No	No
2	Yes	Yes
3	Yes	No

By default, ModemWorks accepts only keyboard input.

Also See
PR#, SCRN

INT

```
& INT (numvar)
& INT ON
& INT STOP
& INT = numexp [,...]
& ON INT GOTO line
```

These commands manage ModemWorks' interrupt key feature.

`&INT(numvar)` Returns the ASCII character code (0-127) of the key used to generate a cancel-key interrupt.

`&INT ON` Enables cancel key recognition for the input devices (modem or keyboard) selected by the &IN# command.

`&INT STOP` Disables cancel key recognition.

`&INT=` Defines a list of ASCII character codes to use as cancel keys recognized during input and output. A maximum of eight numeric expressions may be listed, each separated by a comma.

`&ON INT GOTO` Specifies the line to branch to when a cancel-key interrupt occurs.

Unpredictable results occur if &INT ON is executed without previously issuing &ON INT GOTO.

Sample Program

```
10 & INT= 3, 27, 127
20 & ON INT GOTO 50
30 & INT ON
40 PRINT "*"; : GOTO 40
50 PRINT : & INT (CH)
60 PRINT "The interrupt key was ASC(" CH ")"
```

Sample Run

Line 10 sets up three cancel keys, Control-C, Escape, and Delete. It infinitely prints asterisks until a cancel key is pressed, then program flow diverts to Line 50. &INT (CH) assigns CH the ASCII value of the cancel key that was pressed.

Also See

IN#

IOCTL

& IOCTL (*numexp*) [,*numvar*]

Pronounced *I/O Control*, &IOCTL performs a variety of functions dealing with full-screen terminal emulation. Its first form,

```
& IOCTL (2)
```

causes control function 2, a screen clear, to be performed. If online, the corresponding terminal emulation codes for function 2 are sent to the port as well. The effect is that the screen clears on the Apple, as well as on the screen of a remote computer.

&IOCTL takes at least one argument, an I/O control code. Function 1, which positions the cursor on the screen, requires two additional arguments specifying the cursor coordinates:

```
& IOCTL (1,4,9)
```

This positions the cursor at column 4, row 9. Rows and columns are numbered starting with 0, so the above example places the cursor on the fifth column of the tenth row.

&IOCTL can be used in a passive mode to determine if the current terminal supports a given function. For example:

```
& IOCTL (4),CLEOL
```

This performs no emulation, but instead returns a value of 0 or 1 in the numeric variable CLEOL. From the table on the next page, note that &IOCTL (4) is the *clear to end of line* function. If the current terminal can perform a clear to end of line sequence, the CLEOL variable contains 1. If the terminal does not provide this feature, zero is returned. When offline, a result of 1 is always returned.

&IOCTL respects the output mode set by the &PR# command.

Code	Function
1	Position the cursor at X,Y coordinates
2	Clear the screen and home the cursor
3	Clear from cursor to end of screen
4	Clear from cursor to end of line
5	Insert line
6	Delete line
7	Insert a space at the cursor
8	Delete the character under the cursor
9	Place the cursor at the “home” position
10	Beep the speaker
11	Carriage return (no linefeed)
12	Inverse video
13	Normal video
14	Scroll the screen up
15	Scroll the screen down
16	Move cursor up
17	Move cursor down
18	Move cursor right
19	Move cursor left
20	Move cursor to next tab stop
21	Move cursor to next tab stop using spaces
22	Erase the line the cursor is on
23	Turn on continuous insert mode
24	Turn off insert mode
25	Turn on underlining (inverse)
26	Turn off underlining (normal)
27	Turn on MouseText
28	Turn off MouseText

Also See

PR#, PUT, TSET

MSET

& MSET (*numexp*)

Sets the attributes of a modem based on a modemcap file at the address in memory specified by *numexp*. Once used, the memory which held the modemcap can be reused.

Sample Program

```
10 PRINT CHR$(4) "BLOAD Intel.9600EX, A768,
TØ"
20 & MSET (768)
30 PRINT "Intel 9600EX modem installed."
```

Lines 10 BLOADs a modemcap file into memory at location 768. Line 20 instructs the Modem module to use the information at 768 to control an external modem. Modemcap files have a file type of $00.

NULLS

& NULLS (*numexp*)

Specifies a delay, in 1/10 second increments, after a carriage return is sent to the modem during standard output. This delay approximates the effect of sending null characters. Nulls are often sent in data transmission to allow slower terminals time to catch up. *numexp* is a value from 0 to 255.

See Also

TAB

ON HANGUP GOTO

& ON HANGUP GOTO *line1* [, *line2*]

Specifies the program lines to GOTO upon losing a connection, or an inactivity time-out occurs. If ModemWorks detects that a connection has been lost, program flow is diverted to the *line1*. If an inactivity time-out occurs, program flow diverts to *line1*, unless the optional *line2* line is given. This command must be issued at the beginning of every BASIC program that comprises a BBS.

NOTE: Carrier checking or inactivity timing are not activated unless the &CHK ON or &TIMER ON commands have been issued first.

Sample Program

```
10 & ON HANGUP GOTO 50, 60
20 & CHK ON
30 & TIMER ON
40 INPUT A$: PRINT "Still on-line": GOTO 40
50 PRINT "Lost Carrier" : END
60 PRINT "Timed Out" : END
```

Sample Run

The program loops in Line 40. When the carrier is lost, execution branches to Line 50 as specified in Line 10. If an inactivity occurs, execution diverts to Line 60 (and the connection is maintained).

Also See

CHK, FN, TIMER

ON INT GOTO

& ON INT GOTO *line*

See INT.

PAGE

& PAGE CLEAR
& PAGE DEF *numvar*
& PAGE LEN *numexp*
& PAGE ON
& PAGE STOP
& PAGE$ = *strexp*
& PAGE

An extensive set of commands to manage screen *paging*. Paging is a feature that causes the computer to ask for a key press when the text screen fills. When you’ve read the screen, press a key and the text continues to scroll up the screen until it fills again.

Here is a summary of each PAGE command:

`&PAGE CLEAR` Clears the lines counter so that the user is not prompted until the screen fills again.

`&PAGE DEF` *`numvar`* Sets *numvar* to 0 or 1 based on the paging enabled state.

`&PAGE LEN` *`numexp`* Defines the page length in *numexp* lines. The number of lines should be the screen's height in rows minus one (the default is 23).

`&PAGE ON` Turns screen paging ON.
`&PAGE STOP` Stops screen paging.

`&PAGE$ =` *`strexp`* Defines the prompt string. The prompt is initially set to "—More—".

`&PAGE` Forces prompting immediately, as if the screen had filled.

Sample Program

```
10 & PAGE ON
20 & LIST "A.TEXT.FILE"
30 & PAGE STOP
```

Sample Run

Paging is turned on in Line 10. A text file, named A.TEXT.FILE, is displayed with the AmperWorks command, &LIST, in Line 20. While the file is being displayed, you are prompted with —More— when the screen fills. When the end of the file is reached, paging is turned off in Line 30.

If interrupt key handling is turned on, you can press **Q** or a cancel key at the —More— prompt to cancel the output, halting the display.

Pressing the minus sign key (-) at the —More— prompt disables paging, as if &PAGE STOP were issued.

PICKUP

& PICKUP [(*numexp*)]

Lifts the phone off-hook and transmits a carrier with the specified *numexp* mode. The following mode values are recognized:

0	Answer (BBS mode—default)
1	Originate (terminal mode)
2	Quiet (transmitter is silent)

Quiet mode is useful for making the telephone line busy.

Also See
HANGUP

PR#

& PR# *numexp*

Selects output to the modem and/or screen, or neither, during standard output. *numexp* is a number from 0 to 3, and has these effects:

numexp	Console Output	Modem Output
0	No	Yes
1	No	No
2	Yes	Yes
3	Yes	No

By default, ModemWorks sends output only to the Console.

Also See
IN#, SCRN

PUT

& PUT (*strexp*)

Sends characters in *strexp* to the port. The characters are sent regardless of the output setting, but not displayed on the Console. A carriage return is not sent at the end, unless explicitly provided.

&PUT can be used to send commands to external modems. For example, to adjust the dialing speed for most external modems, the following could be used:

```
        & PUT ("ATS11=40" + CHR$(13))
```

Certain commands can cause the modem to become inoperable with ModemWorks. If this happens, the &USR 6 instruction initializes your modem for use with ModemWorks.

Also See

PR#, USR

RCV

& RCV (*strexp1* [, *strexp2*]), *numvar* [, *strvar*]

Receives a file using a file transfer protocol, a special method of receiving files with error detection and correction. The actual protocol used depends on the type of Receive Tool module resident. The size of the received file is limited by available disk space. See the file **MW.Protocols** on the ModemWorks disk for details on each protocol.

The arguments &RCV accepts are:

strexp1 Pathname of the file to receive the data.

strexp2 (Optional). Contains flag characters specific to the transfer module being used.

numvar The status of the transfer is returned in this numeric variable. Typical values are:

0 No errors, transfer was successful
1 Transfer cancelled—too many errors
2 Transfer was cancelled by user

strvar (Optional). Contains the name of the received file if transmitted by the sender. When receiving a batch of files from a batch-mode sender, keep issuing the &RCV command until it returns an empty file name (*strvar* = “”).

If too many errors occur, the transfer is cancelled by ModemWorks. It is possible to manually cancel by pressing esc on the keyboard, or by receiving two Control-X characters from the remote computer.

NOTE: File transfer commands cannot be issued from immediate mode.

Also See
SND

SCRN

& SCRN (*numexp*)

Invokes various screen modes, responsible for managing transparent I/O with the modem, including features such as cancel-keys, tab expansion, carrier loss detection, and other functions related to input and output. Values for *numexp* are:

0 Apple I/O (disconnects ModemWorks I/O)
1 ModemWorks 40-column I/O
2 ModemWorks 80-column I/O
3 Reconnects ModemWorks I/O
4 Turns off printer output
5 Turns on simultaneous printer output

If many ModemWorks commands seem to be inoperative, it is probably because &SCRN has not been issued.

Also See
IN#, PR#

SLOT

& SLOT (*numexp1* [,*numexp2*])

Defines the slot (in *numexp1*) of your modem interface card, and optionally the slot (in *numexp2*) of your printer interface. This command also initializes the port and your modem, preparing it for use with ModemWorks. You must issue this command before using any other ModemWorks commands, otherwise unpredictable results may occur.

The Apple IIGS Modem Port is mapped to slot 2.

Also See
SCRN, USR

SLOW

`& SLOW`

Slows the Apple IIGS to run at its lowest speed, the speed of a normal Apple II running at 1MHz. This is useful when executing commands that require consistent timing from one computer to the next. For accurate delay loops without having to slow down the computer, use the &WAIT command.

Also See
FAST, WAIT

SND

& SND (*strexp1* [, *strexp2*]), *numvar*

Sends a file using a file transfer protocol, a special method of sending files with error detection and correction. The actual protocol used depends on the type of Send Tool module resident. See the file **MW.Protocols** on the ModemWorks disk for details on each protocol.

The arguments &SND accepts are:

strexp1 Pathname of the file to send.

strexp2 (Optional). Contains flag characters specific to the transfer module being used.

numvar The status of the transfer is returned in this numeric variable. Typical values are:

0	No errors, transfer was successful
1	Transfer was cancelled—too many errors
2	Transfer was cancelled by user

When sending a batch of files with a batch-mode protocol, keep issuing the &SND command for each file. When done, issue &SND with an empty filename string (equal to “”), and use the same flag characters.

If too many errors occur, the transfer is cancelled by ModemWorks. It is possible to manually cancel by pressing esc on the keyboard, or by receiving two Control-X characters from the remote computer.

NOTE: File transfer commands cannot be issued from immediate mode.

Also See
RCV

TAB

& TAB (*numexp*)

Sets tab character expansion or preservation. When a tab character (ASCII 9) is sent to the screen, ModemWorks moves the cursor to the next tab column. Tab stops are positioned at every 8th column.

Some terminal programs do not know what to do with tabs, unfortunately. When *numexp* is 1 (the default), ModemWorks converts tabs to the appropriate number of space characters when sending them to the modem. If *numexp* is zero, tabs are preserved.

Also See
NULLS

SPEED

```
& SPEED= numexp
```

Sets the total number of bits transmitted per second (or baud rate). Standard speed values for modem usage are (0 to 4):

0	300
1	1,200
2	2,400
3	9,600
4	19,200

These values are returned by &FN 2.

To communicate between two computers connected by a null-modem cable, &SPEED= can activate 15 different port speeds from 50 to 19,200 bits per second. Use one of the following 15 values and add 128:

1 = 50	6 = 300	11 = 3,600
2 = 75	7 = 600	12 = 4,800
3 = 110	8 = 1,200	13 = 7,200
4 = 135	9 = 1,800	14 = 9,600
5 = 150	10 = 2,400	15 = 19,200

For example, to set 75 bps, use

```
      & SPEED= 2 + 128
```

or

```
      & SPEED= 130
```

These values (1 to 15) are returned by &FN 8 and &FN 9.

Also See

FN

TERM

& TERM [(*numexp1*) [,*numexp2*]]

Invokes a terminal mode if online and a Terminal Tool module is resident. The optional *numexp1* specifies full (0) or half (1) duplex echo modes. Terminal emulation is provided by the Console module.

To exit terminal mode, hold down the command key while typing any other key. The ASCII value of the key plus 128 is returned in the optional *numexp2*. If carrier is lost while in terminal mode, &TERM exits automatically, returning 0 in *numexp2*.

Sample Program

```
10 & CALL "555-1212"
20 & WAIT FOR CARRIER
30 & TERM (0),CMD : REM Full duplex mode
40 IF NOT CMD THEN PRINT "Carrier Lost": END
50 PRINT "You pressed Command-" CHR$(CMD-
128)
```

The Terminal module includes a built-in command (command-control-2) which toggles a "debugging" mode. Control characters are displayed with a caret (e.g. ^H), and characters with their high bits set are shown in inverse.

Also See

TSET

TIMER

& TIMER (*numexp*)
& TIMER ON [*numexp*]
& TIMER STOP

These commands manage inactivity timing. If the user has been inactive for a specified period of time, ModemWorks can take action.

`&TIMER(`*`numexp`*`)` Sets the inactivity period. *numexp* is a value between 1 and 65,535 seconds (over 18 hours).

`&TIMER ON` Turns on inactivity timing. If the user has been inactive for the specified duration, ModemWorks asks, “Hello?”, and waits for the user to press a key. If no key is entered after 15 seconds, ModemWorks displays, “Bye...” and diverts program flow to the line specified by the & ON HANGUP GOTO command. The optional *numexp* argument controls ModemWorks’ prompting of the user. If *numexp* is 1, no prompting is done, and ModemWorks jumps immediately to the & ON HANGUP GOTO line.

`&TIMER STOP` Turns off inactivity timing.

Unpredictable results may occur if &TIMER ON is issued without previously using &ON HANGUP GOTO.

Also See
ON HANGUP GOTO

TRACE

& TRACE *numexp*

Sets the kind of action ModemWorks takes if immediate program mode is detected. This is a special security feature allowing you to prepare the computer for error situations that might send a caller into Applesoft immediate mode or the monitor.

The *numexp* value determines the kind of action taken:

0	Nothing happens
1	Hangup, remove ModemWorks, reboot computer
2	Hangup, remove ModemWorks, stay in BASIC

Hanging up and removing ModemWorks renders the computer inaccessible to callers. Rebooting is useful for restarting your bulletin board system if set up in a turn-key configuration.

Once your program establishes this protection, you'll want to make sure that you can turn it off whenever *you* need to return to immediate mode.

TSET

& TSET (*numexp*)

Installs a terminal emulation capability file (termcap). *numexp* is an address in memory where the termcap resides. The terminal information is copied into memory provided by the Console module, so you can reuse the memory at *numexp*.

Sample Program

```
10 PRINT CHR$(4) "BLOAD ANSI, A768, TØ"
20 & TSET (768)
30 PRINT "ANSI emulator installed."
```

Line 10 loads the ANSI termcap file into memory at address 768, a 200-byte area of free memory. Line 20 tells the Console module that an emulation template resides at that location. Terminal emulation template files have a file type of $00.

Also See

IOCTL

USR

& USR *numexp*

Performs procedures unique to your modem or port. Values for *numexp* are:

0 *reserved*
1 *reserved*
2 *reserved*
3 Modem speaker is off always
4 Modem speaker is on until a connection is made
5 Modem speaker is on always
6 Reset modem
7 Port reports a constant carrier signal
8 Port reports the actual carrier signal
9 DTR line is pulled low
10 DTR line is held high

&USR 7 forces the driver into reporting a carrier signal , even though one may not exist. This is useful so that &TERM, or similar commands which require a carrier in order to operate, can be used while offline. &USR 8 returns the driver to the standard carrier reporting mode.

Also See
FN

WAIT

& WAIT *numexp*

Suspends program execution for *numexp* seconds, useful for providing precise delays in your applications. &WAIT accepts an argument from 1 to 65,535 seconds (over 18 hours). This eliminates the need for inaccurate FOR-NEXT delay loops.

&WAIT can be cancelled by pressing a defined interrupt key.

Sample Program

```
10 HOME
20 PRINT "Pausing for 10 seconds...";
30 & WAIT 10
40 PRINT "done!"
```

Also See

FAST, INT, SLOW

WAIT FOR

`& WAIT [`*`numexp`*`] FOR `*`strexp`*` [,`*`numvar`*`]`

Waits *numexp* seconds (default is 45) for the text pattern contained in *strexp* to be received. All processes stop while incoming characters are examined and displayed. If *numexp* is negative, incoming data is not shown. Case is ignored, so “Login” and “LOGIN” appear to be identical.

If the connection is lost, a key is pressed, or the specified time has elapsed, 0 is returned in the optional *numvar*. If the pattern is found, *numvar* contains 1.

Sample Program

```
100 & WAIT FOR "secret woid",FOUND
110 IF NOT FOUND THEN STOP
120 PRINT "Access granted!"
```

WAIT FOR CALL

`& WAIT FOR CALL [,`*`numvar`*`]`

Suspends a program indefinitely until the phone rings or until a key is pressed. The optional *numvar* contains 1 if a ring is detected, or zero if a key is pressed.

See the next command for a sample program.

Also See

FN

WAIT FOR CARRIER

& WAIT [*numexp*] FOR CARRIER [,*numvar*]

Waits *numexp* seconds (default is 45) for the modem to establish a connection or until a key is pressed.

Upon return, the optional *numvar* contains a status value:

0	Connection established
1	Cancelled by keypress
2	No connection
3	Busy
4	No dial tone
5	No answer
6	Voice cancel

NOTE: Non-zero values are failed connection codes. Only zero indicates no errors and a successful connection. Not all modems can report busy signals, no dialtone, and no answer conditions.

Sample Program

```
10 & HANGUP
20 PRINT "Waiting...";
30 & WAIT FOR CALL, RING
40 IF NOT RING THEN END
50 PRINT "ring!" : & PICKUP
60 & WAIT FOR CARRIER, RESULT
70 IF RESULT > 0 THEN 20
80 & ON HANGUP GOTO 10
90 & CHK ON
100 PRINT "Online!"
```

Also See

FN, PICKUP

ASCII Chart

Low				High	Low				High	Low					High	Low				High
0	$00	^@	128	$80	32	$20	SPC	160	$A0	64	$40	@	🍎	192	$C0	96	$60	‘	224	$E0
1	$01	^A	129	$81	33	$21	!	161	$A1	65	$41	A	🍏	193	$C1	97	$61	a	225	$E1
2	$02	^B	130	$82	34	$22	"	162	$A2	66	$42	B	◣	194	$C2	98	$62	b	226	$E2
3	$03	^C	131	$83	35	$23	#	163	$A3	67	$43	C	⧖	195	$C3	99	$63	c	227	$E3
4	$04	^D	132	$84	36	$24	$	164	$A4	68	$44	D	✓	196	$C4	100	$64	d	228	$E4
5	$05	^E	133	$85	37	$25	%	165	$A5	69	$45	E	☑	197	$C5	101	$65	e	229	$E5
6	$06	^F	134	$86	38	$26	&	166	$A6	70	$46	F	⏎	198	$C6	102	$66	f	230	$E6
7	$07	^G	135	$87	39	$27	'	167	$A7	71	$47	G	≣	199	$C7	103	$67	g	231	$E7
8	$08	^H	136	$88	40	$28	(	168	$A8	72	$48	H	←	200	$C8	104	$68	h	232	$E8
9	$09	^I	137	$89	41	$29	)	169	$A9	73	$49	I	…	201	$C9	105	$69	i	233	$E9
10	$0A	^J	138	$8A	42	$2A	*	170	$AA	74	$4A	J	↓	202	$CA	106	$6A	j	234	$EA
11	$0B	^K	139	$8B	43	$2B	+	171	$AB	75	$4B	K	↑	203	$CB	107	$6B	k	235	$EB
12	$0C	^L	140	$8C	44	$2C	,	172	$AC	76	$4C	L	‾	204	$CC	108	$6C	l	236	$EC
13	$0D	^M	141	$8D	45	$2D	-	173	$AD	77	$4D	M	↵	205	$CD	109	$6D	m	237	$ED
14	$0E	^N	142	$8E	46	$2E	.	174	$AE	78	$4E	N	█	206	$CE	110	$6E	n	238	$EE
15	$0F	^O	143	$8F	47	$2F	/	175	$AF	79	$4F	O	⇤	207	$CF	111	$6F	o	239	$EF
16	$10	^P	144	$90	48	$30	0	176	$B0	80	$50	P	⇥	208	$D0	112	$70	p	240	$F0
17	$11	^Q	145	$91	49	$31	1	177	$B1	81	$51	Q	⤓	209	$D1	113	$71	q	241	$F1
18	$12	^R	146	$92	50	$32	2	178	$B2	82	$52	R	⤒	210	$D2	114	$72	r	242	$F2
19	$13	^S	147	$93	51	$33	3	179	$B3	83	$53	S	─	211	$D3	115	$73	s	243	$F3
20	$14	^T	148	$94	52	$34	4	180	$B4	84	$54	T	└	212	$D4	116	$74	t	244	$F4
21	$15	^U	149	$95	53	$35	5	181	$B5	85	$55	U	→	213	$D5	117	$75	u	245	$F5
22	$16	^V	150	$96	54	$36	6	182	$B6	86	$56	V	▒	214	$D6	118	$76	v	246	$F6
23	$17	^W	151	$97	55	$37	7	183	$B7	87	$57	W	▓	215	$D7	119	$77	w	247	$F7
24	$18	^X	152	$98	56	$38	8	184	$B8	88	$58	X	⊏	216	$D8	120	$78	x	248	$F8
25	$19	^Y	153	$99	57	$39	9	185	$B9	89	$59	Y	⊐	217	$D9	121	$79	y	249	$F9
26	$1A	^Z	154	$9A	58	$3A	:	186	$BA	90	$5A	Z	▕	218	$DA	122	$7A	z	250	$FA
27	$1B	^[	155	$9B	59	$3B	;	187	$BB	91	$5B	[	◆	219	$DB	123	$7B	{	251	$FB
28	$1C	^\	156	$9C	60	$3C	<	188	$BC	92	$5C	\	⚌	220	$DC	124	$7C	|	252	$FC
29	$1D	^]	157	$9D	61	$3D	=	189	$BD	93	$5D	]	╬	221	$DD	125	$7D	}	253	$FD
30	$1E	^^	158	$9E	62	$3E	>	190	$BE	94	$5E	^	⊡	222	$DE	126	$7E	~	254	$FE
31	$1F	^_	159	$9F	63	$3F	?	191	$BF	95	$5F	_	▏	223	$DF	127	$7F	DEL	255	$FF
Low				High	Low				High	Low					High	Low				High

ProDOS File Types

Type	Hex	Dec	Description
UNK	$00	0	Unknown
BAD	$01	1	Bad Blocks
PCD	$02	2	Apple /// Pascal Code
PTX	$03	3	Apple /// Pascal Text
TXT	$04	4	ASCII Text
PDA	$05	5	Apple /// Pascal Data
BIN	$06	6	General Binary
FNT	$07	7	Apple /// Font
FOT	$08	8	Graphics
BA3	$09	9	Apple /// BASIC Program
DA3	$0A	10	Apple /// BASIC Data
WPF	$0B	11	Word Processor
SOS	$0C	12	Apple /// SOS System
DIR	$0F	15	Folder
RPD	$10	16	Apple /// RPS Data
RPI	$11	17	Apple /// RPS Index
AFD	$12	18	Apple /// AppleFile Discard
AFM	$13	19	Apple /// AppleFile Model
AFR	$14	20	Apple /// AppleFile Report Format
SCL	$15	21	Apple /// Screen Library
PFS	$16	22	PFS Document
ADB	$19	25	AppleWorks Data Base
AWP	$1A	26	AppleWorks Word Processor
ASP	$1B	27	AppleWorks Spread Sheet
TDM	$20	32	Desktop Manager Document
8SC	$29	42	Apple II Source Code
8OB	$2A	43	Apple II Object Code
8IC	$2B	44	Apple II Interpreted Code
8LD	$2C	45	Apple II Language Data
P8C	$2D	46	ProDOS 8 Code Module
FTD	$42	66	File Type Names
GWP	$50	80	Apple IIGS Word Processor
GSS	$51	81	Apple IIGS Spread Sheet
GDB	$52	82	Apple IIGS Data Base
DRW	$53	83	Drawing
GDP	$54	84	Desktop Publishing
HMD	$55	85	Hypermedia
EDU	$56	86	Educational Data
STN	$57	87	Stationery
HLP	$58	88	Help
COM	$59	89	Communications
CFG	$5A	90	Configuration
ANM	$5B	91	Animation
MUM	$5C	92	Multimedia
ENT	$5D	93	Entertainment
DVU	$5E	94	Development Utility

Continued . . .

ProDOS File Types (Continued)

Type	Hex	Dec	Description
BIO	$6B	107	PC Transporter BIOS
TDR	$6D	109	PC Transporter Driver
PRE	$6E	110	PC Transporter Pre-Boot
HDV	$6F	111	PC Transporter Volume
WP	$A0	160	WordPerfect Document
GSB	$AB	171	Apple IIGS BASIC Program
TDF	$AC	172	Apple IIGS BASIC TDF
BDF	$AD	173	Apple IIGS BASIC Data
SRC	$B0	176	Apple IIGS Source
OBJ	$B1	177	Apple IIGS Object
LIB	$B2	178	Apple IIGS Library
S16	$B3	179	GS/OS Application
RTL	$B4	180	GS/OS Run-time Library
EXE	$B5	181	GS/OS Shell Application
PIF	$B6	182	Permanent Initialization
TIF	$B7	183	Temporary Initialization
NDA	$B8	184	New Desk Accessory
CDA	$B9	185	Classic Desk Accessory
TOL	$BA	186	Tool
DRV	$BB	187	Device Driver
LDF	$BC	188	Load File
FST	$BD	189	GS/OS File System Translater
DOC	$BF	191	GS/OS Document
PNT	$C0	192	Packed Super Hi-Res Picture
PIC	$C1	193	Super Hi-Res Picture
ANI	$C2	194	Animation
PAL	$C3	195	Palette
OOG	$C5	197	Object Oriented Graphics
SCR	$C6	198	Script
CDV	$C7	199	Control Panel
FON	$C8	200	Font
FND	$C9	201	Finder Data
ICN	$CA	202	Icons
MUS	$D5	213	Music Sequence
INS	$D6	214	Instrument
MDI	$D7	215	MIDI
SND	$D8	216	Sampled Sound
DBM	$DB	219	Relational Data Base File
LBR	$E0	224	Archival Library
ATK	$E2	226	AppleTalk Data
R16	$EE	238	EDASM 816 Relocatable File
PAS	$EF	239	Pascal Area
CMD	$F0	240	BASIC Command
LNK	$F8	248	EDASM Linker
OS	$F9	249	GS/OS System File
INT	$FA	250	Integer BASIC Program
IVR	$FB	251	Integer BASIC Variables
BAS	$FC	252	Applesoft BASIC Program
VAR	$FD	253	Applesoft BASIC Variables
REL	$FE	254	Relocatable Code
SYS	$FF	255	ProDOS 8 System Application

Error Codes

0 **NEXT Without FOR**: a NEXT was encountered which had no matching FOR.

2 **Range Error**: an invalid argument value was specified.

3 **No Device Connected**: the given slot has no disk drive installed.

4 **Write Protected Disk**: unable save data unless write-enabled.

5 **End of Data**: an attempt was made to read data past the end of a file.

6 **Path Not Found**: the path to a filename was not found.

7 **File Not Found**: the specified file was not found.

8 **I/O Error**: the drive went offline or the disk has a media defect.

9 **Disk Full**: no room exists on the disk storing more data.

10 **File Locked**: the file is protected against modification or removal.

11 **Invalid Option**: an option not allowed for a certain command was used.

12 **No Buffers Available**: not enough memory for further disk functions.

13 **File Type Mismatch**: an invalid attempt was made to access a special file.

14 **Program Too Large**: you've written a FAT and SLOPPY program.

15 **Not Direct Command**: command was issued from immediate mode.

16 **Syntax Error**: a filename is illegal or a program statement misspelled.

17 **Directory Full**: the root volume contains too many filenames.

18 **File Not Open**: an attempt was made to read or write from an closed file.

19 **Duplicate File Name**: a RENAME or CREATE used on an existing filename.

20 **File Busy**: an attempt to re-OPEN or modify an OPEN file's name was made.

21 **File Still Open**: upon entering immediate mode, a file was found OPEN.

22 **RETURN Without GOSUB**: a RETURN with no matching GOSUB.

42 **Out of Data**: an attempt was made to READ past the last DATA item.

53 **Illegal Quantity**: an out-of-range value was used with a certain command.

69 **Overflow**: you used an awfully BIG or amazingly SMALL number.

77 **Out of Memory**: program code and variables have used up all free memory.

90 **Undef'd Statement**: a line number which does not exist was referenced.

107 **Bad Subscript**: an array subscript is larger than the given DIMension.

120 **Redim'd Array**: an attempt was made to reDIMension an existing array.

133 **Division by Zero**: division by zero is undefined (remember your algebra?)

163 **Type Mismatch**: a numeric or string value was used incorrectly.

176 **String Too Long**: the given string was larger than was allowed.

191 **Formula Too Complex**: go easy on the machine, Einstein.

224 **Undef'd Function**: reference to an undefined FuNction was made.

254 **Reenter**: user input was not of the type or format required.

255 **Control-C Interrupt**: *control*-C was pressed.

Licensing

As stated on the inside cover of this manual, this is a copyrighted software product. It may not be distributed in any way without permission of the Morgan Davis Group. To obtain authorization to include Morgan Davis Group software with your commercial products, write or call and request a Universal Software Licensing Agreement. Be sure to include the title of the Morgan Davis Group software you wish to license:

http://www.morgandavis.net

BASIC Syntax

Throughout this manual abbreviations are used to clarify special syntaxes or conditions for command usage. This appendix quickly explains what they mean and how they're used.

strexp

A string is defined as a group of letters, numbers, symbols, or control codes. A string expression, or *strexp* as used in this manual, is any combination of strings and their various forms in BASIC. Examples of string expressions:

```
X$
"Hello, World."
"this" + "that"
CHR$(4) + "OPEN" + FILE$
CHR$(ASC(MID$(Q$, I, 1)) - 2) + "yuck!"
```

strvar

With some ampersand commands that return string information, a string variable, *strvar*, is required. When a *strvar* is called for, a string expression is *not* allowed. Examples of string variables:

```
X$
NAME$(7)
```

boolexp

A Boolean expression, *boolexp*, is any logical operation that results in a TRUE or FALSE numeric value. This includes numeric or string operations used *conditionally*. In BASIC, a TRUE value is anything other than zero (usually one), while FALSE is *always* zero. Some examples:

```
"A" = B$
"A" < "B" OR "B" < "C"
((I - J) OR Q) AND C < (D + 33 * (NOT X))
```

numexp

A numeric expression, *numexp*, is any combination of numbers, numeric variables, or arithmetic functions that result in a numeric value. Examples:

```
X
2 + 2
ASC(MID$(B$, 5, 1)) + 64 * (C / 2)
PI - INT(LOG(X) / SIN(Y) * Y * 20))
```

numvar

A numeric variable, *numvar*, is used when a command returns a numeric value. Examples:

```
X
Q2%
J(3 + I)
```

Optional Arguments

Some commands accept optional parameters, shown within [ ] brackets in this manual. Do not include the brackets when you enter the commands into BASIC.

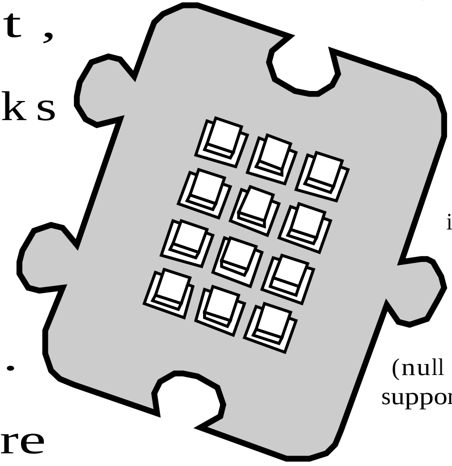
Index

NOTES

COMMUNICATIONS FOR BASIC PROGRAMMERS

Integrated modules let your BASIC programs access the full power of your modem. A logical language extension to Applesoft, ModemWorks makes adding communications features easy. Whether you are creating the next great bulletin board system, or a custom terminal program, ModemWorks gives you all the tools, power, and flexibility needed to create world-class applications.

- ► Efficient modular design
- ► Lightning-fast performance
- ► X, Y and ZMODEM protocols
- ► Communications event handling
- ► Terminal emulation (ANSI, VT-220, VT-100, and more)



Compatible with the Apple IIe and Apple IIGS, built in serial ports, Super Serial Card and clones, including internal Hayes-compatible modem cards. Supports over 50 popular modems. Allows the user to easily add support for additional modems. Computer-to-computer (null modem) connections are also supported. Runs under Apple ProDOS.

- ► High-speed modem support
- ► Crash-proof security feature
- ► Transparent input and output
- ► Supports popular printers
- ► Flexible and infinitely expandable
- ► Sensible command names
- ► Easy to learn and use

Morgan Davis Group
www.morgandavis.net