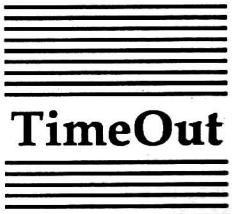
TIMEOUT ULTRAMACROS™

THE MOST POWERFUL MACRO PROGRAM EVER WRITTEN FOR APPLEWORKS™!



TimeOut UltraMacros™

Power Macros for AppleWorks™
by Randy Brandt

Published by
BEAGLE BROS, INC.
6215 Ferris Square, Suite 100
San Diego, CA 92121
(619) 452-5500

Contents

Chapter 1 **Welcome to TimeOut UltraMacros 1**
So What's a Macro? 2
Special Features of UltraMacros 3
New AppleWorks Commands 3
Bug Extermination 3
Mousing Around 3

Chapter 2 **Installation 5**
TimeOut Compatibility 6
Installing TimeOut 7
The TimeOut Menu 8
Location of TimeOut Applications 8
Location of AppleWorks 9
Copying Applications to the TimeOut Application Disk 9
Reinstalling TimeOut 9
Installing UltraMacros 10
Starting up AppleWorks with TimeOut 11
Accessing TimeOut Applications 12
Memory usage 13
Control-Reset Patch 13

Chapter 3 **UltraMacros Tutorial 15**
Using the New UltraMacros Commands 16
Recording Your Own Macros 17
Creating Custom Macros 20
Built-In Macros 20
Creating a Macro File 21
Creating your very own UltraMacro 21

Chapter 4 UltraMacros Reference 25
The Anatomy of a Macro 26
Tokens 26
Local and Global Macros 28
Calling other macros 30
Reserved Macros 31
<sa-del> SOLID-APPLE-DELETE 31
<ahead> SOLID-APPLE-. 31
<back> SOLID-APPLE-, 31
<date> SOLID-APPLE-' 31
<date2> SOLID-APPLE-" 31
<time> SOLID-APPLE-= 32
<time24> SOLID-APPLE-+ 32
<find> SOLID-APPLE-RETURN 32
<findpo> SOLID-APPLE-^ 33
<print> BOTH-APPLE-CONTROL-P 34
New Open-Apple Commands 34
no token OPEN-APPLE-X 34
"sa-del" OPEN-APPLE-DELETE 34
"getstr" OPEN-APPLE-0 34
<oa-ctrl-@> OPEN-APPLE-CONTROL-@ 35
<uc> OPEN-APPLE-: 35
<lc> OPEN-APPLE-; 35
<insert> OPEN-APPLE-! 35
<zoom> OPEN-APPLE-@ 35
<read> OPEN-APPLE-^ 36
<disk> OPEN-APPLE-& 36
<path> OPEN-APPLE-* 36
<cell> OPEN-APPLE-- 36
<store> OPEN-APPLE-< 37
<recall> OPEN-APPLE-> 37
<inc> OPEN-APPLE-CONTRL-W 37
<dec> OPEN-APPLE-CONTROL-A 37
<bell> OPEN-APPLE-CONTROL-G 38
<nosleep> OPEN-APPLE-CONTROL-N 38
<clear> OPEN-APPLE-CONTROL-X 38
Special UltraMacros Tokens 38
<input> 38
<id#> 39
<ifkey> 39
<key> 39
<begin> 40

<rpt> 40
<stop> 40
Special UltraMacros Tokens with parameters 40
Defining Numeric Variables 41
Defining String Variables 42
Parameters 43
<asc STRING VAR> 45
<chr$ NUM> 45
<getstr NUM> 45
<goto MACRO> 45
<hilight NUM EXP,NUM EXP,NUM EXP,NUM EXP> 46
<left STRING VAR,NUM> 46
<len STRING VAR> 46
<msg STRING> 46
<onerr OPTION> 47
<posn VAR,VAR> 48
<pr# NUM EXP> 48
<print> 49
<rem STRING> 50
<right STRING VAR,NUM> 51
<screen NUM EXP,NUM EXP,NUM EXP> 51
<str$ VAR NAME> 52
<val STRING VAR> 52
<wait NUM EXP> 52
<wake MACRO at NUM EXP:NUM EXP> 53
If-Then-Else Logic 54
<if> 54
<ifnot> 55
<then> 56
<else> 56
<elseoff> 57
For experienced 6502 programmers only! 57
<call> 58
<poke> 58
<peek> 59
Macro Compiler 59
Compile a new set of macros 60
Macro Compiler Errors 60
Display current macro set 63
Macro Options 64
1. Launch a new Task 64
2. Create a Task File 65

3. Save macro table as default set 65
4. Deactivate macros 66
5. Reactivate single-stepping 66
6. Version 66
7. Other Activities 67
Other Activities 67
1. Set cursor blink 67
2. Set mouse button response 68
3. Set mouse response 68
4. Deactivate the mouse 68
5. Reactivate Key-Lock 69
6. Reactivate screen preserver 70
7. Set screen preserver delay 70
Data Converter 71
Allowing Control-@ 72
Mouse Control 72
Linking Files 73
Startup Menus 73
Task Files 74
The Special Case of Macro 0 (zero) 76
A Macro Explained 78
TimeOut MacroTools 79
Changes from Super MacroWorks 79
Macro Token List 81

Appendix **TimeOut Utilities 83**
Using the Utilities 84
Configure 84
Load to memory 85
Dump from memory 85
Change memory status 86
Change name 86
Sort Menu 86

Key Chart 87

Help! **Customer Support Information 89**

Index 91

Chapter 1

Welcome to TimeOut UltraMacros

TimeOut UltraMacros™ is a powerful addition to the TimeOut family of AppleWorks™ enhancements, adding macro capability and numerous new commands that can be used with any AppleWorks or TimeOut application. Because UltraMacros is compatible with all other TimeOut packages, it is a solid foundation to build on.

So What's a Macro?

A macro is a single keystroke that does the work of many keystrokes. An AppleWorks macro is a SOLID-APPLE key command; you simply hold down the SOLID-APPLE key while pressing another key and a predefined sequence of keystrokes is performed. For example, you can set up a macro like SOLID-APPLE-N that types your name and address, or use a macro like SOLID-APPLE-I to indent a paragraph three spaces (one keystroke instead of the usual seven). Macros save you a lot of typing and a lot of time. Also, the fewer keystrokes, the fewer chances for making errors.

> *NOTE: The SOLID-APPLE key on the Apple IIe and IIc has been replaced by the OPTION key on the IIgs. If you have a IIgs, think OPTION whenever this manual mentions SOLID-APPLE.*
>
> *The IIgs numeric keypad does not add extra keys for macro users. ENTER is the same as RETURN, CLEAR is CONTROL-X, and the other keys are simply duplicates of their main keyboard equivalents.*

Remember that macros are SOLID-APPLE commands. The AppleWorks OPEN-APPLE commands perform the same functions as before.

UltraMacros is a very flexible package. A wide variety of built-in macros are provided on the UltraMacros disk. They can be used "as-is" or changed to suit your own needs. You can also design

completely new macros once you get more familiar with UltraMacros. Novices and experts alike will appreciate the many new OPEN-APPLE commands which are also included.

Special Features of UltraMacros

In addition to macros, UltraMacros provides other features that make your life at the keyboard easier and more productive. These include new AppleWorks commands, an AppleWorks bug fix, and mouse control.

New AppleWorks Commands

UltraMacros adds new OPEN-APPLE and SOLID-APPLE commands that save you time and effort. For example, you can press SOLID-APPLE-= at any time to enter the current time (if you have a clock). Read about the new commands starting on page 16. A complete list of new commands starts on page 31.

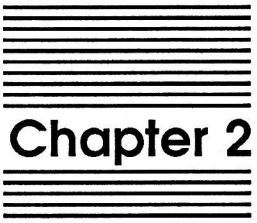
Bug Extermination

AppleWorks has a bug that doesn't allow you to enter CONTROL-@ for printer or interface card definitions. See page 72 for details on solving this problem.

Mousing Around

UltraMacros allows you to use a mouse to scroll rapidly though AppleWorks and to make menu selections. See page 72 for details.

> *Note: This manual assumes that you are familiar with AppleWorks and the AppleWorks method of doing things (e.g. selecting menu options, etc.) If you're not, then refer to the appropriate sections of your AppleWorks manuals.*



Chapter 2

Installation

This chapter tells you how to install TimeOut and add the UltraMacros commands to your *AppleWorks Startup* disk.

Before you can use macros and the new commands, you must update a copy of your *AppleWorks Startup* disk. (Note: The floppy disk version of AppleWorks uses two disks: a STARTUP disk and a PROGRAM disk. If you are running AppleWorks from a hard disk, 3.5" disk or RAM disk, there is only one disk.)

You actually need to make two modifications to AppleWorks; TimeOut must be added so that you can use the TimeOut applications on the UltraMacros disk, and then a file called *ULTRA.SYSTEM* must be added to your *AppleWorks STARTUP* disk. This is the actual macro program.

TimeOut UltraMacros is provided on both 5.25-inch and 3.5-inch disks. Before using your *TimeOut UltraMacros* disk, please take a moment to make a backup copy of the disk you will be using. Since the disks are not copy-protected, you may use any standard disk duplication program such as *Filer, System Utilities, Copy II Plus,* or Beagle Bros' *TimeOut FileMaster.* Be sure to write-protect your original disk so you don't accidentally overwrite it. Please do this now.

> *NOTE: Since the 5.25-inch UltraMacros disk is two-sided, be sure to duplicate both sides.*

Beagle Bros software isn't copy-protected. That makes it easier for you to use and easier to make backup copies. Please support us in our commitment to supply friendly, easy-to-use software by not giving away copies to your friends.

TimeOut Compatibility

TimeOut is compatible with AppleWorks versions 2.0 and later. If you have an earlier version, contact your dealer about getting an update from Apple. You must also have the USA version. TimeOut does not work with foreign language versions of AppleWorks.

TimeOut is compatible with most enhancements to AppleWorks including *Applied Engineering's desktop expander and Checkmate's desktop expander.* UltraMacros is not compatible with *Pinpoint, SuperMacroWorks or AutoWorks.*

If you are installing TimeOut with other AppleWorks enhancements, we recommend that you install TimeOut last.

Installing TimeOut

Once your backup copy has been made, boot your *TimeOut UltraMacros* disk by placing it in your boot drive and turning your computer off and back on or by pressing Control-Open-Apple-Reset. Then press `T` for `Install TimeOut`.

Soon the title screen will appear. The menu contains three options:

```
1. Update AppleWorks

2. Read NOTES

3. Quit
```

Select `Read NOTES`. This will inform you of any changes to TimeOut UltraMacros that have been made since this instruction manual was printed.

After you have read the *NOTES*, you will return back to the TimeOut startup screen. This time select `Update AppleWorks`.

The TimeOut Menu

You must now specify whether or not you would like TimeOut to sort the list of TimeOut applications in the TimeOut menu. The names will be sorted alphabetically if you specify `Yes`. Otherwise, they will appear in the same order as they do in the disk catalog. By specifying `No`, you decide the order of the names in the TimeOut menu by placing them on your TimeOut applications disk in the order that you want.

Multiple TimeOut Application Disks

If you have several TimeOut applications and are using 5.25-inch disks, you may need more than one TimeOut application disk. Specify `Yes` when asked `Do you need more than one TimeOut Application disk?` All application disks must have the same name (for example: /TIMEOUT).

Location of TimeOut Applications

The next step is to indicate where TimeOut should look for the TimeOut applications. All of the TimeOut applications must be placed on the same disk (or if they won't all fit on one disk they may be on several disks as long as the disks have the same name). This can be a hard disk, a RAM disk, a floppy disk, or any ProDOS disk device. (See *Copying Applications to the TimeOut Application disk* on page 9.) The choices you have are:

```
1. AppleWorks STARTUP disk

2. Slot and Drive

3. ProDOS directory
```

If you have a hard disk or a 3.5-inch disk, you may want to place your TimeOut applications in the same directory or subdirectory with your AppleWorks STARTUP program (APLWORKS.SYSTEM).

If you have more than one disk drive, you may want to dedicate one drive to your TimeOut applications. You may specify either `Slot and Drive` or `ProDOS directory` to indicate where the TimeOut applications disk will be. For more information on ProDOS directories, see the section in your AppleWorks manual called *ProDOS, prefix for filenames.*

Location of AppleWorks

The next step is to indicate where your AppleWorks STARTUP program is so TimeOut can be installed. You may specify either `Slot and Drive` or `ProDOS directory`. After indicating the location of AppleWorks, press a key and your AppleWorks STARTUP program will be updated with TimeOut.

Copying Applications to the TimeOut Application Disk

If you are using other TimeOut applications, you will need to copy the applications on the UltraMacros disk to your TimeOut application disk. Otherwise, you have the option of using the UltraMacros disk as your TimeOut application disk or you can use any other disk (i.e. the AppleWorks Startup disk, a RAM disk, or a hard disk).

If you're not going to use the UltraMacros disk as your application disk, select 2 from the final installation menu to copy the UltraMacros applications to your application disk. You will need to specify either the slot and drive or the pathname of the application disk.

Reinstalling TimeOut

After you have already installed TimeOut, if you need to change the applications disk location or the order of the menu, you can reinstall TimeOut by following the same steps for initial installation. This will only work if you have not installed any

other AppleWorks enhancement programs since you installed TimeOut. If you have then you may need to completely reconfigure AppleWorks.

Installing UltraMacros

If you're already using Super MacroWorks, you may wish to read about the changes on UltraMacros (page 79) before continuing with this installation.

1. Make sure you have a copy of your *AppleWorks STARTUP* disk which has already been modified with TimeOut. If you're using a desktop expander, install it before installing UltraMacros. Make sure that your AppleWorks disk is functional with all other modifications installed before attempting to add UltraMacros.

2. Boot the *UltraMacros* disk and press `U` for `Install UltraMacros`. The macro program (*ULTRA.SYSTEM*) will be loaded into memory.

3. Remove the UltraMacros disk from the main disk drive, insert the *AppleWorks STARTUP* disk and press RETURN.

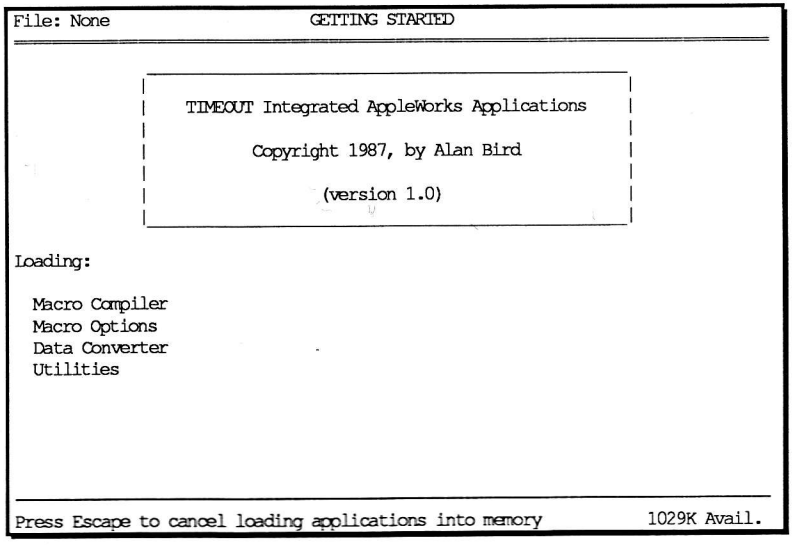
4. The UltraMacros installation program will attempt to modify your *AppleWorks STARTUP* disk. If it can't find it, it will prompt you to enter the pathname where AppleWorks can be found. You probably won't have AppleWorks in a subdirectory unless you understand subdirectories and pathnames already. In any case, complete pathname information can be found in manuals such as the ProDOS User's Manual. A pathname ALWAYS begins with a "/" followed by the disk name. Any subdirectories following the disk name are preceded by "/"'s as well.

5. Your *AppleWorks STARTUP* disk will be updated with UltraMacros' macros and new commands. A message will appear on the screen when the updating is complete.

6. Press RETURN; the built-in macros and commands supplied with the UltraMacros disk are now ready for use.

Starting up AppleWorks with TimeOut

When you start up AppleWorks with TimeOut installed, you should see the TimeOut title screen before you reach the AppleWorks main menu. TimeOut will scan your TimeOut applications disk looking for TimeOut applications.



Note: If you receive a message indicating that TimeOut is getting errors trying to load the TimeOut applications, it means that TimeOut is unable to find the applications. At this point you must either insert your applications disk (if you have not already done so) and try again or specify a different location. If you have inserted your applications disk and are still getting errors, you either do not have any applications on the disk or your disk has been damaged. You will need to create a new applications disk.

If you do not see a TimeOut title screen, you have not installed TimeOut correctly. Go back to page 6 and start over.

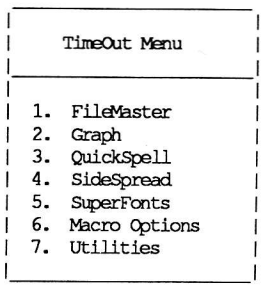
As TimeOut identifies each TimeOut application, they are listed on the screen. An asterisk ("*") before the application name indicates that it is memory-based. You may press Escape at any time to cancel loading memory-based applications (see pages 13, 85-86 for an explanation of memory-based applications).

If you have specified that you are using multiple TimeOut application disks, insert each disk and answer `Yes` when asked `Read another TimeOut application disk?` Answer `No` when the last application disk has been read.

> *Note: If you use a program selector such as the Apple Desktop you must select ULTRA.SYSTEM instead of APLWORKS.SYSTEM. The APLWORKS.SYSTEM file has been renamed to APLWORKS.SYS so that booting ProDOS will automatically start ULTRA.SYSTEM. If you select APLWORKS.SYS, you will run normal AppleWorks without any macros.*

Accessing TimeOut Applications

While you are using AppleWorks, you may call up the TimeOut menu at any time by holding down the Open-Apple key and simultaneously pressing the Escape key. A menu similar to the following will appear (you may have fewer or more applications than this):



Use the up and down arrow keys or type a number to select an application, then press Return (press Escape if you don't want to make a selection). If your applications are not memory-based, be sure your TimeOut applications disk is in the drive when you press Return. Otherwise, you will be prompted to insert your TimeOut application disk. If you are using multiple application disks, be sure to insert the correct one.

Memory usage

You will notice with TimeOut installed that you have slightly less desktop memory for your AppleWorks documents. TimeOut itself takes up some of the memory.

Memory-resident TimeOut applications also take up desktop memory. If you are short on desktop memory, reconfigure your applications so they are disk-based.

However, for maximum speed, make your TimeOut applications memory-based or run them from a RAM disk.

> *Note: UltraMacros is always in memory. However, it resides in a special place that doesn't take up any desktop memory.*

Control-Reset Patch

When you install TimeOut on your AppleWorks Startup disk, TimeOut makes a patch to AppleWorks so that Control-Reset will take you to the Main Menu instead of the machine-language monitor.

Chapter 3

UltraMacros Tutorial

This section introduces you to some of the new commands included with UltraMacros. It assumes that you've already booted AppleWorks and are in a Word Processor file (UltraMacros works everywhere in AppleWorks, but the Word Processor is a good place to test things).

Using the New UltraMacros Commands

UltraMacros adds several new SOLID-APPLE and OPEN-APPLE commands to AppleWorks. Try these for starters:

1. Press SOLID-APPLE-'. The date set when AppleWorks was started up will be displayed in this format: August 14, 1987. If an error beep happens instead, UltraMacros is not installed. Go back to the previous section and read about adding UltraMacros to AppleWorks.

2. Now place the cursor on the first letter of the date that you entered in step 1 and press OPEN-APPLE-;. The letter will be changed to lower case. For example, "August 14, 1987" would now be "august 14, 1987".

3. Press OPEN-APPLE-:. The next letter will be switched to upper case. Get the picture? You can change the case of any letter with the colon/semi-colon key. If you hold down OPEN-APPLE and press the key normally, letters become lower case; if you also hold down the shift key, the letters become upper case. The cursor is always changed to overstrike mode so that extra letters aren't inserted.

4. Place the cursor in the middle of a word and press OPEN-APPLE-DELETE or SOLID-APPLE-DELETE. The character under the cursor will be *gobbled*, but the cursor will remain at the same spot. This command actually executes two normal AppleWorks keystrokes- a RIGHT-ARROW followed by DELETE.

These new OPEN-APPLE commands can be used anywhere that AppleWorks prompts you for input. For example, the OPEN-APPLE-DELETE command can be used to change a file name after you press OPEN-APPLE-N. Or you can press SOLID-APPLE-' to enter the date when the Data Base asks you to "Type report date" when you're printing a file.

A complete list of new commands is given starting on page 31. You can experiment with some of them now, or read on to learn all about the handy OPEN-APPLE-X command.

Recording Your Own Macros

This section tells you how to record your keystrokes so that they can be played back later with one keypress. It assumes that you've already booted AppleWorks and are currently in a Word Processor file.

1. Press OPEN-APPLE-X. The title at the top of the screen changes to RECORD A MACRO. If it doesn't, go back to page 6 and read about adding UltraMacros to your *AppleWorks STARTUP* disk.

2. The prompt at the bottom tells you to `Select macro key:`. Press the `T` key to record a "Test" macro. SOLID-APPLE-T will be used to activate this macro later. The bottom right side of the screen will show `Recording T` and the cursor will stop blinking.

3. You are now in record mode and whatever you type (mouse moves are included) will be memorized. Type your name and press RETURN.

4. Now press CONTROL-@ to end the macro definition. (On IIgs's and some other II's you don't have to press the shift key; on some you do.) The cursor will start blinking and the bottom right side of the screen will show `Done macro T`.

5. You've just recorded your first macro! To use it, hold down the SOLID-APPLE key and press `T`. Your name is typed much faster than it was entered, unless you're an incredible typist. This macro can now be used anywhere that AppleWorks expects keyboard input- in a file, naming a file, naming a Data Base category, etc.

6. Now press OPEN-APPLE-X and press `T` again. This message appears on the bottom line: `Replace global macro T?`. Because you already have a macro "T" in your macro set, UltraMacros lets you decide if you want to destroy the original macro. This is a safeguard to avoid accidentally erasing a macro.

7. Press `Y` to replace your "T" macro. You can now record the new macro. This time, press OPEN-APPLE-1 first. The cursor jumps to the beginning of the file. Any OPEN-APPLE command can be recorded as part of a macro.

8. Now press RETURN followed by the UP-ARROW and enter your name on the top line of the file. Press CONTROL-@ to end the macro.

9. Press OPEN-APPLE-9 to jump to the end of the file, and then press SOLID-APPLE-T to try out the modified macro "T". It will jump to the top of the file, insert a line, and enter your name.

   You can press OPEN-APPLE-X to start recording macros from virtually anywhere in AppleWorks or in a TimeOut application. You may NOT press OPEN-APPLE-X when the TimeOut OPEN-APPLE-ESCAPE menu is visible on the screen, or when you're in the middle of an OPEN-APPLE-0 *getstring* command (page 34). The command will be ignored. However, you may start a macro within AppleWorks and then press OPEN-APPLE-ESCAPE or OPEN-APPLE-0 as part of the macro.

   Most keys can be recorded as macros; see page 31 for a list of reserved macros. Keep in mind that there is no difference between upper and lower case macro names, and that all "named" keys such as RETURN and TAB have CONTROL-key equivalents (eg. TAB is actually CONTROL-I).

10. Press OPEN-APPLE-X to record another macro.

11. Now press RETURN for the macro key. A `Reserved ^M` message appears on the bottom right of the screen. This is because CONTROL-M is the same as RETURN, and it's a reserved macro name.

    To record a BOTH-APPLE macro, hold down the OPEN-APPLE key when entering the macro name:

12. Press OPEN-APPLE-X to enter record mode.
13. Press OPEN-APPLE-T. The `Recording T` message will appear.
14. Enter a friend's name and press CONTROL-@ to end the macro.
15. Press SOLID-APPLE-T. Your name is entered as defined above.
16. Press BOTH-APPLE-T. Your friend's name appears.

All keys can have both SOLID-APPLE and BOTH-APPLE definitions. Macros that are reserved SOLID-APPLE commands, such SOLID-APPLE-RETURN, can be defined as BOTH-APPLE macros. The Macros Ultra sample includes BOTH-APPLE definitions for the named keys such as TAB, ESCAPE, etc. Of course, they can easily be changed. It's a good idea to use BOTH-APPLE macros for potentially dangerous macros (like quit AppleWorks without saving files) that you don't want to execute accidentally.

Holding down the SOLID-APPLE and OPEN-APPLE keys simultaneously to execute a BOTH-APPLE command is easy on a IIgs, but a little more complex on an older Apple II. The preferred method is to use the thumb and a finger on one hand to hold down BOTH-APPLE's and to use the other hand to press the desired key. Try BOTH-APPLE-CONTROL-^ if you're incredibly talented.

You can record a macro for anywhere from 2 to 4,000 or so keystrokes, depending on how many macro keystrokes are already in memory. If the macro table is full, the `Done macro` message will appear as soon as you press the new macro key, unless you're replacing an existing macro. Then you'll be able to enter as many keystrokes as the original macro contained.

The only exception is macro 0 (zero). You can always enter up to 80 keystrokes, but it also automatically stops recording at 80 keystrokes. Read about this special macro on page 76.

Any macros recorded using OPEN-APPLE-X are lost when you exit AppleWorks. There are two ways to make the macros permanent. One way is to use the Macro Options (page 64) third option `Save macro table as default set`. This saves all active macros "as-is". The next section explains how to edit existing macros and then make them permanent.

Creating Custom Macros

This section tells you how to create custom macros by editing a macro file, compiling the changed macros, and then saving them on disk.

Built-In Macros

The "built-in" macros are those macros which are part of ULTRA.SYSTEM and are available whenever you start AppleWorks. These macros (except for a few reserved macros) can be changed at any time to anything you wish.

Newcomers to macros should study and use these built-in macros before attempting to create their own non-recorded macros.

1. Boot AppleWorks and insert the UltraMacros disk.

2. Add the Word Processor file Macros Ultra to the Desktop from the UltraMacros disk and print it for a handy reference. These are the sample macros included with ULTRA.SYSTEM. All of them are available for use if you've added UltraMacros to your AppleWorks disk.

3. Examine the printout while reading this manual's descriptions of how macros are made. You can modify this file to create your own custom macros. Modifying existing macros is a good way to learn about writing your own macros.

Creating a Macro File

A macro file is any AppleWorks Word Processor file which contains macro definitions. You can create a custom macro file by adding an existing macro file to the Desktop and changing the definitions, or by using the Macro Compiler's `Display current macro set` option to list the current macros into a file. There's nothing magical about the macro definitions in the Word Processor. They must be "compiled" into true macro codes (page 59) to be used by UltraMacros.

Creating your very own UltraMacro

Here's a step-by-step look at creating your first custom macro definition and making it a permanent part of AppleWorks.

1. Start up your UltraMacros version of AppleWorks.
2. Insert the UltraMacros disk and add the file Macros Ultra to the desktop. Use OPEN-APPLE-N to change its name to "Macros Mine". Macro file names don't have to start with "Macros"; it just makes it easier to find them that way.
3. Go back to the AppleWorks Main Menu and then make a new Word Processor file called TEST.
4. Press SOLID-APPLE-B to see a sample "begin a memo" macro.
5. Press SOLID-APPLE-N to see the author's daughter's name.
6. Now press OPEN-APPLE-Q and return to the Macros Ultra file.
7. Use the OPEN-APPLE-F command to find "Heather". You should see the following macro definition:

```
N:<awp>Heather Brandt! name of a little "Lassie"
lover
```

8. Change `Heather Brandt` to your name. Ignore the <awp> for now. Just make sure that your name is immediately after <awp> and is followed by a "!". You could cheat and use OPEN-APPLE-R to replace it, but that won't teach you anything.

9. Now use the OPEN-APPLE-F command to find "B:<awp". You should see the following macro definition:

```
B:<awp><rtn><rtn>
Date:  <date><rtn><rtn><rtn>
From:  <sa-n><rtn>
       JEM SOFTWARE<rtn>
       P.O. Box 20920<rtn>
       El Cajon, CA  92021<rtn><rtn><rtn><rtn>
To:  ! begin a memo
```

The first line contains special bracketed codes called tokens. Each <rtn> represents a carriage return.

The second line also contains <date>. This token will always print the date which was set when AppleWorks was first started.

The third line contains <sa-n>. This is the same as pressing SOLID-APPLE-N; that's why the macro printed the name given in macro-N. Switch back to the TEST file and look at what the macro did until you can see the relationship between the macro definition and the result.

See page 81 for a complete list of tokens that you can use within your macros.

10. Replace the sample address between the `From:` and `To:` lines with your own address. Start each line in the same column as the "J" in JEM and end with a <rtn>.
11. Now press OPEN-APPLE-ESCAPE and select `Macro Compiler`. If it isn't on the TimeOut menu, go back to the section on installing TimeOut and make sure you copy all of the applications from the UltraMacros disk to the TimeOut application disk.
12. Press RETURN to select `Compile a new set of macros.`
13. Press RETURN to select `No` for the `Pause each line?` prompt.
14. Press RETURN to select `Beginning` for the `Compile from?` prompt.
15. The compiler will then scan the entire file, converting the text and tokens into UltraMacros' codes.
16. Press OPEN-APPLE-Q and select the TEST file.

17. Press SOLID-APPLE-B to see the memo macro with your name included within it. Now press SOLID-APPLE-N to see just your name.
18. If the macros don't work the way you think they should, go back and examine the definition again, comparing it to the samples in steps 7 and 9 of this section.
19. Now press OPEN-APPLE-ESCAPE and select `Macro Options`.
20. If your AppleWorks Startup disk is not in a drive, remove your Program disk and insert the Startup disk.
21. Select `Save current macros as default set`.
22. Press RETURN to select `No` for `Activate auto-startup macro?`".
23. The ULTRA.SYSTEM file will be updated with the new macro definitions. Reinsert your AppleWorks program disk if necessary and press RETURN to return to AppleWorks.
24. Save your "Macros Mine" file and exit AppleWorks.
25. Now restart AppleWorks and make a new Word Processor file.
26. Press SOLID-APPLE-B. Voila! Your new definition is in effect along with all of the other Macros Ultra sample macros.
27. Press SOLID-APPLE-B again, but this time tap the ESCAPE key immediately afterwards. The macro is halted before it finishes. You can press ESCAPE to stop any run-away macros.

Chapter 4

UltraMacros Reference

This chapter explains in excruciating detail the capabilities of UltraMacros. In fact, it's everything you ever wanted to know about macros but were afraid to ask.

The Anatomy of a Macro

Before you can start creating your own macros, you need to understand how a macro is built. The *syntax* of a command is the set of rules governing the organization and usage of that command. In an English sentence, "He here is" would be improper syntax because the "is" should precede the "here". In a like manner, macro commands must be organized in such a way that UltraMacros can understand what you want to have accomplished.

Tokens

Take a look at the macros in the Macros Ultra file (the macros come after the word START and before the word END). Each macro is made up of a series of normal characters and special tokens.

A token is a code word enclosed in <brackets> that represents a special keystroke or macro command. For example, the token <rtn> represents the RETURN key, and the token <left> represents the LEFT-ARROW key. The macro compiler converts these readable tokens into the equivalent invisible command codes within the macro.

Here's a macro a few lines into the Macros Ultra file:

C:<awp><oa-O>CN<rtn><esc>! center text

Each macro begins with a character or token that represents the key used with SOLID-APPLE to activate the macro. In this example, the character "C" indicates that this macro is executed by pressing SOLID-APPLE-C.

Next comes a colon, followed by a token that designates where the macro will work; this macro is for the Word Processor.

Next come the keystrokes and tokens that actually make up the macro. In this example there are five keystrokes: OPEN-APPLE-O, C, N, RETURN and ESCAPE.

An exclamation mark (!) signals the end of the macro definition. Any text after the "!" is ignored. In this example the words "center text" describe what the macro does. They are not considered part of the macro.

Here are some of the tokens that you can use to create macros:

<del>	DELETE key
<esc>	ESCAPE key
<rtn>	RETURN key
<tab>	TAB key
<left>	LEFT-ARROW key
<right>	RIGHT-ARROW key
<up>	UP-ARROW key
<down>	DOWN-ARROW key
<spc>	SPACE BAR key

The tokens for OPEN-APPLE, SOLID-APPLE, BOTH-APPLE and CONTROL commands use the abbreviations oa, sa, ba and ctrl followed by a hyphen and the appropriate key. Here are some examples:

<oa-1>	OPEN-APPLE-1
<sa-B>	SOLID-APPLE-B
<ba-right>	BOTH-APPLE-RIGHT-ARROW
<sa-ctrl-C>	SOLID-APPLE-CONTROL-C

UltraMacros adds a number of unchangeable OPEN-APPLE (page 34) and SOLID-APPLE (page 31) commands to AppleWorks.

Tokens may be entered in upper or lower case, but no spaces are allowed between the letters making up the token. For example, <rtn>, <RTN>, and <Rtn> are all valid tokens for the RETURN key, but <r tn> is not valid.

Multiple consecutive tokens can be used without brackets around each individual token. Just separate the tokens with spaces and/or colons. For example, two UP-ARROW commands followed by a LEFT-ARROW can be represented as <up><up><left>, <up up left>, <up : up : left>, or <up><up left>.

The compiler also allows you to include comments between the <brackets>. Comments are surrounded by curly {brackets}. The previous example could include a comment like this:

<up : up : {this text gets ignored by the compiler} left>

The macro compiler will ignore the curly brackets and everything between them. No macro table space is wasted by using comments. The previous sample will compile into three bytes- two UP-ARROW codes and one LEFT-ARROW code.

Note: If the curly brackets are not between token brackets, they will be treated as normal text. DON'T use token brackets <> inside of the curly brackets {}.

a:<all { comment } stop>!	this one is o.k.
a:<all { --> see? }>!	this one isn't

Local and Global Macros

Each macro must be classified as either local or global. A global macro is one that works anywhere. A local macro is one that works only within a specific application (the Word Processor, the Data Base, the Spreadsheet, or a TimeOut application).

In a macro definition, the token just after the colon indicates whether the macro is local or global:

<all>	ALL applications (global)
<awp>	AppleWorks Word Processor only
<adb>	AppleWorks Data Base only
<asp>	AppleWorks Spreadsheet only
<ato>	A TimeOut application only

You can't have more than one global macro with the same name (the second one will never be used), but you can give the same name to several local macros as long as they are in different applications.

The order in which macro definitions appear in a file is important. When you select a macro, UltraMacros starts at the beginning of the macro table and searches for the first macro with the specified name. When a match is found, the application definition is checked.

1. If the macro is type <all>, it is executed regardless of where you are within AppleWorks or TimeOut.
2. If the macro is type <ato>, it is executed only if you are currently in a TimeOut application.
3. If the macro is an AppleWorks application type, UltraMacros checks to see if you're in the specified application. If so, the macro is executed; if not, it keeps searching.

From this you can see that if multiple macros are created with the same name, the local AppleWorks macros should be first, followed by the TimeOut macros, followed by the global macros.

TimeOut is part of AppleWorks, so you don't have to use <ato> in macros which are designed for TimeOut; <ato> just makes sure that the macros will not run outside of a TimeOut application.

> *Note: BOTH-APPLE macros are not considered the same as SOLID-APPLE macros even if they use the same key. A key such as "A" could conceivably have eight completely different definitions; a BOTH-APPLE and SOLID-APPLE command for <awp>, <adb>, <asp> and <ato>.*

Recorded macros (those defined using OPEN-APPLE-X) are global by default. If you want to make a recorded macro local, list the macros into a file (see page 63) and use AppleWorks to change the <all> token to a local token. Then recompile the macro set.

Calling other macros

One macro can call another macro in two different ways:

y:<all><sa-left oa-M>T<down left rtn>! delete a line
9:<awp : oa-9 : up : goto sa-y>! delete the last line in a file

In the first example, SOLID-APPLE-Y calls macro SOLID-APPLE-LEFT to move the cursor to the left column; UltraMacros then returns to SOLID-APPLE-Y and the current line is moved to the clipboard. In the second example, SOLID-APPLE-9 uses the <goto> command to send control to macro SOLID-APPLE-Y. UltraMacros never returns to SOLID-APPLE-9 because <goto> is a "one-way" command.

Users with Basic programming experience can think of the first example as a GOSUB and the second as a GOTO (how fitting). Just remember that using a macro NAME will continue the current macro when the called macro is finished, and that using GOTO means that the macro will never come back.

"Macro nesting" occurs when a macro calls a macro which calls a macro... UltraMacros has to remember where to back up to when the current level is finished. The limit is 18 levels. A macro which calls itself will also execute 18 times and then stop.

1:<all>*<sa-1>! print 18 asterisks

To execute a procedure more often, use <begin> and <rpt> along with variables (they are explained later).

<ba-1>:<all : A = 120 : begin : print "*" : A = A - 1: if A > 0 then rpt>! print 120 asterisks

CAUTION: When you're about to delete a macro from a file, make sure the macro isn't needed by another macro in the file. Use the OPEN-APPLE-F command to search for references to the macro. For example, if you plan to delete macro F, search for "sa-f".

Reserved Macros

The special macros listed below cannot be re-recorded, changed or deleted; you must use them "as is". You can use these macros at any time (unless otherwised noted): directly from the keyboard (press the appropriate key along with SOLID-APPLE), while recording a macro (press the appropriate key along with SOLID-APPLE), or in a macro definition (use the appropriate token).

<sa-del> SOLID-APPLE-DELETE

Deletes the character under the cursor.

<ahead> SOLID-APPLE-.

Finds the first blank space to the right of the cursor position. This macro works wherever AppleWorks allows you to edit characters, including Word Processor files, Data Base categories, at Find prompts, and when AppleWorks prompts you to enter names.

<back> SOLID-APPLE-,

Finds the first blank space to the left of the cursor position.

<date> SOLID-APPLE-'

Displays the date in this format: August 10, 1987 (handy for dating letters or Data Base and Spreadsheet reports).

<date2> SOLID-APPLE-"

Displays the date in this format: 08/10/87 (handy for dating transactions in the Spreadsheet).

<time> SOLID-APPLE-=

Displays the time in this format: 1:42 pm. If you don't have a clock, the time will always be 12:00 am.

> *NOTE: If you have a IIgs and 12:00 am is always given for the time, you'll need to copy the PRODOS file from the UltraMacros disk (it's a newer version that can read the IIgs clock) to your AppleWorks startup disk. Use TimeOut FileMaster or Apple's System Utilities to copy it.*

<time24> SOLID-APPLE-+

Displays the time in this format: 13:42. In the Data Base, if a category has the word TIME included in its name, AppleWorks converts 24-hour times to 12-hour times. For example, 21:46 is converted to 9:46 PM.

<find> SOLID-APPLE-RETURN

In the Word Processor: moves the cursor to the next carriage return marker. This command only works when the `Type entry...` message is visible at the bottom of the screen.

At any numbered inverse bar menu or file list: searches for the text stored in macro 0 (zero) and leaves the cursor at that item. If the text is not found, the current macro will stop. If the current macro was called from another macro, the calling macro will continue. This means that macros containing <find> should end with a <stop> if they are called from another macro. This way the calling macro continues only if the item is not found. See the Control-P phone macro in Macros Ultra for an example.

If you're at a file list and want to find a file in a hurry, press OPEN-APPLE-0 to define macro 0 (zero) with the name and then press SOLID-APPLE-RETURN to find the file. From within a

macro, you can use this command to automatically load files by name.

This command also works with the OPEN-APPLE-Q Desktop Index and the OPEN-APPLE-ESC TimeOut menu. Macros can find desktop files by name, or start TimeOut applications by name. Define macro 0 (zero) ahead of time; you can not use OPEN-APPLE-0 while the OPEN-APPLE-ESCAPE TimeOut menu is on the screen.

Use the ability to search a menu to find printers by name when you aren't sure what order they'll be in.

NOTE: The <find> command only uses the first 15 characters in Macro 0 ($0).

NOTE: <find> can be used with <store> and <recall> to link files. See page 73 for details.

<findpo> SOLID-APPLE-^

in the Word Processor only:

Moves the cursor to the next caret (^). The caret may be a printer option caret or a text caret which is part of the document. This macro only works when the `Type entry or ...` message is visible at the bottom of the screen.

This macro is easier to use for locating printer options in a file than the OPEN-APPLE-F command, which requires you to know which option you're searching for (and its two-letter code). <findpo> simply searches for the next caret in the file regardless of what it represents.

A macro can use the <screen> command to check what kind of option was found by <findpo>. For example, this macro finds the next superscript or subscript code, but ignores all other printer codes or carets:

<ctrl-s>:<awp><findpo : $8 = screen 42,24,4 : if $8 = "Subs" then stop else if $8 = "Supe" then stop else rpt>!

<print> BOTH-APPLE-CONTROL-P

The print token is for use within macros only. See page 49 for a full description of how <print> is used.

New Open-Apple Commands

The following commands can be used directly from the keyboard as well as from within macros. If you're recording a macro, press the appropriate key along with OPEN-APPLE. To use the command in a macro definition, use the token.

no token OPEN-APPLE-X

Begin recording a macro. This command must be used from the keyboard only; it can't be used within a macro. See page 17.

"sa-del" OPEN-APPLE-DELETE

Deletes the character the cursor is on. This command is identical to SOLID-APPLE-DELETE.

Note: When recording a macro, you must use SOLID-APPLE-DELETE.

"getstr" OPEN-APPLE-0 (zero)

Presents a ">" prompt on the bottom line of the screen, allowing up to 60 characters to be entered for defining macro 0 (zero). This command is used from the keyboard only. Do not use it while recording a macro. See the description of the true <getstr> token on page 45.

<oa-ctrl-@> OPEN-APPLE-CONTROL-@

Sends a CONTROL-@ to AppleWorks. Use this while recording or defining a macro. If you just use CONTROL-@ the macro will stop at that point. CONTROL-@ is used only for printer and interface definitions.

> *NOTE: AppleWorks 2.0 has a bug which doesn't allow CONTROL-@ to be entered. See page 72 for info on squashing this bug.*

<uc> OPEN-APPLE-:

Changes the character at the cursor to upper case.

<lc> OPEN-APPLE-;

Changes the character at the cursor to lower case.

<insert> OPEN-APPLE-!

Turns on the insert cursor (the blinking underscore). To turn on the overstrike cursor (blinking rectangle), use this command followed by an OPEN-APPLE-E command.

<zoom> OPEN-APPLE-@

Forces zoom OUT (hides printer options in the Word Processor, shows values rather than labels in the spreadsheet, and shows multiple-record layout in the Data Base). Follow this command with OPEN-APPLE-Z to zoom in.

<read> OPEN-APPLE-^

From the keyboard, OPEN-APPLE-^ will read the character at the current cursor position into macro 0 (zero). You can use the arrow keys to move the cursor to a new position before reading another character.

While recording a macro, OPEN-APPLE-^ will read the character at the current cursor position into the macro being recorded (the character will become text in the macro definition).

In a macro definition, <read> will read the character at the current cursor position and add it to macro 0 (zero).

<disk> OPEN-APPLE-&

Reads the current volume (disk) name or subdirectory pathname into macro 0 (zero). This command can only be used when a list of files is being displayed. A brief flash at the top left of the screen indicates that the command was executed.

<path> OPEN-APPLE-*

Reads the current volume name or subdirectory name AND the currently highlighted file name into macro 0 (zero). This command can only be used when a list of files is being displayed. A brief flash at the top left of the screen indicates that the command was executed.

<cell> OPEN-APPLE--

Reads the contents of the current Spreadsheet cell, Data Base category or Word Processor line into macro 0 (zero). Move the cursor to the cell and use the command. A brief flash at the top left of the screen indicates that it was executed. See <ba--> in Macros Ultra.

The current layout and display settings do not affect <cell>. In the Spreadsheet it uses the literal value or label as displayed on the cell indicator line, and in the Data Base it uses the full category entry as shown in the single-record layout.

From within a macro, use <cell> as part of any string definition like this:

c:<asp : $3 = cell : down : print $3>! copy a cell

<store> OPEN-APPLE-<

Stores the current contents of macro 0 (zero), up to 15 characters, in a special unused area of a Word Processor, Data Base or Spreadsheet file. The name being stored is displayed at the bottom right of the screen. This command is designed for linking files (page 73), but it may be used for any other purpose you think of.

<recall> OPEN-APPLE->

Sets macro 0 (zero) equal to the text stored by the <store> command. <store> would be rather useless if the information couldn't be recalled.

<inc> OPEN-APPLE-CONTRL-W

Increments the character at the current cursor position. For example, "a" becomes "b", "1" becomes "2", etc. See the Key Chart (page 87) for a complete list of characters in the order they're incremented and decremented.

<dec> OPEN-APPLE-CONTROL-A

Opposite of <inc>; decrements the character at the current cursor position.

<bell> OPEN-APPLE-CONTROL-G

Sounds the AppleWorks error bell once. It's handy for getting someone's attention.

<nosleep> OPEN-APPLE-CONTROL-N

Cancels the currently defined "sleeping" macro, if any. See the description of <wake> for more information.

<clear> OPEN-APPLE-CONTROL-X

Clears all numeric variables to 0 and all string variables to no definition. A brief flash at the top left of the screen indicates that the command was executed.

IIgs users can press OPEN-APPLE-CLEAR; the CLEAR key on the numeric keypad is the same as CONTROL-X.

Special UltraMacros Tokens

The following tokens are for use within macro definitions only. None of them are keyboard commands, and they can not be recorded using the OPEN-APPLE-X command. They are used as is (i.e. they require no parameters).

<input>

Allows you to enter text or OPEN-APPLE commands until RETURN is pressed (the RETURN is NOT passed on to AppleWorks). To exit this command without pressing RETURN, enter CONTROL-@. The macro will be aborted.

<id#>

Returns the unique id number of the current TimeOut application. If TimeOut is not active, a zero will be returned. This token can't be used by itself; it must be part of a variable definition or other numeric expresssion.

a:<all : A = id# : $1 = "This TimeOut application is #" + str$ A : msg $1>! determine the TO application number

a:<ato : a = id# : if a = 7 then msg ' FileMaster ' stop else if a = 8 then msg ' Macro Compiler ' stop>! act differently for each

<ifkey>

Checks to see if a specific key has been pressed (exact matches only) and if so continues. This is not part of the if-then-else logic.

a:<all : sa-b rpt>!
b:<all : ifkey rtn then print "Return was pressed">!
or
b:<all : ifkey>A<then print "A was pressed">!

<key>

Pauses until a key is pressed. The keypress is NOT passed along to AppleWorks. When used by itself, key is simply a pause feature.

In an equation, key returns the value of the key pressed. For example:

a:<all : A = key : if A < 128 then print A>!

If the user presses RETURN, A will be 13, and if the user holds down OPEN-APPLE while pressing the key 128 will be added to the key value. This example will only print the keystroke if OPEN-APPLE is not pressed.

<begin>

This does nothing unless used with <rpt>. It marks the restarting point for repeating part of a macro instead of repeating the entire macro.

<rpt>

Repeats part or all of the current macro by searching backwards from the <rpt> token until a <begin> is found, or until the beginning of the macro is reached.

No commands after <rpt> will ever be executed (unless they're part of an IF-THEN-ELSE statement). A conditional command must be used to exit the macro or it will run continuously. For example:

g:<all : bell : rpt>! could drive you crazy; press Escape to exit

h:<all : print "This part executes once" : begin : bell : rpt>! prints a message, then beeps like crazy

<stop>

Stops all macro activity immediately. Use it to stop a nested macro from returning to the calling macro, or to get out of a <rpt> situation. For an example, go back to the <findpo> sample on page 33.

Special UltraMacros Tokens with parameters

The next group of tokens require additional parameters. Most parameters involve variables, so a description of UltraMacros' variables is next. The token definitions are continued following the variable section.

Defining Numeric Variables

Numeric variables may be defined many different ways. You must be careful that you don't accidently redefine a variable if another macro expects to use that variable later.

We suggest leaving variables U, V, W, X, Y, Z as "throw-away" variables. Assume that they can be redefined indescriminately by any and all macros. We also suggest reserving variable Q for recording the number of a file you leave via the OPEN-APPLE-Q (get the connection?) and that you want to return to later.

Start up AppleWorks and insert the UltraMacros disk. Add the file Macros Ultra to the Desktop and examine the variable usage in it. See macros SOLID-APPLE-1 and SOLID-APPLE-2 for examples of using variable Q.

Here's a chart showing the various ways to define numeric variables and use them in conditional macros:

condition	var	operator	operand	
			X	variable
if	A	>	7	decimal number
(define)	thru	=	$10	hexadecimal number
ifnot	Z	<	key	keyboard input
			asc $0	ASCII value of first char
			len $1	length of a string
			val $2	value of a string
			peek	value at an address
			id#	TimeOut application #

Remember those crazy mix-and-match animal cards when you were a kid? This is the same idea, except that a variable can only be defined using the "equals" operator. Otherwise you can pick any item out of each category and use them together in a macro.

Any number of operands can be chained together using the four basic math operators (+ - / *). No parentheses are allowed. The equations are strictly evaluated left to right with no other precedence.

Defining String Variables

String variables may be defined in many different ways. Literal strings may be surrounded by single or double quotation marks:

a:<all : $8 = "This is a literal text string">!
a:<all : $9 = 'This example has "quotation marks" in it'>!

Strings may be defined as the current date or time in these four formats:

a:<all : $0 = date : $1 = date2 : $2 = time : $3 = time24>!

Strings may be defined as the current Spreadsheet cell, Data Base category or Word Processor line:

a:<all : $8 = cell>!

A portion of the screen may be used to define a string (see the description of <screen> on page 51):

a:<all : $6 = screen 7,1,15>!

A string may be defined by user input from the keyboard:

a:<all : $3 = getstr 15>!

See the description of getstr (page 45) for more information.

Finally, a string may be defined exactly like another string. In this example, $7 is made identical to the current value of $2:

a:<all : $7 = $2>!

Here's a chart showing the various ways to define string variables and use them in conditional macros:

condition	str	operator	operand	
			"text"	a literal string
			$2	another string variable
if	$0	>	date	,date2
(define)	thru	=	time	,time24
ifnot	$9	<	getstr	keyboard input
			cell	db category, ss cell
			screen	80-column text screen
			chr$	ASCII value of a variable
			str$	string equivalent of a variable
			left	left portion of a string
			right	right portion of a string

Any number of operands can be chained together using concatenation (+). No parentheses are allowed. The equations are strictly evaluated left to right with no other precedence. Any characters beyond 80 are ignored.

Parameters

Here are some of the possible parameters for UltraMacros tokens:

MACRO
a macro name such as SA-B or BA-CTRL-D

NUM (number)
a literal decimal number from 0 to 65535
a literal hexadecimal number from $0 to $FFFF
a variable name from A to Z (the value of the variable is used)

NUM VAR
a variable name from A to Z (the value of the variable is used)

NUM2
a NUM (see above)
a <key> token
a <peek> token
a <len> token
a <val> token
a <id#> token
a <asc> token

NUM EXP (numeric expression)
a NUM (see above)
a NUM2 (see above) if the compiler gives an error, NUM2 is unavailable for this particular command definition
a NUM or NUM2 equation; NUM's must be connected by +, -, /, or *; the equation is evaluated from left to right. No other precedence is used.

The range of values is 0 to 65535 and the numbers "wrap around" if the range is exceeded. For example, 0 - 1 = 65535 and 65534 + 3 = 1. Because only integer numbers are allowed, division will only return the quotient.

STRING
a literal string surrounded by quotes

STRING VAR
a string name from $0 to $9

STRING EXP (string expression)
a STRING (see above)
a <chr$> token
a <str$> token
a <date> token
a <date2> token
a <time> token
a <time24> token
a <cell> token
a <screen> token
a <getstr> token
a STRING equation; STRING EXP's must be connected by "+" only; the equation is evaluated from left to right until the maximum length of 80 characters in reached.

<asc STRING VAR>

Converts the first character of a string to its ASCII equivalent.

x:<all $0 = "z" : x = asc $0 + 2: print x>! prints "124"

<chr$ NUM>

Prints the ASCII value of a variable. See the Key Chart for a complete list. For example, the following macro will print the number 1:

a:<all : X = 49 : print chr$ X>!

As you can see on the chart, 49 is the code for an upper case 1. If X was equal to 177, an OPEN-APPLE-1 command would be executed. This command is handy for sending special codes to your printer along with the <pr#> token.

<getstr NUM>

Presents a ">" prompt on the bottom line of the screen, allowing up to 60 characters to be entered. (This is similar to OPEN-APPLE-0.) In the Data Base, <getstr> must always be followed by <rtn>.

a:<all : $1 = getstr 8 : print $1>! read in 8 characters and print them

<goto MACRO>

Sends control to the specified macro name. If goto is not used, the second macro will return to the original macro and continue there. It will behave like a gosub. Goto just jumps to the named macro and keeps on going. No nesting occurs when goto is used.

a:<all : sa-b : print "The end">!
b:<all : if A = 4 then goto sa-c else rtn>!
c:<all : print $4>!

<hilight NUM EXP,NUM EXP,NUM EXP,NUM EXP>

Allows you to invert any portion of the AppleWorks screen. This sample will invert the entire screen:

h:<all : hilight 1,1,80,24>!

The first parameter is the left column (1-80)
The second parameter is the top row (1-24)
The third parameter is the right column (1-80)
The fourth parameter is the bottom row (1-24)

If the first parameter is 0, the specified rows will have all highlighting cancelled (the text will be changed to normal). The right column value is ignored if the left column is 0.

<left STRING VAR,NUM>

Extracts the leftmost number of characters specified from a string.

a:<all : $1 = "Beagle Bros" : $2 = left $1,6 : print $2>! prints Beagle

<len STRING VAR>

Returns the length of the specified string as part of a variable equation. For example:

a:<all : A = len $0 : B = len $1 : if A > B then print "$0 is longer">!

<msg STRING>

This command prints a message on the screen immediately below the current data window (i.e. on the dash "-----" or underline "_____" dividing line). The command syntax is identical to <print>. Messages are displayed in inverse text unless the message string is surrounded by double quotation marks:

```
m:<all : msg " Normal text " : key      { display normal text; wait }
msg ' Inverse message ' : key           { display inverse text; wait }
$9 = date + " " + time : msg $9 : key   { display date & time; wait }
msg "">! do sample messages and then erase all messages
```

Whenever a message is displayed, the remainder of the line is filled with the second to last character that was already on that line (i.e. the character above the "p" in `A-? for help`). This automatically erases the vestiges of previous longer messages. As the example shows, a null message erases the entire line. A message expression must always be followed by a : or >.

<onerr OPTION>

Allows you some control over what happens if an error occurs. An error is defined as a keystroke that causes AppleWorks or a TimeOut application to ring the error bell. Normally a macro continues on without regard to the error (the error bell is silenced as well). There are three onerr options:

1:<all : onerr stop>! stop current macro after an error

If the macro was called from another macro, control returns to the calling macro. This does not shut down all macros; only the current macro is ended.

2:<all : onerr off>! revert to normal; ignore all errors

Resets the onerr status to normal, so macros ignore the errors, for better or for worse.

3:<all : onerr goto sa-h>!

On any error, execute the named macro and then return to the calling macro where the error occurred.

The onerr status is always reset to normal when a sequence of macros is done executing.

<posn VAR,VAR>

Assigns the current cursor position to the two variables following the token. The AppleWorks application affects the command like this:

	First variable	Second variable
Word Processor	column	line
Data Base	category	record
Spreadsheet	column	row

If the cursor is not in one of these three applications, both variables will be set to zero. <posn> is compatible with TimeOut applications that use the AppleWorks applications. For example, TimeOut Graph works in the Spreadsheet, so <posn> can be used with it.

Start up AppleWorks and add the file Macros Ultra from the UltraMacros disk to the Desktop. Examine the cursor-positioning macros for examples of <posn> usage.

<pr# NUM EXP>

Determines where the <print> command sends its information.

<pr# 0> sends all <print> characters to AppleWorks. This is the normal state of affairs.

<pr# 1> sends the characters to the first printer in your AppleWorks printer list. Because AppleWorks' limit is 3 printers, the <pr#> limit is also 3. You must use <pr# 0> to reset the <print> command when you're done.

Start AppleWorks and add the files Macros Printer and Macros Dialer from the UltraMacros disk to the Desktop. Examine them for sample <pr#> usage. <pr#> may not work with all interface cards. It does work with the printer and modem ports on the IIgs.

<print>

Print has the most variations of any single UltraMacros command. The compiler will be happy to point out any errors you might make, but studying this section will make you much less error-prone.

Printing Text

Like Applesoft, <print> allows a literal text string to be printed. You may use either double or single quotes around the text. The limit is 70 characters of text at a time.

a:<all : print "Literal text <rtn>">! prints "Literal text <rtn>"

The <rtn> is NOT converted to an actual RETURN.

a:<all : print '"double" quotes inside "single" quotes'>! will print as "double" quotes inside "single" quotes

Printing Numeric Variables

Print can be used to display the value of any numeric variable. For example, if variable Q holds the desktop number of a specific file, this macro sequence would return you to that file:

2:<all : oa-q : print Q : rtn>!

A text string may precede the variable, like this:

a:<all : print "Variable A is now " A>!

When printing numeric variables, a "$" immediately after the print statement will cause the variable's hexadecimal value to be displayed in either two or four characters.

a:<all : A = 8 : print$ "Hex A = $" A>! will print "Hex A = $08"
a:<all : X = 61453 : print$ X>! will print "F00D"

Numeric variables can also be printed as characters rather than numbers. The <chr$> token converts the numeric value to the

equivalent key command. See the Key Chart for a complete list. Here's a sample:

a:<all : X = 185 : Y = $41 : print chr$ X : print chr$ Y>!

The Key Chart shows us that 185 is an OPEN-APPLE-9 and that $41 is an upper case "A". This sample will jump to the end of the file and then print an "A".

Printing Strings

The ten string variables may be printed by themselves only. No other options may be used when printing strings. These strings may contain text or command keystrokes. To define a string with commands instead of text, just define macro 0 (zero), the same as $0, and then use a macro like this:

a:<all : $2 = $0 : print $2>! execute macro 0 (zero)

Because macro 0 (zero) and $0 are the same thing,

a:<all : print $0>! is exactly the same as
a:<all : sa-0>!

NOTE: ALL PRINT STATEMENTS MUST BE FOLLOWED BY A COLON ":" OR ">". Other tokens can be followed by spaces and then another token, but <print> is an exception.

<rem STRING>

This command allows you to imbed a remark in the middle of a macro. The remark does not get used in the macro. The command syntax is identical to <print>. Just surround the remark with single or double quotation marks.

The difference between using <rem> as opposed to just adding a comment after the "!" mark or by using the curly brackets {...} is that <rem>'s are preserved when the macro is assembled, so they are still present when you use the Macro Compiler to list the current macros into a Word Processor file. We prefer to use the curly

brackets and to make sure we keep our Word Processor source files handy, but it's your option.

Here's a macro from the Macros Ultra file that deletes from the cursor to the end of the file. Actually, it moves the data to the clipboard so that it can be undone. It's a simple example, but it does illustrate how to use <rem>'s:

```
Z:<all : oa-M>T<rem "move to the the clipboard" :
        oa-9 : rtn : rem "jump to the end and do it" :
        left>!      zap to end of file
```

Comments and <rem>'s make it easier to follow the logic of a macro, especially if you want to change a macro you wrote last month or last year.

<right STRING VAR,NUM>

Extracts the rightmost number of characters specified from a string.

a:<all : $1 = "Beagle Bros" : $2 = right $1,4 : print $2>! prints Bros

<screen NUM EXP,NUM EXP,NUM EXP>

Reads any part of the AppleWorks screen into a string variable. It is used like this:

s:<all : $1 = screen 7,1,15>! read current file name from top line

The first parameter is the left column (1-80)
The second parameter is the line (1-24)
The third parameter is the length (1-80)

Screen treats all characters as normal text, regardless of how they appear on the screen; it can't be used to tell if something was highlighted.

<str$ VAR NAME>

Converts a numeric variable to a decimal character string. It must be used as part of an equation. Here are some examples:

x:<all : A = 4 : $3 = " A = " + str$ A + " " : print $3>! prints " A = 4 "
y:<all : B = $ff : $1 = str$ B : print $1>! prints "255" because 255 is the decimal equivalent of the hexadecimal $ff used to define B.

This command is handy for including variables in a message:

a:<ato : I = id# : $1 = "TimeOut ID#" + str$ I : msg $1>! show id#

<val STRING VAR>

<val> is the opposite of <str$>. It converts a string variable to a numeric value and must also be used as part of an equation. If the specified string starts with a non-numeric character, the value will always be zero. If the first character is a number, it (and all other numbers following immediately after it) will be converted to a numeric value. Here are some examples:

a:<all : $3 = "test4" : A = val $3> A will be 0
a:<all : $2 = "48612" : X = val $2> X will be 48612
a:<all : $8 = "280ZX" : V = val $8> V will be 280
a:<all : $0 = "14.88" : B = val $0> B will be 14

<wait NUM EXP>

Delays a macro for a set amount of time, or until a key is pressed. The actual delay will vary depending on your computer. Experiment to find the approximate delay needed for a second or a minute on your computer. Here's a macro you can use to calculate delay values.

D:<all : msg "Enter a delay value " : $0 = getstr,5 : D = val $0 : bell : wait D : bell : rpt>! test wait

One suggested use for the <wait> command is to allow a user to browse through a large document without having to touch the keyboard:

b:<awp : D = 400 : wait D : down : rpt>! adjust D as desired

<wake MACRO at NUM EXP:NUM EXP>

Puts a macro to "sleep" and wakes it at a designated 24-hour time. After a <wake> command has been issued, you can work normally, using macros and any UltraMacros commands as always. When the clock's time matches the sleeping macros, it springs to life. Use it to set alarms, automatically save a file every few minutes, or shut everything down at 5:00!

The following example will start macro "A" at noon. Then when macro "A" wakes up, it will set macro "C" to wake up at 5:00 pm.

a:<all : wake sa-b at 12:00>!
b:<all : bell : bell : bell : msg "It's lunch time!" : wake sa-c at 17:00>!
c:<all : bell : bell : bell : msg ' Quitting time '>!

Only one macro can be "sleeping" at a time, but as shown in the previous example, each macro that "wakes up" can put another macro to "sleep". The time must be given in 24-hour format (0:00 to 23:59); variables may be used:

x:<all : M = 50 : H = 7 : goto sa-s>!
s:<all : oa-s : M = M + 10 : if M = 60 then M = 0 : H = H + 1 :
elseoff wake sa-s at H:M>!

When SOLID-APPLE-X is pressed, the hour and minute variables are set. The current file is saved, and macro S is set to wake up at 8:00 am. It will then save the current file every 10 minutes until the <nosleep> command is used.

Note: Of course, this command won't be too useful if you don't have a clock in your Apple.

If-Then-Else Logic

One of UltraMacros' best features is its true conditional capability utilizing if-then-else logic. Other macro programs may allow a conditional "if" or two based on keyboard input, but UltraMacros allows a full range of conditional commands using the numeric and string variables. Five tokens are involved with conditional logic: if, ifnot, then, else, elseoff.

<if>

The key to all conditional macros, <if> is always followed by a numeric or string variable

a:<all : if A

which is followed by an operator (greater than >, less than <, or equals =)

a:<all : if A =

which is followed by the expression to be evaluated.

a:<all : if A = 5

If the statement is true, the macro continues normally. If the statement is not true, the macro ends (unless an <else> is present later in the macro).

a:<all : if A = 5 goto sa-5>! if A is not 5, the macro stops here

<ifnot>

Same as <if>, except that the statement must be false for the macro to continue normally.

a:<all : ifnot A = 5 goto sa-5>! if A IS 5, the macro stops here

All numeric conditionals must start with one of these six formats:

if A =	or	ifnot A =
if B >	or	ifnot B >
if C <	or	ifnot C <

The equation is then completed with any valid numeric expression such as:

a:<all : if A = C + 4 then print A : else stop>!

All string conditionals must start with one of these six formats:

if $0 =	or	ifnot $3 =
if $1 >	or	ifnot $4 >
if $2 <	or	ifnot $5 <

The equation is completed with a string expression, which could be another string variable name, a literal text string, or one of the legal string definition tokens (date, date2, time, time24, screen):

a:<all : if $0 = "literal" then print $0>!

t:<all : ifnot $6 > time then goto sa-t>!

2:<all : if $2 < screen 1,7,15 then stop>!

8:<all : ifnot $8 = $9 then print "They're not the same.">!

<then>

Does absolutely nothing but take up one byte of space. It's used to make macro if-then-else logic more readable:

a:<all : if A > 4 then C = 3>! this looks better than:

a:<all : if A > 4 C = 3>!

<else>

This part of if-then-else logic reverses the true-false condition of the logic. If the statement is true, then execute the first part, else execute the second part. There is no limit to the number of else's in one macro.

Whenever an else is encountered during a macro, the macro skips ahead to the next else or to the end of the macro, whichever is found first. It can be used during debugging to keep part of a macro from executing so a different part can be properly tested:

a:<all : print "A " : else print "one" : else print "two">! test part 2

This macro will always print "A two". Later the macro could become:

a:<all : print "A " : print "one" : else print "two">! test part 1 only

Now the macro will always print "A one". When finished, the macro could be:

a:<all : print "A " : if A = 1 then print "one" : else print "two">!

There is no direct connection between if and else, so they can be used independently, although they make a good team. Here's a sample:

a:<all : if A = 5 then print "Five" : else print "Not Five">!

This does just what you'd expect. If variable A is equal to 5, the word "Five" is printed. If variable A is not equal to 5, the words "Not Five" are printed.

<elseoff>

Does nothing unless used with else (See above). It's purpose is to cancel the conditional status of a macro and cause any commands following the "elseoff" to be executed regardless of any preceding "if" conditions.

For example, if a conditional macro is supposed to print a phrase at the end regardless of what other text is printed, you'd do this:

a:<all : if A = 5 then print "A is five" : else print "A is not five" : elseoff print " at this time.">!

If A is five it would print "A is five at this time." and if A is not five, it would print "A is not five at this time."

If the "elseoff" is removed, the macro will print either "A is five" or "A is not five at this time."

For experienced 6502 programmers only!

The following three UltraMacros tokens are very specialized and shouldn't be used unless you understand exactly what you want them to do. They were included because we thought there might still be a few hackers out there who like to deal directly with their Apples. Besides that, we can always use them to write some pretty powerful macros ourselves.

See the file Macros Special on the UltraMacros disks for some examples.

<call>

The call token is used to run machine language subroutines. It simply does a jsr to the address specified. It's up to you to make sure that the address is valid and that the routine will return to the macro via an rts with all bank switches set properly.

A good place to poke in machine language subroutines is the AppleWorks temporary work buffer from $800 to $9FF.

> *CAUTION: The buffer is destroyed by AppleWorks disk access and by a few UltraMacros commands. Be careful!*

When a macro is operating, the alternate zero page is active, as well as the second bank of $D000 memory. Page 1 of the 80-column display is active. If you change any of these, they MUST be restored before your routine returns control to UltraMacros or AppleWorks will surely die.

The call command is a bonus feature and must be used carefully by experts only. Know what you're doing before you use this!

<poke>

Poke is a handy, albeit dangerous, command. Use it to build machine language subroutines for use with <call>.

It can also be used to directly change some flags within AppleWorks. For example, the <insert> token forces the insert cursor on. To force the overstrike cursor on, you can use this:

O:<all : poke $10F1,1>! force overstrike cursor active

The insert cursor continues to blink until another key is pressed, and then you see that the cursor is changed. To make it instantly change, add an invalid key like this:

O:<all : poke $10F1,1 : ctrl-x>! force overstrike cursor active and make it immediately obvious

We aren't authorized to provide a list of AppleWorks addresses. You'll have to explore on your own. Some information is available from bulletin board systems. Last time we heard, AppleWorks author Bob Lissner had an AppleWorks board somewhere in Nevada with a lot of handy info. Sorry, we can't provide a phone number.

<peek>

Peek returns the value found at the specified address. This example from the Macros Ultra file uses <peek> to determine the current file number:

```
1:<all><q = peek $C54 : oa-q esc>!   leave "1" file; go to main menu
2:<all  : oa-q print q : rtn>!       return "2" the file we left
```

Macro Compiler

This application allows you to compile new macros. It scans a Word Processor document containing macro definitions and converts them into a form useable by TimeOut UltraMacros. It can also display the current macro set by listing the active macro definitions into a Word Processor file.

If you've been following the manual sequentially, you've already used the Compiler to include your name in the default macro set. In any case, add a Word Processor macros file to the Desktop and press OPEN-APPLE-ESCAPE. Select `Macro Compiler`. You will see two options.

Compile a new set of macros

1. Press RETURN to compile a new set of macro definitions.

2. A `Pause each line?` prompt appears. If you press `Y` for `YES`, you will "single-step" through the compiling process and will be asked to press a key to continue after each line is processed. If you choose `N` for `NO`, the compiler will race along and won't stop until an error occurs or until compiling is completed.

3. A `Compile from?` prompt appears. You may choose to compile from the very beginning of the file or from whatever line the cursor was on when you pressed OPEN-APPLE-ESCAPE to call up TimeOut. If you press `B` to compile from the beginning, the compiler will scan the file until it finds the word "START" on a line by itself.

4. Each Word Processor line will flash on the screen as it is scanned for tokens and text data. The current macro name will be displayed, and a running count of the total bytes used will also be shown.

5. If the file is compiled without error, a message will indicate the success and you will be prompted to press a key to return to the file.

6. If an error occurs, the compiler will attempt to give you as much information as possible. When you press a key and return to the Word Processor, the cursor will be on or near the error.

Macro Compiler Errors

There are several errors which stop the compiler. Macros up to the error are useable, but no macros following the error line are compiled. When you press a key after reading the error message, you will be returned to the file with the cursor on or near to the error.

No errors

The compiler recognized the entire macro file as a valid macro set. That doesn't mean that the macros don't contain potential execution errors. It does mean that all command tokens were used properly.

No Start found

The compiler couldn't find the word START in the file. The word START must be on a line by itself, immediately preceded and followed by carriage returns (no spaces). This message can't occur if you select `Compile from cursor`.

Reserved macro name

An attempt was made to define a reserved macro (see page 31). Remember that the named keys such as RETURN and TAB have CONTROL-KEY equivalents.

Incorrect application name

A valid local/global designation couldn't be found. Valid tokens are <all>, <awp>, <adb>, <asp> and <ato>. Every macro must start with the macro name, followed by a colon ":" followed by the application name.

Table full

The entire macro table was used up. This refers to the actual macro bytes generated and is not related to the size of the Word Processor file being compiled. The only solution is to create more macro files.

Syntax error

The catch-all error message. Anything the compiler doesn't recognize is a syntax error. This is usually a misspelled token name.

Not enough parameters

The UltraMacros' token being compiled requires one or more additional parameters. See page 40 for a description of all tokens which use additional parameters.

Too many parameters

The UltraMacros' token being compiled requires at least one fewer parameter. See page 40 for a description of all tokens which use additional parameters.

Logic error

The macro definition was illogical. You might not think so, but guess who's the boss? Remember that UltraMacros is very strict and limited in its string and numeric variable usage.

String too long

A sequence of text characters used in a string definition exceeds the character limit.

END found in line

The compiler was stopped by an END command, not by an error. All macros up to the END line are useable.

Stopped by Escape

You pressed the ESCAPE key during compiling to halt the process. All macros already compiled are useable.

Display current macro set

This option will list all of the current macros into a Word Processor file. The listing can then be modified and recompiled as desired. The primary purpose of this option is to display recorded macros.

1. Add an AppleWorks Word Processor file to the Desktop. It can be an existing file, or a new one.

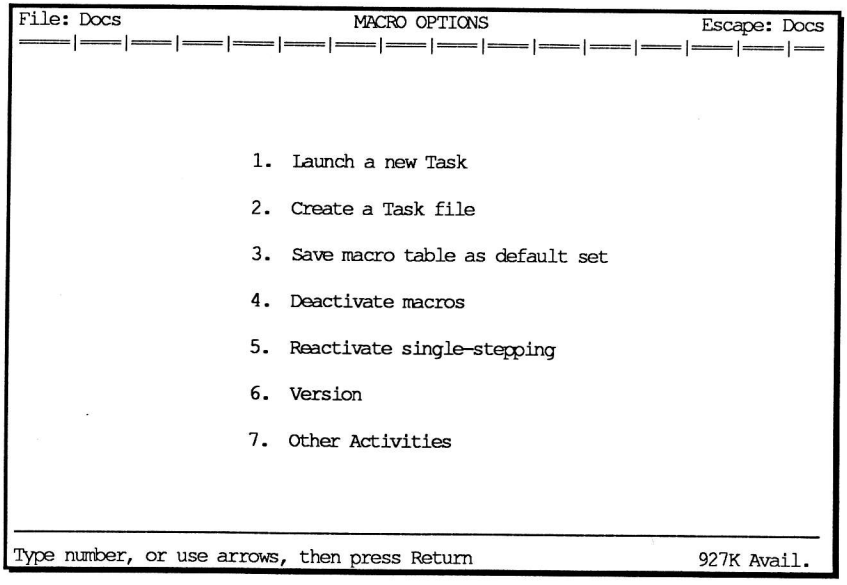
2. Press OPEN-APPLE-ESCAPE to call up the TimeOut menu. Select `Macro Compiler` from the menu.

3. Select `Display current macro set`.

4. The current set of macros will be added to the file in token format. This file can then be changed and recompiled as desired.

Macro Options

The Macro Options TimeOut application contains UltraMacros' user-definable options. From within AppleWorks, press OPEN-APPLE-ESCAPE and select `Macro Options` to get this menu:



The first two options on this menu deal with Task files. Read page 74 for a description of what Task files are and how they work.

1. Launch a new Task

When selected, this option reads the AppleWorks disk and looks for Task files. The names are shown on the screen. Press ESCAPE to return to the `Macro Options` menu, or select the task you want to launch.

2. Create a Task File

This option takes whatever macros are currently active in AppleWorks and saves them as a Task file on your AppleWorks startup disk.

First you are prompted to enter a name for the new Task file. Enter a legal ProDOS name (you know, 1 to 15 characters beginning with a letter and containing only letters, periods or numbers).

If the AppleWorks startup disk is not found, you will be prompted to insert it. At this time you can press Escape to cancel, or you can insert the startup disk and press Return.

When the macros have been updated, put the previous disk back in the drive.

3. Save macro table as default set

This option takes whatever macros are currently active in AppleWorks and saves them into the file "ULTRA.SYSTEM" on your AppleWorks startup disk. These macros will be then be available whenever you start AppleWorks. No compiling will be necessary.

Before the macros are saved you are asked if you want to activate the auto startup macro. If you choose `Yes`, the first macro in the macro set will be automatically run each time AppleWorks is started.

If the "ULTRA.SYSTEM" file is not found, you will be prompted to insert your AppleWorks startup disk. At this time you can press Escape to cancel, or you can insert the startup disk and press Return. When the macros have been updated, put the previous disk back in the drive.

4. Deactivate macros

This option completely disconnects UltraMacros from AppleWorks. Choose this option if you wish to print using the Applied Engineering print buffer. After deactivating, the menu will appear as `4. Reactivate macros.`

When you're through printing, select the option again to get your macro power back.

5. Reactivate single-stepping

This option turns on "single-step" mode, which is useful for "debugging" complex macros. The menu will change to `5. Deactivate single-stepping.`

When a macro sends a character to AppleWorks, this option forces a pause before each character actually gets to AppleWorks; the character is sent only after you press a key. If you press ESCAPE the macro will stop, but single-step mode will remain active.

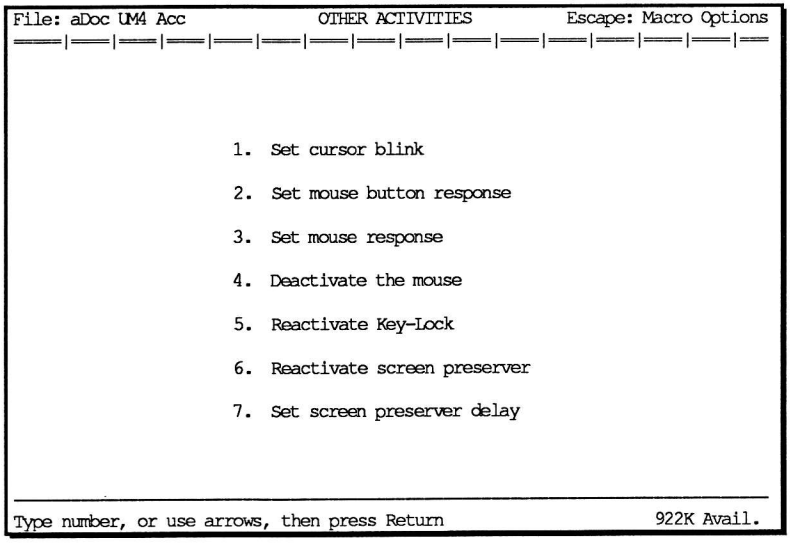
Select `Deactivate single-stepping` to return to normal speed.

6. Version

This option shows the current version of the UltraMacros package. You will need to know this number if you ever contact Beagle Bros with questions about UltraMacros.

7. Other Activities

Choose this to display the following menu:



Other Activities

The options on this menu allow you to make changes to how UltraMacros functions. When you exit this menu by pressing ESCAPE, you will be asked if you want to save any changes to your AppleWorks startup disk. If you answer `Yes`, the current settings will be in effect every time you start AppleWorks.

1. Set cursor blink

Choose this option to adjust the speed at which AppleWorks' cursor flashes. The current setting is displayed and you're prompted to enter a new value. Enter a number from 1-255 where 1 is the fastest blink rate available and 255 is the slowest.

2. Set mouse button response

Choose this option to adjust how long the mouse button delays after it's used to select menu options. If you find yourself jumping several menu steps at a time when you press the mouse button, you should increase the delay.

The current setting is displayed and you're prompted to enter a new value. Enter a number from 1-255 where 1 is the shortest delay available and 255 is the longest.

3. Set mouse response

Choose this option to adjust how far the mouse has to travel horizontally or vertically before the cursor moves.

The current horizontal setting is displayed and you're prompted to enter a new value. Enter a number from 1-255 where 1 is the most responsive and 255 is the least responsive. Press Escape when the desired number is entered.

The current vertical setting is displayed and the same procedure is followed to change it.

Apple IIgs users can also use the Control Panel Options to change the high speed mouse option to "yes" or "no".

4. Deactivate the mouse

This option makes UltraMacros ignore your mouse card. It is useful for IIc users who don't have a mouse. They do have mouse cards built-in, and the cards sometimes give false readings as though a mouse was being moved. If you experience sporadic random cursor moves on any kind of Apple II, give this option a try.

Select the option; the menu will now be redisplayed as `4. Reactivate the mouse`. Options 1 and 2 will now give a `Mouse not found` error if selected.

5. Reactivate Key-Lock

Key-Lock is a special feature designed primarily for physically disabled AppleWorks users who have a difficult time pressing the OPEN-APPLE or SOLID-APPLE keys in combination with other keys. This option makes the OPEN-APPLE and SOLID-APPLE keys "lock on" until another key is pressed.

Here's how it works. First, select the option. Because Key-Lock is now activated, the menu option will now be redisplayed as `5. Deactivate Key-Lock`.

Press Esc a few times to get back to AppleWorks. Get in a word processor document and press OPEN-APPLE. An inverse O appears in the bottom right corner of the screen. The cursor stops blinking and UltraMacros waits until you press another key.

If you press OPEN-APPLE again, the command will be cancelled and the normal cursor will appear. If you press another key, such `1`, the OPEN-APPLE version of the key will be entered in AppleWorks. In this case, the cursor will jump to the top of the document. Now press OPEN-APPLE again followed by `9` and you'll jump to the end.

To execute a macro, press the SOLID-APPLE key. An inverse `S` will appear. As before, you can press OPEN-APPLE to cancel, or another key to execute a command. To start a "BA" macro, press the OPEN-APPLE key followed by the SOLID-APPLE key. The inverse `O` will change to a `B` and the desired macro key can be pressed.

6. Reactivate screen preserver

The screen preserver will automatically blank your screen if there has been no key press or mouse move for a specified amount of time. This avoids monitor "burn-in" (i.e. scorching of the phosphors on the monitor screen).

Here's how to use it. First, select the option. Because the preserver is now activated, the menu option will now be redisplayed as `6. Deactivate screen preserver.`

Now sit and wait a few seconds. The screen will blank out, but don't despair. Just press a key to restore it. The screen preserver is tied to the blinking cursor; if an inverse bar menu such as OPEN-APPLE-C `Copy Text?` is active, the screen will never be blanked. The screen preserver is also ignored while the TimeOut OPEN-APPLE-ESCAPE menu is on the screen. It does work within TimeOut applications, though.

To cancel the screen preserver completely, just reselect it from the menu.

7. Set screen preserver delay

Choose this option to adjust how long the screen preserver waits before it blanks the screen. The delay is based on the number of cursor blinks, so your cursor blink rate directly affects it.

The current setting is displayed and you're prompted to enter a new value. Enter a number from 1-255 where 1 is the shortest delay and 255 is the longest.

Data Converter

The TimeOut Data Converter allows you to quickly and easily transfer data between spreadsheet and data base files.

To transfer data from a spreadsheet to a data base file, use the Open-Apple-C command to copy some spreadsheet rows to the clipboard. Press Open-Apple-Escape and select `Data Converter` from the TimeOut menu. The data on the clipboard will instantly be converted into data base data.

Create a new data base file or load one in from disk. Place the cursor where you would like to insert the spreadsheet data, press Open-Apple-C and select `From clipboard`. Each spreadsheet row will now be inserted into your data base file. Each column from the spreadsheet will become a data base category.

To transfer from a data base file to a spreadsheet file, copy from the data base file to the clipboard, call up the TimeOut menu, select `Data Converter`, and copy from the clipboard to the spreadsheet file. Each category from the data base will become a spreadsheet column.

If you're using the Applied Engineering expanded clipboard, you must use the TimeOut Utilities to properly configure the Data Converter.

Allowing Control-@

AppleWorks 2.0 contains a bug which prevents users from entering a CONTROL-@ in a printer or interface definition. Here's a fix:

1. Boot the UltraMacros disk and press `F` for `Fix AppleWorks Bug`.

2. The CONTROL-@ patch program is run. Follow the instructions.

Note: Use OPEN-APPLE-CONTROL-@ to enter the CONTROL-@ code while recording a macro. If you press CONTROL-@, the macro will stop. Likewise, a macro definition must use OPEN-APPLE-CONTROL-@ or it will stop.

Mouse Control

Once UltraMacros has been installed on your AppleWorks disk, you can use an Apple mouse to speed scrolling and menu selection. Here's how:

Move the mouse to position the cursor just like using the arrow keys. Hold down the OPEN-APPLE key to move the cursor farther and faster within a file.

Press the mouse button to do one of the following:

(a) select an option from any AppleWorks or TimeOut menu (the same as pressing the RETURN key)
(b) scroll quickly through an AppleWorks file (the direction--up or down--is the same as the last vertical mouse movement)

You can adjust mouse sensitivity or deactivate the mouse using the Macro Options TimeOut application. See page 68 for details.

Unless you specify otherwise, the mouse is always active (if you have one). If you have a IIe with a mouse interface card but no mouse is plugged in, you must deactivate the mouse; otherwise you

will probably be faced with an out-of-control cursor that jumps wildly when you least expect it. This can also happen on IIc's that don't have a mouse plugged in.

Mouse Tip: The mouse is especially handy for creating Data Base single record layouts. Just hold down the OPEN-APPLE key and use the mouse to drag categories into position.

Linking Files

You can use the <store> and <recall> commands to link AppleWorks files together. Linking allows you to do things like print an unlimited number of files with one keypress, or have a custom set of macros loaded automatically with any Spreadsheet file.

The <store> command saves the first 15 characters of macro 0 (zero), generally a name, in an unused area of a Word Processor or Spreadsheet file. The <recall> command puts the name back in macro 0 (zero).

Start up AppleWorks and insert the UltraMacros disk. Add the Word Processor file "Linking Samples" to the Desktop and study it for some sample linking macros and ideas.

Startup Menus

UltraMacros allows you to create handy menus which allow you to choose what you want AppleWorks to do next. You can use Task files (see next section), or create menus within AppleWorks files.

Start up AppleWorks and insert the UltraMacros disk. Add the Word Processor file *Macros Startup* to the Desktop and study it for a sample menu macro set.

Task Files

Task files are precompiled sets of macros that have been saved on the *AppleWorks STARTUP* disk as system files. They are called "Task" files because they allow you to quickly and easily execute a specific task.

New tasks can be launched several different ways:

1. From within AppleWorks by using the TimeOut Macro Options application (page 64).
2. From outside of AppleWorks by using a program selector such as Apple Desktop, MouseDesk, Alan Bird's Program Selector, ProSel, ECP or Squirt.
3. From outside of AppleWorks by typing `-TASK.NAME` from Basic with the *AppleWorks STARTUP* disk in the current drive.

To see a sample task file, start AppleWorks and insert the UltraMacros disk. Add the Word Processor file Task Sample to the Desktop.

When a task is launched from outside of AppleWorks, it first loads ULTRA.SYSTEM, which in turn loads AppleWorks. The first macro in the set of macros is then executed. The first macro from Task Sample looks like this:

```
]:<all : rtn : rtn :  sa-1>! get to main menu and run second macro
```

The macro enters the two RETURN keystrokes needed to get past the AppleWorks copyright message and accept the current date. It then jumps to the next macro in the set.

Note: See Task Sample for replacement first macros if you don't have a clock or if you need to pause to remove the AppleWorks STARTUP disk and insert the PROGRAM disk.

The second macro should actually begin the task. This is the macro that's executed when the task is launched from within AppleWorks. The sample second macro looks like this:

1:<all oa-q esc rtn down down rtn rtn>Rename Me<rtn>!

It uses the standard <oa-q esc> sequence to jump to the AppleWorks Main Menu. This is a good technique to make sure your macros work the same no matter where you are when the SOLID-APPLE key is pressed. The macro then executes the necessary keystrokes to add an new Word Processor file to the Desktop called "Rename Me".

This example is simple, but it illustrates the potential of Task files. Here are some possible Task file uses:

1. Add a specific group of files to the AppleWorks Desktop.
2. Copy a 3.5" disk to a RAM disk using TimeOut FileMaster.
3. Load a Spreadsheet file and graph it with TimeOut Graph.
4. Load a Word Processor file and print it with TimeOut SuperFonts.
5. Any complex and repetitive task, such as printing out a weekly report using Data Base or Spreadsheet data.

It's a good idea to create a "Default Macros" task file first and save it on your AppleWorks STARTUP disk (see page 65). Then when a task is completed, you can quickly get back to your default set of macros. In fact, a task file can even do that automatically after it's completed its task.

> *NOTE: Programmers interested in licensing a special type of task files for use on disks they sell should read the file "ProgrammerTasks" on the UltraMacros disk.*

The Special Case of Macro 0 (zero)

Macro 0 (zero) is a unique macro because it's used by many special UltraMacros commands. Use macro 0 (zero) only for very temporary macros, because it can be redefined quite easily before you know it. Here's a list of ways that macro 0 (zero) can be redefined:

1. Press OPEN-APPLE-X to record up to 80 characters; the keystrokes are passed on to AppleWorks as they're entered.
2. Press OPEN-APPLE-0 to enter up to 60 characters; the keystrokes are not passed on to AppleWorks (page 34).
3. Press OPEN-APPLE-^ to read the character under the cursor into macro 0 (page 36).
4. Press OPEN-APPLE-& to read the current disk name into macro 0 (page 36).
5. Press OPEN-APPLE-* to read the current path into macro 0 (page 36).
6. Press OPEN-APPLE-- to read the current Data Base category, Spreadsheet cell or Word Processor line into macro 0 (page 36).
7. Press OPEN-APPLE-> to recall the STORE'd file name into macro 0 (page 37).
8. Use <$0 = ...> to define it from within a macro (page 42).

Once macro 0 (zero) has been defined, the keystrokes may be sent to AppleWorks by pressing SOLID-APPLE-0. A handy example is copying a data base category and pasting it into the Word Processor, into the Spreadsheet, or into another Data Base category.

1. Press OPEN-APPLE-- while in the Data Base. The current category is read into macro 0 (zero).
2. Move to the new location in the Data Base or anywhere else.
3. Press SOLID-APPLE-0 and the category is duplicated.

The contents of macro 0 (zero) are also used by the SOLID-APPLE-RETURN <find> command (page 32) to determine what it looks for. If you have a lot of files on a disk, try this:

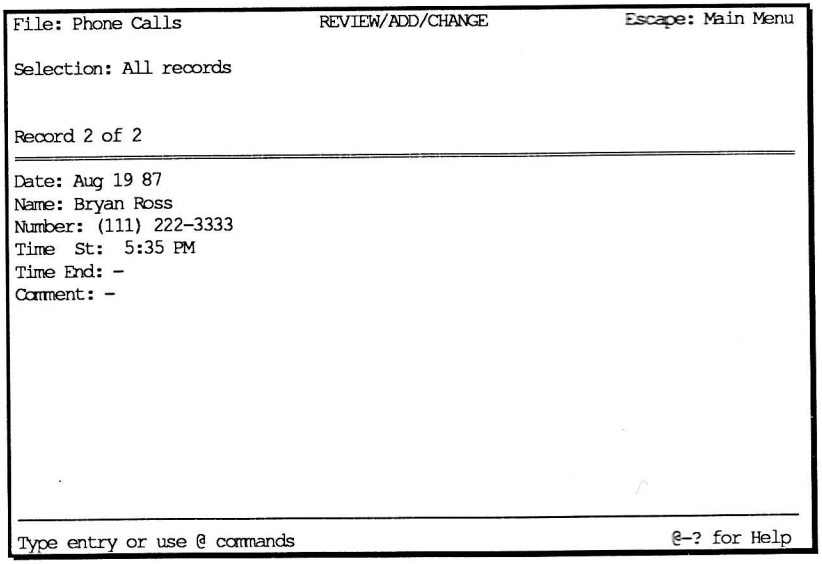
1. Go to the `Add files` menu.
2. Press OPEN-APPLE-0. You will be prompted with a >.
3. Enter the file name you wish to find and press Return.
4. Press SOLID-APPLE-RETURN. The cursor will jump to the specified name or stop at the end of the file list if it couldn't find a match.
5. Press Return to load the file.

Examine the Spreadsheet file called Macro 0 Memo on the UltraMacros disk for a handy chart showing the possible macro 0 (zero) uses.

A Macro Explained

Here's some information about a handy macro that uses just a few of UltraMacros' powerful commands. Hopefully it will give you a better understanding of how UltraMacros' commands can be utilized.

1. Start AppleWorks and insert the UltraMacros disk in a drive.
2. Add the Data Base files *Addresses* and *Phone Calls* to the Desktop, along with the *Macros Ultra* Word Processor file.
3. Select `Macros Ultra` and compile it (page 59). Then use OPEN-APPLE-F to find `Phone Calls`.
4. You'll see a detailed macro description. Examine it briefly.
5. Switch to the "Addresses" file and put the cursor on a name. Press SOLID-APPLE-CONTROL-P. In an instant you'll see a screen something like this:



All you need to do is press SOLID-APPLE-= when the conversation is over (to enter the time), and you've got a detailed record of the call. How'd the macro do it? Examine it step-by-step. The curly {brackets} allow you to imbed comments between the <token

brackets>. These comments are not stored in the macro table, so they don't waste any space.

TimeOut MacroTools

MacroTools is a TimeOut disk full of sample macros, TimeOut tips, special TimeOut applications, and a surprise or two. UltraMacros beginners will appreciate the powerful ready-to-use macros and instructional macro tips.

Veteran macro maniacs will love the programming ideas and samples on the disk. They'll especially like "Debug", a TimeOut application that displays all kinds of useful macro table information that only a macro programmer could enjoy. It even allows variables to be examined and modified!

Check the NOTES file on the UltraMacros disk for more info about MacroTools.

Changes from Super MacroWorks

Here is a brief summary of changes for users converting from Super MacroWorks to UltraMacros. Several obvious differences relate to TimeOut. All compiling, saving of compiled macros, macro listing, changing mouse options, etc., is done using the TimeOut applications. The resulting saving of memory space allows the actual UltraMacros program (ULTRA.SYSTEM) to be more powerful.

The following new tokens have been added UltraMacros:

asc, ato, begin, call, cell, clear, else, elseoff, getstr, goto, hilight, id#, left $, nosleep, onerr, peek, poke, posn, print, pr#, rem, right $, screen, then, time24, val, wait, wake

A significant improvement in an existing command is the new ability of <find> to search any numbered highlighted bar menu. It

works with the OPEN-APPLE-Q Desktop Index, the OPEN-APPLE-ESCAPE TimeOut menu and with any normal menu such as the list of printers you're given before printing a file.

The following tokens are not available. They are unnecessary because their functions can be duplicated with existing commands:

end?, list, menu, resume, swap, OPEN-APPLE-#, OPEN-APPLE-$, Lprint, compile, save0, load0, 0=, if0, var=, incvar, decvar, varnot, var

The "var" commands have been replaced with a full set of 26 numeric variables. The other commands can be replaced with the new UltraMacros commands as described in the file *Macros From SMW* on the UltraMacros disk.

Macro 0 (zero) has been expanded to 80 keystrokes, and it can be defined in a few more ways.

The msg token is more powerful now (allowing date/time, strings), but no longer has the ability to do MouseText characters. You'll have to settle for normal and inverse text. Notice that it uses the "correct" line in the Spreadsheet now.

The ability to automatically download a Power Print font at startup time was dropped.

> *NOTE: You may leave Super MacroWorks on the AppleWorks STARTUP disk. The UltraMacros installation program will rename SUPER.SYSTEM to SUPER.SYSOLD.*

To use UltraMacros, you can then boot your AppleWorks STARTUP disk, choose "ULTRA.SYSTEM" with a program selector, or type "-ULTRA.SYSTEM" from Basic.

To use Super MacroWorks, type "-SUPER.SYSOLD" from Applesoft Basic or choose "SUPER.SYSOLD" with a program selector.

Macro Token List

Here's a complete alphabetical listing of all UltraMacros tokens, along with the page number where a full description can be found.

Token	Page	Summary
adb	28	AppleWorks application code
ahead	31	find next space
all	28	AppleWorks application code
asc	45	convert a string character to ASCII
asp	28	AppleWorks application code
ato	28	TimeOut application code
awp	28	AppleWorks application code
ba-	19, 27	both-apple macro name
back	31	find previous space
begin	40	start of repeatable section (see rpt)
bell	38	sound the AppleWorks error bell
call	58	execute a machine language subroutine
cell	36	read current cell or category to macro 0 (zero)
chr$	45	return ASCII equivalent of a variable
clear	38	set all numeric and string variables to 0
ctrl-	27	control key
date	31	display date in this format: August 19, 1987
date2	31	display date in this format: 08/19/87
dec	37	decrement the cursor character
del	27	delete key
disk	36	read disk name to macro 0 (zero)
down	27	down-arrow key
else	56	reverses the conditional status of a macro
elseoff	57	return macro to being unconditional
esc	27	escape key
find	32	find a name or carriage return
findpo	33	find a printer option
getstr	45	define a string from the keyboard
goto	45	jump to specified macro
hilight	46	invert specified screen portion
id#	39	returns TimeOut application number
if	54	execute macro if condition is true
ifkey	39	checks if specified key was pressed
ifnot	55	execute macro if condition is not true

works with the OPEN-APPLE-Q Desktop Index, the OPEN-APPLE-ESCAPE TimeOut menu and with any normal menu such as the list of printers you're given before printing a file.

The following tokens are not available. They are unnecessary because their functions can be duplicated with existing commands:

end?, list, menu, resume, swap, OPEN-APPLE-#, OPEN-APPLE-$, Lprint, compile, save0, load0, 0=, if0, var=, incvar, decvar, varnot, var

The "var" commands have been replaced with a full set of 26 numeric variables. The other commands can be replaced with the new UltraMacros commands as described in the file *Macros From SMW* on the UltraMacros disk.

Macro 0 (zero) has been expanded to 80 keystrokes, and it can be defined in a few more ways.

The msg token is more powerful now (allowing date/time, strings), but no longer has the ability to do MouseText characters. You'll have to settle for normal and inverse text. Notice that it uses the "correct" line in the Spreadsheet now.

The ability to automatically download a Power Print font at startup time was dropped.

> *NOTE: You may leave Super MacroWorks on the AppleWorks STARTUP disk. The UltraMacros installation program will rename SUPER.SYSTEM to SUPER.SYSOLD.*

To use UltraMacros, you can then boot your AppleWorks STARTUP disk, choose "ULTRA.SYSTEM" with a program selector, or type "-ULTRA.SYSTEM" from Basic.

To use Super MacroWorks, type "-SUPER.SYSOLD" from Applesoft Basic or choose "SUPER.SYSOLD" with a program selector.

Macro Token List

Here's a complete alphabetical listing of all UltraMacros tokens, along with the page number where a full description can be found.

Token	Page	Summary
adb	28	AppleWorks application code
ahead	31	find next space
all	28	AppleWorks application code
asc	45	convert a string character to ASCII
asp	28	AppleWorks application code
ato	28	TimeOut application code
awp	28	AppleWorks application code
ba-	19, 27	both-apple macro name
back	31	find previous space
begin	40	start of repeatable section (see rpt)
bell	38	sound the AppleWorks error bell
call	58	execute a machine language subroutine
cell	36	read current cell or category to macro 0 (zero)
chr$	45	return ASCII equivalent of a variable
clear	38	set all numeric and string variables to 0
ctrl-	27	control key
date	31	display date in this format: August 19, 1987
date2	31	display date in this format: 08/19/87
dec	37	decrement the cursor character
del	27	delete key
disk	36	read disk name to macro 0 (zero)
down	27	down-arrow key
else	56	reverses the conditional status of a macro
elseoff	57	return macro to being unconditional
esc	27	escape key
find	32	find a name or carriage return
findpo	33	find a printer option
getstr	45	define a string from the keyboard
goto	45	jump to specified macro
hilight	46	invert specified screen portion
id#	39	returns TimeOut application number
if	54	execute macro if condition is true
ifkey	39	checks if specified key was pressed
ifnot	55	execute macro if condition is not true

works with the OPEN-APPLE-Q Desktop Index, the OPEN-APPLE-ESCAPE TimeOut menu and with any normal menu such as the list of printers you're given before printing a file.

The following tokens are not available. They are unnecessary because their functions can be duplicated with existing commands:

end?, list, menu, resume, swap, OPEN-APPLE-#, OPEN-APPLE-$, Lprint, compile, save0, load0, 0=, if0, var=, incvar, decvar, varnot, var

The "var" commands have been replaced with a full set of 26 numeric variables. The other commands can be replaced with the new UltraMacros commands as described in the file *Macros From SMW* on the UltraMacros disk.

Macro 0 (zero) has been expanded to 80 keystrokes, and it can be defined in a few more ways.

The msg token is more powerful now (allowing date/time, strings), but no longer has the ability to do MouseText characters. You'll have to settle for normal and inverse text. Notice that it uses the "correct" line in the Spreadsheet now.

The ability to automatically download a Power Print font at startup time was dropped.

> *NOTE: You may leave Super MacroWorks on the AppleWorks STARTUP disk. The UltraMacros installation program will rename SUPER.SYSTEM to SUPER.SYSOLD.*

To use UltraMacros, you can then boot your AppleWorks STARTUP disk, choose "ULTRA.SYSTEM" with a program selector, or type "-ULTRA.SYSTEM" from Basic.

To use Super MacroWorks, type "-SUPER.SYSOLD" from Applesoft Basic or choose "SUPER.SYSOLD" with a program selector.

Macro Token List

Here's a complete alphabetical listing of all UltraMacros tokens, along with the page number where a full description can be found.

Token	Page	Summary
adb	28	AppleWorks application code
ahead	31	find next space
all	28	AppleWorks application code
asc	45	convert a string character to ASCII
asp	28	AppleWorks application code
ato	28	TimeOut application code
awp	28	AppleWorks application code
ba-	19, 27	both-apple macro name
back	31	find previous space
begin	40	start of repeatable section (see rpt)
bell	38	sound the AppleWorks error bell
call	58	execute a machine language subroutine
cell	36	read current cell or category to macro 0 (zero)
chr$	45	return ASCII equivalent of a variable
clear	38	set all numeric and string variables to 0
ctrl-	27	control key
date	31	display date in this format: August 19, 1987
date2	31	display date in this format: 08/19/87
dec	37	decrement the cursor character
del	27	delete key
disk	36	read disk name to macro 0 (zero)
down	27	down-arrow key
else	56	reverses the conditional status of a macro
elseoff	57	return macro to being unconditional
esc	27	escape key
find	32	find a name or carriage return
findpo	33	find a printer option
getstr	45	define a string from the keyboard
goto	45	jump to specified macro
hilight	46	invert specified screen portion
id#	39	returns TimeOut application number
if	54	execute macro if condition is true
ifkey	39	checks if specified key was pressed
ifnot	55	execute macro if condition is not true

inc	37	increment cursor character
input	38	accept keystrokes until return is pressed
insert	35	force the insert cursor on
key	39	wait for keypress and return its value
lc	35	force cursor character to lower case
left	27	left-arrow key
left	46	returns the left part of a string
len	46	return length of a string
msg	46	display message on screen
nosleep	38	cancel sleeping macro
oa-	27, 34	open-apple key
onerr	47	set error handling conditions
path	36	read disk and file name to macro 0 (zero)
peek	59	return value of a memory address
poke	58	store data in a memory address
posn	48	returns cursor position
print	34, 49	print variables
pr#	48	send print output to screen or printer
read	36	read character under cursor to macro
recall	37	recall store'd text to macro 0 (zero)
rem	50	remark within a macro
right	27	right-arrow key
right	51	returns the right part of a string
rpt	40	repeat the current macro
rtn	27	return key
sa-	27	solid-apple key
screen	51	read screen portion into macro 0 (zero)
spc	27	space bar key
stop	40	unconditionally stop all macro activity
store	37	store macro 0 (zero) in an Awp or Asp file
str$	52	return string equivalent of a variable
tab	27	tab key
then	56	filler token (does nothing but look pretty)
time	32	display time in this format: 7:28 pm
time24	32	display time in this format: 19:28
up	27	up-arrow key
uc	35	force cursor character to upper case
val	52	return numeric value of a string
wait	52	wait until keypress or time is up
wake at	53	set sleeping macro
zoom	35	force zoom out

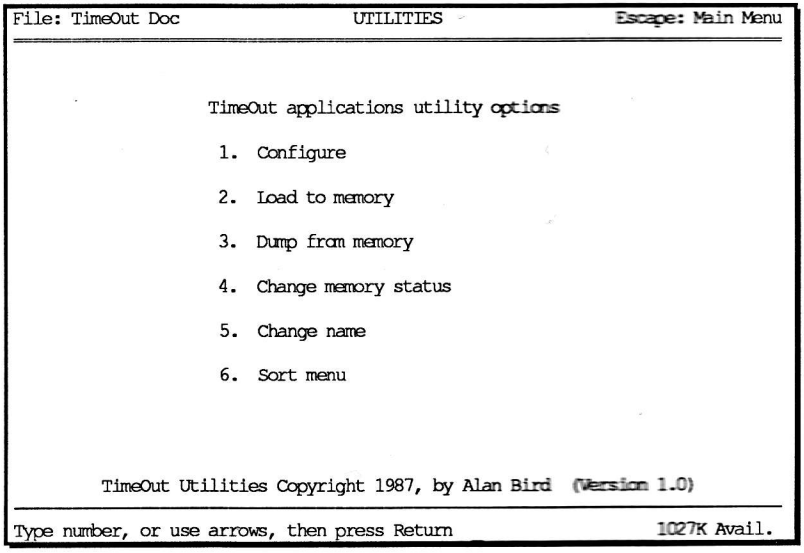
Appendix

TimeOut Utilities

The TimeOut Utilities application is provided with all TimeOut products. It has several functions that give you a lot more flexibility in using your TimeOut applications.

Using the Utilities

To use the TimeOut Utilities, make sure that the file *TO.UTILITIES* has been copied to your TimeOut applications disk. Start up AppleWorks and press Open-Apple-Escape to call up the TimeOut menu. Select `Utilities`. You will see the following screen:



Configure

The `Configure` option allows you to set new defaults for your TimeOut applications. Configurable options might include printer type, default font, location of files needed by the application, etc. Not all TimeOut applications have configurable options.

To configure an application, select `Configure` from the Utilities menu. Then select the application you want to configure. You will then see a menu indicating what options may be configured for that particular application. You should also see the current value for each option in brackets [ ].

Select an option that you would like to change. Enter or select the new value for that option. Make sure that the TimeOut applications disk is in a drive so that the application can be updated with the new value. The next time you use the application, it will use the new value that you have supplied.

After you are finished updating configurable options, press Escape to return to the Utilities main menu.

Load to memory

TimeOut applications are either disk-resident or memory-resident. If an application was configured as disk-resident when you started up AppleWorks, you can load it into memory using the `Load to memory` option. Just select the option from the Utilities menu and select which application you would like to load.

Dump from memory

If you receive a message from AppleWorks indicating that it was unable to complete an option because of insufficient desktop memory, you may need to dump one or more TimeOut applications that are memory-resident. Select `Dump from memory` from the Utilities menu and select which application you would like to dump. Notice that the amount of free memory indicated in the lower right hand portion of the screen increases with each application you dump. Applications that are *dumped* are returned to disk-resident status for the remainder of the AppleWorks session.

Change memory status

This option allows you to indicate whether a TimeOut application is disk- or memory-resident. Note that this only indicates how the application will be treated when you start up AppleWorks. To load an application into memory or to return it to the disk for the current AppleWorks session, you will need to use the `Load to memory` option or the `Dump from memory` option.

Change name

This option allows you to change the name of the application as it appears in the TimeOut menu. The Beagle Bros staff carefully selects a good name for each application. However, you have the flexibility of renaming it if you wish.

If the new name you enter is longer than the old name, the name change will not be reflected in the TimeOut menu until the next time you start up AppleWorks.

Sort Menu

When you apply TimeOut to your *AppleWorks STARTUP* disk, you are given the option of indicating whether or not you want the TimeOut menu automatically sorted by application name. If you choose not to have the menu sorted, you can still sort it after starting up AppleWorks by selecting `Sort menu` from the Utilities menu.

Key Chart

	Low		High	Low		High	Low		High	Low		High
	0	ctrl-@	128	32	spc	160	64	@	192	96	`	224
	1	ctrl-A	129	33	!	161	65	A	193	97	a	225
	2	ctrl-B	130	34	"	162	66	B	194	98	b	226
	3	ctrl-C	131	35	#	163	67	C	195	99	c	227
	4	ctrl-D	132	36	$	164	68	D	196	100	d	228
	5	ctrl-E	133	37	%	165	69	E	197	101	e	229
	6	ctrl-F	134	38	&	166	70	F	198	102	f	230
	7	ctrl-G	135	39	'	167	71	G	199	103	g	231
left	8	ctrl-H	136	40	(	168	72	H	200	104	h	232
tab	9	ctrl-I	137	41	)	169	73	I	201	105	i	233
down	10	ctrl-J	138	42	*	170	74	J	202	106	j	234
up	11	ctrl-K	139	43	+	171	75	K	203	107	k	235
	12	ctrl-L	140	44	,	172	76	L	204	108	l	236
return	13	ctrl-M	141	45	-	173	77	M	205	109	m	237
	14	ctrl-N	142	46	.	174	78	N	206	110	n	238
	15	ctrl-O	143	47	/	175	79	O	207	111	o	239
	16	ctrl-P	144	48	0	176	80	P	208	112	p	240
	17	ctrl-Q	145	49	1	177	81	Q	209	113	q	241
	18	ctrl-R	146	50	2	178	82	R	210	114	r	242
	19	ctrl-S	147	51	3	179	83	S	211	115	s	243
	20	ctrl-T	148	52	4	180	84	T	212	116	t	244
right	21	ctrl-U	149	53	5	181	85	U	213	117	u	245
	22	ctrl-V	150	54	6	182	86	V	214	118	v	246
	23	ctrl-W	151	55	7	183	87	W	215	119	w	247
	24	ctrl-X	152	56	8	184	88	X	216	120	x	248
	25	ctrl-Y	153	57	9	185	89	Y	217	121	y	249
	26	ctrl-Z	154	58	:	186	90	Z	218	122	z	250
escape	27	ctrl-[	155	59	;	187	91	[	219	123	{	251
	28	ctrl-\	156	60	<	188	92	\	220	124	|	252
	29	ctrl-]	157	61	=	189	93	]	221	125	}	253
	30	ctrl-^	158	62	>	190	94	^	222	126	~	254
	31	ctrl-_	159	63	?	191	95	_	223	127	del	255

Customer Support Information

If you have questions or problems that your dealer can't answer, you can contact the Beagle Bros Technical Support Staff for expert assistance.

Before calling, check the instruction manual to see if it contains the information you need. Write down a complete description of the problem, the version number of the software, and the names and version numbers of any other AppleWorks enhancement programs you're using.

If you have a modem, you may also receive Tech Support on our 24-hour Customer Support System. The system provides an electronic mail and conferencing system, along with the latest information about product updates and changes.

Technical Support:
(619) 452-5502 9 am to 5 pm, weekdays (pacific time)

Modem Tech Support:
(619) 452-5565 24 hours, everyday

Or, you can write to:

Beagle Bros, Inc.
6215 Ferris Square, Suite 100
San Diego, CA 92121


Help!

Index

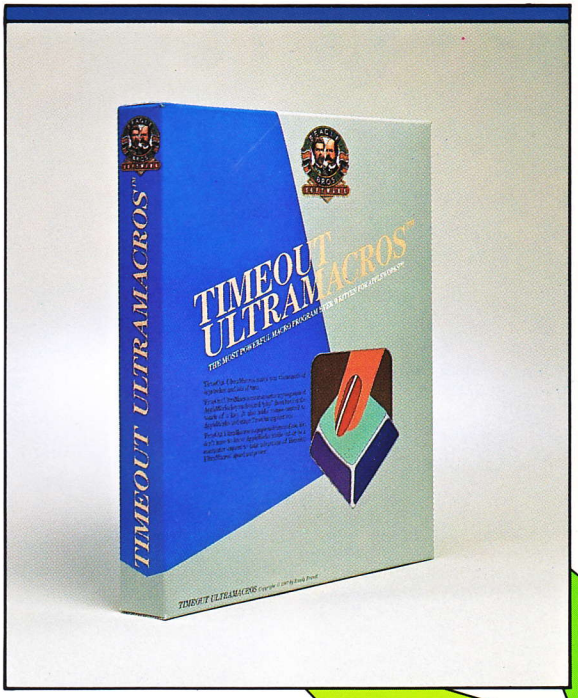
Accessing TimeOut 12
Apple IIgs 2
AppleWorks Startup disk 6
Built-in macros 20
Calling other macros 30
Change memory status 86
Change name 86
Compatibility 6
Compiler 59
Configure 84
Control-@ bug fix 3, 72
Control-Reset 13
Copying applications 9
Creating a macro 21
Cursor 67
Data converter 71
Deactivate 66, 68
Desktop expanders 7
Display macros 63
Dump from memory 85
Errors 60
Global macros 28
Help 89
Installation 6, 7, 10
Key Chart 87
Key-lock 69
Linking files 73
List macros 63
Load to memory 85
Local macros 28
Logic 54
Macro 0 (zero) 76
Macro Compiler 59

Macro Options 64
Macro table 65
Macro token list 81
MacroTools 79
Memory resident 13
Memory usage 13
Mouse 3, 68, 72
Nesting 30
New Open-Apple commands 34
Notes 7
Numeric variables 41
Open-Apple key 2
Option key 2
Options 64
Other Activities 67
Parameters 40, 43
Re-installing 9
Reactivate 66, 69
Recording a macro 17
Reserved macros 31
Screen Preserver 70
Single stepping 66
Sleep 53
Solid-Apple key 2
Sort menu 86
Starting up 11
Startup menus 73
String variables 42
SuperMacroWorks 79
Task 64, 74
Technical support 89
TimeOut applications 8
TimeOut menu 8
TO.UTILITIES 84
Token list 81
Tokens 26, 38
Troubleshooting 89
ULTRA.SYSTEM 6
Utilities 84
Version 6, 66

BEAGLE BROS, INC. • 6215 Ferris Square, Suite 100 • San Diego, CA 92121 • (619) 296-6400

TimeOut UltraMacros from Beagle Bros is the indispensable productivity tool every AppleWorks owner needs.

No other single program has the flexibility and power of UltraMacros. Here are just a few of the exciting features UltraMacros adds to AppleWorks®...



• ***Mouse Control:*** Use the mouse to select menu items and zip through your Word Processor, Spreadsheet and Data Base files.

• ***Screen Saver:*** UltraMacros can blank the screen when you stop typing. This will add years to the life of your monitor!

• ***Set the Cursor Rate:*** Customize the speed the cursor blinks, from fast to not at all.

• ***Key Lock:*** UltraMacros allows the Open-Apple and Option keys to be pressed and released independently of other keys — a great benefit for the handicapped user.

• ***Macros:*** The greatest invention since the wheel!

Just what is a Macro?

A macro is a pre-defined series of keystrokes that you can replay by pressing just one key. Normally, when you press a key, you get just one key's worth of input. Macros can save you countless hours of typing.

Here's how UltraMacros can save you time writing letters.

Since you start your letters the same way every time, UltraMacros can automate the process. First, press Open-Apple-X to start recording a macro. Then, select the key you would like to use to activate this macro. Let's choose "L" for letter.

Begin typing as you usually do. Enter the date, address and any other information you want. To end the macro recording, press Shift-Control-@. Congratulations! You will never have to type that information again! Next time, just press Option-L. Your text will be typed automatically — in a fraction of a second.

You can have over 180 recorded macros available at one time,

with up to 4,000 total keystrokes.

The macros you create can be saved together on disk, all ready to go to work next time you start up AppleWorks.

Additional macro sets can be created and saved on disk. You can switch macro sets any time — right in the middle of AppleWorks —so there is really no limit to the number of macros you can have.

UltraMacros is not limited to the Word Processor.

Your macros can work anywhere in AppleWorks: the Word Processor, Data Base, Spreadsheet, other TimeOut applications, and in between. For example, it takes at least six keystrokes to add a file to the desktop. Record a macro to do it in one!

One of the most impressive aspects of UltraMacros is its programmability.

UltraMacros includes a complete programming language that gives you absolute control over AppleWorks. Over 75 commands let you move the cursor, input and print strings anywhere, read the screen, find data in a file, read a menu, and much, much more.

By the way, don't worry if this programming stuff doesn't make sense. It's not required.

Full IF-THEN-ELSE logic with string and numeric variables is supported. Program UltraMacros to read the clock and activate macros at preset times. You can PEEK and POKE memory locations, or even CALL your own machine language routines.

UltraMacros compiles your macro file inside AppleWorks, creating macros that are ready to use. With hundreds of sample macros on the disk, it's easy to get started.

MacroTools and MacroTools II are collections of special tools and ready-made macro files for UltraMacros.

MacroTools lets you do all this and more:

- Choose activities from your own custom menus.
- Change the pitch and duration of the AppleWorks bell.
- Eject 3.5 inch disks from the drive.
- Auto-save files at preset intervals.
- Add a clock, mini calculator and file encrypter.

MacroTools II lets you do even more:

- Print Word Processor files in 2 or 3 columns.
- Simulate 60 to 90 categories in the Data Base.
- Reset the date AppleWorks uses.
- Send printer codes directly to the printer.
- Turn AppleWorks into a typewriter.

Other great TimeOut Applications:

DeskTools	$49.95
DeskTools II	$49.95
FileMaster	$49.95
Graph	$89.95
MacroTools	$25.00
MacroTools II	$25.00
PowerPack	$49.95
ReportWriter	$79.95
SideSpread	$49.95
SpreadTools	$59.95
SuperFonts	$69.95
TeleComm	$69.95
Thesaurus	$49.95
UltraMacros	$59.95

TimeOut UltraMacros is just one in a long line of time-saving, productivity software that AppleWorks users have depended on for over two years.

Please detach and send with payment.

50% Off

TIMEOUT ULTRAMACROS. (Now only $29.95!)

Send Coupon plus $3.50 shipping and handling with check, MasterCard, Visa or money order to: **Beagle Bros, Inc., 6215 Ferris Square, Suite 100, San Diego, CA 92121.**
Offer Expires Dec 31, 1989. Not valid with any other coupon.

Name _____

Address _____ City _____ State _____ Zip _____

Card # _____ Exp. _____ Signature _____

This offer valid for TimeOut UltraMacros only. Offer does not apply toward any other TimeOut purchase.
Claris Corporation has no responsibility for the fulfillment by Beagle Bros of its obligations with respect to this offer.
AppleWorks is a registered trademark of Apple Computer, Inc. licensed to Claris Corporation.
In California add 7% sales tax.