MachV

COLLABORATORS			
	TITLE : MachV		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		November 23, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

MachV

1.1 MachV.guide

Documetation for MachV and SetMachV - Release 5.0 Version 37.9

October 22, 1994

1.2 MachV.guide/Introduction

```
Introduction
************
```

   MachV is a multi-function utility program that provides many mouse enhancements as well as hotkeys for both shell and Workbench users. While there are other programs that do some of these, none provide the special functions or let you set up different settings for each window, screen or program. If fact, the real power of MachV is in its special functions. Macros can be written using these functions to do any mouse action from the keyboard, run programs at a desired time, or interface with ARexx.

   A macro may have a gadget in the clock which will execute the macro when clicked on. The gadget's image may be created with an icon editing tool like IconEdit, or it may display the name of the macro.

   Keystrokes and mouse movements may be recorded and played back at the touch of a key or the click of a gadget.

   Macros may be split into multiple parts with each part functioning when its turn comes.

   You can quickly view the contents of the clipboard. The contents may be pasted (even to windows that do not support the clipboard) or printed. Files may be loaded directly to the clipboard or saved from the clipboard.

   SetMachV is a separate program that is used to change all options, edit macros and create alarms.

   Both MachV and SetMachV require Workbench 2.04 or higher.

   Both programs will use the users preferred language when run under Workbench 2.1 or higher.

   See Preferred Language.

   IMPORTANT
 1. All hotkeys referred to in this documentation are included in the sample MachV.prefs file. In order to do the examples, MachV needs to load this settings file. You may change the hotkeys later.

 2. MachV.020 has been optimized for the 68020 and 68030 processor chips. It will not run on a 68000 or 68010. MachV.020 available only to registered users.

 3. Some later versions of amigaguide.library will not display a backslash (\) unless it is doubled (\). If you are using a version that does this, you must use an editor to replace all backslashes with 2 backslashes.

   Sample Hotkeys
   Terms Used

```
   String Matching
   Starting MachV
   Terminating MachV
   Preferred Language
```

1.3 MachV.guide/Sample Hotkeys

```
Sample Hotkeys
==============

   These hotkeys are in the MachV.prefs file included on the disk. They
are required in order to perform the examples in this documentation.

'Left-Amiga-J'
     Window to front. ('J'ump to front)

'Right-Amiga-K'
     Window to back.  ('K'ick back)

'Left-Amiga-N'
     Screen to front and activate window.

'Right-Amiga-M'
     Screen to back and activate window.

'Left-Amiga-ESC'
     Create a new shell.

'Alt-ESC'
     Start or stop macro recording.

'Ctrl-Alt-ESC'
     Start or stop named macro recording.

'Ctrl-Left-Amiga-ESC'
     Run SetMachV.

'Ctrl-Left-Amiga-B'
     Blank the screen.

'Ctrl-Left-Amiga-N'
     Execute a named macro.

'Ctrl-Left-Amiga-C'
     Toggle the clock on/off.

'Ctrl-Left-Amiga-V'
     View the clipboard.

'Ctrl-Left-Amiga-Q'
     Send a close gadget message.

'Alt-Help'
     "Quick Edit" a macro.
```

1.4 MachV.guide/Terms Used

```
Terms Used
==========
```

All keys are enclosed in <>. <Ctrl>, <Ctrl-ESC>, <Alt-Shift-F1> etc.

* "Qualifiers" mean the keys <Ctrl>, <Shift>, <Alt>, <Left-Amiga>
  (C=), and <Right-Amiga>.

* "Hotkey" means any key pressed in combination with zero or more
  qualifiers. A hotkey has a macro attached.

* "Macro" is a quoted string of characters that are played back as
  keystrokes, special functions that are to perform some particular
  action, or a combination of these. A macro may be attached to a
  hotkey or to no key at all. A single key may have up to 32 macros
  attached to it, each with a unique qualifier combination. For
  example, <F1> may have a macro and so may <Ctrl-F1>,
  <Ctrl-Alt-F1>, <Shift-F1> etc.

  A macro that is not attached to a hotkey is usually executed by
  its name, by clicking on its gadget in the clock window, or
  automatically executed at some specified time.

* "Macro Name" indicates a descriptive name for a macro. It may be
  up to 14 characters long. Named macros may be executed by their
  name. The name may also be used as its gadget text.

  Some names have special significance. A macro named "Blank" will
  be executed when screen blanking would otherwise occur. "Unblank"
  will be executed if a "Blank" macro was executed and the screen
  should be unblanked.

  A macro name "MachVQuit" will execute when MachV is terminated.

  If you have multiple configurations, a macro whose name matches the
  configuration name will be executed when that configuration
  becomes active.

* "CLI and Shell" are used interchangeably. They mean the system
  window that you use to issue commands like: dir, info, list, copy,
  date etc.

1.5 MachV.guide/String Matching

In most cases when attempting to match title bar names, macro names
or configuration titles, the case is insignificant. Also, if no wild
cards are used, matching stops when the source string ends. For example:

```
        workb matches Workbench
        workb#? matches Workbench
        workbench matches Workbench
        work?e  does not match Workbench (wild card used, full string must match)
```

    Exact matches are required for certain macro names. These are:

    Blank, Unblank, MachVQuit and the joystick directions: JoyUp,
JoyUpRight, JoyRight, JoyDownRight, JoyDown, JoyDownLeft, JoyLeft,
JoyUpLeft, JoyFire1, JoyFire2 and JoyFire3.

1.6 MachV.guide/Starting MachV

```
Starting MachV
==============
```

    You can start MachV by double clicking on its icon, by putting it in
the WBStartup drawer, by entering in your s:User-Startup "Run >nil:
<nil: MachV" or typing from the shell the same command.

    You may follow the command with the settings file to load. For
example:

    Run >nil: <nil: MachV ENVARC:MachV/MachV.prefs

    or

    Run >nil: <nil: MachV Sample.prefs

    The ">nil: <nil:" prevents the startup message which displays your
registered named and serial number and allows the shell window to close.

    In your s:User-Startup file you may use something like this after
your path command:

    Stack 10000

    Run >nil: <nil: MachV

    When you run MachV with no file name to load as a settings file, it
automatically attempts to load MachV.prefs from the same directory that
MachV was run from. Failing that, it tries to load:

    ENVARC:MachV/MachV.prefs.

    ENVARC: is the assigned named for sys:Prefs/Env-Archive. This is the
directory that many programs save their preferences.

    If MachV still cannot locate the file, it will use some default
setting, but there WILL BE NO MACROS! Use SetMachV to load MachV.prefs
using the menu selection Project/Open...

    You may also specify where SetMachV is to be found. Use SETMACH= or

```
SM= followed by the full path and name.

   Run MachV SETMACH=sys:utils/SetMachV

   The settings file and the path for SetMachV may also be entered in
the Tool Types gadget in the icon.  Use:

   SETTINGS=path/filename.prefs    SETMACH=path/SetMachV

   After MachV is running, the only obvious indication of the presence
of MachV is the (optional) title bar clock. The real power of MachV is
lying in the background waiting to assist you. Exactly what it can do
will depend on your settings file.

   In order to start a new shell with a current directory other than
MachV's, enter in your s:shell-startup file the command "cd Work:" or
whatever is the directory that you wish to be current.

   *Note:*

   You may put MachV in the WBStartup drawer, but do not put SetMachV
there as it will be automatically run when you reset the computer.
```

1.7 MachV.guide/Terminating MachV

```
Terminating MachV
=================

   There are several ways to remove MachV.

  1. Activate the clock and press <Ctrl-C>.

  2. Double click on its icon.

  3. Type MachV in a shell.

  4. Issue a "Break C" command.

  5. Execute a MSC_QUIT macro.
```

1.8 MachV.guide/Preferred Language

```
Preferred Language
==================

   MachV and SetMachV can use languages other than English if run under
AmigaDOS 2.1 or higher. The user selects his preferred language using
the "Prefs/locale" program provided with 2.1. Then when MachV or
SetMachV is run, gadget, menu and message text is in the selected
language.
```

MachV opens a "catalog" named:

locale:catalogs/your_language/machv/machv.catalog

SetMachV opens:

locale:catalogs/your_language/machv/setmachv.catalog

PrintMacros opens:

locale:catalogs/your_language/machv/printmacros.catalog

These must be placed in the users preferred catalog directory. For example, for German users, the catalogs would be placed in the directory:

locale:catalogs/deutsch/machv

French users would place them in:

locale:catalogs/français/machv

The script CopyLanguages is provided to copy the catalogs to your Locale: directory.

Currently, these are the only languages supported. If you are interested in other languages, write to us.

1.9 MachV.guide/SetMachV

SetMachV
****

The main window of SetMachV is where you do most of the work with macros. Macros may be created, modified, copied or appended in this window.

All other features are accessed through sub-windows "Clip Features", "Mouse Features", "Clock Features" and "Miscellaneous Features".

You may also create new or delete additional configurations with the list gadget and the "Add" and "Del" gadgets located in the upper left corner.

There are (at least) five ways to run SetMachV.

1. The easiest is to click the left button on the clock and then press the right button.

2. Use the hotkey <Ctrl-Left-Amiga-ESC>, which has EXE_SETMACH as its macro.

3. Type SetMachV in your shell window.

4. Double click on the SetMachV icon.

5. Execute an EXE_CMD"SetMachV" macro.

   Method 1 and 2 are preferred because MachV will signal SetMachV to
move to the front immediately if it is already running, while
EXE_CMD"SetMachV" must load SetMachV from the disk.

   See Macros, and see Functions in Macros for information on these
macros.

   NOTE

 * MachV searches for a program named SetMachV in the path as it was
   set when MachV was first run. You may also specify the name and
   path on the command line or in the icon's Tool Types gadget with
   SETMACH=path/name.

 * Do not put SetMachV in the WBStartup drawer as it will be
   automatically run when you reset the computer.

   Using SetMachV
   SetMachV Gadgets
   Function Completion
   Loading MachV.prefs
   SetMachV Preferences
   Terminating SetMachV

1.10 MachV.guide/Using SetMachV

Using SetMachV
==============

   Changing The Current Hotkey
   Selecting By Name
   Modifying Macros
   Function Completion
   Copying and Appending
   Copying To a Named Macro
   Copying To Another Configuration
   Add Macro
   Del Macro
   Buffer Size
   SetMachV Help
   Flags
   Sorting Macros
   Printing Your Macros

1.11   MachV.guide/SetMachV Gadgets

```
SetMachV Gadgets
================
```

   Some gadgets have a letter underlined. Pressing that letter will
have the same effect as clicking on the gadget. Note that this will not
happen if a string gadget is active.

   Pressing <Return> will close the active window or quit SetMachV if
the main window is active.

   Pressing <up-arrow> and <down-arrow> has the same effect as clicking
on the up and down gadgets located under the current hotkey text. These
gadgets step you from one key to the next.

   Pressing <Tab> will activate the first string gadget in the active
window.

   When the "Macro" string gadget is active, pressing <Tab> will move
the cursor to the right edge. Repeated pressing will shift the text to
the left. Pressing <Shift-Tab> has the opposite effect. This is useful
when you have very long macros.

   SetMachV also provides function name completion when you press
<Alt-Tab> when the "Macro" string gadget is active.

   See Function Completion.

1.12   MachV.guide/Changing the Current Hotkey

```
Changing the Current Hotkey
===========================
```

   The current Hotkey is displayed in the text box just under the
"Select Hotkey" gadget. The qualifiers needed for this hotkey will be
highlighted below and to the left. The actual macro will be displayed
both in the "Macros" list box and in the string gadget underneath. If
more than one macro is attached to a particular key, they will also be
displayed in the list box. The macro's name, if any, will be displayed
in both the "Macro Names" list box and the string gadget underneath.

   It is important to note that the two list boxes are independent. A
macro name and its associated macro may not be the same horizontally if
you use the scroll gadgets on the side. They will, however, be directly
opposite each other after you select a macro or hotkey.

   To modify or delete a macro, you first must make that macro the
current one. The simplest way is to click on "Select Hotkey" and then
when prompted by the

                Select A Hotkey.
          Use Caps Lock For No Key.

                   Press ESC To Abort.

   message, press the desired hotkey. (If you want to make a mouse
button current, hold down your selected qualifier keys and click INSIDE
the message box.)

      If you have Clock Gadgets, you may also click on the desired clock
gadget that you want to become current.

      Because you can abort by pressing <ESC>, you cannot select <ESC>
this way. If you want to select <ESC>, click on "Select Hotkey" and
press <Ctrl-ESC> or <Shift-ESC> (for example) and then click on the
qualifier gadgets until none are highlighted.

      Also, because you rarely want a macro attached to the left mouse
button and because it is easy to accidently click it, you must select
it in a similar manner as for <ESC>.

      You may step through all the keys that have macros one by one with
the up and down gadgets. These are located next to the <Shift>
qualifier gadget. Only the keys that have macros attached will be
displayed. You may also use the up and down arrow keys.

      If the macro is in the list box, you may click on it or on its name.
The macro will then be displayed in the string gadget and the
qualifiers for the hotkey will be highlighted.

      You may also click on the qualifier gadgets to make a hotkey
current. If you select qualifiers that have no macro, the "Macro"
string gadget will be empty, ready to accept a new macro.

      If you have a non-standard keyboard, additional keys will display
their hexidecimal keycode in the current hotkey gadget.

1.13   MachV.guide/Selecting by Name

Selecting by Name
=================

      The "Macro Names" list box displays the (possibly) descriptive names
for the macros.

      If you want to display a macro by its name, click on "Select Named"
and select or enter the name in the list gadget that will pop up. The
string gadget may not be initially activated. You may activate it by
clicking in it, or by pressing the <Tab> key.

      You may also select a named macro by entering the name in the "Macro
Names" string gadget and pressing <Shift-Return>. Pressing just
<Return> will simply change the name of the current macro.

      To display macros that are not attached to any key, click on "Select
Hotkey" and then press <Caps Lock>.

1.14   MachV.guide/Modifying Macros

```
Modifying Macros
================
```

    Once that you have made the desired macro current, you may modify or
delete it, or copy or append it to another hotkey.

    Click in the "Macro" string gadget or press <Tab> to activate it.
Make your changes and press <Return>. Press <Right-Amiga-X> or click on
"Del Macro" to delete it. Pressing <Tab> will move the cursor to the
right edge of the gadget. Repeated pressing will shift the text left.
Pressing <Shift-Tab> will move the cursor left. This is useful when
editing long macros.

    To enter the example (see Recording Macros), click on "Select Hotkey"
and then press <Alt-D>. Now enter "dir df0:\N" (with the quotes!) in the
"Macro" string gadget and press <Return>. That's it! "dir df0:<Return>"
will be played back when you press <Alt-D>. If you recorded the previous
example, the macro will already be there.

    For "quick completion" of function names see Function Completion.

1.15   MachV.guide/Copying and Appending

```
Copying and Appending
=====================
```

    "Copy" will copy a macro from one hotkey to another and "Append" will
append a macro to another hotkey.

    First make the hotkey that you want to copy or append from the
current hotkey, then select "Copy" or "Append". When prompted by the
flashing message, press the desired hotkey combination that you want to
copy or append to.

    See Copying to Another Configuration.

1.16   MachV.guide/Copying to a Named Macro

```
Copying to a Named Macro
========================
```

    To copy or append to an unkeyed named macro, proceed like a normal
copy but press <Caps Lock> when prompted for a hotkey and then enter or
select the name from the list gadget that will appear.

1.17 MachV.guide/Copying to Another Configuration

```
Copying to Another Configuration
================================
```

   Proceed like a normal copy but when the message is flashing, select
the desired configuration by clicking on the "Next" or "Prev" gadgets.
Do not use "Select Cfg". When the desired configuration is current,
press the destination hotkey.

   CAUTION!

   Once the "Select A Hotkey." message is flashing, any key that you
press will receive the macro...except <ESC>. Use this key to abort. You
cannot copy to the <ESC> key.

1.18 MachV.guide/Add Macro

```
Add Macro
=========
```

   Selecting "Add Macro" will prompt you to select a new hotkey unless
the "NO KEY" macros are currently being displayed. If they are, a new
macro will be created. You may use <Caps Lock> to select a macro by
name.

1.19 MachV.guide/Del Macro

```
Del Macro
=========
```

   Clicking on "Del Macro" will delete the current macro. This is the
same as clicking in the current macro string gadget and pressing
<Right-Amiga-X> and <Return>.

1.20 MachV.guide/Buffer Size

```
Maximum Macro Size
==================
```

   The default maximum macro size is 512 characters but this may be
changed in the "Buffer Size" gadget. This controls the size of dynamic

buffers which are allocated only during macro recording or when
SetMachV is running. The actual macros take up only as much space as
they need plus a little bit of overhead. The maximum size allowed is
32,767 characters. If you try to enter too small a size, the minimum
needed for your largest macro will be entered for you.

   Note: If you are recording all mouse movements, you will need to set
the buffer size considerably higher than 512.

   Curiosity note: Why 512? Why not 500? 512 is one half of 1024 which
is 1K. Programmers have a hard time switching from bits, bytes and Ks
(among other things).

1.21   MachV.guide/SetMachV Help

Help
====

   Clicking on "Help" (or pressing <Help>) will open a window showing
all the escaped characters and special functions that may be used in
macros.

   The functions will be displayed in lowercase unless you hold down the
<Shift> key while clicking on "Help" or pressing <Help>.

   Move through the pages of help by clicking on "Page" or pressing <P>.
Move backwards by holding the <Shift> key.

1.22   MachV.guide/Flags

Flags
=====

   Macros may have seven flags attached to them that are selected from
the "Flags" window. Click on "Flags" or press <F> to view or change
these flags.

Write Protect
     When checked, you cannot record over the existing macro. It may
     still be modified in SetMachV. It is a good idea to write protect
     macros like your NewShell macro and the macro that starts and
     stops recording.

     Hex representation - 0x80

Disabled
     When this flag is checked, the macro will not execute and the key
     will be passed through to other programs.

     Hex representation - 0x40

Temporary
     A hotkey may be marked as temporary by checking this flag. When
     you save your settings file the name and flags will be saved but
     not the macro. If you commonly use particular keys for recording
     "scratch" macros, it may be a good idea to mark these as temporary.

     Hex representation - 0x20

Relative
     When this is checked, all mouse moves will be relative to the
     active window. Mouse actions that select icons should probably be
     recorded to a macro that is marked as relative so that when
     playing back the macro it will not matter where you have placed
     the window containing the icons.

     Hex representation - 0x10

Clock Gadget
     Selecting this flag will create a gadget for this macro if it has
     a name and if "Gadgets" is selected in the Clock Features window.

     Hex representation - 0x08

Non Recursive
     If you do not want this macro to be recursive, this should be
     selected. You might want to do this if you want to intercept a key
     combination, perform some function and then pass on the key.

     For example, if you wanted to send the clock to another screen
     when you press <Right-Amiga-Q>, set the "Non Recursive" flag for
     this macro:

     CLK_TO_NEXT_SCREEN "\RQ"

     Hex representation - 0x04

Global
     Macros in the first configuration with this flag set will be
     accessible from other configurations. This eliminates the need
     for duplicate macros.

     Hex representation - 0x02

Auto Execute
     This macro will be executed based on the time supplied at the
     beginning of the macro. All EXE_AUTO functions should have this
     flag set. This flag is automatically set when you create an
     EXE_AUTO macro in SetMachV.

     Hex representation - 0x01

No Execute Message
     When checked, the 'Executing Macro' message will not be displayed
     in the clock.

     Hex representation - 0x100

The "Clear All" and "Set All" gadgets in the "Flags" window will
clear or set the flags that are displayed for all macros in the current
configuration. For example, you could check the "Write Protected" flag
and then click on "Set All". All macros would then be protected.

These may also be set with the macro function:

MSC_SET_FLAGS"xx,name"

MSC_SET_FLAGS"40,StartStop"
     Set the "Disabled" flag for the macro named 'StartStop'.

MSC_SET_FLAGS"80,#?"
     Set the "Write Protect" flag on all macros. The #? is the AmigaDos
     wildcard for 'all'.

MSC_SET_FLAGS"40,NewShell,0"
     Clear the "Disabled" flag for the macro named 'NewShell'. The '0'
     as the third argument indicates 'Clear'. A '1' is the default and
     indicates 'Set'.

At the bottom of the "Flags" window are two integer gadgets. You may
position a clock gadget with these. They are disabled if the current
macro does not have a gadget. See Quick Edit.

1.23   MachV.guide/Sorting Macros

Sorting Macros
==============

If a key has just a few macros, it probably doesn't need to be
sorted. But if you have a large number of named macros, it is easier to
locate them if they are sorted.

First make the hotkey current (or "NO KEY" for unkeyed macros),
select "By Name" or "By Macro" and then click on "Do Sort".

1.24   MachV.guide/Printing Your Macros

Printing Your Macros
====================

If you find that it is getting difficult to remember all your macros,
SetMachV provides the ability to print them in a table format.

Click on "Print Macros". A small window will appear with three
gadgets. You can select to have the output sent to a window for
viewing, sent to your printer, or sent to a file for later editing.

When the desired destination has been selected, click on "Go". If
you have selected to have the output sent to a window, after each
window full of macros have been displayed, press <Return> to continue.
Press <Ctrl-C> or click on the close gadget to quit.

Just the current configuration will be printed. To print another
configuration, make it active and then select "Print Macros".

"PrintMacros" is actually a separate program. SetMachV merely runs
this program when you click on the gadget. PrintMacros must be in the
current path or in the same directory as SetMachV. The source for
PrintMacros is included as an example of interfacing with MachV.

1.25 MachV.guide/Function Completion

```
Function Completion
===================
```

Function name completion is a quick way to enter function names in
the macro string gadget. When entering a function, pressing <Alt-Tab>
will complete the entry for you. If what you have entered is ambiguous,
a window with possible completions is opened. You may then select the
desired function by pressing the associated letter. You may also press
<Return> or click on the close gadget to close the window.

Text entered will be in the same case as you entered. If you typed:

win_to_f<Alt-Tab>

win_to_front

will be entered. If you typed:

Msc_C<Alt-Tab>

Msc_Chg_Dir

will be entered. If you typed:

MSC_B<Alt-Tab>

MSC_BLANK_SCREEN

will be entered.

Functions that do not require the 3 letter prefix may also be
completed using this method.

z<Alt-Tab>

expands to:

zoom_window

1.26 MachV.guide/Loading MachV.prefs

```
Loading MachV.prefs
===================
```

   MachV.prefs is a text file that contains all your settings and
macros. It is loaded when MachV is first run. If MachV is unable to
find the file, you can load it using SetMachV. Run MachV and then
running SetMachV, use the Project/Open... menu selection and pick the
MachV.prefs file from the file requester.

   See Starting MachV.

1.27 MachV.guide/SetMachV Preferences

```
SetMachV Preferences
====================
```

   SetMachV may open its main window on the Workbench screen or on a
custom interlaced or non-interlaced screen. If desired, the custom
screen may scroll up from the bottom. The screen's title bar may be
made accessible by selecting "Preferences/Screen Options/Title Bar"
from the menu.

   The changes will take affect immediately. If you want it to always
open the way that you have selected, save from the "Project/Save As..."
or "Project/Save" menu items. This saves all changes, macros, options
etc.

1.28 MachV.guide/Terminating SetMachV

```
Terminating SetMachV
====================
```

   You may quit SetMachV in several ways. Simply press <Return> while
SetMachV's main window is active, click on the close gadget, select
"Project/Quit" from the menu or press <Right-Amiga-Q>. These all
accomplish the same thing – shut down SetMachV.

   You may also select "Project/Hide" from the menu or press
<Right-Amiga-H>. This will close the SetMachV screen or window but will
leave SetMachV in memory waiting for you to run SetMachV again.

1.29 MachV.guide/Macros

```
Macros
******
```

   The real power of MachV is its macro ability. A macro may simply be
keystrokes or mouse movements that can be played back. But they can also
perform many window and screen functions, bring up a list of windows or
screens, display the clipboard and execute other programs. Macros may
pause for input, execute ARexx programs, send environment variables as
keystrokes and display alarms.

   Recording Macros
   Correcting Mistakes
   Executing Macros
   Aborting Execution
   Clearing a Macro
   Accidents Happen!
   Recording Mouse Button Clicks
   Recording All Moves
   Recording Menu and Gadget Selections
   Notes On Recording
   Using Escaped Characters
   Functions in Macros
   Function Arguments
   Multi-Function Macros
   Pausing
   Comments
   Quick Edit
   Auto Executing Macro
   Executing Programs

1.30 MachV.guide/Recording Macros

```
Recording Macros
================
```

   One of the more important functions is MAC_RECORD_KEYED. This is the
function that starts or stops a recording session. In the sample
MachV.prefs file, this is attached to the key combination <Alt-ESC>. By
pressing this, and then pressing the key combination that you want for
the hotkey, you begin a recording session. Press any keys or mouse
buttons that you want recorded and then press <Alt-ESC> again. The
macro is now ready to be played back.

   A practice session may help you to understand. To do this, you must
be in a Shell window. You also need a MAC_RECORD_KEYED macro already
setup and attached to <Alt-ESC>. You should have this already if MachV
loaded the sample MachV.prefs file.

     Press <Alt-ESC>   You will hear a beep and the clock will
                          display the message "Select Hotkey..."

Press <Alt-D>   The clock will display "Recording Macro..."

Now type <dir df0:> without the <> and press <Return>.

The directory of the disk in df0: will be displayed.

Now press <Alt-ESC> again to end recording.

You may now press <Alt-D> and your recorded keystrokes will be
played back.

You will hear a short beep for each keystroke and mouse button
entered during recording.

Besides having text in macros, you may include any special key by
preceding a letter with a backslash. These are called escape characters.
Some examples:

\F1 is Function key 1

\E is the ESC key

\5 is numeric key pad 5

These will be entered for you during recording. You can see all of
them by pressing "Help" in SetMachV. The above macro will actually be
recorded as "dir df0:\N".

This macro may be viewed with SetMachV by clicking on "Select
Hotkey" and then pressing <Alt-D>. The macro should be in the top of
the Macro list box and in the string gadget underneath. You may edit it
if you like or delete all the characters to free the hotkey.

Notice that while you did not type the quotes, they appeared in the
macro in SetMachV. All text that you want played back as keystrokes
must be enclosed in quotes.

You may also "Quick Edit" the macro by pressing <Alt-Help> and then
<Alt-D> when the QuickEdit window appears. Press <Return> or click on
the close gadget to close the edit window.

See Quick Edit.

REMEMBER

All keystrokes in a macro must be enclosed in double quotes. If you
need to enter double quotes within the macro, you must precede them
with a backslash. Example:

"This is a macro with a \" in it."

Also note:

Some editors or word processors may not seem to accept played back
keystrokes immediately. This can usually be corrected by setting the
"Playback Priority" to 0 or to whatever the priority of the application

is.

1.31 MachV.guide/Executing Macros

```
Executing Macros
================
```

   A macro may be executed by simply pressing the hotkey that it is
attached to. In addition, they may be executed by name, by clicking on
a "gadget" in the clock, or at a specific time and date.

   In the supplied MachV.prefs file, there is a function:

   MAC_SELECT_NAMED""

   attached to <Ctrl-Amiga-N>. By pressing this, a window with a
scrollable alphabetized list of all macros that have names will appear.
You may select from this list and the macro will execute.

   The string gadget below the list will be initially empty and active.
After you have made a selection, the string gadget will be preset with
the previously used named and will be inactive. This allows the use of
<O> and <C> for the "Ok" and "Cancel" gadgets. You may activate the
string gadget by pressing the <Tab> key.

   The search for a macro name that matches what you enter in the string
gadget is not case sensitive and you need to enter just enough
characters to match the macro name. "Proc" will match "ProCalc".

   A macro may also call itself if the Non-Recursive flag is not
checked. The number of times is limited by the variable
machv_recursion_depth. The macro attached to F10:

   "abc\F10"

   will type:

   abcabcabcabcabcabcabcabcabc

   See Environment Variables.

1.32 MachV.guide/Aborting Execution

```
Aborting Execution
==================
```

   Pressing <ESC> during macro execution will abort further execution.
An exception to this is when pausing for variable length input.

   In this case, press <ESC> to terminate input and <Ctrl-ESC> to abort

the macro.

1.33 MachV.guide/Correcting Mistakes

```
Correcting Mistakes
===================
```

   If you are recording a macro and you press <Backspace> to correct a
typing error, the error and the backspace will be recorded. You may use
<Ctrl-H> to correct a keystroke without recording the mistake or the
backspace. A notable exception to this is gnuemacs which uses <Ctrl-H>
for online help. This is not a real problem though, because gnuemacs
has good macro recording built in.

   Notes:

 1. This works only with programs that accept <Ctrl-H> as a backspace.

 2. If you press any non-printable key that you wish to erase,
    <Ctrl-H> will delete it from the buffer, but it cannot undo
    whatever action that key performed.

 3. Because of this feature, you cannot record <Ctrl-H>. If you need
    to, you may enter it as \Ch in the macro from SetMachV.

1.34 MachV.guide/Clearing a Macro

```
Clearing a Macro
================
```

   To clear a macro, that is, to free it from a hotkey, press
<Alt-ESC>, then the hotkey, then <Alt-ESC> again. This records no
keystrokes which means that the macro is empty and the hotkey is free.

   You may also clear macros using SetMachV or Quick Edit.

   See Quick Edit.

1.35 MachV.guide/Accidents Happen!

```
Accidents Happen!
=================
```

   If you should accidently press the <Alt-ESC> hotkey, the clock will
display "Select A Hotkey...". Simply press <Alt-ESC> again. No harm
done! You may also press <ESC> which aborts a macro.

If you accidently press <Alt-ESC> and then more keys, the first key
pressed after <Alt-ESC> will become a hotkey with the following keys
the macro. You can press <Alt-ESC> again to stop recording (you cannot
simply press <ESC> since MachV must allow you the ability to record
that key). Then press <Alt-ESC> (to start recording), then the
"accidental" hotkey, then <Alt-ESC> again to stop recording.

Now you may not even remember what key you pressed that became the
hotkey. Then the easiest thing to do is run SetMachV and step through
each hotkey.

1.36   MachV.guide/Recording Mouse Button Clicks

Recording Mouse Button Clicks
=============================

Mouse button clicks may be recorded, or entered manually in the
"Macro" string gadget. When recording a button click, MachV records the
mouse position, which button it was and whether it was pressed or
released. For example:

   XY"155,075" LBD

   (Note that recorded MOU_ and PAU_ functions are optimized to not
record the 3 letter prefix.)

When played back, the mouse will be moved to x = 155, y = 75 and a
left button down will be sent.

A SECONDS"nn" will be inserted for every nn seconds delay between
clicks. This will allow for menus to drop or requesters and windows to
open. You may adjust or delete them from your macro if desired. Values
with tenths such as SECONDS".4" or SECONDS"1.6" are allowed.

If you record just the left button, you might see something like
this:

   XY"155,075" LBD SECONDS"1" LBU

unless you moved the mouse between pressing and releasing the mouse
button, in which case the new x and y values would be entered before
the LBU.

Mouse coordinates are relative to the left and top edge of the
screen. For coordinates relative to the left and top edge of a window,
set the "Relative Moves" flag in the Flags window. See Flags. This
flag must be set prior to the recording of mouse movements.

1.37   MachV.guide/Recording All Moves

```
Recording All Moves
===================
```

   If you need all mouse movements recorded, for a painting
demonstration for example, you need to set the Playback Delay value in
Mouse Features somewhere between 1-99. These represent delays of 50ths
of a second between moves. 1 = 1/50, 2 = 2/50 etc. This is needed
during playback by many programs or the movements may come to quickly.

   Note that this value must be set to some positive number both BEFORE
recording and during playback. During playback, you may get acceptable
(although fast) results by setting "Play Delay" and "Playback Priority"
to 0.

   During recording, the clock message should show "Recording
Macro...ALL".

   Recording all moves requires MUCH more memory than you would
normally need. Use it only if you need to. You may need to increase the
recording buffer. See Buffer Size.

1.38   MachV.guide/Recording Menu and Gadget Selections

```
Recording Menu and Gadget Selections
====================================
```

   When a menu item has a subitem it may be necessary to bring the
mouse down to the menu item, click the left button on it, then move the
mouse over the subitem and click the left button or release the right
button.

   Sometimes gadgets will not respond to a quick button down/up. It may
require pausing between the down and up. See Pausing.

   Do not use a pause larger than one second after the right button is
pressed for a menu selection. The macro may stop executing.

   For a better method, see Sending IDCMP Messages directly to a program
with the functions:

   See WIN_SEND_MENU.

   See WIN_SEND_GADGET_DOWN.

   See WIN_SEND_GADGET_UP.

   See WIN_SEND_CLOSE.

1.39   MachV.guide/Notes On Recording

Notes On Recording
==================

   The mouse will be moved to the same position as when recorded which
may or may not be over the desired gadget or icon. See Flags, for
relative movements.

   If you are pressing a qualifier such as <Shift>, it (the \S) will be
placed before a XY"xxx,yyy" and will affect the mouse movement and the
first click. For example, the left button down will have the <Shift>
qualifier attached with this macro:

   "\S" XY"330,122" LBD

   Do not depend on the SunMouse to activate a window during play back.
When recording, click in the window to activate it, even if it is
activated by the SunMouse.

   If you have "Mouse To Menu" selected, you also must press and
release the left button after the mouse is over the desired menu.

1.40   MachV.guide/Using Escaped Characters

Using Escaped Characters
========================

   All non-printable keys are preceded by a backslash "\". For example,
\B is the Backspace key, \H is the help key and \F3 is function key 3.

   These qualifiers modify the key or mouse activity immediately
following them:

        \C Control.
        \S <Shift>   (Left Shift played back).
        \A Alternate (Left Alt played back).
        \L Left-Amiga (C=).
        \R Right-Amiga.

        "\Cc\u\A\S\l"

   This plays back the key combinations:

        Ctrl-C Up-Arrow Alt-Shift-Left-Arrow

   You do not need \S or \A before any character that is printable. For
example <Alt-E> displays the copyright symbol ©. You can press <Alt-E>
instead of entering \Ae.

   Note that all escape characters must be used in the case as shown.

   Also, the backslash itself and double quote marks must be preceded
by a backslash. For example:

```
"A backslash \"

"A double quote \""
```

When a character is also on the numeric keypad, that key is the one that is sent when played back. This can cause problems if you have a macro attached to the keypad key. For example, suppose you have a macro:

```
"/*  */\l\l\l"
```

This will send the keypad key for '*' since on the standard portion of the keyboard the '*' is on a shifted key (<Shift-8>). To play this back not using the keypad, you would need to use the following:

```
        "/\S8  \S8/\l\l\l"
    or
        "/\S*  \S*/\l\l\l"
```

On the American keyboard, the following keys are affected: (,), *, and +.

1.41 MachV.guide/Functions in Macros

```
Functions in Macros
===================
```

All special functions have the same basic format – they all begin with an (usually) optional three letter catagory code followed by an underscore, followed by the function. Examples:

```
SCR_WB_TO_FRONT          Send the Workbench screen to the front.

WIN_FRONT_TO_BACK        Send the front window to the back.
```

The functions:

```
        Win_Back_To_Front    Clk_To_Front
        Scr_Back_To_Front    Win_To_Front
                             Scr_To_Front
        Win_Front_To_Back
        Scr_Front_To_Back
```

are ambiguous without the prefix. Be sure to use the prefix in these cases.

Functions are not case sensitive: CLP_VIEW, Clp_View and clp_view are all acceptable. Without the prefix, you can use VIEW or View or view.

Some functions require function arguments which follow the function and are enclosed in double quotes. Examples:

```
EXE_CMD"NewShell"        Create a new shell.
```

```
    WIN_ACTIVATE"Workbench"   Activate the Workbench window.
```

In order to tell whether or not a quoted string is an argument or keystrokes, spaces are not allowed between the function and its arguments.

```
    WIN_ACTIVATE"NewShell"   is okay but

    WIN_ACTIVATE "NewShell"   is not.
```

See Function Arguments.

Functions cannot be recorded, they must be entered using SetMachV. The two exceptions are mouse functions and PAU_SECONDS. These are recorded when needed.

The functions are described in their individual sections. See Windows, see Screens.

1.42 MachV.guide/Function Arguments

```
Function Arguments
==================
```

Many functions accept one or more arguments that are often optional. For many WIN_ functions the first argument specifies which window to apply the function to.

All window functions operate on the window as selected in this manner:

```
Top window
     WIN_function"T" WIN_function"T,arg2,arg3,..."

Active window -
     WIN_function"A" WIN_function"A,arg2,arg3,..."

Under mouse window -
     WIN_function"M" WIN_function"M,arg2,arg3,..."

Titled window -
     WIN_function"title" WIN_function"title,arg2,arg3,..."
```

There is only the remote possiblity of a conflict if you must specify a window whose title is one of these single characters. In these cases you could use:

```
   WIN_function"(A)"      WIN_function"(M)"      WIN_function"(T)"
```

Titles may also include AmigaDos wild cards or environment variables. See Window Titles, and see Environment Variables.

When using "title", you may precede the title with the '$' symbol to indicate a window based on the name of the program that owns it.

Example:

```
WIN_ACTIVATE"$program".
```

This MUST be the first character. It cannot be used together with wildcards.

Use the "Select Title" in SetMachV to determine the correct title to use. Use the list only to browse through the titles – selecting one will set it as your configuration title.

As in quoted text, if you need a backslash \ or double quote ", you must precede it with a backslash. In argument strings, commas are argument separators. You must also precede them with a backslash if they are not intended to separate arguments.

This is most frequently encountered in EXE_CMD functions. As an example, you should use:

```
EXE_CMD"echo >speak: \"time for lunch\, now!\""
```

1.43 MachV.guide/Multi-Function Macros

```
Multi-Function Macros
=====================
```

A vertical bar "|" is used to separate a macro into sections. Each section will be executed in turn. For example, when this macro is executed the first time,

```
      WIN_SEND_MENU",Icon" | WIN_SEND_MENU",Name" |
      WIN_SEND_MENU",Date" | WIN_SEND_MENU",Size"
```

the menu selection with the text "Icon" will be sent to the active window. The next time, the "Name" selection will be sent, and so on.

This macro can be useful – it alternately sends ctrl-s and ctrl-q which can start and stop output to the shell or in a term program.

```
"\Cs" | "\Cq"
```

You may use one hotkey to alternate between two functions:

```
SET_MOUSE_PORT"0" | SET_MOUSE_PORT"1"
```

You are not limited to one function in each section, you could do this:

```
   SET_MOUSE_PORT"0" CLK_CUSTOM_MSG"Port 0" | SET_MOUSE_PORT"1"
   CLK_CUSTOM_MSG"Port 1"
```

A hotkey could toggle the size of a window with this:

```
WIN_SIZE"100,10" | WIN_SIZE"640,200"
```

1.44 MachV.guide/Pausing

```
Pausing Within A Macro
======================
```

   Sometimes it is necessary to pause during macro execution. There are
three types of pauses; pause for 0.1 - 99.9 seconds, pause for fixed
length input, and pause for variable length input. Examples:

```
PAU_SECONDS"1"
     Pause for one second.

PAU_SECONDS".3"
     Pause for .3 seconds.

PAU_SECONDS"2.5"
     Pause for 2.5 seconds.

PAU_FIX_INPUT"5"
     Pause for five keystrokes.

PAU_VAR_INPUT
     Pause for a variable number of keystrokes. Terminate input with
     the <ESC>. To abort the macro during input, press <Ctrl-ESC>.

PAU_VAR_INPUT"44"
     Pause for a variable number of keystrokes. The argument "44" is
     optional. When supplied, it becomes the terminate keycode used to
     end input. In this case, 44 is <Return>. The default key is 45,
     <ESC>. To abort the during input, press <Ctrl-TerminateKey>
     (<Ctrl-ESC> unless you used the optional argument).
```

   The pause for input functions may be useful in data entry macros
where the macro performs some functions, waits for some user input, and
then continues with more functions. The macro may even call itself over
and over. The number of times is limited by the variable
machv_recursion_depth.

   See Environment Variables.

   An easy way to setup such a macro would be to record a complete data
entry loop, and then, using SetMachV, enter the pause in the
appropriate place.

   A data entry macro might look like this:

   MOU_XY"150,40" MOU_LBD MOU_LBU PAU_VAR_INPUT MOU_XY"460,23" MOU_LBD
MOU_LBU "\F1"

   With this attached to the <F1> key (for example), the mouse would
click at 150,40 (possibly a string gadget), wait for a variable number
of keystrokes followed by <ESC>, click at 460,23 (which might be a
"Next" gadget), and then execute <F1> which would then cause it to

repeat.

   Environment variables can be set with the PAU_FIX_INPUT and
PAU_VAR_INPUT functions. See Environment Variables.

1.45 MachV.guide/Comments

```
Comments
========
```

   For the purpose of identifying portions of your macro or for adding
comments, any text may be inserted between /* */.

   XY"167,134" /* move to open gadget. */ LBD LBU /* click on OPEN */

1.46 MachV.guide/Quick Edit

```
Quick Edit
==========
```

   You can quickly edit a macro by executing the MAC_QUICK_EDIT
function. This is <Alt-Help> in the sample MachV.prefs file. You may
press any hotkey while the Quick Edit window is active and its macro
will be displayed.

   If you have gadgets in the clock, you may click on the gadget to
make its macro display in the Quick Edit window. In addition, the
QuickEdit window expands to display two XY position string gadgets. You
may move a gadget by entering its coordinates here.

   While the Quick Edit window is active, all hotkeys pressed will
display their macro in the edit string gadget. If the window is not
active, hotkeys will function normally.

   If the string gadget IS active, pressing:

  1. A hotkey will display the macro for that hotkey.

  2. <Return> will cause MachV to accept the macro.

  3. <Tab> will move the cursor to the right edge of the gadget.
     Repeated pressing will shift the text left.

  4. <Shift-Tab> will move the cursor left.

   If the string gadget is NOT active, pressing:

  1. <Tab> will activate it.

  2. <Return> will close the Quick Edit window.

3. A hotkey will display that hotkey's macro.

4. The underlined letter on the Save gadget will save and close.

5. Any other key will create a new empty macro attached to that key
   combination.

   You may save your changes by clicking on "Save". This does the same
thing as running SetMachV and saving. Note that you only need to save
to make your changes permanent.

   If an ARexx macro is executed while this window is open, the macro
will be displayed in the string gadget.

1.47 MachV.guide/Auto Executing Macro

```
Auto Executing Macro
====================
```

   Macros may be automatically execution in several ways. Some use the
EXE_AUTO function and some execute based on the name of the macro.

   In many cases you may find the Clock Alarms in the Clock Features
window in SetMachV easier to use.

   The general forms are:

```
        EXE_AUTO"hh:mm"               Execute at a specific time (every day).

        EXE_AUTO"hh:mm,mm-dd-yy"      Execute at a specific time and date.

        EXE_AUTO"hh:mm,Day,Day,..."   Execute at a specific time and day.

        EXE_AUTO"+n"                 Execute every "n" minutes.
```

   The arguments of this macro are optional, but when supplied indicate
the time and possibly the date or day that you wish the macro to
execute. Normal AmigaDos wild cards are allowed. The "+n" form
indicates that the macro should execute every "n" minutes.

   Time must be in 24 hour format.

   The "Auto Execute" flag in the "Flags" window must be set. This is
done automatically when you enter an Exe_Auto macro but you may set it
yourself.

   Examples:

```
EXE_AUTO"09:30"
     Execute at 9:30 am. (Note the leading 0)

EXE_AUTO"14:45"
     Execute at 2:45 pm. (Use 24 hour time)
```

```
EXE_AUTO"??:15"
     Execute every 15 minutes past the hour.

EXE_AUTO"+5"
     Execute every 5 minutes.

EXE_AUTO"08:45,07-22-94"
     Execute at 8:45 am on July 22, 1994.

EXE_AUTO"19:00,??-10-??"
     Execute at 7:00 pm on the 10th of the month.

EXE_AUTO"19:00,04-1(1|2|3)-94"
     Execute at 7:00 pm on 04-11-94 and 04-12-94 and 04-13-94.

EXE_AUTO"11:00,Mon,Sat"
     Execute at 11:00 am every Monday and Saturday.

EXE_AUTO"12:00,Mon"
     Execute at 12:00 noon every Monday.

EXE_AUTO""
     Execute when MachV is first run.
```

This function is used in combination with other functions. For example:

```
EXE_AUTO"??:15" MSC_BEEP
     Will beep every 15 minutes past the hour.

EXE_AUTO"+15" MSC_BEEP
     Will beep every 15 minutes.
```

A popular combination of macros is this:

```
     EXE_AUTO"??:15" MSC_BEEP

     EXE_AUTO"??:30" MSC_BEEP PAU_SECONDS".5" MSC_BEEP

     EXE_AUTO"??:45" MSC_BEEP PAU_SECONDS".5" MSC_BEEP PAU_SECONDS".5"
                     MSC_BEEP

     EXE_AUTO"??:00" MSC_BEEP PAU_SECONDS".5" MSC_BEEP PAU_SECONDS"1"
                     MSC_BEEP PAU_SECONDS".5" MSC_BEEP
```

These will beep once at 15 after, twice on the half hour, three times at three quarters after, and four times on the hour. These are 4 separate macros and may of course, be unkeyed.

If days are specified, they may be just long enough to match the day that is displayed in the clock. That is, 'Tuesday', 'Tues', or even 'tu' will match Tuesday. Ranges are not supported. You must enter each day.

To make EXE_AUTO macros in the first configuration execute when other

configurations are active, check the macro's "Global" flag in the
"Flags" window.

    To enter these as unkeyed macros, click on "Select Hotkey" and press
<Caps Lock> when prompted. The current hotkey gadget should now show "NO
KEY" and all the qualifier gadgets should be disabled.

    Now click on "Add Macro". The string gadget under the "Macros" list
box should be activated. Enter the first macro and press <Return>. The
macro should now be displayed in both the string gadget and in the list
box.

    Click on "Add Macro" again and repeat these steps for the other three
macros.

    Note: EXE_AUTO macros attached to "NO KEY" in the first
configuration of multiple configuration setups will always get
executed, even if that configuration is not the current one. You need
not duplicate auto executing macros in all your configurations.

1.48 MachV.guide/Executing Programs

Executing Programs
==================

    Many programs can be run from a hotkey by simply putting

    EXE_CMD"program"

in a macro. For example, EXE_CMD"run myapp" would run a program named
myapp. Usually, you would want to use "Run" in your command string to
keep from suspending MachV during the application's execution.

    The macro attached to <Left-Amiga-ESC> that opens a new shell is an
example of this kind of function.

    SCR_WB_TO_FRONT EXE_CMD"NewShell \"con:0/0/640/100/NewShell\""

    This macro first brings the Workbench screen to the front and then
executes the NewShell program. Note that the double quotes within
double quotes must be preceded by a backslash.

    If you need a comma in the quoted string, use \, otherwise it will be
treated as an argument separator.

    If the program run returns an error code, that will be displayed in
the clock. If the program set a DOS error code the error message will
also be displayed. If Debug is on, SetMachV will be run.

    Avoid Re-Running Programs:

    You may follow the command with a window or screen title. If that
window or screen is found, it will be brought to the front. If it is
not found, the command will be executed. For example:

```
    EXE_CMD"Run >nil: FinalCopy,Final Copy"
```

    If the screen titled "Final Copy" is found, it will be brought to the
front. Otherwise the program will be run.

    The title is optional and may be omitted. If supplied, MachV searches
first for a window title match and then for a screen title match. It
needs to be just long enough to match the title and it is case
insensitive, "final c" will match "Final Copy". You may also use
AmigaDos wild cards to match a title.

```
    exe_cmd"ttx,#?.(c|h)"
```

    This will run the program "ttx" (TurboText), but if it finds a
window or screen whose title ends in .c or .h, that window or screen
will be brought to the front.

    If the screen or window does not have a title, or has one that
changes, you may use the task name prefaced by a '$'. Click on "Select
Title" to list task names. Example of running the text viewer View80:

```
    EXE_CMD"View80,$View80"
```

    Program Stack Size:

    Programs executed with EXE_CMD will have the same stack size as it
was set when MachV was first run. Many programs need a larger stack
than the default 4000. You should use the stack command in your
User-Startup file to at least 10000. You can often check a program's
icon (using them Workbench menu selection "Information") to see what
the recommended size is for that particular program.

1.49 MachV.guide/Windows

```
Windows
*******
```

    This chapter deals with the most powerful and complicated macro
functions. For this reason, they are described here as well as listed
in the Appendix.

  Window Titles
  Moving and Sizing
  Organizing
  Zooming
  Killing
  Sending IDCMP Messages
  Selecting String Gadgets
  Selecting Windows
  Window Waiting
  Window Open

1.50 MachV.guide/Window Titles

```
Window Titles
=============
```

Window and screen functions that require a title may use standard
AmigaDos wild cards. This allows you to specify any window that ends in
.doc, for example, by using #?.doc. If no wild cards are used, the
title in the function needs to be just long enough to match the actual
title. "Workb" will match "Workbench". All comparisions are case
insensitive.

Examples:

```
WIN_TO_FRONT"#?.(c|h)"
    Matches any window title ending in .c or .h

WIN_SEND_MENU"$cygnusEd,Open..."
    Sends the menu selection "Open..." to a window that is owned by the
    program CygnusEd.

WIN_TO_FRONT"NewSh#?"
    The first window found with a title beginning with "NewSh" will be
    brought to the front.
```

Title matching also works when executing programs:

```
EXE_CMD"Run >nil: FinalCopy,Final#?"
```

This brings the window or screen that has a title beginning with
'Final' to the front if it is found, otherwise, the program is run.

You can match a window with no title by using two vertical bars.

```
WAIT_WINDOW"||"
```

Windows may be specified by 'T', 'A', 'M' or title.

```
    T  =  Top window.

    A  =  Active window.

    M  =  Window under mouse.
```

See Environment Variables, for using variables in titles.

1.51 MachV.guide/Moving and Sizing

Moving and Sizing Windows
=========================

     There are many functions for moving and sizing windows. Here are
some examples:

WIN_TO_FRONT"title"
     Move titled window to the front.

WIN_TO_BACK"A"
     Move active window to the back.

WIN_BACK_TO_FRONT
     Move the back window to the front.

WIN_FRONT_TO_BACK
     Move the front window to the back.

WIN_MOVE_WINDOW"title,100,10"
     Move a window to 100x, 10y. Use +0 to leave a position alone.

WIN_MOVE_WINDOW"title,+0,+10"
     Move a window down 10 pixels.

WIN_SIZE_WINDOW"title,640,200"
     Size a window to 640x 200y.

WIN_SIZE_WINDOW"title,+10,+0"
     Grow a window 10 pixels wider.

WIN_CHANGE_WINDOW"title,le,te,w,h"
     Move and size a window to position le = left edge, te = top edge,
     w = width, h = height.

WIN_CHANGE_WINDOW"A,le,te,w,h"
     Move and size the active window.

     When sizing, an omitted number is assumed to be 0, which will leave
that dimension unchanged. When moving, a 0 will move to the pixel
coordinate 0. If it is omitted, it will not move in that direction. For
example:

WIN_MOVE_WINDOW"A,,200"
     Will move the active window to the Y coordinate 200 and the X
     position will not change.

WIN_MOVE_WINDOW"A,0,100"
     Will move the active window to the Y coordinate 100 and to the X
     coordinate 0.

     If the requested size is to large for the window's position, the
window will be moved. If it is to large for the screen, the screen size
will be used. Finally, if it is to small, it will be ignored.

     So, to make a full sized window, it is safe to use:

```
WIN_SIZE_WINDOW"A,1500,1500"
```

To move and size at one time, use:

```
WIN_CHANGE_WINDOW"A,50,10,500,110"
```

This will move the active window to a left edge of 50, top edge of 10, and size to a width of 500 and height of 110.

1.52 MachV.guide/Organizing

```
Organizing
==========
```

You can arrange all windows on a screen in layers. You really need to try it to see what it actually does. Also see SET_NO_LAYER.

The effect is something like this:

```
                 _____
                |*| Utilities_____@|@|
                |_____|_|
                |*| System_____@|@|
                |_____|_|
                |*| Tools_____@|@|
                |                        | |
                |                        | |
                |                        | |
                |_____| |
                |_____|/|
```

Simply put WIN_LAYER in a macro attached to a key combination like <Ctrl-Amiga-L>.

This is a useful function when using TurboText(tm) which opens many windows.

1.53 MachV.guide/Zooming

```
Zooming
=======
```

You can simulate a click on the zoom gadget with WIN_ZOOM_WINDOW. The window to be zoomed must have a zoom gadget or the function is ignored.

```
WIN_ZOOM_WINDOW"A"        Zoom the active window.

WIN_ZOOM_WINDOW"MachV"    Zoom the MachV clock window.
```

This function does not use the mouse, it executes a ZipWindow()

function call.

1.54 MachV.guide/Killing

```
Killing Windows
===============
```

   Use WIN_KILL_WINDOW to close the active window. This is NOT the same
as clicking on the close gadget. It closes a window that may be owned by
another program. This function is included to provide a way of closing
an errant window that a program left behind. USE WITH CAUTION! MAY
CAUSE A SYSTEM CRASH!

   A useful macro to use this with is:

   WIN_SELECT_WINDOW"Window To Kill?" WIN_KILL_WINDOW

   See Sending IDCMP Messages, to send a close message.

   You may also specify a title: WIN_KILL_WINDOW"Errant window title"

1.55 MachV.guide/Sending IDCMP Messages

```
Sending IDCMP Messages
======================
```

   IDCMP messages (Intuition's Direct Communications Message Ports) are
the system's method of notifying a program that an event has taken
place. This event may be a close gadget, menu selection, key pressed or
released, or a gadget down or up message.

   Many programs do not provide keyboard equivalents for menu
selections. You can create your own by using:

   WIN_SEND_MENU"Title,Text"

or

   WIN_SEND_MENU"Title,m,i,s"

   You can send other IDCMP messages to any window that is able to
receive them. When a user clicks on the close gadget, for example,
intuition (part of the operating system) sends a special message to the
program that owns the window. The program then closes the window. MachV
can also send these messages from macros by using WIN_SEND_CLOSE. You
may find it convenient to create a close gadget macro attached to the
hotkey <Ctrl-LeftAmiga-C> or <LeftAmiga-Q>. Pressing this will then
close the window.

   To send gadget messages, you need to find the desired gadget number.

The program AllGadgets may be used to find these numbers. It must be
run from the shell while the window with the gadget in it is open. You
cannot send messages to many gadgets, especially those created with the
GadTools library. If AllGadgets shows GadgetIDs of 0, they were
probably created by GadTools.

    Include that gadget number in the functions:

    WIN_SEND_GADGET_DOWN

    WIN_SEND_GADGET_UP

WIN_SEND_GADGET_DOWN"A,5"
    Send a gadget down message from gadget number 5 to the active
    window.

WIN_SEND_GADGET_UP"A,1"
    Send a gadget up message from gadget number 1 to the active window.

    Some programs need only the gadget up message while others need both
the down and up message. Some programs may also even need a pause
between the two. You will need to experiment with your particular
application.

    You may select a string gadget (that is, perform the same task as if
you clicked in it with the mouse) with the function WIN_SELECT_GADGET.
See Selecting String Gadgets.

    Menu selections may be sent using either the menu item text or its
menu, item and subitem number.

    WIN_SEND_MENU"Workbench,All Files"

will simulate a "Window/Show/All Files" menu selection.

    WIN_SEND_MENU"Workbench,1,7,1" does the same thing.

    If you have difficulty getting the program to respond to your macro
using the item text, try the menu and item numbers. Menu, menu item and
menu subitem numbers start at 0. If you wish to send a menu select
message of the 2nd item in the 1st menu, you would use:

    WIN_SEND_MENU"A,0,1"

    In rare cases, the program that receives the message may not reply
to it and leave MachV waiting forever. Simply press <ESC>.

                        Window Message Examples

WIN_SEND_CLOSE"A"
    Send Close gadget message to active window.

WIN_SEND_CLOSE"Prefs"
    Send Close gadget message to the window titled Prefs.

WIN_SEND_MENU"A,1,6,0"
    Send a menu selection to the active window, menu number 1, item

        number 6, subitem number 0.

WIN_SEND_MENU"A,Quit"
     Find the menu selection "Quit" in the active window and send it.

   Note

   Console windows that have a close gadget or menus are not expecting a
normal intuition message and cannot be closed with WIN_SEND_CLOSE or
receive WIN_SEND_MENU functions. A console window is a special type of
window that uses the console device instead of intuition for its input
and output. Shell windows and "Ed" are two examples of console windows.

   If you have "Debug" selected and you don't want SetMachV running when
you execute a WIN_SEND_CLOSE macro, use this macro:

   msc_debug"off" WIN_SEND_CLOSE"A" msc_debug"on"

1.56   MachV.guide/Selecting String Gadgets

Selecting String Gadgets
========================

   Many 2.04 compatible programs allow you to select a string gadget by
pressing <Tab>. For those that don't (like Workbench's Information), the
function:

   WIN_SELECT_GADGET

   may be used. This will select a string gadget in the active window if
another string gadget is not already active. The next gadget in the
window may be selected by pressing <Return> (to deselect the current
string gadget) and then executing this macro. To avoid the necessity of
pressing <Return>, you could use:

   "\N" WIN_SELECT_GADGET

   which sends a <Return> before activating the string gadget.

   A good hotkey to attach this macro to would be <Left-Amiga-Tab>.
This way, you could press <Tab> with programs that do select gadgets,
and <Left-Amiga-Tab> for those that don't, or to all programs. Don't
attach this to <Alt-Tab> as that will conflict with "Function
Completion" in SetMachV.

   Note that many programs that don't activate the first gadget with
<Tab> do activate subsequent gadgets with <Tab>.

   To select a gadget in a specific window, use:

   WIN_ACTIVATE"title" WIN_SELECT_GADGET

1.57 MachV.guide/Selecting Windows

```
Selecting Windows
=================
```

    You can have a list box open with a list of all windows that are
currently open. The selected window will then become the active window.

```
WIN_SELECT_WINDOW
     Select a window from a list box and make it active. Uses the
     default list box title "Select A Window".

WIN_SELECT_WINDOW"List Box Title"
     Select a window from a list box and make it active. Enclose your
     desired title in quotes.
```

    If a window does not have a title, a pointer to that window will be
displayed in hexidecimal form. e.g. 0x7c5e3d0. If there is only one
unnamed window then it's usually easy to know which it is. Some windows
may have a blank title. If you don't see the title or pointer in the
list, try clicking where one might be in the list.

    You may also activate and bring a window to the front with:

    Win_Activate_Next Win_To_Front"A"

    See WIN_ACTIVATE_NEXT, and WIN_TO_FRONT, for more information.

1.58 MachV.guide/Window Waiting

```
Window Waiting
==============
```

    A macro can wait for a window to become active before continuing.
This is better than using a pause function which may wait too long or
too short. MachV continues to function normally while waiting, but you
should not call another wait function before it returns.

    The title can be an AmigaDos pattern if desired. Example:

    WIN_SEND_MENU"A,Quit" WIN_WAIT_WINDOW"Are You Sure#?" "O"

    This sends a menu selection of 'Quit' to the active window, waits
for a window titled 'Are You Sure...' to become active, then sends the
letter 'O' (which may simulate clicking on an 'Okay' gadget).

    When WIN_WAIT_WINDOW returns, it still may not be safe to do a
function like WIN_SEND_MENU -- the menu strip or user port may not be
set yet. These cases are especially likely if other tasks are also
running. You may follow the window title with a 'U' or 'M' or both.
When the desired window becomes active, MachV will wait until the
MenuStrip or UserPort is set, or upto 2 seconds. The order of 'U' or
'M' does not matter.

```
WIN_WAIT_WINDOW"$terminus1,U,M"
     Wait for the window owned by the program 'terminus1' to become
     active and set both its UserPort and MenuStrip.

WIN_WAIT_WINDOW"$Workbench,M,U,4"
     Wait for a Workbench window upto a maximum of 4 seconds instead of
     2. Allowed values are 1 - 9 seconds.

WIN_WAIT_WINDOW"$Workbench,M,,5"
     Wait for a window owned by Workbench to set its MenuStrip. Wait no
     longer than 5 seconds.
```

   A program may still change a menu strip or user port after
WIN_WAIT_WINDOW returned but before you could send a message. A pause
may be still be necessary:

```
   WIN_WAIT_WINDOW"Serial,U" PAU_SECONDS".5" "bbo"
```

   This will wait for the window 'Serial' to become active, then the
user port to be set, then it will pause for .5 seconds, then send the
keystrokes 'bbo'.

   You can match an untitled window by using two vertical bars in the
title.

```
   WIN_WAIT_WINDOW"||"
```

   The message "Waiting For Window" is displayed in the clock while
waiting.

   Abort a WIN_WAIT_WINDOW function by pressing ESC.

   Note that this function returns when the desired window becomes
active or immediately if it is already active.

1.59 MachV.guide/Window Open

```
Window Open
===========
```

   A ARexx script may test to see if a window is open with:

```
   WIN_WINDOW_OPEN"title"
```

   This returns a '1' in the ARexx variable RESULT if the window is
open.

```
   /* ARexx example of testing to see if a window is open. */

   OPTIONS RESULTS

   ADDRESS MACHV_REXX 'WINDOW_OPEN"Pref"'
   if RESULT = "1" then
```

```
        say "The window is open"
    else
        say "The window is not open"

    See ARexx, for more information.
```

1.60 MachV.guide/Screens

```
Screens
*******
```

The screen functions operate in a similar manner to the window
functions. You can bring screens to the front or send them to the back,
you can select a screen from a list which will bring it to the front
and you can close an errant screen (cautiously!).

Functions that require a title may include AmigaDos wild cards. See
Window Titles.

SCR_BACK_TO_FRONT
    Send the back screen to the front.

SCR_TO_FRONT"title"
    Send titled screen to the front.

SCR_FRONT_TO_BACK
    Send front screen to the back.

SCR_TO_BACK"title"
    Send titled screen to the back.

SCR_WB_TO_FRONT
    Send Workbench screen to the front.

SCR_WB_TO_BACK
    Send Workbench screen to the back.

SCR_SELECT_SCREEN"Optional title"
    Select a screen from a list box using the default list box title
    "Select A Screen". Bring the screen to the front.

SCR_KILL_SCREEN
    Request confirmation then close the front screen. USE WITH
    CAUTION! Some programs in some situations do not close their
    custom screen when they terminate. With this function, you can
    close it, but be careful, the system may crash!

    Normally this is used with SCR_SELECT_SCREEN like:

    SCR_SELECT_SCREEN SCR_KILL_SCREEN

    You may also specify a title:

    SCR_KILL_SCREEN"Bad screen"

You can replace Workbench's standard combination <Left-Amiga-N> with
the macro SCR_BACK_TO_FRONT to bring the back screen (instead of the
Workbench screen) to the front. Replace <Left-Amiga-M> with
SCR_FRONT_TO_BACK. You can add WIN_ACTIVATE"M" to these, which will
activate the window under the mouse after the screen is moved. Or use
WIN_ACTIVATE"T" to activate the top (front) window.

The <Amiga-N> macro might look like this:

SCR_BACK_TO_FRONT WIN_ACTIVATE"M"

1.61 MachV.guide/Environment Variables

Environment Variables
****

Environment variables can be used as function arguments or output as
text. They can be set with the PAU_FIX_INPUT and PAU_VAR_INPUT
functions, the SET_VARIABLE"name,value,optional flags" function, or, if
using global variables, from the shell using "setenv name value".

Environment variables can be either global - accessible from
anywhere, or local - accessible only from MachV.

The variable format is ${varname}. This will expand to the value of
varname when the function executes. For example, if "window" was set to
"Workbench", this would bring it to the front:

WIN_TO_FRONT"${window}"

A special variable "MachVar" is set to the name of an icon dropped
into the clock window. A macro could then use that name. For example:

EXE_CMD"show ${MachVar}"

See Clock Displays, for more about "MachVar" and AppWindows.

Other possible uses of variables:

PAU_FIX_INPUT"1,clipunit" CLP_PASTE"${clipunit}"

This pauses for you to enter one number. That number is then set to
the variable "clipunit" which is then used as an argument for
CLP_PASTE. Using the above macro, you can paste from clipboard units 0
- 9.

PAU_VAR_INPUT",file" EXE_CMD"${editor} ${file}"

This pauses for a variable number of keystrokes which are stored in
the variable "file" when you press <ESC>. The program whose name is
stored in "editor" is then run using the variable 'file' as an
argument. Note the ',' in ",file". The first argument (the input
termination key) is optional, it defaults to <ESC> (hex 45). This could

have been written:

    PAU_VAR_INPUT"45,file"

    During a PAU_VAR_INPUT or PAU_FIX_INPUT function, characters are
displayed in the clock as they are entered. Ctrl-H may be used to
correct mistakes.

    Note that the variable name is used without the ${} when using it as
an argument in a setting type function. If ${} were used, then the
value of the variable would be used instead.

    EXE_CMD"ttx ${MachVar}"

    This will run the editor ttx with the arguments from the variable
"MachVar". This variable is set when an icon is dropped into the clock.

    From within text in a macro:

    "The contents of var is ${var}"

    "This is the username environment var ${username}"

    To use ${} without expanding it, you must escape the $ with a \.

    "This is the variable \${var}"

    Using variables lets you change a setting in just one place. For
example, in your user-startup file you could set an overscan width
variable:

    SetEnv OSWidth 680

    And then use it in your NewShell command:

    EXE_CMD"NewShell con:0/0/${OSWidth}/200/Shell"

    Or,

    SetEnv shell NewWShell

    EXE_CMD"${Shell} con:0/0/${OSWidth}/200/${Shell}/CLOSE"

    There is also the variable machv_recursion_depth which limits the
number of times a macro may call itself. It defaults to 10. This
variable is either global or local, depending on your setting in
Miscellaneous Features.

  Global and Local Environment Variables

1.62 MachV.guide/Global and Local Environment Variables

Global and Local Environment Variables
======================================

    In the Miscellaneous Features window you may set certain parameters
used by MachV when handling variables.

    If "Local/Global Vars" is selected then expanding a variable will
first try a local variable and then a global variable. You could
instead select either "Global Only Vars" or "Local Only Vars".

    Both PAU_FIX_INPUT and PAU_VAR_INPUT functions and the SET_VARIABLE
function can accept a third argument which is the hexidecimal
representation of these options.

    The values for these options are:

        Local then global   -   0
        Global   -   100
        Local   -   200
        Save Var   -   1000

    SET_VARIABLE"test,abcd,1100"

    This would set the global variable "test" to "abcd" and also save it
to ENVARC:. Only global variables may be saved.

    PAU_VAR_INPUT"45,test,100"

    This waits for variable length input terminated with ESC (45) and
then sets the global variable 'test' to that input.

    Also in the Miscellaneous window, you may select "Save Vars". This
will cause all global variables to be saved to ENVARC: when a
PAU_XXX_INPUT or SET_FUNCTION is used.

1.63 MachV.guide/Clipboard Features

Clipboard Features
****

    The clipboard is a special software device used by many
applications. When you perform a copy or cut in the application, it
copies the text to the clipboard. This text is then available both to
the application that copied or cut it and to any other application that
uses the clipboard.

    MachV has several functions that enhance this feature. You may view
the contents of the clipboard, paste the clipboard to an application's
window that does not support the clipboard, print the clipboard
contents and you can transfer one unit to another.

    The clipboard supports multiple buffers or "units", 0 - 255. All
functions have an optional argument that specifies the clipboard unit

to use. If you do not provide the argument, unit 0 or the last unit
selected will be used.

   The clipboard functions all require that the iffparse.library is in
your libs: directory.

  Viewing The Clipboard
  Pasting The Clipboard
  Printing The Clipboard
  Transfering To Other Units
  Clip Features

1.64   MachV.guide/Viewing The Clipboard

```
Viewing The Clipboard
=====================
```

   This opens a window and displays the contents of the clipboard. From
this viewer window, the unit number may be changed by entering a number
in the string gadget, by clicking on the Next/Prev gadgets, by pressing
a function key from <F1> – <F10> which will show units 0 – 9, or by
pressing the <Shift-Up> and <Shift-Down> arrow keys. Close the viewer
window with either the close gadget or by pressing <ESC>. Pause the
display scrolling by pressing any key – stop it by pressing <Return>.
The viewer window will automatically update itself when the contents of
the currently displayed unit have changed.

   At the beginning of a clip, the clipboard viewer displays:

```
    Clipboard Unit # U   (nnnnnn bytes)
    ---------------------------------
```

   "U" is the unit number and "nnnnnn" is the clip size.

   You may enter a clipboard unit number in the "Copy To ->#" gadget to
copy the current unit to another unit. Press <Tab> to activate the
string gadgets.

   The "Load" gadget allows you to load a file to the unit currently
displayed.

   The "Save" gadget allows you to save the unit currently displayed to
a file.

   Both of these will open a file requester for you to select from. If
you preceed the file name with a '$', you may load or save an
environment variable.

   You may append a clip to a file by specifying a file name preceded
with ">>" in the ClipViewer file requester. Select the desired
directory and enter ">>filename".   _____
Drawer |ram: | ~~~~~~~~~~~~~~~~~~~~~~~~~~
        _____   File |>>dummy |

~~~~~~~~~~~~~~~~~~~~~~~~~ If the file does not exist, it will be created.

The clipboard viewer is also an AppWindow. This means that you can drop icons into it and they will be immediately copied to the clipboard and displayed in the viewer.

See Transfering To Other Units.

Because the clipboard viewer is a "console" window, you may use the mouse to highlight text and press <Right-Amiga-C> to copy it to the clipboard. The text may be edited, highlighted and then copied to the clipboard. The editing features are rather limited. Press "Help" to show the editing keys.

The view window may be kept opened if desired and zoomed to a large or small size. MachV will function normally while it is open.

You could quickly view the clipboard and close the viewer with a macro like this:

CLP_VIEW PAU_SECONDS"1" WIN_SEND_CLOSE"A"

The Clip Viewer executes the system program "ConClip" when changing units. This keeps console window copying and pasting in sync with the viewer. If you change units often, it is recommended that you make this program resident with:

Resident >nil: c:ConClip pure add

just before the line that reads "c:ConClip" in your startup-sequence. This will reduce disk access when changing units.

You may also change the current unit with CLP_CURRENT_UNIT"N" where N = the desired unit number.

Note

Some programs do not correctly handle the clipboard device. If the viewer refuses to display a unit, it was caused by improper usage by an application. At a minimum, the clipboard viewer should show:

```
    Clipboard Unit # 0  (     0 bytes)
    ---------------------------------
   ** This Unit Is Empty **
```

If it doesn't, the only recourse usually is, unfortunately, to reset the computer.

1.65 MachV.guide/Pasting The Clipboard

Pasting The Clipboard
=====================

The function CLP_PASTE will paste the contents of the clipboard to
the active window. You would normally use this to paste to a window
that does not provide clipboard support, or to one that only supports
unit #0 and you want to paste from a different unit. You can even paste
to a string gadget!

This function sends the text as keystrokes so it is not as fast as a
normal paste. You may choose to not send an 'up' keystroke for each key
sent by not checking "Send Up Keys". When checked, an 'up key' message
is sent for each 'down key' that is sent from a macro. This doubles
the time required to send keystrokes, but some programs expect to
receive key up messages.

The two strings in the gadgets "Prepend" and "Append" will be sent
before and after each line. The delay between characters and lines sent
are of interest mainly to telecommunication users. They are in 50ths of
a second.

You can remove linefeeds when pasting a paragraph by checking
"Remove Line Feeds" in SetMachV and you can send both a return and line
feed for each line feed by checking "Line Feeds To CRLF".

To abort pasting, press <ESC>.

1.66 MachV.guide/Printing The Clipboard

Printing The Clipboard
======================

The function CLP_PRINT will print the contents of the clipboard.
Most word processors provide this ability, but now you can print text
from the shell. In a shell, you can select text with the mouse and
press <Right-Amiga-C> to copy it to the clipboard. CLP_PRINT could then
be used to print it, or CLP_PASTE to paste it.

You may also click on "Print" in the clipboard viewer.

To abort printing, press <ESC>.

1.67 MachV.guide/Transfering To Other Units

Transfering To Other Units
==========================

The function CLP_TRANSFER will transfer the clipboard contents from
one unit to another. This can be extremely useful when your application
supports only unit 0. You could create one or more macros that copy from
unit 0 and one or more that copy back to unit 0. This will allow you to
copy and paste from multiple buffers. For example:

```
CLP_TRANSFER"0,1"
     Transfer unit 0 to unit 1.

CLP_TRANSFER"1,0"
     Transfer unit 1 to unit 0.

CLP_TRANSFER"0,2"
     Transfer unit 0 to unit 2.

CLP_TRANSFER"2,0"
     Transfer unit 2 to unit 0.
```

You could follow these with CLP_VIEW to verify the transfer:

```
CLP_TRANSFER"0,1"  CLP_VIEW
```

or close the viewer in one second:

```
CLP_TRANSFER"0,1" CLP_VIEW"1" PAU_SECONDS"1" WIN_SEND_CLOSE"Clip
Viewer"
```

You may also tranfer to or from a file or environment variable and the clipboard by using the file or variable name.

```
CLP_TRANSFER"ram:temp,0"
     Transfer file named ram:temp to unit 0.

CLP_TRANSFER"$username,0"
     Transfer environement variable username to unit 0.

CLP_TRANSFER"0,$TERM"
     Transfer unit 0 to variable "TERM".

CLP_TRANSFER"0,>>capture"
     Append unit 0 to the file "capture".

CLP_TRANSFER"1,dh0:texts/scratch"
     Transfer unit 1 to file "dh0:texts/scratch".
```

Notes:

If the file name starts with a digit, a clipboard unit will be assumed.

Currently, only text of the IFF type FTXT may be viewed. ILBM support may be added later.

1.68 MachV.guide/Clip Features

```
Clip Features
=============
```

The window "Clip Features" in SetMachV contains some options that affect text pasting using CLP_PASTE.

```
Prepend Chars
Append Chars
     These are string gadgets that contain characters to send before
     and after text pasting.

Char Delay
Line Delay
     These are delays in 50ths of a seconds.

Remove Line Feeds
     Removes the end of line characters so you may paste a paragraph to
     a word processor.

Line Feeds To CRLF
     Send a carraige return before the line feed.

Send Up Keys
     This will send the up key message after each key down is sent.
     This is not normally needed, but some programs may require it.
     When selected, text being pasted takes almost twice as long to
     paste.
```

1.69   MachV.guide/Mouse Features

```
Mouse Features
**************
```

   All mouse features are adjusted or toggled on or off in this section
of SetMachV. Some features may also be modified with macros.

   To open the window, click on the "Mouse Features" gadget or press
<M>.

   Many mouse features have an associated string gadget in which you
may enter the name of a window or an AmigaDos pattern. This can be used
to limit when a function can be used. For example, setting ~(MachV#?)
in the SunMouse gadget will prevent SunMouse from activating the clock
(or any other window that begins with MachV).

```
  Mouse Accelerator
  Mouse Off
  SunMouse
  Auto PopTop
  Key Activate
  Window Cycling
  Click To Front
  Mouse To Menu
  Menu Wrap
  Button Swap
  Alternate Mouse Port
  Middle Mouse Button
  Mouse Freeze
```

```
Playback Delay
```

1.70   MachV.guide/Mouse Accelerator

```
Mouse Accelerator
=================
```

   The mouse speed and threshold may be adjusted. "Speed" is an
acceleration factor applied to mouse movements. The "Threshold" adjusts
the sensitivity of the mouse (how fast the mouse must move before the
accelerator kicks in).

   Low values decrease the sensitivity. Set "Speed" to 0 to disable the
accelerator.

   The accelerator also allows you to use negative values to slow it
down. This is sometimes useful for fine control in painting programs.
The values may range from -9 to 9. The speed may also be set by using
the macro function MOU_SPEED.

1.71   MachV.guide/Mouse Freeze

```
Mouse Freeze
============
```

   You can freeze vertical and horizontal mouse movements by executing
the macros:

   MOU_FREEZE_VERT and MOU_FREEZE_HORZ

   This is helpful when attempting to line up icons for example.

   An interesting possibility is to effectively "lock out" other users
by executing both macros and a MSC_FREEZE_KEYS function. If someone
does not know the hotkey combination, he would be unable to move the
mouse or type. The unauthorized user could reboot, but you could use
this macro which is an auto executing macro:

   EXE_AUTO"" FREEZE_VERT"Tog" FREEZE_HORZ"Tog" FREEZE_KEYS"Tog"

   You would need to manually execute this macro to unfreeze the mouse
and keyboard.

1.72   MachV.guide/Mouse Off

```
Mouse Off
=========
```

   You can have the mouse disappear after a set number of seconds or
when you press any key (except qualifiers). This is useful when typing
- the mouse stays out of the way.

   Set "Mouse Off" to 0 to disable this feature. You can set the delay
time with MOU_OFF in a macro.

   Notes:

   1. The mouse will always briefly appear and then disappear when a
      window opens.

   2. The mouse will reappear if you use the Left Amiga Arrow keys to
      move the pointer.

   3. This feature seems to offset the hotspot of the pointer by one
      pixel when using an Amiga 1200 or 4000. If this is a problem, set
      "Mouse Off" to 0.

1.73 MachV.guide/SunMouse

```
SunMouse
========
```

   You can activate a window by simply moving the mouse into it - no
need to click. This is called "SunMouse", named for the way in which
mice behave on Sun workstations.

   To prevent windows that the mouse is merely passing over from
becoming active, a window is not activated until the mouse comes to a
stop. The SunMouse does not activate a window if a string gadget is
active.

   The "SunMouse" may be turned on and off with MOU_SUNMOUSE in a macro.

1.74 MachV.guide/Key Activate

```
Key Activate
============
```

   Instead of (or in addition to) using the SunMouse feature, you may
select "Key Activate" so the window that is under the mouse will become
active when you press a key.

1.75   MachV.guide/Auto PopTop

```
Auto PopTop
===========
```

   A window pops to the top when it becomes active if this is set.

   Note!

   This may cause unexpected results if you click on a window's depth
gadget to bring it to the front - it will be brought to the front by
Auto PopTop, but then releasing the mouse button will cause the depth
gadget to send it to the back.

1.76   MachV.guide/Window Cycling

```
Window Cycling
==============
```

   This provides an easy way to move windows and screens to the back.

   With the mouse over a window, press and hold the left button and
press and release the right button. The window will move to the back.

   The new top window will also become the active window.

   A screen may be moved to the back in these situations:

  1. If the mouse is over a "backdrop" window. (This type of window
     cannot be depth arranged.

  2. There is only one window on the screen.

  3. The mouse is not over a window.

1.77   MachV.guide/Click To Front

```
Click To Front
==============
```

   You can bring a window to the front by clicking anywhere in it a set
number of times. The number in the cycle gadget may be from 0 to 4. 0
disables this feature.

   Note:

   You may experience problems with some programs with this feature set
to 1 or 2. This is especially true when double clicking on icons. The
only solution to this is to set it to 0,3 or 4.

1.78   MachV.guide/Mouse To Menu

```
Mouse To Menu
=============
```

   When this feature is on, every time you press the right mouse button
the pointer will jump to the menu bar. When you release the button it
will return to where it was.

   This feature may also be changed with MOU_TO_MENU in a macro.

1.79   MachV.guide/Menu Wrap

```
Menu Wrap
=========
```

   With this option selected the mouse will "wrap" to the leftmost menu
item when the mouse moves to the right of the rightmost menu. It will
wrap around to the right side when you move to the left of the first
menu.

   Note:

   There are a few programs that, because of their menu layout, you may
not be able to select some items. Your only option is to disable "Menu
Wrap".

1.80   MachV.guide/Button Swap

```
Button Swap
===========
```

   Selecting this will swap the left and right buttons. This also means
that the Window Cycling combination is also swapped. You will need to
use Right/Left instead of Left/Right.

   When recording a macro, a right button will be recorded and played
back as a left button. This allows a macro to be played back correctly
even if you turn "Button Swap" off after recording.

1.81   MachV.guide/Alternate Mouse Port

```
Alternate Mouse Port
====================
```

   If you wish to use port 1 instead of port 0 for mouse input, you may

change it with "Mouse Port" or with the SET_MOUSE_PORT function.

    You could plug a trackball or other input device into port 1 and
easily switch between the two.

    Note:

    When changing ports in Mouse Features, once you select another port,
you must use a device in that port to select back. Or, use <Left-Amiga
Left-Alt> to simulate a left button click on the gadget.

1.82 MachV.guide/Middle Mouse Button

Middle Mouse Button
===================

    Some third party mice have a middle button that may be put to good
use. While most users prefer to have the middle button behave like the
<Shift> key, it can function as any qualifier that you want. It may, of
course, also have a macro attached to it.

    To change the middle mouse button to a qualifier, simply select the
desired qualifiers in the "Mouse Features" window.

1.83 MachV.guide/Playback Delay

Playback Delay
==============

    When recording and playing back mouse movements, you may set a delay
in this gadget. The values are in 50ths of a second. See
Recording All Moves, for more information.

1.84 MachV.guide/Clock Features

Clock Features
****

    To open this window, click on the "Clock Features" gadget or press
the letter <L>.

  Clock Displays
  Clock Borders
  Clock On and Off
  PopTop
  Float

```
Gadgets
Clock To Screen
Clock Gadgets
Clock Alarms
Online Meter
Low Memory Warning
Alternating Date and Memory Display
Clock Position
Clock Size
Time Position and Color
Horizontal and Vertical Clocks
12 and 24 Hour Time
Clock Font
Date Format
Custom Messages
```

1.85 MachV.guide/Clock Displays

```
Clock Displays
==============
```

   You may select from a wide variety of clocks by clicking on the
gadget that cycles through the available types. These are shown below.

```
        Clock+Memory
        _____

        |Chip  456 Fast  357  4:32:45|
        ------------------------------
        Memory Only
        _____

        |Chip  456 Fast  357|
        --------------------
        Clock+Date
        _____

        |05-01-92  12:32:45|
        -------------------
        Clock+Date (DOS Format)
        _____

        |05-Apr-92 12:32:45|
        -------------------
        Clock Only
        _____

        |12:32:45|
        ----------
        Coordinates
        _____

        |XXX  YYY|
        ----------
```

   Selecting "Sec." will display seconds with the time. The clock will
adjust its width if "Gadgets" is not selected. ("Sec." is located at
the bottom of the "Clock Features" window.)

   Besides showing time and memory, the clock also displays many helpful

messages such as "Select Hotkey...", "Recording Macro...", "Executing
Macro..." and "Macro Aborted!".

   The "Gadget Bar" clock is a special type. It is not really a clock
at all but a clock-like window in which gadgets may be placed. Macros
may be executed by clicking on their gadgets. See Clock Gadgets, for
more information.

   See Clock Borders, for changing the border.

   The clock window is an AppWindow in which you may drop icons. Icons
dropped into this window will set the environment variable "MachVar" to
the full path and name of the icon. If multiple icons are dropped, the
variable will contain all names separated by a space. e.g.

   dh0:src/app/main.c ram:readme

   This may be very useful when used with a macro (which may have a
gadget) such as:

   EXE_CMD"view80 ${MachVar}"

   You may want to delete or unset this variable afterwards:

   EXE_CMD"view80 ${MachVar}" Set_Variable"MachVar,"

   See Environment Variables.

1.86 MachV.guide/Clock Borders

Clock Borders
=============

   Clocks that have gadgets can modify the border somewhat. Select
these in "Clock Features":

   * Right Border The size gadget uses the right border.

   * Bottom Border The size gadget uses the bottom border.

   * Borderless The clock has no border.

1.87 MachV.guide/Clock On And Off

Clock On And Off
================

   The "Clock" gadget in the "Clock Features" window turns the clock on
and off.

Within a macro, a clock may be toggled with CLK_CLOCK, sent to the
front screen with CLK_TO_FRONT and the next screen with
CLK_TO_NEXT_SCREEN.

1.88 MachV.guide/PopTop

```
Clock To Top Window
===================
```

You may select to have the clock pop to the top window by selecting
"PopTop". Whenever a window opens on top of the clock, the clock will
move to the top.

This is not the same as Clock To Screen.

1.89 MachV.guide/Float

```
Clock Float
===========
```

If you are using a virtual Workbench screen (one that is larger than
the display area), the screen can automatically be scrolled with the
mouse. This is a feature of Workbench. The clock would normally remain
in its original position but if you select "Float", the clock will keep
its position relative to the display's left and top edge.

1.90 MachV.guide/Gadgets

```
Gadgets
=======
```

Selecting this will allow the clock to accept macro gadgets. Any
macro with the Clock Gadgets flag set and a name will have a gadget in
the clock.

See Clock Gadgets.

1.91 MachV.guide/Clock To Screen

```
Clock To Screen
===============
```

The clock may also move to any screen by entering "#?" or "(#?||)"

in the string gadget under the "Clock Type" gadget. You may use any
AmigaDos pattern that you wish. Entering nothing in this gadget will
keep the clock on the Workbench screen. "(Workbench#?|Final Copy#?)"
will let it open only on the Workbench or Final Copy screen. Two
vertical bars '||' will match untitled screens.

                                    CAUTIONS

     These must be used with some caution as not all programs work
properly with another window on their screen. Some programs refuse to
close their screen while the clock is on it. Some programs terminate
but leave their screen behind. If you feel adventurous, you can close a
screen that has been left behind with SCR_KILL_SCREEN in a macro.
Killing a screen may cause a system crash if the program owning the
screen still expects it to be there.

     This option also replaces the CloseScreen vector. You may experience
some incompatibility if you, for example, run another program that also
replaces this vector, run MachV and then terminate the first program.
The general rule for running programs that do this is to terminate them
in the reverse order that they were run.

     When a program refuses to close its screen if another window is on
it, you could create this macro: CLK_TO_NEXT_SCREEN "\RQ", and in
Flags, set the "Non Recursive" flag and attach it to <Right-Amiga-Q>.
This will put the clock on another screen and then pass <Right-Amiga-Q>
on to the program. You may also need a PAU_SECONDS"n" at the end of
this macro.

1.92   MachV.guide/Clock Gadgets

Clock Gadgets
=============

     After you become more familiar with macros, you may wish to create
gadgets to execute them. The section assumes that you understand how
create and use macros.

     A macro gadget is a gadget in the clock that you may click on to
execute the macro. These gadgets may display the macro's name or it may
use a custom image created with an icon editing program. The name of
the icon should be the same as the macro name plus ".info".

     Any clock type can accept macro gadgets if the Gadgets gadget is
selected. A macro will have a gadget in the clock if it has a name and
if the "Clock Gadget" flag gadget in the Flags window is selected.

     A "dock" type window can be created with gadgets executing many
functions before running the desired program. See the configuration
file dock.prefs" for an example.

   Adding Gadgets
   Global Gadgets

```
  Custom Images
  Moving Gadgets
  Updating the Gadgets
  Using the Gadgets
  Gadget Clock
  Helpful Hints
```

1.93   MachV.guide/Adding Gadgets

```
Adding Gadgets
--------------
```

   Adding a gadget is simply a matter of selecting the "Clock Gadget"
gadget in the Flags window. The desired macro to be "gadgetized" must
be the current one (the macro must be in the "Macros" string gadget)
and it must have a name.

   The gadget will be immediately added to the clock. MachV will
attempt to place it in a reasonable location, but see Moving Gadgets.
The window will be resized vertically if necessary to enable the gadget
to be visible.

1.94   MachV.guide/Global Gadgets

```
Global Gadgets
--------------
```

   If you want a gadget to appear in all configurations, you may create
it in the first configuration as a "No Key" macro and select both the
"Gadget" flag and the "Global" flag.

   The gadget's position will be the same in all configurations.

   Macros attached to a key and also have a gadget will not appear in
other configurations.

1.95   MachV.guide/Custom Images

```
Custom Images
-------------
```

   If there is an icon with the same name as the macro in the same
directory as your MachV.prefs file, the icon's image will be used
instead of the macro's name. For example, if you have a macro named
"ClipView" and an icon named "ClipView.info" in the same directory as
your MachV.prefs file, the icon's image will be used.

1.96 MachV.guide/Moving Gadgets

```
Moving Gadgets
--------------
```

   Gadgets may be placed anywhere in the clock. Hold the <Alt> key and
click on the gadget (release the left mouse button). Then while still
holding the <Alt> key down, move the gadget outline to the desired
location and release the <Alt> key.

   The position may be more precisely set either in the Flags window of
SetMachV or in the Quick Edit window. See Quick Edit.

1.97 MachV.guide/Updating the Gadgets

```
Updating the Gadgets
--------------------
```

   In order to reduce disk access, MachV does not check for .info files
when changing screens and reopening the clock window. Instead, it
checks only when first run and when you click on the "Clock Gadget"
flags gadget or the "Gadgets" checkbox in Clock Features.

   It also updates whenever a file is written to the MachV.prefs
directory. It does this so you can change the image using IconEdit and
see the results immediately in the clock.

1.98 MachV.guide/Using the Gadgets

```
Using the Gadgets
-----------------
```

   Simply clicking on the gadget will execute the macro. If the macro
requires an active window to act upon or for output, the window that
was previously active before clicking on the gadget will be used.

   The clock can be "zoomed" to a larger size to expose more gadgets
and then zoomed back to normal size.

   In SetMachV, after clicking on "Select Hotkey", you may click on a
gadget to make it current.

1.99 MachV.guide/Gadget Clock

```
Gadget Clock
------------
```

A clock with "Gadgets" checked or "Gadget Bar" selected will also
have a zoom and sizing gadget. The size may also be set with the
"Width" and "Height" gadgets in "Clock Features". These gadgets are
disabled if you do not have "Gadgets" or "Gadget Bar" selected.

You may enter -1 in the Left Edge, Top Edge, Width and Height
gadgets and the current values will be entered for you when you press
<Return>.

The letter "Z" will replace the first character in the clock display
when the window is in its zoomed state.

1.100   MachV.guide/Helpful Hints

Helpful Hints

If you want to have very many gadgets in the clock, it is best to
use the "zoom" gadget to expand and contract the window. Resizing it
with the size gadget means that you will have to resize it back
manually.

A clock with gadgets has a "normal" size and position, and an
alternate "zoomed" size and position. MachV will remember its size in
each state.

To save your "zoomed" window size and position, save your settings
with the clock in its zoomed state.

If you want to create gadgets for all or most of your macros, you
can set the "Clock Gadget" gadget in the Flags window and then click on
"Set All". This will create gadgets for all macros that have names.

You may want to create a zoom macro for the clock. Here is an
example that will also ensure that the window is up front:

    WIN_TO_FRONT"MachV" WIN_ZOOM_WINDOW"MachV"

1.101   MachV.guide/Clock Alarms

Clock Alarms
============

Although you may create auto-executing macros by entering the
EXE_AUTO function in a macro, a simple interface for creating alarms
can be found in the "Clock Features" window.

Under "Alarms" are the gadgets "Create/Modify", and "Delete".
Clicking on either of these will prompt you to enter or select a name

from a list of macro alarms.

   All alarms are placed in the first configuration. They will be "NO
KEY" macros. They will have the 'Auto Execute' and 'Global' flags set.

  Date and Time
  Alarm Message
  Beeps
  Say Text
  Later
  Auto Delete
  Auto Save
  Delete

1.102   MachV.guide/Date and Time

Date and Time

   You may select the days or date of the alarm, but not both. You may
also set the time or repeat interval. If you enter the date instead of
checking the days, you may use the same AmigaDOS wild cards as for
EXE_AUTO functions. For example:

   ??-23-92 Will execute on the 23rd of every month.

   You should one date format. The format that was used to create the
alarm must be the same as when the alarm executes. Also, the date
separator must be the same as entered in the "Sep." gadgets, or a "?"
may be used, such as 12?03?94.

   You may also use wild cards in the time:

   ??:15 Will execute at 15 minutes past any hour.

   Enter only the hour and minutes separated by a colon. Do not use
seconds.

   Instead of entering the time, you may check the "Repeat" box and
enter the number of minutes in the "Minutes" gadget. You may not select
both time and repeat.

1.103   MachV.guide/Alarm Message

Alarm Message

   The desired message may be entered in the long string gadget above
the "Say Text" and "Show Text" gadgets. Up to 80 characters may be used.

Below this you may select Say Text and "Show Text".

If you select "Show Text", a requester will be displayed when the
alarm executes. You will then be able to click on "Okay", "Later" or
"Remove".

Click on "Okay" to shut off the alarm (closes the requester).

Click on "Later" to have the alarm to come back in 5 minutes.

Click on "Remove" to delete the macro and save your configuration
without the alarm macro.

Change the "come back later" delay value in the "Later" gadget. This
gadget is disabled unless "Show Text" is checked.

If you have selected "Auto Delete", only the "Okay" gadget will be
in the requester.

If you have selected "Auto Save", your configuration will be saved
after the alarm executes. This would be used together with "Auto
Delete" to save after deleting the alarm.

1.104 MachV.guide/Beeps

```
Beeps
-----
```

You may choose to have a number of beeps instead of or in addition to
speaking and showing of a message. Simply enter the number of beeps
desired in the "Beeps" gadget.

1.105 MachV.guide/Say Text

```
The Say Command
---------------
```

The command used to speak the message may be changed if desired. The
default is "echo >speak:". You could use:

echo >speak:opt/p65/s125

or

say FEMALE PITCH 200

or execute some entirely different program.

1.106 MachV.guide/Later

```
Later
-----
```

When an alarm executes and you have "Show Text" checked, you may
click on "Later". The alarm will execute again a set number of minutes
later.

Change the "come back later" delay value in the "Later" gadget in
"Alarms" interface window. This gadget is disabled unless "Show Text"
is checked.

When an alarm puts up a requester AND if "Later" is > 0 AND if you do
not click on any gadget in the alarm requester, the alarm will
re-execute when the "Later" time expires. The message will be spoken
again if "Say Text" is selected. It will beep again if "Beeps" were
greater than 0. "Show Text" must be selected for this feature to
function.

In other words, if the alarm puts up a requester and you do not
respond to it, the alarm will execute again when the "Later" time
expires. A new requester is not put up.

If you enter 0 in the "Later" string gadget it will not come back
unless you click on the "Later" requester gadget which will cause a
default 5 minute delay.

Note that you cannot make an alarm repeat or come back without "Show
Text" seleted because there needs to be a "Later" gadget to click on.

1.107 MachV.guide/Auto Delete

```
Auto Delete
-----------
```

Selecting this will cause the alarm to delete itself after
executing. Your configuration without the alarm will be saved if you
also select Auto Save.

This option simply adds a MAC_DELETE function to the alarm macro.

1.108 MachV.guide/Auto Save

```
Auto Save
---------
```

Selecting this will save your configuration after the alarm
executes. This is usually used in combination with Auto Delete.

```
This option adds a CFG_SAVE function to the alarm macro.
```

1.109 MachV.guide/Delete

```
Delete
------
```

   Clicking on "Delete" will prompt you to enter or select a name from
a list of macro alarms. The macro will then be deleted. It will remain
in your MachV.prefs file until you save your settings.

1.110 MachV.guide/Online Meter

```
Online Meter
============
```

   You may select to have the time portion of your clock keep track of
your online costs by selecting "Meter". Clear your costs with "Meter
Reset". Put the online rate (cents per hour) in the "Rate" gadget.

   You may use CLK_SET_RATE in a macro to set the rate and
CLK_RESET_METER to set the meter to 0.

   If the rate is set to 0, the meter will measure in time instead of
dollars. This is useful if you need to keep track of how much time you
are spending on a project. The time meter does not run when the screen
is blanked. (You are not working then, are you?)

1.111 MachV.guide/Low Memory Warning

```
Low Memory Warning
==================
```

   If you would like to be alerted to low memory situations, enter the
memory warning level in the "Low Mem" gadget. When available chip
(graphics) memory drops below this value (in kilobytes), the clock's
text color will change. The color will depend what you have selected
for the normal text and background color. See Time Position and Color.

1.112 MachV.guide/Alternating Date And Memory Display

```
Alternating Date And Memory Display
==================================
```

   If you want the clock's display to alternate between memory and
time, and the date, enter the number of seconds to show the memory and
time in the "Date/Time" gadget.

1.113 MachV.guide/Clock Position

```
Clock Position
==============
```

   You can change the clock's position by entering values in the "Left
Edge" and "Top Edge" gadgets. An alternate method is to position the
clock with the mouse and then enter a -1 in these gadgets. The current
values will then be entered for you. Turn off Float when doing this.

1.114 MachV.guide/Clock Size

```
Clock Size
==========
```

   You only can change the size of the clock if "Gadgets" or the clock
type "Gadget Bar" is selected. You may set these values in "Width" and
"Height". Otherwise, these gadgets are disabled.

1.115 MachV.guide/Time Position and Color

```
Time Position and Color
=======================
```

   The left and top edge of the time may be set in the "Time" section.
These are normally 3 and 1, but if you have gadgets in your clock, you
may wish to position the time someplace else.

   The text and background color may also be changed to your liking
with the two palette gadgets. These will display the number of colors
of the screen that the clock is currently on. The actual colors will be
those of the screen that SetMachV is on.

   If you change the number of colors of the Workbench screen, you may
need to re-run MachV to update certain visual attributes.

1.116   MachV.guide/Horizontal and Vertical Clocks

```
Horizontal and Vertical Clocks
==============================
```

   You can select "Vert" or "Horz" to change the clock's orientation. In
some applications users find it convenient to place the clock in the
left border area.

   You can also change the orientation by activating the clock with the
mouse and pressing <Alt-H> or <Alt-V>.

1.117   MachV.guide/12 and 24 Hour Time

```
12 and 24 Hour Time
===================
```

   Select 12 Hr or 24 Hr to show the desired format.

1.118   MachV.guide/Clock Font

```
Clock Font
==========
```

   You may select the font for the clock to use by entering it in the
"Clock Font" gadget. Enter the font name (with or without out the .font
extension) and the point size. The following are all valid:

```
        Opal 12
        Diamond 15
        TermISO.font 8
        pearl, 8
        topaz.font, 11
```

   If you want to use the same font that is used for the screen, enter
nothing.

   The diskfont library must be available to use this feature. Only
non-proportional fonts are recommended,

   The font selected will also be used in the text of any clock gadgets
in the clock.

   Each configuration may have its own font.

1.119   MachV.guide/Date Format

```
Date Format
===========
```

The date is available in four formats selectable in the "Clock
Features" window. As you cycle through these, the date will be
displayed above the gadget. You may also choose your desired separator
character in the adjacent string gadget labeled "Sep.".

The date format selected will be used in the clock and in
TXT_DATE_PASTE and TXT_DATE_PRINT macros functions.

These two features may also be changed with the Set Functions:

SET_DATE_FORMAT"DOS/INT/USA/CDN"

SET_DATE_SEPARATOR"-"

The date format and separator character must be the same as is used
in any clock alarms that you may have.

1.120 MachV.guide/Custom Messages

```
Custom Messages
===============
```

Many messages are displayed in the clock. They range from error
messages to prompts to custom messages.

In a macro you can display custom messages with:

CLK_CUSTOM_MSG"message"

This could be combined with other functions like:

MOU_SET_PORT"1" CLK_CUSTOM_MSG"Using Port 1"

See Macros, for additional information.

These messages may also be spoken by selecting Speak Messages in the
Miscellaneous Features window. The device "Speak:" must be mounted.

1.121 MachV.guide/Miscellaneous Features

```
Miscellaneous Features
**********************
```

These features are accessible from the "Miscellaneous Features"
window.

```
     Priorities
     Screen Blanking
     Macro Screen Blanking
     Splines
     Password
     Suspend
     No Caps Lock
     Speak Messages
     Beep
     Hold Qualifiers
     Debug
     Trace
     Gameport
```

1.122 MachV.guide/Priorities

```
Priorities
==========
```

   All tasks in the Amiga's multitasking operating system run at a
particular priority. Higher priority tasks have higher numbers and will
run while tasks with lower priorities wait. Most user tasks run at a
priority of 0 and have an equal share of time. MachV runs well at 1.
MachV also has the ability to change its priority temporarily while
playing back a macro. A priority of 1 is usually okay here too. You may
experiment with these numbers and your particular applications to find
what works best. Don't try to set them very high though, usually no
more than 6 and certainly no more than 10. By setting the "Playback
Priority" to 0, the application and MachV will alternate between
sending and receiving of keywstokes.

   The playback priority may also be temporarily changed from a macro
with MSC_PRIORITY. Using this you can run an application at a specific
priority. For example, this macro will run the program at a priority of
1:

   MSC_PRIORITY"1" EXE_CMD"Run FinalCopy"

   The priority of the input handler may also be set. Valid priorities
are 51 - 127. 55 is recommended, but if you have other programs with
their own input handler, you may want to change this. Changes will not
take effect until you save and run MachV again. Note that this is NOT
the same thing as task priority. This is the priority of the handler
that intercepts input before intuition has a chance to see it. It must
be set above intuition which is 50.

1.123 MachV.guide/Screen Blanking

```
Screen Blanking
===============
```

   After a preset length of time, you may have the screen either go
blank or just dim. If you select a dimness of 0 and have Splines on,
there will be a line pattern drawn on the blanked screen.

   If you select a dimness value of 1-15, the front screen will go dim.
This uses no extra memory because it doesn't open is own screen.

   The screen may also be blanked with MSC_BLANK_SCREEN in a macro.
Executing this macro will immediately blank the screen. The dimness and
spline settings may also be changed in this macro. See MSC_BLANK_SCREEN.

   By selecting "Quick Blank" and setting some positive value for the
Mouse Off feature, the screen can be blanked when the mouse turns off
by first moving the mouse to the upper left corner.

   You may also choose to require a password before the screen will
unblank. See Password.

   If the screen cannot be opened because of low memory, the front
screen will be set to black.

   If another screen opens up while the screen blanker is on, the
blanking screen will be closed.

   Instead of using these methods to blank the screen, you may use an
external blanker or even run some animation. This is done by
Macro Screen Blanking.

1.124   MachV.guide/Macro Screen Blanking

```
Macro Screen Blanking
=====================
```

   If you wish, you can execute a macro named "Blank" instead of
blanking the screen. With this feature, you could run an animation or
any other program. In addition, you could send a message to a custom
screen blanker telling it to blank. The function:

MSC_PUTMSG"PortName,xx,xx,xx"

   could be used for this. A sample screen blanking program is
included. This requires some knowledge of the "C" programming language.

   If a macro was executed instead of the normal screen blanking, a
macro named "Unblank" will execute when the user presses a key or moves
the mouse. This could be a macro that tells another program to quit or
unblank. These macros are examples:

```
     Name          Macro
     ----          -----
```

```
    Blank            EXE_CMD"Run movie animfile"
    Unblank          WIN_SEND_CLOSE"MovieWindowTitle"

      or maybe just

    Unblank          WIN_SEND_CLOSE"A"
```

A macro named "MachVQuit" will execute when MachV is terminated. This could be used to send a message to a screen blanker telling it to quit.

1.125 MachV.guide/Splines

```
Splines
=======
```

When you have selected 0 in "Blanking Dimness", and you have "Splines" on, screen blanking will include line pattern drawing.

   Spline features:

  1. Runs as a separate task at -1 priority. If you have other tasks
     running, the lines will move very slowly. But this allows your main
     task(s) to work faster.

  2. Opens a screen to your overscan height (in interlace if your
     default public screen is). Width is 640.

  3. If it cannot open the screen, the front screen will be set to
     black.

  4. Lines drawn vary between straight and curved, and between open and
     closed loops.

  5. If you do not see the lines being drawn, it is because:

       1. "Dimness" is not 0.

       2. Splines are not selected.

       3. There was not enough memory to open the screen.

       4. There was not enough memory to start the task.

1.126 MachV.guide/Password

```
Password
========
```

If a password is entered in the "Pasword" gadget in the Miscellaneous Features window of SetMachV, MachV will require that password before

unblanking the screen. If the screen is merely dimmed, you will not be
prompted for the password. Just type it in and press Return.

   The password may also be set with the macro function SET_PASSWORD.

1.127   MachV.guide/Suspend

```
Suspend
=======
```

   MachV may be temporarily suspended by clicking on the "Suspend"
gadget.

   You may also create a macro that has a MSC_SUSPEND function. Pressing
the hotkey that this macro is attached to will either suspend or
reactivate MachV.

   You must use the same hotkey to un-suspend MachV. Also, if you use
an <Alt> or <Shift> qualifier, you need to un-suspend with the
<Left-Shift> or <Left- Alt> qualifier.

1.128   MachV.guide/No Caps Lock

```
No Caps Lock
============
```

   Selecting "No Caps Lock" on will disable your Caps Lock key. The
light will remain on after it is pressed, however.

   You still can use it by pressing <Left-Amiga-CapsLock>.

1.129   MachV.guide/Speak Messages

```
Speak Messages
==============
```

   All messages that are displayed in the clock may be spoken (except
"Executing Macro") when this option is selected. The "speak:" device
must be mounted.

   See MSC_SPEAK.

1.130   MachV.guide/Beep

```
Beep
====
```

   MachV will beep to prompt you for certain actions, when you set an
alarm or when you execute a macro that has MSC_BEEP in it.

   You can use a custom sound for the beep. Just enter the name and
complete path of the sound file in the "Beep File Name" gadget in the
"Miscellaneous Features" window. The sound may be an IFF 8SVX sound
file or a "raw data" file with the file size in bytes 1 to 4 and the
period value in bytes 5 and 6. Many sound digitizing programs provide
the ability to save in a raw format. Several samples of both are
provided. The sound file should be as small as possible. Less than 500
bytes is recommended although you could use up to 3000 to 4000 byte
files.

   This sound is not the sound used when recording macros.

   You may specify the number of beeps with MSC_BEEP"nn" such as:

   MSC_BEEP"1" or MSC_BEEP"4"

   The volume is set in the "Miscellaneous Features", but may be
over-ridden with a second argument:

   MSC_BEEP"1,30" or MSC_BEEP"4,64"

   You can also load a new sound file with:

   MSC_LOAD_BEEP"Path/Beep Name"

1.131   MachV.guide/Hold Qualifiers

```
Hold Qualifiers
===============
```

   If "Hold Qualifiers" is checked, you may press and release any
qualifier and it will be remembered and added to the next non-qualifier
key or mouse button. The "held" qualifiers will then be released. See
below for holding for multiple keystrokes.

   Many disabled people find it difficult if not impossible to type key
combinations such as <Ctrl-Shift-F1>. So you could for example, press
<Ctrl> then <Shift>, and then <F1>. The <Ctrl> and <Shift> would be
added to the <F1> just as if they were pressed at the same time.

   This feature is not intended for someone who doesn't need it as it
sometimes may cause unexpected results. For example, if you press <Alt>
and then change your mind, the <Alt> will still be added to the next
key pressed.

Pressing <Caps Lock> or the right mouse button will clear the
qualifiers. Pressing a qualifier again will clear that qualifier.

Qualifiers may be held for multiple keystrokes if you first press
<Alt-CapsLock>. For example, if you press

<Alt-CapsLock> <Shift> <1> <2> <3> <CapsLock>

!@# will be the characters displayed (usa keymap).

This may also be used with Workbench's extended selection. Press the
<Shift> key, then click on your icons.

When using "Hold Qualifiers", the keys being held are displayed in
the clock.

            C = Control
            S = Shift
            A = Alternate
            L = Left Amiga
            R = Right Amiga

This feature may also be controlled by a macro function. See
MSC_HOLD_QUALS.

1.132 MachV.guide/Debug

Debug
=====

With this feature checked, MachV will be in the debugging mode. When
a macro is executed and for whatever reason fails, SetMachV will be run
and the offending macro will be displayed with the cursor on the
function that failed. This may be disabled by using MSC_DEBUG"Off" in a
macro.

This is useful with the WIN_SEND_CLOSE macro because most shell
windows cannot receive this message.

MSC_DEBUG"Off" WIN_SEND_CLOSE"A" MSC_DEBUG"On"

This turns debugging off, send a close window message to the active
window, and then turns debugging back on.

1.133 MachV.guide/Trace

Trace
=====

As a debugging aid, you can have each macro dumped as it executes to

a console window or serial port. This is not to be confused with 'Debug'
which runs SetMachV when an error is detected. Select 'Trace Serial' and
'Trace Console' in the 'Miscellaneous Features' window, or set it with
the MSC_TRACE function.

    Examples:

MSC_TRACE"Tog,S"
      Toggle serial tracing.

MSC_TRACE"Tog,C"
      Toggle console tracing.

    As each function is executed, the remaining portion of the macro
will be sent. If you executed the macro 'WIN_BACK_TO_FRONT
WIN_ACTIVATE"T"', you will see:

      WIN_BACK_TO_FRONT WIN_ACTIVATE"T"
      WIN_ACTIVATE"T"

    Console Tracing

    A window will be opened for output. This window will close when a
MSC_TRACE function turns off tracing. Pressing Ctrl-C will remove MachV.

    Serial Tracing

    Output will be sent to the serial device at whatever rate you have
selected by 'Prefs/Serial' or your terminal program. You can run a
terminal program to receive this output.

    See MSC_TRACE.

1.134 MachV.guide/Gameport

Gameport
========

    Macros may also be executed from a joystick or similar device
plugged into gameport 2. This is designed mainly for users with extreme
mobility impairments. Because the gameport operates differently from
the keyboard, a different implementation is necessary.

    First, gameport monitoring must be turned on. You use the function:

    MSC_GAMEPORT"On"

    to do this. If another program is using the gameport, this function
will fail. While MachV is using the gameport, no other program can use
it. To turn it off, use the function:

    MSC_GAMEPORT"Off"

    To toggle between on and off, use:

```
MSC_GAMEPORT"Tog"
```

   For a macro to be executed, it must have the appropriate name. When
the user presses the joystick forward or up, it must have the name
"JoyUp". The four names are:

```
JoyUp
JoyRight
JoyDown
JoyLeft
```

   There is an optional second argument to the gameport function. It
specifies the number of allowed directions. There are only two valid
values that may be used, 4 and 8. When 8 is used, you have the
following directions in addition to the four above:

```
JoyUpRight
JoyDownRight
JoyDownLeft
JoyUpLeft
```

   These are the "in between" positions. Since it can be difficult to
position precisely on these, the default mode is 4 directions.

   You could create this macro named "JoyDown":

```
EXE_CMD"run application"
```

   When the user pushes the joystick down (or back), the program
"application" would be run.

   The three names:

```
JoyFire1,
JoyFire2
JoyFire3
```

   are used for the fire buttons.

   Note that all names must be spelled exactly as shown. Joyup is not a
match.

   If you want to always have the gameport monitored, you can have it
turn on automatically when MachV is run by using one of the following
macros:

```
EXE_AUTO"" MSC_GAMEPORT"On"
```

```
EXE_AUTO"" MSC_GAMEPORT"On,4"
```

```
EXE_AUTO"" MSC_GAMEPORT"On,8"
```

```
EXE_AUTO"" MSC_GAMEPORT"Tog,8"
```

   This macro should be attached to "No Key" in the first configuration.
See Changing the Current Hotkey.

See MSC_GAMEPORT.

1.135 MachV.guide/ARexx

ARexx
*****

   MachV has a complete ARexx interface. You can execute ARexx
programs, send ARexx messages to other ports, execute MachV macros from
a file and inquire about the status of many of MachV's features.

  Sending Commands To An ARexx Port
  Sending Commands From ARexx To MachV
  Executing ARexx Programs
  Executing ARexx Commands
  MachV Status

1.136 MachV.guide/Sending Commands To An ARexx Port

Sending Commands To An ARexx Port
=================================

   MSC_PUTMSG_AREXX"PORTNAME,command,results_variable"

   Use this to send a command directly to an ARexx port. With this
function, you can avoid using 'run rx command'. This function is
synchronous! It will not return until replied to by the receiving
process. If for some reason the receiving process will not reply, MachV
will quit waiting if you press <ESC>. This can cause a crash if the
process does at some point try to reply.

   Examples:

   MSC_PUTMSG_AREXX"TURBOTEXT0,openfile ${ttxfile}"

   MSC_PUTMSG_AREXX"EMACS1,(fill-paragraph nil)"

   MSC_PUTMSG_AREXX"rexx_ced,cedtofront"

   There is an optional 3rd argument. If supplied, the message to the
ARexx port will request a result and use the 3rd argument as the name
of an environment variable to store it. The variable will be local or
global, depending on the variable setting in Miscellaneous Features.

   Eamples:

   This sets the environment variable ttx_word to the word under the
cursor when using TurboText.

```
MSC_PUTMSG_AREXX"TURBOTEXT0,getword,ttx_word"
```

This sets the environment variable ced_line to the line under the
cursor.

```
MSC_PUTMSG_AREXX"rexx_ced,status 55,ced_line"
```

1.137   MachV.guide/Sending Commands From ARexx To MachV

```
Sending Commands From ARexx To MachV
====================================
```

Macros may be sent to MachV from ARexx programs with the ARexx
command:

```
address 'MACHV_REXX' 'macro string'
```

MachV will execute the macro just as if you pressed the hotkey. MachV
will return a result of 10 if the macro fails. Example:

```
address 'MACHV_REXX' 'WIN_TO_FRONT"AppTitle"
WIN_SIZE"AppTitle,500,50"'
```

If the Quick Edit window is open, the ARexx macro will be shown in
its string gadget instead of being executed. This is a debugging aid –
it shows exactly what MachV is actually receiving.

Macros sent to MachV from ARexx programs may be executed
synchronously or asynchronously. An asynchronous macro will return
before it has actually begun execution. To do this, simply prefix the
macro with a "0". Examples:

```
address 'MACHV_REXX' 'clk_clock"off"'          synchronous
```

```
address 'MACHV_REXX' '0 clk_clock"off"'      asynchronous
```

When this is executed from the shell, the prompt will return
immediately:

```
address 'MACHV_REXX' '0 pau_seconds"3" clk_clock"off"'
```

As many macros as you wish may be sent to MachV. They will be
executed in turn.

You may also send a command to an ARexx port with:

```
MSC_PUTMSG_AREXX
```

1.138   MachV.guide/Executing ARexx Programs

```
Executing ARexx Programs
========================
```

   You can execute ARexx programs with the function EXE_AREXX_COMMAND.
This eliminates the need for "rx" or "run rx" and also allows scripts
with just MachV macros without "address MACHV_REXX".

   The default filename extension is .mach.

   If your ARexx program or script ends with a "return value", that will
be set to the environment variable that is specified as the 2nd
argument. If you had this program named rexx:test.mach –

```
    /* Return an answer to MachV which will set an env variable. */

      parse arg arg1

      a=2
      return a*arg1
```

   and you executed this macro:

   EXE_AREXX_COMMAND"test 5,answer"

   the env variable "answer" will be set to 10. The passed argument
could also have been a variable:

   EXE_AREXX_COMMAND"test ${count},answer"

1.139 MachV.guide/Executing ARexx Commands

```
Executing ARexx Commands
========================
```

   This allows you to execute arexx functions directly. A trivial
example:

   EXE_AREXX_STRING"return time(hours),results"

   The env variable "results" would get set to hours past midnight.

   You could insert text that is self-incrementing with this:

   EXE_AREXX_STRING"return ${count}+1,count" "${count}"

   This has ARexx increment the variable count, and then the results
are sent as keystrokes.

1.140   MachV.guide/MachV Status

```
MachV Status
============
```

   Check the condition of various flags with:

   MSC_STATUS"N"

   "N" may be one of the following:

```
   sunmouse           0        splines            14
   keyactivate        1        quick blank        15
   mouse to menu      2        speak messages     16
   menu wrap          3        debug              17
   window cycle       4        button swap        18
   auto poptop        5        hold quals         19
   keyclick           6        hotkeys enabled    20
   clock              7        send up keys       21
   vertical clock     8        nocapslock         22
   clock top window   9        remove lf          23
   clock float        10       crlf               24
   clock gadgets      11       freeze vert        25
   seconds            12       freeze horz        26
   meter              13       keysoff            26
```

   MSC_STATUS returns a '1' in the ARexx variable RESULT if the flag is
set. Example:

```
       address 'MACHV' 'status"17"'
       if RESULT = 1 then
          say 'debugging is on'
       else
          say 'debugging is off'
```

1.141   MachV.guide/Configurations

```
Configurations
**************
```

   You may have more than one configuration within your MachV.prefs
file. The title of each may match window or screen titles and become
active when that window becomes active.

   Let's assume that you've set some macros attached to function keys
and other features that work well in a shell. Now you want to run an
application program such as WordPerfect which uses the function keys.
You could disable all macros while using WordPerfect, but there is an
alternative. You can add another configuration to the one that you have
already set up and modify it to work with WordPerfect.

   To do this, select "Add" in SetMachV. A duplicate configuration will
be made and numbered 2. Add whatever macros you want for this

configuration, and change any settings that you want.

    You can click on "Select Title" to obtain a list of available titles.
They are listed window titles first, then screen titles, and finally,
program names (which are prefaced with the '$'). Selecting one of these
will place it in your configuration title string gadget.

    If WordPerfect (or your desired application) is not running, you can
manually enter the title by typing in " WordPerfect" (without the
quotes). Notice the space before the WordPerfect, that's like it is in
the title bar. You may need to add '#?' if the window title is longer.
Examples:

        " WordPerfect#?"   " WordP#?"   "ProWri#?"   "Final#?"

    In your first configuration, you could type in the title
"AmigaShell" or whatever is in the title bar of your shell. Now, when a
WordPerfect window becomes active, the second configuration becomes
active. Click in your shell window and its configuration becomes active.

    If you are using the SunMouse feature you do not need to click in the
window – the configuration will become active when the window becomes
active.

    Configuration titles may use case-insensitive standard AmigaDOS wild
cards to match a variety of windows.

    In order to avoid confusion, configurations will not change
automatically when SetMachV is running.

  Changing Configurations
  Default Configuration
  Windows and Screens With No Title
  Attach To The Program Name
  Adding And Deleting
  Global Hotkeys
  Loading And Saving
  The Settings File

1.142 MachV.guide/Changing Configurations

Changing Configurations
=======================

    You may have as many configurations as you like. Each one may be
attached to a window or screen through its title. The title may use
standard AmigaDOS wild card pattern matching to match a window's title.

    An "@" symbol as the first character will lock a configuration which
may then only be changed from SetMachV or by executing a macro with a
function such as CFG_NEXT. When a configuration is changed from within
a macro, there will be no further execution of that macro. That is,
with the macro:

```
CFG_NEXT WIN_ACTIVATE"title"
```

The WIN_ACTIVATE"title" will not be executed.

You may put in multiple window titles by enclosing them in () separated by the vertical bar | (next to the Backspace key).

```
(Ashell#?|TextEd#?| WordP#?)
```

Up to a maximum of 80 characters may be used. This example shows a configuration that will become active when you click in a window titled "Ashell", "TextEd", or " WordPerfect".

You can also use something like this:

```
#?.c
```

to match all windows with C source files. Or:

```
#?.(c|h)
```

to match .c and .h files.

You may find that when a file requester opens, your configuration changes. You could include its title also. For example, the above #?.(c|h) could be:

```
(#?.(c|h)|Select File#?)
```

When a new configuration is made active, its title will appear in the clock for about one second.

If you name a macro the same name as the configuration title, that macro will be executed when the configuration changes. That could be something simple like a MSC_BEEP to let you know that it did change.

When searching for a macro name when configurations are changing, the macro name can be matched by the wild cards in the configuration name. For example, a macro named "gads.c" will execute when a configuration named #?.c becomes active. So would a macro named #?.c.

When a configuration first becomes active, it will not change to another if the window that opens is owned by the same task. This keeps it from changing when something like a requester opens.

If it does not change when you think it should, try activating a window that is owned by another task (like the MachV clock) and then activate the desired window.

1.143 MachV.guide/Default Configuration

```
Default Configuration
=====================
```

The first configuration with no title will be activated when no title matches. This means that you can set up a default configuration that will be active when you click in a window that does not match any of your configuration titles.

1.144 MachV.guide/Windows and Screens With No Title

```
Windows and Screens With No Title
=================================
```

Not all windows or screens have real titles. It may be just text written where the title should be. MachV cannot locate these titles and thus cannot change to the proper configuration automatically unless you attach to the program name.

You can manually change to these configurations with a CFG_NEXT or CFG_CHANGE macro.

In SetMachV, you can click on "Select Title" to obtain a list of available titles. Scroll down until you see ">> Programs <<" followed by program names (which are prefaced with the '$'). Selecting one of these will place it in your configuration title string gadget.

When the window that is owned by that program becomes active, the configuration will change.

See Attach To The Program Name.

1.145 MachV.guide/Attach To The Program Name

```
Attach To The Program Name
==========================
```

You may change to untitled windows by specifying the program's name. Do this by prefacing the configuration title with the "$" symbol followed by the program name. For example: a configuration titled "$dh0:DPaint" will become active when you click in the Deluxe Paint window. Click on "Select Title" for a list of program or task names available.

Note that you can't mix program names and window/screen titles. When matching program names, the first character must be the '$'. Example:

    $(dpaint|FinalCopy)

    Not:

    ($dpaint|$FinalCopy)

1.146   MachV.guide/Adding And Deleting

```
Adding And Deleting
===================
```

   "Add" duplicates the current configuration and then makes that one
current. You may then add macros and change other settings in the new
configuration.

   Enter the name for this configuration in the string gadget under
"Select Title". This title usually matches a window, screen or program
name. When a window becomes active that has a title that matches it,
that configuration becomes the current one. The title can use AmigaDos
wildcards.

   Delete a configuration with the "Del" gadget. You cannot delete the
sole remaining configuration.

1.147   MachV.guide/Global Hotkeys

```
Global Hotkeys
==============
```

   If you have some hotkeys in your first configuration that you would
like to be available to all configurations, simply check that hotkey's
"Global" flag in the Flags window. Here are some points to remember:

  1. Only hotkeys in the first configuration may be made global.

  2. When a configuration other than the first one is active and a key
     is pressed, MachV first checks the current configuration for a
     hotkey and then checks the first configuration for a global
     hotkey. This allows local hotkeys to override global ones.

  3. When executing macros from the select list, the list will contain
     local and global macros.

  4. When selecting by name from SetMachV, only local macros will be
     listed.

  5. Macros that execute when a configuration is changed, that is,
     macros with the same name as the configuration title, may be
     global.

  6. Screen blanking macros may be global.

  7. The MSC_SET_FLAGS function operates only on macros in the current
     configuration - NOT on global macros.

   For example, if you always want your <Left-Amiga-ESC> NewShell
hotkey to be available no matter what configuration is current, you
just need to create the macro in the first configuration and check the
"Global" flag.

If you want to use a global hotkey in another configuration as a
normal key, you can override it by either just marking that key as
"Temporary" or by entering the key itself as a "Non Recursive" macro.
See Flags.

Assume that F5 is global and you wish to use it as a normal F5 in
another configuration. Run SetMachV and select the desired
configuration and then select F5. Open the Flags window and click on
the "Temporary" flag.

Another way would be to select F5 and enter "\F5" (with the quotes!)
 as its macro. Then click on the "Non Recursive" flag. This will simply
pass the F5 key through.

1.148 MachV.guide/Loading And Saving

Loading And Saving
==================

For any changes to be permanent, you just need to select
"Project/Save" or "Project/Save As..." from the menu. When MachV is
first run, it looks for MachV.prefs from the command line or Tool
Types, then from the current directory. If it is not found, it tries to
load ENVARC:MachV/MachV.prefs.

Select "Project/Open" to load other settings files.

MachV will also reload your MachV.prefs file if you edit it with an
editor and save. In other words, MachV is notified of any changes to
ENVARC:MachV (or whatever your prefs directory is) and reloads the
MachV.prefs file and updates gadget imagery.

Configurations can also be saved with the function CFG_SAVE and
loaded with the function CFG_LOAD.

You may also save from the "Quick Edit" window.

1.149 MachV.guide/The Settings File

The Settings File
=================

This section describes the settings file itself. Most users will not
need to do anything to it, as it is read and saved automatically.

The MachV.prefs file is straight ASCII text which may be edited with
any text editor that can save normal ASCII text.

Multiple configurations are all saved together under one file name.

Each configuration within this file must start with ID=5.0. The
specific order of everything else is not important.

    Long lines are broken into smaller lengths (252 characters) and
terminated with a backslash. These allows some text editors that cannot
handle long lines to be used to edit a settings file.

    Macros must be in the following format:

    Macro=qualifiers key{macroname}[XXX]Macro body

    Set Escape Characters for the keys and qualifiers. The name should
be no more than 12 characters. To enter no name use {}. The [XXX] is the
hexidecimal representation of the selected flags. If the macro has a
gadget, the left and top edges will also be saved. Example:

    Macro={Slow Mouse}[008,525,10]MOU_SPEED"-1"

    This macro named "Slow Mouse" is a "NO KEY" macro with a gadget at
525x 10y.

    More Examples:

    Macro=\Lj{WinToFront}[000]WIN_BACK_TO_FRONT

    This is the <Left-Amiga-J> window to front hotkey.

    Macro=\L\E{NewShell}[080]SCR_WB_TO_FRONT EXE_CMD"NewShell
\"con:0/0/640\-/100/NewShell\""

    This is the <Left-Amiga-ESC> NewShell command.

    Macro=\A\E2{StartStop}[080]MAC_RECORD_KEYED

    This is <Alt-ESC> and begins or ends a "recording to key" recording
session.

1.150 MachV.guide/Macro Functions

```
Macro Functions
***************
```

```
  CLK_CUSTOM_MSG
  CLK_CLOCK
  CLK_TO_FRONT
  CLK_TO_NEXT_SCREEN
  CLK_RESET_METER
  CLK_SET_RATE
  CLK_ALARM
  CLP_PASTE
  CLP_PRINT
  CLP_TRANSFER
  CLP_VIEW
```

```
CLP_CURRENT_UNIT
CFG_CHANGE
CFG_NEXT
CFG_SELECT
CFG_LOAD
CFG_SAVE
EXE_AUTO
EXE_CMD
EXE_AREXX_COMMAND
EXE_AREXX_STRING
EXE_SETMACH
MAC_SELECT_NAMED
MAC_EXECUTE_NAMED
MAC_FINISH
MAC_MACROS
MAC_RECORD_KEYED
MAC_RECORD_NAMED
MAC_QUICK_EDIT
MAC_DELETE
MOU_LBD
MOU_MBD
MOU_RBD
MOU_XY
MOU_OFF
MOU_SPEED
MOU_THRESHOLD
MOU_SUNMOUSE
MOU_TO_MENU
MOU_FREEZE_VERT
MOU_FREEZE_HORZ
MSC_FREEZE_KEYS
MSC_BEEP
MSC_LOAD_BEEP
MSC_BLANK_SCREEN
MSC_CHG_DIR
MSC_PRIORITY
MSC_DEBUG
MSC_GAMEPORT
MSC_HOLD_QUALS
MSC_PUTMSG
MSC_PUTMSG_AREXX
MSC_QUIT
MSC_SET_FLAGS
MSC_SPEAK
MSC_STATUS
MSC_SUSPEND
MSC_TRACE
PAU_SECONDS
PAU_FIX_INPUT
PAU_VAR_INPUT
SCR_BACK_TO_FRONT
SCR_TO_FRONT
SCR_FRONT_TO_BACK
SCR_TO_BACK
SCR_WB_TO_FRONT
SCR_WB_TO_BACK
SCR_SELECT_SCREEN
```

```
      SCR_KILL_SCREEN
      SET_DATE_FORMAT
      SET_DATE_SEPARATOR
      SET_NO_LAYER
      SET_PASSWORD
      SET_MOUSE_PORT
      SET_TASK_PRIORITY
      SET_VARIABLE
      TXT_TIME_PASTE
      TXT_TIME_PRINT
      TXT_DATE_PASTE
      TXT_DATE_PRINT
      WIN_ACTIVATE
      WIN_ACTIVATE_NEXT
      WIN_TO_FRONT
      WIN_TO_BACK
      WIN_BACK_TO_FRONT
      WIN_FRONT_TO_BACK
      WIN_MOVE_WINDOW
      WIN_LAYER
      WIN_SIZE_WINDOW
      WIN_CHANGE_WINDOW
      WIN_ZOOM_WINDOW
      WIN_SEND_CLOSE
      WIN_SEND_GADGET_DOWN
      WIN_SEND_GADGET_UP
      WIN_SEND_MENU
      WIN_SEND_OTHER
      WIN_SELECT_GADGET
      WIN_SELECT_WINDOW
      WIN_WAIT_WINDOW
      WIN_WINDOW_OPEN
```

1.151 MachV.guide/CLK_CUSTOM_MSG

```
CLK_CUSTOM_MSG
==============
```

   This function displays a message in the clock. It takes one argument,
the message to display.

```
CLK_CUSTOM_MSG"Hello"
```
     Display the message "hello" in the clock.

   You could combine this with another function to tell you what
happened. For example:

```
   SET_MOUSE_PORT"0"  CLK_CUSTOM_MSG"Mouse Port 0"
```

1.152 MachV.guide/CLK_CLOCK

```
CLK_CLOCK
=========
```

   This function turns the clock on and off. It takes one argument.

```
CLK_CLOCK"Tog"
```
     Toggle the clock on or off.

```
CLK_CLOCK"Off"
```
     Turn the clock off.

```
CLK_CLOCK"On"
```
     Turn the clock on.

1.153 MachV.guide/CLK_TO_FRONT

```
CLK_TO_FRONT
============
```

   This function sends the clock to the top screen. It takes no
arguments.

```
    CLK_TO_FRONT
```

1.154 MachV.guide/CLK_TO_NEXT_SCREEN

```
CLK_TO_NEXT_SCREEN
==================
```

   This function sends the clock to the next screen. It takes no
arguments.

```
    CLK_TO_NEXT_SCREEN
```

1.155 MachV.guide/CLK_RESET_METER

```
CLK_RESET_METER
===============
```

   This function resets the meter. It takes no arguments.

```
    CLK_RESET_METER
```

1.156 MachV.guide/CLK_SET_RATE

```
CLK_SET_RATE
============
```

   This function sets the rate for the meter. It takes one argument.

```
CLK_SET_RATE"600"
```
     Set the rate to use to 6.00 per hour.

1.157 MachV.guide/CLK_ALARM

```
CLK_ALARM
=========
```

   This function executes an alarm. It takes one to four arguments. It
is used automatically by the clock alarms interface. It is usually
prefaced by an EXE_AUTO function.

   1. The alarm message.

   2. If 1, the message will be spoken.

   3. If 1, the message will be displayed in a requester.

   4. Minutes to wait before coming back. Requres the above argument = 1.

```
CLK_ALARM"Dentist appointment,1,1,5"
```
     Display a requester with the message "Dentist appointment", speak
     the message, wait five minutes if user clicks on "Later".

```
CLK_ALARM"Time for Seinfeld!,1,0"
```
     Say the message.

1.158 MachV.guide/CLP_PASTE

```
CLP_PASTE
=========
```

   This function pastes (sends as keystrokes) the contents of the
clipboard. It takes one optional argument, a clipboard unit number from
0 to 255.

```
CLP_PASTE"1"
```
     Paste the clipboard to the active window from unit 1.

```
CLP_PASTE
```
     Paste the clipboard to the active window from the current unit..

   See Pasting The Clipboard.

1.159   MachV.guide/CLP_PRINT

```
CLP_PRINT
=========
```

   This function prints the contents of the clipboard. It takes one
optional argument, a clipboard unit number from 0 to 255.

```
CLP_PRINT"1"
```
     Print the clipboard from unit 1.

```
CLP_PRINT
```
     Print the clipboard from the current unit.

   See Printing The Clipboard.

1.160   MachV.guide/CLP_TRANSFER

```
CLP_TRANSFER
============
```

   This function transers the contents of the clipboard from one unit to
another. It takes two arguments, a source and destination clipboard unit
number from 0 to 255.

   You may also tranfer to or from a file or environment variable and
the clipboard by using the file or variable name.

```
CLP_TRANSFER"0,1"
```
     Transfer the contents of the clipboard from unit 0 to unit 1.

```
CLP_TRANSFER"ram:temp,0"
```
     Transfer file named ram:temp to unit 0.

```
CLP_TRANSFER"0,$capture"
```
     Transfer unit 0 to variable "capture".

   See Transfering To Other Units.

1.161   MachV.guide/CLP_VIEW

```
CLP_VIEW
========
```

   This function opens a window and displays the clipboard. It takes one
optional argument that specifies the desired clipboard unit.

```
CLP_VIEW
```

See Viewing The Clipboard.

1.162   MachV.guide/CLP_CURRENT_UNIT

```
CLP_CURRENT_UNIT
================
```

This function changes the current unit. It takes one argument, the
desired clipboard unit.

```
CLP_CURRENT_UNIT"N"
```

See Viewing The Clipboard.

1.163   MachV.guide/CFG_CHANGE

```
CFG_CHANGE
==========
```

This function changes to another configuration. It takes one
argument.

```
CFG_CHANGE"Workbench"
    Change to a configuration "Workbench".
```


1.164   MachV.guide/CFG_NEXT

```
CFG_NEXT
========
```

This function changes to the next configuration. It takes no
arguments.

```
CFG_NEXT
    Change to the next configuration.
```


1.165   MachV.guide/CFG_SELECT

```
CFG_SELECT
==========
```

   This function opens a list box with available configurations. It
takes one optional argument, the title to display in the list box.

```
   CFG_SELECT
```


1.166   MachV.guide/CFG_LOAD

```
CFG_LOAD
========
```

   This function loads a new configuration (prefs) file. It takes one
argument.

```
CFG_LOAD"ENVARC:MachV/backup.prefs"
     Load the prefs file ENVARC:MachV/backup.prefs.
```


1.167   MachV.guide/CFG_SAVE

```
CFG_SAVE
========
```

   This function saves a configuration (prefs) file. It takes one
optional argument, a file name. If no file name, the current name is
used.

```
CFG_SAVE"ENVARC:MachV/MachV.prefs"
     Save the prefs file ENVARC:MachV/MachV.prefs.
```


1.168   MachV.guide/EXE_AUTO

```
EXE_AUTO
========
```

   This function executes a macro at a specific time. It takes one to
seven arguments. The first is the time, optionally then, either the
date, or day(s) of the week. The standard AmigaDos wild cards ? and #
may be used in these arguments.

```
EXE_AUTO"17:35" ...more macro functions...
     Execute this macro at 5:35pm.

EXE_AUTO"15:30,12-09-93" ...more macro functions...
     Execute this macro at 3:30pm on 12-09-93.
```

```
EXE_AUTO"15:30,??-05-??" ...more macro functions...
     Execute this macro at 3:30pm on the 5th of every month.

EXE_AUTO"5:00,Mon" ...more macro functions...
     Execute this macro at 5:00am on every Monday.

EXE_AUTO"15:30,Mon,Wed,Fri" ...more macro functions...
     Execute this macro at 3:30pm on every Monday, Wednesday and Friday.

EXE_AUTO"+30" ...more macro functions...
     Execute every 30 minutes.

EXE_AUTO"" ...more macro functions...
     Execute when MachV is first run.

   See Auto Executing Macro.
```

1.169 MachV.guide/EXE_CMD

```
EXE_CMD
=======

   This function runs a program. It takes one or two arguments.

EXE_CMD"Run ProWrite"
     Run the program "ProWrite"

EXE_CMD"Packer20,PowerPack"
     If the window or screen title is found, bring it to the front,
     otherwise run the program Packer20.

EXE_CMD"Run FinalCopy,Final Copy"
     Run FinalCopy unless the screen titled "Final Copy" is open.

EXE_CMD"${shell}"
     Environment variables may also be used.

   See Executing Programs, and See Environment Variables.
```

1.170 MachV.guide/EXE_AREXX_COMMAND

```
EXE_AREXX_COMMAND
=================

   This function eliminates the need for "rx" or "run rx" and also
allows scripts with just MachV macros without "address MACHV_REXX". It
takes one or two arguments.

EXE_AREXX_COMMAND"test.rexx"
```

    Execute the program test.rexx.

EXE_AREXX_COMMAND"test,results"
    Execute the program test.mach (default extension .mach) and store
    any returned value in the environment variable "results".

    See ARexx, and See Environment Variables.

1.171 MachV.guide/EXE_AREXX_STRING

EXE_AREXX_STRING
================

    This function will instruct ARexx to execute the first argument as an
ARexx program. It takes one or two arguments.

AREXX_STRING"return time(hours),results"
    Execute the ARexx function time(hours) and store the results in
    "results".

EXE_AREXX_COMMAND"test,results"
    Execute the program test.mach (default extension .mach) and store
    any returned value in the environment variable "results".

    See ARexx, and See Environment Variables.

1.172 MachV.guide/EXE_SETMACH

EXE_SETMACH
===========

    This function runs SetMachV. It takes no arguments. If SetMachV is
already running, it will be brought to the front.

    EXE_SETMACH

    See SetMachV.

1.173 MachV.guide/MAC_SELECT_NAMED

MAC_SELECT_NAMED
================

    This function opens a list box to select a macro name from and
executes that macro. It takes one optional argument.

MAC_SELECT_NAMED"Macro Selection"

```
    Open the list box. "Macro Selection" is the optional title, it
    replaces the default "Select A Name" title.
```

1.174 MachV.guide/MAC_EXECUTE_NAMED

```
MAC_EXECUTE_NAMED
=================
```

This function executes a named macro. It takes one argument, the macro name.

```
MAC_EXECUTE_NAMED"zoom"
    Execute a macro named "zoom".
```

1.175 MachV.guide/MAC_FINISH

```
MAC_FINISH
==========
```

This function completes a word or phrase by executing a macro based on characters already typed that match a macro's name. It takes no arguments.

```
    MAC_FINISH
```

If you have a macro "Polyglot Software" (with quotes) with a name of "Polyglot" (without quotes) attached to the key <F1>, and you type Pol<F1>, then "yglot Software" will be typed out. In other words, it skips what you have already typed and then sends the rest of the macro as keystrokes. You may include special keys like "\N". The macro must be enclosed in quotes. The matching is case insensitive. It will attempt local then global macros.

1.176 MachV.guide/MAC_MACROS

```
MAC_MACROS
==========
```

This function turns all macros on and off. It takes one argument.

```
MAC_MACROS"Tog"
    Toggle macros on or off.
```

```
MAC_MACROS"Off"
    Turn macros off.
```

```
MAC_MACROS"On"
```

Turn macros on.

You must use the same hotkey to enable macros. You cannot depend on
"Hold Qualifier" to add the appropriate qualifiers when enabling macros.

1.177   MachV.guide/MAC_RECORD_KEYED

MAC_RECORD_KEYED
================

This function starts or stops macro recording. It takes no arguments.

MAC_RECORD_KEYED

See Recording Macros.

1.178   MachV.guide/MAC_RECORD_NAMED

MAC_RECORD_NAMED
================

This function starts or stops macro recording to a named macro. It
takes no arguments.

MAC_RECORD_NAMED

See Recording Macros.

1.179   MachV.guide/MAC_QUICK_EDIT

MAC_QUICK_EDIT
==============

This function opens a window to edit a macro. It takes no arguments.

MAC_QUICK_EDIT

See Quick Edit.

1.180   MachV.guide/MAC_DELETE

```
MAC_DELETE
==========
```

   This function deletes a macro. It takes one optional argument, a
macro name. If no name is provided, this function will delete the macro
that contains it. Deletion occurs after the entire macro has executed.

```
MAC_DELETE
     Delete this macro.
```

```
MAC_DELETE"Temp"
     Delete a macro named "Temp"
```

1.181 MachV.guide/MOU_LBD

```
MOU_LBD
=======
```

   These functions simulate left button down and up.

```
MOU_LBD
     Send a left button down.
```

```
MOU_LBU
     Send a left button up.
```

1.182 MachV.guide/MOU_MBD

```
MOU_MBD
=======
```

   These functions simulate middle button down and up.

```
MOU_MBD
     Send a middle button down.
```

```
MOU_MBU
     Send a middle button up.
```

1.183 MachV.guide/MOU_RBD

```
MOU_RBD
=======
```

   These functions simulate right button down and up.

```
MOU_RBD
      Send a right button down.
```

```
MOU_RBU
      Send a right button up.
```

1.184 MachV.guide/MOU_XY

```
MOU_XY
======
```

This function moves the mouse to the specified x/y coordinates. It takes two arguments.

```
   MOU_XY"xxx,yyy"         Move the mouse to xxx,yyy.
```

1.185 MachV.guide/MOU_OFF

```
MOU_OFF
=======
```

This function sets the number of seconds to wait for the mouse pointer to turn off. It takes one argument.

```
MOU_OFF"4"
      Set "Mouse Off" to 4 seconds.
```

1.186 MachV.guide/MOU_SPEED

```
MOU_SPEED
=========
```

This function sets the mouse accelerator. It takes one argument, a value in the range of -9 to 9. 0 disables the accelerator.

```
MOU_SPEED"-3"
      Set "Mouse Speed" to -3.
```

1.187 MachV.guide/MOU_THRESHOLD

```
MOU_THRESHOLD
=============
```

```
    This function sets the mouse thrshold. It takes one argument, a value
in the range of 0 to 9.

MOU_THRESHOLD"5"
    Set "Mouse Threshold" to 5.
```

1.188 MachV.guide/MOU_SUNMOUSE

```
MOU_SUNMOUSE
============

    This function turns the SunMouse on and off. It takes one argument.

MOU_SUNMOUSE"Tog"
    Toggle the SunMouse on or off.

MOU_SUNMOUSE"Off"
    Turn the SunMouse off.

MOU_SUNMOUSE"On"
    Turn the SunMouse on.
```

1.189 MachV.guide/MOU_TO_MENU

```
MOU_TO_MENU
===========

    This function turns Mouse To Menu on and off. It takes one argument.

MOU_TO_MENU"Tog"
    Toggle "Mouse To Menu" on or off.

MOU_TO_MENU"Off"
    Turn "Mouse To Menu" off.

MOU_TO_MENU"On"
    Turn "Mouse To Menu" on.
```

1.190 MachV.guide/MOU_FREEZE_VERT

```
MOU_FREEZE_VERT
===============

    This function turns the freezing of vertical mouse on and off. It
takes one argument.

MOU_FREEZE_VERT"Tog"
```

```
     Toggle freezing of vertical mouse movements.

MOU_FREEZE_VERT"On"
     Freeze vertical mouse movements.

MOU_FREEZE_VERT"Off"
     Unfreeze vertical mouse movements.
```

1.191 MachV.guide/MOU_FREEZE_HORZ

```
MOU_FREEZE_HORZ
===============

   This function turns the freezing of horizontal mouse on and off. It
takes one argument.

MOU_FREEZE_HORZ"Tog"
     Toggle freezing of horizontal mouse movements.

MOU_FREEZE_HORZ"On"
     Freeze horizontal mouse movements.

MOU_FREEZE_HORZ"Off"
     Unfreeze horizontal mouse movements.
```

1.192 MachV.guide/MSC_FREEZE_KEYS

```
MSC_FREEZE_KEYS
===============

   This function locks out keyboard input. It takes one argument, 'On',
'Off' or 'Tog'.

MSC_FREEZE_KEYS"Tog"
     Toggle keyboard input on or off.

MSC_FREEZE_KEY"Off"
     Keyboard input off.

MSC_FREEZE_KEYS"On"
     Keyboard input on.

   See MOU_FREEZE_VERT.

   See MOU_FREEZE_HORZ.
```

1.193 MachV.guide/MSC_BEEP

```
MSC_BEEP
========
```

   This function produces a beep. It takes two optional arguments, the
number of beeps, and the volume (range 0-64).

```
MSC_BEEP"3,64"
```
      Produce 3 beeps at full volume.

   See MSC_LOAD_BEEP, for loading a new beep sound file.

1.194 MachV.guide/MSC_LOAD_BEEP

```
MSC_LOAD_BEEP
=============
```

   This function loads a new beep sound file. It takes one argument, the
path and name of an IFF sound file.

```
MSC_LOAD_BEEP"ENVARC:MachV/vibra.8SVX"
```
      Load the beep sound file 'vibra.8SVX'.

   See Beep, for more information.

1.195 MachV.guide/MSC_BLANK_SCREEN

```
MSC_BLANK_SCREEN
================
```

   This function blanks the screen. It takes two optional arguments, the
screen dimness and splines on/off switch. If either is supplied, it will
change the previous setting.

```
MSC_BLANK_SCREEN
```
      Blank the screen. Use dimness as set in Miscellaneous Features.

```
MSC_BLANK_SCREEN"0"
```
      Blank completely black.

```
MSC_BLANK_SCREEN"0,on"
```
      Blank completely black and turn on splines.

```
MSC_BLANK_SCREEN"0,off"
```
      Blank completely black and turn off splines.

```
MSC_BLANK_SCREEN"4"
```
      Blank at dimness of 4.

   See Screen Blanking.

1.196   MachV.guide/MSC_CHG_DIR

```
MSC_CHG_DIR
===========
```

   This function changes the current directory. It takes one argument,
the directory to change to. Some programs require that you be in a
specific directory before executing the program. With this, you may do
so temporarily.

```
MSC_CHG_DIR"dh0:games"
     Change to dh0:Games.

   Example:

   MSC_CHG_DIR"dh0:games/arcade" EXE_CMD"Run SuperActionGame"

   See EXE_CMD.
```

1.197   MachV.guide/MSC_PRIORITY

```
MSC_PRIORITY
============
```

   This function changes the priority of MachV until the end of the
macro. Programs executed in the same macro after changing the priority
will have the new priority. After the macro is completed, the original
priority for MachV will be restored.

```
MSC_PRIORITY"3"
     Change to priority of 3.

   Example:

   MSC_PRIORITY"4" EXE_CMD"run myapplication"

   The priority of myapplication will be 4.
```

1.198   MachV.guide/MSC_DEBUG

```
MSC_DEBUG
=========
```

   This function turns debugging on nd off. It takes one argument.

```
MSC_DEBUG"Tog"
     Turn debugging on or off.

MSC_DEBUG"Off"
     Turn debugging off.
```

```
MSC_DEBUG"On"
     Turn debugging on.
```

    When debugging is on, any macro that has a mispelled function or for
any reason fails will run SetMachV and place the cursor on the offending
function.

    See Debug.

1.199   MachV.guide/MSC_GAMEPORT

```
MSC_GAMEPORT
============
```

    This function turns gameport monitoring on and off. It takes one
required argument and one optional argument.

```
MSC_GAMEPORT"Tog"
     Turn gameport monitoring on or off.

MSC_GAMEPORT"Off"
     Turn gameport monitoring off.

MSC_GAMEPORT"On,8"
     Turn gameport monitoring on using 8 directions.
```

    Valid macro names that will get executed are:

```
                        JoyUp
        JoyUpLeft                 JoyUpRight
    JoyLeft                                   JoyRight
         JoyDownLeft            JoyDownRight
                      JoyDown
```

    Also, the names JoyFire1, JoyFire2, and JoyFire3 are used for the
fire buttons.

1.200   MachV.guide/MSC_HOLD_QUALS

```
MSC_HOLD_QUALS
==============
```

    This function turns "Hold Qualifiers" on and off. It takes one
argument.

```
MSC_HOLD_QUALS"Tog"
     Turn "Hold Qualifiers" on or off.

MSC_HOLD_QUALS"Off"
```

     Turn "Hold Qualifiers" off.

MSC_HOLD_QUALS"On"
     Turn "Hold Qualifiers" on.

   See Hold Qualifiers.

1.201 MachV.guide/MSC_PUTMSG

MSC_PUTMSG
==========

   This function sends a message to a port. It takes upto four
arguments.

   1. Port name. The name (case sensitive) of the destination port.

   2. The message class.

   3. The message code.

   4. The message qualifiers.

MSC_PUTMSG"blanker,400,45,0"
     Send a RAWKEY message to a port named "blanker". The values are in
     hex. The function actually simulates an IntuiMessage.

   You will probably never need this function, but is provided for
those rare occasions.

   See MSC_PUTMSG_AREXX, and also see WIN_SEND_OTHER.

1.202 MachV.guide/MSC_PUTMSG_AREXX

MSC_PUTMSG_AREXX
================

   This function sends a message to an ARexx port. It takes upto 3
arguments.

   1. Port name. The name (case sensitive) of the destination port.

   2. The ARexx command.

   3. Optional results variable.

MSC_PUTMSG_AREXX"rexx_ced,cedtofront"
     This sends a 'to front' command directly to the CED ARexx port.

   See Sending Commands To An ARexx Port.

1.203 MachV.guide/MSC_QUIT

```
MSC_QUIT
========
```

   This function terminates MachV. It takes no arguments.

```
MSC_QUIT
     Tell MachV to quit.
```

1.204 MachV.guide/MSC_SET_FLAGS

```
MSC_SET_FLAGS
=============
```

   This function sets (or clears) various macro flags. It takes two or three arguments.

  1. The flag to set or clear. See below.

  2. The name of the macro.

  3. 0 to clear the flags.

```
MSC_SET_FLAGS"80,#?,0"
     Clears the 'Write Protect" flag in all macros.
```

   The flags are:

   * 80 = Write Protect

   * 40 = Disable

   * 20 = Temporary

   * 10 = Relative

   * 08 = Gadget

   * 04 = Non recursive

   * 02 = Global

   See Flags.

1.205 MachV.guide/MSC_SPEAK

```
MSC_SPEAK
=========
```

   This function turns message speaking on and off. It takes one
argument.

```
MSC_SPEAK"Tog"
```
     Turn the speaking on or off.

```
MSC_SPEAK"Off"
```
     Turn the speaking off.

```
MSC_SPEAK"On"
```
     Turn the speaking on.

1.206   MachV.guide/MSC_STATUS

```
MSC_STATUS
==========
```

   This function checks the condition of various flags. It takes one
argument, a status flag. It can be called only from ARexx. Result is "1"
if the status is true.

     MSC_STATUS"7" Check to see if the clock is on.

   See MachV Status, for all flags.

1.207   MachV.guide/MSC_SUSPEND

```
MSC_SUSPEND
===========
```

   This function suspends MachV. It takes no arguments.

     MSC_SUSPEND Suspend MachV.

   See Suspend.

   Note

   You must use the same hotkey to un-suspend MachV.

1.208   MachV.guide/MSC_TRACE

```
MSC_TRACE
=========
```

    This function turns tracing on and off. It takes two arguments.

```
MSC_TRACE"Tog,S"
      Toggle serial tracing.

MSC_TRACE"Tog,C"
      Toggle console tracing.

MSC_TRACE"off,S"
      Serial tracing off.

MSC_TRACE"off,C"
      Console tracing off.

MSC_TRACE"on,S"
      Serial tracing on.

MSC_TRACE"on,C"
      Console tracing on.
```

    See Trace.

1.209 MachV.guide/PAU_SECONDS

```
PAU_SECONDS
===========
```

    This function pauses for the specified number of seconds. Whole
seconds and tenths are allowed. MachV stops functioning during this
time.

```
PAU_SECONDS"n.n"
      Pause for n.n seconds.
```

1.210 MachV.guide/PAU_FIX_INPUT

```
PAU_FIX_INPUT
=============
```

    This function pauses for a fixed number of keystrokes. It has one
argument, the number of keystrokes.

```
PAU_FIX_INPUT"3"
      Pause for 3 keystrokes.
```

1.211   MachV.guide/PAU_VAR_INPUT

```
PAU_VAR_INPUT
=============
```

   This function pauses for a variable number of keystrokes. It has two
optional arguments, the key to terminate input (in hex) and a variable
to store the input in. Default termination key is <ESC>, hex 45.

```
PAU_VAR_INPUT",varname"
```
     Pause for keystrokes until <ESC> is typed, then store input in
     varname. See Environment Variables.

```
PAU_VAR_INPUT"44"
```
     Pause for keystrokes until <Return> is typed (hex 44).

1.212   MachV.guide/SCR_BACK_TO_FRONT

```
SCR_BACK_TO_FRONT
=================
```

   This function sends the back screen to the front. It takes no
arguments.

```
SCR_BACK_TO_FRONT
```
     Send the back screen to the front.

1.213   MachV.guide/SCR_TO_FRONT

```
SCR_TO_FRONT
============
```

   This function sends the titled screen to the front. It takes one
argument, a title. AmigaDos wildcards are ok.

```
SCR_TO_FRONT"Workbench"
```
     Send the Workbench screen to the front.

1.214   MachV.guide/SCR_FRONT_TO_BACK

```
SCR_FRONT_TO_BACK
=================
```

   This function sends the front screen to the back. It takes no
arguments.

SCR_FRONT_TO_BACK
     Send the front screen to the back.

1.215 MachV.guide/SCR_TO_BACK

SCR_TO_BACK
===========

   This function sends the titled screen to the back. It takes one
argument, a title. AmigaDos wildcards are ok.

SCR_TO_BACK"Workench"
     Send the Workench screen to the back.

1.216 MachV.guide/SCR_WB_TO_FRONT

SCR_WB_TO_FRONT
===============

   This function sends the Workbench screen to the front. It takes no
arguments.

SCR_WB_TO_FRONT
     Send Workbench screen to the front.

1.217 MachV.guide/SCR_WB_TO_BACK

SCR_WB_TO_BACK
==============

   This function sends the Workbench screen to the back. It takes no
arguments.

SCR_WB_TO_BACK
     Send Workbench screen to the back.

1.218 MachV.guide/SCR_SELECT_SCREEN

SCR_SELECT_SCREEN
=================

   This function opens a list box of available screens. It takes one
optional argument, a title for the list box. When a screen is selected,

it will be brought to the front.

SCR_SELECT_SCREEN
     Open a list box to select a screen from. Use the default list box
     title "Select A Screen".

SCR_SELECT_SCREEN"Kill Screen?"
     Open a list box to select a screen from. Use the list box title
     "Kill Screen?"

1.219   MachV.guide/SCR_KILL_SCREEN

SCR_KILL_SCREEN
===============

     SCR_KILL_SCREEN

   Request confirmation then close the front screen. USE WITH CAUTION!
Some programs in some situations do not close their custom screen when
they terminate. With this function, you can close it, but the system may
crash if the program still expects to have its screen!

1.220   MachV.guide/SET_DATE_FORMAT

SET_DATE_FORMAT
===============

   This function sets the date format to use. It takes 1 argument.

DOS
     File system format. 18-Feb-92

INT
     International.   92-02-18

USA
     United States.   02-18-92

CDN
     Canadian.   18-02-92

SET_DATE_FORMAT"INT"
     Set date format to INT (international).

1.221   MachV.guide/SET_DATE_SEPARATOR

```
SET_DATE_SEPARATOR
==================
```

   This function sets the separator character. It takes 1 argument.

```
SET_DATE_SEPARATOR"/"
```
     Set the separator character to /. "-" is default. e.g. 01-01-92

1.222   MachV.guide/SET_NO_LAYER

```
SET_NO_LAYER
============
```

   This function sets the window titles that you do not want layered
with the function WIN_LAYER_WINDOWS.

```
SET_NO_LAYER"Workbench,MachV"
```
     Set the titles Workbench and MachV so that WIN_LAYER will not
     affect them. Maximum of 10 titles of 10 characters each.

1.223   MachV.guide/SET_PASSWORD

```
SET_PASSWORD
============
```

   You may set the screen unblanking password with this function.
Normally you set it in the Miscellaneous Features window. If set, the
password will be required before the screen will unblank.

   Example:

   Set_Password"foo" Msc_Blank_Screen

   Sets the password to "foo" and blanks the screen.

   See Password.

1.224   MachV.guide/SET_MOUSE_PORT

```
SET_MOUSE_PORT
==============
```

   This function sets which port to use for the mouse. It takes 1
optional argument, the port number 0 or 1.

```
SET_MOUSE_PORT"1"
```

```
    Set mouse port to port 1.

SET_MOUSE_PORT"0"
    Set mouse port to port 0.

SET_MOUSE_PORT
    Toggle the mouse port.
```

1.225 MachV.guide/SET_TASK_PRIORITY

```
SET_TASK_PRIORITY
=================
```

   This function sets the priority of a task. It takes 2 arguments, a
task name and a priority.

```
SET_TASK_PRIORITY"MyApp,1"
    Set the priority of a "MyApp" to 1.
```

   Task names can be found by clicking on "Select Title" in SetMachV.

1.226 MachV.guide/SET_VARIABLE

```
SET_VARIABLE
============
```

   This function sets an environment variable. It takes 2 required
arguments and 1 optional argument.

  1. Variable name.

  2. The value to set it to.

  3. Optional flags.

```
SET_VARIABLE"file,test.c,100"
    Sets the global environment variable 'file' to 'test.c'. The
    optional '100' indicates global.
```

   The optional flags are the hexidecimal numbers:

```
0
    Local then global

100
    Global

200
    Local
```

```
1000
     Save Variable to ENVARC:

SET_VARIABLE"fname,readme,1100"
     This sets the global variable fname to 'readme' and saves it to
     ENVARC:
```

1.227 MachV.guide/TXT_TIME_PASTE

```
TXT_TIME_PASTE
==============
```

   This function pastes the current time. It has 2 optional arguments.

```
TXT_TIME_PASTE
     Paste (send as kestrokes) the current time.

TXT_TIME_PASTE"am,pm"
     The optional time function arguments "am,pm" cause the argument
     text to follow the time. If the time is before noon, the first
     argument will be sent otherwise the second is sent. This text may
     be anything upto 80 characters. Example: TXT_TIME_PASTE" ante
     meridiem, post meridiem"
```

1.228 MachV.guide/TXT_TIME_PRINT

```
TXT_TIME_PRINT
==============
```

   This function prints the current time. It has 2 optional arguments.

```
TXT_TIME_PRINT
     Print the current time.

TXT_TIME_PRINT"am,pm"
     The optional time function arguments "am,pm" cause the argument
     text to follow the time. If the time is before noon, the first
     argument will be sent otherwise the second is sent. This text may
     be anything upto 80 characters. Example: TXT_TIME_PRINT" ante
     meridiem, post meridiem"
```

1.229 MachV.guide/TXT_DATE_PASTE

```
TXT_DATE_PASTE
==============
```

   This function pastes the current date. It has one optional argument,

"long", which causes the long form of the date to be used. The short
form is the same as selected for the clock.

TXT_DATE_PASTE
     Paste (send as keystrokes) the current date.

TXT_DATE_PASTE"long"
     Paste the long form of the current date. Example: December 7, 1993

1.230   MachV.guide/TXT_DATE_PRINT

TXT_DATE_PRINT
==============

   This function prints the current date. It has one optional argument,
"long", which causes the long form of the date to be used. The short
form is the same as selected for the clock.

TXT_DATE_PRINT
     Print the current date.

TXT_DATE_PRINT"long"
     Print the long form of the current date. Example: December 7, 1993

1.231   MachV.guide/WIN_ACTIVATE

WIN_ACTIVATE
============

   This function activates a window. It takes 1 argument, a window
title, (AmigaDos wildcards ok, or 'A', 'T', or 'M').

     WIN_ACTIVATE"title" Activate the window by the title.

WIN_ACTIVATE"M"
     Activate the window under the mouse.

WIN_ACTIVATE"T"
     Activate the top window.

   This function is often used after a SCR_FRONT_TO_BACK and
SCR_BACK_TO_FRONT.

   SCR_BACK_TO_FRONT WIN_ACTIVATE"T"

1.232   MachV.guide/WIN_ACTIVATE_NEXT

```
WIN_ACTIVATE_NEXT
=================
```

   This function activates the next window in the list maintained by the
operating system. It is often used to cycle through all windows. It can
take 1 argument, a window title (AmigaDos wildcards ok). Examples:

```
WIN_ACTIVATE_NEXT
     Activate the next window.

WIN_ACTIVATE_NEXT"~(MachV#?)"
     This will activate the next window that does not begin with a
     title 'MachV'.

WIN_ACTIVATE_NEXT"~(Csh#?|Workbench#?)"
     This will exclude any window with a title beginning with 'Csh' or
     'Workbench'.

WIN_ACTIVATE_NEXT"(GNU#?|Csh#?|Workb#?)"
     This will activate only windows that begin with 'GNU', 'Csh' or
     'Workb'.

   *Note*
```

   A window with no title pointer (w->Title == NULL) will match no
matter what your pattern is.

1.233   MachV.guide/WIN_TO_FRONT

```
WIN_TO_FRONT
============
```

   This function brings a window to the front. It takes 1 argument, a
window title, (AmigaDos wildcards ok, or 'A', 'T', or 'M').

```
WIN_TO_FRONT"title"
     Move the titled window to the front.

WIN_TO_FRONT"A"
     Move the active window to the front.

WIN_TO_FRONT"M"
     Move the window under the mouse to the front.
```

1.234   MachV.guide/WIN_TO_BACK

```
WIN_TO_BACK
===========
```

This function sends a window to the back. It takes 1 argument, a
window title, (AmigaDos wildcards ok, or 'A', 'T', or 'M').

```
WIN_TO_BACK"title"
     Move the titled window to the back.
```

```
WIN_TO_BACK"A"
     Move the active window to the back.
```

```
WIN_TO_BACK"T"
     Move the top window to the back.
```

```
WIN_TO_BACK"M"
     Move the window under the mouse to the back.
```

1.235 MachV.guide/WIN_BACK_TO_FRONT

```
WIN_BACK_TO_FRONT
=================
```

This function sends the back window to the front. There are no
arguments.

```
WIN_BACK_TO_FRONT
     Move the back window to the front.
```

1.236 MachV.guide/WIN_FRONT_TO_BACK

```
WIN_FRONT_TO_BACK
=================
```

This function sends the front window to the back. There are no
arguments.

```
WIN_FRONT_TO_BACK
     Move the front window to the back.
```

1.237 MachV.guide/WIN_MOVE_WINDOW

```
WIN_MOVE_WINDOW
===============
```

This function moves a window. It takes 3 arguments:

1. Window title (AmigaDos wildcards ok, or 'A', 'T', or 'M').

2. X position or a change if +/-.

```
  3. Y position or a change if +/-.
```

WIN_MOVE_WINDOW"title,40,10"
    Move titled window to 40x, 10y. Use +0 to not move.

WIN_MOVE_WINDOW"title,+10,-20"
    Move titled window left 10 pixels and up 20.

WIN_MOVE_WINDOW"A,0,20"
    Move active window to 0x, 20y.

WIN_MOVE_WINDOW"A,+0,+30"
    Move active window down 30 pixels. do not move horizontally.

1.238 MachV.guide/WIN_LAYER

```
WIN_LAYER
=========
```

   This function arranges all windows into layers. It takes no
arguments.

```
WIN_LAYER
     Layer all windows on the screen.
```

   See SET_NO_LAYER.

1.239 MachV.guide/WIN_SIZE_WINDOW

```
WIN_SIZE_WINDOW
===============
```

   This function sizes a window. It takes 3 arguments:

  1. Window title (AmigaDos wildcards ok, or 'A', 'T', or 'M').

  2. Width or a change if +/-.

  3. Height or a change if +/-.

WIN_SIZE_WINDOW"title,640,200"
    Size titled window to 640x200.

WIN_SIZE_WINDOW"A,+10,-20"
    Increase width of the active window by 10 and decrease height by
    20 .

1.240   MachV.guide/WIN_CHANGE_WINDOW

```
WIN_CHANGE_WINDOW
=================
```

   This function changes the size and position of a window. It takes 5
arguments:

   1. Window title (AmigaDos wildcards ok, or 'A', 'T', or 'M').

   2. The left edge.

   3. The top edge.

   4. The width.

   5. The height.

```
WIN_CHANGE_WINDOW"title,le,te,w,h"
     Move and size the titled window to position le = left edge, te =
     top edge, w = width, h = height.

WIN_CHANGE_WINDOW"A,le,te,w,h"
     Move and size the active window.
```

1.241   MachV.guide/WIN_ZOOM_WINDOW

```
WIN_ZOOM_WINDOW
===============
```

   This function zooms a window. It takes 1 argument, a window title,
(AmigaDos wildcards ok, or 'A', 'T', or 'M').

```
WIN_ZOOM_WINDOW"title"
     Zoom the titled window.

WIN_ZOOM_WINDOW"A"
     Zoom the active window.

WIN_ZOOM_WINDOW"T"
     Zoom the top window.
```

1.242   MachV.guide/WIN_SEND_CLOSE

```
WIN_SEND_CLOSE
==============
```

   This function closes a window. It takes 1 argument, a window title,
(AmigaDos wildcards ok, or 'A', 'T', or 'M').

```
WIN_SEND_CLOSE"title"
     Send Close gadget message to titled window.

WIN_SEND_CLOSE"A"
     Send Close gadget message to active window.

WIN_SEND_CLOSE"T"
     Send Close gadget message to top window.
```

1.243   MachV.guide/WIN_SEND_GADGET_DOWN

```
WIN_SEND_GADGET_DOWN
====================
```

   This function sends a gadget down message to a window. It takes 2
arguments, a window title (AmigaDos wildcards ok, or 'A', 'T', or 'M')
and the gadget id number.

```
WIN_SEND_GADGET_DOWN"A,7"
     Send gadget down message to active window gadget number 7.

WIN_SEND_GADGET_DOWN"title,19"
     Send gadget down message to titled window gadget number 19.
```

1.244   MachV.guide/WIN_SEND_GADGET_UP

```
WIN_SEND_GADGET_UP
==================
```

   This function sends a gadget up message to a window. It takes 2
arguments, a window title (AmigaDos wildcards ok, or 'A', 'T', or 'M')
and the gadget id number.

```
WIN_SEND_GADGET_UP"A,34"
     Send gadget down message to active window gadget number 34.

WIN_SEND_GADGET_UP"title,9"
     Send gadget down message to titled window gadget number 9.
```

1.245   MachV.guide/WIN_SEND_MENU

```
WIN_SEND_MENU
=============
```

   This function sends a menu selection message to a window. It takes
2, 3 or 4 arguments:

1. Window title (AmigaDos wildcards ok, or 'A', 'T', or 'M')

2. Menu item text. If no item text, this is the menu number.

3. Menu item number.

4. Menu subitem number.

WIN_SEND_MENU"A,Quit"
     Find the menu item whose text matches "Quit" and send the
     appropriate menu message.

WIN_SEND_MENU"A,0,3,1"
     Send a menu selection to active window, menu number 0, item number
     3, subitem number 1 (optional).

WIN_SEND_MENU"title,0,2"
     Send a menu selection to titled window, menu number 0, item number
     2.

1.246   MachV.guide/WIN_SEND_OTHER

WIN_SEND_OTHER
==============

   This function sends any other IDCMP message to a window. It takes 4
arguments:

1. Window title (AmigaDos wildcards ok, or 'A', 'T', or 'M')

2. Message class.

3. Message code.

4. Message qualifier.

WIN_SEND_OTHER"title,400,11,8"
     Send <Ctrl-Q> IDCMP_RAWKEY (400), key Q (11), control (8) to the
     titled window. All numeric arguments are in hex.

1.247   MachV.guide/WIN_SELECT_GADGET

WIN_SELECT_GADGET
=================

   This function activates a string gadget in the active window. It
takes no arguments.

WIN_SELECT_GADGET
     Activate a string gadget in the active window.

1.248 MachV.guide/WIN_SELECT_WINDOW

```
WIN_SELECT_WINDOW
=================
```

   This function opens a list box of all windows on a screen. It has 1 optional argument, a title for the list box.

   When a window is selected, it will be made active. You may then perform some other action on it such as bringing it to the front.

WIN_SELECT_WINDOW
     Open the select list box. Use default list box title "Select A Window".

WIN_SELECT_WINDOW"All Windows"
     Open the select list box. Use the title "All Windows" in the list box.

1.249 MachV.guide/WIN_WAIT_WINDOW

```
WIN_WAIT_WINDOW
===============
```

   This function waits for a window to open. It has 1 required argument and 3 optional arguments.

  1. Window title (AmigaDos wildcards ok, or 'A', 'T', or 'M').

  2. U = Wait for UserPort to be set.

  3. M = Wait for menustrip to be set.

  4. Maximum number of seconds to wait (1 - 9 seconds).

WIN_WAIT_WINDOW"Title,U,M,4"
     Wait for a window to become active. Then wait for user port 'U', menu strip 'M' and maximum of 4 seconds.

WIN_WAIT_WINDOW"Serial,U,,6"
     Wait for a window to become active. Then wait for user port 'U' for a maximum of 6 seconds.

1.250 MachV.guide/WIN_WINDOW_OPEN

```
WIN_WINDOW_OPEN
===============
```

   This function is tests to see if a window is open. It is executed
only from ARexx.

```
WIN_WINDOW_OPEN"Title"
```
      Test to see if a window is open. This returns a '1' in the ARexx
      variable RESULT if the window is open.

1.251 MachV.guide/Escape Characters

```
Escape Characters
*****************
```

   Note that all are upper case except the arrow keys and Enter.

   Function Keys

```
   \F1  Function key 1     \F4  Function key 4     \F7  Function key 7
   \F2  Function key 2     \F5  Function key 5     \F8  Function key 8
   \F3  Function key 3     \F6  Function key 6     \F9  Function key 9
                                                   \F10 Function key 10
```
   Numeric Keypad Keys

```
   \1   Numeric Keypad 1  \4   Numeric Keypad 4  \7   Numeric Keypad 7
   \2   Numeric Keypad 2  \5   Numeric Keypad 5  \8   Numeric Keypad 8
   \3   Numeric Keypad 3  \6   Numeric Keypad 6  \9   Numeric Keypad 9
                                                 \10  Numeric Keypad 10
```
   \e   Numeric Keypad ENTER

   The following are for the usa keymap. For other keymaps use the
character on the keycap.

```
   \-  Numeric Keypad -              \*  Numeric Keypad *
   \(  Numeric Keypad (              \/  Numeric Keypad /
   \)  Numeric Keypad )              \.  Numeric Keypad .
   \+  Numeric Keypad +
```

   Other Special Keys

```
   \B  Backspace                \u  Up Arrow
   \D  Delete                   \d  Down Arrow
   \E  Escape                   \l  Left Arrow
   \H  Help                     \r  Right Arrow
   \N  Return (Newline)
   \T  Tab
   \   Backslash
```

   Qualifier Keys

```
   \A  Alternate
   \C  Control
```

```
\S  Shift
\L  Left Amiga
\R  Right Amiga
```

1.252 MachV.guide/Common Questions

```
Common Questions
****************
```

   Q Why can't I close my CLI window after running MachV?

      Use Run >nil: MachV. This will allow the CLI window to close.

   Q Why does MachV stop functioning after I run an application from a
      EXE_CMD"program" macro?

      Unless you use "Run" in the macro, MachV will wait for the
      application to terminate. You should use something like
      EXE_CMD"Run MyProgram".

   Q I can't drag select text or gadgets with the left mouse button?

      Click To Front is probably set to 1. Try another value.

   Q Why does my application crash when I run it from the NewShell or
      from an EXE_CMD"" macro?

      Programs executed with EXE_CMD (such as NewShell) will have a
      stack the size that was set when MachV was run. These and ones
      that you may run from a new shell may need a larger stack. Either
      set the stack before running MachV, or after you create a NewShell
      use the stack command to set your stack to a larger value. The
      default stack size is 4000. Many programs need 10000 or more.

   Q Why does my EXE_CMD function return an error code?

      If you are trying to start a program from a macro and you have
      entered the command as you normally would type it from the shell,
      the problem is often embedded quotes, commas or backslashes. For
      example, use:

      EXE_CMD"echo >speak: \"time for lunch\, now!\""

      Not:

      EXE_CMD"echo >speak: "time for lunch, now!""

      An error code is also returned if the program cannot be found. It
      is often helpful to turn "Speak Messages" and "Debug" on when
      trying to locate these problems.

   Q Why am I in the c: directory when I create a New Shell?

      MachV's current directory is the one that was current when MachV

was run. If you bring up a New Shell and find yourself in the c: directory for example, it is because that was the current directory when MachV was run. You probably did a "cd c:" somewhere before running MachV.

Q Why does the NewShell function start in wb_2.x:WBStartup?

You started MachV from this directory. In order to start a new shell with a current directory other than that, enter in your s:shell-startup file the command "cd Work:" or whatever is the directory that you wish to be current.

Q Why am I getting a Close Screen Vector message?

MachV replaces the CloseScreen vector with one that will allow MachV to close its clock and other windows. If another program also replaced the vector, you will be given the opportunity to terminate that program before the original vector is replaced. If you get a requester with the message Close Screen Vector Was Changed!, either terminate the program that changed the vector and click Retry or just click Cancel.

Q How can I display only the total memory in the clock?

You can fool the clock into displaying your total memory even if you have fast ram. In your startup-sequence, turn off your fast ram with a command like "NoFastMem", run MachV, then run "NoFastMem" again.

1.253 MachV.guide/Some Useful Macros

Some Useful Macros
****

The following are some macros that you might find useful. The suggested hotkeys may, of course, be changed to your own preferences.

Attach this to <Ctrl-Amiga-A> to send Workbench the Show All Files menu selection.

WIN_SEND_MENU"A,All Files" or WIN_SEND_MENU"A,1,7,1"

   or this multi-function macro:

WIN_SEND_MENU"A,All Files" | WIN_SEND_MENU"A,Only Icons"

Attach this to <Ctrl-Amiga-Q> to close the active window.

WIN_SEND_CLOSE"A"

Attach this to an easy to remember key to start your word processor.

```
    MSC_CHG_DIR"dh1:FinalCopy" EXE_CMD"Run FinalCopy -cl"
    --------------------------------------------------------------------

    C programmers might like this attached to <Left-Amiga-F>.

    "for (i=;i<;i++) {\l\l\l\l\l\l\l\l\l\l"

    And this attached to <Left-Amiga-P>.

    "printf(\"\n\" );\l\l\l\l\l\l\l"

    This will output a commented $VER: statement.

    "/* $VER: ()\l" set_date_separator"." txt_date_paste
    set_date_separator"-" "\r\N *\N*/\u\u\S\r\S\u\r\r\r\r"

    It will look like this: (with cursor before the (01.01.92))

    /* $VER:  (01.01.92)
     *
     */

    (Use SET_DATE_FORMAT"CDN" to output dd.mm.yy instead of mm.dd.yy)
    --------------------------------------------------------------------

    Give this a name of 640x200 and execute it by name to change
    screenmode and overscan. (This is all one macro)

    EXE_CMD"overscan sys:prefs/presets/overscan640x240.prefs use"
    EXE_CMD"screenmode sys:prefs/presets/screenmode640x240.prefs use"
    --------------------------------------------------------------------

    Name this "Kill Screen". CAUTION! This will crash unless the screen is
    REALLY lonely.

    SCR_SELECT_SCREEN"Screen To Kill?" SCR_KILL_SCREEN
```

1.254 MachV.guide/Copyright and Disclaimer

```
Copyright and Disclaimer
************************
```

```
    Disclaimer
```

This product is sold "as is" without representation or warranty of any kind, either expressed or implied. The user assumes all risks and responsibilities related to its use.

1.255 MachV.guide/Index

Index
*****

1.256 MachV.guide/Function Index

```
Function Index
**************
```

```
WIN_SEND_GADGET_UP                      WIN_SEND_GADGET_UP
WIN_SEND_MENU                           Sending IDCMP Messages
WIN_SEND_MENU                           Multi-Function Macros
WIN_SEND_MENU                           WIN_SEND_MENU
WIN_SEND_OTHER                          WIN_SEND_OTHER
WIN_SIZE                                Multi-Function Macros
WIN_SIZE_WINDOW                         Moving and Sizing
WIN_SIZE_WINDOW                         WIN_SIZE_WINDOW
WIN_TO_BACK                             WIN_TO_BACK
WIN_TO_FRONT                            Window Titles
WIN_TO_FRONT                            WIN_TO_FRONT
WIN_WAIT_WINDOW                         Window Waiting
WIN_WAIT_WINDOW                         WIN_WAIT_WINDOW
WIN_WINDOW_OPEN                         Window Open
WIN_WINDOW_OPEN                         WIN_WINDOW_OPEN
WIN_ZOOM_WINDOW                         Zooming
WIN_ZOOM_WINDOW                         Helpful Hints
WIN_ZOOM_WINDOW                         WIN_ZOOM_WINDOW
```