Genera 8.0 Reference Cards

Notation Conventions

The documentation uses a number of notation conventions — a kind of shorthand — to represent keys, Lisp objects, command names, and mouse commands in a simple way. To get the most out of the documentation, you should be familiar with these conventions.

Modifier Key Conventions

Modifier keys are designed to be held down while pressing other keys. They do not themselves transmit characters. A combined keystroke like META-X is pronounced "meta x" and written as m-X. This notation means that you press the META key and, while holding it down, press the X key.

Modifier keys are abbreviated as follows:

CONTROL	c-
META	m-
SUPER	s-
HYPER	h-
SHIFT	sh-
SYMBOL	sy-

Modifier keys can be used in combination, as well as singly. For example, the notation c-m-Y indicates that you should hold down both the CONTROL and the META keys while pressing Y.

Modifier keys can also be used, both singly and in combination, to modify mouse commands. For example, the notation sh-Left means hold down the SHIFT key while clicking Left on the mouse and c-m-Middle means hold down CONTROL and META while clicking Middle.

The keys with white lettering (like X or SELECT) all transmit characters. Combinations of these keys should be pressed in sequence, one after the other (for example, SELECT L). This notation means that you press the SELECT key, release it, and then press the L key.

LOCAL is an exception to this rule. Despite its white lettering, you must hold it down while pressing another key, or it has no effect. For example, to brighten the image on your monitor, you would hold down LOCAL while pressing B.

Documentation Conventions

This documentation uses the following notation conventions:

cond, **zl:hostat**	Printed representation of Lisp objects in running text.
RETURN, ABORT, c-F	keys on the Symbolics Keyboard.
SPACE	Space bar.
login	Literal typein.
(make-symbol "foo")	Lisp code examples.
(**function-name** *arg1* &optional *arg2*)	Syntax description of the invocation of **function-name**.
arg1	Argument to the function **function-name**, usually expressed as a word that reflects the type of argument (for example, *string*).
&optional	Introduces optional argument(s).
Show File, Start	Command Processor command names and Front-end Processor (FEP) command names appear with the initial letter of each word capitalized.
m-X Insert File, Insert File (m-X)	Extended command names in Zmacs, Zmail, and Symbolics Concordia appear with the m-X notation either preceding the command name, or following it in parentheses. Both versions mean press m-X and then type the command name.
[Map Over]	Menu items. Click Left to select a menu item, unless other operations are indicated. (See the section "Mouse Command Conventions".)
Left, Middle, Right	Mouse clicks.
sh-Right, c-m-Middle	Modified mouse clicks. For example, sh-Right means hold down the SHIFT key while clicking Right on the mouse, and c-m-Middle means hold down CONTROL and META while clicking Middle.

Mouse Command Conventions

The following conventions are used to represent mouse actions:

1. Square brackets delimit a menu item.

2. Slashes (/) separate the members of a compound mouse command.

3. The standard clicking pattern is as follows:

   - For a single menu item, always click Left. For example, the following two commands are identical:

     [Previous]
     [Previous (L)]

   - For a compound command, always click Right on each menu item (to display a submenu) except the last, where you click Left (to cause an action to be performed). For example, the following two compound commands are equivalent:

     [Map Over / Move / Hardcopy]
     [Map Over (R) / Move (R) / Hardcopy (L)]

4. When a command does not follow the standard clicking order, the notation for the command shows explicitly which button to click. For example:

   [Map Over / Move (M)]
   [Previous (R)]

Zmacs

Getting Help in Zmacs

HELP COMMAND

You can get several different kinds of help in Zmacs by first pressing the HELP key, then any of the following keys:

A	Displays documentation for all the commands whose name or the first line of whose documentation contains a string you supply. Try HELP A Register to see how this works.
C	Displays documentation for a command you supply. Try HELP C c-X c-S.
D	Displays documentation for an extended (m-X) command. Try HELP D Show Documentation. (You can also type HELP C m-X Show Documentation.)
L	Displays the last sixty keystrokes you typed. This can be useful if you're not sure what you've just done.
U	Offers to undo changes to the buffer.
V	Displays all the Zmacs variables whose names contain a substring you supply. Try point.
W	Finds out whether an extended command you supply is bound to a key. Try Save File.
SPACE	Repeats the most recent HELP command. For example, if you use HELP D, then press HELP SPACE, you are prompted for the name of another extended command.

HELP WITH EXTENDED COMMANDS

Extended commands (the m-X commands) put you in a small area of the screen with full editing capabilities (a *minibuffer*) for entering names and arguments. Several kinds of help are available in a minibuffer.

COMPLETE	Completes as much of the current command as possible.
HELP	Gives information about special characters and possible completions.
c-?	Shows commands that *start with* the characters you have typed.
END or RETURN	Completes the command, and then executes it.
c-/	Lists extended commands *containing* the strings you've typed so far.
Mouse-Right	Gives you a menu of completions for the command you are entering. Click Left on a completion in the menu to select it.

See the section "Zmacs Completion".

Zmacs Commands for Regions, Buffers, Screens, and Windows

Region Operations

c-SPACE	Sets the mark, a delimiter of a region. You move the cursor from mark to create a region; Zmacs marks (underlines) the region. Use with region commands such as c-W, m-W, c-Y, and m-X Format Region.
c-W	Kills region (c-Y yanks it back at point).
m-W	Copies the region into kill history (c-Y yanks it back at point).
c-Y	Yanks back the last thing killed.
c-X c-J	Changes the character style of a region to one that you specify.
c-X c-L	Lowercases the region.
c-X c-U	Uppercases the region.
c-m-\	Indents each line in the region; for example, c-0 c-m-\ indents each line to the left margin.
c-sh-C	Compiles the region.
c-sh-E	Evaluates the region.
m-X Write Region	Writes the region to a file you specify.

Buffer Operations

c-X c-F	Reads a file into a buffer for editing.
c-X B	Selects a different buffer you specify; default is the last one.
c-X c-B	Displays a menu of available buffers; lines are mouse-sensitive.
c-X K	Kills a buffer you specify; default is the current one.
m-<	Moves to the beginning of the current buffer.
m->	Moves to the end of the current buffer.
c-m-L	Selects the most recently selected buffer in this window (toggles between two buffers).

Screen Operations

SCROLL or c-V	Shows next screen.
m-SCROLL or m-V	Shows previous screen.
c-0 c-L	Moves the line where point is to line 0 (top) of the screen.
c-m-R	Repositions the window to display all of the current definition, if possible.
c-SCROLL	Exposes and scroll the typeout window forward one screenful.
c-m-SCROLL	Exposes and scroll the typeout window back one screenful.
m-X Split Screen	Makes several windows split among the buffers as specified.

Window Operations

c-X 2	Splits the screen in two windows, using the current buffer and the previously selected buffer (the one that c-m-L would select).
c-X 1	Resumes single window display, using the current window.
c-X O	Moves cursor to other window.
c-m-V	Shows next screen of the buffer in the other window; with a numeric argument scrolls that number of lines — positive for forward, negative for backward.
c-X 4	Splits the screen into two windows and asks what should be shown in the other window.
c-X ^	Changes the size of the current window by the number of lines you indicate with a numeric argument. A positive numeric argument expands the window, a negative argument shrinks it.
c-X 8	Splits the screen into two windows, showing the current region in the top window.
m-X Compare Windows	Compares two windows of a split screen, starting at current point in each, stopping at first difference between them, moving point in each window to that difference.

Other Useful Zmacs Commands

Commands for Yanking and Undoing

m-X Undo — Returns the buffer to its state before the last command, querying first. You can also invoke it by pressing c-m-? U, HELP U, or c-HELP U.

c-sh-U — Undoes the last change to the buffer without querying; repeating the command several times undoes successive changes in reverse order.

m-X Redo — Redoes the previous undo, querying first.

c-sh-R — Redoes previous undo without querying; repeating the command redoes successive uses of undo.

c-Y — Yanks back the last thing killed.

c-sh-Y — Yanks back the last killed text matching a string you supply.

m-Y — After a c-Y, yanks back things previously killed; used after a c-Y to cycle through the kill ring. After a c-sh-Y, yanks back killed text matching the string you supply.

m-sh-Y — After any yank command, prompts for a string and yanks the element, from the appropriate history, that matches the string.

c-0 c-Y — Displays the kill history; click on a line to yank the text back at point.

c-m-Y — Repeats the last minibuffer command.

c-m-sh-Y — Repeats the last minibuffer command that matches a string.

Writing Files

c-X c-S — Writes the current buffer into a new version of the current file name.

c-X c-W — Writes the current buffer into a file with a different name.

m-X Save File Buffers — Offers to save each file whose buffer has been modified.

m-X Kill or Save Buffers — Offers a menu of buffers to Kill or Save.

Search and Replace

c-S *string* — "Incremental" search; searches while you are entering the string; terminate search with END.

c-R *string* — "Incremental" backward search; terminate search with END.

c-? *string1* RETURN *string2* RETURN — Replaces *string1* with *string2* throughout the buffer.

m-? *string1* RETURN *string2* RETURN — Replaces *string1* with *string2* throughout the buffer, querying for each occurrence of *string1*; press SPACE to replace and move to next occurrence, RUBOUT to skip and move to next occurrence, or HELP to see all options (see HELP C m-?). Note that RETURN and other characters cancel the command.

Zmacs Movement and Deletion Commands

Zmacs offers many commands for moving around and deleting characters, words, and other entities. The commands are similar for different entities: For example, c-F moves the cursor forward one character and m-F moves it forward one word. These commands also take numeric arguments, so you can specify how many units to move: For example, c-3-F moves the cursor forward three characters and m-5-F moves it forward five words.

Character Operations

c-B	Moves left (back) a character.
c-F	Moves right (forward) a character.
RUBOUT	Deletes a character left.
c-D	Deletes a character right.
c-T	Transposes the two characters around point; at the end of a line, transposes the two characters before point — ht -> th

Word Operations

m-B	Moves left (back) a word.
m-F	Moves right (forward) a word.
m-RUBOUT	Kills a word left (c-Y yanks it back at point).
m-D	Kills a word right (c-Y yanks it back at point).
m-T	Transposes the two words around point (if only -> only if).
m-C	Capitalizes the word following point.
m-L	Lowercases the word following point.
m-U	Uppercases the word following point.

Line Operations

c-A	Moves to the beginning of the line.
c-E	Moves to the end of the line.
c-O	Opens up a line for typing.
c-X c-O	Closes up any blank lines around point.
c-P	Moves up (previous) a line.
c-N	Moves down (next) a line.
CLEAR-INPUT	Kills backward from point to the beginning of the line (c-Y yanks it back).
c-K	Kills from point to the end of the line (c-Y yanks it back).

Sentence Operations

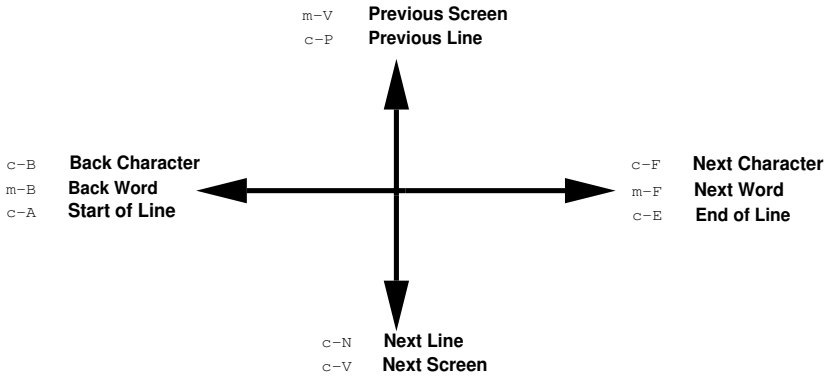
m-A	Moves to the beginning of the sentence.
m-E	Moves to the end of the sentence.
c-X RUBOUT	Kills backward from point to the beginning of the sentence (c-Y yanks it back at point).
m-K	Kills from point to the end of the sentence (c-Y yanks it back at point).

Paragraph Operations

m-[	Moves to the beginning of the paragraph.
m-]	Moves to the end of the paragraph.
m-Q	Fills the current paragraph (see HELP A Auto fill).
c-*n* c-X F	Sets the right margin to *n* (example: c-6 c-5 c-X F).

Summary of Zmacs Movement and Deletion Commands

	Forward	Backward	Begin	End	Delete Forward	Delete Backward	Transpose
Character	c-F	c-B			c-D	RUBOUT	c-T
Word	m-F	m-B	m-B	m-F	m-D	m-RUBOUT	m-T
Lisp Form	c-m-F	c-m-B	c-m-A	c-m-E	c-m-K	c-m-RUBOUT	c-m-T
Line	c-N	c-P	c-A	c-E	c-K	CLEAR INPUT	c-X c-T
Sentence	m-E	m-A	m-A	m-E	m-K	c-X RUBOUT	



Some Basic Zmail Commands

What the Zmail Menu Commands Do

These commands let you do almost anything in Zmail. Most of them are also available from the keyboard as single-keystroke commands, such as `S` for [Save].

For more information, see the mouse documentation line when you position the mouse over a command, or look through the Zmail documentation in "Zmail".

[Profile]	Customizes the Zmail environment.
[Configure]	Changes the Zmail window's appearance.
[Survey]	Displays summary lines of messages in an easily scrollable format.
[Sort]	Sorts messages by different criteria, such as Forward by Date.
[Quit], `Q`	Expunges and saves mail files, then sends you back to the window you used prior to Zmail.
[Save], `S`	Expunges and saves mail files, leaving you in Zmail.
[Get Inbox], `G`	Reads in your new mail.
[Map Over]	Performs an operation (such as Delete or Hardcopy) on all messages in the current mail sequence.
[Delete], `D`	Deletes the current message and moves to the next one.
[Next], `N`	Moves to the next message that is not deleted.
[Jump]	Selects a message based on a filter. (Note: This is *not* equivalent to `J`, which moves to a message in the current sequence.)
[Move], `O`	Moves the current message to another sequence.
[Undelete], `U`	Searches backward for a deleted message, "undeletes" and selects it.
[Previous], `P`	Moves to the last previous message that is not deleted.
[Keywords], `L`	Adds keywords to the current message.
[Next], `N`	Moves to the next message that is not deleted.
[Reply], `R`	Starts a reply to the current message.
[Continue], `C`, `RESUME`	Resumes sending a message.
[Mail], `M`	Starts a message.
[Other]	Repeats the last command given, or gives you a menu of additional commands.

Getting Help in Zmail

m-X Describe Command
: Displays detailed help text for a Zmail command you specify, similar to HELP D in Zmacs.

HELP
: Gives brief help for Zmail single-character commands and menu items. HELP * gives brief help for all Zmail extended commands.

m-X Apropos
: Lists brief help for any Zmail command whose name or first line of documentation contains the string you supply; similar to HELP A in Zmacs.

Moving Around in Zmail

Scrolling the Message Window

SCROLL, SPACE, and c-V
: Moves forward one screen in the message.

m-SCROLL, BACKSPACE, and m-V
: Moves back one screen in the message.

m->
: Moves to the last screen of the message.

m-<, .
: Moves to the first screen of the message.

Scrolling the Summary Window

c-m-V
: Scrolls the summary window forward.

c-m-sh-V
: Scrolls the summary window backward.

Moving among Messages

There are many ways you can select a message to read it, reply to it, delete it, and so on.

- Click Left on the message's line in the summary window.
- Click on [Next] or press N to select the next undeleted message.
- Click on [Previous] or press P to select the previous undeleted message.
- Type the number of a message, followed by J, to select a specific message. For example, 35J selects message #35.
- Use the m-X Occur command to get a mouse-sensitive list of lines from messages containing a string you supply.

Using Sequences

Your mail messages are grouped in *sequences*. A sequence is a mail buffer or a collection you have hand-selected; it's any group of messages you have in a Zmail buffer. You can use these sequences in much the same way you use buffers in Zmacs.

c-m-L	Selects the mail sequence you most recently used before the current one. Numeric arguments (such as c-2 c-m-L) reselect the second most recent sequence you used, and so on.
[Select (R)]	Clicking Right on the [Select] menu item gives you a menu of sequences from which to choose.
c-X c-F	Reads a mail file you specify into Zmail. If no file with that name exists, creates an empty sequence that you can save into a mail file with the name you specified.
c-X c-S	Saves the current sequence. If you give it a numeric argument (such as c-0 c-X c-S), it expunges the sequence first.

Some Commonly Used Zmail Commands

Zmail offers several dozen extended commands. (In Zmail, press HELP, then * in response to the prompt, to see a complete list.) Here are a few useful ones; you'll undoubtedly add more of your own as you become more familiar with Zmail.

Commands Available by Clicking Right on a Summary Line

Zmail provides several different ways of doing things to messages, such as deleting, forwarding, and hardcopying them. In addition to the menu commands and extended commands, you can click Right on a message's line in the summary window to get a menu of different actions that can be taken on the message. Here are some of the commands available from this menu.

[Forward]	Forwards the current message to recipients you specify. This creates a new message and adds text you enter.
[Redistribute]	Resends the current message to recipients you specify. This is similar to forwarding, but doesn't create a new message or let you add text to the message.
[Select Conversation]	Creates a conversation of all messages referenced by the current message, and selects the conversation. This is useful for viewing all messages in an exchange.
[Keywords]	Associates keywords with the message.
[Move]	Moves the message to another sequence.

Miscellaneous Extended (m-X) Commands

Note that in Zmail, unlike in Zmacs, you can press just X rather than m-X to enter an extended command. Here are a few of the commands that are available.

m-X Add File References
: Places the pathname to a file in the header of a message. When the message is received, the pathname serves as the default for Zmail's Compile File, Edit File, Format File, Hardcopy File, Load File, and Show File commands.

m-X Bug
: Starts a bug report.

m-X Delete and Save Message
: Deletes the current message and saves its headers and text on the kill ring.

m-X Delete Conversation by References
: Deletes the conversation of all messages referenced by the current message. This is useful when you don't wish to see any of the messages in a long exchange.

m-X Describe Command
: Displays the full help text for a Zmail command.

m-X Edit Current Message, c-R, click Left on message window
: Lets you use all Zmacs commands to edit the current message. END saves the edits, ABORT discards them.

m-X Hardcopy Message
: Sends the current message to the default text printer.

m-X Kill Ring Save Message
: Saves the current message's text and headers on the kill ring, without deleting the message.

m-X Mark Survey
: Creates a collection from messages whose summary lines you mark.

m-X Revoke Message
: Either deletes a message you've already sent from the recipients' inboxes, or sends a message with a "Revokes" header if they have read it.

Some Useful CP Commands

The following commands are among those available in Lisp Listeners; use the Command Processor's Help command to see a complete list of commands. For more information on any of the commands: See the section "Dictionary of Command Processor Commands". When you're prompted for keywords, get in the habit of pressing HELP, c-/, or c-? to see what's available; many times, the keywords give you interesting options. And move the mouse around output generated by the commands — part or all of the output can be mouse-sensitive.

Getting Out of Trouble

Use the following commands with caution!

Reset Network	This resets all your existing network connections. If only one connection is a problem, use the File System display in Peek (SELECT P) to reset it.
Initialize Mouse	This is useful if your mouse has gotten lost or stuck.
Initialize Mail	Use this command only as a last resort when your Zmail is hopelessly stuck.
Halt Machine	This command puts you into the FEP, where you can execute any FEP command, including Start (to warm boot) and Boot (to cold boot).
Boot Machine	This commands boots your machine directly, without going to the FEP, using a boot file you specify. It warns you if any files are open and asks you to confirm that you want to halt and reboot your machine.

Operating on Files

Copy File	The :Create Directories Yes keyword lets you copy the file into a new directory.
Create File	Lets you create a new file or a new version of an existing file from a Lisp Listener or a breakpoint. This is particularly useful when you are using a remote terminal.
Delete File	A file you delete from a directory on a Symbolics computer remains until you expunge the directory.
Edit File	Enters the editor and reads in a file you specify. You can specify Simple Editor, which uses the input editor and is particularly useful from remote terminals.
Format File	Processes and displays a file containing Zmacs formatting directives. The :Destination keyword sends the output to a supported printer.
Hardcopy File	Keywords let you specify number of copies to print, whether to delete the file after printing, and several other characteristics.
Load File	You can compile a file before loading it with the :Compile keyword.
Rename File	You can't rename a file to a directory on another host.
Show File	If a file contains any Zmacs formatting directives, use Format File.

Operating on Directories

Create Directory	This creates subdirectories as well as top-level directories.
Edit Directory	The :Version keyword lets you specify the files you want to edit.
Expunge Directory	This removes deleted files. Remember, files don't really go away until they're expunged.
Show Directory	You can click `s-sh`-Middle on the command's output to change the way the files are listed.
Show FEP Directory	The keywords let you specify, among other things, a machine name, the type of files to display, and whether and how to highlight files in use.

Operating on the Output History

Clear Output History
: This defaults to the current window's history, but you can specify any dynamic window's history.

Copy Output History
: You can copy a window's output to any buffer, file, printer, stream, or window.

Show Output History
: You can show the history of any dynamic window at any location available through the :Output Destination keyword.

Using Software Systems

Distribute Systems
: Once you've created a software system, you use this command to put it on tape for distribution.

Load System
: If you load the same system(s) every time you boot, put a `(load-system)` form in your init file.

Load Patches
: This displays patch comments while it loads the patches; to turn off the display, specify :Show No.

Report Bug
: This is the same as `m-X` Bug in Zmacs and Zmail.

Show System Modifications
: Keywords let you identify patches made by particular authors on particular dates, and so on.

Other Useful CP Commands

The following commands are also available in Lisp Listeners; use the Command Processor's Help command to see a complete list of commands. For more information on any of the commands: See the section "Dictionary of Command Processor Commands". When you're prompted for keywords, get in the habit of pressing `HELP` or `c-?` to see what's available; many times, the keywords give you interesting options. And move the mouse around output generated by the commands — part or all of the output can be mouse-sensitive.

Looking at Lisp

Find Symbol — Keywords let you tailor the packages that are searched.

Set Lisp Context — The default context is Common-Lisp.

Show Expanded Lisp Code
Try expanding `(defmacro form ()())`.

Show Object — The :Type keyword lets you specify the type of object to look for.

Show Source Code — This works only when the code is currently in an editor buffer.

Getting Hardcopy

Hardcopy File — Keywords let you specify number of copies to print, whether to delete the file after printing, and several other characteristics.

Show Printer Status
The output is mouse-sensitive, and can be used as input to the Delete Printer Request command.

Delete Printer Request
Give it a request by clicking on the one of the requests displayed by the Show Printer Status command.

Set Printer — This doesn't do anything to the printer itself, but specifies which printer should handle your text and/or bitmap hardcopy requests.

Controlling Your Environment

Set Command Processor
This lets you change the Command Processor's mode and prompt, and say whether or not you want commands to prompt.

Show Command Processor Status
This tells you what the current mode and prompts are.

Set Time — This sets the local time on your machine.

Start GC — The keywords :Immediately By-Area and :Cleanup Ask do interesting things.

Show GC Status — Check the output to see if you need to garbage collect.

Getting a World

Copy Flod Files — On 3600-family machines, flod files are EPROM overlay files used by the FEP; on Ivory-based machines, these files are resident in the software.

Copy Microcode — The world you run on your machine must have the correct microcode.

Copy World — If your site doesn't support netbooting, you might need this command to copy new Lisp worlds. (Netbooting is not supported on Ivory-based machines.)

Other Useful Commands

Edit Namespace Object — Try editing the object `USER` *your-user-name*.

Expunge Directory — This removes all deleted files from the directory you specify.

Help — The :Format Detailed keyword gives you every available command.

Login — Logging in allows you to run customizations in your init file.

Select Activity — This duplicates the action of the `SELECT` key.

Show Differences — Using this command is like doing `m-X` Source Compare in Zmacs. Check the :Ignore keyword for interesting options.

Scan Mail — This lets you view messages from an inbox one at a time without entering Zmail.

Show Documentation — When you use this command at a Lisp Listener, a bookmark for the topic you display is placed in Document Examiner's Background Viewer.

Show Hosts — If you don't name specific machines, you get information on all machines on the Chaosnet to which your machine is attached.

Ending a Session

Save File Buffers — By default, it queries about each buffer to be saved.

Save Mail Buffers — With the :Expunge keyword, you can specify whether or not to expunge each buffer before saving it.

Logout — This command always asks you about saving file and mail buffers.

Customizing Your Environment

When you load a file or set a variable (for example, specifying that your hardcopies are sent to a certain printer, changing the character style of the screen display, or changing the appearance of the command prompt), you alter the default system behavior in your environment for the rest of the time you remain logged in. This type of per-session customization does not remain in effect in your machine after you log out or cold boot. If you load a file or set a variable for an intentionally temporary effect, this is fine.

However, if you decide that you want these changes to be put into effect every time you log in (permanently in your environment), you can save them in an *init file*, thereby instructing the system to automatically execute this sequence of commands every time you log in.

An init file is typically named `lispm-init.lisp`, and lives in your home directory.

The following functions are often used in init files. For more information on each function, see its documentation.

tv:set-screen-options &rest *vars-and-vals* &key *(:screen* **tv:main-screen***)* &allow-other-keys
Allows you to set the screen options under program control.

si:*kbd-repeat-key-enabled-p*
Controls whether or not the REPEAT key is enabled.

si:*kbd-repetition-interval*
Controls the speed of repetition of characters when the REPEAT key is held down.

si:*kbd-auto-repeat-enabled-p*
Controls whether or not keys repeat if held down (auto-repeat).

si:set-auto-repeat-p *key* &optional *(state* **t***) (console* **sys:*console***)*
Allows you to specify keys that should not auto-repeat even if auto-repeat is enabled.

si:*kbd-auto-repeat-initial-delay*
Controls how long you must hold down a key before auto-repetition starts.

tv:screen-brightness *basic-screen*
Returns the brightness of the screen as a floating-point number between 0 and 1.

tv:*dim-screen-after-n-minutes-idle*
Controls the length of time a console must be idle before its screen dims.

tv:*screen-dimness-percent*
Controls the brightness value of the screen when it is dimmed.

sys:console-volume &optional *(console* **sys:*console***)*
Returns the current volume setting for the console.

si:*disable-noise-strings*
Controls whether the Command Processor prints prompts within command lines.

zwei:*history-menu-length*
The maximum number of history elements displayed.

hardcopy:set-default-text-printer *device*
Specifies the printer to be used for all of the hardcopy commands except the screen copy command.

hardcopy:set-default-bitmap-printer *device*
Specifies the printer to be used for screen copies (by the FUNCTION Q command).

zwei:*revert-unedited-buffers-for-new-versions*
Controls the prompting behavior of Refind File, Refind All Files, and Revert Buffer if a newer version of the buffer file exists on disk.

Debugger

Debugger Commands for Displaying Information

The Debugger commands below help you examine stack frames and analyze the information you receive.

Note that most Debugger commands have one or more corresponding key-binding accelerators, which means you can press a combination of one or more keys in place of the command. For example, you can type the accelerator c-m-Z instead of the command :Analyze Frame. In the list below, the accelerator for each command, if any, follows the command's name.

For more information on the Debugger, see the section "Debugger". For a complete list of Debugger commands, press c-HELP in the Debugger or see the section "Using the Debugger". Also, see the section "Summary of Debugger Commands".

Commands for Viewing a Stack Frame

:Show Argument (c-m-A)
: Displays the value of one or all arguments in the current frame.

:Show Frame (REFRESH, c-L, m-L)
: Redisplays the error message for the current frame, then lists the name of the function and its arguments in the current frame.

:Show Local (c-m-L)
: Displays the value of one or all local variables for the function in the current frame.

:Show Compiled Code (c-X D)
: Displays the disassembled code for a function.

:Show Source Code (c-X c-D)
: Displays the source code for a function. This command works only when your code resides in an editor buffer.

Commands for General Information Display

:Analyze Frame (c-m-Z)
: Analyzes the erroneous frame and locates the source code of the current error.

:Describe Last (c-m-D)
: Executes the Lisp **describe** function on the most recently displayed value and leaves * set to that value.

:Show Backtrace (c-B, m-B, c-m-B)
: Displays the control stack, which keeps a record of all active functions (those that have been called but have not yet returned).

:Show Special
: Displays the special-variable binding of a symbol in the context of the current frame's environment.

Debugger Commands for Stack Motion and Continuing Execution

The Debugger commands below help you move around in the stack, restart execution, and edit a function.

Note that most Debugger commands have one or more corresponding key-binding accelerators, which means you can press a combination of one or more keys in place of the command. For example, you can type the accelerator c-E instead of the command :Edit Function. In the list below, the accelerator for each command, if any, follows the command's name.

For more information on the Debugger, see the section "Debugger". For a complete list of Debugger commands, press c-HELP in the Debugger or see the section "Using the Debugger". Also, see the section "Summary of Debugger Commands".

Commands for Moving Through the Stack

:Bottom Of Stack (m->)
Moves to the bottom of the stack, displays the least recent frame, and makes that frame current.

:Next Frame (LINE, c-N, m-N, c-m-N)
Moves down one frame, to the next less-recent frame — the calling frame — displays information about that frame, and makes it current.

:Previous Frame (RETURN, c-P, m-P, c-m-P, c-m-U)
Moves up one frame, to the next most-recent frame — the frame that the current frame called — displays information about that frame, and makes it current.

:Set Current Frame (click Left on a frame)
Makes the frame you click on the current frame.

:Top Of Stack (m-<)Moves to the top of the stack — the frame where the error occurred — displays the most recent frame, and makes it current.

Commands to Continue Execution

:Abort (ABORT, c-Z) Depending on the context of its use: Returns to top level, returns to the previously invoked Debugger, or executes the restart-handler instruction that appears in the list of proceed and restart options.

:Proceed (RESUME) Depending on the context of its use: Continues the execution of the program or process that has been suspended, executes the proceed-handler instruction that appears in the list of proceed and restart options, or returns to the previously invoked Debugger.

:Reinvoke (c-m-R) Restarts execution of the function in the current frame.

:Return (c-R) Returns from the current frame.

Command to Enter Another Activity

:Edit Function (c-E)
Enters the Zmacs editor to bring up the current function or any other function for editing.

Getting Out of Trouble

Command Processor Commands

You can use several Command Processor commands to get out of trouble:

Initialize Mail	Reloads Zmail; use only when Zmail is hopelessly stuck.
Initialize Mouse	Restarts the mouse process; use when your mouse does not respond.
Reset Network	Resets all network connections to your machine; use when your machine cannot communicate with any other machines on your network.

Getting Unstuck

If your machine or a process becomes wedged to the point that it seems frozen, try pressing FUNCTION, then hold down CONTROL and press CLEAR-INPUT. Usually, this will unwedge you.

If all else fails, you can stop Lisp and go to the FEP by using the CP command Halt Machine. (On a MacIvory, you can use the Transfer to FEP choice on the Ivory pull-down menu.) If you can't get a Lisp Listener to respond, you can press h-c-FUNCTION. (Note that on the UX-family machine, you can use h-c-FUNCTION only at the cold load window, not at the Genera console.)

Note: You should do this only under drastic circumstances. Once you're in the FEP, you can type the FEP command Start to warm boot, if you think you left your world in a salvageable state; or, you can type the command Boot to cold boot your machine. For more information, see the section "Recovering From Errors and Stuck States".

Other Recovery Procedures

Sometimes the status line displays a process state that indicates a problem. Some of these problems can be solved as follows.

`Server Unit Lock`	Use `SELECT P` to go into Peek, then press `F` for File System. Click on the access path to the host, then on [Reset].
`Wait Forever`	Select a different window, then reselect the one you were in.
`Output Hold`	Press `FUNCTION ESCAPE`; if that puts you in the Debugger, use `ABORT`.
`Sheet Lock`	Same as for `Output Hold`.
`Arrest`	Press `FUNCTION - A` (that is, a three-key sequence).
`Lock`	Try `FUNCTION 0 S` to see if any windows want to type out. If that does not help, press `c-ABORT`.
`Selected`	Press `FUNCTION 0 S`.
`(no window)`	Use the mouse or `SELECT` key to select the window you want.

You can press `SUSPEND` to get to a Lisp read-eval-print loop. You can press `c-m-SUSPEND` to force the current process into the Debugger.

The SYMBOL Key

Printing Special Characters with the SYMBOL Key

You can generate special characters (primarily mathematical symbols) by pressing the SYMBOL key in combination with other keys. Here is the complete list of SYMBOL key combinations; you can look at this list online at any time by pressing SYMBOL-HELP.

```
▪  Center-Dot              Symbol-'
α  Alpha                   Symbol-A
^  And-sign                Symbol-q
ε  Epsilon                 Symbol-E
ʎ  Lambda                  Symbol-L
δ  Delta                   Symbol-D
±  Plus-Minus              Symbol-:
∞  Infinity                Symbol-i
⊂  Left-Horseshoe          Symbol-t
∩  Up-Horseshoe            Symbol-e
∀  Universal-Quantifier    Symbol-u
⊗  Circle-X                Symbol-*
←  Left-Arrow              Symbol-j
≠  Not-Equals              Symbol-Equal-sign
≤  Less-Or-Equal           Symbol-,
≡  Equivalence             Symbol-'
∫  Integral                Symbol-/
↓  Down-Arrow              Symbol-h
β  Beta                    Symbol-B
¬  Not-sign                Symbol-Minus-sign
π  Pi                      Symbol-P
ɣ  Gamma                   Symbol-G
↑  Up-Arrow                Symbol-g
⊕  Circle-Plus             Symbol-Plus-sign
∂  Partial-Delta           Symbol-p
⊃  Right-Horseshoe         Symbol-y
∪  Down-Horseshoe          Symbol-r
∃  Existential-Quantifier  Symbol-o
⇄  Double-Arrow            Symbol-l
→  Right-Arrow             Symbol-k
◊  Lozenge                 Symbol-Escape
≥  Greater-Or-Equal        Symbol-.
∨  Or-sign                 Symbol-w
```

Other SYMBOL-HELP Information

The SYMBOL-HELP display includes the following information about keys that perform special functions:

ABORT	Throws to command level.
c-ABORT	Throws to command level immediately.
m-ABORT	Throws out of all levels.
c-m-ABORT	Throws out of all levels immediately.
FUNCTION	Performs asynchronous commands.
SELECT	Selects a program.
REFRESH	Refreshes the screen.
CLEAR-INPUT	Erases typein.
NETWORK	Provides commands for Terminal program.
ESCAPE	Shows Input Editor history commands.
COMPLETE	Completes partial input.
c-m-FUNCTION	Runs keyboard macros (editor).
SUSPEND	Gets to a read-eval-print loop.
c-SUSPEND	Suspends immediately.
m-SUSPEND	Gets to the Debugger.
c-m-SUSPEND	Gets to the Debugger immediately.
RESUME	Continues from a break or error.
RETURN	Inserts a carriage return.
LINE	Goes to the next line and indent (editor).
END	Terminates input.
HELP	Prints information about the current context.
SYMBOL-HELP	Displays help for the SYMBOL and special-function keys.
SCROLL	Scrolls forward by screens.
m-SCROLL	Scrolls backward by screens.
h-c-FUNCTION	Stops the machine and connects you to the FEP. (On the UX machine, this works only at the cold load window, not the Genera console.)
LOCAL-G	Rings the bell (press the LOCAL and G keys simultaneously).
LOCAL-*n* LOCAL-C	(Where *n* is a digit from 1 to 4) changes the contrast.
LOCAL-D	Makes the screen dimmer. LOCAL-B makes it brighter.

LOCAL-Q	Makes the audio quieter. LOCAL-L makes it louder.

Note that LOCAL key combinations do not work on the UX-family or MacIvory machine.

The SQUARE, CIRCLE, TRIANGLE, and HYPER keys are reserved for user customizations.

The SELECT Key

You can select an activity (such as Zmail, a Lisp Listener, or Document Examiner) by first pressing the SELECT key, then another key, as shown below.

SELECT =	SELECT Key Selector
SELECT %	Metering Interface
SELECT C	Converse
SELECT D	Document Examiner
SELECT E	Editor
SELECT F	File system operations
SELECT I	Inspector
SELECT L	Lisp Listener
SELECT M	Zmail
SELECT N	Notifications
SELECT P	Peek
SELECT Q	Frame-Up
SELECT T	Terminal
SELECT X	Flavor Examiner

To create a new activity of the specified type, hold down the CONTROL key while typing the letter. For instance, to create a new Lisp Listener, press SELECT c-L. If you press SELECT by accident, press RUBOUT. (That is, SELECT RUBOUT does nothing.) You can also select activities by using the Select Activity command.

The FUNCTION Key

Note: You first press FUNCTION, then one of the keys in the first column below; you do not have to hold down FUNCTION while you press the second key.

The following are some of the more useful FUNCTION key commands; for a complete listing of FUNCTION key commands, press FUNCTION HELP.

C	Changes the screen from its current state (black-on-white or white-on-black) to the other state.
F	Displays a list of users logged in to various machines at a site or on a network. FUNCTION 0 F prompts you for the name of a machine and/or a user; see the FUNCTION HELP display to see if other numeric arguments (such as FUNCTION 1 F) produce additional listings.
M	Turns **MORE** processing for all windows on or off, depending on the current setting.
c-M	Turns **MORE** processing for the selected window on or off, depending on the current setting.
Q	Hardcopies the screen on the default printer. When you precede it a numeric argument, you get a menu that allows you to change the defaults for hardcopying.
c-Q	Hardcopies the current window on the default printer.
m-Q	Hardcopies the screen, without the status line, on the default printer.
S	Selects the window you last used before the current one. For example, if you are editing in Zmacs after having sent mail from Zmail, pressing FUNCTION S returns you to Zmail. Numeric arguments (such as FUNCTION 2 S) reselect the second most recent window you used, and so on.
T	Changes how the current window handles input and output notifications. See the FUNCTION HELP display for details.
CLEAR-INPUT	Discards typeahead.
ESCAPE	Helps correct stuck states such as Output Hold or Sheet Lock; these process states show up in the status line at the bottom of the screen.
RUBOUT	Does nothing; use it to cancel FUNCTION if you pressed it by accident.

FEP Commands

The following are the most commonly used FEP commands. Except where noted, each command is available on both 3600-family and Ivory-based machines.

For a complete description of all FEP commands, see the section "FEP Commands".

Add Paging File FEP Command
: Adds a file to be used for virtual memory swap space during the current boot session. On 3600-family machines, the syntax is Add Paging-File.

Boot FEP Command
: Executes the commands specified in a boot file. (On the MacIvory, this is automatically done for the user whenever Lisp is started).

Clear Machine FEP Command
: Clears Lisp memory. It is optional on Ivory-based machines.

Clear Paging Files FEP Command
: Clears the list of paging files added by the Add Paging File command. On 3600-family machines, the syntax is Clear Paging-Files.

Clear Screen FEP Command
: On 3600-family machines, clears the console's screen. On Ivory-based machines, it clears the Ivory FEP window.

Continue FEP Command
: Returns you to Lisp from the FEP.

Debug FEP Command
: Enters you at the FEP Debugger on 3600-family machines or the IFEP Debugger on Ivory-based machines.

Declare More Paging Files FEP Command
: Declares files you specify to be paging files in addition to those already declared. On 3600-family machines, the syntax is Declare More Paging-Files.

Declare Paging Files FEP Command
: Declares files you specify to be paging files. On 3600-family machines, the syntax is Declare Paging-Files.

Enable IDS FEP Command
: Enables the Incremental Disk Save facility.

Hello FEP Command
: Runs a file of commands, hello.boot, to initialize the machine.

Initialize Hardware Tables FEP Command
: Initializes the FEP's hardware tables.

Load Microcode FEP Command
: Loads microcode from the local disk into the 3600-family machine's memory.

Load World FEP Command
: Loads the Lisp world into the machine, and adds any declared paging files.

Mount FEP Command
: Mounts a specified device.

Netboot FEP Command
: Loads a netboot core from a local disk into the machine's memory.

Reset FEP FEP Command
: Initializes the FEP's memory.

Scan FEP Command
: Reads the specified overlay (flod) file.

Set Boot Options FEP Command
: Sets default values for *keywords* on the local Ivory-based machine.

Set Chaos Address FEP Command
: Sets the Chaos address of the local 3600-family machine. Ivory-based machines must use the Set Network Address FEP command.

Set Default Disk Unit FEP Command
: Sets the disk unit to which Lisp and the FEP should default for all subsequent disk references. This command is available only on 3600-family machines.

Set Display-string FEP Command
: Displays a string you specify on the front panel of Symbolics machine models 3600, 3640, 3645, 3670, and 3675.

Set Ethernet Address FEP Command
: Sets the Ethernet address of the local machine. On 3600-family machines, the syntax is Set Ethernet-Address.

Set LMFS FSPT Unit FEP Command
: Sets the default location for the file named `fspt.fspt` to the disk unit specified.

Set Network Address FEP Command
: Sets the primary network type and address of the local machine. On 3600-family machines, the syntax is Set Network-Address.

Set Prompt FEP Command
: Sets the FEP command prompt to a string you specify.

Show Directory FEP Command
: Displays the contents of a specified directory in the FEP file system.

Show File FEP Command
: Displays the contents of a file in the FEP file system.

Show Paging Files FEP Command
: Shows declared and added paging files. On 3600-family machines, the syntax is Show Paging-files.

Shutdown FEP Command
: Halts the FEP and powers down some machine models (querying appropriately).

Start FEP Command
: Transfers control to the loaded Lisp world.

format Function

BASIC FEATURES

SYNTAX

(**format** *destination control-string arguments*)

destination	**t** prints on the default device (usually the screen). **nil** doesn't print at all, but forces **format** to return a character string containing the output. Otherwise, *destination* must be a stream.
control-string	A character string that is printed verbatim, except for directives that start with ~ (tilde).
arguments	Displayed in place of the directives; the way the substitution is done depends on the specific directive.

PRINTING DATA

These directives each consume one argument

Print any data		*Print integers*		*Print floating-point numbers*	
~**A**	as with **princ**	~**B**	in binary	~**E**	in exponential notation
~**S**	as with **prin1**	~**O**	in octal	~**F**	in traditional notation
		~**D**	in decimal	~**G**	selects "best" notation
		~**X**	in hex		
		~*n*R	in radix n		

PRINTING BLANK LINES

These directives consume no arguments

~%	Print a carriage-return (like **terpri**)
~&	Print a carriage-return only if needed to get to beginning of line (like **fresh-line**)

COLUMN WIDTH AND OUTPUT PRECISION

AUTOMATICALLY INSERTING SPACES UP TO A DESIRED COLUMN WIDTH

~B, ~O, ~D, ~X, ~R, ~E, ~F, ~G pad on the left
~A, ~S pad on the right

Put the number of columns right after the tilde. Example: **~20D**

Exception! With **~R**, put the column width after the radix. Example: **~12,20R**

TAB TO DESIRED COLUMN

~*n*T	Inserts blank space to get out to column number *n*. Doesn't work for files.

LIMITING PRECISION OF FLOATING-POINT OUTPUT

~E, ~F, and **~G** can take a second prefix parameter that controls the number of digits printed after the decimal point. Example: **~20,4F** (This is much like FORTRAN's **F20.4.**)

PRINTING NUMBERS IN FANCY FORMATS

SIGNS AND COMMAS

~B, ~O, ~D, and **~X** allow an optional : and an optional @ just before the letter.

: (colon)	Puts in commas every three digits
@ (at-sign)	Puts in an explicit **+**-sign if the number is positive

Examples: **~20:D** **~@O** **~35:@B**

SELECTING THE PAD CHARACTER

~B, ~O, ~D, and **~X** usually pad on the left with spaces. You can make them pad with any character by supplying a second directive parameter. *Example:* **~10,'*D** pads with stars.

SPELLING OUT CARDINAL AND ORDINAL NUMBERS

~R	(With no radix) prints the argument in English, like **thirty-four**

Variants: **~:R thirty-fourth** **~@R XXXIV** **~:@R XXXIIII**

SELECTING SINGULAR AND PLURAL

~:P	Prints an **s,** but not if the previous argument was 1. *Example:* **"Pat ~R dog~:P."**
~:@P	Like above, but chooses between **y** and **ies.** *Example:* **"Pat ~R pupp~:@P."**

CONTROLLING CASE

~@(...~)	Capitalizes the first word of the enclosed text. *Example:* **"~@(~R~) dog~:P"**

Variants: **~:(Thirty-Four** **~:@(THIRTY-FOUR**

CONDITIONALS AND SELECTION

DO SOMETHING ONLY IF A BOOLEAN ARGUMENT IS TRUE

˜@[...˜] Uses the enclosure only if the next argument is non-**nil**. If the next argument is **nil**, it skips the enclosure and discards the argument.

SELECT BETWEEN TWO ACTIONS BASED ON A BOOLEAN ARGUMENT

˜:[...˜;...˜] Uses the first enclosure if the argument is **nil**, and the second otherwise.

SELECT FROM AMONG N ACTIONS BASED ON AN INTEGER ARGUMENT

˜[...˜;...˜;...˜] Takes an integer argument (starting with 0), and selects the enclosure indexed by it. There can be any number of enclosures.

˜n[... Overrides the argument and selects enclosure *n*.

˜:; If you use this as the last separator, the last enclosure becomes the default.

loop Syntax