LABORATORY FOR
COMPUTER SCIENCE

MASSACHUSETTS
INSTITUTE OF
TECHNOLOGY

A Primer for the Lisp Machine

Massachusetts Institute of Technology
Computation Structures Group Memo 237
Last updated : March 22, 1984

Bhaskar Guharoy

This document gives a basic introduction to the Symbolics 3600 machines and the CADR Lisp machines.

545 TECHNOLOGY SQUARE, CAMBRIDGE, MASSACHUSETTS 02139

Table of Contents

Introduction to the LISP Machine

1. Introduction

The LISP machine is a single user machine originally designed at the MIT Artificial Intelligence Laboratory for running Lisp programs efficiently. The CADR machines are exclusively MIT products, while the 3600 machines are products of Symbolics Inc. Almost all of the software running on the CADR machines have been written by Daniel Weinreb, David Moon, Richard Stallman and other people associated with them.

2. Getting started

Notation : In this manual, the following notation will be adhered to as far as possible, except where the usage is so obvious that no special font is needed.

Special function keys	<In this font>. For example, <Help>.
Non-Control Keys	**In this font.**
Format information	*In this font.*
System Names	**In this font.** Examples of systems are **Lisp**, **Editor** etc.

When you walk up to a free Lisp machine, look at the line of information displayed at the bottom of the screen (this line is called the **who-line**). The format of the who-line is :
<month/day/year hour/minute/second login-name **USER** : *string status-string>*
The machine is free if the *login-name* field is empty. The field *status-string* normally shows **Cold Booted** if the machine has been cold-booted, displays a string with a right-arrow (if writing) or a left-arrow (if reading) a file, or is empty (that is the usual case). If the clock is not getting updated, then the machine has crashed.

Booting

The booting sequences are different for the CADR and the 3600. There are two basic types of booting sequences possible, the warm boot and the cold boot.

Warm Booting : If the machine crashes, try warm booting it first. This way, the world that existed when the machine crashed will not be entirely thrown away.

CADR : Press both pairs of <Control> and <Meta> keys and the <Return> key simultaneously to warm boot the machine. It takes about 20 seconds to boot up.

3600 : Go to the Lisp window by typing <SELECT><L>. Then type
halt
to the Lisp listener. The 3600 responds by asking a question whose answer should be **Y** or **N**. Type **Y** if you really want the machine to be booted. **N** aborts the command for booting. If you replied **Y** to the question, you will find that **Fep>**
appears on the screen. Type
start

to warm boot. You may also press the <Help> key after the legend **Fep>** appears to find out the other options for booting.

Cold Booting : Cold booting of a machine is done either when the machine has crashed during a session and cannot be brought up by warm booting, or if you are starting a session. Cold booting causes the machine to throw out all previous bindings and clears the main memory. You may avoid the cold booting at the beginning of a session (cold booting does take some time!) if the who-line says **Cold-Booted** and the machine is up.

CADR : Press both pairs of <Control> and <Meta> keys and the <Rubout> key simultaneously to cold boot it. The Cadr requires about a minute to cold boot.

3600 : Observe the same actions as described for warm booting. Instead of typing **start** when
Fep>
appears on the screen, type
boot
to cold boot the machine. The machine can be completely flushed and booted by toggling the switches on the front panel of the 3600 cabinet. You are advised not to try this unless you know what you are doing.

3. Logging In

CADR
After booting up the machine, log into the machine by typing

```
(login 'your-login-name-on-host-computer 'name-of-host-computer t)
```

Thus, if your login name on the MIT-XX is Foobaz, then the login command is

```
(login 'foobaz 'xx t)
```

Any computer that is on the Chaosnet can serve as a host. A net host is needed since the Lisp machine does not store the files on its own disk. All the files that you create have to be stored on a computer on the net; that computer will be the file-server for the Lisp machine.

3600
Type **login foobaz** to the Lisp window. XX has been defined to be the default file-server for all the 3600 machines in the FLA group. Take further action depending on the responses of the 3600.

4. Getting Help

The Lisp machine has an extensive help facility built into it. Locate the <HELP> key on the keyboard. Use it whenever you have any doubts. The screen will display helpful information depending on the window in which you need help.

Press <Terminal> <Help>. This will give you all the information that you need to be able to use the <Terminal> key. (On the 3600 the <Terminal> key is called the <Local> key).

Press <System> <Help>. If you cannot understand what it says, read on. (On the 3600 this is the <Select> key).

Press <Help>. If you are typing into a Lisp window, it will give you a list of all the input editor commands that the Lisp-Listener understands. On other windows it will give you (hopefully) helpful information on the window that you are typing into.

5. The Window System

The Lisp machine has a number of windows that are created at the time of booting. Each window can have a different program running in it. Each window occupies a region on the screen. If the current window occupies the entire screen, then the other windows will not be displayed, but may still be active. These windows may be killed or new ones created by the user during the session. The main types of windows are **Lisp**, **Edit**, **Mail**, **Peek**, **Inspect** **Supdup**, **Telnet**, and **Converse**. All these windows can coexist simultaneously. However, since there is only one keyboard, one of the windows have to be **selected** so that your commands can be sent to it. Each window that is created by the system after cold booting occupies the entire screen. In this note, we shall not discuss multiple windows on single screen in any detail. On the 3600, the **Supdup** and **Telnet** windows are replaced by a single window called the **Terminal** window.

5.1. The Lisp Window

This window is normally characterized by the legend `Lisp Listener` *n* at the bottom left corner of the window, n being an integer. You can type in Lisp forms which will be immediately evaluated. Most of the editor commands of the Zmacs editor are recognized by the Lisp Listener. Type <Help> to find out the Input editor commands. <Control>C produces the last Lisp form that you typed to the listener. This is helpful for editing the forms and evaluating them immediately without having to go into the editor. Type <Terminal><Help> to find out about the meaning of the keys.

The default base for arithmetic that is used by the Lisp Listener is base 8. If you want the font to be decimal, then type

```
(setq base 10. ibase 10.)
```

Further details may be obtained from the **Lisp Machine Manual.**

If you have defined a function named `foo` in the Lisp window whose definition you want to edit and you do not have the definition in any editor buffer or in a file, then type

```
(ed 'foo)
```

An editor window will appear and the window asks you

```
Grind the definition of FOO in its own buffer? Y or N
```

Type `y`. The definition of `foo` will appear in another window and you may now edit it and evaluate it as outlined in the section on the editor.

The mouse The mouse is the small squarish box that is attached to the Lisp machine keyboard through an umbilical cord. It has three buttons/pads on it. The mouse controls the movement of the small arrow that is initially at the bottom right hand corner of the screen. Press the right button. A menu is displayed on the screen. This menu is referred to as the System menu. Select an item by moving the arrow over it. Selecting an item under the heading `Programs` is the same as giving a <System> command. To kill a window, select the `Kill` item. You should try out the items displayed on the left column of the menu on your own. If you are scared, ask *somebody who knows.*

The functions of the mouse keys are displayed on the line of information in reverse video just above the who-line.

5.2. The Editor

The editor commands are very similar to those of Emacs. You will be typing into an editor buffer. Following is a short summary of frequently used commands. <Control> and <Meta> are abbreviated as C- and M- respectively. Note that the definition of certain keys is different from that on the Heath terminals or the VT100. On the Lisp Keyboard, <Meta> means the key named **META**, and not the **ESCAPE** key. For command completions, use the key named **COMPLETE**.

C-X C-V	Visit a file and load into this buffer. If you have loaded and edited something in the current buffer, then you cannot load another file into it (unlike in Emacs). You need C-X C-F.
C-X C-F	Visit a file in a new buffer. The current buffer remains intact.
C-X C-S	Save the buffer in specified file. Normally the machine specifies a default file name. If you want any component of that name to be altered, then type in the name of the file in which you want the buffer to be stored in. For example, suppose the name of the buffer is **Main**; if you command **C-X C-S** a small rectangle appears below the buffer area with a notification about the default path name **xx:ps:<foo>main.lisp** (if you logged into XX as Foo and your directory is on PS). If you want the buffer to be stored in a file named **myfile.lisp** on the same directory, just type **myfile** and it will put in the rest of the pathname. If you want a different extension, say **mss**, then type **myfile.mss**. If you want it to be saved in a different directory, you must specify the rest of the pathname so that the machine accesses the correct directory on the correct file-server. If you have trouble with the saving mechanism (machine reporting **Invalid directory access** or some such thing, try specifying the entire pathname in the format **xx:ps:<foo>myfile.lisp**. If you still have trouble, contact a person who is familiar with the machine.
C-X C-W	Save buffer in specified file.
C-X b	Select a buffer.
M-X Evaluate Buffer	Evaluates the buffer assuming that the buffer contains Lisp forms. Suppose a function **FOO** is edited in the buffer. By evaluating the function definition of **FOO**, the definition is available in the lisp listener from which you went into the editor, i.e., the most recently active Lisp window.
M-X Compile Buffer	Compiles the buffer. The compiled definitions are available in the most recently active Lisp listener.
C-M-F	Go to the end of the current S-expression.

C-M-B	Go to the beginning of the current S-expression.

The common editor command descriptions are available in the Emacs manual.

Using the mouse Use the mouse to see some more command descriptions. Press the right key once. You will get a menu of the editor functions available through the mouse. Move the pointer over the menu item and then press <Help>. If you are unable to understand the information, contact a person who can explain it to you. Some of the interesting items that appear on the menu are :

Arglist	Gives the argument list of the name of the function that you type. It searches through the various Lisp buffers in the editor world.
Edit Definition	Edit the definition of the function. Type the name of the function.
Kill or Save Buffers	Meaning is obvious. Try it out.
Compile Region	Compiles the region. A region is the region of text between the point and the mark. The point is the place where the blinker currently is. The mark may be set by placing the blinker at one end of the region to be selected and typing <Control>@. The selected region is identified by the underlined part of the buffer.

5.3. The network system - Telnet and Supdup

You may log into any remote computer over the Chaosnet that serves each Lisp machine. The new 3600s are on the Ethernet rather than on the Chaosnet, but that does not affect the accessibility of the computers on the net. There are two network protocol programs in use - the Telnet and the Supdup. The latter is the more reliable of the two, especially with regard to terminal interfacing with other computers. Supdup is invoked by typing <System>S, or by selecting through the mouse. Telnet may be invoked through <System>T. Then provide the name of the computer to which you wish to be connected. If the connection can be established, then you may proceed with logging into the host computer as you would do over any other terminal. The Lisp machine now serves as a dumb terminal. Of course, the power of the lisp machine is a key-stroke away - just type <System>L to go to the Lisp world.

The 3600 machines do not offer the user the choice between **Supdup** and **Telnet** protocols. It offers instead a composite window called the **Terminal** window which may be invoked by typing <Select>T. The machine first tries to connect to the desired computer using the Supdup mechanism; if it fails in that it tries to use the Telnet facilities.

5.4. The Inspect facility

This is a facility for poking around in the Lisp data structures. It can be invoked by typing <System-I>. See Operating the Lisp Machine and the Lisp Machine Manual for more details.

6. Outline of an example session

In this section, <Control> and <Meta> keys are indicated by **C-** and **M-** respectively.

Step 1 : Locate a free Lisp machine.

Step 2 : Boot up the machine to clear it of the leftovers from the previous session.

Step 3 : Log into the machine.

Step 4 : Now the hacking can proceed. There are numerous possibilities. You may directly type in Lisp forms to the Lisp listener which evaluate the form immediately after you complete typing it in. If any errors occur, you may get back the last form you typed minus the last closing parenthesis (to prevent the Lisp listener from evaluating it again) by saying <Control>C. You may edit this form and evaluate by putting in the closing parenthesis.

Suppose you want to create a Lisp file containing some functions. Type <System>E. You are now in the editor. You will be typing into a buffer called **MAIN** or ***buffer-1*** or some other name.. After you have written down the stuff, save it in your directory using the commands **C-X C-S**. You may now load this buffer into the Lisp listener by saying **M-X Evaluate Buffer**. If any errors are detected, you may edit the buffer and then evaluate again. To compile the buffer, type **M-X Compile Buffer**. The compiled Lisp function will be loaded into the Lisp world. To visit another file, command **C-X C-F** and then supply the file name.

Suppose you logged in as **foobaz** on MIT-XX and your directory is on **ps:<foobaz>**. Suppose a file **foo.lisp** exists on your directory. Then to load the file into the Lisp world, type

```
(load 'foo)
```

The Lisp machine automatically assumes that you are on **ps:<foobaz>**, provided that you logged in correctly, and that the extension of the file is **lisp**. If you have the XX as the file server and want to load a file **frob.lsp** from **src:<sys>**, then type

```
(load "oz:src:<sys>frob.lsp")
```

If an error occurs in the evaluation of a Lisp form, you may enter the window-oriented debugger by commanding <Control-Meta>W. Then press <Help> for more information.

Step 5 : Now we have reached the end of the session. You may kill all the editor buffers using the command **M-X Kill Or Save Buffers** and using the mouse as required.

Logging out is an important part of using a Lisp machine. The command is

```
(logout)
```

Do not forget to do this, else others may think that you are still using the machine and will not use it. After logging out, you may flush the machine by cold booting it. It is good practice; since garbage is not left over from session to session, more store is available, there is less paging and the system performance improves.

7. Further reading

The following manuals should provide additional information.

Operating the Lisp Machine

Gives more information on the Lisp machine environment. Particularly good once you have used the Lisp machine a couple of times and want to step beyond this primer.

Lisp Machine Manual

A mammoth compilation of all the features that the Lisp environment supports. It becomes outdated immediately after it is published, due to the numerous patches that are made on the Lisp systems every few days. Should be consulted when you are quite familiar with the machine. The following chapters might be found to be helpful at the beginning - Ch. 1-3 (for an introduction to Lisp), 4.1, 4.2, 4.4 (these sections are on iterations and conditional constructs), 5.1 - 5.7, 5.9 (these are on list manipulation), 6.1 - 6.4, 7.1 - 7.6, Introduction of Ch.8, 8.2 - 8.3 (arrays in Lisp), Ch. 10 (upto 10.3 for a good introduction to Lisp functions), 11.1 (on function closures), 16.1 - 16.2 (on the Lisp compiler), 21.1 (only the introductory portions), 21.2.1 - 21.2.2 , 21.3 - 21.4, 21.9.1 (Ch.21 is on I/O), Ch 22 (upto the introduction of 22.2 for the filing system; 22.2.1 is for later), Ch. 23 for a good introduction to the Chaosnet (upto 23.1), Introduction of Ch. 27, 27.7 - 27.8, 27.11 (all these are very helpful for debugging; get into the debugger upon an error and press <Help>), 32.7 (for login information), 32.8 (for dribble files - the output of a lisp session is directed to a file specified called **`Dribble file`** or **`Wallpaper file`** due to the verbosity of the output).

Programming in Lisp

by Pat Winston and B.K.P Horn : An excellent primer for Lisp progamming.

The Window System Manual

This is for the advanced users. You can do really fancy stuff with this. You are now on your way to become a member of that distinguished body of humans who are classified as **The Lisp Hackers.**