Chapter 3

Loading and Executing a Program

This chapter describes how to load and run programs within Prism. To learn:

- **How to load a program into Prism,** see Section 3.1.
- **How to associate a core file with a loaded program,** see Section 3.2.
- **How to attach to and detach from a running process,** see Section 3.3.
- **How to attach to a CM-2 or CM-200 and perform other CM-related tasks**, see Section 3.4. This section is for CM-2 and CM-200 users only.
- **How to execute a program,** see Section 3.5.
- **How to change the current file and the current function,** see Section 3.6.
- **How to specify the directories to be searched for source files,** see Section 3.7.

We assume in this chapter that you already have an executable program that you want to run within Prism. You can also develop the program from scratch by calling up an editor within Prism; see Chapter 7.

3.1 Loading a Program

Before you can execute, debug, or analyze the performance of a program in Prism, you must first load the program into Prism. Only one program can be loaded at a time.

Copyright © 1991–1994 Thinking Machines Corporation

As described in Chapter 2, you can load a program into Prism by specifying its name as an argument to the **prism** command. If you don't use this method, you can load a program once you are in Prism by using one of the methods described below.

3.1.1 From the Menu Bar

Choose the **Load** selection from the **File** menu. (It is also by default in the tear-off region.) A dialog box appears, as shown in Figure 9.

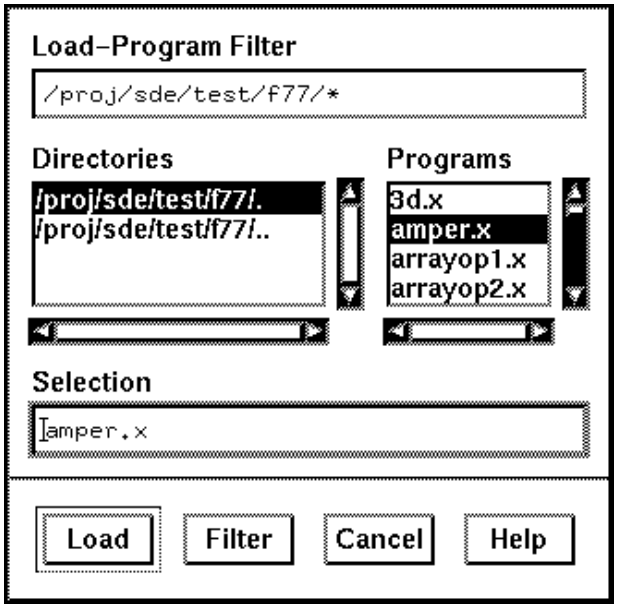

Figure 9. The Load dialog box.

To load a program, you can simply double-click on its name, if the name appears in the **Programs** scrollable list. Or, you can put its pathname in the **Selection** box, then click on **Load**. To put the file's pathname in the **Selection** box, you can either type it directly in the box, or click on its name in the **Programs** list. The **Programs** list contains the executable programs in your current working directory; see Section 2.10.1.

Use the **Load-Program Filter** box to control the display of filenames in the **Programs** list; the box uses standard UNIX filters. For example, you can click on a directory in the **Directories** list if you want to change to that direc-

Version 2.0, April 1994
Copyright © 1991–1994 Thinking Machines Corporation

tory. But the **Programs** list does not update automatically to show the programs in the new directory. Instead, the filter changes to *directory-name*/*, indicating that all programs in *directory-name* are to be displayed. Click on **Filter** to display the filenames of the programs. Or simply double-click on the directory name in the **Directories** list to display the programs in the directory.

If you want to use a different filter, you can edit the **Load-Program Filter** box directly. For example, change it to *directory-name*/*prog**** to display only programs beginning with *prog*.

Click on **Cancel** or press the **Esc** key if you decide not to load a program.

3.1.2 From the Command Window

Issue the **`load`** command on the command line, with the name of the executable program as its argument. For example:

```
load myprogram
```

The program you specify is loaded.

3.1.3 What Happens When You Load a Program

Once a program is successfully loaded:

- The program's name appears in the **Program** field in the main window.
- The source file containing the program's main function appears in the source window.
- The **Load** dialog box disappears (if you loaded the program using this box).
- The status region displays the message **not started**.

You can now issue commands to execute, analyze, and debug this program.

Copyright © 1991–1994 Thinking Machines Corporation

If Prism can't find the source file, it displays a warning message in the command window. Choose the **Use** selection from the **File** menu to specify other directories in which Prism is to search; see Section 3.7.

3.1.4 Loading Subsequent Programs

Only one program can be loaded at a time. If you have a program loaded and you want to switch to a new program, simply load the new program; the previously loaded program is automatically unloaded. If you want to start fresh with the current program, issue the **`reload`** command with no arguments; the currently loaded program is reloaded into Prism.

3.2 Associating a Core File with a Loaded Program

As mentioned in Chapter 2, you can have Prism associate a core file with a program by specifying its name after the name of the program on the **`prism`** command line.

You can also do this by loading the program and then issuing the **`core`** command, specifying the name of the corresponding core file as its argument.

In either case, Prism reports the error that caused the core dump and loads the program with a **stopped** status at the location where the error occurred. You can then work with the program within Prism. You can, for example, examine the stack and print the values of variables. You cannot, however, continue execution from the current location; if you were running the program on a CM-2 or CM-200, you cannot print the values of parallel variables or CM-resident arrays.

3.3 Attaching to and Detaching from a Running Process

As we described in Chapter 2, you can load a running process into Prism by specifying the name of the executable program and the process ID of the corresponding running process on the **`prism`** command line.

Version 2.0, April 1994
Copyright © 1991–1994 Thinking Machines Corporation

You can also attach to a running process from within Prism. To do this:

1. Find out the process's process ID by issuing the UNIX command **ps** (or **cmps** if the process is running on a CM-5 or under timesharing on a CM-2 or CM-200).
2. Load the executable program for the process into Prism.
3. Issue the **attach** command on the Prism command line, using the process's process ID as the argument.

With either method of attaching to the process, the process is interrupted; a message is displayed in the command window giving its current location, and its status is **stopped**. You can then work with the program in Prism as you normally would. The only difference in behavior is that it does not do its I/O in a special Xterm window; see Section 3.5.2.

To detach from a running process, issue the command **detach** from the Prism command line. The process continues to run in the background from the point at which it was stopped in Prism; it is no longer under the control of Prism. Note that you can detach any process in Prism via the **detach** command, not just processes that you have explicitly attached.

3.4 Attaching to and Detaching from a CM-2 or CM-200

NOTE: Read this section only if you are going to run programs on a CM-2 or CM-200 series Connection Machine system.

To run a program on a CM-2 or CM-200, you must be attached to a CM resource (that is, one or more sequencers on the CM). You can attach in any of the standard ways. For example, you can issue the **cmattach** command before entering Prism, and then issue the **prism** command from a **cmattach** subshell.

You can also attach, detach, cold boot, turn Paris safety on and off, and obtain information about CM users from within Prism. To attach, you must be running Prism from a front end that is physically connected to a CM-2 or CM-200 series Connection Machine system. The **CM** menu discussed in this section does not appear if you are not running Prism from a front end.

Copyright © 1991–1994 Thinking Machines Corporation

For complete information on attaching, detaching, cold booting, and turning safety on and off, see the *CM User's Guide*.

3.4.1 Attaching from within Prism

You can attach to a CM resource from within Prism. Attaching automatically cold boots the resource.

From the menu bar: Choose the **CMattach** selection from the **CM** menu. A dialog box is displayed; see Figure 10. By default, you are attached to the highest-numbered sequencer available on the first CM available, whether it is timeshared or exclusive.

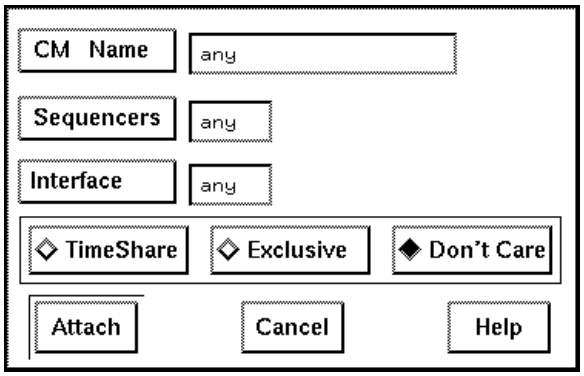

Figure 10. The CMattach dialog box.

In the **CM Name** text-entry box, specify the name of the CM to which you want to attach. The default is **any**, meaning that you will accept any available CM. Left-click on **CM Name** to display a menu of available CMs. You can either click on a choice from this menu or type the name of the CM in the box.

In the **Sequencers** box, specify the number(s) of the sequencer(s) to which you want to attach. The default is **any**, meaning that you will accept any available sequencer(s). Left-click on **Sequencers** to display a menu of legal sequencer sets. You can either click on a choice from this menu or type your choice in the box. If you choose **any** in both boxes, you are attached to the highest-numbered sequencer on the first CM that is available.

Version 2.0, April 1994
Copyright © 1991–1994 Thinking Machines Corporation

In the **Interface** box, specify the number of the front-end bus interface by which you want to attach to the CM. Once again, the default is **any**, meaning that you will accept any available interface. Left-click on **Interface** to display a menu of interfaces. You can either click on a choice from this menu or type your choice in the box.

Click on **TimeShare** if you want to attach to a timeshared CM resource. Click on **Exclusive** if you want to attach to a CM resource running in exclusive mode. In both cases, the diamond next to the word turns black, indicating your choice. **Don't Care** is the default choice; it means that you will take whatever resource is available, whether it is timeshared or exclusive. You can choose only one of the three.

When you have specified the information, click on **Attach** to attach to the CM resource (if it is available). You receive messages in the history region of the command window telling you if the attach was successful. You may also receive messages from CM System Software in the xterm from which you started Prism.

Click on **Cancel** or press the **Esc** key to close the dialog box without attempting to attach.

Note that this dialog box gives you less control over the characteristics of the CM resource than other methods. For example, you cannot specify the version of the microcode that the resource is to run. For maximum control over the characteristics of the CM resource, you must attach from within your program or by calling **`cmattach`** before you enter Prism.

From the command window: Issue the **`cmattach`** command to attach to a CM. By default you are attached to the highest-numbered sequencer available on the first CM available. The Prism version of **`cmattach`** accepts this subset of the shell-level **`cmattach`** options:

`-e`	Attach in exclusive mode only.
`-t`	Attach in timeshared mode only.
`-C` *name*	Attach to the specified CM.
`-i` *number*	Attach via the specified interface.
`-S` *seq-set*	Attach to the specified sequencer or sequencer set.

Copyright © 1991–1994 Thinking Machines Corporation

Auto-Attaching in Prism

As of Version 6.1 of CM System Software, a program can attach automatically to an available CM resource, run, and then detach. This works in Prism — with one complication. If an auto-attached program has a segmentation fault, it will remain attached to the CM; this allows you to examine the values of CM-resident data. To detach, you must issue this command from the command line to force the program to exit:

```
cont kill
```

(This also kills the program.) You cannot use the **CMdetach** selection described below. **CMdetach** detaches Prism from a CM; it cannot detach an auto-attached program.

3.4.2 Detaching from within Prism

You can detach from a CM resource from within Prism, whether or not you attached to it from within Prism.

From the menu bar: Choose the **CMdetach** selection from the **CM** menu. A dialog box is displayed, asking if you really want to detach. Click on **Detach** to proceed. Click on **Cancel** or press the **Esc** key to close the dialog box without detaching.

You receive a message in the history region of the command window when you are detached.

From the command window: Issue the **`cmdetach`** command to detach from the CM resource; in this case, you are not asked if you are sure you want to detach.

3.4.3 Cold Booting

To cold boot the CM resource to which you are attached, choose the **CMcoldboot** selection from the **CM** menu, or issue the **`cmcoldboot`** command from the command window. The Prism version of **`cmcoldboot`** accepts all the options supported by the shell-level **`cmcoldboot`**.

Version 2.0, April 1994
Copyright © 1991–1994 Thinking Machines Corporation

3.4.4 Turning Safety On and Off

To turn Paris safety on or off for the **CM** resource to which you are attached, choose the **CMsetsafety** selection from the **CM** menu. This is a toggle switch. Choosing the selection fills in the toggle box to the left of the menu selection; safety is now on. Choosing it again turns safety back off. From the command window, issue **`cmsetsafety on`** to turn safety on, and **`cmsetsafety off`** to turn it off.

Note that safety should be turned off when you are collecting performance data; otherwise your results will be inaccurate. If you attempt to run a program with collection and safety both on, Prism turns safety off and displays a message informing you of this. (It will then turn safety back on when you turn collection off.)

3.4.5 Obtaining Information about CM Users

Choose the **CMfinger** selection from the **CM** menu to display information about what CMs are available and who is using them.

From the command window, issue the **`cmfinger`** command to obtain the same output.

3.5 Executing a Program

To execute a program, you must first load it, as described in Section 3.1. If the program uses a CM-2 or CM-200**,** you must be attached; see Section 3.4. Once you start the program running, you can step through it, and interrupt and continue execution.

See Section 10.4 for information on executing a program in Node Prism.

Copyright © 1991–1994 Thinking Machines Corporation

3.5.1 Running a Program

To run a program:

- **From the menu bar:** If you have no command-line arguments you want to specify, choose the **Run** selection from the **Execute** menu; execution starts immediately. (The **Run** selection by default is in the tear-off region.)

  If you have command-line arguments, choose the **Run (args)** selection from the **Execute** menu. A dialog box is displayed, in which you can specify any command-line arguments for the program; see Figure 11. If you have more arguments than fit in the input box, they scroll to the left. Click on the **Run** button to start execution.

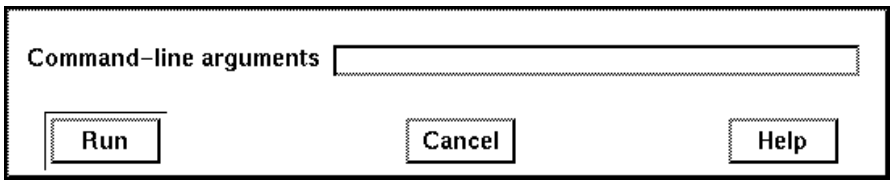

Figure 11. The Run (args) dialog box.

- **From the command window:** Issue the **`run`** command, including any arguments to the program on the command line. You can abbreviate the command to **`r`**. If you have already run the program, you can issue the **`rerun`** command to run it again, using the same argument list you previously passed to the program. In both cases, you can redirect input or output using **`<`** or **`>`** in the standard UNIX manner.

When the program starts executing, the status region displays the message **running**.

You can continue to interact with Prism while a program is running, but many features will be unavailable. Unavailable selections are grayed out in menus. If you issue a command that cannot be executed while the program is running, it is queued until the program stops.

Version 2.0, April 1994
Copyright © 1991–1994 Thinking Machines Corporation

3.5.2 Program I/O

Prism by default creates a new window for a program's I/O. This window persists across multiple executions and program loads, giving you a complete history of your program's input and output. If you prefer, you can display I/O in the Xterm from which you invoked Prism; see Section 9.3.

3.5.3 Stepping through a Program

You must begin execution by choosing **Run** or **Run (args)** (or issuing **`run`** from the command line). If execution stops before the program finishes (for example, because you have set a breakpoint), you can then step through the program, as described in this section. To step through the entire program, set a breakpoint at the first executable line, and then run to it. (See Section 4.3 for information on setting breakpoints.)

NOTE: If you compiled your CM Fortran program with the **`-cmprofile`** option instead of **`-g`**, a single step may execute several lines of code; this is caused by the way **`-cmprofile`** creates the symbol table information.

From the menu bar:

- Choose the **Step** selection from the **Execute** menu to execute the next line of the program. (It is by default in the tear-off region.) **Step** steps into any functions called on that line.
- Choose the **Next** selection from the **Execute** menu to execute the next statement of the program. (It is also by default in the tear-off region.) **Next** steps over any function called in the line, considering the function to be a single statement.
- Choose the **Stepout** selection from the **Execute** menu to execute the current function, then return to its caller.

The execution pointer moves to indicate the next line to be executed.

From the command window: Issue the **`step`**, **`next`**, or **`stepout`** command from the command line to perform the same action as the equivalent menu-bar selection; **`return`** is a synonym for **`stepout`**. In addition, you can specify the number of lines to be executed as an argument to **`step`** and **`next`**, and you can specify as an argument to **`stepout`** the number of levels of the call stack that you want to step out.

Copyright © 1991–1994 Thinking Machines Corporation

The **stepi** and **nexti** commands are also available for stepping by machine instruction. The address and instruction are displayed in the command window.

If execution takes considerable time — for example, because **Next** calls a long-running function — the status changes to **running**. You can use Prism, but many commands will be unavailable. Unavailable selections are grayed out in menus.

3.5.4 Interrupting and Continuing Execution

To interrupt execution, choose **Interrupt** from the **Execute** menu or type **Ctrl-c**. The status changes to **stopped,** and the source window updates to show the point at which execution stopped.

To continue execution after a program has been interrupted, choose **Continue** from the **Execute** menu, or issue the **cont** command from the command line. (Or you can step through the program, as described above.)

Continue and **Interrupt** are available by default in the tear-off region.

3.5.5 Status Messages

Prism displays the status messages listed in Table 1 before, during, and after the execution of a program.

Table 1. Status messages.

Message	Displayed when:
initial	Prism starts up without a program loaded.
loading	Prism is loading a program.
not started	A program is loaded but not yet started.
running	A program is running.
stopped	A program has stopped at a breakpoint or signal.
terminated	A program has run to completion and the process has gone away.

Version 2.0, April 1994
Copyright © 1991–1994 Thinking Machines Corporation

3.6 Choosing the Current File and Function

Prism uses the concepts of *current file* and *current function.*

The current file is the source file currently displayed in the source window. The current function is the function or procedure displayed in the source window. You might change the current file or function if, for example, you want to set a breakpoint in a file that is not currently displayed in the source window, and you don't know the line number at which to set the breakpoint.

In addition, changing the current file and current function changes the scope used by Prism for commands that refer to line numbers without specifying a file, as well as the scope used by Prism in identifying variables; see Section 2.9.1 for a discussion of how Prism identifies variables. The scope pointer (−) in the line-number region moves to the current file or current function to indicate the beginning of the new scope.

To change the current file:

- **From the menu bar:** Choose the **File** selection from the **File** menu. A window is displayed, listing in alphabetical order the source files that make up the loaded program. Click on one, and it appears in the **Selection** box; click on **OK**, and the source window updates to display the file. Or simply double-click, rapidly, on the source file. You can also edit the filename in the **Selection** box.

  NOTE: The **File** window displays only files compiled with the `-g` switch.

Copyright © 1991−1994 Thinking Machines Corporation

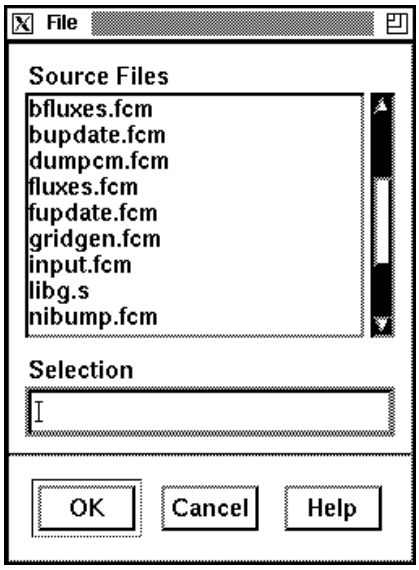

Figure 12. The File window.

- **From the command window:** Issue the **`file`** command, with the name of a file as its argument. The source window updates to display the file.

To change the current function or procedure:

- **From the menu bar:** Choose the **Func** selection from the **File** menu. A window is displayed, listing the functions in the program in alphabetical order. (CM Fortran procedure names are converted to all lowercase.) Click on one, and it appears in the **Selection** box; click on **OK**, and the source window updates to display the function. Or simply double-click on the function name in the list. You can also edit the function name in the **Selection** box.

  By default, the **Func** window displays only functions in files compiled with the **`-g`** switch. To display all functions in the program, click on the **Select All Functions** button. The button then changes to **Show `-g` Functions**; click on it to return to displaying only the **`-g`** functions.

- **From the command window:** Issue the **`func`** command with the name of a function or subroutine as its argument. The source window updates to display the function.

Version 2.0, April 1994
Copyright © 1991–1994 Thinking Machines Corporation

- **From the source window:** Select the name of the function in the source window by dragging the mouse over it while pressing the **Shift** key. When you let go of the mouse button, the source window is updated to display the definition of this function. **NOTE:** Do not include the arguments with the function, just its name.

Note that if the function you choose is in a different source file from the current file, changing to this function also has the effect of changing the current file.

3.7 Creating a Directory List for Source Files

If you have moved a source file, or if for some other reason Prism can't find it, you can explicitly add its directory to Prism's search path.

- **From the menu bar:** Choose the **Use** selection from the **File** menu. This displays a dialog box, as shown in Figure 13. To add a directory, type its pathname in the **Directory** box, then click on **Add**. To remove a directory, click on it in the directory list; its pathname appears in the **Directory** box; then click on **Remove.**

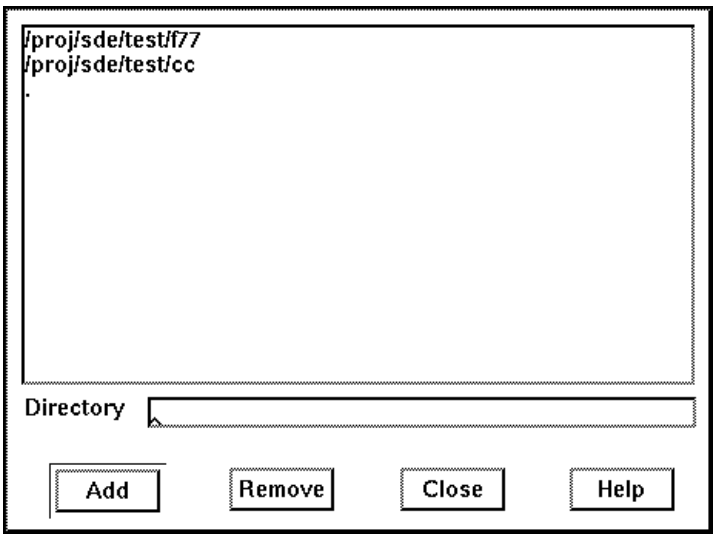

Figure 13. The Use dialog box.

Copyright © 1991–1994 Thinking Machines Corporation

- **From the command window:** Issue the **`use`** command on the command line. Specify a directory as an argument; the directory is added to the front of the search path. Issue **`use`** with no arguments to display the list of directories to be searched.

NOTE: No matter what the contents of your directory list is, Prism searches for the source file first in the directory in which the program was compiled.

Version 2.0, April 1994
Copyright © 1991–1994 Thinking Machines Corporation