CRAY RESEARCH, INC.

Cray TotalView Debugger Reference Card

SQ–2519 2.0

Before you start

Before you can use the Cray TotalView debugger, the `PrgEnv` module must be loaded. To load this module, enter:

```
module load modules PrgEnv
```

Compiling your program

Symbol tables are needed only for routines of interest.

C programs on Cray systems:

No optimization:	`cc -g` *program* `.c`
Partial optimization:	`cc -Gp` *program* `.c` (PVP only)
Full optimization:	`cc -Gf` *program* `.c`

C programs on SPARC systems:

No optimization:	`cc -g -xs` *program* `.c`

C++ programs on Cray systems:

No optimization:	`CC-g` *program* `.c`
Partial optimization:	`CC-Gp` *program* `.c` (PVP only)
Full optimization:	`CC-Gf` *program* `.c`

FORTRAN 77 programs on Cray systems:

No optimization:	`cf77 -g` *program* `.f` `cft77 -ez -o off` *program* `.f`
Partial optimization:	`cf77 -G` *program* `.f` (PVP only) `cft77 -ez` *program* `.f` (PVP only)
Full optimization:	`cft77 -ed` *program* `.f`
Autotasking:	`cf77 -Zp -G` *program* `.f`

FORTRAN 77 programs on SPARC systems:

No optimization:	`cf77 -g -xs` *program* `.f`

Fortran 90 programs on Cray and SPARC systems:

No optimization:	`f90 -g` *program* `.f90`
Partial optimization:	`f90 -G1` *program* `.f90`
Autotasking	`f90 -Gl -OtasK3` *program* `.f90`

`totalview` command line

`totalview [-L] [`*Xoptions*`] [-c` *corefile*`] [-X` *npes*`] [-V] [-p` *pid*`]` *filename*

Option	Description
`-L`	Invokes the line-mode interface to TotalView.
Xoptions	Specifies standard X Window System option values for TotalView to use.
`-c` *corefile*	Specifies name of a core file to be debugged.
`-X` *npes*	Specifies number of processor elements (PEs) in the partition.
`-V`	Prints TotalView version, date, and time.
`-p` *pid*	Specifies process ID of a live process that, upon invocation of TotalView, is stopped and control of which is transferred to TotalView.
filename	Specifies name of an executable file to be debugged. The name can be a quoted string containing a complete program invocation including arguments and standard I/O redirection.

For all command-line options, see the `totalview(1)` man page.

Command-line examples

To invoke TotalView for a program that aborts and produces a core file:

`totalview -c core` *myprog*

To invoke TotalView using a quoted string for the file name of a program:

`totalview "a.out -`*stuff* `<` *input*`"`

For more examples, see the `totalview(1)` man page.

TotalView user interface

TotalView uses one of two interfaces: an X Window System interface or a line-mode interface. If you set the `DISPLAY` environment variable before starting TotalView, the debugger will bring up the X Window System interface instead of the line-mode interface. If you specify the `-L` option on the `totalview` command line, TotalView invokes the line-mode interface even if the `DISPLAY` variable is set.

TotalView X Window System interface

To open a menu, press the `ALT` key and the underlined letter in the menu name. To select a menu option, press the `CONTROL` key and another key (shown as ^ *key*). You also can select a menu or option by placing the cursor over the item and pressing the left mouse button. Three dots (`...`) after a menu option indicate that selecting the option will open a dialog box. To request help, click on the `Help` menu button on any TotalView window.

Process window menu summary

The process window is the most frequently used window. The following list summarizes common menu options and gives corresponding key sequence.

`File menu options`	Key sequence
`TotalView Window`	^M
`Save Window`	
`Save Window As...`	
`Edit`	
`Close All Process Windows`	
`Close Current Window`	^F
`Quit`	^Q

`Control menu options`	Key sequence
`Run`	
`Run (args)...`	
`Interrupt`	^C
`Cont`	^G
`Cont to`	^R
`Step (source line)`	^S
`Stepi (instruction line)`	^I
`Next (source line)`	^N
`Nexti (instruction line)`	^X
`Return (run out of function)`	^O
`Set PC to Selection`	^P
`Set PC to Absolute Value`	
`Set Command Arguments`	
`Delete Process`	
`Reload Executable File`	

`Events menu options`	Key sequence
`Show Breakpoints...`	^B
`Suppress Breakpoints`	
`Load Breakpoints`	
`Save Breakpoints`	

Display menu options	Key sequence
Variable	^V
Address(s)...	
Registers	
Expression Evaluator...	

Source menu options	Key sequence
Function...	^L
File...	^K
Current Location (PC)	^W
Goto Line...	
Set Search Directory...	^D

Search menu option	Key sequence
Search...	^/

Pset menu option	Key sequence
Goto Process...	

Screen button functions

Button	Description
`Print...`	Prints value of a variable or expression. Output goes to the array browser window and shows a two-dimensional slice of a multidimensional array.

Mouse button functions

Button	Description
Left	Selects an item. Selecting a line number for an executable line of code sets a breakpoint at that line.
Right	Dives into an item. Diving is the action of displaying more information about an item including variables and breakpoints.

TotalV iew command-line interface

To obtain more information about the format of and examples of each command listed below, type `help` *commandname* at the `totalview` prompt when running TotalV iew in line mode.

Exiting TotalV iew

Command	Description
`quit`	Quits the debugger.

Breakpoints

Command	Description
`status`	Lists active breakpoints.
`delete` *<number >*	Deletes breakpoint. Use number from `status` command output.
`delete all`	Deletes all breakpoints.
`clear`	Clears breakpoint at current stopping point.
`clear` *<line>*	Clears breakpoint at source line.
`stop at` *<line>*	Stops execution at source line.
`stop in` *<proc>*	Stops execution when *<proc>* is called.

Controlling execution

Command	Description
`run`	Executes program from the beginning.
`cont`	Executes program from current execution address.
`cont to` *<line>*	Continues program execution to source line.
`step`	Executes program one source line at a time. Will step into calls.
`next`	Executes program one source line at a time. Will NOT step into calls.
`interrupt`	Stops execution of all processors.

Displaying and changing data

Command	Description
`print` *<exp>*	Prints the value of a variable or expression. Multiple items are separated by commas.
`assign var=`*exp*	Changes the value of a variable.
`whatis` *<name >*	Prints declaration of a variable.
`examine`	Displays memory contents.

Changing current function

Command	Description
`func`	Prints current function name.
`func` *<func>*	Changes current function.
`where`	Lists functions currently active in program.
`up`	Moves up one call.
`down`	Moves down one call.

Accessing source files

Command	Description
`search` *<string >*	Searches forward for string.
`bsearch` *<string >*	Searches backward for string.
`file`	Prints current source file name.
`file` *<file>*	Changes current source file.
`list`	Lists 10 source lines.
`list 1,0`	Lists entire source file.
`use` *<directory >*	Changes directory search path.

Debugging process sets

Command	Description
`pset`	Displays status of processors in Pset (Process set).
`pset` *<number >*	Changes current processor within Pset (Process set).

Status inquiries and debugger environment

Command	Description
`tvdbenv`	Displays debugger environment.
`tvdbinit`	Uses a `.tvdbinit` file to initialize the debugging session.
`info`	Shows information about the program being debugged.

`debugview` command line

The `debugview` command may be used in conjunction with the TotalV iew debugger. This command invokes the postmortem core analyzer that provides a program traceback and interprets the program memory dump in terms of source language symbols.

`debugview [-B] [-D] [-Y] [-V]` *filename corefile*

See the `debugview(1)` man page for complete descriptions of command-line options. See the command descriptions for the compiler you are using to determine the options required to produce a symbol file.