

NAME

pathdb – The PathScale Debugger

SYNOPSIS

pathdb [-help][-nx][-q][-cd=*dir*][-e *prog*][-c *core*][-x *cmds*][-d *dir*]

DESCRIPTION

The **pathdb** debugger enables you to "see into" a program as it executes, or investigate what happened at the moment a program crashed.

The **pathdb** program can be used with C, C++, and Fortran programs. The shell command **pathdb** invokes the program. Type **help** to view the command-line Help system. Type **quit** to exit the program.

The **pathdb** program can be run with no arguments or options. Generally you would start the **pathdb** program with an argument specifying the executable program you want to debug.

pathdb program

OPTIONS

The PathScale debugger supports a number of options. All of the **pathdb** options and command line arguments are processed in sequential order. The order of commands and options is important when the '-x' option is used.

-fullname

Print filename and line number information about current line. Primarily used internally for interfacing pathdb with emacs.

-nt

No terminal, do not execute with command-line editing. Option available for compatibility with GDB.

-nw

Option available for compatibility with GDB. Pathdb ignores this option.

-nx

Do not execute .pathdbrc file. Option available for compatibility with GDB.

-q, -quiet

Run in "quiet" mode; do not print out any introductory or copyright messages.

-subverbose

Produces diagnostic output about the subscription management for the debugger.

-v, -version

Print the **pathdb** version and exit.

-x file Execute **pathdb** commands from the file *file*.

COMPLETE COMMAND SET

advance Continue execution until the specified location is reached.

alias With no args, show all aliases. With one arg, show the named alias. With two args, set the named alias to the named value

attach Attach to a running process

backtrace

Show the stack backtrace for the specified number of levels. The default is for all levels.

break [*file:*]*function*

Set a breakpoint at a *function*, line number, or static address. A *file* can be specified.

call Call a function in the program being debugged. Record the value and print it if the type is not void.

catch Create a catchpoint breakpoint for the given event.

cd Change the current working directory to that specified. No arg means to change to the home directory.

- clear** Clear breakpoints at the specified location (line or address).
- commands** Specify a set of commands to be executed when the breakpoint is activated.
- condition** Set or clear the condition on the specified breakpoint.
- continue** Continue execution of the program being debugged (after stopping at a breakpoint).
- define** Define a new command as a sequence of existing commands.
- delete** Delete breakpoints, watchpoints, or displays.
- detach** Detach from a running process that is currently being controlled by the debugger.
- dir** Set the search directory for source file searches.
- disable** Disable the numbered breakpoints.
- disassemble** Disassemble the current function, or the addresses specified.
- display** Print the value of the expression every time the debugger stops the program being debugged.
- down** Move the current stack frame down the stack (to the called functions) by the specified number of levels.
- echo** Print some text to the screen.
- edit** [*file*] View the current program line.
- enable** Enable or set the disposition on a set of breakpoints.
- env** Show the values of all the environment variables.
- exec** Specify the file to be used for the executable image.
- file** Change the file being debugged.
- finish** Continue execution until the current function returns.
- frame** Select the specified frame number or show the current frame.
- handle** Specify what to do with signals when they are raised by the program being debugged.
- hbreak** Set a hardware-assisted breakpoint at the specified location.
- help** [*name*] Display information about the **pathdb** command *name* or display general help information for **pathdb**.
- history** Show all the commands typed by the user.
- if** Execute a sequence of commands if the expression evaluates to non-zero.
- ignore** Set the ignore-count for the specified breakpoint.
- info** Provide information on the named entity.
- kill** Kill the program being debugged.
- list** [*file*] List the source lines at the specified locations.
- memdump** Dump the address specified in hex and ASCII.
- next** Step the program being debugged by a number of lines, stepping over called functions
- nexti** Step the program being debugged by a number of instructions, stepping over call instructions.

- output** Print the value of the expression with no line feed and do not insert the value into a debugger variable.
- print** Print the value of the expression and put result in debugger variable.
- process** Switch to another process.
- processes**
List all the processes being debugged.
- ptype** Print the type of the expression.
- pwd** Print the current working directory.
- quit** Exit the **pathdb** program.
- rerun** Step backwards by a number of commands.
- return** Return from the current function with the value specified.
- run** [*arglist*]
Run your program using **pathdb** (with an *arglist* if included).
- watch** Create a watchpoint for a read of the specified location.
- set** Set the value of control parameters or program variables.
- set env** Set or print.
- step** Step the program being debugged by a number of lines, stepping into called functions. If no number is specified, then step one line.
- where** Display the program stack (function backtrace).
- x** Examine memory at the specified address, using the format specified.

COPYRIGHT

Copyright 2004, 2005 PathScale, Inc. All Rights Reserved.

SEE ALSO

See the *PathScale Debugger User Guide*.