GDB QUICK REFERENCE  GDB Version 4

Essential Commands

gdb program [core] debug program [using coredump core]
g command show command list
b [file:]function set breakpoint at function [in file]
run [arglist] start your program [with arglist]
bt print backtrace
p expr display the value of an expression
n next line
s single step

Starting GDB

gdb program begin debugging program
 gdb program core debug coredump core produced by program
gdb --help describe command line options

Stopping GDB

quit exit GDB; also q or EXIT (eg C-C) terminate current command, or send to running process

Getting Help

help list classes of commands
help class one-line descriptions for commands in class
help command describe command

Executing Your Program

run arglist start your program with arglist
run run run program with current argument list
run ...<inf>outf/ start your program with input, output redirected
kill kill running Program
tty dev use dev as stdin and stdout for next run
set arglist specify arglist for next run
set args specify empty argument list
show args display argument list
show env show all environment variables
show env var show value of environment variable
set env variable set environment variable
unset env var remove var from environment

Shell Commands

cd dir change working directory to dir
ps print working directory
make ... call "make"
sHELL cmd execute arbitrary shell command string

Breakpoints and Watchpoints

break [file:]line set breakpoint at line number [in file]
b [file:]line set breakpoint at file [in file]
break [file:]func set breakpoint at func [in file]
break -offset set break at offset lines from current stop
break -addr set breakpoint at address addr
break break ... if expr break conditionally on nonzero expr
cond n [expr] new conditional on breakpoint n; make unconditional if no expr
break ... temporary break; disable when reached
break regex break on all functions matching regex
watch expr set a watch for expression expr
catch r break at C++ handler for exception r
info break show defined breakpoints
info watch show defined watches
break delete breakpoints at next instruction
clear delete breakpoints at entry to fun()
clear delete breakpoints on source line
delete [n] delete breakpoints [or breakpoint n]
disable [n] disable breakpoints [or breakpoint n]
enable [n] enable breakpoints [or breakpoint n]
enable once [n] enable breakpoints [or breakpoint n]; disable again when reached
enable del [n] enable breakpoints [or breakpoint n]; delete when reached
ignore n count ignore breakpoint n, count times
commands n [silent] command-list every time breakpoint n is reached. silent suppresses default display
end command-list

Program Stack

backtrace [n] print trace of all frames in stack; or of n frames—inmost first if n<0, outermost if n>0
bt [n] select frame number n or frame at address n; if no frame, display current frame
up n select frame n frames up
down n select frame n frames down
info frame [addr] describe selected frame, or frame at addr arguments of selected frame
info frame local variables of selected frame
info reg [rn]... register values [or regs rn] in selected frame
info all-reg [rn] all-reg includes floating point frame
info catch exception handlers active in selected frame

Execution Control

continue [count] continue running; if count specified, ignore this breakpoint next count times
step [count] execute until another line reached; repeat count times if specified
stepi [count] step by machine instructions rather than source lines
next [count] execute next line, including any function calls
nexti [count] next machine instruction rather than source line

until [location] run until next instruction [or location]
finish run until selected stack frame returns
return [expr] pop selected stack frame without executing (return value)
signal num resume execution with signal (name if 0)
jump expr resume execution at specified line number or address
jump <expr> evaluate expr without displaying it; use for altering program variables

Display

print [ff] [expr] show value of expr [or last value] $ according to format f
p [ff] [expr] show value of expr [or last value] $ according to format f
x [ff] [expr] show hexadecimal value of expr [or last value] $
d signed decimal
u unsigned decimal
o octal
b binary
a address, absolute and relative
f floating point
call [ff] [expr] like print but does not display void
x [Ndx] [expr] examine memory at address expr; optional format spec follows %
N count of how many units to display
w unit size; one of
b individual bytes
h halfwords (two bytes)
w words (four bytes)
g giant words (eight bytes)
printing format. Any printf format, or
s null-terminated string
i machine instructions
dissasm [addr] display memory as machine instructions

Automatic Display

display [ff] [expr] show value of expr [each time program stops] according to format f
display display all enabled expressions on list
undisplay n remove number(s) n from list of automatically displayed expressions
disable disp n disable display for expression(s) number n
enable disp n enable display for expression(s) number n
info display numbered list of display expressions
### Expressions

- `expr`: an expression in C, C++, or Modula-2 (including function calls), or;
- `add`<br>`@`: `an array of `n` elements beginning at address `addr`;
- `type`: `var` or `function` `nm` defined in file `file`;
- `addr`: `read memory at addr as specified type`;
- `n`: `n`th displayed value;
- `value`: `n`th displayed value previous to `value` at address `addr`;
- `conv`: `display all convenience variables`;
- `show`: `show last 10 values or surrounding $n$`;
- `symbol`: `show where symbol` s is stored;
- `func`: `show names, types of defined functions (all, or matching regex)`;
- `var`: `show names, types of global variables (all, or matching regex)`;
- `what`: `show data type of expr or $` without evaluating `p`;
- `type`: `describe type, struct, union, or enum`;

### GDB Scripts

- `source`: `read, execute GDB commands from file`;
- `define`: `create new GDB command cmd`;
- `command`: `execute script defined by command-list`;
- `document`: `create online documentation for new GDB command cmd`;

### Signals

- `handle`: `specify GDB actions for signal`;
- `print`: `announce signal`;
- `no-print`: `be silent for signal`;
- `stop`: `halt execution on signal`;
- `nostep`: `do not halt execution`;
- `pass`: `allow your program to see signals`;
- `noskip`: `do not skip signals`;
- `info`: `show table of signals, GDB action for each`;

### Debugging Targets

- `target`: `connect to target machine, process, or file`;
- `help target`: `display available targets`;
- `attach`: `connect to another process`;
- `detach`: `release target from GDB control`;

### Controlling GDB

- `set`: `set one of GDB's internal parameters`;
- `param`: `display current setting of parameter`;
- `show`: `show current source path`;
- `list`: `list previous ten lines`;
- `list-lines`: `display source surrounding lines, specified as`;
- `file`: `show where symbol` s is stored;
- `func`: `show names, types of defined functions (all, or matching regex)`;
- `var`: `show names, types of global variables (all, or matching regex)`;
- `what`: `show data type of expr or $` without evaluating `p`;
- `type`: `describe type, struct, union, or enum`;

### Source Files

- `dir`: `add directory names to front of source path`;
- `dir`: `clear source path`;
- `list`: `show next ten lines of source`;
- `list`: `show previous ten lines`;
- `list-lines`: `display source surrounding lines, specified as`;
- `file`: `show where symbol` s is stored;
- `func`: `show names, types of defined functions (all, or matching regex)`;
- `var`: `show names, types of global variables (all, or matching regex)`;
- `what`: `show data type of expr or $` without evaluating `p`;
- `type`: `describe type, struct, union, or enum`;

### GDB under GNU Emacs

- `M-x gdb`: `run GDB under Emacs`;
- `C-h m`: `describe GDB mode`;
- `M-x`: `step one line (step)`;
- `M-x`: `step one instruction (stepi)`;
- `C-c C-f`: `finish current stack frame (finish)`;
- `C-c C-c`: `continue (cont)`;
- `M-a`: `up arg frames (up)`;
- `M-d`: `down arg frames (down)`;
- `C-x &`: `copy number from point, insert at end`;
- `C-x 0`: `in source file`;

### GDB License

- `copyright`: `Display GNU General Public License`;
- `show copying`: `There is NO WARRANTY for GDB`;
- `show warranty`: `Display full no-warranty statement`;

---

Roland Pesch, pesch@cygnus.com

The author assumes no responsibility for any errors in this card.

This card may be freely distributed under the terms of the GNU General Public License.
Please contribute to the development of this card by annotating it.

GDB itself is free software, you are welcome to distribute copies of it under the terms of the GNU General Public License. There is absolutely no warranty for GDB.