GDB QUICK REFERENCE

Essential Commands

gdb program [core] debug program [using core dump core]
b [file:function] set breakpoint at function in file
run [arglist] start your program with arglist
bt backtrace: display program stack
p expr display the value of an expression
c continue running your program
s next line, stepping over function calls
n next line, stepping into function calls

Starting GDB

gdb start GDB, with no debugging files
gdb program begin debugging program
gdb program core produce core by program
gdb --help describe command line options

Stopping GDB

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

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 your 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 args arglist specify empty argument list
set args display argument list
show environment show all environment variables
show env var show value of environment variable var
set env var string set environment variable var
unset env var remove var from environment

Shell Commands

cd or dr change working directory to dr
pwd 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
eg break main.c:37
break [file:function] set breakpoint at function in file
break +offset set break at offset lines from current stop
break -offset
break *addr set breakpoint at address addr
break set breakpoint at next instruction
break ... if expr break conditionally on nonzero expr
cond n [expr] new conditional expression on breakpoint
ignore this breakpoint in next count
break ... temporary break disable when reached
rbreak regx set a watchpoint for expression expr
watch expr set a watchpoint on function expr
catch x break at C++ handler for exception x
info break show defined breakpoints
info watch show defined watchpoints
clear delete breakpoints at next instruction
clear [file:fun] delete breakpoints at entry to fun()
clear [file:line] 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 again when reached
able del n delete when reached
ignore n count ignore breakpoint n, count times
commands n [silent] execute GDB command list every time breakpoint n is reached. [silent]
c [command-list] suppress 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 n=0, display current frame
frame [n] select frame n frames up
select frame n frames down
up n
frame [n] select frame n frames down
down n info frame [addr] describe selected frame, or frame at addr
info frame [addr] arguments of selected frame
info locals local variables of selected frame
info reg [rn] register values [for regs rn] in selected frame
info all-reg [rn] all-reg includes floating point
info catch exception handlers active in selected frame

Execution Control

continue [count] continue running; if count specified, ignore this breakpoint next count times

c [count] execute until another line reached; repeat count times if specified
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
ni [count] next machine instruction rather than source line
			
until [location] run until next instruction (or location)
finish return [expr] resume execution with signal s none if 0
jump line resume execution at specified line number or address
jump *address evaluate expr without displaying it; use for altering program variables

Display

print [f] [expr] show value of expr [or last value] according to format f
p [f] [expr] show value of expr [or last value] according to format f
x hexdecimal
d signed decimal
u unsigned decimal
q octal
b binary
a address, absolute and relative
z character
f floating point

Shell Commands

call [f] [expr] like print but does not display void
x [NU] [expr] examine memory at address expr;
N count of how many units to display unit size; one of
b individual bytes
h halfwords (two bytes)
word words (four bytes)
q qwords (eight bytes)
pr printing format, or
s null-terminated string
i machine instructions

Shell Commands

display [f] [expr] show value of expr each time program steps [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

addr@len
an array of len elements beginning at addr

file nm
a variable or function nm defined in file

{type} addr
read memory at addr as specified type

$ n
most recent displayed value

$n
nth displayed value

$ n
nth displayed value from previous to $ n

$L
last address examined with x

$L
value at address L

$var
convenience variable; assign any value

show values [ n]
show last 10 values or surrounding $n

show convenience
display all convenience variables

Symbol Table

info address
show where symbol is stored

info func [ reg]
show names, types of defined functions (all, or matching reg)

info var [ reg]
show names, types of global variables (all, or matching reg)

what is [ expr]
show data type of expr or $ without evaluating; p type gives more detail

p type [ expr]
describe type, struct, union, or enum

GDB Scripts

source script
read, execute GDB commands from file

define cmd [command-list]
create new GDB command cmd; execute script defined by command-list

end [command-list]
end of command-list

document cmd [help-text]
create online documentation for new GDB command cmd

end [help-text]
end of help-text

Signals

handle signal act
specify GDB actions for signal

print
announce signal

noprint
be silent for signal

stop
halt execution on signal

nostop
do not halt execution

pass
allow your program to see signal

noppass
do not allow your program to see signal

info signals
show table of signals, GDB action for each

Debugging Targets

target type param
connect to target machine, process, or file

help target
display available targets

attach param
connect to another process

release target from GDB control

GDB under GNU Emacs

M-x gdb
run GDB under Emacs

C+h m
describe GDB mode

M+s
step one line (step)

M+i
next line (next)

M-j
step one instruction (stepi)

C-c C-f
finish current stack frame (finish)

C-c C-
continue (cont)

M-m
up arg frames (up)

M-d
down arg frames (down)

C-x &
copy number from point, insert at end

C-x FPC
(in source file) set break at point

GDB License

show copying
Display GNU General Public License

show warranty
There is NO WARRANTY for GDB.

Display full no-warranty statement.

Source Files

dir names
add directory names to front of source path

dir
clear source path

show dir
show current source path

list
show next ten lines of source

list -
show previous ten lines

list lines
display source surrounding lines, specified as:

[file] num
line number in named file

[file] function
beginning of function in named file
	off
off lines after last printed

-off
off lines previous to last printed

address
line containing address

list [1]
from line 1 to line l

info line num
show starting, ending addresses of compiled code for source line num

show current source file
list all source files in use

search following source lines for reg

search preceding source lines for reg

Working Files

file [file]
use file for both symbols and executable
with no arg, discard both

core [file]
read core as coredump or discard

exec [file]
use file as executable only, or discard

symbol [file]
use symbol table from file or discard

load file
dynamically link file and add its symbols

add-sym file addr
read additional symbols from file, dynamically loaded at addr

info files
display working files and targets in use

path dirs
add dirs to front of path searched for executable and symbol files

show path
list names of shared libraries currently loaded

Roland Pesch (pesch@cygnus.com), January 1992—Revision: 1.99
The author assumes no responsibility for any errors on this card.

This card may be freely distributed under the terms of the GNU
General Public License.

Please contribute to 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.