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]
bpt backtrace: display program stack
prexpr display the value of an expression
c continue running your program
n next line, stepping over function calls
s next line, stepping into function calls

Starting GDB

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

Stopping GDB

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

Getting Help

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

Executing your Program

run arglist start your program with arglist
run start 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 arglist arguments 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 var
set env var string set environment variable var
unset env var remove var from environment

Shell Commands

cd dir change working directory to dir
ped 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 eg: break main.c:37
break [file:]func set breakpoint at func [in file]
break +offset set at offset lines from current stop
break -offset break at offset lines from previous stop
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 n; make unconditional if no expr
tbreak temporary break; disable when reached
rbreak regex break on all functions matching regex
watch expr set a watchpoint for expression expr
catch x break at C++ handler for exception x
info break show defined breakpoints
info watch show defined breakpoints
clear delete breakpoints at next instruction
clear [file:]func delete breakpoints at entry to func()
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 breakpoints [or breakpoint n]; disable again when reached
enable del [n] enable breakpoints [or breakpoint n]; disable when reached
global n count ignore breakpoint n, count times
commands n count execute GDB commands list every time breakpoint n is reached
command-list command-list suppresses default display
end end of command-list

Program Stack

backtrace [n] print trace of all frames in stack; or of n frames—in reverse if n>0, forward if n<0
frame [n] select frame number n or frame at address n; if n=0, 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
info args arguments of selected frame
info locals local variables of selected frame
info reg [rm] register values for registers rm in selected frame
info all-reg [rm] frame; 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
n [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 pop selected stack frame without executing setting return value
signal num resume execution with signal s (none if 0)
jump line resume execution at specified line number
jump address evaluate expr without displaying it; use for altering program variables

Display

print [/i] [expr] show value of expr or last value $; according to format f
f hexadecimal
x hex
s signed decimal
u unsigned decimal
o octal
t binary
a address, absolute and relative
character
c character
f floating point
call [/i] expr like print but does not display void
x [/i] expr examine memory at address expr; optional format spec follows sh
N count of how many units to display unit size, one of
b individual bytes
h halfwords (two bytes)
w words (four bytes)
g giant words (eight bytes)
/ printing format. Any print format, or
i machine instructions

Automatic Display

display [/i] expr show value of expr each time program stops according to format f
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