

Summary of GDB commands for x86-64 Systems

| Command | Effect |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------|
| Starting: | |
| gdb | |
| gdb <file> | |
| Running and stopping | |
| quit | Exit gdb |
| run | Run program |
| run 1 2 3 | Run program with command-line arguments 1 2 3 |
| kill | Stop the program |
| quit | Exit gdb |
| Ctrl-d | Exit gdb |
| Note: Ctrl-C does not exit from gdb, but halts the current gdb command | |
| Breakpoints | |
| break sum | Set breakpoint at the entry to function sum |
| break *0x80483c3 | Set breakpoint at address 0x80483c3 |
| delete 1 | Delete breakpoint 1 |
| disable 1 | Disable the breakpoint 1 (gdb numbers each breakpoint you create) |
| enable 1 | Enable breakpoint 1 |
| delete | Delete all breakpoints |
| clear sum | Clear any breakpoints at the entry to function sum |
| Execution | |
| stepi | Execute one instruction |
| stepi 4 | Execute four instructions |
| nexti | Like stepi, but proceed through function calls without stopping |
| step | Execute one C statement |
| continue | Resume execution until the next breakpoint |
| until 3 | Continue executing until program hits breakpoint 3 |
| finish | Resume execution until current function returns |
| call sum(1, 2) | Call sum(1,2) and print return value |
| Examining code | |
| disas | Disassemble current function |
| disas sum | Disassemble function sum |
| disas 0x80483b7 | Disassemble function around 0x80483b7 |
| disas 0x80483b7 0x80483c7 | Disassemble code within specified address range |
| print /x \$rip | Print program counter in hex |
| print /d \$rip | Print program counter in decimal |
| print /t \$rip | Print program counter in binary |
| Examining data | |
| print /d \$rax | Print contents of %rax in decimal |
| print /x \$rax | Print contents of %rax in hex |
| print /t \$rax | Print contents of %rax in binary |
| print /d (int)\$rax | Print contents of %rax in decimal after sign-extending lower 32-bits. |
| You need this to print 32-bit, negative numbers stored in the lower 32 bits of %rax. For example, if the lower 32-bits of %rax store 0xffffffff, you will see | |

```

(gdb) print $rax
$1 = 4294967295
(gdb) print (int)$rax
$2 = -1
(gdb)

```

```

print 0x100          Print decimal representation of 0x100
print /x 555         Print hex representation of 555
print /x ($rsp+8)    Print (contents of %rsp) + 8 in hex
print *(int *) 0xbffff890 Print integer at address 0xbffff890
print *(int *) ($rsp+8) Print integer at address %rsp + 8
print (char *) 0xbffff890 Examine a string stored at 0xbffff890

x/w  0xbffff890      Examine (4-byte) word starting at address
                        0xbffff890
x/w  $rsp             Examine (4-byte) word starting at address in $rsp
x/wd $rsp            Examine (4-byte) word starting at address in $rsp.
                        Print in decimal
x/2w $rsp            Examine two (4-byte) words starting at address
                        in $rsp
x/2wd $rsp           Examine two (4-byte) words starting at address
                        in $rsp. Print in decimal
x/g  $rsp            Examine (8-byte) word starting at address in $rsp.
x/gd $rsp           Examine (8-byte) word starting at address in $rsp.
                        Print in decimal
x/a  $rsp            Examine address in $rsp. Print as offset from
                        previous global symbol.
x/s  0xbffff890      Examine a string stored at 0xbffff890
x/20b sum            Examine first 20 opcode bytes of function sum
x/10i sum            Examine first 10 instructions of function sum

```

(Note: the format string for the 'x' command has the general form
 x/[NUM][SIZE][FORMAT] where

NUM = number of objects to display
 SIZE = size of each object (b=byte, h=half-word, w=word,
 g=giant (quad-word))
 FORMAT = how to display each object (d=decimal, x=hex, o=octal, etc.)

If you don't specify SIZE or FORMAT, either a default value, or the last
 value you specified in a previous 'print' or 'x' command is used.

)

Useful information

```

backtrace          Print the current address and stack backtrace
where              Print the current address and stack backtrace

info program       Print current status of the program)
info functions     Print functions in program
info stack         Print backtrace of the stack)
info frame        Print information about the current stack frame
info registers     Print registers and their contents
info breakpoints  Print status of user-settable breakpoints

display /FMT EXPR Print expression EXPR using format FMT
                  every time GDB stops
undisplay          Turn off display mode
help              Get information about gdb

```