[**使用 gdb 对程序进行汇编级调试**](http://www.cnblogs.com/diylab/archive/2009/07/16/1524483.html)

开始/重新开始  
  ( gdb) r     ( run )  
    
  下断点  
  (gdb) b \*0x0804ce2b  
   b 表示  break  
    
  单步步过  
  (gdb) ni    (next instruction)  
  单步步入    
  (gdb) si    ( step instruction )  
  继续执行  
  ( gdb )c  
    
  执行到返回  
  (gdb) finish  
    
  disas   
    
  反汇编一段指令。可以带零个、一个或两个参数。第一个参数是反汇编开始地址，第二个参数是反汇编结束地址。  
    
  如果没有参数，则反汇编当前的函数。  
    
    
  内存读/写断点  
  watch \*(int  \*)0x8049aa4  
  在 0x8049aa4 处下写断点，断点的范围为 4个字节。  
  gdb支持更大范围的内存写断点。但在汇编条件下，我还没有找到设置更大内存写断点的方法。  
  rwatch ，awatch 用法同 watch ，分别表示读断点和 读写断点。  
  读写断点依赖于 gcc可用的硬件高度寄存器。  
    
    
    
  寄存器窗口  
    
  (gdb) display  
    
  设定要观察的变量的内容。这些变量的值在程序每次被断下来后显示在屏幕上。  
    
  例如，我们经常要关注  eax, ebx, ecx,edx的内容，则用 display 设置他们为观察变量。  
    
  (gdb) display  /x $eax  
    
  (gdb) display  /x $ebx  
    
  (gdb) display  /x $ecx  
    
  (gdb) display  /x $edx  
    
    
  (gdb) until   
相当于 od  的 f4  
    
    
  特色功能：  
  1、设置反汇编代码使用的指令集  
  (gdb) set disas intel  
    
  设置反汇编代码使用的指令集，可选择 intel 指令集或 AT&T指令集.  
    
  该指令只能用于x86平台。  
    
    
  捕获 “段错误”的信号  
  (gdb) handle SIGSEGV  
  
  抛出异常时捕获  
  (gdb) catch throw  
    
  查看栈帧。  
  (gdb) where  
     
  强制返回  
  (gdb) return   
  程序直接从当前行跳转到 return 处。如果函数有返回值，则加在 return 命令之后。例如，　return 1。跳转中，栈平衡是自动维护的。修改程序代码段：

By default, GDB opens the le containing your program's executable code (or the corele) read-only. This prevents accidental alterations to machine code; but it also prevents you from intentionally patching your program's binary. If you'd like to be able to patch the binary, you can specify that explicitly with the set write command. For example, you might want to turn on internal debugging ags, or even  
to make emergency repairs.

set write on

exec-file

The dump and append commands write data to a file, and the restore command reads data from a file back into the inferior’s memory.

写内存：

To store values into arbitrary places in memory, use the `{...}' construct to generate a value of specied type at a specied address (see Section 8.1 [Expressions], page 63). For example, {int}0x83040 refers to memory location 0x83040 as an integer (which implies a certain size and representation in memory), and set {int}0x83040 = 4  
stores the value 4 into that memory

参考资料：

 <debugging with gdb> by Richard Stallman, Roland Pesch etc, Published by the Free Software Foundation