# 操作系统调试教程

在编写程序的过程中，我们希望能够对操作系统进行调试，因此本教程将介绍使用如何bochs进行操作系统调试。教程主要分为两部分，一部分是关于使用bochs自带的调试器进行调试，另一部分是关于使用bochs+gdb+CLion进行源码级调试，这两部分分别对应了我们编译的bochs-disasm和bochs-gdb。

## 使用bochs自带的调试器进行调试

使用bochs自带的调试器进行调试优点在于，调试期间可以直接查看寄存器和内存的内容，对于汇编代码的调试非常有效，而其也有一个很大的缺点，就是许多操作没有提供图形化界面（如按钮等），需要用户手动输入命令来实现。下面先介绍一下如何使用bochs自带的调试器进行调试。

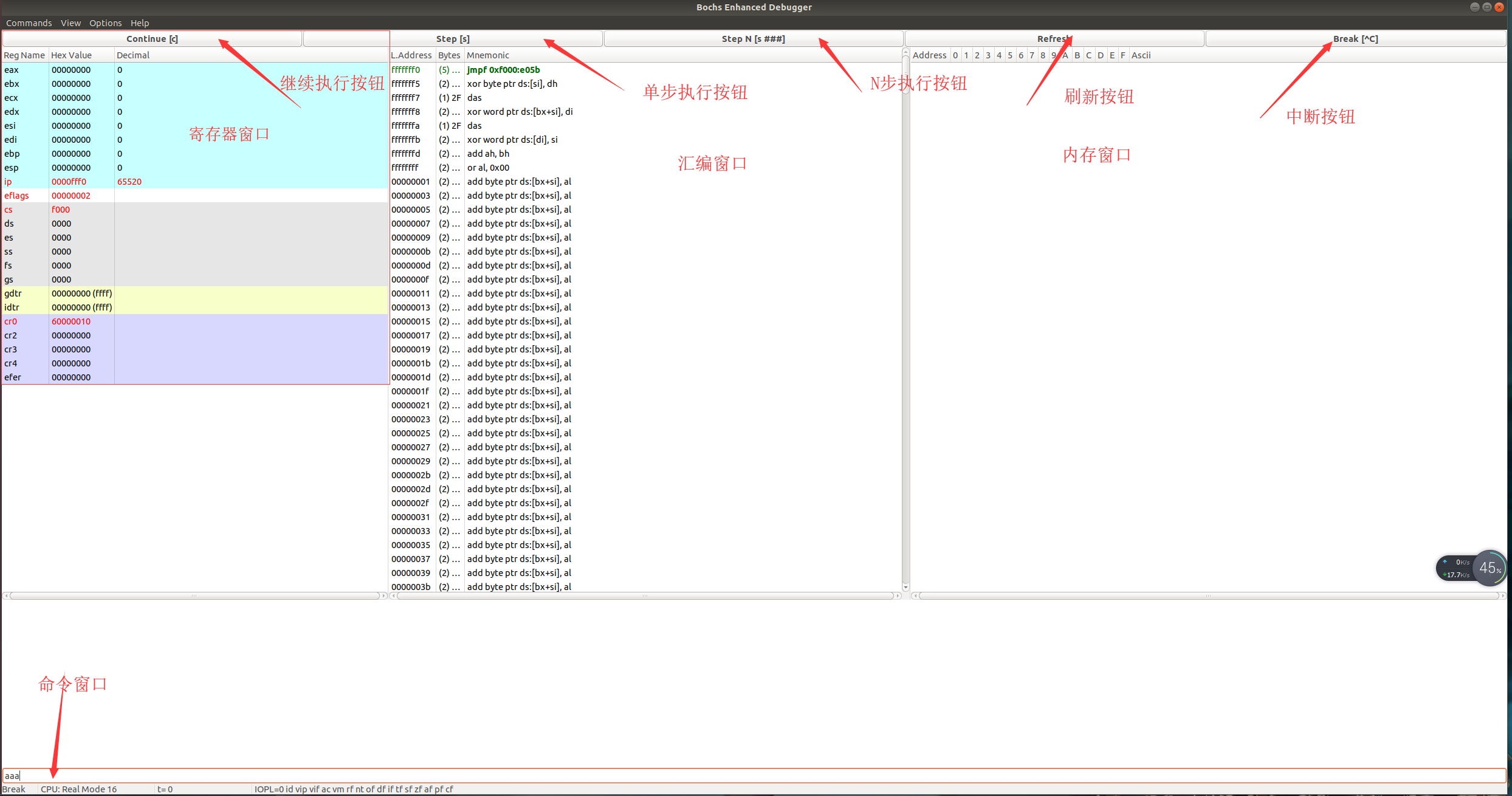
1. 修改bochsrc 文件

如果要使用bochs自带的调试器调试操作系统，那么需要在bochsrc文件中加入以下一行：

display\_library:x, options="gui\_debug"

1. 运行bochs-disasm

在终端中输入“bochs-disasm -f bochsrc”，会出现两个窗口，分别对应调试器窗口和bochs虚拟机的窗口。调试器窗口如下所示:



调试器窗口

调试器界面大致分为四部分，分别为寄存器窗口、汇编窗口、内存窗口以及命令窗口。寄存器窗口显示了寄存器的内容，汇编窗口显示了运行的指令，内存窗口用于显示某些内存的内容，命令窗口则用于运行指令。

在调试器的上方有五个按钮，分别为*Continue、Step、Step N、Refresh、Break*。其中Continue按钮的作用是，使程序继续运行；*Step*的作用是，是程序执行一条指令后停止；*Step N*作用是，使程序运行N条指令停止；*Break*的作用是，使当前运行的程序停止运行。

1. 调试

这里介绍一些常用的操作：

1. 给程序下断点

下断点的方法有两种，一种是在反汇编窗口中选中要下断点的指令，然后点击*“Command/Breakpoint”*即可。；另一种办法是在命令窗口输入”*vb \*<address>*”，其中<address>为要下断点的指令的地址，例如输入“vb \*0x0000:0x7c00”会给*0x7c00*处的指令下一个断点。

1. 查看某个地址的内容

点击“*view/physical Memdump*“，然后输入要查看的内存的物理地址就可以查看相应物理地址处的内容。而使用”view/Linear Memdump“则可以查看线性地址对应的内存处的地址。

1. 使用bochs提供的命令

在调试器的命令窗口中，我们可以通过输入命令并回车来完成操作，比如，输入“vb \* 0x0000:0x1234”可以给0x1234处的地址下断点。如果要使用调试器提供的指令请参考bochs的官方文档<http://bochs.sourceforge.net/cgi-bin/topper.pl?name=New+Bochs+Documentation&url=http://bochs.sourceforge.net/doc/docbook/user/index.html>。

## Bochs+gdb+CLion进行源码级别调试

使用Bochs+gdb+CLion进行调试的好处是，对于C语言编写的内核可以进行源码级别的调试，缺点是要求内核必须是elf文件，同时不能调试引导扇区。调试方法如下：

1. 修改bochsrc 文件

如果要使用此方式调试操作系统，那么需要在bochsrc文件中加入以下一行：

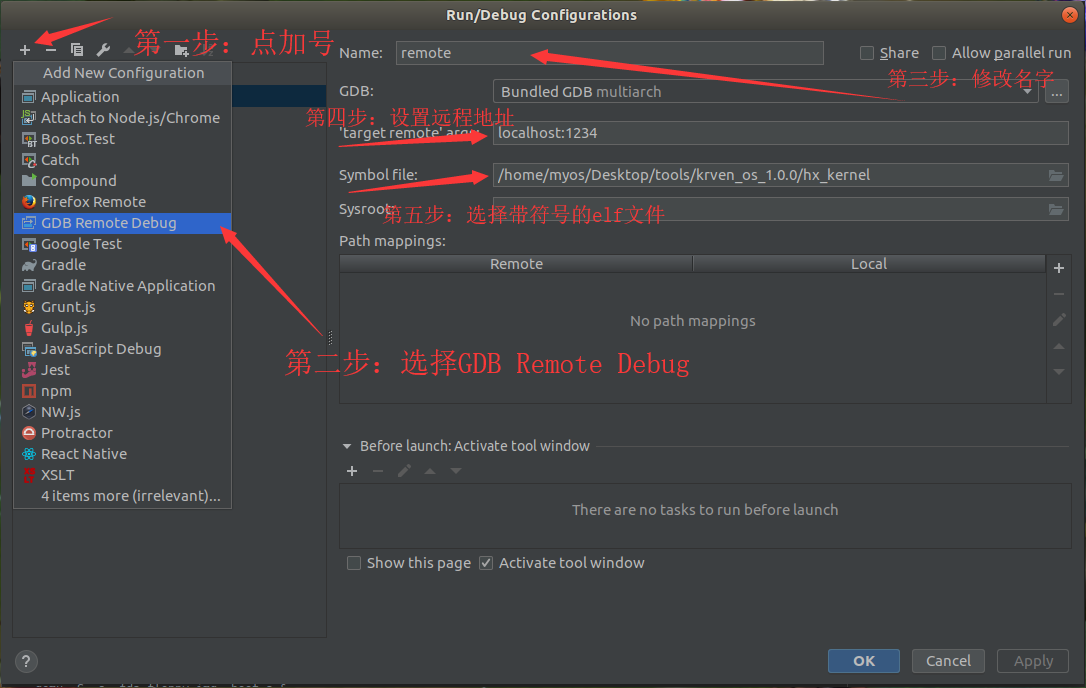
gdbstub: enabled=1, port=1234, text\_base=0, data\_base=0, bss\_base=0

1. 运行bochs-gdb

在终端中输入““bochs-gdb -f bochsrc”，会出现一个窗口，同时终端中会显示“Wait for gdb connection on port 1234“。

1. 为Clion配置远程调试器

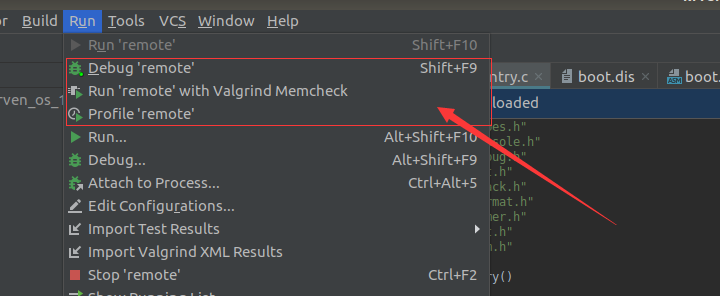
在CLion中，打开“run/Edit Configurations”会出现以下界面。



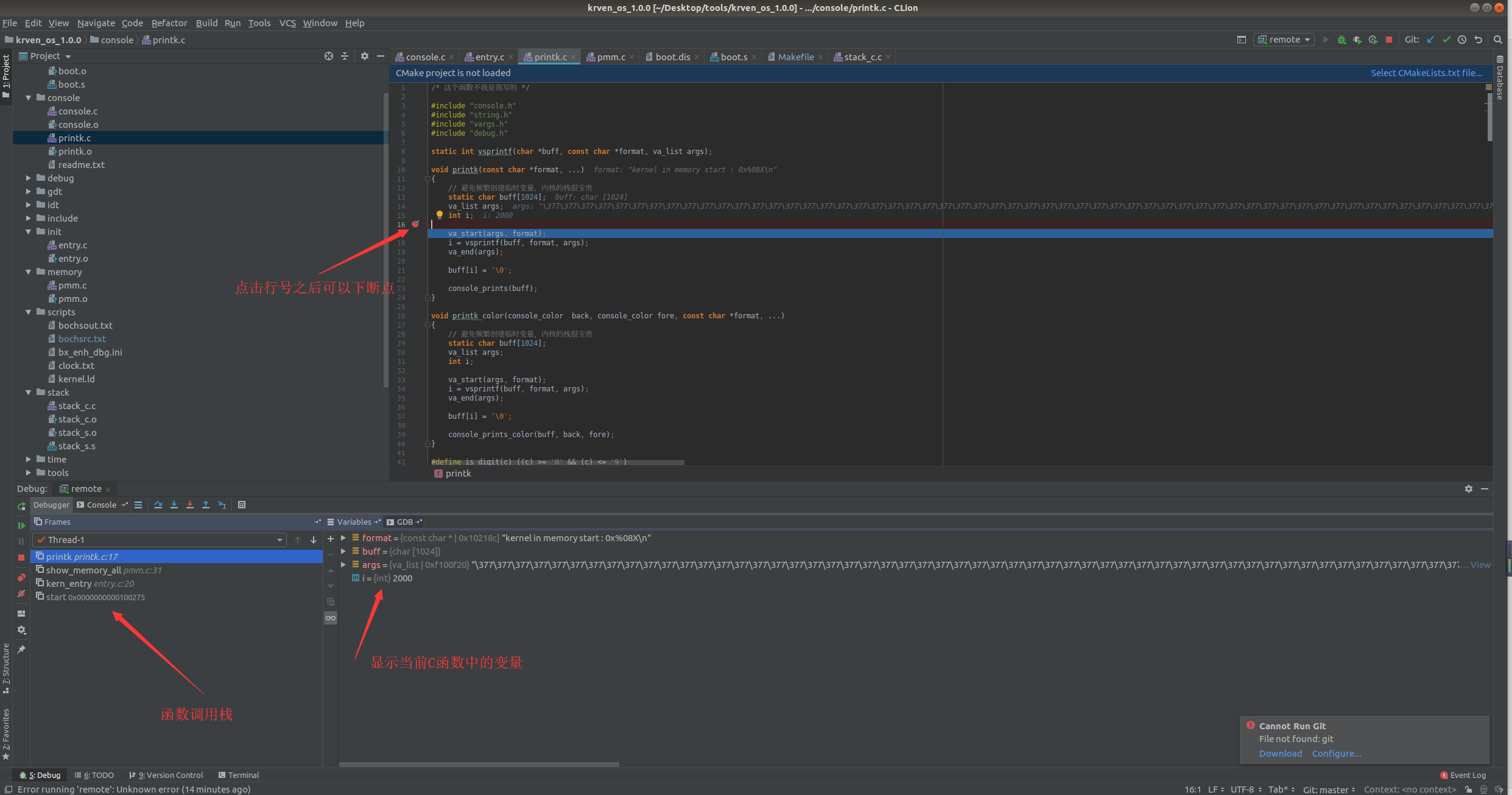
按照图中的步骤进行配置即可。第五步中带符号的elf文件是通过在编译时使用“-g”选项编译得到的（使用gcc、ld和nasm时都要添加）。

1. 打开远程调试

配置完毕之后，“run”中会出现几个新的选项，如下图所示：



在bochs-gdb开启的情形下，选择其中的“Debug ‘remote’”即可开始调试。调试界面如下图所示：



我们可以通过“run”下的一些选项来进行调试，如下图所示。其中，“Step Over”的作用是使当前程序向下执行一条语句，如果遇到函数调用则执行函数调用；“Step Into”也是使当前程序乡下执行一条语句，但是遇到函数调用时会进入函数调用；“Run to Cursor”的作用是使程序运行到我们编辑界面的光标处；”View Breakpoints”的作用是查看下过的断点。

