【Linux学习系列十三：使用gdb和gdbserver构建在线调试环境】

2019-10-04

版 本 历 史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本** | **作者** | **参与者** | **日期** | **备注** |
| V1.0 | Topsemic |  | 2019/10/04 | 创建 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[1.**引言** 4](#_Toc21079386)

[2.**环境介绍** 4](#_Toc21079387)

[2.1.**硬件** 4](#_Toc21079388)

[2.2.**软件** 4](#_Toc21079389)

[3.**Buildroot配置** 5](#_Toc21079390)

[4.**新建测试程序** 7](#_Toc21079391)

[5.**在线调试** 8](#_Toc21079392)

[6.**结束语** 11](#_Toc21079393)

# 1.**引言**

单片机一般使用Jlink通过SWD或者JTAG接口直接在IDE中在线调试，Linux应用程序通常是加printf输出log去调试，这种方式简单，但是有些隐藏的程序bug只通过加打印信息不那么容易定位，这时可以通过类似单片机调试的gdb调试来实现，本篇为大家介绍linux环境下在线调试环境的搭建，希望对大家有所帮助。

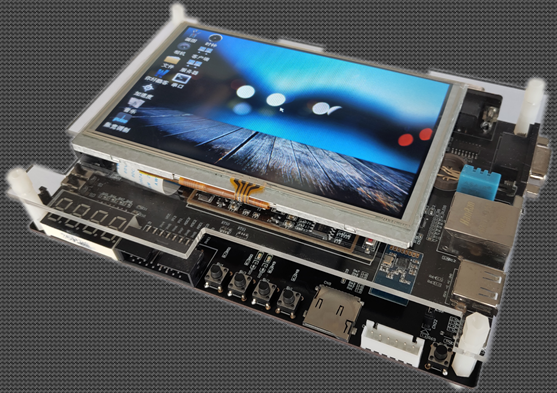
GDB, the GNU Project debugger, allows you to see what is going on `inside' another program while it executes -- or what another program was doing at the moment it crashed.

它的工作原理是：在主机Ubuntu下运行gdb，在嵌入式板子上运行gdbserver，这样就可以在线调试了。

# 2.**环境介绍**

## 2.1.**硬件**

1) 网上的一个第三方做的NUC972开发板：



有兴趣购买的朋友，可以去他们的淘宝店购买：

[https://s.click.taobao.com/X8mza8w](https://s.click.taobao.com/t?e=m%3D2%26s%3DVqeqPgNPc7IcQipKwQzePOeEDrYVVa64LKpWJ%2Bin0XJRAdhuF14FMYcqQp9jM3JMRitN3%2FurF3weYrs2z1V%2BWARwPYjsuD9IQ67nx0X4I%2FwbK4NckI8ZycEnxKDasWxncCadkoABCrnmL8IX2r%2Bngfx81NZbGhxUxiXvDf8DaRs%3D)

## 2.2.**软件**

1) Uboot继续使用之前文章用的，无须改动。

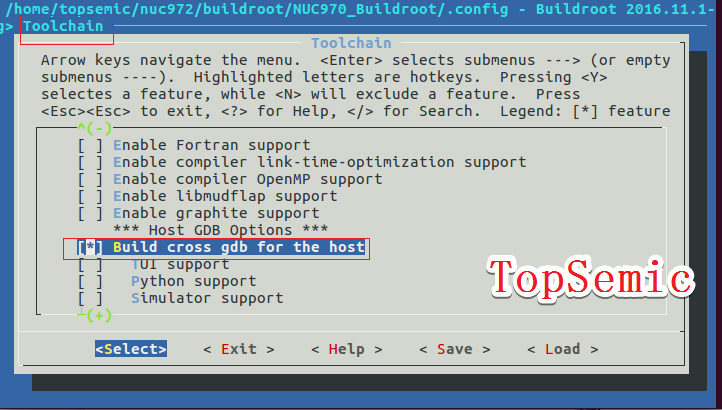
2) Kernel在上一篇基础上，无须改动。

3) Rootfs在上一篇用Buildroot生成的基础上，需要做一定的改动，用来生成gdbserver。

# 3.**Buildroot配置**

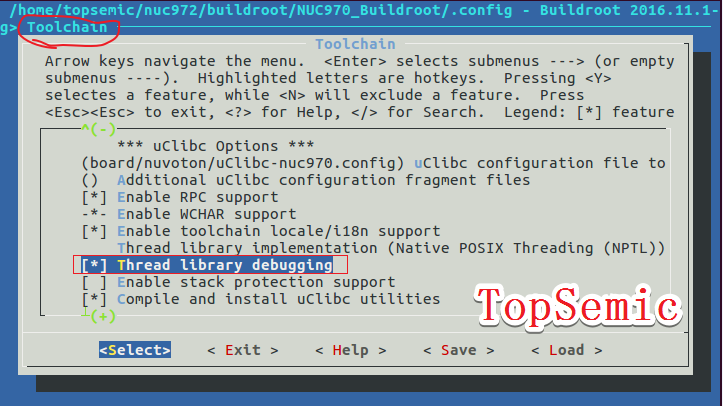
Buildroot里需要做一定的配置，用来生成gdb和gdbserver，步骤如下：

1. 确认Toolchain | Build cross gdb for the host 是否选中，这个默认是选中的。

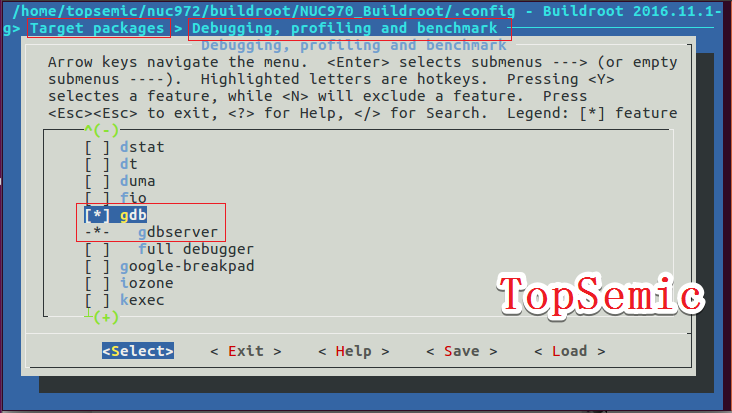


这个的作用是：Build a cross gdb that runs on the host machine and debugs programs running on the target. It requires 'gdbserver' installed on the target。

1. 选中Toolchain下的Thread library debugging，注意一定得先选中这个，不然第三步无法执行。



1. 选中Target packages | Debugging, profiling and benchmark->gdb和gdbserver



上面的作用是：

This option allows to build gdbserver and/or the gdb debugger for the target. For embedded development, the most common solution is to build only 'gdbserver' for the target, and use a cross-gdb on the host.

1. 保存，编译即可。

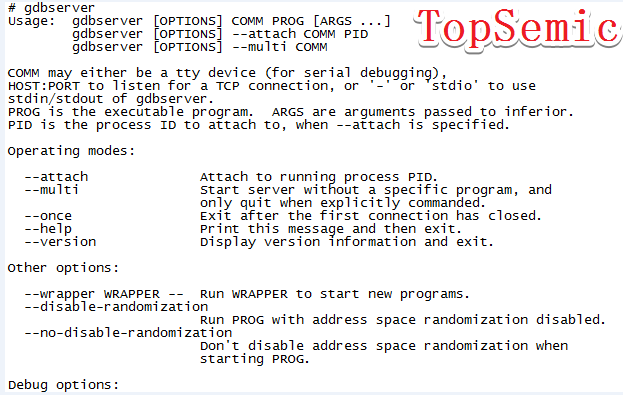
生成的gdb位于：/home/topsemic/nuc972/buildroot/NUC970\_Buildroot/output/host/usr/bin

目录中

生成的gdbserver位于：

/home/topsemic/nuc972/buildroot/NUC970\_Buildroot/output/target/usr/bin 目录中

1. 将上述gdbserver直接放到板子的/usr/bin目录里即可，然后登录板子输入gdbserver，可以看到如下信息，说明板子的gdbserver已经搭建好了。



# 4.**新建测试程序**

1）新建一个测试程序gdbtest.c

#include <stdio.h>

int main()

{

char s[64] = "Welcome to www.topsemic.com";

int a = 1;

int c = a\*2;

int \*ptr = NULL;

printf("s is :%s\n", s);

printf("c is : %d\n", c);

\*ptr = 20;

printf("%d\n",\*ptr);

return 0;

}

2）交叉编译

topsemic@topsemic-virtual-machine:~/nuc972/examples/gdbserver$ arm-linux-gcc gdbtest.c -o gdbtest -g

注：arm-linux-gcc gdbtest.c -o gdbtest -g其中”-g”参数表示进行 GDB 编译。

这个程序放到板子里运行结果如下：



我们用下面的在线调试方法去看看什么原因导致的Segmentation fault

# 5.**在线调试**

调试前，将板子和PC之间通过网线相连接，步骤如下：

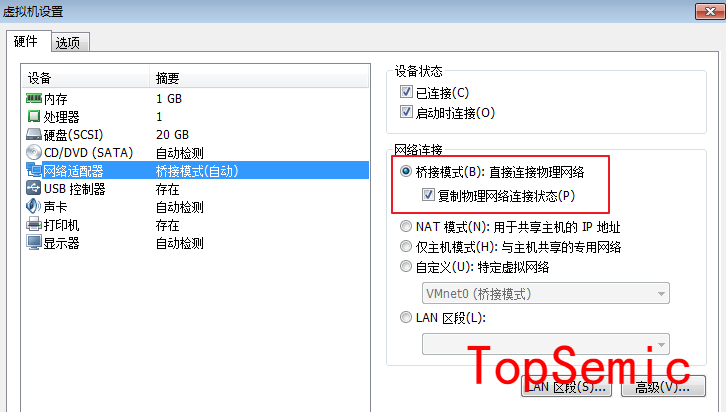
1. 在开发板可执行程序所在的目录下，执行如下命令启动gdbserver：



命令格式： gdbserver <Host\_IP>:<Ports><Program><Arguments...>

192.168.0.80 为Ubuntu 的 IP 地址， 1234 为连接的端口号

注：需要先将虚拟机Ubuntu的IP配置为固定的192.168.0.80，这个设置方法在《Linux学习系列八：操作网口》中有介绍

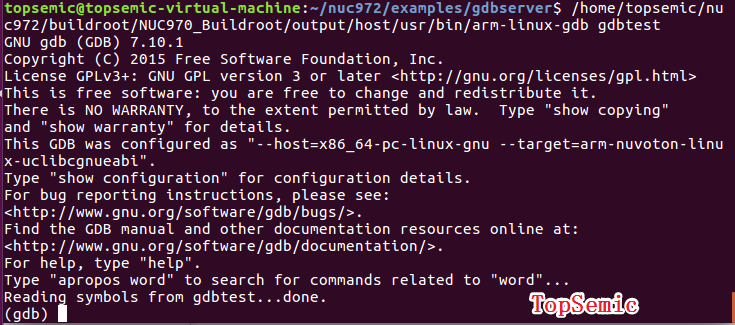




1. 在Ubuntu下启动gdb调试，命令格式：

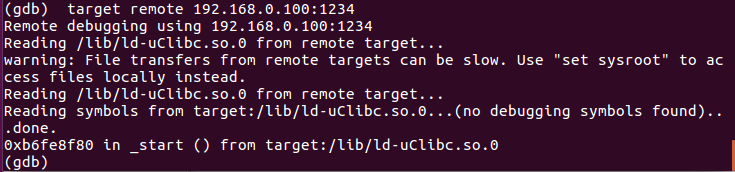
<GDB 可执行程序路径> <应用程序路径>

topsemic@topsemic-virtual-machine:~/nuc972/examples/gdbserver$ /home/topsemic/nuc972/buildroot/NUC970\_Buildroot/output/host/usr/bin/arm-linux-gdb gdbtest



1. 在弹出的上述对话框(gdb)后输入以下命令，连接开发板

(gdb)target remote 192.168.0.100:1234

其中192.168.0.100 是开发板的IP地址

4）之后就可输入如下 GDB 调试命令，其他调试命令的详细用法请输入”help 命令名称”查阅。  
命令： l，参看代码。  
命令： b main，在 main 处设置断点。  
命令： b 9，在第六行设置断点。  
命令： c，继续执行。

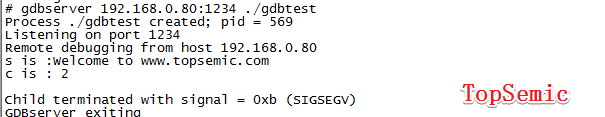
命令： n，单步执行。

命令： q，退出gdb。  
一直输入 c， 直到程序结束。





单步调试，同时查看板子上打印的信息



可以看到板子程序执行的过程和Ubuntu上加的断点运行的进度一致，另外可以

发现是因为line 10 导致的Segmentation fault，这样就定位到了出问题的地方。

注：<https://man.linuxde.net/gdb> 可以看到详细的gdb命令用法。

# 6.**结束语**

本期相关的资料在<https://github.com/TopSemic/NUC972_Linux> Lesson14中

本篇为大家介绍了Linux下使用gdb和gdbserver构建在线调试环境，欢迎大家多交流，可以在网页下方留言讨论，或者发邮件：[Topsemic@sina.com](mailto:Topsemic@sina.com) ，微信公众号如下，欢迎关注：



也可以加入微信群，与我们一起交流

