

1.主题

Tina5.0 修改glibc C库源码进行调试的一种方法

2.问题背景

软件：Tina5.0

背景：在调试一些Tina5.0上的软件包，有时候可能会发现应用上调用了一些系统接口，并没有实际调用到底层驱动的一些接口上。此时，通过strace工具对应用进行调试，会发现应用里虽然调用了该系统接口，但是并没有产生系统调用。此时，就需要对C库的源码进行查阅debug，或者是需要直接对C库的源码进行修改来调试。

本文，主要是介绍了一种修改C库源码，并且对C库进行编译，最终使得自己对C库的修改在小机端上运行的一种**调试方法**。

说明：本方法主要提供了一种调试方法，在实际产品量产中，不建议这样直接修改C库源码。

3.问题描述

3.1复现步骤

无

3.2具体表现

无

4.问题分析

无

5.根本原因

无

6.解决办法

这里以修改mr527_evb方案的aio_write()系统函数为例。

6.1 找到对应的工具链以及C库版本

在source build/envsetup.sh以及lunch选择后对应的方案后，运行：

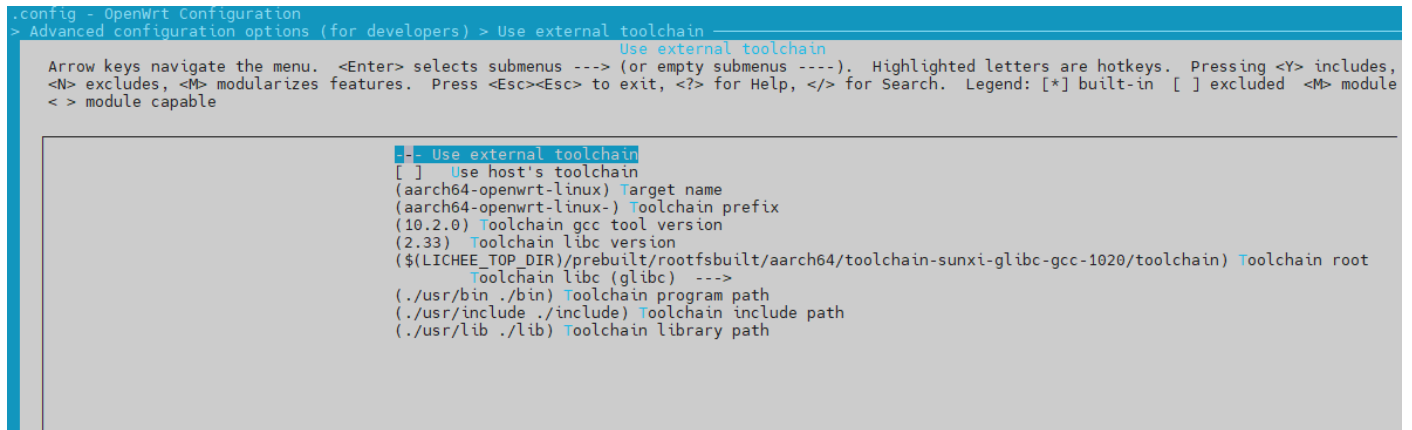
```
m menuconfig
```

找到以下配置：

```
-> Advanced configuration options (for developers) (DEVEL [=y])\
```

```
-> Use external toolchain
```

就得到以下页面：



```
.config - OpenWrt Configuration
> Advanced configuration options (for developers) > Use external toolchain
Use external toolchain
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty submenus ----). Highlighted letters are hotkeys. Pressing <Y> includes,
<N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [*] built-in [ ] excluded <M> module
< > module capable

-- Use external toolchain
[ ] Use host's toolchain
(aarch64-openwrt-linux) Target name
(aarch64-openwrt-linux-) Toolchain prefix
(10.2.0) Toolchain gcc tool version
(2.33) Toolchain libc version
($(LICHEE_TOP_DIR)/prebuilt/rootfsbuilt/aarch64/toolchain-sunxi-glibc-gcc-1020/toolchain) Toolchain root
Toolchain libc (glibc) --->
(./usr/bin ./bin) Toolchain program path
(./usr/include ./include) Toolchain include path
(./usr/lib ./lib) Toolchain library path
```

通过该页面，我们能够得到：

1. gcc工具链的版本为10.2.0，参考配置《(10.2.0) Toolchain gcc tool version》。
2. C库为glibc，参考配置《Toolchain libc (glibc)》。
3. C库版本为2.33，参考配置《(2.33) Toolchain libc version》。

6.2 找到对应的C库源码

通过6.1章节可知，现在想要找到的glibc版本为2.33，一般C库源码是存放到了dl仓库目录下的，路径为：
:\${root_dir}/openwrt/dl

```
$ find -name "glibc*"
```

```
./glibc-2.37-d8e1a7590d375159fb5aac07ad8111ab4699e994.tar.xz
```

```
./glibc-2.33-55446dd8a2d7b84d966fe4248427c02845b036d4.tar.xz
```

那么就能找到glibc-2.33的源码压缩包glibc-2.33-55446dd8a2d7b84d966fe4248427c02845b036d4.tar.xz，如果dl目录下找不到对应的压缩包，就需要自行到glibc的官网进行下载了。

通过命令tar -zxvf glibc-2.33-55446dd8a2d7b84d966fe4248427c02845b036d4.tar.xz，即可解压压缩包，对源码进行查阅。

6.3 修改配置

首先需要修改系统menuconfig配置：

把原先的Use external toolchain取消选中

```
-> Advanced configuration options (for developers) (DEVEL [=y])\
```

```
   Use external toolchain
```

然后选上Toolchain Options，按照以下配置：

```
-> Advanced configuration options (for developers) (DEVEL [=y])\
```

```
   Toolchain Options
```

```
    Binutils Version (Binutils 2.35.1)
```

```
    GCC compiler Version (gcc 10.x)
```

```
     Build/install fortran compiler?
```

```
    C Library implementation (Use glibc)
```

上述四个配置的说明：

1. 不同工具链版本对应着不同的Binutils版本，source完环境变量后，运行objdump – version即可知道当前使用的版本了。
2. gcc版本，10.20就选10.x。
3. 选择编译工具链。
4. C库选择glibc。

提示：这里最好独立创建一份新的代码进行编译C库，因为然后选上Toolchain Options之后，需要进行make distclean才能正常编译SDK，不然可能会引发各种编译错误，导致原来的方案编译不通过。将编译方案的代码和编译C库的代码独立开来，方便调试。

6.4 编译C库

我们找到了aio_write的源码路径为：C库路径/sysdeps/pthread/aio_write.c。

修改完C库的源码后，需要编译C库然后让其实际生效到小机端中。

然后，对解压出来的，已经修改过的C库源码，需要重新压缩。

```
tar -Jcvf glibc-2.33-55446dd8a2d7b84d966fe4248427c02845b036d4.tar.xz glibc-2.33/
```

最后，运行编译C库的命令：

```
m openwrt_rootfs toolchain/glibc/compile -j32 V=s
```

6.5 将修改过后的C库生效到小机端

运行cout找到方案的生成文件目录：

```
$ cout
```

```
$ find -name glibc-2.33*
```

```
./build_dir/toolchain/glibc-2.33-final #C库编译生成物
```

```
./build_dir/toolchain/glibc-2.33 #C库解压过来的源码
```

```
./build_dir/toolchain/glibc-2.33-headers #C库头文件
```

前面说到，我们修改的是aio_write，查看C库源码的aio_write.c目录下的Makefile，得知最终

aio_write.c会生成一个librt.so的文件。

然后，在小机端的目录/lib也找到了librt.so

```
root@TinaLinux:/# ls lib/librt-2.23.so
```

```
lib/librt-2.23.so
```

```
root@TinaLinux:/#
```

最后，直接通过adb push 本地的librt.so到小机端，并且重新命名为librt-2.23.so，即可**生效**。

若还需要重新修改C库源码，重复章节6.4和6.5的动作即可。