

六 Qtopia2.2.0 系统开发环境搭建以及编译镜像

搭建 Qtopia2.2.0 开发环境，需要先搭建 Android 的编译环境，然后在 Android 编译环境的基础上，再搭建 Qtopia2.2.0 编译环境。

Qtopia2.2.0 的编译环境看似复杂，用户只要抓住几个要点就可以了。

第一：编译器。编译器在光盘中都有提供，在需要使用的步骤中，说明其在光盘中的位置。

第二：设置环境变量。环境变量设置后，编译的时候，系统才能找到编译器。

第三：库文件。搭建过程中会给通过执行简单的脚本命令来安装库文件，复杂的步骤变的简单有效。

第四：源码。官网下载的 Qtopia2.2.0 的源文件有少量的 Bug，经过迅为工程师的修改已经可以直接使用，源码修改这一步用户可以直接跳过。

如果用户是使用“搭建好的 Ubuntu 镜像”，则只需要改一下环境变量，系统里面的工具和库文件都已经安装完毕了。

6.1 uboot 的编译

Qtopia2.2.0 系统中 Uboot 和 Android4.0.3 的 Uboot 源码,编译器，参数配置，编译都是通用的，参考 5.3.1 小节。

6.2 Linux 内核的编译

Qtopia2.2.0 系统中 Linux 内核和 Android4.0.3 中的 Linux 内核源码是一样的，编译环境和编译方法也一样，参考 5.2 小节。

6.2.1 参数配置

Qtopia2.2.0 文件系统对应的内核，源码以及编译环境都和 Android4.0.3 的内核一样。

主要是配置文件不一样。

内核的编译是组合式配置文件，基本的配置文件名是 “config_for_linux_YY_elite” ，YY 表示用下表所示的参数替代。

| 硬件分类 | 配置文件 |
|---------------------|------------------------------|
| 核心板 SCP 1G 或者 2G 内存 | config_for_linux_scp_elite |
| 核心板 POP 1G 内存 | config_for_linux_pop_elite |
| 核心板 POP 2G 内存 | config_for_linux_pop2G_elite |

6.2.2 编译生成内核镜像举例

这里以 SCP 1G 核心板为例编译 zImage 内核镜像,那么配置文件为

“config_for_linux_scp_elite” 。

将光盘 “06_源码_uboot 和 kernel” 目录下的压缩包

“iTop4412_Kernel_3.0_xxx.tar.gz” 拷贝到 Ubuntu ，然后解压，得到文件夹

“iTop4412_Kernel_3.0 ” ，如下图所示。

```
root@ubuntu: /home/topeet/android4.0
root@ubuntu: /home/topeet/android4.0# ls
CodeSign4SecureBoot_POP  iTop4412_Kernel_3.0_20151120.tar.gz
CodeSign4SecureBoot_SCP  iTop4412_uboot
iTop4412_Kernel_3.0      iTop4412_uboot_20151119.tar.gz
root@ubuntu: /home/topeet/android4.0#
```

进入文件夹 “iTop4412_Kernel_3.0 ” ，使用命令

“cp config_for_linux_scp_elite .config” 覆盖自带的配置文件，如下图所示。

```
root@ubuntu: /home/topeet/android4.0# cd iTop4412_Kernel_3.0
root@ubuntu: /home/topeet/android4.0/iTop4412_Kernel_3.0# cp config_for_linux_scp
_elite .config
root@ubuntu: /home/topeet/android4.0/iTop4412_Kernel_3.0#
```

然后使用编译命令 “make zImage” ，如下图所示。

```
root@ubuntu:/home/topeet/android4.0# cd iTop4412_Kernel_3.0
root@ubuntu:/home/topeet/android4.0/iTop4412_Kernel_3.0# cp config_for_linux_scp
_elite .config
root@ubuntu:/home/topeet/android4.0/iTop4412_Kernel_3.0# make zImage
```

编译中，如下图所示。

```
root@ubuntu:/home/topeet/android4.0# cd iTop4412_Kernel_3.0
root@ubuntu:/home/topeet/android4.0/iTop4412_Kernel_3.0# cp config_for_linux_scp
_elite .config
root@ubuntu:/home/topeet/android4.0/iTop4412_Kernel_3.0# make zImage
scripts/kconfig/conf --silentoldconfig Kconfig
CHK include/linux/version.h
CHK include/generated/utsrelease.h
make[1]: `include/generated/mach-types.h' is up to date.
CALL scripts/checksyscalls.sh
CHK include/generated/compile.h
CC arch/arm/kernel/setup.o
LD arch/arm/kernel/built-in.o
```

编译完成，如下图所示。

```
CC      init/version.o
LD      init/built-in.o
LD      .tmp_vmlinux1
KSYM    .tmp_kallsyms1.S
AS      .tmp_kallsyms1.o
LD      .tmp_vmlinux2
KSYM    .tmp_kallsyms2.S
AS      .tmp_kallsyms2.o
LD      vmlinux
SYSMAP  System.map
SYSMAP  .tmp_System.map
OBJCOPY arch/arm/boot/Image
Kernel: arch/arm/boot/Image is ready
AS      arch/arm/boot/compressed/head.o
GZIP    arch/arm/boot/compressed/piggy.gzip
AS      arch/arm/boot/compressed/piggy.gzip.o
CC      arch/arm/boot/compressed/misc.o
CC      arch/arm/boot/compressed/decompress.o
SHIPPED arch/arm/boot/compressed/lib1funcs.S
AS      arch/arm/boot/compressed/lib1funcs.o
LD      arch/arm/boot/compressed/vmlinux
OBJCOPY arch/arm/boot/zImage
Kernel: arch/arm/boot/zImage is ready
root@ubuntu:/home/topeet/android4.0/iTop4412_Kernel_3.0#
```

文件夹 “iTop4412_Kernel_3.0 ” 下的 “arch” --> “arm” --> “boot” 会生成镜像文件 “zImage” ，这个 zImage 镜像可以给 **SCP 1G** 和 **SCP 2G** 的核心板使用，如下图所示。

```
LD      .tmp_vmlinux1
KSYM    .tmp_kallsyms1.S
AS      .tmp_kallsyms1.o
LD      .tmp_vmlinux2
KSYM    .tmp_kallsyms2.S
AS      .tmp_kallsyms2.o
LD      vmlinux
SYSMAP  System.map
SYSMAP  .tmp_System.map
OBJCOPY arch/arm/boot/Image
Kernel: arch/arm/boot/Image is ready
AS      arch/arm/boot/compressed/head.o
GZIP    arch/arm/boot/compressed/piggy.gzip
AS      arch/arm/boot/compressed/piggy.gzip.o
CC      arch/arm/boot/compressed/misc.o
CC      arch/arm/boot/compressed/decompress.o
SHIPPED arch/arm/boot/compressed/lib1funcs.S
AS      arch/arm/boot/compressed/lib1funcs.o
LD      arch/arm/boot/compressed/vmlinux
OBJCOPY arch/arm/boot/zImage
Kernel: arch/arm/boot/zImage is ready
root@ubuntu:/home/topeet/android4.0/iTop4412_Kernel_3.0# ls arch/arm/boot/
bootp  compressed  Image  install.sh  Makefile  zImage
root@ubuntu:/home/topeet/android4.0/iTop4412_Kernel_3.0#
```

6.3 Qtopia2.2.0 编译的环境以及编译

针对 Qt 文件系统，迅为电子在 iTOP-4412 开发板上移植的是 Qtopia2.2.0 版本和 Qte4.7.1 版本，用户在参照本章节后编译后的文件图形界面是 Qtopia2.2.0 版本。Qte4.7.1 的编译方法则在第七章。

如果用户使用的是“搭建好的 Ubuntu 镜像”，则只需要修改一下环境变量。

6.3.1 编译器和基本库文件的安装

Qtopia2.2.0 文件系统的编译器和 Android4.0.3 的编译器不一样，Qtopia2.2.0 的编译器包含在用户光盘 “08_源码_QtE 以及 qtopia2.2.0 文件系统” 文件夹的压缩包 “arm-linux-4.4.1.tar.q” 中，如下图所示。

| 名称 | 修改日期 | 类型 | 大小 |
|---|--------------|-------|----------|
| patch | 2015/7/4 ... | 文件夹 | |
| 3rdpart-lib-for-Qtopia2.2.0.tar.gz | 2015/7/4 ... | 360压缩 | 8,618... |
| arm-linux-4.4.1.tar.gz | 2015/7/4 ... | 360压缩 | 301,6... |
| arm-linux-gcc-4.3.2.tar.gz | 2015/7/4 ... | 360压缩 | 86,03... |
| ARM-qtopia-free-src-2.2.0.tar.gz | 2015/7/4 ... | 360压缩 | 79,16... |
| iTop4412_Kernel_3.0_20150109.tar.gz | 2015/7/4 ... | 360压缩 | 131,0... |
| PC-qtopia-free-src-2.2.0.tar.gz | 2015/7/4 ... | 360压缩 | 51,14... |
| qt-everywhere-opensource-src-4.7.1_20141224.ta... | 2015/7/4 ... | 360压缩 | 205,9... |
| root_20150123.tar.gz | 2015/7/4 ... | 360压缩 | 79,41... |

将压缩包拷贝到 Ubuntu 系统下，如下图所示。

```
root@ubuntu:~# ls /usr/local/arm/
arm-2009q3  arm-2009q3.tar.bz2  arm-linux-4.4.1.tar.gz
root@ubuntu:~#
```

接着将压缩包解压到 Ubuntu 系统的文件夹 “usr” --> “local” --> “arm” 下，进入 “/usr/local/arm” 目录使用解压命令 “tar -vxf arm-linux-4.4.1.tar.gz” 解压 “arm-linux-4.4.1.tar.gz” ，如下图所示。

```
root@ubuntu:~# ls /usr/local/arm/  
arm-2009q3  arm-2009q3.tar.bz2  arm-linux-4.4.1.tar.gz  
root@ubuntu:~# cd /usr/local/arm/  
root@ubuntu:/usr/local/arm# tar -vxf arm-linux-4.4.1.tar.gz
```

如下图所示，解压完成，生成了文件夹“4.4.1”文件夹。

```
4.4.1/bin/arm-linux-gcc-4.4.1
4.4.1/bin/arm-none-linux-gnueabi-ar
4.4.1/bin/arm-none-linux-gnueabi-addr2line
4.4.1/bin/arm-none-linux-gnueabi-ld
4.4.1/bin/arm-linux-gcc
4.4.1/bin/arm-linux-sprite
root@ubuntu:/usr/local/arm# ls
4.4.1 arm-2009q3 arm-2009q3.tar.bz2 arm-linux-4.4.1.tar.gz
root@ubuntu:/usr/local/arm#
```


然后安装 X11 的 SDK 库，执行命令 “apt-get install libx11-dev libxext-dev libxtst-dev”，如下图所示。

```
root@ubuntu:/usr/local/arm# ls
4.4.1 arm-2009q3 arm-2009q3.tar.bz2 arm-linux-4.4.1.tar.gz
root@ubuntu:/usr/local/arm# cd
root@ubuntu:~#
root@ubuntu:~# apt-get install libx11-dev libxext-dev libxtst-dev
```

安装库过程提示是否要继续，如下图所示，选择 “y”，继续。

```
The following NEW packages will be installed:
  libxi-dev libxtst-dev x11proto-record-dev
The following packages will be upgraded:
  libxi6 libxtst6
2 upgraded, 3 newly installed, 0 to remove and 503 not upgraded.
Need to get 348 kB of archives.
After this operation, 997 kB of additional disk space will be used.
Do you want to continue [Y/n] y
```

如下图所示，更新完成。

```
Selecting previously unselected package libxtst-dev.
Unpacking libxtst-dev (from .../libxtst-dev_2%3a1.2.0-4ubuntu0.1_amd64.deb) ...
Processing triggers for man-db ...
Setting up libxi6 (2:1.7.1.901-1ubuntu1-precise3) ...
Setting up libxtst6 (2:1.2.0-4ubuntu0.1) ...
Setting up libxi-dev (2:1.7.1.901-1ubuntu1-precise3) ...
Setting up x11proto-record-dev (1.14.1-2) ...
Setting up libxtst-dev (2:1.2.0-4ubuntu0.1) ...
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
root@ubuntu:~#
```

接着修改环境变量，如下图所示，在 root 目录下（使用 cd 命令之后就会回到 root 目录）使用命令 “vim .bashrc”

```
root@ubuntu: ~
root@ubuntu:~#
root@ubuntu:~# cd
root@ubuntu:~# vim .bashrc
```

使用 vim 编辑器打开环境变量文件 “.bashrc” 后，修改 Qtopia2.2.0 编译器的路径，添加 “export PATH=\$PATH:/usr/local/arm/4.4.1/bin”。在文件 “.bashrc” 的最后一行。然后注释掉其它编译器，例如下图所示的 arm-2009q3 编译器。

```
fi
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
# if [ -f /etc/bash_completion ] && ! shopt -oq posix; then
#   . /etc/bash_completion
# fi
#export PATH=$PATH:/usr/local/arm/arm-2009q3/bin
export PATH=$PATH:/usr/local/arm/4.4.1/bin
-- INSERT --
```

修改完成后保存退出 “.bashrc” 文件。

```
root@ubuntu: ~
root@ubuntu:~#
root@ubuntu:~# cd
root@ubuntu:~# vim .bashrc
root@ubuntu:~#
```

更新一下环境变量，如下图所示，使用命令 “source .bashrc ” 更新环境变量。

```
root@ubuntu: ~
root@ubuntu:~#
root@ubuntu:~# cd
root@ubuntu:~# vim .bashrc
root@ubuntu:~# source .bashrc
root@ubuntu:~#
```

这里测试一下编译器是否正确安装，执行下命令 “arm-none-linux-gnueabi-gcc -v” ，如下图所示。

```
root@ubuntu:~#
root@ubuntu:~# cd
root@ubuntu:~# vim .bashrc
root@ubuntu:~# source .bashrc
root@ubuntu:~# arm-none-linux-gnueabi-gcc -v
```

如下图所示，可以看到系统显示 arm-gcc 编译器的版本为 “gcc version 4.4.1” 。

```
3-respin-linux-lite/obj/host-libs-2009q3-67-arm-none-linux-gnueabi-t686-pc-linux
-gnu/usr --disable-libgomp --enable-poison-system-directories --with-build-time-
tools=/scratch/julian/2009q3-respin-linux-lite/install/arm-none-linux-gnueabi/bi
n --with-build-time-tools=/scratch/julian/2009q3-respin-linux-lite/install/arm-n
one-linux-gnueabi/bin
Thread model: posix
gcc version 4.4.1 (Sourcery G++ Lite 2009q3-67)
root@ubuntu:~#
```

注意，在前面搭建 Android4.0.3 编译环境的时候，其中提到了一步操作 “降低 gcc 版本”，但是前面那个 “gcc” 是 x86 的编译器（通过命令 #gcc -v 可以查看其版本）。这里用到的 “gcc” 编译器是 arm 编译器，它们是两个完全不同的编译器，大家不要弄混了。

6.3.2 Qtopia2.2.0 源文件和补丁文件

在 Ubuntu 环境中，“root” 目录下新建文件夹 “yizhi”，具体操作如下，在 Ubuntu 命令行中，执行命令 “cd /root” 和 “mkdir yizhi”。

这里需要注意的是，新建的文件夹一定要在这个 “root” 文件夹下建立，而且一定要使用 “yizhi” 这个名字。

如下图所示。

```
Thread model: posix
gcc version 4.4.1 (Sourcery G++ Lite 2009q3-67)
root@ubuntu:~# cd /root/
root@ubuntu:~# mkdir yizhi
root@ubuntu:~# ls
yizhi
root@ubuntu:~#
```

接着找到用户光盘 “08_源码_QtE 以及 qtopia2.2.0 文件系统” 文件夹下的压缩包 “ARM-qtopia-free-src-2.2.0.tar.gz” 如下图所示。

| 名称 | 修改日期 | 类型 | 大 |
|---|--------------|-------|-----|
| patch | 2015/7/4 ... | 文件夹 | |
| 3rdpart-lib-for-Qtopia2.2.0.tar.gz | 2015/7/4 ... | 360压缩 | 8,4 |
| arm-linux-4.4.1.tar.gz | 2015/7/4 ... | 360压缩 | 30 |
| arm-linux-gcc-4.3.2.tar.gz | 2015/7/4 ... | 360压缩 | 86 |
| ARM-qtopia-free-src-2.2.0.tar.gz | 2015/7/4 ... | 360压缩 | 79 |
| iTop4412_Kernel_3.0_20150109.tar.gz | 2015/7/4 ... | 360压缩 | 13 |
| PC-qtopia-free-src-2.2.0.tar.gz | 2015/7/4 ... | 360压缩 | 51 |
| qt-everywhere-opensource-src-4.7.1_20141224.ta... | 2015/7/4 ... | 360压缩 | 20 |
| root_20150123.tar.gz | 2015/7/4 ... | 360压缩 | 79 |

将压缩包 “ARM-qtopia-free-src-2.2.0.tar.gz” 拷贝到前面新建的 “yizhi” 文件夹中，如下图所示。

```
root@ubuntu:~# cd /root/
root@ubuntu:~# mkdir yizhi
root@ubuntu:~# ls
yizhi
root@ubuntu:~# ls yizhi/
ARM-qtopia-free-src-2.2.0.tar.gz
root@ubuntu:~#
```

进入 “yizhi” 目录，使用命令 “tar -vxf ARM-qtopia-free-src-2.2.0.tar.gz” 解压压缩包，如下图所示。

```

root@ubuntu:~# ls
yizhi
root@ubuntu:~# ls yizhi/
ARM-qtopia-free-src-2.2.0.tar.gz
root@ubuntu:~# cd yizhi/
root@ubuntu:~/yizhi# tar -vxf ARM-qtopia-free-src-2.2.0.tar.gz

```

解压压缩包后得到源码文件 “qtopia-free-src-2.2.0.tar.gz” 和脚本文件 “build” , 如下图所示。

```

root@ubuntu:~# ls yizhi/
ARM-qtopia-free-src-2.2.0.tar.gz
root@ubuntu:~# cd yizhi/
root@ubuntu:~/yizhi# tar -vxf ARM-qtopia-free-src-2.2.0.tar.gz
build
qtopia-free-src-2.2.0.tar.gz
root@ubuntu:~/yizhi# ls
ARM-qtopia-free-src-2.2.0.tar.gz build qtopia-free-src-2.2.0.tar.gz
root@ubuntu:~/yizhi#

```

这里需要注意的是，用户光盘里面提供的 qtopia2.2.0 源文件是经过修改的，没有 bug 的代码，如果大家对 qtopia2.2.0 官网的源码感兴趣，可以在网盘下载源码（网盘中的源码是从 qt 官网下载的源代码）。但是假如用户在这里使用 qtopia2.2.0 官网的源码，在最后编译的时候会报很多错，这些错误需要额外的方法去排除，具体排除的方法可以参考后面的附录二。

用户光盘 “08_源码_QtE 以及 qtopia2.2.0 文件系统” --> “patch” 文件夹下的压缩包 “tslib.tar.gz” 是触摸的库文件，如下图所示。

| 名称 | 修改日期 | 类型 | 大小 |
|------------------|--------------|-------|--------|
| libICE.so.6.3.0 | 2015/7/4 ... | 0 文件 | 83 KB |
| libSM.so.6.0.1 | 2015/7/4 ... | 1 文件 | 26 KB |
| libuuid.so.1.3.0 | 2015/7/4 ... | 0 文件 | 14 KB |
| libXext.so.6.4.0 | 2015/7/4 ... | 0 文件 | 55 KB |
| libXmu.so.6.2.0 | 2015/7/4 ... | 0 文件 | 87 KB |
| libXt.so.6.0.0 | 2015/7/4 ... | 0 文件 | 324 KB |
| tslib.tar.gz | 2015/7/4 ... | 360压缩 | 59 KB |

将触摸的库文件拷贝到 Ubuntu，然后到 Ubuntu 系统的 “usr” --> “local” 文件夹下，使用命令 “tar -vxf tslib.tar.gz” 解压，如下图所示。

```

root@ubuntu:~#
root@ubuntu:~# cd /usr/local/arm/
root@ubuntu:/usr/local/arm# ls
4.4.1 arm-2009q3 arm-2009q3.tar.bz2 arm-linux-4.4.1.tar.gz tslib.tar.gz
root@ubuntu:/usr/local/arm# tar -vxf tslib.tar.gz
tslib/bin/
tslib/bin/ts_print_raw
tslib/bin/ts_calibrate
tslib/bin/ts_test
tslib/bin/ts_harvest
tslib/bin/ts_print

```

生成的文件夹“tslib”，如下图所示。

```
tslib/lib/pkgconfig/tslib-0.0.pc
tslib/lib/libts.la
tslib/lib/libts-0.0.so.0
tslib/lib/libts-0.0.so.0.1.1
root@ubuntu:/usr/local/arm# ls
4.4.1      arm-2009q3.tar.bz2      tslib
arm-2009q3 arm-linux-4.4.1.tar.gz  tslib.tar.gz
root@ubuntu:/usr/local/arm#
```

6.3.3 库文件和编译 Qtopia2.2.0

编译 Qtopia2.2.0 文件还需要一些额外的 6 个库文件，这 6 个库文件全部在用户光盘

“08_源码_QtE 以及 qtopia2.2.0 文件系统” --> “patch” 文件夹下。这六个库文件分别是：
libXext.so.6.4.0, libXmu.so.6.2.0, libSM.so.6.0.1, libICE.so.6.3.0, libXt.so.6.0.0,
libuuid.so.1.3.0

如下图所示。

| 名称 | 修改日期 | 类型 | 大小 |
|------------------|--------------|-------|--------|
| libICE.so.6.3.0 | 2015/7/4 ... | 0 文件 | 83 KB |
| libSM.so.6.0.1 | 2015/7/4 ... | 1 文件 | 26 KB |
| libuuid.so.1.3.0 | 2015/7/4 ... | 0 文件 | 14 KB |
| libXext.so.6.4.0 | 2015/7/4 ... | 0 文件 | 55 KB |
| libXmu.so.6.2.0 | 2015/7/4 ... | 0 文件 | 87 KB |
| libXt.so.6.0.0 | 2015/7/4 ... | 0 文件 | 324 KB |
| tslib.tar.gz | 2015/7/4 ... | 360压缩 | 59 KB |

其中的 5 个库文件，包括“libXext.so.6.4.0”、“libXmu.so.6.2.0”、“libSM.so.6.0.1”、“libICE.so.6.3.0”、“libXt.so.6.0.0”，全部拷贝到 Ubuntu 系统的文件夹“usr” --> “lib32” 下。

然后创建链接文件，具体操作如下，进入 Ubuntu 系统的文件夹“usr” --> “lib32” 下，然后在 Ubuntu 命令行中，执行下面的命令：

```
ln -s libXext.so.6.4.0 libXext.so.6
```

创建链接文件 libXext.so.6

```
ln -s libXext.so.6 libXext.so
```

创建链接文件 libXext.so

```
ln -s libXmu.so.6.2.0 libXmu.so.6
```

创建链接文件 libXmu.so.6

```
ln -s libXmu.so.6 libXmu.so
```

创建链接文件 libXmu.so

```
ln -s libSM.so.6.0.1 libSM.so.6
```

创建链接文件 libSM.so.6

```
ln -s libSM.so.6 libSM.so
```

创建链接文件 libSM.so

```
ln -s libICE.so.6.3.0 libICE.so.6
```

创建链接文件 libICE.so.6

```
ln -s libICE.so.6 libICE.so
```

创建链接文件 libICE.so

```
ln -s libXt.so.6.0.0 libXt.so.6
```

创建链接文件 libXt.so.6

```
ln -s libXt.so.6 libXt.so
```

创建链接文件 libXt.so

如下图所示，文件拷贝完成之后，执行创建链接的命令。

```
root@ubuntu: /usr/lib32
root@ubuntu:~# cd /usr/lib32/
root@ubuntu:/usr/lib32# ln -s libXext.so.6.4.0 libXext.so.6
root@ubuntu:/usr/lib32# ln -s libXext.so.6 libXext.so
root@ubuntu:/usr/lib32# ln -s libXmu.so.6.2.0 libXmu.so.6
root@ubuntu:/usr/lib32# ln -s libXmu.so.6 libXmu.so
root@ubuntu:/usr/lib32# ln -s libSM.so.6.0.1 libSM.so.6
root@ubuntu:/usr/lib32# ln -s libSM.so.6 libSM.so
root@ubuntu:/usr/lib32# ln -s libICE.so.6.3.0 libICE.so.6
root@ubuntu:/usr/lib32# ln -s libICE.so.6 libICE.so
root@ubuntu:/usr/lib32# ln -s libXt.so.6.0.0 libXt.so.6
root@ubuntu:/usr/lib32# ln -s libXt.so.6 libXt.so
root@ubuntu:/usr/lib32#
```

拷贝剩下的文件 “libuuid.so.1.3.0” 到 Ubuntu 系统的 “lib32” 文件夹下，然后在 Ubuntu 命令行中，执行下面的命令：

```
ln -s libuuid.so.1.3.0 libuuid.so.1
```

创建链接文件 libuuid.so.1

```
ln -s libuuid.so.1 libuuid.so
```

创建链接文件 libuuid.so

如下图所示，文件拷贝完成之后，执行创建链接的命令。

```
root@ubuntu:/usr/lib32# cd /lib32
root@ubuntu:/lib32# ln -s libuuid.so.1.3.0 libuuid.so.1
root@ubuntu:/lib32# ln -s libuuid.so.1 libuuid.so
root@ubuntu:/lib32#
```

库文件全部处理完成后，接着就可以编译 Qtopia2.2.0 源码了，使用命令 “cd /root/yizhi” 进入 Qtopia2.2.0 源码文件夹，执行编译脚本命令 “./build”，如下图所示。

```
root@ubuntu:/lib32# ln -s libuuid.so.1.3.0 libuuid.so.1
root@ubuntu:/lib32# ln -s libuuid.so.1 libuuid.so
root@ubuntu:/lib32# cd /root/yizhi/
root@ubuntu:~/yizhi# ./build
```


编译 qtopia2.2.0 源文件是一个比较漫长的过程。

编译完成后会在 Ubuntu 系统文件夹 “root” --> “yizhi” 下生成文件夹 “qtopia-free-2.2.0” ，这个文件夹就是编译好的 Qtopia2.2.0 文件，如下图所示。

```
make[7]: Leaving directory `/root/yizhi/qtopia-free-2.2.0/qtopia/etc/themes/medi
aplayer/techno'
make[6]: Leaving directory `/root/yizhi/qtopia-free-2.2.0/qtopia/etc/themes'
make[5]: Leaving directory `/root/yizhi/qtopia-free-2.2.0/qtopia/etc/themes'
make[4]: Leaving directory `/root/yizhi/qtopia-free-2.2.0/qtopia/src'
make[3]: Leaving directory `/root/yizhi/qtopia-free-2.2.0/qtopia/src'
make[2]: Leaving directory `/root/yizhi/qtopia-free-2.2.0/qtopia/src'
make[1]: Leaving directory `/root/yizhi/qtopia-free-2.2.0/qtopia'
root@ubuntu:~/yizhi# ls
ARM-qtopia-free-src-2.2.0.tar.gz  qtopia-free-2.2.0
build                             qtopia-free-src-2.2.0.tar.gz
qtopia2.2.0Makelog
root@ubuntu:~/yizhi#
```

在编译好的 Qtopia2.2.0 文件夹下，我们需要接着处理一下字库文件。具体操作如下，在 Ubuntu 命令行中，执行命令 “cp -r /root/yizhi/qtopia-free-2.2.0/qt2/lib/fonts/helvetica* /root/yizhi/qtopia-free-2.2.0/qtopia/image/opt/Qtopia/lib/fonts/” ，如下图所示。

```
root@ubuntu:~/yizhi# ls
ARM-qtopia-free-src-2.2.0.tar.gz  qtopia-free-2.2.0
build                             qtopia-free-src-2.2.0.tar.gz
qtopia2.2.0Makelog
root@ubuntu:~/yizhi# cp -r /root/yizhi/qtopia-free-2.2.0/qt2/lib/fonts/helvetica
* /root/yizhi/qtopia-free-2.2.0/qtopia/image/opt/Qtopia/lib/fonts/
```

然后把文件夹“Qtopia”拷贝到 Ubuntu 系统的 opt 文件夹下，具体操作如下，在 Ubuntu 命令行中，执行命令 “cp -r /root/yizhi/qtopia-free-2.2.0/qtopia/image/opt/Qtopia /opt” ，如下图所示。

```
root@ubuntu:~/yizhi# ls
ARM-qtopia-free-src-2.2.0.tar.gz  qtopia-free-2.2.0
build                             qtopia-free-src-2.2.0.tar.gz
qtopia2.2.0Makelog
root@ubuntu:~/yizhi# cp -r /root/yizhi/qtopia-free-2.2.0/qtopia/image/opt/Qtopia
* /root/yizhi/qtopia-free-2.2.0/qtopia/image/opt/Qtopia/lib/fonts/
root@ubuntu:~/yizhi# cp -r /root/yizhi/qtopia-free-2.2.0/qtopia/image/opt/Qtopia
/opt
root@ubuntu:~/yizhi#
```

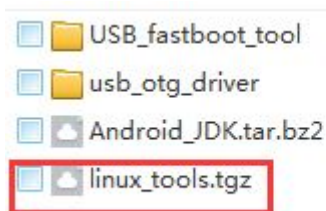
6.3.4 第三方库文件

在前一小节中，编译的时候用到了第三方库文件 “3rdpart-lib-for-Qtopia2.2.0.tar.gz”。由于这个库文件直接包含在提供的编译器压缩包 “arm-linux-4.4.1.tar.gz” 中，用户在前面 6.3.1 小节中，解压编译器压缩包的时候，库文件就已经直接解压到 Ubuntu 系统中了，所以在编译 QT 的时候，用户不用进行额外的处理就可以直接编译生成 QT 文件系统。

如果用户使用自己下载的编译器，那么就需要自己编译第三方库文件，第三方库文件具体的编译方法可以参考附录一。

6.3.5 生成 system.img

生成可以下载的 system.img 文件需要工具 “mkimage”，这个工具在用户光盘 “02_编译器以及烧写工具” → “tools” 文件夹下的压缩包 “linux_tools.tgz” 中，如下图所示。



拷贝压缩包到 Ubuntu 系统的 “/” 目录下，注意目录是 “/”。

```
root@ubuntu:~/yizhi# cd /
root@ubuntu:/# ls
bin    dev    initrd.img  lib64      media  proc  sbin    sys    var
boot  etc    lib         linux_tools.tgz  mnt    root  selinux tmp    vmlinuz
cdrom home lib32      lost+found  opt    run   srv     usr
```

进入 “/” 目录，然后使用命令 “tar -vxf linux_tools.tgz”，将压缩包解压。

```
root@ubuntu:/# cd /
root@ubuntu:/# tar -vxf linux_tools.tgz
```

解压后如下图所示，在 “/usr/local/bin/” 目录下生成了两个文件。

```
root@ubuntu:/# ls /usr/local/bin/
make_ext4fs  mkimage
root@ubuntu:/#
```

使用命令 “cd /home/topeet/” 进入 topeet 目录，然后使用命令 “mkdir Linux+QT”

新建一个 “Linux+QT” 文件夹，如下图所示。

```
root@ubuntu:~# cd /home/topeet/
root@ubuntu:/home/topeet# mkdir Linux+QT
root@ubuntu:/home/topeet# ls
android4.0      Desktop      examples.desktop  Pictures      Videos
Android_JDK    Documents   Linux+QT          Public
Android_JDK.tar.bz2 Downloads   Music            Templates
root@ubuntu:/home/topeet#
```

找到用户光盘 “08_源码_QtE 以及 qtopia2.2.0 文件系统” 目录下的压缩包

“root.tar.gz”，如下图所示。

| 名称 | 修改日期 |
|---|--------------|
| patch | 2015/7/4 ... |
| 3rdpart-lib-for-Qtopia2.2.0.tar.gz | 2015/7/4 ... |
| arm-linux-4.4.1.tar.gz | 2015/7/4 ... |
| arm-linux-gcc-4.3.2.tar.gz | 2015/7/4 ... |
| ARM-qtopia-free-src-2.2.0.tar.gz | 2015/7/4 ... |
| iTop4412_Kernel_3.0_20150109.tar.gz | 2015/7/4 ... |
| PC-qtopia-free-src-2.2.0.tar.gz | 2015/7/4 ... |
| qt-everywhere-opensource-src-4.7.1_20141224.ta... | 2015/7/4 ... |
| root_20150123.tar.gz | 2015/7/4 ... |

拷贝用户光盘 “linux” 目录下的压缩包 “root.tar.gz” 到新建的 “Linux+QT” 文件夹

下，如下图所示。

```
root@ubuntu:/home/topeet#
root@ubuntu:/home/topeet# cd Linux+QT/
root@ubuntu:/home/topeet/Linux+QT# ls
root_20150123.tar.gz
root@ubuntu:/home/topeet/Linux+QT#
```

使用命令 “tar -vxf root_20150123.tar.gz” 解压压缩包，如下图所示。

```
root@ubuntu:/home/topeet# cd Linux+QT/
root@ubuntu:/home/topeet/Linux+QT# ls
root_20150123.tar.gz
root@ubuntu:/home/topeet/Linux+QT# tar -vxf root_20150123.tar.gz
```

解压后会生成文件夹“root”，如下图所示。

```
root/root/Settings/contacts.conf
root/root/Settings/Beam.conf
root/root/Settings/MineSweep.conf
root/root/Settings/Categories.xml
root/root/Settings/handwriting.conf
root/dev/
root@ubuntu:/home/topeet/Linux+QT# ls
root root_20150123.tar.gz
root@ubuntu:/home/topeet/Linux+QT#
```

然后把前面编译生成的文件夹“Qtopia”拷贝到解压出来的“opt”文件夹中，具体操作如下，在 Ubuntu 命令行中，执行命令“cp -r /root/yizhi/qtopia-free-2.2.0/qtopia/image/opt/Qtopia /home/topeet/Linux+QT/root/opt”

```
root/root/Settings/handwriting.conf
root/dev/
root@ubuntu:/home/topeet/Linux+QT# ls
root root_20150123.tar.gz
root@ubuntu:/home/topeet/Linux+QT# cp -r /root/yizhi/qtopia-free-2.2.0/qtopia/image/opt/Qtopia /home/topeet/Linux+QT/root/opt
```

注意红色的 topeet 是用户文件夹，如果用户自己搭建环境，则需要替换成自己设置的用户名。

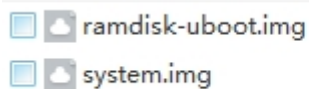
在执行上面的操作后，最后执行生成二进制文件的命令，在目录“/home/topeet/Linux+QT”中，使用命令“make_ext4fs -s -l 314572800 -a root -L linux system.img root”，如下图所示。

```
root root_20150123.tar.gz
root@ubuntu:/home/topeet/Linux+QT# cp -r /root/yizhi/qtopia-free-2.2.0/qtopia/image/opt/Qtopia /home/topeet/Linux+QT/root/opt
root@ubuntu:/home/topeet/Linux+QT# make_ext4fs -s -l 314572800 -a root -L linux system.img root
Creating filesystem with parameters:
  Size: 314572800
  Block size: 4096
  Blocks per group: 32768
  Inodes per group: 6400
  Inode size: 256
  Journal blocks: 1200
  Label: linux
  Blocks: 76800
  Block groups: 3
  Reserved block group size: 23
Created filesystem with 10701/19200 inodes and 51541/76800 blocks
root@ubuntu:/home/topeet/Linux+QT#
```

执行这一步后，在“Linux+QT”文件夹中就生成了“system.img”文件，如下图所示。


```
system.img root
Creating filesystem with parameters:
  Size: 314572800
  Block size: 4096
  Blocks per group: 32768
  Inodes per group: 6400
  Inode size: 256
  Journal blocks: 1200
  Label: linux
  Blocks: 76800
  Block groups: 3
  Reserved block group size: 23
Created filesystem with 10701/19200 inodes and 51541/76800 blocks
root@ubuntu:/home/topeet/Linux+QT# ls
root  root_20150123.tar.gz  system.img
root@ubuntu:/home/topeet/Linux+QT#
```

最后 Qtopia2.2.0 系统还需要一个镜像文件 “ramdisk-uboot.img” ，这个镜像文件是通用的，可以直接用编译好的镜像。这个镜像在用户光盘 “04_镜像_QT 文件系统” → “system” 中，如下图所示。



A screenshot of a file manager interface showing two files: "ramdisk-uboot.img" and "system.img". Each file has a small icon to its left, consisting of a blue square and a grey square. The files are listed vertically, with "ramdisk-uboot.img" on top and "system.img" below it.

那么到这一步，Qtopia2.2.0 需要的全部镜像就都已经制作完成。