龙芯内核交叉编译及安装

吴松伟
2012-7-26

本文主要讲述如何在 x86 平台上编译龙芯的内核，以及如何进行安装，包括 U 盘安装和光盘安装。

一、环境

x86 环境系统为 ubuntu 12.04，64 位。

龙芯环境为 loongson 3A。

二、编译

因为是要交叉编译，所以不能用系统里自带的 gcc 等工具，需要重新编译。这里使用 gcc 4.6.0，需要准备以下源码：

1.binutils-2.21.1.tar.bz2
2.gcc-4.6.0.tar.bz2

交叉编译主要是要得到供编译内核所使用的库及 gcc 工具，而所谓交叉则需要指定 host 及 target，因此需要在 configure 时指定这三项。

2.1, 编译 binutils

主要命令如下，脚本详见附录１：

先建一个目录用于安装的目的地，我的为/home/louis/toolchain/gcc4.6/install，注意它只能绝对路径。

```
cd build-binutils
./configure --prefix=/home/louis/toolchain/gcc4.6/install       \
--build=x86_64-linux-gnu  --host=x86_64-linux-gnu --target=mips64el-linux  \
--enable-shared -v
make
make install
```

2.2,编译 gcc 4.6.0

直接编译 gcc 会报以下错误：

gcc configure: error: Building GCC requires GMP 4.2+, MPFR 2.3.1+ and MPC 0.8.0+，所以要首先安装它们，可参考「１」解决，主要步骤如下：

在 ftp://gcc.gnu.org/pub/gcc/infrastructure/下载 gmp,mpfr,mpc 的源代码。

「注」这里提供的 gmp 等都不是最新的，实验时将 gmp 换成了 gmp-5.0.1，下载地址为：

ftp://ftp.gnu.org/gnu/gmp/gmp-5.0.1.tar.bz2，其他两项觉得麻烦，没有采用最新的，版本为 mpfr-2.4.2 和 mpc-0.8.1。

由于依赖关系，需要先安装 gmp，再安装 mpfr，最后安装 mpc。

安装是典型的三步曲，configure，make 和 make install，这里强烈建议再加上 make check。

「注」--prefix 的作用是指定安装的目录。

```
cd ./gmp-5.0.1/
./configure --prefix=/usr/local/gmp-5.0.1
make
make check
sudo make install
cd ..
```

这一步过程中可能会报与 M4 相关的错误，只用 sudo apt-get install m4 安装即可。

mpfr 和 mpc 的安装方法与 gmp 类似。不过要注意配置的时候要把 gmp 与 mpfr 的依赖关系选项加进去，具体配置命令分别如下：

```
./configure –prefix=/usr/local/mpfr-2.4.2 --with-gmp=/usr/local/gmp-5.0.1
./configure –prefix=/usr/local/mpc-0.8.1 --with-gmp=/usr/local/gmp-5.0.1 –with-mpfr=/usr/local/mpfr-2.4.2
```

安装好这三个库之后，就可以正式开始安装 gcc 了。

当然了链接的时候，需要刚刚编译的 3 个 lib。

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/mpc-0.8.1/lib:/usr/local/gmp-5.0.1/lib:/usr/local/mpfr-2.4.2/lib
```

如果不想每打开一个终端都设置一次的话，可以将它加入到~/.bashrc 文件中。

之后就可以编译 gcc 了，与编译 binutils 一样，需要指定 host,target 以及最终安装路径。

```
cd ./gcc-4.6.0
./configure –prefix=/home/louis/toolchain/gcc4.6/install \
--build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=mips64el-linux \
--disable-shared --enable-languages=c --disable-threads \
-v --with-gmp=/usr/local/gmp-5.0.1 –with-mpfr=/usr/local/mpfr-2.4.2 –with-mpc=/usr/local/mpc-0.8.1
make all-gcc
make install-gcc
```

gcc 的编译也可以写一个脚本来完成，与附录 1 类似。

2.3 编译内核与模块

下载地址：http://www.loongson.cn/dev/wiki/%E9%A6%96%E9%A1%B5

我下载的是单路内核，压缩包名为 kernel36-loongson320120606.tar.gz。解压后名字为 linux-loongson-all，发现其实是已经编译好的。没有关系，我们还是可以重新编译。

这一步我们需要得到两样东西，内核和驱动模块。这一节主要讲述如何编译内核，编译模块，最终取得内核与模块。不过从龙芯官网的安装说明及其提供的文件系统来看，它似乎未提供模块。下载的内核是已经编译过的，我们将其重新编译以了解整个完整流程。

2.3.1 编译

首先要保证 LD_LIBRARY_PATH 已经将 gmp,mpfr 以及 mpc 库的路径加载进去，没有的话，可以参考 2.2 节。

编过内核的都知道，编译的第一步是通过 make menuconfig 去选择要编译到内核中的项，生成.config 文件，由于这里的内核是已经编译过的，所以这一步可以省去。

开始编译内核及模块，由于我的机器是 4 核的，所以这里-j 参数为 4。CROSS_COMPILE 的路径即为前面编译的 gcc 的路径，注意 CROSS_COMPILE 只是前缀，并不是某个文件完整路径。

```
cd linux-loongson-all
make -j 4 ARCH=mips CROSS_COMPILE=/home/louis/toolchain/gcc4.6/install/bin/mips64el-linux-
```

编译完内核后一般需要编译模块，不过此源代码在编译模块时会出错，原因未深究。按照龙芯官网的安装说明来看，似乎它也没有提供模块。另外一个内核版本为 2.6.36-11.lemote 的内核则可以编译出内核模块，需要的可以和我联系。编译方法为：

```
make modules -j 4 ARCH=mips CROSS_COMPILE=/home/louis/toolchain/gcc4.6/install/bin/mips64el-linux-
```

2.3.2 收集内核模块

我们知道，内核模块在安装的时候会在/lib/modules 下生成一个与当前内核版本同名的文件夹，我们需要这个文件夹，但是又不想将这个内核安装到自己的机器中，怎么办呢？有两种方法可以解决。

1，在 make modules_install 时指定参数 INSTALL_MOD_PATH，比如：

```
mkdir /home/louis/modules_install
make modules_install INSTALL_MOD_PATH=/home/louis/modules_install ARCH=mips CROSS_COMPILE=/home/louis/toolchain/gcc4.6/install/bin/mips64el-linux-
```

以上命令则会将内核模块安装到/home/louis/modules_install 目录中。

2，用 chroot 命令来解决。

chroot 的作用是将指定的目录当成文件系统的根目录，所以的操作都不会超出此目录，也正是由于这个原因，我们需要将常用工具及库拷贝到此文件夹下，因为 chroot 后，已经看不到真实的根目录，也就找不到相应的命令了。

创建一个新目录，假设叫 fakeroot，将/bin, /sbin/, /usr/bin,/lib, /lib64 以及编译好的内核源码等目录拷过来,cd ./fakeroot/

```
sudo chroot .
cd linux-loongson-all
make modules_install
```

这种方法会将内核安装到/lib/modules 下，由于这里已经使用了 chroot，所以会安装到./fakeroot/lib/modules 下。

这种方法比较繁琐，掌握它有一个用处是用于定制 ISO 镜像，将诸如 mysql 等常用服务或者我们自己写的一些需要服务安装到此文件系统中。

「注」从龙芯官网的安装说明及其提供的文件系统来看，它是不带这些内核模块的。所以如果内核模块编译失败，则此步可以省略。

三、安装

这里主要讲 U 盘安装与光盘安装，假设 BIOS 已经装好。

3.1 制作 U 盘安装

3.1.1 安装过程说明

主要参考 http://www.loongson.cn/dev/wiki/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F%E5%AE%89%E8%A3%85%E8%AF%B4%E6%98%8E

从网页说明中可以看到，需要准备安装目录，主要包括两个文件夹，boot 和 package，这两个文件夹都可以从网页中获取。可以看到，package 只有内核，没有文件系统，我们还需要准备文件系统。文件系统也可以在网页中下载，我是从 http://www.loongson.cn/dev/wiki/Rehl6-64%E7%89%88%E6%9C%AC%E4%B8%8B%E8%BD%BD%E5%9C%B0%E5%9D%80 下载的 rhel6-mips-64-20120502-Alpha.tar.gz。

先整个安装过程大致流程如下：

1，开机后运行 BIOS

2，BIOS 会解析 boot/boot.cfg，将各种选择项显示到屏幕上。

3，根据选择，比如做了以下选择：

```
title 3A780E-1-way-usb-ext2/3-install
       kernel /dev/fs/ext2@usb0/boot/vmlinux
       initrd /dev/fs/ext2@usb0/boot/initrd.img.gz
       args console=tty rdinit=/sbin/init usb_install
```

系统则会以 boot 目录下的 vmlinux 作为内核启动，将 initrd.img.gz 解压后的内容加载到内存中作为文件系统，然后运行/sbin/init 程序。

4，/sbin/init 程序会引导进行磁盘分区等操作，然后将 package 中的内核安装到文件系统中，并将文件系统拷贝到磁盘分区上。

我们知道，系统要启动，内核与内核模块是必不可少的，这里下载的文件系统中并没有内核模块，前面我们准备好的内核模块就有用了。

3.1.2 制作安装 U 盘

1，准备好一个 U 盘，并格式化为 ext2 或者 ext3 格式。

2，下载 boot 及 package 目录，拷贝到 U 盘中。用前面我们自己编的内核替换 package/kernel/kernel-3A780E-1，替换后名称依然不变，为 kernel-3A780E-1。选单路还是双路，根据自己系统而定。

3，下载 rhel6-mips-64-20120502-Alpha.tar.gz，解压后将 boot 文件清空（boot 目录中的内容是否清空对系统启动没有影响，因为安装过程会将 vmlinux 等覆盖，只是此目录中的所有文件都无用，所以我将其清空）。如果能编译出内核模块的话，将前面准备好的存储内核模块的文件夹拷贝到 lib/modules/下。

4，重新压缩，重命名为 loongson.tar.gz，并将它拷贝到 package 下。

此后就可以用此 U 盘来安装系统了，安装过程见网页中教材。可见此处我们解压的目的是为了将内核模块拷贝过去，如果无内核模块，则直接将 rhel6-mips-64-20120502-Alpha.tar.gz 重命名为 loongson.tar.gz 即可。当然，如果我们要定制安装某些服务，则可以解压后，利用前面讲到的 chroot 命令，去将服务安装到此文件系统中。前提是你的系统与此文件系统属于相同的系统，而如果当前系统是 ubuntu，而下载的文件系统是 redhat6.0 的，安装后可能会存在问题。

3.2 制作安装光盘

1，按照制作 U 盘安装盘的步骤，将 boot 及 package 目录准备好，包括文件系统。

2，利用以下命令烧写光盘：

比如有目录 install_package，里面包含了已经准备好的 boot 和 package 目录，

mkisofs -relaxed-filenames -allow-lowercase -graft-points -allow-multidot -pad -r -l -J -d -v -V "cd_install" -hide-rr-moved -o rhel6-mips-64-20120502-Alpha.iso ./install_package

再在 ubuntu 上右键点击 rhel6-mips-64-20120502-Alpha.iso，利用 Brasero 记录到光盘中即可。

小结：

从整个过程来看，这里算不上完整意义上的安装，它的主要思路是先将 rhel6 安装到某台电脑中，从而得到我们所需要的文件系统，只需要将此文件系统拷贝到磁盘中，并将为 mips 平台编译的内核及驱动模块存放到相应的路径即可。

按照这相思路，完全有可能自己制作一个 UBUNTU 等安装系统。

附录 1：

```
#!/bin/sh

cross_host="x86_64-linux-gnu"
cross_target="mips64el-linux"
dell_install_path="/home/louis/code/toolchain/gcc4.6/install"

#mkdir -pv ./install

# build binutils-2.21

if [ ! -e ./binutils-2.21.1 ]; then
```

```
    echo "tar xf binutils-2.21.1a.tar.bz2, please waiting..."
    echo
    tar xf binutils-2.21.1a.tar.bz2
fi

if [ -e ./build-binutils ]; then
    rm -rf ./build-binutils
fi

mkdir -pv ./build-binutils
cd build-binutils
../binutils-2.21.1/configure --prefix=$dell_install_path        \
--build=$cross_host --host=$cross_host --target=$cross_target  \
--enable-shared -v
make
make install
cd ../

echo "End of building binutils toolchain"
```

参考目录：

[1] http://blog.csdn.net/demon__hunter/article/details/6419787

[2] http://www.loongson.cn/dev/wiki/Loongson_Kernel%E7%BC%96%E8%AF%91%E4%B8%8E%E4%BD%BF%E7%94%A8

[3] http://www.loongson.cn/dev/wiki/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F%E5%AE%89%E8%A3%85%E8%AF%B4%E6%98%8E