# Ingenic GCC Cross-Compiler Tool Chain Handbook

# 1 Installation and usage of Ingenic GCC cross-compile

This section describes the installation and use of Ingenic GCC cross compiler tool chain.

In order to offer convenient for users to develop based on Linux, the following GCC cross-compilers were provided:

Linux 2.6 system:
• GCC-4.1.2 + GLIBC-2.3.6

Linux 2.4 system:
• GCC-4.1.2 + GLIBC-2.3.2
• GCC-3.3.1 + GLIBC-2.3.2

These cross-compilers running on host environment of Linux, are used to generate the code which can run on a 32-bit Ingenic processor.

Users can choose compiler's version according to the following

1. Compile U-Boot-1.1.6: optional GCC-4.1.2 or GCC-3.3.1
2. Compile Linux 2.4 kernel: optional GCC-3.3.1
3. Compile Linux 2.6 kernel: optional GCC-4.1.2
4. Compile busybox and applications: choice GCC-4.1.2 or GCC-3.3.1

Note that the users select a particular version of compiler; make sure the same version between GLIBC of applications depends on the root file system and compiler. For Ingenic Linux systems, Linux 2.4 system uses glibc 2.3.2 Dynamic Library, Linux 2.6 system uses glibc 2.3.6 dynamic library.

You can Download the following files from Website (http://www.ingenic.cn):

• mipseltools-gcc412-lnx26.tar.gz: Linux 2.6 system compiler, Linux host version
• mipseltools-gcc412-lnx24.tar.gz: Linux 2.4 system compiler, Linux host version
• mipseltools-gcc412-lnx24-cygwin.tar.gz: Linux 2.4 system compiler, Windows Cygwin version
• mipseltools-gcc331-lnx24.tar.gz: Linux 2.4 system compiler, Linux host version

• mipseltools-gcc331-lnx24-cygwin.tar.gz: Linux 2.4 system compiler, Windows Cygwin version

Users should use Linux host as a development environment which is recommended, use the GCC-3.3.1 compiler based on the Linux 2.4 kernel, use the GCC-4.1.2 compiler based on the Linux 2.6 kernel

Install Linux 2.6 compiler, the following steps:

1, install the GCC compiler to the working directory, for example as follows:

```
# cd /opt
# tar xzf mipseltools-gcc412-lnx26.tar.gz
```

2, set the GCC compiler's path:

```
# export PATH = / opt/mipseltools-gcc412-lnx26/bin: $ PATH
```

Install Linux 2.4 compiler, the following steps:

1, the GCC compiler is installed to the working directory, for example as follows:

```
# cd / opt
# tar xzf mipseltools-gcc331-lnx24.tar.gz
```

2, set the GCC compiler path:

```
#export PATH = /opt/mipseltools-gcc331-lnx24/bin: $PATH
```

Then the compiler has installed. MIPS cross-compiler prefix is "mipsel-linux-", now you can use the mipsel-linux-gcc and mipsel-linux-g++ to compile the procedure.

The syntax of Ingenic GCC cross compiler tool chain isompatible with the GNU GCC, please refer to the GNU GCC's user manual.

The following are examples of helloworld.c:

```
# include <stdio.h>
int main (void)
(
printf ( "Hello world! \ n");
return 0;
)
```

Compiler generates helloworld:

```
# mipsel-linux-gcc -O2 -o helloworld helloworld.c
```

If you can correctly build helloworld, shows the compiler has been installed properly you can normally use.

# 2 Ingenic Linux 2.6 compiler

Ingenic Linux 2.6 compiler uses GCC 4.1.2 and GLIBC 2.3.6, now introdues how to produce the source package which is needed by compiler and steps .

Souce package needed:

- linux-2.6.24.3 headers
- binutils-2.17
- gcc-4.1.2
- glibc-2.3.6
- glibc-ports-2.3.6
- glibc-linuxthreads-2.3.6
- gdb-6.0
- e2fsprogs-1.40.4
- zlib-1.2.3
- jpeg-6b
- lcms_1.16
- libpng-1.2.24
- libmng-1.0.10

you can get above pachage and patch from the blow website.

ftp://ftp.ingenic.cn/3sw/01linux/00toolchain/jz-gcc412-glib236-src.tar.gz

there are four steps to produce cross-compiler:

- Compile binutils
- Compile gcc of bootstrap，generate glibc tool.
- Compile glibc，need to specify kernel header files.
- Compile the complete gcc 和 g++.

Next, produce the script.

```
#!/bin/sh

############################################################
# Modify following variables to yours
TARBALL_PATH=/home/jlwei/work-gcc/tarball
PATCHES_PATH=/home/jlwei/work-gcc/patches
LINUX_HEADERS_PATH=/home/jlwei/work-gcc/linux-headers-2.6.24.3
INSTALL_PATH=/opt/mipseltools-gcc412-lnx26
```

```
BUILD_PATH=`pwd`
#############################################################

BINUTILS_VER=binutils-2.17
GCC_VER=gcc-4.1.2
GLIBC_VER=glibc-2.3.6
GLIBC_PORTS_VER=glibc-ports-2.3.6
GLIBC_LINUXTHREADS_VER=glibc-linuxthreads-2.3.6
GDB_VER=gdb-6.0

unset CFLAGS
unset CXXFLAGS

export PATH=${INSTALL_PATH}/bin:$PATH

echo "----------------------------------"
echo "@@ Building binutils ..."
echo "----------------------------------"

# prepare binutils source
cd ${BUILD_PATH}
rm -rf ${BINUTILS_VER} binutils-build
tar jxf ${TARBALL_PATH}/${BINUTILS_VER}.tar.bz2

# configure and build binutils
mkdir -v binutils-build
cd binutils-build

../${BINUTILS_VER}/configure --target=mipsel-linux
--prefix=${INSTALL_PATH}
make CFLAGS="-O2"

# install binutils
make install

echo "----------------------------------"
echo "@@ Building bootstrap gcc ..."
echo "----------------------------------"

# prepare gcc source
cd ${BUILD_PATH}
rm -rf ${GCC_VER} gcc-build

tar jxf ${TARBALL_PATH}/${GCC_VER}.tar.bz2
```

```
cd ${BUILD_PATH}/${GCC_VER}/libiberty
cat strsignal.c | sed -e 's/#ifndef HAVE_PSIGNAL/#if 0/g' >junk.c
cp -f strsignal.c strsignal.c.fixed; mv -f junk.c strsignal.c

# configure and build gcc
cd ${BUILD_PATH}
mkdir -v gcc-build
cd gcc-build

../${GCC_VER}/configure --target=mipsel-linux \
      --host=i686-pc-linux-gnu --prefix=${INSTALL_PATH} \
      --disable-shared --disable-threads --disable-multilib \
      --enable-languages=c

make CFLAGS="-O2" all-gcc

# install gcc
make install-gcc

echo "----------------------------------"
echo "@@ Building glibc ..."
echo "----------------------------------"

# prepare glibc source
cd ${BUILD_PATH}
rm -rf ${GLIBC_VER} glibc-build

tar jxf ${TARBALL_PATH}/${GLIBC_VER}.tar.bz2
cd ${GLIBC_VER}
tar jxf ${TARBALL_PATH}/${GLIBC_LINUXTHREADS_VER}.tar.bz2
tar jxf ${TARBALL_PATH}/${GLIBC_PORTS_VER}.tar.bz2
mv ${GLIBC_PORTS_VER} ports
# apply patch
patch -Np1 -i ${PATCHES_PATH}/${GLIBC_VER}-jz.patch
# remove nptl
rm -rf nptl nptl_db

# configure and build glibc
cd ${BUILD_PATH}
mkdir -v glibc-build
cd glibc-build

export CC="mipsel-linux-gcc"
export libc_cv_forced_unwind=yes
```

```
export libc_cv_c_cleanup=yes

../${GLIBC_VER}/configure \
      --host=mips-linux \
      --build=i686-pc-linux-gnu \
      --enable-add-ons \
      --enable-shared \
      --with-cpu=mips32 \
      --prefix=/usr \
      --with-headers=${LINUX_HEADERS_PATH}

make CFLAGS="-O2"

# install glibc
export GLIBC_INSTALL=${BUILD_PATH}/glibc-inst

cd $BUILD_PATH
rm -rf glibc-inst
mkdir -v glibc-inst
cd glibc-build

make install_root=${GLIBC_INSTALL} install

cd ${GLIBC_INSTALL}; tar zcf $BUILD_PATH/glibc-build/glibc-lib.tgz lib
cd ${INSTALL_PATH}; tar zxf $BUILD_PATH/glibc-build/glibc-lib.tgz
cd ${GLIBC_INSTALL}/usr; tar cfz
$BUILD_PATH/glibc-build/glibc-usr.tgz .
cd ${INSTALL_PATH}/mipsel-linux; tar xzf
$BUILD_PATH/glibc-build/glibc-usr.tgz
tar xfz $BUILD_PATH/glibc-build/glibc-lib.tgz

# install linux kernel headers
cd ${LINUX_HEADERS_PATH}
cp -afr {asm,asm-mips,asm-generic,linux,mtd,scsi,sound}
${INSTALL_PATH}/mipsel-linux/include

# fixed libc.so and libpthread.so
sed -i -e 's/\/usr\/lib\///g'
${INSTALL_PATH}/mipsel-linux/lib/libpthread.so
sed -i -e 's/\/usr\/lib\///g' ${INSTALL_PATH}/mipsel-linux/lib/libc.so

sed -i -e 's/\/lib\///g'
${INSTALL_PATH}/mipsel-linux/lib/libpthread.so
sed -i -e 's/\/lib\///g' ${INSTALL_PATH}/mipsel-linux/lib/libc.so
```

```
# install localedata
cd ${BUILD_PATH}/glibc-build
cp -v ${PATCHES_PATH}/${GLIBC_VER}-localedata-Makefile
${BUILD_PATH}/${GLIBC_VER}/localedata/Makefile
cp -v ${PATCHES_PATH}/${GLIBC_VER}-localedata-SUPPORTED
${BUILD_PATH}/${GLIBC_VER}/localedata/SUPPORTED
make localedata/install-locales install_root=${INSTALL_PATH}

echo "---------------------------------"
echo "@@ Building final gcc ..."
echo "---------------------------------"

# prepare gcc source
cd $BUILD_PATH
rm -rf ${GCC_VER} gcc-build

tar jxf ${TARBALL_PATH}/${GCC_VER}.tar.bz2
cd ${BUILD_PATH}/${GCC_VER}/libiberty
cat strsignal.c | sed -e 's/#ifndef HAVE_PSIGNAL/#if 0/g' >junk.c
cp -f strsignal.c strsignal.c.fixed; mv -f junk.c strsignal.c

# apply patches
cd ${BUILD_PATH}/${GCC_VER}
patch -p0 < ${PATCHES_PATH}/gcc-4.1_bug27067.patch

# configure and build gcc
cd ${BUILD_PATH}
mkdir -v gcc-build
cd gcc-build

export CC="gcc"

../${GCC_VER}/configure --target=mipsel-linux \
      --host=i686-pc-linux-gnu --prefix=${INSTALL_PATH} \
      --disable-multilib --enable-shared --enable-languages=c,c++ \
      --with-headers=${INSTALL_PATH}/mipsel-linux/include

make CFLAGS="-O2"

# install gcc
make install

# remove sys-include
```

```
rm -rf ${INSTALL_PATH}/mipsel-linux/sys-include

cd $BUILD_PATH

# fix symlink
echo "Fix symlink ..."

cd ${INSTALL_PATH}/mipsel-linux/lib

rm libanl.so
ln -s libanl.so.1 libanl.so

rm libBrokenLocale.so
ln -s libBrokenLocale.so.1 libBrokenLocale.so

rm libcrypt.so
ln -s libcrypt.so.1 libcrypt.so

rm libdl.so
ln -s libdl.so.2 libdl.so

rm libm.so
ln -s libm.so.6 libm.so

rm libnsl.so
ln -s libnsl.so.1 libnsl.so

rm libnss_compat.so
ln -s libnss_compat.so.2 libnss_compat.so

rm libnss_dns.so
ln -s libnss_dns.so.2 libnss_dns.so

rm libnss_files.so
ln -s libnss_files.so.2 libnss_files.so

rm libnss_hesiod.so
ln -s libnss_hesiod.so.2 libnss_hesiod.so

rm libnss_nisplus.so
ln -s libnss_nisplus.so.2 libnss_nisplus.so

rm libnss_nis.so
ln -s libnss_nis.so.2 libnss_nis.so
```

```
rm libresolv.so
ln -s libresolv.so.2 libresolv.so

rm librt.so
ln -s librt.so.1 librt.so

rm libthread_db.so
ln -s libthread_db.so.1 libthread_db.so

rm libutil.so
ln -s libutil.so.1 libutil.so

# install mxu_as and jz_mxu.h
cp -v ${PATCHES_PATH}/mxu_as ${INSTALL_PATH}/bin
chmod +x ${INSTALL_PATH}/bin/mxu_as
cp -v ${PATCHES_PATH}/jz_mxu.h ${INSTALL_PATH}/mipsel-linux/include

echo "-----------------------------------"
echo "@@ Building GDB ..."
echo "-----------------------------------"

cd $BUILD_PATH
rm -rf ${GDB_VER} gdb-build

tar jxf ${TARBALL_PATH}/${GDB_VER}.tar.bz2
mkdir -v gdb-build
cd gdb-build

export CC="gcc"

../${GDB_VER}/configure --target=mipsel-linux
--prefix=${INSTALL_PATH}
make
make install


echo "@@ Building binutils, glibc, gcc and gdb OK"
```

The below is a script to compile dynamic library from the third party：

```
#!/bin/sh

#############################################################################
# Modify these variables to yours
THIRDPARTY_PATH=/home/jlwei/work-gcc/thirdparty
INSTALL_PATH=/opt/mipseltools-gcc412-lnx26/mipsel-linux
BUILDDIR=`pwd`
export PATH=/opt/mipseltools-gcc412-lnx26/bin:$PATH # mipsel-linux-gcc
#############################################################################


echo "@@ Building e2fsprogs ..."

cd ${BUILDDIR}
tar xzf ${THIRDPARTY_PATH}/e2fsprogs-1.40.4.tar.gz
cd e2fsprogs-1.40.4

./configure --prefix=${INSTALL_PATH} --host=mipsel-linux --enable-elf-shlibs
--disable-tls CC=mipsel-linux-gcc LD=mipsel-linux-ld
make
make install-libs

# fix symbol links
cd ${INSTALL_PATH}/lib

rm libblkid.so
ln -s libblkid.so.1 libblkid.so

rm libcom_err.so
ln -s libcom_err.so.2 libcom_err.so

rm libe2p.so
ln -s libe2p.so.2 libe2p.so

rm libext2fs.so
ln -s libext2fs.so.2 libext2fs.so

rm libss.so
ln -s libss.so.2 libss.so

rm libuuid.so
```

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


echo "@@ Building zlib ..."

cd ${BUILDDIR}
tar xzf ${THIRDPARTY_PATH}/zlib-1.2.3.tar.gz
cd zlib-1.2.3

CC="mipsel-linux-gcc" AR="mipsel-linux-ar cr"
RANLIB="mipsel-linux-ranlib" ./configure --prefix=${INSTALL_PATH}
make
make install

CC="mipsel-linux-gcc" AR="mipsel-linux-ar cr"
RANLIB="mipsel-linux-ranlib" ./configure --shared --prefix=${INSTALL_PATH}
make
make install


echo "@@ Building libjpeg ..."

cd ${BUILDDIR}
tar xzf ${THIRDPARTY_PATH}/jpegsrc.v6b.tar.gz
cd jpeg-6b

CC=gcc ./configure --prefix=${INSTALL_PATH} --enable-shared --enable-static

sed -i -e 's/CC= gcc/CC= mipsel-linux-gcc/g' Makefile
sed -i -e 's/AR= ar rc/AR= mipsel-linux-ar rc/g' Makefile
sed -i -e 's/AR2= ranlib/AR2= mipsel-linux-ranlib/g' Makefile

make
mkdir -p ${INSTALL_PATH}/man/man1
make install


echo "@@ Building lcms ..."

cd ${BUILDDIR}
tar xzf ${THIRDPARTY_PATH}/lcms_1.16.tar.gz
cd lcms-1.16
```

```
./configure --prefix=${INSTALL_PATH} --target=mipsel-linux --host=mipsel-linux
CC=mipsel-linux-gcc
make
make install


echo "@@ Building libpng ..."

cd ${BUILDDIR}
tar xzf ${THIRDPARTY_PATH}/libpng-1.2.24.tar.gz
cd libpng-1.2.24
./configure --prefix=${INSTALL_PATH} --host=mipsel-linux CC=mipsel-linux-gcc
CFLAGS="-I${INSTALL_PATH}/include" LDFLAGS="-L${INSTALL_PATH}/lib"
make
make install


echo "@@ Building libmng ..."

cd ${BUILDDIR}
tar xzf ${THIRDPARTY_PATH}/libmng-1.0.10.tar.gz
cd libmng-1.0.10
cp makefiles/makefile.linux makefile

# replace gcc to mipsel-linux-gcc
sed -i -e 's/CC=gcc/CC=mipsel-linux-gcc/g' makefile

# replace ar to mipsel-linux-ar
sed -i -e 's/ar rc/mipsel-linux-ar rc/g' makefile

# replace ranlib to mipsel-linux-ranlib
sed -i -e 's/RANLIB=ranlib/RANLIB=mipsel-linux-ranlib/g' makefile

# replace /usr/local to ${INSTALL_PATH}
sed -i -e 's/\/usr\/local/\/opt\/mipseltools-gcc412-lnx26\/mipsel-linux/g' makefile

make
make install


echo "@@ Building tslib ..."

cd ${BUILDDIR}
tar xzf ${THIRDPARTY_PATH}/tslib-jz.tar.gz
```

```
cd tslib-jz/tslib-0.1.1
./autogen.sh
echo "ac_cv_func_malloc_0_nonnull=yes" > config.cache
./configure --prefix=${INSTALL_PATH} --host=mipsel-linux --cache-file=config.cache

make
make install


echo "@@ Building alsa ..."

cd ${BUILDDIR}
tar xzf ${THIRDPARTY_PATH}/alsa-tools.tar.gz
cd alsa-tools/alsa-lib-1.0.15
./configure --prefix=${INSTALL_PATH} AR=mipsel-linux-ar CC=mipsel-linux-gcc
CXX=mipsel-linux-g++ CXX=mipsel-linux-g++ --host=mipsel-linux --enable-shared=yes
--enable-static=no --target=mips-linux --with-debug=no --with-alsa-devdir=/dev
--with-softfloat  LDFLAGS="-lm"

make
make install

echo "Build thirdparty libs done."
```

## Network resource：

http://www.gnu.org/：GNU homepage
http://ftp.gnu.org/gnu/binutils/：binutils download website
http://gcc.gnu.org/：GCC homepage
http://ftp.gnu.org/gnu/glibc/：glibc download website
http://e2fsprogs.sourceforge.net/：e2fsprogs homepage
http://www.zlib.net/：zlib homepage
http://www.ijg.org/：libjpeg homepage
http://www.littlecms.com/downloads.htm：lcms download website
http://www.libpng.org/：libpng homepage
http://www.libmng.com/：libmng homepage