Crosscompiling Linux/m68k from the scratch

a quick'n'dirty non-standard mini-howto

Frank Meurer

February 2000, Ver. 0.0.2

1 Preface

2000-01-02
Ok, here we go. You have a fine installed i386 linux with your gcc located in /usr/bin? You want to compile software for m68k-linux on your i386-linux?
So let's build a crosscompiler.

Changes of this document:
0.0.2 page layout altered

2 Prework

It's very easy to use the GNU-tools: the gcc and the GNU binutils. The binutils are UTILS to generate BINaries, i.e. GNU as (assembler), GNU ld (linker).
Our native i386 gcc and its files are under /usr (called "prefix"). We will install our cross-compiler, binutils and more under /usr/local. It's easy to do so because many Makefiles and configure-scripts use /usr/local as default prefix, you can install libraries there not caring about your main libs and at least it offers us a simple way to switch between the "normal" gcc in /usr/bin and the cross-gcc (which will be installed in /usr/local/bin) by modifying the environment variable PATH.
I strongly recommend to use not the root-account. Please change with root-permissions the ownerships under /usr/local:

```
chown <your-private-login-name> /usr/local /usr/local/*
```

Now you should login with as *your-private-login-name* and do all following stuff with this account.

3 binutils

Let's start with the binutils. Get the source package for the GNU binutils (I used version 2.9.1) and unpack it (i.e. under /usr/local/src). If you don't know where to get the package or how to unpack it it's better for you to stop here! (Actually you can't continue if you don't get and unpack the package. ;-)
Go to the binutils-dir:

```
cd binutils-2.9.1 #or binutils-2.8 or binutils-2.?.?
```

Remove old files (don't worry about errors) and run the configure-script:

```
make distclean
./configure --prefix=/usr/local --enable-shared \
  --target=m68k-unknown-linux
```

If you get errors at this early stage you have IMHO (in my humble opinion) one or more of the following "misconfigurations":

1. Your gcc is not properly installed (use rpm- or deb-packages!).
2. Your linux is too new for the binutils.
3. Your binutils are too new for your linux.
4. You have a buggy binutils package.

Start the binutils compilation:

```
make
```

If you get errors then one or more of the above four reasons are possible. Make the info-pages:

```
make info
```

These pages are newer kinds of "man-pages". It's not necessary to build them but it's strongly recommended! Always remember: RTFM (Read The F...... Manual) So please make, install and read the docs!
Now you have the binutils and the docs compiled. Install them:

```
make install
```

You will (you should!) find the following new files:

```
/usr/local/bin/m68k-unknown-linux-size
/usr/local/bin/m68k-unknown-linux-objdump
/usr/local/bin/m68k-unknown-linux-ar
/usr/local/bin/m68k-unknown-linux-strings
/usr/local/bin/m68k-unknown-linux-ranlib
/usr/local/bin/m68k-unknown-linux-c++filt
/usr/local/bin/m68k-unknown-linux-objcopy
/usr/local/bin/m68k-unknown-linux-addr2line
/usr/local/bin/m68k-unknown-linux-nm
/usr/local/bin/m68k-unknown-linux-strip
/usr/local/bin/m68k-unknown-linux-as
/usr/local/bin/m68k-unknown-linux-gasp
/usr/local/bin/m68k-unknown-linux-ld
/usr/local/lib/libbfd-2.9.1.so.0.0.0
/usr/local/lib/libiberty.a
/usr/local/lib/libbfd.la
/usr/local/lib/libbfd.a
/usr/local/lib/libopcodes-2.9.1.so.0.0.0
/usr/local/lib/libopcodes.la
/usr/local/lib/libopcodes.a
/usr/local/include/bfd.h
/usr/local/include/ansidecl.h
/usr/local/include/bfdlink.h
/usr/local/m68k-unknown-linux/bin/nm
/usr/local/m68k-unknown-linux/bin/strip
/usr/local/m68k-unknown-linux/bin/ar
/usr/local/m68k-unknown-linux/bin/ranlib
/usr/local/m68k-unknown-linux/bin/as
/usr/local/m68k-unknown-linux/bin/ld
/usr/local/m68k-unknown-linux/lib/ldscripts/m68kelf.x
/usr/local/m68k-unknown-linux/lib/ldscripts/m68kelf.xbn
/usr/local/m68k-unknown-linux/lib/ldscripts/m68kelf.xn
/usr/local/m68k-unknown-linux/lib/ldscripts/m68kelf.xr
/usr/local/m68k-unknown-linux/lib/ldscripts/m68kelf.xs
/usr/local/m68k-unknown-linux/lib/ldscripts/m68kelf.xu
/usr/local/m68k-unknown-linux/lib/ldscripts/m68klinux.x
/usr/local/m68k-unknown-linux/lib/ldscripts/m68klinux.xbn
/usr/local/m68k-unknown-linux/lib/ldscripts/m68klinux.xn
/usr/local/m68k-unknown-linux/lib/ldscripts/m68klinux.xr
/usr/local/m68k-unknown-linux/lib/ldscripts/m68klinux.xu
/usr/local/man/man1/m68k-unknown-linux-ar.1
/usr/local/man/man1/m68k-unknown-linux-nlmconv.1
```

```
/usr/local/man/man1/m68k-unknown-linux-c++filt.1
/usr/local/man/man1/m68k-unknown-linux-nm.1
/usr/local/man/man1/m68k-unknown-linux-objdump.1
/usr/local/man/man1/m68k-unknown-linux-ranlib.1
/usr/local/man/man1/m68k-unknown-linux-size.1
/usr/local/man/man1/m68k-unknown-linux-strings.1
/usr/local/man/man1/m68k-unknown-linux-strip.1
/usr/local/man/man1/m68k-unknown-linux-objcopy.1
/usr/local/man/man1/m68k-unknown-linux-addr2line.1
/usr/local/man/man1/m68k-unknown-linux-as.1
/usr/local/man/man1/m68k-unknown-linux-ld.1
```

4 Kernel-headers

Now copy the standard include files and the kernel include files:

```
cp -af /usr/include \
  /usr/local/m68k-unknown-linux/
cp -af /usr/src/linux/include/linux/ \
  /usr/local/m68k-unknown-linux/include/
cp -af /usr/src/linux/include/asm-m68k/ \
  /usr/local/m68k-unknown-linux/include/asm
```

Be sure that you have valid kernel includes of a legal kernel for m68k!

5 Compiler

Run the configure-script for generating a correct Makefile:

```
make distclean
./configure --prefix=/usr/local --target=m68k-unknown-linux \
  --with-gnu-ld --with-gnu-as
```

The output should be something like *"Links are now set up to build a cross-compiler for m68k-unknown-linux..."* If the output is something like *"... native compiler ..."* then you're doing something wrong (⇒RTFM). Now try to build the compiler:

```
make
```

If you only want to make a C-crosscompiler (no C++ or Objective-C) then you can use:

```
make LANGUAGES=c
```

If you're running into trouble about "makeinfo" then you can use:

```
make MAKEINFO=/bin/true
```

If you're running into trouble while compiling ./libgcc2.c, then you've maybe lost (or forget to copy) header files. Try:

```
make LIBGCC2_INCLUDES=/usr/include
```

If you want to use "GNULIBC_1" aka libc5, then you can try this:

```
make "T_CFLAGS=-DUSE_GNULIBC_1"
```

Now compilation of the cross-compiler should be finished. Install it:

```
make install
```

Here's the list of the new files you should now find:

```
/usr/local/bin/m68k-unknown-linux-g++
/usr/local/bin/m68k-unknown-linux-c++
/usr/local/bin/protoize
/usr/local/bin/unprotoize
/usr/local/bin/m68k-unknown-linux-gcc
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/syslimits.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/objc/hash.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/objc/list.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/objc/sarray.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/objc/objc.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/objc/objc-api.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/objc/NXConstStr.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/objc/Object.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/objc/Protocol.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/objc/encoding.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/objc/typedstream.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/stdarg.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/stddef.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/varargs.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-alpha.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-h8300.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-i860.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-i960.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-mips.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-m88k.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-pa.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-pyr.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-sparc.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-clipper.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-spur.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/iso646.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/va-ppc.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/proto.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/math-68881.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/limits.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/README
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/float.h
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/cc1
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/cc1obj
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/cc1plus
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/crtbegin.o
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/crtbeginS.o
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/crtend.o
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/crtendS.o
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/specs
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/SYSCALLS.c.X
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/cpp
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/libgcc.a
/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/libobjc.a
/usr/local/m68k-unknown-linux/bin/gcc
/usr/local/man/man1/m68k-unknown-linux-g++.1
/usr/local/man/man1/m68k-unknown-linux-gcc.1
/usr/local/man/man1/cccp.1
/usr/local/info/cpp.info
/usr/local/info/cpp.info-1
/usr/local/info/cpp.info-2
/usr/local/info/cpp.info-3
/usr/local/info/gcc.info
/usr/local/info/gcc.info-1
/usr/local/info/gcc.info-10
/usr/local/info/gcc.info-11
/usr/local/info/gcc.info-12
```

```
/usr/local/info/gcc.info-13
/usr/local/info/gcc.info-14
/usr/local/info/gcc.info-15
/usr/local/info/gcc.info-16
/usr/local/info/gcc.info-17
/usr/local/info/gcc.info-18
/usr/local/info/gcc.info-19
/usr/local/info/gcc.info-2
/usr/local/info/gcc.info-20
/usr/local/info/gcc.info-21
/usr/local/info/gcc.info-22
/usr/local/info/gcc.info-23
/usr/local/info/gcc.info-24
/usr/local/info/gcc.info-25
/usr/local/info/gcc.info-3
/usr/local/info/gcc.info-4
/usr/local/info/gcc.info-5
/usr/local/info/gcc.info-6
/usr/local/info/gcc.info-7
/usr/local/info/gcc.info-8
/usr/local/info/gcc.info-9
```

Now there's one little difficulty left. The cross-compiler doesn't know about floating point values. Check it:

```
cat /usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/float.h
```

If the file's size is very small, maybe it's only containing one "error"-line like *"#error float.h values not known for cross-compiler"* then you should use a predfined or default float.h for m68k. Copy it to the right place:

```
cp /your-own-path-to-the-right-include/float.h \
  /usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include/float.h
```

You can try the float.h of your i386-linux (look at /usr/lib/ gcc-lib/ i?86-?-linux/ ?.?.?.?/ include/ float.h) and use *this* file. The file float.h on OpenBSD/m68k differs slightly: The LDBL_* defines are the same as the DBL_* ones. Don't know if the greater values of i386-linux will work fine on m68k. ;-(
Now check your PATH:

```
echo $PATH
```

If it doesn't contain "/usr/local/bin" or "/usr/local/m68k-unknown-linux/bin" (i.e. is only "/bin:/usr/bin"), then you should add a path to your new crosscompile-tools:

```
export PATH="/usr/local/bin:/usr/local/m68k-unknown-linux/bin:"$PATH
```

Maybe you need to alter your MANPATH too:

```
export MANPATH="/usr/local/man:"$MANPATH
```

6 libc

Your compiler, assembler and more tools are now successfully installed. You can now cross-compile by typing "gcc -b m68k-unknown-linux test.c". Compilation and assembling will perfectly run - but what's that?
"/usr/local/m68k-unknown-linux/bin/ld: cannot open crt1.o: No such file or directory"
Hmm... sounds like some object files are missing. Object files? Archives? Libraries? Libraries! Get the libc5- or gnulibc2-source (glibc2=libc6, glibc1=libc5), and then:

```
mkdir libc-5.4.46 # I used libc5!
cd libc-5.4.46
tar xzf /path-to-libc-source/libc-5.4.46.tar.gz
```

```
cd libc
./configure
```

The configure-script needs some input. Please satisfy the script with the correct answers you should know now (er... NYS=yes). Compare the generated "config.in" with my "config.in":

```
cat config.in
```

My "config.in":

```
STATIC_SHARED=
MAKE=make
SPEED=fast
HOST_ROOTDIR=/usr/local
HOST_BINDIR=/usr/local/m68k-unknown-linux/bin
TARGET_ROOTDIR=/usr/local
ifeq ($(ELF), true)
TARGET_MACHINE=m68k-unknown-linux
TARGET__MACHINE=m68k-linux
else
TARGET_MACHINE=m68k-unknown-linuxaout
TARGET__MACHINE=m68k-linuxaout
endif
TARGET_OS=linux
TARGET_ARCH=m68k
TARGET_ARCH_x86=
MALLOC=dl-malloc
OLD_GCC=true
GCCVERSION=2.7.2.3
NYS=true
NYSDIR=nys
MAKE=make NYS=true
GCC_ARCH_INC_DIR=/usr/local/lib/gcc-lib/m68k-unknown-linux/2.7.2.3/include
```

Stop! It seems that I've forgotten some important thing. Hmmm... er... we should take a look in the docs:

```
less README
less README.nys
```

Read it carefully and try to understand it. And now (you probably already know it):

```
make
make install
```

It should install the follwing files:

```
/usr/local/lib/libc.so.5.4.46
/usr/local/lib/libm.so.5.0.9
/usr/local/usr/include/.... many, many header-files
/usr/local/usr/lib/crt1.o
/usr/local/usr/lib/crtbegin.o
/usr/local/usr/lib/crtbeginS.o
/usr/local/usr/lib/crtend.o
/usr/local/usr/lib/crtendS.o
/usr/local/usr/lib/crti.o
/usr/local/usr/lib/crtn.o
/usr/local/usr/lib/libbsd.a
/usr/local/usr/lib/libc.a
/usr/local/usr/lib/libm.a
/usr/local/usr/lib/libg.a
```

```
/usr/local/usr/lib/libc_p.a
/usr/local/usr/lib/libgmon.a
/usr/local/usr/lib/gcrt1.o
/usr/local/usr/lib/libmcheck.a
```

Now move the object files to a better place:

```
mv /usr/local/usr/lib/* /usr/local/m68k-unknown-linux/lib/
```

Make a symlink:

```
cd /usr/local/lib/
ln -s libc.so.5.4.46 libc.so
```

You can test the cross-compiler very easy:

```
echo "int main() { exit(0); }" >dummy.c
gcc -b m68k-unknown-linux -Wall -o dummy dummy.c
file dummy
```

The output should be "dummy: ELF 32-bit MSB executable, Motorola 68000, version 1, not stripped". You can do some optimization:

```
gcc -b m68k-unknown-linux -Wall -O2 -fomit-frame-pointer \
  -o dummy dummy.c
```

Or some more for MC68030 and MC68881/68882:

```
gcc -b m68k-unknown-linux -Wall -O2 -fomit-frame-pointer \
  -m68881 -m68030 -o dummy dummy.c
```

Check the linker:

```
gcc -b m68k-unknown-linux -Wall -Xlinker --verbose \
  -o dummy dummy.c 2\>\&1|more
```

The output must *not* contain any path like /usr/lib or /lib.

Frank Meurer <meurer@gmx.STOP-UCE.de>