# Lab.1: Cross Toolchain for ARM Linux (Binutils, GCC)

## Purpose

Learn how to build a cross compiler from its source code.

## Steps

1. Download source codes from:
    1. http://ftp.tsukuba.wide.ad.jp/software/gcc/snapshots/7.3.0-RC-20180122/
    2. http://ftp.gnu.org/gnu/binutils/
    3. `ftp://ftp.gnu.org/gnu/glibc/glibc-2.27.tar.gz`
    4. `git clone git@github.com:raspberrypi/linux.git --depth 1`
2. Build Cross Binutils:
    1. `mkdir WORK`
    2. `cd WORK`
    3. `mv ../binutils-2.30.tar.gz ./`
    4. `tar -zxvf binutils-2.30.tar.gz`
    5. `mkdir build_binutils`
    6. `cd build_binutils`
    7. `../binutils-2.30/configure --prefix=$HOME/WORK/crossgcc1 --target=arm-linux-gnueabihf`
    8. `make`
    9. `make install`
3. Build a Bare-metal Cross Compiler:
    1. `sudo dnf install gmp-devel mpfr-devel libmpc-devel`
    2. `mv ../gcc-7.3.0-RC-20180122.tar.gz ./`
    3. `tar -zxvf gcc-7.3.0-RC-20180122.tar.gz`
    4. `mkdir build_gcc1`
    5. `cd build_gcc1`
    6. `export PATH=$PATH:$HOME/WORK/crossgcc1/bin`

    7.
       ```
       ../gcc-7.3.0-RC-20180122/configure --prefix=$HOME/WORK/crossgcc1 --target=arm-linux-gnueabihf --enable-languages=c --without-headers --disable-libmudflap --disable-libatomic --with-arch=armv6 --disable-shared --enable-static --disable-decimal-float --disable-libgomp --disable-libitm --disable-libquadmath --disable-libsanitizer --disable-libssp --disable-threads --with-float=hard --with-fpu=vfp --with-newlib
       ```
    8. `make -j 7`
    9. `make install`
4. Install kernel headers
    1. `cd linux`
    2. `make headers_install ARCH=arm INSTALL_HDR_PATH=$HOME/WORK/sysroot/usr`
5. Build EGLIBC
    1. `tar -zxvf glibc-2.27.tar.gz`
    2. `mkdir build_eglibc`
    3. `cd build_eglibc`

    4. `../glibc-2.27/configure --prefix=/usr --host=arm-linux-gnueabihf --target=arm-linux-gnueabihf --with-headers=$HOME/WORK/sysroot/usr/include --includedir=/usr/include --enable-add-ons --disable-multilib`
    5. `make -j 7`
    6. `make install install_root=$HOME/WORK/sysroot`

6. Build Cross Binutils
    1. `mkdir build_binutils2`
    2. `cd build_binutils2`
    3. `../binutils-2.30/configure --prefix=$HOME/WORK/crossgcc2 --target=arm-linux-gnueabihf --with-sysroot=$HOME/WORK/sysroot`
    4. `make`
    5. `make install`

7. `export PATH=$PATH:$HOME/WORK/crossgcc2/bin`
8. Build a Cross Compiler
    1. `mkdir build_gcc2`
    2. `cd build_gcc2`
    3. `../gcc-7.3.0-RC-20180122/configure --prefix=$HOME/WORK/crossgcc2 --target=arm-linux-gnueabihf --enable-languages=c --with-sysroot=$HOME/WORK/sysroot --with-arch=armv6 --with-fpu=vfp --with-float=hard --disable-libmudflap --enable-libgomp --disable-libssp --enable-libquadmath --enable-libquadmath-support --disable-libsanitizer --enable-lto --enable-threads=posix --enable-target-optspace --with-linker-hash-style=gnu --disable-nls --disable-multilib --enable-long-long`
    4. `make -j 7`
    5. `make install -j 7`

9. Test:
    1. `$HOME/WORK/crossgcc2/bin/arm-linux-gnueabihf-gcc test.c`

## Discussion

At first, I configured gcc with --enable-languages=c,c++. This made the make phase of gcc fail every time. So, I spent 2.5 days fixing this which is really frustrating. Eventually, I gave up c++ and the building process went smoothly.

After the tool chain was successfully built, I tested `arm-linux-gnueabihf-gcc test.c` and it gave the result: `fatal error: stdio.h: No such file and directory`. So, I consulted Professor Chen and found out that it was my `$PATH` which is misconfigured.

As of now, the cross compiler finally works.