

GCC Toolchain for EPOS

The following howto is actually an executable script. With a bit of luck, you can simply copy&past it into a shell. It illustrates how to compile the GCC toolchain for x86, but you can select virtually any GCC target as long as it is ELF.

□□□□□□

```
#/bin/sh # GCC Toolchain for EPOS on x86 (ia32) BINUTILS=2.20 GCC=4.4.4 GMP=4.3.2
MPFR=2.4.2 NEWLIB=1.18.0 TARGET=i686-elf PREFIX=/usr/local/ia32/gcc-$GCC
PROG_PREFIX=ia32- BUILD=/tmp CONTRIB=/usr/local/contrib/linux/development # Wget cd
$CONTRIB if [ ! -e binutils-$BINUTILS.tar.bz2 ]; then wget
ftp://ftp.gnu.org/gnu/binutils/binutils-$BINUTILS.tar.bz2 fi if [ ! -e gcc-$GCC.tar.bz2 ]; then wget
ftp://ftp.gnu.org/gnu/gcc/gcc-$GCC/gcc-$GCC.tar.bz2 fi if [ ! -e gmp-$GMP.tar.bz2 ]; then wget
ftp://gcc.gnu.org/pub/gcc/infrastructure/gmp-$GMP.tar.bz2 fi if [ ! -e mpfr-$MPFR.tar.bz2 ]; then wget
ftp://gcc.gnu.org/pub/gcc/infrastructure/mpfr-$MPFR.tar.bz2 fi if [ ! -e gcc-g++-$GCC.tar.bz2 ]; then
wget ftp://ftp.gnu.org/gnu/gcc/gcc-$GCC/gcc-g++-$GCC.tar.bz2 fi if [ ! -e newlib-$NEWLIB.tar.gz ];
then wget ftp://sources.redhat.com/pub/newlib/newlib-$NEWLIB.tar.gz fi # Unpack cd $BUILD tar jxf
$CONTRIB/binutils-$BINUTILS.tar.bz2 tar jxf $CONTRIB/gcc-$GCC.tar.bz2 cd gcc-$GCC tar jxf
$CONTRIB/gmp-$GMP.tar.bz2 ln -s gmp-$GMP gmp tar jxf $CONTRIB/mpfr-$MPFR.tar.bz2 ln -s mpfr-
$MPFR mpfr cd .. tar jxf $CONTRIB/gcc-g++-$GCC.tar.bz2 tar zxf $CONTRIB/newlib-$NEWLIB.tar.gz
# Binutils cd $BUILD mkdir $TARGET-binutils-build cd $TARGET-binutils-build ../binutils-
$BINUTILS/configure --target=$TARGET --prefix=$PREFIX --program-prefix=$PROG_PREFIX make
make install ln -s $PREFIX/bin/$PROG_PREFIX""ar $PREFIX/bin/$TARGET-ar ln -s
$PREFIX/bin/$PROG_PREFIX""as $PREFIX/bin/$TARGET-as ln -s $PREFIX/bin/$PROG_PREFIX""ld
$PREFIX/bin/$TARGET-ld ln -s $PREFIX/bin/$PROG_PREFIX""ranlib $PREFIX/bin/$TARGET-ranlib cd
$BUILD rm -rf $TARGET-binutils-build # GCC (C-only) cd $BUILD mkdir $TARGET-gcc-build cd
$TARGET-gcc-build export PATH=$PATH:$PREFIX/bin ../gcc-$GCC/configure --target=$TARGET --
prefix=$PREFIX --program-prefix=$PROG_PREFIX --enable-languages=c --disable-libssp make make
install ln -s $PREFIX/bin/$PROG_PREFIX""gcc $PREFIX/bin/$TARGET-cc cd $BUILD rm -rf $TARGET-
gcc-build # Newlib cd $BUILD cd newlib-$NEWLIB ./configure --target=$TARGET --prefix=$PREFIX --
program-prefix=$PROG_PREFIX make make install # GCC (C and C++) cd $BUILD mkdir $TARGET-
gcc-build cd $TARGET-gcc-build # This is what we wanted to do, but it produces a compiler that
disagrees about # the definition of "size_t" with the rest of the world ../gcc-$GCC/configure --
target=$TARGET --prefix=$PREFIX --program-prefix=$PROG_PREFIX --enable-languages=c,c++ --
disable-nls --disable-multilib --disable-libssp --with-newlib --disable-shared --disable-threads --enable-
target-ospace --with-gnu-as --with-gnu-ld --without-headers # so this was used # ../gcc-
$GCC/configure --target=$TARGET --prefix=$PREFIX --program-prefix=$PROG_PREFIX --enable-
languages=c,c++ --disable-nls --disable-shared --with-gnu-as --with-gnu-ld --disable-threads --disable-
libssp make make install cd $BUILD rm -rf $TARGET-gcc-build # Clean up find $PREFIX/bin -type l -
exec rm \{ } \; cd $BUILD rm -rf binutils-$BINUTILS rm -rf gcc-$GCC rm -rf newlib-$NEWLIB
```