Building a toolchain for use with eCos

Introduction

The eCos source code is designed for compilation with a GNU toolchain. Pre-built toolchains are available for a number of target architectures and we recommend that such a toolchain is used where available. Developers targetting one of the other architectures must build a toolchain themselves at present. This page details the steps required to download toolchain sources and build them on either a Windows host (assuming Cygwin has already been installed) or a Linux host. Please note that building GNU toolchains under Windows 95/98/ME has been found to be unreliable. We recommend that users avoid building the development tools on these systems if possible. Note also, that the instructions presented here are intended for building a cross toolchain for use with eCos only.

Downloading source code

The GNU toolchain sources are made available for download in many component parts. The following parts are required for use with eCos:

- GNU binary utilities - including the GNU assembler (as) and GNU linker (ld)
- GNU compiler collection core - core GCC functionality including the C compliler (gcc)
- GNU compiler collection g++ - the GCC C++ compiler (g++)
- Newlib - a C library intended for use with embedded systems
- GDB - the GNU debugger

It is recommended that you use a mirror site when downloading each component. This will result in a faster download, reduced internet congestion and a reduced load on the central servers.

GNU binary utilities

It is recommended that eCos is built with the most recent official release of the GNU binary utilities (binutils). This is available for download from the binutils area of the GNU FTP site or any of its mirror sites. You should download a file with a name of the form binutils-*version*.tar.bz2 or click on the following link to download version 2.13.1 directly:

- binutils-2.13.1.tar.bz2 (9.5MB)

GNU compiler collection

It is recommended that eCos is built with GCC 3.2.1 at present. This is available for download from the GCC area of the GNU FTP site or any of its mirror sites. Click on each of the following links to download the core and g++ components of version 3.2.1 directly:

- gcc-core-3.2.1.tar.gz (12.8MB)
- gcc-g++-3.2.1.tar.gz (2.5MB)

Newlib

Newlib provides support for building the run-time elements of C++ within the toolchain. It is recommended that eCos is built with the the most recent official release of Newlib. This is available for download from the Newlib area of sources.redhat.com or any of its mirror sites. You should download a file with a name of the form newlib-*version*.tar.gz or click on the following link to download version 1.11.0 directly:

- newlib-1.11.0.tar.gz (4.2MB)

GDB

Instructions for downloading GDB are available via the GDB home page. However, a graphical user interface for GDB named *Insight* is also available. Insight has a separate home page. The Insight sources are a superset of the standard GDB sources and it is still possible to use command-line GDB after building from the Insight sources. We therefore recommend downloading the most recent official release of Insight rather than standard GDB. Both Insight and standard GDB are available for download from the GDB area of sources.redhat.com or any of its mirror sites. You should download a file with a name of the form insight-*version*.tar.bz2 or gdb-*version*.tar.bz2 or click on one of the following links to download directly:

- gdb-5.3.tar.bz2 (10.7MB)
- insight-5.3.tar.bz2 (17.0MB)

The remaining instructions assume that you have downloaded the Insight tarball. If you choose to download the standard GDB tarball, please adapt the instructions accordingly.

Preparing the sources for building

Once the tools sources have been downloaded, they must be prepared before building. These instructions assume that the tool sources will be extracted in the `/src` directory hierarchy. Other locations may be substituted throughout.

Ensure that the file system used has sufficient free space available. The contents of each archive will expand to occupy approximately six times the space required by the compressed archive itself. To extract the downloaded sources, you will need to have both the *bzip2* and *gzip* compression utilities. The *patch* and *tar* utilities are also required. All these tools are supplied with most Linux distributions and with Cygwin.

The following steps should be followed at a *bash* shell prompt:

1. Create a directory to contain the tool sources, avoiding directory names containing spaces as these can confuse the build system:

   ```
   mkdir -p /src
   ```

2. Extract the sources from each downloaded tarball in turn:

   ```
   cd /src
   bunzip2 < binutils-2.13.1.tar.bz2 | tar xvf -
   gunzip < gcc-core-3.2.1.tar.gz | tar xvf -
   gunzip < gcc-g++-3.2.1.tar.gz | tar xvf -
   gunzip < newlib-1.11.0.tar.gz | tar xvf -
   bunzip2 < insight-5.3.tar.bz2 | tar xvf -
   ```

3. Having unpacked the toolchain tarballs, it is now necessary to apply a number of patches to correct known problems with the pristine source code. Click on each of the following links to download the patches:

   - binutils-2.13.1-v850-hashtable.patch
   - gcc-3.2.1-arm-multilib.patch
   - insight-5.3-tcl_win_encoding.patch

   Then apply the patches:

   ```
   patch -p0 < binutils-2.13.1-v850-hashtable.patch
   patch -p0 < gcc-3.2.1-arm-multilib.patch
   patch -p0 < insight-5.3-tcl_win_encoding.patch
   ```

   If any of the patches fail to apply, you should see if there is some obvious problem by attempting to apply the patch manually. If this is not possible, please report the problem to the ecos-discuss mailing list. However, if the *patch* utility reports the following message:

   ```
   Reversed (or previously applied) patch detected!
   Assume -R? [n]
   ```

   then type *n* because this indicates that the patch has already been applied in the master sources.
4. Finally, since Newlib is compiled as part of the GCC build, it is necessary to move the required source code into the GCC directory hierarchy:

   ```
   mv newlib-1.11.0/newlib gcc-3.2.1
   ```

```
mv newlib-1.11.0/libgloss gcc-3.2.1
```

Building the toolchain

Before attempting to build the tools, ensure that the GNU native compiler tools directory is on the PATH and precedes the current directory. The following build procedures will fail if `.` is ahead of the native tools in the PATH.

These instructions assume that the tools will be built in the `/tmp/build` directory hierarchy and installed to `/gnutools`. Other locations may be substituted throughout. The appropriate GNU architecture identifier for the intended eCos target platform should be substituted for the word *`TARGET`* as follows:

Architecture	**`TARGET`**
ARM (including StrongARM, XScale)	`arm-elf`
Intel x86	`i386-elf`
Matsushita AM3x	`mn10300-elf`
Motorola 68K/ColdFire	`m68k-elf`
MIPS32	`mipsisa32-elf`
NEC V850	`v850-elf`
NEC VR4300	`mips64vr4300-elf`
PowerPC	`powerpc-eabi`
Renesas H8/300H	`h8300-elf`
Renesas SuperH	`sh-elf`
Toshiba TX39	`mips-tx39-elf`
Toshiba TX49	`mips-tx49-elf`

The following steps should be followed at a *bash* shell prompt:

1. Configure the GNU binary utilities:

```
mkdir -p /tmp/build/binutils
cd /tmp/build/binutils
/src/binutils-2.13.1/configure --target=TARGET \
  --prefix=/gnutools -v 2>&1 | tee configure.out
```

If there are any problems configuring the tools, you can refer to the file `configure.out` as a permanent record of what happened.

2. Build and install the GNU binary utilities:

   ```
   make -w all install 2>&1 | tee make.out
   ```

   If there are any problems building the tools, you can refer to the file `make.out` as a permanent record of what happened.

3. Configure GCC, ensuring that the GNU binary utilities are at the head of the PATH:

   ```
   PATH=/gnutools/bin:$PATH ; export PATH
   mkdir -p /tmp/build/gcc
   cd /tmp/build/gcc
   /src/gcc-3.2.1/configure --target=TARGET \
     --prefix=/gnutools --enable-languages=c,c++ \
     --with-gnu-as --with-gnu-ld --with-newlib \
     --with-gxx-include-dir=/gnutools/TARGET/include \
     -v 2>&1 | tee configure.out
   ```

4. Build and install GCC:

   ```
   make -w all install 2>&1 | tee make.out
   ```

5. Configure Insight:

   ```
   mkdir -p /tmp/build/gdb
   cd /tmp/build/gdb
   /src/insight-5.3/configure --target=TARGET \
     --prefix=/gnutools -v 2>&1 | tee configure.out
   ```

6. Build and install Insight:

   ```
   make -w all install 2>&1 | tee make.out
   ```

Following successful building and installation of each set of tools, the associated build tree may be deleted to save space if necessary. The toolchain executable files will be located at `/gnutools/bin`. This directory should be added to the head of your PATH.

Troubleshooting

If you encounter difficulties in building or using the development tools, first check the eCos FAQ and the ecos-discuss mailing list archive to see if the topic has come up before. Initial queries may be directed to the ecos-discuss list. However, there are other mailing lists which may be more appropriate if a problem is clearly related to a particular tool:

- Binutils mailing list
- GDB mailing list
- CrossGCC mailing list (for problems specific to GCC cross-compilation)

- GCC mailing list (for more general GCC problems)

Before sending messages to the mailing lists, please consult the web sites associated with each list, to see if there is any relevant documentation or FAQs:

- Binutils home page
- GDB home page
- Insight home page
- GCC web site
- CrossGCC FAQ

It is also worth noting that all these mailing lists have searchable archives.