GNU Toolchain for Atmel ARM Embedded processors: 4.8.4.371

Introduction

The ARM GNU Toolchain supports Atmel$^®$ ARM devices. The ARM toolchain is based on the free and open-source GCC. This toolchain is built from sources publlished by ARM's "GNU Tools for ARM Embedded Processors" project at launchpad.net (https://launchpad.net/gcc-arm-embedded). The toolchain includes compiler, assembler, linker, binutils (GCC and binutils), GNU Debugger (GDB with builtin simulator) and Standard C library (newlib, newlib nano).

Table of Contents

1. Supported configuration

1.1 Supported Hosts

This release includes the following items

- Bare metal EABI pre-built binaries for running on a Windows host

- Bare metal EABI pre-built binaries for running on a Linux host

- Bare metal EABI pre-built binaries for running on a Mac OS X host

1.2 Supported Targets

- Bare metal ARM EABI only

2. Downloading, Installing and Upgrading

The ARM GNU toolchain provided by Atmel® is available for download and install in one of the following ways.

2.1 Downloading/Installing on Windows

- If you just want to try the Atmel ARM GNU toolchain alone, you can download it from here[1]

- If you want to try the Atmel ARM GNU Toolchain along with Atmel studio, you can download and install Atmel studio 6.0 or (newer) which will also install the Atmel® ARM GNU toolchain. See Atmel studio release notes for more details.

2.2 Downloading/Installing on Linux

For Linux, the Atmel® ARM GNU Toolchain is available as tar.gz archive which can be extracted using the tar utility. In order to install, simply extract to the location, from where you want to run from. Linux builds are available from here[2].

Note   64-bit version of libncurses and libc are required to run the tools.

2.3 Downloading/Installing on Mac OS

For Mac, the Atmel® ARM GNU Toolchain is available as tar.gz archive which can be extracted using the tar utility. In order to install, simply extract to the location, from where you want to run from. MAC builds are available from here[3]

2.4 Upgrading from previous versions

If the Atmel® ARM GNU Toolchain is installed by Atmel studio installation, it can be upgraded from Atmel Gallery[4]

If the toolchain is installed separately using one of the (Windows, Linux, Mac) installers, upgrading is not supported. You can install the new package side-by-side of the old package and use it.

---

[1] http://distribute.atmel.no/tools/opensource/Atmel-ARM-GNU-Toolchain/4.8.4
[2] http://distribute.atmel.no/tools/opensource/Atmel-ARM-GNU-Toolchain/4.8.4
[3] http://distribute.atmel.no/tools/opensource/Atmel-ARM-GNU-Toolchain/4.8.4
[4] http://gallery.atmel.com

3. Layout and Components

Listed below are some of the directories that you might want to look, to have a high level understanding of what is packaged inside the Atmel® ARM GNU Toolchain. The layout is identical in Windows, Linux and Mac OS.

3.1 Layout

The layout of the installation is as follows.

- **INSTALLDIR**

  The directory where the ARM GNU Toolchain is installed in the target machine.

- **INSTALLDIR\bin**

  The ARM software development programs.This directory should be in your PATH environemnt variable. (Note : If you are using this toolchin from within Atmel Studio, please configure Atmel studio appropriately). This includes

  - GNU Binutils

  - GCC

- **INSTALLDIR\arm-none-eabi\lib**

  The directory which have the ARM newlib libraries, startup files and linker scripts.

- **INSTALLDIR\arm-none-eabi\include**

  ARM-newlib header files. This is where the system include files will be searched for by the toolchain.

- **INSTALLDIR\lib**

  GCC libraries, other libraries and headers.

- **INSTALLDIR\libexec**

  GCC program components.

3.2 Components

The components used to build this toolchain along with their version number can be found here[1].

---

[1] http://distribute.atmel.no/tools/opensource/Atmel-ARM-GNU-Toolchain/4.8.4

4.   Toolset background

ARM GNU toolchain is a collection of executable software development tools for the Atmel® ARM processors. These software development tools include:

1.   Compiler

2.   Assembler

3.   Linker

4.   Archiver

5.   File converter

6.   Other file utilities

7.   C Library

8.   Debugger

4.1   Compiler

The compiler is the GNU compiler collection, or GCC. This compiler is inredibly flexible and can be hosted on many platforms, it can target many different processors/operating systems(backends), and can be configured for multiple different languages (frontends).

The GCC included is targeted for the ARM processor, and is configured to compile C, and C++.

Because this GCC is targeted for the ARM, the main executable that is created is prefixed with the target name: `arm-none-eabi-gcc`. It is also referred to as ARM GCC.

`arm-none-eabi-gcc` is just a driver program. The compiler itself is called cc1.exe for C, or cc1plus.exe for C++. Also the preprocessor cpp.exe will usually automatically be prefixed with the target name arm-none-eabi-cpp.exe. The actual set of component programs called is usually derived from the suffix of each soruce code file being processed.

GCC compiles a high-level computer lanugage into assembly, and that is all. It cannot work alone. GCC is coupled with another project, GNU Binutils, which provides the assembler, linker, librarian and more. Since GCC is just a driver program, it can automatically call the assembler and linker directly to build the final program.

4.2   Assembler, Linker, Librarian

GNU Binutils is a collection of binary utilities. This also includes the assembler,**as**. Sometimes you will see it referenced as GNU as or **gas**. Binutils includes the linker, **ld**; the librarian or archiver, **ar**. There are many other programs included that provide various functionality.

Binutils is configured for the ARM target and each of the programs is prefixed with the target name. So you have programs such as:

*   **arm-none-eabi-as**: The GNU Assembler

*   **arm-none-eabi-ld**: The GNU Linker

*   **arm-none-eabi-ar**: The GNU Archiver, Create, modify, and extract from archives (libraries)

*   **arm-none-eabi-ranlib**:Generate index of archive (library) contents

*   **arm-none-eabi-objcopy**:Copy and translate object files

*   **arm-none-eabi-objdump**:Display information from object files including disassembly

*   **arm-none-eabi-size**:List section size, total size

*   **arm-none-eabi-nm**:List symbol from object files.

*   **arm-none-eabi-strings**:List printable strings from files

- **arm-none-eabi-strip**:Discard symbols

- **arm-none-eabi-readelf**:Display the contents of ELF file formats

- **arm-none-eabi-addr2line**:Convert addresses to file and line

- **arm-none-eabi-c++filt**:Filter to demangle encoded C++ symbols

- **arm-none-eabi-gdb**:Debugger to debug the target

See the binutils user manual for more information on what each program can do.

4.3 C Library

Newlib is the Standard C Library for ARM GCC. Newlib is the C library intended for use on embedded systems. It is a conglomeration of sevaral library parts. The library is ported to support ARM processor.

In addition to standard C library, newlib-nano also added to the toolchain package. Newlib-nano is newlib branch optimized for code size by ARM (https://launchpad.net/gcc-arm-embedded). To use newlib-nano, users should provide additional gcc link option "--specs=nano.specs". For more details please refer to the readme[1] file for more information.

4.4 Debugging

- The toolchain distribution ships the `arm-none-eabi-gdb` which can be used for debugging purposes.

- Atmel Studio provides faclities to debug the executable produced by this toolchain.Note that `Atmel Studio` is currently free to the public, but it is not Open Source.

4.5 Source code

Atmel have packaged GDB 7.8 with this version of ARM GNU toolchain. This is done to improve the debugger quality. More information about this version of gdb please refer here[2]. Note : The original launchpad distribution ships GDB 7.7.

The source code details and build configurations are available in the release[3] and readme[4].Please refer them for more information.

---

[1] http://distribute.atmel.no/tools/opensource/Atmel-ARM-GNU-Toolchain/4.8.4
[2] http://www.gnu.org/software/gdb/news/
[3] http://distribute.atmel.no/tools/opensource/Atmel-ARM-GNU-Toolchain/4.8.4
[4] http://distribute.atmel.no/tools/opensource/Atmel-ARM-GNU-Toolchain/4.8.4

5. New and Noteworthy

This chapter lists the new and noteworthy items for the ARM GNU Toolchain release.

5.1 Supported devices

SAM3A	SAM3N	SAM3S	SAM3U	SAM3X	SAM4C
SAM4E	SAM4L	SAM4N	SAM4S	SAM4SP	SAM7S/SE
SAM7X/XC	SAM7L	SAM9XE	SAM9N	SAM9R	SAM9G
SAM9X	SAM9M	SAMD	SAMG	SAMR	

5.2 Noteworthy

● ARM GDB built from gdb 7.8. See "Source code" section in this document for more details.

● Please read the launchpad's release[1] document, which has more information on new features and fixes, included with this release.

● The toolchain is configured to produce DWARF-2 binaries by default, which is supported by atmel software debugger tools.

● The toolchain is configured to produce code for **ARM thumb instruction set** by default. This is done, in order to maintain compatibility with previous versions.

---

[1] http://distribute.atmel.no/tools/opensource/Atmel-ARM-GNU-Toolchain/4.8.4

6. Contact Information and Disclaimer

6.1 Contact

For support on Atmel® ARM GNU Toolchain, please contact support@atmel.com

6.2 Disclaimer

Atmel® ARM GNU toolchain is distributed free of charge for the purpose of developing applications for Atmel® SAM devices. THIS SOFTWARE IS PROVIDED BY ATMEL "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT ARE EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Atmel Enabling Unlimited Possibilities®