



GNU Toolchain for Atmel AVR8 Embedded Processors

Introduction

The Atmel AVR 8-bit GNU Toolchain (3.6.1.1752) supports all AVR 8-bit devices. The AVR 8-bit Toolchain is based on the free and open-source GCC compiler. The toolchain includes compiler, assembler, linker and binutils (GCC and Binutils), Standard C library (AVR-libc) and GNU Debugger (GDB).

Table of Contents

Introduction	1
1. Installation Instructions	3
1.1. System requirements	3
1.1.1. Hardware requirements	3
1.1.2. Software Requirements	3
1.2. Downloading, Installing and Upgrading	3
1.2.1. Downloading/Installing on Windows	3
1.2.2. Downloading/Installing on Linux	3
1.2.3. Upgrading from previous versions	3
1.3. Layout	3
2. Toolset Background	5
2.1. Component Versions	5
2.2. Compiler	5
2.3. Assembler, Linker, Librarian and More	5
2.4. C Library	6
2.5. Debugging	6
2.6. Source Code	6
3. Bugs and New Features	7
3.1. Notable Bugs Fixed	7
3.2. Known Issues	7
4. Supported Devices	8
5. Contact Information and Disclaimer	10
5.1. Disclaimer	10

1. Installation Instructions

1.1 System requirements

1.1.1 Hardware requirements

- Minimum processor Pentium 4, 1GHz
- Minimum 512 MB RAM
- Minimum 500 MB free disk space

AVR 8-bit GNU Toolchain has not been tested on computers with less resources, but may run satisfactorily depending on the number and size of the projects and the user's patience.

1.1.2 Software Requirements

- Windows 2000, Windows XP, Windows Vista, Windows 7 (x86 or x86-64) or Windows 8 (x86 or x86-64)
- AVR 8-bit GNU Toolchain is not supported on Windows 98, NT or ME.
- The toolchain should work on the Linux distributions Fedora, RedHat Enterprise, Arch Linux and Ubuntu for both 32-bits and 64-bits architecture. AVR 8-bit GNU Toolchain may very well work on other distributions. However those are untested and unsupported.

1.2 Downloading, Installing and Upgrading

The AVR8 GNU toolchain provided by Atmel is available for download and install in one of the following ways.

1.2.1 Downloading/Installing on Windows

- If you want to try the Atmel AVR8 GNU toolchain alone, you can download it from [here](#)¹
- If you want to try the Atmel AVR8 GNU Toolchain along with Atmel Studio, you can download and install Atmel Studio 7 or (newer) which will also install the Atmel AVR8 GNU toolchain. See Atmel Studio release notes for more details.

1.2.2 Downloading/Installing on Linux

For Linux, the Atmel AVR8 GNU Toolchain is available as a 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 it from. Linux builds are available from [here](#)².

1.2.3 Upgrading from previous versions

If the Atmel AVR8 GNU Toolchain is installed by Atmel Studio installation, refer Atmel Studio documentation to upgrade.

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.

1.3 Layout

Listed below are some directories you might want to know about.

`<install_dir>` = The directory where you installed AVR 8-bit GNU Toolchain.

- `<install_dir>\bin`
The AVR software development programs. This directory should be in your ``PATH`` environment variable. This includes:
 - GNU Binutils
 - GCC

¹ <http://www.atmel.com/tools/ATMELAVRTOOLCHAINFORWINDOWS.aspx>

² <http://www.atmel.com/tools/ATMELAVRTOOLCHAINFORLINUX.aspx>

- GDB
- <install_dir>\avr\lib
avr-libc libraries, startup files, linker scripts, and stuff.
- <install_dir>\avr\include
avr-libc header files for AVR 8-bit.
- <install_dir>\avr\include\avr
header files specific to the AVR 8-bit MCU. This is where, for example, `#include <avr/io.h>` comes from.
- <install_dir>\lib
GCC libraries, other libraries, headers and stuff.
- <install_dir>\libexec
GCC program components
- <install_dir>\doc
Various documentation.

2. Toolset Background

AVR 8-bit GNU Toolchain is a collection of executable, open source software development tools for the Atmel AVR 8-bit series of microcontrollers. It includes the GNU GCC compiler for C and C++.

2.1 Component Versions

GCC: 5.4.0

binutils: 2.26.20160125

avr-libc: "2.0.0"

gdb:

2.2 Compiler

The compiler is the GNU Compiler Collection, or GCC. This compiler is incredibly flexible and can be hosted on many platforms, it can target many different processors/operating systems (back-ends), and can be configured for multiple different languages (front-ends).

The GCC included in AVR 8-bit GNU Toolchain is targeted for the AVR 8-bit microcontroller and is configured to compile C or C++.

CAUTION: There are caveats on using C++. See the avr-libc FAQ. C++ language is not fully supported and has some limitations. libstdc++ is unsupported.

Because this GCC is targeted for the AVR 8-bit MCUs, the main executable that is created is prefixed with the target name: ``avr-gcc`` (with `.exe` extension on MS Windows). It is also referred to as AVR GCC.

``avr-gcc`` is just a "driver" program only. The compiler itself is called ``cc1.exe`` for C, or ``cc1plus.exe`` for C++. Also, the preprocessor ``cpp.exe`` will usually automatically be prepended with the target name: ``avr-cpp``. The actual set of component programs called is usually derived from the suffix of each source code file being processed.

GCC compiles a high-level computer language 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.

2.3 Assembler, Linker, Librarian and More

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.

Note that while the assembler uses the same mnemonics as proposed by Atmel, the "glue" (pseudo-ops, operators, expression syntax) is derived from the common assembler syntax used in Unix assemblers, so it is not directly compatible to Atmel assembler source files.

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

- `avr-as`: The Assembler.
- `avr-ld`: The Linker.
- `avr-ar`: Create, modify, and extract from archives (libraries).
- `avr-ranlib`: Generate index to archive (library) contents.
- `avr-objcopy`: Copy and translate object files.
- `avr-objdump`: Display information from object files including disassembly.
- `avr-size`: List section sizes and total size.
- `avr-nm`: List symbols from object files.
- `avr-strings`: List printable strings from files.
- `avr-strip`: Discard symbols.

- *avr-readelf*: Display the contents of ELF format files.
- *avr-addr2line*: Convert addresses to file and line.
- *avr-c++filt*: Filter to demangle encoded C++ symbols.
- *avr-gdb*: GDB, the GNU debugger, allows you to see what is going on 'inside' another program targeted to AVR, while it executes.

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

2.4 C Library

avr-libc is the Standard C Library for AVR 8-bit GCC. It contains many of the standard C routines, and many non-standard routines that are specific and useful for the AVR 8-bit MCUs.

In addition to avr-libc libraries, Host IO library (libhostio.a) is integrated to this toolchain. This Host IO library allows the target to use the host's file system and console I/O to perform various avr I/O operations.

NOTE: The actual library is currently split into two main parts, libc.a and libm.a, where the latter contains mathematical functions (everything mentioned in <math.h>, and a bit more). Also, there are additional libraries which allow a customization of the printf and scanf function families. avr-libc contains documentation on how to use (and build) the entire toolset, including code examples. The avr-libc user manual also contains the FAQ on using the toolset.

2.5 Debugging

Atmel Studio provides a debugger and also provides simulators for the parts that can be used for debugging as well. Note that 'Atmel Studio' is currently free to the public, but it is not Open Source. The GNU debugger is now shipped along with the toolchain.

2.6 Source Code

Atmel AVR 8-bit GNU Toolchain uses modified source code from GCC, Binutils and AVR-LibC. The source code and the build scripts used for building the packaged binaries are available [here](http://distribute.atmel.no/tools/opensource/Atmel-AVR-GNU-Toolchain/3.6.1)¹.

Please refer to the README for the instructions on how to use the supplied script to build the toolchain.

¹ <http://distribute.atmel.no/tools/opensource/Atmel-AVR-GNU-Toolchain/3.6.1>

3. Bugs and New Features

3.1 Notable Bugs Fixed

Issue #AVRTC-856:

Fix broken source line to address mapping DWARF info when linker relaxation is enabled.

Issue #AVRTC-857:

Improve const data handling for devices that see flash memory in data address space (memory mapped flash) e.g, tiny40, tiny416 Backport of gcc's PR 78093, 81072 and respective binutils patches

Issue #AVRTC-861:

Enabled clock_prescale_set function in power.h for missed devices (e.g. tiny87, 88, mega88A, 88PA etc)

3.2 Known Issues

Issue #AVRTC-731:

For AVRTINY architecture, libgcc implementation has some known limitations. Standard C / Math library implementation is very limited or not present.

Issue #AVRTC-732:

Program memory images beyond 128KBytes are supported by the toolchain, subject to the limitations mentioned in "3.17.4.1 EIND and Devices with more than 128 Ki Bytes of Flash" at <http://gcc.gnu.org/onlinedocs/gcc/AVR-Options.html>

Issue #AVRTC-733:

Named address spaces are supported by the toolchain, subject to the limitations mentioned in "6.16.1 AVR Named Address Spaces" at <http://gcc.gnu.org/onlinedocs/gcc/Named-Address-Spaces.html#AVR%20Named%20Address%20Spaces>

4. Supported Devices

Most of the AVR8 devices are supported by this toolchain. Users can get new devices support from Atmel Device Family Packs (DFP). Download DFPs from [here](http://packs.download.atmel.com/)¹.

Using DFPs with this toolchain:

- Download DFP which has required device support. (e.g. ATmega328PB is part of Atmel ATmega Series DFP.)
- Unzip downloaded *.atpack file to packs directory (e.g. /home/packs/).
- Invoke avr-gcc with additional option -B to tell gcc where to look for device specific information and -I for device header include path.

e.g. avr-gcc -mmcu=atmega328pb -B /home/packs/Atmel.ATmega_DFP.1.0.86/gcc/dev/atmega328pb/ -I /home/packs/Atmel.ATmega_DFP.1.0.86/include/

avr2

at90s2313	at90s2343	at90s4414	at90s8515
at90s2323	attiny22	at90s4433	at90c8534
at90s2333	attiny26	at90s4434	at90s8535

avr25

ata5272	attiny4313	attiny85	attiny87
ata6616c	attiny44	attiny261	attiny48
attiny13	attiny44a	attiny261a	attiny88
attiny13a	attiny441	attiny461	attiny828
attiny2313	attiny84	attiny461a	attiny841
attiny2313a	attiny84a	attiny861	at86rf401
attiny24	attiny25	attiny861a	
attiny24a	attiny45	attiny43u	

avr3

at43usb355	at76c711
------------	----------

avr31

atmega103	at43usb320
-----------	------------

avr35

ata5505	at90usb82	atmega16u2	attiny1634
ata6617c	at90usb162	atmega32u2	
ata664251	atmega8u2	attiny167	

avr4

ata6285	atmega48a	atmega88pa	at90pwm2b
ata6286	atmega48p	atmega88pb	at90pwm3
ata6289	atmega48pa	atmega8515	at90pwm3b
ata6612c	atmega48pb	atmega8535	at90pwm81
atmega8	atmega88	atmega8hva	
atmega8a	atmega88a	at90pwm1	
atmega48	atmega88p	at90pwm2	

avr5

ata5702m322	atmega168pb	atmega329a	atmega649p
ata5782	atmega169	atmega329p	atmega6490
ata8210	atmega169a	atmega329pa	atmega16hva
ata5790	atmega169p	atmega3290	atmega16hva2
ata5790n	atmega169pa	atmega3290a	atmega32hvb
ata5791	atmega16hvb	atmega3290p	atmega6490a
ata5795	atmega16hvbrevb	atmega3290pa	atmega6490p
ata5831	atmega16m1	atmega32c1	atmega64c1
ata8510	atmega16u4	atmega32m1	atmega64m1
ata6613c	atmega32a	atmega32u4	atmega64hve
ata6614q	atmega32	atmega32u6	atmega64hve2

¹ <http://packs.download.atmel.com/>

atmega16 atmega16a atmega161 atmega162 atmega163 atmega164a atmega164p atmega164pa atmega165 atmega165a atmega165p atmega165pa atmega168 atmega168a atmega168p atmega168pa	atmega323 atmega324a atmega324p atmega324pa atmega325 atmega325a atmega325p atmega325pa atmega3250 atmega3250a atmega3250p atmega3250pa atmega328 atmega328p atmega328pb atmega329	atmega406 atmega64 atmega64a atmega640 atmega644 atmega644a atmega644p atmega644pa atmega645 atmega645a atmega645p atmega6450 atmega6450a atmega6450p atmega649 atmega649a	atmega64rfr2 atmega644rfr2 atmega32hvbrevb at90can32 at90can64 at90pwm161 at90pwm216 at90pwm316 at90scr100 at90usb646 at90usb647 at94k m3000
avr51 atmega128 atmega128a atmega1280	atmega1281 atmega1284 atmega1284p	atmega128rfa1 atmega128rfr2 atmega1284rfr2	at90can128 at90usb1286 at90usb1287
avr6 atmega2560	atmega2561	atmega256rfr2	atmega2564rfr2
avrxmega2 atxmega8e5 atxmega16a4 atxmega16d4 atxmega16e5	atxmega32a4 atxmega32c3 atxmega32d3 atxmega32d4	atxmega16a4u atxmega16c4 atxmega32a4u atxmega32c4	atxmega32e5
avrxmega3 attiny212 attiny214 attiny412 attiny414	attiny416 attiny417 attiny814 attiny816	attiny817 attiny1614 attiny1616 attiny1617	attiny3214 attiny3216 attiny3217
avrxmega4 atxmega64a3 atxmega64d3	atxmega64a3u atxmega64a4u	atxmega64b1 atxmega64b3	atxmega64c3 atxmega64d4
avrxmega5 atxmega64a1	atxmega64a1u		
avrxmega6 atxmega128a3 atxmega128d3 atxmega192a3 atxmega192d3 atxmega256a3	atxmega256a3b atxmega256a3bu atxmega256d3 atxmega128a3u atxmega128b1	atxmega128b3 atxmega128c3 atxmega128d4 atxmega192a3u atxmega192c3	atxmega256a3u atxmega256c3 atxmega384c3 atxmega384d3
avrxmega7 atxmega128a1	atxmega128a1u	atxmega128a4u	
avrtiny attiny4 attiny5	attiny9 attiny10	attiny20 attiny40	
avr1 at90s1200 attiny11	attiny12 attiny15	attiny28	

5. Contact Information and Disclaimer

For support on Atmel AVR 8-bit GNU Toolchain, visit [design support](http://www.atmel.com/design-support/)¹.

Users of AVR 8-bit GNU Toolchain are also welcome to discuss on the AVRfreaks website forum for AVR Software Tools.

5.1 Disclaimer

AVR 8-bit GNU Toolchain is distributed free of charge for the purpose of developing applications for Atmel AVR processors. AVR 8-bit GNU Toolchain comes without any warranty.

¹ <http://www.atmel.com/design-support/>



Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA T: (+1)(408) 441.0311 F: (+1)(408) 436.4200 | www.atmel.com

© 2017 Atmel Corporation. / Rev.: 42372A-MCU-12/2017

Atmel®, Atmel logo and combinations thereof, Enabling Unlimited Possibilities®, AVR®, tinyAVR®, XMEGA®, megaAVR®, and others are registered trademarks or trademarks of Atmel Corporation in U.S. and other countries. Windows®, and others, are registered trademarks of Microsoft Corporation in U.S. and or other countries. Other terms and product names may be trademarks of others.

DISCLAIMER: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN THE ATMEL TERMS AND CONDITIONS OF SALES LOCATED ON THE ATMEL WEBSITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and products descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

SAFETY-CRITICAL, MILITARY, AND AUTOMOTIVE APPLICATIONS DISCLAIMER: Atmel products are not designed for and will not be used in connection with any applications where the failure of such products would reasonably be expected to result in significant personal injury or death ("Safety-Critical Applications") without an Atmel officer's specific written consent. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Atmel products are not designed nor intended for use in military or aerospace applications or environments unless specifically designated by Atmel as military-grade. Atmel products are not designed nor intended for use in automotive applications unless specifically designated by Atmel as automotive-grade.