

AVR1927: XMEGA-A1 Xplained Getting Started Guide



Features

- Easy to reprogram with just a USB cable and a preprogrammed boot loader
- Easy to debug code with JTAG-based debugger/emulator
- Can be used with Atmel® AVR® STK®600, Atmel AVR Dragon™, Atmel AVR JTAGICE mkII, and Atmel AVR ONE!

8-bit **AVR**[®]
Microcontrollers

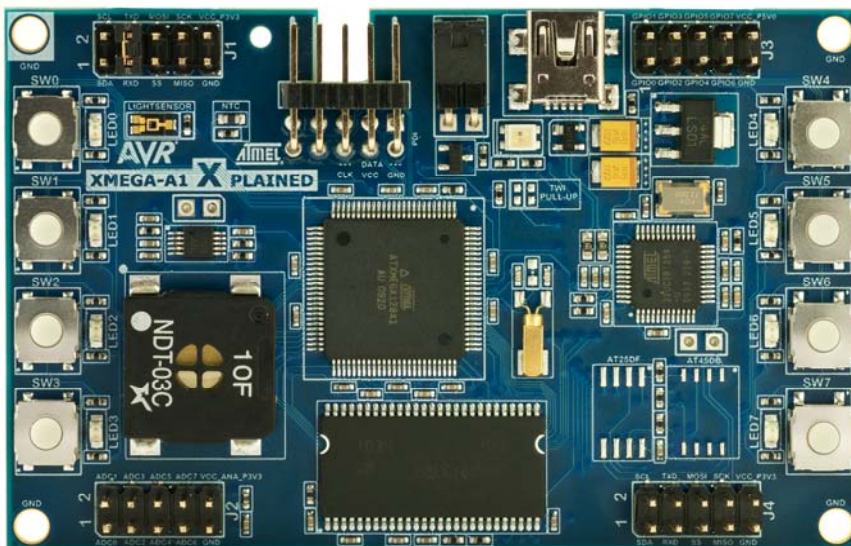
1 Introduction

The Atmel XMEGA-A1 Xplained evaluation kit is a hardware platform for evaluating the [Atmel ATxmega128A1](#) microcontroller.

The kit offers a range of features that enable the Atmel AVR XMEGA[®] user to get started using AVR XMEGA peripherals right away and understand how to integrate the AVR XMEGA device in their own design. This getting started guide will walk you through all the basic steps needed to start using the XMEGA-A1 Xplained evaluation kit.

Application Note

Figure 1-1. XMEGA-A1 Xplained evaluation kit.



Rev. 8372A-AVR-04/11





2 What is needed to get started?

To be able to run the preprogrammed code, you need the Atmel XMEGA-A1 Xplained evaluation kit and a USB cable (Standard-A to Mini-B or Mini-AB) to connect the board to a PC.

When connecting the XMEGA-A1 Xplained to a PC, the operating system will request a driver file for installing the serial communication driver. This driver file is available in the zip file associated with this document. The driver file supports both 32- and 64-bit versions of Windows® XP and Windows 7. Driver installs are not necessary on Linux® operating systems.

To modify the example code or write new code and compile it, you need a toolchain for Atmel AVR microcontrollers and an IDE to edit and debug code. Atmel provides both the [Atmel AVR Studio® 4](#) and the Atmel AVR GNU Toolchain. Other IDEs and toolchains are also available, such as the IAR™ Embedded Workbench.

If you also want to debug the code at runtime, you'll need a debugger like the Atmel AVR JTAGICE mkII or Atmel AVR ONE! More information on these tools is available from the Atmel website. See the reference section later in this document for more details.

2.1 Documentation for XMEGA-A1 Xplained

The [AVR1924: XMEGA-A1 Xplained Hardware User's Guide](#) package includes all in-depth information about the kit, such as schematics, BOM, assembly drawings, 3D plots, and more. The package is available from the Atmel website:

www.atmel.com/xplained

2.2 AVR Software Framework

The AVR Software Framework (ASF) provides software drivers and libraries to build applications for Atmel AVR devices. It has been designed to help develop and glue together the different components of a software design. It can easily integrate into an operating system (OS) or run as a standalone product.

The new AVR Software Framework (version 2+) can be used to get the latest software updates. It can be found here:

http://atmel.com/dyn/products/tools_card.asp?tool_id=4192&category_id=163&family_id=607&subfamily_id=2138

2.3 Integrated development environment for Atmel AVR microcontrollers

2.3.1 AVR Studio 4

[AVR Studio 4](#) is a free integrated development environment (IDE) for Atmel AVR 8-bit MCUs. It can be used to write code, program, and debug a microcontroller.

Atmel AVR Studio 4

www.atmel.com/avrstudio

2.3.2 Embedded Workbench for Atmel AVR

IAR Embedded Workbench® is an integrated development environment with project management tools and editor.

IAR Embedded Workbench for Atmel AVR

<http://www.iar.com/website1/1.0.1.0/107/1/>

2.4 FLIP (BatchISP)

BatchISP is a command line tool for programming the flash memory and EEPROM of the board controller, and is part of the **FLIP** installation. It can be used to communicate with the preprogrammed boot loader on the board controller.

FLIP (FLexible In-system Programmer) for programming:

http://atmel.com/dyn/products/tools_card.asp?tool_id=3886

2.5 Toolchain

In order to build an application from source code, a toolchain, which includes the assembler, compiler, linker, and other required tools, is needed. The code for this kit is compatible with the AVR GNU and IAR AVR toolchains.

2.5.1 AVR GNU Toolchain

This is a free toolchain that can be downloaded from the Atmel webpage. It includes all the necessary tools, such as C/C++ compiler, assembler and linker, for AVR application development. The toolchain integrates with the **Atmel AVR Studio 4** application through the AVR GCC plug-in.

AVR GNU Toolchain (free C/C++ compiler for AVR 8- and 32-bit MCUs)

http://www.atmel.com/dyn/products/tools_card.asp?tool_id=2725

2.5.2 IAR AVR Toolchain

This is a commercial toolchain that includes all the necessary tools, such as C/C++ compiler, assembler, and linker, for AVR application development. A 30-day evaluation version as well as a 4k (code size limited) kick-start version is available from the IAR website.

IAR Embedded Workbench for Atmel AVR

<http://www.iar.com/website1/1.0.1.0/107/1/>

2.6 Programmers and debuggers

Programmers and debuggers are needed when either the application is programmed without using the boot loader or when debugging support is needed. Several different tools are available that can be used directly with the Xplained kit. The most common tools are described in the following chapters.

2.6.1 AVR Dragon

Atmel AVR Dragon is a low-cost debug/programming tool that can be used for code development with the XMEGA-A1 Xplained kit.

Atmel AVR Dragon (on-chip programming and debugging tool)

http://atmel.com/dyn/products/tools_card.asp?tool_id=3891



2.6.2 AVR JTAGICE mkII

Atmel AVR JTAGICE mkII is a mid-range debug/programming tool that can be used for code development with the XMEGA-A1 Xplained kit.

Atmel AVR JTAGICE mkII (on-chip programming and debugging tool)

http://www.atmel.com/dyn/products/tools_card.asp?tool_id=3353

2.6.3 AVR ONE!

Atmel AVR ONE! is a high-end debug/programming tool that can be used for code development with the Atmel XMEGA-A1 Xplained kit.

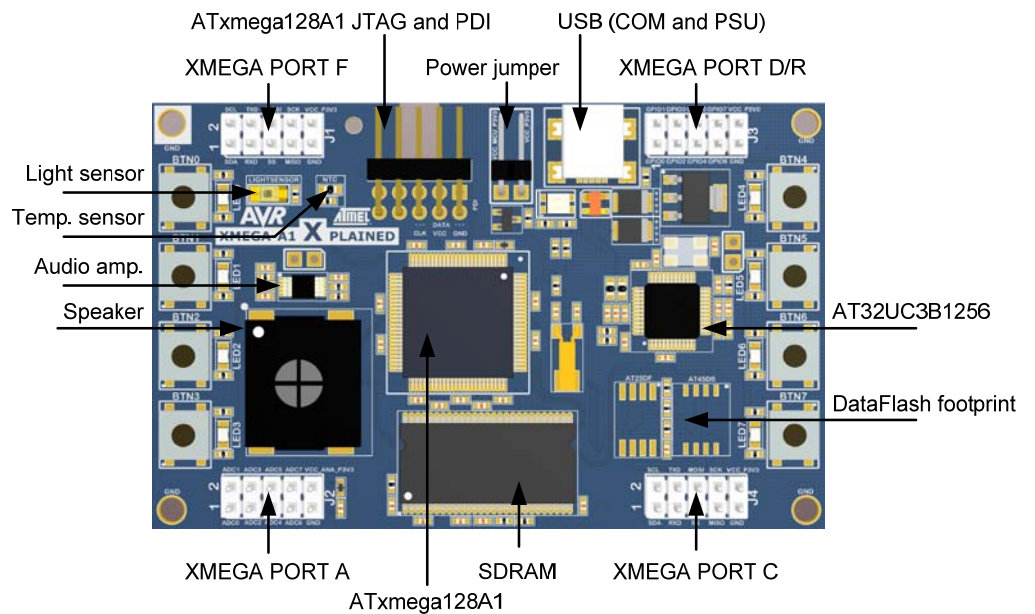
Atmel AVR ONE! (on-chip programming and debugging tool)

http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4279

3 Board overview

Figure 3-1 shows the features available on the board. For more information on the XMEGA-A1 Xplained hardware, take a look at the Atmel application note [AVR1924: XMEGA-A1 Xplained Hardware User's Guide](#).

Figure 3-1. Overview of the XMEGA-A1 Xplained kit.



4 Connecting the board

Connect a USB cable between the board and a PC or a USB power supply to power it up. That is all that is needed. When power is applied, the Power/Status LED will light up green.

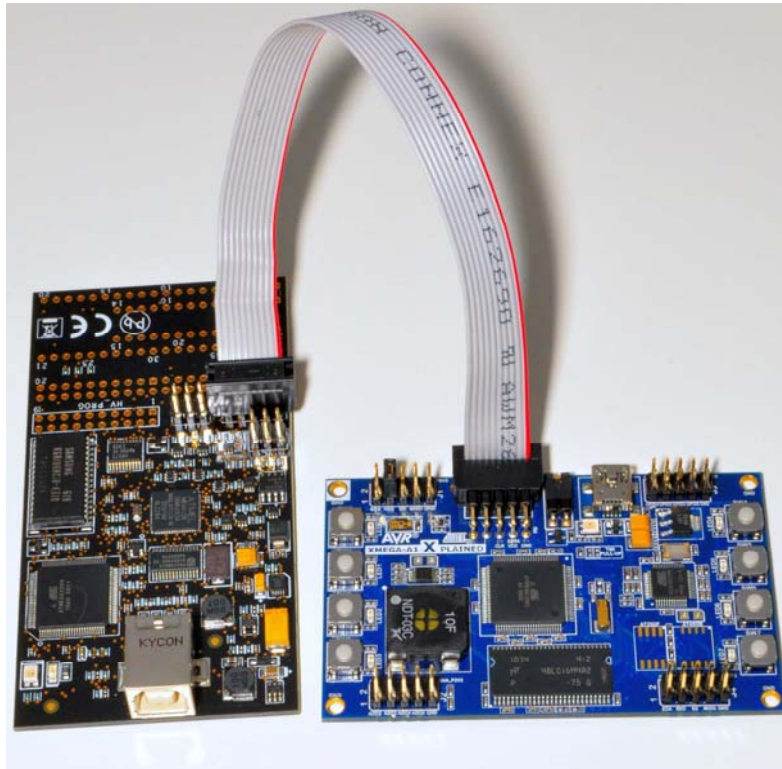
WARNING

Do not power the board without having the jumper attached next to the USB connector or an ammeter mounted. Otherwise, the device may be damaged by power sourcing through I/O pins. The power measurement header is located next to the USB connector, as shown in [Figure 3-1](#).

4.1 Connecting AVR Dragon

A 10-pin header cable is needed to connect the Atmel AVR Dragon tool to the Atmel XMEGA-A1 Xplained board. Connect the cable between the JTAG connector on the AVR Dragon kit and the XMEGA-A1 Xplained JTAG connector. Take a look at [Figure 4-1](#) to see how the connection should be made.

Figure 4-1. Connecting the AVR Dragon kit to an Xplained board.



4.2 Connecting AVR JTAGICE mkII

The grey connector on the Atmel AVR JTAGICE mkII probe has to be used when connecting to the XMEGA-A1 Xplained board. See [Figure 4-2](#) for reference on how to make this connection. The notch of the grey connector must be placed into the cut-out of the board.

Figure 4-2. Connecting the AVR JTAGICE mkII probe to an Xplained board.



4.3 Connecting AVR ONE!

The Atmel AVR ONE! JTAG connector can be connected to the Atmel XMEGA-A1 Xplained kit with a standoff adapter, as shown in [Figure 4-3](#).

Figure 4-3. Connecting the AVR ONE! debugger to an Xplained board.



5 Programming the kit

5.1 Programming via the boot loader

The preprogrammed boot loader on the [Atmel ATxmega128A1](#) offers the possibility to program the device via the UART interface. Because the default firmware of the board controller acts as a UART-to-USB bridge, the [ATxmega128A1](#) can be programmed from the PC via the USB connection to the kit.

The boot loader on the [ATxmega128A1](#) is evoked by pushing the mechanical switch (SW0) during reset. Programming can be performed by using the BatchISP command line tool from the [FLIP](#) installation.

5.1.1 Prerequisite

To program the device, you need BatchISP, which can be found under the [FLIP](#) installation directory. You need to install [FLIP version 3.4.2](#) or later.

To program the new firmware into the device using BatchISP, you need to have the UART-to-USB gateway in the board controller intact.

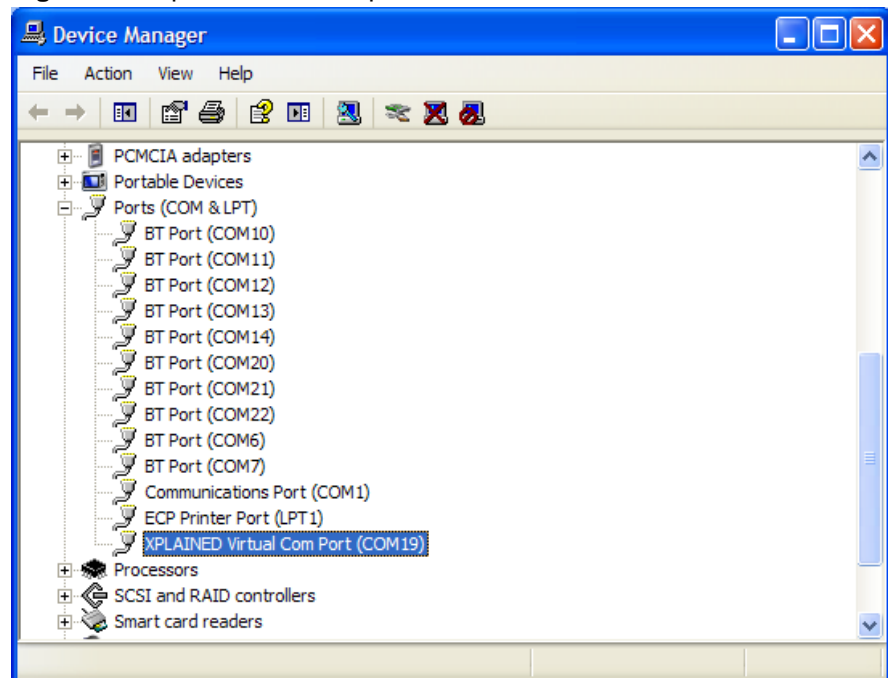
[FLIP](#) needs an XML configuration file for the device. If using [FLIP v3.4.2](#), the XML configuration file is not included. Copy the XML file from the zip file associated with this application note, and put the file in the `Flip 3.4.2\bin\PartDescriptionFiles` installation directory.

5.1.2 Step-by-step guide

To program the device, please follow these steps:

1. Disconnect the USB cable from the kit
2. Press and hold the button (SW0) while re-attaching the USB cable
3. Open the device manager and locate the COM port used by the kit:

Figure 5-1. Xplained kit COM port.



4. Run the command below using the correct COM port:

```
Batchisp -device ATXMEGA128A1
         -hardware RS232
         -port COM19
         -baudrate 115200
         -operation onfail abort
memory flash
erase f
blankcheck
loadbuffer your_firmware_file.hex
program verify
start reset 0
```

5. Your firmware should now be running.

NOTE

The boot loader can only be synchronized once after power-up, which means that you can only connect to the target once using BatchISP. If you were not able to do everything you needed to do on the first try, you will need to power cycle the board and run the command again. An example of programming the device via the boot loader can be seen in [Figure 5-2](#).

Figure 5-2. Example of programming with BatchISP.

```

C:\temp>batchisp -device ATXMEGA128A1 -hardware RS232 -port COM19 -baudrate 115200 -operation onfail abort memory flash erase f blankcheck loadbuffer your_firmware_file.hex program verify start reset 0
Running batchisp 1.2.4 on Wed Feb 09 13:32:41 2011

ATXMEGA128A1 - RS232 - COM19 - 115200

Device selection..... PASS
Hardware selection..... PASS
Opening port..... PASS
Synchronizing target..... PASS
Reading Bootloader version..... PASS      1.0.0
Selecting FLASH..... PASS
Erasing..... PASS
Blank checking..... PASS      0x00000 0x1ffff
Parsing HEX file..... PASS      your_firmware_file.hex
Programming memory..... PASS      0x00000 0x011e0
Verifying memory..... PASS      0x00000 0x011e0
Starting Application..... PASS      RESET  0

Summary: Total 12 Passed 12 Failed 0

C:\temp>
```

The boot loader supports reading and writing of the following memories:

- Flash (0x00000 - 0x1FFFF, application section of the flash)
- EEPROM (0x000 - 0x7FF)
- User (0x000 - 0x1FF, user signature row)

For more information about the BatchISP command line tool, run “batchisp -h” from a command line to get a list of options and parameters. You can also read the [FLIP](#) help file for information.



5.2 Restoring the boot loader

When the boot loader has been erased, it can be restored with any of the previously mentioned programming tools.

The hex file for the boot loader can be found in the zip file associated with this application note and is named *atxmega128a1_batchisp_uart_bootloader.hex*

NOTE

The boot loader section is protected, and can only be erased with an external programmer.

6 Example applications

Several examples are available in the ASF that can be downloaded from the Atmel website ASF section:

http://atmel.com/dyn/products/tools_card.asp?tool_id=4192&category_id=163&family_id=607&subfamily_id=2138

Each example has full source code with documentation, as well as ready-made binaries to get you started quickly. The examples can be compiled using either GCC or IAR compilers.

6.1 Preprogrammed firmware

The [Atmel ATxmega128A1](#) MCU on the Atmel XMEGA-A1 Xplained board is preprogrammed with a UART boot loader and default example firmware.

6.1.1 Functionality of the preprogrammed firmware

The preprogrammed firmware in the [ATxmega128A1](#) MCU includes a memory game where the user can enter a pattern by pressing the buttons on the Xplained board in any given pattern. When the pattern is stored, the user can try to copy the pattern by following the feedback given by the board.

The board has some configurations available to change the mode of operation. The configuration can be changed by placing jumpers on PORTA (J2). The available configurations are:

- With no jumpers attached on the port, the board shows parallel running lights. The running lights mark the timeout until a button press sequence is stored to EEPROM, and a game of memory is started using the saved button presses. If no buttons have been pressed when the board times out, the timeout will be reset
- If a jumper is placed between port pins 0 and 1 during power-up (reset), the board will try to load existing button presses stored in EEPROM, if any, and start playing memory directly
- If a jumper is placed between port pins 2 and 3 during power-up or reset, the board will play tones from the chromatic scale, starting at an A of 441Hz, instead of playing the sound samples

6.2 More example projects and tutorials

For further exploration of the [ATxmega128A1](#), please see the application notes that are applicable for the device. Also, there are several example projects and drivers available in the AVR Software Framework modules for the [ATxmega128A1](#). Ready-made projects targeted just for the XMEGA-A1 Xplained kit are also available.

For basic training to get familiar with the [ATxmega128A1](#) device, Atmel application notes AVR1500 to AVR1510 are a good starting point. These application notes are available from the Atmel website:

http://atmel.com/dyn/products/tools_docs.asp?category_id=163&family_id=607&subfamily_id=1965&tool_id=4506





7 How to compile example projects

The compilers generate two files when compiling a project. The hex file contains the actual data that will be programmed into the device. This hex file can be programmed into the device using one of the programmers listed in Section 2.6 or by using the preprogrammed boot loader.

The other file generated is a debug file, which contain both the data to be programmed into the device and debug information to make the code readable during debugging. This file can be opened in [Atmel AVR Studio 4](#) for single-stepping and debugging. The file extension of the debug file depends on the compiler, and will be either .elf for AVR GNU Toolchain or .dbg for IAR Embedded Workbench.

7.1 AVR GNU Toolchain

To make this project using AVR GNU Toolchain, open a shell/dos prompt and navigate to the `gcc` subfolder. Type “make” and press enter. The binary and debug files are created as:

```
xmega_applications_xplain_demo_atxmega128a1_xmega_a1_xplained.hex  
xmega_applications_xplain_demo_atxmega128a1_xmega_a1_xplained.elf
```

7.2 IAR Embedded Workbench

To open the project in IAR Embedded Workbench, navigate to the `iar` subfolder and open the project:

```
xmega_applications_xmega_a1_xplained_demo_atxmega128a1_xmega_a1_xplained.eww
```

The project can be compiled by pressing the F7 key on your keyboard or by selecting “make” from the project menu. The binary and debug files are created as:

```
xmega_applications_xmega_a1_xplained_demo_atxmega128a1_xmega_a1_xplained.hex  
xmega_applications_xmega_a1_xplained_demo_atxmega128a1_xmega_a1_xplained.dbg
```

8 References

[www.atmel.com/Xplained]

[www.atmel.com/avrstudio]

8.1 IAR compiler

This is a commercial C/C++ compiler available for AVR 8-bit MCUs. A 30-day evaluation version as well as a 4k (code size limited) kick-start version is available from the IAR website:

IAR Embedded Workbench for Atmel AVR

<http://www.iar.com/website1/1.0.1.0/107/1/>

8.2 AVR Software Framework

This is a large library of drivers and code available for download and free use from the Atmel website:

Atmel AVR Software Framework (ASF)

http://atmel.com/dyn/products/tools_card.asp?tool_id=4192&category_id=163&family_id=607&subfamily_id=2138



9 EVALUATION BOARD/KIT IMPORTANT NOTICE

This evaluation board/kit is intended for use for **FURTHER ENGINEERING, DEVELOPMENT, DEMONSTRATION, OR EVALUATION PURPOSES ONLY**. It is not a finished product, and may not (yet) comply with some or any technical or legal requirements that are applicable to finished products, including, without limitation, directives regarding electromagnetic compatibility, recycling (WEEE), FCC, CE, or UL (except as may be otherwise noted on the board/kit). Atmel supplied this board/kit "AS IS," without any warranties, with all faults, at the buyer's and further users' sole risk. The user assumes all responsibility and liability for proper and safe handling of the goods. Further, the user indemnifies Atmel from all claims arising from the handling or use of the goods. Due to the open construction of the product, it is the user's responsibility to take any and all appropriate precautions with regard to electrostatic discharge and any other technical or legal concerns.

EXCEPT TO THE EXTENT OF THE INDEMNITY SET FORTH ABOVE, NEITHER USER NOR ATMEL SHALL BE LIABLE TO EACH OTHER FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES.

No license is granted under any patent right or other intellectual property right of Atmel covering or relating to any machine, process, or combination in which such Atmel products or services might be or are used.

Mailing Address: Atmel Corporation, 2325 Orchard Parkway, San Jose, CA 95131

Copyright © 2011, Atmel Corporation

10 Table of contents

Features 1

1 Introduction 1

2 What is needed to get started? 2

 2.1 Documentation for XMEGA-A1 Xplained 2

 2.2 AVR Software Framework 2

 2.3 Integrated development environment for Atmel AVR microcontrollers 2

 2.3.1 AVR Studio 4 2

 2.3.2 Embedded Workbench for Atmel AVR 3

 2.4 FLIP (BatchISP) 3

 2.5 Toolchain 3

 2.5.1 AVR GNU Toolchain 3

 2.5.2 IAR AVR Toolchain 3

 2.6 Programmers and debuggers 3

 2.6.1 AVR Dragon 3

 2.6.2 AVR JTAGICE mkII 4

 2.6.3 AVR ONE! 4

3 Board overview 5

4 Connecting the board 6

 4.1 Connecting AVR Dragon 6

 4.2 Connecting AVR JTAGICE mkII 6

 4.3 Connecting AVR ONE! 7

5 Programming the kit 8

 5.1 Programming via the boot loader 8

 5.1.1 Prerequisite 8

 5.1.2 Step-by-step guide 8

 5.2 Restoring the boot loader 10

6 Example applications 11

 6.1 Preprogrammed firmware 11

 6.1.1 Functionality of the preprogrammed firmware 11

 6.2 More example projects and tutorials 11

7 How to compile example projects 12

 7.1 AVR GNU Toolchain 12

 7.2 IAR Embedded Workbench 12

8 References 13

 8.1 IAR compiler 13

 8.2 AVR Software Framework 13

9 EVALUATION BOARD/KIT IMPORTANT NOTICE 14





10 Table of contents 15



Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chou-ku, Tokyo 104-0033
JAPAN
Tel: (+81) 3523-3551
Fax: (+81) 3523-7581

© 2011 Atmel Corporation. All rights reserved. / Rev.: CORP072610

Atmel®, Atmel logo and combinations thereof, AVR®, AVR® logo, AVR Studio®, STK®, XMEGA®, and others are registered trademarks or trademarks of Atmel Corporation or its subsidiaries. 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 product 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.