AVR1005: Getting started with XMEGA

Features

- **XMEGA™ vs. megaAVR®**
- **XMEGA feature and peripherals**
- **Tool-chain - programming and debugging**
- **In-system Programming**
- **Hardware design considerations**

1 Introduction

The AVR® product portfolio has been expanded with the XMEGA family. Many ask if this is a new architecture and are uncertain how their experience with megaAVR transfers to AVR XMEGA. This document briefly introduces the similarities and differences between the two AVR families, and provides an overview of the available tool chain. It targets both experienced AVR users and newbies that want an introduction and to getting started.

The ideas behind the XMEGA actually have their origin from the end users themselves. AVR users have occasionally mentioned that they wanted lower power consumption, stronger peripheral modules, more timers, DMA, multiple interrupt levels, larger memories and more CPU performance. Well, you've got it! In addition you have the new and innovative Event System, Crypto Engines, 12-bit ADC and DAC, real-time calibration of oscillators, CRC check, and other special features.

Further, these new devices scale better in terms of feature set, pin count and memory size. The peripheral types within a family are similar and are organized in the same way for all family members. Porting code from one XMEGA to another has therefore become a trivial task.

Figure 1. ATxmega128A1 - the first XMEGA.

2 XMEGA introduction (and similarities and differences from megaAVR)

The XMEGA is an 8-bit AVR. It uses the AVR CPU and architecture, but with additional features and improved (redesigned) peripheral modules. The XMEGA uses the same instruction set, the same development tools and compilers as any other AVR. Experienced AVR users will therefore be familiar with most of the tool chain. The CPU can run faster than before and at lower voltage. Further, there is added a multi-level interrupt controller, Direct Memory Access Controller and the Event System. The DMA and Event System can be run in IDLE mode (sleep) and together with 1.6V operation and other low power features we are talking about the 2nd generation picoPower™ Technology.

2.1 Common questions about XMEGA

To cover some of the questions asked when designers are introduced to XMEGA:

The XMEGA and megaAVR are not pin compatible. This was not possible to achieve due to the desire to make the device more modular, e.g. by replicating functions and modules on each IO port. To give an example: The UART, TWI and SPI modules are always located on the same pins on a port that has these features. This makes it much easier for the firmware designer to port code between devices, but unfortunately causes pin incompatibility.

XMEGA is operating on supply voltages from 1.6V to 3.6V. This is partly due to the manufacturing process, which is selected from a power consumption point of view. The manufacturing process that XMEGA is produced in offers unmatched low power operation, but is limited to 3.6V operation. 5V operation is thus not possible for the first XMEGA families. Though the manufacturing process is designed for low voltage operation, the lower operation voltage does not mean decreased CPU performance: The XMEGA runs at up to 32MHz already from 2.7V. From 1.6V main clock frequency up to 12MHz is supported.

2.2 Structures and Migration

The XMEGA is structured to easily port between the devices. For instance, PORTA and PORTB will be dedicated the analog interfaces ADC and DAC or two ACs. PORTC, D and E will be for serial interfaces. Also, all modules are defined in structures (`typedef struct`). This means that for instance the registers of the ADC can be accessed by the ADC structure. Further, since the ADC is defined in this structure, the idea is that this structure will be the same for all A family devices. So if you have started with an ATxmega64A4, it should be a quick task to migrate to an ATxmega128A4 or an ATxmega256A1; because you are using the same structure and thus you can use the exact same code too. Why didn't we do this with megaAVR? With megaAVR it is not possible to make one structure of a module to define them all, because the different parts may have different designs. With XMEGA, there is only one design for each module within a device family.

The XMEGA families are distinguished by a letter: the first XMEGA family available is ATxmega[xxx]A[#] ("A" family). Before the family letter you will find a number, which refers to the Flash memory size, and after the family letter you will find the family member ID, that identifies the individual devices. Devices with the same device ID has the exact same feature set and the same number of pins, but can be available in various memory sizes.

2.3 Manual and Datasheets

The structure of the XMEGA makes it possible to make one (large) manual to describe all the modules and peripherals of a device family. The details that will make the parts different from each other, like the typical characteristics and the available peripherals, will be described in the (smaller) device specific data sheets.
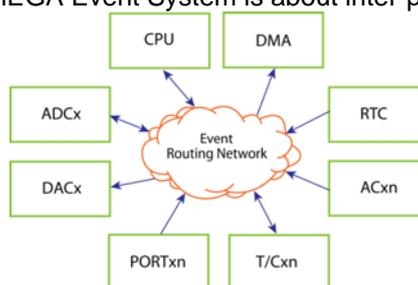
For instance, all functional descriptions could be collected for all A1, A3 and A4 devices in the XMEGA A family. The XMEGA A Manual is thus an in-depth documentation for all modules that are available in the device A family. On the other side, the device-specific datasheet for the sub-family A3 describes the details that are not in common with A1 and A4. This would be typical characteristics, the actual feature set, the number of peripherals available, pin counts and more. The XMEGA manual and datasheets may be downloaded from www.atmel.com/avr -> datasheets.

2.4 Event System

The Event System is a totally new and innovative feature with the XMEGA. You will not find the Event System in the tinyAVR® nor the megaAVR, in fact, not on any other MCU at the time of writing this document.

The Event System is communication between the peripherals. There are eight channels the user can set up in between peripherals. An event channel is not a bus, but a dedicated channel where only one event can be signaled at a time. Since the event is on a dedicated channel, it will be very predictable and fast. Actually, there will be a maximum delay of 2 chip clock cycles. Also, you may have a lot less interrupts in your system. This is because any interrupt also could be an event. With events, the CPU is not involved and there are no interrupt service routines (ISRs). In all, the Event System is extremely powerful since it unloads the CPU and it can save power and increase the performance in the system due to the reduced number of interrupts. In addition the event system can run while the CPU is sleeping in the IDLE sleep mode.

Figure 2. The XMEGA Event System is about inter-peripheral communication

Let's look at some examples. You can for instance use the Event System to chain two Timer/Counters to get one 32-bit Timer/Counter. Or you could use one event to synchronize two modules. For instance, you could use a pin change event to do an ADC conversion and an input capture on the Timer/Counter to get exact time-stamps for each conversion. Yet another example could be to let an event trigger ADC conversions to sample music from a microphone. At conversion complete, the DMA copies the result(s) to SRAM. When the user presses a switch, an event will be generated and the DMA copies further to the DAC to play back a recording – and all this could be done in sleep mode. Please refer to application note "AVR1001: Getting Started with the XMEGA Event System" for more information. (To see how to set up the DMA Controller, please refer to "AVR1304: Using the XMEGA DMA Controller".)

2.5 2nd Generation PicoPower

The XMEGA is now taking the picoPower Technology to the next level. All XMEGA devices are now tested for 1.6V operation. This means that all the peripherals and modules that can run at 3.6V can also run down to 1.6V, or using a power supply of 1.8V with ±10% accuracy. This applies for all the modules, even for Flash, EEPROM and the analog peripherals. A lower voltage will also give lower current consumption and you may discharge the batteries even lower. There are new designs for the watchdog and Brown-Out circuits. The watchdog oscillator runs typically at 1µA. The current consumption for the Brown Out Detector (BOD) is also 1µA when set up with the new 1kHz sampled mode. An external RTC clock can run and allow for a total power consumption of only 650nA in Power Save sleep mode. By combining the internal RC for active mode, you have low current consumption while at sleep, always high accuracy timing and fast start-up time. Of all the low-power features, maybe the greatest would be to reduce the active mode and replace it by running more often in IDLE mode using the DMA and Event System. Going from Active to IDLE mode and running at lower clock speeds could easily reduce the current consumption with more than 50%.

2.6 Clock system

Clock source and frequency is changed run-time in XMEGA, as opposed to megaAVR, where fuses are used to set the clock source and frequency. Both a 32MHz and 2MHz internal RC oscillator is selectable. The 2MHz RC oscillator is the default clock source at start-up. The 2MHz RC oscillator starts oscillating very quickly after startup. This should be seen in contrast to external crystals and ceramic resonators, which need quite some time to start oscillating. By always starting from the 2MHz RC oscillator the device can start executing code, e.g. doing miscellaneous initialization, while the external crystal build up the oscillation amplitude. Once the crystal oscillator is ready for use the clock switch is performed from firmware. The system clock frequency can be changed run-time through the use of the PLL and prescalers.

By using the internal PLL, a clock signal can be multiplied (1x, 2x, 3x, …, 31x) instead of prescaled. You can for instance increase the main clock signal (max 32MHz) or increase the speed of your peripherals, for instance achieve up to 128MHz clock input of the Timer/Counters to get high resolution PWM. A DFLL module is also implemented which can run-time calibrate the internal oscillators. With better accuracy of the internal oscillators also timing demanding applications like serial communication can be done at higher speeds and at lower cost.

Further, the feature "External Clock Source Failure Monitor" can detect if an external clock fails (e.g. physical damage to crystal) and automatically switch to internal 2MHz RC oscillator. The application note "AVR1003: Using the XMEGA Clock System" takes this topic further.

2.7 Multi-Level Interrupt Controller

Since the XMEGA has even more interrupt sources than megaAVR, a multi-level interrupt controller has been added. This makes it easier to handle many interrupt sources in a timely and according to given priority. If for instance a high rate interrupt source is served so often that other interrupts on the same level will not be served, then it could be a good idea to lower the priority level of the high rate interrupt. This way it is possible to avoid the so called starving of an interrupt source. Another way to guarantee that all interrupts will be served is to use Round Robin scheduling. A high

level interrupt source can preempt a lower level priority interrupt service routine. That way there will be a minimal delay until the high priority interrupt service routine is executed. Firmware designers that have not used the new XMEGA multi-level interrupt controller may want to refer to the application note "AVR1305: XMEGA Interrupts and the Programmable Multi-level Interrupt Controller" for more information.

2.8 Leading Analog Features

With the XMEGA we are introducing leading analog features for an 8-bit MCU. Both the Analog to Digital Converter (ADC) and the Digital to Analog Converter (DAC) have 12-bits resolution. The ADC can sample up to 2 mega samples per second and can give 4 (nearly) simultaneous results at a time. Combining two ADCs, 8 simultaneous results can be given. There is also a feature called event or interrupt on conversion complete. This feature will only give interrupts (set the flag) on the conversion complete when the result is below or above a certain limit. So if you are not interested in results above for instance 1V, you will not bother the CPU with them either.

The DAC can give two outputs even when only using one module. This is because it has a separate sample and hold circuit that will for instance allow stereo output with only one DAC. For more information about ADC and DAC, please refer to the application notes "AVR1300: Using the XMEGA ADC" and AVR1301: Using the XMEGA DAC".

The Analog Comparator (AC) has a new feature called window mode when using two AC modules at a time. Then you can input two boundaries (low/high level) and the input signal to compare. The result would then be below, inside or above the window. The result of the AC can also be output on an I/O pin. Please refer to "AVR1302: Using the XMEGA Analog Comparator" for more information and features.

2.9 External Bus Interface

The External Bus Interface available in XMEGA supports much larger memories, up to 16MB. Both SRAM and SDRAM memories are now supported. SDRAM offers lower cost for larger memories, but is not possible to combine with other memory mapped units that require a SRAM-like bus interface. Due to the support for large memories, the instruction timing for instructions that access memories are slightly changed, and further when compiling for large memory models the data types used for addressing is affected, since 24-bit is required rather than 16-bit. Please refer to application note "AVR1312: Using the XMEGA External Bus Interface" for more information.

2.10 Crypto Engine

When doing communication protocols, both for wired and wireless applications, cryptography is needed to keep the transmitted information secure.

XMEGA implements both Advanced Encryption Standard (AES) and Data Encryption Standard (DES), and can perform both encryption and decryption. The AES is a hardware module that will totally unload the CPU. The module can provide 1.25Mbps encrypted communication with a 128-bit key length. Please refer to the application notes "AVR1318: Using the XMEGA built-in AES accelerator" and "AVR1317: Using the XMEGA built-in DES accelerator" for more information and details.

2.11 Special Features

There are also some nice "special features" with the XMEGA. The flash section can for instance be divided into an Application Table area to provide fail safe EEPROM emulation. This means that you can store variables into an area that is separated from the program code in flash. While at memories, user calibration memory for oscillators can for instance be stored in a special memory section. The section is fully accessible (read and write) from the application and an external programming interface. But the section will not be affected by chip erase, something that would very convenient during a debugging session. Another feature is the serial numbers. All XMEGA parts will have their own unique ID. A unique device signature is useful for random number seeding in addition to for instance identifying nodes in a network. Also it is possible to use the XMEGA to calculate CRC of the flash to check consistency during a life-cycle and ensuring correct result after updates.

3 Where to start when writing firmware for XMEGA?

The XMEGA is a new platform based on the AVR CPU, with new peripheral modules designed to ease the use and boost the performance. This means that the peripheral drivers for the megaAVR cannot be reused on XMEGA. To minimize the effort for the firmware designers, Atmel® has provided a full set of drivers and application notes that will guide the designer that use the new features and modules.

You would want to start at the beginning and learn how we have implemented the drivers. As for much other AVR related information, it is described in an application note. The application note "AVR1000: Getting Started Writing C-code for XMEGA" is an introduction to the coding style that has been chosen and also describes how the peripheral modules are accessed; Assess of peripheral modules have become "object oriented", which opens new options for reuse and portability. In addition to the application notes, you may ask your local FAE for upcoming XMEGA trainings.

You should now be equipped to try out the various peripheral modules. Application notes are available for most features and peripheral modules. Please refer to the Atmel web site www.atmel.com/xmega to find the complete list of available application notes.

4 XMEGA tool-chain (and "what tools can be used?")

The tool chain for XMEGA is the same as for megaAVR. You can use The Embedded Workbench® AVR compiler from IAR® (version 5.x) or the free GCC compiler WinAVR (revision 20080610 or later; http://winavr.sourceforge.net/). Both come with the header files with names and addresses for registers and bits needed to access the peripherals. All code distributed with the XMEGA application notes compiles both with IAR and GCC.

All firmware for AVR tools are provided with AVR Studio® . For best performance when using XMEGA and AVR Studio, version 4.15 build 623 or later is recommended.

To get the XMEGA up and running you can use the Starter Kit STK®600. The STK600 is similar to the STK500, but supports a wider range of devices from tinyAVR, megaAVR to AVR32 UC3 and including XMEGA devices. To getting started using the STK600 and XMEGA, please refer to application note "AVR1900: Getting started with ATxmega128A1 on STK600".

The STK600 comes with several socket boards and router boards; the ones needed for the ATxmega128A1 is the STK600-TQFP100 and the routing board STK600-RC100X-13. This setup is shown in Figure 3. Please refer to the AVR Studio® online help to determine which device and router boards other XMEGAs require.

Figure 3. STK600 – breakout, clock generator, programmer but no debugging.

To program the XMEGA the on-board programmer on the STK600 can be used. Alternatively, the JTAGICE mkII can be used. The AVR Studio simulator offers a debugging environment that can be used if hardware is not available. The code can run in the AVR Studio simulator as if programmed into a real device. If you want to debug on the actual hardware (silicon), an AVR Dragon, JTAGICE mkII or AVRONE! is required.

The Dragon, AVR ONE! And JTAGICE mkII can be used both with AVR Studio as front-end or from IAR C-SPY®, which is part of the Embedded Workbench AVR suit from IAR.

Figure 4. JTAGICE mkII also functions as an on-chip debugger. Please note that your JTAGICE mkII should have a serial number that starts with "B0" or be marked with "A09-0041" for PDI to work.

As another alternative, the AVRISP® mkII can be used to program the XMEGA. The AVRISP mkII is a programmer only (no on-chip debugging) that also supports all tinyAVRs, megaAVRs and XMEGAs. To use AVRISP mkII with XMEGA, the AVR Studio version 4.15 build 623 or later is required.

There are more details about programming in chapter 5.

Figure 5. AVRISP mkII supports PDI interface to do programming only.

The AVR® DRAGON™ can as of AVR Studio version 4.18 be used to program and debug Tiny, Mega and XMEGA devices. It supports JTAG programming and debugging for all XMEGA devices with JTAG interface, but so far only supports PDI interface for the A4 subfamily.

Figure 6. AVR DRAGON™ supports debugging and programming via JTAG / PDI.

The most versatile alternative, AVR ONE! supports both JTAG and PDI programming for all XMEGA devices, as well as the new liveDebug feature which allows you to attach to a running host and start debugging to see what it was doing without having to reset the device. The AVR ONE! also supports higher clock frequencies than eg. JTAGICE mk II.
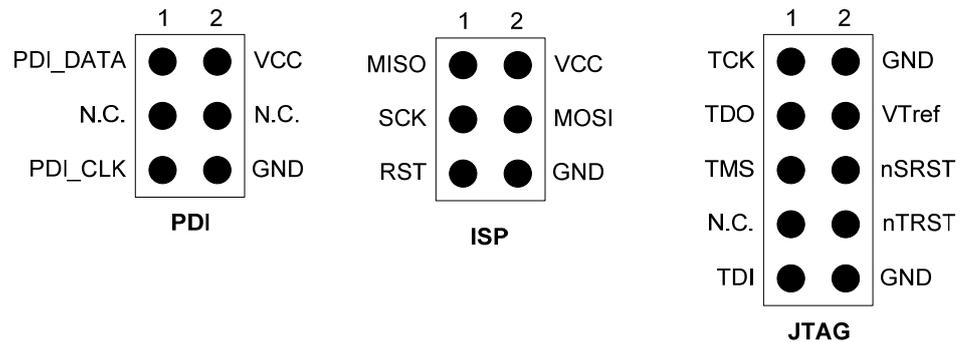
Figure 7. AVR ONE! supports liveDebug as well as faster PDI and JTAG interfaces.

5 In-system programming and debugging the XMEGA ("What's PDI?")

The XMEGA doesn't have the SPI based In-System Programming (ISP) interface for external programming, which has been used for megaAVR. Nor does it have the debugWIRE interface. These have been replaced by a two wire "Programming and Debugging Interface" (PDI). This interface can, as the name indicates, be used for both programming and debugging.

Figure 8. PDI, ISP and JTAG connect – Atmel Tools standard.

Programming and Debugging through the JTAG interface is still possible. The JTAGICE mkII can do both PDI and JTAG programming and debugging. Please note that your JTAGICE mkII should have a serial number that starts with "B0" or be marked with "A09-0041" for PDI to work.

Figure 9. Hardware revision 0 of JTAGICE mkII starts with A0 and does not support PDI

Figure 10. Hardware revision 1 has serial numbers with B0 and A09-0041. Supports PDI.

Programming through the PDI interface is also possible using the Dragon, AVRISP mkII and the existing 6-pins interface.

Please refer to the AVR Studio online help and the XMEGA A Manual for updated and more detailed information.

5.1 Hardware design requirements to make PDI work

The PDI interface is a synchronous half-duplex UART interface. The two lines, PDI_DATA and PDI_CLK, must therefore be balanced. If you place a strong pull-up and decoupling cap on the PDI_CLK, which is also the Reset line, the clock and data will no longer be synchronized correctly. Therefore, during development you should remove any pull-up and decoupling capacitors. This also applies if using the PDI interface for in-system programming the XMEGA in production.

6 Hardware design considerations ("Do all Vcc pins need decoupling caps?")

The XMEGA is capable of running at higher clock frequencies than the megaAVR, and to ensure that the supply is stable and to reduce emissions it is recommended to place decoupling caps on all supply pins – 100nF is a good starting point. All decoupling caps must be connected to the Vcc bus externally; don't use the XMEGA's internal Vcc distribution to charge the decoupling caps.

If you run at low clock frequencies you might be able to omit/share decoupling caps on some supply pins, but the general recommendation still applies. Test it if you are in doubt.

Headquarters

International

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

Atmel Asia
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
Hong Kong
Tel: (852) 2245-6100
Fax: (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
Tel: (33) 1-30-60-70-00
Fax: (33) 1-30-60-71-11

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Product Contact

Web Site
http://www.atmel.com/

Technical Support
avr@atmel.com

Sales Contact
www.atmel.com/contacts

Literature Request
www.atmel.com/literature

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 ATMEL'S TERMS AND CONDITIONS OF SALE LOCATED ON ATMEL'S WEB SITE, 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 OF 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's products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

© **2009 Atmel Corporation. All rights reserved**. Atmel®, logo and combinations thereof, AVR®, AVR Studio®, STK®, megaAVR®, tinyAVR®, and others, are registered trademarks. XMEGA™ and others are trademarks of Atmel Corporation or its subsidiaries. Other terms and product names may be trademarks of others.

8169B-AVR-11/09