Atmel AVR1629: XMEGA ADC Oversampling

8-bit Atmel Microcontrollers

Application Note

Features

- Increasing the Atmel® AVR® XMEGA® ADC resolution by oversampling
- Averaging and decimation
- Software has been implemented as Atmel AVR Studio® ASF example project for the Atmel XMEGA-A3BU Xplained kit
- Configuration option in source code:
  - To select number of additional bit (resolution) to be achieved by oversampling
  - To select ADC configuration like ADC input pin, ADC REF Source, REF Voltage
- Results are displayed on LCD available in the XMEGA-A3BU Xplained kit:
  - Raw ADC count and calculated analog input voltage (in volt) are displayed
  - For comparison, both oversampled and normal results are displayed

1 Introduction

The XMEGA controller offers an analog to digital converter with 12-bit resolution. In most cases 12-bit resolution is sufficient, but in some cases higher accuracy is desired. Special signal processing techniques can be used to improve the resolution of the measurement. By using a method called 'Oversampling and Decimation' higher resolution might be achieved, without using an external ADC. For example by using 12-bit XMEGA ADC, a 16-bit result could be achieved with oversampling technique. This application note explains the method, and conditions needed to be fulfilled to make this method work properly. This application note also provides source code as per explained theory to achieve this oversampling technique.

Rev. 8498A-AVR-03/12

2 Theory of operation

This chapter explains how oversampling works with all necessary mathematical details.

2.1 Sampling frequency

The Nyquist Theorem states that a signal must be sampled at least twice as fast as the bandwidth of the signal to accurately reconstruct the waveform; otherwise, the high-frequency content will alias at a frequency inside the spectrum of interest (pass band). The minimum required sampling frequency, in accordance to the Nyquist Theorem, is the Nyquist frequency.

Equation 2-1. The Nyquist frequency.

$$f_{nyquist} > 2 \cdot f_{signal}$$

where f_{signal} is the highest frequency of interest in the input signal. Sampling frequencies above $f_{nyquist}$ are called 'oversampling'. This sampling frequency, however, is just a theoretical absolute minimum sampling frequency. In practice the user usually wishes the highest possible sampling frequency, to give the best possible representation of the measured signal, in time domain. One could say that in most cases the input signal is already oversampled.

The sampling frequency is a result of prescaling the CPU clock; a lower prescaling factor gives a higher ADC clock frequency. At a certain point, a higher ADC clock will decrease the accuracy of the conversion as the Effective Number of Bits, ENOB, will decrease. All ADCs has bandwidth limitations. For Atmel XMEGA A series devices, to get a 12-bits resolution on the conversion result, the ADC clock frequency should be maximum 2MHz. When the ADC clock is 2MHz, the sampling frequency is 2Msps, which confines the upper frequency in the sampled signal to ~1MHz.

2.2 Oversampling and decimation

The oversampling technique requires a higher amount of samples. These extra samples can be achieved by oversampling the signal. For each additional bit of resolution, n, the signal must be oversampled four times. In which frequency the signal to be sampled with is given by Equation 2-2. To get the best possible representation of an analog input signal, it is necessary to oversample the signal this much, because a larger amount of samples will give a better representation of the input signal, when averaged. This is to be considered as the main ingredient of this application note, and will be further explained by the following theory and examples.

Equation 2-2. Oversampling frequency.

$$f_{oversampling} = 4^n \cdot f_{nyquist}$$

2.3 Noise

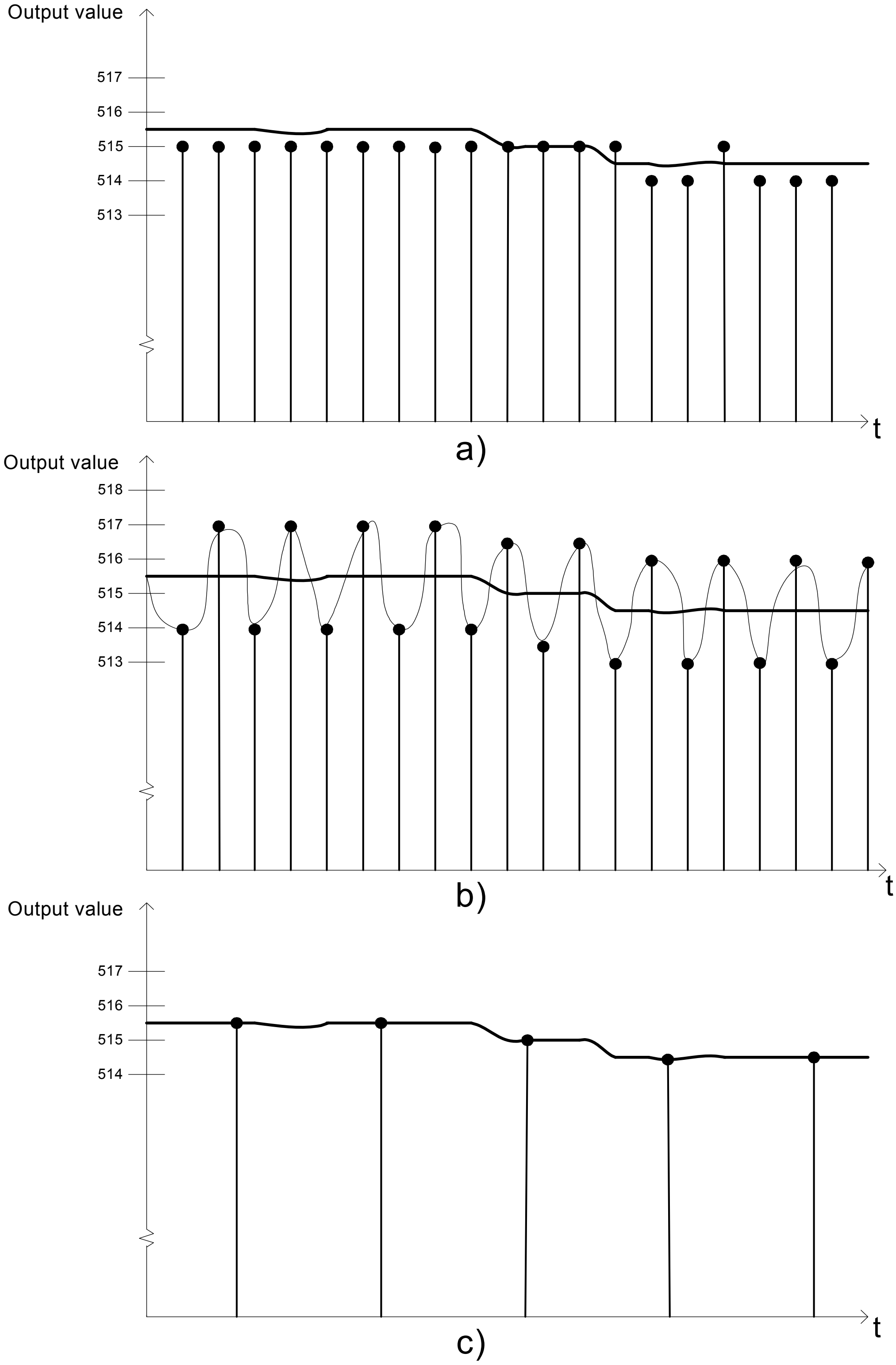
To make this method work properly, the signal-component of interest should not vary during a conversion. However another criterion for a successful enhancement of the resolution is that the input signal has to vary when sampled. This may look like a contradiction, but in this case variation means just a few LSB. The variation should be seen as the noise-component of the signal. When oversampling a signal, there should be noise present to satisfy this demand of small variations in the signal. The quantization error of the ADC is at least 0.5 LSB. Therefore, the noise amplitude has to exceed 0.5 LSB to toggle the LSB. Noise amplitude of 1-2 LSB is even better because this will ensure that several samples do not end up getting the same value.

Criteria for noise, when using the decimation technique:

- The signal-component of interest should not vary significantly during a conversion
- There should be some noise present in the signal
- The amplitude of the noise should be at least 1 LSB

Normally there will be some noise present during a conversion. The noise can be thermal noise, noise from the CPU core, switching of I/O-ports, variations in the power supply and others. This noise will in most cases be enough to make this method work. In specific cases though, it might be necessary to add some artificial noise to the input signal. This method is refereed to as dithering. Figure 2-1 (a) shows the problem of measuring a signal with a voltage value that is between two quantization steps. Averaging four samples would not help, since the same low value would be the result. It may only help to attenuate signal fluctuation. Figure 2-1 (b) shows that by adding some artificial noise to the input signal, the LSB of the conversion result will toggle. Adding four of these samples halves the quantization steps, producing results that gives better representations of the input value, as shown in Figure 2-1 (c). The ADCs ‘virtual resolution’ has increased from 10 to 11 bits. This method is refereed to as Decimation and will be explained further in Section 2.4.

Figure 2-1. Increasing the resolution from 10-bit to 11-bit.

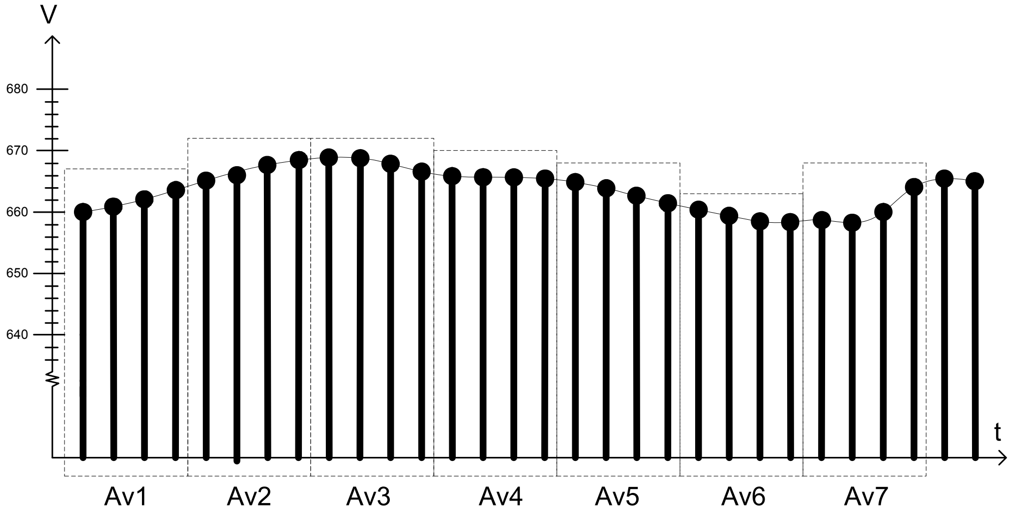


Another reason to use this method is to increase the signal-to-noise ratio. Enhancing the Effective Number of Bits, ENOB, will spread the noise over a greater binary number. The noise' influence on each binary digit will decrease. Doubling the sampling frequency will lower the in-band noise by 3dB, and increase the resolution of the measurement by 0.5 bits.

2.4 Averaging

The conventional meaning of averaging is adding *m* samples, and dividing the result by *m*, which is refereed to as normal averaging. Averaging data from an ADC measurement is equivalent to a low-pass filter and has the advantage of attenuating signal fluctuation or noise, and flatten out peaks in the input signal. The Moving Average method is very often used to do this. It means taking *m* readings, place them in a cyclic queue and average the most recent *m*. This will give a slight time delay, because each sample is a representation of the last m samples. This can be done with or without overlapping windows. Figure 2-2 shows seven (Av1-Av7), independently Moving Average results without overlapping.

Figure 2-2. Moving Average principle.



It is important to remember that normal averaging does not increase the resolution of the conversion. Decimation, or interpolation, is the averaging method, which combined with oversampling, increases the resolution.

The extra samples, *m*, achieved by oversampling the signal are added, just as in normal averaging, but the result are not divided by *m* as in normal averaging. Instead the result is right shifted by *n*, where *n* is the desired extra bit of resolution, to scale the answer correctly. Right shifting a binary number once is equal to dividing the binary number by a factor of 2.

As seen from Equation 2-2, increasing the resolution from 12- to 16-bits (that is, additional 4-bit resolution), requires the summation of 4^4 (256) 12-bit values. A sum of 256 12-bit values generates a 20-bit result where the last four bits are not expected to hold valuable information.

To get 'back' to 16-bit representation, it is necessary to scale the result. The scale factor, *sf*, given by Equation 2-3, is the factor, which the sum of 4^n samples should be divided by, to scale the result properly. *n* is the desired number of extra bit.

Equation 2-3.

$$sf = 2^n$$

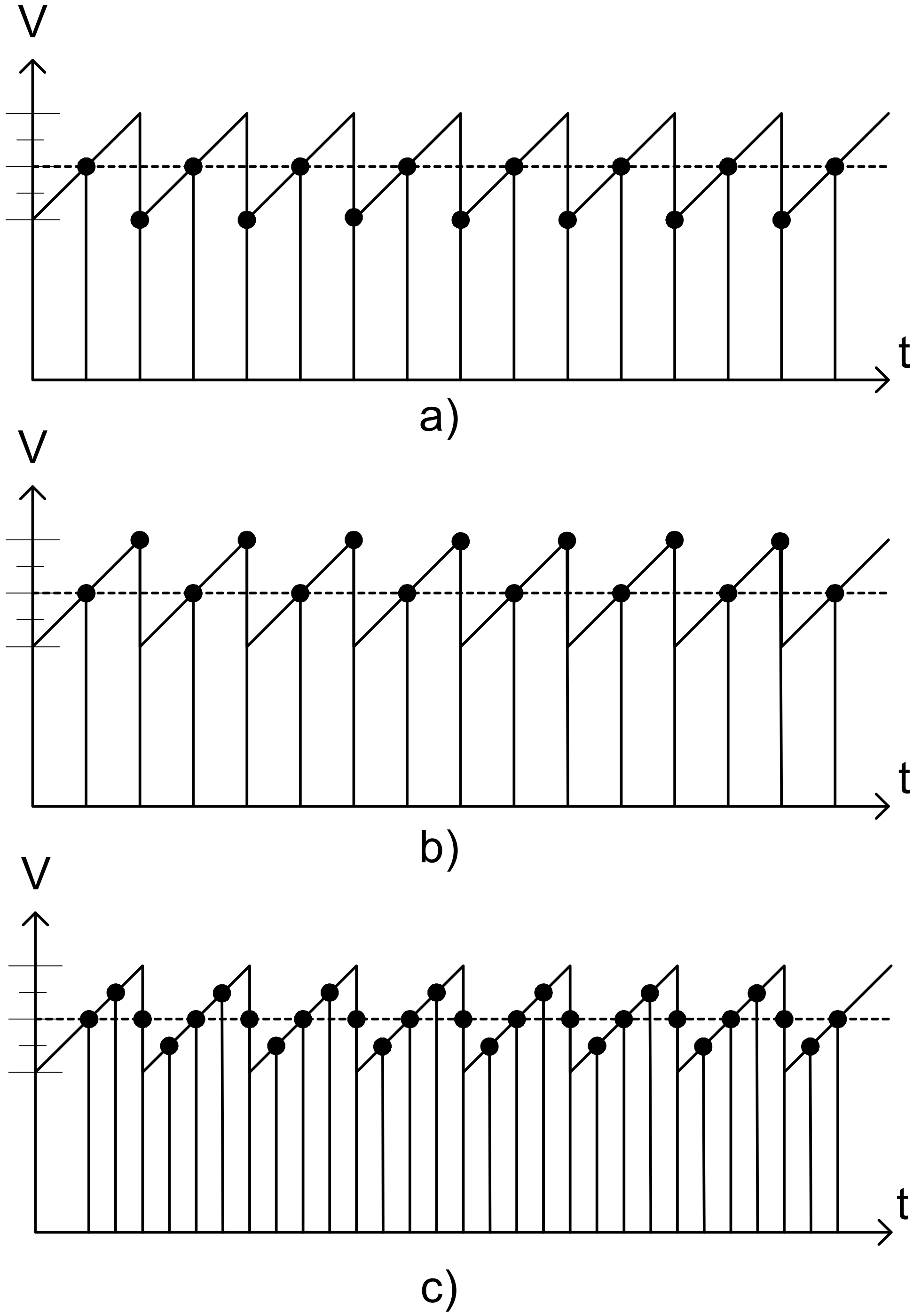
As explained in the case above (increasing resolution from 12-bit to 16-bit), the scaling factor, *sf*, is 2^4 which is equal to 16.

2.5 When will 'Oversampling and Decimation' work?

Normally a signal contains some noise, this noise very often has the characteristic of Gaussian noise, more commonly known as white noise or thermal noise, recognized by the wide frequency spectrum and that the total energy is equally divided over the entire frequency range. In these cases the method of 'Oversampling and decimation' will work, if the amplitude of the noise is sufficient to toggle the LSB of the ADC conversion.

In other cases it might be necessary to add artificial noise signal to the input signal, this method is referred to as dithering. The waveform of this noise should be Gaussian noise, but a periodical waveform will also work. What frequency this noise signal should have depends on the sampling frequency. A rule of thumb is: "When adding m samples, the noise signals period should not exceed the period of m samples". The amplitude of the noise should be at least 1 LSB. When adding artificial noise to a signal, it is important to remember that noise has a mean value of zero; insufficient oversampling therefore may cause an offset, as shown in Figure 2-3.

Figure 2-3. Offset caused by insufficient sampling.



The stippled line illustrates the averaged value of the saw-tooth signal. Figure 2-3 (a) will cause a negative offset. Figure 2-3 (b) will cause a positive offset. In Figure 2-3 (c) the sampling is sufficient, and offset is avoided. To create an artificial noise signal, one of the AVRs counters can be used. Since the counter and the ADC use the same clock source, this gives the possibility of synchronizing the noise and the sampling frequencies to avoid offset.

3 Source code

The oversampling software has been implemented as an example project in Atmel AVR Studio ASF. The ASF project name is ADC Oversampling Demo application for Atmel XMEGA-A3BU Xplained.

This chapter explains how the oversampling demo application works and also how different configuration parameters should be changed to obtain different oversampling levels.

ADC offset error and gain error correction has been taken care of in source code. The user has to take care about configuration of correct gain error value, which will vary from device to device. Details have been given in the configuration section.

For complete details on variables, functions, macros etc. used in software, the user can go through ASF help for this demo application.

3.1 How the oversampling demo project works

The oversampling demo project has been prepared and tested for the XMEGA-A3BU Xplained kit. You can refer Atmel application note "Atmel AVR1923: XMEGA-A3BU Xplained Hardware User Guide" for more details about XMEGA-A3BU Xplained kit.

ADCB from the target device Atmel ATXmega256A3BU has been used for sampling the input signal and this ADC has been configured in differential, signed, 12-bit resolution, 250Ksps and external reference on AREFB pin.

During ADC initialization, ADC offset error will be calculated with similar ADC configuration, which will be used for input signal sampling. By using this offset value, offset error correction will be done on each sample read in runtime. Here the user has to note that while powering the target, reference voltage should be present on the AREFB pin. Otherwise, measured reading will not be accurate because of wrong offset error calculation. If this happens, the user can unplug and plug the power to target board by keeping reference voltage AREFB pin.

After offset error correction, ADCB will be configured in free running mode to sample input signal and start oversampling process.

3.1.1 Oversampling configuration

Refer the header file 'conf_oversampling.h', under src/config folder from oversampling demo ASF project.

Here you will be able to select what level of oversampling to be used. For example: increasing from 12-bit resolution to 16-bit resolution or 12-bit to 14-bit etc. Regarding details on how to change each configuration item has been given in the source code itself as comment against each configuration parameters.

Configuration option has been provided for:

- Analog pin will be used for feeding positive input of external analog signal. For example PIN1 (that is, PB1 for ADCB)
- Analog pin will used for feeding negative input of external analog signal. For example PIN2 (that is, PB2 for ADCB)
- Reference voltage source
- Reference voltage value in microvolt

3.1.2 How to start working with the demo project

To start working with the oversampling demo project in default configuration made in 'conf_oversampling.h', you have to make the hardware connections in Atmel XMEGA-A3BU Xplained kit as described here:

- Connect the 2.5V reference voltage to Pin 1 of header J2 (marked as ADC0)
- Connect the positive input of the external analog signal (which has to be measured in differential mode) to Pin 2 of header J2 (marked as ADC1)
- Connect the negative input of the external analog signal to Pin 3 of header J2 (marked as ADC2)

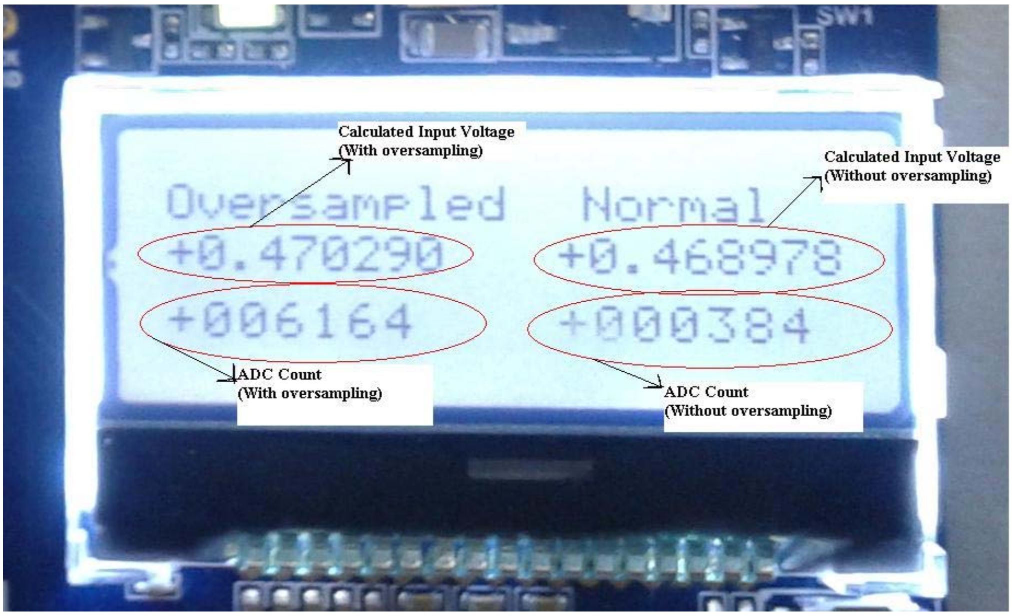
After making the necessary hardware setup, you can build and download the hex file of the Atmel XMEGA ADC oversampling demo project from ASF.

3.1.3 Results display on LCD

The results will be displayed on the LCD available with the XMEGA-A3BU Xplained kit. For better understanding and comparison, oversampled result and also result with single sample (that is, without oversampling) will be displayed on the LCD. Both ADC count and calculated analog input voltage (in volt) will be displayed.

A screenshot of the LCD display is shown in Figure 3-1. In Figure 3-1, it has been marked to explain what different content in the display is.

Figure 3-1. Screenshot of the LCD display.



4 Recommended reading

It is recommended to read the following application notes to get to know more on the Atmel XMEGA ADC and oversampling theory.

Below here there are listed application notes and other XMEGA related application notes with source code, which are available from the Atmel website link; http://atmel.com/dyn/products/product_docs.asp?category_id=163&family_id=607&subfamily_id=1965&part_id=4308

- AVR120: Characterization and Calibration of the ADC on an AVR. This application note explains various ADC characterization parameters given in the datasheets and how they effect ADC measurements
- AVR1300: Using the Atmel AVR XMEGA ADC. This application note describes the basic functionality of the XMEGA ADC with code examples to get up and running quickly
- AVR1505: XMEGA training - ADC. This application note is a training document on how to use ADC from AVR Xplain evaluation kit which features the Atmel ATXMEGA128A1 microcontroller examples to get up and running quickly
- Atmel AVR042: AVR Hardware Design Considerations. This application note covers most of the problems encountered with power supply design and other physical design problems. This application note is available from link; http://atmel.com/dyn/resources/prod_documents/doc2521.pdf
- AVR1923: XMEGA-A3BU Xplained Hardware User Guide. This is a hardware user guide to start work with XMEGA-A3BU Xplained kit. This application note is available from link; http://www.atmel.com/tools/XMEGA-A3BUXPLAINED.aspx?tab=documents

5 Resources

- Atmel XMEGA manual and datasheets
    - http://www.atmel.com/xmega
- Atmel AVR Studio 5
    - http://www.atmel.com/avrstudio
- IAR Embedded Workbench® compiler
    - http://www.iar.com/

6 Atmel technical support center

Atmel has several support channels available:

- Web portal: http://support.atmel.no/ All Atmel microcontrollers
- Email: avr@atmel.com All Atmel AVR products
- Email: avr32@atmel.com All 32-bit AVR products

Please register on the web portal to gain access to the following services:

- Access to a rich FAQ database
- Easy submission of technical support requests
- History of all your past support requests
- Register to receive Atmel microcontrollers' newsletters
- Get information about available trainings and training material

7 Table of contents

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, Milennium 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
16F, Shin Osaki Kangyo Bldg.
1-6-4 Osaki Shinagawa-ku
Tokyo 104-0032
JAPAN
Tel: (+81) 3-6417-0300
Fax: (+81) 3-6417-0370