AN2538

AVR126: ADC of megaAVR® in Single-Ended Mode

Introduction

Microchip megaAVR® devices have a successive approximation Analog-to-Digital Converter (ADC) capable of conversion rates up to 15 ksps with a resolution of 10 bits. It features a flexible multiplexer, which allows the ADC to measure the voltage at multiple single-ended input pins and an internal channel from the bandgap reference in the device. Single-ended input channels are referred to ground.

This application note describes the basic functionality of the ADC in Microchip megaAVR devices in single-ended mode with code examples on Microchip ATmega88 to get started. The code examples are written in 'C' language and have been tested on the Microchip STK®600 starter kit for functionality.

Features

- Up to 10 bits resolution
- Up to 76.9 ksps for ATmega88
- Up to 15 ksps at maximum resolution
- Auto-triggered and single conversion mode
- Optional left adjustment for ADC result read out
- Sleep mode noise canceler
- Driver source code included for ATmega88:
  - ATmega88 ADC - Single conversion mode
  - ATmega88 ADC - Free-running mode and conversion complete interrupt
  - ATmega88 ADC - Auto triggering using Timer0 compare event as trigger source
  - ATmega88 ADC - Measurement of bandgap reference voltage
  - ATmega88 ADC - Noise Reduction Sleep

Table of Contents

1. Module Overview

This chapter provides an overview of the functionality and basic configuration options of the ADC. Getting Started describes the basic steps to configure and run the ADC module with register description details.

1.1 ADC Operation

To make use of the ADC, the Power Reduction ADC bit in the Power Reduction register (PRR.PRADC) must be written to '0'. The ADC module must be disabled before PRR.PRADC can be written to '0'. The ADC module converts the analog input voltage to a 10-bit digital value. The minimum value represents GND and the maximum value denotes the reference voltage used. The reference voltage is chosen by the Reference Selection bit group in the ADC Multiplexer Selection register (ADMUX.REFS).

The analog input channel for conversion is selected by writing the appropriate value to the Analog Channel Selection bit group in ADMUX (ADMUX.MUX). This includes the ADC input pins along with internal voltage from the temperature sensor, GND, and the fixed bandgap reference voltage. To enable the ADC, the ADC Enable bit in the ADC Control and Status A register (ADCSR.ADEN) must be written to '1'. The channels selected for conversion will not go into effect until ADCSR.ADEN is written to '1'. Before entering Sleep mode, the ADC module can be disabled by writing ADCSR.ADEN to '0'. This reduces the power consumption caused by ADC.

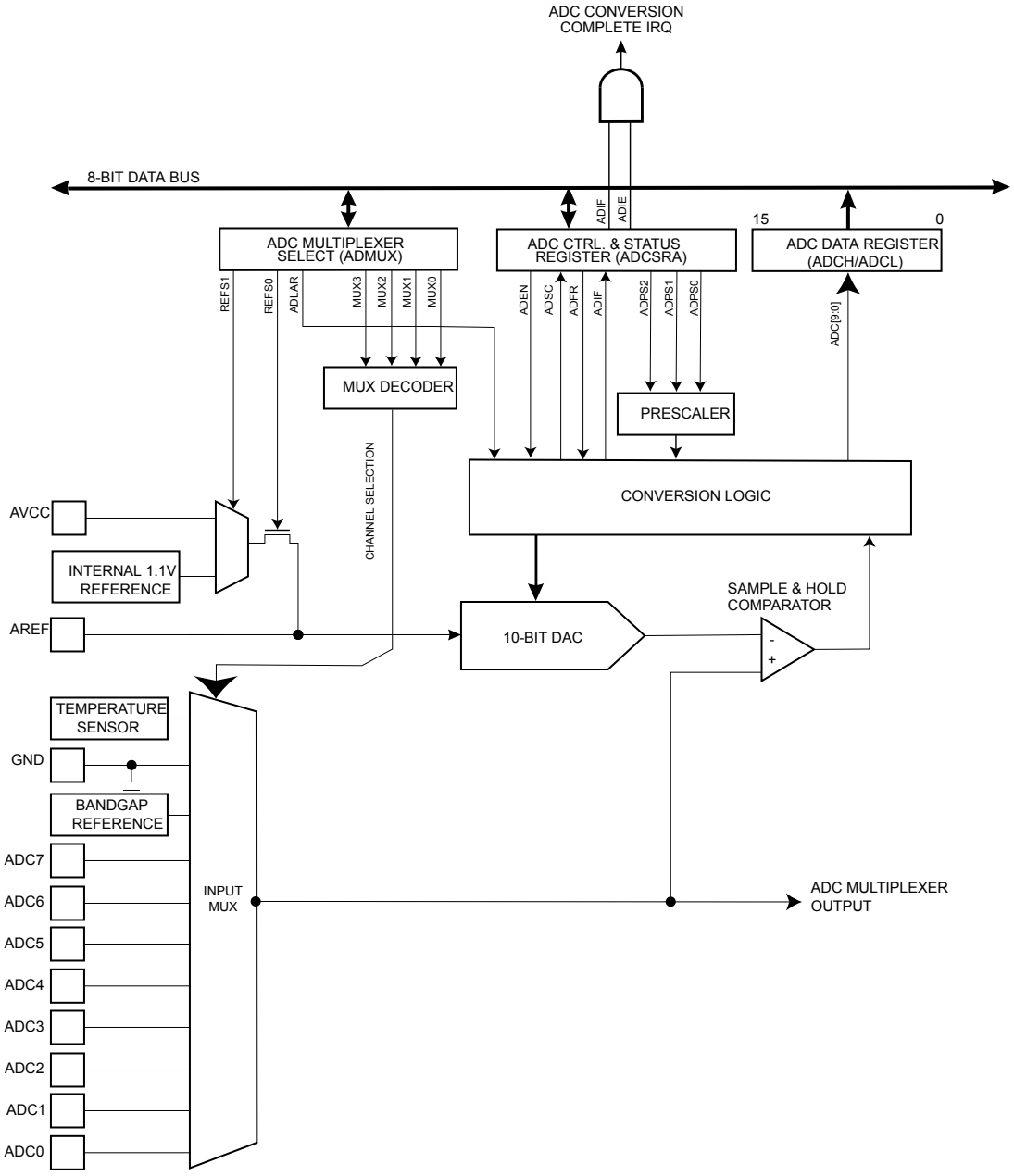
Note: Refer to the device data sheet for details on voltage references and input channels available for the different megaAVR devices.

The 10-bit digital value after conversion is stored in ADCH and ADCL. ADCH holds the higher byte and ADCL holds the lower byte. Optionally, left adjustment of the result can be done by setting the ADLAR bit in the ADMUX register if necessary. If ADLAR is enabled and the application needs only 8-bit accuracy then ADCH alone can be read. Otherwise, ADCL must be read first followed by ADCH, to ensure that the content of the Data Registers belongs to the same conversion. Access to ADC is blocked once ADCL is read. It is re-enabled only after ADCH is read.

The ADC module has one interrupt, which is triggered once a conversion is complete. If an interrupt occurs between reading ADCL and ADCH, it will get triggered and the result will be lost.

The figure below shows the block diagram of the ADC in the megaAVR devices.
Note: Dependent on the feature set of the different megaAVR devices, there might be some variations to the block diagram. Refer to the device data sheet for further information.

Figure 1-1. Analog to Digital Converter Block Diagram



1.2 Input Source

The input sources for the ADC are the analog voltage inputs that the ADC can measure and convert. Two types of measurements can be selected:

- Single-ended input
- Internal input

1.2.1 Single-Ended Input

For single-ended measurements all analog input pins can be used as inputs. All single-ended channels are referred to GND. The analog input voltages cannot be higher than the reference voltage selected for ADC.

1.2.2 Internal Inputs

Two internal analog signals can be selected as input and measured by the ADC:

- Temperature sensor
- Bandgap voltage

The voltage output from an internal temperature reference can be measured with the ADC, and the voltage output will give an ADC result presenting the current temperature in the microcontroller.

The bandgap voltage is an accurate voltage reference inside the microcontroller that is the source for other internal voltage references.

1.3 Starting a Conversion

In single conversion mode, for starting a conversion, the ADC Start Conversion bit in the ADC Control and Status A register (ADCSRA.ADSC) must be written to '1'. This bit remains set until the conversion is completed. Once the conversion is completed the hardware will clear ADCSRA.ADSC.

In the auto triggered mode, the conversion is automatically triggered by various sources. To enable auto triggering, the ADC Auto Trigger Enable bit in ADCSRA (ADCSRA.ADATE) must be written to '1'. The trigger source is selected by writing the ADC Trigger Select bit group in the ADC Control and Status B register (ADCSRB.ADTS) accordingly. The auto triggering mode provides a method of starting conversions at fixed intervals, which are configurable, based on the trigger source.

The ADC Interrupt Flag bit in ADCSRA (ADCSRA.ADIF) will be set even if the specific interrupt or global interrupts are disabled. Thus a conversion can be triggered using ADCSRA.ADIF without causing an interrupt. Then the ADC operates in a free-running mode, in which the next conversion is triggered once the previous conversion completes and sets ADCSRA.ADIF.

Note: In the auto triggered mode, ADCSRA.ADIF must be manually cleared for the next event to generate an interrupt. For free-running mode, the ADC will perform successive conversions independent of whether ADCSRA.ADIF is cleared or not. The first conversion must be started by writing ADCSRA.ADSC to '1'.

1.4 ADC Clock and Conversion Timing

The ADC can prescale the system clock to provide an ADC clock that is between 50 kHz and 200 kHz to get maximum resolution. If an ADC resolution less than 10 bits is required, the ADC clock frequency can be higher than 200 kHz, but it is not recommended to use an ADC clock with a frequency higher than 1 MHz. At 1 MHz we can expect maximum 8 bits of resolution.

The prescaler value is selected by writing the ADC Prescaler Select bit group in the ADC Control and Status A register (ADCSRA.ADPS) accordingly. When initiating a single-ended conversion by writing the ADC Start Conversion bit in ADCSRA (ADCSRA.ADSC), the conversion starts at the following rising edge of the ADC clock cycle.

The first conversion after the ADC is enabled (ADCSRA.ADEN=1) takes 25 ADC clock cycles in order to initialize the analog circuitry. Then, for further conversions, it takes 13 ADC clock cycles (13.5 for Auto triggered conversions). When bandgap voltage is used as input to ADC it will take a certain time for the voltage to stabilize. The start-up time for the bandgap reference voltage is available in the data sheet section *System and reset characteristics*.

1.5 Changing Channel or Reference Selection

The Analog Channel Selection and Reference Selection bit groups in the ADC Multiplexer Selection register (ADMUX.MUX and ADMUX.REFS) are buffered through a temporary register to which the CPU has random access.

If auto trigger mode is used, then ADMUX can be safely updated:

- when the ADC Auto Trigger bit or ADC Enable bit in the ADC Control and Status A register (ADCSRA.ADATE or ADCSRA.ADEN) is cleared
- or, during conversion, minimum one ADC clock cycle after the trigger event
- or, after a conversion, before the ADC Interrupt Flag bit in ADCSRA (ADCSRA.ADIF) used as trigger source is cleared

By doing this, the new settings will affect the next ADC conversion.

In single conversion mode, the channel must be selected before starting the conversion. It is recommended to wait until the conversion completes before changing the channel, which will take effect one clock cycle after writing the ADC Start Conversion bit in ADCSRA (ADCSRA.ADSC) to '1'.

In free-running mode, the channel must be selected before starting the conversion. However, it is recommended to wait until the first conversion is completed before changing the channel, which will take effect one clock cycle after writing the ADC Start Conversion bit in ADCSRA (ADCSRA.ADSC) to '1'. But, since the next conversion has already started automatically, the changes will be reflected in the next following conversion.

1.6 ADC Noise Canceler

The ADC of the megaAVR has a noise canceler that enables conversion during sleep mode, which reduces the noise induced from the CPU core and other peripherals. This feature is available in the ADC Noise Reduction and Idle mode. To use this feature:

- Ensure that ADC is enabled and not busy converting. Single conversion mode must be selected and the ADC conversion complete interrupt must be enabled.
- Enter ADC noise reduction or idle mode. The ADC will start a conversion once the CPU has been halted.
- If no other interrupts occur before the ADC conversion completes, the ADC interrupt will wake-up the CPU and execute the ADC conversion complete interrupt routine. If another interrupt wakes up the CPU before the ADC conversion is complete, that interrupt will be executed, and an ADC conversion complete interrupt request will be generated when the ADC conversion completes. The CPU will remain in active mode until a new sleep command is executed.

1.7 Conversion Result

Once the ADC completes a conversion, the ADC Interrupt Flag bit in the ADC Control and Status A register (ADCSRA.ADIF) will be set to '1' and the 10-bit result will be available in the ADCH and ADCL registers.

For single conversion, the result is:

$$ADC = \frac{V_{IN}.1024}{V_{REF}}$$

where V_{IN} represents the analog input voltage and V_{REF} represents the selected reference voltage. '0x000' represents GND and '0x3FF' represents the selected reference voltage minus one LSB.

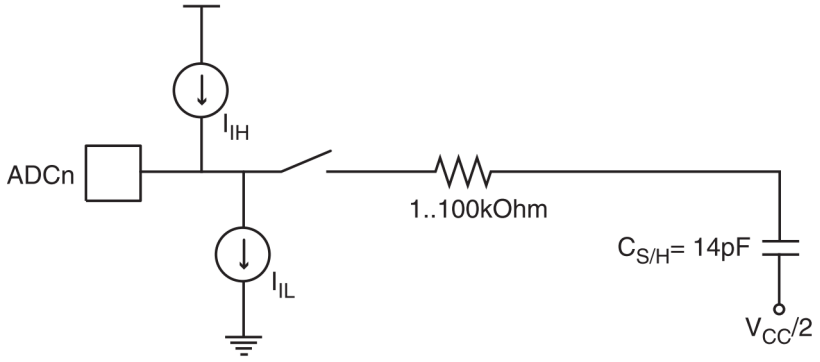
1.8 Analog Input Circuitry

The analog input circuitry for single-ended channels is shown in the figure below. An analog source applied to the ADC input pin is subjected to pin capacitance and input leakage of that pin, even if it is not selected as input for the ADC. When a particular channel is selected, it should drive the sample and hold (S/H) capacitor through the series resistance, which is the combined resistance of the input path.

The ADC module is optimized for analog signals with an output impedance of 10 kΩ or less. It is important to ensure that the source impedance is either 10 kΩ or less because the sampling time will be negligible for such a source.

If the source impedance is higher than 10 kΩ then the time to charge the capacitor will increase and the result will not be accurate. For example, if the voltage divider used at the ADC uses a resistor network, make sure that the source impedance is less than 10 kΩ. Low impedance must be used for slowly varying signals since this minimizes the time for charge transfer. Frequency components higher than the Nyquist frequency ($f_{ADC/2}$) must be removed with a low-pass filter, in order to avoid distortion from unpredictable signal convolution. Refer to the device data sheet for impedance value.

Figure 1-2. Analog Input Circuitry



1.9 Best Practices for Improving ADC Performance

The performance of an ADC depends on the quality of the input signals and power supplies. The following points should be taken into consideration for improving the accuracy of ADC measurements:
Note: Some points mentioned below might not be applicable for all megaAVRs based on their feature set. Refer to the device data sheet for further information.

- Understand the ADC, its features, and how they are intended to be used
- Understand the application requirements
- Ensure that the source impedance is not too high compared to the sampling rate used. If the source impedance is too high, the internal sampling capacitor will not be charged to the correct level and the result will be inaccurate.
- It is important to take great care of the analog signal paths like analog reference (VREF) and analog power supply (AVCC). Use filtering if the analog power supply is connected to a digital power supply. That is, the AVCC pin on the device should be connected to the digital supply pin (VCC) via an LC network. For more information about this LC network, refer to the respective device data sheet.

- Keep analog signal paths as short as possible. It is also important that the impedance of the PCB tracks is not too high, as it will result in a longer charging period for the sample/hold capacitor of the ADC.
- Ensure that the analog tracks run over the analog ground plane
- Avoid having the analog signal path close to digital signal paths with high switching noise, such as communication lines and clock signals
- Consider decoupling of the analog signal between signal input and ground for single-ended inputs
- Avoid toggling of port pins while the ADC conversion is in progress in order to prevent the switching noise affecting the accuracy. The ADC is most sensitive to switching of the I/O pins powered by AVCC (PORTC).
- Disable digital input on the corresponding ADC channel to minimize the power consumption
- Switch off all unused peripherals by writing the corresponding bit in the Power Reduction register (PRR) to '1'
- Use the ADC noise reduction mode to get more accurate results
- Wait until the ADC, reference, or sources are stabilized before sampling, as some sources (for example, bandgap) need time to stabilize after they are enabled
- Before triggering an ADC conversion, wait until the ADC completes any ongoing conversions. Ensure that enough time is given for the reference and input source to be stabilized. For example, the bandgap voltage needs a certain amount of time to stabilize when it is selected as ADC input.
- Apply offset and gain calibration to the measurement
- Use oversampling to increase resolution and eliminate random noise
- AVCC must not differ more than ±0.3V from VCC
- The reference voltage can be made more immune to noise by connecting a capacitor between the AREF pin and ground
- For differential signals, the decoupling has to be between the positive and negative inputs. The decoupling capacitor value depends on the input signal. If the signals are switching fast, the decoupling capacitor must be lower.
- Whenever the input MUX setting or reference voltage selection is modified, it is recommended to discard the first conversion result
- When switching to a differential channel (with gain settings), the first conversion result may have a poor accuracy due to the required settling time for the automatic offset cancellation circuitry. Thus, it is better to discard the first sample result.
- Linear interpolation methods such as one-point (offset) calibration and two-points (offset and gain) calibration method can be used based on the application’s needs

2. Getting Started

This chapter walks you through the basic steps for getting started with simple ADC conversion, and experimenting with its MUX settings. The necessary registers are described along with relevant bit settings. In the code supplied (see chapter Get Source Code from Atmel | START) with this application note, multiple steps, which write to the same register, are combined and the effective value is written to the register in a single step for simplicity.

2.1 General Instructions to Test the Code on STK600

- Place the ATmega88 device on the STK600 using the specific routing card and socket card (STK600-RC032M-29 and STK600-TQFP32)
- Try accessing the device from the menu **Tools → Device Programming** in Atmel Studio
- The voltage from AREF1 can be used as input to the ADC. Adjust the AREF1 voltage via the menu **Tools → Device Programming → Board settings** in Atmel Studio.
- Connect AREF1 on the STK600 to the ADC input channel ADC0 (PC0) used in the examples
- On the STK600, connect PB0 to LED0, and PB1 to LED1. These pins are used in the following examples to give a visual indication.

2.2 Single Conversion Mode

Task: Single Conversion on ADC channel 0

In this program the `initialize()` routine is used to initialize the ADC module. The `convert()` routine has to be called whenever the application needs an ADC conversion.

1. Set the MUX bit fields (MUX3:0) in ADC's MUX register (ADMUX) equal to 0000 to select ADC Channel 0.
2. Set the ADC Enable bit (ADEN) in ADC Control and Status Register A (ADCSRA) to enable the ADC module.
3. Set the ADC Pre-scalar bit fields (ADPS2:0) in ADCSRA equal to 100 to prescale the system clock by 16.
4. Set the Voltage Reference bit fields (REFS1:0) in ADMUX equal to 11 to select Internal 1.1V reference.
5. Set the Start Conversion bit (ADSC) in ADCSRA to start a single conversion.
6. Poll (wait) for the Interrupt Flag (ADIF) bit in the ADCSRA register to be set, indicating that the conversion is completed.
7. After the ADIF bit becomes high, read the ADC data register pair (ADCL/ADCH) to get the 10-bit ADC result.

2.2.1 Test Steps

1. Build the project and load the .hex file into the device.
2. Make the arrangements on STK600 as described in General Instructions to Test the Code on STK600.
3. Adjust the voltage applied to PC0 and check whether the LED indication is updated depending on the voltage.
4. If the voltage provided on PC0 is higher than ~0.5V, LED0 connected to PB0 will remain OFF, otherwise LED0 will remain ON. LED1 connected to PB1 will always twinkle indicating that the code is running.

2.3 Free-Running Interrupt Mode

Task: Free-running conversion on ADC channel 0. Use of conversion complete interrupt.

In this program the `initialize()` routine is used to initialize the ADC module. In free-running mode, a new conversion will be started immediately after a conversion completes. The ADSC bit remains high during a conversion. The time between two consecutive ADC samples depends on the ADC conversion time.

1. Repeat steps 1-4 from the Single Conversion Mode.
2. Set the ADC Interrupt Enable bit (ADIE) in ADCSRA equal to 1 to enable the ADC conversion complete interrupt.
3. Set the Auto Trigger Enable bit (ADATE) in ADCSRA equal to 1 to enable auto triggered mode. By default, the Auto Trigger Source bit fields (ADTS2:0) in ADC Control and Status Register B (ADCSRB) is set to 000, which represents free-running mode.
4. Set the Start Conversion bit (ADSC) in ADCSRA to start the first conversion.
5. After the conversion is done (ADIF bit becomes high) the CPU executes the ADC interrupt service routine where the ADC data register pair (ADCL/ADCH) is read to get the 10-bit ADC result.

Test Steps

Refer to Test Steps.

2.4 Auto Triggering Using Timer0 Compare Event as Trigger Source

Task: Auto triggered conversion on ADC channel 0 by using Timer as trigger source.

In this program, the `initialize_adc()` routine is used to initialize the ADC module. The `initialize_timer()` routine configures the Timer0 for comparing A match event. The ADC result is read inside the corresponding ISR as soon as the conversion is completed. The ADC module will start the conversion whenever the timer reaches its compare match value (that is, in this example, every 10 milliseconds).

Note that by changing the 'OCR0A' compare register and timer0 clock prescaler bits in 'TCCR0B' register, the conversion interval (from 10 milliseconds) can be changed as per the application needs.

1. Repeat steps 1-4 from the Single Conversion Mode.
2. Set the ADC Interrupt Enable bit (ADIE) in ADCSRA to enable the ADC interrupt.
3. Set the Auto Trigger Enable bit (ADATE) in ADCSRA to enable the auto triggered mode.
4. Set the Auto Trigger Source bit fields (ADTS2:0) in ADC Control and Status Register B (ADCSRB) to 011 to use the Timer0 Compare Match A event as an ADC start trigger.
5. Set the Waveform Generation Mode bit fields (WGM1:0) in TCCR0A to '10' (Clear Timer on Compare Match).
6. Set the Timer0 Clock Select bit fields (CS2:0) in TCCR0B to '011' (ClkIO/64 → Prescaler).
7. Set the Output Compare register A to the desired value (in this example 156) so that the ADC will convert the analog value to a 10 milliseconds (compare match event) interval.
8. After the conversion is done (ADIF bit becomes high) the CPU executes an ADC interrupt service routine where the ADC data register pair (ADCL/ADCH) is read to get the 10-bit ADC result. An ADC conversion is triggered whenever the configured Timer0 compare match happens.

Test Steps

Refer to Test Steps.

2.5 Measurement of Bandgap Reference Voltage

Task: Measure the bandgap reference voltage.

The `measure_gnd()` routine measures the ground value. The `measure_bandgap()` routine measures the internal band gap reference voltage.

1. Repeat steps 1-3 from the Single Conversion Mode.
2. Set the Voltage Reference bit fields (REFS1:0) in ADMUX equal to 01 to select AVCC as ADC voltage reference.
3. Set the ADC Interrupt Enable bit (ADIE) in ADCSRA equal to 1 to enable the ADC interrupt.
4. If the switch is pressed, wait till the switch is released, then it starts to measure GND by setting MUX bit fields (MUX3:0) equal to 1111. This is done to discharge the capacitor of the ADC.
5. While measuring the ground the ADC interrupt and auto trigger (free-running) modes are disabled.
6. Set the Start Conversion bit (ADSC) in ADCSRA to start the conversion.
7. After the conversion is done (ADIF bit becomes high) read the ADC data register pair (ADCL/ADCH) to get the 10-bit ADC result value.
8. Configure the MUX bit field (MUX3:0) equal to 1110 to select the bandgap reference voltage as ADC input. It should be measured after a 70 µs delay, because of the start-up time for the Bandgap reference.
9. Polling method is used for checking conversion complete. The Auto triggering mode is disabled.
10. Set the Start Conversion bit (ADSC) in ADCSRA to start the conversion.
11. After the ADIF bit becomes high, read the ADC data register pair (ADCL/ADCH) to get the 10-bit ADC result.

Test Steps

1. Connect PB5 to one of the switches available on STK600.
2. Open the project in Atmel Studio 7. Press Alt+F5 to start debugging.
3. If debugWIRE is not already enabled, Atmel Studio will prompt to enable debugWIRE.
4. After it goes to debug mode, set a breakpoint at the end of the function `measure_bandgap`.
5. Run the code and press the switch connected to PB5 and release it shortly.
6. At this point, execution will hit the breakpoint set inside the `measure_bandgap` function.
7. Add the variable `bg_val` to the debug watch window to see the ADC reading. The ADC result register can also be checked via I/O view (using the menu **Debug → Windows → I/O**).

2.6 Noise Reduction Sleep

Task: Use of ADC noise reduction sleep.

In this program, the `initialize()` routine is used to initialize the ADC module. The example uses ADC noise reduction sleep mode. Enter the sleep mode by executing 'sleep' instruction. The ADC will start a conversion after the CPU has been halted. As the ADC conversion complete interrupt is enabled, the Interrupt Service Routine for this interrupt will be triggered as soon as the conversion is completed. ADSC bit remains high during a conversion. When the execution of ISR is finished, the execution comes to the main routine and it executes 'sleep' again, which triggers another conversion.

1. Repeat steps 1-4 from the Single Conversion Mode.
2. Set the ADC Interrupt Enable bit (ADIE) in ADCSRA equal to 1 to enable the ADC conversion complete interrupt.

3. Set the Auto Trigger Enable bit (ADATE) in ADCSRA equal to 1 to enable auto triggered mode. By default, the Auto Trigger Source bit fields (ADTS2:0) in ADC Control and Status Register B (ADCSRB) is set to 000, which represents free-running mode.
4. Set the Start Conversion bit (ADSC) in ADCSRA to start the first conversion.
5. After the conversion is done (ADIF bit becomes high) the CPU executes the ADC interrupt service routine where the ADC data register pair (ADCL/ADCH) is read to get the 10-bit ADC result.

Test Steps

Refer to Test Steps.

3. Get Source Code from Atmel | START

The example code is available through Atmel | START, which is a web-based tool that enables configuration of application code through a Graphical User Interface (GUI). The code can be downloaded for both Atmel Studio 7.0 and IAR Embedded Workbench® via the direct example code-link(s) below, or the *BROWSE EXAMPLES* button on the Atmel | START front page.

Atmel | START web page: http://start.atmel.com/

Example Code

- AVR126 ADC single conversion:
  - http://start.atmel.com/#example/Atmel%3AAVR126_ADC%3A1.0.0%3A%3AApplication%3AAVR126_ADC_single_conversion%3A
- AVR126 ADC free run mode:
  - http://start.atmel.com/#example/Atmel%3AAVR126_ADC%3A1.0.0%3A%3AApplication%3AAVR126_ADC_free_run_mode%3A
- AVR126 ADC bandgap measure:
  - http://start.atmel.com/#example/Atmel%3AAVR126_ADC%3A1.0.0%3A%3AApplication%3AAVR126_ADC_bandgap_measure%3A
- AVR126 ADC auto triggered:
  - http://start.atmel.com/#example/Atmel%3AAVR126_ADC%3A1.0.0%3A%3AApplication%3AAVR126_ADC_auto_triggered%3A
- AVR126 ADC Interrupt:
  - http://start.atmel.com/#example/Atmel%3AAVR126_ADC%3A1.0.0%3A%3AApplication%3AAVR126_ADC_Interrupt%3A
- AVR126 ADC Noise Reduction Sleep:
  - http://start.atmel.com/#example/Atmel%3AAVR126_ADC%3A1.0.0%3A%3AApplication%3AAVR126_ADC_Noise_Reduction_Sleep%3A

Press *User guide* in Atmel | START for details and information about example projects. The *User guide* button can be found in the example browser, and by clicking the project name in the dashboard view within the Atmel | START project configurator.

Atmel Studio

Download the code as an .atzip file for Atmel Studio from the example browser in Atmel | START, by clicking *DOWNLOAD SELECTED EXAMPLE*. To download the file from within Atmel | START, click *EXPORT PROJECT* followed by *DOWNLOAD PACK*.

Double-click the downloaded .atzip file and the project will be imported to Atmel Studio 7.0.

IAR Embedded Workbench

For information on how to import the project in IAR Embedded Workbench, open the Atmel | START User guide, select *Using Atmel Start Output in External Tools*, and *IAR Embedded Workbench*. A link to the Atmel | START user guide can be found by clicking *About* from the Atmel | START front page or *Help And Support* within the project configurator, both located in the upper right corner of the page.

4. Device Data Sheet Reference

This application note describes the ATmega88's ADC module. The operation of on-chip ADC on different megaAVR devices are almost similar. Refer the respective device data sheet for more details. For example, some megaAVR device variants (such as ATmega48P) have on-chip temperature sensor as one of the ADC input, some devices have differential ADC channels with configurable gains and 2.56V internal voltage reference (such as ATmega2560), some devices support high ADC speed such as 125 ksps (like AT90PWM1). Refer to data sheet section *Analog to Digital Converter* for detailed information on the ADC of the specific device.

Data sheet section '*Electrical Characteristics → ADC Characteristics*, *Electrical Characteristics → System and Reset Characteristics* contains ADC related characterization data such as INL, DNL, absolute accuracy, conversion time, offset error, gain error, input resistance, band gap reference start-up time, etc.

Data sheet section *Errata* contains information about any known ADC related errata for a device. If there are any erratas which have a workaround mentioned, ensure that the workarounds have been implemented.

5. Driver Implementation

This application note includes a source code package with a basic ADC driver implemented in C. It is developed with Atmel Studio 7 for the ATmega88 device. Note that this ADC driver is not intended for use with high-performance code. It is designed as a library to get started with the ADC.

Following are the features covered by the examples in this application note:

1. Single conversion mode.
2. Free-running mode and conversion complete interrupt.
3. Auto triggering using Timer0 compare event as trigger source.
4. Measurement of bandgap reference voltage.
5. Noise Reduction Sleep.

6. Recommended Reading

- **AVR042:** AVR® Hardware Design Considerations
  This application note provides answers to some of the questions and problems faced when starting designs involving AVR® microcontrollers.
- **AVR120:** Characterization and Calibration of the ADC on an AVR
  This application note explains various ADC (Analog to Digital Converter) characterization parameters, how they affect ADC measurements, how to measure them, and how to perform run-time compensation.
- **AVR121:** Enhancing ADC Resolution by Over Sampling
  This application note explains the method called 'Oversampling and Decimation' and which conditions need to be fulfilled to make this method work properly to achieve a higher resolution without using an external ADC.
- **AVR122:** Calibration of the AVR's Internal Temperature Reference
  This application note describes how to calibrate and compensate the temperature measurements from the ATtiny25/45/85. It can also be used on other AVR microcontrollers with internal temperature sensors.
- **AVR125:** ADC of tinyAVR® in Single-Ended Mode
  This application note describes the basic functionality of the ADC in tinyAVR devices in single-ended mode with code examples on ATtiny88 to get started. The code examples are written in assembly language and C language.
- **AVR127:** Understanding ADC Parameters
  This application note discusses about the basic concepts of analog-to-digital converter (ADC) and the various parameters that determine the performance of an ADC. These ADC parameters are of high importance since they are part of deciding the accuracy of the ADC's output.

7. Revision History

Doc Rev.	Date	Comments
A	08/2017	Converted to Microchip format and replaced the Atmel document number 8444B. Refurbished the content.
8444B	03/2016	Added new example.
8444A	10/2011	Initial document release.

The Microchip Web Site

Microchip provides online support via our web site at http://www.microchip.com/. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Customer Change Notification Service

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at http://www.microchip.com/. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-2120-7

Quality Management System Certified by DNV

ISO/TS 16949

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Asia Pacific Office** Suites 3707-14, 37th Floor Tower 6, The Gateway Harbour City, Kowloon **Hong Kong** Tel: 852-2943-5100 Fax: 852-2401-3431 **Australia - Sydney** Tel: 61-2-9868-6733 Fax: 61-2-9868-6755 **China - Beijing** Tel: 86-10-8569-7000 Fax: 86-10-8528-2104 **China - Chengdu** Tel: 86-28-8665-5511 Fax: 86-28-8665-7889 **China - Chongqing** Tel: 86-23-8980-9588 Fax: 86-23-8980-9500 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 Fax: 86-571-8792-8116 **China - Hong Kong SAR** Tel: 852-2943-5100 Fax: 852-2401-3431 **China - Nanjing** Tel: 86-25-8473-2460 Fax: 86-25-8473-2470 **China - Qingdao** Tel: 86-532-8502-7355 Fax: 86-532-8502-7205 **China - Shanghai** Tel: 86-21-3326-8000 Fax: 86-21-3326-8021 **China - Shenyang** Tel: 86-24-2334-2829 Fax: 86-24-2334-2393 **China - Shenzhen** Tel: 86-755-8864-2200 Fax: 86-755-8203-1760 **China - Wuhan** Tel: 86-27-5980-5300 Fax: 86-27-5980-5118 **China - Xian** Tel: 86-29-8833-7252 Fax: 86-29-8833-7256	**China - Xiamen** Tel: 86-592-2388138 Fax: 86-592-2388130 **China - Zhuhai** Tel: 86-756-3210040 Fax: 86-756-3210049 **India - Bangalore** Tel: 91-80-3090-4444 Fax: 91-80-3090-4123 **India - New Delhi** Tel: 91-11-4160-8631 Fax: 91-11-4160-8632 **India - Pune** Tel: 91-20-3019-1500 **Japan - Osaka** Tel: 81-6-6152-7160 Fax: 81-6-6152-9310 **Japan - Tokyo** Tel: 81-3-6880- 3770 Fax: 81-3-6880-3771 **Korea - Daegu** Tel: 82-53-744-4301 Fax: 82-53-744-4302 **Korea - Seoul** Tel: 82-2-554-7200 Fax: 82-2-558-5932 or 82-2-558-5934 **Malaysia - Kuala Lumpur** Tel: 60-3-6201-9857 Fax: 60-3-6201-9859 **Malaysia - Penang** Tel: 60-4-227-8870 Fax: 60-4-227-4068 **Philippines - Manila** Tel: 63-2-634-9065 Fax: 63-2-634-9069 **Singapore** Tel: 65-6334-8870 Fax: 65-6334-8850 **Taiwan - Hsin Chu** Tel: 886-3-5778-366 Fax: 886-3-5770-955 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 Fax: 886-2-2508-0102 **Thailand - Bangkok** Tel: 66-2-694-1351 Fax: 66-2-694-1350	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4450-2828 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **France - Saint Cloud** Tel: 33-1-30-60-70-00 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-67-3636 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-7289-7561 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820