



## Using the analog-to-digital converter of the STM8A microcontroller

---

### **Introduction**

The purpose of this application note is to explain how to use the analog-to-digital converter implemented in the STM8A microcontroller family. The document provides useful information on how to configure the ADC registers and microcontroller resources and use the analog-to-digital converter in different modes.

The STM8A firmware library, containing source code of all the examples described in this application note, can be downloaded from the STMicroelectronics website: [www.st.com](http://www.st.com).

# Contents

- 1      ADC hardware description ..... 3**
  - 1.1    General properties ..... 3
  - 1.2    ADC operation ..... 5
  
- 2      Summary of the features ..... 7**
  
- 3      Configuring ADC registers for conversion ..... 9**
  - 3.1    ADC setup ..... 9
    - 3.1.1    ADC clock ..... 9
    - 3.1.2    ADC on/off control (power on) ..... 9
    - 3.1.3    Channel selection ..... 10
    - 3.1.4    Conversion on external trigger ..... 10
    - 3.1.5    Conversion modes ..... 10
  - 3.2    Storing converted values ..... 17
  - 3.3    ADC interrupt management ..... 17
  
- 4      Practical application ..... 18**
  - 4.1    Areas of use ..... 18
  - 4.2    Hardware connection examples ..... 18
  - 4.3    Methods for precision improvement ..... 20
  
- 5      Design recommendations ..... 21**
  
- 6      Revision history ..... 22**

# 1 ADC hardware description

## 1.1 General properties

STM8A family microcontrollers include an analog-to-digital converter which has up to 16 multiplexed inputs. The ADC resolution is 10 bits. The number of external analog inputs depends on the package size of the particular STM8A family device.

The ADC is a successive approximation analog-to-digital converter. Conversion can be performed in continuous mode or single mode. The digital result is then stored in registers. Because the ADC resolution is 10 bits and the ADC data register length is 16 bits, the analog-to-digital converter result is stored in two 8-bit registers, and the data is either right or left aligned (this is selectable).

The event used to start conversion can be generated by software or by the STM8A microcontroller's internal timer 1 and timer 2<sup>(a)</sup> (any type of timer event can be used). The start of A/D conversion can also be triggered by an external pin.

The reference voltage for the analog-to-digital converter depends on the package. It comes either from an external source—in this case the reference must be connected to two external pins—or is connected internally to analog power supply pins. The value of this reference voltage is limited to within the range from 2.75 V up to the voltage applied on the  $V_{DDA}$  input. The measured voltage must be between  $V_{REF+}$  and  $V_{REF-}$ . Resolution can be increased by so-called analog zooming—using a smaller reference voltage around the range of values to be measured.

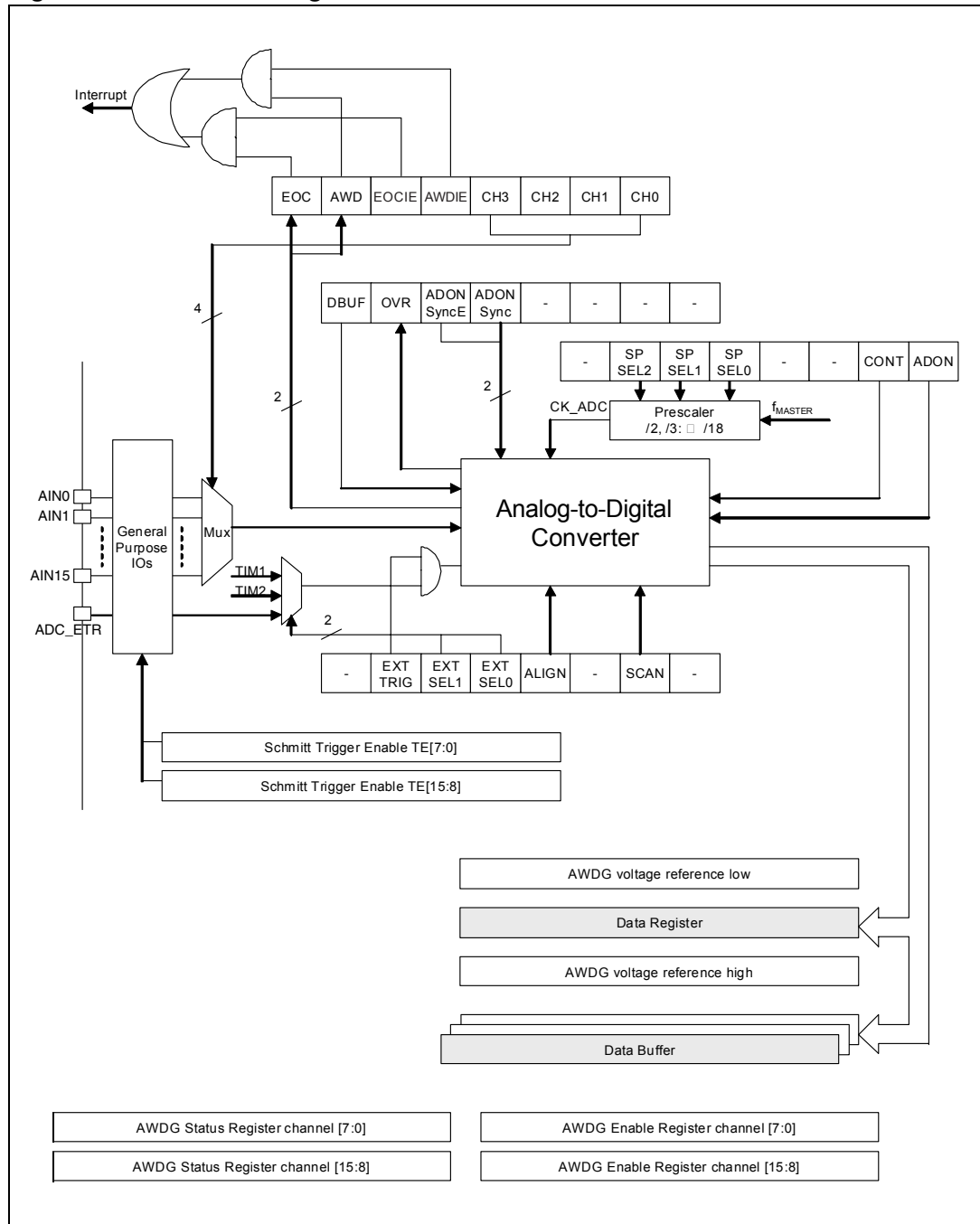
If needed, an interrupt can be generated at the end of conversion and if an analog watchdog event occurred.

The analog-to-digital converter is driven by a clock derived from the MCU master clock through a programmable divider. This allows you to select the ADC clock speed according to your application requirements.

---

a. Trigger feature available on selected devices

Figure 1. ADC block diagram



1. Refer to STM8A reference manual for ADC registers bit description

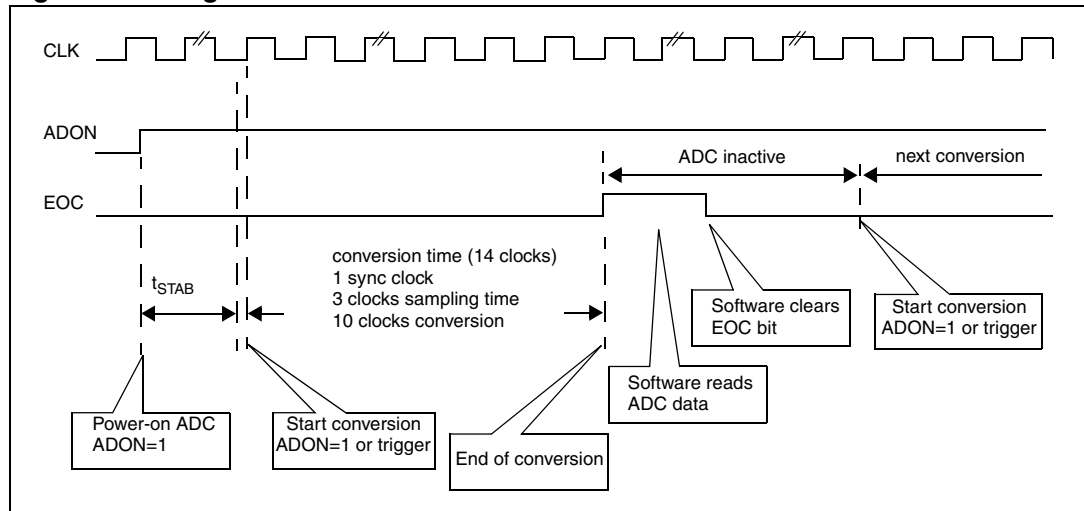
## 1.2 ADC operation

The analog-to-digital converter supports two basic conversion modes: single mode and continuous mode.

Figure 2 and Figure 3 describe single and continuous mode analog-to-digital converter operation.

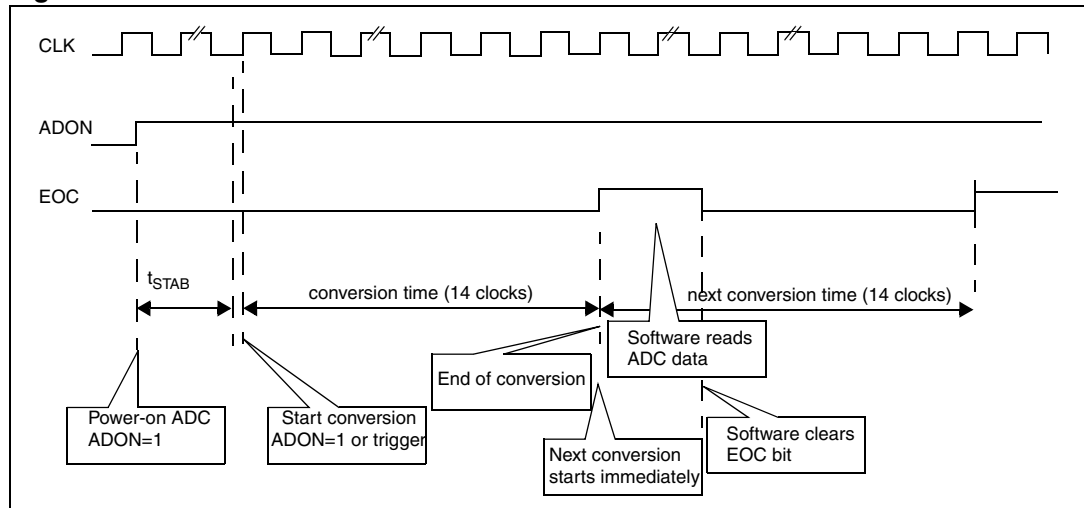
In single conversion mode, the analog-to-digital converter performs only one conversion on the selected channel.

**Figure 2. Single conversion mode**



In continuous conversion mode, the analog-to-digital converter starts another conversion as soon as the previous one is concluded.

**Figure 3. Continuous conversion mode**

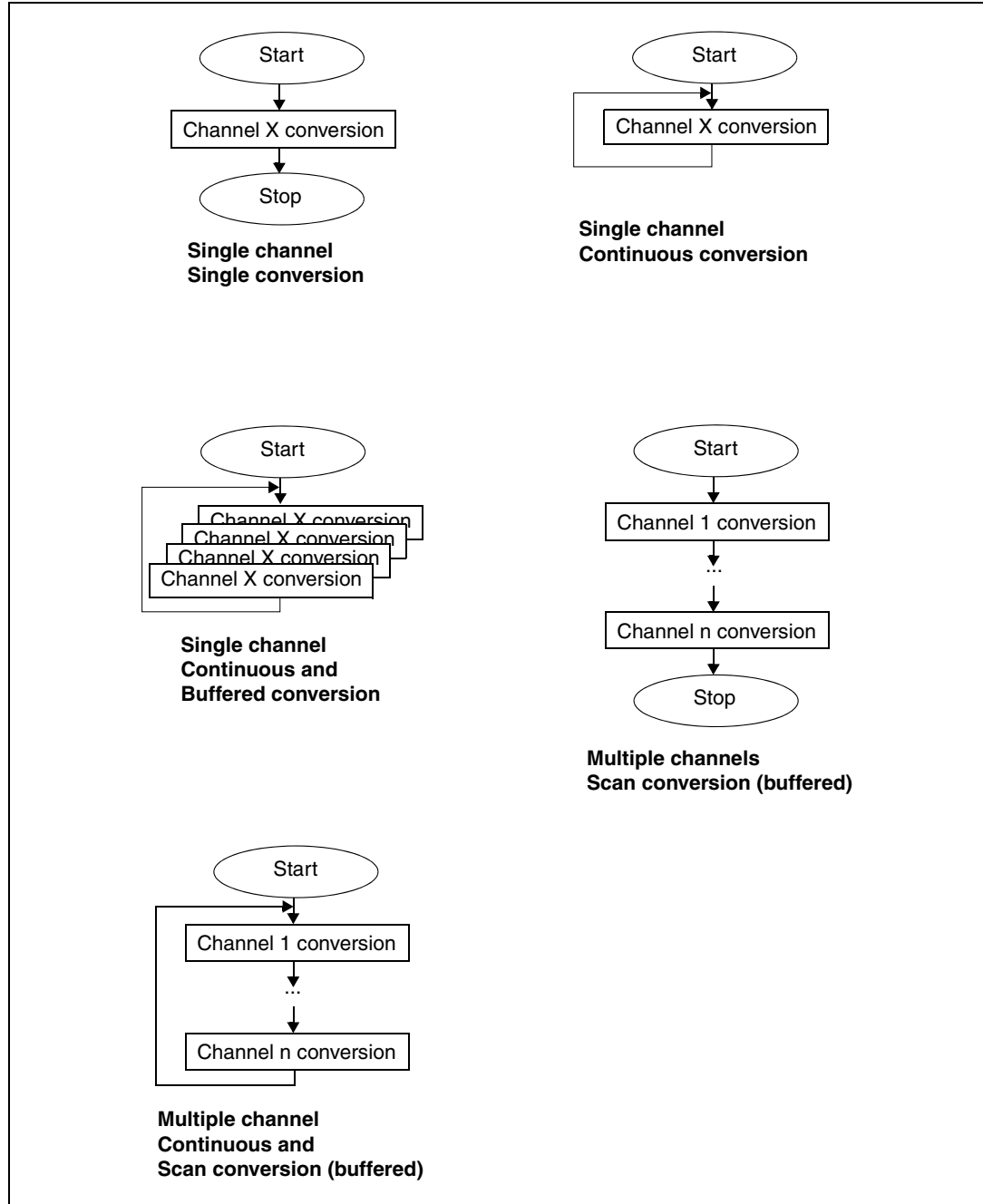


As shown in these figures, after ADC power on, the analog-to-digital converter needs a stabilization time  $t_{STAB}$  (equivalent to one conversion time  $t_{CONV}$ ) before it starts converting accurately. For subsequent conversions there is no stabilization delay and ADON needs to be set only once.

The ADC conversion time takes 14 clock cycles. In addition to the basic conversion modes, some packages have extended features for scan mode and buffered continuous mode.

Figure 4 summarizes all the possible analog-to-digital converter operating modes.

Figure 4. ADC operating modes



## 2 Summary of the features

The STM8A family contains two types of analog-to-digital converter which are named ADC and ADC2. Depending on the device, it may contain both (ADC and ADC2) or only one (ADC).

There are two variants of the first analog-to-digital converter ADC:

1. With standard features
2. With extended features for scan mode, buffered continuous mode and analog watchdog

The second analog-to-digital converter, ADC2, contains a features subset of the first ADC module.

Refer to the STM8A datasheet for more information on the available features.

**Table 1. ADC and ADC2 features on different devices<sup>(1)</sup>**

Module	Features			
	Parameter	32 KB die based <sup>(2)</sup>	128 KB die based <sup>(2)</sup>	256 KB die based <sup>(2)</sup>
ADC	Resolution	10-bit		
	Programmable prescaler	$f_{\text{MASTER}}$ divided by 2 to 18		
	Operating modes	Single or continuous mode		
		Scan and buffered continuous mode	—	Scan and buffered continuous mode
	Clock speed	$f_{\text{ADC}} = 1$ to 4 MHz		
	ADC input range	$V_{\text{SSA}} = V_{\text{IN}} = V_{\text{DDA}}$		
	Conversion speed	min. 3.5 $\mu\text{s}$ @ 4 MHz $f_{\text{ADC}}$		
	Number of input channels	up to 10	up to 16	up to 16
	Triggers	Timer 1 and external ADC_ETR pin	External ADC_ETR pin	Timer 1, Timer 2 and external ADC_ETR pin
	Data format	left or right alignment		
	ADC interrupt	EOC and AWD	EOC	EOC and AWD
	Analog watchdog	Yes	—	Yes
	Schmitt trigger enable/disable	Yes	Yes	Yes
	Synchronization mechanism between ADC and ADC2	—	—	Yes
	Analog zooming	—	Through dedicated VREF pins	Through dedicated VREF pins

**Table 1. ADC and ADC2 features on different devices<sup>(1)</sup> (continued)**

Module	Features			
	Parameter	32 KB die based <sup>(2)</sup>	128 KB die based <sup>(2)</sup>	256 KB die based <sup>(2)</sup>
ADC2	Resolution	—	—	10-bit
	Programmable prescaler	—	—	$f_{\text{MASTER}}$ divided by 2 to 18
	Operating modes	—	—	Single or continuous mode
	Clock speed	—	—	$f_{\text{ADC}} = 1$ to 4 MHz
	ADC input range	—	—	$V_{\text{SSA}} = V_{\text{IN}} = V_{\text{DDA}}$
	Conversion speed	—	—	min. 3.5 $\mu\text{s}$ @ 4 MHz $f_{\text{ADC}}$
	Number of input channels	—	—	12
	Triggers	—	—	Timer1 and Timer2
	Data format	—	—	left or right alignment
	ADC interrupt	—	—	EOC and AWD
	Analog watchdog	—	—	Yes
	Schmitt trigger enable/disable	—	—	Yes
	Synchronization mechanism between ADC and ADC2	—	—	Yes
Analog zooming	—	—	—	

1. For feature details, refer to the STM8A reference manual and the device datasheet.
2. Die information is stored in Chip ID registers. Refer to the reference manual for further details.



## 3 Configuring ADC registers for conversion

### 3.1 ADC setup

Before starting an ADC conversion, ADC setup is done using the following steps:

- ADC clock selection (define conversion speed)
- Channel selection
- Power on the ADC (set ADON bit)
- Wait for 14 ADC clock cycles ( $14 * t_{ADC}$ )
- Conversion mode selection (single or continuous)
- Trigger source selection and only after trigger enable
- Apply trigger source or set ADON bit

#### 3.1.1 ADC clock

The clock supplied to the ADC peripheral is the prescaled  $f_{MASTER}$  clock. The prescaler is applied on bit SPSEL[2:0] in the ADC\_CR1 register and it can be selected from  $f_{MASTER}/2$  to  $f_{MASTER}/18$ .

The prescaler selection must ensure a maximum 4 MHz ADC frequency.

ADC conversion time depends on the ADC clock frequency, the sampling time duration and the ADC resolution. The minimum ADC conversion time is 3.5  $\mu$ s @ 4 MHz ADC clock frequency.

*Note: It is recommended to change bits SPSEL[2:0] when the ADC is in power-down mode. This is because, internally, there may be a glitch in the clock during this change. If this change is made when ADC is not in power-down, the user is required to ignore the first converted result.*

#### 3.1.2 ADC on/off control (power on)

The ADC can be powered on by setting the ADON bit in the ADC\_CR1 register. When the ADON bit is set for the first time, it wakes up the ADC from power-down mode.

After power ON, wait for stabilization time and then, to start a conversion, either set the ADON bit again in the ADC\_CR1 register or use an external trigger source (timers or ADC\_ETR pin).

At the end of conversion, the ADC remains powered on and the ADON bit has to be set only once to start the next conversion. If the ADC is not used for a long time, it is recommended to switch it off to decrease power consumption. This is done by clearing the ADON bit.

When the ADC is powered on, the output stage of the selected channel is disabled; therefore it is recommended to select the channel first before powering on the ADC.

*Note: If any bit of the ADC\_CR1 register other than ADON is changed at the same time, the conversion is not triggered. This is to prevent triggering an erroneous conversion.*

*Moreover, be aware that any bit manipulation instruction on the ADC\_CR1 register can start a new conversion due to the fact that the ADON bit is read as '1' and then written back as '1' by this operation.*

### 3.1.3 Channel selection

There are up to 16 input channels in ADC and up to 12 input channels in ADC2 (the maximum number of channels per ADC peripheral depends on the MCU package size). External input channels can be selected by bits CH[3:0] in the ADC\_CSR register. If the channel is changed during an ongoing conversion, the current conversion is reset and a new start pulse is sent to ADC.

### 3.1.4 Conversion on external trigger

Conversion can also be started by an external trigger event after ADC power ON.

The external trigger source can be a timer<sup>(b)</sup> TRGO event or a rising edge on the external ADC\_ETR<sup>(b)</sup> pin. Bit EXTTRIG is set in the ADC\_CR2 register to enable the conversion by external trigger and the bits EXTSEL[1:0] select the trigger source as either the timer or the external trigger pin.

The different software sequences (shown below) to select the trigger source and enable the external trigger depend on the ADC state (powered-ON or powered-OFF).

#### Case 1: ADC is in powered-ON state

ADC\_CR2 = value1; Select the source (EXTSEL[1:0] bits)

ADC\_CR2 |= 0x40; Enable external trigger (EXTTRIG bit)

#### Case 2: ADC is in powered-OFF state

When the ADC is in powered-OFF state, the user is free to choose between the above software sequence, that is, for ADC powered-ON state or the sequence below:

ADC\_CR2 = value; Select the source and enable the external trigger

### 3.1.5 Conversion modes

#### Single conversion mode

In single conversion mode, the ADC performs one conversion on the channel selected by bits CH[3:0] in the ADC\_CSR register. At the end of conversion, the EOC bit is set and the result can be read from the registers ADC\_DRH and ADC\_DRL. The interrupt is generated if the EOC interrupt is enabled in the ADC\_CSR register.

#### Example 1

- $f_{\text{MASTER}} = 16 \text{ MHz}$
- $f_{\text{ADC}} = 4 \text{ MHz}$
- Channel = AIN1
- Single conversion mode, left alignment mode
- Starting ADC by software

---

b. Refer to the datasheet for the details on the trigger source available.

For [Example 1](#), the ADC registers should be configured as follows:

ADC\_CR1:SPSEL[2:0] = 010 →  $f_{ADC} = 4 \text{ MHz}$  (that is,  $f_{MASTER}/4$ )

ADC\_CSR:CH[3:0] = 0001 → AIN1 channel selected

ADC\_CR1:ADON = 1 → ADC power on

Wait ( $14/f_{ADC}$ ) → stabilization time before starting the conversion

**ADC\_CR1:CONT = 0 → single conversion mode**

*Note:* Do not perform the operation “ADC\_CR1:CONT = 0 → single conversion mode” if the CONT bit is already 0 (see note in [Section 3.1.2](#)).

ADC\_CR2:ALIGN = 0 → left alignment

ADC\_CR1:ADON = 1 → starting the conversion by software

Wait (ADC\_CSR:EOC = 1) → end of conversion

See the result of the conversion in ADC data registers ADC\_DRH and ADC\_DRL.

### Continuous conversion mode

In continuous conversion mode, the ADC starts another conversion as soon as it finishes the last one. A new conversion starts automatically. For this conversion mode, set the CONT bit to ‘1’ in the ADC\_CR1 register.

#### Example 2

- $f_{MASTER} = 4 \text{ MHz}$
- $f_{ADC} = 1 \text{ MHz}$
- Channel = AIN2
- Continuous conversion model, right alignment
- Starting ADC by external trigger event on ADC\_ETR pin

For [Example 2](#), the ADC registers should be configured as follows:

ADC\_CR1:SPSEL[2:0] = 010 →  $f_{ADC} = 1 \text{ MHz}$  (that is,  $f_{MASTER}/4$ )

ADC\_CSR:CH[3:0] = 0010 → AIN2 channel is selected

ADC\_CR1:ADON = 1 → ADC power on

Wait ( $14/f_{ADC}$ ) → stabilization time before starting the conversion

ADC\_CR1:CONT = 1 → continuous conversion mode

ADC\_CR2:ALIGN = 1 → left alignment

ADC\_CR2:EXTSEL[1:0] = 01 → External trigger on ADC\_ETR

ADC\_CR2:EXTTRIG = 1 → conversion on external event is enabled

Rising edge on ADC\_ETR pin

Wait (ADC\_CSR:EOC = 1) → end of conversion

If you want to save the result of the current conversion, save the values of data register ADC\_DRH and ADC\_DRL before starting the next conversion.

### Practical use

Continuous conversion mode is useful for collecting a set of analog values from an external analog signal (for example, to read an AC voltage waveform).

### Single scan mode

Single scan mode is started by triggering a conversion while the SCAN bit is set in the ADC\_CR2 register and the CONT bit is 0 in the ADC\_CR1 register. The scan is performed from channel 0 up to the channel n programmed in bits CH[3:0] in register ADC\_CSR.

Do not clear the SCAN bit while the conversion sequence is in progress. Single scan mode can be stopped immediately by clearing the ADON bit in the ADC\_CR1 register.

### Example 3

- $f_{\text{MASTER}} = 4 \text{ MHz}$
- $f_{\text{ADC}} = 1 \text{ MHz}$
- Single scan mode from AIN0 to AIN5 channel
- Right alignment
- Starting ADC by software

For [Example 3](#), the ADC registers should be configured as follows:

ADC\_CR1:SPSEL[2:0] = 010 →  $f_{\text{ADC}} = 1 \text{ MHz}$  (that is,  $f_{\text{MASTER}}/4$ )

ADC\_CSR:CH[3:0] = 0101 → select the scan from AIN0 to AIN5

ADC\_CR1:ADON = 1 → ADC power on

*Wait ( $14/f_{\text{ADC}}$ )* → stabilization time before starting the conversion

ADC\_CR2:SCAN = 1 → scan mode enable

ADC\_CR2:ALIGN = 1 → right alignment

ADC\_CR1:ADON = 1 → starting the conversion by software

*Wait (ADC\_CSR:EOC = 1)* → end of conversion

*Read the result of the conversion in ADC\_DBxRH and ADC\_DBxRL data buffer registers, where  $x = 0..5$ .*

### Practical use

Single scan mode is useful for collecting different analog values from external sensors. This conversion mode permits a faster conversion operation than the simple single conversion mode (which requires conversion to be performed channel per channel).

### Continuous scan mode

Continuous scan mode is started by triggering a conversion while the SCAN bit is set in the ADC\_CR2 register and CONT bit is set in the ADC\_CR1 register. The scan is performed from channel 0 up to the channel n programmed in bits CH[3:0], and, when the EOC bit is set, a new scan is started automatically. The data buffer registers must be read before the completion of new scan sequence, otherwise the data buffers are overwritten.

Do not clear the SCAN bit while scan conversion is in progress. Continuous scan mode can be stopped immediately by clearing the ADON bit in the ADC\_CR1 register. Alternatively, if

the CONT bit is cleared while conversion is ongoing, conversion stops the next time the last channel has been converted.

#### Example 4

- $f_{\text{MASTER}} = 4 \text{ MHz}$
- $f_{\text{ADC}} = 2 \text{ MHz}$
- Continuous scan from AIN0 to AIN3 channel
- Right alignment
- Starting ADC by TIM1

For *Example 4*, the ADC registers should be configured as follows:

ADC\_CR1:SPSEL[2:0] = 000 →  $f_{\text{ADC}} = 2 \text{ MHz}$  (that is,  $f_{\text{MASTER}}/2$ )

ADC\_CSR:CH[3:0] = 0011 → channel from AIN0 to AIN3

ADC\_CR1:ADON = 1 → ADC power on

Wait ( $14/f_{\text{ADC}}$ ) → stabilization time before starting the conversion

ADC\_CR1:CONT = 1 → continuous conversion mode

ADC\_CR2:SCAN = 1 → scan mode enable

ADC\_CR2:ALIGN = 1 → right alignment

ADC\_CR2:EXTSEL[1:0] = 00 → Internal TIM1 TRGO event

ADC\_CR2:EXTTRIG = 1 → conversion on external event is enabled

Timer initialization for trigger output: The timer has to be configured in PWM mode, with a period of at least 7  $\mu\text{s}$ , in order to guarantee a good ADC conversion.

$f_{\text{MASTER}} = 4 \text{ MHz}$ , Prescaler = 1, TIM1 counter clock = 2 MHz

PWM signal @50 kHz (period = 20  $\mu\text{s}$ )

TIM1\_SMCR:MSM = 1 → TIM1 configured as master mode

TIM1\_CR2:MMS[6:4] = 100 → OC1REF signal is used as trigger output (TRGO)

TIM1\_CR1:CEN = 1 → Timer counter enable

ADC conversion will start at each rising edge event of OC1REF signal

Wait (ADC\_CSR:EOC = 1) → end of conversion

*Read the result of the current conversion from ADC\_DBxRH and ADC\_DBxRL registers ( $x = 0..3$ ) before the end of next conversion. The OVR flag is set if one of the buffer registers is overwritten.*

- Note:
- 1 *In scan mode, do not use the bit manipulation instruction (BRES) to clear the EOC flag because it performs a read-modify-write on the whole ADC\_CSR register, reading the current channel number from the CH[3:0] register and writing it back. This changes the last channel number for the scan sequence. The correct way to clear the EOC flag in continuous scan mode is to load a byte in the CSR register from a RAM variable, clearing the EOC flag and reloading the last channel number for the scan sequence.*
  - 2 *When using scan mode, it is not possible to use channels AIN0 to AINn in output mode because the output stage of each channel is disabled when it is selected by the ADC multiplexer.*

### Buffered continuous mode

The buffered mode is enabled in the continuous conversion mode if bit DBUF = 1 in the ADC\_CR3 register. The data buffer registers are filled with the results for up to 16 consecutive conversions performed on a single channel.

#### Example 5

- $f_{\text{MASTER}} = 2 \text{ MHz}$
- $f_{\text{ADC}} = 0.5 \text{ MHz}$
- Buffered continuous mode for channel 3
- Left alignment
- Starting ADC by software

For [Example 5](#), the ADC registers should be configured as follows:

ADC\_CR1:SPSEL[2:0] = 010 →  $f_{\text{ADC}} = 0.5 \text{ MHz}$  (that is,  $f_{\text{MASTER}}/4$ )

ADC\_CSR:CH[3:0] = 0011 → AIN3 channel

ADC\_CR1:ADON = 1 → ADC power on

Wait ( $14/f_{\text{ADC}}$ ) → stabilization time before starting the conversion

ADC\_CR1:CONT = 1 → continuous conversion mode

ADC\_CR3:DBUF = 1 → data buffer enable

ADC\_CR2:ALIGN = 0 → left alignment

ADC\_CR1:ADON = 1 → starting the conversion of the channel 3 by software

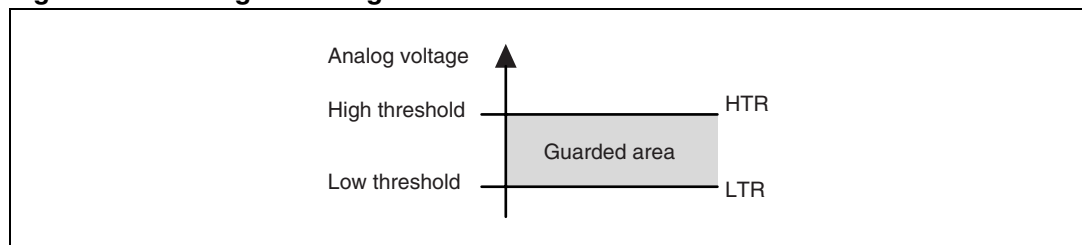
Wait ( $\text{ADC\_CSR:EOC} = 1$ ) → end of conversion

The results of the 16 consecutive conversions should be read from on ADC\_DBxRH and ADC\_DBxRL registers ( $x = 0..15$ ) before the end of the next conversions. The OVR flag is set if one of the buffer registers is overwritten.

### Analog watchdog

In continuous mode, the analog watchdog is always enabled. The AWD analog watchdog flag is set if the analog voltage converted by ADC is out of the guard band limited by low and high threshold values. The low and high threshold values are programmed in the ADC\_HTR and ADC\_LTR registers.

**Figure 5. Analog watchdog**



The ADC\_HTR and ADC\_LTR values can be calculated by the formula

$$\text{ADC\_xTR} = \frac{V_{\text{AIN}}}{V_{\text{REF+}} - V_{\text{REF-}}} \cdot (2^N - 1),$$

where  $V_{AIN}$  is the higher or lower voltage value of the guarded area and N is the ADC resolution.

### Example 6

- $f_{MASTER} = 4$  MHz
- $f_{ADC} = 2$  MHz
- Channel = AIN0 and AIN1
- Continuous scan mode, left alignment
- Analog watchdog:  
Monitoring AIN0 within 1.0 V and 4.0 V  
 $V_{REF+} = 5.0V$ ,  $V_{REF-} = 0.0 V$
- Starting ADC by software

For [Example 6](#), the ADC registers should be configured as follows:

`ADC_CR1:SPSEL[2:0] = 000` →  $f_{ADC} = 2$  MHz (that is,  $f_{MASTER}/2$ )

`ADC_CSR:CH[3:0] = 1100` → AIN0 to AIN12 channels

`ADC_CR1:ADON = 1` → ADC power on

*Wait ( $14/f_{ADC}$ )* → stabilization time before starting the conversion

`ADC_CR1:CONT = 1` → continuous conversion mode

`ADC_CR2:SCAN = 1` → scan mode enable

`ADC_HTRL = 0x03;`

`ADC_HTRH = 0xCC;` → ADC high threshold is 819

`ADC_LTRL = 0x01;`

`ADC_LTRH = 0x33;` → ADC low threshold is 205

`ADC_AWCRL = 0x01;` watchdog enabled on AIN0

`ADC_CR1:ADON = 1` → starting the conversion by software

*Wait (`ADC_CSR:AWD = 1`)* → Analog watchdog event has occurred

*Read the result of the conversion in `ADC_DBxRH` and `ADC_DBxRL` data buffer registers, where  $x = 0..12$ . If any of the conversion happens out of high and low limit, the AWD flag in `ADC_CSR` register and the corresponding bits in the `AWSRL` register will be set to '1'.*

Depending on whether or not the EOC interrupt, the AWD interrupt, or both, are enabled, there could be different combination of managing the analog watchdog. Please refer to the STM8A reference manual for details.

**Note:** *In the buffered continuous mode, the `ADC1_AWCRx` register indicates which data buffered has to be monitored.*

### Synchronization between two ADCs

This feature can be applied only if the device contains both ADC modules (ADC and ADC2). The synchronization mechanism can only be used for single conversion and can be triggered either by software or by hardware.

To start the synchronous conversion, the ADSE bit must be set to '1' in the CR3 registers of both ADC and ADC2.

When the ADS bit of the master ADC module is set, or another event triggers a conversion on the master ADC module, a conversion on the slave module (ADC2) is triggered synchronously.

#### Example 7

- $f_{\text{MASTER}} = 4 \text{ MHz}$
- $f_{\text{ADC}} = f_{\text{ADC2}} = 2 \text{ MHz}$
- Channels: AIN2 on ADC and AIN1 on ADC2
- Left alignment
- Starting ADC and ADC2 by software

For [Example 7](#), the ADC registers should be configured as follows:

```
//ADC2 setting - SLAVE
```

```
ADC2_CSR:CH[3:0] = 0001 → AIN1 channel
```

```
ADC2_CR1 ADON = 1; → ADC2 power on
```

```
Wait ( $14/f_{\text{ADC}}$ ) → stabilization time before starting the conversion
```

```
//ADC setting - MASTER
```

```
ADC_CSR:CH[3:0] = 0010 AIN2 channel
```

```
ADC_CR1 ADON = 1; → ADC power on
```

```
Wait ( $14/f_{\text{ADC}}$ ) → stabilization time before starting the conversion
```

```
ADC_CR3:ADSE = 1; //enable sync on ADC
```

```
ADC2_CR3:ADSE = 1; //enable sync on ADC2
```

```
ADC_CR3:ADS = 1; //ADS start trigger, parallel conversion is started
```

*Note:* Before starting synchronization conversion, the two ADC modules have to be programmed with the same clock prescaler values. It is recommended to program the prescaler register before switching on the ADCs.



## 3.2 Storing converted values

In the scan mode and buffered continuous mode, the converted ADC values are stored in the data buffer registers ADC\_DBxRH and ADC\_DBxRL, where x = 1..15.

In the normal mode, the converted ADC values are stored in the ADC data registers ADC\_DRH and ADC\_DRL. ADC\_DRH and ADC\_DRL also contain the last converted value in scan and buffered continuous mode.

Different data alignment modes (left or right) can be selected depending on the application. If right alignment mode is chosen, the LSB register (ADC\_DRL) must be read before the MSB register (ADC\_DRH), otherwise the values of register ADC\_DRH are not updated. If left alignment mode is chosen, the MSB register (ADC\_DRH) must be read before the LSB register (ADC\_DRL), otherwise the values of register ADC\_DRL are not updated. This prevents reading data register values from two different conversions. Therefore the reading order must be respected.

### Practical use

Left alignment is useful for fast read access to the conversion result in 8-bit resolution.

Right alignment is useful for fast read access when measuring low level signals. In cases where the result does not affect the high order bits, you can read only the LSB byte.

## 3.3 ADC interrupt management

ADC interrupt sources are selectable from one or more of the following events:

- End of conversion (EOC) in nonbuffered mode—generated after each AD conversion
- End of conversion (EOC) in buffered mode—generated after data buffer registers are full
- Analog watchdog alert—when converted values reach the high or low threshold values

You can enable the ADC interrupt to give the main program fast asynchronous event notification at the end of conversion(s). This means the main program does not need to poll the ADC conversion status flags but only has to process the ADC data results (for example, after storing a set of conversions in a buffer). The reception of the ADC converted data values is performed by an interrupt routine in the background. In addition, the analog watchdog interrupt can be used for monitoring a given analog range on the measured channels.

### Register settings:

ADC\_CSR: AWDIE, EOCIE bits

## 4 Practical application

### 4.1 Areas of use

- Temperature measurement
  - Process calibration
  - Device thermal protection
  - Fan control
- Power supply measurement
  - Auto save configuration
  - Battery charging/protection
- Measurement of physical values in other types of applications
  - Automation—from sensors with analog outputs
  - Household—for example, automatic lights dimmers, weather stations, thermometers, security sensors
  - Industry—for example, lights, thermostats, humidity control
  - Electrical quantity measurements—for example, voltage, current, capacity, resistance

### 4.2 Hardware connection examples

Figure 6. Simple unipolar DC signal measurement

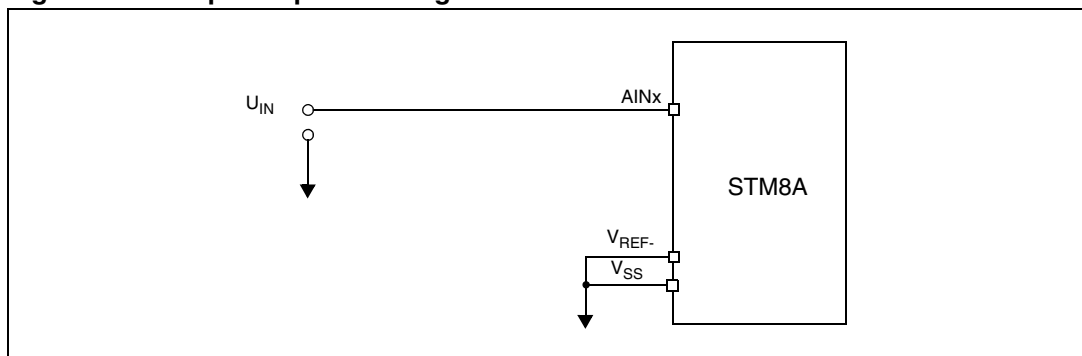
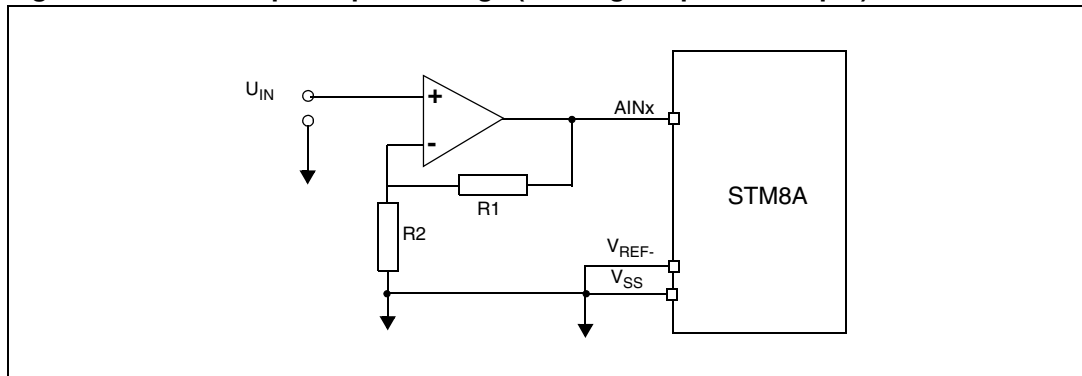
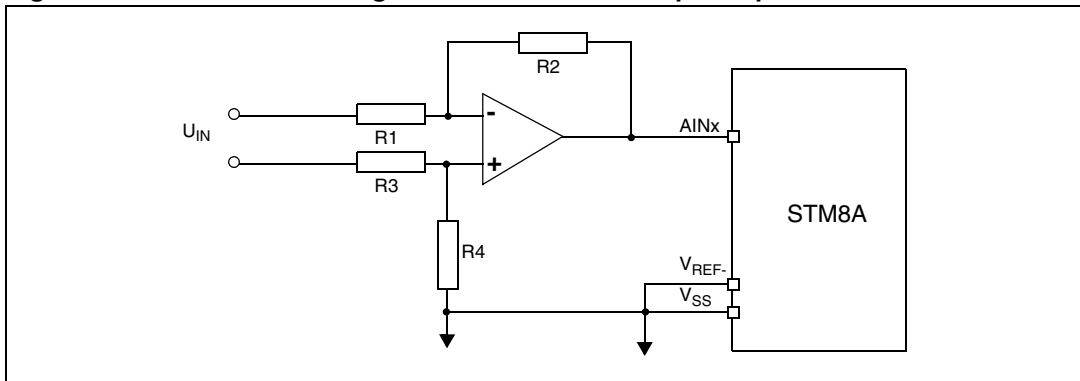


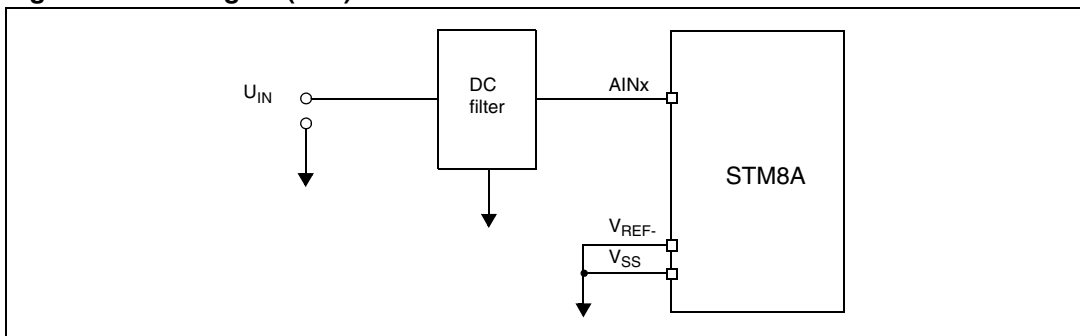
Figure 7. External preamplifier usage (with high impedance input)



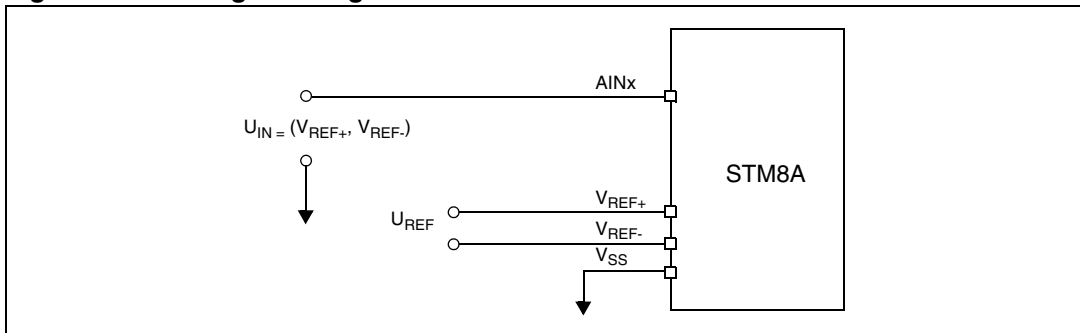
**Figure 8. Differential DC signal measurement with preamplifier**



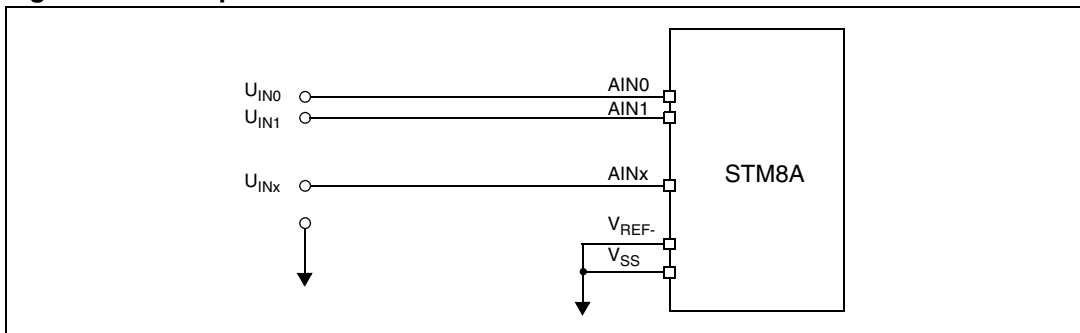
**Figure 9. AC signal (FFT) measurement**



**Figure 10. Analog zooming**



**Figure 11. Multiple channel measurement**



## 4.3 Methods for precision improvement

This section presents some hardware and software methods to minimize errors that can occur in analog-to-digital conversions.

### 4.3.1 Hardware methods

- Analog zooming (use appropriate  $V_{REF}$  voltage and  $V_{REF}$  offset)
  - selects reference voltage between input signal ranges
  - gives full ADC range – minimum voltage per bit
- White noise added to measured signal
  - wobbling of input signal over several bits makes it possible to use averaging (if input signal is very stable)
  - white noise gives independence from sampling frequency
- Hardware design considerations
  - grounding
  - reference voltage filtering
  - supply filtering
  - preamplifier usage
  - frequency independence

### 4.3.2 Software methods

- Averaging samples
  - averaging decreases speed but can improve accuracy
- Digital filtering (50/60 Hz suppression from DC value)
  - proper sampling frequency to be set (trigger from Timer1 is useful in this case)
  - software post processing to be performed on sampled data (comb filter)
- Fast Fourier Transform (FFT) for AC measurements
  - shows harmonic parts in measured signal
  - slower due to more computational power requirements
- Calibration of ADC—offset, gain, bit weight calibration
  - decreases internal ADC errors
  - internal ADC structure must be known

## 5 Design recommendations

The main design recommendations for using the analog-to-digital converter are listed below.

- Grounding of analog/digital power
  - implement star topology
- $V_{DDA}$ ,  $V_{SSA}$  filtering
  - implement RC, LC filtering
  - avoid noise from digital power supply
- $V_{REF}$  selection—offset, value, precision
  - reference voltage source precision and stability to correspond to the required precision in the application and the ADC's capability
  - reference voltage source value and precision to correspond to the expected measurement range
- Source impedance vs. input impedance knowledge
  - use input buffers for measured signal
  - impedance relation with required conversion speed
- External preamplifier usage
  - for low (and also high) level signals
  - amplifier speed and precision properties
  - amplifier dependency on frequency
- Appropriate ADC mode, speed and trigger to be selected

## 6 Revision history

**Table 2. Document revision history**

Date	Revision	Changes
16-Oct-2009	1	Initial release.

**Please Read Carefully:**

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST's terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

**UNLESS OTHERWISE SET FORTH IN ST'S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.**

**UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER'S OWN RISK.**

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2009 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

[www.st.com](http://www.st.com)

