AN2834
Application note

How to get the best ADC accuracy in STM32 microcontrollers

Introduction

STM32 microcontrollers embed up to four advanced 12-bit ADCs (depending on the device). A self-calibration feature is provided to enhance ADC accuracy versus environmental condition changes.

In applications involving analog-to-digital conversion, ADC accuracy has an impact on the overall system quality and efficiency. To improve this accuracy, the errors associated with the ADC and the parameters affecting them must be understood.

ADC accuracy does not only depend on ADC performance and features, but also on the overall application design around the ADC.

This application note aim is to help understand ADC errors and explain how to enhance ADC accuracy. It is divided into three main parts:

- a simplified description of ADC internal structure to help understand ADC operation and related ADC parameters
- explanations of the different types and sources of ADC errors related to the ADC design and to external ADC parameters such as the external hardware design
- recommendations on how to minimize these errors, focusing on hardware and software methods

Contents

List of tables

List of figures

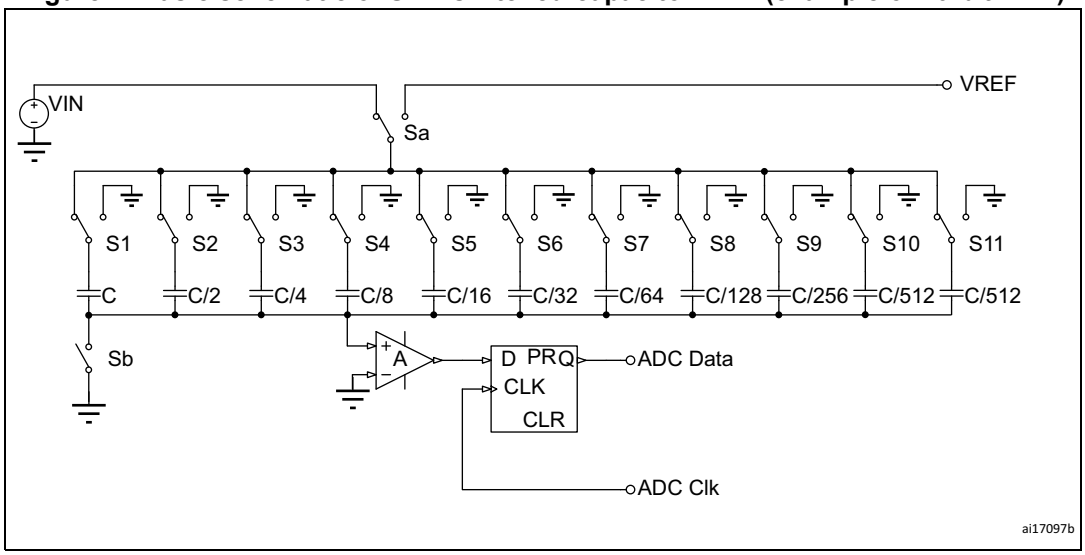
1 ADC internal principle

1.1 SAR ADC internal structure

The ADC embedded in STM32 microcontrollers uses the *SAR (successive approximation register)* principle, by which the conversion is performed in several steps. The number of conversion steps is equal to the number of bits in the ADC converter. Each step is driven by the ADC clock. Each ADC clock produces one bit from result to output. The ADC internal design is based on the switched-capacitor technique.

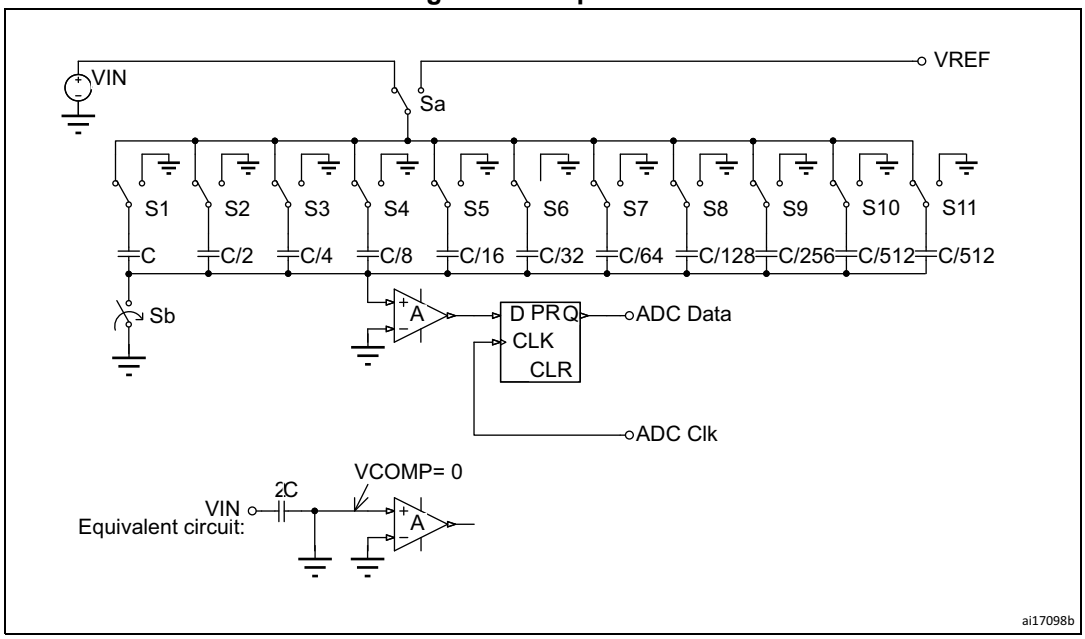
The following figures (*Figure 1* to *Figure 6*) explain the principle of ADC operation. The example given below shows only the first steps of approximation but the process continues till the LSB is reached.

Figure 1. Basic schematic of SAR switched-capacitor ADC (example of 10-bit ADC)



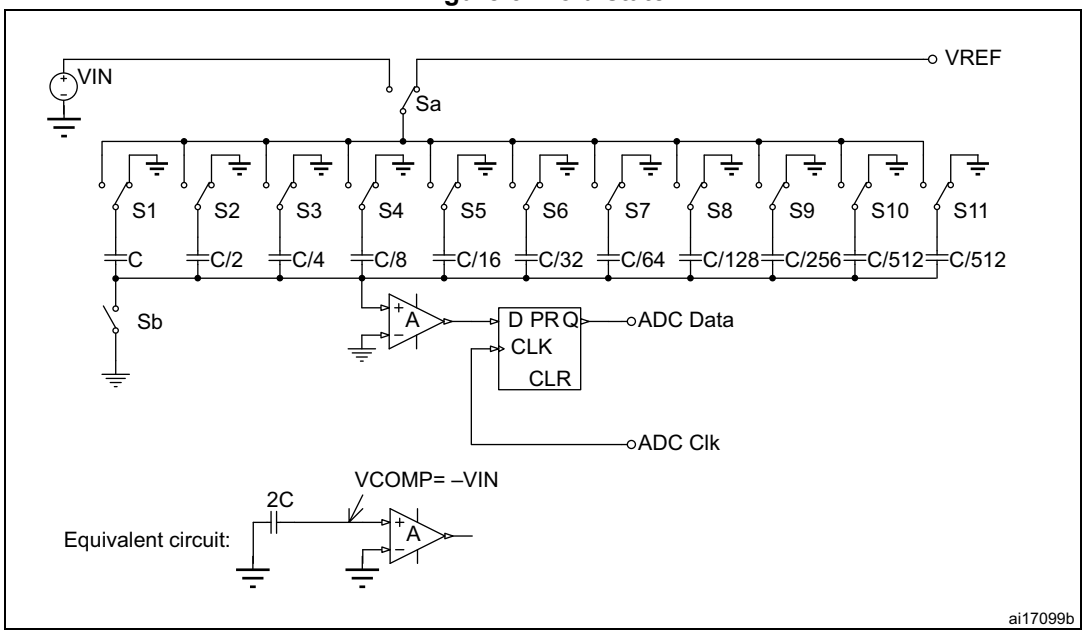
1. Basic ADC schematic with digital output.

Figure 2. Sample state



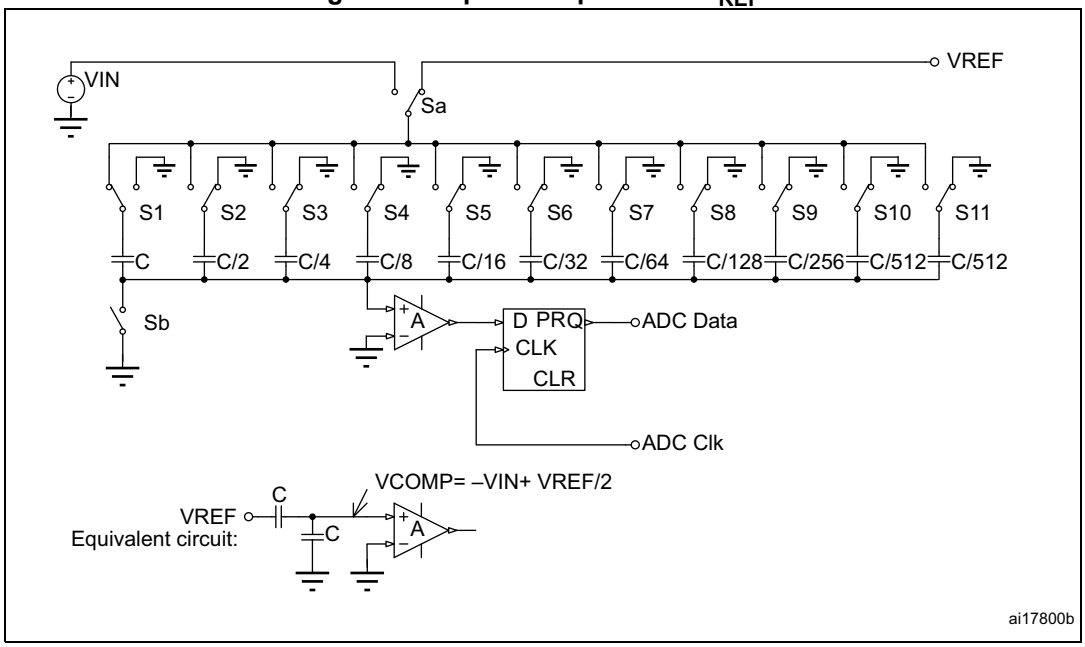
1. Sample state: capacitors are charging to V_{IN} voltage. Sa switched to V_{IN}, Sb switch closed during sampling time.

Figure 3. Hold state



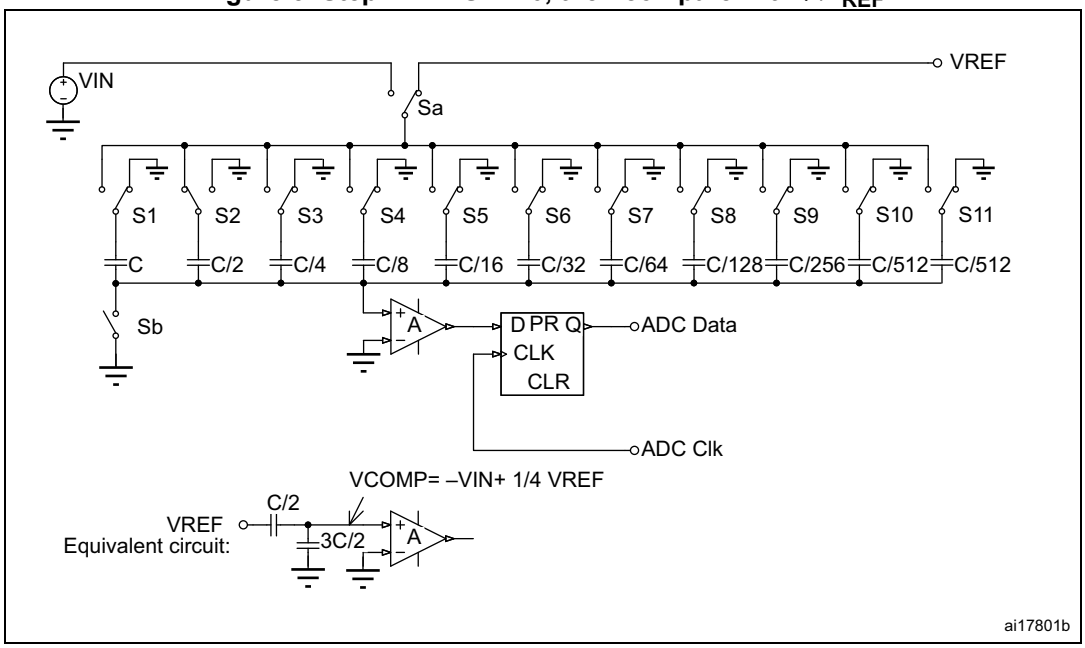
1. Hold state: the input is disconnected, capacitors hold input voltage. Sb switch is open, then S1-S11 switched to ground and Sa switched to V_{REF}.

Figure 4. Step 1: Compare with $V_{REF}/2$



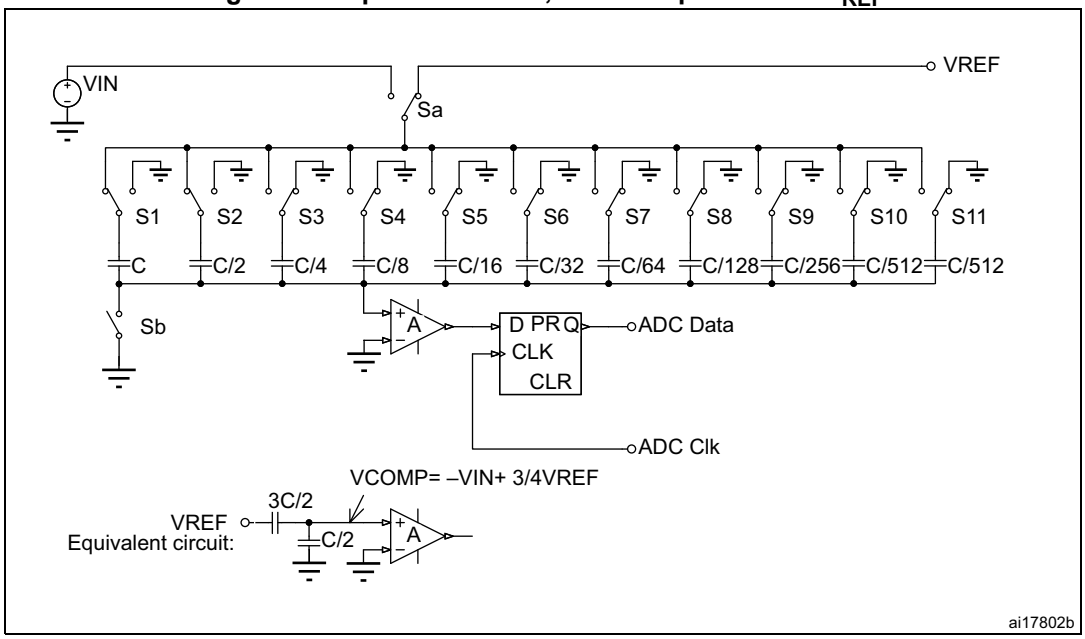
1. First approximation step. S1 switched to V_{REF}.

Figure 5. Step 2: If MSB = 0, then compare with ¼V_{REF}



1. Compare with ¼V_{REF}; if MSB =1. S1 switched back to ground. S2 switched to V_{REF}.

Figure 6. Step 2: If MSB = 1, then compare with ¾V_{REF}



1. Compare with ¾V_{REF}; if MSB =0. S1 remained switched to ground. S2 switched to V_{REF}.

2 ADC errors

This section lists the main errors that have an effect on A/D conversion accuracy. These types of errors occur in all A/D converters and conversion quality depends on their elimination. These error values are specified in the ADC characteristics section of the STM32 microcontroller datasheets.

Different accuracy error types are specified for the STM32 ADC. For easy reference, accuracy errors are expressed as multiples of 1 LSB. The resolution in terms of voltage depends on the reference voltage. The error in terms of voltage is calculated by multiplying the number of LSBs by the voltage corresponding to 1 LSB (1 LSB = $V_{REF+}/2^{12}$ or $V_{DDA}/2^{12}$).

2.1 Errors due to the ADC itself

2.1.1 Offset error

The offset error is the deviation between the first actual transition and the first ideal transition. The first transition occurs when the digital ADC output changes from 0 to 1. Ideally, when the analog input ranges between 0.5 LSB and 1.5 LSB, the digital output should be 1. Still ideally, the first transition occurs at 0.5 LSB. The offset error is denoted by E_O. The offset error can easily be calibrated by the application firmware.

Example

For the STM32 ADC, the smallest detectable incremental change in voltage is expressed in terms of LSBs:

1 LSB = $V_{REF+}/4096$ (on some packages, $V_{REF+} = V_{DDA}$).

If V_{REF+} = 3.3 V, the input of 402.8 µV (0.5 LSB = 0.5 × 805.6 µV) should ideally lead to the generation of a digital output of 1. In practice, however, the ADC may still provide a reading of 0. If a digital output of 1 is obtained from an analog input of 550 µV, then:

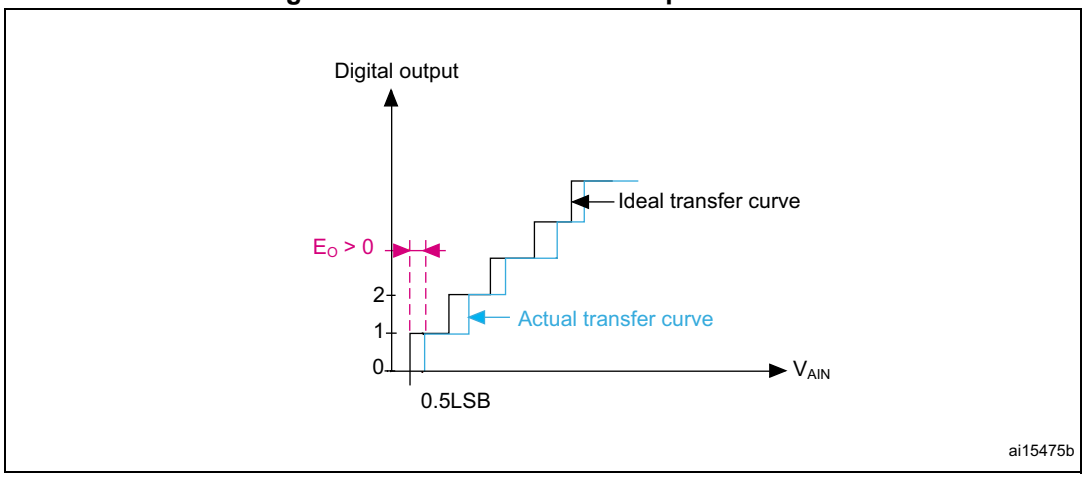
Offset error = Actual transition – Ideal transition

E_O = 550 µV – 402.8 µV = 141.2 µV

E_O = 141.2 µV / 805.6 µV = 0.17 LSB

When an analog input voltage greater than 0.5 LSB generates the first transition, the offset error is positive (refer to *Figure 7* for an example of positive offset error).

Figure 7. Positive offset error representation

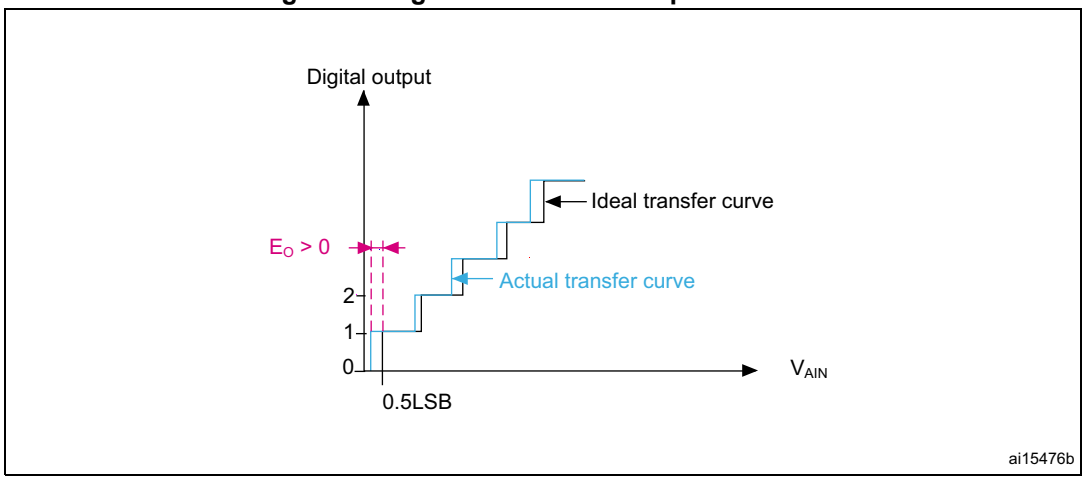


1. The error offset, E_O, is shown in magenta.

When an analog input voltage of less than 0.5 LSB generates the first transition, the offset error is negative (refer to *Figure 8* for an example of negative offset error).

If the analog input voltage (V_{AIN}) is equal to V_{SSA} and the ADC generates a non-zero digital output, the offset error is negative. This means that a negative voltage generates the first transition.

Figure 8. Negative offset error representation



1. The error offset, E_O, is shown in magenta.

2.1.2 Gain error

The gain error is the deviation between the last actual transition and the last ideal transition. It is denoted by E_G.

The last actual transition is the transition from 0xFFE to 0xFFF. Ideally, there should be a transition from 0xFFE to 0xFFF when the analog input is equal to V_{REF+} – 0.5 LSB. So for V_{REF+}= 3.3 V, the last ideal transition should occur at 3.299597 V.

If the ADC provides the 0xFFF reading for $V_{AIN} < V_{REF+}$ – 0.5 LSB, then a negative gain error is obtained.

Example

The gain error is obtained by the formula below:

E_G = Last actual transition – ideal transition

If V_{REF+} = 3.3 V and V_{AIN} = 3.298435 V generate a transition from 0xFFE to 0xFFF then:

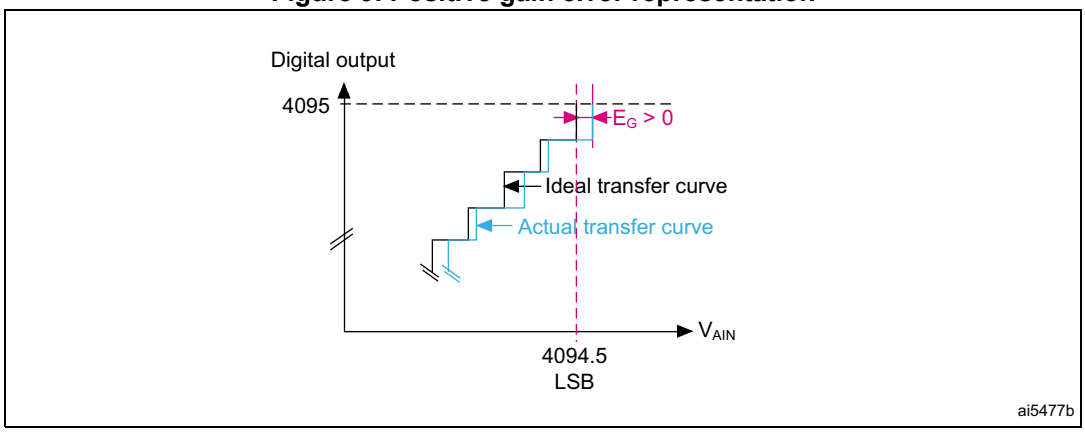
E_G = 3.298435 V – 3.299597 V

E_G = –1162 µV

E_G = (–1162 µV / 805.6 V) LSB = –1.44 LSB

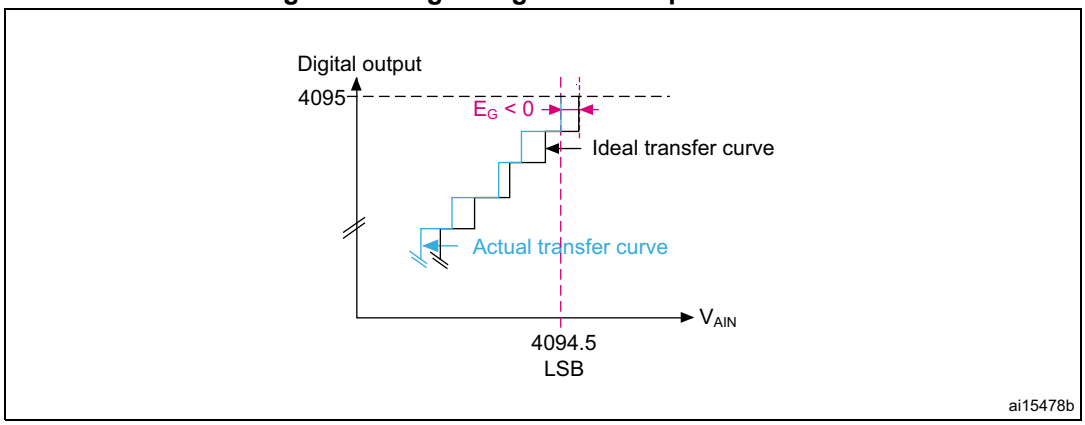
If a full scale reading (0xFFF) is not obtained for V_{AIN} equal to V_{REF+}, the gain error is positive. This means that a voltage greater than V_{REF+} will cause the last transition. *Figure 9* shows a positive gain error while *Figure 10* shows a a negative gain error.

Figure 9. Positive gain error representation



1. The gain error, E_G, is shown in magenta.

Figure 10. Negative gain error representation



1. The gain error, E_G, is shown in magenta.

2.1.3 Differential linearity error

The differential linearity error (DLE) is the maximum deviation between the actual and ideal steps. Here 'ideal' does not refer to the ideal transfer curve but to the ADC resolution. The DLE is denoted by E_D. It is represented in *Figure 11*.

E_D = Actual step width – 1 LSB

Ideally, an analog input voltage change of 1 LSB should cause a change in the digital code. If an analog input voltage greater than 1 LSB is required for a change in digital code, a differential linearity error is observed. The DLE therefore corresponds to the maximum additional voltage that is required to change from one digital code to the next.

The DLE is also known as the differential non-linearity (DNL) error.

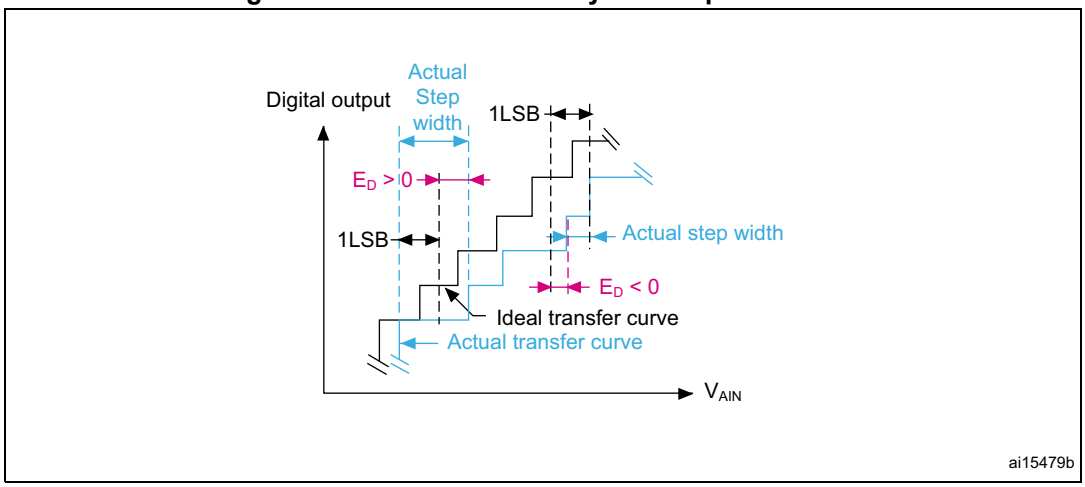
Example

A given digital output should correspond to an analog input range. Ideally, the step width should be 1 LSB. Let us assume that the digital output is the same over an analog input voltage range of 1.9998 V to 2.0014 V, the step width will be:

2.0014 V – 1.9998 V = 1.6 mV.

E_D is thus the voltage difference between the higher (2.0014 V) and the lower (1.9998 V) analog voltages minus the voltage corresponding to 1 LSB.

Figure 11. Differential linearity error representation



1. The differential linearity error, E_D, is shown in magenta.

If V_{REF+} = 3.3 V, an analog input of 1.9998 V (0x9B1) can provide results varying between 0x9B0 and 0x9B2. Similarly, for an input of 2.0014 V (0x9B3), the results may vary between 0x9B2 and 0x9B4.

As a result, the total voltage variation corresponding to the 0x9B2 step is:

0x9B3 – 0x9B1, that is, 2.0014 V – 1.9998 V = 1.6 mV (1660 µV)

E_D = 1660 µV – 805.6 µV

E_D = 854.4 µV

E_D = (854.4 µV/805.6 µV) LSB

E_D = 1.06 LSB

Let us assume that no voltage greater than 2.0014 V will result in the 0x9B2 digital code when the step width is less than 1 LSB, E_D is negative.

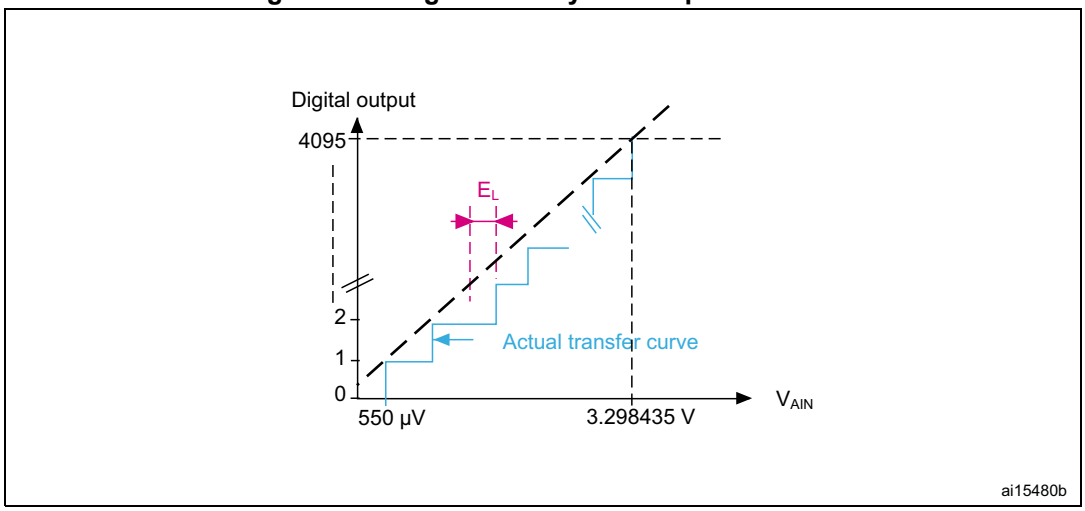
2.1.4 Integral linearity error

The integral linearity error is the maximum deviation between any actual transition and the endpoint correlation line. The ILE is denoted by E_L. It is represented in *Figure 12*.

The endpoint correlation line can be defined as the line on the A/D transfer curve that connects the first actual transition with the last actual transition. E_L is the deviation from this line for each transition. The endpoint correlation line thus corresponds to the actual transfer curve and has no relation to the ideal transfer curve.

The ILE is also known as the integral non linearity error (INL). The ILE is the integral of the DLE over the whole range.

Figure 12. Integral linearity error representation



1. The integral linearity error, E_L, is shown in magenta.

Example

If the first transition from 0 to 1 occurs at 550 µV and the last transition (0xFFE to 0xFFF) occurs at 3.298435 V (gain error), then the line on the transfer curve that connects the actual digital codes 0x1 and 0xFFF is the endpoint correlation line.

2.1.5 Total unadjusted error

The total unadjusted error (TUE) is the maximum deviation between the actual and the ideal transfer curves. This parameter specifies the total errors that may occur, thus causing the maximum deviation between the ideal digital output and the actual digital output. TUE is the maximum deviation recorded between the ideal expected value and the actual value obtained from the ADC for any input voltage.

The TUE is denoted by E_T. It is represented in *Figure 13*.

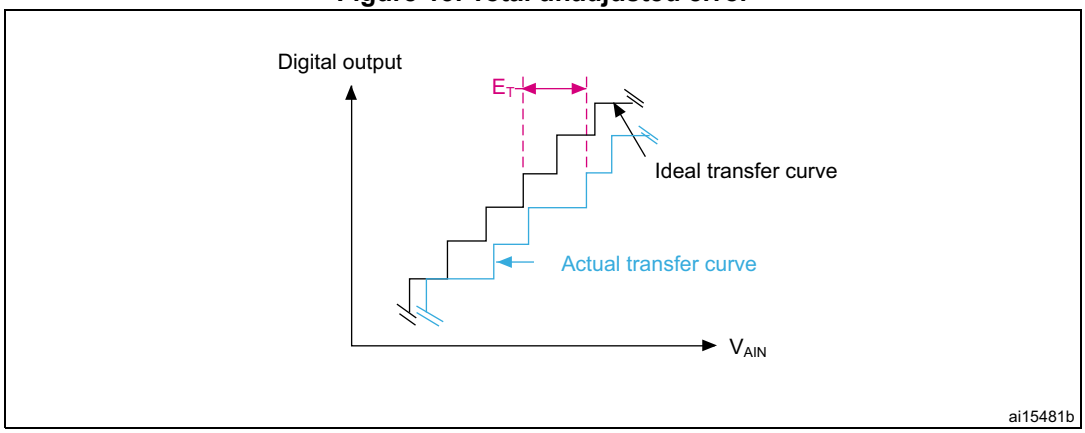
The TUE is not the sum of E_O, E_G, E_L, E_D. The offset error affects the digital result at lower voltages whereas the gain error affects the digital output for higher voltages.

Example

If V_{REF+} = 3.3 V and V_{AIN} = 2 V, the ideal result is 0x9B2. However, if the conversion result of 0x9B4 is obtained, the deviation may result from the offset since the DLE and ILE errors occur simultaneously.

TUE = absolute (actual value – ideal case value) = 0x9B4 – 0x9B2 = 0x2 = 2 LSB

Figure 13. Total unadjusted error



1. The total unadjusted error, E_T, is shown in magenta.

2.2 Errors due to the ADC environment

2.2.1 Reference voltage noise

As the ADC output is the ratio between the analog signal voltage and the reference voltage, any noise on the analog reference causes a change in the converted digital value. V_{DDA} analog power supply is used on some packages as the reference voltage (V_{REF+}), so the quality of V_{DDA} power supply has influence on ADC error.

For example, with an analog reference of 3.3 V (V_{REF+} = V_{DDA}) and a 1 V signal input, the converted result is:

(1/3.3) × 4095 = 0x4D9

However, with a 40 mV peak-to-peak ripple in the analog reference, the converted value becomes:

(1/3.34) × 4095 = 0x4CA (with V_{REF+} at its peak).

Error = 0x4D9 – 0x4CA = 15 LSB

The SMPS (switch-mode power supply) usually embeds internal fast-switching power transistors. This introduces high-frequency noise in the output. The switching noise is in the range of 15 kHz to 1 MHz.

2.2.2 Reference voltage / power supply regulation

Power supply regulation is very important for ADC accuracy since the conversion result is the ratio of the analog input voltage to the V_{REF+} value.

If the power supply output decreases when connected to V_{DDA} or V_{REF+} due to the loads on these inputs and to its output impedance, an error will be introduced in the conversion result.

$\text{Digital code} = \frac{V_{AIN}(2^N - 1)}{V_{REF+}}$, where N is the resolution of the ADC (in our case N = 12).

If the reference voltage changes, the digital result changes too.

For example:

If the supply used is a reference voltage of 3.3 V and V_{AIN} = 1 V, the digital output is:

$$\text{Digital}_{output} = \frac{1 \times (2^{12} - 1)}{3.3} = 0x4D9$$

If the voltage supply provides a voltage equal to 3.292 V (after its output connection to V_{REF+}), then:

$$\text{Digital}_{output} = \frac{1 \times (2^{12} - 1)}{3.292} = 0x4DC$$

The error introduced by the voltage drop is: 0x4DC – 0x4D9 = 3 LSB.

2.2.3 External reference voltage parameters

In case of usage external source for reference voltage (on V_{REF+} pin) there are important parameters of this external reference source. Three reference voltage specifications must be considered: temperature drift, voltage noise, long term stability.

2.2.4 Analog input signal noise

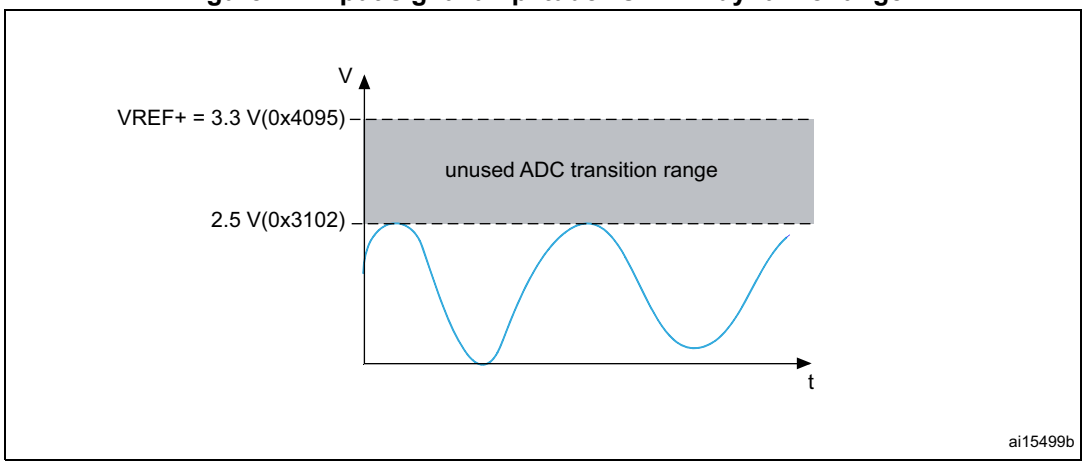
Small but high-frequency signal variation can result in big conversion errors during sampling time. This noise is generated by electrical devices, such as motors, engine ignition, power lines. It affects the source signal (such as sensors) by adding an unwanted signal. As a consequence, the ADC conversion results are not accurate.

2.2.5 ADC dynamic range bad match for maximum input signal amplitude

To obtain the maximum ADC conversion precision, it is very important that the ADC dynamic range matches the maximum amplitude of the signal to be converted. Let us assume that the signal to be converted varies between 0 V and 2.5 V and that V_{REF+} is equal to 3.3 V. The maximum signal value converted by the ADC is 3102 (2.5 V) as shown in *Figure 14*. In this case, there are 993 unused transitions (4095 – 3102 = 993). This implies a loss in the converted signal accuracy.

See *Section 3.2.5: Matching the ADC dynamic range to the maximum signal amplitude on page 26* for details on how to make the ADC dynamic range match the maximum input signal amplitude.

Figure 14. Input signal amplitude vs. ADC dynamic range



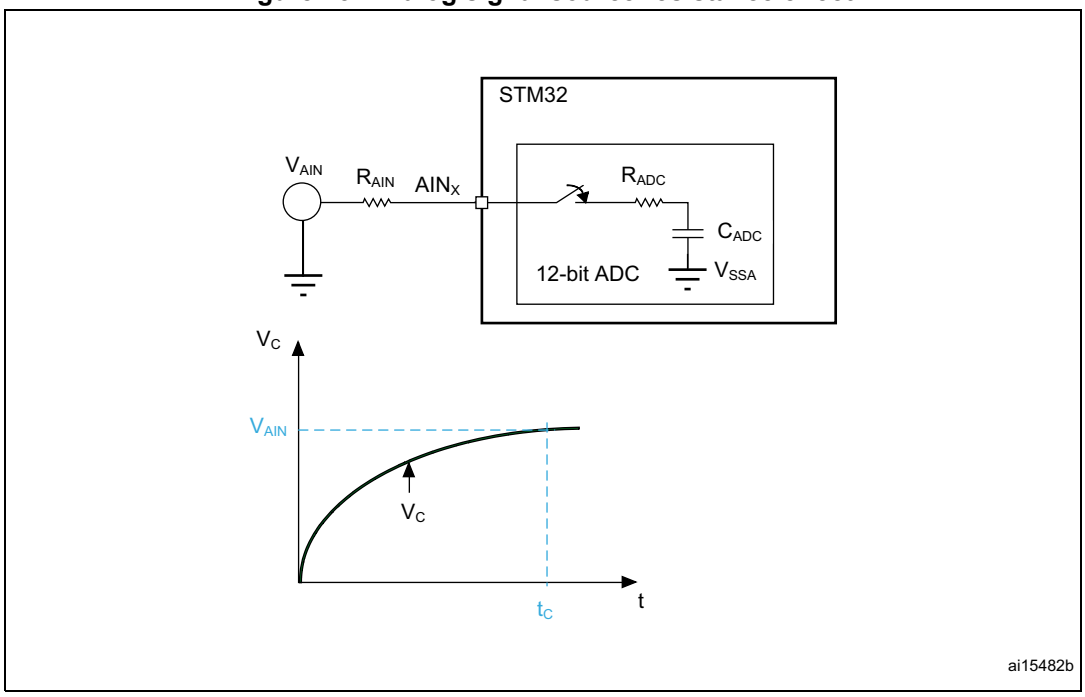
2.2.6 Effect of the analog signal source resistance

The impedance of the analog signal source, or series resistance (R_{AIN}), between the source and pin, causes a voltage drop across it because of the current flowing into the pin. The charging of the internal sampling capacitor (C_{ADC}) is controlled by switches with a resistance R_{ADC}.

With the addition of source resistance (with R_{ADC}), the time required to fully charge the hold capacitor increases. *Figure 15* shows the analog signal source resistance effect.

The effective charging of C_{ADC} is governed by R_{ADC} + R_{AIN}, so the charging time constant becomes $t_c = (R_{ADC}+R_{AIN}) \times C_{ADC}$. If the sampling time is less than the time required to fully charge the C_{ADC} through R_{ADC} + R_{AIN} ($t_s < t_c$), the digital value converted by the ADC is less than the actual value.

Figure 15. Analog signal source resistance effect



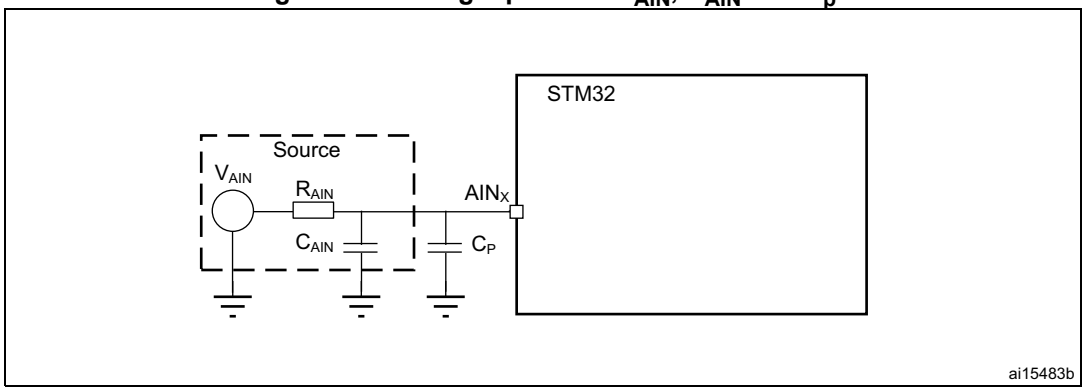
1. t_c is the time taken by the C_{ADC} capacitor to fully charge: $V_c = V_{AIN}$ (with max.1/2 LSB error)
   V_c: capacitor (C_{ADC}) voltage
   $t_c = (R_{ADC} + R_{AIN}) \times C_{ADC}$

2.2.7 Effect of source capacitance and parasitic capacitance of the PCB

When converting analog signals, it is necessary to account for the capacitance at the source and the parasitic capacitance seen on the analog input pin (refer to *Figure 16*). The source resistance and capacitance form an RC network. In addition, the ADC conversion results may not be accurate unless the external capacitor (C_{AIN} + C_p) is fully charged to the level of the input voltage. The greater value of (C_{AIN} + C_p), the more limited the source frequency.

The external capacitance at the source and the parasitic capacitance are denoted by C_{AIN} and C_p, respectively.

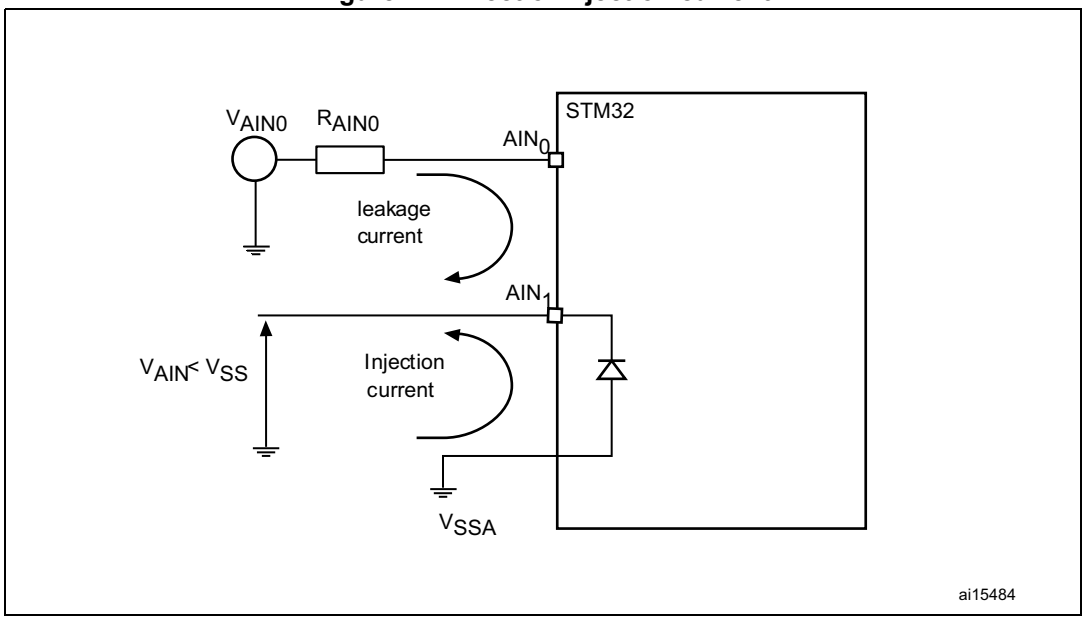
Figure 16. Analog input with R_{AIN}, C_{AIN} and C_p



2.2.8 Injection current effect

A negative injection current on any analog pin (or a closely positioned digital input pin) may introduce leakage current into the ADC input. The worst case is the adjacent analog channel. A negative injection current is introduced when $V_{AIN} < V_{SS}$, causing current to flow out from the I/O pin. This is illustrated in *Figure 17*.

Figure 17. Effect of injection current



2.2.9 Temperature influence

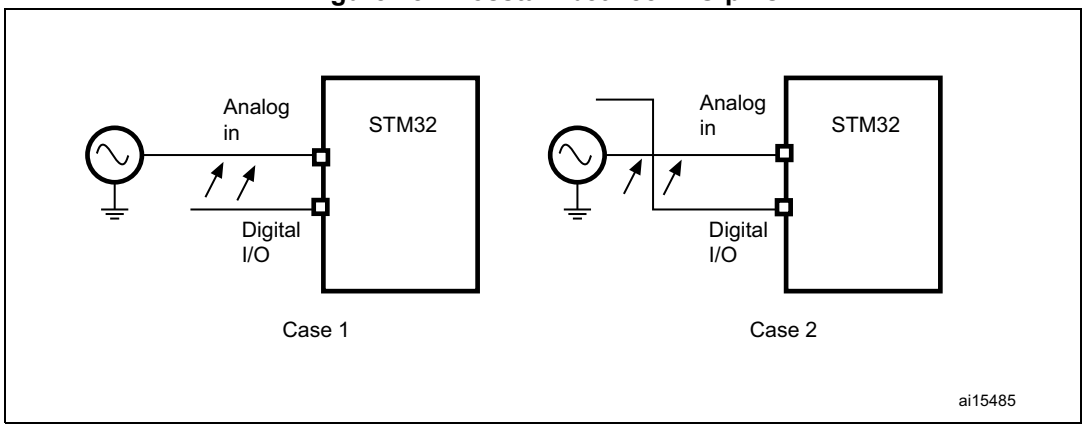
The temperature has a major influence on ADC accuracy. Mainly it leads to two major errors: offset error drift and gain error drift. Those errors can be compensated in the microcontroller firmware (refer to *Section 3.2.8* for the temperature-compensation methods).

2.2.10 I/O pin crosstalk

Switching the I/Os may induce some noise in the analog input of the ADC due to capacitive coupling between I/Os. Crosstalk may be introduced by PCB tracks that run close to each other or that cross each other.

Internally switching digital signals and I/Os introduces high-frequency noise. Switching high-sink I/Os may induce some voltage dips in the power supply caused by current surges. A digital track that crosses an analog input track on the PCB may affect the analog signal (see *Figure 18*).

Figure 18. Crosstalk between I/O pins

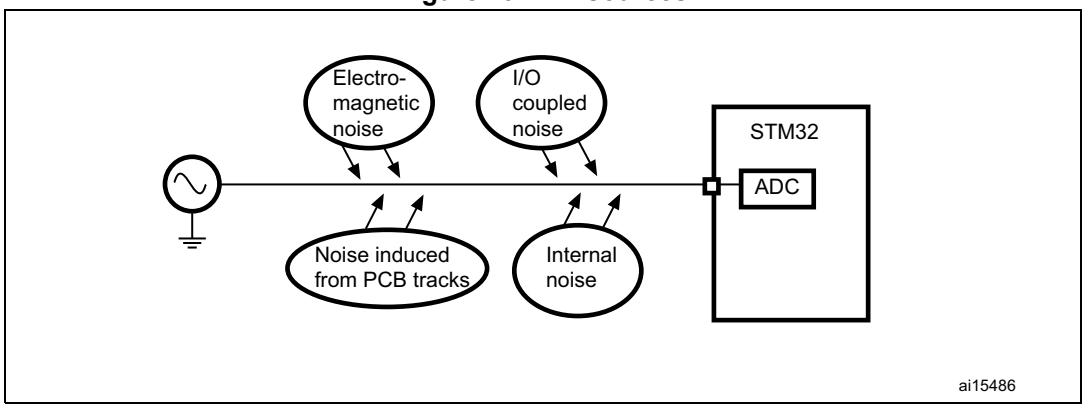


1. Case 1: Digital and analog signal tracks that pass close to each other.
2. Case 2: Digital and analog signal tracks that cross each other on a different PCB side.

2.2.11 EMI-induced noise

Electromagnetic emissions from neighboring circuits may introduce high-frequency noise in an analog signal because the PCB tracks may act like an antenna (See *Figure 19.*).

Figure 19. EMI sources



3 How to get the best ADC accuracy

3.1 Reduce the effects of ADC-related ADC errors

The TUE is not the sum of all the E_O, E_G, E_L, E_D errors. It is the maximum deviation that can occur between the ideal and actual digital values. It can result from one or more errors occurring simultaneously.

As the ILE is the integral of the DLE, it can be considered as the indicator of the maximum error. Do not add the DLE and ILE together to calculate the maximum error that may occur at any digital step.

The maximum error values specified in the datasheet are the worst error values measured in laboratory test environment over the given voltage and temperature range (see device datasheet).

The ILE and DLE are dependent on the ADC design. It is difficult to calibrate them. They can be calibrated by the measured ADC curve stored in the microcontroller memory but this needs calibration of each individual device in final application.

Offset and gain errors can be easily compensated using the STM32 ADC self-calibration feature or by microcontroller firmware.

3.2 Minimize ADC errors related to external environment of ADC

3.2.1 Reference voltage / Power supply noise minimization

Power supply side

Linear regulators have a better output in terms of noise. The mains must be stepped down, rectified and filtered, then fed to linear regulators. It is highly recommended to connect the filter capacitors to the rectifier output. Please refer to the datasheet of the used linear regulator.

If you are using a switching power supply, it is recommended to have a linear regulator to supply the analog stage.

It is recommended to connect capacitors with good high-frequency characteristics between the power and ground lines. That is, a 0.1 µF and a 1 to 10 µF capacitor should be placed close to the power source.

The capacitors allow the AC signals to pass through them. The small-value capacitors filter high-frequency noise and the high-value capacitors filter low-frequency noise. Ceramic capacitors are generally available in small values (1 pF to 0.1 µF) and with small voltage ratings (16 V to 50 V). It is recommended to place them close to the main supply (V_{DD} and V_{SS}) and analog supply (V_{DDA} and V_{SSA}) pins. They filter the noise induced in the PCB tracks. Small capacitors can react fast to current surges and discharge quickly for fast-current requirements.

Tantalum capacitors can also be used along with ceramic capacitors. To filter low-frequency noise, you can use high-value capacitors (10 µF to 100 µF), which are generally electrolytic. It is recommended to put them near the power source.

To filter high-frequency noise, a ferrite inductance in series with the power supply can be used. This solution leads to very low (negligible) DC loss unless the current is high because the series resistance of the wire is very low. At high frequencies, however, the impedance is high.

STM32 microcontroller side

In most STM32 microcontrollers, the V_{DD} and V_{SS} pins are placed close to each other. So are the V_{REF+} and V_{SSA} pins. A capacitor can therefore be connected very close to the microcontroller with very short leads. For multiple V_{DD} and V_{SS} pins, use separate decoupling capacitors.

The V_{DDA} pin must be connected to two external decoupling capacitors (10 nF Ceramic + 1 µF Tantalum or Ceramic). Refer to *Figure 20* and *Figure 21* for decoupling examples.

For STM32 microcontrollers delivered in 100/144-pin packages, it is possible to improve the accuracy on low-voltage inputs by connecting a separate external ADC reference voltage input on V_{REF+} (refer to *Section 3.2.5*). The voltage on V_{REF+} may range from 2.4 V to V_{DDA}. If a separate, external reference voltage is applied on V_{REF+}, two 10 nF and 1 µF capacitors must be connected on this pin. In all cases, V_{REF+} must be kept between 2.4 V and V_{DDA}.

Figure 20. Power supply and reference decoupling for 100- and 144-pin packages

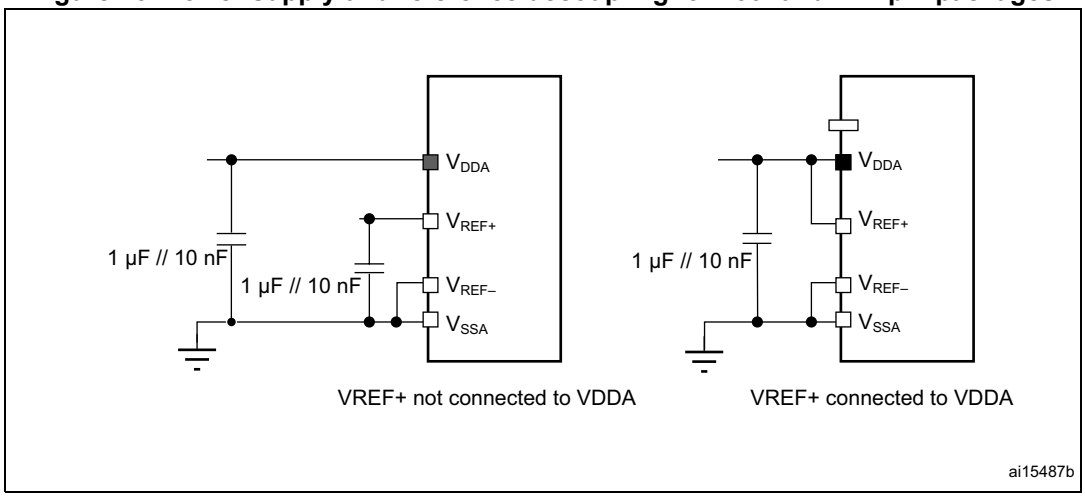
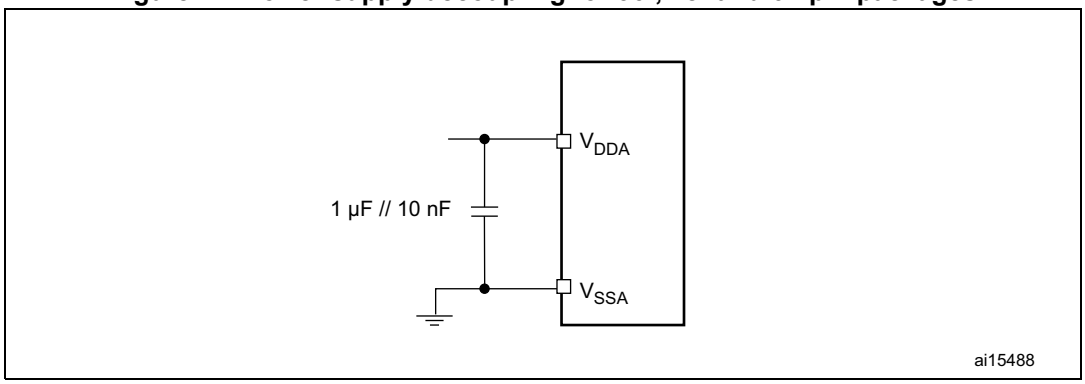


Figure 21. Power supply decoupling for 36-, 48- and 64-pin packages



3.2.2 Reference voltage / Power-supply regulation

The power supply should have good line and load regulation since the ADC uses V_{REF+} or V_{DDA} as the analog reference and the digital value is the ratio of the analog input signal to this voltage reference. V_{REF+} must thus remain stable at different loads.

Whenever the load is increased by switching on a part of the circuit, the increase in current must not cause the voltage to decrease. If the voltage remains stable over a wide current range, the power supply has good load regulation.

For example, for the LD1086D2M33 voltage regulator, the line regulation is 0.035% typical when V_{IN} varies from 2.8 V to 16.5 V (when I_{load} = 10 mA), and the load regulation is 0.2% when I_{load} varies from 0 to 1.5 A (please refer to the LD1086 series datasheet for details).

The lower the line regulation value, the better the regulation. Similarly, the lower the load regulation value, the better the regulation and the stability of the voltage output.

It is also possible to use a reference voltage for V_{REF+}, for instance the LM236, which is a voltage reference diode of 2.5 V (refer to LM236 datasheet for more details).

3.2.3 Analog-input signal noise elimination

Averaging method

Averaging is a simple technique where you sample an analog input several times and take the average of the results by software. This technique is helpful to eliminate the effect of noise on the analog input in case of an analog voltage that does not change often.

The average has to be made on several readings that all correspond to the same analog input voltage. Make sure that the analog input remains at the same voltage during the time period when the conversions are done, otherwise you will add up digital values corresponding to different analog inputs, and you will introduce errors.

In the STM32L0 and STM32L4 microcontrollers, the ADC hardware oversampling feature can be used for averaging. This feature simply performs the sum of a given number of ADC raw samples into one final sample. This final sample can then be right shifted to reduce the bit width caused by multiple ADC samples accumulation. All these operations (accumulation and right-bit shifting) are performed by hardware. The STM32L0 and STM32L4 microcontrollers support hardware oversampling up to 256 input samples.

Adding an external filter

Adding an external RC filter eliminates the high frequency. An expensive filter is not needed to deal with a signal that has frequency components above the frequency range of interest. In this case, a relatively simple low-pass filter with a cutoff frequency f_C just above the frequency range of interest will suffice to limit noise and aliasing. A sampling rate consistent with the highest frequency of interest will suffice, typically two to five times f_C.

Note: The R and C that form the external filter should have values that match the conditions described in Section 3.2.4 and Section 3.2.7.

3.2.4 Adding white noise or triangular sweep to improve resolution

This method combines hardware and software techniques to improve precision. From a software point of view, this method uses averaging (oversampling) and from a hardware point of view, it uses signal modification/spreading/dithering.

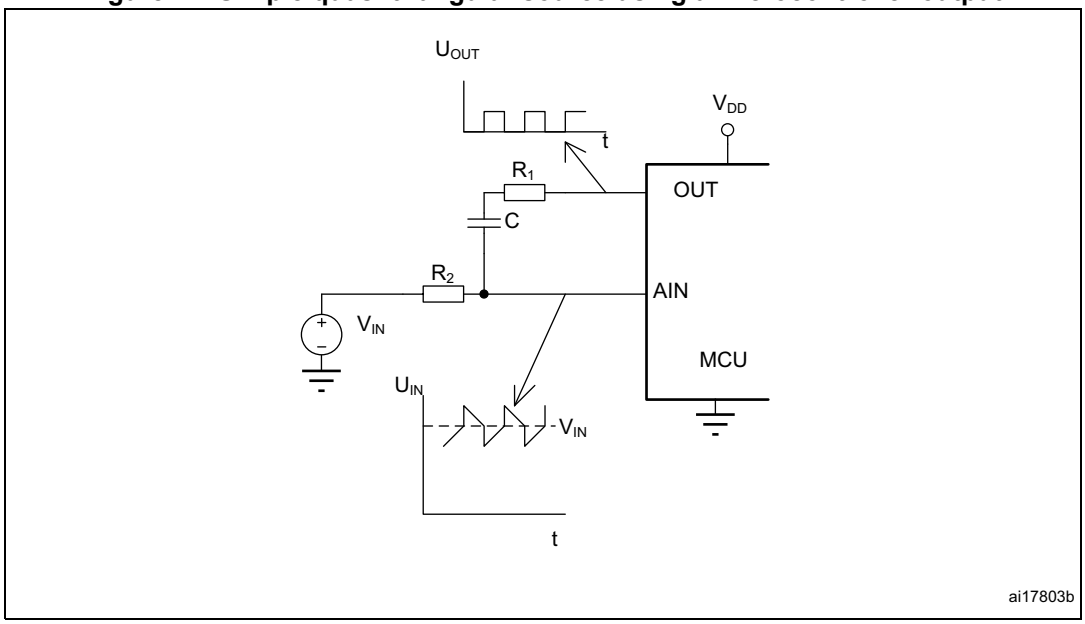
Averaging can be used in cases where the input signal is noisy (some signal change is necessary in order to be able to calculate an average) and the requirement is to obtain the mean value of a signal. A problem appears when the input signal is a very stable voltage without noise. In this case, when the input signal is measured, each data sample is the same. This is because the input signal level is somewhere between two ADC word levels (e.g. between 0x14A and 0x14B). Therefore it is not possible to determine the input voltage level more precisely (e.g. if the level is near to 0x14A or near to 0x14B level).

The solution is to add noise or some signal change (with uniform signal distribution e.g. triangular sweep) to the input signal which pushes its level across 1-bit ADC level (so that the signal level changes below 0x14A and above 0x14B level). This causes the ADC results to vary. Applying software averaging to the different ADC results, produces the mean value of the original input signal. The STM32L0 and STM32L4 microcontrollers feature hardware oversampling, which can be used instead of software oversampling.

As an example, this method can be implemented by using a triangular generator with RC coupling to the input signal (white noise generation is more complicated). Care must be taken not to modify the mean value of the original input signal (so, capacitive coupling must be used).

A very simple implementation of the quasi-triangular source which is generated directly by the STM32 microcontroller is on *Figure 22*.

Figure 22. Simple quasi-triangular source using a microcontroller output



3.2.5 Matching the ADC dynamic range to the maximum signal amplitude

This method improves accuracy by a proper selection of the reference voltage or by using a preamplifier stage to obtain the maximum possible resolution using the full ADC output range.

Selecting a reference voltage (method for devices delivered in 100-/144-pin packages only)

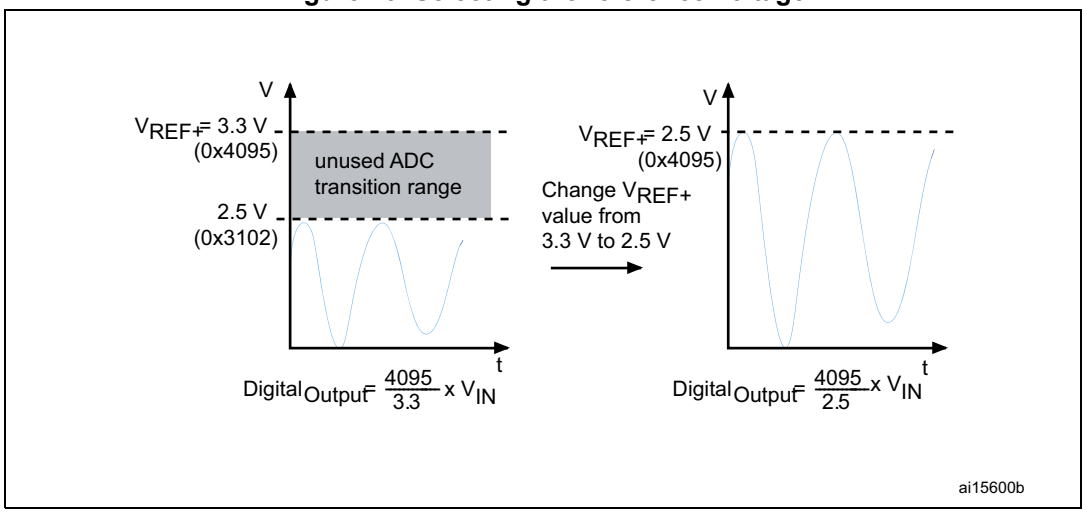
The reference voltage is selected in the expected range of the signal to be measured. If the measured signal has an offset, then the reference voltage should also have a similar offset. If the measured signal has a defined maximum amplitude, then the reference voltage should also have a similar maximum value. By matching this reference voltage to the measurement signal range, we obtain the maximum possible resolution using the full ADC output range.

In STM32 microcontrollers delivered in 100- and 144-pin packages, the ADC reference voltage is connected to the external V_{REF+} and V_{REF-} pins that should be tied to ground. This makes it possible to match the reference voltage and the measured signal range.

For example, if the measured signal varies between 0 V and 2.5 V, it is recommended to choose a V_{REF+} of 2.5 V, possibly using a reference voltage like LM235 (see LM235 datasheet for more details). *Figure 23* illustrates these conditions.

Note: The voltage on V_{REF+} may range between 2.4 V and V_{DDA}.

Figure 23. Selecting the reference voltage



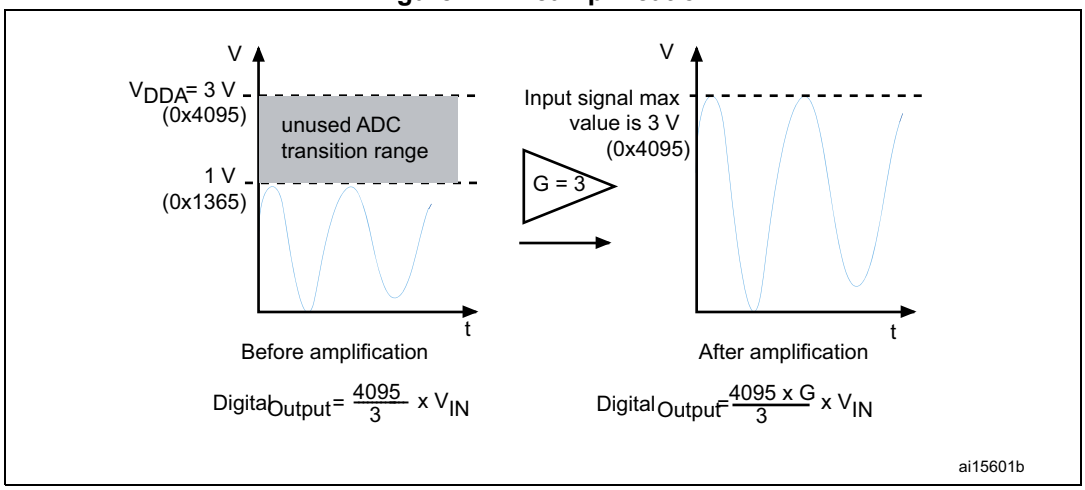
Using a preamplifier

If the measured signal is too small (in comparison with the ADC range), then an external preamplifier can be useful. This method can be implemented whatever the STM32 package, and more specifically in packages that do not have a V_{REF+} input.

For example, if the measured signal varies between 0 V to 1 V and V_{DDA} is set to 3 V, the signal can be amplified so that its peak-to-peak amplitude is similar to the V_{DDA} value. The gain is then equal to 3 (see *Figure 24* for an example).

This amplifier can adapt the input signal range to the ADC range. It can also insert offsets between the input signal and the ADC input. When designing the preamplifier, care must be taken not to generate additional errors (such as additional offset, amplifier gain stability or linearity, frequency response).

Figure 24. Preamplification



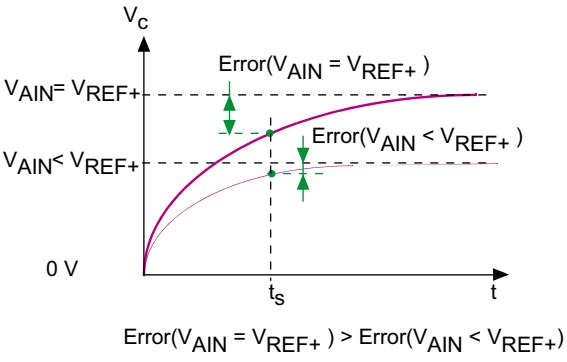
3.2.6 Analog source resistance calculation

Let us assume that the maximum error allowed is equal to 1/2 LSB. We will calculate the maximum source resistance allowed.

V_c is the voltage across the internal C_{ADC} capacitor (refer to *Figure 15*).

Then we have: $\text{Error} = V_{AIN} - V_c = \frac{1}{2}\text{LSB}$

Figure 25. Worst case error: $V_{AIN} = V_{REF+}$



Let t_s be the sampling time.

$t_s = T_S/f_{ADC}$, where T_s is the sampling time evaluated by cycles (1)

For a given t_s, the error corresponding to $V_{AIN} = V_{REF+}$ is greater than the error corresponding to $V_{AIN} < V_{REF+}$ because the C_{ADC} capacitor takes more time to charge from 0 V to V_{AIN} when $V_{AIN} = V_{REF+}$ than it takes when $V_{AIN} < V_{REF+}$ (refer to *Figure 25*). So $V_{AIN} = V_{REF+}$ is the worst case to be taken into account in the demonstration of the maximum source resistance.

$$\text{Error} = V_{REF+} - V_{REF+}\left(1 - e^{-\frac{t_s}{R_{max}C_{ADC}}}\right) = \frac{1}{2} \cdot \frac{V_{REF+}}{2^N}$$

where:

$R_{max} = (R_{AIN} + R_{ADC})\max(2)$

N is the ADC resolution (in our case N = 12)

This gives: $e^{-\frac{t_s}{R_{max}C_{ADC}}} = \frac{1}{2^{N+1}}$. Thus: $R_{max} = \frac{t_s}{C_{ADC} \cdot \ln(2^{N+1})}$ (3)

By combining equations (1), (2) and (3), we obtain final formula:

$$R_{AINmax} = \frac{T_s}{f_{ADC} \cdot C_{ADC} \cdot \ln(2^{N+1})} - R_{ADCmax}$$

Example for STM32F1 parameters

f_{ADC} = 14 MHz, C_{ADC} = 8 pF, R_{ADC}max = 1 kΩ and for T_s = 7.5, the maximum source resistance allowed for an error equal to 1/2 LSB is:

$$R_{AINmax} = \frac{7.5}{14 \cdot 10^{6} \cdot 8 \cdot 10^{-12} \cdot \ln(2^{12+1})} - 1k\Omega$$

That is:

R_{AINmax} = 6.4 kΩ

Note: The use of a follower amplifier can reduce the resistance of the source effect because of its high input impedance and its very low output impedance. It isolates R_{AIN} from R_{ADC}. However, the amplifier introduces an offset error that should be taken into account.

In case of longer sampling times and reduced number of ADC clocks, better results can be obtained. It is possible to further increase the allowed external resistance by decreasing the ADC clock frequency or selecting a lower resolution. Refer to the datasheet of your device to obtain the exact values of the RC parameters.

3.2.7 Source frequency condition vs. source and parasitic capacitors

The external capacitance will not allow the analog input voltage to be exactly the same as V_{AIN} if the capacitor is not fully charged by the analog source.

If the analog input signal changes, then the analog signal frequency (F_{AIN}) should be such that the time period of this analog signal is at least: $10 \times R_{AIN} \times (C_{AIN} + C_p)$.

T_{AIN} = analog signal time period = $1/F_{AIN}$.

We have: $T_{AIN} \geq 10 \times R_{AIN} \times (C_{AIN} + C_P)$

Therefore: $F_{AIN} \leq \frac{1}{10 \times R_{AIN} \times (C_{AIN} + C_P)}$

For example:

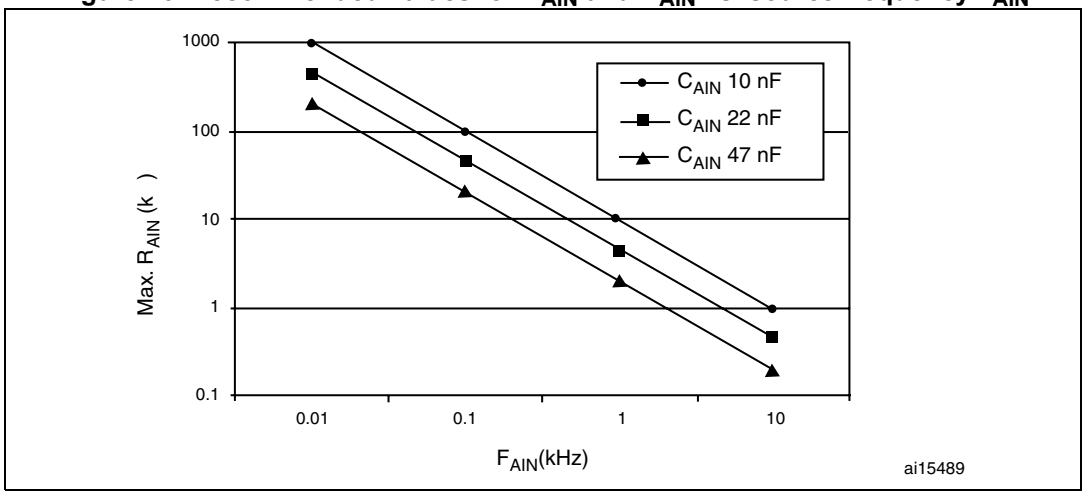
For R_{AIN} = 25 kΩ, C_{AIN} = 7 pF, C_p = 3 pF, this gives:

$$F_{AINmax} = \frac{1}{10 \times 25 \times 10^{3} \times (7+3) \times 10^{-12}}$$

Thus, the maximum frequency of the source will be: $F_{AINmax} = 400\ \text{kHz}$.

So for the above defined source characteristics (capacitance and resistance), the frequency of the source must not exceed 400 kHz, otherwise the ADC conversion result will be not accurate.

Figure 26. Recommended values for R_{AIN} and C_{AIN} vs. source frequency F_{AIN}



3.2.8 Temperature-effect compensation

One method is be to fully characterize the offset and gain drift and provide a lookup table in memory to correct measurement according to temperature change. This calibration involves additional cost and takes time.

The second method consists in recalibrating the ADC when the temperature change reaches given values, by using the internal temperature sensor and the ADC watchdog.

3.2.9 Minimizing injection current

Check the application to verify whether any digital or analog input voltage can be less than V_{SS} or V_{SSA}. If it is the case, a negative injection current will flow from the pins. The effect on the accuracy will be greater if a digital input is close to the analog input being converted.

Negative current injection on any of the standard (non-robust) analog input pins should be avoided as this would significantly reduce the accuracy of the conversion being performed on another analog input.

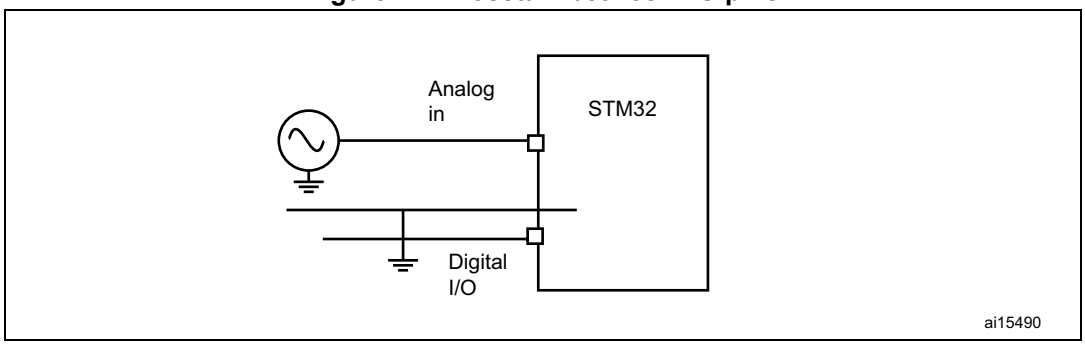
It is recommended to connect a Schottky diode between V_{SSA} and the I/O pin that can give birth to the negative injection current.

The ADC accuracy will not be affected by positive injection currents within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ (refer to the appropriate STM32 datasheet, I/O port characteristics section).

3.2.10 Minimizing I/O pin crosstalk

The noise produced by crosstalk can be reduced by shielding the analog signal by placing ground tracks across it. *Figure 27* shows the recommended grounding between signals.

Figure 27. Crosstalk between I/O pins



3.2.11 EMI-induced noise reduction

You can reduce EMI noise using proper shielding and layout techniques. The possible sources of emission must be physically separated from the receptors. They can be separated electrically by proper grounding and shielding.

Shielding technique

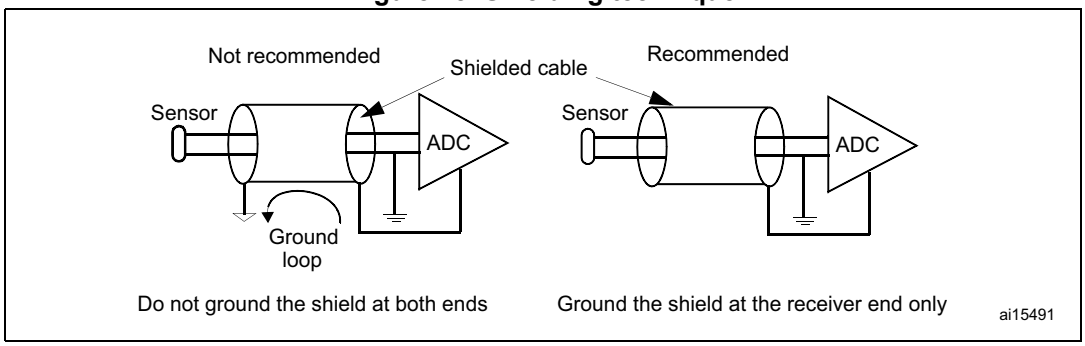
Placing ground tracks alongside sensitive analog signals provides shielding on the PCB. The other side of the two-layer PCB should also have a ground plane. This prevents interference and I/O crosstalk affecting the signal (see *Figure 28*).

Signals coming from distant locations (such as sensors) should be connected to the PCB using shielded cable. Care should be taken to minimize the length of the paths of these types of signal on the PCB.

The shield should not be used to carry the ground reference from the sensor or analog source to the microcontroller. A separate wire should be used as ground. The shield should be grounded at only one place near the receiver such as the analog ground of the microcontroller. Grounding the shield at both ends (source and receiver) might lead to the creation of ground loops, with the result of current flowing through the shield. In this case, the shield acts like an antenna and the purpose of the shielding is lost.

The shielding concept also applies to grounding the chassis of the application if it is metallic. And it also helps remove EMI and EMC interference. In this case the mains earth ground is used to shield the chassis. Similarly DC ground can be used for shielding in case of the earth ground not being available.

Figure 28. Shielding technique



3.2.12 PCB layout recommendations

Separating the analog and digital layouts

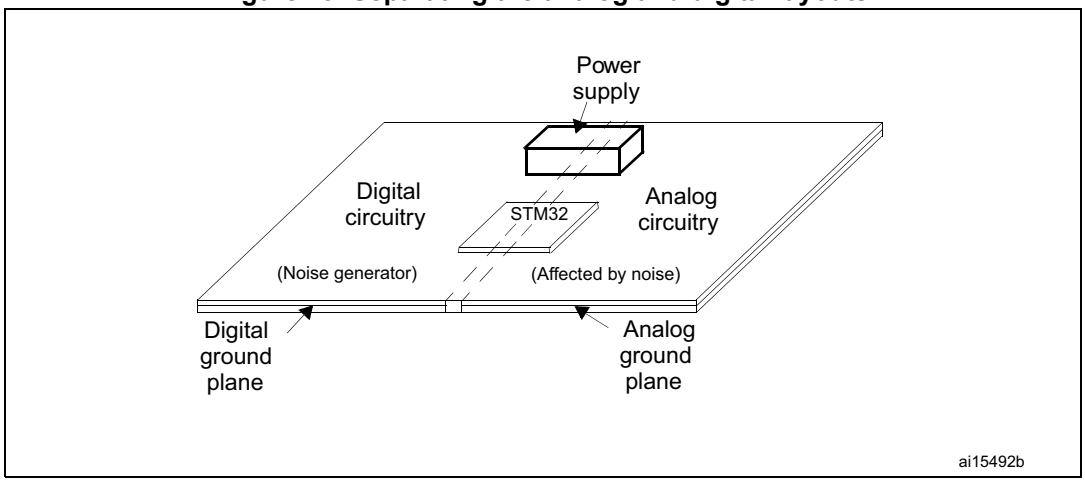
It is recommended to separate the analog and digital circuitry on the PCB (see *Figure 29*). This also avoids tracks crossing each other. The tracks carrying digital signals may introduce high-frequency noise in analog signals because of coupling.

The digital signals produce high-frequency noise because of fast switching.

Coupling of a capacitive nature is formed due to the metal connections (tracks) separated by the dielectric provided by the PCB base (glass, ceramic or plastic).

It is recommended to use different planes for analog and digital grounds. If there is a lot of analog circuitry then an analog ground plane is recommended. The analog ground must be placed below the analog circuitry.

Figure 29. Separating the analog and digital layouts

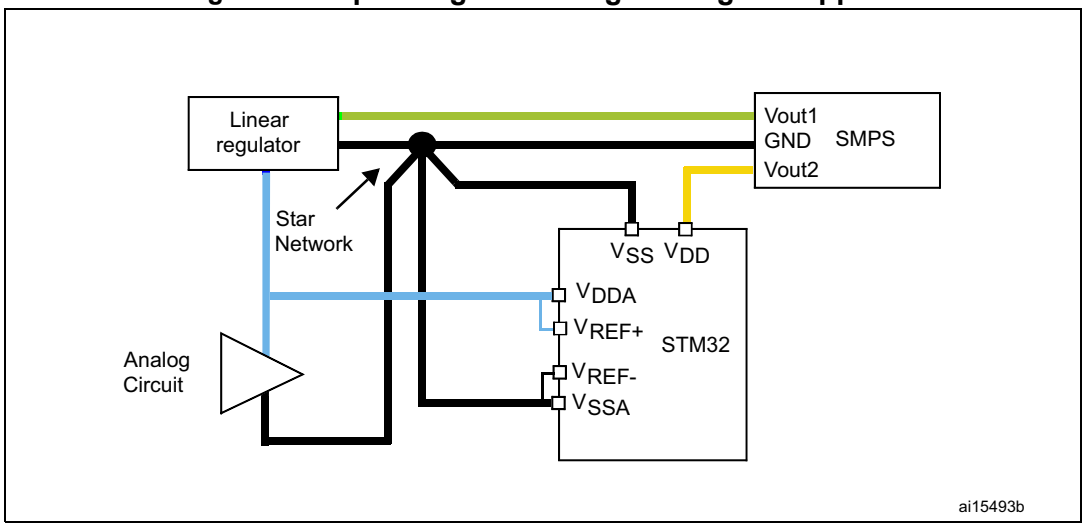


Separating the analog- and digital-circuit power supplies

It is desirable to have separate analog and digital power supplies in cases where there is a lot of analog and digital circuits external to the microcontroller (see *Figure 30*). Depending on the STM32 package, different analog and digital power supply and ground pins are available. The V_{DDA}/V_{REF+} and V_{DD} pins can be powered from separate power supplies.

If you use a switching-type power supply for the digital circuitry, you should use a separate linear supply for the analog circuit. Also, if you expect a lot of noise on the DC power supply due to I/O switching etc., it is preferable to use a separate supply for the analog circuit.

Figure 30. Separating the analog and digital supplies



It is also recommended to connect the analog and digital grounds in a star network. This means that you must connect the analog and digital grounds at only one point. This prevents the introduction of noise in the analog power supply circuit due to digital signal switching. This also prevents current surges from affecting the analog circuit.

Using separate PCB layers for the supply and ground

- Two-layer PCBs

  For two-layer PCBs, it is recommended to provide a maximum ground plane area. The power supply (V_{DD}, V_{DDA}) should run through thick tracks. The two layers can have their ground shorted together via multiple connections in the overlap region if the two layers feature the same ground signals. The unused PCB area can be used as the ground plane.

  The other convention is to connect the unused PCB area on one layer to the positive supply (V_{DD}) and the unused area on the other layer, to ground. The advantage is a reduced inductance for power and ground signals. The maximum ground area provided for ground on the PCB results in a good shielding effect and reduces the electromagnetic induction susceptibility of the circuit.

- Multilayer PCBs

  Wherever possible, try to use multilayer PCBs and use separate layers on the PCB for power and ground. The V_{DD} and V_{SS} pins of the various devices can be directly connected to the power planes, thus reducing the length of track needed to connect the supply and ground. Long tracks have a high inductive effect. The analog ground can be connected at one point to this ground plane. If so, it should be close to the power supply.

  A full ground plane provides good shielding and reduces the electromagnetic induction susceptibility of the circuit.

- Single-layer PCBs

  Single-layer PCBs are used to save cost. They can be used only in simple applications when the number of connections is very limited. It is recommended to fill the unused area with ground. Jumpers can be used to connect different parts of the PCB.

3.2.13 Component placement and routing

Place the components and route the signal traces on the PCB so as to shield analog inputs.

Components like resistors and capacitors must be connected with very short leads. You can use surface-mounted device (SMD) resistors and capacitors. You can place SMD capacitors close to the microcontroller for decoupling purposes.

Use wide tracks for power, otherwise the series resistance of the tracks would cause a voltage drop. Indeed, narrow power tracks have a non-negligible finite resistance, so that high load currents through them would cause a voltage drop across them.

Quartz crystals must be surrounded by ground tracks/plane. The other side of the two-layer PCB below the crystal should preferably be covered by the ground plane. Most crystals have a metallic body that should be grounded. You should also place the crystal close to the microcontroller. You can use a surface-mounted crystal.

3.3 Software methods to improve precision

- Averaging samples:
  - Averaging decreases speed but can improve accuracy
- Digital filtering (50/60 Hz suppression from DC value)
  - A proper sampling frequency is set (the trigger from timer is useful in this case).
  - Software post-processing is performed on sampled data (e.g.comb filter for 50 Hz noise and its harmonics suppression).
- Fast Fourier Transform (FFT) for AC measurements
  - This methods allows showing harmonic parts in measured signal.
  - It is slower due to the use of more computation power.
- ADC calibration: offset, gain, bit weight calibration

  ADC calibration decreases internal ADC errors. However, the internal ADC structure must be known.
- Minimizing the internal noise generated by CPU

  The application has to be designed
  - to use minimum disturbance from the microcontroller during ADC conversion.
  - to minimize digital signal changes during sampling and conversion (digital silence).

3.3.1 Averaging samples

The principle of this method is to increase ADC precision but decrease ADC conversion speed (oversampling). If the measured analog signal produces unstable ADC values, then the mean value of the given input signal can be obtained by averaging a set of values. Variation can be caused by signal noise or noise generated by the microcontroller itself (high speed digital signals capacitively coupled to the analog input signal).

Averaging is performed by choosing an appropriate number of samples to be averaged. This number depends on the required precision, minimum conversion speed and the level of other ADC errors (if another error has a greater influence on ADC precision, then increasing the number of averaging values has no effect on total measurement precision).

In STM32L0 and STM32L4 microcontrollers, averaging can be performed by using the hardware oversampling feature: the ADC performs built-in hardware averaging according to configurable parameters (number of samples to average and final right bit shift of result).

The advantage of averaging is to improve ADC precision without any hardware changes. The drawback is that the conversion speed is lower as well as the frequency response (it is equivalent to decreasing effective sampling frequency).

3.3.2 Digital signal filtering

This method uses digital signal processing techniques.

In principle, averaging is also a simple digital filter with a specific frequency response. However if the noise frequency spectrum is known, a digital filter can be designed which minimizes noise influence and maximizes ADC frequency response. For example, if the noise in the measured signal is coming from the 50 Hz power lines, then an appropriate digital filter suppresses only the 50 Hz frequency and delivers data signal without this noise.

The disadvantage of this method is that it requires appropriate microcontroller processing power and resources: CPU speed and data/program memory usage.

3.3.3 FFT for AC measurement

In some specific cases the application needs to know the amplitude of an AC signal with a given frequency. In this case the effective value of an AC signal can also be obtained by using a relatively slow sampling speed (in comparison to the measured signal frequency). For example, when measuring an AC mains signal (which is near-to-sinusoidal and has relatively low harmonics content), it is sufficient to choose a sampling frequency 32 times greater than the mains frequency (50 Hz). In this case harmonics of up to the 15th order can be obtained. The amplitude of 15th harmonics in the main signal is very small (the next order harmonics can be neglected). The calculated effective value of the mains signal is obtained with high precision because the effective values of harmonics are added to the total AC harmonic value as:

$$U_{ef} = \sqrt{U_1^2 + U_2^2 + \ldots + U_n^2}$$

So if the 15th harmonics amplitude is only 1% (0.01) from the 1st harmonics (50 Hz), then its contribution to the total effective value will be only 0.01% (because the square addition in the above equation gives: $0.01^2 = 0.0001$).

The principle of this method is therefore to sample the AC signal with a known frequency and then perform post-processing on the FFT for each measured period. Because the number of sampling points per measured signal period is small (32 points for example) then the performance needed for FFT processing is not so high (only 32-point FFT for example).

This method is well adapted for AC measurement of signals with lower distortion. The drawback is that it requires precise signal sampling:

- The frequency of the measured signal must be known and the ADC sampling frequency must be set exactly as a 2^n multiplier of the measured frequency.
- The input signal frequency is measured by another method.
- The ADC sampling frequency is tuned by programming the prescaler and MCU master clock selection (if sampling is performed with an inaccurate clock an interpolation can be used to obtain samples at the required points).

3.3.4 ADC calibration

This method requires knowledge of the internal ADC structure and of how the ADC converter is implemented inside the microcontroller. This is necessary to design a physical/mathematical model of the ADC implementation.

A proper physical model (which is usually a schematic diagram) is used as a basis for describing it mathematically. From the mathematical model, each element in the model can be obtained by a set of equations (for example, resistor/capacitor values which represent bit weights). To solve these equations, it is necessary to perform a set of practical measurements and obtain a set of solvable equations.

From the measured values and mathematical computation of the model, all the known values of model elements (resistors, voltages, capacitors,...) can be put into the schematic diagram.

As a result, instead of the ADC schematic with the designed values, an ADC schematic with the real values for a given microcontroller can be obtained.

Computed model parameters are stored in the microcontroller memory after calibration and used in post-processing to correct ADC values.

3.3.5 Minimizing internal CPU noise

When the CPU operates, it generates a lot of internal and external signal changes which are transferred into the ADC peripheral through capacitive coupling. This disturbance influences ADC precision (unpredictable noise due to different microcontroller operations).

To minimize influences of the CPU (and of other peripherals) on ADC, it is necessary to minimize digital signal changes during sampling and conversion time (digital silence). This is done using one of the following methods (applied during sampling and conversion time):

- minimizing I/O pin changes
- minimizing internal CPU changes (CPU stop, wait mode)
- stopping clock for unnecessary peripherals (timers, communications...)

3.4 High impedance source measurement

This section describes the ADC measurement behavior of STM32 ADC when a signal source with high internal impedance is used. It explains how to design an application to reach the requested precision and provides workarounds.

3.4.1 ADC input stage problem

The ADC embedded in STM32 devices is a switched-capacitor ADC. Switched capacitors work also as sampling capacitors (see *Section 1.1* for a detailed explanation).

When a signal comes from a voltage source with high internal impedance (for instance, 150 kΩ), an additional error can be seen in measurement results. Error signals have also been observed on the ADC input pin, as shown in *Figure 33* (if the voltage source has zero voltage: U_{in} = 0 V, R_{in} = 150 kΩ, C_{ext} = 0 pF):

Figure 31. Typical voltage source connection to ADC input

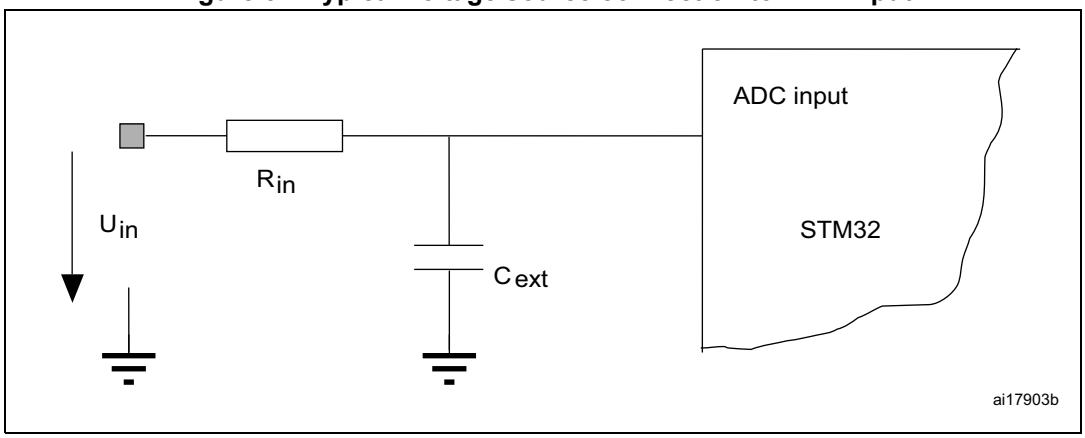
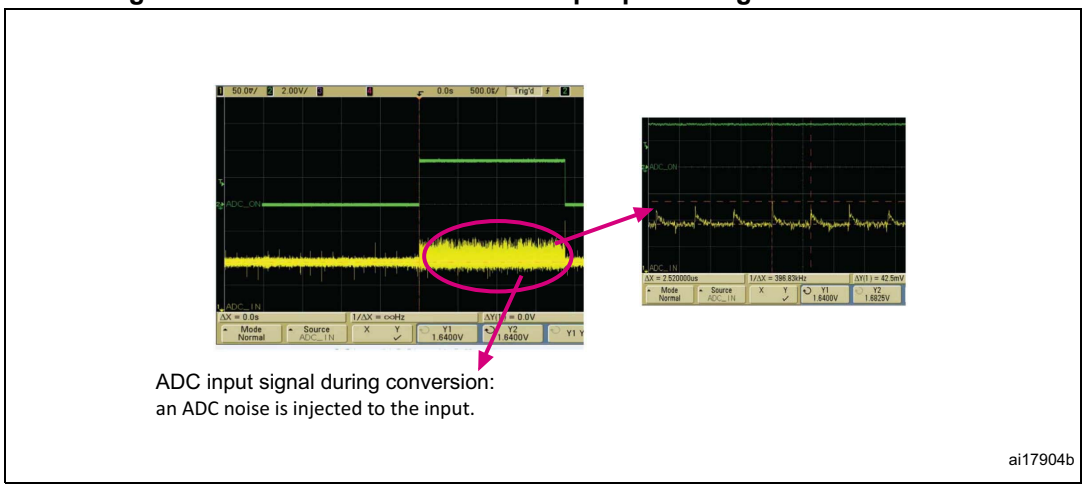


Figure 32. Noise observed on ADC input pin during ADC conversions

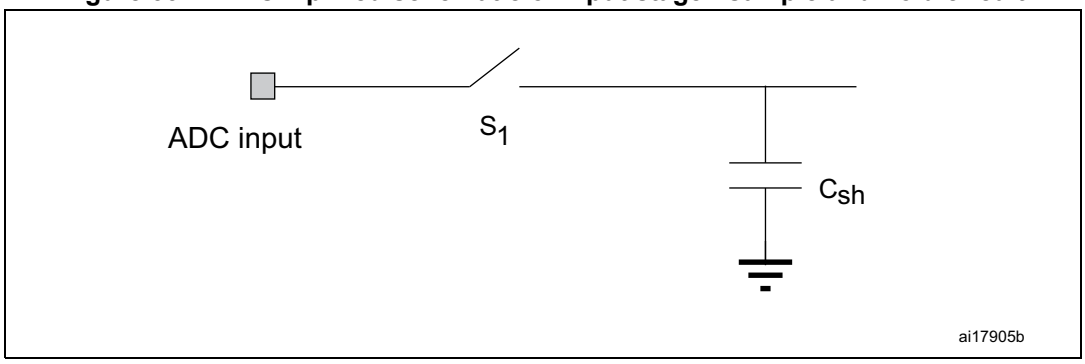


3.4.2 Explanation of the behavior

The explanation of this additional pin noise and additional measurement error (in case a signal source with high internal impedance is used) comes from the internal ADC structure: its input sampling circuit.

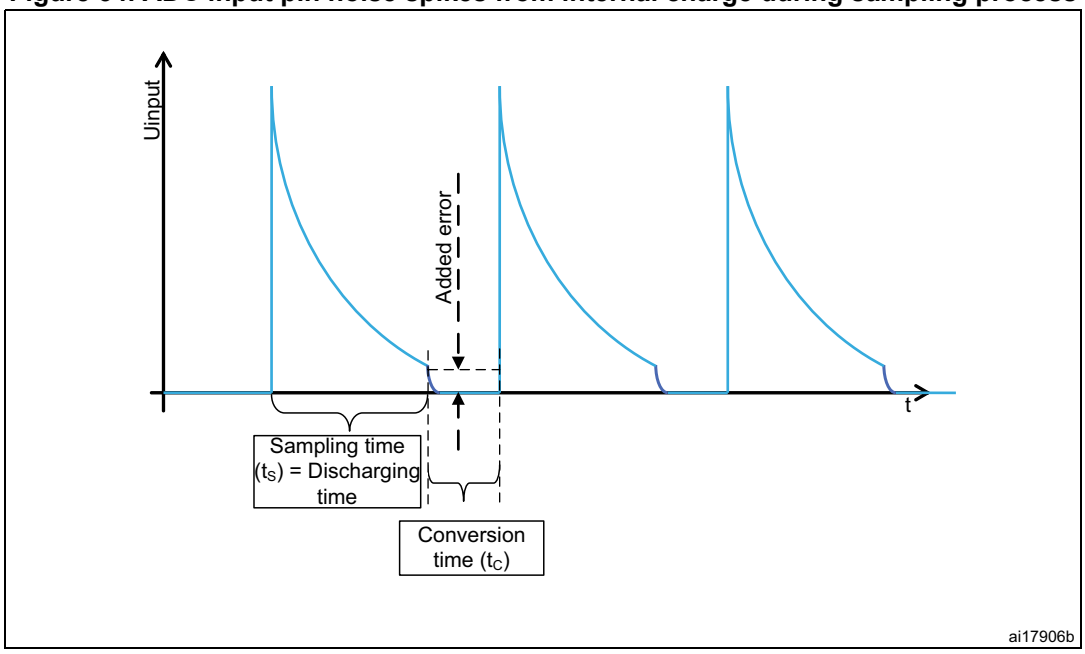
Figure 33 shows a simplified schematic of the input stage (sample and hold circuit).

Figure 33. ADC simplified schematic of input stage - sample and hold circuit



The spikes (noise) present on ADC input pin during conversions are related to the sampling switch (S_1). If the switch is closed, some charge (coming from the sample and hold capacitor C_{sh} or caused by another effect) is transferred to the input pin. Then this charge starts discharging through the source impedance (R_{in}). The discharge process ends at the end of the sampling time (t_S) when the switch S_1 is opened. The remaining undischarged voltage remains on the capacitor C_{sh} and ADC measures this voltage. If the sampling time (t_S) is too short, the remaining voltage does not drop under 0.5 LSB and ADC measurement shows an additional error. *Figure 34* illustrates this process.

Figure 34. ADC input pin noise spikes from internal charge during sampling process



Note that a non-zero external capacitance C_{ext} (parasitic pin capacitance) also exists, so during conversion time the pin capacitance is discharged through source impedance R_{in}.

3.4.3 Minimizing additional errors

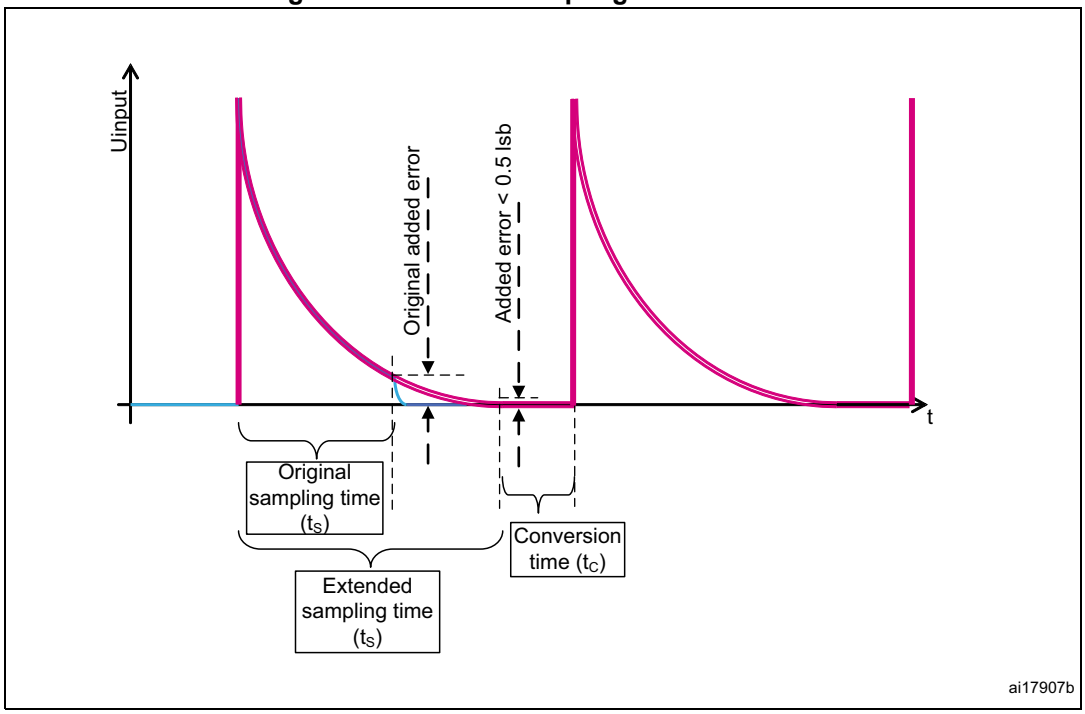
Workaround for high impedance sources

To solve the additional error problem, the sampling time (T_S) can be increased by configuring ADC settings in MCU firmware, so that the C_{sh} charge is discharged through the source impedance R_{in}. The time constant (R_{in} x C_{sh}) is the reference for choosing the sampling time. To calculate the sampling time cycles, use this formula (for a maximum error of 1/2 LSB, see also *Section 3.2.6*):

$$T_S \geq f_{ADC} \cdot (R_{in} \cdot C_{sh}) \cdot \ln(2^{N+1}) \quad [\text{cycles}]$$

The ADC clock (f_{ADC}) is another important factor, since slowing down the ADC clock increases the sampling time.

Figure 35. Effect of sampling time extension



If the maximum register value of the sampling time (T_S) setting is reached and the problem is still present, you need a more complex solution which is applicable also for measurements of source with extra high internal impedance (see *Section : Workaround for extra high impedance sources*).

Note that for this application you must take into account not only the internal sampling capacitance, but also any external parasitic capacitance (in parallel to C_{ext}), such as pin capacitance or PCB path capacitance.

Do not add any external capacitor (C_{ext}) to the input pin when applying this above workaround. Its capacity will increase the timing constant (R_{in} x C_{sh} || C_{ext}) and the problem will remain.

Workaround for extra high impedance sources

This workaround combines both hardware and software changes.

Hardware change

The hardware change consists in adding a large external capacitor (C_{ext}) to the input pin. The capacity size connected to the input pin must reach the value that causes the discharging of the internal sampling capacity C_{sh} to the external capacitor C_{ext} without increasing the voltage on C_{ext} to more than 0.5 LSB.

Example

If the internal capacitor (C_{sh} = 16 pF) is charged to full scale (U_{max}, which corresponds to 4096 LSB), then the external capacitor C_{ext} must be charged at maximum 0.5 LSB voltage level (U_{lsb}) after discharging C_{sh} to it. The capacity of C_{ext} will then be:

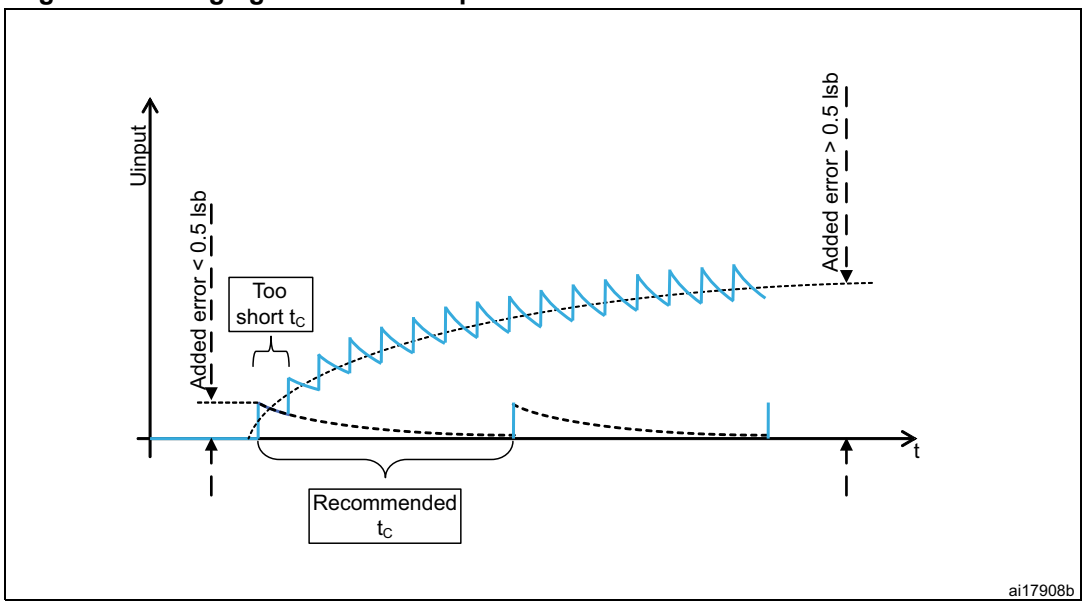
$$C_{ext} \geq C_{sh} \cdot \frac{U_{max}}{U_{lsb}} = 16pF \cdot \frac{4096}{0.5} \approx 131nF$$

The closest larger standard value chosen here is: C_{ext} = 150 nF.

If the internal sampling capacitor C_{sh} is not charged to full voltage range (4096 level) before sampling, the C_{ext} value can be computed by replacing “4096” in the formula above. Calculating with 4096 level gives precise measurement results also in the case of ADC input channels switching (C_{sh} was charged from different ADC input in the previous measurement).

A side effect of this hardware workaround is the cyclical charging of C_{ext} which must be taken into account. Each ADC conversion transfers charge from C_{sh} to C_{ext}. One transfer charges the C_{ext} below 0.5 LSB, as described above, but more transfers can charge C_{ext} to larger values if it is not discharged between two conversions. *Figure 36* shows an example of this scenario where the ADC measurement is performed faster.

Figure 36. Charging the external capacitor with too short time between conversions



Software change

The side effect mentioned above can be solved by software. The objective is to create a delay in order to let C_{ext} discharge through R_{in} (not measure so often) giving enough "discharge time" between ADC conversions. The "discharge time" (t_C) is equal to the transferred charge from C_{sh} to C_{ext} (charging) and from C_{ext} to R_{in} (discharging). The assumption is that $C_{ext} >> C_{sh}$.

$$Q_{charging} = Q_{sh} = C_{sh} \cdot U_{max}$$

$$Q_{discharging} = \frac{U_{lsb}}{R_{in}} \cdot \int_0^{t_C} e^{-\frac{t}{R_{in}C_{ext}}} dt$$

where:

U_{lsb} ..... 0.5 LSB voltage level

U_{max} ..... 4096 LSB voltage level (worst case)

$Q_{charging} = Q_{discharging}$

$$C_{sh} \cdot U_{max} = \frac{U_{lsb}}{R_{in}} \cdot \int_0^{t_C} e^{\frac{t}{R_{in}C_{ext}}} dt$$

Simplification of the above formula gives the final formula for the required waiting time between conversions:

$$t_C = -(R_{in} \cdot C_{ext}) \cdot \ln\left[1 - \frac{C_{sh}}{C_{ext}}\frac{U_{max}}{U_{lsb}}\right]$$

This final formula shows dependency between the external capacitor C_{ext} and the required waiting time between two conversions if the precision U_{lsb} is needed.

From the same formula you can see that the argument in logarithm must be positive and therefore there is a condition for the minimal value of C_{ext}:

$$\left[1 - \frac{C_{sh}}{C_{ext}}\frac{U_{max}}{U_{lsb}}\right] > 0$$

$$1 > \frac{C_{sh}}{C_{ext}}\frac{U_{max}}{U_{lsb}}$$

$$C_{ext} > C_{sh} \cdot \frac{U_{max}}{U_{lsb}}$$

Choosing a larger C_{ext} decreases more the time between conversions (t_C).

An extra large C_{ext} ($C_{ext} >> C_{sh} \cdot \frac{U_{max}}{U_{lsb}}$) enables sampling more often.

However, increasing C_{ext} limits the frequency bandwidth of measurement signal (increasing the “external” timing constant $R_{in} \cdot C_{ext}$).

The formulas below show how to choose the optimal C_{ext} value: signal bandwidth in correlation with sample time. Signal bandwidth is characterized by an “external” timing constant, so optimal solution is to charge C_{ext} during t_C:

$$(R_{in} \cdot C_{ext}) = t_C$$

$$(R_{in} \cdot C_{ext}) = -(R_{in} \cdot C_{ext}) \cdot \ln\left[1 - \frac{C_{sh}}{C_{ext}} \frac{U_{max}}{U_{lsb}}\right]$$

$$-1 = \ln\left[1 - \frac{C_{sh}}{C_{ext}} \frac{U_{max}}{U_{lsb}}\right]$$

$$e^{-1} = 1 - \frac{C_{sh}}{C_{ext}} \frac{U_{max}}{U_{lsb}}$$

After simplification we obtain the final formula for optimal C_{ext}:

$$C_{ext} = \frac{C_{sh} \frac{U_{max}}{U_{lsb}}}{1 - e^{-1}} \approx 1,58 \cdot C_{sh} \frac{U_{max}}{U_{lsb}}$$

and the corresponding waiting time between conversions:

$$t_C \approx -(R_{in} \cdot C_{ext}) \cdot \ln\left[1 - \frac{1}{1,58}\right] \approx (R_{in} \cdot C_{ext})$$

Practically the firmware must not program the ADC in continuous mode but only in single mode and must ensure that there will be a time gap between conversions with duration equal to t_C. This adding of waiting time is the software change which must be applied together with the hardware change (adding an external capacitor C_{ext}).

Without implementation of t_C waiting time in software (for instance, running a conversion just after the first one) the external capacitor C_{ext} will be cyclically charged from the C_{sh} capacitor. After a lot of cycles the voltage on C_{ext} will reach a quite high error value (as previously shown in *Figure 36*).

A practical example of implementation for STM32L1 ADC is shown below:

C_{sh} = 16 pF ..... ADC property

R_{in} = 150 kΩ ..... signal source property

U_{max} = 4096 LSB ..... ADC property

U_{lsb} = 0.5 LSB ..... required precision

$$C_{ext} = 1,58 \cdot C_{sh}\frac{U_{max}}{U_{lsb}} = 1,58 \cdot 16pF \cdot \frac{4096}{0.5} \approx 207nF \Rightarrow 220nF$$

$$t_C = -(R_{in} \cdot C_{sh}) \cdot \ln\left[1 - \frac{C_{sh}}{C_{ext}}\frac{U_{max}}{U_{lsb}}\right] = -(150k\Omega \cdot 220nF) \cdot \ln\left[1 - \frac{16pF}{220nF}\frac{4096}{0.5}\right] \approx 29891\mu s \Rightarrow 30ms$$

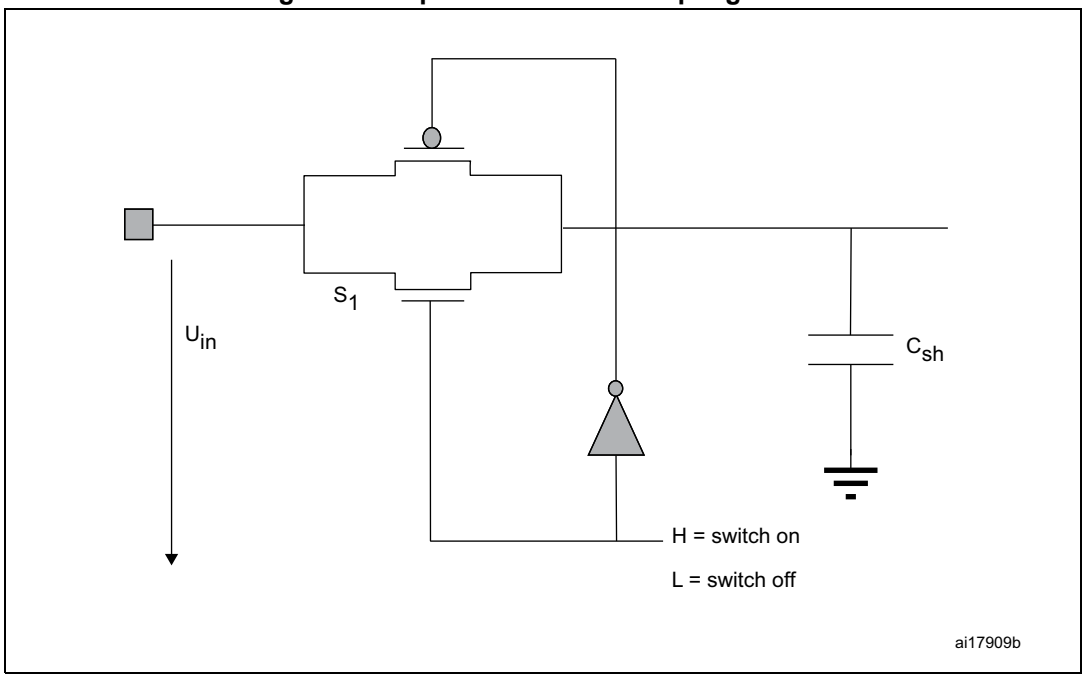
3.4.4 Source of described problem - ADC design

The following sections list some possible causes for the charging of the internal sampling capacitor C_{sh}. This is not an exhaustive list; only the main possible sources of the ADC design are mentioned.

Parasitic switch capacitance effect

The sampling switch inside ADC sampling circuit (see *Figure 33*) is not ideal. In reality the sample and hold switch (S_1) is designed as 2 transistors (PMOS and NMOS, see *Figure 37*):

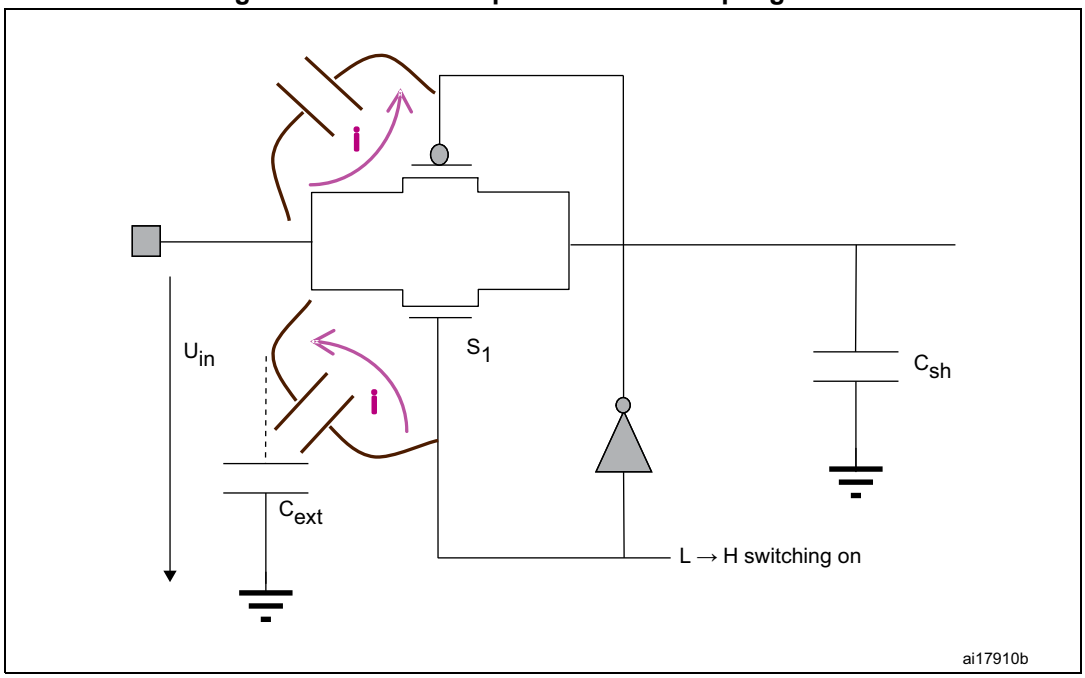
Figure 37. Implementation of sampling switch



The switch is controlled by the gate voltages of transistors (inverted signal on PMOS transistor). This design is a standard bidirectional switch (for rail to rail range of input U_{in} voltages). Both transistors have parasitic capacitances between gate and source.

If those capacitances are charged (close to the switch), then their charge can be transferred to the sampling capacitor (see *Figure 38*).

Figure 38. Parasitic capacitances of sampling switch



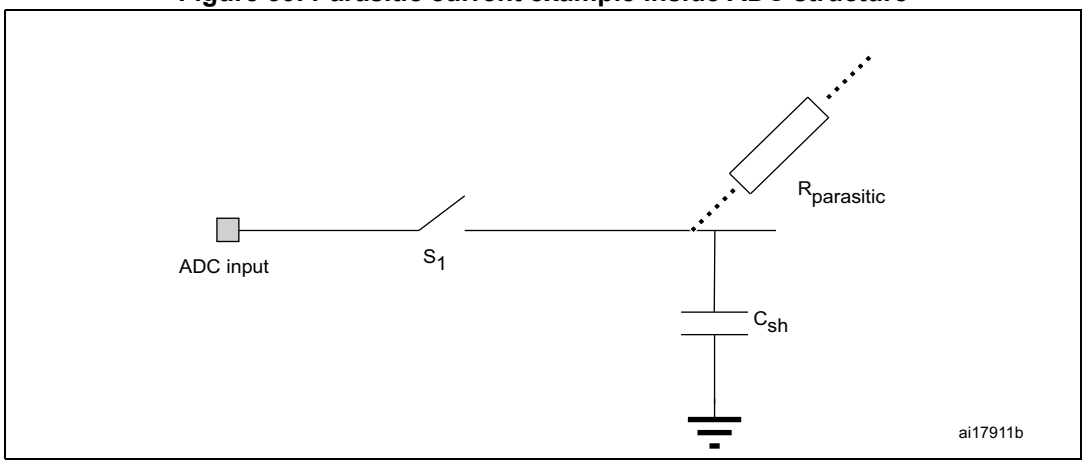
This charging and discharging currents (PMOS and NMOS asymmetric capacitances) can cause charge transfer to sampling capacitor C_{sh}.

Internal charging of sampling capacitor

It is possible that after the conversion process (successive approximation process in SAR type of ADC) the sample and hold capacitor C_{sh} is charged to some voltage. The reason can be:

- some leakage current to C_{sh} (parasitic current inside ADC structure, see *Figure 39*)
- residual charge transfer from the switches when ADC structure is switched back to default state before next conversion
- other reasons (related to internal ADC parasitic structures)

Figure 39. Parasitic current example inside ADC structure



4 Conclusion

This application notes describes the main ADC errors and then methods and application design rules to minimize STM32 ADC errors and obtain the best ADC accuracy.

The choice of method depends on the application requirements and is always a compromise between speed, precision, enough computation power and design topology. The published methods lead to a precision improvement and are optimized for the design of an ADC converter using the SAR (successive approximation register) principle.

5 Revision history

Table 1. Document revision history

Date	Revision	Changes
14-Nov-2008	1	Initial release.
16-Sep-2013	2	Extended to STM32Fx Series and STM32L1 Series devices. Added *Section 1.1: SAR ADC internal structure*. Added *Section 3.4: High impedance source measurement*. Added *Section 3.3: Software methods to improve precision*. Text improvements and additions. Changed the Disclaimer on the final page.
15-Feb-2017	3	Document scope extended to all STM32 microcontrollers. Updated *Figure 5: Step 2: If MSB = 0, then compare with ¼VREF* and *Figure 6: Step 2: If MSB = 1, then compare with ¾VREF*. Updated *Section 3.3: Software methods to improve precision* introduction. Added STM32L0/L4 ADC hardware oversampling in *Section 3.2.3: Analog-input signal noise elimination*, *Section 3.2.4: Adding white noise or triangular sweep to improve resolution* and *Section 3.3.1: Averaging samples*. Harmonized hexadecimal notation to ‘0x’. Harmonized least significant bit term to ‘LSB’. Updated figures look-and-feel and ground symbol. Color legend added when required.