How to use ADC Oversampling techniques to improve signal-to-noise ratio on STM32 MCUs

Introduction

All STMicroelectronics microcontrollers embed an ADC (analog-to-digital converter) with a given resolution (number of bits) and sampling rate.

For most applications, this resolution is sufficient, but in some cases where a higher accuracy is required, oversampling, and decimating the input signal can be implemented to avoid the use of an external ADC solution and the associated increase in application power consumption.

比较特定硬件单元和软件实现过采样的功耗

This application note presents the oversampling principle, then describes hardware and software oversampling implementation using a specific unit that is available on certain STM32 MCUs. It then compares the two possibilities in terms of power consumption. 两种软件实现提高ADC分辨率的方法，基于使用ADC的虽大采样率对输入信号进行过采样并对采样信号抽取以提高分辨率

For the software implementation, two ADC resolution improvement methods are described. These are based on oversampling the input signal with the maximum sampling rate of the ADC used, and decimating the input signal to enhance its resolution. The embedded software (STSW-STM32014 or X-CUBE-ADC_OVSP) delivered with this application note gives implementation examples for these two methods, and applies them to both medium- and high-density STM32F1 series products, as well as all STM32F3 series and STM32Lx series products. 硬件过采样是使用硬件ADC过采样引擎实现的

For the hardware implementation, an overview of the on-chip hardware analog-to-digital converter (ADC) oversampling engine is provided. It is integrated in the STM32 products listed in Table 1.

The main user benefit of hardware oversampling is increased SNR (signal-to-noise ratio) with less CPU interaction, resulting in overall lower power consumption compared with the software-based implementation. 硬件过采样增加了信噪比，减少CPU交互，与基于软件实现情况相比，功耗更低

Formulas are provided to determine the oversampling ratio or the hardware oversampling unit configuration to use according to the desired resolution improvement. These theoretical formulas are compared to practical use cases.

根据期望的分辨率改进“ 过采样比 ” 或硬件过采样单元配置的公式，将醴陵市与实际示例比较

Table 1. **Applicable products**

Type	Series
Microcontrollers	STM32U5 series, STM32U0 series, STM32H7 series, STM32H5 series, STM32F7 series, STM32F4 series, STM32F2 series, STM32F0 series, STM32L1 series, STM32F3 series, STM32F1 series, STM32L4+ series, STM32L4 series, STM32L0 series, STM32L5 series, STM32G4 series, STM32G0 series, STM32WB series

1 General information

This document applies to STM32 Arm®-based microcontrollers.

Note: Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

2 Oversampling as a way to improve the quality of signal acquisition

过采样是一种提高信号采集质量的方法

ADC将模拟信号转换成一串数字代码。是通过对模拟输入信号之间的误差称为量化误差。而量化分辨率取决于二进制输出字长，通常在6~18位的范围内。输入信号与量化信号之间的误差称为量化误差。

2.1 Quantization of noise and signal-to-noise ratio

量化噪声和信噪比

Analog-to-digital converters (ADCs) transform analog signals into an array of digital codes. It is carried out by performing amplitude quantization of the analog input signal. The quantization resolution depends on the binary output word length, normally in the range of 6 to 18 bits. The error between the input signal and the quantized signal is called the quantization error.

在对信号进行数字化时，理想转换器的最大误差为±½LSB(最低有效位)

The maximum error for an ideal converter when digitizing a signal is ±½ LSB (least significant bit), as shown in the transfer function (left side of Figure 1).

The LSB is also often called a quantum (q). Assuming that the user has an N-bit analog-to-digital converter (ADC) and a voltage reference, VAREF, the quantum, q, is the minimum distance between two adjacent ADC codes. Moreover, it is defined as follows:

LSB通常被称为量子q，假设用户有一个名为的ADC和VAREF，量子q是两个相邻ADC码之间最小的距离

$$q = \frac{V_{AREF}}{2^N} \quad (1)$$

where:

$$q/2a < t < +q/2a$$

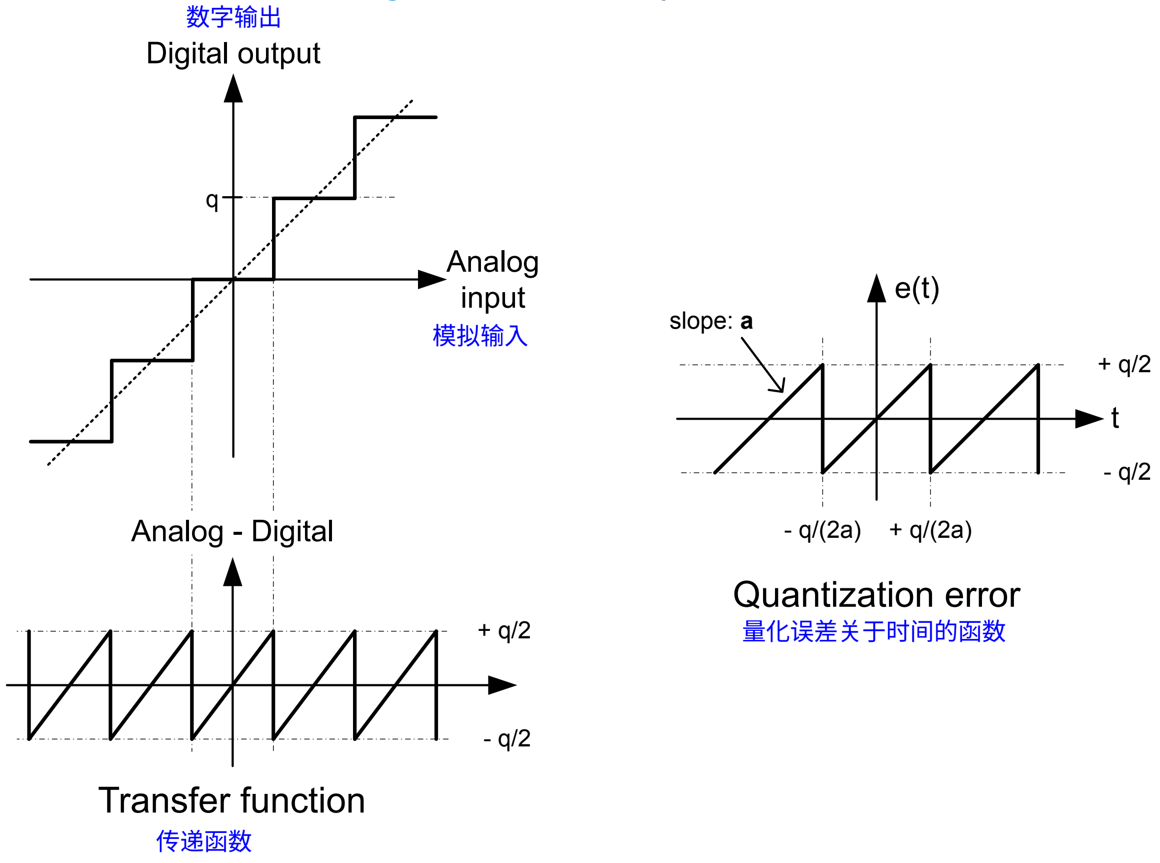
with -q / 2a < t < +q / 2a

The quantization error e(t) as a function of time is shown in the right side of Figure 1.

This is approximated by a sawtooth signal, as it is considered that an uncorrelated sawtooth waveform is a good representation of the quantization error for any AC signal, and that it behaves like wideband noise.

锯齿波信号可以很好的表示量化误差，像宽带噪声

Figure 1. Ideal N-bit ADC quantization



The quantization error e(t) is defined as: 量化误差e(t)的定义式：

$$e(t) = a \times t \quad with \quad -q/2a < t < +q/2a \quad (2)$$

Hence, the RMS value of e(t) is:

量化误差的均方根

SNR(信噪比)是ADC噪声与输入信号功率的比值对于理想的ADC，假设信噪比等于量化噪声与输入信号的比值，不考虑其它的噪声源

$$\sqrt{\bar{e}(t)^2} = \sqrt{\frac{a}{q}\cdot\int_{q/(2a)}^{(-q)/(2a)}\left(e(t)^2\right)dt} = \frac{q}{\sqrt{12}} \tag{3}$$

The SNR (signal-to-noise ratio) is the ratio of the ADC noise to the input signal power. For an ideal ADC, it is assumed that the SNR is equal to the ratio of the quantization noise to the input signal. No other noise source is considered.

对于满量程输入正弦波表示如下：

For a full-scale input sine wave this is expressed as follows:

满量程的均方根值是s(t)=q*2^N/(2*2^0.5)

$$s(t) = q \times 2^{(N-1)} \times \sin(2\pi ft) \tag{4}$$

Using equations (3) and (4), the SNR of an ideal N-bit converter (ADC resolution) is calculated as follows:

利用3式和4式，计算理想N位转换器的信噪比(ADC分辨率)：

$$SNR = 6.02 \times N + 1.76\,dB \tag{5}$$

SNR=20log10(rms value of FS input/rms value of quantization noise)

当有效位数N增大时，信噪比增加6.02dB。对于真正的ADC，必须要考虑不同的误差源：偏移、增益-INL(积分非线性)和DNL(微分非线性)，这些描述在MUC数据手册中有。这些误差降低了ADC的理想分辨率，并决定了ADC实际的有效位数(ENOB)。提高信噪比可以提高ADC的有效位数

It is important to note that the RMS quantization noise is measured over the full Nyquist bandwidth (from DC up to Fs/2). RMS量化噪声是在整个奈奎斯特带宽(从DC到Fs/2)上测量的

It can be seen that when the SNR increases, the ADC effective number of bits (N in the equation 5) increases.

Note also that for a real ADC, different error sources must be considered: offset, gain - INL (integral nonlinear) and DNL (differential nonlinear). A brief description of these errors can be found in the STM32 MCU datasheets. These errors degrade the ideal ADC resolution and determine the real effective number of bits of the ADC (ENOB). Improving the SNR enhances the effective number of bits of the ADC. The following section demonstrates that sampling the input signal rates higher than the Nyquist frequency improves the SNR. The Nyquist frequency is discussed in the next paragraph.

奈奎斯特定理与抗混叠低通滤波器弛豫

2.2 Nyquist theorem and antialiasing low-pass filter relaxation

奈奎斯特定理指出，为了重建模拟输入信号，必须以大于输入信号最大频率分量两倍的速率(采样频率)对信号进行采样

The Nyquist theorem states that to reconstruct the analog input signal, the signal must be sampled at a rate Fs (sampling frequency) that is greater than twice the maximum frequency component of the input signal.

Noncompliance with the Nyquist theorem causes aliasing effects and the analog signal cannot be fully reconstructed from the input samples. 如果不符合奈奎斯特定理，模拟信号就不能从输入样本中恢复

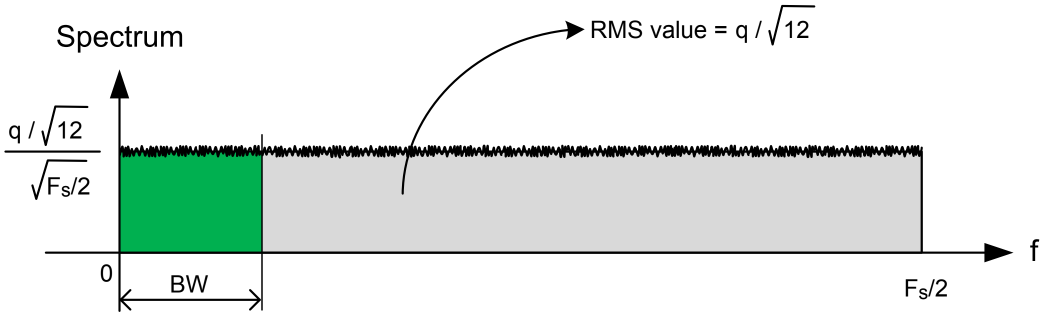
Therefore, for most applications, a low-pass filter is required at the ADC input to filter frequencies lower than half of the sampling frequency. It is difficult to handle the filter constraints with low sampling frequencies. The oversampling consists of sampling the analog input signal at higher rates than the Nyquist frequency limit, filtering the samples, and reducing the sample rate by decimation. Using this method relaxes the antialiasing low-pass filter constraints. 所以，对于大多数应用，ADC输入端需要一个低通滤波器来过滤低于采样频率一半的频率。低采样频率下的滤波器约束很难处理。过采样以高于奈奎斯特频率极限的速率对模拟输入信号进行采样，对样本进行滤波，并通过抽取来降低采样率。使用这样的方法实现抗混叠低通滤波器弛豫。

通过过采样实现的处理增益

2.3 Processing gain achievable with oversampling

In most cases, we can consider that the quantization noise is uncorrelated with respect to the input signal. In this condition, the quantization noise is approximately Gaussian and spreads more or less uniformly over the Nyquist bandwidth (see Figure 2). 在大多数情况下，认为量化噪声相对于输入信号是不相关的。在这种情况下近似为高斯噪声，并且在奈奎斯特带宽上或多或少的均匀扩散

Figure 2. Quantization noise spectrum



然而，在采样时钟和信号谐波相关的特定条件下，量化噪声会产生相关。它的能量主要集中在信号的谐波中。在量化噪声不表现为随机噪声的情况下，必须要应用Dithering

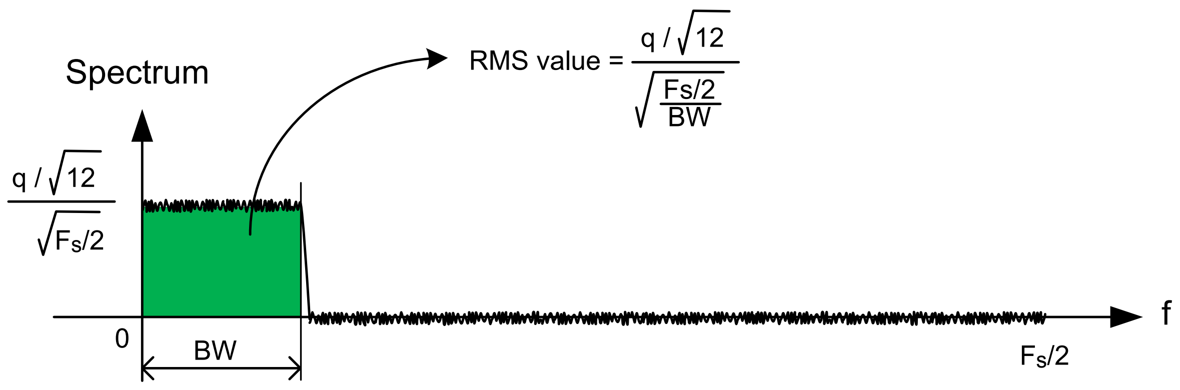
However, under certain conditions where the sampling clock and the signal are harmonically correlated, the quantization noise becomes correlated. In addition, its energy is concentrated in the harmonics of the signal. In conditions where the quantization noise does not appear as random noise, dithering must be applied (see Section 2.4 Dithering).

In many applications, the useful signal occupies a bandwidth (BW) smaller than Fs/2.

在许多情况下，目标信号占用的带宽小于Fs/2

If digital filters are used to remove the noise outside the BW (this filter can be more precise than the antialiasing ones mentioned before), the total RMS noise is reduced (Figure 3); the RMS value of the quantization noise is divided by a ratio that depends on the useful bandwidth (BW) with respect to the sampling rate (Fs).

Figure 3. Quantization noise gain


Spectrum
RMS value = $\frac{q/\sqrt{12}}{\sqrt{\frac{Fs/2}{BW}}}$
$\frac{q/\sqrt{12}}{\sqrt{F_S/2}}$
0
BW
f
$F_S/2$

通过滤除带宽外的噪声，将处理增益考虑在内，重新表述信噪比表达式

We can then reformulate the previous SNR expression taking into account this processing gain, by filtering the out-off band noise:

$$SNR = 6.02 \times N \times\ + 1.76\,dB + 10xLog_{10}OSR \quad (6)$$

This expression is valid over a bandwidth, BW, with an oversampling ratio given by:

该表达式对带宽有效，过采样率为：

$$OSR = FS/(2 \times BW) \quad (7)$$

2.4 Dithering

The technique presented above works well for a white quantization noise. 上述方法对于白噪声效果比较好

However, if the sampling clock and the signal are harmonically correlated (in this case the quantization noise becomes correlated as well), or when the input signal amplitude is smaller than q/2, the processing gain does not work properly. 但是如果采样时钟和信号是谐波相关的(在该情况下，量化噪声也变得相关)，或者当输入信号幅度小于q/2时，处理增益就不能正常生效了

这是因为第一种情况下量化噪声不再是随机的，第二种情况下，当信号小于量化步长时，理论上没有代码转换

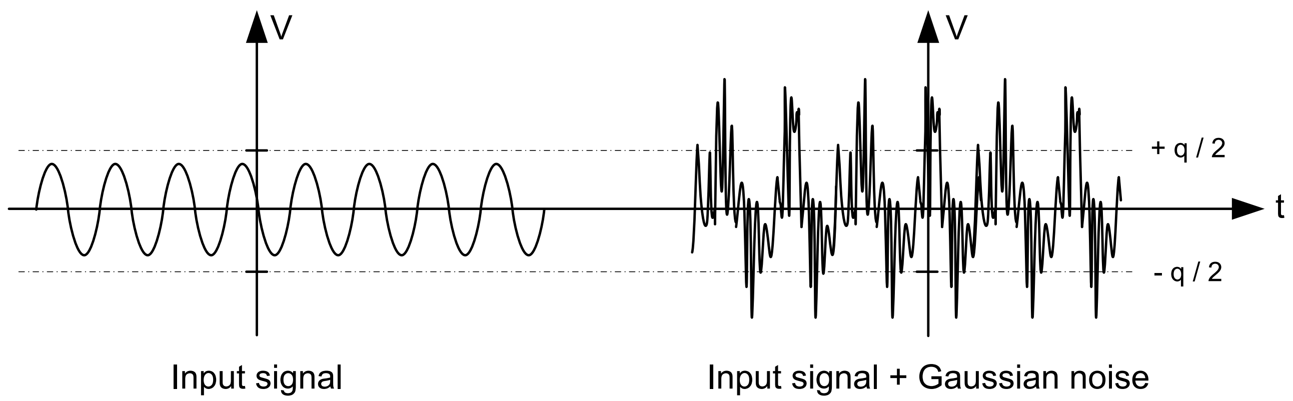
This is because for the first case the quantization noise is no longer random, and for the second case there are (in theory) no code transitions when the signal is smaller than the quantization step.

One way to solve these issues is to use the dithering technique, where a small Gaussian noise is added to the input signal (see the left side of Figure 4), to obtain a signal (see the right part of the figure) that can ensure LSB toggling.

Dithering also ensures that the quantization noise is always random, independently from the input signal.

使用抖动技术解决这些问题，在输入信号加一个小的高斯噪声，获得一个可以确保LSB切换的噪声，噪声还确保量化噪声是始终随机的，独立于输入信号

Figure 4. Dithering technique



输入信号　　　　输入信号+高斯噪声

如果对噪声进行塑造，信噪比的影响就可以大大降低，例如：使得抖动噪声在带宽内过滤，仅在带宽外存在

The impact of SNR can be much reduced if the noise is shaped; for example if the dithering noise is filtered in the wanted bandwidth, and is only present outside of that bandwidth.

可以使用DAC产生抖动信号，或者通过配置PWM模式下的定时器和其他电子元件产生抖动信号

The embedded DAC can be used for generating the dithering signal. Also, in Section 3.2 **Oversampling using triangular dither**, we generate the dithering signal by means of a timer configured in PWM mode, and some additional electronic components.

If the application does not require the capture of signals smaller than the quantization step, and if the quantization error can be considered as wideband noise, the dithering technique can be omitted.

如果应用程序不需要捕获小于量化补偿的信号，并且量化误差可以视为带宽噪声，则可以省略抖动技术。

3 Software oversampling

两种软件过采样实现方法。每种都有优缺点，加以对比。

This section presents two software-oversampling implementation methods. Each has advantages and disadvantages, which are compared.

The embedded software delivered with this application note is available in the STSW-STM32014 (or X-CUBE-ADC_OVSP) package.

3.1 Oversampling using white noise 使用白噪声实现过采样

3.1.1 Oversampled signal SNR with white input noise 过采样信号信噪比与输入白噪声

Equation 6 in Section 2.3) gives the SNR obtained when oversampling the input signal with a sample rate OSR times faster than the Nyquist frequency, and low-pass filtering the signal band:

2.3节中，输入信号进行采样率OSR大于奈奎斯特频率1倍的过采样，并对信号频率进行低通滤波后得到的信噪比：

$$SNR = 6.02 \times N \times + 1.76\,dB + 10xLog_{10}OSR$$

This shows that each doubling of the sampling frequency reduces the in-band noise by 3 dB, and increases the measurement resolution by 0.5 bit. Therefore, a 6 dB SNR gain is required to add 1 resolution bit to the ADC. In general, if *p* additional bits are required by the application, the ADC sampling frequency should be at least:

$$F_{OVS} = 4^p \times F_s \quad (8)$$

其中Fs为ADC采样频率

Where F_s is the ADC sampling frequency used.

结果表明，采样频率每增加一倍，带内噪声降低3dB，测量分辨率提高0.5bit，因此需要6dB信噪比增益来为ADC增加1个分辨率有效位，一般来说，如果应用程序需要额外的p位，则ADC采样频率至少为：

3.1.2 Decimation 抽取

平均意味着添加m每个样本并将结果除以m。平均来自ADC测量的多个数据相当于一个低通滤波器，他可以衰减信号波动和噪声，因此，平均通常用于平滑和去除输入信号中的尖峰

Averaging means adding m samples and dividing the result by m. Averaging several data from an ADC measurement is equivalent to a low-pass filter, which attenuates the signal fluctuation and noise. Averaging is therefore often used to smooth and remove spikes from the input signal.

Note: *Normal averaging does not increase the resolution of the conversion because the sum of m N-bit samples divided by m is an N-bit representation of the sample.*

平均不会增加转换的分辨率，因为m个n位的样本除以m还是n位表示，抽取是一种平均方法，当与过采样结合时，抽取提高了ADC的分辨率

Decimation is an averaging method. When combined with oversampling, decimation improves the ADC resolution.

事实上，加上4^pADC N位采样，可以得到N+2p位信号表示，为了有额外的有效位，总和向右移动P位。

In fact, adding 4^p (4 power of p) ADC N-bit samples, gives a representation of the signal on N+2p bits. To have p additional effective bits, the sum is shifted to the right by p bits.

This FIR filter with equal filter coefficients enables the user to filter the oversampling frequency by giving an output sample computed from the OSR input samples. 该FIR滤波器具有相等的滤波系数，用户能够通过给出从OSR输入样本计算的输出样本来过滤采样频率

The oversampling method limits the maximum input frequency bandwidth. In the case of the STM32F1 series, STM32F3 series and STM32Lx series (with maximum sampling rate around 1 Msps), signals having components up to 500 kHz can be processed by the ADC. If for example, two additional resolution bits are required, the maximum input frequency is 500 kHz/16 = 31.25 kHz when the oversampling uses white noise.

过采样的方法限制了最大输入频率带宽。比如最大采样率为1Msps情况下，ADC可以处理分量高达500KHz的信号。如果需要两个额外的分辨率位，当过采样使用白噪声时，最大输入频率为500KHz/16=31.25KHz

3.1.3 When is this method efficient? 这种方法什么时候有效？

为了使用过采样和抽取方法正常工作，必须满足以下要求：

For the oversampling and decimating method to work properly, the following requirements must be satisfied:

输入信号中需要有噪声，这种噪声要近似于白噪声，在要作用的频带上具有均匀地功率谱密度。

- There should be some noise in the input signal. This noise must approximate the white noise with a uniform power spectral density over the frequency band of interest.
- The noise amplitude must be sufficient to toggle the input signal randomly from sample to sample by an amount of at least 1 LSB. Otherwise, the input samples would have the same representation, and the sum and average operations would not give any extra resolution. For most applications, the internal ADC thermal noise and the input signal noise are sufficient to use this method. If the thermal noise does not have a high-enough amplitude to toggle the input signal randomly, then a dithering operation must be applied (see part 2.4). Regarding this point, two questions can be raised. The first is *How to evaluate the ADC noise and test its Gaussian criteria?* and *How to generate white noise if needed?*.

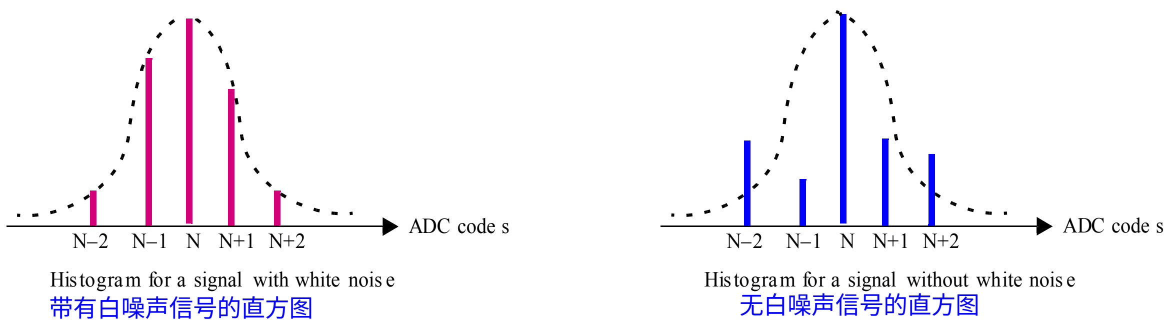
A practical way of detecting the Gaussian criteria of the input signal noise is to see the distribution of a clean DC signal over the ADC codes. The histogram method can be used to verify if the input noise follows a Gaussian distribution. The example in Figure 5 shows two possible situations.

噪声幅度必须足以使输入信号以至少1LSB的量从一个样本切换到另一个样本，否则，输入样本将具有相同的表示，并且求和平均操作不会提供任何额外的分辨率。

对于大多数应用内部ADC热噪声和输入信号噪声足以使用这种方法。如果热噪声没有足够高的幅度来随机切换输入信号，则必须应用抖动操作。关于这一点会有两个问题。如何评估ADC噪声并测试其高斯准则？如果需要，如何产生白噪声？

检测输入信号噪声的高斯准则的一种方法是看到一个干净的直流信号的ADC编码。直方图法可以用来验证输入噪声是否服从高斯分布。如图5。

Figure 5. **Histogram analysis**



In the case where an external noise dither must be added to the input signal, the thermal noise generated by a diode or a resistor can be injected into the input signal. 在输入信号必须加入外部噪声抖动的情况下，可以将二极管或电阻产生的热噪声注入到输入信号中

The input noise must not correlate with the useful input signal, and the input signal should have an equal probability of being between two adjacent ADC codes. This means that this method does not work for systems using a feedback process. 输入得噪声必须与有用的输入信号无关，并且输入信号在两个相邻ADC编码之间的概率相等，意味着该方法不适用于使用了反馈的系统。

实现方法

3.1.4 Implementation method on STM32F1, STM32F3, and STM32Lx series devices

This method describes the different steps undertaken to implement and test the oversampling method on the STM32F1 series, STM32F3 series and STM32Lx series devices. 要使该方案正常工作，必须有白噪声使输入信号随机切换1/2LSB，所以必须要考虑应用程序的环境噪声。

According to the previous section, to make this solution work properly, there must be some white noise to make the input signal toggle randomly by 1/2 LSB. For this, the application environment noise must be considered.

第一步包括计算ADC热噪声，以确定是否必须将外部白噪声注入输入信号。在典型应用板上，计算的噪声不仅包括ADC内部噪声，还包括不同板组件和布局可能产生的噪声。

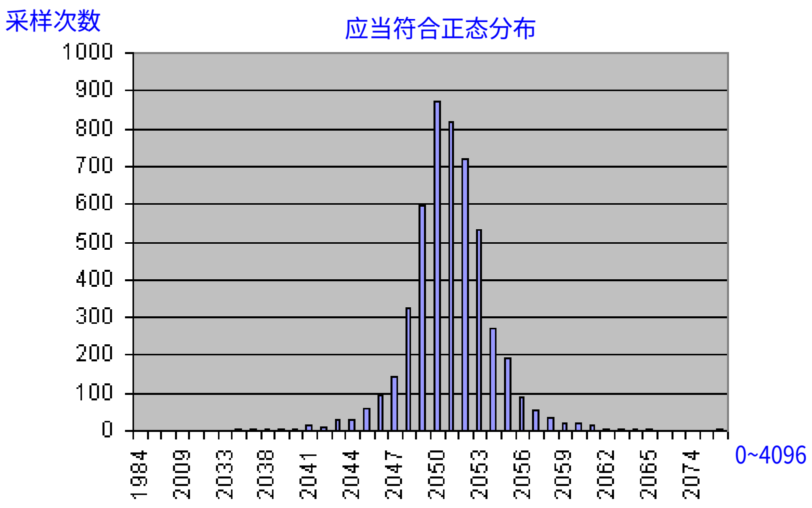
The first step consists in computing the ADC thermal noise to conclude if external white noise must be injected into the input signal. In a typical application board, the computed noise does not include only the ADC internal noise but also the possible noise generated by the different board components and the layout. Therefore, this evaluation depends on the application board but the methodology remains the same.

直方图用于不同的直流输入电压。该输入电压被采样5000次

The histogram method is used for different DC input voltages. This input voltage is sampled a large number of times (example 5000). The related distribution can be easily interpreted using a spreadsheet.

For example, for a 1.65 V dc input voltage applied on the STM3210B-EVAL evaluation board, the histogram shown in Figure 6 is detected.

Figure 6. Histogram analysis for DC = 1.65 V



ADC的热噪声可以从这个直方图中计算出来，使用该ADC测试噪声，需要完成以下操作：

The ADC thermal noise can be computed from this histogram (although this can be shown, it is not the objective of this application note and the details are not offered here).

To carry on this ADC noise test, the user must do the following:

- Uncomment the line `#define_Themal_Noise_Measure` in the *oversampling.h* file.

取消oversampling.h文件中的#define Themal_Noise_Mrasure行注释

- Configure the `Total_Samples_Number` which is the number of ADC conversion operations. It must be smaller than 65535. The DMA channel is configured to store the number of ADC samples in a RAM buffer. At the end of the transfer, an interrupt is generated and the number of occurrences of each ADC code is computed.

配置Total_Samples_Number，这是ADC转换操作的数量。必须小于65535.MDA通道被配置在RAM缓冲区中存储ADC样本的数量。在传输结束时，产生中断，并计算每个ADC编码出现的次数

- To compute the occurrence of the ADC codes, a variable giving the relevant ADC codes is defined.

为了计算ADC编码的出现次数，定义了一个给出相关ADC编码的变量

When the code is run, `Relevant_ADC_Samples` ADC samples and their corresponding number of occurrences are displayed on the HyperTerminal. The HyperTerminal configuration is 8-bit data, no parity, 115 200 baud rate. If the effective number of ADC samples found is smaller than the defined `Relevant_ADC_Samples` variable, then 0 is displayed for both ADC code and ADC code occurrences. The user can capture them and build a histogram.

当代码运行时，Relevant_ADC_SamplesADC样本及其相应的出现次数显示在串口助手，在串口助手上配置为8为数据，无奇偶校验，波特率115200，如果发现的ADC样本有效数量小于定义的Relevant ADC_Samples变量，则ADC编码和ADC编码出现的次数显示都为0，可以捕获它们并构建一个直方图

3.1.4.1 *Embedded-software flowchart for oversampling using white noise*

使用白噪声实现过采样的软件流程图

The STM32F1 series, STM32F3 series and STM32Lx series on-chip ADC conversion frequency is fixed to 1 MHz. The ADC DMA channel is configured to transfer the number of oversampled inputs from the ADC data register to a buffer in RAM. This transfer is configured to occur one time. At the end of the DMA transfer, an interrupt is triggered and the oversampled result is computed.

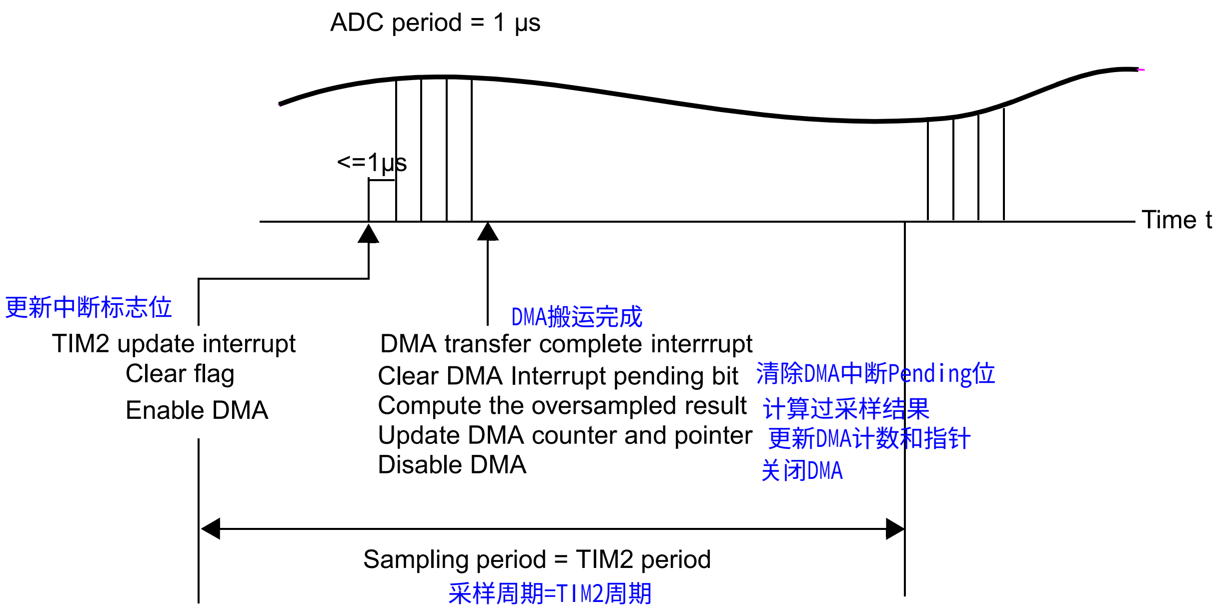
ADC DMA通道配置为将ADC数据寄存器的过采样输入数据传输到RAM的缓冲区，此传输配置只发生一次，在DMA传输结束时，触发中断并计算采样结果

The general-purpose timer TIM2 is used to generate the input signal sampling frequency. For this, the TIM2 reference clock is configured at 1 µs. Its period determines the input signal sampling period. It is defined in the oversampling.h file as `#define Input_Signal_Sampling_Period`. When the TIM2 update interrupt is triggered, the DMA is reenabled and the converted ADC values can be treated.

Figure 7 summarizes the implemented functionality.

通用定时器TIM2用于产生输入信号采样频率。为此，将TIM2参考时钟配置为1us。其周期决定了输入信号的采样周期，他在oversampling.h文件中定义为#define Input_Signal_Sampling_Period。当TIM2更新中断被触发时，DMA被重新启用，转换后的ADC值可以被处理

Figure 7. **Oversampling using a white noise flowchart**



过采样的数据在DMA传输完成中断中计算。处于同步的原因，建议在第二个终端中读取结果。在这种实现中，TIM2周期必须大于ADC转换OSR采样所需的时间，而且大于ADC中断执行时间

The oversampled datum is computed in the DMA transfer complete interrupt. For synchronization reasons, it is recommended to read it in the second TIM2 interrupt. Note that with this implementation, the TIM2 period must be greater than the time required by the ADC to convert OSR samples, and greater than the ADC interrupt execution time.

If the sampling frequency required by the application is exactly OSR µs, then the user is not required to use the timer TIM2 to generate the input sampling frequency. However, the DMA must be configured to be functional in continuous mode and the DMA transfer complete interrupt must be updated accordingly. The oversampled datum is usually computed in the DMA transfer complete interrupt.

如果应用程序所需的采样频率正好是OSR us，那么用户不需要使用TIM2生成输入采样频率。但是DMA必须配置在连续模式下工作，并且DMA传输完成中断必须相地更新。过采样数据通常在DMA传输完全中断中计算。

3.1.4.2 *Oversampling using white noise - result evaluation* 过采样使用白噪声--结果评估

To evaluate the oversampling method, the user must uncomment the `#define Oversampling_Test` line and configure the number of samples with an enhanced resolution. 要评估过采样方法，必须取消#define overampling_test行注释，并以增强的分辨率配置样本数量

When this line is uncommented, a buffer is created in the RAM to store the oversampled data. The buffer contents are then displayed on the HyperTerminal. The HyperTerminal configuration must be 8-bit data, no parity, and 115 200 baud rate. The user can capture them into a *.txt* file and then compare the expected results to the real ones.

当这一行没有注释的时候在RAM中创建一个缓冲区来存储过采样的数据，然后在串口助手上显示缓冲区的内容。串口助手的配置必须是8位数据，无奇偶校验，波特率为115200可以将他们捕获到.txt文件中，然后将与其结果与实际结果进行比较、

To evaluate the new enhanced ADC, a ramp with a 50 Hz frequency and a 1 V amplitude is input to the ADC and sampled using the oversampling algorithm every 100 µs. 为了评估新的增强ADC，将频率为50Hz、幅值为1V的斜坡输入到ADC，并使用过采样算法每100us采样一次

The embedded software example related to this method is located in the *WhiteNoiseMethod* folder.

The oversampling algorithm using white noise is run with the same ramp (50 Hz frequency and 1 V amplitude). Both Figure 8 and Figure 9 give the ADC oversampled data as a function of time in µs. Figure 8 is the result of adding one bit while Figure 9 is the result of adding two additional bits to the ADC on-chip resolution.

使用白噪声的过采样算法以相同的斜坡(50Hz/1V)运行，图8和图9给出了ADC过采样数据作为时间函数。图8是增加1为的结果，图9是增加两位的结果

When the ramp is sampled without using any extra software resolution, with a 3.3 V reference supply, 1 V corresponds to the digital value 1250. 当斜坡在不使用任何额外软件分辨率的情况下采样时，使用3.3V参考电源，1V对应数值1250

When one additional bit is added, 1 V is sampled as 2500 and when two additional bits are added, 1 V is sampled as 5000. 当额外增加1bit时，1V被采样为2500，当增加两个额外的比特时，1V被采样为5000

This means that the environment contains enough noise for this method to work.
这就意味着环境中包含着足够的噪声，使得该方法能够工作

Figure 8. **Ramp samples with 1 additional bit**

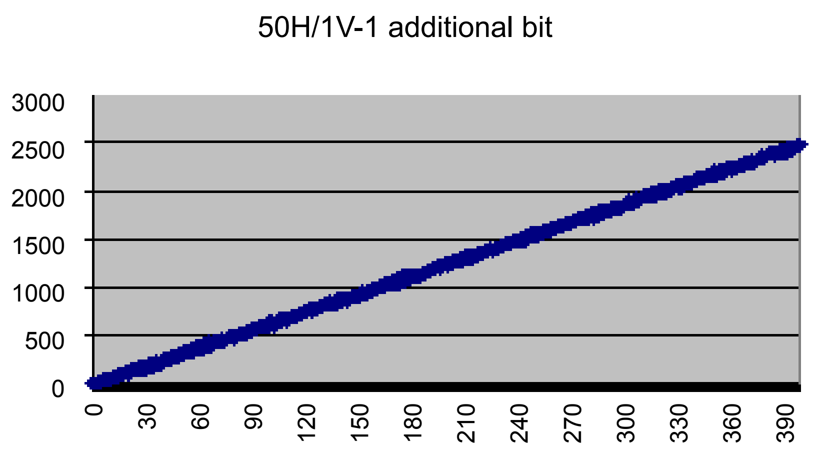
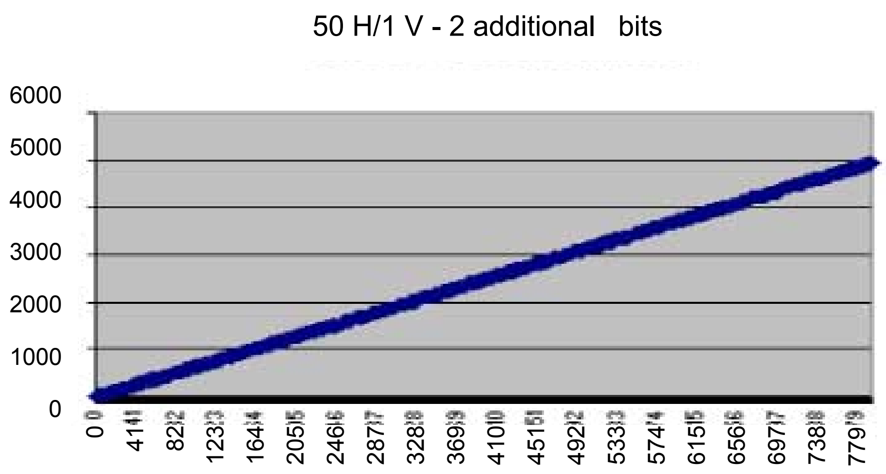


Figure 9. **Ramp samples with 2 additional bits**



3.2 Oversampling using triangular dither

三角抖动过采样

Assuming that the input signal is between two successive quantization steps q0 and q1 during the oversampling period, then the converter may convert it either to q0 or q1. Adding extra p bits of resolution means determining the relative position of the input signal between q0 and q1.

通过适当的三角信号的加入，量化起产生一系列的q1s和q0s,在给定的间隔内对q1的出现次数进行平均，可以确定输入信号在较低和较高量化步骤之间的相对位置

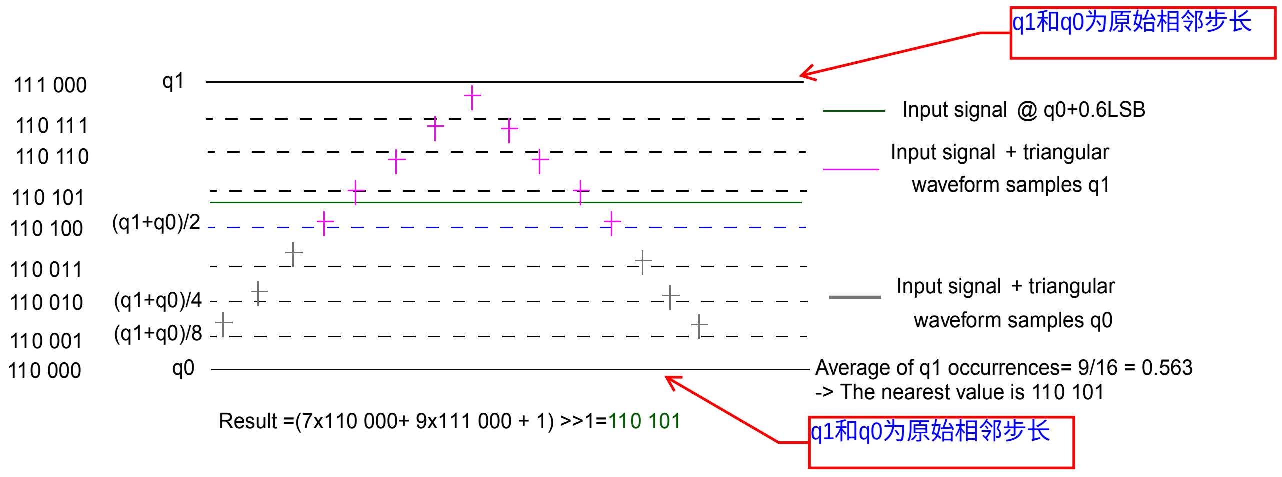
With the addition of an appropriate triangular signal, the quantizer generates a series of q1s and q0s. Averaging the q1 occurrences over a given interval determines the relative position of the input signal between the lower and the higher quantization steps. 加入的三角波在周期上应该等于ADC采样时间再乘以过采样倍数，三角波在幅度上为n+.5LSB

The theory states that the best results are achieved when dithering the input signal using a triangular waveform with a period of OSR times the ADC sampling period and an amplitude of n + 0.5 LSB where n = 0,1,2,3.

The theory behind this method is quite complicated, so this Figure 10 is an example to illustrate how this method works. In this example, the ADC on-chip resolution is 3 and 3 extra bits are added by embedded software. The input signal is assumed to have an amplitude of q0+ 0.6LSB (q0 = 6 in this example). To add three additional bits, the input signal is sampled 2.23 times (16 times).

理论很复杂，下图是工作示例，例子中片上的ADC分辨率为3bit，并通过软件增加额外的3bit，输入信号的幅度为q0+0.6LSB,其中q0=6，为了增加额外3bit，输入信号被采样2.23次

Figure 10. How to perform oversampling by adding a triangular signal


q1和q0为原始相邻步长
111 000
q1
110 111
110 110
110 101
110 100
(q1+q0)/2
110 011
110 010
(q1+q0)/4
110 001
(q1+q0)/8
110 000
q0
Input signal @ q0+0.6LSB
Input signal + triangular waveform samples q1
Input signal + triangular waveform samples q0
Average of q1 occurrences= 9/16 = 0.563
-> The nearest value is 110 101
Result =(7x110 000+ 9x111 000 + 1) >>1=110 101
q1和q0为原始相邻步长

If the input signal is not correlated with the triangular waveform, then it is demonstrated that the gain in the SNR is equal to: 如果输入信号与三角波不相关，则可以证明信噪比中的增益关系式如下：

$$SNR_{Gain} = 20\log\left(\frac{OSR}{2}\right) \tag{9}$$

Therefore, each doubling of the sampling frequency improves the SNR by 6 dB and adds 1 bit of ADC resolution.

In general, to add p-bit extra resolution, the oversampling frequency must be equal to:

每增加1倍采样率，就可以增加1位有效ADC位数，实际有效位数会比这个少，如果需要增加p位ADC分辨率，施加三角波扰动过采样的频率应当满足：

$$F_{OVS} = 2 \cdot 2^{p} F_s \tag{10}$$

3.2.1 When does this method work?

In order to make this method work, the input signal must not vary by more than ± 0.5 LSB during the oversampling period and must not correlate with the triangular dither signal.

3.2.2 Implementation method on STM32F1, STM32F3, and STM32Lx Series devices

In order to implement the second solution, the following is needed:

- An operational amplifier to perform the sum of the input signal and the triangular waveform. For this, an op-amp inverter/summing stage is required. An STMicroelectronics LMV321 can be used.
- A triangular waveform with a period of OSR times the ADC conversion rate. The user can either use a signal generator or one of the on-chip timers and an RC network to generate this triangular signal. Indeed, the on-chip timer generates a PWM signal with a duty cycle varying from 0 to 100%. This PWM output can be filtered with an RC filter to generate a triangular signal varying from 0 to VDD. In order to generate an amplitude of 0.5 LSB, then the output is first passed through a capacitor (to cut the DC component) and then divided by the prescaler R2/R3 (see Figure 11. Hardware requirements of oversampling by adding a triangular signal). This prescaler is equal to the ADC number of words.

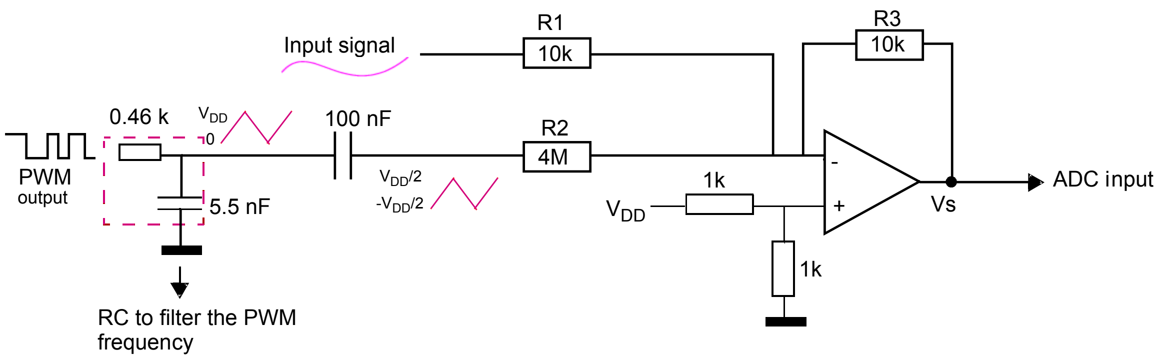
三角波周期和采样周期保持有效倍数关系，三角波可以通过定时器或者RC电路来实现。实际上是定时器产生一个占空从0~100%变化的PWM信号，通过RC滤波产生一个从0到VDD的三角波。为了产生0.5LSB的幅值，输出首先通过电容（以切断直流分量），然后除以预分频器R2/R3，硬件要求过采样

输入信号不能在运放之后改变，因此R1等于R3

- The input signal must not be changed after the op-amp. For this reason, R1 should be equal to R3.
- The sum of the input signal and the triangular dither is inverted. For this purpose, a 3.3 V offset is required on the positive entry of the op-amp. After the oversampled data are computed, this offset is subtracted to give the input signal estimation with an extra resolution.

输入信号和三角抖动的和是反转的，为此，在运放的正极输入上需要3.3V的便宜，在计算过采样数据后，减去该偏移量已获得额外分辨率的输入信号

Figure 11. **Hardware requirements of oversampling by adding a triangular signal**



3.2.2.1 *Embedded-software flowchart for oversampling using triangular dither*

The STM32F1 series, STM32F3 series and STM32Lx series on-chip ADC conversion frequency is fixed at 1 MHz. The ADC DMA channel is configured to transfer the number of oversampled inputs from the ADC data register to a buffer in RAM. This transfer is configured to occur one time. At the end of the DMA transfer, an interrupt is triggered and the oversampled result is computed.

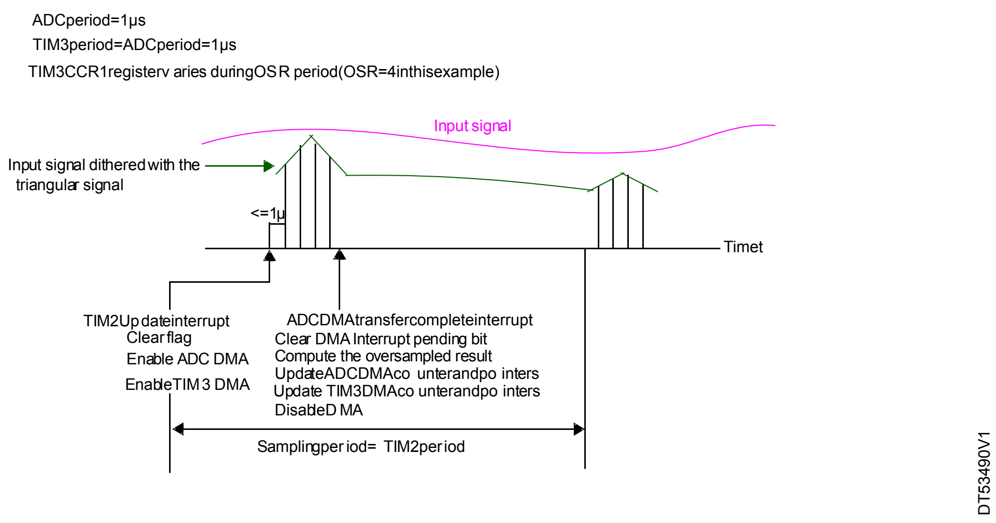
The general-purpose timer TIM2 is used to generate the input signal sampling frequency. For this, the TIM2 reference clock base is configured at 1 µs. Its period determines the input signal sampling period. It is defined in the oversampling.h file by `#define Input_Signal_Sampling_Period`.

The triangular dither is generated using the timer TIM3 configured in PWM mode by updating the Capture Compare Register CCR1. The timer TIM3 period must be equal to the ADC conversion rate and CCR1 must be updated OSR times where OSR is the oversampling factor. To do this, the possible CCR1 values are first computed and stored into a RAM buffer, then the DMA transfer is used to update the CCR1 register, removing the need for interrupts.

Note that the ADC conversion rate limits the oversampling factor. For example, in the case where the ADC is running at 1 MHz, the STM32F1 series is operating at 56 MHz. To have a period of 1 µs, the autoreload register of the timer TIM3 must be equal to 55. The maximum number of additional bits is then 4.

When a TIM2 update interrupt is triggered, the ADC and TIM3 DMA are reenabled and the converted ADC values can be treated to compute the new sample with the extra resolution bits. Figure 12 summarizes the implementation.

Figure 12. Oversampling using triangular dither flowchart


ADCperiod=1µs
TIM3period=ADCperiod=1µs
TIM3CCR1registerv aries duringOSR period(OSR=4inthisexample)
Input signal
Input signal dithered with the triangular signal
<=1µ
Timet
TIM2Updateinterrupt
Clear flag
Enable ADC DMA
EnableTIM3 DMA
ADCDMAtransfercompleteinterrupt
Clear DMA Interrupt pending bit
Compute the oversampled result
UpdateADCDMAco unterandpo inters
Update TIM3DMAco unterandpo inters
DisableD MA
Samplingperiod= TIM2period
DT53490V1

在过采样期间输入信号的变化不得超过±0.5LSB

For this method to work, the input signal must not vary by more than ±0.5LSB during the oversampling period. This means that for STM32F1 series, STM32F3 series or STM32Lx series devices operating from a 3.3 V VREF+, the maximum allowed variations of the input signal during the oversampling period is ~0.4 mV.

3.3V的ADC输入信号最大允许
变化为~0.4mv

On the other side, a triangular waveform with an amplitude of 0.5 LSB means a 0.4 mV amplitude when operating the STM32F1 series, STM32F3 series or STM32Lx series from a 3.3 V VREF+. The application environment must therefore not be very noisy. Any disturbance of the triangular waveform has an impact on the computed oversampled data.

振幅为0.5LSB意味着振幅为0.4mv，要求程序环境不能非常嘈杂，三角波行的任何扰动都会对计算出的过采样数据产生影响

According to the implementation, the triangular waveform is generated by means of the STM32 timer and an RC filter that cuts the 1 MHz timer frequency. The timer PWM output signal is integrated to provide a triangular signal with a 3.3 V amplitude. The division is done with the ratio R3/R2.

The embedded software related to this method is located in the *TriangularDitherMethod* directory.

示例中的三角波是由定时器和RC滤波产生的，滤波器消减了1MHz定时器频率，timer的PWM信号提供一个3.3V的三角波，R3/R2进行缩放

3.3 Comparison of software oversampling methods 软件过采样方法的比较

The first method based on oversampling and averaging using white noise provides a half-bit additional resolution for each doubling of the oversampling rate. The maximum input frequency is drastically decreased with the additional number of additional bits.

基于过采样和使用白噪声的平均，过采样率每增加1倍，就能提供0.5bit的额外分辨率，最大输入频率随着额外的bit增加而急剧下降

For applications where this gain is sufficient, it is a good choice. It requires the presence of white noise in the input signal to make the signal toggle between two adjacent ADC codes. In general, the ADC thermal noise is sufficient and there is no need to add external hardware to act as an external white noise source. This makes the solution more cost effective.

这个对增益足够的应用程序是比较友好的，它要求输入信号存在白噪声，使信号在两个相邻的ADC代码之间切换。一般ADC的热噪声是足够的，不需要额外增加外部白噪声。

The second method based on dithering the input signal using a triangular waveform and computing its relative position between two quantized steps provides one more bit for each doubling of the oversampling rate. This is twice the improvement given by the first method. To make this method work, the input signal must not correlate with the triangular signal and must not have a variation greater than 0.5 LSB during the oversampling period. However, external hardware is needed to add the input signal and the triangular waveform.

使用三角波对输入信号进行抖动，并计算其在量化步骤之间的相对位数，每增加1倍过采样率就增加1bit，这种改进是第一种方法的两倍，为了是这种方法有效，输入信号必须与三角信号不相关，并且在过采样器件不得大于0.5LSB，需要外部硬件来添加输入信号和三角波

Table 2 summarizes the main differences between the two methods. It is not possible to say that one method is better than the other. Each method has its advantages and limitations. The user must select the one that better meets their application requirements (sampling frequency, number of effective bits, and so on).

Table 2. Oversampling using white noise versus oversampling using triangular dither

Implementation conditions	Oversampling using white noise	Oversampling using triangular dither
Oversampling factor to add p bits to the ADC on-chip resolution	4^p	2.2^p
Maximum input signal frequency	f_{ADC} max /(2.4^p)	f_{ADCmax}/(2.2.(2^p))
Dither signal	White noise with an amplitude of at least 1 LSB	Triangular signal with an amplitude of n+0.5LSB

过采样因子增加精度p位提升的ADC精度

最大信号输入频率

白噪声至少为1LSB

三角波幅值为n+.5LSB，即最少0.5LSB

Implementation conditions	Oversampling using white noise	Oversampling using triangular dither
External hardware	External white noise source needed if the input signal noise is not sufficient.	Triangular waveform generator: an on-chip timer can be used. In this case, an RC network is used to filter the PWM frequency. An op-amp is needed to add the triangular waveform and the input signal.

如果噪声源不够则需要外部白噪声

使用片上定时器产生三角波，要用RC网络过滤PWM频率

需要运放将三角波和输入信号相加

3.4 Hints for software oversampling

3.4.1 What is the maximum number of bits that can be added to the on-chip ADC resolution?

It can be easily shown that increasing the on-chip ADC resolution decreases the maximum frequency component of the input signal.

1MHz的ADC过采样2位时，输入信号允许的最大输入频率为：

For example, when using the STM32F1 series, STM32F3 series or STM32Lx series ADC at 1 MHz and two additional bits are required by the application, then the maximum input frequency is divided by:

- 16 when using the white noise method (62.5 kHz) 1MHz/4^2
- 4 when using the triangular dither method (125 kHz). 1MHz/2^2

输入信号数值是在过采样期间完成的

For the two methods, the estimation of the input signal is done during an oversampling period of OSR times the ADC conversion rate. In the case the ADC is running at 1 MHz, the input signal estimation is done over OSR µs. The signal must not vary by more than 1/2LSB for the white noise method and, by ±0.5LSB for the triangular waveform method.

- When using the white noise method, the maximum number of bits that can be added to the ADC resolution depends only on the input signal. 使用白噪声时，ADC分辨率的最大位数取决于输入信号
- When using the triangular dither method, the maximum number of bits that can be added to the ADC resolution does not depend only on the input signal. In fact, the steps defining the triangular signal depend on the ADC and APB frequencies. The timer period should be equal to the ADC rate:
  - $2x(2^P) \le$ timer period
  - $P \le \log2$ (timer period / 2)

使用三角抖动方法时，可以添加到ADC分辨率的最大位数不仅取决于输入信号还与ADC和APB频率，定时器周期等于ADC频率

In our example, running the ADC with a rate of 1 µs causes the STM32F1 series to operate at 56 MHz, which means that the timer period must be equal to 55. The maximum number of bits that can be added in this case is 4.

3.4.2 Taking advantage of the STM32 DAC implementation

Some STM32F1 series, STM32F3 series and STM32Lx series devices come with a DAC (digital-to-analog converter) that can be used in the oversampling method to avoid the use of external components.

The DAC can be used in the two oversampling methods as follows:

白噪声不足DAC用于幅值可编程的白噪声，由伪随机算法生成

In the first method, the DAC can be used to generate a white-noise waveform with programmable amplitude that can be injected into the input signal if noise is not sufficient. The waveform is generated thanks to the implemented pseudorandom algorithm. For more details, refer to the STM32F1 series, STM32F3 series and STM32Lx series reference manuals.

DAC用来产生三角波，就不需要RC电路来过滤定时器的PWM频率

In the second method, the DAC can be used to generate the triangular waveform. This removes the need for any additional external RC circuitry to filter the timer PWM frequency.

Note: *This is not implemented in the software described in this application note.*

3.4.3 Taking advantage of the STM32F1 series, STM32F3 series and STM32L4 series dual ADC mode implementation

In some STM32F1 series, STM32F3 series and STM32L4 series devices, the dual ADC mode is an interesting feature that allows two ADCs to convert at the same time. Using the dual ADC fast interleave mode, the same channel is converted alternately by ADC2 and ADC1. The time separating two successive samples is 7 ADC clock cycles. The input signal is therefore oversampled faster. In the example described in this application note, a sample is obtained every 1 µs. Using the dual ADC fast interleave mode, it is possible to have a sample every 7 ADC clock cycles that is every 0.5 µs when running the ADC at 14 MHz.

Note: *This hint is not implemented in the software given within the application note.*

Dual ADC模式允许两个ADC同时转换，可以使得过采样速度快1倍

3.4.4 Taking advantage of the hardware ADC oversampling implementation

On some STM32 devices, the ADC implements the oversampling feature in hardware. This feature is presented in Section 4 Hardware oversampling, and a comparison between hardware and software oversampling in Section 5 Hardware versus software oversampling comparison.

4 Hardware oversampling

This part presents the hardware oversampling unit available in the products listed in Table 1. Applicable products.

The main benefit that the user can get from the hardware oversampling is increased SNR (signal-to-noise ratio) with less CPU interaction, resulting in overall lower power consumption compared with the software-based implementation. 可以通过硬件过采样更好的增加信噪比，更少的与CPU交互，与基于软件的实现相比，总体功耗更低

4.1 Hardware oversampling feature overview 硬件过采样特性

Note: *This section concerns the STM32L4 series and information could slightly differ for other products. The dedicated documentation should be consulted.*

The hardware oversampling engine accumulates the results of ADC conversions. The accumulated output data can be right-shifted (and rounded) to provide selected bit-depth in relation to OSR. The output value is not updated every sampling period, but once N samples are accumulated, therefore, the output data rate is decimated by a factor of OSR. 硬件过采样机制累积ADC转换的结果，累积的输出数据可以右移，以提供与OSR相关的选定位深度。输出值不会在每个采样周期更新，但是一旦累积了N个样本，输出数据就会被过采样因子平均

The result is the average of accumulated samples as follows: 其结果为累积样本的平均值：

$$Result = \frac{1}{M} \times \sum_{0}^{N-1} Conversion(t_n) \qquad (11)$$

Where both N and M can be adjusted: 式中的N和M都可以调整

- N is the oversampling ratio. It is set with the OVFS[2:0] bits in the ADC_CFGR2 register. It can be a factor between 2x and 256x. N是过采样比，由ADC_CFGR2寄存器的OVFS[0:2]位设置，可以为2~256
- M is the division coefficient (right bit shift). It is set with the OVSS[3:0] bits in the ADC_CFGR2 register. It can allow to right shift the sum up to 8 bits. M为除法系数(右移)，它由ADC_CFGR2寄存器中的OVSS[3:0]位设置，允许右移8位

In the case of STM32L4 series, the oversampling engine begins summing N samples. The sum is then right shifted by M bits. The engine keeps the 16 least significant bits after the shift, and rounds the result to the nearest value according to the bits removed by the shifting. 过采样机制对N个样本求和，然后右移M位，该机制在为以后保留16位最低有效位，并根据移位根据选择的位数将结果舍入到最接近的值

The final result is saved in the ADC_DR data register and because of the 16-bit truncation, it cannot be represented on more than 16 bits. 最终结果保存在ADC_DR数据寄存器，由于16位截断，它不能超过16位表示

当累积N位的N个采样时，结果最多可以编码到X+(ln(N)/ln(2))

How to operate the bit-depth obtained with oversampling 设置过采样的深度

When N samples of X bits are accumulated, the result can be coded on up to X + (ln(N) / ln(2)) bits.

For example, if the oversampling ratio N is 256x and the samples accumulated are on 12 bits, the sum of N terms will be on 20 bits since ln(256) / ln(2) = 8 and 12 + 8 = 20. 如：过采样比N为256，累积的样本为12位，则N相之和为20位，因为ln(256)/ln(2)=8，12+8=20

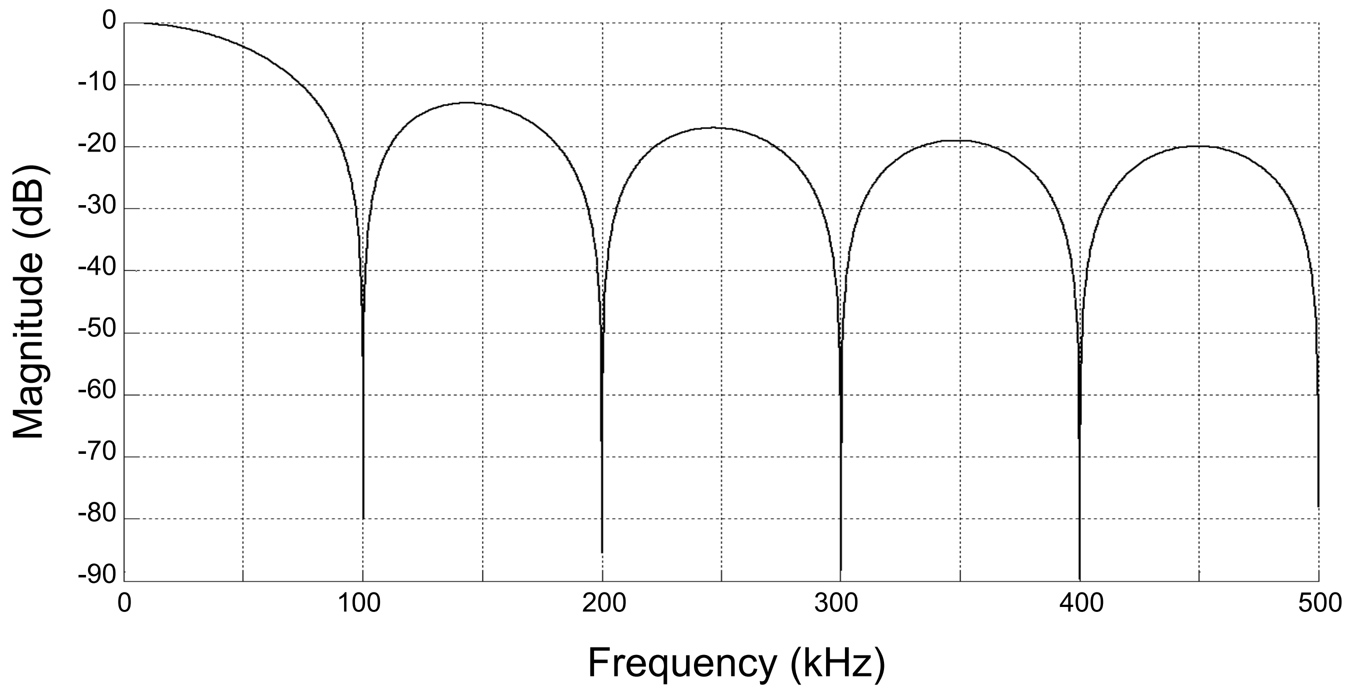
Next, the right shifting, which is up to 8 bits has to be taken into account. 接下来右移8位平均

Finally, the bit-depth is given by X + (ln(N) / ln(2)) – M but is limited to 16 bits because of the truncation. 最后位深度由X+(ln(N)/ln(2))-M得出，但由于截断则被限制为16位

Note: *The number of bits X for a sample depends on the product used and can be found in its datasheet.* 位数X需要查看芯片数据表

The *Accumulate and average* stage can be thought of as a kind of digital filter (often called accumulate-and-dump). The frequency response of such filter is equivalent to a first order Cascaded-integrator-comb (CIC1) Hogenauer filter. The frequency response in case of sampling frequency 1 MHz and OSR = 10 can be seen in Figure 13. Frequency response of accumulate-and-dump filter.
累计和的平均阶段可以认为是一种数字滤波器(通常称为累积和转储)。该滤波器的频率响应相当于一阶级联积分梳状滤波器，采样频率为1MHz，OSR=10的频率响应如下图：

Figure 13. **Frequency response of accumulate-and-dump filter**
累积-转储滤波器频率响应



Although this is not a perfect low-pass filter, the very high attenuation of the sampling frequency is a useful property. It is effective in canceling the out-of-band noise resulting in an increased signal-to-noise ratio.
虽然这不是一个完美的低通滤波器，但采样频率的衰减就是一个有用的特性，他可以有效的消除带外噪声，从而提高信噪比

5 Hardware versus software oversampling comparison

硬件和软件过采样比较

ADC过采样的方法可以硬件实现也可以软件实现

The ADC oversampling method can be implemented by hardware or by developing a dedicated software routine.

硬件实现的优点是数据处理比软件在相同的样本数下功耗更低，但是硬件过采样并不是每个产品都可使用的

The advantage of hardware implementation is that the total energy budget needed for processing the ADC acquired samples is reduced in comparison to the software implementation where all the data processing needs to be done by the core. However, the hardware oversampling unit is not available on every product.

Two test projects emulating the common data acquisition tasks have been developed and executed on the same system to evaluate the energy difference and to demonstrate how much energy can be saved by using the hardware oversampling. 在同一个系统上开发并执行两个模拟数据采集任务，评估能量差异，并演示使用硬件过采样可以节省的能量

5.1 Software implementation

The project demonstrating the software oversampling implementation method consists of the following steps, which are repeated every 100 ms: 软件实现，每100ms重复一次

1. Configuring the system/data acquisition. 配置系统/数据采集
2. Capturing of 64 samples by ADC and storing them in the memory by using DMA while the core is in Sleep low-power mode. 当Core处于睡眠低功耗模式时，通过ADC捕获64个采样并使用DMA将其存储在memory中
3. Processing the data acquired by the CPU to get an oversampled value. CPU处理获取的数据以获得过采样值
4. Putting the system in Stop mode for the rest of the 100 ms interval. 在剩下的100ms间隔内，将系统置于Stop模式

5.2 Hardware implementation

硬件实现，执行相同的任务，但是数据处理是由ADC过采样机制实现的，因此CPU可以在采样和过采样期间处于休眠状态

The project showing the hardware implementation carries out the same task, except that the data processing is done by the ADC oversampling engine. Hence, the CPU can be inactive during the acquisition and oversampling:

1. Configuring the system/data acquisition. 配置系统/数据采集
2. Capturing of 64 samples and processing them by the ADC oversampling engine while the core is in Sleep low-power mode. 当Core处于休眠低功耗模式时，通过ADC过采样机制捕获64个样本进行处理
3. Putting the system in Stop mode for the rest of the 100 ms interval.
在剩下的100ms间隔时间内，系统将处于Stop模式

5.3 Results

The energy consumption for the data acquisition and processing task, and the average current consumption for the whole 100 ms period for both demonstration projects are detailed in Table 3.

Table 3. Comparison of SW and HW implementation of ADC oversampling technique

硬件过采样的数据采集和处理时间比软件过采样略快但是没有质的加速

Implementation	Data acquisition and processing time	Acquisition task charge		Average current (during 100 ms)
Hardware	6.06 ms	896 pAh	3.23 µC	37 µA
Software	6.80 ms	1099 pAh	3.96 µC	44 µA

The hardware oversampling implementation can save about 20% of the energy consumed to complete the acquisition and data processing task with lower coding effort and CPU time.
硬件过采样可以实现节省20%的功耗，以更低的代码量和CPU负载完成采集和数据处理任务

6 ENOB (effective number of bits) measurement

The formulas to apply for a desired resolution improvement using each method are presented in the table Table 4.
各方法所需分辨率的公式：

改善ENOB的方案

Table 4. **Formulas for ENOB improvement**

Method	Formula x为ADC分辨率，OSR为过采样比 *(X is the ADC resolution OSR is the oversampling ratio)*
Hardware oversampling	Resolution = X + (ln (OSR) / ln(2)) – M M为硬件过采样的分频系数 *M is the division coefficient of the hardware oversampling engine*
Software oversampling with white noise	Resolution = X + p with OSR = 4^p
Software oversampling with dithering	Resolution = X + p with OSR = $2*2^p$

In practice, this resolution never reaches its theoretical value. A good indication of the efficiency of an ADC is to determine its ENOB (efficient number of bits). This parameter can be considered as a 'real-life' resolution that takes into account potential noise, distortion, and circuit imperfections. It also gives a good indication of its dynamic performance. 在实际中，分辨率没有达到理论值。衡量ADC效率的很好指标是确定ENOB(有效位数)，该参数可视为考虑到潜在噪声、失真和电路缺陷的"实际"分辨率，并给出其动态性能的良好指标

It is good practice to measure this parameter to verify that the resolution of an ADC is not degraded too much by its implementation and configuration. 测量此参数以验证ADC的分辨率不会因其实现和配置方式降低很多

The ENOB can be determined by several methods. In the context of this document, a formula that links it to two other parameters is used: SINAD (signal-to-noise and distortion ratio) and THD+N (total harmonic distortion + noise). ENOB可以通过几种方法确定。本文中使用一个公式将其与另外两个参数联系起来：SINAD(信噪比)和THD+N(总谐波失真率+噪声)

The formulas are as follows:

$$ENOB = \left(SINAD - 1.76 + 20_{\log}\left(Full\_scale\_amp / Input\_amp\right)\right)/6.02 \quad (12)$$

$$ENOB = \left([THD + N] - 1.76 + 20\log_{10}(Full\_scale\_amp - Input\_amp)\right)/6.02 \quad (13)$$

Full_scale_amp is the maximum amplitude that can be measured by the ADC. 是ADC可以测量的最大幅度

Input_amp is the amplitude of the signal applied to the ADC. 是应用于ADC的信号幅度

These formulas result directly from equation (5). The difference is that noise and distortions are taken into account by replacing the SNR by the SINAD or the THD+N, making it closer to a real-life situation. The amplitude of the input signal used for ENOB measurement is also taken into account thanks to the ratio Full-scale amp./ Input amp. Indeed, if the amplitude of the input signal does not fill the full amplitude reading ability of the ADC, this has to be considered when computing a ratio featuring the level of this signal.

Note: If the bandwidth of the measurement is DC to Fs/2 (the Nyquist bandwidth, Fs is the sampling frequency), THD + N is equal to SINAD. That is what we consider for the two formulas above.

The following steps can be followed to measure the ENOB of an ADC:

- With a high precision signal generator, inject a sinusoid on one of the tested ADC channels with a frequency respecting the maximums given in Table 2. Oversampling using white noise versus oversampling using triangular dither for software oversampling, or fADCmax/(2*N) for hardware oversampling with N being the oversampling ratio of the oversampling engine. The sinusoid amplitude should be 90% of the ADC full-scale to avoid saturation.
- Configure the ADC to acquire some samples of the signal. The best is to get a rounded number of the signal period. 4096 is a good example but might need to be adjusted in function of the frequency of the input signal.
- Make the successive binary codes operated by the ADC available for measurement (parallel/serial transmission, file recording…).
- Analyze the ADC measured signal with a frequency analyzer capable to do SINAD or THD+N measurement.
- With the SINAD or THD+N measurement features, get the value for one of these two parameters. Measuring both parameters enable doing a comparison of the ENOB values obtained.
- Apply the formula to determine the ENOB of the ADC (see Eq. (12) or Eq. (13)).

To provide practical data and be able to analyze the effect of oversampling on the ENOB, the above steps have been followed with the following equipment and tools.

The input sinusoid has been generated with the analog output of the **Audio Precision AP2722 Audio Analyzer**. Several input frequencies have been tested to analyze the effect on the ENOB obtained. The signal output by the AP2722 is 0-centered, so a conversion stage is needed to set its amplitude between 0 and 3.0 V (VDD=VDDA=3.3 V on STMicroelectronics EVAL and Nucleo boards).

Note: This conversion stage behaves like an HP filter and influences the signal measured by the ADC (lower signal resolution for lower frequencies). This can give a good representation of a real-life use case.

The STM32L476G-EVAL board has been used to process the ADC measurement and transmitting/recording it. The ADC sampling is done on the pin PA4 linked to the STM32L4 ADC1 Channel 9 and to ground through a 4.7nF capacitor to filter high frequency noise. This pin is available on the connector CN7 of the EVAL board.

The application running on the STM32L476G-EVAL board saves 4096 ADC samples in RAM thanks to the DMA peripheral.

The oversampling unit is used to analyze its effect and configured as presented in table 6.

A timer is set up to trigger the transfer of each sample (even the ones used for oversampling). The frequency of this timer is adjusted according to the oversampling configuration wanted.

When the 4096 samples are saved, they are transferred through the STM32 UART interface to be recovered in a file on a PC thanks to a Python script.

The resulting file is formatted so that it can be processed with MATLAB®.

The sampling rate chosen for the test is 12.5 kHz (when oversampling is used, this is the final sampling rate) to be able to keep a constant ADC clock frequency (80 MHz) and sampling time (12.5 cycles).

MATLAB® enables computing the SINAD or THD+N of the ADC signal. It has a native sinad function that is used to analyze the ADC signal saved into the file created previously. Then, applying eq.X gives us the ENOB measured.

To emphasize oversampling effects, each oversampling ratio possible has been tested while fixing the right shift coefficient to target the best resolution offered.

Table 5 presents the resolution that can be achieved in function of the oversampling ratio and the right shifting. 16 bits have been targeted. For the ratio values 2, 4 and 8 a left shifting (respectively by 3, 2 and 1 bit) has been processed to achieve the 16-bite target.

Table 5. Theoretical ENOB values for hardware oversampling unit versus configuration

OSR\M	Hardware oversampling unit coefficient (M)[1]								
	0	1	2	3	4	5	6	7	8
2	13	12	11	10	9	8	7	6	5
4	14	13	12	11	10	9	8	7	6
8	15	14	13	12	11	10	9	8	7
16	16	15	14	13	12	11	10	9	8
32	**17**	16	15	14	13	12	11	10	9
64	**18**	**17**	16	15	14	13	12	11	10
128	**19**	**18**	**17**	16	15	14	13	12	11
256	**20**	**19**	**18**	**17**	16	15	14	13	12

1. *Bold entries have no significance since the hardware oversampling unit output is limited to 16-bit data width.*

Table 6. **Practical ENOB measurement with the hardware oversampling unit versus configuration**

OSR	Hardware oversampling unit coefficient (M)(1)								
	0	1	2	3	4	5	6	7	8
2	13	12	11	10	9	8	7	6	5
4	14	13	12	11	10	9	8	7	6
8	15	14	13	12	11	10	9	8	7
16	16	15	14	13	12	11	10	9	8
32	-	16	15	14	13	12	11	10	9
64	-	-	16	15	14	13	12	11	10
128	-	-	-	16	15	14	13	12	11
256	-	-	-	-	16	15	14	13	12

1. *Bold entries have no significance since the hardware oversampling unit output is limited to 16-bit data width.*

Table 7. **Practical ENOB measurement with the hardware oversampling unit versus configuration**

OSR	Software oversampling with white noise		Software oversampling with dithering	
	Theoretical resolution	Practical ENOB	Theoretical resolution	Practical ENOB
2	-	-	12	-
4	13	-	13	-
8	-	-	14	-
16	14	-	15	-
32	-	-	16	-
64	15	-	17	-
128	-	-	18	-
256	16	-	19	-

As mentioned before, the signal test input is a 3.0Vpp sinusoid and the sampling rate is 12.5 kHz. Thus, to respect the Nyquist criteria, the following frequencies have been tested: 500 Hz, 1 kHz, 1.5 kHz, 2 kHz, 2.5 kHz.

Table 8. **Effect of oversampling on ENOB**

OVS ration	OVS right shift	OVS left shift	ENOB				
			500 Hz	1 kHz	1.5 kHz	2 kHz	2.5 kHz
None	None	None	10.4126	10.3622	10.3543	10.2774	10.3084
2	None	3	10.6245	10.6302	10.9172	10.8618	10.9524
4	None	2	10.9567	11.2234	11.2249	11.3322	11.3531
8	None	1	11.1692	11.3454	11.4646	11.5817	11.7653
16	None	None	11.2158	11.4962	11.6551	11.8414	11.9138
32	1	None	11.2718	11.6126	11.8103	12.0408	12.2725
64	2	None	11.3109	11.6220	11.8968	12.1124	12.3626
128	3	None	11.3259	11.6568	11.9050	12.1579	12.6038
256	4	None	11.3582	11.7032	11.9301	12.2419	12.8259

Note: For reference, after the conversion stage and at the ADC pin level, the following THD+N values were measured: -82.5 dB at 500 Hz (this is equivalent to 13.55 ENOB according to eq. Y), -93 dB at 1 kHz (15.29 ENOB), -96 dB at 1.5 kHz (15.79 ENOB), -97 dB at 2 kHz (15.96 ENOB) and -98 dB at 2.5 kHz (16.12 ENOB). At the audio precision analyzer output, which is before the conversion stage, -105 dB were measured (17.28 ENOB).

In the STM32L476 datasheet, it is given that the typical ENOB of the ADC is 10.5 (the ADC is configured as single-ended).

Thus, Table 8 shows that it is possible to get over this typical value and even over the real ADC resolution. However, the theoretical 16-bit target stays far from the results and corresponds more to an idea of the performance of the oversampling configuration.

The result also highlights that a higher oversampling ratio gives a better ENOB despite limiting the sampling frequency and so the input signal frequency.

7 Conclusion

This application note has explained the basics of the oversampling technique used to improve the SNR performances (and thus the effective resolution) of ADCs integrated in most of the STM32 microcontrollers.

The cornerstones of the oversampling technique are:

- The RMS quantization noise of an ADC is q / √(12), over the Nyquist bandwidth (q is the ADC quantum: LSB value)
- If the wanted bandwidth is smaller than the Nyquist bandwidth, the quantization noise is reduced in proportion by using a filter to remove the out of band noise
- Dithering can be used if the quantization noise does not behave like a wideband noise

The hardware implementation of the ADC oversampling technique reduces the time and energy needed by the CPU for the data processing tasks. It results in lowering the overall power consumption.

When the hardware oversampling unit is not available on the STM32 used, it is still possible to implement entirely the technique via software as it has been presented in this document.

The effect of oversampling on the effective ADC resolution (ENOB) has also been analyzed. With oversampling it is possible to get the effective resolution over the real one.

Revision history

Table 9. Document revision history

Date	Version	Changes
25-Sep-2023	1	Initial version.

Contents

List of tables

List of figures