Source: http://onmyphd.com/?p=analog.digital.converter

Analog-Digital Converters

What do you need to know to understand this topic?

- Basic electronics
- Digital representation

Sections

What is an analog-digital converter?

An Analog-Digital Converter (ADC) is a widely used electronic component that converts an analog electric signal (usually a voltage) into a digital representation. The ADCs are at the front-end of any digital circuit that needs to process signals coming from the exterior world. Its schematic symbol is:

The output of a microphone, the voltage at a photodiode or the signal of an accelerometer are examples of analog values that need to be converted so that a microprocessor can work with them.

How does the ADC convert a signal?

Many ways have been developed to convert an analog signal, each with its strengths and weaknesses. The choice of the ADC for a given application is usually defined by the requirements you have: if you need speed, use a fast ADC; if you need precision, use an accurate ADC; if you are constrained in space, use a compact ADC.

All ADCs work under the same principle: they need to convert a signal to a certain number of bits N. The sequence of bits represents the number and each bit has the double of the weight of the next, starting from the Most Significant Bit (MSB) up to the Least Significant Bit (LSB). In a nutshell, we want to find the sequence of bits b_{N-1}, b_{N-2}, ..., b_0 that represents the analog value V_{in} as

$$V_{in} = \sum_{n=0}^{N-1} b_n 2^n \frac{V_{ref}}{2^N}.$$

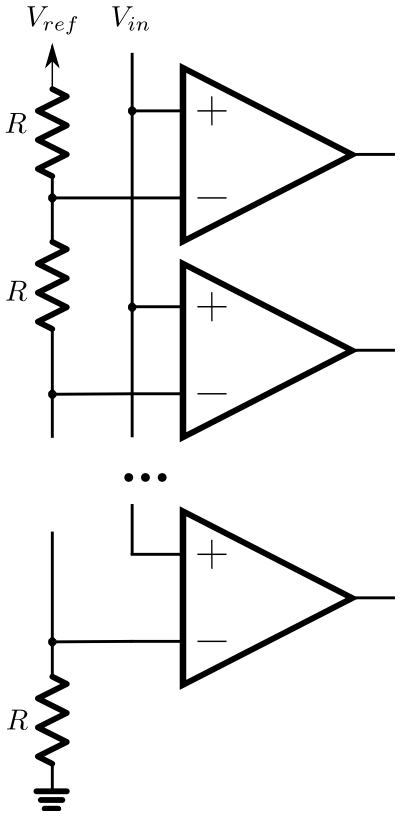
The MSB has weight $V_{ref}/2$, the next $V_{ref}/4$, etc., and the LSB has weight $V_{ref}/2^N$. Therefore, more bits leads to more precision in the digital representation. Here we simplify the range to be between 0 and V_{ref}, although the range may be between any two values.

Let's talk about the following ADCs (although there are more):

- Flash
- Pipelined
- Successive-Approximations Register (SAR)
- Integrating or Dual-slope
- Sigma Delta ($\Sigma\Delta$)

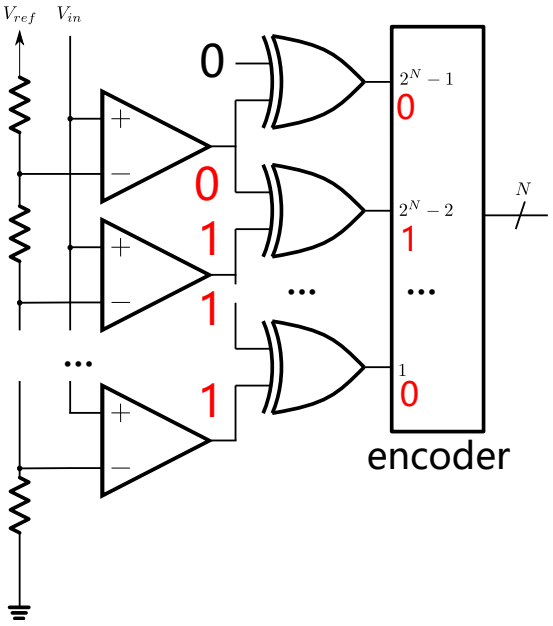
Flash ADCs

Flash converters have a resistive ladder that divides the reference voltage in 2^N equal parts. For each part, a comparator compares the input signal with the voltage supplied by that part of the resistive ladder. The output of all the comparators is like a thermometer: the higher the input value, more comparators have their outputs high from bottom to top. A dedicated component called "Priority Encoder" translates this gauge into a binary code, which corresponds to the position of the last comparator with high output, counting from the bottom up.



The Priority Encoder

The Priority Encoder has to find the position of the last comparator with high output, starting from the bottom. That means that it should find the position where neighboring comparators have different outputs (all below have output high and all above have output low). That can be simply done by XORing the outputs of neighboring comparators and feeding their outputs to a digital encoder. Only one XOR has its output active and the encoder will translate that position into a binary representation. If there are 2^N comparators, the encoder outputs a N-bit number.



Strengths

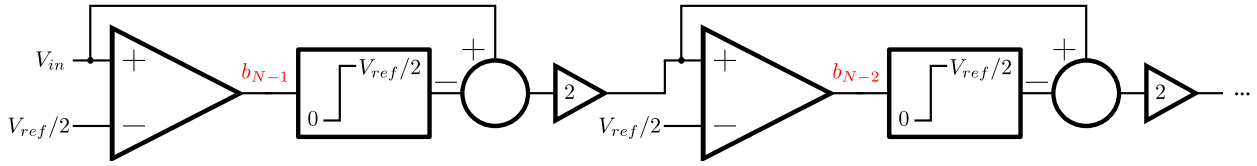
- Very fast, converts instantly

Weaknesses

- It doubles in size for each bit added to the representation. N bits require 2^{N-1} comparators.
- It has a high input capacitance (the input capacitance of a comparator multiplied by 2^{N-1})
- It consumes a lot of power

Pipelined ADCs

Pipelined converters convert the input in a number of steps proportional to the number of bits. At each step, the input signal is compared to half the reference value. If it is higher, half the reference value is subtracted to the input and the bit corresponding to that step is 1. Otherwise, it is 0. In either cases, the remaining value is doubled and passed to the next stage. Note that each stage is taking care of one bit, so a new value can be applied to the input every cycle.



Strengths

- The number of stages increases only with the number of bits
- As fast as the flash ADC

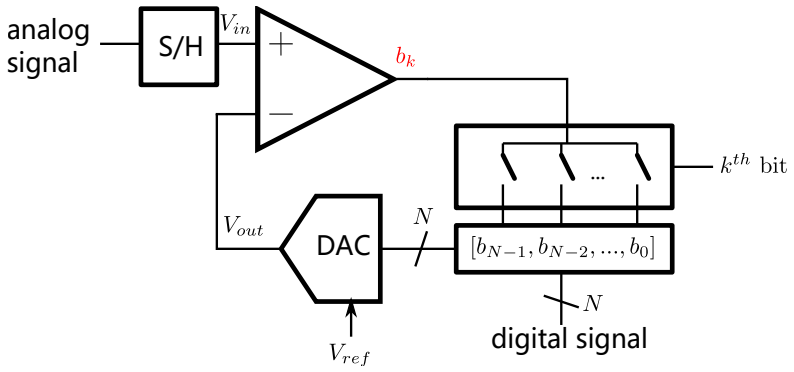
Weaknesses

- High latency - For any analog value, it takes N cycles to output the corresponding binary representation
- Any error introduced in the doubling or subtraction operations passes to the following stages

Successive Approximation Register (SAR) converters

A Successive Approximation Register converter evaluates each bit at a time, from the most to the least significant bits. They successively approach the output of a digital-analog converter (DAC) in them to the input voltage. The input of the DAC is stored in a N bit register, which is also the output of the ADC.

Let's see the flow of this ADC with the aid of the picture below. First, the analog signal is sampled and kept fixed. If the input value is changed during the conversion, the result can be completely wrong. Then, the bit $N-1$ of the register is set to 1 and every other bit to 0. Since the reference voltage of the DAC is V_{ref}, its output is set to $V_{ref}/2$. The output of the comparator b_k is latched to the MSB b_{N-1}, i.e., if $V_{in} < V_{ref}/2$, then b_{N-1} is reset to 0, otherwise it stays 1. By successively setting the next bit to 1, comparing the output of the DAC with the input voltage and latching the result in the same bit, **the converter is generating a signal from the register that is successively approximating the input value** (hence its name).

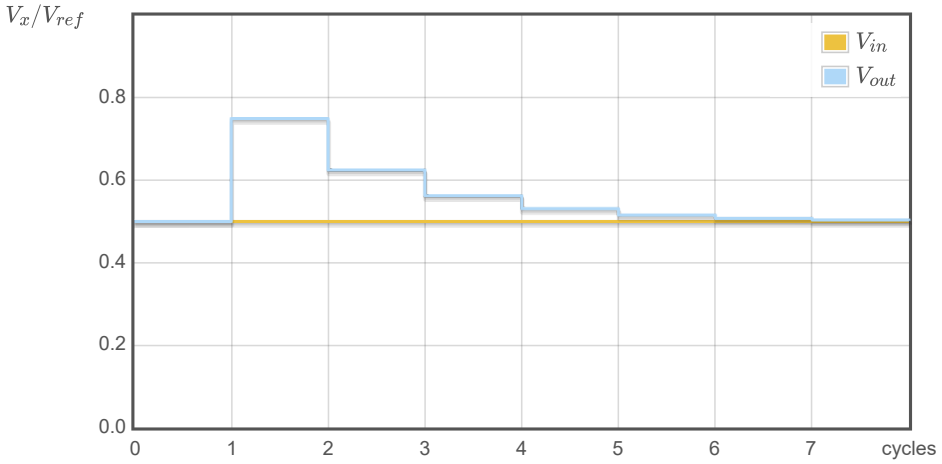


SAR ADC architecture

In a nutshell, a SAR follows these two steps for each bit, from most to least significant bit, after resetting the register value to 0:

- **Comparison:** Set bit to 1 and compare the output of the DAC with the input voltage
- **Latching:** Latch the result of the comparator to the same bit in the register

The interactive plot below shows the input signal and the output of the DAC during the comparison phase.



The slider below controls the input voltage of an 8-bit (8 cycles) SAR ADC. When you change the input voltage, you can see in the plot above that the output of the DAC tends to get closer to the input signal as more bits are defined. To see if a particular bit was set or cleared during the latching phase, you have to see if the output of the DAC at the next cycle is above or below the value of the previous cycle.

V_{in} =0.5V

Strengths

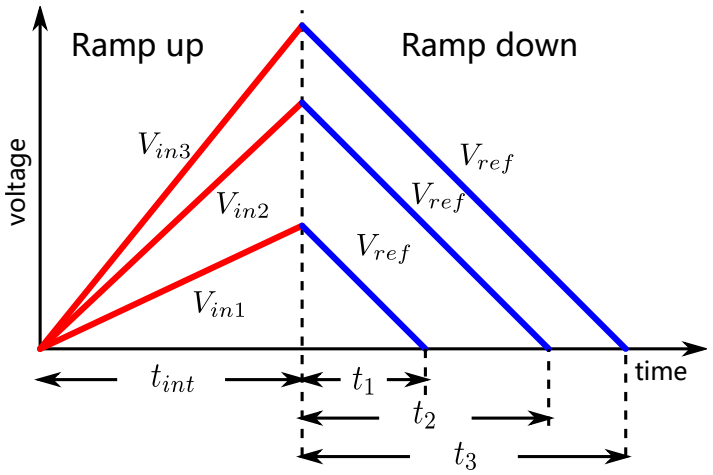
- It uses only one comparator
- Low power consumption

Weaknesses

- The DAC grows with the number of bits
- They take as many cycles to convert the signal as the number of bits
- The component mismatch in the DAC limits its linearity (and therefore of the ADC) to around 12bits

Integrating or Dual-slope converters

The dual-slope are very precise, but slow converters that use counters to generate the output. As its name suggests, this converter has 2 phases, the first where a voltage ramps up with a certain slope, and the second where the same voltage ramps down with a different slope.



Ramp up

First, **a voltage ramps up with slope proportional to the input voltage V_{in} for a fixed period of time**. This can be achieved, for example, with a current source proportional to the input voltage charging a capacitor. The voltage at the end of that integration time is:

$$V_{out}(t_{int}) = \frac{V_{in}}{\tau} t_{int} + V_{initial},$$

where t_{int} is the integration time, $1/\tau$ is the slope proportionality factor and $V_{initial}$ is the initial voltage, say zero.

Ramp down

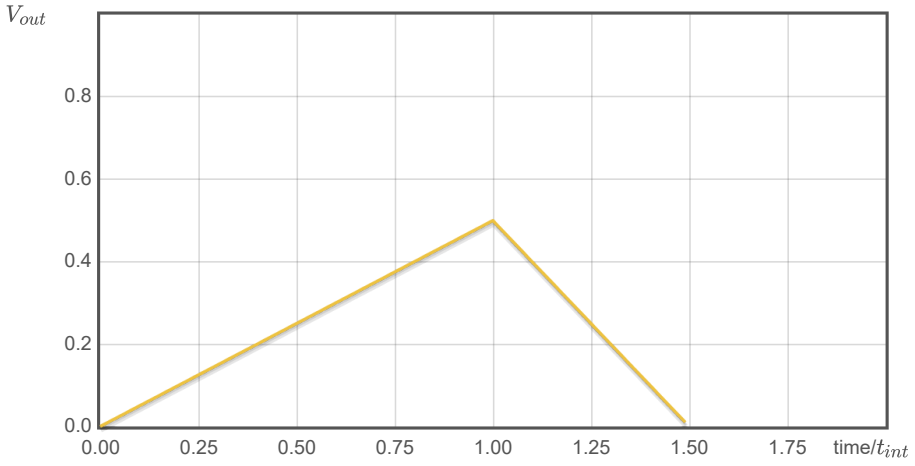
Second, **the output voltage ramps down with slope proportional to a fixed voltage V_{ref}**. Note that in the first phase, the slope is variable and the integration time is fixed. Now, the slope is fixed and the integration time is variable: **the voltage ramps down until it reaches zero**, which, by intuition, should take a period of time proportional to the input voltage. Let's see if intuition is right. The time it takes for the voltage to reach zero is:

$$0 = \frac{-V_{ref}}{\tau} t + V_{out}(t_{int})$$

$$0 = \frac{-V_{ref}}{\tau} t + \frac{V_{in}}{\tau} t_{int}$$

$$t = \frac{V_{in}}{V_{ref}} t_{int}$$

where we have assumed that $V_{initial} = 0$. As you can see, the time it takes the voltage to reach zero is indeed proportional to V_{in}, while the other terms are known. During ramp down, a counter counts **the number of clocks until the output voltage reaches zero**. The number in the counter is then proportional to the input voltage.



The slider below controls the input voltage of the ADC. When you change the input voltage, you can see in the plot above that a) the slope of the ramp up changes with V_{in} and b) V_{out} reaches 0 in a period of time proportional to V_{in}.

V_{in} =0.5V

Strengths

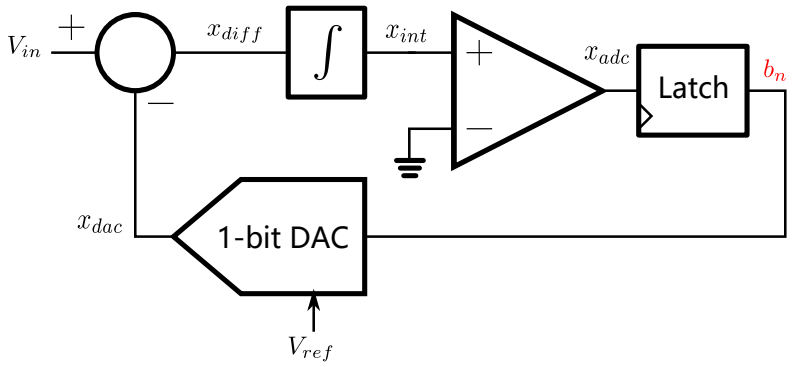
- Very precise. The sources of errors are only the comparison with zero and the clock period.

Weaknesses

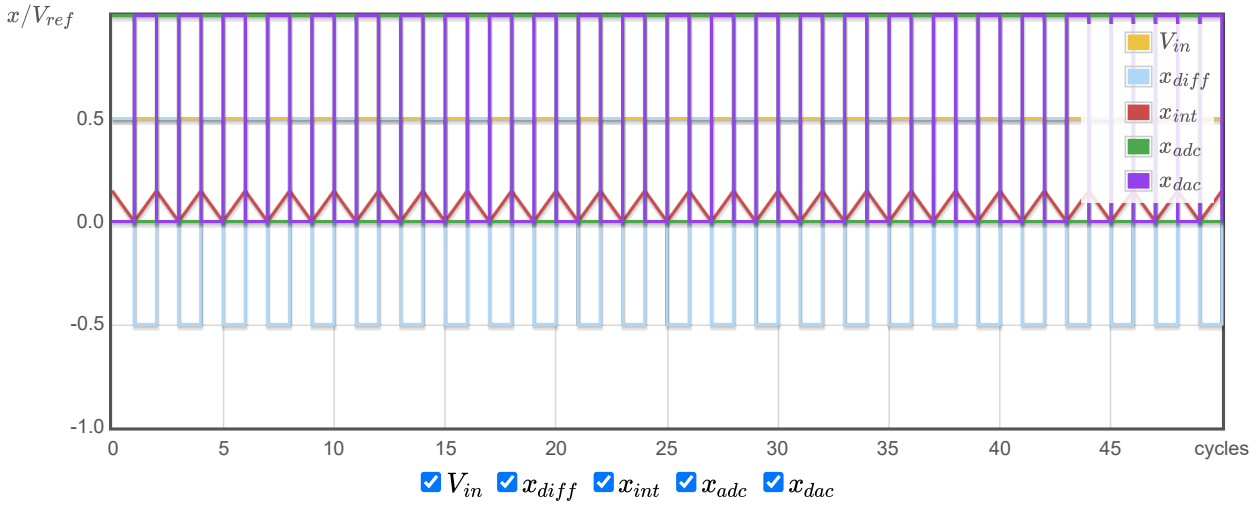
- Slow. The ADC needs time to ramp up and down the output voltage and doubles with each bit added to the representation, for a fixed clock period.

Sigma-Delta converters ($\Sigma\Delta$)

The sigma-delta converter is unique in that **it samples the signal in a much higher frequency than the Nyquist frequency**. For that reason it is also called **oversampling converter**. It converts the input signal V_{in} by integrating the error between a reference signal x_{dac} that can be either V_{ref} or zero and the input signal. Then, the ouput of the integrator x_{int} is compared with zero. That comparator result x_{adc} is sampled and sets the reference signal x_{dac} to V_{ref} or zero in the next cycle. This process is repeated over and over and the streams of 1s and zeros coming out of the second comparator average out to the input value. **To convert that bit stream into a binary code, a decimation filter is used**.



Sigma-Delta ADC architecture



The slider below controls the input voltage of a Sigma-Delta ADC. You can see that the number of cycles that $x_{adc} = 1$ is proportional to V_{in}

V_{in} =0.5V

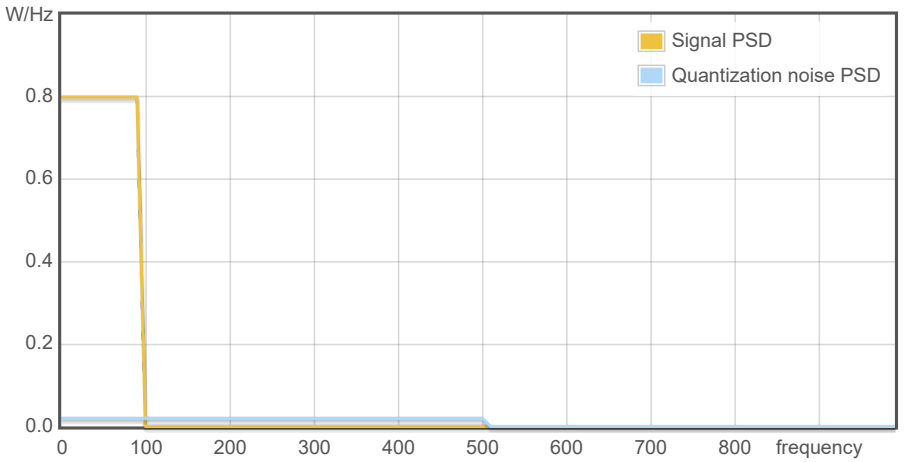
Oversampling in Sigma-Delta converters

Oversampling is the sampling at a frequency much higher than the Nyquist frequency, i.e., at a much higher frequency than the double of the maximum frequency of the signal. Although I talk about oversampling in the sigma-delta converters, oversampling can be applied in any converter. Oversampling is often associated to sigma-delta converters because they can only operate in this mode, while others can operate closer to the Nyquist frequency.

So what is so good about oversampling? The Quantization noise has a power related to the range of the LSB (Q):

$$P_{qn} = \left(\frac{V_{ref}}{2^N}\right)^2 \frac{1}{12} = \frac{Q^2}{12}$$

In the sampling process, this power gets concentrated in the frequency band between 0 and half the sampling frequency due to aliasing. Now, the power is the area of a rectangle with one side equal to the frequency band and another side equal to the Power Spectral Density (PSD - Power per Hz). If the area (quantization noise power) is to be kept the same, extending the frequency will reduce the PSD. Therefore, for the band of the signal, there is less noise power as we increase the sampling frequency.



The slider below controls the sampling frequency of the sigma-delta converter. As the sampling frequency increases, and since the power remains the same, the power per Hertz is reduced. f_s *= sampling frequency.*

f_s =1000Hz

Afterwards, an analog or digital low-pass filter can be applied to filter out the frequencies above the band of interest (the band where the signal lies).

But how does oversampling influences the Signal-Noise Ratio? Let's say that we double the sampling frequency (the rectange doubles its width, but reduces its height by half). Keeping the low-pass filter corner frequency constant, the quantization noise is reduced by half, which doubles the SNR. So, to the SNR in decibels:

$$SNR = 6.02N + 1.76$$

we add the term $10\log(\frac{f_s/2}{f_c})$, where f_c is the corner frequency of the low-pass filter and f_s is the sampling frequency. At a sampling rate equal to the Nyquist frequency ($2f_c$), this term is 0. In general, for some sampling frequency f_s, the SNR becomes:

$$SNR = 6.02N + 1.76 + 10\log(\frac{f_s/2}{f_c})$$

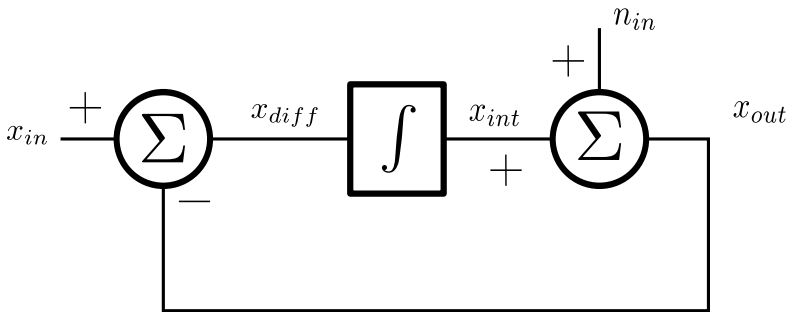
We can also denote the sampling frequency has a multiple k of the Nyquist frequency, simplifying the above equation to

$$SNR = 6.02N + 1.76 + 10\log(k)$$

Since $10\log(4) \approx 6$, multiplying the sampling frequency by 4 is like adding another bit in the SNR, as far as quantization noise is concerned.

Noise shaping

Added to the oversampling frequency, and a particular benefit of the sigma-delta converters, **the noise spectrum is shaped by a high-pass filter caused by the integrator inside the ADC**. Imagine the simplified version of the ADC shown below:



Linearized version of the Sigma-Delta ADC

We removed the 1-bit ADC, 1-bit DAC and the latch to make the system linear. x_{in} is the input signal, n_{in} is the quantization noise (notice that it is added just before where the comparator would be) and x_{out} is the output signal. The transfer function for this system can easily be determined in the Laplace domain:

$$x_{out} = \frac{x_{in} - x_{out}}{s} + n_{in}$$

$$x_{out}\left(1 + \frac{1}{s}\right) = \frac{x_{in}}{s} + n_{in}$$

$$x_{out} = \frac{x_{in}}{\left(1 + \frac{1}{s}\right)s} + \frac{n_{in}}{\left(1 + \frac{1}{s}\right)}$$

$$x_{out} = \frac{x_{in}}{s+1} + \frac{s}{s+1}n_{in}$$

The input signal is low-pass filtered, but the quantization noise is high-pass filtered. That means for the bandwidth of interest, the noise power is reduced even further by noise shaping.

The decimation filter

The decimation filter converts the bitstream at the output of the converter into a binary representation. If the bitstream is made of K bits, the decimation filter counts the number of 1s and stores the number in $\log_2(K)$ bits.

Strengths

- Due to a large oversampling, the quantization noise spectral density is reduced
- It allows noise shaping (quantization noise is attenuated at lower frequencies)
- Very simple circuits

Weaknesses

- Requires the decimation filter in the end
- Only useful in applications requiring low sampling rate, such as audio

Characteristics of ADCs

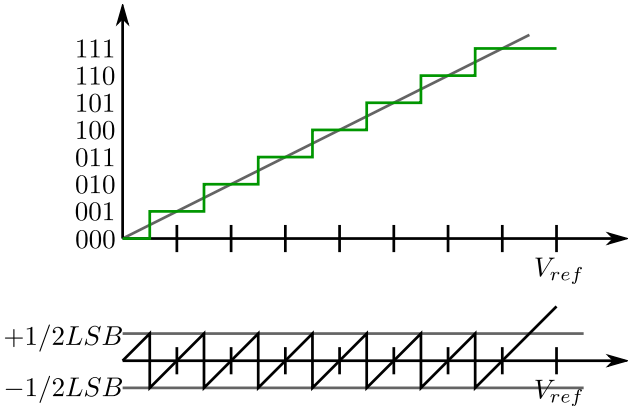
There are several metrics used to characterize an ADC. Some are related to speed and resolutions, while some of them are related to errors caused by the ADC.

ADC coding scheme

Since the ADC converts a continuous signal to a discrete representation, the ADC coding scheme can be represented by a staircase, in which a range of values of the input correspond to the same step. That range Q corresponds to the LSB and is the smallest input value that the ADC can distinguish

$$Q = \frac{V_{ref}}{2^N}.$$

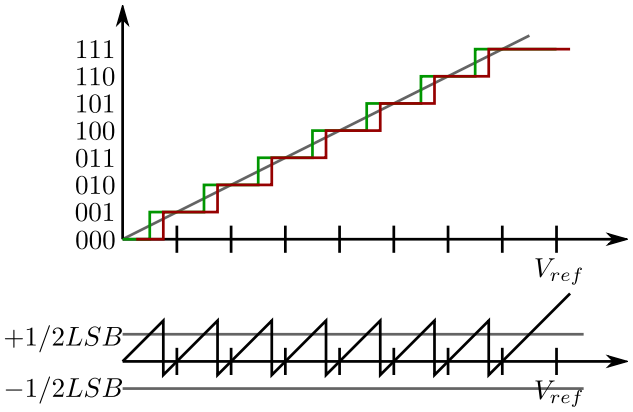
An ideal coding scheme would look as the following picture:



Even in the ideal case, there is always a minimum error of $\pm 1/2LSB$. That is called the **quantization error**. Several imperfections in the ADCs can change the staircase, introducing errors in the output code.

Offset

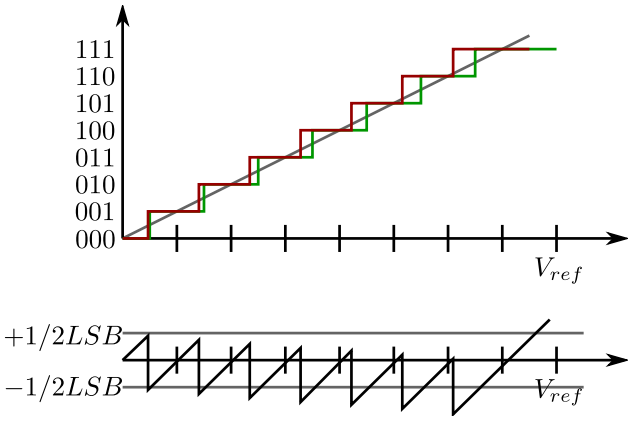
The offset is a deviation of the staircase in the input axis. The output code is changing at the wrong input, but the offset is equal for the whole range.



An offset of the converter can be caused by an offset in the comparator of a SAR converter.

Gain error

The gain error is a change in the slope of the staircase. It accumulates the error, leading to larger errors for higher output codes.



A gain error can be caused by an uncalibrated voltage reference. The output code will scale with the voltage reference and different voltage references will lead to different switching points of the output code.

Integral and Differential Nonlinearity

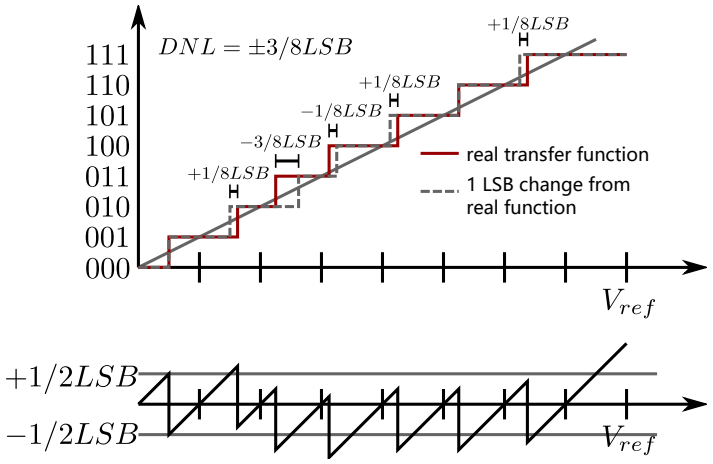
Integral Nonlinearity (INL) and Differential Nonlinearity (DNL) are two different ways of measuring the nonlinearity of a converter. In the ideal staircase, it is necessary to change the input by 1 LSB to change the output code by 1 LSB.

DNL

The DNL measures, for each code, how much more or less the input has to change to reach the next code in relation to the previous step.

$$DNL(i) = \frac{V_i - V_{i-1}}{1LSB} - 1$$

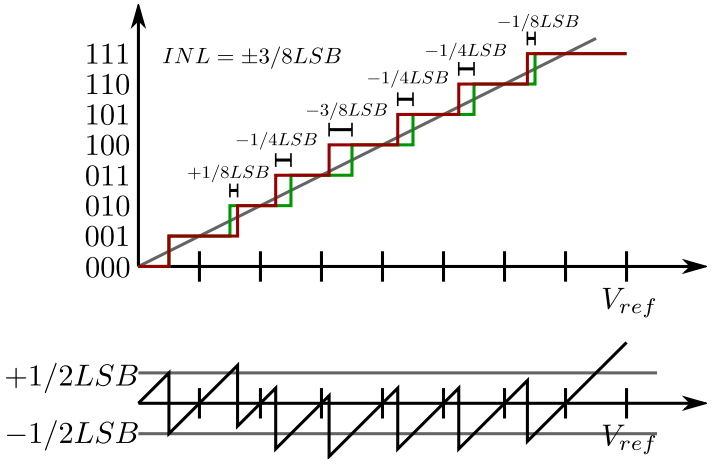
If the change $V_i - V_{i-1}$ is 1 LSB, the DNL = 0; if the change is more than 1 LSB, the DNL if positive; if the change is less than 1 LSB, the DNL is negative. It should be noted that a DNL larger than -1 LSB may lead to missing codes.



INL

The INL indicates how much the real transfer function deviates from the ideal staircase. Therefore, it measures, for each code, how much more or less the input has to change to reach the next code in relation to the ideal staircase. If $V_i^\star$ is the ideal voltage that transitions from input code $i-1$ to code i and V_i is the real one, the INL is

$$INL(i) = \frac{V_i - V_i^\star}{1LSB} - 1$$



If the change is 1 LSB, the INL = 0; if the change is more than 1 LSB, the INL if positive; if the change is less than 1 LSB, the INL is negative.

Resolution

Resolution is defined by the number of bits the output code has. This metric means little without accounting with the errors described above. For instance, an ADC with 12bit and DNL = 0 is better than an ADC with 15 bit and DNL $< \pm$ 4 LSB.

Conversion speed

The conversion speed, defined as samples per second, measures how fast the ADC can accurately convert analog values.

When designing an ADC

You may have noticed that all analog-digital converters have comparators. When designing any ADC, the specifications of the comparators are of paramount importance. That is because:

- They must have an input-referred offset voltage below 1 LSB
- They must have an input-referred noise below 1 LSB
- They must resolve a comparison for differential inputs below 1 LSB

Now, imagine a 12-bit ADC with a reference voltage V_{ref} of 2.5V.

$$1LSB = \frac{V_{ref}}{2^{12}} = 610\mu V$$

The comparator requirements become very demanding for high-resolution ADCs.