Chapter 4:

AD & DA Conversion

In the Time and Frequency Domains

Contents

References

[1] M.J.Roberts, "Signals and Systems_Analysis using Transform Methods and Matlab", Tata McGraw-Hill, 2005

[2] L.F.Chaparro , „Signals and Systems using Matlab", Academic Press, 2015

1. Overview of AD-DA Chain

Since most of the signals that we measure and transmit are in analog form, but most of the signal processing is done nowadays in digital form, it is essential for us to consider the conversion of analog-into-digital and digital-into-analog signals.

Figure 1 below shows the overview of such a conversion chain. An initially analog signal (e.g. acquired from a sensor or received via an antenna), is filtered by an anti-aliasing, low-pass filter, before being converted by an ADC (analog to digital converter).

The digital signal can be then processed by the DSP (digital signal processor), and its digital output can be converted back into an analog signal by the DAC (digital to analog converter). Before sending this signal further, it is usually required to smooth it out using again a low-pass filter, which is here called a reconstruction or anti-imaging filter.

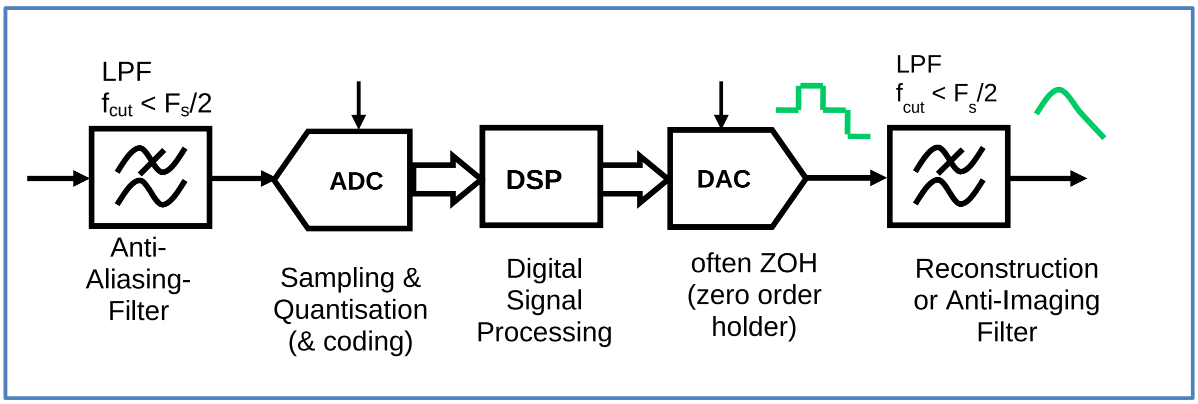


Figure 1 : AD/DA Conversion Chain

2. ADC steps: sampling and quantisation

In order to correctly design or select the blocks in the AD/DA conversion chain, we need to be able to answer questions like:

- How fast do we need to sample the analog signal in order to keep its original information?
- How much quantisation noise is introduced by the ADC?

The first question is better addressed in the frequency domain, and defined by the Shannon-Theorem or Nyquist criterium.

The sampling frequency Fs must be at least twice as big as the highest frequency fmax occurring in in the analog signal. Thus the following holds: Fs > 2 . fmax

If this criteria is not fulfilled, the signal will be distorted, because the signal components with frequencies above Fs/2 will be “shifted in the frequency domain” by the aliasing effect.

There are basically 2 ways to prevent aliasing. One is to select a higher sampling frequency (which is not always possible for a given hardware), and the other is to deploy an anti-aliasing filter (AAF), which limits the bandwidth of the incoming filter.

The anti-aliasing filter is therefore built before an ADC input, and it is an analog low-pass filter, with a cut-off frequency below 40% of Fs. If you are working with a filter of lower order (e.g. 1st order), you may need to decrease even further the cut-off frequency to cause enough attenuation of the frequency components which would be aliased.

Figure 2 below represents the two methods to avoid aliasing.

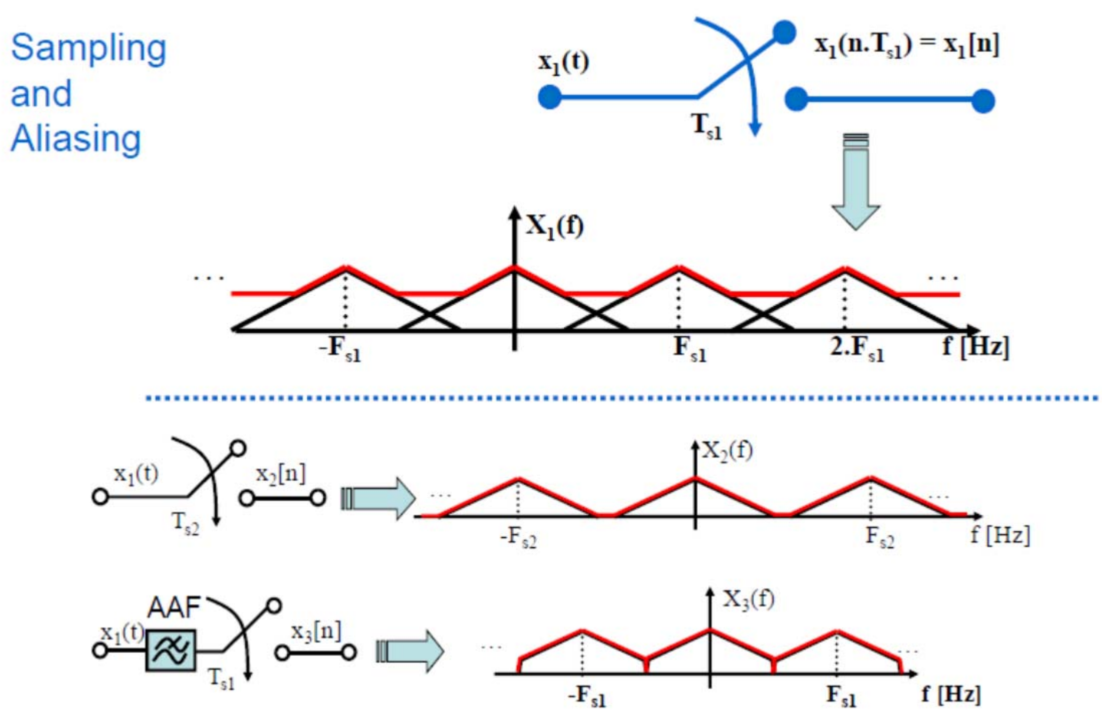


Figure 2 : Sampling, aliasing effect and anti-aliasing filter
The aliasing effect can be avoided by either increasing the sampling frequency (which is not always possible), or by low pass filtering the continuous signal before sampling.

Let us consider now, why the sampling of a continuous signal x(t) causes its spectrum X(f) to appear shifted around multiples of Fs.

The ideal sampling of a continuous time function x(t), can be represented as the multiplication of x(t) with a comb function p(t). The comb function p(t) is a periodic sequence of dirac impulses with period Ts. This is called the ideal impulse sampling and it is shown below in figure 3.

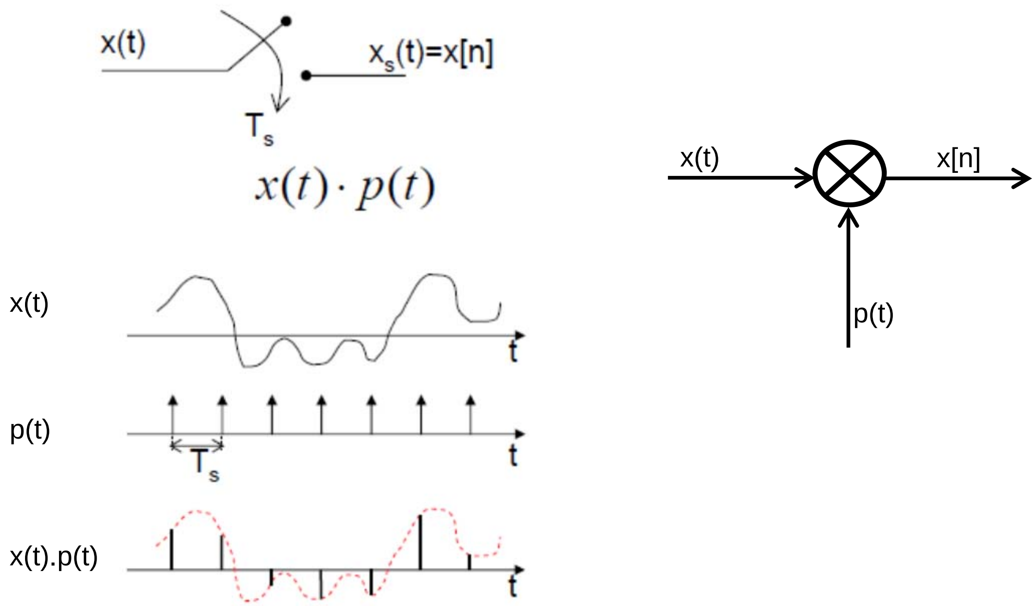


Figure 3 The ideal sampling corresponds to multiplication with train of impulses in time domain. The periodic train of impulses spaced by Ts is called a comb function.

Since p(t) is a periodic function, we can calculate its Fourier series representation. This has already been done in chapter 3, and it gives constant ck coefficients equal to 1/Ts.

$$p(t)=\sum_{n=-\infty}^{+\infty}\delta\left(t-nT_S\right)=\sum_{k=-\infty}^{+\infty}c_k\cdot\exp\left(j2\pi kF_S t\right) \qquad \text{with} \qquad c_k=\frac{1}{T_S}$$

Therefore we can also express p(t) by its Fourier series, and use this result to obtain a mathematical equation for $x_s(t) = x[n]$.

$$p(t)=\sum_{k=-\infty}^{+\infty}\frac{1}{T_s}\cdot\exp\left(j2\pi kF_S t\right) \qquad \text{and} \qquad x_S(t)=x(t)\cdot p(t)=\frac{1}{T_s}\sum_{k=-\infty}^{+\infty}x(t)\cdot\exp\left(j2\pi kF_S t\right)$$

Now, in order to determine the spectrum $X_s(f)$ of time discrete or time sampled function $x_s(t)$ we only need to consider the linearity and frequency shift properties of the Fourier transformation.

The linearity property helps us to deduce that, since we have the sum of several components in the time domain (x(t) times the complex exponential), we will have the sum of several spectral components in the frequency domain.

The frequency shift property helps us to find the spectrum of a single component: x(t) times the complex exponential. This will result the same shape as the original spectrum X(f), but shifted around $k.F_S$.

$$x_S(t) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} x(t) \cdot \exp\left(j2\pi k F_S t\right) \quad \xleftrightarrow{FT} \quad X_S(f) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} X(f - k \cdot F_S)$$

The copies of the original spectrum around multiples of the sampling frequency Fs are called the image spectra, as already shown in figure 2.

The effect of the sampling can also be considered directly in the frequency domain, using the multiplication versus convolution property of the Fourier Transformation. Refer to the section "Fourier Transformation for Discrete Signals" in chapter 3 "Fourier Transformation" of the script, for the demonstration.

An example of sampling and the resulting periodic spectrum with the image spectra is shown below in figure 4. The frequency range of the original spectrum is called the 1st Nyquist zone. And the frequency range of the image spectra are the 2nd and higher Nyquist zones.

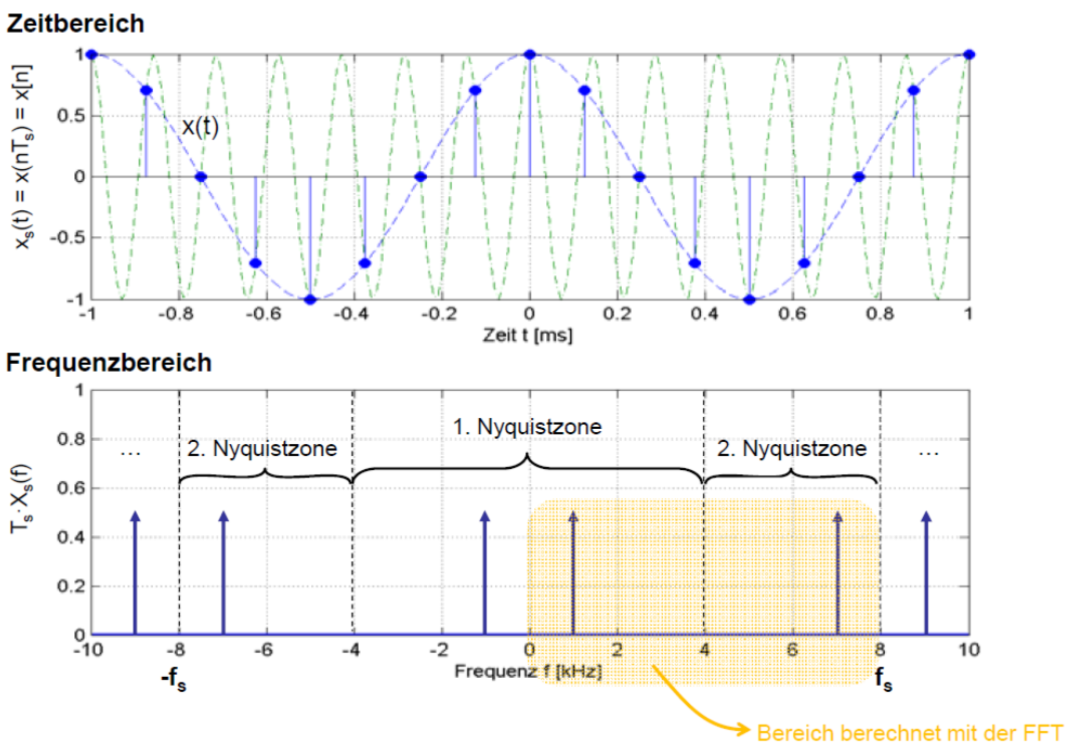


Figure 4 : A continuous sinusoidal time signal with fundamental frequency 1kHz, is sampled with Fs = 8KHz. Visualization of the periodic spectrum of the discrete signal, containing the original, plus the image spectra.

So far in chapters 3 and 4 we used the DFT algorithmus to calculate numerical approximation of the spectrum of discrete signals. At this point, we can consider the question:

Which part of the periodic spectrum of the discrete signal is calculated with the DFT?

The DFT calculates the spectrum in the frequency range 0—Fs, and assumes that this part is repeated periodically shifted by multiples of Fs (because of the sampling effect).

The second signal transformation (besides sampling) that happens within the ADC is the quantisation. Because the digital output signal $x_q[n]$, is not only discrete in time, but also in amplitude. It has a limited number of possible amplitude values, which are defined by the resolution (number of bits available) to represent the digital output signal.

The characteristic curve of the quantisation in an example ADC (with only 3 bits resolution) is shown in figure 5 below. Besides the quantisation steps one can also recognize the allowable input value range for the ADC, which is called the dynamic range ("Aussteuerbereich").

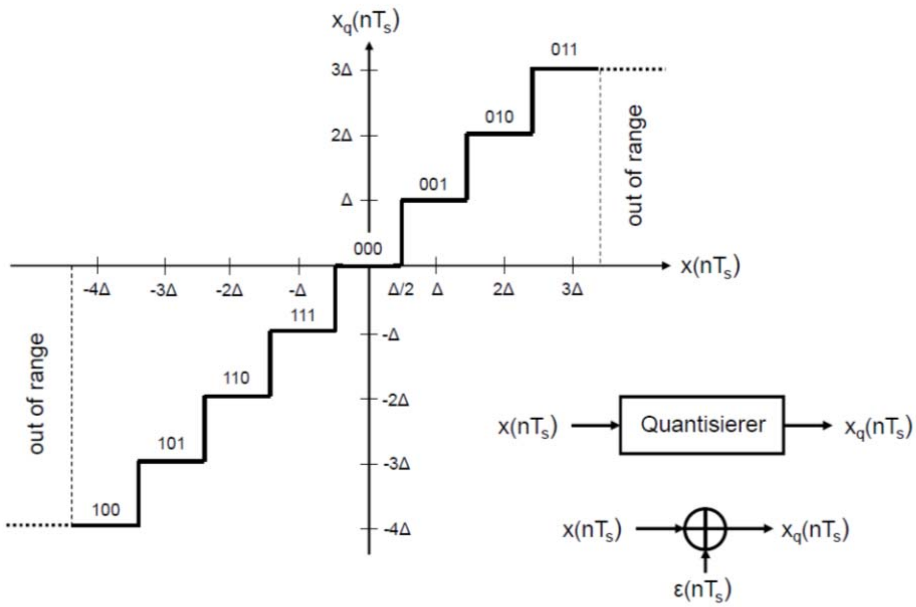


Figure 5 : QuantisationCharacteristic curve (Kennlinie)

After the quantisation the time signal $x_q[n]$ has a modified amplitude value as compared to the original amplitude in $x_s[n]$. The difference in the amplitude is an error which is introduced by the quantisation and it is called the quantisation error.

The quantisation error has a maximum amplitude equals to half of the size of the quantisation step (difference between two consecutive possible amplitude values), and it decreases by half if you increase by 1 bit the resolution of the ADC.

The amount of noise introduced by the quantisation can also be expressed in terms of a power ratio. How much power has the pure signal component as compared to the power of the noise. This ratio is called SNR (signal to noise ratio) and it varies with 6dB/bit.
Summarizing, the operations of the ADC discussed in this section are:

- **Sampling** : multiplication in the time domain with comb-function. Causes the spectrum to become periodic around multiples of Fs. Aliasing can occur, and should be prevented with an anti-aliasing filter.

- **Quantisation:** value approximation with fix number of bits (equivalent to addition of quantisation noise). Every bit reduces noise (or signal to noise ratio SNR) by 6 dB. This can also be observed in the SFDR (spurious free dynamic range) verified in laboratory 4.

3. DAC steps: holding and smoothing

Now what happens on the side of the DAC? Also two operations:

- Holding the discrete values over a Ts interval, to produce a continuous time signal;
- Smoothing out the staircase output of the holder, which is equivalent in the frequency domain to attenuate the image spectra (generated by the sampling).

Let us investigate these two operations in detail.

The most simpler holder is called a zero order holder (ZOH), and it simply keeps the value of the discrete signal constant during an interval of Ts, until the next discrete samples comes in. The output of the zero order holder looks like a staircase signal (see figure 6).

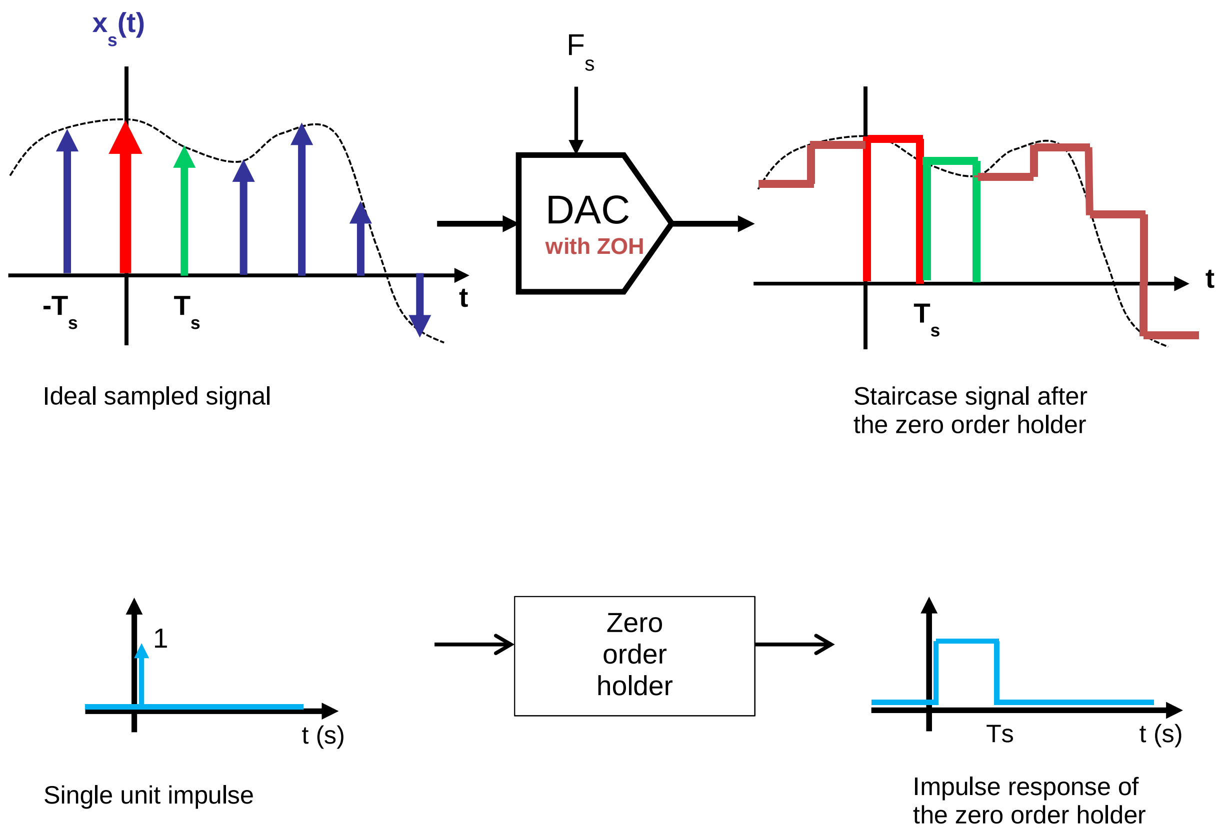


Figure 6: Zero order holder (ZOH)

In order to characterize the ZOH it is useful to consider its answer to a single unit impulse. This gives a rectangular pulse with width Ts. We can then transform this impulse response with the Fourier transformation, which gives us the frequency response of the ZOH. The frequency response of the ZOH has the shape of a sinc, with zero crossings at multiples of 1/Ts = Fs .

The ZOH already attenuates a bit the image spectra, but usually not enough. Therefore it is common to deploy after the ZOH an additional low pass filter (LPF), called reconstruction or anti-imaging filter. Figure 7 represents the spectra of both the ZOH and the reconstruction LPF.

The cut off frequency of the reconstruction LPF is usually around 40% of Fs.

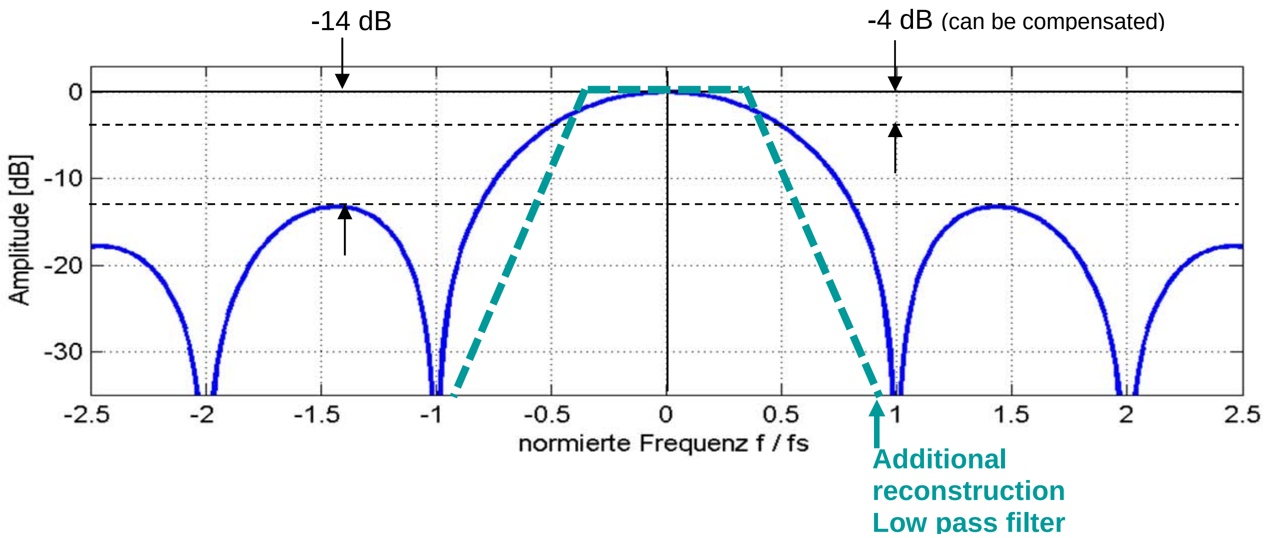
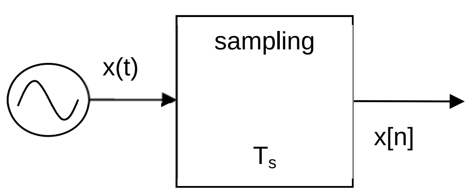


Figure 7: Spectrum of zero order holder (ZOH) and anit-imaging or reconstruction low pass filter (LPF)

Question 4-1

(a) Start a new Matlab script, and define two cosine signals $x_1(t)$ and $x_2(t)$ with equal amplitude and frequencies of 1kHz und 7KHz. Use for this definition a time vector with fine resolution, such that:
tstep < min{Period)} /100

(b) Define now in your Matlab script the corresponding discrete signals x1[n] and x2[n], that would result after sampling $x_1(t)$ and $x_2(t)$ with a sampling frequency Fs = 8kHz.



Hint:
you can define the discrete signals based on a new „rough“ time vector with tstep=1/8kHz, or you can pick on $x_1(t)$ and $x_2(t)$ only the points left after the sampling.

(c) Prepare in Matlab plots of the “time continuous” signals x1(t) and x2(t) and of the time discrete signals x1[n] and x2[n] .
Hint: in order to better visualize the continuous and discrete characteristics of the signals, use the commands *plot()* and *stem()*.

(d) After the sampling, can you still distinguish the frequency of the discrete signals x1[n] and x2[n] ? If these were unknown signals at the input of an ADC, which frequency would you suppose for them, when observing the signals at the output of the ADC? Which of these two signals has been distorted by the aliasing effect?

(e) In your Matlab script (without anti-aliasing filter), which Fs would be necessary to correctly sample x1(t) and x2(t). Verify your answer with a plot.

Question 4-2 Example of sinus function step by step over the AD-DA chain

Draw a series of sketches in time and frequency domain representing the signal $x_1(t)$ from the previous exercise along the AD-DA chain.