Digital Signal Processing

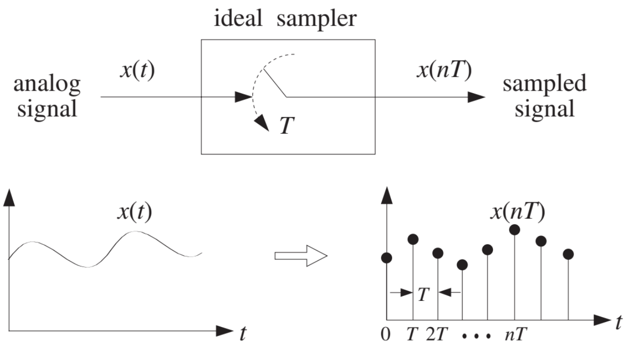
Vu Nhat Huy

Faculty of Electrical and Electronics Engineering
Department of Telecommunications Engineering
Ho Chi Minh City University of Technology
Contact: *huy.vunhat108@hcmut.edu.vn*

Ho Chi Minh city - Ngay 29 thang 8 nam 2025

Contents

Sampling is to convert a continuous time signal into a discrete time signal. The analog signal is periodically measured at every T seconds.



$$x(n) \equiv x(nT) = x(t = nT),\ n = \ldots -2, -1, 0, 1, 2, 3\ldots$$

- T: sampling interval or sampling period (second);
- Fs=1/T: sampling rate or frequency (samples/second or Hz)

Aliasing of Sinusoids In general, the sampling of a continuous-time sinusoidal signal $x(t) = A\cos(2\pi F_0 t + \theta)$ at a sampling rate $F_s = 1/T$ results in a discrete-time signal $x(n)$.

The sinusoids $x_k(t) = A\cos(2\pi F_k t + \theta)$ is sampled at F_s, resulting in a discrete time signal $x_k(n)$.

If $F_k = F_0 + kF_s, k = 0, \pm1, \pm2, \ldots,$ then $x(n) = x_k(n)$.

Sampling Theorem For accurate representation of a signal $x(t)$ by its time samples $x(nT)$, two conditions must be met:

1) The signal $x(t)$ must be band-limited, i.e., its frequency spectrum must be limited to $F_{\max}$.
2) The sampling rate F_s must be chosen at least twice the maximum frequency $F_{\max}$. $F_s \geq 2F_{\max}$
   - $F_s = 2F_{\max}$ is called Nyquist rate.
   - $F_s/2$ is called Nyquist frequency.
   - $[-F_s/2, F_s/2]$ is Nyquist interval.

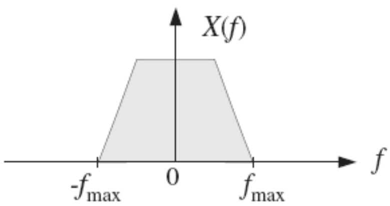


Figure 1: Typical band-limited spectrum

Practical antialiasing prefilter

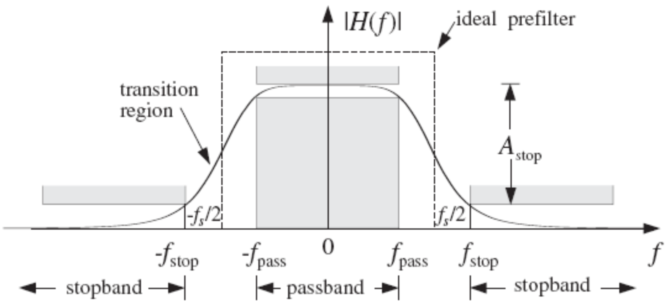
- A lowpass filter: Passband $[-F_{pass}, F_{pass}]$ is the frequency range of interest for the application ($F_{max} = F_{pass}$).
- The stopband frequency F_{stop} and the minimum stopband attenuation Astop dB must be chosen appropriately to minimize the aliasing effects.
- The Nyquist frequency $F_s/2$ is in the middle of transition region.

$$F_s = F_{pass} + F_{stop}$$

- The attenuation of the filter in decibels is defined as (where f_0 is a convenient reference frequency, typically taken to be at DC for a lowpass filter):

$$A(F) = -2\log_{10}\left|\frac{H(F)}{H(F_0)}\right| \quad (dB)$$

- $\alpha_{10} = A(10F) - A(F)$ **(dB/decade)**: the increase in attenuation when F is changed by a factor of ten.
- $\alpha_2 = A(2F) - A(F)$ **(dB/octave)**: the increase in attenuation when F is changed by a factor of two.



Quantization process The quantized sample $x_Q(nT)$ is represented by **B bit**, which can take 2^B possible values.

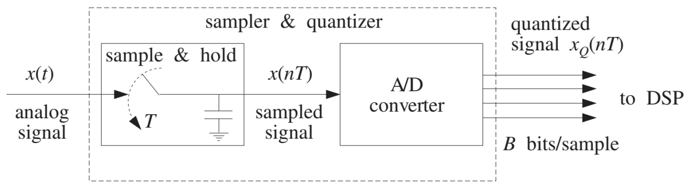


Figure 2: Analog to digital conversion

An A/D is characterized by a **full-scale range R** which is divided into 2^B quantization levels. Typical values of R in practice are between 1-10 volts.

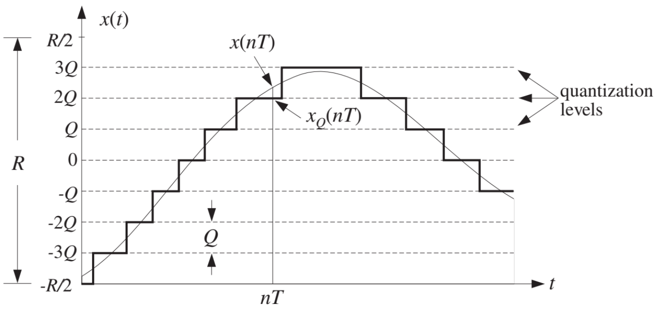


Figure 3: Signal quantization

Quantizer resolution or quantization width (step): $Q = \dfrac{R}{2^B}$

- A **bipolar** ADC: $-\dfrac{R}{2} < x_Q(nT) < \dfrac{R}{2}$
- A **unipolar** ADC: $0 < x_Q(nT) < R$

Quantization by ***rounding***: replace each value x(nT) by the ***nearest*** quantization level.
Quantization by ***truncation***: replace each value x(nT) by its ***below nearest*** quantization level.
Quantization error: $e(nT) = x_Q(nT) - x(nT)$. Consider rounding quantization: $-\dfrac{Q}{2} < e < \dfrac{Q}{2}$

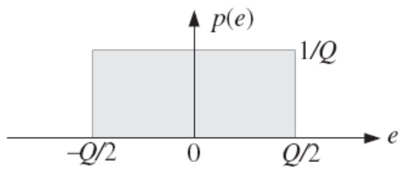


Figure 4: Uniform probability density of quantization error

Root-mean-square (rms) error: $e_{rms} = \sigma_q = \sqrt{\overline{e^2}} = \dfrac{Q}{\sqrt{12}}$

R and Q are the ranges of the signal and quantization noise, then the **signal to noise ratio (SNR)** or **dynamic range** of the quantizer is defined as

$$SNR_{dB} = 10\log_{10}\left(\frac{\sigma_x^2}{\sigma_q^2}\right) = 20\log_{10}\left(\frac{R}{Q}\right) = 20\log_{10}(2^B) = 6B\ dB$$

which is referred to as **6 dB bit rule.**

Digital to Analog Converters (DACs)

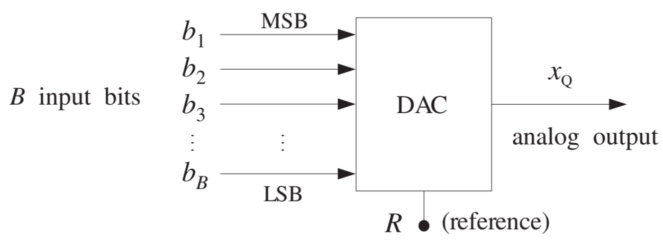


Figure 5: B-bit D/A converter

Vector B input bits: $b = [b_1, b_2, \ldots, b_B]$. Note that b_B is the least significant bit (LSB) while b_1 is the most significant bit (MSB).

For unipolar signal, $x_Q \in [0, R)$; for bipolar $x_Q \in [-R/2, R/2)$.

- ***Unipolar natural binary*** $x_Q = R(b_1 2^{-1} + b_2 2^{-2} + ... + b_B 2^{-B}) = Qm$ where m is the integer whose binary representation is $b = [b_1, b_2, \ldots, b_B]$.

$$m = b_1 2^{B-1} + b_2 2^{B-2} + ... + b_B 2^0$$

- ***Bipolar offset binary***: obtained by shifting the x_Q of unipolar natural binary converter by half-scale R/2:

$$x_Q = R(b_1 2^{-1} + b_2 2^{-2} + ... + b_B 2^{-B}) - \frac{R}{2} = Qm - \frac{R}{2}$$

- ***Two's complement code***: obtained from the offset binary code by complementing the most significant bit, i.e., replacing b_1 by $\overline{b_1} = 1 - b_1$.

$$x_Q = R(\overline{b}_1 2^{-1} + b_2 2^{-2} + ... + b_B 2^{-B}) - \frac{R}{2}$$

A/D converters

A/D converters quantize an analog value x so that is is represented by B bits $b = [b_1, b_2, \ldots, b_B]$

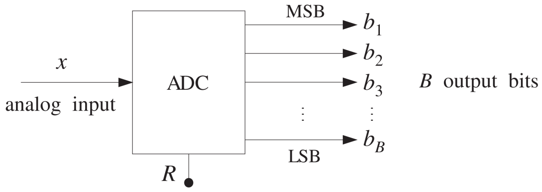


Figure 6: B-bit A/D converter

One of the most popular converters is the successive approximation A/D converter

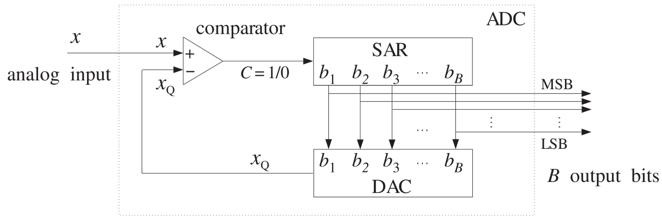


Figure 7: Successive approximation A/D converter

After B tests, the **successive approximation register (SAR)** will hold the correct bit vector b.

❖ Successive approximation algorithm

for each x to be converted, do:
 initialize $\mathbf{b} = [0, 0, \dots, 0]$
 for $i = 1, 2, \dots, B$ *do:*
  $b_i = 1$
  $x_Q = \mathrm{dac}(\mathbf{b}, B, R)$
  $b_i = u(x - x_Q)$

Truncation quantization

for each x to be converted, do:
 $y = x + Q/2$
 initialize $\mathbf{b} = [0, 0, \dots, 0]$
 for $i = 1, 2, \dots, B$ *do:*
  $b_i = 1$
  $x_Q = \mathrm{dac}(\mathbf{b}, B, R)$
  $b_i = u(y - x_Q)$

Rounding quantization

where the unit-step function is defined by

$$u(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

for each x to be converted, do:
 $y = x + Q/2$
 initialize $\mathbf{b} = [0, 0, \dots, 0]$
 $b_1 = 1 - u(y)$
 for $i = 2, 3, \dots, B$ *do:*
  $b_i = 1$
  $x_Q = \mathrm{dac}(\mathbf{b}, B, R)$
  $b_i = u(y - x_Q)$

Two's complement

	natural binary		offset binary		2's C
$b_1b_2b_3b_4$	m	$x_Q = Qm$	m'	$x_Q = Qm'$	$b_1b_2b_3b_4$
—	16	10.000	8	5.000	—
1 1 1 1	15	9.375	7	4.375	0 1 1 1
1 1 1 0	14	8.750	6	3.750	0 1 1 0
1 1 0 1	13	8.125	5	3.125	0 1 0 1
1 1 0 0	12	7.500	4	2.500	0 1 0 0
1 0 1 1	11	6.875	3	1.875	0 0 1 1
1 0 1 0	10	6.250	2	1.250	0 0 1 0
1 0 0 1	9	5.625	1	0.625	0 0 0 1
1 0 0 0	8	5.000	0	0.000	0 0 0 0
0 1 1 1	7	4.375	−1	−0.625	1 1 1 1
0 1 1 0	6	3.750	−2	−1.250	1 1 1 0
0 1 0 1	5	3.125	−3	−1.875	1 1 0 1
0 1 0 0	4	2.500	−4	−2.500	1 1 0 0
0 0 1 1	3	1.875	−5	−3.125	1 0 1 1
0 0 1 0	2	1.250	−6	−3.750	1 0 1 0
0 0 0 1	1	0.625	−7	−4.375	1 0 0 1
0 0 0 0	0	0.000	−8	−5.000	1 0 0 0

Figure 8: Converter codes for B = 4 bits, R = 10 volts

Unit sample sequence (unit impulse): $\delta(n) \begin{cases} 1 & for\ n = 0 \\ 0 & for\ n \neq 0 \end{cases}$

Unit step signal: $\delta(n) \begin{cases} 1 & for\ n \geq 0 \\ 0 & for\ n < 0 \end{cases}$

Input/output rules.

A discrete-time system is a processor that transform an input sequence $x(n)$ into an output sequence $y(n)$.

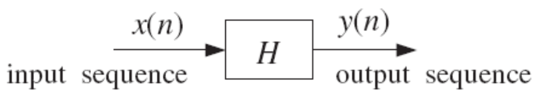


Figure 9: Discrete-time system

Sample-by-sample processing: $\{x_0, x_1, x_2, ..., x_n\} \xrightarrow{H} \{y_0, y_1, y_2, ..., y_n\}$ that is, $x_0 \xrightarrow{H} y_0$, $x_1 \xrightarrow{H} y_1, ...$ and so on.

Block processing: $x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \end{bmatrix} \xrightarrow{H} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \end{bmatrix} = y$

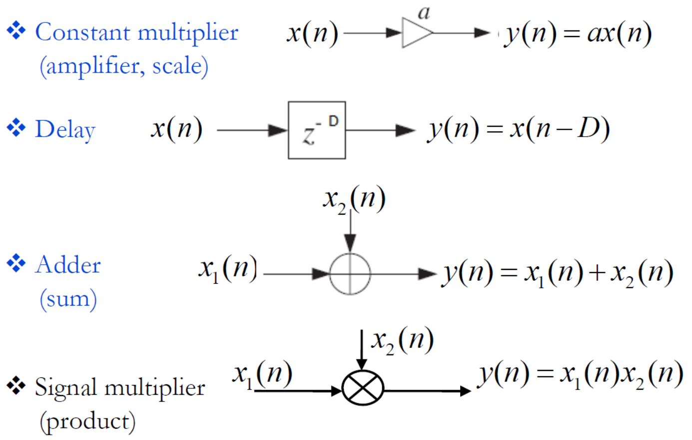


Figure 10: Basic building blocks of DSP systems

Linearity and time invariance

A **linear system** has the property that the output signal due to a linear combination of two input signals can be obtained by forming the same linear combination of the individual outputs.

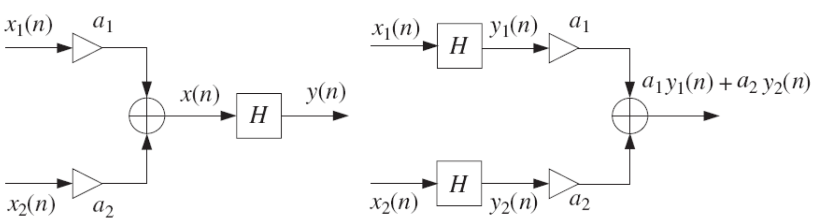


Figure 11: Testing linearity

If $x(n) = a_1x_1(n) + a_2x_2(n) \rightarrow y(n) = a_1y_1(n) + a_2y_2(n)\ \forall a_1, a_2 \rightarrow$ linear system. Otherwise, the system is nonlinear.

A **time-invariant system** is a system that its input-output characteristics do not change with time.

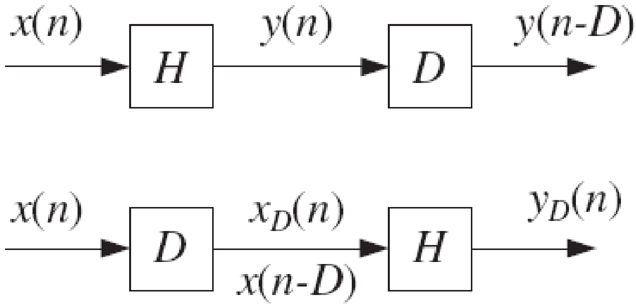


Figure 12: Testing time invariance

If $y_D(n) = y(n-D)\ \forall D \rightarrow$ time-invariant system. Otherwise, the system is time-variant.

Impulse repsonse

Linear time-invariant (LTI) systems are characterized uniquely by their impulse response sequence $h(n)$, which is defined as the response of the systems to a unit impulse $\delta(n)$.

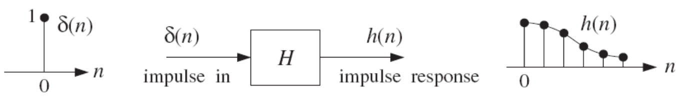


Figure 13: Impulse response of an LTI system

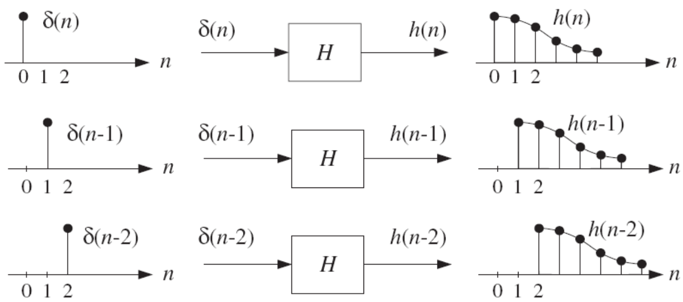


Figure 14: Delayed impulse response of an LTI system

A **finite impulse response (FIR) filter** has impulse response $h(n)$ that extend only over a finite time interval, say $0 \leq n \leq M$.

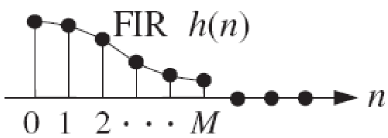


Figure 15: FIR impulse response

- M: filter order; $L_h = M + 1$: the length of impulse response.
- $h = \{h_0,\ h_1,\ ...,\ h_M\}$ is referred by varius name such as filter coefficients, filter weights or filter taps.
- FIR filtering equation: $y(n) = \sum_{m=0}^{M} h(m)x(n-m) = h(n) * x(n)$

A **infinite impulse response (IIR) filter** has impulse response $h(n)$ of infinite duration, say $0 \leq n \leq \infty$.

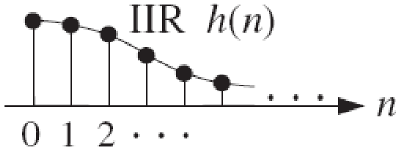


Figure 16: IIR impulse response

IIR filtering equation: $y(n) = \sum_{m=0}^{M} h(m)x(n-m) = h(n) * x(n)$. The I/O equation of IIR filters are expressed as the recursive differences equation.

Causality and Stability

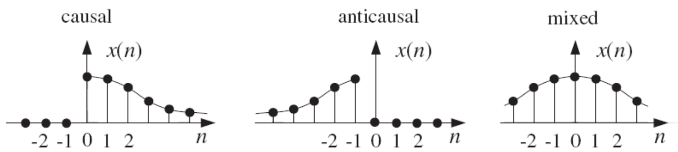


Figure 17: Causal, Anticasual and Mixed signals

LTI systems can also classified in terms of causality depending on whether $h(n)$ is casual, anticausal or mixed. A system is stable (BIBO) if **bounded inputs** ($|x(n)| \leq A$) always generate **bounded outputs** ($|y(n)| \leq B$).

A LTI system is stable $\Leftrightarrow \sum_{n=-\infty}^{\infty} (h(n)) < \infty$