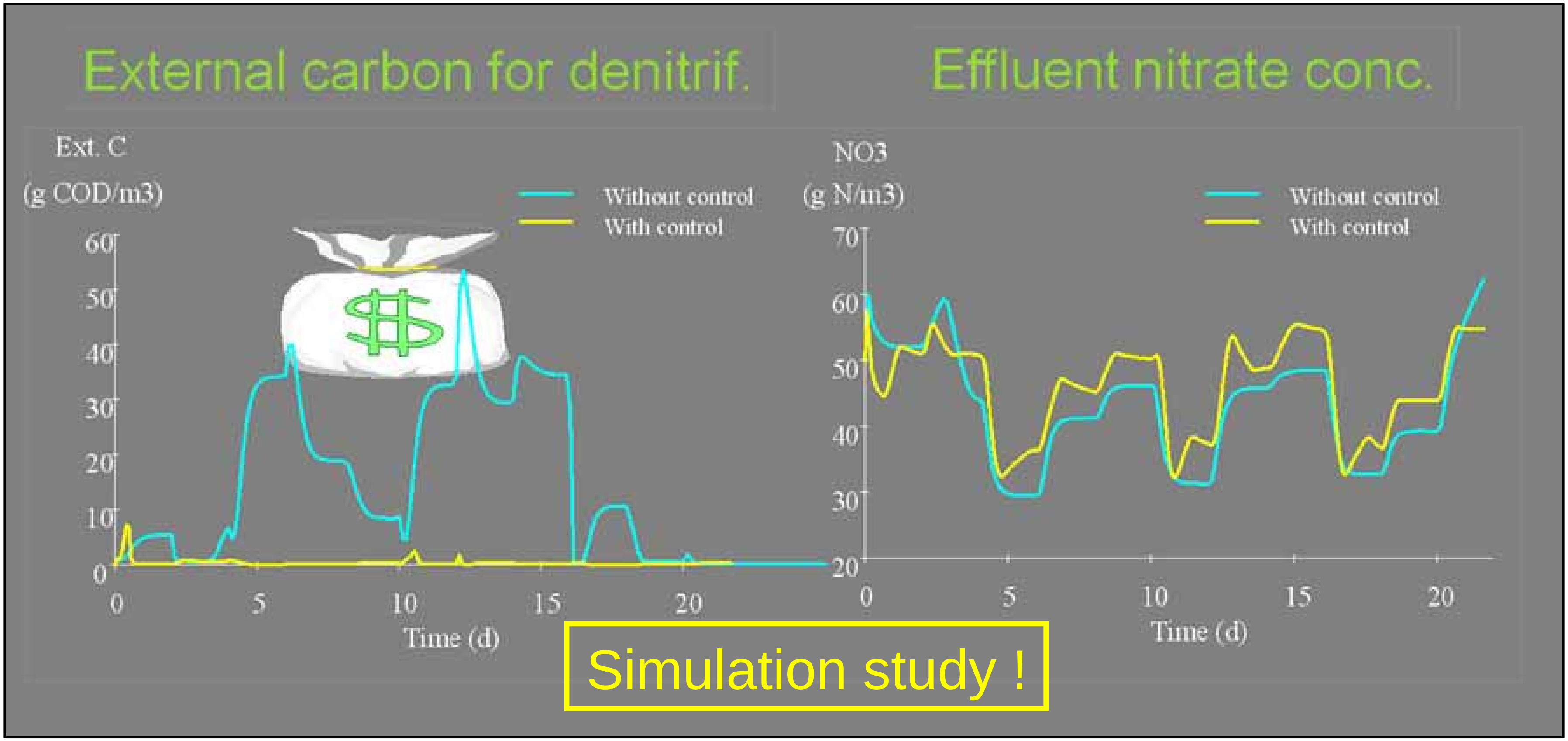
Introduction to Control Theory

Peter A. Vanrolleghem

Summer school on modelling MBR processes
July 15-17, 2008, Ghent - Belgium

Why control ?

- Example: Industrial WWTP with a N-problem
  <u>Question:</u> Implement control or not ?



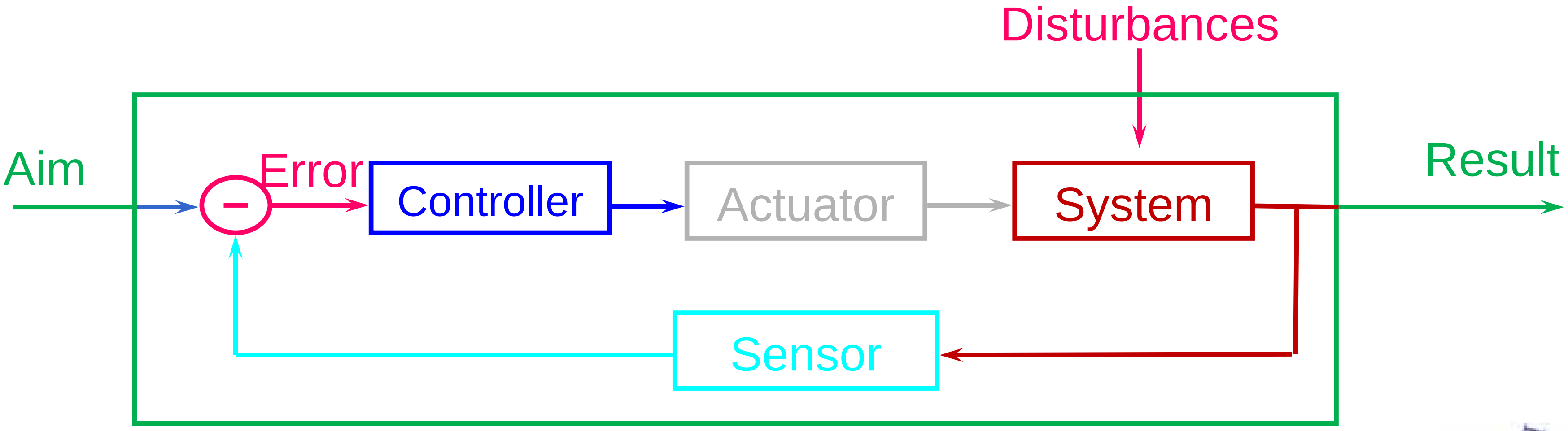
Simulation study !

Process control: Objectives

- Regulation:
  - Guarantee stability of the results
  - Suppress the effect of disturbances
  - *Disturbance rejection*
  - e.g. temperature in fridge
- Servo:
  - Optimize process operation
  - Track optimal operating conditions
  - *Setpoint tracking*
  - e.g. driving a car

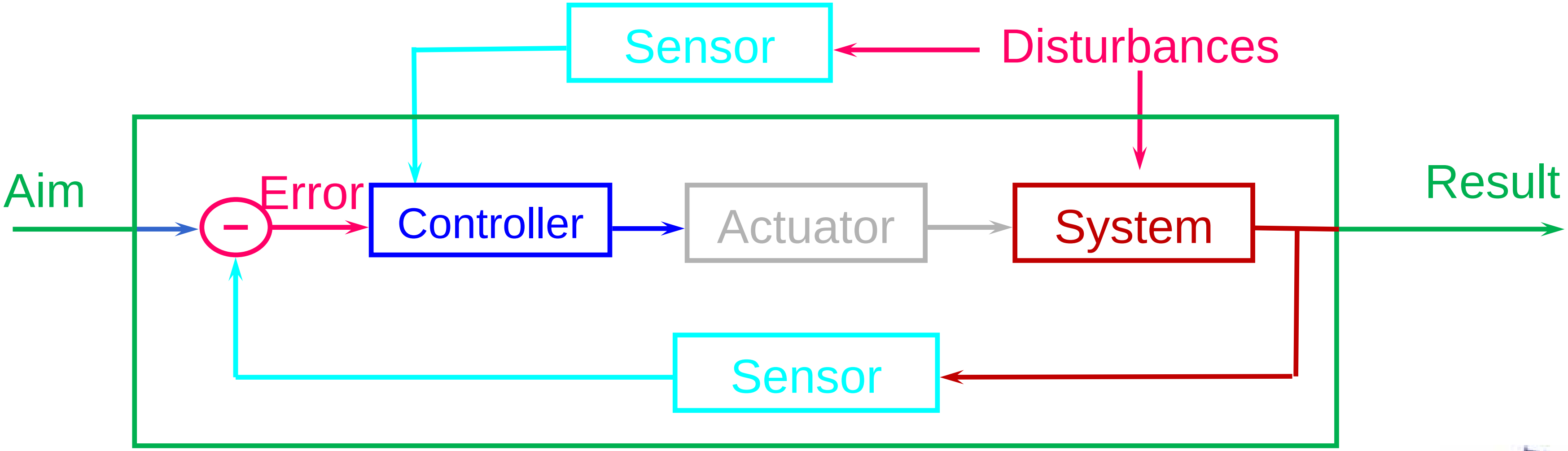
Process Control: Concepts

- *Feedback Control*
  1) Deviation of objective is determined
  2) Controller “calculates” control action
  3) Deviation is eliminated



Process control: Concepts

- *Feedforward control*
    1) Disturbance is determined
    2) Potential future deviation of aim is “calculated”
    3) Control action compensates for predicted deviations



Process control: Control structure

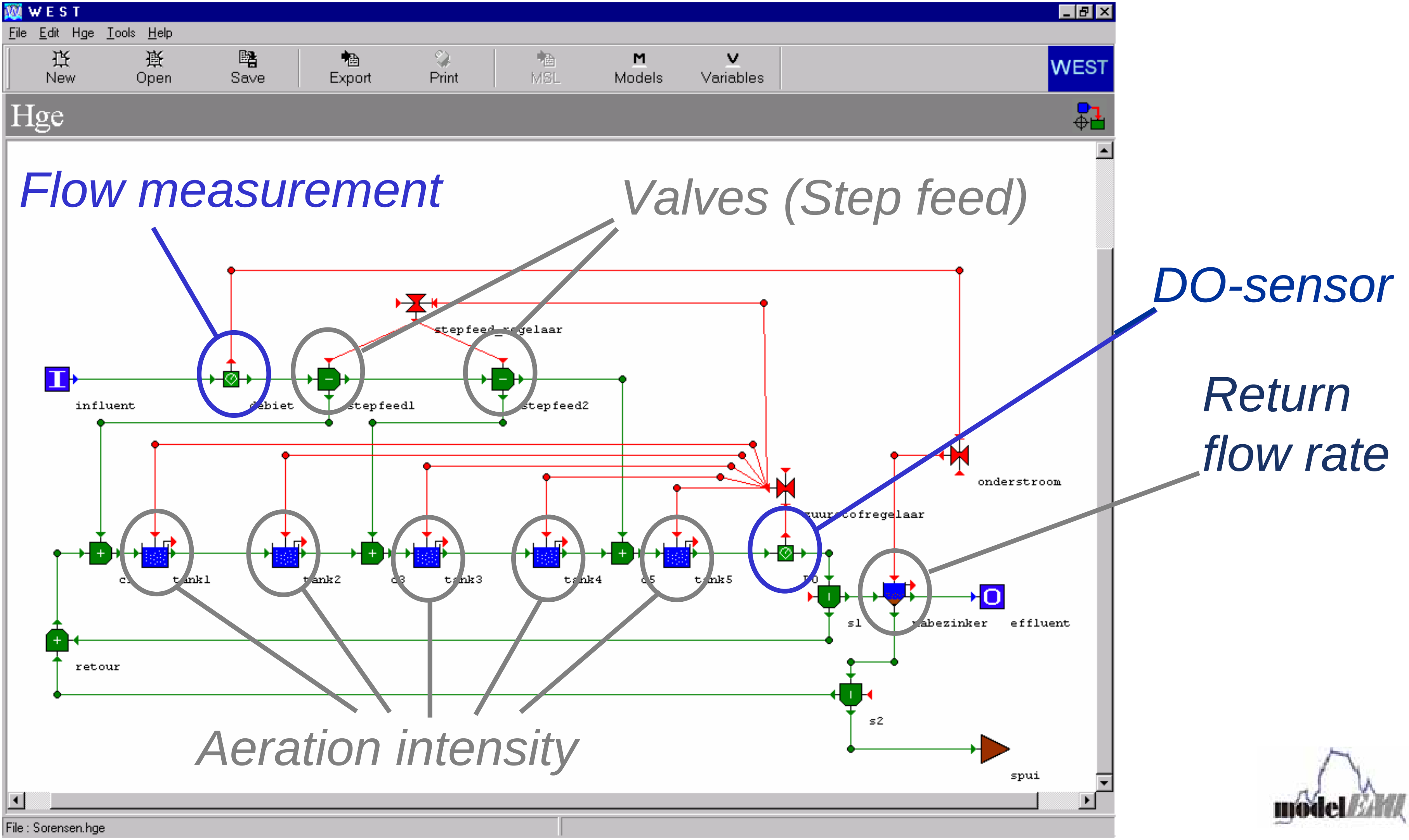
- Inputs of a controller (measurements)
  - flow rates (gas - liquid)
  - levels
  - pH
  - temperature
  - oxygen concentration
  - NH_4-concentration
  - ...

- Outputs of a controller (manipulations)
  - influent flow rate
  - return flow rate
  - aeration intensity
  - internal recirculation
  - acid/base addition
  - dosage of nutrients
  - ...

Control structure

SISO : Single Input / Single Output controller
MIMO : Multiple Input / Multiple Output controller

Process control: Control structure



Flow measurement

Valves (Step feed)

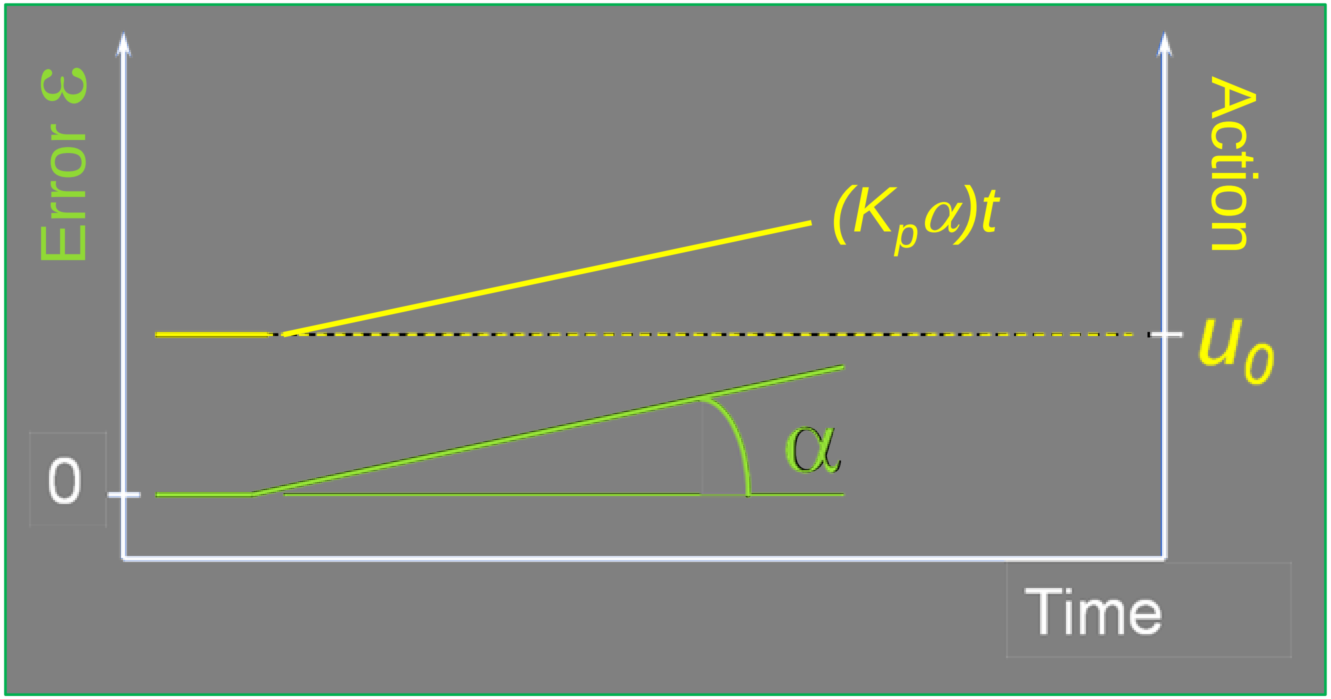
DO-sensor

Return flow rate

Aeration intensity

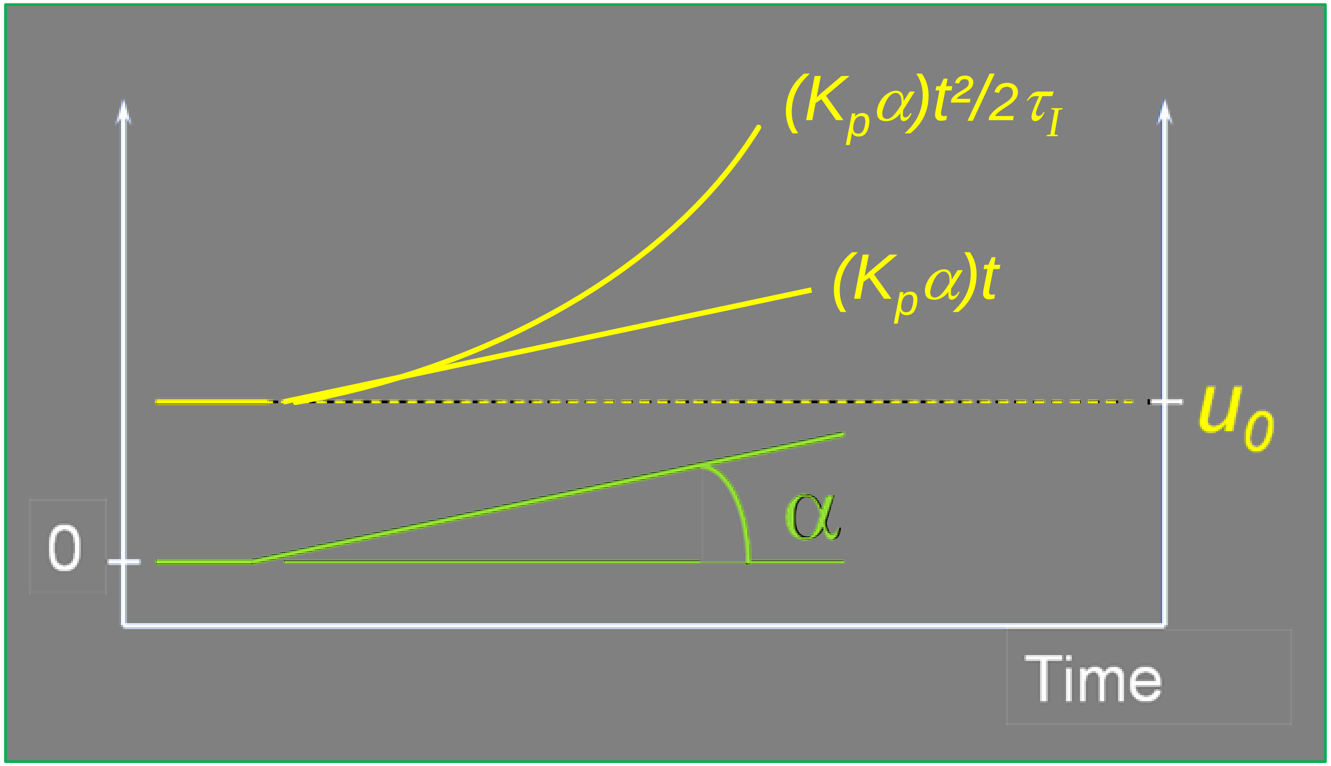
Control algorithms: PID-control

$$u = u_0 + K_p\left[\varepsilon(t)\right]$$



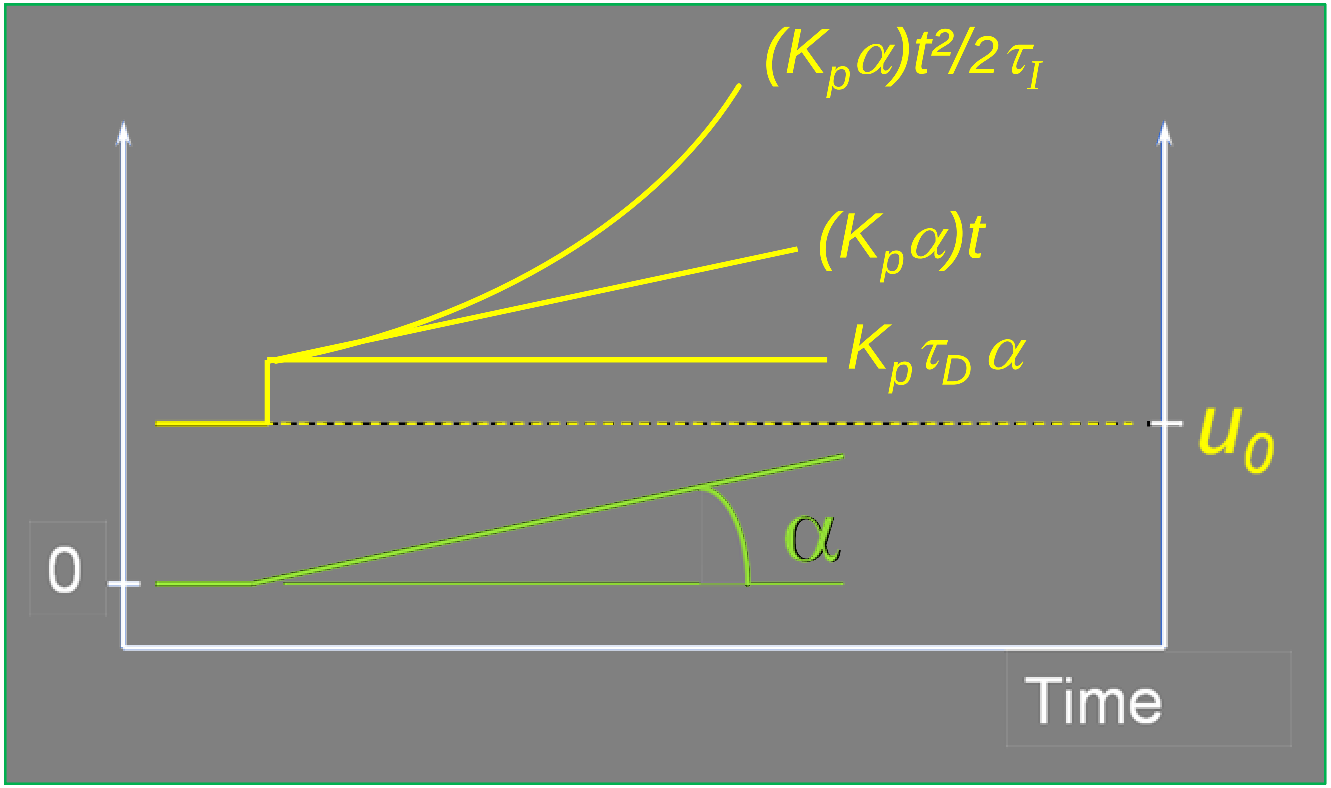
Control algorithms: PID-control

$$u = u_0 + K_p\left[\varepsilon(t) + \frac{1}{\tau_I}\int_0^t \varepsilon(t)\,dt\right]$$

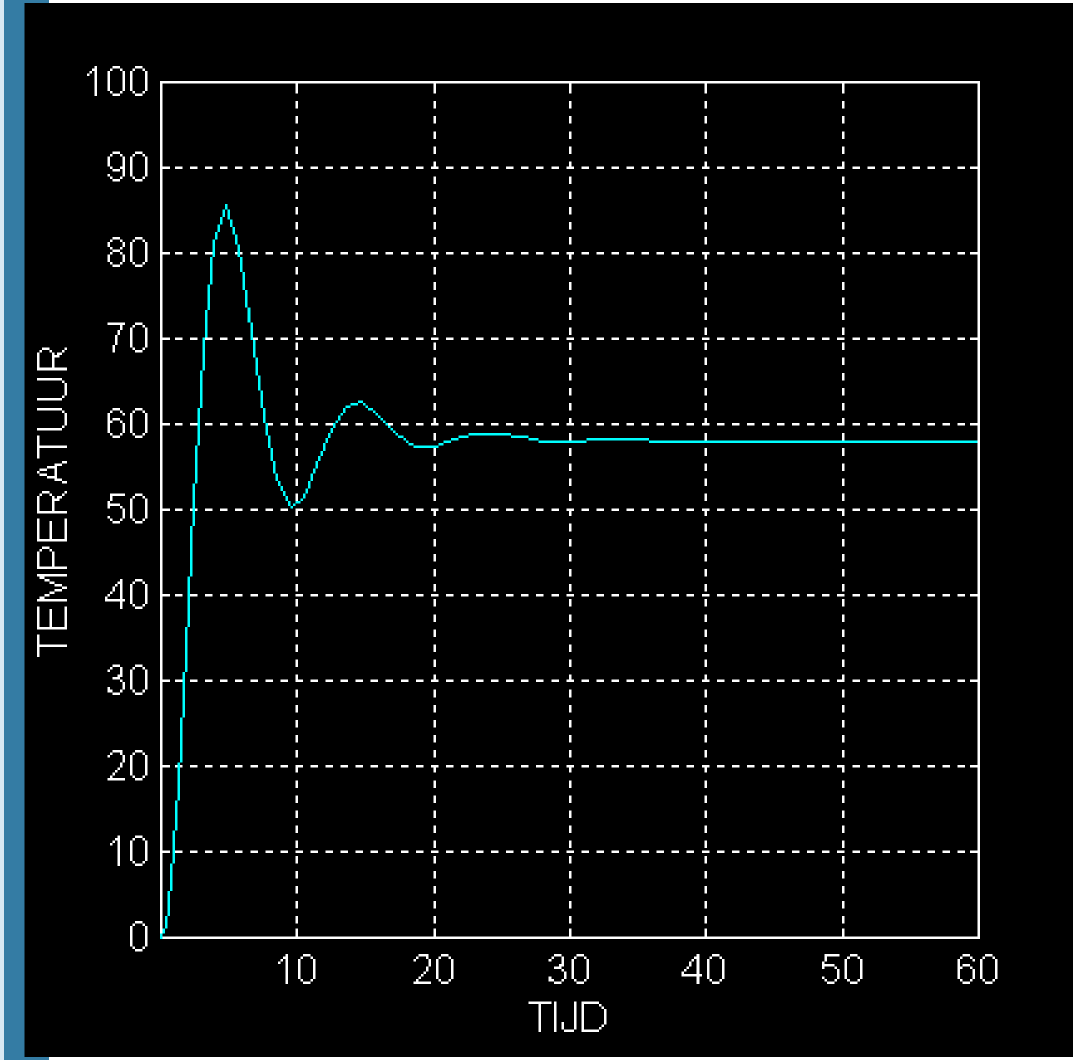


Control algorithms: PID-control

$$u = u_0 + K_p\left[\varepsilon(t) + \frac{1}{\tau_I}\int_0^t \varepsilon(t)dt + \tau_D\frac{d\varepsilon(t)}{dt}\right]$$



Tuning of controllers



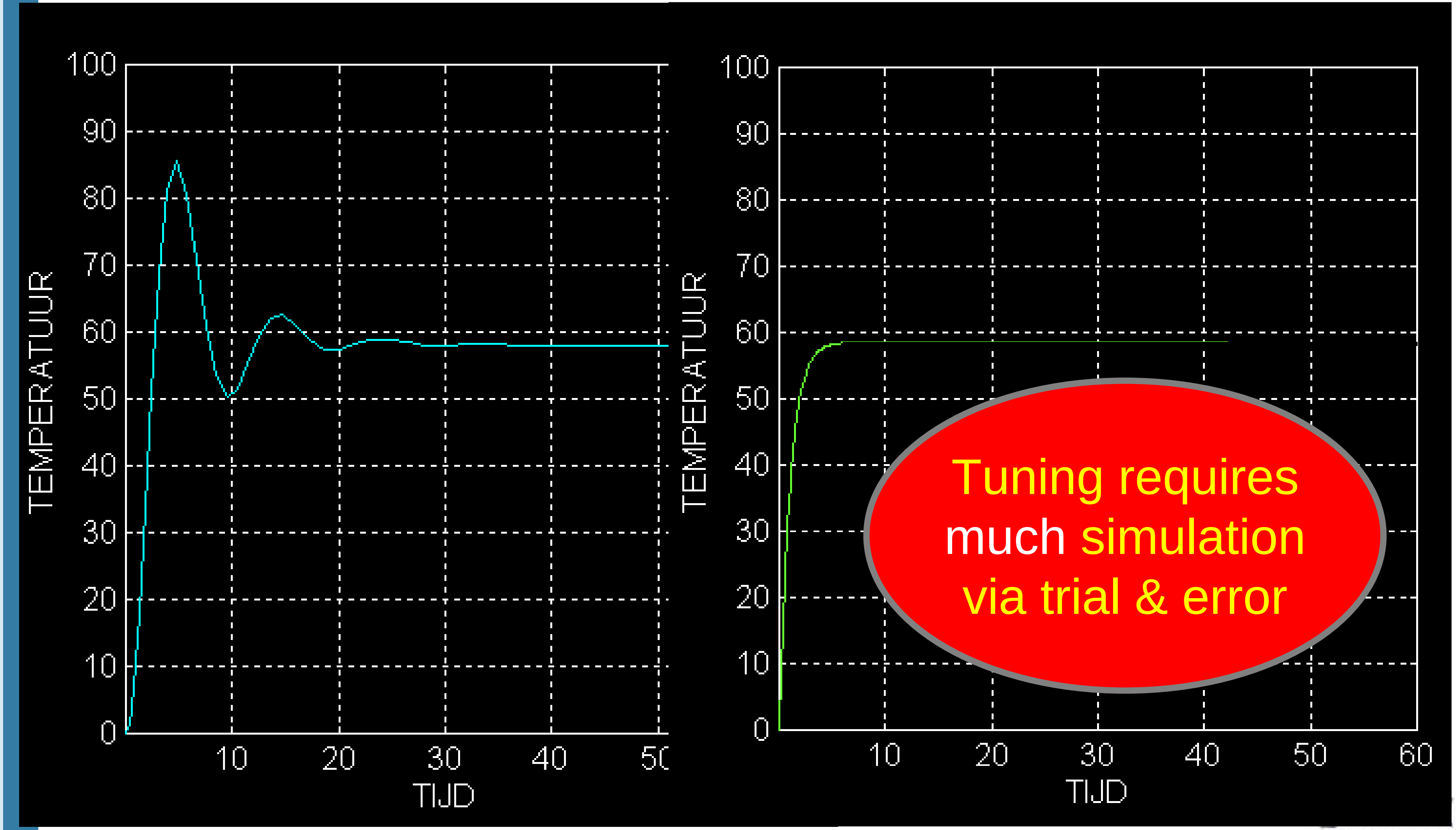
Simulation shows a badly tuned PID-controller

Large overshoot

Tuning of the controller (K_p, t_I en t_p) leads to good control

(fast & without overshoot)

Tuning of controllers



Tuning requires much simulation via trial & error

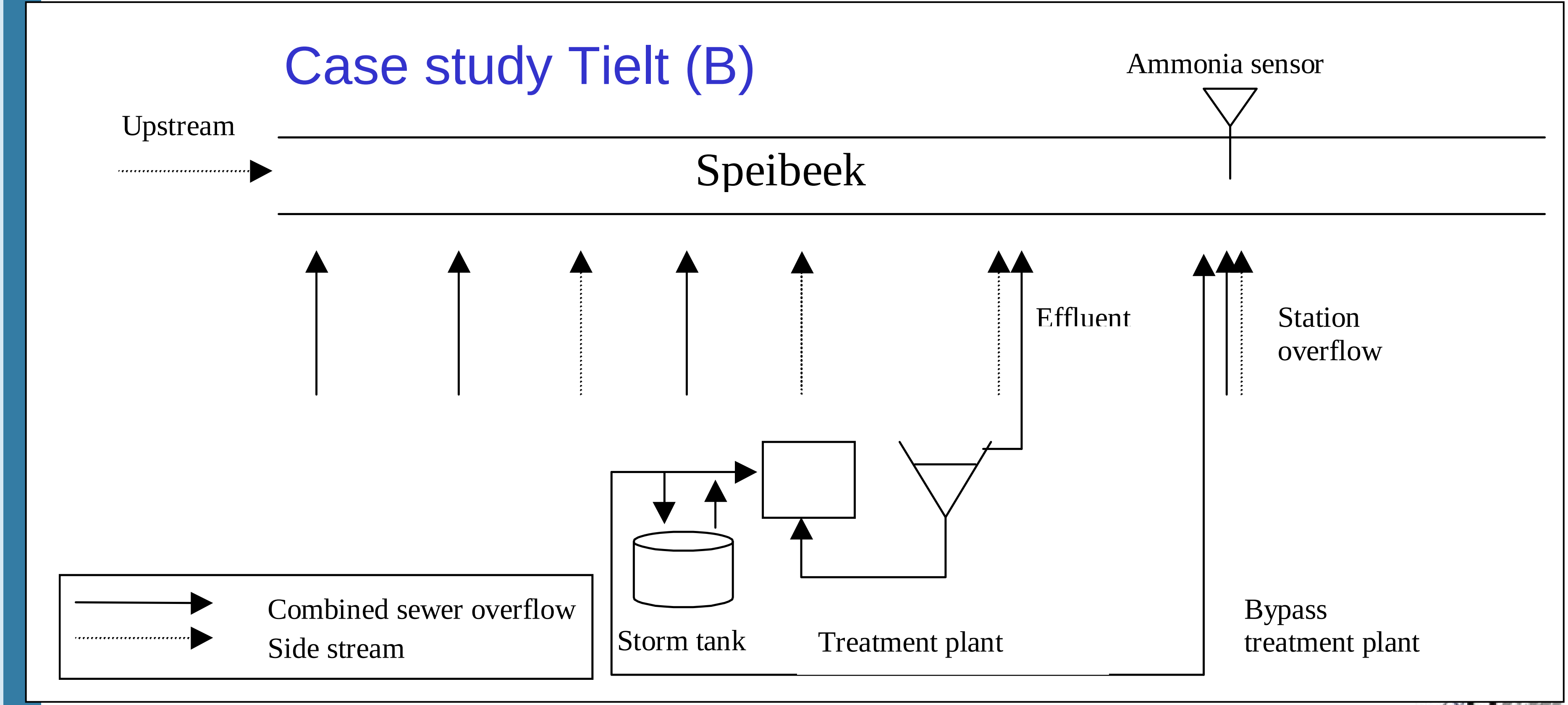
Feedforward controllers

- Aeration control:
  - *Aeration intensity:* $K_L a = \alpha F_{in} + \beta$
  - *Mass balance* for O_2: $\frac{dS_O}{dt} = K_L a \left(S_O^{sat} - S_O \right) - \frac{Q}{V} S_O - r$
  - With measurement of respiration rate *r* and flow Q we calculate K_La such that O_2 reaches its setpoint S^*_O :

$$F_{in} = \frac{1}{\alpha} \left(\frac{\frac{Q}{V} S_O^* + r}{S_O^{sat} - S_O^*} - \beta \right)$$

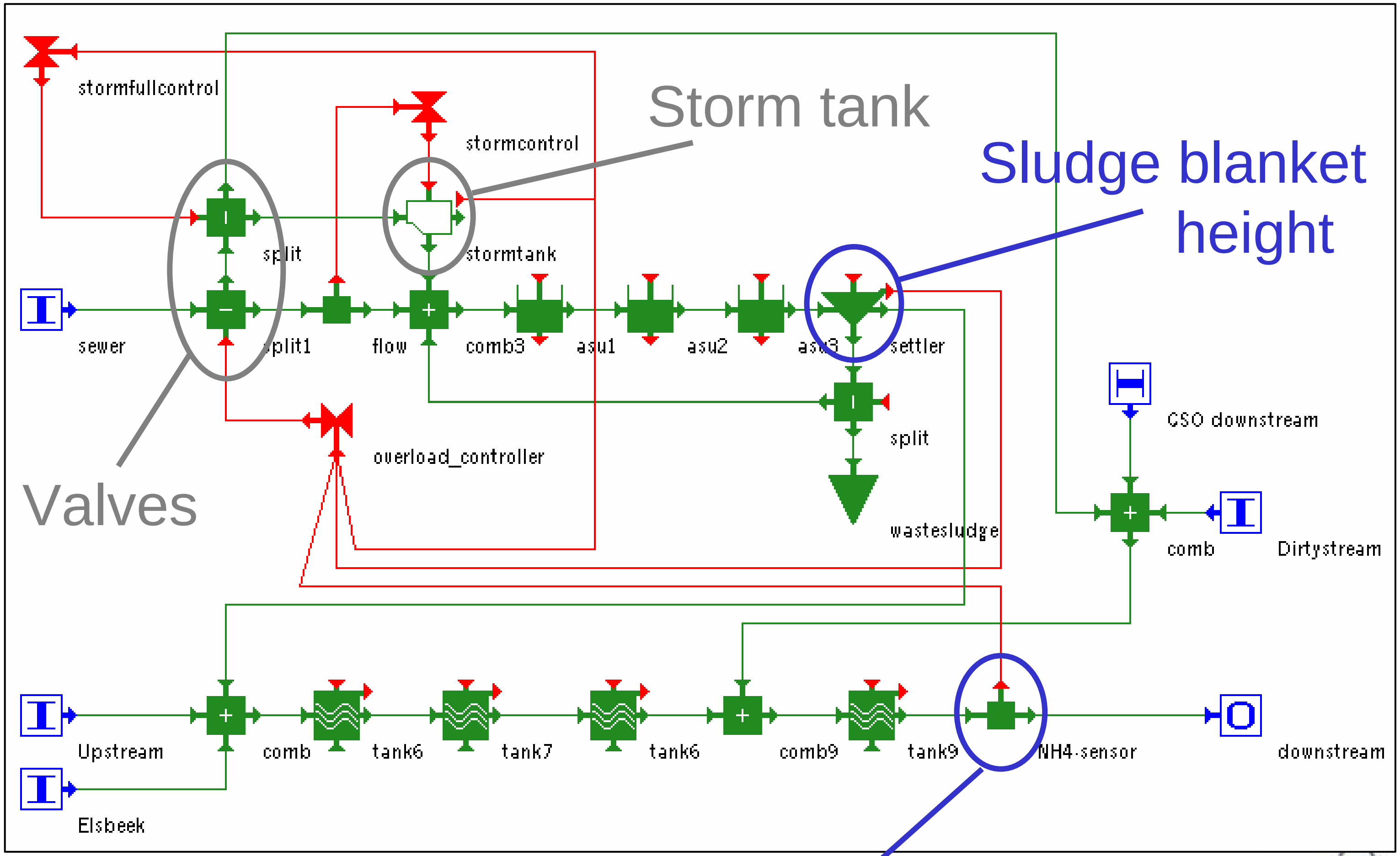
Controller using prediction of river water quality

cfr. new EU Water Framework Directive !



Case study Tielt (B)

Controller using prediction of river water quality



Storm tank

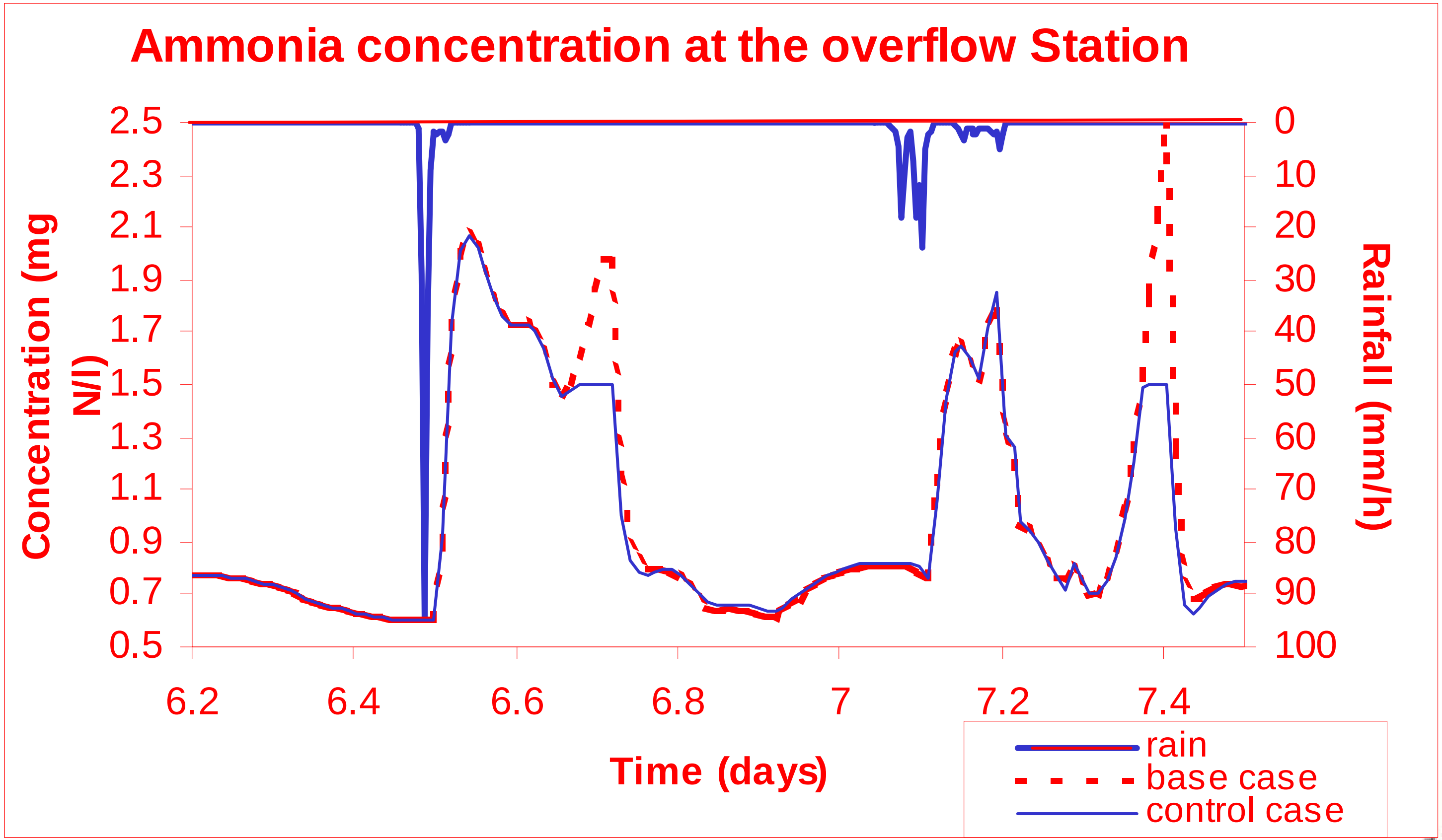
Sludge blanket height

Valves

Predictor of river water quality

Controller using prediction of river water quality

Simulation results...

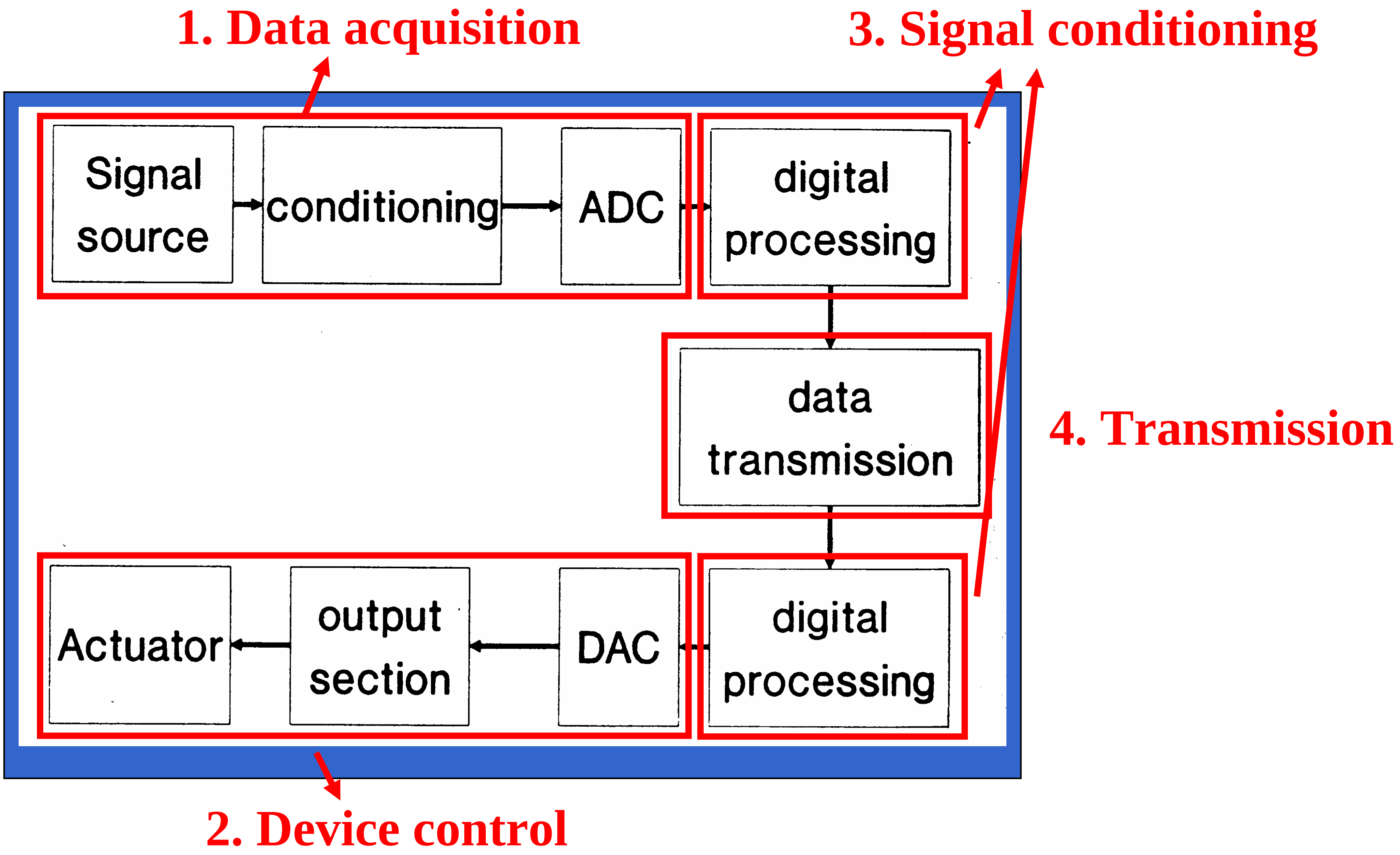


Introduction to data-acquisition and control

Ingmar Nopens

Summer school on modelling MBR processes
July 15-17, 2008, Ghent - Belgium

The Key Diagram



Measurement Principles

→ all signals = electrical signals (small amplitude)

4 groups of signals	*examples*
- low voltage (< 1 µV)	pH, ORP
- low current (< 1 µA)	DO, conductivity
- low resistance (< 100 mΩ)	T
- high resistance (> 1 GΩ)	

Low level signals: $V_S = 1\mu A \times R_S$

e.g. $V_S = 0.02 \text{ pH} \times 0.059 \text{ V/pH} = 1.18 \text{ mV}$

$\ll 1\ \mu A \times 250\ M\Omega = 250\ V$

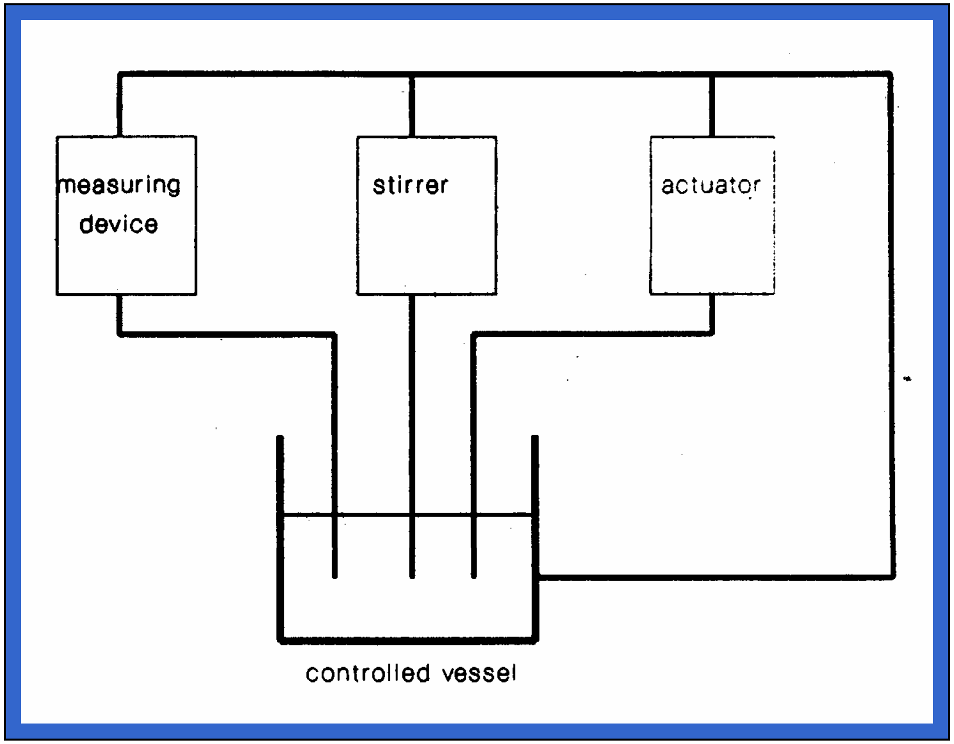
Signal Conditioning

Remark: *galvanic isolation*

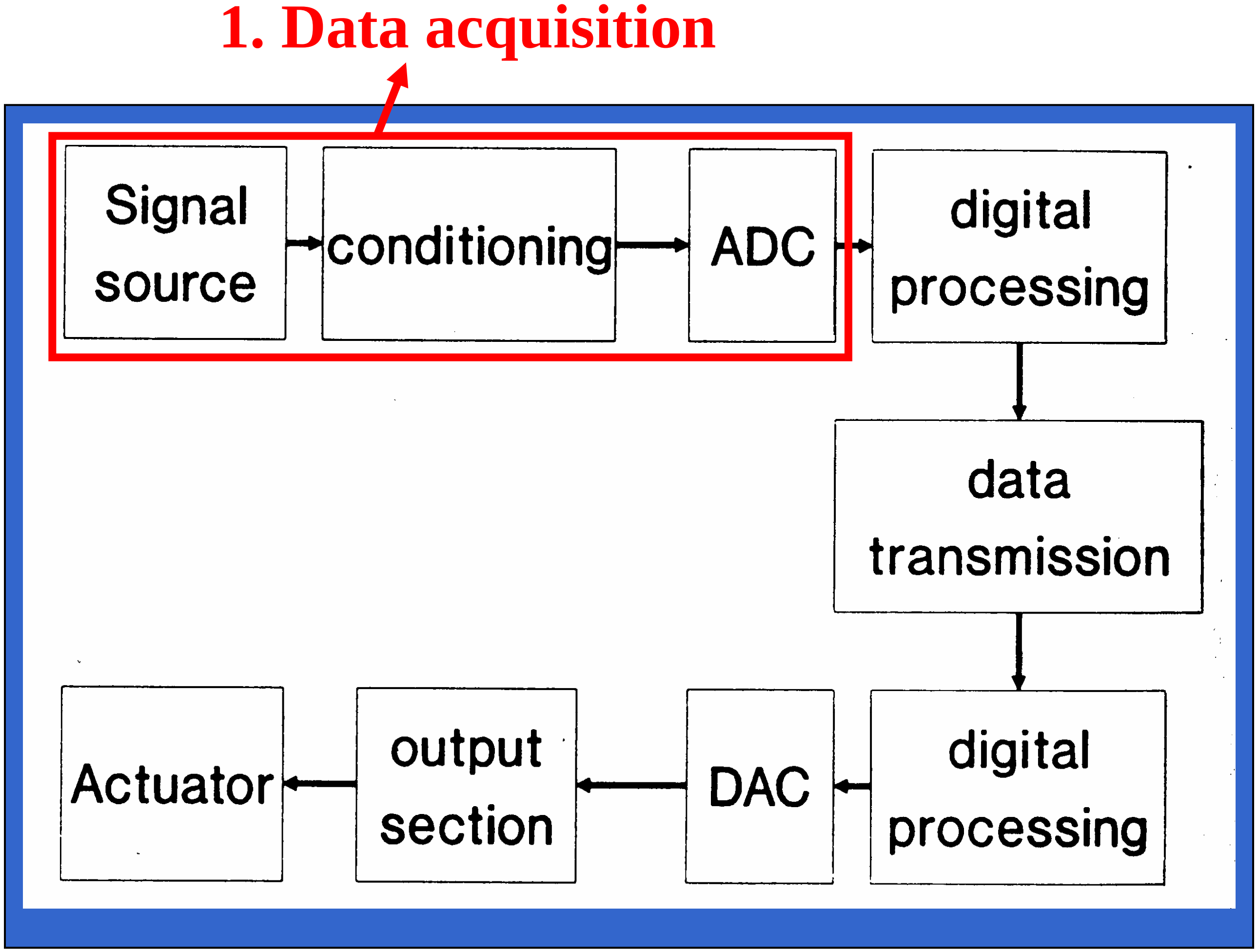
Ground loops!

→ common ground

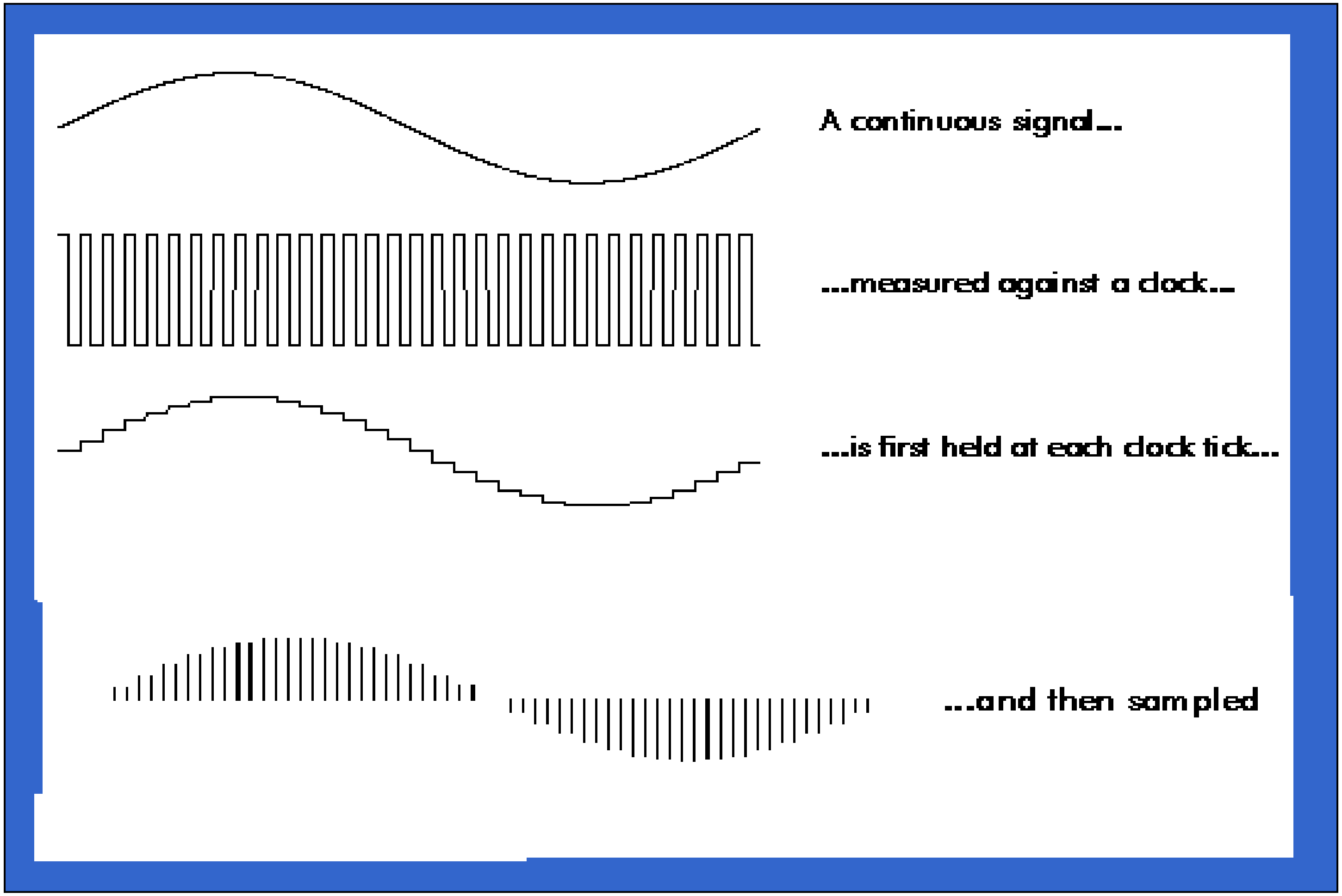
→ galvanic isolation



The Key Diagram

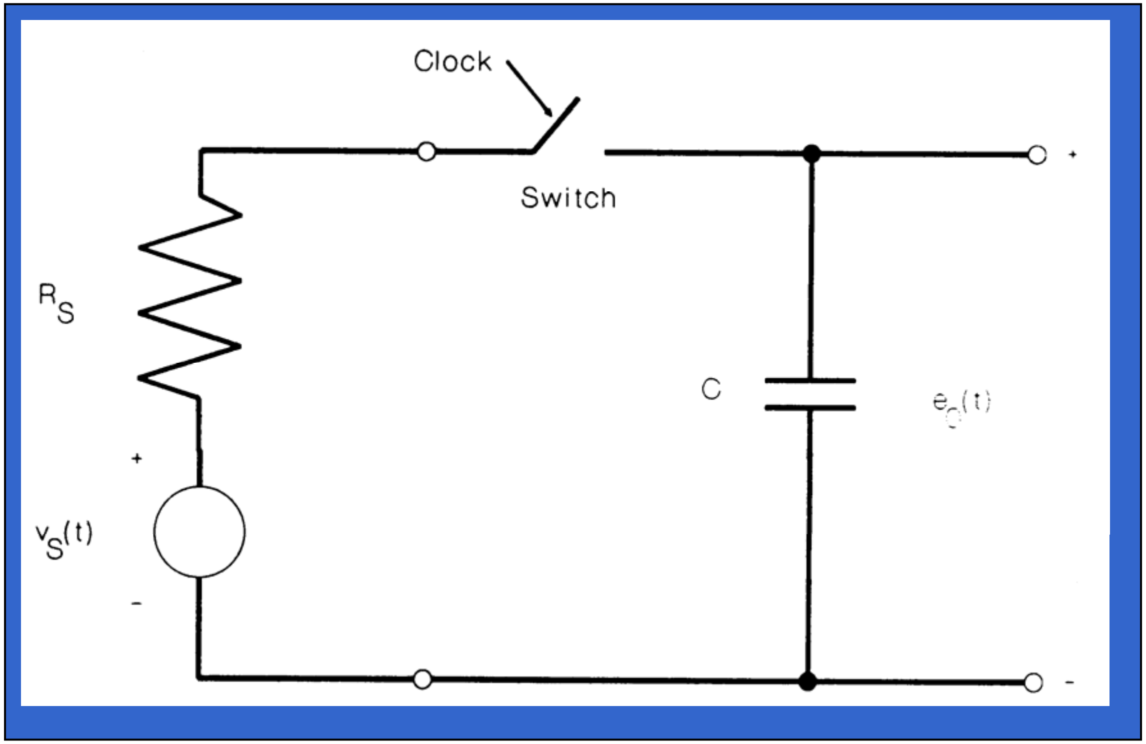


Data conversion



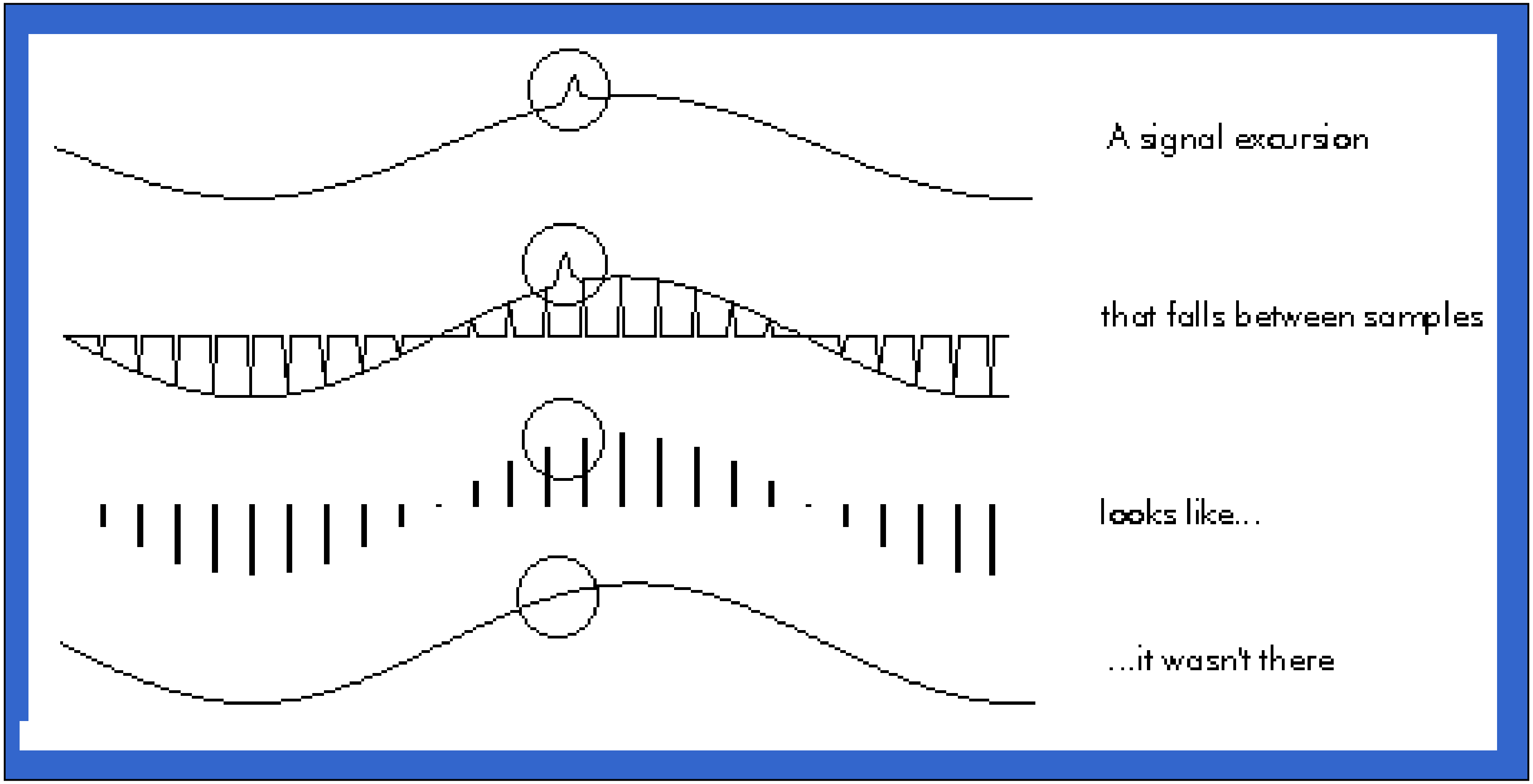
Data conversion

→ Sample and hold circuit



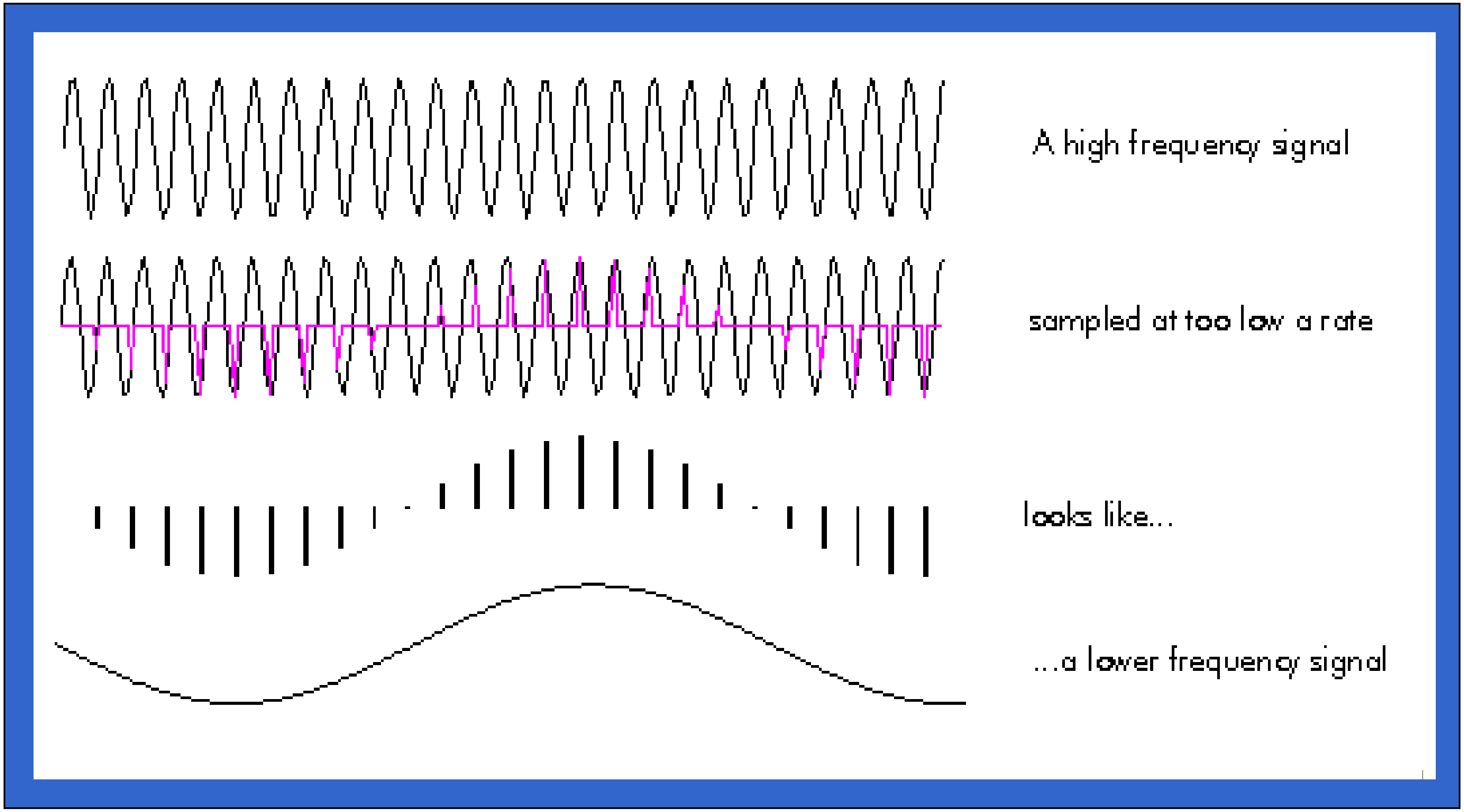
Data conversion

→ Sampling frequency > highest signal frequency (Nyquist)



Data conversion

→ aliasing problem



Data conversion (ADC)

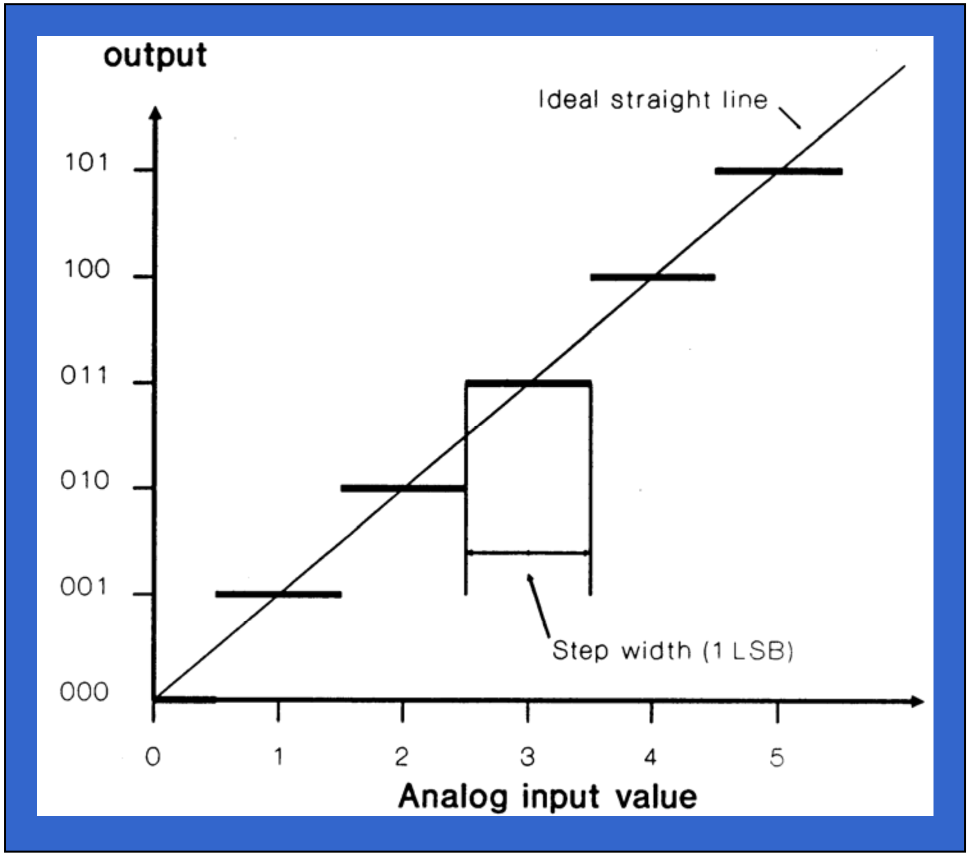
- speed [meas./sec]
- resolution [bits]

e.g. *8-bit converter*: 256 discrete states
→ accuracy: 1/256 = 0.4%

10-bit converter:
→ accuracy: 1/1024 = 0.1%

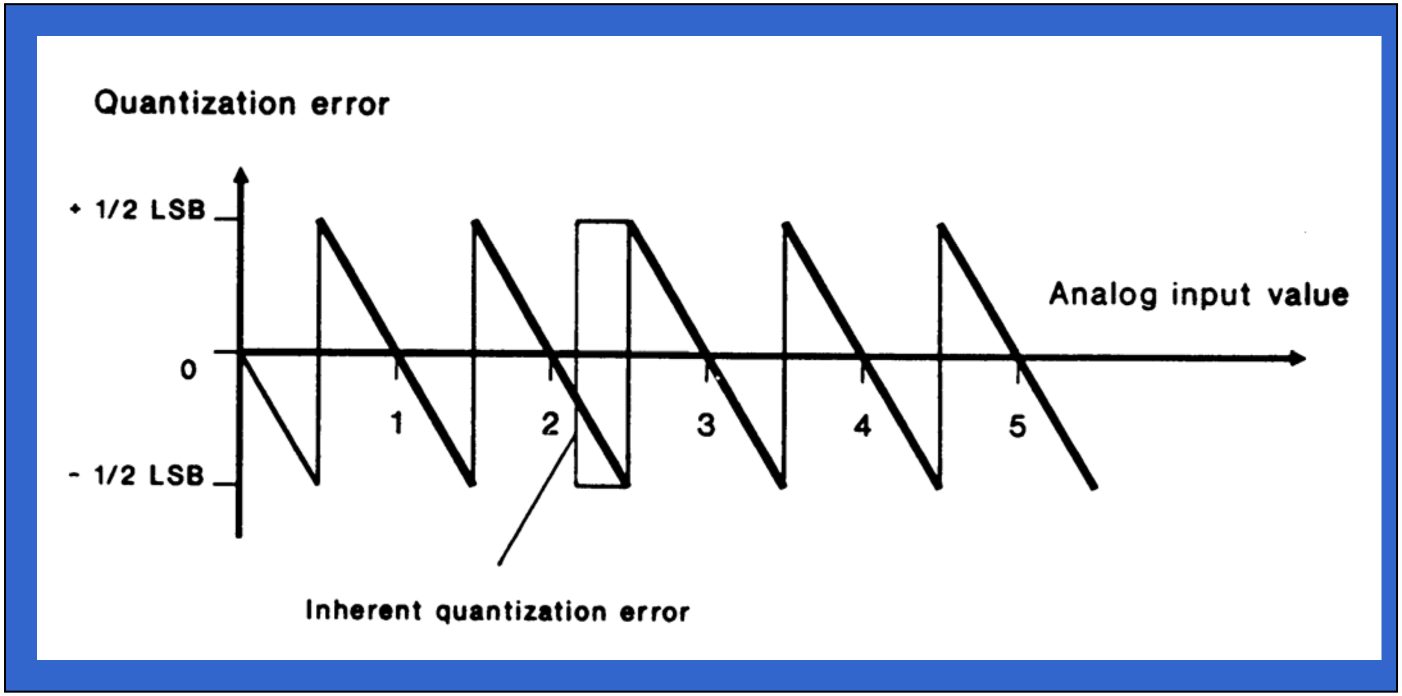
resolution=2^{bits}

- Least Significant Bit (LSB)



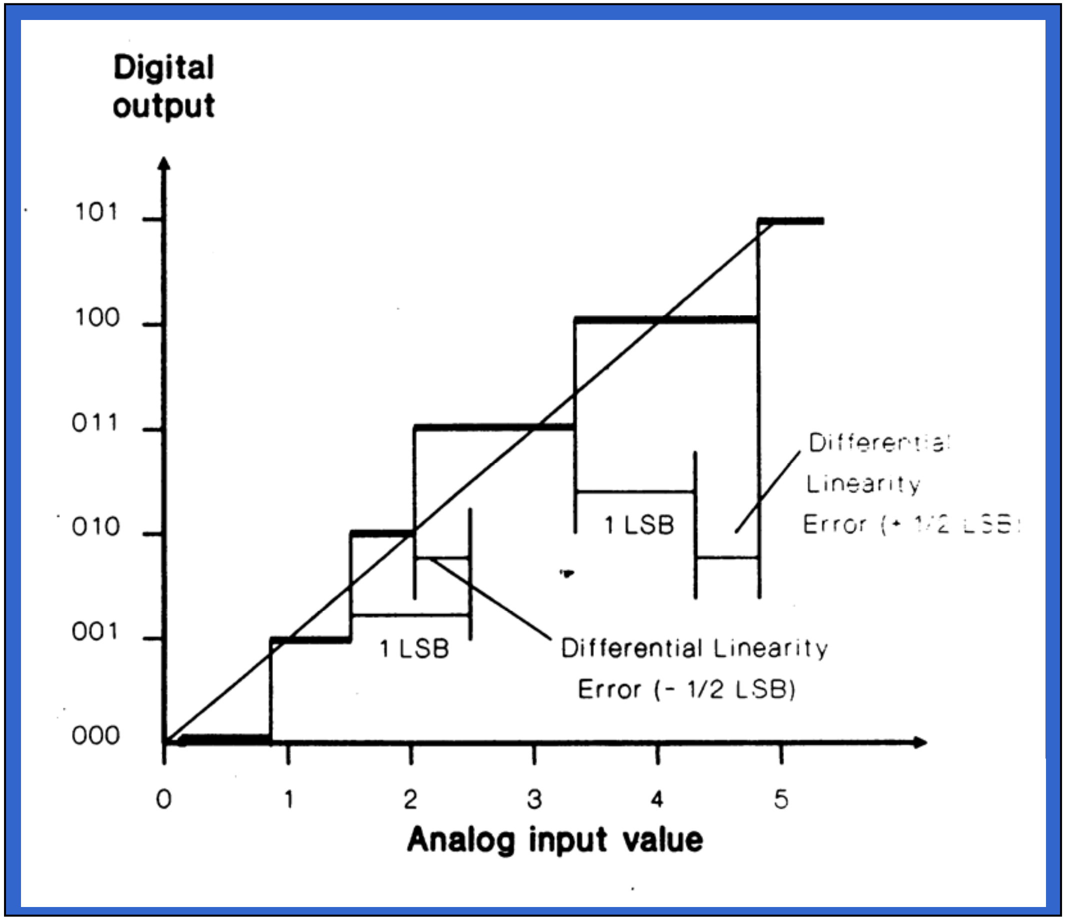
Data conversion (ADC)

Quantization error inherent to ADC = 1/2 LSB



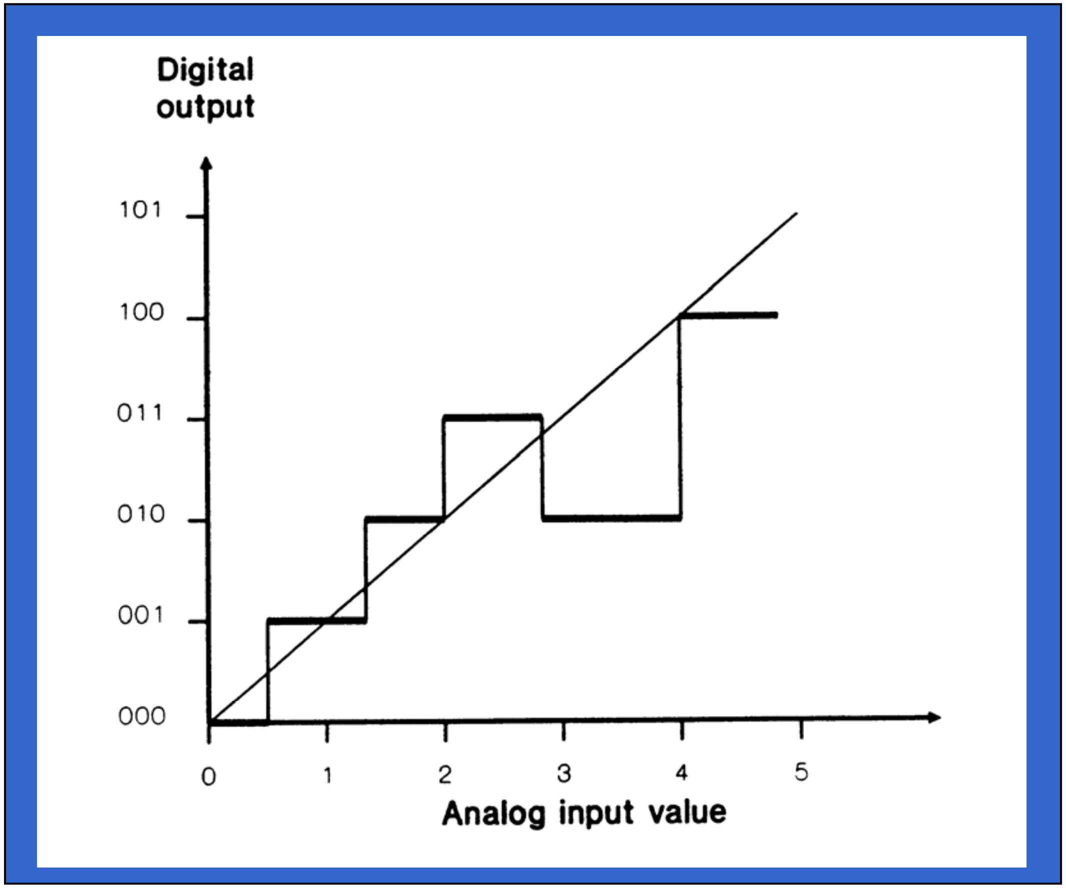
Data conversion (ADC)

Error of non-linearity



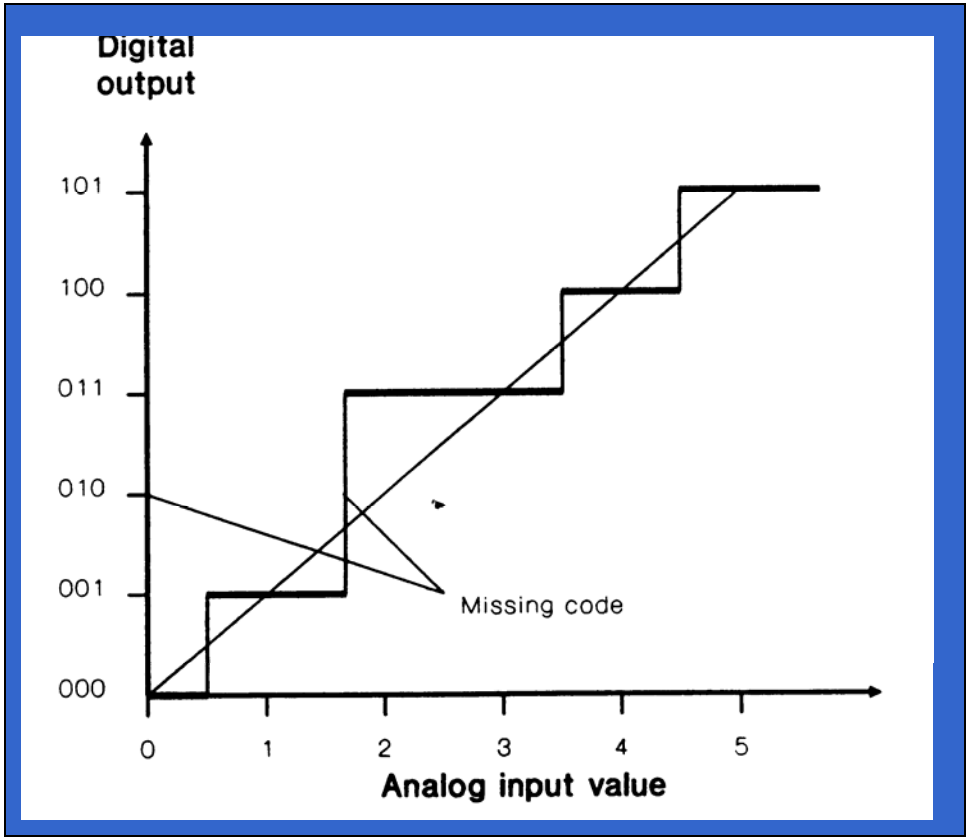
Data conversion (ADC)

Error of non-monotonicity

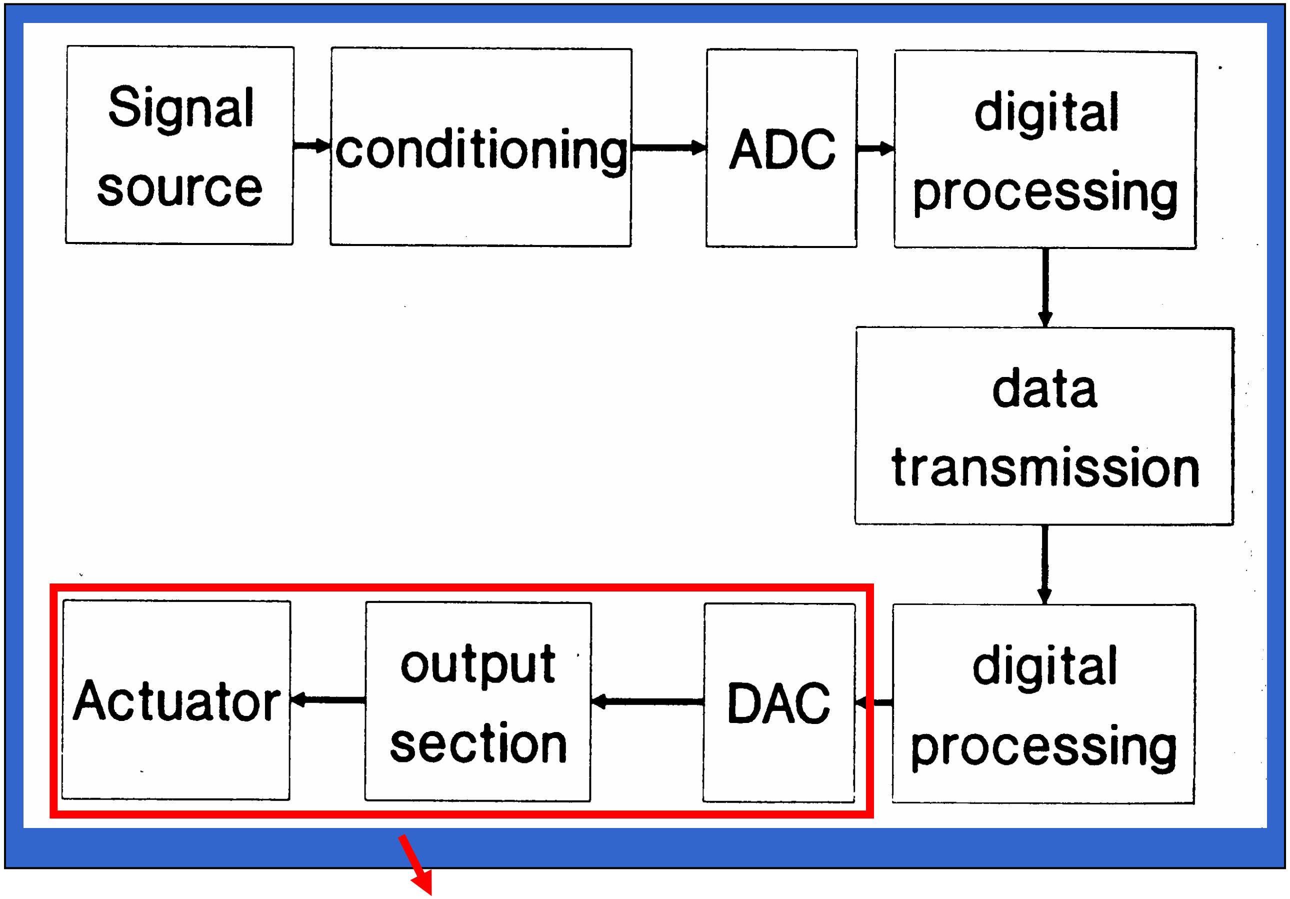


Data conversion (ADC)

Error of missing codes



The Key Diagram



Signal source

conditioning

ADC

digital processing

data transmission

digital processing

DAC

output section

Actuator

2. Device control

Data conversion (DAC)

- X = k.A.B with: X = analog signal

k = constant

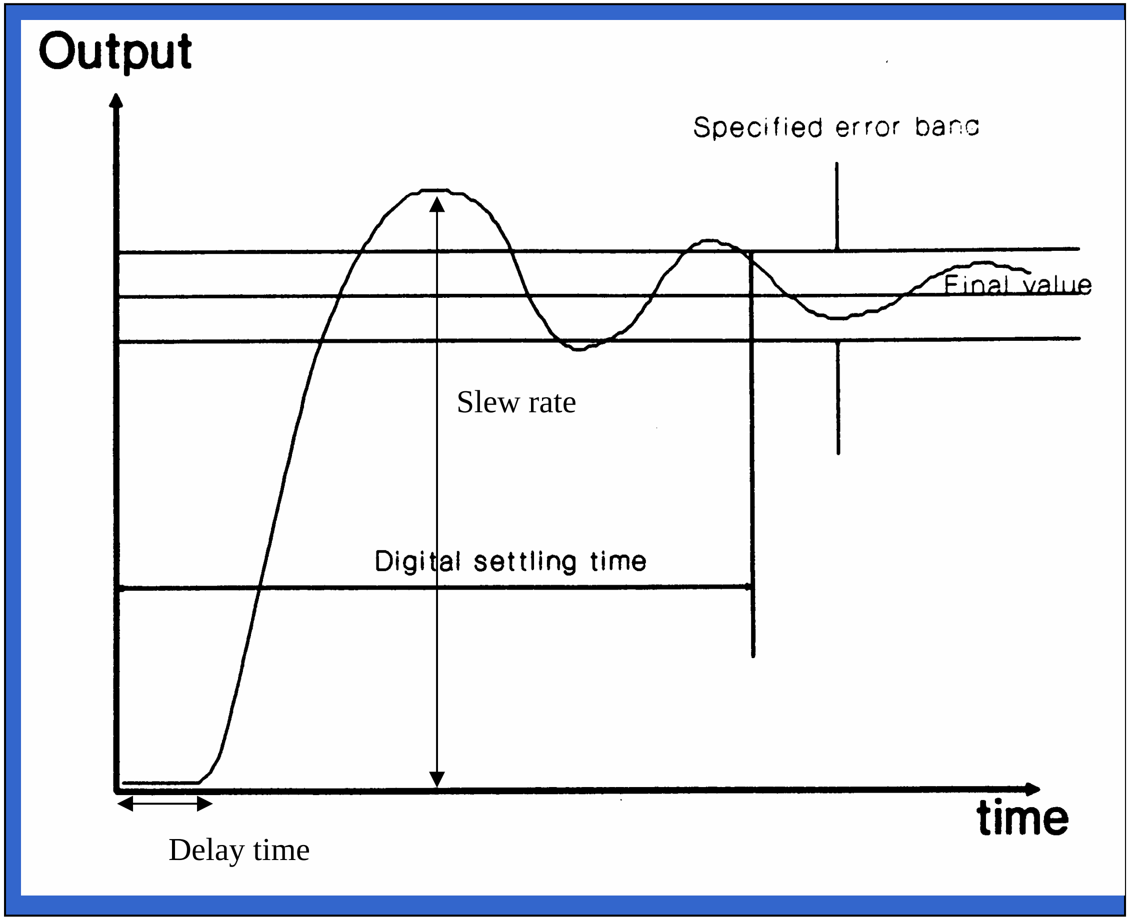
A = analog reference voltage or current

B = binary signal

- DAC characteristics → step response

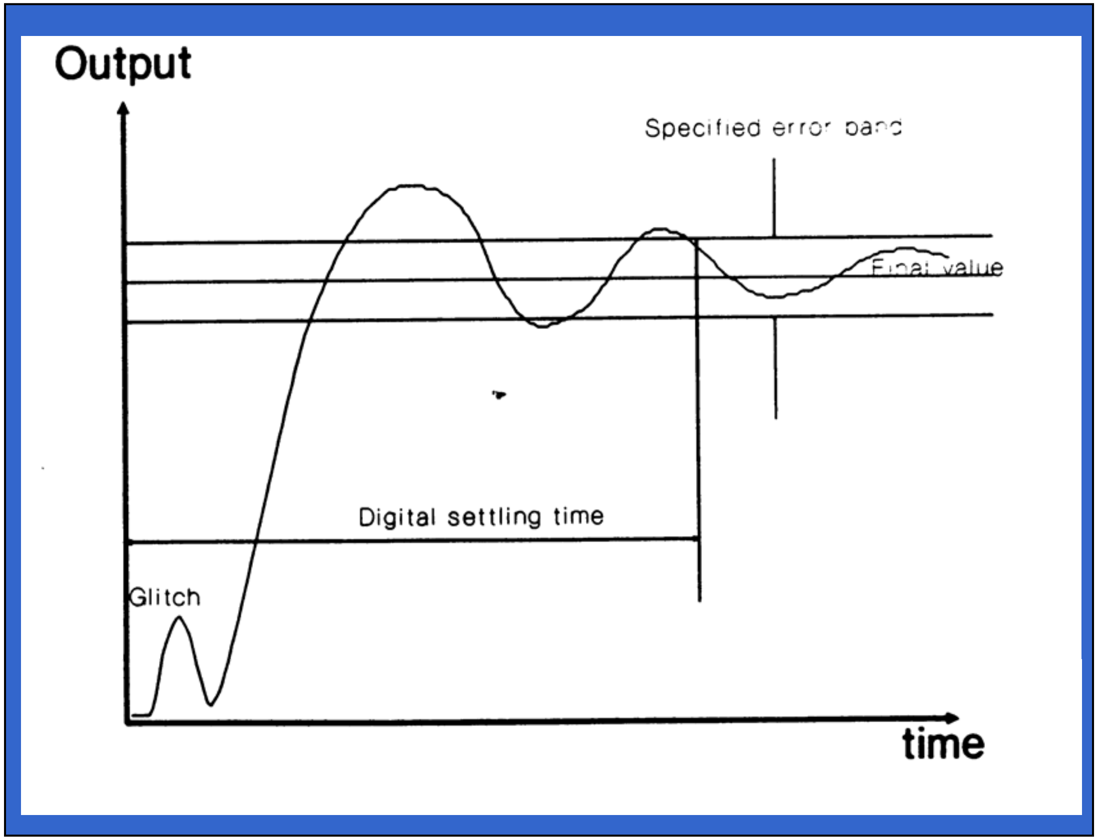
Data conversion (DAC)

Step response



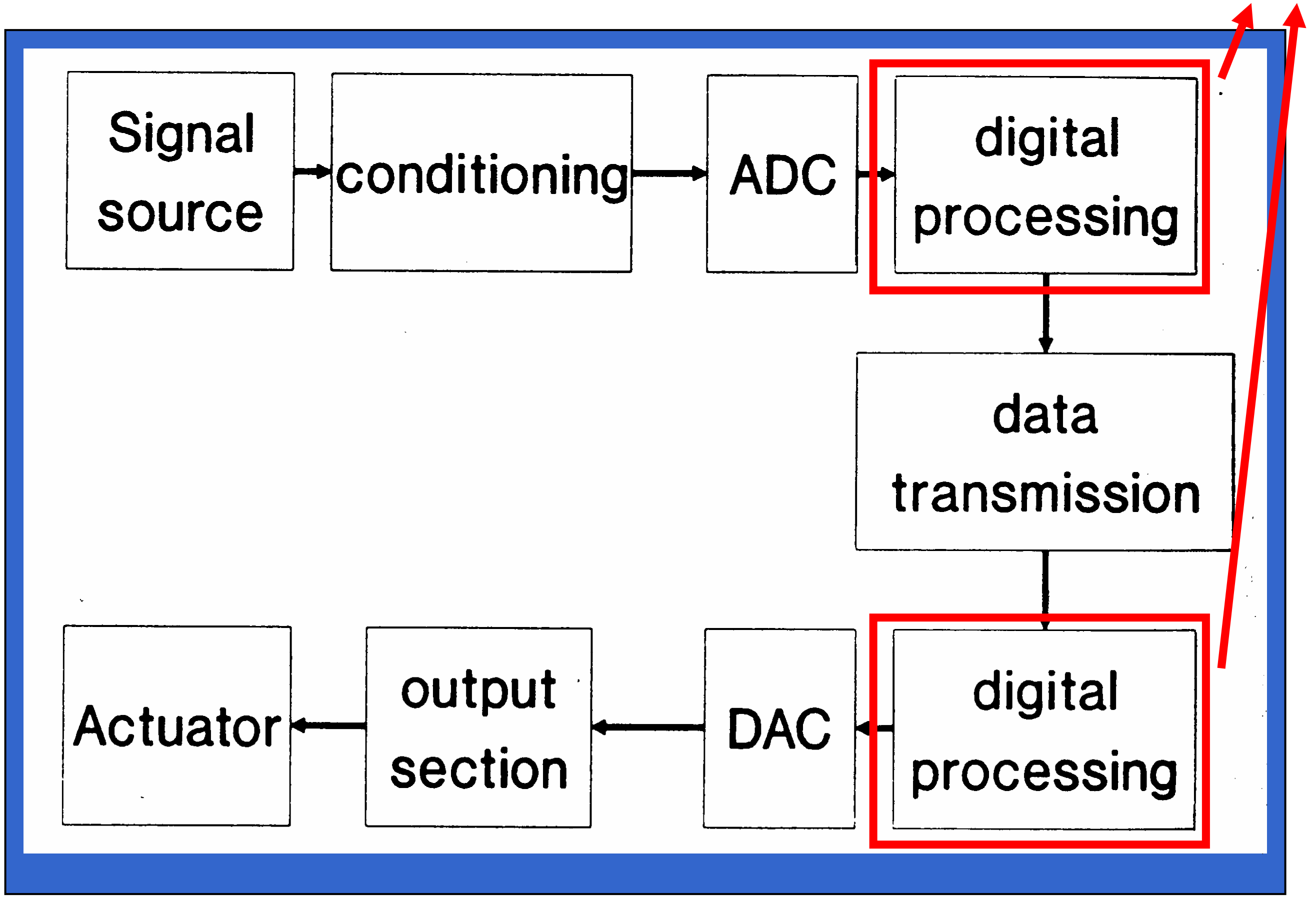
Data conversion (DAC)

Glitch → transient behaviour (glitch area)



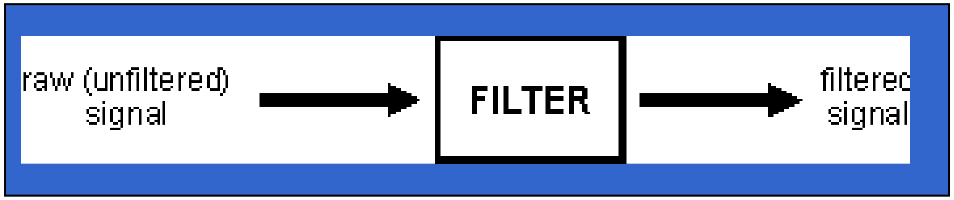
The Key Diagram

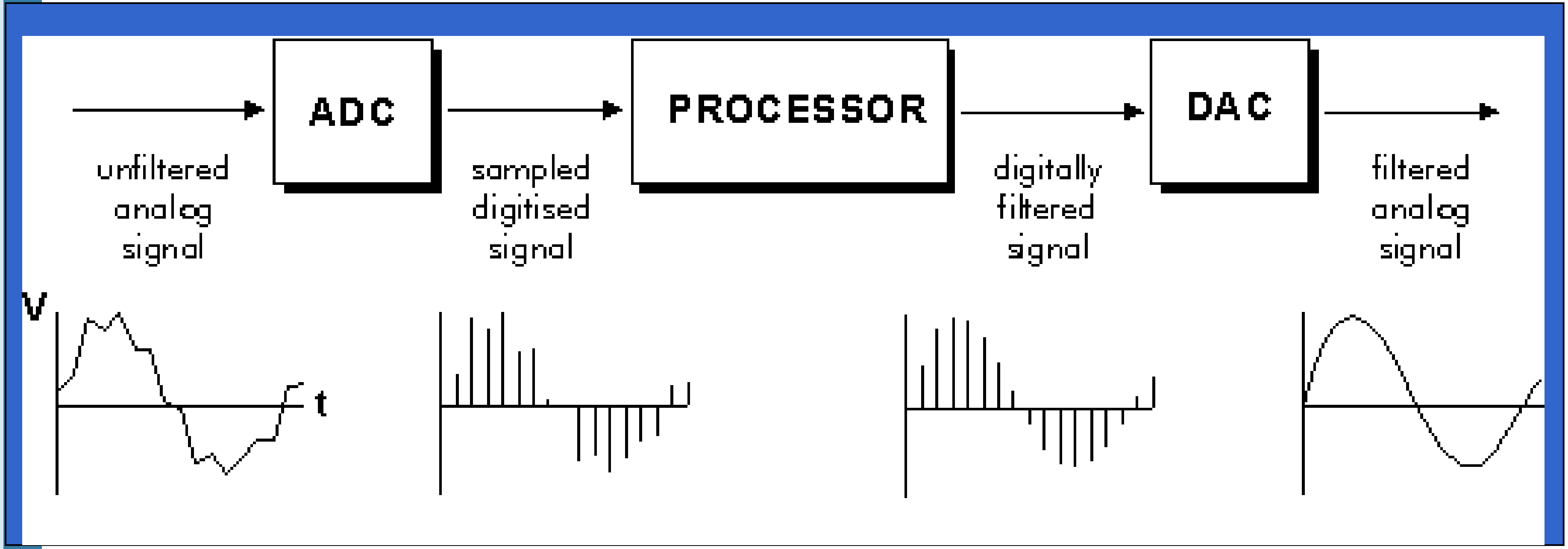
3. Signal conditioning



Filtering

Analog and digital filters

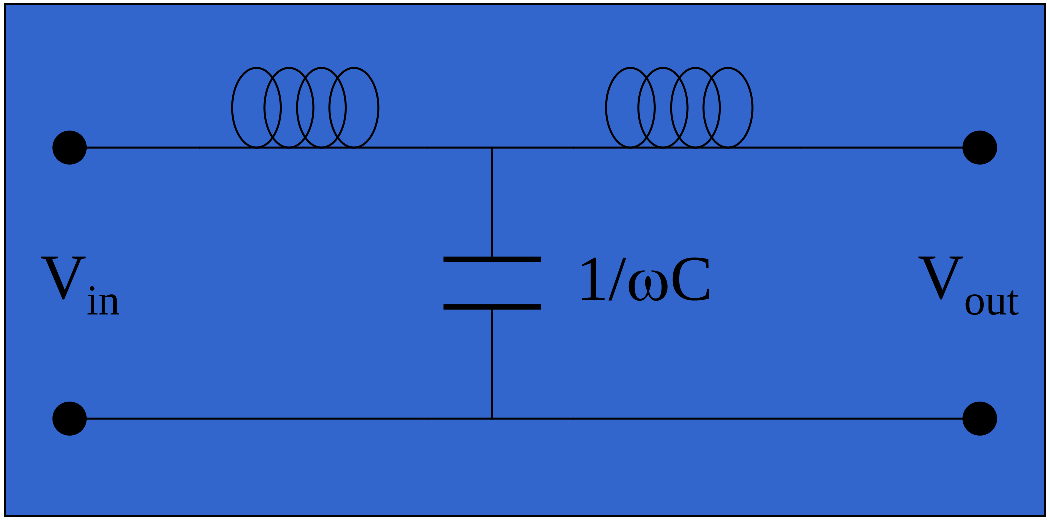




Analog Filtering

Analog filters

e.g. low-pass filter



Digital Processing

Digital filters

- history *'60* → *DSP (Digital Signal Processor): embedded controllers; focussed on specific functions*

  → *PC: series of general functions*

- advantages digital filters
  - *programmable*
  - *simple and compact*
  - *stabile (no drift)*
  - *low signal frequencies*
  - *adaptive digital filters*
- disadvantages digital filters
  - *aliasing*

Digital Processing

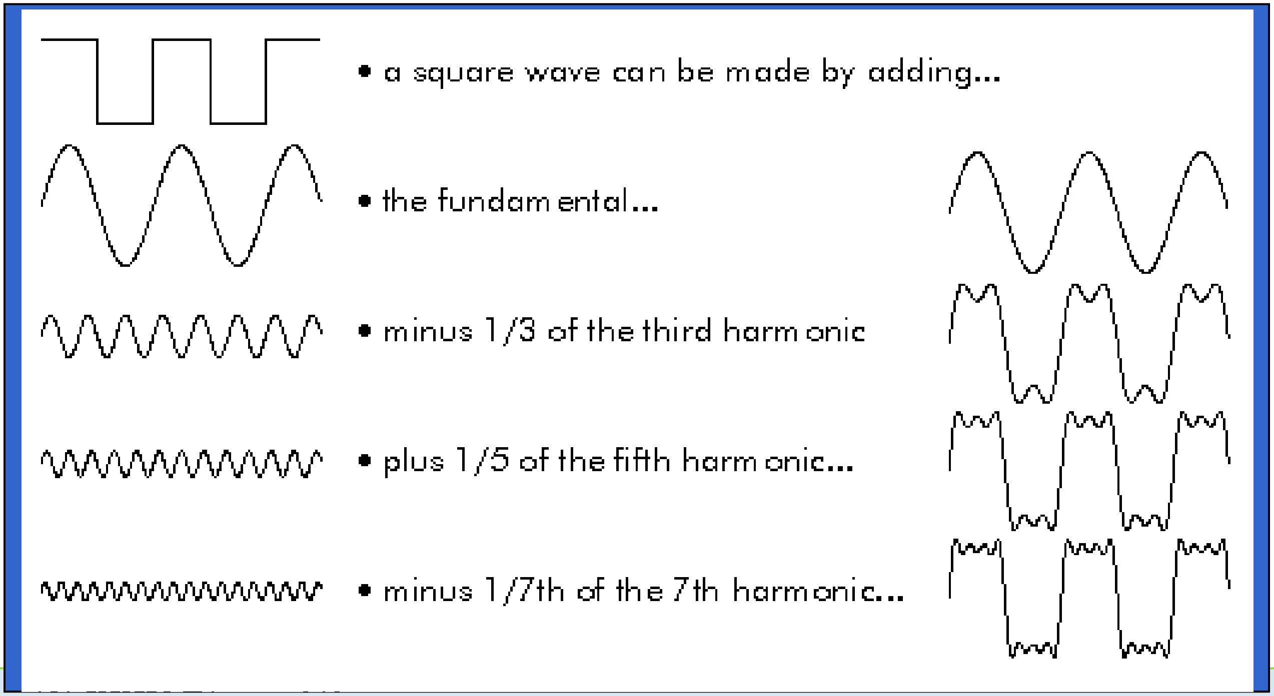
Signal conditioning in FREQUENCY DOMAIN

Fourier → signal = series of sine functions

→ only 3 characteristics: amplitude, phase and frequency

→ reduction of information

e.g. block wave



- a square wave can be made by adding...
- the fundamental...
- minus 1/3 of the third harmonic
- plus 1/5 of the fifth harmonic...
- minus 1/7th of the 7th harmonic...

Digital Processing

- *mathematical technique*

Fourier Transform (TF) → integral form!

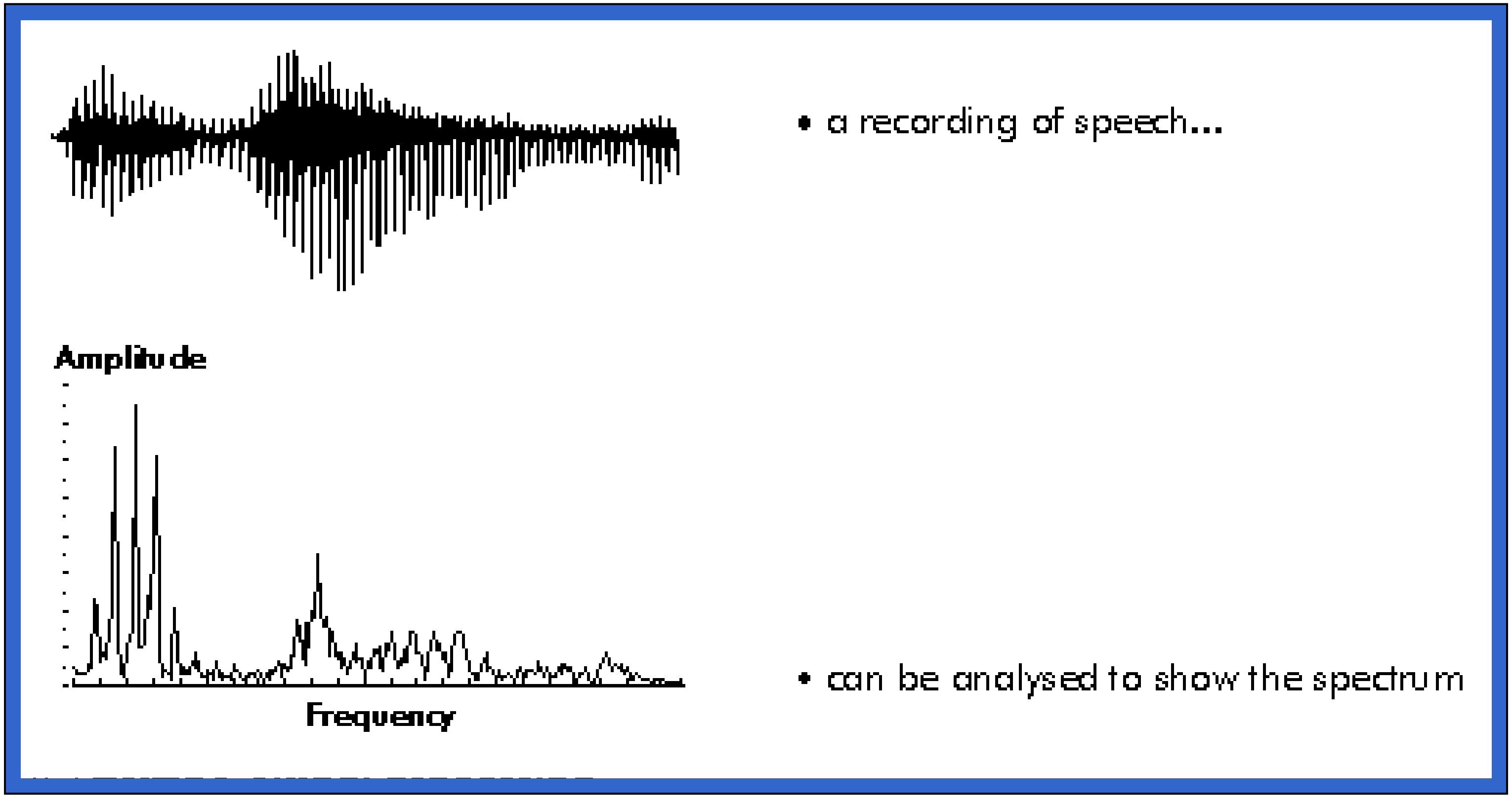
Discrete Fourier Transform (DFT) → discrete equivalent

Fast Fourier Transform (FFT) → practical calculation method

$$f(t) = \sum_{k=1}^{\infty} b(k)\sin(k\varpi t) + a(k)\cos(k\varpi t)$$

Digital Processing

e.g.

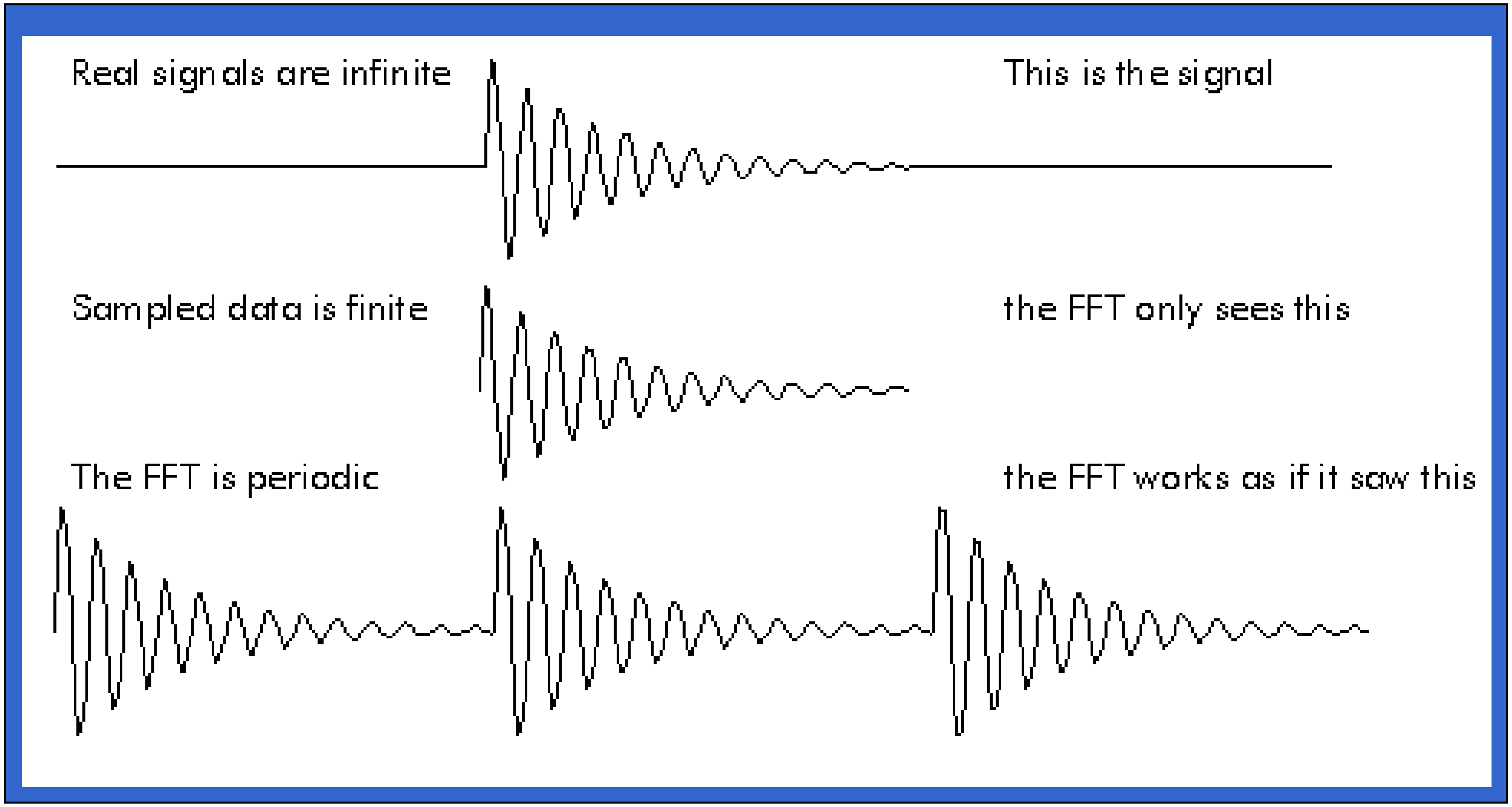


- a recording of speech...
- can be analysed to show the spectrum

Digital Processing

! Remark: FFT only works properly if the signal is periodic

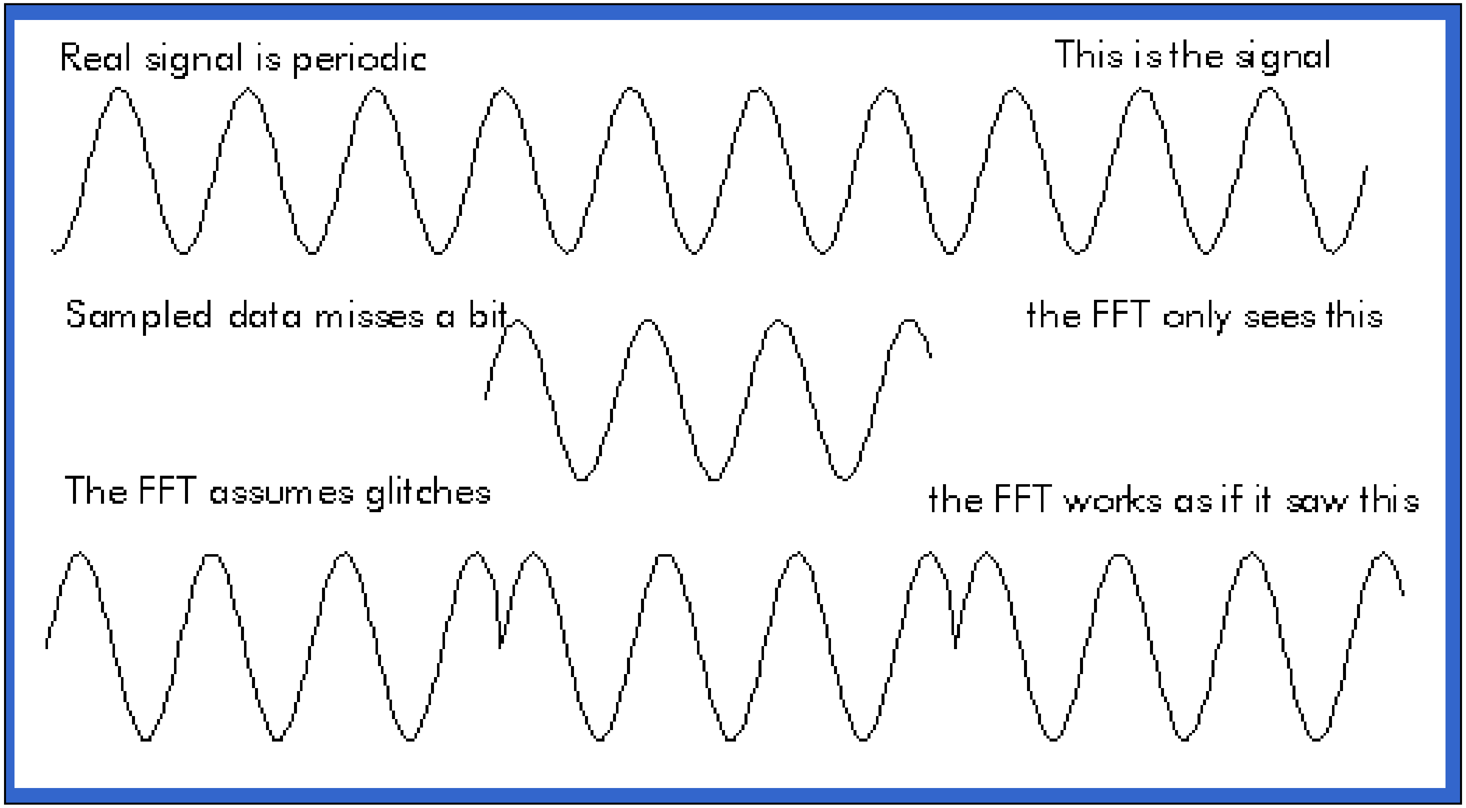
e.g. 1



Digital Processing

! Remark: FFT only works properly if signal is complete

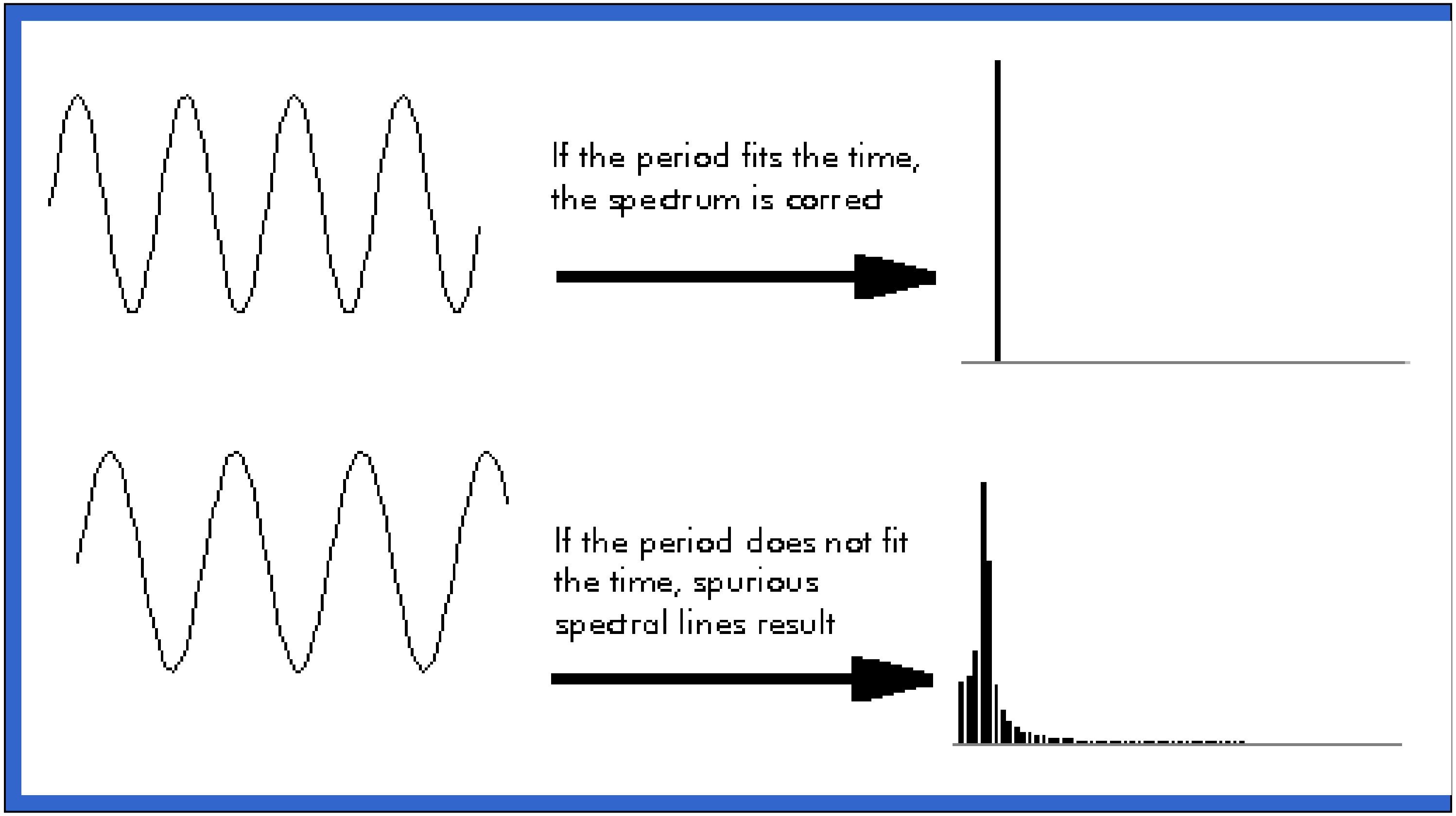
e.g. 2



Digital Processing

! Remark: FFT only works properly if signal is complete

e.g. 2



If the period fits the time, the spectrum is correct

If the period does not fit the time, spurious spectral lines result

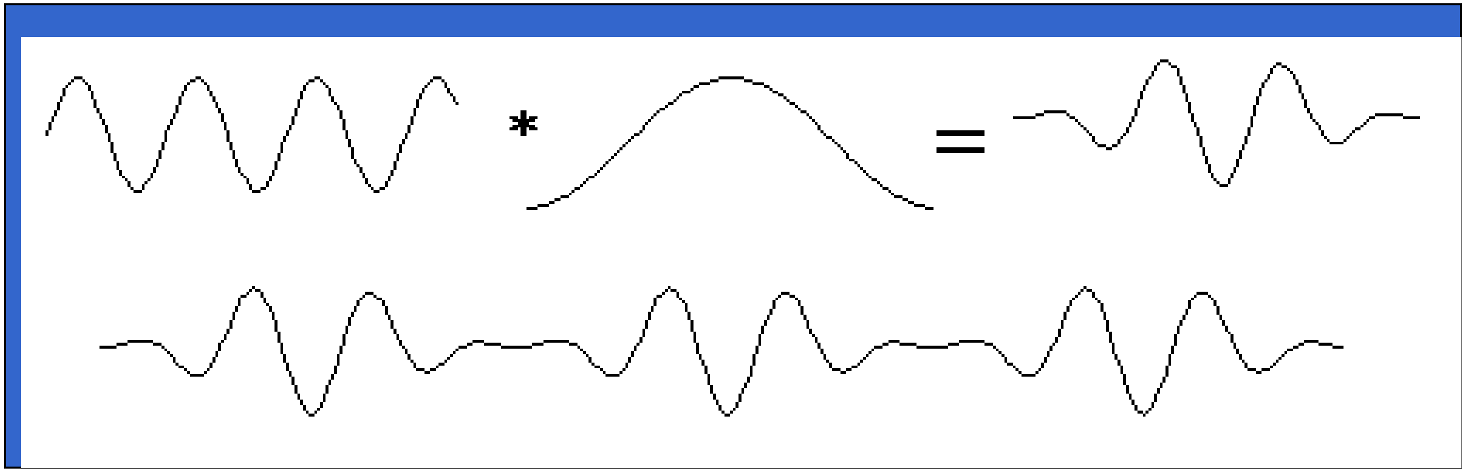
Digital Processing

To narrow the spectrum one needs to reduce the *glitches* artificially by smoothly connecting the signal ends

→ multiply the signal with a *window function*:
one forces the signal to go to zero at the signal's end

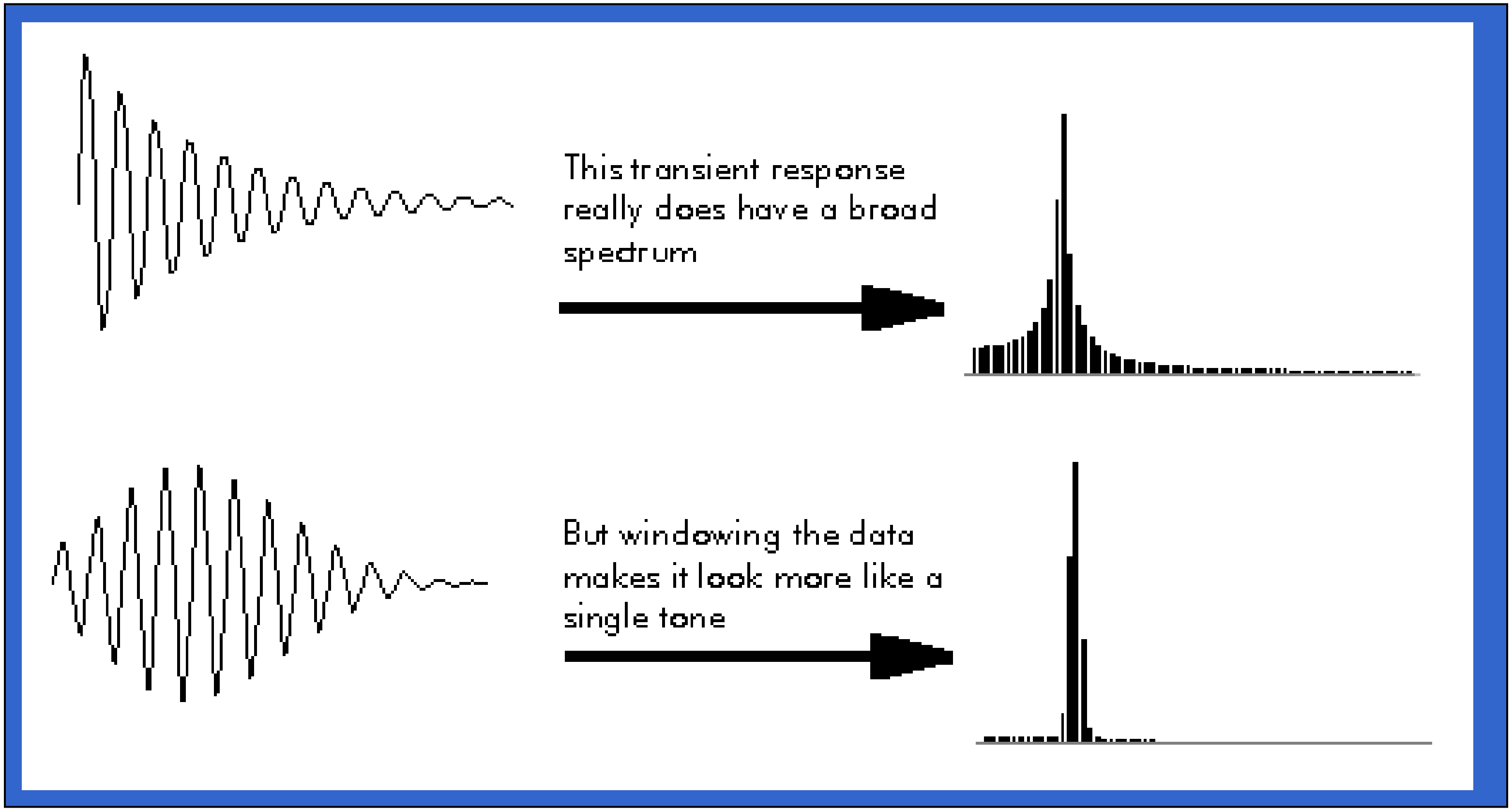
Consequence: more narrow spectrum, but distortion of signal occurs
so, necessity of correct choice of window function

e.g. 1



Digital Processing

e.g. 2



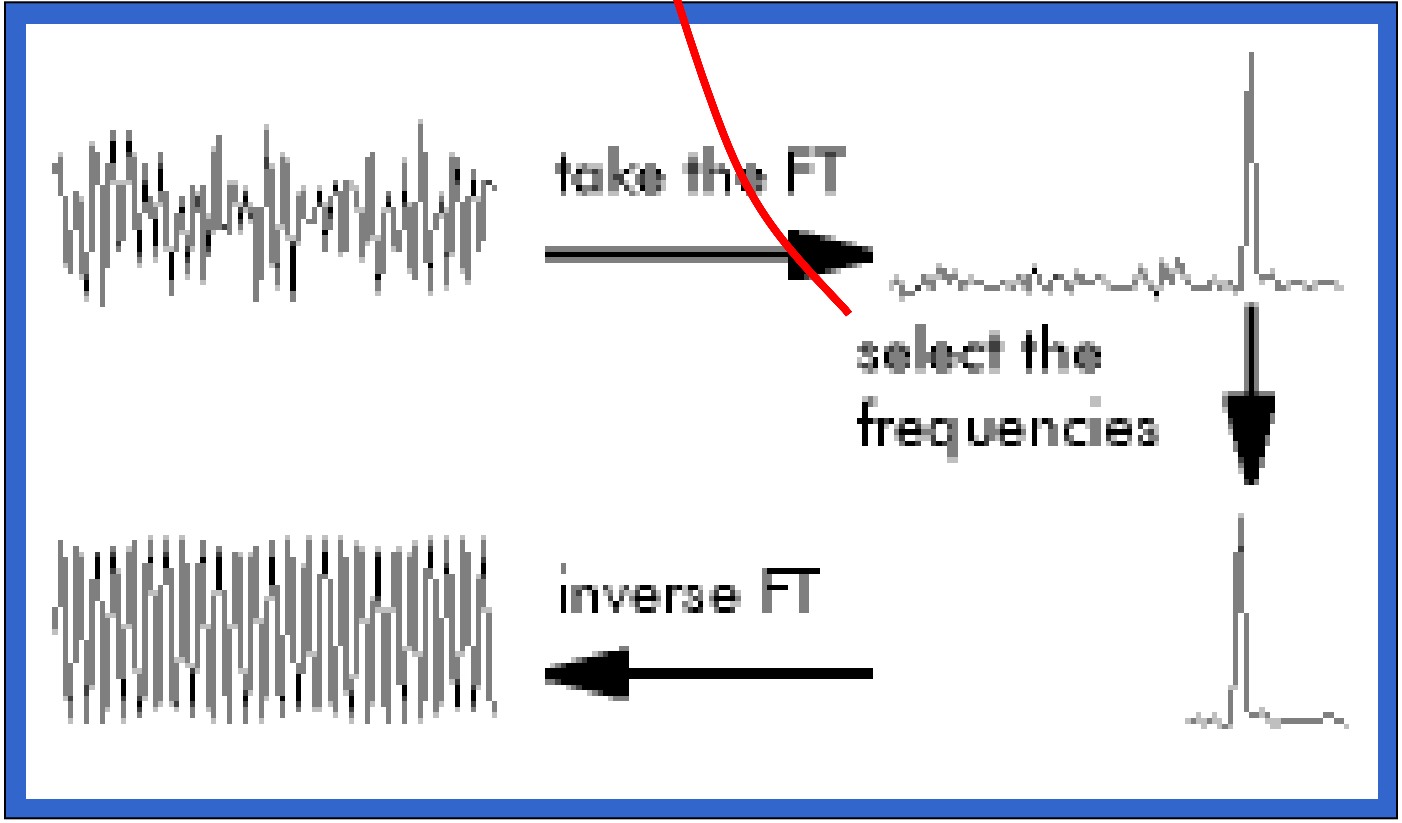
This transient response really does have a broad spectrum

But windowing the data makes it look more like a single tone

Digital Processing

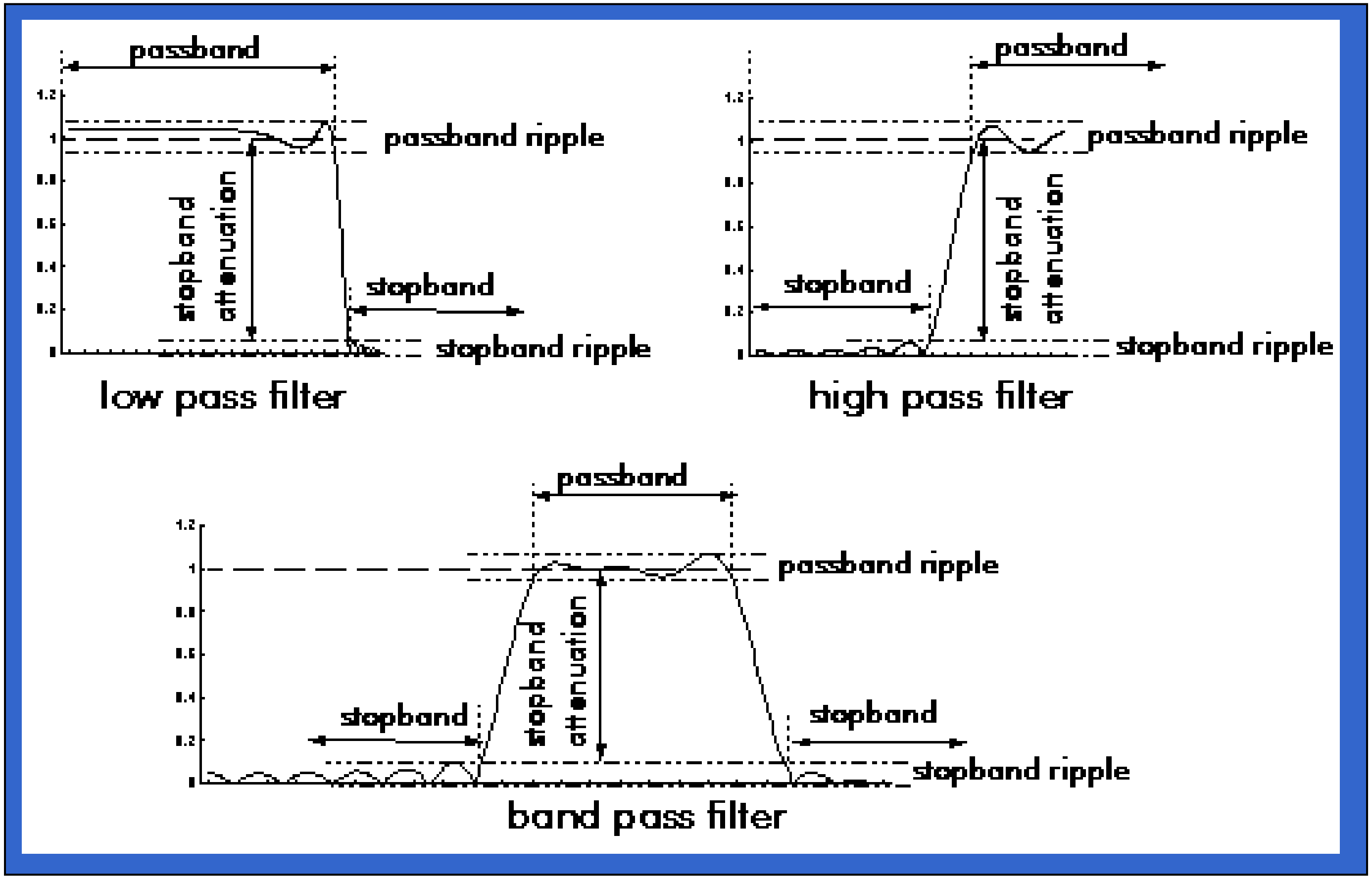
Fourier filtering

$$f(t)=\sum_{k=1}^{\infty} b(k)\sin(k\varpi t)+a(k)\cos(k\varpi t)$$



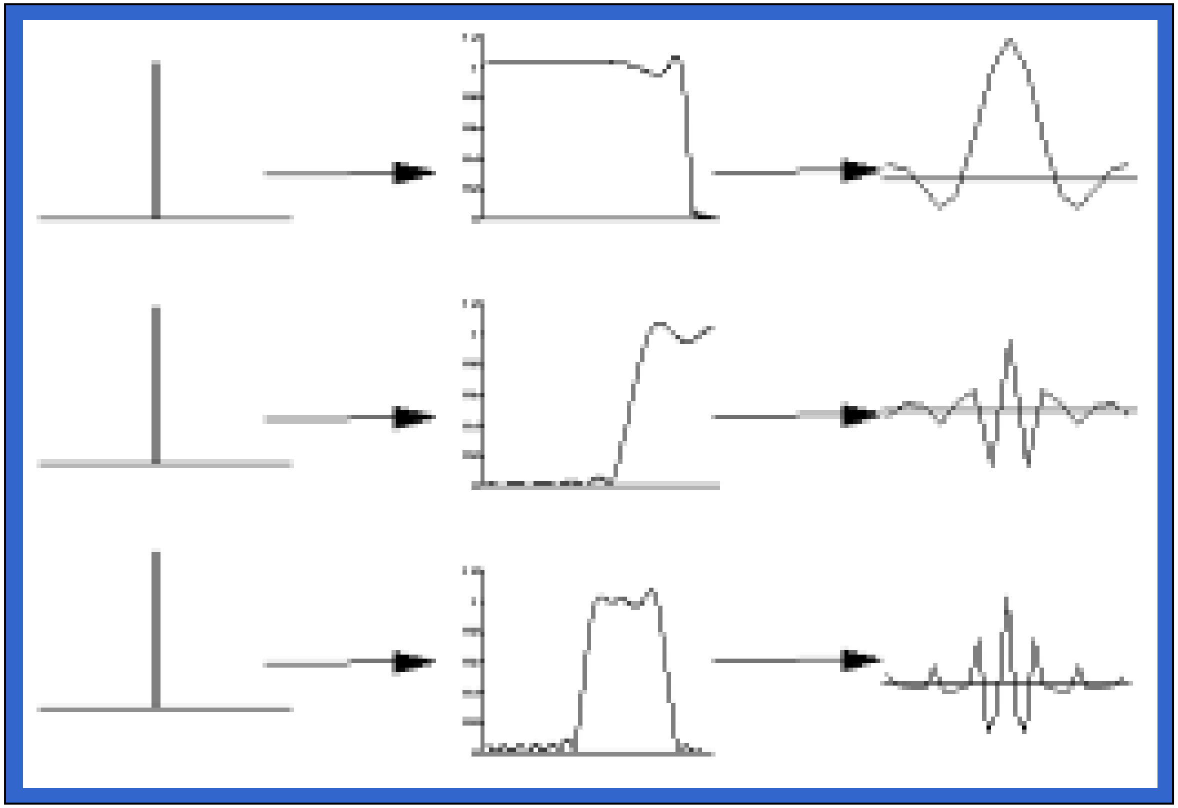
Digital Processing

Digital filter characteristics = choice of frequencies



Digital Processing

Other characteristic is *pulse response*



low-pass filter

high-pass filter

band-pass filter

Digital Processing

Signal conditioning in TIME DOMAIN

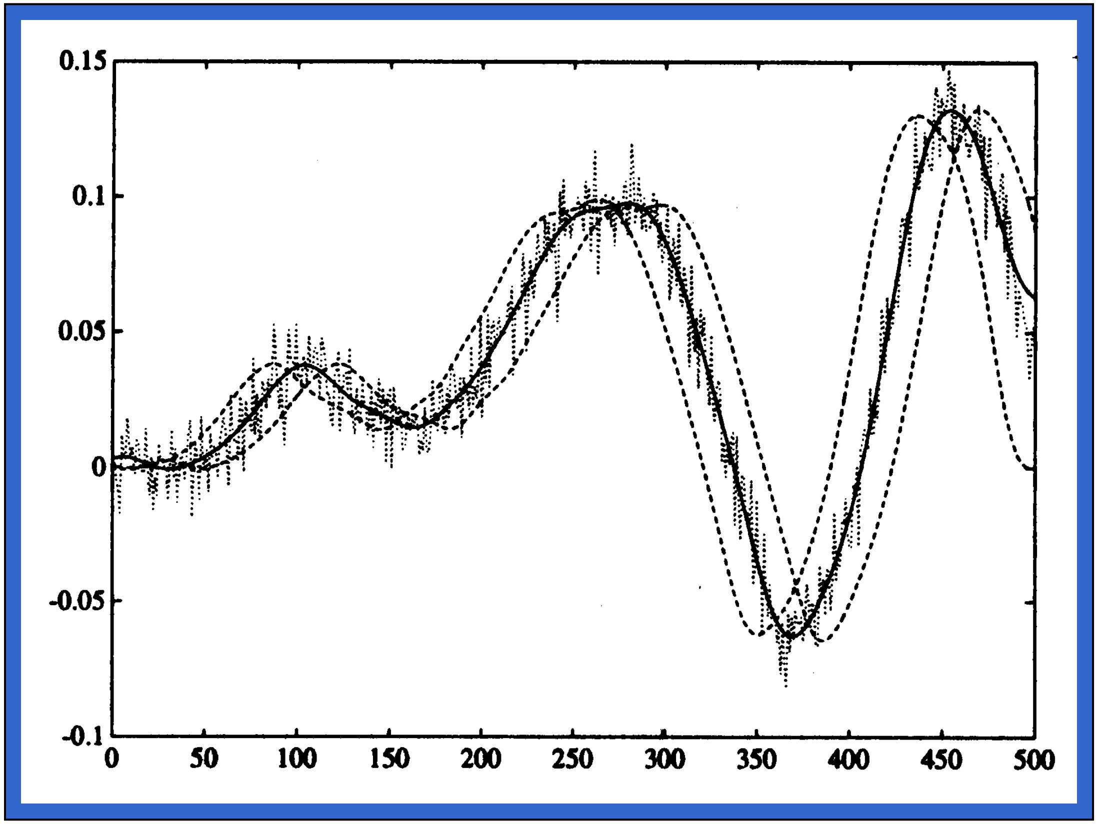
- take averages: $y'(k) = [y(k-1) + 2\ y(k) + y(k+1)\ ]/4$
- for very noisy signals with “outliers”: take $\text{MEDIAN}[\ y(k-j) \ldots y(k+j)\ ]$

- for on-line application: $y'(k) = [y(k-2) + 2\ y(k-1) + 4\ y(k)\ ] / 7$

PROBLEM: ***Data shift in time***

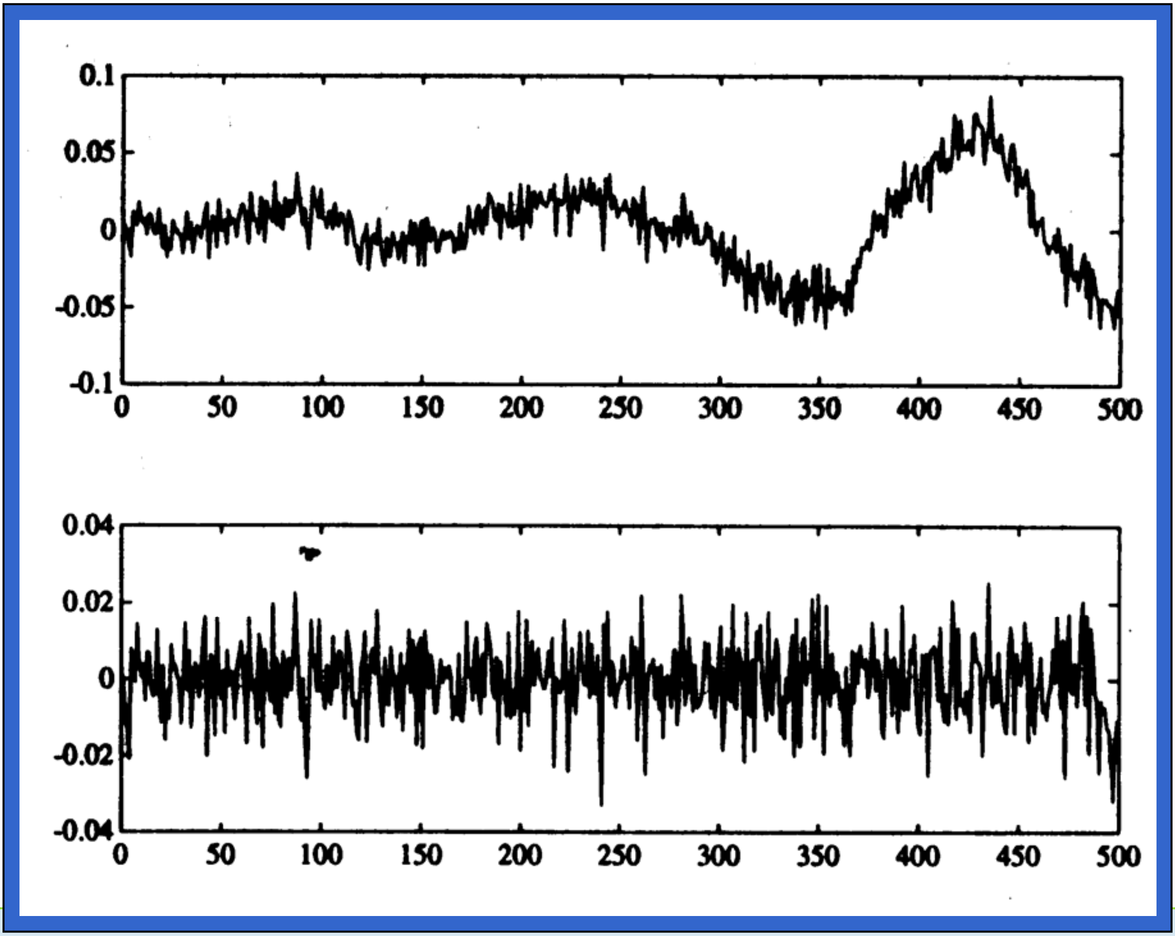
Digital Processing

Phase shift from forward, backward and forward/backward low-pass filter



Digital Processing

Noise remaining after forward (top) and forward/backward filter (bottom)



Digital Processing

Peak shaving

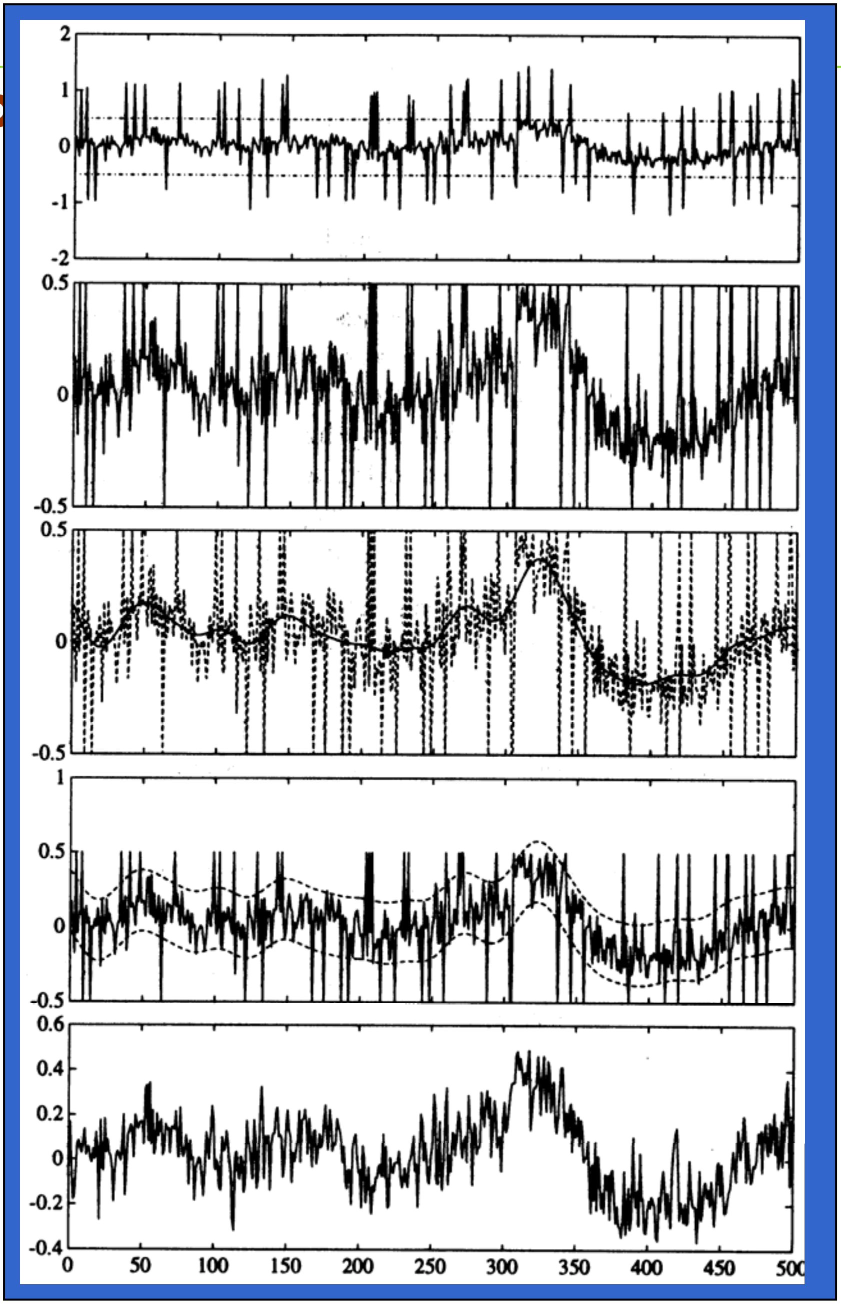
noise peaks introduced by the sensor or the transmission lines (e.g. by on/off switching of devices)

→ need for peak shaving!

a) Clipping the signal amplitudes

$$\tilde{s}_k = f_k . s_k \qquad \text{with} : f_k = \begin{cases} s_{max} \cdot \dfrac{1}{s_k} & \text{if } s_k \geq s_{max} \\ 1 & \text{if } s_{min} < s_k < s_{max} \\ s_{min} \cdot \dfrac{1}{s_k} & \text{if } s_k \leq s_{min} \end{cases}$$

Digital P



a) clipping the signal amplitudes

b) computation of trend signal $\bar{s}_k$ of $\tilde{s}_k$

c) computation of standard deviation

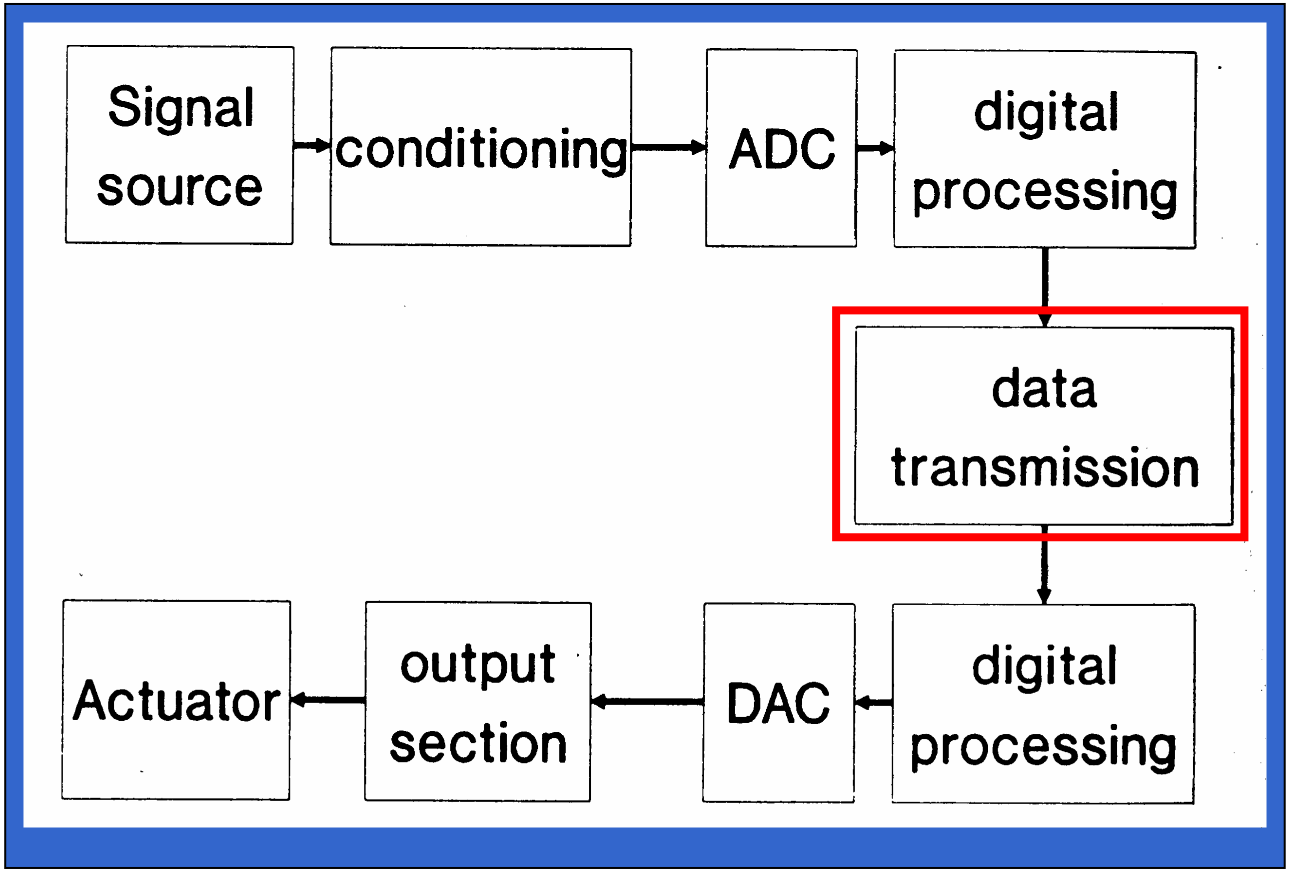
$$\sigma = \sqrt{\sum_{k=1}^{N} \left[\left(\tilde{s}_k - \bar{s}_k \right) - s_a \right]^2}$$

$$\text{with } s_a = \text{average of } \tilde{s}_k - \bar{s}_k$$

d) interpolation of samples outside the band defined as

$$s_k = \begin{cases} \bar{s}_k + \alpha\sigma & \text{upper limit} \\ \bar{s}_k - \alpha\sigma & \text{lower limit} \end{cases}$$

The Key Diagram



4. Transmission

Data Transmission

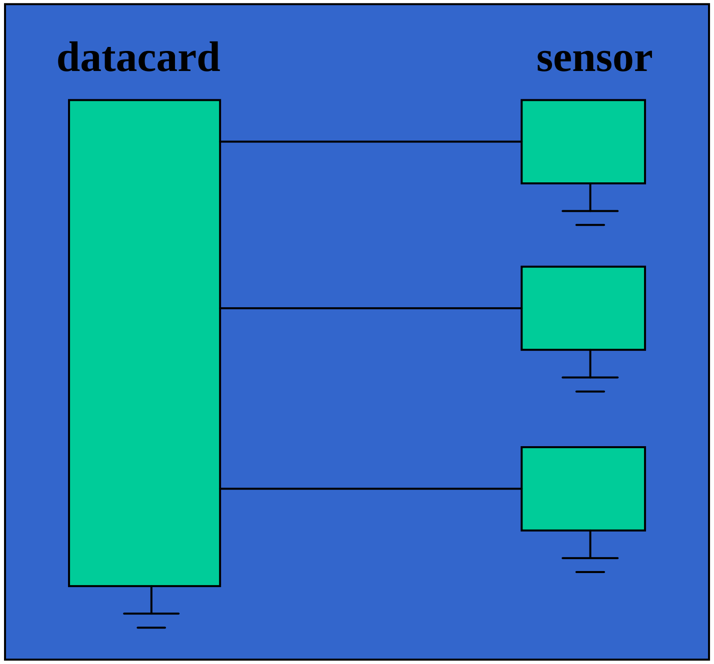
Introduction

single ended - differential signals

analog - digital signals

Data Transmission

Single ended data transmission



Advantages:

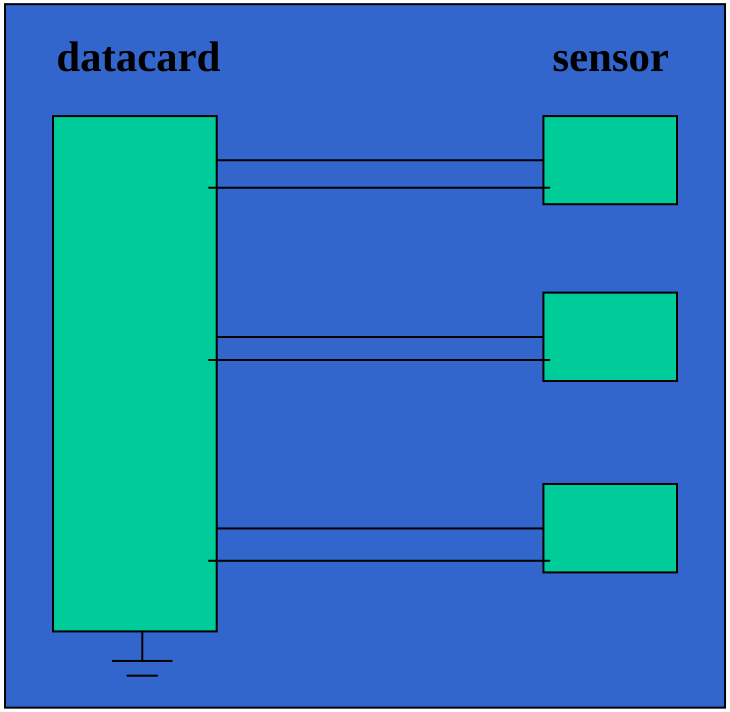
- low cost
- simple

Disadvantages:

- noise sensitive
- low transmission speed
- short lines
- grounding

Data Transmission

Differential data transmission



Advantages:

- fast transmission speed
- long transmission lines
- noise insensitive
- no grounding

Disadvantages:

- more costly

Data Transmission

Analog data transmission

→ signal amplification

A-signal: 0-20 mA and 4-20 mA (4mA □cable rupture)

± immune to noise

total resistance < 600 Ω

V-signal: 0-1 V, 0-5 V and 0-10 V

sensitive to noise

total resistance > 100 kΩ

Data Transmission

Digital data transmission

Only 2 signal levels: 0 and 1 => low noise sensitivity

Synchronization of sender-receiver needed

Fault detection possible:
- parity
- check of sums
- redundancy

Transfer speed (baudrate) [bits/s]

→ digital speed > real information speed:
(due to fault detection needs)

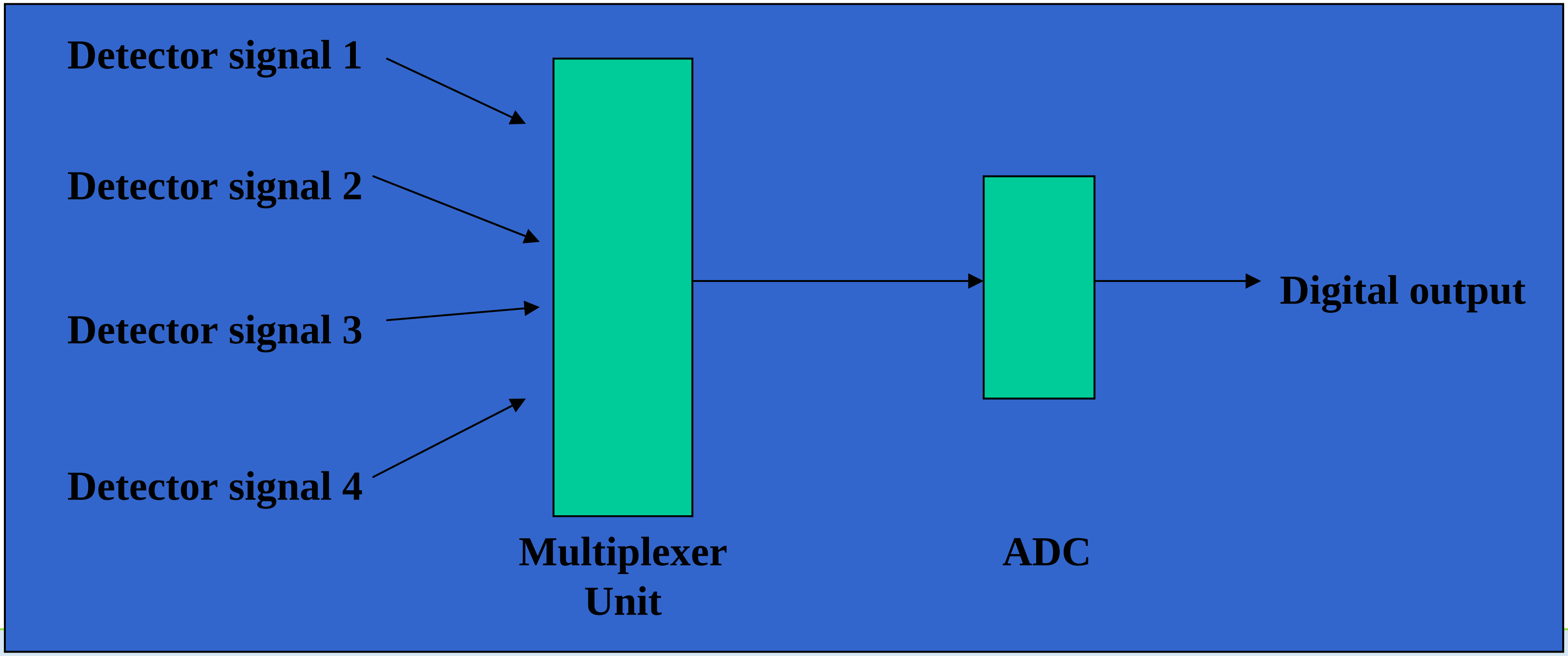
Data Transmission

Multiplexing of digital data

Multichannel data-acquisition with single ADC

! *Crosstalk*: interchannel interference → loss of data integrity

Control of *sampling rate* and *sampling sequence*



Data Transmission

Parallel and serial data communication for digital data

a) <u>Parallel communication</u>

Distribute bytes over several transmission lines

! Need for synchronization

! Limited cable length

Data Transmission

b) Serial communication bits are sent sequentially over 1 cable

→ **RS-232 interface** (single ended)

devices with a standard RS-232 interface can *not* be connected to each other without any problems due to:

- existence of many “232” norms
  (EIA-232, RS-232-C, RS-232-D, EIA/TIA-232-F)
- existence of many connector types (DB25, DB9, OEM)
- speed of transmission
- number of bits per byte
- number of stop bits: 1 or 2? Parity bit?
- protocol of data transmission:
  direct transmission after data generation *or* data storage

Data Transmission

→ **RS-422** and **RS-485** (differential)

advantages over RS-232:

- higher transmission speed
- longer transmission lines
- less noise sensitive

Data Transmission

Internal data transport

data is transported to RAM or CPU of control device

How?

- ADC has memory location characteristics for CPU
- port address

→ last available data, does not have to be the most recent one

Data Transmission

CPU copies data to another location to avoid eventual overwriting. To do so, there exist several possibilities:

- *polling*: continuous CPU monitoring of the memory location
- *hardware or software protocol*: data exchange at specific times
- *interrupt driven strategy*: data exchange at every moment
  (disadvantage: no other tasks during data logging)
- *Direct Memory Access*: DMA-chip realizes data transfer

  (advantage:CPU can execute other tasks meanwhile)
  (disadvantage: costly)

Control Devices

- **Control devices**

- *PC*

speed not only dependent on hardware, but also on the OS (DOS vs. MS Windows)

- *PLC* (Programmable Logic Controllers)

program is run sequentially and repeated

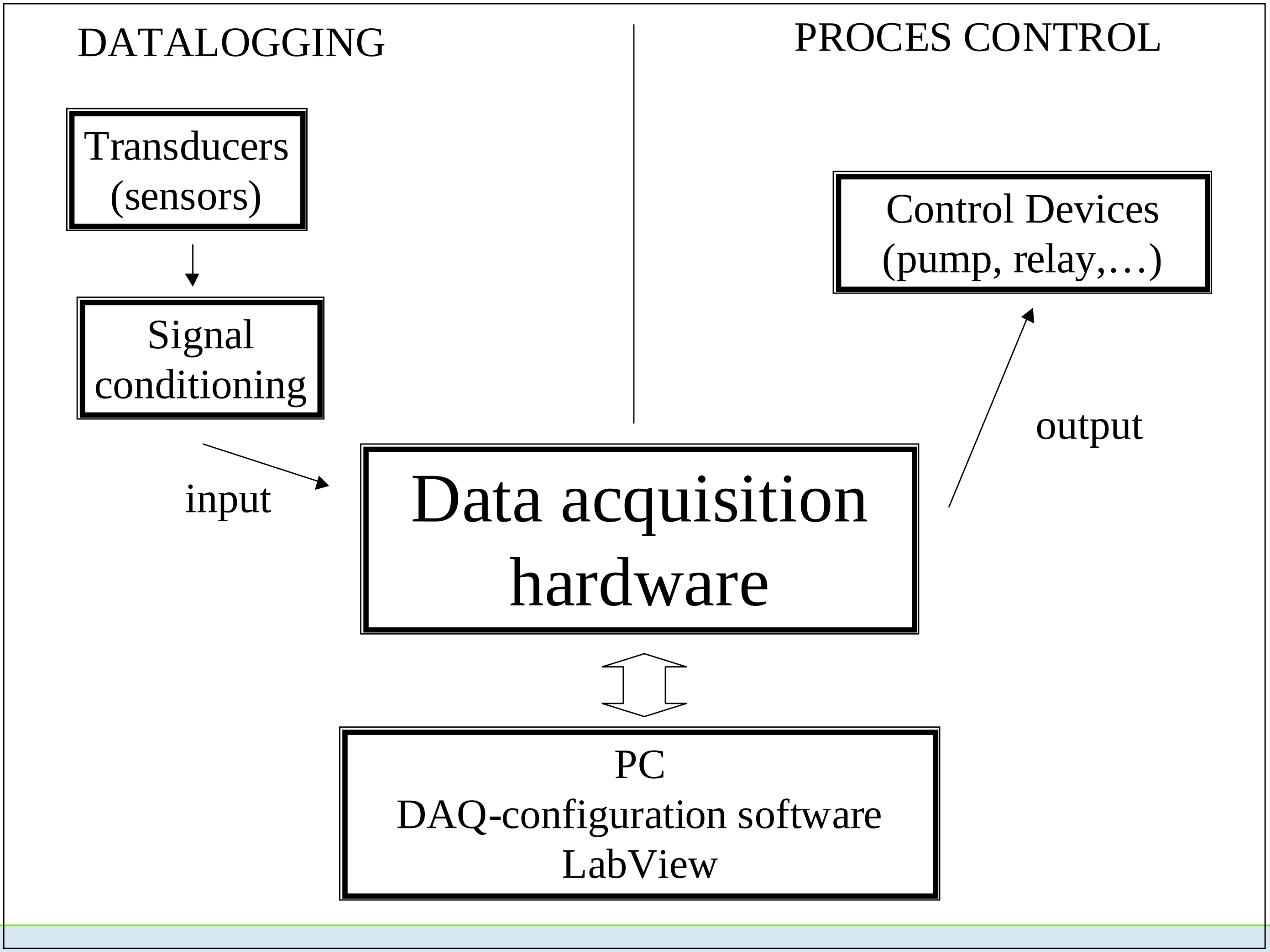
skipping lines or jumping back is impossible

advantage: infinite loops are impossible
a fixed run time

disadvantage: little flexible
no complete control algorithms

DAQ in a research environment

building your own DAQ system



DATALOGGING

PROCES CONTROL

Transducers (sensors)

Signal conditioning

input

Data acquisition hardware

output

Control Devices (pump, relay,…)

PC
DAQ-configuration software
LabView

DAQ in a research environment

building your own DAQ system

- Datalogging types
  - Analog input (AI)
    - e.g. pH, DO, conductivity, ORP, T
    - amplification step required (transmitter) for each signal
    - transmitter reads weak signal, sends amplified signal (0-20mA, 4-20mA)
    - typically has calibration features on board

  - Digital input (DI)
    - e.g. level sensor
    - signal takes 2 discrete values, typical 0 or 5 V

DAQ in a research environment

building your own DAQ system

- Datalogging types
  - Connect to DAQ hardware
  - Serial devices can be connected directly to PC (if serial bus available); RS232 cards available as well
  - only for simple systems with limited number of signal (e.g. analytical balance)

DAQ in a research environment

building your own DAQ system

- Device control
  - Analog output
    - e.g. pump
    - typical output delivered by DAQ-hardware between 0-10V
  - Digital output
    - e.g. valve
    - small output voltage delivered by DAQ hardware
    - couple with relay (on-off circuit) and power supply (typically 24V DC)

DAQ in a research environment

building your own DAQ system

- Choose your DAQ hardware

  - Number and type of signals?

  - separate cards for AI, AO, DI/O
  - multipurpose cards

  - important features:
    - AI: sampling rate, resolution, input range
    - AO: resolution

    - the better, the more expensive !!

DAQ in a research environment

building your own DAQ system

- DAQ software
  - e.g. Labview by National Instruments (www.ni.com)
  - allows channel configuration
  - low level graphical programming
  - write your own “Virtual Instruments” for DAQ and control of you lab-scale system
    - from simple to complex
    - with all flexibility you want
    - publish it to the web to check system status wherever you are
    - …

- Hyperterminal: feature in Windows to read data from serial devices