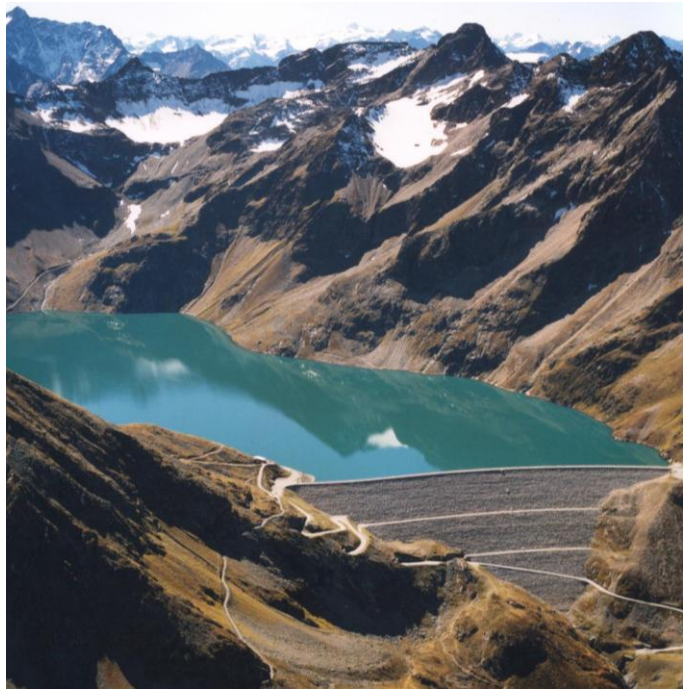
When Code Moves Metal

An Introduction to PLCs for Software Engineers

Matthias Seehauser – Zeugwerk GmbH | Engineering Kiosk Meetup | Innsbruck, 19.03.2026

What is producing/powering all those goods?

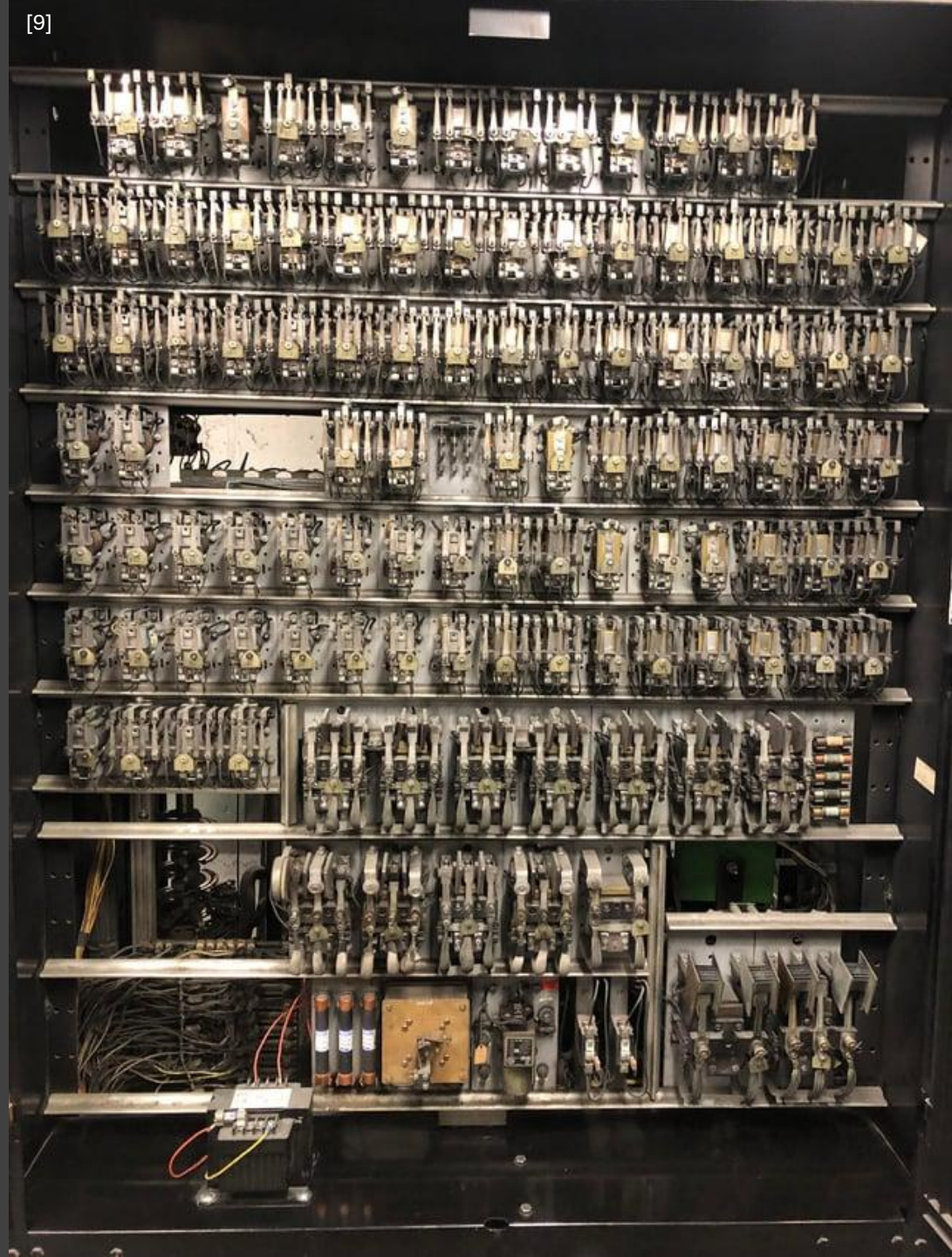


What do all these machines have in common?

Right, the control system – a PLC!

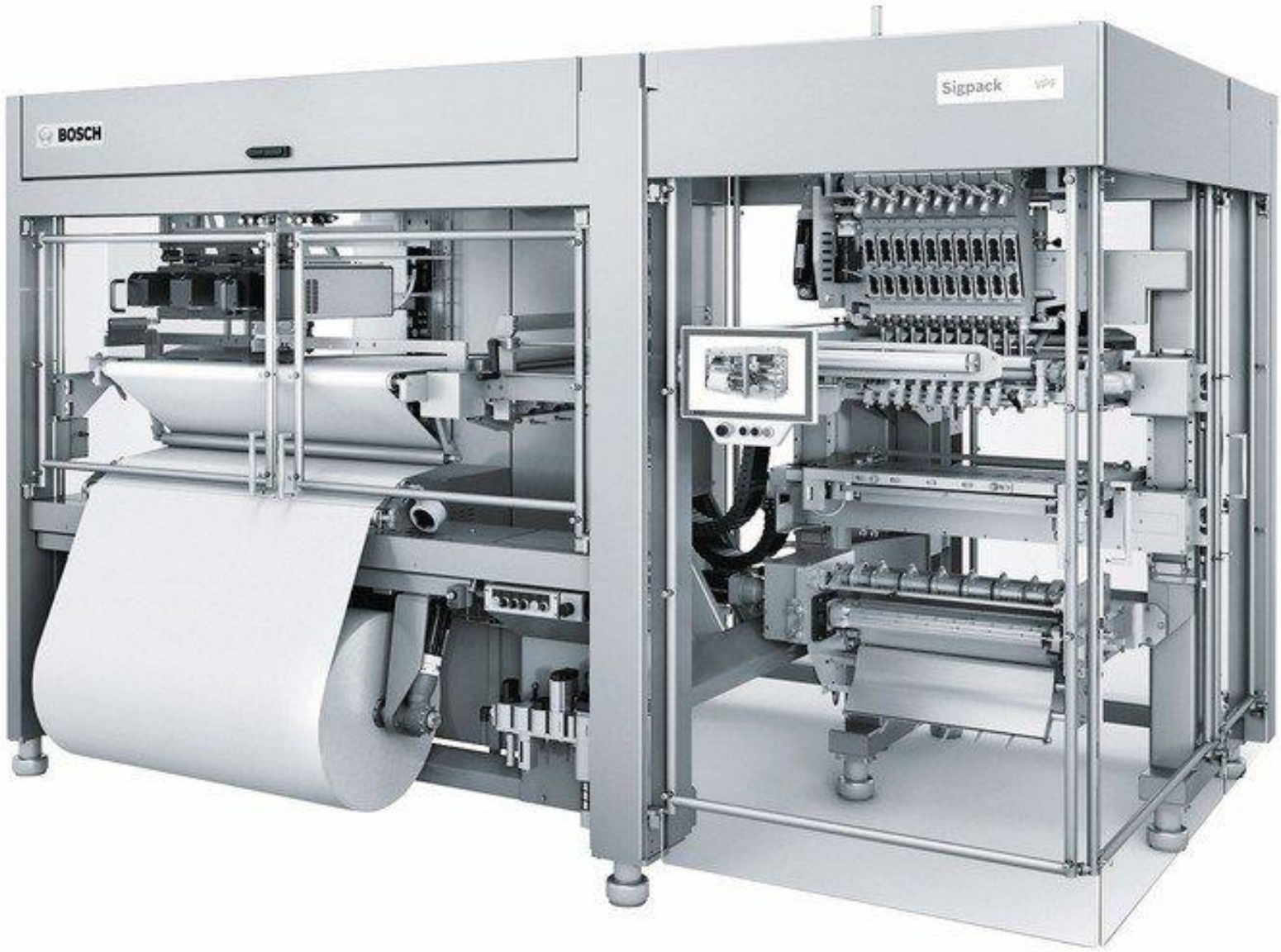
History of PLCs

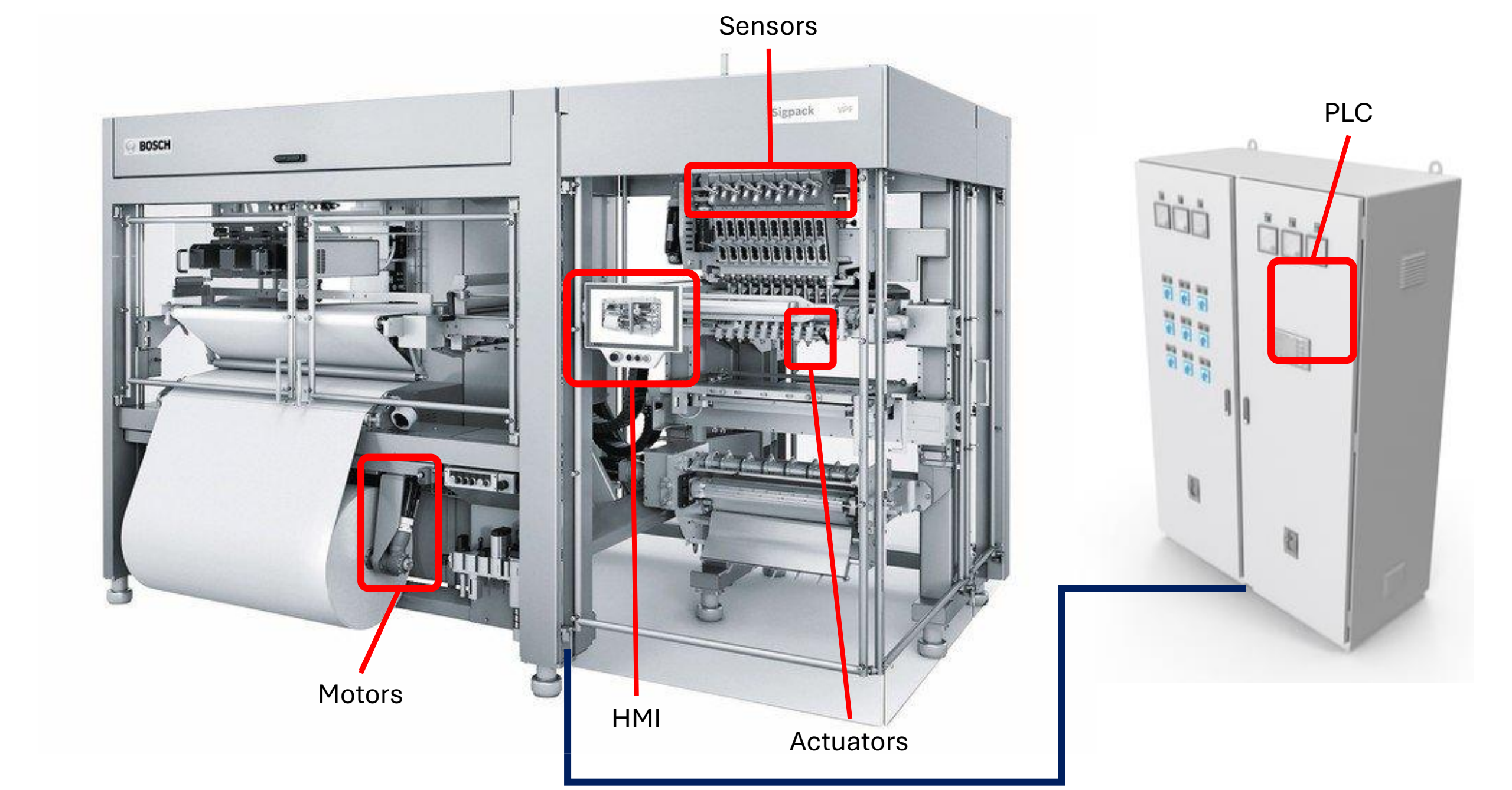
- It is an industrial computer running special software for automation
- Originated 1960s in the automotive industry replacing relay logic systems
- These relay circuits made it difficult to
  - find and fix „bugs“
  - Adapt to changing processes
  - Scaling was hard


[9]

Just for completeness…

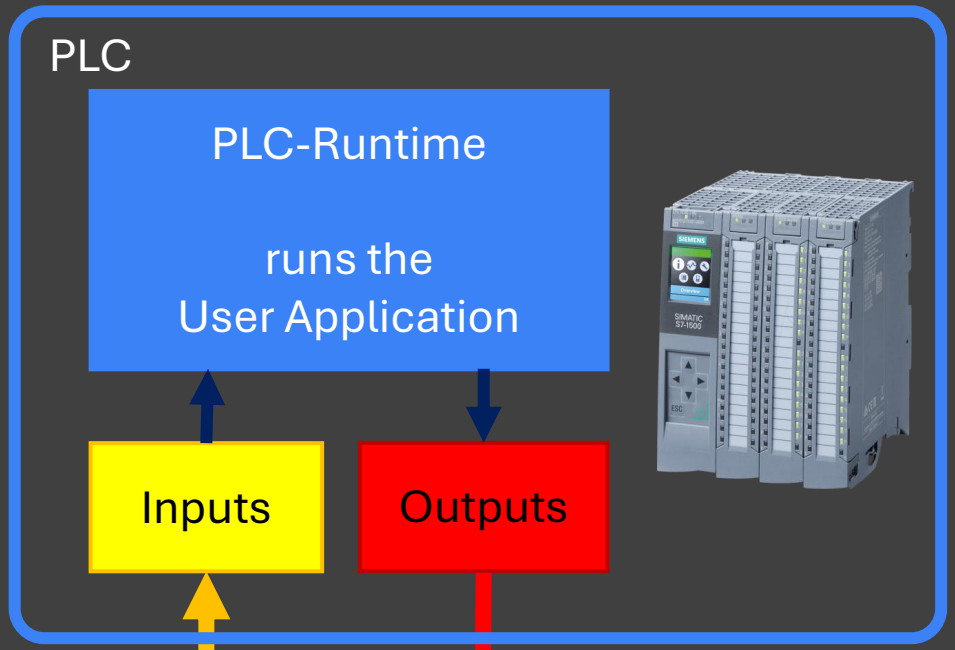
- 1969 first PLC was build it was the Modicon 084 – modular digital controller build by Dick Morley and Bedford Associates
- 1970 Allen Bradley introduced the programmable logic controller Odo Josef Struger – „PLC acronym“
- Struger also played a leadership role in developing the IEC61131-3 PLC programming language standards

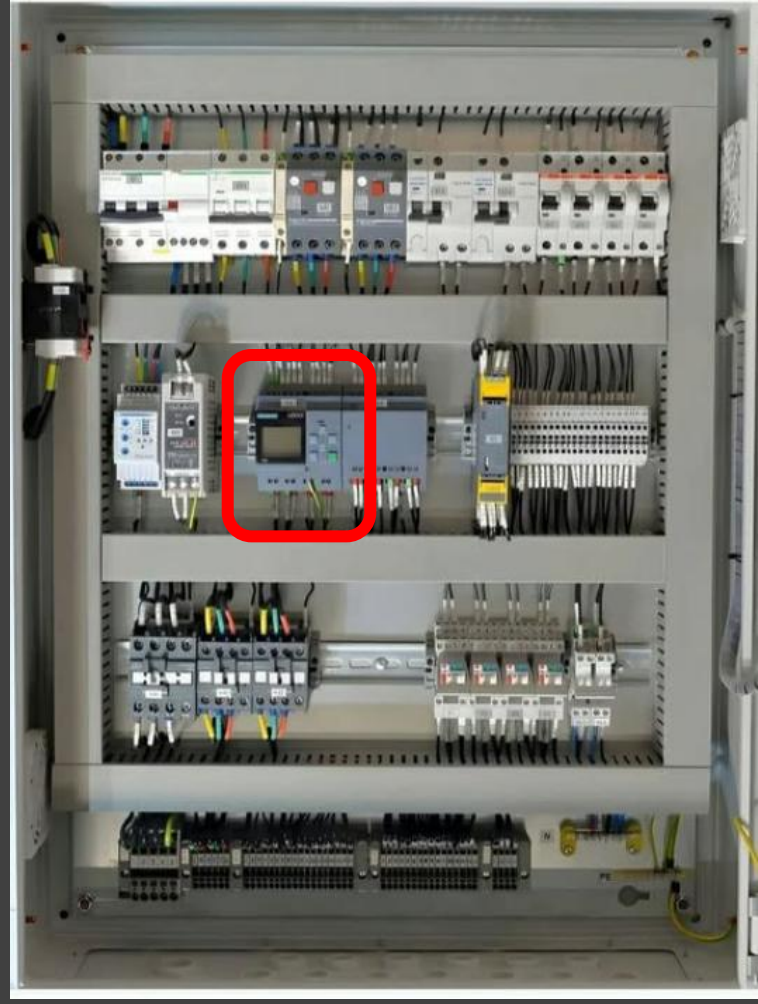

BOSCH
Sigpack

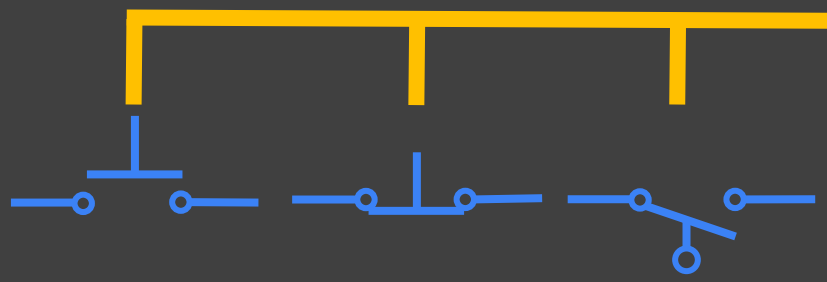

Sensors
PLC
Motors
HMI
Actuators
BOSCH
Sigpack

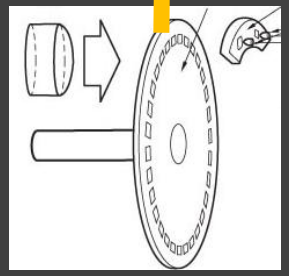
Architecture

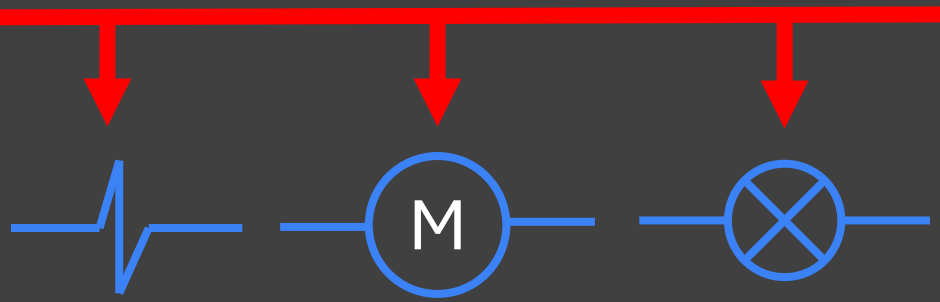
Engineering PC











Types

[5]

- PLC
  - Runs on dedicated controller
  - Firmware takes care of its function
  - Often comes with hardware inputs and outputs
  - Fieldbus devices can be attached
- SoftPLC
  - Runs on ruggedized PCs (x86) or embedded controller (ARM)
  - Runtime runs with high priority on an operating system (Linux, Windows)
  - mostly no hardware inputs or outputs onboard → Fieldbus necessary

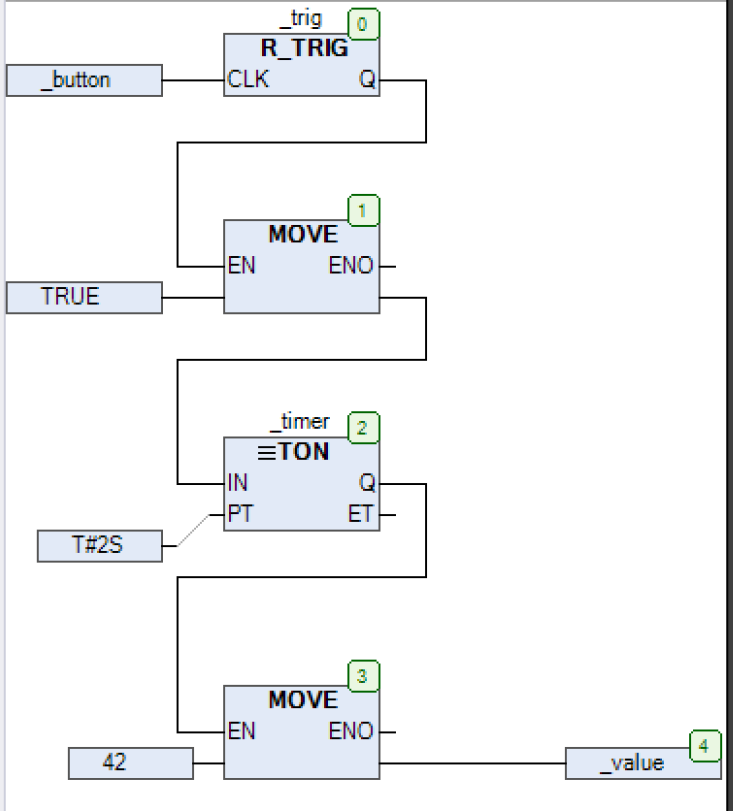
User Application

- Mostly programmed in IEC61131-3 standardized languages
- These have some strict rules
  - Avoid non-deterministic operations, like system calls or dynamic memory allocation.
  - Programs run in a **main loop**
  - No waiting allowed
  - The program cycle time needs to be smaller than the parametrized cycle time → deterministic behavior / predictable timing
  - Always leave some headroom for PLC runtime background tasks and worst-case execution paths in your program.

Programming languages

CfcExample

```
1  PROGRAM CfcExample
2  VAR
3    _timer : Tc2_Standard.TON;
4    _trig : Tc2_Standard.R_TRIG;
5    _button : BOOL;
6    _value : UINT;
7  END_VAR
```



StExample

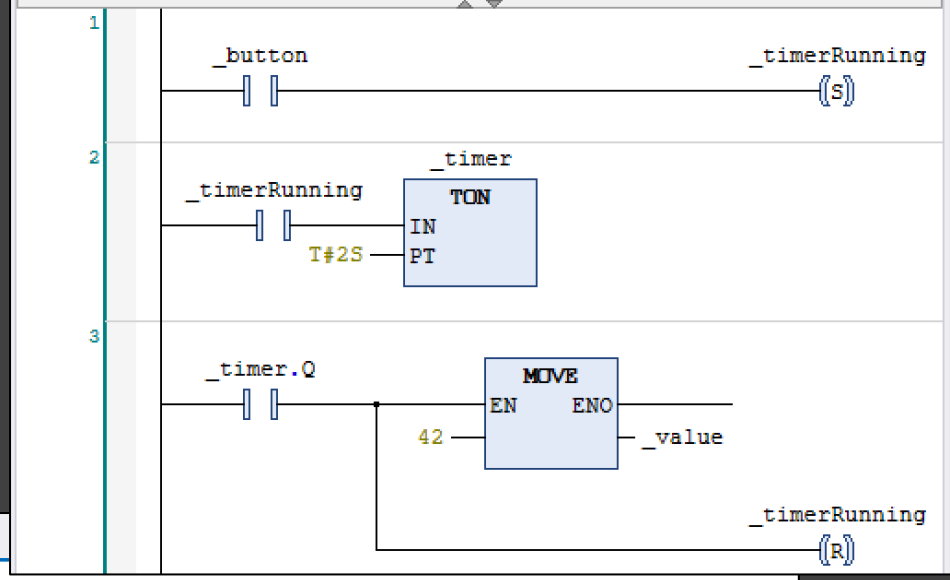
```
1  PROGRAM StExample
2  VAR
3    _timer : Tc2_Standard.TOF;
4    _trig : Tc2_Standard.R_TRIG;
5    _button : BOOL;
6    _value : UINT;
7  END_VAR
```

```
1  _trig(CLK:=_button);
2  IF _trig.Q THEN
3    _timer(PT:=T#2S, IN:=TRUE);
4  END_IF
5  _timer(IN:=TRUE);
6  IF _timer.Q THEN
7    _value := 42;
8  END_IF
```

KopExample

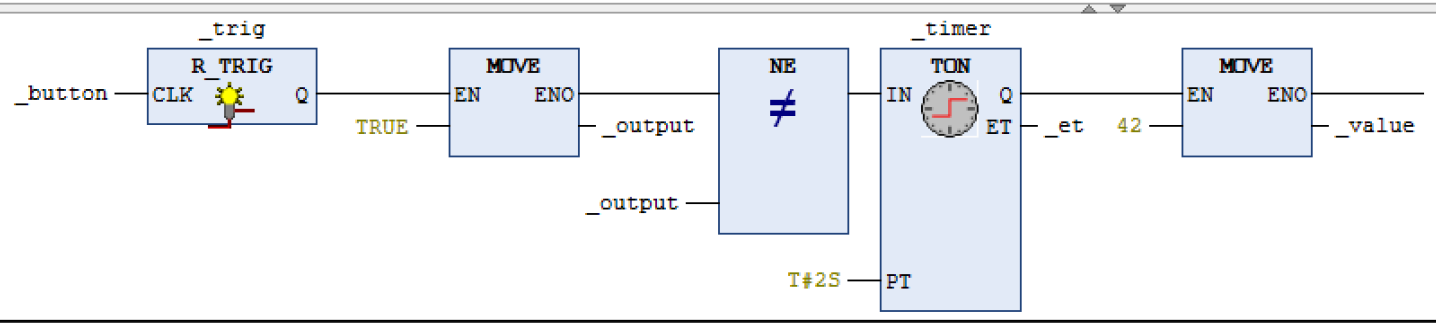
```
1  PROGRAM KopExample
2  VAR
3    _timer : Tc2_Standard.TON;
4    _button : BOOL;
5    _timerRunning : BOOL;
6    _value : UINT;
7  END_VAR
```

100 %



FupExample

```
1  PROGRAM FupExample
2  VAR
3    _timer : Tc2_Standard.TON;
4    _trig : Tc2_Standard.R_TRIG;
5    _button : BOOL;
6    _output : BOOL;
7    _value : UINT;
8    _et : TIME;
9  END_VAR
```



PLC Runtime

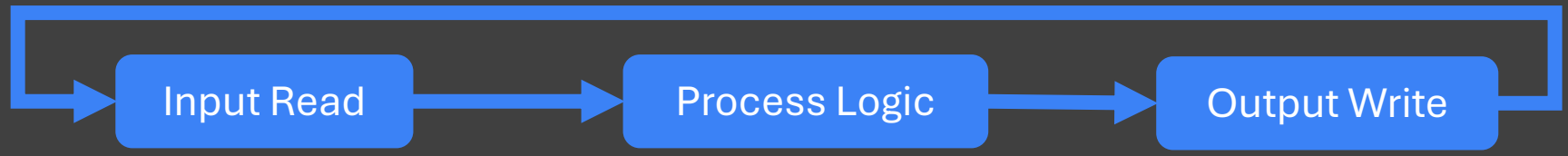
The IPO Principle: **I**nput Read → **P**rocess Logic →**O**utput Write

Contrary to IT → most systems work event driven

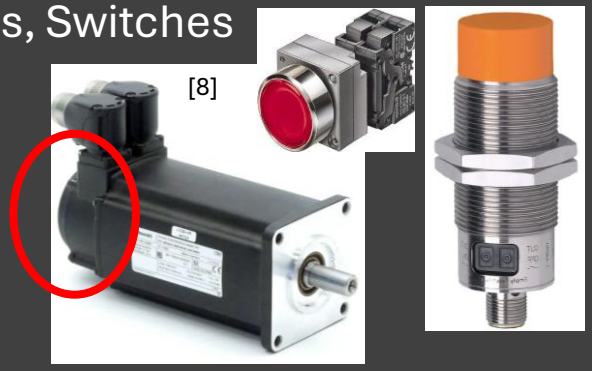
PLC Runtime

The IPO Principle: **I**nput Read → **P**rocess Logic →**O**utput Write

Contrary to IT → most systems work event driven



Sensors, Switches



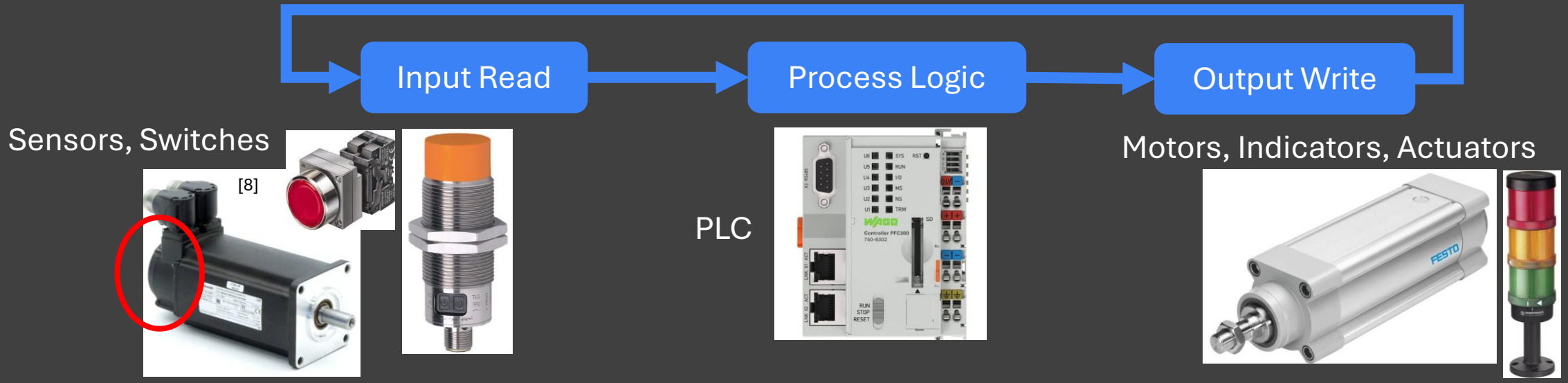
PLC

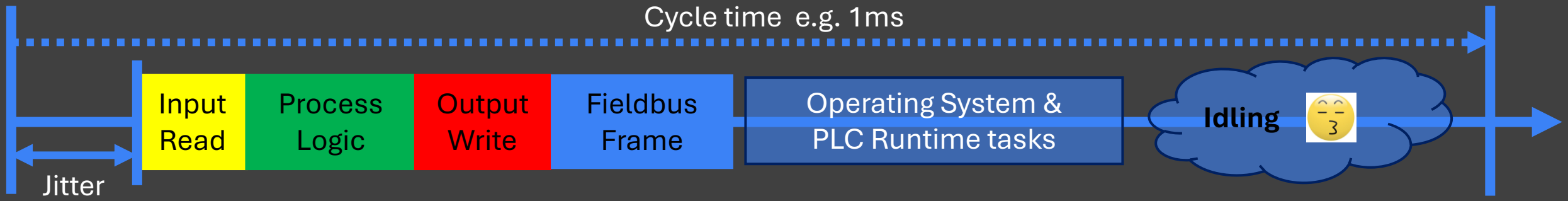
Motors, Indicators, Actuators

PLC Runtime

The IPO Principle: **I**nput Read → **P**rocess Logic →**O**utput Write

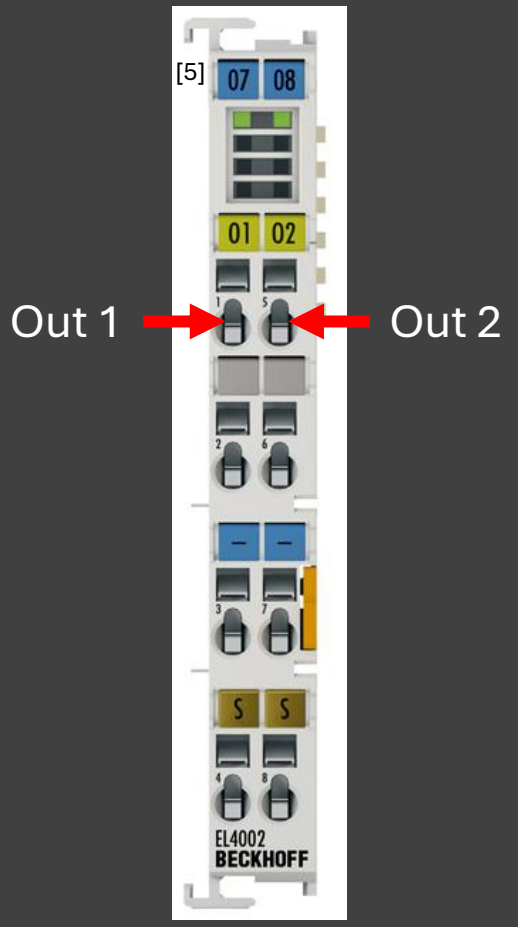
Contrary to IT → most systems work event driven



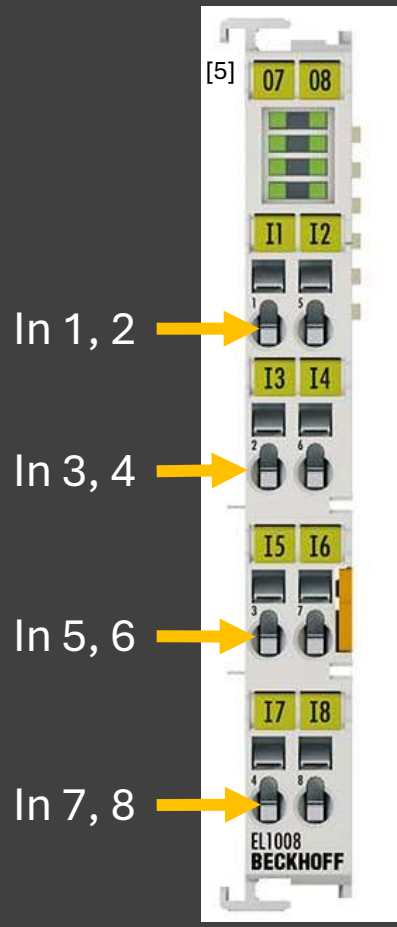


Accessing Hardware

- Standardized Reading and Writing of IO‘s
- Occurs in a fixed cycle to ensure deterministic values
- Hardware is accessed through direct memory mapping
  → no drivers
- Examples:
  - EL4002: 2 channel analogue output (~100€)
  - EL1008: 8 channel digital Input (~33€)



Memory mapping
4 Byte
2 Byte each Output

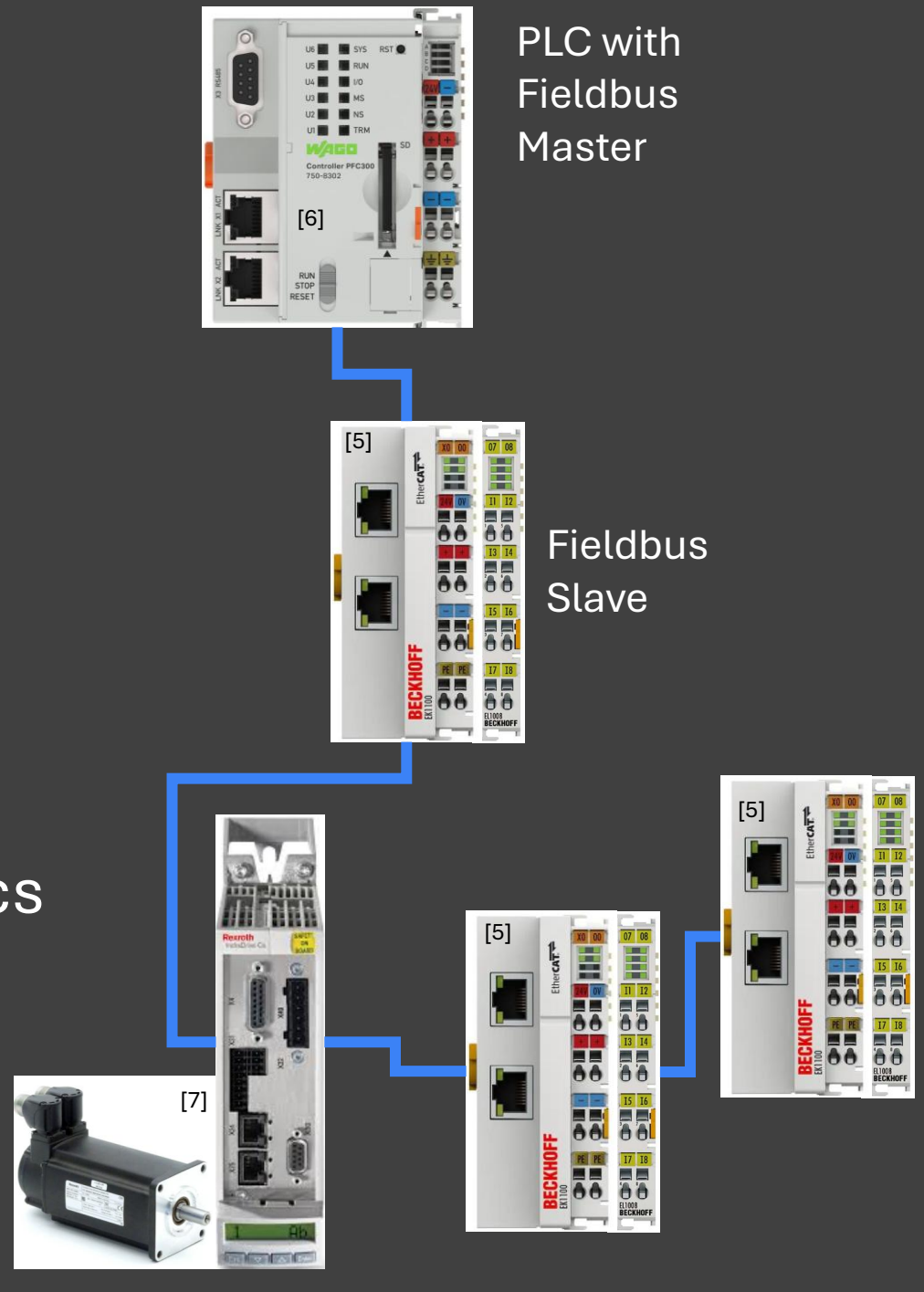


Memory mapping
1 Byte
1 Bit each Input

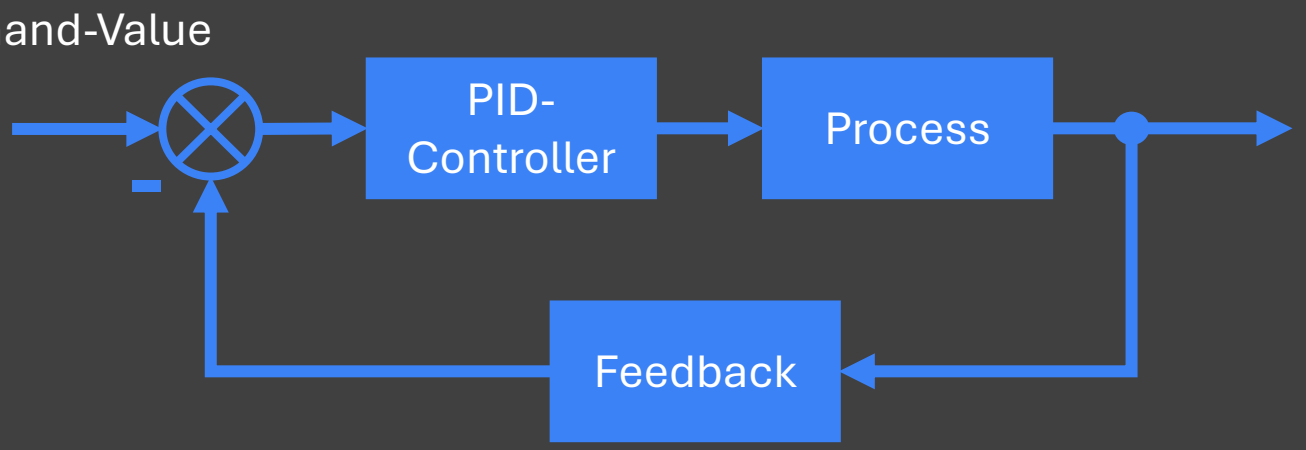
Fieldbus

Fieldbus systems offer deterministic data exchange, and seamless integration automation devices

- Deterministic communication
- Integration of distributed I/O and field devices
- High reliability and redundancy features
- Reduced wiring and improved diagnostics



Real-Time

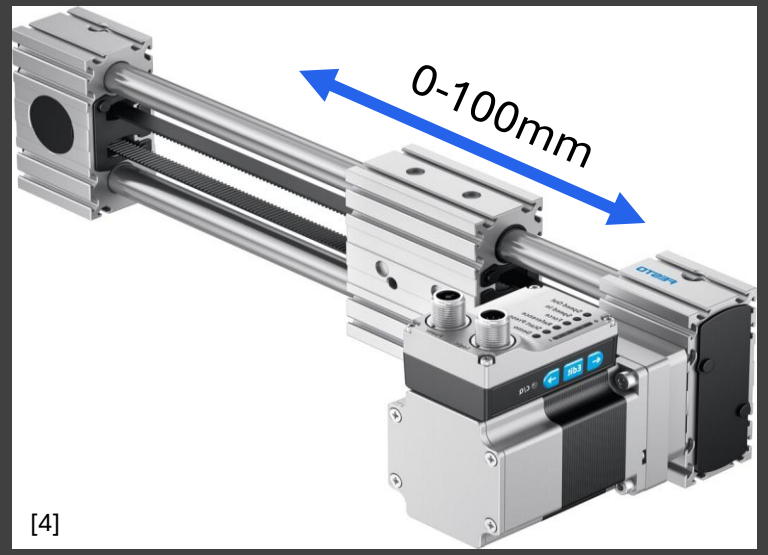


Definition: Real-Time means that a system responds to events within a guaranteed time limit

It does not mean fast, it means deterministic (predictable timing)

In automation systems several processes require deterministic commanding of values

- Control loops
- Movement of axis (single or multiple)

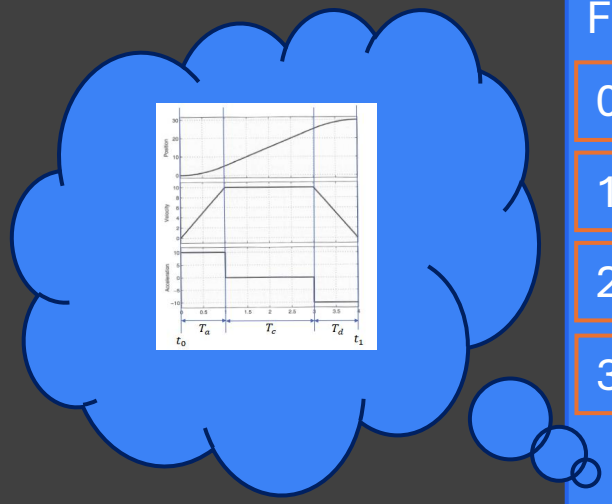


[4]

Example: axis positioning from 0mm to 100mm in 5 seconds

Let‘s move a motor

1. Switch motor-control-unit into ready state
2. Calculating the Trajectory in the PLC → Start 0mm End-position 100mm
3. Send one trajectory position after another in each PLC-Cycle via the Ethernet Fieldbus interface to the motor-control-unit
4. The motor follows this trajectory and moves → until there is no change in position anymore (Standstill)

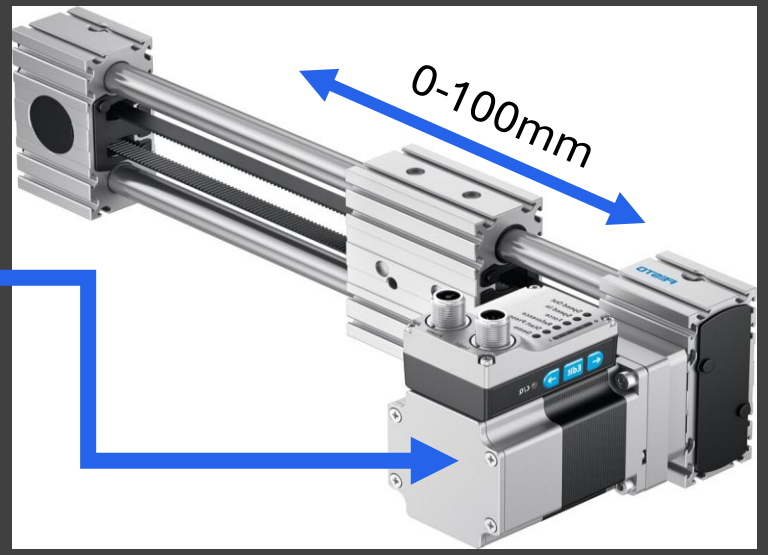


Fieldbus Master

- 0: 0.000mm
- 1: 0.001mm
- 2: 0.004mm
- 3: 0.013mm
- ⋮
- 500: 100.000mm

PLC

Motor with Linear Axis



Let‘s move a motor

Acceleration Phase

$$\begin{cases} q(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2 \\ \dot{q}(t) = a_1 + 2 \cdot a_2 \cdot t \\ \ddot{q}(t) = 2 \cdot a_2 \end{cases}$$

$$\begin{cases} a_0 = q_0 \\ a_1 = 0 \\ a_2 = \dfrac{v_v}{2 \cdot T_A} \end{cases}$$

$$\ddot{q}(t) = 2 \cdot a_2$$
$$\dot{q}(t) = 2 \cdot a_2 \cdot t + a_1$$
$$q(t) = a_2 \cdot t^2 + a_1 \cdot t + a_0$$

1) $q(t = 0) = q_0$
2) $\dot{q}(t = T_A) = v_v$
3) $\dot{q}(t = 0) = v_0 = 0$

3) $0 = a_1 + 2 \cdot a_2 \cdot 0$
$a_1 = 0$
2) $v_v = 0 + 2 \cdot a_2 \cdot T_A$
$a_2 = \dfrac{v_v}{2 \cdot T_A}$
1) $q_0 = a_0 + 0 \cdot t + \frac{v_v}{2 \cdot T_A}$
$a_0 = q_0$

Constant Velocity Phase

$$\begin{cases} q(t) = b_0 + b_1 \cdot t \\ \dot{q}(t) = b_1 \\ \ddot{q}(t) = 0 \end{cases}$$

$$b_1 = v_v$$

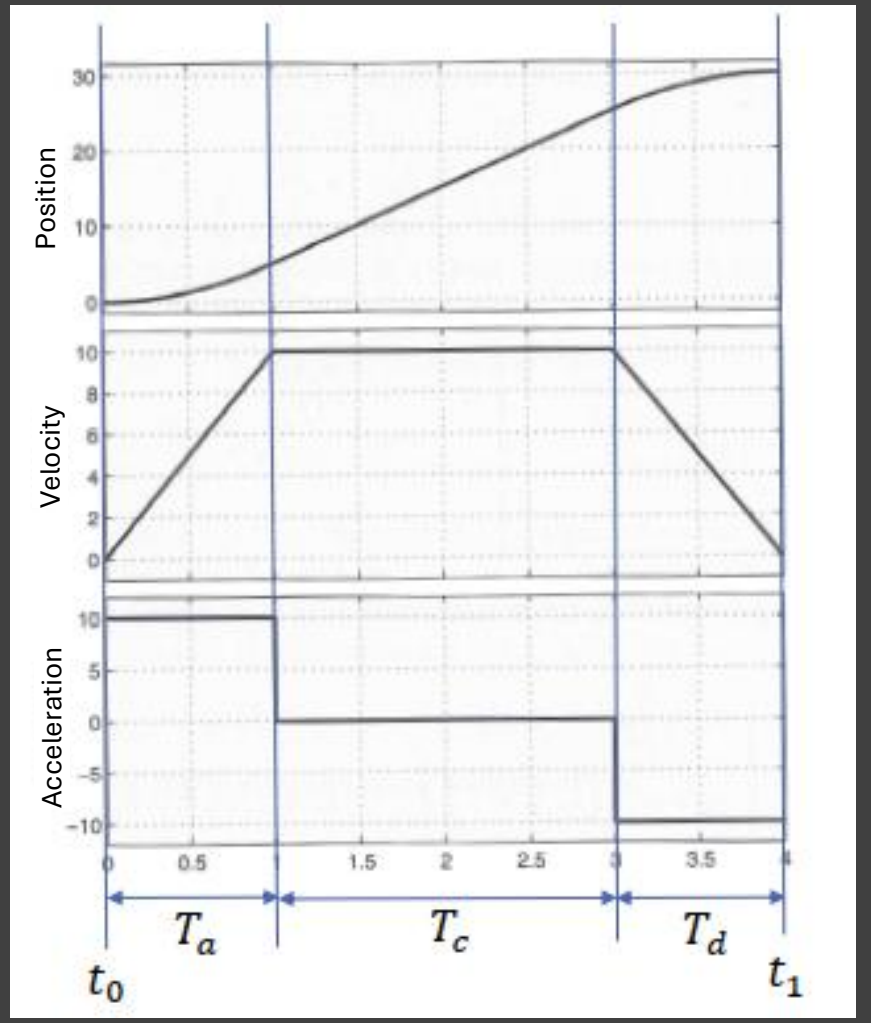
$$q(T_a) = q_0 + \frac{v_v \cdot T_a}{2} = b_0 + v_v \cdot T_a$$

$$b_0 = q_0 - \frac{v_v \cdot T_a}{2}$$

$$\ddot{q}(t) = 0$$
$$\dot{q}(t) = b_1$$
$$q(t) = b_0 + b_1 \cdot t$$

$$\dot{q}(t = T_A) = v_v$$
$$q(t = T_A) = q_0 + \frac{v_v \cdot T_A}{2}$$

$$b_1 = v_v$$
$$q(T_A) = q_0 + \frac{v_v \cdot T_A}{2} - v_v \cdot T_A = b_0 + v_v \cdot T_A$$
$$b_0 = q_0 - \frac{v_v \cdot T_A}{2}$$



[10]

Let‘s move a motor

Deceleration Phase

$$\begin{cases} q(t) = c_0 + c_1 \cdot t + c_2 \cdot t^2 \\ \dot{q}(t) = c_1 + 2 \cdot c_2 \cdot t \\ \ddot{q}(t) = 2 \cdot c_2 \end{cases}$$

$$\begin{cases} c_0 = q_0 - \dfrac{v_v \cdot {t_1}^2}{2 \cdot T_a} \\ c_1 = \dfrac{v_v \cdot t_1}{T_a} \\ c_2 = -\dfrac{v_v}{2 \cdot T_a} \end{cases}$$

$$q(t) = c_2 \cdot t^2 + c_1 \cdot t + c_0$$
$$\dot{q}(t) = 2 \cdot c_2 \cdot t + c_1$$
$$\ddot{q}(t) = 2 \cdot c_2$$

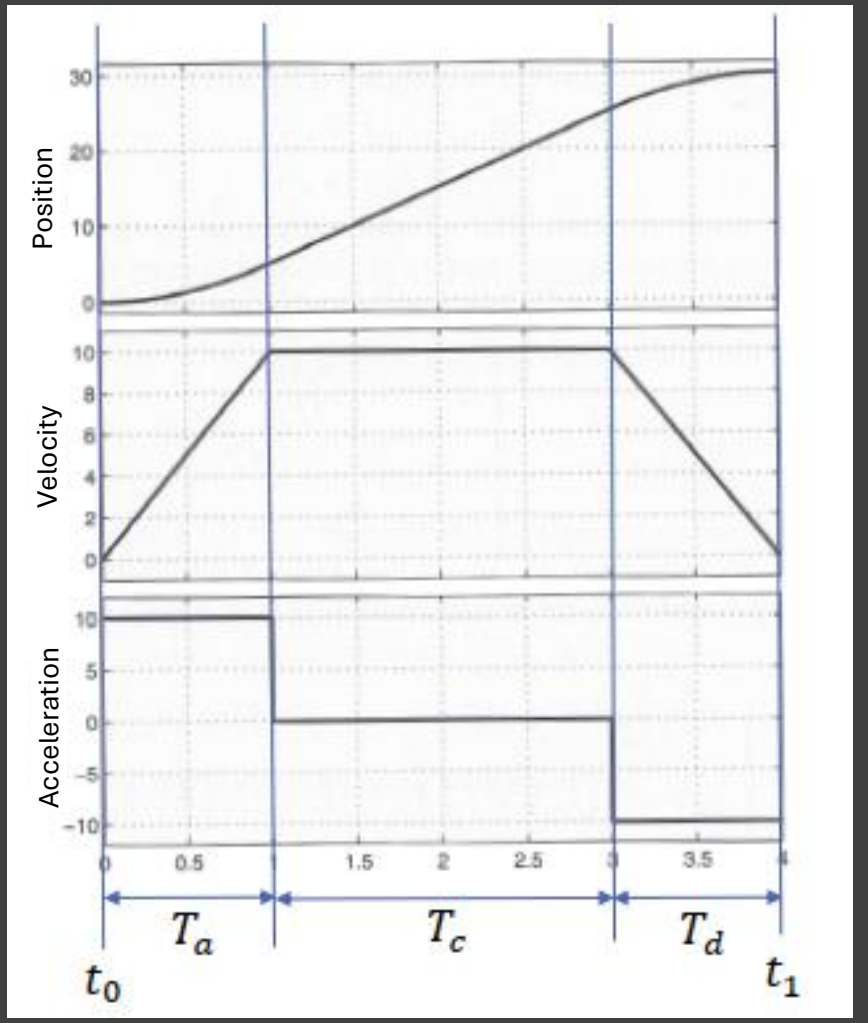
1) $\dot{q}(t = t_1) = 0$
2) $\dot{q}(t = t_1 - T_A) = v_v$
3) $q(t = t_1) = q_1$

1) $0 = c_1 + 2 \cdot c_2 \cdot t_1$
$c_1 = -2 \cdot c_2 \cdot t_1$

2) $v_v = -2 \cdot c_2 \cdot t_1 + 2 \cdot c_2 \cdot (t_1 - T_A)$
$v_v = -2 \cdot c_2 \cdot t_1 + 2 \cdot c_2 \cdot t_1 - 2 \cdot c_2 \cdot T_A$
$v_v = c_2 \cdot (-2 \cdot T_A)$
$c_2 = -\dfrac{v_v}{2 \cdot T_A}$

$c_1 = -2 \cdot \left(-\dfrac{v_v}{2 \cdot T_A}\right) \cdot t_1$
$c_1 = \dfrac{v_v \cdot t_1}{T_A}$

3) $q_1 = c_0 + \frac{v_v \cdot t_1}{T_A} \cdot t_1 + \left(-\frac{v_v}{2 \cdot T_A}\right) \cdot {t_1}^2$
$q_1 = c_0 + \dfrac{v_v \cdot {t_1}^2}{T_A} - \dfrac{v_v \cdot {t_1}^2}{2 \cdot T_A}$
$q_1 = c_0 + \dfrac{v_v \cdot {t_1}^2}{2 \cdot T_A} \rightarrow c_0 = q_1 - \dfrac{v_v \cdot {t_1}^2}{2 \cdot T_A}$

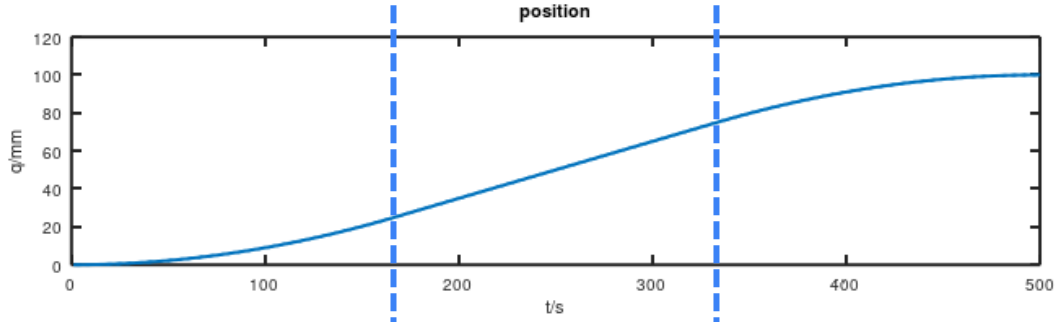


Let‘s move a motor

Calculating those equations with Octave

```
clear;
clc;

q0 = 0;     % start position in millimeter
q1 = 100;   % end position in millimeter
ts = 0.01; % sample time in seconds
t1 = 5;     % time for movement in seconds

[q, qp, qpp] = linearTrajectoryWithParabolicBlendsV2( 0, 100, 0.01, 4 );

subplot(3,1,1);
plot(q);
title('position');
xlabel('t/s');
ylabel('q/mm');
subplot(3,1,2);
plot(qp);
title('speed');
xlabel('t/s');
ylabel('qp/mm/s');
subplot(3,1,3);
plot(qpp);
title('acceleration');
xlabel('t/s');
ylabel('qpp/mm/s²');
```

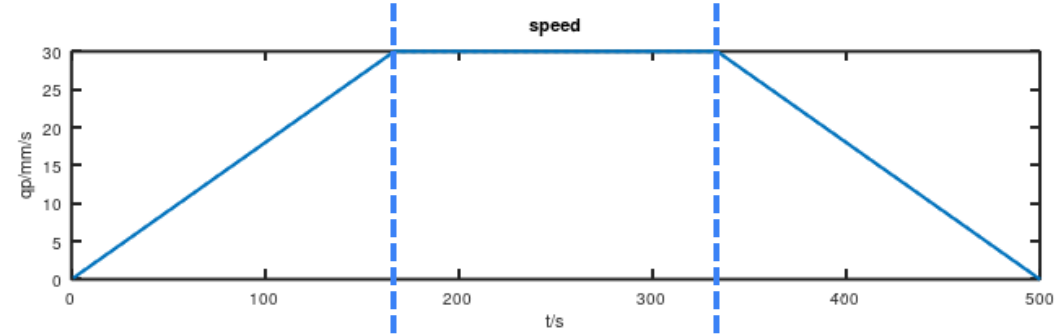
```
% Trajectory Generation Trapezoid/3
% Inputs: q0 Startposition, q1 Endposition,
%         ts Sample time (for interpolation use)
%         t1 time for movement
% Outputs: q Position
%          qp Velocity
%          qpp Acceleration
function [q, qp, qpp] = linearTrajectoryWithParabolicBlendsV2( q0, q1, ts, t1 )
  % initial steps
  steps = t1/ts;

  q=zeros(1,steps);
  qp=zeros(1,steps);
  qpp=zeros(1,steps);

  t=0;
  t0=0;

  % calculation of time for each phase
  Ta=(t1-t0)/3;
  Td=Ta;
  Tc=Ta;

  % calculation of velocity and acceleration
  vv=3*(q1-q0)/(2*(t1-t0));
  a=9*(q1-q0)/(2*(t1-t0)^2);

  % looping till end position is reached
  for i=1:steps
    t=t+ts;
    if( t>=t0 && t<=(t0+Ta) )
      q(i)=q0+0*t+vv/(2*Ta)*t^2;
      qp(i)=0+2*vv/(2*Ta)*t;
      qpp(i)=2*vv/(2*Ta);
    end

    if( t>(t0+Ta) && t<=(t0+Ta+Tc) )
      q(i)=q0-(vv*Ta)/2+vv*t;
      qp(i)=vv;
      qpp(i)=0;
    end

    if( t>(t0+Ta+Tc) && t<=(t0+Ta+Tc+Td) )
      q(i)=q1-(vv*t1^2)/(2*Td)+(vv*t1)/Td*t+(-vv/(2*Td))*t^2;
      qp(i)=(vv*t1)/Td+2*(-vv/(2*Td))*t;
      qpp(i)=2*(-vv/(2*Td));
    end
  end
end
```

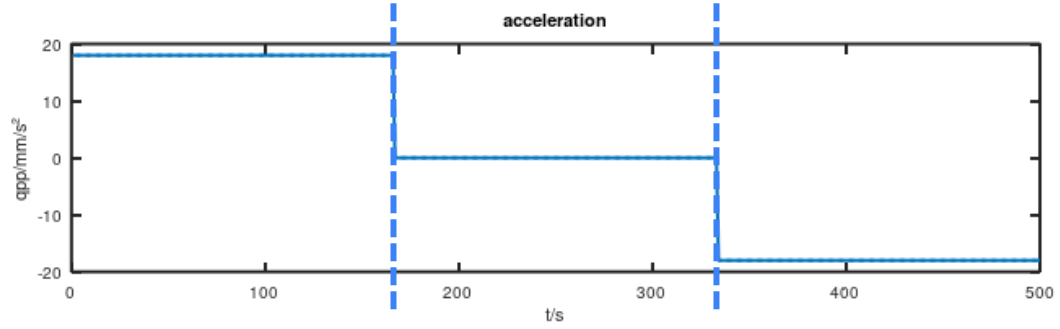


Acceleration

Constant Velocity

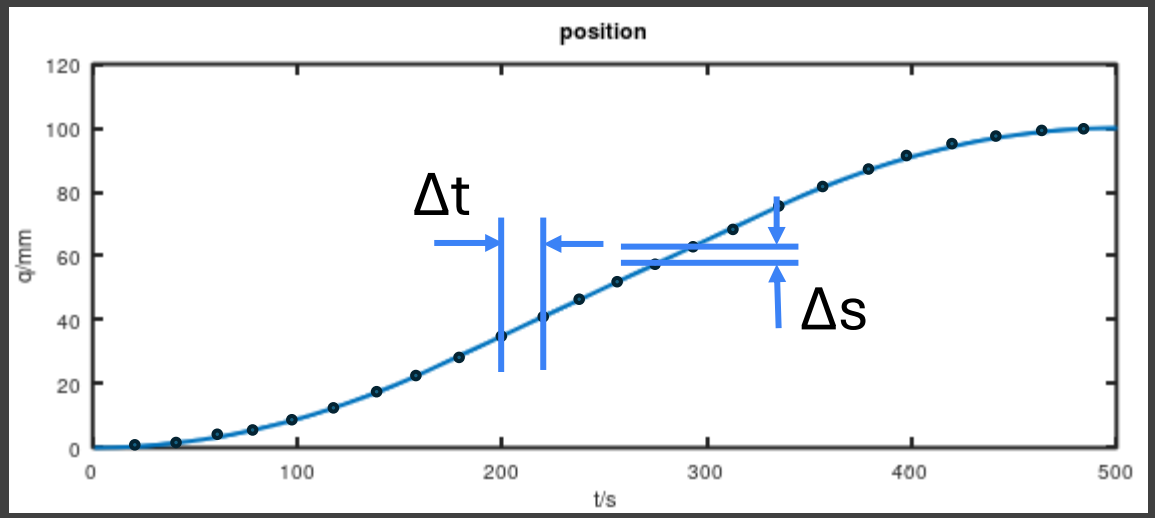
Deceleration





Generated Trajectory:

- Distance: 0 → 100mm
- Cycle-Time: 0.01 sec
- Ramp-Time: 5 sec
- Generated points: 500



Inconsistent command timing causes jerky servo motion.

Even one missed command value triggers an error on the drive

Why is RealTime important - Examples?

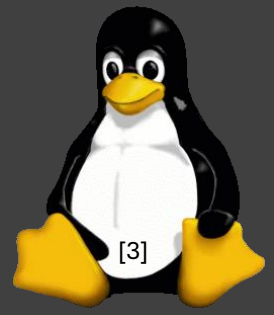
- Trumpf TruLaser Series Bevel Cut https://www.youtube.com/watch?v=yFqm-QqgdeE
- Schubert Packaging Pralines https://www.youtube.com/watch?v=I4FL2EbGhaE
- CNC Machine Turbo charger wheel: https://www.youtube.com/watch?v=zB5YumxzQHs
- Schubert Packaging (cardboard box) https://www.youtube.com/watch?v=DTCMNqPRciI&t=19s

Make vs. Buy

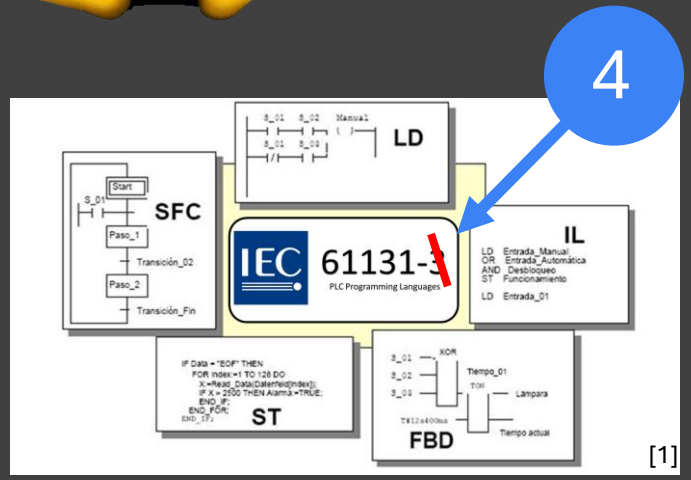
Why not just use a Raspberry Pi, instead pay $$ for Siemens Beckhoff control systems?

- **Environment:** EMC (electrical Noise, Vibrations, Temperature ranges)
- **Lifecycle:** The „10+ Year Rule“ Parts need to be available at least 10 years to buy replaceable parts
- **Liability / Safety**: Certified Hardware (SIL/TÜV), There has to be a guarantee that an emergency stop works → life saving
- **The "Batteries Included" Software Stack:** A PLC isn't just hardware; it's a specially tuned Real-Time Runtime + Middleware. You don't have to write low-level C-drivers for I/O cards, motor controllers, or complex fieldbus protocols (like PROFINET or EtherCAT)
  → just concentrate on your process problems to solve

Summary & Outlook


[3]

- The worlds IT and OT are merging
- Trend towards SoftPLCs with OS like Linux & BSD
- Modern programming languages, modern paradigms in IEC based languages
- Modern development paradigms are coming, Unit-Testing, CI/CD
- Slowly but steady → LLMs are entering the room



[1]

[2]

Thank you – Any Questions?

Automation engineering connects code, hardware and real processes.

Matthias Seehauser


Zeugwerk GmbH - zeugwerk.at


linkedin.com/in/seesle/


bsky.app/profile/seehma.at

Resources and Links

- [1] https://www.automationreadypanels.com/plc-systems/iec-61131-standard-for-industrial-automation-programming/?srsltid=AfmBOorLMPwZata6YshM3ZgiDHbL0Ro-FXUFAmmhVZo6W668ikZ8T_PM
- [2] https://timesofindia.indiatimes.com/technology/tech-news/explained-what-is-anthropics-ai-tool-that-wiped-285-billion-off-software-stocks-in-a-single-day/articleshow/127892310.cms
- [3] https://de.wikipedia.org/wiki/Linux
- [4] https://itg-motor.com/what-is-a-servo-motor-linear/
- [5] https://www.beckhoff.com/de-de/produkte/i-o/ethercat-klemmen/
- [6] https://www.wago.com/de/automatisierungstechnik/sps-entdecken/pfc300
- [7] https://store.boschrexroth.com/en/us/p/compact-converter-r911343260
- [8] https://store.boschrexroth.com/de/at/ms2n-performance?srsltid=AfmBOooJGEgnnvOEJ2yba7JnbLanfkTKzho2lninstZW8OfEwpcy3Wqu
- [9] https://www.reddit.com/r/cableporn/comments/c3hco3/55_year_old_elevator_relay_board_in_a_22_story/
- [10] Luiggi Biagiotti, Claudio Melchiorri; Trajectory Planning for Automatic Machines and Robots; Springer 2008
- [11] https://www.festo.com/de/de/p/zahnriemenachseneinheit-id_ELGE_TB/
- [12] https://www.schubert.group/de/magazin/zukunftsweisende-technologien-auf-der-anuga-foodtec-in-koeln/