School of Electrical Engineering

Department of Electrical Engineering and Automation

ELEC 8201 Control & Automation

Industrial Automation Software

Valeriy Vyatkin

Aalto University

2022

Course logistics

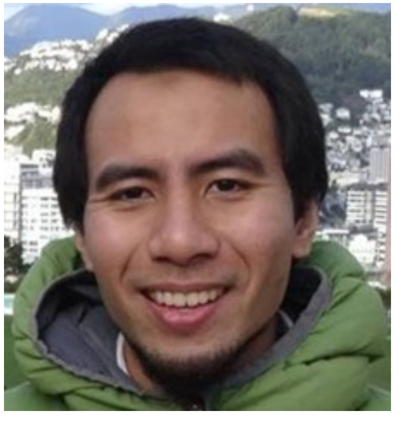
This part contributes to 1/3 of the mark

- 4 lectures
- 4 quizes
- 2 homeworks
- 2 exam questions

Course team

Prof. Valeriy Vyatkin

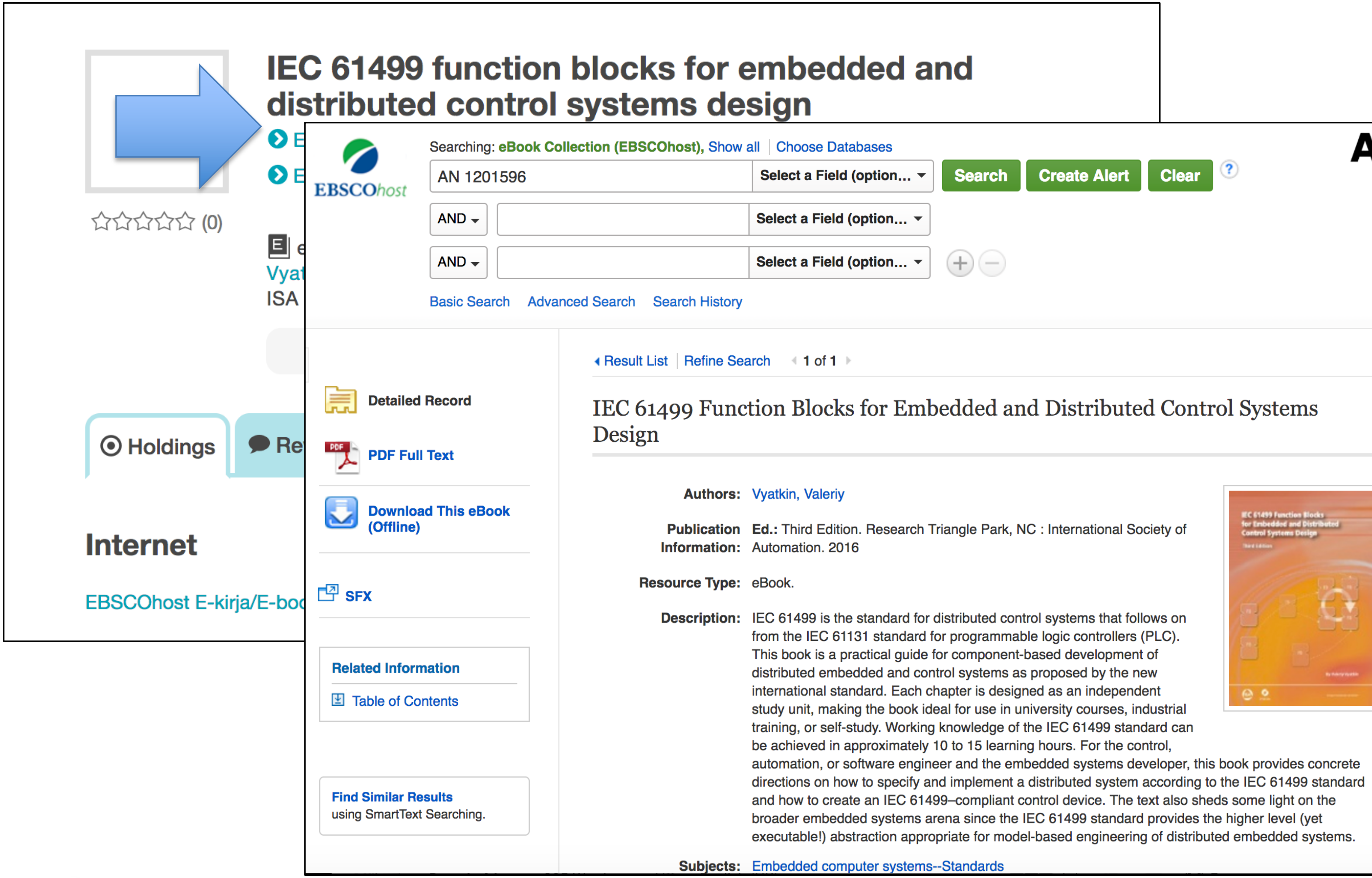
Dr. Udayanto Dwi Atmojo



Mr. Pranay Jhunjhunwala

Aalto University

Accessing the E-book



Tools

- We will be using NxtStudio
- Install it on your personal laptops, or use the virtual machine image
  - Refer to the **VDI Instructions** guide on MyCourses

Online course on IEC 61499 – free access

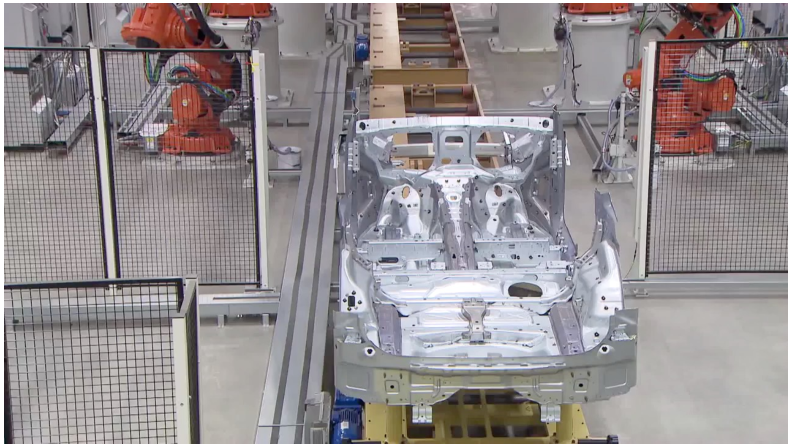
- Subscribe to the following YouTube channel https://www.youtube.com/channel/UCAXM7DKhEqJlo_zxOscz7rA and watch the introductory video.

- Enrol to this online study platform (for free) and go through the available material watching and repeating on your laptops.
  1. https://training.flexbridge.se/courses/basic-introduction-to-iec-61499?coupon=aalto-2022-spring
  2. https://training.flexbridge.se/courses/distributed-automation-programming-with-iec-61499-basic-introduction?coupon=aalto-2022-spring

Quizes in the Automation part

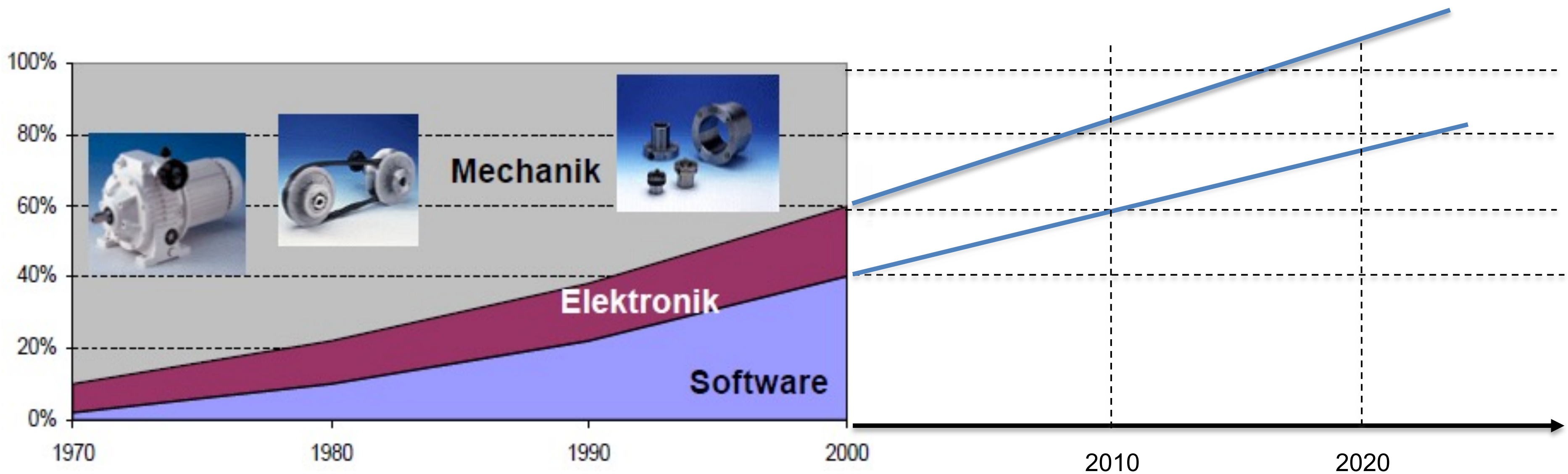
- Quizes are released after each lecture
- Submission deadline is in 24 hours

What is Industrial Automation?



What is Industrial Automation?

Software Ratio in Industrial Automation

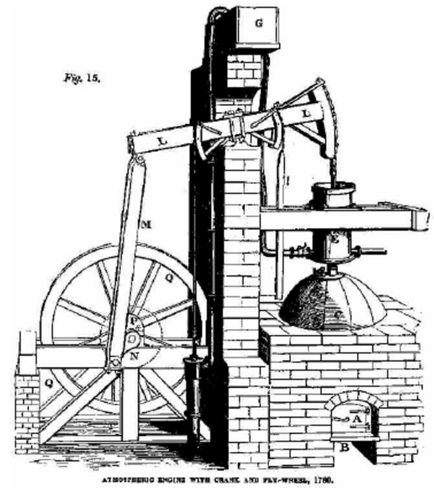


According to Verband Deutscher Maschinen- und Anlagenbau e.V. (VDMA)

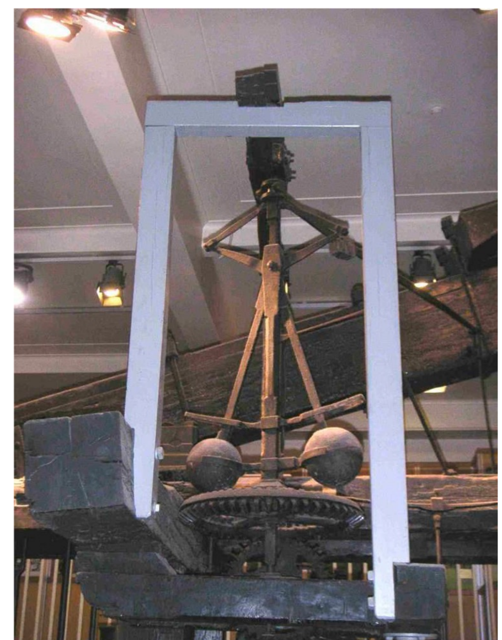
Generations of Automation Systems

1. Relay based controllers (40s – 50s)
2. Microprocessor based PLCs (70s)
3. Multifunctional PLCs
4. Industrial Networks
5. Internet of Things

Generation 0. Mechanical regulators

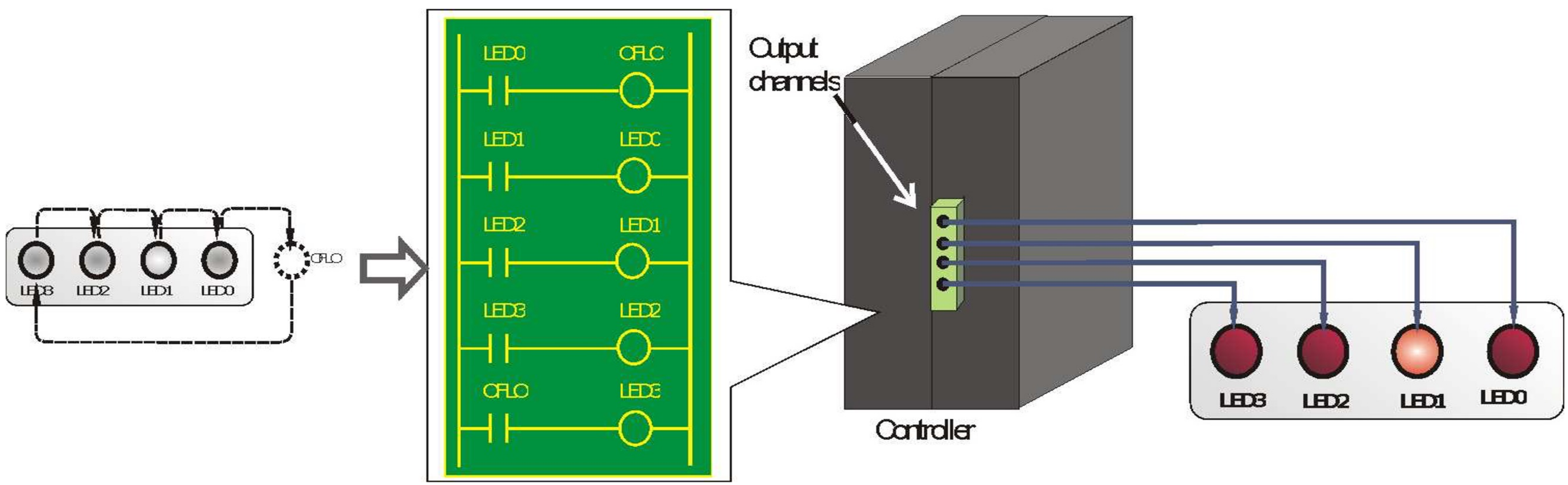


Mine in Cornwall equipped with first steam engine in 1790. First <u>industrial</u> regulator invented by James Watt to keep the wheel's speed constant.

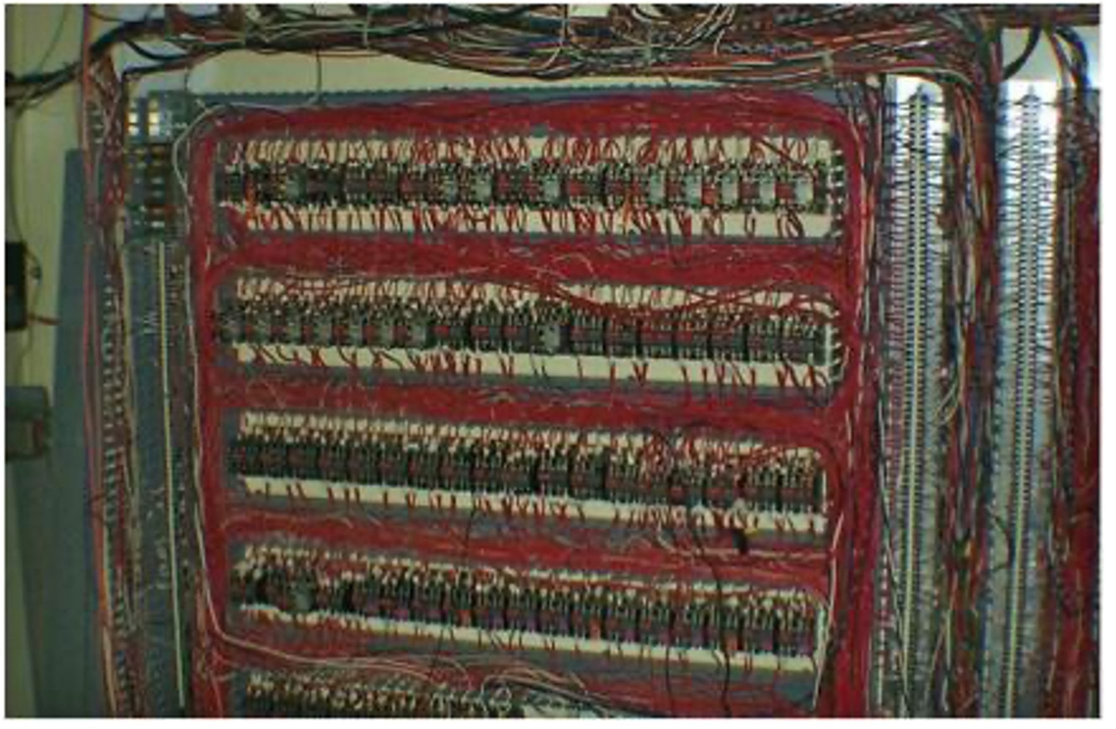


Generation 1: Relay Ladder Circuits

Hard-wired ladder logic circuits were widely used to control industrial equipment. This explains current popularity of the Ladder Diagram language for programming industrial controllers



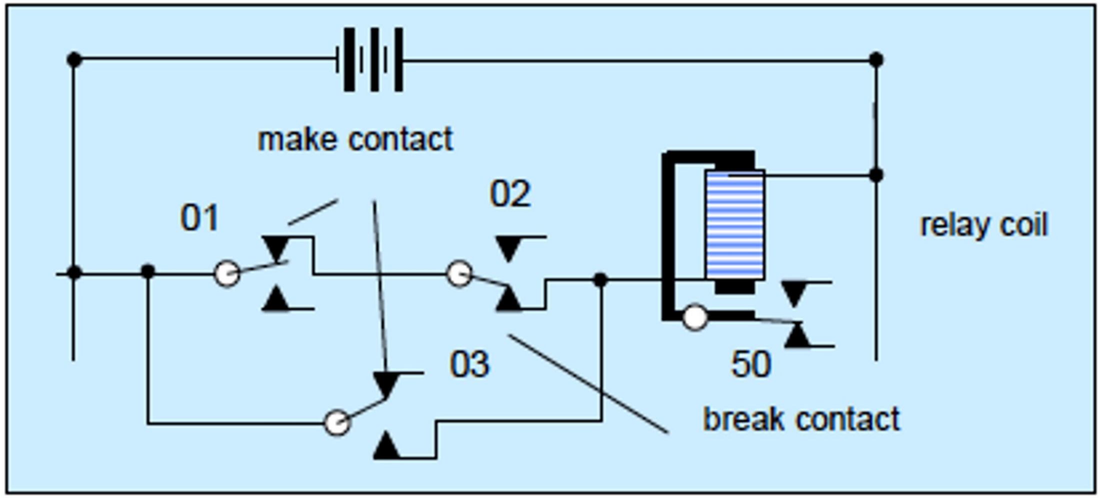
Relay Ladder Logic

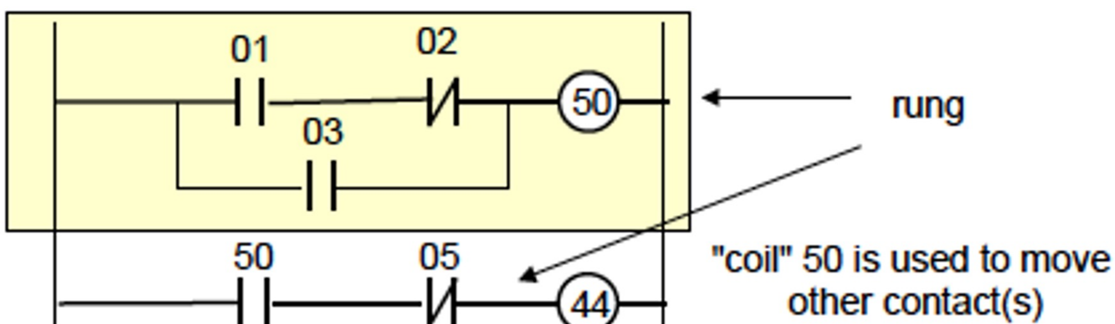


Control logic implemented as hard-wired **relays**

Aalto University

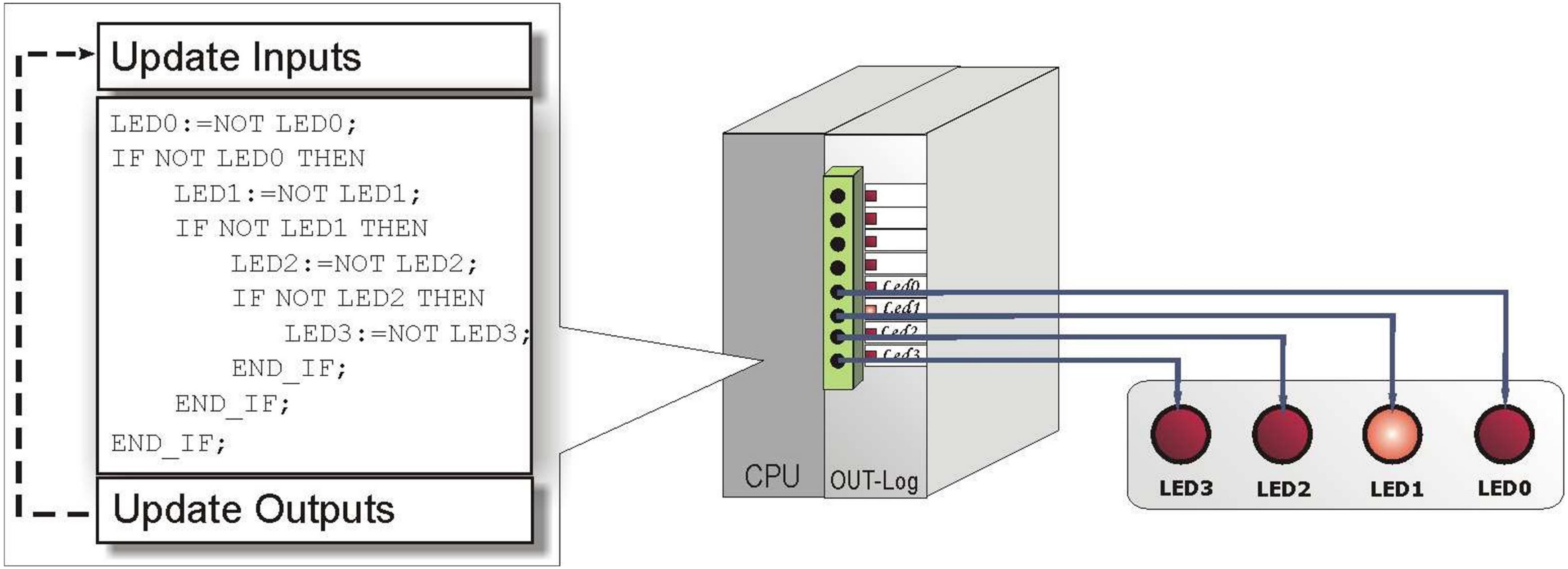
Electric Circuit and Ladder Diagram



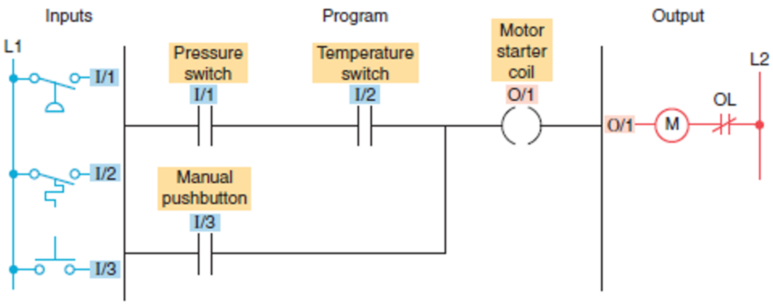


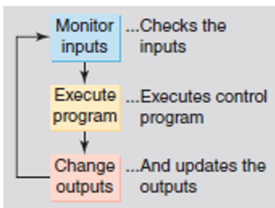
Generation 2: Programmable Logic Controllers (PLCs)

PLCs – specially hardened industrial computers - tremendously improved flexibility of automation systems

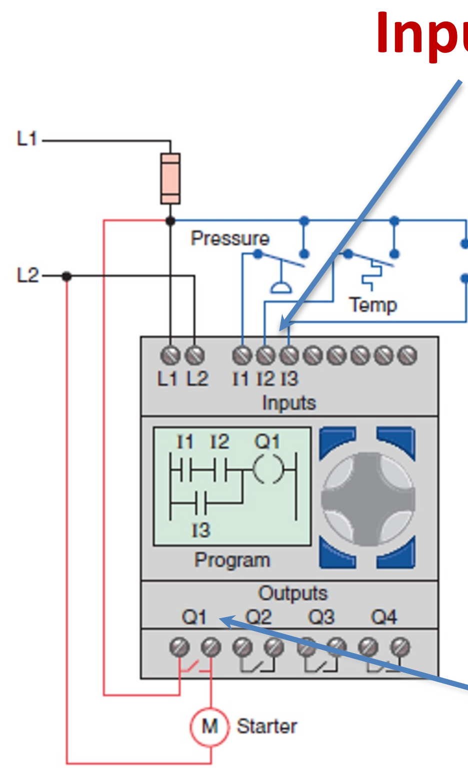


PLC Programming with Ladder Logic





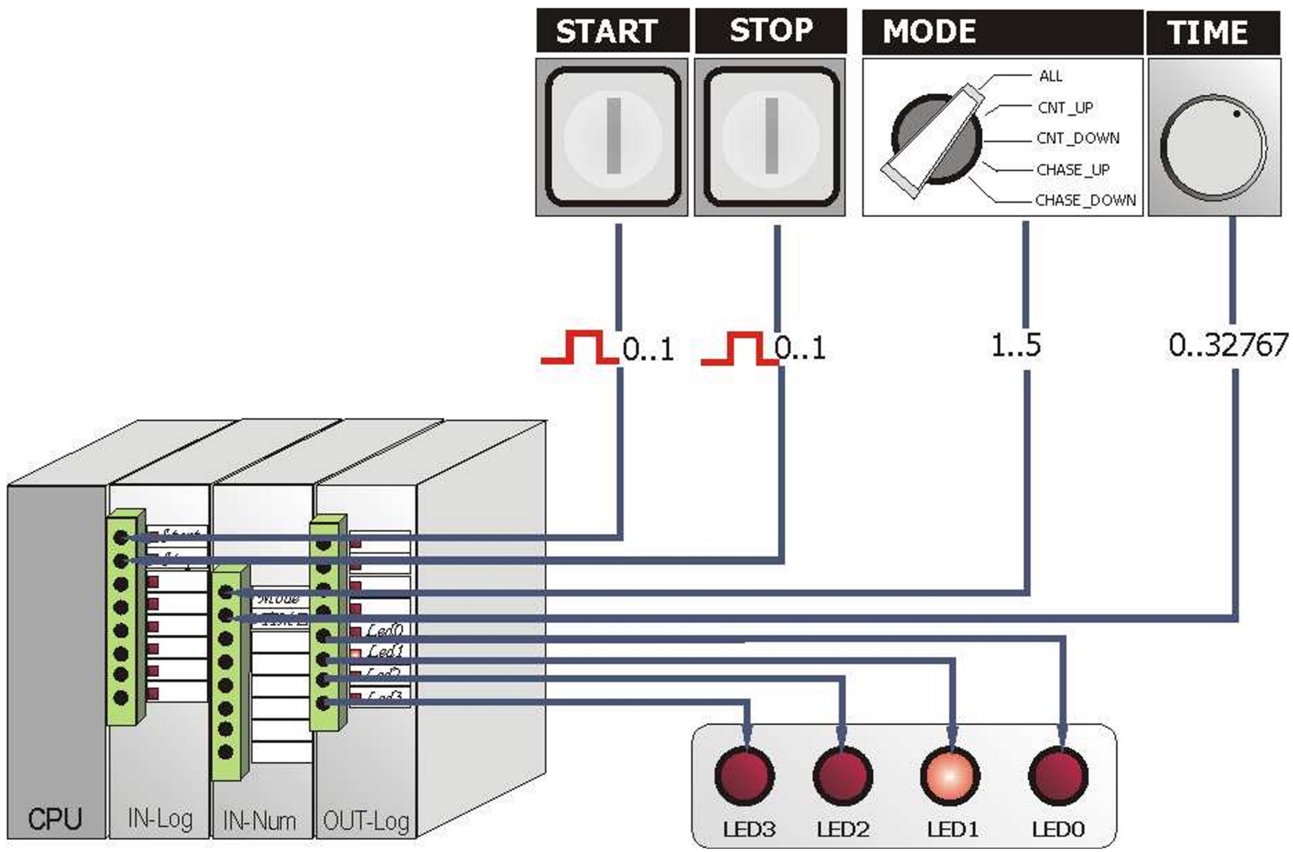
Process control PLC ladder logic program with typical addressing scheme.



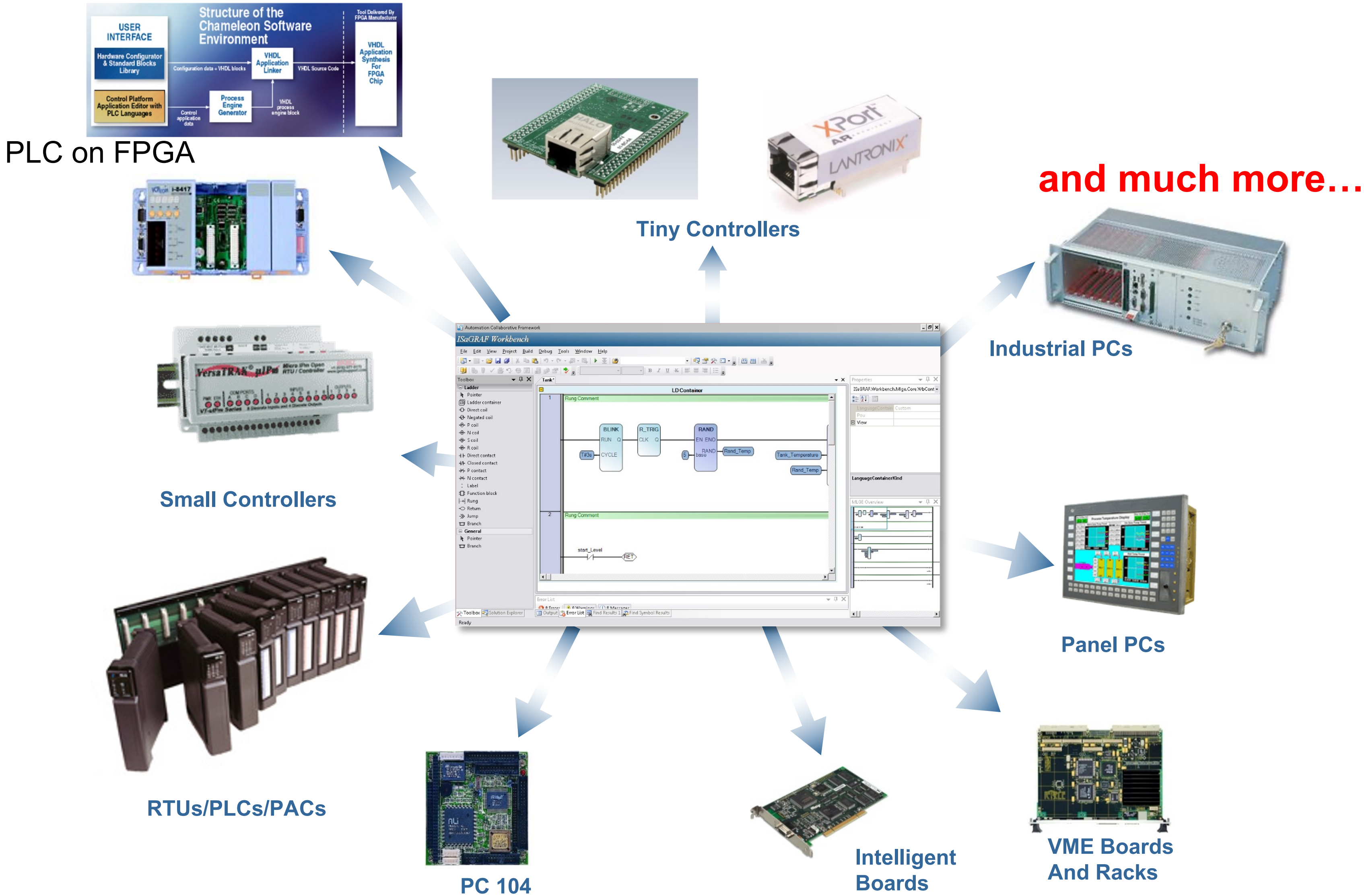
Typical wiring required to implement the process control scheme
using a fixed PLC controller.

Generation 3: Multifunctional PLCs

Modern PLCs know many programming languages and have versatile and easily expandable architecture.

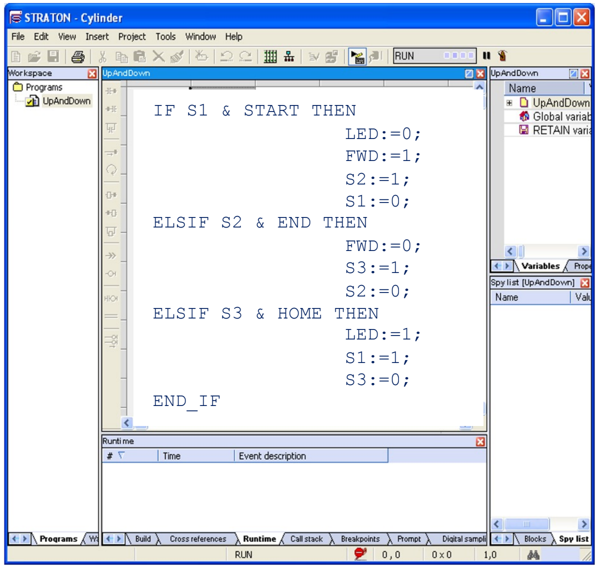


Programmable Logic Controllers: Form Factors

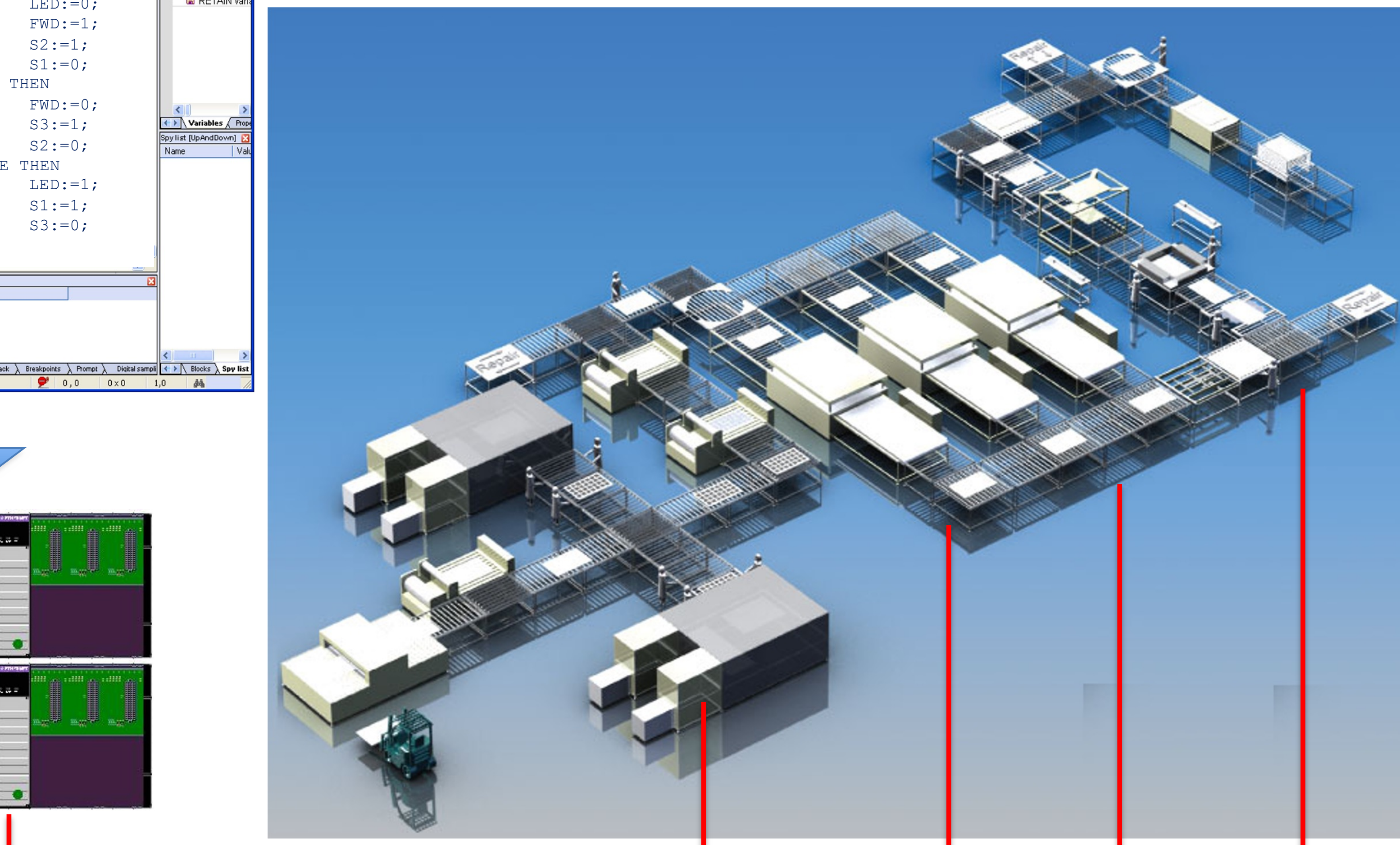


Aalto University

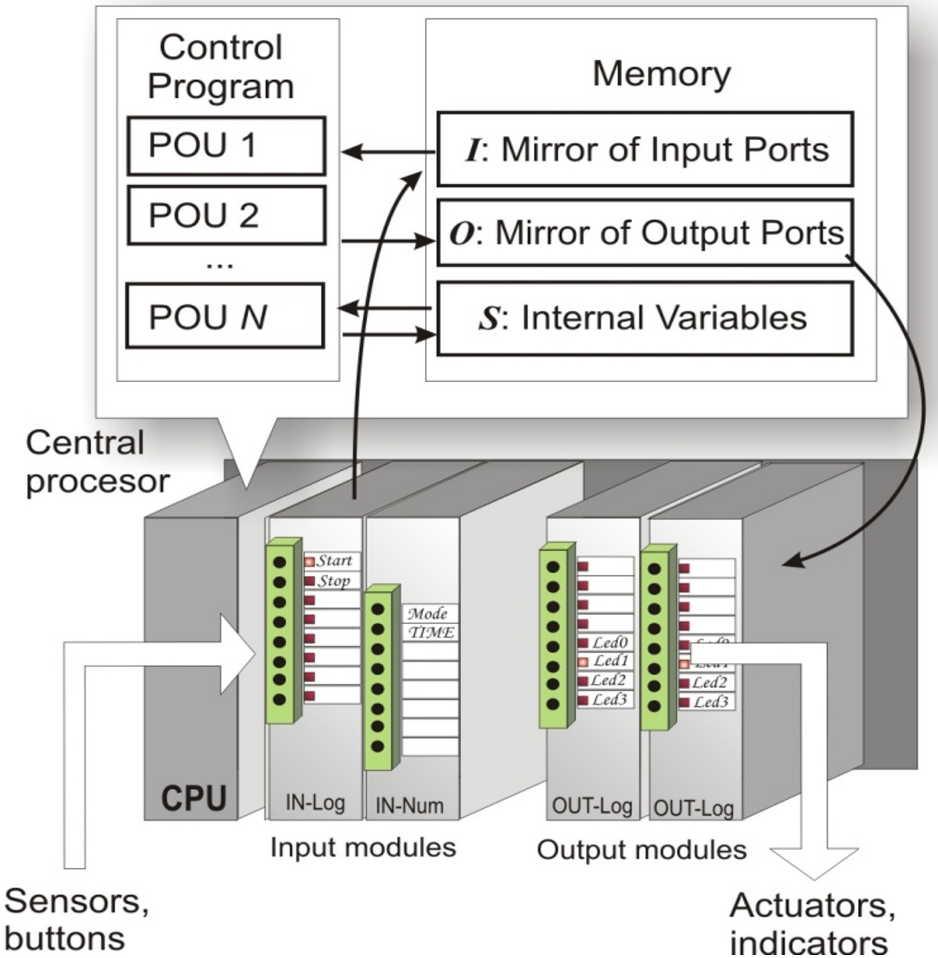
Generation 4: Industrial Networks

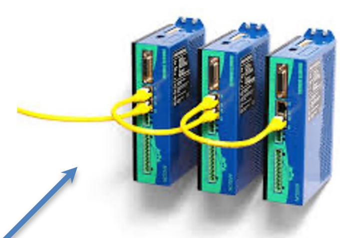


PLC

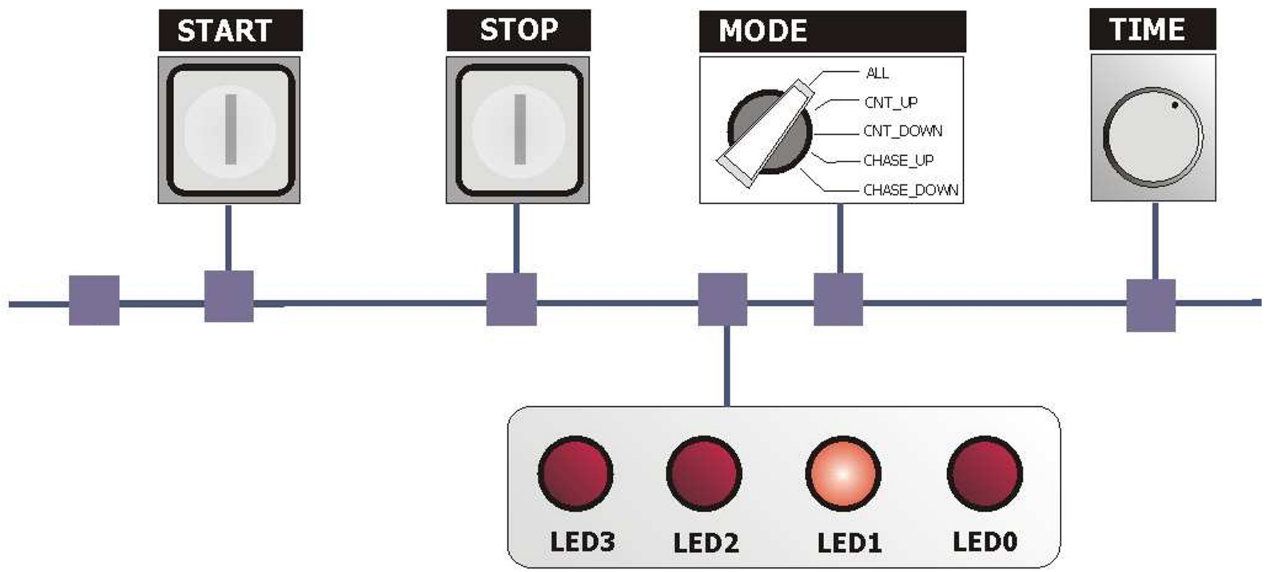


PLC as Integration Platform

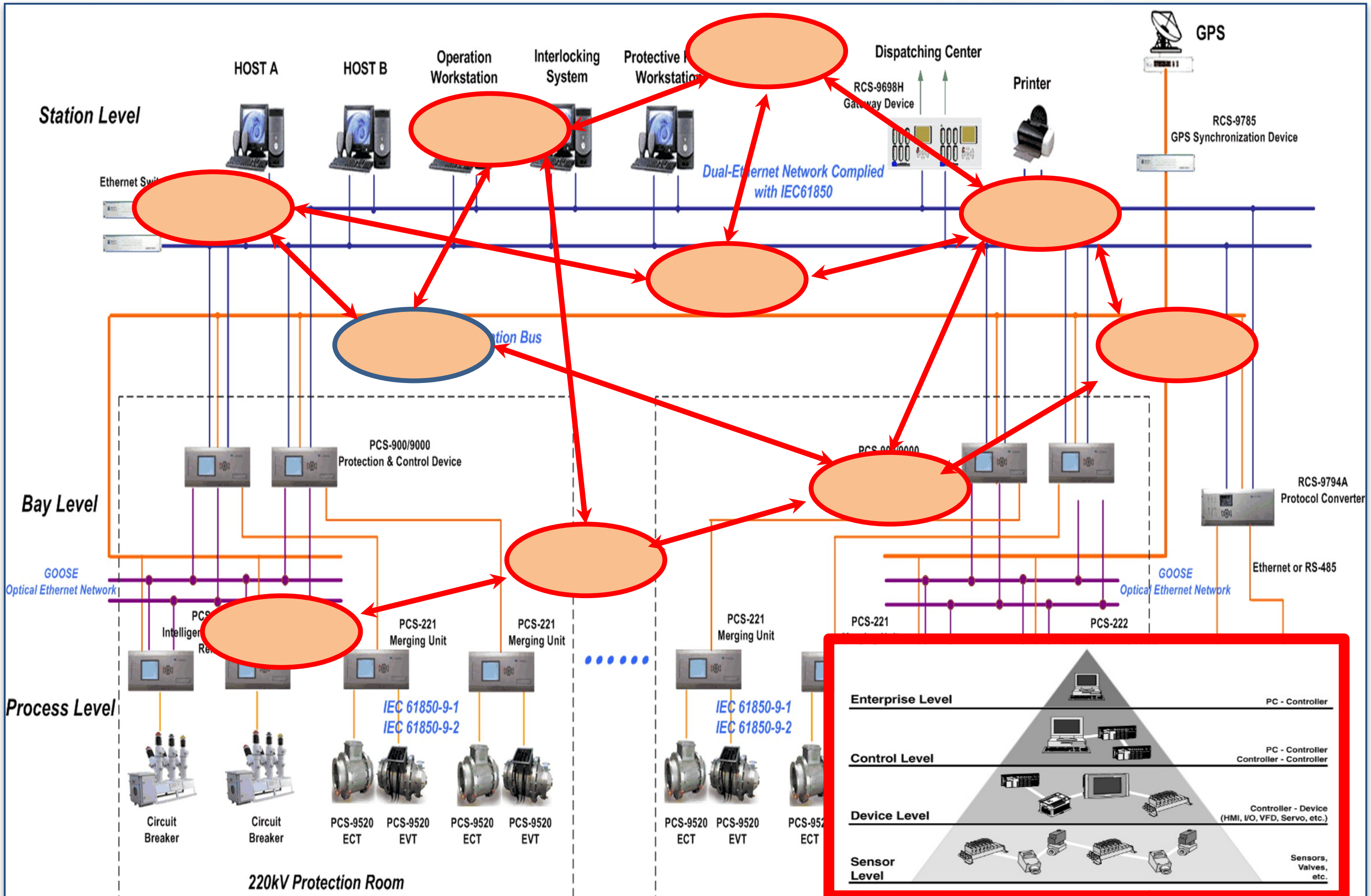




Generation 5: From Networking to Internet of Things

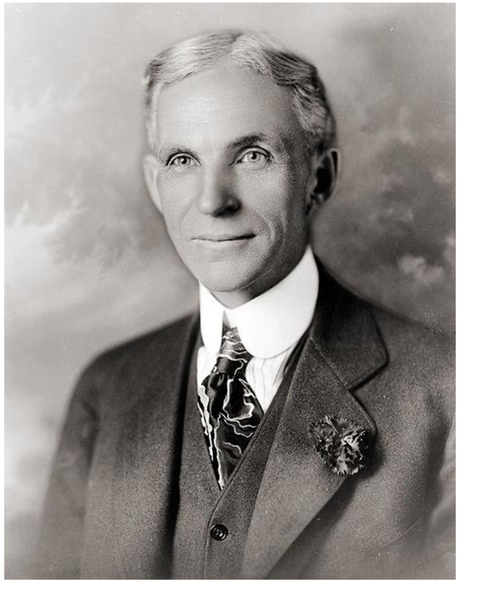


Machine To Machine Communication, Internet of Things



Production in the past…

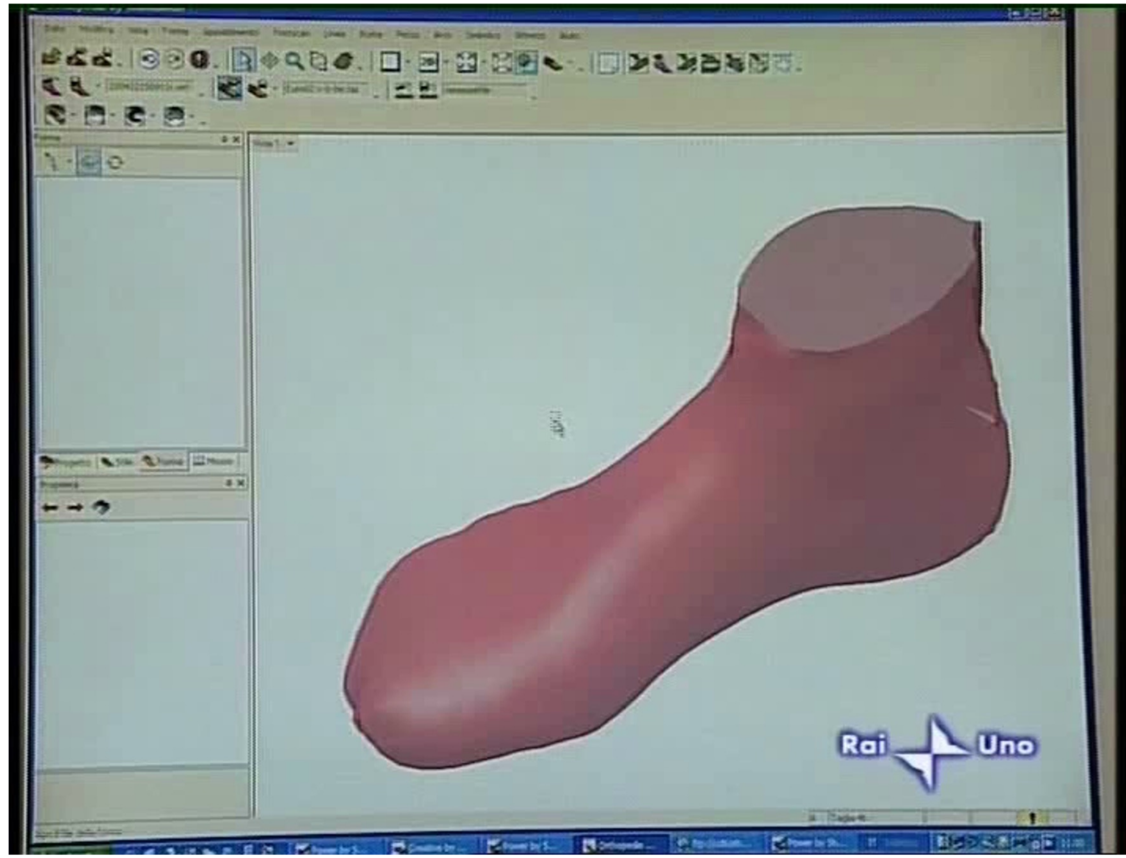
"Any customer can have a car painted any colour that he wants so long as it is black!"



Henry Ford (1863-1947)
Source of photo: Wikipedia

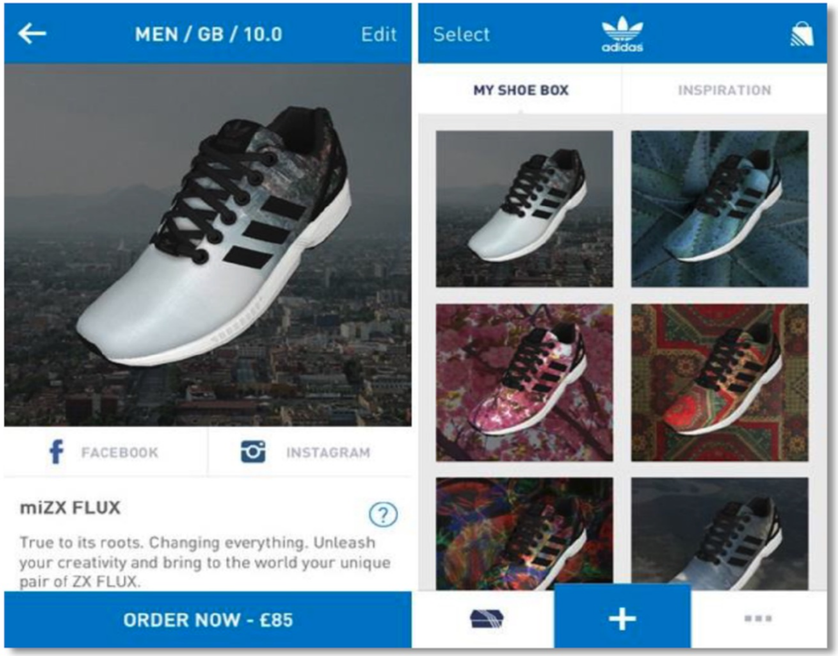
1926 Ford Model T
Source of photo: Boldride

XXI Century: Manufacturing to order!



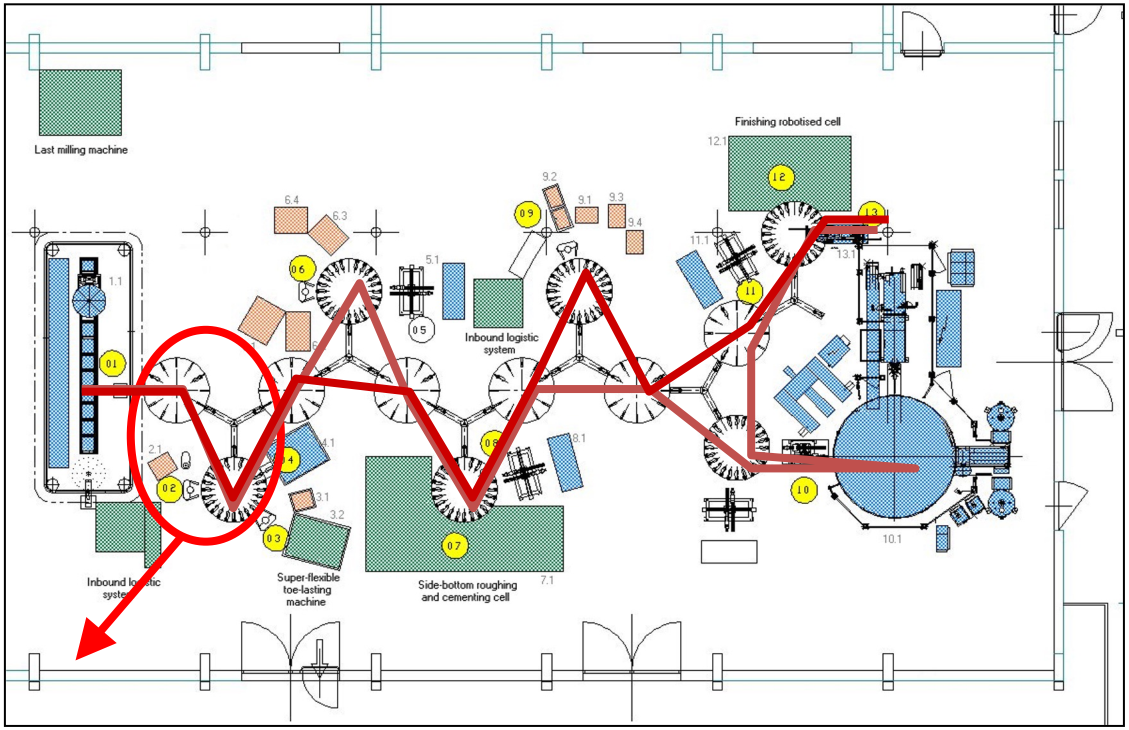
Aalto University

The Adidas Speedfactory: Bringing Sports Shoes Production back to Germany by Industrie 4.0 for Mass Customization



- The costumers can design their own short shoes using an App.
- Since the customer wants to receive his personalized product on the next day or faster, long logistic chains from low-wage countries are no longer acceptable in the era of mass customization.
- Thus, adidas decided to open various "speedfactories" for personlized shoes in Germany close to the customer, using Cyber-physical production systems (CPPS).

Flexible Manufacturing

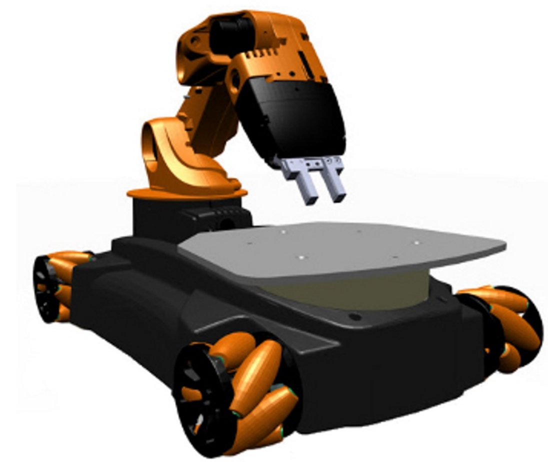


Industry 4.0: New Factory Floor

Industrial robots

3D printers

Mobile machines

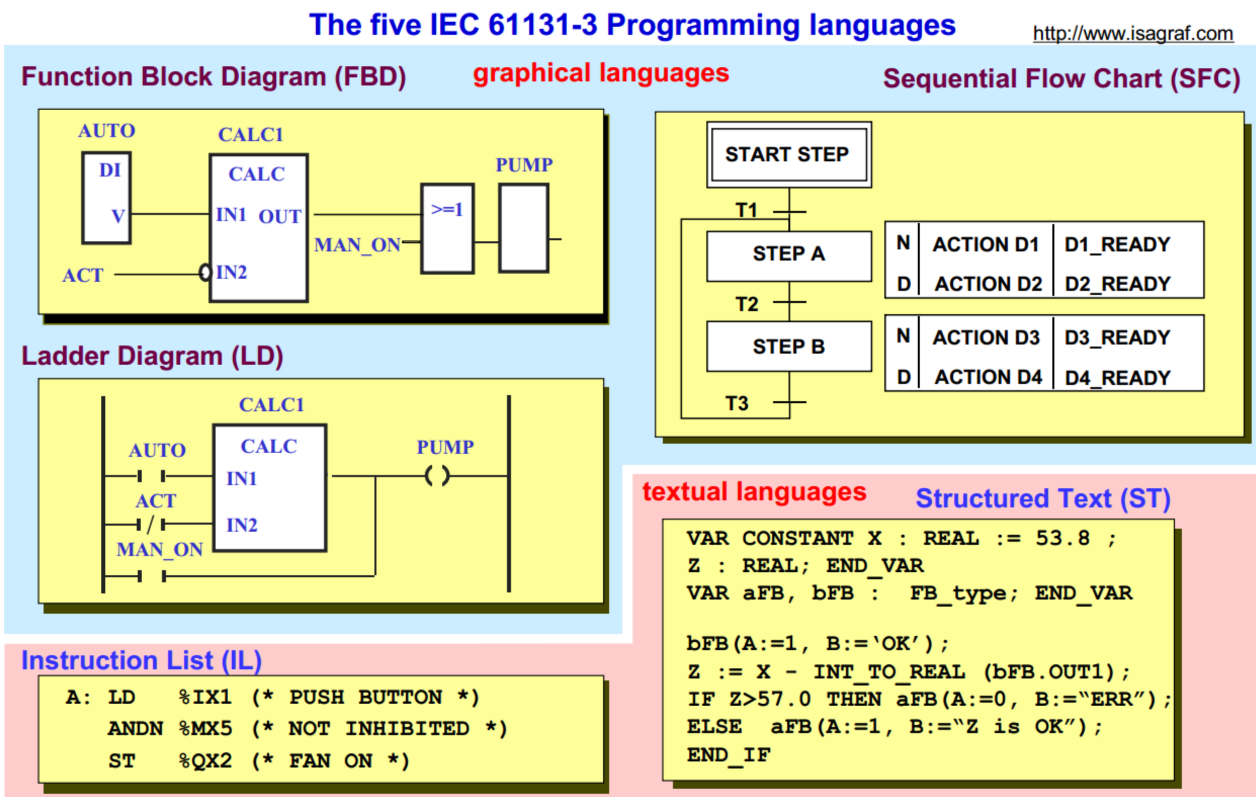


Aalto Factory of the Future at Scanautomatic Fair 2018

Aalto University

Computations in Controllers

IEC 61131-3 Programming Languages



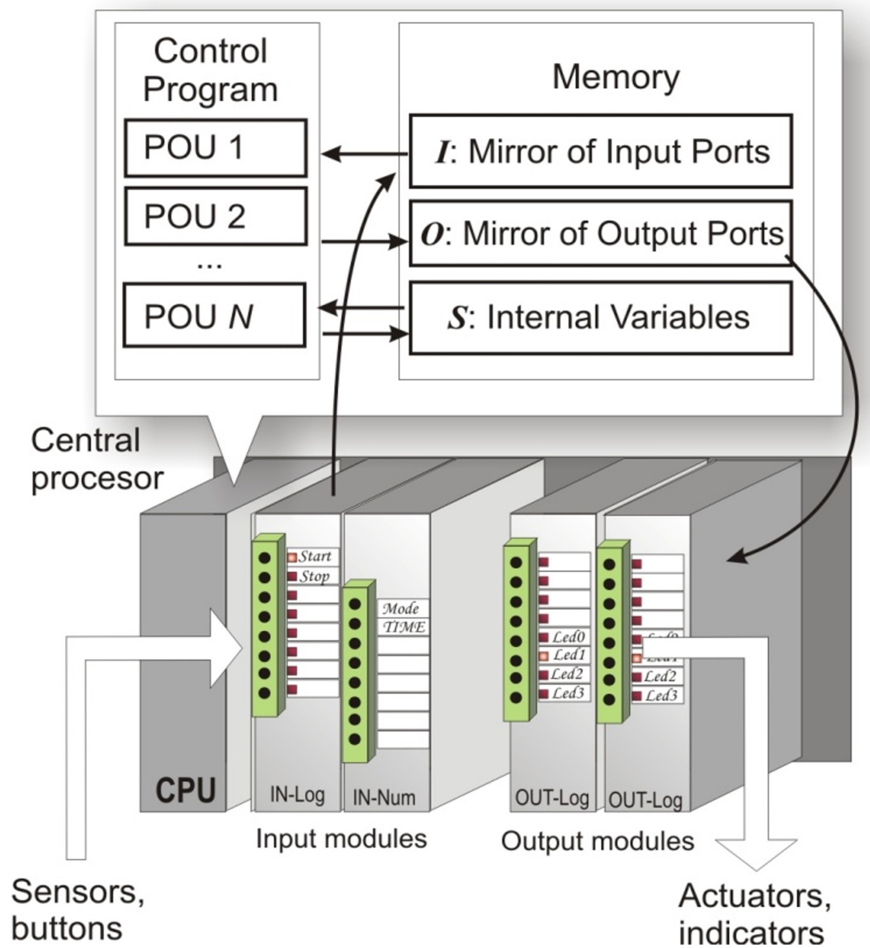
Software Tools

- *CoDeSys*
  - Software tool for developing and engineering IEC 61131-3 controller applications
  - a soft PLC from 3S-Smart Software Solutions GmbH
- *TwinCAT*
  - OEM version of CoDeSys for Beckhoff, under Visual Studio Shell
- *ISaGRAF*
- *KW*
- Tools of *SIEMENS, Rockwell, ABB*…

Key Features of IEC 61131-3

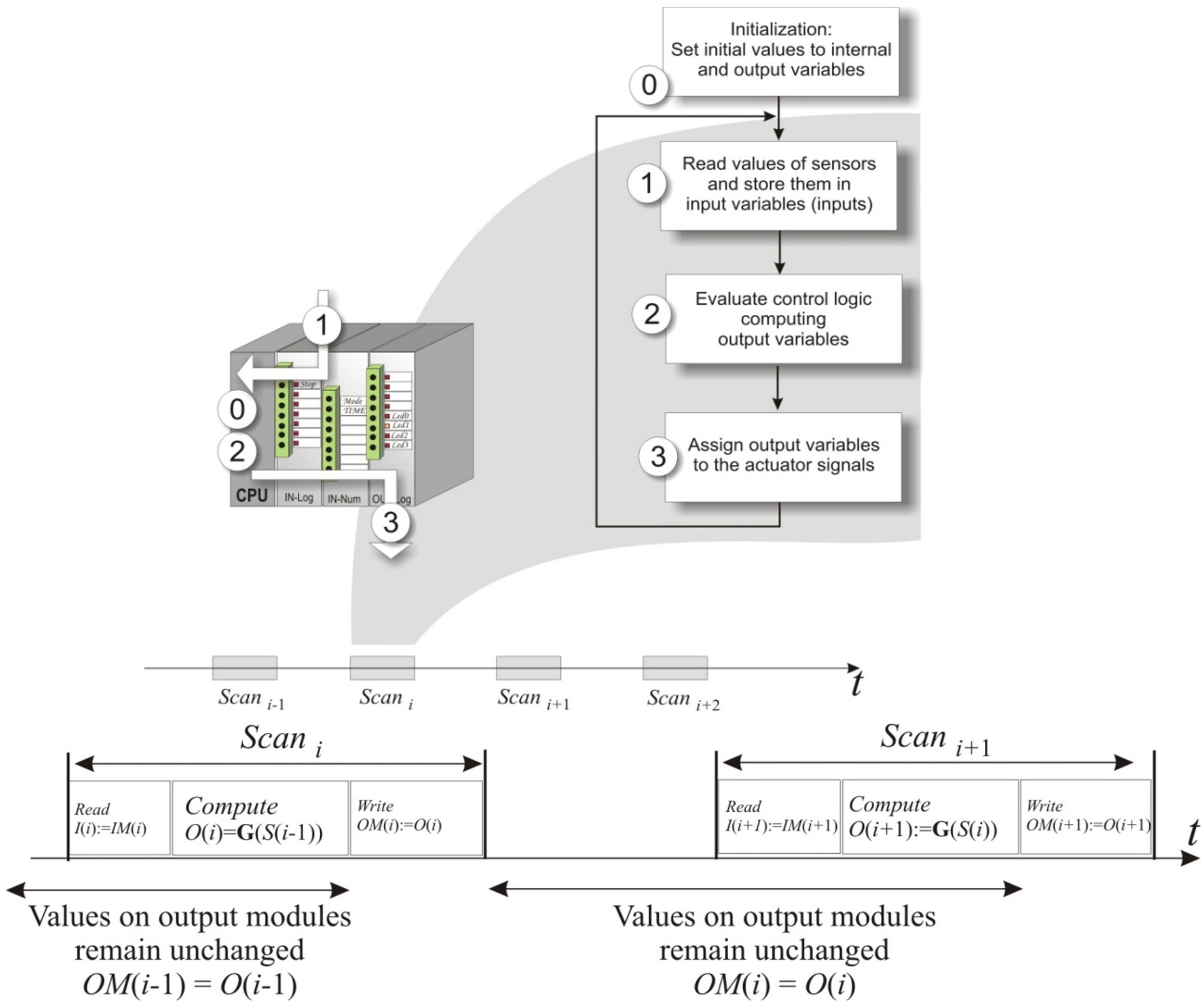
- IEC 61131-3 is the most important automation standard in industry.
- 80% of all PLCs support it, all new developments base on it. Depending on the country, some languages are more popular.
- Structured software - *through use of Configuration, Resource, and Program Organization Units (POU)*
- Software encapsulation - *through use of POU, and complex data types*
- Strong Data Typing - *through languages that restrict operations to only apply to appropriate types of data*
- Execution control - *through use of tasks*

PLC



Control Program
POU 1
POU 2
...
POU N
Memory
I: Mirror of Input Ports
O: Mirror of Output Ports
S: Internal Variables
Central procesor
Start
Stop
Mode
TIME
Led0
Led1
Led2
Led3
Led2
Led3
CPU
IN-Log
IN-Num
OUT-Log
OUT-Log
Input modules
Output modules
Sensors, buttons
Actuators, indicators

Cyclic Program Execution

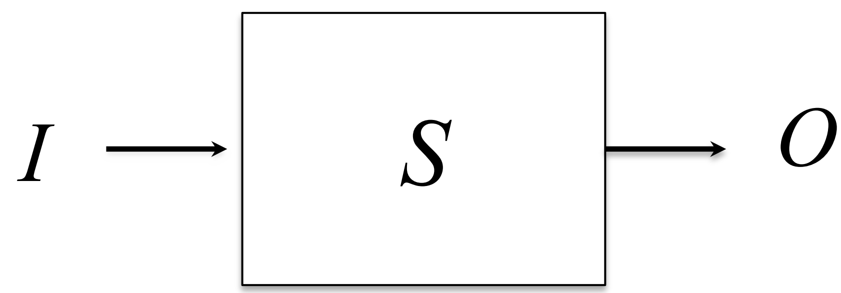


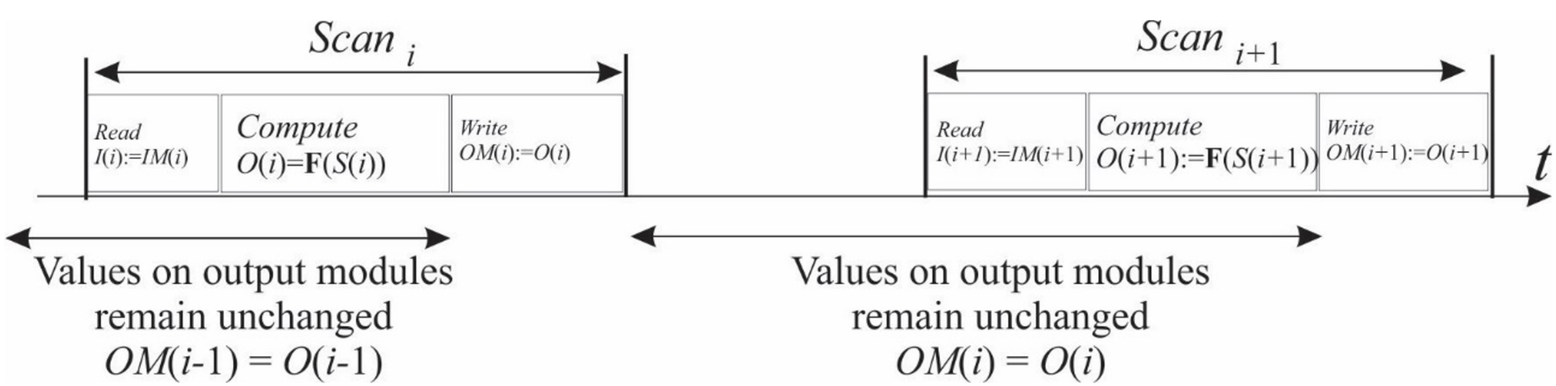
Reasons

- Reactivity:
- Timeliness:
- Reliability: if input is read with an error, the error will be corrected in the next scan.

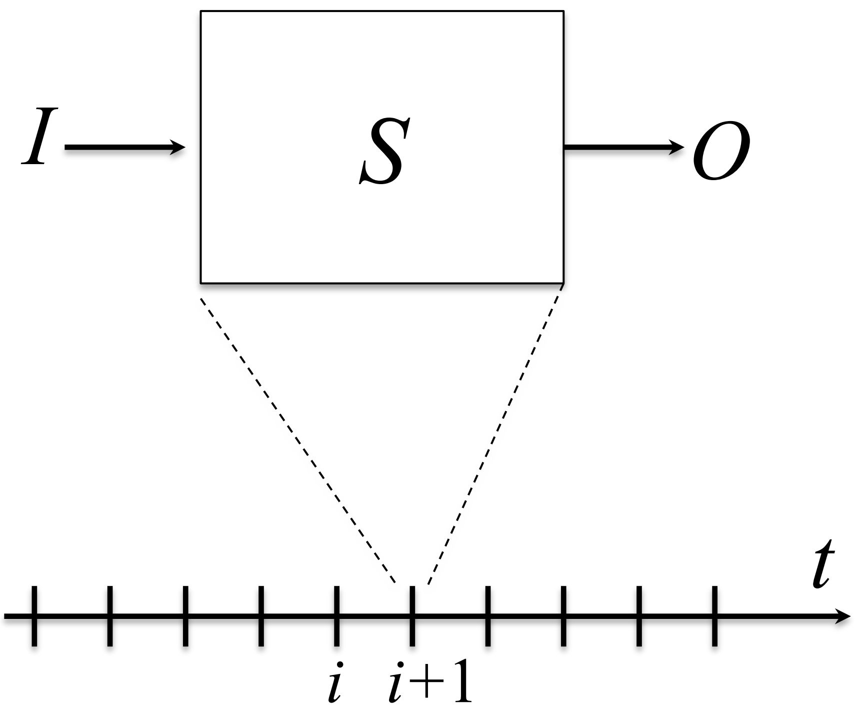
Formal Models of Automation Logic

Let us denote by I and O Boolean vectors of inputs and outputs, and S is Boolean vector of state variables.
Then the semantics of the controller can be described by the following system of Boolean assignments:





General Models of PLC execution



Combinatorial:

$$O(i)=\mathbf{F}(I(i))$$

Moore type state machine:

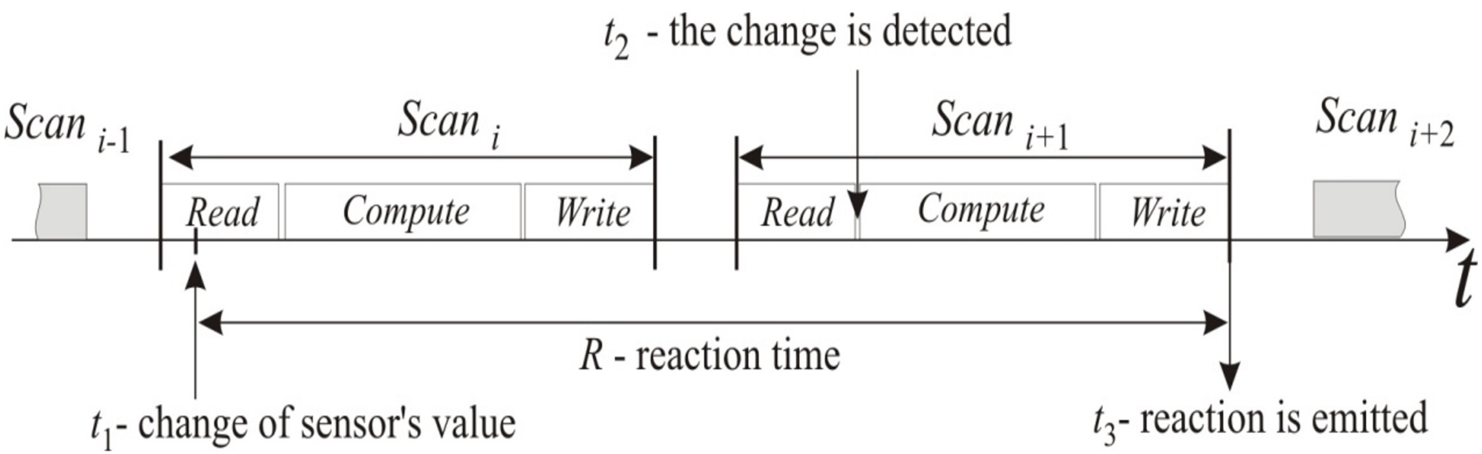
$$\begin{cases} S(i+1) := \mathbf{T}(I(i+1), S(i)); \\ O(i+1) := \mathbf{F}(S(i+1)), \end{cases}$$

Mealy type state machine:

$$\begin{cases} S(i+1) := \mathbf{T}(I(i+1), S(i)); \\ O(i+1) := \mathbf{F}(I(i+1), S(i)); \end{cases}$$

Aalto University

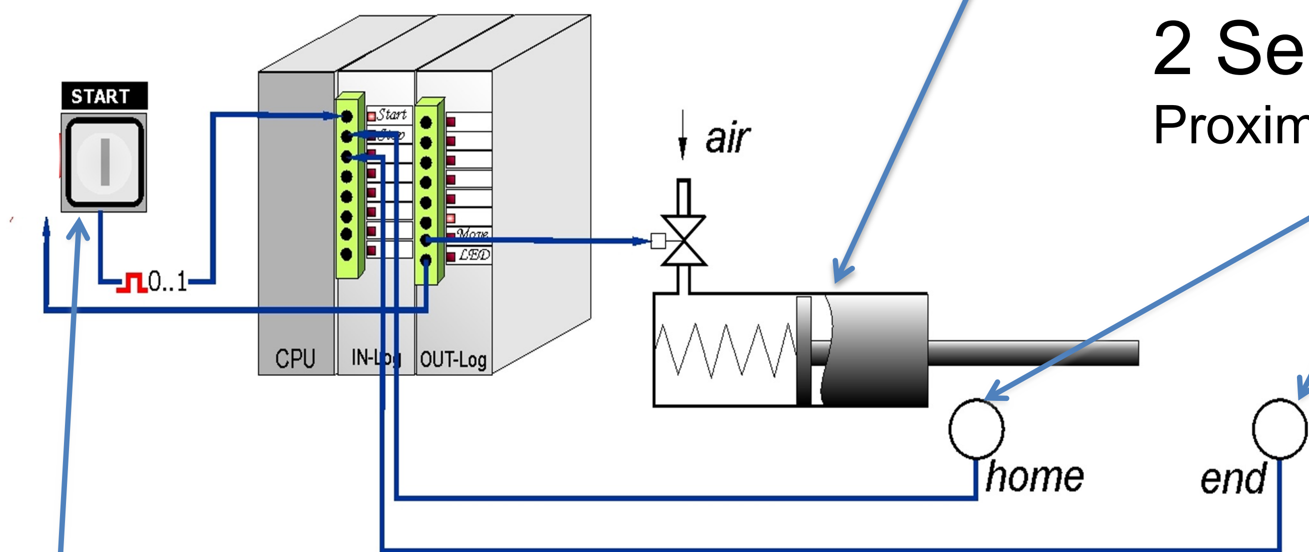
Reaction of PLC



Case-study: Single acting Pneumatic Cylinder

1 Actuator: Cylinder's valve

2 Sensors
Proximity sensor

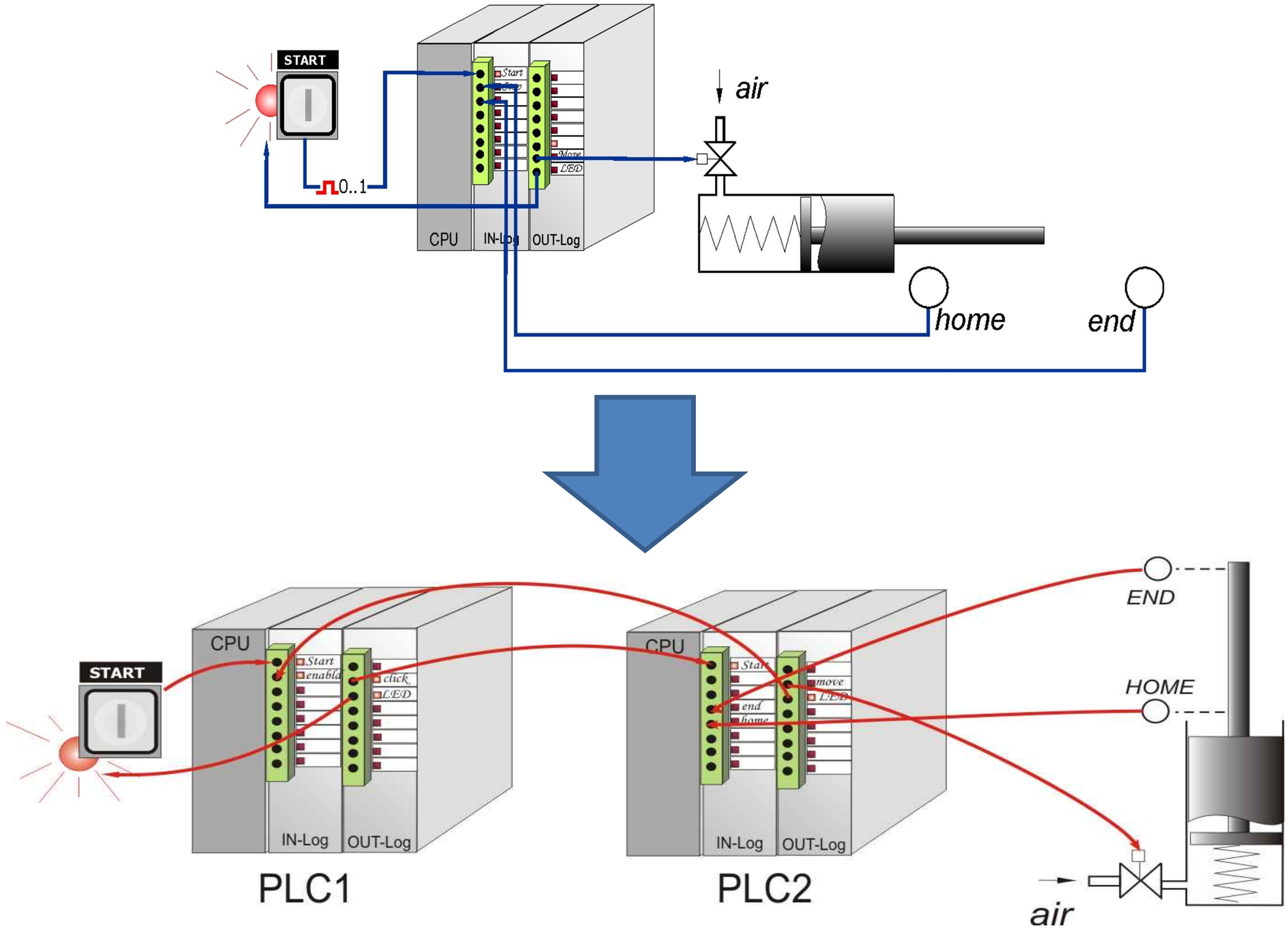


3rd Sensor

A Push Button, which generates the start event

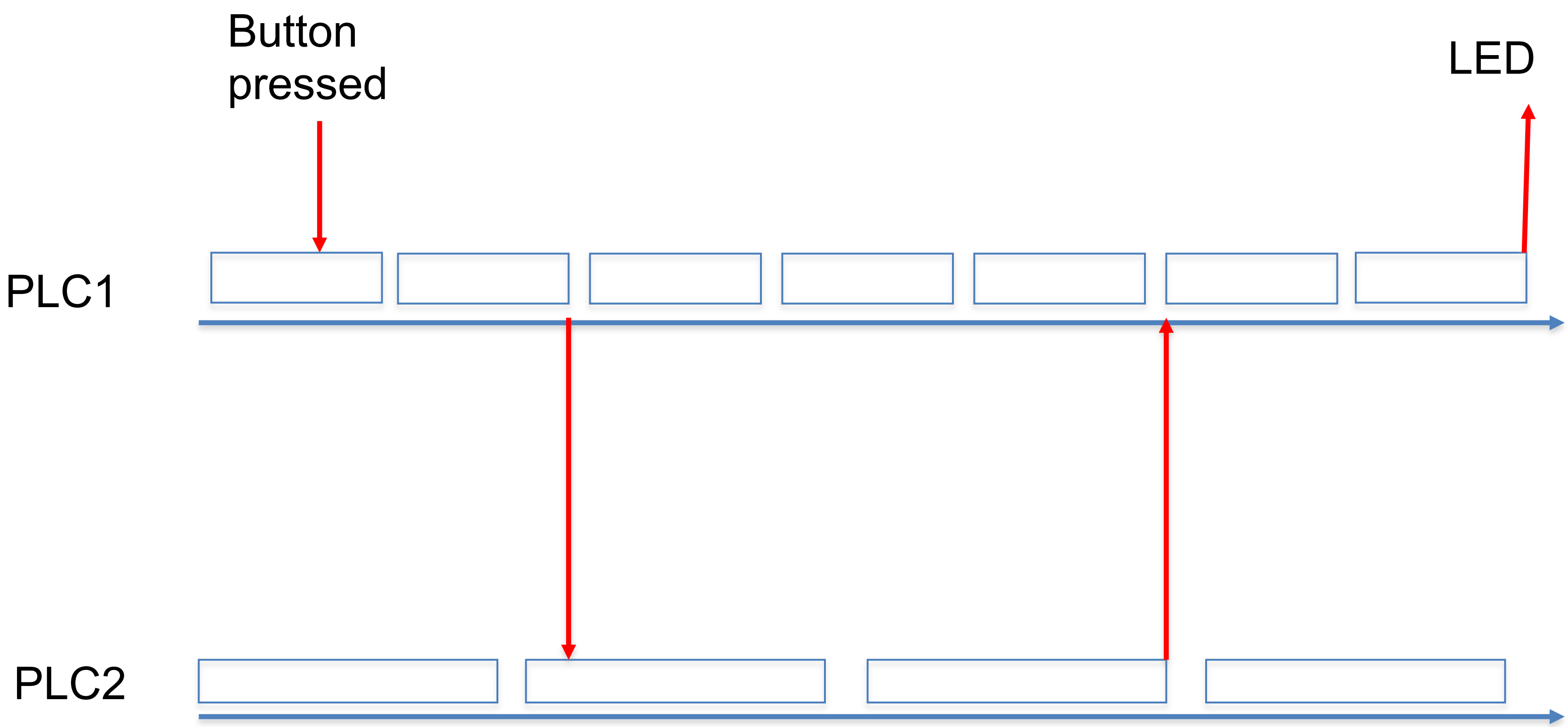
A **Proximity Sensor** is a **sensor** able to detect the presence of nearby objects without any physical contact.

Distributed PLC Systems



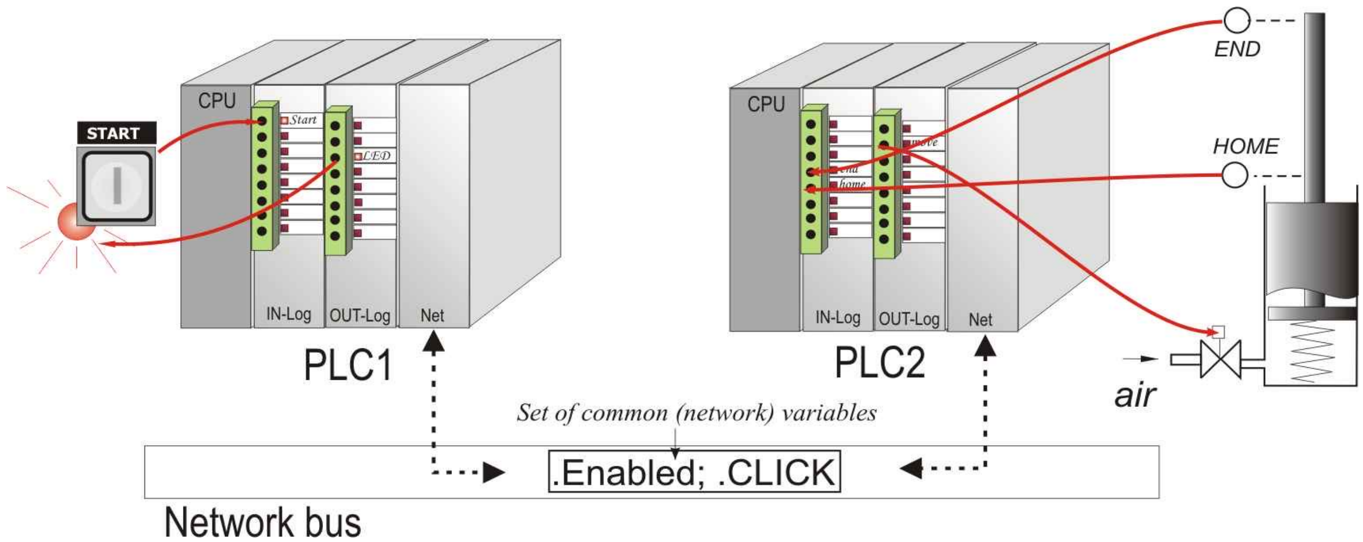
CLICK:= *START*;
LED:=*enable*;

Working and Reaction Time

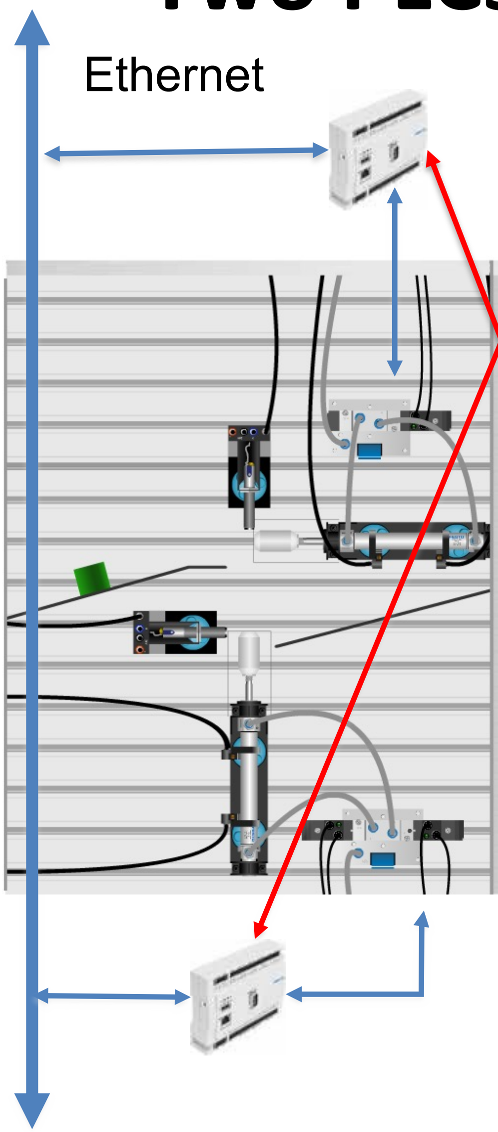


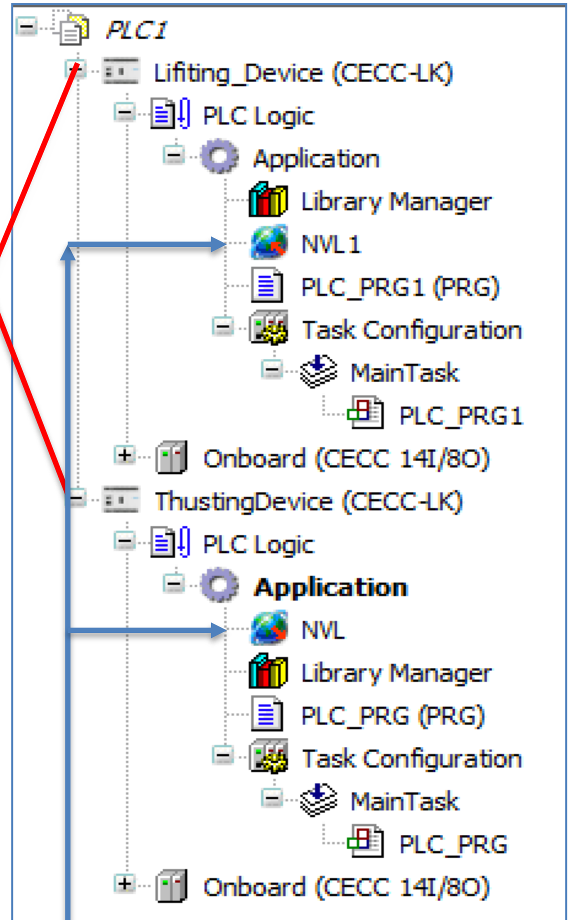
Networking PLCs

Declare .Enabled and .CLICK as shared variables



Two PLCs in the Lifting Luggage example





Network variable declaration

```
VAR_GLOBAL
  SharedVariable: BOOL;
END_VAR
```

```
PROGRAM PLC_PRG1
VAR
      LCExtended : BOOL;
      LCRetracted: BOOL;
      LuggageArrived: BOOL;
      ExtendLC: BOOL;
      RetractLC: BOOL;
END_VAR
IF LuggageArrived AND  NOT LCExtended THEN
      ExtendLC := TRUE;
      RetractLC := FALSE;
ELSIF LCExtended AND SharedVariable THEN
      ExtendLC := FALSE;
      RetractLC := TRUE;
END_IF
```

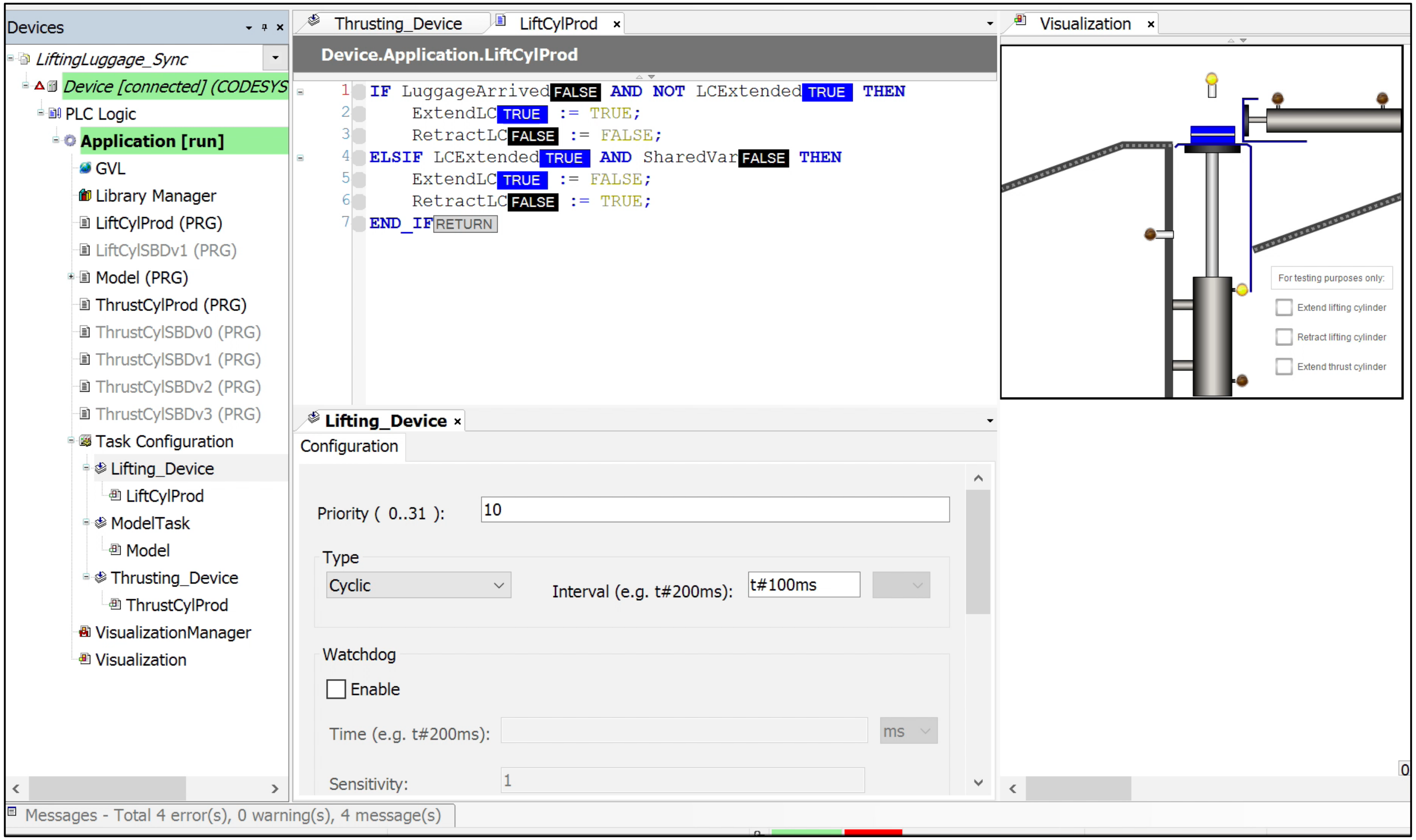
Sensors: LCExtended, LCRetracted, LuggageArrived → Actuator: ExtendLC

```
PROGRAM PLC_PRG
VAR
      TCRetracted: BOOL;
      TCExtended: BOOL;
      LuggageRaised: BOOL;
      ExtendTC: BOOL;
      LuggageAway: BOOL;
END_VAR
IF LuggageRaised AND TCRetracted THEN
      ExtendTC:= TRUE;
      LuggageAway := FALSE;
      SharedVariable := FALSE;
ELSIF TCExtended AND NOT LuggageRaised THEN
      ExtendTC:= FALSE;
      SharedVariable := TRUE;
END_IF
```

Sensors: TCRetracted, TCExtended, LuggageRaised → Actuator: ExtendTC

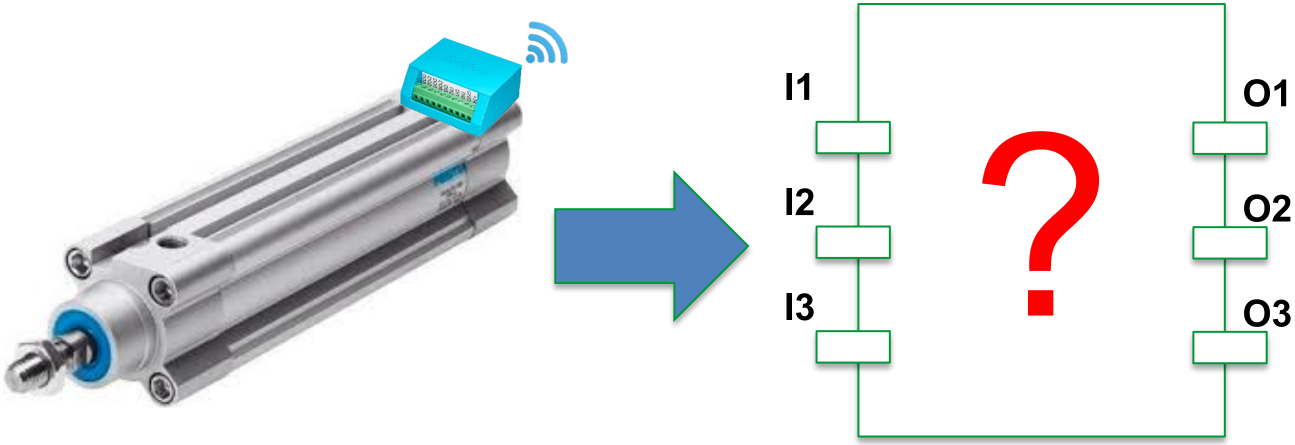
Aalto University

Using PLCs in a Distributed System: Working

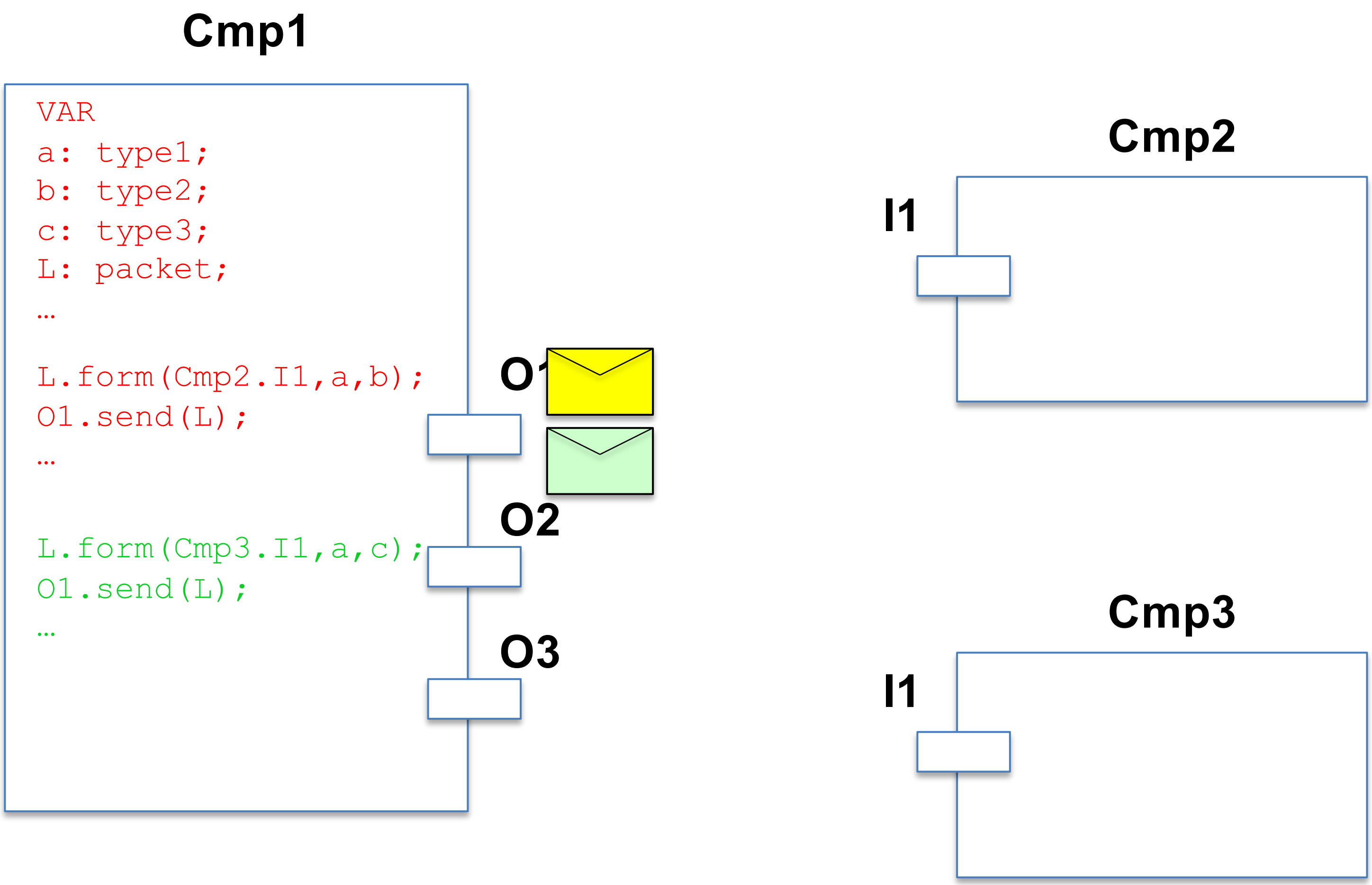


Intelligent Automation Component

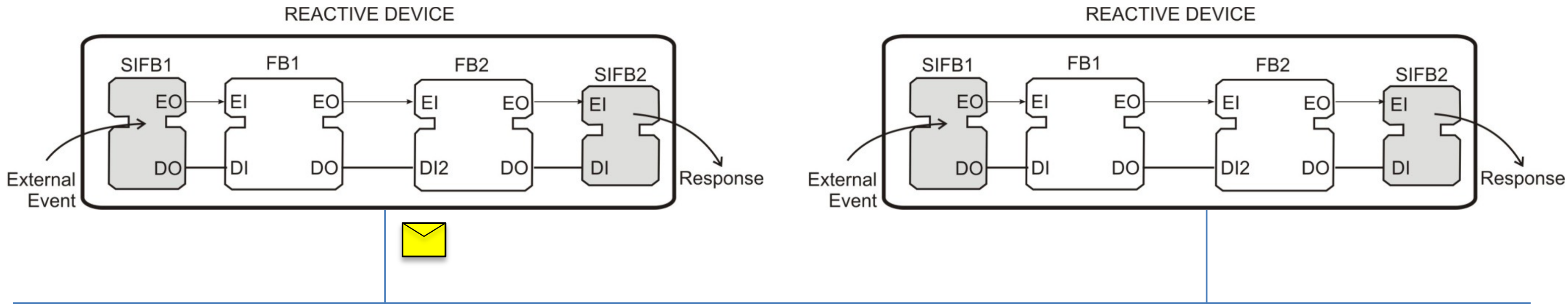
Cylinder Software Component



Communicating components



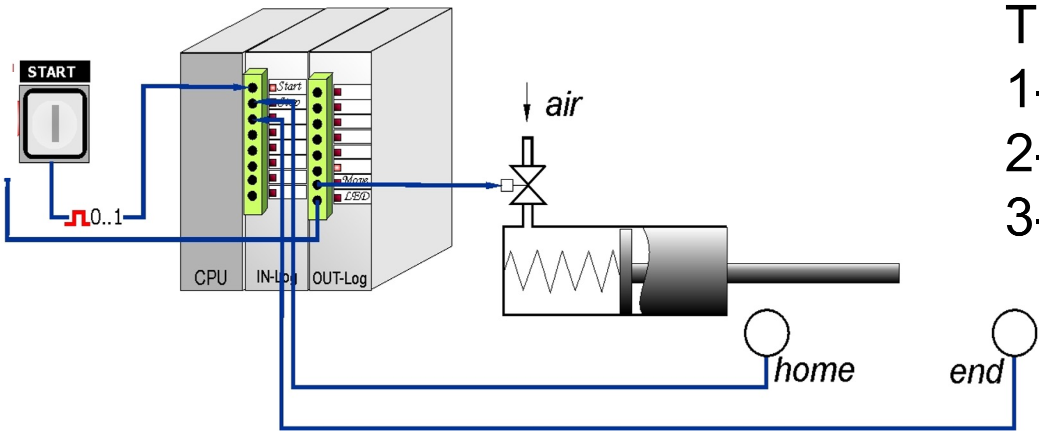
Event-driven interaction



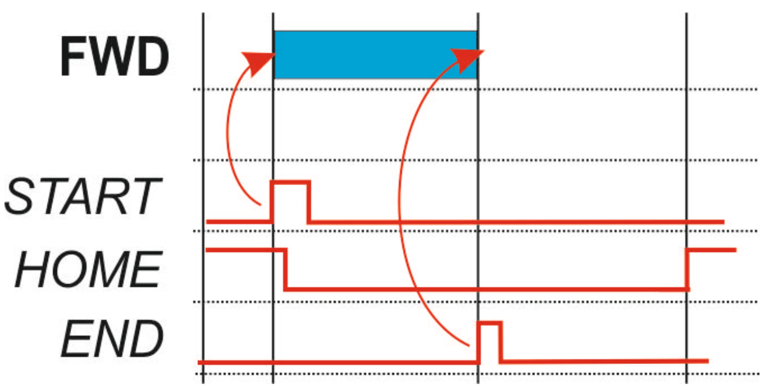
Application Logic Design

Example: Pneumatic Cylinder with Return Spring

Single acting cylinder has only one control signal **FWD** (move forward)



This system has 3 input signals
1- HOME
2- START
3- END

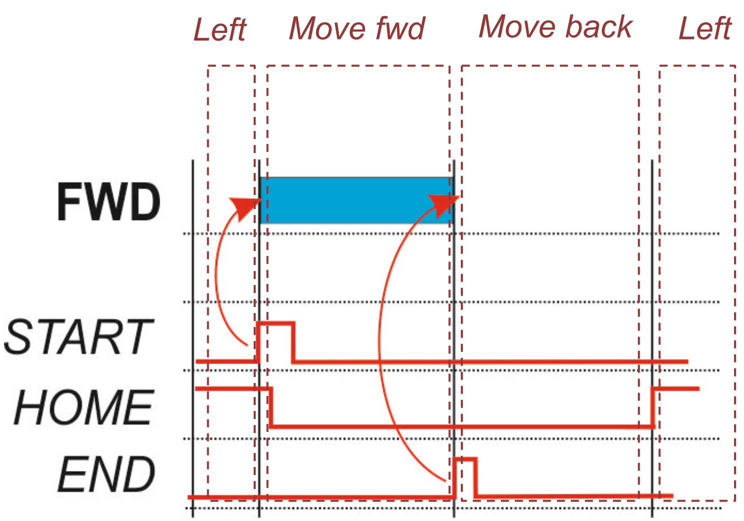


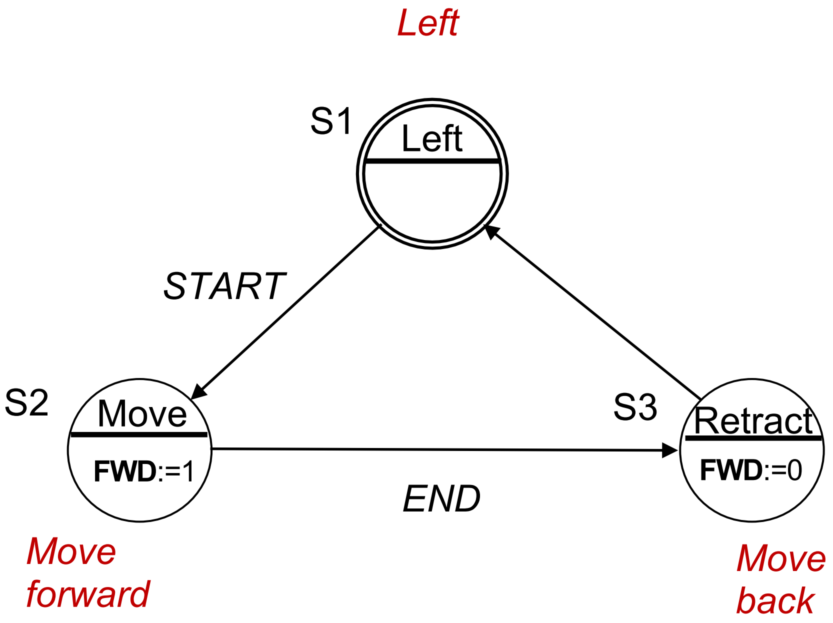
Actions and Signals:

- **HOME and END** signals are generated by 2 proximity sensors that are located in the 2 ending positions
- **START** signal is generated by the Start button.

Resulting state-machine

Scenario

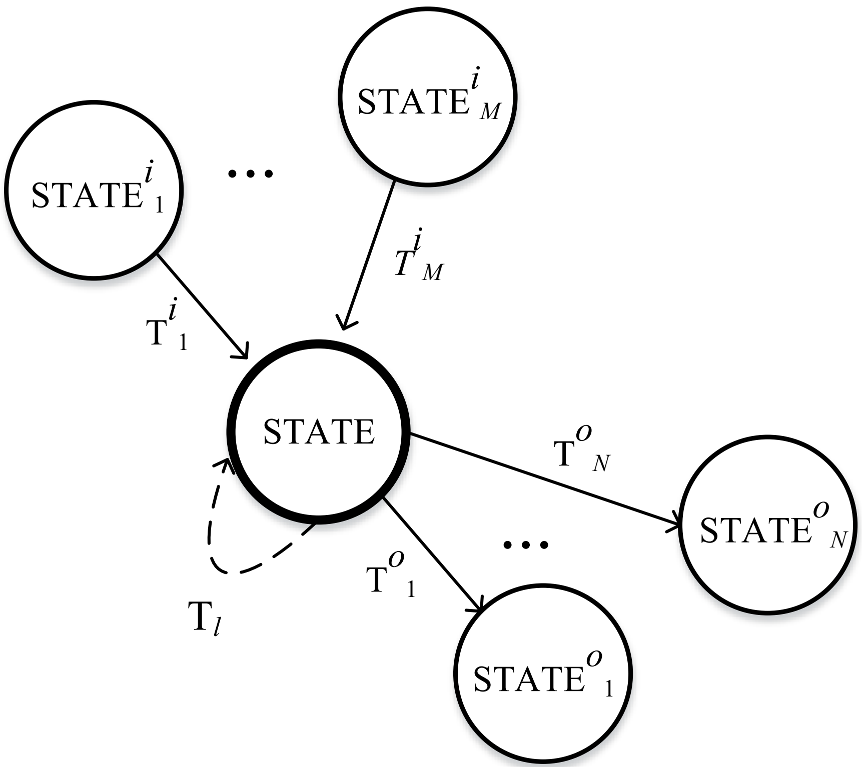




Algorithm

1. Identify stable states in the system's behavior.
   - A valid scenario from the system functionality is a good starting point
2. For each state define the output signals that shall be true in that state.
3. Define transitions from state to state.
4. Define transitions conditions

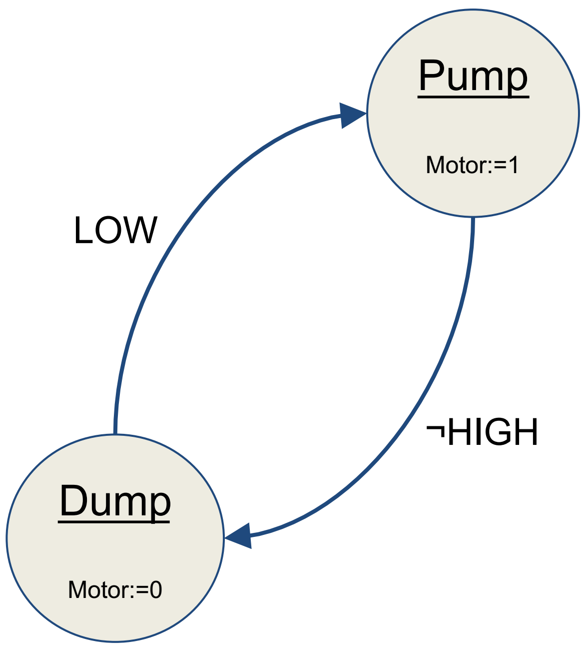
State machine design conventions



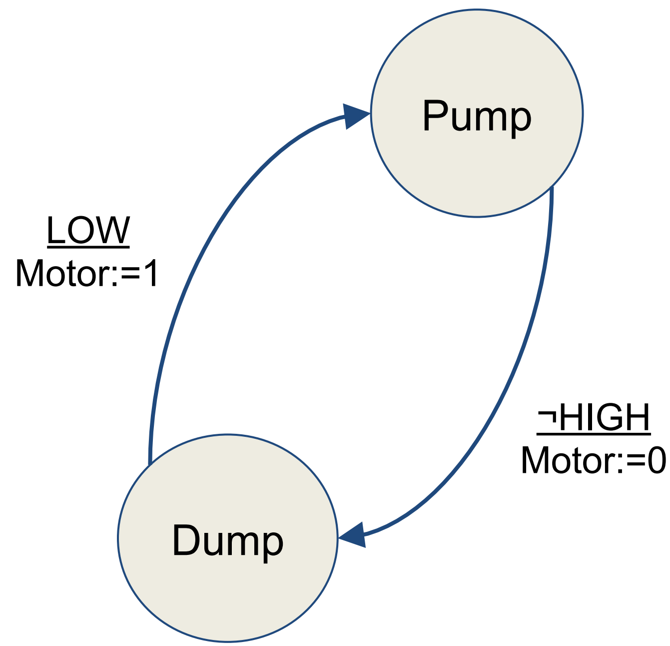
1. We are drawing state machines without loopback arcs.
2. Completeness of the outgoing conditions is guaranteed by the ELSE assumption
   1. at least one outgoing transition is true.
3. Orthogonality of outgoing transitions
   1. at most one outgoing transition is true
   2. provision of determinism.

Moore vs. Mealy state machines

Moore-type FSM

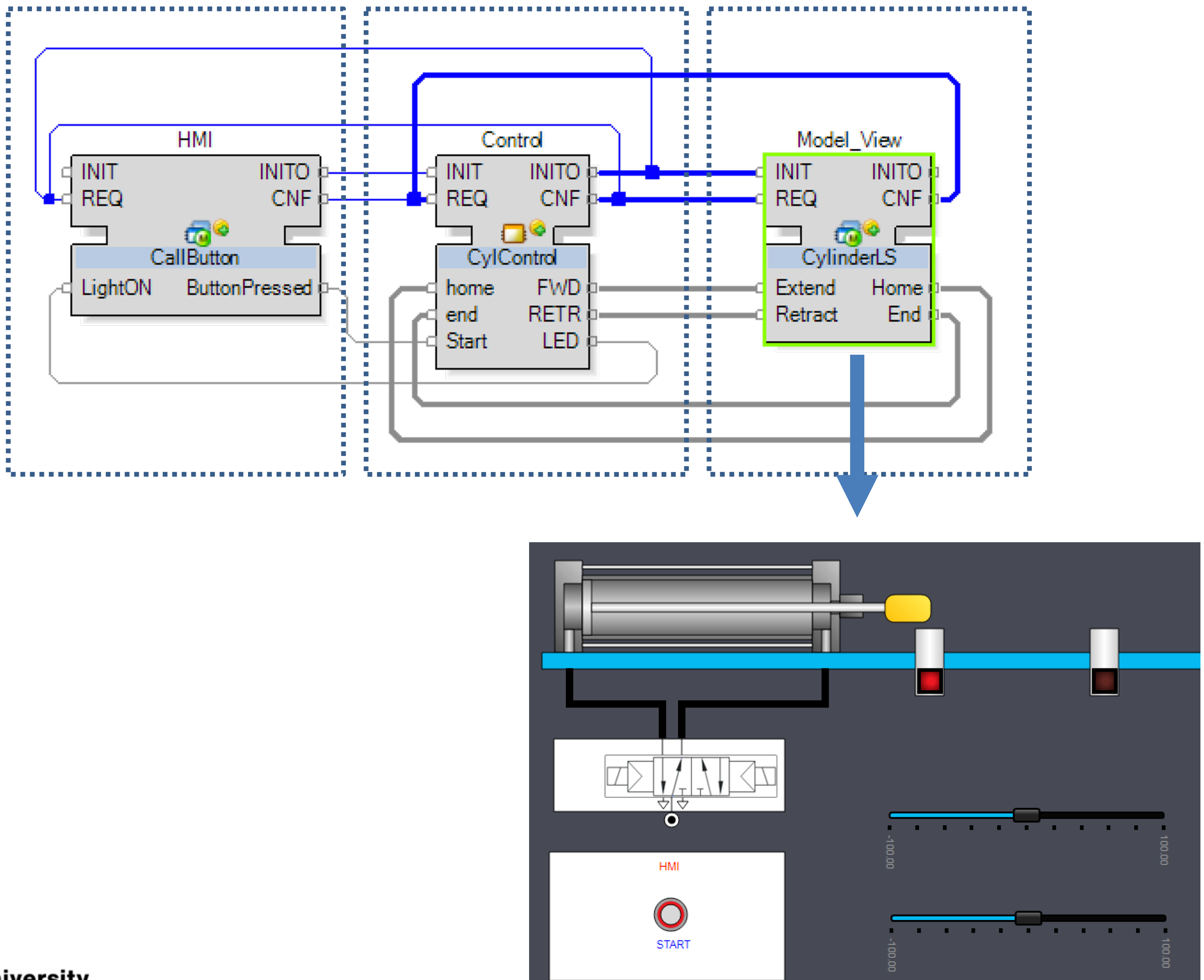


Mealy-type FSM



We will be using Moore machines throughout this lecture.

Simulation-based testing framework



What to remember?

- Generations of automation systems.
- What is the advantage of flat architecture compared to the "ICT pyramid"?
- Why intelligent machines are needed at the factory floor?
- What is main difference of combinatorial logic from state-based logic?
- What are pros and contras of using simulation in the loop?