# 1501217 Model Based Design

Course Description:
Model based design software; creating the model; simulating the model; analyzing simulation results; connecting to hardware; Software and Programming for Industrial Automation*.
(*modified in the framework of an Erasmus + project: Asean Factori 4.0 Across South East Asian Nations: From Automation and Control Training to the Overall Roll-out of Industry 4.0 609854-EPP-1-2019-1-FR-EPPKA2-CBHE-JP)

Learning outcome:
1. Students are able to implement engineering models.
2. Students are able to simulate engineering models.
3. Students are able to work on engineering model of Industry 4.0.

Lecturer:
Assoc. Prof. Punnarumol Temdee, Ph.D.
Asst. Prof. Roungsan Chaisricharoen, Ph.D.
Asst. Prof. Santichai Wicha, Ph.D.
Lect. Chayapol Kamyod, Ph.D.

Credit: 3(2-2)
Lecture: 30 Hours (6 hours of modified content)
Lab: 30 Hours (6 hours of modified content)

Assessments:

| | |
|---|---|
| Attendance | 10% |
| HW/CW | 20% |
| Midterm | 25% |
| Final | 25% |
| Project | 20% |

Lecture (seminar):

| Content | Hours |
|---|---|
| Concept of engineering models | 4 |
| Ideal vs practical behaviors | 4 |
| Implementation of engineering models | 4 |
| Mathematic based models | 4 |
| Probabilistic based models | 4 |
| Intelligent based models | 4 |
| Models of industrial HW* | 2 |
| Industrial automation models* | 2 |
| Industrial programing languages* | 2 |

(*modified in the framework of an Erasmus + project: Asean Factori 4.0 Across South East Asian Nations: From Automation and Control Training to the Overall Roll-out of Industry 4.0 609854-EPP-1-2019-1-FR-EPPKA2-CBHE-JP)

Lab (internship):

| Content | Hours |
| --- | --- |
| Physics of engineering models | 4 |
| Model simulations | 4 |
| System identification | 4 |
| Multi-system integration | 4 |
| Probabilistic models | 4 |
| Intelligent models | 4 |
| Time-based model of digital outputs* | 2 |
| Model of digital outputs via ladder diagram* | 2 |
| Model of analog outputs* | 2 |

(*modified in the framework of an Erasmus + project: Asean Factori 4.0 Across South East Asian Nations: From Automation and Control Training to the Overall Roll-out of Industry 4.0 609854-EPP-1-2019-1-FR-EPPKA2-CBHE-JP)

# 1501217 Model Based Design

Program: Bachelor program in Computer Engineering

Credit: 3(2-2) Lecture: 30 Hours Lab: 30 Hours

Co-funded by the Erasmus+ Programme of the European Union

1st Semester, Academic Year: 2023

Assoc. Prof. Punnarumol Temdee, Ph.D.

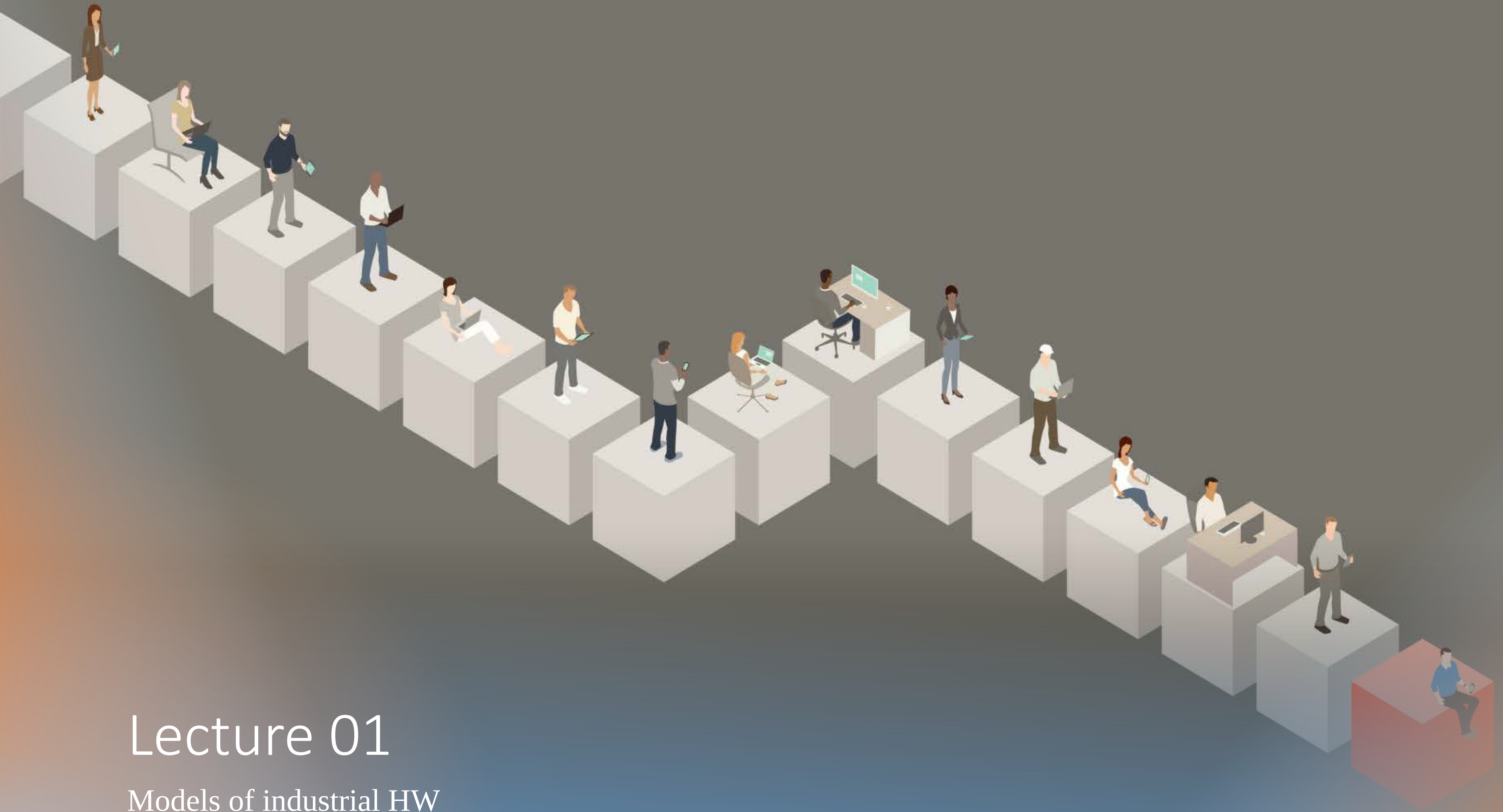
Asst. Prof. Roungsan Chaisricharoen, Ph.D.

Asst. Prof. Santichai Wicha, Ph.D.

Lect. Chayapol Kamyod, Ph.D.

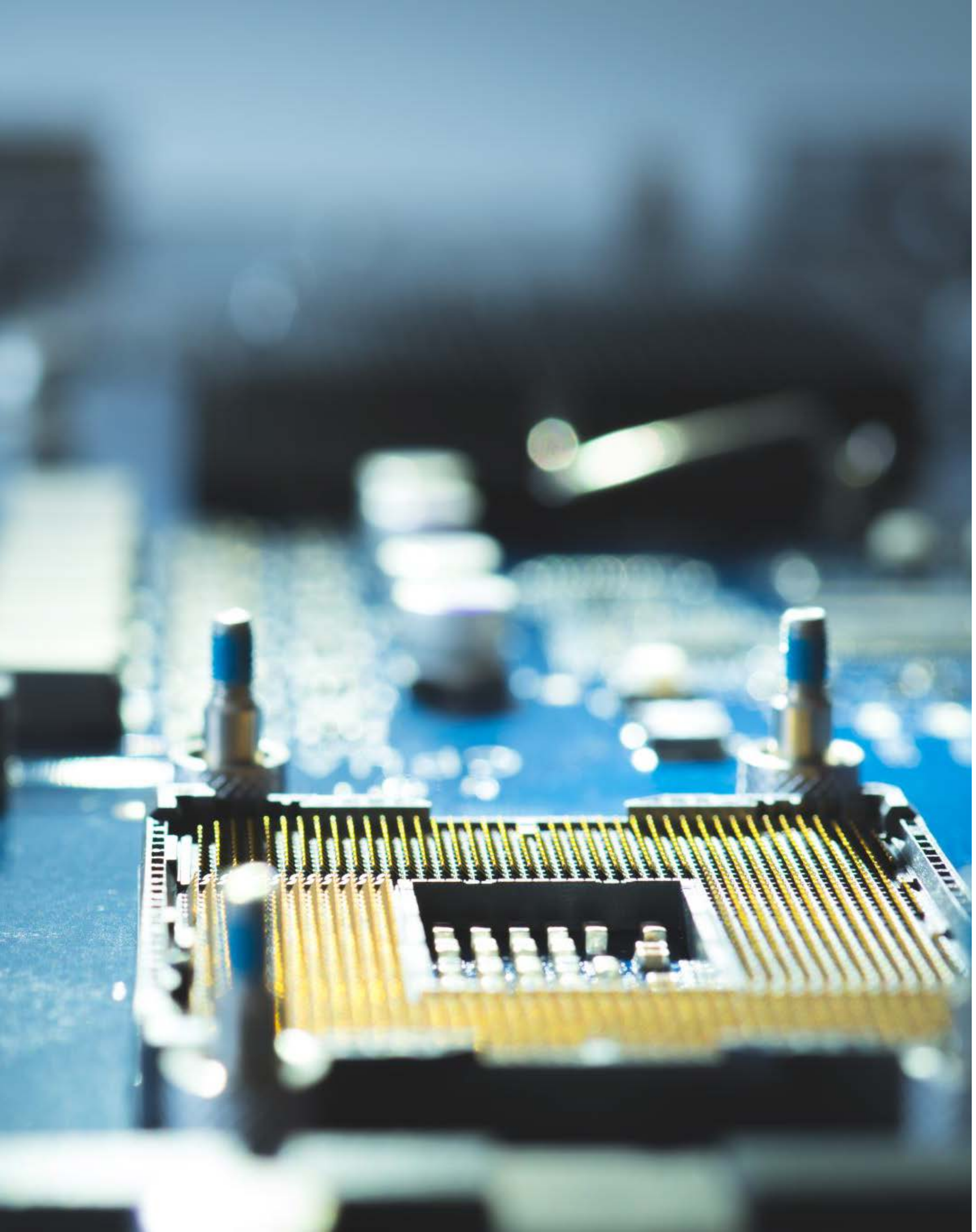
This course has been modified in the framework of an Erasmus + project: Asean Factori 4.0 Across South East Asian Nations: From Automation and Control Training to the Overall Roll-out of Industry 4.0

609854-EPP-1-2019-1-FR-EPPKA2-CBHE-JP



# Lecture 01

Models of industrial HW



# Overview of Model-Based Design

- Define MBD as an efficient approach to designing complex control systems and software that integrates simulation and automatic code generation
- Explain how MBD is used in industries such as automotive, aerospace, and electronics to develop embedded systems
- Highlight the key components: the model, design environment, simulation, automatic code generation, and verification/validation

# Benefits of Model-Based Design

- Illustrate how MBD can reduce the time from design to implementation through rapid prototyping and testing
- Discuss how MBD leads to higher quality products by enabling early detection of design flaws
- Highlight the ability of MBD to streamline design iterations and improve collaboration between software, mechanical, and electrical teams

# Key Concepts in Model-Based Design

1

Define modeling in the context of MBD as the process of creating executable specifications for system behaviors

2

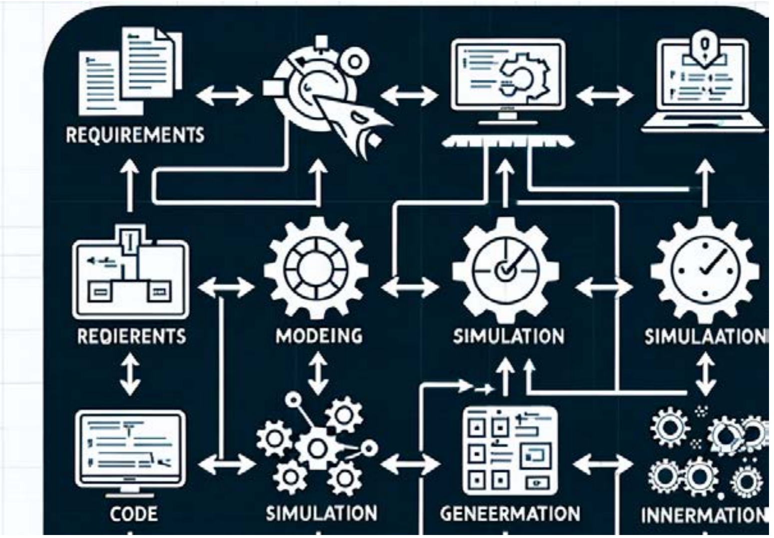
Explain simulation as the method for exploring and testing the performance of models without physical prototypes

3

Clarify verification (the process of checking that the model meets a set of requirements) and validation (ensuring the model accurately represents the real-world system)

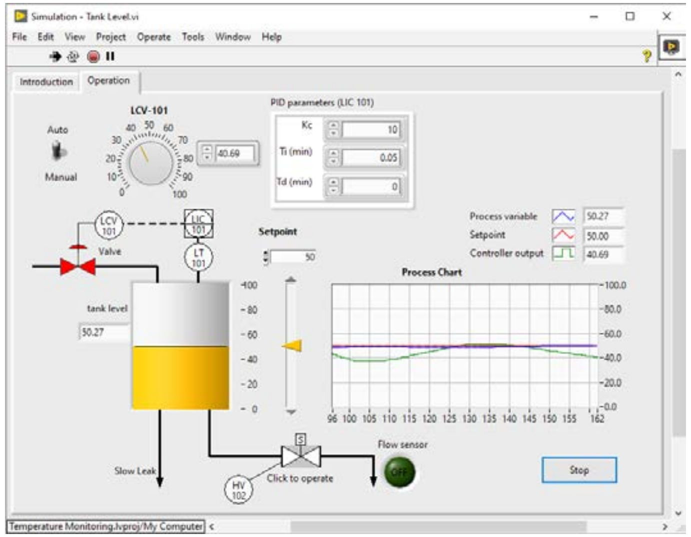
# Model-Based Design Workflow

Requirements → Modeling → Simulation → Code Generation → Hardware Implementation → Testing & Verification



# Tools for Model-Based Design

MATLAB/Simulink, LabVIEW, and others




MATLAB®

MATLAB® & SIMULINK®

# Case Study: Application of MBD

DETAIL A REAL-WORLD SCENARIO WHERE MBD WAS APPLIED SUCCESSFULLY, SUCH AS IN THE DEVELOPMENT OF AN AUTOMOTIVE CONTROL SYSTEM

EXPLAIN THE CHALLENGES FACED AND HOW MBD ADDRESSED THEM

SHARE THE RESULTS AND IMPROVEMENTS SEEN FROM USING MBD

# Challenges in Model-Based Design

ADDRESS COMMON ISSUES SUCH AS MODEL COMPLEXITY, COMPUTATIONAL DEMANDS, AND INTEGRATING MBD WITH EXISTING DEVELOPMENT PROCESSES

OFFER STRATEGIES FOR OVERCOMING THESE CHALLENGES, LIKE MODEL SIMPLIFICATION AND ENHANCED COMPUTATIONAL RESOURCES

DISCUSS THE IMPORTANCE OF TRAINING AND SKILL DEVELOPMENT TO EFFECTIVELY LEVERAGE MBD

# The Future of Model-Based Design

Discuss trends like the integration of artificial intelligence and machine learning into MBD

Explore the potential for MBD in emerging fields like autonomous vehicles and smart grid technology

Predict how these trends will shape the future of system design and development

# Understanding Engineering Models

### Fundamental Tools in Engineering:

Engineering models simplify and structure complex real-world systems for better understanding by engineers

Models are the cornerstone of engineers' design, analysis, and problem-solving

### Crucial Role in Design:

**Models aid engineers in visualizing and planning systems before physical construction.**

**Models enable engineers to experiment for efficient solutions.**

### Crucial Role in Analysis:

Models aid in analysing and predicting system behaviour.

Analysis reveals system weaknesses for informed decisions

### Crucial Role in Optimization:

Models optimize by adjusting parameters for performance

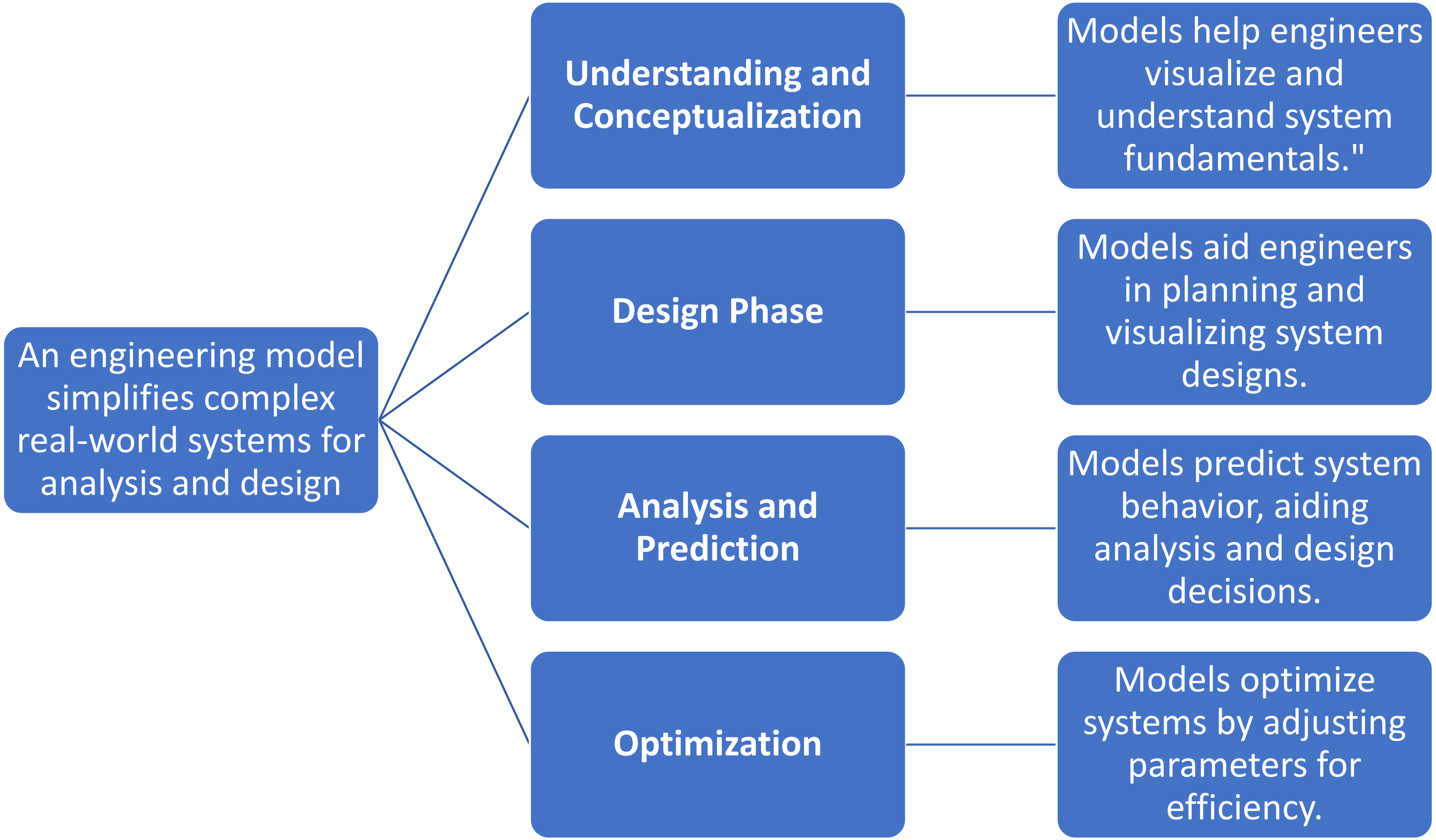
Optimization is iterative, refining designs for the best outcome.

### Versatility Across Systems and Processes:

Versatile models apply to diverse engineering domains

Used for analysing and designing diverse systems.

# Concept of Engineering Model



# Concept of Engineering Model

## 01

**Verification and Validation:**

- Models verify and validate designs to ensure real-world functionality.

## 02

**Communication:**

- Models aid communication, making complex ideas understandable

## 03

**Risk Assessment:**

- Models assess project risks, enabling issue identification and mitigation

## 04

**Iterative Improvement:**

- Models aid iterative design, driving continuous improvement.

# Ideal vs. Practical Behavior

## Idealized

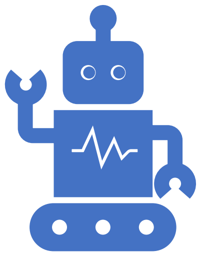
Simplified Assumptions

Theoretical Precision

Conceptual Clarity

Limited Realism

Common in Theoretical Sciences



## Practical Behavior

Incorporate Real Factors:

Complexity

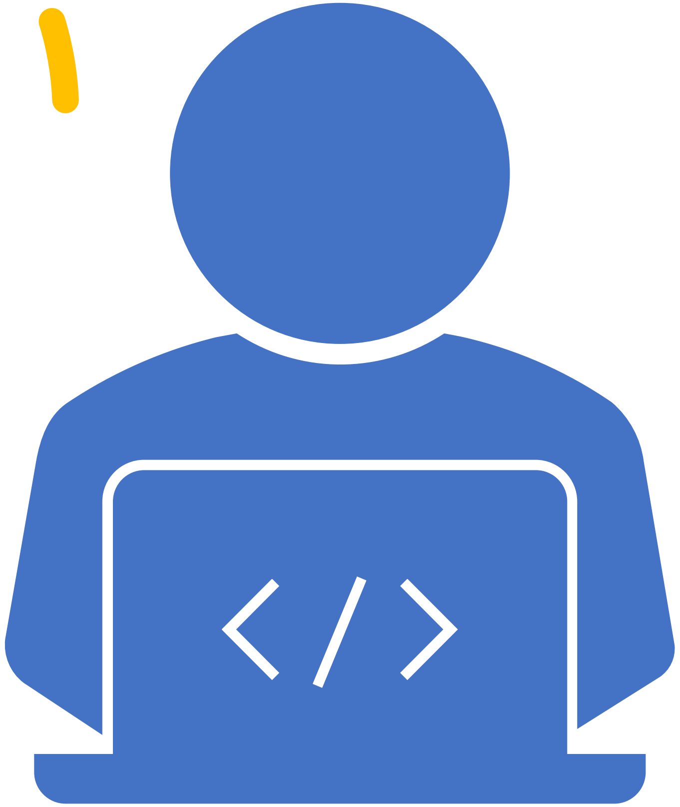
Practical Application

Risk Assessment

Optimization and Decision-Making:

# Implementation of Engineering Model

- Engineering models are used to solve real-world problems and make decision
  - Problem Identification
    - ➢ Engineering models are used to solve real-world problems and make decisions
  - Data Collection.
    - ➢ Start by identifying the problem, such as product design or process optimization.
  - Model Selection
    - ➢ Gather crucial data as the foundation for an accurate model.
  - Model Development
    - ➢ Select the right model type based on the problem: math, prototypes, or simulations
  - Parameter Estimation
    - ➢ Create the model using equations, algorithms, or physical representations, considering assumptions.
  - Model Validation
    - ➢ Estimate model parameters from data for real-world alignment



# Implementation of Engineering Model

| | |
|---|---|
| Model Verification | Estimate model parameters from data for real-world alignment. |
| Simulation or Analysis | Simulate system behavior with the model for insights and improvements. |
| Optimization | Optimize the system using the model to maximize desired outcomes |
| Decision-Making | Make informed decisions based on the model's insights |
| Documentation | Document the model for transparency and future reference |
| Integration | Integrate the model into the project for ongoing decision-making. |

# Lecture 02 Industrial automation models

# Mathematics-Based Model

**Mathematical Abstraction**

Math equations represent and simplify real-world systems for analysis.

**Fundamental Laws**

Fundamental laws form the mathematical basis for engineering models.

**Differential Equations**

Differential equations describe dynamic behaviors in engineering systems."

**Linear and Nonlinear Models**

Math models: Linear for simplicity, nonlinear for complexity."

**Optimization**

Math optimization finds best solutions for efficiency and cost.

# Mathematics-Based Model (Cont.)

### Simulation

- Numerical methods and simulations analyse complex models, revealing insights

### Control Systems

- Control theory uses math, equations, and transfer functions for dynamic system regulation.

### Finite Element Analysis (FEA)

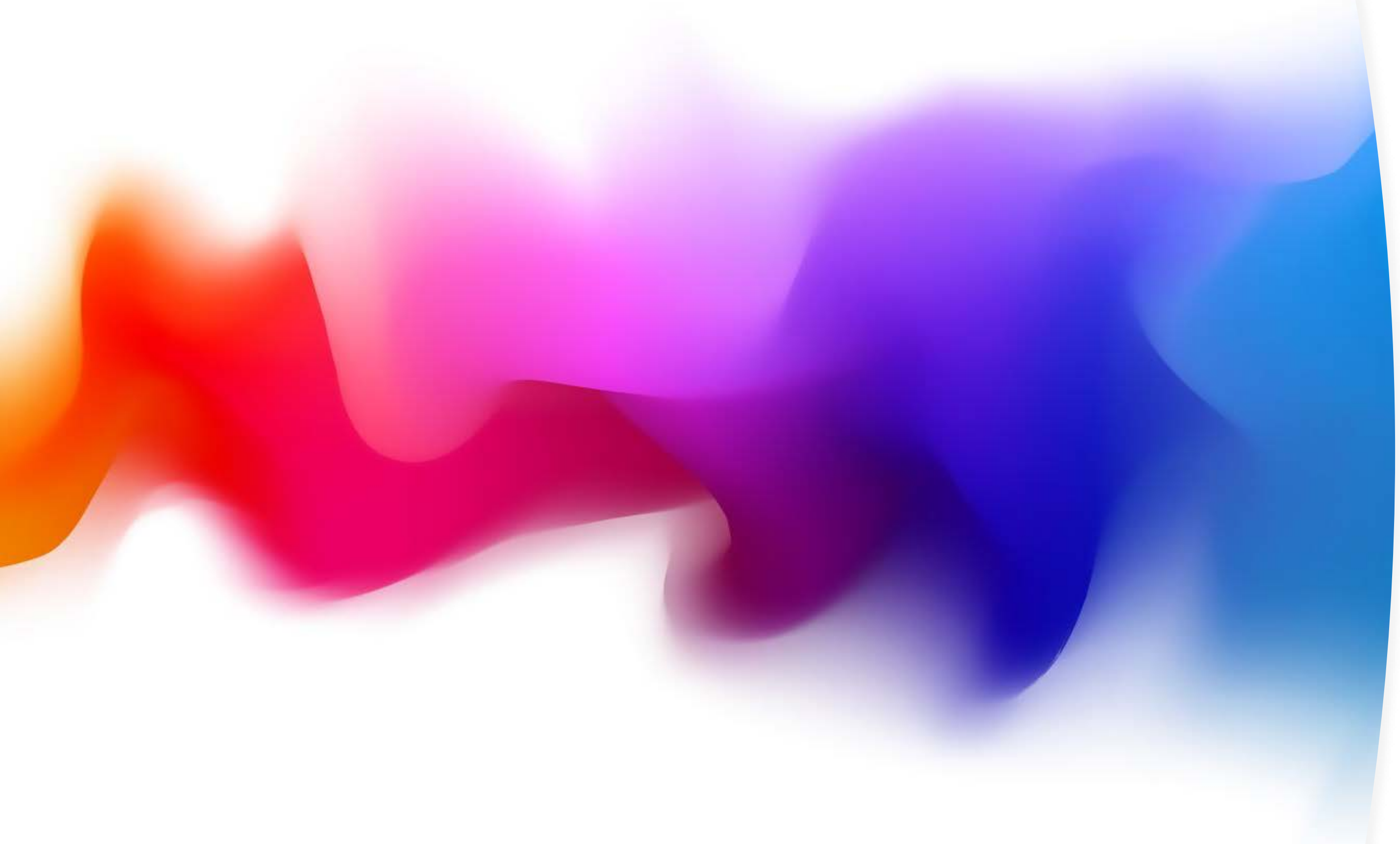
- FEA divides complex systems, aiding structural analysis and design.

### Statistical Models

- Probability and stats analyse data, predict reliability, and handle uncertainty in engineering models

### Modelling Complex Systems

- Math modeling covers weather, environment, and finance, blending equations and data.



# Probability-Based Model

- Probability and statistics in engineering models account for uncertainty, variability, and risk.
  - **Uncertainty Characterization**
  - **Data Analysis**
  - **Probabilistic Modeling**
  - **Monte Carlo Simulations**

# Probability-Based Model

- Probability and statistics in engineering models account for uncertainty, variability, and risk.
  - **Uncertainty Characterization**
  - **Data Analysis**
  - **Probabilistic Modeling**
  - **Monte Carlo Simulations**
  - **Reliability Analysis**
  - **Sensitivity Analysis**
  - **Risk Assessment**
  - **Design for Reliability**
  - **Statistical Quality Control**

# Probability and Statistics in Engineering

- Importance of probability and statistics in engineering decisions
- Key statistical concepts for engineering analysis
- Understanding variability, uncertainty, and risk in engineering contexts

# Importance of Probability and Statistics in Engineering Decisions

| | |
|---|---|
| **Foundational Role in Decision Making** | •Enable engineers to make informed decisions based on data rather than intuition.<br>•Provide a means for quantifying the likelihood of various outcomes and assessing risks. |
| **Driving Design Under Uncertainty** | •Allow for the design of systems that are robust under varying conditions.<br>•Help in predicting performance and reliability of engineering systems. |
| **Optimizing Resources** | •Aid in resource allocation by predicting and mitigating potential issues before they arise.<br>•Ensure cost-effective use of materials and processes. |
| **Quality Control and Improvement** | •Statistical methods are key in monitoring production processes to maintain and improve quality.<br>•Six Sigma and other quality improvement methodologies rely heavily on statistical analysis. |
| **Risk Assessment and Management** | •Facilitate the identification, analysis, and mitigation of risks in engineering projects.<br>•Enable the development of safety protocols and failure analysis. |
| **Innovation and Development** | •Support the development of new technologies through the analysis of experimental data.<br>•Essential in fields like biomedical engineering, where statistical analysis underpins innovation. |
| **Regulatory Compliance** | •Necessary for meeting industry standards and regulatory requirements, which often demand statistical proof of compliance. |
| **Predictive Maintenance** | •Use historical data to predict when maintenance should be performed, leading to better planning and reduced downtime. |

# Key Statistical Concepts for Engineering Analysis

| | |
|---|---|
| **Descriptive Statistics** | Tools to describe and summarize data: mean, median, mode, range, variance, and standard deviation.<br>Importance of understanding data distributions and shape characteristics: skewness and kurtosis. |
| **Inferential Statistics** | Using sample data to make inferences about a larger population.<br>Concepts of hypothesis testing, confidence intervals, and p-values. |
| **Probability Distributions** | Different types of distributions (normal, binomial, Poisson, etc.) used to model various data behaviors.<br>Application of probability distributions in failure rate modeling and life data analysis. |
| **Regression Analysis** | Techniques for modeling the relationship between dependent and independent variables.<br>Use in predictive modeling and trend analysis. |
| **Design of Experiments (DoE)** | Systematic methods to determine the relationship between factors affecting a process and the output of that process.<br>Use in process optimization and determining cause-and-effect relationships. |
| **Statistical Process Control (SPC)** | Methods for monitoring, controlling, and improving processes through statistical analysis.<br>Use of control charts and process capability analysis. |
| **Reliability Engineering** | Statistical methods in assessing system durability and maintenance requirements.<br>Use of Weibull analysis, survival analysis, and fault tree analysis. |
| **Risk Analysis** | Quantitative methods for assessing risk and its impact on decision-making.<br>Use in safety engineering and financial risk assessment. |

# Understanding Variability, Uncertainty, and Risk in Engineering Contexts

### Variability

- Definition: Variability is the inherent spread in a dataset due to differences in manufacturing, environmental conditions, and user operations.
- Impact on Engineering: Affects quality control, tolerance design, and performance consistency.
- Managing Variability: Use statistical measures like standard deviation and variance, and employ techniques such as SPC (Statistical Process Control).

### Uncertainty

- Definition: Uncertainty refers to the lack of complete certainty about the model or data, often due to incomplete knowledge.
- Types of Uncertainty: Can arise from measurement errors, incomplete sampling, and model approximations.
- Addressing Uncertainty: Incorporate safety factors, conduct sensitivity analyses, and use Bayesian methods for improving model predictions.

### Risk

- Definition: Risk is the potential of losing something of value and is often quantified as the probability of an undesirable event times its consequences.
- Risk in Engineering: Critical in decision-making, especially in safety-critical systems, financial planning, and disaster management.
- Risk Management: Identify, analyze, and prioritize risks followed by coordinated application of resources to minimize, monitor, and control the probability or impact of unfortunate events using tools like FMEA (Failure Modes and Effects Analysis) and risk matrices.

### Reliability and Safety Engineering

- Reliability Engineering: Ensures a system performs without failure under stated conditions for a specified period.
- Safety Engineering: Focuses on designing systems to be safe and to minimize the risk of accidents and malfunctions.

### Quantitative Techniques

- Probabilistic Analysis: Uses probability to assess variability and uncertainty.
- Monte Carlo Simulations: Perform risk assessment and decision analysis under uncertainty.

# Uncertainty Characterization

- Different types of uncertainties: aleatory and epistemic
- Tools and techniques for measuring uncertainty
  - Sensitivity analysis
  - Uncertainty propagation
  - Expert elicitation
- Case studies demonstrating the impact of uncertainty characterization in engineering

# Defining Aleatory and Epistemic Uncertainty

### Aleatory Uncertainty

Definition: Also known as 'inherent' or 'stochastic' uncertainty, it arises from the natural variability in systems or processes.

Characteristics: Irreducible and unpredictable, often modeled using probability distributions.

Examples: Variation in material properties, environmental conditions, or load demands.

### Epistemic Uncertainty

Definition: Results from a lack of knowledge or information about the system or environment. It is also referred to as 'systematic' uncertainty.

Characteristics: Reducible with additional information, data, or research.

Examples: Uncertainty in modeling assumptions, incomplete data, or uncertain parameters.

### Managing Uncertainties in Engineering

For Aleatory: Employ robust design principles to accommodate natural variability and ensure system reliability.

For Epistemic: Improve data collection, conduct more experiments, or refine models to reduce uncertainty.

### Implications for Design and Decision Making

Necessity to understand both types of uncertainty for effective risk management.

Influence on safety factors, design margins, and maintenance schedules.

# Data Analysis

- The role of data analysis in engineering problem-solving
- Descriptive, inferential, and computational data analysis methods
- Visualizing data to understand trends and patterns
- Utilizing statistical software for data analysis in engineering
- Examples of data analysis applications in civil, mechanical, and electrical engineering

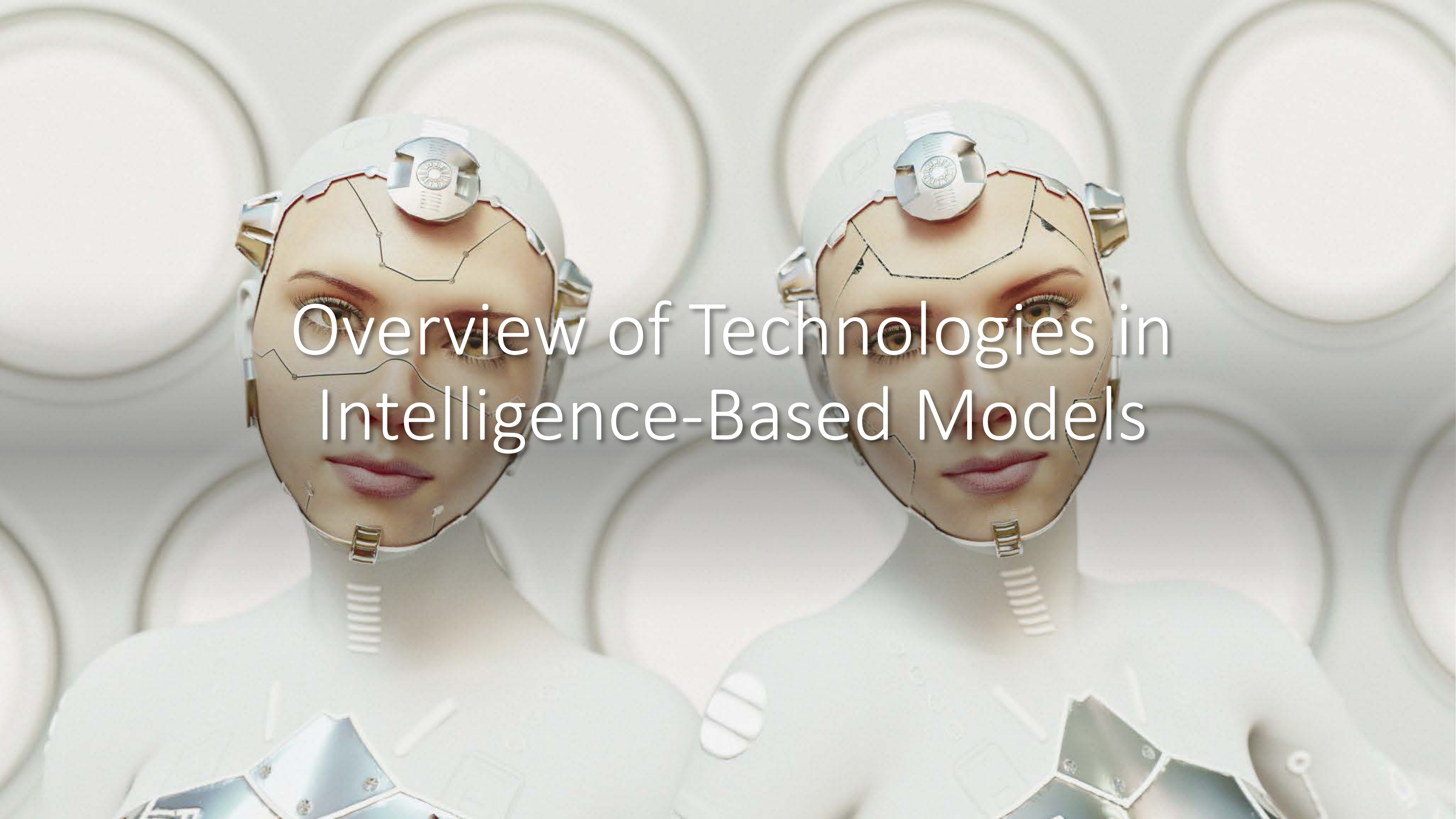
# Probabilistic Modeling

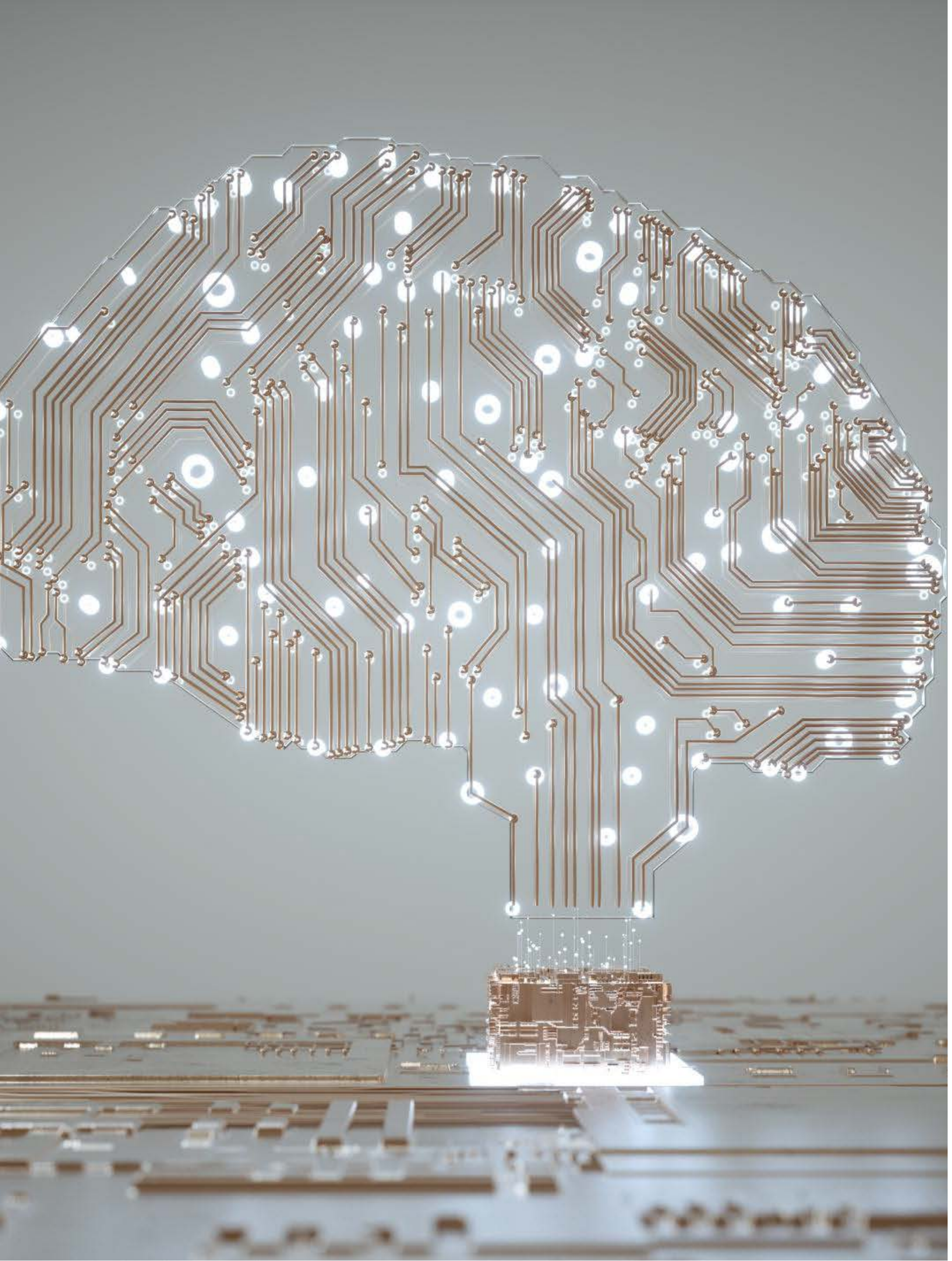
- Probabilistic models and their necessity in engineering
- Steps to build a probabilistic model:
  - Defining the problem and objectives
  - Selecting the appropriate probability distribution
  - Estimating parameters and fitting models to data
- Validation of models against empirical data
- Probabilistic models in action: Reliability engineering and risk assessment

# Monte Carlo Simulations

- Monte Carlo methods as a probabilistic simulation tool
- Process of setting up a Monte Carlo simulation:
  - Defining the model and inputs
  - Running simulations with random sampling
  - Analyzing the results to infer probabilities and risks
- Applications of Monte Carlo simulations in complex engineering systems
- Advantages of Monte Carlo simulations in cost estimation and project planning

# Overview of Technologies in Intelligence-Based Models





# Artificial Intelligence (AI):

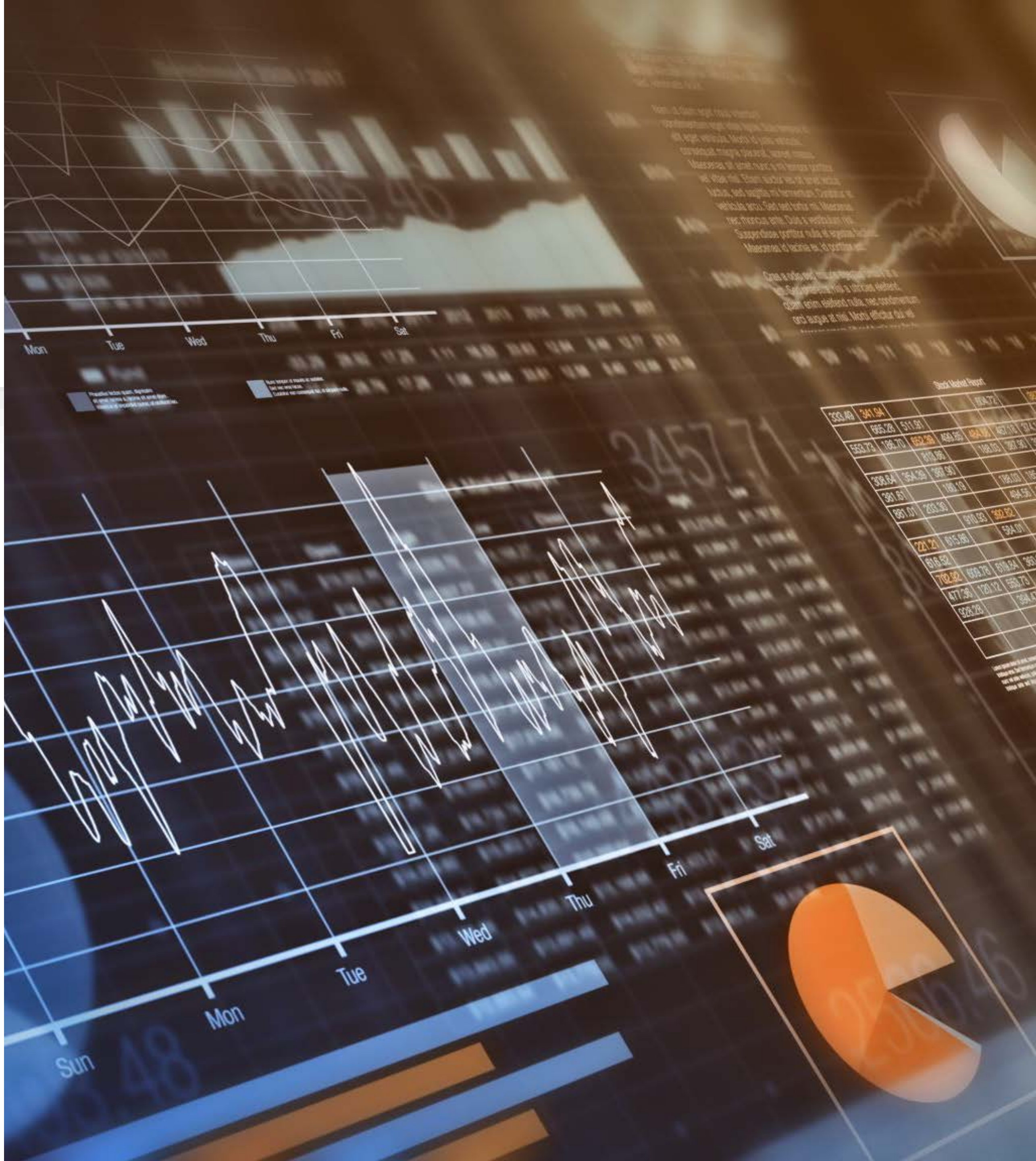
- **Description:** Briefly describe AI as a technology that simulates human intelligence processes by machines, especially computer systems.
- **Role in the Model:** Explain how AI is used for decision-making processes, learning from data, and automating complex tasks within the model.

# Machine Learning (ML):

- **Description:** Define ML as a subset of AI that allows systems to automatically learn and improve from experience without being explicitly programmed.
- **Role in the Model:** Discuss how ML algorithms analyze data, identify patterns, and make predictions, enhancing the model's ability to adapt and evolve.

# Data Analytics:

- **Description:** Outline data analytics as the process of analyzing raw data to find trends and answer questions.
- **Role in the Model:** Illustrate how data analytics provides the foundation for insights, supporting AI and ML in making informed decisions and predictions.

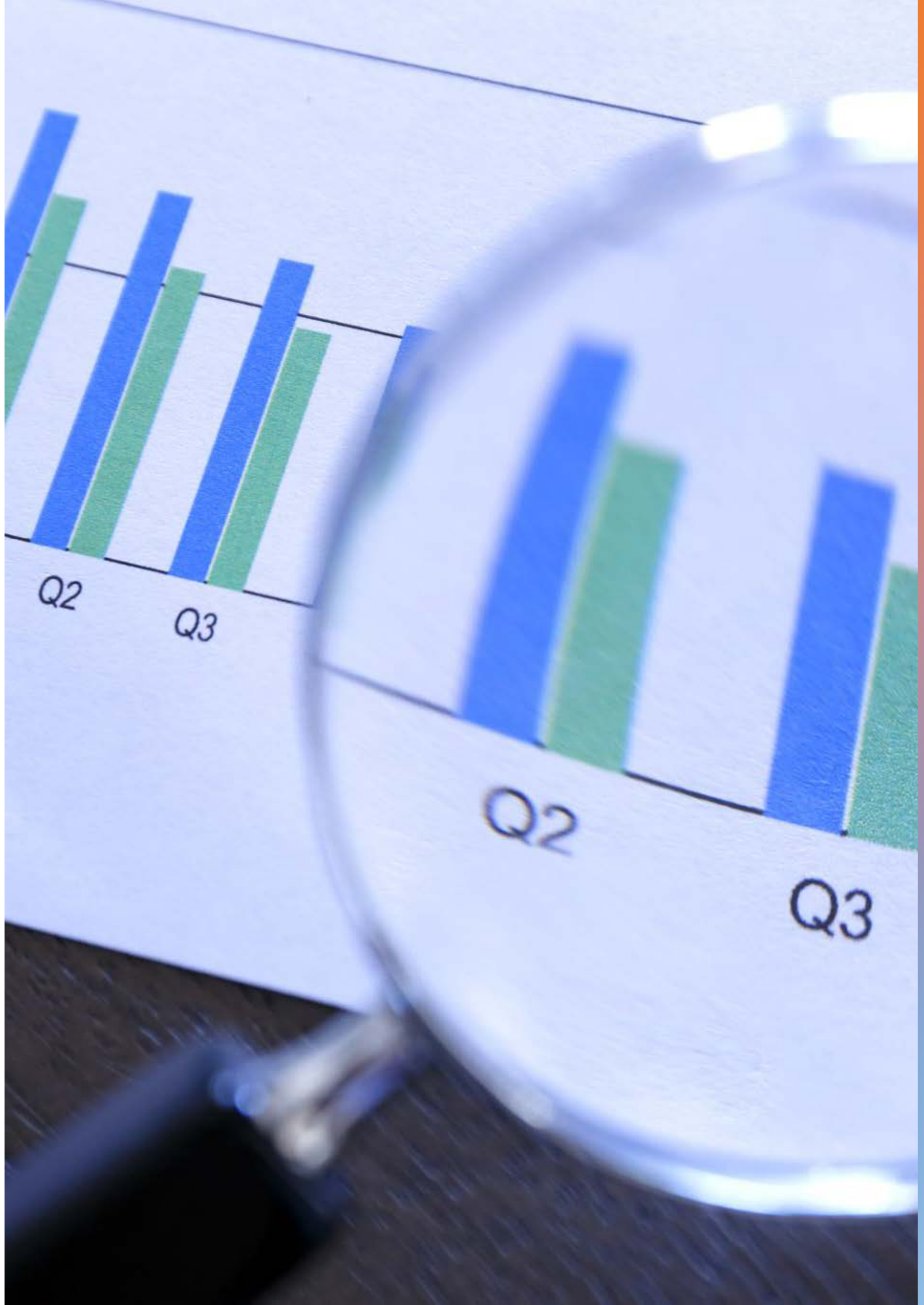


# Real-World Applications and Success Stories of Intelligence-Based Models

**Healthcare: Predictive Analytics in Patient Care**

# Real-World Applications and Success Stories of Intelligence-Based Models

**Finance: AI in Risk Management and Fraud Detection**



# Real-World Applications and Success Stories of Intelligence-Based Models

**Retail: Personalized Customer Experiences**

# Real-World Applications and Success Stories of Intelligence-Based Models

- **Manufacturing: Predictive Maintenance**

# Real-World Applications and Success Stories of Intelligence-Based Models

- **Environmental Science: Climate Change Analysis**

# System supervision/ Maintenance

- **Continuous Monitoring:** Automated monitoring for performance, errors, and potential issues.
- **Regular Updates and Upgrades:** Implementing scheduled updates to improve algorithms and incorporate technological advancements.
- **Data Quality Management:** Ensuring accuracy and integrity of input data through validation and cleaning processes.
- **Fault Detection and Resolution:** Protocols for quick detection and effective resolution of system faults or anomalies.
- **Performance Evaluation:** Regular assessment against key performance indicators (KPIs) and objectives, with adjustments as needed.
- **User Support and Training:** Providing ongoing support and training for system users, including resources like helpdesks and manuals.

# Conclusion

**Streamlined Development Process:** Model-Based Design significantly streamlines the development process from conceptualization to implementation.

**Enhanced Collaboration:** Facilitates better collaboration between interdisciplinary teams through shared models and simulations.

**Error Reduction:** Reduces the likelihood of errors by allowing early detection and resolution during the design phase.

**Cost-Effective:** Minimizes development costs by reducing the need for physical prototypes and iterative testing.

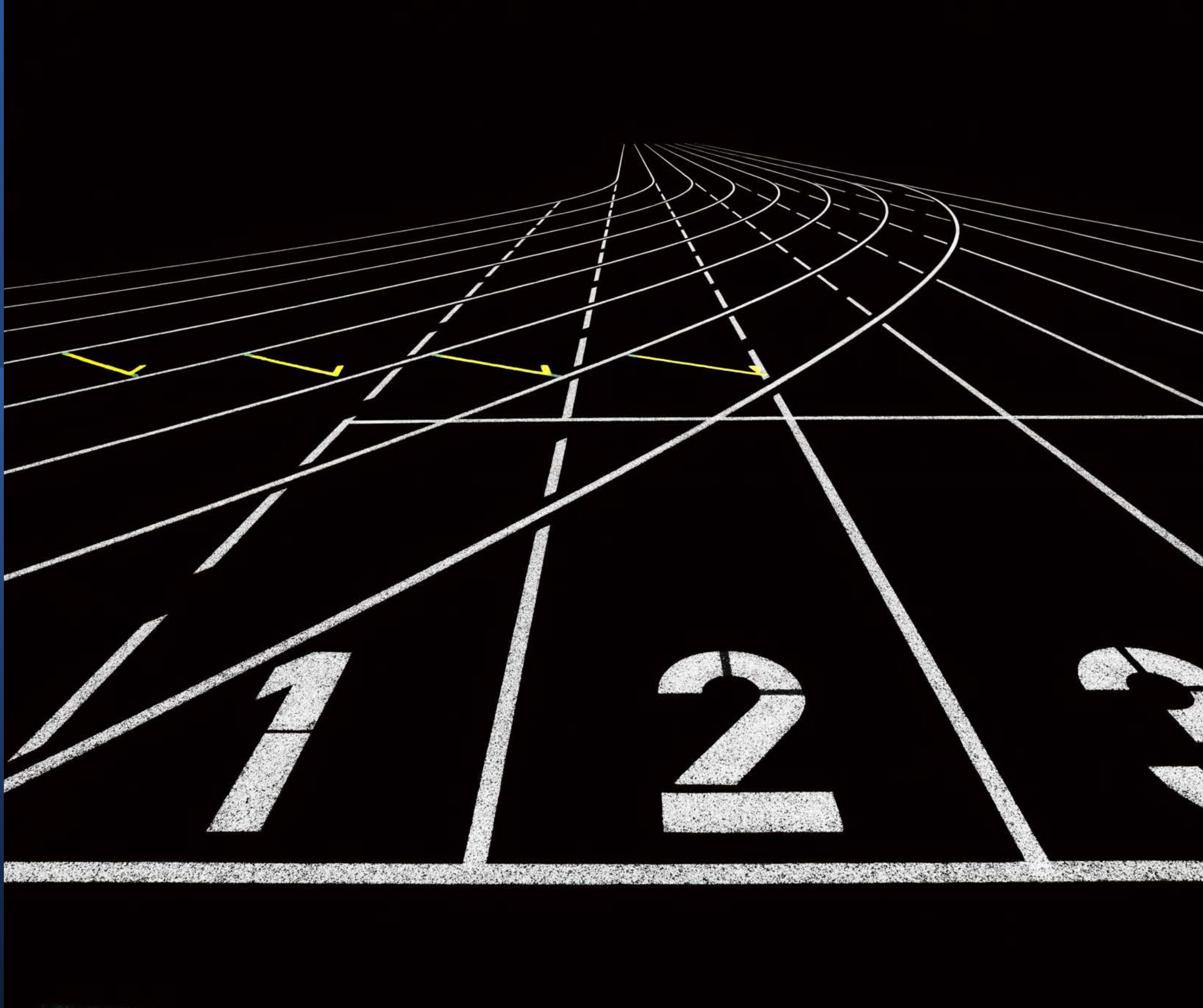
**Faster Time-to-Market:** Accelerates the overall product development cycle, enabling faster time-to-market.

**Scalability and Flexibility:** Offers scalability and flexibility in design, accommodating changes and updates efficiently.

**Improved Quality and Reliability:** Enhances the quality and reliability of the final product through thorough testing and validation of the model.

**Supports Innovation:** Enables innovation by allowing designers to easily experiment with and evaluate new ideas and concepts.

# Exercise



# Exercise : Design and Simulation of a Smart Thermostat System

**Objective:** To design a model of a smart thermostat system using Model-Based Design principles and simulate its behavior under different environmental conditions.

**Tools Required:** Software for modeling and simulation such as MATLAB/Simulink, or any other MBD software.

**Exercise Steps:**

**1.Conceptualization:**

- Define the functional requirements for a smart thermostat system (e.g., temperature regulation, user interface, connectivity).

**2.Model Creation:**

- Step 1: Design a block diagram model of the thermostat system. Include components such as temperature sensors, control logic, user interface, and actuators.
- Step 2: Define the parameters and algorithms for temperature control logic (e.g., PID controller).

**3.Simulation Setup:**

- Set up various environmental scenarios to test the thermostat, such as varying outside temperatures, different user settings, and simulated faults.

**4.Run Simulations:**

- Conduct simulations to observe how the thermostat model responds to the different scenarios.
- Document the system's behavior and performance in each case.

**5.Analysis and Iteration:**

- Analyze the simulation results to identify any issues or areas for improvement in the model.
- Make necessary adjustments to the model and rerun simulations to validate changes.

**6.Reporting:**

- Prepare a report summarizing the design process, simulation results, analyses, and any iterations made to the model.
- Include insights on how Model-Based Design facilitated the development process.

**Expected Outcomes:**

A functioning model of a smart thermostat that meets the defined requirements.

An understanding of how the system behaves under various conditions.

Insights into the advantages of using Model-Based Design in developing and testing a complex system.

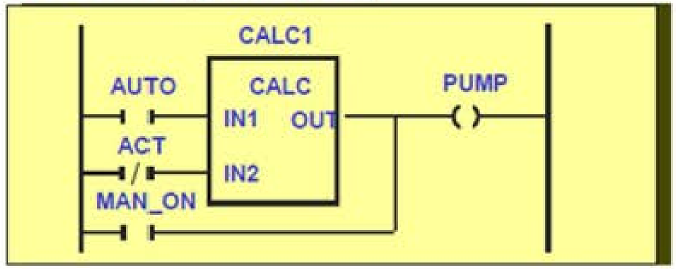
# Lecture 03 Industrial programing languages

# PLC Languages: IEC 61131

## Instruction List (IL)

```
A: LD   %IX1 (* PUSH BUTTON *)
   ANDN %MX5 (* NOT INHIBITED *)
   ST   %QX2 (* FAN ON *)
```

## Ladder Diagram (LD)

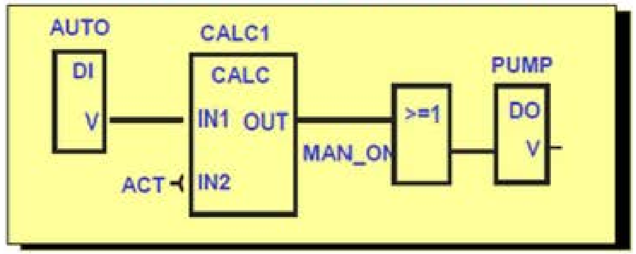



## Structured Text (ST)

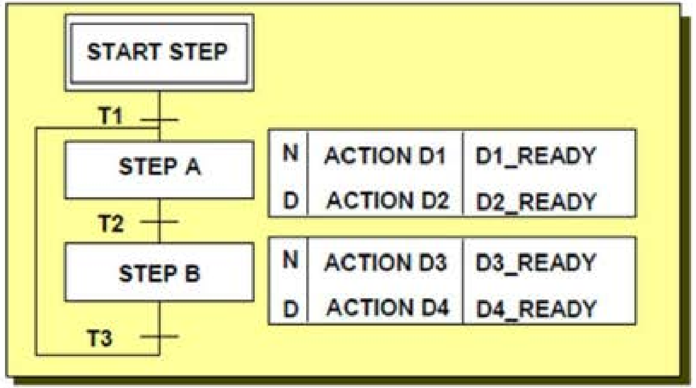
```
VAR CONSTANT X : REAL := 53.8 ;
Z : REAL; END_VAR
VAR aFB, bFB :  FB_type; END_VAR

bFB(A:=1, B:='OK');
Z := X - INT_TO_REAL (bFB.OUT1);
IF Z>57.0 THEN aFB(A:=0, B:="ERR");
ELSE  aFB(A:=1, B:="Z is OK");
END_IF
```

## Function Block Diagram (FBD)




## Sequential Flow Chart (SFC)

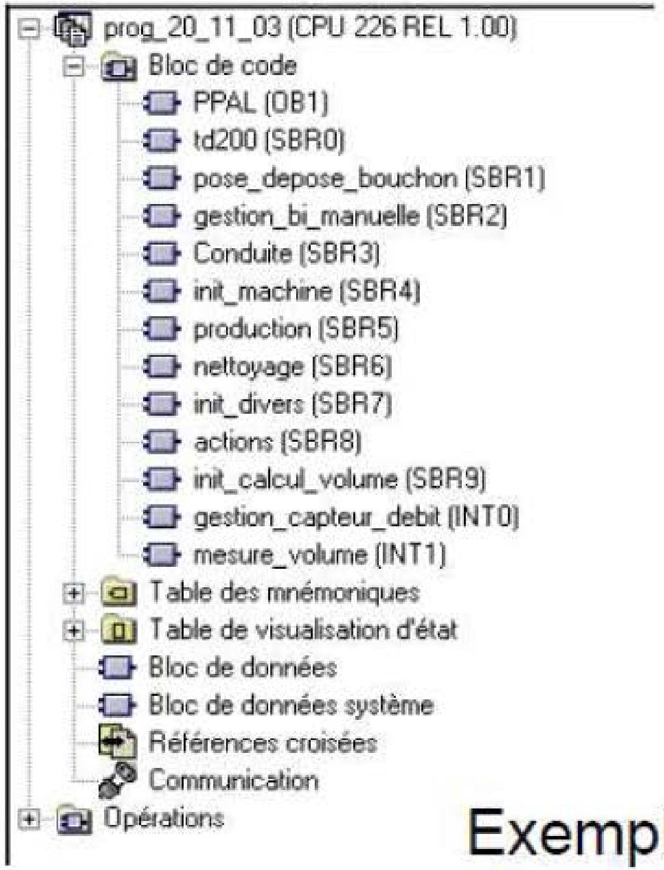


# Comparaison des langages

| LANGAGE | AVANTAGES | INCONVENIENTS |
|---|---|---|
| **LD** | facile à lire et à comprendre par la majorité des électriciens<br>langage de base de tout PLC | suppose une programmation bien structurée |
| **FBD** | Très visuel et facile à lire | Peut devenir très lourd lorsque les équations se compliquent |
| **ST** | Langage de haut niveau (langage pascal)<br>Pour faire de l'algorithmique | Pas toujours disponible dans les ateliers logiciels |
| **IL** | langage de base de tout PLC type assembleur | très lourd et difficile à suivre si le programme est complexe<br>Pas visuel. |
| **SFC** | Description du fonctionnement (séquentiel) de l 'automatisme.<br>Gestion des modes de marches<br>Pas toujours accepté dans l'industrie… | Peu flexible |

# Multi-langages, multi-programmes !

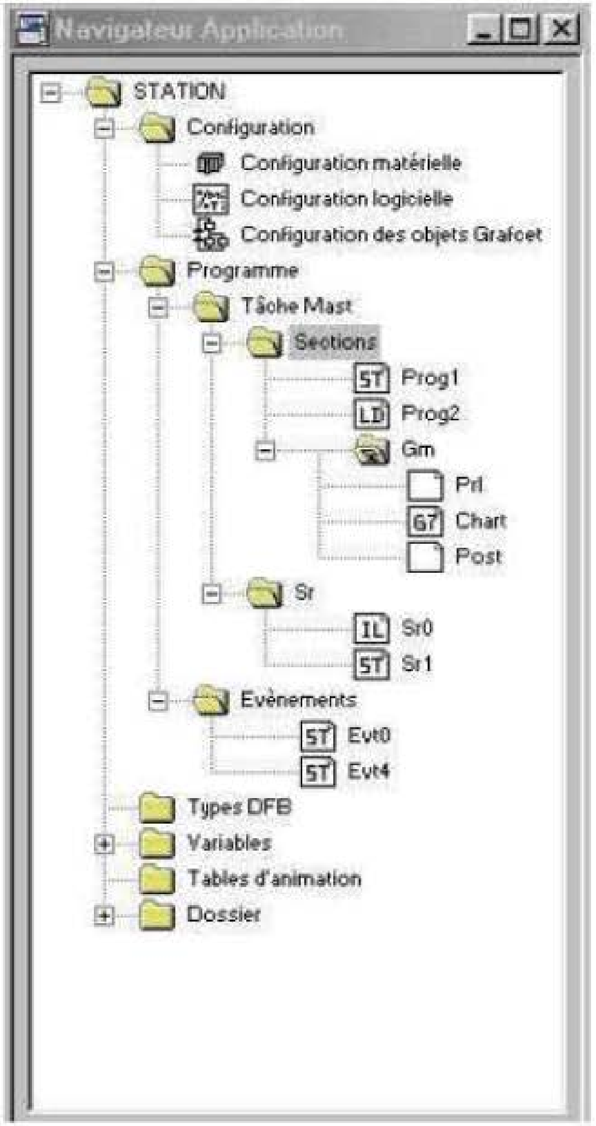

Exemple
Isagraf



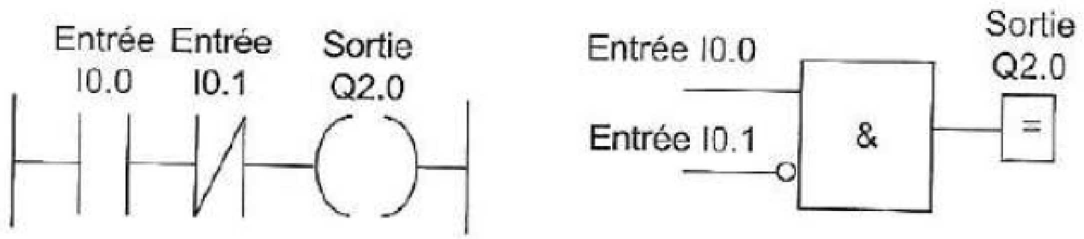

Exemple
Siemens

# Multi-langages, multi-programmes !

Exemple Schneider

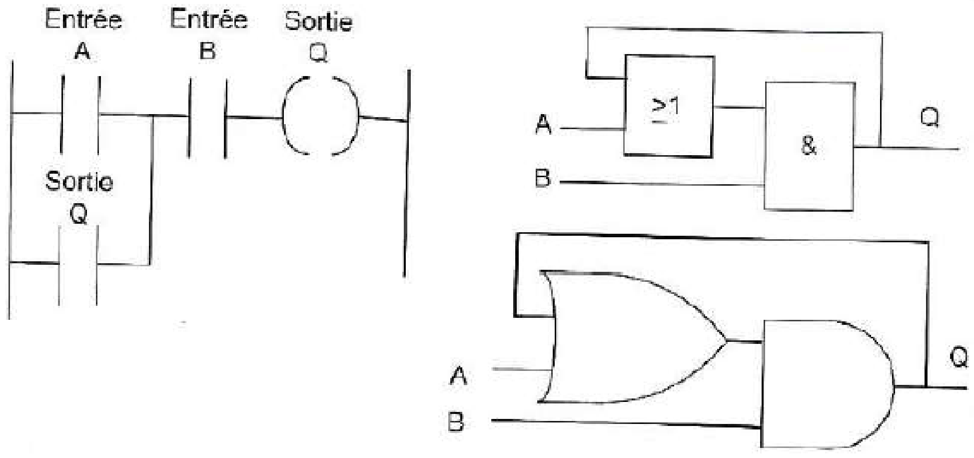


# Equivalence between Ladder diagrams (LD) and Function Block diagrams (FBD)

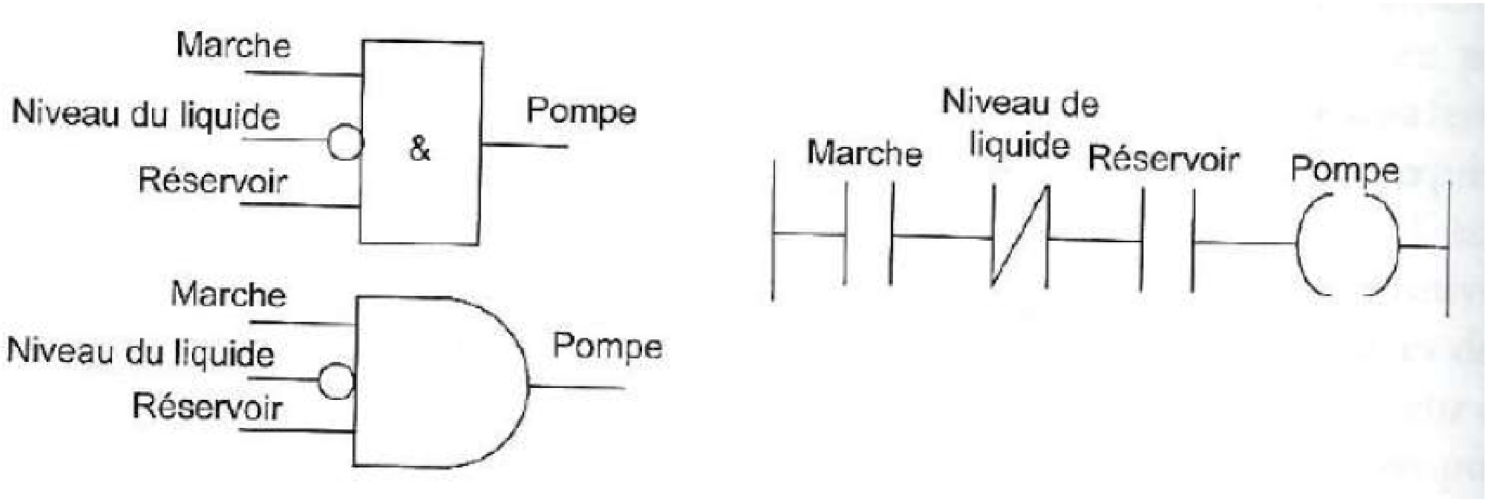



Latching circuit:
The output Q is fed back as an input (this is a "relay" that stores the state of Q)

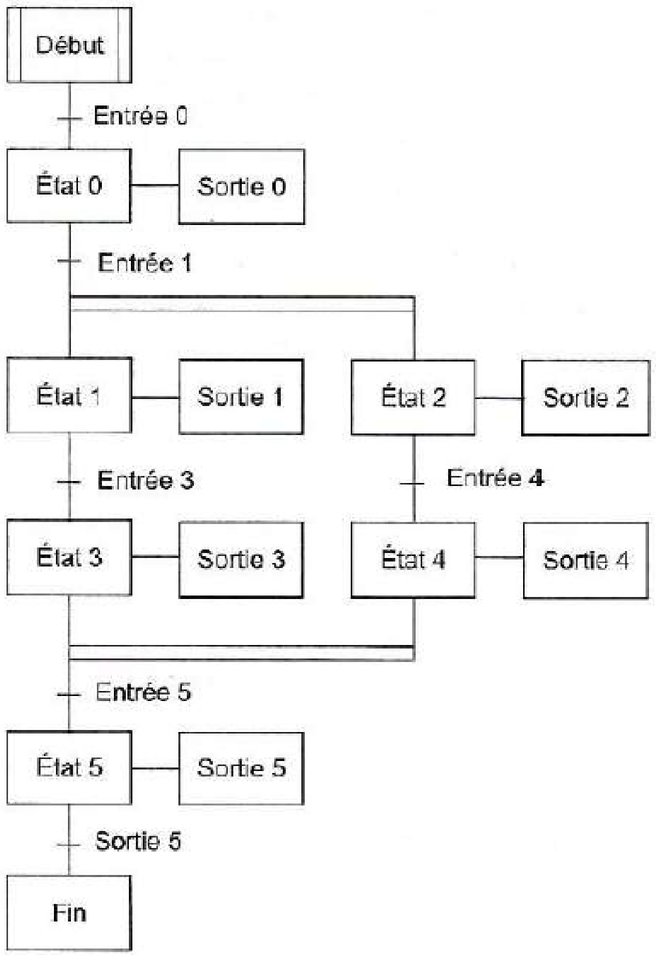
Circuit à verrouillage

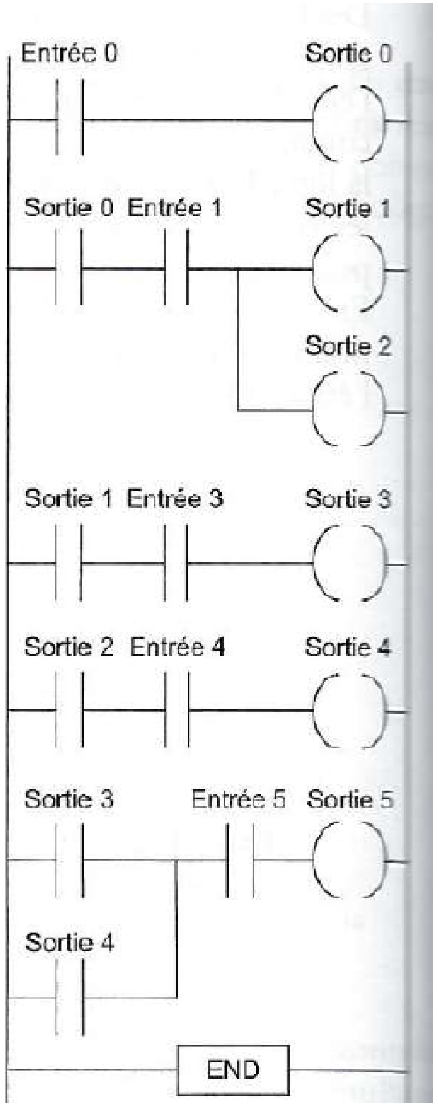


# Equivalence between Ladder diagrams (LD) and Function Block diagrams (FBD)



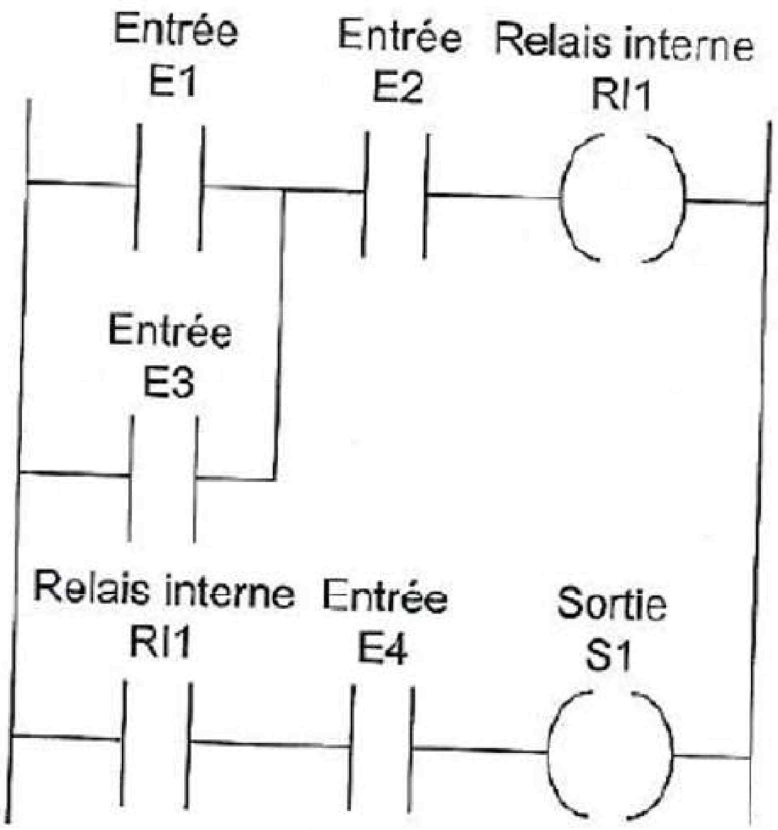
# Equivalence between Ladder diagrams (LD) and Sequential Flow Chart (SFC)



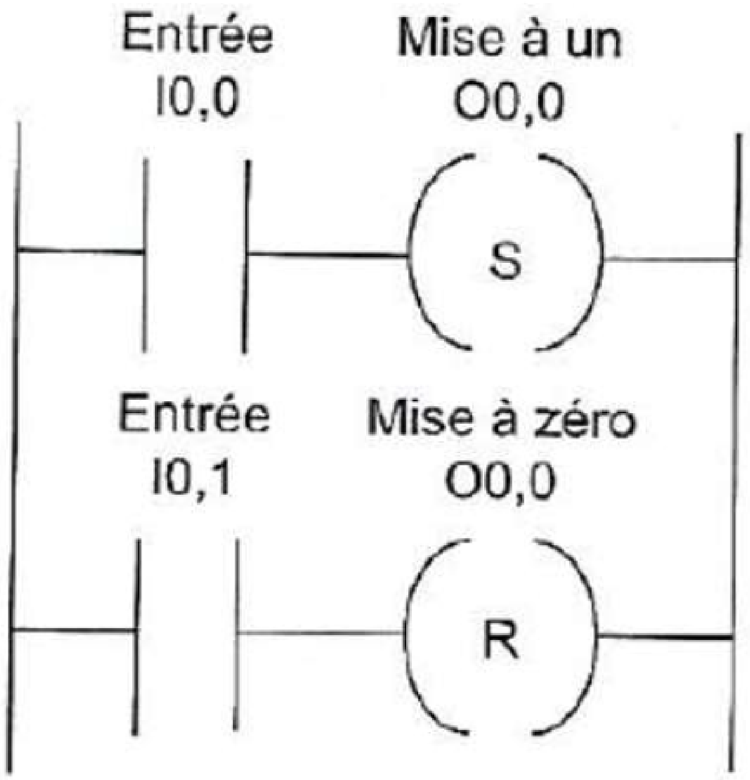



# Internal Relay

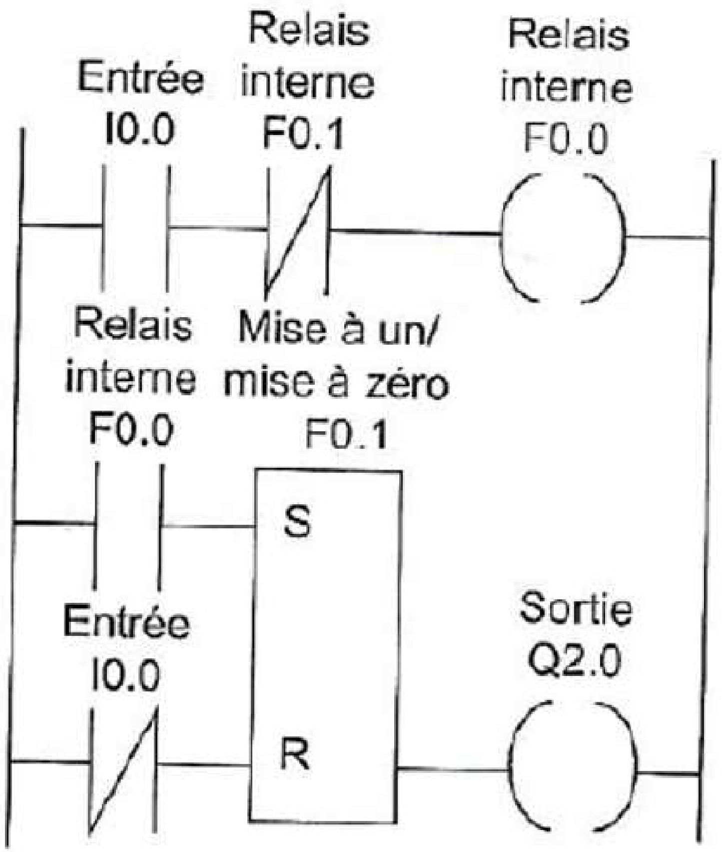
- An internal relay allows to keep (memorize) a value (bit)
- It is equivalent to a memory
- It is <> from an I/O
- In Siemens, it can be managed as a Flag (F)



# Setting to 0 and setting to 1 of an internal relay



# Flip-flop solution (FR : bascule)




(a)

# Les constructeurs

Schneider Electric

WAGO

ABB

SIEMENS

PHOENIX CONTACT

Honeywell

Allen-Bradley

Rockwell Automation

When good enough just isn't good enough

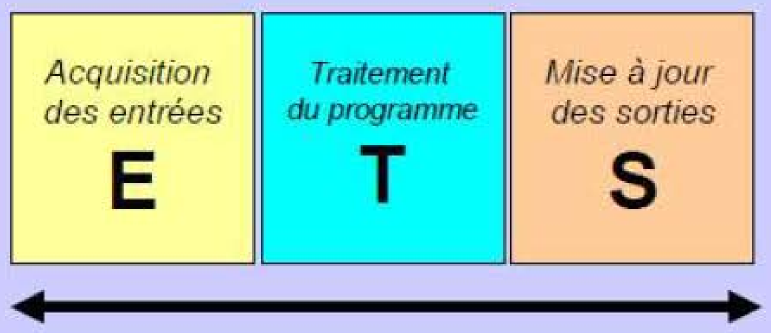
Weidmüller

HITACHI

Inspire the Next

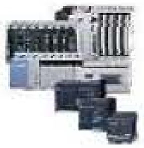
NAiS Matsushita Electric Works-
Automation Controls Company

# Tâche Automate

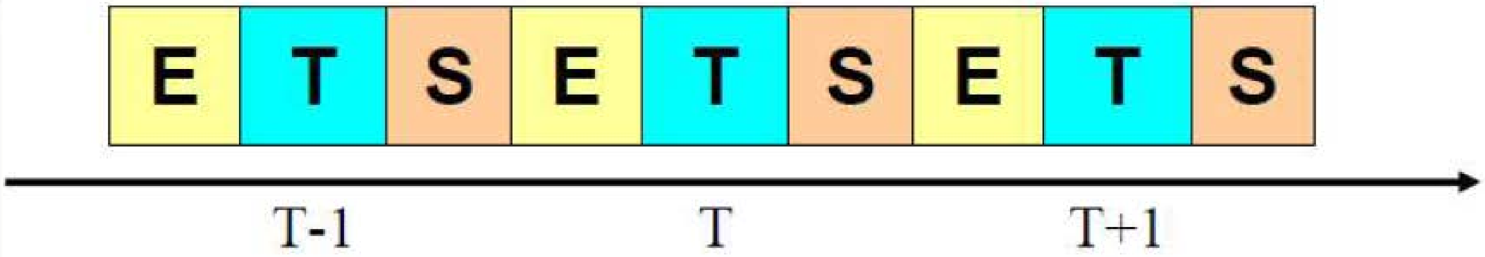



**Temps de cycle**

$T_C = T_E + T_T + T_S$



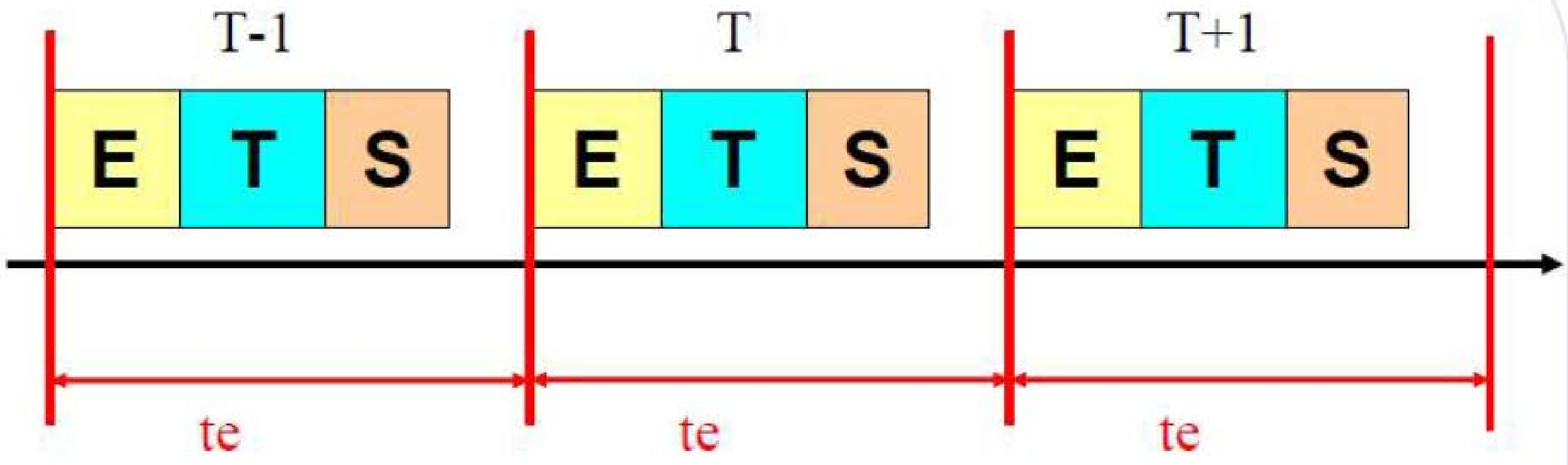
# fonctionnement mono-tâche cyclique (asynchrone)





Ce type de fonctionnement consiste à enchaîner les cycles les uns après les autres.

# fonctionnement mono-tâche périodique (synchrone)




Dans ce mode de fonctionnement, l'acquisition des entrées, le traitement du programme et la mise à jour des sorties s'effectue de façon périodique te ms selon un temps défini par configuration API .

# Exercise

- Discuss the logic difference between a sequential program and PLC concurrent behaviors.

# 1501217 Model Based Design

Program: Bachelor program in Computer Engineering

Credit: 3(2-2)    Lecture: 30 Hours    Lab: 30 Hours

FACTORI 4.0 Erasmus +

Co-funded by the Erasmus+ Programme of the European Union

1st Semester, Academic Year: 2023


Assoc. Prof. Punnarumol Temdee, Ph.D.

Asst. Prof. Roungsan Chaisricharoen, Ph.D.

Asst. Prof. Santichai Wicha, Ph.D.

Lect. Chayapol Kamyod, Ph.D.



This course has been modified in the framework of an Erasmus + project: Asean Factori 4.0 Across South East Asian Nations: From Automation and Control Training to the Overall Roll-out of Industry 4.0

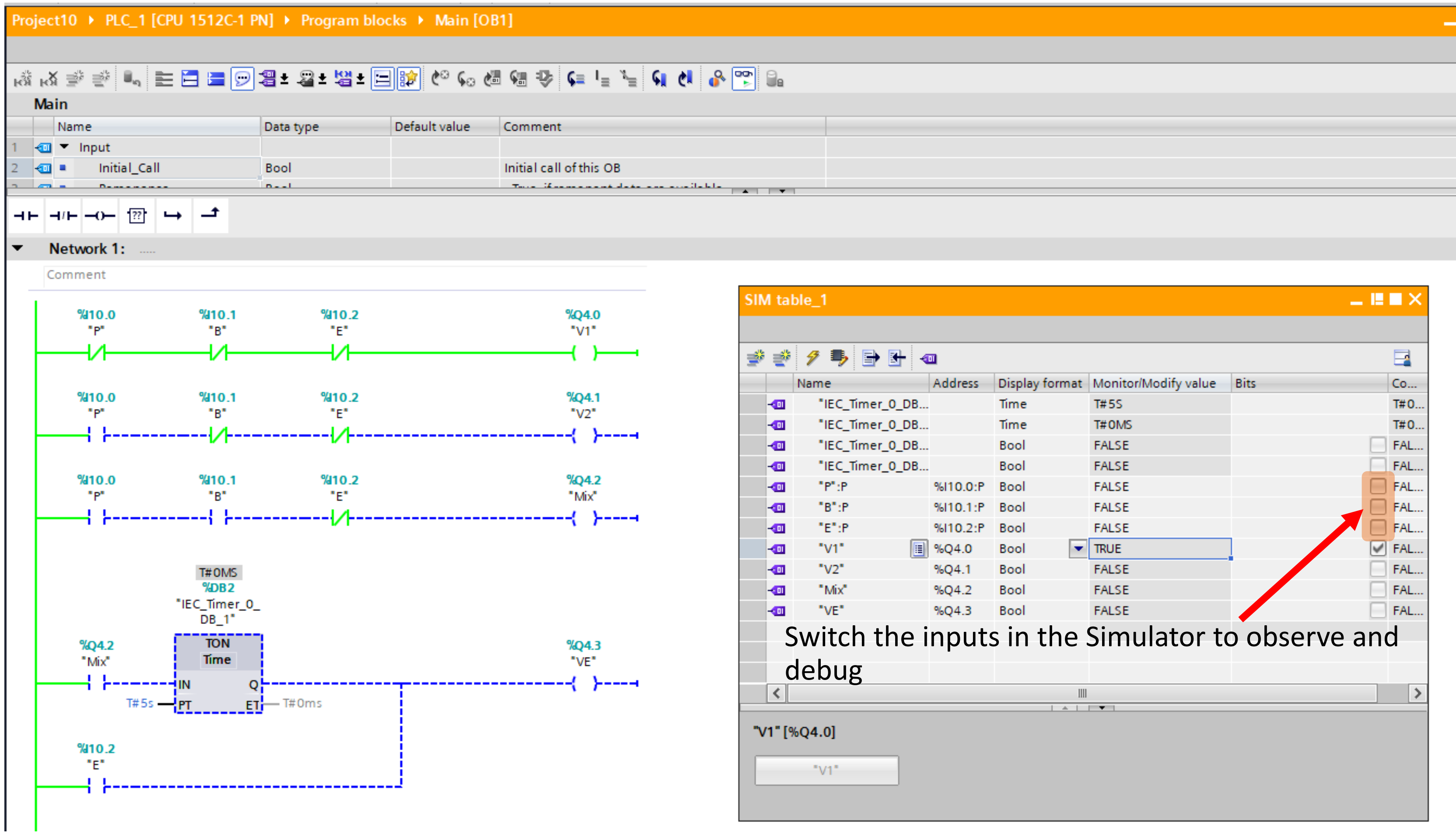
609854-EPP-1-2019-1-FR-EPPKA2-CBHE-JP

# Lab 01

Time-based model of digital outputs

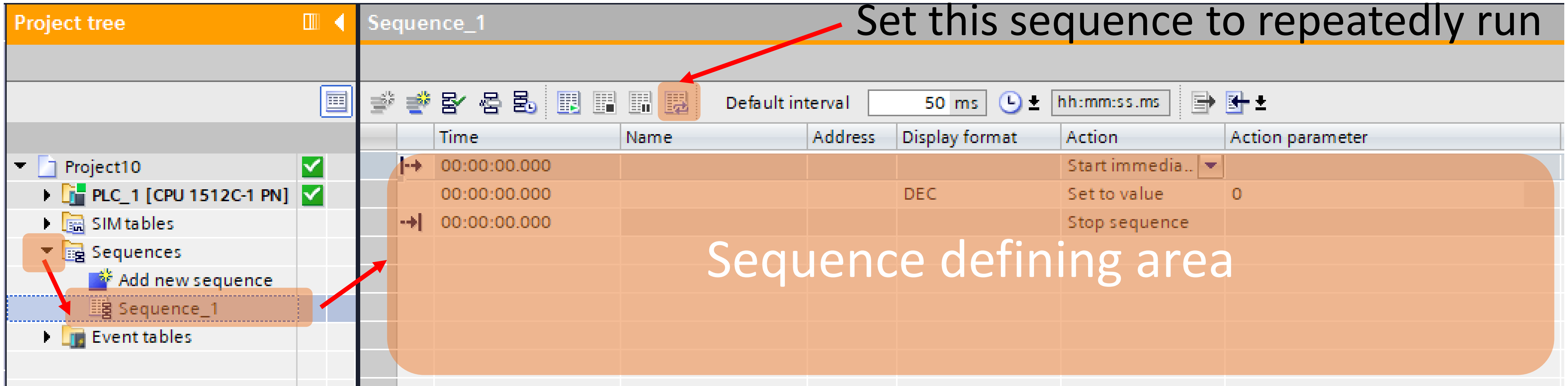
# Manual Simulation

- User has to directly set inputs
  - Not convenient
  - Not realistic
  - Can be missed in case of a complex system where more than one inputs are to be set in parallel
- Two basic options
  - Set sequence in the simulator
  - Programming a ladder logic that emulates responses



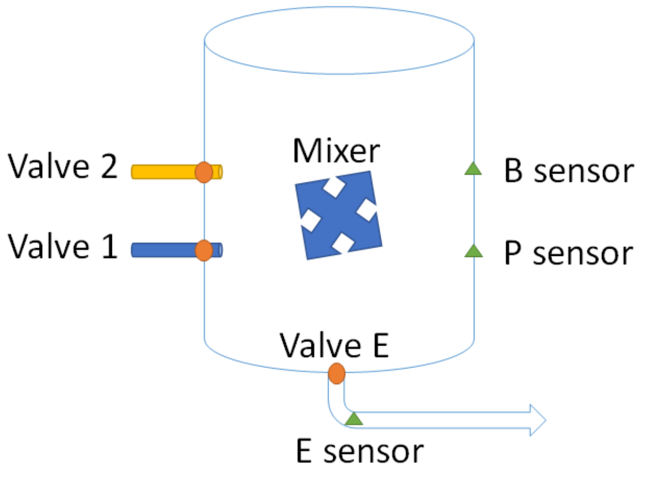
# Simulation Sequence

- Siemens PLC SIM allows users to set the sequence of inputs/outputs in the simulator
- The sequence is time-based, the user has to set
  - When and what parameter to be set
  - Multiple sequences can be defined but you can run just one sequence at any instance
  - To start a sequence, there are two options
    - Immediately start
    - Wait for triggered condition



# Response Designing

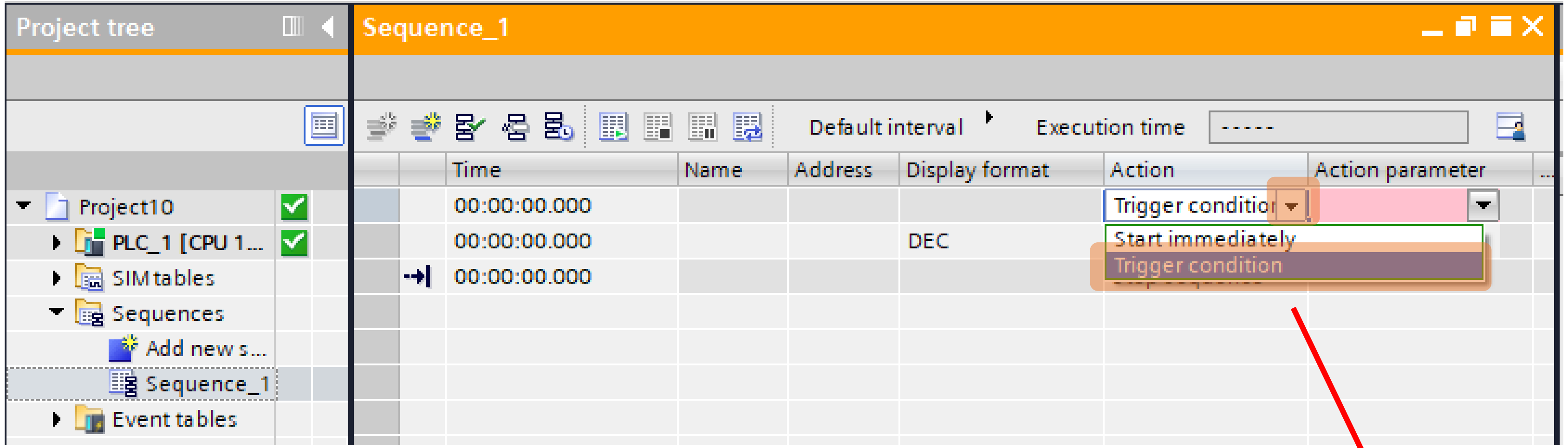
- The defined response must be relevant to the practical process
- Timing and sequencing must be properly study and verify
- Only inputs to the PLC are to be sequenced
  - Outputs are determined by the PLC operations

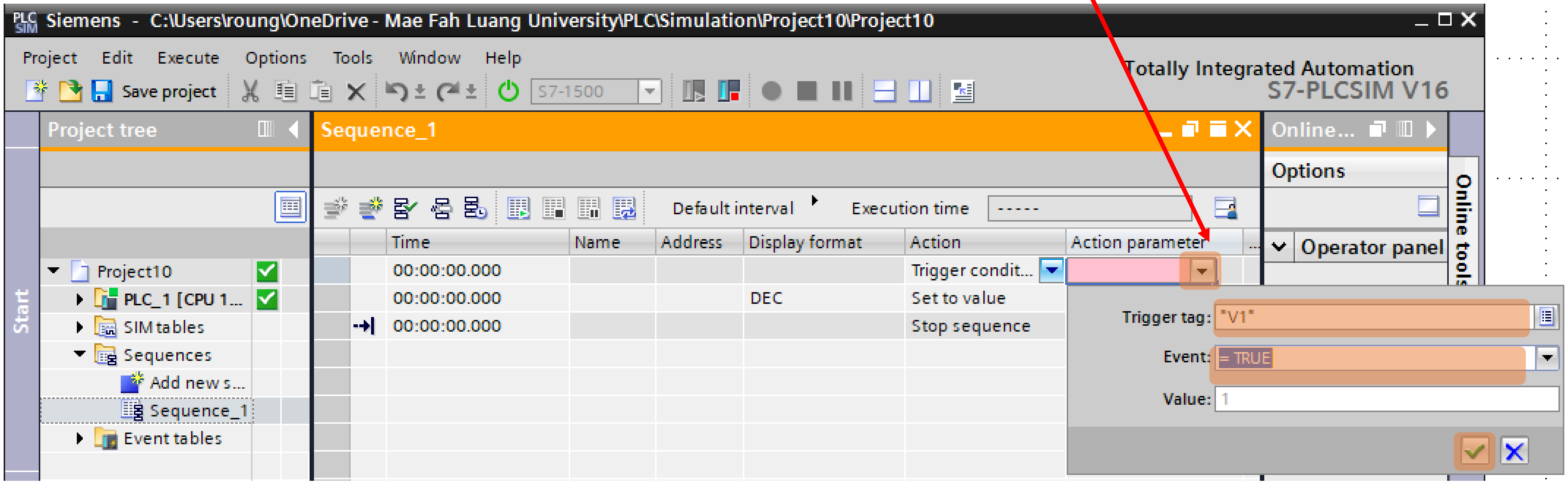



Example: Mixing tank problem

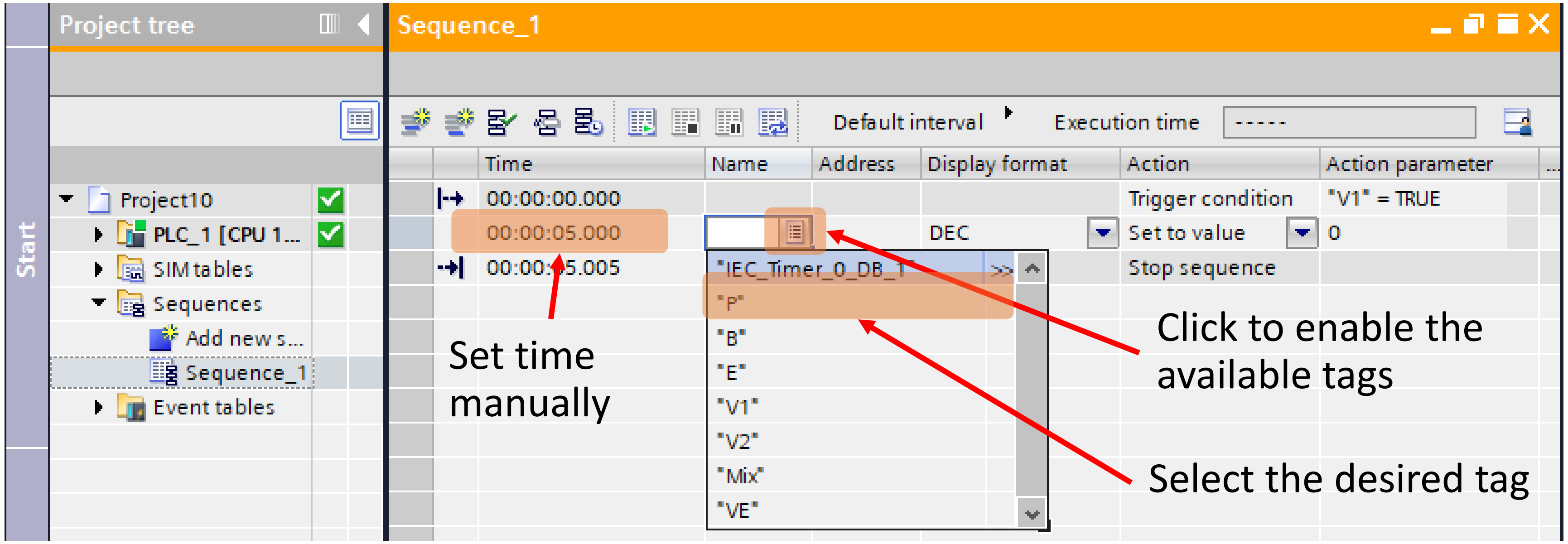
- Triggered as V1 is on
- P sensor is on after V1 is on for 5s
- B sensor is on after V2 is on for 5s
- E sensor is on after the mixer is on for 5s
  - Because VE is on after the mixer is run for 5s
- B sensor is off right after VE is on
- P sensor is off after VE is on for 5s
- E sensor is off after VE is on for 10s
- As all sensors are off, PLC set V1 to on and the sequence is triggered again

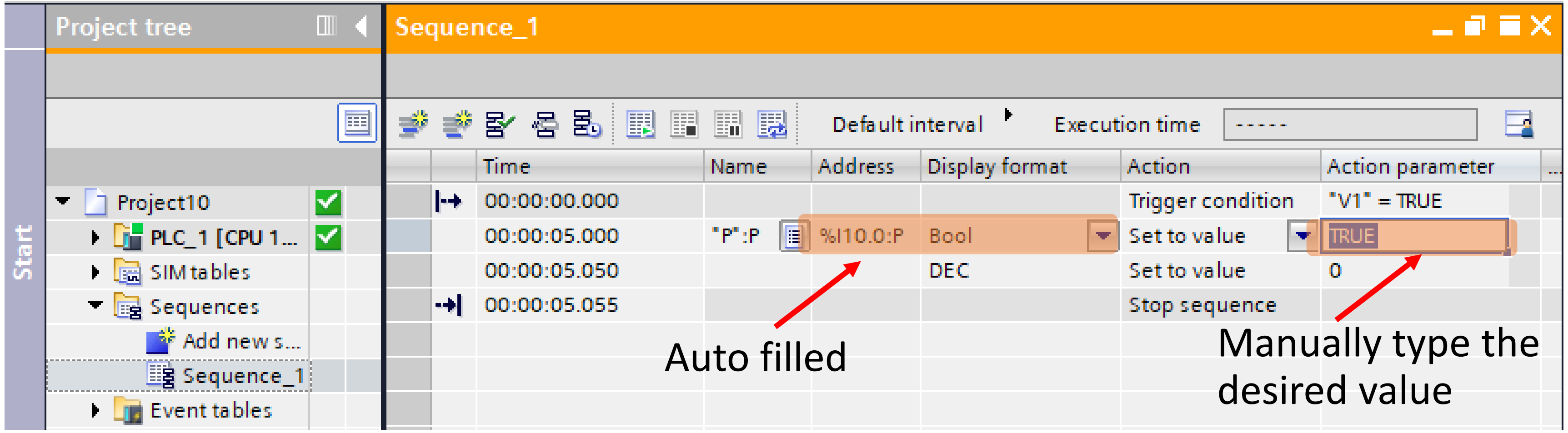
# Setting the sequence – Triggering condition



# Adding the following sequences

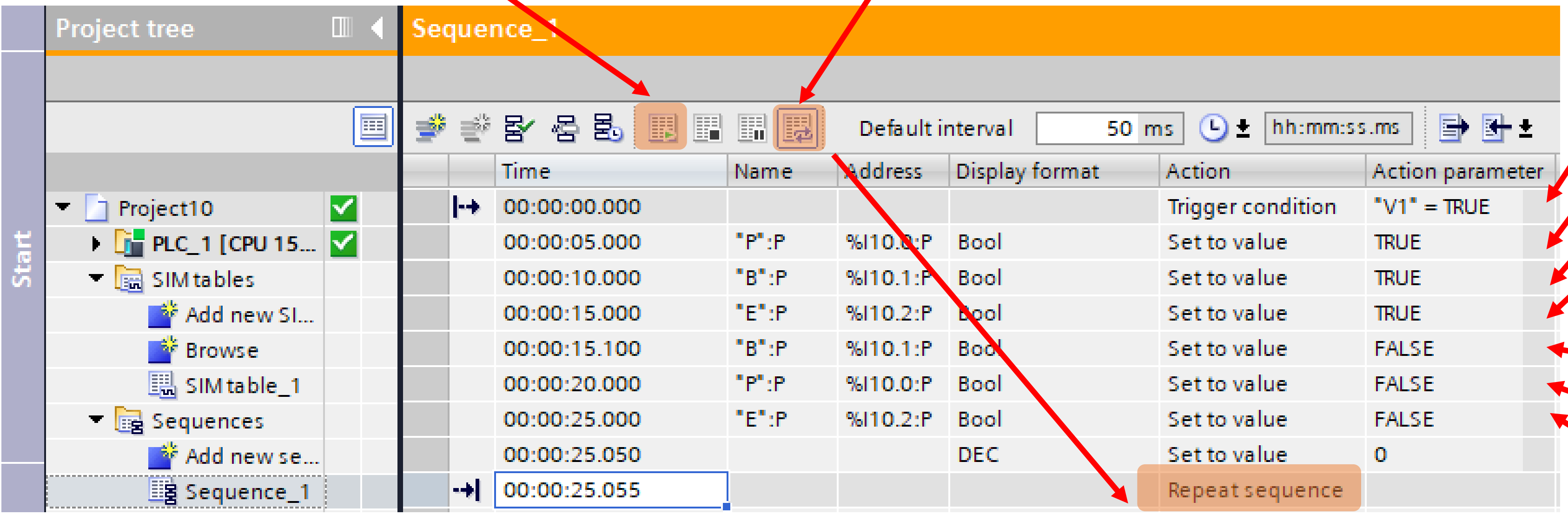


# Complete the sequence

Click to run this sequence

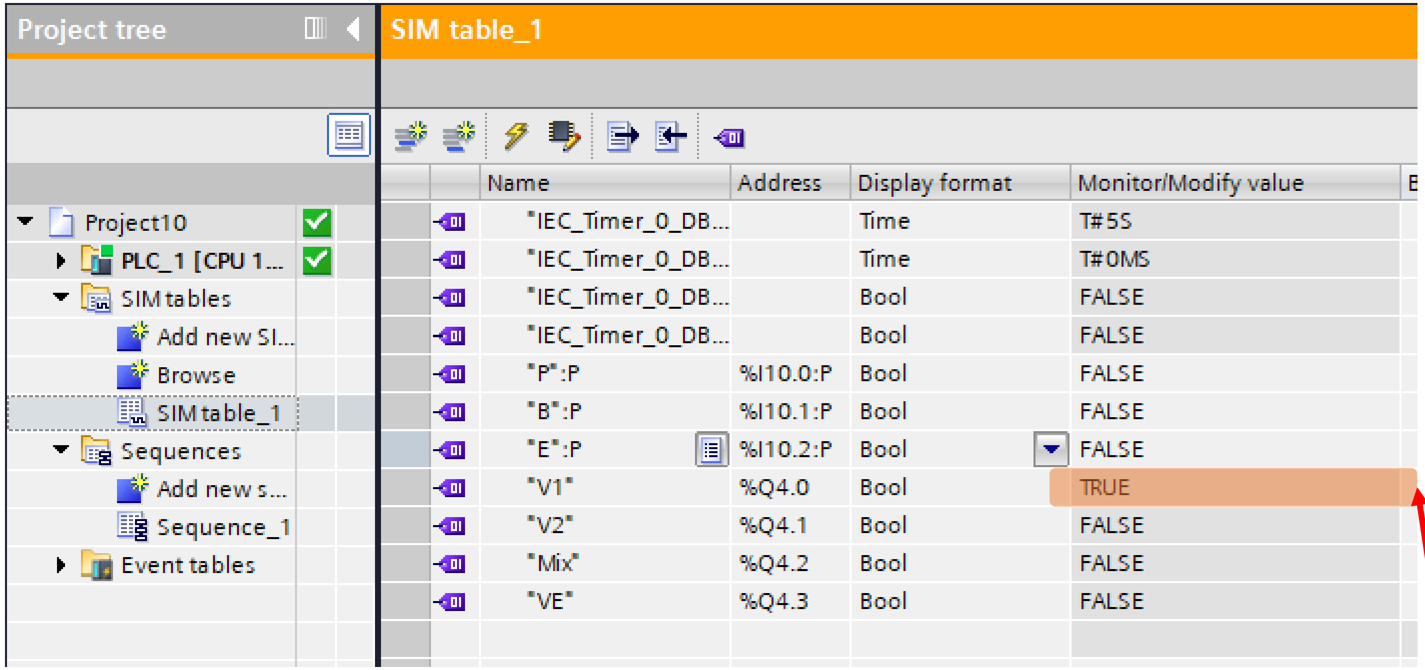
As all sensors are off, PLC set V1 to on and the sequence is triggered again




Example: Mixing tank problem

- Triggered as V1 is on
- P sensor is on after V1 is on for 5s
- B sensor is on after V2 is on for 5s
- E sensor is on after the mixer is on for 5s
  - Because VE is on after the mixer is run for 5s
- B sensor is off right after VE is on
- P sensor is off after VE is on for 5s
- E sensor is off after VE is on for 10s

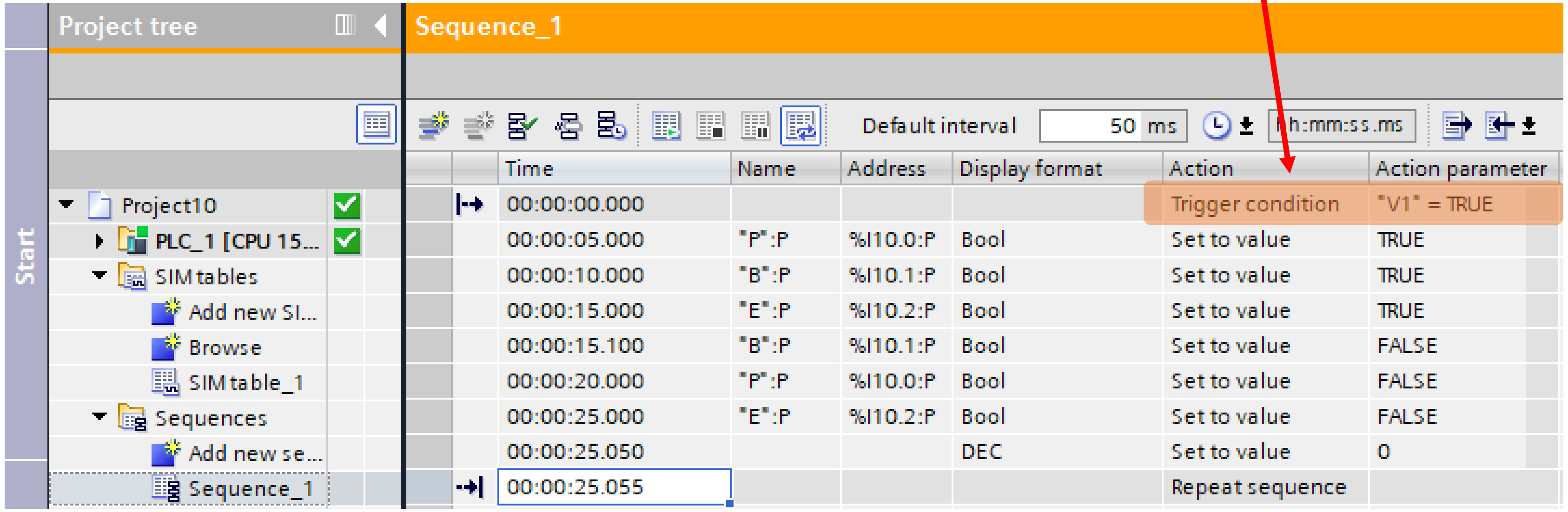
# Initiating the sequence




Open the SIM table to check whether the triggering condition is met?

- In this case, the sequence will start if and only if V1 is switched from ‘FALSE’ to ‘TRUE’

V1 is already ‘TRUE’ before starting the sequence, the sequence will not be triggered




So, we have to manually run the first round to bring V1 off and then on again which will trig the sequence

# Exercise

- Design and simulate a mixer system with 2 mixers
  - The first mixer run 5 seconds
  - The second mixer run 10 seconds
  - The second mixer run after the first mixer finished

# Lab 02

Model of digital outputs via ladder diagram

# Response emulator via Ladder logic

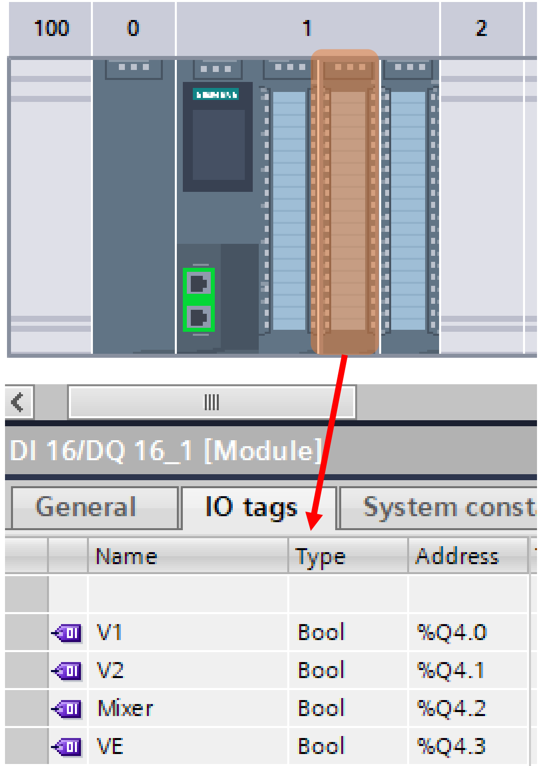
- Using the simulation sequence to simulate a practical response can be quite limited and does not reflect the real process
- Example: E sensor is on after B is on for 5s

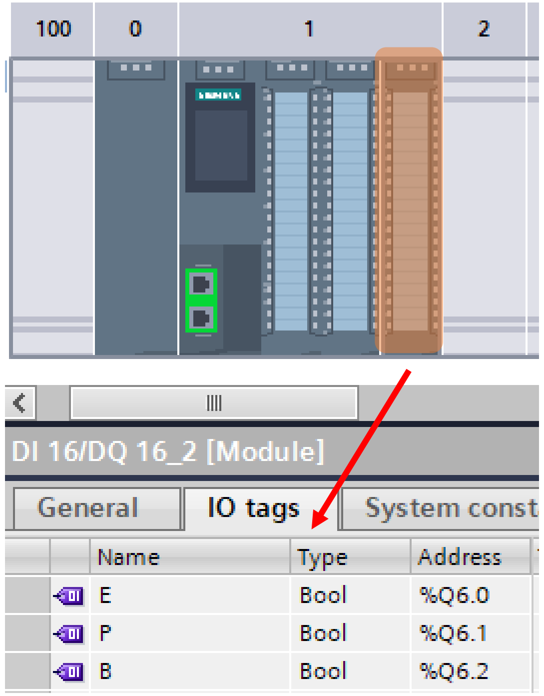
| | | | | | |
|---|---|---|---|---|---|
| 00:00:10.000 | "B":P | %I10.1:P | Bool | Set to value | TRUE |
| 00:00:15.000 | "E":P | %I10.2:P | Bool | Set to value | TRUE |

  - It is based on the assumption that the mixer is run right after ‘B’ is on and it required 5s to open VE which will turn ‘E’ on.
  - So, these hidden processes is not shown in the sequence
  - In a complicate case, it can cause over simplified response simulator, which can be serious as the main purpose of simulation is to verify the designing solution.
- So, instead of sequence simulation, the ladder logic can be used to emulate response if there is enough computing resource left in the PLC
  - It also benefits in the term of process checking and troubleshooting as both the control and response sides are collaboratively designed and verified each other

# Preparation

- The first step to do is replacing the exist input tags to be output tags
  - PLC is used to control output not the input
  - In practice, inputs are to be changed according to real situations, not by PLC
- In emulation, these inputs are to be manipulated by PLC. So, they must be changed to use the PLC's output




The process is simple.

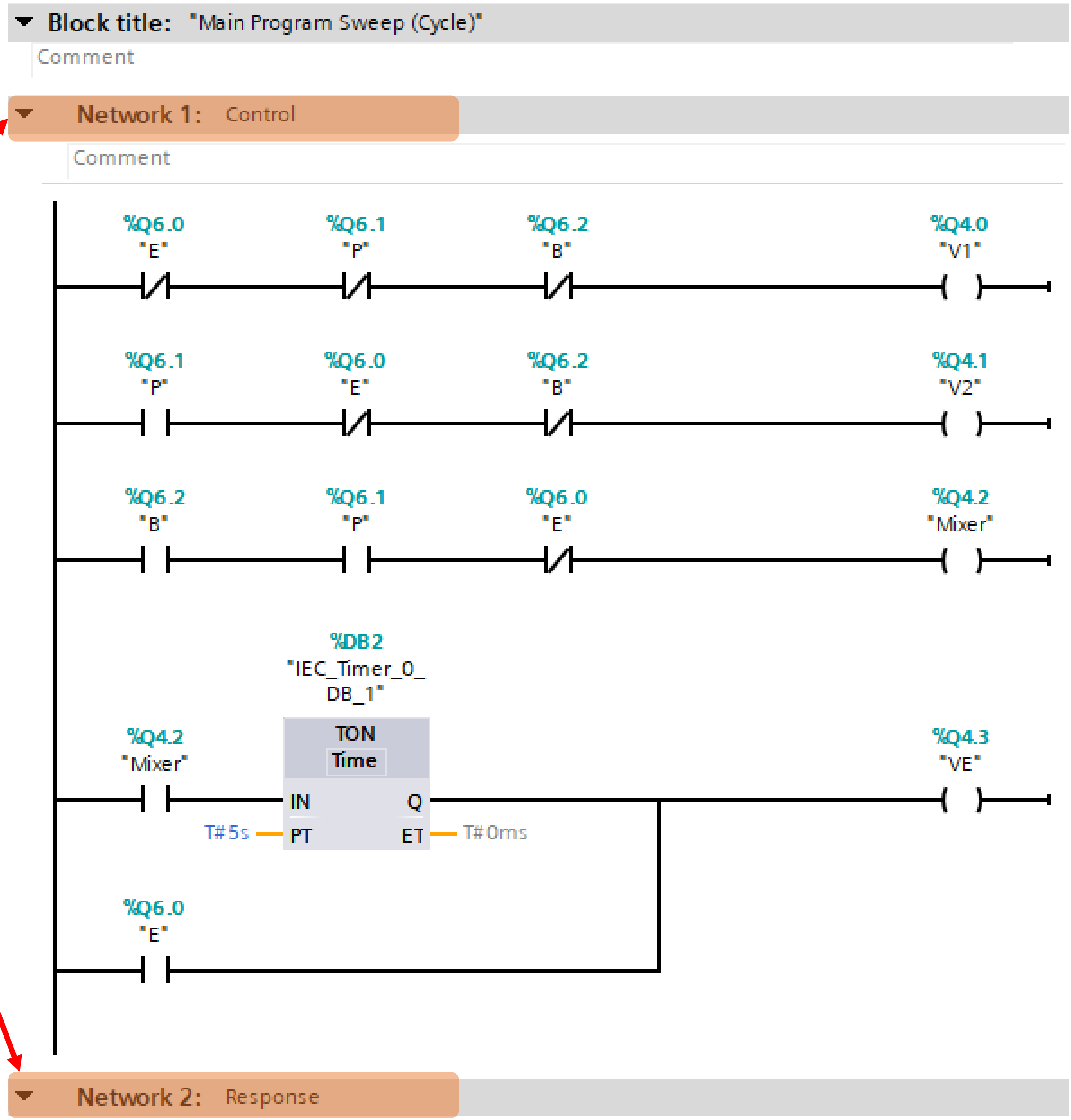
- In the current example, there are three inputs: P, B, and E.
- So, the output of another digital module is employed instead

**Default tag table**

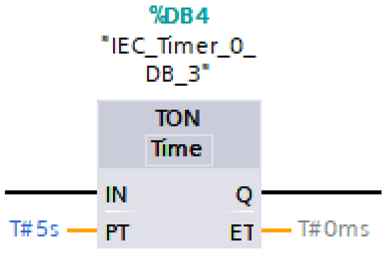
| | Name | Data type | Address |
|---|---|---|---|
| 1 | V1 | Bool | %Q4.0 |
| 2 | V2 | Bool | %Q4.1 |
| 3 | Mixer | Bool | %Q4.2 |
| 4 | VE | Bool | %Q4.3 |
| 5 | E | Bool | %Q6.0 |
| 6 | P | Bool | %Q6.1 |
| 7 | B | Bool | %Q6.2 |

# Separate the control and response logics

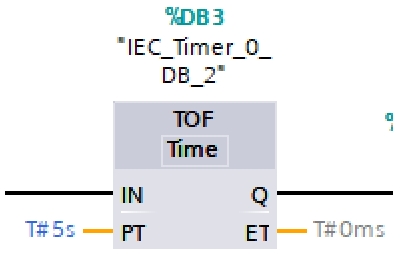
- For the benefit of maintenance and debug
  - The TIA Portal allows dividing the main ladder into networks
- The existed ladder is named ‘Control’
- The new ladder is named ‘Response’
- It is like labelling different parts of a program



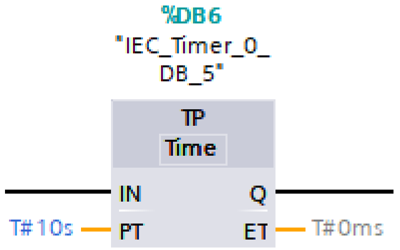
# Important Timers



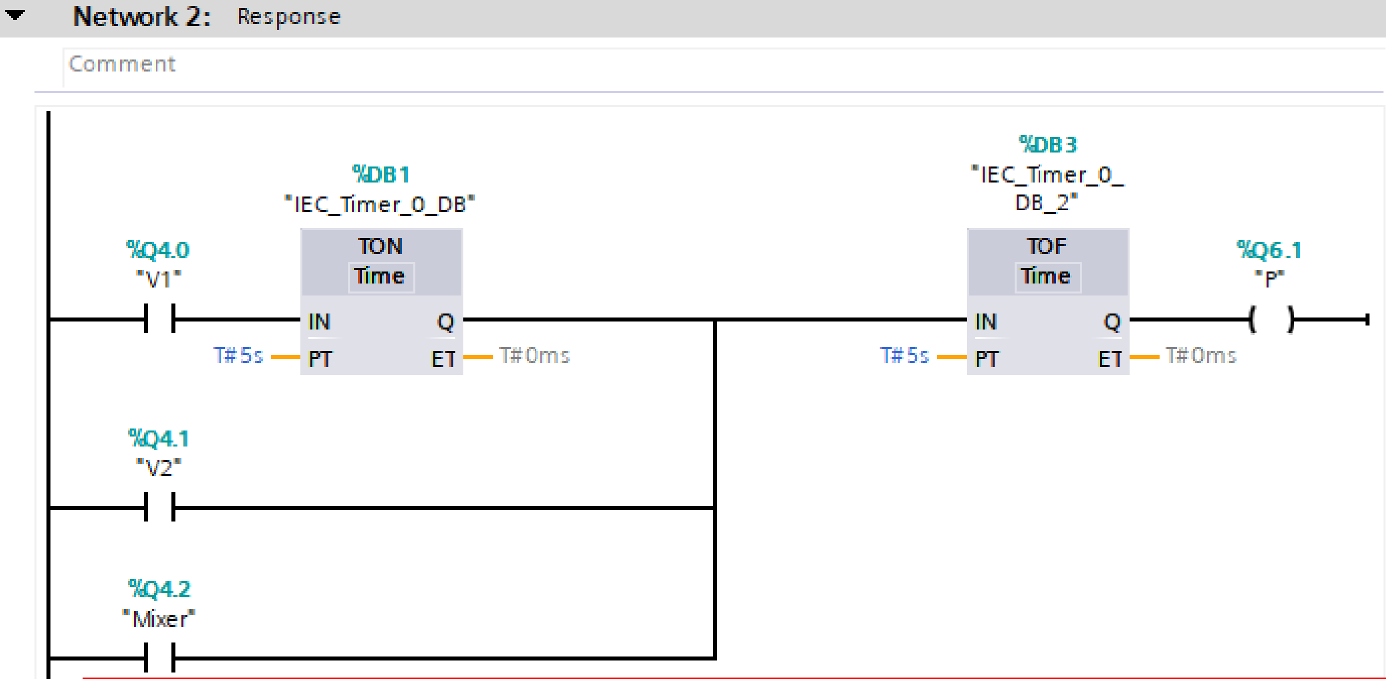

T-ON: If 'IN' is on, Q will be on in 5 seconds




T-Off: If 'IN' is off, Q will be off in 5 seconds




T-Pulse: If 'IN' is on, Q is immediately on for 10 seconds




In filling:

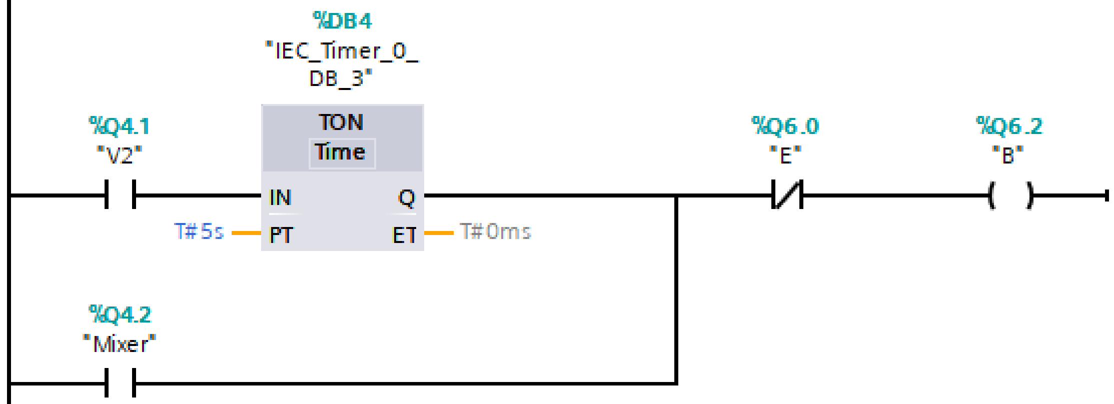
- ‘P’ will be on if V1 is on for 5s
- ‘P’ is still on while V2 is on

In mixing:

- ‘P’ is on while the mixer is on

In draining:

- V1, V2 and Mixer is off while draining
- It takes 5s after the mixer stop for the liquid to be drop below ‘P’




In filling:

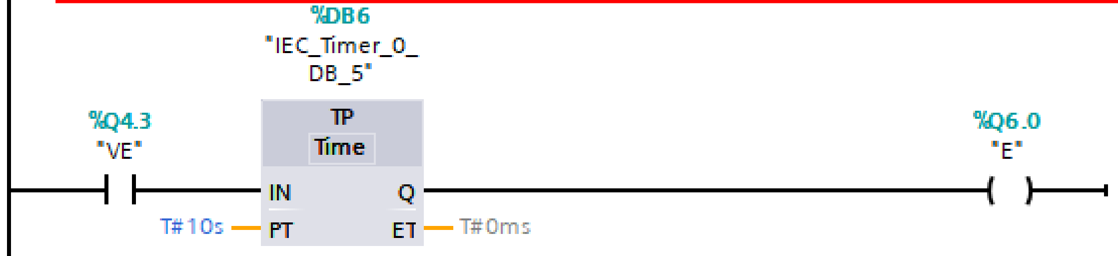
- ‘B’ will be on if V2 is on for 5s

In mixing:

- ‘B’ is on while the mixer is on

In draining:

- V2 and Mixer is off while draining
- As ‘B’ indicate the maximum level, ‘B’ is immediately off if ‘E’ is on




‘E’ is not on while filling and mixing

‘E’ is on just after ‘VE’ is on

‘E’ is on for just 10s as the whole tank is drained

- 5s from ‘B’ to ‘P’ and another 5s from ‘P’ to empty

# Exercise

- Adapt the example into a problem of mixing 3 types of liquid
  - 4 sensors
    - Three level sensors: P, B, and L (P at low level, B at medium level, L at maximum level)
    - A draining sensor: E
  - 10 seconds filling from
    - Empty to P
    - P to B
    - B to L
  - Mixer is working for 10s
- Provide automatic simulation in:
  - Sequencing
  - Ladder emulation

# Lab 03

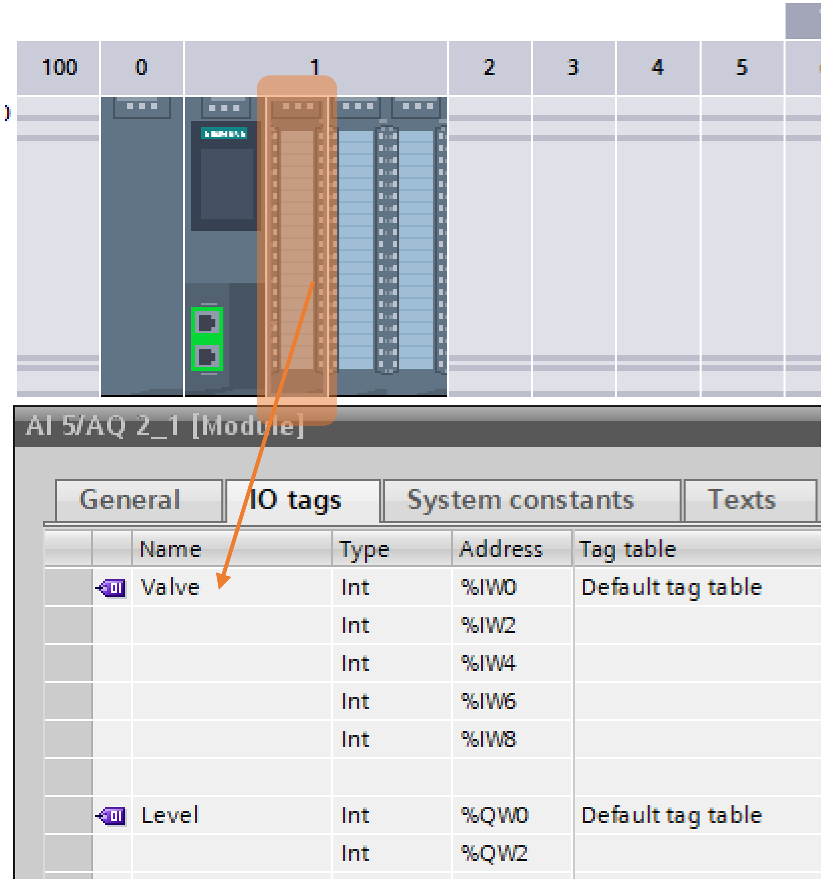
Model of analog outputs

# Problem configuration

- An automatic bottle filling machine
- Input the desired level, the machine will automatically fill up to its value.
- There is a sensor sensing the level of liquid which is increasing as the filling is going on.
- The valve stop as the desired level is reached
- There is the ‘reset’ switch to initiate a new filling

# Setting up the project

- Follow the example from two previous labs
  - Create new project named “Lab03”
  - Add PLC
  - Config the network
- Config analog ports
  - Analog I/O module is in the first slot as shown in the figure on the right
  - Inputs are the addresses labeled as “%IW_ ”
  - Outputs are the address labeled as “%QW _”
  - Use %IW0 and %QW0



# Design the parallel logic

- “Level” is measuring the liquid level in a bottle
- “Valve” is the target level to be filled, treated as an analog input
- In practical situation:
    - Level is input and must be read from analog input port
- In simulation:
    - Level is simulated via the ladder program
    - Use output port to simulate
- Operation
    - The PLC sent the read Valve value as the analog output to the automatic filling machine
    - The machine will stop as the liquid level reach the value sent by the PLC
    - A “Reset” digital input is pressed to reset the Level to 0

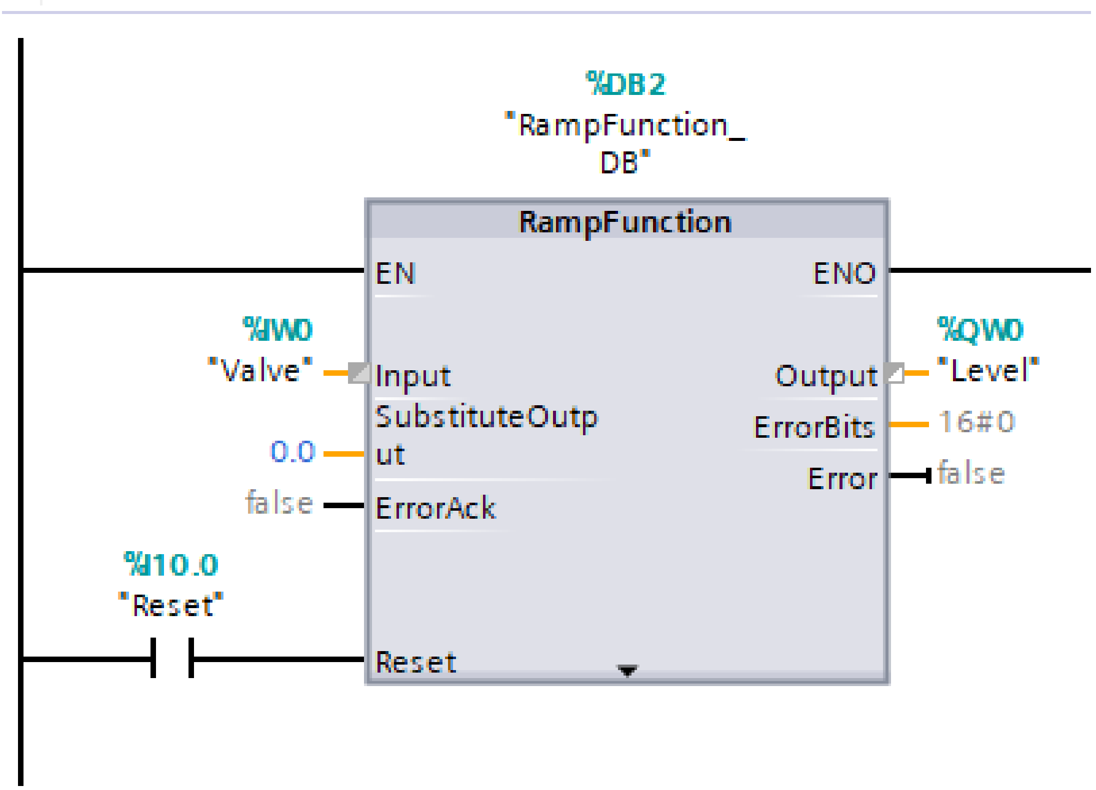
# PLC tags

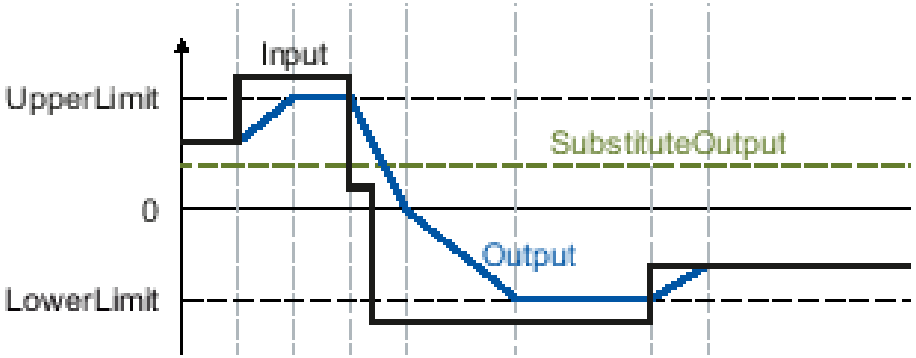
**Default tag table**

| | Name | Data type | Address | Retain | Acces... | Writa... | Visibl... |
|---|---|---|---|---|---|---|---|
| 1 | Level | Int | %QW0 | ☐ | ☑ | ☑ | ☑ |
| 2 | Valve | Int | %IW0 | ☐ | ☑ | ☑ | ☑ |
| 3 | Reset | Bool | %I10.0 | ☐ | ☑ | ☑ | ☑ |
| 4 | <Add new> | | | ☐ | ☑ | ☑ | ☑ |

# Filling machine simulation via “RampFunction” block

- One of the simplest block to emulate the response of this filling machine is called “RampFunction”.
- It will continuously increase or decrease its output to the level of its input as shown in the below timing diagram.

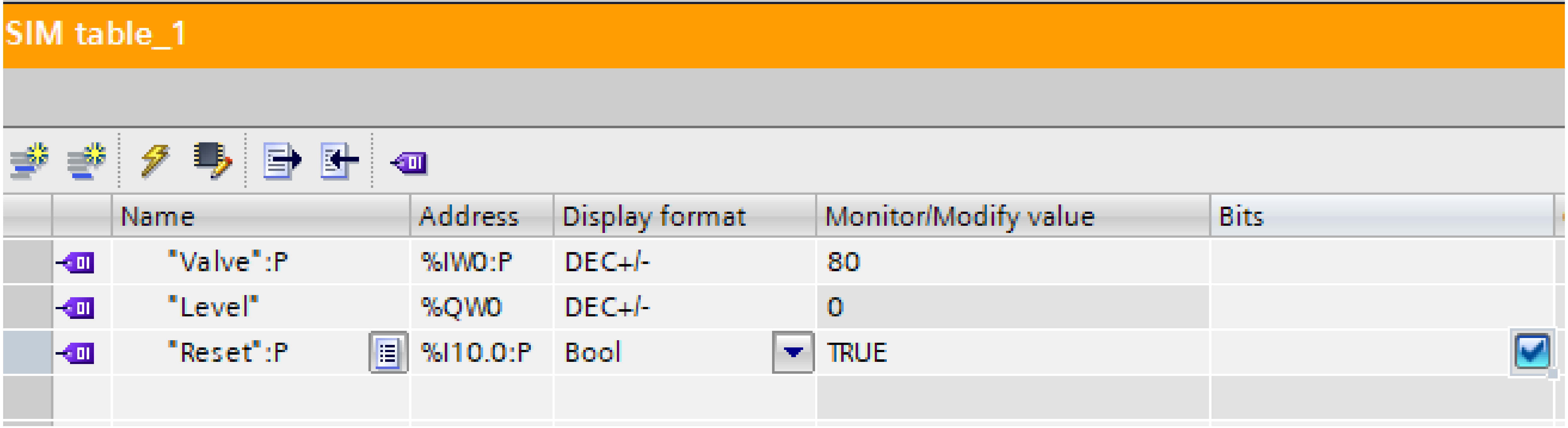



If the Reset is “true”, the output is set to “SubstituteOutput” which is default at “0”.

Simulation

- Set the digital input “Reset” to TRUE
- Set the analog input “Valve” to 80

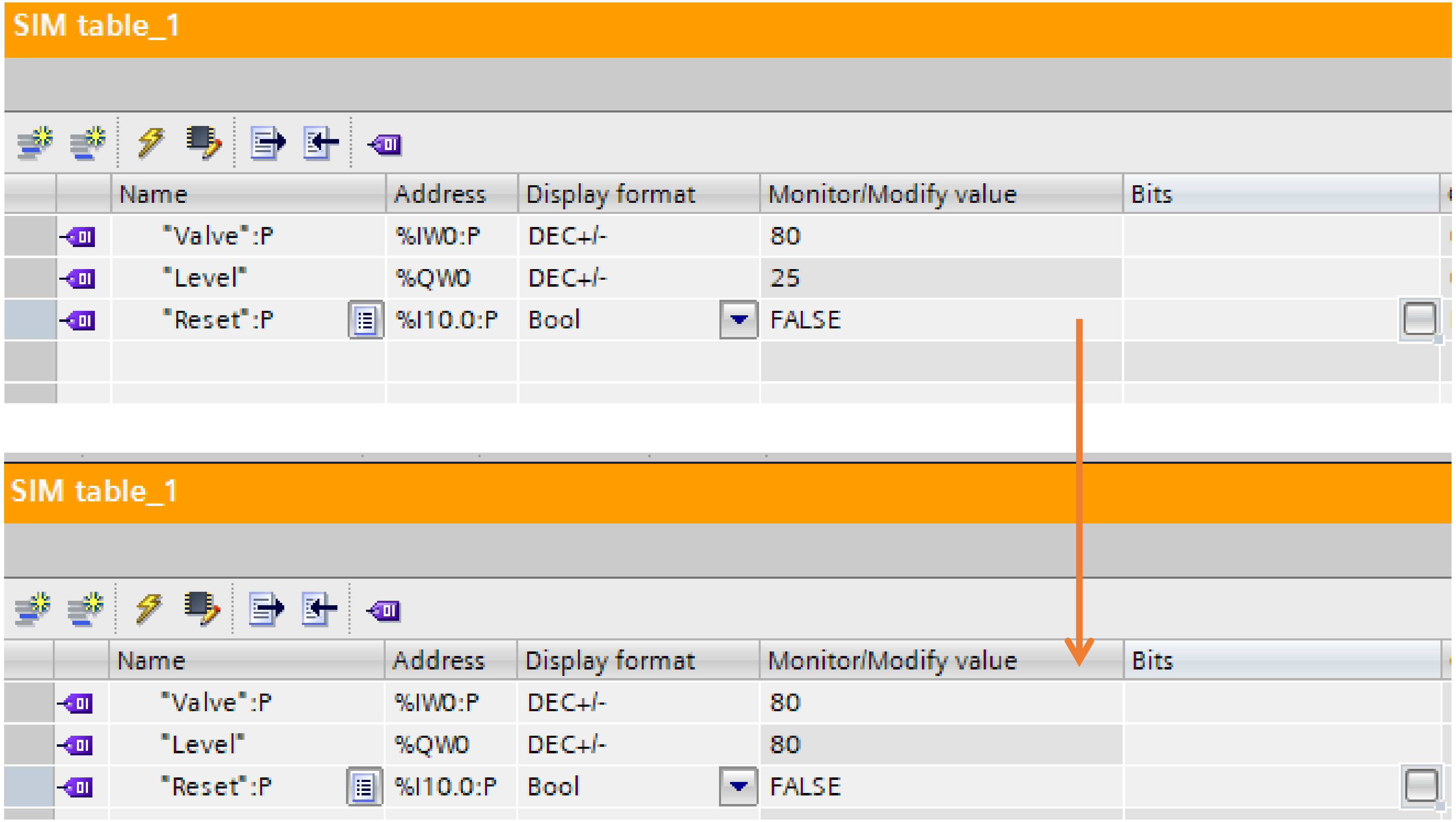


SIM table_1

| Name | Address | Display format | Monitor/Modify value | Bits |
|---|---|---|---|---|
| "Valve":P | %IW0:P | DEC+/- | 80 | |
| "Level" | %QW0 | DEC+/- | 0 | |
| "Reset":P | %I10.0:P | Bool | TRUE | |

# Simulation

- Switch the “Reset” to FALSE
- The analog output “Level” will gradually increasing until it reach 80



# Exercise

- Try to set the input “Valve” to 200 and observe the result
- Try to set the input “Valve” to -200 and observe the result
- Modify the example to be the system of 2 valves for filling 2 bottles simultaneously