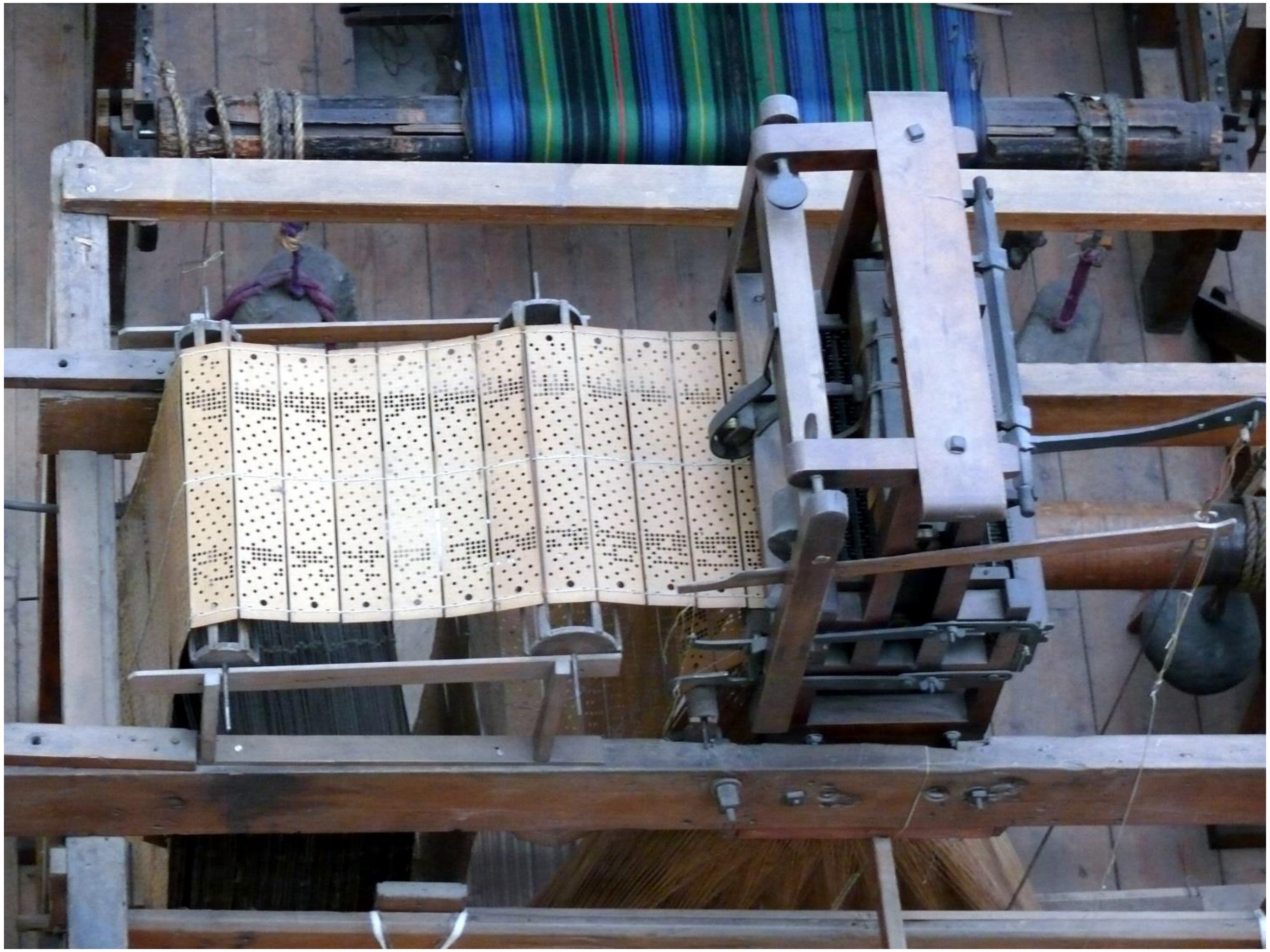
PLCs

Industrial Control

Gábor KOVÁCS
gkovacs@iit.bme.hu

BME FACULTY OF ELECTRICAL ENGINEERING AND INFORMATICS

Early years of automation



Jacquard's loom (1801) – mechanical control

Late 19th century

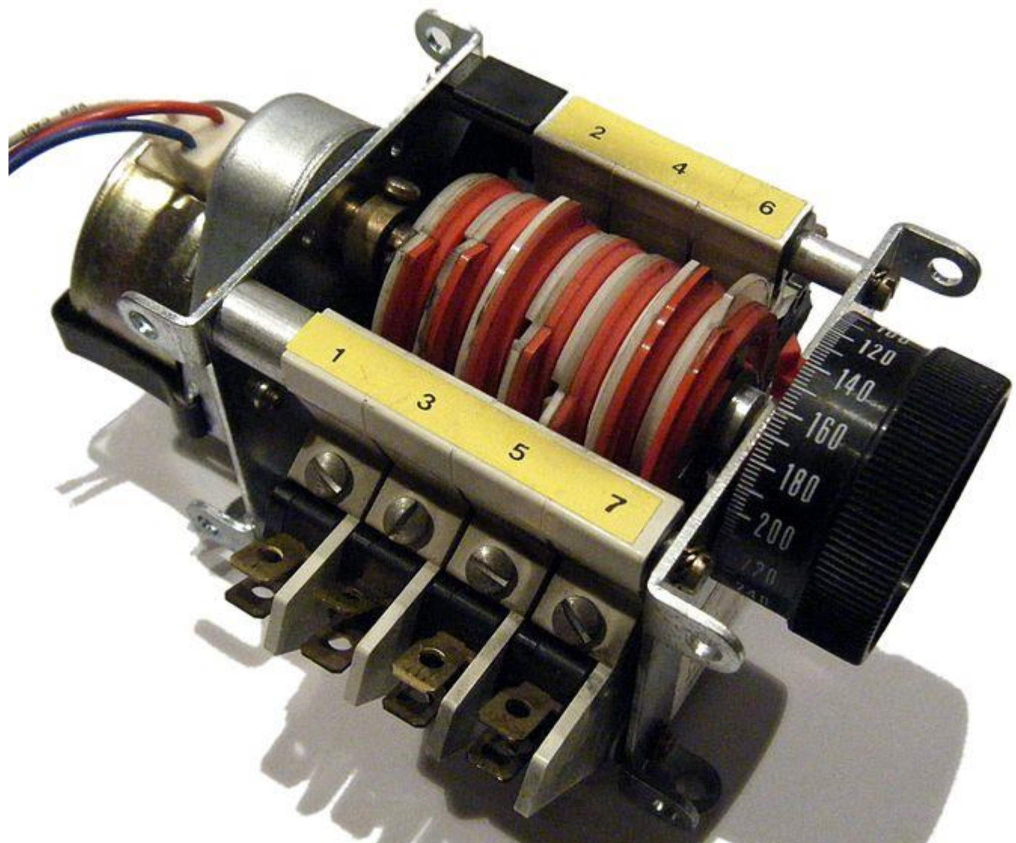
- Pneumatic control
  - bulky devices
  - need for continuous air supply
  - moving parts failing frequently

Early 20th century

- Electromechanical control
- Key: relay

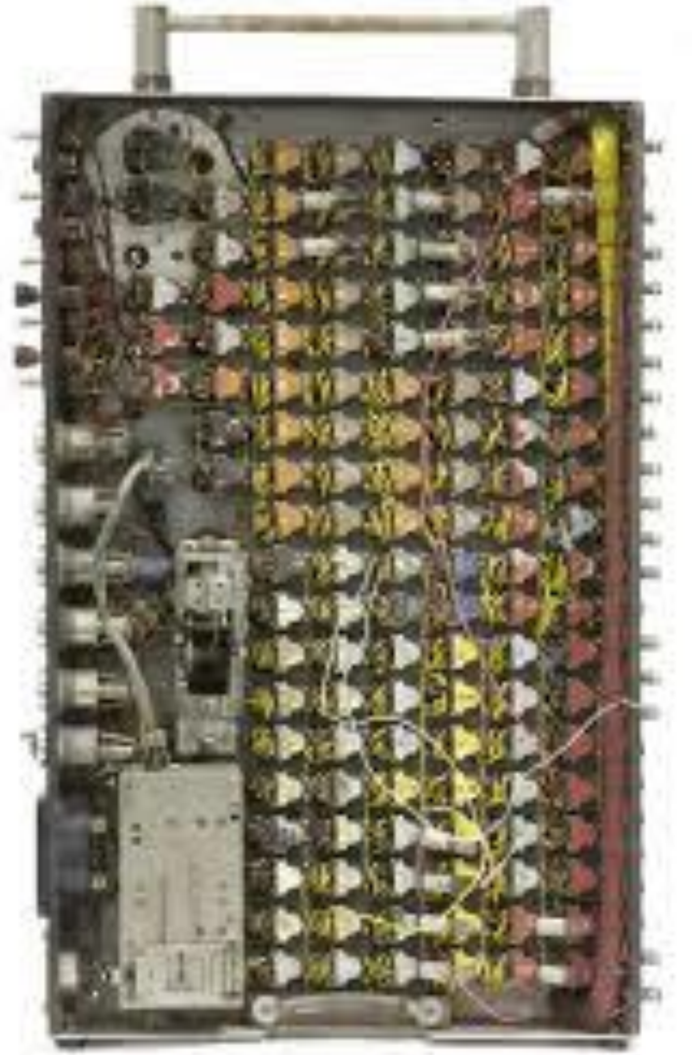
Relay control

- Boolean logic
- Camshaft timers and counters
- Drum controllers



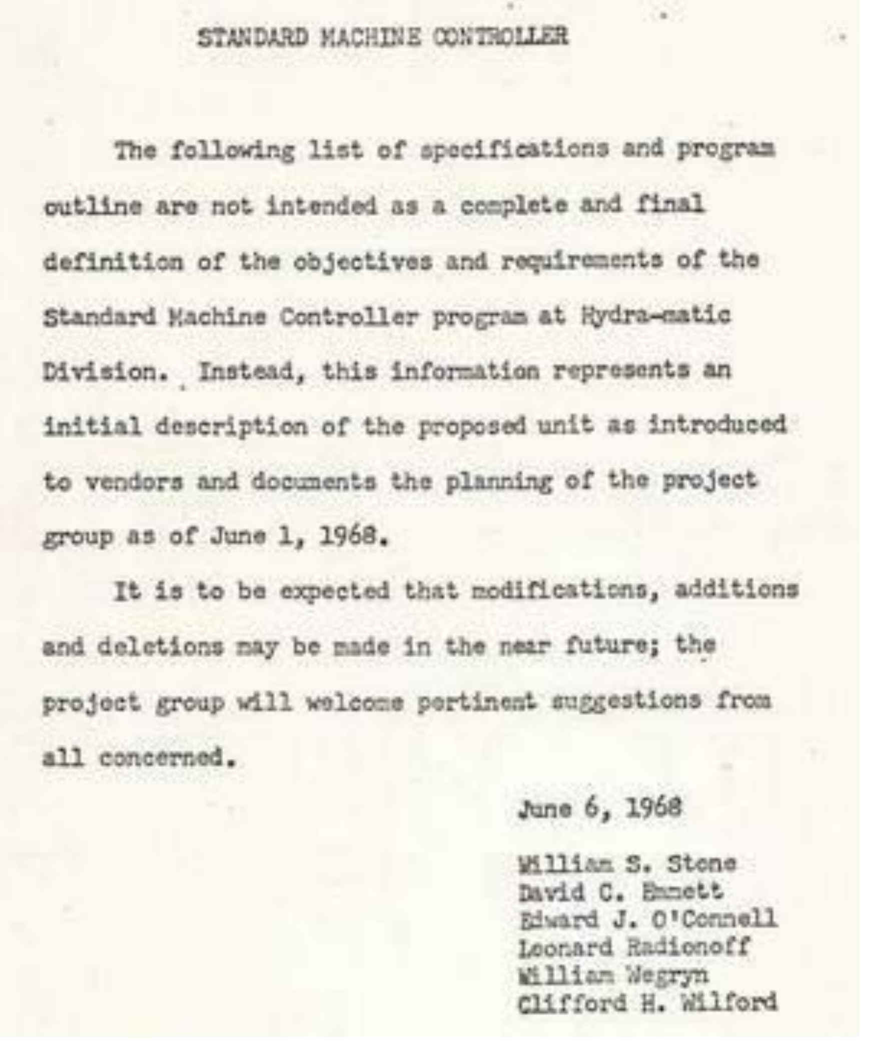
Relay control

- More compact (still bulky)
- Lower power consumption
- Less reliable
- Wired logic: complex circuits are hard to understand and repair



Birth of the PLC

- 1968 – General Motors proposal for a control device, which is
  - simple and reconfigurable
  - safe and reliable
  - has better cost/benefit ratio than electromechanical systems
  - able to operate in harsh industrial environment



STANDARD MACHINE CONTROLLER

The following list of specifications and program outline are not intended as a complete and final definition of the objectives and requirements of the Standard Machine Controller program at Hydra-matic Division. Instead, this information represents an initial description of the proposed unit as introduced to vendors and documents the planning of the project group as of June 1, 1968.

It is to be expected that modifications, additions and deletions may be made in the near future; the project group will welcome pertinent suggestions from all concerned.

June 6, 1968

William S. Stone
David C. Emmett
Edward J. O'Connell
Leonard Radionoff
William Wegryn
Clifford H. Wilford

Modicon 084

The proud father

Richard E. „Dick” Morley

(1932-2017)

„Engineering means taking something, changing it and making it more useful to our culture, our community, our country, whatever you like.”

Why is it called a PLC?

- **P**rogrammable
- **L**ogic
- **C**ontroller

PLCs are way more than logical controllers

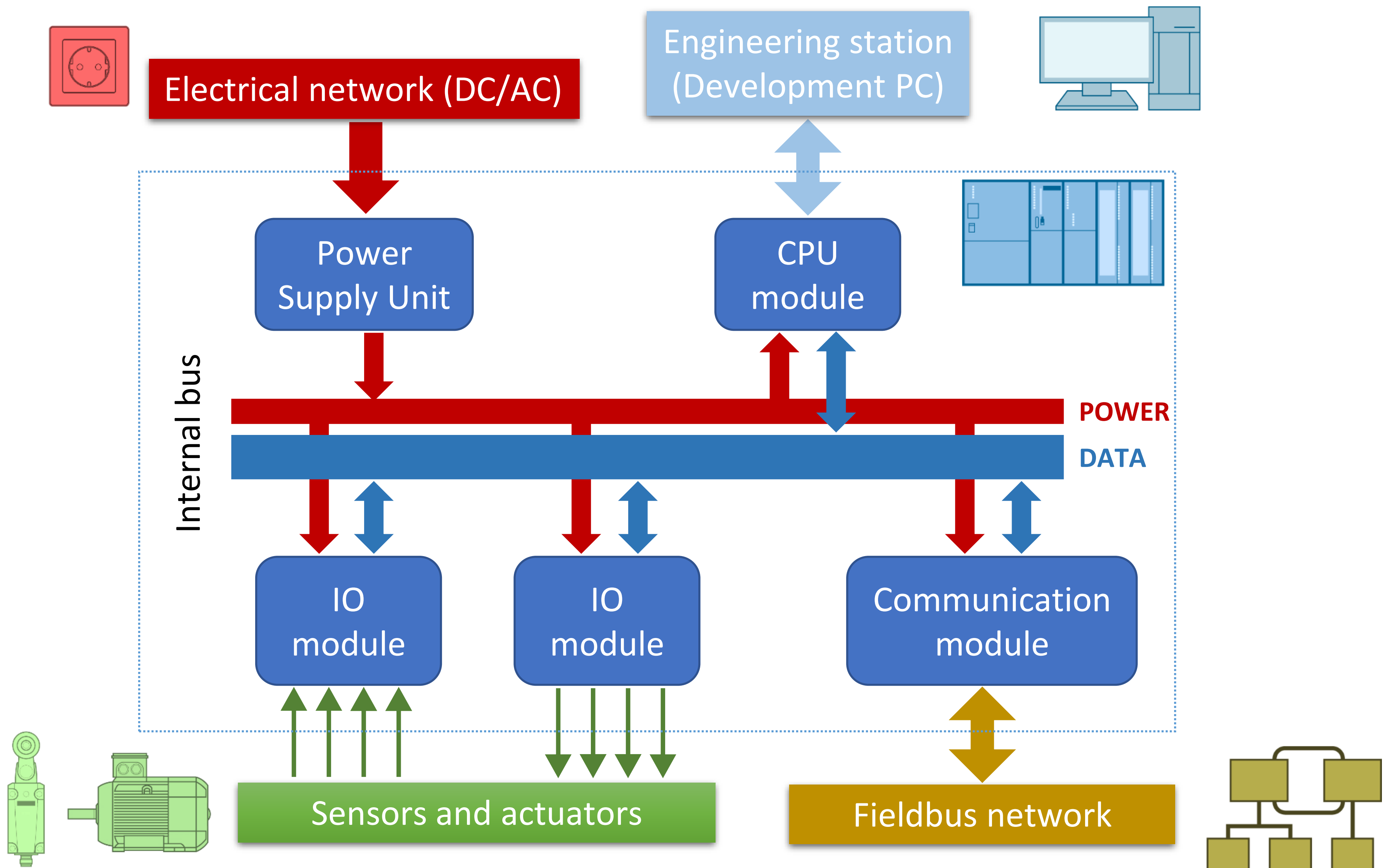
- Arithmetic operations
- Floating point numbers
- Timers and counters
- Control algorithms (e.g. PID)
- Multi-axis motion control
- Soft computing methods (fuzzy logic)
- …

Then what a PLC is?

- **The PLC is an industrial embedded computer**

- Industrial: robust and reliable
- Embedded: all necessary peripherials are included
- Computer: microprocessor-based

Architecture of a PLC



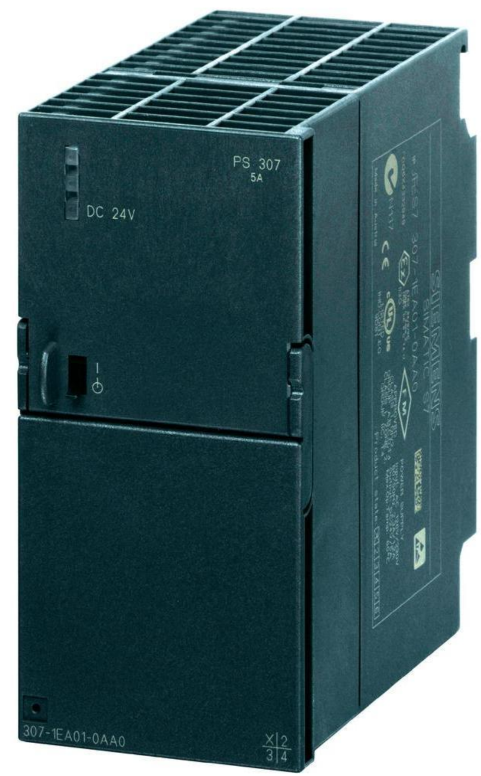
Backplane

(rack / chassis)

- Power and data connection between the modules
- Vendor-specific pinout and form factor
- Current models use internal connectors instead

Power Supply Unit (PSU)

- Reliable power supply
- Input: electrical network
  - 110/230 VAC
  - 24 VDC
- Output: voltage levels required by other modules
  - 24 VDC
  - 5 VDC
  - 3.3 VDC
  - …



CPU

- Processor
- Memory (with battery backup)
- Internal storage (flash memory)
- Programming interface
  - proprietary (rare)
  - open (USB, Ethernet)
- Programming interface often provides communication options
  - Ethernet
  - RS232/RS485

I/O modules

- I/O modules connect the PLC to sensors and actuators
- Digital I/O
- Analog I/O
- Specialty modules
  - fast counters, encoder inputs
  - thermocouple and RTD inputs
  - …

Communication modules

- Connects the PLC to a fieldbus network
- Physical layer
  - RS232/485
  - Ethernet
  - CAN
  - Fiber optics
  - …

PLC types


140
CPS 124 20
140
CPU 651 60 S
140
NOE 771 11
140
SDI 953 00 S
140
SDO 953 00 S
ESC
MOD
ENTER
Modicon
TSX Premium
Telemecanique
Modicon M340
DDI1602
24 V
Reset
Ethernet
CANopen
Telemecanique

NanoPLCs

- Known as:
  - NanoPLC
  - PicoPLC
  - Programmable Relay Controller
  - Smart Relay
- Up to 15 IO ports
- Compact form factor
- Some models might be programmed without external devices
- Suitable for the most simple tasks only

Micro PLCs

- Compact form factor: CPU + IOs in the same housing
- Mostly equipped with some kind of communication interface
- 12-32 IOs
- Up to 128 IOs by extension modules

Mid-range PLCs

- Modular devices
- Up to 512 IOs, wide variety of modules
- Flexible and versatile devices

Large-scale PLCs

- 512+ IO (up to 4096)
- Modular devices in rack configuration
- For most complex and demanding applications
- Have been replaced by distributed controller systems

Special PLCs

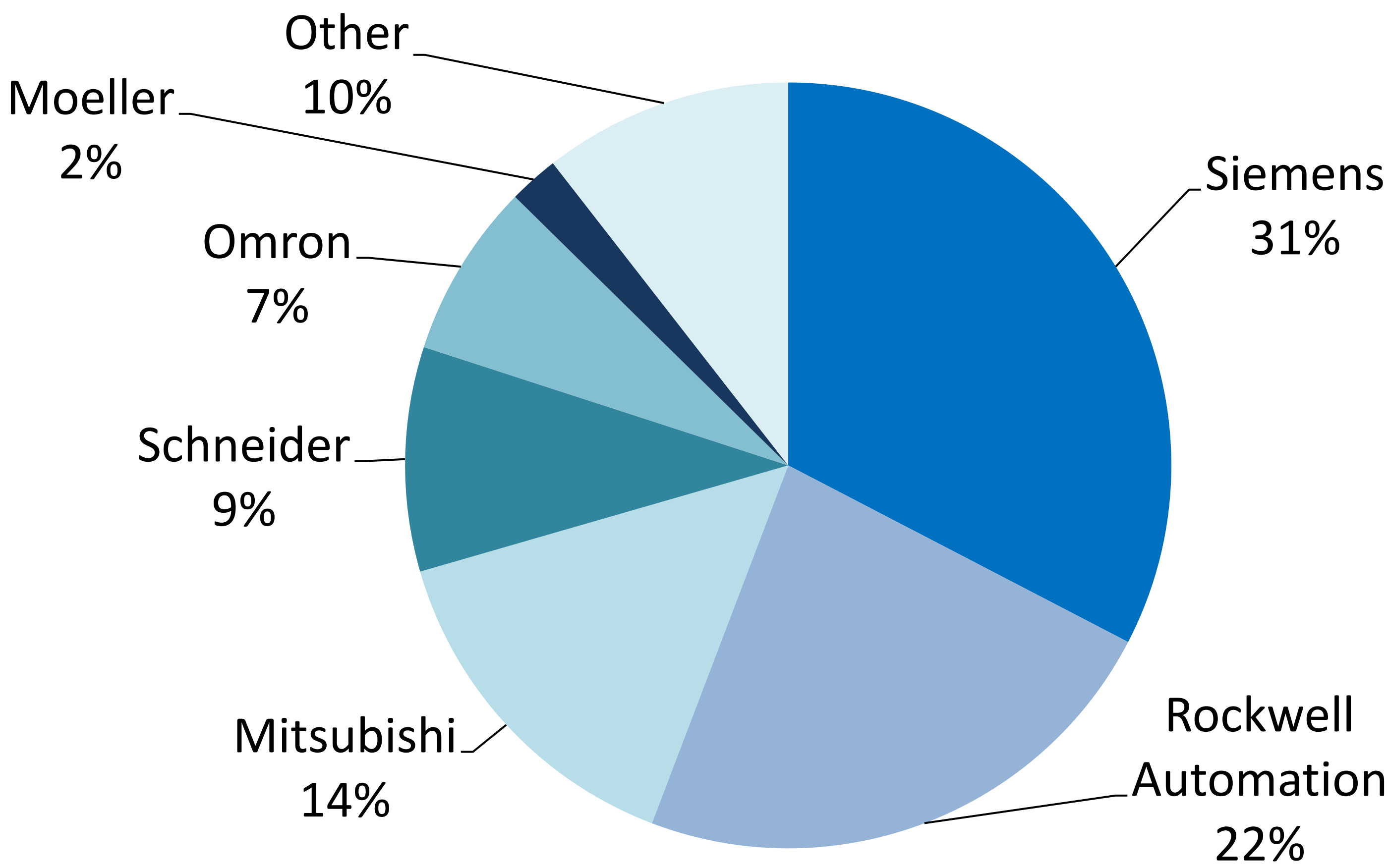
- Ex
  - explosion proof models
  - requires special hardware solutions

- Fail-Safe (Safety)
  - special, redundant hardware architecture
  - supported by special software design techniques

Soft PLCs

- PLC application runs besides/over an ordinary operating system (Linux / Windows)
- Allows execution of complex non-control applications (e.g. image processing)
- General-purpose devices can be used
  - industrial PC (x86)
  - Raspberry Pi
  - …

Market share of PLC manufacturers



Source: Global Industrial Automation, Credit Suisse (2017)

- Logo! nanoPLC
- Simatic S7
  - S7-1500
  - S7-1200
  - S7-300
  - S7-400

Rockwell Automation

Allen-Bradley

by ROCKWELL AUTOMATION

- Owner of the Allen-Bradley brand
- Micro800 series
- CompactLogix
- ControlLogix

MITSUBISHI
ELECTRIC

- Alpha nanoPLC
- FX series
- L series
- System Q

Schneider Electric

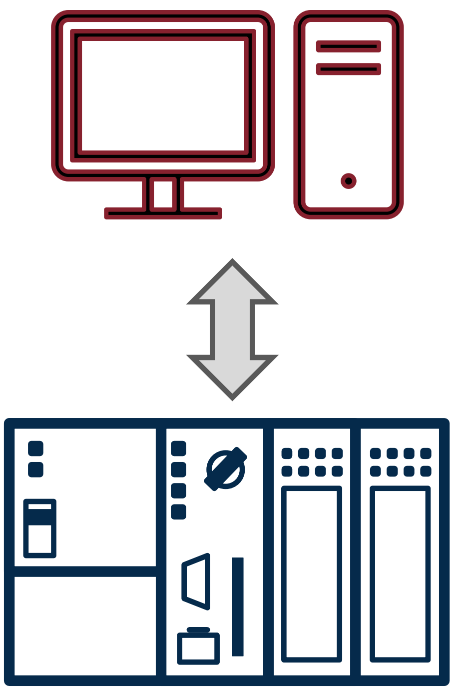
- Owner of the brands Modicon and Telemecanique
- Zelio nanoPLC
- Twido micro PLC
- Modicon series

OMRON
Industrial Automation

- CP series
- CJ series
- CS series

- EasyRelay series
- EasyControl series
- XC series

PLC software



Development environment
(*Programming And Debugging Tool – PADT*)

- PLC configuration
- Application development
- Simulation
- Application and data transfer
- PLC monitoring
- Documentation

User program

Operating system / Firmware / Runtime

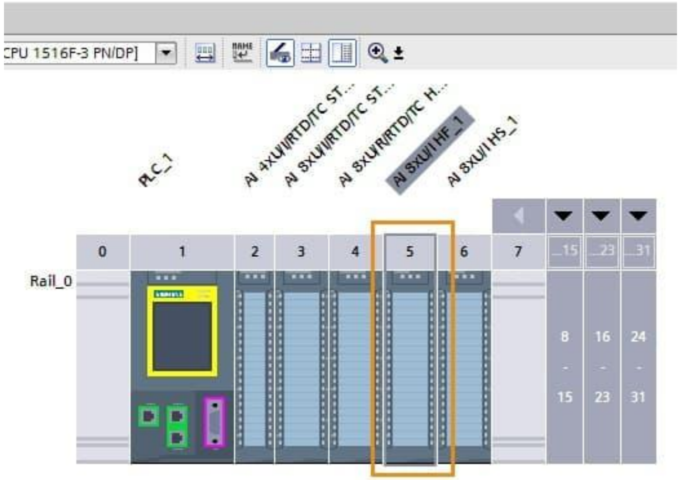
Development environment

(Programming And Debugging Tool – PADT)

- Development environment running on PC
- Indispensable for PLC-based automation
- Necessary for configuration, application development, program transfer, debugging

PLC configuration

- Possible features of the application depend on
  - PLC/CPU model
  - number and type of IOs available
  - communication options
  - PLC firmware/OS version



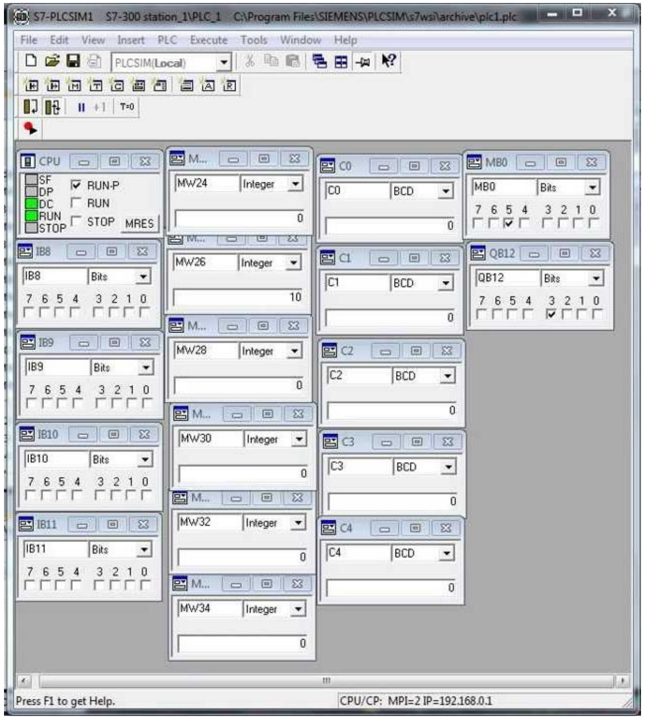
- Development environment allows
  - specification of PLC configuration (modules used and their configuration)
  - firmware/OS update
  - configuration of the whole control system (multiple PLCs, HMIs, drives) in a single project – only supported by complex development environments

Application development

- Goal: help the user in efficient development of the control application
- Support of multiple programming languages
- Compile and build
- User-friendly features
  - easy-to-use environment
  - autocomplete / autodeclare

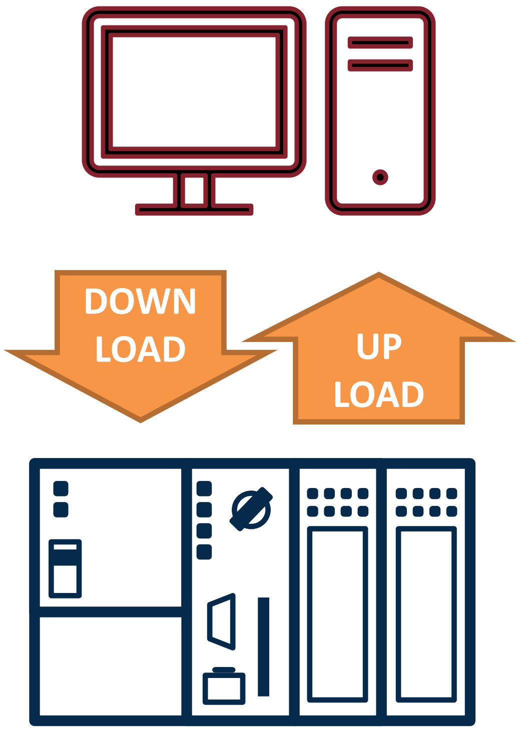
Simulation

- PLC code is executed by an emulator running on the PC
- Memory can be monitored
- Debugging tools (e.g. breakpoints)
- Virtual outputs can be observed, virtual inputs can be set
- Safe and flexible way to test and validate the application



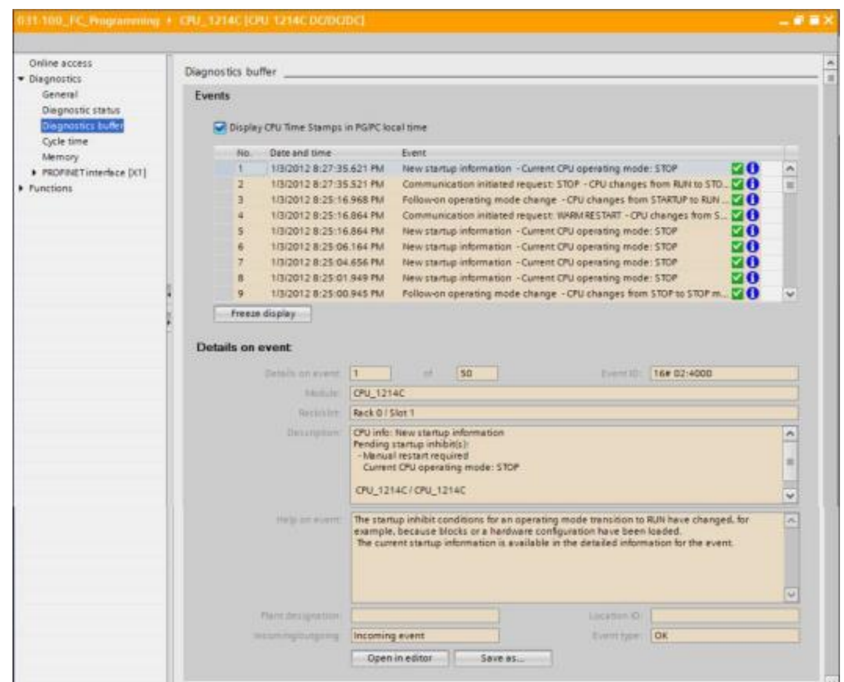
Application and data transfer

- Download
  - PC → PLC
  - application and data
- Upload
  - PLC → PC
  - application and data
- Download and upload might be possible without stopping the PLC



PLC monitoring

- Online monitoring of the memory
- Monitoring of program execution
- Manual override of outputs *(force)*
- Starting, stopping, resetting the PLC
- Error diagnostics



Documentation

- Documentation has paramount importance in industrial environment
- PLCs can store source code and comments beside the application
- Development environments support automated documentation
- Documentation includes
  - configuration settings
  - memory configuration (variables and data types)
  - program code
  - communication settings

Development environments

- Vendors provide specific development environments for their PLC families
- State-of-the-art tools are complex, integrated environments which allow configuration of other control devices beside the PLCs (HMIs, drives etc.)

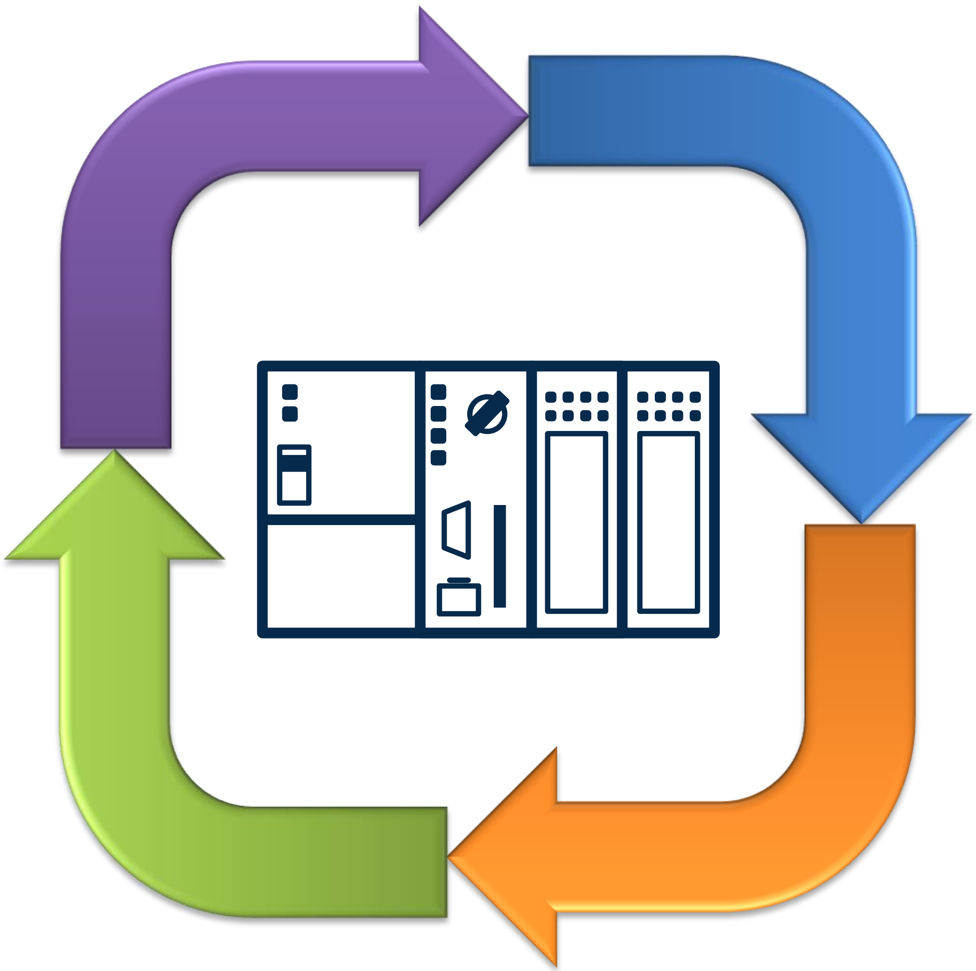
- Siemens: TIA portal (Step 7)
- Rockwell: Studio 5000, Connected Components Workbench
- Schneider: EcoStruxure, TwidoSuite
- Mitsubishi: iQ Works suite

CODESYS

- Vendor-independent development environment
- Base of several vendor-specific environments (Schneider, Beckhoff, Bosch-Rexroth etc.)

- Most standard-compliant environment
- Supports every standard programming language
- Simulator is included
- Integrated HMI-editor and visualization system
- Programming tool (Development System) is available free of charge
- Fully functional demo licenses for soft PLCs

PLC operating model



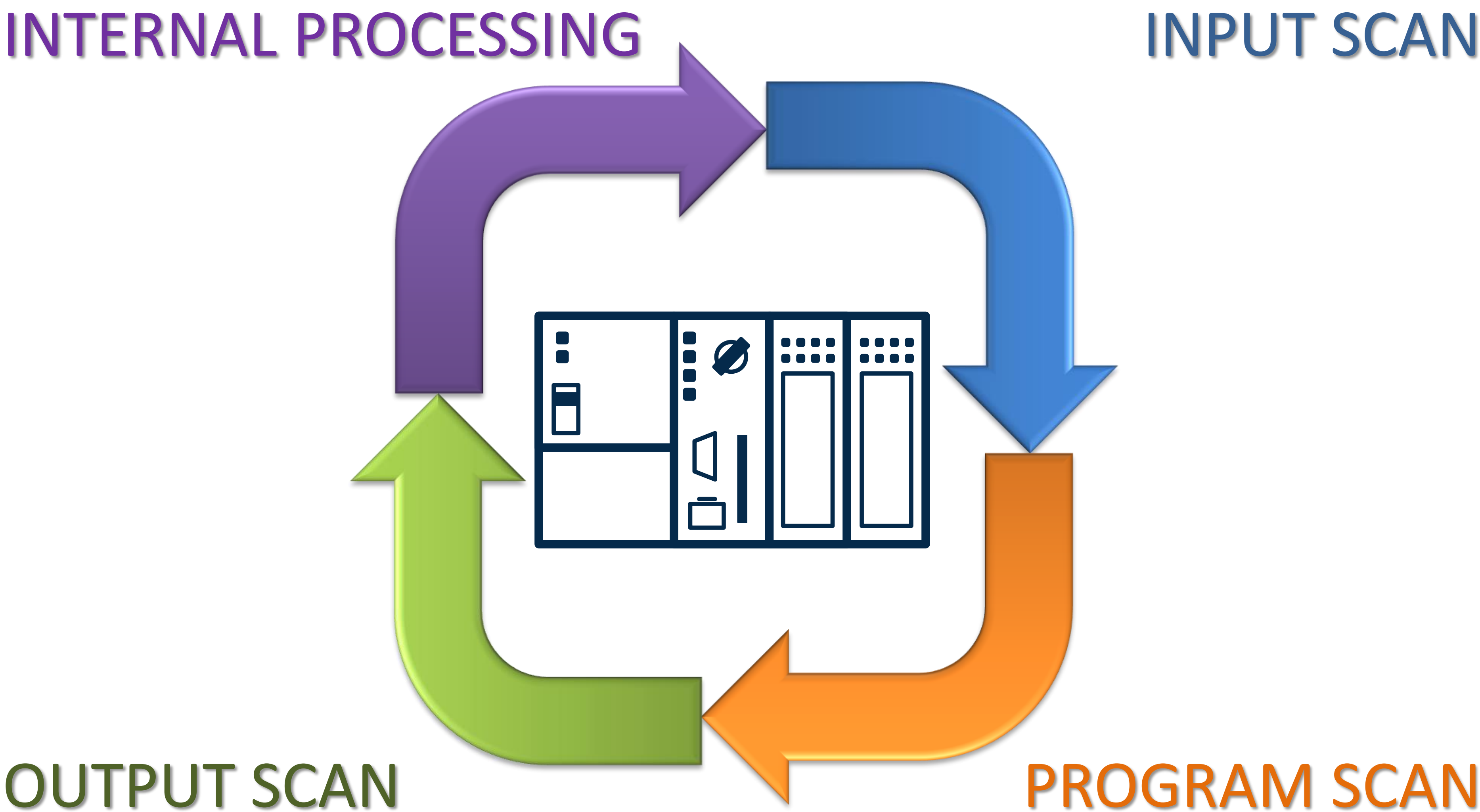
Operation of a desktop computer

- Complex operating system
- Multitasking
- Non-deterministic operation

Operation of a PLC

- Roles of the PLC
  - observe the operation of the technology by sensors connected to its inputs
  - evaluate the control algorithm
  - change the operation of the technology by actuators connected to its outputs
- Requirements
  - reliability
  - deterministic operation

PLC cycle



PLC memory model

Program memory

USER PROGRAM

Data memory

USER MEMORY

- Memory where the user program can store its data
- Might be missing in case of nanoPLC-s

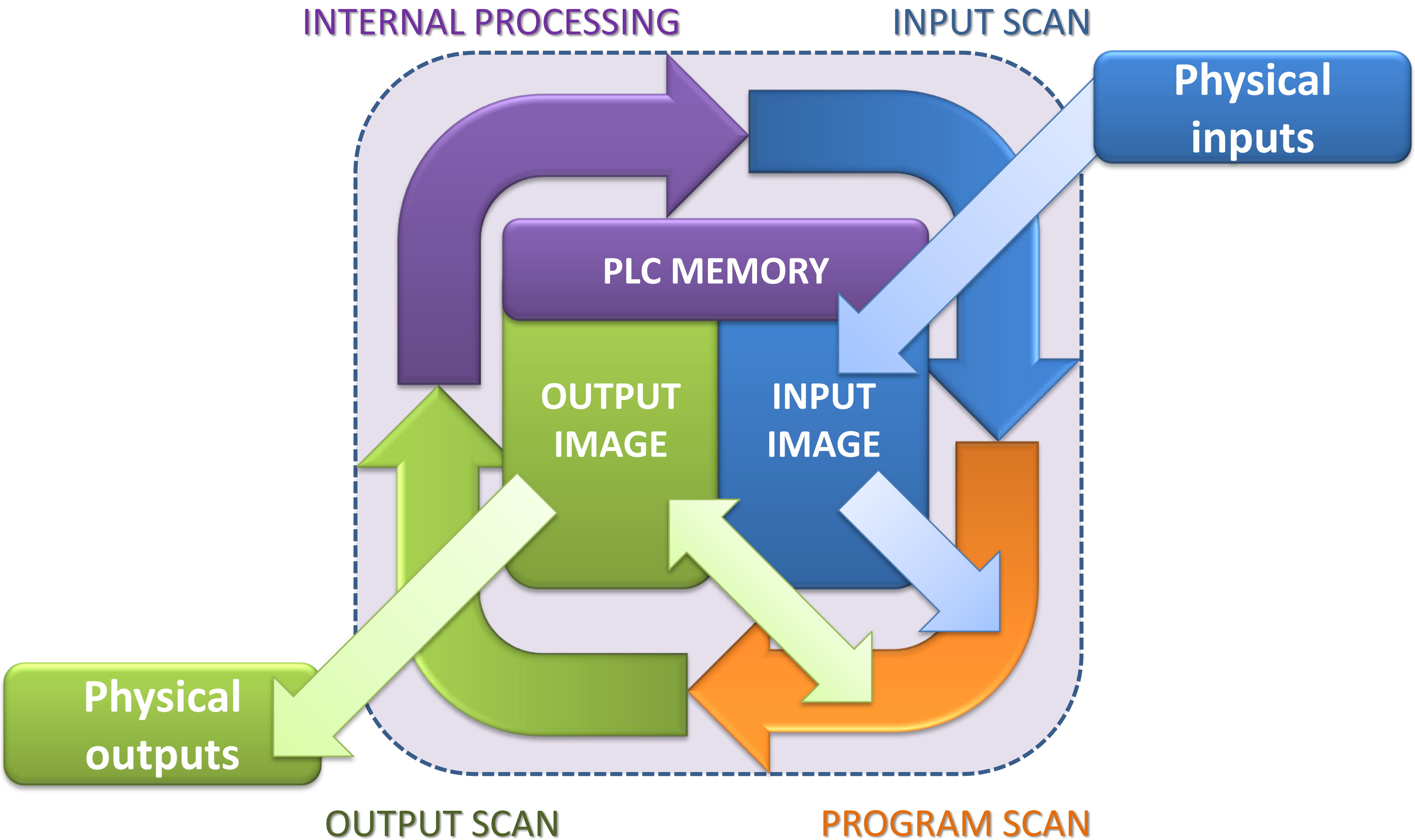
SYSTEM MEMORY

- Configuration data
- Status information
- Timing information
- General purpose registers
- Stacks

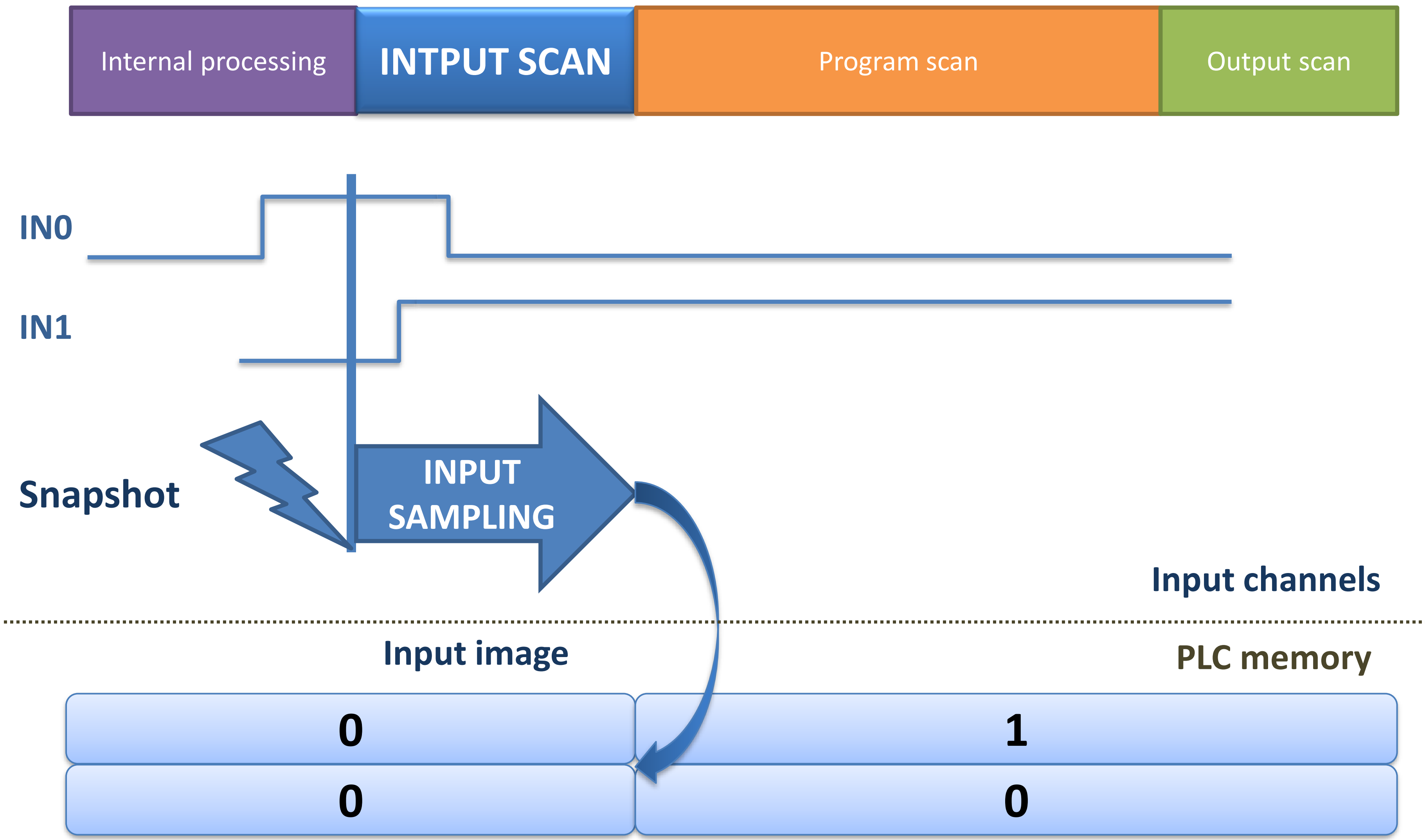
INPUT IMAGE

OUTPUT IMAGE

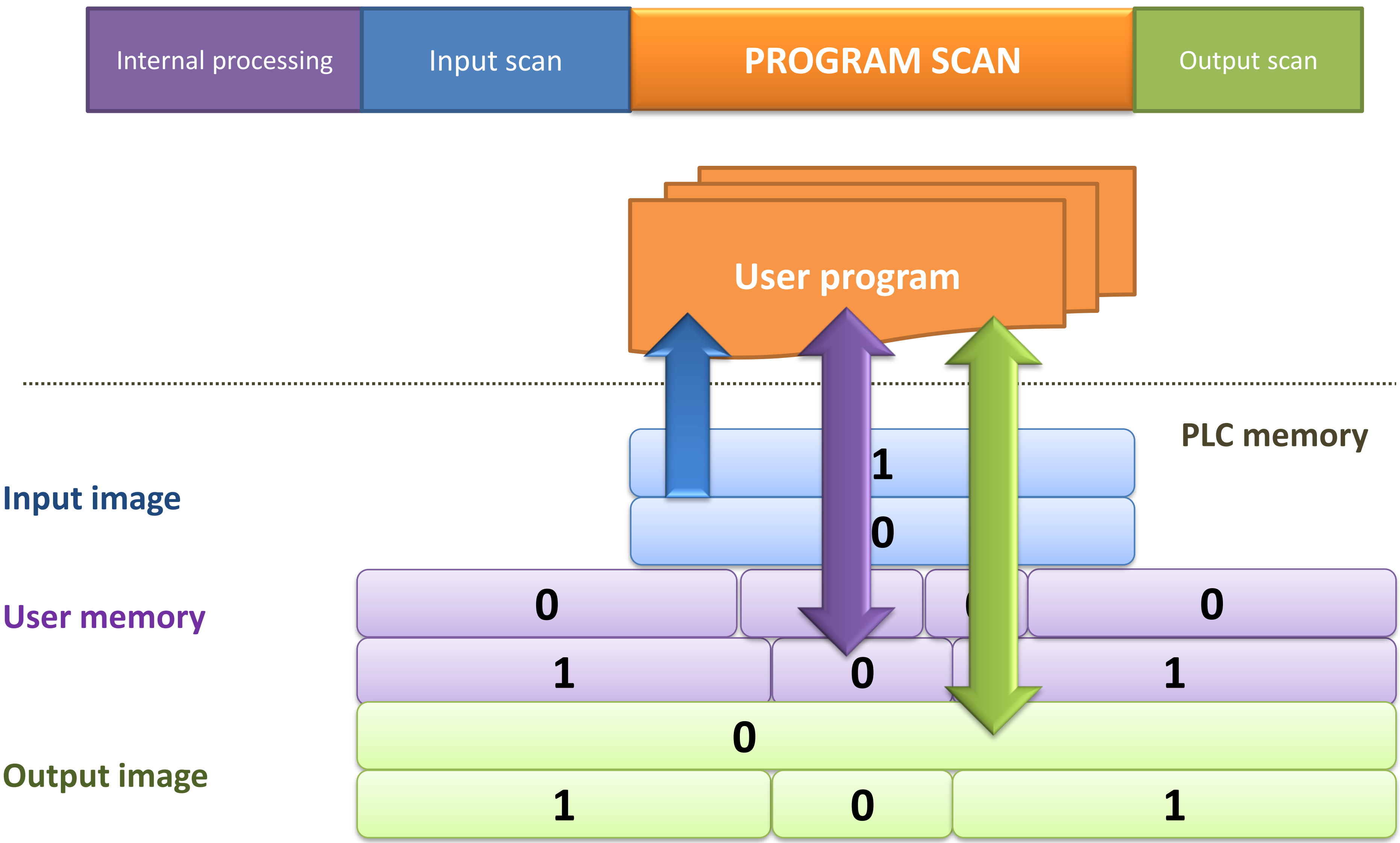
Input and output image



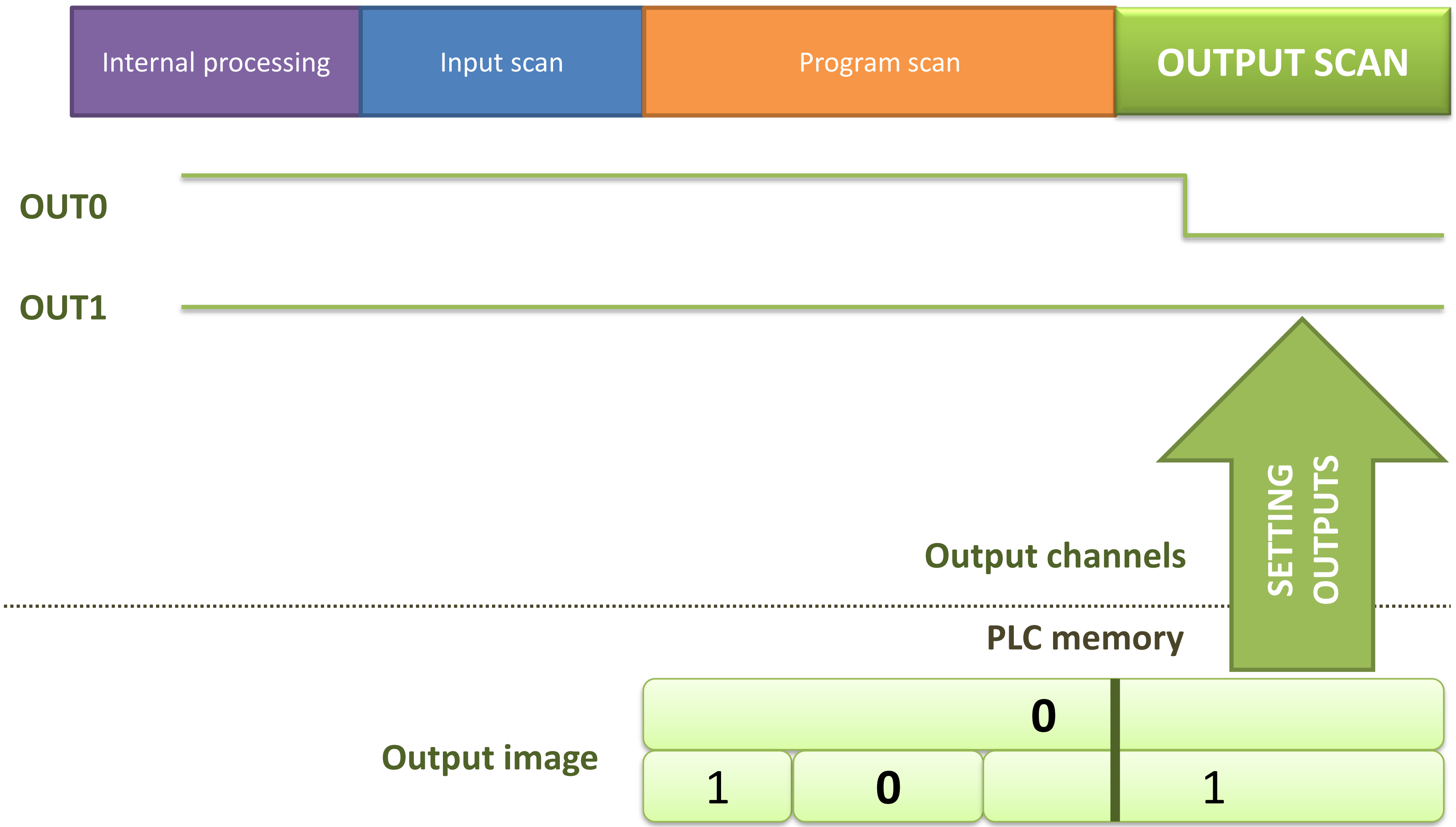
Input scan


Internal processing
INTPUT SCAN
Program scan
Output scan
IN0
IN1
Snapshot
INPUT
SAMPLING
Input channels
Input image
PLC memory
0
1
0
0

Program scan



Output scan


Internal processing
Input scan
Program scan
OUTPUT SCAN
OUT0
OUT1
SETTING OUTPUTS
Output channels
PLC memory
Output image
0
1
0
1

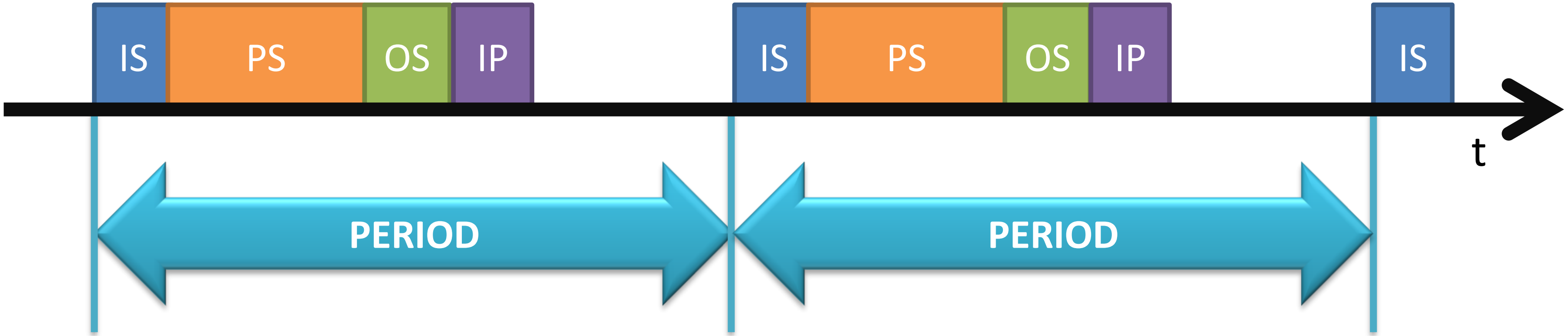
Internal processing

- PLC executes functions of the operating system
  - cycle control
  - self test
  - communication
  - timing

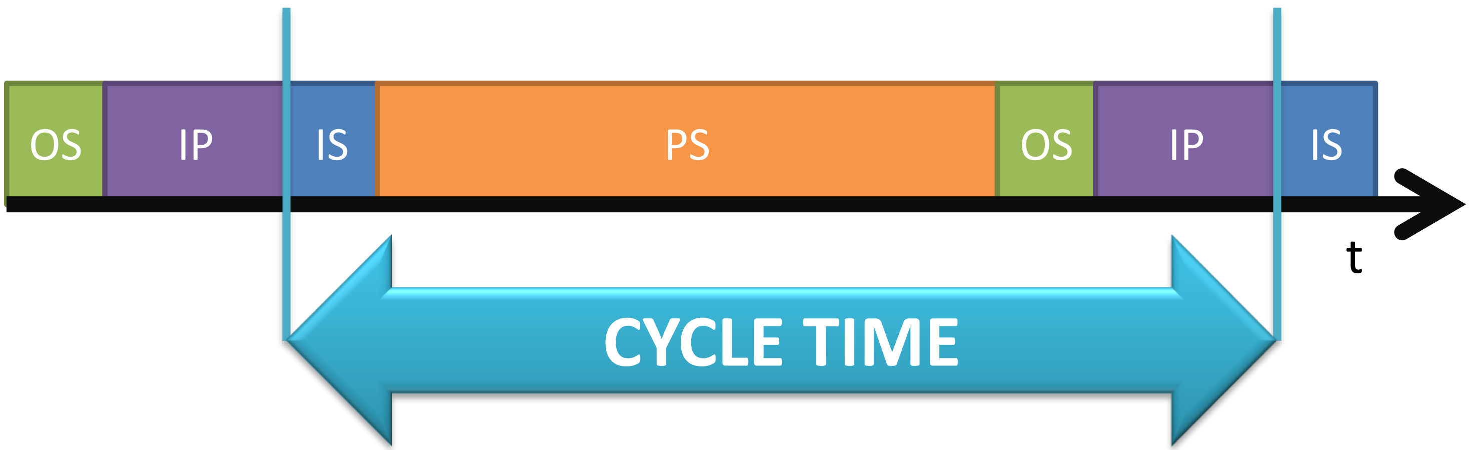
Cyclic vs periodic execution

Cyclic execution

Periodic execution



Cycle time



Cycle time: time of a whole PLC cycle

Components of the cycle time

- **Input / output scan**, duration depends on
  - number of inputs/outputs
  - type of inputs/outputs (digital/analog)
- **Program scan**, duration depends on
  - PLC/CPU model
  - number and type of instructions
    - counter increment: $0.05 - 1.2\mu s$
    - SQRT: $0.5 - 8.1\mu s$
    - timer operations: $3 - 11\mu s$
- **Internal processing**, duration depends on
  - PLC / CPU model
  - Number and complexity of OS features used (e.g. communication)

Cycle time – illustrative example

- Configuration
  - S7-314 CPU (Siemens S7-300 series)
  - 2 × SM321 32DC 24V digital input module
  - 2 × SM322 32DC 24V digital output module

- User program
  - 1.5 ms execution time

- No communication

Cycle time – illustrative example

Input scan

• General overhead		147 µs
• Read of input bytes	+	8×13.6 µs
	=	**0.26 ms**

Program scan

• Execution time of instructions		1.5ms
• CPU multiplier for S7-314	×	1.15
	=	**1.8 ms**

Output scan

• General overhead		147 µs
• Write of output bytes	+	8×13.6 µs
	=	**0.26 ms**

Internal processing

• Cycle control		1 ms
• Timer control	+	30×8 µs
	=	**1.24 ms**

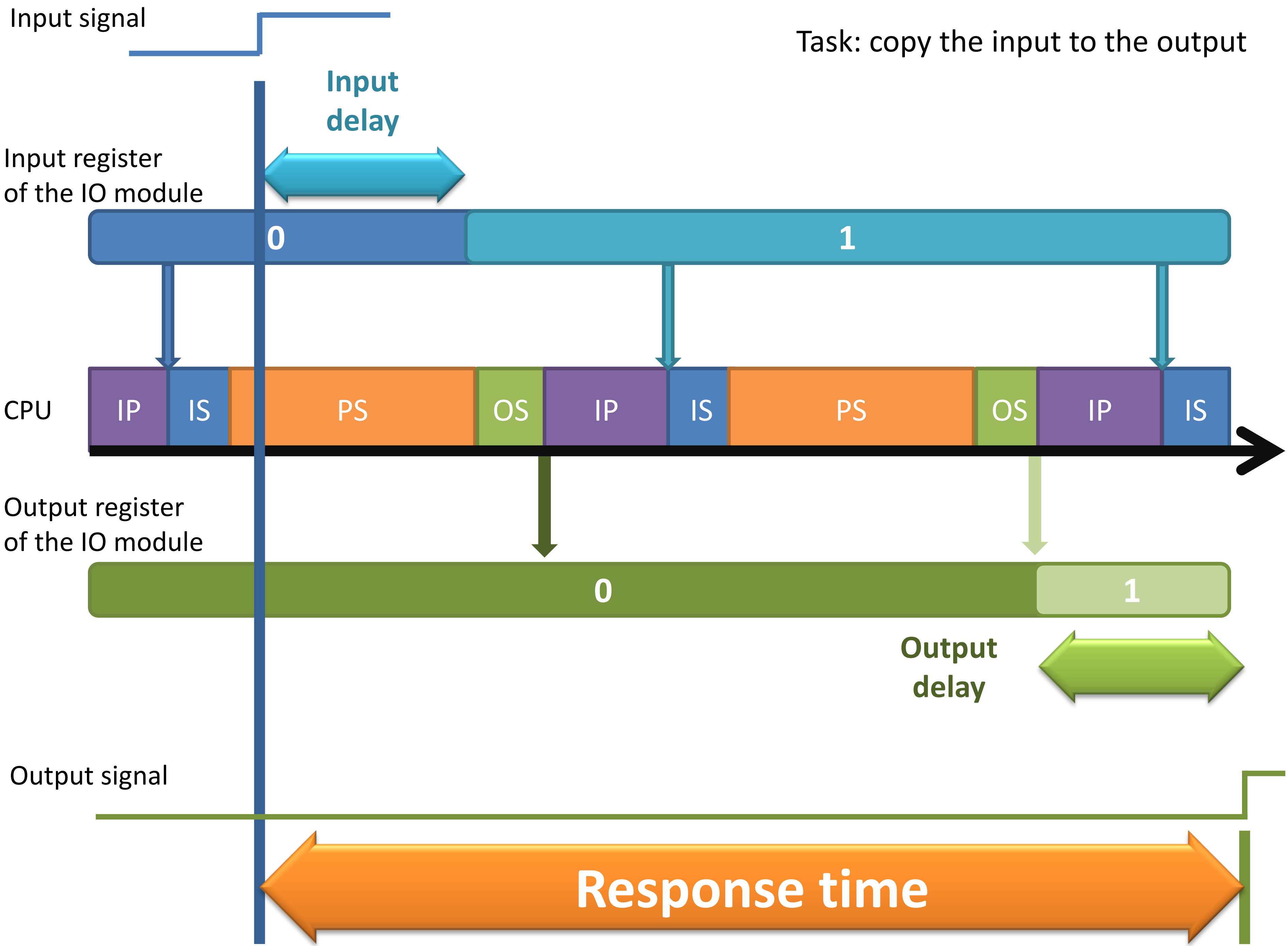
	0.26 ms
+	**1.8 ms**
+	**0.26 ms**
+	**1.24 ms**
=	**3.56 ms**

Calculation of the cycle time

- Simple PLCs
  - number of instructions and IOs are limited
  - it means an upper limit for execution time
- Complex PLCs
  - cycle time is calculated automatically by the development environment
  - possibilities for tuning the execution time of internal processing

Response time

- Time required to reply an external stimulus
- „If the value of an input is changed, how much time does it take for the controller to change its outputs appropriately?”



Input signal
Task: copy the input to the output
Input delay
Input register of the IO module
0
1
CPU
IP
IS
PS
OS
IP
IS
PS
OS
IP
IS
Output register of the IO module
0
1
Output delay
Output signal
Response time

Response time

- Worst case:

	Input delay
+	$\approx 2 \times$ Cycle time
+	Output delay
=	Response time

- Since input and output delays are significantly lower than the cycle time, response time can be approximated by the double of the cycle time

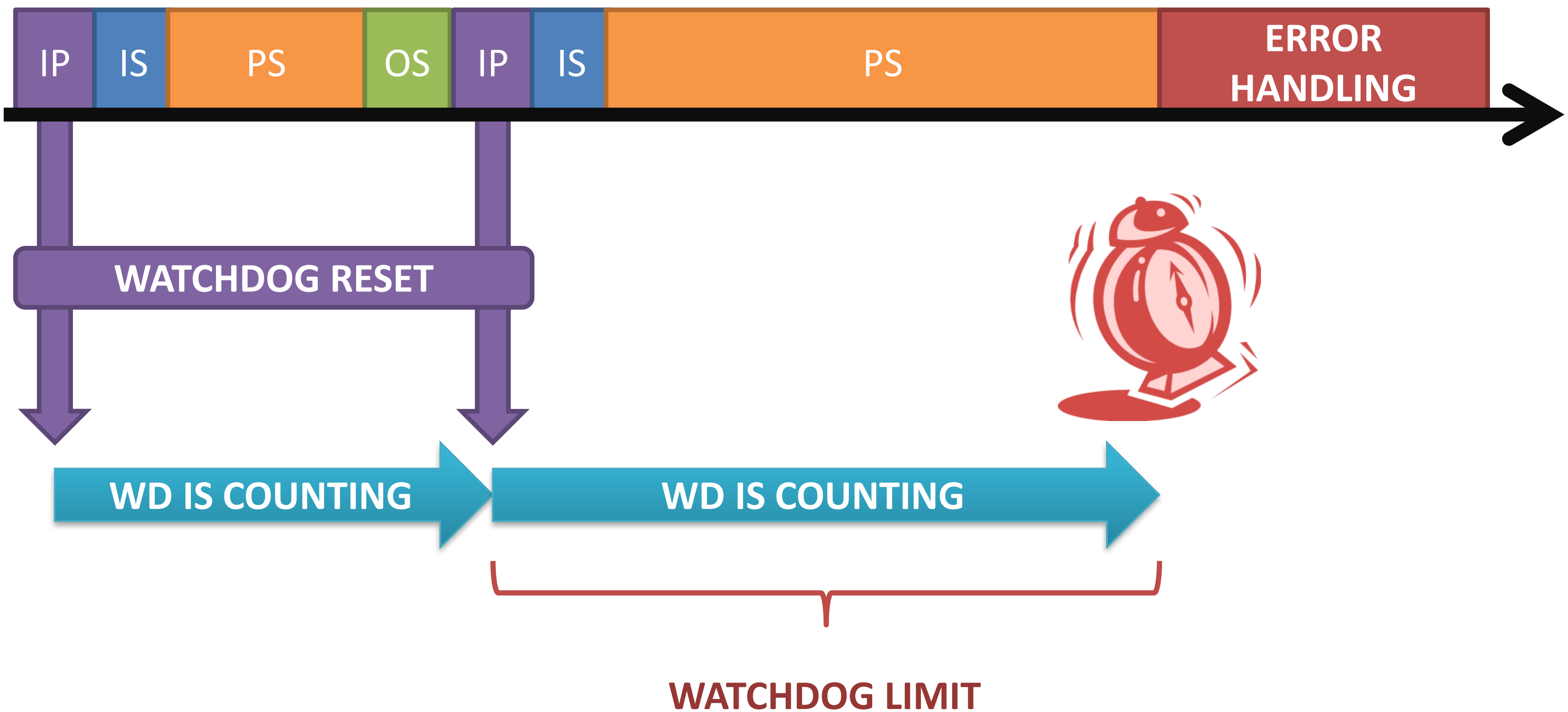
Deterministic operation

- Cycle time is known (at least an upper bound)
- Input and output delays are also known (at least an upper bound)
- **We have an upper bound for the response time!**

Real-time systems

- Systems which are able to react a stimulus in a well-defined finite time are called **real-time** systems.
- **PLCs are real-time systems.**

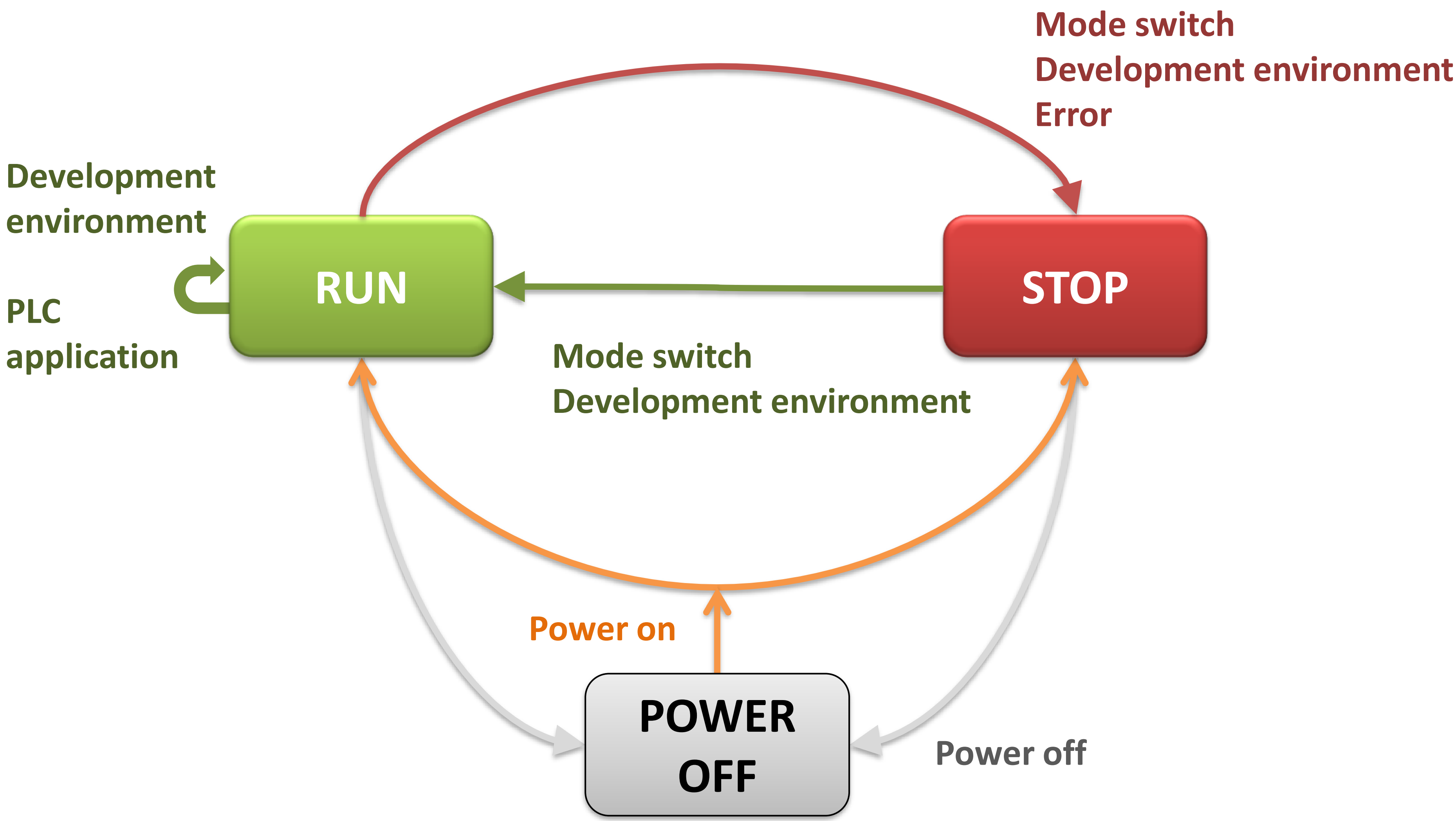
Watchdog



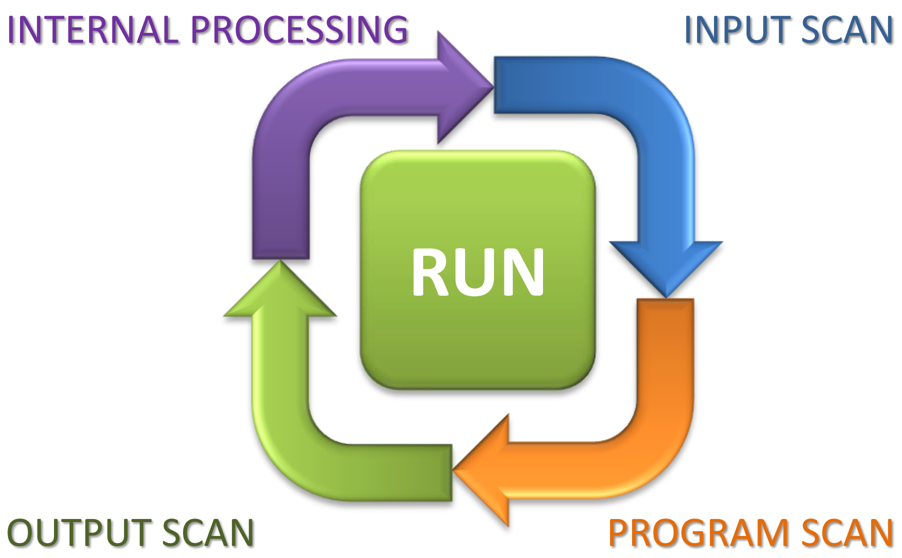
Error handling might be:

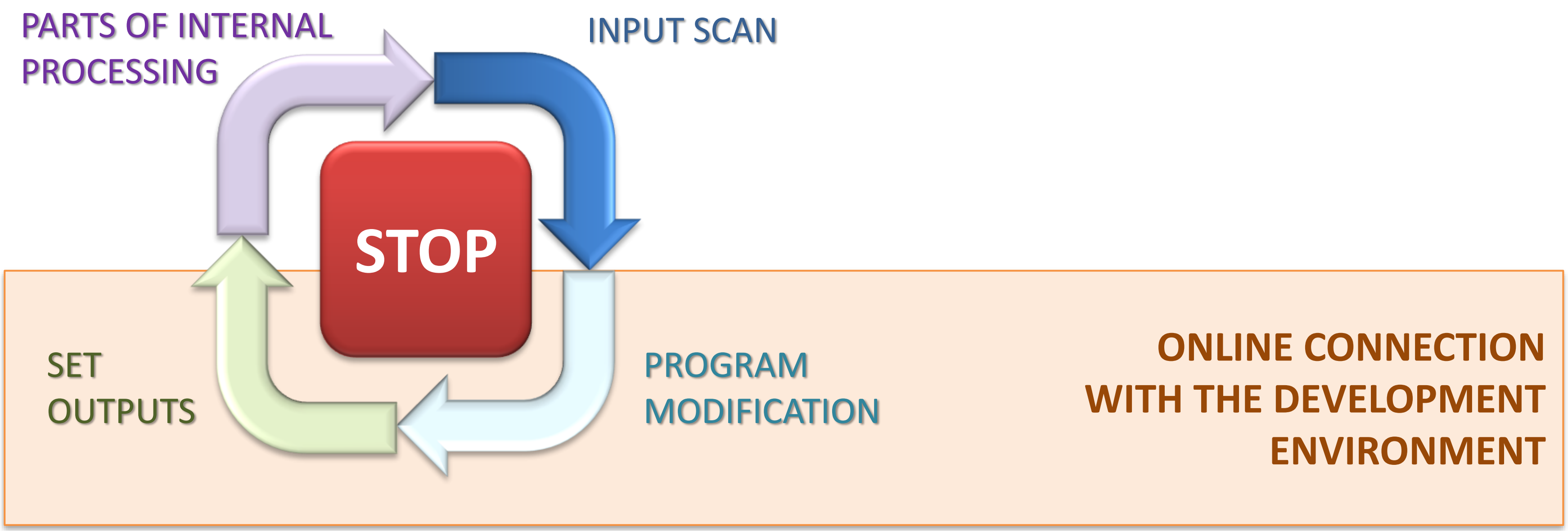
- shutdown, set outputs to safe states, raise an alarm
- start of an error handling routine

Operational modes of a PLC



Run and Stop modes





PLC startup

- **Cold start**
  - variables are set to their initial value
  - user program is started from the beginning
- **Warm start**
  - state of the PLC is restored to the state before stop
  - application data is restored
  - user program is started from the beginning (parts of the cycle after stop are omitted)