Using the Process Image

CoDeSys

Smart Software Solutions

We software Automation.

Using the Process Image

How can you assign variables to the physical I/Os in CoDeSys ?

Smart Software Solutions

We software Automation.

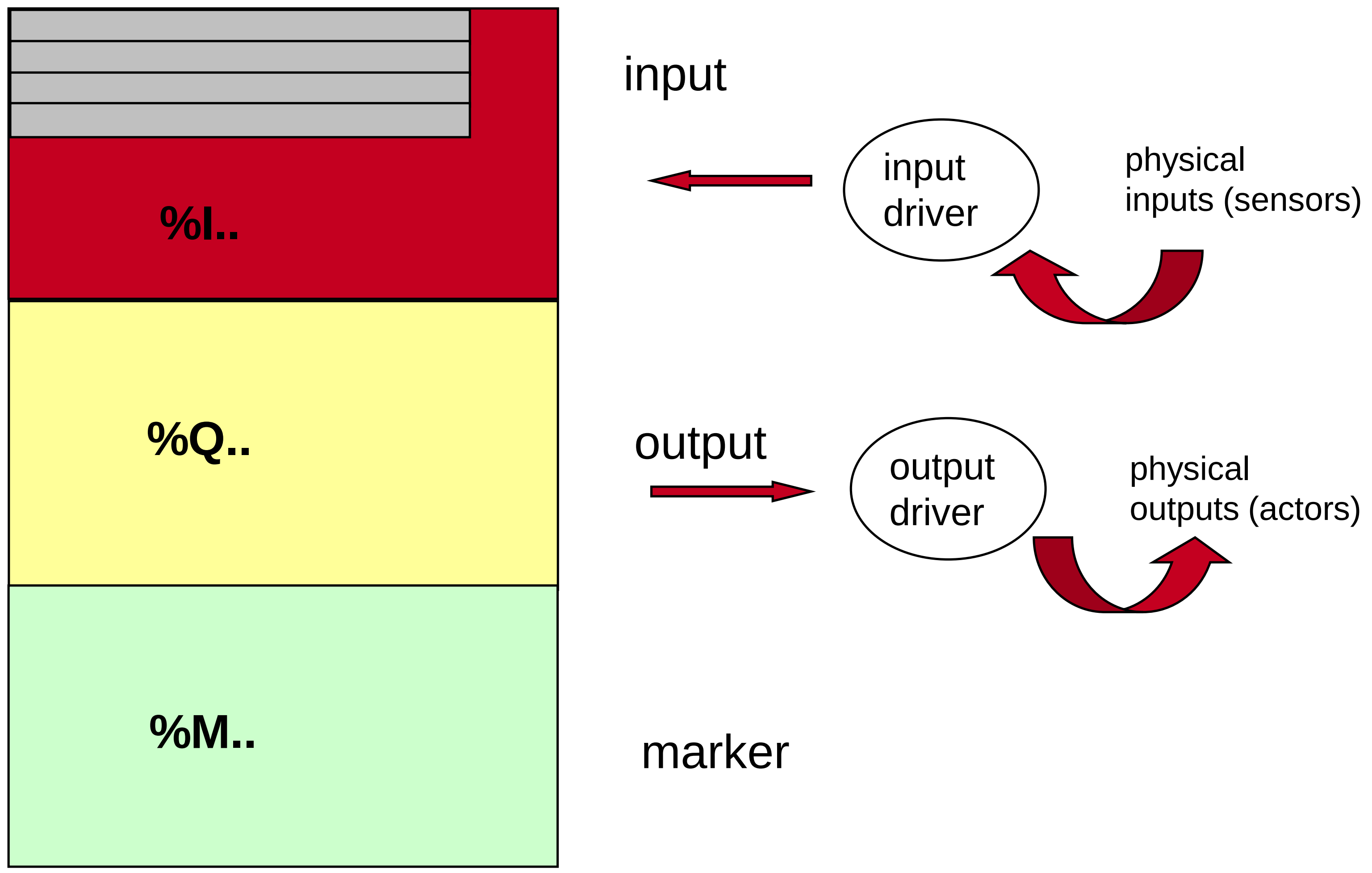
Introduction

After this module you will be familiar with all the different ways to assign variables to the process image.

Agenda

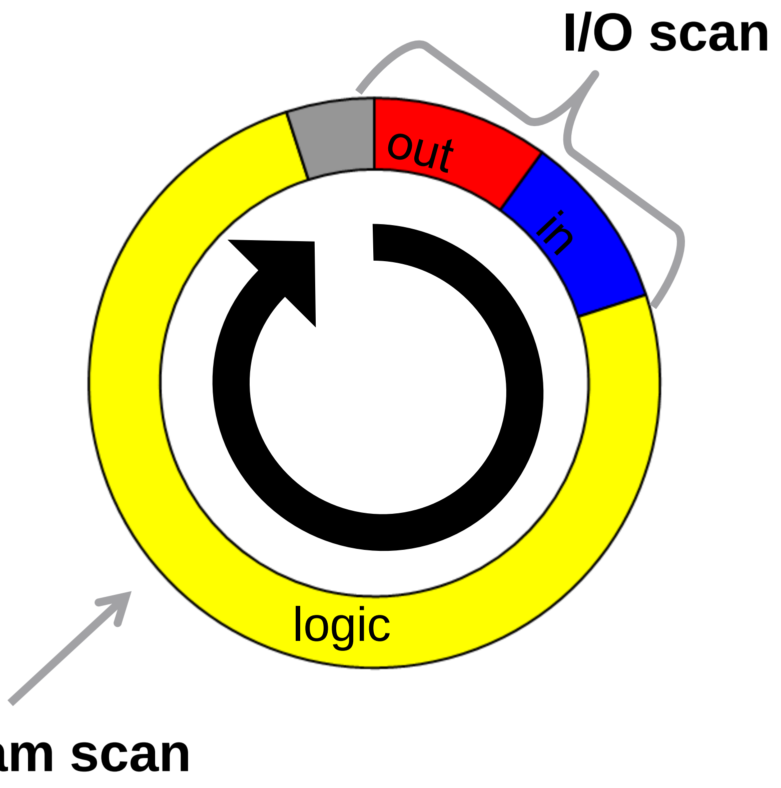
- common
  - process (data) image
  - PLC cycle
  - address syntax
- using the process image
  - configurator
    - create new global variables
    - map to existing variables
  - assign by declaration
  - variable configuration (VAR_CONFIG)

Process Image



PLC Cycle

PLC Device



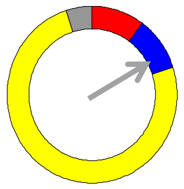
🟦 poll inputs

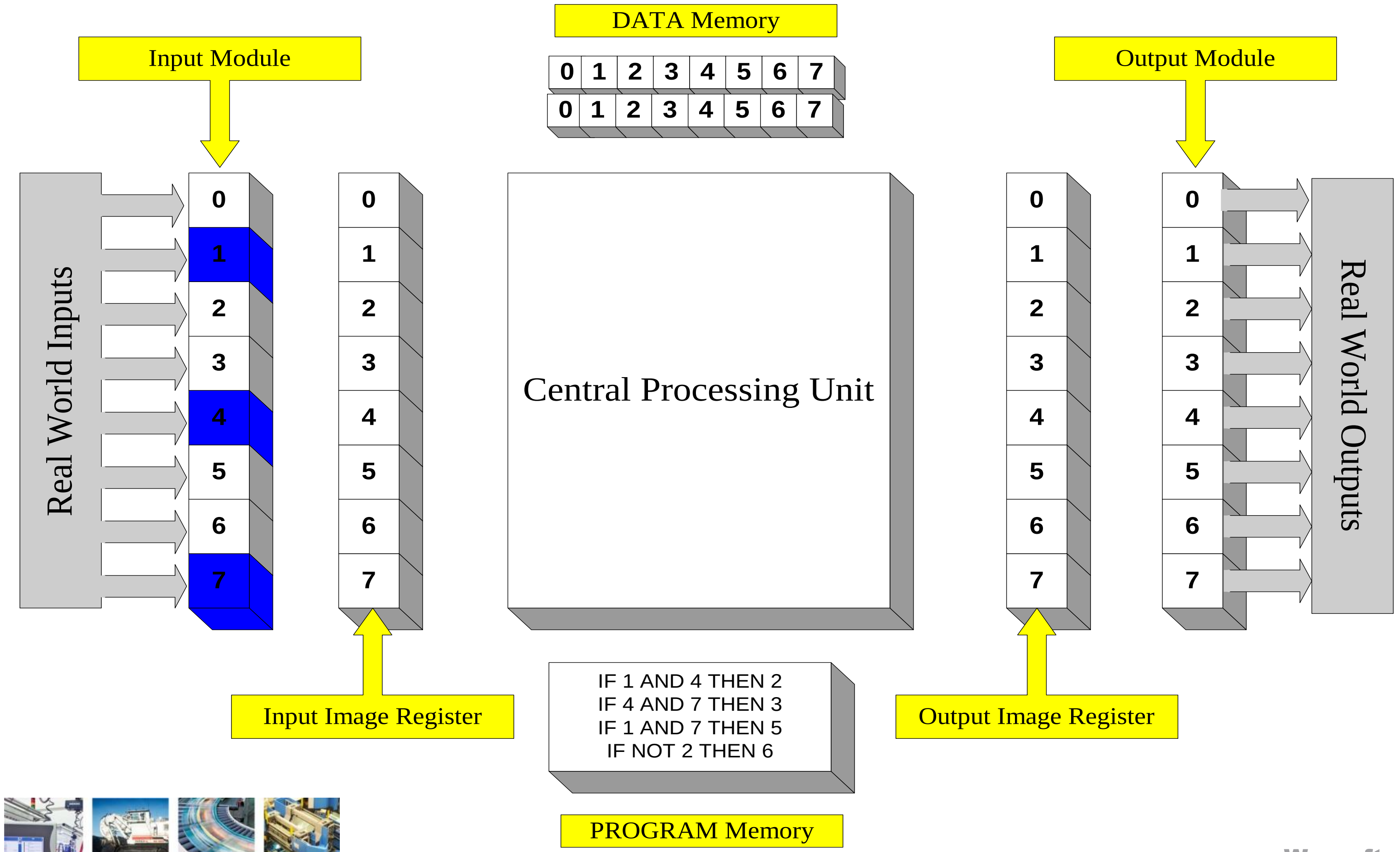
🟨 evaluate logic

⬜ communication

🟥 post outputs

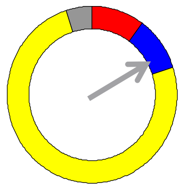
ScanCycle – Process Image

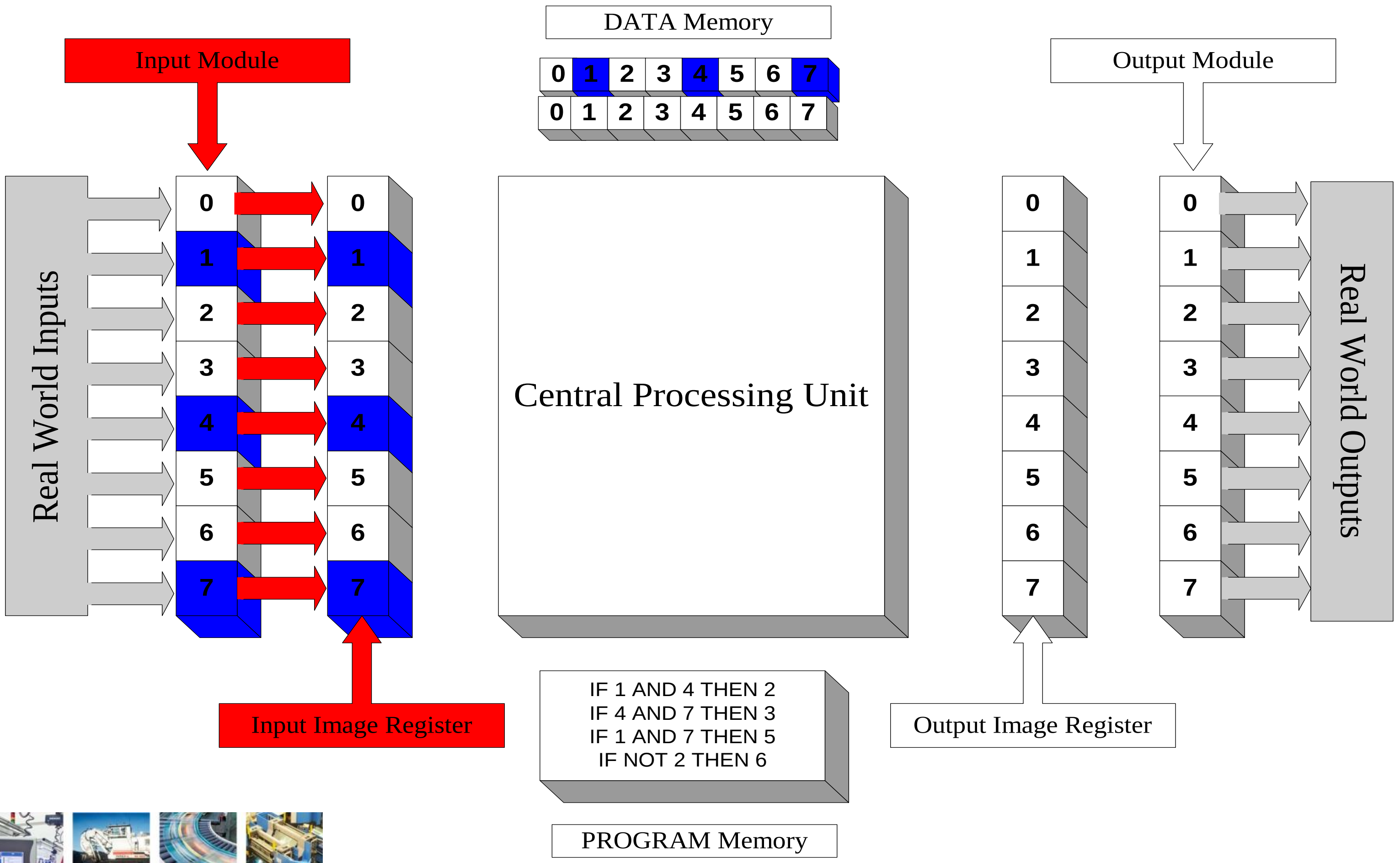




```
IF 1 AND 4 THEN 2
IF 4 AND 7 THEN 3
IF 1 AND 7 THEN 5
IF NOT 2 THEN 6
```

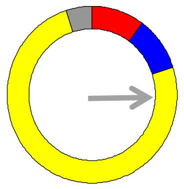
I/O Scan – Input Segment

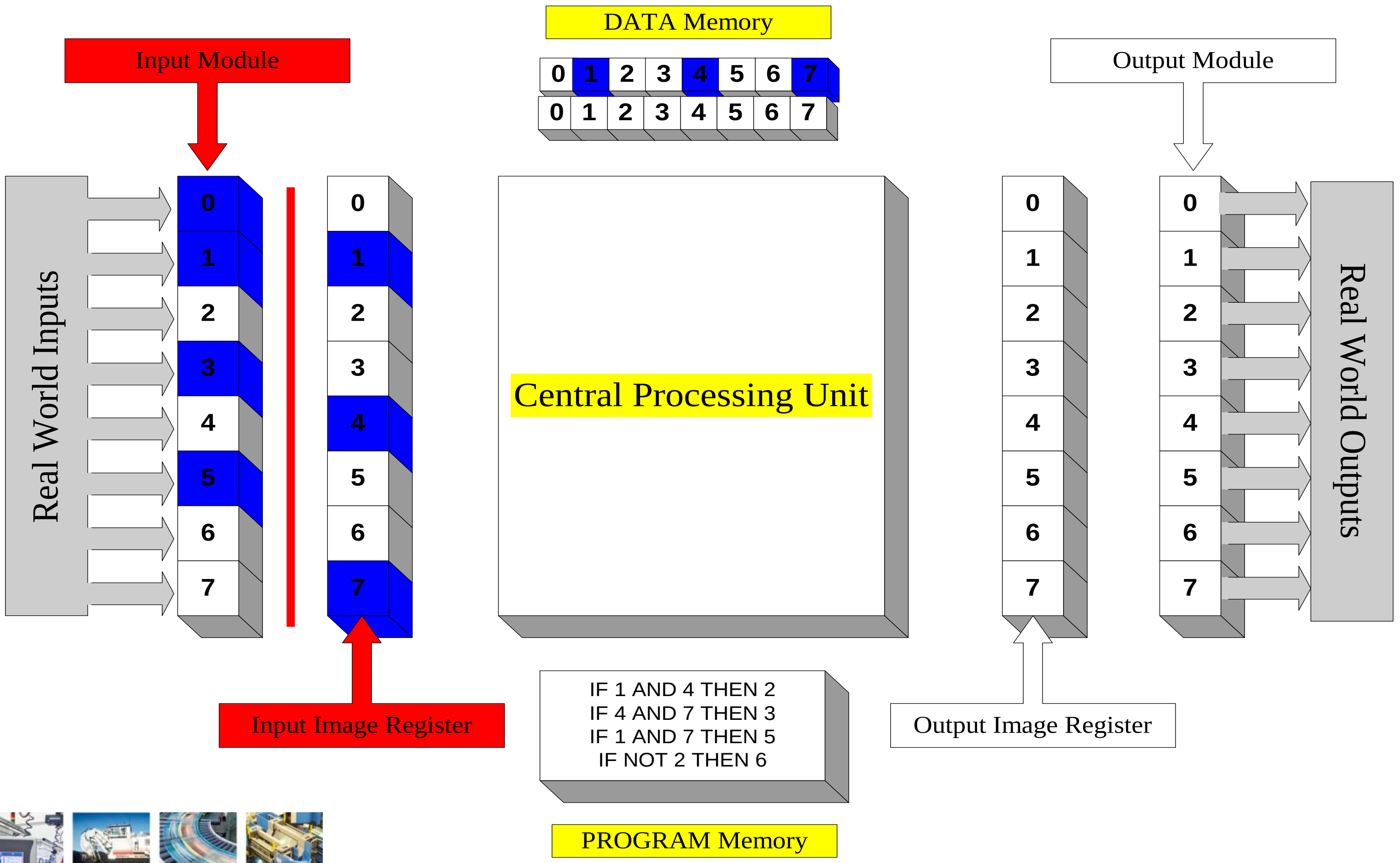




```
IF 1 AND 4 THEN 2
IF 4 AND 7 THEN 3
IF 1 AND 7 THEN 5
IF NOT 2 THEN 6
```

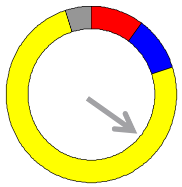
End I/O Scan – Begin Program Scan

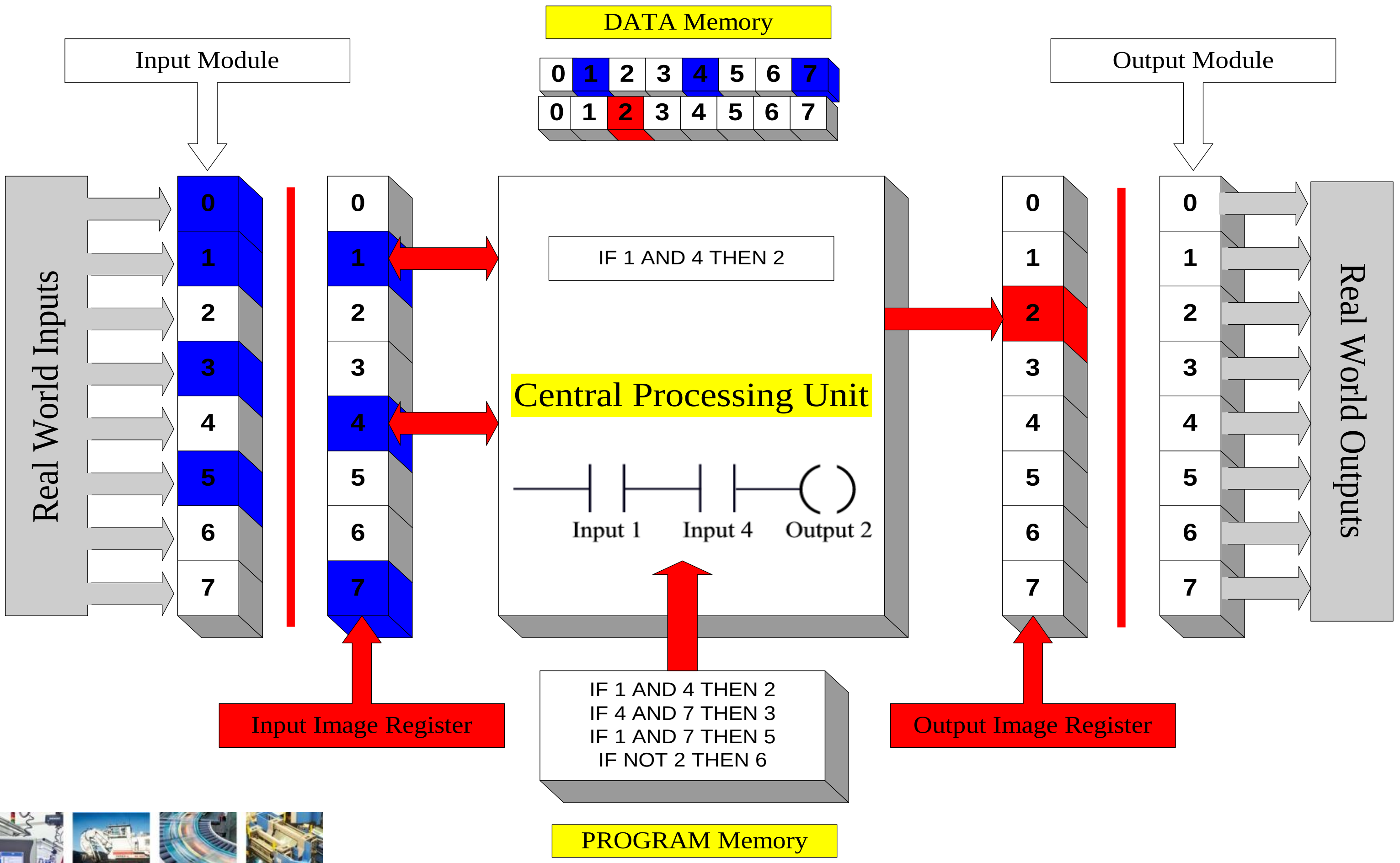




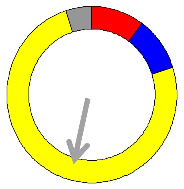
```
IF 1 AND 4 THEN 2
IF 4 AND 7 THEN 3
IF 1 AND 7 THEN 5
IF NOT 2 THEN 6
```

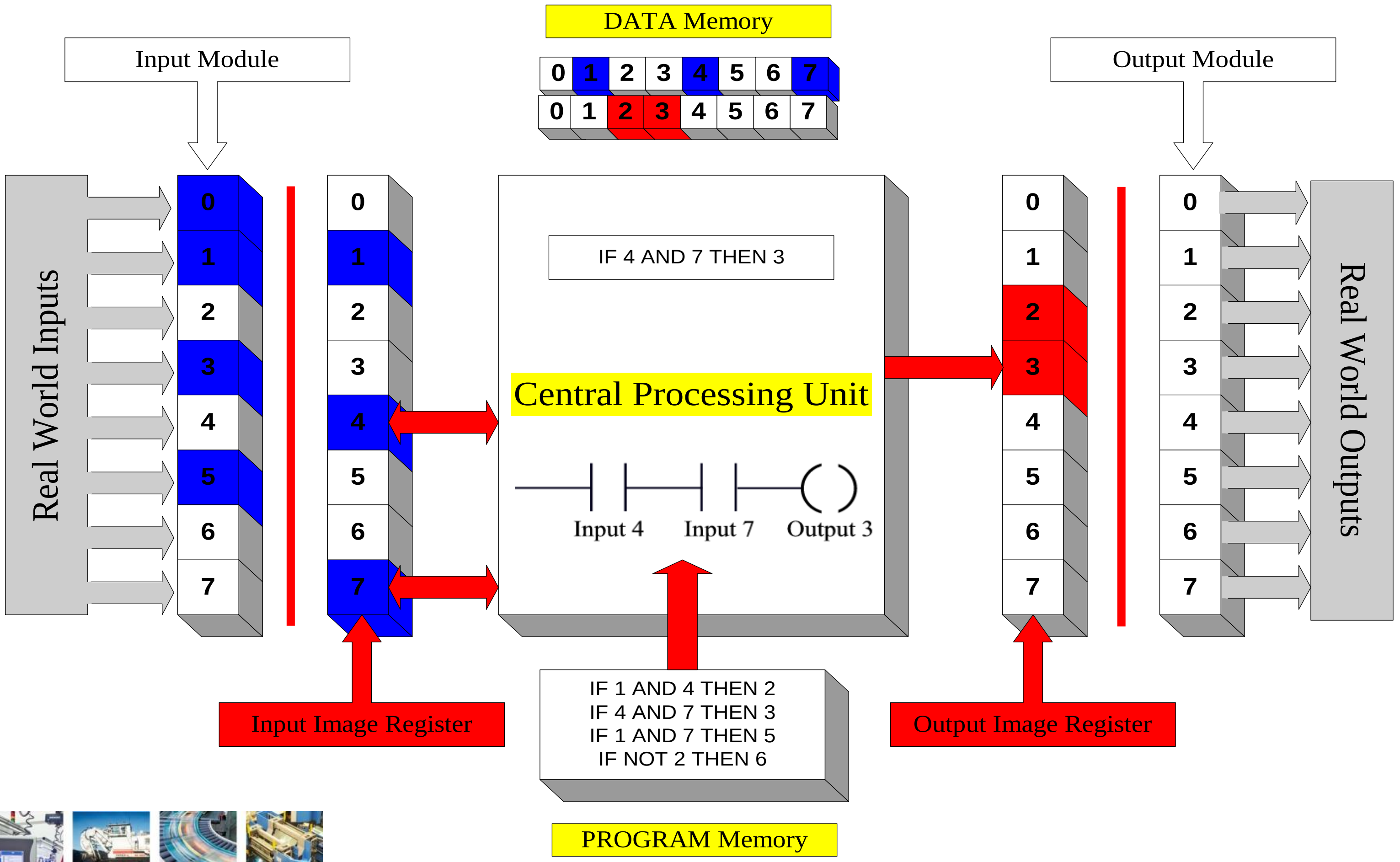
Evaluate Rung 1





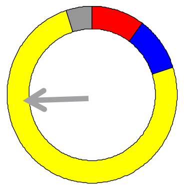
Evaluate Rung 2

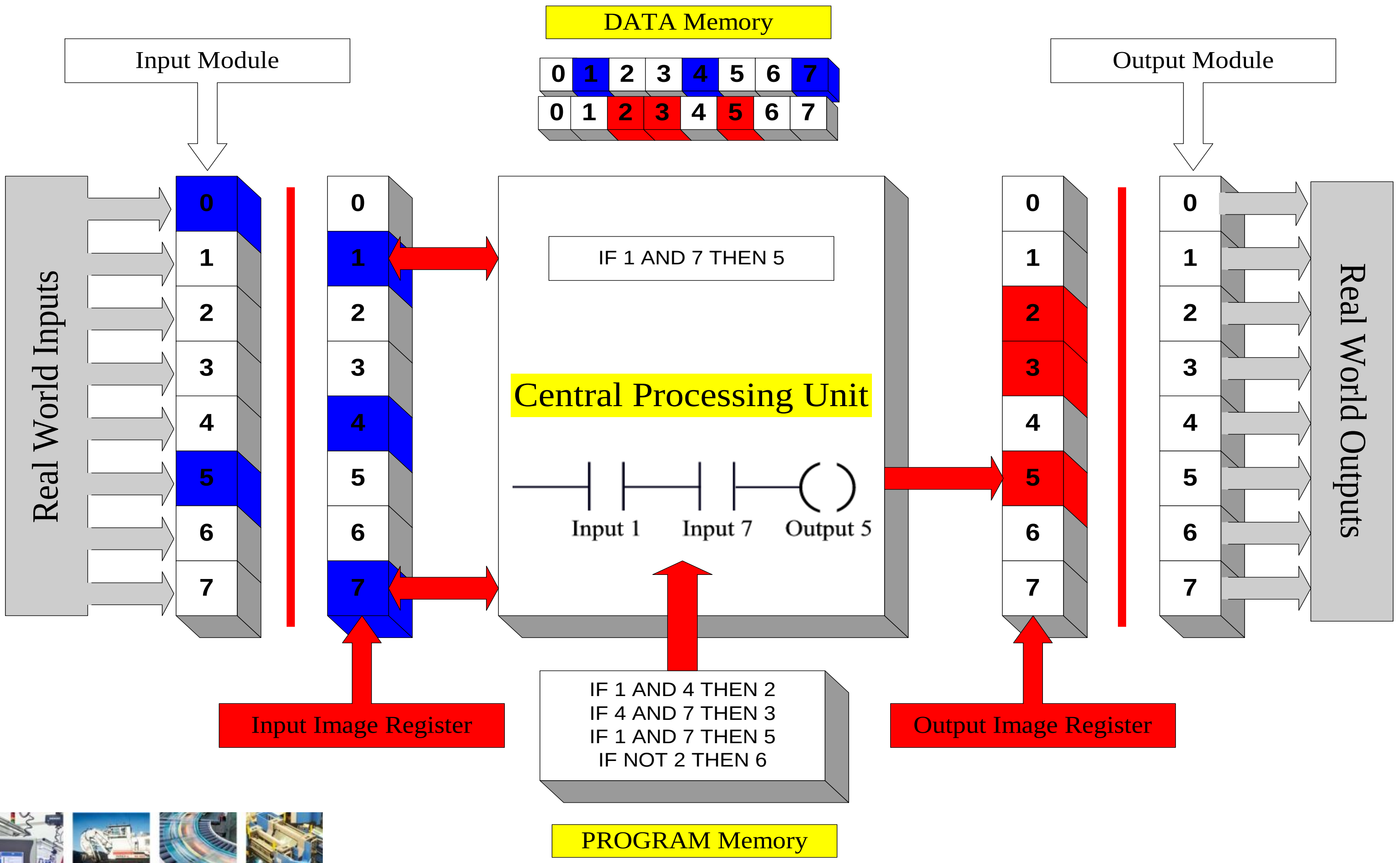




IF 1 AND 4 THEN 2
IF 4 AND 7 THEN 3
IF 1 AND 7 THEN 5
IF NOT 2 THEN 6

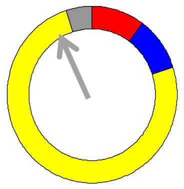
Evaluate Rung 3

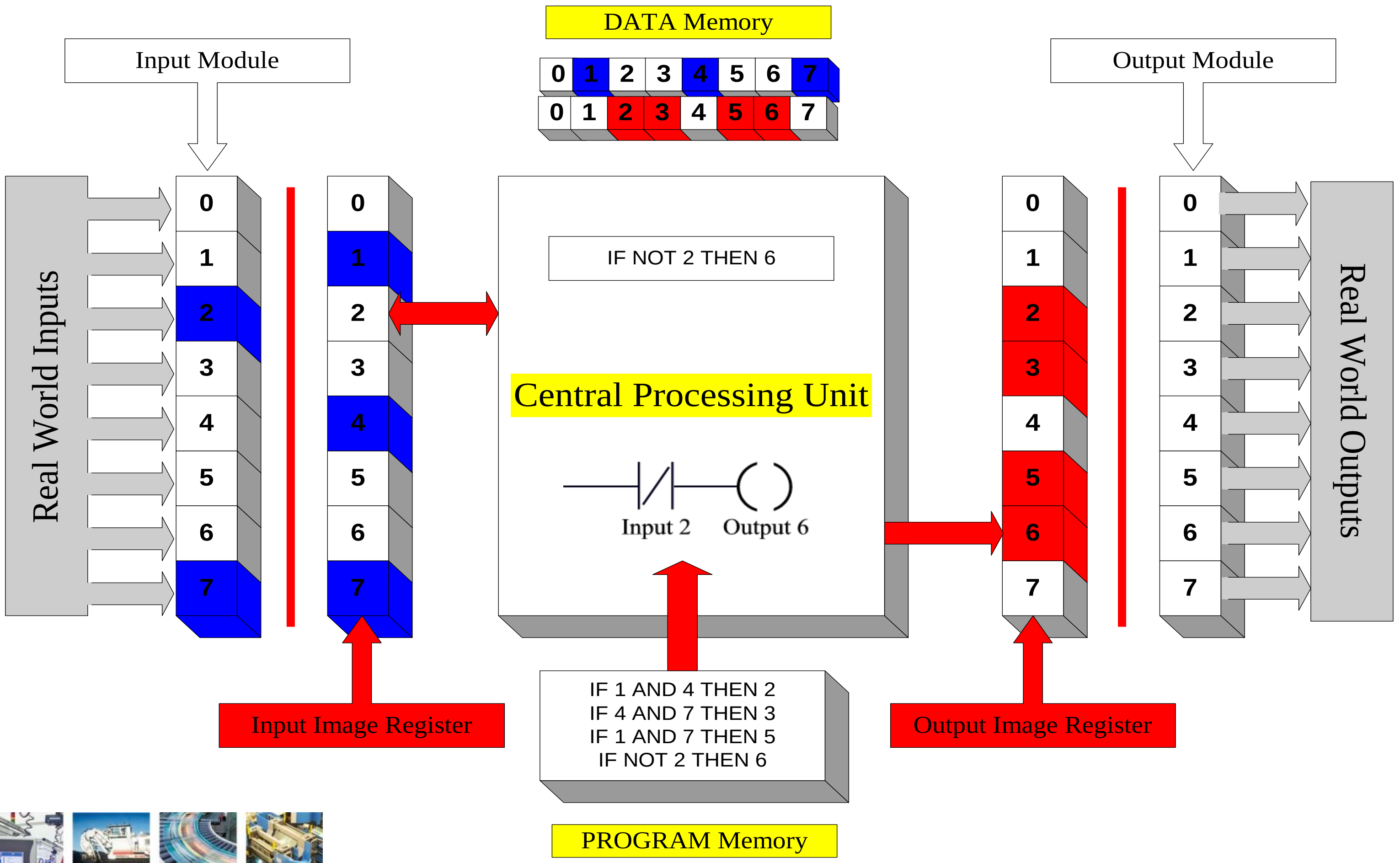




IF 1 AND 4 THEN 2
IF 4 AND 7 THEN 3
IF 1 AND 7 THEN 5
IF NOT 2 THEN 6

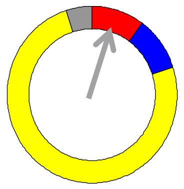
Evaluate Rung 4

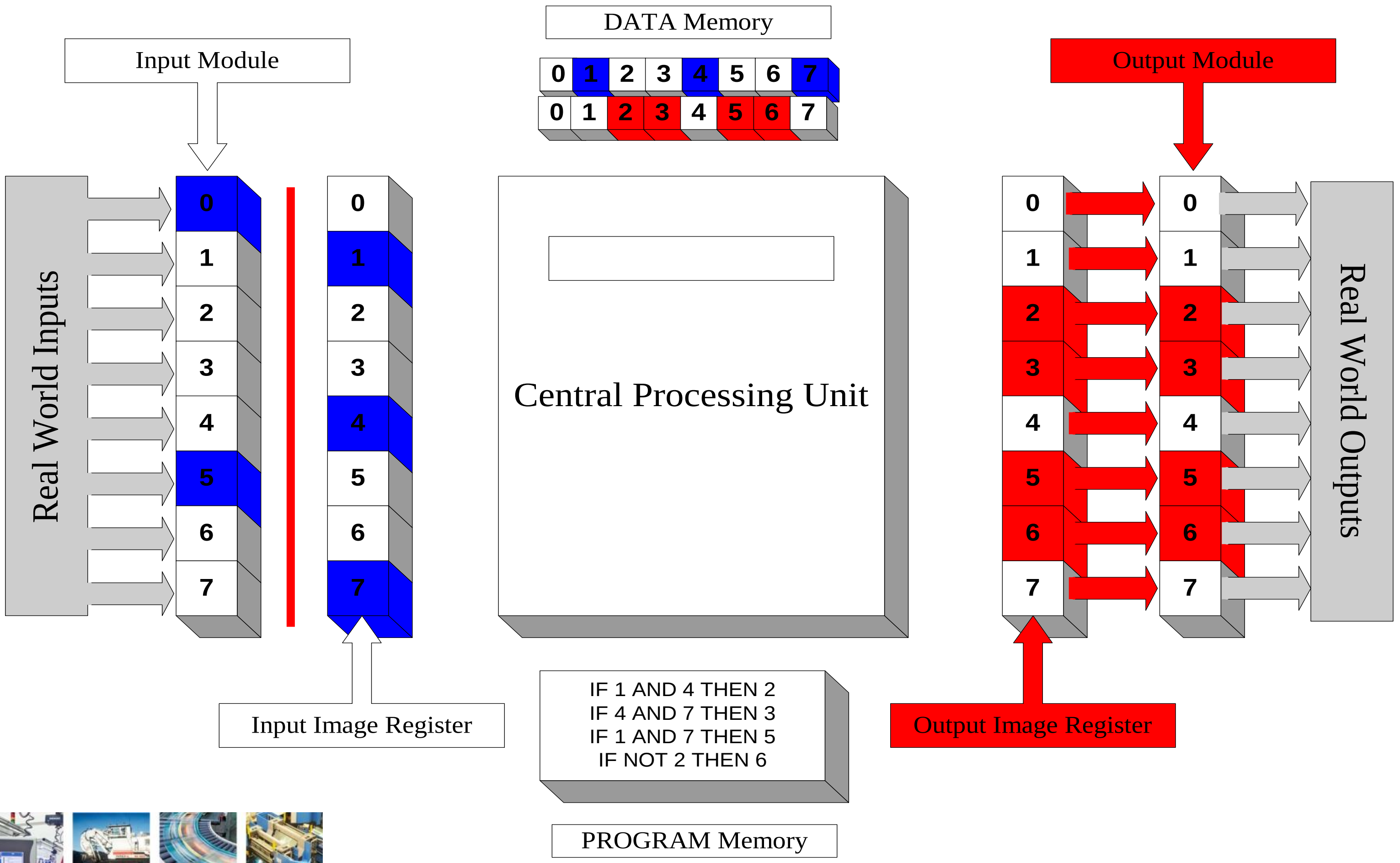




PLC Cycle

Post Outputs





Address Syntax

%•X	0.0 – 0.7	1.0 – 1.7	2.0 – 2.7	3.0 – 3.7	4.0 – 4.7	5.0 – 5.7
%•B	0	1	2	3	4	5
%•W	0		1		2	
%•D	0				1	

- Percentage sign '%'

%IX$\boxed{1}$.1 %QW2 %MD1

- Area prefix

I	input
Q	output
M	marker

- Size

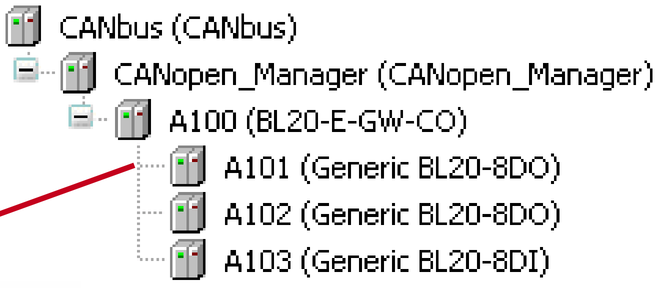
X	single bit
None	*single bit*
B	byte (8 bits)
W	word (16 bits)
D	double word (32 bits)

• I,Q oder M

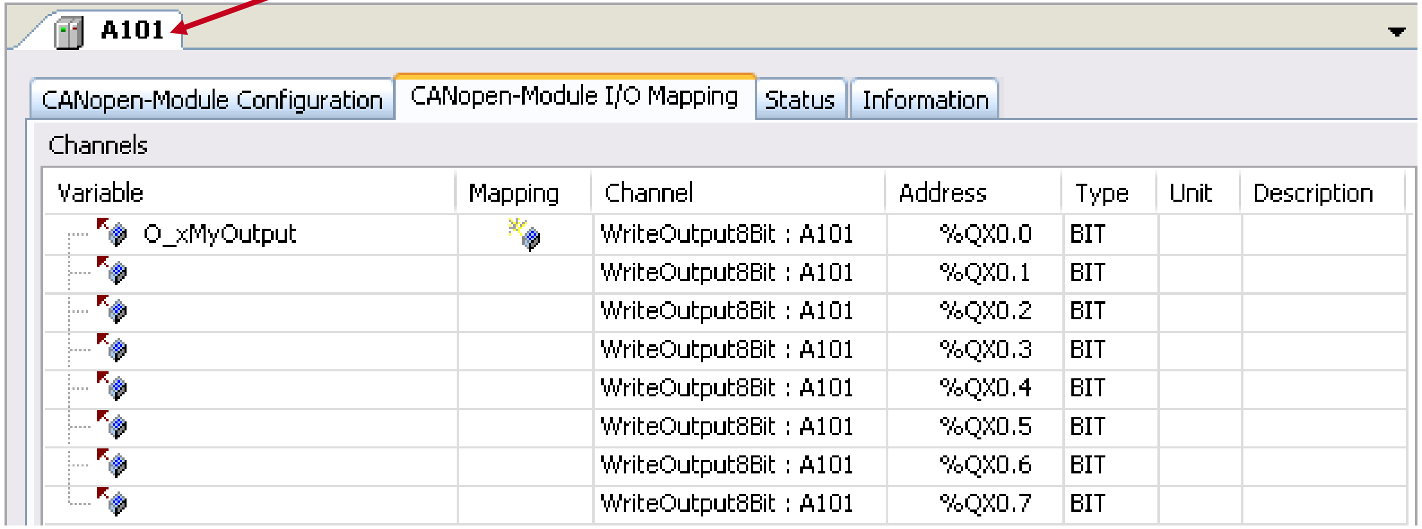
Configurator

= Create New Variable

- get access in program


CANbus (CANbus)
CANopen_Manager (CANopen_Manager)
A100 (BL20-E-GW-CO)
A101 (Generic BL20-8DO)
A102 (Generic BL20-8DO)
A103 (Generic BL20-8DI)

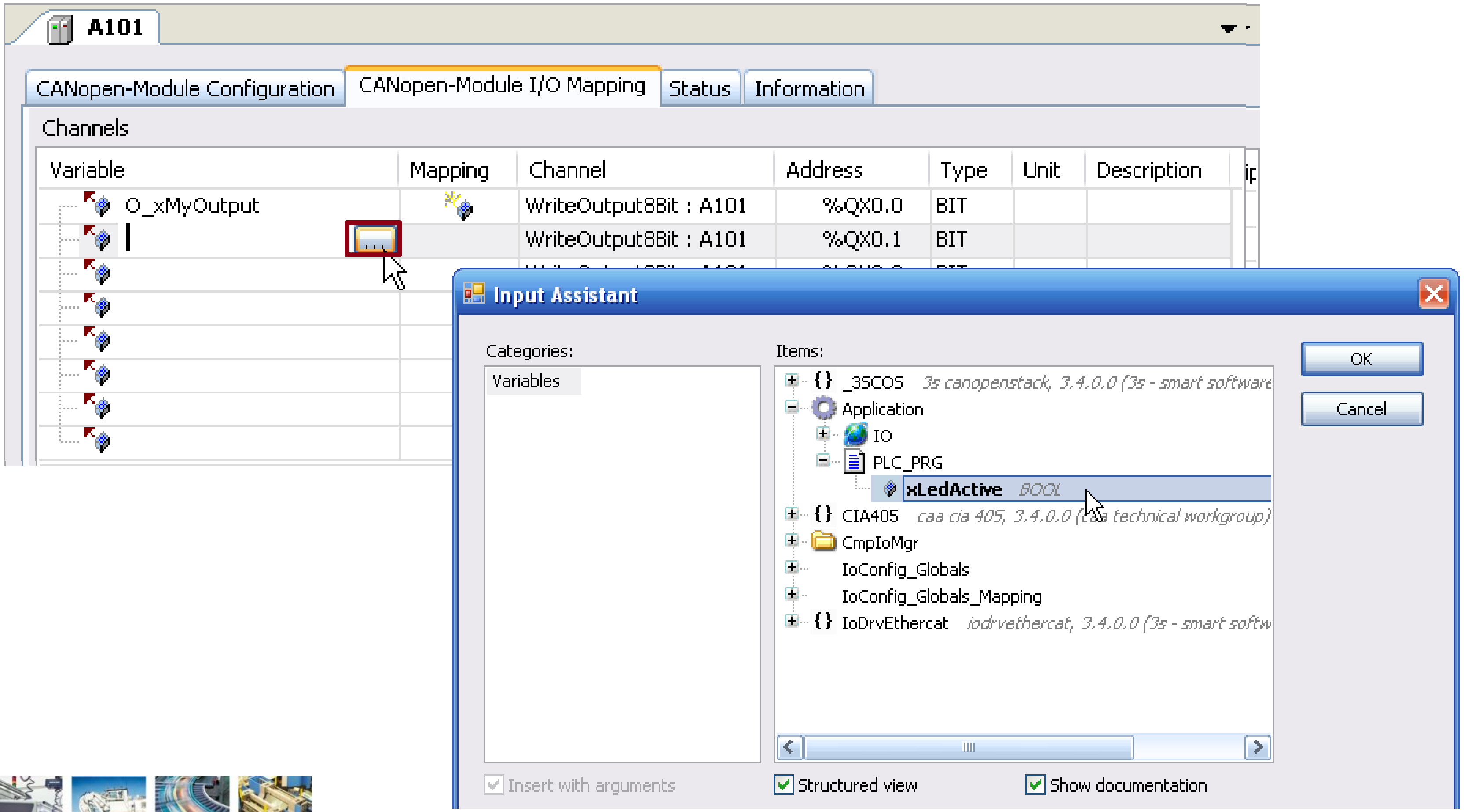
```
O_xMyOutput := TRUE;

IoConfig_Globals_Mapping.O_xMyOutput := TRUE;
```



Variable	Mapping	Channel	Address	Type	Unit	Description
O_xMyOutput		WriteOutput8Bit : A101	%QX0.0	BIT		
		WriteOutput8Bit : A101	%QX0.1	BIT		
		WriteOutput8Bit : A101	%QX0.2	BIT		
		WriteOutput8Bit : A101	%QX0.3	BIT		
		WriteOutput8Bit : A101	%QX0.4	BIT		
		WriteOutput8Bit : A101	%QX0.5	BIT		
		WriteOutput8Bit : A101	%QX0.6	BIT		
		WriteOutput8Bit : A101	%QX0.7	BIT		

= Map to Existing Variable

- use the “Input Assistant” to select variable



Variable	Mapping	Channel	Address	Type	Unit	Description
O_xMyOutput		WriteOutput8Bit : A101	%QX0.0	BIT		
		WriteOutput8Bit : A101	%QX0.1	BIT		

Items:
- _3SCOS *3s canopenstack, 3.4.0.0 (3s - smart software*
- Application
  - IO
  - PLC_PRG
    - **xLedActive** *BOOL*
- CIA405 *caa cia 405, 3.4.0.0 (caa technical workgroup)*
- CmpIoMgr
- IoConfig_Globals
- IoConfig_Globals_Mapping
- IoDrvEthercat *iodrvethercat, 3.4.0.0 (3s - smart softw*

☐ Insert with arguments ✓ Structured view ✓ Show documentation

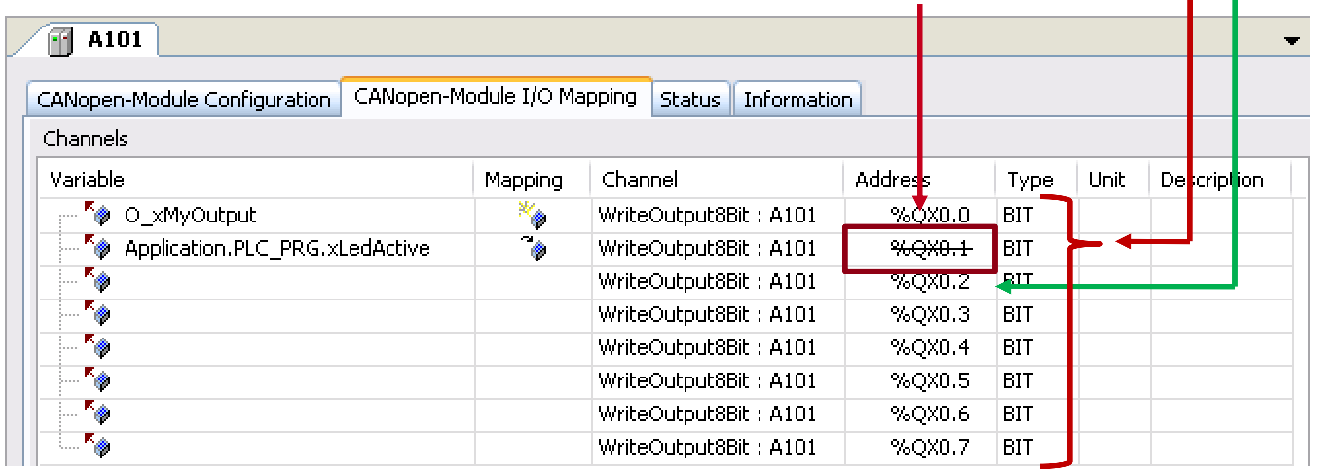
Assign by Declaration

Declare in PROGRAM or GVL

- Example

```
xValue   AT %QX0.2  : BOOL;

byValue  AT %QB0    : USINT;
```

except this address



A101

CANopen-Module Configuration | CANopen-Module I/O Mapping | Status | Information

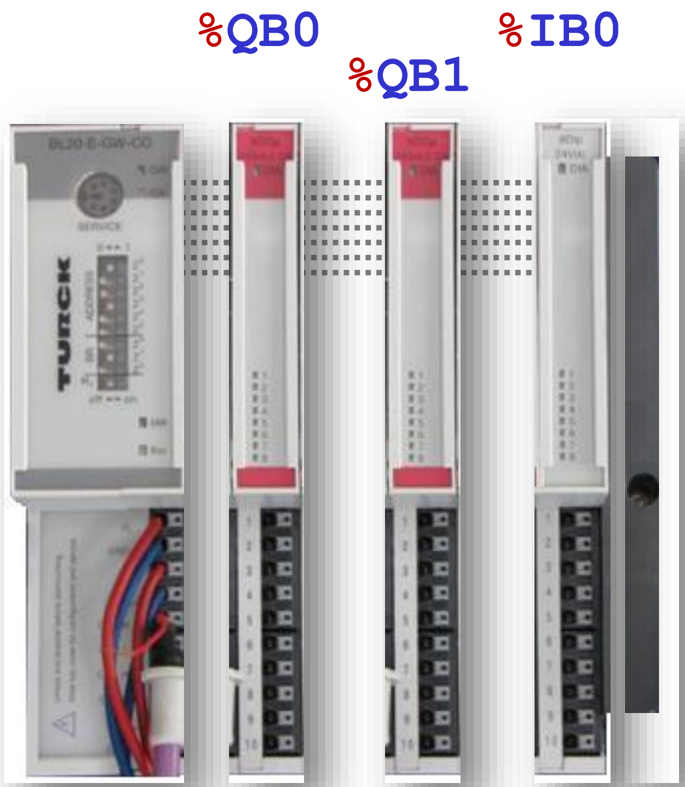
Channels

Variable	Mapping	Channel	Address	Type	Unit	Description
O_xMyOutput		WriteOutput8Bit : A101	%QX0.0	BIT		
Application.PLC_PRG.xLedActive		WriteOutput8Bit : A101	~~%QX0.1~~	BIT		
		WriteOutput8Bit : A101	%QX0.2	BIT		
		WriteOutput8Bit : A101	%QX0.3	BIT		
		WriteOutput8Bit : A101	%QX0.4	BIT		
		WriteOutput8Bit : A101	%QX0.5	BIT		
		WriteOutput8Bit : A101	%QX0.6	BIT		
		WriteOutput8Bit : A101	%QX0.7	BIT		

Variable Configuration (VAR_CONFIG)

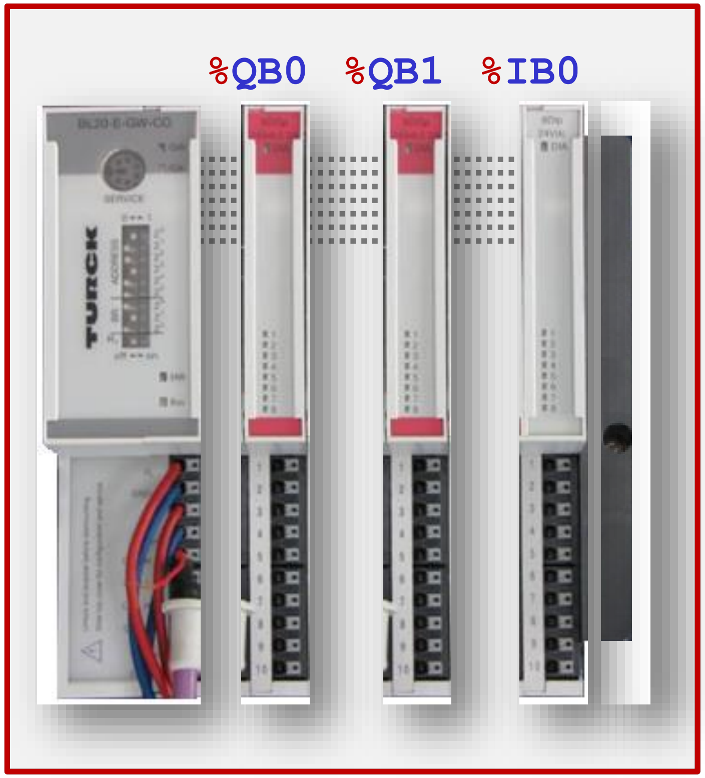
Assign Structures or FB Instances to I/Os

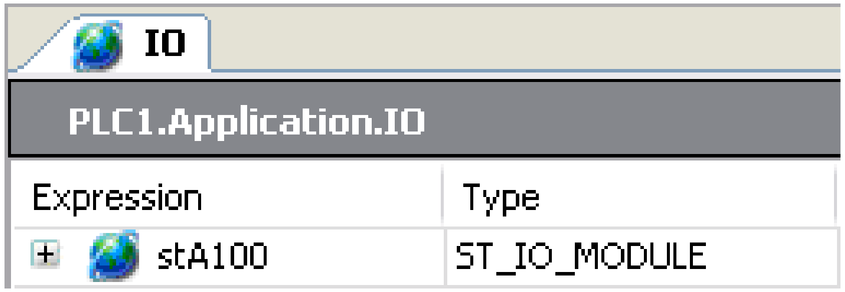
- Example
  - CANopen slave
    - 8 DO
    - 8 DO
    - 8 DI



Variable Configuration (VAR_CONFIG)

Target



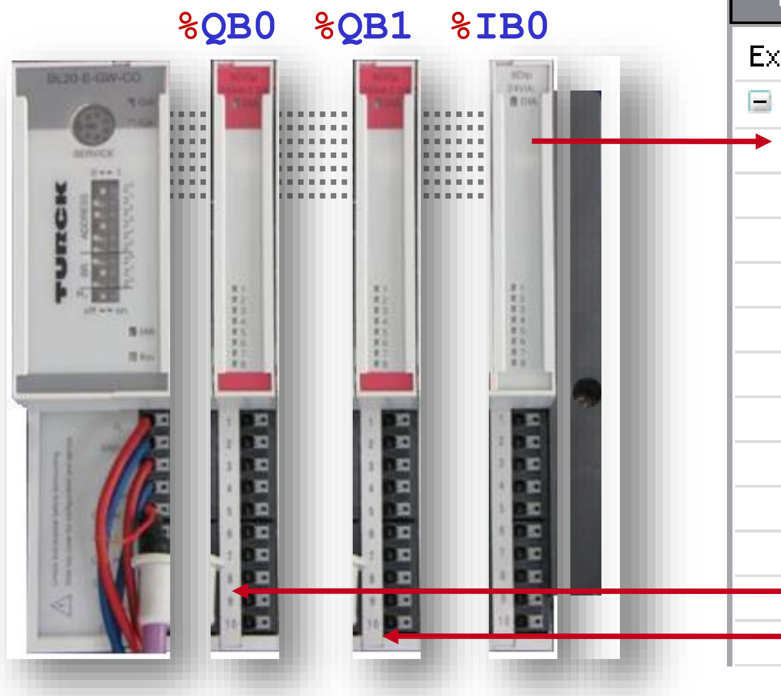


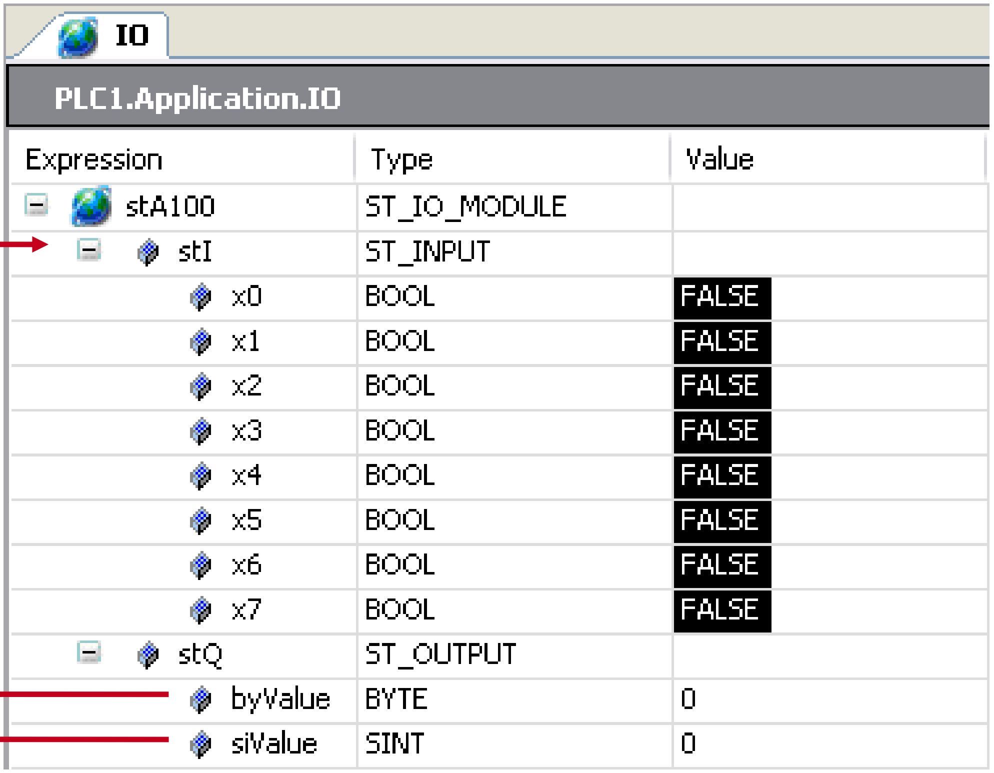
PLC1.Application.IO

Expression	Type
stA100	ST_IO_MODULE

Variable Configuration (VAR_CONFIG)

Target





IO

PLC1.Application.IO

Expression	Type	Value
stA100	ST_IO_MODULE	
stI	ST_INPUT	
x0	BOOL	FALSE
x1	BOOL	FALSE
x2	BOOL	FALSE
x3	BOOL	FALSE
x4	BOOL	FALSE
x5	BOOL	FALSE
x6	BOOL	FALSE
x7	BOOL	FALSE
stQ	ST_OUTPUT	
byValue	BYTE	0
siValue	SINT	0

Variable Configuration (VAR_CONFIG)

How to do it?

IO

```
VAR_GLOBAL
    stA100 : ST_IO_MODULE;
END_VAR
```

```
TYPE ST_IO_MODULE :
STRUCT
    stI AT %I* : ST_INPUT;
    stQ AT %Q* : ST_OUTPUT;
END_STRUCT
END_TYPE
```

```
TYPE ST_INPUT :
STRUCT
    x0 : BIT;
    x1 : BIT;
    x2 : BIT;
    x3 : BIT;
    x4 : BIT;
    x5 : BIT;
    x6 : BIT;
    x7 : BIT;
END_STRUCT
END_TYPE
```

IO_CONFIG

```
VAR_CONFIG
    IO.stA100.stI AT %IB0 : ST_INPUT;
    IO.stA100.stQ AT %QB0 : ST_OUTPUT;
END_VAR
```

```
TYPE ST_OUTPUT :
STRUCT
    byValue : BYTE;
    siValue : SINT;
END_STRUCT
END_TYPE
```

Variable Configuration (VAR_CONFIG)

Now it is very easy to install a new device with the same setup.

%QB0 %QB1 %IB0

%QB2 %QB3 %IB1

Variable Configuration (VAR_CONFIG)

Now it is very easy to install a new device with the same setup.

%QB2 %QB3 %IB1

IO

```
VAR_GLOBAL
    stA100 : ST_IO_MODULE;
    stA200 : ST_IO_MODULE;
END_VAR
```

IO_CONFIG

```
VAR_CONFIG
    IO.stA100.stI AT %IB0 : ST_INPUT;
    IO.stA100.stQ AT %QB0 : ST_OUTPUT;

    IO.stA200.stI AT %IB1 : ST_INPUT;
    IO.stA200.stQ AT %QB2 : ST_OUTPUT;
END_VAR
```

Summary

- There are four different ways to assign variables to the process image:

  - with configurator
    - create new variables
    - map to existing variables
  - declaration
    - assign variable to address
    - VAR_CONFIG