HITACHI
Inspire the Next

HX & CODESYS Trainning

2019.3.12

HITACHI
Inspire the Next

Lineup

Programming software		Compact type	Modular type
Ladder Editor (since 1997)	LD	MICRO-EH 	EH-150 
Control Editor (since 2006)	LD	MICRO-EHV 	EH-150 EHV series 
HX-CODESYS (since 2016) (old version: EHV-CODESYS)	LD, FBD, IL, ST, SFC, CFC (IEC61131-3)	MICRO-EHV+ 	EH-150 EHV+ series 
			HX series 

Expansion units are compatible.
I/O modules are compatible.

HITACHI
Inspire the Next

Outline of IEC61131-3

Outline of IEC61131-3

HITACHI
Inspire the Next

Features

- Standardized PLC programming way
- Programming with the use of variable name
- 5 programming languages
- Structured program
- High re-usability of program / function

Outline of IEC61131-3

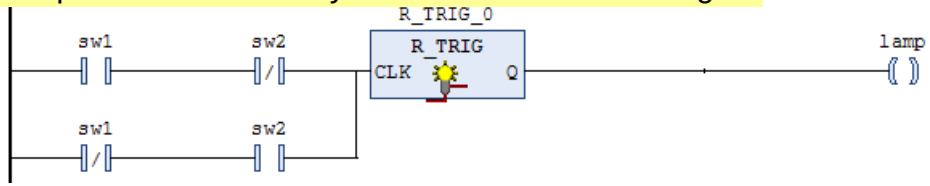
HITACHI
Inspire the Next

IEC 4 languages + 1 element + non-IEC 1 language

LD (Ladder Logic)

Usage: Interlock circuit etc.

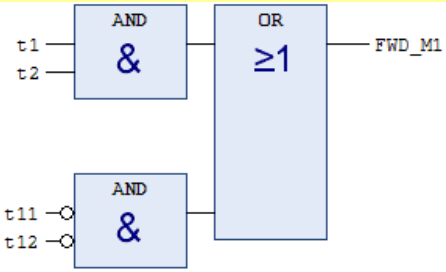
Graphic editor for binary variables and interlocking



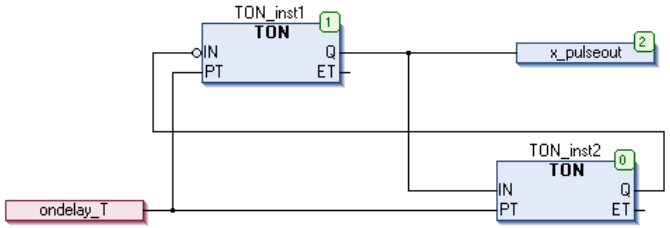
FBD (Function Block Diagram)

Usage: Data flow etc.

Graphic editor for networking and data flow



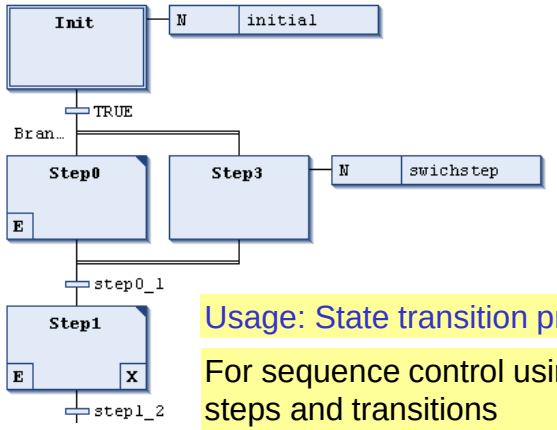
CFC (Continuous Function Chart)



Usage: emphasis on visibility, small applications, etc.

FBD editor with free layout incl. feedback paths

SFC (Sequential Function Chart)



Usage: State transition processing etc.

For sequence control using steps and transitions

ST (Structured Text)

```
count_M3 := count_M3 + 1;
L2_wait_timer (IN:= FALSE, PT:= T#3.6S);
L2_wait_timer (IN:= TRUE);
FOR i := 0 TO count_T
    L2_wait_timer();
END_FOR;
elp_wait:= L2_wait_timer.ET;
IF count_GS < 24 THEN
    WHILE test_out5 < 10 DO
        max_R2:=max_R2+1;
    END_WHILE
END_IF;
```

Usage: Conditional branching, repeating etc.

For conditional branching and loop

IL (Instruction List)

```
1   CAL     TON_IL(
              IN:= ton_input,
              PT:= T#50000S,
              ET=> ton_elp)
    LD      TON_IL.Q
    ST      ton_output
2   LD      Lift_A
    OR      sta_L4_covyr
    ST      IL_sample_out
```

Usage: Miniaturization of applications etc.

For small application or restoring old programs

Outline of IEC61131-3

HITACHI
Inspire the Next

IEC data types (Elementary data types)

No.	Group	Data type	Size	Range
1	Boolean	BOOL	1	0, 1 (TRUE, FALSE)
2	Bit array	BYTE	8	0 ～ 255
3		WORD	16	0 ～ 65,535
4		DWORD	32	0 ～ 4,294,967,295
5		LWORD	64	0 ～ 2^{64} -1
6	Integer	SINT	8	-128 ～ 127
7		INT	16	-32,768 ～ 32,767
8		DINT	32	-2,147,483,648 ～ 2,147,483,647
9		LINT	64	-2^{63} ～ 2^{63} -1
10	Unsigned integer	USINT	8	0 ～ 255
11		UINT	16	0 ～ 65,535
12		UDINT	32	0 ～ 4,294,967,295
13		ULINT	64	0 ～ 2^{64} -1
14	Real number	REAL	32	±1.175…x10^{-38} ～ 3.402…x10^{38}
15		LREAL	64	±2.225…x10^{-308} ～ 1.797…x10^{308}
16	Time	TIME	32	0 ～ 4,294,967,295 ms
17		LTIME	64	2^{64} -1 ms

Outline of IEC61131-3

HITACHI
Inspire the Next

IEC data types (Elementary data types)

No.	Group	Data type	Size	Format
18	1-byte string	STRING	8x n	‘abc’
19	2-byte string	WSTRING	16x n	“abc”
20	Date	DATE	32	d#2016-12-31 or DATE#2016-12-31
21	Date and Time	DATE_AND_TIME	32	dt#2016-12-31-23:59:59 or DATE_AND_TIME#2016-12-31-23:59:59
22	Time	TIME_OF_DAY	32	tod#23:59:59 or TIME_OF_DAY#23:59:59

IEC data types (User-defined types)

No.	Group	Format
1	Array	`ARRAY [0..255] OF WORD;` `ARRAY [0..11, 0..22, 0..33] OF REAL; // 3-dimentioal array`
2	Partial range data type	`WORD(0..4095)`

3	Enumerator	`TYPE COLOR:`
		`(Red, Yellow, Green);`
		`END_TYPE`

4	Structure	`TYPE STRUCT_sample`
		`STRUCT`
		`  ID : WORD;`
		`  Flag : BOOL;`
		`  Weight : REAL;`
		`END_STRUCT`
		`END_TYPE`

Outline of IEC61131-3

HITACHI
Inspire the Next

Numeric literals

No.	Group	Example	Remarks
1	Integer	`-12, 0, +123`	Decimal number
2	Real number	`0.0, 3.14159, 1.234_567`	Underline is ignored
3	Exponent	`1.2E6, 1.23E+3, 1.23E-6`	
4	Binary	`2#11, 2#1110_1111`	Underline is ignored
5	Hexadecimal	`16#FF, 16#1234_FFFF`	Underline is ignored
6	BOOL	`FALSE, TRUE`	

Time and Date literals

No.	Group	Example	Remarks

1	Time	`TIME#30ms, TIME#25h15m`
		or
		`T#30ms, T#25h15m`
			d: Day
			h: Hour
			m: Minute
			s: Second
			ms: Milli second
			us: Micro second

2	Date	`DATE#2016-1-1, DATE#1999-12-31` `Or D#2016-1-1, D#1999-12-31`	
3	Time of day	`TIME_OF_DAY#15:29:59, TIME_OF_DAY#15:30` `Or TOD#15:29:59, TOD#15:30`	
4	Date and time	`DATE_AND_TIME#2016-12-31-23:59:59` `Or DT#2016-12-31-23:59:59`	

Outline of IEC61131-3

HITACHI
Inspire the Next

IEC identifier (Variable)

Allowed characters : a to z, A to Z, 0 to 9 and _ (underscore)

Remarks

- Beginning with a number is not allowed.
- Multiple leading or multiple embedded underlines are not allowed.
- Upper and lower case letters are identical. (”abc” and ”ABC” are identical)
- Reserved words (FOR, WORD, AND, etc.) are not allowed.

Outline of IEC61131-3

HITACHI
Inspire the Next

IEC Instructions (FUNCTION)

Group	Name	Description	Group	Name	Description
Numeric operations	ABS	Absolute value	Selections	SEL	Input multiplexer
	SQRT	Square root		MAX	Maximum value
	LN	Natural logarithm		MIN	Minimum value
	LOG	Logarithm		LIMIT	Limiter
	EXP	Exponential function		MUX	Expandable multiplexer
	SIN	Sine	Comparisons	GT	Greater than
	COS	Cosine		GE	Greater or Equal
	TAN	Tangent		EQ	Equal
	ASIN	Arc sine		LE	Less or Equal
	ACOS	Arc cosine		LT	Less than
	ATAN	Arc tangent		NE	Not Equal
Arithmetic operations	ADD	Addition	Strings	LEN	Length
	MUL	Multiplication		LEFT	Extraction of left edge letter
	SUB	Subtraction		RIGHT	Extraction of right edge letter
	DIV	Division		MID	Extraction of middle letter
	MOD	Surplus		CONCAT	Concatenate
	EXPT	Power		INSERT	Insert
	MOVE	Assignment		DELETE	Delete
Logical operations	AND	Logical conjunction		REPLACE	Replace
	OR	Logical disjunction		FIND	Find
	XOR	Exclusive OR			
	NOT	Negation			

Refer Help for details (CODESYS Development System > Reference, Programming > Operators)

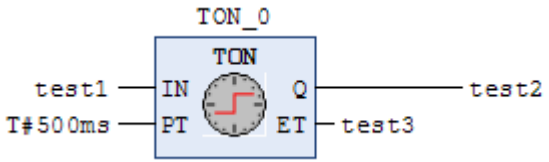
Outline of IEC61131-3

HITACHI
Inspire the Next

IEC Instructions (FUNCTION BLOCK)

Group	Name	Description
Numeric operations	SR	Bistable function block (set dominant)
	RS	Bistable function block (reset dominant)
Edge detection	R_TRIG	Rising edge
	F_TRIG	Falling edge
Counter	CTU	Up counter
	CTD	Down counter
	CTUD	Up-down counter
Timer	TP	Pulse
	TON	On-Delay Timer
	TOF	Off-Delay Timer

Example) TON at LD/FBD and ST



Equivalent

```
TON_0(IN:=test1, PT:=T#500MS, Q=>test2, ET=>test3);
```

HITACHI
Inspire the Next

IEC Instructions (Dedicated in ST)

Group	Name
Addition	+
Subtraction	−
Multiplication	*
Division	/
Surplus	MOD
Comparisons	<, >, <=, >=
Equality	=
Inequality	<>
Logical conjunction	AND
Logical disjunction	OR
Exclusive OR	XOR
Brackets	()
Pointer reference	^
Negation	NOT
Power	EXPT

Group	Name	Description
Assignment	:=	Right to left
Function call	<Function name>()	Function or function block call
Return	RETURN	Exit from function, function block or program

Branch	IF - THEN
	…
	ELSIF - THEN
	…
	ELSE
	…
	END_IF
		A group of statements is to be executed only if the associated Boolean expression evaluates to TRUE.

	CASE … OF ELSE … END_CASE	Multidirectional branching in accordance with value of a variable
Iteration	FOR … TO … BY … DO … END_FOR	Iteration the number of prepared times
	WHILE - DO … END_WHILE	Iteration until the condition gets FALSE
	REPEAT … UNTIL … END_REPEAT	Iteration until the condition gets TRUE
	CONTINUE	Skip remain part of one loop
	EXIT	Exit iteration loop

Outline of IEC61131-3

IEC Instructions (Dedicated in LD)

Group	Name	Description
Static contact		Contact
		Negated contact
Edge detection contact		Rising edge detection contact
		Falling edge detection contact
General coil		Coil
		Negated coil
Latch coil		Set coil
		Reset coil

Outline of IEC61131-3

HITACHI
Inspire the Next

What's POU?

POU: Program Organization Unit ≒ Program sheet
⇒ The minimum unit of program. There are 3 types of POU.

- Program (PRG)
- Function Block (FB)
- Function (FUN)

Type	Called by Task	How to call	I/O limitation	Store data
PRG	✓	Direct call	Free	✓
FB	-	Instance call	Free	✓
FUN	-	Direct call	Only 1 Output	-

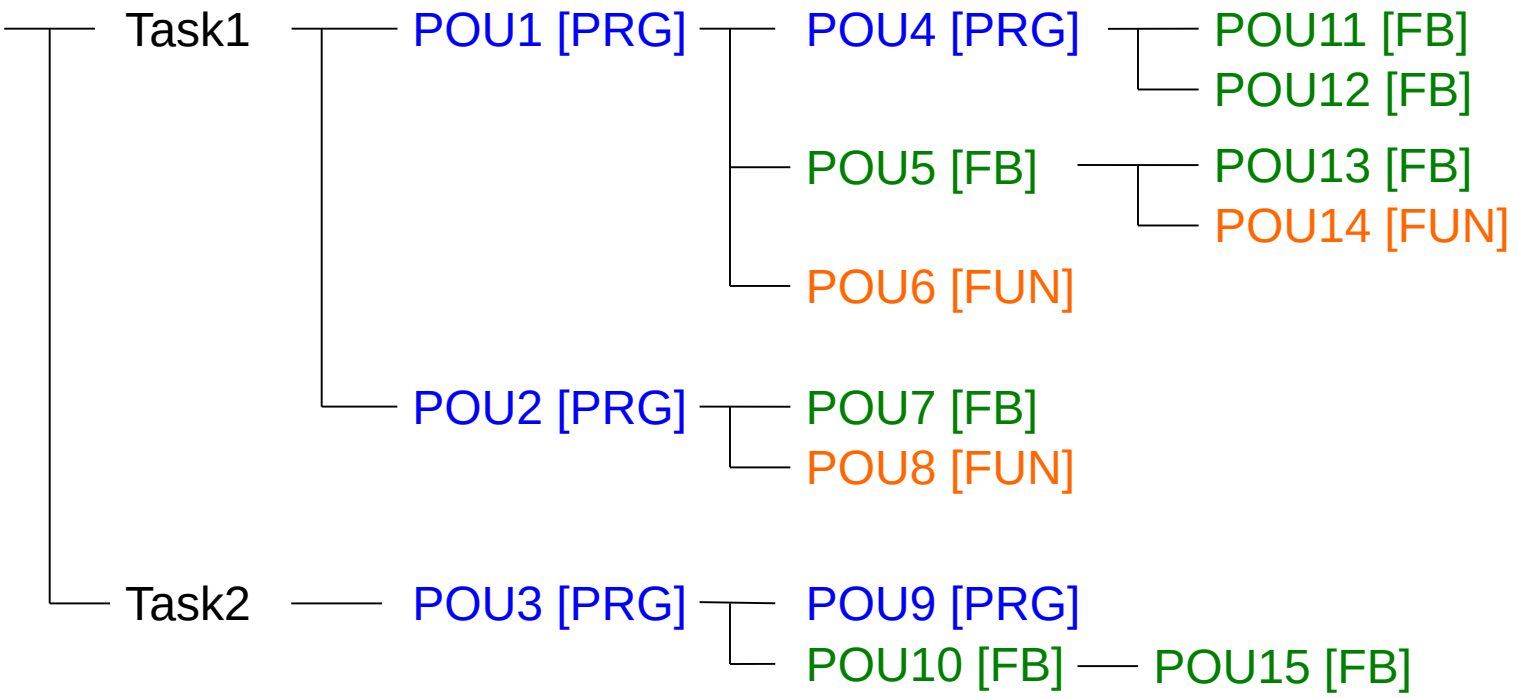
【Note】Note: Do not mix up FB and FBD. FB is one of the type of POU.
FBD is a name of programming language.

Outline of IEC61131-3

What's Task?

Task: Handles priority and execution cycle of POU.

Software model of Task and POU



* POU = Program Organization Unit

Outline of IEC61131-3

HITACHI
Inspire the Next

Global variables / Local variables

- Global variables: can be used commonly in all POUs in a project
- Local variables: are valid in a POU where the variable is defined.

Example

Global variable

Name: Test, Data type: WORD

```
1  {attribute 'qualified_only'}
2  VAR_GLOBAL
3      Test: WORD;
4  END_VAR
5
```

Local variable

Name: Test, Data type: WORD

```
1  PROGRAM Test_PRG
2  VAR
3      Test: WORD;
4  END_VAR
```

Usage example in POU “Test_PRG”

GVL | Test_PRG

```
1  PROGRAM Test_PRG
2  VAR
3      Test: WORD;
4  END_VAR
5
```

```
1
2  GVL.Test:=100;   // Global
3
4  Test:=100;   // Local
```

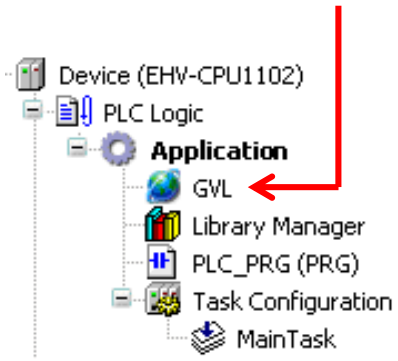
Advantage of IEC61131-3 programming

HITACHI
Inspire the Next

Global variables / Local variables

Local variables

Global variable list



PRG_A

```
VAR
  test : INT;
END_VAR
=======
Test := 1;
```

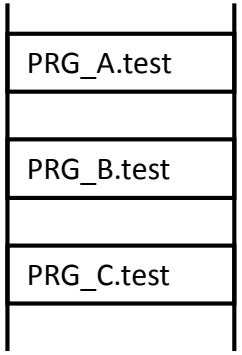
PRG_B

```
VAR
  test : INT;
END_VAR
=======
Test := 2;
```

PRG_C

```
VAR
  test : INT;
END_VAR
=======
Test := 3;
```

Data memory



Global variables

GVL

```
VAR_GLOBAL
  test : INT;
END_VAR
```

PRG_A

```
VAR
END_VAR
=======
Test;
```

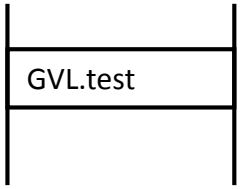
PRG_B

```
VAR
END_VAR
=======
Test;
```

PRG_C

```
VAR
END_VAR
=======
Test;
```

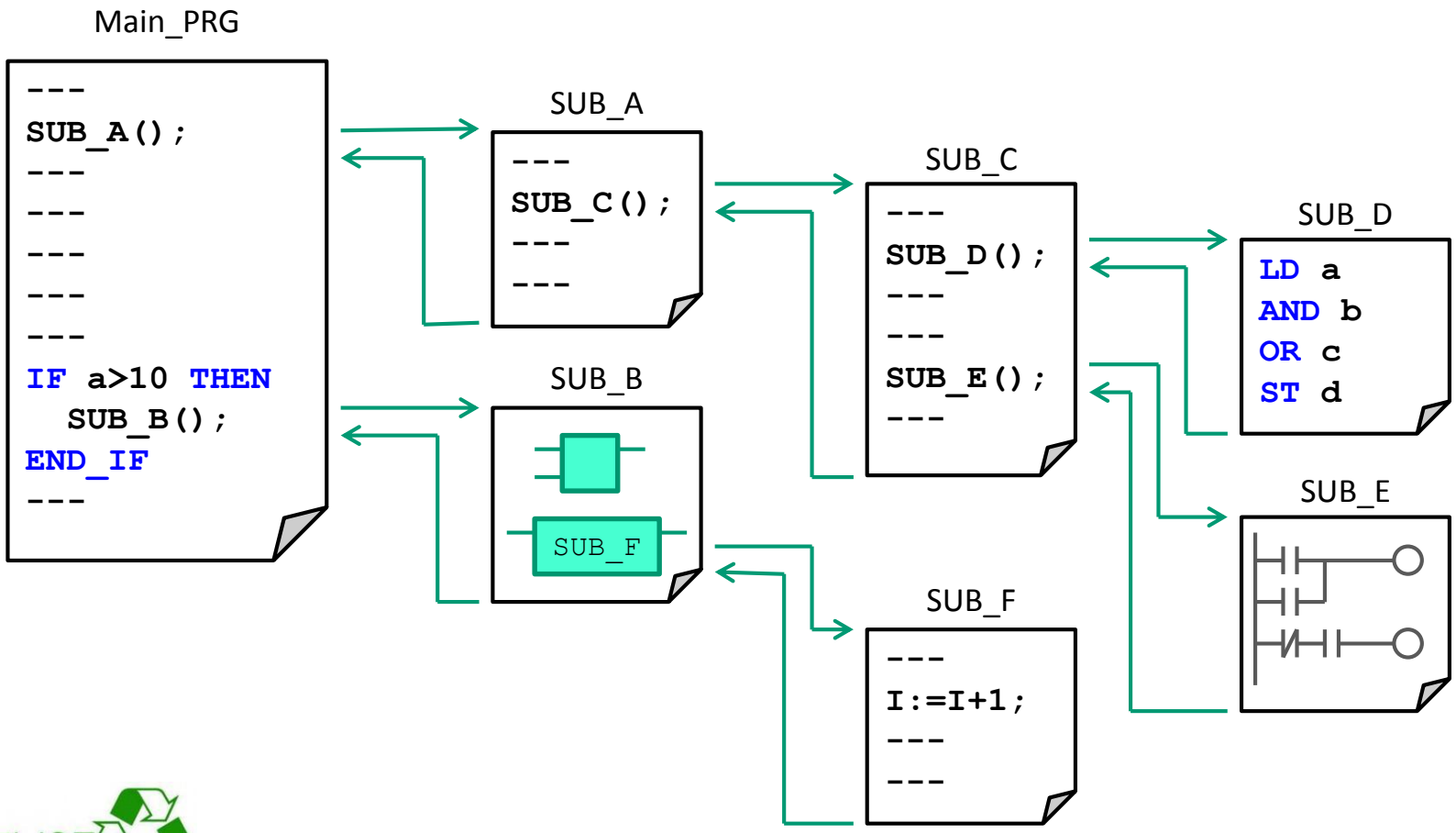
Data memory



- Effective programming among multiple developers
- High program reusability

HITACHI
Inspire the Next

Structured Program



REUSE
REDUCE
RECYCLE

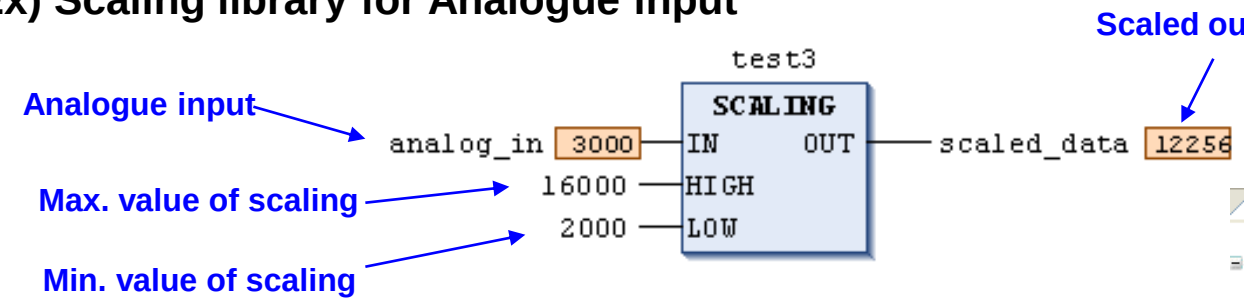
- Effective programming among multiple developers
- High program reusability

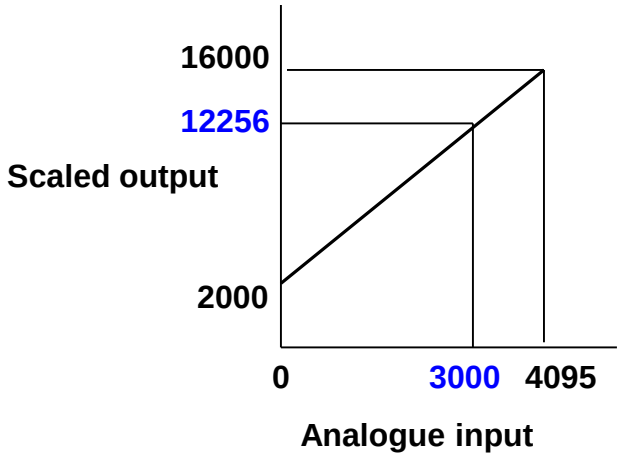
HITACHI
Inspire the Next

Library

If you make frequently used functions as library, you can easily call them from other programs. ➔ Program reusability

Ex) Scaling library for Analogue input





Program

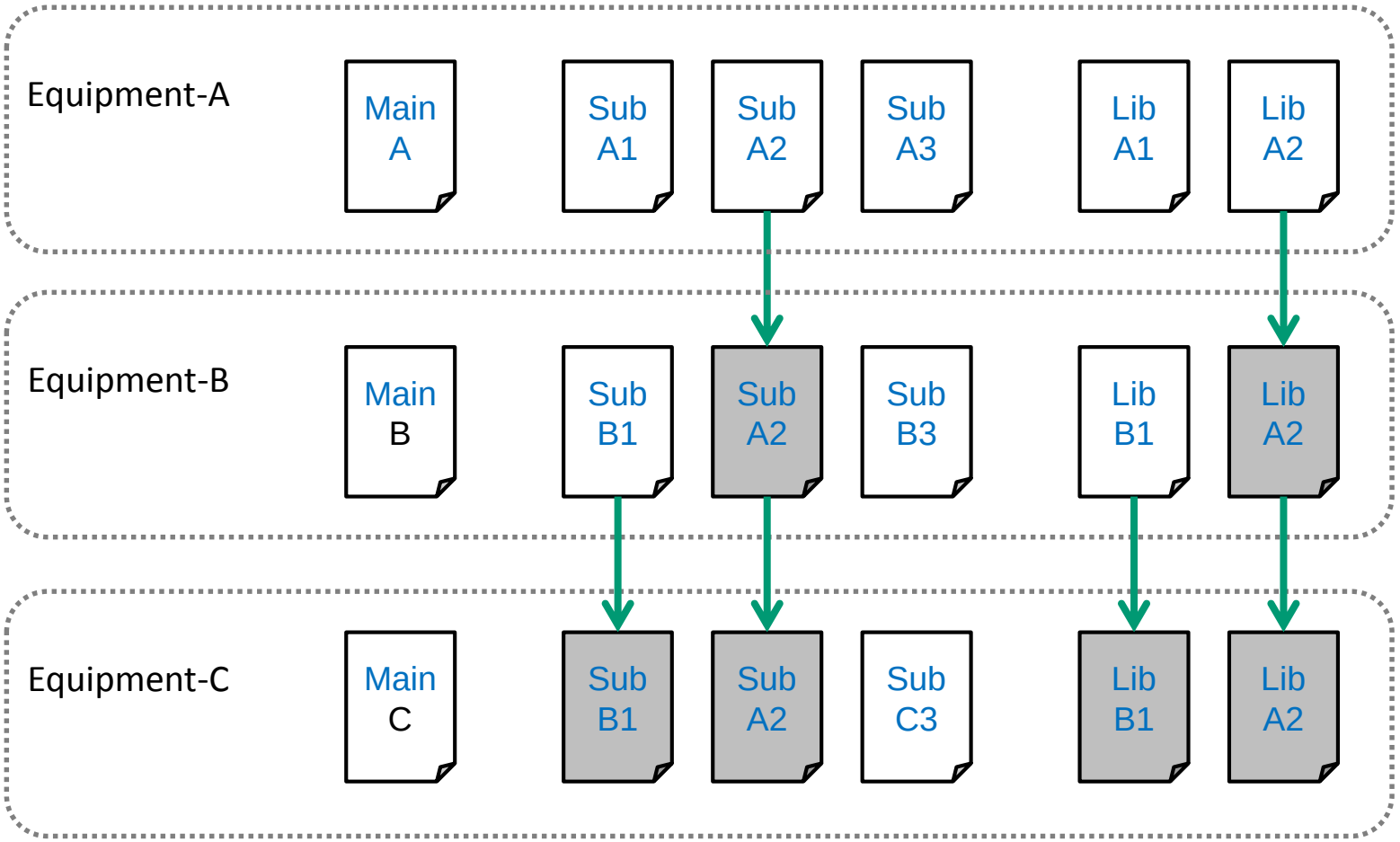
```
SCALING
1   FUNCTION_BLOCK SCALING
2   VAR_INPUT
3       IN: WORD;
4       HIGH: WORD;
5       LOW: WORD;
6   END_VAR
7   VAR_OUTPUT
8       OUT: WORD;
9   END_VAR
10  VAR
11      lIN: REAL;
12      lHIGH: REAL;
13      lLOW: REAL;
14      lOUT: REAL;
15  END_VAR
16
```

```
1
2   lIN:=WORD_TO_REAL(IN);
3   lHIGH:=WORD_TO_REAL(HIGH);
4   lLOW:=WORD_TO_REAL(LOW);
5   lOUT:= ((lHIGH - lLOW)/4095*lIN) + lLOW;
6   OUT:=REAL_TO_WORD(lOUT);
```

REUSE
REDUCE
RECYCLE

Advantage of IEC61131-3 programming

HITACHI
Inspire the Next

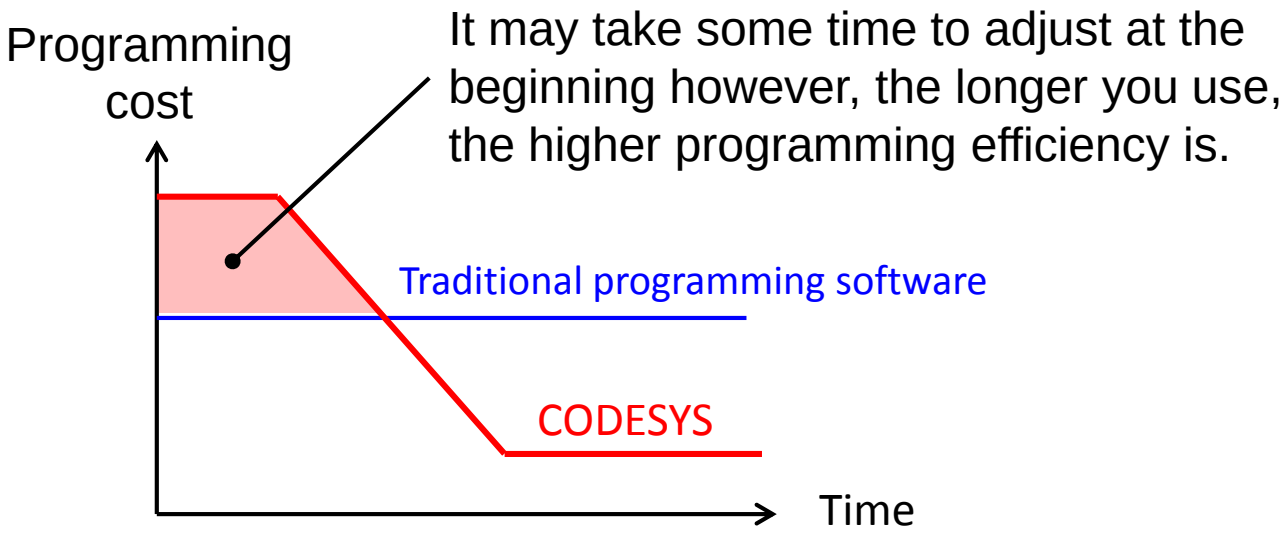


Features like Global variables, Structured programming, Library make Programming / debugging efficiency increased as you make programs.

Advantage of IEC61131-3 programming

High efficiency programming

➔ Cost reduction for customers



HITACHI
Inspire the Next

Features of CODESYS

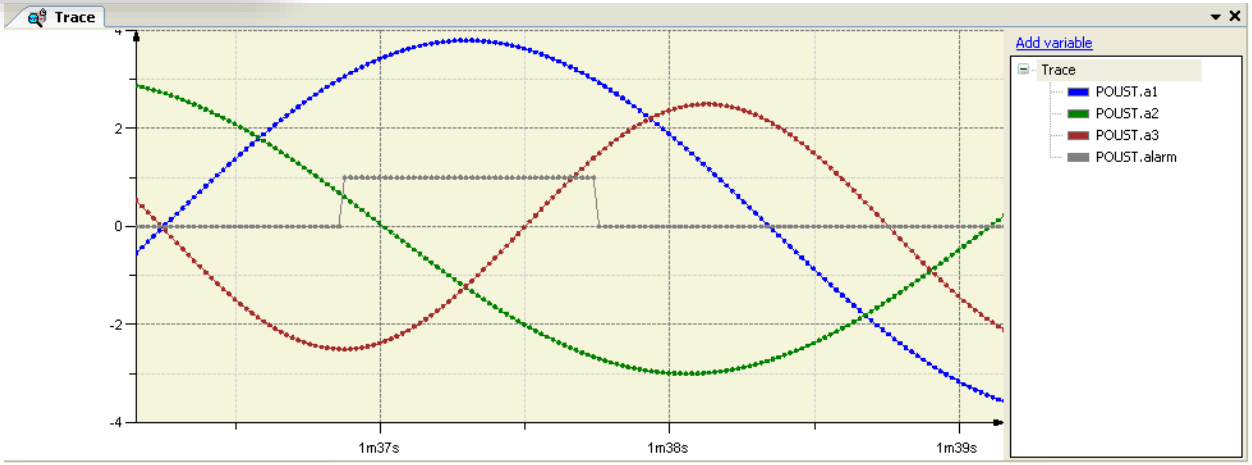
HITACHI
Inspire the Next

Visualization (Simple SCADA) and Trace

Visualization: Displayed on CODESYS

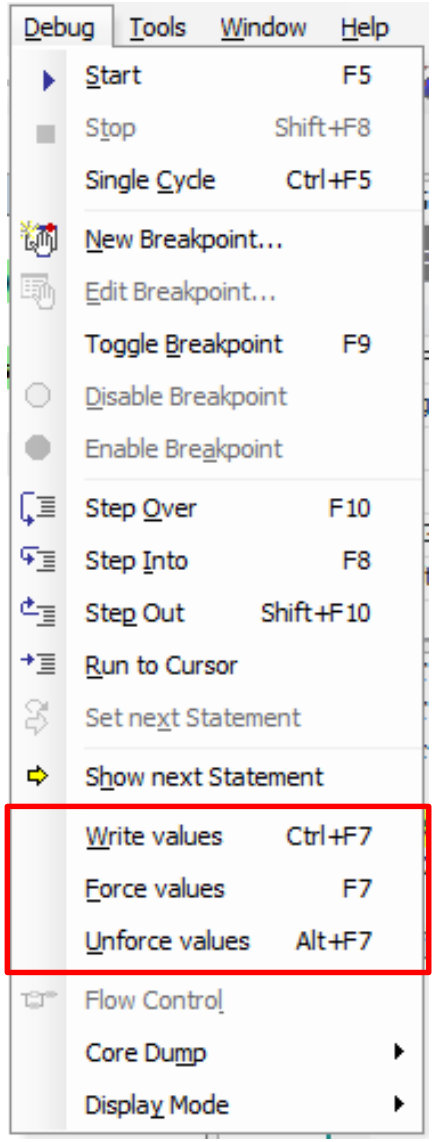
Web Visualization: Displayed on web browser

Trace



HITACHI
Inspire the Next

Write value / Force value



Write value (Set a prepared value to a variable)

```
test 0 := var1 0<3> * 2;
```

[Ctrl] + [F7]

```
test 6 := var1 3 * 2;
```

Force value (Fix value of a variable regardless of program)

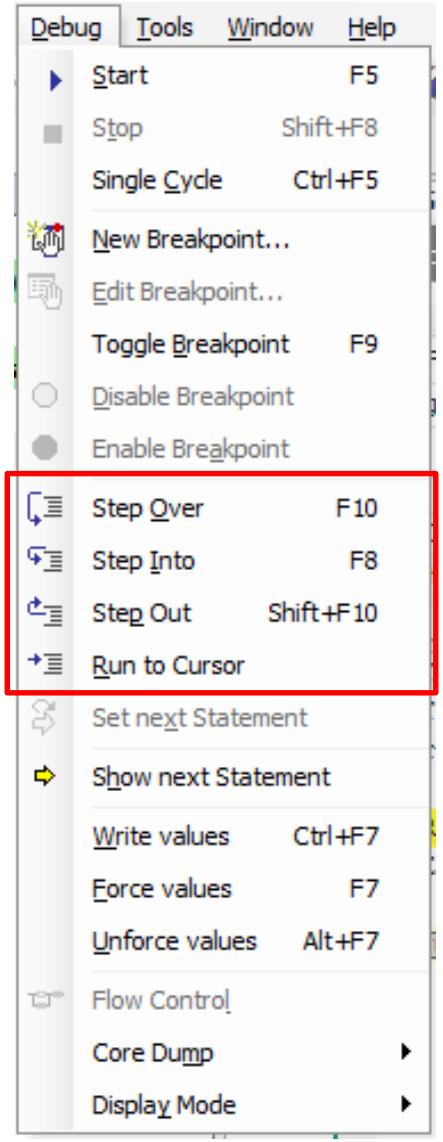
```
test 6<100> := var1 3 * 2;
```

[F7]

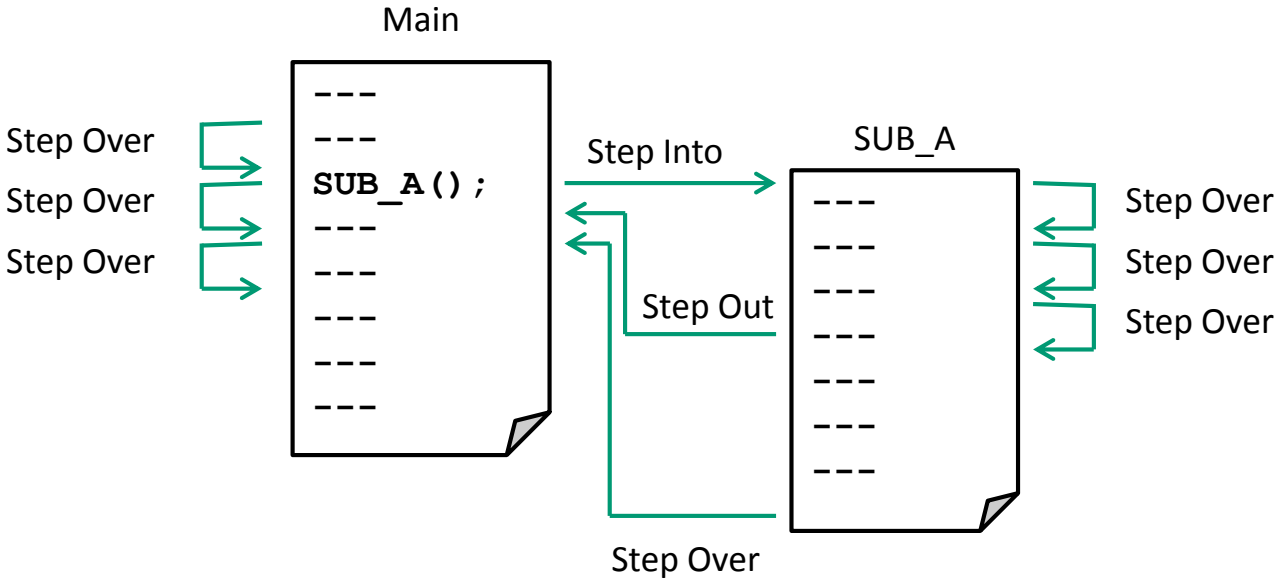
```
test F 100 := var1 3 * 2;
```

HITACHI
Inspire the Next

Single scan, Break point, step execution

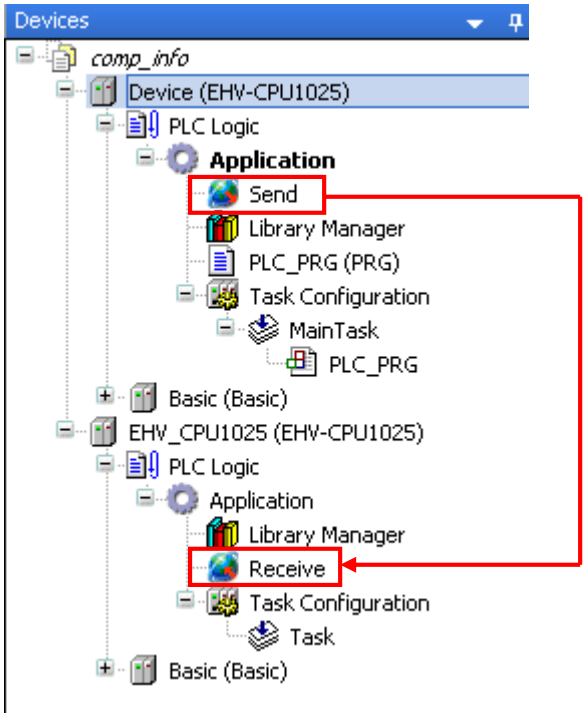


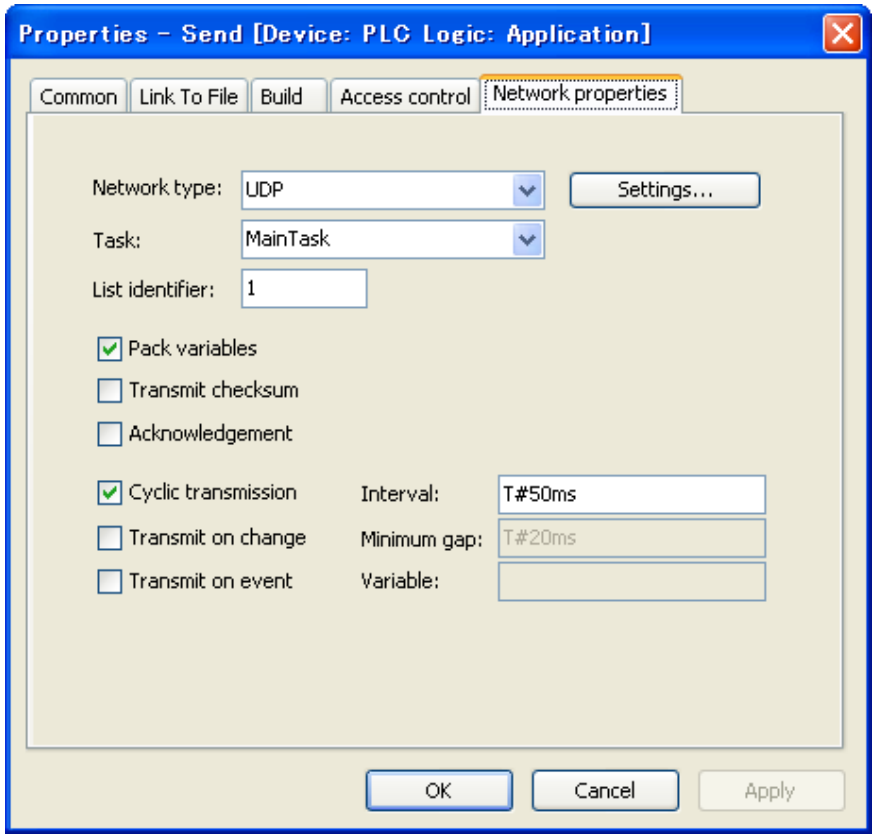

```
11  IF test1  0  >6 THEN
12      S2();
13      test1  0  :=0;
14  END_IF;
15
16  test3  1266  :=test3  1266  +1;
17
```



HITACHI
Inspire the Next

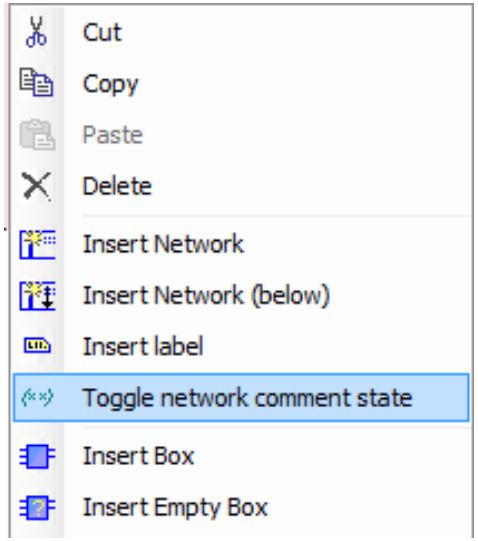
Global network variable: Automatic UDP communication



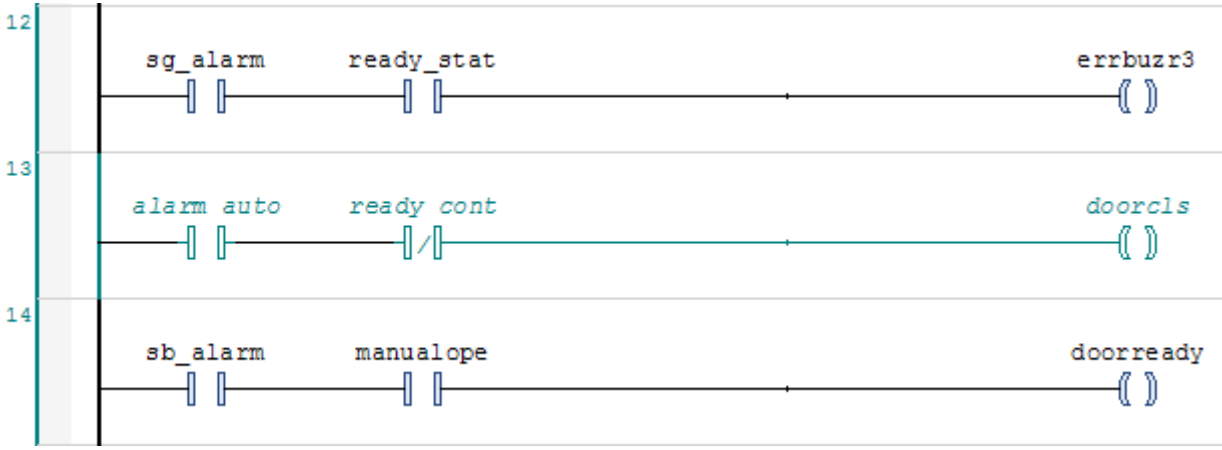


HITACHI
Inspire the Next

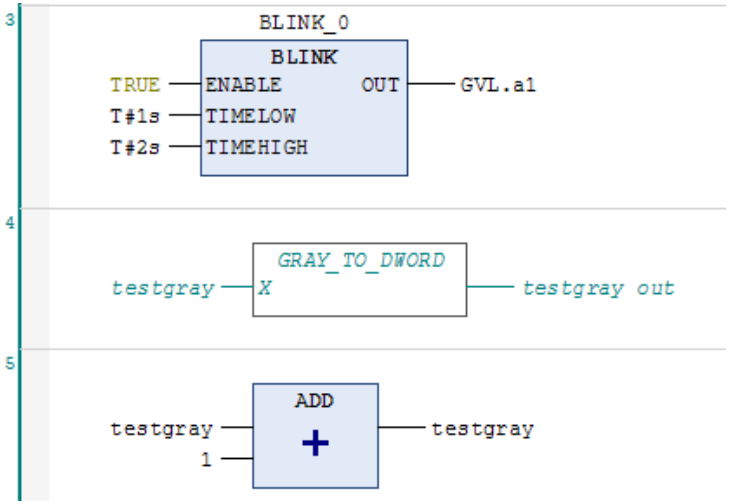
Comment out (Deactivation of instructions)



LD

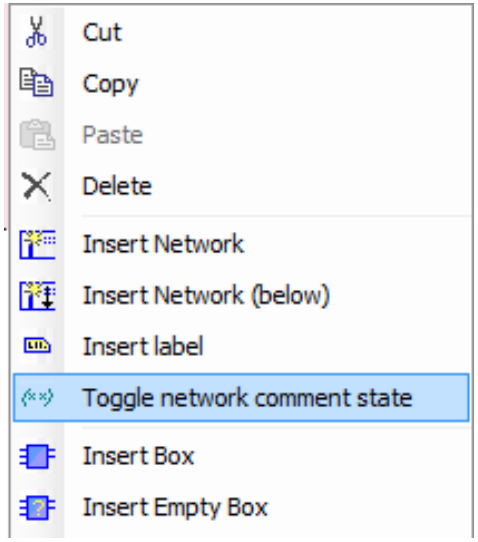


FBD



HITACHI
Inspire the Next

Comment out (Deactivation of instructions)



ST

Single line: //

```
12  // ********** TRACE data **********
13  trace_data1:=SIN(k1);
14  trace_data2:=1.5*SIN(k1*3+1);
15  //trace_data3:=2*SIN(k1+2);
16  k1:=k1+0.01;
17  trace_x1:=trace_x1+1;
18  trace_data4:=trace_x1.6;
19
```

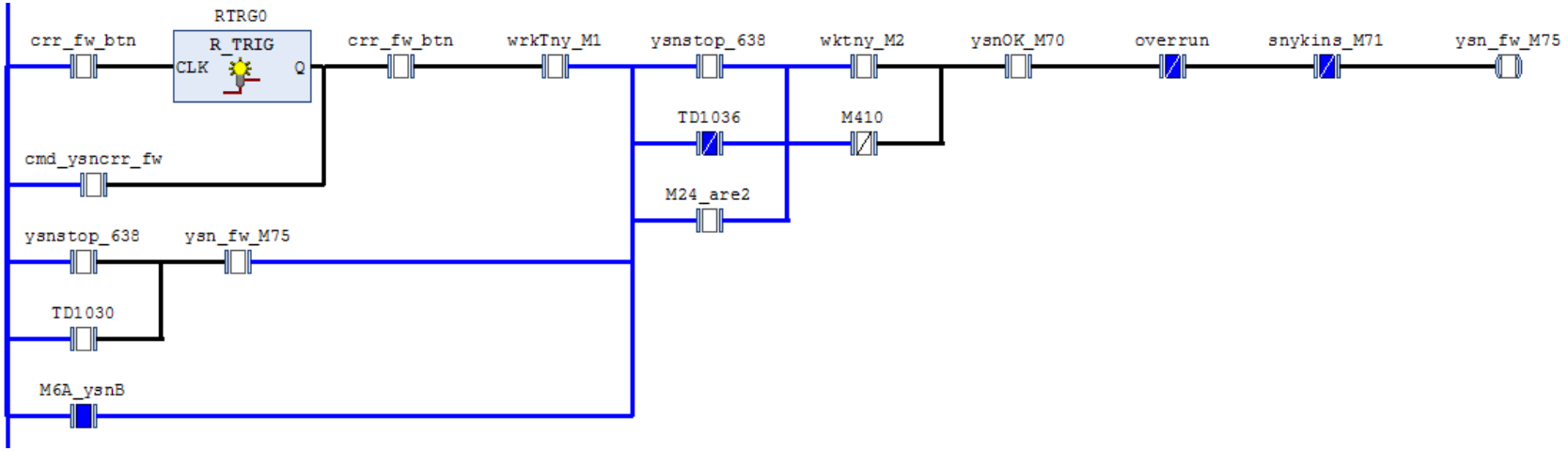
Multiple lines: (* *)

```
12  // ********** TRACE data **********
13  trace_data1:=SIN(k1);
14  trace_data2:=1.5*SIN(k1*3+1);
15  trace_data3:=2*SIN(k1+2);
16  (*
17  k1:=k1+0.01;
18  trace_x1:=trace_x1+1;
19  trace_data4:=trace_x1.6;
20  *)
```

Features of CODESYS

HITACHI
Inspire the Next

Monitor of Ladder: Displayed at blue line for conducted circuit



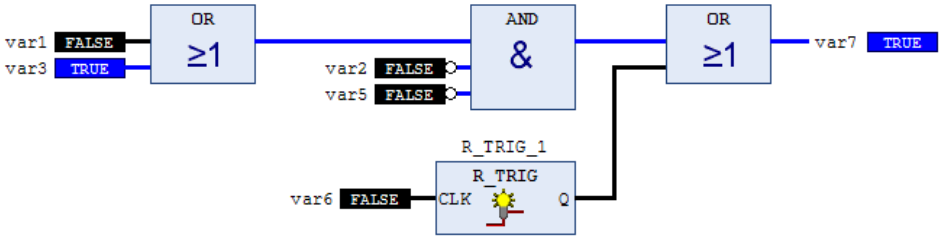
Flow control: Currently executed instructions are displayed in green.

```
IF x 11 > 100 THEN
    x 12 := 0;
ELSIF x 11 > 50 THEN
    x 12 := x 12 + 2;
    inst1(a 111 := 111, b 222 := 222, c 333 := 333);
    inst1f(a 111 := 111, b 222 := 222, c 333 := 333);
ELSE
    x 12 := x 11 + 1;
    inst2(a 432 := 432, b 543 := 543, c 654 := 654);
    inst2f(a 432 := 432, b 543 := 543, c 654 := 654);
END_IF
```

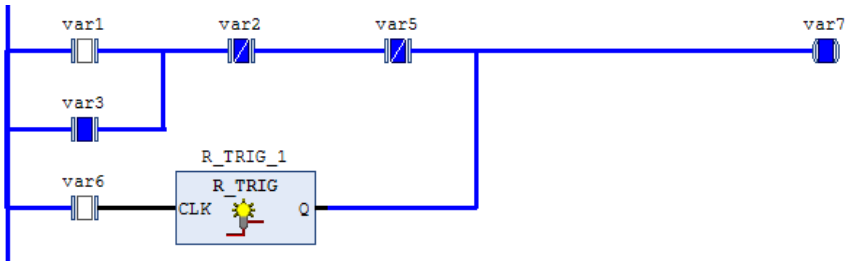
HITACHI
Inspire the Next

Switchable among FBD/LD/IL at a shortcut key

FBD : [Ctrl] + [1]



LD : [Ctrl] + [2]

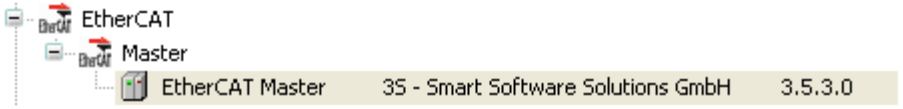


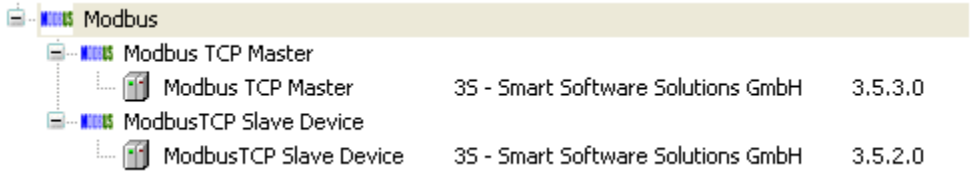
IL : [Ctrl] + [3]

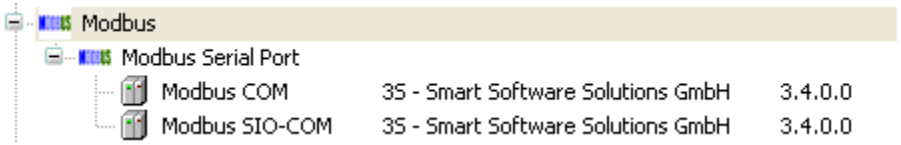
```
1  CAL     R_TRIG_1(
     CLK:= var6)          FALSE
   LD      var1           FALSE
   OR      var3           TRUE
   ANDN    var2           FALSE
   ANDN    var5           FALSE
   OR      R_TRIG_1.Q     FALSE
   ST      var7           TRUE
```

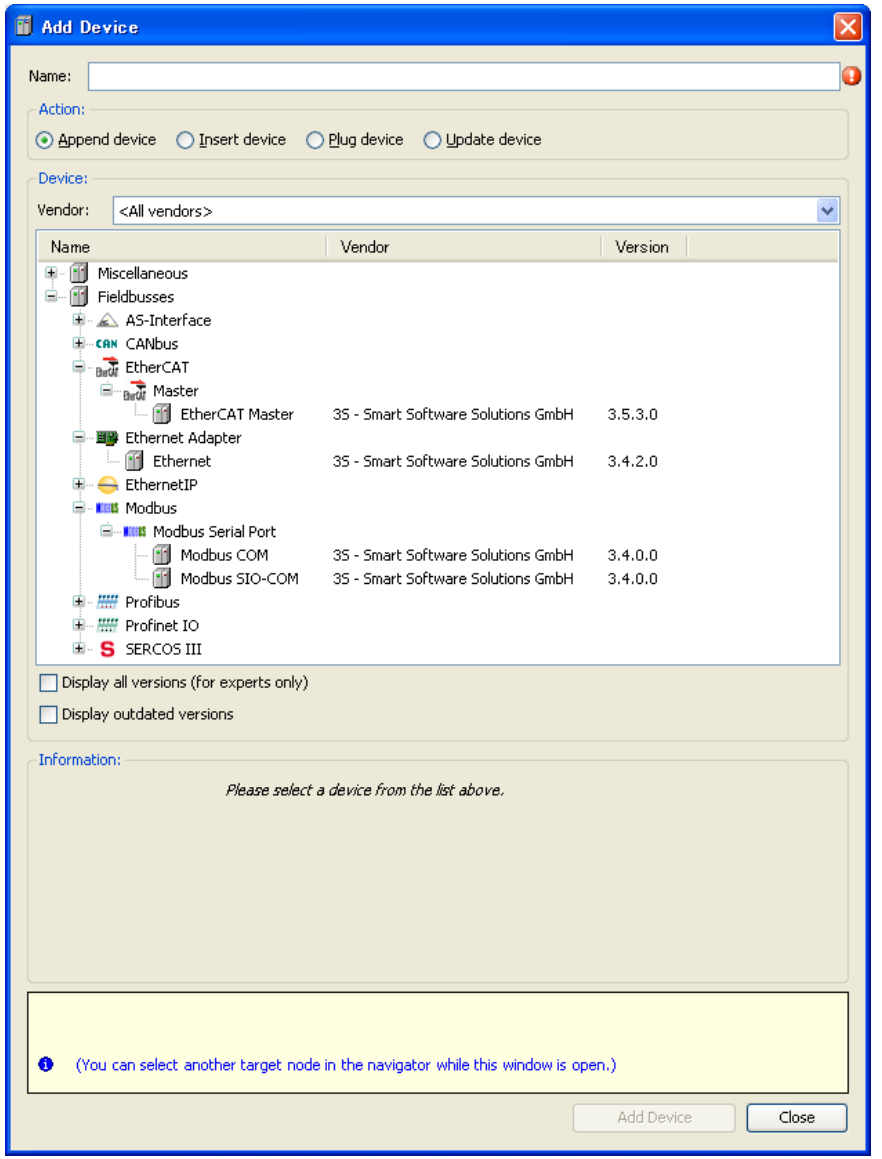
Features: Fieldbus

- EtherCAT master
- Modbus-TCP master
- Modbus-TCP slave
- Modbus-RTU master





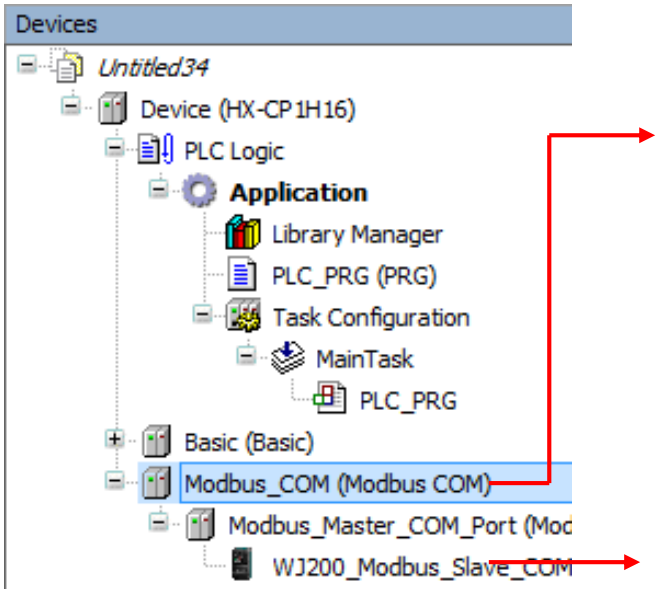


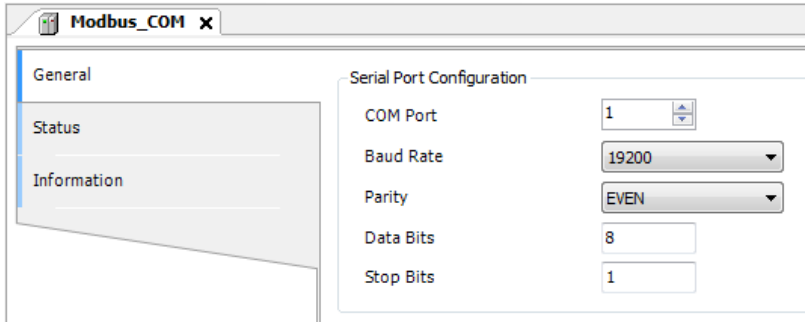


HITACHI
Inspire the Next

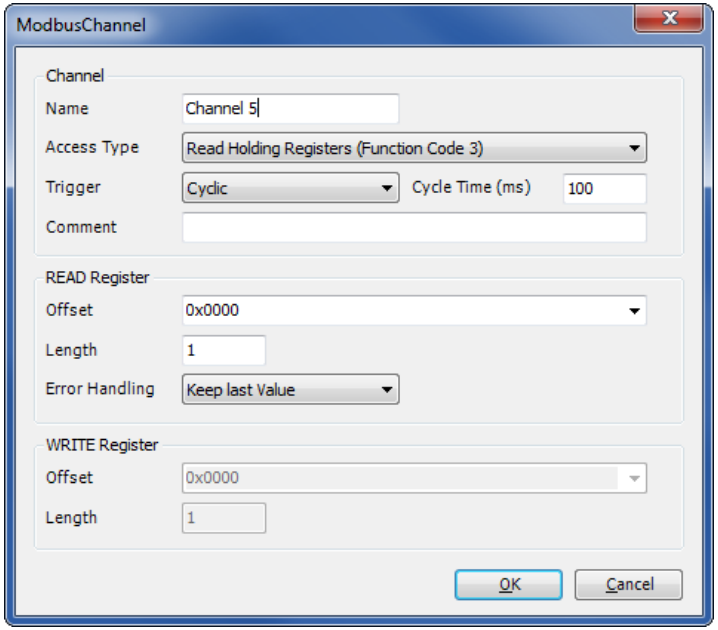
Programless Modbus communication

Ex) Modbus-RTU Master





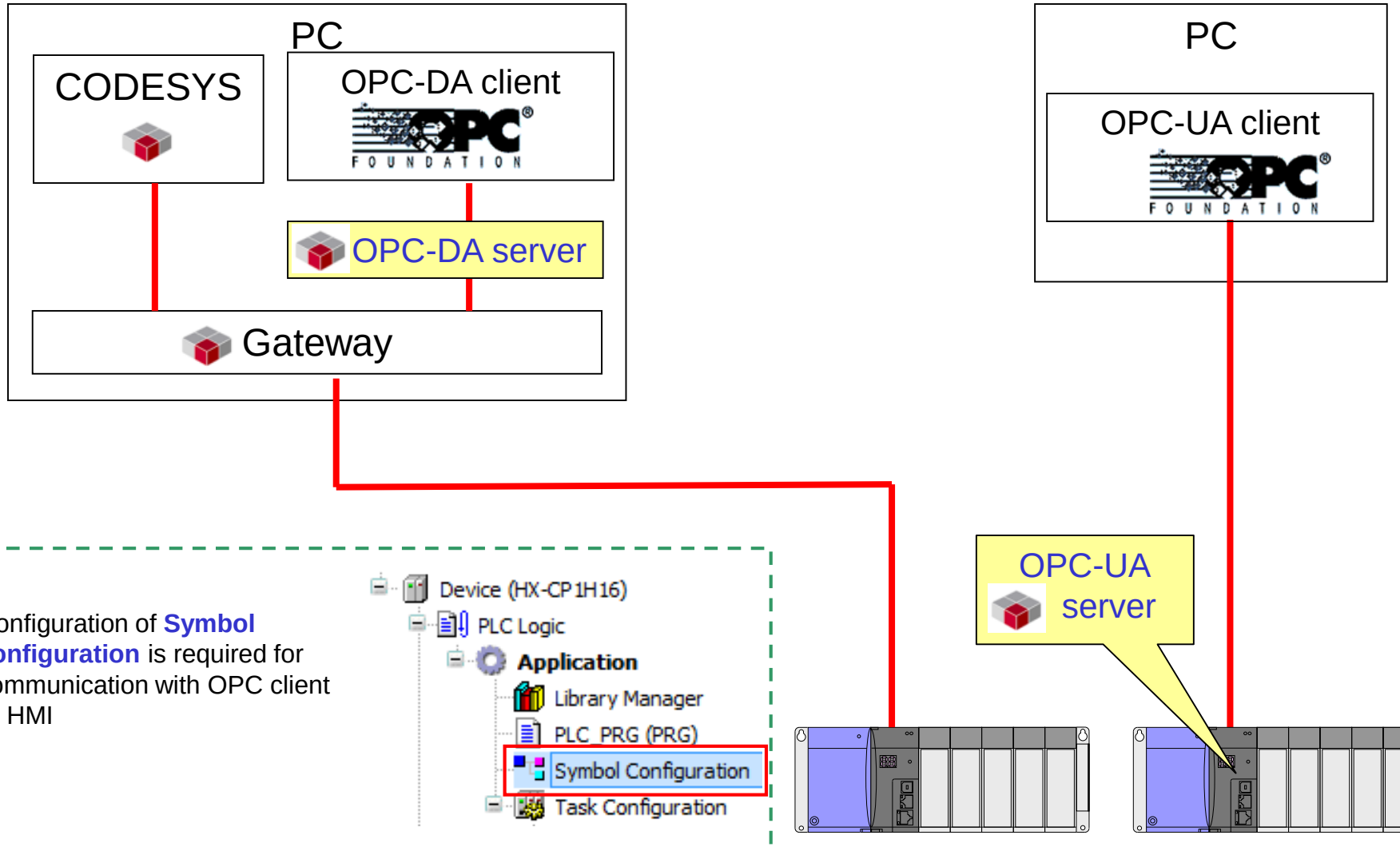
Communication setting



Read Holding register 0000 at 100ms cycle

HITACHI
Inspire the Next

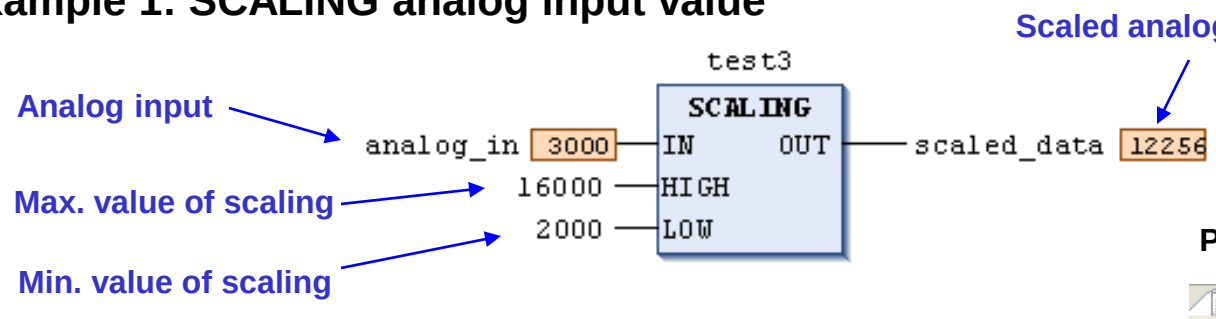
OPC-DA・OPC-UA communication

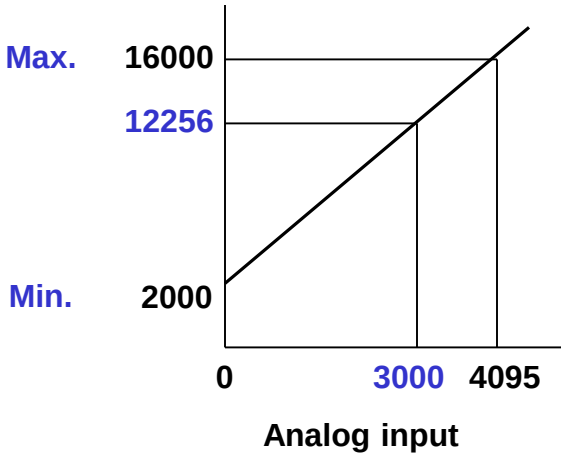


Features: Library

Frequently used program or function can be registered as library, which can be called from other projects.

Example 1: SCALING analog input value





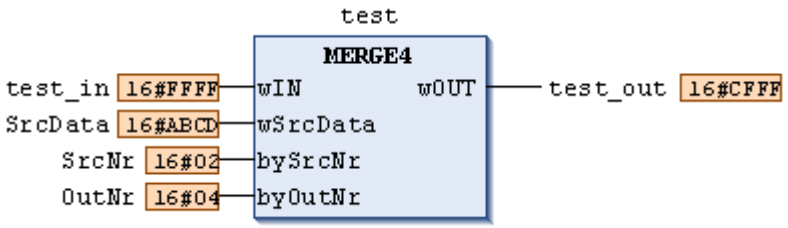
Program in the FB (possible to hide)

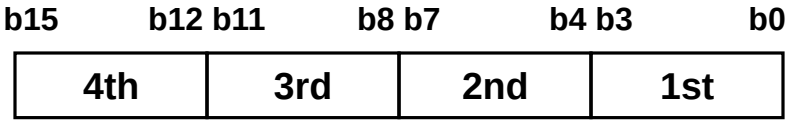
SCALING

```
1   FUNCTION_BLOCK SCALING
2   VAR_INPUT
3       IN: WORD;
4       HIGH: WORD;
5       LOW: WORD;
6   END_VAR
7   VAR_OUTPUT
8       OUT: WORD;
9   END_VAR
10  VAR
11      lIN: REAL;
12      lHIGH: REAL;
13      lLOW: REAL;
14      lOUT: REAL;
15  END_VAR
16
```

```
1
2   lIN:=WORD_TO_REAL(IN);
3   lHIGH:=WORD_TO_REAL(HIGH);
4   lLOW:=WORD_TO_REAL(LOW);
5   lOUT:= ((lHIGH - lLOW)/4095*lIN) + lLOW;
6   OUT:=REAL_TO_WORD(lOUT);
```

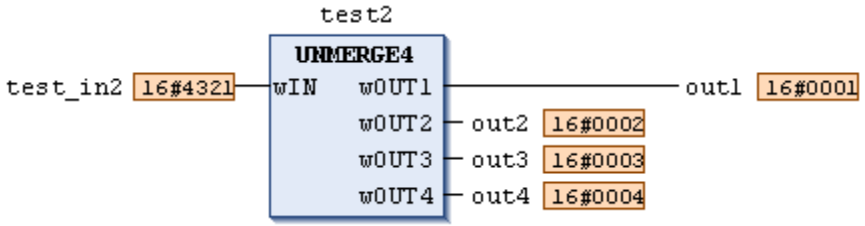
Example 2: MERGE4





2nd 4bit data (“C”) of SrcData is written to 4th 4bit data of “test_in”

Example 3: UNMERGE4



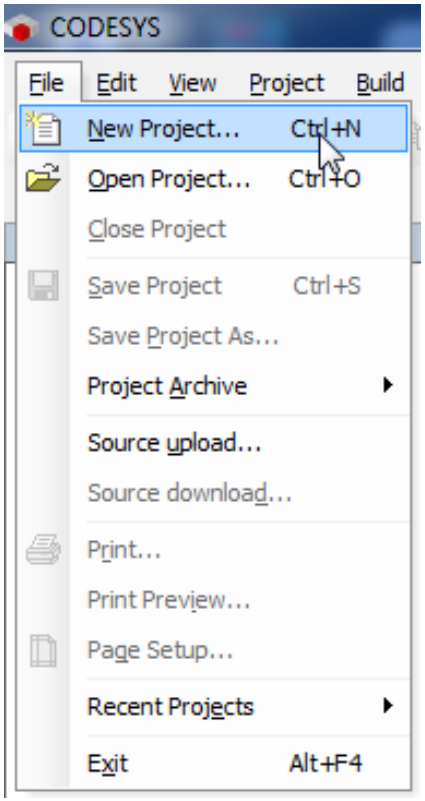
HITACHI
Inspire the Next

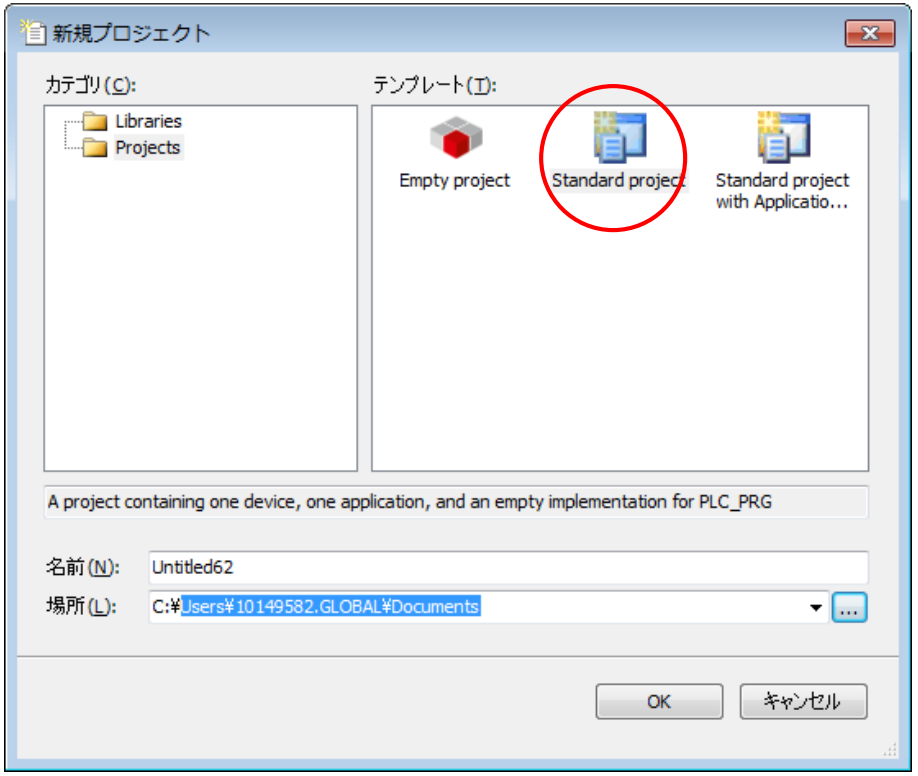
Programming

Preparation for Programming

1. Select [File] – [New Project] menu
2. Select [Standard project]

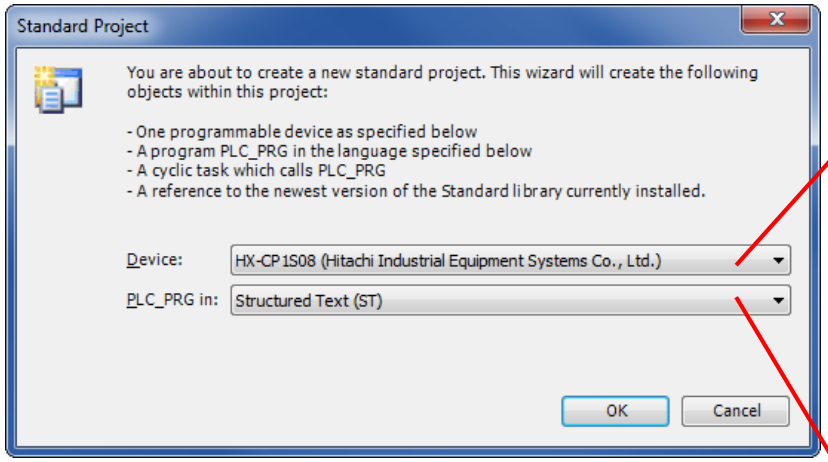
   (Empty project: requires you adding all components manually)

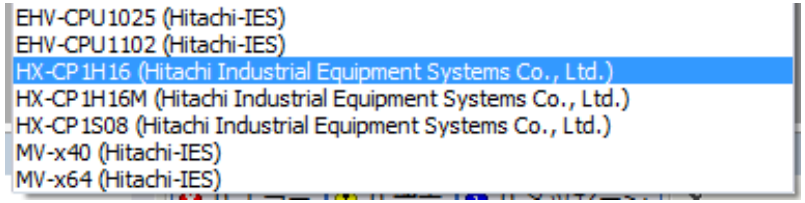


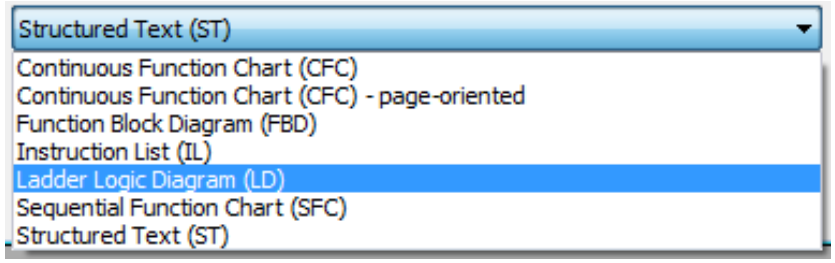


Preparation for Programming

1. Select CPU type (This setting can be changed later)
2. Select programming language of default POU







CPU type

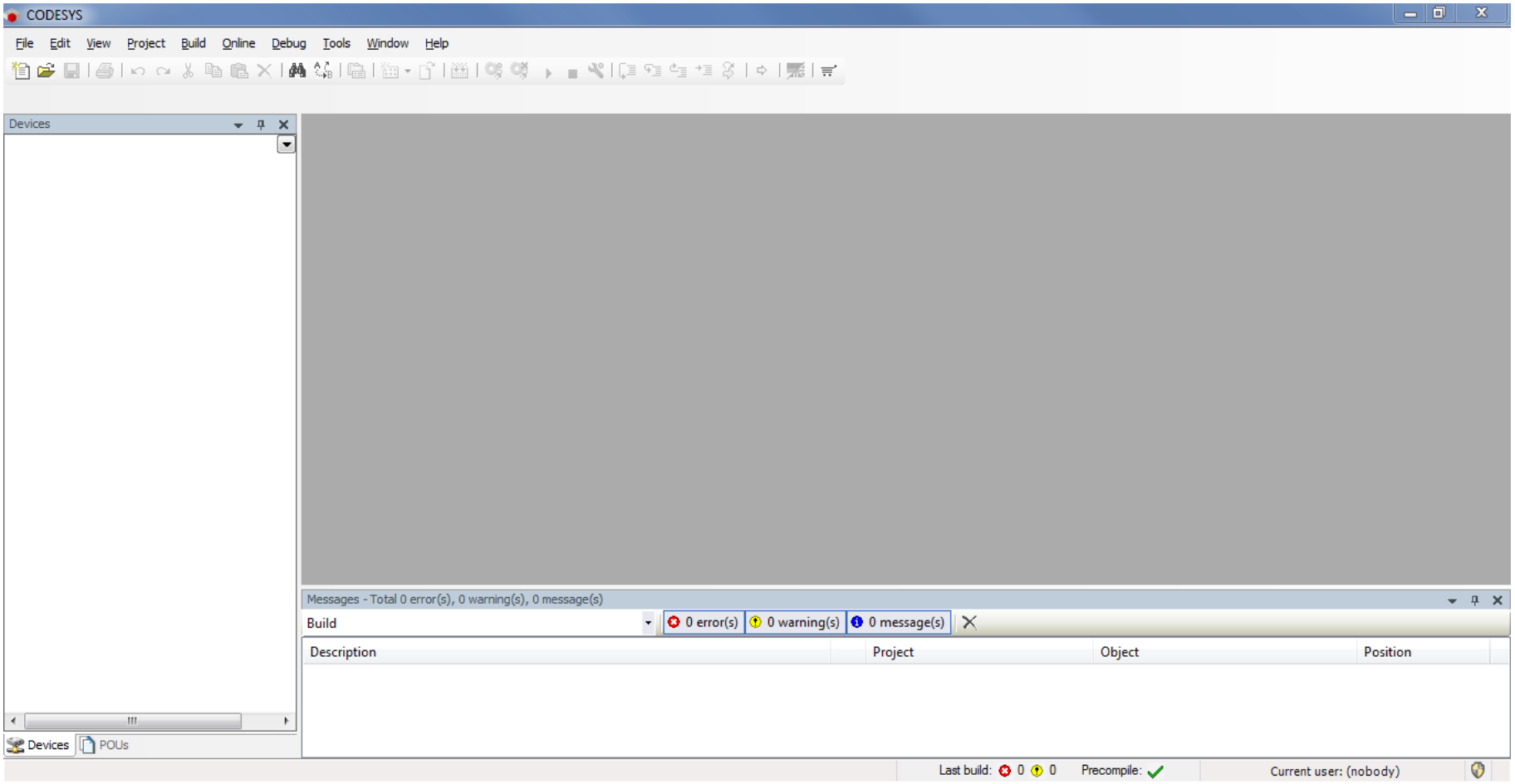
- HX-CP1S08
- HX-CP1H16
- EHV-CPU1102
- EHV-CPU1025
- CODESYS Control Win V3
  (Soft PLC operates on PC)

Programming language

- CFC (Continuous Function Chart)
- FBD (Function Block Diagram)
- IL (Instruction List)
- LD (Ladder Logic Diagram)
- SFC (Sequential Function Chart)
- ST (Structured Text: Similar to C)

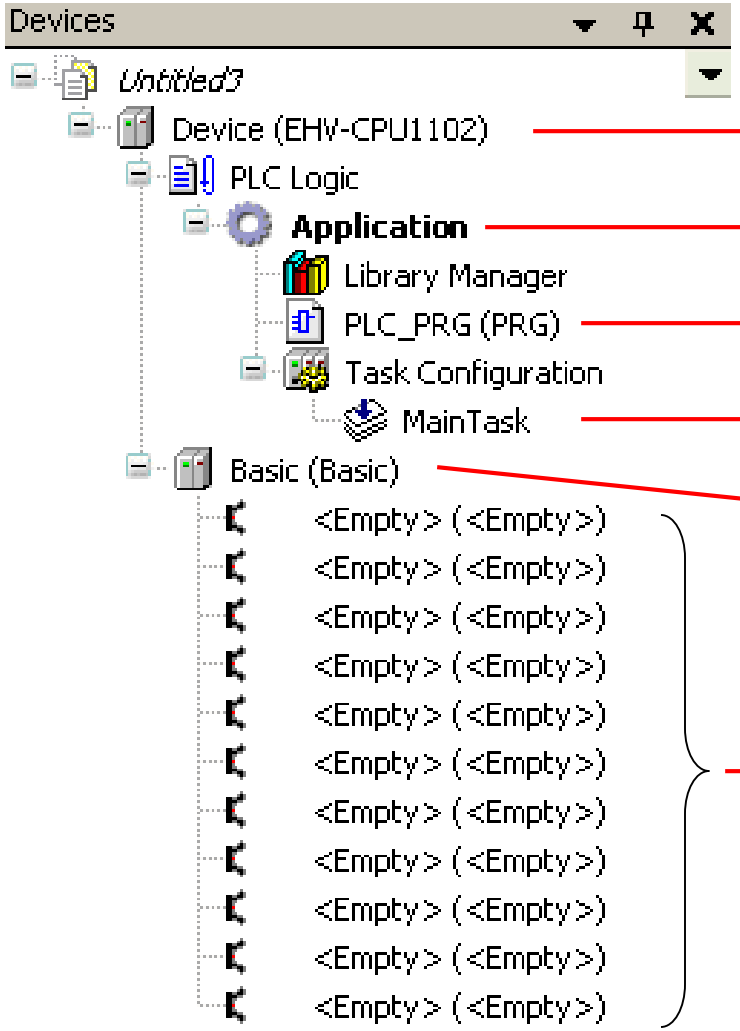
HITACHI
Inspire the Next

Default window layout



Before programming

Project Tree



Device = CPU

Application (set of project components)

POU* (like program sheet)

Task (executing POU)

Basic base

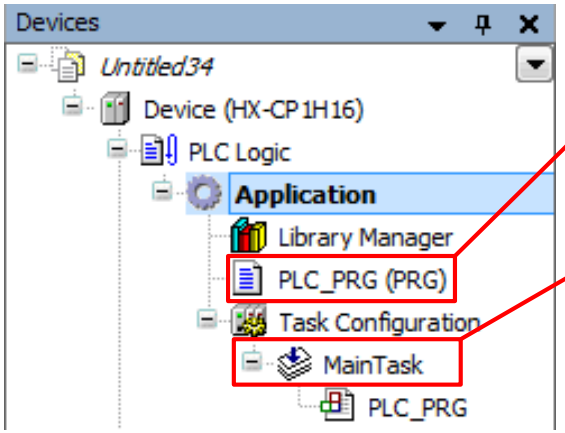
Slots for I/O modules (fixed as 11 slots)

* POU = Program Organization Unit

Preparation for Programming

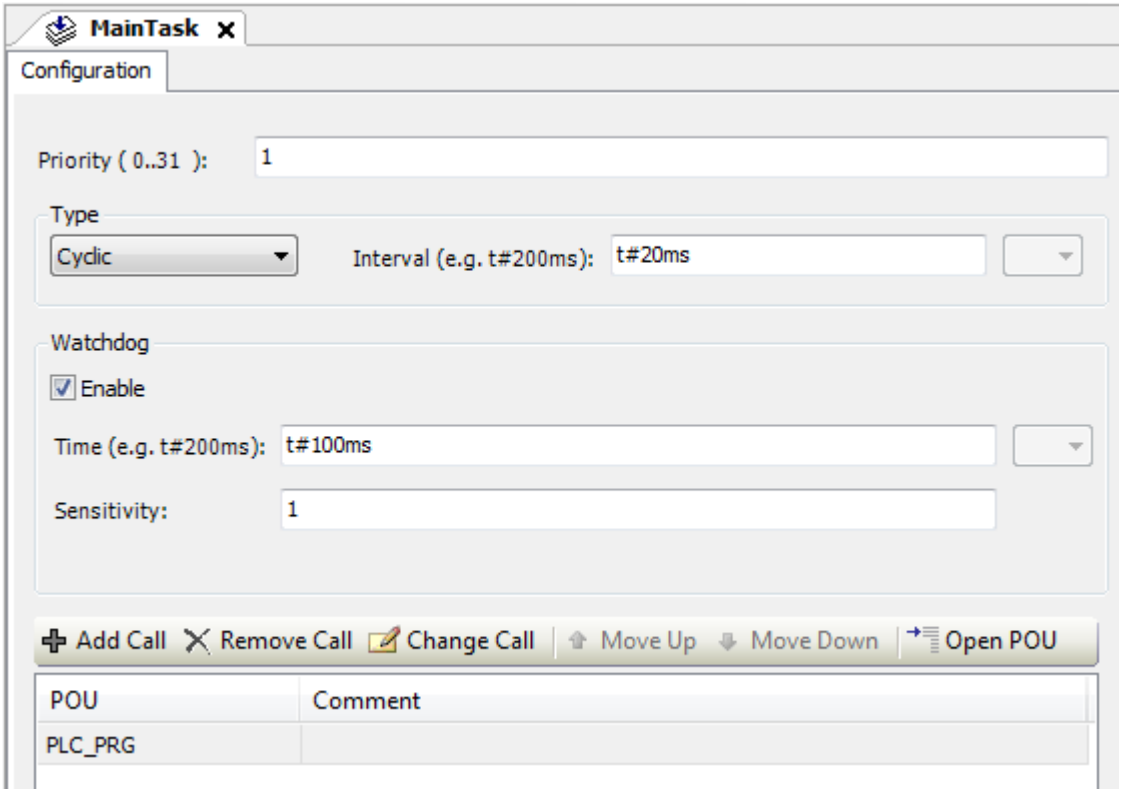
HITACHI
Inspire the Next

Project tree



POU: User program

Task: Handles priority and execution cycle of POU



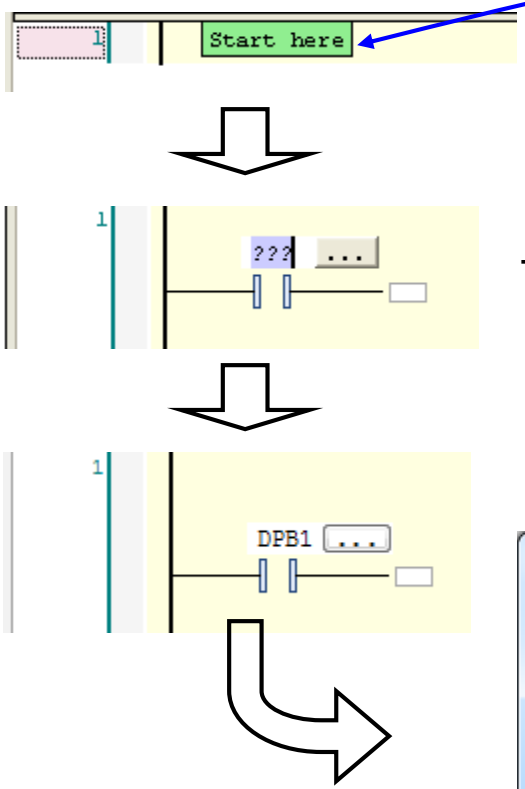
Maximum number of Tasks: 49

Max. number of tasks: 49
Max. number of cyclic tasks: 32
Max. number of freewheeling tasks: 1
Max. number of event tasks: 8
Max. number of status tasks: 8

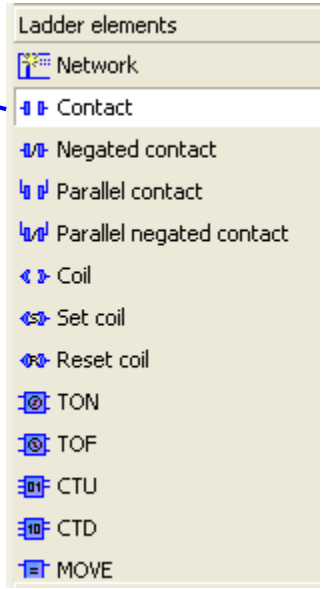
Brief Program (with use of internal I/O only)

HITACHI
Inspire the Next

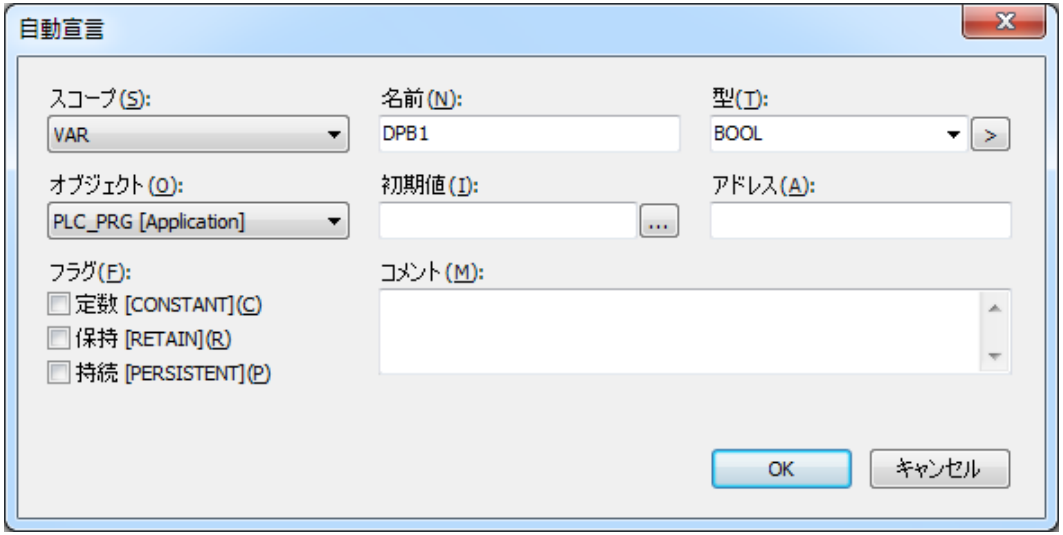
Drug “Contact” to **Start here**



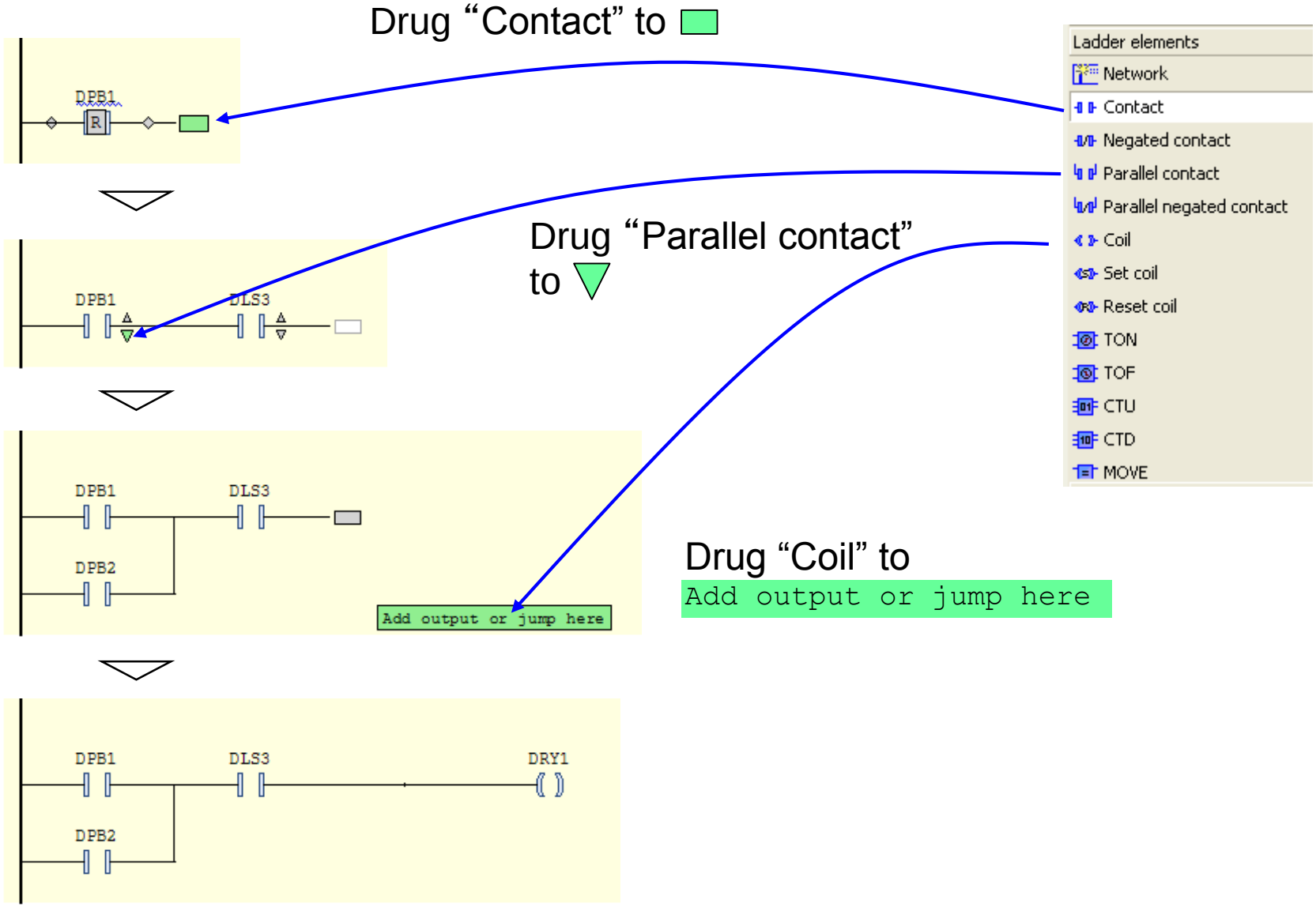
Type optional variable name



Declare window automatically appears



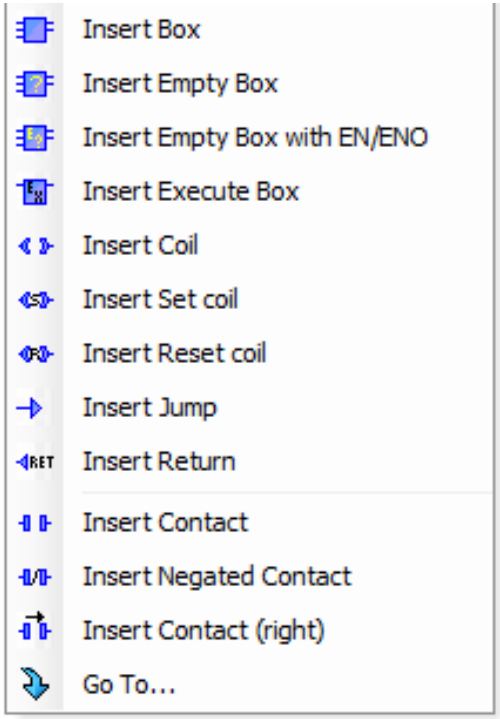
Brief Program (with use of internal I/O only)



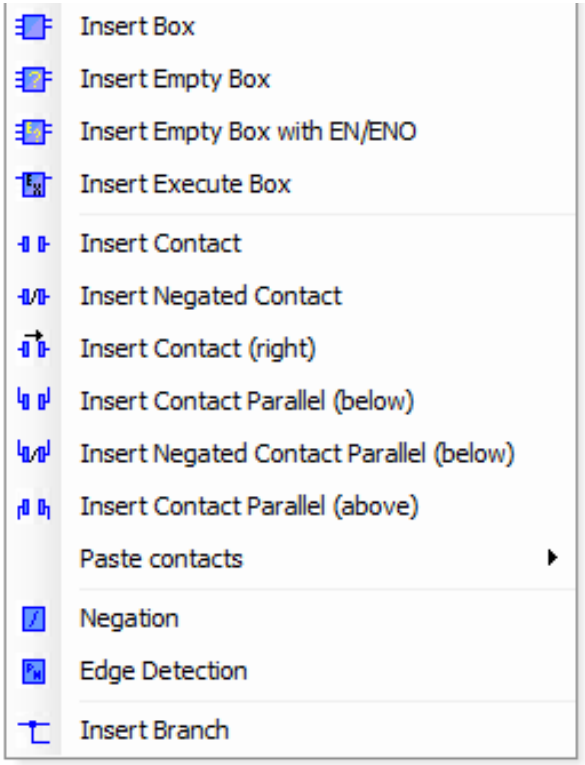
Brief Program (with use of internal I/O only)

A symbol can also be inserted by right-clicking or shortcut keys.

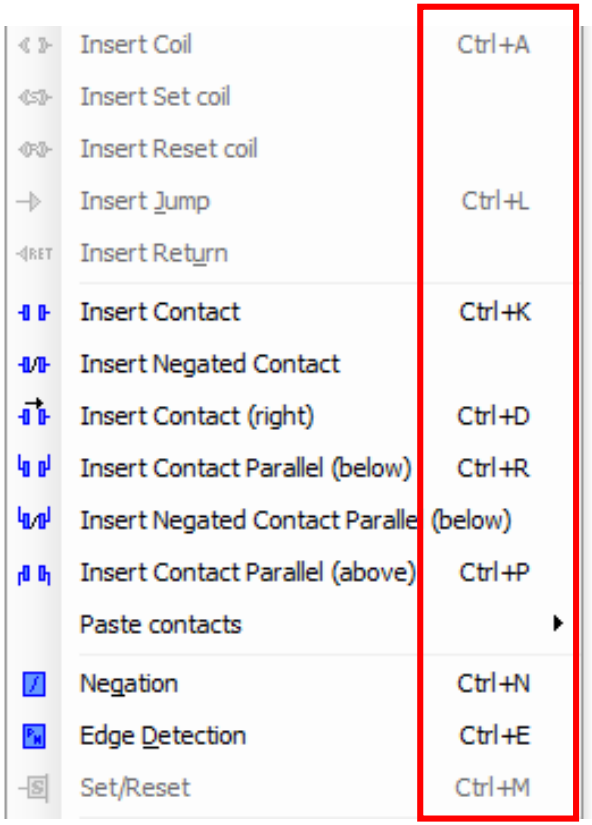
Right-clicking
in blank space of circuit.



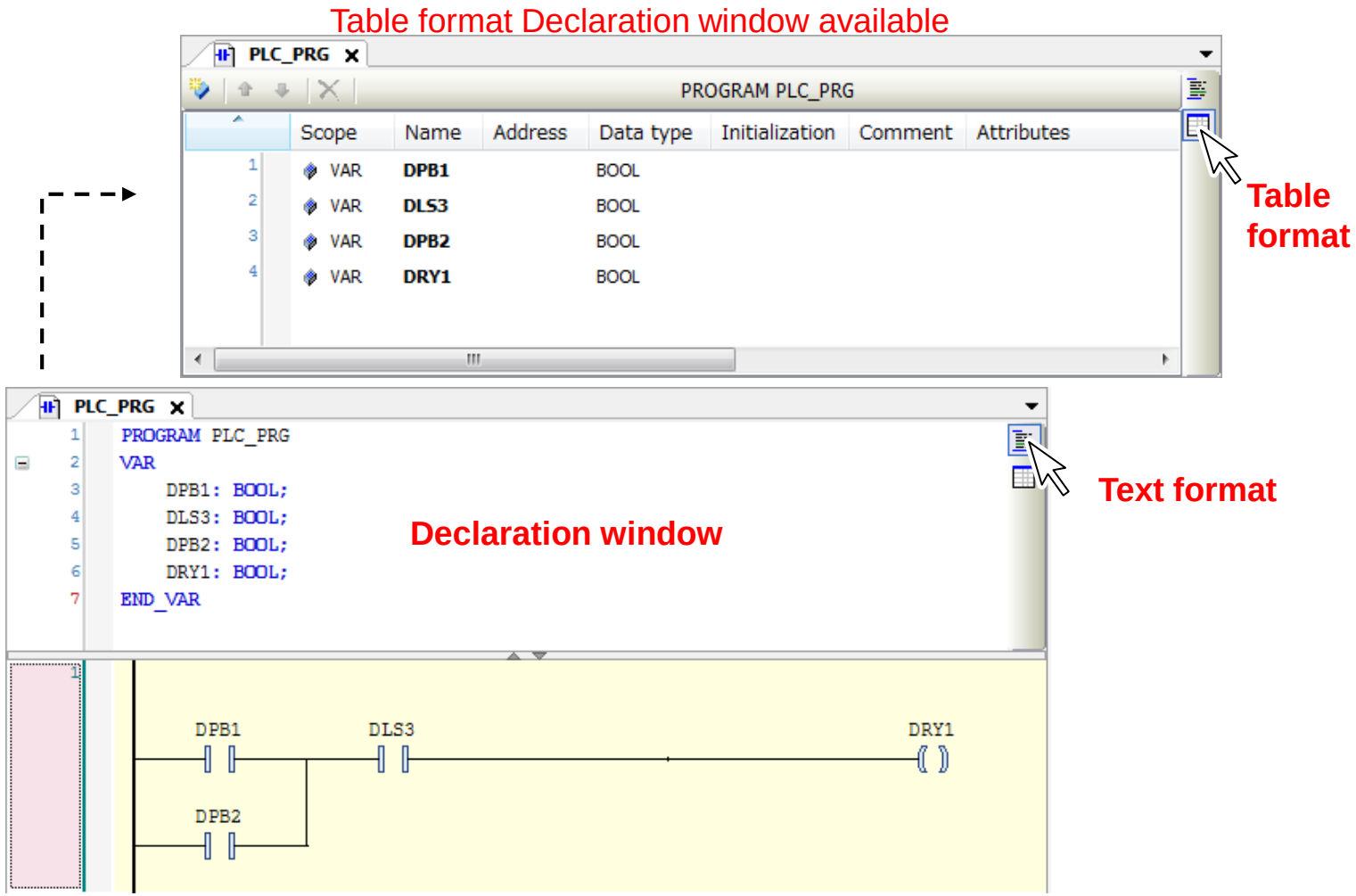
Right-clicking
on a Contact symbol.



A example of shortcut keys



Brief Program (using internal I/O only)



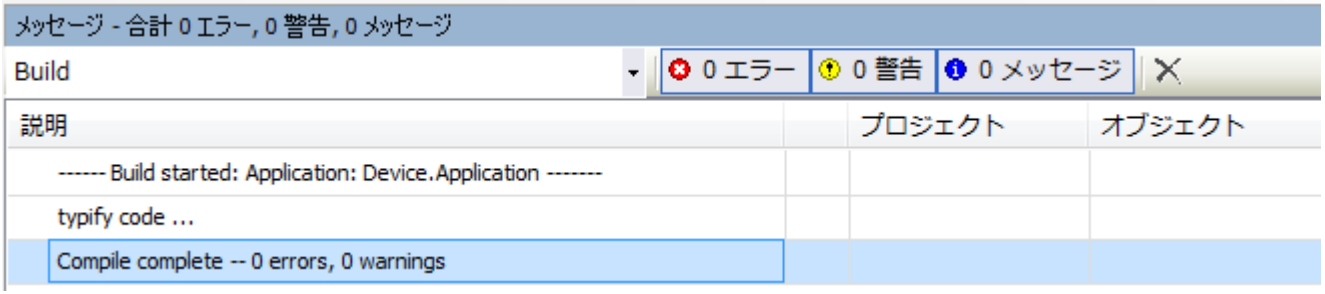
(Note) The variables declared in each POU are dealt as local variables.

Brief Program (using internal I/O only)

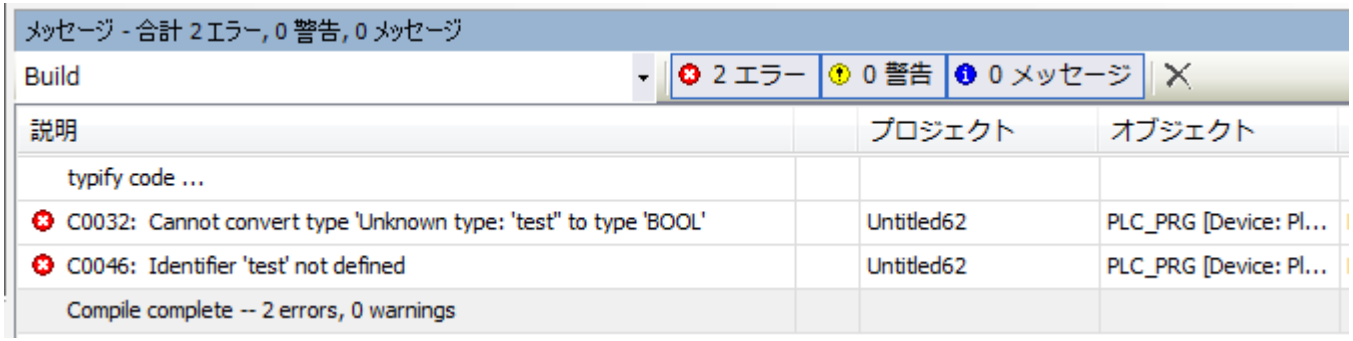
HITACHI
Inspire the Next

Compile the project by clicking Build after completing programming
(icon or [F11] or menu [Build] - [Build])

OK if no error is detected



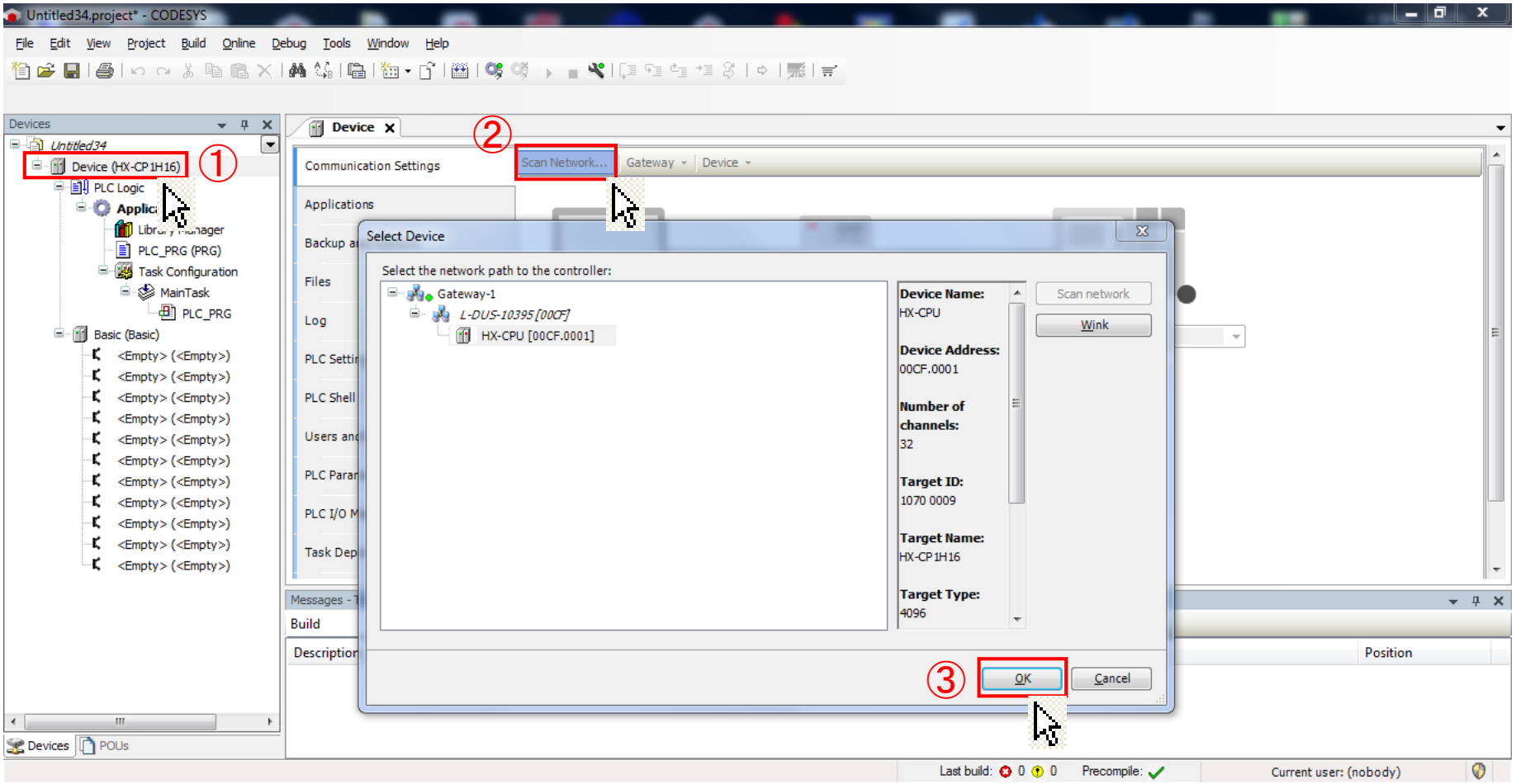
Modify program if any error is detected



Online

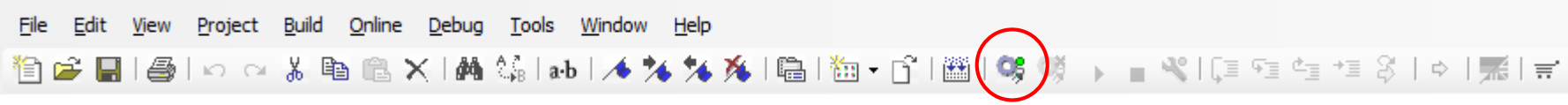
① Double-click “Device” ② Click [Scan network…]

③ Click [OK] on “Select Device” window after choosing the target device

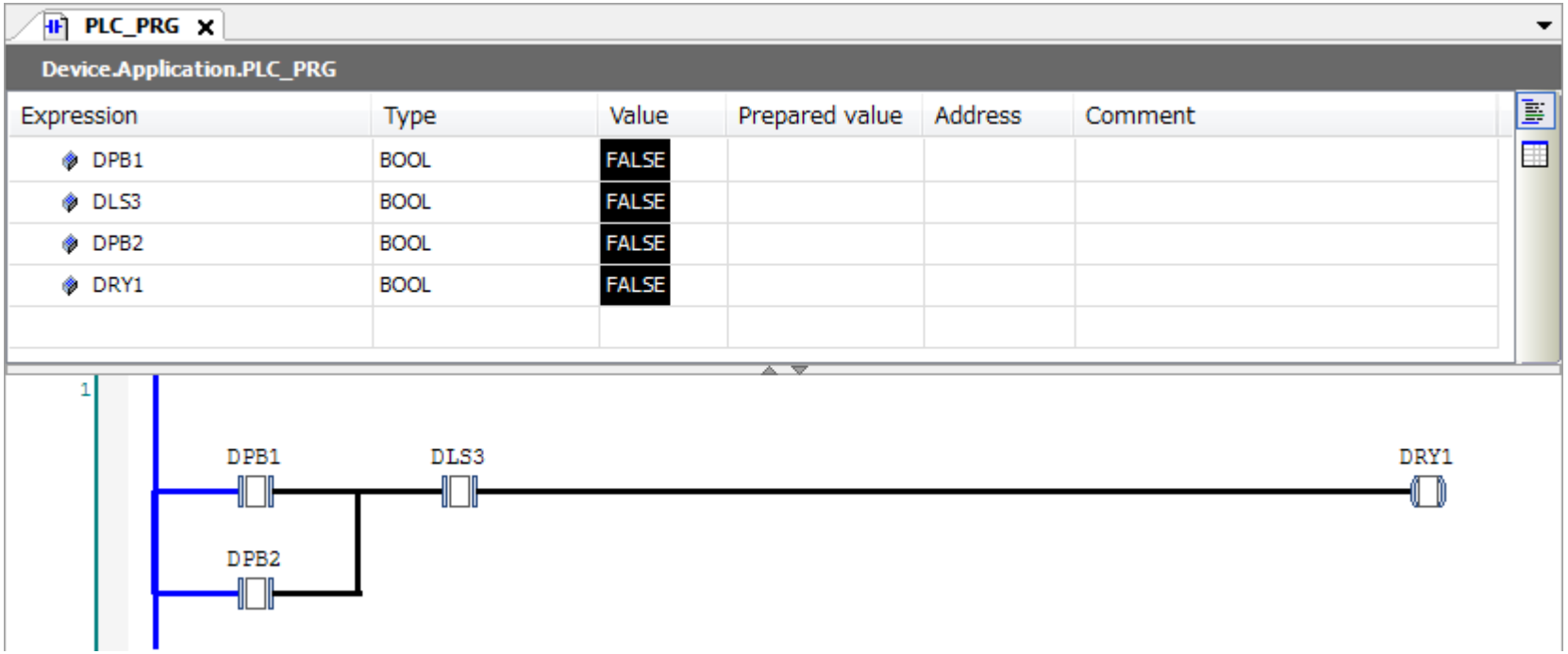


Online

Go Online by icon or [Alt]+[F8] or menu [Online] – [Login]



Program is automatically downloaded to PLC when logging in and the window switches to Online monitor

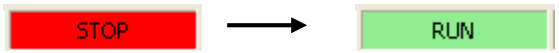


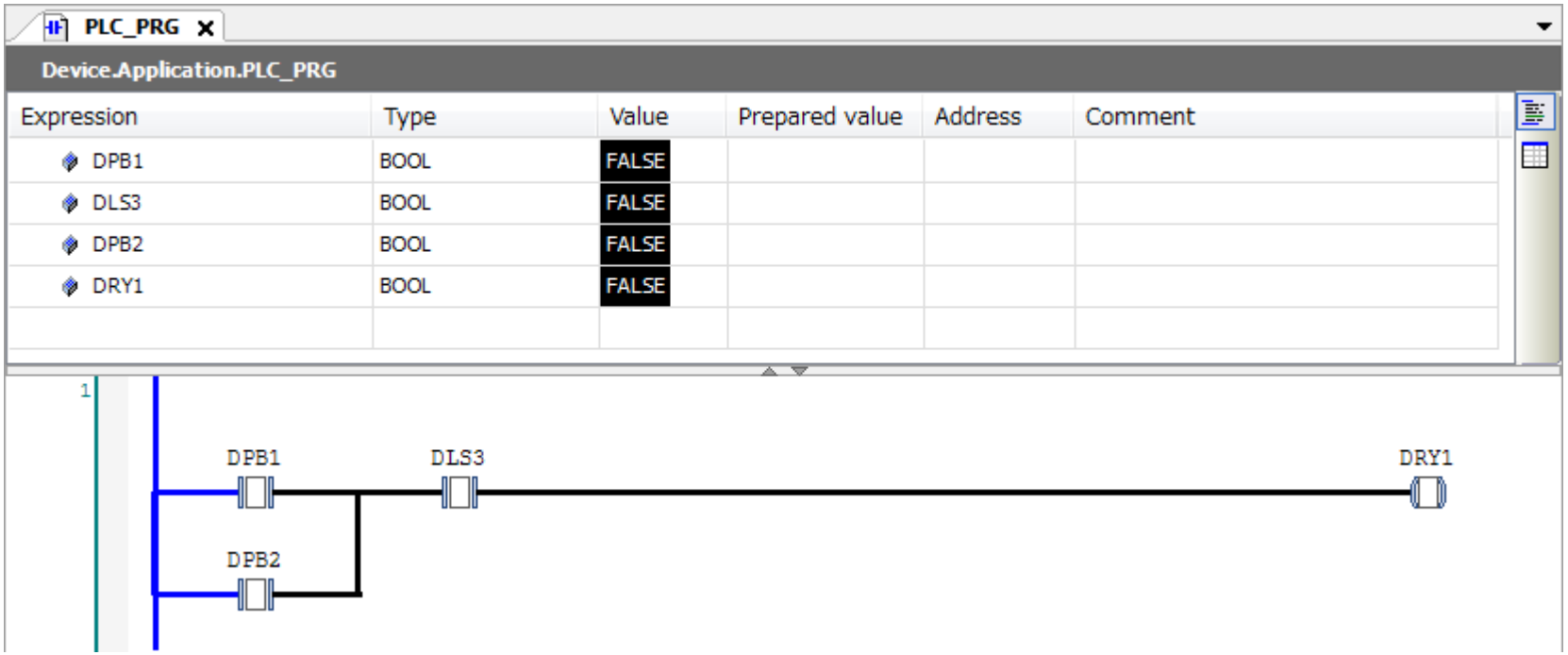
Commissioning (Write value)

HITACHI
Inspire the Next

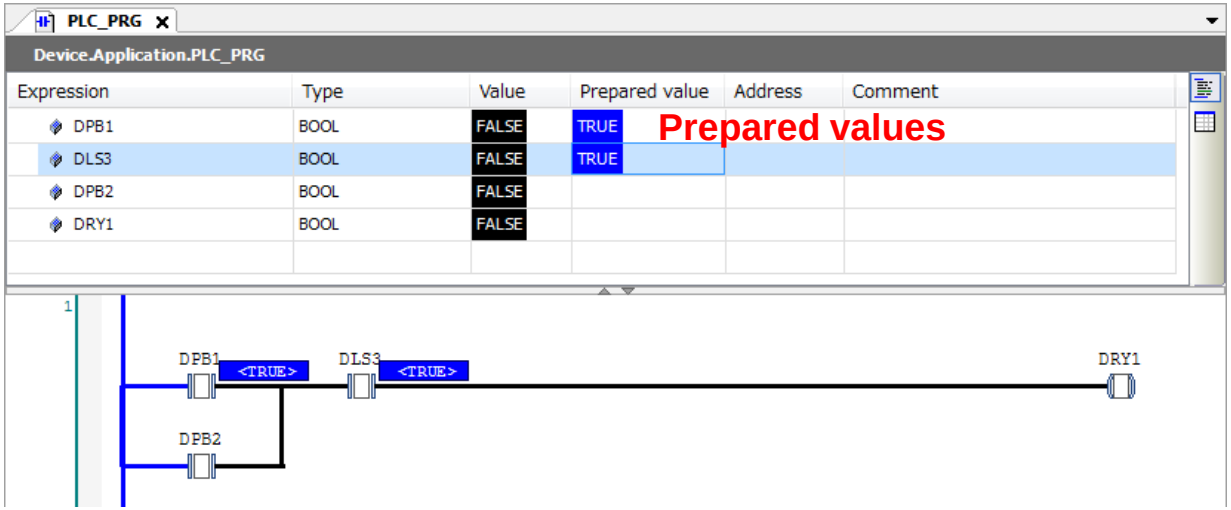
Run

Run by icon or [F5] or menu [Debug] – [Start]



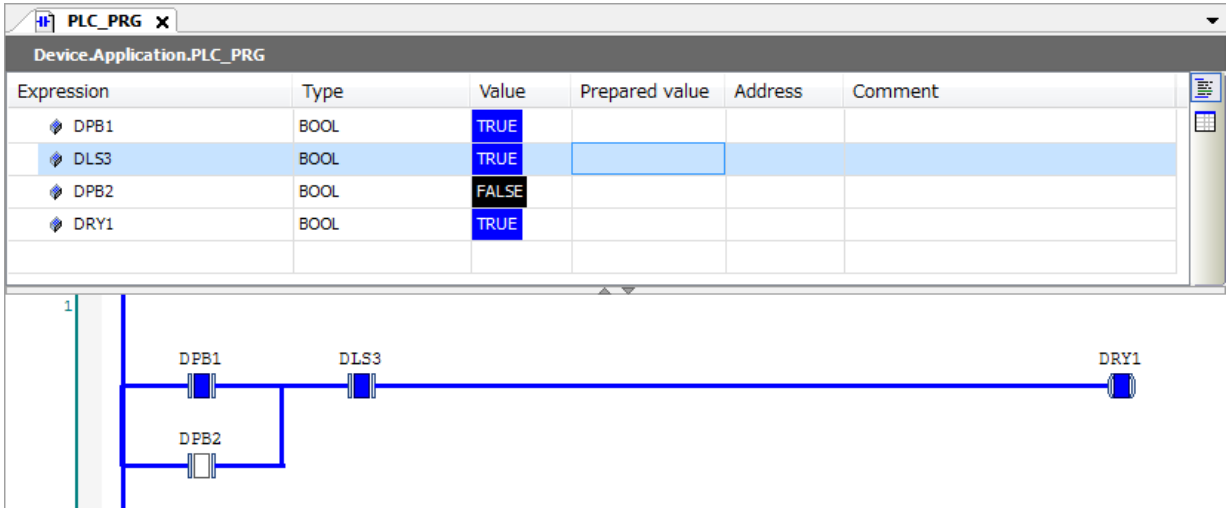


Commissioning (Write value)



① Prepare TRUE by double-clicking DPB1, DLS3

② Apply the prepared value for PLC by [Ctrl]+[F7] or menu [Debug] - [Write value]



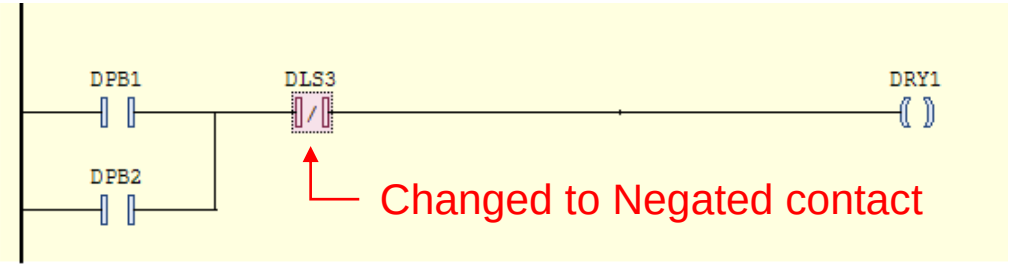
Commissioning (Online change)

HITACHI
Inspire the Next

Procedure

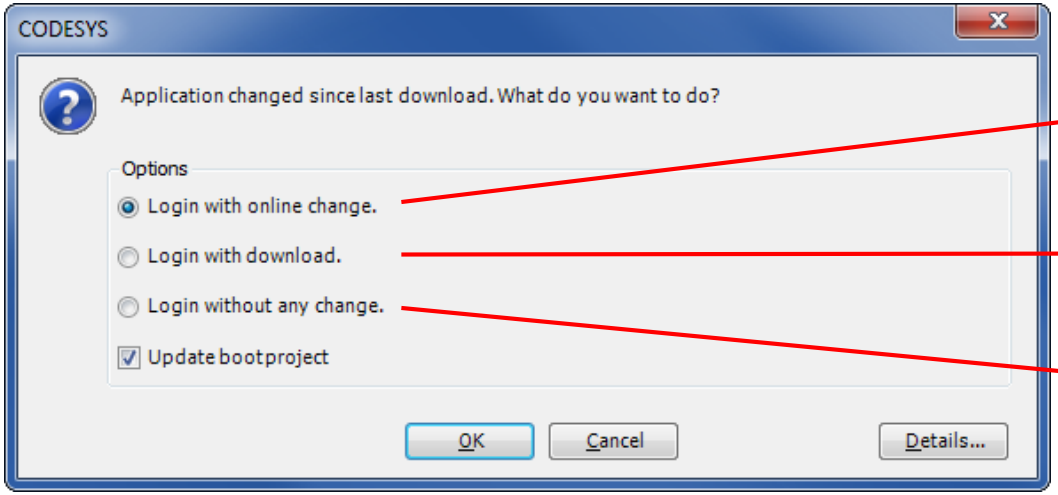
1. Logout (Off line)

2. Change program



3. Login (Online)

Select “Login with online change”



Online change

Program download
(CPU stops while downloading)

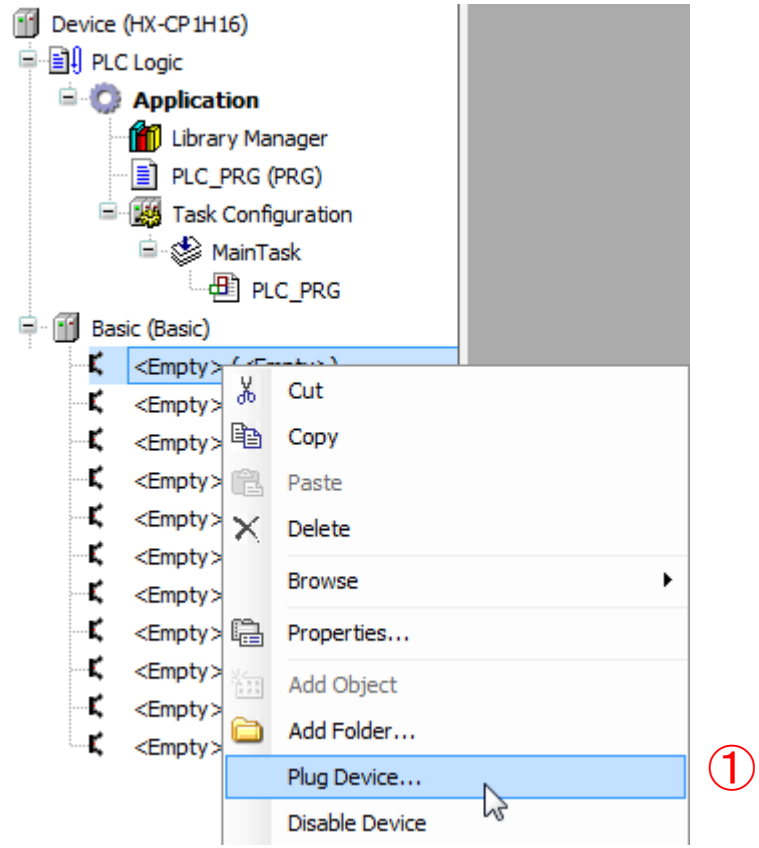
Login without changing

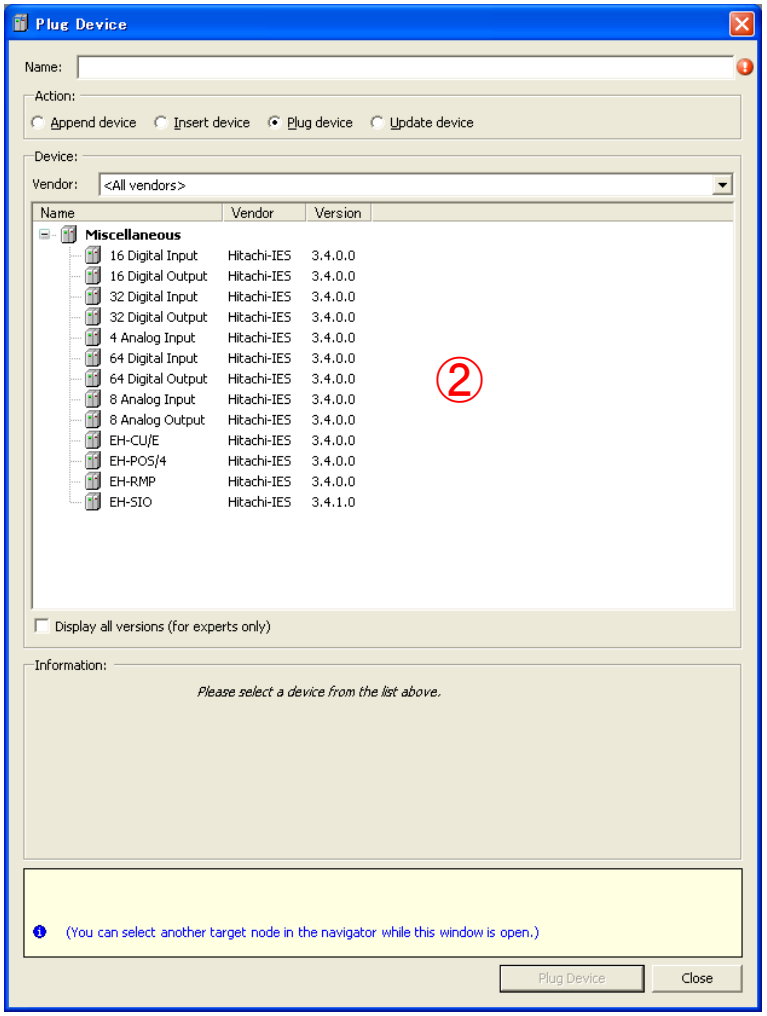
I/O Assignment (Program using External I/Os)

HITACHI
Inspire the Next

I/O assignment (Manual configuration)

① Select [Plug Device] after right-clicking a target empty slot under “Basic”





② Select I/O module at Plug Device window

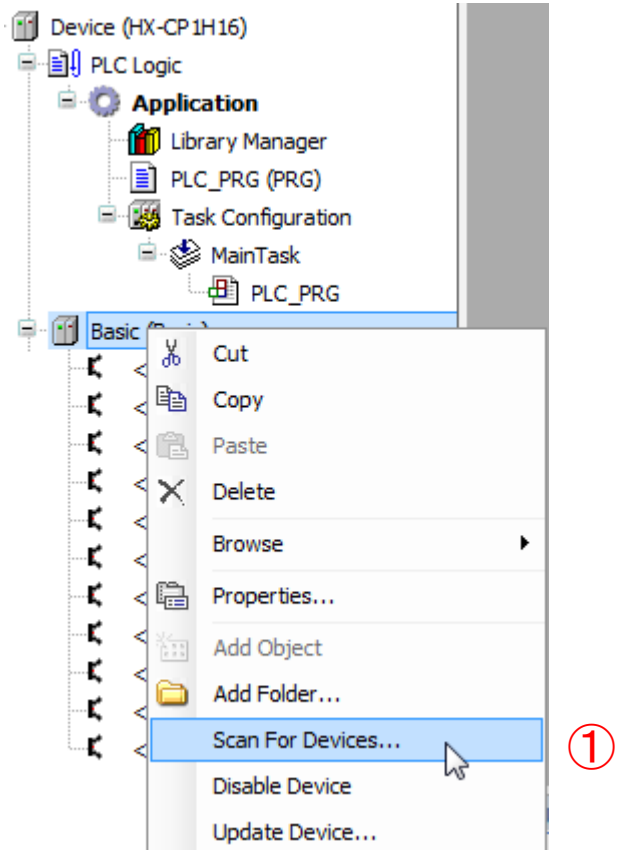
(continuous configuration without closing the dialog available)

I/O Assignment (Program using External I/Os)

HITACHI
Inspire the Next

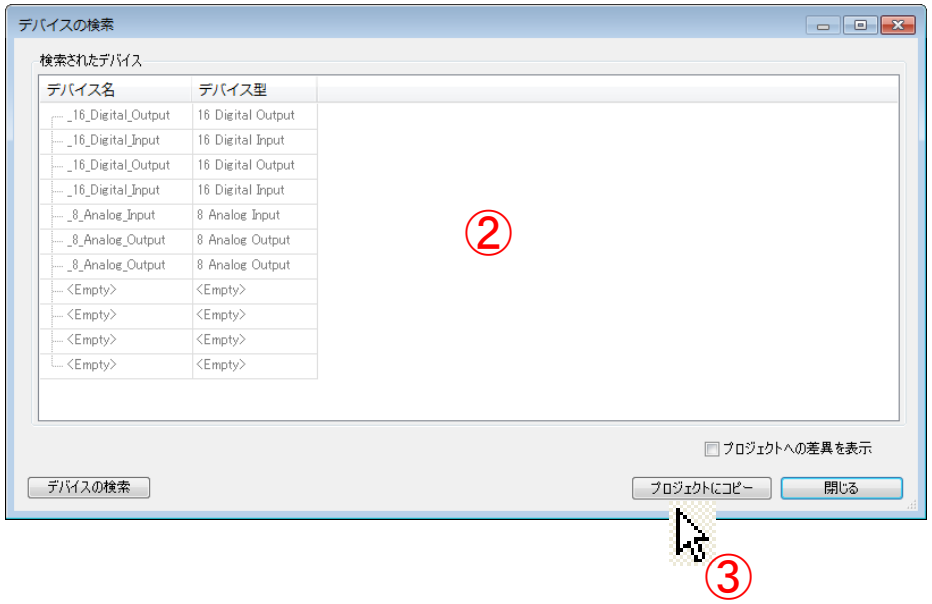
I/O assignment (Scan Device)

① Select [Scan For Devices] after right-clicking “Basic”



② I/O modules mounted on base are detected

③ Click [Copy All Devices to Project]



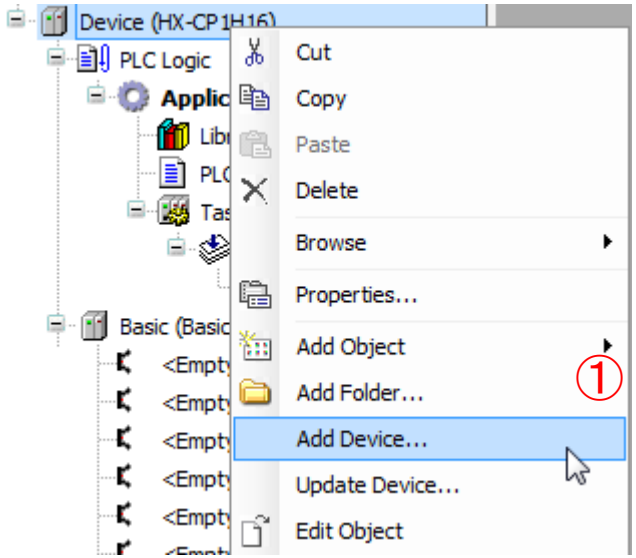
(Note) Scan For Devices is available while Off-line and works once going online to PLC.

I/O Assignment (Program using External I/Os)

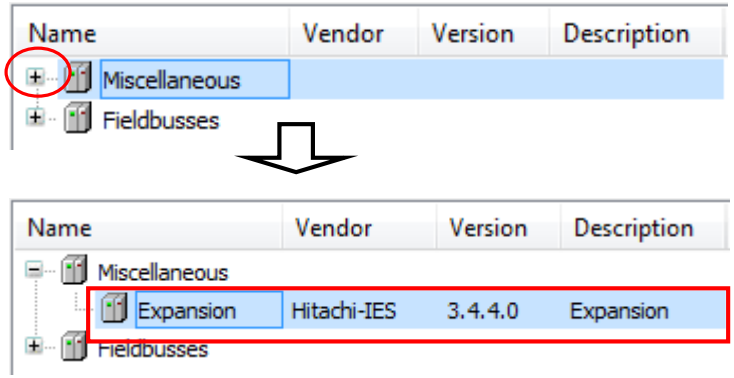
HITACHI
Inspire the Next

I/O assignment (Expansion unit)

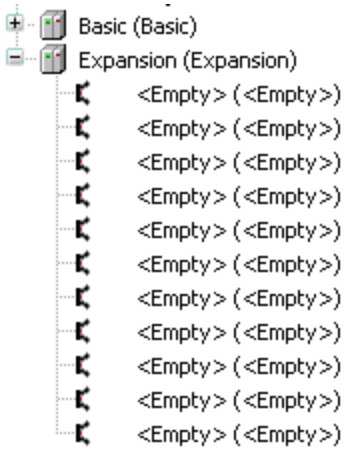
① Select [Add Device] after right-clicking “Device”



② Click [+] on “Miscellaneous”



③ Double-click “Expansion”



(Note) “Scan For Devices” is valid within its base unit.
It is required to operate “Scan For Devices” for each base unit when using expansion units.

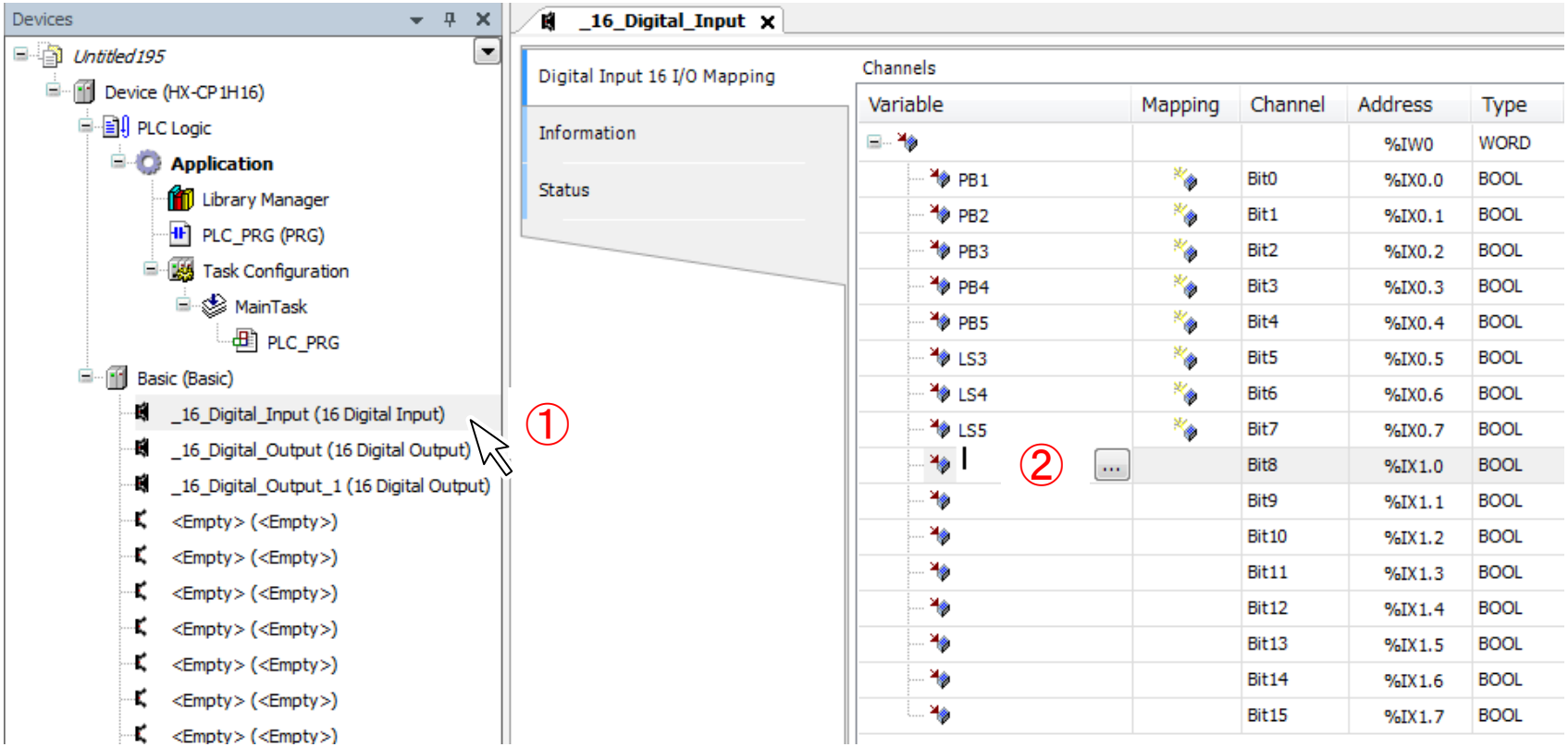
I/O Assignment (Program using External I/Os)

HITACHI
Inspire the Next

Programming with the use of External I/Os

① Open I/O Mapping window by double-clicking an assigned I/O module

② Enter Variable Name for each I/O



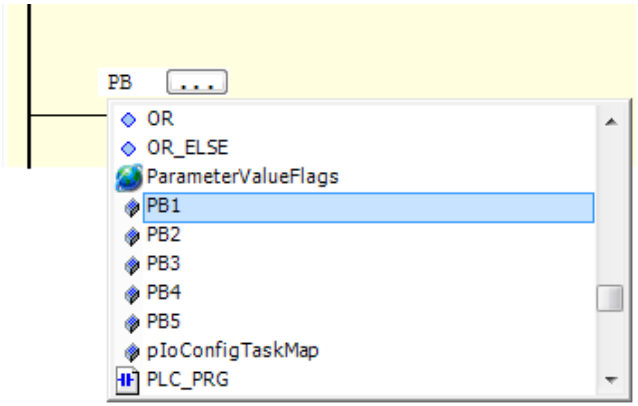
(Note) Declared variables in I/O Mapping window are treated as Global variables.

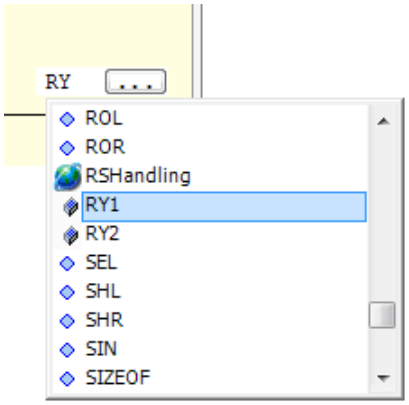
I/O Assignment (Program using External I/Os)

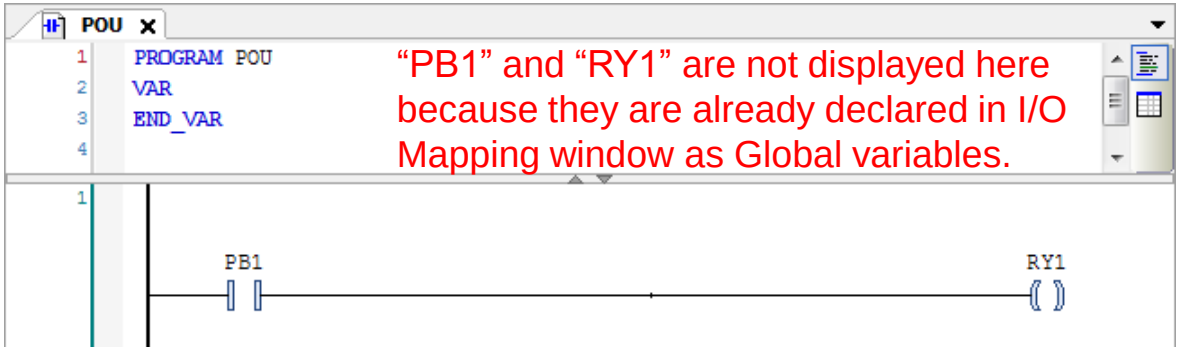
Programming with the use of External I/Os

③ Global variables are listed in Intellisense when typing initial letter.

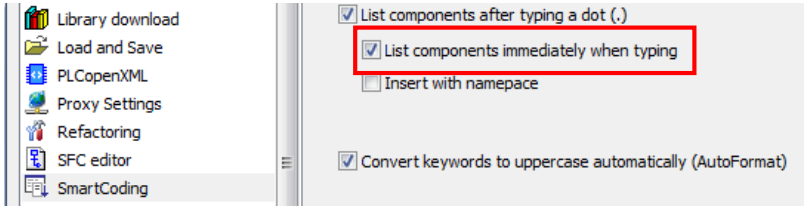
④ Select the one and ENTER







"PB1" and "RY1" are not displayed here because they are already declared in I/O Mapping window as Global variables.

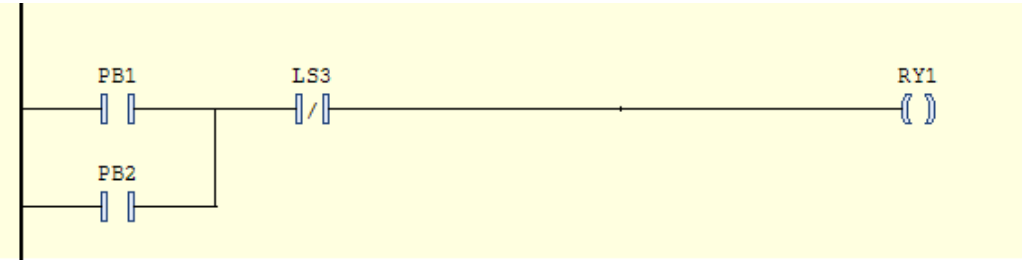


※ Intellisense can be activate/deactivate by configuring "List components immediately when typing" option in [Tool] - [Option] - [SmartCoding]

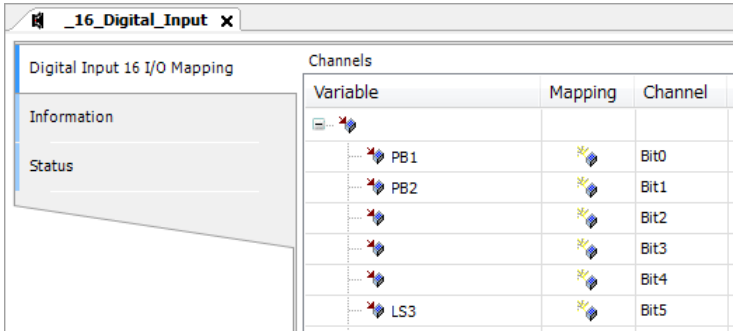
I/O Assignment (Program using External I/Os)

HITACHI
Inspire the Next

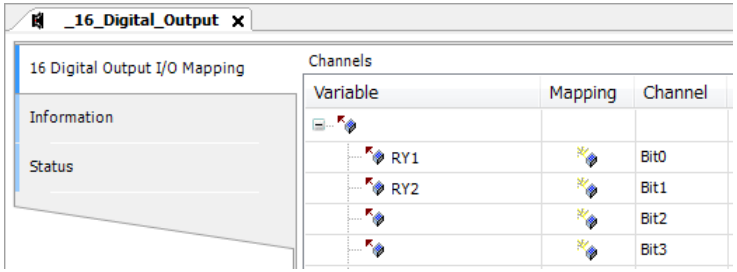
Programming with the use of External I/Os



- 16 Digital Input

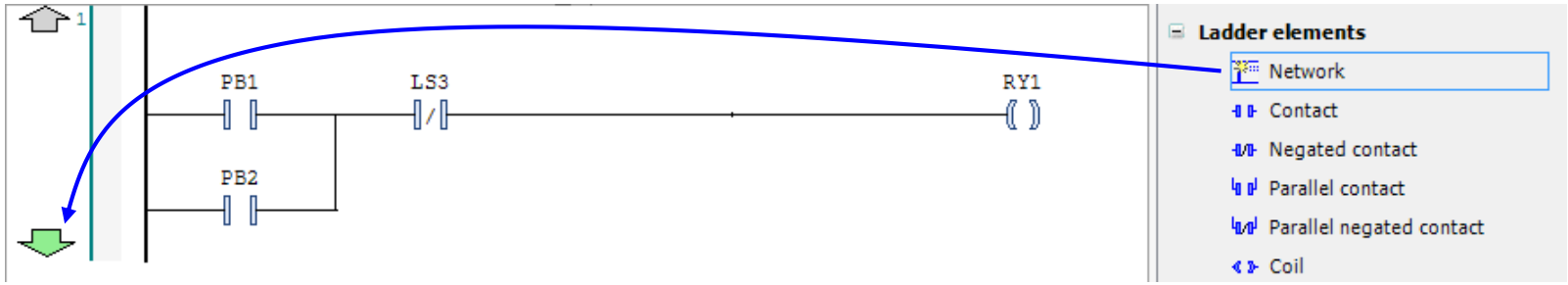


- 16 Digital Output



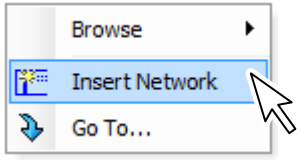
Let’s download program and RUN CPU!

Insert Line



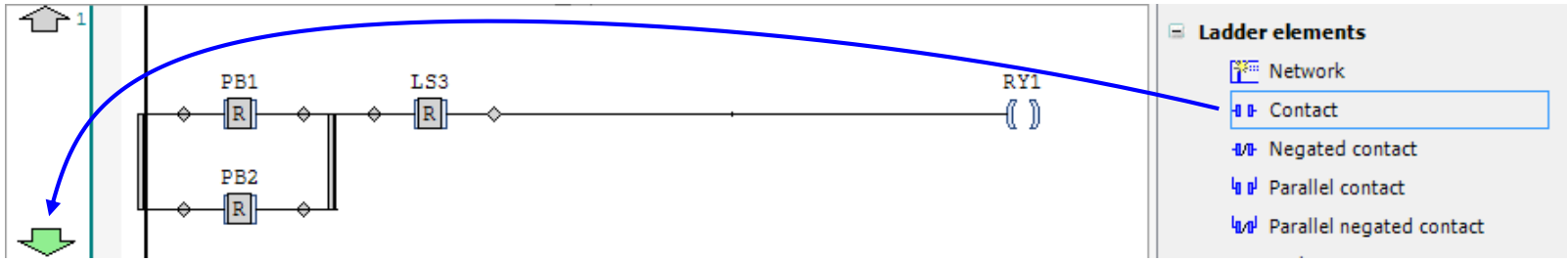
Drag “Network” to

Insert Line (Another way 1)



Click [Insert Network] after right-clicking blank space of program editor window

Insert Line (Another way 2)

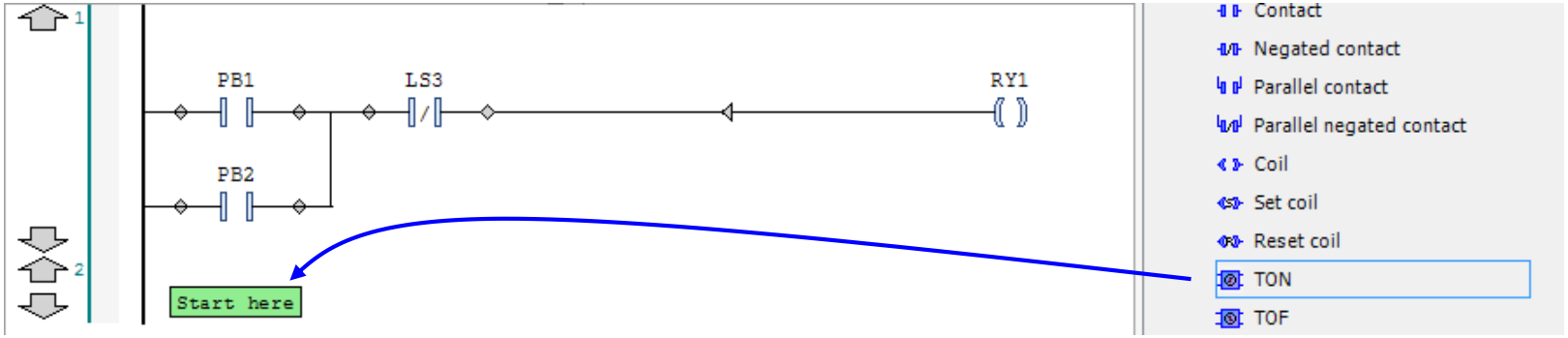


Drag a symbol you would like to add to next line to

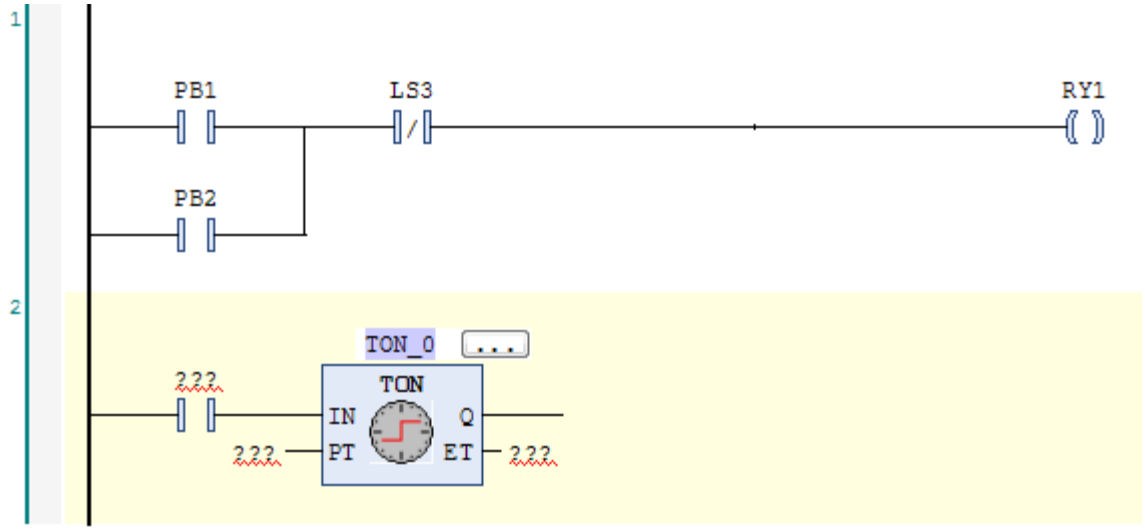
Add FunctionBlock (timer)

HITACHI
Inspire the Next

Add “On delay timer”

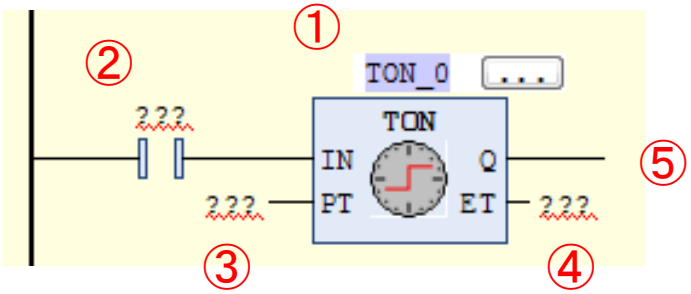


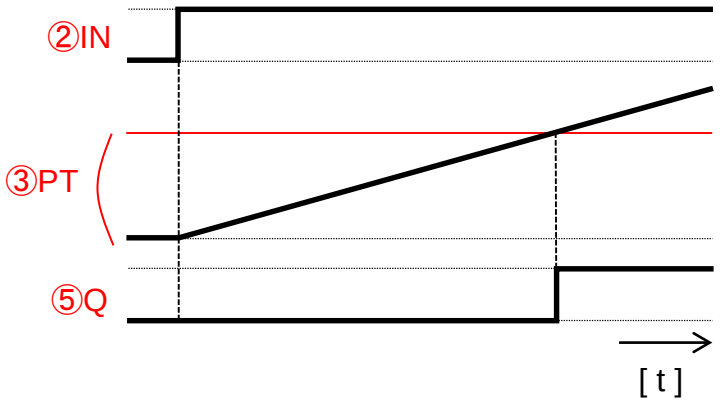
Drag “TON” to `Start here`



Add FunctionBlock (timer)

Add “On delay timer”





① FunctionBlock needs a identifier (instance).

② [input] IN: Rising edge starts counting up ET.

③ [input] PT: Upper limit for counting up ET (delay time).

(* Example: T#10s)

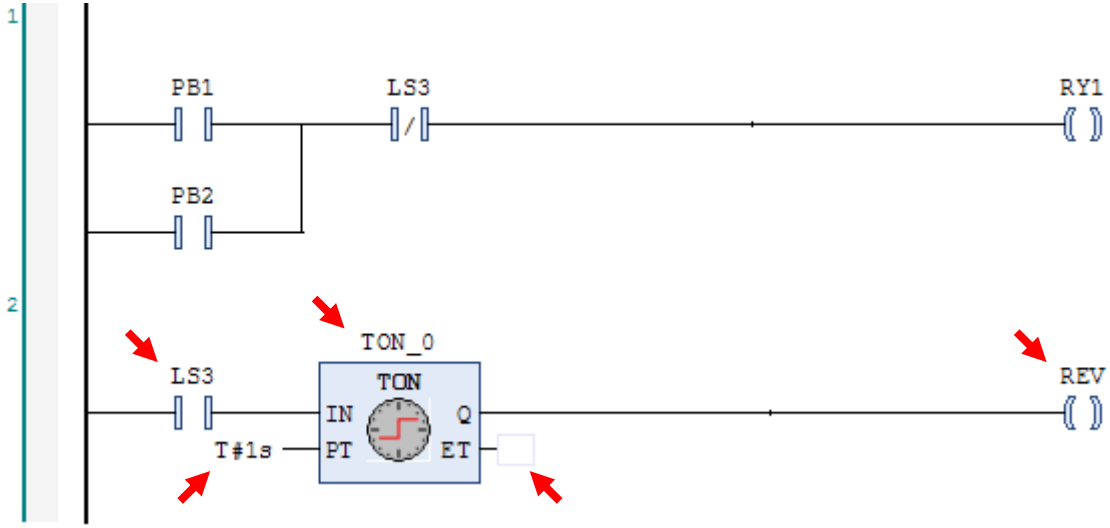
④ [output] ET: current state of delay time.

⑤ [output] Q: Gets a rising edge as soon as ET has reached the upper limit PV (delay time is over).

Add FunctionBlock (timer)

HITACHI
Inspire the Next

Add “On delay timer”

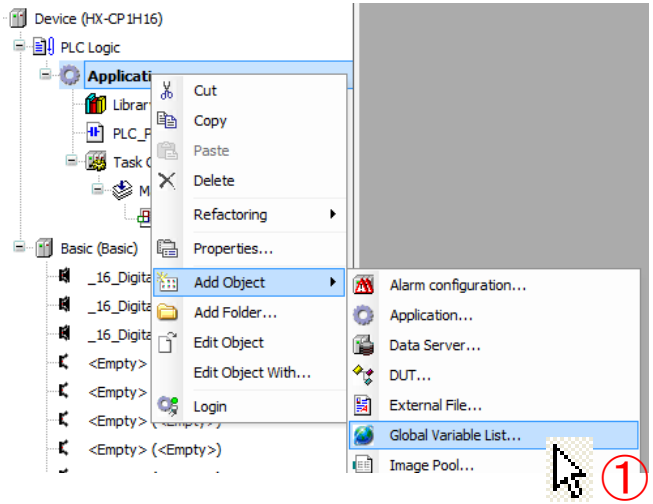


Global Variables and Local Variables

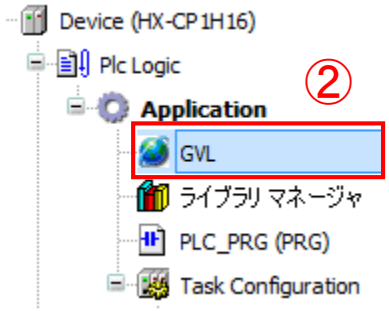
Local variables which are declared in each POU are valid only in the POU

Global variables can be accessed from all POUs in a project

① Select [Add Object] - [Global Variable List] after right-clicking “Application”

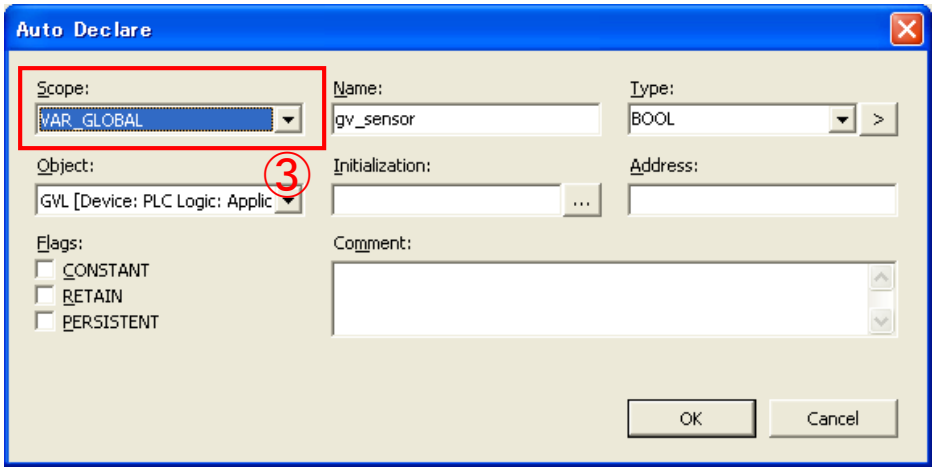


② Global variable list is added to the project

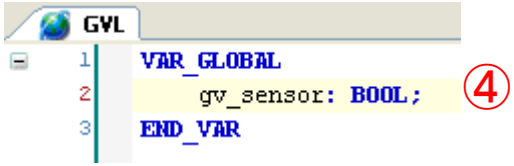


Global variable list

③ Select “VAR_GLOBAL” at Scope when declaring a variable



④ The variable is registered to GVL

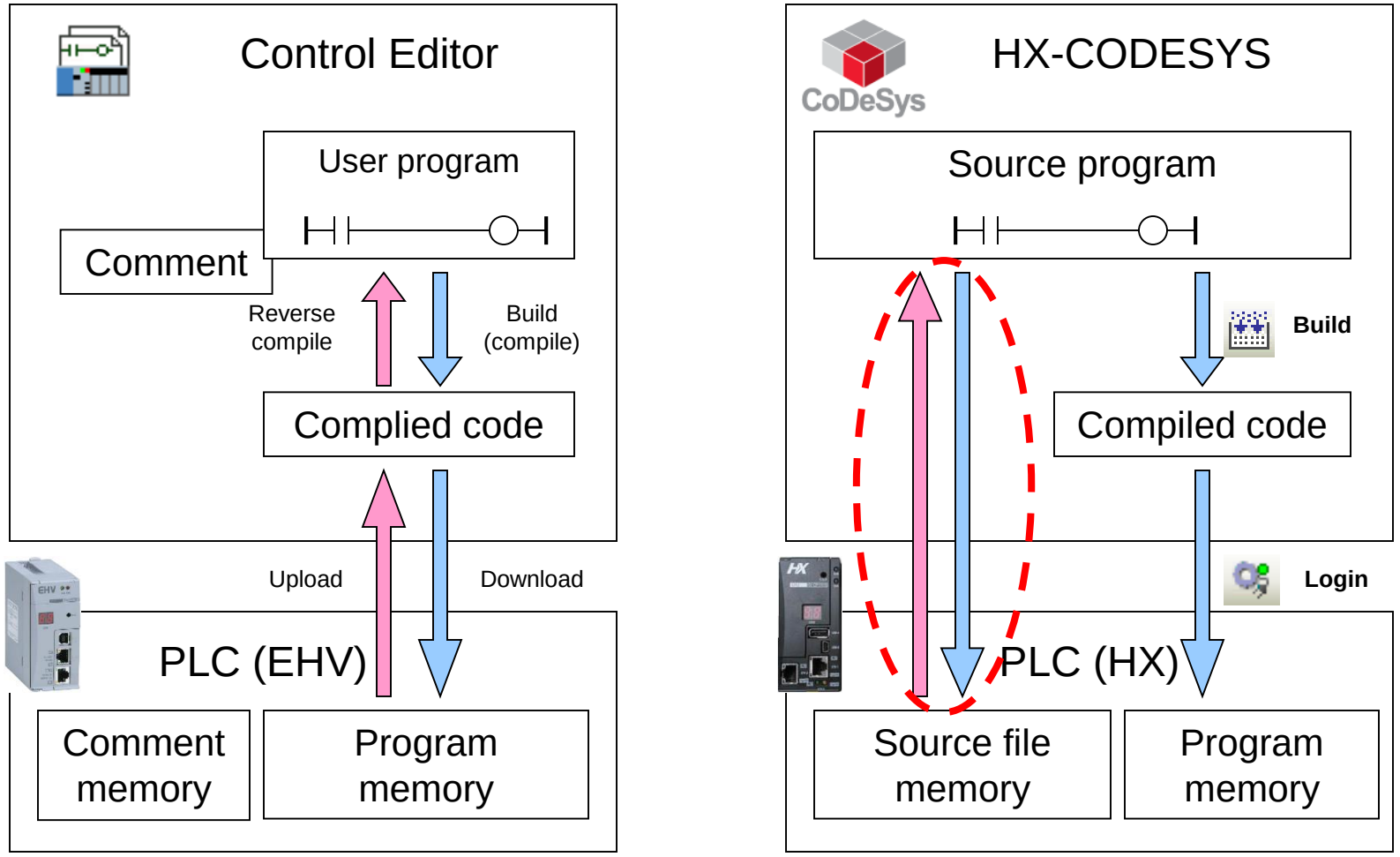


⑤ put “**`GVL.`**” when programming

Ex) **`GVL.gv_sensor`**

Differences between Hitachi PLC and CODESYS PLC

Flow of program up/downloading



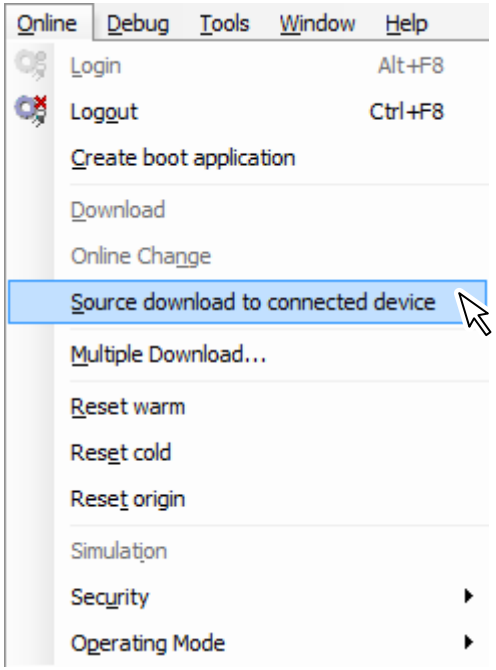
Source upload/download

HITACHI
Inspire the Next

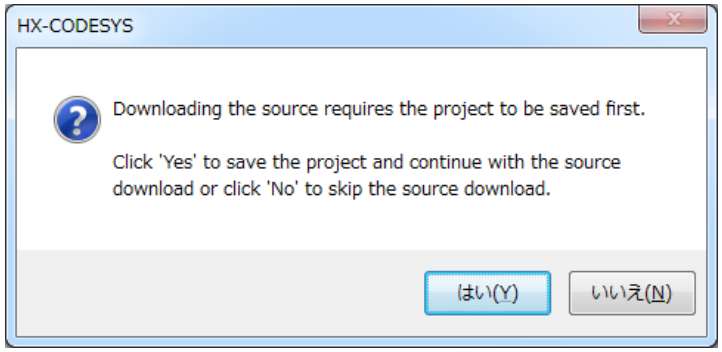
Source download

① Go Online by [Login]

② Click [Online]-[Source download to connected device]

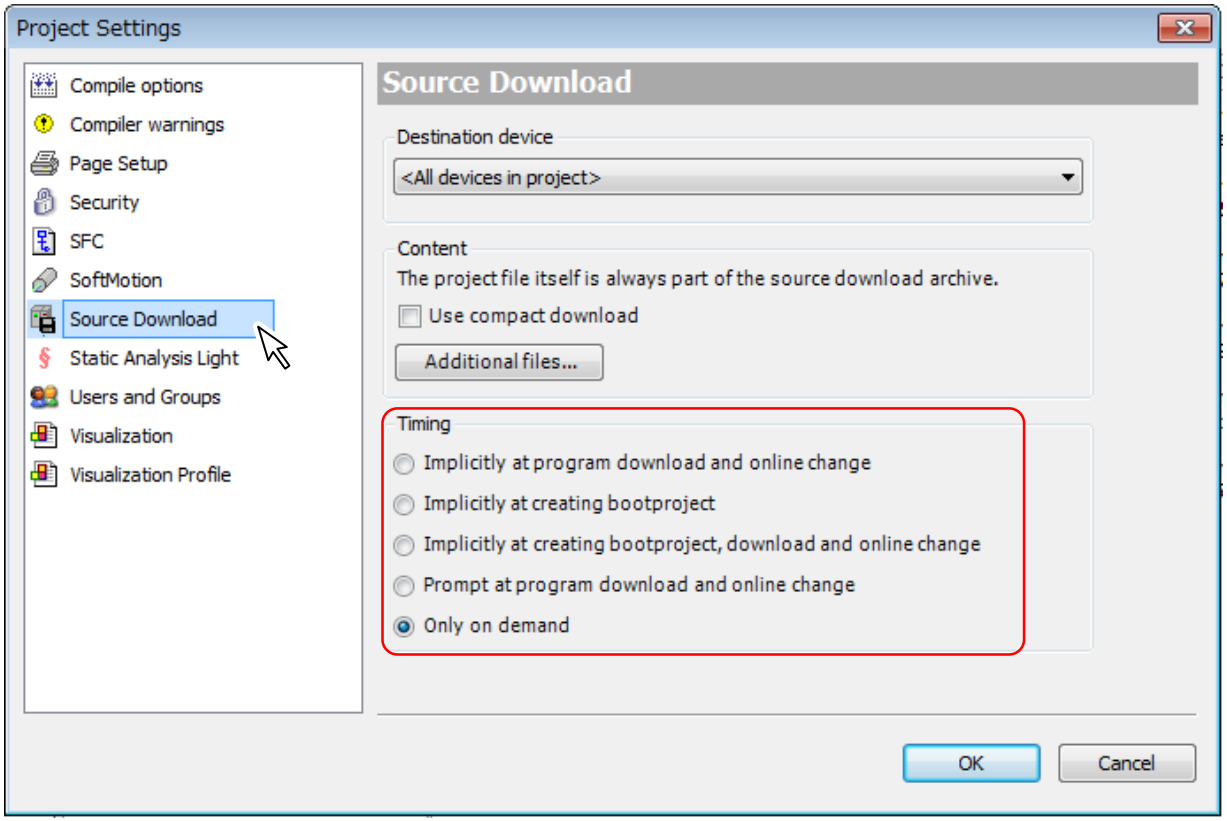


If the source program is not saved as a file, the following dialog window will be displayed. Click [Yes] to save the project and continue with the source download.



Source download (Another way)

Downloaded timing of source file can be set in the menu [Project]-[Project Settings]-[Source Download]. The default setting is [Only on demand].

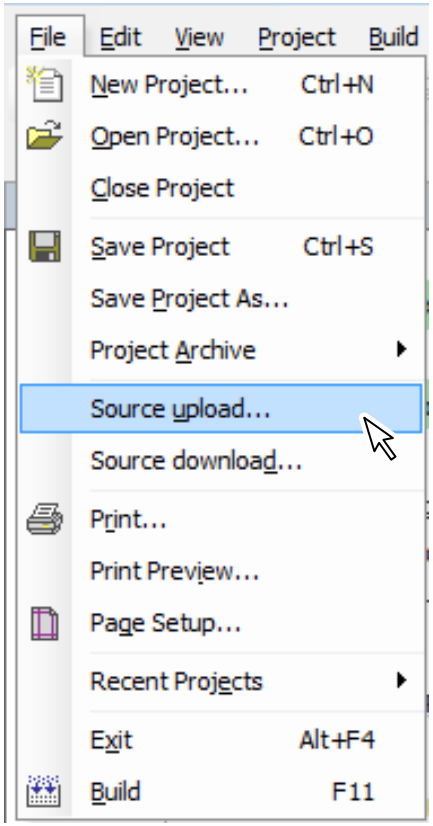


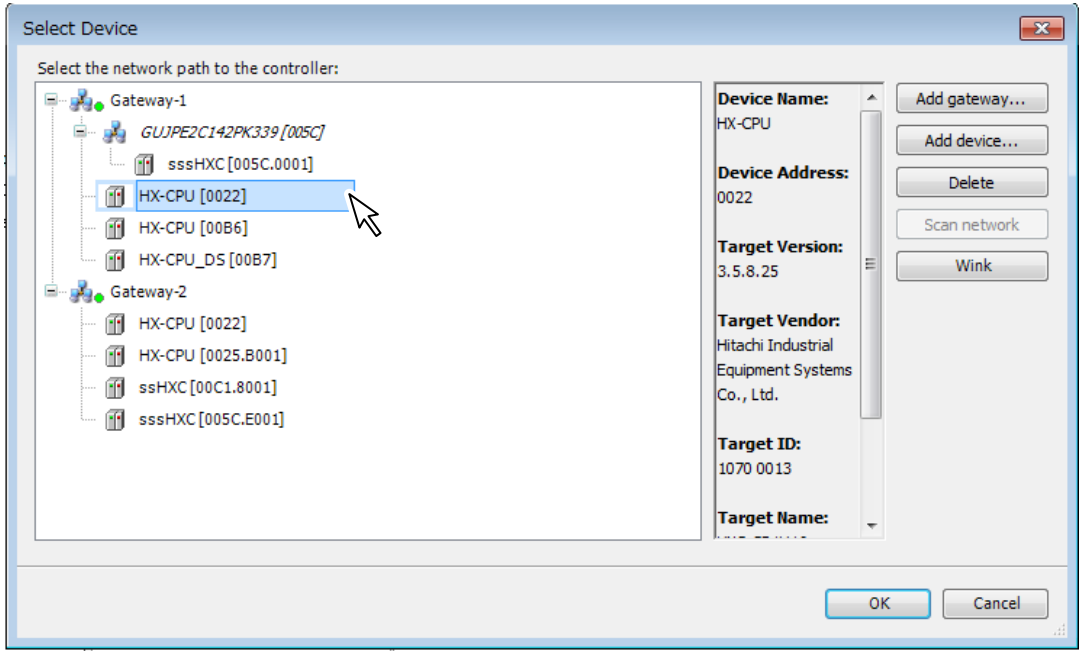
Source upload/download

Source upload

① Click [File]-[Source upload]

② Select a device to upload the source program in [Select Device] window.



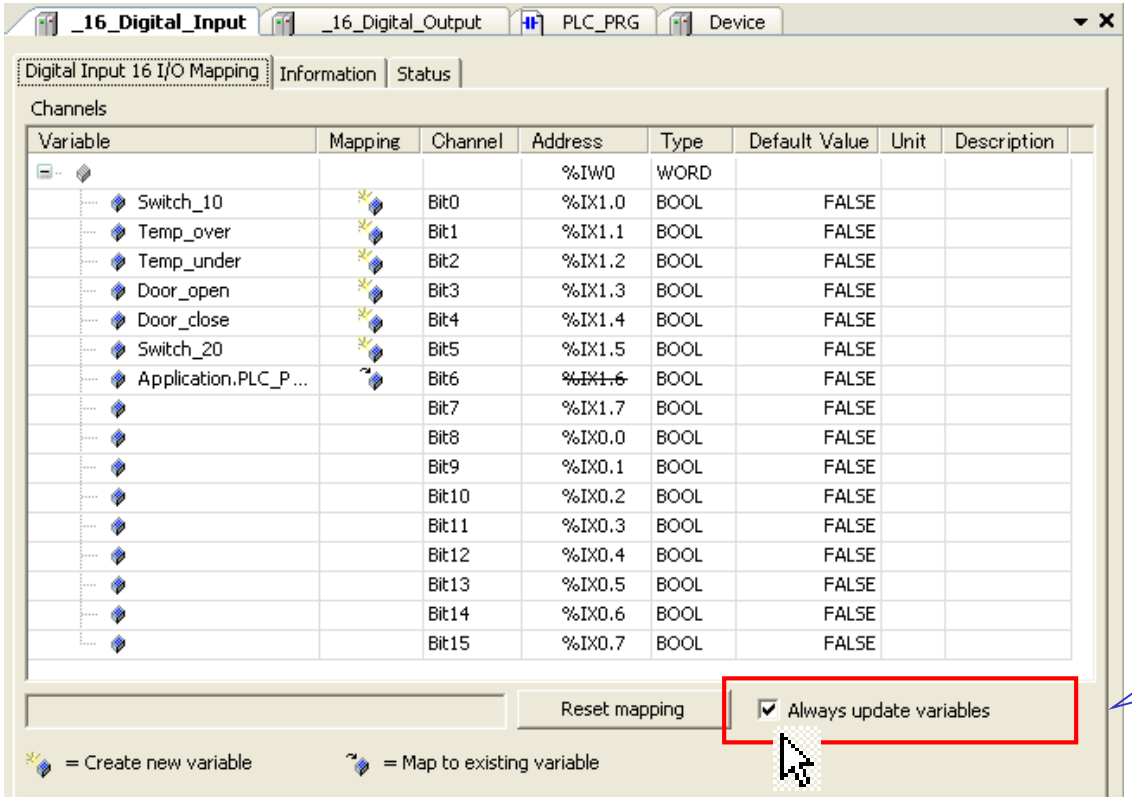


Differences between Hitachi PLC and CODESYS PLC

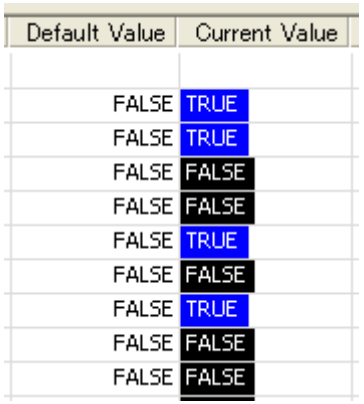
HITACHI
Inspire the Next

I/O refresh

Only I/Os used in POU are updated. So if login with empty program and digital input is activated, the status is not seen correctly in the mapping table.



Login



Enable “Always update variables” to update all I/Os.

Possible to set in “PLC settings”

Differences between Hitachi PLC and CODESYS PLC

HITACHI
Inspire the Next

Run/Stop status and Run/Stop switch

Run/Stop from CoDeSys works only when RUN/STOP switch position is in RUN.

Operation / Switch position	Stop by CoDeSys	Run by CoDeSys	Power ON
STOP	Stop	Stop	Stop
RUN	Run➔Stop	Stop➔Run	Run

In case of EH/EHV, remote operation works only when switch position is in STOP.

Operation / Switch position	Stop by PC	Run by PC	Power ON
STOP	Run➔Stop	Stop➔Run	Stop
RUN	Run	Run	Run

Differences between Hitachi PLC and CODESYS PLC

HITACHI
Inspire the Next

Stop and Reset

CPU has 3 different operations, Stop, Reset warm and Reset cold to stop PLC.

Stop or Reset warm can be assigned to stop position of RUN/STOP switch. The default setting is Reset warm.

When CPU stops with error (exception), Reset warm or Reset cold must be operated to reset error status.

CPU status / Operations	Run/Stop	Error	variables	Retain variables
Stop	Run➔Stop	Remains	Remains	Remains
Reset warm	Run➔Stop	Reset error	Return to default	Remains
Reset cold	Run➔Stop	Reset error	Return to default	Return to default

Differences between Hitachi PLC and CODESYS PLC

HITACHI
Inspire the Next

Status of variables and program for users' operations

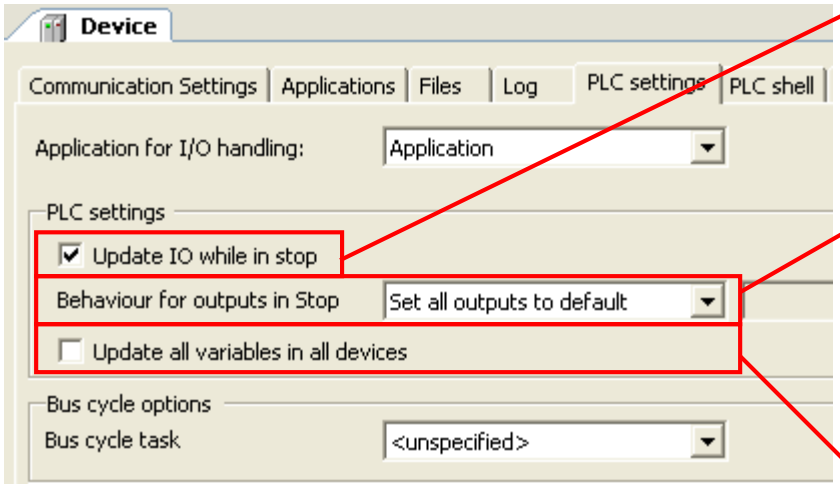
target / Operations	VAR	VAR RETAIN	VAR PERSISTENT	Application (RAM)	Boot application (FLASH)
STOP	-	-	-	-	-
Reset warm	X	-	-	-	-
Reset cold	X	X	-	-	-
Reset origin	X	X	X	X	X
Download	X	X	-	updated	Depends on setting
Online Change	-	-	-	update	Depends on setting
Reboot PLC (Power OFF/ON)	X	-	-	X	-

-: remain X: initialized

Differences between Hitachi PLC and CODESYS PLC

HITACHI
Inspire the Next

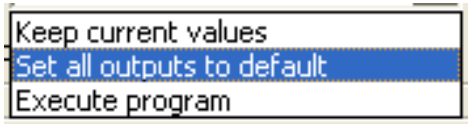
PLC settings



Update IO while in stop
(default: enable)

Behavior for output in stop (Note1)

- Keep current values
- Set all outputs to default (Note2)
- Execute program



Update all variables in all devices
(independent from used or not.)
Default : disable

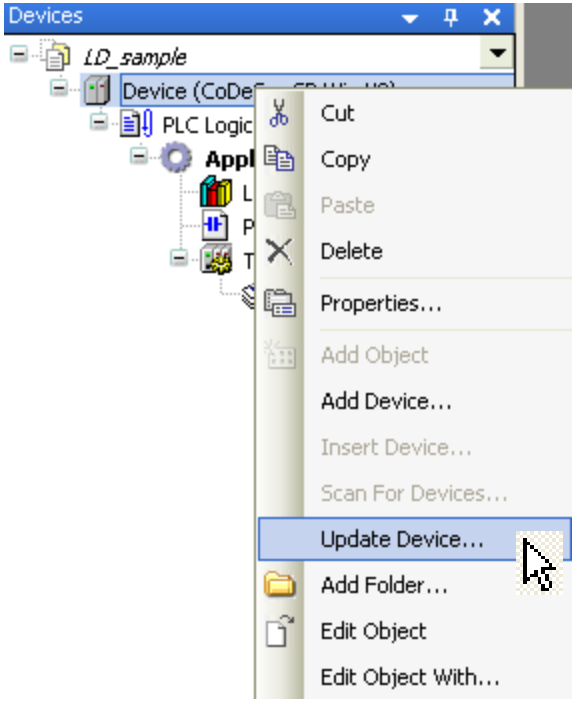
Note1: This parameter is valid only when “Reset all outputs in STOP” is set as No.
If this is Yes, all I/Os are initialized as 0 by hardware.

Note2: Default values are configured in I/O Mapping table.

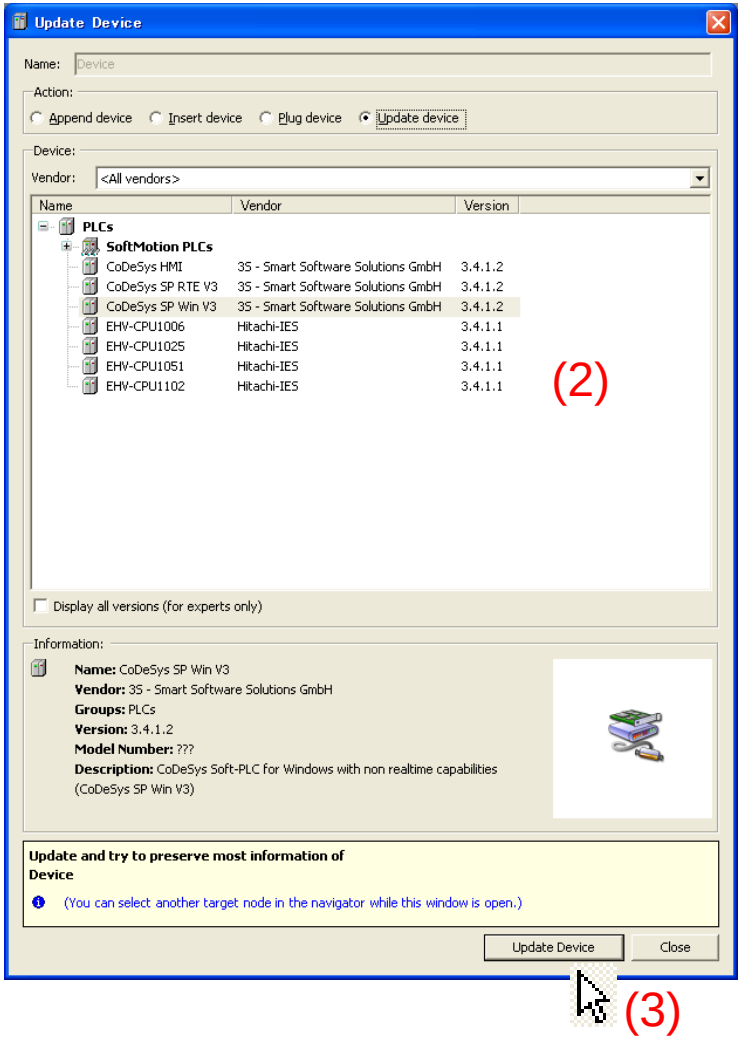
Additional notes

Change CPU type

(1) Right mouse click at Device in the project tree and choose Update Device.



(1)



(2)

(3)

(2) Choose CPU in the dialog.

(3) Click [Update Device]

END