

CPDev – LD programming

CONTENTS

• Introduction	1
• START-STOP diagram	1
Diagram elements	1
• New program	2
Project	2
Program name and language	2
• LD graphic editor	3
Tree	3
Elements with EN	4
Find element	4
Connections without or with EN	4
Editor options	5
• Development steps	5
Global variables	6
Rung	6
Task. Compilation. Simulation	6
• First rung of START-STOP	6
Two straight rungs	6
Branch point and manual connection	7
Contacts and coils as variables	7
Verification	9
Editing	9
Automatic connections disabled	9
• Second rung of START STOP	10
Elements and connections	10
Instance names	11
Constants	11
Comments	11
Verification	11
Automatic connections disabled	11
• Task and compilation	12
Task	12
Compilation – Build	12
• Online mode	13
Go online	13
LD diagram online	13
Value list	15
Data sources	15
• CPSim simulator	15
Global variables	16
Control panel	16

Individual view	16
Variable list	17
Sessions	17
• Elements with EN	17
Rung conditions	17
Functions	18
Input/output negation	18
Manual connections	18
START STOP with EN	18
• Other issues	18
Changing LD program name	18
Library warnings	19
Automatic save	19
• Standard examples	19
Protection against frequent restarts	19
Start-up horn	20
Blinking light	20
Two heaters	20
• Examples with EN	21
Single pushbutton	21
Low/High alarms	22
• Sequential controls	22
Greenhouse	22
Reverse switching	23
• More on connections	24
Parallel connections	24
Warnings for EN	25
Wrong connections	25
• Printing	25
Size adjustment	25
Page setup	26
Print	26

15 February, 2018

Editor version: LD 0.4.2.1

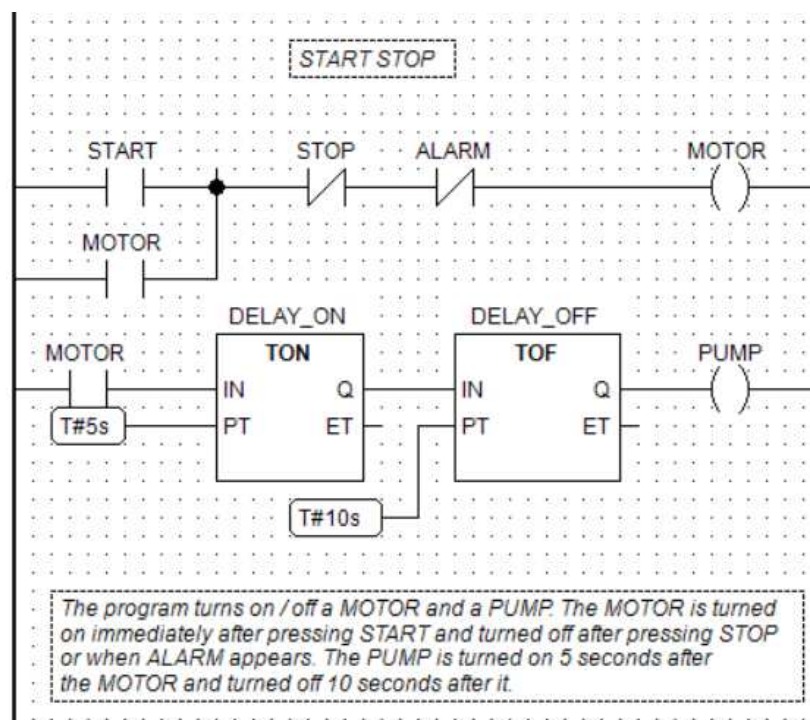
Introduction

CPDev environment supports programming in graphic LD language (*Ladder Diagram*) by means of *LDEditor* module (external component). Users can create programs and trace values of variables during simulation. In this instruction creating a typical program in LD language involving contacts, coils and function blocks is described in details. Other examples are also provided, such as common switching circuits, sequences and simple processing of other variables than BOOL. It is assumed that the user is familiar with basics of using CPDev environment.

START-STOP diagram

Diagram elements

- Final program in LD language is shown below. It contains:
 - contacts START, MOTOR
 - negated contacts STOP, ALARM
 - coils MOTOR, PUMP
 - instances DELAY_ON, DELAY_OFF of function blocks TON, TOF
 - constants T#5s, T#10s
 - comments.

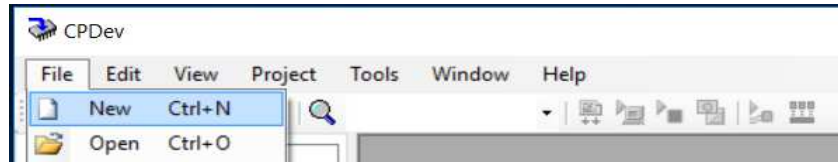


- Main part of this instruction describes how to create and simulate such program. Other examples are presented later.

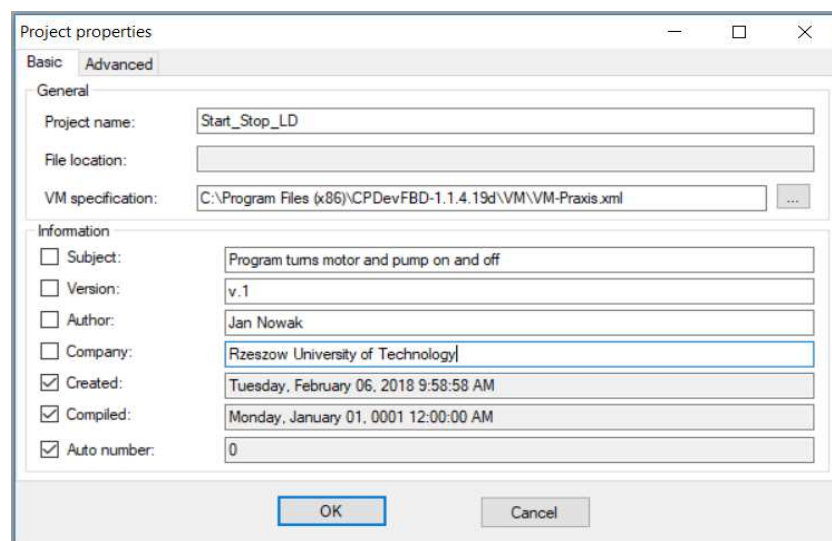
New program

Project

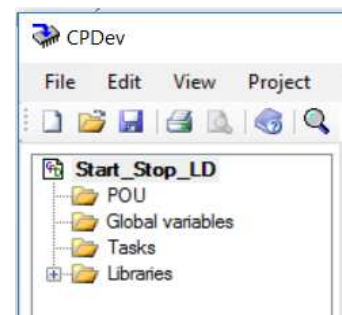
- New project is defined by selecting *File* → *New* in CPDev main menu or clicking *New* icon in the toolbar.



- Enter *Project name*, here *Start_Stop_LD*, and eventually other data in *Project properties* window.



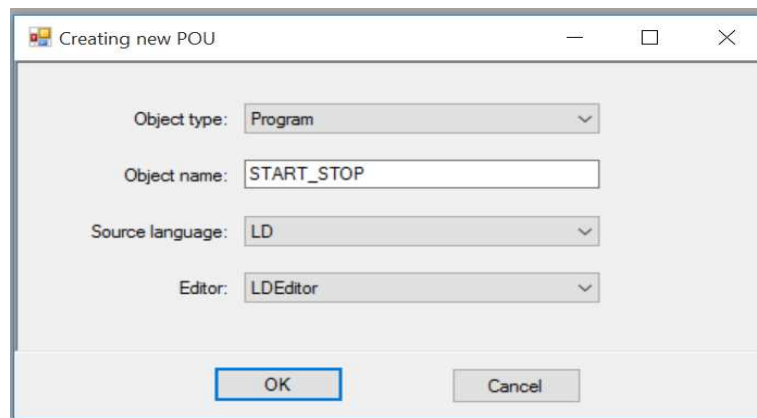
- File location, compilation date and number will be filled in automatically while saving the project.
VM specification indicates path to virtual machine description file.
- Left part of CPDev window presents initial tree of the *Start_Stop_LD* project.



Program name and language

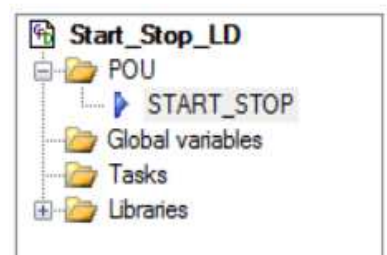
- Program can be added to the project in two ways:
 - from project tree:
 - select project name, i.e. *Start_Stop_LD*
 - from context menu, select *Add Item* → *Program* or select *POU* and from its context menu choose *Add* → *Program*
 - from CPDev main menu:
 - select *Project*
 - from drop-down menu, select *Item* → *Add* and select *Program* in *Adding new element* window.

- Enter *Object name*, here START_STOP, and choose LD language (default LDEditor appears).



- Project tree involves now START_STOP program.

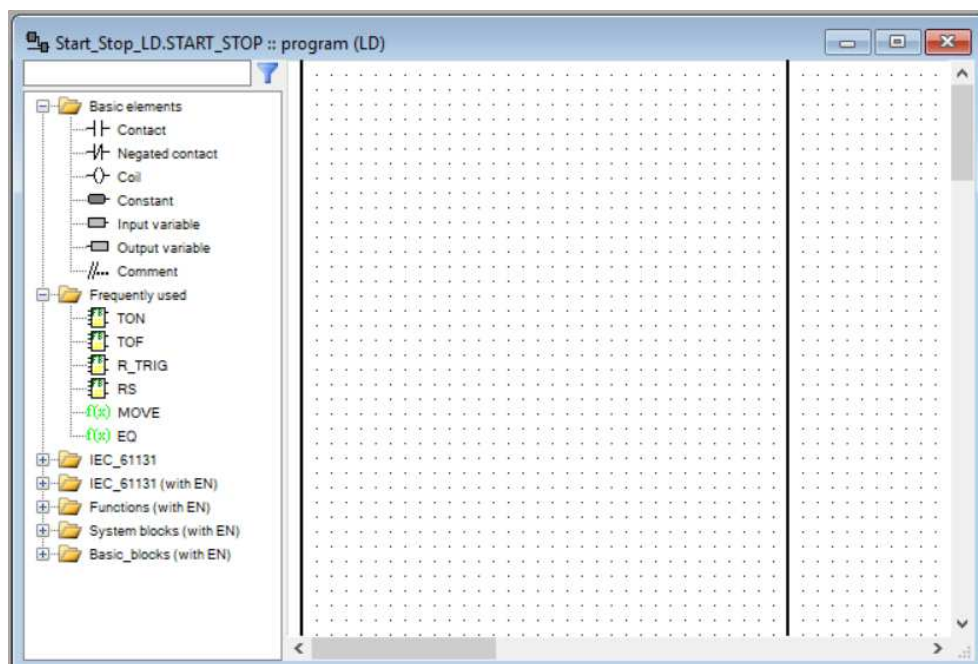
Remark. For consistency with other editors, one can also select *function* or *function block* as object type. However, editor window will not open (with notification at the bottom message list). Functions and function blocks can be created in ST and FBD languages (function blocks also in SFC).



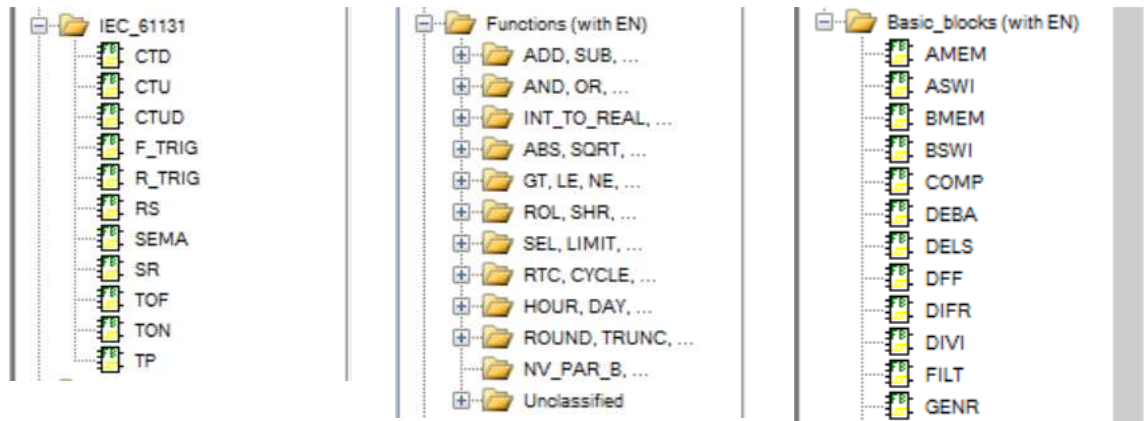
LD graphic editor

Tree

- Left part of LD editor window contains tree whose elements can be dragged or selected and dropped on the drawing board on the right, between vertical lines (power rails).
- The right line moves automatically if element is moved too close or placed beyond it.
- Grey line farther to the right (move scrollbar or maximize) is the limit of first printed page. Grey lines partition total work area into a few pages (see *Printing*).



- The tree involves:
 - basic elements, such as contact, coil, input/output variable, etc.
 - a few frequently used blocks and functions
 - blocks of *IEC_61131* library in standard form (without EN)
 - blocks of the same library, but with additional ENable input (*with EN*)
 - functions, system blocks, and blocks of *Basic_blocks* library, also *with EN*.

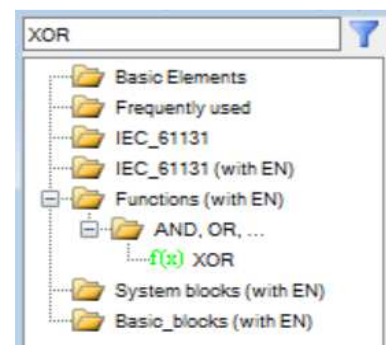


Elements with EN

- Contrary to standard blocks of *IEC_61131* library, all functions and blocks *with EN* are executed only when TRUE appears at EN input. Such elements allow to implement in LD language some simple diagrams typical for FBD.
- *Functions* branch consists of subbranches, such as arithmetic ADD, SUB, ..., logic AND, OR, ..., conversions INT_TO_REAL, ..., etc.
- *Basic_blocks* library involves special arithmetic blocks, switches, memory blocks, comparator, flip-flops, pulsers, filters, and some nonlinearities (see basic CPDev instruction).
- Hardware dependent *System_blocks* such as NV_PAR_... store parameters in nonvolatile memory (BOOL, INT, REAL, TIME). aPON and aSTR raise alarms, i.e. alarm power-on (“warm” restart) and alarm start (“cold”).

Find element

- Expanding the tree to find rarely used function or function block could be cumbersome. To make it easier, user can simply write a part of name in the text box above the tree to trigger automatic filtering of elements.
- Function subbranches and/or libraries with elements including the written text are presented. For example, after typing XOR corresponding gate appears in the tree. Another text may result in a few elements.

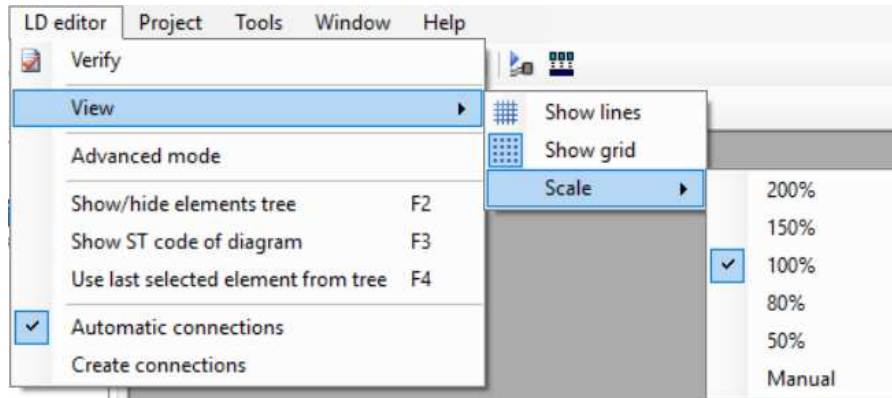


Connections without or with EN

- First input and first output of a block without EN (from *IEC_61131* library) must be connected to other rung elements, such as contact, coil or another such block.
- EN input of a block or function *with EN* can be connected to:
 - left power rail
 - rung-like circuit involving contacts or blocks without EN.

- Other inputs of blocks or functions without or *with EN* must be connected directly to **input variables or constants** (not to contacts, branch points or outputs of other blocks).
- Other outputs of blocks without EN (other than the first one) must be connected directly to **output variables** (not to coils, branch points or inputs of other blocks) or left unconnected.
- Outputs of blocks or functions *with EN* must be connected directly to **output variables** or, except the first one, left unconnected.

Editor options



- The following options are available in *LD editor* menu:
 - *Verify* – checks diagram completeness (elements, connections, etc.)
 - *View* – contains options of diagram presentation:
 - *Show lines* – switches on/off supplementary lines on the diagram
 - *Show grid* – switches on/off grid and grey lines on the diagram
 - *Scale* – chooses scale ratio
 - *Advanced mode* – switches on/off advanced mode, where some extra functions and blocks are available (and visible in the tree)
 - *Show/hide elements tree* **F2**
 - *Show ST code of diagram* **F3** (after *Build*)
 - *Use last selected element from tree* **F4**
 - *Automatic connections* – while placing elements on the diagram, automatically creates connections between inputs and outputs at the same level and with power lines
 - *Create connections* – if clicked, supplements missing connections between inputs and outputs at the same level or with power lines (if needed).

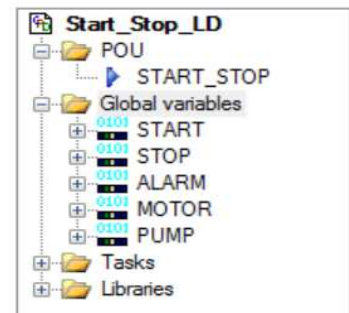
Remark. *Automatic connections* option speeds up creation of straight rungs, involving contacts, coils and blocks without EN (from *IEC_61131* library). *Create connections* typically connects remaining inputs and outputs of blocks and functions. Besides the two options one can always connect the elements manually.

Development steps

It is recommended to create LD diagram successively rung-after-rung, with verification of connections after each rung. Verification is also executed automatically every 10 minutes (if change detected, see *Automatic save* in *Other issues*).

Global variables

- Global variables are usually declared at the beginning of creating the project (see basic CPDev instruction).
- Declared global variables shown in the project tree are included into hint list of the editor.
- The list appears in *Variable properties* window (see below).



Rung

- Place contacts, coils, blocks, etc. on the drawing board. If *Automatic connections* option is enabled, connections between elements and power lines are created automatically giving a set of straight rungs. Temporary names are provided automatically.
- To transform straight rung into parallel branch of another main rung, remove connection to the right line, add branch point in the main and connect last element of the parallel branch with the branch point.
- Change temporary names of:
 - contacts and coils into names of global variables and target names of local variables
 - input and output variables as above
 - function block instances into target names.
- Write in values and types of constants. Make sure that types of input/output variables are consistent. Take printing into account.
- *Verify* connections of the rung (editor option).

Remark. Contacts and coils of P, N, R, S type can be chosen while changing names (see below).

Task. Compilation. Simulation

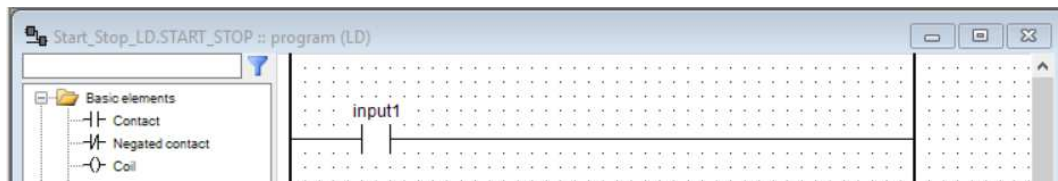
- Declare task, select cycle and assign program (programs) to the task.
- During compilation the open diagram is verified again and the whole project translated into ST language. If the ST translation does not have errors, it is compiled into VMASM code of virtual machine.
- Online operating mode and CPSim simulator provide tracing of variable values during execution.

First rung of START-STOP

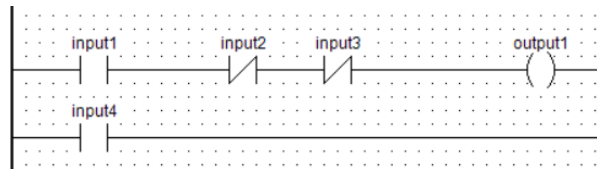
Two straight rungs (initially)

- **Place an element** on the diagram:
 - select the element in the tree on the left
 - drag it to the target location (or select and drop)

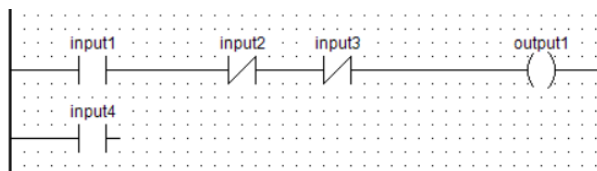
Automatic connections enabled



- place successive elements.

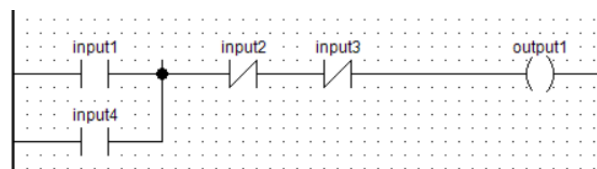
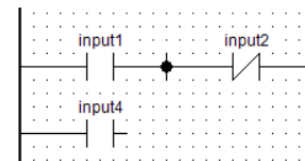


- **Remove connection** of the lower rung to the right line:
 - select the connection (becomes red)
 - press *Delete*.



Branch point and manual connection

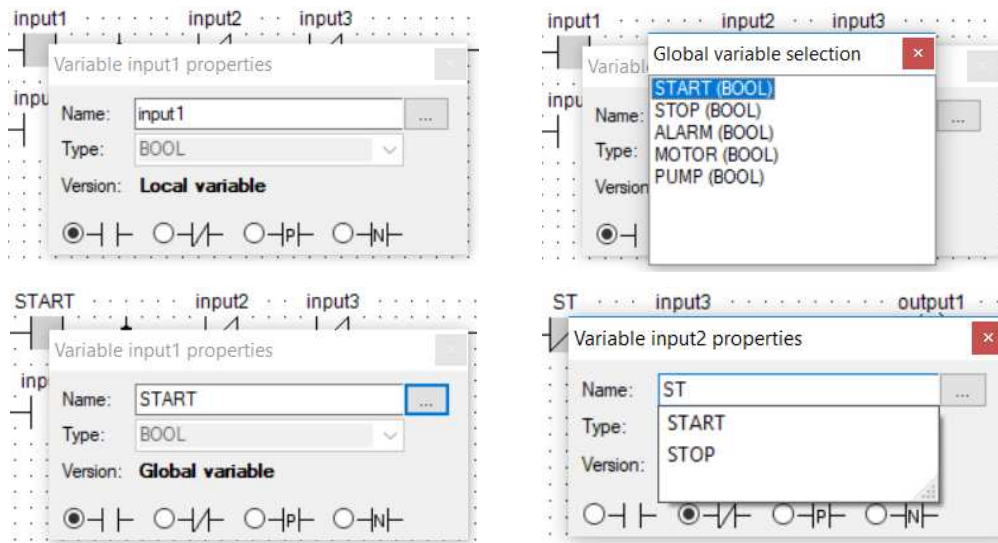
- **Right-click** at suitable place in the upper rung to add branch point with two tips.
- Connections can be drawn from input to output or from output to input.
- **Create connection** manually:
 - select or point at the input or output, which will be the beginning of the connection (tip color changes to red)
 - press left mouse button and drag cursor to the output or input which will be the end of the connection (while moving the cursor dashed line from the beginning is drawn)
 - release left mouse button what creates the connection with **path calculated and drawn automatically**.



Contacts and coils as variables

- *Variable properties* window allows to change **name** of variable associated with contact or coil from temporary name such as **input1** or **output1**. To open the window, **double-click** the element or select it and choose *Properties* option from context menu.

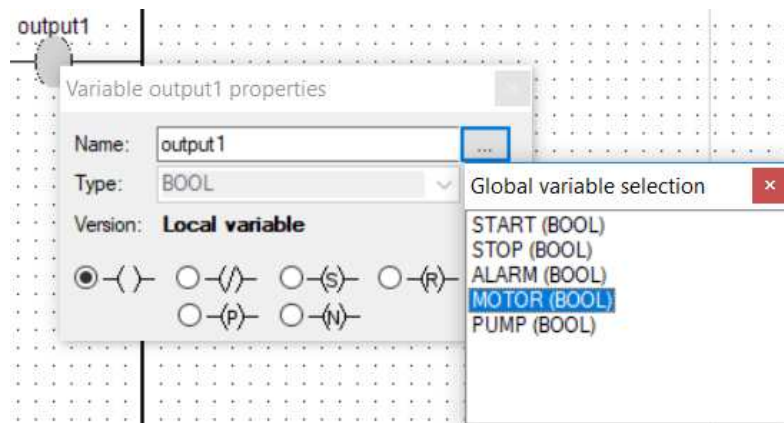
- If the project has been rebuilt after declaration of global variables, pressing the **ellipsis** button (...) or **F2** opens the new *Global variable selection* window where the user can choose the variable without typing. The choice is confirmed by pressing *Enter*, *Escape* or closing the window.



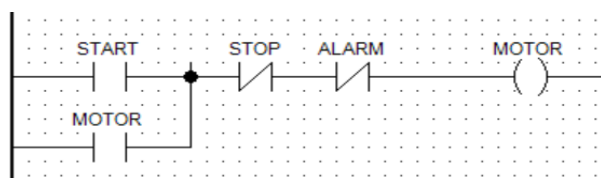
- Variable properties* window also provides **name prompting** functionality as an alternative to the ellipsis button or **F2**. This may be convenient for long global variable list.
- Variables with names other than in the global variable list automatically become local.

Remark. CPDev compiler does not distinguish lower-case and upper-case characters. So for instance *name1* and *NAME1* denote the same variable.

- Variable properties* window also allows to change **type** of contact or coil into standard, negated or P, N, R, S.

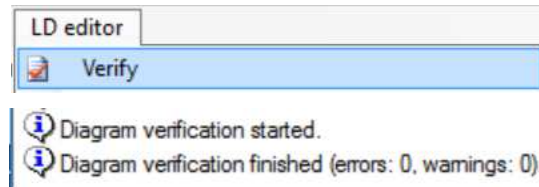


- To close *Variable properties* window, press *Enter*, *Escape* or closing button. First rung is now ready.



Verification

- Choose *Verify* option from the editor menu. It checks completeness of the diagram (connections) and consistency of types of inputs and outputs. Results are reported at the bottom of CPDev window.



- If LD-to-ST translation of the verified diagram is of interest, press **F3** (see below). Open diagram is automatically saved every 10 minutes into temporary directory followed by verification (see *Automatic save*).

Editing

- Selection:
 - select an element (becomes grey); successive elements are selected with *Ctrl* key pressed
 - alternatively, press left mouse button at any empty place on the diagram and surround the elements.
- Moving:
 - with element/elements selected, press and hold left mouse button while moving the cursor to new location
 - release the button.

Remarks. Element cannot be moved into new place if there is not enough room. It is shown in red and returns to previous place after releasing the button.

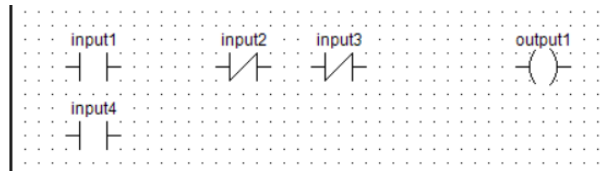
Moving an element (or elements) updates connections automatically **calculating relevant paths anew**. They may be somewhat different than before.

- Removing:
 - press *Delete* after selecting element/elements
 - individual elements can also be removed by right-click, selecting *Delete* in the window and left-click.
- Copy and paste:
 - select element/elements or a rung
 - copy (*Ctrl+C*)
 - move cursor to new location
 - paste (*Ctrl+V*).

Remark. Before copying a rung it is recommended to enable *Automatic connections*, as it supplements connections to power lines.

Automatic connections disabled

- With *Automatic connections* option disabled the elements placed on the drawing board are initially separated (keep horizontal alignment).

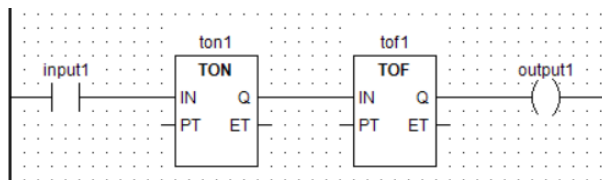


- Connections may be created manually as above or by clicking *Create connections* and making suitable corrections.

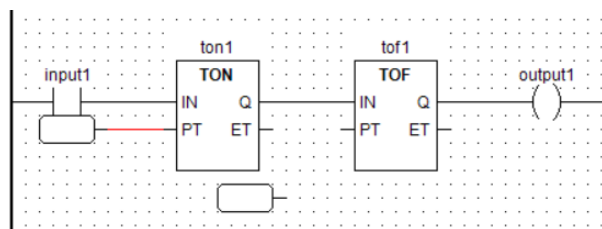
Second rung of START STOP

Elements and connections

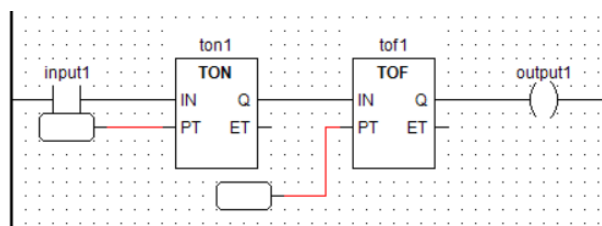
- Blocks from the standard *IEC_61131* library, such as TON and TOF, are placed in the rung **in the same way** as contacts or coils (*Automatic connections* enabled). Temporary instance name appears above the block.



- A constant placed on the diagram is automatically connected to block input at the same level. If there is no such input, it is left unconnected.



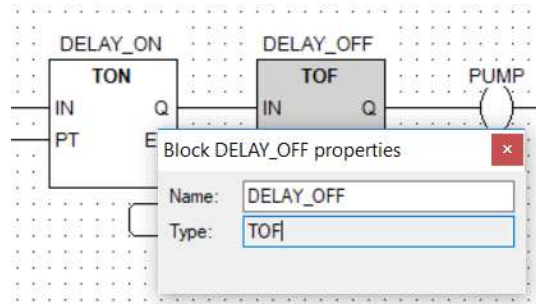
- Remaining connections are created **manually** as before (select input or output, drag cursor to another end with left button pressed, release the button).



- Connections between undefined or inconsistent types are shown in red.

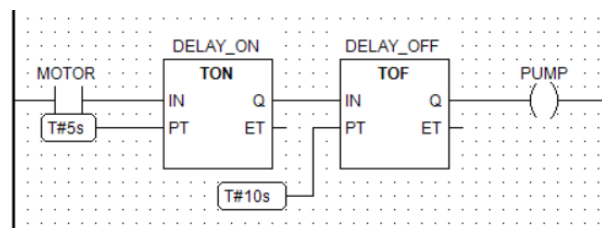
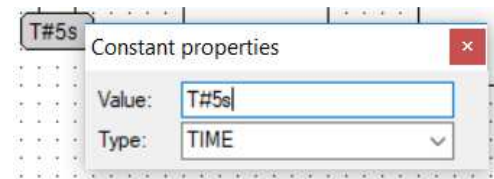
Instance names

- **Double-click** the block to change temporary name in *Block properties* window.
- Block width is updated automatically every time when length of the name is changed.

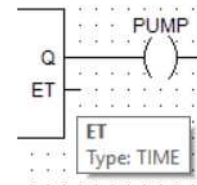


Constants

- **Double-click** the empty rectangle (rounded) to enter value and type in *Constant properties* window. Type may be prompted automatically according to value.
- Second rung assumes the final form.

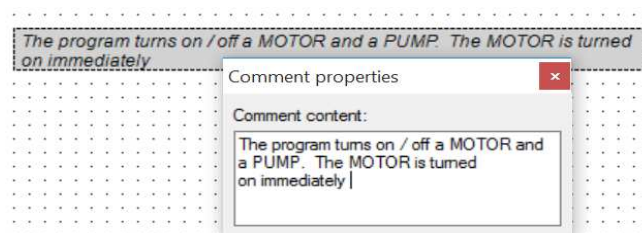


- For information, type of each input/output of any element is shown in tooltip after indicating the element's tip.



Comments

- After adding a comment to the diagram, double-click the empty rectangle. The window, where content can be defined, is presented.

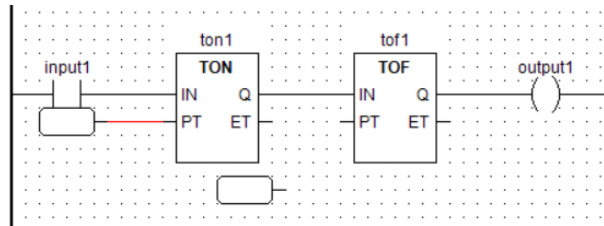


Verification

- *Verify* the diagram using the editor option. Press **F3**, if ST translation is of interest.

Automatic connections disabled

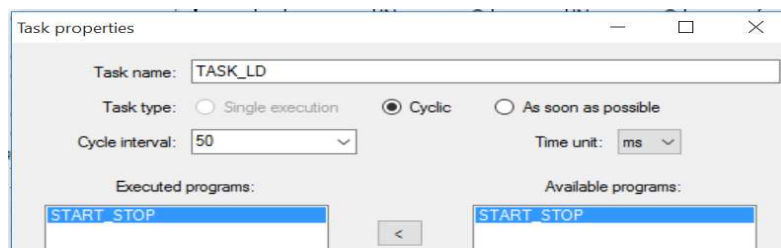
- The elements, initially separated, are connected manually as before. Alternatively, one may click *Create connections* to supplement missing connections of inputs to outputs at the same level or to power lines (if needed).



Task and compilation

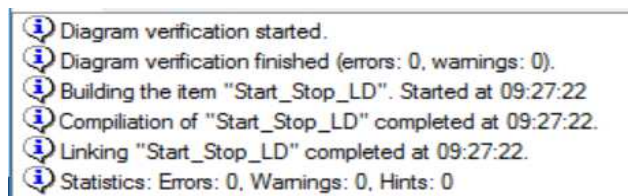
Task

- Task can be added to the project in two ways:
 - from project tree by *Tasks* → *Add task* or by *Start_Stop_LD* (project name) → *Add item* → *Task*
 - from CPDev main menu by *Project* → *Item* → *Add* → *Task* (in *Adding new element* window).
- Enter *Task name*, here *TASK_LD*, choose *Cycle interval* and *Executed programs* from *Available* ones.



Compilation – Build

- By clicking *Build* icon in the toolbar or selecting *Project* → *Build* in the menu the whole project is first translated into ST language and then compiled into VMASM code for virtual machine. If errors occur, compilation is stopped. Warnings are reported, however the program can still be compiled. The project may be eventually saved before compilation (*Environment options* → *Editing*).
- Verification runs automatically before compilation provided that editor window is open.



Remark. Program may be compiled even without task definition, however a warning will appear.

- ST code of the translated diagram may be shown by pressing **F3** (click the diagram first). It is split into parts corresponding to rungs.

```

Code in ST language

001 PROGRAM Start_Stop
002
003 VAR_EXTERNAL
004   START: BOOL;
005   STOP: BOOL;
006   ALARM: BOOL;
007   MOTOR: BOOL;
008   PUMP: BOOL;
009 END_VAR
010
011 VAR
012   out_contact_START_90_100 (*$HIDDENONLINE*) : BOOL;
013   out_contact_STOP_200_100 (*$HIDDENONLINE*) : BOOL;
014   out_contact_ALARM_270_100 (*$HIDDENONLINE*) : BOOL;
015   out_contact_MOTOR_90_150 (*$HIDDENONLINE*) : BOOL;
016   out_contact_MOTOR_70_210 (*$HIDDENONLINE*) : BOOL;
017   out_bp_110_90 (*$HIDDENONLINE*) : BOOL;
018   DELAY_ON: TON;
019   out_DELAY_ON_Q (*$HIDDENONLINE*) : BOOL;
020   DELAY_OFF: TOF;
021   out_DELAY_OFF_Q (*$HIDDENONLINE*) : BOOL;
022 END_VAR
023
024 out_contact_START_90_100 := START;
025 out_contact_MOTOR_90_150 := MOTOR;
026 out_bp_110_90 := out_contact_START_90_100 OR out_contact_MOTOR_90_150;
027 out_contact_STOP_200_100 := out_bp_110_90 AND NOT STOP;
028 out_contact_ALARM_270_100 := out_contact_STOP_200_100 AND NOT ALARM;
029 MOTOR := out_contact_ALARM_270_100;
030
031 out_contact_MOTOR_70_210 := MOTOR;
032 DELAY_ON(IN:=out_contact_MOTOR_70_210,PT:=T#5s,Q=>out_DELAY_ON_Q);
033 DELAY_OFF(IN:=out_DELAY_ON_Q,PT:=T#10s,Q=>out_DELAY_OFF_Q);
034 PUMP := out_DELAY_OFF_Q;
035
036 END_PROGRAM

```

- Local variables with (*\$HIDDENONLINE*) directives created automatically do not appear on variable value list in online mode.

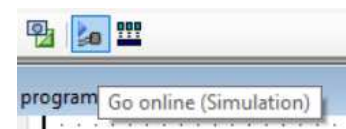
Online mode

Tracing execution of a project is provided by CPDev online mode. The project may consist of a number of programs written in the same or different languages. Tracing is particularly convenient for graphic languages, such as LD.

Go online

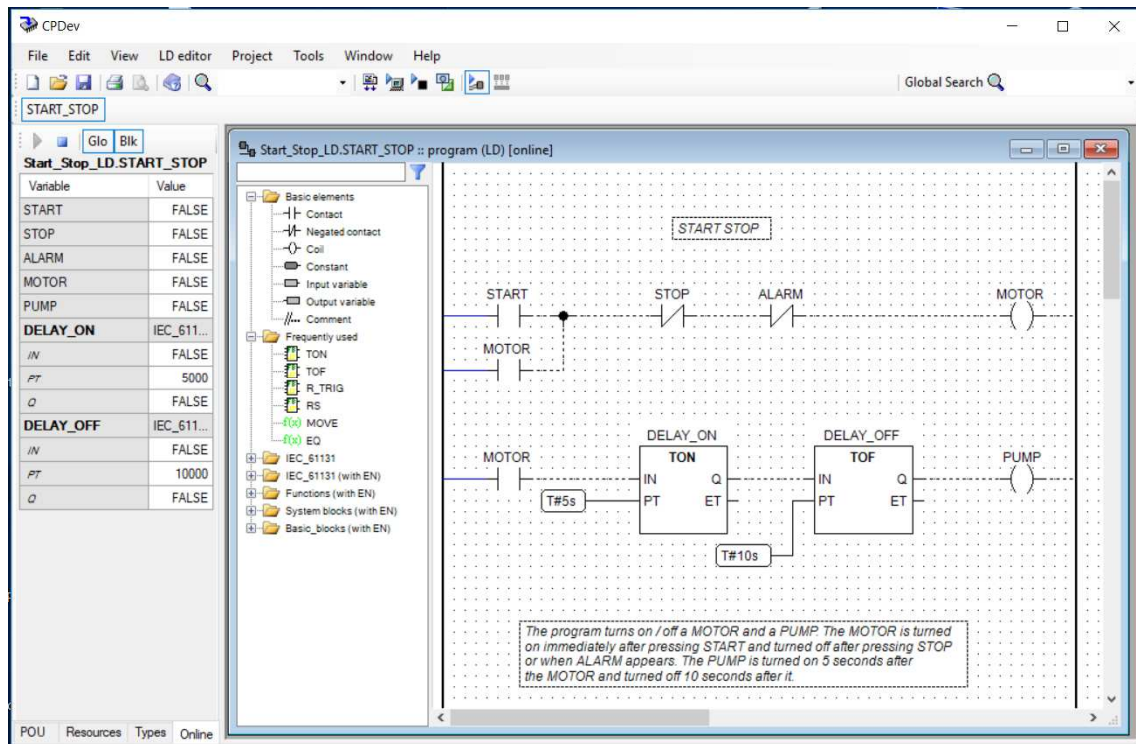
Before going into online mode, the project must be built (rebuilt after changes).

- Click *Go online* icon in the toolbar (becomes framed) or select *Project* → *Go online* in the menu.
- The icon indicates that *Data sources* are set to *Simulation*.

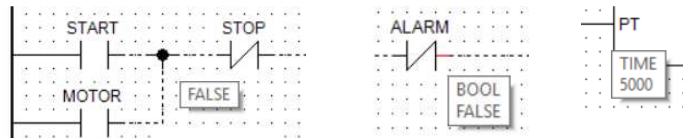


LD diagram online

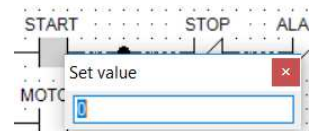
- CPDev window in online mode involves “live” LD diagram and list of variable values on the left. BOOL values are distinguished by continuous (TRUE) or dotted (FALSE) connections. Values of other types are written above connections, except connections to constants.



- Value may also be shown in tooltip after indicating a connection or tip of element. Type appears as well if tip is indicated (TIME in milliseconds).



- Change value:**
 - select input (contact, variable)
 - double-click
 - enter new value, accept or close the window.
- Change is reflected on the value list.
TRUE, FALSE may be entered as 1,0.
Constants cannot be changed in online mode.



- If a few diagrams are open, the one on the top is "live". It is indicated by [online] after name of the program.

Value list

- List with values involves variables associated with “live” diagram, so global and local variables, and function blocks with inputs/outputs. If no program is open, global variables are shown only.
- Change value:**
 - click current value (becomes blue)
 - enter new value, accept.
- Icons above the list provide:
 - start simulation again (if stopped before)
 - stop simulation temporarily (still in online mode)
 - show/hide global variables
 - show/hide inputs/outputs of function blocks.
 Stop freezes current values and LD diagram. Start resumes online operation.
- The open list is associated with appearing *Online* tab at the bottom. Clicking *POU* tab on the left opens project tree, for instance to choose another program for tracing. Then click *Online* again.

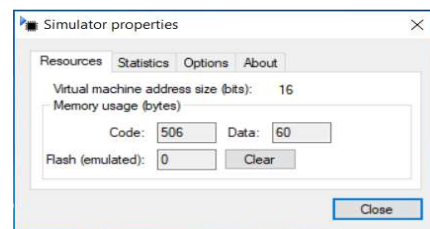
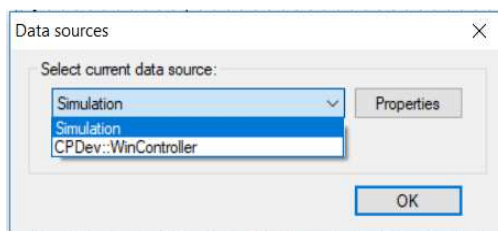
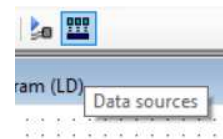
Variable	Value
START	TRUE



Start_Stop_LD.START_STOP	
Variable	Value
START	FALSE
STOP	FALSE
ALARM	FALSE
MOTOR	FALSE
PUMP	FALSE
DELAY_ON	
IN	FALSE
PT	5000
Q	FALSE
DELAY_OFF	
IN	FALSE
PT	10000
Q	FALSE

Data sources

- Clicking *Data sources* icon shows data sources for online mode, beginning with *Simulation*. Other sources, if shown, represent specific controllers for which online mode means *commissioning*. They may involve WinController, an advanced CPDev runtime for Windows.

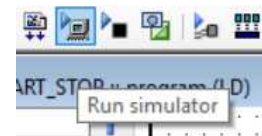


- Some data of simulated project are shown in *Simulator properties* window.

CPSim simulator

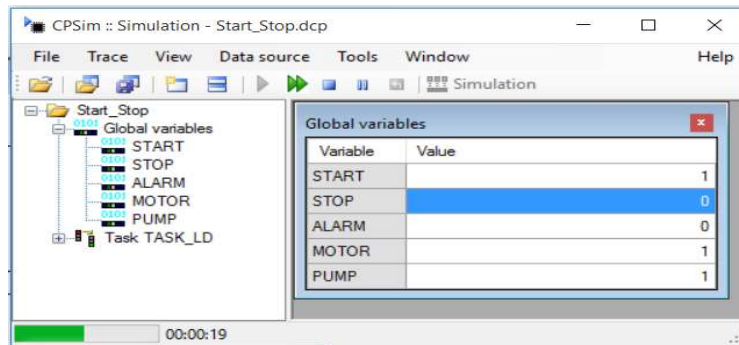
Another way of tracing variable values during execution is provided by CPSim simulator. It is more flexible than online mode allowing to assign most important variables to panels with buttons and “LEDs”, as well as to individual views or custom lists.

- CPSim is run by clicking *Run simulator* icon in the toolbar or by selecting *Project* → *Run simulator* (also *Tools* → *Simulator*).



Global variables

- CPSim window presents initially the list of global variables. Execution begins by clicking *Start trace* icon or by choosing *Trace → Start*.

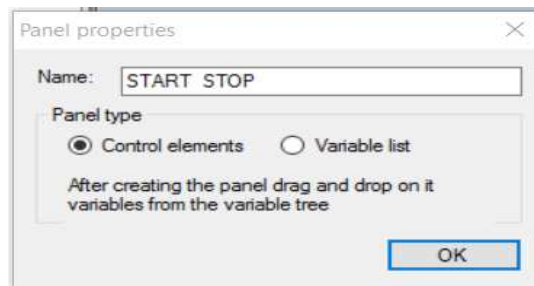


- Values of those global variables which are inputs to the program can be changed. More on CPSim can be found in *Help → Programming instruction*.

Remark. Values of boolean variables are displayed as 0 (FALSE) and 1 (TRUE).

Control panel

- More convenient way of testing is provided by selecting *Group panel* icon or relevant option in *View*.



- Type name in *Panel properties* window, accept and drag required variables from simulation tree on the left into the rectangle panel (which grows accordingly).



- Test the program by pressing input buttons (BOOL) or by typing values in input cells (also other types).

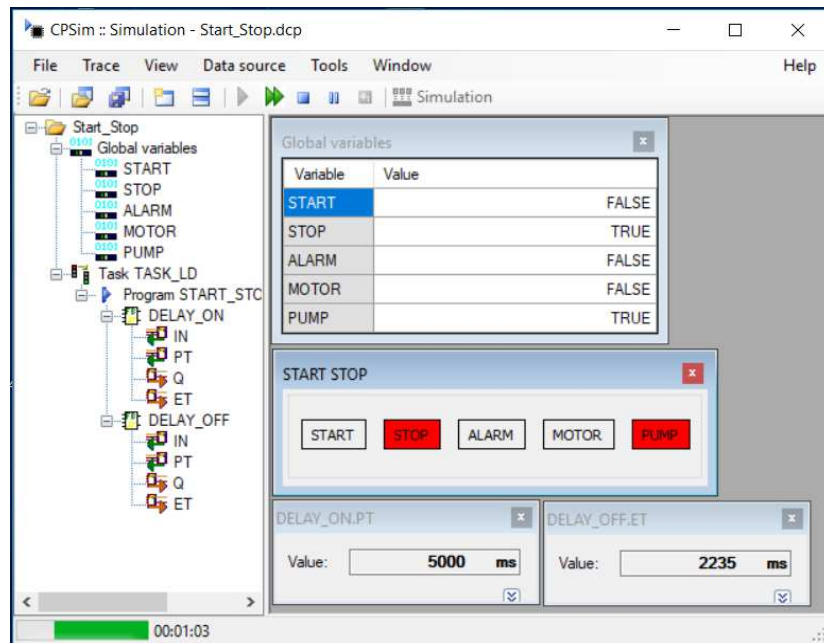
Individual view

- Single variable can be traced by dragging it from the tree into view area. Individual views are used for important local variables and inputs/outputs of function blocks (PT here).



Variable list

- By choosing *Variable list* in *Panel properties* window a list of variables dragged from the tree can be created. Inputs/outputs of function blocks or local variables used in the diagram (if any) are usually assigned to such list (which looks the same as the list of global variables).



Sessions

- Lists, panels and views created for simulations can be saved for future use by selecting *Trace* → *Save session* or by Yes answer to the question *Do you want to save the trace session?* asked on CPSim exit. Session data are saved in the project folder as SCP file.
- Complex project may require a few simulation sessions with different views. Data of each session should be saved in separate file and run by *Trace* > *Open session...*

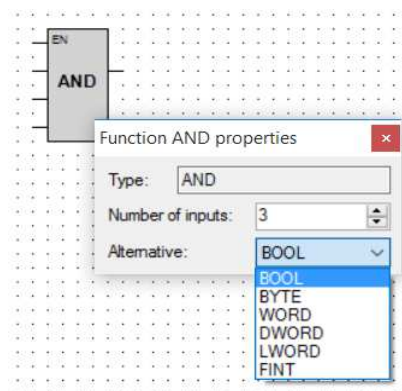
Elements with EN

Rung conditions

- Rung may involve only **one element with EN**, i.e. block or function dragged from the editor tree.
- Connection rules are given in *Connections without or with EN* (earlier).

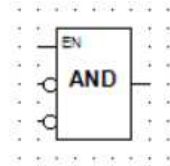
Functions

- Number of inputs** of some functions can be selected as follows:
 - open *Function properties* window (double-click)
 - choose suitable number of inputs (2 ÷ 15).
- Number of inputs can also be incremented or decremented by selecting the function on the diagram and pressing] or [bracket key (+1 or -1).
- Alternative type** of variable can be selected from the list opened by clicking default entry.



Input/output negation

- **Right-click** particular input or output of function or function block to negate it (boolean signals only). EN input can be negated as well.

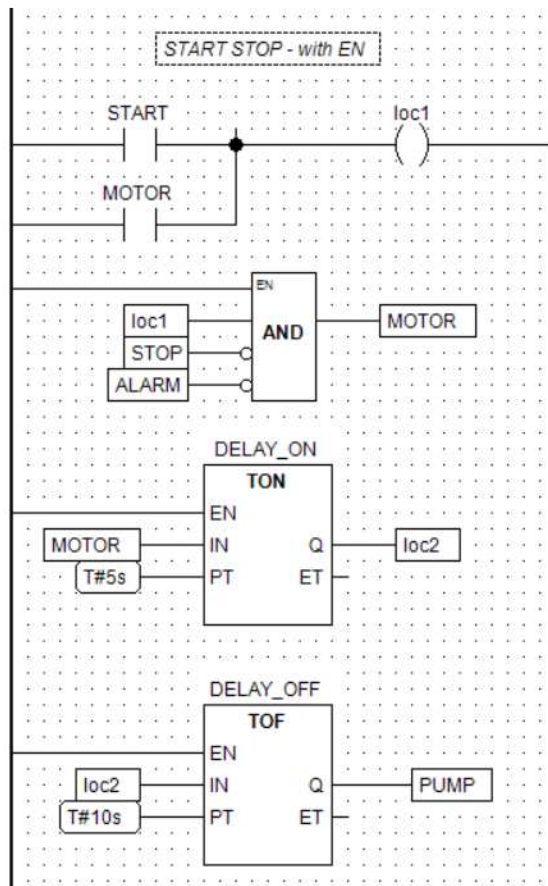


Manual connections

- Manual connections are preferred for a rung involving element *with EN* because inputs and outputs of such element, except EN, must be connected directly to other inputs, outputs or constants (not to contacts, coils or power lines). Hence *Automatic connections* option of the editor should be disabled.

START STOP with EN

- Comments:
 - local variables *loc1*, *loc2* transfer data between the rungs
 - function and blocks *with EN* occupy one rung each
 - except EN, inputs of the function and blocks must be connected to variables or constants (not to contacts)
 - output of the function and first outputs of the blocks must be connected to variables; other outputs may remain unconnected
 - temporary variable names are changed in *Variable properties* window (double-click)
 - *Create connections* option is preferred, with some manual corrections.



Other issues

Changing LD program name


Program name given initially needs to be changed sometimes in the final version.

- To change program name:
 - open the diagram (if closed)
 - select the program in the project tree
 - choose *Project* → *Items* → *Rename*
 - enter new name in the tree
 - accept (*Enter*)
 - build and save the project.

Library warnings

When library is linked to a project, CPDev automatically stores library version, timestamp and number of compiler version which created the library (LCP file). When the project is open again and built, those markers are compared with markers of actual library and compiler. Inconsistencies trigger warnings.

- The following warning indicates outdated library:

 Declared library timestamp *date* is not equal to timestamp found in library *name* from file "C:\ *location*"

If library content and compiler version have not been changed, ignore the warning and proceed with the project. Warning will not appear on next opening.

Otherwise:

- remove the library from the project tree by selecting the library and choosing *Remove* from context menu
- import actual library into the project by *Project* → *Import* → *Library* from appropriate file.
- New version of CPDev compiler includes recompiled system libraries, i.e. *IEC_61131.lcp* and *Basic_blocks.lcp*. So only user library, if any, needs recompilation, export as LCP file into given location (*Project* → *Export* → *Library*) and import into the project as above.

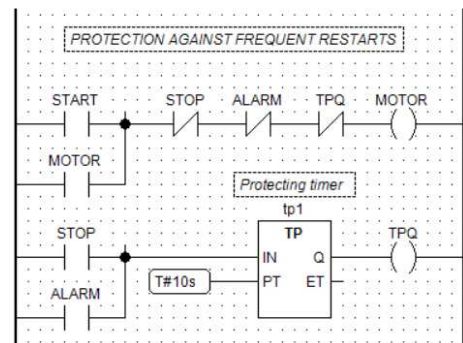
Automatic save

- Every 10 minutes (default) CPDev creates a file in **temporary directory** (see operating system settings) called *CPDev_ProjectNameOrFileName_EightHexNumbers.xml* with source code of the project. This is a back-up in case of CPDev crash.
- If LD diagram has been changed within that period, it is automatically verified (if open).
- Automatic save period may be changed in *Environment options* → *Editing*.

Standard examples

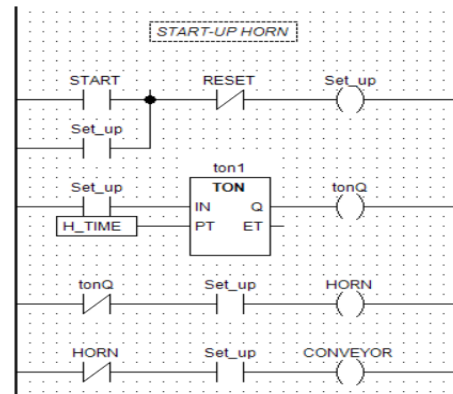
Protection against frequent restarts

- After STOP or ALARM (thermic sensor) another restart possible only after 10 seconds.
- CPSim: control panel and individual view of ET output (*elapsed time*).



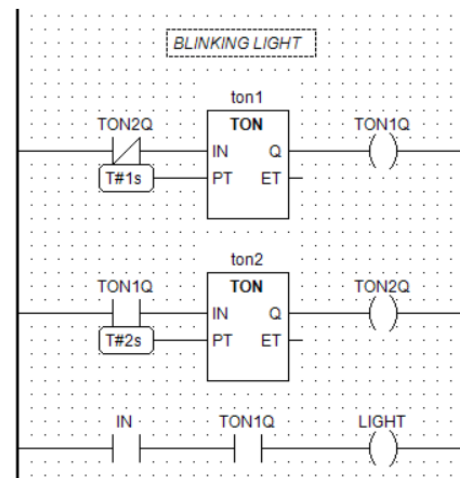
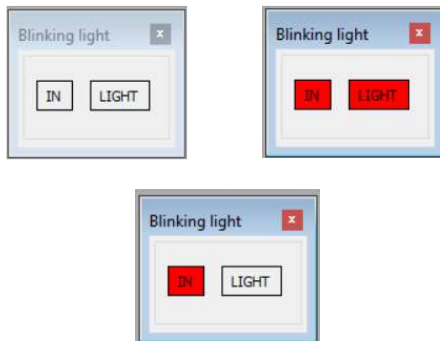
Start-up horn

- START turns HORN on for time set by H_TIME (10s). Then CONVEYOR begins to move.
- CPSim: control panel while HORN is on.



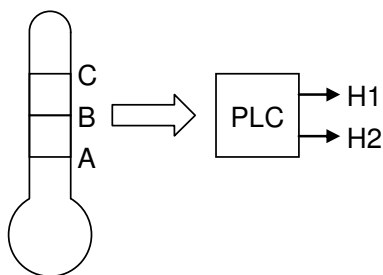
Blinking light

- LIGHT blinks when IN is on. The two timers implement a pulse generator (oscillator) operating continuously.
- CPSim: IN and LIGHT at different steps.



Two heaters

- Equipment set-up



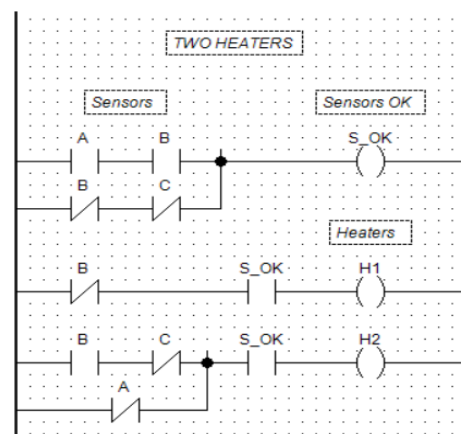
Sensors A, B, C

0 – temperature below the level

1 – temperature above or at the level

Heaters H1, H2

0 – off, 1 – on

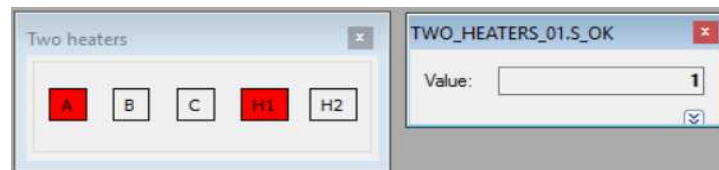


- Control task (temperature)

$t < A$	→	H1	H2	– both heaters on
$A \leq t < B$		H1	-	
$B \leq t < C$		-	H2	
$C \leq t$		-	-	
- Sensors OK

$$CBA \rightarrow 000, 001, 011, 111 \rightarrow S_OK = AB + \overline{B}\overline{C}$$
- Solution (e.g. Karnaugh):

$$H1 = \overline{B} \cdot S_OK, \quad H2 = (\overline{A} + \overline{B}\overline{C}) \cdot S_OK$$
- CPSim: control panel and individual view of S_OK.

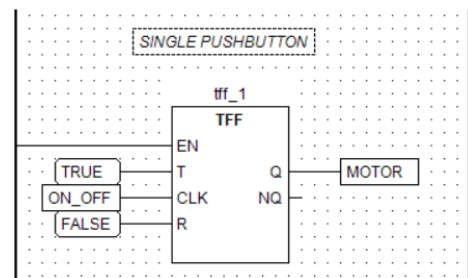
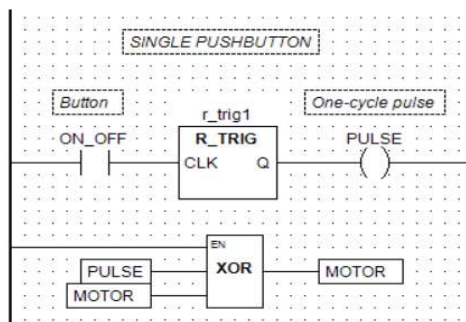
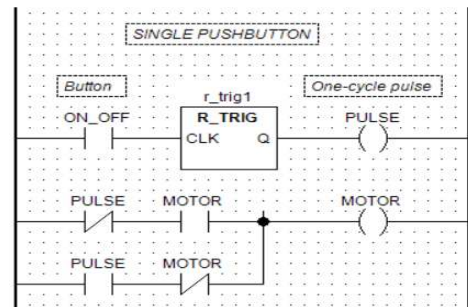


Examples with EN

Single pushbutton

Second and third diagram, not the first one, include elements *with EN*.

- First pressing ON_OFF turns MOTOR on and another turns it off.
- Implementations:
 - standard
 - XOR gate instead of contacts
 - TFF flip-flop from *Basic_blocks* library.

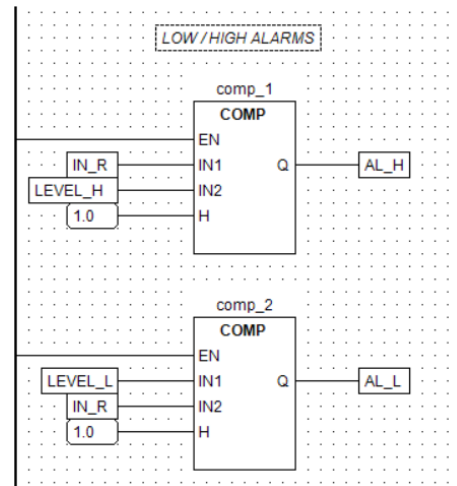


- CPSim: ON_OFF input and MOTOR at different steps.



Low/High alarms

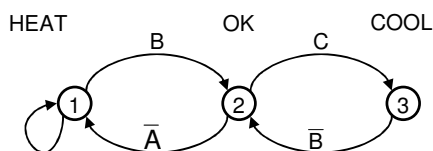
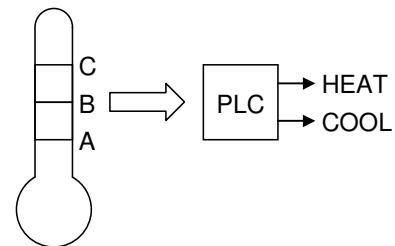
- AL_H or AL_L arise when IN_R (REAL) violates LEVEL_H (90%) or LEVEL_L (10%) alarm level. COMPArators are set with 1% hysteresis (*Basic_blocks*).
- CPSim: alarms for IN set to 20% or 91%.



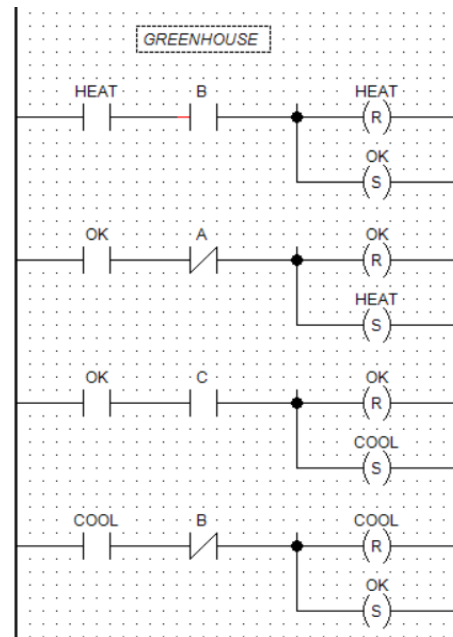
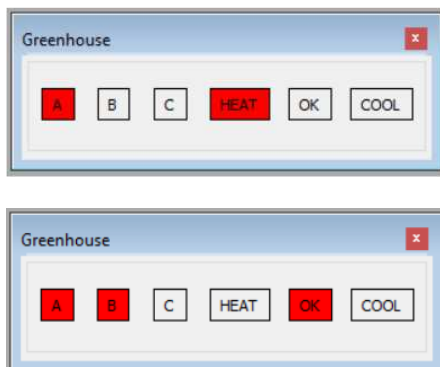
Sequential controls

Greenhouse

- PLC keeps temperature in a greenhouse between low A and high C, preferably near middle B. If the temperature drops below A, heater HEAT is turned on. It is turned off when the temperature reaches B. Fan COOL is turned on when the temperature exceeds C and turned off when it drops below B.
- State diagram

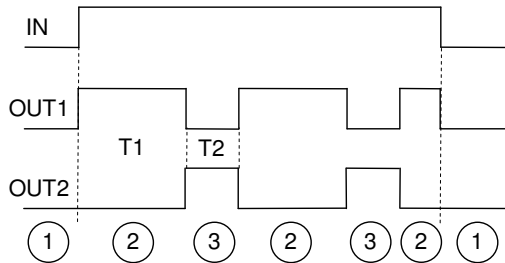


- Implementation involves R, S coils. Rungs correspond to transitions in the state diagram.
- CPSim: HEAT and OK states.

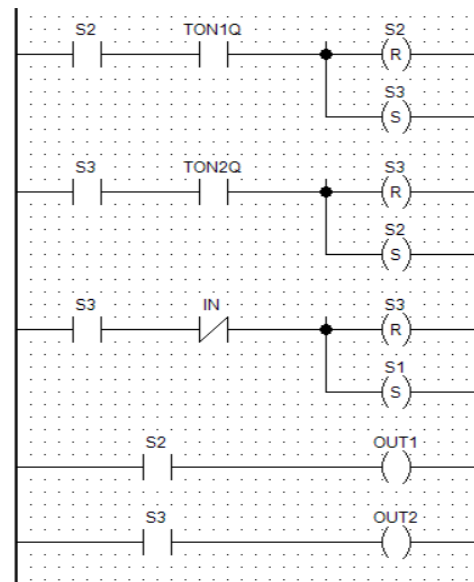
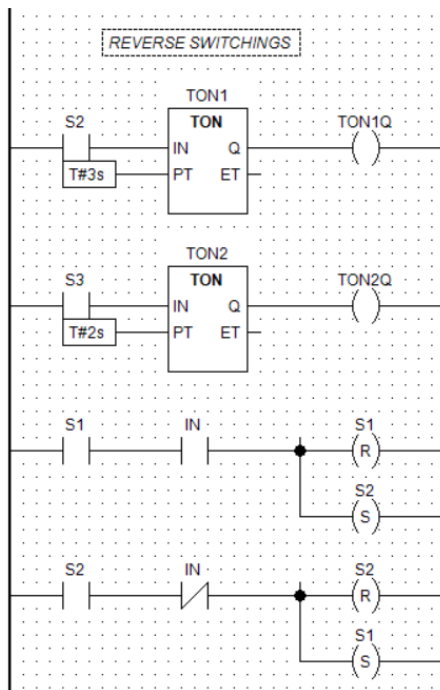
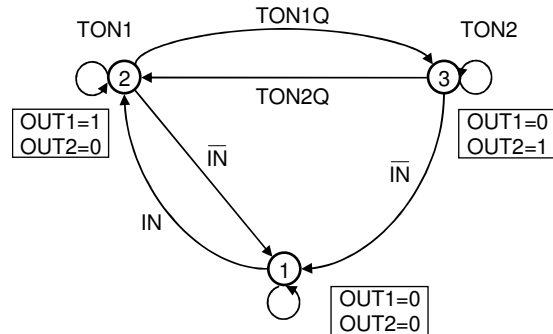


Reverse switching

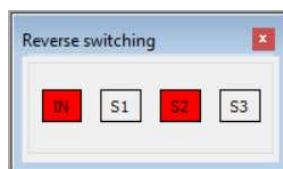
- If IN is on then OUT1 is on and OUT2 is off, or conversely. OUT1 is on for time T1 (3s), OUT2 for T2 (2s). If IN is off, OUT1 and OUT2 are also off.



- State diagram

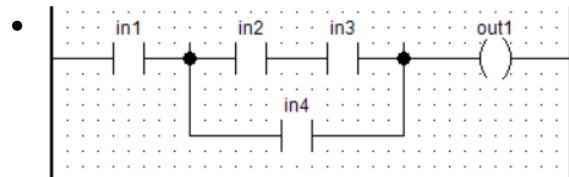


- CPSim: states S1, S2, S3 of the sequence.

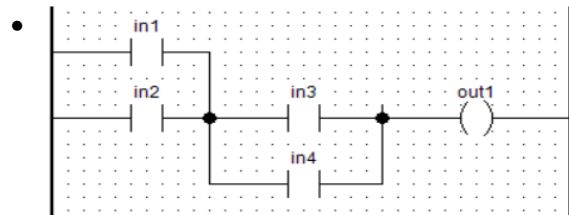


More on connections

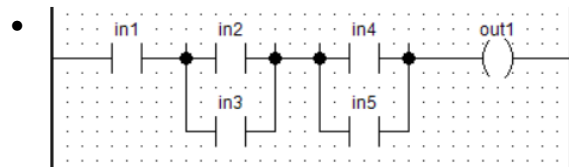
Parallel connections



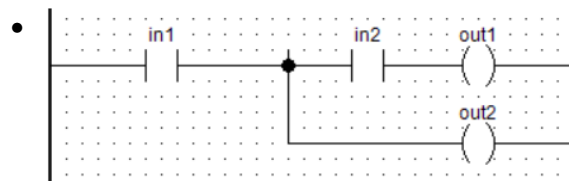
– $\text{out1} = \text{in1} \text{ AND } (\text{in2} \text{ AND } \text{in3} \text{ OR } \text{in4})$



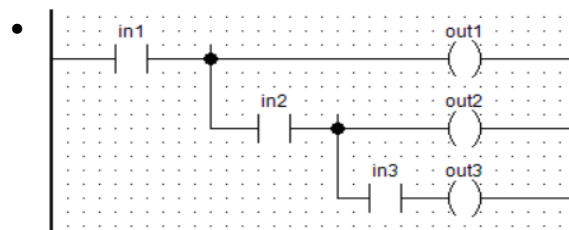
– $\text{out1} = (\text{in1} \text{ OR } \text{in2}) \text{ AND } (\text{in3} \text{ OR } \text{in4})$



– $\text{out1} = \text{in1} \text{ AND } (\text{in2} \text{ OR } \text{in3}) \text{ AND } (\text{in4} \text{ OR } \text{in5})$

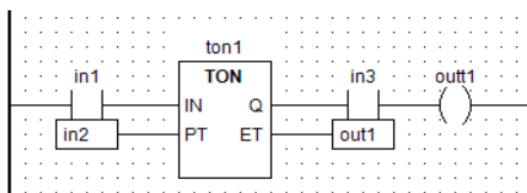


– $\text{out1} = \text{in1} \text{ AND } \text{in2}, \text{ out2} = \text{in1}$



– $\text{out1} = \text{in1}, \text{ out2} = \text{in1} \text{ AND } \text{in2}$
 $\text{out3} = \text{in1} \text{ AND } \text{in2} \text{ AND } \text{in3}$

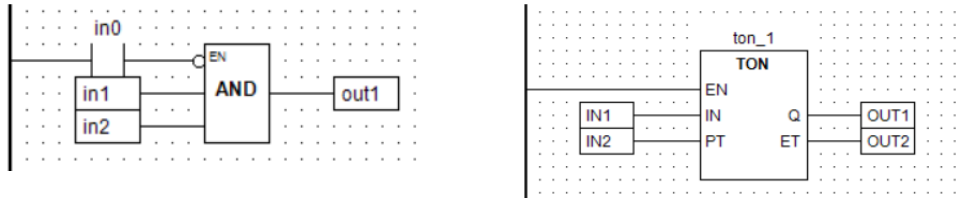
Function block without EN



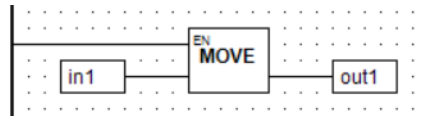
– block surrounded by contacts inside rung

Warnings for EN

- Warnings *No contacts/coils exist* may appear after verification of diagram involving rungs merely with elements *with EN*. They indicate that **connections are correct**, although untypical for standard LD diagrams.



- MOVE function provides **assignment** of input to output variable.



Wrong connections

- input variable mixed with contact and coil
- direct connection of input variable to output variable (use MOVE instead)
- second input of function (*with EN*) connected to contact (input variable or constant required)
 - third input not connected directly to input variable
 - output not connected directly to output variable
- trivial errors

Printing

Size adjustment

- Grey lines partition total work area of the editor into printed pages. Final size of the diagram may be decided before printing to fill in the pages conveniently.

Remark. Diagrams presented above are fairly tight (for consistency). They may be widened taking into account the grey lines, as the START_STOP printout shown farther.

Page setup

- Choose *Tools* → *Environment options* → *Configuration* → *Page setup*.

Configuration

Projects Editing Colors Miscellaneous Compiler Help Page setup

Border margins

Left (cm): 0.5 Right (cm): 0.5

Top (cm): 0.5 Bottom (cm): 0.5

☒ Draw border Padding (cm): 0.3

Paper/Orientation

Paper size: A4

☒ Portrait ☐ Landscape

Printing scale

User scale

User scale (%): 100

☒ Draw title block

Title block fields

☒ Subject ☒ Company

☒ Project name ☒ Project version ☒ POU name ☒ Author

☒ Filename ☒ Date ☒ Page number

☒ Update settings upon printing session

OK Cancel

- Upper part of the options involves *Border margins*, *Paper size*, *Orientation* and *Printing scale* (independent of the editor scale).
- Number of pages being printed is determined automatically according to the size of the diagram and printing scale (user selected). Small diagrams need one page, large ones a few.
- Lower part specifies what fields appear in the title block (table) printed at the end of each page. Table changes somewhat if some fields are not selected.

Subject: Program turns motor and pump on and off			Company: Rzeszow University of Technology	
Project: Start_Stop_LD	Version: v.1	POU: START_STOP	Author: Jan Nowak	
File: C:\Users\Leszek Trybus\Documents\CPDev - programy, instrukcje\Projekty...			Date: 08.02.2018 11:20	Page: 1 / 1

- Data for *Subject*, *Company*, *Project*, *Version*, *Author* and *File* are copied from *Project properties* window. Active window determines *POU* name.

Remark. Options of *Page setup* are kept in CPDev database (*Environment options*), so they apply to each printing till changed.

Print

- Choose *File* → *Print* in the main menu or press **Ctrl+P** keys or icon in the toolbar (direct print).
- Select printer and preferences in printer selection window. Change of preferences and printout adjusts grey lines and *Page setup* options accordingly.
- Press *Print*.

Remark. Printing to virtual printer such as PDF Creator or Microsoft Print to PDF is convenient way to verify expected printout. Vertical orientation (portrait) suits LD, SFC and horizontal (landscape) FBD.

