

PENKO Engineering B.V.

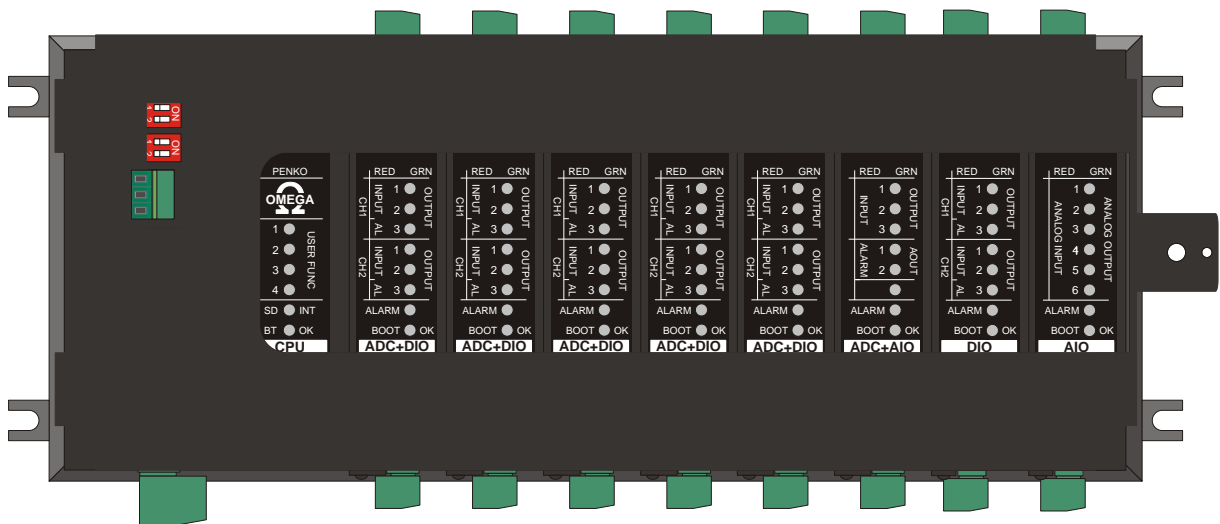
Your Partner for Fully Engineered Factory Solutions

Getting started with CodeSys



CODESYS

&



PENKO

an ETC Company

Getting started with CodeSys

CONTENTS

Pre-requirements.....	3
Step 1: Prepare the development environment.....	4
A) Open the environment	4
B) Install the omega device in the device repository	4
Step 2: Creating a new project.....	5
A) Create a new project	5
B) Identify the project	5
C) Select the Omega device	6
Step 3: Write your first application.....	6
A) Install required library	7
B) Add the installed library to the current project.....	8
C) Library documentation	9
D) Select PLC_PRG (PRG).....	9
E) Write a blink program: example 1	10
Step 4: Run your program.....	11
A) Select the Omega device	11
B) Download the program to the device.....	12
C) Program in running mode.....	13
Write a blink program: example 2	14
Appendix I: Blink program example 1	15
Appendix II: Blink program example 2.....	16

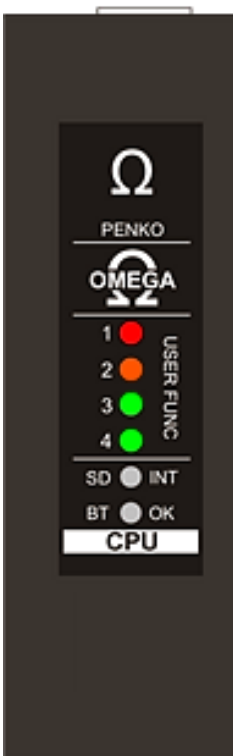
Getting started with CodeSys

PRE-REQUIREMENTS

This tutorial is designed to create your first PLC application by using CodeSys and a Penko device. There is no need for previously proven software development skills to successfully complete this tutorial. All that is required is listed below.

- ✓ Installed the CodeSys IDE V3.5 or higher from store.codesys.com on your computer. The CodeSys store requires registration before you can download the software.
- ✓ Omega device with CodeSys license up and running in your office or factory network¹.
- ✓ Download the Penko BSP Omega library² from penko.com.
- ✓ Download the Omega device description file from penko.com (PENKO Omega device.xml)

The goal is to create, upload and run a blink application in the Omega. This tutorial provides two different example applications for controlling the LED's. There are four user function LED's on the Omega CPU card. Each LED can either be off, red, green, or yellow.



¹ Having trouble with the getting the Omega up and running? Consult the omega manual at penko.com for this.

² board support package (BSP)

Getting started with CodeSys

STEP 1: PREPARE THE DEVELOPMENT ENVIRONMENT

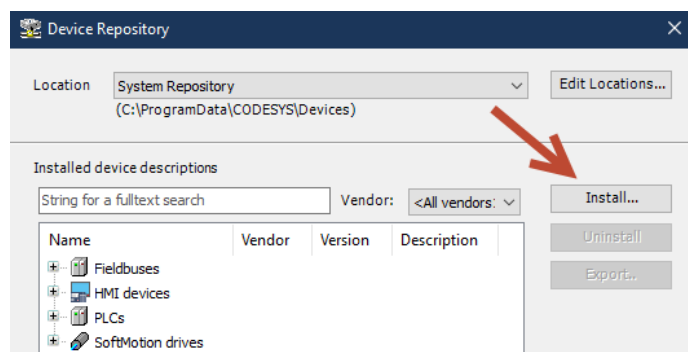
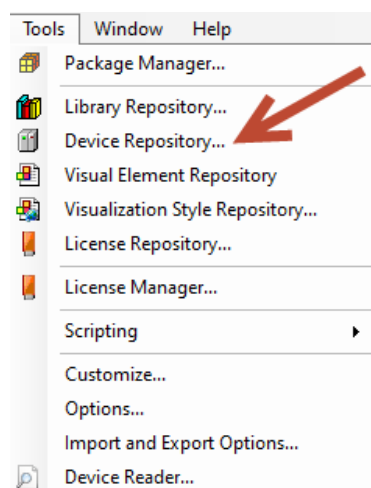
Install the CodeSys environment as downloaded from the CodeSys store onto your PC. This is needed to start writing program code for the Omega CodeSys PLC.

A) OPEN THE ENVIRONMENT

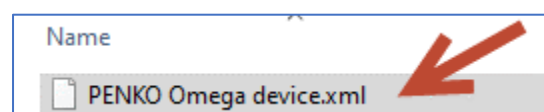
Open your installed CodeSys IDE (programming environment) version.



B) INSTALL THE OMEGA DEVICE IN THE DEVICE REPOSITORY

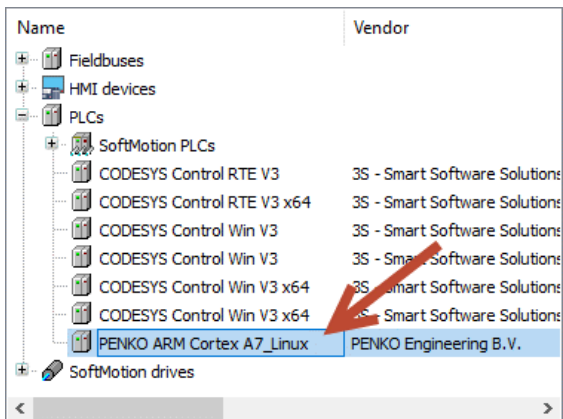


Select the "PENKO Omega device.xml" file.



Getting started with CodeSys


Now the Omega has been added as a device to the CodeSys IDE.






Close this window to go back to the main screen.

STEP 2: CREATING A NEW PROJECT

A) CREATE A NEW PROJECT

 CODESYS V3.5 SP15 Patch 2

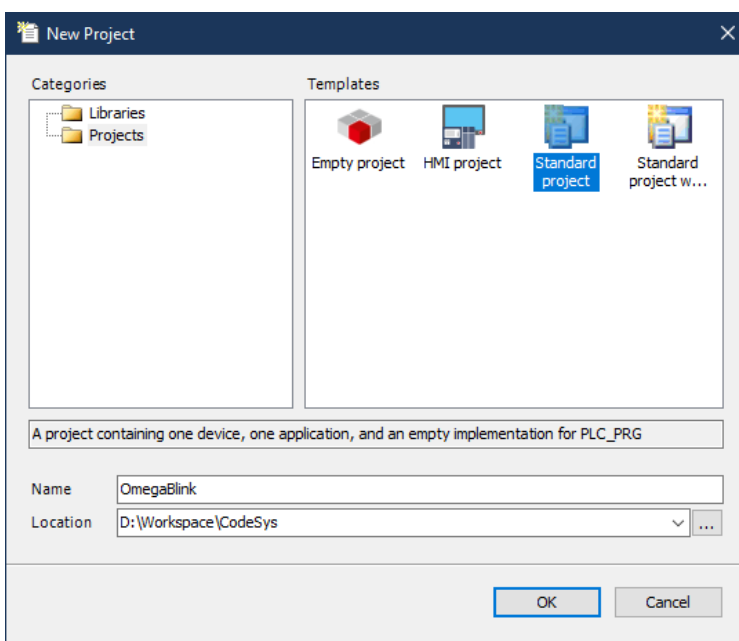
Basic operations

-  New Project...
-  Open Project...
-  Open Project from PLC...

Recent projects

B) IDENTIFY THE PROJECT

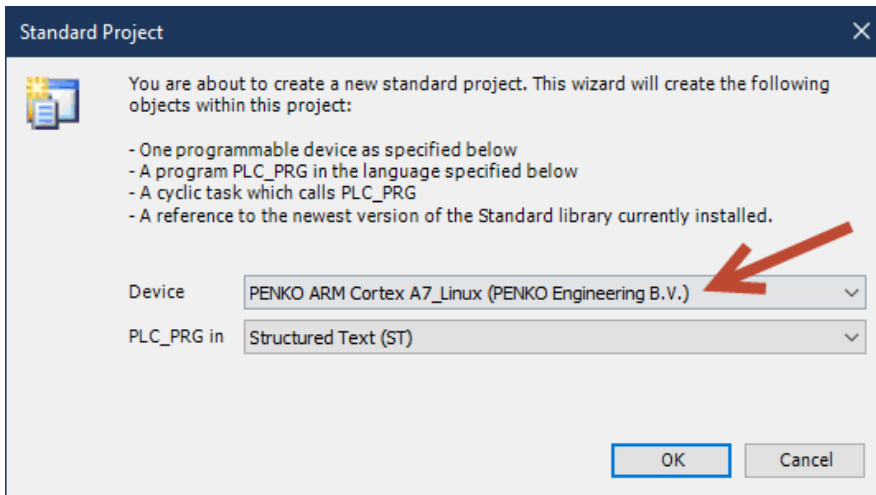
For this tutorial we use the "standard project". Select this option and fill-in an appropriate project name and folder location.



Getting started with CodeSys

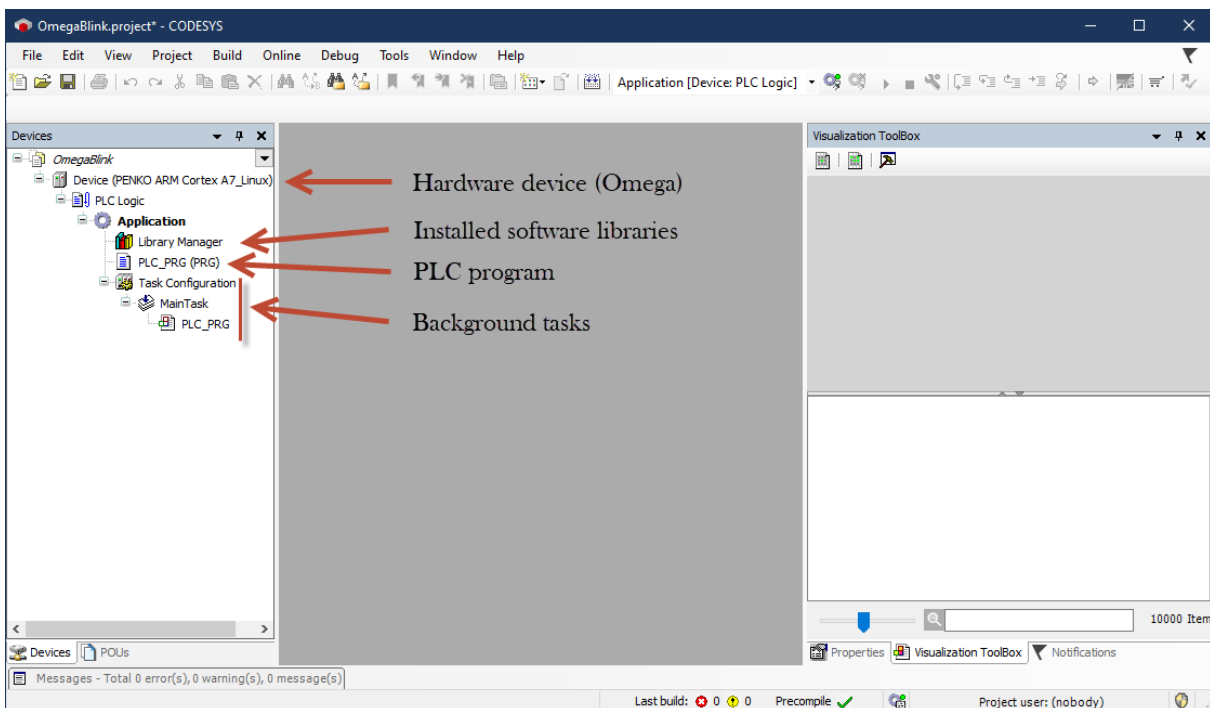
C) SELECT THE OMEGA DEVICE

Select the Penko omega device and keep the PLC_PRG at “Structured Text (ST)”.



STEP 3: WRITE YOUR FIRST APPLICATION

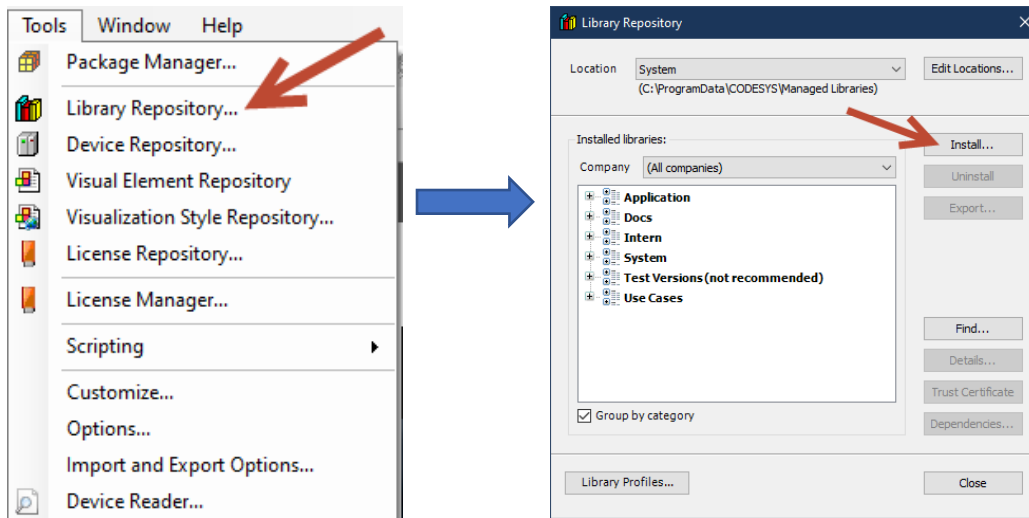
After preparing the environment, the project will open in an Integrated Development Environment (IDE) where the application can be created. Below the initial screen with several items added by default. The menu bar on the left gives easy access to everything needed to create your program.



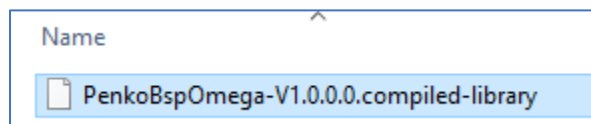
Getting started with CodeSys

A) INSTALL REQUIRED LIBRARY

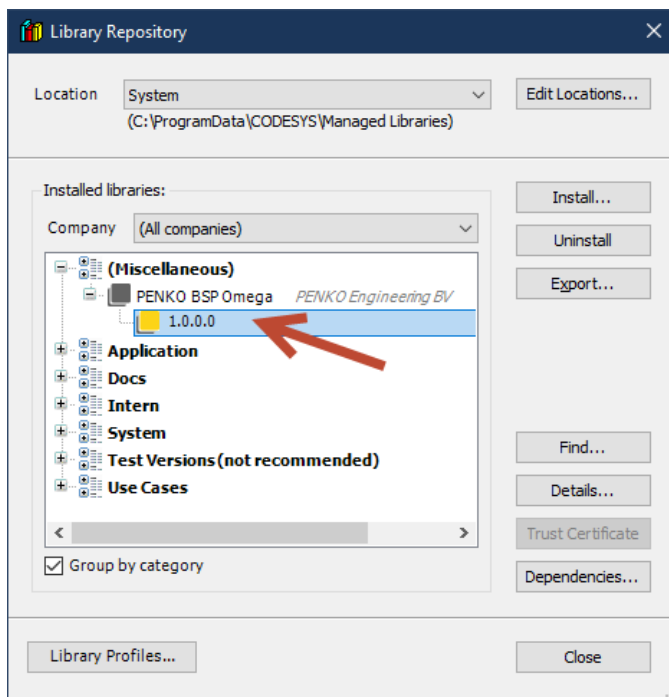
Before we can start programming, the required Penko library to control the LED's must be installed. Go under "Tools" to "Library Repository" and hit the install button.



Select the downloaded Penko BSP Omega library file.



After the library has been installed it appears in the library list under the "Miscellaneous" category.

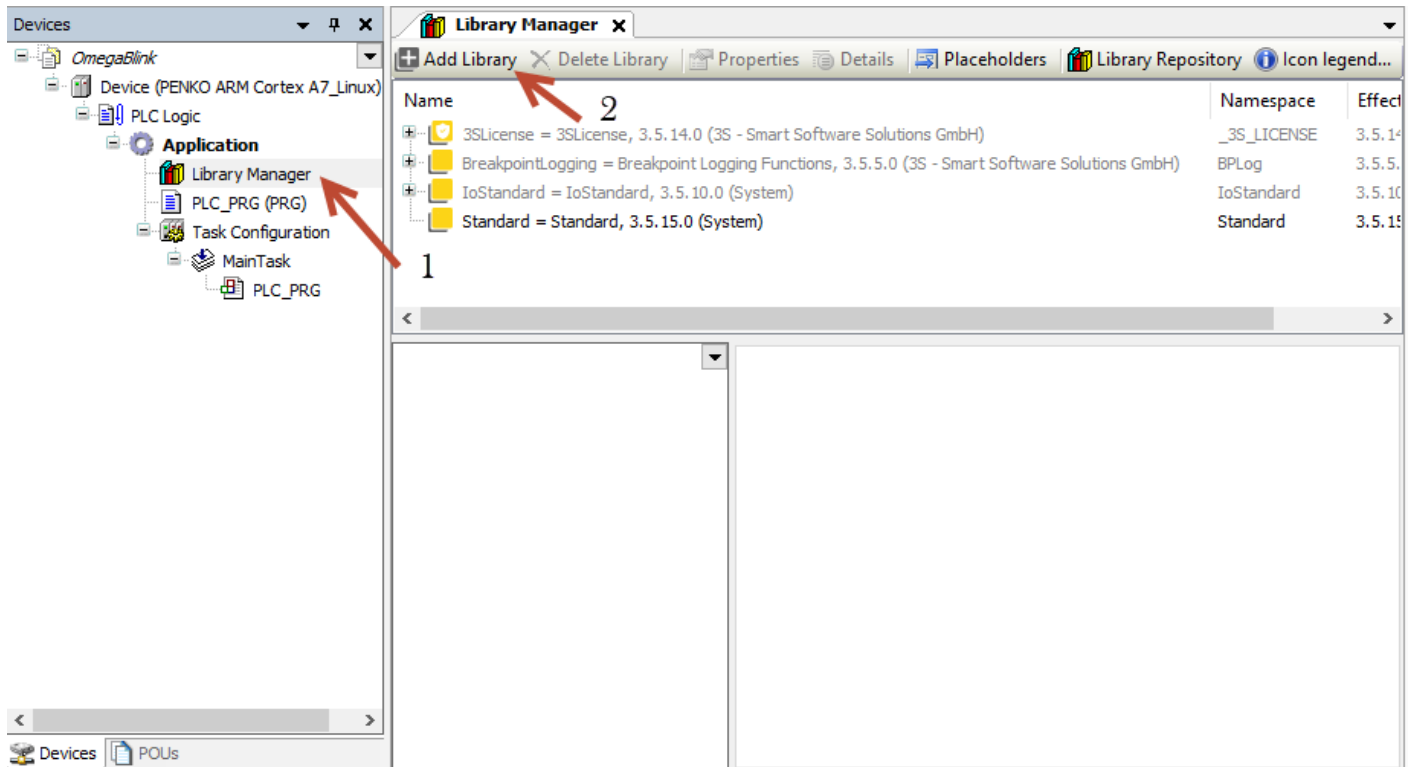


Close this window.

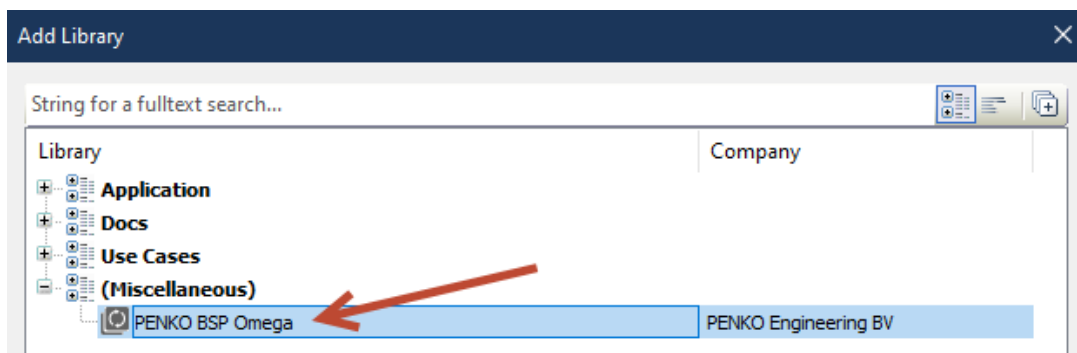
Getting started with CodeSys

B) ADD THE INSTALLED LIBRARY TO THE CURRENT PROJECT

Now add the just installed library to this project. To do this, go to the library manager (1) and hit the “Add library” button (2).



Select under the “(Miscellaneous)” category the “Penko BSP Omega” library.

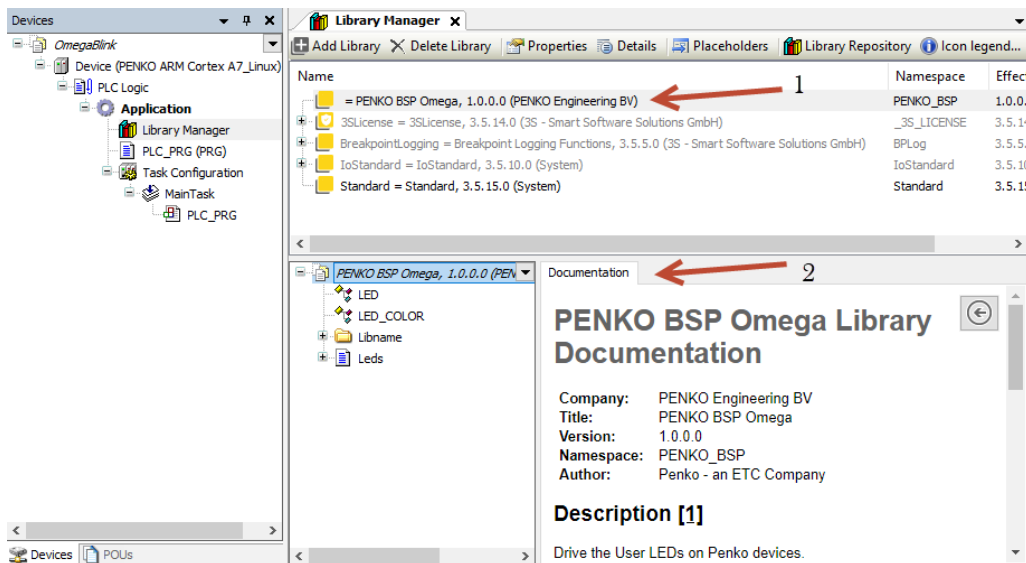


Getting started with CodeSys

C) LIBRARY DOCUMENTATION

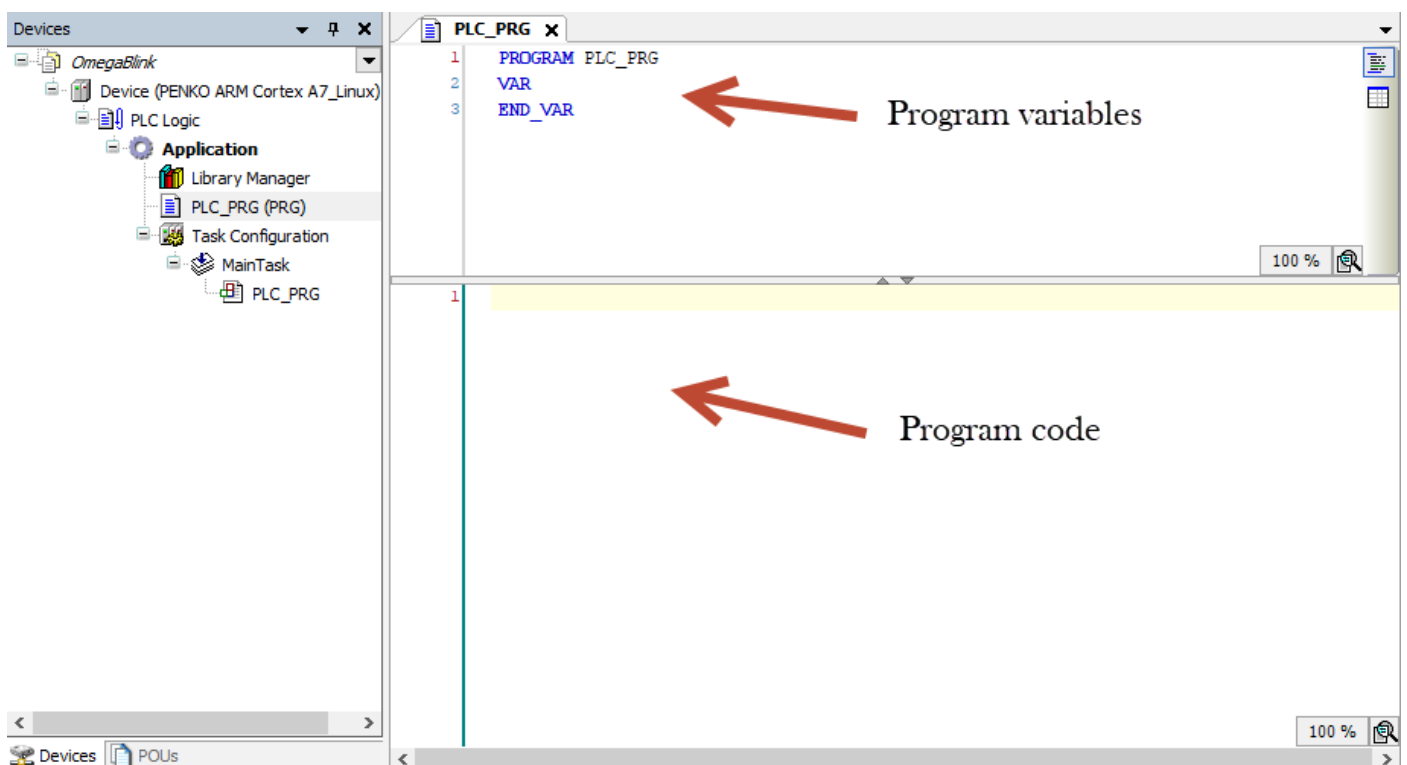
Now the Penko BSP Omega library has been added to the project (1).

To view all the library functions, see the included documentation (2).



D) SELECT PLC_PRG (PRG)

PLC_PRG is the default name for new programs and selected under the Main Task to be executed. To develop your application, there are two important areas. The bottom part where the program code is constructed. The top part is used to set the program name and to define the program's variables between VAR and END_VAR.



Getting started with CodeSys

E) WRITE A BLINK PROGRAM: EXAMPLE 1

All the code below can be found as text in Appendix I: Blink program example 1. The available LED control functions are described in the included library documentation as mentioned before in the library manager.

First, we start by defining some variables in the top area. These variables are used to hold the state of the LED's and by changing the value we can change the LED color, turn a LED on or off and control each available LED.

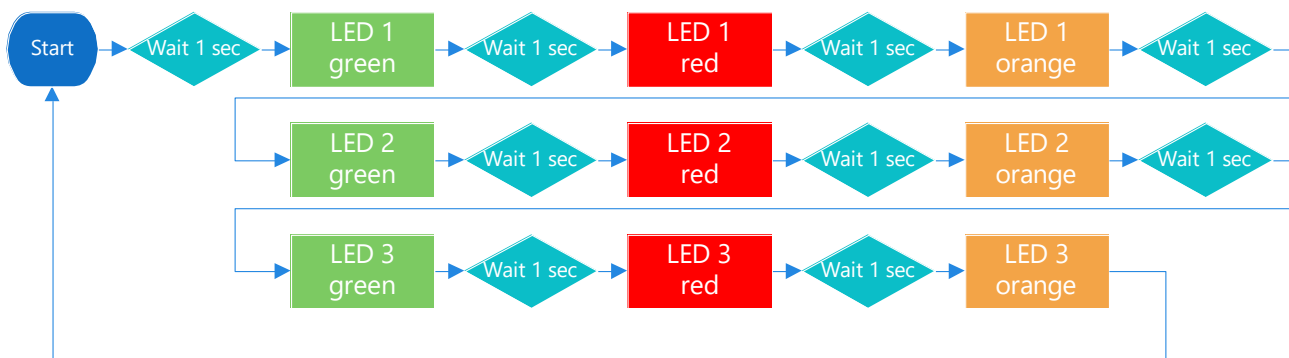
All text on a line after '//' is a comment. It is not part of the code but used to document the program.

```
Library Manager  PLC_PRG x
1  PROGRAM PLC_PRG
2  VAR
3      leds: PENKO_BSP.Leds;    // Get the LED's from the library
4      Delay: TON;              // Time to determine blink rate
5      color_ctr: INT := PENKO_BSP.LED_COLOR.GREEN; // Start at color green
6      led_ctr: INT;            // For selecting the LED (LED1, LED2, LED3, LED4)
7  END_VAR
```

After defining the variables, we start writing some program code in the lower area to control the LED's. In the code below LED4 is always turned on in the color red. The LED's, 1, 2 and 3 change color one after the other in a fixed sequence from green to red to orange. To turn off the previous LED's the led.setToBits(0) is called.

```
1  Delay(IN:=TRUE, PT:=T#1S); // Turn on the timer and set it to one second
2  IF NOT(Delay.Q) THEN      // Wait till the timer has reached one second
3      RETURN;               // Timer not at one sec? Don't execute the rest of the code below
4  END_IF
5  Delay(IN:=FALSE);         // Turn off the timer
6  leds.setToBits(0);        // Turn off all LED's
7
8  leds.setColor(PENKO_BSP.LED.LED4, PENKO_BSP.LED_COLOR.RED); // Always turn on LED4 in red
9
10 leds.setColor(led_ctr, color_ctr); // Set the LED number and color
11 color_ctr := color_ctr + 1;        // Change color
12
13 IF color_ctr = PENKO_BSP.LED_COLOR.NUM THEN
14     color_ctr := PENKO_BSP.LED_COLOR.GREEN; // Go back to green
15     led_ctr := led_ctr + 1;                // go to the next LED
16     IF led_ctr = PENKO_BSP.LED.LED4 THEN
17         led_ctr := PENKO_BSP.LED.LED1;    // Go back to LED1
18     END_IF
19 END_IF
```

Below the flowchart of the above program.

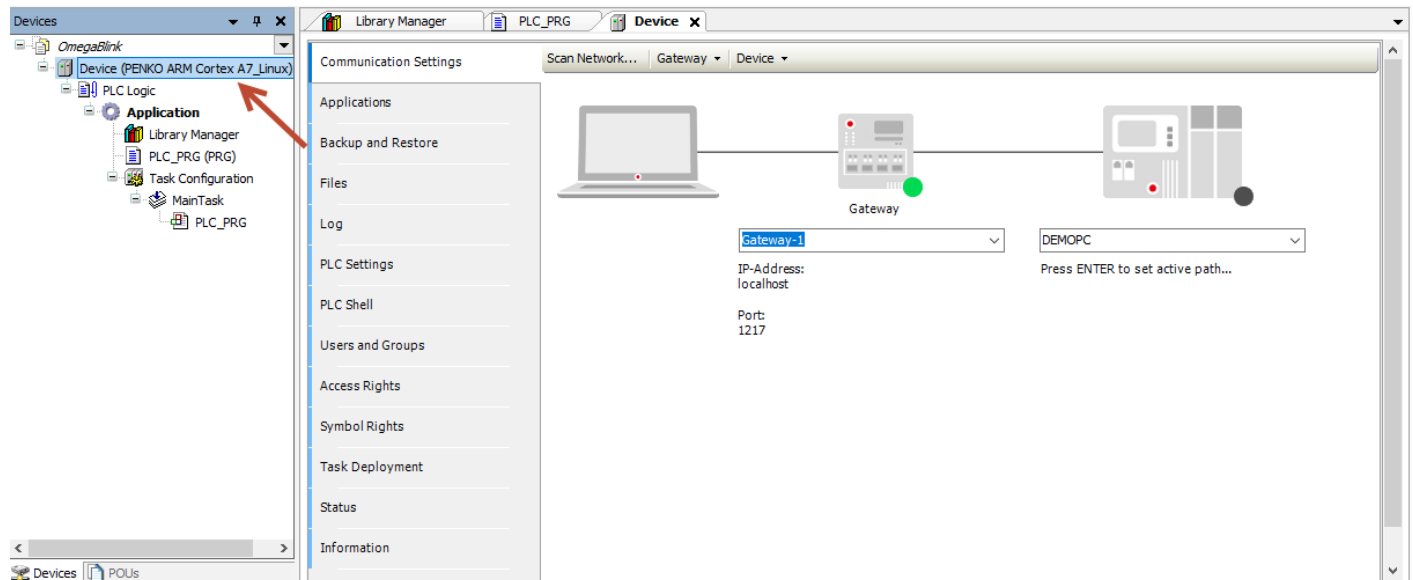


Getting started with CodeSys

STEP 4: RUN YOUR PROGRAM

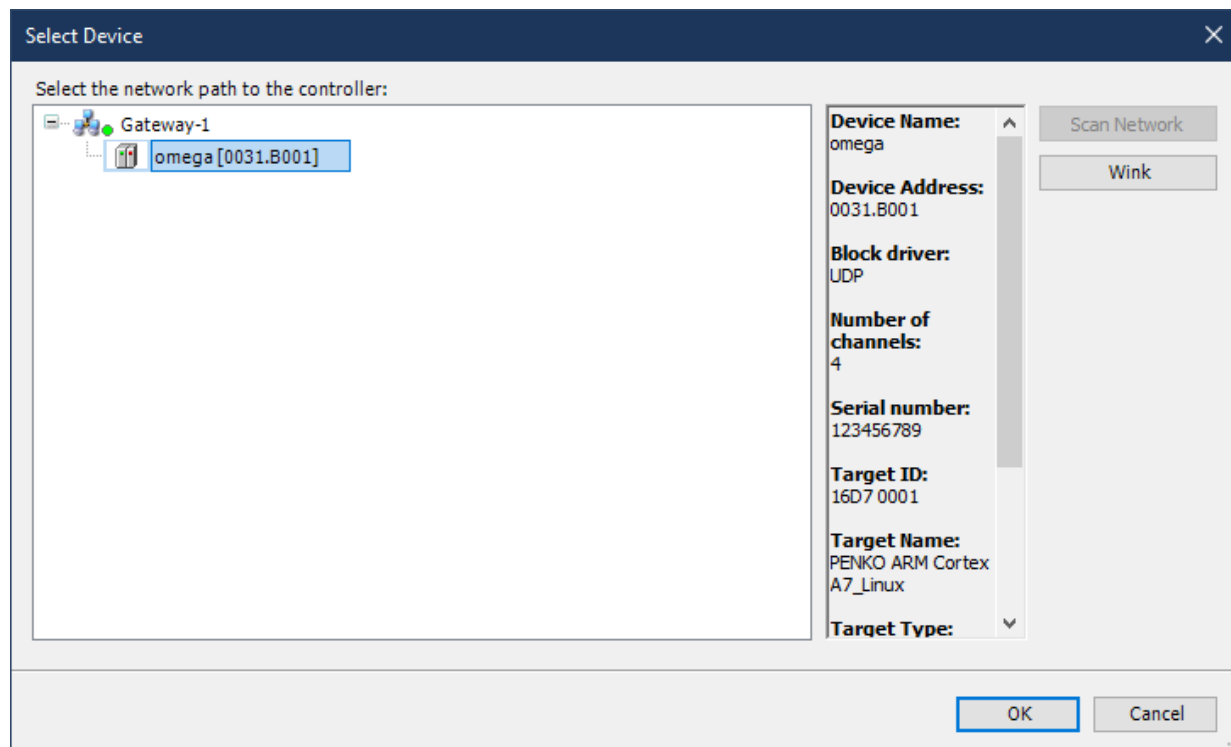
After writing the program code, go to the device item. In the next steps, the code from example 1 is used.

A) SELECT THE OMEGA DEVICE



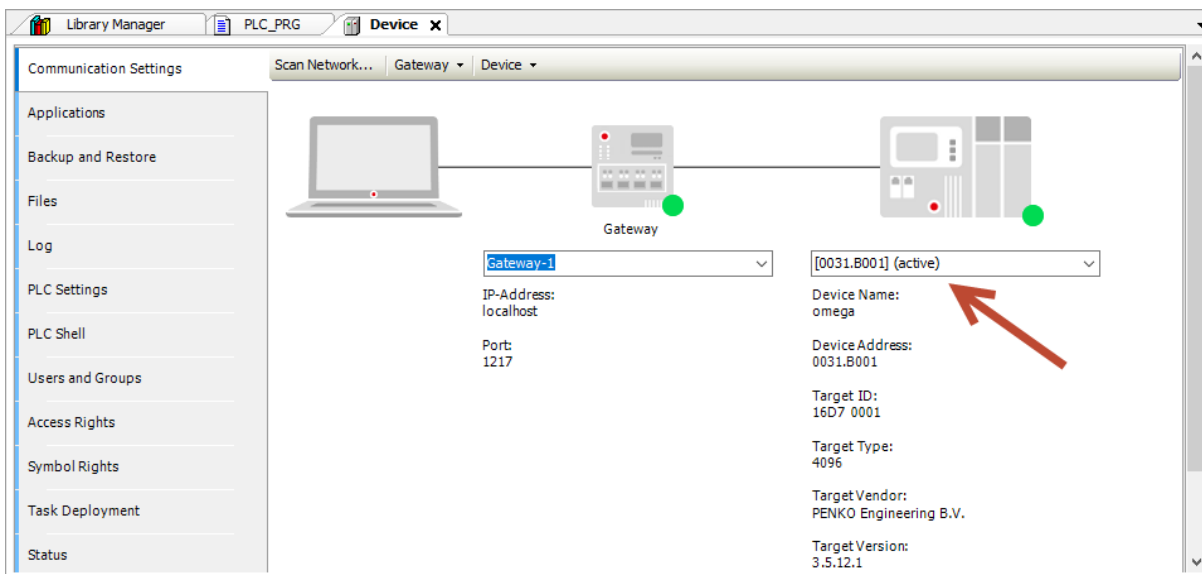
The gateway status circle is already green.

Now hit the “Scan network” button to find the Omega device in your network.



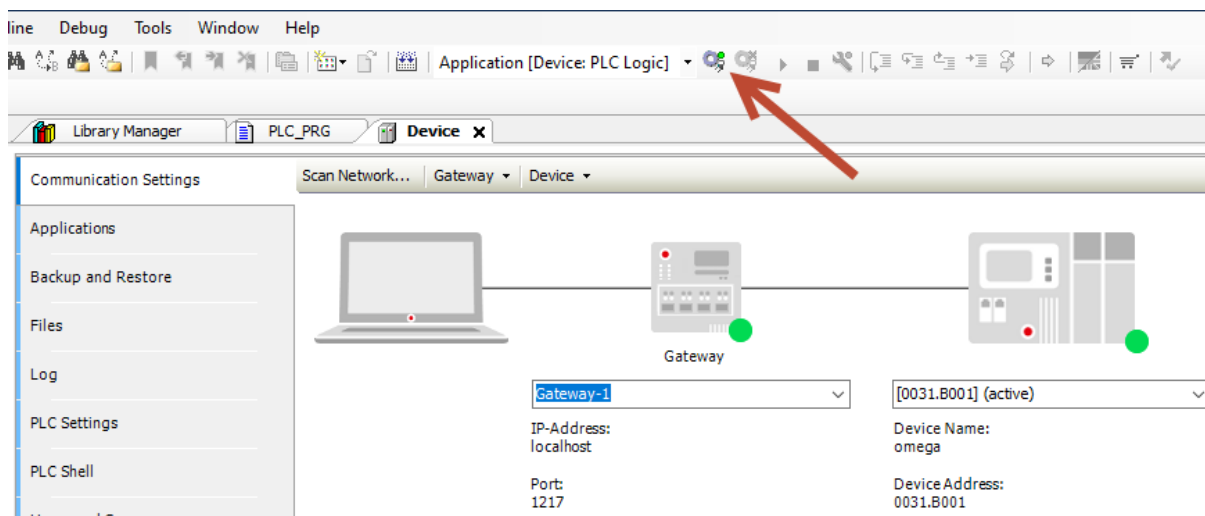
Getting started with CodeSys

Now the omega device is selected.

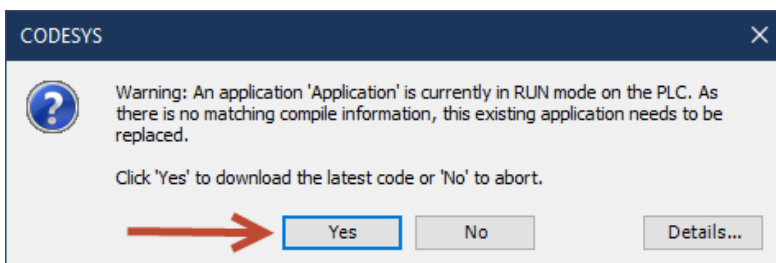


B) DOWNLOAD THE PROGRAM TO THE DEVICE

Hit the "login" button to download the program to the device.

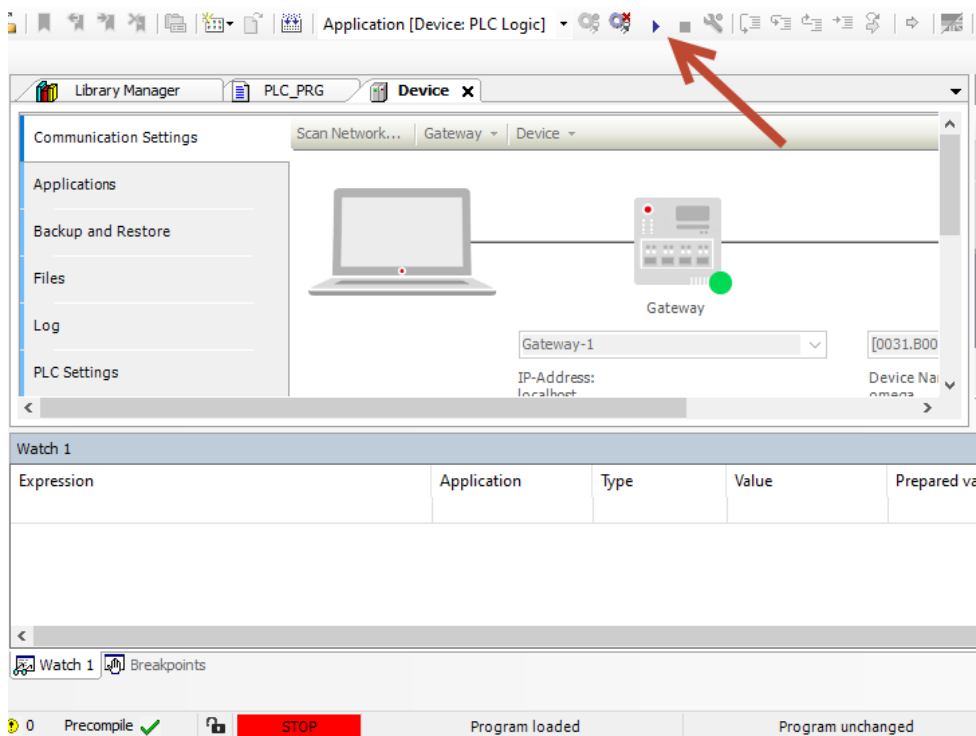


A pop-up appears, hit "Yes".



Getting started with CodeSys

After the program has been downloaded, start the program by clicking on the start symbol.



C) PROGRAM IN RUNNING MODE

Go back to the PLC_PRG to see if the program is running. The values behind the color_ctr and led_ctr are changing and the LED's on the Omega CPU card are changing color every second.

Expression	Type	Value	Prepared value	Address	Comment
leds	PENKO_BSP.Leds				Get the LED's from the library
Delay	TON				Time to determine blink rate
color_ctr	INT	4			Start at color green
led_ctr	INT	1			For selecting the LED ...1, LED2, LED3, LED4)

```
1 Delay(IN:=TRUE, PT:=T#1s, S_ODT:=T#1s); // Turn on the timer and set it to one second
2 IF NOT(Delay.QFALSE) THEN // Wait till the timer has reached one second
3   RETURN; // Timer not at one sec? Don't execute the rest of the code below
4 END_IF
5 Delay(IN:=FALSE); // Turn off the timer
6 leds.setToBits(0); // Turn off all LED's
7
8 leds.setColor(PENKO_BSP.LED.LED4, PENKO_BSP.LED_COLOR.RED); // Always turn on LED4 in red
9
10 leds.setColor(led_ctr, color_ctr); // Set the LED number and color
11 color_ctr := color_ctr + 1; // Change color
12
13 IF color_ctr = PENKO_BSP.LED_COLOR.NUM_COLORS THEN
14   color_ctr := PENKO_BSP.LED_COLOR.GREEN; // Go back to green
15   led_ctr := led_ctr + 1; // go to the next LED
16 IF led_ctr = PENKO_BSP.LED.LED4 THEN
17   led_ctr := PENKO_BSP.LED.LED1; // Go back to LED1
18 END_IF
19 END_IF RETURN
```

Getting started with CodeSys

WRITE A BLINK PROGRAM: EXAMPLE 2

All the code below can be found as text in Appendix II: Blink program example 2. The available LED control functions are described in the included library documentation as mentioned before in the library manager.

Repeat Step 2 to create a new project.

At step 3 at chapter D write the following program:

Again, we start by defining some variables for controlling the LED's. These variables are used to change the LED color, turn a LED on or off and control each available LED.

```
Library Manager  PLC_PRG x  Device
1  PROGRAM PLC_PRG
2  VAR
3      leds: PENKO_BSP.Leds;  // Get the LED's from the library
4      Delay: TON;             // Time to determine blink rate
5      state: UINT := 0;       // Start at 0 (all LED's off)
6  END_VAR
```

In the code below, multiple LED's are turned on by using the setOnMask() function. The similar function setMaskOff can be used to turn off multiple LED's. The code loops throw all the 16 different states (0 to 15) to show the output for each state. The output is also described in the table below the code.

```
1  Delay(IN:=TRUE, PT:=T#500MS); // Turn on the timer and set it to one second
2  IF NOT(Delay.Q) THEN           // Wait till the timer has reached one second
3      RETURN;                    // Timer not at one sec? Don't execute the rest of the code below
4  END_IF
5  Delay(IN:=FALSE);              // Turn off the timer
6  leds.setToBits(0);            // Turn off all LED's
7
8  leds.setOnMasked(state);       // Turn one or more LED on at once
9  state := state + 1;            // Go to the next state
10
11 IF state > 15 THEN              // If all 16 states reached
12     state := 1;                // Go back to only the first LED on
13 END_IF
```

State	LED1	LED2	LED3	LED4
0	OFF	OFF	OFF	OFF
1	ON	OFF	OFF	OFF
2	OFF	ON	OFF	OFF
3	ON	ON	OFF	OFF
4	OFF	OFF	ON	OFF
5	ON	OFF	ON	OFF
6	OFF	ON	ON	OFF
7	ON	ON	ON	OFF
8	OFF	OFF	OFF	ON
9	ON	OFF	OFF	ON
10	OFF	ON	OFF	ON
11	ON	ON	OFF	ON
12	OFF	OFF	ON	ON
13	ON	OFF	ON	ON
14	OFF	ON	ON	ON
15	ON	ON	ON	ON

Finally, repeat step 4 to run the program.

Getting started with CodeSys

APPENDIX I: BLINK PROGRAM EXAMPLE 1

Program variables:

```
PROGRAM PLC_PRG
VAR
    leds: PENKO_BSP.Leds;    // Get the LED's from the library
    Delay: TON;              // Time to determine blink rate
    color_ctr: INT := PENKO_BSP.LED_COLOR.GREEN; // Start at color green
    led_ctr: INT;            // For selecting the LED (LED1, LED2, LED3, LED4)
END_VAR
```

Program code:

```
Delay(IN:=TRUE, PT:=T#1S);    // Turn on the timer and set it to one second
IF NOT(Delay.Q) THEN          // Wait till the timer has reached one second
    RETURN;                   // Timer not at one sec? Don't execute the rest of the code below
END_IF
Delay(IN:=FALSE);             // Turn off the timer
leds.setToBits(0);            // Turn off all LED's

leds.setColor(PENKO_BSP.LED.LED4, PENKO_BSP.LED_COLOR.RED); // Always turn on LED4 in red

leds.setColor(led_ctr, color_ctr); // Set the LED number and color
color_ctr := color_ctr + 1;      // Change color

IF color_ctr = PENKO_BSP.LED_COLOR.NUM THEN
    color_ctr := PENKO_BSP.LED_COLOR.GREEN;    // Go back to green
    led_ctr := led_ctr + 1;                     // go to the next LED
    IF led_ctr = PENKO_BSP.LED.LED4 THEN
        led_ctr := PENKO_BSP.LED.LED1;        // Go back to LED1
    END_IF
END_IF
```

Getting started with CodeSys

APPENDIX II: BLINK PROGRAM EXAMPLE 2

Program variables:

```
PROGRAM PLC_PRG
VAR
    leds: PENKO_BSP.Leds; // Get the LED's from the library
    Delay: TON;           // Time to determine blink rate
    state: UINT := 0;     // Start at 0 (all LED's off)
END_VAR
```

Program code:

```
Delay(IN:=TRUE, PT:=T#500MS); // Turn on the timer and set it to one second
IF NOT(Delay.Q) THEN          // Wait till the timer has reached one second
    RETURN;                   // Timer not at one sec? Don't execute the rest of the code below
END_IF
Delay(IN:=FALSE);             // Turn off the timer
leds.setToBits(0);            // Turn off all LED's

leds.setOnMasked(state);      // Turn one or more LED on at once
state := state + 1;           // Go to the next state

IF state > 15 THEN             // If all 15 states reached
    state := 1;               // Go back to only the first LED on
END_IF
```


Getting started with CodeSys



About PENKO

At PENKO Engineering we specialize in weighing. Weighing is inherently chemically correct, independent of consistency, type or temperature of the raw material. This means that weighing any kind of material guarantees consistency and thus, it is essential to sustainable revenue generation in any industry. As a well-established and proven solution provider, we strive for the ultimate satisfaction of custom design and/or standard applications, increasing your efficiencies and saving you time, saving you money.

Whether we are weighing raw materials, components in batching, ingredients for mixing or dosing processes, - or weighing of static containers and silos, or - in-motion weighing of railway wagons or trucks, by whatever means required during a process, we are essentially forming vital linkages between processes and businesses, anywhere at any time. We design, develop and manufacture state of the art technologically advanced systems in accordance with your strategy and vision. From the initial design brief, we take a fresh approach and a holistic view of every project, managing, supporting and/or implementing your system every step of the way. Curious to know how we do it? www.penko.com

Certifications

PENKO sets high standards for its products and product performance which are tested, certified and approved by independent expert and government organizations to ensure they meet – and even – exceed metrology industry guidelines. A library of testing certificates is available for reference on:

www.penko.com/nl/publications_certificates.html

PENKO Professional Services

PENKO is committed to ensuring every system is installed, tested, programmed, commissioned and operational to client specifications. Our engineers, at our weighing center in Ede, Netherlands, as well as our distributors around the world, strive to solve most weighing-system issues within the same day. On a monthly basis PENKO offers free training classes to anyone interested in exploring modern, high-speed weighing instruments and solutions. Training sessions on request:

www.penko.com/training



PENKO Distributor

A complete overview you will find on: www.penko.com/Find-A-Dealer



PENKO Engineering B.V. • Schutterweg 35, NL 6718C Ede • Tel +31 (0) 318525630 • info@penko.com

Web • www.penko.com • Copyright © 2014 ETC All rights reserved. 7600M1083-EN-R1 CODESYS GETTING STARTED.DOCX