CONTROL YOUR LIVING SPACE

Application manual

CODESYS PLC module with KNX interface EK-IA1-TP

Contents

1 Scope of the document

This application manual describes application details for the A1.0 release of the ekinex® CODESYS PLC module with KNX interface EK-IA1-TP.

The document is aimed at the system configurator as a description and reference of device features and application programming. For further installation, mechanical and electrical details of the device please refer to the technical description datasheet.

Application manual and software tools are available for download either at www.ekinex.com or from other sites as described later in the manual.

Item	File name (## = release)	Version	Device rel.	Update
Technical datasheet	STEKIA1TP_EN.pdf	-	A1.0	03 / 2014
Application manual	MAEKIA1TP_EN.pdf	-		
Interface tool installer	ETS2PLC.msi	1.0		

You can access the most up-to-date version of the full documentation for the device using following QR code:

2 Product description

2.1 Generic product features

The ekinex® CODESYS PLC module with KNX interface EK-IA1-TP is a modular device for rail mounting that allows to access data (communication objects) on the KNX bus and operate on them for logic processing.

The device is equipped with an integrated bus communication module and is designed for rail mounting in distribution boards.

The logic processing core is a PLC with a structure and a set of languages that comply to the international IEC 61131-3 standard.

The PLC in its basic version is not meant to have onboard inputs or outputs (as with common industrial automation PLCs): it is specifically targeted to operate on the communication objects that are defined and used in a KNX installation.

The PLC is capable of addressing these objects in a mostly transparent way by seeing them as own internal variables. The binding of these variables to the appropriate communication objects is managed by a firmware library; the definition of these bindings is made through a helper software tool that extracts the references to communication objects from an existing ETS project and compiles the corresponding definitions to be copied into the PLC source code.

Since the access to KNX objects is completely defined on the PLC side, and also because of the extreme flexibility and configurability if its tasks, the device needs not – and indeed does not – have an ETS application program. For this reason, it does not appear in the ETS project as a component.

The programmer and the configurator, however, shall of course have a good awareness of how both sides of the system operate on the common object base, in order to achieve the desired operation and avoid unwanted interference.

The device is powered by the KNX bus line with a 30 VDC SELV voltage and does not require auxiliary power.

2.2 Technical features

2.2.1 Hardware features

CPU	ARM CORTEX M3 110MHz
Memory on board	1 MB RAM 1 MB Flash 32 kB FRAM
USB ports	1 - reserved for programming and debug
Ethernet ports	none (future expansion)
Serial RS485 ports	none (future expansion)
Real-time clock	Yes
Power supply	through the KNX bus; no auxiliary supply required.

2.2.2 Software features

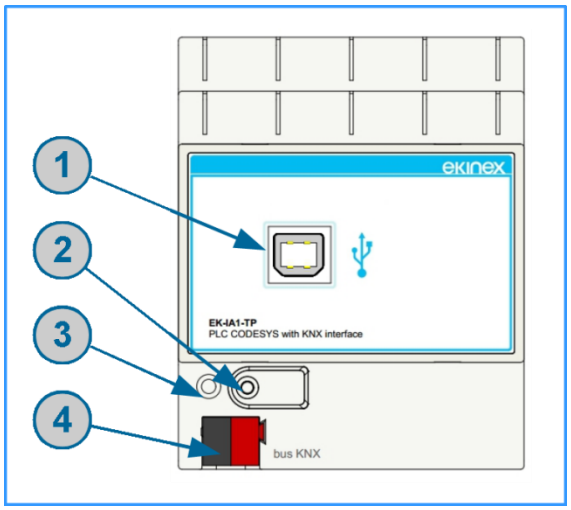
PLC Runtime	Single task CODESYS V3
PLC programming language(s)	All CODESYS V3 / IEC 61131-3: • IL (Instruction List) - Textual • ST (Structured Text) - Textual • FBD (Function Block Diagram) - Graphical • LD (Ladder Diagram) - Graphical • SFC (Sequential Function Chart) - Graphical
Number of KNX communication objects that can be managed	442
Size of PLC memory*	Program..... 128 kB Variables..... 64 kB Retentive area .....256 Bytes
minimum cycle time	2 ms

* The actual size of PLC memory available to the user might be slightly less depending on usage by the CODESYS runtime.

For further technical information, please also refer to the product datasheet STEKIA1TP_EN.pdf available on the ekinex website www.ekinex.com.

3 Switching, display and connection elements

The device connections are shown below:



1) USB connector for the connection to a PC
2) KNX programming pushbutton
3) KNX programming LED
4) Terminal block for KNX bus line

Figure 1: Switching, display and connection elements

The device should be connected to a PC through a common USB cable (Type A to Type B plugs) of good quality; the connection is only needed during the programming phase (and possibly during operation monitoring). The device is powered through the KNX bus; the KNX plug must therefore be connected to an active KNX line during both programming and, obviously, operation.

Since the device has no ETS application program, and therefore there is no need for programming through ETS, the KNX programming pushbutton and LED have currently no purpose; they are provided for future use. However, since the LED is switched on and off by activating the pushbutton, they can be used as a raw check that the internal processor of the device is actually running.

4 Configuration

This section contains the information required in order to correctly:

- verify that the PC to be used for development and programming complies to the required specifications;
- find and download all necessary software tools for development;
- setup a PC with the CODESYS development tools and environment;
- install the specific ekinex PLC libraries in the CODESYS environment;
- setup the PC for the connection to the PLC for programming and online monitoring;
- install and use the ETS2PLC ekinex® software tool to import the KNX communication object definition into the PLC source code.

The development of the user application can, and usually will, be performed mostly offline; the device programming takes place in the commissioning phase.

Warning: please do not connect the device with the USB cable until instructed to do so in the following procedures. Doing otherwise will not harm either the device or the software installation, but may result in different operations to be performed other than those described.

4.1 PC hardware and software requirements

The hardware and software requirements for the development PC, for each tool respectively, are as follows.

- **CODESYS development environment**: Windows XP/7/8 (32/64 Bit); suitable PC hardware for the used Windows platform.

  Further requirements may arise during installation (e.g. Microsoft .NET framework, VisualC++ redistributable packages etc.)

- **ekinex ETS2PLC tool**: Windows XP/7/8 (32/64 Bit); Microsoft .NET framework 2.0 or newer.

4.2 Software procurement

The required software can be downloaded free of charge from following sources:

- **CODESYS development environment (IDE)**: from the 3S/CODESYS website www.codesys.com.
  Please follow the instructions on the website to find and download the most recent version available of the development environment, together with available documentation, tutorials and everything else is required for development.
  Registration is required to access the download: an account can be created free of charge.
  The installation file should have a name like "`CODESYS V3.x SPx Patch x`" (according to latest release numbers) and has a size roughly around 480MB (as of version `CODESYS V3.5 SP4 Patch 1`).
  Please be aware that the CODESYS development software is made available by 3S – Smart Software Solutions, under its own terms and conditions. Ekinex is not liable for the software in subject nor it is able to offer specific support regarding its installation or usage, although the ekinex technical service will gladly try to answer customer's enquiries in this matter to the best of its abilities.

- **ekinex ETS2PLC tool, ekinex PLC libraries for CODESYS, sample projects**: from the ekinex website www.ekinex.com.
  Please refer to the website page dedicated to the product to find and download the most recent versions available.
  All required files, which will be referenced in following sections, are grouped together in a single archive file; please extract these files in a convenient location on your PC where you will be able to easily retrieve them during the installation procedure.

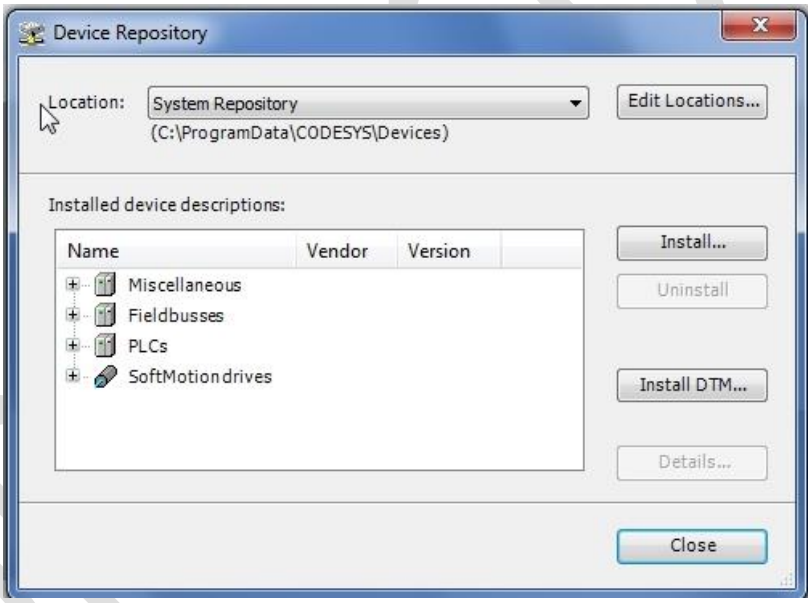
4.3 Setup of the CODESYS IDE

The CODESYS IDE (Integrated Development Environment) can be installed from the file downloaded from the 3S website; after installation is finished, a few configuration operations are required in order to be able to address the ekinex KNX PLC.

Please notice: in following section, all references will be made to version `CODESYS V3.5 SP4 Patch 1`*. Other releases might have slight differences such as different menu names, different options etc.*

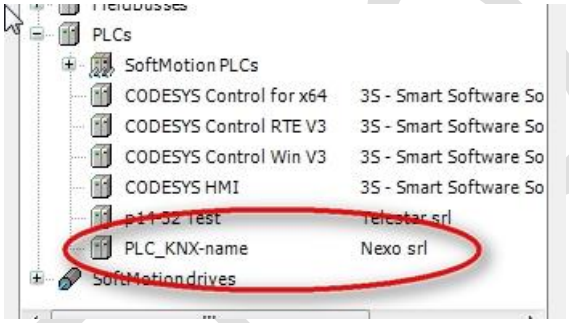
4.3.1 Installation of the ekinex KNX PLC device

Open the IDE and choose menu option *Tools / Device repository...*



Click on the "*Install...*" button; make sure that the file filter option is set to "*Device description files (.devdesc.xml)*", which is not the default, and select file `PLC_knx-flash.devdesc.xml` contained in the ekinex archive.

After the import, the device will appear in the list of installed items (under the "*PLCs*" category):



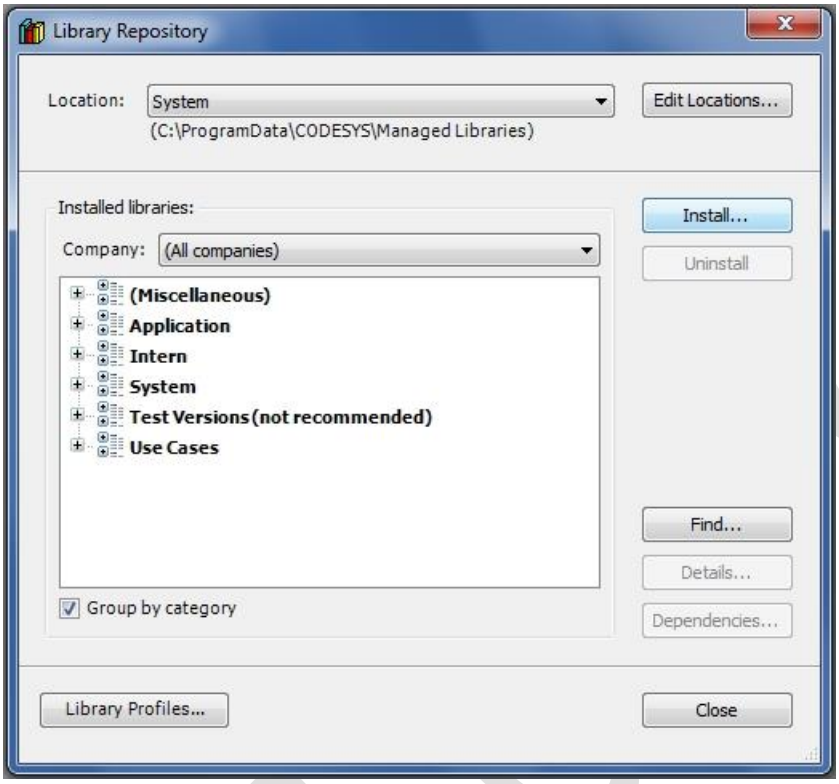
Warning: in other releases of the environment, the Device Repository feature may not be available: the relevant menu item is named "Tools / Install Device..." and directly opens the device file selection dialog.

This procedure needs just be followed once as the environment is first installed; the device will then be available for all subsequent projects.

4.3.2 Installation of ekinex KNX PLC libraries

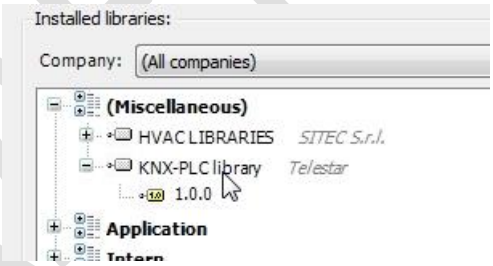
The procedure is very similar to the one described for the device installation.

In the CODESYS IDE, choose menu option *Tools / Library repository...*



Click on the "*Install..."* button; make sure that the file filter option is set to "*Library files (.library)*", which is not the default, and select file `plc_knx_lib.library` contained in the ekinex archive.

After import, the library will appear in the list of installed items (under the "*Miscellaneous*" category):



Warning: in other releases of the environment, the Library Repository feature may not be available: the relevant menu item is named "Tools / Install Library..." and directly opens the library file selection dialog.

This procedure needs just be followed once as the environment is first installed; the library will then be available for all subsequent projects.

Once the library is succesfully installed, it can be added to an existing project by double-clicking on the *Library Manager* item on the project tree in the left-hand pane, then "*Add library"* (see also the sample project creation below).

4.3.3 Setup and connection to the ekinex PLC

In order to allow the connection to the physical device, the Gateway component of the CODESYS system must be configured properly. The Gateway acts as a proxy between the IDE and a PLC device, either physical or virtual, and to this purpose has an interface on both sides.

The interface between the Gateway and the IDE is a TCP/IP channel, which is correctly configured by default; on the device side, the default configuration is initially also TCP/IP. This is appropriate for the connection to the "soft-PLC" / PLC simulator, which comes preinstalled, made available by CODESYS for development; the interface channel to the ekinex KNX PLC, though, is through a USB interface which uses a virtual COM port, which needs to be configured manually.

Apply power to the PLC by connecting it to a KNX bus, then connect it to your PC with the USB cable[1].

The operating system should detect the new device and prompt you for the correct driver: choose to specify where to find the driver files on your PC, and select the folder containing the `telestar.inf` and `telestar-win7.inf` files from the archive downloaded from the ekinex website.

After the installation completes, a message should report that a vitrual COM port has been created:

If the message does not appear, open the right-click menu of the "*My Computer*" icon on the desktop, item "*Manage*", then "*Device manager*" and look into group "*Ports (COM & LPT)*". An entry "*Telestar USB VirtualCom port (COMx)*" shoud be there:

In either case, look for the "(COM*x*)" part, where the "*x*" is the number assigned by the system to the PLC port (and should remain unchanged even if the PLC is disconnected and later reconnected).

Close the CODESYS IDE (if open).

Locate the "`C:\%programfiles(x86)%\3S CODESYS\GatewayPLC\Gateway.cfg`" file[2] and open it with a text editor. Locate the lines in the table below and modify / add them as shown, *inserting the appropriate COM port number obtained from the step above* (in the example COM 5 is used):

Original	*Modified*

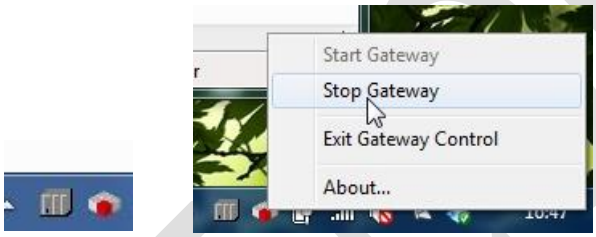
`...`
`[CmpRouter]`
`...`
`0.MainNet=ether x`
	`...`
	`[CmpRouter]`
	`...`
	`0.MainNet=ether x`

[1] Please use a good quality cable, possibly with a limited length (< 2m) in order to prevent problems in device recognition or communication.

[2] The unit letter C: may be different according to your installation.

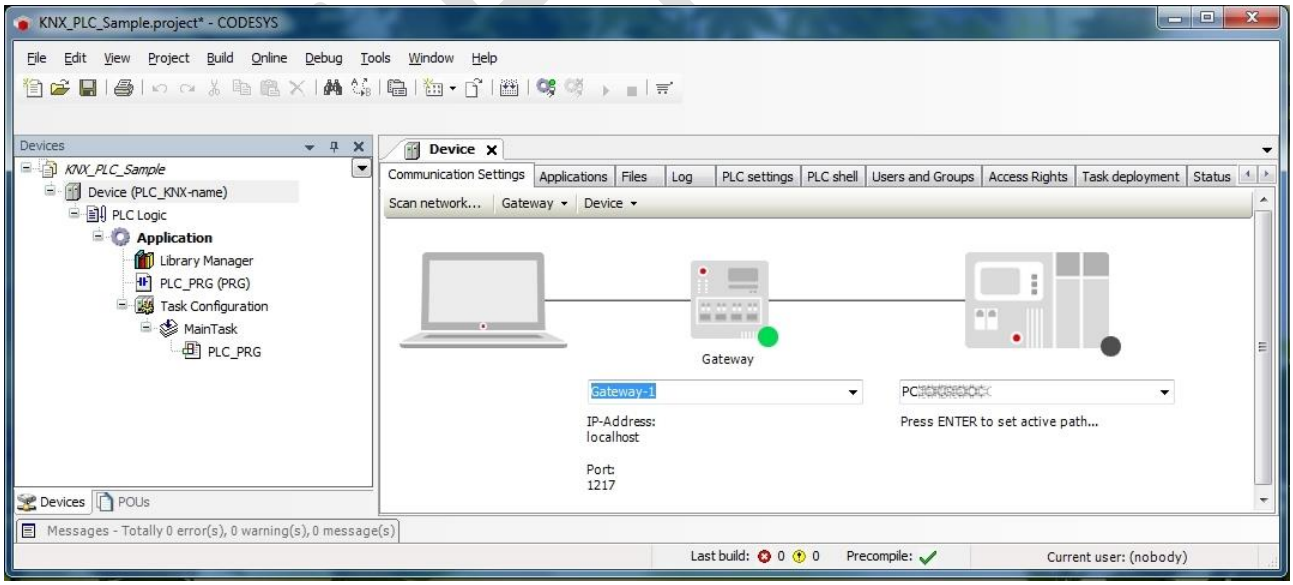
0.NumSubNets=1 0.SubNet.0.Interface=BlkDrvShm 1.MainNet=BlkDrvTcp ... [CmpBlkDrvCom] ... ;Com.0.Port=1 ;Com.0.Name=MyCom ;Com.0.Baudrate=115200 ;Com.0.EnableAutoAddressing=1 ...	0.NumSubNets=1 0.SubNet.0.Interface=BlkDrvShm 1.MainNet=BlkDrvTcp **2.MainNet=BlkDrvCom** ... [CmpBlkDrvCom] ... **Com.0.Port=5** **Com.0.Name=RSRCom** **Com.0.Baudrate=115200** **Com.0.EnableAutoAddressing=1** ...

Once the corrected configuration file is saved, stop and restart the Gateway service through the icon in the system traybar:

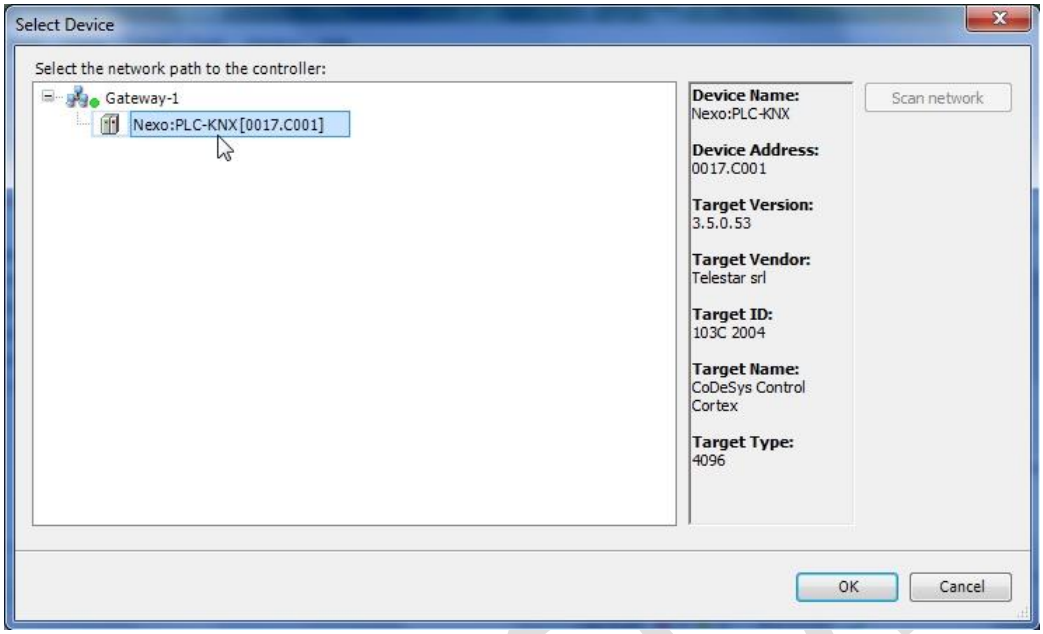


Now start the CODESYS IDE and open a project (such as the basic one just described) configured to use the KNX PLC.

With a double-click on the "*Device*" icon (the root of the tree in the left-hand pane), the gateway is displayed on the right side.



Click on the "*Scan network...*" button and a dialog box opens:



The "PLC-KNX" item should appear; if it does not, once the scan is finished try and launch it again with the button on the right. If the scan still fails, check following points:

- the device is still recognized correctly by the system, i.e. it is displayed (in the correct section and without a yellow question mark sign) in the system's Device Manager; if the device does not appear at all, check if it's is still connected and powered;
- that the steps described above for gateway configuration have all been correctly performed, with particular focus on the gateway restart operation.

If the device correctly appears on the list, select it and click "*OK*" to make it the active device; the right pane should appear as in the picture below.

Now the PLC can be accessed by selecting the menu command "*Online / Login*".

5 Commissioning

After the device has been programmed according to user requirements, if programming has been made in a separate KNX installation, the device can be connected the final KNX network; no further operation is necessary.

6 Function description

The functionality of the device depends entirely on the application program written by the user.

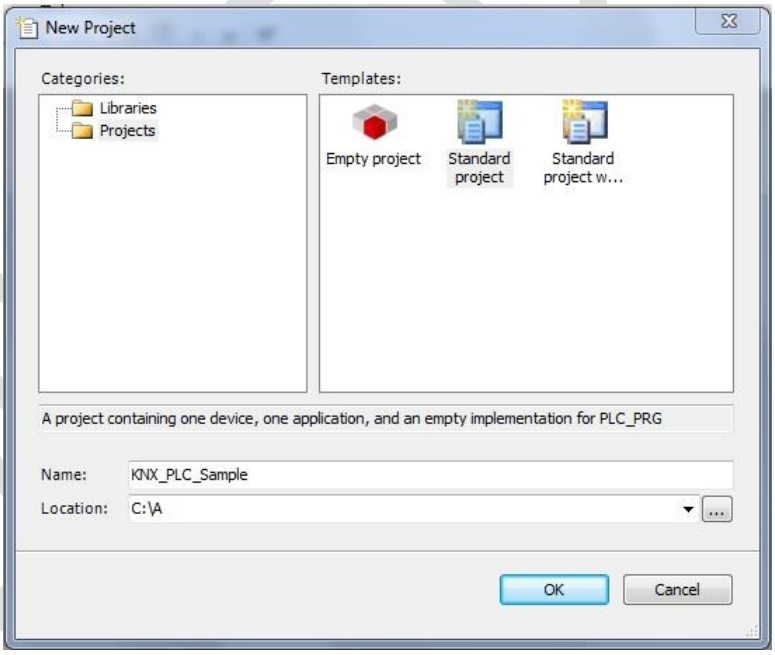
This section describes the details required for following operations:

- set up a new basic project using the CODESYS IDE;
- export the group address list from an existing KNX ETS project;
- use the provided library functions to handle the specifics of KNX communication object interface.

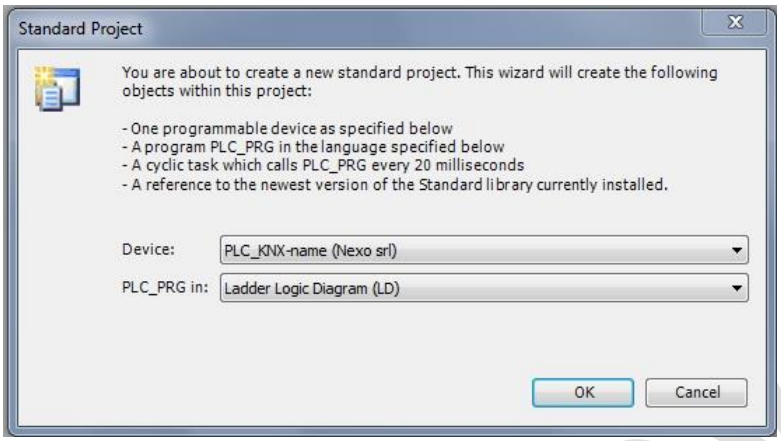
6.1 Creation of a sample CODESYS project

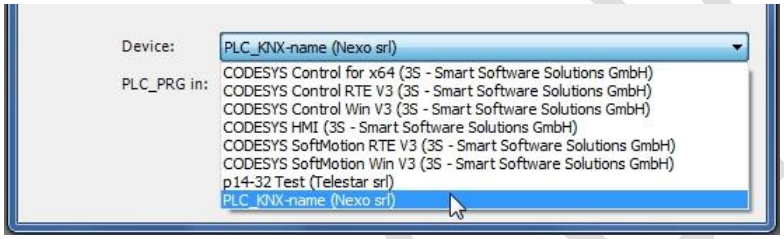
Though the subject is not specific to this device, it will be briefly shown how to create from scratch a basic project with a KNX PLC.

Start by selecting "*File / New project...*", assign a name and then select "*Standard project*":

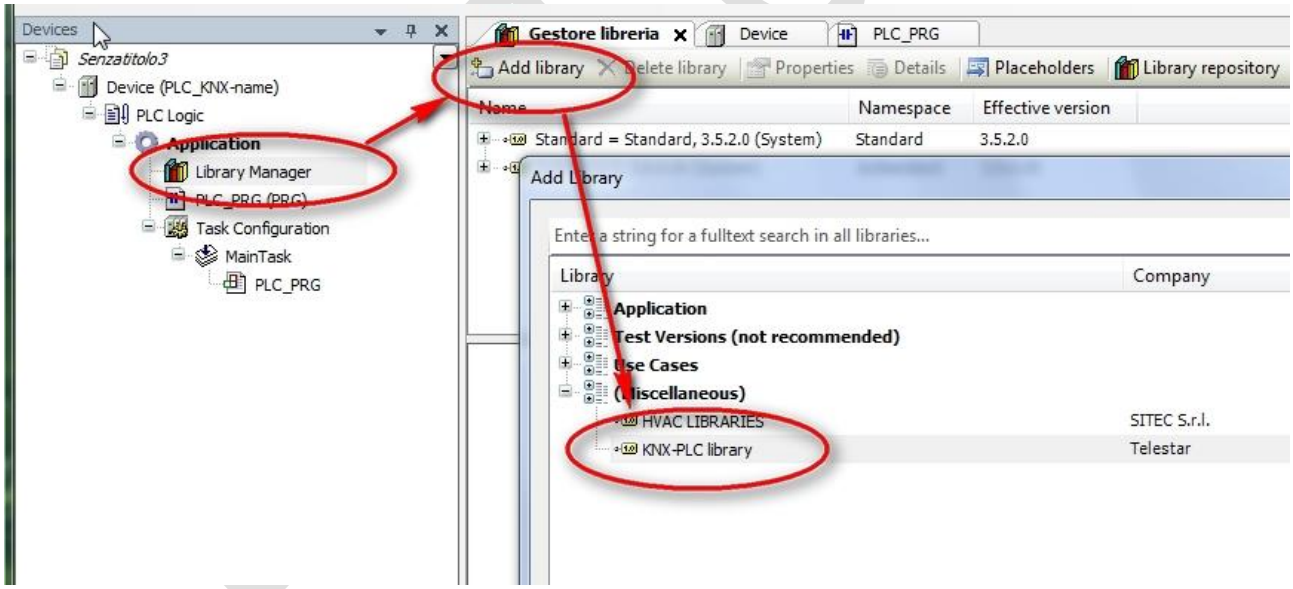


In the following dialog, make sure to select the correct device type and choose the language you prefer:

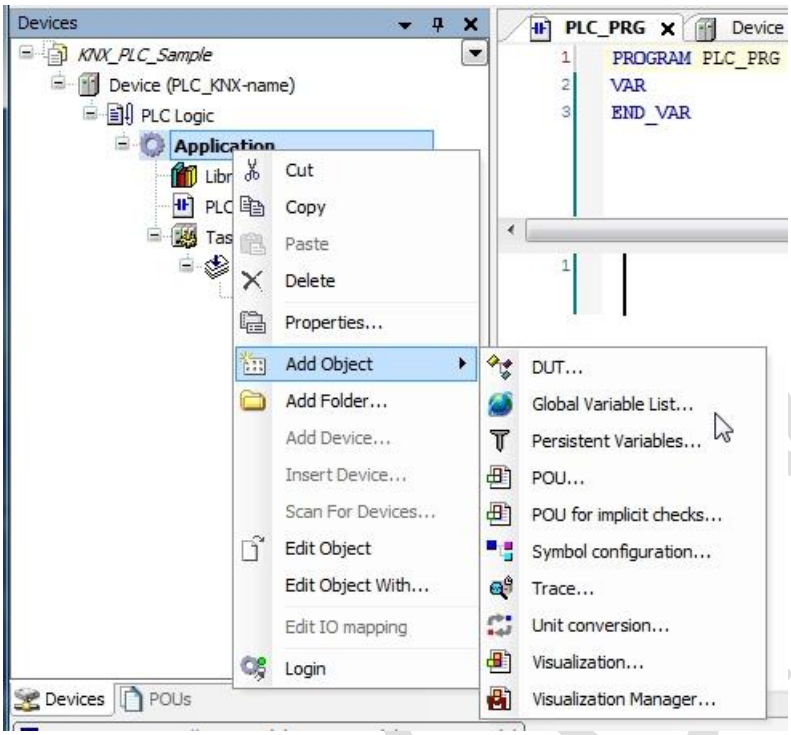




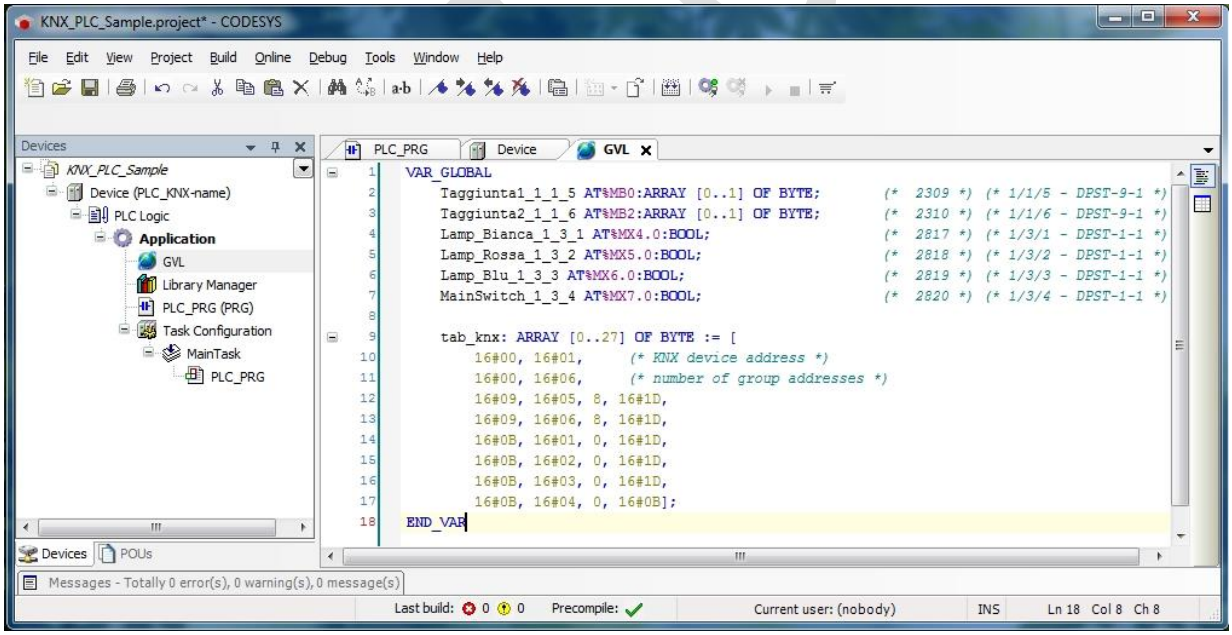
Once the project is opened, add the KNX PLC library by double-clicking on the *Library Manager* item on the project tree in the left-hand pane, then "*Add library*":



By right-clicking on the "*Application*" tree item, select "*Add Object / Global variable list*":



Keep the default name (“GVL”) or pick your own if you prefer; this unit is where the source code for the KNX interface variables will be inserted later (more details in the following sections).

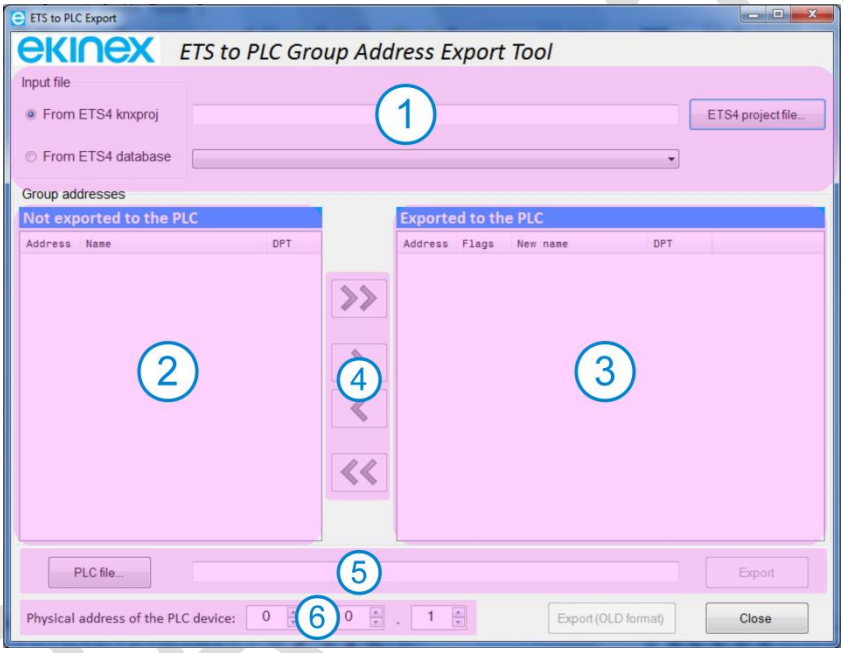


6.2 Use of the ekinex ETS2PLC tool

The ekinex ETS2PLC tool is a program that allows to extract all group address references from an ETS4 project (either in form of a project file or a database); the output is written to a text file that can be directly imported into a CODESYS program.

ETS2PLC can be launched from the icon that has been put on the desktop during installation (unless the user chose otherwise):

The program window is made by following sections:



1. ETS project selection 2. List of ETS group addresses that will not appear in the output file 3. List of ETS group addresses that will appear in the output file	4. Buttons to select where the group address items will be listed 5. Output file selection button and name 6. Group address to be assigned to the PLC

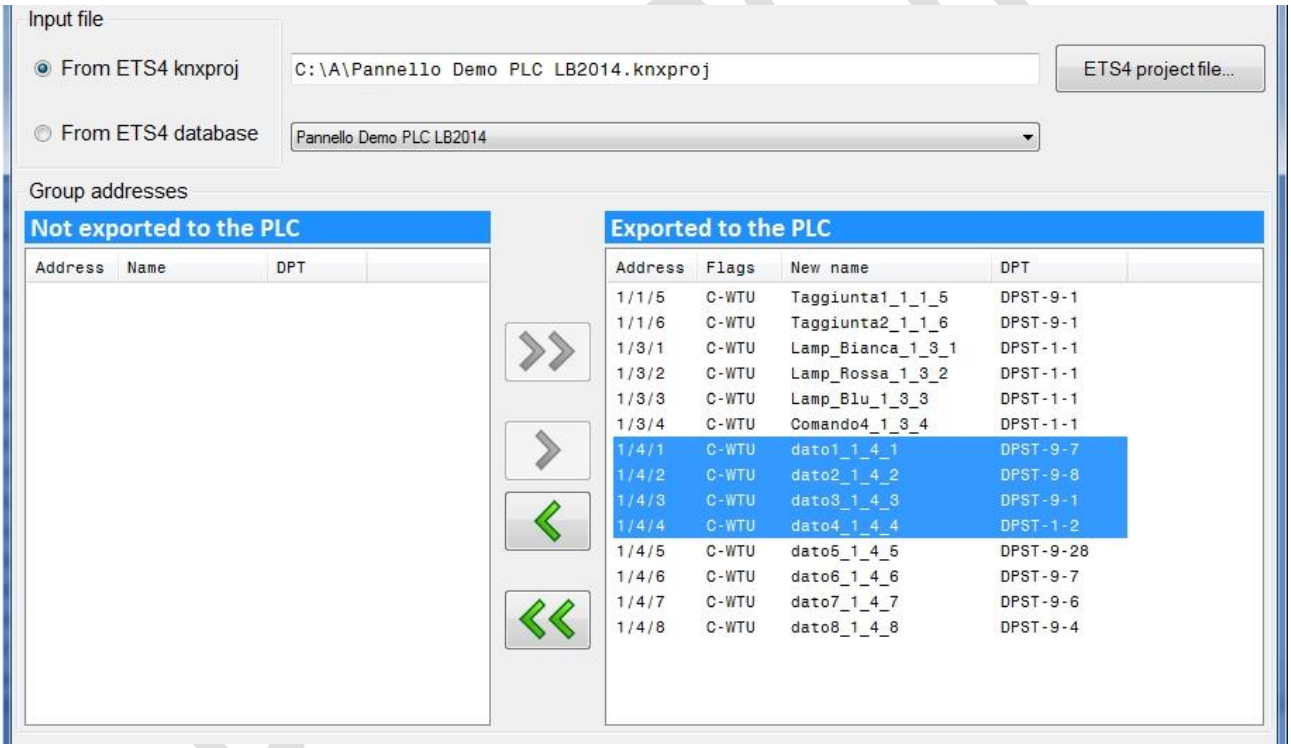
6.2.1 Project selection

The first step is to select an ETS4 project from which the group address list will be imported; the project source file can be in the form either of a `.knxproj` project or a `.mdf` ETS database.

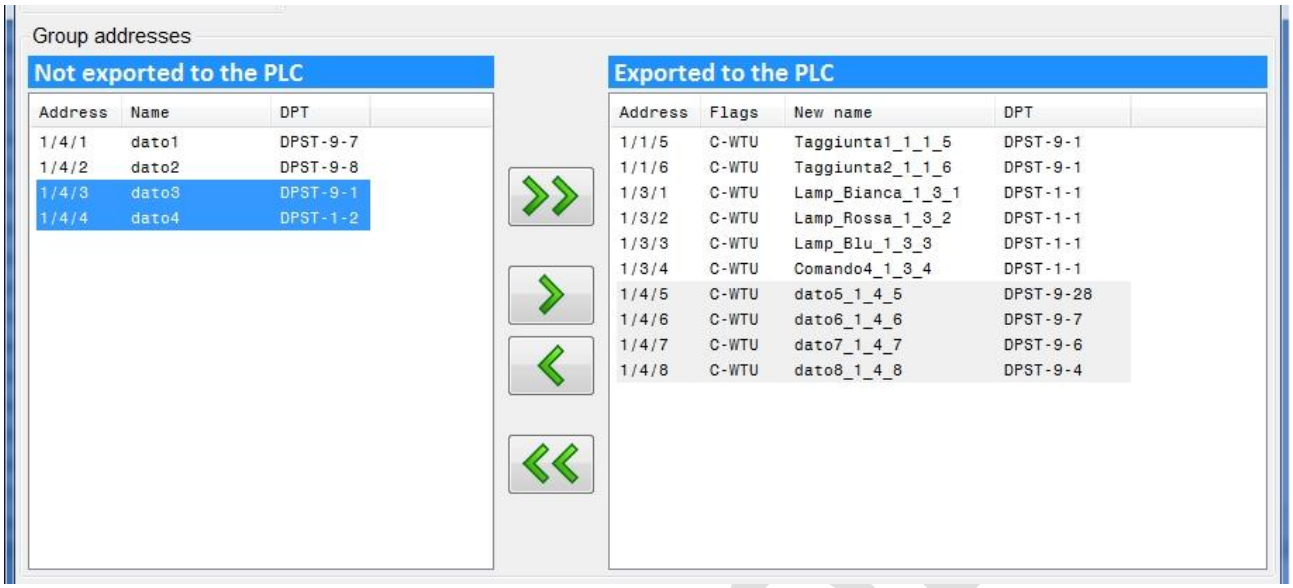
Either way, after the user has selected the desired file, the list of the contained projects (only one in the case of a `.knxproj`) appears in the combo box.

6.2.2 Selection of group addresses

The group addresses of the selected project are read and displayed in the selection panes: as a default, all group addresses are initially listed for export, i.e. a PLC variable will be created for each one of them.



If not all addresses are required in the output file, or some excluded addresses later need to be brought back into the export set, they can be selected one or more at a time (through the usual Windows operations mouse left click, Shift-click and Ctrl-click) and then moved with the single-arrow buttons in the middle. Double arrows move the entire address block from one side to another.

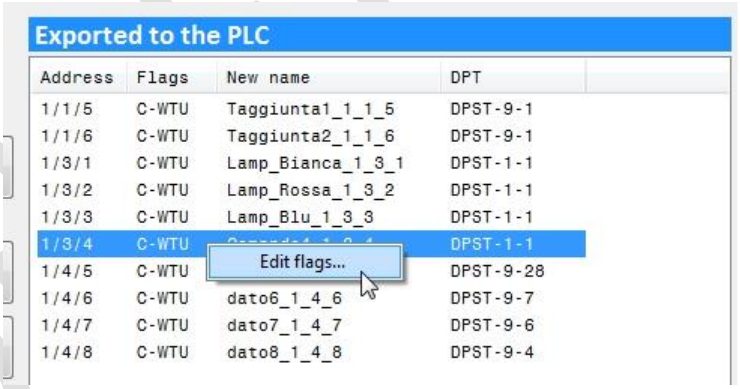


As a shortcut, an item can also be moved to the other pane with a left double-click, with the two exceptions described below.

6.2.3 Flag editing

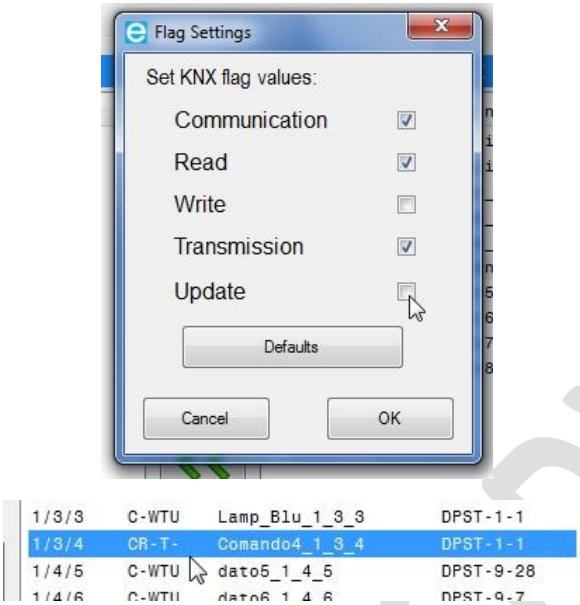
A double-click on the “Flags” field in the right pane allows the user to edit the flags that will be attributed to the variable in the PLC; this can also be done through the command in the right-click context menu .

These flags are exactly the same as the ones assigned in ETS to each variable instance bound to a group address, and have the same meaning; since they are related to the variable instance, and not to the group address itself, they must be defined also for the variable that is being created right now in the PLC program.

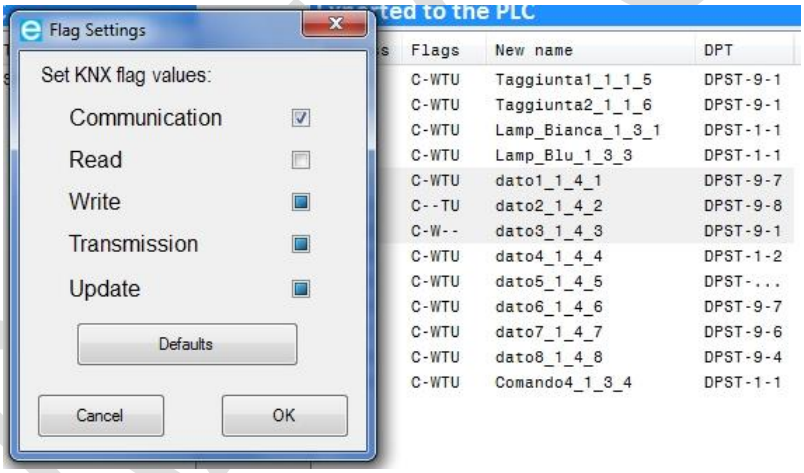


All newly exported variables will have the default flag value of `C/W/T/U`, which essentially means that the PLC variable takes its value from the bus whenever possible, and transmits its new value on the bus after any internal change, but it’s not readable by other devices. For further details see the section that describes the ekinex function library for the PLC.

The flag edit function brings up a flag selection dialog that allows the user to select the desired flag combination.



The flag editing function can be applied to several items at once: the modifications will be applied to all of the selected items. If any of the flags does not have the same value for all selected objects, its checkbox appears greyed; when confirming the dialog content, flags which have a greyed checkbox will not be modified. A button allows to quickly set the default value for all flags if desired.



If the item is moved back to the left pane, and later back to the right pane, the setting made by the user to the flags' value is kept.

6.2.4 Name editing

A double-click on the on the "New name" field in the right pane allows the user to edit the assigned name for the variable; the proposed name is formed with the information from the group address item, though it can be set to any name desired (within the constraints imposed for the variable names in CODESYS).

Warning: if the item is moved back to the left pane, and later back to the right pane, the modified name is not kept.

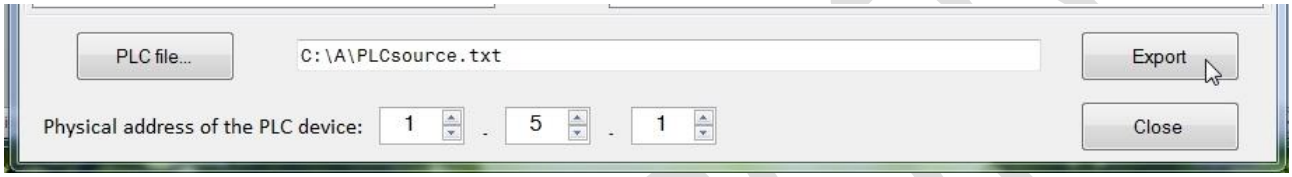
Variable names can also be easily modified later directly in the source code generated by the tool.

6.2.5 Setting the PLC address

All the group addresses encountered so far are defined in ETS, and therefore they are used by devices that "exist" (i.e. have a physical address) in the KNX project at hand. The PLC though, which has not been defined in the ETS configuration, does not yet have an "identity", that is, a physical address of its own.

In order for it to communicate on the bus, such physical address must be assigned. It is not essential that this address is "known" within ETS (though of course it is required to be unique); it is essential though that it is known, and used, by the PLC.

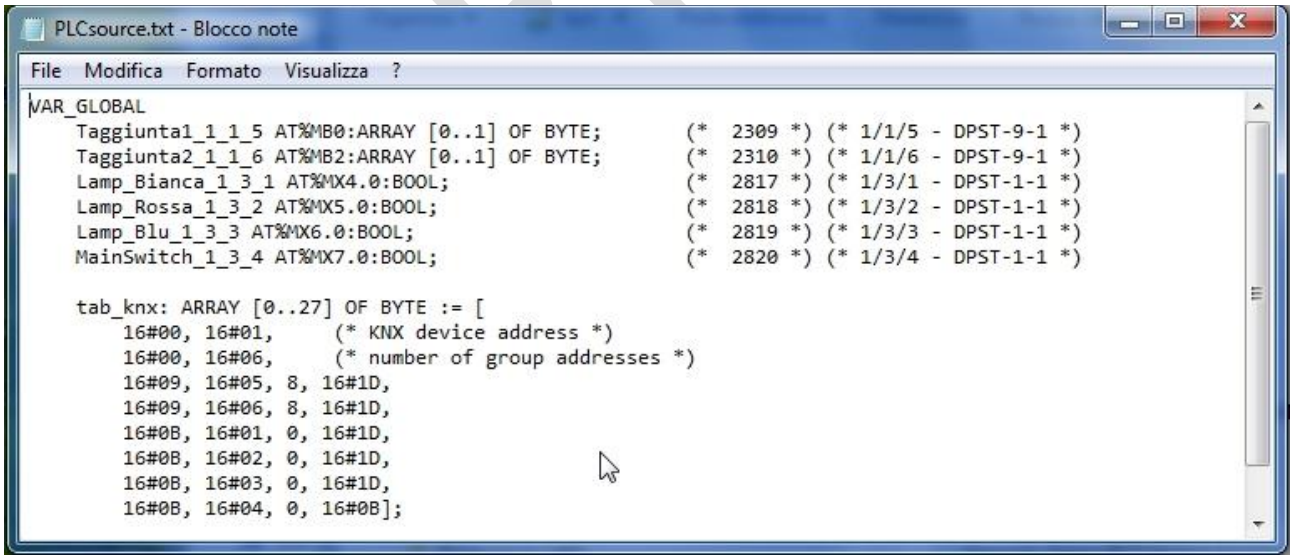
For this reason, the physical address of the PLC can be conveniently configured through the corresponding selection boxes; this does a little more than insert a corresponding line in the output file, though it is a convenient reminder.



6.2.6 Production of the output file

Once all settings have been made, the destination file (which is a pure text format) can be selected with the "*PLC file...*" button; the file name appears in the text box. Clicking on the "*Export*" button writes the output to the file and opens it in the default system editor; from there it can be further edited, verified, copied and pasted directly into the CODESYS PLC source code (in the *Global Variable List* section mentioned above).

In the following picture there is an example of the output generated by the sample project used in this section: it can be compared with the above pictures for information.



The content of this file is mostly for internal use of the PLC program compiler, although a few elements can be easily recognized.

The variable listing in the first part can easily be modified by the user; for its exact meaning and syntax description, please refer to CODESYS language documentation.

Once the desired result is obtained, the program can be closed.

Warning: closing the program causes all selections and modifications made by the user to be lost. Currently, there is no provision for saving a project export setup.

6.2.7 Later modifications to a project

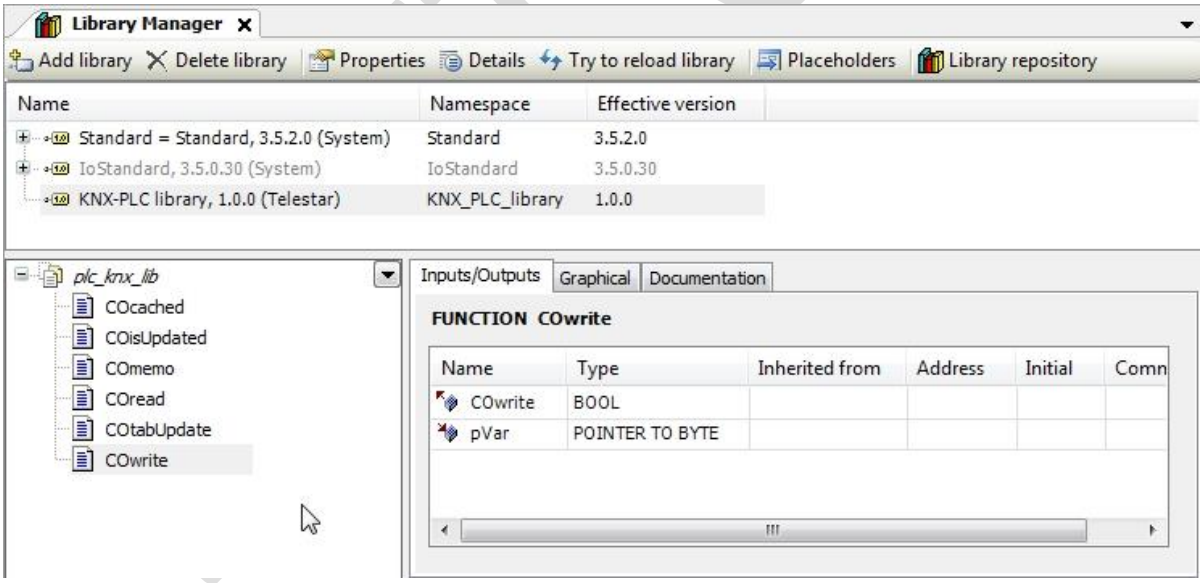
Sometimes, after data have been exported from a project and used in the associated PLC program, later changes to the project require to be also reflected in the PLC program, making a new export necessary.

Since there is currently no way to store the changes and settings made by the user during the export procedure, such activity would require to input all settings all over again, with the risk of introducing typos or other kinds of inconsistencies.

It is therefore advisable, whenever changes are limited, to directly modify the “translated” text in the PLC source code; in case of larger modifications that affect the project to a wider extent, a new “transitional” export file can be produced from the updated project. In this file it is not particularly important to compile all the same names previously assigned to variables (as long as they remain identifiable), but rather to be able to correctly extract the variable type definition (first part of the output listing) and allocation information (second part). This information can then be used for comparison with the current PLC source version in order to identify and manually perform the required changes.

6.3 Description of the KNX PLC library

The ekinex KNX PLC library contains a few functions that handle the interface variables with regard to their peculiarities as instances of KNX objects.



6.3.1 General handling of KNX interface variables

There are two actual instances of the interface values: those managed by the KNX firmware stack (the KNX objects) and those in the PLC memory area (the PLC variables). These instances are normally kept synchronized by the system:

- if a communication object in the KNX stack changes value, the corresponding variable is automatically updated at the beginning of the cycle.

- if a variable changes value within a PLC cycle, its value is automatically transmitted to the bus at the end of the cycle;

The library functions allow to perform a further operations on these values (such as forcing the value transmission etc.).

KNX objects are sent on the bus or received under definite conditions. This behaviour is well-defined through the use of KNX flags; the flags associated to each variable used are defined during the import phase.

Besides the behaviour defined by flags, which is handled at the KNX communication stack level (also as a lower level with respect to the PLC program), it is also possible to force the transmission or request of data in an explicit way through the library functions which are described later in this section.

The standard behavior of flags is summarized in following table:

C	Communication	Acts as a “connection switch” for the object with respect to communication; an object for which the communication flag is not set is effectively isolated from the bus (i.e. can be seen as non-existent from this point of view).
R	Read	States that this object is authoritative for any read request that shoud be issued by a device on the bus; this means that the device which contains this instance of the object will be the one (and must be the only one) that responds to a read request.
W	Write	Whenever a new value corresponding to the object’s group address is notified on the bus, this flag defines that the object should take on the value. If the flag is false, the value of the object is unaffected by bus messages.
T	Transmit	States that an internally triggered change of value for this object must cause a telegram to be sent on the bus to notify the new value. If the flag is false, the value can be changed by the device, but modified values are not transmitted outside.
U	Update	This flag acts similarly to the Write flag, but with an important distinction. Whereas the Write flag refers to explicit value notifications (which normally occur upon value changes and are “broadcast” for the purpose of keeping all objects up to date), the Update flag causes the value to be modified even in the case that a request for that group address is issued on the bus and a device sends the value (which may not have, and usually hasn’t, changed) in reply. Update allows a sort of “eavesdropping” to take advantage of data exchanges already taking place in order to ensure, at the earliest convenience, that objects are current (e.g. for initialization after startup).

6.3.2 Function *COtabUpdate*

IN	*Pointer to PTab*	*Base address of the Group Address definition table*
OUT	*Bool*	*Return value*

Initializes or updates the definition of the Communication object table in the KNX stack according to the PLC variable table definition.

This function must be called once at the beginning of the PLC program execution; the input argument is the starting address (named tab_knx in the standard import file) of the object definition table.

```
// In the variable definition section of the program unit:
initTab: BOOL := TRUE;

//...

IF initTab THEN
     COtabUpdate(ADR(tab_knx));
     initTab := FALSE;
END_IF
```

6.3.3 Function *COisUpdated*

IN	*Pointer to PVar*	*address of the referenced object variable*
OUT	*Bool*	*Return value*

Returns TRUE if the referenced variable has been updated from the bus since last access.

An associated internal update flag is reset after the function is called (and also when a *CORead* is performed: see *CORead* description), and is set when a bus update occurs.

```
IF COisUpdated(ADR(Switch1_1_3_1)) THEN
     //...
END_IF
```

6.3.4 Function *COcached*

IN	*Pointer to PVar*	*address of the referenced object variable*
OUT	*Bool*	*Return value*

Updates the referenced variable with the value from the KNX stack, without issuing any request on the bus.

The two values would normally be already synchronized, unless a bus update has occurred from the start of the PLC cycle: only in this case *COcached* actually updates the variable value.

```
a = Switch1_1_3_1;
COcached(ADR(Switch1_1_3_1))
b = Switch1_1_3_1;
IF a <> b THEN
     //...
END_IF
```

6.3.5 Function *COread*

IN	*Pointer to PVar*	*address of the referenced object variable*
OUT	*Bool*	*Return value*

Issues an update request on the bus for the referenced object.

The variable is updated with the value from the KNX stack (same as performing a *COcached*), and the function returns immediately. When the requested value is received from the bus, the variable will be

updated at the next beginning of a cycle, or otherwise it can be monitored with *COisUpdated* and explicitly read as soon as available.

A call to *COread* also resets the internal update marker of the object, in order to allow the immediate use of *COisUpdated*.

```
a = Switch1_1_3_1;
// Send read request
// (also resets update marker)
COread(ADR(Switch1_1_3_1));
// wait for the reply from the bus
// ... do something
IF COisUpdated(ADR(Switch1_1_3_1)) THEN
     b = Switch1_1_3_1;
     IF a <> b THEN
          //...
     END_IF
END_IF
```

6.3.6 Function *COmemo*

IN	*Pointer to PVar*	*address of the referenced object variable*
OUT	*Bool*	*Return value*

Writes the value of the referenced variable to the KNX stack, without issuing any transmission on the bus.

When a variable value is updated by the PLC, the corresponding object is flagged as modified and therefore marked for transmission at the end of the cycle. The *COmemo* function essentially resets this flag, thus allowing to modify the variable without issuing unneeded transmission requests; the transmission can e.g. be performed later or at scheduled intervals.

```
start = TIME();
//...
IF (TIME() - start) < T#10s THEN
     // Accumulate maximum value over period
     IF ValueA > ValueB_1_3_9 THEN
          ValueB_1_3_9 := ValueA;
          COmemo(ADR(ValueB_1_3_9));     // Prevent untimely transmission
     END_IF
ELSE
     // End of period:
     // trigger transmission now
     IF NOT transmissionDone THEN
          transmissionDone = TRUE;
          COwrite(ADR(ValueB_1_3_9));
     END_IF
END_IF
//...
```

6.3.7 Function *COwrite*

IN	*Pointer to PVar*	*address of the referenced object variable*
OUT	*Bool*	*Return value*

Writes the value of the referenced variable to the KNX stack, then force its immediate transmission on the bus.

Normally, whenever a (different) value is written to the variable, this is marked to be transmitted on the bus at the end of the PLC cycle. The *COwrite* function allows to force transmission immediately, and regardless whether the value has actually changed.

See function *COmemo* for an example.

6.4 Warning

- Installation, electrical connection, configuration and commissioning of the device can only be carried out by qualified personnel
- Opening the housing of the device causes the immediate end of the warranty period
- ekinex® KNX defective devices must be returned to the manufacturer at the following address: EKINEX S.p.A. Via Novara 37, I-28010 Vaprio d'Agogna (NO) Italy

6.5 Other information

- This application manual is aimed at installers, system integrators and planners
- For further information on the product, please contact the ekinex® technical support at the e-mail address: support@ekinex.com or visit the website www.ekinex.com
- KNX® and ETS® are registered trademarks of KNX Association cvba, Brussels