UM2409
User manual

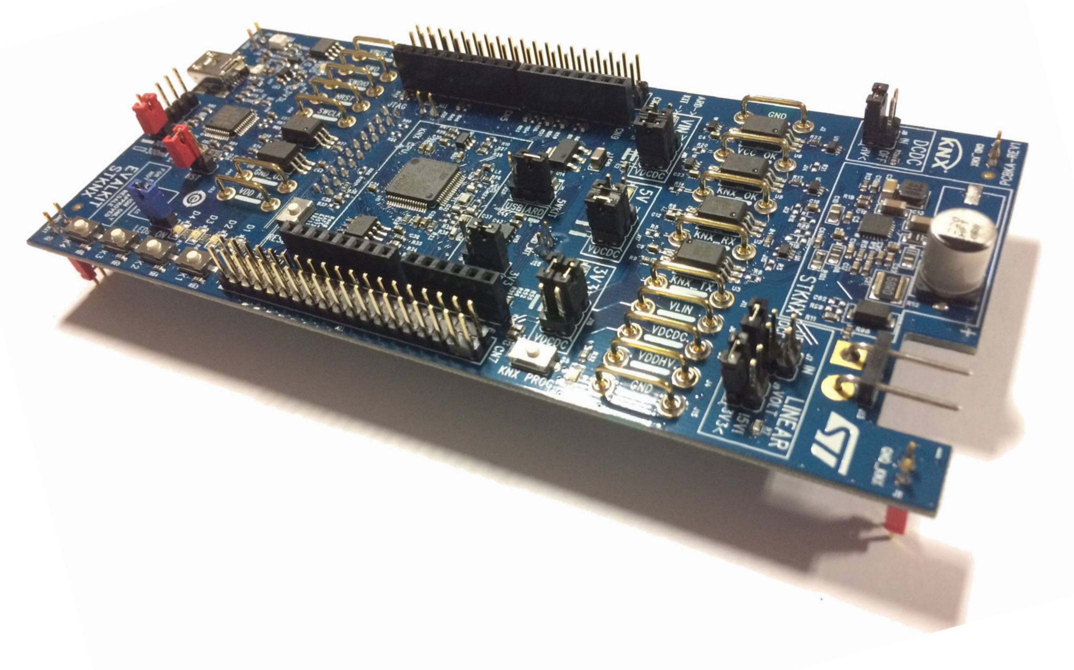
Quick start guide STKNX evaluation board (EVALKITSTKNX)

Introduction

The EVALKITSTKNX is a hardware platform to evaluate and to develop applications for the STKNX miniature transceiver with the STM32F103 microcontroller.

This document gives an overview of the demonstration software architecture and describes how to use this software with the evaluation kit.

Figure 1. STKNX evaluation board



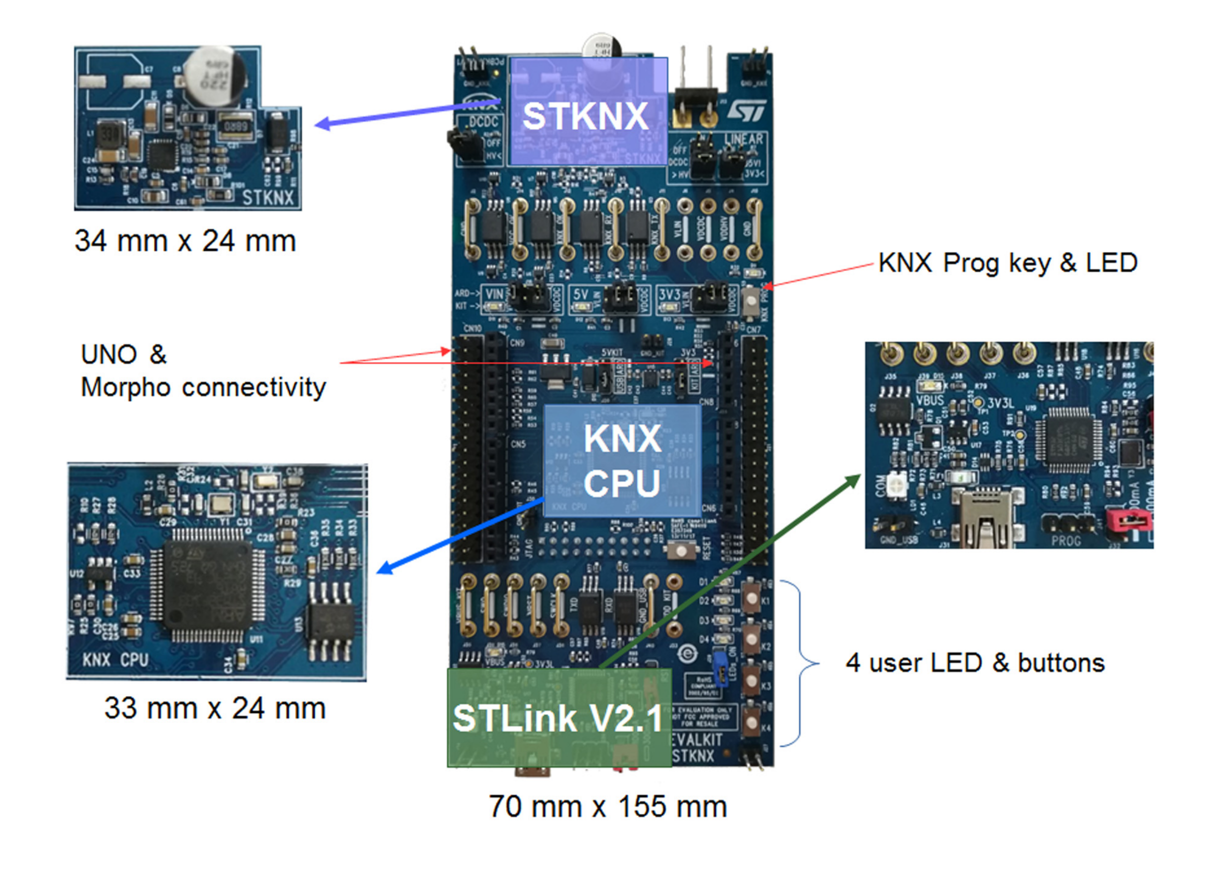
Contents

1 Overview

1.1 EVALKITSTKNX development platform overview

1.1.1 Board description

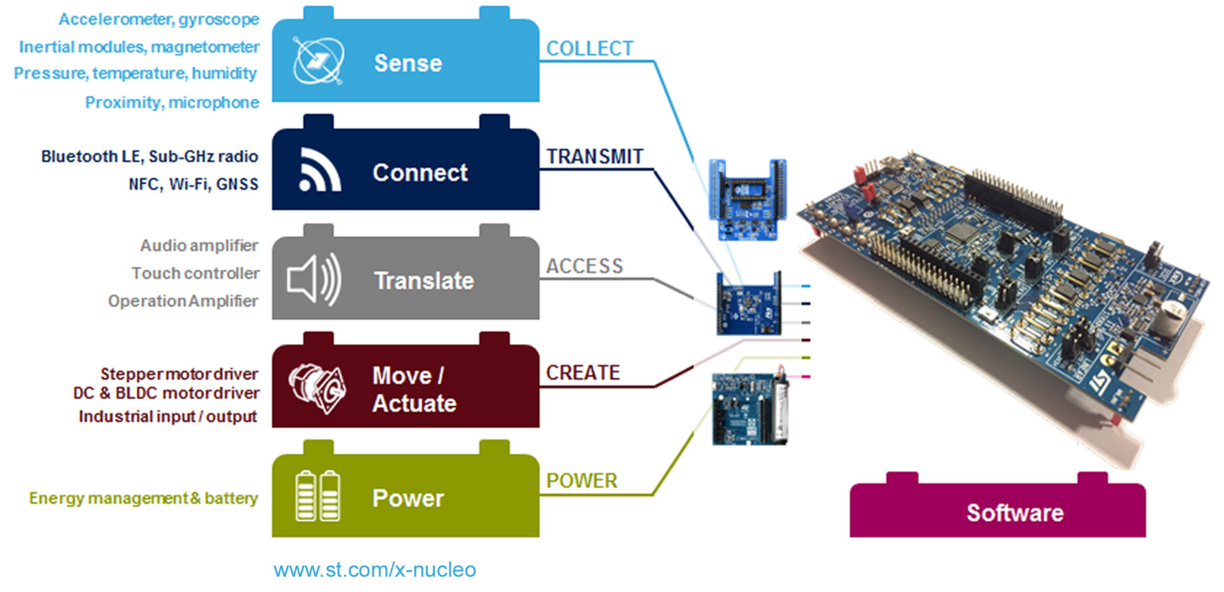
Figure 2. Board description



1.2 Development platform spirit

- The **EVALKITSTKNX** has been developed in the spirit of the STM32^{TM} Nucleo boards.
- Expansion boards with additional functionality can be **plugged directly on top of the Eval Kit** development board or stacked on another expansion board.

Figure 3. Building blocks



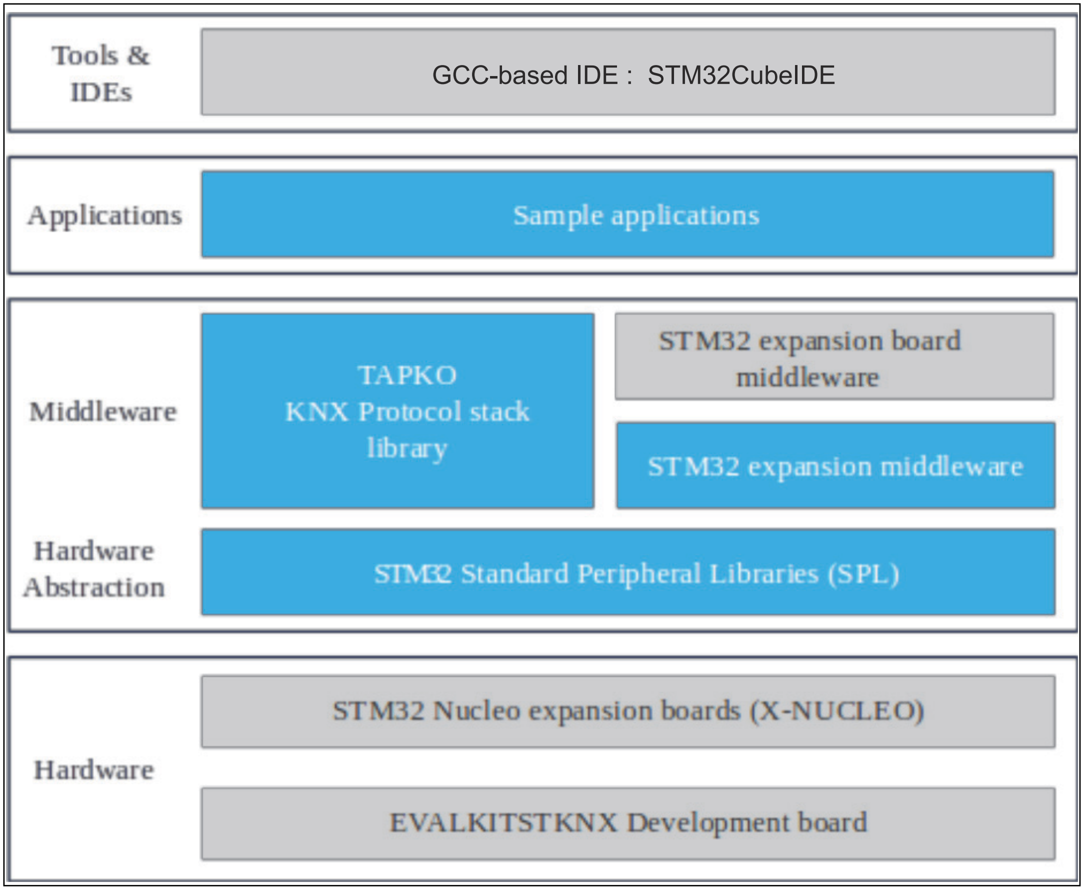
2 Software overview and setup

2.1 Software components

- The **EVALKITSTKNX** comes with a complete software package. A sample application is provided.
- Access to hardware features is simplified with the utilization of the **STM32 Standard Peripheral Libraries** (SPL)
- A demonstration version of the **TAPKO's KNX protocol stack** is provided as a binary file.
- The software package is compatible STM32CubeIDE. Versions of this IDE exist for Windows®, Linux®, and macOS®.

Note: As an alternative, Tapko© also provide a demonstration firmware for the EVALKITSTKNX: https://www.tapko.de/fr/products/detail/detail/News/kaistack-evalkitstknx/

Figure 4. System architecture

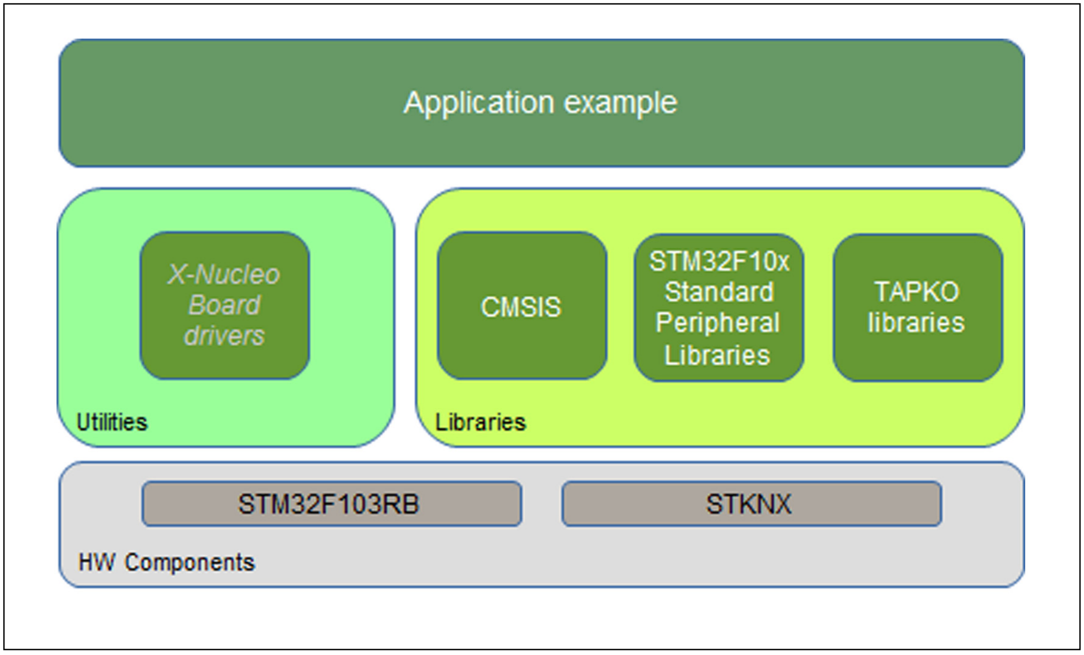

Tools & IDEs
GCC-based IDE : STM32CubeIDE
Applications
Sample applications
Middleware
TAPKO KNX Protocol stack library
STM32 expansion board middleware
STM32 expansion middleware
Hardware Abstraction
STM32 Standard Peripheral Libraries (SPL)
Hardware
STM32 Nucleo expansion boards (X-NUCLEO)
EVALKITSTKNX Development board

2.2 Software overview

2.2.1 STSW-KITSTKNX software description

- This software, running on the STM32F103, demonstrates the STKNX capabilities.
- It is built on the top of the STM32 Standard Peripheral Libraries (SPL) that eases access to STM32 features.
- It uses a demonstration version of the TAPKO KNX protocol stack.
- Example to demonstrate actuator and sensor.

Figure 5. Overall software architecture



2.3 Software prerequisites

- A Linux computer, a Windows computer or a Mac OSX computer with STM32CubeIDE development environment.
- EVALKITSTKNX firmware example
- ST-LINK/V2-1 USB driver in case of Windows® OS
- ST-LINK/V2-1 firmware upgrade
- KNX ETS5 (Engineering Tool Software) Application
  - Official tool, provided by KNX.org, used to configure KNX network
  - For the demonstration, version 5.6.3 is used

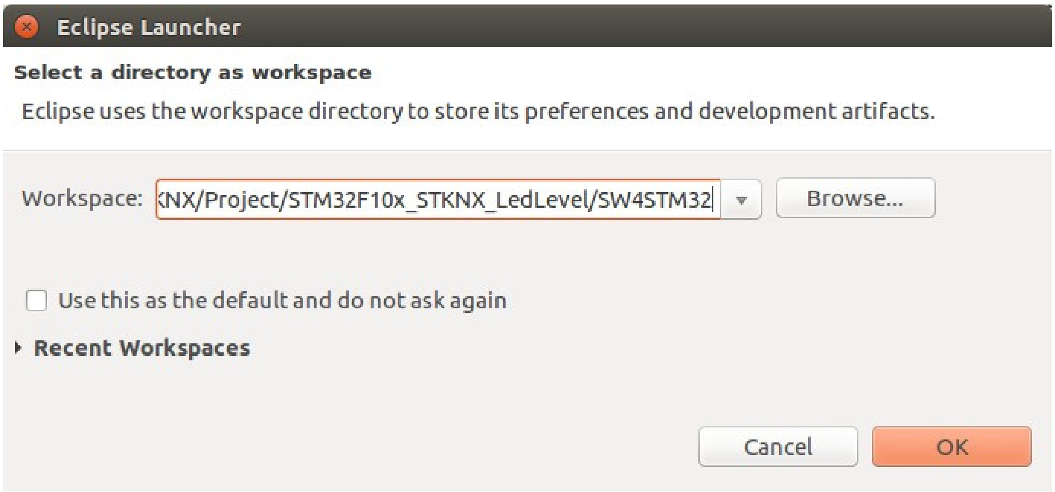
Warning: You must register to have access to download page

2.4 Install and build firmware with STM32CubeIDE

1. Download the STSW-KITSTKNX package and extract it in your file system.

Open STM32CubeIDE and when requested to select a directory as workspace, browse to:
<your path>/STM32F103RB-STKNX/Project/<Project example>/SW4STM32
with <Project example> that can be STM32F10x_STKNX_inOut4,
STM32F10x_STKNX_LedLevel or STM32F10x_STKNX_PseudoLedDimmer

Figure 6. Workspace selection



Eclipse Launcher

Select a directory as workspace

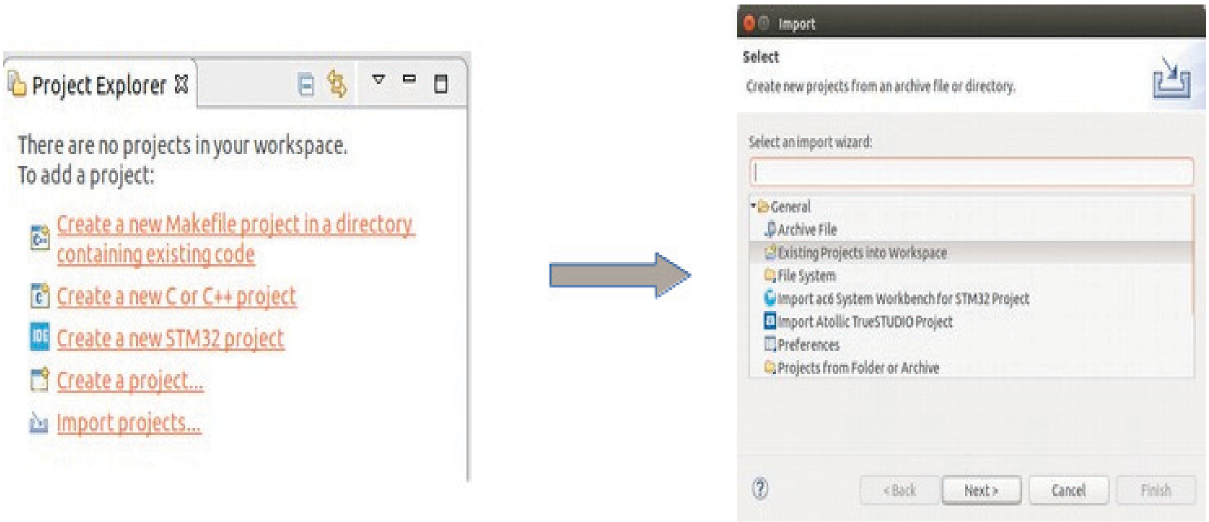
Eclipse uses the workspace directory to store its preferences and development artifacts.

Workspace: ‹NX/Project/STM32F10x_STKNX_LedLevel/SW4STM32

□ Use this as the default and do not ask again

2. In the **Project Explorer** panel, right click and select *Import* → *General* → *“Existing Projects into Workspace*”.

Figure 7. Import project

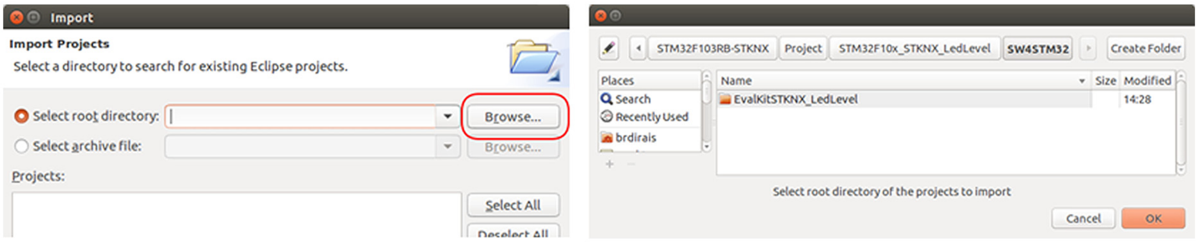


There are no projects in your workspace.
To add a project:

Project Explorer
Create a new Makefile project in a directory containing existing code
Create a new C or C++ project
Create a new STM32 project
Create a project...
Import projects...
Import
Select
Create new projects from an archive file or directory.
Select an import wizard:
General
Archive File
Existing Projects into Workspace
File System
Import ac6 System Workbench for STM32 Project
Import Atollic TrueSTUDIO Project
Preferences
Projects from Folder or Archive
< Back
Next >
Cancel
Finish

3. In the **Import** windows, click on *Browse* and on *OK* in the next window.

Figure 8. Select project directory



Import

Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory: | Browse...

○ Select archive file: Browse...

Projects:

Select All

Deselect All

STM32F103RB-STKNX Project STM32F10x_STKNX_LedLevel SW4STM32 Create Folder

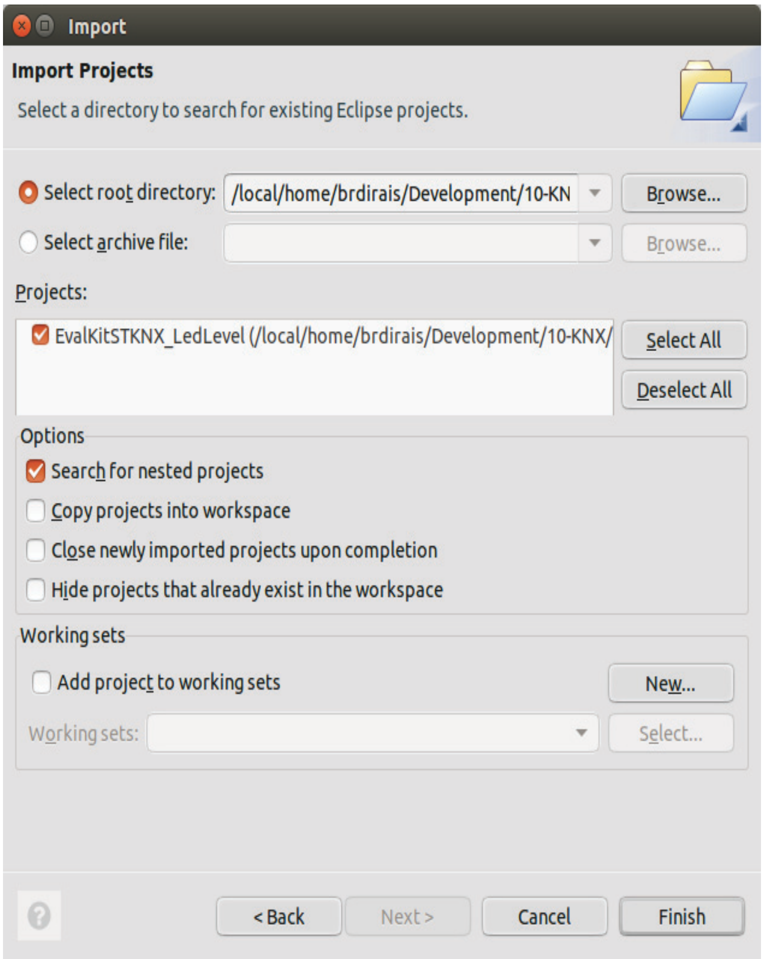
Places	Name	Size	Modified
Search	EvalKitSTKNX_LedLevel		14:28
Recently Used			
brdirais			

Select root directory of the projects to import

Cancel OK

4. The project to import has been automatically selected. You can now click on *Finish*. The project is imported.

Figure 9. Select project to import



Import

Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory: /local/home/brdirais/Development/10-KN Browse...

○ Select archive file: Browse...

Projects:

☑ EvalKitSTKNX_LedLevel (/local/home/brdirais/Development/10-KNX/

Select All

Deselect All

Options

☑ Search for nested projects

☐ Copy projects into workspace

☐ Close newly imported projects upon completion

☐ Hide projects that already exist in the workspace

Working sets

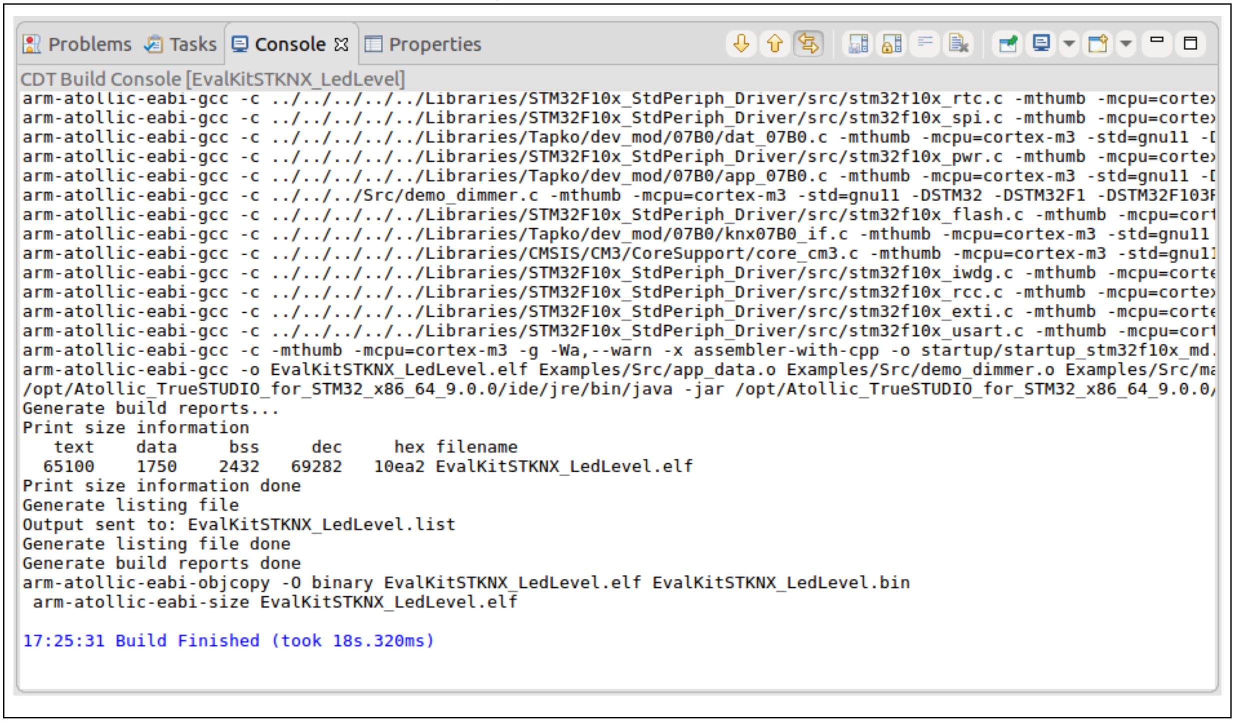
☐ Add project to working sets New...

Working sets: Select...

< Back Next > Cancel Finish

5. Select the project name in the **Project Explorer**, and select *Clean Project* in the contextual menu.
6. Click on the build icon or select *Build Project* in the contextual menu.

Figure 10. Console view



```
CDT Build Console [EvalKitSTKNX_LedLevel]
arm-atollic-eabi-gcc -c ../../../../../Libraries/STM32F10x_StdPeriph_Driver/src/stm32f10x_rtc.c -mthumb -mcpu=cortex
arm-atollic-eabi-gcc -c ../../../../../Libraries/STM32F10x_StdPeriph_Driver/src/stm32f10x_spi.c -mthumb -mcpu=cortex
arm-atollic-eabi-gcc -c ../../../../../Libraries/Tapko/dev_mod/07B0/dat_07B0.c -mthumb -mcpu=cortex-m3 -std=gnu11 -D
arm-atollic-eabi-gcc -c ../../../../../Libraries/STM32F10x_StdPeriph_Driver/src/stm32f10x_pwr.c -mthumb -mcpu=cortex
arm-atollic-eabi-gcc -c ../../../../../Libraries/Tapko/dev_mod/07B0/app_07B0.c -mthumb -mcpu=cortex-m3 -std=gnu11 -D
arm-atollic-eabi-gcc -c ../../../Src/demo_dimmer.c -mthumb -mcpu=cortex-m3 -std=gnu11 -DSTM32 -DSTM32F1 -DSTM32F103F
arm-atollic-eabi-gcc -c ../../../../../Libraries/STM32F10x_StdPeriph_Driver/src/stm32f10x_flash.c -mthumb -mcpu=cort
arm-atollic-eabi-gcc -c ../../../../../Libraries/Tapko/dev_mod/07B0/knx07B0_if.c -mthumb -mcpu=cortex-m3 -std=gnu11
arm-atollic-eabi-gcc -c ../../../../../Libraries/CMSIS/CM3/CoreSupport/core_cm3.c -mthumb -mcpu=cortex-m3 -std=gnu11
arm-atollic-eabi-gcc -c ../../../../../Libraries/STM32F10x_StdPeriph_Driver/src/stm32f10x_iwdg.c -mthumb -mcpu=corte
arm-atollic-eabi-gcc -c ../../../../../Libraries/STM32F10x_StdPeriph_Driver/src/stm32f10x_rcc.c -mthumb -mcpu=cortex
arm-atollic-eabi-gcc -c ../../../../../Libraries/STM32F10x_StdPeriph_Driver/src/stm32f10x_exti.c -mthumb -mcpu=corte
arm-atollic-eabi-gcc -c ../../../../../Libraries/STM32F10x_StdPeriph_Driver/src/stm32f10x_usart.c -mthumb -mcpu=cort
arm-atollic-eabi-gcc -c -mthumb -mcpu=cortex-m3 -g -Wa,--warn -x assembler-with-cpp -o startup/startup_stm32f10x_md.
arm-atollic-eabi-gcc -o EvalKitSTKNX_LedLevel.elf Examples/Src/app_data.o Examples/Src/demo_dimmer.o Examples/Src/ma
/opt/Atollic_TrueSTUDIO_for_STM32_x86_64_9.0.0/ide/jre/bin/java -jar /opt/Atollic_TrueSTUDIO_for_STM32_x86_64_9.0.0/
Generate build reports...
Print size information
   text    data     bss     dec     hex filename
  65100    1750    2432   69282   10ea2 EvalKitSTKNX_LedLevel.elf
Print size information done
Generate listing file
Output sent to: EvalKitSTKNX_LedLevel.list
Generate listing file done
Generate build reports done
arm-atollic-eabi-objcopy -O binary EvalKitSTKNX_LedLevel.elf EvalKitSTKNX_LedLevel.bin
 arm-atollic-eabi-size EvalKitSTKNX_LedLevel.elf

17:25:31 Build Finished (took 18s.320ms)
```

7. Once build is finished, you can start debugging the project. Click on the Debug icon or select *Debug As* → *Embedded C/C++ Application* in the contextual menu.

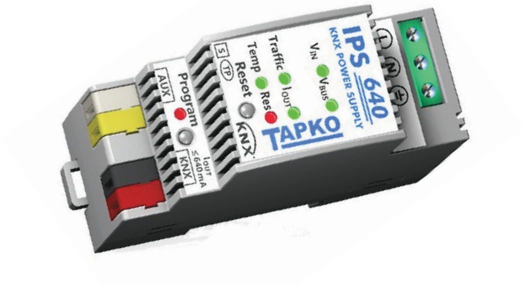
   Or you can drag and drop the binary (*EvalKitSTKNX_LedLevel.bin*), available in **Debug**, to the virtual drive that is mounted when you connect the EVALKITSTKNX board to your computer. The name of this virtual drive is STKNX.

2.5 Hardware prerequisites for all examples

The following hardware is required for all examples.

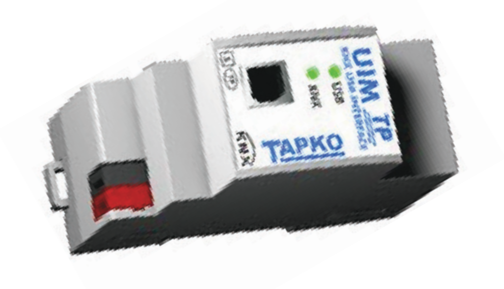
- One KNX power supply

Figure 11. KNX power supply



- One KNX TP interface (USB-KNX interface)

Figure 12. KNX TP interface



- A type A to type B USB cable

Figure 13. USB cable

3 Setup and demonstration examples

3.1 LedLevel example

3.1.1 Hardware prerequisites

In addition to the hardware described in *Section 2.5*, hardware prerequisites for this example are:

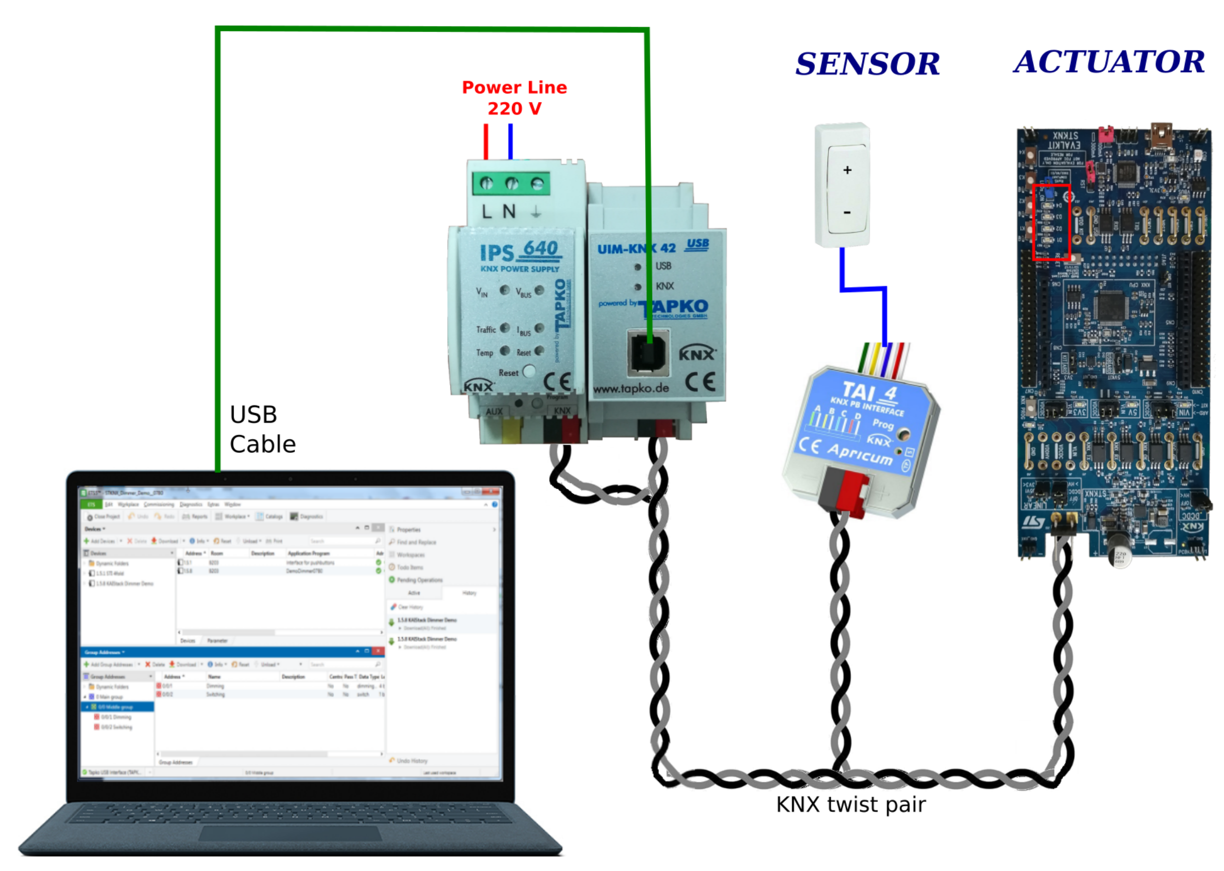
- One EVALKITSTKNX board
- One KNX sensor: Apricum TAI-KNX 4

Figure 14. TAI-KNX 4 module sensor

3.1.2 Setup the simple KNX network

1. Setup the KNX network in accordance to description in *Figure 15*.

Figure 15. LEDLevel demonstration network



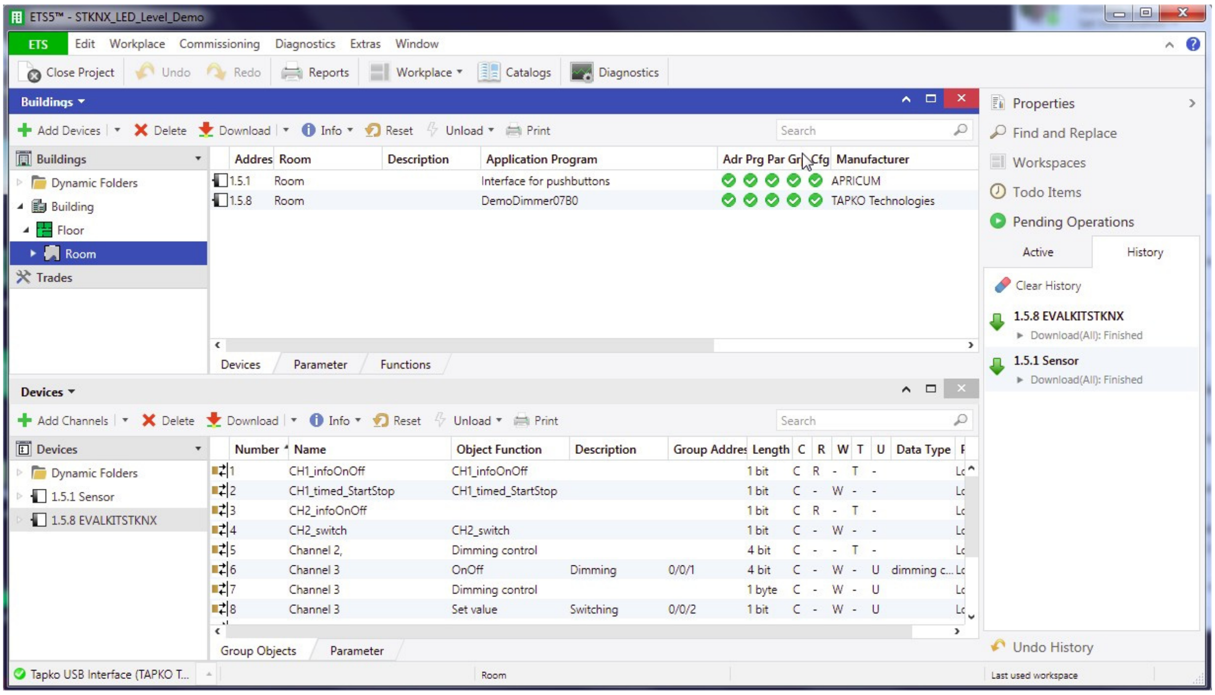
3.1.3 Network configuration with ETS5.

In the ETS5 application, import the project STKNX_LedLevel_Demo.knxproj available in STSW-KITSTKNX/Project/STM32F10x_STKNX_LedLevel/ETS5_ProjectFile

Download the full configuration in the two EVALKITSTKNX devices

1. Select the device
2. In Download menu, select “Download all”

Figure 16. ETS5 application screenshot



Addres	Room	Description	Application Program	Adr	Prg	Par	Gr	Cfg	Manufacturer
1.5.1	Room		Interface for pushbuttons	✓	✓	✓	✓	✓	APRICUM
1.5.8	Room		DemoDimmer07B0	✓	✓	✓	✓	✓	TAPKO Technologies

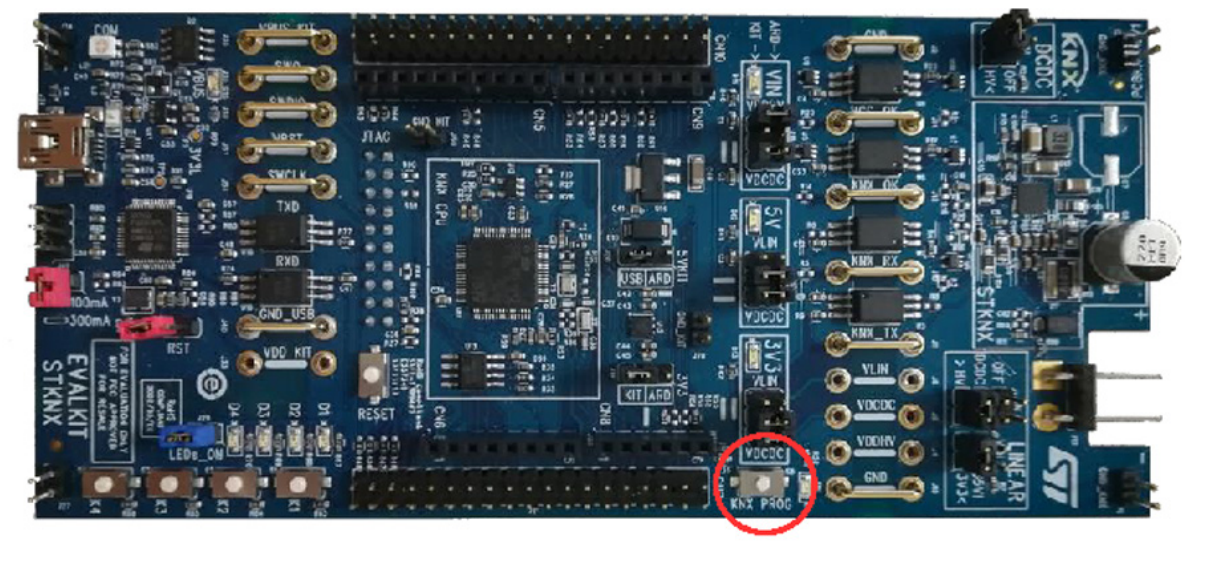
1.5.8 EVALKITSTKNX
Download(All): Finished

1.5.1 Sensor
Download(All): Finished

Number	Name	Object Function	Description	Group Addres	Length	C	R	W	T	U	Data Type	F
1	CH1_infoOnOff	CH1_infoOnOff			1 bit	C	R	-	T	-		Lc
2	CH1_timed_StartStop	CH1_timed_StartStop			1 bit	C	-	W	-	-		Lc
3	CH2_infoOnOff				1 bit	C	R	-	T	-		Lc
4	CH2_switch	CH2_switch			1 bit	C	-	W	-	-		Lc
5	Channel 2,	Dimming control			4 bit	C	-	-	T	-		Lc
6	Channel 3	OnOff	Dimming	0/0/1	4 bit	C	-	W	-	U	dimming c...	Lc
7	Channel 3	Dimming control			1 byte	C	-	W	-	U		Lc
8	Channel 3	Set value	Switching	0/0/2	1 bit	C	-	W	-	U		Lc

3. When asked in the ETS5 application, press the “KNX PROG” button of the EVALKITSTKNX board.

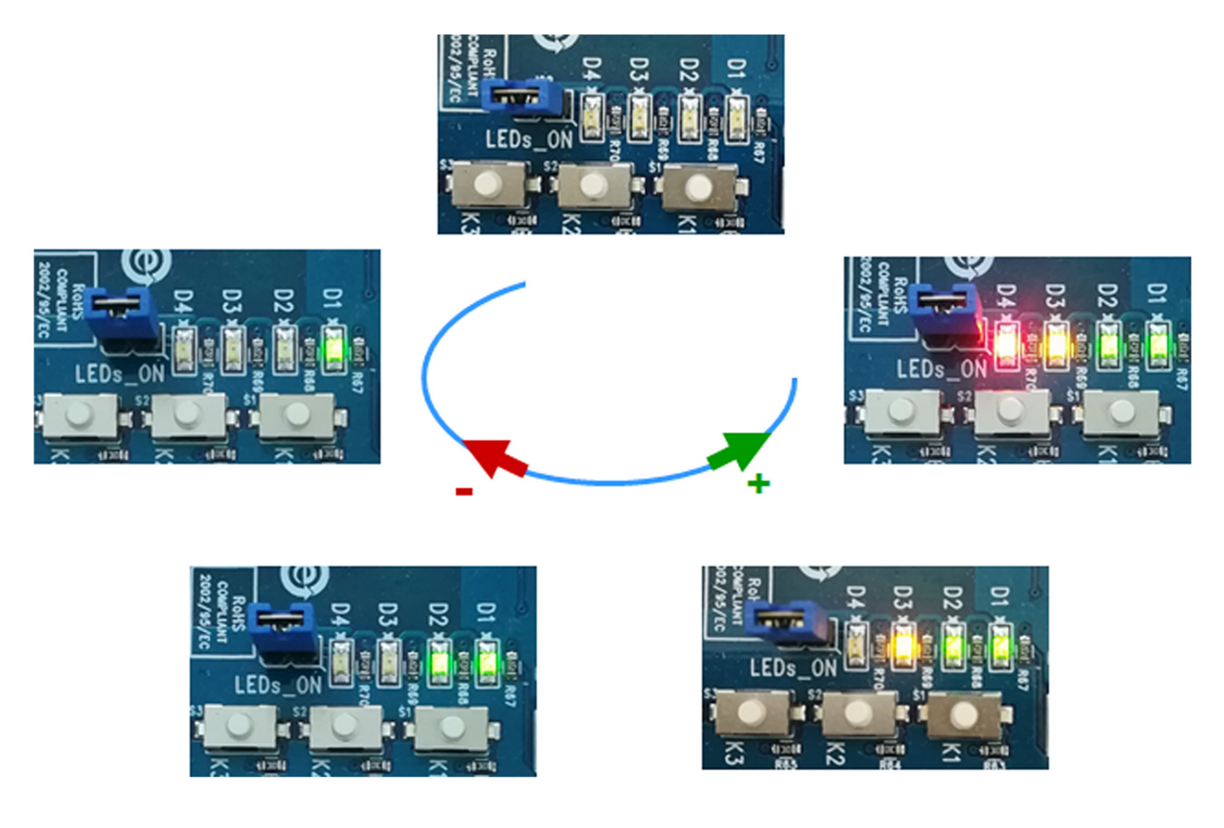
Figure 17. KNX PROG button on EVALKITSTKNX board



3.1.4 Tests:

- ON button of the sensor is used to increase the number of the LED switched on.
- OFF button of the sensor is used to decrease the number of the LED switched on.

Figure 18. Test description



3.2 4in – 4out exam

3.2.1 Hardware prerequisites

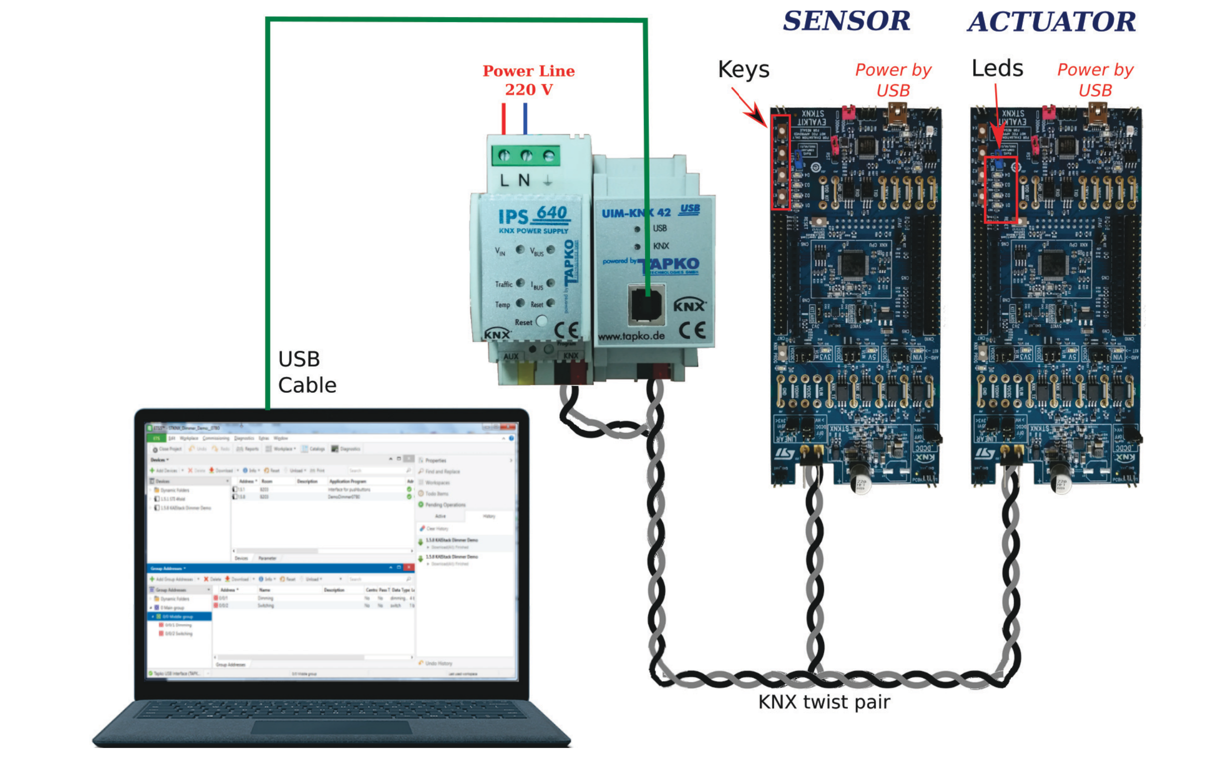
In addition of hardware descibed in *Section 2.5*, hardware prerequisites for this example is:

- Two STSW-KITSTKNX boards

3.2.2 Setup the simple 4in-4out KNX network

Setup the KNX network in accordance to description in *Figure 20*.

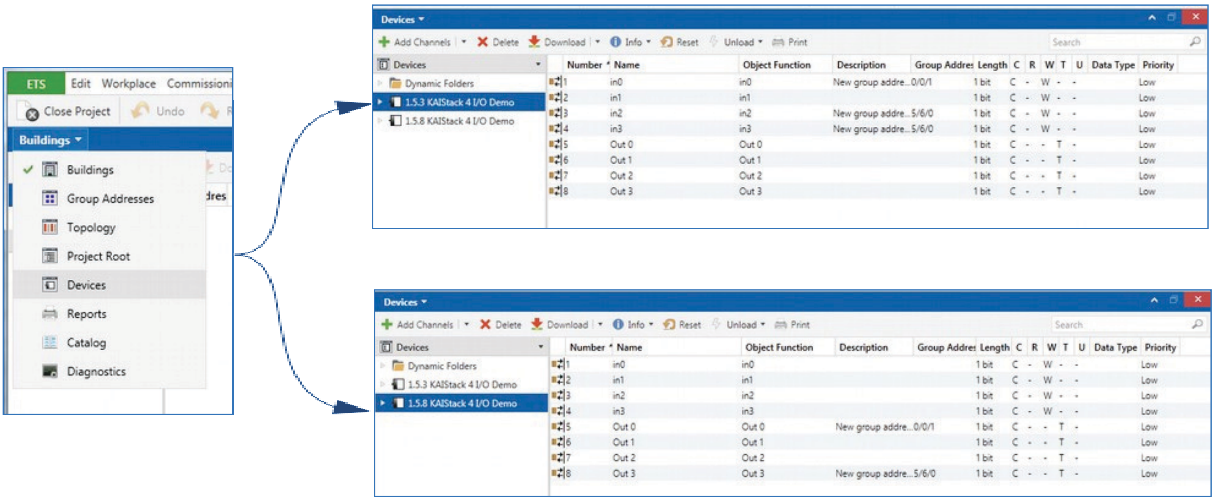
Figure 19. 4in - 4out KNX demonstration network



3.2.3 Network configuration with ETS5

- In the ETS5 application, import the project STKNX_4in4out_Demo.knxproj available in STSW-KITSTKNX/Project/STM32F10x_STKNX_LedLevel/ETS5_ProjectFile
- Once opened, in the Project workspace, select the Devices view. In this view, you can display the functions configured for the demonstration application as show in *Figure 21*.

Figure 20. ETS5 Devices view



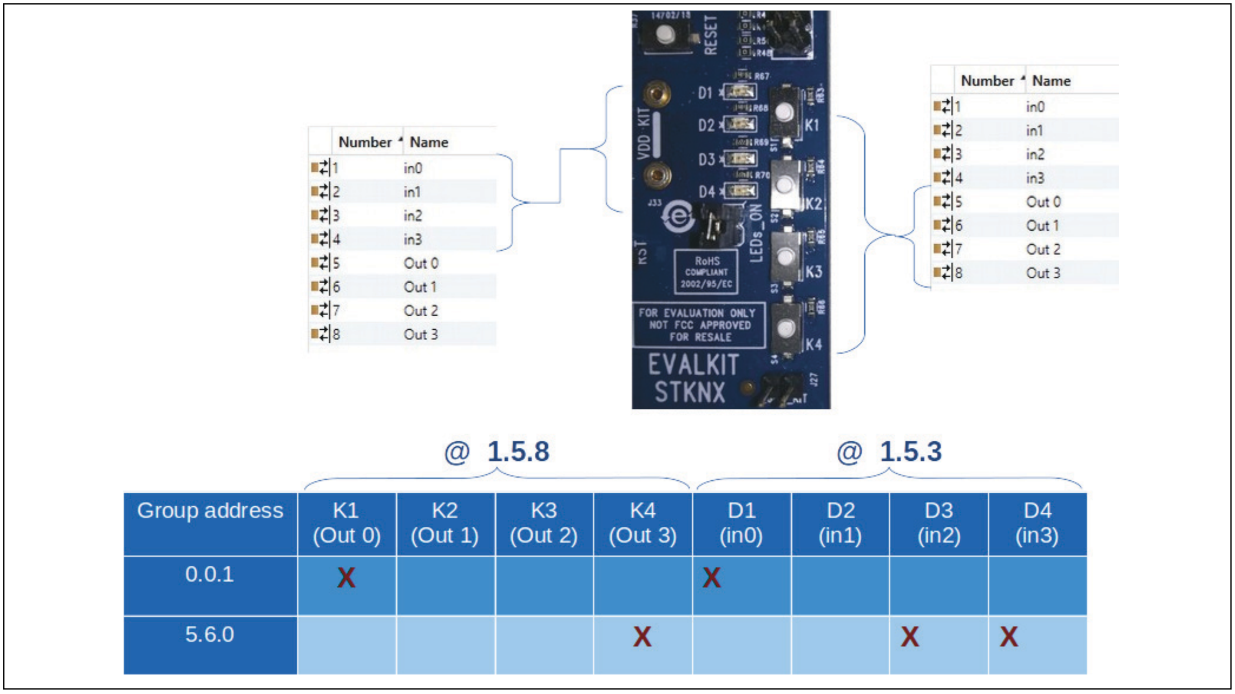
Number	Name	Object Function	Description	Group Addres	Length	C	R	W	T	U	Data Type	Priority
1	in0	in0	New group addre...	0/0/1	1 bit	C	-	W	-	-		Low
2	in1	in1			1 bit	C	-	W	-	-		Low
3	in2	in2	New group addre...	5/6/0	1 bit	C	-	W	-	-		Low
4	in3	in3	New group addre...	5/6/0	1 bit	C	-	W	-	-		Low
5	Out 0	Out 0			1 bit	C	-	-	T	-		Low
6	Out 1	Out 1			1 bit	C	-	-	T	-		Low
7	Out 2	Out 2			1 bit	C	-	-	T	-		Low
8	Out 3	Out 3			1 bit	C	-	-	T	-		Low

Number	Name	Object Function	Description	Group Addres	Length	C	R	W	T	U	Data Type	Priority
1	in0	in0			1 bit	C	-	W	-	-		Low
2	in1	in1			1 bit	C	-	W	-	-		Low
3	in2	in2			1 bit	C	-	W	-	-		Low
4	in3	in3			1 bit	C	-	W	-	-		Low
5	Out 0	Out 0	New group addre...	0/0/1	1 bit	C	-	-	T	-		Low
6	Out 1	Out 1			1 bit	C	-	-	T	-		Low
7	Out 2	Out 2			1 bit	C	-	-	T	-		Low
8	Out 3	Out 3	New group addre...	5/6/0	1 bit	C	-	-	T	-		Low

- In Device configuration:
  - LEDs D1 to D4 are linked to functions “in0” to “in3”

– Keys K1 to K4 are linked to functions “Out 0” to “Out 3”

Figure 21. Group address description



Number	Name
1	in0
2	in1
3	in2
4	in3
5	Out 0
6	Out 1
7	Out 2
8	Out 3

Number	Name
1	in0
2	in1
3	in2
4	in3
5	Out 0
6	Out 1
7	Out 2
8	Out 3

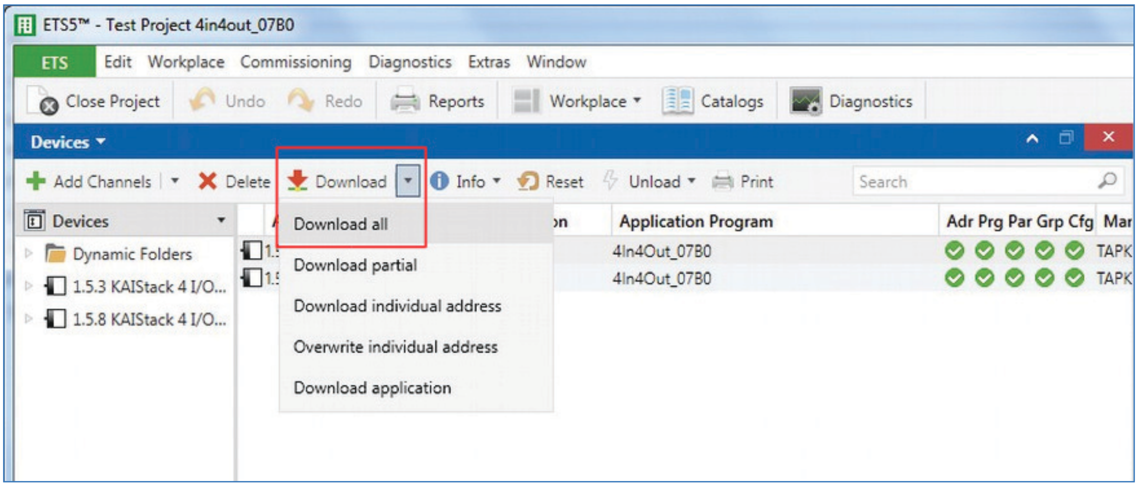
D1
D2
D3
D4
K1
K2
K3
K4
RESET
VDD KIT
LEDs_ON
RoHS COMPLIANT 2002/95/EC
FOR EVALUATION ONLY NOT FCC APPROVED FOR RESALE
EVALKIT STKNX
@ 1.5.8
@ 1.5.3

Group address	K1 (Out 0)	K2 (Out 1)	K3 (Out 2)	K4 (Out 3)	D1 (in0)	D2 (in1)	D3 (in2)	D4 (in3)
0.0.1	X				X			
5.6.0				X			X	X

- Download the full configuration in the two EVALKITSTKNX devices

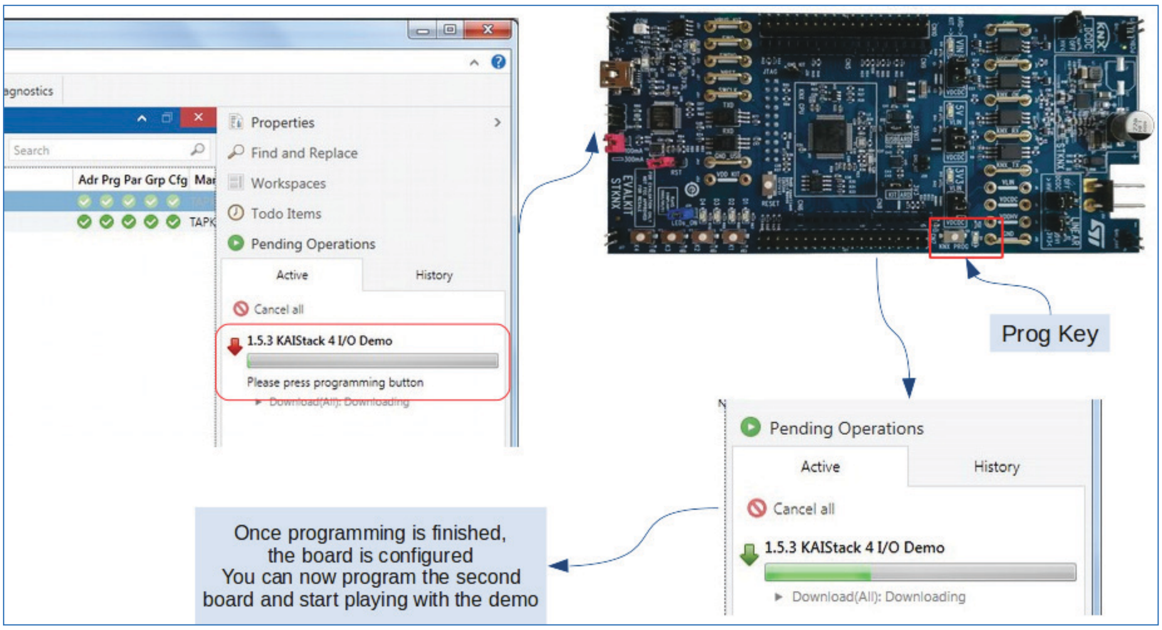
1. Select the device
2. In Download menu, select “Download all”

Figure 22. ETS5 download devices configuration



- To start download, the tool request you to press the “KNX PROG” key on the board.

Figure 23. Download process



1.5.3 KAIStack 4 I/O Demo

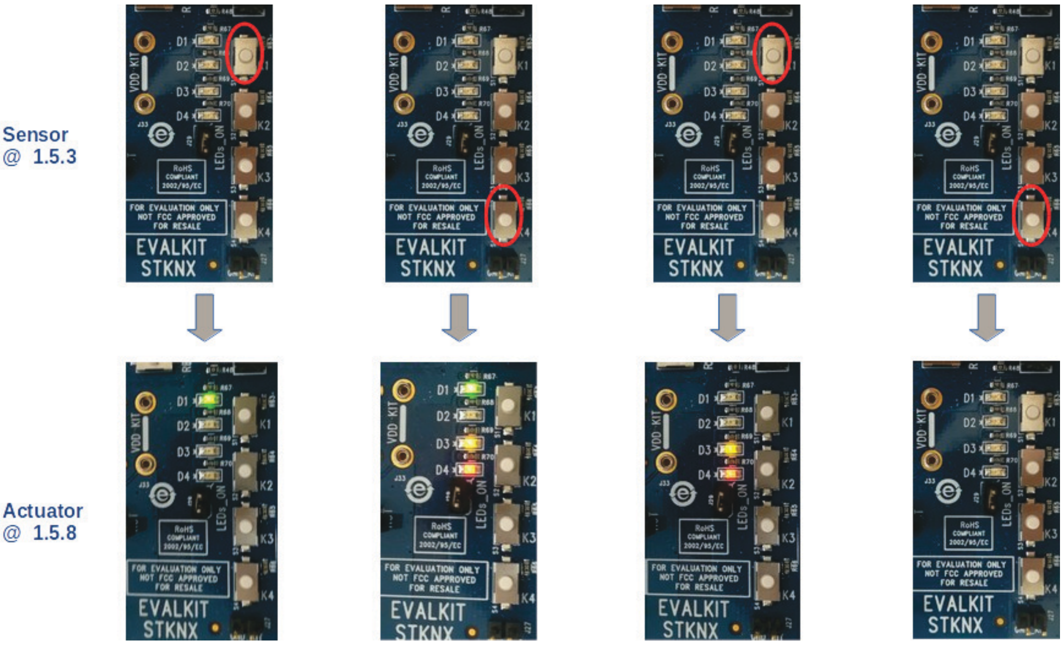
Please press programming button

Download(All): Downloading

Once programming is finished, the board is configured
You can now program the second board and start playing with the demo

3.2.4 Tests

Figure 24. Test description



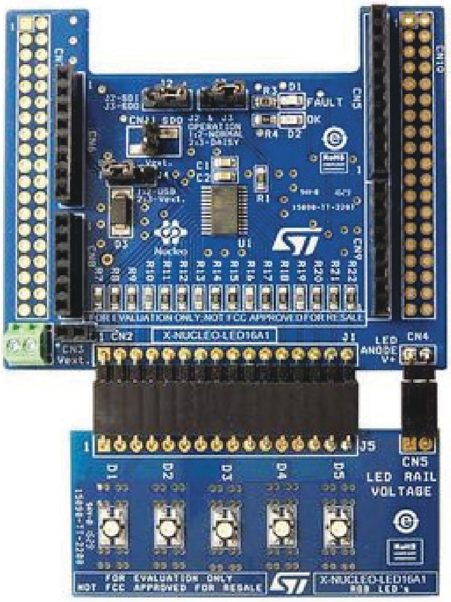
3.3 Pseudo LED dimmer example

3.3.1 Hardware prerequisites

In addition to the hardware described in *Section 2.5*, hardware prerequisites for this example are:

- One EvalKitSTKNX board
- One Led Driver Nucleo board X-NUCLEO-LED16A1

Figure 25. Nucleo board X-NUCLEO-LED16A1

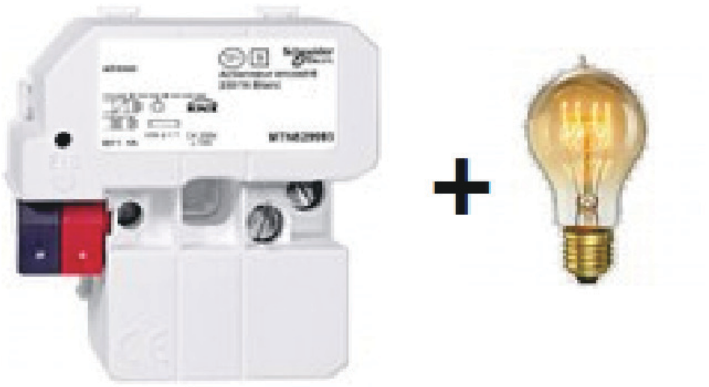


- One KNX sensor : Apricum TAI-KNX 4

Figure 26. TAI-KNX 4 module sensor

- One KNX bulb actuator (optional). Example : MTN629993

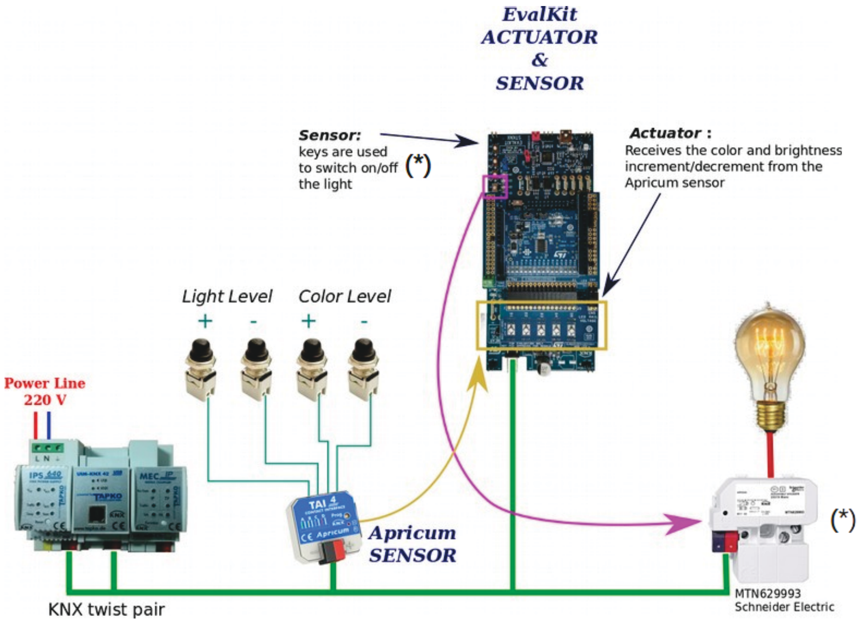
Figure 27. MTN629993 KNX bulb actuator



3.3.2 Setup the simple Pseudo LED Dimmer KNX network

- Setup the KNX network in accordance to description in *Figure 28*

Figure 28. Pseudo LED dimmer demonstration network



3.3.3 Network configuration with ETS5

- In the ETS5 application, import the project STKNX_PseudoDimmer_Demo.knxprojavailable in STSWKITSTKNX/Project/STM32F10x_STKNX_PseudoLedDimmer/ETS5_ProjectFile

- Refer to previous example for more information on how to import a KNX project in ETS5 and how to download the full configuration in both STI and EVALKITSTKNX devices

3.3.4 Tests

- Use the Light level and Color level connected to the Apricum sensor to change LED display on X-NUCLEO-LED16A1 board.
- Use the key 1 of the EvalKitSTKNX board to switch on/off the light.

4 Revision history

Table 1. Document revision history

Date	Revision	Changes
23-Jul-2018	1	Initial release.
28-May-2020	2	Updates made to the following Sections: *Section 2.3*, *2.4*,and *2.5* *Section 3.1*, *3.1.1*,and *3.1.3* *Section 3.2*, *3.2.1*, *3.2.2*, *3.2.3*, and *3.2.4* *Section 3.3*, *3.3.1*, *3.3.2*, *3.3.3*, and *3.3.4* The following Figures have been updated/added: *Figure 4*, *9*, *20*, *21*, *22*, *23*, *24*, *25*, *26*, *27*,and *28*

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2020 STMicroelectronics – All rights reserved