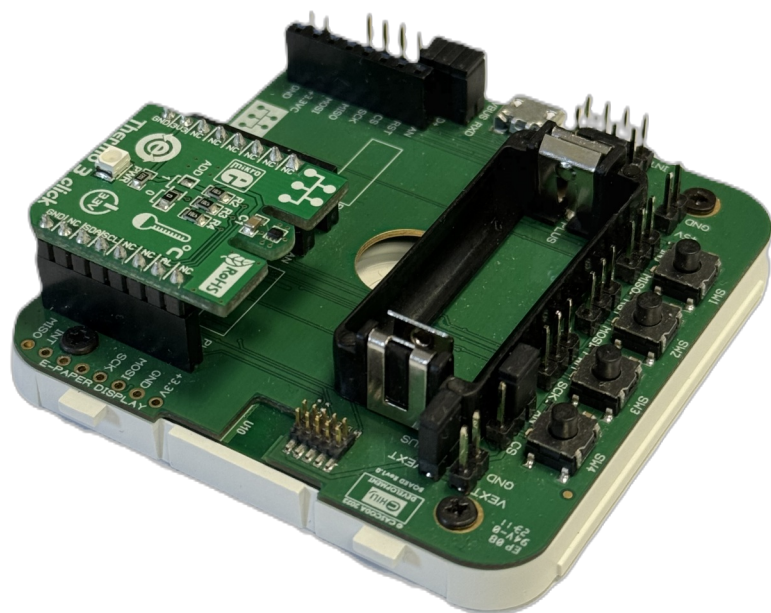
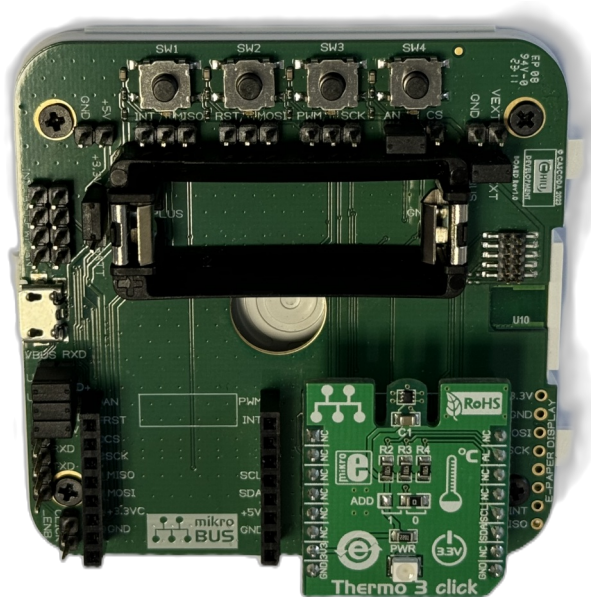


# Room Temperature Sensor Demo

## Manual



# Table of contents

- [Room Temperature Sensor Demo](#)
- [Table of contents](#)
- [Room Temperature Sensor Demo Features](#)
  - [About Cascoda's KNX IoT Development Board](#)
  - [Using the Devboard as a Room Temperature Sensor Demo](#)
  - [Demo Overview and Requirements](#)
- [General information](#)
  - [Document Version information](#)
  - [Used Terms](#)
  - [Safety instructions](#)
  - [Issues](#)
  - [Contact information](#)
- [Setup and Configuration](#)
  - [Device Startup](#)
  - [Hardware Setup](#)
  - [Flashing the Firmware](#)
    - [Troubleshooting](#)
  - [Commissioning](#)
    - [Thread Commissioning](#)
    - [KNX Commissioning](#)
      - [Adding the Device to a Topology](#)
      - [Creating a Configuration](#)
      - [Downloading the ETS configuration](#)
      - [Resetting the device](#)
- [Running the Demo with ETS](#)
- [Software Bill of Materials](#)
  - [Cascoda SDK](#)
  - [tinycbor](#)
  - [mbedtls](#)
  - [Openthread](#)
- [KNX device information](#)
  - [Data points](#)
  - [Parameters](#)

# Room Temperature Sensor Demo Features

The room temperature sensor demo runs on Cascoda's KNX IoT Development Board, equipped with the Thermo 3 Click from MikroElektronika.

## About Cascoda's KNX IoT Development Board

Cascoda's KNX IoT Development Board provides everything you need to develop your KNX IoT over Thread application with the Cascoda Chili module. It features Cascoda's SMARTRange™ technology to provide long-range Thread connectivity for whole-house coverage. In addition, Cascoda provides easy integration with Cascoda's dashboard and our KNX IoT Hub for easy configuration and backhaul connectivity.

The KNX-IoT Development Kit contains two Development Boards, and is available [from our distributors](#). You can view the [Getting Started Guide](#), and if you want detailed information on the Development Board you may [download the datasheet](#).

The KNX-IoT Development Board works seamlessly with ChiliCuisine™, Cascoda's low-code development environment.

### Features:

- Certified Thread-based radio supporting meshed networking, with long range
- Easy integration with Cascoda's Border Router for IPv4/IPv6 backhaul connectivity
- Ultra-low-power battery operation & battery charging
- Four programmable buttons & LEDs
- E-Paper display integration
- A library of sensor & actuator plug-in boards, to develop your solution
- Automatic code generation for your chosen target combination of sensors/actuators/switches/leds

### Specification:

- USB or UART connectivity
- 5V or 3.3V supply options
- 3.7V Li/LiPo battery integration with charging from a 5V supply or USB
- Two Mikroelektronika Click™ slots for a huge range of plug-in sensor/actuator connectivity
- Integrated chip antenna and an external antenna option
- Dimensions 66 x 64mm

# Using the Devboard as a Room Temperature Sensor Demo

Once the device is fully installed and commissioned, it operates as a room temperature sensor, which senses and sends the temperature once every 20 seconds. It can send the following data, to be received by another KNX entity (e.g. a thermostat, or the ETS group monitor):

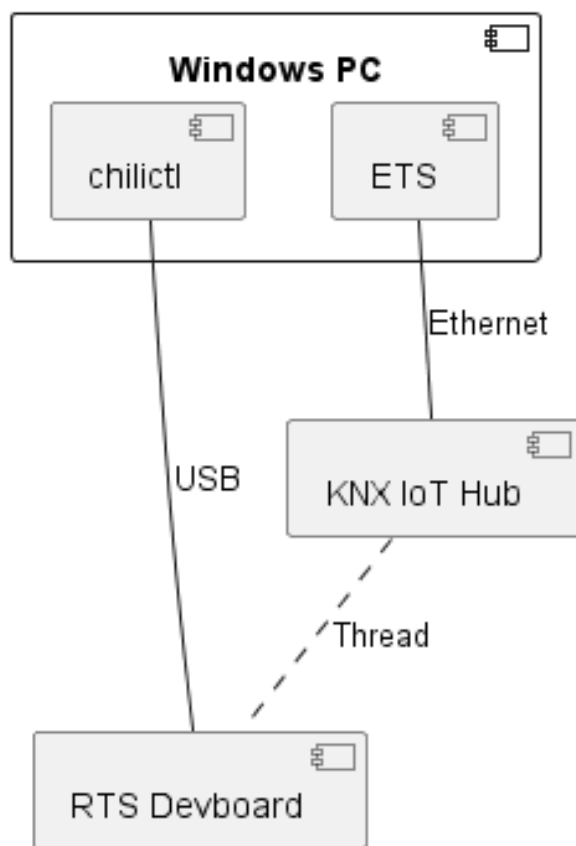
- RTS: Current temperature value

Therefore, using appropriate counterpart actuator, you can receive/observe the current temperature sensed by the device.

The Room Temperature Sensor demo can be powered either via USB, or with a CR123A 3V Non-Rechargeable Lithium Battery.

**NOTE!** This demo application is not low-power. It is therefore not meant to be run for very long periods of time using a battery. Please contact us if you require a low-power version of this demo.

## Demo Overview and Requirements



The Demo consists of using the Group Monitor on ETS to observe the temperature values that are sensed and sent by the Room Temperature Sensor. In order to do this, you will need the following:

- Hardware requirements
  - A Windows PC
  - A Thread Border router. Cascoda recommends using the [KNX IoT Hub](#).
  - A development board with jumpers provided
  - Thermo 3 click
  - USB-A to Micro-USB cable
- Software requirements
  - ETS version 6.3.0 or later
  - Cascoda Windows Tools
  - An application binary
  - An application binary
  - A serial number binary
  - A project file to load into ETS

This manual will show you how to set up the hardware, flash the device, commission it onto the Thread network and onto KNX, and then finally being able to control it using ETS.

# General information

## Document Version information

This manual is amended periodically and will be brought into line with new software releases. The change status (date) can be found in the contents header. If you have a device with a later software version, please check [www.cascoda.com](http://www.cascoda.com) to find out whether a more up-to date version of the manual is available.

## Used Terms

Sign	Description
<b>DANGER!</b>	Indicates an immediately hazardous situation which will lead to death or severe injuries if it is not avoided.
<b>CAUTION!</b>	Indicates a potentially hazardous situation which may lead to trivial or minor injuries if it is not avoided.
<b>WARNING!</b>	Indicates a situation which may lead to damage to property if it is not avoided.
<b>NOTE!</b>	Indicates a situation which may lead to possible (known) side effects.

## Safety instructions

**CAUTION!** This demo can be powered either via USB or with a CR123A 3V Non-Rechargeable Lithium Battery. Since the battery is non-rechargeable, do *not* connect both the USB cable and the battery at the same time, as this may cause a fire! Do follow the exact jumper configurations mentioned in this document, making sure to remove and place the appropriate jumpers when switching between using USB or Battery.

## Issues

Questions about the product?

You can reach the technical service of Cascoda under Tel. +44 (0)2380 638 111 or [support@cascoda.com](mailto:support@cascoda.com).

We need the following information to process your service request:

- Type of appliance (model name or item number)
- Description of the problem
- Serial number or software version

- Source of supply (dealer/installer who bought the device from Cascoda )

For questions about KNX functions:

- Version of the device application
- ETS version used for the project

## Contact information

info@cascoda.com

Threefield House,

Threefield Lane,

Southampton,

SO14 3LP, UK

# Setup and Configuration

## Device Startup

The device starts up when plugged in via USB, or when a CR123A 3V Non-Rechargeable Lithium Battery is inserted in the battery slot.



# Hardware Setup

For this demo, you will need the following hardware:

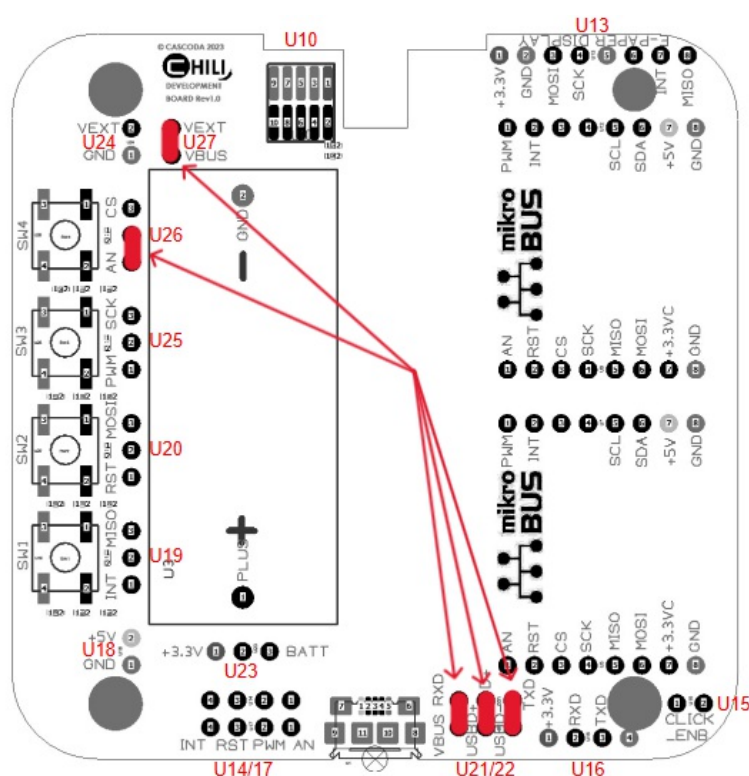
- Development board with jumpers provided
- Thermo 3 click
- USB-A to Micro-USB cable
- (Optional) A CR123A 3V Non-Rechargeable Lithium Battery

**NOTE!** Before and during the steps required to flash the firmware, the device has to be powered via USB. After flashing the firmware, the USB is no longer required, so the device can then be powered with a battery. Make sure that you are using the correct jumper configuration when using the USB, and then make sure to change the jumper configuration if switching to battery power!

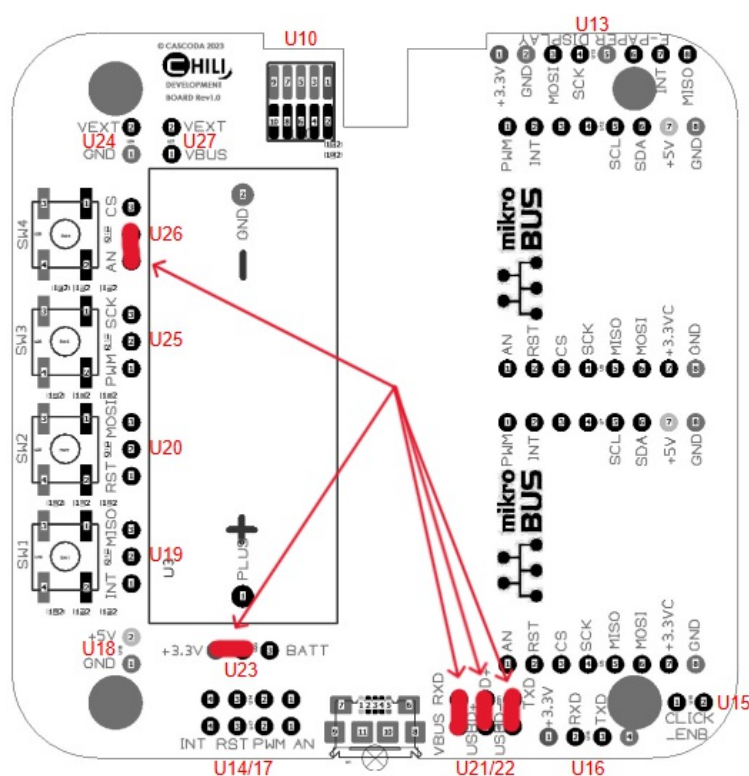
To set up the hardware, follow the following steps:

1. When using the KNX IoT Development board, the jumpers need to be placed in the correct position, and this differs from application to application, as well as depending on which means of power is provided. For this demo, please make sure that your jumper configuration exactly matches what is shown in the pictures below (see the arrows pointing to the red blocks, which represent the jumpers). Also, make sure that none of the jumpers are loose.

- a. If powering via USB, use this jumper configuration:



b. If powering with a CR123A 3V Non-Rechargeable Lithium Battery, use this jumper configuration:



- Place the Thermo 3 click, as shown in the pictures at the front page of this document. The end with the little carved out notch should face the battery slot. You can insert the Click board on the right or left slot, both are the same.
- If powering via USB, connect the USB cable into the USB port on the development board. If powering with a battery, insert the battery into the battery slot with the correct orientation (see + symbol on the battery slot case).

**WARNING!** Power off the device, by either removing the battery or unplugging the USB cable, before moving the jumpers around, e.g. to transition from using USB power to battery power, or vice versa.

# Flashing the Firmware

**WARNING!** DO NOT ENGAGE WITH THIS SECTION IF YOU HAVE NOT FIRST FOLLOWED THE [HARDWARE SETUP](#) SECTION. DOING SO COULD RESULT IN DAMAGE TO YOUR DEVICE.

Make sure you have the following before getting started:

- An installation of Cascoda's Windows Tools. Please [download CascodaWindowsTools.zip and run the installer within](#). Two of the tools will be necessary for this guide, namely `chilictl.exe` and `serial-adapter.exe`. NOTE: By default, these tools are added to your PATH, enabling their execution in a shell in any directory. However, if this did not occur, you will only be able to execute the tools from within the directory in which they are installed. The default installation directory is `C:\Program Files(x86)\Cascoda Windows Tools`.
- (Only needed in rare circumstances) The bootloader binary `ldrom_hid.bin`
- The application binary `knx_rts_reed.bin` for the RTS demo
- The serial number binary `029B00000261.bin` for the RTS demo

Once you have all of the above, follow the steps below:

1. Connect the development board to a Windows PC via USB.
2. Determine the development board's unique USB serial number (note: this is unrelated to the KNX serial number, which we talk about in the rest of this document), by using the Cascoda Windows Tool `chilictl.exe`, using the command shown below (**`chilictl.exe list`**). Note that this will list all connected devices, so if you have multiple devices and want to identify which one is the devboard, then run the command once with the devboard disconnected, then again with it connected, so that you can identify the new one that appears:

```
$ chilictl.exe list
2024-11-11 10:37:19.574 NOTE: Host Cascoda SDK v0.23-27-g0392f99 Jun 17 2024
Device Found:
  Device: Chili2
  App: mac-dongle
  Version: v0.24-793-gbbace5f62-dirty
  Serial No: 4EF5FF03014D29B0
  Path: \\?\hid#vid_0416&pid_5020#a&82a8bb7&0&0000#{4d1e55b2-f16f-11cf-88cb-001111000030}
  Available: Yes
  External Flash Chip Available: Yes
Device Found:
  Device: Chili2
  App: knx_rts_reed
  Version: v0.23-1181-ga6649ff63
  Serial No: 2611E85EB0B44540
  Path: \\?\hid#vid_0416&pid_5020#a&a0e79f7&2&0000#{4d1e55b2-f16f-11cf-88cb-001111000030}
  Available: Yes
  External Flash Chip Available: Yes
```

3. Flash the application binary `knx_rts_reed.bin` using `chilictl.exe`. Use the command shown in the picture below (**`chilictl.exe flash -s <USB-serial-number> -cf knx_rts_reed.bin`**). The number that comes after `-s` should be the serial number that you have identified in the previous step. The result should look like the image below. If instead you get an error, please

jump to the [Troubleshooting section](#).

```
$ chilictl.exe flash -s 2611E85EB0B44540 -cf knx_rts_reed.bin
Last write time of knx_rts_reed.bin is: 11/08/2024 13:36
2024-11-11 10:38:53.260 NOTE: Host Cascoda SDK v0.23-27-g0392f99 Jun 17 2024
1 devices found.
Flasher [2611E85EB0B44540]: INIT → REBOOT
Flasher [2611E85EB0B44540]: REBOOT → ERASE
Flasher [2611E85EB0B44540]: ERASE → FLASH
Flasher [2611E85EB0B44540]: FLASH → VERIFY
Flasher [2611E85EB0B44540]: VERIFY → VALIDATE
Flasher [2611E85EB0B44540]: VALIDATE → COMPLETE
```

4. Double check that the application binary is now flashed onto the device, by running the same command as step 2 (**chilictl.exe list**), and making sure that the "App" name is "knx\_rts\_reed".
5. Flash the KNX serial number binary 029B00000261.bin using chilictl.exe, with the command shown below (**chilictl.exe flash -s <USB-serial-number> -m 029B00000261.bin**).

**NOTE!** This serial number binary should only be flashed on 1 device per network partition. If two or more devices on the same Thread network have been flashed with this serial number binary, the demo won't work. If you need to have multiple RTS devices on the same network, please contact us so that we can provide you with a different unique serial number.

```
$ chilictl.exe flash -s 2611E85EB0B44540 -m 029B00000261.bin
2024-11-11 10:40:34.759 NOTE: Host Cascoda SDK v0.23-27-g0392f99 Jun 17 2024
1 devices found.
Flasher [2611E85EB0B44540]: INIT → REBOOT
Flasher [2611E85EB0B44540]: REBOOT → ERASE
Flasher [2611E85EB0B44540]: ERASE → FLASH
Flasher [2611E85EB0B44540]: FLASH → VERIFY
Flasher [2611E85EB0B44540]: VERIFY → VALIDATE
Flasher [2611E85EB0B44540]: VALIDATE → COMPLETE
```

Your device is now ready to start the commissioning process!

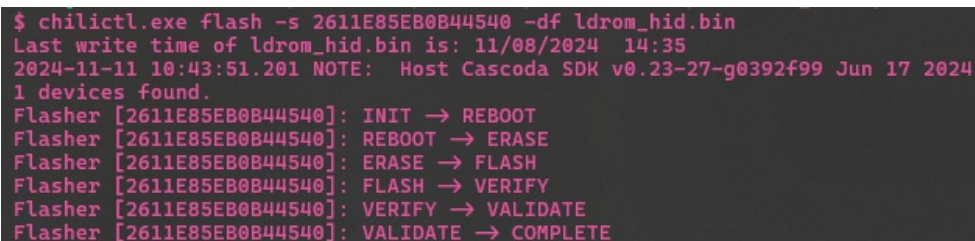
## Troubleshooting

If you get an error that looks like the image below when trying to flash the application binary, then this probably indicates that your development board does not have a bootloader on it. To fix this, try the steps below:

```
$ chilictl.exe flash -s 2611E85EB0B44540 -cf knx_rts_reed.bin
Last write time of knx_rts_reed.bin is: 11/08/2024 13:36
2024-11-11 10:43:16.117 NOTE: Host Cascoda SDK v0.23-27-g0392f99 Jun 17 2024
1 devices found.
Flasher [2611E85EB0B44540]: INIT → REBOOT
Flasher [2611E85EB0B44540]: REBOOT → FAIL
Error: Failed to discover device [2611E85EB0B44540] after reboot to DFU
Error: Early termination - NOT_FOUND
Error: FAIL
```

1. Disconnect the development board from the PC, and connect it again. THIS STEP IS NECESSARY.
2. Repeat step 2 of [the previous subsection](#), and make sure that your device is listed. If it is not listed, make sure that you have disconnected/reconnected the device as per the previous step.

- Flash the bootloader binary `ldrom_hid.bin` using `chilictl.exe`. Use the command shown in the picture below (**`chilictl.exe flash -s <USB-serial-number> -df ldrom_hid.bin`**).



```
$ chilictl.exe flash -s 2611E85EB0B44540 -df ldrom_hid.bin
Last write time of ldrom_hid.bin is: 11/08/2024 14:35
2024-11-11 10:43:51.201 NOTE: Host Cascoda SDK v0.23-27-g0392f99 Jun 17 2024
1 devices found.
Flasher [2611E85EB0B44540]: INIT → REBOOT
Flasher [2611E85EB0B44540]: REBOOT → ERASE
Flasher [2611E85EB0B44540]: ERASE → FLASH
Flasher [2611E85EB0B44540]: FLASH → VERIFY
Flasher [2611E85EB0B44540]: VERIFY → VALIDATE
Flasher [2611E85EB0B44540]: VALIDATE → COMPLETE
```

- If this was a success, then you can go back to step 3 of [the previous subsection](#), and that step should now succeed. If you are still seeing some error in flashing the application, please double check that you are entering all of the commands exactly in the correct order and exactly as shown (except for the USB serial number that comes after `-s`, which will depend on your particular device).

# Commissioning

After powering the device using the USB cable, the device will enter a commissioning phase. Thread commissioning needs to be done first, since this will enable IPv6 communication. Next, KNX commissioning needs to be done using ETS (version ETS 6.3.0 or later).

For both Thread and KNX commissioning, the QR code below will be needed.



In case you are unable to scan the QR code, you can instead copy/paste the following:  
41S029B00000261+3ZEUI:1AEE09D5FDFC8668.P:VE3YPB1W60.PA:M95993H1HX

Check out the youtube video [here](#), demonstrating the process of doing Thread and KNX Commissioning using a QR code scanner.

## Thread Commissioning

Thread commissioning is adding the device to the thread network. To be able to do so, one needs to have a Thread Border router. Cascoda recommends using the [KNX-IOT-HUB](#).

NOTE: The steps below assume that you already have a Thread network set up using the KNX IoT Hub. If you don't, then click [here](#) for instructions on how to do that (follow the part regarding using the Web UI).

1. Hover over "Thread" in the menu bar at the top, and click on "Add Device".
2. There is an input field to put QR code information. Using a QR code scanner, scan the QR code provided in this manual and then click "Submit".

CASCODA®  
Innovation in IoT

OpenWrt

Status ▾

System ▾

Services ▾

Network ▾

Thread ▾


Logout

No password set!  
There is no password set on this router. Please configure a root password to protect the web interface.

Add Joiner in Network: OpenThread

QR Code Entry

Manual Entry



Request Camera Permissions

[Scan an Image File](#)

41S029B00000261+3ZEUI:1AEE0

Back to View

Submit

3. Now wait for the device to join the Thread network. When it does, you will see it show up in the "Neighbors" table.

CASCODA®  
Innovation in IoT

OpenWrt

Status ▾

System ▾

Services ▾

Network ▾

Thread ▾

Logout

REFRE

No password set!  
There is no password set on this router. Please configure a root password to protect the web interface.

Thread View: OpenThread (wpan0)  
This is the list and topograph of your thread network.

List

Topology Graph

IPv6 Addresses

Leader Situation of Network

Leader Router Id	Partition Id	Weighting	Data Version	Stable Data Version
37	618116596	64	27	20

Neighbors

RLOC16	Role	Age	Avg RSSI	Last RSSI	Mode	Extended MAC	
0x9402	Child	6	-31	-32	rdn	066bd20a17a739bd	
Pending	New Joiner	Pending	Pending	Pending	Pending	b7092300cef82669	Remove

NOTE: The above steps are enough to move forward with the demo. See [here](#) for more details about Thread commissioning.

## KNX Commissioning

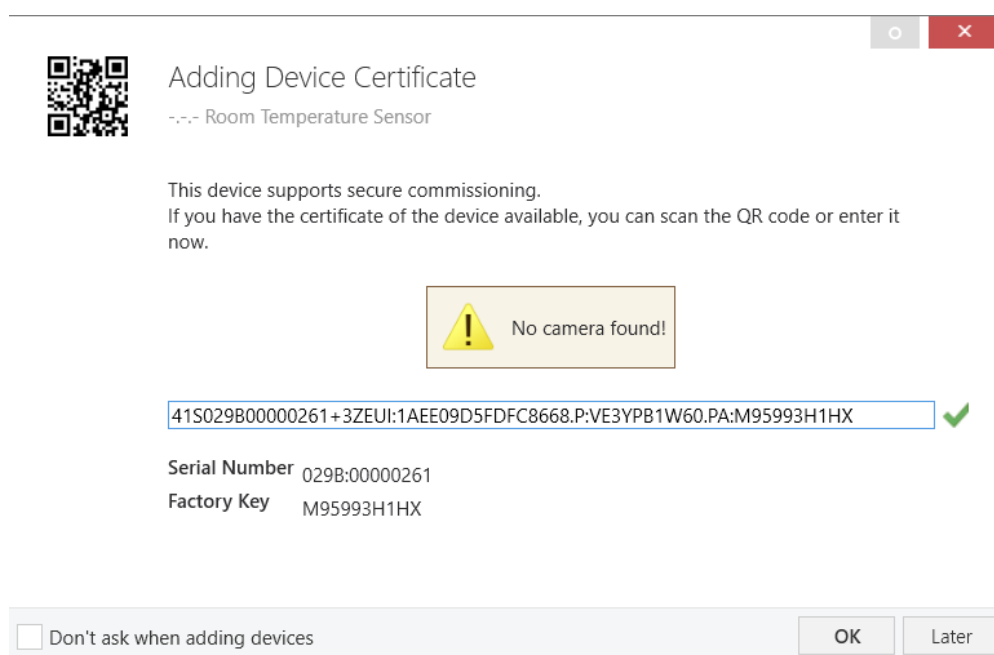
KNX commissioning is adding the device to an ETS project. Since KNX IoT is a secure KNX protocol, one needs to have the security credentials and the serial number of the device. This information is contained in the QR code.



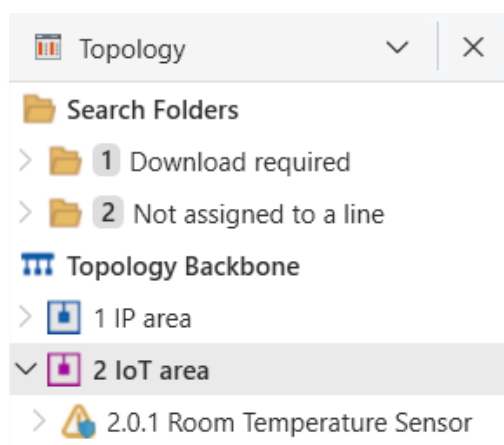
The device can only be added to a KNX IoT Area or Line. When the device is added to a KNX IoT area or Line, the credentials can be supplied. ETS can scan the QR code with the camera (or 2D bar code scanner).

## Adding the Device to a Topology

1. Open the project provided in ETS version 6.3.0 or later.
2. Open the "Devices" panel, and select the device that you will use for this demo. In the "Properties" section on the right, click "Add Device Certificate". A window will pop up, with an input field for the QR code information. Use a QR code scanner to scan the QR code provided in this manual. (There is also an option to use a camera).



3. Open a "Topology" panel. Drag and drop the device into an IoT area.



## Creating a Configuration

These steps create the group objects that will be used in the s-mode messages. By linking the group objects to the communication objects (data points), one can send s-mode messages to actuators, and receive s-mode messages from sensors.



1. Open a "Group Addresses" panel.
2. Select "Group Addresses", right click, select "Add Main Groups", click OK.
3. Select the newly created main group, right click, select "Add Middle Groups", click OK.
4. Select the newly created middle group, right click, select "Add Group Addresses", increase the count to however many group addresses you want (e.g. the same as the number of Group Objects that you would like to link), then press "OK".
5. Now open the "Group Objects" tab for your device, and link each group object that you want by right clicking, "Link with...", and then selecting the group address.



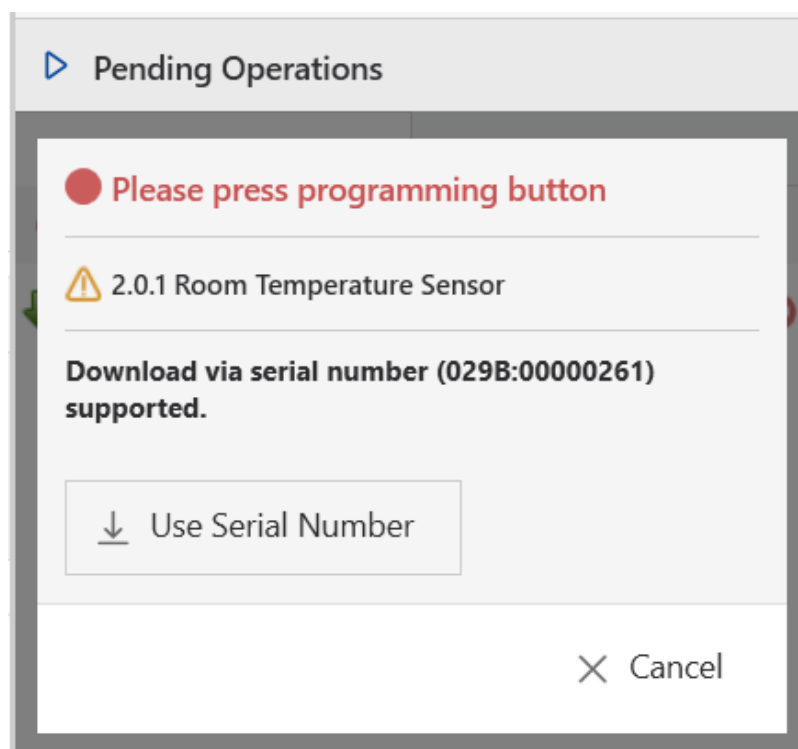
## Downloading the ETS configuration

The downloading of the configuration can happen when the ETS data for the device is created, i.e.:

- The parameters are set
- The communication objects are connected

This has already been done with the previous two steps. Therefore, you can now download the configuration by selecting the device in the "Devices" panel, right click, hover over the "Download" option, and click "Download All". After waiting a few seconds, there will be a popup asking you to either:

- download by serial number
- download with programming mode



The download by serial number does not require any interaction with the device. All you have to do is click on the "Use Serial Number" button.

The download with programming button requires pressing the programming button. The **PROG** button is the SW4 button, which needs to be held down for 1 second. While the **PROG** button is pressed, the LED is on. When the **PROG** button is released and the device is in programming mode the LED will start blinking. Disabling the programming mode can be achieved by holding down the **PROG** button again for 1 second.

NOTE: The device has to be attached to a Thread network in order for it to enter programming mode. In case you are unable to put the device into programming mode, double check that it is indeed attached to a Thread network. And also make sure you are not releasing the button too soon (hold it down for > 1 second).

Once the download is complete, the device becomes fully operational and functional.

## Resetting the device

**NOTE!** This section is informational. If you are simply following the steps to run the Demo with ETS, please do not reset the device!

The device allows resetting of KNX and Thread in separate steps. This allows that the KNX configuration can be reset to factory default, without resetting the connectivity part.

- Reset KNX

Reset of KNX is achieved by pressing the **PROG** button for 5 seconds. While the **PROG** button is pressed, the LED is on.

When the **PROG** button is released (after 5 sec), the LED will quickly flash 2 times.

**NOTE!** KNX Reset: this means that also the security credentials are removed.  
Hence ETS will download newly created device keys.

- Reset Thread

Reset of Thread is achieved by pressing the **PROG** button for 10 seconds. While the **PROG** button is pressed, the LED is on. When the **PROG** button is released (after 10 sec), the LED will slowly flash 3 times.

**NOTE!** Thread Reset: This means that the device needs to be added to the thread network again.

# Running the Demo with ETS

**NOTE!** If you follow this section and the demo still does not work, please refer back to the [hardware setup](#) section, and double check that your device is set up exactly as described. If the jumper configuration does not match, or if the click board is inserted incorrectly, this can result in the demo not working.

You can run a demo using only 1 RTS device and ETS. The Group Monitor can be used as an actuator, receiving and displaying the messages coming from the RTS, in a similar way that a KNX IoT Counterpart product would receive and display or act on the temperature messages.

NOTE: This section assumes that you have followed the Thread Commissioning and KNX Commissioning steps, and thus have already downloaded a configuration to your device using ETS.

With that being said, follow the steps below to observe the messages being sent by the RTS:

1. Open the Group Monitor (Panels > Diagnostics > Group Monitor), and click "Start".
2. Wait, and observe the temperature messages get displayed on the group monitor. The RTS demo application senses and sends the temperature once every 20 seconds.

The screenshot displays the ETS 2.0.1 Room Temperature Sensor interface. The top bar includes a title bar with a warning icon and the text "2.0.1 Room Temperature Sensor", and a toolbar with icons for Add, Delete, Download, and Undo. Below this is a "Group Objects" panel with a search bar and a table listing objects. The table has columns: Se, Number, Name, Object Function, Linked with, Other Linked, Length, C, R, W, T, U, Data Type, and Priority. One object is listed: "1 RTS" with "OUT url:/p/o\_1\_1 dpa.321..." as the Object Function, "0/0/1 New grou..." as Linked with, "2 bytes" as Length, and "temperatu...Low" as Data Type. Below the Group Objects panel is a "Parameters" panel with a toolbar containing Start, Stop, Clear, Open, Save, Print, Replay Telegrams, Stop Replay, and Options. The main area of the Parameters panel shows "Group Address" (empty), "Data point type" (1.\* 1-bit), "Delay time[sec]" (0), and "Last received value" (000). There are "Write" and "Read" buttons. At the bottom is a table with columns: Destination Name, Building Function, Building Part, Hop Type, DPT, and Info. The table contains four rows of data, all with "New group address" as the Destination Name and "ValueWrite" as the Hop Type. The DPT values are "9.001 temperature (°C)" and the Info values are "25.375 | 25.38 °C", "25.4375 | 25.44 °C", "25.375 | 25.38 °C", and "25.25 | 25.24 °C".

Se	Number	Name	Object Function	Linked with	Other Linked	Length	C	R	W	T	U	Data Type	Priority
	1	RTS	OUT url:/p/o_1_1 dpa.321...	0/0/1 New grou...		2 bytes	C	R	-	T	-	temperatu...Low	

Destination Name	Building Function	Building Part	Hop Type	DPT	Info
New group address			ValueWrite	9.001 temperature (°C)	25.375   25.38 °C
New group address			ValueWrite	9.001 temperature (°C)	25.4375   25.44 °C
New group address			ValueWrite	9.001 temperature (°C)	25.375   25.38 °C
New group address			ValueWrite	9.001 temperature (°C)	25.25   25.24 °C

# Software Bill of Materials

This paragraph contains the list of used open source software in this product.

Name	Version	License
Cascoda SDK	0.25	BSD-3-Clause
tinycbor	v0.6.0	MIT
mbedtls	2.16.2	Apache-2.0
Openthread	knx-v1.0.0	BSD-3-Clause

## Cascoda SDK

- Description: Cascoda development
- License: BSD-3-Clause
- Version: 0.25
- URL: <https://github.com/Cascoda/cascoda-sdk>
- Notes: Chili2D/S SDK, various drivers

## tinycbor

- Description: CBOR implementation
- License: MIT
- Version: v0.6.0
- URL: <https://github.com/intel/tinycbor>
- Notes: used for CBOR encoding/decoding

## mbedtls

- Description: security constructs
- License: Apache-2.0
- Version: 2.16.2
- URL: <https://github.com/ARMmbed/mbedtls>
- Notes: used for encryption/decryption

## Openthread

- Description: OpenThread, IPv6
- License: BSD-3-Clause

- Version: knx-v1.0.0
- URL: <https://github.com/Cascoda/openthread>
- Notes: Cascoda's port of OpenThread

# KNX device information

Info Field	Value
Manufacturer	cascoda
Model	thermo 3 click
Order_number	
Hardware_type	0000000000000
Hardware version	[0, 1, 2]
Firmware version	[1, 1, 0]
Sleepy Device	No

## Data points

url	name	instance	resource type	interface type	data type
"/p/o_1_1"	RTS	1	321.51	if.s	DPT_Value_temp

## Parameters

No parameters defined.