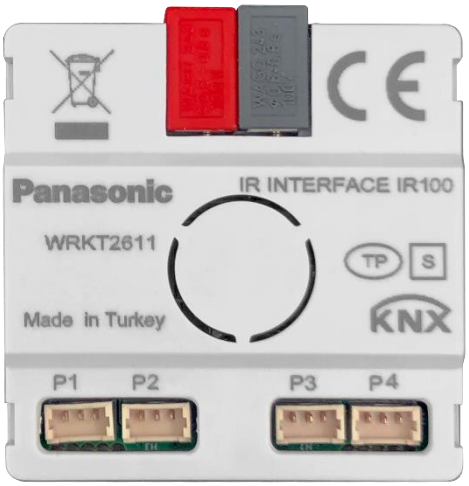
KNX IR Interface IR100

Reference Manual



KNX IR Interface IR100	WRKT26115NC

V 1.0

Panasonic

Contents

1 List of Abbreviations

Abbreviation	Description
AC	Air conditioning
AF	Auxiliary function
Ch	Channel
DCA	Device configuration app
DPT	Data point type
ETS	Engineering tool software
IR	Infrared
Mac	Macro
KNX Communication Flags	
C	Communication
R	Read
W	Write
T	Transmit
U	Update

2 Product description

2.1 General information

The KNX IR Interface is an intelligent transmitter and receiver for infrared (IR) signals. This allows IR signals from various manufacturers to be read, stored and sent. The fields of application are from private residential buildings to the office buildings. KNX IR Interface allows to manage any device that has an IR receiver, from televisions and air conditioners, to DVD players and amplifiers, control over all appliances that use IR communication is possible. Also, KNX IR Interface allows to control KNX devices such as lights, blinds and thermostats with any IR remote control.
The device is supplied with power via the KNX bus voltage and no additional power supply is required.

2.2 Main features

- Small size 37 x 37 x 11.7 mm; can fit in standard electrical box.
- 3 transmitter ports and 1 port can be used as transmitter port or receiver port.
- 100 channels can be used to transfer commands from KNX to IR or from IR to KNX.
- KNX to IR channels can send an IR code repeatedly with configurable time delays.
- IR to KNX channels can be used to control lights, blinds and thermostats, and call scenes.
- 16 macros with 10 parts each can be used to send multiple IR codes sequentially.
- 4 AC modules can be used to control 4 AC split units with KNX group objects.
- AC modules have multiple group objects for control and status with KNX standard datapoint types.
- Each AC module has 8 scenes.
- 8 auxiliary functions can be used as logic gate, sequencer / counter, converter, scene actuator or send after reset function.
- All channels, macros, AC modules and auxiliary functions have lock input object.
- Teaching IR codes and loading AC configuration files operations are done with a Device Configuration App (DCA) in ETS.
- Up to 250 IR codes can be taught to the device.
- Taught IR codes can be exported to and imported from a file.

2.3 Technical information

Supply voltage	21 … 32 V DC via KNX Bus
Operating temperature	-5°C … +45°C
Type of protection	IP 20 to EN60529
Safety class	III to IEC 60664-1
IR transmitter maximum transmitting distance (*)	5.5 m
IR transmitter transmitting angle(*)	40°

* If the device will be used to control AC units the IR transmitter should be mounted directly to the IR receiver of the AC unit to get the best performance.

2.4 Dimensional drawings

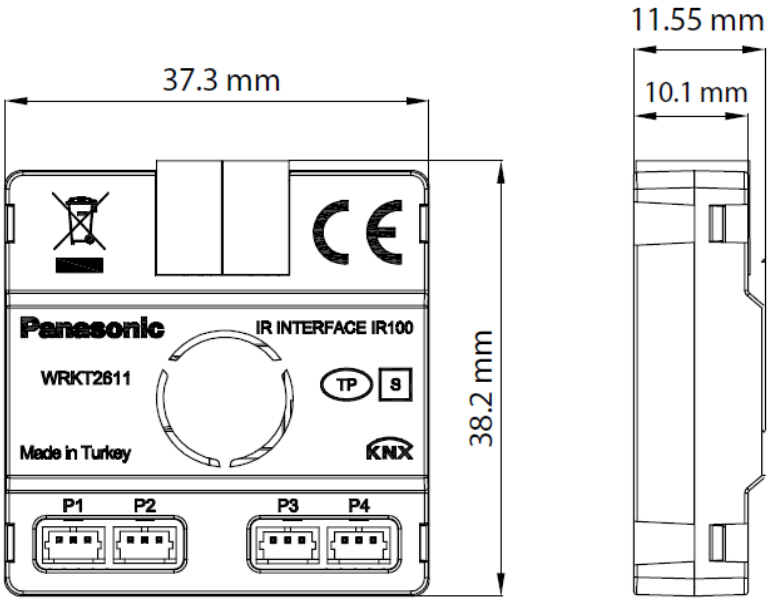


Figure 1 Top and side view of the KNX IR Interfere

2.5 Installation

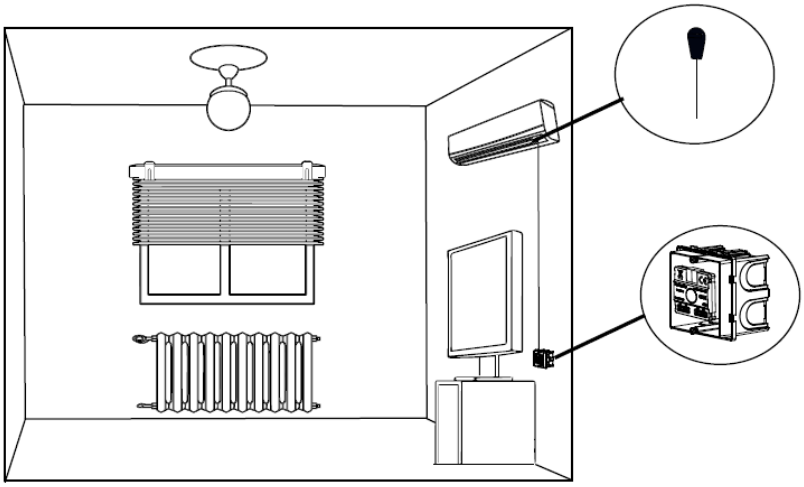
2.5.1 Device Installation

The device is installed in a flush-mounted or wall-mounted box.

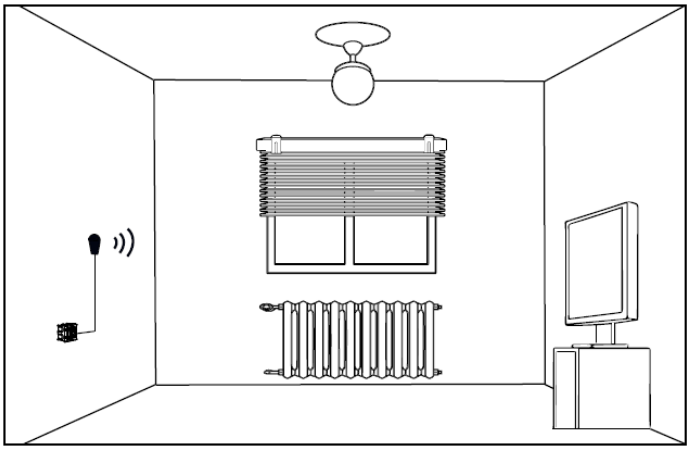
2.5.2 Transmitter Installation

The IR transmitter of the KNX IR Interface device can be mounted in two ways.

- Directly to the IR receiver of the controlled device.

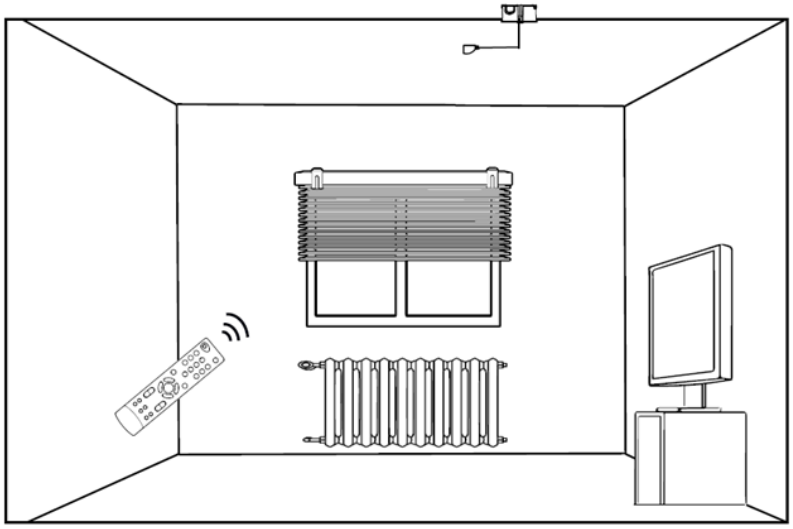


- Near the controlled device or on the opposite wall.



2.5.3 Receiver Installation

The receiver of the KNX IR Interface device can be installed on the walls, ceilings or desktops.
The KNX IR receiver should be close to the user and at a point where it can receive IR codes accurately.
The optimal receiving distance is 10 m when the IR receiver and the IR remote control are arranged in a straight line. The angle of the half receiving distance is ± 45°.
Please note that the range depends on the IR remote control and its battery charge level.



2.5.4 Installation Notes

- Make sure nothing blocks IR signals in front of the IR receiver and the IR transmitter. Even a sheet of paper can block IR signals effectively.
- IR receiver and transmitter cables should be installed far away from power lines.
- Sunlight, heat sources and artificial lighting are infrared sources and they can interface with IR signals. Avoid mounting the IR transmitters and receivers under sunlight, near artificial lights and in very bright illuminance environments.

3 Product ETS database

Manufacturer	Panasonic
Product family	Interfaces
Product type	IR Interface
Product name	IR Interface IR100

4 Configuration

The device is unloaded in factory default. User can define the device behavior and connect to another KNX devices in ETS.

4.1 Configuration Prerequisites

- The latest version of ETS should be downloaded and installed.
- The latest version of IR Interface ETS database should be imported to ETS Catalogs.
- The latest version of IR Interface DCA should be downloaded and installed in ETS. (Available free of charge at KNX online shop).

4.2 Configuration Steps

1- Connect the interface to the KNX bus.
2- Add the KNX IR Interface device to your ETS project from ETS Catalogs.
3- Give the interface an individual address then download it to the device.
4- Access the DCA tab to write IR codes and AC remote configurations to the device.
5- Access the Parameters tab to configure the parameters of the interface.
6- Link the communication group objects of the KNX IR Interface with group address from your project.
7- Download the KNX IR interface application.

Notes:

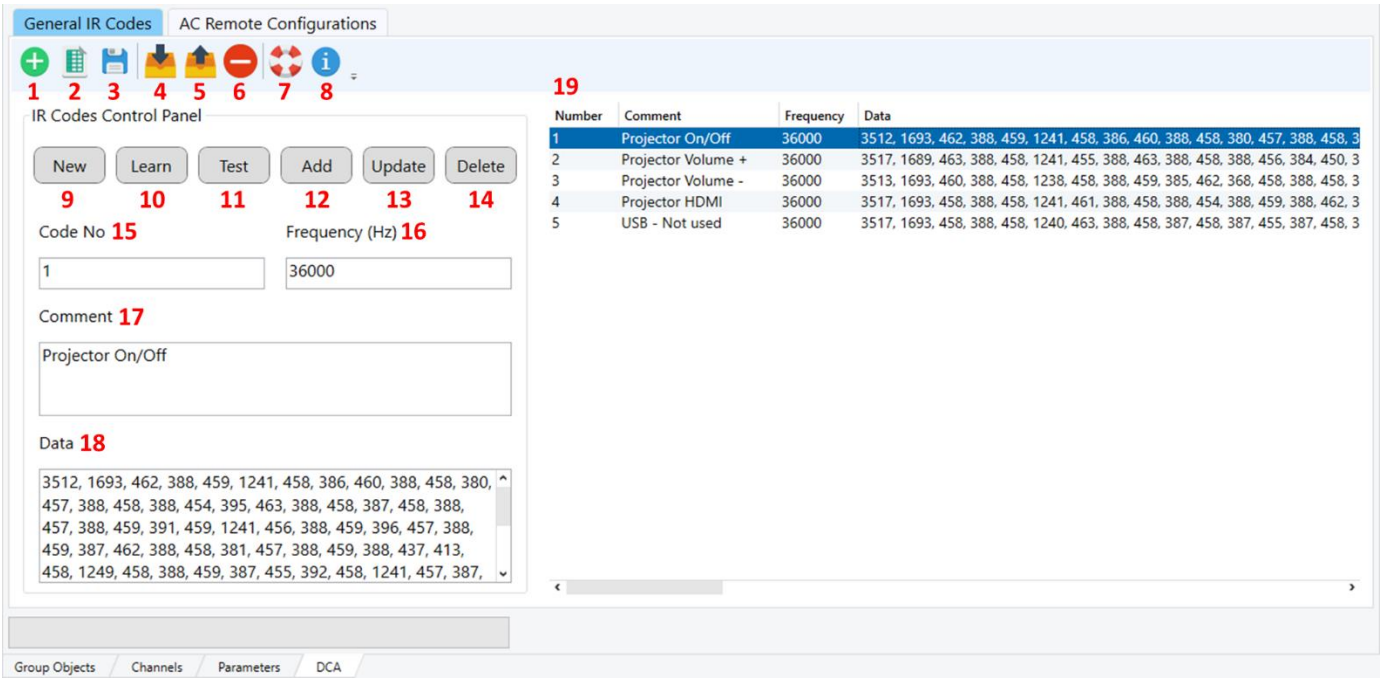
- The user must download an individual address to the device before configuring it in DCA to be able to communicate with the device.
- Coping the device database in ETS doesn't copy the configurations that have been done in the DCA. **The IR codes and the AC configurations should be written again for the copied devices.**

5 Device Configurations in DCA

IR Interface DCA has two main tabs:

5.1 General IR Codes Configurations:

This tab is used to record the IR codes of the remote controls which the user uses, test them and write them to the device. Its interface as below.



1- New IR codes config file button: This button is used to create a new IR codes configuration file.
2- Open IR codes config file button: This button is used to open a previously saved IR codes config file.
3- Save IR codes config file button: This button is used to save the current learned IR codes collection in an IR codes config file to load it on another device or to reconfigure it in the future.
4- Write IR codes to the device button: Pressing this button starts writing the current IR codes collection to the device process.
5- Read IR codes from the device button: Pressing this button starts reading previously loaded IR codes from the device process. The read codes are loaded without its comments because code comments are not written to the device in the writing process.
6- Delete IR codes from the device button: Pressing this button starts deleting IR codes from the device process.
7- Help button: This button shows the help document of the DCA.
8- Info button: This button shows some information about the DCA.
9- New IR code button: This button is used to create a new IR code record. The number of the new code is the highest number of the current IR codes + 1.
10- Learn IR code button: This button is used to record the IR signal of a button from the remote control. The user should press the remote control button which he wants to record in 5 seconds after pressing this button. After receiving an IR signal from the device, its data is loaded to the Data field automatically.

11- Test IR code button: This button is used to check if the received/saved IR code data is correct. This button sends the IR code data from “Data” field to the device in order to transmit them to the first 3 transmitter ports.
12- Add IR code button: This button adds the IR code that is in the control panel to the learned IR codes collection.
13- Update IR code button: The user can select an IR code from the learned IR codes collection (19) to preview its info, edit it, test its code data or re-teach a code signal. The user should press this button after modifying an IR code info to save its modifications.
14- Delete IR code button: This button is used to delete a previously added IR code from the learned IR codes collection after selecting it.
15- IR code number filed: This number links the IR codes with the channels and macros in Parameters section. Although the maximum downloadable code number is 250, IR codes with code number higher than 250 can be added and exported to a file.
16- IR code frequency field: the default frequency is 38000 Hz. IR codes frequencies cannot be read by the receiver. The user should modify the default frequency only if he knows the frequency of the remote control.
17- IR code comment field: This field is used to give IR codes a meaningful name. The comments of the IR codes are not downloaded to the device but they are saved in the exported IR codes config files.
18- IR code data field: This field is filled with IR code signal data automatically in the learning process. The user can copy and paste the data but he can’t edit them directly.
19- The learned IR codes collection: This table shows the list of the learned IR codes, selecting a code from this list shows its info and data in the control panel.

5.1.1 Teaching IR Codes Steps

1- Connect the IR receiver to port 4 and the IR transmitter/s to the other ports.
2- Connect the device to KNX bus and make sure that it has an individual address.
3- In the DCA tab, press button (1) to create a new IR config file or button (2) to open a previously saved one and append IR codes to it.
4- Press button (9) to create a new IR code, the code number is given automatically.
5- Press button (10) to start the learning process. When you see “Please press a button” popup, press the button from the remote control which you want to teach in 5 seconds. If the device receives an IR code, it will close the popup and load the code data to “Data” field. In this process the head of the remote control should be close to the IR receiver (about 5 cm).
6- Position an IR transmitter to the controlled device (TV, HiFi, etc.) and put it so close. Then press button (11) to test the received IR code data. If the controlled device reacted as it received the original code from the remote control then the code it taught correctly, else repeat step (5).
7- Add a comment describes the function of the taught IR code.
8- Press button (12) to add the new IR code to the learned IR codes collection.
9- Repeat steps (5, 6, 7, and 8) to teach the codes of the remote control button that you want.
10- Press button (4) to write the taught IR codes to the device and button (3) to save them to a file.

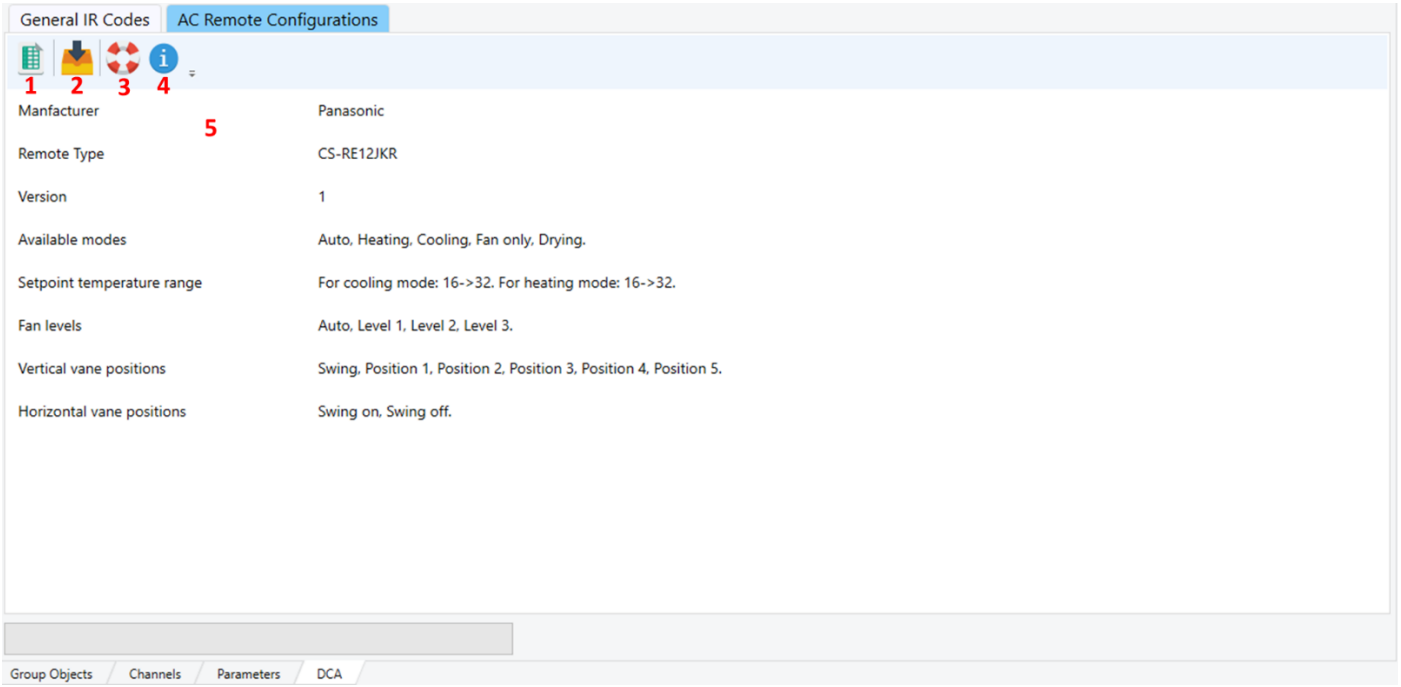
5.1.2 Modifying IR Codes’ Info and Data Steps

1- Select the IR code that you want to modify from the learned IR codes collection (13).
2- Modify the fields that you want. You can re-teach a new IR code by pressing button (10).

3- Press Update button (14) to apply the changes. Press New button (9) or select another IR code from the collection to ignore the changes.

5.2 AC Remote Configurations

This tab is used to load AC remote configurations from a file and write it to the device. Its interface as below.



1- Open AC remote config file button: This button is used to open an AC configuration file.
2- Write AC remote configurations to the device: Pressing this button starts writing the loaded AC remote configurations to the device.
3- Help button: This button shows the help document of the DCA.
4- Info button: This button shows some information about the DCA.

6 General Settings

6.1 General Parameters

Name	Values	Description
Startup delay	0…**3**…250 s	This parameter defines the delay time for startup in seconds. After bus voltage recovery, the device always waits the delay time to expire before sending telegrams to the bus.
Port 4 is used as	IR transmitter **IR receiver**	Port 4 on the device can be used as an IR receiver or an IR transmitter. This parameter define its functionality.
Used channel number	0…**1**...100	This parameter defines how many channel will be used. Channel parameter pages are generated up to this parameter value.
Used macro number	0...16	This parameter defines how many macro will be used. Macro parameter pages are generated up to this parameter value.
Used AC module number	0...4	This parameter defines how many AC module will be used. AC module parameter pages are generated up to this parameter value.
Send device in operation telegram	**No** Yes	Selecting Yes enables “Device In Operation” telegram.
Device in operation telegram value	**Off** On	This parameter is visible if “Send device in operation telegram” parameter is set to “Yes”. It specifies the value of the telegram that will be sent from “Device In Operation” object.
Device in operation telegram cycle time	00:00:01…**00:01:00**…09:06:07	This parameter sets the time interval at which the “Device In Operation” group object sends a telegram cyclically.

6.2 General Group Objects

No	Object Name	Function	Size	Datapoint Type	Flags				
					C	R	W	T	U
1	*Device In Operation*	*Trigger*	1 Bit	1.001 Switch	C	R		T	

This object is available if “Send device in operation telegram” parameter is set to “Yes”. It is used to monitor the presence of the device on KNX bus. After startup delay time the device sends telegrams cyclically to this object according to “Device in operation telegram value” and “Device in operation telegram cycle time” parameters values.

No	Object Name	Function	Size	Datapoint Type	Flags				
					C	R	W	T	U
2, 3, 4, 5	*IR Port x*	*Transmit Code Number*	1 Byte	5.010 Counter pulses	C		W		

Object 5 is available if “Port 4 is used as” parameter is set to “IR transmitter”. These object can be used to send an IR code from a loaded IR codes collection to transmitter ports independently of KNX->IR channels and macros.

No	Object Name	Function	Size	Datapoint Type	Flags				
					C	R	W	T	U
6	*IR Port 4*	*Received Code Number*	1 Byte	5.010 Counter pulses	C	R		T	

This object is available if “Port 4 is used as” parameter is set to “IR receiver”. It sends the code number of a received IR code from the loaded IR codes collection.

7 Channels

There are two types of channels:

1- KNX->IR channels: send a loaded IR code when it receives a specific telegram from its input object. These channels are used to control IR devices like TVs and DVD players with KNX devices like switches, touch panels and smartphones.

2- IR->KNX channels: send a specific telegram to its output object when the device receives a loaded IR code. These channels are used to control KNX devices like switching-blind actuators, dimming actuators and thermostats with IR remote controls.

It's possible to control IR devices with its not original remote controls by linking the output of an IR->KNX channel to the input of a KNX->IR channel.

Each channel has a lock function that can be used to block a channel operation.

7.1 KNX->IR Channels

7.1.1 KNX->IR Channel Parameters

Name	Values	Description
Channel name		The user can give a channel an optional name that describes it functionality. This parameter value has no effect on the channel work.
Channel type	Not used **KNX->IR** IR->KNX	This parameter selects the type of the channel. If "Not used" is selected, the channel is disabled.
Input object type	**1-bit** Scene number 1-byte	This parameter specifies the DPT of the input object.
Send code at *Send code at scene number* *Send code at value*	**Any telegram value** On telegrams only Off telegrams only **1**…64 **0**…255	This parameter specifies the telegram value that triggers the channel to send its IR code.
Code number	0..**1**..250	This parameter specifies which IR code will be sent when the channel receives a proper input telegram. The code number must be set according to the set configuration in the DCA.
Send IR code to port 1	No **Yes**	Selecting "Yes" transmits the IR code to port 1 when the channel is triggered.
Send IR code to port 2	**No** Yes	Selecting "Yes" transmits the IR code to port 2 when the channel is triggered.

Send IR code to port 3	**No** Yes	Selecting "Yes" transmits the IR code to port 3 when the channel is triggered.
Send IR code to port 4	**No** Yes	This parameter is available if "*Port 4 is used as" parameter in "General Settings" page is set to "IR transmitter".* Selecting "Yes" transmits the IR code to port 4 when the channel is triggered.
Send IR code after delay	**0**..300 x100ms	This value is the period (in milliseconds) that the device will wait when the channel is triggered before it starts transmitting the IR code. "0" value means the device will send the code immediately.
IR code transmission count	**1**…65535	This parameter determines how many times the code will be sent when the channel is triggered.
Delay before repeating IR code	1…**10**…300	This parameter is available if "IR code transmission count" parameter value is greater than 1. It specifies the period –in milliseconds- the device will wait before it starts resending the IR code. Note: It's recommended to not use low delay time between the repeated codes because some consumer IR devices is not capable of receiving quickly repeated IR codes and may miss some IR codes.
Lock status after bus return	**Unlocked** Locked Read from bus As before bus failure	This parameter determines the status of the channel's lock function after bus voltage return. If "Read from bus" is selected, the device will send a read request for the lock object of the channel, if no response is received the channel will be unlocked.

7.1.2 KNX->IR Channel Group Objects

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
Ch x	*1-bit Input* *Scene Number Input* *1-Byte Input*	1 Bit 1 Byte 1 Byte	1.001 Switch 17.001 Scene number 5.010 Counter pulses	C		W		

This object DPT is set according to "Input object type" parameter value. Sending a telegram that complies with "Send code at" parameter value triggers the channel to start sending its IR code.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
Ch x	*Lock Enable/Disable*	1 Bit	1.003 Enable	C	R	W	T	U

This object is used to enable or disable the lock function of the channel.

7.2 IR->KNX Channels

7.2.1 IR->KNX Channel Parameters

Name	Values	Description
Channel name		The user can give a channel an optional name that describes it functionality. This parameter value has no effect on the channel work.
Channel type	Not used **KNX->IR** IR->KNX	This parameter selects the type of the channel. If “Not used” is selected, the channel is disabled.
Code number	0..**1**..250	This parameter specifies which IR code triggers the channel to send KNX telegram when the device receives it. The code number must be set according to the set configuration in the DCA.
Output object type	**1-bit** Up / Down Toggle Dimming control HVAC mode Scene number Percentage 1-byte 2-byte Float	This parameter specifies the DPT of the output object.
Send KNX telegram after delay	**0**…300	This value is the period (in milliseconds) that the device will wait before it start sending the KNX telegram when the proper IR code is received. “0” value means the device will send the telegram immediately.
Send cyclically	**No** Yes	Selecting “Yes” sends the KNX telegram cyclically when the channel is triggered.
Cycle time	00:00:00…**00:00:10**…09:06:07	This parameter is available if “Send cyclically” parameter is set to “Yes”. It defines the time period between the repeated KNX telegrams
Lock status after bus return	**Unlocked** Locked Read from bus As before bus failure	This parameter determines the status of the channel’s lock function after bus voltage return. If “Read from bus” is selected, the device will send a read request for the lock object of the channel, if no response is received the channel will be unlocked.

7.2.2 IR->KNX Channel Group Objects

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
Ch x	*1-bit Output* *Up / Down* *Toggle* *Dimming Control Output* *HVAC Mode Output* *Scene Number Output* *Percentage Output* *1-Byte Output* *2-Byte Output* *Float Output*	1 Bit 1 Bit 1 Bit 4 Bit 1 Byte 1 Byte 1 Byte 1 Byte 2 Byte 2 Byte	1.001 Switch 1.008 Up/Down 1.001 Switch 3.007 Dimming control 20.102 HVAC mode 17.001 Scene number 5.001 Percentage 5.010 Counter pulses 7.001 Pulses 9.003 Kelvin/Hour	C			T	

This object DPT is set according to “Output object type” parameter value. The channel sends a telegram with a specific value to this object when the device receives an IR code matches the code that is specified in “Code number” parameter.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
Ch x	*Lock Enable/Disable*	1 Bit	1.003 Enable	C	R	W	T	U

This object is used to enable or disable the lock function of the channel.

8 Macros

Macros are used to send sequential and repeated IR codes automatically.

For example, the user can assign a macro to work when cinema scene is called to turn on the TV, switch to the movies channel and turn the volume up. In offices and meeting rooms, a macro can be set to turn on the projector and switch to HDMI input when presentation scene is called.

Each macro has an input object and consists of 10 parts. When the macro is triggered, the IR codes of the macro parts are sent in order.

8.1 Macro Parameters

Name	Values	Description
Macro name		The user can give a macro an optional name that describes it functionality. This parameter value has no effect on the macro work.
Input object type	**1-bit** Scene number 1-byte	This parameter specifies the DPT of the input object.
Send code at *Send code at scene number* *Send code at value*	**Any telegram value** On telegrams only Off telegrams only **1**…64 **0**…255	This parameter specifies the telegram value that triggers the macro to start sending its IR codes.
Send IR code to port 1	No **Yes**	Selecting “Yes” transmits the IR codes to port 1 when the macro is triggered.
Send IR code to port 2	**No** Yes	Selecting “Yes” transmits the IR codes to port 2 when the macro is triggered.
Send IR code to port 3	**No** Yes	Selecting “Yes” transmits the IR codes to port 3 when the macro is triggered.
Send IR code to port 4	**No** Yes	This parameter is available if *“Port 4 is used as” parameter in “General Settings” page is set to “IR transmitter”.* Selecting “Yes” transmits the IR codes to port 4 when the macro is triggered.
Lock status after bus return	**Unlocked** Locked Read from bus As before bus failure	This parameter determines the status of the macro’s lock function after bus voltage return. If “Read from bus” is selected, the device will send a read request for the lock object of the macro, if no response is received the macro will be unlocked.
Part x Code number	0…**1**…250	The macro sends the IR codes of its parts sequentially. This parameter specifies which IR code will be sent when the part turn comes.

		The code number must be set according to the set configuration in the DCA. Macro sends no code if the code number is 0.
Port x Send IR code after delay	**0**..300 x100ms	This value is the period (in milliseconds) that the device will wait before it starts transmitting the IR code of the macro part. "0" value means the device will send the code immediately.
Port x IR code transmission count	**1**…65535	This parameter determines how many times the IR code of the macro part will be sent.
Port x Delay before repeating IR code	1…**10**…300	This parameter is available if "Port x IR code transmission count" parameter value is greater than 1. It specifies the period –in milliseconds- the device will wait before it starts resending the IR code of the macro part. Note: It's recommended to not use low delay time between the repeated codes because some consumer IR devices is not capable of receiving quickly repeated IR codes and may miss some IR codes.

8.2 Macro Group Objects

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
Mac x	*1-bit Input* *Scene Number Input* *1-Byte Input*	1 Bit 1 Byte 1 Byte	1.001 Switch 17.001 Scene number 5.010 Counter pulses	C		W		

This object DPT is set according to "Input object type" parameter value. Sending a telegram that complies with "Send code at" parameter value triggers the macro to start sending the IR codes of macro parts sequentially.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
Mac x	*Lock Enable/Disable*	1 Bit	1.003 Enable	C	R	W	T	U

This object is used to enable or disable the lock function of the macro.

9 AC Modules

Since the IR codes structure of the AC remotes is too complex compared to the IR codes of the other consumer IR devices remotes, it's not possible to control AC units with the same way we control normal consumer IR devices.

The AC modules form the interface between the KNX system and AC split units from a wide range of manufacturers. They convert KNX telegrams to IR codes and sends them to the split unit. The split unit's functions can therefore be operated via KNX using any operating device (switch, touch panel, mobile phone etc.)

AC modules support the following functions:

- Switching the split unit on/off
- Setting the operation mode (Auto, Heating, Cooling, Fan only, Drying) using 1-byte group object or 1-bit enabling group objects for each mode.
- Switching between heating and cooling operation modes with 1-bit simplified operation mode group object.
- Changing the temperature setpoint using 2-byte temperature group object or 1-bit increase/decrease group object.
- Limiting the temperature setpoint for cooling and heating operation modes.
- Fan level control via 1-byte object, 1-bit auto-manual switchover group object or 1-bit increase decrease fan level group object.
- Vertical and horizontal vane swing and position control using 1-byte group object, 1-bit swing on/off group object or 1-bit increase/decrease position group object.
- Lock function
- 8 scenes
- Status group objects for all controllable functions.

Please note:

- Before setting an AC module parameters and linking its group objects, the user needs to load AC remote configurations file and write it to the device using the DCA.
- Different split units sometimes have a different range of functions. Not all functions are available on every split unit. In other words, when parameterizing an AC module using ETS, the user needs to check whether the AC unit actually supports a particular function. Sometimes certain functions are shown in ETS application but are not supported by the split unit. For example: Drying mode may not be supported in some AC units. After opening a specific AC remote configuration file in the DCA, the range of the supported function are displayed.
- Communication with the split unit is unidirectional. This means that the AC module sends IR codes to the split unit, but receives no status feedback from it. So if the split unit is being operated in parallel with a remote control, the state of the AC module may differ from the actual state of the split unit. The same applies if the split unit is not ready to receive. In these cases, the user first needs to send an IR code via KNX to re-synchronize the status values.

9.1 AC Module Parameters

9.1.1 Split Unit Parameters

Name	Values	Description
Manufacturer *Remote type*		These parameters show the last written AC remote manufacturer name and remote type with the DCA. These parameters values are updated automatically by the DCA when writing AC remote configurations operation finishes successfully. Note: If the device is copied in ETS, these parameters values are copied too. However, the AC remote configurations have to be written to the copied device so the AC modules work properly.
Split unit state after main power return	**As before power outage** Specific state	These parameters define the behavior of the controlled AC split unit when the main power is returned after a power outage. These parameters are necessary to supply the synchronization with the controlled split unit. The user should select "As before power outage" if the split unit doesn't change its state and continue to work as before power outage when the main power returns. The user should select "Specific state" if the split unit starts with a specific state whatever its state was before the power outage. The user may need to cut the electric of the split unit manually many times to figure out the proper parameters values. If the user couldn't figure out the proper values for some state parameters, he should select "Unchanged".

9.1.2 General Parameters

Name	Values	Description
AC module name		The user can give the AC module an optional name to describe it. For example: Meeting room AC. This parameter value has no effect on the AC module work.
Send IR code to port 1	No **Yes**	Selecting "Yes" transmits the IR codes of the AC module to port 1.
Send IR code to port 2	**No** Yes	Selecting "Yes" transmits the IR codes of the AC module to port 2.
Send IR code to port 3	**No** Yes	Selecting "Yes" transmits the IR codes of the AC module to port 3.
Send IR code to port 4	**No** Yes	This parameter is available if "*Port 4 is used as" parameter in "General Settings" page is set to "IR transmitter".* Selecting "Yes" transmits the IR codes of the AC module to port 4.

Send IR code	**Only if AC state is changed** Always	If "Only if AC state is changed" is selected, The AC module sends IR codes to the split unit only if its state is changed via KNX objects. If "Always" is selected, The AC module sends IR codes to the split unit each time it receives a telegram from its objects even if the split unit state isn't changed. Note: The user should select "Always" If the split unit will be operated in parallel with a remote control.
Send state code after bus return	**No** Yes	If "Yes" is selected, the AC module sends an IR code to the split unit at the startup of the device.
AC state after bus return	**Latest state before bus failure** Specific state	This parameter is available if "Send state code after bus return" parameter is set to "Yes". If "Latest state before bus failure" is selected, The last sent code before bus voltage failure is resent to the split unit. If "Specific state" is selected, the user can specify the status for each function (power, setpoint, operation mode, etc.)

9.1.3 Operation Modes Parameters

Name	Values	Description
Enable simplified operation mode object	**No** Yes	Selecting "Yes" enables the 1-bit object "Simplified Operation Mode" which enables the user to switch the split unit operation mode between heating and cooling via 1-bit telegram.
Enable 1-bit object for operation modes	**No** Yes	Selecting "Yes" enables the 1-bit objects "Enable Operation Mode" which give the ability to switch to a specific mode by sending (1-bit) "Enable" telegram to its object.

9.1.4 Setpoint Parameters

Name	Values	Description
Enable increase/decrease setpoint object	**No** Yes	Selecting "Yes" enables the 1-bit object "Setpoint Increase / Decrease" which enables the user to change the setpoint with 1-bit telegram.
Enable setpoint limitation	**No** Yes	Selecting "Yes" enables the 1-bit objects "Setpoint Limitation Enable/Disable". When the limitation is enabled, if a temperature outside the temperature setpoint range is sent to the AC module, the highest/lowest permissible value respectively is sent to the unit. If the user enables the setpoint limitation while the current temperature setpoint is outside the temperature setpoint range, the temperature setpoint will be set to the upper or lower limit of the range.

		Note: setpoint limitation can be applied for heating and cooling modes only.
Cooling mode minimum setpoint	16..**20**..32	This parameter is available if “Enable setpoint limitation” parameter is set to “Yes”. This parameter determines the lower allowed temperature setpoint for cooling mode while the setpoint limitation is enabled.
Cooling mode maximum setpoint	16…**32**	This parameter is available if “Enable setpoint limitation” parameter is set to “Yes”. This parameter determines the highest allowed temperature setpoint for cooling mode while the setpoint limitation is enabled.
Heating mode minimum setpoint	**16**..32	This parameter is available if “Enable setpoint limitation” parameter is set to “Yes”. This parameter determines the lower allowed temperature setpoint for heating mode while the setpoint limitation is enabled.
Heating mode maximum setpoint	16…**25**…32	This parameter is available if “Enable setpoint limitation” parameter is set to “Yes”. This parameter determines the highest allowed temperature setpoint for heating mode while the setpoint limitation is enabled.
Limitation status after bus return	Disabled **Enabled** Read from bus As before bus failure	This parameter is available if “Enable setpoint limitation” parameter is set to “Yes”. This parameter determines the temperature setpoint limitation status after bus return. If “Read from bus” is selected, the device will send a read request for the limitation object of the AC module, if no response is received the limitation will be disabled.

9.1.5 **Fan Parameters**

In fan parameters, the lowest fan level indicates the lowest fan speed, the highest fan level indicates the highest fan speed.

Name	Values	Description
Fan level object type	**1-byte, 0-255** 1-byte, 0%-100%	This parameter specifies the DPT of “Fan Level” object. If “1-byte, 0-255” is selected, further parameters will be displayed to let the user specify the object values that activate each level. Receiving a telegram with value that has no match in fan level value parameters doesn’t have any effect. If “1-byte, 0%-100%” is selected, the value ranges for each fan level will be displayed. Receiving a value in a given range activates its fan level. The value ranges for each fan level are given in tables below.

		Note: The number of the supported fan levels is set automatically after a successful "Write AC configurations to the device" process in the DCA.
Activate fan auto with fan level object	**No** Yes	This parameter allows the user to activate auto fan level with the 1-byte fan level object. If "Fan level object type" parameter is set to "1-byte, 0-255", an additional parameter will be displayed to let the user enter "Fan Level" object value that will activate fan auto level. If "Fan level object type" parameter is set to "1-byte, 0%-100%", the user will be able to activate fan auto level by sending "0%" to "Fan Level" object.
Enable fan auto-manual switchover object	**No** Yes	This parameter enables "Fan Auto-Manual Switchover" object which allows the user to activate/deactivate fan auto level via 1-bit object.
Fan auto-manual object value	**On = auto, Off = manual** Off = auto, On = manual	This parameter specifies the value of "Fan Auto-Manual Switchover" object that activates/deactivates auto level.
Enable fan level increase/decrease object	**No** Yes	This parameter enables "Fan Level Increase/Decrease" object which allows the user to switch to the next-previous fan level via 1-bit object. Fan levels order is as below: Auto – Level 1- Level 2 – Level 3…. The maximum supported fan level.
Rollover with fan increase/decrease object	**No** Yes	If "No" is selected, the fan level will stay at the maximum/minimum level when an increase/decrease telegram is received. If "Yes" is selected, the fan level will return to the minimum level when an increase telegram is received while the current level is at the maximum level. Similarly, it will return to the maximum level when a decrease telegram is received while the current level is at the minimum level. Rollover feature example: If the maximum fan level of an AC unit is 3, Sending increase telegram repeatedly to "Fan Level Increase/Decrease" object will give the next sequence: Auto – Level 1 – Level 2 – Level 3 – Auto – Level 1 … Similarly, Sending decrease telegram repeatedly to "Fan Level Increase/Decrease" object will give the next sequence: Level 3 – Level 2 – Level 1 – Auto – Level 3 – Level 2 …

Table 1 Fan level value ranges for object Fan Level 0%-100% if auto level isn't included

Maximum supported fan level	Level 1 range	Level 2 range	Level 3 range	Level 4 range	Level 5 range	Level 6 range
Level 3	0% - 33%	34% - 66%	67% - 100%	-	-	-
Level 4	0% - 25%	26% - 50%	50% - 75%	75% - 100%	-	-
Level 5	0% - 20%	21% - 40%	41% - 60%	61% - 80%	81% - 100%	-
Level 6	0% - 16%	17% - 33%	34% - 50%	51% - 66%	67% - 83%	84% - 100%

Table 2 Fan level value ranges for object Fan Level 0%-100% if auto level is included

Maximum supported fan level	Auto level range	Level 1 range	Level 2 range	Level 3 range	Level 4 range	Level 5 range	Level 6 range
Level 3	0%	1% - 33%	34% - 66%	67% - 100%	-	-	-
Level 4	0%	1% - 25%	26% - 50%	50% - 75%	75% - 100%	-	-
Level 5	0%	1% - 20%	21% - 40%	41% - 60%	61% - 80%	81% - 100%	-
Level 6	0%	1% - 16%	17% - 33%	34% - 50%	51% - 66%	67% - 83%	84% - 100%

9.1.6 Vanes Parameters

In vertical vane parameters, the lowest vane position indicates the top vane position, the highest vane position indicates the bottom vane position.
In horizontal vane parameters, the lower vane position indicates the left vane position, the highest vane position indicates the right vane position.
Note: If vane positioning feature is not supported for the vane, only “Vane swing on/off object value” parameter is displayed.

Name	Values	Description
Vane position object type	**1-byte, 0-255** 1-byte, 0%-100%	This parameter specifies the DPT of “Vane Position” object. If “1-byte, 0-255” is selected, further parameters will be displayed to let the user specify the object values that activate each position. Receiving a telegram with value that has no match in vane position value parameters doesn’t have any effect. If “1-byte, 0%-100%” is selected, the value ranges for each vane position will be displayed. Receiving a value in a given range activates its vane position. The value ranges for each vane position are given in tables below. Note: The number of the supported vane position is set automatically after a successful “Write AC configurations to the device” process in the DCA.
Activate vane swing vane position object	**No** Yes	This parameter allows the user to activate vane swinging with the 1-byte vane position object. If “Vane position object type” parameter is set to “1-byte, 0-255”, an additional parameter will be displayed to let the user enter “Vane Position” object value that will activate vane swinging. If “Vane position object type” parameter is set to “1-byte, 0%-100%”, the user will be able to active vane swinging by sending 0% to “Vane Position” object.
Enable vane swing on/off object	**No** Yes	This parameter enables “Vane Swing” object which allows the user to activate/deactivate vane swinging via 1-bit telegrams.
Vane swing on/off object value	**1 = swing on , 0 = swing off** 0 = swing on, 1 = swing off	This parameter specifies the value of “Vane swing” object that activates/deactivates swinging.

Enable vane position increase/decrease object	**No** Yes	This parameter enables "Vane Position Increase/Decrease" object which allows the user to step to the next-previous vane position via 1-bit telegram. Vane position order is as below: Swing on – Position 1- Position 2 – Position 3…. The maximum supported vane position.
Rollover with position increase/decrease object	**No** Yes	If "No" is selected, the vane position will stay at the maximum/minimum position when an increase/decrease telegram is received. If "Yes" is selected, the vane will return to swing when an increase telegram is received while the current position is at the maximum position. Similarly, it will return to the maximum position when a decrease telegram is received while the vane is swinging. Rollover feature example: If the maximum vane position is 3, Sending increase telegram repeatedly to "Vane Position Increase/Decrease" object will give the next sequence: Swing – Position 1 – Position 2 – Position 3 – Swing – Position 1 … Similarly, Sending decrease telegram repeatedly to "Vane Position Increase/Decrease" object will give the next sequence: Position 3 – Position 2 – Position 1 – Swing – Position 3 – Position 2 …

Table 3 Vane position value ranges for object Vane Position 0%-100% if swing isn't included

Maximum supported position	Position 1 range	Position 2 range	Position 3 range	Position 4 range	Position 5 range	Position 6 range
Position 2	0% - 50%	51% - 100%	-	-	-	-
Position 3	0% - 33%	34% - 66%	67% - 100%	-	-	-
Position 4	0% - 25%	26% - 50%	50% - 75%	75% - 100%	-	-
Position 5	0% - 20%	21% - 40%	41% - 60%	61% - 80%	81% - 100%	-
Position 6	0% - 16%	17% - 33%	34% - 50%	51% - 66%	67% - 83%	84% - 100%

Table 4 Vane position value ranges for object Vane Position 0%-100% if swing is included

Maximum supported position	Swing range	Position 1 range	Position 2 range	Position 3 range	Position 4 range	Position 5 range	Position 6 range
Position 2	0%	1% - 50%	51% - 100%				
Position 3	0%	1% - 33%	34% - 66%	67% - 100%	-	-	-
Position 4	0%	1% - 25%	26% - 50%	50% - 75%	75% - 100%	-	-
Position 5	0%	1% - 20%	21% - 40%	41% - 60%	61% - 80%	81% - 100%	-
Position 6	0%	1% - 16%	17% - 33%	34% - 50%	51% - 66%	67% - 83%	84% - 100%

9.1.7 Lock Parameters

While lock function is enabled, all sent telegrams to the AC module objects are ignored.

Name	Values	Description
Enable lock function	**No** Yes	Selecting "Yes" enables the 1-bit "Lock" object and displays the other lock function parameters.
Lock status after bus return	**Unlocked** Locked Read from bus As before bus failure	This parameter determines the lock function status after bus voltage return. If "Read from bus" is selected, the device will send a read request for the lock object of the AC module, if no response is received the AC module will be unlocked.
AC state when lock enabled	**Unchanged** User-defined	This parameter determines the state of the AC module when lock function is enabled.
AC state when lock disabled	**Unchanged** User-defined As before lock	This parameter determines the state of the AC module when lock function is disabled.

9.1.8 Scenes Parameters

Name	Values	Description
Enable scene function	**No** Yes	Selecting "Yes" enables the 1-byte "Scene Number" object and displays the other scene function parameters.
AC state x assigned to scene number	**Not assigned** Scene 1 Scene 2 . . Scene 64	This parameter determines which scene number will activate the AC state when it is called via "Scene Number" object. Selecting a value other than "Not assigned" shows the state parameters.

9.1.9 Status Objects Parameters

Name	Values	Description
Enable status objects	**No** Yes	Selecting "Yes" displays all status objects parameters.
Send status values	**Don't send, update only** On change	"Don't send, update only": the status objects are updated when its values are changed but not send to the bus. The user can read each object value individually or send a telegram to "Request Status Values" object to get all status values at a time. On change: The status objects are sent to the device when its values are changed.

Enable request status values object	**No** Yes	This parameter enables 1-bit "Request status values" object. Sending a proper telegram to this object sends all enabled status objects values to the bus.
Send status values when request object is	Off **On** Off or On	This parameter determines which telegram value of "Request Status Values" object will trigger the AC module to send all enabled status objects values.

9.2 AC Module Group Objects

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Power On/Off*	1 Bit	1.001 Switch	C		W		

This object switches the split unit on and off.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Power Status*	1 Bit	1.001 Switch	C	R		T	

This object is available if "Enable power status object" parameter is set to "Yes".
This object indicates the status of the On/Off function.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Simplified Operation Mode*	1 Bit	1.100 Cooling/Heating	C		W		

This object is available if "Enable simplified mode group object" parameter is set to "Yes".
This object switches between heating and cooling operation modes.
Receiving 1 activates heating operation mode and receiving 0 activates cooling operation mode.
It is possible to operate the unit in parallel using "Operation Mode" object.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Operation Mode*	1 Byte	20.105 HVAC control mode	C		W		

This object sets the operation mode of the split unit.
The operation modes are set when this object receives a telegram value as follows:
0 = Auto, 1 = Heating, 3 = Cooling, 9 = Fan only, 14 = Drying.
All other values are discarded.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Operation Mode Status*	1 Byte	20.105 HVAC control mode	C	R		T	

This object is available if “Enable operation mode status object” parameter is set to “Yes”.
This object indicates the status of the operation mode.
0 = Auto, 1 = Heating, 3 = Cooling, 9 = Fan only, 14 = Drying.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Enable Auto Mode* *Enable Heating Mode* *Enable Cooling Mode* *Enable Fan Only Mode* *Enable Drying Mode*	1 Bit	1.003 Enable	C		W		

These objects are available if “Enable 1-bit objects for operation modes” parameter is set to “Yes”.
Sending Enable to an object activates its mode, sending Disable has no effect.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Auto Mode Enabled Status* *Heating Mode Enabled Status* *Cooling Mode Enabled Status* *Fan Only Mode Enabled Status* *Drying Mode Enabled Status*	1 Bit	1.003 Enable	C	R		T	

These objects are available if “Enable 1-bit enabled mode status objects” parameter is set to “Yes”.
These objects indicate if its operation mode is enabled or not.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Setpoint Increase / Decrease*	1 Bit	1.007 Step	C		W		

This object is available if “Enable setpoint increase/decrease” parameter is set to “Yes”.
This object increases or decreases the temperature setpoint by intervals of 1 kelvin.
If the temperature reaches the upper or lower setpoint temperature limit, further telegrams have no effect.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Setpoint Limitation Enable/Disable*	1 Bit	1.003 Enable	C		W		

This object is available if “Enable setpoint limitation” parameter is set to “Yes”.
This object enables/disabled the temperature setpoint limitation.

If the user activates the temperature setpoint limitation while the current temperature setpoint is outside the temperature setpoint range, the temperature setpoint will be set to the upper or lower limit of the range.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Setpoint*	2 Bytes	9.001 Temperature (°C)	C		W		

This object is used to set the setpoint of the AC module.
If this object receives values outside the supported range (or limitation range when it's enabled), the setpoint is set to the upper or lower limit.

This object ignores the decimal part of the received temperature (24°C = 24.2 °C = 24.9 °C).

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Setpoint Status*	2 Bytes	9.001 Temperature (°C)	C	R		T	

This object is available if "Enable setpoint status" parameter is set to "Yes".
This object indicates the current setpoint value.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Fan Level Increase/Decrease*	1 Bit	1.007 Step	C		W		

This object is available if "Enable fan level increase/decrease object" parameter is set to yes.
This object increases or decreases the fan level by one level.
Fan levels order: Auto – Level 1 – Level 2 – Level 3 - … maximum supported level.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Fan Level 0 – 255*	1 Byte	5.100 Fan stage	C		W		

This object is available if "Fan level object type" parameter is set to "1-byte, 0-255".
This object is used to activate a fan level when it receives a telegram matches the level value that is specified with "Fan level value" parameter.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Fan Level Status 0 – 255*	1 Byte	5.100 Fan stage	C	R		T	

This object is available if “Fan level object type” parameter is set to “1-byte, 0-255”, and “Enable fan level status” parameter is set to “Yes”.
This object indicates the active fan level by sending its value that is specified with “Fan level value” parameter to the bus.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Fan Level 0% – 100%*	1 Byte	5.001 Percentage	C		W		

This object is available if “Fan level object type” parameter is set to “1-byte, 0%-100%”.
This object activates a fan level when it receives a telegram value in the range of this level.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Fan Level Status 0% –100%*	1 Byte	5.001 Percentage	C	R		T	

This object is available if “Fan level object type” parameter is set to “1-byte, 0%-100%”, and “Enable fan level status” parameter is set to “Yes”.
This object indicates the active fan level by sending the upper limit of its range to the bus.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Fan Auto-Manual Switchover*	1 Bit	1.001 Switch	C		W		

This object is available if “Fan auto-manual switchover object” parameter is set to “Yes”.
This object is used to switch between fan auto and the other levels according to the received telegram value and “Fan auto-manual object value” parameter.
If manual value is received the last fan level before fan auto is activated.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Fan Auto-Manual Status*	1 Bit	1.001 Switch	C	R		T	

This object is available if “Enable fan auto-manual status object” parameters is set to “Yes”.
This object indicates if the current fan level is auto level or the other levels.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Vane Position Increase/Decrease*	1 Bit	1.007 Step	C		W		

This object is available if “Enable vane position increase/decrease object” parameter is set to yes.
This object increases or decreases the vane position by one position.
Vane positions order: Swing – Position 1 – Position 2 – Position 3 - … maximum supported position.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Vane Position 0 – 255*	1 Byte	5.010 Counter pulses	C		W		

This object is available if “Vane position object type” parameter is set to “1-byte, 0-255”.
This object is used to activate a vane position when it receives a telegram matches the position value that is specified with “Vane position value” parameter.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Vane Position Status 0 – 255*	1 Byte	5.010 Counter pulses	C	R		T	

This object is available if “Vane position object type” parameter is set to “1-byte, 0-255”, and “Enable vane position status” parameter is set to “Yes”.
This object indicates the active vane position by sending its value that is specified with “Vane position value” parameter to the bus.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Vane Position 0% – 100%*	1 Byte	5.001 Percentage	C		W		

This object is available if “Vane position object type” parameter is set to “1-byte, 0%-100%”.
This object activates a vane position when it receives a telegram value in the range of position.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Vane Position Status 0% –100%*	1 Byte	5.001 Percentage	C	R		T	

This object is available if “Vane position object type” parameter is set to “1-byte, 0%-100%”, and “Enable vane position status” parameter is set to “Yes”.
This object indicates the active vane position by sending the upper limit of its range to the bus.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Vane Swing On/Off*	1 Bit	1.001 Switch	C		W		

This object is available if “Enable vane swing on/off object” parameter is set to “Yes”.
This object is used to switch vane swing on/off according to the received telegram value and “Vane swing object value” parameter.
When swing off value is received, if vane positioning is supported the last activated fixed position before swing will be activated, If vane positioning is not supported the vane will stop swinging only.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Vane Swing Status*	1 Bit	1.001 Switch	C	R		T	

This object is available if “Enable vane swing status object” parameter is set to “Yes”.
This object indicates if the vane is swinging or not.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Scene Number*	1 Byte	17.001 Scene number	C		W		

This object is available if “Enable scene function” parameter is set to “Yes”.
This object is used to activate the AC state that is assigned to the received scene number.

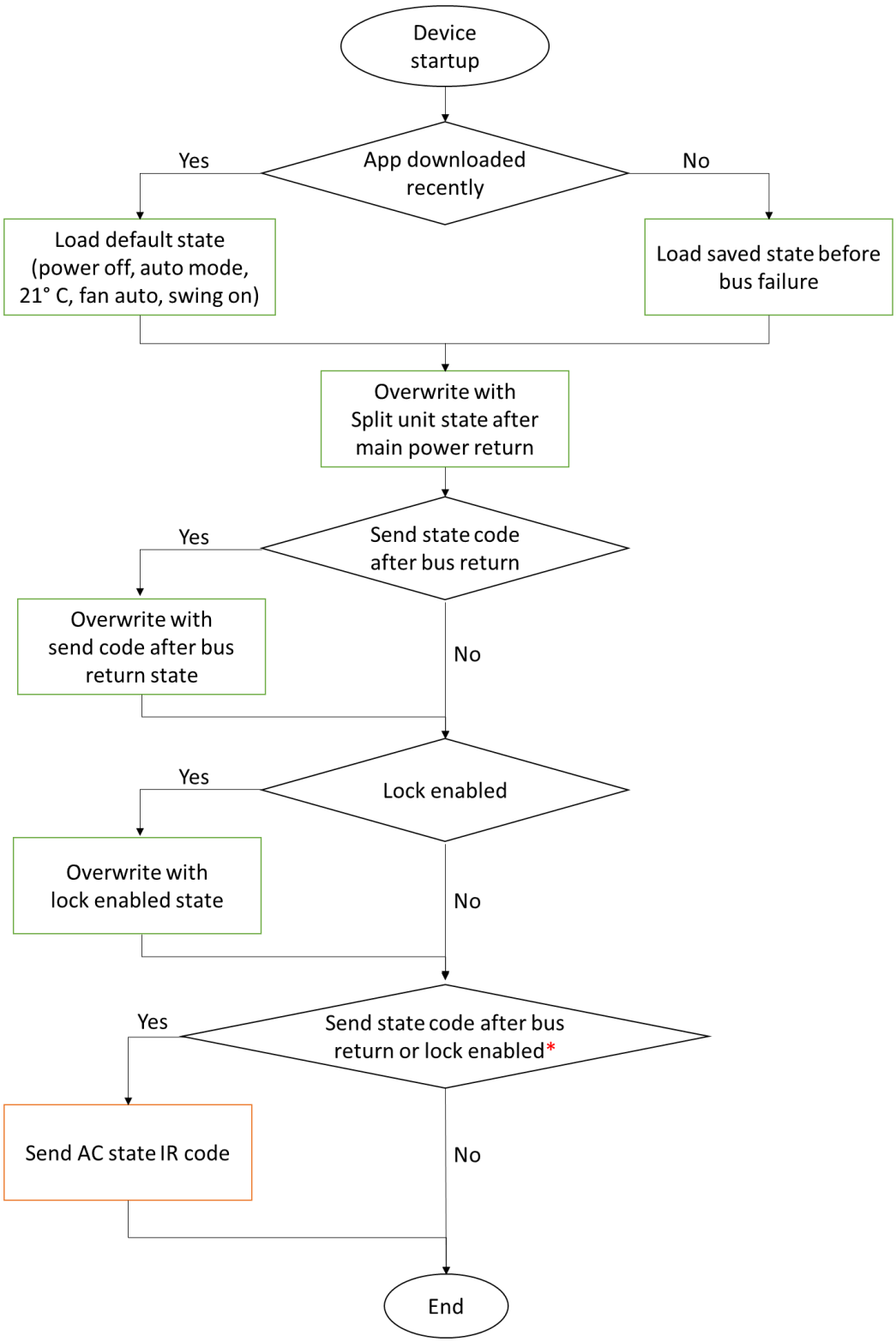
Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Request Status Values*	1 Bit	1.001 Switch 1.017 Trigger	C		W		

This object is available if “Enable request status values object” parameter is set to “Yes”.
When this object receives a proper telegram according to “Send status values when request status object value is” parameter, all enabled status objects values are sent to the bus.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AC x	*Lock*	1 Bit	1.003 Enable	C	R	W	T	U

This object is available if “Enable lock function” parameter is set to “Yes”.
This object is used to enable/disable the lock function of the AC module. When lock function is activated all sent telegrams to the AC module objects are ignored.

9.3 AC Module Behavior at Start-up



*In case of a state IR code will be sent because the lock is enabled at the startup,
"Send IR code: only if state is changed – always" parameter value is taken into the account.

10 Auxiliary Functions

The KNX IR Interface have 8 independent auxiliary functions. The user can choose its type from “Auxiliary Functions” page.

There are 5 types of auxiliary functions:

1. Logic gate
2. Sequencer / Counter
3. Converter
4. Scene actuator
5. Send after reset

Each type has own objects and parameters and all types have lock feature.

10.1 Logic Gate

This function acts as a logic gate with maximum 8 x 1-bit input objects, and one output object.
The output object type can be 1-bit, scene, percentage, 1-byte, 2-byte or float.

10.1.1 Logic Gate Parameters

Name	Values	Description
Auxiliary function name		The user can give the auxiliary function a name for documentation purposes. This parameter value has no effect on the function work.
Logic gate type	**AND** OR XOR NAND NOR XNOR One hot NOT	AND: gives a true output only if all its inputs are 1. OR: gives a true output if one or more of its inputs are 1. XOR: gives a true output when the number of 1 inputs is odd. NAND: Its output is true if any of the inputs are 0. NOR: produces an output which is false only if all its inputs are 1. XNOR: gives a true output when the number of 1 inputs is even. One hot: gives a true output if there is only one input is 1. For example: The gate gives true with these input values: 0-1-0-0-0-0 or 0-0-0-0-0-1 or 1-0-0-0-0-0 The gate gives false with these input values: 0-0-0-0-0-0 or 0-1-1-0-0-0 or 1-1-1-1-1-1 NOT: produces an inverted version of the input at its output. It is also known as an inverter.
Number of used inputs	**2** … 8	This parameter is shown if the logic gate type is not “NOT” gate. It defines how many inputs the logic gate will have.
Number of used NOT gates	**1** … 4	This parameter is shown if the logic gate type is “NOT” gate. It defines how many NOT gate the auxiliary function will have.
Input X polarity	**Normal** Inverted	This parameter is shown if the logic gate type is not “NOT” gate. It is used to invert the input object value, e.g. receiving 1 from the input object gives 0 to the gate.
Input X value after mains voltage recovery	**0** 1 As before bus failure Read from bus Block output until new telegram is received	This parameter is shown if the logic gate type is not “NOT” gate. If “Read from bus” is selected, the function will send read request for the input object. If no response is received, the input value will be 0. If “Block output until new telegram is received” is selected, the output value will not be sent to the bus until the logic gate receives a telegram from this input.
Output object type	**1-bit** Scene number Percentage	This parameter is shown if the logic gate type is not “NOT” gate.

	1-byte 2-byte Float	This parameter defines the DPT of the output object of the logic gate, and according to its value two parameters will be shown to specify the output values for true and false.
Send output telegram when	**A new input telegram is received** Output changes	If "A new input telegram is received" the gate will send an output telegram every time a telegram is sent to an input even if the output state doesn't change. If "Output changes" the gate will send an output telegram only when its state changes from true to false or from false to true.
Send output after delay	**No** Yes	This parameter is shown if the logic gate type is not "NOT" gate. It enables a delay before sending the output value.
Delay time	00:00:00…**00:00:10**…09:06:07	This parameter is available if "Send output after delay" parameter is set to "Yes". It defines the delay time before the output value is sent.
Send output cyclically	**No** Yes	This parameter is shown if the logic gate type is not "NOT" gate. It enables sending the output value cyclically to the bus.
Cycle time	00:00:00…**00:00:10**…09:06:07	This parameter is available if "Send output cyclically" parameter is set to "Yes". It defines the time period between the repeated output telegrams
Lock status after bus return	**Unlocked** Locked Read from bus As before bus failure	This parameter determines the status of the function's lock after bus voltage return. If "Read from bus" is selected, the device will send a read request for the lock object of the function, if no response is received the function will be unlocked.

10.1.2 Logic Gate Group Objects

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Logic Gate *AF n – NOT Gate x*	*Input x* *Input*	1 Bit	1.002 Boolean	C		W	T	U

The input objects of the logic gate.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Logic Gate	*Output*	1 Bit 1 Byte 1 Byte 1 Byte 2 Bytes 2 Bytes	1.001 Switch 17.001 Scene number 5.001 Percentage 5.010 Counter pulses 7.001 Pulses 9.003 Kelvin/Hour	C	R		T	
AF n – NOT Gate x	*Output*	1 Bit	1.002 Boolean					

The output object of the logic gate. Its DPT is specified by "Output object type" parameter.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Logic Gate	*Lock*	1 Bit	1.003 Enable	C	R	W	T	U

This object is used to enable/disable the lock of the auxiliary function.

10.2 Sequencer / Counter

The sequencer function sends the next/previous step value when it receives a valid telegram from its input object. The number of steps is configurable. An additional 1-bit telegram can be send with specific value for each step along side with output telegram. The output object type can be different for each step, or the same for all steps.

The counter function increases/decreases the output value when it receives a valid telegram from its input object. Its input and output objects types can be selected from many different types. The counter limits and step amount are configurable.

10.2.1 Sequencer / Counter Parameters

Name	Values	Description
Auxiliary function name		The user can give the auxiliary function a name for documentation purposes. This parameter value has no effect on the function work.
Function type	**Sequencer** Counter	The sequencer function sends the next/previous step value when it receives a valid telegram from its input object. The counter function increases/decreases the output value when it receives a valid telegram from its input object.
Sequencing / Counting type	**Cyclic** Up-Down Two directions	This parameter defines the function behavior. Cyclic: The function goes up always and when it reaches its last step / upper limit it returns to the first step / lower limit. For example, in a sequencer with 3 steps the sequence goes as below: Step 1 – Step 2 –Step 3 – Step1 – Step 2… Up-Down: The function goes up at the beginning and then it changes the direction every time it reaches its limits. For example, in a sequencer with 3 steps the sequence goes as below: Step 1 – Step 2 –Step 3 – Step2 – Step 1 – Step 2 – Step 3 … Two directions: the user selects the input value where the function will go up, and the input value where the function will go down. For example, on telegrams for up, off telegrams for down.
Input object type	**1-bit** Scene number	This parameters defines the DPT of the input object.
Next step at *Next step at scene number*	**Any input telegram value** On telegrams only Off telegrams only **1**…64	This parameter defines which input value will trigger the function to go to the next step.
Previous step at scene number	1…**2**…64	This parameter is available if “Sequencing / Counting type” is set to “Two directions”.

		It defines which input value will trigger the function to go to the previous step.
Sequencer parameters		
Number of steps	**2**…5	This parameter defines how many step will be used in the sequencer function.
Number of output objects	**One output object for all steps** One output object for each step	This parameter defines how many object will be used for the steps. If "One output object for each step" is selected, each step can have an independent object with a different DPT.
Use additional 1-bit output object	**No** Yes	This parameter enables an additional 1-bit object that can be used to send a 1-bit telegram with a specific value for each step along side with the step output telegram. If it is enabled, further parameters will be shown to enable the user to enter the additional 1-bit object value for each step.
Output object type	**1-bit** Scene number Percentage 1-byte 2-byte Float	This parameter is available if "Number of output objects" parameter is set to "One output object for all steps". It defines the DPT of the sequencer's steps object.
Step x output object type	**1-bit** Scene number Percentage 1-byte 2-byte Float	This parameter is available if "Number of output objects" parameter is set to "One output object for each step". It defines the DPT of a step object.
Step x output value	**0**…1 **1**…64 **0%**...100% **0**…255 **0**…65535 -670760…**0**…670760	This parameter defines the telegram value that will be sent through the step/s object when the step is reached.
Step x additional 1-bit object value	**0**..1	This parameter is available if "Use additional 1-bit output object" parameter is set to "Yes". This parameter defines the telegram value that will be sent through the additional 1-bit object when the step is reached.
After bus return start from	**Step 1** Last sent step before bus failure	This parameter defines the behavior of the function after bus voltage return.
Counter parameters		
Counter output object type	**Scene number** Percentage 1-byte 2-byte Float	This parameter defines the DPT of the counter's output object.
Counter lower limit	**1**…64 **0%**...100%	This parameter defines the counter lower limit.

	0…255 **0**…65535 -670760…**18**…670760	
Counter upper limit	1…**64** 0%...**100%** 0…**255** 0…**65535** -670760…**30**…670760	This parameter defines the counter upper limit.
Counter step	**1**…64 0%...**1%**...100% 0…**1**…255 0…**1**…65535 -670760…**1**…670760	This parameter defines the counter step.
After bus return start from	**Lower limit** Last sent value before bus failure	This parameter defines the behavior of the function after bus voltage return.
Send output after delay	**No** Yes	This parameter enables a delay before sending the output value.
Delay time	00:00:00…**00:00:10**…09:06:07	This parameter is available if “Send output after delay” parameter is set to “Yes”. It defines the delay time before the output value is sent.
Send output cyclically	**No** Yes	This parameter enables sending the output value cyclically to the bus.
Cycle time	00:00:00…**00:00:10**…09:06:07	This parameter is available if “Send output cyclically” parameter is set to “Yes”. It defines the time period between the repeated output telegrams.
Lock status after bus return	**Unlocked** Locked Read from bus As before bus failure	This parameter determines the status of the function’s lock after bus voltage return. If “Read from bus” is selected, the device will send a read request for the lock object of the function, if no response is received the function will be unlocked.

10.2.2 Sequencer / Counter Group Objects

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Sequencer / Counter	*1-Bit Input* *Scene Number Input*	1 Bit 1 Byte	1.001 Switch 17.001 Scene Number	C		W		

The input object of the sequencer / counter function. Its DPT is specified by “Input object type” parameter.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Sequencer / Counter	*Output*	1 Bit 1 Byte 1 Byte 1 Byte 2 Bytes 2 Bytes	1.001 Switch 17.001 Scene number 5.001 Percentage 5.010 Counter pulses 7.001 Pulses 9.003 Kelvin/Hour	C	R		T	

This object is available if “Function type” parameter is set to “Counter”, or if it is set to “Sequencer” and “Number of output objects” parameter is set to “One output object for all steps”.
Its DPT is specified by “Output object type” or “Counter output object type” parameters.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Sequencer / Counter	*Output x*	1 Bit 1 Byte 1 Byte 1 Byte 2 Bytes 2 Bytes	1.001 Switch 17.001 Scene number 5.001 Percentage 5.010 Counter pulses 7.001 Pulses 9.003 Kelvin/Hour	C	R		T	

This object is available if “Function type” parameter is set to “Sequencer” and “Number of output objects” parameter is set to “One output object for each step”.
Its DPT is specified by “Step x output object type” parameter.
It represents the output object of a step in the sequencer function.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Sequencer / Counter	*Lock*	1 Bit	1.003 Enable	C	R	W	T	U

This object is used to enable/disable the lock of the auxiliary function.

10.3Converter

The converter function is used to convert data point types and/or telegram values.
There are 5 converter types:

1- User customized: The user specifies the input and the output object types, the comparison statement (greater than, equal to, between, etc…) for the input value, and the output values for the comparison result.
2- 8 x 1-bit => 1 x 1-byte: Combines 8 1-bit objects into a 1-byte object
3- 1 x 1-byte => 8 x 1-bit: separates a 1-byte object to 8 1-bit objects
4- 2 x 1-byte => 1 x 2-byte: Combines 2 1-byte objects into a 2-byte object
5- 1 x 2-byte => 2 x 1-byte: separates a 2-byte object to 2 1-byte objects

For all converter types the converter can be bidirectional (converts the telegrams in two directions).

10.3.1 Converter Parameters

Name	Values	Description
Auxiliary function name		The user can give the auxiliary function a name for documentation purposes. This parameter value has no effect on the function work.
Converter type	**User customized** 8 x 1-bit => 1 x1-byte 1 x 1-byte => 8 x 1-bit 2 x 1-byte => 1 x 2-byte 1 x 2-byte => 2 x 1-byte	This parameter defines the type of the converter. In user customized converters, the DPT of the input and the output objects, the converting conditions and the output values can be specified by the user. This type is used to convert KNX telegrams to another types and values. The other converter types are used to convert telegrams to another types only. Its values are not changed.
Bidirectional	**No** Yes	The bidirectional converter converts the telegrams in two directions. From X terminal object\s to Y terminal object\s and from Y terminal object\s to X terminal object\s. If “Yes” is selected, “Send output after delay” and “Send output cyclically” parameters cannot be used.
User customized converter parameters		
X/Y terminal object type	**1-bit** Scene number Percentage 1-byte 2-byte Float	This parameter defines the DPT of the terminal object. If the converter is not bidirectional, X terminal is the input and Y terminal is the output, else both terminals can be operated as input and output.
If received X/Y terminal object value is	**Equal to** Unequal to Lower than	This parameter defines the condition of the converter.

	Equal or lower than Greater than Equal or greater than Between	If the terminal object type is “1-bit”, only two conditions can be chosen, “Equal to” and “Unequal to”.
Value *Scene number*	**0**…1 **0%**...100% **0**…255 **0**…65535 -670760…**0**…670760 **1**…64	This parameter defines the test value of the condition.
And value *And scene number*	**0%**...100% **0**…255 **0**…65535 -670760…**0**…670760 **1**…64	This parameter is available if “If received X/Y terminal object value” parameter is set to “Between”. It defines the second test value of “Between” condition.
Then send to Y/X terminal value *Scene number*	**0**…1 **0%**...100% **0**…255 **0**…65535 -670760…**0**…670760 **1**…64	This parameter defines the telegram value to be sent when the condition is met.
Else	**Don’t send telegram** Send telegram	This parameter defines the behavior of the converter when the condition is not met.
Value *Scene number*	**0**…1 **0%**...100% **0**…255 **0**…65535 -670760…**0**…670760 **1**…64	This parameter is available if “Else” parameter is set to “Send telegram”. It defines the telegram value to be sent when the condition is not met.
Send telegram when	**New telegram is received** Terminal object value changes	If “New telegram is received” is selected, the converter sends telegrams to a terminal every time it receives new telegrams from the other terminal. If “Terminal object value changes” is selected, the converter sends telegrams to a terminal only when its object value is changed.
Send output after delay	**No** Yes	This parameter enables a delay before sending the output value.
Delay time	00:00:00…**00:00:10**…09:06:07	This parameter is available if “Send output after delay” parameter is set to “Yes”. It defines the delay time before the output value is sent.
Send output cyclically	**No** Yes	This parameter enables sending the output value cyclically to the bus.
Cycle time	00:00:00…**00:00:10**…09:06:07	This parameter is available if “Send output cyclically” parameter is set to “Yes”. It defines the time period between the repeated terminal telegrams

Converter behavior after bus return	**Wait for new telegrams** Read X terminal objects Read Y terminal objects	This parameter defines the converter behavior after bus voltage return. If "Wait for new telegrams" is selected, no action will be taken. If "Read X/Y terminal objects" is selected, the converter will send read request for the terminal objects to the bus after bus return. Note: To be able to read terminal objects from the bus, the terminal should be an input terminal.
Lock status after bus return	**Unlocked** Locked Read from bus As before bus failure	This parameter determines the status of the function's lock after bus voltage return. If "Read from bus" is selected, the device will send a read request for the lock object of the function, if no response is received the function will be unlocked.

10.3.2 Converter Group Objects

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Converter	*X Terminal - Input*	1 Bit	1.001 Switch					
		1 Byte	17.001 Scene number					
	X Terminal - Input	1 Byte	5.001 Percentage	C		W	T	U
	X Terminal – Input / Output	1 Byte	5.010 Counter pulses	C	R	W	T	U
		2 Bytes	7.001 Pulses					
		2 Bytes	9.003 Kelvin/Hour					

This object is available if the converter type is "User customized".
If the converter is bidirectional X terminal is used as an input and an output for the converter, else it is used as an input only.
This object DPT is determined by "X terminal object type" parameter.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Converter	*X Terminal – 1-Byte (LSB) - Input*	1 Byte	5.010 Counter pulses	C		W	T	U
	X Terminal – 1-Byte (LSB) - Output			C	R	W	T	
	X Terminal – 1-Byte (LSB) – Input / Output			C	R	W	T	U
	X Terminal – 1-Byte (MSB) – Input			C		W	T	U
	X Terminal – 1-Byte (MSB) - Output			C	R	W	T	
	X Terminal – 1-Byte (MSB) – Input / Output			C	R	W	T	U

These objects are available if the converter type is "2 x 1-byte => 1 x 2-byte" or "1 x 2-byte => 2 x 1-byte".
LSB objects represents the least significant byte of the terminal. MSB objects represents the most significant byte of the terminal.
If the converter type is "2 x 1-byte => 1 x 2-byte", X terminal is used as an input terminal. If the converter type is "1 x 2-byte => 2 x 1-byte", X terminal is used as an output terminal.
If the converter is bidirectional, X terminal is used as an input and output terminal.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Converter	*X Terminal – Bit i – Input* *X Terminal – Bit I – Output* *X Terminal – Bit i – Input / Output*	1 Bit	1.001 Switch	C C C	 R R	W W W	T T T	U U

These objects are available if the converter type is “8 x 1-bit => 1 x1-byte” or “1 x 1-byte => 8 x 1-bit”.
Each object represents the i-th bit of the terminal.
If the converter type is “8 x 1-bit => 1 x1-byte”, X terminal is used as an input terminal.
If the converter type is “1 x 1-byte => 8 x 1-bit”, X terminal is used as an output terminal.
If the converter is bidirectional, X terminal is used as an input and output terminal.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Converter	*Y Terminal - Output* *Y Terminal – Input / Output*	1 Bit 1 Byte 1 Byte 1 Byte 2 Bytes 2 Bytes	1.001 Switch 17.001 Scene number 5.001 Percentage 5.010 Counter pulses 7.001 Pulses 9.003 Kelvin/Hour	C C	 R	W W	T T	U U

This object is available if the converter type is “User customized”.
If the converter is bidirectional Y terminal is used as an input and an output for the converter, else it is used as an output only.
This object DPT is determined by “Y terminal object type” parameter.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Converter	*Y Terminal – 1-Byte - Input* *Y Terminal – 1-Byte - Output* *Y Terminal – 1-Byte – Input / Output*	1 Byte	5.010 Counter pulses	C C C	 R R	W W W	T T T	U U

This object is available if the converter type is “8 x 1-bit => 1 x1-byte” or “1 x 1-byte => 8 x 1-bit”.
It represents the byte object of the terminal.
If the converter type is “8 x 1-bit => 1 x1-byte”, Y terminal is used as an output terminal.
If the converter type is “1 x 1-byte => 8 x 1-bit”, Y terminal is used as an input terminal.
If the converter is bidirectional, Y terminal is used as an input and output terminal.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Converter	*Y Terminal – 2-Byte - Input* *Y Terminal – 2-Byte - Output* *Y Terminal – 2-Byte – Input / Output*	2 Bytes	7.001 Pulses	C C C	 R R	W W W	T T T	U U

This object is available if the converter type is “2 x 1-byte => 1 x 2-byte” or “1 x 2-byte => 2 x 1-byte”.

It represents the 2-byte object of the terminal.
If the converter type is “2 x 1-byte => 1 x 2-byte”, Y terminal is used as an output terminal.
If the converter type is “1 x 2-byte => 2 x 1-byte”, Y terminal is used as an input terminal.
If the converter is bidirectional, Y terminal is used as an input and output terminal.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Converter	*Lock*	1 Bit	1.003 Enable	C	R	W	T	U

This object is used to enable/disable the lock of the auxiliary function.

10.4Scene Actuator

Scene actuator function has one scene input and 8 outputs. It sends the configured output values when it receives the set scene number. The output values can be from different types.
Scene actuator values can be overridden when a “Learn” telegram is received.
The installer can keep the learned output values after an ETS download operation and don’t overwrite it.
A delay time can be set before the first output’s telegram, and another one between the outputs’ telegrams.

10.4.1 Scene Actuator Parameters

Name	Values	Description
Auxiliary function name		The user can give the auxiliary function a name for documentation purposes. This parameter value has no effect on the function work.
Scene number for actuating	**1**…64	This parameter defines the scene number that will trigger the function to send its output telegrams.
Overwrite output values at download	No **Yes**	If “No” is selected, The previously downloaded or learned output values will persist after the ETS download operation, and the new downloaded output values will be ignored. If “Yes” is selected, the downloaded output values will overwrite the previously downloaded or learned ones. Note: If “No” will be selected, the output types shouldn’t be changed else the output values are undefined.
Save output values with learn telegrams	**No** Yes	If “Yes” is selected, the output values can be overwritten with the last sent values to the output object when a learn scene telegram is received.
Stop sending telegrams if a scene with different number is called	**No** Yes	Due to the configurable time delays parameters of the scene actuator function, actuating a scene may take long time and may not be executed immediately. If “Yes” is selected, the scene actuator will stop sending its output telegrams when it receives a different scene number than its scene number. For example: “Yes” should be selected when “Good bye” scene number is wanted to cancel “Welcome” scene number because they have opposite operations.
Actuating startup delay	**00:00:00…**09:06:07	This parameter defines the delay time before sending the first output value to the bus.
Delay between output telegrams	00:00:00…**00:00:02**…09:06:07	This parameter defines the delay time between the output telegrams. 00:00:00 value means all telegrams are send together without delay time.
Output x type	**Not used** 1-bit	This parameter defines the DPT of the x-th output object. If “Not used” is selected no telegram is sent for this output.

	Scene number Percentage 1-byte 2-byte Float	
Output x value	**0**…1 **1**…64 **0%**...100% **0**…255 **0**…65535 -670760…**0**…670760	This parameter defines the telegram value that will be sent for the x-th output. Note: This value can be overwritten when learn scene telegram is received if "Save output values with learn telegrams" parameter is set to "Yes".
Lock status after bus return	**Unlocked** Locked Read from bus As before bus failure	This parameter determines the status of the function's lock after bus voltage return. If "Read from bus" is selected, the device will send a read request for the lock object of the function, if no response is received the function will be unlocked.

10.4.2 Scene Actuator Group Objects

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Scene Actuator	*Scene Number Input*	1 Byte	17.001 Scene number	C			T	

Sending a scene number that matches the value of "Scene number for actuating" parameter to this object, triggers the scene actuator to start sending its output values.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Scene Actuator	*Output x*	1 Bit 1 Byte 1 Byte 1 Byte 2 Bytes 2 Bytes	1.001 Switch 17.001 Scene number 5.001 Percentage 5.010 Counter pulses 7.001 Pulses 9.003 Kelvin/Hour	C		W	T	

This object is available if the "Output x type" parameter is not set to "Not used".
Its DPT is specified with "Output x type" parameter.
It sends the x-th output value when the scene actuator is triggered.
If "Save output values with learn telegrams" parameter is set to "Yes", sending learn scene telegram to "Scene Number Input" object after writing a value to this object overwrites the output value with the last written object value.

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Converter	*Lock*	1 Bit	1.003 Enable	C	R	W	T	U

This object is used to enable/disable the lock of the auxiliary function.

10.5Send After Reset

Send after reset function can be used to:

1- Send a telegram with specific value or a read request when the device starts up.
2- Save the last sent value before an electric cut off and resend it when the bus voltage returns.
3- Send a read request or a telegram cyclically to the bus.

10.5.1 Send After Reset Parameters

Name	Values	Description
Auxiliary function name		The user can give the auxiliary function a name for documentation purposes. This parameter value has no effect on the function work.
After reset send	**Value telegram** Read request	This parameter defines what to send after reset.
Output object type	**1-bit** Scene number Percentage 1-byte 2-byte Float	This parameter defines the DPT of the output object of this function.
Output value	**0**…1 **1**…64 **0%**...100% **0**…255 **0**…65535 -670760…**0**…670760	This parameter is available if “After reset send” parameter is set to “Value telegram”. It defines the telegram value that will be sent after reset.
Overwrite output value when a telegram is received	**No** Yes	This parameter is available if “After reset send” parameter is set to “Value telegram”. If “Yes” is selected, receiving a telegram from the output object overwrites the output value. This feature can be used to save the last sent value before an electric cut off and resend it when the bus voltage returns.
Send output after delay	**No** Yes	This parameter enables a delay before sending the output value at startup.
Delay time	00:00:00…**00:00:10**…09:06:07	This parameter is available if “Send output after delay” parameter is set to “Yes”. It defines the delay time before the output value is sent.
Send output cyclically	**No** Yes	This parameter enables sending the output value cyclically to the bus.
Cycle time	00:00:00…**00:00:10**…09:06:07	This parameter is available if “Send output cyclically” parameter is set to “Yes”. It defines the time period between the repeated output telegrams

Lock status after bus return	**Unlocked** Locked Read from bus As before bus failure	This parameter determines the status of the function's lock after bus voltage return. If "Read from bus" is selected, the device will send a read request for the lock object of the function, if no response is received the function will be unlocked.

10.5.2 Send After Reset Group Objects

Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Send After Reset	*Output*	1 Bit 1 Byte 1 Byte 1 Byte 2 Bytes 2 Bytes	1.001 Switch 17.001 Scene number 5.001 Percentage 5.010 Counter pulses 7.001 Pulses 9.003 Kelvin/Hour	C	R	W	T	U

This object DPT is specified with "Output object type" parameter.
It sends the output value or read request after the device starts up according to the set parameters.
If "Overwrite output value when a telegram is received" parameter is set to "Yes", receiving a telegram from this object overwrites the output value.

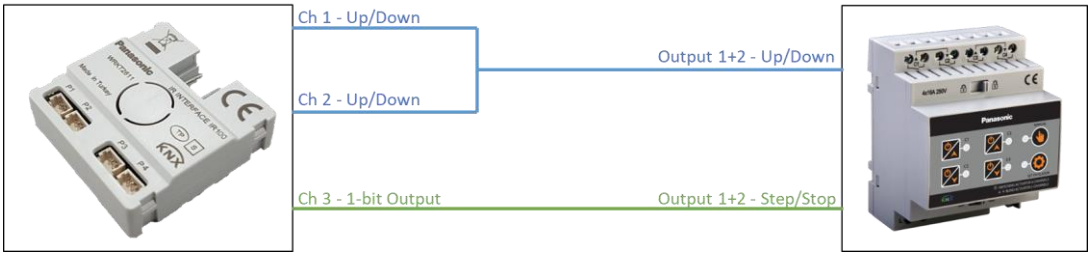
Object Name	Function	Size	Datapoint Type	Flags				
				C	R	W	T	U
AF n – Converter	*Lock*	1 Bit	1.003 Enable	C	R	W	T	U

This object is used to enable/disable the lock of the auxiliary function.

11 Some Examples of Typical Applications

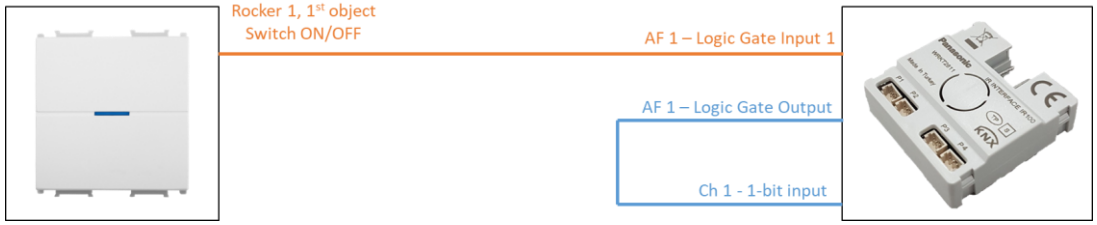
11.1 Controlling Blinds with an IR Remote

It's possible to control KNX devices with IR remotes using IR->KNX channels.
In this example, 3 buttons from an IR remote are used to control blinds which are connected to a KNX Mix Actuator using the KNX IR interface. The first button moves the blinds up, the second button moves the blinds down, and the third button stops the blinds movement.

Used devices	KNX IR Interface IR100 (WRKT26115NC) with an IR receiver connected to port 4 KNX Mix Actuator 4 Channels MX104 (WRKT4604E)
Objects linking	
KNX IR Interface parameters	• Port 4 is used as: IR receiver • Used channel number: 3 • Ch 1 – Channel type: IR->KNX • Ch 1 – Code number: 1 (where 1 is the first button code number) • Ch 1 – Output object type : Up / Down • Ch 1 – Output object value: Up • Ch 2 – Channel type: IR->KNX • Ch 2 – Code number: 2 (where 2 is the second button code number) • Ch 2 – Output object type : Up / Down • Ch 2 – Output object value: Down • Ch 3 – Channel type: IR->KNX • Ch 3 – Code number: 3 (where 3 is the third button code number) • Ch 3 – Output object type : 1-bit • Ch 3 – Output object value: Off
KNX Mix Actuator parameters	• Group 1 – Output 1 Selection: Shutter\Blind • Output 1+2 Shutter\Blind Settings – Shutter selection : Blind (With slats)
	The unmentioned parameters can be the default or user defined parameters
Notes	The three buttons' IR codes must be taught to the device using the DCA

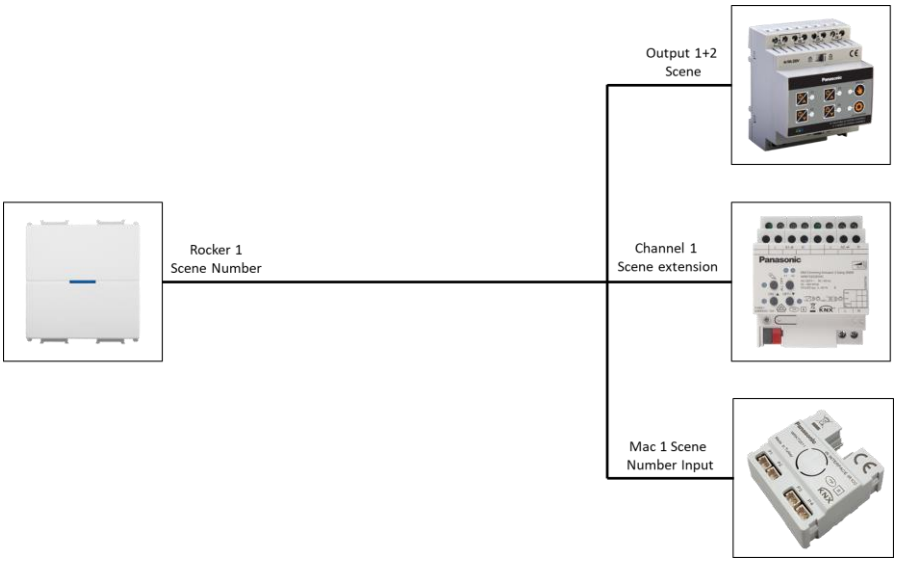
11.2 Turning TV On/Off with One IR code

Some consumer IR devices have one button to switch the device On/Off. If the device is intended to be controlled with KNX->IR channels, sending on telegram to the input object of the channel while the device is turned on already, turns the device off. To avoid this behavior, IR codes must be sent only when the object value changes from on to off or from off to on. This can be achieved by using the auxiliary functions in the KNX IR Interface device.
Note: The communications with consumer IR devices are unidirectional. This means that the KNX IR Interface sends IR codes to the controlled consumer IR devices, but receives no status feedback from it.

Used devices	KNX IR Interface IR100 (WRKT26115NC) with an IR transmitter connected to port 1 KNX Modular Switch 1 gang (WRKT6121) (or any switching devices like touch panels, smart mobile phones etc.)
Objects linking	
KNX IR Interface parameters	• Used channel number: 1 • Ch 1 – Channel type: KNX->IR • Ch 1 – Input object type : 1-bit • Ch 1 – Send code at: Any telegram value • Ch 1 – Code number: 1 (where 1 is switch device on/off button code number) • Ch 1 – Send IR code to port 1: Yes • Ch 1 – Code transmission count: 1 • Auxiliary function 1: Logic gate • AF 1 – Logic gate type: OR • AF 1 – Number of used inputs:2 • AF 1 – Input 1 & 2 polarity: Normal • AF 1 – Input 1 & 2 value after bus return: 0 • AF 1 – Output object type: 1-bit • AF 1 – Output value for true: 1 • AF 1 – Output value for false: 0 • **AF 1 – Send output telegram when: Output changes** • AF 1 – Send output cyclically: No
KNX Modular Switch parameters	• General – Function of Rocker 1: switching • Rocker 1 – Object type for 1st object of rocker: switching (1bit)
	The unmentioned parameters can be the default or user defined parameters
Notes	Switch device on/off button's IR codes must be taught to the device using the DCA

11.3 Sending Sequential IR Codes with KNX Scenes

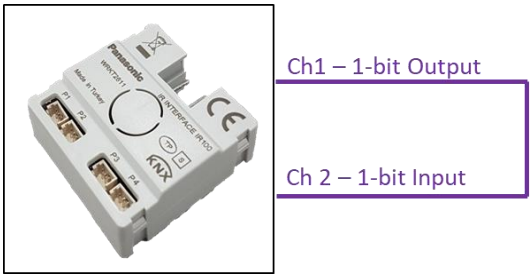
IR->KNX channels can be used to activate KNX scenes with IR remotes. KNX->IR channels and macros can participate in KNX scenes to send IR codes.
In this example, a demonstration of how KNX devices and KNX IR Interface can be used in “Start Presentation” scene. In a meeting room, when “Start Presentation” scene is called, the blinds are closed using KNX Mix Actuator, the lights are dimmed down using KNX Dimming Actuator, and the projector is turned on and configured to use the HDMI input with the KNX IR Interface.

Used devices	KNX IR Interface IR100 (WRKT26115NC) with an IR transmitter connected to port 1 KNX Mix Actuator 4 Channels MX104 (WRKT4604E) KNX Dimming Actuator 2 gangs 300W (WRKT5502E) KNX Modular Switch 1 gang (WRKT6121) (or any device can call KNX scenes like touch panels, smart mobile phones etc.)
Objects linking	
KNX IR Interface parameters	• Used macro number: 1 • Mac 1 – Input object type: Scene number • Mac 1 – Send code at scene number: 1 • Mac 1 – Send IR code to port 1: Yes • Mac 1 – Part 1 – Code number: 1 (where 1 is projector’s switch on button IR code number) • Mac 1 – Part 2 – Code number: 2 (where 2 is projector’s HDMI input button IR code number) • Mac 1 – Part 2 – Send IR code after delay: 100 x 100 ms (because the projector takes 10 seconds at the startup to be able to receive IR codes).
KNX Mix Actuator parameters	• Group 1 – Output 1 Selection: Shutter\Blind • Output 1+2 Shutter\Blind Settings – Shutter selection : Blind (With slats) • Output 1+2 Shutter\Blind Settings – Scene: Enabled • Output 1+2 Shutter\Blind Settings – Scene – Scene 1– Scene number: 1 • Output 1+2 Shutter\Blind Settings – Scene – Scene 1– Position: 100% • Output 1+2 Shutter\Blind Settings – Scene – Scene 1– Slat percentage: 100%

KNX Dimming Actuator parameters	• K1 – Enabled functions – Scene function: enabled • K1 – Scenes – Scene 1 activatable by scene number: 1 • K1 – Scenes – Brightness value for scene 1: 20%
KNX Modular Switch parameters	• General – Function of Rocker 1: scene • Rocker 1 – Scene number for upper side: 1
	The unmentioned parameters can be the default or user defined parameters
Notes	Projector remote control buttons' IR codes must be taught to the device using the DCA

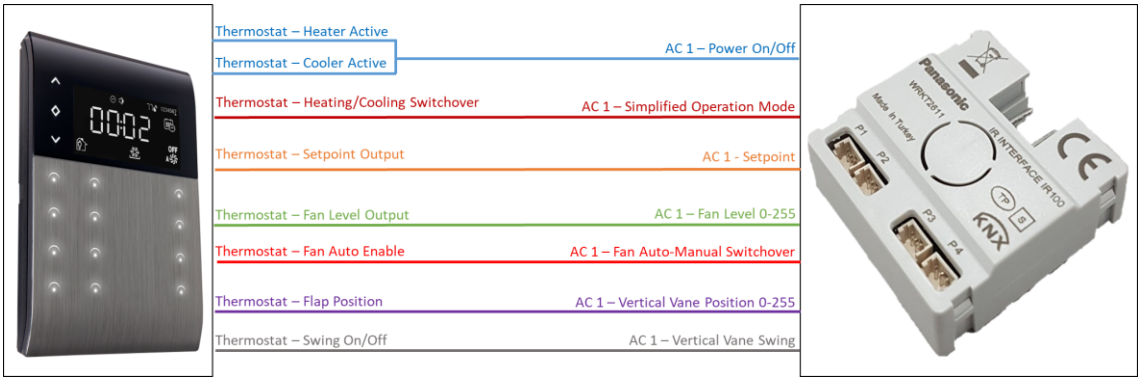
11.4 Controlling Many Different Consumer IR Devices with One IR Remote

It's possible to control many consumer IR devices with its not original remote control by linking the output of an IR->KNX channel to the input of a KNX->IR channel.
In this example, the KNX IR Interface is used to turn on/off a satellite TV receiver when a TV is turned on/off by its remote control.

Used devices	KNX IR Interface IR100 (WRKT26115NC) with an IR transmitter connected to port 1 and an IR receiver connected to port 4
Objects linking	
KNX IR Interface parameters	• Port 4 is used as: IR receiver • Used channel number: 2 • Ch 1 – Channel type: IR->KNX • Ch 1 – Code number: 1 (where 1 is the code number of TV on/off button IR code) • Ch 1 – Output object type : 1-bit • Ch 1 – Output object value: On • Ch 1 – Send cyclically: No • Ch 2 – Channel type: KNX->IR • Ch 2 – Input object type : 1-bit • Ch 2 – Send code at: On telegrams only • Ch 2 – Code number: 2 (where 2 is the code number of satellite TV receiver on/off button IR code) • Ch 2 – Send IR code to port 1: Yes • Ch 2 – IR code transmission count: 1
	The unmentioned parameters can be the default or user defined parameters
Notes	All remote control buttons' IR codes must be taught to the device using the DCA

11.5 Controlling an AC Split Unit with a KNX Thermostat

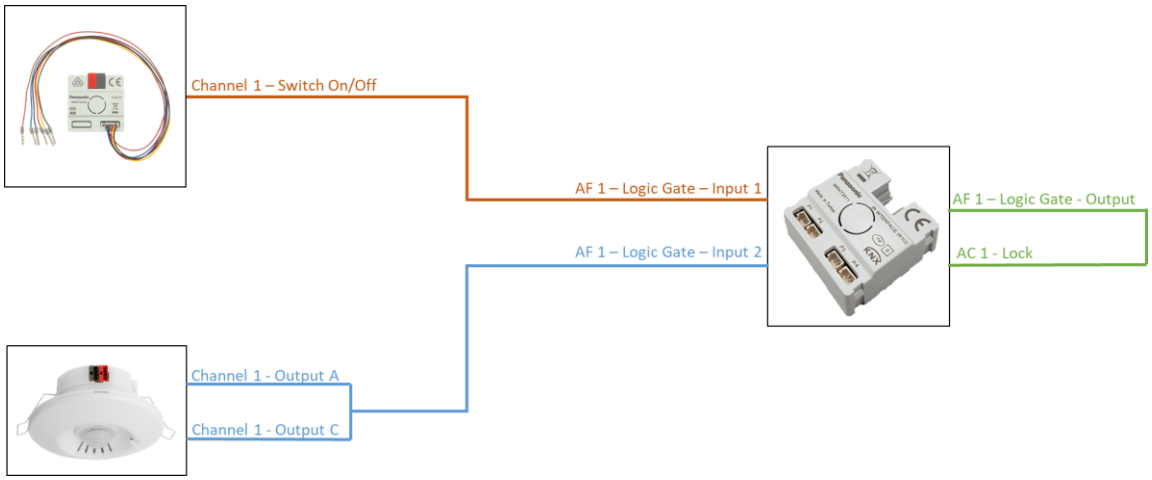
AC modules are used as an interface between the KNX system and AC split units from a wide range of manufacturers.
In this example, the thermostat function of the KNX Multi-Functional Switch is connected to an AC module from the KNX IR Interface so the user can control an AC split unit from KNX devices. The controlled AC unit has heating and cooling operation modes, 3 fan levels and 5 vertical vane positions.

Used devices	KNX IR Interface IR100 (WRKT26115NC) with an IR transmitter connected to port 1 KNX Multi-Functional Switch MS104 (WRKT62145FA)
Objects linking	
KNX IR Interface parameters	• Used AC module number: 1 • AC 1 – General Settings – Send IR codes to port 1: Yes • AC 1 – Operation Modes Settings – Enable simplified operation mode object: Yes • AC 1 – Fan Settings – Fan level object type: 1-byte, 0-255 • AC 1 – Fan Settings – Activate fan auto with fan level object: No • AC 1 – Fan Settings – Fan level 1 value: 1 • AC 1 – Fan Settings – Fan level 2 value: 2 • AC 1 – Fan Settings – Fan level 3 value: 3 • AC 1 – Fan Settings – Enable auto-manual switchover object: Yes • AC 1 – Fan Settings – Fan auto-manual switchover object value: On=auto, Off=manual • AC 1 – Vertical Vane Settings – Vane position object type: 1-byte, 0-255 • AC 1 – Vertical Vane Settings – Activate vane swinging with vane position object: No • AC 1 – Vertical Vane Settings – Vane position 1 value: 1 • AC 1 – Vertical Vane Settings – Vane position 2 value: 2 • AC 1 – Vertical Vane Settings – Vane position 3 value: 3 • AC 1 – Vertical Vane Settings – Vane position 4 value: 4 • AC 1 – Vertical Vane Settings – Vane position 5 value: 5 • AC 1 – Vertical Vane Settings – Enable vane swing on/off object: Yes • AC 1 – Vertical Vane Settings – Vane swing on/off object value: 1=swing on, 0=swing off

KNX Multi-Functional Switch parameters	• Thermostat – Controller General – Temperature control function: Heating and cooling • Thermostat – Controller General – Switching mode between heating and cooling: Automatically • Thermostat – Controller General – Switchover object value: Heating = 1, Cooling = 0 • Thermostat – Heating Control – Fan control: Enable • Thermostat – Cooling Control – Fan control: Enable • Thermostat – Setpoints – Sending current setpoint cyclically: Not cyclically, only when changed • Thermostat – Setpoints – Manual setpoint setting step: 1K • Thermostat – Fan Control – Number of fan level: 3 levels • Thermostat – Fan Control – Sending fan level cyclically: Not cyclically, only when changed • Thermostat – Fan Control – Fan object type: 1-Byte object, configurable values 0-255 • Thermostat – Fan Control – Fan level 1 custom value: 1 • Thermostat – Fan Control – Fan level 2 custom value: 2 • Thermostat – Fan Control – Fan level 3 custom value: 3 • Thermostat – Fan Control – Fan level auto custom: No • Thermostat – Fan Control – Fan switchover auto to manual: Auto enable is 1, auto disable is 0 • Thermostat – Fan Control – Swing function: Enable • Thermostat – Fan Control – Flap position 1 custom value: 1 • Thermostat – Fan Control – Flap position 2 custom value: 2 • Thermostat – Fan Control – Flap position 3 custom value: 3 • Thermostat – Fan Control – Flap position 4 custom value: 4 • Thermostat – Fan Control – Flap position 5 custom value: 5 • Thermostat – Fan Heating – Fan auto: Enable • Thermostat – Fan Heating – Returning to auto fan: Disable • Thermostat – Fan Cooling – Fan auto: Enable • Thermostat – Fan Cooing – Returning to auto fan: Disable
	The unmentioned parameters can be the default or user defined parameters
Notes	AC remote configuration must be written to the device using the DCA before setting parameters and linking objects In this example, the KNX IR Interface should be controlled through KNX Multi-Functional Switch only. To avoid turning the AC unit off then on sequentially when the Multi-Functional Switch switchover between heating and cooling modes, an auxiliary function as an OR logic gate with output delay can be used.

11.6 Switching an AC Split Unit Off According to a Window Contact and Occupancy Status

In order to save energy, the KNX IR Interface can be programmed to switch an AC split unit off and lock it when a window is opened or in case of absence of the movement in the room. When the window is closed again or in case of presence of a movement in the room, the AC split unit is switched to its old state. A delay time (2 minutes) is set before switching the AC split unit off and before returning to the old state.

Used devices	KNX IR Interface IR100 (WRKT26115NC) with an IR transmitter connected to port 1 KNX 2 CH I/O Interface (WRKT2402XXX) with a window contact connected to channel 1. KNX Ceiling Type Presence Detector (WRKT32005NC)
Objects linking	
KNX IR Interface parameters	• AC 1 – Lock Settings – Enable lock function: Yes • AC 1 – Lock Settings – Lock status after bus return: unlocked • AC 1 – Lock Settings – AC state when lock enabled: User-defined • AC 1 – Lock Settings – AC state when lock enabled – Split unit power: off • AC 1 – Lock Settings – AC state when lock disabled: As before lock • Auxiliary Functions – Auxiliary function 1: Logic gate • AF 1 – Auxiliary function name: Lock AC module when window is not closed or when no presence • AF 1 – Logic gate type: OR • AF 1 – Number of used inputs: 2 • AF 1 – Input 1 polarity: Inverted • AF 1 – Input 1 value after bus return: Read from bus • AF 1 – Input 2 polarity: Inverted • AF 1 – Input 2 value after bus return: 1 • AF 1 – Output object type: 1-bit • AF 1 – Output value for true: 1 • AF 1 – Output value for false: 0 • AF 1 – Send output after delay: Yes • AF 1 – Delay time: 00:02:00

KNX 2 CH I/O Interface	• Channel type – Channel 1 is: Input • Channel 1 (Input) – Function of the channel: Switching • Channel 1 (Input) – Reaction by closing the contact: On • Channel 1 (Input) – Reaction by opening the contact: Off • Channel 1 (Input) – Send telegram cyclically: No • Channel 1 (Input) – Reaction when restoring the bus supply: Update with current state
KNX Ceiling type presence detector parameters	• Channel 1: Presence detection • Channel 1 – Control depends on: Presence only • Channel 1 – Begin of control send (A): Switch (On) • Channel 1 – End of control send (C): Switch (Off)
	The unmentioned parameters can be the default or user defined parameters
Notes	AC remote configuration must be written to the device using the DCA before setting parameters and linking objects