



Gateway overview and operation

Content

■ **1. General information, gateway properties and possible use-cases**

■ **2. Operators instructions I – Configuration**

■ **3. Operators instructions II – Initial setup / Flashing the firmware**

■ **4. Developers instructions – Development / Adjusting the firmware**

General information

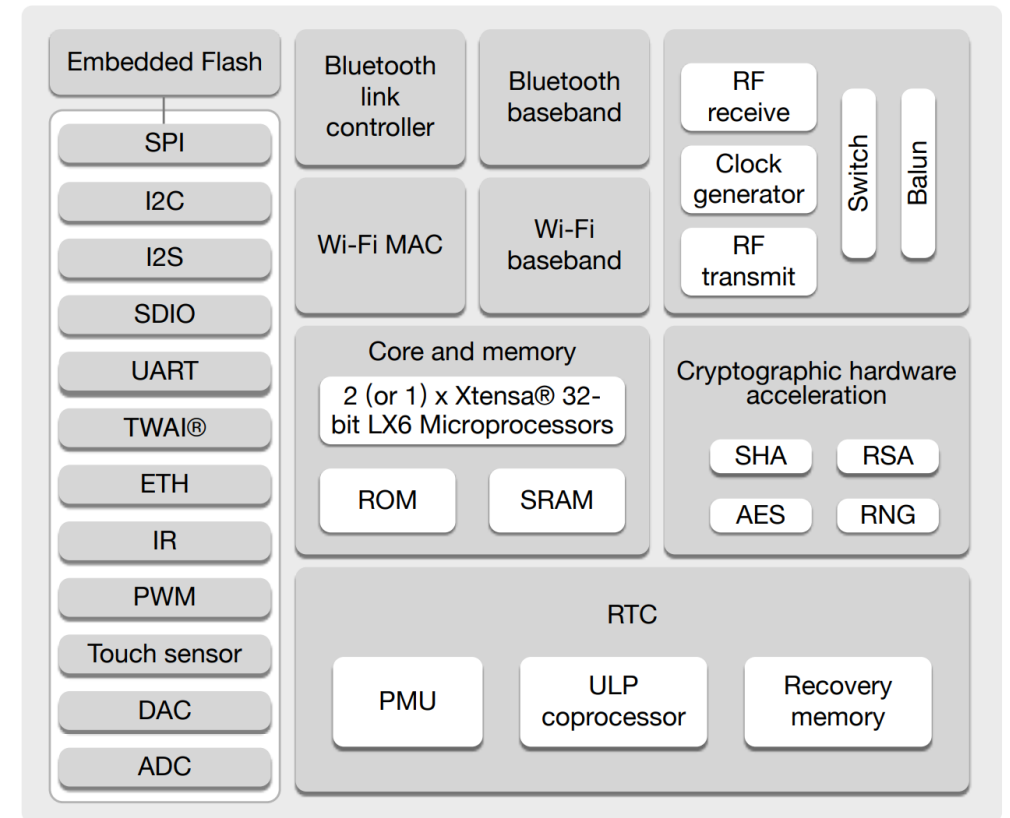
Microcontroller

■ Core component is an ESP32-WROOM32D

- ≡ 32-bit, dual-core 40 MHz
- ≡ 16MB flash storage
- ≡ 34 programmable GPIOs
- ≡ 2,4 GHz WiFi
- ≡ Bluetooth, Bluetooth low energy

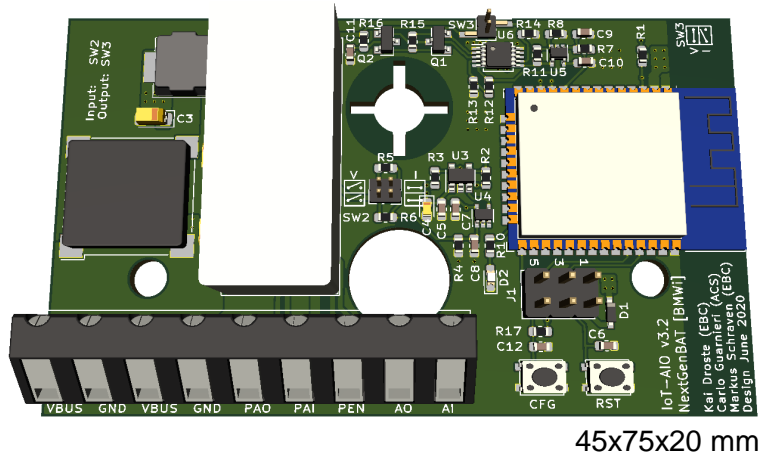
■ Programming languages

- ≡ C (ESP-IDF)
- ≡ C++ (Arduino IDE)
- ≡ Micropython



Data sheet: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf

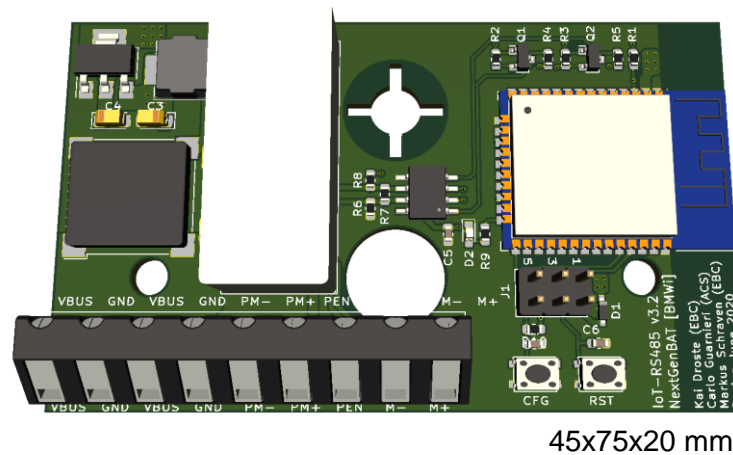
Gateway overview



Analog

- 0(2) – 10 V
- 0(4) – 20 mA
- 1x Input, 1x Output (single-ended)
- MCP3021, XTR111

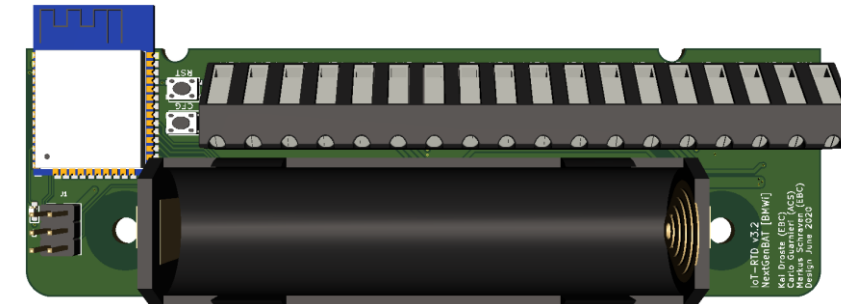
Price: 36,07 €/pcs.



Digital

- RS485
 - Modbus RTU
 - (BACnet MS/TP, not yet implemented)
- Communication with one field device (Software)
- MAX13487

31,67 €/pcs.



45x120x17 mm

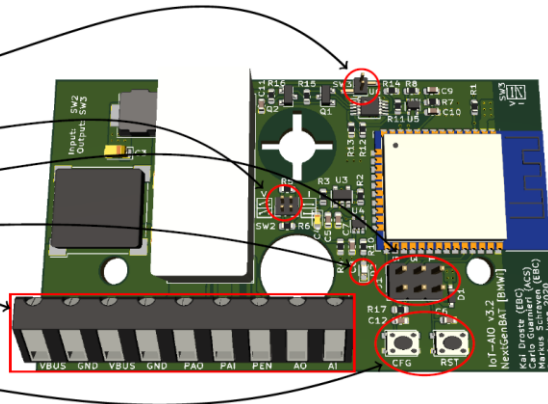
Temperature

- RTD-probe
 - ≡ 2,3,4 wires
- Optional Battery operated (LiFePO4, 18650 cells)
- 24 VDC add-on supply
- 2 RTD-probes
- MAX31865

53,32 €/pcs.

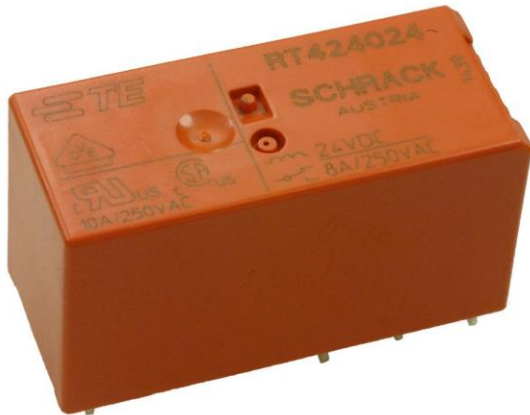
Analog Gateway

- Input source header (SW3)
- Output source header (SW2)
- Programming interface
- LED
- Terminal
- Reset, Config Button



VBUS	+ 24 V	Input
GND	GND	Input
VBUS	+ 24	Output to field device
GND	GND	Output to field device
PAO	AO to plc	Not in use
PAI	AI to plc	Not in use
PEN	+ 24 V	Enable Gateway (necessary)
AO	0 -10 V / 0 - 10 mA	Analog output (single-ended)
AI	0 -10 V / 0 - 10 mA	Analog input (single-ended)

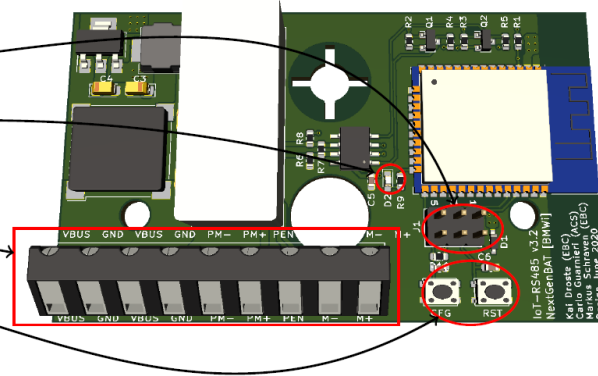
Pins labeled with „Not in use“ are used for the switching mechanism. More details can be found in [chapter 2 of the instruction manual](#) or slide 16 in this presentation.



- Relay
- Switching between internal and external signal path (“to PLC”), compare slide 16

Digital Gateway

- Programming interface
- LED
- Terminal
- Reset, Config Button



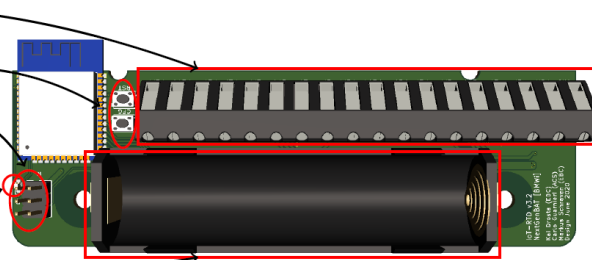
VBUS	+ 24 V	Input
GND	GND	Input
VBUS	+ 24	Output to device
GND	GND	Output to device
PM-	M- to plc	Not in use
PM+	M+ to plc	Not in use
PEN	+ 24 V	Enable Gateway (necessary)
M-	RS485 - Line	
M+	RS485 + Line	

Pins labeled with „Not in use“ are used for the switching mechanism. More details can be found in [chapter 2 of the instruction manual](#) or slide 16 in this presentation.

Temperature Gateway

1.2 Temperature Gateway

- Terminal
- Reset, Config Button
- Programming interface
- LED
- Battery tray



GND	Do not connect	Not in use
PEN	Do not connect	Not in use
1I-		RTD probe 1
1V-		RTD probe 1
1V+		RTD probe 1
1I+		RTD probe 1
P1I-	to plc	RTD probe 1 not in use
P1V-	to plc	RTD probe 1 not in use
P1V+	to plc	RTD probe 1 not in use
P1I+	to plc	RTD probe 1 not in use
2I-		RTD probe 2
2V-		RTD probe 2
2V+		RTD probe 2
2I+		RTD probe 2
P2I-	to plc	RTD probe 2 not in use
P2V-	to plc	RTD probe 2 not in use
P2V+	to plc	RTD probe 2 not in use
P2I+	to plc	RTD probe 2 not in use

Pins labeled with „Not in use“ are used for the switching mechanism. More details can be found in [chapter 2 of the instruction manual](#) or slide 16 in this presentation.

Configuration interface



- Web server is accessed via the IP address of the gateway, see slide 13
 - ≡ The IP address can be discovered using a BLE beacon
- Configuration exclusively via web server
- https://github.com/RWTH-EBC/OSIGBApp/blob/main/Firmware/doc/aio_configurator_request.html

IoT AIO Configurator

%s
%s

WiFi:

Access Point:

SSID: %s

PASS: %s

Station:

SSID: %s

PASS: %s

MQTT:

IoT Agent:

Crypt: None Host: %s Port: %d

User Name: %s Password: %s

Topics:

API-Key: %s

Device-ID: %s

Publish: Aedifion: [load_balance_grop/project_handle] %s QoS 0 [< 1]

Subscribe: Empty for Aedifion %s QoS 0 [< 1]

Encoding: JSON - {"key": "value"}

*TOPIC: /<api-key>/<device-id>/<mode>

Analog Gateway Configuration:

INPUT:

Phys: Disabled

Sensor: Name: %s Min: %2f Max: %2f

Mode: Periodic %2f

OUTPUT:

Phys: Disabled

Actuator: Name: %s Min: %2f Max: %2f

Digital Gateway Configuration

Device 1

Device Number (Modbus) %d

Status 1 Disable Name: %s Address: %d Value Correction: %d

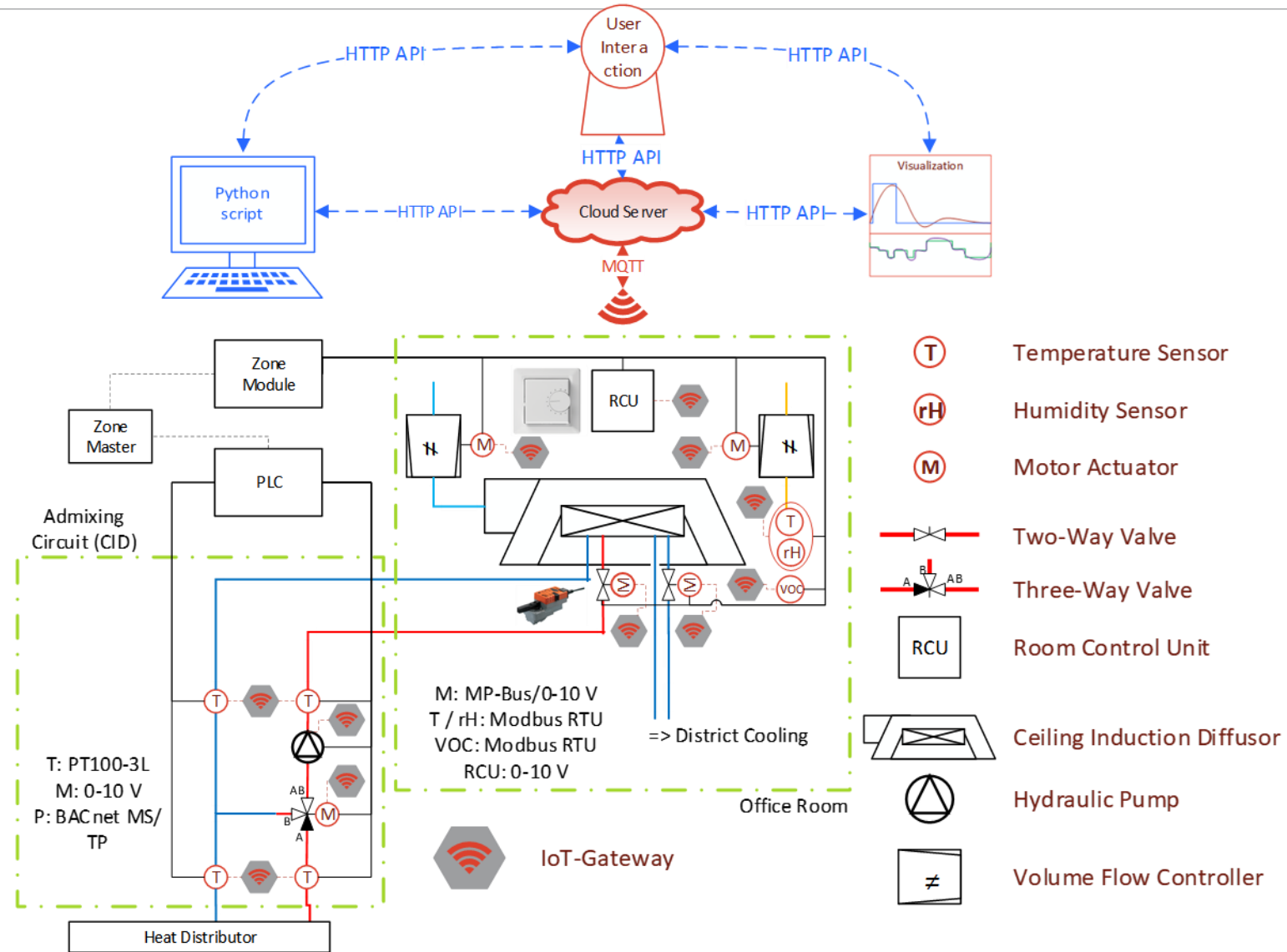
Command Sampling Rate [ms] / Value [-] %d Periodic %2f

Status 2 Disable Name: %s Address: %d Value Correction: %d

Command Sampling Rate [ms] / Value [-] %d Periodic (sam %2f

Submit

Example application



Further available information and repositories

- Firmware (Programmed in C using ESP-IDF)
 - ≡ [OSIGBApp/Firmware at main · RWTH-EBC/OSIGBApp \(github.com\)](#)
- Hardware (PCB design in KiCAD)
 - ≡ [OSIGBApp/Schematics at main · RWTH-EBC/OSIGBApp \(github.com\)](#)
- Instructions manual
 - ≡ [OSIGBApp/Documentation at main · RWTH-EBC/OSIGBApp \(github.com\)](#)

Operators instructions I - Configuration

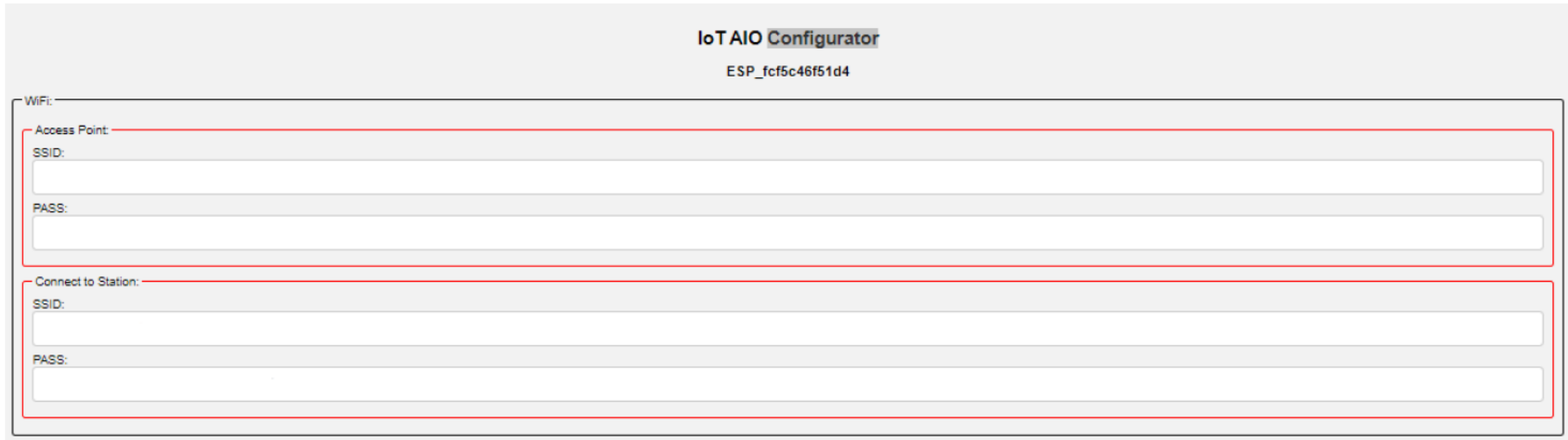
Accessing the web server interface

- Requirements: Firmware is already flashed, see operators instructions II (slides 26 onwards)

- In delivery state, the gateway is not able to connect to a WiFi network, directly
 - ≡ First, the CFG button has to be pressed between 1 s and 3 s to initialize the access point interface
 - = The AP will be displayed as ESPxxxx in the available WiFi connections (AP mode)
 - The gateway sets up an individual network. The default password is 123456789
 - = Connect to the WiFi AP with an end device of your choice (e. g. computer, smartphone, etc.)
 - = The web server interface can be accessed with a web browser and the IP address 192.168.4.1

- When already connected to a WiFi network
 - ≡ The web interface is now accessed via the IP of the gateway
 - ≡ For finding the IP, you can, for instance:
 - = Use BLE beacon => Press CFG button for < 1 s: LED blinks 2x
 - The IP address is provided as BLE-Beacon, which can be received with a Bluetooth-enabled device. e. g.:
 - Bluetooth Beacon Interactor (Windows Store)
 - eBeacon (IOS)
 - = Alternatively, checking IP in router config (might be incomprehensible with many gateways in the network)

WiFi configuration



The image shows a web-based configuration interface titled "IoT AIO Configurator" with the MAC address "ESP_fc5c46f51d4" displayed below the title. Under the "WiFi:" section, there are two main areas: "Access Point:" and "Connect to Station:". Each area contains input fields for "SSID:" and "PASS:". The "Access Point:" section is highlighted with a red border, and the "Connect to Station:" section is also highlighted with a red border.

■ Access Point

- ≡ Set name and password for the AP of the Gateway (e. g. credentials when activating the AP configuration)
- ≡ Should usually not be touched, as the gateway will most likely be operated connected to a network. Default credentials are SSID=MAC address of the ESP, and PW=12345679

■ Station

- ≡ SSID and password of the WiFi network the Gateway should be connected to.
 - = Note that it might not be possible to connect with special encrypted networks like WPA2 enterprise (Eduroam)

MQTT configuration

MQTT:

IoT Agent:

Crypt: Host: Port:

User Name: Password:

Topics:

API-Key:

Device-ID:

Publish: Aedifion: [load_balance_group/project_handle]
 QoS 0 [< 1] ▼

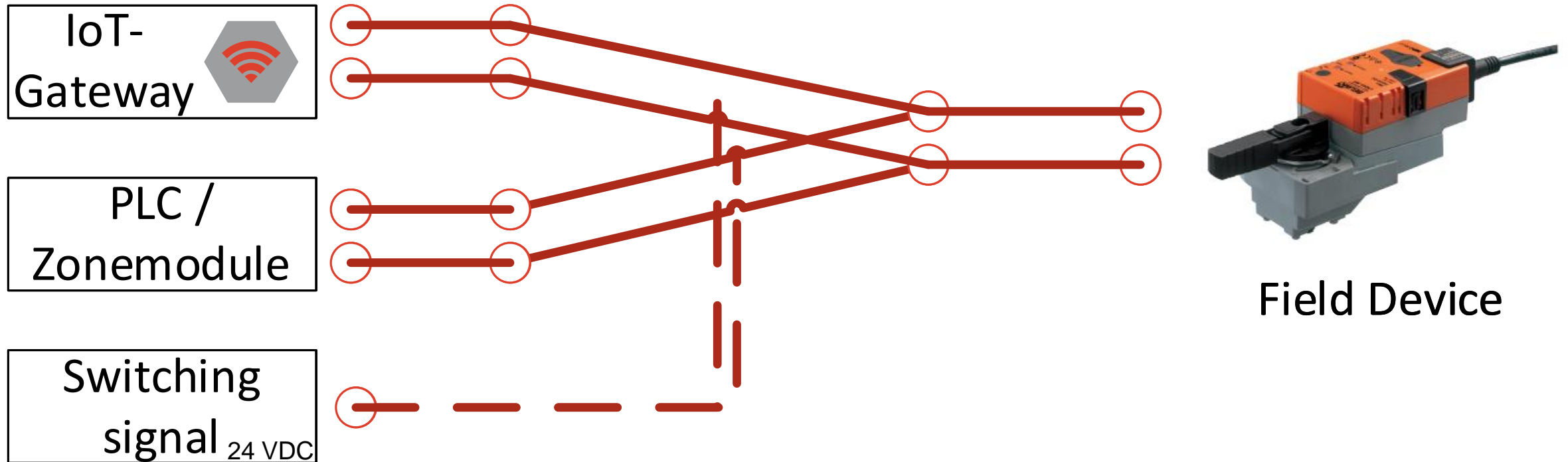
Subscribe: Empty for Aedifion
 QoS 0 [< 1] ▼

Encoding:
 ▼

*TOPIC: /<api-key>/<device-id>/<mode>

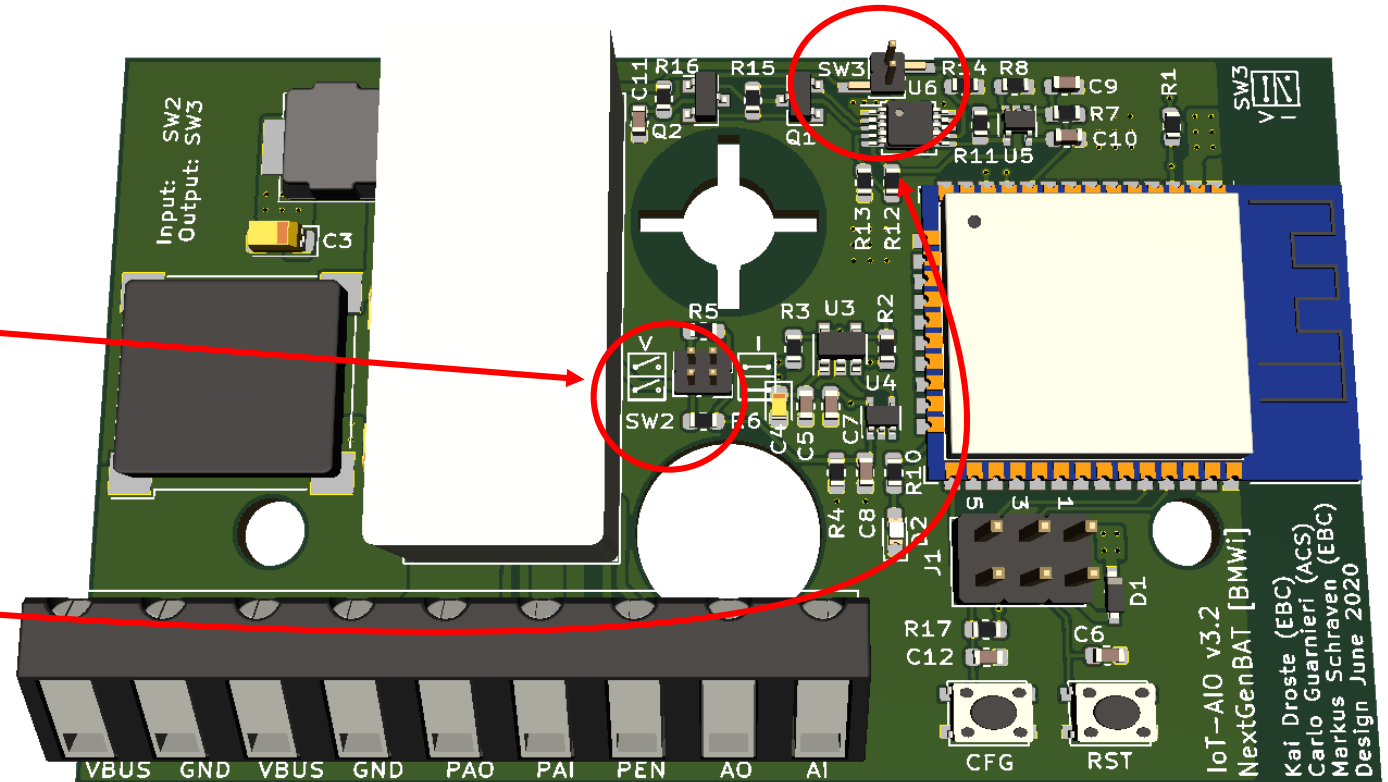
- Supports JSON and special encoding for aedifion and FIWARE platform
- A detailed manual is located in this same repository:
 - ≡ [OSIGBApp/Documentation at main · RWTH-EBC/OSIGBApp \(github.com\)](#)

Switching between SotA and IoT mode



Hardware configuration: Analog Gateway - Switching voltage and current signal mode

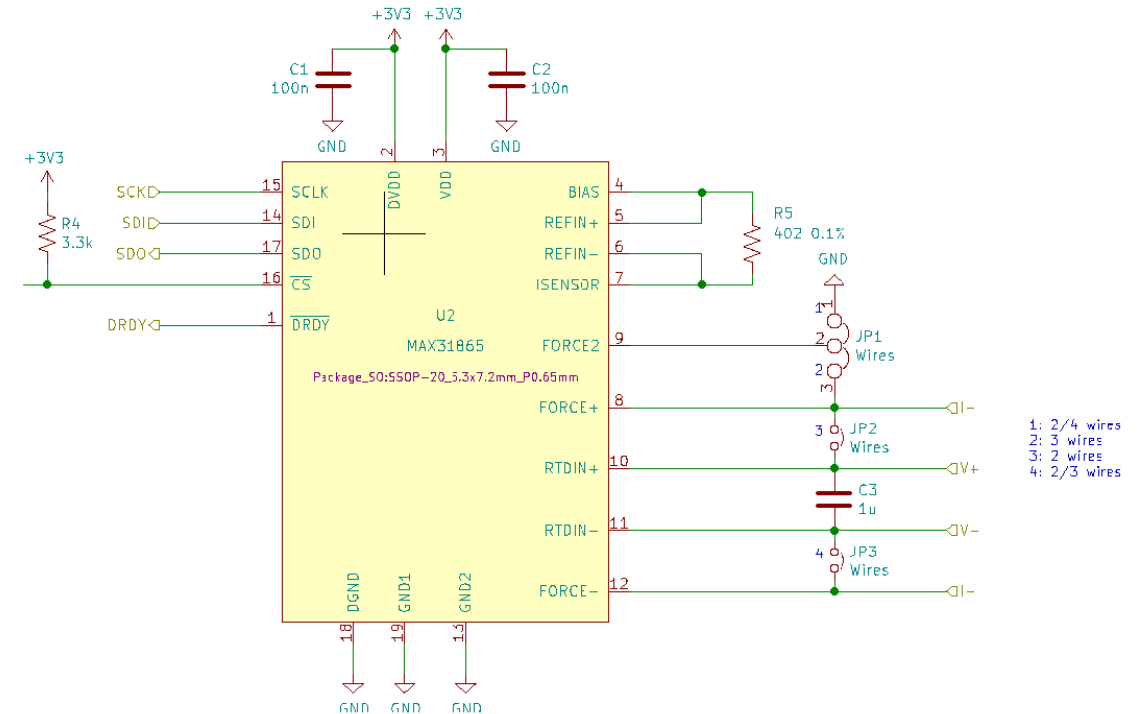
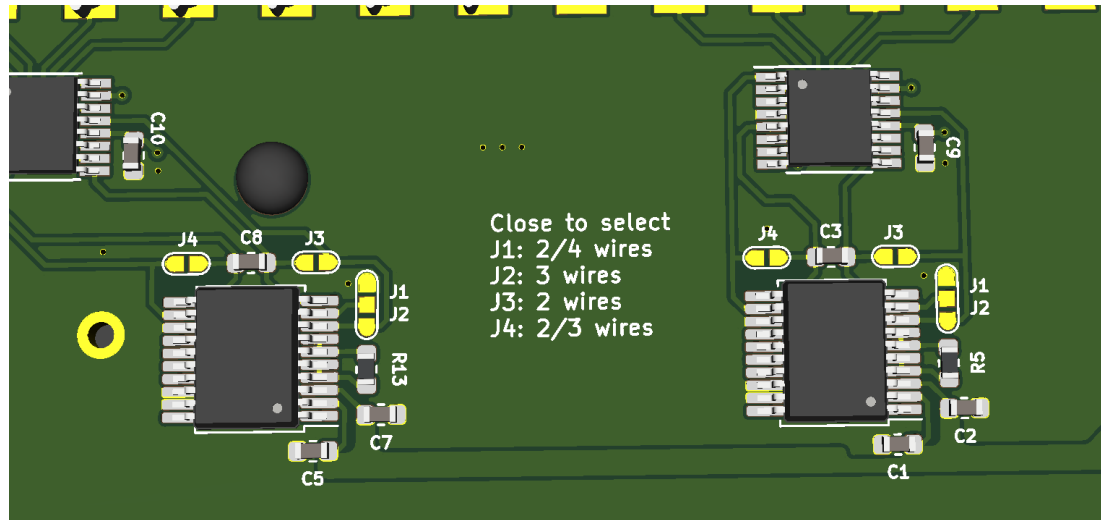
- 3x2 pin header jumper required
 - ≡ SW2: Output signal mode, requires 2 jumpers
 - ≡ SW3: Input signal mode, requires 1 jumper
- SW2: Output
 - ≡ Voltage mode: no jumpers attached
 - ≡ Current mode: both jumpers attached
- SW3: Input
 - ≡ Voltage mode: jumper attached
 - ≡ Current mode: no jumper attached



Hardware configuration: Temperature GW I - Connecting the RTD element

■ Connecting 2-, 3-, 4-wire RTD elements

≡ The MAX31865 (IC for the temperature measurement) has to be configured by soldering connections (J1-J4)



■ Terminal pin connections:

- ≡ For a 2-wire probe, connect V+ and V-
- ≡ For a 3-wire probe, connect I+, V+ and V-
- ≡ For a 4-wire probe, connect all four pins I+, V+, I- and V-

Datenblatt MAX31865: [MAX31865.pdf \(maximintegrated.com\)](https://www.maximintegrated.com/datasheet/MAX31865.pdf)

Hardware configuration: Temperature GW II - Adjusting the RTD type/reference resistor

- In order to connect a different type than a PT100 probe, e. g. a PT1000, the reference resistors R13 for channel 1 and R5 for channel 2 (see slide 19) have to be exchanged (soldering required).
 - ≡ The resistor should amount to 4 time the nominal value, e. g.
 - = PT100 requires a 400 Ω resistor
 - = PT1000 requires a 4 k Ω resistor
 - ≡ Additionally, in the configuration (web interface), the type/reference needs to be adjusted
 - = PT1000 is implemented as additional choice in the drop-down menu, different resistors require software adjustments

Datenblatt MAX31865: [MAX31865.pdf \(maximintegrated.com\)](https://www.maximintegrated.com/datasheets/datasheet.cfm?id=31865)

Hardware configuration: Temperature GW III - 24 VDC power supply

- Originally, the temperature Gateways have been designed for battery operation
 - ≡ While this seems practical, the fully charged battery (1000 mAh) lasted merely roughly 2 weeks significantly reducing the practical benefits
- In order to allow 24 VDC supply (same as the other Gateways) a Buck-Converter can be used (transforming 24 VDC to 3.3 VDC)
 - ≡ E. g. Buck-Converter: VXO7803-500 [VXO78-500 Series Datasheet - Non-Isolated | CUI Inc](#)
 - ≡ The Buck-Converter can be soldered directly to the positive and negative pole of the battery tray



units: mm [inch]
tolerance: $\pm 0.50 [\pm 0.020]$
pin diameter tolerance: $\pm 0.10 [\pm 0.004]$

PIN CONNECTIONS		
PIN	+OUTPUT	-OUTPUT
1	+VIN	+VIN
2	GND	-VOUT
3	+VOUT	GND

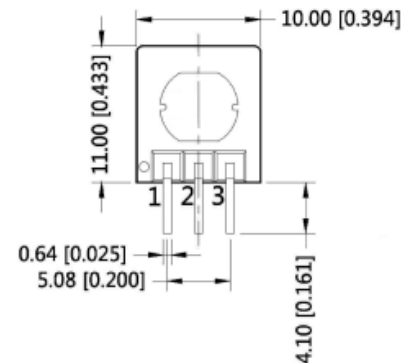
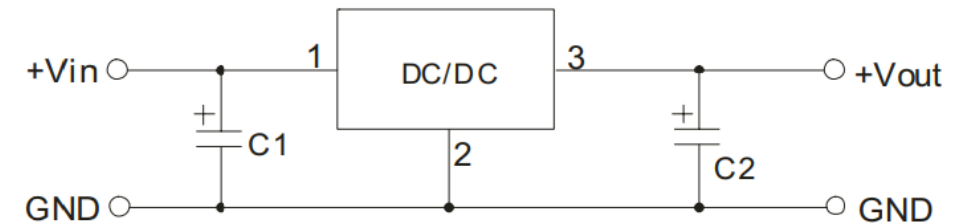


Figure 1

Positive Output Application Circuit



Model Number	C1, C3 (ceramic capacitor)	C2, C4 (ceramic capacitor)
VXO7803-500	10 μ F/50 V	22 μ F/10 V

Software configuration: Digital GW I - Configuration of Modbus coils and registers

Digital Gateway Configuration

Device 1

Device Number (Modbus): 1

Status 1: Read

Name: Belimo_out Address: 4 Value Correction: 100

Command: 3 Sampling: Periodic Rate [ms] / Value [-]: 1000

Status 2: Write

Name: Belimo_in Address: 0 Value Correction: 100

Command: 6 Sampling: Periodic (same rate as Register 1) Rate [ms] / Value [-]: 1000

- Device Number:
 - ≡ Device ID: Device address within the bus
- Status (read & write holding registers)
 - ≡ Read
 - ≡ Write
- Register (function code)
 - ≡ E.g. 3: Read holding register or 6: write single register
 - ≡ See table on the right
- Address:
 - ≡ Register address to be read or written

■ Typical function codes

Commonly used public function codes						
Code	Hex	Function	Type			
01	01	Read Coils	Single Bit Access	Data Access		
02	02	Read Discrete Inputs				
05	05	Write Single Coil				
15	0F	Write Multiple Coils				
03	03	Read Holding Registers	16 bit Access		Data Access	
04	04	Read Input Register				
06	06	Write Single Register				
16	10	Write Multiple Registers				
22	16	Mask Write Register				
23	17	Read/Write Multiple Registers				
24	18	Read FIFO queue	File record access			Data Access
20	14	Read File Record				
21	15	Write File Recore				
07	07	Read Exception Status				
08	08	Diagnostic	Diagnostics	Data Access		
11	0B	Get Com event counter				
12	0C	Get Com Event Log				
17	11	Report Server ID				

Software configuration: Digital GW II - Configuration of Modbus coils and registers

■ Example Modbus register for a volume flow control actuator

Modbus Register Description

No.	Adr.	Description Comment	Range / Enumeration	Unit	Scaling	Values Default	Access
1	0	Setpoint Setpoint for actuator between Min (Register 106) and Max (Register 107)	0...10'000	%	0.01	0	R/W
2	1	Override Control Overriding setpoint with defined values	0:None 1:Open 2:Close 3:Min 4:Mid 5:Max	-	-	0	R/W
3	2	Command Initiation of actuator functions for service and test. After command is sent, register returns to None(0). With Reset(4) all Malfunction and Service Information (Register 105) Information can be reset.	0:None 1:Adaption 2:Test 3:Sync 4:Reset	-	-	0	R/W
4	3	Actuator Type	0:Actuator not connected 1:Air/Water 2:VAV / EPIV 3:Fire 4:Energy Valve 5:6way EPIV	-	-	-	R
5	4	Relative Position	0...10'000	%	0.01	-	R
6	5	Absolute Position The unit depends on the device: [°] for actuators with rotary movement [mm] for actuators with linear movement	0...max angle/stroke	° mm	1 1	-	R
7	6	Relative volumetric flow Relative volumetric flow of Vnom	0...100	%	0.01	-	R

■ Typical function codes

Commonly used public function codes				
Code	Hex	Function	Type	
01	01	Read Coils	Single Bit Access	Data Access
02	02	Read Discrete Inputs		
05	05	Write Single Coil		
15	0F	Write Multiple Coils		
03	03	Read Holding Registers	16 bit Access	
04	04	Read Input Register		
06	06	Write Single Register		
16	10	Write Multiple Registers		
22	16	Mask Write Register		
23	17	Read/Write Multiple Registers		
24	18	Read FIFO queue		
20	14	Read File Record	File record access	
21	15	Write File Recore		
07	07	Read Exception Status	Diagnostics	
08	08	Diagnostic		
11	0B	Get Com event counter		
12	0C	Get Com Event Log		
17	11	Report Server ID		

Software configuration: Digital GW III - Configuration of Modbus coils and registers - Example

■ Example configuration of a Belimo LM24A-MOD

Digital Gateway Configuration

Device 1

Device Number (Modbus)

1

Status 1

Read

Status 2

Write

Name: Belimo_out Address: 4 Value Correction: 100

Command: 3 Sampling: Periodic Rate [ms] / Value [-]: 1000

Name: Belimo_in Address: 0 Value Correction: 100

Command: 6 Sampling: Periodic (same rate as Register 1) Rate [ms] / Value [-]: 1000

Operation	No.	Address	Register	Access
	1	0	Setpoint [%]	R / W
	2	1	Override control	R / W
	3	2	Command	R / W
	4	3	Actuator type	R
	5	4	Relative position [%]	R
	6	5	Absolute position [°] [mm]	R
	7	6	Relative volumetric flow [%]	R
	8	7	Absolute volumetric flow [m³/h]	R
	9	8	Sensor value 1 [mV] [-]	R
	10	9	—	—
	11	10	Absolute volumetric flow in unit selected	LowWord
	12	11		HighWord
	13	12	Setpoint analog [%]	R

Belimo Modbus registers

Functionalities implemented to the CFG button

Time [sec]	Function	Reaction(LED)
<1	Eddystone Beacon enable config mode	- -
>1 <3	Enable Access Point (works only if not connected to station)
>3 <7	Starts OTA update	Finished: - - - Fail : ...
>7	Reset to default settings	

- = Long flash of the LED (1sec)

. = Short flash of the LED (<0.5 sec)

Operators instructions II - Initial setup / Flashing the firmware

Operators instructions II - Initial setup / Flashing the firmware - Menuconfig

- For the firmware to be flashed, it has to be compiled first on your computer
- Start a terminal (CMD/PowerShell)
- cd into the repository of the Gateway software
- Start the ESP-IDF
 - ≡ This obviously requires the ESP-IDF to be installed and configured
 - ≡ For further information on installing and configuring the ESP-IDF, see <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html>
<https://docs.espressif.com/projects/esp-idf/en/latest/esp32s2/get-started/#step-4-set-up-the-environment-variables>
 - ≡ Alternatively, one can use the preconfigured terminal from the ESP-IDF and navigate into the repository from there

```
kdros@T14 ~\AppData\Local\Microsoft\WindowsApps
> cd C:\git\iotdevelopment\software\software.c\
kdros@T14 C:\git\iotdevelopment\software\software.c master
> C:\git\esp-idf\export.ps1
Setting IDF_PATH: C:\git\esp-idf
Adding ESP-IDF tools to PATH...
```

Name

esp-idf
iotdevelopment

Operators instructions II - Initial setup / Flashing the firmware - Menuconfig

- For practicality, the repository already contains 3 different configurations (analog, temperature, digital)
 - ≡ sdkconfig.aio
 - ≡ sdkconfig.tmp
 - ≡ sdkconfig.dio
- To choose a configuration, copy and rename one of them to sdkconfig
- Subsequently, the configuration should be validated in the menuconfig

```
kdros@T14 C:\git\iotdevelopment\software\software.c master  
> Copy-Item .\sdkconfig.aio sdkconfig
```

```
kdros@T14 C:\git\iotdevelopment\software\software.c master  
> idf.py menuconfig
```

```
SDK tool configuration --->  
Build type --->  
Application manager --->  
Bootloader config --->  
Security features --->  
Serial flasher config --->  
Partition Table --->  
IoT Gateway Interface --->  
Compiler options --->  
Component config --->  
Compatibility options --->
```

```
Select the target interface (Analog) --->  
Analog Configuration ----  
Optional Config --->  
WiFi Configuration ----
```

In this example, the analog configuration has been copied → the menuconfig shows „analog“ as selected interface → configuration is correct
The menu can be closed hitting the ESC key

- Leave the menuconfig hitting the ESC key

Operators instructions II - Initial setup / Flashing the firmware - Build / Connecting the programmer

Disclaimer: All commands shown here are executed in Windows Powershell or Windows Terminal

■ The firmware can now be compiled with the selected configuration

≡ idf.py build

```
kdros@T14 C:\git\iotdevelopment\software\software.c master  
> idf.py build
```

■ Programmer

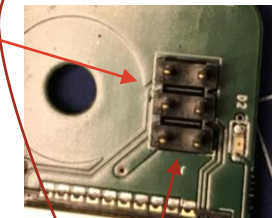
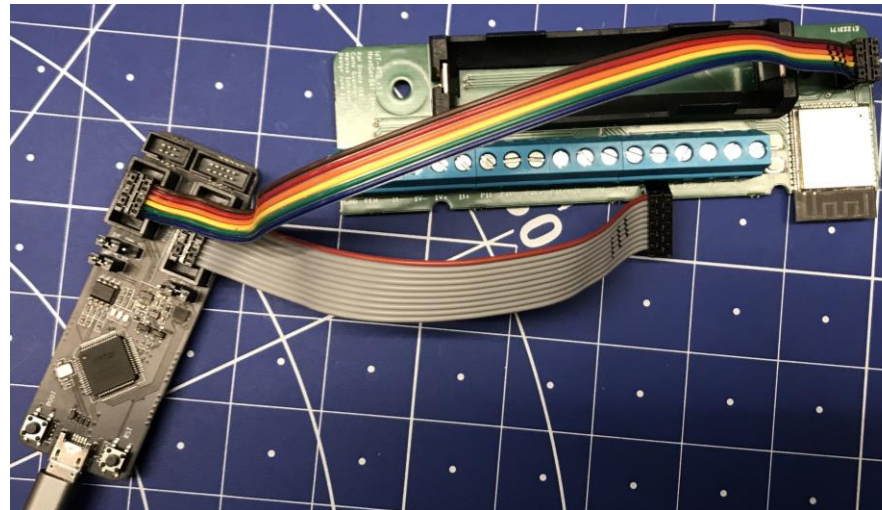
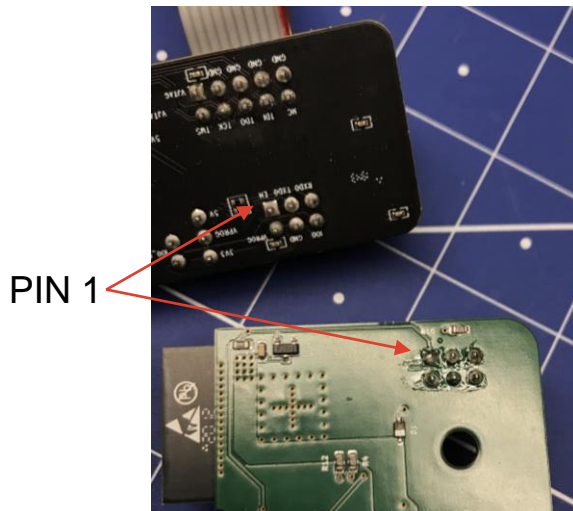
≡ 6-pole cable → used for flashing the firmware

≡ 8-pole cable → JTAG (can be used for an inline debugging of the ESP)

≡ The 6-pole cable has to be connected in a way, that it matches the Pin assignment on the Gateway!

= E. g., compare the PCB solder point: The squared solder point marks PIN 1 on each device

= From top perspective, PIN 1 is observed by an interruption of the white frame



Operators instructions II - Initial setup / Flashing the firmware - Flash, Monitor

■ Flashing the firmware to the Gateway

- ≡ idf.py flash

```
kdros@T14 > C:\git\iotdevelopment\software\software.c master  
> idf.py flash
```

■ Serial monitor (for debugging and checking the version/firmware)

- ≡ idf.py monitor
- ≡ Basically a command line printing info, debug and error messages
- ≡ The debug level can be adjusted in the menuconfig (idf.py menuconfig)
- ≡ Might be necessary to specify the port by adding -p com6 (e. g. for serial port COM6)

```
kdros@T14 > C:\git\iotdevelopment\software\software.c master  
> idf.py monitor
```

■ Commands can be combined / executed one after the other in one command (flash also incorporates the build command)

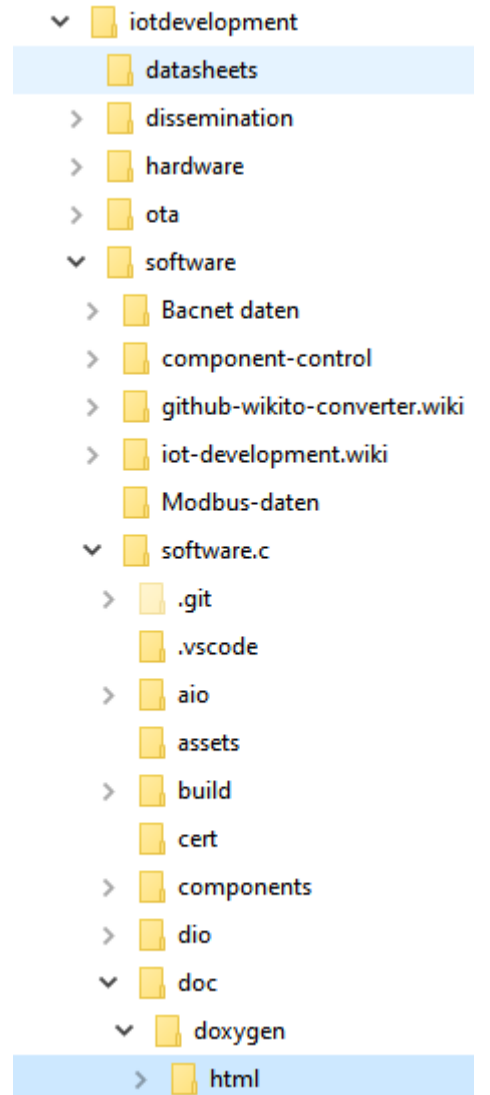
- ≡ E. g., idf.py flash monitor
- ≡ Or idf.py menuconfig flash monitor

```
kdros@T14 > C:\git\iotdevelopment\software\software.c master  
> idf.py flash monitor
```

Developers instructions - Development / Adjusting the firmware

Developers instructions - Development / Adjusting the firmware

- The header files (*.h) are, for the most part, commented within the code
 - ≡ From these, a Doxygen documentation has been created
 - = It is accessed via the index.html in [OSIGBApp/Firmware/doc/doxygen/html](#) at main · RWTH-EBC/OSIGBApp (github.com)
 - ≡ In VSCode, you can access the related header file to a function by marking the function and pressing F12
 - = This may help to better understand the origin and operating principle of the selected function
- For adding new variables that are coupled to the configuration (web interface), please consider the steps documented at [OSIGBApp/aio_configurator.md](#) at main · RWTH-EBC/OSIGBApp (github.com) The process is quite tedious (currently requires multiple conversions and a specific order) and will be revised in future development



VSCode

- If you are in a terminal (powershell), VSCode can be started entering „code .“ (will be executed in the current working directory / same directory as the terminal)

- VSCode overview:

- ≡ File explorer

- ≡ Git integration

- = Current branch

- ≡ ESP-IDF integration

- = VSCode integrated ESP-IDF tasks (e. g. flash, build, ...)

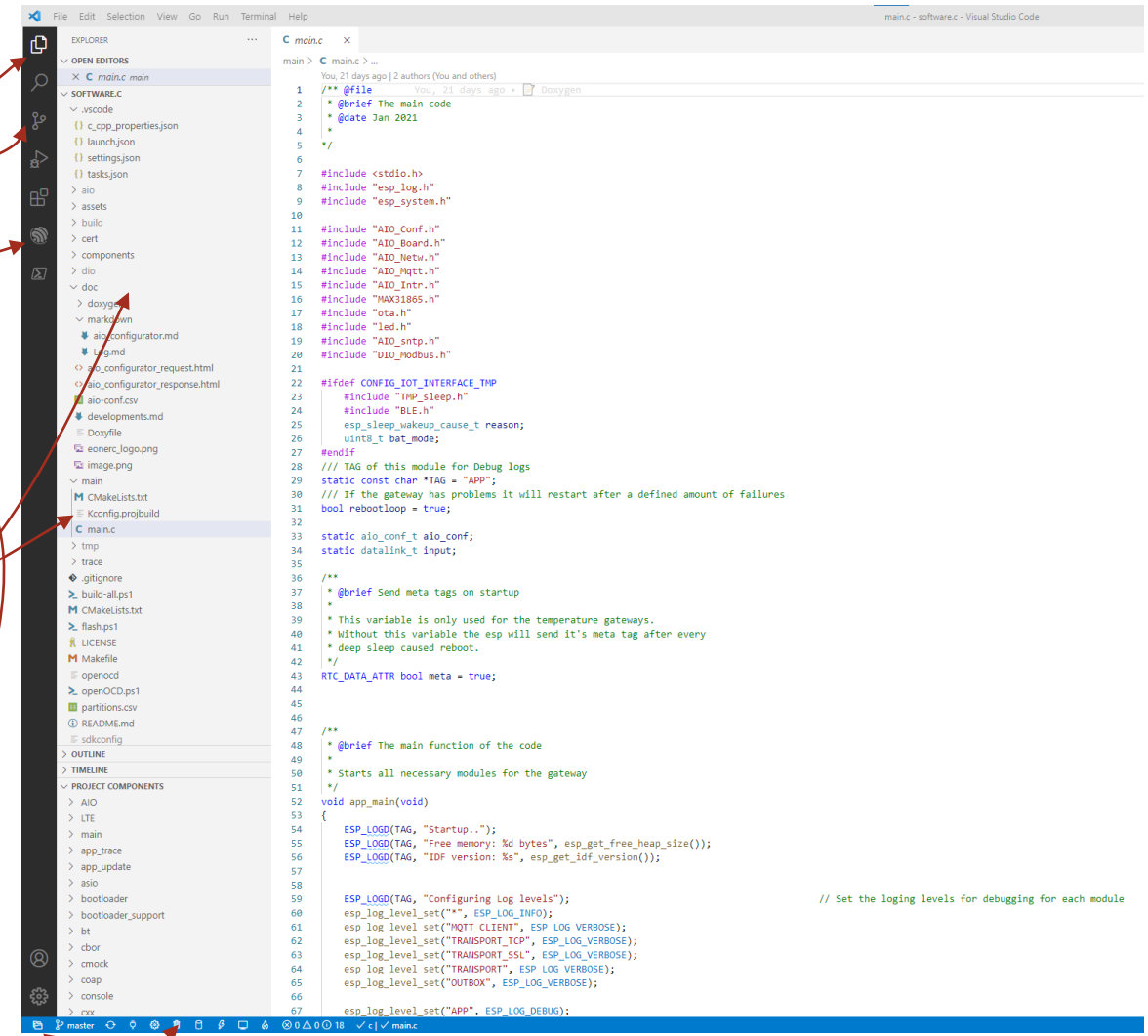
- This feature requires the ESP-IDF AddOn to be installed to VSCode

- ≡ Main application main.c

- = GW interface-related code in components

- ≡ The folders „aio“, „dio“, „tmp“, „build“ contain compiled files

- = They are generated by idf.build

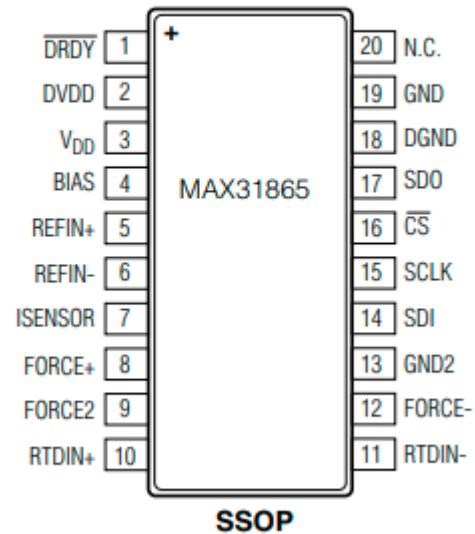


Measurement Accuracy I

■ Max31865 RTD to digital converter

- ≡ 15 Bit ADC resolution
 - = 0.03125 °C
- ≡ → 0.5 °C
- ≡ 21ms (max) conversion Time

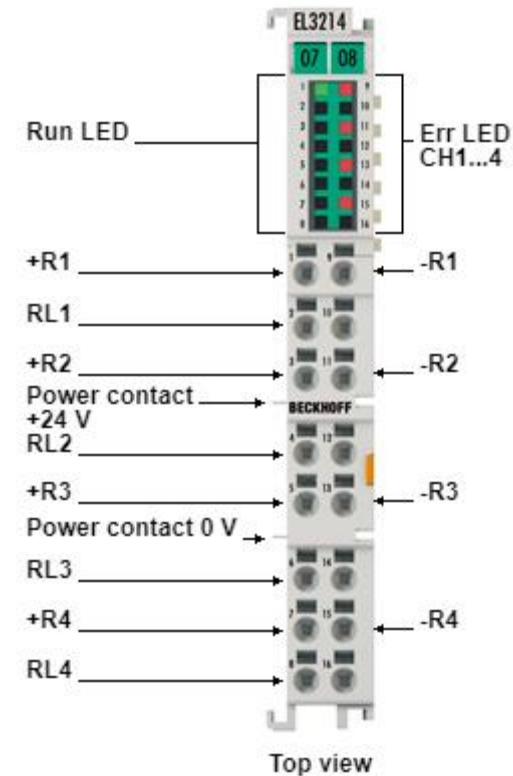
TOP VIEW



<https://datasheets.maximintegrated.com/en/ds/MAX31865.pdf>

■ EL3214 | 4-channel input terminal PT100 (RTD)

- ≡ 0.1 °C resolution
- ≡ 170 ms sampling rate

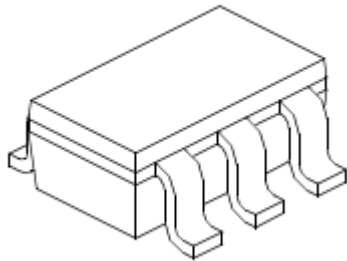


<https://www.beckhoff.de/default.asp?ethernetcat/el3214.htm>

Measurement Accuracy II

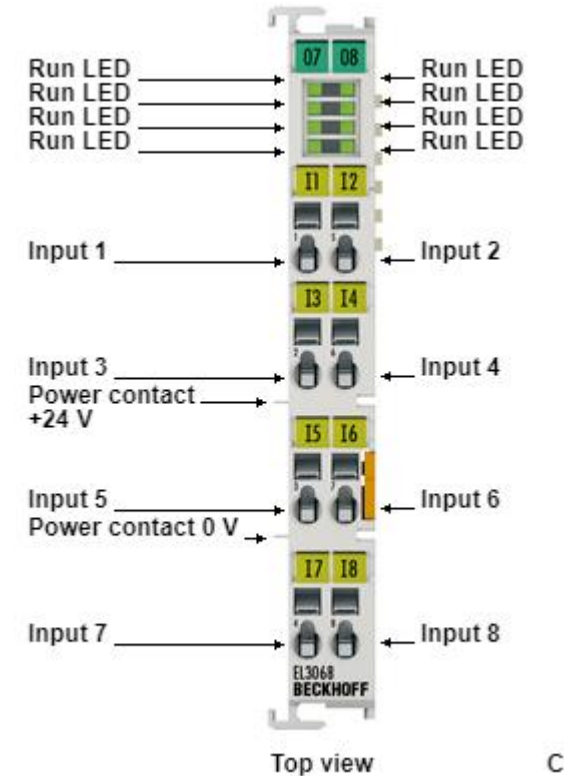
■ MCP3021 analog to digital converter

≡ 10 Bit resolution



■ EL3068 | 8-channel analog input terminal 0...10 V

≡ 12 Bit resolution



<https://eu.mouser.com/datasheet/2/268/21805a-74229.pdf>

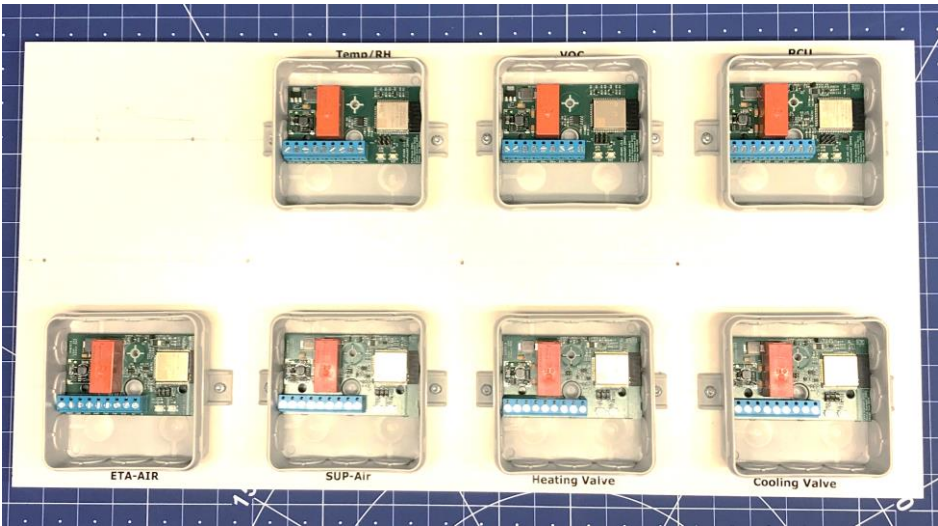
<https://www.beckhoff.de/default.asp?ethercat/el3068.htm>

Power consumption

- Digital GW
 - ≡ Standby 20 mA, 24 V, 0,48 W
- Temp GW
 - ≡ Standby 10-20 mA, 24 V, 0,24-0,48 W
- Analog GW
 - ≡ Standby 20 mA, 24 V, 0,48 W

Versorgungsspannung	24 V AC ± 15 %
Anschlussleistung	2 VA ohne Peripherie
Betriebstemperatur	0 – 50 °C
Zulässige Luftfeuchte	10 – 90 % r.F., nicht kondensierend
Schutzklasse	III (Schutzkleinspannung)
Schutzgrad	IP 20
EG-Konformität	EMV nach 2014/30/EU, ROHS 2011/65/EU
Einbauort	Schaltschrank, Wand oder Decke
Befestigung	Schraub- oder Hutschienebefestigung
Abmessungen	156 × 90 × 45 mm
Gewicht	270 g

<https://www.trox.de/zonenmodule/x-air-zmo-f9b1738e5431ff98>





Contact

E.ON Energy Research Center
Mathieustraße 10
52074 Aachen
Germany