MODBUS/TCP and Beckhoff controllers

The functionality of the **DOMIQ/Base** in software version 1.7.5 has been updated with support of the MODBUS/TCP and MODBUS/UDP protocols. The **Base** module plays the role of the master device. It enables integration with many commonly used automation devices like PLCs or usage of the protocol converters from MODBUS/TCP to MODBUS RTU using a standard Ethernet network.

In this tutorial you can find:

- basic information about the MODBUS/TCP protocol
- configuration description of the **Base** module in order to integrate it using MODBUS/TCP
- integration description of the **Base** module with a modular **Beckhoff** PLC.

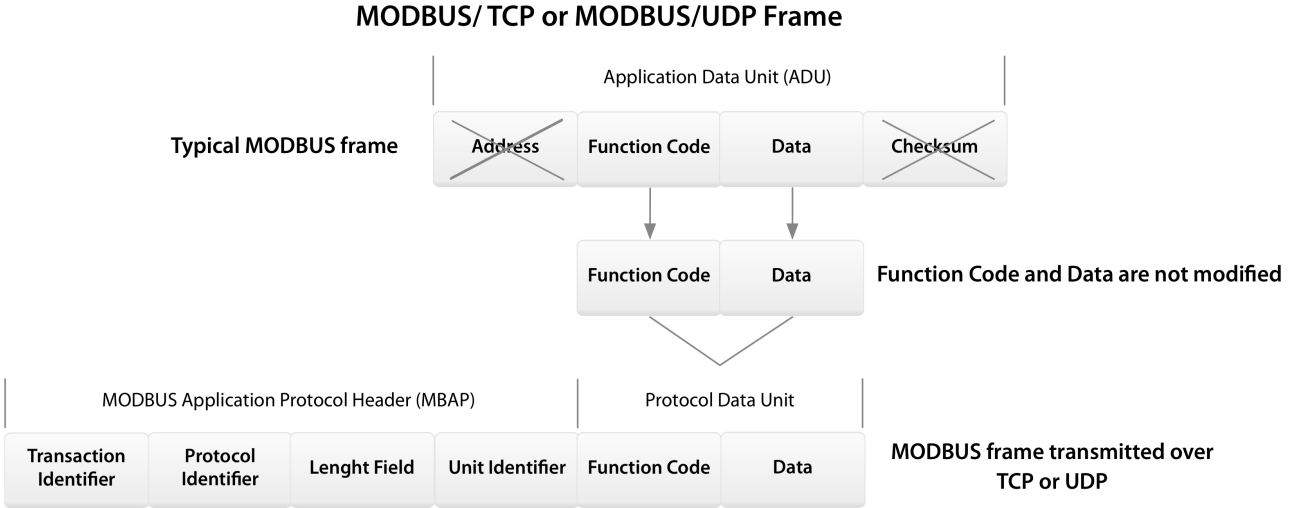
1. MODBUS/TCP Protocol

MODBUS/TCP is a modification of a standard MODBUS RTU protocol, in which the transport layer based on RS-485 was replaced by the TCP/IP protocol. Some devices use a non-standard variant, in which the connectionless UDP protocol was used instead of TCP.

TCP and IP protocols play only the transport role - they enable the exchange of data between devices. The exchanged data are interpreted exactly the same, as in the case of MODBUS RTU, including the requirement of a single response after each query.

The main task of the TCP protocol is to monitor that all data packets are transmitted correctly between devices in the order of sending, while the IP protocol (Internet Protocol) is responsible for sending the data to the recipients through many indirect devices-routers.

The encapsulation of the MODBUS packet was shown in the below scheme:



MODBUS/TCP and MODBUS/UDP packets are sent to the port 502, which is reserved for this purpose.

MODBUS/TCP has several features which surpass the MODBUS communication using the serial port:

- easy connection of many controllers; any distance
- faster data transmission
- using the already existing network structure (routers, switches, wiring etc.)
- reliability of the data transmission thanks to correctness control mechanisms in the TCP protocol and automatic repetitions in case of errors or lost packets. This function is not available in the UDP variant.

2. Configuration of the Base module

In this chapter we present how to prepare the **Base** module for the communication using MODBUS/TCP.

1. Choose the **MODBUS** tab.
2. Add a new interface and click on it in order to fill in its properties.
   - Enter the name (without spaces). The name of the interface is used while creating the names of identifiers (see below).
   - Set *TCP* as **Type**.
   - In the **IP Address** field enter the IP address of the slave device.
   - The **Port** field is filled in by default with the value of **502**. However there are devices which use another TCP port. In such case enter the port number in this field.
3. Add a new device:
   - Enter the name (without spaces). The name of the device is used while creating the names of identifiers (see below).
   - In the **Description** field you can add any description of the device.
   - In the **Address** field enter the address of the slave device.
4. Add the registers that you want to read/save and then set their parameters: type, address and name.

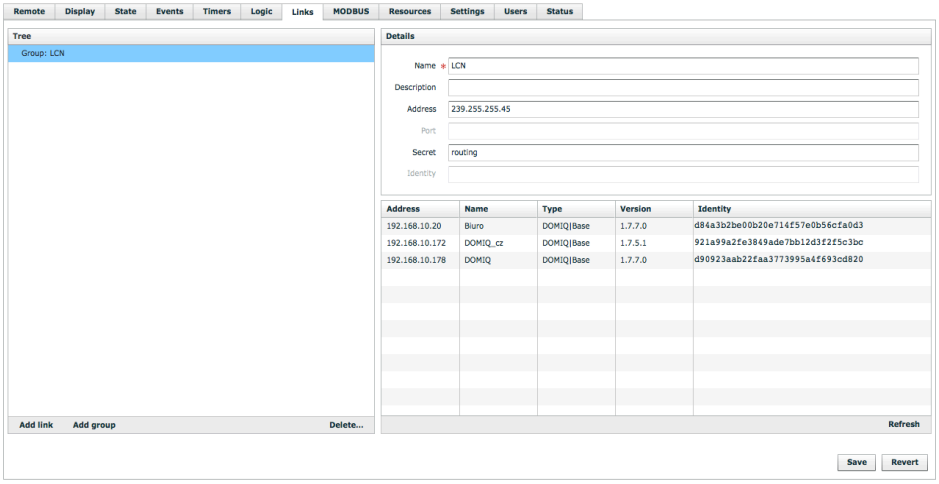
If the devices are connected and set correctly, then the value of the read/saved registers are displayed in the **State** tab.

The identifiers have the following syntax:

`MODBUS.<interface>.<device>.<register>`, e.g. *MODBUS.bk.dev.temp*.

The values of the MODBUS registers can be displayed on the visualizations or **Remote** application and connected with events, timers etc.

The example configuration used in the integration of the **Base** module with the **Beckhoff** controller was presented in the picture below:



3. Integration with the modular Beckhoff controllers

MODBUS/TCP protocol enables integration of the **DOMIQ** system with any device which supports this protocol. One of the available options is the integration of the **DOMIQ** system with the modular **Beckhoff** controllers.

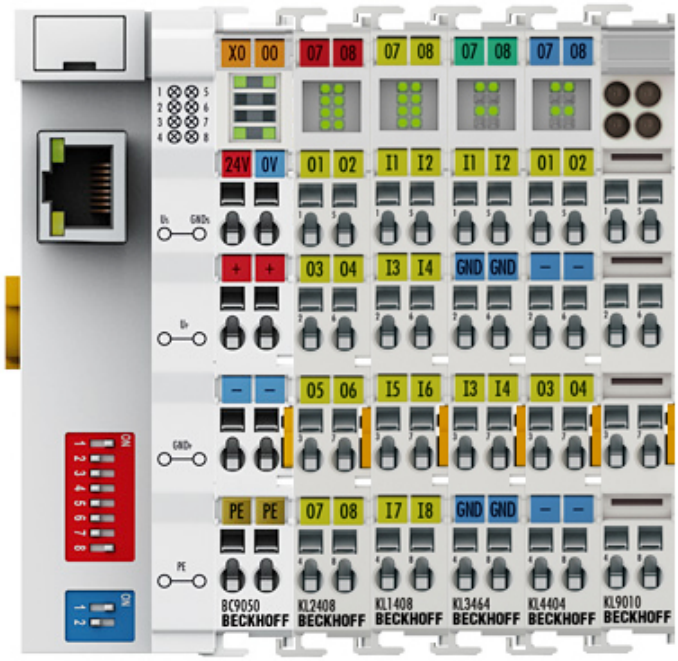
Beckhoff offers a wide range of devices which are commonly used in automation installations, on the basis of which you can create an intelligent installation:

- dimmers (e.g. **KL2751**);
- relays (e.g. **KL2641** lub **KM2604**);
- modules to control the engines (e.g. **KL2552** lub **KL2791**) – to control recuperation;
- digital outputs (e.g. **KL2408**) and analog outputs (e.g. **KL4408**);
- digital input modules (e.g. **KL1408**) and analog input modules (e.g. **KL3408**);
- sensors (temperature, humidity, pressure);
- devices monitoring the energy network;
- and many other.

A device which controls the inputs and outputs directly can be a network coupler which also plays the role of MODBUS/TCP interface, e.g. **Beckhoff BK9050** or one of the available PLCs, e.g. **BC9050**. It is recommended to use the PLC when there is the need of creating logical relationships or faster reaction times than it is possible using the **Base** module.

An important feature of the **Beckhoff** controllers is the fact, that they can be easily extended. You can connect 64 extension modules by default or 255 if the bus is extended.

The example configuration of the controller is shown on the next page.



Extension modules can be exchanged for other ones anytime, without the need to reconfigure the controller.

The **Base** module has function of the master device in relation to the **Beckhoff** controllers. It means, that when you integrate the **DOMIQ** system with the **Beckhoff** controller, you can use all its functionalities. The data from the **Beckhoff** controller can be visualized or used in events, timers, lua scripts etc. The outputs can be freely controlled from the **DOMIQ** user interface.

3.1. Additional configuration remarks

During the configuration of the **Beckhoff** controller, in addition to the way of initialization of the IP address, it is also necessary to set the periodic reset of the register No. 4384 - communication watchdog. When the controller receives the first message, restart the register with a set frequency, max. every second. Otherwise the controller stops the communication itself.

The periodic reset can be performed as follows:

1. Add the read-out from the register No. 4384 in the **MODBUS** tab. Set the type *uint16* and enter *wd*.
2. Add an event:
   - In the **Channel** field enter: `E.MODBUS.<interface>.<device>.<register>` e.g. *E.MODBUS.bk.dev.wd*.
   - In the field **Data** enter: *1000*.
3. Add a command:
   - In the field **Name** enter: `C.MODBUS.<interface>.<device>.<register>`, in this case *C.MODBUS.bk.dev.wd*.
   - In the field **Value** enter: *1000*.

4. Reading the inputs and controlling the outputs

The implementation of the MODBUS protocol in the **Base** module allows you not only to read the registers (all available types) and inputs, but also save the values in the registers (only **uint16** and **int16**) and control the outputs (**coil**). The registers, inputs and outputs are read after adding them in the **MODBUS** tab and filling in its parameters (type, name, address).

In the **Beckhoff** controllers the extension modules are counted from the left and the numbers of the inputs and outputs are given in this order. If you connect two modules of the digital outputs, e.g. **KL2408**, then the outputs of the first module have the numbers from 1 to 8 and the second from 9 to 16 etc.

The registers from 0x0000 to 0x00FF are assigned to the inputs and 0x0800 to 0x08FF to the outputs. For example, if there are eight digital inputs connected to the **BK9050** module, e.g. **KL1408**, the input states are stored in the registers from 0 to 7.

There are standard commands which are used to save the values of the registers/outputs. In order to save the value in the register just send the following command:

`C.MODBUS.<interface>.<device>.<register>=value`

e.g. `C.MODBUS.dev.bk.temp=25`.

The output values are saved similarly. The difference is that the output accepts one of four commands: *on*, *off*, *1* or *0*. For example: `C.MODBUS.dev.bk.light=on` or `C.MODBUS.dev.bk.light=0`.

It is also possible to control MODBUS/TCP from the **Display** visualization or **Remote** menu using channel elements. In such case use the following channel identifiers:

`MODBUS.<interface>.<device>.<register>`, e.g. `MODBUS.dev.bk.light`

Using the described functionality you can create the building automation installation based on the **DOMIQ** modules and a single **Beckhoff** controller or use it to complete the installation functionalities based on the **LCN** system.