

UC20-SL2000-OLAC-EC

Application Note | Modbus TCP Server/Slave

Abstract:

The Application Note describes how to set up a UC20-SL2000 controller to run as Modbus TCP Server/Slave device. Therefore, it is necessary to install the Modbus TCP Server software unit on the controller.

Hardware reference

No.	Component name	Article No.	Hardware / Firmware version
1	UC20-SL2000-OLAC-EC	2638920000	-

Software reference

No.	Software name	Article No.	Software version
1	u-create studio	2660130000	1.20.2 patch 4 (or higher)
2	QModMaster (Download link)	-	-

File reference

No.	Name	Description	Version
1	AN0004-UC20-SL2000 ModbusTCPServer.zip	Example project which sets up a Modbus server and writes variables of different data types to Modbus registers	-

Contact

Weidmüller Interface GmbH & Co. KG
Klingenbergstraße 26
32758 Detmold, Germany
www.weidmueller.com

For any further support please contact your
local sales representative:
<https://www.weidmueller.com/countries>

Content

1	Warning and Disclaimer.....	4
2	Requirements	5
3	Install Modbus TCP Server Software Unit.....	6
3.1	Option 1: Create a new Target.....	6
3.2	Option 2: Add the Software unit to a running system	6
4	Modbus TCP Server	9
4.1	Configuration of Modbus server	9
5	Example Program.....	11
5.1	Global Variables	11
5.2	Expert entries	12
5.3	Symbol Configuration	12
5.4	Program Code	13
5.5	Run the Example	13

1 Warning and Disclaimer

Warning

Controls may fail in unsafe operating conditions, causing uncontrolled operation of the controlled devices. Such hazardous events can result in death and / or serious injury and / or property damage. Therefore, there must be safety equipment provided / electrical safety design or other redundant safety features that are independent from the automation system.

Disclaimer

This Application Note / Quick Start Guide / Example Program does not relieve you of the obligation to handle it safely during use, installation, operation and maintenance. Each user is responsible for the correct operation of his control system. By using this Application Note / Quick Start Guide / Example Program prepared by Weidmüller, you accept that Weidmüller cannot be held liable for any damage to property and / or personal injury that may occur because of the use.

Note

The given descriptions and examples do not represent any customer-specific solutions, they are simply intended to help for typical tasks. The user is responsible for the proper operation of the described products. Application notes / Quick Start Guides / Example Programs are not binding and do not claim to be complete in terms of configuration as well as any contingencies. By using this Application Note / Quick Start Guide / Example Program, you acknowledge that we cannot be held liable for any damages beyond the described liability regime. We reserve the right to make changes to this application note / quick start guide / example at any time without notice. In case of discrepancies between the proposals Application Notes / Quick Start Guides / Program Examples and other Weidmüller publications, like manuals, such contents have always more priority to the examples. We assume no liability for the information contained in this document. Our liability, for whatever legal reason, for damages caused using the examples, instructions, programs, project planning and performance data, etc. described in this Application Note / Quick Start Guide / Example is excluded.

Security notes

In order to protect equipment, systems, machines and networks against cyber threats, it is necessary to implement (and maintain) a complete state-of-the-art industrial security concept. The customer is responsible for preventing unauthorized access to his equipment, systems, machines and networks. Systems, machines and components should only be connected to the corporate network or the Internet if necessary and appropriate safeguards (such as firewalls and network segmentation) have been taken.

2 Requirements

To work through this document, it is necessary to install the u-create studio software on an Engineering PC and connect the patch cable between the controller and the pc.

Furthermore, a Modbus TCP client simulator is used in the application example, which must also be executed on the engineering pc.

To understand the content of this document, basic knowledge about handling projects in u-create studio is required.



Please install the newest u-create studio version + patches to avoid problems with the Modbus server.

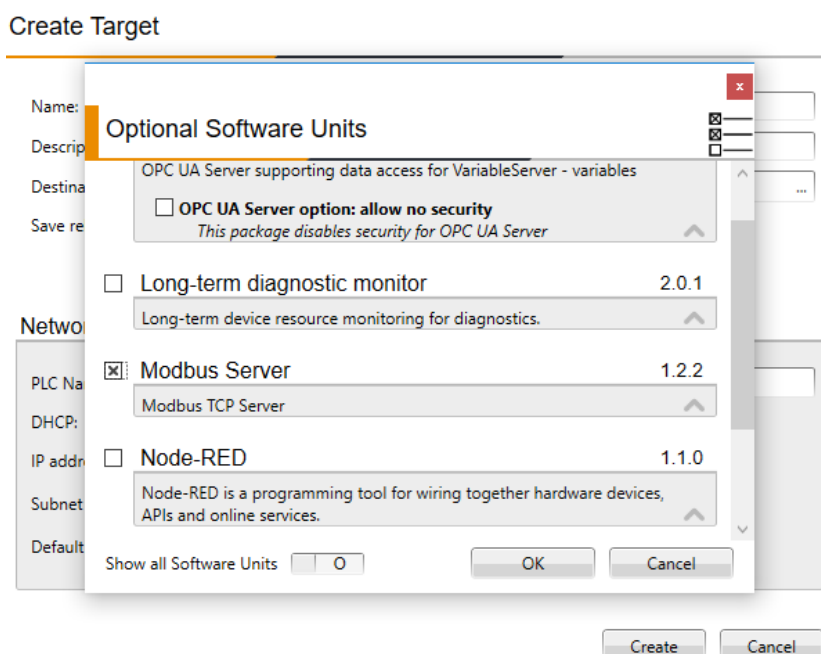
3 Install Modbus TCP Server Software Unit

There are two options to install the software unit. If you have a working PLC/Linux image running on your controller, it is possible to add the missing software unit. Another possibility is to create a new target with the Modbus software unit.

3.1 Option 1: Create a new Target

To create a target and install it on the u-control you first need an empty micro SD card. Insert it into the computer and follow the next steps:

1. Start u-create studio and create a new project or open an existing project
2. Find Project in your toolbar and click **Create Target**
3. Select destination path (SD-card)
4. Select **Modbus Server** in Optional Packages list



5. Create target (this will take several minutes)
6. Insert micro SD card into your u-control and reboot the device. Orange flashing **RUN** led indicates the installation process. When installation is finished successfully **RUN** led is constantly green.
7. Remove SD card from u-control.

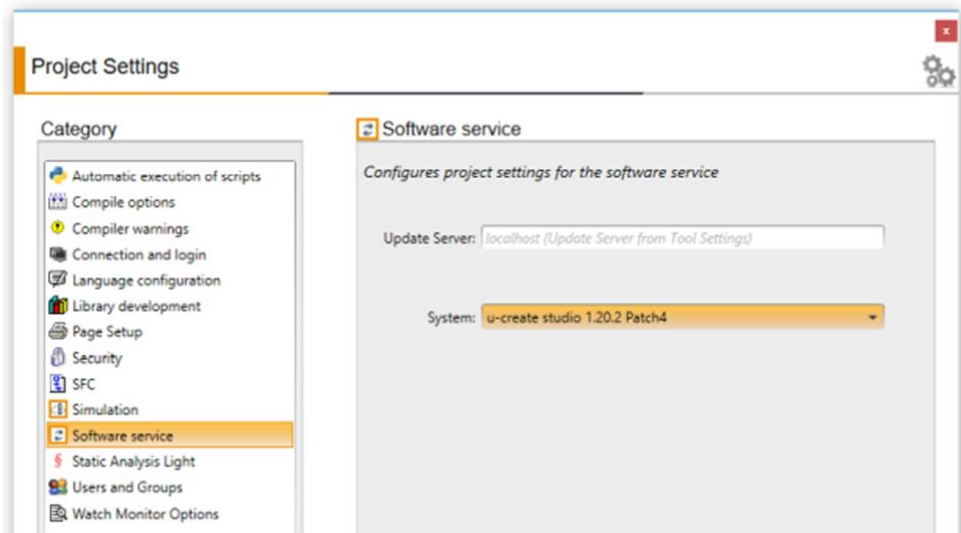
3.2 Option 2: Add the Software unit to a running system

The **Add/Remove Software Units** feature depends on communication via port 1744.

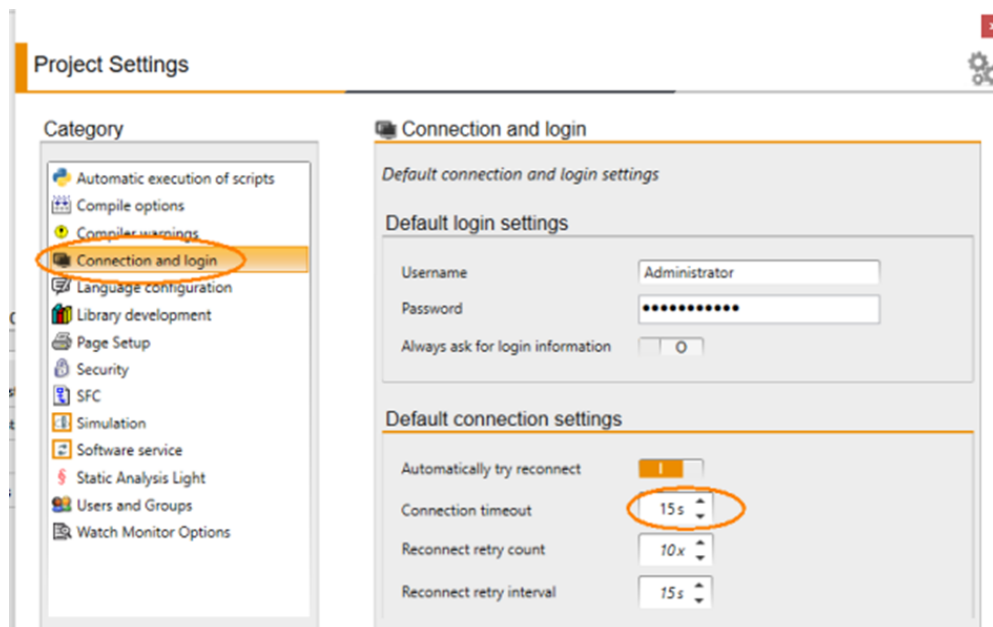
Therefore, you have to allow this communication via incoming and outgoing firewall rules for port 1744 TCP and UDP by adding the corresponding rules manually on your PC. Nevertheless, it can be forbidden or blocked due to company rules.



Please check also that you have selected the same software service version in your project as on the controller.

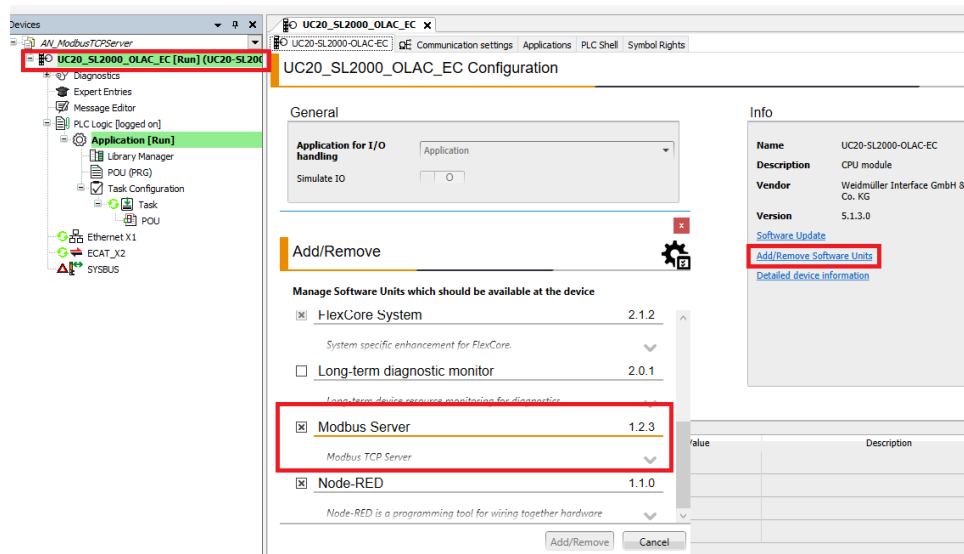


It is useful to increase the **Connection timeout** too. (Default: 5s)



Now it should be possible to add the software unit.

1. Open the current project and login to the controller.
2. Go to the controller configuration and add the Software unit



4 Modbus TCP Server

The configuration of the Modbus Server must be done via the expert entries inside the u-create studio software. These expert entries are configuration settings for the Modbus server running on the Linux system. The settings are read only once when the Modbus Server started.

How to configure the Modbus server is also explained in the u-create studio online help ([Configuration->Configuration of Modbus server](#))

4.1 Configuration of Modbus server

To communicate with the Modbus server on the control the following configuration entries must be set via **Expert Entries** directly on the node control:

```
[ModBus]

    enableRTU = 0

// Modbus TCP Server

    [ModBus.TCP]

        IP      = "192.168.101.100"           //IP-Address of this control
        PORT    = 502                        //Modbus TCP-Port
        TIMEOUT = 600                        //Necessary input parameter
```

The IEC variables, which are shared with the Modbus server must be released first. After that they can be registered at the Modbus server via the **Expert Entries** configuration. The following entries must be added:

```
[ModBusReg]

    [ModBusReg.Adr:0]

//Register value at server side is multiplied by this value for the client
(e.g. "Factor = 0.1" -> a value from e.g. "10" at the server represents value
"1" at the client)

        Factor          = 1.0

// Variablenname must be the same as in the variable browser
```

```

        Variablenname    = "APPL.Application.GVL.iIntReg0"

//Read only := 0, Read/Write := 1

        WritePermission = 0

[ModBusReg.Adr:1]
        Factor           = 0.1
        Variablenname    = "APPL.Application.GVL.wWordReg1"
        WritePermission = 0

[ModBusReg.Adr:2]
        Factor           = 1.0
        Variablenname    = "APPL.Application.GVL.xBoolReg2"
        WritePermission = 0

[ModBusReg.Adr:10]
        Factor           = 100.0
        Variablenname    = "APPL.Application.GVL.rRealReg10"
        WritePermission = 0

...

[ModBusReg.Adr3:1:200] //Array: [<startindex>:<endindex>]
        Factor = 1.0
        Variablenname = "APPL.Application.GVL.var_array1" //Array1 1:200
        WritePermission = 0

...

```



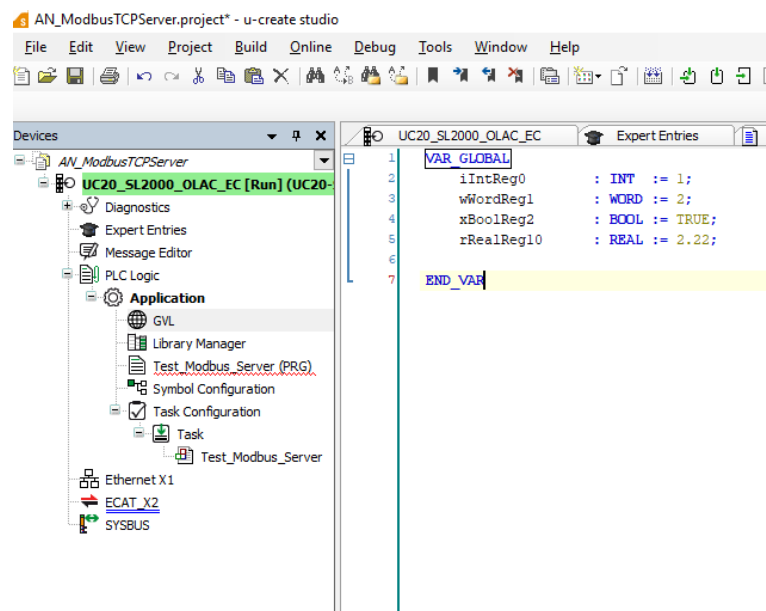
To call released variables via Modbus server the server must be restarted. This must be done mandatorily in the application via function **MODUS_Restart()** of the library **K_Modbusbase**.

5 Example Program

Attached you find a Modbus Server example project which sets up a Modbus Server and writes variables of different data types to Modbus registers.

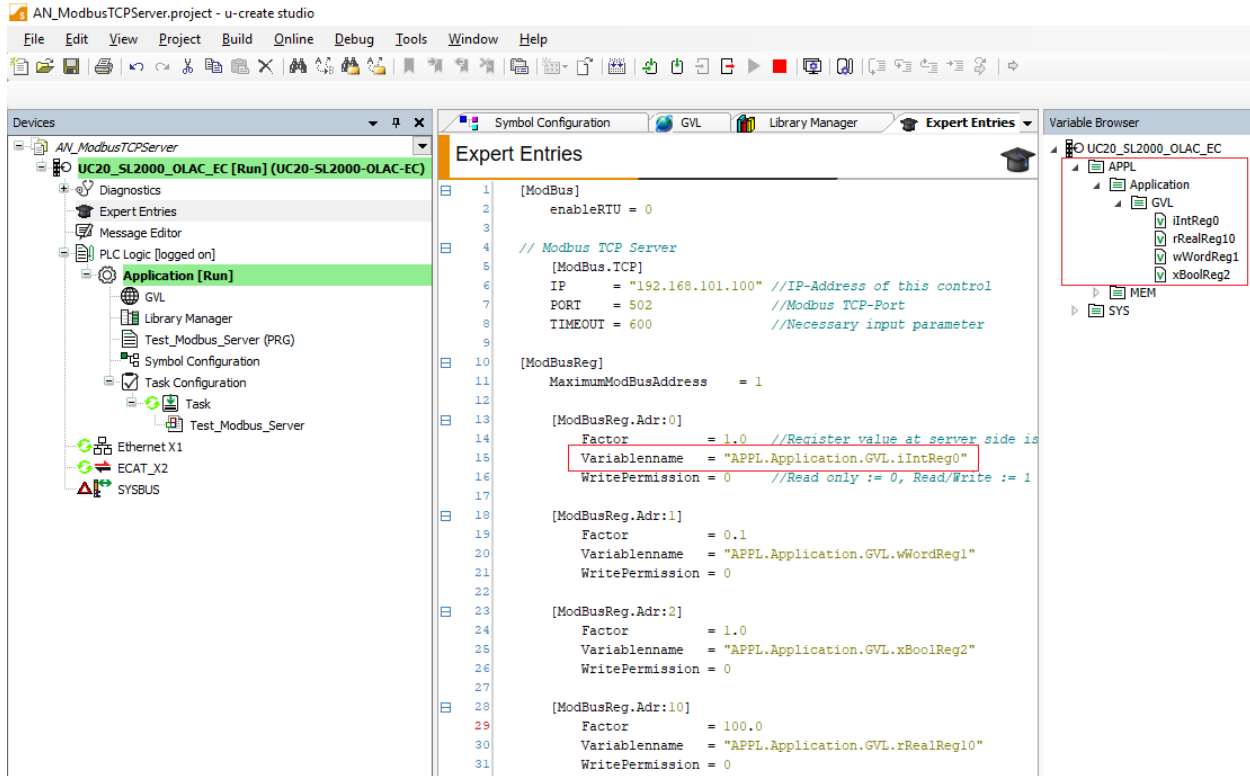
5.1 Global Variables

The variables that are mapped to the Modbus Server registers are created as global variables. It is also possible to use local variables.



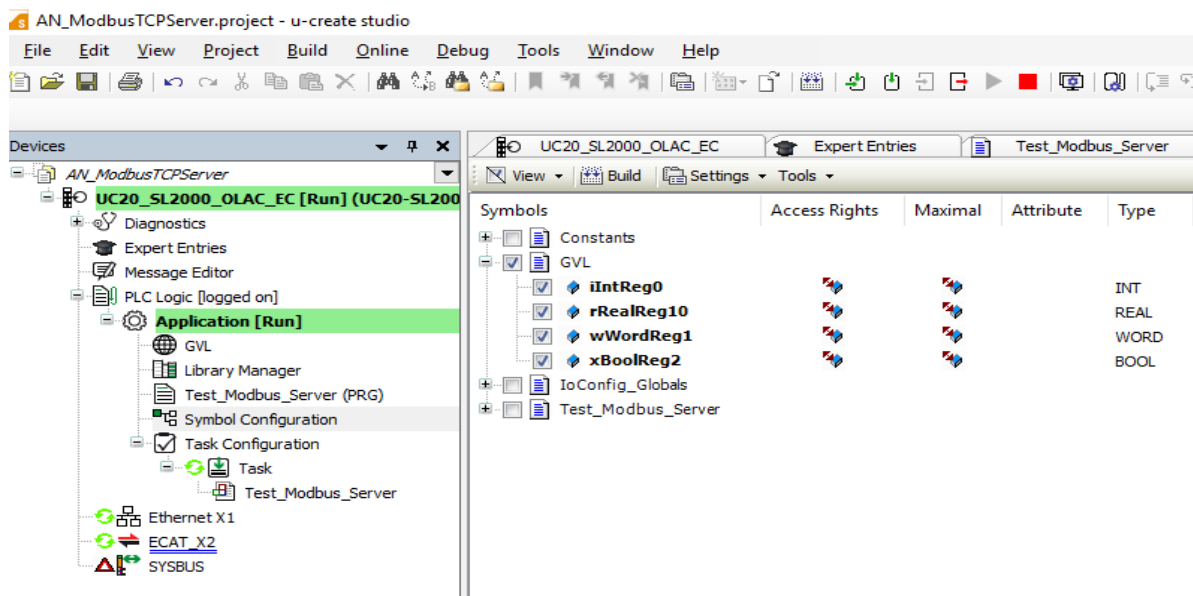
5.2 Expert entries

The expert entries show the Modbus Server configuration from the controller and also the variables that are mapped to the Registers.



5.3 Symbol Configuration

The variables, which are shared with the Modbus client must be released here.



5.4 Program Code

To call released variables via Modbus server the server must be restarted. This can be done via function **ModbusServer_Restart()**. In the example there is also code to generate arbitrary data for the Modbus registers.

5.5 Run the Example

1. Run the PLC program and observe the behavior of the program.
2. Open the simulation program **QModMaster** and execute the Modbus TCP communication.

Function Code: 3

Start Address: 0

Number of Registers: 3

The screenshot shows the UC20_S12000_OLAC_ECApplication.Test_Modbus_Server ladder logic and the QModMaster interface. The ladder logic includes the following code:

```

1 //Configuration see Expert Entries, also Symbol Configuration is probable
2
3 IF mbDoInit[FALSE] THEN
4   mbDoInit[FALSE] := FALSE;
5   eState[ModbusStat] := K_Modbus.K_ModbusBase.ModbusServer_Restart();
6 END_IF
7
8 (*Register 0*)
9 iIntReg0[-360] := iIntReg0[-360] + 1;
10 IF ( iIntReg0[-360] >= 1000 ) THEN
11   iIntReg0[-360] := -1000;
12 END_IF
13
14 (*Register 1*)
15 wWordReg1[28900] := wWordReg1[28900] + 100;
16 IF ( wWordReg1[28900] >= 65000 ) THEN
17   wWordReg1[28900] := 0;
18 END_IF
19
20 (*Register 2*)
21 blinkBool(ENABLE[TRUE] := TRUE, TIMEHIGH[T#1s] := T#1s, TIMELOW[T#1s] := T#1s, OUT[TRUE] => xBoolReg2[TRUE]);
22
23

```

The QModMaster interface shows the following configuration:

- Modbus Mode: TCP
- Slave Addr: 1
- Scan Rate (ms): 1000
- Function Code: Read Holding Registers (0x03)
- Start Address: 0
- Data Format: Dec
- Signed: ☐
- Number of Registers: 3
- Result: 65176, 28900, 1
- Status: Packets: 1958, Errors: 0

3. Another example to read single Modbus Address 10

Function Code: 3

Start Address: 10

Number of Registers: 1

```

(*Register 0*)
iIntReg0[-34] := iIntReg0[-34] + 1;
IF ( iIntReg0[-34] >= 1000 ) THEN
  iIntReg0[-34] := -1000;
END_IF

(*Register 1*)
wWordReg1[1500] := wWordReg1[1500] + 100;
IF ( wWordReg1[1500] >= 65000 ) THEN
  wWordReg1[1500] := 0;
END_IF

(*Register 2*)
blinkBool(ENABLE[TRUE] := TRUE, TIMEHIGH[T#1s] := T#1s, TIMELOW[T#1s] := T#1s, OUT[TRUE] => xBoolReg2[TRUE]);

(*Register 10*)
rRealReg10[360] := rRealReg10[360] + 2.22;
IF ( rRealReg10[360] >= 1000 ) THEN
  rRealReg10[360] := 0;
END_IF

```

The screenshot shows the QModMaster interface with the following configuration:

- Modbus Mode: TCP
- Slave Addr: 1
- Scan Rate (ms): 1000
- Function Code: Read Holding Registers (0x03)
- Start Address: 10
- Data Format: Dec
- Signed: ☐
- Number of Registers: 1
- Result: 35964



For debugging it is possible to show the bus communication via the simulation tool.

