# **1 IBH OPC Editor: Modbus connection**

With the IBH OPC Editor (version 7.3.4 or higher), a Modbus configuration can be transferred to the IBH Link UA (firmware V4.12 or higher).

# **1.1 New Server Connection**

The *New Server Connection command* from the *Edit* menu or clicking the icon opens the *New Server Connection* dialog box.





# **1.1.1 Server Connection**

To establish a connection to an OPC UA server, the connection data must be specified. The *New server connection* dialog box makes it easier to specify the connection data.

#### Note:

The connection data specified by the *New server connection* dialog box will be displayed in the right part of the Project window after completion.

This connection data can be changed at any time in the right part of the project window. If a selection is possible, the changes can be made via drop-down list boxes.

#### New server connection dialog box

The fields for the general settings for the connection to an OPC UA server must be filled out.

#### Name of the server connection

The name is freely selectable.

Name of the server connection: IBH Link UA

#### Server address

If the *IBH Link UA port* to be used for connection is in a network with *a DHCP server*, replace *localhost* with the actual *host name*.

Server address:		
Host name or IP address:	localhost	
-		Server address:
Port:	48010	Host name or IP address: 192.168.1.14
_		
		Port: 48010

If there is **no DHCP server**, replace **localhost** with the IBH Link UA Port IP address (192.168.1.14).

Mew server connection	X
Name of the server connection:	IBH Link UA - ModBus connection
Host name or IP address	192,168,1,14
Port:	48010
C URL opc.tcp://192.16	68.1.14:48010
	Check connection
Security settings:	Click
None     M	essage mode:
C Basic <u>1</u> 28Rsa15	🕞 Signatur
C Basic256	C Signature and Encryption
C BasicSha256	
Login: ( Anonymous) C User name and password	
liser name:	
– Pass <u>w</u> ort:	Store
Session Name:	PLC416 - ModBus connection name
<u>Q</u> K <u>C</u> ancel	Help

#### **Check connection**

Check connection

dialog box, the connection to the online connected IBH Link UA can be tested. Information about the successful connection is displayed.



After completing the New Server Connection

#### **Security settings**

In this field the security procedure and the message mode can be selected.

If a security procedure is selected, certificates must be exchanged between the *IBH OPC Editor* and the OPC UA Server (IBH Link UA). The suggested value is *None*.

Security settings: ----• Non<u>e</u> • Basic<u>1</u>28Rsa15

- C Basic<u>2</u>56
- 🔘 BasicSha2<u>5</u>6

As a safety procedure, you can set *None*, Basic128Rsa15, *Basic256* or *BasicSha256*.

#### Message mode

If a security procedure is used	Message mode:	
as well as Signature and	Message mode: —	• Signatur
Encryption are available.	Signatur	C Signature and Encryption
Otherwise without.	C Signature <u>a</u> no	Encryption

#### Login

In this field, the user name can be specified with the associated password. The default login mode is *Anonymous*.

	Login: C Anonymous G User name and password User name: Admin Password: Password V Store
--	--

#### Session name

The name of the session is freely selectable and can be left blank.

	Session Name:	PLC416 - ModBus connection
OK Confi	rm) Cancel	Help

# 1.2 Add Modbus configuration

The right part of the project window displays the name of the OPC UA server with host name / IP address.

Right-clicking on the name of the server connection opens the context menu. The *Add Modbus Configuration* command starts the configuration process.

😹 Workshop ModBus US.opu - OPCUAEdit				-		×
File Edit Help						
🗋 💕 🛃 🦛 💿						_
Project	🔺 i 🗙		Name of the server conr	nection		
au 🚓 🖌 V 🗈 🕾 L 🗛 🚳			Name	IBH Link UA - N	fodBus cor	nnection
		-	Server address			
Here a TBH Link DA - Modeus connection			Host name / Address	192.168.1.14		
right click			Port	48010	0.1.14.40	1010
New server connection	· II		Security settings	ope.tep.77132.	100.1.14.40	010
New control			Security nolicity	None		
Properties			Message mode	None		
Flopenes		-	Authentication settings			
Add external data			Login	Anonymous		
Add modbus configuration			Session name	PLC416 - Mode	lus connec	tion
Delete						
Cut						
Сору						
Insert						
Transfer selected configuration to the OPC UA Server						
Import						
Export						
Project Server R Certificates						
and a second to		-			CADE	NUM

In the right part of the project window, the *Modbus configuration* window opens.



The existing configuration can be saved at any time.



## 1.2.1 Add new Modbus device



If Modbus configuration is marked in the left part of the project window, the Modbus device properties dialog box can be opened by clicking on the New device icon.



🌆 Modbus device pr	operties	×
Device name:	PLC416_ModBus_Server	
Interface:		
Ethernet	URL or IP address: 192.168.1.10	
C Serial	Connection parameters:	
Description:	Tank Level SoftPLC CPU 416, Modbus-Connection	
Device revision:	1.0 The information is freely selectable.	
Hardware revision:	1.0 The fields can be left blank.	
Manufacturer:	IBHsoftec GmbH	
Model:	PLC416 V4.16	
RevisionCounter:		
Serial number:	4711	
Software revision:		
ок	Cancel Help	

If the device name and the interface are specified, the dialog box can be confirmed with *OK*.

#### 1.2.2 Modbus TCP interface / Modbus variable

The Modbus organization specifies the form in which variables can be present.

Device manufacturers follow these guidelines and provide devicespecific tables that list the address of the variable and how it is addressed.

In order to define the Modbus variable as OPC tag in the *Modbus* Variable Properties dialog box of the IBH OPC UA Editor, knowledge of the fieldbus node with its data, data coding, addressing and transactions is required.

Following are excerpts from specifications of the Modbus organization, which are necessary for the definition of the OPC tags. The tables / descriptions of the fieldbus node provide this information in a similar form.

## **Modbus Functions – Partially –** (Modbus Organization)

functions. **Physical Discrete Inputs Read Discrete Inputs** 

The action of reading and writing variables is determined by

Internal Bits	Read Colls	
or	Write Single Coil	
Physical coils	Write Multiple Coils	
Physical Input Registers	Read Input Register	
	Read Holding Registers	
	Write Single Register	
Internal Registers	Write Multiple Registers	
or Physical Output Registers	Read/Write Multiple Registers	
	Mask Write Register	
	Read FIFO queue	
File record cooce	Read File record	
File record access	Write File record	
	Read Exception status	
	Diagnostic	
Diagnostics	Get Com event counter	
Diagnostics	Get Com Event Log	
	Report Server ID	
	Read device Identification	
Other	Encapsulated Interface Transport	
	CANopen General Reference	

Most Modbus devices do not provide all the functions. However, writing or reading functions are usually supported by all. Therefore, the IBH OPC UA Editor only supports the functions for reading and writing values.

# Modbus data formats

Name data access (Storage area)	Quantity Data type	Access Type memory access	Comment
Discrete Input physical inputs process image	<b>1 Bit</b> (Single bit)	Read-Only	These data types can be provided by an I / O system.
Coils Discrete outputs physical outputs process image	<b>1 Bit</b> (Single bit)	Read-Write	These data types can be changed by an application program.
Input Registers	<b>16-Bit</b> Wort	Read-Only	These data types can be provided by an I / O system.
Holding Registers	<b>16-Bit</b> Wort	Read-Write	These data types can be changed by an application program.

The Modbus devices provide memory in which variables are available that can be accessed using the access type specified in the table.



Modbus devices usually have separate memory areas per table type.

The addresses of the memory areas and their access options are provided by the Modbus device manufacturers, mostly in tabular form.

# Example - Listing: Access to variables (fictitious Modbus devices manufacturer information)

	Addressing 1 Bit Register		Start address	End address	Access	Description / Storage area
	<b>X</b> <sup>(1)</sup>		0x0000	0x00CF	Read/Write	Process data interface.
ata		X <sup>(2)</sup>	0x00D0	0x00FF	Read/Write	Physical inputs, Process image
s Da	X <sup>(3)</sup>		0x0100	0x01CF	Read-only	
ces		X <sup>(4)</sup>	0x01D0	0x01FF	Read-only	December 1. Interfaces
Pro		X <sup>(5)</sup>	0x0200	0x02FF	Read/Write	Process data interface. Physical outputs, Process image
	X <sup>(6)</sup>		0x0300	0x03FF	Read/Write	, ,

#### Modbus register addressing

	Addressing		Start	End	A00000	Description / Storage area	
	1 Bit	Register	address	address	Access	Description / Storage area	
		<b>X</b> <sup>(7)</sup>	0x0400	0x040F	Read-only	Status register	
		X <sup>(8)</sup>	0x0410		Read-only	Process image length in bits, analog outputs	
gnosis		x	0x0411		Read-only	Process image length in bits, analog inputs	
Diaç		x	<b>X</b> 0x0412		Read-only	Process image length in bits, digital outputs	
		x	0x0413		Read-only	Process image length in bits, digital inputs	
		X <sup>(9)</sup>	0x0420	0x042F	Read/Write	Watchdog Register	
cial ster		X <sup>(10)</sup>	0x0430	0x043F	Read/Write	Error Register	
Spec		X	0x0440	0x044F	Read/Write	Command Register	
~ R		Х	0x0450	0x045F	Read/Write	Internal Register	

(nn) As an example, for the definition of a variable exists.

#### Note:

The address information in the Modbus devices Manufacturer specifications are often in hexadecimal form. These addresses are to be converted into a decimal address for input in the IBH OPC UA Editors.

# 1.3 Defining Modbus variables in the IBH OPC UA Editors

Right-clicking on the Modbus device name (PLC 416 ModBus Server) opens the context menu.



The *New Variable* command opens the *Modbus Variable Properties* dialog box.

This dialog defines the variables that the OPC server should connect to. This can be a write variable, a read variable or a read / write variable.

Properties modbus variable			×	
<u>N</u> ame:		Unit ID: 255		
Type: Access C Boolean C UInt16 C Int16 C UInt22 C UInt32	s: d: Ø Bead Address: <sup>(1)</sup> ReadCols Ø ReadCols	Write: Write Address: WriteSingleCoil WriteSingleCoil	ible	×
C Int32	Number: 1	Name:	Access:	Unit ID: 255
Analog limit values:	Minimum:	C <u>B</u> oolean C UInt <u>1</u> 6	Read: ✓ Read Address:	Write:
History:	Sampling interval (sec): 0.5	C Int1 <u>6</u> C UInt <u>3</u> 2	C ReadInputRegisters	WriteMultipleRegisters
UA node recognition:	ns=9;s=PLC416_ModBus_Server.	⊂ Int32 ⊂ Eloat	Number: 1	32-bit swap word order
		Analog limit values:	Minimum:	Ma <u>xi</u> mum: 0
		History:	Sampling interval (sec): 0.5	Buffer sige: 1000
		UA node recogniti	on: s=9;s=PLC416_ModBus_Server.	
		QK	Cancel	Help

## 1.3.1 Properties dialog box Modbus variable

# Modbus TCP / RTU interface

Modbus device manufacturers describe in tables the Modbus fieldbus node functions. Based on these descriptions, the definitions of the accesses to variables are defined.

#### Name (Variable)

The name can be selected freely but must correspond to the OPC UA specification (letters **A-Z**, **a-z**, numbers **0-9**, no special characters, no symbols, no dots, no colon.) As a special character only \_ under line.

#### **Unity-ID**

For Modbus TCP the Unity-ID = 255. This is set at default. For Modbus RTU, the address must be set according to the slave address.

nit ID:	255
---------	-----

#### Туре

#### Boolean

One (1) bit information (**Boolean**), which can have the states TRUE (1) and FALSE (0). A variable of type **Boolean** occupies 1 bit in a register address (16 bits). An array of 16 variables of type Boolean occupy an entire register address (16 bits).

#### UInt16

Positive 16-bit **unidirectional integer** (number) between 0 and 65,535 (2<sup>o</sup> to 2<sup>+16</sup>). A **UInt16** variable occupies a register address (16 bits).

## Int16

Positive or negative 16-bit integer (number) between -32,767 and +32,767 ( $-2^{+15}$  to  $2^{+15}$  -1). An Int16 variable occupies a register address (16 bits).

#### UInt32

Positive 32-bit unidirectional integer (number) between 0 and 4,294,967,295 (2<sup>o</sup> to 2<sup>+32</sup>). A **UInt32** variable occupies two (16-bit) register addresses.



#### Int32

Positive or negative 32-bit integer (number) between -2,147,483,654 and +2,147,483,654 (- $2^{+31}$  to  $2^{+31}$  -1). An **Int32** variable occupies two (16-bit) register addresses.

#### Float

A variable of data type **Float** represents a fractional number that exists as a 32-bit floating-point number (REAL). A **Float** variable occupies two (16-bit) register addresses.

## 1.3.2 Addresses when accessing read / write

#### Note:

The address information in the Modbus devices Manufacturer specifications are often in hexadecimal form. These addresses are to be converted into a decimal address for input in the IBH OPC UA Editors.

## Data type UInt16 and Int16

A word address (16-bit) is used for accessing variables of a data type. Address 0 addresses the variable that occupies the first 16 bits of a data area. With address 1, the variable occupying the second 16 bits of a data area is addressed. Address 3 addresses the third variable (16-bit) of a data area, and so on.

## Data types UInt32, Int32 and Float

Two words (2 x16-bit = 32 bits) are always required for accessing variables of these data types. Address 0 addresses the first 32-bit variable of these data types. Address 1 must not be addressed because this address would address the low word of the 32-bit variable. Address 2 addresses the variable occupying two words 2 and 3 in the data area. The address 4 addresses the third 32-bit variable. etc.

The order of the two 16-bit words, which consist of the data types UInt32, Int32 and Float, can be set.

• **High-Word** – Low-Word or Low-Word – **High-Word**.

## Data type Boolean

A bit address is used to access bits in the data area.

- Address 0 addresses bit 0 in the word address 0 area.
- Address 1 addresses bit 1 in the word address 0 area.
- Address 16 addresses bit 0 in the word address 1 area.
- Address 66 addresses bit 2 in the word address 3 area.

# 1.4 Access Read Only

## Read Discrete Input (read only – Bit access)

#### Data type Boolean

#### Example: - Fictitious Modbus table (3)

Start Word Address 0111<sub>hex</sub> = Bit Address 1110<sub>hex</sub> = 4368<sub>dec</sub> - Read-only - Bit Access - Physical inputs, Process image. There are 7 input bits to be defined as OPC tags.

This setting addresses variables from the registers of the *Discrete Inputs* whose contiguous status comes from digital inputs. Access: Read: Read: Read: Read Address: 4368 ReadCoils ReadDiscreteInputs Number: 7

The address of the first variable and the number of variables must be specified.

**Screenshot:** 7 individual bits are read from bit address 4368 of the memory area of the physical inputs.

# Read Input Registers (Read Only)

## All data types except Boolean (example: UInt16 - 16-bit)

## Example: - Fictitious Modbus table (7)

Initial register address 400<sub>hex</sub> = 1024<sub>dez</sub> - Read-only - Word access -

The contents of 10 status registers are to be addressed as OPC tags.

This setting addresses variables from the diagnostics area (*Input Registers*). The start register address and the number of registers must be specified.

Screenshot: 10 numbers (unsigned) are read with 16 bits each from word 1024 of the status registers.

Read_Input_Registers_UInt16
Access:
🔽 Read
Address: 1024
ReadInputRegisters
C ReadHoldingRegisters
Number: 10

# 1.5 Access Read and Write

## **Read Coils / Write Single Coil**

### Data type Boolean

#### Example: - Fictitious Modbus table (1)

Register Address  $0010_{hex}$  = Bit Address  $100_{hex}$  =  $256_{dec}$  - Read-Write - Bit Access - Physical Inputs, Process image. Define 1 input bit as OPC tag.

With this setting, variables from the registers of the coils are addressed. The address of the variable and the number of variables (1) must be specified.

Read_Coils_Write_Single_Coils_Boolean	Unit ID: 255
Access:	Write:
🔽 Read	Vrite
Address: 256	Address: 256
ReadCoils ReadDiscreteInputs	WriteSingleCoil
Number: 1	32-bit swap word order

**Screenshot:** 1 single bit is read with bit address 256 of the memory area of the physical inputs.

# **Read Coils**

#### Data type Boolean

Example: - Fictitious Modbus table (1)

If such a variable is only defined as a read variable, it is defined as

an OPC tag with the status **Read**.

Register Address  $0014_{hex}$  = Bit Address 140<sub>hex</sub> = 320<sub>dec</sub> - Read - Bit Access -Physical Inputs, Process image. 8 input bits should be defined as OPC tag.

Screenshot: 8 single bits are read from the bit address 320 of the memory area of the physical inputs.

Read_Coils_Boolean
Access: Read:
Read
Address: 320
C ReadDiscreteInputs
Number: 8

# **Read Coils / Write Multiple Coils**

#### Data type Boolean

#### Example: - Fictitious Modbus table (6)

Register Address  $0310_{hex}$  = Bit Address  $3100_{hex}$  =  $12544_{dec}$  - Read-Write - Bit Access - Physical Inputs, Process image.

12 input bits should be defined as OPC tag.

With this setting, variables are addressed from the registers of the coils whose contents reflect individual bits. These can be individual

Read_Coils_Write_Multiple_Coils_Boolean	Unit ID: 255
Access: Read:	Write:
🔽 Read	Vrite
Address: 12544	Address: 12544
C ReadCoils	WriteMultipleCoils
Number: 12	🔲 32-bit swap word order

outputs as well as individual inputs. The address of the first specified variable and the number of variables are specified.

**Screenshot:** 12 individual bits are defined from the bit address 12544 of the memory area of the physical outputs.

# **Read Holding Register / Write Single Register**

#### (All data types except Boolean)

#### Example: - Fictitious Modbus table (2) - Data type INT16 (integer).

Register address 00D0hex = 208dec - Read-Write - word access -

Physical Inputs, Process image.

One (1) integer number should be defined as OPC tags.

With this setting variables from the *Holding Registers* 

RW_Holding_Reg_Single_Reg_Int	Unit ID: 255
Access:	
Read:	Write:
🔽 Read	Vrite
Address: 208	Address: 208
<ul> <li>ReadInputRegisters</li> <li>ReadHoldingRegisters</li> </ul>	WriteSingleRegister
Number: 1	32-bit swap word order

are addressed whose contents reflect individual registers. This can be, for example, an analogue input. The address of the first variable and the number (1) of the variables are specified.

**Screenshot:** An integer number is defined from the word address 208 of the storage area of the physical outputs.

# Read Holding Registers / Write Multiple Registers (All data types except Boolean)

#### Example: - Fictitious Modbus table (5) - Data type INT16 (integer).

Register Address  $0210_{hex} = 528_{dez}$  - Read-Write - Word Access - Physical Outputs, Process image.

9 integer numbers should be defined as OPC tags.

With this setting variables from the *Holding Registers* are addressed whose contents reflect individual registers.

•	
RW_Holding_Register_Multiple_Reg_Int	Unit ID: 255
Access: Read: IV Read Address: 528	Write: Write Address: 528
C ReadInputRegisters	WriteMultipleRegisters
Number: 9	32-bit swap word order

These can be individual outputs (analog outputs). The address of the first specified variable and the number of variables are specified.

Screenshot: 9 integer numbers from the word address 528 of the memory area of the physical outputs are defined as OPC tags.

## **Read Holding Registers**

#### (All data types except Boolean)

If such a variable is only defined as a read variable, it is defined as an OPC tag with the status **Read**.

# Example: - Fictitious Modbus table (8) - Data type UINT16 unsigned integer number.

Register address  $0450_{hex} = 1104_{dez}$  -Read-Write - word access - special register.

6 unsigned integer numbers should be defined as OPC tags.

With this setting variables from the *Holding Registers* are addressed whose

Read_Holding_Registers_UInt16
Access:
Redu;
₩ <u>R</u> ead
Address: 1104
C ReadInputRegisters
ReadHoldingRegisters
Number: 6

contents reflect individual registers. The address of the first specified variable and the number of variables are specified.

**Screenshot:** 6 numbers (unsigned) from the word address 1104 of the memory area of the special registers are defined as OPC tags.

# Read Holding Registers / Write Multiple Registers (Data types UInt32, Int32, Float)

#### Example: - Data type float (real number).

Register Address  $0460_{hex} = 1120_{dez}$  - Read-Write - Word Access - Physical Outputs, Process image.

You must define 5 floating point numbers as OPC tags.

With this setting	RW_Holding_Reg_Multiple Reg_Float	Unit ID: 255
variables from the	Access:Read:	Write:
Holding Registers are	Read	Vrite
addressed whose	Address: 1120	Address: 1120
contents reflect individual		WriteMultipleRegisters
registers. These can be		
individual outputs	Number: 5	🔲 32-bit swap word order
(analog outputs).		

The address of the first specified variable and the number of variables are specified.

**Screenshot:** 5 floating-point numbers starting from the word address 1120 are defined in the *Holding Registers* as OPC tags.

Note:
UInt32, Int32, and Float numbers occupy two (2) 16-bit (32-bit) words.
The order of the two 16-bit words that make up these data types is adjustable.
High Word - Low Word or Low Word - High Word

## 1.5.1 Analog-limits

Analog limit values can be specified.

-Analog limit values:				
Check limit values	Minimum:	100.0	Maximum:	1000.0

### 1.5.2 History

While **OPC Data Access** provides real-time access to data, **OPC Historical Data Access**, also known as OPC HDA, supports access to data already stored. Activation of a variable as historical data as well as sampling interval and number of values (buffer size) is done via the dialog box.



## **OPC UA Node detection**

The OPC UA Node name of a variable is displayed in the dialog box.

UA node recognition: ns=9;s=PLC4	416_ModBus_Server.RW	_Holding_Reg_Multiple Reg_Float
----------------------------------	----------------------	---------------------------------

# 1.6 Transfer Modbus configuration to the OPC UA server (IBH Link UA)

The variables defined as OPC tags are displayed.



Name	Data type	Access	R address	W address	Number of	Node name
Read_Discrete_Inputs_Boolean	Boolean	R	4368		7	PLC416_ModBus_Server.Read_Discrete_Inputs_Boolean
Read_Input_Registers_Int16	Int16	R	480		4	PLC416_ModBus_Server.Read_Input_Registers_Int16
Read_Input_Registers_UInt16	UInt16	R	1024		10	PLC416_ModBus_Server.Read_Input_Registers_UInt16
Read_Coils_Write_Single_Coils_Boolean	Boolean	RW	256	256	1	PLC416_ModBus_Server.Read_Coils_Write_Single_Coils_Boolean
Read_Coils_Boolean	Boolean	R	320		8	PLC416_ModBus_Server.Read_Coils_Boolean
Read_Coils_Write_Multiple_Coils_Boolean	Boolean	RW	12544	12544	12	PLC416_ModBus_Server.Read_Coils_Write_Multiple_Coils_Boolean
RW_Holding_Reg_Single_Reg_Int	UInt16	RW	208	208	1	PLC416_ModBus_Server.RW_Holding_Reg_Single_Reg_Int
RW_Holding_Register_Multiple_Reg_Int	Int16	RW	528	528	9	PLC416_ModBus_Server.RW_Holding_Register_Multiple_Reg_Int
Read_Holding_Registers_UInt16	UInt16	R	1104		6	PLC416_ModBus_Server.Read_Holding_Registers_UInt16
RW_Holding_Reg_Multiple Reg_Float	Float	RW	1120	1120	5	PLC416_ModBus_Server.RW_Holding_Reg_Multiple Reg_Float

If all Modbus variables are defined as OPC UA tags, the Modbus configuration can be transferred to the OPC UA server.

A right-click on *Modbus configuration* opens the context menu.

Worksho	p ModBus US.opu - OPCUAEdit — 🛛	×
File Edit	Help	
🗋 🞽 🛃		
Project		▼ # ×
🖈 🖆 🗙	3 🗈 🖻 🛧 🏠	
🖃 📲 IBH Li	nk UA - ModBus connection	
	odbus configuration	
[	Delete	]
	Read modbus configuration from server	
	Transfer modbus configuration to server	
	Import	
	Export	
😡 Project 🗵	🕏 Server 🛛 🖾 Certificates	

The command to transfer the Modbus configuration must be confirmed.



The transmitted Modbus configuration is displayed under *Diagnostics* in the IBH Link UA Browser window.

🗱 IBH Link UA - Diagnostics	× +							
← → ♂ ☆	i 192.168.1.14/?_=/diaglistvi	ew		••	· 🖂 🕁	lii1\	🗉 a	≡
OPC server	r is running Logout Update	password						
Network ID	D Connection name	Address	Time	Source	Error number	Error Text		
Security	- PLC416_ModBus_Server	192.168.1.10:502	31.1.2019 12:58:35	PLC	0	Connection est	ablished	
Certificates	lear diagnose 🧭							
Time settings								
OPC Client								_
Diagnostics								
MQTT								
SoftPLC								
MicroSD								
www.ibhsoftec.com Co	i <b>tec</b> entact Wiki					BH Link	UA tr / client	

# 1.7 IBH OPC UA Editor Server Window

A *Modbus configuration* successfully transmitted to the OPC UA server can be displayed online in the server window.

The variables (*Modbus configuration, Modbus device, OPC tag*) are listed in the left-hand server window. Clicking on a variable displays the variable definitions with the status in the right-hand server window. The status of this OPC tag is constantly being updated.

🗱 Workshop ModBus US.opu - OPCUAEdit			- 🗆 X
File Edit Help			
Sever U V V Variable transfer Variable transfer	•	Nodeld Namespacifidex Identifier Type Identifier Properties NodeClass BrowseName DisplayName Description WriteMask UseWriteMask ValueRank AccessLevel UseAccessLevel MinimumSamplingInterval Historiang	9 5 5 5 5 6 6 7 7 7 7 7 7 7 7 7 7 7 7 7 7
RW_Holding_Reg_Multiple Reg_Float     WANdes	•	DataType DataType NamespaceIndex IdentifierType Udentifier Value Value	Boolean 0 Numeric 1 (true, true, tru
	·		CAPS NUM

# **1.8 Modbus connection - examples**

The IBH SoftPLC PLC416 has the option of a Modbus connection. In the example, variables are defined as OPC tags. This Modbus configuration is transmitted to the IBH Link UA and the variables are displayed in the **UAExpert client program**.

The data blocks DB501 (**DBIn - R**ead) and DB502 (**DBOut - R**ead / **W**rite) are created in the PLC416.



The OPC tags from the IBH UA Editor Program *Workshop ModBus US.opu* are used. The file must be opened with the IBH UA Editor and the Modbus configuration must be transferred to the IBH Link UA.

🗱 Workshop ModBus US.opu - OPCUAEdit							– 🗆 X
Ele Edit Help	Name	Data tupo	Arrar	Daddrard	Waddrard	Number of	Node para
Construction     C	<ul> <li>Kasd_Discrete_Inputz_Boolean</li> <li>Kasd_Input_Register_Unit6</li> <li>Read_Input_Register_Unit6</li> <li>Read_Colis_Write_Single_Colis_Boolean</li> <li>Read_Colis_Boolean</li> <li>Read_Colis_Boolean</li> <li>Read_Colis_Boolean</li> <li>Read_Colis_Register_Unit6</li> <li>RW_Holding_Register_Unit6</li> <li>Read_Holding_Register_Unit6</li> <li>RW_Holding_Reg_Multiple Reg_Float</li> </ul>	Boolean Int16 Boolean Boolean Boolean UInt16 Int16 UInt16 Float	R R RW RW RW RW RW RW	4368 480 1024 256 320 12544 208 528 1104 1120	256 12544 208 528 1120	7 4 10 1 8 12 1 9 6 5	PLC416_ModBur_Server.Read_Discrete_Inputz_Boolean PLC416_ModBur_Server.Read_Input_Registery.Int16 PLC416_ModBur_Server.Read_Input_Registery.Int16 PLC416_ModBur_Server.Read_Colis_Write_Single_Colis_Boolean PLC416_ModBur_Server.Read_Colis_Boolean PLC416_ModBur_Server.Read_Colis_Boolean PLC416_ModBur_Server.Read_Colis_Reg_Inst_Boolean PLC416_ModBur_Server.Read_Colis_Reg_Inst_Boolean PLC416_ModBur_Server.Read_Holding_Reg_Inst_Multiple.Reg_Int PLC416_ModBur_Server.Read_Holding_Registery.Int1616 PLC416_ModBur_Server.Read_Holding_Registery.Int1616 PLC416_ModBur_Server.Read_Holding_Registery.Int1616
							CAPS NUM

The STEP 7 PLC program must be opened and the blocks with the system data must be loaded into the PLC416 soft PLC.

🍠 SIMATIC Manager - [ModBus_Variable_US \\TTI-Server\Tti-Data\Workshop\ModBus_Variable US] 📃 🗌	• <b>×</b>
🗃 File Edit Insert PLC View Options Window Help	- 8 ×
📘 🗅 😂   🎛 🐖   X 🗈 🛍   🍙 🗣 💁 🎦 🏥 🏢   🎰   < No Filter > 💽 🍞	<b>%</b> 📾
Image: Source blocks     Image: Source blocks	
Press F1 to get Help. IBI	HSoftPL 🏑

# 1.9 Unified Automation UaExpert - The OPC Unified Architecture Client

The **UaExpert program window** lists the **OPC tags** transmitted by the IBH OPC UA Editor and the corresponding **UA nodes**.

Unified Automation UaExpert - The OPC Unified Architecture	e Clien	t - NewProject*							
<u>File View Server Document Settings H</u> elp									
🗋 🖉 🕞 🗭 🧿 💠 🗕 🌣 🗙 🔧 🖉		8 🛛 🖵							
Project 🗗 🛪	Dat	ta Access View							Θ
Y 🎵 Project	#	Server	Node Id	Display Name	Value	Datatype	Source Timestamp	Server Timestamp	Statuscode
✓	1	BHLinkUA	NS9 String PLC416	Read_Discrete_Inputs_Boolean	{true,true,true,true,true,true,true}	Boolean	15:03:30.154	15:03:30.402	Good
IBHLinkUA@ibhlinkua_001037	3	IBHLinkUA	NS9 String PLC416	Read_Input_Registers_Int10 Read_Input_Registers_UInt16	{65535,3456,12345,567,897,4711,54321,6545,10,25}	UInt16	15:03:30.156	15:03:30.402	Good
Documents     Data Access View	4	IBHLinkUA	NS9 String PLC416 NS9 String PLC416	Read_Coils_Write_Single_Coils_Boolean	true (true true true faire true true true)	Boolean	15:03:30.156	15:03:30.402	Good
5 blances new	6	IBHLinkUA	NS9 String PLC416	Read_Coils_Write_Multiple_Coils_Boolean	{false, false, true, true, false, false, true, true, true, true, true, false}	Boolean	15:03:30.158	15:03:30.402	Good
Address Space 8 ×	8	IBHLinkUA	NS9 String PLC416 NS9 String PLC416	RW_Holding_Reg_Single_Reg_Int RW Holding Register Multiple Reg Int	2048 {-32768.4567.89013269911872440.1.4711}	Unt16 Int16	15:03:30.158	15:03:30.402	Good
😏 No Highlight 👻	9	IBHLinkUA	NS9 String PLC416	Read Holding Registers Ulnt16	{65535 4711 4712 50000 12345 56789}	UInt16	15:03:30.160	15:03:30.402	Good
Cont ^	10	IBHLINKUA	N2912trind(PLC410	Kvv_Holding_Keq_viuitiple Keq_Float	{1.23/79,7.03432,23.23,4.39182,4.27934}	Float	15:03:30.100	15:03:30.402	6000
V 🗀 Objects									
> 💫 Client									
> w Deviceset									
V 📥 Modbus									
> = ClearConfiguration									
PLC416_ModBus_Server									
DeviceManual									
DeviceRevision									
Manufacturer									
Model									
>									
> Int RW_Holding_Reg_Single_Reg_Int									
> 🕮 RW_Holding_Register_Multiple_Reg_Int									
> I Read_Coils_Boolean									
Kead_Coils_Write_Multiple_Coils_Boolean     Market Coils_Write_Coils_Boolean									
Read_Colls_write_Single_Colls_boolean     Read_Discrete Inputs Boolean									
Read Holding Registers Ulnt16									
> Read_Input_Registers_Int16									
> Input_Registers_UInt16									
RevisionCounter									
<ul> <li>SenalNumber</li> <li>SoftwareRegistion</li> </ul>									
Status									
Vrl 🔍									
> 👒 ReadConfiguration									
Status									
> = WriteConfiguration									

The values of the variables can be changed via the diagnostics of the PLC416.

- Example 1: Read\_Discrete\_Inputs\_Boolean, (Read only, bit access, data type Boolean). Initial word address 0111<sub>hex</sub> = 273<sub>dec</sub>, Bit Address 1110<sub>hex</sub> = 4368<sub>dec</sub> word access DB501 DW 546 (byte 546). 7 bits defined as OPC tags.
- Example 2: Read\_Input\_Registers\_Int16, (Read only), All data types except Boolean (Int16), Start Word Address 01E0<sub>hex</sub> = 480<sub>dec</sub> Word Access - DB501 - DW 960 to DW 966. 4 integer numbers defined as OPC tags.
- Example 3: Read\_Input\_Registers\_UInt16, (Read only), All data types except Boolean (UInt16 - unsigned integer number), start word address 400<sub>hex</sub> = 1024<sub>dec</sub> word access - DB501 - DW 2048 to DW 2066. Defines 10 unsigned integer numbers as OPC tags.
- Example 4: Read\_Coils\_Write\_Single\_Coils\_Boolean, (Read-Write, Bit Access, Data Type Boolean), Start Word Address 0010hex = 16dec Bit Address 100<sub>hex</sub> = 256<sub>dec</sub> - Word Access - DB502 -DW 32 (Byte 32). 1 bit defined as OPC tag.
- **Example 5:** Read\_Coils\_Boolean, (read only, bit access, Boolean data type), start word address 0014<sub>hex</sub> = 20<sub>dec</sub> Bit address 140<sub>hex</sub> = 256<sub>dec</sub>

Word access - DB502 - DW 40 (byte 40). 8 bits defined as OPC tags.

- Example 6: Read\_Coils\_Write\_Multiple\_Coils\_Boolean, (Read-Write, Bit Access, Data Type Boolean), Start Word Address 0310<sub>hex</sub> = 784<sub>dec</sub>, bit address 3100<sub>hex</sub> = 12544<sub>dec</sub> word access DB502 DW 1568. 12 bits defined as OPC tags.
- Example 7: RW\_Holdinq\_Reg\_Single\_Reg\_Int, (read-write, data type INT16 integer number), start-word address 00D0<sub>hex</sub> = 208<sub>dec</sub>, word access DB502 DW 416. One (1) integer numbers defined as OPC tag.
- Example 8: RW\_Holdinq\_Register\_Multiple\_Reg\_Int, (read-write, data type INT16 integer number), start-word address 0210<sub>hex</sub> = 528<sub>dec</sub>, word access - DB502 - DW 1056 to DW 1072. 9 integer numbers defined as OPC tags.
- Example 9: Read\_Holding\_Register\_UInt16, (read-write, data type UINT16 unsigned integer number), start word address 0250<sub>hex</sub> = 1104<sub>dec</sub>, word access DB502 DW 2208 to DW 2218. 6 unsigned integer numbers defined as OPC tags.
- Example 10: RW\_Holding\_Reg\_Multiple\_Reg\_Float, (read-write, data type FLOAT floating point number), start word address 0460<sub>hex</sub>
  - = 1120<sub>dec</sub>, 2-word access DB502 DW 2240 to DW 2258.
  - 5 Floating point numbers defined as OPC tags.