

Name der Serververbindung

Der Name ist frei wählbar.

Name der Serververbindung: IBH Link UA - ModBus Anbindung

Serveradresse

Befindet sich der **IBH Link UA Port**, der zur Verbindung genutzt werden soll, in einem Netz mit DNS-Server, ist **localhost** mit dem tatsächlichen **Hostname** zu ersetzen.

Ist kein DHCP-Server vorhanden ist **localhost** mit der **IBH Link UA Port** IP-Adresse (192.168.1.14) zu ersetzen.

URL anzeigen

Der URL des ausgewählte OPC Servers wird angezeigt.

Verbindung prüfen

Nach der vollständigen Ausfüllung des Dialogfeldes **Neue Serververbindung** kann die Verbindung zum online verbundenen IBH Link UA getestet werden. Eine Information über die erfolgreiche Verbindung wird angezeigt.

Verbindung prüfen

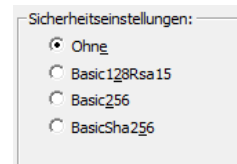
Sicherheitseinstellungen

In diesem Feld können das Sicherheitsverfahren und der Nachrichtenmodus ausgewählt werden.

Wird ein Sicherheitsverfahren gewählt, müssen Zertifikate zwischen dem **IBH OPC Editor** und dem **OPC UA Server** (IBH Link UA) ausgetauscht werden.

Der vorgeschlagene Wert ist **Ohne**.

Als Sicherheitsverfahren können **Ohne**, **Basic128Rsa15**, **Basic256** oder **BasicSha256** eingestellt werden.

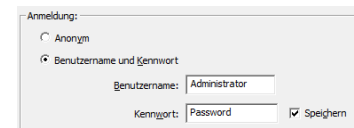


Nachrichtenmodus

Wird ein Sicherheitsverfahren genutzt, stehen Signatur (**Sign**), sowie Signatur und Verschlüsselung (**Sign und Encrypt**) zur Verfügung. Sonst **Ohne**.

Anmeldung

In diesem Feld können Benutzername mit dem dazugehörigen Kennwort festgelegt werden. Der vorgeschlagene Anmeldungsmodus ist **Anonym**.



Sitzungsname

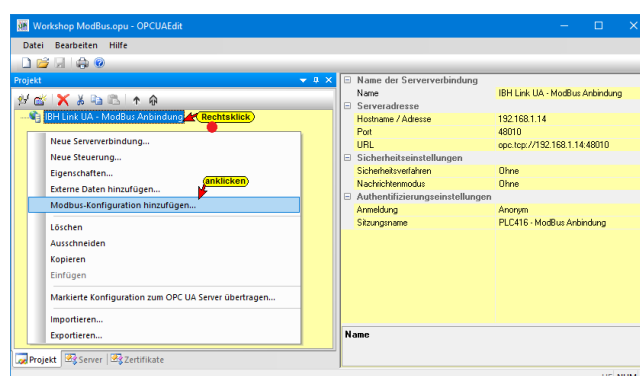
Der Name der Sitzung ist frei wählbar und kann leer bleiben.

Zum Übernehmen der Einstellungen **OK** anklicken. Das Dialogfeld **Neue Serververbindung** wird damit geschlossen.

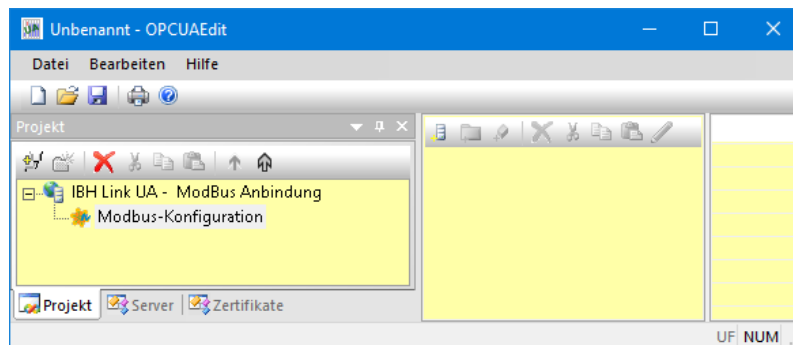
1.2 Modbus-Konfiguration hinzufügen

Im rechten Teil des Projekt-Fensters wird der Name der **OPC UA Server** mit **Hostname / IP-Adresse** angezeigt.

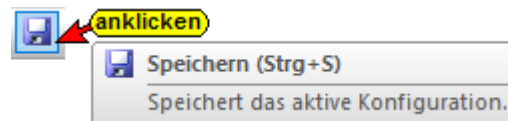
Mit einem Rechtsklick auf den Namen der Serververbindung wird das Kontextmenü geöffnet. Der Befehl **Modbus-Konfiguration hinzufügen** startet den Konfigurationsprozess.



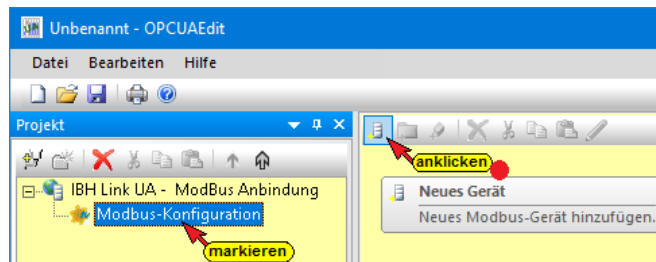
Im rechten Teil des Projekt-Fensters wird das Fenster **Modbus-Konfiguration** geöffnet.



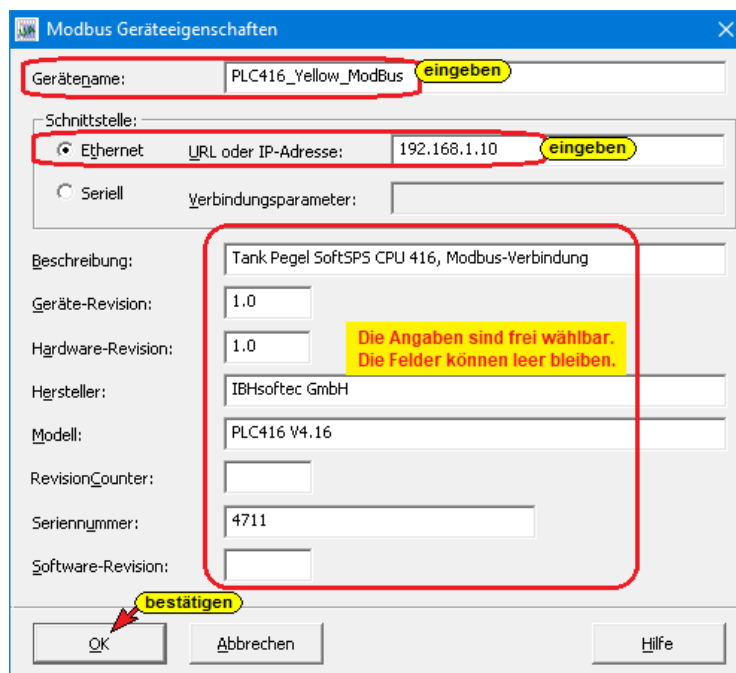
Die vorhandene Konfiguration kann jederzeit gespeichert werden.



1.2.1 Neues Modbus Gerät hinzufügen



Ist im linken Teil des Projekt-Fensters Modbus-Konfiguration markiert kann mit Anklicken des Symbols **Neues Gerät** das Dialogfeld **Modbus Geräteeigenschaften** geöffnet werden.



Sind der Geräte name und die Schnittstelle angegeben kann das Dialogfeld mit **OK** bestätigt werden.

1.2.2 Modbus TCP-Interface / Modbus Variable

Von der Modbus Organisation ist vorgegeben, in welcher Form Variable vorliegen können.

Gerätehersteller folgen diesen Vorgaben und stellen Gerätespezifische Tabellen bereit, in denen die Adressen der Variable und wie diese angesprochen werden, aufgelistet sind.

Um die Modbus Variable im Dialogfeld **Eigenschaften Modbus-Variable** des IBH OPC UA Editors als OPC-Tags zu definieren ist die Kenntnis des Feldbusknotens mit seinen Daten, Datencodierung, Adressierungen und Transaktionen erforderlich.

Im Anschluss sind Auszüge aus Spezifikationen der Modbus Organisation angegeben, die für die Definition der OPC-Tags notwendigen sind. Die Tabellen / Beschreibungen des Feldbusknotens stellen diese Informationen in ähnlicher Form zur Verfügung.

Modbus Funktionen – Auszug – (Modbus Organisation)

Die Aktion des Lesens (**Read**) und Schreibens (**Write**) von Variablen ist mit Funktionen festgelegt.

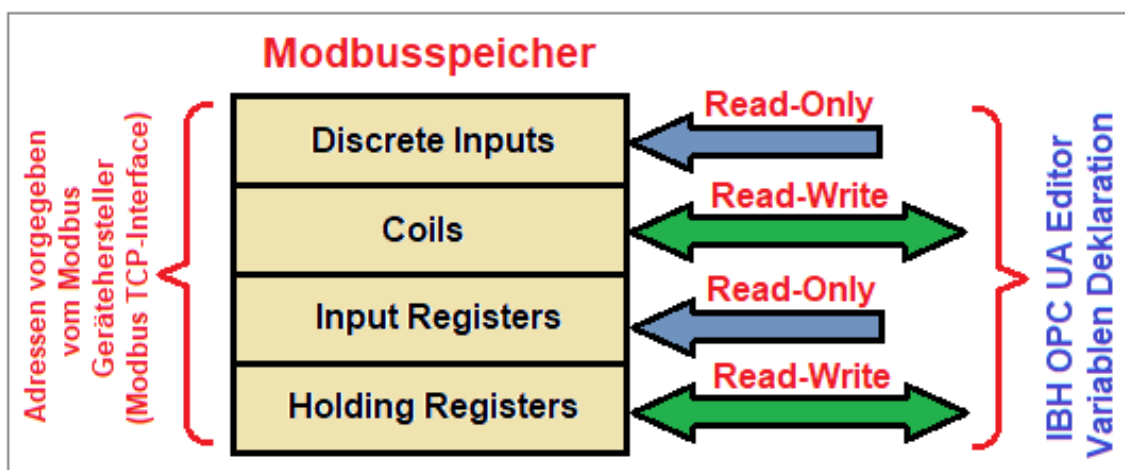
Physical Discrete Inputs	Read Discrete Inputs
Internal Bits or Physical coils	Read Coils
	Write Single Coil
	Write Multiple Coils
Physical Input Registers	Read Input Register
Internal Registers or Physical Output Registers	Read Holding Registers
	Write Single Register
	Write Multiple Registers
	Read/Write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

Die meisten Modbus-Geräte stellen nicht alle Funktionen zur Verfügung. Die Funktionen schreiben oder lesen werden aber in aller Regel von allen unterstützt. Daher unterstützt der IBH OPC UA Editor nur die Funktionen zum Lesen und Schreiben von Werten.

Modbus Datenformate

Name Datenzugriff (Speicherbereich)	Größe Datentyp	Zugangsart Speicherzugriff	Kommentar
Discretes Input Diskrete Eingabe physikalische Eingänge Prozessabbild	1 Bit (Single bit)	Read-Only Nur Lesen	Diese Datentypen kann von einem E / A-System bereitgestellt werden.
Coils Diskrete Ausgabe physikalische Ausgänge Prozessabbild	1 Bit (Single bit)	Read-Write Lesen / Schreiben	Diese Datentypen kann von einem Anwendungsprogramm geändert werden.
Input Registers Eingangsregister	16-Bit Wort	Read-Only Nur Lesen	Diese Datentypen kann von einem E / A-System bereitgestellt werden.
Holding Registers Bestandsregister	16-Bit Wort	Read-Write Lesen / Schreiben	Diese Datentypen kann von einem Anwendungsprogramm geändert werden.

Die Modbus-Geräte stellen Arbeitsspeicher zur Verfügung in dem Variablen vorhanden sind, auf die mit der in der Tabelle angegebenen Zugangsart, zugegriffen werden kann.



Modbus-Geräte haben meistens getrennte Speicherbereiche pro Tabellenart.

Die Adressen der Speicherbereiche und deren Zugriffsmöglichkeiten werden von den Modbus-Geräteherstellern, meist in Tabellenform zur Verfügung gestellt.

Beispiel – Auflistung: Zugriffsmöglichkeiten auf Variable (fiktive Modbus-Gerätehersteller-Angaben)

Modbus -Register Adressierung

	Adressierung		Anfangs- adresse	End- adresse	Zugriff	Beschreibung / Speicherbereich
	1 Bit	Register				
Prozess-daten	X ⁽¹⁾		0x0000	0x00CF	Read/Write	Prozessdaten-Interface. Phys. Eingänge Prozessabbild
		X ⁽²⁾	0x00D0	0x00FF	Read/Write	
	X ⁽³⁾		0x0100	0x01CF	Read-only	
		X ⁽⁴⁾	0x01D0	0x01FF	Read-only	
	X ⁽⁵⁾		0x0200	0x02FF	Read/Write	
	X ⁽⁶⁾		0x0300	0x03FF	Read/Write	
Diagnose		X ⁽⁷⁾	0x0400	0x040F	Read-only	Statusregister
		X ⁽⁸⁾	0x0410		Read-only	Prozessabbildlänge in Bit, analoge Ausgänge
		X	0x0411		Read-only	Prozessabbildlänge in Bit, analoge Eingänge
		X	0x0412		Read-only	Prozessabbildlänge in Bit, digitale Ausgänge
		X	0x0413		Read-only	Prozessabbildlänge in Bit, digitale Eingänge
Sonder- register		X ⁽⁹⁾	0x0420	0x042F	Read/Write	Watchdog-Register
		X ⁽¹⁰⁾	0x0430	0x043F	Read/Write	Fehler-Register
		X	0x0440	0x044F	Read/Write	Kommando-Register
		X	0x0450	0x045F	Read/Write	Interne-Register

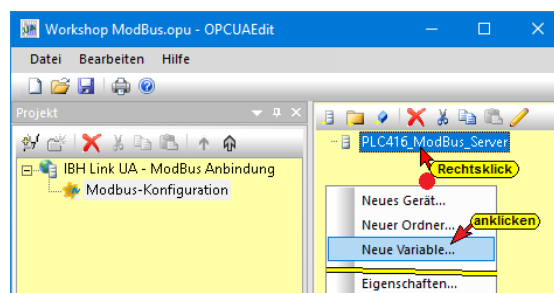
(nn) Als Beispiel für die Definition einer Variablen vorhanden.

Anmerkung:

Die Adressangaben in den Modbus-Gerätehersteller-Angaben sind oft in hexadezimaler Form. Diese Adressen sind in für die Eingabe in den IBH OPC UA Editors in eine dezimale Adresse umzuwandeln.

1.3 Modbus-Variable im IBH OPC UA Editors definieren

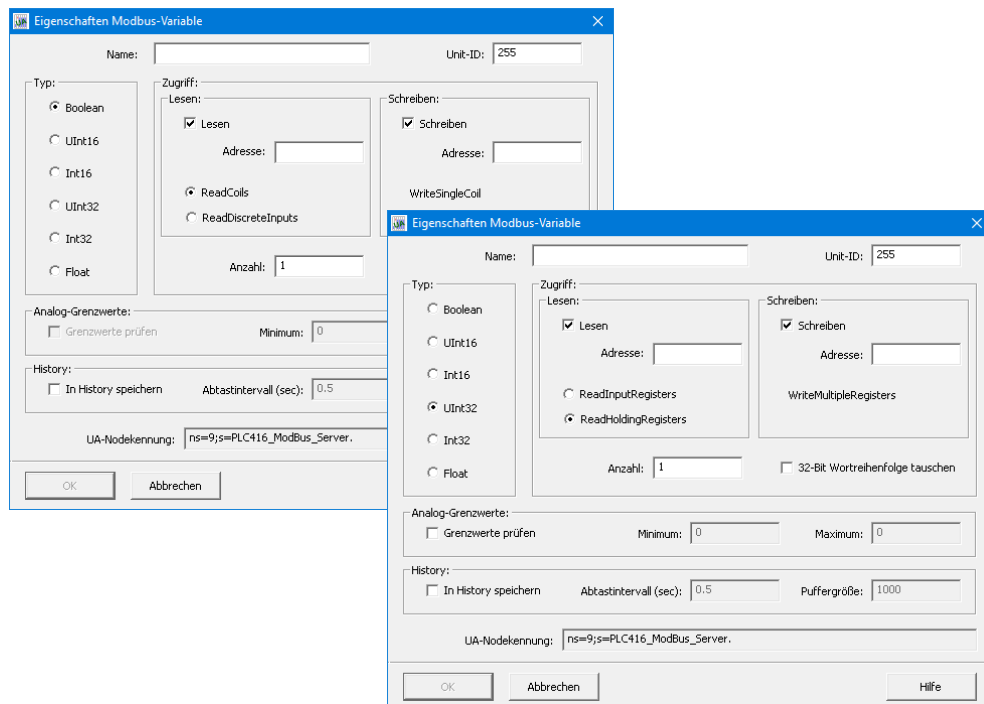
Mit einem Rechtsklick auf den Modbus Gerätenamen (PLC416_ModBus_Server) wird das Kontextmenü geöffnet.



Der Befehl **Neue Variable** öffnet das Dialogfeld **Eigenschaften Modbus-Variable**. In diesem Dialogfeld werden die Variablen, die

der OPC Server verbinden soll definiert. Dies können Schreib-Variable, Lese-Variable oder Schreib- / Lese-Variable sein.

1.3.1 Dialogfeld Eigenschaften Modbus-Variable



Modbus TCP- / RTU-Interface

Modbus-Geräte-Hersteller beschreiben in Tabellen die Modbus Feldbusknoten Funktionen. Aufgrund dieser Beschreibungen werden die Definitionen der Zugriffe auf Variable festgelegt.

Name (Variable)

Der Name ist frei wählbar, muss aber der OPC UA Spezifikation entsprechen (Buchstaben **A-Z**, **a-z**, Ziffern **0-9**, keine Umlaute, keine Leerzeichen, keine Punkte, kein Doppelpunkt. Als Sonderzeichen nur Unterline.

Name:

Unity-ID

Für Modbus TCP ist die Unity-ID = 255. Dies ist bei default eingestellt. Bei Modbus RTU ist die Adresse entsprechend der Slave-Adresse einzustellen.

Unit ID:

Typ

Boolean

Ein (1) Bit Information (**Bool**), die die Zustände TRUE (1) und FALSE (0) haben kann. Eine Variable des Typs **Boolean** belegt 1 Bit in einer Register-Adresse (16 Bit). Ein Array mit 16 Variablen des Typs **Boolean** belegen eine gesamte Register-Adresse (16 Bit).

UInt16

Positive 16 Bit Festpunktzahl (**Unidirectional Integer**) zwischen 0 und 65.535 (2^0 bis 2^{+16}). Eine Variable des Typs **UInt16** belegt eine Register-Adresse (16 Bit).

Int16

Positive- bzw. negative 16 Bit Festpunktzahl (**Integer**) zwischen -32.767 und +32.767 (-2^{+15} bis $2^{+15} - 1$). Eine Variable des Typs **Int16** belegt eine Register-Adresse (16 Bit).

UInt32

Positive 32 Bit Festpunktzahl (**Unidirectional Integer**) zwischen 0 und 4.294.967.295 (2^0 bis 2^{+32}). Eine Variable des Typs **UInt32** belegt zwei (16 Bit) Register-Adressen.

Int32

Positive- bzw. negative 32 Bit Festpunktzahl (**Integer**) zwischen -2.147.483.654 und +2.147.483.654 (-2^{+31} bis $2^{+31} - 1$). Eine Variable des Typs **UInt32** belegt zwei (16 Bit) Register-Adressen.

Float

Eine Variable vom Datentyp Float stellt eine gebrochene Zahl dar, die als 32-Bit-Gleitpunktzahl (REAL) vorhanden ist. Eine Variable des Typs **Float** belegt zwei (16 Bit) Register-Adressen.

Typ:

- Boolean
- UInt16
- Int16
- UInt32
- Int32
- Float

1.3.2 Adressen beim Zugriff Lesen / Schreiben

Anmerkung:

Die Adressangaben in den Modbus-Gerätehersteller-Angaben sind oft in hexadezimaler Form. Diese Adressen sind in für die Eingabe in den IBH OPC UA Editors in eine dezimale Adresse umzuwandeln.

Datentyp UInt16 und Int16

Es wird ein Wort-Adresse (16-Bit) für Zugriffe auf Variable eines Datentyps genutzt. Mit der Adresse 0 wird die Variable, die die ersten 16 Bits eines Datenbereichs belegen, angesprochen. Mit der Adresse 1 wird die Variable, die die zweiten 16 Bits eines Datenbereichs belegen, angesprochen. Die Adresse 3 spricht die dritte Variable (16-Bit) eines Datenbereichs an usw.

Datentyp UInt32, Int32 und Float

Es werden immer zwei Worte (2 x16-Bit = 32Bit) für den Zugriff auf Variable eines Datenbereichs benötigt. Mit der Adresse 0 wird die Variable, die erste 32 Bit Variable eines Datenbereichs angesprochen. Adresse 1 darf nicht angesprochen werden, da mit dieser Adresse das Low -Word der 32 Bit Variablen adressiert würde. Mit der Adresse 2 wird die Variable, die die beiden Worte 2 und Wort 3 im Datenbaustein belegen angesprochen. Die Adresse 4 adressiert die dritte 32-Bit Variable. usw.

Die Reihenfolge der beiden 16 Bit Worte, aus denen die Datentyp UInt32, Int32 und Float bestehen, sind 32-Bit Wortreihenfolge tauschen einstellbar.

- **High-Word** – Low-Word bzw. Low-Word – **High-Word**.

Datentyp Boolean

Es wird ein Bit-Adresse für Zugriff auf Bits in den Datenbereich genutzt.

- Adresse 0 spricht das Bit 0 im Bereich Wort-Adresse 0 an.
- Adresse 1 spricht das Bit 1 im Bereich Wort-Adresse 0 an.
- Adresse 16 spricht das Bit 0 im Bereich Wort-Adresse 1 an.
- Adresse 66 spricht das Bit 2 im Bereich Wort-Adresse 3 an.

1.4 Zugriff nur Lesen (Read Only)

Read Discrete Input (Read only – Bit-Zugriff)

Datentyp Boolean

Beispiel: – Fiktive Modbus-Tabelle (3)

Anfangs-Wort_Adresse 0111_{hex} = Bit
 Adresse 1110_{hex} = 4368_{dez} – Read-only –
 Bit-Zugriff – Phys. Eingänge Prozessabbild.
 Es sollen 7 Eingangsbits als OPC-Tags
 definiert werden.

Mit dieser Einstellung werden Variablen aus den Registern der **Discrete Inputs** adressiert, deren zusammenhängender Status von Digitaleingängen kommt. Die Adresse der ersten Variablen und die Anzahl der Variablen sind anzugeben.

Screenshot: Es werden 7 einzelne Bits ab Bitadresse 4368 des Speicherbereichs der physikalischen Eingänge gelesen.

Read Input Registers (Read only)

Alle Datentypen außer Boolean (Beispiel: Int16 – 16-Bit)

Beispiel: – Fiktive Modbus-Tabelle (4)

Anfangs-Register-Adresse $01E0_{\text{hex}} = 480_{\text{dez}}$
 – Read-only – Wort-Zugriff – Bereich **Read Input Registers**. Es sollen 4 Register als OPC-Tags definiert werden. Der zusammenhängende Inhalt von Digitaleingängen (Analogeingängen) wird adressiert. Die Adresse der ersten Variablen und die Anzahl der Variablen sind anzugeben.

Screenshot: Es werden 4 Festpunktzahlen ab Wort 480 des Bereichs der **Read Input Registers** gelesen.

Read Input Registers (Read only)

Alle Datentypen außer Boolean (Beispiel: UInt16 – 16-Bit)

Beispiel: – Fiktive Modbus-Tabelle (7)

Anfangs-Register-Adresse $400_{\text{hex}} = 1024_{\text{dez}}$
 – Read-only – Wort-Zugriff – Es sollen der Inhalt von 10 Statusregister als OPC-Tags adressiert werden.

Mit dieser Einstellung werden Variablen aus den Diagnosebereich (**Input Registers**) adressiert. Die Startregisteradresse und die Anzahl der Register sind anzugeben.

Screenshot: Es werden 10 Zahlen (Vorzeichenlos) mit je 16-bit ab Wort 1024 der Statusregister gelesen.

1.5 Zugriff Lesen und Schreiben (Read / Write)

Read Coils / Write Single Coil

Datentyp Boolean

Beispiel: – Fiktive Modbus-Tabelle (1)

Register-Adresse 0010_{hex} = Bit Adresse 100_{hex} = 256_{dez} – Read-Write – Bit-Zugriff – Phys. Eingänge Prozessabbild. Es soll 1 Eingangsbits als OPC-Tag definiert werden.

Mit dieser Einstellung werden Variablen aus den Registern der **Coils** adressiert. Die Adresse der Variablen und die Anzahl der Variablen (1) sind anzugeben.

Screenshot: Es wird 1 einzelnes Bit mit der Bitadresse 256 des Speicherbereichs der physikalischen Eingänge gelesen.

Read Coils

Datentyp Boolean

Wird eine solche Variable nur als Lese-Variable definiert, ist diese als OPC-Tag mit dem Status **Read** definiert.

Beispiel: – Fiktive Modbus-Tabelle (1)

Register-Adresse 0014_{hex} = Bit Adresse 140_{hex} = 320_{dez} – Read – Bit-Zugriff – Phys. Eingänge Prozessabbild. Es soll 8 Eingangsbits als OPC-Tag definiert werden.

Screenshot: Es werden 8 einzelnes Bit ab der Bitadresse 320 des Speicherbereichs der physikalischen Eingänge gelesen.

Read Coils / Write Multiple Coils

Datentyp Boolean

Beispiel: – Fiktive Modbus-Tabelle (6)

Register-Adresse 0310_{hex} = Bit Adresse 3100_{hex} = 12544_{dez} – Read-Write – Bit-Zugriff – Phys. Eingänge Prozessabbild.

Es soll 12 Eingangsbits als OPC-Tag definiert werden.

Mit dieser Einstellung werden Variablen aus den Registern der **Coils** adressiert, deren Inhalt einzelne Bits widerspiegeln. Dies können einzelnen Ausgänge wie auch einzelne Eingänge sein. Die Adresse der ersten spezifizierten Variablen und die Anzahl der Variablen sind anzugeben.

Screenshot: Es werden 12 einzelne Bits ab der Bitadresse 12544 des Speicherbereichs der physikalischen Ausgänge definiert.

Read Holding Registers / Write Single Register

(Alle Datentypen außer Boolean)

Beispiel: – Fiktive Modbus-Tabelle (2) – Datentyp INT16 (Festpunktzahl).

Register-Adresse 00D0_{hex} = 208_{dez} – Read-Write – Wort-Zugriff – Phys. Eingänge Prozessabbild.

Es soll eine (1) Festpunktzahl als OPC-Tag definiert werden.

Mit dieser Einstellung werden Variablen aus den **Holding Registern** adressiert, deren Inhalt einzelne Register

widerspiegeln. Dies kann z.B ein Analogeingang sein. Die Adresse der ersten Variablen und die Anzahl (1) der Variablen sind anzugeben.

The screenshot shows a configuration window titled 'RW_Holding_Register_Single_Reg_Int'. It is divided into two main sections: 'Lesen' (Read) and 'Schreiben' (Write). In the 'Lesen' section, the 'ReadHoldingRegisters' radio button is selected. In the 'Schreiben' section, the 'WriteSingleRegister' radio button is selected. Both sections have an 'Adresse' field set to 208. At the bottom, there is an 'Anzahl' field set to 1. A checkbox for '32-Bit Wortreihenfolge tauschen' is unchecked.

Screenshot: Es wird ein eine Festpunktzahl von der Wortadresse 208 des Speicherbereichs der physikalischen Ausgänge definiert.

Read Holding Registers / Write Multiple Registers

(Alle Datentypen außer Boolean)

Beispiel: – Fiktive Modbus-Tabelle (5) – Datentyp INT16 (Festpunktzahl).

Register-Adresse 0210_{hex}
= 528_{dez} – Read-Write –
Wort-Zugriff –
Phys. Ausgänge
Prozessabbild.

Es sollen 9 Festpunktzahlen als OPC-Tag definiert werden.

The screenshot shows a configuration window titled 'RW_Holding_Register_Multiple_Reg_Int'. It is divided into two main sections: 'Lesen' (Read) and 'Schreiben' (Write). In the 'Lesen' section, the 'ReadHoldingRegisters' radio button is selected. In the 'Schreiben' section, the 'WriteMultipleRegisters' radio button is selected. Both sections have an 'Adresse' field set to 528. At the bottom, there is an 'Anzahl' field set to 9. A checkbox for '32-Bit Wortreihenfolge tauschen' is unchecked.

Mit dieser Einstellung werden Variablen aus den **Holding Registern** adressiert, deren Inhalt einzelne Register widerspiegeln. Dies können einzelnen Ausgänge (Analogausgänge) sein. Die Adresse der ersten spezifizierten Variablen und die Anzahl der Variablen sind anzugeben.

Screenshot: Es werden 9 Festpunktzahl ab der Wortadresse 528 des Speicherbereichs der physikalischen Ausgänge als OPC-Tags definiert.

Read Holding Registers

(Alle Datentypen außer Boolean)

Wird eine solche Variable nur als Lese-Variable definiert, ist diese als OPC-Tag mit dem Status **Read** definiert.

Beispiel: – Fiktive Modbus-Tabelle (8) –
Datentyp UINT16 vorzeichenlose Zahl.

Register-Adresse $0450_{\text{hex}} = 1104_{\text{dez}}$ – Read-Write – Wort-Zugriff – Sonderregister.

Es sollen 6 vorzeichenlose Zahlen als OPC-Tag definiert werden.

Mit dieser Einstellung werden Variablen aus den **Holding Registern** adressiert, deren Inhalt einzelne Register widerspiegeln. Die Adresse der ersten spezifizierten Variablen und die Anzahl der Variablen sind anzugeben.

Screenshot: Es werden 6 Zahlen (Vorzeichenlos) ab der Wortadresse 1104 des Speicherbereichs der Sonderregister als OPC-Tags definiert.

Read Holding Registers / Write Multiple Registers

(Datentypen UInt32, Int32, Float)

Beispiel: – Datentyp Float (Gleitpunktzahl).

Register-Adresse $0460_{\text{hex}} = 1120_{\text{dez}}$ – Read-Write – Wort-Zugriff – Phys. Ausgänge Prozessabbild.

Es sollen 5 Gleitpunktzahlen als OPC-Tag definiert werden.

Mit dieser Einstellung werden Variablen aus den **Holding**

Registern adressiert, deren Inhalt einzelne Register widerspiegeln.

Dies können einzelnen Ausgänge (Analogausgänge) sein.

Die Adresse der ersten spezifizierten Variablen und die Anzahl der Variablen sind anzugeben.

Screenshot: Es werden 5 Gleitpunktzahlen ab der Wortadresse 1120 im **Holding Registern** als OPC-Tags definiert.

Anmerkung:

Daten vom Typ **UInt32**, **Int32** und **Float** belegen zwei (2) 16 Bit-Worte (32 Bit). Die Reihenfolge der beiden 16 Bit Worte, aus denen diese Datentypen bestehen, sind einstellbar.

- **High-Word** – Low-Word bzw. Low-Word – **High-Word**

1.5.1 Analoge-Grenzwerte

Analoge Grenzwerte können vorgegeben werden.

Analog-Grenzwerte:
 Grenzwerte prüfen Minimum: 100.0 Maximum: 10000.0

1.5.2 History

Während **OPC Data Access** Zugriff auf Daten in Echtzeit ermöglicht, unterstützt **OPC Historical Data Access**, auch OPC HDA benannt, den Zugriff auf bereits gespeicherte Daten.

Die Aktivierung einer Variablen als historischen Daten sowie Abtastintervall und Anzahl der Werte (Puffergröße) erfolgt über das Dialogfeld.

History:
 In History speichern Abtastintervall (sec): 0.5 Puffergröße: 1000

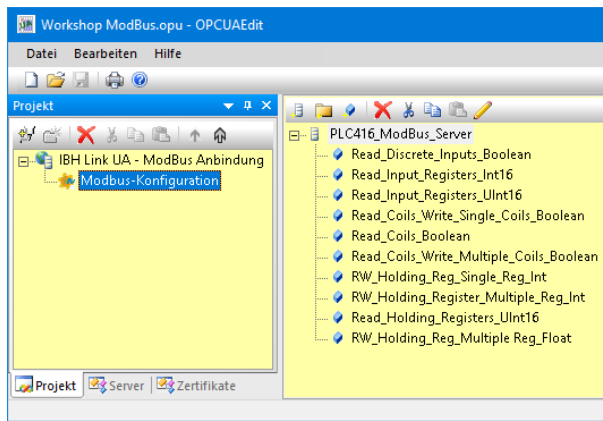
1.5.3 OPC UA Nodeerkennung

Die OPC UA Nodename einer Variablen wird im Dialogfeld angezeigt.

UA-Nodeerkennung: ns=9;s=PLC416_Yellow.DW7_RW_Holding_Reg_Real

1.6 Modbus-Konfiguration zum OPC UA Server (IBH Link UA) übertragen

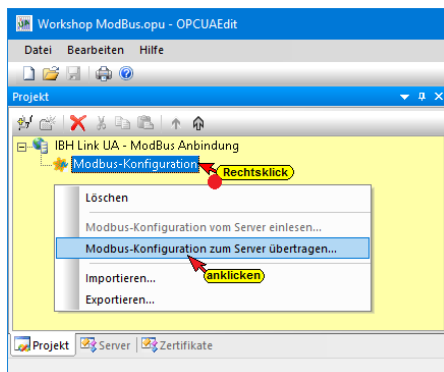
Die als OPC-Tags definierten Variablen werden angezeigt.



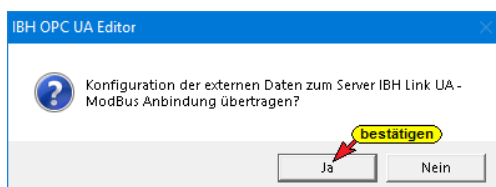
Name	Datentyp	Zugriff	R-Adresse	W-Adresse	Anzahl	Nodename
Read_Discrete_Inputs_Boolean	Boolean	R	4368		7	PLC416_ModBus_Server.Read_Discrete_Inputs_Boolean
Read_Input_Registers_Int16	Int16	R	480		4	PLC416_ModBus_Server.Read_Input_Registers_Int16
Read_Input_Registers_UInt16	UInt16	R	1024		10	PLC416_ModBus_Server.Read_Input_Registers_UInt16
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

Sind alle Modbus-Variablen als OPC UA Tags definiert, kann die Modbus-Konfiguration zum OPC UA Server übertragen werden.

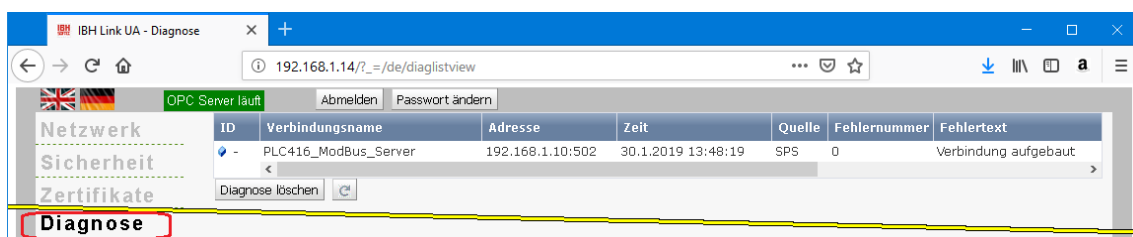
Ein Rechtsklick auf **Modbus-Konfiguration** öffnet das Kontextmenü.



Der Befehl zur Übertragung der **Modbus-Konfiguration** muss bestätigt werden.



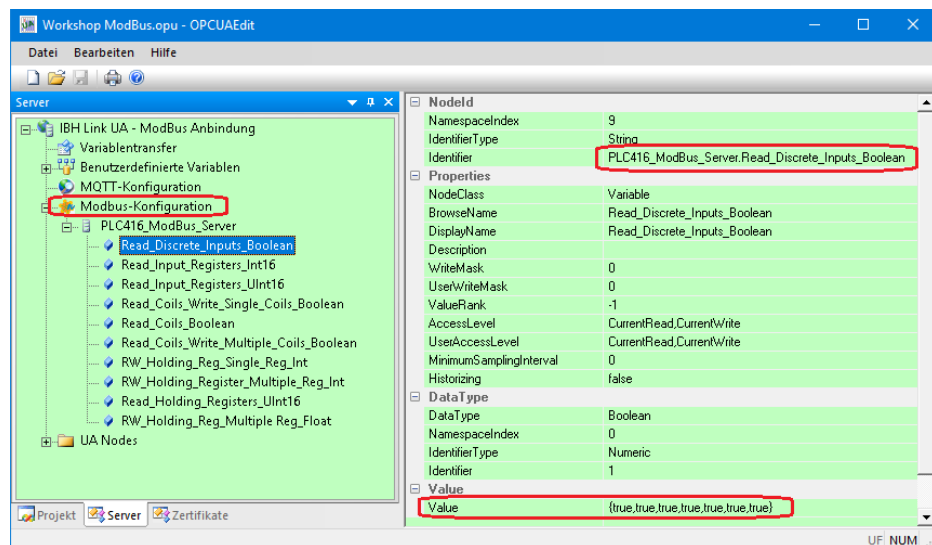
Die Übertragene **Modbus-Konfiguration** wird unter Diagnose im IBH Link UA Browser-Fenster angezeigt.



1.7 IBH OPC UA Editor Server-Fenster

Eine erfolgreich an den OPC UA Server übertragene **Modbus-Konfiguration** kann im Server-Fenster online angezeigt werden.

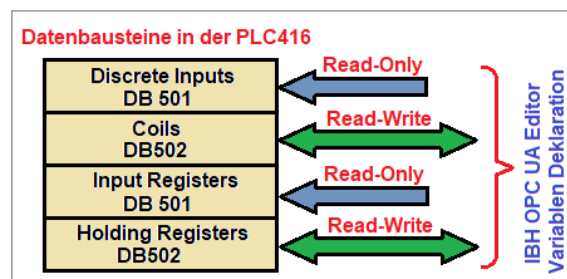
Im linken Server-Fenster sind die Variablen (**Modbus-Konfiguration, Modbus-Gerät, OPC-Tag**) aufgelistet. Mit Anklicken einer Variablen werden im rechten Server-Fenster die Variablen-Definitionen mit dem Status angezeigt. Der Status dieses OPC-Tags wird laufend erneuert.



1.8 Modbus Anbindung – Beispiele

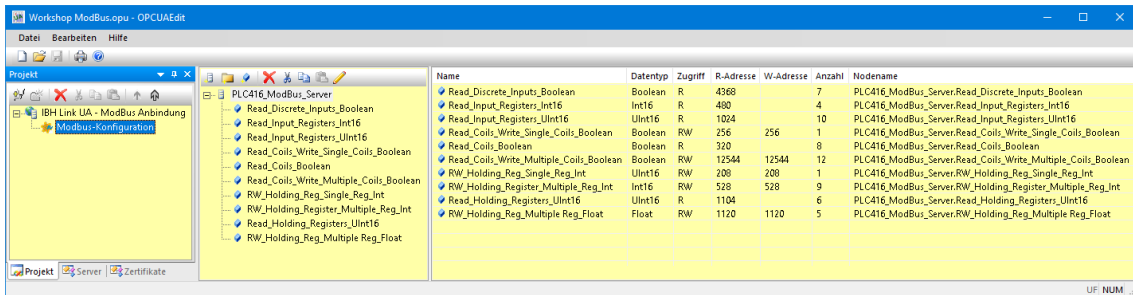
Die IBH SoftSPS PLC416 hat die Möglichkeit einer Modbus Anbindung. Im Beispiel werden Variable als OPC-Tags definiert. Diese Modbus-Konfiguration wird an den IBH Link UA übertragen und die Variablen im **UAExpert Client Programm** angezeigt.

Die Datenbausteine DB501 (**DBIn – Read**) und DB502 (**DBOut – Read / Write**) sind in der PLC416 angelegt.

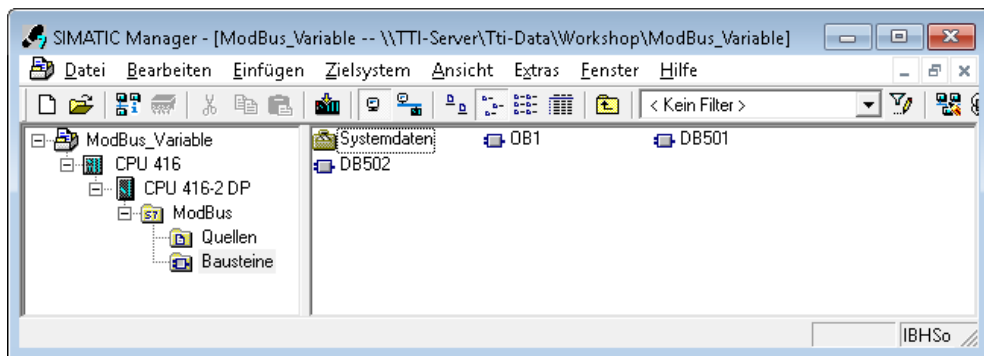


Es werden die OPC-Tags aus dem IBH UA Editor Programm **Workshop ModBus.opu**. Die Datei ist mit dem IBH UA Editor zu

öffnen und die Modbus-Konfiguration an den IBH Link UA zu übertragen.

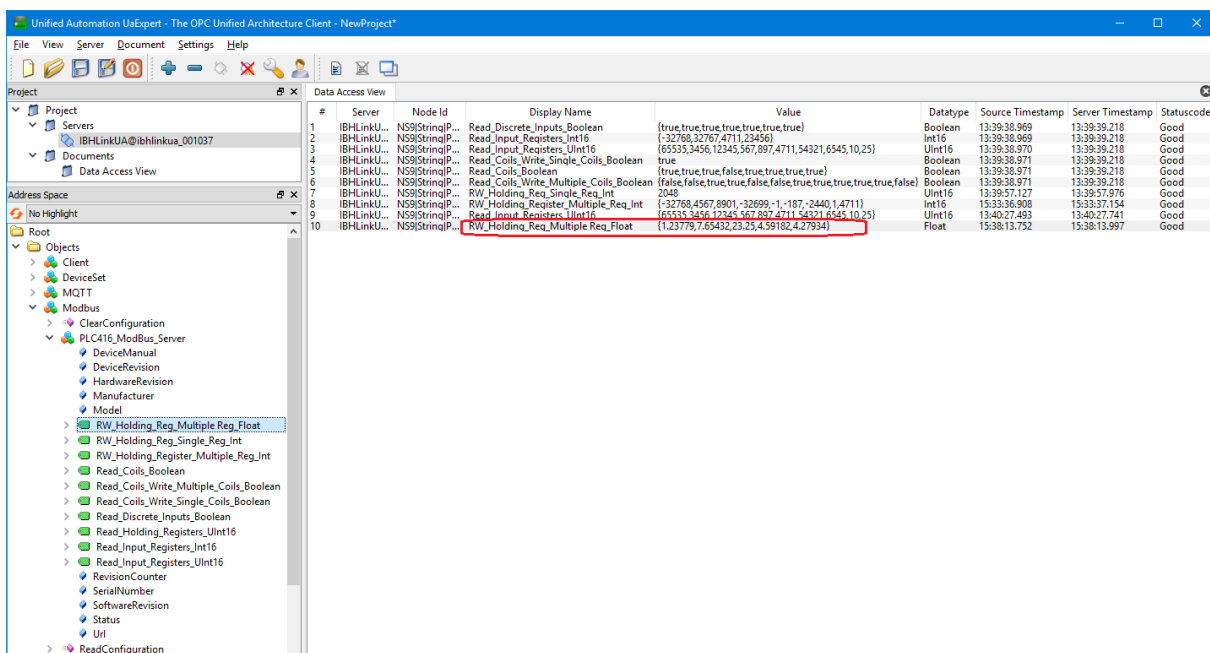


Das STEP 7 SPS Programm ist zu öffnen und die Bausteine mit den Systemdaten ist in die Soft-SPS PLC416 zu laden.



1.8.1 Unified Automation UaExpert - The OPC Unified Architecture Client

Im UaExpert – Programm-Fenster werden die vom IBH OPC UA Editor übertragenen **OPC-Tags** und die dazugehörigen **UA Nodes** aufgelistet.



Über die Diagnose des PLC416 können die Werte der Variablen verändert werden.

- Beispiel 1: Read_Discrete_Inputs_Boolean**, (Read only, Bit-Zugriff, Datentyp Boolean). Anfangs-Wort-Adresse $0111_{\text{hex}} = 273_{\text{dez}}$, Bit Adresse $1110_{\text{hex}} = 4368_{\text{dez}}$ – Wort-Zugriff – DB501 – DW 546 (Byte 546). 7 Bits als OPC-Tags definiert.
- Beispiel 2: Read_Input_Registers_Int16**, (Read only), Alle Datentypen außer Boolean (Int16), Anfangs-Wort-Adresse $01E0_{\text{hex}} = 480_{\text{dez}}$ Wort-Zugriff – DB501 – DW 960 bis DW 966. 4 Festpunktzahlen als OPC-Tags definiert.
- Beispiel 3: Read_Input_Registers_UInt16**, (Read only), Alle Datentypen außer Boolean (UInt16 – vorzeichenlose Zahl), Anfangs-Wort-Adresse $400_{\text{hex}} = 1024_{\text{dez}}$ Wort-Zugriff – DB501 – DW 2048 bis DW 2066. 10 vorzeichenlose Zahlen als OPC-Tags definiert.
- Beispiel 4: Read_Coils_Write_Single_Coils_Boolean**, (Read-Write, Bit-Zugriff, Datentyp Boolean), Anfangs-Wort-Adresse $0010_{\text{hex}} = 16_{\text{dez}}$ Bit Adresse $100_{\text{hex}} = 256_{\text{dez}}$ – Wort-Zugriff – DB502 – DW 32 (Byte 32). 1 Bit als OPC-Tags definiert.
- Beispiel 5: Read_Coils_Boolean**, (Nur Read, Bit-Zugriff, Datentyp Boolean), Anfangs-Wort-Adresse $0014_{\text{hex}} = 20_{\text{dez}}$ Bit Adresse $140_{\text{hex}} = 256_{\text{dez}}$ Wort-Zugriff – DB502 – DW 40 (Byte 40). 8 Bit als OPC-Tags definiert.
- Beispiel 6: Read_Coils_Write_Multiple_Coils_Boolean**, (Read-Write, Bit-Zugriff, Datentyp Boolean), Anfangs-Wort-Adresse $0310_{\text{hex}} = 784_{\text{dez}}$, Bit Adresse $3100_{\text{hex}} = 12544_{\text{dez}}$ – Wort-Zugriff – DB502 – DW 1568. 12 Bit als OPC-Tags definiert.
- Beispiel 7: RW_Holding_Reg_Single_Reg_Int**, (Read-Write, Datentyp INT16 Festpunktzahl), Anfangs-Wort-Adresse $00D0_{\text{hex}} = 208_{\text{dez}}$, Wort-Zugriff – DB502 – DW 416. Eine (1) Festpunktzahl als OPC-Tag definiert.
- Beispiel 8: RW_Holding_Register_Multiple_Reg_Int**, (Read-Write, Datentyp INT16 Festpunktzahl), Anfangs-Wort-Adresse $0210_{\text{hex}} = 528_{\text{dez}}$, Wort-Zugriff – DB502 – DW 1056 bis DW 1072. 9 Festpunktzahl als OPC-Tag definiert.
- Beispiel 9: RW_Holding_Register_UInt16**, (Read-Write, Datentyp UINT16 vorzeichenlose Zahl), Anfangs-Wort-Adresse $0250_{\text{hex}} = 1104_{\text{dez}}$, Wort-Zugriff – DB502 – DW 2208 bis DW 2218. 6 vorzeichenlose Zahlen als OPC-Tag definiert.
- Beispiel 10: RW_Holding_Reg_Multiple_Reg_Float**, (Read-Write, Datentyp FLOAT Gleitpunktzahl), Anfangs-Wort-Adresse $0460_{\text{hex}} = 1120_{\text{dez}}$, 2-Wort-Zugriff – DB502 – DW 2240 bis DW 2258. 5 Gleitpunktzahlen als OPC-Tag definiert.