FAST MOVING TECHNOLOGY

Modbus Configuration CS8C / CS9

Technical documentation

White paper

TSS.0000.016.04-A - 23/06/2020
Modbus Configuration

A "readme.pdf" document may be delivered on the robot's DVD. It contains the documentation addenda and errata.

Table of Contents

History

Revision	Modification	Date (yyyy-mm-dd)	By
A	Initial White Paper release (corresponding to the revision 2 of the previous document Modbus Configuration)	2020-05-29	A.JAFFRE
B			
C			

Version

That document has been tested with:

- SRC : s8.10.2
- Safety : 1.003 / SafePMT 3.0.0.28
- SYCON.net : 1.500.180608.23664 (SRS 2019.7.1)

Keyword

Fieldbus, Modbus, Configuration, Client, Server, Input, Output, CS9, CS8C, Word, Bit, Address, Registers

1 Preliminary

DANGER

Instructions drawing the reader's attention to the risks of accidents that could lead to serious bodily harm if the steps shown are not complied with. In general, this type of indication describes the potential danger, its possible effects and the necessary steps to reduce the danger.
It is essential to comply with the instructions to ensure personal safety..

SAFETY

Instructions drawing the reader's attention that its responsibility is engaged if the steps shown are not complied with.
It is essential to comply with the instructions to maintain the robot safety level.

Caution

Instructions directing the reader's attention to the risks of material damage or failure if the steps shown are not complied with. It is essential to comply with these instructions to ensure equipment reliability and performance levels.

ELECTRICAL risk

Instructions drawing the reader's attention to the risks of electrical shock.
It is essential to comply with the instructions to ensure personal safety..

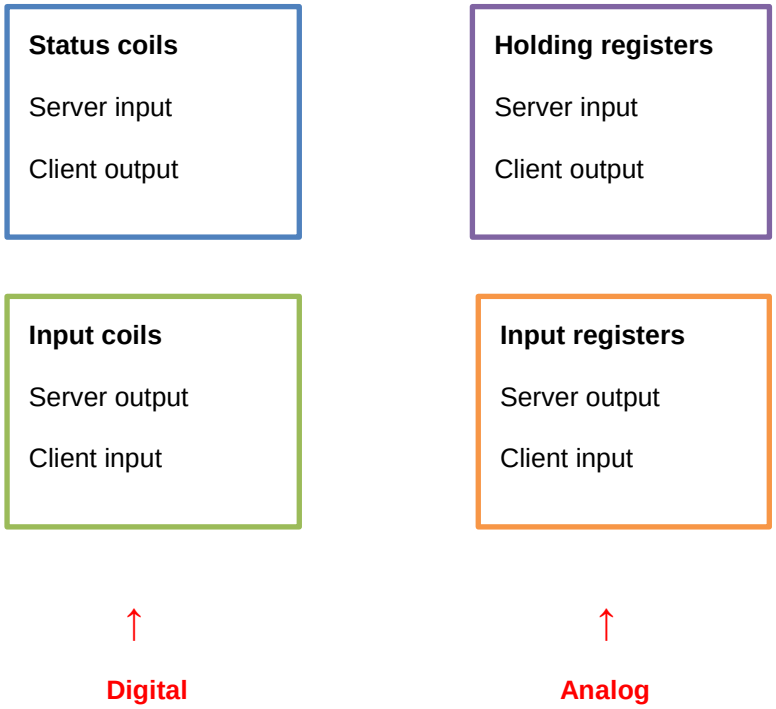
Information

Supplies further information, or underlines a point or an important procedure. This information must be memorized to make it easier to apply and ensure correct sequencing of the operations described.

2 Modbus simplified

Below explanation are very simplified.

In Modbus, we have 4 databases, depending on element type:



Modbus name	Prefix	Address (max)	For a server (slave)	For a client (master)	FC
Status coil	**0**	**0** 00001 to **0** 65536	Input (R)	Output (R/W)	1, 5, 15
Input coil	**1**	**1** 00001 to **1** 65536	Output (R/W)	Input (R)	2
Input register	**3**	**3** 00001 to **3** 65536	Output (R/W)	Input (R)	4
Holding register	**4**	**4** 00001 to **4** 65536	Input (R)	Output (R/W)	3, 6, 16, 23

Sometimes, address evolve from 0 to 65535 instead of 1 to 65536.

Coils are single bits (0 or 1), digital inputs/outputs while registers are 16 bits unsigned integer (0 to 65535), analog inputs/outputs.

FC are the function codes used to query the server.

Each equipment can choose what he wants to support and in which quantity. This is where the fun start.

3 Exchange table

Before starting to make a modbus configuration, you have to write your exchange table in order to know the mapping you need between server input/output and client input/output.

In order clarify; we will always speak the same way regarding input and output.

In the following explanation, we will always express them from the master point of view (client in Modbus terminology)

Ideally, if equipment's are organized the same way, it looks like that:

Digital input						
Client (master)			Server (slave)			
Word	Bit	Input name	Output name	Address	Bit	Word
1	1	DI_01	DO_01	100001	1	1
	2	DI_02	DO_02	100002	2	
	3	DI_03	DO_03	100003	3	
	4	DI_04	DO_04	100004	4	
	5	DI_05	DO_05	100005	5	
	6	DI_06	DO_06	100006	6	
	7	DI_07	DO_07	100007	7	
	8	DI_08	DO_08	100008	8	
	9	DI_09	DO_09	100009	9	
	10	DI_10	DO_10	100010	10	
	11	DI_11	DO_11	100011	11	
	12	DI_12	DO_12	100012	12	
	13	DI_13	DO_13	100013	13	
	14	DI_14	DO_14	100014	14	
	15	DI_15	DO_15	100015	15	
	16	DI_16	DO_16	100016	16	
2	17	DI_17	DO_17	100017	17	2
	18	DI_18	DO_18	100018	18	
	19	DI_19	DO_19	100019	19	
	20	DI_20	DO_20	100020	20	
	21	DI_21	DO_21	100021	21	
	22	DI_22	DO_22	100022	22	
	23	DI_23	DO_23	100023	23	
	24	DI_24	DO_24	100024	24	
	25	DI_25	DO_25	100025	25	
	26	DI_26	DO_26	100026	26	
	27	DI_27	DO_27	100027	27	
	28	DI_28	DO_28	100028	28	
	29	DI_29	DO_29	100029	29	
	30	DI_30	DO_30	100030	30	
	31	DI_31	DO_31	100031	31	
	32	DI_32	DO_32	100032	32	

Digital output						
Client (master)			Server (slave)			
Word	Bit	Output name	Input name	Address	Bit	Word
1	1	DO_01	DI_01	000001	1	1
	2	DO_02	DI_02	000002	2	
	3	DO_03	DI_03	000003	3	
	4	DO_04	DI_04	000004	4	
	5	DO_05	DI_05	000005	5	
	6	DO_06	DI_06	000006	6	
	7	DO_07	DI_07	000007	7	
	8	DO_08	DI_08	000008	8	
	9	DO_09	DI_09	000009	9	
	10	DO_10	DI_10	000010	10	
	11	DO_11	DI_11	000011	11	
	12	DO_12	DI_12	000012	12	
	13	DO_13	DI_13	000013	13	
	14	DO_14	DI_14	000014	14	
	15	DO_15	DI_15	000015	15	
	16	DO_16	DI_16	000016	16	
2	17	DO_17	DI_17	000017	17	2
	18	DO_18	DI_18	000018	18	
	19	DO_19	DI_19	000019	19	
	20	DO_20	DI_20	000020	20	
	21	DO_21	DI_21	000021	21	
	22	DO_22	DI_22	000022	22	
	23	DO_23	DI_23	000023	23	
	24	DO_24	DI_24	000024	24	
	25	DO_25	DI_25	000025	25	
	26	DO_26	DI_26	000026	26	
	27	DO_27	DI_27	000027	27	
	28	DO_28	DI_28	000028	28	
	29	DO_29	DI_29	000029	29	
	30	DO_30	DI_30	000030	30	
	31	DO_31	DI_31	000031	31	
	32	DO_32	DI_32	000032	32	

Analog input				
Client (master)		Server (slave)		
Word	Input name	Output name	Address	Word
1	AI_01	AO_01	400001	1
2	AI_02	AO_02	400002	2
3	AI_03	AO_03	400003	3
4	AI_04	AO_04	400004	4

Analog output				
Client (master)		Server (slave)		
Word	Output name	Intput name	Address	Word
1	AO_01	AI_01	300001	1
2	AO_02	AI_02	300002	2
3	AO_03	AI_03	300003	3
4	AO_04	AI_04	300004	4

4 CS8C versus CS9 Modbus server

CS8C and CS9 Modbus server configuration are not compatible.

You will have to rebuild/adapt your configuration on both side (controller and equipment).

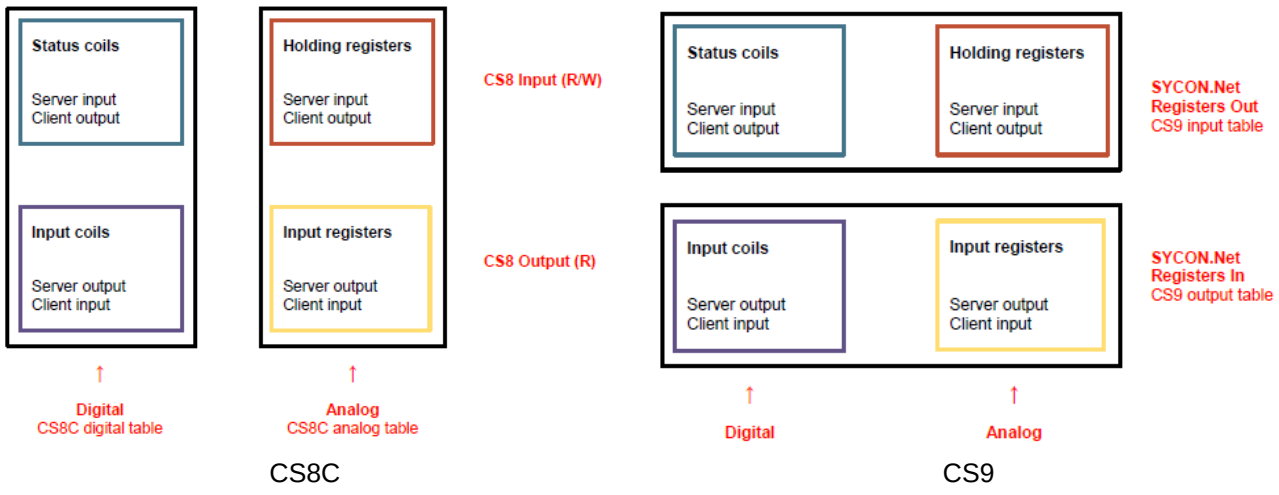
Modbus specification define only bit and word data types (coils/discrete input, holding register/input register). Other data types are manufacturer specific.

Modbus specification gives manufacturer a lot of leeway on how to manage/store data.

Some of CS8C and CS9 Modbus server configuration difference are:

- Different way of storing data:
  - CS8C use one table for bit and one for word
  - CS9 use one table for input and one table for output
  - CS9 use a different bit arrangement (8 to 15 and then 0 to 7)
- Different way of managing others data types: the two word used for float / real32 representation are reversed between CS8C and CS9.

4.1 Data storage illustration



4.2 Data types illustration

Here below a simple comparison done with Modbus doctor with the different data types. As you can see, to get the same value, Modbus doctor configuration need to be different between CS8C and CS9.

Modbus doctor (https://www.kscada.com/modbusdoctor.html) is a freeware, not supported by STÄUBLI, used only for easy illustration of the difference.

CS8

		9						Unsigned values		Modbus doctor configuration				
Type	**Quantity**	**Nb register**	**Value**	**Byte 0**	**Byte 1**	**Byte 2**	**Byte 3**	**Register 0**	**Register 1**	**swap byte**	**swap word**	**unsigned**	**Display mode**	**Value**
bit	16	1	15 0 10000000 00000010 32770	7 0 00000010 2	15 8 10000000 128			640						
word	1	2	65534	255	254			65534				X	16 bits word	65534
			65535	255	255			65535				X	16 bits word	65535
dword	1	2	4294967294	255	254	255	255	65534	65535			X	32 bits word	4294967294
float	2	4	-3,1415	14	86	192	73	3670	49225				32 bits float	-3,1415
			3,1415	14	86	64	73	3670	16457				32 bits float	3,1415

CS9

		20						Unsigned values		Modbus doctor configuration				
Type	**Quantity**	**Nb register**	**Value**	**Byte 0**	**Byte 1**	**Byte 2**	**Byte 3**	**Register 0**	**Register 1**	**swap byte**	**swap word**	**unsigned**	**Display mode**	**Value**
bit	16	1	15 0 10000000 00000010 32770	7 0 00000010 2	15 8 10000000 128			640						
byte	2	1	1	1				511					8 bit byte	1
			255	255									8 bit byte	255
signed8	2	1	-128	128				32895					8 bit byte	128
			127	127									8 bit byte	127
unsigned8	2	1	1	1				511					8 bit byte	1
			255	255									8 bit byte	255
word	1	1	65534	255	254			65534				X	16 bits word	65534
signed16	2	2	-32768	128	0			32768					16 bits word	-32768
			32767	127	255			32767					16 bits word	32767
unsigned16	1	1	65534	255	254			65534				X	16 bits word	65534
dword	1	2	4294967294	255	255	255	254	65535	65534		X	X	32 bits word	4294967294
signed32	2	4	-2147483648	128	0	0	0	32768	0		X		32 bits word	-2147483648
			2147483647	127	255	255	255	32767	65535		X		32 bits word	2147483647
unsigned32	1	2	4294967294	255	255	255	254	65535	65534		X	X	32 bits word	4294967294
real32	2	4	-3,1415	192	73	14	86	49225	3670		X		32 bits float	-3,1415
			3,1415	64	73	14	86	16457	3670		X		32 bits float	3,1415

Sign not taken into account by Modbus doctor on 8 bit byte

5 Modbus server on CS8C

Coils are merged internally in a single table with first the Status coils and then the Input coils.

From the CS8C point of view, first the digital input than the digital output.

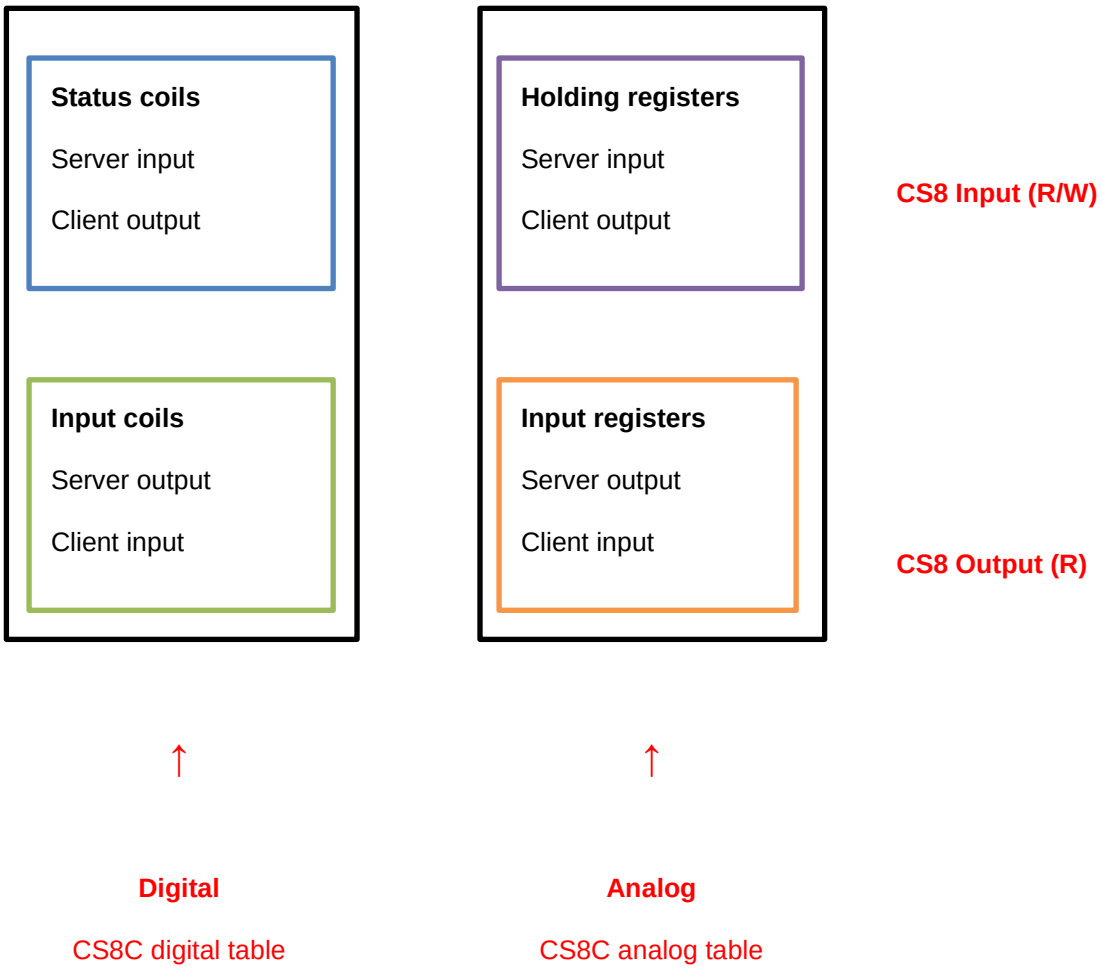
From the master point of view, first the digital output than the digital input.

Registers are merged internally in a single table with first the Holding registers and then the Input registers.

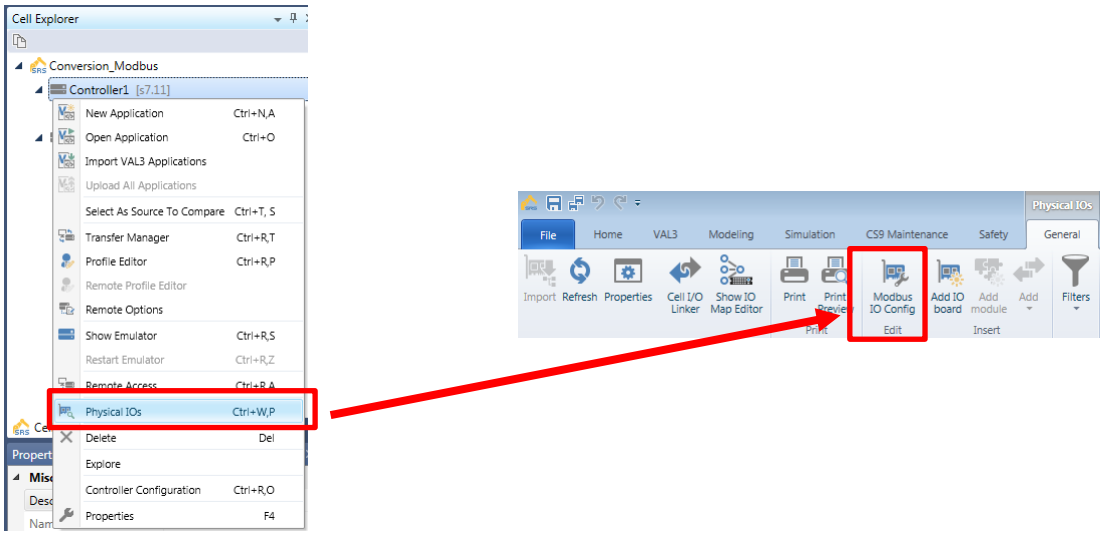
From the CS8C point of view, first the analog input than the analog output.

From the master point of view, first the analog output than the analog input.

Depending on what client equipment support, it could be interesting to have coils quantity as a multiple of 16 to prevent trouble with system managing only words.

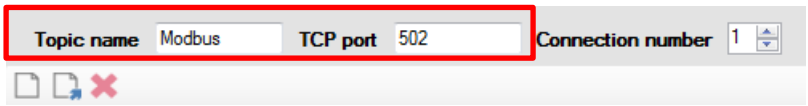


In SRS, select the controller in the cell explorer, make a right click on it and select “Physical IOs”, then “Modbus IO config”.



Do not change “Topic name” and “TCP port”.

Connection number can be adjust if needed (more than one client, client disconnections …)



Add a new item at the bottom of the list

Add a new item above the selected item

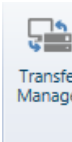
Delete the selected item

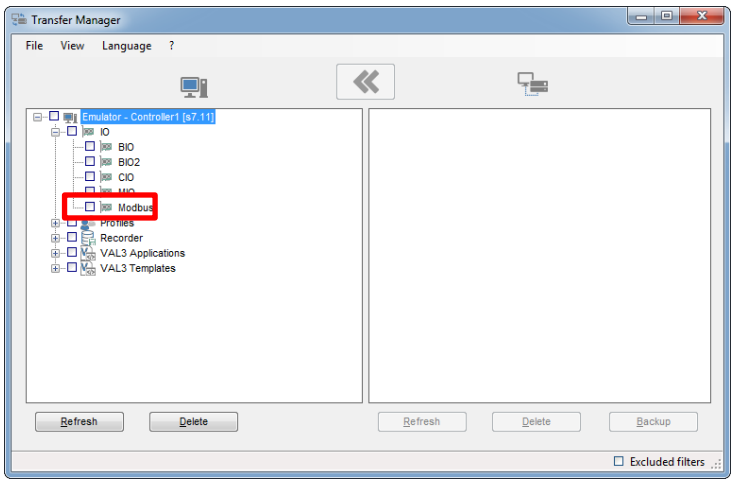
Do the proper configuration

Save it as “modbus.xml” (SRS place you in the right folder)

With “Transfer manager”, send the “Modbus” configuration to the controller.

Reboot the controller in order to take into account the new configuration.





5.1 Example with a PLC/screen configured as a client

In this example, the client has 12 digital outputs, 10 digital inputs, 2 analog outputs, 4 analog inputs.

In order to prevent issues we will round up digital outputs and inputs to the next 16 multiple.

PLC / screen, client (master)				CS8C, server (slave)				
	Word	Bit	Output name	Input name	Address	Bit	Word	
Digital output	1	1	DO_01	DI_01	000001	1	1	Digital input
		2	DO_02	DI_02	000002	2		
		3	DO_03	DI_03	000003	3		
		4	DO_04	DI_04	000004	4		
		5	DO_05	DI_05	000005	5		
		6	DO_06	DI_06	000006	6		
		7	DO_07	DI_07	000007	7		
		8	DO_08	DI_08	000008	8		
		9	DO_09	DI_09	000009	9		
		10	DO_10	DI_10	000010	10		
		11	DO_11	DI_11	000011	11		
		12	DO_12	DI_12	000012	12		
		13	DO_free_01	DI_free_01	000013	13		
		14	DO_free_02	DI_free_02	000014	14		
		15	DO_free_03	DI_free_03	000015	15		
		16	DO_free_03	DI_free_04	000016	16		
Digital input	2	1	DI_01	DO_01	100017	17	2	Digital output
		2	DI_02	DO_02	100018	18		
		3	DI_03	DO_03	100019	19		
		4	DI_04	DO_04	100020	20		
		5	DI_05	DO_05	100021	21		
		6	DI_06	DO_06	100022	22		
		7	DI_07	DO_07	100023	23		
		8	DI_08	DO_08	100024	24		
		9	DI_09	DO_09	100025	25		
		10	DI_10	DO_10	100026	26		
		11	DI_free_01	DO_free_01	100027	27		
		12	DI_free_02	DO_free_02	100028	28		
		13	DI_free_03	DO_free_03	100029	29		
		14	DI_free_04	DO_free_04	100030	30		
		15	DI_free_05	DO_free_05	100031	31		
		16	DI_free_06	DO_free_06	100032	32		

PLC / screen, client (master)			CS8C, server (slave)			
	Word	Input name	Output name	Address	Word	
Analog output	1	AO_01	AI_01	300001	1	Analog input
	2	AO_02	AI_02	300002	2	
Analog input	1	AI_01	AO_01	400003	3	Analog output
	2	AI_02	AO_02	400004	4	
	3	AI_03	AO_03	400005	5	
	4	AI_04	AO_04	400006	6	

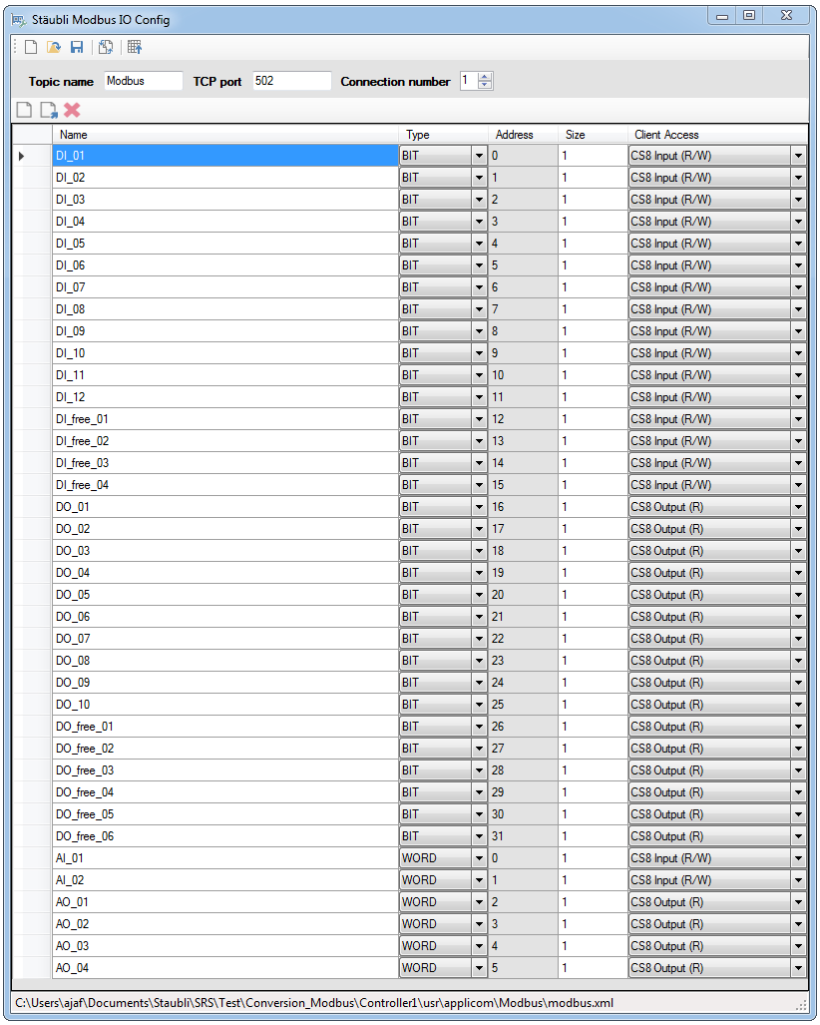
Start by adding 16 items that you configure as “bit” and “CS8 input (R/W)”. Name them properly; they will correspond to client digital outputs (12 digital outputs and 4 free to reach 16).

Add 16 items that you configure as “bit” and “CS8 output (R)”. Name them properly; they will correspond to client digital inputs (10 digital inputs and 6 free to reach 16).

Add 2 items that you configure as “word” and “CS8 input (R/W)”. Name them properly; they will correspond to client analog outputs (2 analog outputs).

Add 4 items that you configure as “word” and “CS8 output (R)”. Name them properly; they will correspond to client analog inputs (4 analog inputs).

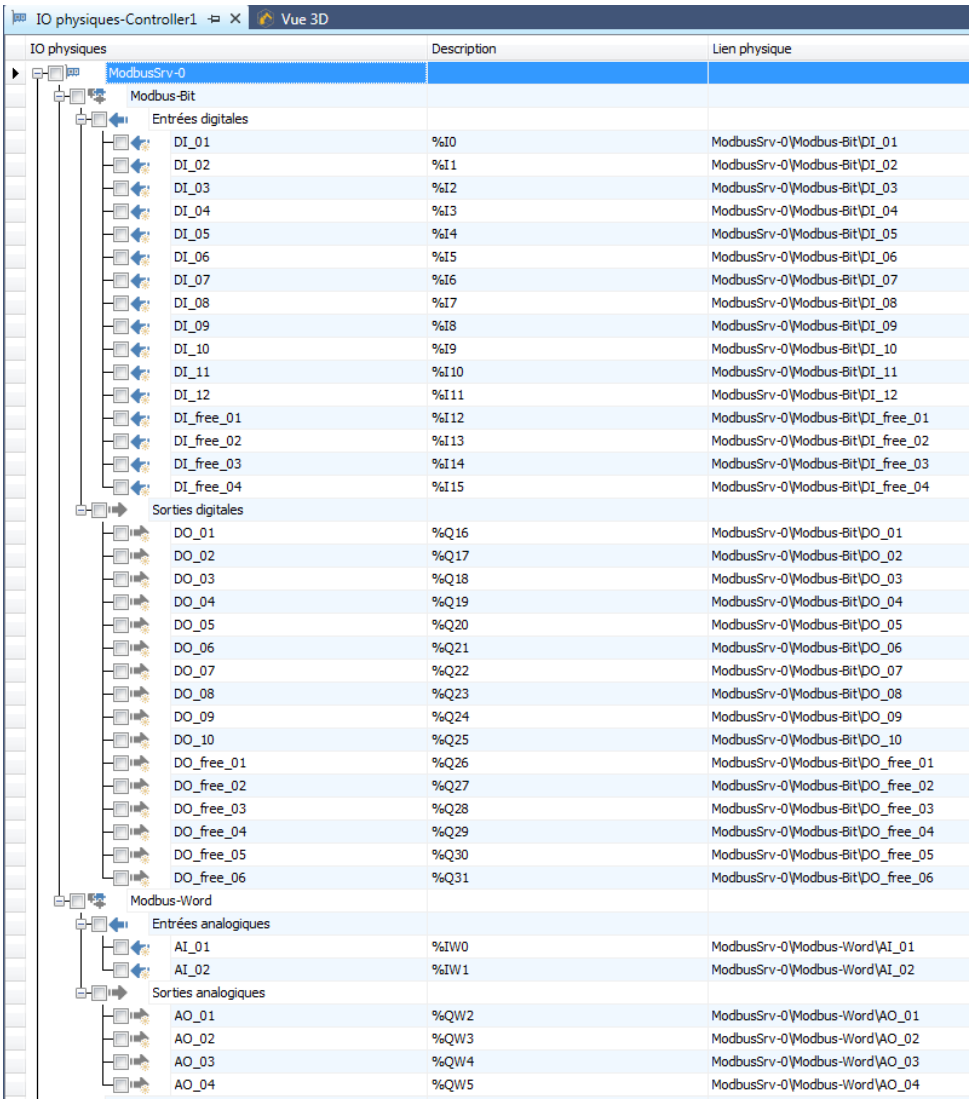
You should get below result.



Name	Type	Address	Size	Client Access
DI_01	BIT	0	1	CS8 Input (R/W)
DI_02	BIT	1	1	CS8 Input (R/W)
DI_03	BIT	2	1	CS8 Input (R/W)
DI_04	BIT	3	1	CS8 Input (R/W)
DI_05	BIT	4	1	CS8 Input (R/W)
DI_06	BIT	5	1	CS8 Input (R/W)
DI_07	BIT	6	1	CS8 Input (R/W)
DI_08	BIT	7	1	CS8 Input (R/W)
DI_09	BIT	8	1	CS8 Input (R/W)
DI_10	BIT	9	1	CS8 Input (R/W)
DI_11	BIT	10	1	CS8 Input (R/W)
DI_12	BIT	11	1	CS8 Input (R/W)
DI_free_01	BIT	12	1	CS8 Input (R/W)
DI_free_02	BIT	13	1	CS8 Input (R/W)
DI_free_03	BIT	14	1	CS8 Input (R/W)
DI_free_04	BIT	15	1	CS8 Input (R/W)
DO_01	BIT	16	1	CS8 Output (R)
DO_02	BIT	17	1	CS8 Output (R)
DO_03	BIT	18	1	CS8 Output (R)
DO_04	BIT	19	1	CS8 Output (R)
DO_05	BIT	20	1	CS8 Output (R)
DO_06	BIT	21	1	CS8 Output (R)
DO_07	BIT	22	1	CS8 Output (R)
DO_08	BIT	23	1	CS8 Output (R)
DO_09	BIT	24	1	CS8 Output (R)
DO_10	BIT	25	1	CS8 Output (R)
DO_free_01	BIT	26	1	CS8 Output (R)
DO_free_02	BIT	27	1	CS8 Output (R)
DO_free_03	BIT	28	1	CS8 Output (R)
DO_free_04	BIT	29	1	CS8 Output (R)
DO_free_05	BIT	30	1	CS8 Output (R)
DO_free_06	BIT	31	1	CS8 Output (R)
AI_01	WORD	0	1	CS8 Input (R/W)
AI_02	WORD	1	1	CS8 Input (R/W)
AO_01	WORD	2	1	CS8 Output (R)
AO_02	WORD	3	1	CS8 Output (R)
AO_03	WORD	4	1	CS8 Output (R)
AO_04	WORD	5	1	CS8 Output (R)

Save and close

You now have all your IOs available in SRS and you just have to link VAL3 variables with physical IOs.



IO physiques	Description	Lien physique
ModbusSrv-0		
Modbus-Bit		
Entrées digitales		
DI_01	%I0	ModbusSrv-0\Modbus-Bit\DI_01
DI_02	%I1	ModbusSrv-0\Modbus-Bit\DI_02
DI_03	%I2	ModbusSrv-0\Modbus-Bit\DI_03
DI_04	%I3	ModbusSrv-0\Modbus-Bit\DI_04
DI_05	%I4	ModbusSrv-0\Modbus-Bit\DI_05
DI_06	%I5	ModbusSrv-0\Modbus-Bit\DI_06
DI_07	%I6	ModbusSrv-0\Modbus-Bit\DI_07
DI_08	%I7	ModbusSrv-0\Modbus-Bit\DI_08
DI_09	%I8	ModbusSrv-0\Modbus-Bit\DI_09
DI_10	%I9	ModbusSrv-0\Modbus-Bit\DI_10
DI_11	%I10	ModbusSrv-0\Modbus-Bit\DI_11
DI_12	%I11	ModbusSrv-0\Modbus-Bit\DI_12
DI_free_01	%I12	ModbusSrv-0\Modbus-Bit\DI_free_01
DI_free_02	%I13	ModbusSrv-0\Modbus-Bit\DI_free_02
DI_free_03	%I14	ModbusSrv-0\Modbus-Bit\DI_free_03
DI_free_04	%I15	ModbusSrv-0\Modbus-Bit\DI_free_04
Sorties digitales		
DO_01	%Q16	ModbusSrv-0\Modbus-Bit\DO_01
DO_02	%Q17	ModbusSrv-0\Modbus-Bit\DO_02
DO_03	%Q18	ModbusSrv-0\Modbus-Bit\DO_03
DO_04	%Q19	ModbusSrv-0\Modbus-Bit\DO_04
DO_05	%Q20	ModbusSrv-0\Modbus-Bit\DO_05
DO_06	%Q21	ModbusSrv-0\Modbus-Bit\DO_06
DO_07	%Q22	ModbusSrv-0\Modbus-Bit\DO_07
DO_08	%Q23	ModbusSrv-0\Modbus-Bit\DO_08
DO_09	%Q24	ModbusSrv-0\Modbus-Bit\DO_09
DO_10	%Q25	ModbusSrv-0\Modbus-Bit\DO_10
DO_free_01	%Q26	ModbusSrv-0\Modbus-Bit\DO_free_01
DO_free_02	%Q27	ModbusSrv-0\Modbus-Bit\DO_free_02
DO_free_03	%Q28	ModbusSrv-0\Modbus-Bit\DO_free_03
DO_free_04	%Q29	ModbusSrv-0\Modbus-Bit\DO_free_04
DO_free_05	%Q30	ModbusSrv-0\Modbus-Bit\DO_free_05
DO_free_06	%Q31	ModbusSrv-0\Modbus-Bit\DO_free_06
Modbus-Word		
Entrées analogiques		
AI_01	%IW0	ModbusSrv-0\Modbus-Word\AI_01
AI_02	%IW1	ModbusSrv-0\Modbus-Word\AI_02
Sorties analogiques		
AO_01	%QW2	ModbusSrv-0\Modbus-Word\AO_01
AO_02	%QW3	ModbusSrv-0\Modbus-Word\AO_02
AO_03	%QW4	ModbusSrv-0\Modbus-Word\AO_03
AO_04	%QW5	ModbusSrv-0\Modbus-Word\AO_04

5.2 Example with a WAGO/STÄUBLI Modbus client (5109-8888)

That specific client has built-in:

- 2 bit output: a life bit and one unused
- 2 analog output: a version number and one unused

It is after those that, respectively, bit output then bit input and analog output then analog input appears.

We have 12 digital outputs, 10 digital inputs, 2 analog outputs, 4 analog inputs

WAGO (5109-8888), client (master)			CS8C, server (slave)		
	Bit	Output name	Input name	Bit	
Digital output	1	LifeBit	LifeBit	1	Digital input
	2	NullBit	NullBit	2	
	3	DO_01	DI_01	3	
	4	DO_02	DI_02	4	
	5	DO_03	DI_03	5	
	6	DO_04	DI_04	6	
	7	DO_05	DI_05	7	
	8	DO_06	DI_06	8	
	9	DO_07	DI_07	9	
	10	DO_08	DI_08	10	
	11	DO_09	DI_09	11	
	12	DO_10	DI_10	12	
	13	DO_11	DI_11	13	
	14	DO_12	DI_12	14	
Digital input	1	DI_01	DO_01	15	Digital output
	2	DI_02	DO_02	16	
	3	DI_03	DO_03	17	
	4	DI_04	DO_04	18	
	5	DI_05	DO_05	19	
	6	DI_06	DO_06	20	
	7	DI_07	DO_07	21	
	8	DI_08	DO_08	22	
	9	DI_09	DO_09	23	
	10	DI_10	DO_10	24	

WAGO (5109-8888), client (master)			CS8C, server (slave)		
	Word	Input name	Output name	Word	
Analog output	1	mbVersion	mbVersion	1	Analog input
	2	mbUnused	mbUnused	2	
	3	AO_01	AI_01	3	
	4	AO_02	AI_02	4	
Analog input	1	AI_01	AO_01	5	Analog output
	2	AI_02	AO_02	6	
	3	AI_03	AO_03	7	
	4	AI_04	AO_04	8	

Start by adding 2 items that you configure as “bit” and “CS8 input (R/W)”. Name them respectively LifeBit and NullBit, they correspond to 2 internal bit of WAGO 5109-8888.

Add 12 items that you configure as “bit” and “CS8 input (R/W)”. Name them properly; they will correspond to WAGO 5109-8888 digital outputs.

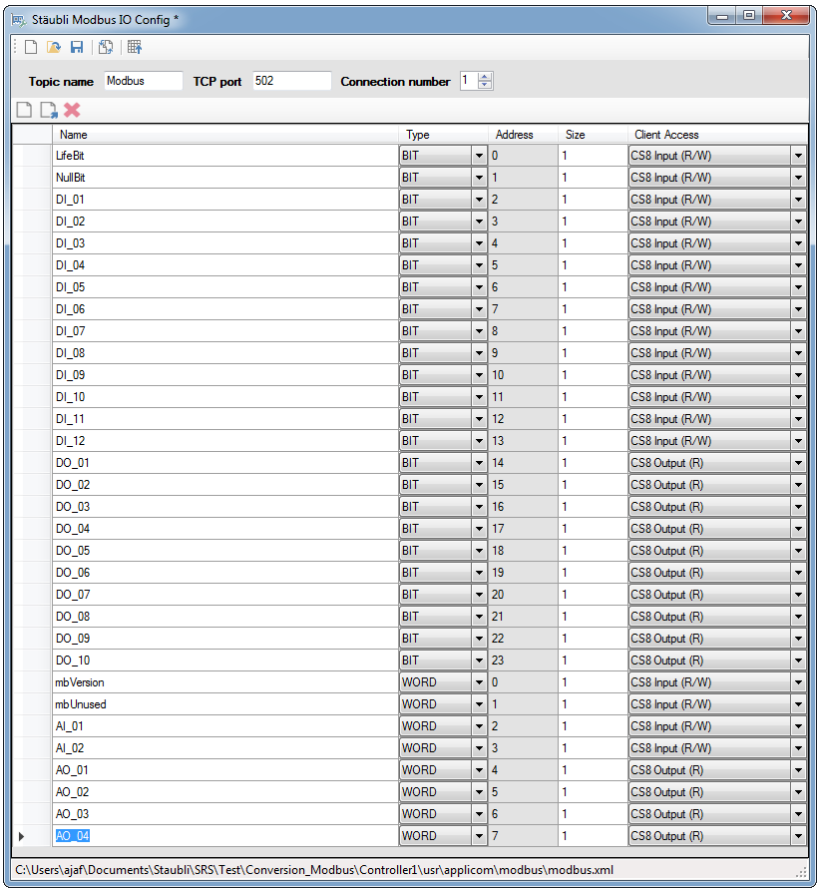
Add 10 items that you configure as “bit” and “CS8 output (R)”. Name them properly; they will correspond to WAGO 5109-8888 digital inputs.

Add 2 items that you configure as “word” and “CS8 input (R/W)”. Name them respectively mbVersion and mbUnused.

Add 2 items that you configure as “word” and “CS8 input (R/W)”. Name them properly; they will correspond to WAGO 5109-8888 analog outputs (2 analog outputs).

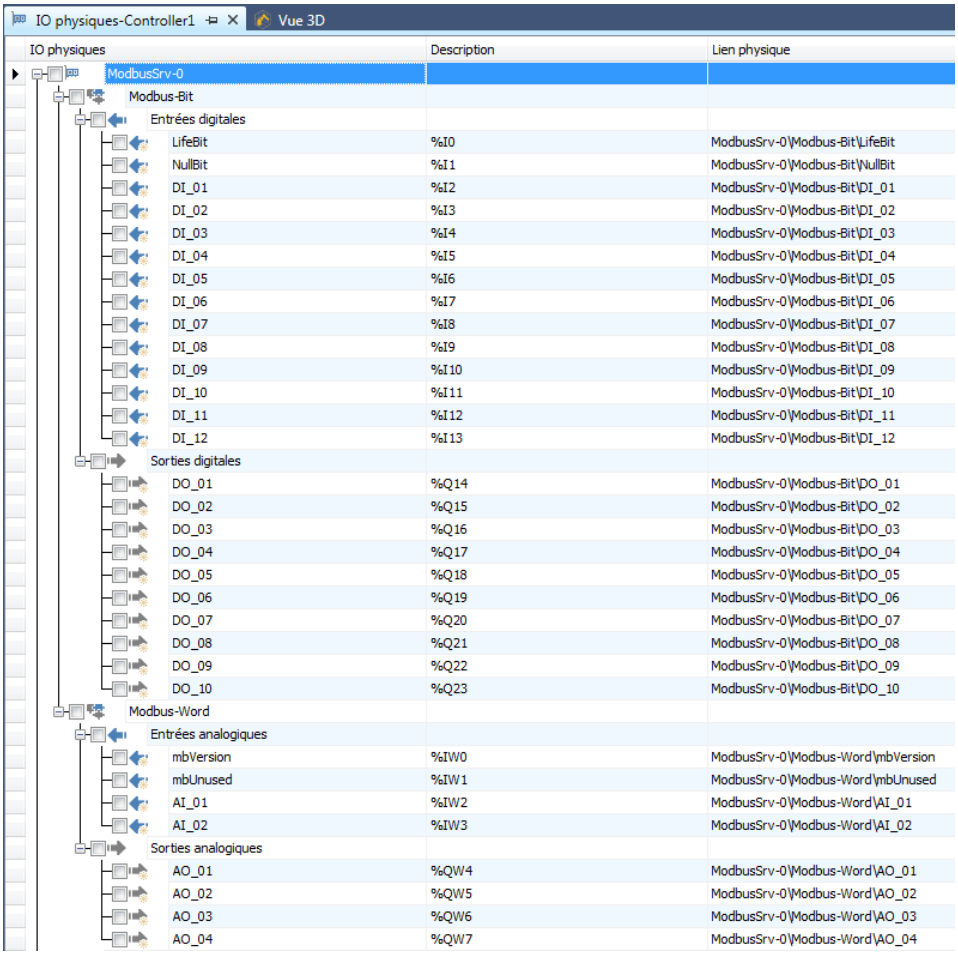
Add 4 items that you configure as “word” and “CS8 output (R)”. Name them properly; they will correspond to WAGO 5109-8888 analog inputs (4 analog inputs).

You should get below result.



Name	Type	Address	Size	Client Access
LifeBit	BIT	0	1	CS8 Input (R/W)
NullBit	BIT	1	1	CS8 Input (R/W)
DI_01	BIT	2	1	CS8 Input (R/W)
DI_02	BIT	3	1	CS8 Input (R/W)
DI_03	BIT	4	1	CS8 Input (R/W)
DI_04	BIT	5	1	CS8 Input (R/W)
DI_05	BIT	6	1	CS8 Input (R/W)
DI_06	BIT	7	1	CS8 Input (R/W)
DI_07	BIT	8	1	CS8 Input (R/W)
DI_08	BIT	9	1	CS8 Input (R/W)
DI_09	BIT	10	1	CS8 Input (R/W)
DI_10	BIT	11	1	CS8 Input (R/W)
DI_11	BIT	12	1	CS8 Input (R/W)
DI_12	BIT	13	1	CS8 Input (R/W)
DO_01	BIT	14	1	CS8 Output (R)
DO_02	BIT	15	1	CS8 Output (R)
DO_03	BIT	16	1	CS8 Output (R)
DO_04	BIT	17	1	CS8 Output (R)
DO_05	BIT	18	1	CS8 Output (R)
DO_06	BIT	19	1	CS8 Output (R)
DO_07	BIT	20	1	CS8 Output (R)
DO_08	BIT	21	1	CS8 Output (R)
DO_09	BIT	22	1	CS8 Output (R)
DO_10	BIT	23	1	CS8 Output (R)
mbVersion	WORD	0	1	CS8 Input (R/W)
mbUnused	WORD	1	1	CS8 Input (R/W)
AI_01	WORD	2	1	CS8 Input (R/W)
AI_02	WORD	3	1	CS8 Input (R/W)
AO_01	WORD	4	1	CS8 Output (R)
AO_02	WORD	5	1	CS8 Output (R)
AO_03	WORD	6	1	CS8 Output (R)
AO_04	WORD	7	1	CS8 Output (R)

You now have all your IOs available in SRS and you just have to link VAL3 variables with physical IOs.



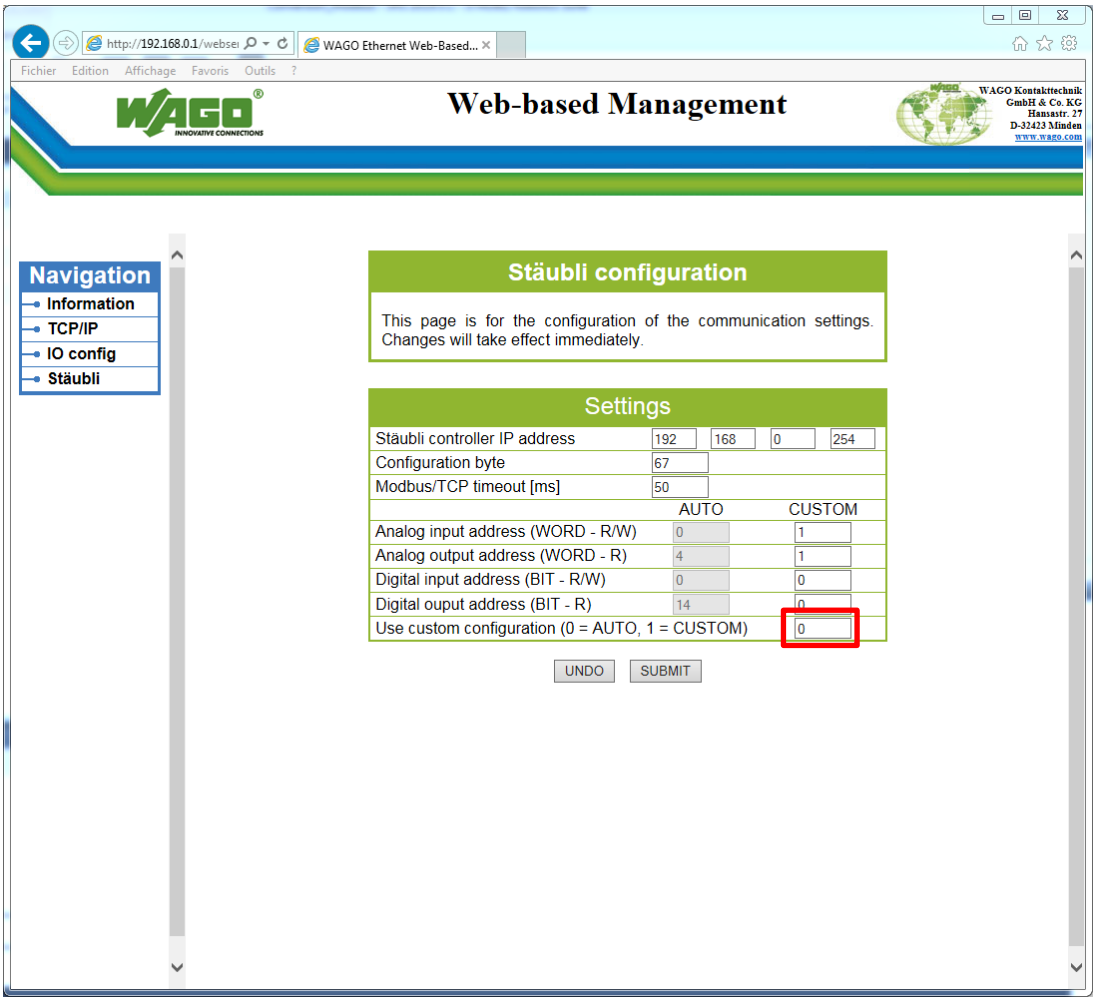
IO physiques-Controller1 | Vue 3D

IO physiques	Description	Lien physique
ModbusSrv-0		
Modbus-Bit		
Entrées digitales		
LifeBit	%I0	ModbusSrv-0\Modbus-Bit\LifeBit
NullBit	%I1	ModbusSrv-0\Modbus-Bit\NullBit
DI_01	%I2	ModbusSrv-0\Modbus-Bit\DI_01
DI_02	%I3	ModbusSrv-0\Modbus-Bit\DI_02
DI_03	%I4	ModbusSrv-0\Modbus-Bit\DI_03
DI_04	%I5	ModbusSrv-0\Modbus-Bit\DI_04
DI_05	%I6	ModbusSrv-0\Modbus-Bit\DI_05
DI_06	%I7	ModbusSrv-0\Modbus-Bit\DI_06
DI_07	%I8	ModbusSrv-0\Modbus-Bit\DI_07
DI_08	%I9	ModbusSrv-0\Modbus-Bit\DI_08
DI_09	%I10	ModbusSrv-0\Modbus-Bit\DI_09
DI_10	%I11	ModbusSrv-0\Modbus-Bit\DI_10
DI_11	%I12	ModbusSrv-0\Modbus-Bit\DI_11
DI_12	%I13	ModbusSrv-0\Modbus-Bit\DI_12
Sorties digitales		
DO_01	%Q14	ModbusSrv-0\Modbus-Bit\DO_01
DO_02	%Q15	ModbusSrv-0\Modbus-Bit\DO_02
DO_03	%Q16	ModbusSrv-0\Modbus-Bit\DO_03
DO_04	%Q17	ModbusSrv-0\Modbus-Bit\DO_04
DO_05	%Q18	ModbusSrv-0\Modbus-Bit\DO_05
DO_06	%Q19	ModbusSrv-0\Modbus-Bit\DO_06
DO_07	%Q20	ModbusSrv-0\Modbus-Bit\DO_07
DO_08	%Q21	ModbusSrv-0\Modbus-Bit\DO_08
DO_09	%Q22	ModbusSrv-0\Modbus-Bit\DO_09
DO_10	%Q23	ModbusSrv-0\Modbus-Bit\DO_10
Modbus-Word		
Entrées analogiques		
mbVersion	%IW0	ModbusSrv-0\Modbus-Word\mbVersion
mbUnused	%IW1	ModbusSrv-0\Modbus-Word\mbUnused
AI_01	%IW2	ModbusSrv-0\Modbus-Word\AI_01
AI_02	%IW3	ModbusSrv-0\Modbus-Word\AI_02
Sorties analogiques		
AO_01	%QW4	ModbusSrv-0\Modbus-Word\AO_01
AO_02	%QW5	ModbusSrv-0\Modbus-Word\AO_02
AO_03	%QW6	ModbusSrv-0\Modbus-Word\AO_03
AO_04	%QW7	ModbusSrv-0\Modbus-Word\AO_04

Below, corresponding WAGO coupler configuration set to use automatic configuration.

Login: admin

Password: wago



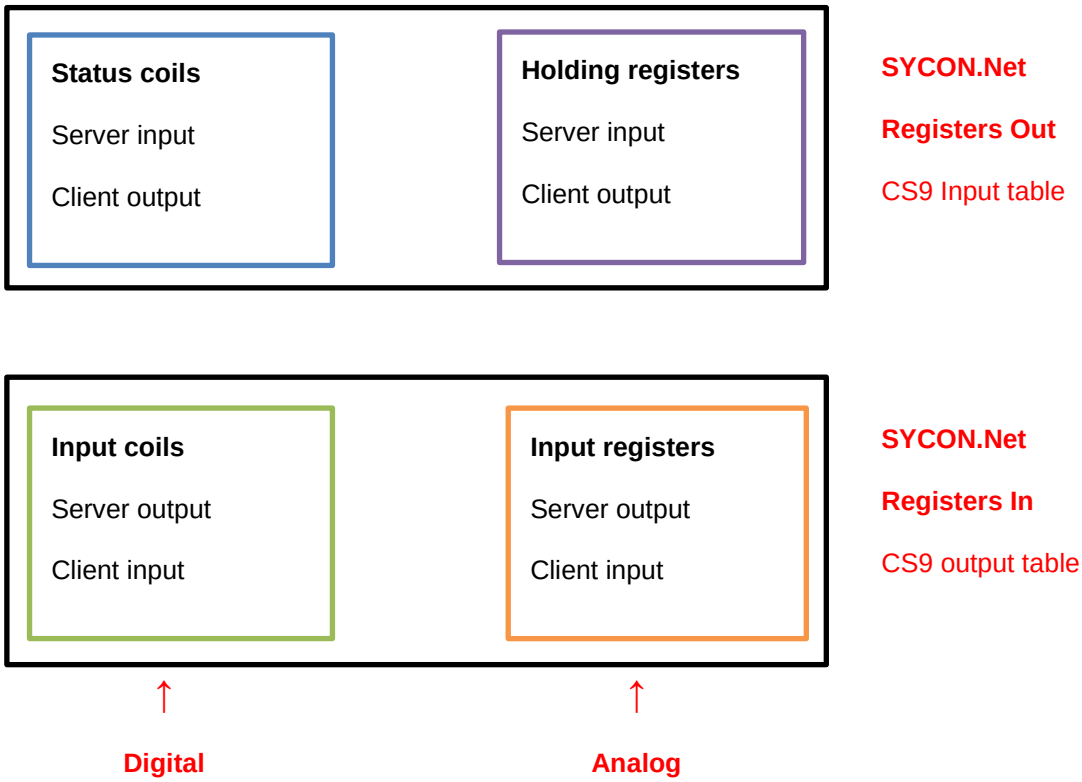
6 Modbus server on CS9

All is managed internally as word but stored as 2 bytes.

Holding registers follow status coils in a single table.

Input registers follow input coils in a single table.

That is why you must always have a multiple of 16 for coils to fill a word multiple.

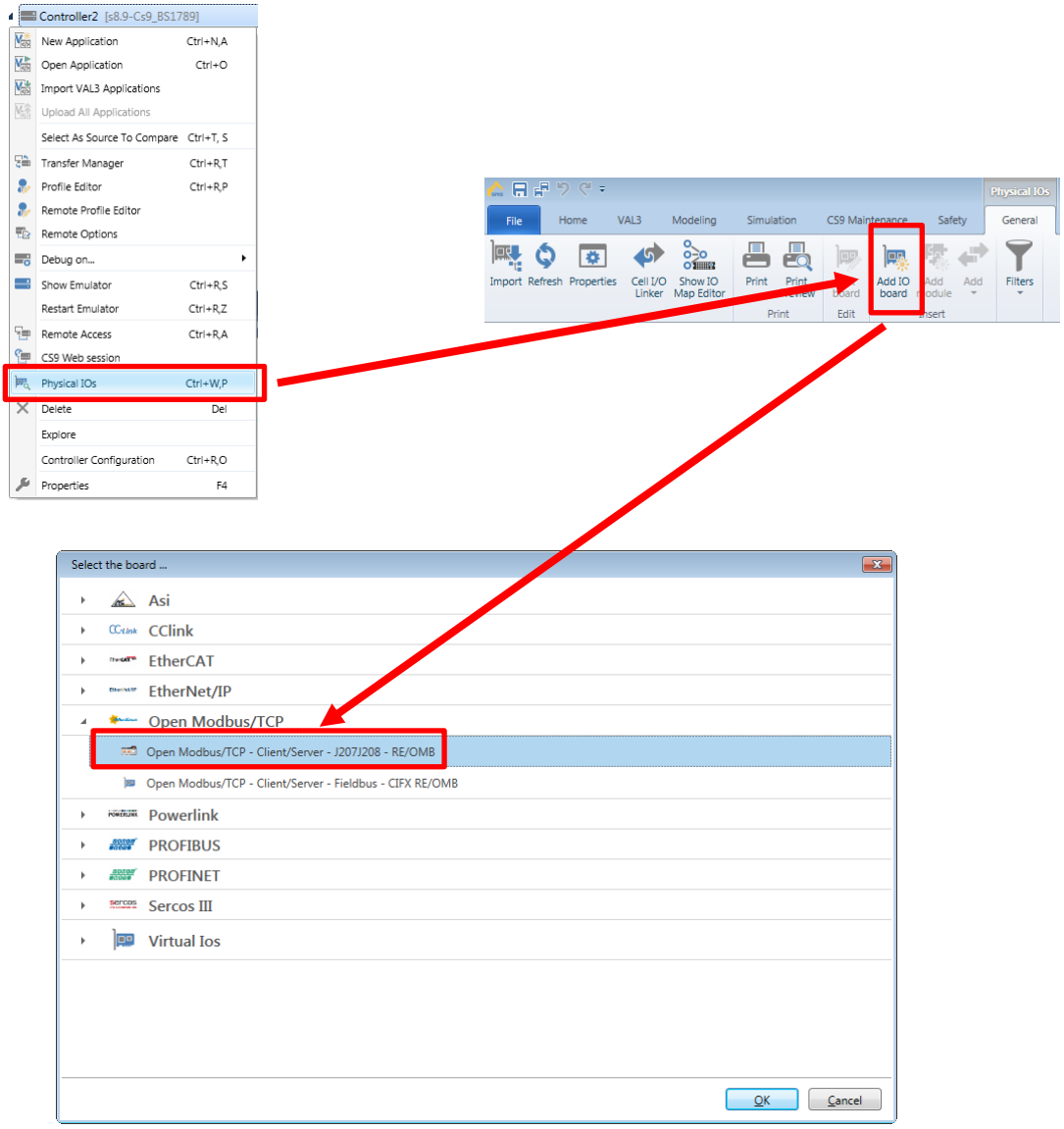


For data transmission, Modbus protocol specify that values must be encoded as big endian (Most Significant Byte first then Last Significant Byte).

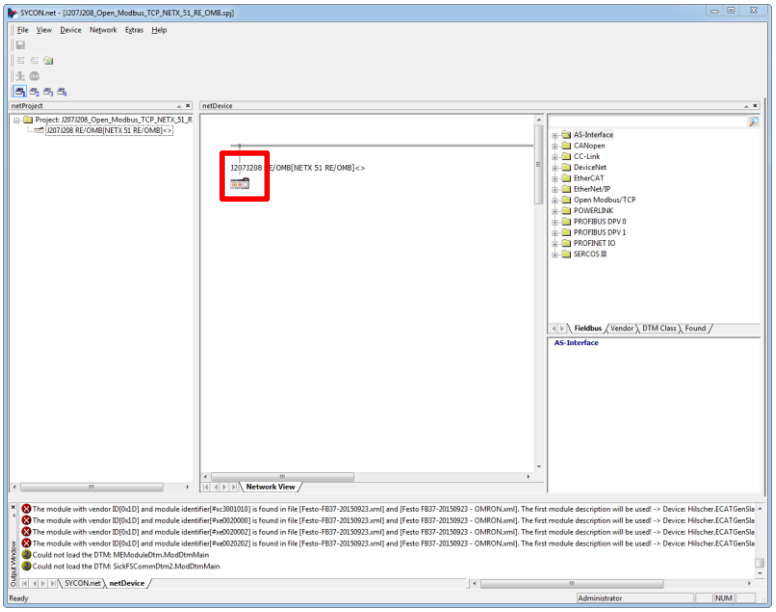
On CS9 this also apply on coils for their internal storage, thus you will have a swap between first 8 coils and last 8 coils:

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Coil	9	10	11	12	13	14	15	16	1	2	3	4	5	6	7	8

In SRS, select the controller in the cell explorer, make a right click on it and select “Physical IOs”, “Add IO board”, “Open Modbus/TCP – Client/Server – J207/J208 – RE/OMB”



This open SYCON.net. Double click on J207/J208

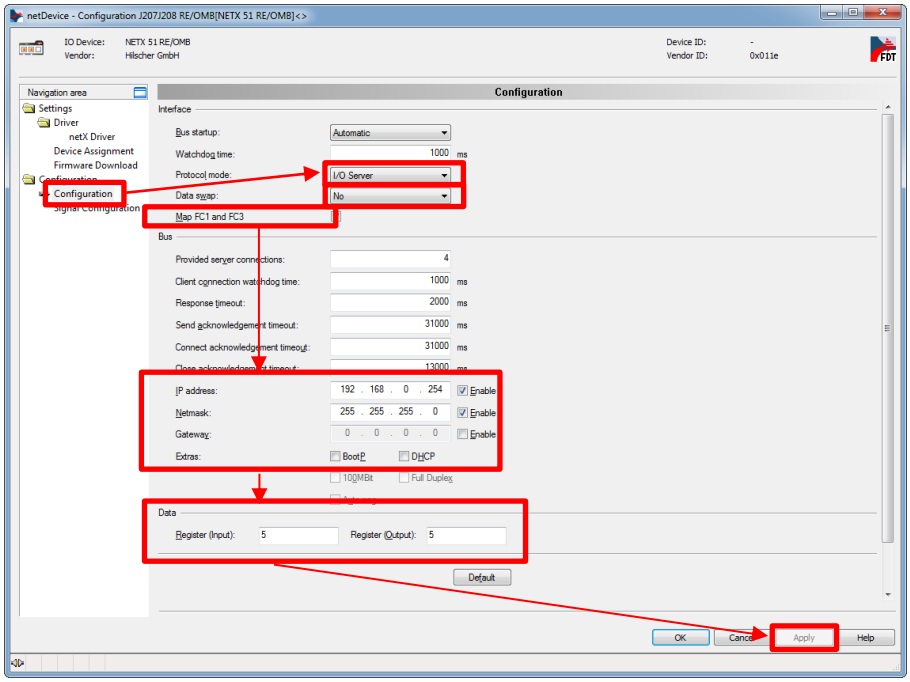


Select “Configuration” branch, select “I/O Server”. Check “Map FC1 and FC3” if input and output are merged in a single table (one for digital IO, one for analog IO).

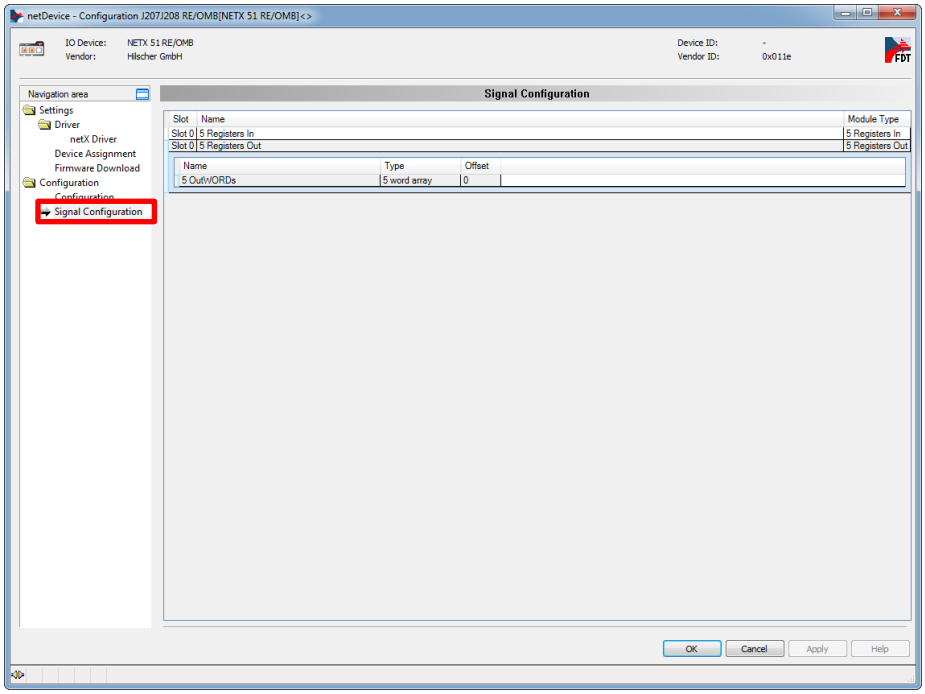
Uncheck “DHCP”, enable and set “IP address” and “Netmask”. Here, you specify the server IP address.

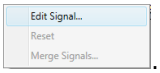
Fill “Register Input” and “Register Output” size. Be careful as all detailed configuration you will do on next step will be lost if you change thoses values.

“Apply”.



Select “Signal Configuration” branch to access word arrays, which has been built.

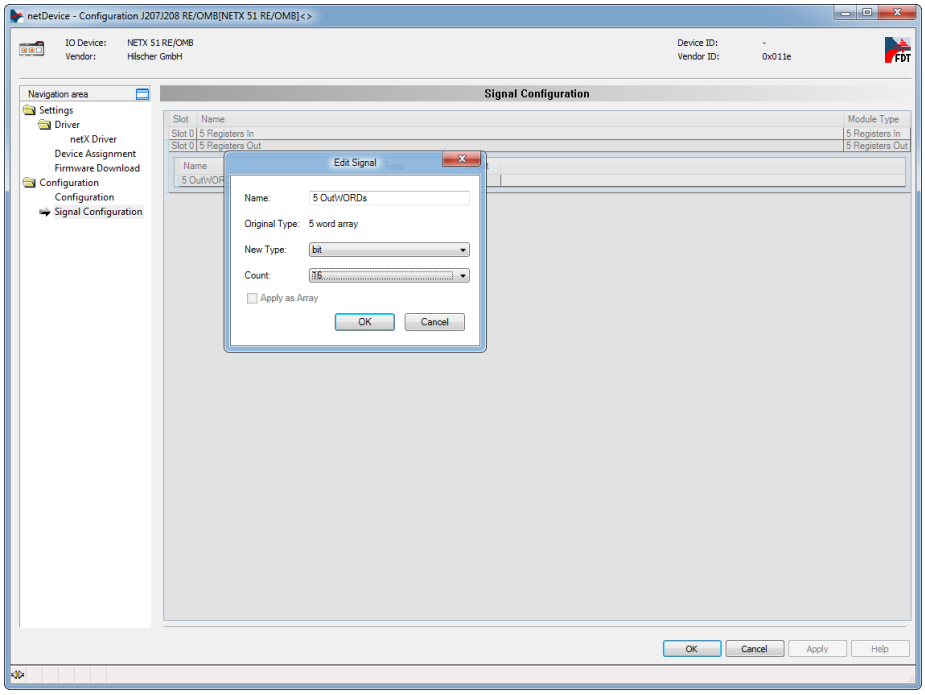


Right click on the array and select “Edit Signal” 

.

You can now split / merge / rename items.

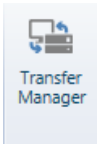
Press “Apply” after each change to do it.

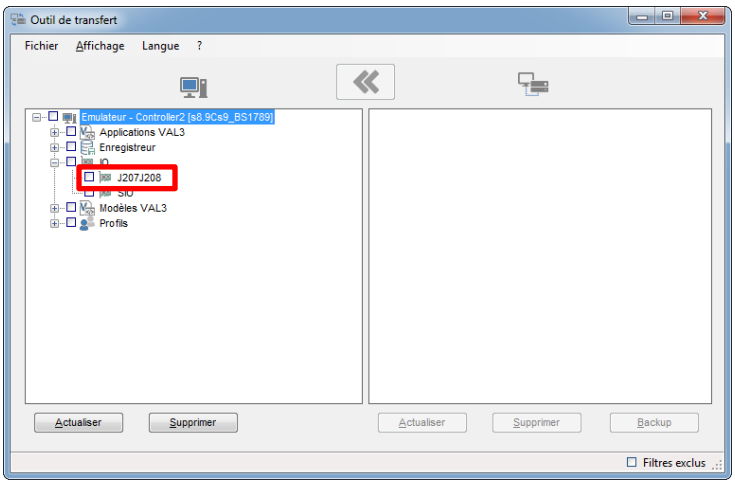


When finished, press “Ok” then “File”, “Save” and close SYCON.

With “Transfer manager”, send the “J207/J208” configuration to the controller.

Reboot the controller in order to take into account the new configuration.





6.1 Example with a PLC/screen configured as a client

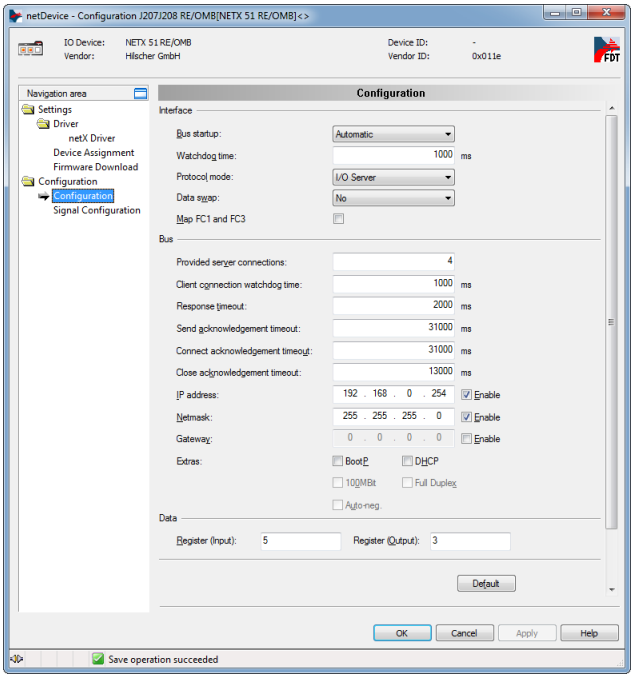
In this example, the client has 12 digital outputs, 10 digital inputs, 2 analog outputs, 4 analog inputs.

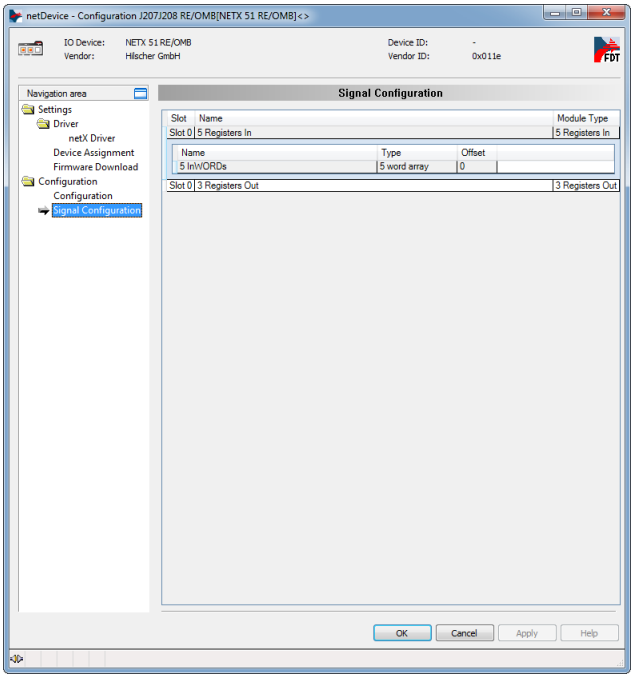
In order to prevent issues we will round up digital outputs and inputs to the next 16 multiple.

PLC / screen, client (master)				CS9, server (slave)				
	Word	Bit	Output name	Input name	Address	Bit	Word	
Digital output	1	1	DO_01	DI_01	000009	9	1	Digital input
		2	DO_02	DI_02	000010	10		
		3	DO_03	DI_03	000011	11		
		4	DO_04	DI_04	000012	12		
		5	DO_05	DI_05	000013	13		
		6	DO_06	DI_06	000014	14		
		7	DO_07	DI_07	000015	15		
		8	DO_08	DI_08	000016	16		
		9	DO_09	DI_09	000001	1		
		10	DO_10	DI_10	000002	2		
		11	DO_11	DI_11	000003	3		
		12	DO_12	DI_12	000004	4		
		13	DO_free_01	DI_free_01	000005	5		
		14	DO_free_02	DI_free_02	000006	6		
		15	DO_free_03	DI_free_03	000007	7		
		16	DO_free_04	DI_free_04	000008	8		
Analog output	2		AO_01	AI_01	300002		2	Analog input
	3		AO_02	AI_02	300003		3	

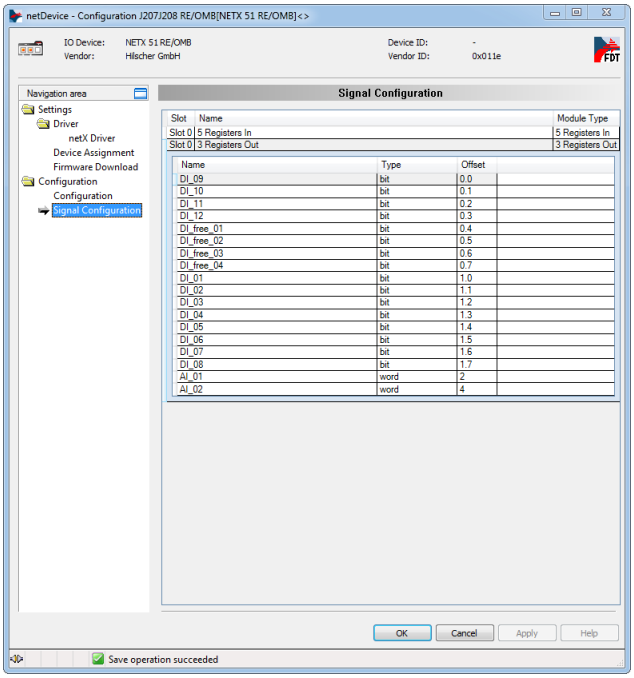
PLC / screen, client (master)				CS9, server (slave)				
	Word	Bit	Output name	Input name	Address	Bit	Word	
Digital input	1	1	DI_01	DO_01	100009	9	1	Digital output
		2	DI_02	DO_02	100010	10		
		3	DI_03	DO_03	100011	11		
		4	DI_04	DO_04	100012	12		
		5	DI_05	DO_05	100013	13		
		6	DI_06	DO_06	100014	14		
		7	DI_07	DO_07	100015	15		
		8	DI_08	DO_08	100016	16		
		9	DI_09	DO_09	100001	1		
		10	DI_10	DO_10	100002	2		
		11	DI_free_01	DO_free_01	100003	3		
		12	DI_free_02	DO_free_02	100004	4		
		13	DI_free_03	DO_free_03	100005	5		
		14	DI_free_04	DO_free_04	100006	6		
		15	DI_free_05	DO_free_05	100007	7		
		16	DI_free_06	DO_free_06	100008	8		
Analog input	2		AI_01	AO_01	400002		2	Analog output
	3		AI_02	AO_02	400003		3	
	4		AI_03	AO_03	400004		4	
	5		AI_04	AO_04	400005		5	

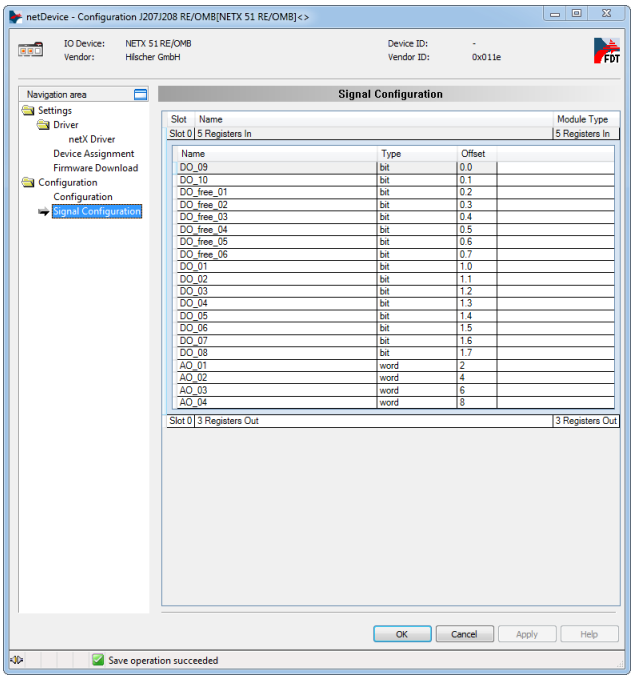
Above exchange table shows that we need 5 register (input) and 3 register (output).



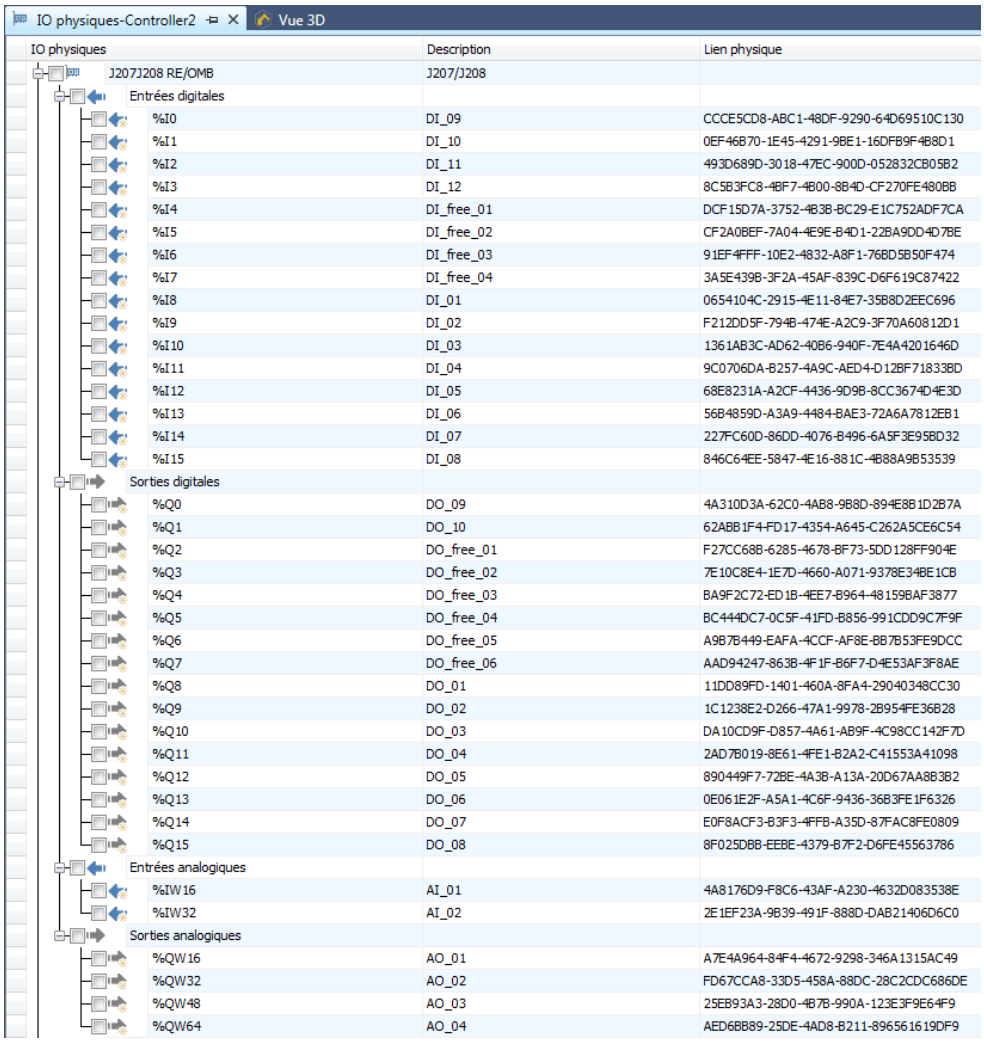


In “Signal Configuration”, split “Registers In” and “Registers Out” as defined in the exchange table.


netDevice - Configuration J207J208 RE/OMB[NETX 51 RE/OMB]<>
IO Device: NETX 51 RE/OMB
Vendor: Hilscher GmbH
Device ID: -
Vendor ID: 0x011e
Navigation area
Settings
Driver
netX Driver
Device Assignment
Firmware Download
Configuration
Configuration
Signal Configuration
Signal Configuration
Slot Name Module Type
Slot 0 5 Registers In 5 Registers In
Slot 0 3 Registers Out 3 Registers Out
Name Type Offset
DI_09 bit 0.0
DI_10 bit 0.1
DI_11 bit 0.2
DI_12 bit 0.3
DI_free_01 bit 0.4
DI_free_02 bit 0.5
DI_free_03 bit 0.6
DI_free_04 bit 0.7
DI_01 bit 1.0
DI_02 bit 1.1
DI_03 bit 1.2
DI_04 bit 1.3
DI_05 bit 1.4
DI_06 bit 1.5
DI_07 bit 1.6
DI_08 bit 1.7
AI_01 word 2
AI_02 word 4
OK
Cancel
Apply
Help
Save operation succeeded


netDevice - Configuration J207J208 RE/OMB[NETX 51 RE/OMB]<>
IO Device: NETX 51 RE/OMB
Vendor: Hilscher GmbH
Device ID: -
Vendor ID: 0x011e
Navigation area
Settings
Driver
netX Driver
Device Assignment
Firmware Download
Configuration
Configuration
Signal Configuration
Signal Configuration
Slot Name Module Type
Slot 0 5 Registers In 5 Registers In
Name Type Offset
DO_09 bit 0.0
DO_10 bit 0.1
DO_free_01 bit 0.2
DO_free_02 bit 0.3
DO_free_03 bit 0.4
DO_free_04 bit 0.5
DO_free_05 bit 0.6
DO_free_06 bit 0.7
DO_01 bit 1.0
DO_02 bit 1.1
DO_03 bit 1.2
DO_04 bit 1.3
DO_05 bit 1.4
DO_06 bit 1.5
DO_07 bit 1.6
DO_08 bit 1.7
AO_01 word 2
AO_02 word 4
AO_03 word 6
AO_04 word 8
Slot 0 3 Registers Out 3 Registers Out
OK
Cancel
Apply
Help
Save operation succeeded

You now have all your IOs available in SRS and you just have to link VAL3 variables with physical IOs.



IO physiques	Description	Lien physique
J207J208 RE/OMB	J207/J208	
Entrées digitales		
%I0	DI_09	CCCE5CD8-ABC1-48DF-9290-64D69510C130
%I1	DI_10	0EF46B70-1E45-4291-9BE1-16DFB9F4B8D1
%I2	DI_11	493D689D-3018-47EC-900D-052832CB05B2
%I3	DI_12	8C5B3FC8-4BF7-4B00-8B4D-CF270FE480BB
%I4	DI_free_01	DCF15D7A-3752-4B3B-BC29-E1C752ADF7CA
%I5	DI_free_02	CF2A0BEF-7A04-4E9E-B4D1-22BA9DD4D7BE
%I6	DI_free_03	91EF4FFF-10E2-4832-A8F1-76BD5B50F474
%I7	DI_free_04	3A5E439B-3F2A-45AF-839C-D6F619C87422
%I8	DI_01	0654104C-2915-4E11-84E7-35B8D2EEC696
%I9	DI_02	F212DD5F-794B-474E-A2C9-3F70A60812D1
%I10	DI_03	1361AB3C-AD62-40B6-940F-7E4A4201646D
%I11	DI_04	9C0706DA-B257-4A9C-AED4-D12BF71833BD
%I12	DI_05	68E8231A-A2CF-4436-9D9B-8CC3674D4E3D
%I13	DI_06	56B4859D-A3A9-4484-BAE3-72A6A7812EB1
%I14	DI_07	227FC60D-86DD-4076-B496-6A5F3E95BD32
%I15	DI_08	846C64EE-5847-4E16-881C-4B88A9B53539
Sorties digitales		
%Q0	DO_09	4A310D3A-62C0-4AB8-9B8D-894E8B1D2B7A
%Q1	DO_10	62ABB1F4-FD17-4354-A645-C262A5CE6C54
%Q2	DO_free_01	F27CC68B-6285-4678-BF73-5DD128FF904E
%Q3	DO_free_02	7E10C8E4-1E7D-4660-A071-9378E34BE1CB
%Q4	DO_free_03	BA9F2C72-ED1B-4EE7-B964-48159BAF3877
%Q5	DO_free_04	BC444DC7-0C5F-41FD-B856-991CDD9C7F9F
%Q6	DO_free_05	A9B7B449-EAFA-4CCF-AF8E-BB7B53FE9DCC
%Q7	DO_free_06	AAD94247-863B-4F1F-B6F7-D4E53AF3F8AE
%Q8	DO_01	11DD89FD-1401-460A-8FA4-29040348CC30
%Q9	DO_02	1C1238E2-D266-47A1-9978-2B954FE36B28
%Q10	DO_03	DA10CD9F-D857-4A61-AB9F-4C98CC142F7D
%Q11	DO_04	2AD7B019-8E61-4FE1-B2A2-C41553A41098
%Q12	DO_05	890449F7-72BE-4A3B-A13A-20D67AA8B3B2
%Q13	DO_06	0E061E2F-A5A1-4C6F-9436-36B3FE1F6326
%Q14	DO_07	E0F8ACF3-B3F3-4FFB-A35D-87FAC8FE0809
%Q15	DO_08	8F025DBB-EEBE-4379-B7F2-D6FE45563786
Entrées analogiques		
%IW16	AI_01	4A8176D9-F8C6-43AF-A230-4632D083538E
%IW32	AI_02	2E1EF23A-9B39-491F-888D-DAB21406D6C0
Sorties analogiques		
%QW16	AO_01	A7E4A964-84F4-4672-9298-346A1315AC49
%QW32	AO_02	FD67CCA8-33D5-458A-88DC-28C2CDC686DE
%QW48	AO_03	25EB93A3-28D0-4B7B-990A-123E3F9E64F9
%QW64	AO_04	AED6BB89-25DE-4AD8-B211-896561619DF9

6.2 Example with a WAGO/STÄUBLI Modbus client (5109-8888)

That specific client has built-in:

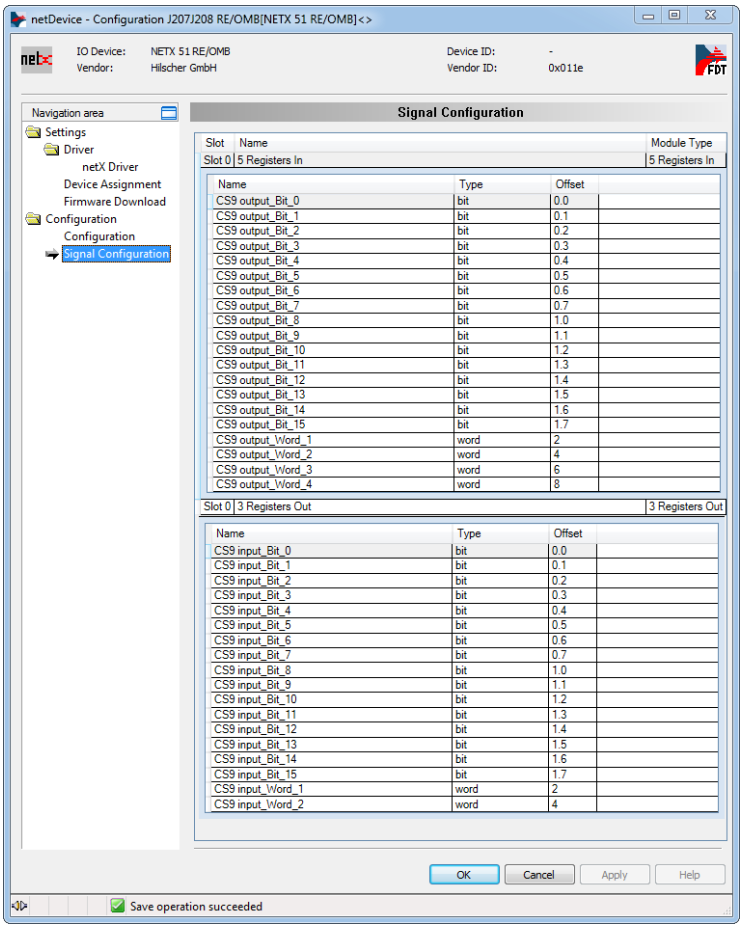
- 2 bit output: a life bit and one unused
- 2 analog output: a version number and one unused

We have 12 digital outputs, 10 digital inputs, 2 analog outputs, 4 analog inputs

WAGO (5109-8888), client (master)				CS9, server (slave)			
	Word	Bit	Output name	Input name	Bit	Word	
Digital output	1	1	LifeBit	LifeBit	9	1	Digital input
		2	NullBit	NullBit	10		
		3	DO_01	DI_01	11		
		4	DO_02	DI_02	12		
		5	DO_03	DI_03	13		
		6	DO_04	DI_04	14		
		7	DO_05	DI_05	15		
		8	DO_06	DI_06	16		
		9	DO_07	DI_07	1		
		10	DO_08	DI_08	2		
		11	DO_09	DI_09	3		
		12	DO_10	DI_10	4		
		13	DO_11	DI_11	5		
		14	DO_12	DI_12	6		
		15	DO_free_01	DI_free_01	7		
		16	DO_free_02	DI_free_02	8		
Analog output	2		mbVersion	mbVersion		2	Analog input
	3		mbUnused	mbUnused		3	
	4		AO_01	AI_01		4	
	5		AO_02	AI_02		5	

WAGO (5109-8888), client (master)				CS9, server (slave)			
	Word	Bit	Output name	Input name	Bit	Word	
Digital input	1	1	DI_01	DO_01	9	1	Digital output
		2	DI_02	DO_02	10		
		3	DI_03	DO_03	11		
		4	DI_04	DO_04	12		
		5	DI_05	DO_05	13		
		6	DI_06	DO_06	14		
		7	DI_07	DO_07	15		
		8	DI_08	DO_08	16		
		9	DI_09	DO_09	1		
		10	DI_10	DO_10	2		
		11	DI_free_01	DO_free_01	3		
		12	DI_free_02	DO_free_02	4		
		13	DI_free_03	DO_free_03	5		
		14	DI_free_04	DO_free_04	6		
		15	DI_free_05	DO_free_05	7		
		16	DI_free_06	DO_free_06	8		
Analog input	2		AI_01	AO_01		2	Analog output
	3		AI_02	AO_02		3	
	4		AI_03	AO_03		4	
	5		AI_04	AO_04		5	

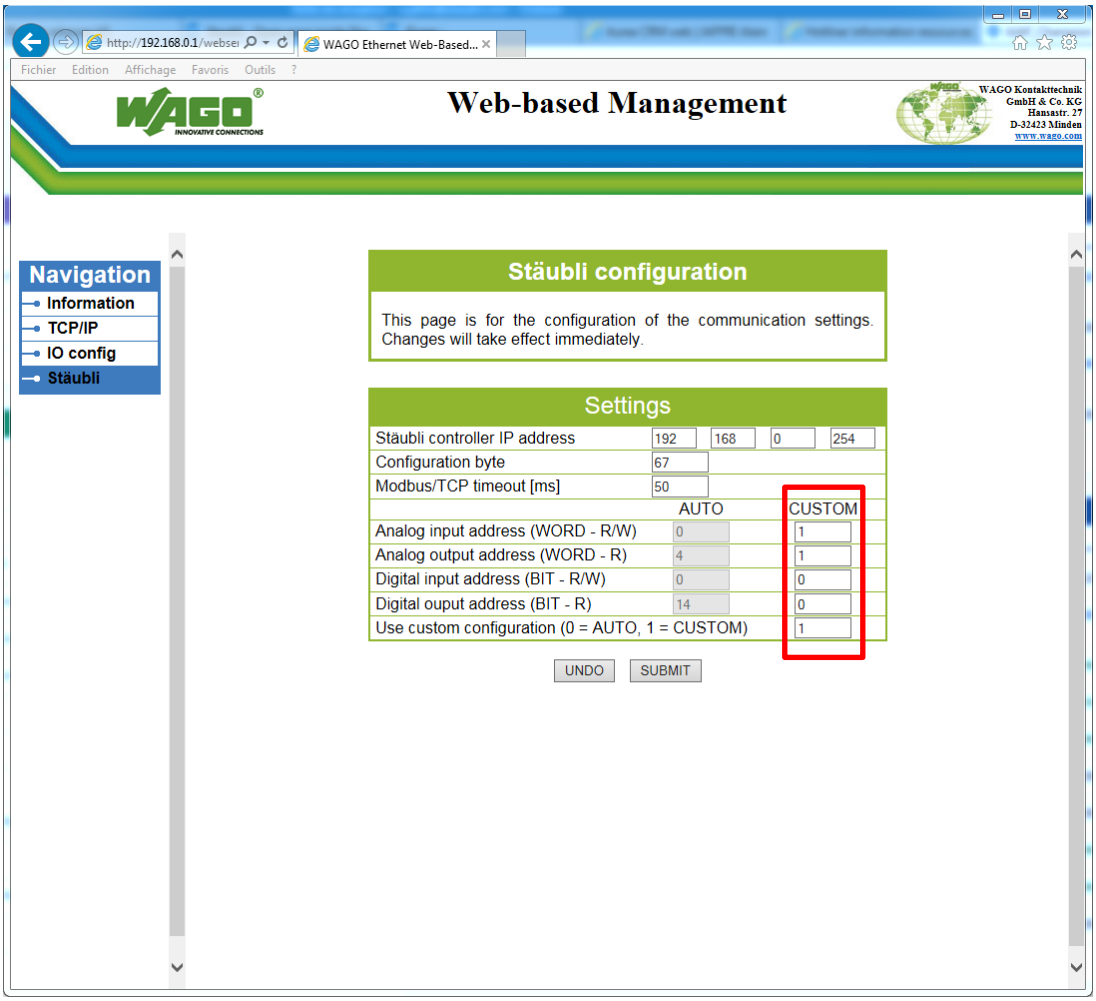
Below a sample SRS configuration for such client with 12 digital inputs, 8 digital outputs, 2 analog inputs and 4 analog outputs.



Below, corresponding WAGO coupler configuration where analog input and output must be shifted by the amount of word used for the bits and thus use a custom configuration.

Login: admin

Password: wago



7 Modbus client on CS9

Internally, all is managed as word but stored in 2 bytes.

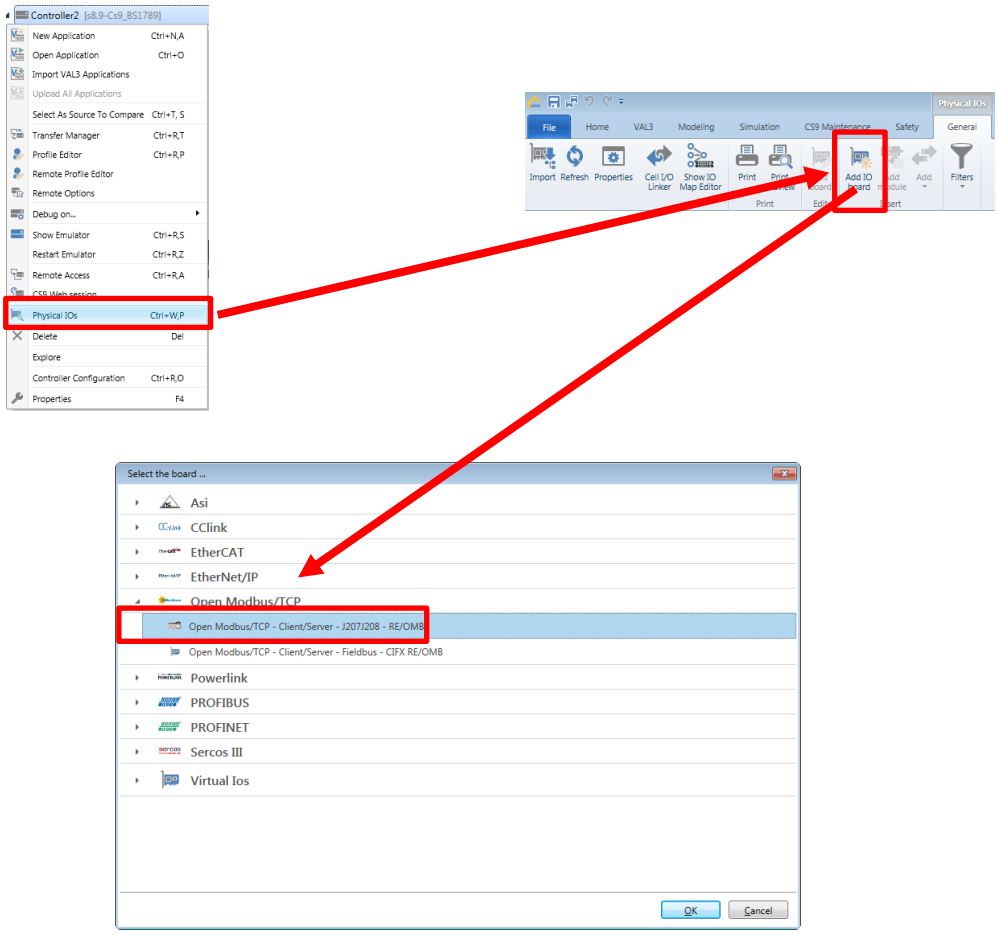
For data transmission, Modbus protocol specify that values must be encoded as big endian (Most Significant Byte first then Last Significant Byte).

On CS9 this also apply on coils for their internal storage, thus you will have a swap between first 8 coils and last 8 coils:

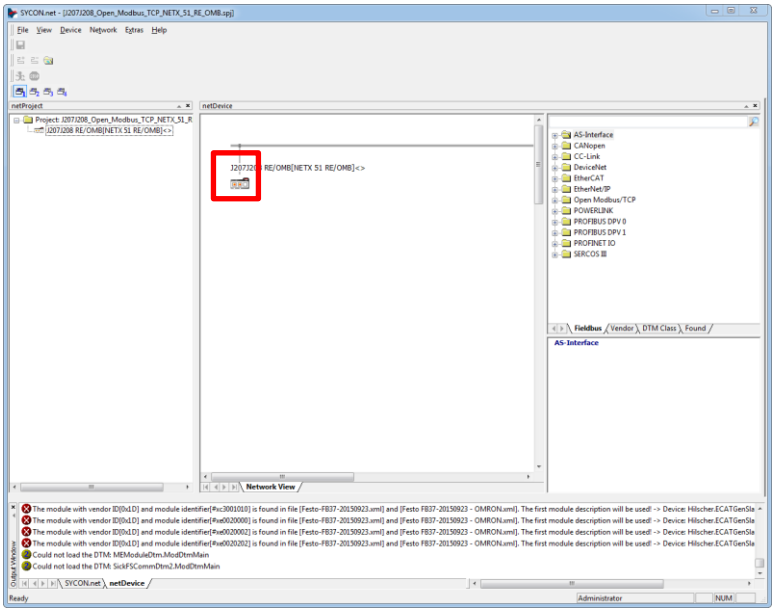
Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Coil	9	10	11	12	13	14	15	16	1	2	3	4	5	6	7	8

If you use “Data swap”, coils (digital IOs) will be back in order but the 2 bytes used of registers (analog IOs) will be swapped also and thus may give erroneous value.

In SRS, select the controller in the cell explorer, make a right click on it and select “Physical IOs”, “Add IO board”, “Open Modbus/TCP – Client/Server – J207/J208 – RE/OMB”



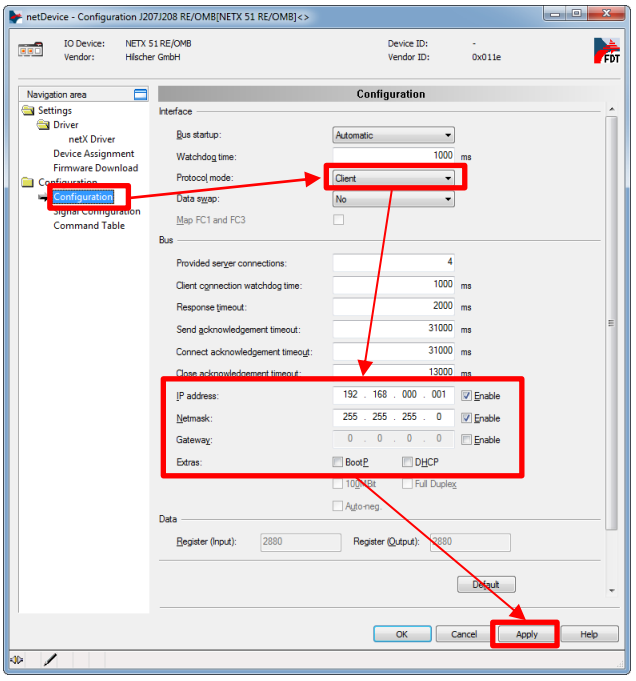
This open SYCON.net. Double click on J207/J208



Select “Configuration” branch, select “Client”.

Uncheck “DHCP”, enable and set “IP address” and “Netmask”. Here, you specify the client address.

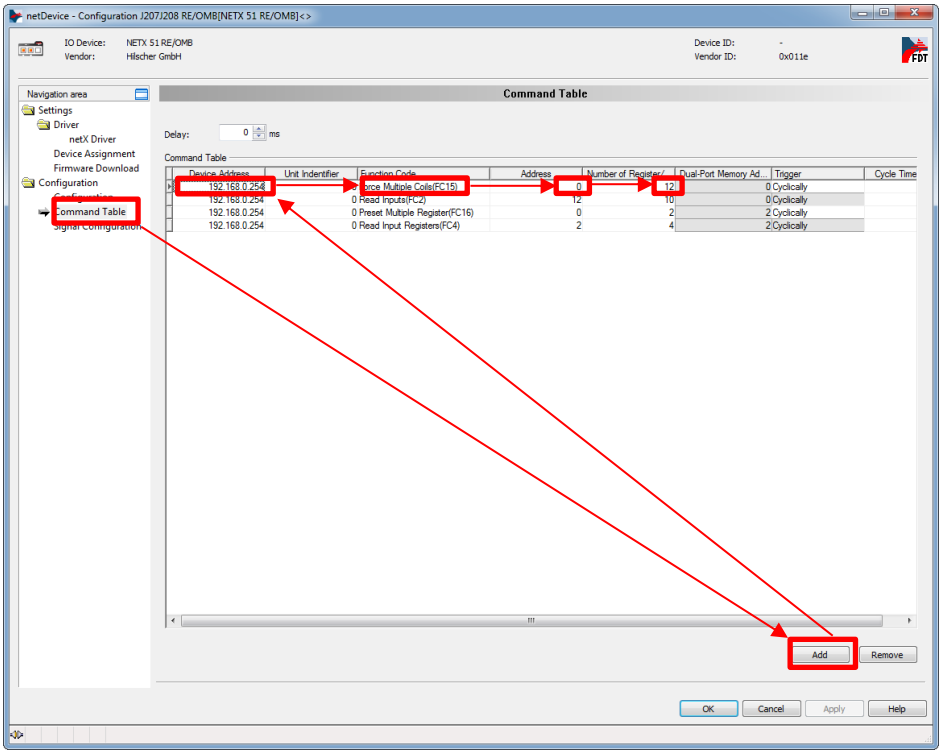
“Apply”.



Select “Command table” branch.

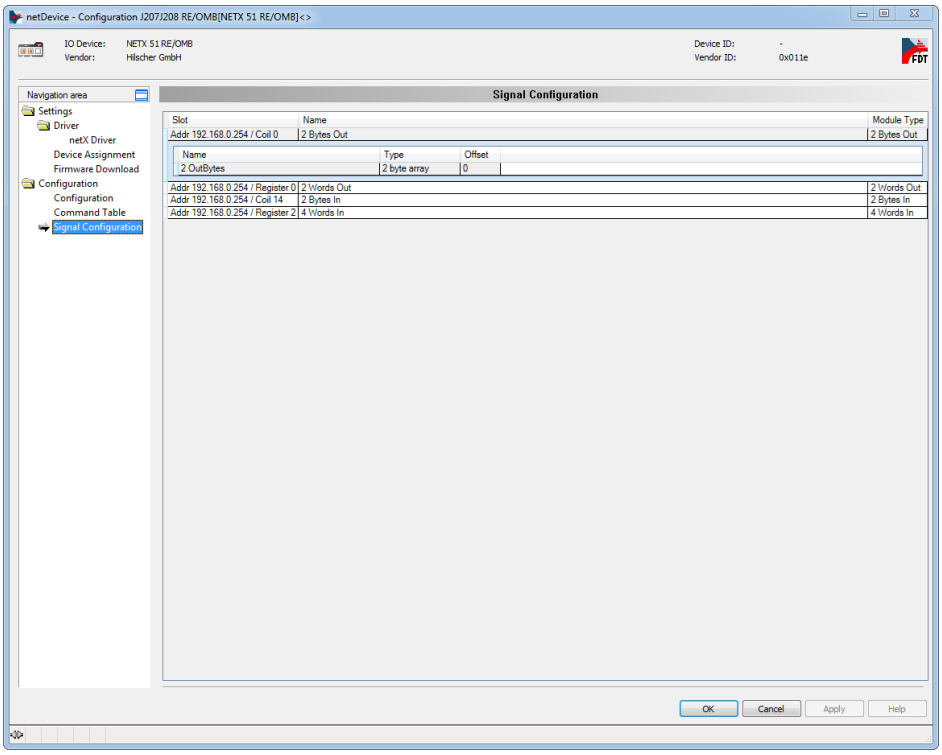
Add as many device as needed

For each device, you specify the server IP address in “Device Address”, Select the proper “Function code”, specify the start “Address” and the “Number of Register”.



Select “Signal configuration” branch.

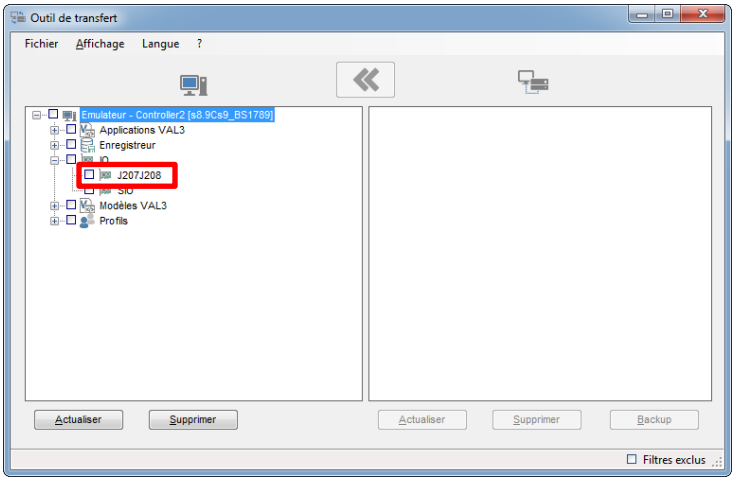
Split “Bytes Out”, “Words Out”, “Bytes In”, “Words In” as defined in the exchange table.



When finished, press “Ok” then “File”, “Save” and close SYCON.

With “Transfer manager”, send the “J207/J208” configuration to the controller.

Reboot the controller in order to take into account the new configuration.

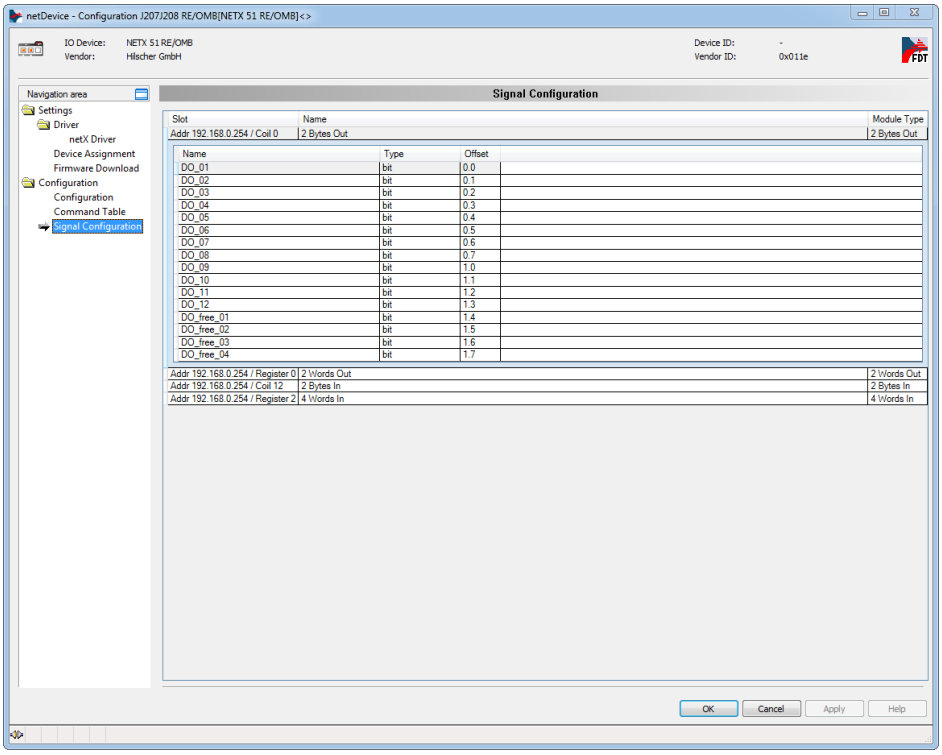


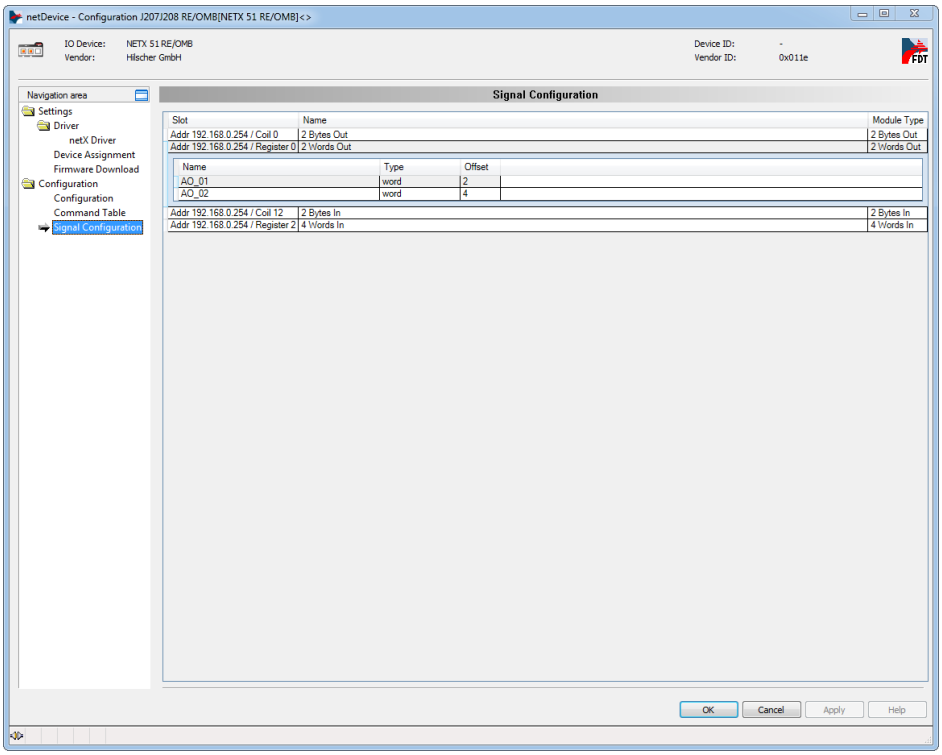
7.1 Example with a CS9 configured as a client connected to a CS8C as server

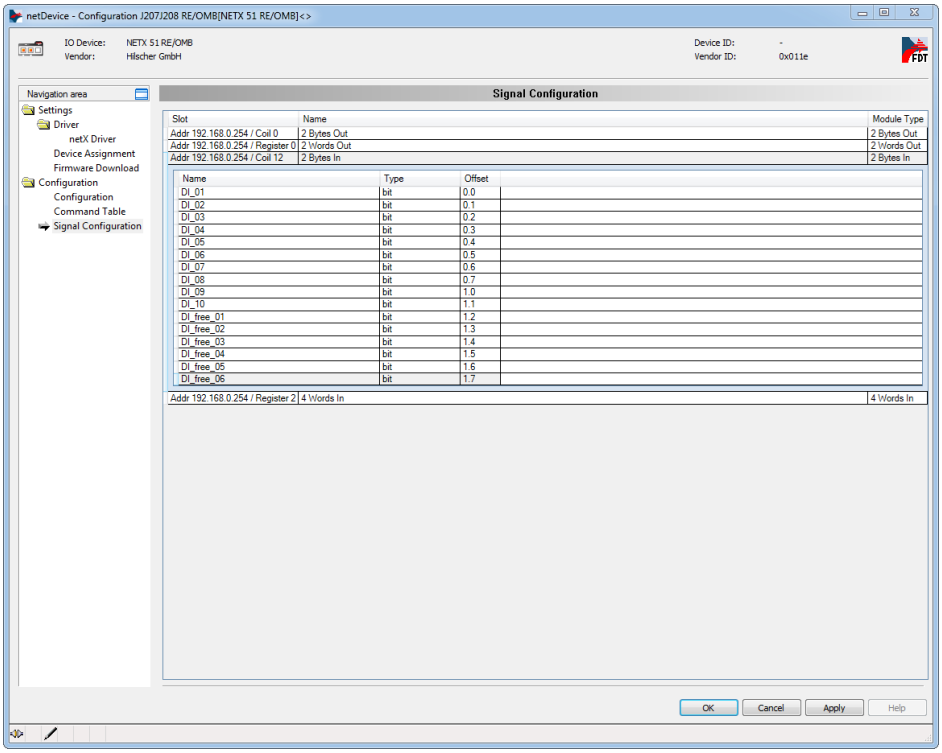
We have:

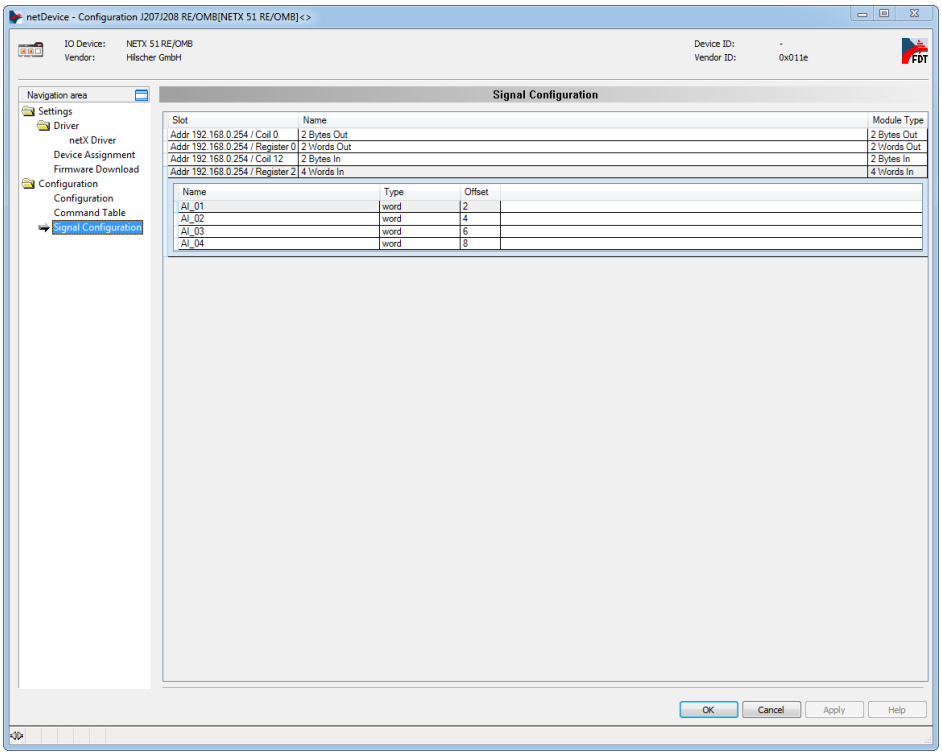
- 12 digital outputs, FC15, start at address 0, quantity 12
- 10 digital inputs, FC2, start at address 12 (the 12 digital outputs), quantity 10
- 2 analog outputs, FC16, start at address 0, quantity 2
- 4 analog inputs, FC4, start at address 2 (the 2 analog outputs), quantity 4

CS9 client (master)				CS8C, server (slave)			
	Word	Bit	Output name	Input name	Bit	Word	
Digital output	1	1	DO_01	DI_01	9	1	Digital input
		2	DO_02	DI_02	10		
		3	DO_03	DI_03	11		
		4	DO_04	DI_04	12		
		5	DO_05	DI_05	13		
		6	DO_06	DI_06	14		
		7	DO_07	DI_07	15		
		8	DO_08	DI_08	16		
		9	DO_09	DI_09	1		
		10	DO_10	DI_10	2		
		11	DO_11	DI_11	3		
		12	DO_12	DI_12	4		
		13	DO_free_01	DI_free_01	5		
		14	DO_free_02	DI_free_02	6		
		15	DO_free_03	DI_free_03	7		
		16	DO_free_04	DI_free_04	8		
Digital input	1	1	DI_01	DO_01	9	1	Digital output
		2	DI_02	DO_02	10		
		3	DI_03	DO_03	11		
		4	DI_04	DO_04	12		
		5	DI_05	DO_05	13		
		6	DI_06	DO_06	14		
		7	DI_07	DO_07	15		
		8	DI_08	DO_08	16		
		9	DI_09	DO_09	1		
		10	DI_10	DO_10	2		
		11	DI_free_01	DO_free_01	3		
		12	DI_free_02	DO_free_02	4		
		13	DI_free_03	DO_free_03	5		
		14	DI_free_04	DO_free_04	6		
		15	DI_free_05	DO_free_05	7		
		16	DI_free_06	DO_free_06	8		
Analog output	2		AO_01	AI_01		2	Analog input
	3		AO_02	AI_02		3	
Analog input	2		AI_01	AO_01		2	Analog output
	3		AI_02	AO_02		3	
	4		AI_03	AO_03		4	
	5		AI_04	AO_04		5	


netDevice - Configuration J207J208 RE/OMB[NETX 51 RE/OMB]< >
IO Device: NETX 51 RE/OMB
Vendor: Hilscher GmbH
Device ID: -
Vendor ID: 0x011e
Navigation area
Settings
Driver
netX Driver
Device Assignment
Firmware Download
Configuration
Configuration
Command Table
Signal Configuration
Signal Configuration
Slot
Name
Module Type
Addr 192.168.0.254 / Coil 0
2 Bytes Out
2 Bytes Out
Name
Type
Offset
DO_01 bit 0.0
DO_02 bit 0.1
DO_03 bit 0.2
DO_04 bit 0.3
DO_05 bit 0.4
DO_06 bit 0.5
DO_07 bit 0.6
DO_08 bit 0.7
DO_09 bit 1.0
DO_10 bit 1.1
DO_11 bit 1.2
DO_12 bit 1.3
DO_free_01 bit 1.4
DO_free_02 bit 1.5
DO_free_03 bit 1.6
DO_free_04 bit 1.7
Addr 192.168.0.254 / Register 0
2 Words Out
2 Words Out
Addr 192.168.0.254 / Coil 12
2 Bytes In
2 Bytes In
Addr 192.168.0.254 / Register 2
4 Words In
4 Words In
OK
Cancel
Apply
Help


netDevice - Configuration J207J208 RE/OMB[NETX 51 RE/OMB]< >
IO Device: NETX 51 RE/OMB
Vendor: Hilscher GmbH
Device ID: -
Vendor ID: 0x011e
Navigation area
Settings
Driver
netX Driver
Device Assignment
Firmware Download
Configuration
Configuration
Command Table
Signal Configuration
Signal Configuration
Slot
Name
Module Type
Addr 192.168.0.254 / Coil 0
2 Bytes Out
2 Bytes Out
Addr 192.168.0.254 / Register 0
2 Words Out
2 Words Out
Name
Type
Offset
AO_01 word 2
AO_02 word 4
Addr 192.168.0.254 / Coil 12
2 Bytes In
2 Bytes In
Addr 192.168.0.254 / Register 2
4 Words In
4 Words In
OK
Cancel
Apply
Help


netDevice - Configuration J207J208 RE/OMB[NETX 51 RE/OMB] < >
IO Device: NETX 51 RE/OMB
Vendor: Hilscher GmbH
Device ID: -
Vendor ID: 0x011e
Navigation area
Settings
Driver
netX Driver
Device Assignment
Firmware Download
Configuration
Configuration
Command Table
Signal Configuration
Signal Configuration
Slot
Name
Module Type
Addr 192.168.0.254 / Coil 0
2 Bytes Out
2 Bytes Out
Addr 192.168.0.254 / Register 0
2 Words Out
2 Words Out
Addr 192.168.0.254 / Coil 12
2 Bytes In
2 Bytes In
Name
Type
Offset
DI_01 bit 0.0
DI_02 bit 0.1
DI_03 bit 0.2
DI_04 bit 0.3
DI_05 bit 0.4
DI_06 bit 0.5
DI_07 bit 0.6
DI_08 bit 0.7
DI_09 bit 1.0
DI_10 bit 1.1
DI_free_01 bit 1.2
DI_free_02 bit 1.3
DI_free_03 bit 1.4
DI_free_04 bit 1.5
DI_free_05 bit 1.6
DI_free_06 bit 1.7
Addr 192.168.0.254 / Register 2
4 Words In
4 Words In
OK
Cancel
Apply
Help


netDevice - Configuration J207J208 RE/OMB[NETX 51 RE/OMB] < >
IO Device: NETX 51 RE/OMB
Vendor: Hilscher GmbH
Device ID: -
Vendor ID: 0x011e
Navigation area
Settings
Driver
netX Driver
Device Assignment
Firmware Download
Configuration
Configuration
Command Table
Signal Configuration
Signal Configuration
Slot
Name
Module Type
Addr 192.168.0.254 / Coil 0
2 Bytes Out
2 Bytes Out
Addr 192.168.0.254 / Register 0
2 Words Out
2 Words Out
Addr 192.168.0.254 / Coil 12
2 Bytes In
2 Bytes In
Addr 192.168.0.254 / Register 2
4 Words In
4 Words In
Name
Type
Offset
AI_01 word 2
AI_02 word 4
AI_03 word 6
AI_04 word 8
OK
Cancel
Apply
Help

8 Appendix

8.1 Modbus function code

Extract from Modbus specification (http://www.modbus.org/specs.php)

Data type	Mode	Function name	Function code (FC)
Bit	Digital input (read only)	Read Discrete Inputs	2
	Digital output (read / write)	Read Coils	1
		Write Single Coil	5
		Write Multiple Coils	15
Word (16 bits)	Analog input (read only)	Read Input Registers	4
	Analog output (read / write)	Read Holding Registers	3
		Write Single Register	6
		Write Multiple registers	16