APPLICATION EXAMPLE

AC500 V3 – MODBUS TCP

COMMUNICATION WITH FIXED AND DYNAMIC REGISTER MAPPING

Contents

1 Disclaimer

A. For customers domiciled outside Germany /

Für Kunden mit Sitz außerhalb Deutschlands

„Warranty, Liability:

The user shall be solely responsible for the use of this products described within this file. ABB shall be under no warranty whatsoever. ABB's liability in connection with application of the products or examples provided or the files included within these products, irrespective of the legal ground, shall be excluded. The exclusion of liability shall not apply in the case of intention or gross negligence. The present declaration shall be governed by and construed in accordance with the laws of Switzerland under exclusion of its conflict of laws rules and of the Vienna Convention on the International Sale of Goods (CISG)."

„Gewährleistung und Haftung:

Der Nutzer ist allein für die Verwendung des in diesem Dokument beschriebenen Produkte und beschriebenen Anwendungsbeispiele verantwortlich.

ABB unterliegt keiner Gewährleistung. Die Haftung von ABB im Zusammenhang mit diesem Anwendungsbeispiel oder den in dieser Datei enthaltenen Dateien - gleich aus welchem Rechtsgrund - ist ausgeschlossen. Dieser Ausschluss gilt nicht im Falle von Vorsatz oder grober Fahrlässigkeit. Diese Erklärung unterliegt Schweizer Recht unter Ausschluss der Verweisungsnormen und des UN-Kaufrechts (CISG)."

B. Nur für Kunden mit Sitz in Deutschland

„Gewährleistung und Haftung:

Die in diesem Dokument beschriebenen Anwendungsbeispiele oder enthaltenen Dateien beschreiben eine mögliche Anwendung der AC500 bzw. zeigen eine mögliche Einsatzart. Sie stellen nur Beispiele für Programmierungen dar, sind aber keine fertigen Lösungen. Eine Gewähr kann nicht übernommen werden.

Der Nutzer ist für die ordnungsgemäße, insbesondere vollständige und fehlerfreie Programmierung der Steuerungen selbst verantwortlich. Im Falle der teilweisen oder ganzen Übernahme der Programmierbeispiele können gegen ABB keine Ansprüche geltend gemacht werden.

Die Haftung von ABB, gleich aus welchem Rechtsgrund, im Zusammenhang mit den Anwendungsbeispielen oder den in dieser Datei enthaltenen Beschreibung wird ausgeschlossen. Der Haftungsausschluss gilt jedoch nicht in Fällen des Vorsatzes, der groben Fahrlässigkeit, bei Ansprüchen nach dem Produkthaftungsgesetz, im Falle der Verletzung des Lebens, des Körpers oder der Gesundheit oder bei schuldhafter Verletzung einer wesentlichen Vertragspflicht. Im Falle der Verletzung einer wesentlichen Vertragspflicht ist die Haftung jedoch auf den vertragstypischen, vorhersehbaren Schaden begrenzt, soweit nicht zugleich ein anderer der in Satz 2 dieses Unterabsatzes erwähnten Fälle gegeben ist. Eine Änderung der Beweislast zum Nachteil des Nutzers ist hiermit nicht verbunden.

Es gilt materielles deutsches Recht unter Ausschluss des UN-Kaufrechts."

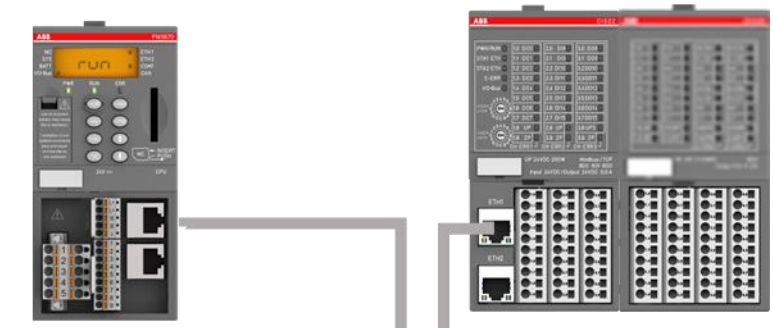
2 Introduction

2.1 Compatibility

The application example explained in this document has been used with the below engineering system versions. They should also work with other versions, nevertheless some small adaptations may be necessary, for future versions.

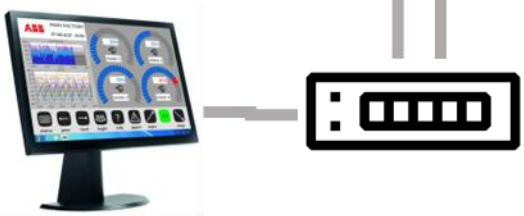
- AC500 V3 PLC
- Automation Builder 2.4.1 or newer

3 Hardware Setup



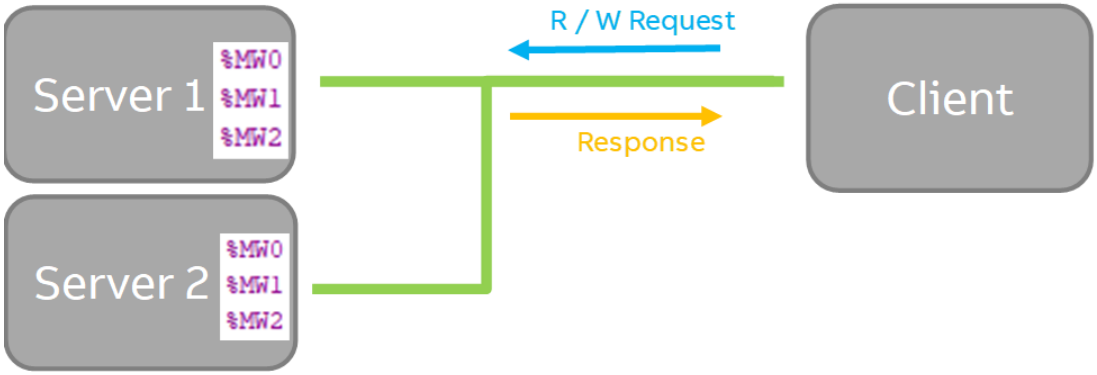
PM5675
192.168.3.131
255.255.240.0
Port: 502
Client

CI522 + DO526
192.168.3.101
255.255.240.0
Port: 502
Server



Engineering PC
192.168.3.135
255.255.240.0

4 Function Chart Modbus TCP



- In Modbus TCP mostly the Client is the PLC or control unit. Servers are for example sensos, actors or another AC500 PLC. The Client reads or writes the Modbus register values from the servers. These are declared as e.g. %MW0.
- Modbus is using the client – server architecture, this means the client starts a read or write request and is able to communicate to multiple servers simultaneously.
- Standard Port for Modbus TCP is Port 502.

4.1 Function codes

A function code is a command in number format, which specifies if the message is a read or write request, if one or multiple bits are manipulated. In this example, only two function codes are used.

Note:

A description about the ModbusTCP and its functionality is available in the Automation Builder online help.

PLC Automation with V3 CPUs > PLC integration (hardware) > System technology for AC500 V3 products > System technology of CPU and overall system > Communication with Modbus TCP > Protocol description.

4.1.1 Function code 16

Function code 16 is a basic functionality for Modbus. It is a write command for multiple server output registers. In this example, for the CI522 Module only one register will be written.

4.1.2 Function code 3

Function code 3 is a basic functionality for Modbus. It is a read command for multiple server output registers. In this example, for the CI522 Module only one register will be read.

4.1.3 Function code 22

Not only the AC500 CPUs and the CI52x modules can use the function code 22, but also the CM577 and the CM597 ethernet coupler. This function block is used to write within a register and calculates the new register value bitwise using the existing value and the sent value. For calculation AND parameter or the OR parameter can be used.

4.1.4 Function code 23

Like function code 22, function code 23 is also supported on the CM577 and the CM597 ethernet coupler. This function code can not only write multiple registers, but also is able to read multiple registers. This advantage allows to read and write multiple S500 devices in one rush. For this, they have to be attached directly to an AC500 PLC or an CI52x module (dynamic mode).

4.2 Address register in Server

The server provides register, which will be written or read. The read- and writeable registers from the CI522 module are similar to the registers in the CI521 Modbus module. They start at 1000_{hex} and go to $2AFF_{hex}$. In the table below, you see the fixed start addresses of the inputs and outputs of up to 10 devices connected to the CI module. The two columns on the right show the usable function codes.

Register (hex)	Description	Readable by	Writeable by
FFA	diagnosis	3,4,23	x
1000	Inputs CI	3,4,23	x
1100	Inputs 1. EXP	3,4,23	x
...	...		x
1A00	Inputs 10. EXP	3,4,23	x
2000	Outputs CI	3,23	6,16,23
2100	Outputs 1. EXP	3,23	6,16,23
...	...		
2A00	Outputs 10. EXP	3,23	6,16,23

Note:

The size of the registers of any device can be found in the Automaton Builder help>Contents. For DO526 navigate to:

Help>Contents>PLC Automation>PLC Automation with V3 CPUs>PLC integration (hardware)>Device specification>I/O modules>Digital I/O modules>S500>DO526 – Digital output module>Internal data exchange: Digital outputs (bytes).

4.3 Fixed mapping

In the fixed mapping, every device uses a 100_{hex} (100 Word) range for its addresses, whether it needs that much or not. With this configuration there will be unused address gaps. A good advantage is, that is that you exactly know at which address a particular module starts. The table below shows different modules. Every module uses exactly a 100_{hex} address range. If a certain module has no inputs or outputs the corresponding registers remain empty.

Fixed Mapping			
Register (hex)	Description	Type	Data
1000	Inputs CI	8 DC, 8 DI, FC	4 BYTE + 4 WORD
1100	Inputs AX522	8 AI	8 WORD
1200	Inputs DC532	16 DI, 16 DC	4 BYTE
1300	Inputs AX521	4 AI	4 WORD
1400	Inputs DC523	24 DC	3 BYTE
1500	Inputs DC532	16 DI, 16 DC	4 BYTE
1600	Inputs AO523	---	---
1700	Inputs AI523	16 AI	16 WORD
1800	Inputs DI524	32 DI	4 BYTE
1900	Inputs AX522	8 AI	8 WORD
1A00	Inputs DC523	24 DC	3 BYTE
2000	Outputs CI	8 DC, 8DO, FC	4 BYTE + 8 WORD
2100	Outputs AX522	8 AO	8 WORD
2200	Outputs DC532	16 DC	2 BYTE
2300	Outputs AX521	4 AO	4 WORD
2400	Outputs DC523	24 DC	3 BYTE
2500	Outputs DC532	16 DC	2 BYTE
2600	Outputs AO523	16 AO	16 WORD
2700	Outputs AI523	---	---
2800	Outputs DI524	---	---
2900	Outputs AX522	8 AO	8 WORD
2A00	Outputs DC523	24 DC	3 BYTE

Advantage to dynamic mapping:

It is easy to calculate the start addresses of each device. This is an advantage if you do not know the exact size of the individual registers.

4.4 Dynamic mapping

In dynamic mapping, the register addresses are put together, so there will not be unused gaps. Every module has its own size of register addresses. To know, at which address a particular module starts, you have to calculate the start address. Every device has a certain range of used addresses. In the Data column, you can get the size of the device registers. For example, CI522 input register has a size of 6 Word, so the start address of the next input register of the next module is $1006_{hex} + 6_{hex} = 1006_{hex}$. The same calculation has to be done with the output registers due to their address rage beginning at 2000_{hex}. If a certain module has no inputs or outputs the corresponding registers remain empty.

Dynamic Mapping

Register (hex)	Description	Type	Data
1000	Inputs CI	8 DC, 8 DI, FC	4 BYTE + 4 WORD
1006	Inputs AX522	8 AI	8 WORD
100E	Inputs DC532	16 DI, 16 DC	4 BYTE
1010	Inputs AX521	4 AI	4 WORD
1014	Inputs DC523	24 DC	3 BYTE
1016	Inputs DC532	16 DI, 16 DC	4 BYTE
---	Inputs AO523	---	---
1018	Inputs AI523	16 AI	16 WORD
1028	Inputs DI524	32 DI	4 BYTE
102A	Inputs AX522	8 AI	8 WORD
1032	Inputs DC523	24 DC	3 BYTE
2000	Outputs CI	8 DC, 8DO, FC	4 BYTE + 8 WORD
200A	Outputs AX522	8 AO	8 WORD
2012	Outputs DC532	16 DC	2 BYTE
2013	Outputs AX521	4 AO	4 WORD
2017	Outputs DC523	24 DC	3 BYTE
2019	Outputs DC532	16 DC	2 BYTE
201A	Outputs AO523	16 AO	16 WORD
---	Outputs AI523	---	---
---	Outputs DI524	---	---
202A	Outputs AX522	8 AO	8 WORD
2032	Outputs DC523	24 DC	3 BYTE

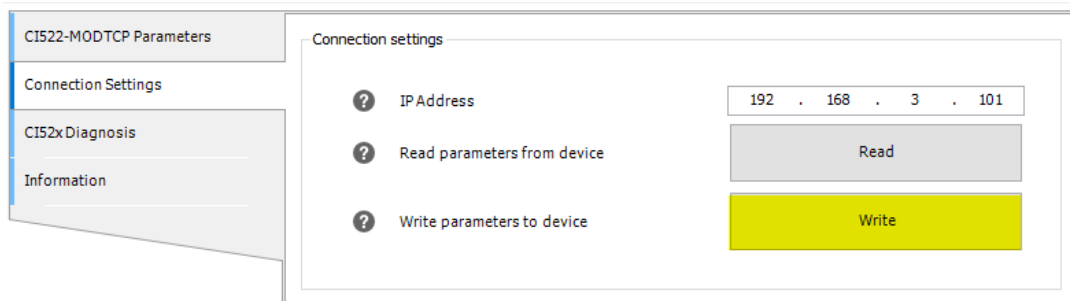
Advantage to fixed mapping:

It is easy to set multiple registers in one write command. For example reset commands.

4.5 Program example description

The program example uses a PM5675 PLC and a CI522 + DO526 module. To test the two mapping methods, we set register variables which are outputs at the same time. At first, we must set the configuration for the fixed and dynamic mapping in the CI module. In the example, the configuration is already done and you only have to upload the config into the CI522 module.

Parameter	Type	Value	Default Value	Unit	Description
Error LED / Failsafe function	Enumeration of BYTE	On	On		Error LED off by error class
Master-IP 0	ARRAY[0..3] OF BYTE	[0,0,0,0]	[0,0,0,0]		Master-IP 0 for write restriction
Master-IP 1	ARRAY[0..3] OF BYTE	[0,0,0,0]	[0,0,0,0]		Master-IP 1 for write restriction
Master-IP 2	ARRAY[0..3] OF BYTE	[0,0,0,0]	[0,0,0,0]		Master-IP 2 for write restriction
Master-IP 3	ARRAY[0..3] OF BYTE	[0,0,0,0]	[0,0,0,0]		Master-IP 3 for write restriction
Master-IP 4	ARRAY[0..3] OF BYTE	[0,0,0,0]	[0,0,0,0]		Master-IP 4 for write restriction
Master-IP 5	ARRAY[0..3] OF BYTE	[0,0,0,0]	[0,0,0,0]		Master-IP 5 for write restriction
Master-IP 6	ARRAY[0..3] OF BYTE	[0,0,0,0]	[0,0,0,0]		Master-IP 6 for write restriction
Master-IP 7	ARRAY[0..3] OF BYTE	[0,0,0,0]	[0,0,0,0]		Master-IP 7 for write restriction
Timeout for bus supervision	BYTE(0..255)	0	0		Timeout for bus supervision. Number of 10ms
I/O mapping structure	Enumeration of BYTE	Fixed mapping	Fixed mapping		Fixed Mapping means each module has its own
Check supply	Enumeration of BYTE	On	On		Check supply
Input delay	Enumeration of BYTE	8 ms	8 ms		Input delay of digital inputs
Fast counter	Enumeration of BYTE	0-No counter	0-No counter		Operating mode fast counter



Next, navigate to the main visualization and choose via tabs between fixed or dynamic mapping. As mentioned above, in fixed mapping you have to execute two write commands. In dynamic mapping, you can set everything with one command.

ABB AG
Eppelheimer Straße 82
69123 Heidelberg, Germany
Phone: +49 62 21 701 1444
Fax: +49 62 21 701 1382
E-Mail: plc.support@de.abb.com
www.abb.com/plc