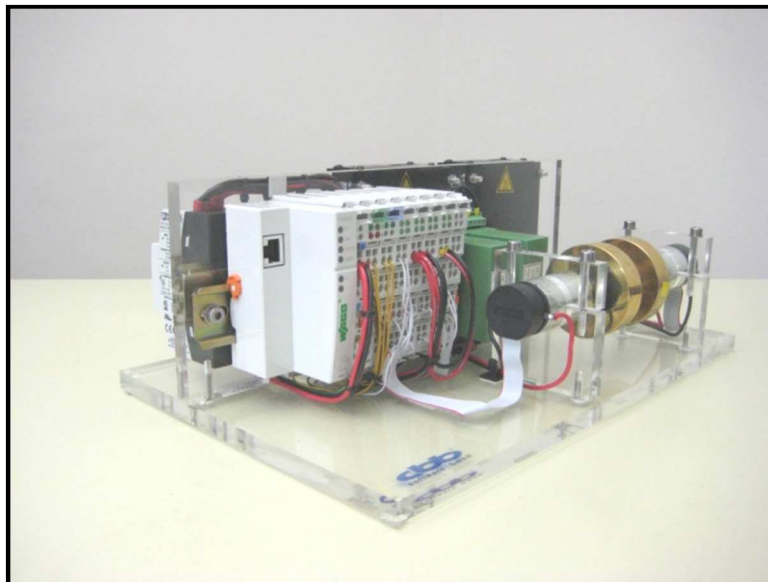


CATS

Control and Automation Training System

User Guide



Version 1.0.0.2
(12th August 2010)

Content

Introduction	4
1. Overview	5
1.1 CATS-1M-Basic.....	6
1.2 CATS-1M-Advanced	6
1.3 CATS-2M-Basic.....	7
1.4 CATS-2M-Advanced	7
1.5 Individual Laboratory Experiments	8
1.6 From the Individual Laboratory Experiment to the Network Experiment.....	8
1.7 Automation/Automatic Control Engineering.....	8
2. Fieldbus Coupler and Terminals	9
2.1 ETHERNET TCP/IP Fieldbus Coupler WAGO 750-342	9
2.2 2-Channels Analog Input Module $\pm 10V$ Wago 750-479	10
2.3 2-Channels Analog Output Module $\pm 10V$ Wago 750-556.....	12
2.4 Incremental Encoder Interfaces WAGO 750-637	13
2.5 End Module WAGO 750-600.....	16
3. CATS Wiring Diagram.....	17
3.1 CATS-1M-Basic.....	17
3.2 CATS-1M-Advanced	18
3.3 CATS-2M-Basic.....	19
3.4 CATS-2M-Advanced	20
4. How to Assign a Static IP Address to Fieldbus Coupler.....	21
4.1 User BootP Server to Assign a Static IP Address	21
4.2 Testing the Function of the Fieldbus Node.....	25
5. LabMap [®] Introduction	26
5.1 Overview	26
5.2 Concepts	26
5.3 From the Individual Laboratory Experiment to Network Experiment.....	27
6. How to Create LabMap [®] Handle.....	28
6.1 LabMap [®] Input and Output Handles.....	28
6.2 Creating New Handles	29
7. LabMap [®] Configuration.....	33

8. Default LabMap® Handles Configuration	35
8.1 CATS-1M-Basic.....	35
8.2 CATS-1M-Advanced	36
8.3 CATS-2M-Basic.....	37
8.4 CATS-2M-Advanced	38

Introduction

Studies at schools and universities are becoming ever more compact and compressed; therefore, the content of automatic control engineering, automation technology, and control engineering must be transferred from theory into practice immediately efficiently, effectively and seamlessly. With the “**CATS**” experimental equipment, the theoretical treatment, simulation and testing of automation scenarios can be structured step by step as they are in industry so they are easy to understand and can be traced practically.

The components of the “**CATS**” experimental equipment can be combined with commercially-available software (e.g. MATLAB/Simulink or LabVIEW) to build a complete individual laboratory space. Experiments with a flexible real time requirement (> 10 ms) can be operated under MS Windows[®] without an additional real time environment.

Contact:

cbb software GmbH
Charlottenstraße 1
23560 Lübeck
Germany

Tel.: 0049 451 39 77 10
Fax: 0049 451 39 77 129
Mail: mailbox@cbb-software.com

Contact person:

Mr. Dipl. Ing. Sascha Heinecke	(Sascha.Heinecke@cbb-software.com)
Mr. Dipl. Ing. Ruifeng Zhang	(Ruifeng.Zhang@cbb-software.com)

1. Overview

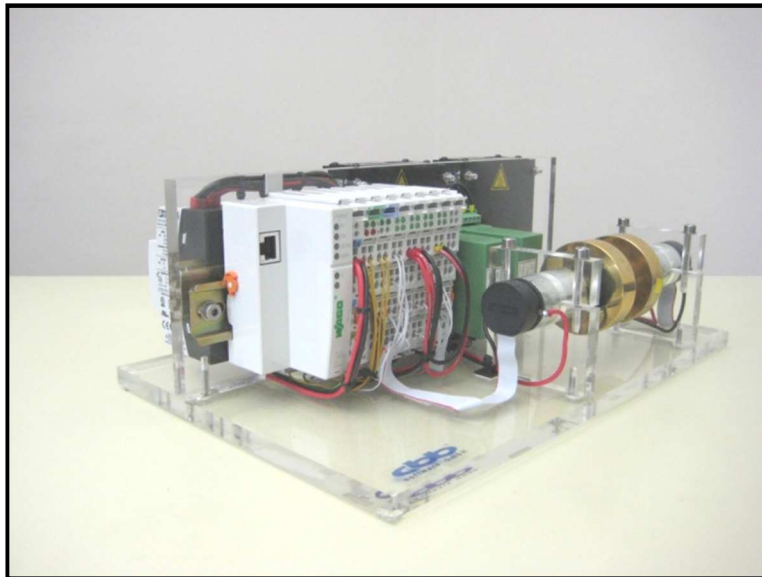


Figure 1 CATS-2M-Advanced Model

As shown in Figure 1, CATS-2M-Advanced Model provides the possibility to measure motors' current during run-time. Therefore, motor can be controlled not only by voltage but also by current. With current measurement, Torque measurement can also be implemented. Controlling 2 motors concurrently also makes the "CATS" device enable to simulate one kind of an electrical motor or component test-stand, as being utilized in the industry.

"CATS-2M-Advanced Model" provides full functionality of CATS devices, nevertheless, according to different requirements, different options are listed in the following table:

Name	Motor	Voltage control	Current Control	Torque measurement	Test-bed simulating
CATS-1M-Basic	1	Yes	No	No	No
CATS-1M-Advanced	1	Yes	Yes	Yes	No
CATS-2M-Basic	2	Yes	No	No	Yes
CATS-2M-Advanced	2	Yes	Yes	Yes	Yes

Table 1 CATS different layouts

All types of CATS device are open to be extended to any level. All necessary boreholes and threads are pre assembled and preconfigured.

For more information about CATS extension, please feel free contact cbb software GmbH: mailbox@cbb-software.com

1.1 CATS-1M-Basic

Hardware:

- Plexiglass Basic Rack
- Phoenix Contact Power Supply
- Maxon DC Motor (18W) with Incremental Encoder
- Maxon DC Servoamplifier
- WAGO ModBus Coupler
- WAGO Incremental Encoder Interface
- WAGO 2-Channel Analog Output (-10V...10V DC)
- WAGO ModBus End Modules
- Flywheel
- Completely assembled (wired and mounted)

Software:

- LabMap[®] (Student Licence)
- LabNet[®]
- Interface to Matlab/Simulink[®] or NI LabVIEW[®] (each current Version)

1.2 CATS-1M-Advanced

Hardware:

- Plexiglass Basic Rack
- Phoenix Contact Power Supply
- Maxon DC Motor (18W) with Incremental Encoder
- Maxon DC Servoamplifier
- WAGO ModBus Coupler
- WAGO Incremental Encoder Interface
- WAGO 2-Channel Analog Output (-10V...10V DC)
- WAGO 2-Channel Analog Input (-10V...10V DC)
- Current Sensor for the DC Engine ("Torquemetre")
- WAGO ModBus End Modules
- Flywheel
- Completely assembled (wired and mounted)

Software:

- LabMap[®] (Student Licence)
- LabNet[®]
- Interface to Matlab/Simulink[®] or NI LabVIEW[®] (each current Version)

1.3 CATS-2M-Basic

Hardware:

- Plexiglass Basic Rack
- Phoenix Contact Power Supply
- 2 x Maxon DC Motor (18W) with Incremental Encoder
- 2 x Maxon DC Servoamplifier
- WAGO ModBus Coupler
- 2 x WAGO Incremental Encoder Interface
- WAGO 2-Channel Analog Output (-10V...10V DC)
- WAGO ModBus End Modules
- 2 x Flywheel
- Completely assembled (wired and mounted)

Software:

- LabMap[®] (Student Licence)
- LabNet[®]
- Interface to Matlab/Simulink[®] or NI LabVIEW[®] (each current Version)

1.4 CATS-2M-Advanced

Hardware:

- Plexiglass Basic Rack
- Phoenix Contact Power Supply
- 2 x Maxon DC Motor (18W) with Incremental Encoder
- 2 x Maxon DC Servoamplifier
- WAGO ModBus Coupler
- 2 x WAGO Incremental Encoder Interface
- WAGO 2-Channel Analog Output (-10V...10V DC)
- WAGO 2-Channel Analog Input (-10V...10V DC)
- 2 x Current Sensor for the DC Engine ("Torquemetre")
- WAGO ModBus End Modules
- 2 x Flywheel
- Completely assembled (wired and mounted)

Software:

- LabMap[®] (Student Licence)
- LabNet[®]
- Interface to Matlab/Simulink[®] or NI LabVIEW[®] (each current Version)

1.5 Individual Laboratory Experiments

With the “CATS” experimental equipment”, it shows how industrial automatic control engineering can be experienced. Work with the original industrial components is also practically-oriented and enables valuable learning.

1.6 Individual Laboratory Experiment to Network Experiment

The networking of individual laboratory experiments enables several students working with any experiment unit to access the network via middleware. With the help of the LabMap[®] software bus, each individual laboratory experiment on the network (Ethernet/ModBus) can be made accessible to several students. In principle, this works like a network printer.

More information about LabMap[®], please see **Chapter 5. LabMap[®] Introduction.**

1.7 Automation/Automatic Control Engineering



Figure 2 Production line

The production line depicted in Figure 2, on which several robots simultaneously assemble a vehicle, can be recreated as a laboratory experiment in the subject areas of automation technology/control engineering.

2. Fieldbus Coupler and Terminals

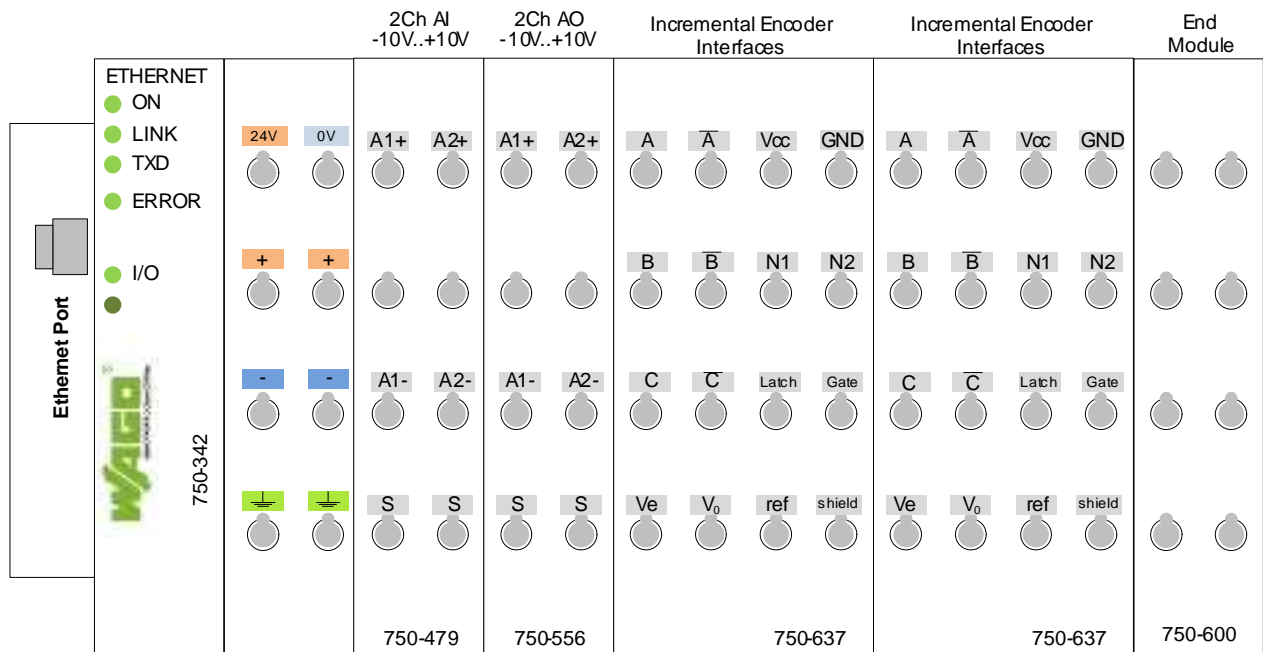


Figure 3 Fieldbus coupler and terminals CATS-2M-Advanced

2.1 ETHERNET TCP/IP Fieldbus Coupler WAGO 750-342

The WAGO 750-342 is an Input/Output module that can communicate via ModBus/TCP. It is connected via TCP/IP, UDP/IP or serial RS232. This protocol and enables this coupler to act as a decentral peripherie..

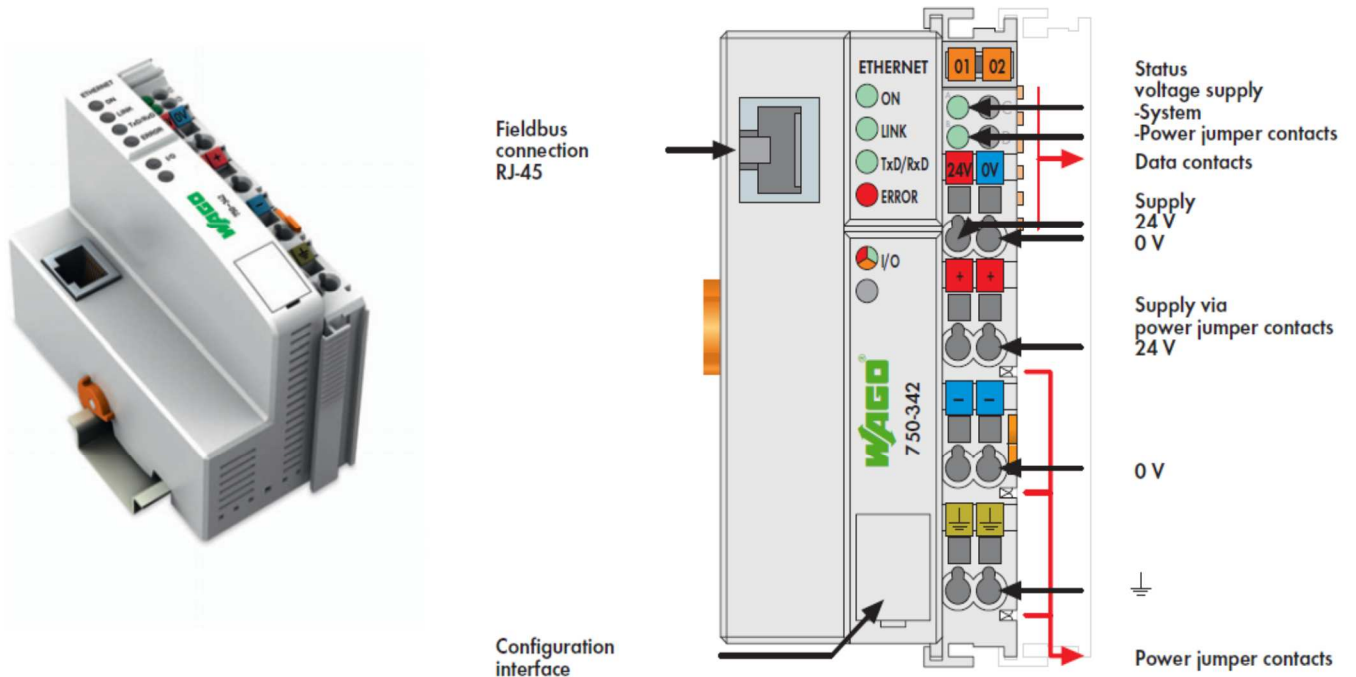


Figure 4 WAGO Ethernet TCP/IP Fieldbus Coupler

The ETHERNET TCP/IP fieldbus coupler supports a number of network protocols to send process data via ETHERNET TCP/IP. Using ETHERNET as a fieldbus makes universal data transmission between the factory and the office possible. Moreover, the ETHERNET TCP/IP fieldbus coupler offers remote maintenance, i.e. processes can be controlled regardless of the location. Process data exchange is done using the MODBUS/TCP protocol. The bus coupler supports all I/O modules and automatically configures, creating a local process image.

For more information about WAGO 750-342 devices, please visit:

Manual Document:

http://www.wago.com/wagoweb/documentation/750/eng_manu/342/m034200e.pdf

Data sheet:

http://www.wago.com/wagoweb/documentation/750/eng_dat/d07500342000en.pdf

2.2 2-Channels Analog Input Module $\pm 10V$ Wago 750-479

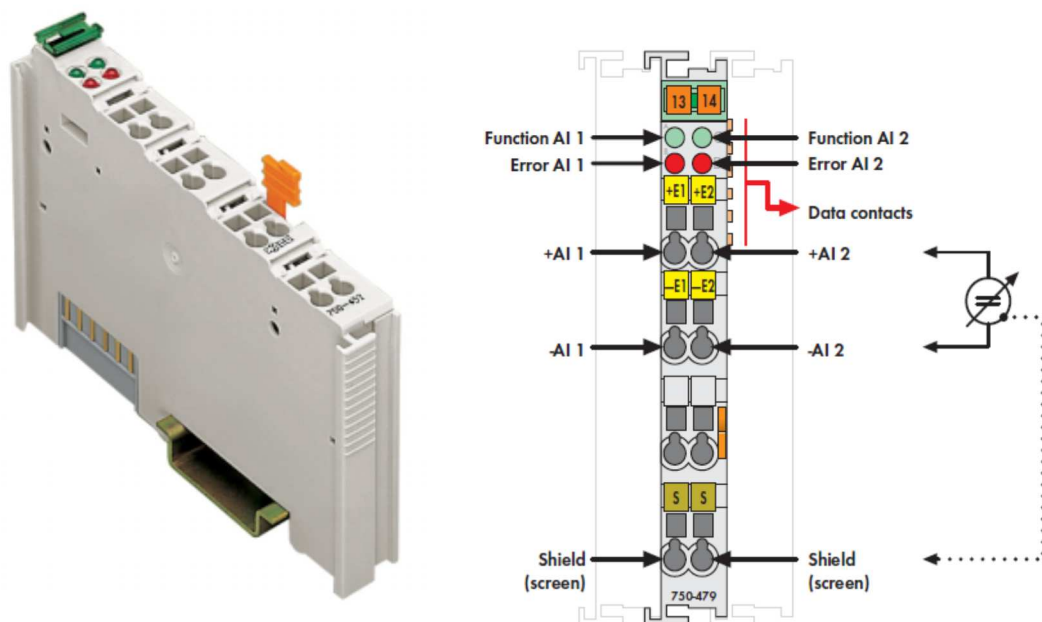


Figure 5 2-Channels Analog Input Module $\pm 10V$ Wago 750-479

This analog input module receives signals with standardized values of $\pm 10 V$. The module has two differential input channels and can receive differential signals via the connections +AI 1 and -AI 1 or +AI 2 and -AI 2. The Shield (screen) is directly connected to the DIN rail. A capacitive connection is made automatically when snapped onto the DIN rail.

The input signal of each channel is electrically isolated and will be transmitted with a resolution of 14 bits (13 bits + sign bit).

The operational readiness and trouble-free internal data bus communication of the channels are indicated via a green Function LED. Over range or underflow of the measuring range is indicated via a red error LED.

Any configuration of the input modules is possible when designing the fieldbus node. Grouping of module types is not necessary. The voltage supply is done via system voltage.

For the standard module 750-479, the input voltage ranging from -10 V to +10 V is scaled on the numerical values ranging from 0x8000 to 0x7FFF.

Process values of Module 750-479				
Input Voltage	Numerical value			Status-byte
± 10 V	Binary	Hex.	Dec.	Hex.
-10.0	1000.0000.0000.0001	0x8001	-32767	0x00
-7.5	1010.0000.0000.0000	0xA000	-24576	0x00
-5.0	1100.0000.0000.0000	0xC000	-16384	0x00
-2.5	1110.0000.0000.0000	0xE000	-8192	0x00
0.0	0000.0000.0000.0000	0x0000	0	0x00
2.5	0010.0000.0000.0000	0x2000	8192	0x00
5.0	0100.0000.0000.0000	0x4000	16384	0x00
7.5	0110.0000.0000.0000	0x6000	24576	0x00
10.0	0111.1111.1111.1111	0x7FFF	32767	0x00

Table 2 Process values of Module 750-479

For more information about WAGO 750-556 devices, please visit:

Manual Document:

http://www.wago.com/wagoweb/documentation/750/eng_manu/modules/m047900e.pdf

Data sheet:

http://www.wago.com/wagoweb/documentation/750/eng_dat/d047900e.pdf

2.3 2-Channels Analog Output Module $\pm 10V$ Wago 750-556

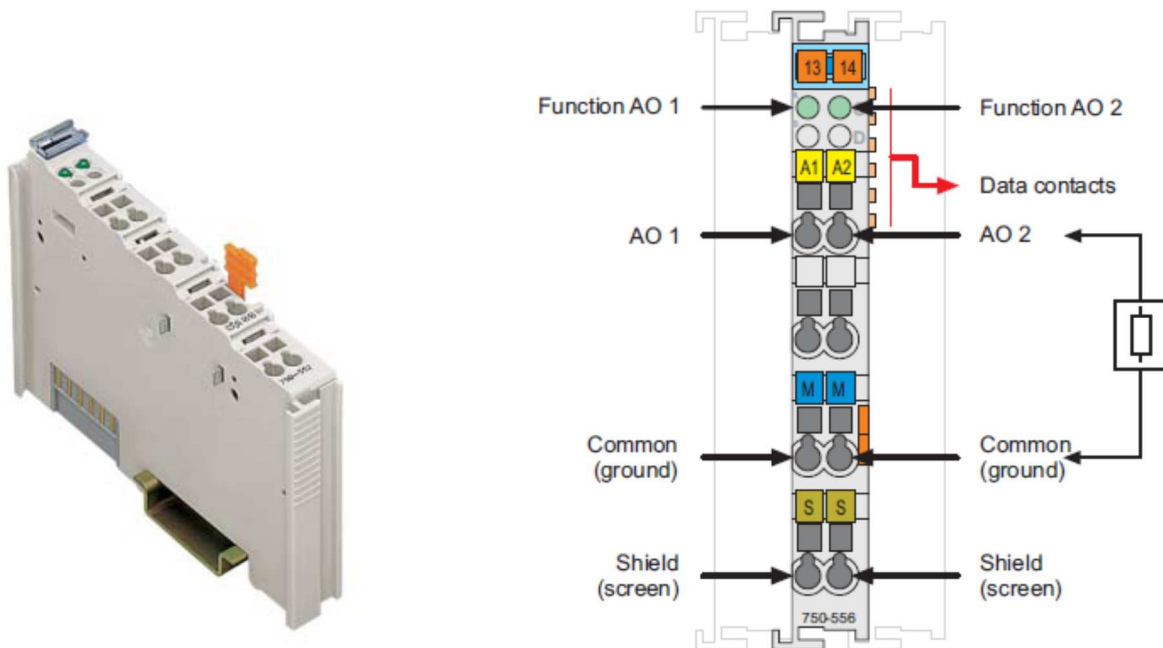


Figure 6 2-Channels Analog Output Module $\pm 10V$ WAGO 750-556

The analog output module 750-556 and its variation create a standardized signal of $\pm 10 V$. The module has two short circuit protected output channels and enables the direct wiring of two 2-conductor actuators to AO 1 and ground or AO 2 and ground. The signals are transmitted via AO 1 or AO 2.

The channels have a common ground and a shield (screen) (S). The shield (screen) is directly connected to the DIN rail. A capacitive connection is made automatically when snapped onto the DIN rail.

The input signal is electrically isolated and will be transmitted with a resolution of 12 bits. The operational readiness and the trouble-free internal data bus communication of the channels are indicated via a function LED.

The voltage supply is done via the internal system voltage.

For the standard module 750-556, the numerical values ranging from 0x8001 to 0x7FFF are scaled on the output voltage ranging from -10 V to +10 V.

Process values of Module 750-556				
Output Voltage	Numerical value			Status-byte
	Binary	Hex.	Dec.	Hex.
± 10 V				
-10.0	1000.0000.0000.0001	0x8001	-32767	0x00
-7.5	1010.0000.0000.0000	0xA000	-24576	0x00
-5.0	1100.0000.0000.0000	0xC000	-16384	0x00
-2.5	1110.0000.0000.0000	0xE000	-8192	0x00
0.0	0000.0000.0000.0000	0x0000	0	0x00
2.5	0010.0000.0000.0000	0x2000	8192	0x00
5.0	0100.0000.0000.0000	0x4000	16384	0x00
7.5	0110.0000.0000.0000	0x6000	24576	0x00
10.0	0111.1111.1111.1111	0x7FFF	32767	0x00

Table 3 Process values of Module 750-556

For more information about WAGO 750-556 devices, please visit:

Manual Document:

http://www.wago.com/wagoweb/documentation/750/eng_manu/modules/m055600e.pdf

Data sheet:

http://www.wago.com/wagoweb/documentation/750/eng_dat/d055600e.pdf

2.4 Incremental Encoder Interfaces WAGO 750-637

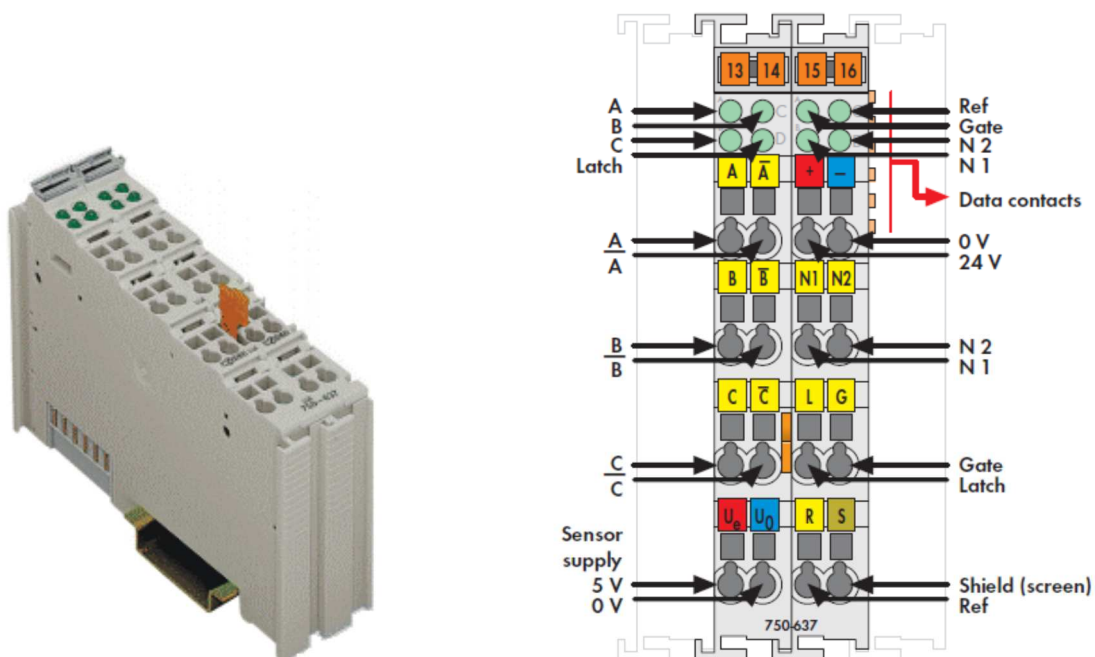


Figure 7 Incremental Encoder Interfaces WAGO 750-637

The I/O module 750-637 represents an interface for any type of incremental encoder with an RS 422 connection. The data width of the encoder module is 32 bits. The current counter reading, the latch value, the set value or the current speed can be mapped into the process data.

Inputs/Outputs	
A, /A, B, /B	Quadrature Input, RS 422
C, /C	Zero reference Input, RS 422
Latch, Gate, Ref	Input, 24V
N1, N2	Output, 24V

Table 4 Inputs/Outputs WAGO 750-637

A counter with quadrature decoder and a latch for the zero pulse can be read and activated by the PLC. The PLC is able to set the counter. Depending on the operating mode, the counter reading is taken into the latch register, if the edge at input “C” or “Latch” is positive. The speed (increments/ms) is automatically recorded and can be transmitted as an alternative to the latch value. The “Gate” input allows to lock the counter. The “Ref” input can be set to activate the zero mark “C”.

The cam outputs N1 and N2 indicate whether the counter value is within a defined range of values. The range can be adjusted for each cam via the PLC.



Attention:

24 V DC is required to supply the I/O modules; from this voltage also the voltage supply to the encoder (Ue, U0) is branched/ tapped.

Using the I/O module 750-637, a 6 byte input and output process image can be transferred to the fieldbus coupler / controller via two logical channels.

The set values are stored in 4 output bytes (D0, D1, D2, D3) and the process data are stored in 4 input bytes (D0, D1, D2, D3). Two control bytes (C0, C1) and two status bytes (S0, S1) are used to select process data and set values as well as to control the data flow.

Input data		Output data	
S0	Status byte S0	C0	Control byte C0
D0	Process data byte 0 (LSB) D0 Set	D0	Set value byte 0 (LSB)
D1	Process data byte 1	D1	Set value byte 1
S1	Status byte S1	C1	Control byte C1
D2	Process data byte 2	D2	Set value byte 2
D3	Process data byte 3 (MSB)	D3	Set value byte 3 (MSB)

Table 5 WAGO 750-637 Process Image

Bit 0 and bit 1 in the control byte C1 determines the process data.

- Counter value
- Latch value
- Velocity

- Set value

The setting is mirrored in status byte S1 in bit 0 and bit 1.

MapPZD (Control Byte C1 / Status Bytes S1, Bit 0 and Bit 1)		
Bit 1	Bit 0	Coding of the Process data
0	0	Counter value
0	1	Latch value
1	0	Velocity (Increments per milliseconds)
1	1	Set value

Table 6 Control Byte C1



Attention:

For CATS, the control byte C1 has to be set as “ $(10)_2$ ”, that is “2” in decimal, so that the encoder works in velocity mode.

For more information about WAGO 750-637 devices, please visit:

Manual Document:

http://www.wago.com/wagoweb/documentation/750/eng_manu/modules/m063700e.pdf

Data sheet:

http://www.wago.com/wagoweb/documentation/750/eng_dat/d07500637000en.pdf

2.5 End Module WAGO 750-600



Figure 8 End Module WAGO 750-600

The end module 750-600 is used to terminate the internal bus of a fieldbus node. This module completes the internal data circuit and ensures correct data flow.

The end module is placed at the end of a fieldbus node.

The end module must be used with all Couplers / Controllers of the WAGO-I/O-SYSTEM 750 to ensure correct data flow!

For more information about WAGO 750-637 devices, please visit:

Manual Document:

http://www.wago.com/wagoweb/documentation/750/eng_manu/modules/m060000e.pdf

Data sheet:

http://www.wago.com/wagoweb/documentation/750/eng_dat/d060000e.pdf

3. CATS Wiring Diagram

3.1 CATS-1M-Basic

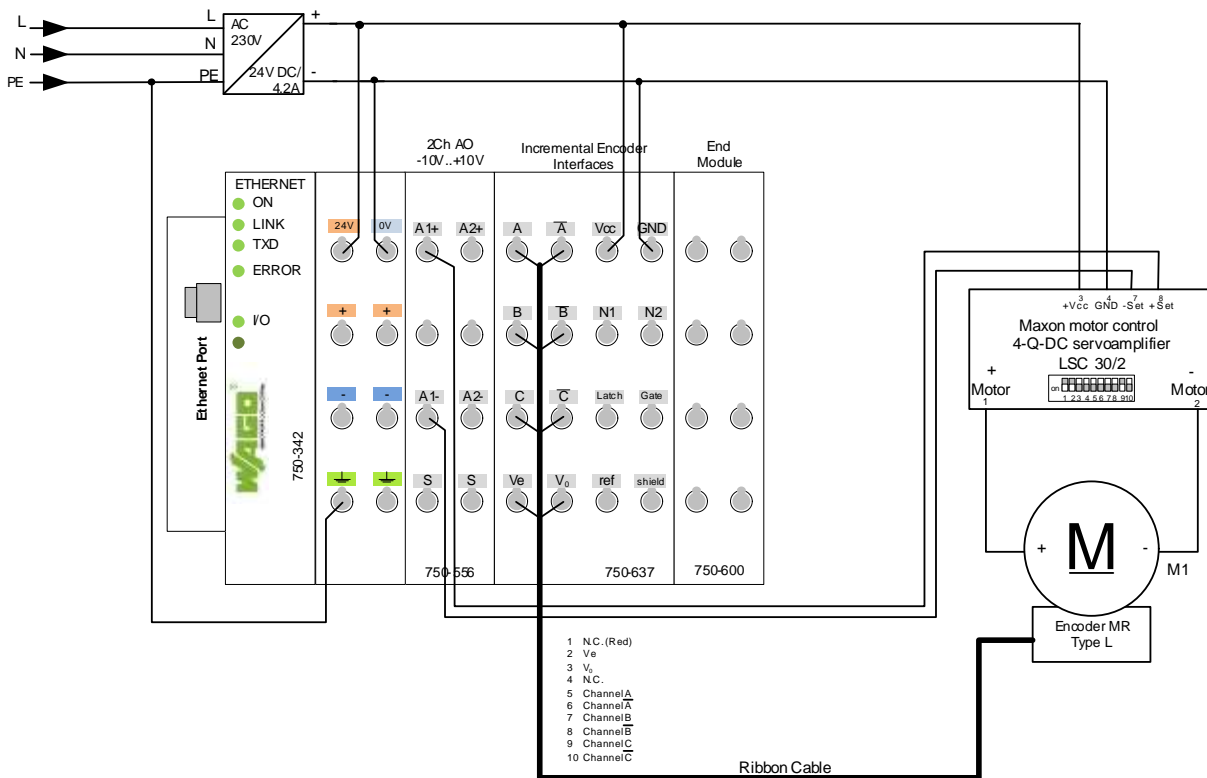


Figure 9 CATS-1M-Basic Wiring Diagram

A supply module is included in the Fieldbus coupler WAGO 750-342. The supply module contains the internal system supply as well as power jumper contacts for the filed supply via I/O module assemblies.

Analog output module WAGO 750-556 generating voltage signals ranging from -10V to +10V should connect to Maxon motor servoamplifier “-Set” and “+Set”.

Incremental encoder WAGO 750-637 connects to motor encoder via a ribbon cable. 24 V DC is required to supply this I/O module.



Attention:

Maxon servoamplifier works under voltage mode. DIP switches 1, 2 and 9 must be set to ON (01).



Figure 10 CATS-1M-basic Maxon servo amplifier DIP switches

3.2 CATS-1M-Advanced

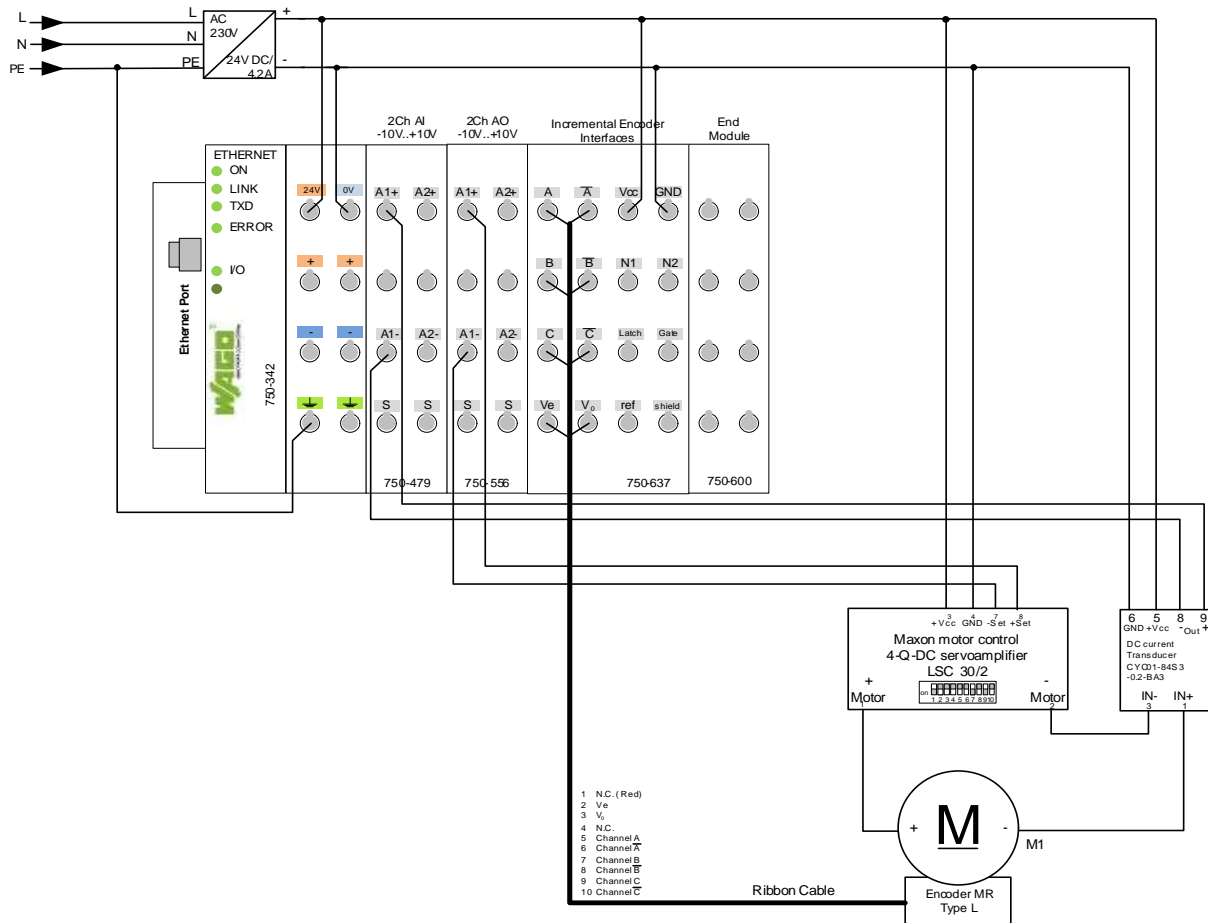


Figure 11 CATS-1M-Advanced Wiring Diagram

A supply module is included in the Fieldbus coupler WAGO 750-342. The supply module contains the internal system supply as well as power jumper contacts for the filed supply via I/O module assemblies.

Analog output module WAGO 750-556 generating voltage signals ranging from -10V to +10V should connect to Maxon motor servoamplifier “-Set” and “+Set”. DC current transducer converts the motor current value into voltage signal, which is measured by WAGO 750-479 2 Channel Analog Input.

Incremental encoder WAGO 750-637 connects to motor encoder via a ribbon cable. 24 V DC is required to supply this I/O module.



Attention:

Maxon servoamplifier works under current mode. DIP switches 2, 3, 4, 5, 6, 8 and 10 must be set to ON (01).

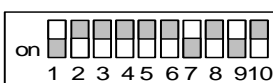


Figure 12 CATS-1M-Advanced Maxon servo amplifier DIP switches

3.3 CATS-2M-Basic

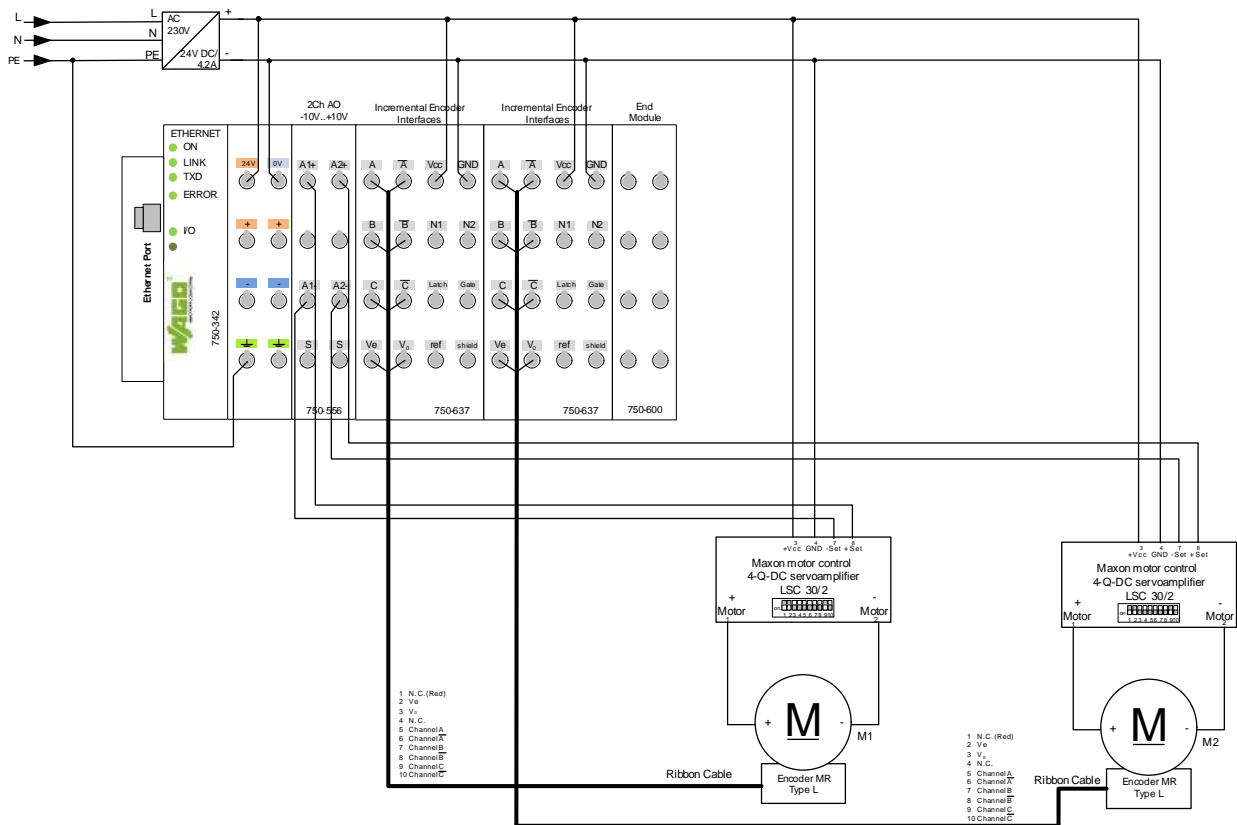


Figure 13 CATS-2M-Basic Wiring Diagram

A supply module is included in the Fieldbus coupler WAGO 750-342. The supply module contains the internal system supply as well as power jumper contacts for the field supply via I/O module assemblies.

Analog output module WAGO 750-556 generating voltage signals ranging from -10V to +10V should connect to Maxon motor servoamplifier “-Set” and “+Set”.

Incremental encoder WAGO 750-637 connects to motor encoder via a ribbon cable. 24 V DC is required to supply this I/O module.



Attention:

Maxon servoamplifier works under voltage mode. DIP switches 1, 2 and 9 must be set to ON (01).

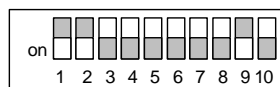


Figure 14 CATS-2M-basic Maxon servo amplifier DIP switches

3.4 CATS-2M-Advanced

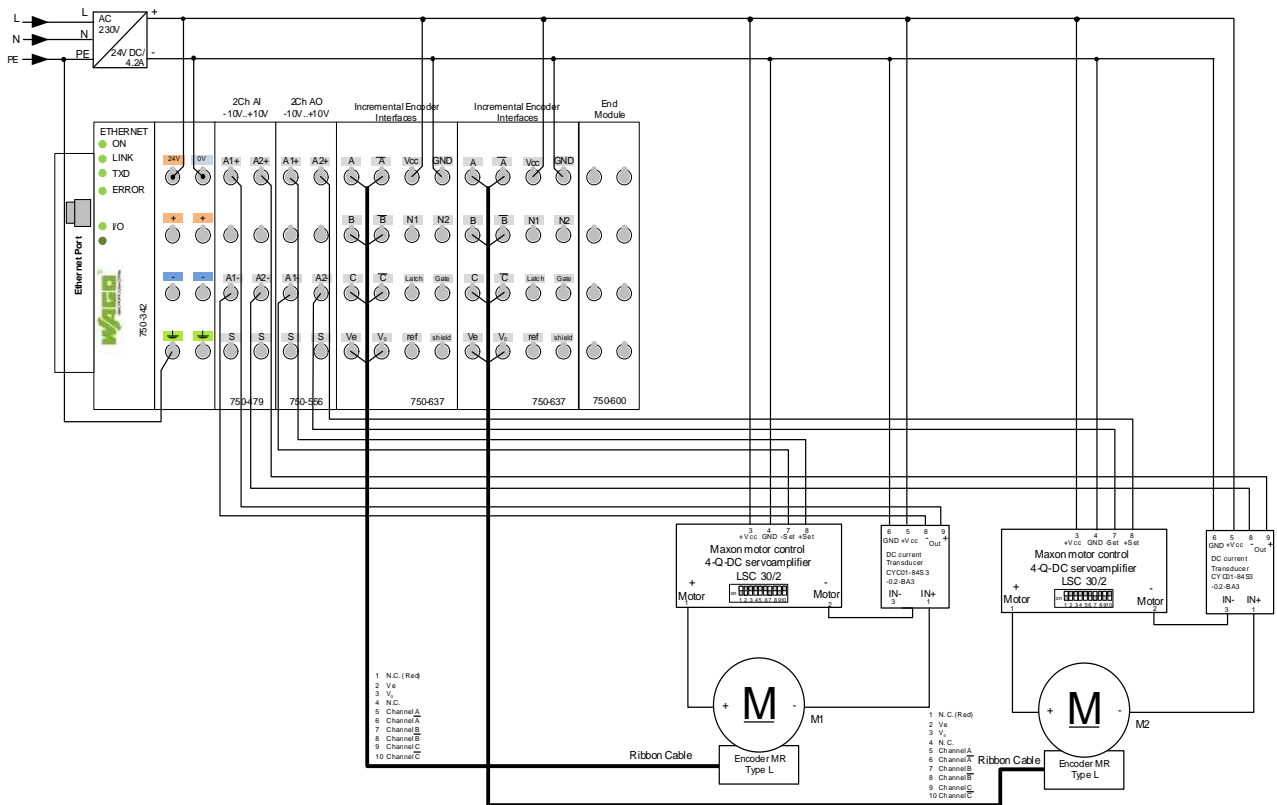


Figure 15 CATS-2M-Advanced Wiring Diagram

A supply module is included in the Fieldbus coupler WAGO 750-342. The supply module contains the internal system supply as well as power jumper contacts for the supply via I/O module assemblies.

Analog output module WAGO 750-556 generating voltage signals ranging from -10V to +10V should connect to Maxon motor servoamplifier “-Set” and “+Set”. DC current transducer converts the motor current value into voltage signal, which is measured by WAGO 750-479 2 Channel Analog Input.

Incremental encoder WAGO 750-637 connects to motor encoder via a ribbon cable. 24 V DC is required to supply this I/O module.



Attention:

Maxon servoamplifier works under current mode. DIP switches 2, 3, 4, 5, 6, 8 and 10 must be set to ON (01).

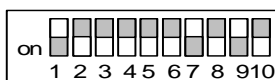


Figure 16 CATS-2M-Advanced Maxon servo amplifier DIP switches

4. Assign a Static IP Address to the Fieldbus Coupler

The following describes how to assign the IP address for the fieldbus node using the WAGO BootP server by way of an example.



Attention:

The IP address can be assigned in a direct connection via a crossover cable or via a parallel cable and a hub. An allocation over a switch is not possible.

Prerequisite for the following steps is the correct installation of the WAGO BootP server.

4.1 BootP Server to assign a Static IP Address

Step 1.

Launch the WAGO BootP Server. The screen below will appear.

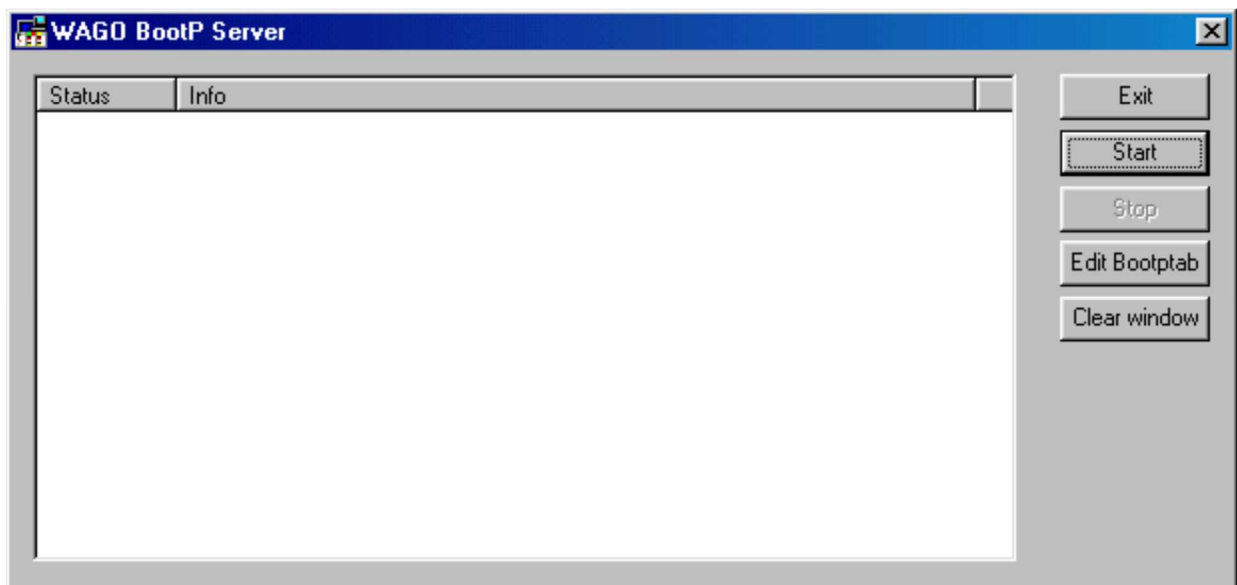
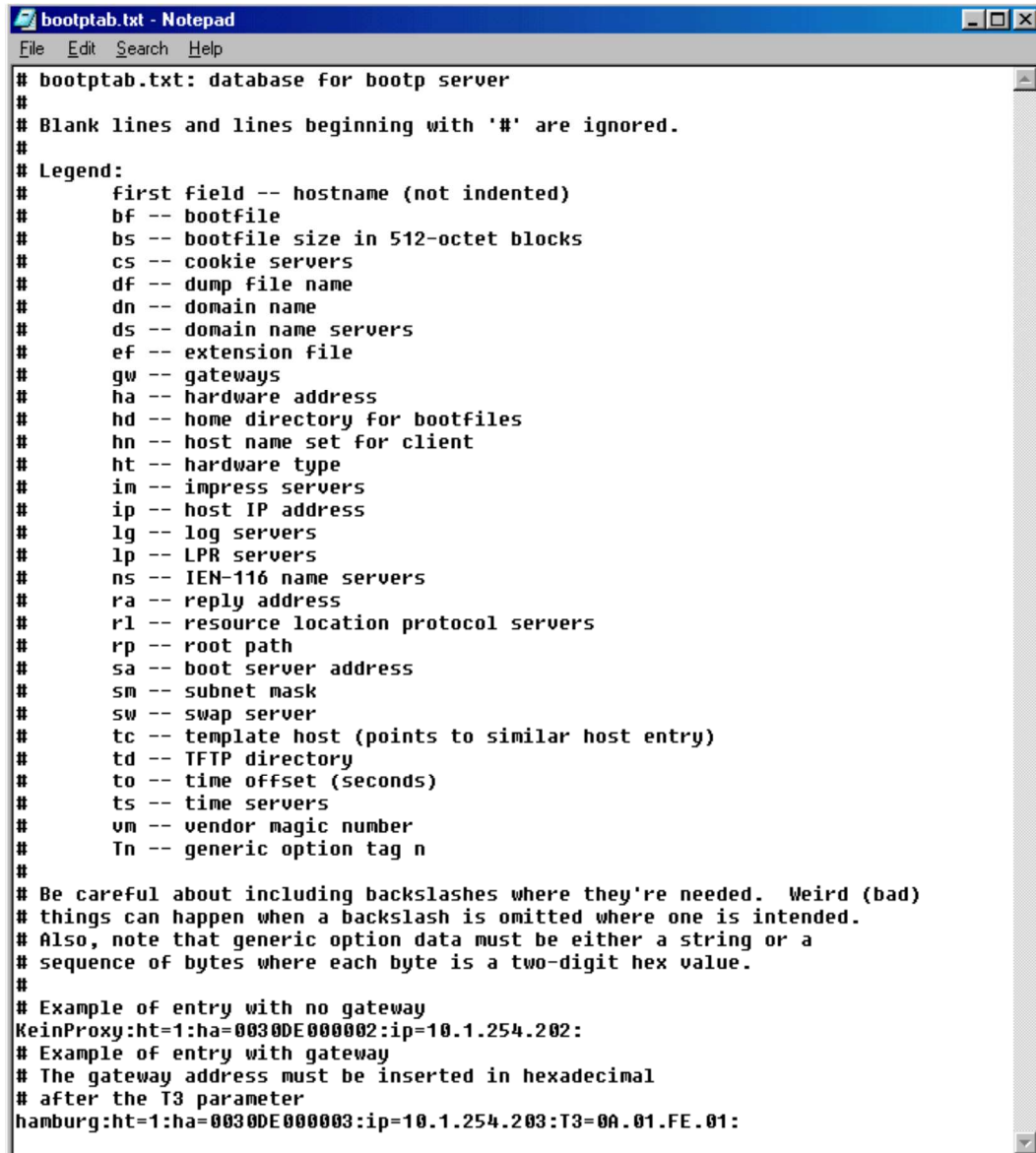


Figure 17 WAGO BootP Server

Step 2.

Click on “**Edit Bootptab**” command button. The Notepad editor will appear with a text file similar to the following:



```

bootptab.txt - Notepad
File Edit Search Help
# bootptab.txt: database for bootp server
#
# Blank lines and lines beginning with '#' are ignored.
#
# Legend:
#   first field -- hostname (not indented)
#   bf -- bootfile
#   bs -- bootfile size in 512-octet blocks
#   cs -- cookie servers
#   df -- dump file name
#   dn -- domain name
#   ds -- domain name servers
#   ef -- extension file
#   gw -- gateways
#   ha -- hardware address
#   hd -- home directory for bootfiles
#   hn -- host name set for client
#   ht -- hardware type
#   im -- impress servers
#   ip -- host IP address
#   lg -- log servers
#   lp -- LPR servers
#   ns -- IEN-116 name servers
#   ra -- reply address
#   rl -- resource location protocol servers
#   rp -- root path
#   sa -- boot server address
#   sm -- subnet mask
#   sw -- swap server
#   tc -- template host (points to similar host entry)
#   td -- TFTP directory
#   to -- time offset (seconds)
#   ts -- time servers
#   vm -- vendor magic number
#   Tn -- generic option tag n
#
# Be careful about including backslashes where they're needed. Weird (bad)
# things can happen when a backslash is omitted where one is intended.
# Also, note that generic option data must be either a string or a
# sequence of bytes where each byte is a two-digit hex value.
#
# Example of entry with no gateway
KeinProxy:ht=1:ha=0030DE000002:ip=10.1.254.202:
# Example of entry with gateway
# The gateway address must be inserted in hexadecimal
# after the T3 parameter
hamburg:ht=1:ha=0030DE000003:ip=10.1.254.203:T3=0A.01.FE.01:

```

Figure 18 bootptab.txt - Notepad

Step 3.

Any line that begins with a “#” symbol is a comment, and will not be processed. Look at the bottom line, beginning with the word “**hamburg**”. If a gateway address is not going to be used (typical), put “#” symbol in front of this line.

Step 4.

Next I check the line beginning with the word “**KeinProxy**”. “**KeinProxy**” is a German term meaning “**Node Name**”. This word can be changed to any descriptor you desire to identify your WAGO Ethernet IO device (e.g. WAGONode1).

The remaining contents of this line need to be modified to set the IP address of the WAGO Fieldbus coupler.

Look on the right side of the WAGO Fieldbus coupler for the unique Mac ID of the device.

Example: MAC ID 0030DE00028A

In the “KeinProxy” line, you will find the characters **ha=**. “**ha**” is short for hardware address, also known as the MAC ID. After “**ha=**”, enter the MAC ID of your device.

Example: ha=0030DE00028A

Also in the KeinProxy Line, you will find the characters “**ip=**”, short for IP address. Fill in the desired IP address for the WAGO Fieldbus coupler. This will be the IP address of the device on the network.

Example: ip=10.1.254.202

If a Gateway is to be used then fill in:

ha= (MAC ID) : ip= (IP Address): T3= (Gateway Address) : sm= (subnet mask)

Step 5

When all data has been entered it must be saved. At the top of the Notepad editor,

Click on **File**

Click on **Save**, to save the text file. (Do not change the file name.)

Step 6

Close the Notepad editor. The WAGO BootP Server is again displayed.

Step 7

Click the **Start** command button. Three or five status messages will appear, similar to the following screen capture:

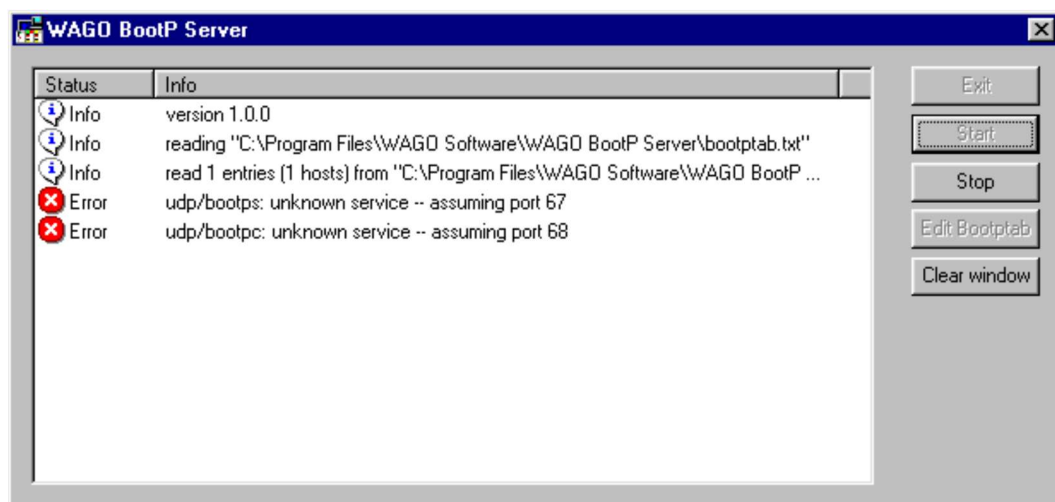


Figure 19 Assigning IP address

Step 8

Turn power off to the WAGO Fieldbus coupler, and wait for at least 20 seconds.

Step 9

It is important to restart the coupler by resetting the hardware. This ensures that the new IP address will be accepted by the coupler. To do this, connect the power to the fieldbus coupler for approx. 15 seconds. Following this, the IP address in the coupler is permanently stored and maintained even once the coupler is removed or following a longer voltage failure. You will see more status messages scroll down, similar to the following screen capture:

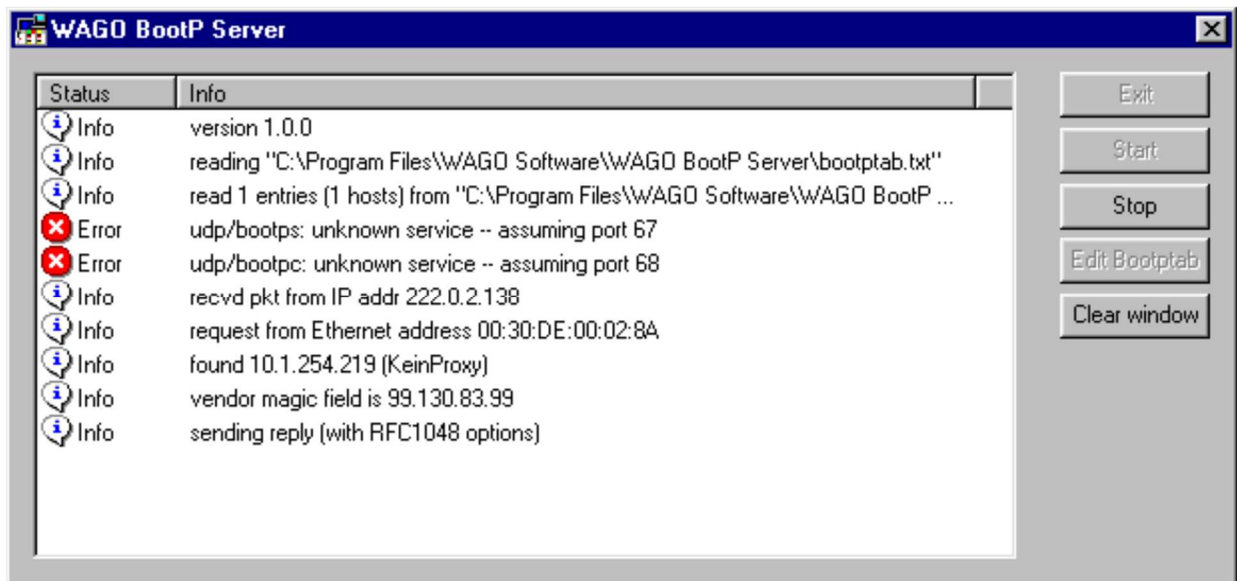


Figure 20 IP Address confirmation

Step 10

Subsequently, click on the **"Stop"** button and then on the **"Exit"** button, to close the BootP Server .

Step 11

Verify the WAGO Fieldbus coupler is operating correctly. Its diagnostic LEDs should be illuminated as follows:

Link	Green
MS (Module Status) Green
NS	(Node Status) Green or Flashing Green
TxD/RxD	Flash as data is Sent/Received
I/O	Green

4.2 Testing the Function of the Fieldbus Connectivity

Step 1

To test the communication with the coupler and the correct assignment of the IP address call up the DOS prompt under Start menu / Program /MSDOS Prompt.or pressing Windows+R

Step 2

Enter the command: "**ping**" with the IP address you have assigned in the

Example: ping 10.1.254.202

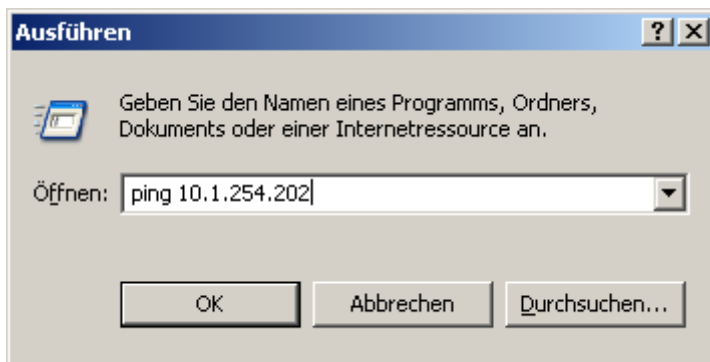


Figure 21 Example for the function test of a fieldbus node

Step 3

When the Return key has been pressed, your PC will receive a response from the coupler, which will then be displayed in the DOS prompt. If the error message: "**Timeout**" appears instead, please compare your entries again to the allocated IP address and make sure the IP address of Fieldbus coupler is in the same subnet as the operating computer.

Step 4

When the test has been performed successfully, you can close the DOS prompt. The network node has now been prepared for communication.

5. LabMap[®] Introduction

5.1 Overview

LabMap[®] is a 32-bit MS-Windows[®] multithreading application providing a high level interface for industrial communication, control, measurement, and logging. It allows asynchronous viewing and modification of the system variables from a potentially unlimited number of applications running on the same or different LAN/WAN network hosts. LabMap[®] has an open architecture supporting smooth integration of new hardware and software components running on different bus architectures. It has integrated support of measurement and physical units and time stamping of the system variables. The software works on OS Microsoft Windows[®] (x86 platforms).

5.2 Concepts

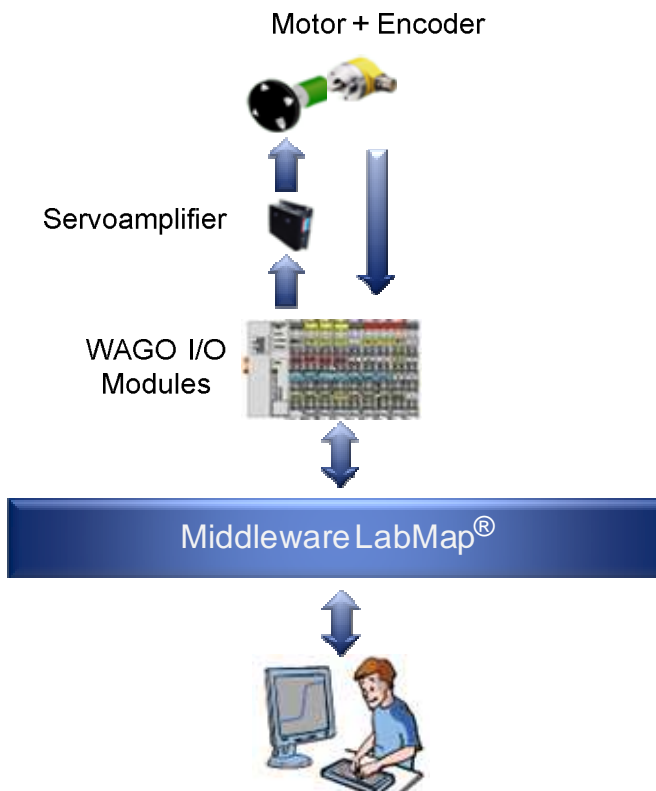


Figure 22 Middleware LabMap[®] in CATS

As Figure 22 shows, LabMap[®] operates as a software bus system. In this Individual laboratory experiment device, LabMap[®] offers an abstraction of the application level from the hardware specific and decoupling the hardware interface modules from the application level. In other words, there are two levels of abstraction supported by the bus system:

- The application interface.
- The hardware driver interface

The application abstraction interface exposes LabMap[®] as a set of variables (handles). Each handles has a type, value, timestamp and I/O direction. The software structure allows user-written applications to access the system variables and parameters over the standard interface independently from the actual hardware configuration.

5.3 From the Individual Laboratory Experiment to Network Experiment

With the help of the LabMap[®] **software bus**, each individual laboratory experiment on the network (Ethernet/ModBus) can be made accessible to several students. In principle, this works like a network printer. The concept “one for all” is also well-suited for modernizing existing laboratory experiments cost-effectively and making them network-capable.

As Figure 23 shows, various individual laboratory experiments can also be operated from one or more workstations.

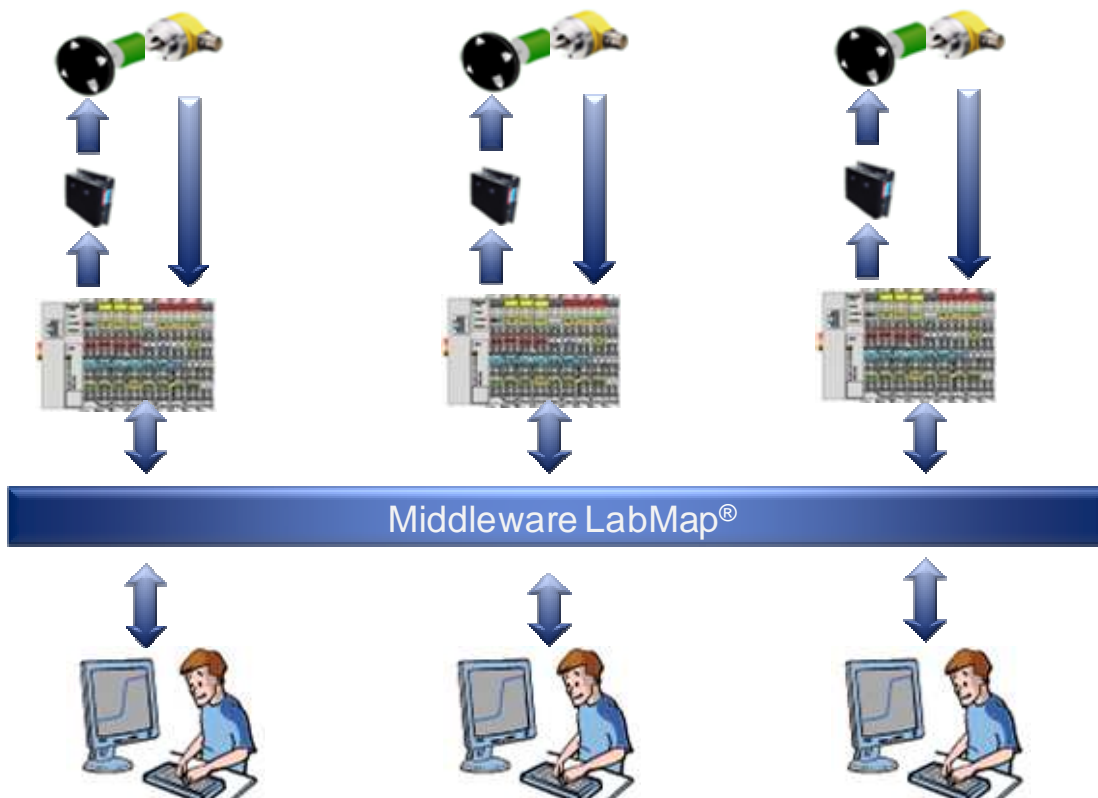


Figure 23 All for all

For more information LabMap[®] software bus, please visit:

Softwarebus LabMap[®] User guide:
http://www.cbb-software.com/docs/labmab_userguide.pdf

Softwarebus LabMap[®] Handbook:
http://www.cbb-software.com/docs/labmap_handbook.pdf

6. How to Create LabMap® Handles

6.1 LabMap® Input and Output Handles

The middleware LabMap® exposes itself as a set of variables (handles). Each handles has a type, value, timestamp and two I/O directions (Input/output).

There are four types of registers supported:

- Integer (32 bit signed integer)
- Real (IEEE 32 bit floating point)
- String (up to 64K bytes long)
- Record (used for optimizing data exchange)

The following Figure 24 shows the relation between LabMap I/O and WAGO I/O:

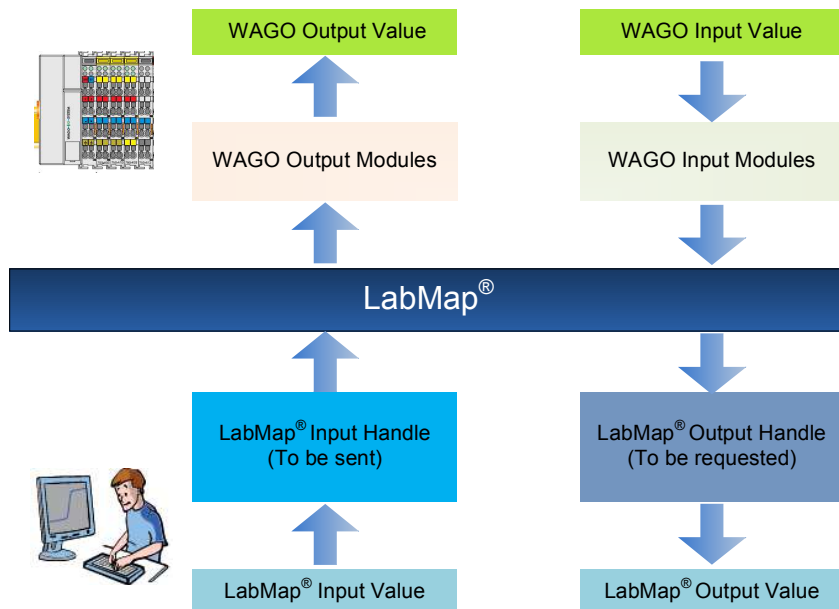


Figure 24 LabMap® Input and Output Directions

As you can see in the Figure 24, a LabMap® Input handle accepts a value from any application. This value, which is setting to a LabMap® Input handle is going to be sent out by a WAGO Output Module. Generally spoken, a LabMap® input handle is corresponding with Output hardware.

For the same reason, a LabMap® output handle gets its output value from Input hardware. This value from input hardware is going to be requested by other software applications e.g. Matlab Simulink, C#, LabVIEW etc.

6.2 Creating New Handles

Prerequisite for the following steps is the correct installation of the LabMap®.

Step 1

Launch the LabMap.exe. The screen below will appear.

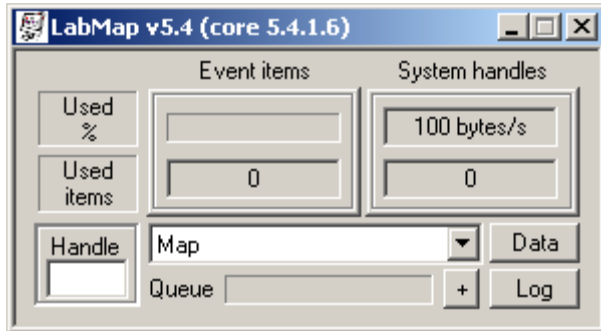


Figure 25 LabMap® control panel

Step 2

Type a handle number (e.g.1001) in the Handle Field as shown in Figure 26 .After pressing “**Enter**”, a “**New Register**” dialog will pop up automatically as shown in Figure 27:

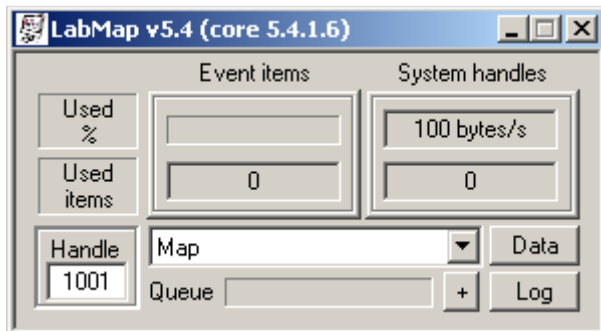


Figure 26 Type a new handle number

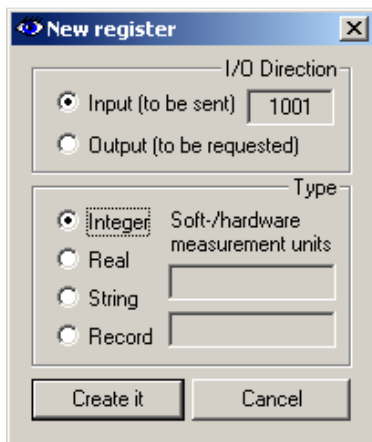


Figure 27 New register dialog

Step 3

As we see in Figure 28, we have to choose “**Input**” or “**Output**” to determine the I/O Direction of this new handle. (Also see Figure 24 in Chapter 6.1)

According to different types of hardware, we should choose the proper “**Type**” . As Figure 28 , click button “**Create it**”.

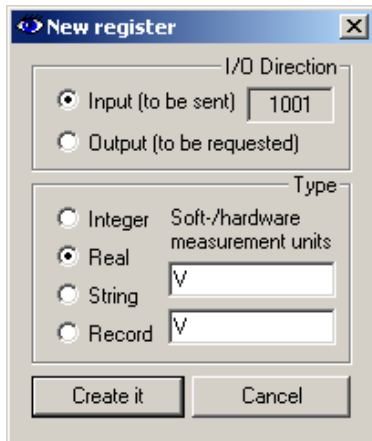


Figure 28 Parameterize new handle

Step 4.

A configuration dialog will pop up after the handle has been created as shown in Figure 29. Give the Handle a name, e.g.“Motor_Voltage” and set the “Timeout” properties and then type in the “**Handled by**” field the name of the used interface, here “**ModBus**” instead of “**Map**” or choose the pull down menu.

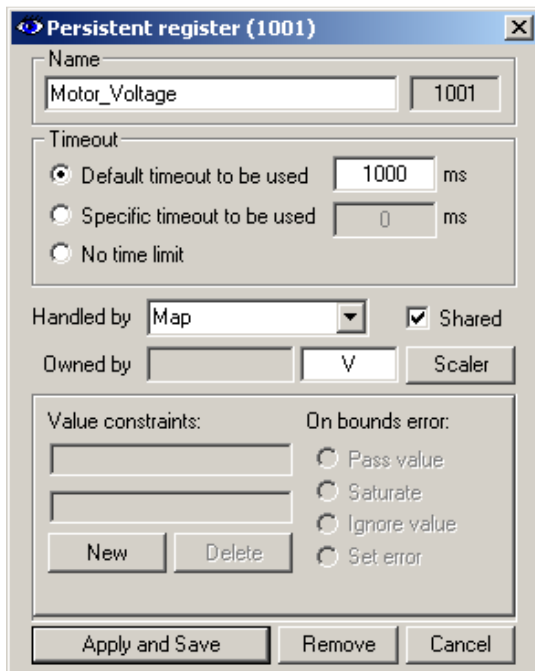
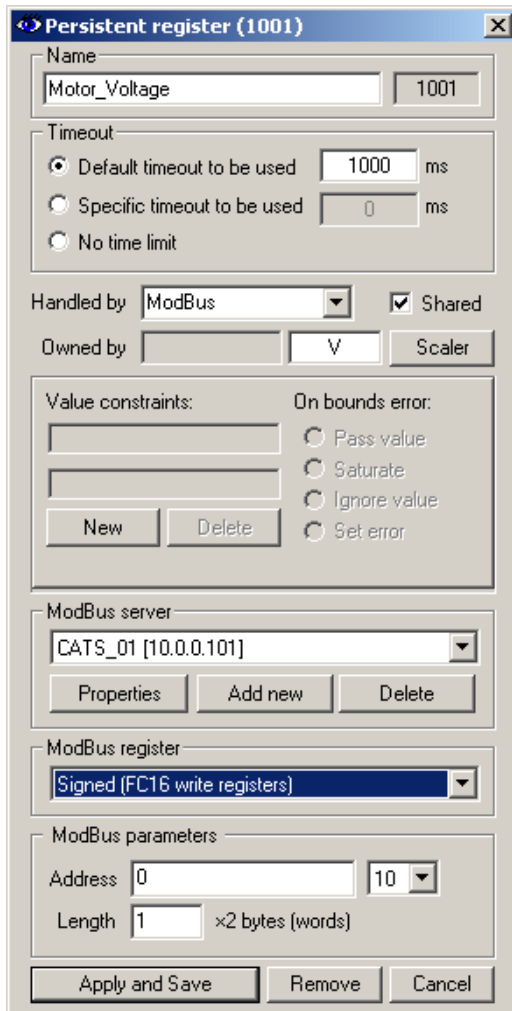


Figure 29 Handle configuration

Step 5

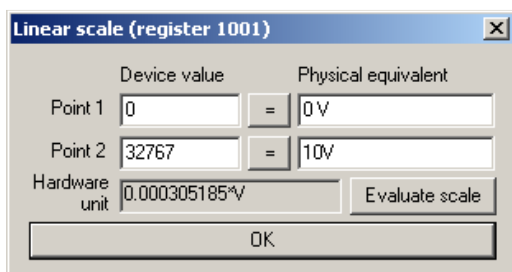
Configure the new handle as shown in Figure 30. If the Combobox in the field “**ModBus server**” is empty, click “**Add new**” and fill in the name and IP Address of the ModBus server. And then click “**OK**”. Click button “**Scale**”, a linear scale dialog for handle 1001 appears. Set it up, likely as shown in Figure 31. For more information about “**linear scale**”, please see “Table 3 Process values of Module 750-556” in chapter 2.1.



The dialog box is titled "Persistent register (1001)". It contains the following fields and options:

- Name:** Motor_Voltage (with a small box containing 1001)
- Timeout:**
 - Default timeout to be used: 1000 ms
 - Specific timeout to be used: 0 ms
 - No time limit
- Handled by:** ModBus (dropdown menu)
- Shared
- Owned by:** V (dropdown menu) and Scaler (button)
- Value constraints:** (empty text boxes)
- On bounds error:**
 - Pass value
 - Saturate
 - Ignore value
 - Set error
- ModBus server:** CATS_01 [10.0.0.101] (dropdown menu)
- Buttons: Properties, Add new, Delete
- ModBus register:** Signed [FC16 write registers] (dropdown menu)
- ModBus parameters:**
 - Address: 0 (text box) and 10 (dropdown menu)
 - Length: 1 (text box) ×2 bytes (words)
- Buttons: Apply and Save, Remove, Cancel

Figure 30 Configuration for Handle 1001



The dialog box is titled "Linear scale (register 1001)". It contains the following fields and options:

- Point 1:** Device value: 0, Physical equivalent: 0V
- Point 2:** Device value: 32767, Physical equivalent: 10V
- Hardware unit:** 0.000305185*V
- Buttons: Evaluate scale, OK

Figure 31 Linear scale

Step 6

Click “**Apply and Save**” in the configuration dialog (See Figure 30), the creation of a new handle is completed. A handle panel pops up as shown in Figure 32.

Click button “**Properties**”, it turns back to the handle configuration dialog.

Click button “**Send the new value**”, a new value is sent to the corresponding hardware.

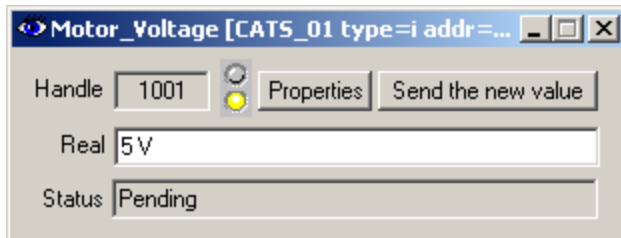


Figure 32 Handle Panel

To get deeper knowledge about LabMap[®] and LabMap[®] handle creation, please visit:

LabMap Handbook:

http://www.cbb-software.com/docs/labmap_handbook.pdf

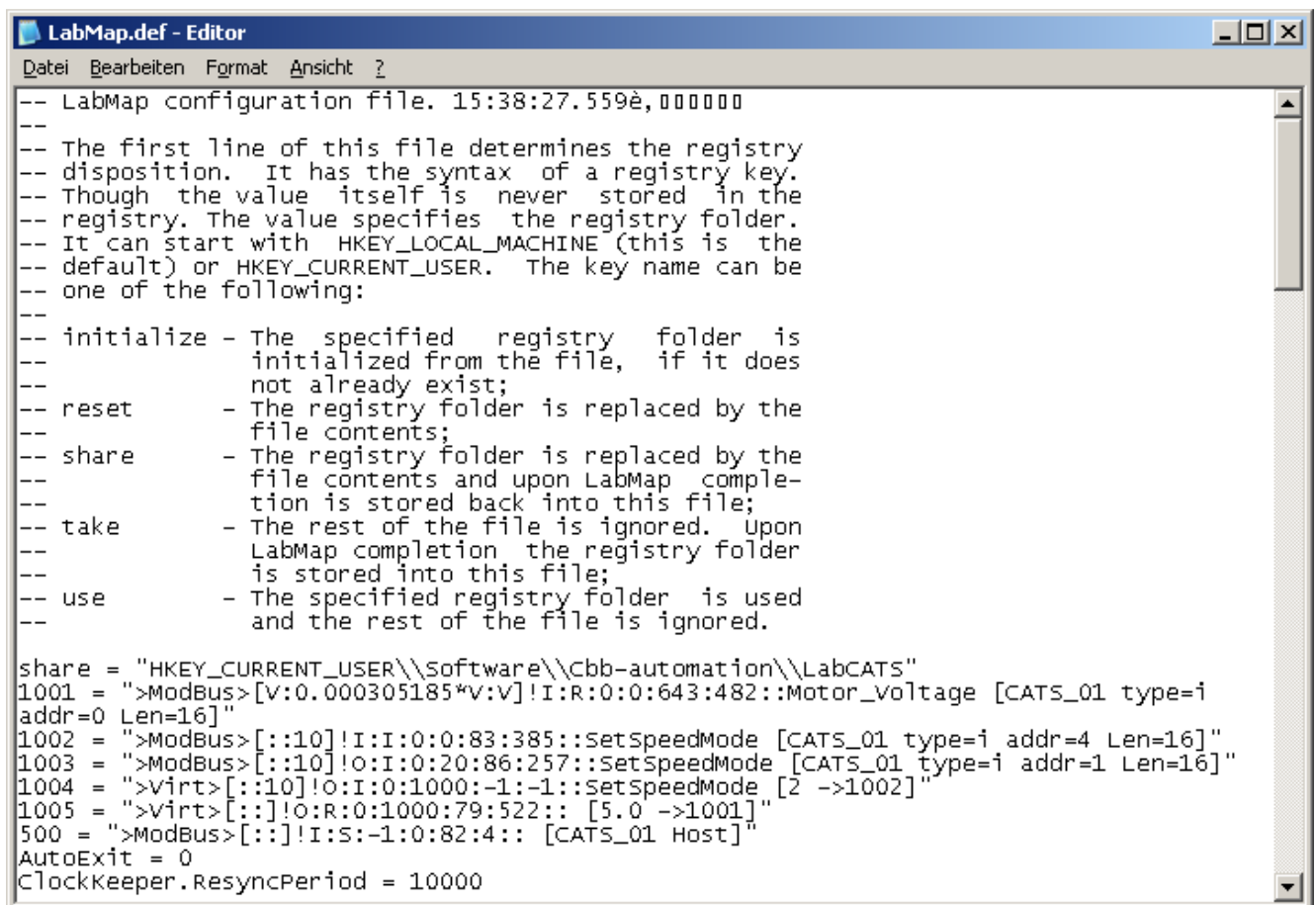
LabMap User Guide:

http://www.cbb-software.com/docs/labmap_userguide.pdf

7. LabMap® Configuration

The configuration data for LabMap® are stored in the windows registry. The location and contents of the registry folder is determined upon start as described follows:

1. If the software is started explicitly through invoking the graphical user interface application “LabMap.exe”, then the first parameter of the command line is the configuration file name. Without the parameter it proceeds to the second step;
2. When the first step does not define the configuration file, or when the software is started indirectly the directory containing the “LabMap.dll” is scanned for the configuration file LabMap.def;
3. When steps 1 and 2 fails the registry folder is used “HKEY_LOCAL_MACHINE / Software / Cbb-automation / LabMap”.



```

LabMap.def - Editor
Datei Bearbeiten Format Ansicht ?
--- LabMap configuration file. 15:38:27.559è,000000
---
--- The first line of this file determines the registry
--- disposition. It has the syntax of a registry key.
--- Though the value itself is never stored in the
--- registry. The value specifies the registry folder.
--- It can start with HKEY_LOCAL_MACHINE (this is the
--- default) or HKEY_CURRENT_USER. The key name can be
--- one of the following:
---
--- initialize - The specified registry folder is
---               initialized from the file, if it does
---               not already exist;
--- reset      - The registry folder is replaced by the
---               file contents;
--- share     - The registry folder is replaced by the
---               file contents and upon LabMap comple-
---               tion is stored back into this file;
--- take      - The rest of the file is ignored. Upon
---               LabMap completion the registry folder
---               is stored into this file;
--- use       - The specified registry folder is used
---               and the rest of the file is ignored.
---
share = "HKEY_CURRENT_USER\\software\\Cbb-automation\\LabCATS"
1001 = ">ModBus>[V:0.000305185*v:v]!I:R:0:0:643:482::Motor_voltage [CATS_01 type=i
addr=0 Len=16]"
1002 = ">ModBus>[::10]!I:I:0:0:83:385::SetSpeedMode [CATS_01 type=i addr=4 Len=16]"
1003 = ">ModBus>[::10]!O:I:0:20:86:257::SetSpeedMode [CATS_01 type=i addr=1 Len=16]"
1004 = ">Virt>[::10]!O:I:0:1000:-1:-1::SetSpeedMode [2 ->1002]"
1005 = ">Virt>[::]!O:R:0:1000:79:522:: [5.0 ->1001]"
500 = ">ModBus>[::]!I:S:-1:0:82:4:: [CATS_01 Host]"
AutoExit = 0
ClockKeeper.ResyncPeriod = 10000

```

Figure 33 LabMap.def file

The file may contain comments start with “--“, “#” or “;”. A comment continues to the end of the source line.

The first line of this file determines the registry disposition. It has the syntax of a registry key. Though the value itself is never stored in the registry, the value specifies the registry folder. It can start under the key “HKEY_LOCAL_MACHINE” (this is the default) or “HKEY_CURRENT_USER”. The rest of the value is folder description. The key name can be one as following (case insensitive):

Key	Disposition
initialize	The specified by the value registry folder is initialized from the file, if it does not already exist. In the latter case the file contents is ignored.
reset	The registry folder is replaced by the file contents.
share	The registry folder is replaced by the file contents and upon LabMap [®] completion is stored back into this file. Therefore, the contents of the file reflect the last configuration of LabMap [®] .
take	The rest of the file is ignored. Upon LabMap [®] completion the registry folder is stored into this file.
use	The specified registry folder is used and the rest of the file is ignored.

Table 7 LabMap configuration data store Keys



Attention:

Editing “LabMap.def” file manually requires exact knowledge on functionalities of each defined key. Improper change in “LabMap.def” file could lead the software to abnormal condition.

For more information about LabMap[®] configuration Keys, please see:

LabMap[®] configuration:

<http://www.cbb-software.com/docs/labmap/labmap.html#configuration>

8. Default LabMap[®] Handles Configuration

8.1 CATS-1M-Basic

Handle No.	Name	I/O direction	Type	Unit
1000	ModBus Host	LabMap [®] Input	String	-----
1001	Motor-1 Voltage	LabMap [®] Input	Real	V
1002	Motor-1 Velocity	LabMap [®] Output	Real	1/min
1003	Encoder Mode	LabMap [®] Input	Integer	-----

Table 8 CATS-1M-Basic default handles list

Handle 1000: This is a ModBus Host Handle which is used to set the IP address of the corresponding ModBus server.

Handle 1001: This handle is used to set the output voltage for the DC motor.

Handle 1002: The rotation velocity (rpm) of the motor can be notified by this handle.

Handle 1003: Decimal value “2” has to be set to “Control Byte C1” of the encoder interface WAGO 750-637, to set the encoder interface to velocity mode.



Attention:

Encoder interface WAGO 750-637 is able to work under “Counter Mode”(0), “Latch Mode”(1), “Velocity Mode”(2) and “Set Value Mode”(3). The work mode of WAGO 750-637 depends on the “**Control Byte C1**”.

More information about WAGO 750-637, please see chapter: 2.4 Incremental Encoder Interfaces WAGO 750-637.

8.2 CATS-1M-Advanced

Handle No.	Name	I/O direction	Type	Unit
1000	ModBus Host	LabMap [®] Input	String	-----
1001	Motor-1 Voltage	LabMap [®] Input	Real	V
1002	Motor-1 Velocity	LabMap [®] Output	Real	1/min
1003	Motor-1 Current	LabMap [®] Output	Real	A
1004	Encoder Mode	LabMap [®] Input	Integer	-----

Table 9 CATS-1M-Advanced default handles list

Handle 1000: This is a ModBus Host Handle which is used to set the IP address of the corresponding ModBus server.

Handle 1001: This handle is used to set the voltage for the motor.

Handle 1002: The rotation velocity of the motor can be required by this handle.

Handle 1003: The current value of the motor can be required by this handle.

Handle 1004: Decimal value "2" has to be set to "Control Byte C1" of the encoder interface WAGO 750-637, so that encoder interface works under velocity mode.



Hint:

Encoder interface WAGO 750-637 is able to work under "Counter Mode"(0), "Latch Mode"(1), "Velocity Mode"(2) and "Set Value Mode"(3). The work mode of WAGO 750-637 depends on the "**Control Byte C1**".

More information about WAGO 750-637, please see chapter: 2.4 Incremental Encoder Interfaces WAGO 750-637.

8.3 CATS-2M-Basic

Handle No.	Name	I/O direction	Type	Unit
1000	ModBus Host	LabMap [®] Input	String	-----
1001	Motor-1 Voltage	LabMap [®] Input	Real	V
1002	Motor-1 Velocity	LabMap [®] Output	Real	1/min
1003	Motor-2 Voltage	LabMap [®] Input	Real	V
1004	Motor-2 Velocity	LabMap [®] Output	Real	1/min
1005	Encoder Mode	LabMap [®] Input	Integer	-----

Table 10 CATS-2M-Basic default handles list

Handle 1000: This is a ModBus Host Handle which is used to set the IP address of the corresponding ModBus server.

Handle 1001: This handle is used to set the output voltage for the Motor-1.

Handle 1002: The rotation velocity (rpm) of tMotor-1 can be notified by this handle.

Handle 1003: This handle is used to set the output voltage for the Motor-2.

Handle 1004: The rotation velocity(rpm) of the Motor-2 can be notified by this handle.

Handle 1005: Decimal value "2" has to be set to "Control Byte C1" of the encoder interface WAGO 750-637, to set the encoder interface to velocity mode.



Hint:

Encoder interface WAGO 750-637 is able to works under "Counter Mode"(0), "Latch Mode"(1), "Velocity Mode"(2) and "Set Value Mode"(3). The work mode of WAGO 750-637 depends on the "**Control Byte C1**".

More information about WAGO 750-637, please see chapter: 2.4 Incremental Encoder Interfaces WAGO 750-637.

8.4 CATS-2M-Advanced

Handle No.	Name	I/O direction	Type	Unit
1000	ModBus Host	LabMap [®] Input	String	-----
1001	Motor-1 Voltage	LabMap [®] Input	Real	V
1002	Motor-1 Velocity	LabMap [®] Output	Real	1/min
1003	Motor-1 Current	LabMap [®] Output	Rea;	A
1004	Motor-2 Voltage	LabMap [®] Input	Real	V
1005	Motor-2 Velocity	LabMap [®] Output	Real	1/min
1006	Motor-1 Current	LabMap [®] Output	Real	A
1007	Encoder Mode	LabMap [®] Input	Integer	-----

Table 11 CATS-2M-Advanced default handles list

Handle 1000: This is a ModBus Host Handle which is used to set the IP address of the corresponding ModBus server.

Handle 1001: This handle is used to set the voltage for the Motor-1.

Handle 1002: The rotation Velocity (rpm) of the Motor-1 can be notified by this handle.

Handle 1003: The current value of the Motor-1 can be notified by this handle.

Handle 1004: This handle is used to set the voltage for the Motor-2.

Handle 1005: The rotation velocity(rpm) of the Motor-2 can be notified by this handle.

Handle 1006: The current value of the Motor-2 can be notified by this handle.

Handle 1007: Decimal value “2” has to be set to “**Control Byte C1**” of the encoder interface WAGO 750-637, to set the encoder interface to velocity mode.



Hint:

Encoder interface WAGO 750-637 is able to works under “Counter Mode”(0), “Latch Mode”(1), “Velocity Mode”(2) and “Set Value Mode”(3). The work mode of WAGO 750-637 depends on the “Control Byte C1”.

More information about WAGO 750-637, please see chapter: 2.4 Incremental Encoder Interfaces WAGO 750-637.