Industrial Ethernet

User Manual

MAC00-EC4/-EC41, MAC00-EI4/-EI41, MAC00-EL4/-EL41, MAC00-EP4/-EP41, MAC00-EM4/-EM41 , MAC00-ES4/-ES41
&
MIS and MILxxx (G2) motors

(Setup and functionality only. Some connector ID's may differ)

JVL A/S
Bregnerødvej 127 - DK-3460 Birkerød
Denmark - Tel: +45 4582 4440
Email: jvl@jvl.dk - www.jvl.dk

JVL
intelligent motors

LB0056-25GB

Revised 16. September 2021

Important
User Information

Warning

The MAC and MIS series of products are used to control electrical and mechanical components of motion control systems.
You should test your motion system for safety under all potential conditions. Failure to do so can result in damage to equipment and/or serious injury to personnel.

Please contact your nearest JVL representative in case of technical assistance. Your nearest contact can be found on our web site www.jvl.dk

JVL A/S
Bregnerødvej 127
DK-3460 Birkerød
Denmark
Tlf. +45 45 82 44 40
Fax. +45 45 82 55 50
e-mail: jvl@jvl.dk
Internet: http://www.jvl.dk

CANopen®	Is a registered trademark of CAN in AUTOMATION - International Users and Manufacturers Group e. V. (CiA), Nürnberg.
DeviceNet®	Is a trademark of ODVA (Open DeviceNet Vendor Association, Inc).
EtherCAT®	Is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
EtherNet/IP®	Is a trademark of ODVA (Open DeviceNet Vendor Association, Inc).
Modbus TCP/IP®	Is a registered trademark of Schneider Electric.
PROFINET IO®	Is a registered trademark of PROFIBUS International, Karlsruhe.
SERCOS®	Is a registered trademark of SERCOS International e.V., Suessen, Germany.

Contents

1 Introduction

1.1 Introduction

MAC family
Ethernet for servos

MIS family
Ethernet for steppers

Industrial Ethernet is becoming more and more popular as it offers

- Very fast response time
- Predictable delay times (deterministic protocol)
- Safe transmission of data

Compared with most of the “classic” non Ethernet based protocols the indstrial Ethernet offers state of the art performance.

The MAC00-Ex4/-Ex41 (Ethernet module for MAC motors) and MIS/MILxxxxxx**E**xxxxxx (MIS/MIL motors with Ethernet option) can be configured by the end user to a number of different Ethernet protocols, for instance

- EtherCAT®
- EtherNetIP®
- Ethernet POWERLINK®
- PROFINET IO®
- Modbus TCP/IP®
- Sercos®
- And more to come

Main Features:

- High speed communication - 100Mbits/sec.
- 2 individual ports on the module offers Daisy chaining possibility.
- Standard M12 circular industrial connectors
- MAC motor module MAC00-**Ex4**: **1** Digital input (24V) and **1** digital output (24V) for local use around the motor
- MAC motor module MAC00-**Ex41**: **4** Digital input (24V) and **2** digital outputs (24V) for local use around the motor
- MIS/MIL motor with Ethernet option offers 8 digital I/O’s. Each I/O terminal can also be an analog input
- Multiple alternative I/O possibilities available on request (OEM applications)
- LED’s for easy monitoring of operation status
- Optional encoder I/O
- Rough design
- Access to all internal motor parameters and registers possible. No need of pre-setup of the motor.
- RS232 connection available for monitoring and setup use for the MAC00-Ex4/-Ex41 modules.
- RS485 connection available for monitoring and setup use for the MIS/MILxxxxxx**E**xxxx motor.

1.2 Module types

1.2.1 Module types (Only applicable for MAC00-Ex4/-Ex41)

The MacMotor Ethernet modules are available for several Ethernet protocols.
The module used for each protocol has its own unique type number, but is based on the exactly same hardware.
A neutral module where no protocol is installed however also exist.

- **Neutral module - no protocol installed.**
  MAC00-Ex4/-Ex41 is a neutral module not setup-up for any particular protocol. The final user can setup it up for any of the available protocols just by using the general MacTalk windows software.
  The visible LED marking, labels etc. only states that its a neutral MAC00-Ex4/-Ex41 module.

- **Pre-loaded module - a specific protocol has been installed.**
  The modules MAC00-EC4/-EC41 (EtherCAT), MAC00-EI4/-EI41 (EtherNetIP), and MAC00-EL4/-EL41 (POWERLINK), MAC00-EP (Profinet), MAC00-EM (Modbus TCP) are setup at delivery with the relevant protocol and also the right LED marking.
  The final user can setup it up for any of the available protocols just by using the general MacTalk windows software.
  The visible LED marking, and type number is unique for each module type.

All modules (when not delivered mounted in a MacMotor) is followed by a little label sheet containing labels for all the available standards and standards to come.

The overall idea is that any module can be changed to another protocol if desired, the modules can stay neutral when it passes the distribution channel and be setup by the end-user simplifying the logistics.

MAC800 users important:
Please notice that only MAC800 motors with a serial number newer than 85000 is compatible with the Ethernet modules MAC00-Exx.

1.2.2 How to change the protocol type

Only 2 steps are needed in this process.

1. Install the intended protocol firmware in the module.
2. Apply or changing the label with LED marking and type number of the module.

The firmware can be setup as follows

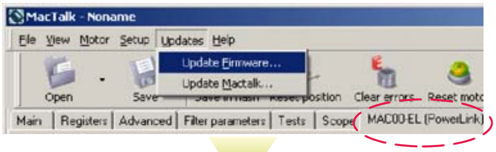
(see next page)

How to setup the module for a different/new protocol

Step 1
Determine which Ethernet protocol you want to use. Have in mind that your Ethernet module may already be setup for a protocol.

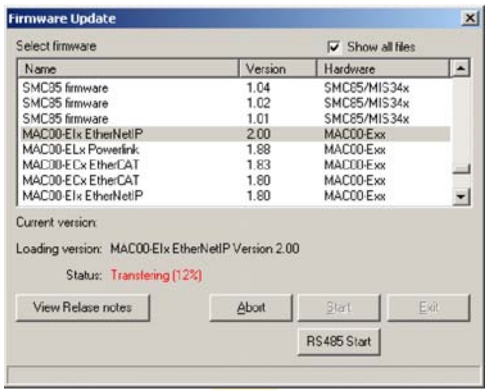
! When changing protocol the module factory defaults are restored.

Step 2
As shown the module is setup as a module with the Ethernet Powerlink protocol.
Choose the *Update Firmware* in the *Updates* menu to setup the module with another protocol.

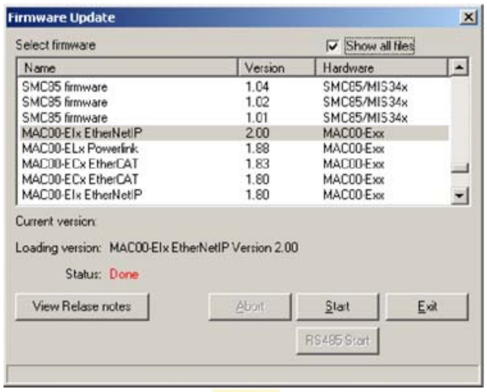


Step 3
Make sure that the checkbox *"Show all files"* is checked.
Select the desired firmware such as EtherNet-IP.
Note that there may exist more than one version. Choose the newest version.

Press *Start* to download the selected firmware.
The status counter will now rise from 0 to 100%.

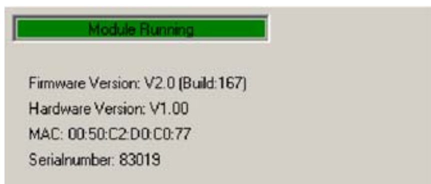


Step 4
When the download process is finished, the status shows *"Done"*.
Also *"Current version"* has changed to the actual downloaded version meaning that the firmware in the module is now changed permanently.



Step 5
The module tab has now changed from Powerlink (EL) to EthernetIP (EI).

Step 6
The firmware version, MAC address etc. can be monitored on the module tab.

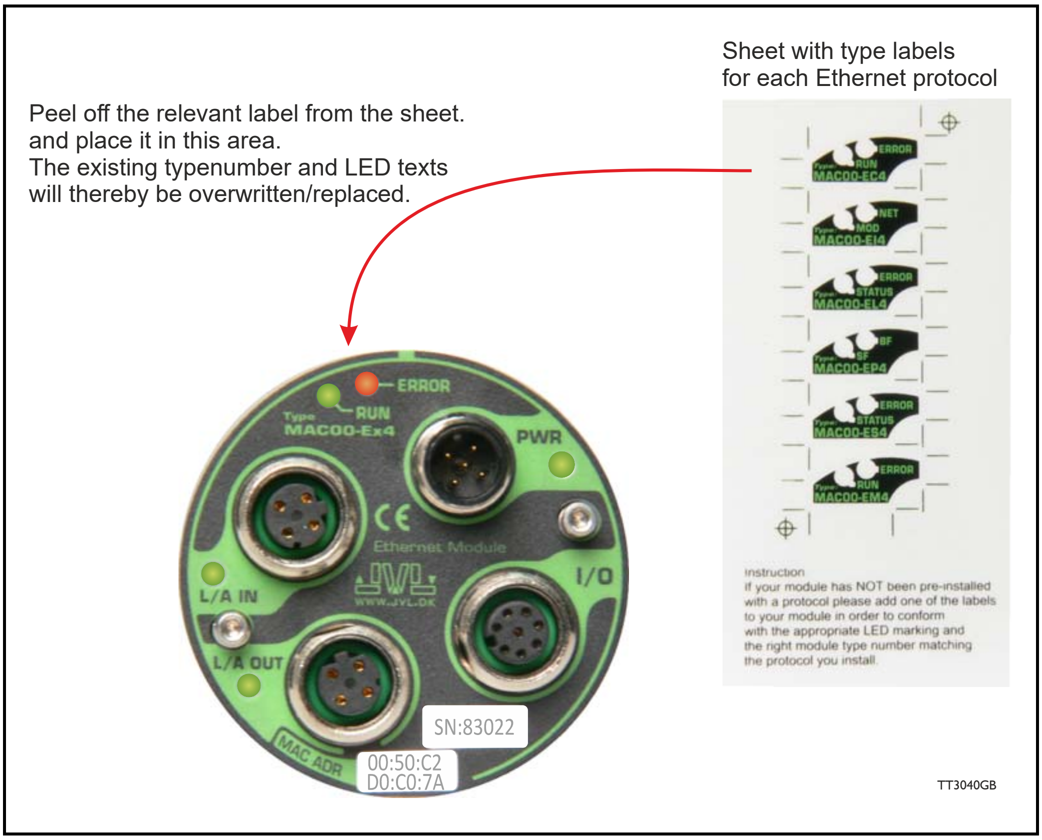


TT3039-02GB

1.2 Module types

Only MAC

Changing the label and typenumber (only MAC products)
This illustration show how to apply the appropriate label in order to change the LED texts and also give the module its unique typenumber after the protocol firmware is loaded.



Changing the label and typenumber (only MIS/MIL products)
No changes need to be done at the MIS/MIL motors. The LED at the rear is universal.

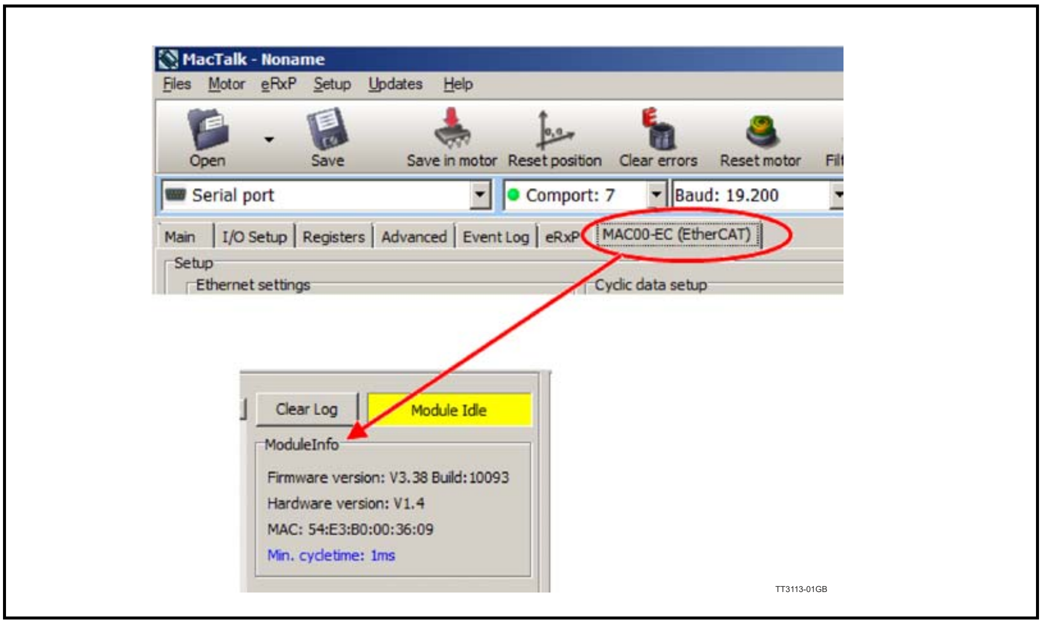
Typenumber overview for MAC and MIS/MIL:

MAC Type	MIS Type	Ethernet Protocol
MAC00-**EC**4/-**EC**41	MIS/MILxxxxxx**EC**xxxxx	EtherCAT
MAC00-**EI**4/-**EI**41	MIS/MILxxxxxx**EI**xxxxx	EtherNET / IP
MAC00-**EL**4/-**EL**41	MIS/MILxxxxxx**EL**xxxxx	EtherNet POWERLINK
MAC00-**EM**4/-**EM**41	MIS/MILxxxxxx**EM**xxxxx	Modbus TCP
MAC00-**EP**4/-**EP**41	MIS/MILxxxxxx**EP**xxxxx	Profinet IO
MAC00-**ES**4/-**ES**41	MIS/MILxxxxxx**ES**xxxxx	Sercos III

1.3 How to find FW/HW version at product

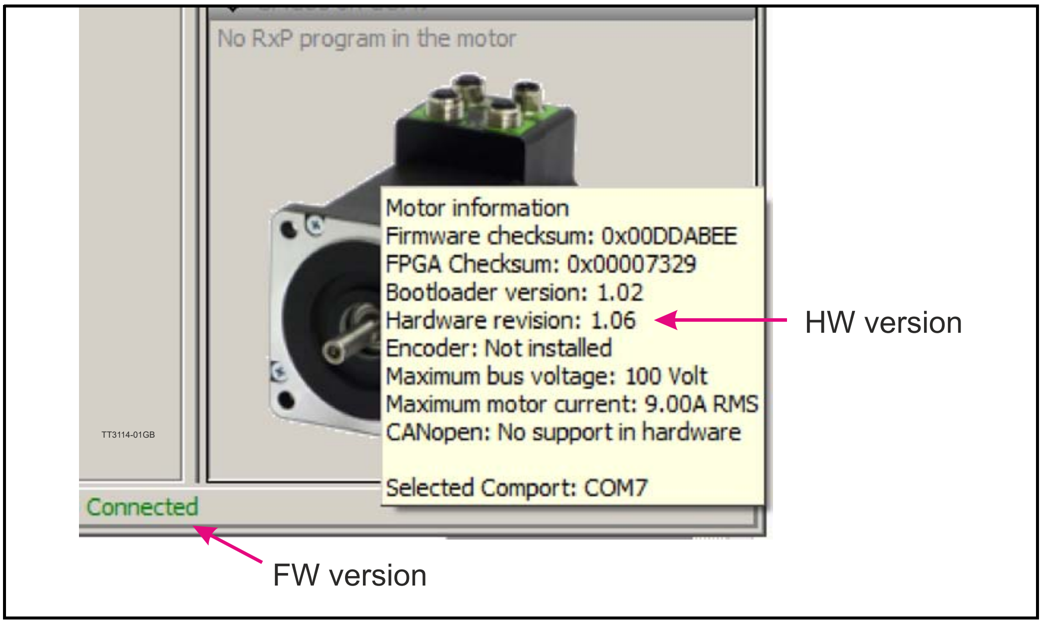
1.3.1 Check Ethernet module version.

The firmware and hardware version of the Ethernet MAC module or the integrated Ethernet module in the MIS/MIL motor can be checked from the MacTalk software when connected to the motor. Select the tab for the Ethernet protocol in use, and check the "Module info" frame. For some protocols and some motors is also the minimum capable cycle time when using a drive profile (CiA402, FSP Drive etc.) listed.



1.3.2 Check motor version.

The hardware version of the motor can be found using MacTalk. Move the mouse curser to the lower left corner and a pop-up box will show with all the relevant info.
The firmware version in the motor can be seen at the green text in the bottom of the picture.



2 General Hardware description

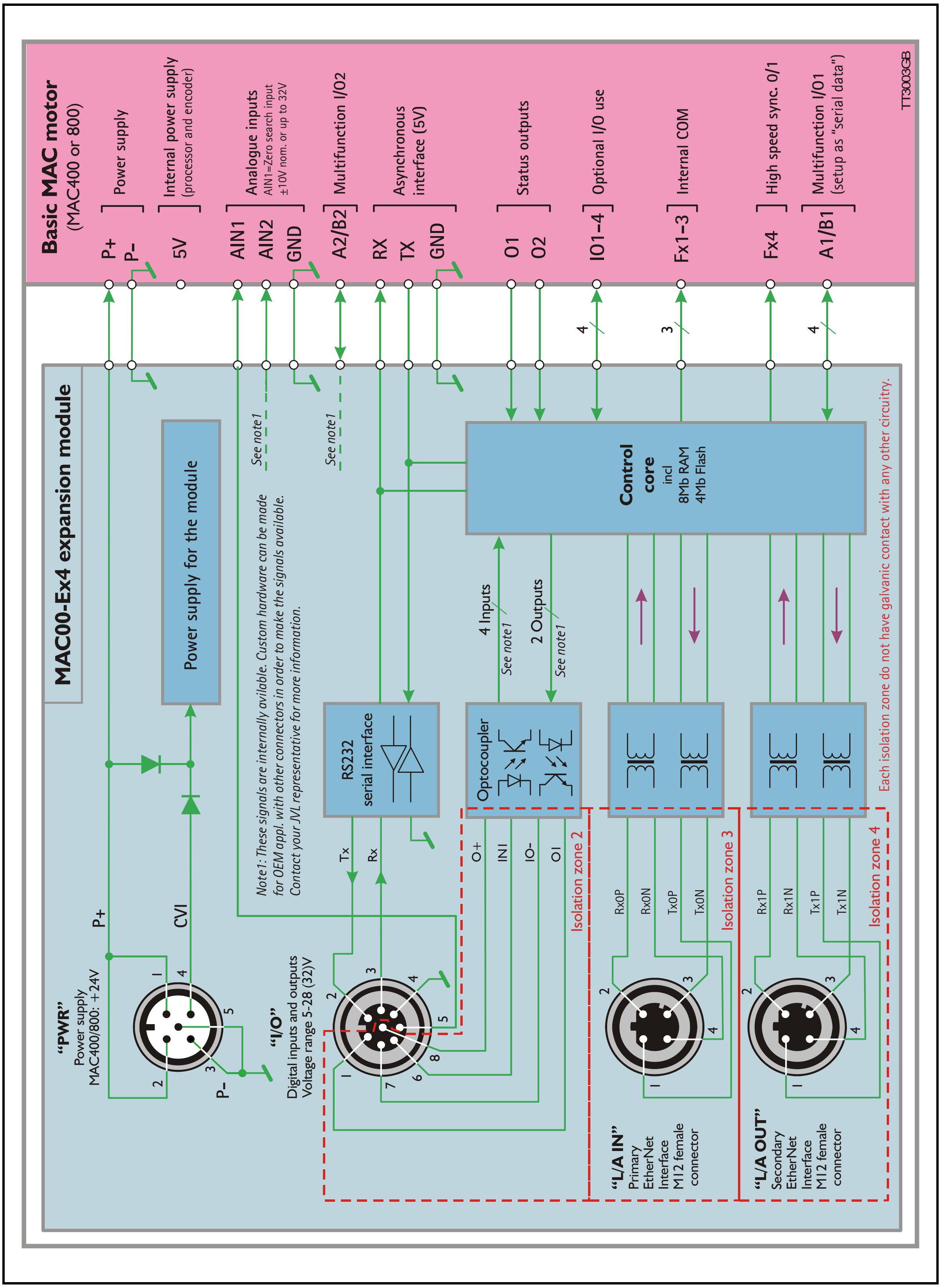
Only MAC

2.1 Hardware introduction

Only MAC

2.1.1 Overall hardware description

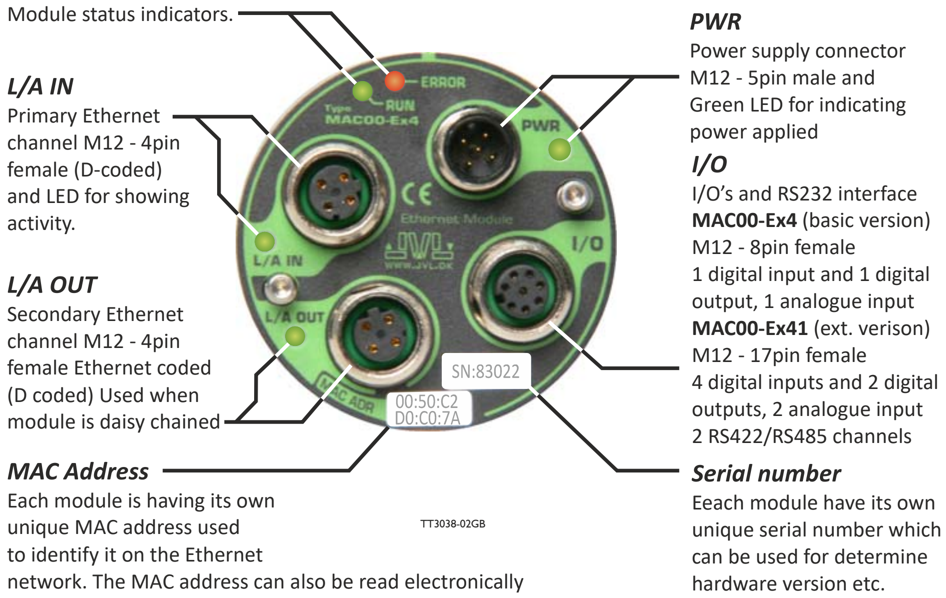
All internal and external main connections can be seen in the illustration below.



2.2 I/O descriptions

2.2.1 Hardware overview

Expansion module MAC00-Ex4 and Ex41 front plate
Neutral module for all the Ethernet protocols
(MAC00-Ex4 shown)



2.2.2 External signals available at the MAC00-Ex4 and Ex41.

Following signals are available.

- **"L/A IN" and L/A OUT" connector.**
  - The Ethernet connection. L/A IN is connected to the upstream master and L/A OUT can be used downstream for the next motors/units in the chain.

- **"I/O" connector.**
  - AIN1 - analogue input +/-10V.
    Can be used as input for the zero search sensor or as general analog input for speed or torque control depending on the what the actual operation mode in the motor has been setup for.
    MAC00-Ex41 offers a second analogue input AIN2. Function similar to AIN1. Please notice that AIN2 is not available if mounted in a MAC050-MAC141.

  - O1 - user output 1
    Can be used as or as general output control able over the Ethernet interface.
    MAC00-Ex41 offers a second digital output (O2). Function similar to O2.

  - RS232 Interface.
    Serial unbalanced interface for connection to a PC or a controller. The protocol is similar to the USB or RS485 interface, which means that all registers/parameters in the motor can be monitored or changed. RS232 is not recommended for long distances (>10m).

  - IN1 - User input 1.
    Can be used as general input which can be read over the Ethernet interface.
    MAC00-Ex41 offers in total 4 digital inputs (IN1, IN2, IN3 and IN4).

- I/O supply and gnd (IO- and O+).
  Used as ground and supply for the user in/output (O1 and IN1).

- 2 RS422/RS485 Multifunction I/O channels
  Only available at the MAC00-Ex41. Can be used for encoder input, full duplex serial communication, encoder output etc.
  Please notice that no multifunction I/O's are available if mounted in a MAC050-MAC141.

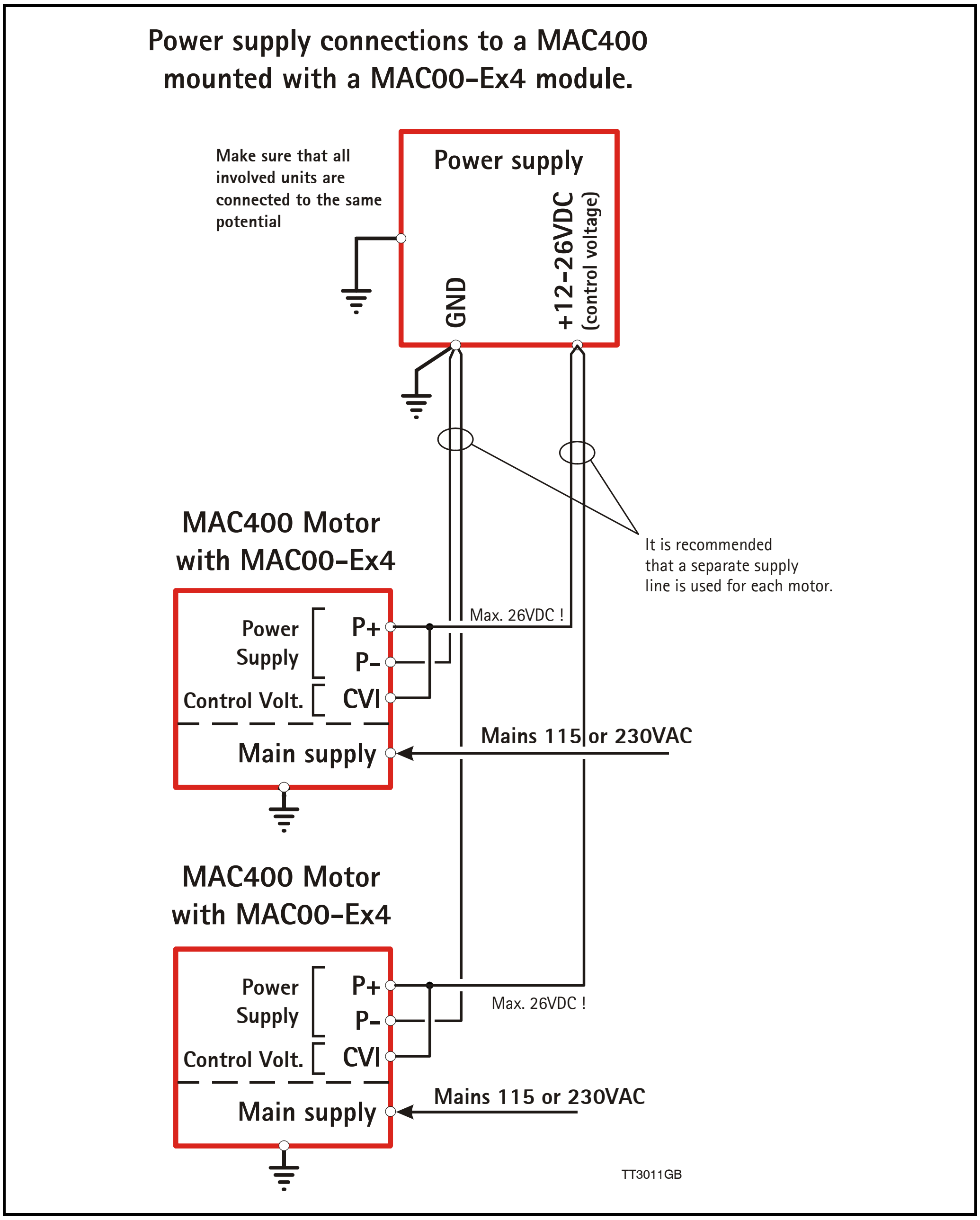
- **"PWR" connector.**
  - 24V supply for the internal control circuitry in the motor.

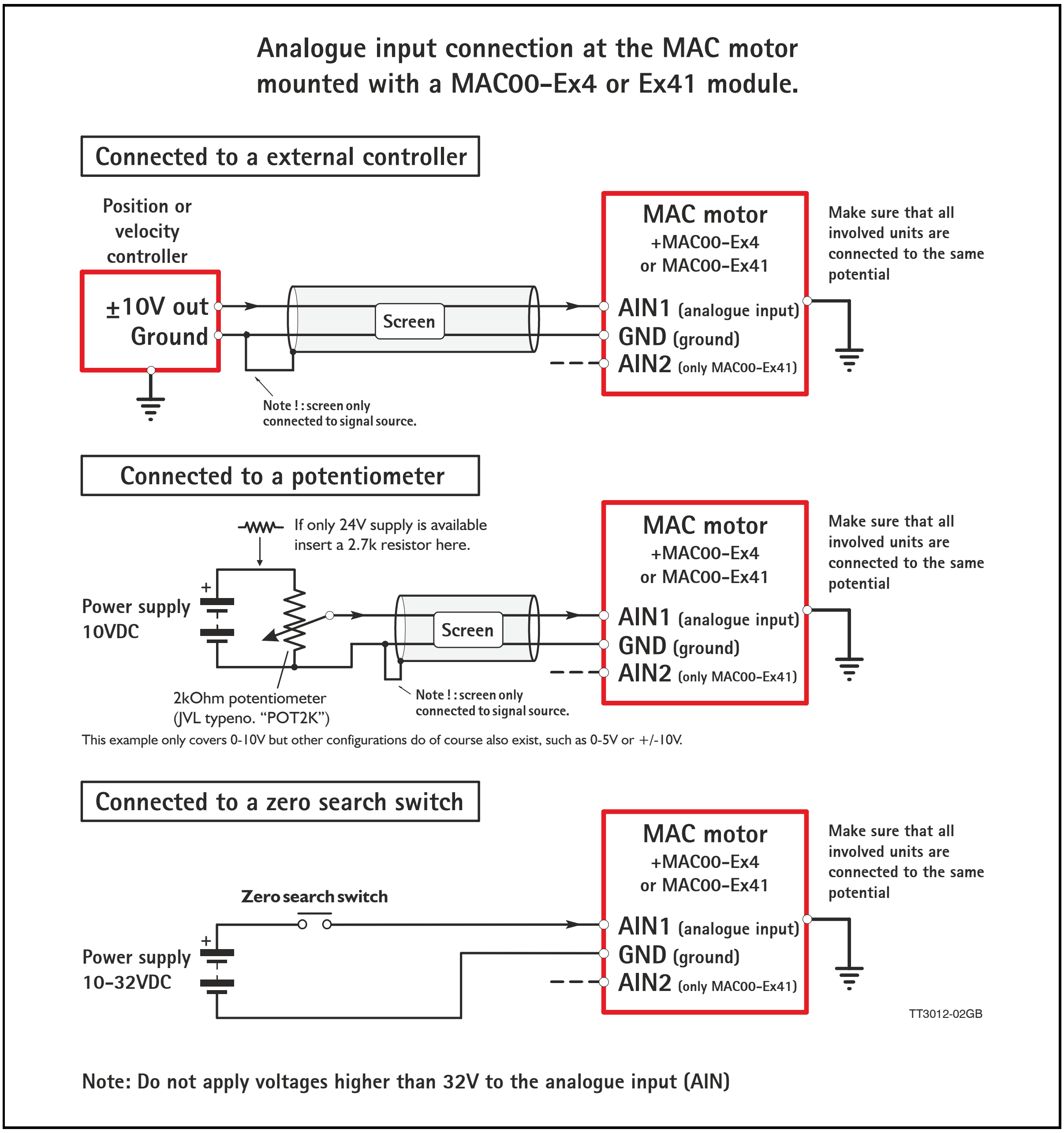
2.2.3 General power supply description

The Ethernet modules can be used in the allmost all the MAC motors but please be aware that to use the MAC50 to 141 they will need the special option : *"A009"* for example *"MAC140-A1-AAAA-A009"*

. The diagram below shows how to connect power to a MAC400 motor mounted with a MAC00-Ex4/-Ex41 module. Please notice that the voltage connected to P+ and/or CVI must stay in the range +12-26VDC. When using a MAC50 to 141 up to 48VDC is allowed.

See also the general power supply description in the MAC motor main manual LB0047. For further information concerning physical connections, see the *Expansion module MAC00-Ex4 (basic version) connector description, page 22*.





2.2.4 Using the analogue input 1 and 2 (AIN1 and AIN2).

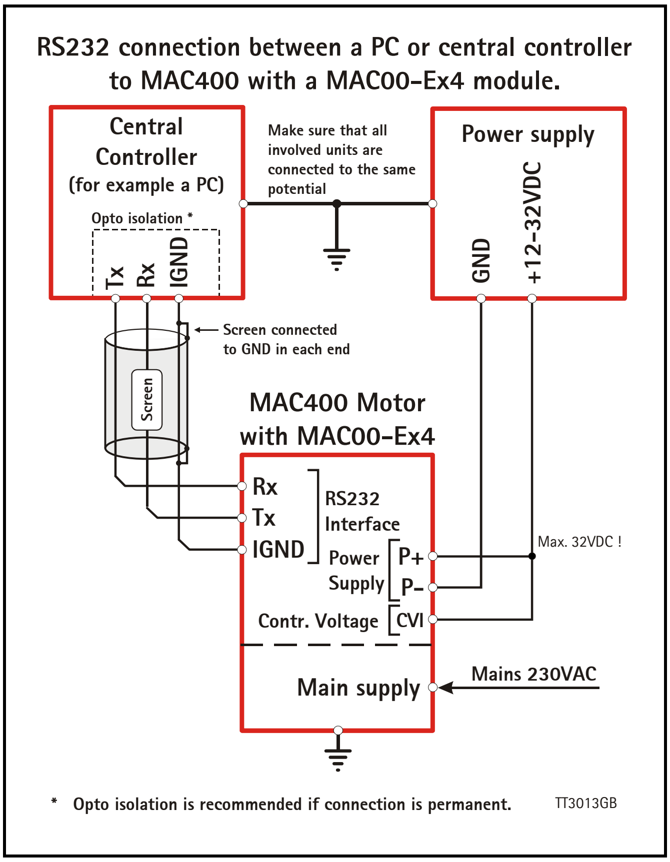
When a MAC00-Ex4 or MAC00-Ex41 module is mounted in the MAC motor, the analogue input(s) is available in the same manner as in the basic motor itself.
The analogue input(s) can be used for several applications and the function of the analogue input is determined by the mode in which the motor is set to operate.
Typically the input(s) is used for controlling the velocity, torque or position of the motor but the input is also used as digital input for zero search or in "Air Cylinder Mode" where it is used as trigger input for the movement done by the motor.
For further information concerning physical connections, see the *Expansion module MAC00-Ex4 (basic version) connector description, page 22*.
Please notice that analogue input 2 (AIN2) is only available at MAC00-Ex41. Please notice that AIN2 is not available if mounted in a MAC050-MAC141.

2.2.5 RS232 - General description.

The RS232 interface is considered the main interface to the motor when the motor is set up using the MacTalk windows software from a PC or from any kind of controller using a RS232 interface.

When connecting the RS232 interface to a PC or controller, the following rules must be followed:

1 Only one motor can be connected at the interface line.
2 Use screened cable.
3 Ensure that GND (interface ground) is also connected.
4 Ensure that all units have a proper connection to safety ground (earth) in order to refer to the same potential.
5 The RS232 interface cable length should not exceed 10 metres.



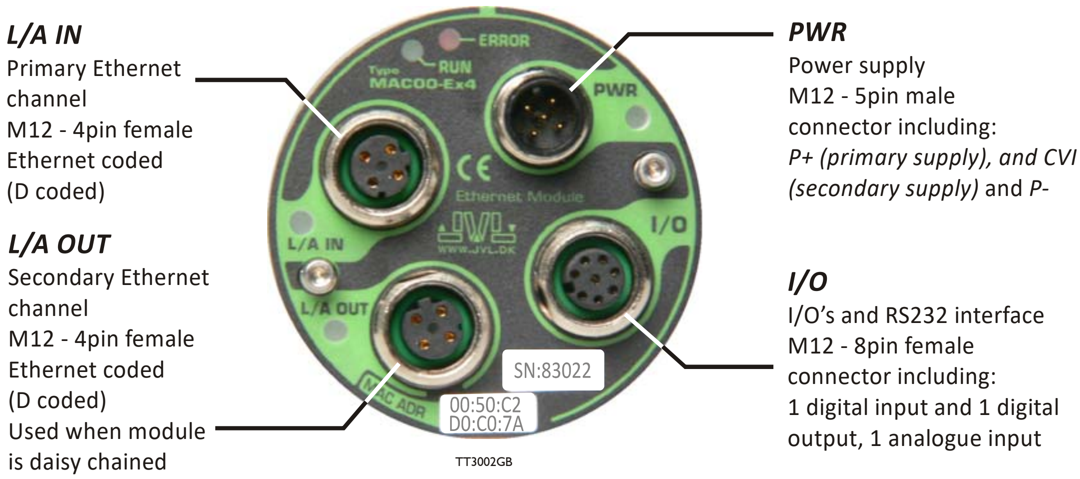
Connectors:
To see the specific connector pin-out please see the chapter *Expansion module MAC00-Ex4 (basic version) connector description, page 22* or *Expansion module MAC00-Ex41 (extended IO) connector description, page 24*

A finished RS232 cable also exist. Please see *Cables for the MAC00-Ex4 (basic version), page 26* or *Cables for the MAC00-Ex41 (extended I/O version), page 27*

2.3 Connector description

Only MAC

Expansion module MAC00-Ex4 front plate



2.3.1 Expansion module MAC00-Ex4 (basic version) connector description

The MAC00-Ex4 offers IP65 protection and M12 connectors which makes it ideal for automation applications where no additional protection is desired. The M12 connectors offer solid mechanical protection and are easy to unplug.

The connector layout:

"PWR" - Power input. M12 - 5pin male connector				
Signal name	**Description**	**Pin no.**	**JVL Cable WI1000-M12F5T05N**	**Isolation group**
P+	Main supply - Connect with pin 2 * When installed in MAC050 to 141 = 12-48VDC When installed in MAC400-4500 = 18-30VDC	1	Brown	1
P+	Main supply - Connect with pin 1 *	2	White	1
P-	Main supply ground. Connect with pin 5 *	3	Blue	1
CVI	Control supply nominal +12-48VDC. **DO NOT connect >50V to this terminal !** A small leakage current may exist on this pin if not used. Connect this terminal to ground if not used.	4	Black	1
P-	Main supply ground. Connect with pin 3 *	5	Grey	1
* Note: P+ and P- are each available at 2 terminals. Make sure that both terminals are connected in order to split the supply current in 2 terminals and thereby avoid an overload of the connector.				

(Continued next page)

(MAC00-Ex4 continued)

"I/O" - I/O's and interface. M12 - 8pin female connector.				
Signal name	**Description**	**Pin no.**	**JVL Cable WI1000-M12 M8T05N**	**Isolation group (See note)**
O1	Output 1 - PNP/Sourcing output	1	White	2
RS232: TX	RS232 interface. Transmit terminal Leave open if unused.	2	Brown	1
RS232: RX	RS232 interface. Receive terminal Leave open if unused.	3	Green	1
GND	Interface ground to be used together with the other signals in this connector. Also ground for the analogue input (AIN1 - pin 5)	4	Yellow	1
AIN1	Analogue input1 ±10V or used for zero search	5	Grey	1
IN1	Digital input 1 - 12-32V tolerant.	6	Pink	2
IO-	I/O ground to be used with the I/O terminals O1 and IN1.	7	Blue	2
O+	Positive supply input to the output circuitry. Connect 5-32VDC to this terminal if using the O1 output.	8	Red	2
"L/A IN" - Ethernet port connector - M12 - 4pin female connector "D" coded				
Signal name	**Description**	**Pin no.**	**JVL Cable WI1046-M12M4S05R**	**Isolation group (See note)**
Tx0_P	Ethernet Transmit channel 0 - positive terminal	1	Brown/White	3
Rx0_P	Ethernet Receive channel 0 - positive terminal	2	Blue/White	3
Tx0_N	Ethernet Transmit channel 0 - negative terminal	3	Brown	3
Rx0_N	Ethernet Receive channel 0 - negative terminal	4	Blue	3
Shield	Outside shield connected to connector housing	Housing	Shield	1
"L/A OUT" - Ethernet port connector. M12 - 4 pin female connector "D" coded				
Signal name	**Description**	**Pin no.**	**JVL Cable WI1046-M12M4S05R**	**Isolation group (see note)**
Tx1_P	Ethernet Transmit channel 1 - positive terminal	1	Brown/White	4
Rx1_P	Ethernet Receive channel 1 - positive terminal	2	Blue/White	4
Tx1_N	Ethernet Transmit channel 1 - negative terminal	3	Brown	4
Rx1_N	Ethernet Receive channel 1 - negative terminal	4	Blue	4
Shield	Outside shield connected to connector housing	Housing	Shield	1

* Note: Isolation group indicate which terminals/circuits that a galvanic connected to each other. In other words group 1, 2, 3 and 4 are all fully independently isolated from each other. Group 1 correspond to the housing of the motor which may also be connected to earth via the DC or AC input supply.

2.3 Connector description

Only MAC

Expansion module MAC00-Ex41 front plate

Extended I/O connections (MAC00-EC41 shown)

L/A IN
Primary Ethernet channel
M12 - 4pin female
Ethernet coded
(D coded)

L/A OUT
Secondary Ethernet channel
M12 - 4pin female
Ethernet coded
(D coded)
Used when module is daisy chained

TT3087-02GB

PWR
Power supply
M12 - 5pin male
connector including:
P+ (primary supply), and CVI (secondary supply) and *P-*

I/O
I/O's and RS232 interface
M12 - 17pin female
connector includes:
4 digital inputs and 2 digital outputs, 2 analogue inputs
2 RS422/RS485 multifunction channels

2.3.2 Expansion module MAC00-Ex41 (extended IO) connector description

The MAC00-Ex41 offers IP65 protection and M12 connectors which makes it ideal for automation applications where no additional protection is desired. The M12 connectors offer solid mechanical protection and are easy to unplug.

The connector layout:

"PWR" - Power input. M12 - 5pin male connector				
Signal name	**Description**	**Pin no.**	**JVL Cable WI1000-M12F5T05N**	**Isolation group**
P+	Main supply - Connect with pin 2 * When installed in MAC050 to 141 = 12-48VDC When installed in MAC400-4500 = 18-30VDC	1	Brown	1
P+	Main supply - Connect with pin 1 *	2	White	1
P-	Main supply ground. Connect with pin 5 *	3	Blue	1
CVI	Control supply nominal +12-48VDC. **DO NOT connect >50V to this terminal !** A small leakage current may exist on this pin if not used. Connect this terminal to ground if not used.	4	Black	1
P-	Main supply ground. Connect with pin 3 *	5	Grey	1
* Note: P+ and P- are each available at 2 terminals. Make sure that both terminals are connected in order to split the supply current in 2 terminals and thereby avoid an overload of the connector.				

(Continued next page)

(MAC00-Ex41 continued)

“I/O” - I/O’s and interface. M12 - 17pin female connector.				
Signal name	**Description**	**Pin no.**	**JVL Cable WI1009M12 M17TxxN**	**Isolation group (see note)**
IN1	Input channel 1. Can be used as digital input	1	Brown	2
GND	Ground intended to be used toghether with the other signals related to isolation group 1 in this connector	2	Blue	1
IN2	Input channel 2. Can be used as digital input	3	White	2
IN3	Input channel 3. Can be used as digital input	4	Green	2
B2- **	RS422/RS485 Multifunction I/O terminal B2-	5	Pink	1
IN4	Input channel 4. Can be used as digital input	6	Yellow	2
A2- **	RS422/RS485 Multifunction I/O terminal A2-	7	Black	1
B2+ **	RS422/RS485 Multifunction I/O terminal B2+	8	Grey	1
OUT+	Positive supply input to the output circuitry (O1, O2). Connect 5-32VDC to this terminal if using one of the O1/O2 outputs. **DO NOT connect >30V to this terminal !**	9	Red	2
A2+ **	RS422/RS485 Multifunction I/O terminal A2+	10	Violet	1
O1	Output 1. Can be used as digital output	11	Grey/pink	2
O2	Output 2. Can be used as digital output	12	Red/blue	2
AIN1	Analog input 1. Can be used as analog input ±10V.	13	White/Green	1
AIN2	Analog input 2. Can be used as analog input ±10V.	14	Brown/Green	1
RS232: RX	RS232 interface. Receive terminal Leave open if unused.	15	White/Yellow	1
IO-	Ground for IN1-4 and O1 and 2. Please notice that this terminal is normally isolated from the main ground and belongs to isolation group 2	16	Yellow/brown	2
RS232: TX	RS232 interface. Transmit terminal Leave open if unused.	17	White/grey	1

“L/A IN” - Ethernet port connector - M12 - 4pin female connector “D” coded				
Signal name	**Description**	**Pin no.**	**JVL Cable WI1046-M12M4S05R**	**Isolation group (See note)**
Tx0_P	Ethernet Transmit channel 0 - positive terminal	1	Brown/White	3
Rx0_P	Ethernet Receive channel 0 - positive terminal	2	Blue/White	3
Tx0_N	Ethernet Transmit channel 0 - negative terminal	3	Brown	3
Rx0_N	Ethernet Receive channel 0 - negative terminal	4	Blue	3
Shield	Outside shield connected to connector housing	Housing	Shield	1

“L/A OUT” - Ethernet port connector. M12 - 4 pin female connector “D” coded				
Signal name	**Description**	**Pin no.**	**JVL Cable WI1046-M12M4S05R**	**Isolation group (see note)**
Tx1_P	Ethernet Transmit channel 1 - positive terminal	1	Brown/White	4
Rx1_P	Ethernet Receive channel 1 - positive terminal	2	Blue/White	4
Tx1_N	Ethernet Transmit channel 1 - negative terminal	3	Brown	4
Rx1_N	Ethernet Receive channel 1 - negative terminal	4	Blue	4
Shield	Outside shield connected to connector housing	Housing	Shield	1

* Note: Isolation group indicate which terminals/circuits that a galvanic connected to each other. In other words group 1, 2, 3 and 4 are all fully independently isolated from each other. Group 1 correspond to the housing of the motor which may also be connected to earth via the DC or AC input supply.
** No connection when module is mounted in a MAC050-MAC141.

2.4 Cable accessories

Only MAC

2.4.1 Cables for the MAC00-Ex4 (basic version)

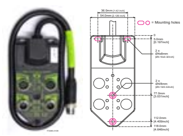
The following cables equipped with M12 connector can be supplied by JVL.

MAC00-Ex4 Connectors				Description	JVL Order no.	Picture
“L/A IN” 4pin male	“L/A OUT” 4pin Female	“I/O” 8pin Female	“PWR” 5pin Male			
		X		RS232 Interface cable. Connects directly from MAC00-Ex4 to a PC Length: 5m (197 inch)	**RS232-M12-1-5-8**	
		X		Cable with M12 **male** 8-pin connector loose wire ends 0.22mm² (24AWG) and screen. Length: 5m (197 inch)	**WI1000-M12M8T05N**	
		X		Same as above but 20m (787 inch)	**WI1000-M12M8T20N**	
			X	Cable (Ø5.5mm) with M12 **female** 5-pin connector loose wire ends 0.35mm² (22AWG) and foil screen. Length: 5m (197 inch)	**WI1000-M12F5T05N**	
			X	Same as above but 20m (787 inch)	**WI1000-M12F5T20N**	
X	X			Ethernet cable with M12 **male** 4pin D coded straight connector, and RJ45 connector (fits into std. Ethernetport)	**WI1046-M12M4S05NRJ45**	
X	X			Ethernet cable with M12 **male** 4pin D coded straight connector, loose ends.	**WI1046-M12M4S05R**	
X	X			Same as above but 15m (590 inch)	**WI1046-M12M4S15R**	
Protection caps. Optional if connector is not used to protect from dust / liquids.						
	X	X		IP67 protection cap for M12 **female** connector.	**WI1000-M12FCAP1**	
X			X	IP67 protection cap for M12 **male** connector.	**WI1000-M12MCAP1**	

Important: Please note that the cables are a standard type. They are not recommended for use in cable chains or where the cable is repeatedly bent. If this is required, use a special robot cable (2D or 3D cable).

2.4.2 Cables for the MAC00-Ex41 (extended I/O version)

The following cables equipped with M12 connector can be supplied by JVL.

MAC00-Ex41 Connectors				Description	JVL Order no.	Picture
"L/A IN" 4pin male	**"L/A OUT"** 4pin Female	**"I/O"** 17pin Female	**"PWR"** 5pin Male			
		(X)		RS232 Interface cable. Connects directly from MAC00-Ex4 to a PC Length: 5m (197 inch) **IMPORTANT:** Only valid if PA0190 is used as adapter.	**RS232-M12-1-5-8**	
		X		Cable with M12 **male** 17-pin connector loose wire ends 0.22mm² (24AWG) and screen. Length: 5m (197 inch)	**WI1009-M12M17T05N**	
		X		Same as above but 20m (787 inch)	**WI1009-M12M17T20N**	
			X	Cable (Ø5.5mm) with M12 **female** 5-pin connector loose wire ends 0.35mm² (22AWG) and foil screen. Length: 5m (197 inch)	**WI1000-M12F5T05N**	
			X	Same as above but 20m (787 inch)	**WI1000-M12F5T20N**	
X	**X**			Ethernet cable with M12 **male** 4pin D coded straight connector, and RJ45 connector (fits into std. Ethernetport)	**WI1046-M12M4S05NRJ45**	
X	**X**			Ethernet cable with M12 **male** 4pin D coded straight connector, loose ends.	**WI1046-M12M4S05R**	
X	**X**			Same as above but 15m (590 inch)	**WI1046-M12M4S15R**	
		X		Junction box for splitting the 17 pin I/O connector into 4 independant connectors. Include also 9 LED's for monitoring the I/O status and communication. Cable length: 0,5m (20 inch)	**PA0190**	
Protection caps. Optional if connector is not used to protect from dust / liquids.						
	X	**X**		IP67 protection cap for M12 **female** connector.	**WI1000-M12FCAP1**	
X			**X**	IP67 protection cap for M12 **male** connector.	**WI1000-M12MCAP1**	

Important: Please note that the cables are a standard type. They are not recommended for use in cable chains or where the cable is repeatedly bent. If this is required, use a special robot cable (2D or 3D cable).

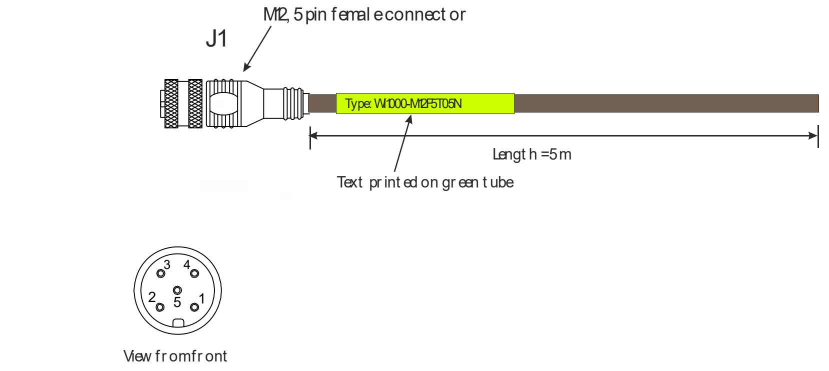
2.4 Cable accessories

Only MAC

Below can be found drawings of the most typical cables used with the Ethernet modules.

2.4.3 Drawing WI1000-M12F5T05N

Cable for connecting power



Color	Signal name	Pin no. J1
Screen		Housing
Brown	P+	1
White	P+	2
Blue	P-	3
Black	(Options)	4
Grey	P-	5

LT0233-10

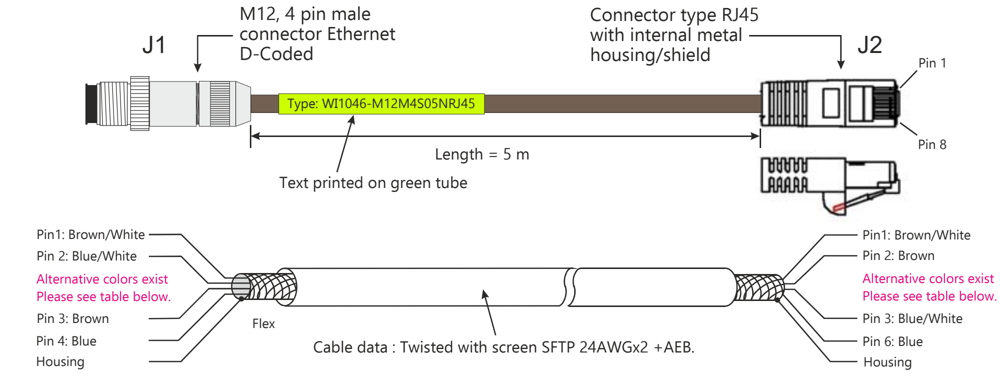
JVL Industri Elektronik A/S
Blokken 42 • DK-3460 Birkerød
Tel: +45 4582 4440
Fax: +45 4582 5550

WI1000-M12F5T05N
M12 shl cable 5 m 5 pin Fem 0°
Power

Version - 10 | Date: 04-07-13
Approved by: - BVJ | Drawn by: PCR

2.4.4 Drawing WI1046-M12M4S05NRJ45

Cable that connects the Ethernet from M12 to RJ45 connectors



Please notice !: 2 versions of the cable type exist:
Type 1: This is the **standard** cable normally stocked
Type 2: This is the **alternative** cable normally not stocked

	Color (Type 1 standard)	Color (Type 2 Alt.)	Signal name	Pin no. J1	Pin no. J2
Twisted pair	Brown/White	Orange/White	Tx+	1	1
	Brown	Orange	Tx-	3	2
Twisted pair	Blue/White	Green/White	Rx+	2	3
	Blue	Green	Rx-	4	6
	Screen	Screen	GND	House	House

LT0179-12

JVL Industri Elektronik A/S
Blokken 42 • DK-3460 Birkerød
Tel: +45 4582 4440
Fax: +45 4582 5550

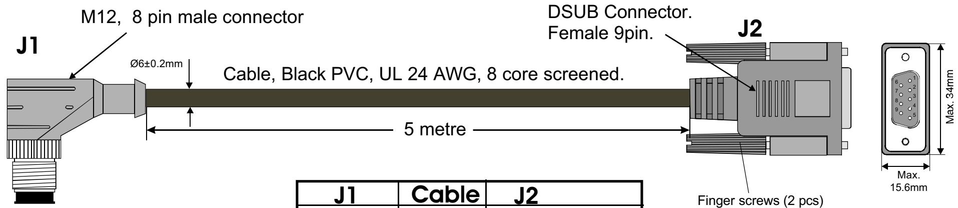
WI1046-M12M4S05NRJ45
M12 shl cable 4 pin male 5m
Ethernet D-Coded RJ45

Version - 1.2 | Date: 8-7-13
Approved by: - BVJ | Drawn by: PCR/BVJ

2.4.5 Drawing RS232-M12-1-5-8

Cable that connects the RS232 from M12 to DSUB connectors.

RS232 communication cable for MAC expansion modules



J1	Cable	J2
Pin no.	Color	Pin no.
1	White	
2	Brown	2
3	Green	3
4	Yellow	5
5	Grey	
6	Pink	
7	Blue	
8	Red	
Not connected	Screen	Housing

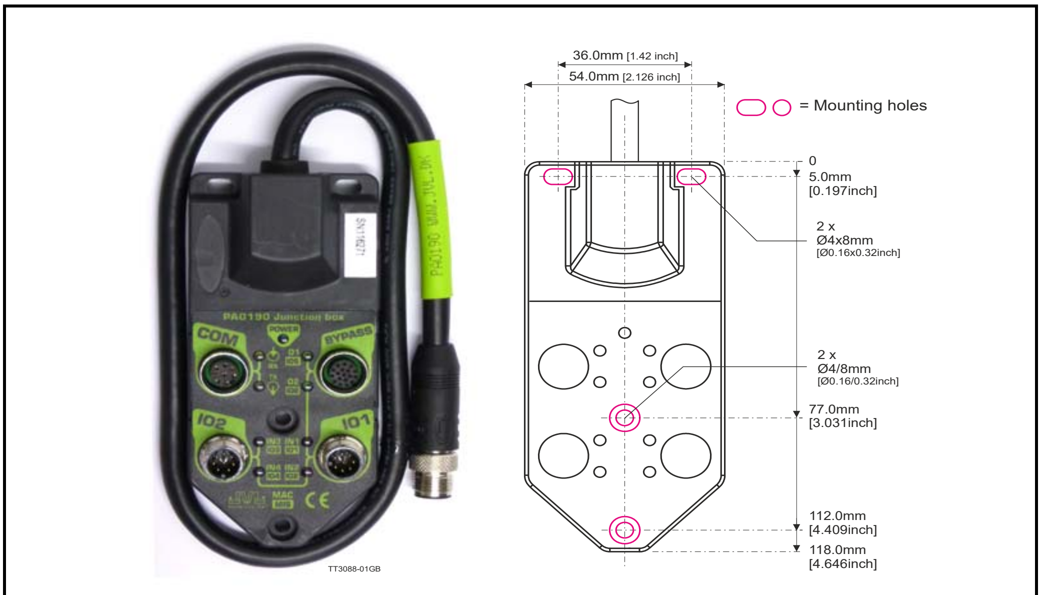
J1 The wires White, Grey, Pink, Blue and Red are not connected and must be insulated
J2 Pin 1, 4, 6, 7, 8, 9 must be left open

LT0082-11

JVL Industri Elektronik A/S Blokken 42 • DK-3460 Birkerød Tel: +45 4582 4440 Fax: +45 4582 5550	**RS232-M12-1-5-8** Interface cable M12 to DSUB		Version - 1.1 Approved by: BVJ - 1.1	Date: 27-9-06 Drawn by: JVJ

2.4.6 Drawing and description of PA0190

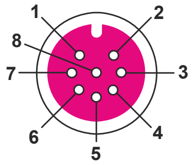
Junction box that splits the connects the signals in the **MAC00-Ex41** "I/O" connector into 4 individual connectors giving an easy and more flexible installation.
Usage hints: The LED's will only work with MIS/MIL or MAC motors where the OUT+ and IO- is supplied from the Ethernet module. See also the I/O description for the module.
If a cable is connected to the "BYPASS" then the Communication pins and GND must be properly connected to valid signals (pins 2,15,17). AND "COM" must not be used. In other words use EITHER the "BYPASS" OR the "COM" connector. Not both.



2.4 Cable accessories

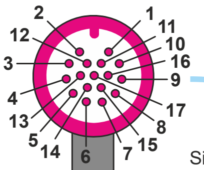
Terminal and LED description of the PA0190 Junction box.

«COM»
8 pin Female



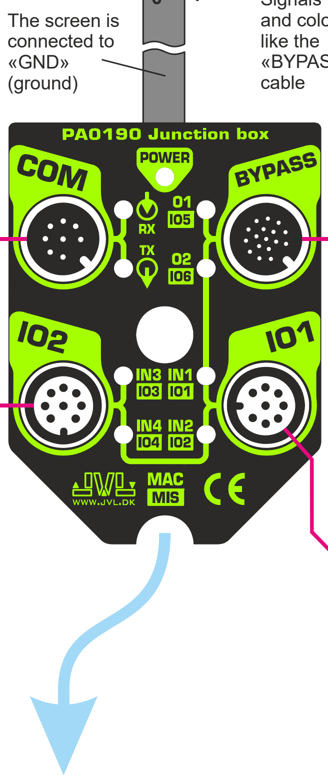
Pin no.	Func.	Color*
Pin 1	IN4	White
Pin 2	TX	Brown
Pin 3	RX	Green
Pin 4	GND	Yellow
Pin 5	RX	Grey
Pin 6	TX	Pink
Pin 7	AIN1	Blue
Pin 8	AIN2	Red
Body	GND	-

Cable from Module
with 17 pin Male connector

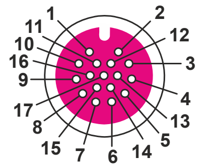


The screen is connected to «GND» (ground)

Signals and colors like the «BYPASS» cable

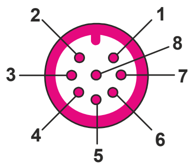


«BYPASS»
17 pin Female



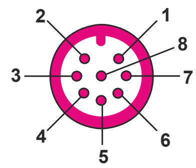
Pin no.	Funct.	Color*
Pin 1	IN1	Brown
Pin 2	GND	Blue
Pin 3	IN2	White
Pin 4	IN3	Green
Pin 5	B2-	Pink
Pin 6	IN4	Yellow
Pin 7	A2-	Black
Pin 8	B2+	Grey
Pin 9	O+	Red
Pin 10	A2+	Violet
Pin 11	O1	GY/PK
Pin 12	O2	RD/BU
Pin 13	AIN1	WH/GN
Pin 14	AIN2	BN/GN
Pin 15	RX	WH/YE
Pin 16	IO-	YE/BN
Pin 17	TX	WH/GY
Body	GND	-

«IO2»
8 pin Male



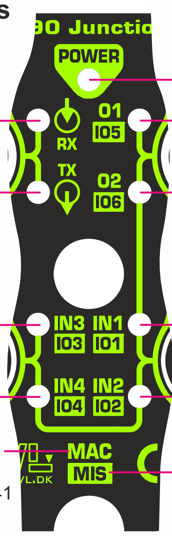
Pin no.	Funct.	Color*
Pin 1	IN4	White
Pin 2	AIN1	Brown
Pin 3	AIN2	Green
Pin 4	IO-	Yellow
Pin 5	O2	Grey
Pin 6	O1	Pink
Pin 7	GND	Blue
Pin 8	O+	Red
Body	GND	-

«IO1»
8 pin Male



Pin no.	Funct.	Color*
Pin 1	IN1	White
Pin 2	IN2	Brown
Pin 3	IN3	Green
Pin 4	IO-	Yellow
Pin 5	O2	Grey
Pin 6	O1	Pink
Pin 7	GND	Blue
Pin 8	O+	Red
Body	GND	-

LED explanations



RX is lit if data is received at the RS232 line.

TX is lit if data is transmitted at the RS232 line.

IN3 is lit if input 3 is activated.

IN4 is lit if input 4 is activated.

Text in NON-inverse must be used when PA0190 is used with the MAC00-Ex41 Ethernet modules.

POWER is lit if terminal ***«OUT+»*** is supplied

O1 is lit if output 1 is activated

O2 is lit if output 2 is activated

IN1 is lit if input 1 is activated

IN2 is lit if input 2 is activated

Text in inverse is NOT relevant when PA0190 is used with the MAC00-Ex41 Ethernet modules.

TT3089-01GB

*** Notes.**
Colors shown are based on the JVL standard cables type:

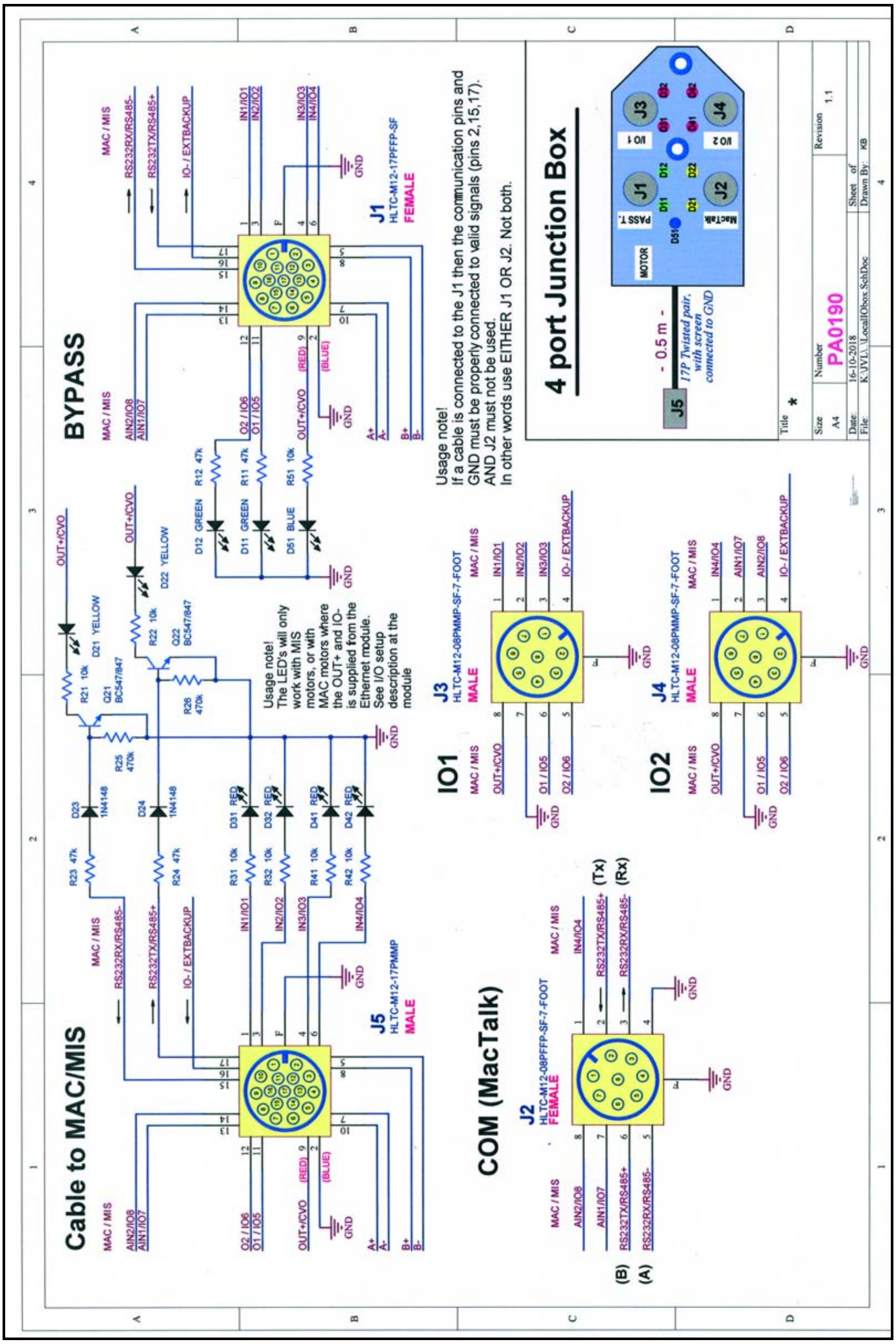
COM connection 8 Pin Male cable WI1000-M12M8TxxN

BYPASS connect. 17 Pin Male cable WI1009-M12M17TxxN

IO1 connection 8 Pin Female cable WI1000-M12F8TxxN

IO2 connection 8 Pin Female cable WI1000-M12F8TxxN

Diagram of the internal details in the PA0190 Junction Box.



3 EtherCAT® Users Guide

3.1 Introduction to EtherCAT®

MAC EtherCAT® Module
Type:
MAC00-**EC**4 (shown) or
MAC00-**EC**41 (extended I/O)

To be used in following servo products:
MAC50, 095, 140 and 141
MAC400 and MAC402
MAC800
MAC1500 and MAC3000

MIS EtherCAT® motors.
Type:
MISxyyzz**EC**xx85 or
MILxyyzz**EC**xx85

TT3041-02GB

3.1.1 Intro to EtherCAT®.

EtherCAT® is a Real Time Ethernet technology which aims to maximize the use of the 100 Mbit, full duplex Ethernet bandwidth. It overcomes the overhead normally associated with Ethernet by employing "on the fly" processing hardware.
An EtherCAT® net consists of a master system and up to 65535 slave devices, connected together with standard Ethernet cabling.

The slave devices process the incoming Ethernet frames directly, extract or insert relevant data and transfer the frame to the next slave device, with a delay of approx. 4μs. The last slave device in the bus segment sends the processed frame back, so that it is returned by the first slave to the master as a kind of response frame.

There are several protocols that can be used as the application layer. In the CANopen over EtherCAT® (CoE) technology, the CANopen protocol is applied to EtherCAT®. CANopen defines Service Data Objects (SDO), Process Data Objects (PDO) and the Object Dictionary structure to manage the parameters. Further information about EtherCAT®, is available from the EtherCAT® technology group http://www.ethercat.org.

3.1 Introduction to EtherCAT®

3.1.2 Abbreviations

Following general used terms are usefull to know before reading the following chapters.

100Base-Tx	100 MBit Ethernet on twisted pairs
CAN	Controller Area Network
CANopen	Application layer protocol used in automation.
CoE	CANopen over EtherCAT®.
DC	Distributed Clock
EMCY	Emergency Object.
EoE	Ethernet over EtherCAT®.
ESI	EtherCAT® Slave Information
ESC	EtherCAT® Slave Controller
ETG	EtherCAT® Technology Group
EtherCAT®	Ethernet Control Automation Technologie
IP	Internet Protocol - IP address ~ the logical address of the device, which is user configurable (not used in EtherCAT®).
MAC	Media Access Controller - MAC address ~ the hardware address of the device (not used in EtherCAT®)
PDO	Process Data Object (for cyclic data)
SDO	Service Data Object (for acyclic data)
SII	Slave Infirmation Interface
XML	eXtensible Markup Language - used for the ESI file.

3.2 Protocol specifications

3.2.1 EtherCAT® - communication

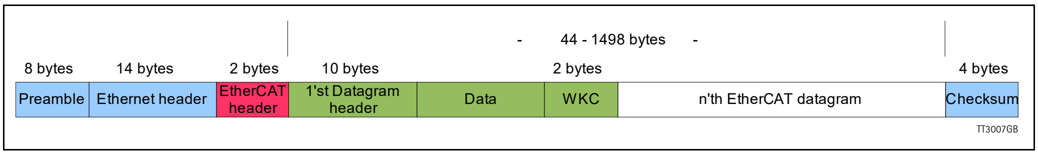
The EtherCAT® fieldbus system is standardised by the EtherCAT® user organisation (ETG). The driving force behind this is the german company, Beckhoff GmbH. Due to the advanced Ethernet technology used for EtherCAT®, in the future, customers can change from other fieldbus systems to EtherCAT® or generally equip new plant models with EtherCAT®.
Communication on EtherCAT® is based on a master/slave operation. The update cycle between master and slave depends on the number of EtherCAT® slaves, the amount of process data of the individual slaves, and the set update time of the master. Due to the ring topology, in every bus cycle only one telegram is sent on the bus. The bus cycle time thus remains exactly the same in every cycle.
Slave addressing can be done in two ways:

- Auto increment addressing
- Fixed node addressing

With Auto increment addressing the master scans the net for slaves, and the slaves are then addressed in the sequence they are physically present on the net. With fixed node addressing, the addresses that each node has programmed, is used.

3.2.2 EtherCAT® frame structure

In EtherCAT®, the data between the master and the slaves is transmitted in Ethernet frames. An EtherCAT® Ethernet frame consists of one or several EtherCAT® telegrams, each addressing individual devices and/or memory areas. The telegrams can be transported either directly in the data area of the Ethernet frame or within the data section of a UDP datagram transported via IP. The EtherCAT® frame structure is pictured in the following figure. Each EtherCAT® telegram consists of an EtherCAT® header, the data area and a working counter (WKC), which is incremented by all EtherCAT® nodes that are addressed by the telegram and have exchanged associated data.



3.2.3 Sync managers

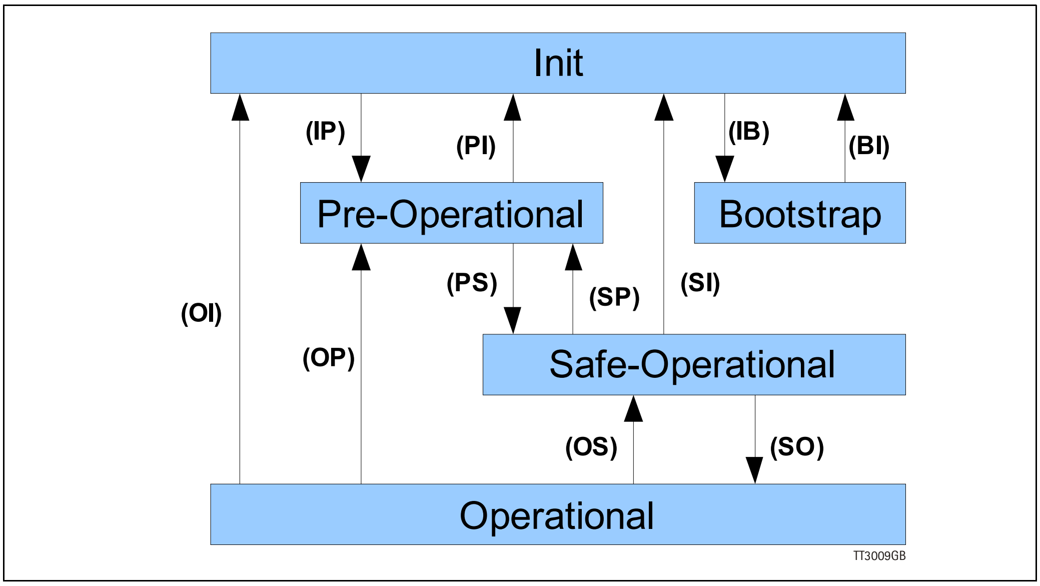
Sync managers control the access to the application memory. Each channel defines a consistent area of the application memory. The adapter module has four sync manager channels. The mailbox protocol (SDO's) and process data (PDO's) are described later in this chapter.

3.2.4 Sync manager watchdog

The sync manager watchdog monitors the output sync managers. If the output data is not updated by the EtherCAT® master within the configured time, the watchdog will activate time out and change the state of the adapter module from Operational to Safe-Operational.
Note: EtherCAT® has been designed so that it provides no way for a slave to monitor the connection to the master if the slave gets no output data.
Note: The drive reaction to a communication fault must be configured in the module write flag register (object 2011 subindex 6 - motor set passive or motor set velocity =0).

3.2.5 EtherCAT® - State machine

Both the master and the slaves have a state machine with the states shown below. After boot the slaves are in INIT state, and then it's up to the master to request state transitions. The standardized EtherCAT® state machine is defined in the following figure. The bootstrap state is not supported.



The module enters the Init state directly after start-up. After this, the module can be switched to the Pre-Operational state. In the Pre-operational state the EtherCAT® mailbox communication is allowed and CoE objects can be accessed by SDOs. After the master has configured the slave, it can switch the module to the Safe-Operational state. In this state input I/O data (PDOs) is sent from the adapter module to the EtherCAT® master, but there is no output I/O data from the master to the module. To communicate output I/O data the master must switch the adapter module to the Operational state.

State description table:

State	Description
Init	State after device initialisation. No Application layer communication (no SDO and PDO communication).
Pre-operational	SDO communication possible. No PDO communication.
Safe-operational	Transmit PDO operational (drive sends data to master)
Operational	Drive fully operational, responds to data via receive PDO
Boot-strap	Not used.

3.2.6 CANopen over EtherCAT®

The application layer communication protocol in EtherCAT® is based on the CANopen DS 301 communication profile and is called CANopen over EtherCAT® (CoE). The protocol specifies the Object Dictionary in the adapter module, in addition to communication objects for exchanging cyclic process data and acyclic messages. In addition to DS301 and the default JVL profile, the MAC00-ECx also supports the DSP402 drive profile *CiA® DSP-402 drive profile, page 53*.

The EtherCAT® module uses the following message types:

- Process Data Object (PDO). The PDO is used for cyclic I/O communication, in other words, process data.
- Service Data Object (SDO). The SDO is used for much slower acyclic data transmission.
- Emergency Object (EMCY). The EMCY is used for error reporting when a fault has occurred in the module or in the drive.

3.2.7 Drive synchronization (only applicable to MAC400+ & MIS/MILxxx)

Distributed clocks

The distributed clock is the primary mechanism built into the EtherCAT network protocol to allow synchronization between the master and slaves in the network. Not every EtherCAT device supports the distributed clock protocol, but those that do can use this mechanism to share a common clock domain across the network. MAC00-ECx supports this when mounted in a MAC400+, and the MIS/MIL also supports this feature.
When the MAC00-ECx is mounted in a **MAC050 - MAC141** DC is **NOT** supported.

When the distributed clock protocol is being used, one clock on the network is selected as the master clock, and all other devices are synchronized to it. The master controller of the network determines which clock will be used as the master clock. The master clock can either reside in the master controller itself, or in one of the slave devices on the network. In many systems the slave devices are able to capture time stamps more accurately than the master controller, so usually the first DC capable slave device in the network is selected as the clock source.

Every EtherCAT slave device which supports the DC feature includes hardware which allows a very accurate local time stamp to be captured when certain registers are written over the network. These time stamps can then be used by the slave device to adjust its local clock to remove the drift between it and the master clock on the network.
The EtherCAT master uses these time stamps to calculate the network delay between devices on the network and to find an offset between each slave's local time and the system time.
Once this offset has been found for each slave, the master writes the offset to a register on the slave's EtherCAT interface hardware. The result is a shared time base for every device on the network which supports the distributed clock protocol.

3.2 Protocol specifications

Sync0 pulse
The distributed clock allows multiple devices on the network to share a common time reference, but doesn't itself provide any real functional synchronization.
Additional hardware is provided on the DC enabled slave devices, which allows a pulse to be generated on the slaves at a fixed period.
This pulse, known as the Sync0 pulse, is used by the slave device to synchronize its internal functions to the network.

The master is responsible for configuring the Sync0 pulse on each slave. Typically, the master finds a sync period which is compatible with all slave devices, and configures the Sync0 signal on all devices to occur simultaneously.

The acceptable sync periods for each slave device can be found in the documentation provided by each device manufacturer. JVL MAC400+ servo motors have an internal position loop with an update rate of 1kHz (1ms) - (alternatively 1.3 or 2.6mS), when used with the MAC00-ECx.
For the synchronization to work, it is needed that the Sync0 period used is an integer multiple of the 1ms position loop update rate. The JVL EtherCAT implementation supports 1 and 2 ms sync0 pulse.The MIS/MIL motor do not have any internal position loop, but nevertheless synchronizes its internal position update to the Sync0 pulse.

Once the Sync0 signal is configured by the master to a multiple of the motor's servo period, the motor will adjust its internal loop to align the start of a servo period with the Sync0 signal.
Since the master typically configures the Sync0 signals of multiple drives on the network to occur simultaneously, the result is simultaneous servo updates on multiple devices.

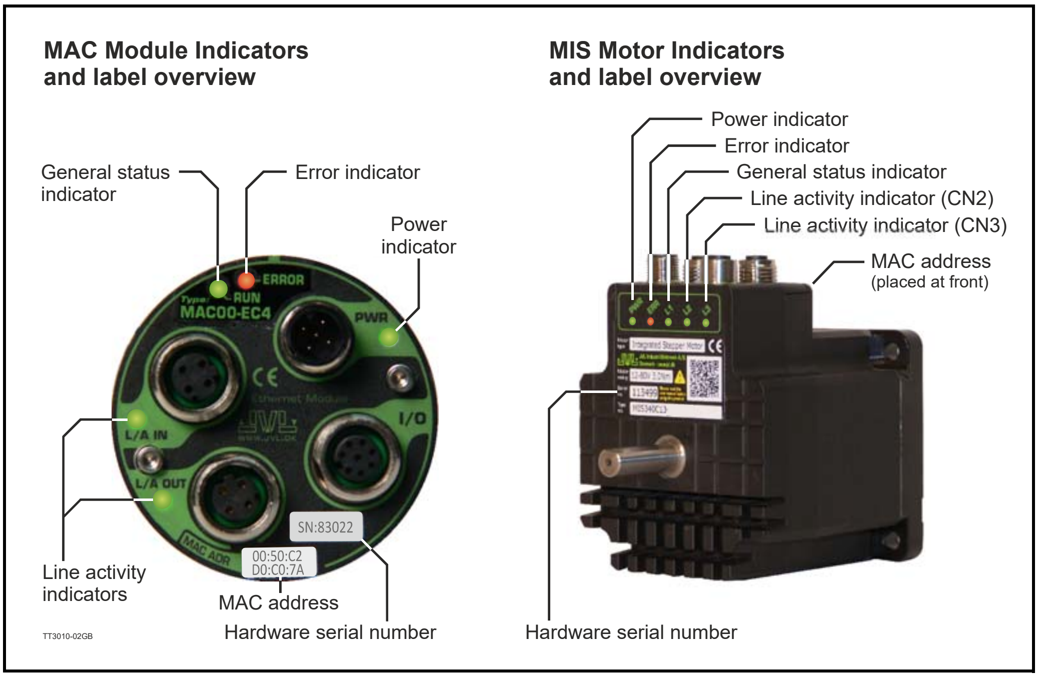
Synchronization specifications
When using synchronization the servo motor has to synchronize to the Distributed Clock of the network. This is done with a PLL circuit which takes a little time to settle.
But when settled it has a maximum jitter of $\pm 1\mu s$.

Settle time of PLL:		
Cycle time	Typical settle time	Max. settle time
1ms	2.4s	5s
2ms	2.6s	5s

3.3 Commisioning

3.3.1 Indicator LED's - description.

The LED's are used for indicating states and faults of the Ethernet. There is one power LED, two link/activity LED's (one for each Ethernet connector), and 2 status LED's.



LED indicator descriptions - Covers both MAC and MIS/MIL (MIx).

LED Text MAC/ MIx	**Colour**	**Constant off**	**Constant on**	**Blinking**	**Single flash**	**Double flash**	**Flickering**
L/A IN / L2	Green	No valid Ethernet connection.	Ethernet is connected.	-	-	-	Activity on line
L/A OUT / L3	Green	No valid Ethernet connection.	Ethernet is connected.	-	-	-	Activity on line
RUN / L1	Green	Device state = INIT	Device state = Operational	Device state = Pre-operational	Device state = Safe-operational	-	-
ERROR / ERR	Red	No error	Critical communication or controller error	General configuration error	Local error	Process data watchdog timeout / EtherCAT® watchdog timeout	Boot up error
PWR / PWR	Red/ Green	Power is not applied.	Power is applied to both motor and modle. **MIS17x,23x:** The LED will lit red constantly if the power supply is too low.	-	-	-	Power is applied to module but no communication with motor.

Notes:
Blinking: Flashing with equal on and off periods of 200ms (2.5Hz). **Single flash:** Repeating on for 200ms and off for 1s. **Double flash:** Two flashes with a period of 200ms followed by 1s off period. **Flickering:** Rapid flashing with a period of approx. 50ms (10 Hz).

3.3 Commisioning

3.3.2 Quick start with TwinCAT (JVL Profile).

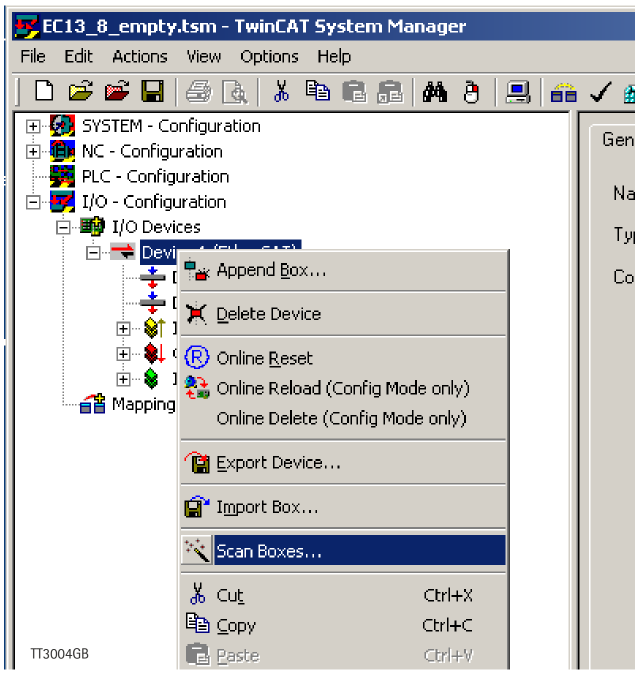
1. Copy the Ethernet slave information file (“JVL ECS V14.XML”) to the folder “..\Twincat\IO\Ethernet\” on the master PC.

2. Apply power, and make sure the *PWR* (power) LED is lit.

3. Connect the Ethernet cable from Master to the L/A IN connector at the MAC module or CN2 at the MIS/MILxxxxxx**EC**xx motor.
   Check that the corresponding LED is lit.

4. Start TwinCAT - system manager on the master, and make sure that a proper Ethernet I/O device is appended (consult your TwinCAT manual).

5. Right click the I/O device, and select "scan boxes".



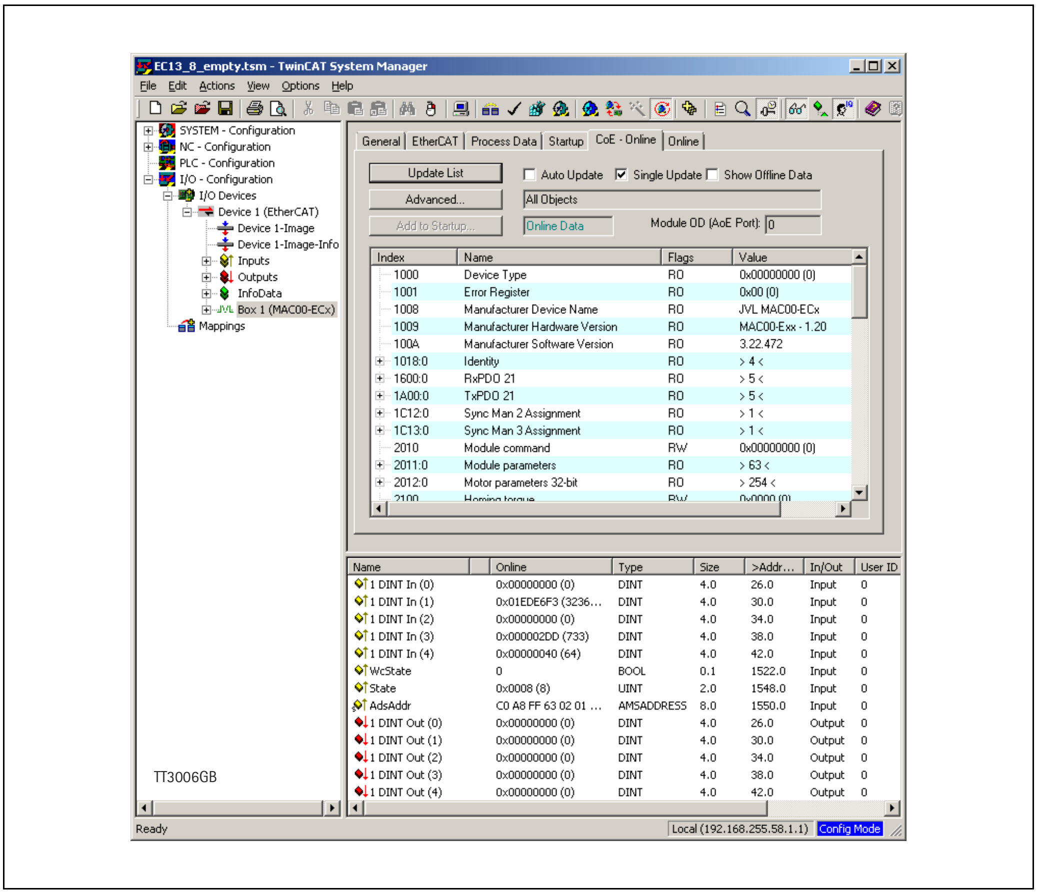
Continued next page

6. The device should now appear in the left side of the TwinCAT window, with a tiny JVL logo.

7. Press F4 (Reload I/O devices), and select the JVL device on the left side of the window.

8. The "L/A IN" LED at the MAC module or "L2" at the MIS/MILxxxxxx**EC**xx motor should now be flashing and the process data should now appear on the bottom right side of the TwinCAT window.

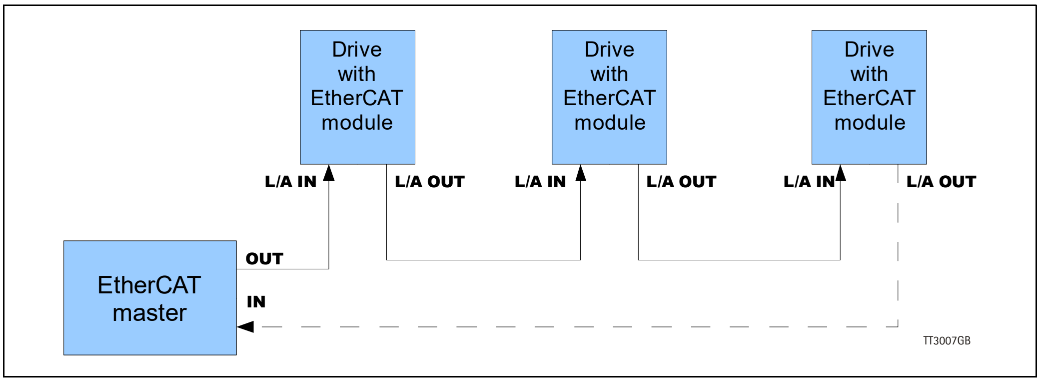
9. By pressing the "CoE online" tab, it's possibly to inspect the CANopen objects, and modify motor and module parameters.



10. If DSP402 drive profile is selected the JVL device is named "Drive" instead of "Box" as shown in the picture.

3.3.3 Mechanical installation

The network cables must be connected to the two M12 connectors (marked "L/A IN" and "L/A OUT") on the module. (Corresponds to CN2 and CN3 at the MIS/MIL motors). The cable from the EtherCAT® master is always connected to the "L/A IN" port. In the line topology, if there are more slave devices in the same line, the next slave device is connected to the port marked "L/A OUT". If there is a redundant ring, the right "L/A OUT" port of the last slave device is connected to the second port of the EtherCAT® master. See the figure below. Standard CAT 5 FTP or STP cables can be used. It is not recommended to use UTP cables in industrial environments, which is typically very noisy.

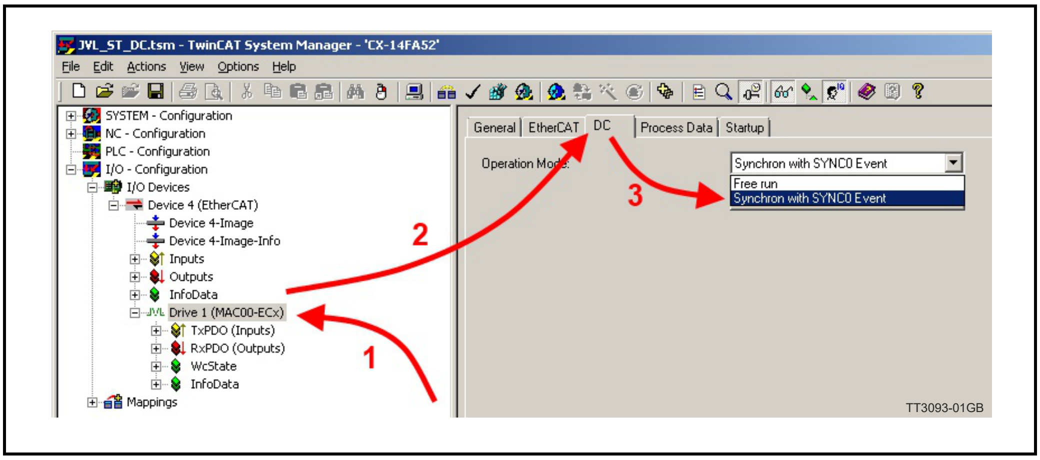


3.3.4 Synchronization configuration

The MAC00-ECx and the MIS/MIL motors supports two different synchronization modes for their process data sync managers. These modes are:

- Free run - *No synchronization. (Requires motor cycle to be 1.0 or 1.3ms.)*
- Synchron with Sync0 Event - Use Distributed Clock, and synchronize to Sync0.

Selection of synchronization mode is in TwinCAT done by selecting the drive and then the DC tab, and there select the appropriate "Operation mode". Please see illustration below.



The "Synchron with Sync0 Event" mode is only accessible in the MIS/MIL and in the MAC00-ECx if mounted in a MAC400+ motor. The MAC050-141 only supports the "Free run" mode.

3.3 Commisioning

Note ! Changes will only become effective after reconfiguring and restarting the EtherCAT master!

Precautions

In a typical EtherCAT system the master will periodically send process data to all devices on the network. Ideally, this process data will be received by the slave devices with a fixed delay relative to the Sync0 signal.
For example, the master may configure the Sync0 period on all slaves to 1 millisecond, and time its communications so that the slaves receive updated process data every millisecond, exactly 50 microseconds before the Sync0 signal occurs.

It's very common in an EtherCAT system for the master to run on a complex PC operating system, and therefore not have the high degree of real time performance that the slaves possess.
In such cases there can be a significant amount of timing jitter on the process data messages that the master sends. For example, if the master has +/- 100 microseconds of jitter on its message transmission timing, then the slave may receive the process data update anywhere from 150 microseconds before Sync0 to 50 microseconds after Sync0. This can cause system level problems such as incorrect trajectory interpolation in cyclic synchronous position mode.

Configuring the process data sync managers to use Sync0 synchronization mode can resolve the problems caused by timing jitter in the master. In this mode the master can compensate for its worst case timing jitter by transmitting the process data to the slaves sufficiently early to ensure that the data will be received before the Sync0 signal. The slaves will not use the process data received until the Sync0 time, so system can remain well synchronized even with a significant amount of timing jitter in the master.

For example, in a system with a cycle time of 1ms and +/-100 microseconds of timing jitter on the master, the master could be configured to transmit its process data with a 300 microsecond offset (30% of the cycle time) from the Sync0 time on the slaves. This would ensure that the slave devices receive the process data well clear of the Sync0 update. Since the slaves are configured in Sync0 synchronization mode, they will not use the updated process data until the Sync0 signal occurs.

Debugging synchronization (Only MAC modules)

The distributed clock and Sync0 signals are all generated internal to the slave devices on the network. This can make it difficult to debug and verify the correct operation of the system synchronization mechanisms. JVL EtherCAT MAC modules provide some useful diagnostic capabilities that can aid the system developer in this area.

One extremely useful tool for debugging synchronization issues is to program a general purpose module output pin to generate a pulse when the Sync0 signal occurs on the drive. Using an oscilloscope, the Sync0 signals of multiple drives can thereby be viewed directly. In a correctly configured system the Sync0 signals of all drives should occur simultaneously with no drift between them.

The function is enabled by issuing command 0x13 to the module command register. The sync0 pulse is then present on the O1 output of the module. Disabling is done with the command 0x14. Please see *Register 15 - Command register, page 260* for information about the module command register, and chapter 2 for how to use the general module I/O's.

3.4 EtherCAT® objects

3.4.1 Process Data Object (PDO/JVL Profile)

PDO's (Process Data Objects) are used for cyclic transfer of time-critical process data between master and slaves. There is one receive PDO and one transmit PDO which is fully user configurable. Tx PDOs are used to transfer data from the slave to the master and Rx PDOs to transfer data from the master to the slave.It is possibly to set up five or eight, 32 bit registers in each PDO, depending on the configuration (*Register 6 - Setup bits, page 256*).
The setup is done with MacTalk or via SDO object 0x2011 subindex 16-31. It requires a save in flash and a power cycle before the new configuration are used. If the configuration of the PDO's, is not altered by the user, the MAC00-EC4/-EC41 module uses the default mapping shown in the tables below.

If module registers is placed in cyclic R/W, then the register number has to be calculated as follows:

Register number = 65536 x sub index.
Example: module command (sub-index 15) = 65536 x 15 = register **983040**

When module registers (register numbers above 65535) are chosen, they **have** to be placed **after** the motor registers in the list of cyclic registers.

NB! If an index is set to zero (No selection), then the following indexes is discarded. Thereby computing resources in the drive are released, which makes much faster cycle times possibly. Please see next paragraph.

Default registers in transmit PDO (Slave > Master) **- Only MAC-ECx**

Object index	Register no.	Motor register short	Motor register description
0	2	MODE_REG	Operating mode
1	10	P_IST	Actual position
2	12	V_IST	Actual velocity
3	169	VF_OUT	Actual torque
4	35	ERR_STAT	Status bits
5	-	-	-
6	-	-	-
7	-	-	-

The motor registers 35, 36, and 211 should NOT be inserted in the cyclic write list, as this may give unpredictable results. For clear of errors, reset of motor etc. please insert the module command register (=983040 in Mactalk) in the cyclic write list and send commands this way.
For a list of commands for the module command register please refer to *Register list., page 254*.

Continued next page

Default registers in receive PDO (Master > Slave) **- Only MAC-ECx**

Object index	Register no.	Motor register short	Motor register description
0	2	MODE_REG	Operating mode
1	3	P_SOLL	Target position
2	5	V_SOLL	Maximum velocity
3	7	T_SOLL	Maximum torque
4	-	-	-
5	-	-	-
6	-	-	-
7	-	-	-

Please notice: Even though all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

Default registers in transmit PDO (Slave > Master) **- Only MIS/MILxxxxxxECxx**

Object index	Register no.	Motor register short	Motor register description
0	2	MODE_REG	Operating mode
1	10	P_IST	Actual position
2	12	V_IST	Actual velocity
3	35	ERR_STAT	Error bits
4	36	WARN_BITS	Warning bits
5	-	-	-
6	-	-	-
7	-	-	-

Default registers in receive PDO (Master > Slave) **- Only MIS/MILxxxxxxECxx**

Object index	Register no.	Motor register short	Motor register description
0	2	MODE_REG	Operating mode
1	3	P_SOLL	Requested position
2	5	V_SOLL	Requested velocity
3	6	A_SOLL	Requested acceleration
4	-	-	-
5	-	-	-
6	-	-	-
7	-	-	-

The MIS/MIL motor registers 24, 35 and 36 should NOT be inserted in the cyclic write list, as this may give unpredictable results. For clear of errors, reset of motor etc. please insert the module command register (=983040 in Mactalk) in the cyclic write list and send commands this way. For a list of commands for the module command register please refer to *Register list., page 254*

3.4.2 Minimum cycle time (JVL Profile)

The minimum cycle time is the minimum amount of time between each cyclic request (PDO) on the Ethernet.
If the module is mounted in MAC050-MAC141 it is possible to add a poll division factor either in the EtherCAT® tab in Mactalk or manually in module register 8 (*Register 8 - Poll division factor., page 258*).

The positions 6-8 is only transferred if enabled, *Register 6 - Setup bits, page 256*.

If operating with values lower than those listed, data loss will occur.

No. of motor registers transmitted in each direction	Motor series MAC050 to MAC141	Motor series MAC400 to MAC4500	Motor series MIS / MIL
1/1	4mS *	360µS *	360µS *
2/2	8mS *	395µS *	395µS *
3/3	12mS *	430µS *	430µS *
4/4	16mS *	465µS *	465µS *
5/5	20mS *	500µS *	500µS *
6/6	24mS *	535µS *	535µS *
7/7	28mS *	570µS *	570µS *
8/8	32mS *	605µS *	605µS *

* The minimum cycle times, is only valid if not sending any acyclic requests while in any operating mode. MODULE registers can be appended as the last registers in the list, at no extra timing cost. Motor register 35 shall be in the cyclic read list, as it is also used internally.

3.4.3 Service Data Objects (SDO)

Service Data Objects (SDOs) are mainly used for transferring non time-critical data, for example, identification, configuration and acyclic data.

3.4.4 Emergency Objects

Emergency Objects (EMCYs) are used for sending fault information from the communication module and the motor to the EtherCAT® network.
They are transmitted whenever a fault occurs in the motor or in the module. Only one Emergency Object is transmitted per fault. EMCYs are transmitted via SDO's.
When the error is no longer present, the module will send a NoError EMCY object once.
The following error codes can be generated:

CANopen Error code	Firmware name	Short description	Applicable to motortype		
			MAC050-MAC141	MAC400-MAC4500	MISxxx MILxxx
0x0000	NO_ERROR	No errors present	X	X	X
0x2221	IPEAK_ERR	Peak error, motor over-current	-	X	-
0x2222	PWM_LOCKED	PWM locked	-	X	-
0x2280	IX_ERR	Phase error	X	-	-
0x3120	UV_ERR	Low AC voltage	-	X	-
0x3210 *	OV_ERR *	Overvoltage on bus *	-	X	X
0x3220	UV_ERR	Undervoltage on bus	X	-	X
0x4210	DEGC_ERR	Temperature too high	-	X	X
0x5112	U24V	Control voltage unstable	-	X	
0x5380	INIT_ERR	Self diagnostics failed	-	X	X
0x5381	STO_ALARM_ERR	Safe torque off alarm	-	X	X
0x5382	FPGA ERROR	Error in accessing FPGA	-	X	-
0x5383	STO_TRIG	STO triggered error		X	X
0x5580	FLASH_ERR	Error in flash write	-	X	-
0x5581	External Memory	Memory error	-	-	X
0x6320	OLD_FILTER	Invalid filter settings	-	X	-
0x7110	UIT_ERR	Regenerative overload	X	X	-
0x7305	INDEX_ERR	Internal encoder error	-	X	X
0x7306	ENC_LOSTPOS	Abs. encoder lost position	-	-	X
0x7307	ENC_REEDERR	Abs. encoder reed error	-	-	X
0x7308	ENC_COMMERR	Abs. encoder com. error	-	-	X
0x7580	SSI_ERR	SSI encoder read error	X	-	X
0x7581	INT_COM_ERR	Internal com. error	X	X	X
0x8180	COM_ERR	Modbus com. Error	-	X	-
0x8181	SLAVE_ERR	Slave error	-	X	-
0x8311	I2T_ERR	Overload	X	X	-
0x8331	FNC_ERR	Function error	X	X	-
0x8480	SPEED_ERR	Overspeed	-	X	-
0x8481	Closed Loop	Closed loop error	-	-	X
0x8611	FLW_ERR	Follow error	X	X	X

* Only triggered in active modes (motor with torque).

Continued next page

Continued

CANopen Error code	Firmware name	Short description	Applicable to motortype		
			MAC050-MAC141	MAC400-MAC4500	MISxxx MILxxx
0x8680	PLIM_ERR	Position limit exceeded	X	X	X
0x8681	NL_ERROR	Neg. limit switch exceeded	X	X	X
0x8682	PL_ERROR	Pos. limit switch exceeded	X	X	X
0x8780	SYNC_ERROR	PLL has lost synchronization to external sync signal.	-	X	-

For a more comprehensive description of the MAC motor errors, please refer to the motor manual - LB0047-xx - chapter 2.7 and search for the firmware name.
The MAC manual can be downloaded using this link: www.jvl.dk...
The structure of the EMCY object is shown in the table below:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
CANopen® error code: MSB (0x10)	CANopen® error code: LSB (0x01)	8-bit error Register = object 0x1001	MAC motor ERR_STAT LSB	MAC motor ERR_STAT	MAC motor ERR_STAT	MAC motor ERR_STAT MSB	Reserved

3.4.5 Object Dictionary

An important part of the CoE protocol is the Object Dictionary, which is different objects specifying the data layout. Each object is addressed using a 16-bit index and possibly a sub index. There are some mandatory objects and some manufacturer specific objects. The objects in the CoE Object Dictionary can be accessed with SDO services.

3.4.6 Mandatory objects:

Name	Index (hex)	Sub Index	Data Type	Read only	Default	Description
Device type	1000		UNSIGNED32	X	0x0	Contains information about the device type.
Error Register	1001		UNSIGNED8	X		This is the mapping error register, and it is part of the emergency object. If some of the sub index are high, an error has occurred. See also *Emergency Objects, page 48*. Mandatory
		Bit 0				Generic error. Mandatory
		Bit 1				Current
		Bit 2				Voltage
		Bit 3				Temperature
		Bit 4				Communication (Overrun)
		Bit 5				Device profile specific
		Bit 6				Reserved
		Bit 7				Manufacturer specific
Manufacturer device name	1008		VISIBLE STRING	X	JVL - MAC00-ECx	
Manufacturer hardware version	1009		VISIBLE STRING	X	1.0	
Manufacturer software version	100A		VISIBLE STRING	X	1.0	Example: Version x.x
Identity object	1018		IDENTITY	X		Contain general information about the module
		0	1..4	X	4h	Number of entries. Mandatory
		1	UNSIGNED32	X	0x0117	Vendor ID, contains a unique value allocated to each manufactor. 117h is JVLs vendor ID. Mandatory.
		2	UNSIGNED32	X	0x0200	Product Code, identifies a specific device version. The MAC00-EC4/-EC41 has the product code 200h
		3	UNSIGNED32	X	-	Revision number.
		4	UNSIGNED32	X	-	Serial number
SyncManager Communication Type	1C00	-	IDENTITY	X	-	Supported communication types
		0	UNSIGNED8	X	4	Number of entries
		1	UNSIGNED8	X	1	Mailbox out
		2	UNSIGNED8	X	2	Mailbox in
		3	UNSIGNED8	X	3	Output process data
		4	UNSIGNED8	X	4	Input process data

3.4.7 Manufacturer specific objects.

The manufacturer specific objects, provides access to all module registers, and all motor registers, as well as a module command object.

	Index (hex)	Sub Index	Type	Read only	Default	Description
Module command	2010	0	UNSIGNED32			Module command object. See possible commands below.
Module parameters	2011	0	UNSIGNED8	X	63	Subindex count
		1	UNSIGNED32	X		Access to module register N
Motor parameters	2012	0	UNSIGNED8	X	254	Subindex count
		N	UNSIGNED32			Access to the motor parameter n
Extended motor parameters	2013	0	UNSIGNED8	X	254	Subindex count
		N	UNSIGNED32			Access to the motor parameter N+255

Note: Module parameters are not automatically saved to permanent memory after a change. The parameters can be saved permanently by applying a "Save parameters to flash" command afterwards.

3.4.8 Object 0x2010 - Subindex 0

This object is used for sending commands to the module and is write only. It is analogue to writing to object 2011 subindex 15.
The possible commands are shown in *See "Register list." on page 254.*

3.4.9 Object 0x2011

The module registers is mapped to object 0x2011. The subindex 3-31 is R/W, the rest is read only.
The register numbers are used as sub indexes in the object. See register descriptions in chapter 8 - page 254.

3.4.10 Object 0x2012

Object 0x2012 are for acyclic view or change of motor registers.
Please find a complete list of register descriptions in the appendix.

Registers relevant for the MAC050 to 141 motors:
Motor registers MAC050 - 141, page 292

Registers relevant for the MAC400 to 4500 motors:
Motor registers MAC400 - 4500, page 301

Registers relevant for the MIS/MILxxx motors:
Motor registers MISxxx, page 320

3.4.11 Object 0x2013 (only applicable to MAC400-4500).

Object 0x2013 are for acyclic view or change of motor registers above 255.
To access a motor register the register number is calculated as follows:

Motor register number = Subindex + 255

3.4.12 EtherCAT® Slave Information file

EtherCAT® Slave Information file (ESI) is a XML file that specify the properties of the slave device for the EtherCAT® master and contains information on the supported communication objects. EtherCAT® Slave Information files for JVL drives are available through your local JVL representative. If TwinCAT is used for master then the XML-file shall be copied to the folder "..\TwinCAT\Io\EtherCAT\".

3.5 CiA® DSP-402 drive profile

3.5.1 Introduction

The MAC00-ECx supports the DSP-402 standard from CiA® http://www.can-cia.com/. Please refer to this standard for full details of the functions. The DSP-402 is only a standard proposal and might be changed in the future. We reserve the right to change future firmware versions to conform to new versions of the standard. Not all of the functionality, described in DSP-402, is supported. But all the mandatory functions are supported. The following operation modes are supported:

Mode name	Short	Mode no.	Comments
Profile Position mode	pp	1	
Profile Velocity mode	pv	3	
Homing Mode	hm	6	
Cyclic Synchron Position	csp *	8	Default PDO addresses this mode. *MAC050-141 only in Free Run mode.*
Cyclic Synchron Velocity	csv *	9	*MAC050-141 only in Free Run mode.*
Cyclic Synchron Torque	cst *	10	*Only MAC400 - MAC4500*

* When using one of the cyclic modes it is strongly recommneded to use Distributed Clock, in order not to loose any cyclic frames.

WARNING: The cyclic modes (8,9,10) normally used by masters are NOT recommended for MAC050-141, as these motors don't support Distributed Clock, and have a minimum cycle time of 16ms when using DSP-402.

Preconditions:
Before the DSP-402 mode with all the described features can be used, the firmware in the MAC00-ECx module or the MIS/MILxxxxxxECxx motor must be updated to at least firmware version 3.36. Besides, version 22 of the XML file must be used "JVL ECS V22.xml" found on the web page http://www.jvl.dk.
See also *How to find FW/HW version at product, page 13*.

- The start mode of the motor must be set to passive.
- No power up Zero searches must be selected.
- If absolute movement is used, the 'resynchronize after passive mode' must be set.
- The DSP-402 drive profile must be enabled and saved to flash (please see next paragraph).

When using DSP-402 mode, manipulating motor parameters with object 0x2012 can corrupt the behavior of the DSP-402 functions. Also be aware that manipulating parameters in MacTalk should be avoided when using DSP-402.

3.5.2 Selecting DSP-402 drive profile

As default the JVL EtherCAT module uses the CiA 402 drive profile. But if it - for some reason - is not selected, then enable it this way:
In MacTalk in the Ethernet tab the checkbox "Enable DSP402 drive profile" is checked, and the "Apply and save" button is pressed.
Then after a power cycle the MAC00-ECx module or the MIS/MILxxxxxxECxx motor will wake up with DSP-402 drive profile enabled instead of the JVL profile.
If already having a TwinCAT project, then delete the JVL box, and do a new scan for boxes. Now the JVL device will appear as a drive instead.

3.5 CiA® DSP-402 drive profile

3.5.3 Supported objects

Most of the DSP402 parameters start up in the module with default values. A few of them are set depending on the motor type the module is mounted in - either the MAC50-141, MAC400+ or the MIS/MILxxxxxxECxx motor.
None of the parameters can be saved to flash in the module. The following table shows the additional object dictionary defined for DSP-402 support. *Continued next page*

3.5 CiA® DSP-402 drive profile

Index (hex)	Sub idx.	Name	Type	Attributes	Default value
Distributed Clock					
0x1C32	0	Synchronized output	U8	RO	-
	1	Synchronization type	U16	RW	0
	2	Cycle time	U32	RW	1000000 - *MIS/MIL and MAC400+* 20000000 - *MAC050-141*
	4	Synchronization types supported	U16	RO	5 - *MIS/MIL and MAC400+* 1 - *MAC050-141*
	5	Minimum cycle time	U32	RO	1000000 - *MIS/MIL and MAC400+* 20000000 - *MAC050-141*
	6	Calc and copy time	U32	RO	71000 - *MAC400+* 439000 - *MIS/MIL* 8000000 - *MAC050-141*
	9	Delay time	U32	RO	450000 - *MAC400+* 2000000 - *MIS/MIL* 2000000 - *MAC050-141*
	12	Cycle time too small	U16	RO	-
	32	Sync error	Boolean	RO	-
0x1C33	0	Synchronized output	U8	RO	-
	1	Synchronization type	U16	RW	= 0x1C32:04
	2	Cycle time	U32	RW	= 0x1C32:04
	4	Synchronization types supported	U16	RO	= 0x1C32:04
	5	Minimum cycle time	U32	RO	= 0x1C32:04
	6	Calc and copy time	U32	RO	20000
	12	Cycle time too small	U16	RO	-
	32	Sync error	Boolean	RO	False
Device data					
6402	0	Motor type	U16	RO	10
6403	0	Motor catalogue number	STR	RO	MACxxx
6404	0	Motor manufacturer	STR	RO	JVL Industri Elektronik A/S
6405	0	http motor catalogue address	STR	RO	www.JVL.dk
6502	0	Supported drive modes	U32	RO	0x00000025

Continued next page

3.5 CiA® DSP-402 drive profile

Index (hex)	Sub idx.	Name	Type	Attri-butes	Default value
6503	0	Drive catalogue number	Str.	RO	MACxxx
6504	0	Drive manufacturer	Str.	RO	JVL Industri Elektronik A/S
6505	0	http drive catalogue address	Str.	RO	www.JVL.dk
Analogue/Digital I/O					
2101*	0	Analog input 1	I16	RO, P	-
2103**	0	Motor temperature	I8	RO, P	-
60FD	0	Digital inputs	U32	RO, P	1 Input available in MAC00-EC4 4 Inputs available in MAC00-EC41 Up to 8 Inp. avail. in MIS/MILxxx
60FE	0	Digital outputs	U8	RO, P	1 Output available in MAC00-EC4 2 Outputs avail. in MAC00-EC41 Up to 8 out avail. in MIS/MILxxx
	1	Physical outputs	U32	RW, P	0
	2	Bit mask	U32	RW, P	0x03 – MAC00-ECx 0xFF – MIS/MILxxx
Device control					
10F3	0	Diagnosis History	U8	RO	See ETG1020 for comprehensive description
	1	Maximum messages	U8	RO	-
	2	Newest message	U8	RO	-
	3	Newest acknowledged mess.	U8	RW	Only values 6-37 are accepted
	4	New message available	U8	RO	-
	5	Flags	U16	RW	Only bit 1 and 2 are writable
	6-37	Diagnosis message	STR	RO	-
603F	0	Error code	U16	RO, P	-
6040	0	Control word (See below for supported features)	U16	RW, P	-
6041	0	Status word (See below for supported features)	U16	RW, P	-
605A	0	Quick stop option code (See below for supported features)	I16	RW	2
6085	0	Quick stop deceleration	U32	RW	50000
6060	0	Modes of operation	I8	RW, P	-
6061	0	Modes of operation display	I8	RO, P	-
6072*	0	Max torque	U16	RW, P	1000
607E	0	Polarity	U8	RW	0
Position parameters					
6064	0	Position actual value	I32	RO, P	-
6067	0	Position window	U32	RW	100
6068*	0	Position window time	U16	RW	6
607A	0	Target position	I32	RW, P	-

Continued next page

3.5 CiA® DSP-402 drive profile

Index (hex)	Sub idx.	Name	Type	Attri-butes	Default value
Position parameters (continued)					
607D	0	Software position limit	U8	RO	2
	1	Min.	I32	RW	0
	2	Max.	I32	RW	0
6080	0	Max motor speed	U32	RW	*Depending on motor type*
6081	0	Profile velocity	U32	RW, P	100
6083	0	Profile acceleration	U32	RW, P	15000
6086	0	Motion profile type	I16	RW	0
60F4	0	Following error actual value	I32	RO, P	-
Velocity parameters					
606B	0	Velocity demand value	I32	RO, P	-
606C	0	Velocity actual value	I32	RO, P	-
606D*	0	Velocity window	U16	RW	100
606E*	0	Velocity window time	U16	RW	6
60FF	0	Target velocity	U32	RW, P	-
Torque parameters					
6071*	0	Target Torque	I16	RW, P	-
6077**	0	Torque actual value	I16	RO, P	-
Homing mode					
2100	0	Homing torque	U16	RW	30
607C	0	Home offset	I32	RW	0
6098	0	Homing method	I8	RW	0
6099	0	Homing speeds	U8	RO	2
	1	Speed during search for switch	U32	RW	50
	2	Speed during search for zero	U32	RW	50
609A	0	Homing acceleration	U32	RW	5000
Factors					
608F	0	Position encoder resolution	U8	RO	2
	1	Encoder increments	U32	RW	*Depending on motor type*
	2	Motor revolutions	U32	RW	1
6091	0	Gear ratio	U8	RO	2
	1	Motor revolutions	U32	RW	1
	2	Shaft revolutions	U32	RW	1
6092	0	Feed constant	U8	RO	2
	1	Feed	U32	RW	*Depending on motor type*
	2	Shaft revolutions	U32	RW	1

”Str”	String, ”I” = Integer, ”U” = Unsigned integer, figures = number of bits.
“RO”	Read Only, ”RW” = Read and Writeable, ”P” = PDO map able.
“Boolean”	-
*	Only available in MAC00-ECx.
**	Only available with MAC400+ and MIS/MILxxx

3.5 CiA® DSP-402 drive profile

WARNING !!! When using the CiA402 objects it is NOT recommended to change motor registers in Mactalk, or by object 0x2012/0x2013, as changes there are NOT reflected in the CiA402 objects.

3.5.4 Supported features in Control word (object 0x6040).

Bit	Meaning	Supported
0	Switch on	Yes
1	Enable voltage	Yes
2	Quick stop	Yes
3	Enable operation	Yes
4	Operation mode specific	Yes (HM: Start homing/PP: New setpoint)
5	Operation mode specific	Yes (PP: Change set immediately)
6	Operation mode specific	-
7	Fault reset	Yes
8	Halt	Yes
9	Operation mode specific	-
10	Reserved	-
11-15	Nanufacturer specific	-

3.5.5 Supported features in Status word (object 0x6041).

Bit	Meaning	Supported
0	Ready to switch on	Yes
1	Switch on	Yes
2	Operation enabled	Yes
3	Fault	Yes
4	Voltage enabled	Yes
5	Quick stop	Yes
6	Switch on disabled	Yes
7	Warning	Yes (Only in MIS/MILxxx)
8	Manufacturer specific	-
9	Remote	-
10	Operation mode specific	Yes (Target reached/Status toggle)
11	Internal limit active	Yes (Only in MIS/MILxxx = Position limit active)
12	Operation mode specific	Yes (Homing done/Set point ack/Drive follows the command value)
13	Operation mode specific	Yes (Following error)
14-15	Manufacturer specific	-

3.5.6 Supported values of quick stop option code (object 0x605A)

Bit	Meaning	Supported
0	Disable drive function	Yes
1	Slow down on slow ramp and transit into Switch On Disabled	Yes
2	Slow down on quick ramp and transit into Switch On Disabled	Yes
3	Slow down on current limit and transit into Switch On Disabled	No/Same as 2
4	Slow down on voltage limit and transit into Switch On Disabled	No/Same as 2
5	Slow down on slow ramp and stay in quick stop active	Yes
6	Slow down on quick stop ramp and stay in quick stop active	Yes
7	Slow down on current limit and stay in quick stop active	No/Same as 6
8	Slow down on voltage limit and stay in quick stop active	No/Same as 6

3.5 CiA® DSP-402 drive profile

3.5.7 Manufacturer specific objects when using CiA402.

The objects in the previous paragraph is described more closely in the CiA402 drive profile documentation, except the manufacturer specific ones which are described in detail here.

Object 0x2100 Homing torque
Only applicable to EtherCAT modules installed in servo motors (MACxxx).
This read-writeable object describes the torque used during torque homing with the manufacturer specific homing modes -1, -2, -3 and -4.
The units of the object are the same as used for other torque objects, for example object 0x6071. It is recommended to set this to a low value in order to avoid damaging the machine, if using torque homing.

Object 0x2101 Analogue input 1
Only applicable to EtherCAT modules installed in servo motors (MACxxx).
In this read only object it is possible to read the status of the motor analog input (ANINP).
It is possible to map this object in the cyclic read PDO.
The range of this object is ±1023 corresponding to ±10V on the input pin. This gives approximately 9.775mV/unit.

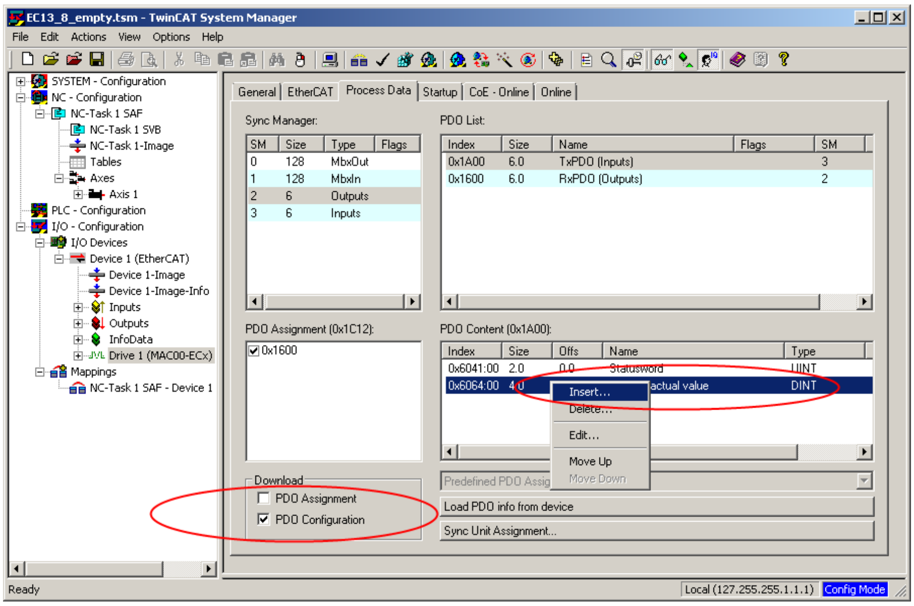
Object 0x2103 - Motor temperature
Only applicable to MAC400+ and MIS/MILxxx motors.
This read only object is the internal temperature of the motor controller, expressed in degrees celcius.
It is possible to map this object in the cyclic PDO.

3.5.8 PDO's (Process Data Objects)

When selecting the DSP-402 drive profile the setup and functioning of the PDO's is very different from the default JVL profile. In the DSP-402 drive profile there is one PDO in each direction. Each PDO can hold up to eight objects and the PDO's are fully dynamic and is altered in TwinCAT, in the "Process data" tab.

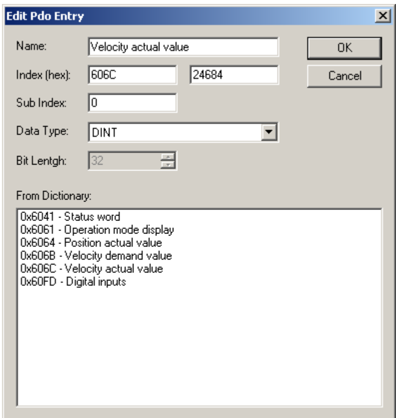
3.5 CiA® DSP-402 drive profile

By right-clicking in the ”PDO Content” window a menu with options appear, and if pressing ”Insert” then a new window will open showing the possible objects to insert in the PDO.



TT3080GB

By selecting an object and pressing “OK” then that object is inserted in the PDO and will be transferred to the MAC00-ECx module or the MIS/MILxxxxxxECxx motor, at next “reload devices” if the “PDO configuration” checkbox is checked.



TT3081GB

For further information about PDO configuration please consult the appropriate manual for the PLC system used.

3.5.9 Supported cycle times

The cycle time is the amount of time between each cyclic request (PDO) on the Ethernet. If the module is mounted in MAC050-MAC141 it is possible to add a poll division factor either in the EtherCAT tab in Mactalk or manually in module register 8 (See chapter 8 - *Register 8 - Poll division factor., page 258*).

	Motor series		
	MAC050-141	MAC400+	MIS/MILxxxxxxECxx
Supported cycle times with Distributed Clock	DC not supported	1 or 2 ms ****	1ms*, 2, 3, 4 ms
CiA402 profile minimum cycle time	16ms	1ms	1ms*
Applicable shift time for a master with max. ±20µs jitter on cyclic frames**	-	0, 10, 20, 30 ***, 40%	0, 10 (,20, 30, 40%)*

* If using MIS/MIL17/23 motor or using MIS/MIL34/43 motor with hardware version 1.6 or newer AND Ethernet hardware version 1.3 or newer ("Min. cycletime: 1ms" is showed in EtherCAT tab, moduleinfo frame in Mactalk).
See also *How to find FW/HW version at product, page 13*.

** If the master has larger jitter than listed the lowest and highest shift time value in the table must not be used.

*** 30% sync0 shift time will only work if cycle time is different from 2ms.

**** If 2ms cycle time is chosen, then it is also nesesary to chose 2ms motor cycle time in Mactalk or select FreeRun.

Refer to *Shift time., page 69* for changing the shift time.
If operating with values lower than listed then the motor will behave unpredictably.

WARNING: As seen in the table above the MAC050-141 don't support Distributed Clock and have a minimum cycle time of 16ms when using DSP-402. For these reasons the Cyclic operation modes normally used by masters are NOT recommended.

3.5.10 Factors

Position factor

The position factor is the relation between the user unit and the internal position unit (counts). The position factor is automatically calculated when the feed constant (Object 0x6092) and gear ratio (Object 0x6091) are set.
Example:
We have a MAC motor with a 3.5:1 gear box connected to a belt drive. The diameter of the drive wheel is 12.4 cm. We want the unit of position to be in millimetres.
The circumference of the drive wheel is 389.56mm (124mm*pi). The parameters should be set as follows:

Object	Name	Value
0x6091 sub index 1	Gear ratio / Motor revolutions	35
0x6091 sub index 2	Gear ratio / Shaft revolutions	10
0x6092 sub index 1	Feed constant / Feed	38956
0x6092 sub index 2	Feed constant / Shaft revolutions	100

Please note that it is not necessary to set the encoder resolution. This is automatically set by the module.

Position factor formula:

$$\text{Position\_factor} = \frac{\text{Gear\_ratio\_Motor\_rev.*Feed\_constant\_Shaft\_Rev.*Position\_encoder\_res.\_Encoder\_Increments}}{\text{Feed\_constant\_Feed*Feed\_constant\_Shaft\_rev.*Position\_encoder\_res.\_Motor\_rev.}}$$

or as objects:

$$\text{Position\_factor} = \frac{\text{Object 6091sub1*Object 6092sub2*Object 608Fsub1}}{\text{Object 6092sub1*Object 6092sub2*Object 608Fsub2}}$$

The Position factor is calculated to in the above example:

$$\text{Position\_factor} = \frac{35*100*4096}{38956*10*1} = 36{,}8$$

The above example is for a MAC50-141. For MAC400, MAC1500 and MAC4500, the number 4096 shall be changed to 8192, for MAC800 the number is 8000.

3.5.11 Operation modes

Changing operation mode
A change of operation mode is nearly always possible. Change between CSP, CSV and CST can be done in any time, but the user is responsible for delivering valid values for the used mode at all times.
It is only possible to change from homing mode to other modes when the homing procedure is finished.

Profile position mode
This mode can be used for positioning where a movement profile can be set up.
The acceleration and maximum velocity can be programmed.
In this mode, both absolute and relative moves are supported. The type of move is selected via bit 6 (abs/rel) in the status word. When a relative move is selected, the type of relative move is dependent on the setup in object 2011h sub index 6.
It is also possible to select different movement modes. This is done using bit 5 (change set immediately) in the status word. When this bit is 0 and a move is in progress, the new set-point is accepted. But the new set-point and profile are not activated before the previous movement is finished. When this bit is 1, the new set-point is activated instantly and the motor will move to the new position with the new profile parameters.

Please note:
- The torque limit that is used during the profile can be set via object 6072h.
- The register L1 (object 2012 subindex 81) is used to select the load factor when the profile is started. If a different load factor is required, this register must be set correctly.

Velocity mode
In this mode the motor runs at a selected velocity. A new velocity can be selected in object 0x60FF and the motor will then accelerate/decelerate to this velocity.
The maximum slippage error is not supported in this mode.

Please note:
- The torque limit that is used during the profile can be set via object 6072h.

Homing mode

In this mode different homing sequences can be initiated. The home sensor must be connected to the AIN input on the module. If end limit sensors are used during the homing sequence, then the sensors should be connected to the appropriate inputs, and they must be enabled via object 0x2011 sub index 11. In the MAC motors the module inputs is used.

In the MIS/MIL motors the registers 125 (I/O active level and I/O type), and 132 (home input mask) have to be correctly set up prior to use. Do this setup by object 0x2012 or in MacTalk in the 'I/O Setup' tab.

The torque limit used during homing is selected via object 0x2100. The unit of this object is the same as other torque objects, e.g. object 0x6072.

The MAC00-ECx module and the MIS/MILxxxxxxECxxxx supports the following homing methods:

Method	Description	Available in MAC	Available in MIS /MIL
-4	Torque homing in positive direction.	X	X
-3	Torque homing in negative direction.	X	X
-2	Torque homing in positive direction and afterwards homing on the index pulse.	X	-
-1	Torque homing in negative direction and afterwards homing on the index pulse.	X	-
0-2	Not supported.	-	-
3	Homing on positive home switch and index pulse to the left.	X	-
4	Homing on positive home switch and index pulse to the right.	X	-
5	Homing on negative home switch and index pulse to the left.	X	-
6	Homing on negative home switch and index pulse to the right.	X	-
7	Start positive (unless home switch is active), reverse on home switch active, stop at index.	X	-
8	Start positive (unless home switch is active), stop at first index after active home switch.	X	-
9	Start positive, reverse on limit switch, stop at first index after active home switch.	X	-
10	Start positive, reverse on limit switch, reverse at homeswitch, stop at index.	X	-
11	Start negative (unless home switch is active), reverse on home switch active, stop at index.	X	-
12	Start negative (unless home switch is active), stop at first index after active home switch.	X	-
13	Start negative, reverse on limit switch, stop at first index after active home switch.	X	-
14	Start negative, reverse on limit switch, reverse at home switch, stop at index.	X	-
15-18	Not supported.	-	-
20	Homing on positive home switch.	X	X
22	Homing on negative home switch.	X	X
24	Start positive (unless home switch is active), stop at active home switch.	X	-
26	Start positive, reverse on limit switch, stop at active home switch.	X	-
28	Start negative (unless home switch is active), stop at active home switch.	X	-
30	Start negative, reverse on limit switch, stop at active home switch.	X	-
31, 32	Not Supported	-	-
33	Start negative, stop at index	X	-
34	Start positive, stop at index	X	-
35	Current position = home position (obsolete)	X	X
37	Position actual = Home offset	X	X

For a comprehensive description of the homing modes 3-37, please consult the CiA DSP402 version 3.0.

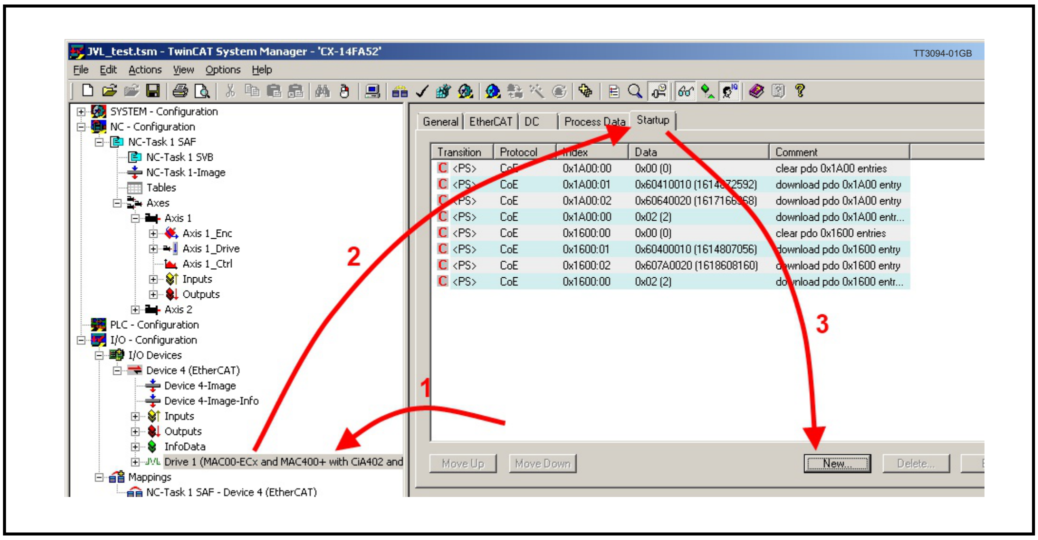
3.5 CiA® DSP-402 drive profile

Please note that you should always use a home offset (object 0x607C) when using torque homing. This is to ensure that the motor moves away from the end limit. The sign of the home offset should be the opposite of the homing direction. For example, when using a negative homing direction, the home offset could be 5000.

Cyclic Synchron Position mode (csp)
This mode is used when synchronization between several drives are needed in position mode. The default PDO addresses this mode. It is the preferred mode for the NC system in TwinCAT. When using CSP mode it is highly recommneded to use Distributed Clock, in order not to loose any cyclic frames.
If wanting to enable this mode as startup mode in TwinCAT then follow the steps below:

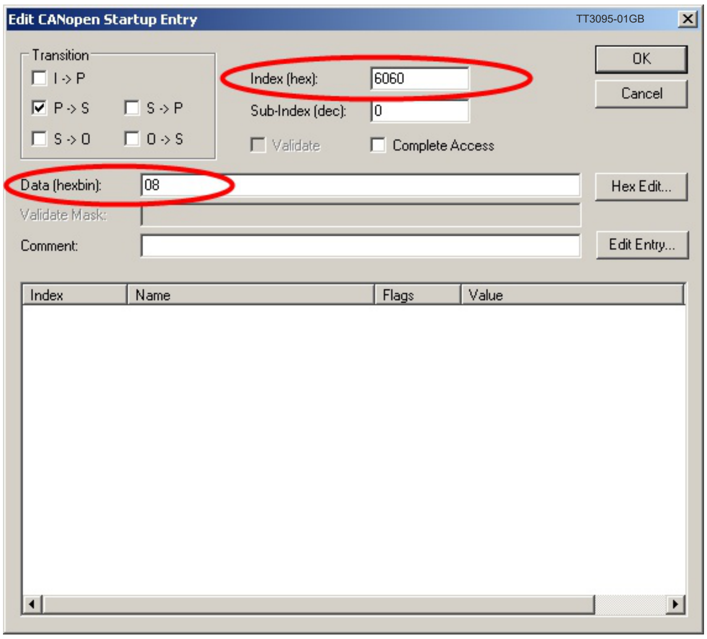
WARNING: The CSP mode is NOT recommended for MAC050-141, as these motors do not support Distributed Clock, and have a minimum cycle time of 16ms when using DSP-402.

Step 1-3.
Select the drive and press the “Startup” tab, then press the “New” button, as shown in the below picture.



3.5 CiA® DSP-402 drive profile

Step 4.
In the ”Edit CANopen Startup Entry” window is then inserted the object no. “6060” as the “Index” value and the value “08” as the “Data” value as shown below.



Note ! Changes will only become effective after reconfiguring and restarting the Ether-CAT master!

Please note:

- The torque limit that is used during the mode can be set via object 6072h before-hand.
- The motor register L1 (object 2012 subindex 81) is used to select the load factor when the mode is started. If a different load factor is required, this register must be set correctly.

3.5 CiA® DSP-402 drive profile

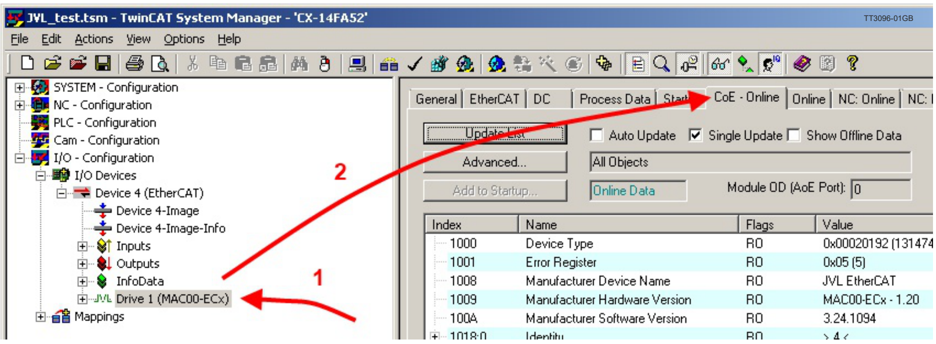
Cyclic Synchron Velocity mode (csv)

This mode is used when synchronization between several drives are needed in velocity mode. When using CSV mode it is highly recommneded to use Distributed Clock, in order not to loose any cyclic frames. To use this mode the default PDO needs to be changed. Please follow the steps below:

WARNING: The CSV mode is NOT recommended for MAC050-141, as these motors do not support Distributed Clock, and have a minimum cycle time of 16ms when using DSP-402.

Step 1-2.

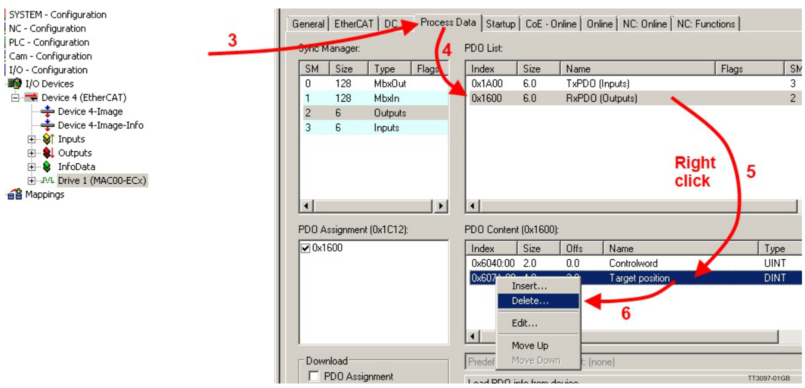
It is easiest to change the PDO in TwinCAT if the drive is connected to TwinCAT and is "online". Then first press the "CoE online" tab in the drive setup. Please see below picture.



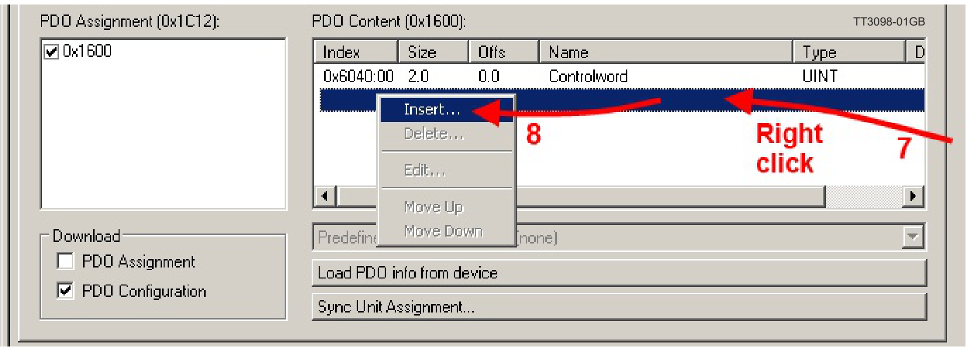
This way the available objects are fetched online from the drive, and don't have to be keyed in manually.

Step 3-6.

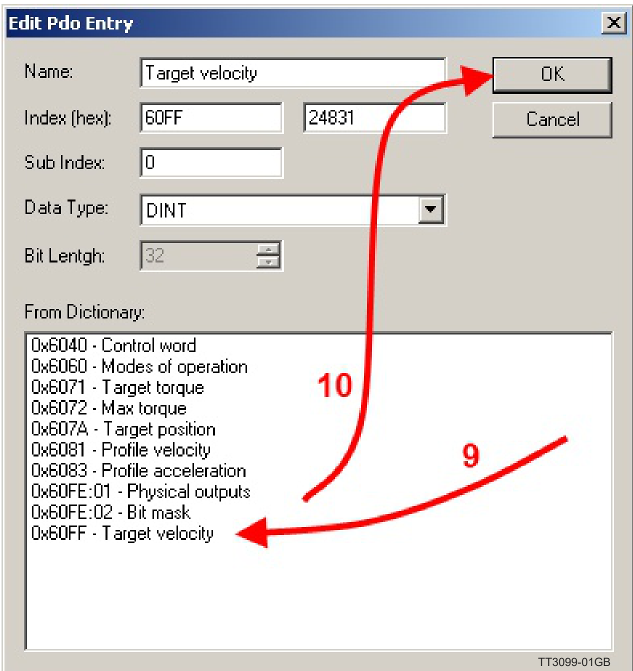
Then press the "Process Data" tab, select "RxPDO" and then right click on Index 0x607A and select "Delete". Answer yes to the confirmation. See steps in the picture below.



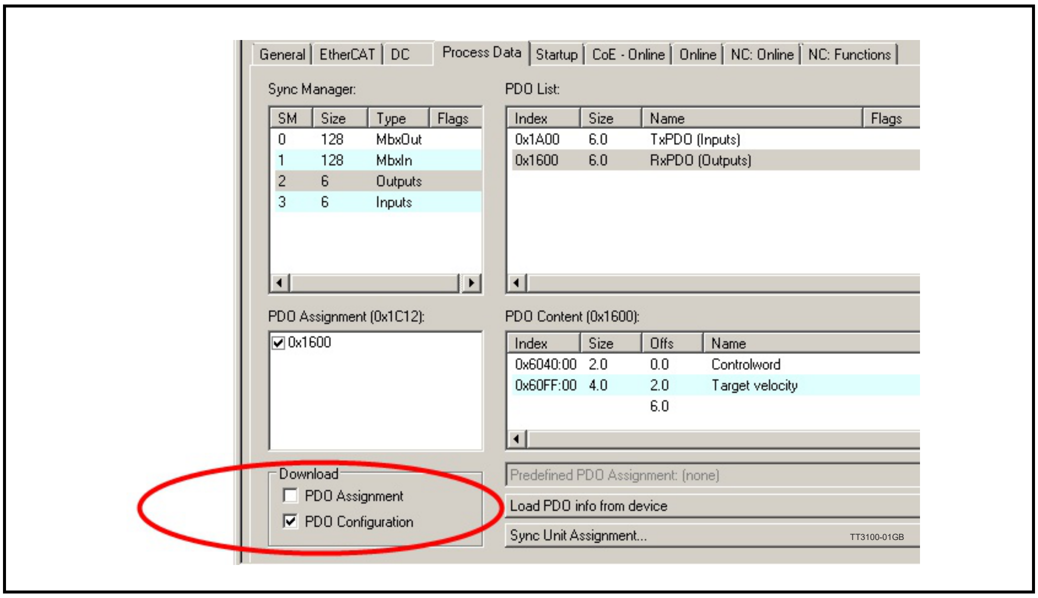
Step 7-8.
Then right click in the bottom of the “PDO Content” and select “Insert”, as shown below.



Step 9-10.
Choose object 0x60FF from the list and press OK



Step 11.
Make sure the "PDO assignment" is unchecked and "PDO Configuration" is cheked. Please see below.



If wanting to enable this mode as startup mode in TwinCAT then please see the procedure in the previous paragraph "Cyclic synchron position mode", and just change the "Data value to "09", instead of "08".

Note ! Changes will only become effective after reconfiguring and restarting the EtherCAT master !

Please note:
The torque limit that is used during the mode can be set via object 6072h beforehand. The register L1 (object 2012 subindex 81) is used to select the load factor when the mode is started. If a different load factor is required, this register must be set correctly.

Cyclic Synchron Torque mode (cst)
This mode is used when synchronization between several drives are needed in torque mode. When using CST mode it is highly recommneded to use Distributed Clock, in order not to loose any cyclic frames. To use this mode the default PDO needs to be changed. Please follow the steps 1-11 from above in "Cyclic Synchron Velocity mode" where the inserted object should be 0x6071 (target torque) instead of object 0x60FF. And then follow the steps 1-4 from "Cyclic synchron position mode", and just change the "Data value to "0A", instead of "08", if wanting the motor to start up in CST mode.

Notes ! Changes will only become effective after reconfiguring and restarting the EtherCAT master !
The register L1 (object 2012 subindex 81) is used to select the load factor when the mode is started. If a different load factor is required, this register must be set correctly.

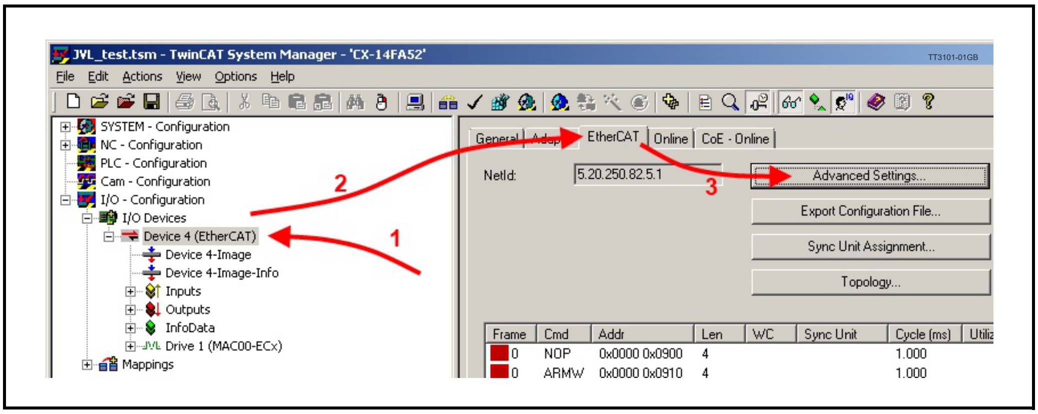
3.5.12 Shift time.

The shift time is the nominal time the cyclic EtherCAT frames are sent before the sync0 pulse is activated. At normal circumstances this setup should not be changed, as it will affect all the devices in the network.

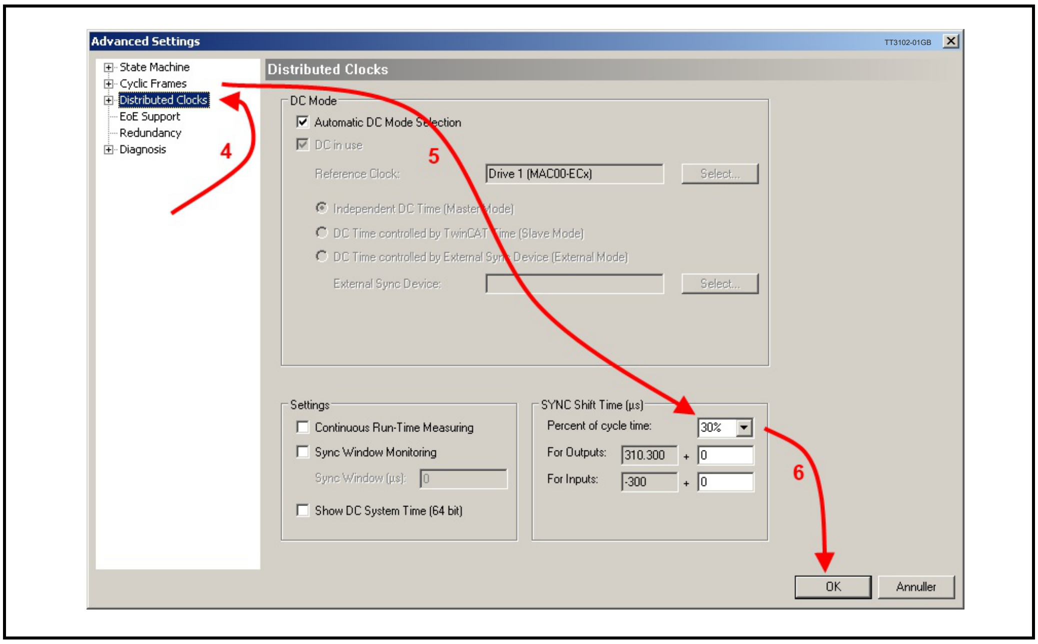
The default in TwinCAT is 30%, meaning that the cyclic EtherCAT frames are sent 30% of the cycle time before the sync0 pulse activates. If for example the cycle time is 1ms then the frames are sent 300μs before the sync0. This is of course nominal and will vary a lot because of timing issues in the EtherCAT master. It is also possible to add a device specific shift time, but then the sync0 pulse of the devices on the network will not be activated simultaneously, unless the same shift time is added to all devices.

Changing general shift time.
As written above changing this setting will affect all devices on the network. So proceed with care !

Step 1-3.
Select the EtherCAT device, select the "EtherCAT" tab and press the "Advanced Settings" button.



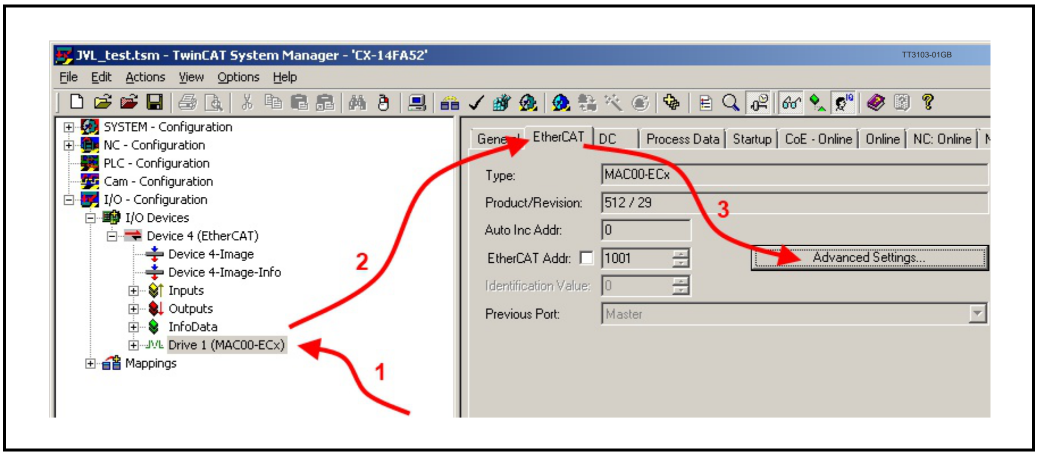
Step 4-6.
Select "Distributed Clocks", change the "Percent of cycle time" to the needed setting (10% - 40%) and press the "OK" button.



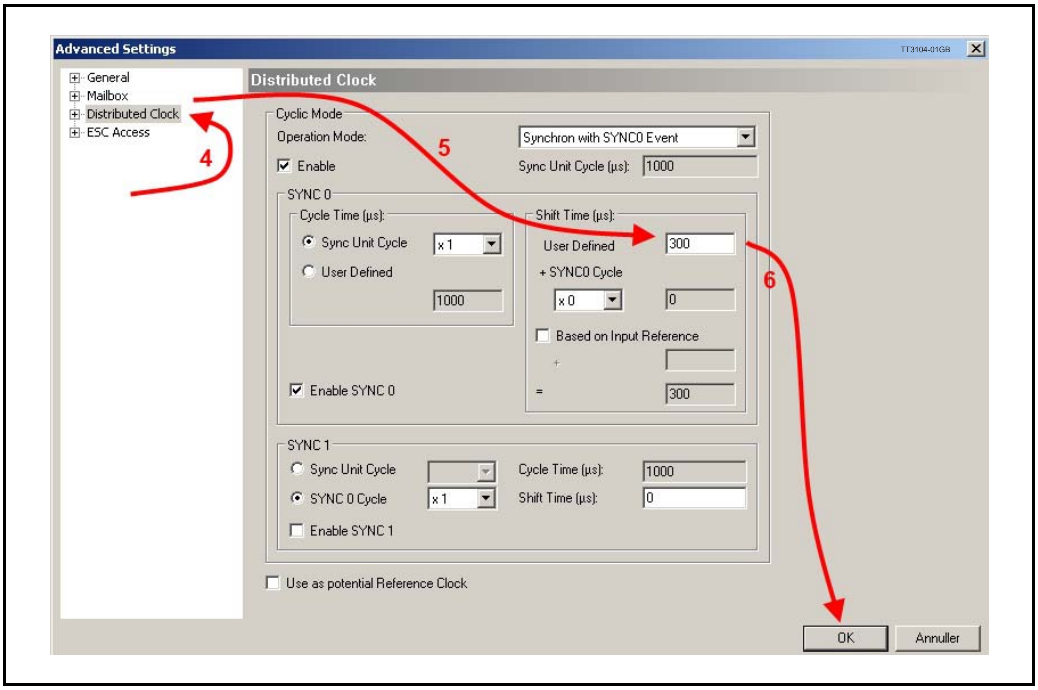
3.5 CiA® DSP-402 drive profile

Adding device specific shift time.
Device specific shift time will delay the sync0 pulse on the specific device. Be aware that if only changing this setting on some devices then the sync0 pulse will **not** appear simultaneously on all devices.

Step 1-3.
Select the drive, then select the “EtherCAT” tab and press the “Advanced Settings” button.



Step 4-6.
Select “Distributed Clocks”, change the “Shift time” “User defined” to the needed setting and press the “OK” button.



3.5.13 AL status codes.

When the JVL motors are set in OP mode for the first time after power-up, with "DC Synchron with Sync0 Event" selected, then the JVL motor can return a status code to the AL register.

AL Status code	Meaning	Course	Action required
0x001A	Synchronization error	The Device is not synchronized	-
0x002C	Syncronization pulses are missing from master	The syncronization pulses (SYNC0) don't appear.	Make sure the syncronization pulses (SYNC0) are correctly setup in the master.
0x001D	Invalid output configuration	-	Change the output PDO to something valid.
0x001E	Invalid input configuration	-	Change the input PDO to something valid.
0x0034	Too many "syncronization master" events missed	The cyclic dataframes from the master did not arrive in time.	Select a slower cycle time or a faster master.
0x8003	DC_CFG is invalid	An unsupported cycle time is used.	Select a supported cycle time.

3.5.14 Limit switches

This paragraph refers to the functioning of limit switches if using CiA402 and the MAC00-ECx module in MACxxx motors. For setup of limit switches if using MIS/MILxxx motors, please refer to the specific MIS/MIL motor manual.
Limit switches can be used during homing and during normal operation in any modes. In normal operation modes the activation of a limit switch causes an error if enabled. In homing mode an activation does not cause an error, but are used to change direction during search if using a homing mode supporting this. The limit switches are disabled as defualt but can be enbled in the object 0x2011 subindex 11 (please refer to paragraph 8.2.11). The setup can be saved to flash, otherwise it has to be sent to the EtherCAT module after every power cycle, in order to work. For saving of EtherCAT module parameters to flash please refer to paragraph 3.6.6.

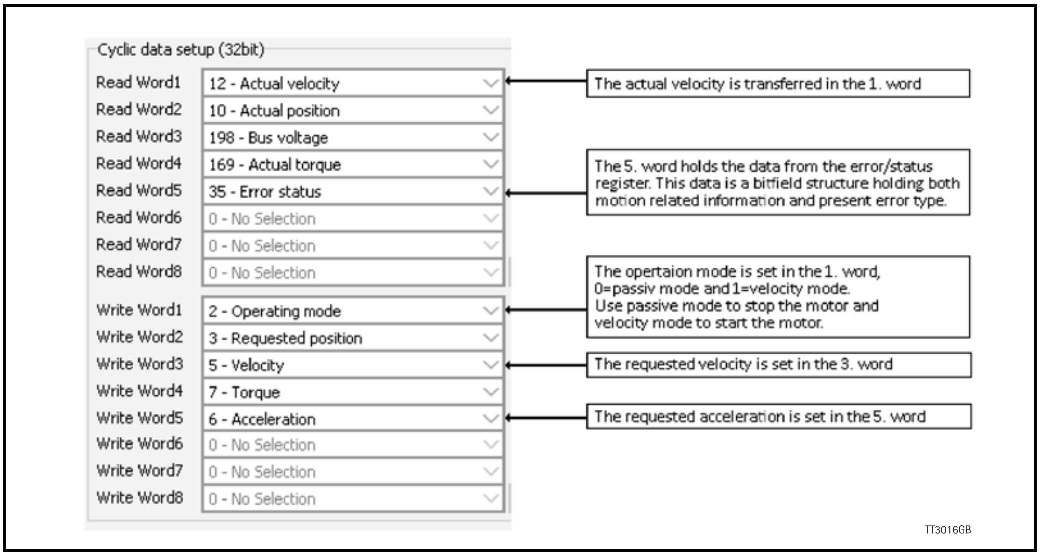
3.6 Examples

3.6.1 Running Velocity control (JVL Profile)

To use the JVL motor in velocity mode the following registers are basically of interest.

1. "Mode" - Mode register register 2
2. "V_SOLL" - Velocity register 5
3. "A_SOLL" - Acceleration register 6
4. "Error/Status" - Error and status register 35

So, to control these registers the cyclic data needs to configured.
From MacTalk the setup is configured as this.



With the settings illustrated above we initiate the velocity mode by writing 0x1 to the first word-value, this is velocity mode.

From the Master the registers is accessed using the PDO's and accessing the registers R/W on words 1-8.

Since different PLC's have different methods of implementation the basic steps is described in the following. (Constant values valid for MAC800, for other motors, please consult the motor manual)

1. Set the needed velocity. V_SOLL = V / 2.77 [rpm]
   Ex. We need the motor to run with a constant speed of 1200 RPM. So, V_SOLL = 1200/2,77 = 433 cnt/smp
2. Set the needed acceleration. A_SOLL = A / 271 [RPM/s^2]
   Ex. We need the motor to accelerate with 100000 RPM/s^2 so, A_SOLL = 100000/271 = 369 cnt/smp^2.
3. Now set the motor into velocity mode and thereby activate the motor.
   Ex. The motor needs to be activated by setting it into velocity mode, so we need to set the mode register to the value 1. Mode = 1 which is velocity mode, now the motor will use the acceleration and the velocity just configured.

Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 and *Motor registers MAC400 - 4500, page 301* and *Motor registers MISxxx, page 320*

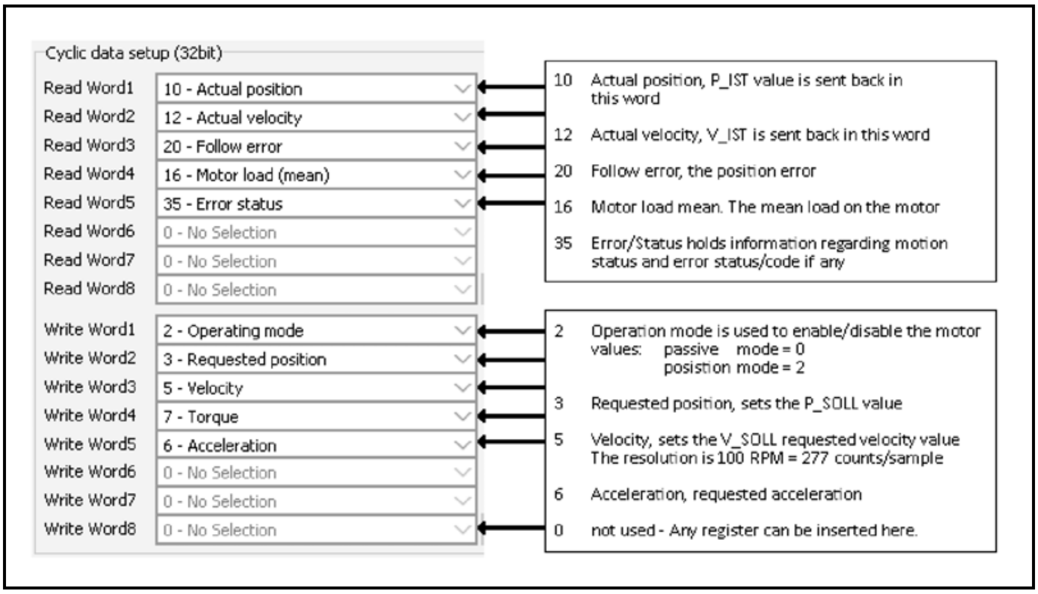
3.6 Examples

3.6.2 Running Position control (JVL profile)

Running the motor in position control requires that the mode register is set for position control. The following registers is of particular interest when position mode is used.

1. "Actual position" -P_IST, register 10
2. "Actual velocity" -V_IST, register 12
3. "Follow error" - The actual position error, register 20
4. "Motor load mean" - average motor load, register 16
5. "Error/Status" -register 35
6. "Requested position" -P_SOLL, register 3
7. "Requested velocity" -V_SOLL, register 5
8. "Requested acceleration" -A_SOLL, register 6

In this mode the position is controlled by applying a requested position to the "P_SOLL" -register and the actual position is monitored in the "P_IST" register. The V_SOLL and A_SOLL registers sets the velocity and acceleration used when positioning occurs.



3.6.3 General considerations

The register 35 in the motor holds information on the actual error/status. So it is crucial that this register is configured in the cyclic data and thereby obtained and monitored in the Master. In case of an error situation the motor will stop and the cause will be present in the register 35 and hence in the I/O -data.
This register also holds information on the motion status such as:
- In position, bit 4
- Accelerating, bit 5
- Decelerating, bit 6

Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 and *Motor registers MAC400 - 4500, page 301*

The JVL motor is basically put into a working mode and into a passive mode where the motor axle is de-energized, by setting register 2 into either 0 = "passive mode" or into one of the supported modes.
(continued next page)

Example.

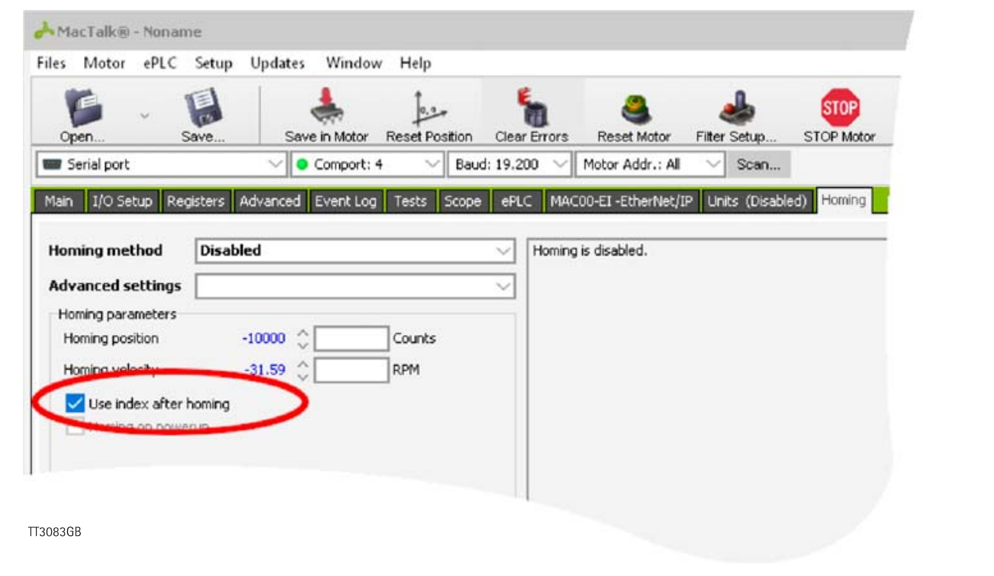
1= "Velocity mode" / 2= "Position mode" / etc.

So in order to Stop or Start the motor this register can be supported in the I/O data or by sending an SDO message.

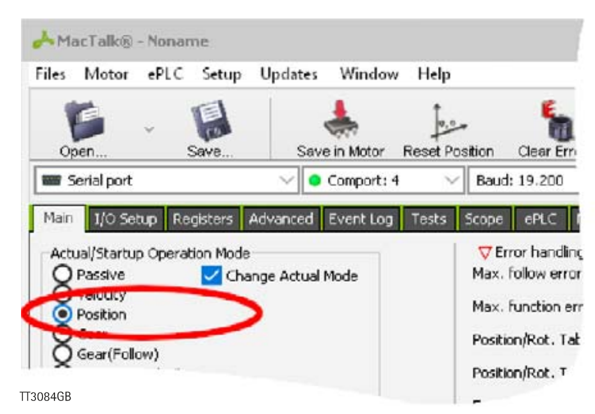
3.6.4 Homing using only cyclic I/O (JVL profile).

When doing a homing (Zero search), with only cyclic I/O, some preconditions have to be met:

Zero search position, zero search velocity and zero search torque (torque only for MAC motors) has to be set in MacTalk in the "Main" tab, and saved in flash in the motor once and for all.

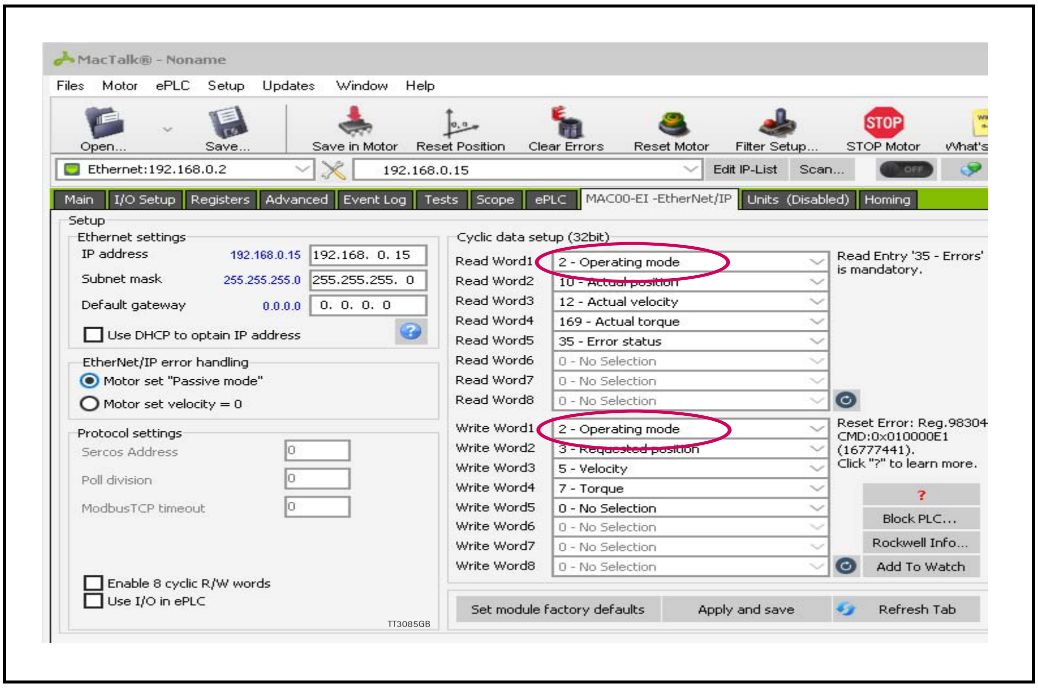


Startup mode should be set to position, for the motor to stay in position after the homing sequence. And this setting should also be saved in flash.



3.6 Examples

Register 2 (Operating mode) has to be present in BOTH the cyclic read words and cyclic write words.



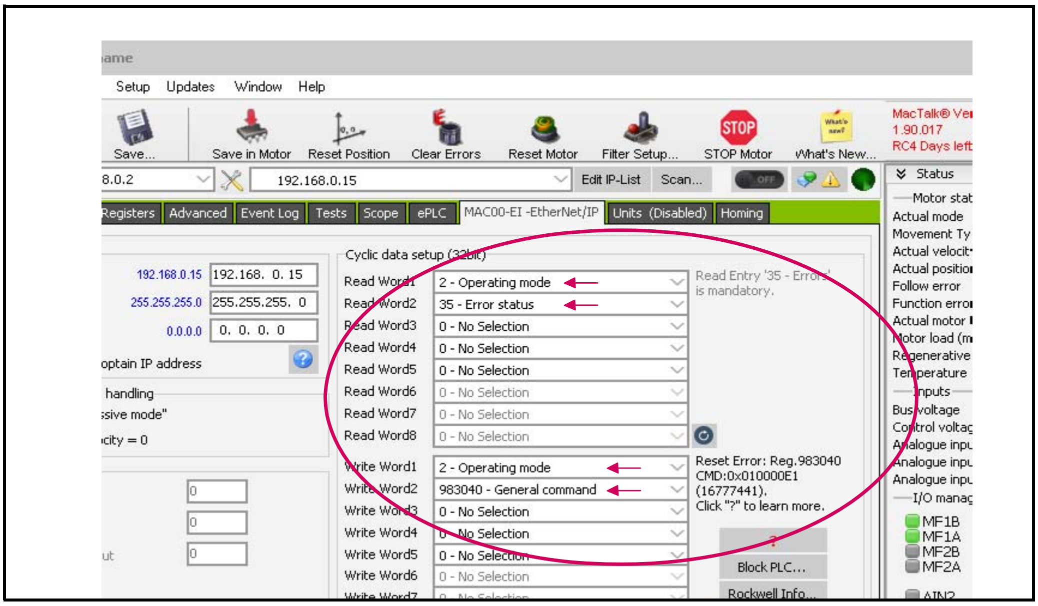
Procedure in the PLC:

- Treat the transmitted Register 2 as "Requested_Mode" and the received register 2 as "Actual_Mode".
- When homing is wanted, set the "Requested_Mode" to one of the values 12, 13 or 14, 25 or 26 depending of the requested homing mode (12 = Torque based zero search mode (only MAC motors). 13 = Forward/only zero search mode. 14 = Forward+backward zero search mode (only MAC motors). 25 = Enc. index (only MAC400+). 26 = Enc. quick index (only MAC400+).). For a comprehensive description of the homing modes, refer to the general MAC motor manual - LB0047-xxGB.
- Observe that the "Actual_Mode" is changing to the homing mode. Now the module is blocking cyclic writes TO the motor. Cyclic reads is still active.
- Wait for register 35 "Error status" bit 4 to be active =IN_POSITION. (Indicates that homing is finished).
- Then change "Requested_Mode" to whatever needed. The blocking of cyclic writes to the motor is then released by the module.

3.6.5 Relative positioning (JVL profile).

There are a number of ways to do relative positioning, but the one explained here is very simple, and can be used with a constant distance, or exchangeable distance, to move every time it is requested.

Preconditions:
Place the module command register (register 983040 in MacTalk) in the cyclic write list. The cyclic setup, could for example look like this:



Procedure in the PLC:

1. Set up register P7 in motor to requested relative offset.
2. Make sure one net cycle has passed, so P7 resides in the motor.
3. Issue command 0x010000F1 (0x01000071 if the device is a MIS/MILxxx) in module command register (register 983040 in MacTalk).
4. Make sure one net cycle has passed, so command is interpreted by the motor.
5. Set module command register to zero. This will prepare the Ethernet module for new commands.
6. If needed then monitor register 35 (Error status): When bit 4 is set (in position), then the move is finished.
7. When a new relative move is requested, go to step 3.

You may also have the P7 register in the cyclic write list, thereby enabling easy change of the relative distance to move.

3.6.6 Save parameters to flash (CiA402 + JVL Profile)

Saving of the parameters to flash (non-volatile memory) only requires a simple non-cyclic command to the EtherCAT module command register which is accessible via object 0x2010.

Save EtherCAT module parameters to flash:
Write the value 0x0000 00010 (= 16 dec.) to object 0x2010.

Save motor parameters to flash:
Write the value 0x0000 00110 (= 272 dec.) to object 0x2010.

4 EthernetIP Users Guide

4.1 Introduction to EthernetIP

MAC Ethernet/IP Module
Type:
MAC00-**EI**4 (shown) or
MAC00-**EI**41 (extended I/O)

To be used in following servo products:
MAC50, 095, 140 and 141
MAC400 and MAC402
MAC800
MAC1500 and MAC3000

MIS Ethernet/IP motors.
Type:
MISxyyzz**EI**xx85 or
MILxyyzz**EI**xx85

TT3042-02GB

4.1.1 Introduction to EtherNet/IP

The JVL MAC00-EIx -module or MIS/MILxxxxxxEIxxxx, makes communication using EtherNet/IP possible with the JVL motor. The Ethernet technology gives the advantages of fast data access using standard off the shelf hardware which again has the advantage of large accessibility and low prices.

The JVL implementation is done in a way that minimizes the complexity of getting a system up and running but still utilizes the benefits of industrial ethernet.
The JVL EtherNet/IP implementation supports both explicit messaging and I/O messages with up to 8 free configurable input and output words.

With a basic knowledge of the JVL motor operation through the register structure and a basic knowledge of the EtherNet/IP technology, a motor can be setup and controlled in a very short time without first doing extensive studies in complex motion control standards e.t.c.

EtherNet/IP is basically divided in 2 groups of data, explicit and I/O messages in other words messages requiring fast data response time and data not so time critical typically used for configuration purposes. In the EtherNet/IP terminology these messages are also called Explicit messages (not time critical, none cyclic exchanged) and I/O messages (time critical, cyclic exchanged).

In the motion control world, time critical data would be actual position, actual status and actual speed and actual torque where data not time critical would be such as motor temperature and setup parameters.

(continued next page).

4.1 Introduction to EthernetIP

EtherNet/IP is object based similar to DeviceNet and follows the standards issued by ODVA.
For more information on EtherNet/IP please visit www.ODVA.org for further details on EtherNet/IP and to get the EtherNet/IP standard specification issued by ODVA.

The JVL implementation supports manufacture specific objects to gain access to each register in the motor.

This manual assumes that the servomotor user manual has been read and a base knowledge using the servomotor and the configuration software MacTalk is acquired.

The examples and screen shots in this manual are taken from MacTalk and a Rockwell RSLogix5000 application.

Please be aware that other PLC vendors than Rockwell exist.

4.1.2 Abbreviations

The below general used terms are useful to know before reading the following chapters.

100Base -Tx	100 MBit Ethernet on twisted pairs.
IP	Internet Protocol - IP address ~ the logical address of the device which is user configurable.
MAC	Media Access Controller - MAC address ~ the hardware address of the device.
MacTalk	A windows PC based program supplied from JVL. This is an overall program to install, adjust and monitor the function of the motor and a module installed in the motor.
TCP	Transfer Control Protocol (an IP based protocol used widely on the internet)
UDP	User Datagram (an IP based protocol used widely on the internet)
DHCP	Dynamic Host Configuration Protocol (Automatic configuration of IP address netmask and gateway from a DHCP server).

4.1.3 Daisy chaining

Up to 64 units (nodes) can be daisy chained. By daisy chained means making a direct cable from the master in the system to motor 1 at the "L/A IN" connector at MAC motors and the "CN2" connector at the MIS/MIL motors.
Continue from motor 1 "L/A OUT" (MAC) or "CN3" (MIS/MIL) to motor 2 “L/A IN”/ CN3 etc.
This method is saving hardware since no switch(es) is needed and can often be the simplest way of doing the wiring.
The disadvantage is that the data will be delayed slightly depending on how many motors that are daisy chained and the network load will be significant if a larger number of motors is connected this way.

Another and more common solution is to use a switch after the master and then distribute data to each node from this switch. This solution has a minimal delay of the data stream.

4.1.4 EthernetIP specification

The JVL implementation supports standard objects as well as manufacturer specific objects to gain access to each register in the motor and in the module.

Supported standard EthernetIP classes

Type	Class
Identity Object, class	0x01
Message router object, class	0x02
Assembly object, class	0x04
TCP/IP interface object, class	0xF5
Ethernet link object, class	0xF6

On top of this the JVL manufacturer specific class objects has been added.
Supporting manufacturer specific classes

Type	Class
Motor registers	0x64
Module registers	0x65

Identity object class 0x01
Holds information about the JVL device on the network. Typical used by other devices to identify devices on the network.
(for further specification please refer to the EtherNet/IP approximately.)

Message router object class 0x02
Handles all messages to/from object's in the device.

Assembly object class 0x04
Object that binds all IO data to a connection point.

TCP/IP interface object class 0xF5
Holds all information on the Ethernet connection, such as the IP-address, Network mask and GateWay.

Ethernet link object class 0xF6
Holds information on link specific counters and instances associated with the communication interface.

Motor registers object class 0x64
Access to registers1 to 255 in the Motor.

Module registers object class 0x65
Access to all registers in the Module.

Extended motor registers object class 0x66
Access to motor register 255-511.

4.2 Using none cyclic messages

None cyclic messages in the EtherNet/IP domain is called Explicit messages.
This message type is typically used to perform configuration and other none-time critical operations.

Explicit messages can be send as a connected or unconnected message.

All registers in the motor and in the EthernetIP module can be accessed explicitly using object classes 0x64 and 0x65 respectively. Please see the paragraphs *Vendor specific JVL object class 0x64, page 85* and *Vendor specific JVL object class 0x65, page 87*

4.2.1 Type definitions:

UINT 16bit
DINT 32bit
STR String of ASCII-chars

4.2.2 Identity object class 0x01

Holds data on different module specific data.
Instance = 1

Attr. ID	Access	Name	Data type	Description
1	R	Vendor ID	UINT	JVL vendor ID = 936 (0x3A8)
2	R	Device Type	UINT	Value=10
3	R	Product code	UINT	Value = 1
4	R	Revision	UINT	Major = 1.byte, minor = 2. byte
5	R	Status	UINT	Status
6	R	Serial number	DINT	Serial number
7	R	Product name	STR	“MAC00-EIx”

See the EtherNet/IP spec. for further details section Vol2 sect.5-3.

Supported Services
0x1 Get_Attribute_All
0x10 Set_Attribute_Single
0xE Get_Attribute_Single

4.2 Using none cyclic messages

4.2.3 Assembly object class 0x04

Holds pre-configured motor registers to be accessed.

Instances:
0x64 Write Data to motor register.
0x65 Read motor register data.

Attr. ID	Access	Name	Data type	Description
3	R/W	Get/Set Assembly	20 bytes	Get/Set all assembly data
4	R	Bytes	UINT	Bytes transferred in assembly

Supported Services
0x10 Set_Attribute_Single
0xE Get_Attribute_Single

This object can be used to access the predefined registers, configured from MacTalk. They are also accessed when using the implicit connection cyclically.
If other registers than the one defined in the assembly object needs to be accessed then the class 0x64 needs to be used. This class accesses each register in the motor for a more dynamically way of controlling registers explicitly.
The vendor specific class 0x64 is specified in details later in this chapter.

4.2.4 TCP/IP object class 0xF5

Holds data on different module specific data.

Attr. ID	Access	Name	Data type	Description
1	0xE	Status	DINT	Status bit-field
2	0xE	Configuration capability	DINT	DINTbit field = 5 (BOOTP+DHCP)
3	0x10	Configuration control	DINT	Bit field = 0 (use NV-setup)
4	0xE	Physical link object	6 bytes	Size + path
5	0x10	TCP/IP settings	22bytes	IP + sub net + GTW info e.t.c.
6	0x10	Host name	DINT	Host name

See the EtherNet/IP spec. for further details section Vol2 sect.5-3.

Supported Services
0x1 Get_Attribute_All
0x10 Set_Attribute_Single
0xE Get_Attribute_Single

To change the IP address, Subnet mask or gateway. The object 0xF5, attr 5 is used. The data format consists of 22 bytes.

Byte 0 - 3 : IP Address, exc. 192.168.0.58 = 0x3A 0x0 0xA8 0xC0
Byte 4 - 7 : Subnet mask, exc. 255.255.255.0 = 0x0 0xFF 0xFF 0xFF
Byte 8 - 11 : Gateway, exc. 192.168.1.1 = 0x1 0x1 0xA8 0xC0
Byte 12 - 21 : Not used, must be set to 0x0

These settings can be read from the motor using the service 0xE, Get attribute single and the motor will return the 22 bytes of the current setting.
Changing the settings can be done by using the service 0x10, set attribute single.

4.2 Using none cyclic messages

4.2.5 TCP/IP object class 0xF6

Holds information for a IEEE 802.3 communication interface

Attr. ID	Access	Name	Data type	Description
1	0xE	Interface speed	DINT	Speed in Mbit/s
2	0xE	Interface status	DINT	Bit field
3	0xE	MAC-address	6 bytes	MAC
4	--	Not Implemented	--	--
5	--	Not Implemented	--	--
6	0x10	Interface Control	DINT	Bit field

See EtherNet/IP spec. for further details Vol2 sect. 5-4

Supported Services
0x1 Get_Attribute_All
0x10 Set_Attribute_Single
0xE Get_Attribute_Single

4.2.6 Vendor specific JVL object class 0x64

Holds pre-configured motor registers to be accessed.

Instances
1 - 255 Motor registers

Attr. ID	Access	Name	Data type	Description
1	0xE / 0x10	Get/Set register	DINT	Get/Set the specified motor register

Supported Services
0x10 Set_Attribute_Single
0xE Get_Attribute_Single

Please notice: Please find a complete list of register descriptions in the appendex. *Motor registers MAC050 - 141, page 292* and *Motor registers MAC400 - 4500, page 301* and *Motor registers MISxxx, page 320*

4.2 Using none cyclic messages

E.g. the motor shall be operated in velocity mode.
This requires that the mode register 2 = 0x1.
Velocity mode is 0x1, Position mode = 0x2 e.t.c.
All modes of operation is further described in the servo manual.
The explicit message is setup as follows.

Package:
Class: 0x64
Service: 0x10 (write data)
Instance: 0x2 (mode register in the motor)
Attribute: 0x1

Data: **0x00 00 00 01**

This will set the mode register in the motor into velocity -mode
Motor Register 2 = 1

To read a value from the motor use the service code 0xE.

After setting the motor into velocity mode it will start running. Now the actual velocity can be read while the motor is running.

Package:
Class: 0x64
Service: 0xE (Read data)
Instance: 0xC (Actual velocity)
Attribute: 0x1

Now the response data is received:

Data: **0x00 00 01 15**

This value 0x115 (hex) is the decimal value 277 which corresponds to 100 RPM. This is the default velocity value.

So basically the motor can be controlled and all needed data can be retrieved using explicit messages. This method is not suitable when data is needed fast and frequently for this purpose I/O messaging (Implicit messaging) is used.

Not only motor registers are accessible using explicit messages, also static data such as serial numbers, network status etc. are accessible. These informations are accessible according to the EtherNet/IP standard and follows the implemented classes: **0x1, 0x4, 0xF5, 0xF6.** These classes are explained in details in the EtherNet/IP standard (obtained from www.ODVA.org) and in

For further info please *See "Examples" on page 106.*

4.2.7 Vendor specific JVL object class 0x65

Holds pre-configured EthernetIP Module registers.

Instances

1 - 63 EthernetIP module registers.
Please see chapter 8 for a complete list with register descriptions.

Attr. ID	Access	Name	Data type	Description
1	0xE / 0x10	Get/Set register	DINT	Get/Set the specified motor register

Supported Services

0x10 Set_Attribute_Single
0xE Get_Attribute_Single

Example: The digital outputs need to be set.

Package:
Class 0x65 (Access module registers)
Service 0x10 (Write data)
Instance 0x07 (Digital outputs register in the module)
Attribute 0x1
Data 0x00 0x00 0x00 0x01 (Set the O1 output)

This will set the O1 output in the EthernetIP module.

Example: Read of digital inputs.

Package:
Class 0x65 (Access module registers)
Service 0x0E (Read data)
Instance 0x47 (Digital inputs register in the module)
Attribute 0x1
Data 0x00 0x00 0x00 0x03

The value 0x03 corresponds to IN1 and IN2 set. (The IN2-IN4 is only available in the MAC00-Ex41 modules).

This method is not suitable when data is needed very fast and frequently. For this purpose I/O messaging (Implicit messaging) should be used.

4.2.8 Vendor specific JVL object class 0x66

Holds pre-configured extended motor registers to be accessed.

Instances

1 - 255 accesses motor registers 256-511. This means instance 1 accesses motor register 256, instance 2 accesses motor register 257 and so forth.

For further information about the use, please refer to *Vendor specific JVL object class 0x64, page 85*

4.3 Using cyclic I/O-messages

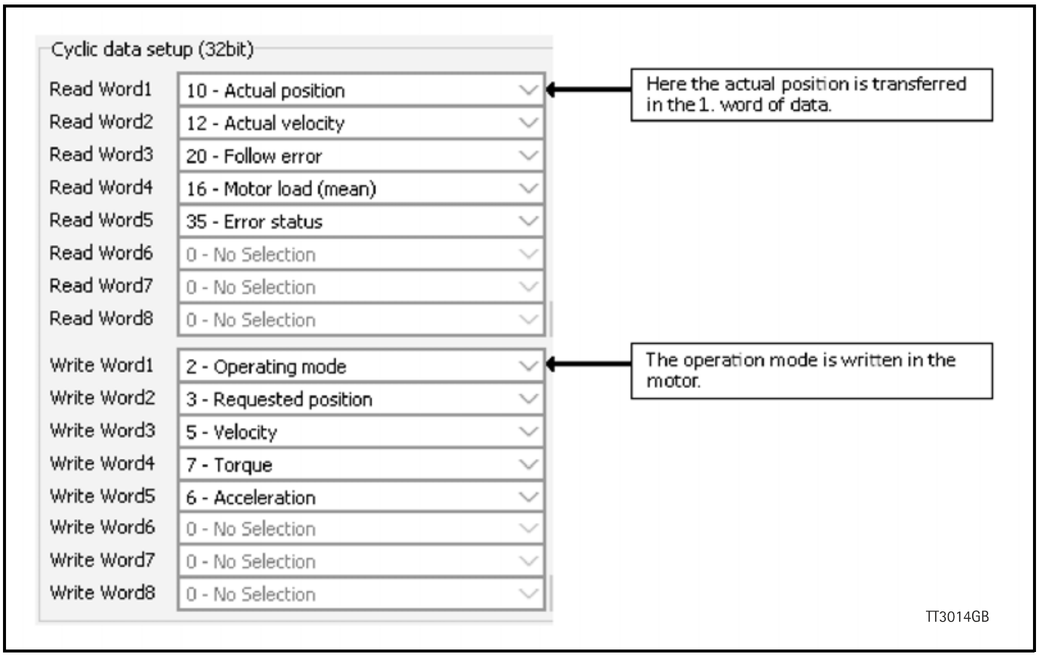
4.3.1 Cyclic messages.

I/O messaging also referred to as Implicit messages is used when data is needed fast and frequent. That is fast dynamic changing data such as position, velocity, torque etc.
It is mandatory to have the error/status register (register 35) as one of the slave to master registers. If not the motor will overrule the configuration and place register 35 anyway.
These data is sent cyclic using the assembly class object 0x04.
If module registers is placed in cyclic R/W, then the register number has to be calculated as follows:

Register number = 65536 x sub index.
Example: module command (sub-index 15) = 65536 x 15 = register **983040**

When module registers (register numbers above 65535) are chosen, they **have** to be placed **after** the motor registers in the list of cyclic registers.
The JVL assembly consists of 8I/O words that are freely configurable. This means that 8 input motor registers can be selected and another 8 motor registers for output purposes. The terms Input and output is considered from the scanner so input is data flowing from the motor to the scanner and output is vice versa.
On the EthernetIP -tab in MacTalk these I/O's are configured.

NB! If an index is set to zero (No selection), then the following indexes is discarded. Thereby computing resources in the drive are released, which makes much faster cycle times possibly. Please see next paragraph.



All words are 4 bytes.
In the example shown above the 5 read words (data read from the motor) are:

Motor register 10 (Actual position)	The actual motor position
Motor register 12 (Actual velocity)	The actual velocity of the motor
Motor register 20 (Follow error)	The actual follow error in the motor movement
Motor register 16 (Motor load - mean)	The load the motor is experiencing over time
Motor register 35 (Error status)	Bit-field that holds both error information and status of movements etc.

4.3 Using cyclic I/O-messages

The motor registers 35, 36, and 211 should NOT be inserted in the cyclic write list, as this may give unpredictable results. For clear of errors, reset of motor etc. please insert the module command register (=983040 in Mactalk) in the cyclic write list and send commands this way. For a list of commands for the module command register please *Register list., page 254*.

The 5 write registers are configured to hold the following data:

Motor register 2 (Operating mode)	0=passive, 1=Velocity, 2=position etc
Motor register 6 (Acceleration)	The requested acceleration to be used.
Motor register 5 (Velocity)	The requested Velocity to be used.
Motor register 7 (Torque)	The max. allowed Torque to be used
Motor register 3 (Requested position)	The requested position if operating mode = 2 (position)

Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 and *Motor registers MAC400 - 4500, page 301* and *Motor registers MISxxx, page 320*

Please notice: Even though all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

4.3.2 Minimum cycle time

The minimum cycle time is the minimum amount of time between each cyclic request on the Ethernet. If operating with values lower than those listed, data loss will occur.

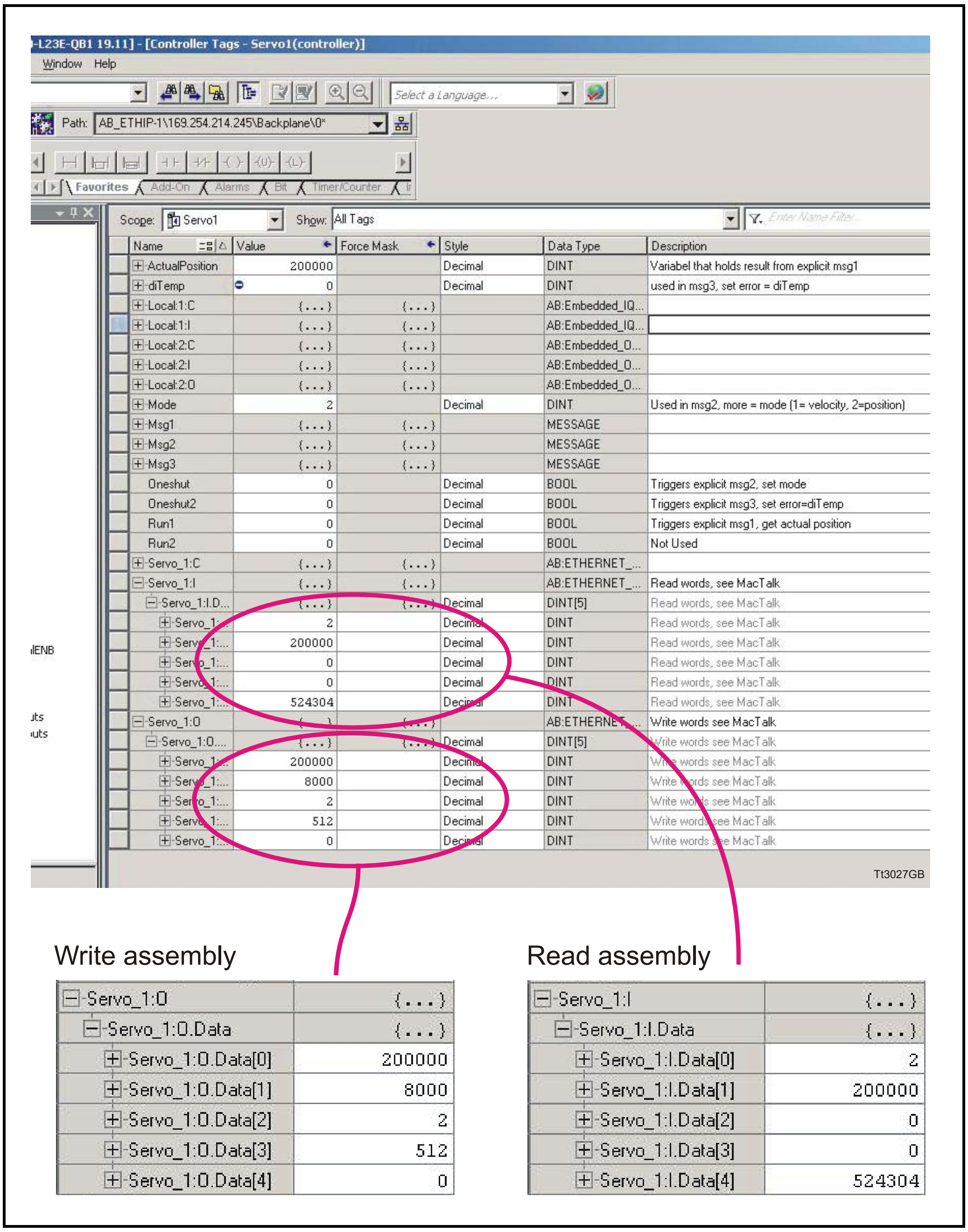
No. of motor registers transmitted in each direction	Motor series MAC050 to MAC141	Motor series MAC400 to MAC4500	Motor series MIS/MILxxx
1/1	4ms *	1ms **	1ms **
2/2	8ms *	1ms **	1ms **
3/3	12ms *	1ms **	1ms **
4/4	16ms *	1ms **	1ms **
5/5	20ms *	1ms **	1ms **
6/6	24ms *	1ms **	1ms **
7/7	28ms *	1ms **	1ms **
8/8	32ms *	1ms **	1ms **

* The minimum cycle times, is only valid if not sending any acyclic requests while in any operating mode. MODULE registers can be appended as the last registers in the list, at no extra timing cost. Motor register 35 shall be in the cyclic read list, as it is also used internally.

** Restrained by the EthernetIP protocol it self.

4.3.3 Cyclic data in the PLC

The complete list of Controller tags defined.



-L23E-QB1 19.11] - [Controller Tags - Servo1(controller)]

Scope: Servo1 Show: All Tags

Name	Value	Force Mask	Style	Data Type	Description
ActualPosition	200000		Decimal	DINT	Variabel that holds result from explicit msg1
diTemp	0		Decimal	DINT	used in msg3, set error = diTemp
Local:1:C	{...}	{...}		AB:Embedded_IQ...	
Local:1:I	{...}	{...}		AB:Embedded_IQ...	
Local:2:C	{...}	{...}		AB:Embedded_O...	
Local:2:I	{...}	{...}		AB:Embedded_O...	
Local:2:O	{...}	{...}		AB:Embedded_O...	
Mode	2		Decimal	DINT	Used in msg2, more = mode (1= velocity, 2=position)
Msg1	{...}	{...}		MESSAGE	
Msg2	{...}	{...}		MESSAGE	
Msg3	{...}	{...}		MESSAGE	
Oneshut	0		Decimal	BOOL	Triggers explicit msg2, set mode
Oneshut2	0		Decimal	BOOL	Triggers explicit msg3, set error=diTemp
Run1	0		Decimal	BOOL	Triggers explicit msg1, get actual position
Run2	0		Decimal	BOOL	Not Used
Servo_1:C	{...}	{...}		AB:ETHERNET_...	
Servo_1:I	{...}	{...}		AB:ETHERNET_...	Read words, see MacTalk
Servo_1:I.D...	{...}	{...}	Decimal	DINT[5]	Read words, see MacTalk
Servo_1:...	2		Decimal	DINT	Read words, see MacTalk
Servo_1:...	200000		Decimal	DINT	Read words, see MacTalk
Servo_1:...	0		Decimal	DINT	Read words, see MacTalk
Servo_1:...	0		Decimal	DINT	Read words, see MacTalk
Servo_1:...	524304		Decimal	DINT	Read words, see MacTalk
Servo_1:O	{...}	{...}		AB:ETHERNET_...	Write words see MacTalk
Servo_1:O....	{...}	{...}	Decimal	DINT[5]	Write words see MacTalk
Servo_1:...	200000		Decimal	DINT	Write words see MacTalk
Servo_1:...	8000		Decimal	DINT	Write words see MacTalk
Servo_1:...	2		Decimal	DINT	Write words see MacTalk
Servo_1:...	512		Decimal	DINT	Write words see MacTalk
Servo_1:...	0		Decimal	DINT	Write words see MacTalk

Tt3027GB

Write assembly

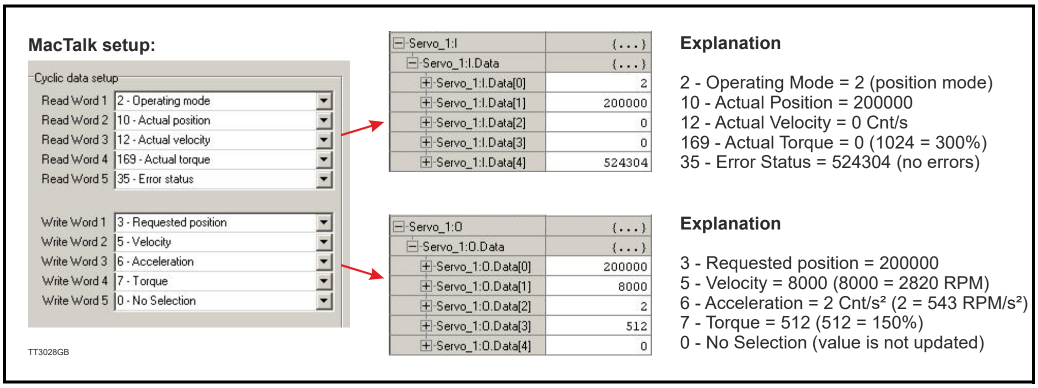
Servo_1:O	{...}
Servo_1:O.Data	{...}
Servo_1:O.Data[0]	200000
Servo_1:O.Data[1]	8000
Servo_1:O.Data[2]	2
Servo_1:O.Data[3]	512
Servo_1:O.Data[4]	0

Read assembly

Servo_1:I	{...}
Servo_1:I.Data	{...}
Servo_1:I.Data[0]	2
Servo_1:I.Data[1]	200000
Servo_1:I.Data[2]	0
Servo_1:I.Data[3]	0
Servo_1:I.Data[4]	524304

4.3 Using cyclic I/O-messages

MacTalk IO assembly setup, seen in the controller tag list and read from the PLC when the connection has been established.



4.4 Commissioning

4.4.1 Necessary equipment

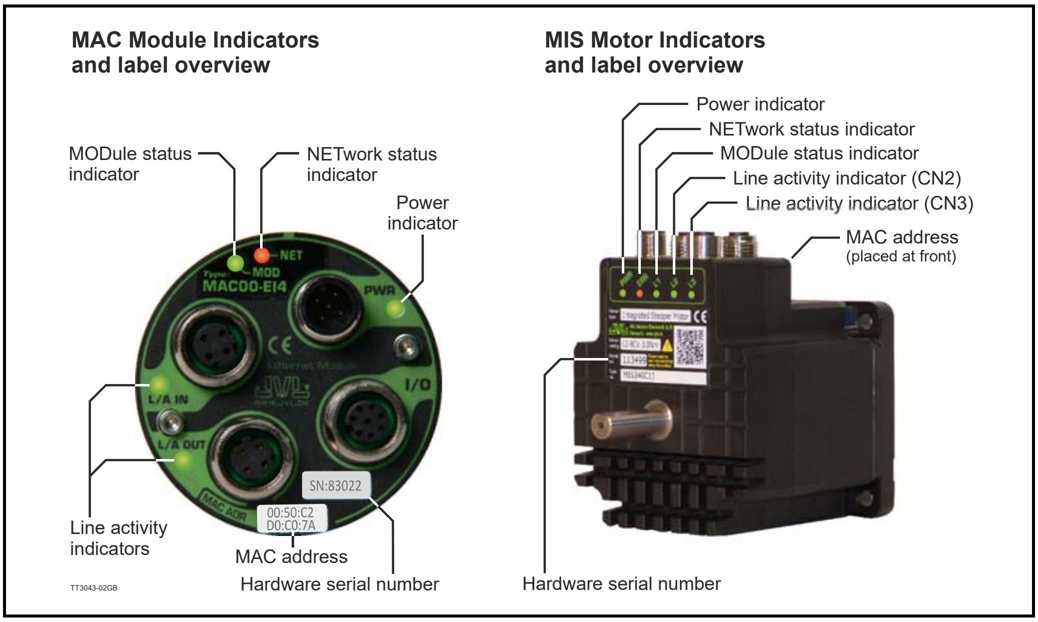
To get started you will need the following equipment.

- MAC motor with an EthernetIP module (MAC00-EIx) or a MIS/MILxxxxxx-EIxxxx motor.
- A PLC or master controller with EthernetIP interface and relevant software
- A PC installed with MacTalk software in order to setup the MAC motor.
- Relevant signal and low voltage cables such as Ethernet cable, 24V power cable, RS232 cable. Please also see the section *Cable accessories, page 26*.
- A 24VDC supply able to deliver min.1000mA@24V pr. motor used.
- Concerning AC high voltage supply for the MAC motor please refer to the general MAC motor user manual (LB0047-xx)

The general MAC or MIS/MIL motor user manual can be downloaded from http://www.jvl.dk

4.4.2 Indicator LED’s - description.

The LED's are used for indicating states and faults of module. There is one power LED, two link/activity LED's (one for each Ethernet connector), and 2 status LED's.



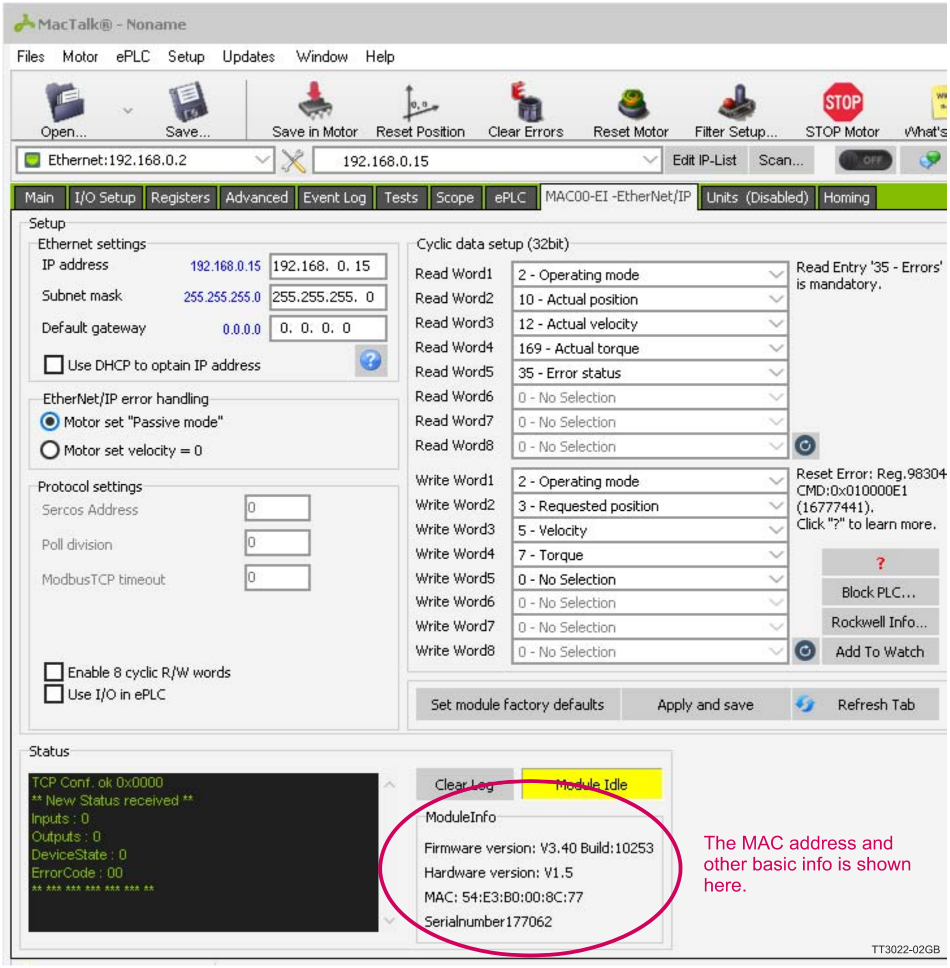
LED indicator descriptions - Covers both MAC and MIS/MIL.

LED Text MAC / MIx	**Colour**	**Constant off**	**Constant on** (Green)	**Blinking** (Green)	**Constant on** (Red)	**Blinking** (Red)	**Blinking** (Red/ Green)	**Flickering**
L/A IN / L2	Green	No valid Ethernet connection.	Ethernet is connected.	-	-	-	-	Activity on line
L/A OUT / L3	Green	No valid Ethernet connection.	Ethernet is connected.	-	-	-	-	Activity on line
MOD / L1	Red/ Green	No power applied	Module status OK	Module not configured	Major module fault	Minor module fault	Self test in progress	-
NET / ERR	Red/ Green	No IP address	CIP connection established	No CIP connection	Duplicate IP address	Connection timeout	Self test in progress	-
PWR	Red/ Green	Power is not applied.	Power is applied.	-	Only **MIS17x** and **MIS23x**: The power supply is too low.	-	-	Power is applied to the module but no communication with the motor

Notes:
Blinking: Flashing with equal on and off periods of 200ms (2.5Hz). **Flickering**: Rapid flashing with a period of approximately. 50ms (10 Hz).

4.4.3 MacTalk Ethernet configuration

The module is by default setup with the following Ethernet configuration:



After adjusting all settings press “Apply and save“
for the settings to take effect and for permanently saving the setup.

Information such as EtherNet/IP firmware version, MAC-address and module status is displayed in the “Status” -field. Notice that the MAC-address is unique for each module and can not be changed.
A label at the front plate of the module also indicate the MAC-address.

Basic use of MacTalk is described in the MAC-motor manual (lit. no. LB0047-xxGB)

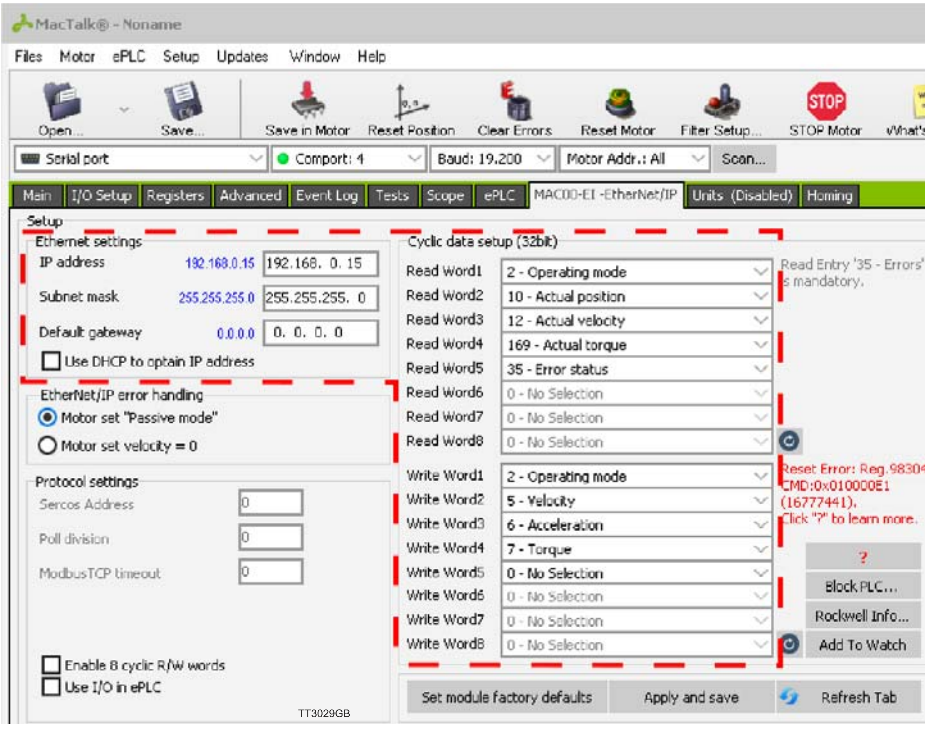
If DHCP is enabled, then make sure a DHCP server is available on the same local network.

4.4 Commissioning

Setting up IP addresses and general usage of the Rockwell CompactLogix PLC with the software package Logix5000 is beyond the scope of this example.

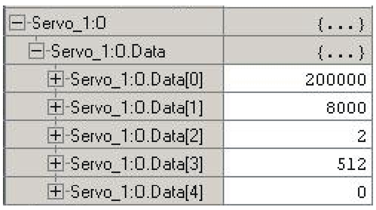
The following guideline is based on the JVL MAC400 motor with the factory setup.

1. Apply 24V, open MacTalk and setup the ethernet settings as required and the IO assembly (cyclic data setup) according to the following:



TT3029GB

2. Press the "Apply and save" -button for permanent storage of the EthernetIP -settings.
3. Switch off the 24V supply while connecting the Ethernet cable to the switch/PLC.
4. Re-apply 24V set the PLC into "RUN" -mode.Now we should be able to control the motor.
5. Start by setting the profile data such as, Velocity, acceleration and Torque. According to the following:

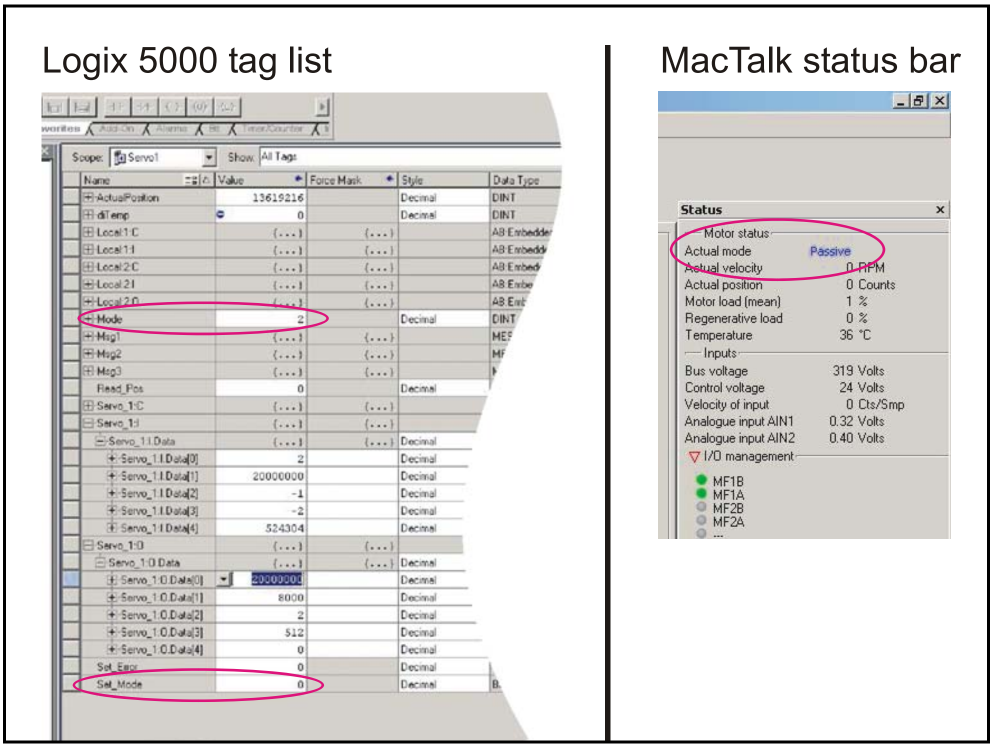


TT3031GB

Explanation

3 - Requested position = 200000
5 - Velocity = 8000 (8000 = 2820 RPM)
6 - Acceleration = 2 Cnt/s² (2 = 543 RPM/s²)
7 - Torque = 512 (512 = 150%)
0 - No Selection (value is not updated)

6. Now we will set the motor into an active mode (position mode), find the Controller tag "Mode" enter 2, find the tag "Set_Mode" enter 1. Now the motor is active and will start moving to the entered position in the "Servo_1:O_Data[0]" which is assigned to the requested position register in the motor. When the motor reaches the position it will stop and hold this position.
From MacTalk the actual mode (see the status-panel) is changed from "Passive" to Position and the motion progress can be followed. Remember to change the "Set_-Mode" tag back to 0 to stop the sending of Msg2 -messages.



Changing the "Servo_1:O_Data[0]"-tag will result in an immediate repositioning of the axle in the motor. This value is defined in the IO assembly and is interchanged cyclic.

To stop the motor set "Mode" = 0 and set "Set_Mode" = 1 to apply the mode setting. Reset "Set_Mode" to 0 again to stop sending Msg2. -messages.

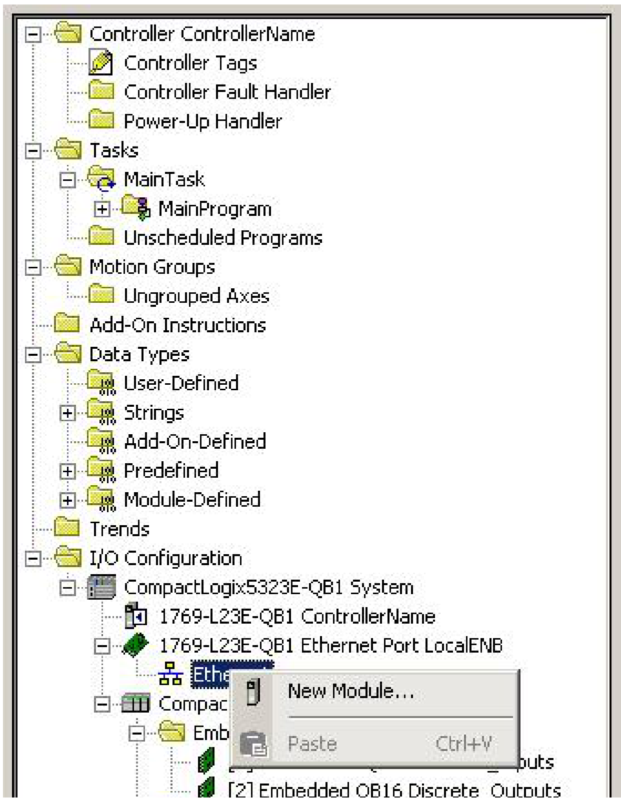
7. To activate the explicit message Msg1 set the commanded position to a far greater value. For example 200000000 as illustrated below.

8. Find the "Read_Pos" -tag and set this to 1. Now the current position of the motor is seen in the "Actual Position" -tag.

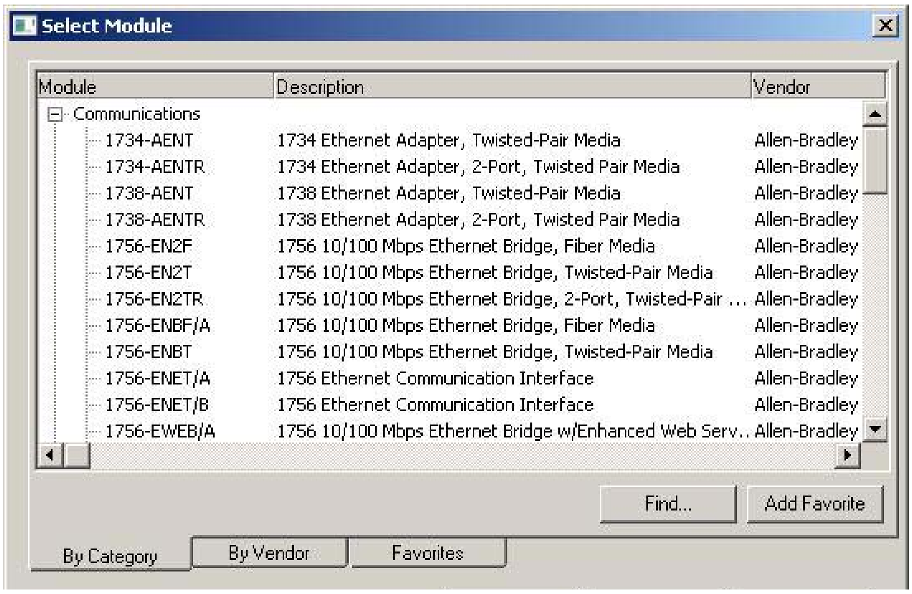
4.4.4 How to setup a Rockwell RSLogix5000 Project.

After creating a new project in the RSLogix5000 application the JVL motor must be added to the Ethernet bus-system in the project.

This is done by right clicking the "Ethernet-Module" icon in the project manager as illustrated below:

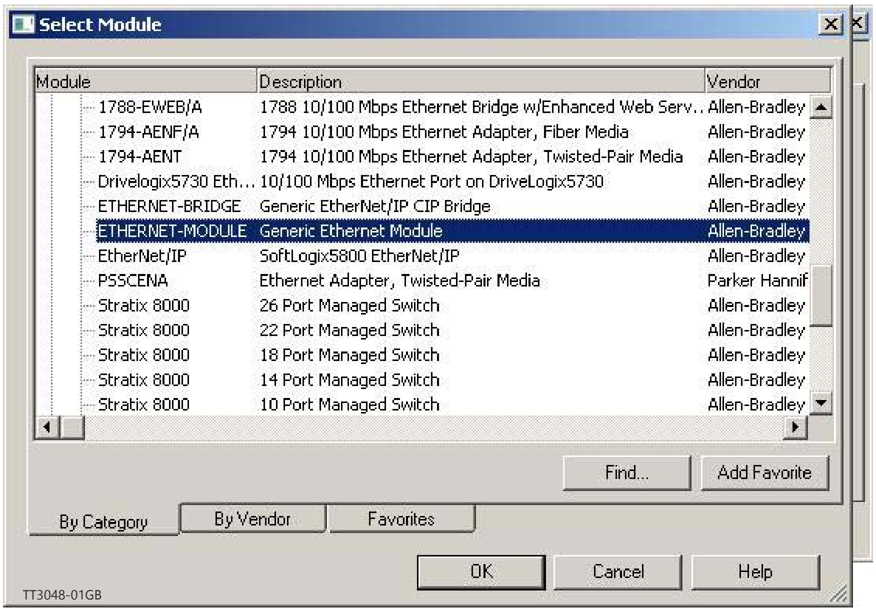


Select "New Module" and the following screen appears:
Expand the "Communications" - list.

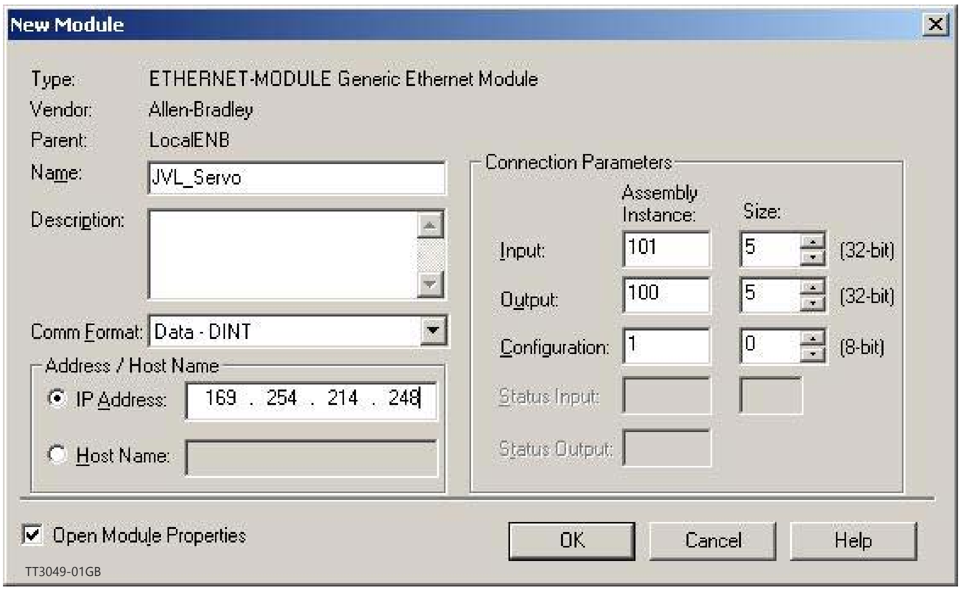


4.4 Commissioning

Find and select the “Generic Ethernet module”.



Now the module parameter needs to be entered.
Fill in the information as illustrated below:



The IP-address illustrated is the factory default and may be changed according to the local settings.

After pressing “Ok” the JVL motor is added to the project and can now be reached by the PLC.

A demonstration video showing how to set-up the system can be seen using this link: http://www.jvl.dk

4.5 Implementation guidelines

4.5.1 Introduction

The following chapters describe the typical usage of the JVL Motor and which registers to use in the different applications.
The chapter should be considered as a general guideline to get started with the EthernetIP integration of the JVL Motor.

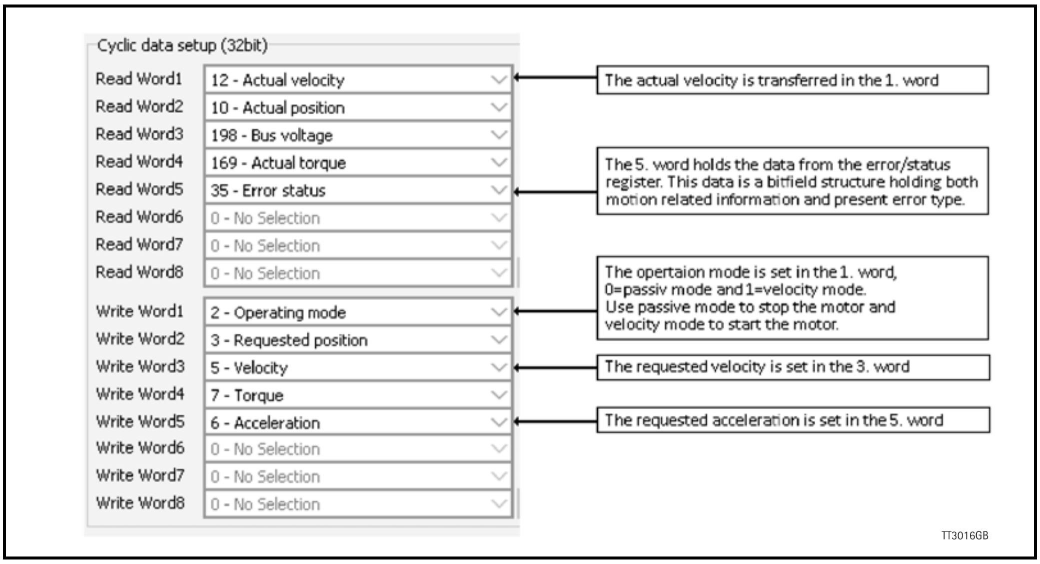
IMPORTANT!: Please notice that the motor will be active and may start moving when the mode register (reg. 2) is set to anything than 0 (passive mode). The MAC400, 800, 1500 and 4500 will require AC supply in order to be active.

4.5.2 Running Velocity control

To use the JVL motor in velocity mode the following registers are basically of interest.

1. “Mode” - Mode register 2
2. “V_SOLL” - Velocity register 5
3. “A_SOLL” - Acceleration register 6
4. “Error/Status” - Error and status register 35

So, to control these registers the assembly object needs to configured.
From MacTalk the setup is configured as this.



With the settings illustrated above we initiate the velocity mode by writing 0x1 to the first word-value, this is velocity mode.

From the scanner the registers is accessed using the assembly object and accessing the registers R/W on words 1-5.

1. Set the needed velocity. V_SOLL = V x 2.77 [rpm]
   Ex. We need the motor to run with a constant speed of 1200 RPM. So, V_SOLL = 1200/2,77 = 433 counts/sample
2. Set the needed acceleration. A_SOLL = A x 271 [RPM/s^2]
   Ex. We need the motor to accelerate with 100000 RPM/s^2 so, A_SOLL = 100000/271 = 369 counts/$sample^2$.
3. Now set the motor into velocity mode and thereby activate the motor.
   Ex. The motor needs to be activated by setting it into velocity mode, so we need to set the mode register to the value 1. Mode = 1 which is velocity mode, now the motor will use the acceleration and the velocity just configured.

Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 and *Motor registers MAC400 - 4500, page 301* or *Motor registers MISxxx, page 320*

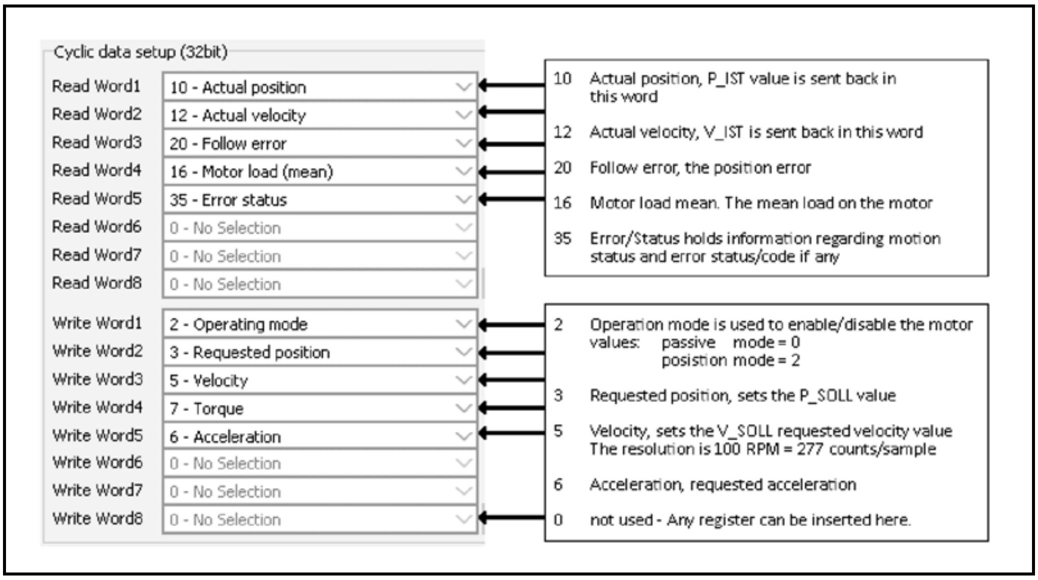
4.5 Implementation guidelines

4.5.3 Running Position control

Running the motor in position control requires that the mode register is set for position control. The following registers is of particular interest when position mode is used.

1. "Actual position" -P_IST, register 10
2. "Actual velocity" -V_IST, register 12
3. "Follow error" - The actual position error, register 20
4. "Motor load mean" - average motor load, register 16
5. "Error/Status" -register 35
6. "Requested position" -P_SOLL, register 3
7. "Requested velocity" -V_SOLL, register 5
8. "Requested acceleration" -A_SOLL, register 6

In this mode the position is controlled by applying a requested position to the "P_SOLL" -register and the actual position is monitored in the "P_IST" register. The V_SOLL and A_SOLL registers sets the velocity and acceleration used when the positioning occurs.



4.5.4 Error/status handling.

The register 35 in the motor holds information on the actual error/status. So it is crucial that this register is configured in the assembly object and thereby obtained and monitored in the scanner. In case of an error situation the motor will stop and the cause will be present in the register 35 and hence in the I/O -data.

This register also holds information on the motion status such as:

- In position, bit 4
- Accelerating, bit 5
- Decelerating, bit 6

Please find a complete list of register descriptions in the appendix. *Motor registers MAC050 - 141, page 292* and *Motor registers MAC400 - 4500, page 301* or *Motor registers MISxxx, page 320*.

The JVL motor is basically put into a working mode and into a passive mode where the motor axle is de-energized, by setting register 2 into either 0 = "passive mode" or into one of the supported modes.

4.6 Configuration with explicit messages

Example.

1= "Velocity mode" / 2= "Position mode" / etc.

So in order to Stop or Start the motor this register can be supported in the I/O data or by sending an explicit message.

Basically a JVL motor works by loading a configuration into RAM memory from the none volatile flash memory when 24V power is applied and the motor is initialized.

The motor only holds one configuration and this configuration can be stored into the NV flash memory.

Several approaches can be used to configure the motor with data and finally saving them permanently in the NV flash.
A very general approach could be by using the PC-based software tool MacTalk, which offers both basic motor setup and control and the possibility to save all parameters in a separate file for backup purposes.
This software package utilizes the serial or network connection to communicate with the motor from any standard Windows PC.

Configuration over EtherNet/IP is possible by using explicit messages to address each register to be setup and then command the motor to save the configuration in flash afterwards for permanent storage.

Using this method the motor only needs to be setup once and is easy achievable from the scanner itself either as an initialization routine each time the PLC initializes, and thereby avoiding the permanent storage in the motor or simply using a configuration routine that sends the required explicit messages to address the needed registers followed by the message to save the settings permanently.

IP-address and other network settings still needs to be setup using MacTalk.

E.g. Setting up a motor sending messages explicitly

We want to change the default motor settings and save them permanently into flash.
The following registers needs to be saved:

The registers needed to be addressed are:

Velocity (V_SOLL) = Register 5
Acceleration (A_SOLL) = Register 6
Torque (T_SOLL) = Register 7

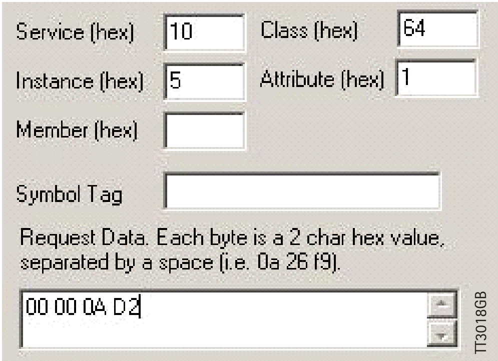
To address individual registers explicitly we use the class 0x64 for the purpose.

4.6 Configuration with explicit messages

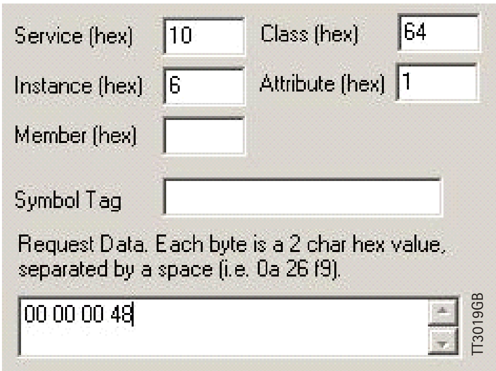
First we change the Velocity setting, we want the motor to spin with 1000 RPM.

The message for addressing V_SOLL is formed:

We need to scale 1000 RPM to the correct value in the motor the factor is 1 RPM = 2.77 counts/sample so we need to send the value 2770 = 0x00000AD2.
The instance refers to the register number, so we need to set instance to 5 (V_SOLL)
Please notice that the value is represented as 32bit.

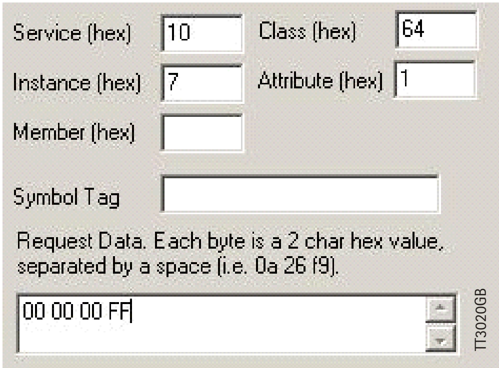


Next we set the acceleration to be used.
We need the acceleration to be 20000 RPM /s2
This value also needs to be scaled, the factor is 1 RPM/s^2 = 0.0036 $counts/sample^2$ so, in order to reach 20000 we need to send the value 72 = 0x00000048.
Acceleration is instance 6 (A_SOLL).

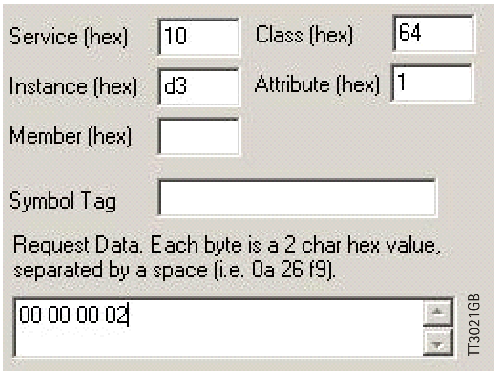


4.6 Configuration with explicit messages

Then configure the maximum motor torque to be used.
The motor can reach a peak torque of 300% the rated value. This value corresponds to 1023 in the register. We need 25% so we write 255 = 0x000000FF to instance 7 (T_SOLL).



And finally we send the command that saves the settings permanently in flash. This is basically a matter of writing the "save in flash" command to the command register 211 in the motor. The command is 2 and the instance is 211 = 0xD3. Value = 0x00000002.
Now the motor saves the setting and resets.
It is required to toggle the 24V power in order to do a internal synchronization.



4.7 Using and Selecting an Ethernet switch

Depending on the network size and requested package interval (RPI) a suitable switch must be used. Also if multiple separated networks needs to be connected a switch is used.

Depending on the actual size of the network different requirements needs to be meet. Generally using EtherNet/IP with a fair package interval a 1 Gbps switch is typical adequate along with the following features:

- Auto negotiation, full duplex 100 MBit
- Port mirroring for network analysing and troubleshooting purposes. This feature makes it possible to route traffic out on a separate port connected to a network analyser for debugging purposes and general performance monitoring.

The JVL EthernerNetIP module has a small build in 2 port switch useful if a small amount of motors is connected in a daisy chaining topology.

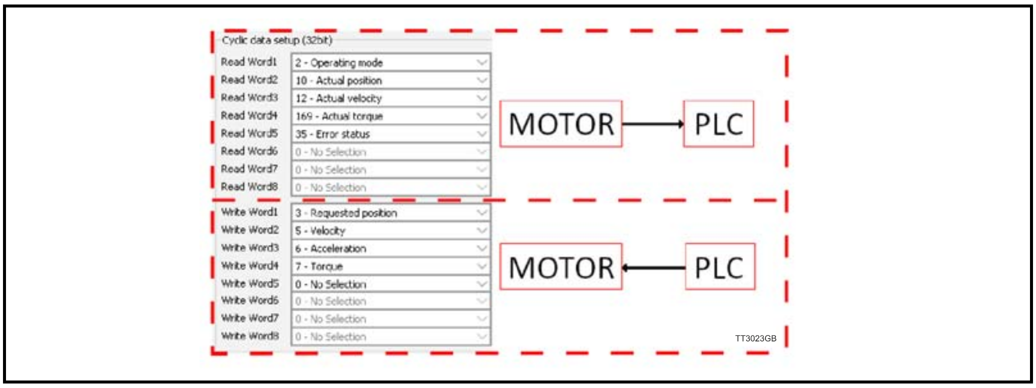
The disadvantage of this approach is that the package RPI timing is reduced as each motor needs to handle the incoming traffic for the other motors connected on the line.

4.8 Examples

4.8.1 Rockwell RSLogix example 1.

This is a simple example demonstrating the usage of both explicit messages and IO-assemblies to control a JVL MAC400 servo motor.
This example holds a few tags to control the inputs and outputs and a 3 rung ladder program to demonstrate the explicit message usage.
With this example it is possible to control the positioning of the motor using the “Position -mode” and set profile data such as velocity, acceleration and torque parameters using the IO-assembly.

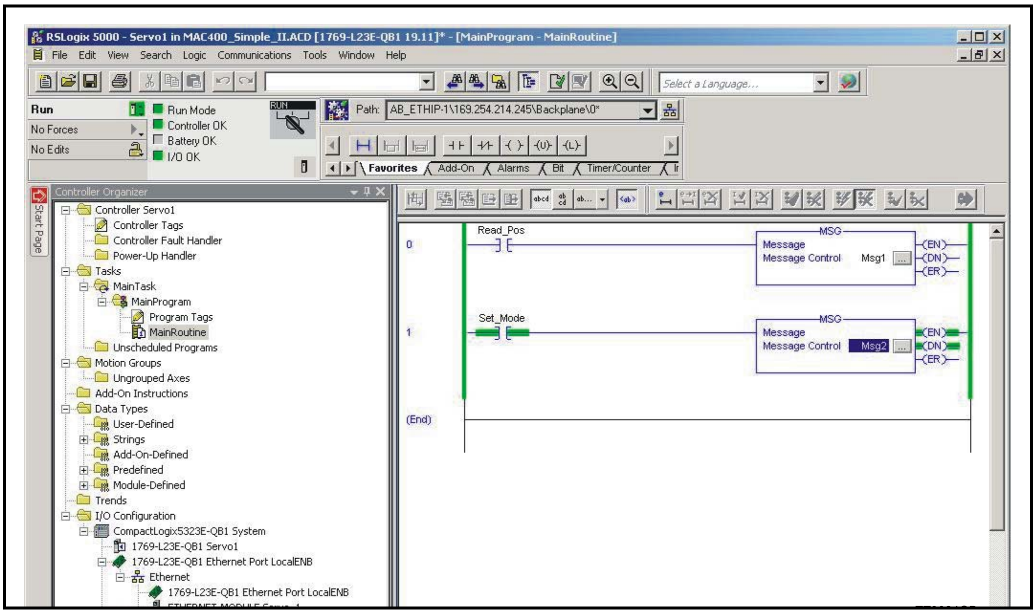
The example is developed for use on a CompactLogix L23E PLC using the Rockwell Logix500 software package and MacTalk from JVL.
The JVL MacTalk application is used to setup the IO assembly to fit the example.
Although this example expects default setup in the JVL motor, the IO assembly needs to be setup according to the following MacTalk setup (located at the EthernetIP tab).



The fixed sized assembly instances is divided into 5 read words and 5 write words.

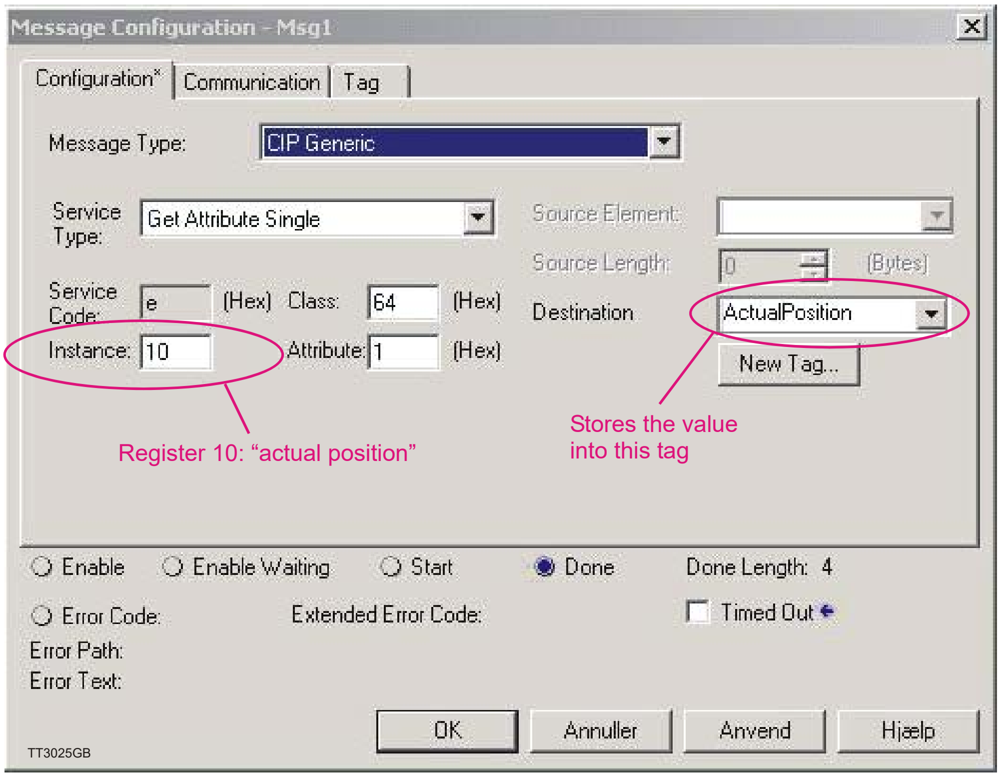
4.8.2 The RSLogix ladder program.

3 different messages for both setting data and retrieving data from the motor. All 3 messages are triggered by separate variables from the controller tag-list.



4.8.3 Message descriptions.

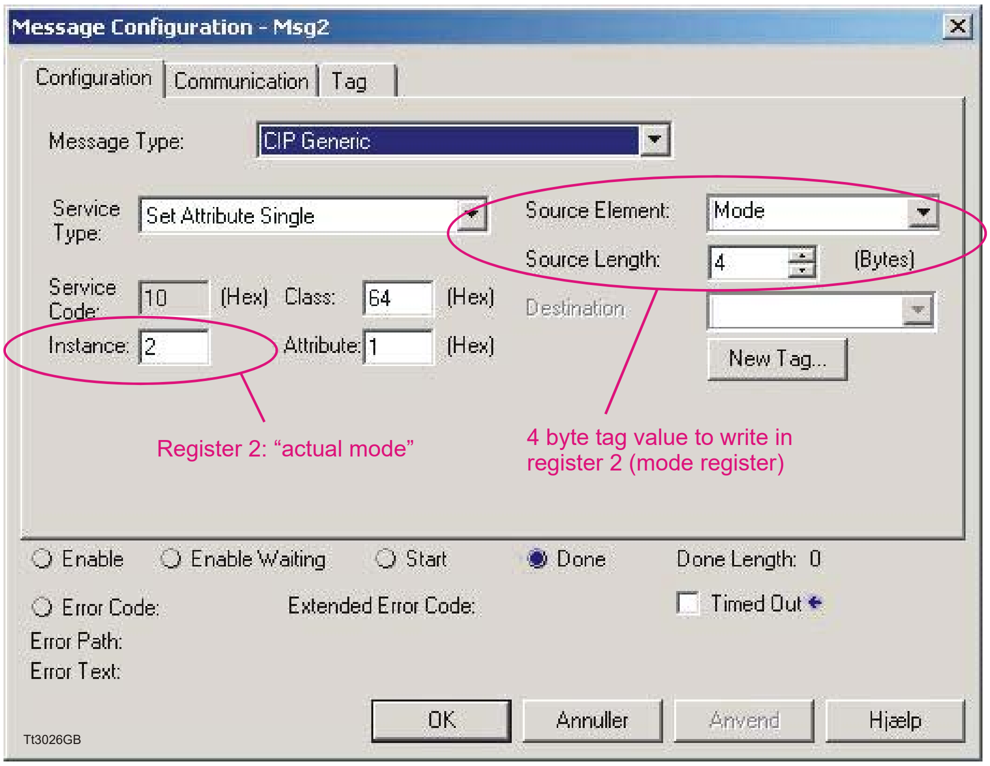
Msg1 reads information from the motor and is setup in the following way:
Reads (GET_ATTRIBUTE_SINGLE) the actual position register in the motor (instance 10) and stores the 4 byte value in the "ACTUAL POSITION" tag.



Please notice the mixed decimal and hexadecimal notation used.

4.8 Examples

Message 2 and 3 (Msg2, Msg3) are writing values to specific registers in the motor. They are configured in the following way:
Writes (SET_ATTRIBUTE_SINGLE) the value from the "MODE"-tag into the motor register 2 (Operation mode).



Please notice the mixed decimal and hexadecimal notation used.

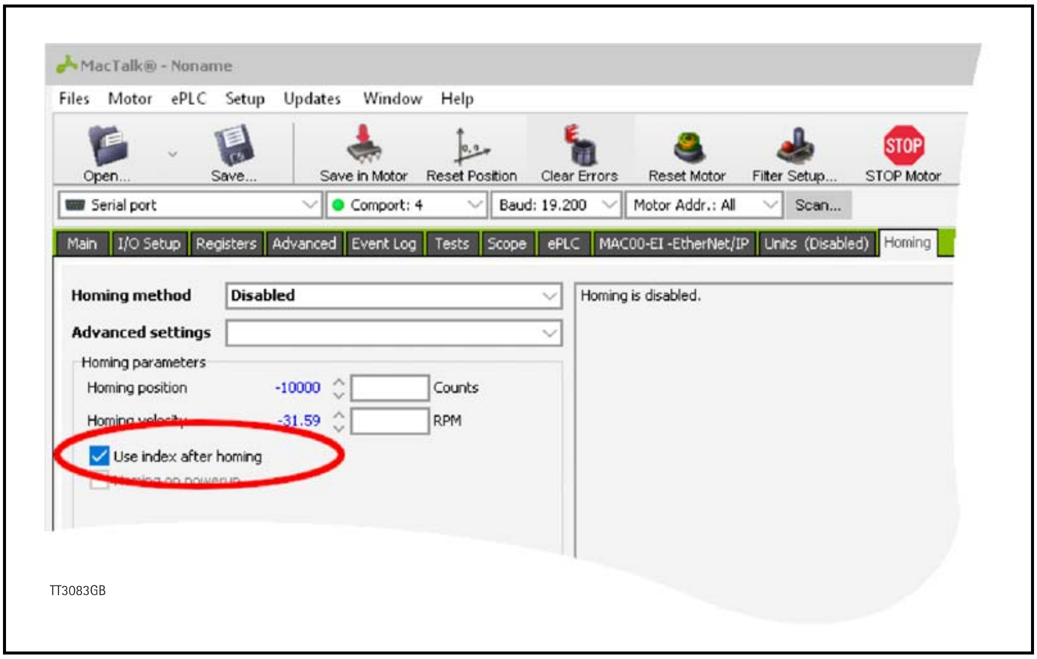
Explicit messages are always 4 bytes long and uses Class 0x64 to access the internal motor registers.
The instance refers to the actual motor register.

Instance = 2 points to the motor active mode -register.

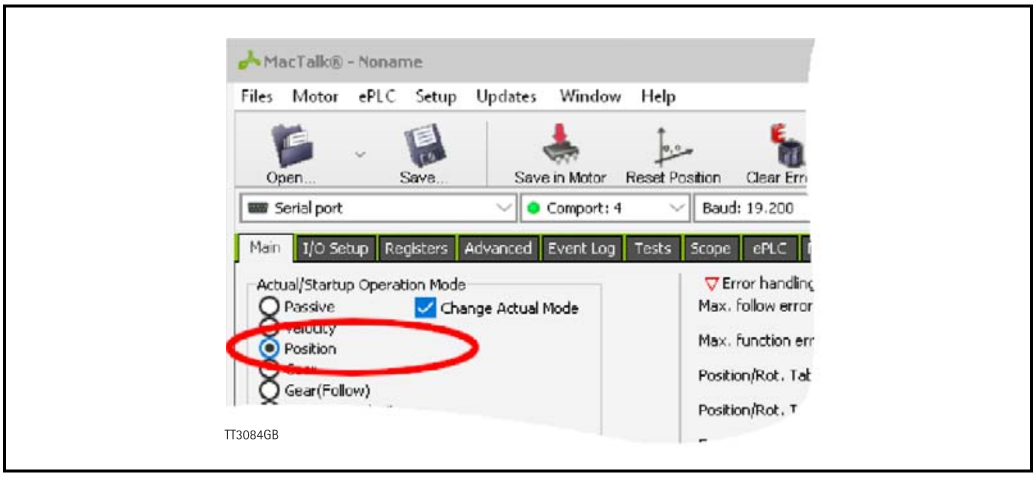
Explicit messages are typical used for configuration purpose or for rare data update situation that does not require a cyclic update timing.

4.8.4 Homing using only cyclic I/O (JVL profile).

When doing a homing (Zero search), with only cyclic I/O, some preconditions have to be met:
Zero search position, zero search velocity and zero search torque (torque only for MAC motors) has to be set in MacTalk in the "Main" tab, and saved in flash in the motor once and for all.

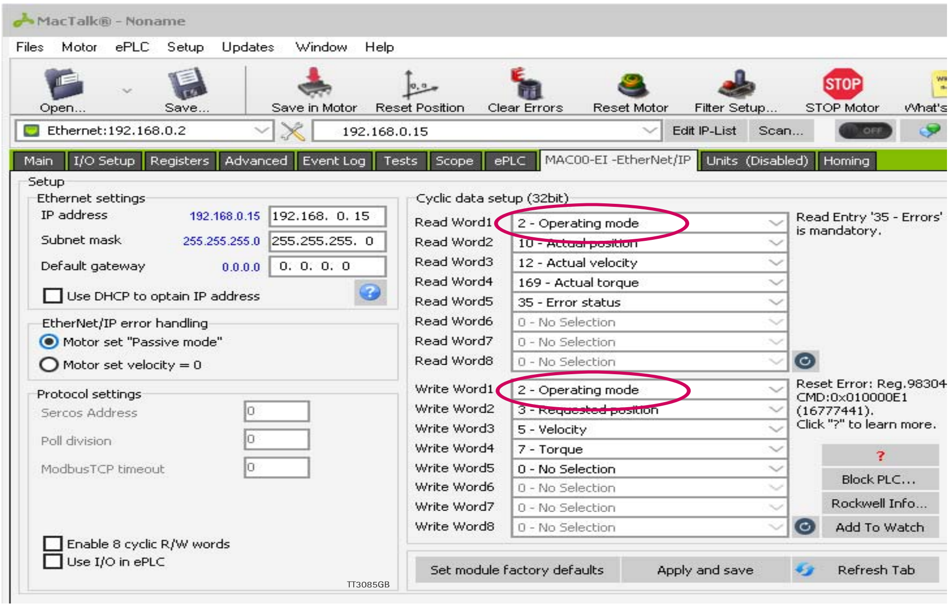


Startup mode should be set to position, for the motor to stay in position after the homing sequence. And this setting should also be saved in flash.



4.8 Examples

Register 2 (Operating mode) has to be present in BOTH the cyclic read words and cyclic write words.



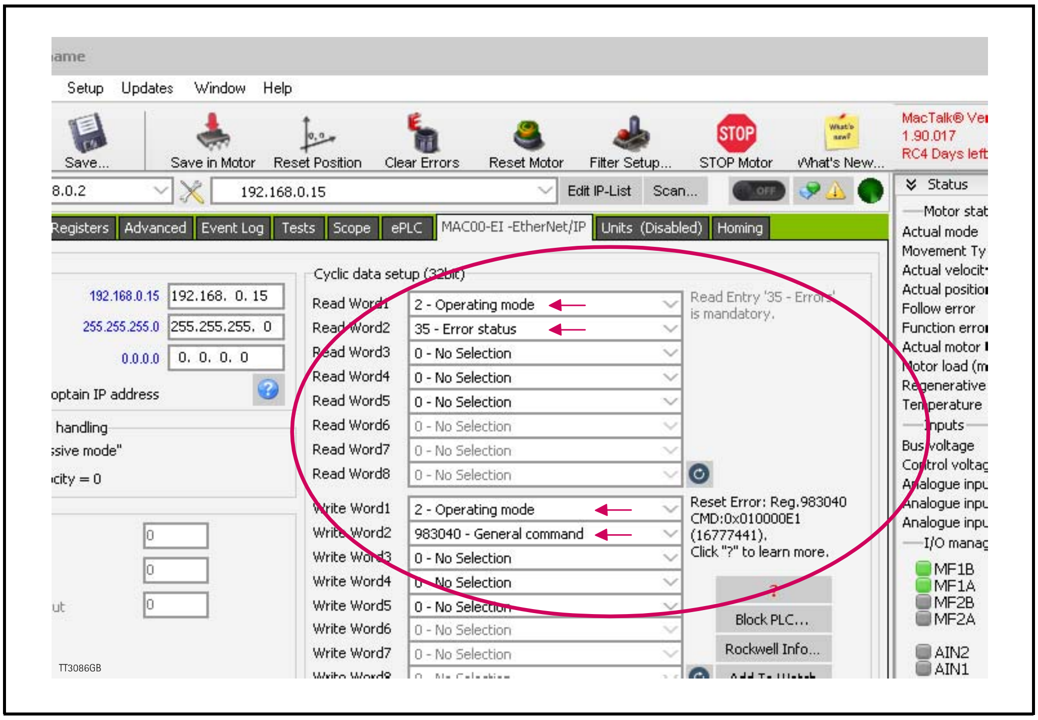
Procedure in the PLC:

- Treat the transmitted Register 2 as "Requested_Mode" and the received register 2 as "Actual_Mode".
- When homing is wanted, set the "Requested_Mode" to one of the values 12, 13 or 14, 25 or 26 depending of the requested homing mode (12 = Torque based zero search mode (only MAC motors). 13 = Forward/only zero search mode. 14 = Forward+backward zero search mode (only MAC motors). 25 = Enc. index (only MAC400+). 26 = Enc. quick index (only MAC400+).). For a comprehensive description of the homing modes, refer to the general MAC motor manual - LB0047-xxGB
- Observe that the "Actual_Mode" is changing to the homing mode. Now the module is blocking cyclic writes TO the motor. Cyclic reads is still active.
- Wait for register 35 "Error status" bit 4 to be active =IN_POSITION. (Indicates that homing is finished).
- Then change "Requested_Mode" to whatever needed. The blocking of cyclic writes to the motor is then released by the module.

4.8.5 Relative positioning.

There are a number of ways to do relative positioning, but the one explained here is very simple, and can be used with a constant distance, or exchangeable distance, to move every time it is requested.

Preconditions:
Place the module command register (register 983040 in MacTalk) in the cyclic write list. The cyclic setup, could for example look like this:



Procedure in the PLC:

1. Set up register P7 in motor to requested relative offset.
2. Make sure one net cycle has passed, so P7 resides in the motor.
3. Issue command 0x010000F1 (0x01000071 if the MIS/MILxxx motor is used) in module command register (register 983040 in MacTalk).4.Make sure one net cycle has passed, so command is interpreted by the motor.
5. Set module command register to zero. This will prepare the Ethernet module for new commands.
6. If needed then monitor register 35 (Error status): When bit 4 is set (in position), then the move is finished.
7. When a new relative move is requested, go to step 3.

You may also have the P7 register in the cyclic write list, thereby enabling easy change of the relative distance to move.

4.9 ODVA Conformance Certificate

ODVA
Networks Built on a Common Industrial Protocol

Declaration of Conformity to
The EtherNet/IP™ Specification

ODVA hereby issues this Declaration of Conformity to *The EtherNet/IP™ Specification* for the product(s) described below. The Vendor listed below (the "Vendor") holds a valid Terms of Usage Agreement, which is incorporated herein by reference, for the EtherNet/IP Technology from ODVA, thereby agreeing that it is the Vendor's ultimate responsibility to assure that its EtherNet/IP Compliant Products conform to *The EtherNet/IP Specification* and that *The EtherNet/IP Specification* is provided by ODVA to the Vendor on an AS IS basis without warranty. NO WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, ARE BEING PROVIDED BY ODVA.

In recognition of the below EtherNet/IP Compliant Product(s) having been EtherNet/IP Conformance Tested at ODVA-authorized Test Service Provider and having received a passing result from ODVA at the Composite Test Revision Level specified below, this Declaration of Conformity authorizes the Vendor to use the EtherNet/IP Certification Marks in conjunction with the specific EtherNet/IP Compliant Product(s) described below, for so long as the Vendor's Terms of Usage Agreement for the EtherNet/IP Technology remains valid.

EtherNet/IP™ conformance tested — *Certification Logo Mark*

EtherNet/IP CONFORMANCE TESTED ™ — *Certification Word Mark*

This Declaration of Conformity is issued on April 9, 2014 on behalf of ODVA by:

Katherine Voss

Katherine Voss
Executive Director

Vendor Information	
Vendor Name	JVL Industri Elektronik A/S

Test Information	
Test Date	October 16, 2013
Composite Test Revision	CT10
ODVA File Number	11205.01

Product Information	**Network Category: Node**
Identity Object Instance	
Vendor ID (Attribute 1)	936
Device Type (Attribute 2)	0x0C
Device Profile Name	Communications Adapter

Products Covered under this Declaration of Conformity (Identity Object Instance)

No.	Product Code (Attribute 3)	Product Name (Attribute 7)	Product Revision (Attribute 4)	SOC File Name
1	1	MAC00-Eix	3.026	MAC00_EIx_hilscherCorr_Aug13_jvl.stc

TT3091-01GB

5 POWERLINK Users Guide

5.1 Introduction to POWERLINK

MAC Powerlink Module
Type:
MAC00-**EL**4 (shown) or
MAC00-**EL**41 (extended I/O)

To be used in following servo products:
MAC50, 095, 140 and 141
MAC400 and MAC402
MAC800
MAC1500 and MAC3000

MIS Powerlink motors.
Type:
MISxyyzz**EL**xx85 or
MILxyyzz**EL**xx85

ETHERNET POWERLINK

TT3044-02GB

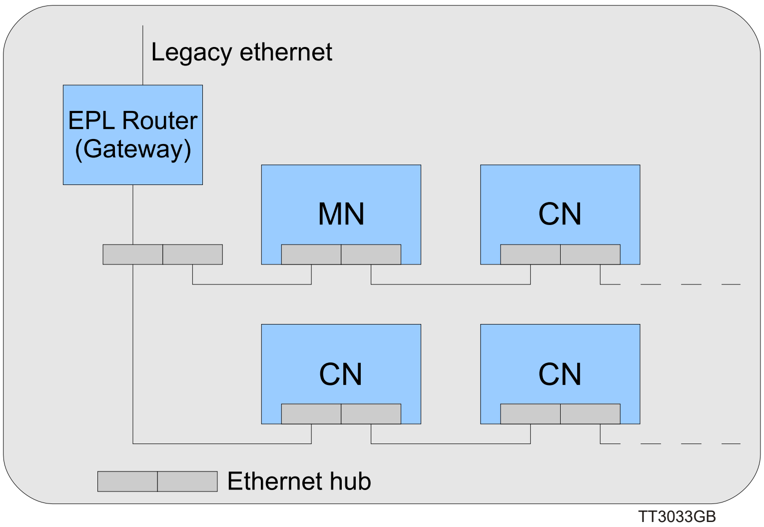
5.1.1 Introduction.

Ethernet Powerlink (EPL) is a proven technology, working in real applications worldwide. It embraces standard Ethernet technology and infrastructure, uses standard CAT5 shielded cabling and does not compromise standard Ethernet frames in order to achieve its results.
Ethernet Powerlink is a truly open technology independently managed by the Ethernet Powerlink Standardization Group (http://www.ethernet-powerlink.org).
Powerlink operates as a protected segment by design, and connects to a none-deterministic Ethernet network via a gateway/router device. This gateway acts as a defensive barrier against attacks by providing fire wall security measures.

5.1 Introduction to POWERLINK

Unlike standard Ethernet, the Slot Communication Network Management (SCNM) ensures that only one node is accessing the network at a time. The schedule is divided into an isochronous phase and an asynchronous phase. During the isochronous phase, time-critical data is transferred, while the asynchronous phase provides bandwidth for the transmission of data that is not time-critical. The Managing Node (MN) grants access to the physical medium via dedicated poll request messages. As a result, only one Controlled Node (CN) has access to the network at a time, and thus no collisions occur. Ethernet POWERLINK applies the same protocol technology as CANopen. It defines SDOs (Service Data Objects), PDOs (Process Data Objects) and the Object Dictionary structure to manage the parameters.

For general technical data please see *Powerlink for MAC or MIS - Technical specifications, page 287*.



5.1 Introduction to POWERLINK

5.1.2 Abbreviations

Following general used terms are useful to know before reading the following chapters.

100Base-Tx	100 MBit Ethernet on twisted pairs
ASnd	Asynchronous Send (POWERLINK frame type)
CAN	Controller Area Network
CANopen	Application layer protocol used in automation.
CN	Controlled Node (slave on Ethernet Powerlink network)
EN	Exception New (flag in POWERLINK frame)
EMCY	Emergency Object.
EPL	Ethernet PowerLink
EPSG	Ethernet PowerLink Standardisation Group
ID	Identifier
IP	Internet Protocol - IP address ~ the logical address of the device, which is user configurable.
MAC	Media Access Controller - MAC address ~ the hardware address of the device.
MacTalk	A windows PC based program supplied from JVL. This is an overall program to install, adjust and monitor the function of the motor and a module installed in the motor.
MN	Managing Node (master on Ethernet Powerlink network)
NAT	Network Address Translation (used in EPL router, to reach destinations outside EPL segment)
NMT	Network Management
PDO	Process Data Object (for cyclic data)
PReq	Poll Request. A frame used in the isochronous phase of the cyclic communication. With Poll Request, the MN requests the CN to send its data.
PRes	Poll Response. A frame used in the isochronous phase of the cyclic communication. The CN responses with a Poll Response frame when it receives a Poll Request from the MN.
SCNM	Slot Communication Network Management; In a POWERLINK network, the MN allocates data transfer time for data from each node in a cyclic manner within a guaranteed cycle time. Within each cycle there are slots for Isochronous Data, and for Asynchronous Data for ad-hoc communication. The SCNM mechanism ensures that there are no collisions during physical network access in any of the net worked nodes thus it provides deterministic communication via Legacy Ethernet.
SDO	Service Data Object (for acyclic data)
SoA	Start of Asynchronous (POWERLINK frame type)
SoC	Start of Cyclic (POWERLINK frame type)
TCP	Transfer Control Protocol (an IP based protocol used widely on the internet)
UDP	User Datagram (an IP based protocol used widely on the internet)
XDD	File extension for the device description file.
XML	Extensible Markup Language - used for the device description file.

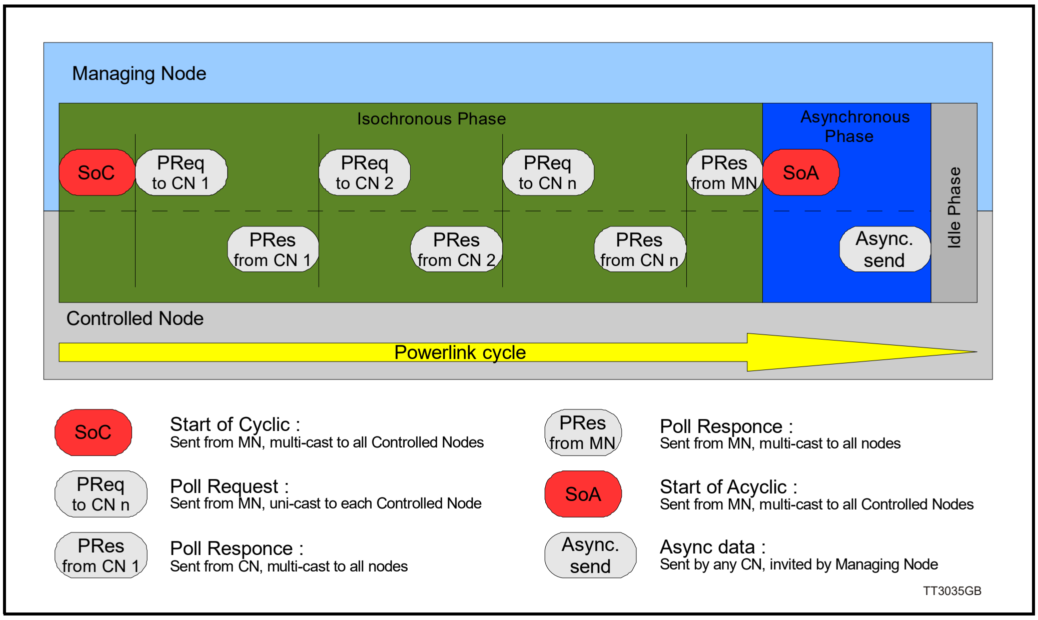
5.2 Protocol specifications

5.2.1 Ethernet Powerlink communication

In an Ethernet POWERLINK network, one of the nodes, for example a PLC, is designated to function as the MN, the master in the network. All other devices operate as CNs, slaves in the network. The MN defines the clock pulse for the synchronization of all devices and manages the data communication cycle. In the course of one clock cycle within which all nodes are addressed, the MN sends Poll Requests (PReq) to all CNs, one after another. They reply immediately to the prompts with Poll Responses (PRes).
The following time phases exist within one cycle:

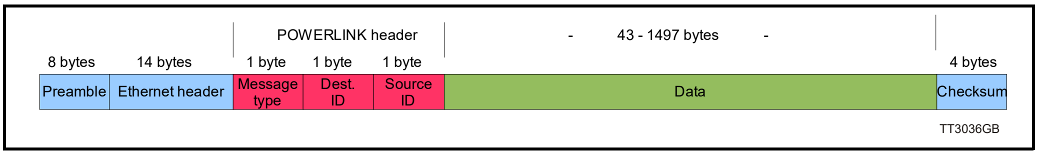
- Isochronous phase
- Asynchronous phase
- Idle phase

The MN first sends a Start of Cycle Frame (SoC) signal to all CNs to synchronize the devices. Payload data exchange then proceeds in the isochronous phase. The asynchronous phase, allows the transfer of large packets that are not time-critical, for example parameterisation data or transfer of IP-based protocols like TCP or UDP. The Idle phase can be 0. It's possibly for the MN to multiplex the time slots in the isochronous phase, in order to service some CN's more often than others. During system start-up the MN applies a reduced POWERLINK cycle, without the isochronous phase, in order to configure the CNs with SDO communication.
For further information, please refer to the Ethernet POWERLINK communication profile specification “EPSG_DS_301_V-1-1-0_01.pdf”, available at the EPSG website http://www.ethernet-powerlink.org.



5.2.2 Ethernet POWERLINK® frame structure

POWERLINK messages are encapsulated in Ethernet II frames. The length of the frame is restricted to the configured size, in order to guarantee the cycle time. Ethernet frames have a minimum length of 64 bytes and a maximum of 1518 (exclusive preamble). The Ethernet POWERLINK header contains only 3 bytes. Message type, destination ID and Source ID. That leaves up to 1497 bytes of payload.



5.2.3 Ethernet POWERLINK CN State machine

In Ethernet POWERLINK, a Controlled Node starts up by a common initialization process. All the states are valid when the device is powered and they are sub-states of the NMT_GS_POWERED superstate.

NMT_GS_INITIALISATION

After system start, the device automatically assumes this state and network functionality begins. NMT_GS_INITIALISATION and all its sub-states are only internal states of the device. In the NMT_GS_RESET_CONFIGURATION sub-state, the node address of the device is identified and it is determined whether it is configured as a MN or CN. The JVL MAC00-ELx is a CN and thus, it enters the NMT CN state machine in the NMT_GS_COMMUNICATING super-state.

NMT_GS_COMMUNICATING

NMT_CS_NOT_ACTIVE

This is a none-permanent state that allows a starting node to recognize the current network state. Time out for SoC, PReq, PRes and SoA frames trigger the device to enter state NMT_CS_BASIC_ETHERNET.

The NMT_CS_PREOPERATIONAL_1

Transition from NMT_CS_NOT_ACTIVE to NMT_CS_PRE_OPERATIONAL_1 is triggered by a SoA or SoC frame being received. In this state CN may send a frame only if the MN has authorized it to do so by a SoA command. There is no PDO communication in this state. Receiving a SoC frame triggers the transition from NMT_CS_PREOPERATIONAL_1 to NMT_CS_PREOPERATIONAL_2.

The NMT_CS_PREOPERATIONAL_2

In this state PReq and PRes data may be invalid because PDO mappings may differ. In NMT_CS_EPL_MODE, error recognition (for example, loss of SoC or PReq) always triggers the transition to NMT_CS_PREOPERATIONAL_1.

The NMT_CS_READY_TO_OPERATE

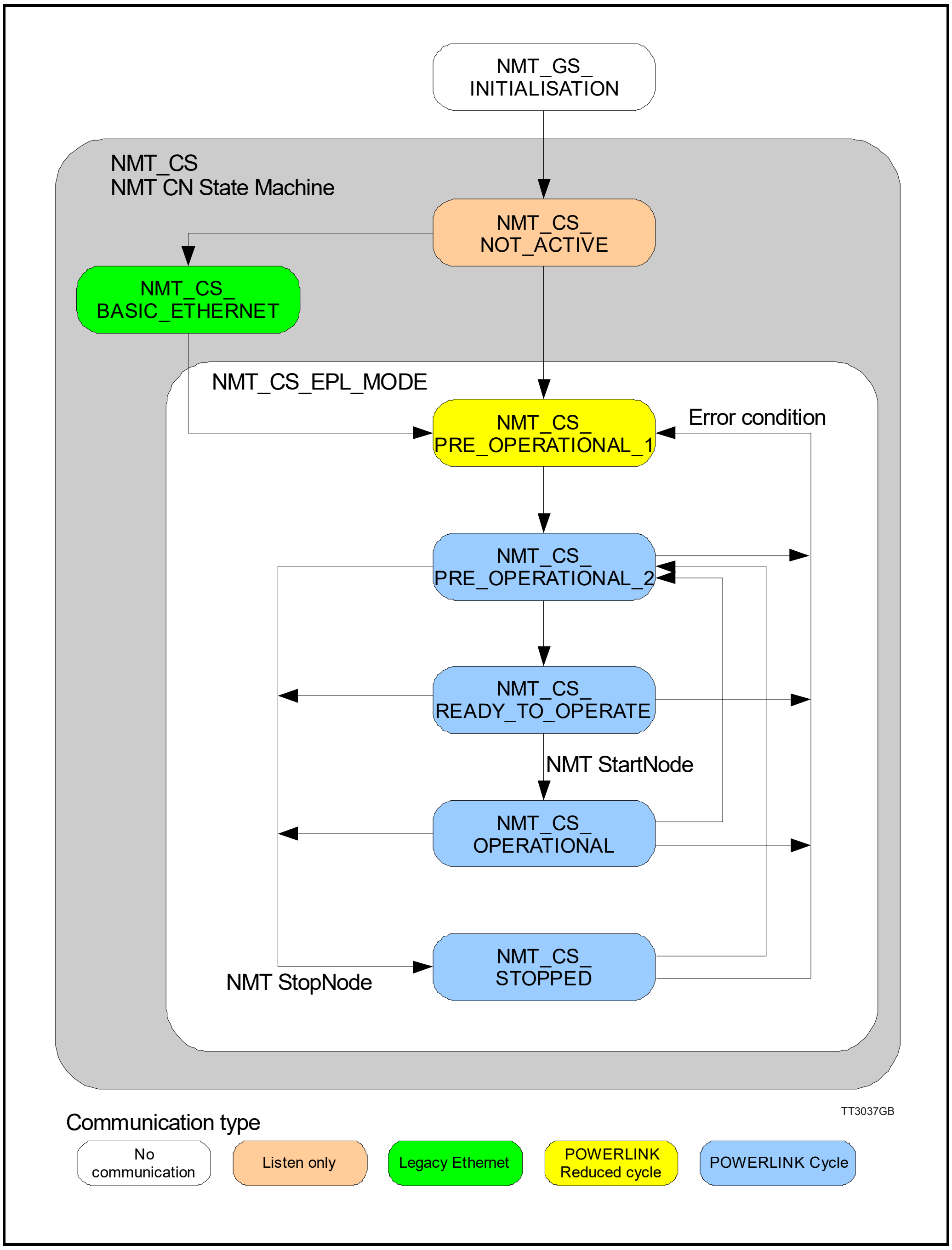
In this state, the CN signals that it is ready to operate to the MN. It responds to the PReq query of the MN by sending a PRes frame.

The NMT_CS_OPERATIONAL

NMT Start Node command triggers the transition from NMT_CS_READY_TO_OPERATE to the NMT_CS_OPERATIONAL. This is the normal operating state of the CN.

The NMT_CS_STOPPED

This state is used for controlled shutdown of a selected CN while the system is still running. In this state, the CN does not participate in cyclic frame exchange, but it still observes SoA frames.



5.2.4 Application layer communication

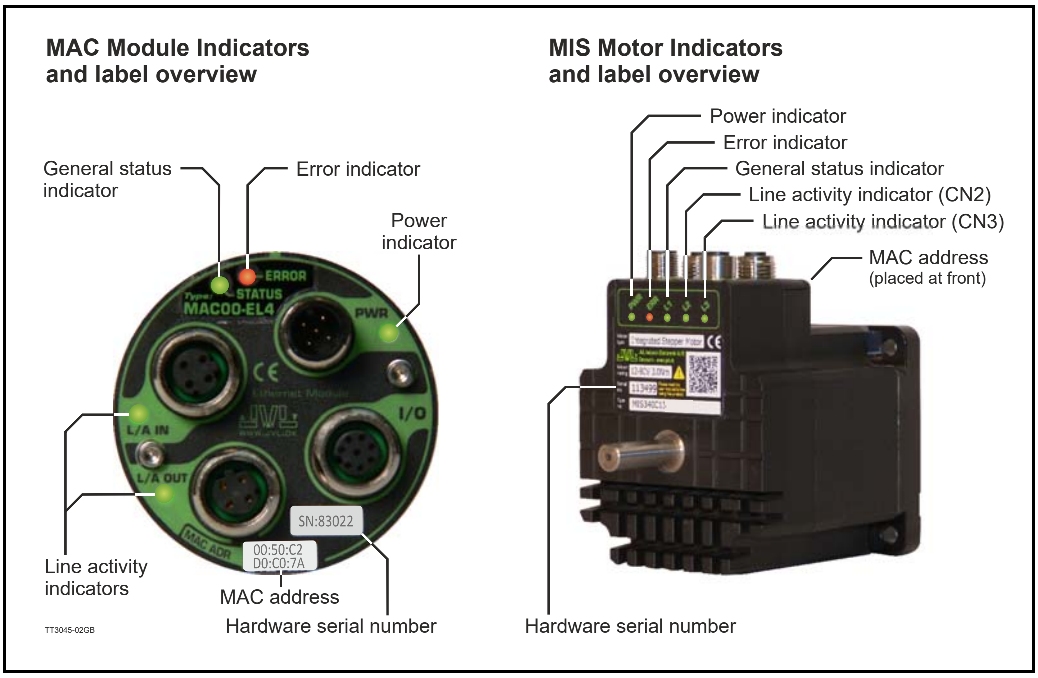
The application layer communication protocol in Ethernet POWERLINK is based on the CANopen DS 301 communication profile. The protocol specifies the Object Dictionary in the adapter module, in addition to communication objects for exchanging cyclic process data and acyclic messages.
The MAC00-ELx module uses the following message types:

- Process Data Object (PDO). The PDO is used for cyclic I/O communication, in other words, process data.
- Service Data Object (SDO). The SDO is used for much slower acyclic data transmission.
- NMT response services. Used for identity and status signalling during start-up and runtime.

5.3 Commissioning

5.3.1 Indicator LED's - description.

The LED's are used for indicating states and faults of module. There is one power LED, two link/activity LED's (one for each Ethernet connector), and 2 status LED's.



LED indicator descriptions - Covers both MAC and MIS/MIL.

LED Text MAC / MIx	Colour	Constant off	Constant on	Blinking	Single flash	Double flash	Triple flash	Flickering
L/A IN / L2	Green	No valid Ethernet connection.	Ethernet is connected.	-	-	-	-	Activity on line
L/A OUT / L3	Green	No valid Ethernet connection.	Ethernet is connected.	-	-	-	-	Activity on line
STATUS / L1	Green	NMT_CS_NOT_ACTIVE	NMT_CS_-OPERA-TIONAL	NMT_CS_STOPPED	NMT_CS_PREOP-ERA-TIONAL1	NMT_CS_PREOP-ERA-TIONAL2	NMT_CS_READY_TO_OP-ERATE	NMT_CS_BASIC_ETH-ERNET
ERROR / ERR	Red	No error	Error					Booting error
PWR	Red/ Green	Power is not applied.	Power is applied to both motor and mod-ule. **MIS17x/23x** The LED will lit red constantly if the power supply is too low.					Power is applied to module but no communi-cation with motor.

Notes:
Blinking: Flashing with equal on and off periods of 200ms (2.5Hz). **Single flash:** Repeating on for 200ms and off for 1s. **Double flash:** Two flashes with a period of 200ms followed by 1s off period. **Triple flash:** Two flash-es with a period of 200ms followed by 1s off period.**Flickering:** Rapid flashing with a period of approximately 50ms (10 Hz).

5.3 Commissioning

5.3.2 Mechanical installation

The network cables must be connected to the two M12 connectors (marked "L/A IN" and "L/A OUT") at the MAC module and "CN2" and "CN3" at the MIS/MIL motors.
The cable from the MN is connected to either of the two ports. In the line topology, if there are more slave devices in the same line, the next slave device is connected to the second port. If there is a redundant ring, the second port of the last slave device is connected to the second port of the MN.
See also the illustration in the chapter *Introduction., page 114*
Standard CAT 5 FTP or STP cables can be used. It is not recommended to use UTP cables in industrial environments, which is typically very noisy.

5.3.3 Quick start

This section describes the steps to configure the PLC, B&R X20 CP1485, with B&R Automation Studio PC software, so that it can be used to control the drive.

Set node ID

1. Connect the RS232 communication cable.
2. Apply power to the motor, and make sure the PWR LED is lit.
3. Open MacTalk and select the "MAC00-EL (Powerlink)" tab.
4. Change the last number in the IP address (= node ID), to one that doesn't conflict with other devices on the sub net.
5. Press "Apply and save".

Installation

6. Connect an Ethernet RJ45-M12 cable to IF3 on the X20 and to L/A IN or L/A OUT at the MAC00-ELx module or the "CN2" and "CN3" at the MIS/MIL motors.
7. Connect power to the X20, and communication cable from the PC with B&R Automation Studio installed to the X20 PLC (either Ethernet or RS232).
8. Make sure power is applied to all devices.

PLC configuration

9. Create a new project in Automation Studio for your PLC, or open an existing project. See B&R documentation for more information.
10. In the Project Explorer window, open the Physical View tab
11. Right-click the node representing the CPU (in this example, X20CP1485-1), and in the pop-up menu, select Open IF3 POWERLINK Configuration. The POWERLINK Configuration window is opened.
12. Make sure that "Activate POWERLINK communication" is set to "on".
13. Close the window and save changes.

Add the XDD file (contains info on the capabilities of the device)

14. In the Tools menu of Automation Studio, select Import fieldbus device...
15. In the Open window find and select the "00000117_MAC00-ELx.xdd" file, and click Open.
    This link can be used : http://www.jvl.dk/default.asp?Action=Details&Item=428

(continued next page)

Associating with MAC00-ELx

16. In the physical view of the project explorer window, right click the CPU node and click Open POWERLINK in the pop-up menu.
17. Right click IF3 in the opened CPU POWERLINK window, and click Insert in the pop-up menu.
18. Select "MAC00-ELx", situated under POWERLINK devices, and click Next.
19. Enter the node ID of the device (set earlier with MacTalk) and optionally a name, and click Next.
20. The "MAC00-ELx" should now be visible in the physical view of the project explorer window.

Building project and transfer to PLC

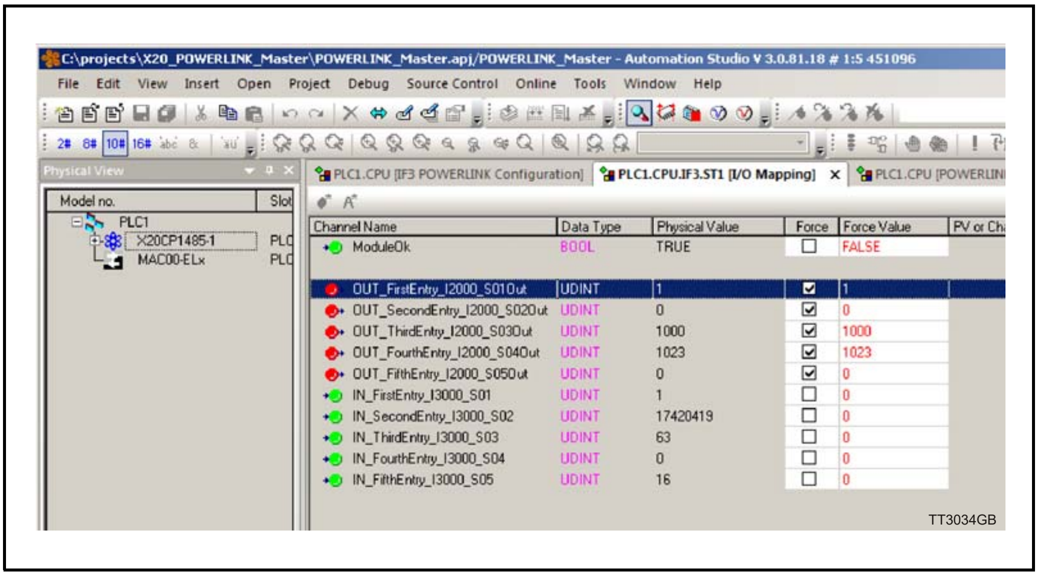
21. Select Build configuration in the Project menu.
22. When the build is finished then click the Transfer button.
23. There may appear a warning. Just ignore and click OK.

Investigating cyclic data

24. Right click "MAC00-ELx" in the physical view of the project explorer window and click Open I/O Mapping.
25. In the View menu click Monitor.
26. You should now be able to see the cyclic I/O registers like in the below picture.
27. If Force is checked for the cyclic outputs, then it's possibly to set register values in the Force Value column that is transferred to the motor.

Start motor

28. If the default register settings is not changed it is possibly to start motor by entering values in the Force Value column.
29. Enter 1023 in OUT_FourthEntry (Torque = 300%).
30. Enter 1000 in OUT_ThirdEntry (477 RPM if MAC140).
31. Enter 1 in OUT_FirstEntry (Mode = Velocity).



5.4 Ethernet POWERLINK objects

5.4.1 Process data objects

PDO's (Process Data Objects) are used for cyclic transfer of time-critical process data between master and slaves. Tx PDOs are used to transfer data from the slave to the master and Rx PDOs to transfer data from the master to the slave.

PDO 21

PDO 21 is fully user configurable. There is one receive PDO and one transmit PDO.
It is possibly to set up five, 32 bit registers in each direction.
The setup is done with MacTalk or via SDO object 0x2011 subindex 16-31. It requires a save in flash and a power cycle before the new configuration are used. If the configuration of the PDO's, is not altered by the user, the MAC00-ELx uses the default mapping shown in the tables below.

If module registers is placed in cyclic R/W, then the register number has to be calculated as follows:

Register number = 65536 x sub index.
Example: module command (sub-index 15) = 65536 x 15 = register **983040**

When module registers (register numbers above 65535) are chosen, they **have** to be placed **after** the motor registers in the list of cyclic registers.

NB! If an index is set to zero (No selection), then the following indexes is discarded. Thereby computing resources in the drive are released, which makes much faster cycle times possibly. Please see next paragraph.

Default registers in transmit PDO 21 (Slave > Master) / Read words in MacTalk

Object index	Register no.	Motor register short	Motor register description
0	2	MODE_REG	Operating mode
1	10	P_IST	Actual position
2	12	V_IST	Actual velocity
3	169	VF_OUT	Actual torque
4	35	ERR_STAT	Status bits

The motor registers 35, 36, and 211 should NOT be inserted in the cyclic write list, as this may give unpredictable results. For clear of errors, reset of motor etc. please insert the module command register (=983040 in Mactalk) in the cyclic write list and send commands this way. For a list of commands for the module command register please *Register Overview, page 254*.

Default registers in receive PDO 21 (Master > Slave)

Object index	Register no.	Motor register short	Motor register description
0	2	MODE_REG	Operating mode
1	3	P_SOLL	Target position
2	5	V_SOLL	Maximum velocity
3	7	T_SOLL	Maximum torque
4	-	-	-

Please notice: Even though all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

5.4 Ethernet POWERLINK objects

5.4.2 Minimum cycle time

The minimum cycle time is the minimum amount of time between each cyclic request (PDO) on the Ethernet.
If operating with values lower than those listed, data loss will occur.

No. of motor registers transmitted in each direction	Motor series MAC050 to MAC141	Motor series MAC400 to MAC4500	Motor series MIS/MILxxx
1/1	4mS *	360µS *	360µS *
2/2	8mS *	395µS *	395µS *
3/3	12mS *	430µS *	430µS *
4/4	16mS *	465µS *	465µS *
5/5	20mS *	500µS *	500µS *
6/6	24mS *	535µS *	535µS *
7/7	28mS *	570µS *	570µS *
8/8	32mS *	605µS *	605µS *

* The minimum cycle times, is only valid if not sending any SDO requests while in any operating mode. MODULE registers can be appended as the last registers in the list, at no extra timing cost. If motor register 35 is not in the list it will be added internally anyway, and has to be added to the minimum cycle time with 2.0ms if MAC050-MAC141, and with 30μs if MAC400-MAC4500 or MIS/MILxxxxxELxxxx.

5.4.3 Service Data Objects

Service Data Objects (SDOs) are mainly used for transferring non time-critical data, for example, identification, configuration and acyclic data.

5.4.4 Object Dictionary

An important part of the protocol is the Object Dictionary, which is different objects specifying the data layout. Each object is addressed using a 16-bit index and possibly a sub index. There are some mandatory objects and some manufacturer specific objects. The objects in the Object Dictionary can be accessed with SDO services.

Mandatory objects:

Name	Index (hex)	Sub Index	Data Type	Read only	Default	Description
Device type	1000		UNSIGNED32	X	0x0	Contains information about the device type.
Error Register	1001		UNSIGNED8	X		This is the mapping error register, and it is part of the emergency object. If some of the sub index are high, an error has occured.
		0				Generic error. Mandatory
		1				Current
		2				Voltage
		3				Temperature
		4				Communication (Overrun)
		5				Device profile specific
		6				Reserved
		7				Manufactor specific
Identity object	1018		IDENTITY	X		Contain general information about the module
		0	1..4	X	0x04	Number of entries. Mandatory
		1	UNSIGNED32	X	0x0117	Vendor ID, contains a unique value allocated to each manufactor. 117h is JVLs vendor ID. Mandatory.
		2	UNSIGNED32	X	0x0200	Product Code, identifies a specific device version. The MAC00-EL4/-EL41 has the product code 200h
		3	UNSIGNED32	X	-	Revision number.
		4	UNSIGNED32	X		Serial number

5.4.5 Manufacturer specific objects.

The manufacturer specific objects, provides access to all module registers, and all motor registers, as well as a module command object.

	Index (hex)	Sub Index	Type	Read only	Default	Description
Module command	2010	0	UNSIGNED32			Module command object. See possible commands below.
Module parameters	2011	0	UNSIGNED8	X	63	Subindex count
		1	UNSIGNED32	X	-	Access to module register N
Motor parameters	2012	0	UNSIGNED8	X	254	Subindex count
		N	UNSIGNED32			Access to the motor parameter n

Note:
Module parameters are not automatically saved to permanent memory after a change. The parameters can be saved permanently by applying a "Save parameters to flash" command afterwards.

5.4.6 Object 0x2010 - Subindex 0

This object is used for sending commands to the module and is write only. The possible commands are listed in the table below.
This object is used for sending commands to the module and is write only. It is analogue to writing to object 2011 subindex 15.
See the description *Register Descriptions., page 255*.

5.4.7 Object 0x2011

The module registers is mapped to object 0x2011. The subindex 3, 6-31 is R/W, the rest is read only.
The register numbers are used as sub indexes in the object. See register descriptions in chapter 8 - *Register Descriptions., page 255*

5.4.8 Object 0x2012

Object 0x2012 are for acyclic view or change of motor registers.

Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 or
Motor registers MAC400 - 4500, page 301 or
Motor registers MISxxx, page 320.

5.5 Network Management Services

Ethernet POWERLINK Network Management (NMT) is node oriented and follows a master/slave relationship.
The MAC00-ELx or MIS/MILxxxxxxELxxxxx is administered as an NMT slave by the master. Ethernet POWERLINK defines five categories of NMT services:

- NMT State Command Services
- NMT Managing Command Services (not supported)
- NMT Response Services
- NMT Info Services (not supported)
- NMT Guard Services (not supported)

NMT State Command Services
The MN controls the state of the CN via NMT State Command Services. See section Ethernet POWERLINK state machine for more information.

NMT Response Services
NMT Response Services are used by the MN to query NMT information from the CN, such as current state, error and setup data. Ethernet POWERLINK specifies the following NMT Response Services:

- NMT State Response
- IdentResponse
- StatusResponse

Via NMT State Response service, the CNs signals their states to the MN. IdentResponse Service is used by the MN to identify configured but unrecognized CNs at system startup or after loss of communication. See Appendix: IdentResponse Frame for more information. The StatusResponse Service is used by the MN to query the current status of CNs that is not communicating isochronously. It is used for error signaling in runtime. If an error occurs, the EN (Error New) flag in the PRes frame is toggled. This notifies the MN that an error has occurred and the MN polls the CN for a StatusResponse that includes error information.

5.6 XML Device Description File

XML Device Description Files (XDD) are XML files that specify the properties of the slave device for the Ethernet POWERLINK master (MN). The description files contain information on the supported communication objects. XDD files for JVL Drives are available through your local JVL representative and http://www.jvl.dk.

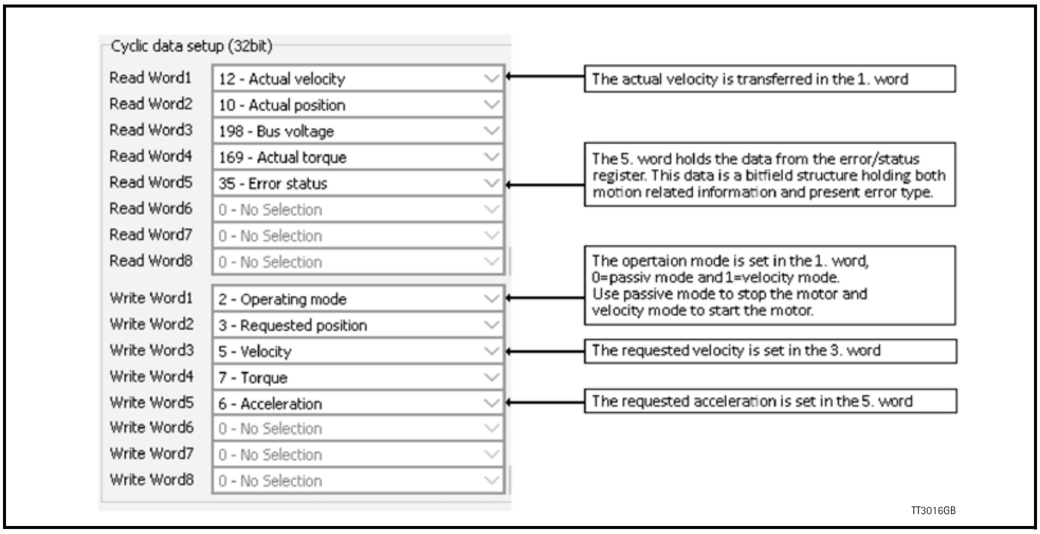
5.7 Examples

5.7.1 Running Velocity control

To use the JVL motor in velocity mode the following registers are basically of interest.

1. "Mode" - Mode register register 2
2. "V_SOLL" - Velocity register 5
3. "A_SOLL" - Acceleration register 6
4. "Error/Status" - Error and status register 35

So, to control these registers the cyclic data needs to configured.
From MacTalk the setup is configured as this.



With the settings illustrated above we initiate the velocity mode by writing 0x1 to the first word-value, this is velocity mode.

From the Master the registers is accessed using the PDO21 and accessing the registers R/W on words 1-5.

Since different PLC's have different methods of implementation the basic steps is described in the following. (Constant values valid for MAC800, for other motors, please consult the motor manual)

1. Set the needed velocity. V_SOLL = V x 2.77 [rpm]
   Ex. We need the motor to run with a constant speed of 1200 RPM. So, V_SOLL = 1200/2,77 = 433 cnt/smp
2. Set the needed acceleration. A_SOLL = A x 271 [RPM/s^2]
   Ex. We need the motor to accelerate with 100000 RPM/s^2 so, A_SOLL = 100000/271 = 369 cnt/smp^2.
3. Now set the motor into velocity mode and thereby activate the motor.
   Ex. The motor needs to be activated by setting it into velocity mode, so we need to set the mode register to the value 1. Mode = 1 which is velocity mode, now the motor will use the acceleration and the velocity just configured.

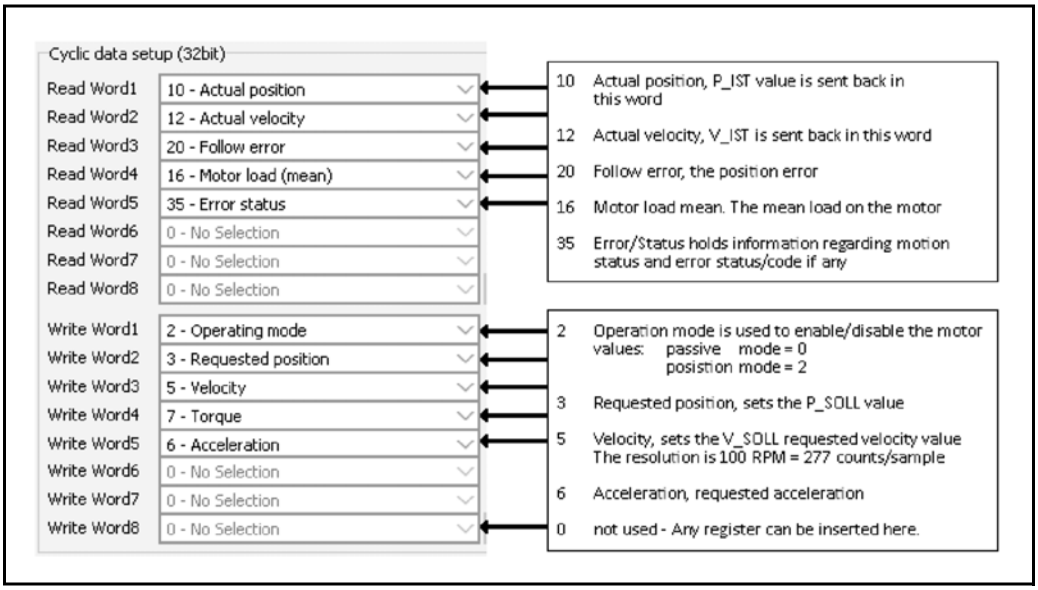
Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 or
Motor registers MAC400 - 4500, page 301 or
Motor registers MISxxx, page 320

5.7.2 Running Position control

Running the motor in position control requires that the mode register is set for position control. The following registers is of particular interest when position mode is used.

1. "Actual position" -P_IST, register 10
2. "Actual velocity" -V_IST, register 12
3. "Follow error" - The actual position error, register 20
4. "Motor load mean" - average motor load, register 16
5. "Error/Status" -register 35
6. "Requested position" -P_SOLL, register 3
7. "Requested velocity" -V_SOLL, register 5
8. "Requested acceleration" -A_SOLL, register 6

In this mode the position is controlled by applying a requested position to the "P_SOLL" -register and the actual position is monitored in the "P_IST" register. The V_SOLL and A_SOLL registers sets the velocity and acceleration used when positioning occurs.



5.7.3 General considerations

The register 35 in the motor holds information on the actual error/status. So it is crucial that this register is configured in the cyclic data and thereby obtained and monitored in the Master. In case of an error situation the motor will stop and the cause will be present in the register 35 and hence in the I/O -data.
This register also holds information on the motion status such as:

- In position, bit 4
- Accelerating, bit 5
- Decelerating, bit 6

Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 and *Motor registers MAC400 - 4500, page 301* or *Motor registers MISxxx, page 320*.

The JVL motor is basically put into a working mode and into a passive mode where the motor axle is de-energized, by setting register 2 into either 0 = "passive mode" or into one of the supported modes.
(continued next page)

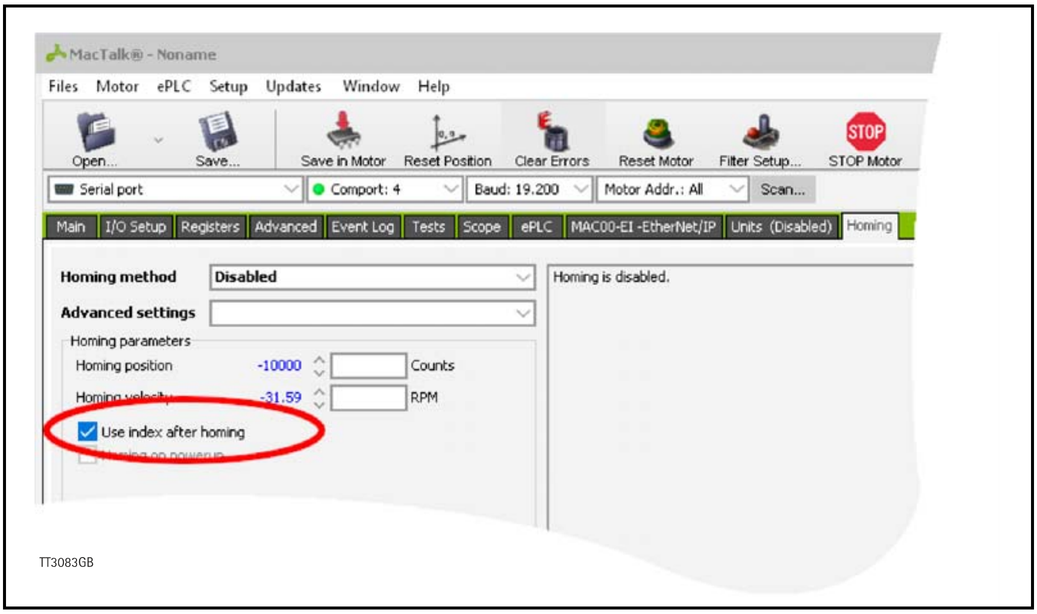
Example.

1= "Velocity mode" / 2= "Position mode" / etc.

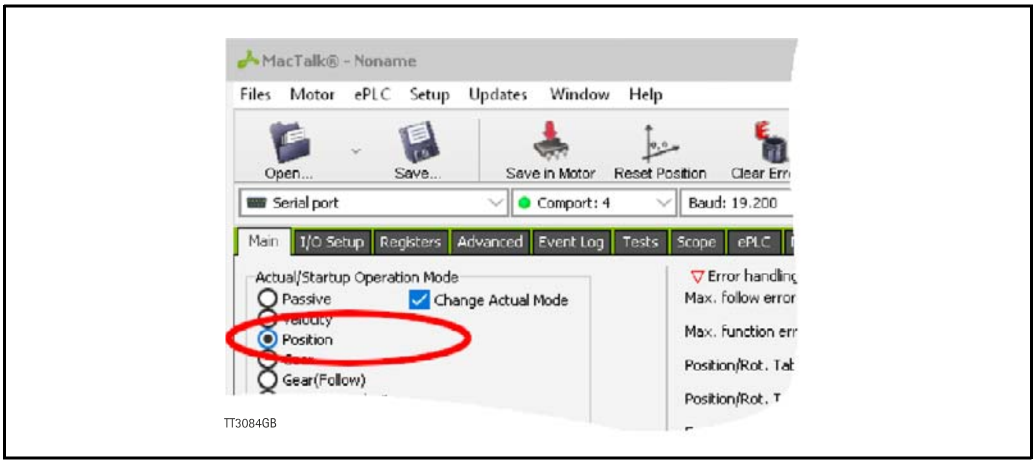
So in order to Stop or Start the motor this register can be supported in the I/O data or by sending an SDO message.

5.7.4 Homing using only cyclic I/O (JVL profile).

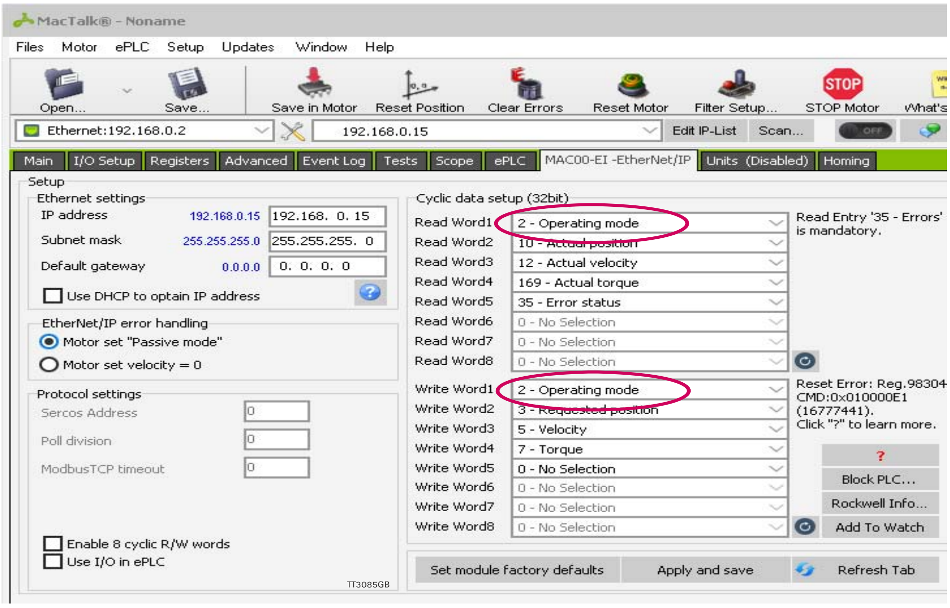
When doing a homing (Zero search), with only cyclic I/O, some preconditions have to be met: Zero search position, zero search velocity and zero search torque (torque only for MAC motors) has to be set in MacTalk in the "Main" tab, and saved in flash in the motor once and for all.



Startup mode should be set to position, for the motor to stay in position after the homing sequence. And this setting should also be saved in flash.



Register 2 (Operating mode) has to be present in BOTH the cyclic read words and cyclic write words.



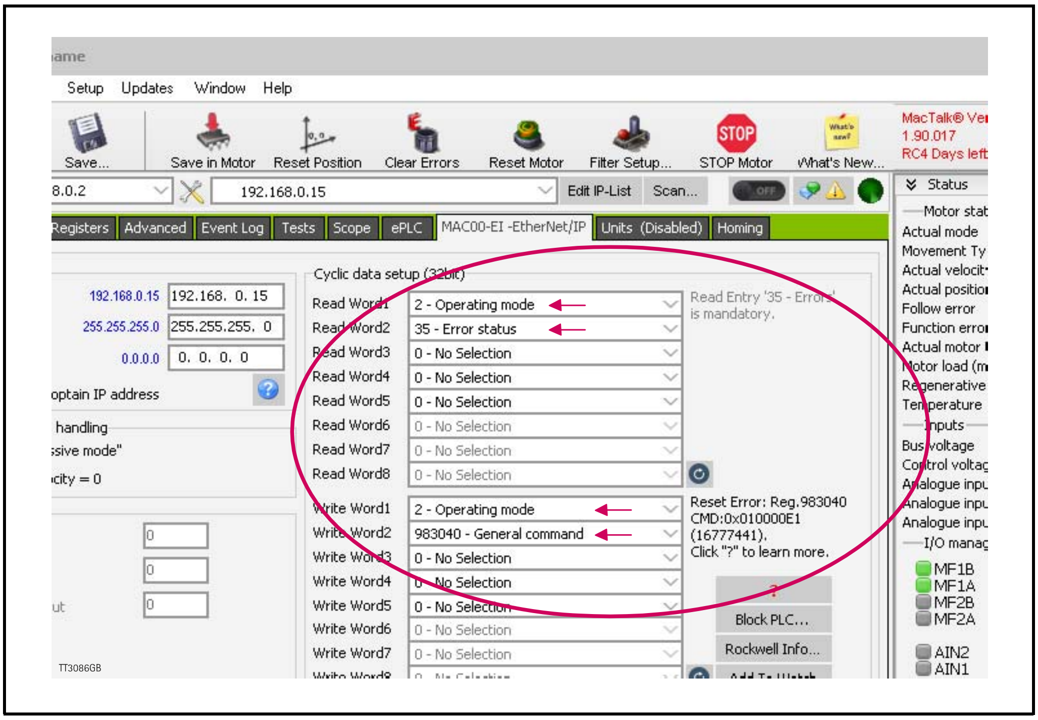
Procedure in the PLC:

- Treat the transmitted Register 2 as "Requested_Mode" and the received register 2 as "Actual_Mode".
- When homing is wanted, set the "Requested_Mode" to one of the values 12, 13 or 14 depending of the requested homing mode (12 = Torque based zero search mode (only MAC motors). 13 = Forward/only zero search mode. 14 = Forward+backward zero search mode (only MAC motors) .). For a comprehensive description of the homing modes, refer to the general MAC motor manual - LB0047-xxGB.
- Observe that the "Actual_Mode" is changing to the homing mode. Now the module is blocking cyclic writes TO the motor. Cyclic reads is still active.
- Wait for register 35 "Error status" bit 4 to be active =IN_POSITION. (Indicates that homing is finished).
- Then change "Requested_Mode" to whatever needed. The blocking of cyclic writes to the motor is then released by the module.

5.7.5 Relative positioning.

There are a number of ways to do relative positioning, but the one explained here is very simple, and can be used with a constant distance, or exchangeable distance, to move every time it is requested.

Preconditions:
Place the module command register (register 983040 in MacTalk) in the cyclic write list.
The cyclic setup, could for example look like this:



Procedure in the PLC:

1. Set up register P7 in motor to requested relative offset.
2. Make sure one net cycle has passed, so P7 resides in the motor.
3. Issue command 0x800000F1 (0x80000071 if the MIS/MILxxxxxxELxxxx motor is used) in module command register (register 983040 in MacTalk).
4. Make sure one net cycle has passed, so command is interpreted by the motor.
5. Set module command register to zero. This will prepare the Ethernet module for new commands.
6. If needed then monitor register 35 (Error status): When bit 4 is set (in position), then the move is finished.
7. When a new relative move is requested, go to step 3.

You may also have the P7 register in the cyclic write list, thereby enabling easy change of the relative distance to move.

6 PROFINET Users Guide

6.1 Introduction to PROFINET IO

MAC - PROFINET IO Module
Type:
MAC00-**EP**4 (shown) or
MAC00-**EP**41 (extended I/O)

To be used in following servo products:
MAC50, 095, 140 and 141
MAC400 and MAC402
MAC800
MAC1500 and MAC3000

MIS - PROFINET IO motors.
Type:
MISxyyzz**EP**xx85 or
MILxyyzz**EP**xx85

6.1.1 Overview

PROFINET IO is a fieldbus protocol that enables communication between programmable controllers and distributed field devices in Ethernet networks.
PROFINET IO uses traditional Ethernet hardware and software to define a network that structures the task of exchanging data, alarms and diagnostics with Programmable Controllers and other automation controllers.
PROFINET IO can be thought of, as PROFIBUS on Ethernet. The protocol classifies devices into IO controllers, IO supervisors and IO devices, which have a specific collection of services.
PROFINET IO uses three different communication channels to exchange data.

- The standard UDP/IP and TCP/IP channel is used for parameterization and configuration of devices and for acyclic operations.
- The Real Time (RT) channel is used for cyclic data transfer and alarms.
- The third channel, Isochronous Real Time (IRT) channel, is used e.g. in some motion control applications , and is in JVL motors coupled to the PROFIdrive profile, which can be selected by use of the Mactalk tool in the majority of motor types.

PROFINET IO devices are structured in slots, and sub-slots, which can contain modules and sub-modules correspondingly. Devices can have almost any number of slots and sub-slots and they can be virtual or real. Device specific data is represented in slot 0, module and sub-module specific data in subsequent slots and sub-slots. One of the benefits of PROFINET IO is the diagnostics and alarm mechanism. Every module and sub-module provides alarm data to the IO controller using the cyclic channel. Diagnostic data can be read non-cyclically from the device by using record data. Properties and services of a PROFINET IO device are described in a GSD file that is written in GSDML (General Station Description Markup Language).

6.1 Introduction to PROFINET IO

The GSD file describes the device specific modules and the method of assigning modules and sub-modules to predefined slots and sub-slots.
There is no theoretical limit for the amount of connected nodes in PROFINET IO network, but in practice, the programmable controllers and number of available network addresses limits the size. The PROFINET IO protocol is specified in the IEC standards 61158 and 61784.
Further information can be obtained from www.PROFINET.com.

6.1.2 Definitions and abbreviations

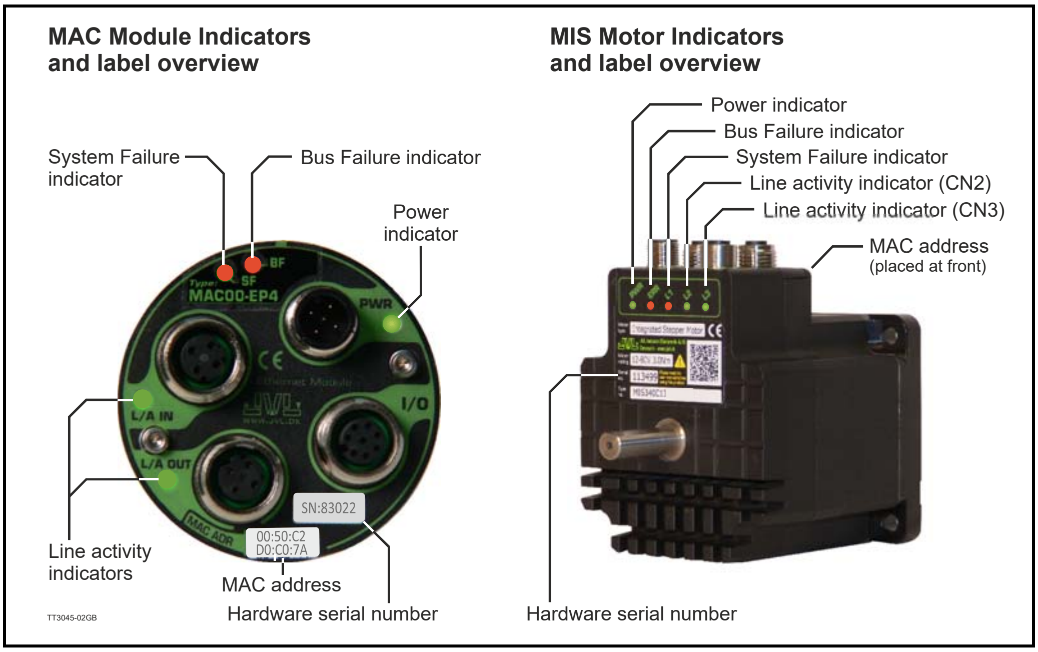
Following general used terms are useful to know before reading the following chapters.

100Base-Tx	100 MBit Ethernet on twisted pairs.
AC	Application Class
Acyclic communication	Communication in which messages are sent once per request.
Cyclic communication	Communication in which process data are sent cyclically at predefined intervals.
DAP	Device Access Point.
DCP	Discovery and Configuration Protocol.
DO	Drive Object
GSD	General Station Description. Device description file in a specified form. Each device (active & passive stations) on PROFINET has to have its own GSD File. GSD files in PROFINET are written in GSDML.
GSDML	General StationDescription Markup Language - is a XML based language used for the device description file.
IO-Controller	Control system with bus initiative. In PROFINET IO terminology, IO-controllers are also called master stations.
IOPSIO	Provider State (state of the provider of cyclic IO data).
IOCSIO	Consumer State (state of the consumer of cyclic IO data).
IPInternet	Protocol - IP address ~ the logical address of the device, which is user configurable.
MAC	Media Access Controller - MAC address ~ the hardware address of the device.
PAP	Parameter Access Point
PZD	Process Data
ST	Standard Telegarm
TCP	Transfer Control Protocol (an IP based protocol used widely on the internet).
UDP	User Datagram Protocol (an IP based protocol used widely on the internet).

6.2 Commissioning

6.2.1 Indicator description

The LED's are used for indicating states and faults of module. There is one power LED, two link/activity LED's (one for each Ethernet connector), and 2 status LED's.



LED indicator descriptions - Covers both MAC and MIS/MIL (MIx).

LED Text MAC / MIx	**Colour**	**Constant off**	**Constant on**	**Blinking**	**Flickering**
L/A IN / L2	Green	No valid Ethernet connection.	Ethernet is connected.	-	Activity on line
L/A OUT L3	Green	No valid Ethernet connection.	Ethernet is connected.	-	Activity on line
SF / L1* (se note below)	Red	No System failures	System failures	DCP signal service is initiated	-
BF / ERR	Red	No Bus failures	Bus failures	No data exchange	-
PWR	Red/ Green	Power is not applied.	Power is applied to both motor and module. **MIS17x/MIS23x**: The LED will lit red constantly if the power supply is too low.	-	Power is applied to module but no communication with motor

Notes:
Blinking: Flashing with equal on and off periods of 200ms (2.5Hz). **Single flash** : Repeating on for 200ms and off for 1s. **Double flash** : Two flashes with a period of 200ms followed by 1s off period. **Triple flash** : Three flashes with a period of 200ms followed by 1s off period. **Flickering** : Rapid flashing with a period of approx. 50ms (10 Hz).

* When powering on without PLC connected the status of the SF/L1 LED is undefined.

6.2 Commissioning

6.2.2 Mechanical installation

The network cables must be connected to the two M12 connectors (marked “L/A IN” and “L/A OUT”) at the MAC module and “CN2” and “CN3” at the MIS/MIL motors.
The cable from the IO CONTROLLER is connected to either of the two ports.
In the line topology, if there are more slave devices in the same line, the next slave device is connected to the second port.
If there is a redundant ring, the second port of the last slave device is connected to the second port of the IO CONTROLLER. If using the topology principle there has to be an exact match between the ports physically connected and the ports wired in the "Topology view" in TIA portal.
Standard CAT 5 STP cables can be used. It is not recommended to use UTP cables in industrial environments, which is typically very noisy.

6.2.3 Network configuration

To enable communication through the Ethernet network, the module needs a valid IP address. This is done either by MacTalk, see the quick start guide or is done by DCP.
In the PROFINET IO protocol, also a device name is required to identify the drive.
IO-controllers and some configuration tools have a protocol called Discovery and Configuration Protocol (DCP) for assigning the IP address and the device name.

The IP address shown in MacTalk, is only a power on default. When a PLC is connected the actual used IP can be another one configured by the PLC.

PROFINET IO and DCP
When the module is initialized, the IP address is transferred to the PROFINET IO communication stack. If there is a need to change the IP address it should be done with a DCP tool (like Siemens Step7). If some of the other methods are used to change the IP address, the module must be restarted to enable any changes.

6.2.4 Configuring the system

After the MAC or MIS/MIL motor has been mechanically and electrically installed according to the instructions in previous chapters, and has been initialized by the drive, the master station must be prepared for communication with the module. Configuration of the master station requires a type definition (GSD) file. In PROFINET IO the GSD file is written in XML based language called GSDML. MAC00-EPx has a GSD file, which is available from www.jvl.com or your local JVL representative.
The filename is ***GSDML-V2.33-JVL-MAC_MIS-20190128.xml***.
The GSD file describes vendor specific features of the module. Please refer to the master station documentation for more information on activating PROFINET IO devices with GSD file.

6.2.5 PROFINET IO in the MAC or MIS/MIL motors

The JVL Profinet uses slots 0 and 1. Slot 0 does not have any sub-slots and the DAP module attached to it represents the device itself.
Other functional modules and sub-modules, which are described in the GSDML file, can be assigned to slot 1 and its sub-slots:

- Slot 0 = Device access point (DAP)
- Slot 1, sub-slot 1 = Vendor object
- Slot 1, sub-slot 1 = Acyclic parameter access

The MAC or MIS/MIL motors provides the following services:

- Cyclic messaging
- Acyclic parameter access mechanism
- Identification & Maintenance functions (I&M)

6.2.6 Dynamic IP and naming

With DCP (Discovery and Configuration Protocol) the IP address and 'Name of station' in the device can be changed on the fly, by the PLC. Therefore the IP address shown in MacTalk is only the power up default, and may not be the actual used IP address after the PLC has established communication.

If checking the “Power up with blank name of station” (factory default) in MacTalk and save the configuration in flash, then the MAC00-EPx / MIS/MILxxxxxxEPxx will always start up without a station name. This enables the possibility of having new devices on stock, and if needed exchange them in the machine without any setup, as the PLC can be programmed to automatically assigning the correct name, when it finds a device without name.

This feature depends on the PLC knowing the topology, so every connection has to be wired in the "Topology view" in TIA portal.

6.2 Commissioning

6.2.7 Quick start guide

This section describes the steps to configure the Siemens ET200S PLC and TIA Portal v11 or newer, so it can be used to control the drive.

Set IP address

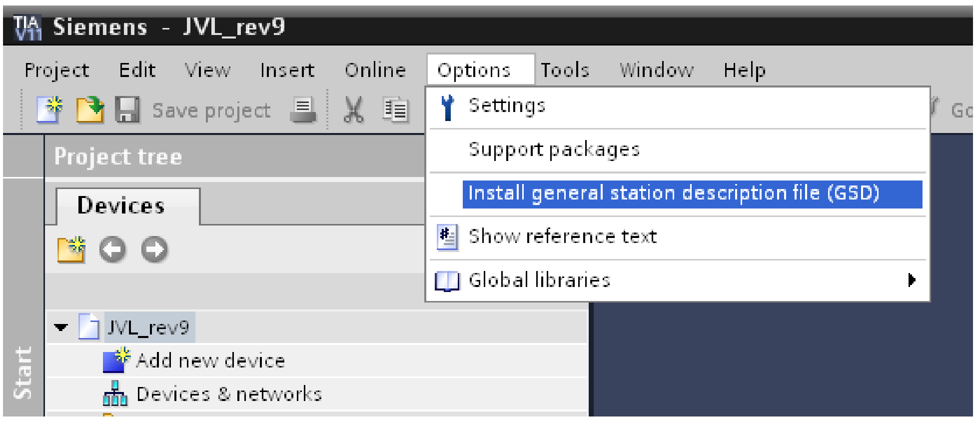
1. If having a MAC motor then Connect the RS232 communication cable, and if having a MIS/MIL motor then connect the RS485 communication cable.
2. Apply power to the motor, and make sure the PWR LED is lit.
3. Open MacTalk and select the "MAC00-EP (PROFINET)" tab.
4. Change the IP address, to one suitable for the network.
5. Press "Apply and save".

Installation

6. Connect an Ethernet RJ45-M12 cable to one of the interfaces on the ET200S and to "L/A IN" and "L/A OUT" at the MAC module and "CN2" and "CN3" at the MIS/MIL motors.
7. Connect power to the ET200S, and Ethernet patch cable from the PC with Siemens TIA Portal v11 installed to the ET200S PLC.
8. Make sure power is applied to all devices.

Add the GSD file (contains info on the capabilities of the device)

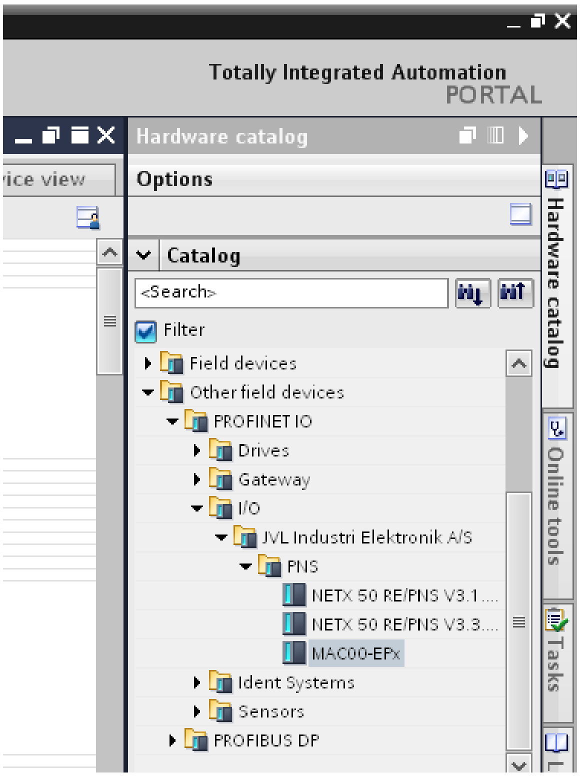
9. In the Options menu of TIA Portal V11, select Install general station description file (GSD).
10. In the "Install general station description file" window find and select the "GSDML-V2.33-JVL-MAC_MIS-20190128.xml" file, and click Install.
11. Follow the on screen instructions.



TT3050-01GB

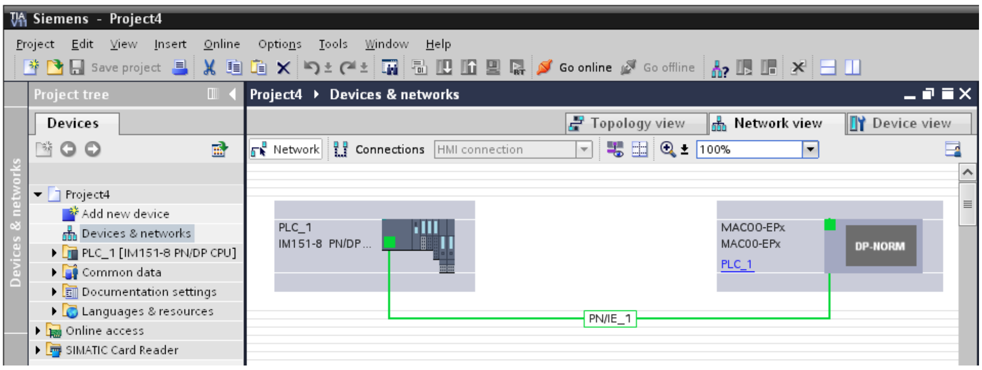
PLC configuration

12. Create a new project in TIA Portal v11 for your PLC, or open an existing project. See Siemens documentation for more information.
13. In the **Hardware catalog** under **Other field devices** / **PROFINET I/O** / **I/O** / **JVL Industri Elektronik A/S** / **PNS** should **MAC00-EPx** reside. See figure at next page.



TT3051-01GB

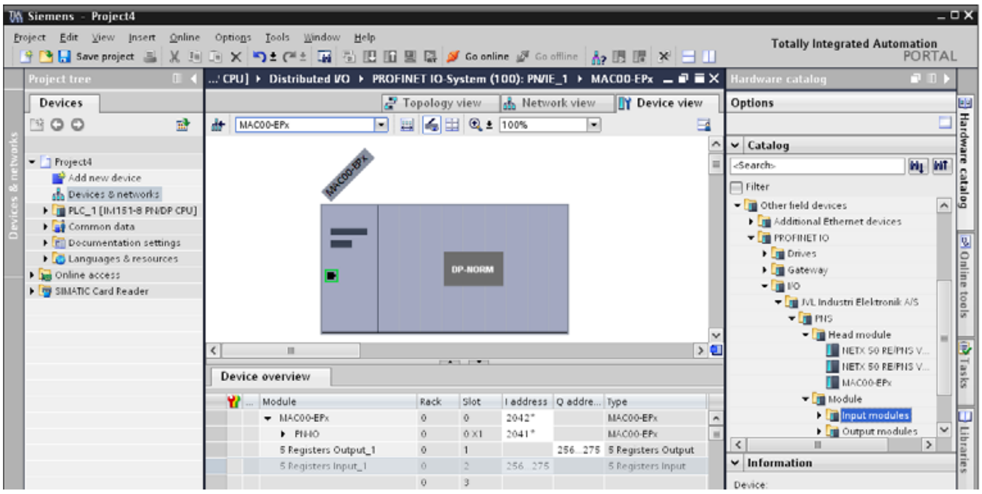
14. Drag and drop the MAC00-EPx to the Network view.
15. Also add your PLC to the Network view (see also Siemens documentation for further info).
16. If using external switches then these must be added too (see Siemens documentation for further info). If not continue to the next step.
17. Connect the two devices, by dragging a line between the small green boxes in each device, and it should now look like below.



TT3052-01GB

Associating with the cyclic data

18. Drag the "**8 registers input**" and "**8 registers output**" from the **hardware catalog** under **Other field devices** / **PROFINET I/O** / **I/O** / **JVL Industri Elektronik A/S** / **PNS** / **Module** / **Input modules** and **Output modules**, and drop them in the **Device overview** of the MAC00-EPx. See illustration next page.



TT3053-01GB

19. It should now be possible to make a PLC application using cyclic communication to the 8 registers input and output (see section 6.3.1 for setting up those with Mactalk).

    There is also an example on the web page www.jvl.dk in the download section, named 'JVL_PN_ex1.zip' which can be downloaded and unzipped. This example is made for MAC140, but can easily be changed to work with MAC400-MAC4500 or the MIS/MIL motors.

6.3 PROFINET objects

6.3.1 Process data

Process Data (PZD) are used for cyclic transfer of time-critical process data between master and slaves, such as position, velocity, torque etc. Transmit PZD are used to transfer data from the slave to the master and receive PZD to transfer data from the master to the slave.
The JVL process data is fully user configurable. It is possibly to set up eight, 32 bit registers in each direction. The setup is done with MacTalk or via parameter object 0x11 sub-index 16-31. It requires a save in flash and a power cycle before the new configuration are used. If the configuration of the PZD, is not altered by the user, the JVL PROFINET module uses the default mapping shown in the tables below. It is mandatory to have the error/status register (register 35) as one of the slave to master registers. If not the motor will overrule the configuration and place register 35 anyway.

If module registers is placed in cyclic R/W, then the register number has to be calculated as follows:

Register number = 65536 x sub index.
Example: module command (sub-index 15) = 65536 x 15 = register **983040**

When module registers (register numbers above 65535) are chosen, they **have** to be placed **after** the motor registers in the list of cyclic registers.

NB! If an index is set to zero (No selection) then the following indexes is discarded. Thereby computing resources in the drive are released, which makes much faster cycle times possibly. Please see next paragraph.

Default registers in transmit PZD (Slave > Master) **- Only MAC-EPx**

Object index	Register no.	Motor register short	Motor register description
0	2	MODE_REG	Operating mode
1	10	P_IST	Actual position
2	12	V_IST	Actual velocity
3	169	VF_OUT	Actual torque
4	35	ERR_STAT	Status bits
5	-	-	-
6	-	-	-
7	-	-	-

The motor registers 35, 36, and 211 should NOT be inserted in the cyclic write list, as this may give unpredictable results. For clear of errors, reset of motor etc. please insert the module command register (=983040 in Mactalk) in the cyclic write list and send commands this way.
For a list of commands for the module command register please refer to *Register Overview, page 254*.

Continued next page

6.3 PROFINET objects

Default registers in receive PZD (Master > Slave) **- Only MAC-EPx**

Object index	Register no.	Motor register short	Motor register description
0	2	MODE_REG	Operating mode
1	3	P_SOLL	Target position
2	5	V_SOLL	Maximum velocity
3	7	T_SOLL	Maximum torque
4	-	-	-
5	-	-	-
6	-	-	-
7	-	-	-

Please notice: Even though all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

Default registers in transmit PZD (Slave > Master) **- Only MIS/MILxxxxxxEPxx**

Object index	Register no.	Motor register short	Motor register description
0	2	MODE_REG	Operating mode
1	10	P_IST	Actual position
2	12	V_IST	Actual velocity
3	35	ERR_STAT	Error bits
4	36	WARN_BITS	Warning bits
5	-	-	-
6	-	-	-
7	-	-	-

Default registers in receive PZD (Master > Slave) **- Only MIS/MILxxxxxxEPxx**

Object index	Register no.	Motor register short	Motor register description
0	2	MODE_REG	Operating mode
1	3	P_SOLL	Requested position
2	5	V_SOLL	Requested velocity
3	6	A_SOLL	Requested acceleration
4	-	-	-
5	-	-	-
6	-	-	-
7	-	-	-

The MIS/MIL motor registers 24, 35 and 36 should NOT be inserted in the cyclic write list, as this may give unpredictable results. For clear of errors, reset of motor etc. please insert the module command register (=983040 in Mactalk) in the cyclic write list and send commands this way. For a list of commands for the module command register please refer to *Register Overview, page 254*

6.3.2 Minimum cycle time

The minimum cycle time is the minimum amount of time between each cyclic request on the Ethernet. If operating with values lower than those listed, data loss will occur.

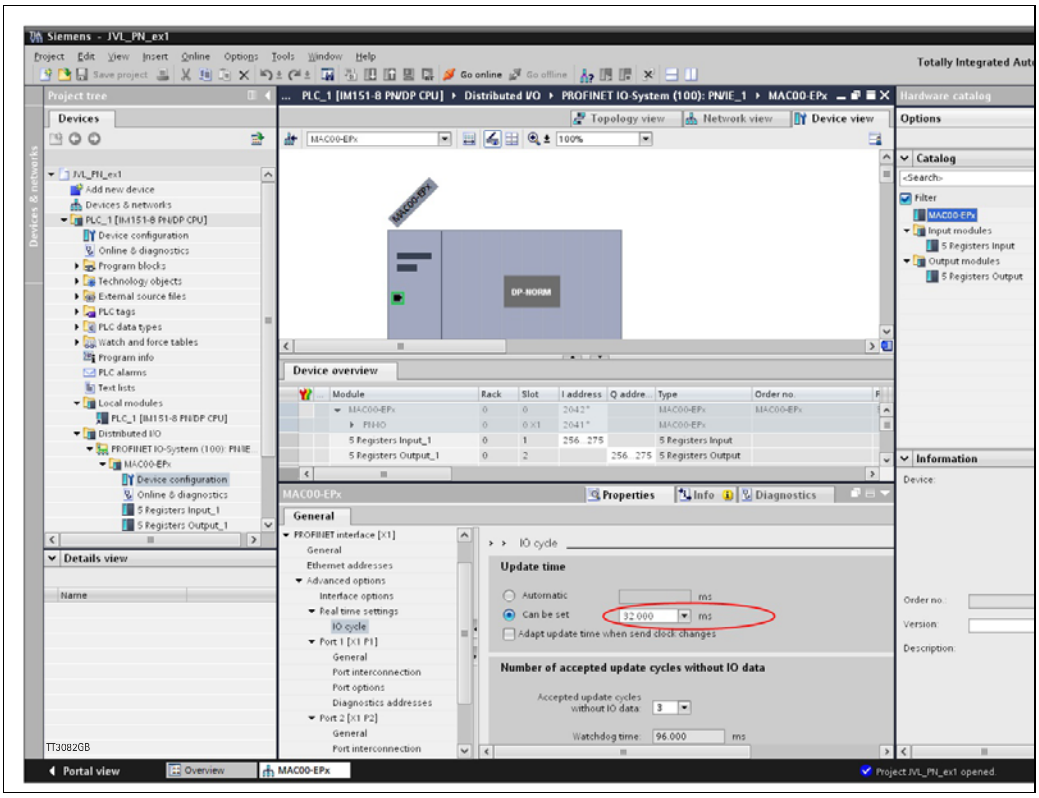
	Motor series		
Number of motor registers transmitted in each direction	**MAC050 to MAC141**	**MAC400 to MAC4500**	**MIS/MILxxx**
1/1	4ms *	1ms *	1ms *
2/2	8ms *	1ms *	1ms *
3/3	12ms *	1ms *	1ms *
4/4	16ms *	1ms *	1ms *
5/5	20ms *	1ms *	1ms *
6/6	24ms *	1ms *	1ms *
7/7	28ms *	1.1ms *	1.1ms *
8/8	32ms *	1.2ms *	1.2ms *

* The minimum cycle times, is only valid if not sending any acyclic requests while in any operating mode. MODULE registers can be appended as the last registers in the list, at no extra timing cost. Motor register 35 shall be in the cyclic read list, as it is also used internally.

6.3 PROFINET objects

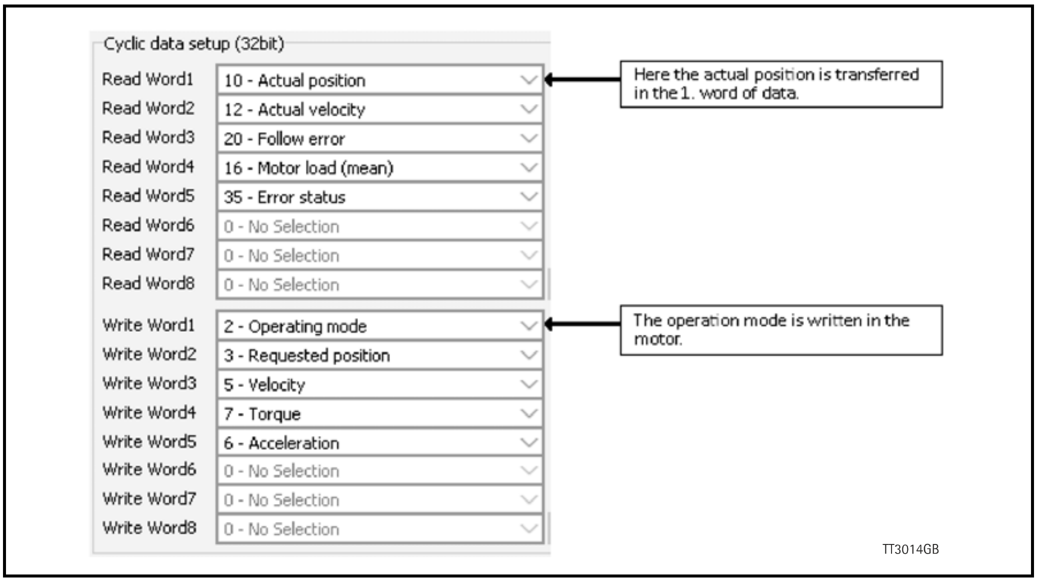
Changing cycle time in TIA Portal V11-V15

In the TIA Portal V11-V15 the cycle time is set up in the properties of each device, under Real time settings / IO Cycle. Please see the picture below. It is done in a similar way in Step7, but this is not shown.



Accessing process data

The PZD is done by setting up the motor registers you want to use with MacTalk or with acyclic parameter access to object 0x11 subindex 16-31. In MacTalk the process data is configured on the PROFINET tab, see below. After change of the registers, remember to press the Apply and save button.



6.3.3 Parameter objects.

The parameter objects provide access to all module registers, and all motor registers, as well as a module command object. The objects in the list can be accessed with acyclic services, *Accessing parameter objects, page 151*.

	Object (hex)	Sub Object	Type	Read only	Default	Description
Module command	0x10	0	UNSIGNED32			Module command object. See possible commands below.
Module parameters	0x11	1	UNSIGNED32	X		Access to module register N
Motor parameters	0x12	0-255	Register dependant	X	254	Access to the motor parameter n (register)

Note: Module parameters are not automatically saved to permanent memory after a change. The parameters can be saved permanently by applying a "Save parameters to flash" command afterwards.

Object 0x10 - Subindex 0

This object is used for sending commands to the module and is write only. It is analogue to writing to object 0x11 subindex 15.
See the description *Register Descriptions., page 255*.

Object 0x11

The module registers is mapped to object 0x11. The subindex 3-31 is R/W, the rest is read only.
The register numbers are used as sub indexes in the object.
See register descriptions in chapter 8 - *Register Descriptions., page 255*

Object 0x12

Object 0x12 are for acyclic view or change of motor registers.
Please find a complete list of register descriptions in the appendex.
Motor registers MAC050 - 141, page 292 or
Motor registers MAC400 - 4500, page 301 or
Motor registers MISxxx, page 320.

Object 0x13 (only applicable to MAC400-4500).

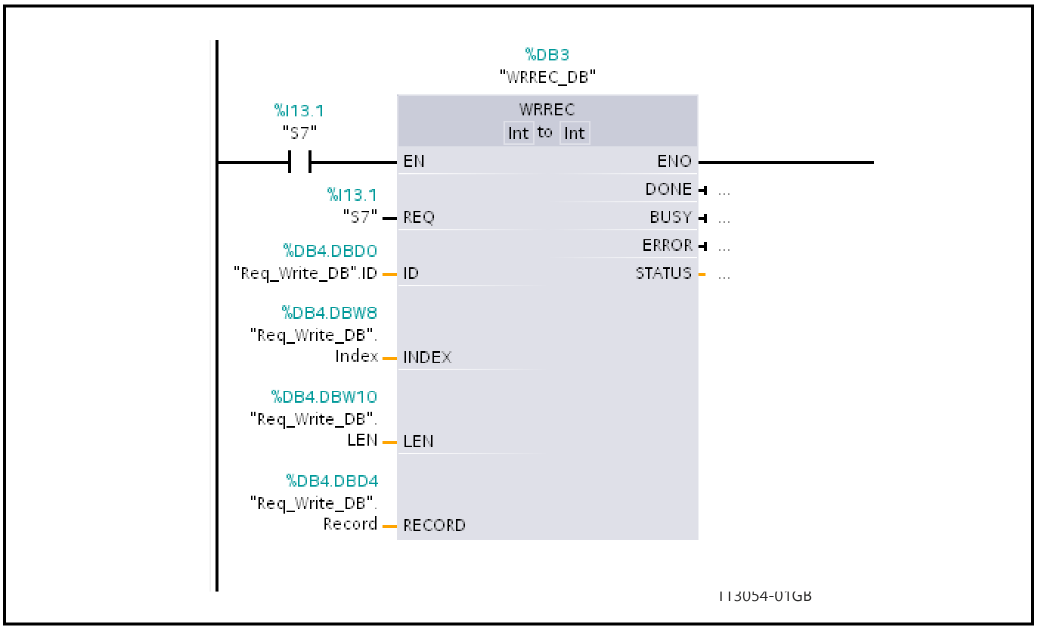
Object 0x13 are for acyclic view or change of motor registers above 255.
To access a motor register the register number is calculated as follows:
Motor register number = Subindex + 255

6.3.4 Accessing parameter objects

Parameter objects are accessible by use of acyclic data. In Siemens Step 7, this is done with the Special Function Blocks SFB52 and SFB 53.

Write parameter

Write to parameters is done with the SFB53, as shown below.



The data block must be setup prior to use, in this example "Req_Write_DB".

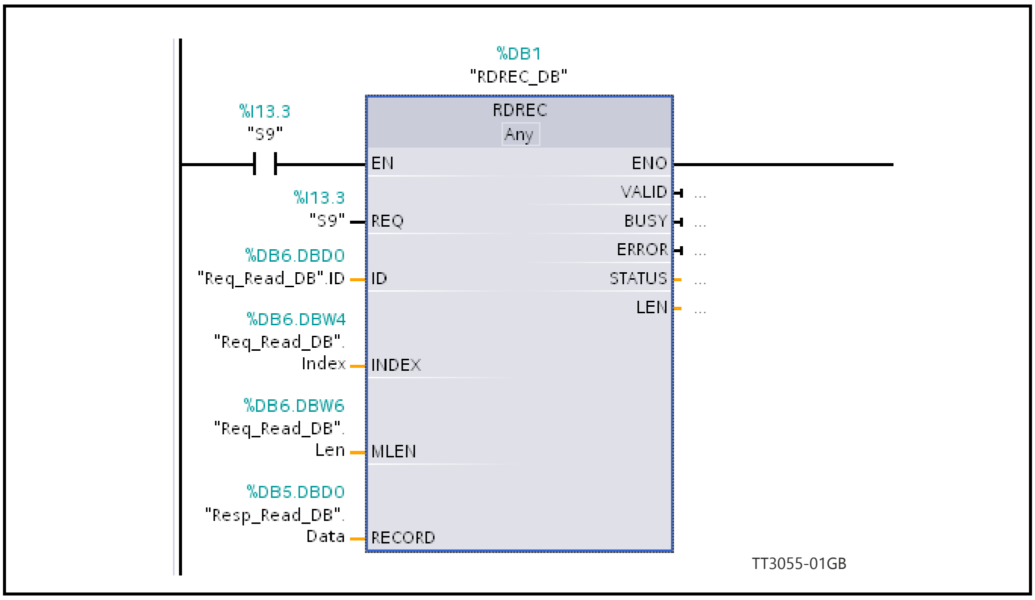
Name	Description	Example
ID	ID of device	2042
Index	Object and subobject to write to High byte = Object, Low byte = Subobject (parameter/register no.)	0x1231 (Object 0x12, parameter 0x31)
LEN	Length of data	4 (always 4 byte = 32 bit)
Record	32 bit data to write	0xFEDCBA98

Example:

Write 0xFEDCBA98 to object 0x12 subobject 0x31 (= motor register no. 49), in JVL device with ID 2042.

Please notice: Even though all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

Read parameter
Read of parameters is done with SFB52, as shown below.



The data block must be setup prior to use, in this example "Req_Read_DB", and the 32 bit result will be in "Resp_Read_DB.Data".

Name	**Description**	**Example**
ID	ID of device	2042
Index	Object and sub-object to read from High byte = Object, Low byte = Subobject (register no.)	0x1122
LEN	Length of data	4 (always 4 byte = 32 bit)

Example:
Read from object 0x11 subobject 0x22 (= module parameter no. 34), in JVL device with ID 2042.

Please notice: Even though all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

6.4 Ethernet switch

6.4.1 Selecting an Ethernet Switch

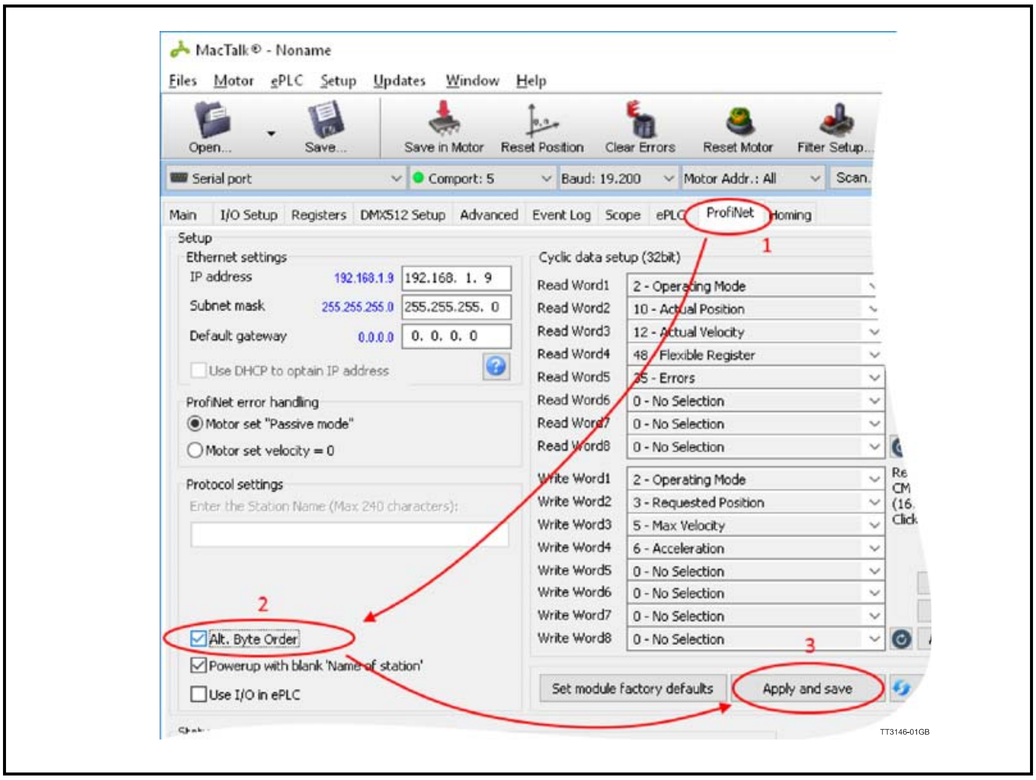
Depending on the network topology and size a suitable switch can be used. Also if multiple separated networks need to be connected a switch is used.
Depending on the actual size of the network different requirements needs to be met.

It is absolutely mandatory that every switch device or other device acting as a switch complies with the Profinet RT protocol (LLDP and PN_PTCP frames must be recognized).

Otherwise the net might very well get congested; because non-Profinet conforming switches will broadcast messages not intended for broadcast.
Besides the Port mirroring function for network analyzing and troubleshooting purposes, can be advantages. This feature makes it possible to route traffic out on a separate port connected to a network analyser for debugging purposes and general performance monitoring.
The JVL Profinet module has build in 2 port switch useful if a limited amount of motors is connected in a daisy chaining topology. This switch is Profinet-IRT capable and will of cause obey the above mentioned demands.

6.4.2 Using MISxxxEP/MILxxxEP or MAC00-EPx with Kuka Robot PLC's.

Kuka robot PLC's use a different endianness (byte order) than standard Siemens PLC's, so it is nesesary to swap the byte order for making it work properly.
This is done by enabling the byte swap in the Profinet module by setting bit 12 in the module register 6 and saving this setup to flash. Please refer to paragraph 9.2.6 for other setup bits in this register, and please refer to paragraph 6.5.6 for saving to flash.
The byte swapping can also be changed via MacTalk by checking the "Alt. byte order" checkbox located in the Profinet tab, and afterwards pressing the "Apply and Save" button.



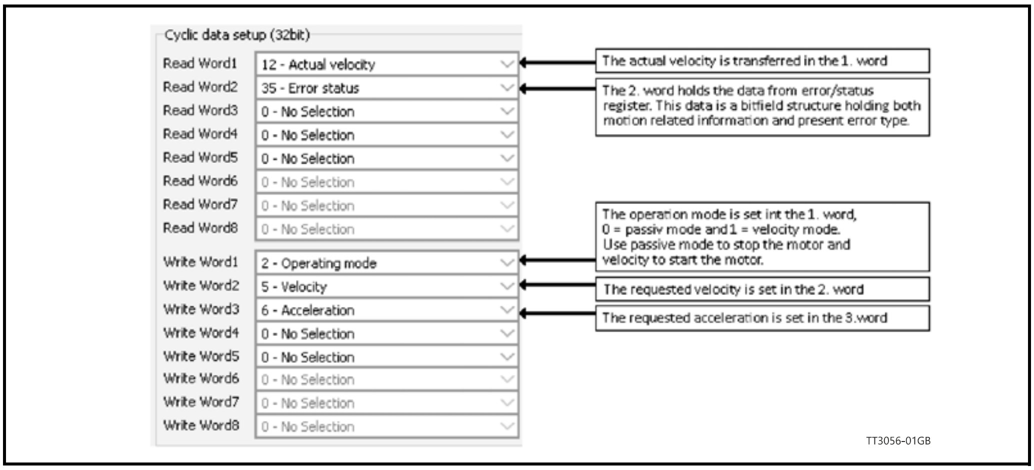
6.5 Examples

6.5.1 Running Velocity control

To use the JVL motor in velocity -mode the following registers basically is of interest.

1. "Mode" - mode register - register 2
2. "V_SOLL" - velocity register 5
3. "A_SOLL" - acceleration register 6
4. "Error/Status" - register 35

So, to control these registers the assembly object needs to be configured.
From MacTalk the setup is configured as this.



With the settings illustrated above we initiate the velocity mode by writing 0x1 to the first word-value, this is velocity mode.

Since different PLC's have different methods of implementation the basic steps is described in the following. (Constant values valid for MAC800, for other motors, please consult the motor manual)

1. **Set the needed velocity**.
   V_SOLL = V x 2.77 [rpm].
   Example: We need the motor to run with a constant speed of 1200 RPM.
   So, V_SOLL = 1200/2,77 = 433 cnt/smp
2. **Set the needed acceleration.**
   A_SOLL = A x 271 [RPM/s2].
   Example: We need the motor to accelerate with 100000 RPM/s2 so, A_SOLL = 100000/271 = 369 cnt/smp2
3. **Now set the motor in velocity mode and thereby activate the motor.**
   Example: The motor needs to be activated by setting it into velocity mode, so we need to set the mode register to the value 1. Mode = 1 which is velocity mode, now the motor will use the acceleration and the velocity just configured.

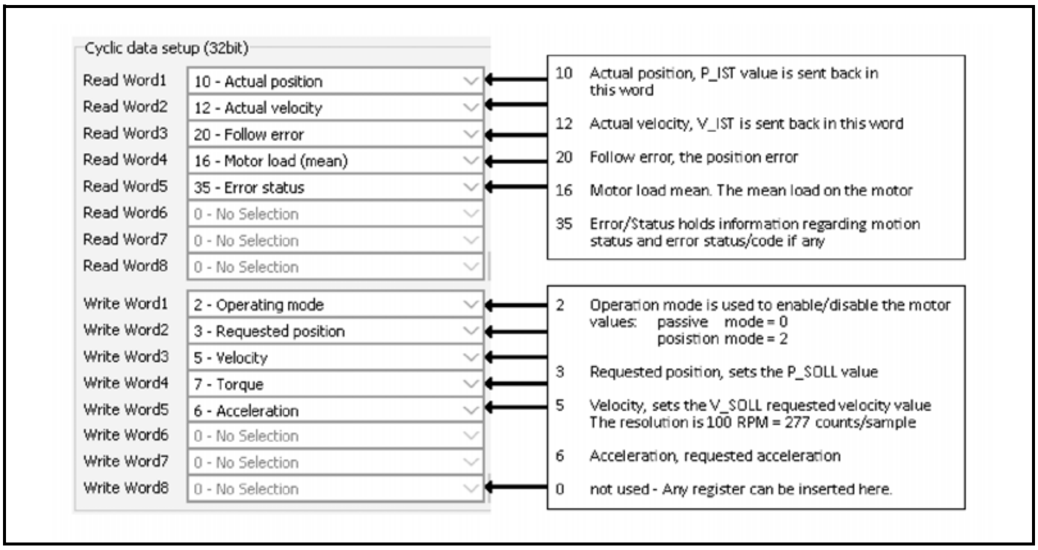
Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 or
Motor registers MAC400 - 4500, page 301 or
Motor registers MISxxx, page 320.

6.5.2 Running Position control

Running the motor in position control requires that the mode register is set for position control. The following registers is of particular interest when position mode is used.

1. "Actual position" -P_IST, register 10
2. "Actual velocity" -V_IST, register 12
3. "Follow error" - The actual position error, register 20
4. "Motor load mean" - average motor load, register 16
5. "Error/Status" -register 35
6. "Requested position" -P_SOLL, register 3
7. "Requested velocity" -V_SOLL, register 5
8. "Requested acceleration" -A_SOLL, register 6

In this mode the position is controlled by applying a requested position to the "P_SOLL" -register and the actual position is monitored in the "P_IST" register. The V_SOLL and A_SOLL registers sets the velocity and acceleration used when positioning occurs.



6.5.3 General considerations

The register 35 in the motor holds information on the actual error/status. So it is crucial that this register is configured in the cyclic data and thereby obtained and monitored in the Master. In case of an error situation the motor will stop and the cause will be present in the register 35 and hence in the I/O -data.
This register also holds information on the motion status such as:

- In position, bit 4
- Accelerating, bit 5
- Decelerating, bit 6

Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 or
Motor registers MAC400 - 4500, page 301 or
Motor registers MISxxx, page 320.

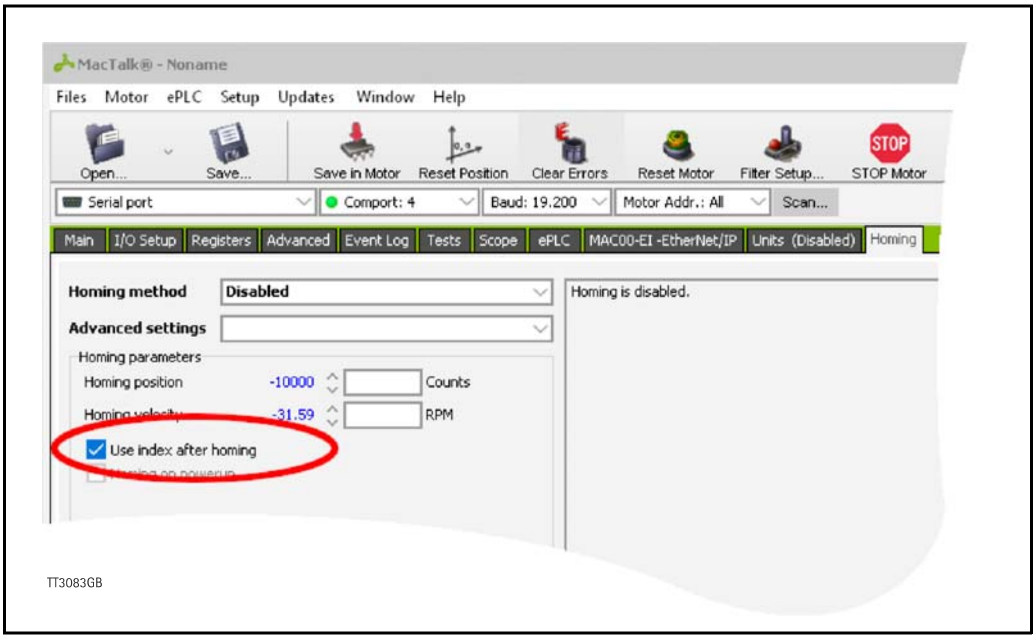
The JVL motor is basically put into a working mode and into a passive mode where the motor axle is de-energized, by setting register 2 into either 0 = "passive mode" or into one of the supported modes.
(continued next page)

Example.
1= "Velocity mode" / 2= "Position mode" / etc.

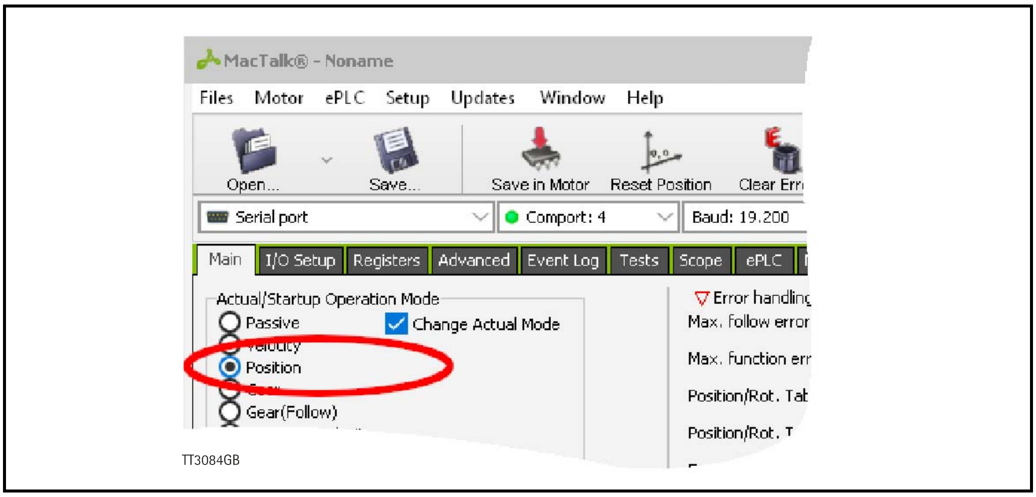
So in order to Stop or Start the motor this register can be supported in the I/O data or by sending an acyclic message.

6.5.4 Homing using only cyclic I/O (JVL profile).

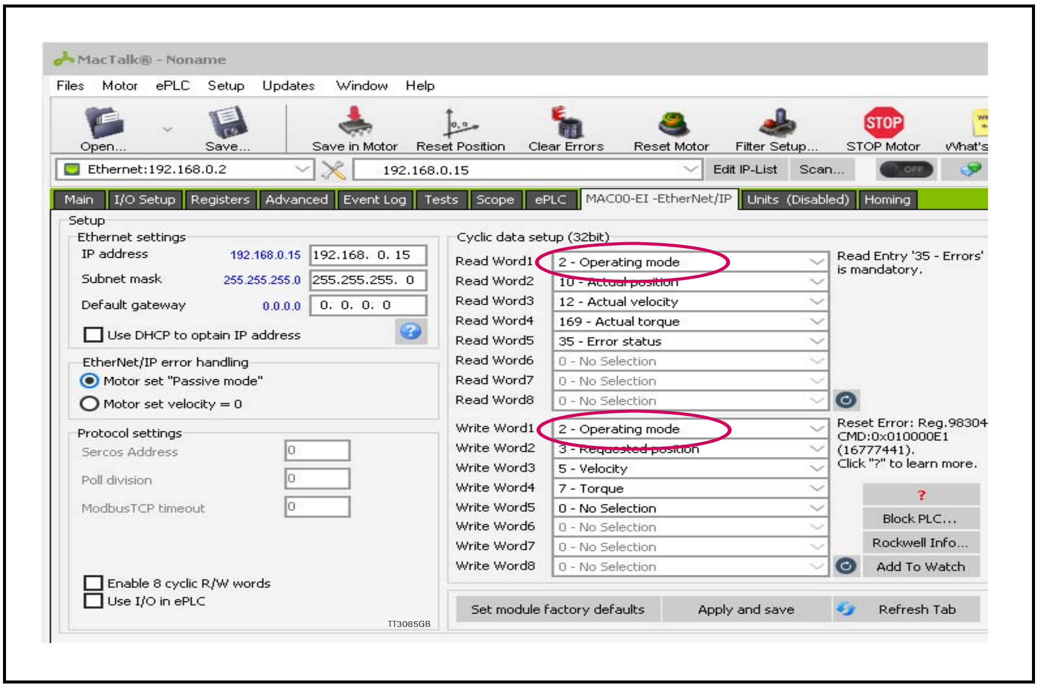
When doing a homing (Zero search), with only cyclic I/O, some preconditions have to be met:
Zero search position, zero search velocity and zero search torque (torque only for MAC motors) has to be set in MacTalk in the "Main" tab, and saved in flash in the motor once and for all.



Startup mode should be set to position, for the motor to stay in position after the homing sequence. And this setting should also be saved in flash.



Register 2 (Operating mode) has to be present in BOTH the cyclic read words and cyclic write words.



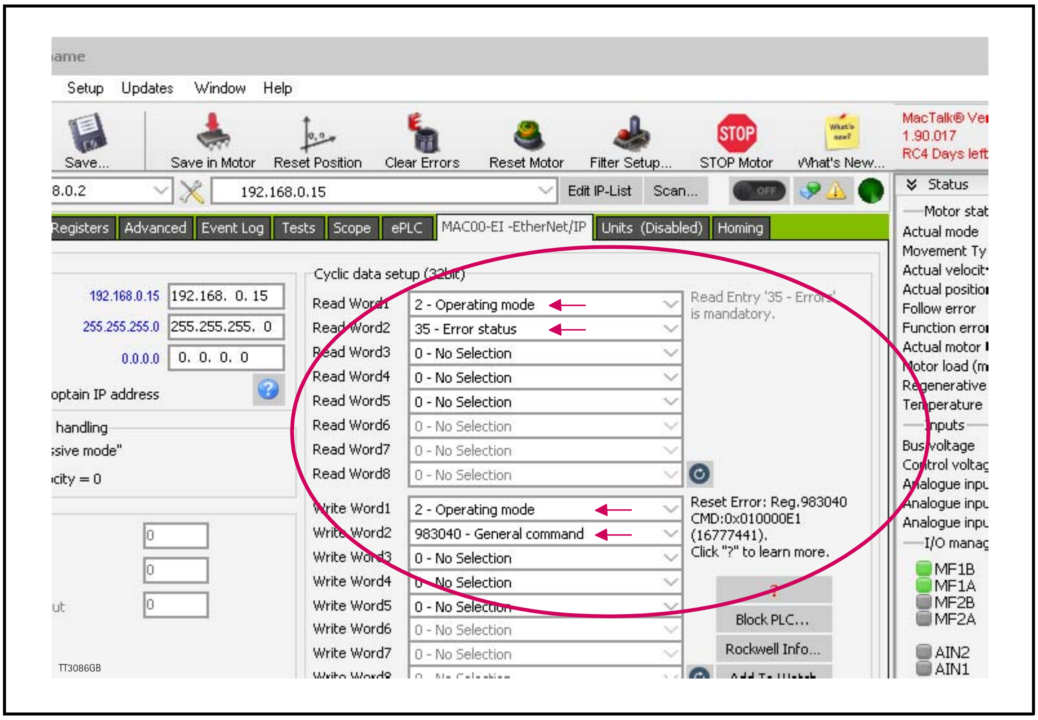
Procedure in the PLC:

- Treat the transmitted Register 2 as "Requested_Mode" and the received register 2 as "Actual_Mode".
- When homing is wanted, set the "Requested_Mode" to one of the values 12, 13, 14, 25 or 26 depending of the requested homing mode (12 = Torque based zero search mode (only MAC motors). 13 = Forward/only zero search mode. 14 = Forward+backward zero search mode (only MAC motors). 25 = Enc. index (only MAC400+). 26 = Enc. quick index (only MAC400+).). For a comprehensive description of the homing modes, refer to the general MAC motor manual - LB0047-xxGB.
- Observe that the "Actual_Mode" is changing to the homing mode. Now the module is blocking cyclic writes TO the motor. Cyclic reads is still active.
- Wait for register 35 "Error status" bit 4 to be active =IN_POSITION. (Indicates that homing is finished).
- Then change "Requested_Mode" to whatever needed. The blocking of cyclic writes to the motor is then released by the module.

6.5.5 Relative positioning.

There are a number of ways to do relative positioning, but the one explained here is very simple, and can be used with a constant distance, or exchangeable distance, to move every time it is requested.

Preconditions:
Place the module command register (register 983040 in MacTalk) in the cyclic write list. The cyclic setup, could for example look like this:



Procedure in the PLC:

1. Set up register P7 in motor to requested relative offset.
2. Make sure one net cycle has passed, so P7 resides in the motor.
3. Issue command 0x010000F1 (0x01000071 if MIS/MILxxxxxxEPxxxxx) in module command register (register 983040 in MacTalk).
4. Make sure one net cycle has passed, so command is interpreted by the motor.
5. Set module command register to zero. This will prepare the Ethernet module for new commands.
6. If needed then monitor register 35 (Error status): When bit 4 is set (in position), then the move is finished.
7. When a new relative move is requested, go to step 3.

You may also have the P7 register in the cyclic write list, thereby enabling easy change of the relative distance to move.

6.5.6 Save parameters to flash

Saving of the Profinet® module or the motor controller registers to flash (non-volatile memory) only requires a simple non-cyclic command to the Profinet® module command register which is accessible via object 0x10 (see paragraph 6.3.3 and 6.3.4)

Save Profinet® module parameters to flash:

- Write the value 16#0000 00010 (= 16 dec.) to object 0x10 subobject 0x00 (=index 0x1000).

Save motor parameters to flash:

- Write the value 16#0000 00110 (= 272 dec.) to object 0x10 subobject 0x00 (=index 0x1000).

6.5.7 Reset motor errors

If the motor gets an error the error can be cleared by issuing a command to the Module command register, which is accessible via object 0x10 (see *Parameter objects., page 150* and *Configuring the system, page 141*)
Or issue the command to the module command register if the module command register is placed in the cyclic write list (refer to *Change of cyclic data setup from master via service channel* for changing the cyclic setup).

Reset motor error command to module command register:

Motortype	Command (hex)	Command (dec)
MIS/MIL	16#0100 0061	16777313
MAC	16#0100 00E1	16777441

If done via non-cyclic data transfer then the command is written to object 0x10 subobject 0x00 (=index 0x1000), then the error is cleared.
If done with cyclic data then follow the below procedure.

1. Send the command value from above table to the mapped variable in the PLC project which points to the Profinet® module command register.
2. Make sure one net cycle has passed, so command is interpreted by the motor.
3. Send the command value
4. Write 0 (zero) to the mapped variable in the PLC project which points to the Profinet® module command register. This will prepare the Profinet® module for new commands.

6.5.8 Change of cyclic data setup from master via service channel

Besides changing the cyclic data setup in the JVL Profinet module with the help of MacTalk it is also possibly to change the setup by means of non-cyclic data.

1. Change the Profinet® module registers for cyclic setup in register 16-23 (object 0x11 subobject 0x10-0x1F).
2. Save the settings in module flash, see *Save parameters to flash*.
3. Do a reboot for the changes to take effect, by writing the value “1” to (object 0x10 subobject 0x00).
   The master will then loose connection, and a reconnect is necessary. And the Profinet module will start up with the new settings.

7 ModbusTCP/IP® Users Guide

7.1 Introduction to Modbus TCP/IP®

MAC - Modbus TCP/IP Module
Type:
MAC00-**EM**4 (shown) or
MAC00-**EM**41 (extended I/O)

To be used in following servo products:
MAC50, 095, 140 and 141
MAC400 and MAC402
MAC800
MAC1500 and MAC3000

MIS - Modbus TCP/IP motors.
Type:
MISxyyzz**EM**xx85 or
MILxyyzz**EM**xx85

7.1.1 Introduction.

Modbus TCP/IP or Modbus TCP — is a Modbus variant used for communications over TCP/IP networks, connecting over port 502.
The JVL implementation also supports Modbus UDP, the same protocol as ModbusTCP but transferred via UDP, which is faster, but with no connection control. It is basically a Modbus RTU without a checksum calculation as lower layers already provide checksum protection.
It is protocol based on the standard TCP/IP protocols so it is applicable anywhere there is standard Ethernet available as it have no special requirements regarding the Ethernet hardware, opposite some of the other industrial Ethernet protocols. Further information about Modbus TCP is available from the Modbus Organization www.modbus.org.

7.1 Introduction to Modbus TCP/IP®

7.1.2 Abbreviations

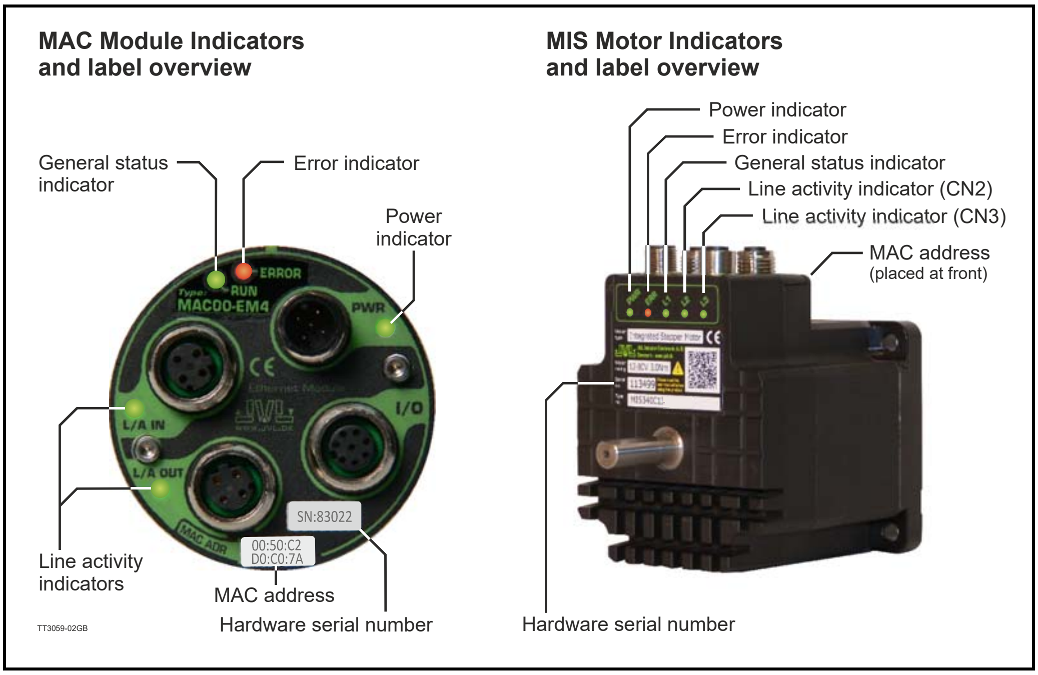
The below general used terms are useful to know before reading the following chapters.

100Base-Tx	100 MBit Ethernet on twisted pairs
IP	Internet Protocol - IP address ~ the logical address of the device which is user configurable.
MAC	Media Access Controller - MAC address ~ the hardware address of the device.
MacTalk	A windows PC based program supplied from JVL. This is an overall program to install, adjust and monitor the function of the motor and a module in stalled in the motor.
TCP	Transfer Control Protocol (an IP based protocol used widely on the internet)
UDP	User Datagram protocol (an IP based protocol used widely on the internet)
DHCP	Dynamic Host Configuration Protocol (Automatic configuration of IP address netmask and gateway from a DHCP server).

7.2 Commissioning

7.2.1 Indicator LED's - description.

The LED's are used for indicating states and faults of module. There is one power LED, two link/activity LED's (one for each Ethernet connector) and 2 status LED's.



LED indicator descriptions - Covers both MAC and MIS/MIL.

LED Text MAC / MIx	**Off**	**Red**	**Orange**	**Green**	**Flickering Green**	**Very slow blinking green**
L/A IN / L2	No valid Ethernet connection	-	-	Ethernet is connected	-	
L/A OUT / L3	No valid Ethernet connection	-	-	Ethernet is connected	-	
RUN / L1	-	Initializing or no valid Ethernet	TCP server open for connections	TCP client connected	-	Wrong constel-lation of IP, NM, and GW
ERROR / ERR	No Errors	Fatal error	-	-	-	
PWR	Power is not applied	Only **MIS17x** and **MIS23x**: Power supply is too low.	-	Power is applied to both motor and module.	Power is applied to module but no communi-cation with motor.	
Notes: **Flickering:** Rapid flashing with a period of approx. 50ms(10Hz).						

7.2.2 Mechanical installation

The network cables must be connected to the two M12 connectors marked "L/A IN" and "L/A OUT" at the MAC module and "CN2" and "CN3" at the MIS/MIL motors. The cable from the IO CONTROLLER is connected to either of the two ports. In the line topology, if there are more slave devices in the same line, the next slave device is connected to the second port. Standard CAT 5 STP cables can be used. It is not recommended to use UTP cables in industrial environments, which is typically very noisy.

7.2.3 Network configuration

To enable communication through the Ethernet network, the module needs a valid IP address. This is either done manually in MacTalk, or DHCP is enabled in Mactalk, and then the IP address, net mask and gateway is automatically obtained from a DHCP server. If DHCP is enabled then a DHCP server has to be available on the local network.

7.2.4 Communication description

Connect to Modbus TCP module by opening a TCP client connection to the module IP address on port 502. It's possibly to have only one open connection at a time.
Or connect via UDP, by just sending UDP requests to port 502. With UDP it is possible to have several masters at the same time. It is even possible to have one TCP master and several UDP masters connected at the same time.

The registers in the motor and in the module are all 32 bit. To comply with the clean 16-bit Modbus standard, a 32-bit register must be read or written as two consecutive 16-bit registers. The register address mapping follows the normal documented register numbers but the address field, must be multiplied by two, so to read or write register 3, P_SOLL, use the address 6. Thereby, enabling transfer of one 32 bit register, as two 16 bit registers, where the least significant 16 bit "register" is transmitted first (see examples).
It is possibly to access both motor registers and Modbus TCP module registers. Motor registers is accessed by addressing register 0x00 – 0x1FE (for motor register 0-255), and module registers is accessed by addressing 0x8000 – 0x807E (for module register 0-64).

The Modbus TCP extension includes 7 additional bytes to the original Modbus protocol which allows for transport over the TCP/IP layers – the MBAP header. So the frame format looks like this (excluding TCP/IP header):
| - MBAP Header - | - Function Code - | - Data - |

The **MBAP Header** (ModBus Application Protocol Header) consists of 7 bytes of information:

Transaction Identifier	2 bytes	Identification of Request/Response transaction – copied from request to response
Protocol Identifier	2 bytes	0 = Modbus protocol
Length	2 bytes	number of following bytes – includes the unit identifier
Unit Identifier	1 byte	identification of remote slave.

Function codes

The MAC00-EMx/MIS/MILxxxxxxEMxx Modbus TCP module supports these function codes:

0x03	Read holding registers
0x04	Read input registers
0x06	Write single register
0x10	Write multiple registers (up to 32 modbus registers = 16 x 32bit registers)
0x17	Read/Write multiple registers

If an error is detected in the received request an exception frame is returned.

| - MBAP Header - | - Function Code - | - Exception code - |

The returned function code, in case of an exception, is the transmitted function code with bit 7 set (means that 0x03 ➔ 0x83, and 0x10 ➔ 0x90, and 0x17 ➔ 0x97).

Exception codes

0x01	Function code not supported
0x02	Not allowed register no.
0x03	Too many registers or uneven no. of registers, as every register in motor/module is 32 bit and requires 2 x 16 bit modbus registers.

(0x03 / 0x04) Read Holding Registers / Read Input Registers
Read of registers. Max. 124 x 16bit registers at a time (=62 x 32bit registers). Please remark that the Modbus protocol is 16 bit and all motor/module registers is 32bit when accesed via the ModbusTCP module. Therefore it is necesary to read two Modbus registers to get one motor/module register.
Only even no. of 16bit registers is supported. With the exception of reading one 16bit modbus register which accesses the upper or lower half of a 32bit motor/module register.The response time is increased slightly for every register added. See *Minimum poll time, page 170* for minimum poll time.

Request:

7 bytes	1 byte	2 bytes	2 bytes
MBAP header	Modbus Cmd. (0x03 / 0x04)	Motor register no. x 2 or module register no. x 2 + 0x8000	Register count

Response:

7 bytes	1 byte	1 byte	2 bytes	2 bytes
MBAP header	Modbus Cmd. (0x03)	Data byte count	Register value low 16bit	Register value high 16bit

Example, read of **module** register 3 (= IP address = 192.168.100.1 = 0xC0.0xA8.0x64.0x01):
Request -
|0x00|0x01|0x00|0x00|0x00|0x06|0x01|0x03|0x80|0x06|0x00|0x02|
Response – (Note the byte order!)
|0x00|0x01|0x00|0x00|0x00|0x07|0x01|0x03|0x04|0x64|0x01|0xC0|0xA8|

Possibly exception responses: 0x02, 0x03.

For further documentation see “Modbus_Application_Protocol_V1_1b.pdf” and “Modbus_Messaging_Implementation_Guide_V1_0b.pdf” found on www.modbus.org.

Please notice: Even though all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

7.2.5 (0x06) Write Single register (From build number 10094)

Write of a single 16bit Modbus protocol register at a time. As all registers in the motor/ ModbusTCP module is 32bit when accessed via the Modbus module, care must be taken to do two writes in the order - uneven first, then even - to use this command. The value is only written to the internal register when the even 16bit (low word) is written. If possible it is recommended to use the command 0x10 Write multiple registers writing two or more modbus registers at a time for writing one or more motor/module registers.

Request:
7 bytes, 1 byte, 2 bytes, 2 bytes
MBAP HeaderModbus cmd. (0x06)Motor register no. x 2 or module register no. x 2 + 0x8000 Register value

Responce:
7 bytes, 1 byte, 2 bytes, 2 bytes
MBAP HeaderModbus cmd. (0x06)Motor register no. x 2 or module register no. x 2 + 0x8000 Register value

Example, write of motor register 3 (= P_SOLL = 0x12345678):
First write uneven 16bit (high word):
Request – (Note the byte order!)
|0x00|0x02|0x00|0x00|0x00|0x0B|0x01|0x06|0x00|0x07|0x12|0x34|
Response -
|0x00|0x02|0x00|0x00|0x00|0x06|0x01|0x06|0x00|0x07|0x12|0x34|

Second write even 16bit (low word):
Request – (Note the byte order!)
|0x00|0x02|0x00|0x00|0x00|0x0B|0x01|0x06|0x00|0x06|0x56|0x78|
Response -
|0x00|0x02|0x00|0x00|0x00|0x06|0x01|0x06|0x00|0x06|0x56|0x78|

Possibly exception responses: 0x02, 0x03.

For further documentation see “Modbus_Application_Protocol_V1_1b.pdf” and “Modbus_Messaging_Implementation_Guide_V1_0b.pdf” found on www.modbus.org.

(0x10) Write Multiple registers
Max. 32 x 16bit write registers at a time (=16 x 32bit registers). Only even no. of 16bit registers are supported

Request:

7 bytes	1 byte	2 bytes	2 bytes	2 bytes	2 bytes
MBAP header	Modbus Cmd. (0x10)	Motor register no. x 2 or module register no. x 2 + 0x8000	Register count	Register value low 16bit	Register value high 16bit

Response:

7 bytes	1 byte	2 bytes	2 bytes
MBAP header	Modbus Cmd. (0x10)	Motor register no. x 2 or module register no. x 2 + 0x8000	Register count

Example, write of motor register 3 (= P_SOLL = 0x12345678):
Request – (Note the byte order!)

|0x00|0x02|0x00|0x00|0x00|0x0B|0x01|0x10|0x00|0x06|0x00|0x02|0x04|0x56|0x78|0x12|0x34|

Response -
|0x00|0x02|0x00|0x00|0x00|0x06|0x01|0x10|0x00|0x06|0x00|0x02|

Possibly exception responses: 0x02, 0x03.

For further documentation see "Modbus_Application_Protocol_V1_1b.pdf" and "Modbus_Messaging_Implementation_Guide_V1_0b.pdf" found on www.modbus.org.

Pleasd notice: Even though all registers is to be transmitted as 32 bit some of them originally derive from 16 bit in the MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

(0x17) Read/Write multiple registers
Simultaneous read and write of registers. Max. 124 x 16bit read registers at a time (=62 x 32bit registers). And max. 32 x 16bit write registers at a time (=16 x 32bit registers). Only even no. of 16bit registers is supported.
The response time is increased slightly for every register added. See *Minimum poll time, page 170* for minimum poll time.

Request:

7 bytes	1 byte	2 bytes	2 bytes	2 bytes	2 bytes	1 byte	2 bytes**	2 bytes**
MBAP header	Modbus Cmd. (0x17)	Read Motor register no. x 2 or Read Module register no. x 2 + 0x8000	Register count Read. (Max. 124)	Write Motor register no. x 2 or Write Module register no. x 2 + 0x8000	Register count Write. (Max. 32)	Write byte count	Register value write low 16bit	Register value write high 16bit

** To be repeated 1-16 times.

Response:

7 bytes	1 byte	1 byte	2 bytes*	2 bytes*
MBAP header	Modbus Cmd. (0x17)	Data byte count	Register value low 16bit	Register value high 16bit

* To be repeated 1-62 times.

Example:
Read of module register 3 (= IP address = 192.168.100.1 = 0xC0.0xA8.0x64.0x01) and write of motor register 3 (= P_SOLL = 0x12345678) in one operation:

Request - (Note the byte order!)

|0x00|0x02|0x00|0x00|0x00|0x0B|0x01|0x17|0x80|0x06|0x00|0x02|0x00|
0x06|0x00|0x02|0x04|0x56|0x78|0x12|0x34|

Response - (Note the byte order!)

|0x00|0x01|0x00|0x00|0x00|0x07|0x01|0x17|0x04|0x64|0x01|0xC0|0xA8|

Possibly exception responses: 0x02, 0x03.

For further documentation see "Modbus_Application_Protocol_V1_1b.pdf" and "Modbus_Messaging_Implementation_Guide_V1_0b.pdf" found on www.modbus.org.

Please notice: Even though all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

7.2.6 Minimum poll time

The minimum poll time is the minimum amount of time between each poll request on the Ethernet. If operating with values lower than those listed, data loss will occur.

No. of polled motor registers (32bit)	Motor series		
	MAC050-MAC141	MAC400-MAC4500	MIS/MILxxx
1	2ms	2ms	2ms
5	10ms	3ms	3ms
10	20ms	4ms	4ms

The minimum poll times is only valid if not sending any requests while in any operating mode. MODULE registers can be appended at no extra timing cost. If motor register 35 is not polled it will be added internally anyway and has to be added to the minimum cycle time with 2ms if MAC050-MAC141.

7.2.7 Quick start guide

This section describes the steps to configure the **MAC00-EMx** module or the **MIS/ MILxxxxxxEMxxxx** motor using the shareware program **Modbus poll**, which can be obtained from the website: http://www.modbustools.com/

Set IP address

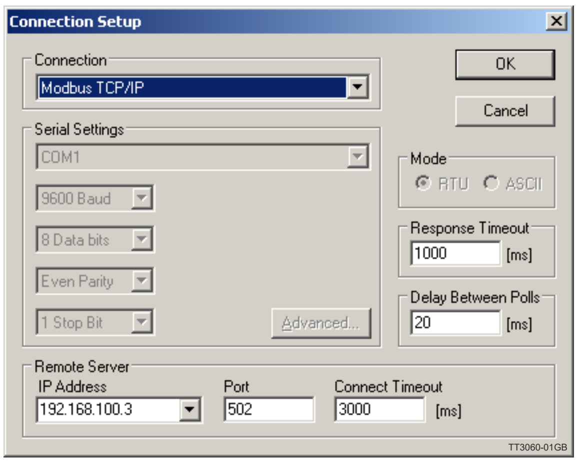
1. Connect the RS232 communication cable.
2. Apply power to the motor, and make sure the PWR LED is lit.
3. Open MacTalk and select the “MAC00-EM (Modbus TCP)” tab.
4. Change the IP address, to one suitable for the network.
5. Press “Apply and save”.

Installation

6. Connect an Ethernet RJ45-M12 cable to the Ethernet interface of the PC with **Modbus Poll** installed and to “L/A IN” or “L/A OUT” at the MAC module and “CN2” or “CN3” at the MIS/MIL motors.
7. Make sure power is applied to all devices.

Connect to MAC00-EMx or MIS/MILxxxxxxEMxxxx

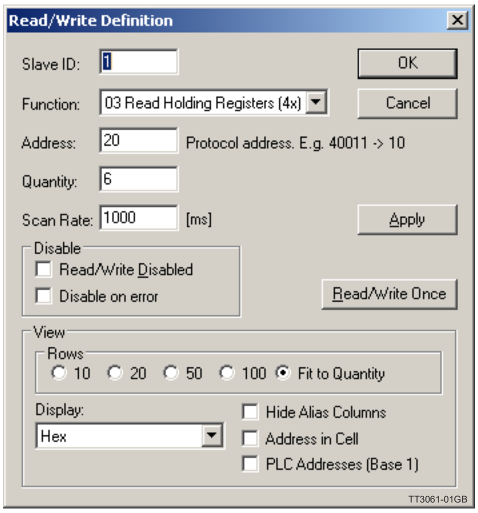
8. In the Connection menu of Modbus Poll select **Connect.**
9. The connection is made by choosing the “Modbus TCP/IP” or the "Modbus UDP/ IP"protocol, the IP address of the motor, and port 502. As seen below.



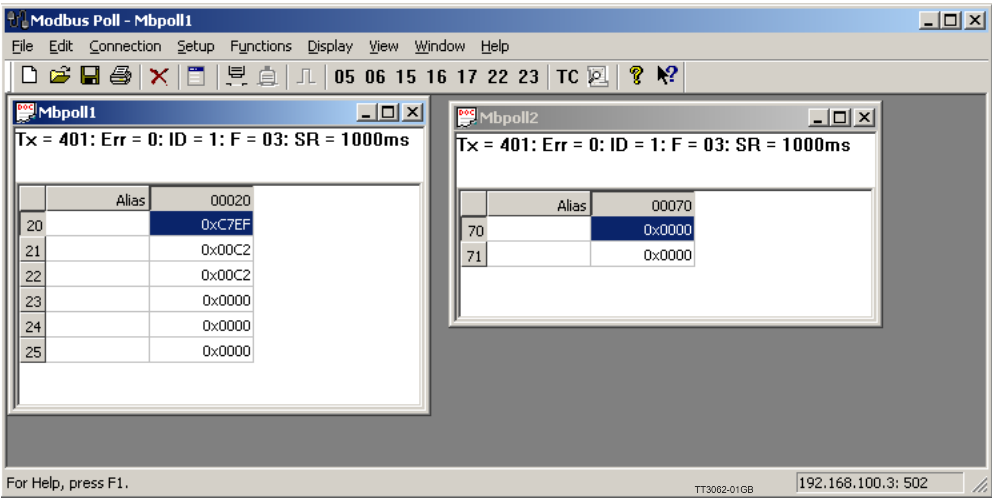
10. The “Run” led on the motor (which is red when powering up) should now change from orange to green when connected to the client (Modbus poll).

Setup poll registers

11. When connected it is possibly to change the polling of registers in the motor by right-clicking in the default “Mbpoll1” window and selecting “read/write definition”. In the shown example below is chosen “Address:” 20 (= register 10), and “Quantity:” 6 (= 3 x 32bit registers). This means that register 10, 11 and 12 is polled.

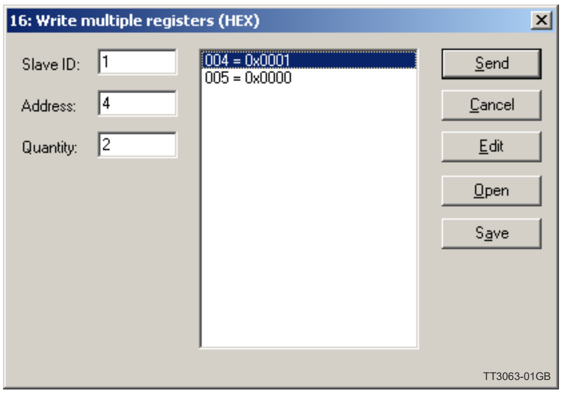


12. By choosing **File** and **New** a second poll window is opened where “Address:” 70 and “Quantity:” 2 is chosen. Meaning that error register 35 is polled. Your screen should now look something like this:



Transmit data to motor

You can transmit data to the motor by chosing **Functions** and **16: Write registers**, and if choosing "Address:" 4, "Quantity:" 2, and data = 0x01 (in address 004 = least significant 16bit) as shown below (mode register = velocity) the motor should start turning. If not then try to also write velocity (reg. 5 = addr. 10), acceleration (reg. 6 = addr. 12) and/or Torque (reg. 7 = addr. 14) to some valid values. Please find a complete list of register descriptions in the appendix:
Motor registers MAC050 - 141, page 292 or
Motor registers MAC400 - 4500, page 301 or
Motor registers MISxxx, page 320.



Please notice: Even though all registers is to be transmitted as 32 bit some of them originally derive from 16 bit in the MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

7.3 Register access

The registers in the motor and in the module are all 32 bit (at least they are when travelling through the module so special care must be taken with those registers in MAC050-141 which originally is 16bit). To comply with the clean 16-bit Modbus standard, a 32-bit register must be read or written as two consecutive 16-bit registers. The register address mapping follows the normal documented register numbers but the address field must be multiplied by two, so to read or write motor register 3, P_SOLL, use the address 6. Thereby enabling transfer of one 32 bit register, as two 16 bit registers, where the least significant 16 bit "register" is transmitted first (see examples in section 7.2.4).
Motor registers are accessed by addressing register 0x00 – 0x1FE (for logic motor register 0-255), and modbus register 0x200 - 0x3FE for extended motor registers (256-511), and module registers is accessed by addressing 0x8000 – 0x807E (for logic module register 0-64). Please find a complete list of register descriptions in the appendix: *Motor registers MAC050 - 141, page 292* or *Motor registers MAC400 - 4500, page 301* or *Motor registers MISxxx, page 320*.

7.3.1 Module registers.

Logic register no.	Modbus address (hex)	Modbus address (dec)	Read only	Default	Description
0	0x8000	32768	X		Not used
1	0x8002	32770	X		High 16 bit of MAC address (placed in low 16 bit of word)
2	0x8004	32772	X		Low 32 bit of MAC address
3	0x8006	32774			IP address
4	0x8008	32776			Net mask
5	0x800A	32778			Gateway
6	0x800C	32780		0x00	Setup bits
7	0x800E	32782			Digital outputs on module
8	0x8010	32784			Reserved for other protocols
9	0x8012	32786			Reserved for other protocols
10	0x8014	32788			Modbus timeout. 0 = timeout function disabled
11-14	0x8016 0X801C				Reserved for future use
15	0X801E	32798			Command register
16 – 31	0x8020 – 0x803E				Reserved for other protocols
32	0x8040	32832	X		Module serial no.
33	0x8042	32834	X		Module hardware version
34	0x8044	32836	X		Module software version
35	0x8046	32838	X		No. of internal motor communication timeouts
36	0x8048	32840	X		No. of retry frames to motor
37	0x804A	32842	X		No. of discarded frames to motor
38	0x804C	32844	X		Total no. of frames to motor
39-46	0x804E – 0X805C		X		Reserved for future use
47	0x805E	32862	X		Digital inputs on module
48	0x8060	32864	X		Status bits
49-63	0x8062 – 0X807E		X		Reserved for future use

Note: Module parameters are not automatically saved to permanent memory after a change. The parameters can be saved permanently by applying a "Save parameters to flash" command afterwards.

7.4 Examples

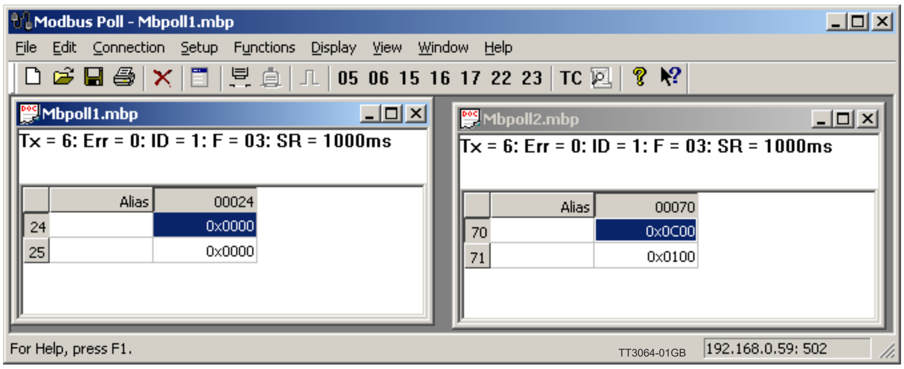
In this section is shown some examples of controlling the motor by Modbus TCP. As master is used the shareware program **Modbus poll** which can be obtained from the website: http://www.modbustools.com/
These examples assume you are already connected to the motor.
For connecting to the motor, please follow the *Quick start guide, page 171*.

7.4.1 Running Velocity control

To use the JVL motor in velocity mode the following motor registers is of interest.

1. "Mode" – mode, register 2
2. "V_SOLL" – velocity, register 5
3. "A_SOLL" – acceleration, register 6
4. "Error/Status" – register 35

So to control these registers setup polling of motor register 12 – actual velocity (modbus address 24), and motor register 35 (modbus address 70).
This could look like shown below.



Now we can monitor the motor errors and the motor velocity.

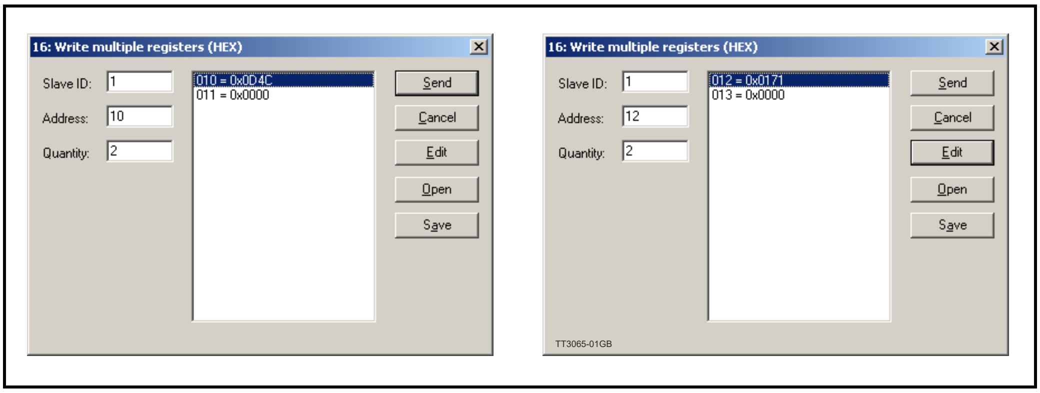
7.4 Examples

Calculate the values needed for velocity and acceleration (constant values valid for MAC400, MAC1500 and MAC4500).

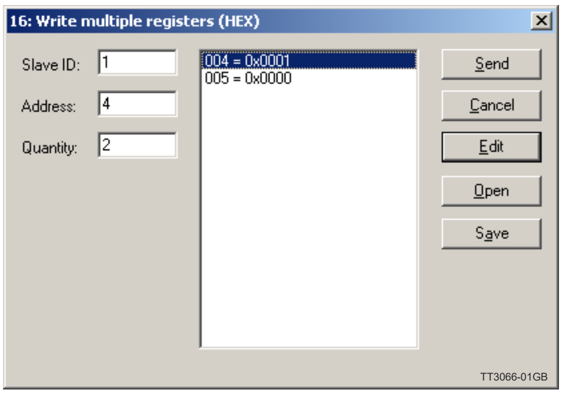
1. **Set the needed velocity**. V_SOLL = V x 2.8369 [rpm]
   Ex. *We need the motor to run with a constant speed of 1200 RPM. So,*
   V_SOLL = 1200 x 2,8369 = 3404 cnt/smp (= 0x0D4C)

2. **Set the needed acceleration**. A_SOLL = A / 271 [RPM/s^2]
   Ex. *We need the motor to accelerate with 100,000 RPM/s^2 so,*
   A_SOLL = 100,000/271 = 369 cnt/smp^2 (= 0x0171)

Insert the calculated values in send frames and send to motor as shown below (modbus address 10-11 = register 5, modbus address 12-13 = register 6). Remember to press the send button for every new value.



Now set the motor in velocity mode and thereby activate the motor.
Ex. *The motor needs to be activated by setting it into velocity mode, so we need to set the mode register to the value 1. Mode = 1 which is velocity mode, now the motor will use the acceleration and the velocity just configured.* (Modbus address 4-5 = register 2).



7.4.2 Running Position control

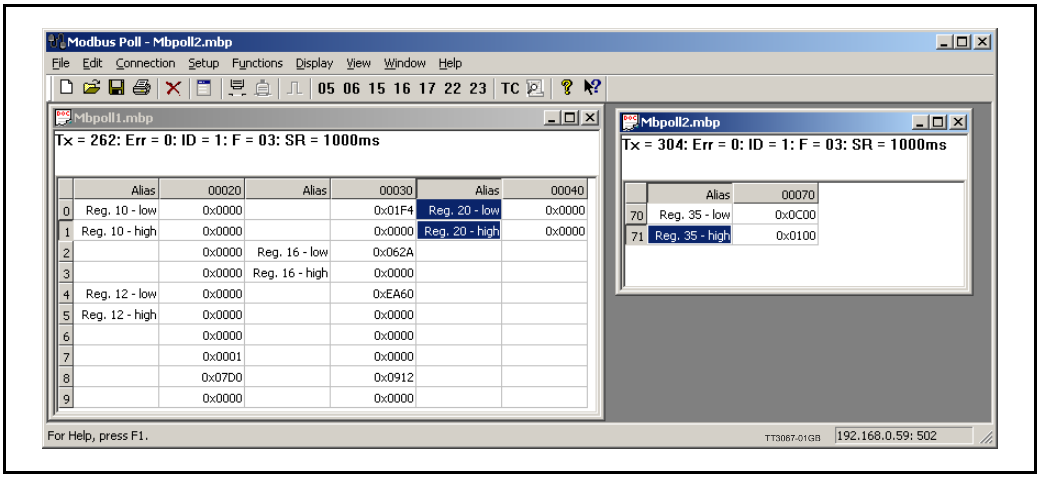
Running the motor in position control requires that the mode register is set for position control. The following registers is of particular interest when position mode is used.

- **Poll registers**
  - "Actual position" -P_IST, register 10
  - "Actual velocity" -V_IST, register 12
  - "Motor load mean" - average motor load, register 16
  - "Follow error" - The actual position error, register 20
  - "Error/Status" -register 35

- **Write registers**
  - “Mode” – mode, register 2
  - "Requested position" -P_SOLL, register 3
  - "Requested velocity" -V_SOLL, register 5
  - "Requested acceleration" -A_SOLL, register 6

In this mode the position is controlled by applying a requested position to the "P_SOLL" -register and the actual position is monitored in the "P_IST" register. The V_SOLL and A_SOLL registers sets the velocity and acceleration used when positioning occurs.

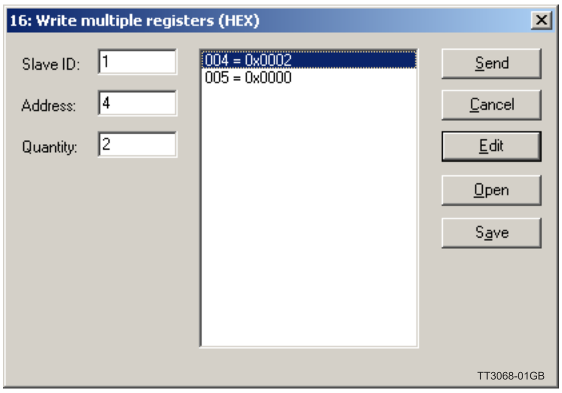
For easy setup we can use a single poll setup for the registers 10,12,16 and 20, and another for register 35, see figure below but it also is possibly to setup one poll instance for every single register.



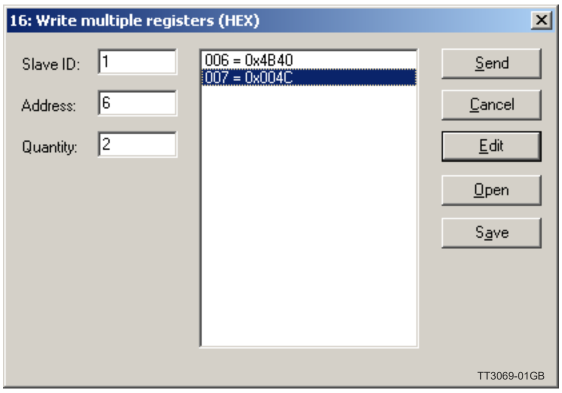
Calculate the values needed for velocity and acceleration and send to the motor, see previous example.

7.4 Examples

Now set the motor into position mode and thereby activate the motor.
Ex. *The motor needs to be activated by setting it into position mode so we need to set the mode register to the value 2. Mode = 2 which is position mode, now the motor will use the acceleration and the velocity just configured.* (Modbus address 4-5 = register 2).



Set a position in the motor by writing a position to register 3 (P_SOLL = Modbus address 6-7), in the example shown below is used position 5,000,000 (= 0x 004C 4B40), remark the order.



7.4.3 General considerations

The register 35 in the motor holds information on the actual error/status. So it is crucial that this register is configured in the polled data and thereby obtained and monitored in the Master. In case of an error situation the motor will stop and the cause will be present in the register 35.
This register also holds information on the motion status such as:

- In position, bit 4
- Accelerating, bit 5
- Decelerating, bit 6

Please find a complete list of register descriptions in the appendix:
Motor registers MAC050 - 141, page 292 or
Motor registers MAC400 - 4500, page 301 or
Motor registers MISxxx, page 320.

The JVL motor is basically put into a working mode and into a passive mode where the motor axle is de-energized, by setting register 2 into either 0 = "passive mode" or into one of the supported modes.

Example.
1= "Velocity mode" / 2= "Position mode" / etc.

8 Sercos®

8.1 Introduction to SERCOS

MAC - Modbus TCP/IP Module
Type:
MAC00-**ES**4 (shown) or
MAC00-**ES**41 (extended I/O)

To be used in following servo products:
MAC50, 095, 140 and 141
MAC400 and MAC402
MAC800
MAC1500 and MAC3000

MIS - Modbus TCP/IP motors.
Type:
MISxyyzz**ES**xx85 or
MILxyyzz**ES**xx85

TT3145-01GB

SERCOS® is an abbreviation for **SE**rial **R**ealtime **CO**mmunication **S**tandard, and is a globally standardized digital interface for communication between master control systems, drive units and other peripherals.

SERCOS® in its third edition – SercosIII – is a Real Time Ethernet technology which aims to maximize the use of the 100 Mbit, full duplex Ethernet bandwidth. It overcomes the overhead normally associated with Ethernet by employing "on the fly" processing.
A SERCOS® net consists of a master system and up to 511 slave devices, connected with standard Ethernet cabling.

The JVL implementation supports both the FSP IO profile and the FSP Drive profile. *FSP Drive profile, page 211* is dedicated the FSP Drive profile which is factory default.
FSP IO / JVL profile, page 223 is dedicated the FSP IO profile which is used by the “JVL” profile also used in other Ethernet based protocols.
The examples and screen shots in this manual are taken from MacTalk and the IndraWorks application for BoschRexroth masters, but please be aware that other SERCOS® PLC vendors exist
For more information on SERCOS® please visit the official homepage at this address:
https://www.sercos.org/

8.1 Introduction to SERCOS

8.1.1 Abbreviations

Following general used terms are useful to know before reading the following chapters.

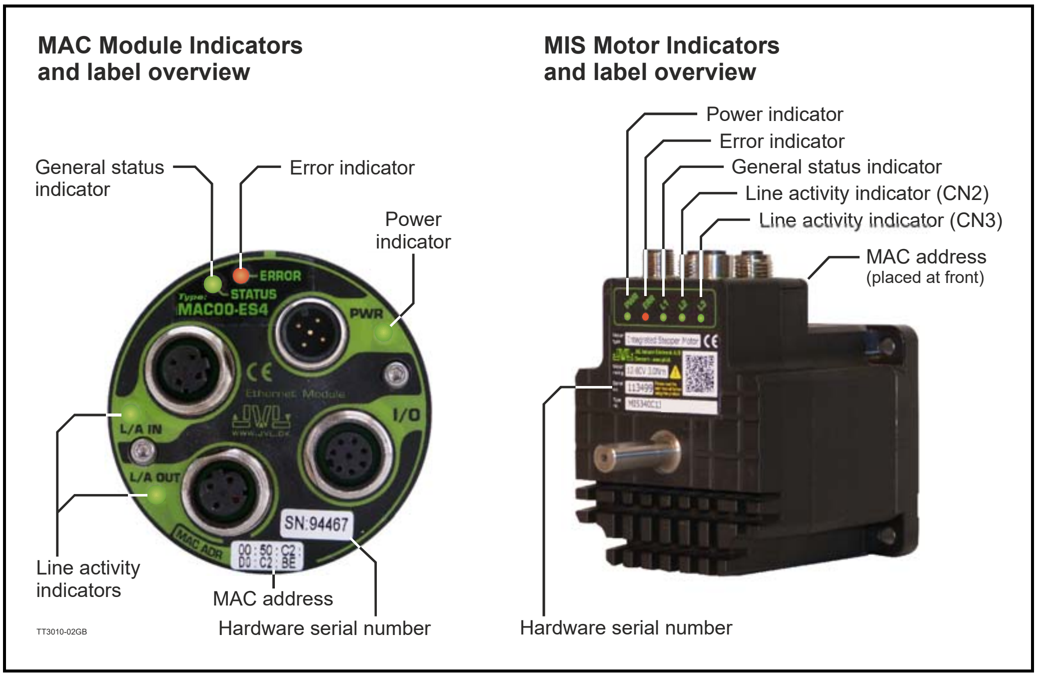
100Base-Tx	100 MBit Ethernet on twisted pairs.
AT	Acknowledge Telegram – Telegram from slave to master (motor to PLC).
Cyclic data	Data which is transferred at every cycle from master to slave or from slave to master.
CP0-CP4	Communication Phases 0-4.
FG	Function Group.
FSP	Function Specific Profile.
FSP IO	Function Specific Profile Input Output (Basic IO profile used by the JVL Profile).
FSP Drive	Function Specific Profile Drive (Standard Drive profile used by Sercos®).
GDP	Generic Device Profile
SDDML	Slave description file in the xml language. Found on the JVL homepage.
IDN	Identification Number (of a parameter).
IP	Internet Protocol - IP address ~ the logical address of the device, which is user configurable.
MAC	Media Access Controller - MAC address ~ the hardware address of the device.
MDT	Master Data Telegram – Telegram from master to slave (PLC to motor).
NRT	Non-Real Time. Default state before Sercos communication is started.
OL	Operation level (motor ready to operate). Normally coupled to communication state machine so motor is in OL when in CP4.
PL	Parameterization level. Normally coupled to communication state machine so motor is in PL when in CP2.
PackProfile	Drive profile originating from SercosII, but still in use (requires no XML file)
Sercos Drive	Drive which is PackProfile compatible.
SCP	Sercos® Communication Profile.

SCP_Fix	Cfg Sercos® Communication Profile Fix ConFiGuration (used by the JVL profile).
SCP_Var	Cfg Sercos® Communication Profile Variable Configuration (used by the FSP Drive).
SCP_Sync	Sercos® Communication Profile Synchronization.
Sercos®	Serial Realtime Communication System
SMP	Sercos® Messaging Protocol.
SVC	Service Channel (Channel for non-cyclic data).
UCC	Unified Communication Channel (UC Channel)

8.2 Commisioning

8.2.1 Indicators LED description

The LED's are used for indicating states and faults of the Sercos® module. There is one power LED, two link/activity LED's (one for each Ethernet connector), and 2 status LED's.



LED indicator descriptions

LED Text MAC / MIx	Colour	Constant off	Constant Orange	Constant Green	Constant Red	Flickering
L/A IN / L2	Green	No valid Ethernet connection.	-	Ethernet is connected.	-	Activity on line
L/A OUT / L3	Green	No valid Ethernet connection.	-	Ethernet is connected.	-	Activity on line
STATUS / L1	Multi color	- See table below -				
ERROR / ERR	Multi color	Sub Device not active	Parameteri-zation level	Operating level	Class 1 Diagnostics error	-
PWR / PWR	Red/ Green	Power is not applied.	Power is applied to both motor and module.	-	Only **MIS17x** and **MIS23x**: The Power supply is too low.	Power is ap-plied to module but no com-munication with motor.

Notes:
Flickering: Rapid flashing with a period of approx. 50ms (10 Hz).

8.2 Commisioning

STATUS/L1 led description.

Color 1	Color 2 (when flashing)	State	Description	Comments
Green	Green	On	CP4 (no error)	Normal operation, cyclic is running.
Green	Orange	Flashing	HP0 - HP2	Hot-plug modes. Flashes once for HP1, twice HP2, continuously for HP0
Green	Dark	Flashing	Loopback	-
Red	Red	On	Communication error.	See *Diagnosis (errors and warnings), page 206* for error codes.
Red	Green	Flashing	Communication error (MST Losses)	See *Diagnosis (errors and warnings), page 206* for error codes.
Red	Orange	Flashing	Application error	See *Diagnosis (errors and warnings), page 206* for error codes.
Red	Dark	Flashing	Watchdog error	Application is not running
Orange	Orange	On	CP0	-
Orange	Green	Flashing	CP1 - CP3	Flashes once for CP1, twice CP2, three times for CP3.
Orange	Dark	Flashing	Identification	-
Dark	Dark	Off	No Sercos® communication	-

8.2.2 Mechanical installation

The MAC00-ES4x Sercos® module and the MIS/MILxxxES Sercos® motor from JVL are both equipped with two M12 Ethernet ports, enabling the possibility for line and ring structures. The network cables must be connected to the two M12 connectors marked "L/A IN" and "L/A OUT" on the MAC motor module, and CN2 and CN3 at the MIS/MIL motors.

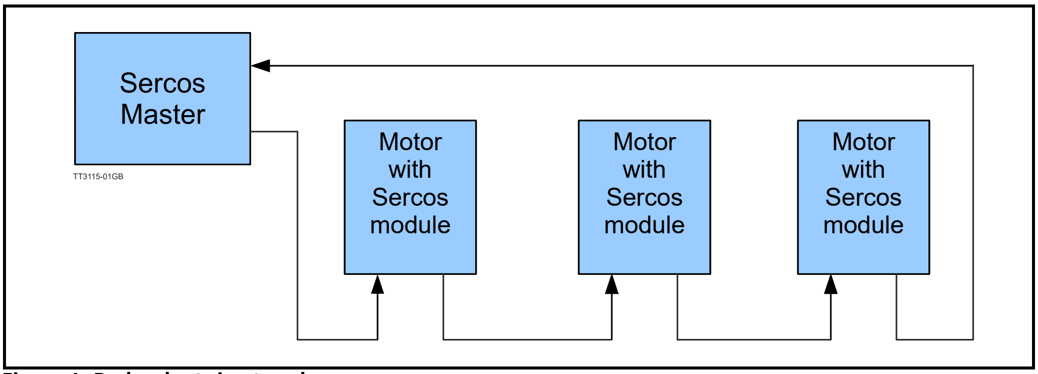


Figure 1: Redundant ring topology.

8.2 Commisioning

The cable from the Sercos® master is connected to one of the "L/A IN", "L/A OUT" ports. In the line topology, if there are more slave devices in the same line, the next slave device is connected to the other free port. If there is a redundant ring, the free port of the last slave device is connected to the second port of the Sercos® master. See the figure below. Standard CAT 5e FTP or STP cables can be used. It is not recommended to use UTP cables in industrial environments, which is typically very noisy.

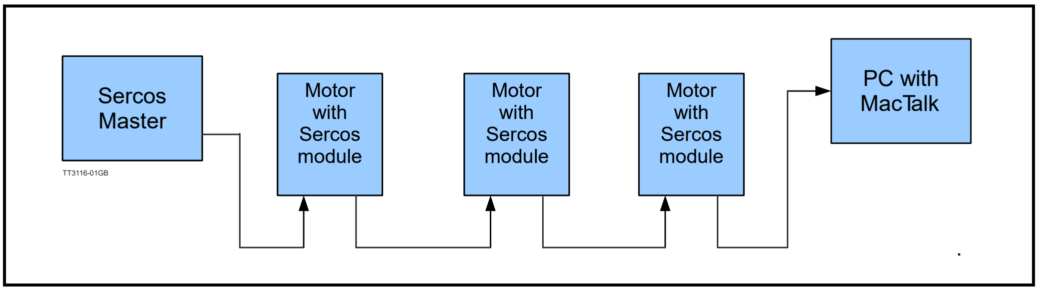


Figure 2: Line topology and MacTalk over Ethernet connection.

If connecting MacTalk over Ethernet to the motors with Sercos® modules, this is done to the free port of the last slave if using line topology. See figure above.

8.2.3 Connect to MacTalk

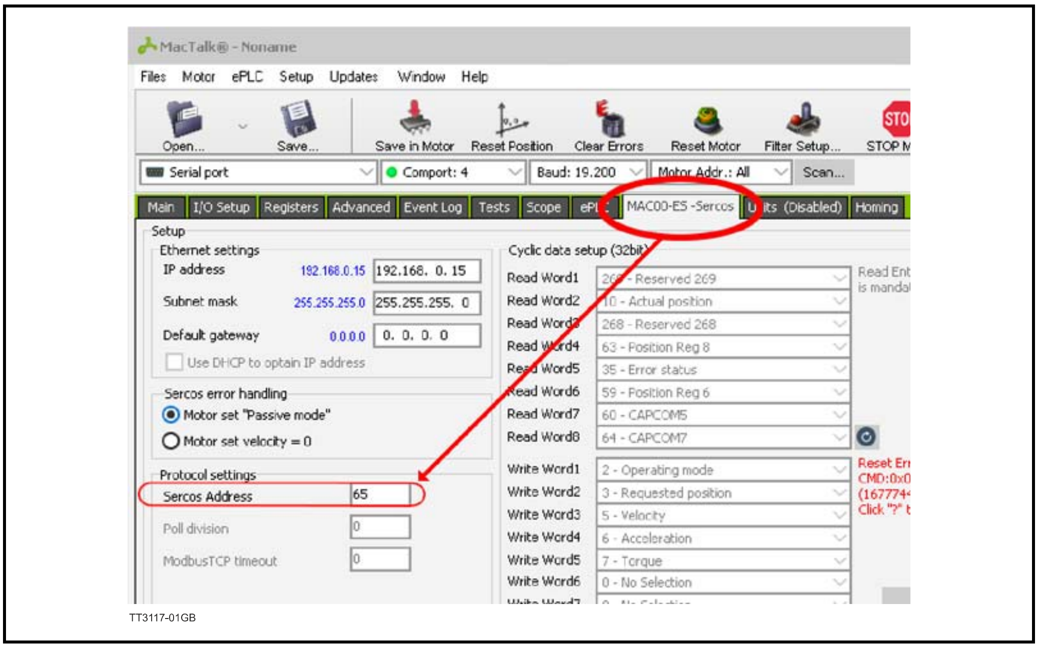
The easiest way to configure the JVL motor and the Sercos® module is with the MacTalk application.
MacTalk can be connected either by serial (RS232 for MAC motors / RS485 for MIS/MIL motors) or by Ethernet. Please refer to the below table for apropriate cable types.

Motor/Module type	JVL Cable type numbers for serial connection	JVL Cable type Ethernet connection
MAC00-ES4	**RS232-M12-1-5-8** + USB RS232 adaptor	WI1046-M12M4SxxNRJ45
MAC00-ES41	PA190 junction box + **RS232-M12-1-5-8** + USB RS232 adaptor.	WI1046-M12M4SxxNRJ45
MIS/MILxxxES	**RS485-M12-1-5-17** + USB RS485 adaptor	WI1046-M12M4SxxNRJ45

Serial connection is done directly from PC with one of the above mentioned cables and the apropriate USB to RS232 or USB to RS485 adaptor.
Ethernet connection is done to the free Ethernet port of the last slave in the Sercos® line please *please see “Mechanical installation” on page 185.*
For setting up the PC for connecting by Ethernet *please see “Using MacTalk over Ethernet” on page 270.*

8.2.4 Setting the slave address

The Sercos® IDN **S-0-1040** is the device address. This parameter can be written using the JVL MacTalk application. The address setting must be unique, meaning each address must only be used once in a Sercos® network. Sercos® also supports automatic slave addressing. If you enter the address 0 for all slave devices, the addressing is executed automatically by the master at power-up, ***if*** the used master supports automatic addressing.

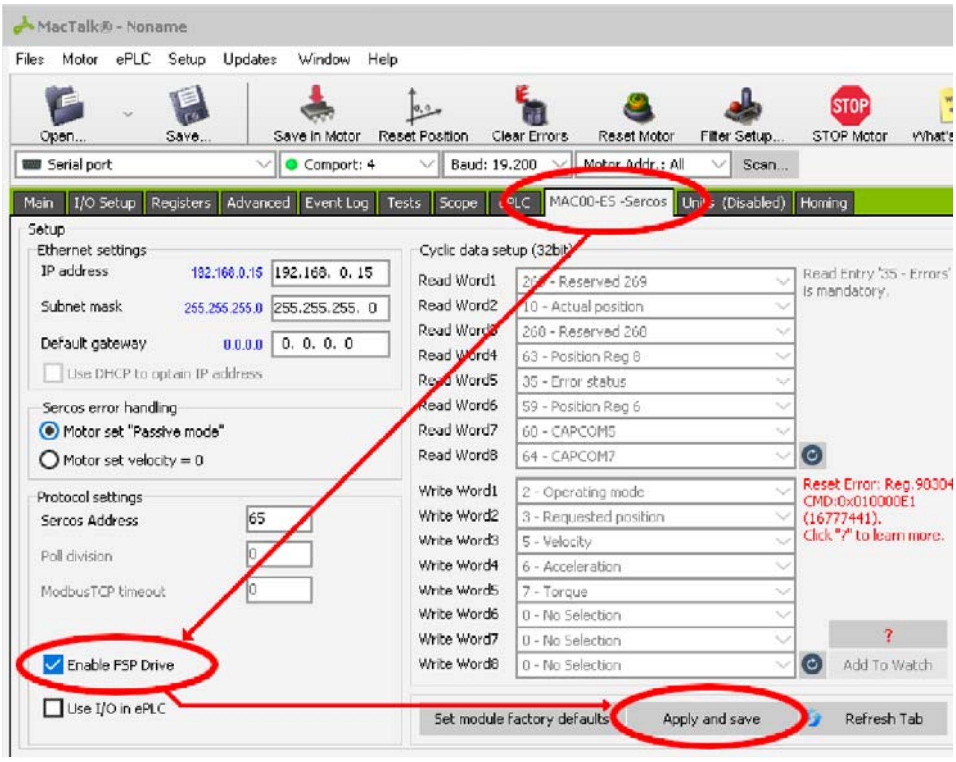


8.2.5 Quick start with Pack Profile and Bosch-Rexroth MLC/XLC master.

This paragraph guides you through the steps needed to jog the motor with PackProfile (Sercos® drive). With the Pack Profile no SDDML file is needed.

8.2.5.1 Enabling FSP Drive / Pack Profile.

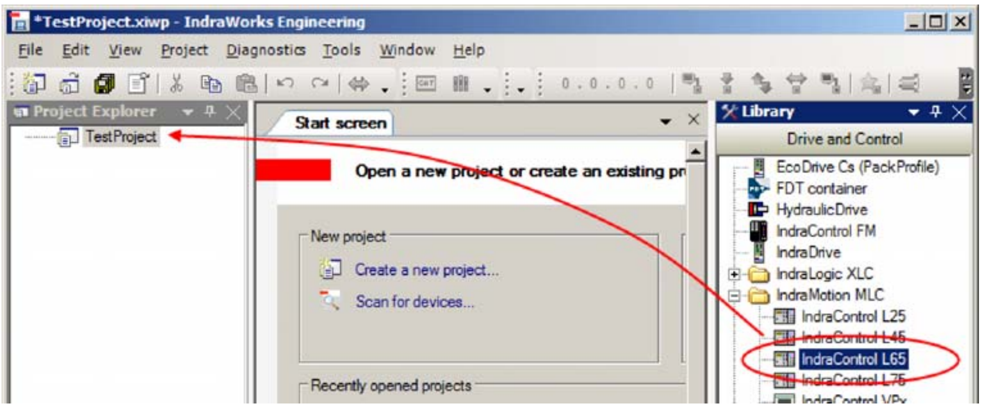
For using the Pack Profile the “FSP Drive / Pack Profile” has to be enabled in the JVL Sercos® module. This is factory default setting, but if not enabled already, then check the “FSP Drive / Pack Profile” in the Sercos® tab in Mactalk, and press the “Apply and save” button. Please refer to the picture below. For connecting MacTalk to the motor *please see “Connect to MacTalk” on page 186.*



TT3118-01GB

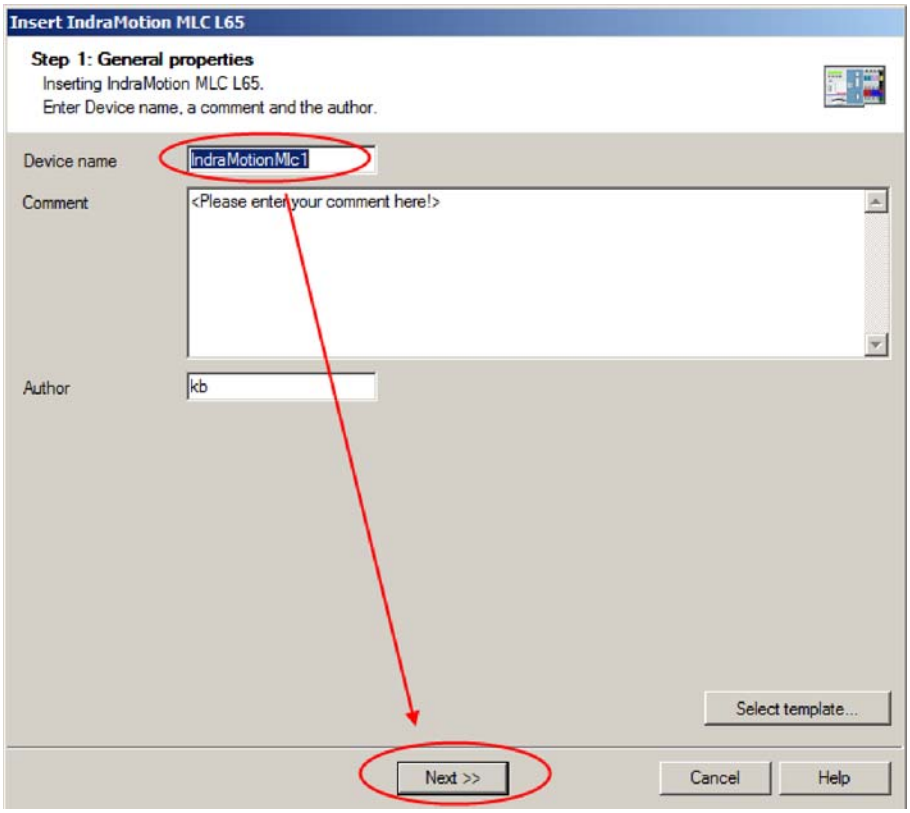
8.2.5.2 Add MLC/XLC master to IndraWorks project

Make sure the PC with IndraWorks is connected to MLC master Ethernet port, and the JVL motor L/A IN port is connected to the MLC master Sercos® port, and that all devices are correctly powered. The PWR and the L/A IN LED's should be on.
Start IndraWorks with an empty project.
Drag and drop the appropriate master from "library / Drive and control" to the project in "Project explorer".



TT3119-01GB

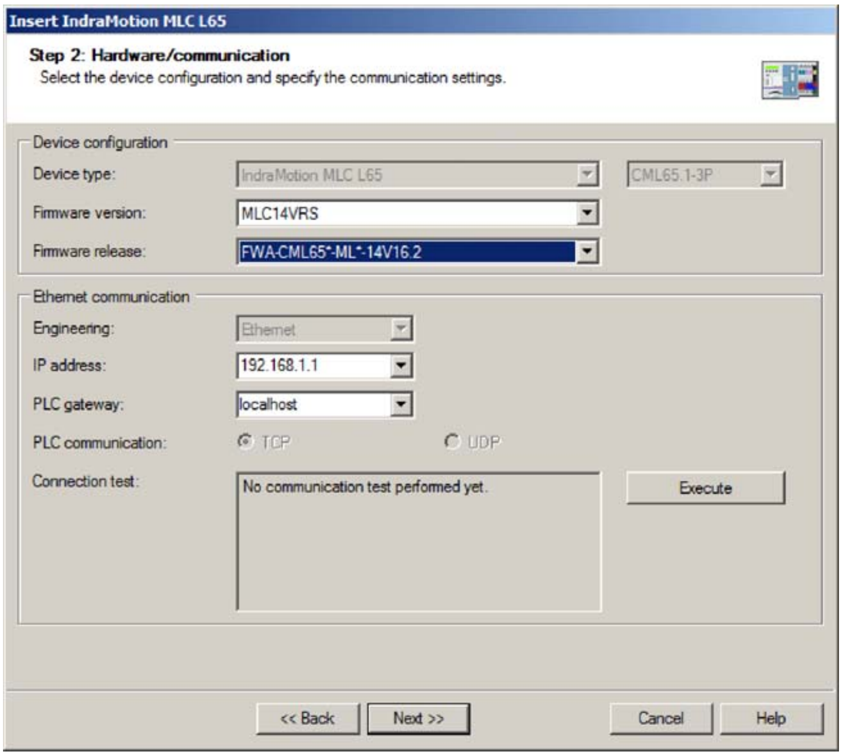
Key in the device name you want to use and press "Next"



TT3120-01GB

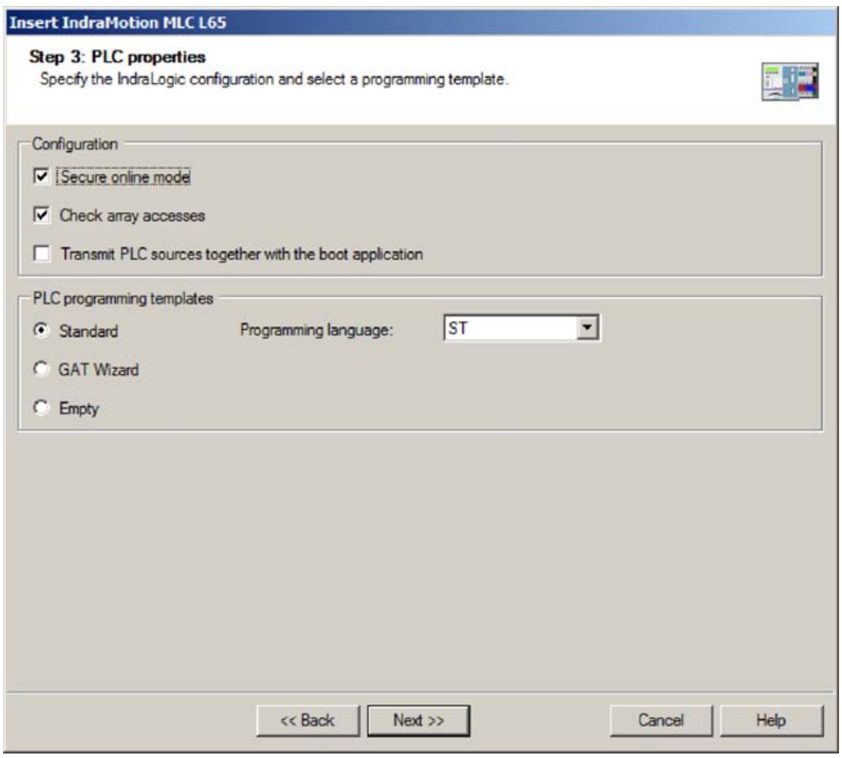
8.2 Commisioning

Make sure the correct IP address and device type is chosen, as shown below in the figure.



TT3121-01GB

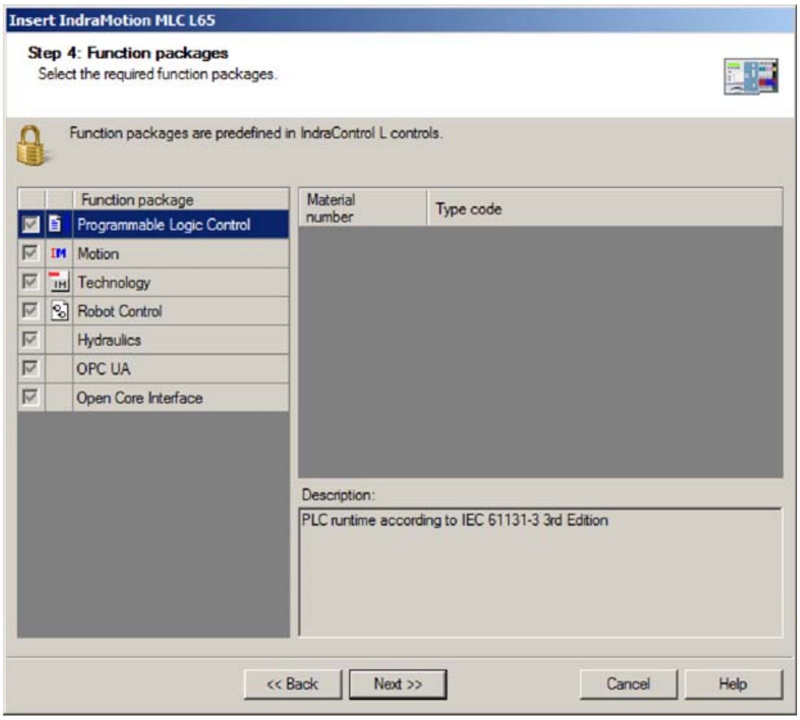
Select the programming language you prefer.



TT3122-01GB

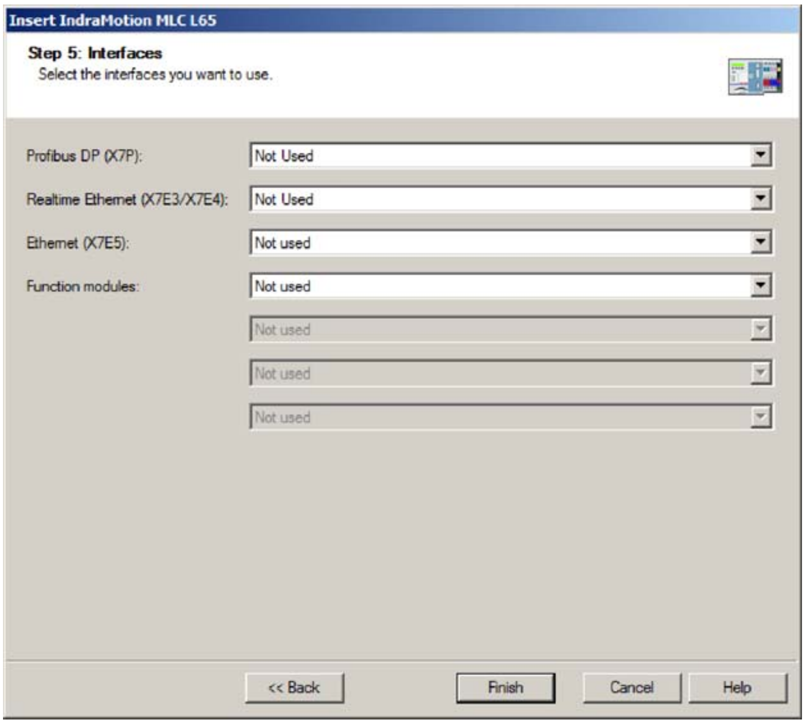
8.2 Commisioning

Press “Next” to step 4.



TT3123-01GB

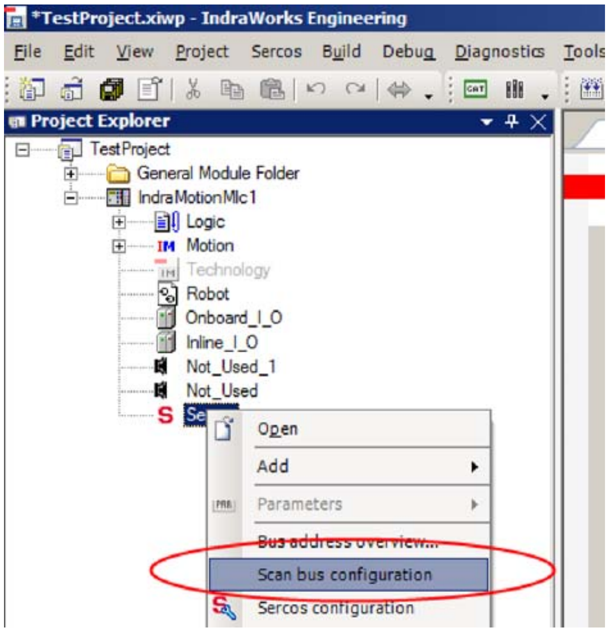
For some masters you may have to select the Sercos® interface. With the MLC L65 shown here it is always present. Press the finish button, and the MLC master is added.



TT3124-01GB

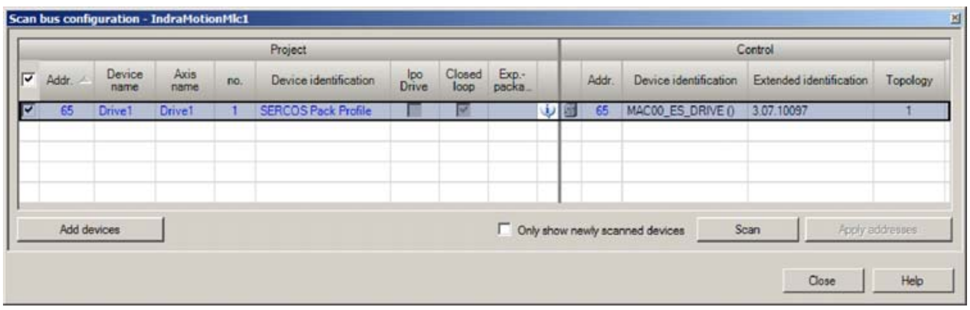
8.2.5.3 Add JVL MIS/MIL or MAC motor with PackProfile to IndraWorks project

Fold out the master options and add the motor by right clicking the “Sercos” entry and selecting the “Scan bus configuration”.



TT3124-01GB

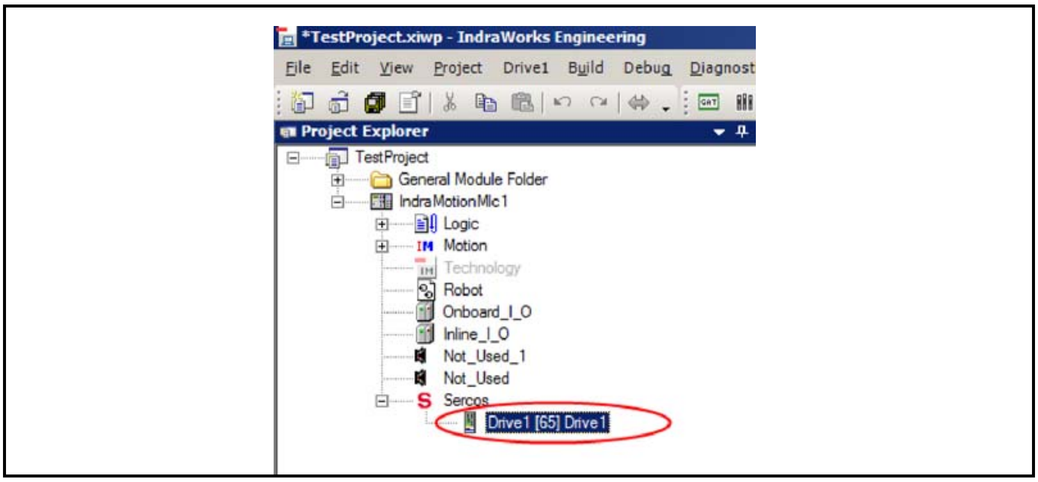
Then after a short while a new window should appear which should look something like this:



TT3126-01GB

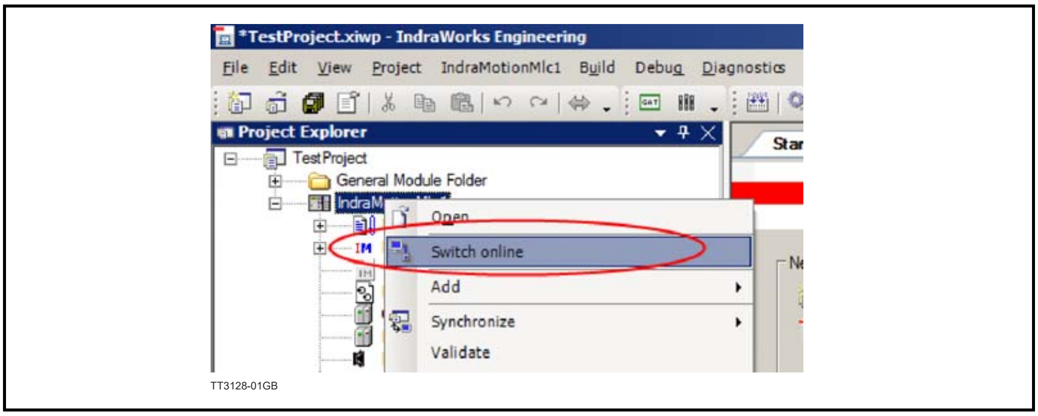
8.2 Commisioning

Press the “Add devices” button and then the “Close” button. Then the motor (Drive) should be added to the project, as shown below.

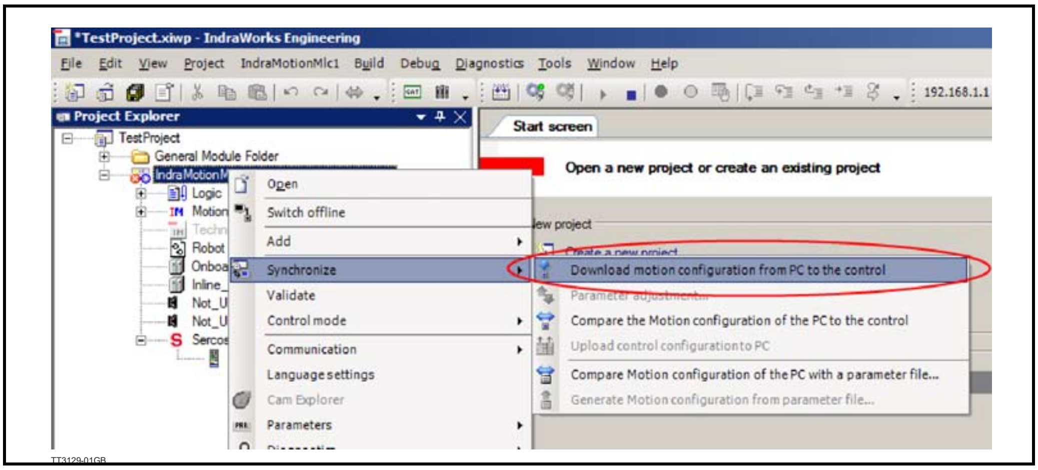


8.2.5.4 Jog the motor

Right click the master and select “Switch online”.

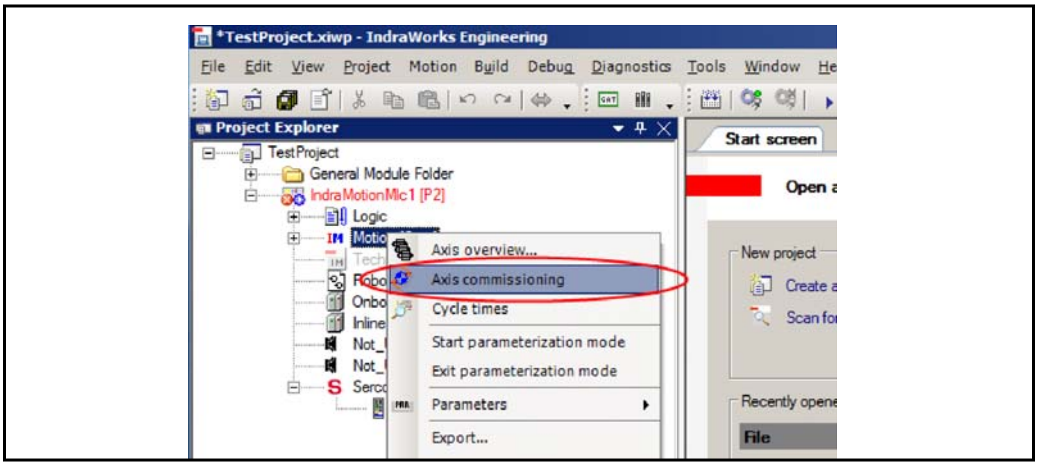


Then download the configuration to the Sercos® master by again right clicking the master and then choose “Synchronize” and “Download motion configuration from PC to the control”. Click “Yes” and “OK” to all pop-up windows that appear.

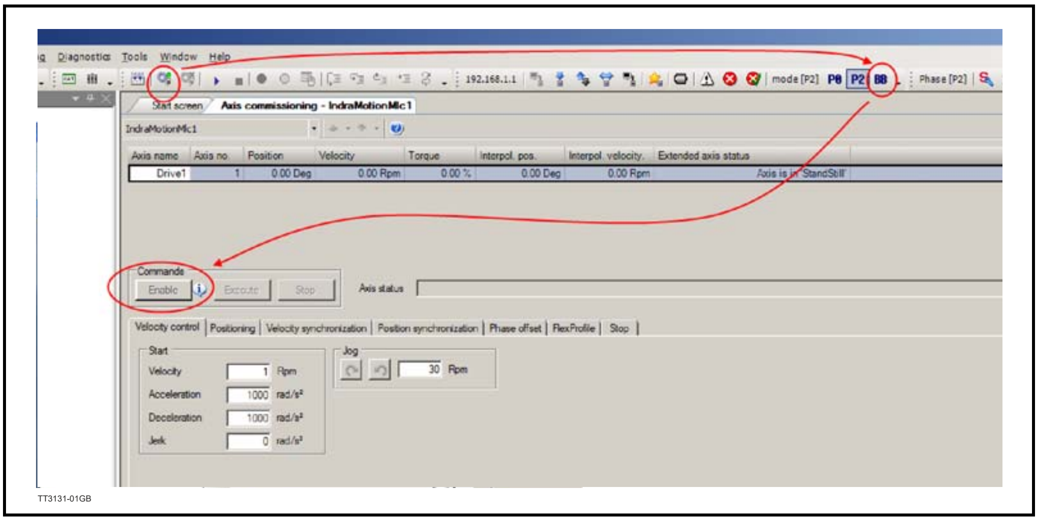


8.2 Commisioning

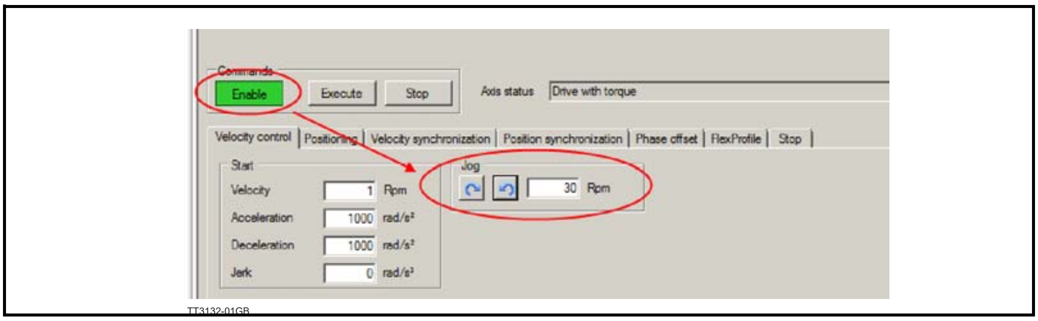
Right click the “Motion” entry and select “Axis commissioning”. Press “OK” to the warning that pops up.



By pressing the “Login” button and answer “yes” to all pop-up windows, and then pressing the “BB” button for shifting to phase 4, the “Enable” button should become active.



Then pressing the “Enable” button should enable the drive (torque on) and then the jog buttons and the other functions are active.



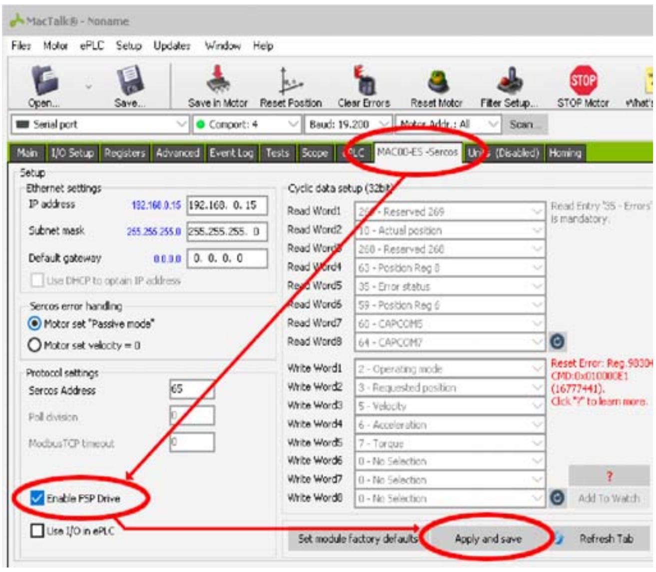
For making a PLC application and controlling the motor/drive this way, please consult the controller manual.

8.2.6 Quick start FSP Drive Profile and Bosch-Rexroth MLC/XLC master.

This paragraph guides you through the steps needed to run the motor with the FSP Drive Profile.

8.2.6.1 Enabling FSP Drive / Pack Profile.

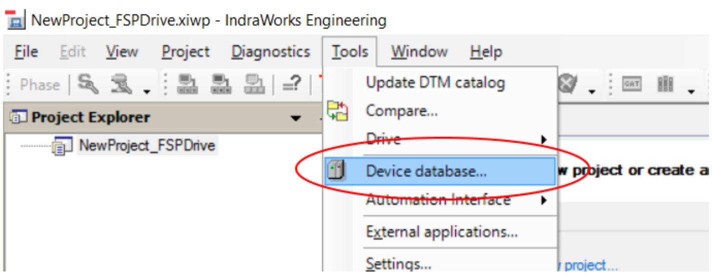
For using the FSPDrive Profile the “FSP Drive / Pack Profile” has to be enabled in the JVL Sercos® module. This is factory default setting, but if not enabled already, then check the “FSP Drive / Pack Profile” checkbox in the Sercos tab in Mactalk, and press the “Apply and save” button. Please refer to the picture below.
For connecting MacTalk to the motor please refer to *Connect to MacTalk, page 186*.



TT3118-01GB

8.2.6.2 Add SDDML file to IndraWorks

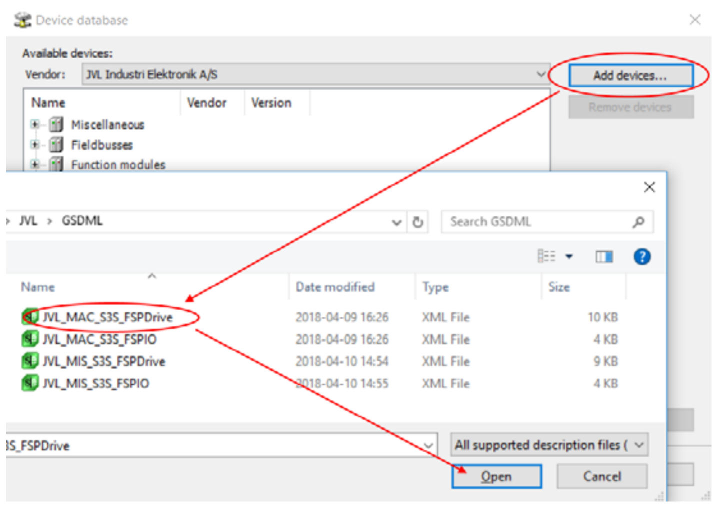
Start IndraWorks with an empty project.
Download the Sercos® SDDML file from the JVL homepage here:
http://www.jvl.dk/default.asp?Action=Details&Item=428
Install the SDDML file in IndraWorks by selecting “Device database” in the tools menu.



TT3133-01GB

8.2 Commisioning

Press the “Add devices” button and find the FSPDrive profile xml file for the used motor (MAC motor in this example), and then press the “Open” button. The SDDML is now installed and the window can be closed.

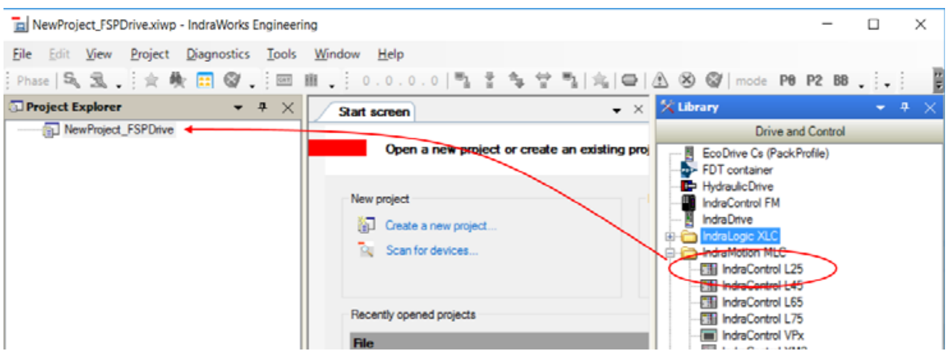


TT3134-01GB

8.2.6.3 Add MLC/XLC master to IndraWorks project

Make sure the PC with IndraWorks is connected to MLC master Ethernet port, and the JVL MAC motor L/A IN port or the JVL MIS/MIL motor CN2 port is connected to the MLC master Sercos® port, and that all devices are correctly powered. The PWR and the L/A IN LED’s should be on if MAC motor is used. If a MIS/MIL motor is used then the PWR and the L2 LED’s should be on.

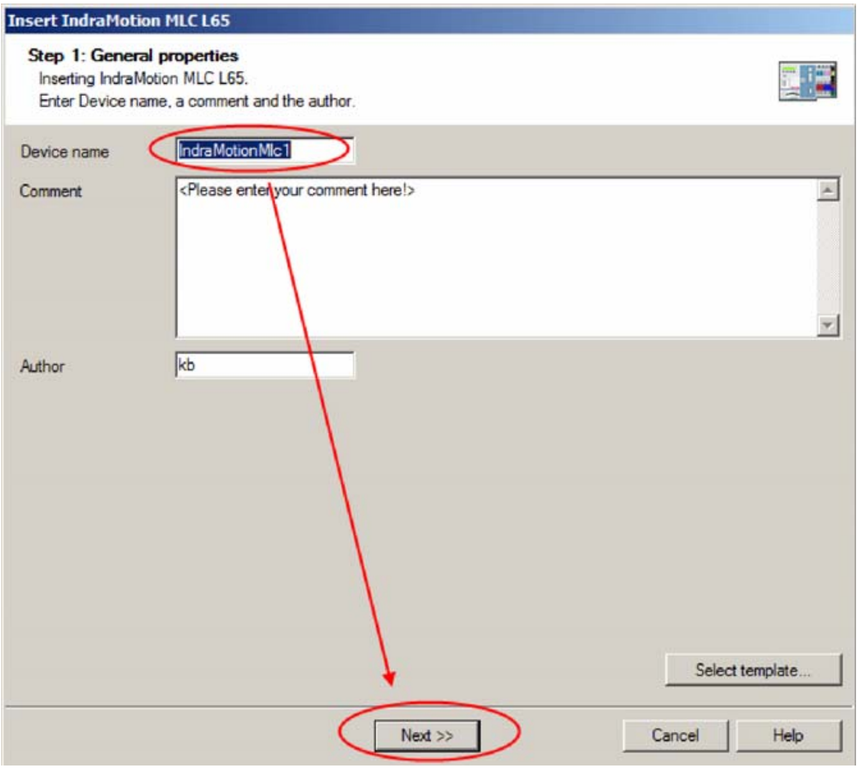
Drag and drop the appropriate master from “library / Drive and control” to the project in “Project explorer”.



TT3135-01GB

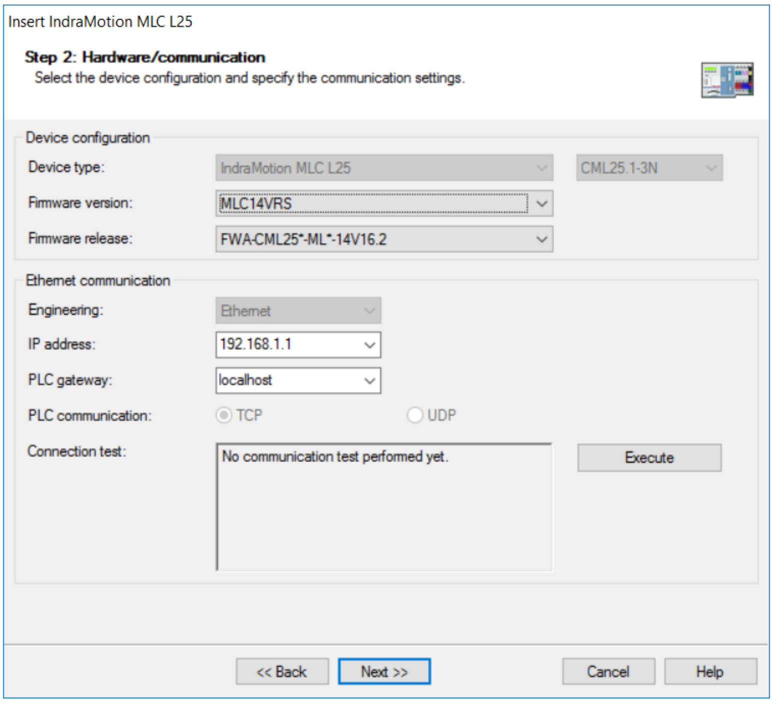
8.2 Commisioning

Key in the device name you want to use and press “Next”



TT3120-01GB

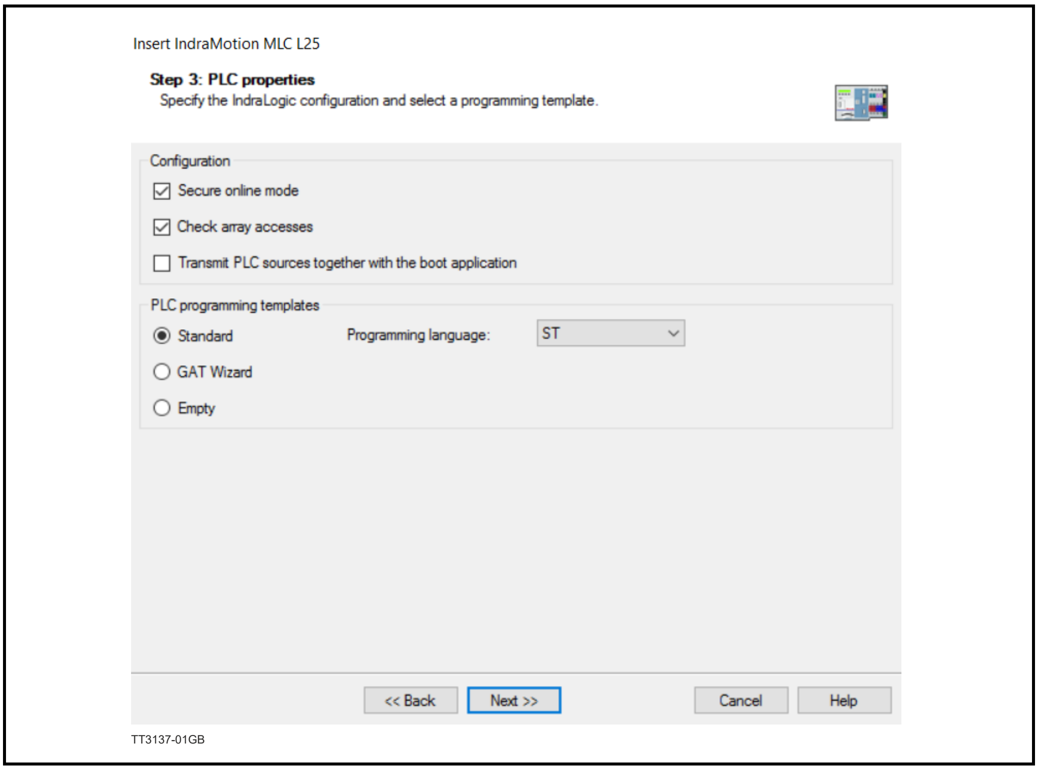
Make sure the correct IP address and device type is chosen, as shown below in the figure.



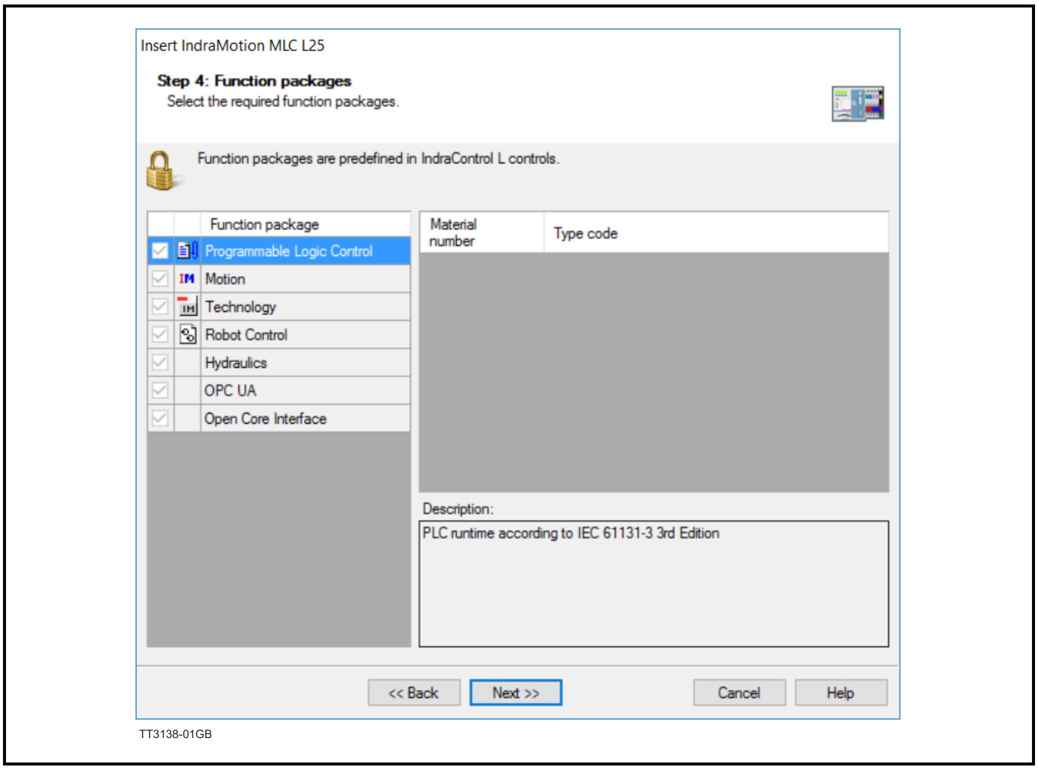
TT3136-01GB

8.2 Commisioning

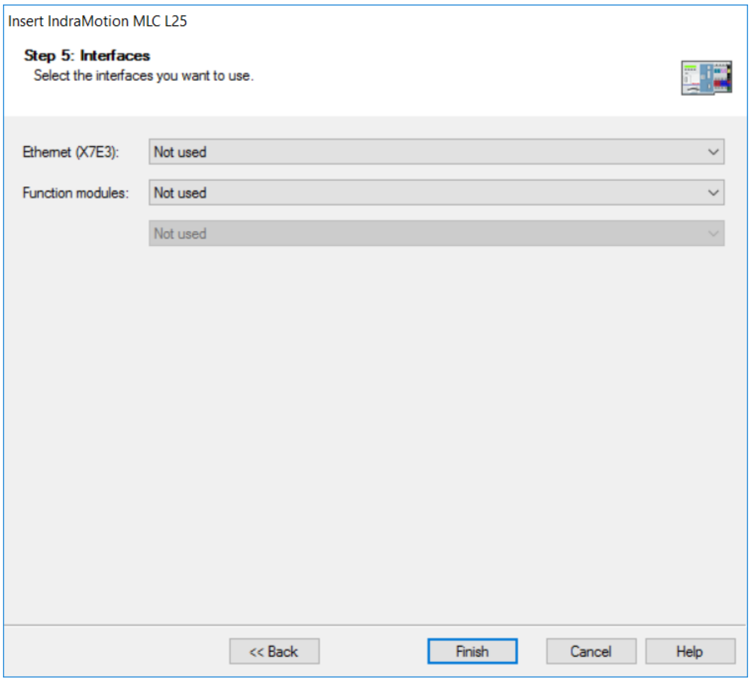
Select the programming language you prefer.



Press “Next” to step 4.



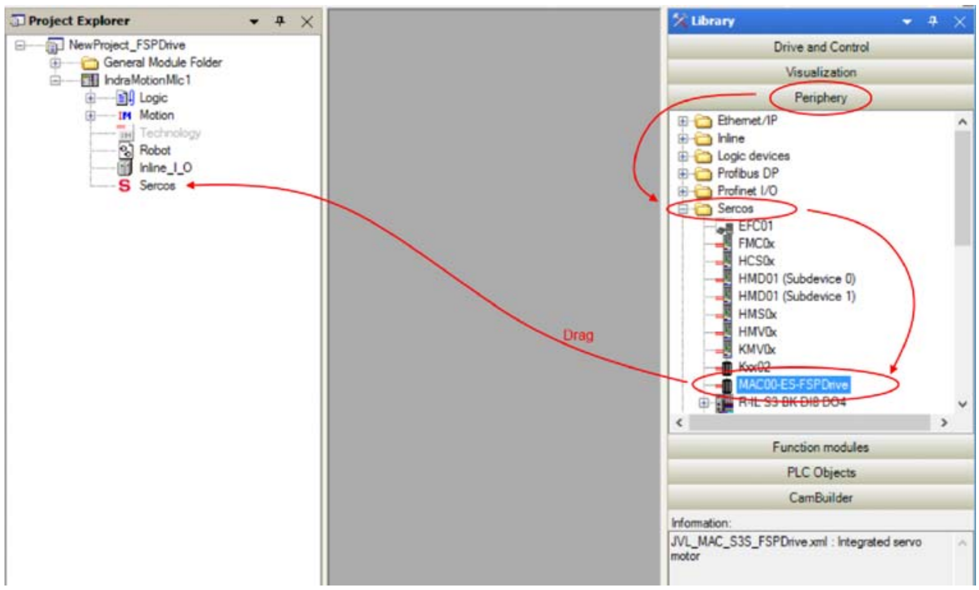
Press the finish button, and the MLC master is added.



TT3139-01GB

8.2.6.4 Add JVL motor with FSP Drive profile to IndraWorks project

In the “Library” section of IndraWorks select “Periphery” and “Sercos”. Then find the “MAC00-ES-FSPDrive” if using a MAC servo motor, or the “MIS/MILxxxxxxES-FSP-Drive” if using a MIS/MIL stepper motor. This is dragged and dropped on the “Sercos” in the “Project explorer” like shown below.



TT3140-01GB

8.3 Sercos Communication

8.3.1 Sercos® communication

The MAC00-ES4x Sercos® module and the MIS/MILxxxES Sercos® motor from JVL are both equipped with two M12 Ethernet ports, enabling the possibility for line and ring structures. The JVL Sercos® implementation is based upon the Sercos® International specification v.1.1.2.

8.3.2 Drive synchronization

The MAC00-ES4x Sercos® module and the MIS/MILxxxES Sercos® motor use the FSP Sync profile for synchronization of motors in the network. This is used by most masters for their FSP Drive/Pack profile implementation. The sync profile is supported by the JVL MAC00-ES4x module in MAC400-MAC4500 servo motors and by JVL MIS/MILxxxES stepper motors with embedded Sercos® module.

8.3.3 Sercos® state machine

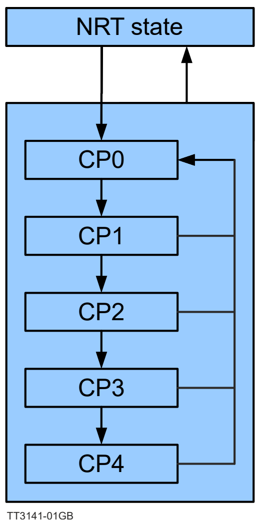
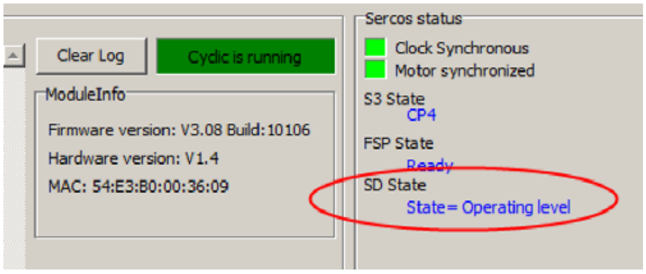


Figure 3: Sercos® communication state machine

Sercos® Communication between the master and the slaves is split into 6 phases (states). Please refer to the state machine figure. From power-up and until the slave receives Sercos® telegrams, the slave is in the NRT (Non-Real Time) state. Communication Phases 0 and 1 (CP0, CP1) identify the slaves on the net. In communication phase 2 (CP2) the time and data structure of the protocols for phase 3 and 4 (CP3, CP4) are prepared and the slaves are configured. At the transition to CP3 the slave parameter settings related to the Sercos® profiles are checked for plausibility. In the event of an error, the switch to CP3 is refused with an error message, and the slave remains in CP2. The phases are run through in ascending order. It is only possibly to drop back a phase by way of CP0. The communication phases are dictated by the master. When CP4 is reached, the initialization is completed.
Only in the case of a communication error the slaves shall switch to NRT state. The communication phases are indicated on the Sercos® status LED, in the front of the MAC motor Sercos® module, and on the back of the MIS/MIL motors.

8.3 Sercos Communication

It is also possible to see the communication phase in the “Sercos status” frame on the Sercos tab in MacTalk when connected. Please see below.



TT3142-01GB

8.3.4 Parameters

The parameters - hereafter referred to as – IDN’s, are presented in numerical order. IDNs with the prefix **S** indicate standard Sercos® implementation, while those with the prefix **P** indicate they are product-specific (manufacturer’s IDN’s).

IDN	Description	Default	Units	Attri-butes
S-0-0011 (4)	Class 1 diagnostic (C1D) – Error flags	-	-	RO, C
S-0-0012 (4)	Class 2 diagnostic (C2D) – Warning flags	-	-	RO, C
S-0-0013 (4)	Class 3 diagnostic (C3D) – Status flags	-	-	RO
S-0-0014	Interface status	-	-	RO
S-0-0015	Telegram type parameter	-	-	
S-0-0017	Displays a list of all IDNs supported by the drive	-	-	RO
S-0-0021	IDN list of invalid operation data for CP2.	-	-	RO
S-0-0022	IDN list of invalid operation data for CP3.	-	-	RO
S-0-0025	Displays a list of all procedure commands supported by the drive.	-	-	RO
S-0-0032 (4)	Gets/sets the primary operation mode of the drive	3	-	RW
S-0-0036 (4)	Velocity command value	-	-	RW, C
S-0-0040 (4)	Velocity feedback value.	-	-	RW, C
S-0-0041 (4)	Homing velocity.	50.0000	RPM	RW
S-0-0042 (4)	Homing acceleration	523.333	Rad/s^2	RW
S-0-0043 (4)	Velocity polarity parameter	0	-	RW
S-0-0044 (4)	Velocity data scaling type	2	-	RW
S-0-0045 (4)	Velocity scaling factor	1	-	RW
S-0-0046 (4)	Velocity scaling exponent	-2	-	RW
S-0-0047 (4)	Position command value	-	-	RW, C
S-0-0049 (4)	Positive position limit value	-	-	RW
S-0-0050 (4)	Negative position limit value	-	-	RW
S-0-0051 (4)	Position feedback value 1 (motor feedback)	-	-	RO, C
S-0-0052 (4)	Reference Position 1	0	-	RW
S-0-0055 (4)	Position polarity parameter	0	-	RW
S-0-0057 (4)	Position window	4.3902	°	RW
S-0-0076 (4)	Position data scaling type	2	-	RW
S-0-0077 (4)	Linear position data scaling factor	1	-	RW

8.3 Sercos Communication

IDN	Description	Default	Units	Attri-butes
S-0-0078 (4)	Linear position data scaling exponent	-2	-	RW
S-0-0079 (4)	Position resolution (counts/rot)	360.0000	°	RW
S-0-0080 (4)	Torque command value	-	-	RW
S-0-0082 (4)	Positive torque limit value	300.0	%	RW,C
S-0-0083 (4)	Negative torque limit value	300.0	%	RW,C
S-0-0084 (4)	Torque feedback value	-	-	RO, C
S-0-0085 (4)	Torque polarity parameter	0	-	RW
S-0-0086 (4)	Torque/force data scaling type	0	-	RW
S-0-0091 (4)	Bipolar velocity limit value	3000.0000	RPM	RW
S-0-0092 (4)	Bipolar torque limit value	300.0	%	RW, C
S-0-0093 (4)	Torque/force data scaling factor	1	-	RW
S-0-0094 (4)	Torque/force data scaling exponent	-2	-	RW
S-0-0095	Diagnostic message	-	-	RO
S-0-0099	Reset class 1 diagnostic procedure command (Clear errors)	-	-	RW
S-0-0103 (4)	Modulo value	147456000	-	RW
S-0-0113 (4)	Maximum motor speed	-	-	RO
S-0-0116 (4)	Resolution of feedback 1	-	-	RO
S-0-0121 (4)	Input revolutions of load gear	1	-	RW
S-0-0122 (4)	Output revolutions of load gear	1	-	RW
S-0-0123 (4)	Feed constant	1	-	RW
S-0-0127	CP3 transition check procedure command	-	-	RW
S-0-0128	CP4 transition check procedure command	-	-	RW
S-0-0129 (4)	Manufacturer class 1 diagnostic (MC1D) – Error flags	-	-	RO
S-0-0130 (3,4)	Probe value 1 positive edge	-	-	RO, C
S-0-0131 (3,4)	Probe value 1 negative edge	-	-	RO, C
S-0-0132 (3,4)	Probe value 2 positive edge	-	-	RO, C
S-0-0133 (3,4)	Probe value 2 negative edge	-	-	RO, C
S-0-0134 (4)	Drive control word	-	-	RW, C
S-0-0135 (4)	Drive status word	-	-	RO, C
S-0-0138 (4)	Bipolar acceleration limit value	523333.33 3	Rad/s^2	RW
S-0-0139 (4)	Park axis procedure command	-	-	RW
S-0-0147 (4)	Homing parameter	0	-	RW
S-0-0148 (4)	Drive controlled homing procedure command	-	-	RW
S-0-0159 (4)	Monitoring window	0	-	RW
S-0-0160 (4)	Acceleration data scaling type	2	-	RW
S-0-0169 (3,4)	Probe control	0	-	RW
S-0-0170 (4)	Probing cycle procedure command	-	-	RW
S-0-0175 (4)	Displacement feedback value encoder 1	-	-	RO
S-0-0181 (2,4)	Manufacturer class 2 diagnostic (MC2D) – Warning flags	-	-	RO
S-0-0187 (4)	IDN-list of configurable data as producer (AT data = Slave to Master)	-	-	RO
S-0-0188 (4)	IDN-list of configurable data as consumer (MDT data = Master to Slave)	-	-	RO
S-0-0189 (4)	Following error	-	-	RO, C
S-0-0191 (4)	Cancel reference point procedure command	-	-	RW
S-0-0192 (4)	IDN-list of all backup operation data	-	-	RO
S-0-0256 (4)	Multiplication factor 1	1	-	RO
S-0-0262 (4)	Load Defaults procedure command	-	-	RW
S-0-0263 (4)	Load working memory procedure command	-	-	RW
S-0-0264 (4)	Backup working memory procedure command	-	-	RW
S-0-0267	Password	-	-	RW
S-0-0279	List of password protected operation data	-	-	RW
S-0-0292 (4)	List of supported operation modes	-	-	RO

8.3 Sercos Communication

IDN	Description	Default	Units	Attri-butes
S-0-0390	Diagnostic message number	-	-	RO
S-0-0398 (3,4)	IDN list of configurable real-time bits as producer	-	IDN	RO
S-0-0399 (3,4)	IDN list of configurable real-time bits as consumer	0-0-0000	IDN	RO
S-0-0401 (3,4)	Probe 1	-	-	RO, C, T
S-0-0402 (3,4)	Probe 2	-	-	RO, C, T
S-0-0403 (4)	Position feedback value status (Returns the home status of the drive.)	-	-	RO
S-0-0405 (3,4)	Probe 1 enable	0	-	RW
S-0-0406 (3,4)	Probe 2 enable	0	-	RW
S-0-0409 (3,4)	Probe 1 positive latched	-	-	RO, C, T
S-0-0410 (3,4)	Probe 2 negative latched	-	-	RO, C, T
S-0-0411 (3,4)	Probe 1 positive latched	-	-	RO, C, T
S-0-0412 (3,4)	Probe 2 negative latched	-	-	RO, C, T
S-0-0420 (4)	Enter Parameterization Level procedure command	-	-	RW
S-0-0422 (4)	Exit Parameterization Level procedure command	-	-	RW
S-0-0423 (4)	IDN-list of invalid data for parameterization level	-	-	RO
S-0-0425 (4)	Sub-device state machine control	-	-	RW
S-0-0426 (3,4)	Measuring data allocation 1 (Probe 1 signal selection)	S-0-0051	IDN	RW
S-0-0427 (3,4)	Measuring data allocation 2 (Probe 2 signal selection)	S-0-0051	IDN	RW
S-0-0428 (3,4)	IDN list of configurable measuring data	S-0-0051	IDN	RO
S-0-0447 (4)	Set absolute position procedure command	-	-	RW
S-0-0448 (4)	Set absolute position control word	-	-	RW
S-0-0530 (4)	Clamping torque (Torque used for homing against a hard stop)	15.0	%	RW
S-0-0531	Checksum of backup operation data	-	-	RO
S-0-1000	List of SCP classes and version	-	-	RO
S-0-1002	Communication cycle time	-	-	RW
S-0-1003	Allowed MST losses	-	-	RW
S-0-1005	Minimum feedback processing time (t5)	-	-	RO
S-0-1006	AT0 transmission starting time (t1)	-	-	RW
S-0-1007	Feedback acquisition capture point (t4)	-	-	RW
S-0-1008	Command value valid time (t8)	-	-	RW
S-0-1009	Device control (C-Dev) offset in MDT	-	-	RW
S-0-1010	Length of MDT's	-	-	RW
S-0-1011	Device status (S-Dev) offset in MDT	-	-	RW
S-0-1012	Length of AT's	-	-	RW
S-0-1013	SVC offset in MDT	-	-	RW
S-0-1014	SVC offset in AT	-	-	RW
S-0-1015	Ring delay	-	-	RW
S-0-1016	Slave delay (P/S)	-	-	RO
S-0-1017	NRT transmission time	-	-	RW
S-0-1019	MAC address	54:E3:B0:xx:xx:xx	-	RO
S-0-1020	IP address	192.168.0.x	-	RW
S-0-1020.0.1	Current IP address	-	-	RO
S-0-1021	Subnet Mask	255.255.255.0	-	RW
S-0-1021.0.1	Current Subnet Mask	-	-	RO
S-0-1022	Gateway address	0.0.0.0	-	RW
S-0-1022.0.1	Current Gateway address	-	-	RO

8.3 Sercos Communication

IDN	Description	Default	Units	Attri-butes
S-0-1023	Sync jitter	-	-	RW
S-0-1024	SYNC delay measuring procedure command	-	-	RW
S-0-1026	Version of communication hardware	-	-	RO
S-0-1027.0.1	Requested MTU	-	-	RW
S-0-1027.0.2	Effective MTU	-	-	RO
S-0-1028	Error counter MST	-	-	RO
S-0-1031	Test pin assignment port 1 & port 2	-	-	RW
S-0-1032	Communication control	-	-	RW
S-0-1035	Error counter port 1 & port 2	-	-	RW
S-0-1035.0.1	Error counter P&S	-	-	RO
S-0-1036	Interframe gap	-	-	RW
S-0-1037	Slave jitter	-	-	RO
S-0-1039	Hostname	-	-	RW
S-0-1035.0.1	Current hostname	-	-	RO
S-0-1040	Sercos® address	-	-	RW
S-0-1041	AT command valid time (t9)	-	-	RW
S-0-1044	Device control	-	-	RO
S-0-1045	Device status	-	-	RO
S-0-1046	List of Sercos® addresses in device	-	-	RO
S-0-1047	Maximum consumer activation time (t11)	-	-	RO
S-0-1048	Activate network settings procedure command	-	-	RW
S-0-1050.0.1	Connection setup	-	-	RW
S-0-1050.0.2	Connection number	-	-	RW
S-0-1050.0.3	Telegram assignment	-	-	RW
S-0-1050.0.4	Max. length of connection	-	-	RO
S-0-1050.0.5	Current length of connection	-	-	RO
S-0-1050.0.6	Mapped cyclic consumer data (AT data = cyclic data from motor to PLC)	-	-	RW
S-0-1050.0.8	Connection control	-	-	RO
S-0-1050.0.10	Producer cycle time	-	-	RW
S-0-1050.0.11	Allowed data losses	-	-	RW
S-0-1050.0.12	Error counter data losses	-	-	RO
S-0-1050.0.20[3,4]	IDN allocation of real time bit	-	-	RW
S-0-1050.0.21[3,4]	Bit allocation of real time bit	-	-	RW
S-0-1050.1.1	Connection setup	-	-	RW
S-0-1050.1.2	Connection number	-	-	RW
S-0-1050.1.3	Telegram assignment	-	-	RW
S-0-1050.1.4	Max. length of connection	-	-	RO
S-0-1050.1.5	Current length of connection	-	-	RO
S-0-1050.1.6	Mapped cyclic producer data (MDT data = Cyclic data from PLC to motor)	-	-	RW
S-0-1050.1.8	Connection control	-	-	RO
S-0-1050.1.10	Producer cycle time	-	-	RW
S-0-1050.1.11	Allowed data losses	-	-	RW
S-0-1050.1.12	Error counter data losses	-	-	RO
S-0-1300.0.1	Component name	MAC00_ES _DRIVE	-	RO
S-0-1300.0.2	Vendor name	JVL	-	RO
S-0-1300.0.3	Vendor code	279	-	RO
S-0-1300.0.4	Device name	MAC00-ES-FSPDrive	-	RO

IDN	Description	Default	Units	Attri-butes
S-0-1300.0.5	Vendor device ID	MAC00_ES_DRIVE	-	RO
S-0-1300.0.9	Software revision	-	-	RO
S-0-1300.0.11	Order number	-	-	RO
S-0-1300.0.12	Serial number	-	-	RO
S-0-1301	List of GDP classes & version	-	-	RO
S-0-1302.0.1	FSP type and version	-	-	RO
S-0-1302.0.3	Application Type	-	-	RW
S-0-1303.0.2	Diagnosis trace control	-	-	RW
S-0-1303.0.3	Diagnosis trace state	-	-	RO
S-0-1303.0.10	Diagnosis trace buffer 1	-	-	RO
S-0-1303.0.11	Diagnosis trace buffer 2	-	-	RO
S-0-1305.0.1	Sercos® current time	-	-	RW
S-0-1601	List of FSP drive classes & version	-	-	RO
P-0-0001 to P-0-0255	Motor registers 1 – 255. **P-0-0001** = Motor register 1, **P-0-0002** = Motor register 2 etc. Please refer to section 13.2.1 for MAC050-141, section 13.3.1 for MAC400-4500, and section 13.4.1 for MIS/MILxxx motors.	-	-	-
P-0-0256 to P-0-0511	Motor registers 256 – 511. Please refer to appendix 11.3.	-	-	-
P-0-3001 to P-0-3063	Sercos® communication module registers 1-63. **P-0-3001** = Sercos® module register 1, **P-0-3002** = Sercos® module register 2 etc. Please refer to section 10.1.1 for detailed descriptions.	-	-	-

"RO" = Read Only, "RW"=Read and Writeable, "C"=Map able to Cyclic data, "T"=Map able to real Time bits.
(1) Only available in MAC00-ESx (MAC servo motor Sercos® module).
(2) Only available in MIS/MILxxxxxxESxxxxx (MIS/MIL stepper motor with integrated Sercos® module).
(3) Only available in MAC400-MAC4500 with MAC00-ES4x.
(4) Only available if drive profile (FSP Drive) selected.

8.3.5 Non-cyclic data via the service channel (SVC)

All S and P parameters listed in the table in the previous paragraph can be read via the service channel. Write access is only possible to parameters which are not write protected. The service channel is initialized during CP1 and is available in CP2-CP4. The service channel is controlled by the SVC controlword. The status is present in the SVC status word. The IDN procedure commands are also transferred by the service channel.

8.3.6 Cyclic data

In cyclic data transfer, new parameters are transferred with every cycle of the net. The parameters transferred are defined by the mapping. With the JVL profile the mapping of the cyclic data is done in MacTalk by selecting motor and/or module registers for cyclic transfer and save this mapping in flash in the Sercos® module. With the FSP Drive this is done in the Sercos® master, by writing to the IDN's **S-0-1050.x.6** in communication phase CP2. In communication phase CP3 these IDN's are then setup in the Sercos® slave for cyclic communication. The JVL Sercos® module only uses freely dynamically cyclic mapping. The standard telegrams from earlier generations of the Sercos® protocol are not supported. The IDN's **S-0-0187** and **S-0-0188** contains lists of the IDN's which are cyclic map able.

8.3 Sercos Communication

Used IDN's for setup of cyclic data with FSP Drive:

IDN	Description	Default	Units	Attributes
S-0-0187 (4)	IDN-list of configurable data as producer (AT data = Slave to Master)			RO
S-0-0188 (4)	IDN-list of configurable data as consumer (MDT data = Master to Slave)			RO
S-0-1050.0.6	Mapped cyclic consumer data (AT data = cyclic data from motor to PLC)			RW
S-0-1050.1.6	Mapped cyclic producer data (MDT data = Cyclic data from PLC to motor)			RW

"RO" = Read Only, "RW" = Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.

(4) Only available if drive profile (FSP Drive) selected.

8.3.7 Diagnosis (errors and warnings)

Used IDN's in Diagnosis.

IDN	Description	Default	Units	Attributes
S-0-0011 (4)	Class 1 diagnostic (C1D) – Error flags	-	-	RO, C
S-0-0012 (4)	Class 2 diagnostic (C2D) – Warning flags	-	-	RO, C
S-0-0013 (4)	Class 3 diagnostic (C3D) – Status flags	-	-	RO
S-0-0095	Diagnostic message	-	-	RO
S-0-0099	Reset class 1 diagnostic procedure command (Clear errors)	-	-	RW
S-0-0129 (4)	Manufacturer class 1 diagnostic (MC1D) – Error flags	-	-	RO
S-0-0181 (2,4)	Manufacturer class 2 diagnostic (MC2D) – Warning flags	-	-	RO
S-0-0390	Diagnostic message number	-	-	RO
S-0-1303.0.2	Diagnosis trace control	-	-	RW
S-0-1303.0.3	Diagnosis trace state	-	-	RO
S-0-1303.0.10	Diagnosis trace buffer 1	-	-	RO
S-0-1303.0.11	Diagnosis trace buffer 2	-	-	RO

"RO" = Read Only, "RW" = Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.

(2) Only available in MIS/MILxxxxxxESxxxxx (MIS/MIL stepper motor with integrated Sercos® module).

(4) Only available if drive profile (FSP Drive) selected.

The JVL Sercos® module supports the standard error bits in IDN **S-0-0011** (C1D), standard warning bits in **S-0-0012** and also some manufacturer specific error bits in IDN **S-0-0129** and some manufacturer specific warning bits in **S-0-0181**. The error bits can be cleared with the command IDN **S-0-0099**. All message, warning and error numbers are saved in the IDN **S-0-1303.0.x** with a time stamp, up to a maximum of 512 entrys.

8.3.7.1 Error messages (C1D)

Errors (Class 1 Diagnostics) are signaled by the slave by setting bit 13 in the status word (**S-0-0135**), and the course of the error is then shown in the parameter **S-0-0011** or **S-0-0129**. And in IDN **S-0-0390** is the error number and in **S-0-0095** the corresponding error description. If an error is indicated in **S-0-0129** then also bit 15 of **S-0-0011** is set. The error bits can be cleared with the command IDN **S-0-0099**.

8.3 Sercos Communication

The definitions of the standard error bits of **S-0-0011** are shown here below.

Bit no	Status code in S-0-0390 (hex)	Error name	Error description	Applicable to motortype		
				MAC050-MAC141	MAC400-MAC4500	MISxxx MILxxx
0	C00F8055	I2T_ERR	Overload	X	X	
1			Amplifier overheating (not supported)			
2	C00F2019	DEGC_ERR	Motor overheating		X	X
3			Cooling error (not supported)			
4	C00F8070	U24V	Control voltage error		X	
5	C00F8022 / C00F2174	SSI_ERR / INDEX_ERR / ENC_COMERR	MAC050-141: SSI read error. MAC400+: Encoder error. MIS/MILxxx: Encoder lost position/ Encoder communication error/ Encoder reed error	X	X	X
6			Error in autocommutation (not supported)			
7	C00F8060	IPEAK_ERR	Overcurrent		X	
8	C00F8025	OV_ERR	Overvoltage		X	X
9	C00F2026	UV_ERR	Undervoltage on bus / Low AC voltage	X	X	X
10		IX_ERR	Phase error in power supply	X		
11	000F9001	PWM_LOCKED	Excessive control deviation		X	
12	000F900A	INT_COM_ERR	Communication error – internal	X	X	X
13	C00F6029	PLIM_ERR	Position limit value exceeded	X	X	X
14			Reserved			
15			Manufacturer specific error (see error in S-0-0129)	X	X	X

The definitions of the manufacturer specific error bits of **S-0-0129** are shown here below.

Bit no	Status code in S-0-0390 (hex)	Error name	Error description	Applicable to motortype		
				MAC050-MAC141	MAC400-MAC4500	MISxxx MILxxx
0	000F9006	INIT_ERR	Self diagnostic failed		X	X
1	000F900C	STO_ALARM_ERR	Safe Torque Off circuit failure		X	X
2	000F9008	FLASH_ERR	Error in write to the internal flash		X	
3	000F9004	OLD_FILTER	Invalid filter settings		X	
4	000F9003	UIT_ERR	Regenerative overload	X	X	
5	C00F2028	FLW_ERR	Follow error	X	X	X
6	C00F8079	SPEED_ERR	Overspeed		X	
7	000F9002	FNC_ERR	Function error	X	X	
8	000F9006	SLAVE_ERR	Slave error		X	
9-10			Reserved			
11	000F9005	COM_ERR	Modbus communication error		X	
12	000F9009	SYNC_ERR	PLL has lost synchronization		X	
13	000F900B	OUT_DRV_ERR	Output driver error			X
14	C00F6034	STO_TRIG	Safe Torque Off (Emergency stop) triggered		X	X
15			Reserved			

8.3.7.2 Warnings (C2D)

Warnings (Class 2 Diagnostics) are signaled by the slave by setting bit 12 in the status word (**S-0-0135**), and the course of the warning is then shown in the IDN **S-0-0012** for standard warnings and in IDN **S-0-0181** for manufacturer specific warnings. And in IDN **S-0-0390** is the warning number and in **S-0-0095** the corresponding warning description.

The definitions of the standard warning bits of **S-0-0012** are shown here below.

Bit no	Status code in S-0-0390 (hex)	Warning name	Error description	Applicable to motortype		
				MAC050-MAC141	MAC400-MAC4500	MISxxx MILxxx
0-8			Reserved			
9	C00E2026	UV_DETECT	Undervoltage in power section		X	
10-14			Reserved			
15			Manufacturer specific warning (see S-0-0181)			X

The definitions of the manufacturer specific warning bits of **S-0-0181** are shown here below. This IDN is only present with MIS motors.

Bit no	Status code in S-0-0390 (hex)	Warning name	Error description	Applicable to motortype		
				MAC050-MAC141	MAC400-MAC4500	MISxxx MILxxx
0	C00E8029	Positive limit	Positive position limit exceeded			X
1	C00E8030	Negative limit	Negative position limit exceeded			X
2-15			Reserved			

8.3.7.3 Status flags (C3D)

Status flags (Class 3 Diagnostics) are signaled by setting a bit in IDN **S-0-0013**, and the course of the status is then also given.

The definitions of the status bits of **S-0-0013** are shown here below.

Bit no	Status code in S-0-0390 (hex)	Warning name	Error description	Applicable to motortype		
				MAC050-MAC141	MAC400-MAC4500	MISxxx MILxxx
0-3			Reserved			
4		Torque limit	The torque limit set in IDN S-0-0092 reached		X	X
5		Speed limit	The speed limit set in IDN S-0-0091 reached		X	X
6		In position	The motor is positioned within the programmed limits of commanded position.		X	X
7-15			Reserved			

8.3.8 Procedure commands

Some of the IDN's are so called procedure commands. This means they are not used for holding any information, but are used for sending commands from the master to the Sercos® slave.
The command is executed when the value '3' (bit 0 and 1 set) is written to the command IDN. For the slave to prepare for a new command the master has to reset the command to '0' (zero) first.

The list of IDN's which are procedure commands is shown here:

IDN	Description	Default	Units	Attri-butes
S-0-0025	Displays a list of all procedure commands supported by the drive.	-	-	RO
S-0-0099	Reset class 1 diagnostic procedure command (Clear errors)	-	-	RW
S-0-0127	CP3 transition check procedure command (executed automatically by master)	-	-	RW
S-0-0128	CP4 transition check procedure command (executed automatically by master)	-	-	RW
S-0-0139 (4)	Park axis procedure command	-	-	RW
S-0-0148 (4)	Drive controlled homing procedure command	-	-	RW
S-0-0170 (4)	Probing cycle procedure command (Obsolete – not recommended for new designs)	-	-	RW
S-0-0191 (4)	Cancel reference point procedure command	-	-	RW
S-0-0262 (4)	Load Defaults procedure command	-	-	RW
S-0-0263 (4)	Load working memory procedure command	-	-	RW
S-0-0264 (4)	Backup working memory procedure command	-	-	RW
S-0-0420 (4)	Enter Parameterization Level procedure command	-	-	RW
S-0-0422 (4)	Exit Parameterization Level procedure command	-	-	RW
S-0-0447 (4)	Set absolute position procedure command	-	-	RW

"RO" = Read Only, "RW" = Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.
(4) Only available if drive profile (FSP Drive) selected.

8.3.9 IP Channel

The IP or NRT channel is primarily used for setup and diagnostic purposes.
With the IP channel it is possible to access all JVL slaves in a Sercos® network using the JVL utility application MacTalk. This can be done directly with a PC or a notebook connected to the free Ethernet port of the last slave if using line setup. Or it can be connected to the master, if the master supports the IP channel. If using the ring setup it is necessary to connect via the master.
For the IP channel to work the slaves has to have correct and compatible IP setup (IP address, subnet mask and gateway address).

A list of IDN's related to the IP channel with default addresses are shown here:

IDN	Description	Default	Units	Attributes
S-0-1019	MAC address	54:E3:B0: xx:xx:xx	-	RO
S-0-1020	IP address	192.168.0.x	-	RW
S-0-1020.0.1	Current IP address	-	-	RO
S-0-1021	Subnet Mask	255.255. 255.0	-	RW
S-0-1021.0.1	Current Subnet Mask	-	-	RO
S-0-1022	Gateway address	0.0.0.0	-	RW
S-0-1022.0.1	Current Gateway address	-	-	RO

"RO" = Read Only, "RW" = Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.

The default last octet of the IP address is calculated from the last octet of the MAC address printed on the sticker of the MAC Sercos® module or the Sercos® MIS/MIL motor. For example if the MAC-ID of a JVL motor is: 54 : E3 : B0 : D0 : C9 : 14, then the IP address is set to: 192.168.0.20. (The MAC address is in hexadecimal and the IP address is in decimal, so converting the 14 from the MAC-ID from hexadecimal to decimal gives 20 in decimal). The IP address of every device in the network has to be unique, and the first octets as indicated by the subnet mask have to be the same in devices that wish to communicate. If the IP setup values are changed, by means of MacTalk then remember to save the values in flash by pressing the “Apply and save” button after the values are changed.

8.3.10 Accessing motor registers and Sercos® module registers

Besides all the standard IDN’s the JVL Sercos® module also support direct accessing of all the native registers in the Sercos® communication module and in the MAC/MIS/MIL motor controller. They are accessed over the service channel using the **P** IDN’s.

IDN	Description	Default	Units	Attri-butes
P-0-0001 to P-0-0255	Motor registers 1 – 255. **P-0-0001** = Motor register 1, **P-0-0002** = Motor register 2 etc. Please refer to appendix section 13.2.1 for MAC050-141, appendix section 13.3.1 for MAC400-4500, and appendix section 13.4.1 for MIS/MILxxx motors.	-	-	-
P-0-0256 to P-0-0511 (3)	Motor registers 256 – 511. Please refer to appendix *Motor registers MAC400 - 4500, page 301, Motor registers MISxxx, page 320*	-	-	-
P-0-3001 to P-0-3063	Sercos® communication module registers 1-63. **P-0-3001** = Sercos® module register 1, **P-0-3002** = Sercos® module register 2 etc. Please refer to chapter 9 for detailed descriptions.	-	-	-

“RO” = Read Only, ”RW” = Read and Writeable, ”C” = Map able to Cyclic data, “T” = Map able to real Time bits.
(3) Only available in MAC400-MAC4500 with MAC00-ES4x.

8.4 FSP Drive profile

The FSP Drive profile is supported by the JVL MAC00-ES4x module when inserted in MAC400 – MAC4500 motors and by MIS/MILxxxES motors. With these motors the FSP Drive profile is factory default, but if not enabled for some reason, please refer to *section 8.2.5.1 Enabling FSP Drive / Pack Profile*. The FSP Drive profile is not supported when the MAC00-ES4x module is inserted in the MAC050 – MAC141 motors.
Please refer to *section 8.5 FSP IO / JVL profile* for using these motors.

8.4.1 Supported cycle times (Pack Profile / FSP Drive and FSP Sync)

When using the Pack Profile or the FSP Drive profile the JVL motors support different cycle times, depending of the motor type. If the ***motion*** cycle time is larger than the ***Sercos*** cycle time then the drive performs an internal interpolation of the cyclic position values if running in position mode. With all supported cycle times the JVL motor are synchronized with the master. Please see supported cycle times in the table below.

	Motor series		
	MAC050 - MAC141	**MAC400 - MAC4500**	**MIS/MILxxx**
FSP Drive *Sercos* cycle times	Only support FSP IO profile without sync.	1, 2 ms	1, 2, 3, 4, 6, 8*, 16**ms
FSP Drive *motion* cycle times	Only support FSP IO profile without sync.	1 – 32 ms in integer multiples of ***Sercos*** cycle time	1 – 32 ms in integer multiples of ***Sercos*** cycle time

* Not recommended at motor speeds above 1000 RPM.
** Not recommended at motor speeds above 500 RPM.

If operating with other values than listed then the motor will refuse to go to CP4.

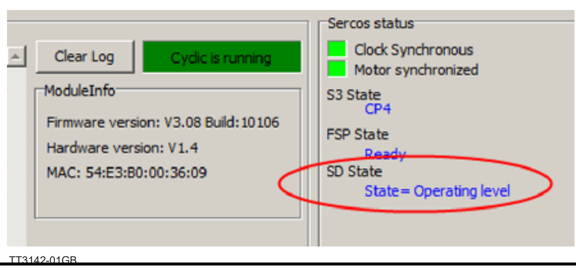
8.4.2 Sub-Device state machine

The JVL Sercos® implementation support the GDP_StM (**G**eneric **D**evice **P**rofile, class **S**tate**M**achine).
Used IDN's with State machine (GDP_StM):

IDN	Description	Default	Units	Attributes
S-0-0420	Enter Parameterization Level procedure command	-	-	RW
S-0-0422	Exit Parameterization Level procedure command	-	-	RW
S-0-0423	IDN-list of invalid data for parameterization level	-	-	RO
S-0-0425	Sub-device state machine control	-	-	RW

"RO" = Read Only, "RW" = Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.

In addition to the communication state machine there is a sub-device state machine, which represents the operating state of the Sercos® application. In general the sub-device state machine has two states, the operating level and the parameterization level. The sub-device state (SD State) can be monitored in the Sercos status frame on the Sercos tab in MacTalk as shown below.



8.4.2.1 Parameterization level (PL)

Parameterization level can be used to change parameters that are write protected in operating level and for which write protection is bound to the sub-device state machine, without leaving communication phase CP3/CP4. This can be useful, if application specific parameters of a specific sub-device have to be adjusted while the applications of other connected slaves remain activated. With the activation of the parameterization level the corresponding monitoring bits of **S-0-0135**, Drive status bits 15 and 14 are set to 0. Parameters which are write protected in parameterization level (PL) cannot be changed.

8.4.2.2 Operating level (OL)

The sub-device is ready for running the application. All monitoring systems are switched on. Parameters which are write protected in operating level (OL) cannot be changed. Bit 4 (Sub-device level) of the Device Status word is set to "0" (operating level (OL) is active).

8.4.2.3 Control of sub-device state machine

In order to control the state machine of the sub-device independent from the communication state machine, the following commands can be executed:

S-0-0420 Activate parametrization level procedure command (PL)
S-0-0422 Exit parameterization level procedure command

The sub-device state machine is normally coupled with the communication state machine. So in communication phases CP0, CP1, CP2 the sub-device state machine is in parameterization level, and in communication phase CP3 and CP4 the sub-device state machine is in operating level.
It is possible to decouple this connection by setting bit 1 of **S-0-0425** Sub-device state machine control.

8.4.3 Operation modes

Used IDN's for drive control, independent of selected operation mode:

IDN	Description	Default	Units	Attributes
S-0-0032	Gets/sets the primary operation mode of the drive	3	-	RW
S-0-0134	Drive control word	-	-	RW, C
S-0-0135	Drive status word	-	-	RO, C
S-0-0292	List of supported operation modes	-	-	RO

“RO” =Read Only, ”RW” = Read and Writeable, ”C” = Map able to Cyclic data, “T” = Map able to real Time bits.

The JVL Sercos® module supports the following operation modes of FSP Drive which is setup in IDN **S-0-0032:**

Mode	Value in S-0-0032	
	Decimal	Hexadecimal
Velocity control	2	0x02
Position control (***default setting***)	3	0x03
Velocity control without following distance	10	0x0A
Position control without following distance	11	0x0B
Synchronous Velocity control	66	0x42
Synchronous Position control	67	0x43

8.4 FSP Drive profile

The requested mode is setup in **S-0-0032** before shifting to operating level (CP4), or in parameterization level if sub-device state machine is used.

8.4.3.1 Controlling the motor.

The motor are controlled with the **S-0-0134** Drive control word and the motor status is shown in **S-0-0135** Drive status word. These IDN's should always be mapped to the cyclic communication MDT and AT.

Drive status word

The JVL Sercos® implementation supports the following bits/features of the drive status word:

Bit	Description	Comments	Supported
15-14	Ready to operate 00 - Drive not ready 01 - Drive ready for main power on 10 - Drive ready and main power applied 11 - Drive enabled	PL is active OL is active, main power off OL is active, main power on, no torque OL is active, torque on	Yes
13	Drive shut-down error in C1D (**S-0-0011**)	1 = Drive is shut-down due to error	Yes
12	Warning in C2D (**S-0-0012**)	When a warning occurs this bit is set to 1	Yes
11	Reserved	-	-
10-8	Current operation mode	= 000. Only supported primary operation mode, where mode is in **S-0-0032**	No
7	Emergency stop	Emergency stop is active	No
6	Reserved	-	-
5	Position feedback value status	1 = Position feedback is referenced to machine zero point	Yes
4	Drive halt	1 = Drive halt is active	Yes
3	Status command value processing	0 = Motor do not follow the commands 1 = Motor follows the commands	Yes
2-0	Reserved	-	-

Drive control word:

Bit	Description	Comments	Supported
15	Drive ON/OFF	0 = Motor OFF 1 = Motor ON	Yes
14	Drive enable	0 = Motor disable 1 = Motor enable	Yes
13	Drive start	0 = Motor halt 1 = Motor start	Yes
12-11	Reserved	-	-
10-8	Selection of Operation mode	Should always be 000	No
7-0	Reserved	-	-

The actual use of **S-0-0134** and **S-0-0135** follows the Sercos® standard version 1.1.2 and is not described in detail here.

8.4.3.2 Operation mode velocity

Operation mode velocity is selected by writing the value “2” to IDN **S-0-0032.** In this state the drive shall follow the velocity command values of the control unit. The velocity loop can be closed with the **S-0-0040** Velocity feedback value. The velocity feedback value refers to the motor encoder.

Used IDN’s in operation mode velocity:

IDN	Description	Default	Units	Attributes
S-0-0036	Velocity command value	-	-	RW, C
S-0-0040	Velocity feedback value.	-	-	RW, C
S-0-0043	Velocity polarity parameter	0	-	RW
S-0-0044	Velocity data scaling type	2	-	RW
S-0-0045	Velocity scaling factor	1	-	RW
S-0-0046	Velocity scaling exponent	-2	-	RW

“RO” = Read Only, ”RW” = Read and Writeable, ”C” = Map able to Cyclic data, “T” = Map able to real Time bits.

8.4.3.3 Operation mode position

Operation mode position is the default mode when using FSP Drive. In the operation mode position control the control unit transmits with its cycle time the position command value to the drive with the **S-0-0047** Position command value. The position loop can be closed with the **S-0-0051** Position feedback value 1, which refers to the motor encoder.

Used IDN’s in operation mode position:

IDN	Description	Default	Units	Attributes
S-0-0047	Position command value	-	-	RW, C
S-0-0051	Position feedback value 1 (motor feedback)	-	-	RO, C
S-0-0055	Position polarity parameter	0		RW
S-0-0057	Position window	4.3902	°	RW
S-0-0076	Position data scaling type	2	-	RW
S-0-0077	Linear position data scaling factor	1	-	RW
S-0-0078	Linear position data scaling exponent	-2	-	RW
S-0-0079	Position resolution (counts/rotation)	360.0000	°	RW
S-0-0189	Following error	-	-	RO, C

“RO” = Read Only, ”RW” = Read and Writeable, ”C” = Map able to Cyclic data, “T” = Map able to real Time bits.

8.4.4 Scaling

With Sercos a number of different scaling methods are possible, including rotational and linear scaling for both position data and velocity data

8.4.4.1 Position scaling

Used IDN's for position scaling:

IDN	Description	Default	Units	Attributes
S-0-0055	Position polarity parameter	0		RW
S-0-0076	Position data scaling type	2	-	RW
S-0-0077	Linear position data scaling factor	1	-	RW
S-0-0078	Linear position data scaling exponent	-2	-	RW
S-0-0079	Position resolution (counts/rotation)	360.0000	°	RW

"RO" = Read Only, "RW" = Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.

With the **S-0-0055** Position polarity parameter, it is possible to invert the position command and/or the feedback values.
The JVL Sercos® implementation supports the following bits of IDN **S-0-0055** position polarity parameter:

Bit	Description	Affected IDN's	Comments	Supported
15-6	Reserved		-	-
5	Underflow / Overflow threshold	-		No
4	Position limit values	S-0-0049 S-0-0050	0 = non-inverted ; 1 = inverted	Yes
3	Position feedback value 2	-		No
2	Position feedback value 1	S-0-0051	0 = non-inverted 1 = inverted	Yes
1	Additive position command value	-		No
0	Position command values	S-0-0047	0 = non-inverted 1 = inverted	Yes

With the IDN **S-0-0076** Position data scaling type, it is possible to setup the type of scaling used for the position IDN's **S-0-0047**, **S-0-0051**, **S-0-0049** and **S-0-0050**.
The JVL Sercos® implementation supports the following bits of IDN **S-0-0076** position data scaling type:

Bit	Description	Bit pattern	Function	Supported
15-8	Reserved		-	-
7	Processing format	0 1	Absolute format Modulo format (S-0-0103)	Yes
6	Data reference	0 1	At the motor shaft At the load (S-0-0121, S-0-0122, S-0-0123)	Yes
5	Reserved	-	-	-
4	Units for scaling	-	-	No
3	Scaling mode	0 1	Preferred scaling * Parameter scaling (S-0-0077, S-0-0078, S-0-0079)	Yes
2-0	Scaling method	000 001 010 011-111	No scaling Linear scaling Rotational scaling Reserved	Yes

* The preferred rotational scaling is 10^{-4}° (=> 360° = 3.600.000). The preferred linear scaling is 10^{-7}m (= 0,1μm).

In the **S-0-0077** and **S-0-0078** IDN's are setup the scaling factor and scaling exponent used when linear scaling is selected in **S-0-0076** bit 3 and bit 2-0.
In the IDN **S-0-0079** is setup the number of counts per rotation used in rotational scaling.

8.4.4.2 Velocity scaling

Used IDN's for velocity scaling:

IDN	Description	Default	Units	Attributes
S-0-0043	Velocity polarity parameter	0	-	RW
S-0-0044	Velocity data scaling type	2	-	RW
S-0-0045	Velocity scaling factor	1	-	RW
S-0-0046	Velocity scaling exponent	-2	-	RW

"RO" = Read Only, "RW" = Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.

With the **S-0-0043** Velocity polarity parameter, it is possible to invert the velocity command and/or the feedback values.
The JVL Sercos® implementation supports the following bits of IDN **S-0-0043** velocity polarity parameter:

Bit	Description	Affected IDN's	Comments	Supported
15-4	Reserved		-	-
3	Velocity feedback value 2	-		No
2	Velocity feedback value 1	S-0-0040	0 = non-inverted ; 1 = inverted	Yes
1	Additive velocity command value 1	-		No
0	Velocity command value 1	S-0-0036	0 = non-inverted ; 1 = inverted	Yes

With the **S-0-0044** Velocity data scaling type, it is possible to setup the type of scaling used for the velocity IDN's **S-0-0036** and **S-0-0040**.
The JVL Sercos® implementation supports the following bits of IDN **S-0-0044** velocity data scaling type:

Bit	Description	Bit pattern	Function	Supported
15-7	Reserved		-	-
6	Data reference	0 1	At the motor shaft At the load (S-0-0121, S-0-0122, S-0-0123)	Yes
5	Time units	-	-	No
4	Units for scaling	-	-	No
3	Scaling mode	0 1	Preferred scaling * Parameter scaling (S-0-0045, S-0-0046)	Yes
2-0	Scaling method	000 001 010 011-111	No scaling Linear scaling Rotational scaling Reserved	Yes

* The preferred rotational scaling is 10^{-4} min^{-1} (=> 1 RPM = value 1000). The preferred linear scaling is 0,001 mm/min.

In the **S-0-0045** and **S-0-0046** IDN's are setup the scaling factor and scaling exponent used when linear parameter scaling is selected in **S-0-0043** bit 3 and bit 2-0.

8.4.4.3 Acceleration scaling

Used IDN's for acceleration scaling:

IDN	Description	Default	Units	Attributes
S-0-0160	Acceleration data scaling type	2	-	RW

"RO" = Read Only, "RW" = Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.

With the **S-0-0160** Acceleration data scaling type, it is possible to setup the type of scaling used for acceleration IDN's.
The JVL Sercos® implementation supports the following bits of IDN **S-0-0160** Acceleration data scaling type:

Bit	Description	Bit pattern	Function	Supported
15-7	Reserved		-	-
6	Data reference	0 1	At the motor shaft At the load (S-0-0121, S-0-0122, S-0-0123)	Yes
5	Time units	-	-	No
4	Units for scaling	-	-	No
3	Scaling mode	0 1	Preferred scaling* Parameter scaling	Only preferred scaling
2-0	Scaling method	000 001 010 011-111	No scaling Linear scaling – *Not supported* Rotational scaling Reserved	Yes – partly

**The preferred rotational scaling is 0,001 rad/s^2.*

8.4.5 Probes

Probes are the capture functionality of Sercos®, which is only available with MAC400-MAC4500 motors.
Used IDN's in probes:

IDN	Description	Default	Unit	Attributes
S-0-0130 (3)	Probe value 1 positive edge	-	-	RO, C
S-0-0131 (3)	Probe value 1 negative edge	-	-	RO, C
S-0-0132 (3)	Probe value 2 positive edge	-	-	RO, C
S-0-0133 (3)	Probe value 2 negative edge	-	-	RO, C
S-0-0169 (3)	Probe control	0	-	RW
S-0-0401 (3)	Probe 1	-	-	RO, C, T
S-0-0402 (3)	Probe 2	-	-	RO, C, T
S-0-0405 (3)	Probe 1 enable	0	-	RW
S-0-0406 (3)	Probe 2 enable	0	-	RW
S-0-0409 (3)	Probe 1 positive latched	-	-	RO, C, T
S-0-0410 (3)	Probe 2 negative latched	-	-	RO, C, T
S-0-0411 (3)	Probe 1 positive latched	-	-	RO, C, T
S-0-0412 (3)	Probe 2 negative latched	-	-	RO, C, T
S-0-0426 (3)	Measuring data allocation 1 (Probe 1 signal selection)	S-0-0051	IDN	RW
S-0-0427 (3)	Measuring data allocation 2 (Probe 2 signal selection)	S-0-0051	IDN	RW
S-0-0428 (3)	IDN list of configurable measuring data	S-0-0051	IDN	RO

"RO"=Read Only, "RW"=Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.
(3) Only available in MAC400-MAC4500 with MAC00-ES4x.

The MAC00-ES4x JVL Sercos® module supports position capture when installed in a MAC400-MAC4500 motor. Here the AIN1 and AIN2 inputs act as the triggers for probe 1 and probe 2, when a programmed analog level is reached (default +5V). **Remark!** Probe 2 is only available when using the extended I/O module – MAC00-ES41.
The IDN **S-0-0169** is used for setting up the probe functionality. Only single measuring is supported, meaning for every capture the probe has to be re-enabled. Enable a specific edge of probe/capture is done by setting one of the bits 0-3 in IDN **S-0-0169** as seen in the below table.

The JVL Sercos® implementation supports the following bits/features of IDN **S-0-0169** probe control:

Bit	Description	Comments	Supported
15-9	Reserved	-	-
8	Activation	Probes are always activated when switching from PL to OL regardless of the state of this bit	No
7	Reserved	-	-
6	Probe 2 continuous measuring	Shall be zero – Only single measuring supported.	No
5	Probe 1 continuous measuring	Shall be zero – Only single measuring supported.	No
4	Reserved	-	-
3	Probe 2 negative edge	1 = Negative edge active	Yes
2	Probe 2 positive edge	1 = Positive edge active	Yes
1	Probe 1 negative edge	1 = Negative edge active	Yes
0	Probe 1 positive edge	1 = Positive edge active	Yes

Start the capture by writing the value “1” to **S-0-0405** or **S-0-0406** for respectively probe 1 and probe 2. When the capture has taking place the enable bit in **S-0-0405** / **S-0-0406** has to be toggled for new use.
IDN’s **S-0-0130** to **S-0-0133** are the captured values of the position data from IDN **S-0-0051** for the respective edges on the capture inputs AIN1 and AIN2.
In the IDN’s **S-0-0409** to **S-0-0412** are bit 0 set when a capture is made. The bit is reset when a new capture cycle is prepared by clearing **S-0-0405/S-0-0406**.
The IDN’s **S-0-0401** and **S-0-0402** are reflecting the present state of the probe inputs.
The IDN’s **S-0-0426** and **S-0-0427** are used for setup of the IDN which is captured at an event on one of the probe inputs. But they are by default setup to IDN **S-0-0051** *actual position*, so no setup of these is actually needed.
The IDN’s **S-0-0401, S-0-0402** and **S-0-0409** to **S-0-0412** are possible to map as real time bits. Please refer to *Real time bits., page 220* for how to do this.

8.4.6 Homing

8.4.6.1 Drive controlled Homing

The drive controlled homing is started by **S-0-0148** *Drive controlled homing procedure command*. Before execution of the command the **S-0-0041** *homing velocity*, **S-0-0042** *homing acceleration* and **S-0-0052** *reference position* needs to be set to desired values. If homing against a hard stop is required then the **S-0-0530** *Clamping torque* needs to be setup also.

Used IDN’s in drive controlled homing:

IDN	Description	Default	Unit	Attributes
S-0-0041	Homing velocity.	50.0000	RPM	RW
S-0-0042	Homing acceleration	523.333	Rad/s^2	RW
S-0-0052	Reference Position 1	0	-	RW
S-0-0147	Homing parameter	0	-	RW
S-0-0148	Drive controlled homing procedure command	-	-	RW
S-0-0403	Position feedback value status (Returns the home status of the drive.)	-	-	RO
S-0-0530	Clamping torque (Torque used for homing against a hard stop)	15.0	%	RW

“RO” = Read Only, ”RW” = Read and Writeable, ”C” = Map able to Cyclic data, “T” = Map able to real Time bits.

Then setup the desired homing type in **S-0-0147** *Homing parameter*. Please see available options in the table below.

Supported options of **S-0-0147** – Homing Parameter.

Bit	Description	Comments	Supported
15-11	Reserved	-	-
10	Homing with Positive Stop (Torque homing)	0 = Without hard stop. 1 = With hard stop (hard stop is used, instead of home switch. Set torque in **S-0-0530** (*Clamping torque*)	Yes
9	Homing with Limit switch	Always 0. Limit switch is not used.	No
8	Drive controlled homing with homing distance	Always 0. Homing distance is not selected	No
7	Position after drive controlled homing	Always 1. Drive is positioned at the reference position (**S-0-0052**)	No
6	Evaluation of position feedback marker pulse (index pulse)	0 = The index pulse is evaluated. 1 = The index pulse is not evaluated.	Yes (1)
5	Evaluation of home switch	Always 0. Home switch is evaluated, but if bit 10 is set then it is not evaluated.	Partly
4	Interpretation in the drive	Always 0. Home switch and homing enable are used	No
3	Homing feedback	Always 0. Motor feedback is used for homing and not the external feedback	No
2	Home switch	Always 1. Home switch is connected to drive	No
1	Position feedback marker pulse (index pulse)	0 = First index pulse after the positive edge of the home switch. 1 = First index pulse after the negative edge of the home switch.	Yes
0	Homing direction	0 = Positive; motor shaft turns clockwise 1 = Negative; motor shaft turns counter-clock wise	Yes

(1) If using MAC motors with absolute multiturn encoder and MIS/MILxxx motors which do not have an index pulse, then this bit is don't care.

8.4.7 Real time bits.

Real time bits are supported as producer (from motor to master) in the MAC00-ES4x module when mounted in a MAC400-MAC4500 motor. The MIS/MIL motors and the other MAC motors do not support real time bits.

Used IDN's with Real Time Bits:

IDN	Description	Default	Unit	Attributes
S-0-0398 (3)	IDN list of configurable real-time bits as producer	S-0-0401, S-0-0402, S-0-0409 - S-0-0412	IDN	RO
S-0-0399 (3)	IDN list of configurable real-time bits as consumer	0-0-0000	IDN	RO
S-0-1050.0.20(3)	IDN allocation of real time bit	0	-	RW
S-0-1050.0.21(3)	Bit allocation of real time bit	0	-	RW

"RO"=Read Only, "RW"=Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.
(3) Only available in MAC400-MAC4500 with MAC00-ES4x.

8.4 FSP Drive profile

The IDN **S-0-0398** contains a list of IDN's which are map able to real time bits as producer, which are at present the IDN's **S-0-0401, S-0-0402** and **S-0-0409** to **S-0-0412.** Setup of the real time bits is done by writing the desired IDN(s) to the list in IDN **S-0-1050.0.20** and the bit number in the corresponding place in list **S-0-1050.0.21**.
In other words:

- For real-time bit 1 the master shall configure list element 0 of **S-0-1050.x.20** and **S-0-1050.x.21**.
- For real-time bit 2 the master shall configure list element 1 of **S-0-1050.x.20** and **S-0-1050.x.21**.

The real time bits 1 and 2 are transferred from the motor to the master in bits 6 and 7 of the Sercos Connection Control word.

8.4.8 IDN Backup.

When the motor is starting after power up the default values for operation data IDN's are loaded from flash.
The IDN **S-0-0192** *IDN-list of all backup operation data* contains a list of all the IDN's which are loaded with default values at power up. When issuing one of the commands **S-0-0262** *Load defaults*, **S-0-0263** *Load working memory* or **S-0-0264** *Backup working memory* it is the same IDN's which are affected. And also the **S-0-0531** *Checksum of backup operation data* is done on the IDN's listed in **S-0-0192.**

Used IDN's with backup:

IDN	Description	Default	Units	Attributes
S-0-0192	IDN-list of all backup operation data	-	IDN	RO
S-0-0262	Load factory defaults procedure command	-	-	RW
S-0-0263	Load working memory procedure command	-	-	RW
S-0-0264	Backup working memory procedure command	-	-	RW
S-0-0531	Checksum of backup operation data	-	-	RO

"RO"=Read Only, "RW"=Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.

Changing of startup values.
1. Write the desired values in the relevant IDN's.
2. Issue procedure command **S-0-0264**.

Then these values will be restored after powerup. Values will also be restored if procedure command **S-0-0263** is issued.
Remark ! Only the IDN's present in the **S-0-0192** list will be backed up.

8.4.9 Modulo.

In MIS motors the Modulo format has to be setup directly in the motor. If used then choose the "Advanced" tab in Mactalk select Turn table mode "Shortest path", set the min. value to zero, and the max value to the needed counts (the MIS motors has 409600 counts per rotation). Then press the "Save in motor" button for activating the setup.
If the Modulo format is selected in IDN **S-0-0076** *position data scaling factor*, the Modulo value defines the range that the drive and control unit shall implement. The Modulo format is not default selected.

Used IDN's with modulo function:

IDN	Description	Default	Units	Attributes
S-0-0076	Position data scaling type	2	-	RW
S-0-0103	Modulo value	147456000	-	RW

"RO"=Read Only, "RW"=Read and Writeable, "C" = Map able to Cyclic data, "T" = Map able to real Time bits.

8.4.10 Interpolation.

If the **motion** cycle time is larger than the **sercos** cycle time then the drive performs an internal interpolation of the cyclic position values if running in position mode. Please refer to *Supported cycle times (Pack Profile / FSP Drive and FSP Sync), page 211* for possible cycle times.

8.4.11 Absolute position.

If an absolute encoder is present in the MIS motor the IDN's in the below table is active. Even though the MAC motors do support absolute encoders they have to be referenced at every power-up, or the homing done bit in the motor must be set manually. Please refer to the MAC motor manual for this.
Only the motor internal encoder is supported, so the IDN **S-0-0448** is only implemented to be compatible with the Sercos function group **FG Set absolute position** it does not contain any functionality. With the **S-0-0447** command the present actual position is taken to be the zero position, and the **S-0-0403** Position feedback value status is set to referenced. With the command **S-0-0191** the referenced status in **S-0-0403** is cancelled. If a homing procedure is completed the IDN **S-0-0403** is also set to active. For MIS motors the state of the IDN **S-0-0403** is kept in non-volatile memory so it is not deleted at power interruptions.

IDN	Description	Default	Units	Attributes
S-0-0175	Displacement feedback value encoder 1	-	-	RO
S-0-0191	Cancel reference point procedure command	-	-	RW
S-0-0403	Position feedback value status	-	-	RO
S-0-0447	Set absolute positino procedure command	-	-	RW
S-0-0448	Set absolute position control word	-	-	RW

8.5 FSP IO / JVL profile

The JVL Profile based on the FSP IO profile is supported by all JVL motors with Sercos®. But for the MAC400 – MAC4500 motors and the MIS/MILxxxES motors it is not factory default. Please refer to *Enabling FSP Drive / Pack Profile., page 195* for enabling the JVL profile.

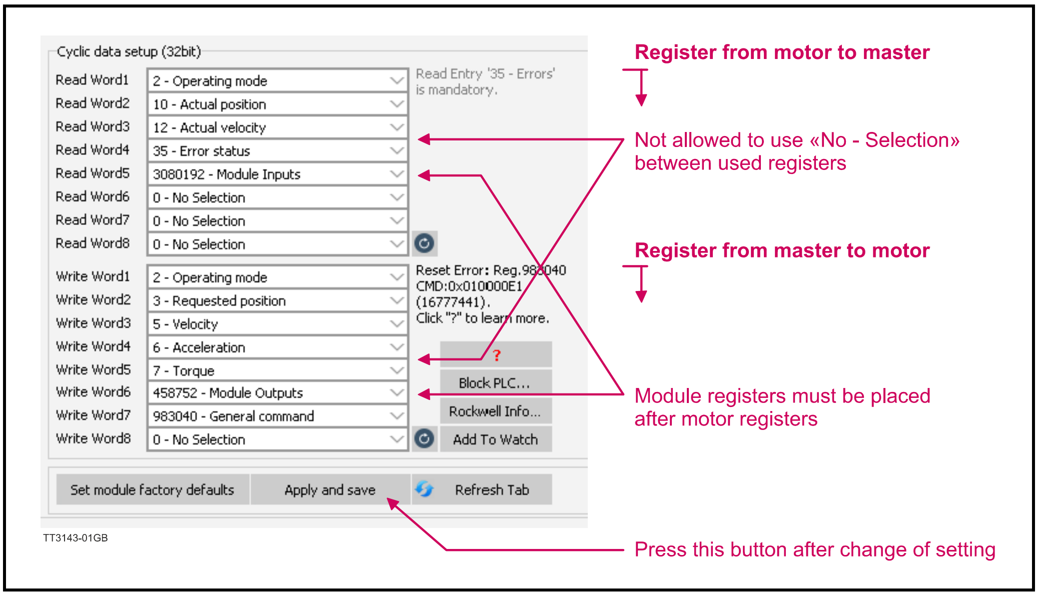
8.5.1 Setting up the cyclic data (FSP IO / JVL Profile)

When using the FSP IO / JVL Profile the cyclic data (the data which is transferred cyclically between master and slave) are setup in the MacTalk program in the Sercos tab. In the below figure is shown an example of a cyclic setup where both motor controller registers and Sercos® module registers are accessed.
Cyclic data is used when data is needed fast and frequent. That is fast dynamic changing data such as position, velocity, torque etc.
It is mandatory to have the error/status register (register 35) as one of the slave to master registers. If not placed by the user then the Sercos module will overrule the configuration and place register 35 anyway.

When module registers (register numbers **above** 65535) are chosen, they have to be placed **after** the motor registers (register numbers **below** 65535) in the list of cyclic registers. If module registers is placed in cyclic R/W, then the register number can be calculated as follows:

Register number = 65536 x sub index.
Example: module command (sub-index 15) = 65536 x 15 = register **983040**

The JVL profile cyclic data consists of up to 8 I/O words that are freely configurable. This means that 8 input registers can be selected and another 8 registers for output purposes. **NB!** If an index is set to zero (No selection), then the following indexes is discarded. Thereby computing resources in the drive are released, which makes much faster cycle times possibly. Please see next paragraph for possible cycle times.
On the Sercos tab in MacTalk these I/O's are configured, as in the example below.



8.5 FSP IO / JVL profile

After change of settings the button "Apply and save" are pressed. Then the settings are saved to flash and the Sercos module does a reboot for the changes to take effect.
Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, *section 13.2* and Motor registers MAC400 - 4500, *section 13.3* and Motor registers MIS/MILxxx, *section 13.4* Sercos Module registers *section 10.1*

Please notice: The motor registers 35, 36, and 211 should NOT be inserted in the cyclic write list, as this may give unpredictable results. For clear of errors please refer to *Reset motor errors (FSP IO / JVL Profile), page 230*. For a list of commands for the module command register please refer to *Register 15 - Command register, page 260*.

Please notice: Even though all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

8.5.2 Minimum cycle time (FSP IO / JVL Profile)

The minimum cycle time is the minimum amount of time between each cyclic request on the Ethernet (Sercos® cycle time).

If operating with values lower than those listed, data loss **will** occur.

No. of motor registers transmitted in each direction	Motor series MAC050 to MAC141 *	Motor series MAC400+ *	Motor series MIS/MIL *
1/1	4ms	500 µs	1ms
2/2	8ms	500 µs	1ms
3/3	12ms	500 µs	1ms
4/4	16ms	500 µs	1ms
5/5	20ms	500 µs	1ms
6/6	24ms	500 µs	1ms
7/7	28ms	500 µs	1ms
8/8	32ms	500 µs	1ms

** The minimum cycle times, is only valid if not sending any non-cyclic requests while in any operating mode. MODULE registers can be appended as the last registers in the list, at no extra timing cost. Motor register 35 shall be present in the cyclic read list, as it is also used internally.*

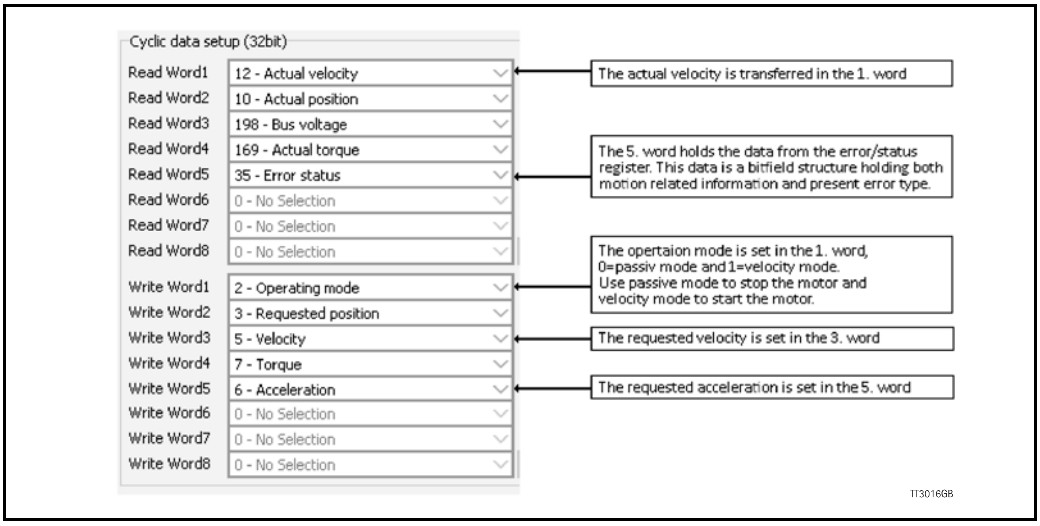
8.6 Examples

8.6.1 Running Velocity control (JVL Profile)

To use the JVL motor in velocity mode the following registers are basically of interest.

1. "Mode" - Mode register register 2
2. "V_SOLL" - Velocity register 5
3. "A_SOLL" - Acceleration register 6
4. "Error/Status" - Error and status register 35

So, to control these registers the cyclic data needs to configured.
From MacTalk the setup is configured as this.



With the settings illustrated above we initiate the velocity mode by writing 0x1 to the first word-value, this is velocity mode.

From the Master the registers is accessed using the cyclic data and accessing the registers R/W on words 1-8.

Since different PLC's have different methods of implementation the basic steps is described in the following. (Constant values valid for MAC800, for other motors, please consult the motor manual)

1. Set the needed velocity. V_SOLL = V x 2.77 [rpm]
   Ex. We need the motor to run with a constant speed of 1200 RPM. So, V_SOLL = 1200/2,77 = 433 cnt/smp
2. Set the needed acceleration. A_SOLL = A x 271 [RPM/s^2]
   Ex. We need the motor to accelerate with 100000 RPM/s^2 so, A_SOLL = 100000/271 = 369 cnt/smp^2.
3. Now set the motor into velocity mode and thereby activate the motor.
   Ex. The motor needs to be activated by setting it into velocity mode, so we need to set the mode register to the value 1. Mode = 1 which is velocity mode, now the motor will use the acceleration and the velocity just configured.

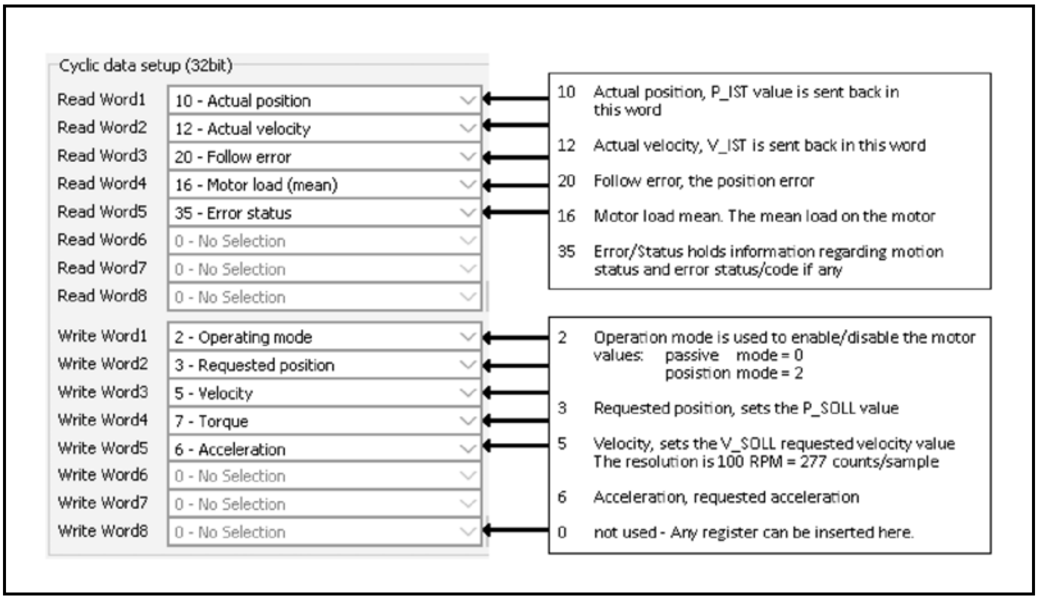
Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 and *Motor registers MAC400 - 4500, page 301* and *Motor registers MISxxx, page 320*

8.6.2 Running Position control (JVL profile)

Running the motor in position control requires that the mode register is set for position control. The following registers is of particular interest when position mode is used.

1. "Actual position" -P_IST, register 10
2. "Actual velocity" -V_IST, register 12
3. "Follow error" - The actual position error, register 20
4. "Motor load mean" - average motor load, register 16
5. "Error/Status" -register 35
6. "Requested position" -P_SOLL, register 3
7. "Requested velocity" -V_SOLL, register 5
8. "Requested acceleration" -A_SOLL, register 6

In this mode the position is controlled by applying a requested position to the "P_SOLL" -register and the actual position is monitored in the "P_IST" register. The V_SOLL and A_SOLL registers sets the velocity and acceleration used when positioning occurs.



8.6.3 General considerations

The register 35 in the motor holds information on the actual error/status. So it is crucial that this register is configured in the cyclic data and thereby obtained and monitored in the Master. In case of an error situation the motor will stop and the cause will be present in the register 35 and hence in the I/O -data.
This register also holds information on the motion status such as:

- In position, bit 4
- Accelerating, bit 5
- Decelerating, bit 6

Please find a complete list of register descriptions in the appendix.
Motor registers MAC050 - 141, page 292 and *Motor registers MAC400 - 4500, page 301*

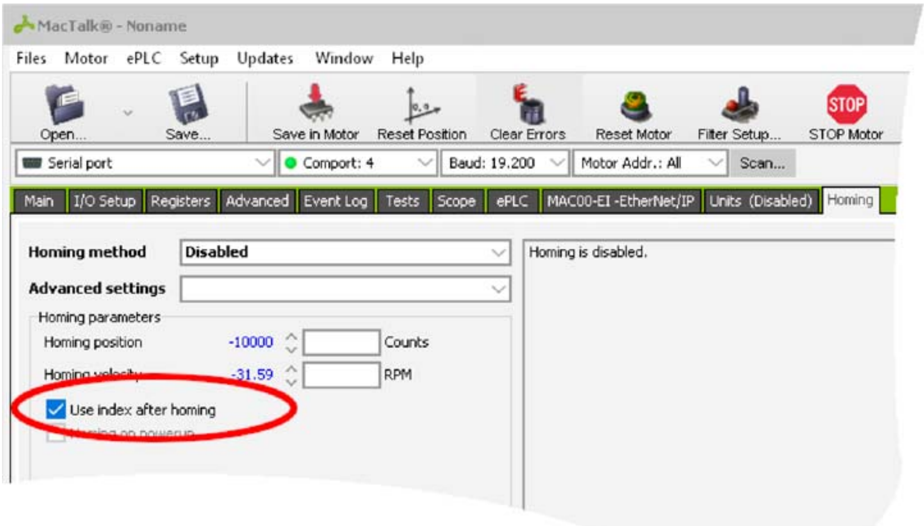
The JVL motor is basically put into a working mode and into a passive mode where the motor axle is de-energized, by setting register 2 into either 0 = "passive mode" or into one of the supported modes.

Example.
1= "Velocity mode" / 2= "Position mode" / etc.

So in order to Stop or Start the motor this register can be supported in the I/O data or by sending an Service channel message.

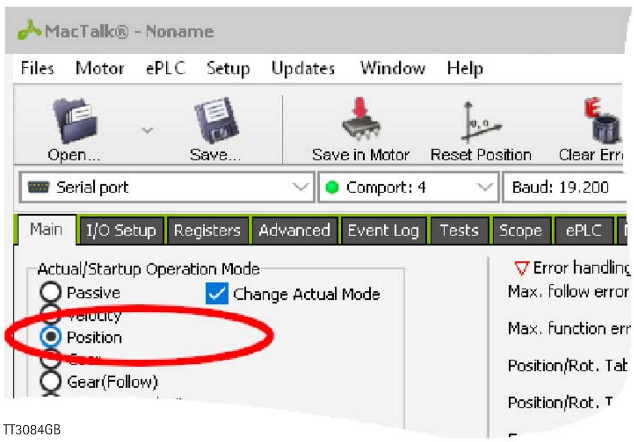
8.6.4 Homing using only cyclic I/O (JVL profile).

When doing a homing (Zero search), with only cyclic I/O, some preconditions have to be met:
Zero search position, zero search velocity and zero search torque (torque only for MAC motors) has to be set in MacTalk in the "Main" tab, and saved in flash in the motor once and for all.



TT3083GB

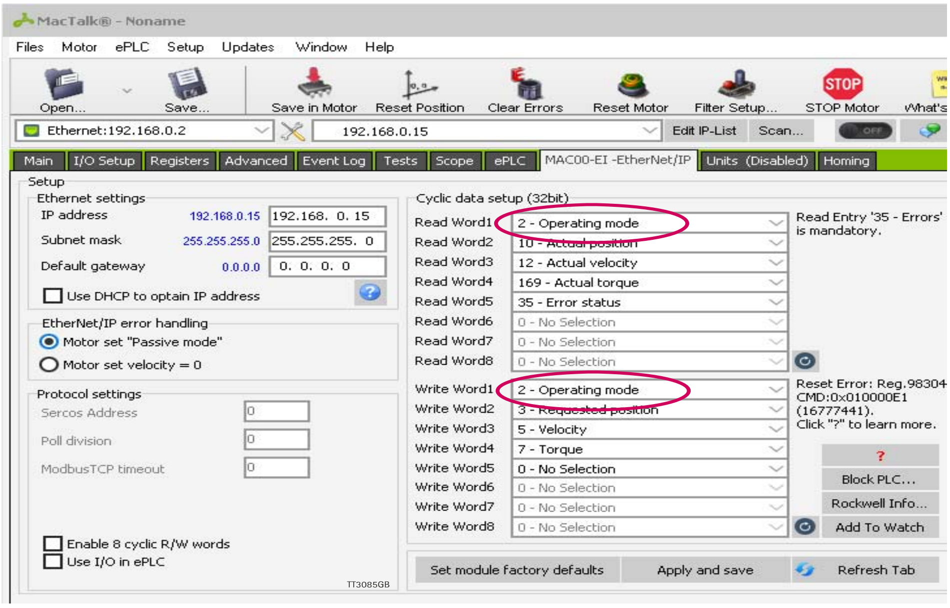
Startup mode should be set to position, for the motor to stay in position after the homing sequence. And this setting should also be saved in flash.



TT3084GB

8.6 Examples

Register 2 (Operating mode) has to be present in BOTH the cyclic read words and cyclic write words.



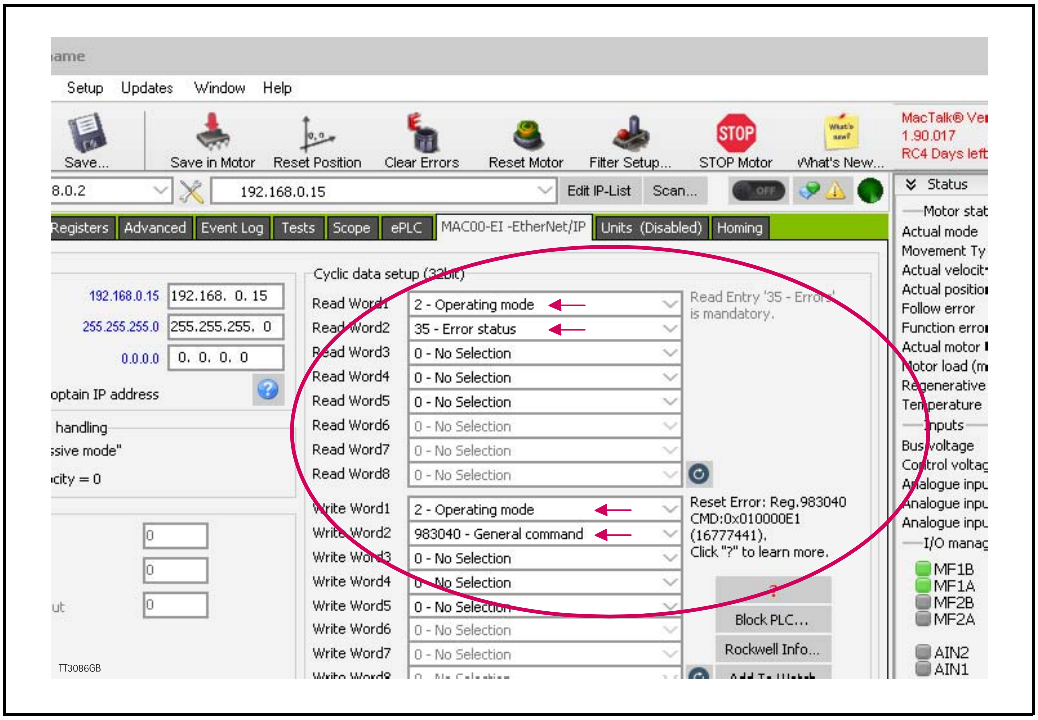
Procedure in the PLC:

- Treat the transmitted Register 2 as "Requested_Mode" and the received register 2 as "Actual_Mode".
- When homing is wanted, set the "Requested_Mode" to one of the values 12, 13 or 14, 25 or 26 depending of the requested homing mode (12 = Torque based zero search mode. 13 = Forward/only zero search mode. 14 = Forward+backward zero search mode (only MAC motors). 25 = Enc. index (only MAC400+). 26 = Enc. quick index (only MAC400+).). For a comprehensive description of the homing modes, refer to the general MAC motor manual - LB0047-xxGB or the MIS/MIL motor manual - LB0058-xxGB. Can be found at *www.jvl.dk*
- Observe that the "Actual_Mode" is changing to the homing mode. Now the module is blocking cyclic writes TO the motor. Cyclic reads is still active.
- Wait for register 35 "Error status" bit 4 to be active =IN_POSITION. (Indicates that homing is finished).
- Then change "Requested_Mode" to whatever needed. The blocking of cyclic writes to the motor is then released by the module.

8.6.5 Relative positioning (JVL profile).

There are a number of ways to do relative positioning, but the one explained here is very simple, and can be used with a constant distance, or exchangeable distance, to move every time it is requested.

Preconditions:
Place the module command register (register 983040 in MacTalk) in the cyclic write list. The cyclic setup, could for example look like this:



Procedure in the PLC:

1. Set up register P7 in motor to requested relative offset.
2. Make sure one net cycle has passed, so P7 resides in the motor.
3. Issue command 0x010000F1 (0x01000071 if the device is a MIS/MILxxx) in module command register (register 983040 in MacTalk).
4. Make sure one net cycle has passed, so command is interpreted by the motor.
5. Set module command register to zero. This will prepare the Ethernet module for new commands.
6. If needed then monitor register 35 (Error status): When bit 4 is set (in position), then the move is finished.
7. When a new relative move is requested, go to step 3.

You may also have the P7 register in the cyclic write list, thereby enabling easy change of the relative distance to move.

8.6.6 Save parameters to flash (FSP IO / JVL Profile)

Saving of the Sercos® module or the motor controller registers to flash (non-volatile memory) only requires a simple non-cyclic command to the Sercos® module command register which is accessible via IDN **P-0-3015**.

Please Refer to *IDN Backup., page 221* for saving FSPdrive IDN's in flash.

Save Sercos® module parameters to flash:
- Write the value 16#0000 00010 (= 16 dec.) to IDN **P-0-3015**.

Save motor parameters to flash:
- Write the value 16#0000 00110 (= 272 dec.) to IDN **P-0-3015**.

8.6.7 Reset motor errors (FSP IO / JVL Profile)

If the motor gets an error the error can be cleared by issuing a command to the **IDN P-0-3015** (Module command register) via the service channel (SVC). Or issue the command to the module command register if the module command register is placed in the cyclic write list (refer to *Setting up the cyclic data (FSP IO / JVL Profile), page 223* for changing the cyclic setup).

Reset motor error command to module command register:

Motortype	Command (hex)	Command (dec)
MIS/MIL	16#0100 0061	16777313
MAC	16#0100 00E1	16777441

If done via the service channel then the command is written to **P-0-3015** and the error is cleared.
If done with cyclic data then follow the below procedure.

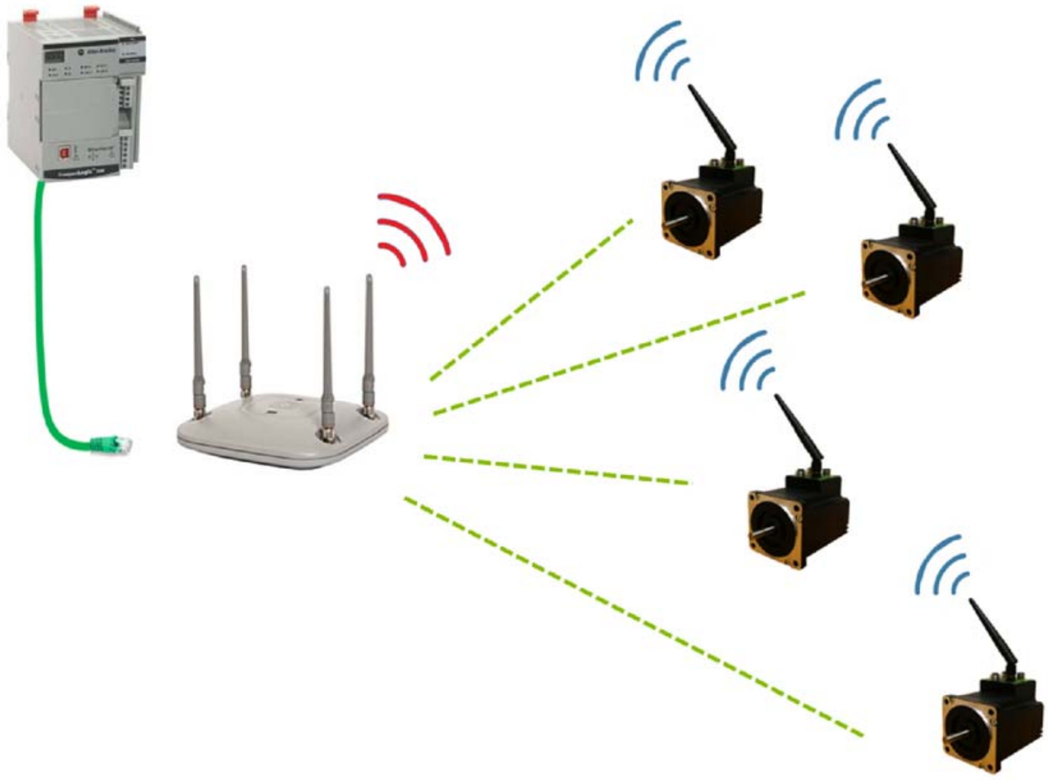
1. Send the command value from above table to the mapped variable in the PLC project which points to the Sercos® module command register.

2. Make sure one net cycle has passed, so command is interpreted by the motor.

3. Send the command value

4. Write 0 (zero) to the mapped variable in the PLC project which points to the Sercos® module command register. This will prepare the Sercos® module for new commands.

9 Industrial WIFI



TT3148-01GB

9.1 Setting up wireless connection.

9.1.1 Setting up wireless connection.

Motors equipped with an Ethernet system is equipped with 2 Ethernet ports and an internal Ethernet switch.
When a motor is equipped with the wireless system, one of the Ethernet ports are used for the wireless connection.

The second Ethernet port is available but only in as wireless configuration.

The following equipment is recommended to get a reliable wireless network up and running.

1. JVL motor equipped with the wireless option and running one of the following Ethernet protocols:
   - EthernetIP
   - ProfiNet
   - ModbusTCP

2. A good wireless access point designed for industrial use with good diagnostic features. In house tests has been conducted with Siemens Scalance W76x, other vendors offers the same features.

3. Industrial grade switch systems to manage the wired connections from the PLC and Access point(s).

4. MacTalk V1.90.019 or newer.

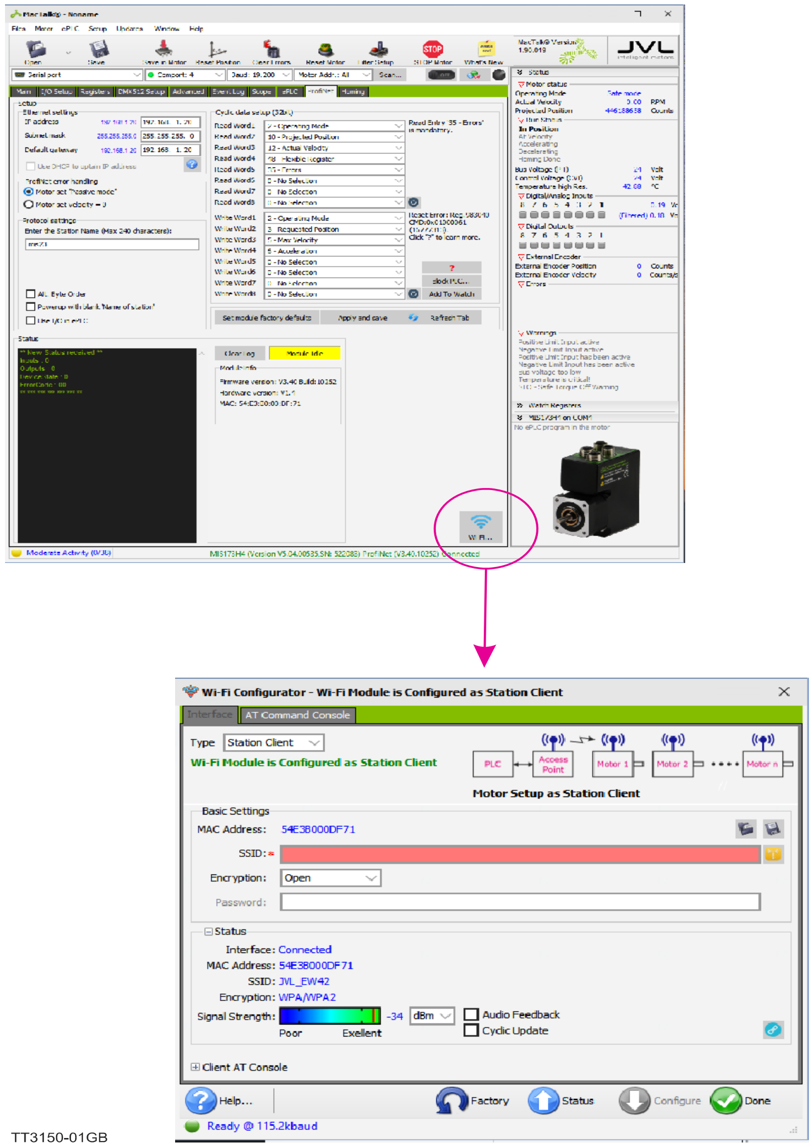
To get a reliable wireless network established requires some planning and a good knowledge of the environment the system is supposed to be working in.
However wireless networks are by definition not as reliable as the wired solution and loss of packages can occur at a random interval so the system must be designed to accommodate that.
Careful planning is also vital to get a good and reliable connection to the wireless nodes.

9.1 Setting up wireless connection.

The JVL wireless network can run in 2 different modes depending on which topology is required.
From MacTalk the configuration is done from the dialog that is accessed by pressing the "WIFI" -button.

TT3150-01GB

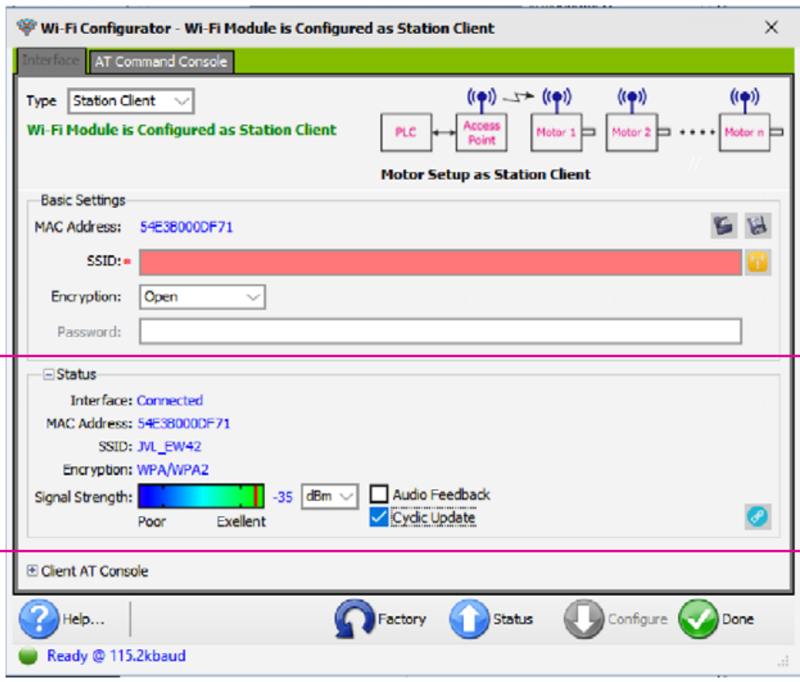
Please note that configuration of the wireless network in the motor is only possible through a serial connection (not via Ethernet). When the wireless connection is running the motor can be accessed over the wireless connection, but the wireless settings are not available.



TT3150-01GB

9.1 Setting up wireless connection.

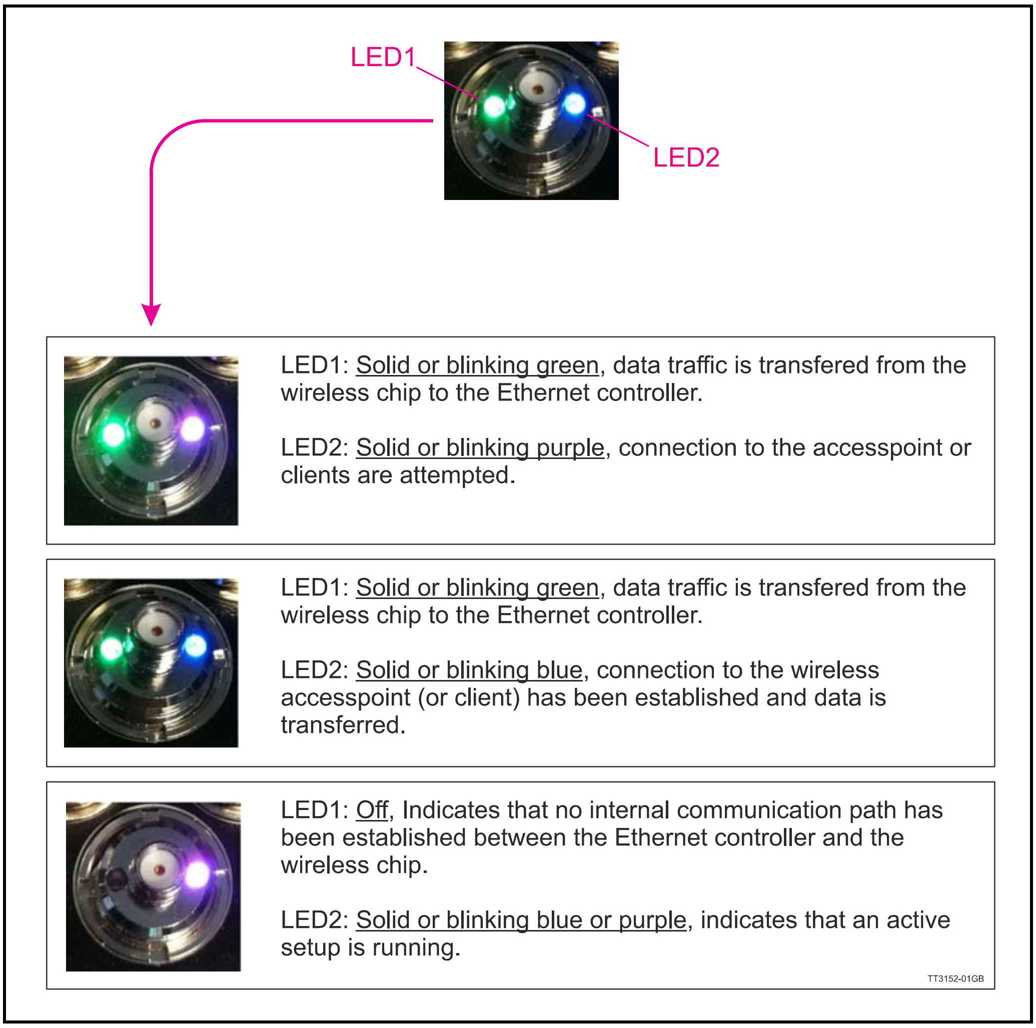
After the connection to the module has been established, the current setup is read and displayed in the "Status" -section.



TT3151-01GB

9.2 Status indicators at the WIFI module

On the WIFI antenna socket 2 LED's indicate the current status of the wireless connection.
Please note that for convenience the antenna has been removed in the following illustration, do not operate the device without the antenna connected.

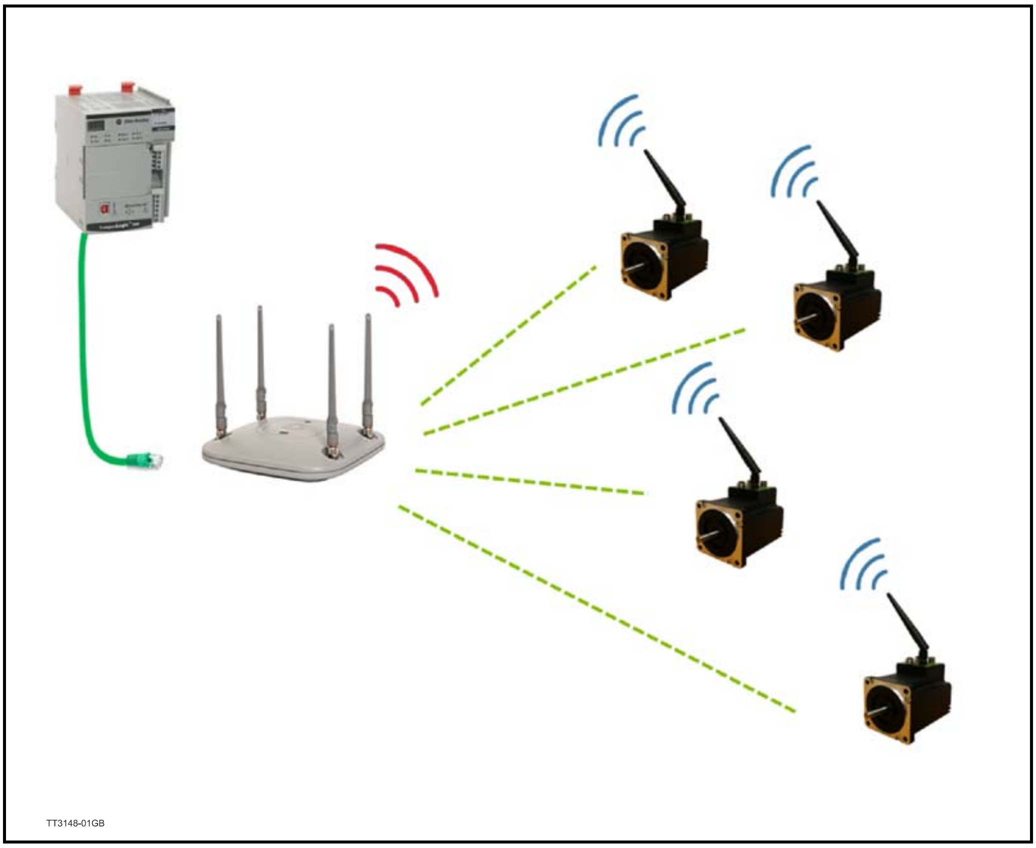


LED overview

LED1	LED2	Description
Green	Green	No wireless setup in the motor, make a new setup from MacTalk.
Green	Purple	Connection attempt to either access point (Configuration #1) or Client (Configuration #2)
Green	Blue	Connection is established, blinking indicates traffic
OFF	Purple	Internal communication is missing between ethernet controller and wireless controller, connection is attempted on the wireless interface.
Off	Blue	Internal communication is missing between ethernet controller and wireless controller, connection is established on the wireless interface.
Off	Red	Internal error in the wireless controller, try setting factory default and make a new setup.

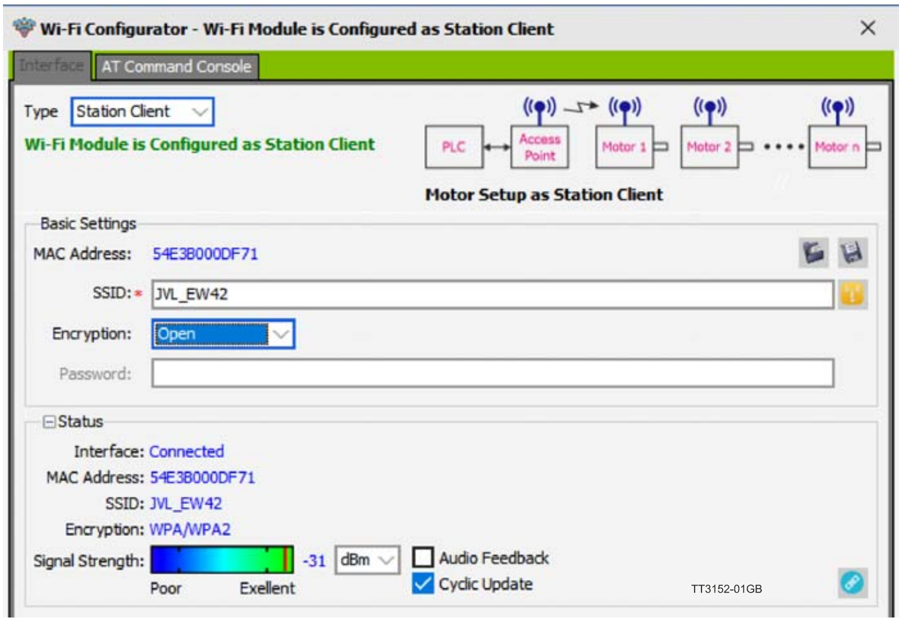
9.3 Configuration #1, Station Client

When the motor is configured as “Station Client”, the secondary Ethernet port on the motor is not available. In other words it is **NOT** possible to connect other equipment to this port and reach it through the wireless connection.
The Access point connected to the PLC works as an access point and the motors connects to this access point and works as clients.



9.3 Configuration #1, Station Client

9.3.1 MacTalk “Station Client” configuration



9.3.2 Basic settings

In this section data for the setup can be entered.

MAC Address: Current MAC addr. of the Ethernet processor in the motor. Notice that the MAC address is a read only register.

SSID: Enter the SSID of the access point.

Encryption: Encryption method used in the access point. WPA/WPA2 can be selected.

Password: Enter the WPA/WPA2 password of the access point. Note, by pressing the “Eye”-button, the entered password is readable.

For more information on how to setup the access point, find the section *Setting up the access point for configuration #1, page 240*

9.3.3 Status

This section holds the actual setup and status of the wireless interface.

Interface: Shows the connection state to an access point [Connected / Disconnected].

MAC Address: Current MAC address used for the wireless interface. This address should be the same as the MAC address printed on the label.

SSID: Current SSID configuration. This is the SSID the motor will try to connect to.

Encryption: Current encryption setting [WPA/WPA2 or OPEN]. Observe that the password is protected from viewing.

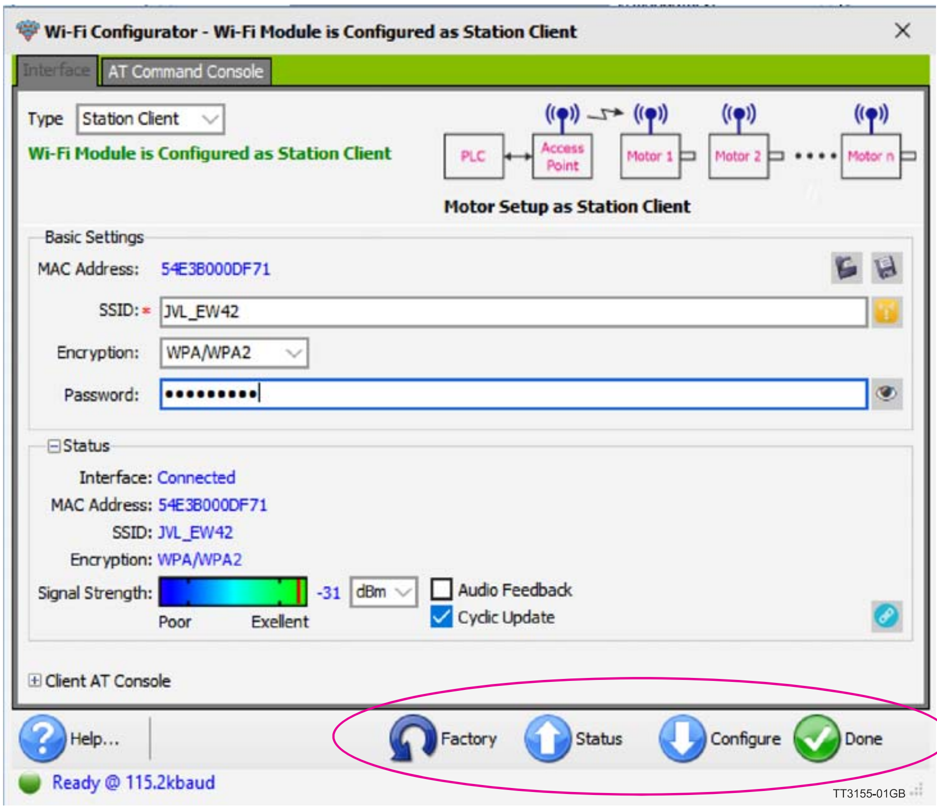
Signal strength: When the motor is connected, the signal strength can be monitored. The units can be presented in either [dBm or %]. In good industrial access points the signal strength of the clients can be monitored as well in the managing software for the access point.

9.3 Configuration #1, Station Client

When the settings has been configured, the setup is transferred to the motor by pressing the "Configure" -button found in the bottom of the dialogue.

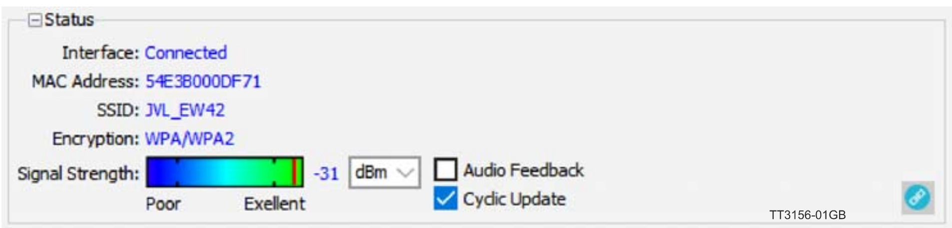
TT3154-01GB

Note: The settings are saved permanent in the motor after they are transferred.



TT3155-01GB

When the transferring has completed, the motor will connect to the access point with the SSID entered, using the encryption and credentials configured.
When the motor has established a wireless connection the status changes from "Disconnected" to "Connected".
The signal strength is also changed from "---" to a value either in dBm or %.



TT3156-01GB

9.3 Configuration #1, Station Client

In the screen-shot above the motor is connected to an access point with the SSID "JVL_EW42", there is no encryption configured.
The signal strength is reported by the motor to be -20dBm (very good).
For a more human readable value the unit can be switched to %.

For the best performance and reliability the signal strength should always be in the "Excellent" -area of the status bar.

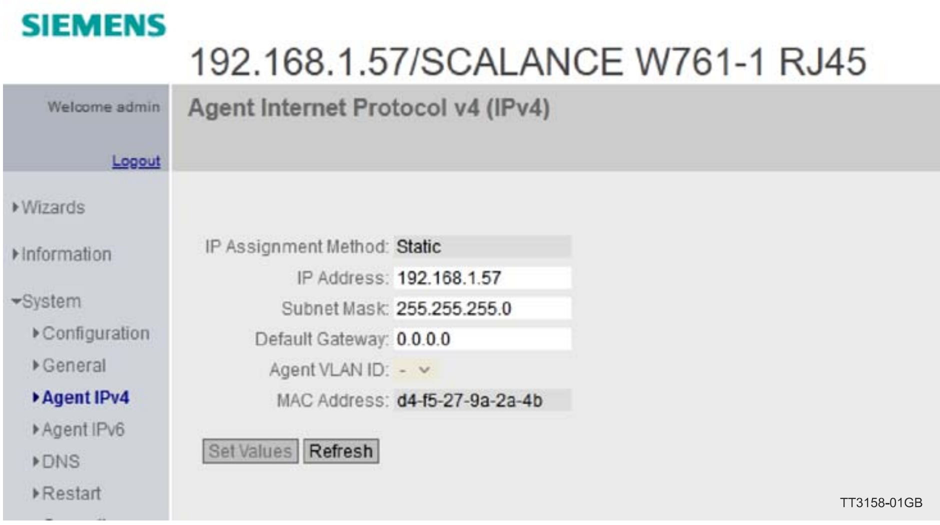
9.3 Configuration #1, Station Client

9.3.4 Setting up the access point for configuration #1

In this section we will cover the basic settings of the access point when a motor is configured as "Station Client".
The access point is the Ethernet wired to wireless gateway on which the PLC is connected.
The access point used is a Siemens Scalance W76x, but the same settings are available in other good industrial access points.

9.3.5 IP address of the access point

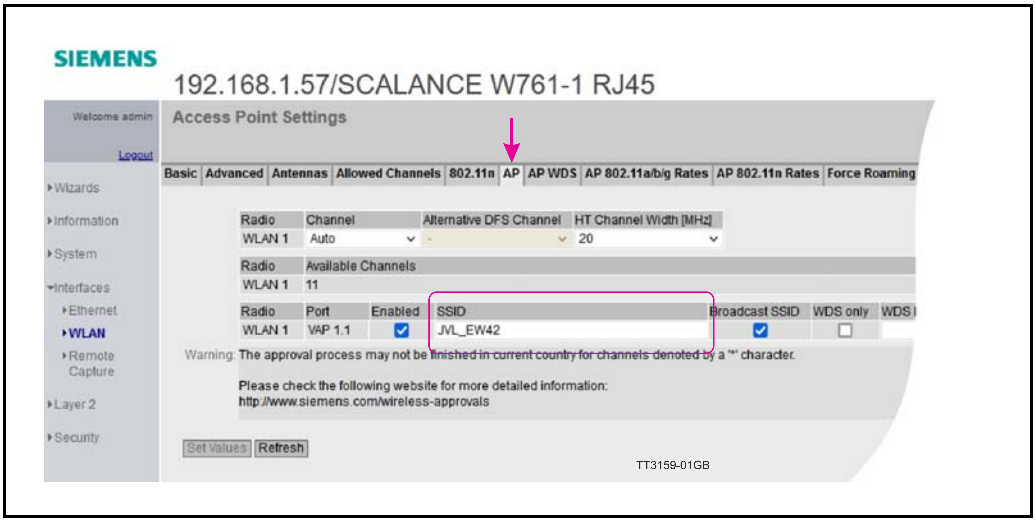
The IP address settings of the wired part must match the PLC settings. In this example all the equipment is running on the subnet 192.168.1.xxx. The access point is configured for the same range with the IP address 192.168.1.57.



The way IP addresses are configured can be manufacturer specific and is covered in the manual for the device.

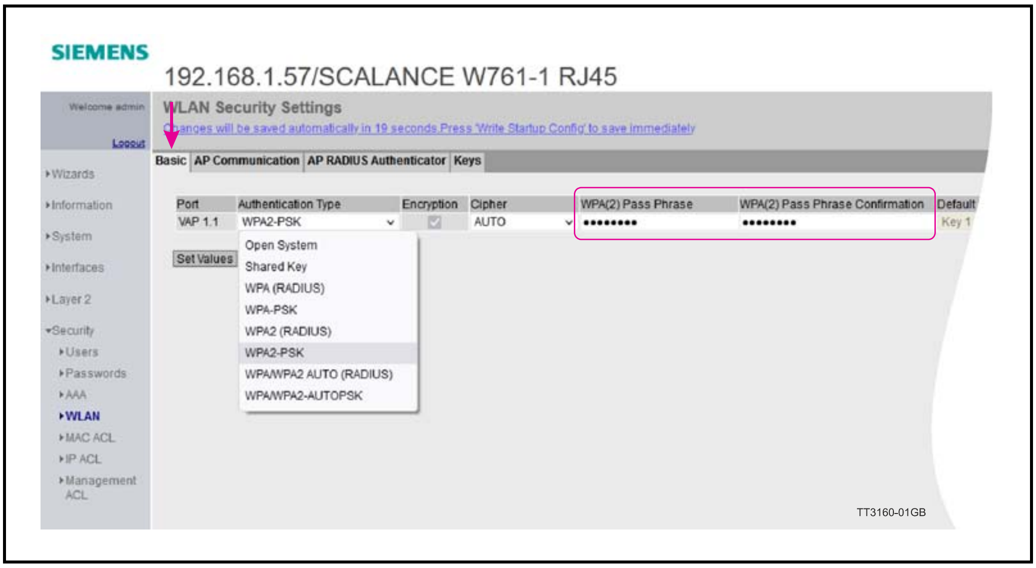
9.3.6 SSID (ServiceSetIdentifier)

The SSID is basaically the identification on the wireless network. The SSID can be either visible or invisible for a scanner. When wireless networks are scanned it is the SSID of the networks that appears in the list.
In this example we use "JVL_EW42" as SSID, in the Siemens configuration this is setup in the following dialog:



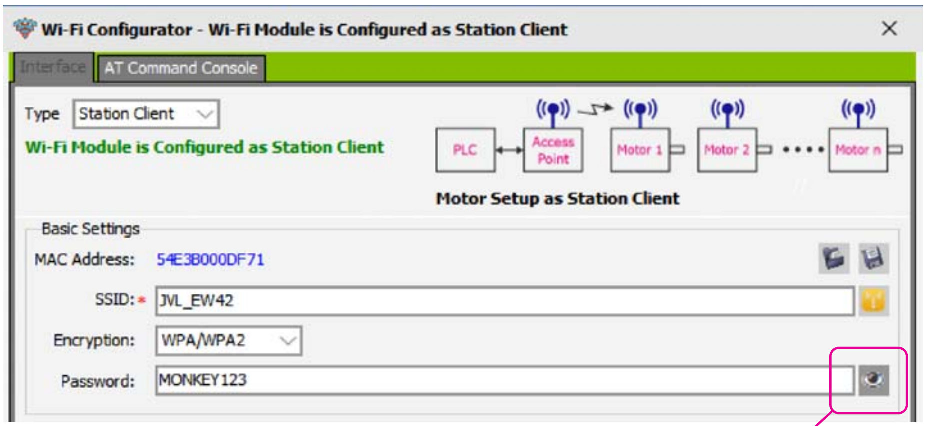
9.3.7 Encryption

The encrytion scheme supported in the motor is WPA/WPA2.
The password "MONKEY123" is chosen and configured in the access point.



9.3 Configuration #1, Station Client

The same password credentials must be configured in the motor. Press the "Configure" button to transfer the settings to the motor.



TT3161-01GB

Hint !
By pressing and holding down the "Eye" -button, the Password is visible.

When everything has been configured, the motor will try establishing a connection to the access point.
when the connection has been established, the LED's on the motor will show the following indication:

TT3162-01GB

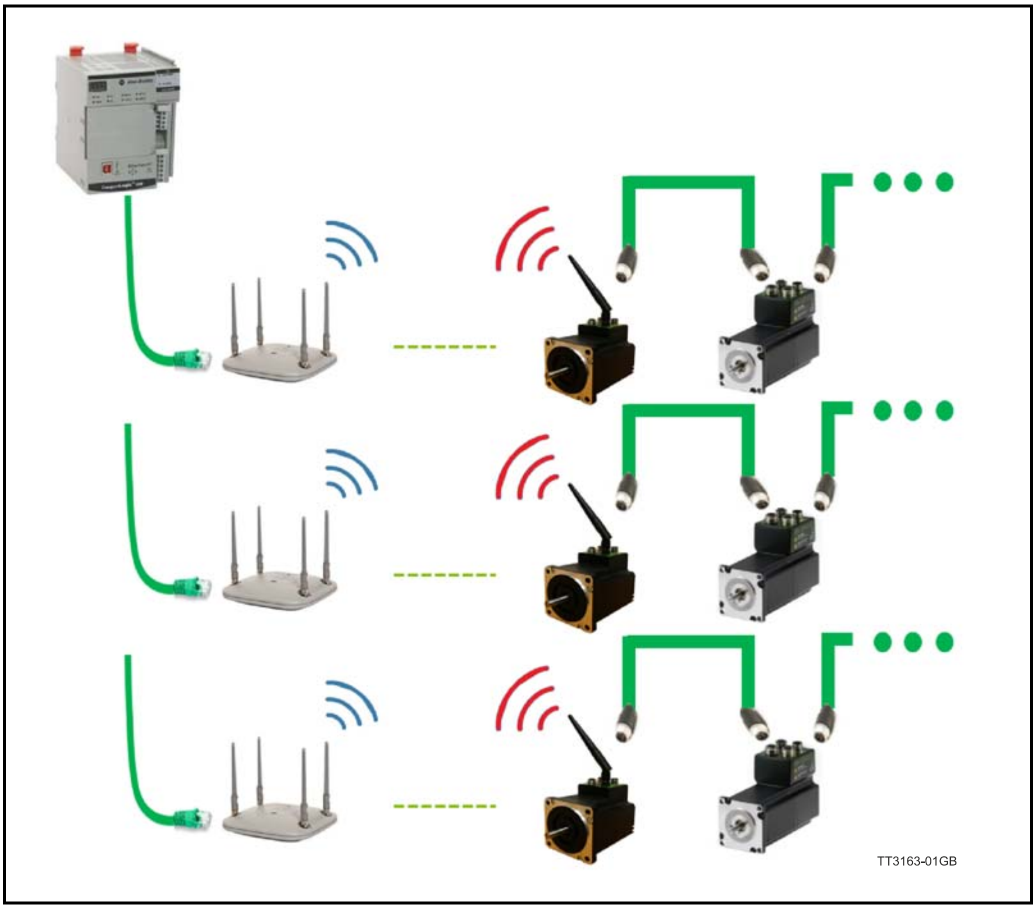
Green and Blue indicates that the internal bridging between the wireless chip and the Ethernet controller has been established.

The blue LED indicates that the wireless connection is established. When data is transmitted the LED's are blinking indicating the traffic.

NOTE : The above image shows a motor without the antenna connected, this is purely for demonstration reasons, do not operate the motor without the antenna connected.

9.4 Configuration #2, Access point

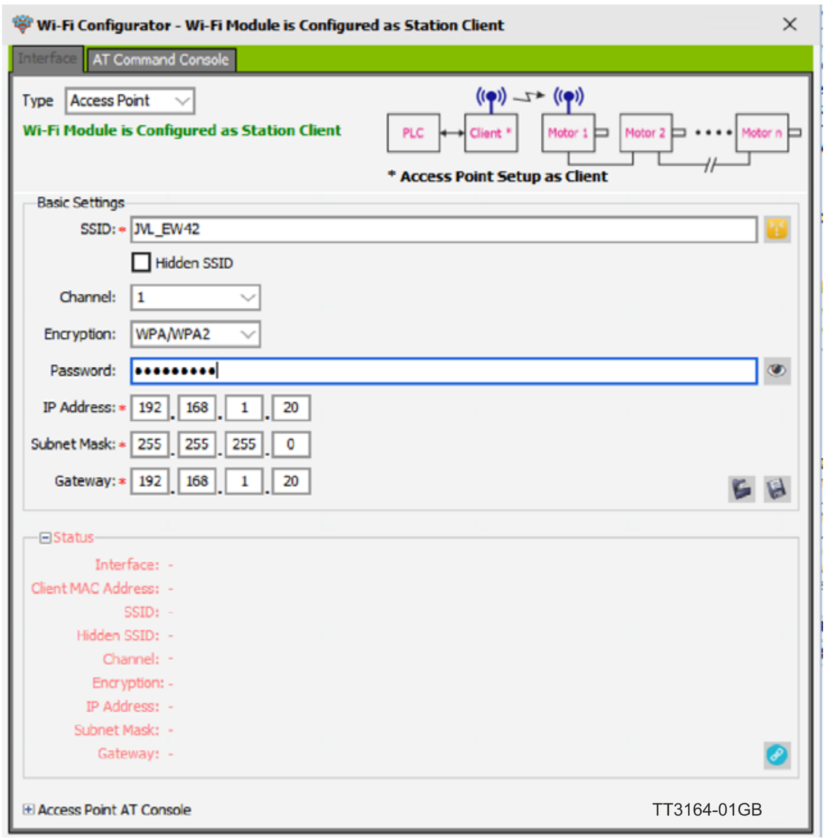
Each wireless motor is configured as an access point and the PLC is connected to a wireless client. This method leaves the secondary Ethernet port in the motor, open for connecting another wired node and the internal switch in the motor will route the packages through the wireless network to and from the PLC to the nodes connected to the port. The drawback of this setup is that it gets very complicated requiring a lot of bandwidth if many wireless "strings" are needed. The PLC and each Client should be connected to a industrial grade managed switch.



9.4 Configuration #2, Access point

9.4.1 Mactalk setup "Access point"

When the motor acts as an access point there are more settings that must be considered. An additional set of IP addresses must be setup.



9.4.2 Basic settings

In this section data for the setup can be entered.

SSID: Enter the SSID for the Access point. This is the ID the motor will use for the Clients to connect to.

Hidden SSID: Check this setting if the SSID should not be visible in a network scan.

Channel: Select the channel for the Access point configuration.

Encryption: Select the encryption that must be used. Note, by pressing the "Eye"-button, the entered password is shown.

Password: Enter the WPA/WPA2 password of the access point. Note, by pressing the "Eye"-button, the entered password is shown.

IP Address: Enter the IP address that the Wireless adapter in the motor should have. Note the clients that connects to this motor must be in the same subnet.

Subnet Mask: Subnet mask settings of the wireless network.

Gateway: Gateway settings the wireless adapter.

For more information on how to setup the access point, find the section *Setting up the accesspoint for the Configuration #2, page 247*

9.4 Configuration #2, Access point

9.4.3 Status

This section holds the actual setup and status of the wireless interface.

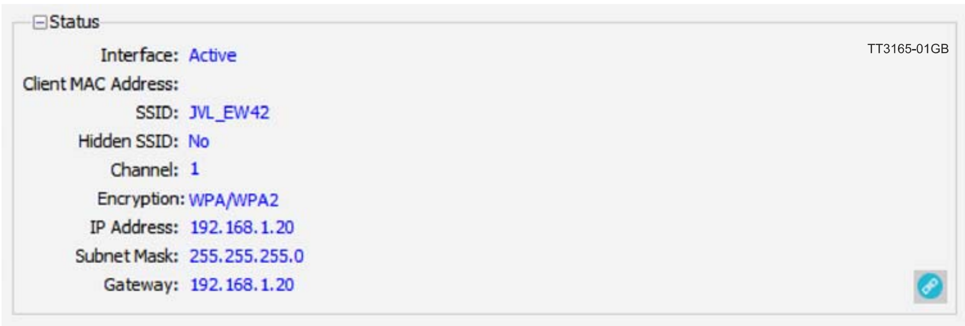
Interface: Shows the connection state to an access point [Connected / Disconnected].

MAC Address: Current MAC address used for the wireless interface. This address should be the same as the MAC address printed on the label.

SSID: Current SSID configuration. This is the SSID the motor will try to connect to.

Encryption: Current encryption setting [WPA/WPA2 or OPEN]. Observe that the password is protected from viewing.

Signal strength: When the motor is connected, the signal strength can be monitored. The units can be presented in either [dBm] or [%].
In good industrial access points the signal strength of the clients can be monitored as well.

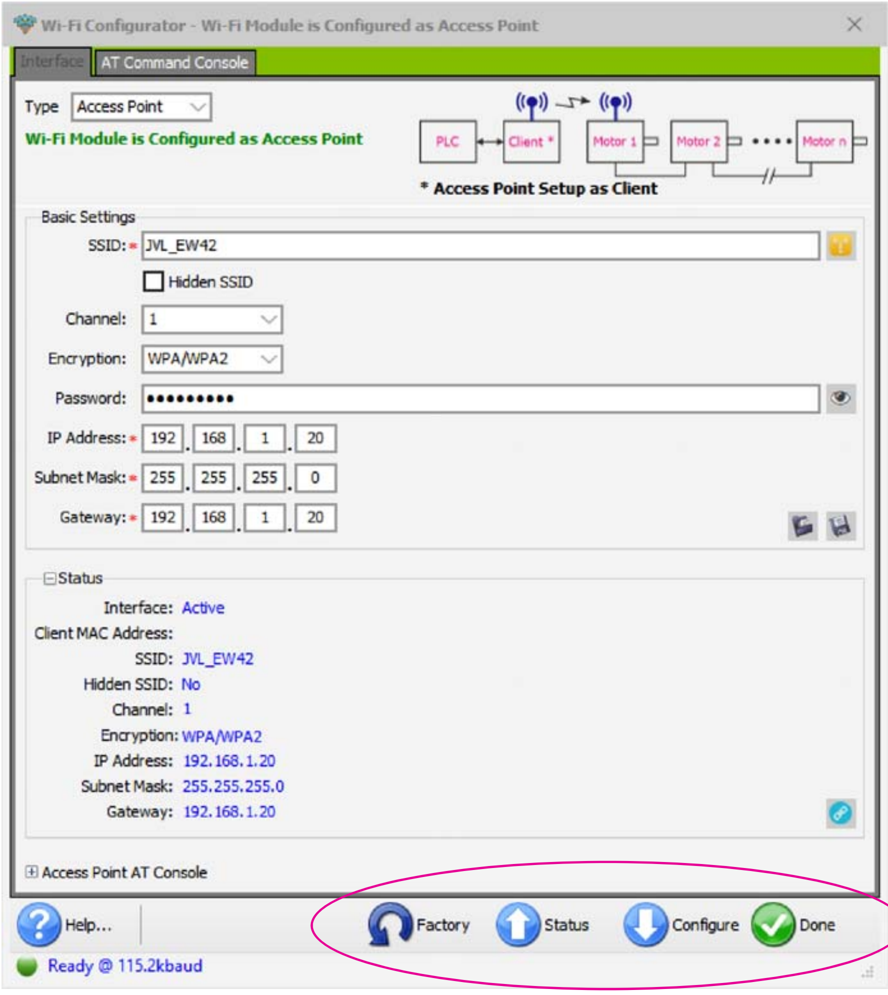


Interface:	Active/Passive
Client MAC Address:	Not used
SSID:	Setting of the SSID that clients can connect to.
Hidden SSID:	Yes/No
Channel:	Active channel number
Encryption:	Encryption method used.
IP Address:	IP address of the wireless access point in the motor.
Subnet Mask:	Subnet mask
Gateway:	Gateway

9.4 Configuration #2, Access point

When the settings has been configured, the setup is transferred to the motor by pressing the "Configure" -button found in the bottom of the dialogue.

TT3154-01GB

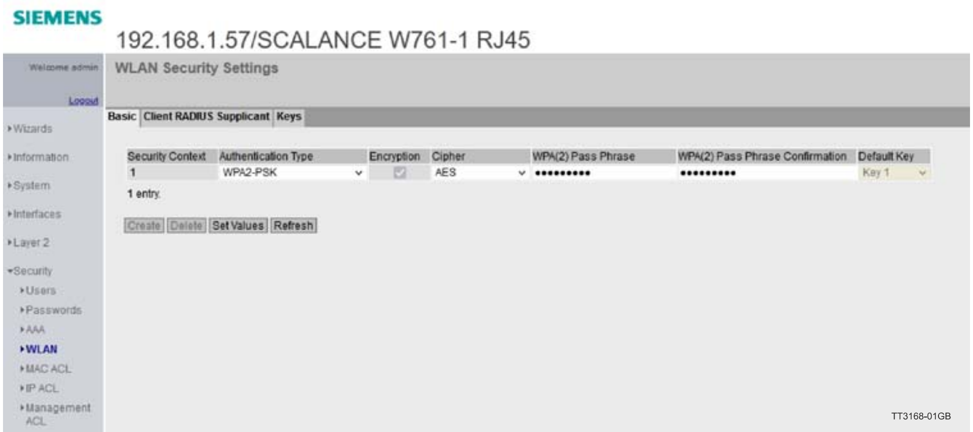


TT3166-01GB

9.4 Configuration #2, Access point

9.4.4 Setting up the accesspoint for the Configuration #2

Observe the Device mode and the Radio mode which is set to "Client".
Encryption is configured according to the encryption settings configured from MacTalk.



The Password is of course defined in the motor, since the motor acts as the access point. The Password is in this case MONKEY123 as configured from MacTalk.

When everything has been configured, the motor will wait for Client(s) to connect, when a client is connected, the LED's on the motor will show the following indication:

TT3169-01GB

Green and Blue indicates that the internal bridging between the wireless chip and the Ethernet controller has been established.

The blue LED indicates that the wireless connection is established. When data is transmittet the LED's are blinking indicating the traffic.

NOTE: The above image shows a motor without the antenna connected, this is purely for demonstration reasons, do not operate the motor without the antenna connected.

9.5 WIFI settings for optimal quality

In some cases it may be necessary to reduce the RPI (Requested Package Interval) compared to a classic wired connection.
The bandwidth of the wireless network can be significantly lower than the bandwidth of a wired connection.

In order to have a stable reliable connection it is imperative needed that the correct settings are achieved in the access point.
Using a good access point designed to work in an automation environment is the first step to success establishing a wireless network.
Wireless networks will always be less deterministic than a traditional wired Ethernet network, so careful planning must be done with considerations of lost connections, lost packages etc.
Most access points designed for industrial use, supports a lot of fine tuning to optimize the infrastructure for the actual usage end existing infrastructure.
This section will cover some of the considerations and features modern access points offer for industrial use.
All demonstrations are carried out using the **Siemens Scalence W761** -Access point and configuration software.

First step in determining how reliable or/and fast a wireless network can be on the plant floor is to analyze the channels in use and the traffic load on each.
Please note that most access points offer running on both 2.4GHz and 5GHz so in case channels are heavily loaded on 2.4GHz, 5 GHz can be used instead.

The JVL wireless system supports running either 2.4GHz or 5GHz.

The channel selection determines whether the motor operates on 2.4GHz or 5 GHz.

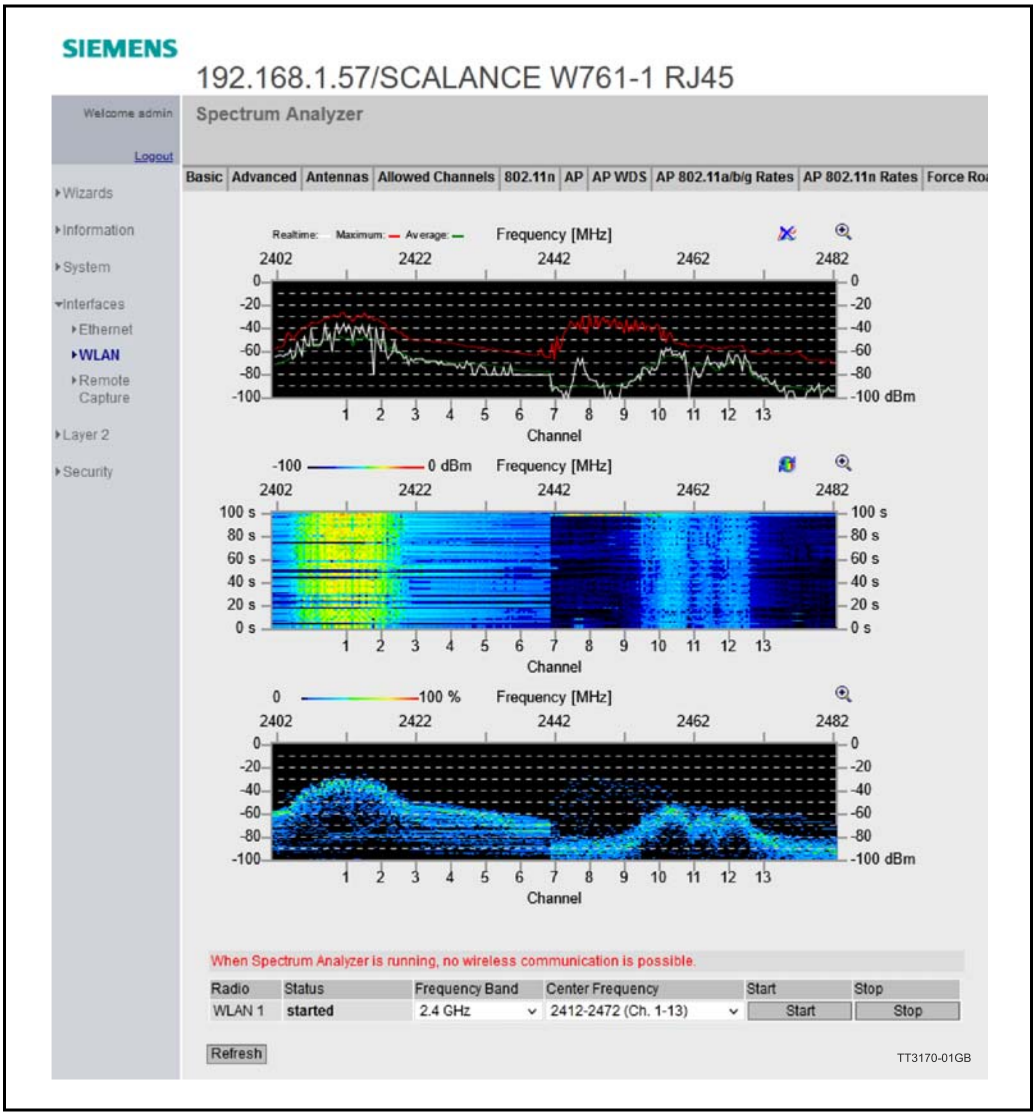
For operation in 2.4GHz, select a channel in the range: **1 - 11**

For operation in the 5GHz band, select a channel in the range: **36 - 64, 100 - 116, 132 - 140.**

NOTE:
Observe the local regulations of allowable channels before commissioning.

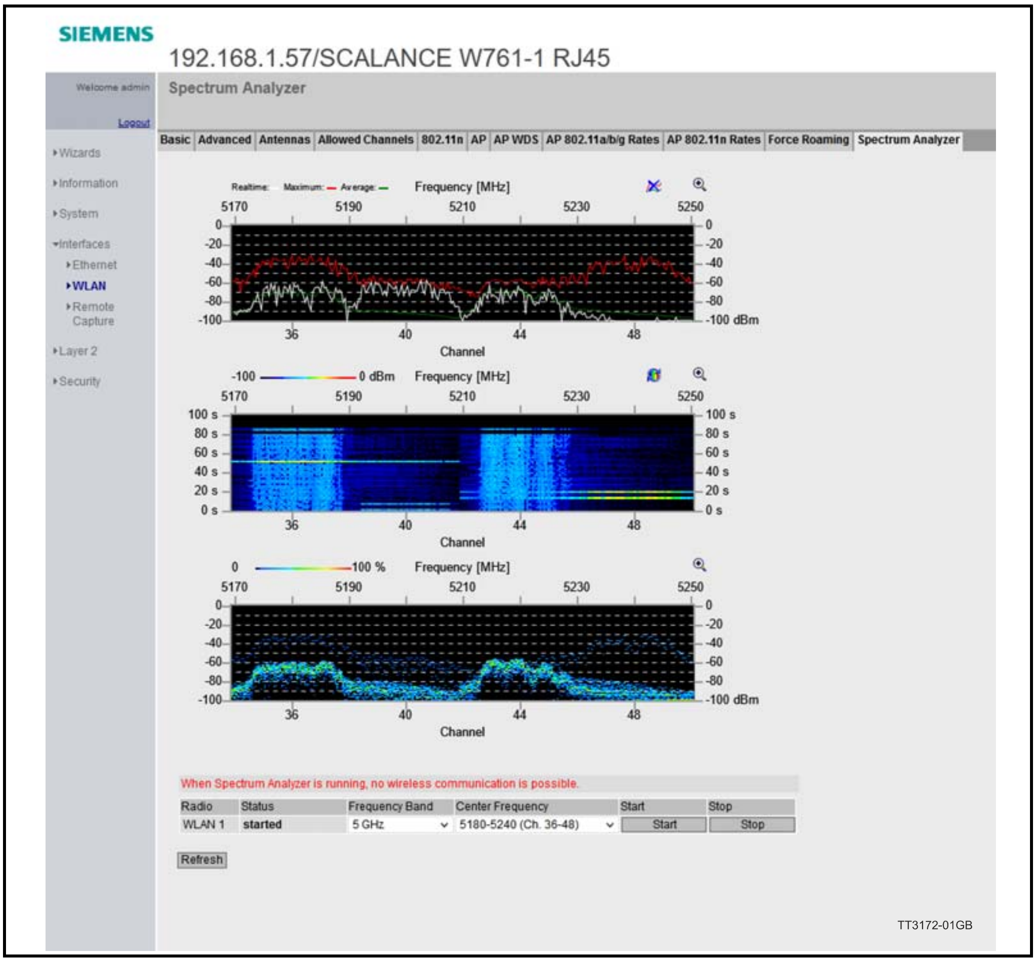
9.5 WIFI settings for optimal quality

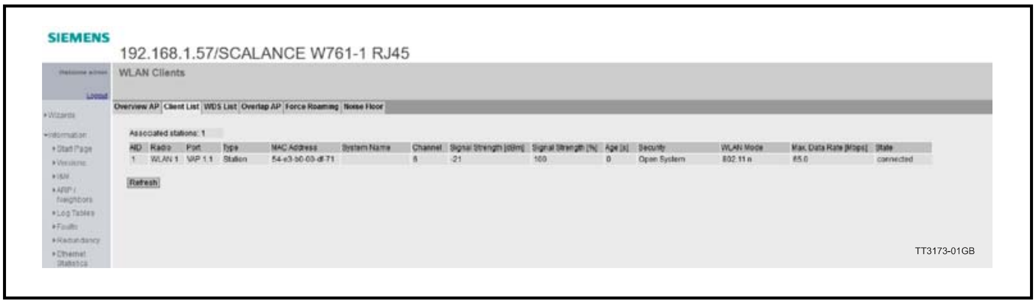
Spectrum analyzer in Siemens Scalence W761 Showing the channel load on the different channels at 2.4GHz.



9.5 WIFI settings for optimal quality

Motor and access point appx. 300mm apart using a std. JVL antenna on the motor.
On the 5GHz frequency band the channel loading on Channel 36-48, leaving Channel 40 as a good choice of operation:





Note the reception quality in the motor can be monitored from the access point web interface.
Note that all the settings are available from TIA portal and can be saved into a TIA portal project.

All JVL examples include the complete project for TIA Portal or Rockwell studio.
A lot of information is available on the internet regarding planning and designing of wireless networks. Covering everything is beyond the scope of this manual.

9.6 Use cases/ example projects

On the JVL web page in the download section all the example projects can be found and downloaded.

Follow the link: https://www.jvl.dk/List/310/Downloads

10 Module Registers

10.1 Register Overview

The module registers are accessible from Mactalk or from the Ethernet with the installed protocol.

10.1.1 Register list.

Register number	Type	Read only	Default	Description
0	UNSIGNED8	X	63	Subindex count
1	UNSIGNED32	X		High 16 bit of MAC address (placed in low 16 bit of word)
2	UNSIGNED32	X		Low 32 bit of MAC address
3	UNSIGNED32			IP address
4	UNSIGNED32			Net mask
5	UNSIGNED32			Gateway
6	UNSIGNED32		0x0	Setup bits
7	UNSIGNED32		0	Digital outputs on module
8	UNSIGNED32		0	Poll division factor
9	UNSIGNED32		0	Station alias
10	UNSIGNED32		-	Modbus TCP time out value
11	UNSIGNED32		-	Input mask
12	UNSIGNED32		-	Default homing method
13	UNSIGNED32		-	Reset delay
14	UNSIGNED32		-	Reserved for future use
15	UNSIGNED32		-	Command register
16	UNSIGNED32		2	Register no. to place in Cyclic Read word, position 1.
17	UNSIGNED32		10	Register no. to place in Cyclic Read word, position 2.
18	UNSIGNED32		12	Register no. to place in Cyclic Read word, position 3.
19	UNSIGNED32		169	Register no. to place in Cyclic Read word, position 4.
20	UNSIGNED32		35	Register no. to place in Cyclic Read word, position 5.
21	UNSIGNED32		-	Register no. to place in Cyclic Read word, position 6.
22	UNSIGNED32		-	Register no. to place in Cyclic Read word, position 7.
23	UNSIGNED32		-	Register no. to place in Cyclic Read word, position 8.
24	UNSIGNED32		2	Register no. to place in Cyclic Write word, position 1.
25	UNSIGNED32		3	Register no. to place in Cyclic Write word, position 2.
26	UNSIGNED32		5	Register no. to place in Cyclic Write word, position 3.
27	UNSIGNED32		7	Register no. to place in Cyclic Write word, position 4.
28	UNSIGNED32		0	Register no. to place in Cyclic Write word, position 5.
29	UNSIGNED32		-	Register no. to place in Cyclic Write word, position 6.
30	UNSIGNED32		-	Register no. to place in Cyclic Write word, position 7.
31	UNSIGNED32		-	Register no. to place in Cyclic Write word, position 8.
32	UNSIGNED32	X	-	Module serial no.
33	UNSIGNED32	X	-	Module hardware version
34	UNSIGNED32	X	-	Module software version
35	UNSIGNED32	X	-	No. of internal motor communication timeouts
36	UNSIGNED32	X	-	No. of retry frames to motor
37	UNSIGNED32	X	-	No. of discarded frames to motor
38	UNSIGNED32	X	-	Total no. of frames to motor
39	UNSIGNED32	X	-	No. of SPI CRC errors
40-46	UNSIGNED32	X	-	Reserved for future use
47	UNSIGNED32	X	-	Digital inputs on module
48	UNSIGNED32	X	-	Status bits
49	UNSIGNED32	X	-	Installed protocol type
50-63				Reserved for future use
N	UNSIGNED32			Access to the motor parameter n

Note: Module parameters are not automatically saved to permanent memory after a change. The parameters can be saved permanently by applying a "Save parameters to flash" command afterwards.

10.2 Register Descriptions.

10.2.1 Register 1 - MAC address MSB.

The 2 most significant bytes of the module MAC address is placed here.

Bit	16-31	0-15
Output	Reserved	16 Most significant bits of MAC address.

10.2.2 Register 2 - MAC address LSB.

The 2 most significant bytes of the module MAC address is placed here.

Bit	0-31
Output	32 Least significant bits of MAC address.

10.2.3 Register 3 - IP address.

This register is the IP address of the device. The use of the IP address is very protocol dependent. With the Powerlink protocol only the node ID part is writeable the rest of the IP address is fixed. With the other protocols the entire address is writeable.

Protocol/Bit	24-31	16-23	8-15	0-7	Notes
EthernetIP, ModbusTCP,	0-255	0-255	0-255	1-254	
Sercos III	0-255	0-255	0-255	1-254	
EtherCAT	- Setup in the PLC project -				Only used for EoE
Profinet,	0-255	0-255	0-255	1-254	Only powerup default. Are usually changed by the PLC later.
Powerlink	192*	168*	100*	NodeID	Only the node ID is writeable

* Fixed.

10.2.4 Register 4 - Netmask.

This register is the netmask of the device. The use of the netmask is very protocol dependent.

Protocol/Bit	24-31	16-23	8-15	0-7	Notes
EthernetIP, ModbusTCP,	0-255	0-255	0-255	1-254	
Sercos III	0-255	0-255	0-255	1-254	
EtherCAT	- Setup in the PLC project -				Only used for EoE
Profinet,	0-255	0-255	0-255	1-254	Only powerup default. Are usually changed by the PLC later.
Powerlink	255*	255*	255*	0*	Read only

* Fixed.

10.2.5 Register 5 - Gateway.

This register is the gateway address of the device. The use of the gateway is very protocol dependent.

Protocol/Bit	24-31	16-23	8-15	0-7	Notes
EthernetIP, ModbusTCP	0-255	0-255	0-255	1-254	
Sercos III	0-255	0-255	0-255	1-254	
EtherCAT	Setup in the PLC project				Only used for EoE
Profinet,	0-255	0-255	0-255	1-254	Only powerup default. Are usually changed by the PLC later.
Powerlink	192*	168*	100*	254*	Read only

* Fixed.

10.2.6 Register 6 - Setup bits

This register is used to setup the module configuration and how the module should react on different events.

Bit	13-31	12	11	10	9	8
Output	Reserved	Swap data-bytes	CiA402 Units	DHCP Enable	Input Mirror	Output Mirror

Bit	7	6	5	4	3	2	1	0
Output	Input debounce	PDO - 8 registers	Mirror registers	Endless relative	Enable drive profile	Clear “Name of station”	Disable error handling	Ethernet error handling.

Change of bits also requires save in flash (save to permanent memory) and a power cycle to be activated.

Bit 0
Ethernet error handling 0* = Set motor to passive mode when error occurs.
1 =Set velocity to 0 when error occurs (active brake).

Bit 1
Disable error handling 0* = Ethernet error handling enabled.
1 =Ethernet error handling disabled.

Bit 2
Clear “Name of station” 0 = "Name of station" is preserved after power up.
1* = "Name of station" is blank after power up. *Only applicable to the Profinet protocol.*

Bit 3
Enable drive profile 0 = JVL drive profile.
1* = Drive profile enabled. (CiA® DSP-402 if EtherCAT®, FSPDrive if Sercos®)

Bit 4
Endless relative 0* = Endless relative disabled.
1 = Endless relative enabled. If relative mode is selected in the control word, then the actual position never changes. When selecting this mode absolute positioning can no longer be used.
This bit only applies for DSP-402 profile.

Bit 5
Mirror registers 0* = No mirror of module registers.
1 = Enable mirror of module registers to start address 0x300. *Only applicable to the ModbusTCP protocol.*

Bit 6
PDO - 8 registers 0* = 5 x 32 bit registers in each PDO.
1 = 8 x 32 bit registers in each PDO. *Requires a save in flash and a power cycle to be activated. Only applicable to JVL profile (not DSP-402 and not FSPDrive).*

Bit 7
Input de-bounce 0* = No input de-bounce
1 = Enable de-bounce on input 1-4, when mirror to motor. (Results in a 15-25ms delay, where normal response time is below 1ms).

Continued next page

Bit 8		
Output mirror	0* =	No output mirror
	1 =	Enable mirror of module outputs from motor error register bit 30-31.
Bit 9		
Input mirror	0* =	No input mirror
	1 =	Enable mirror of module inputs to motor register 210 bit 2-5.
Bit 10		
DHCP enable	0* =	DHCP disabled.
	1 =	Enable of DHCP in module. *Only applicable to Ethernet-IP and ModbusTCP protocols.*
Bit 11		
CiA402 units	0* =	SI units in all CiA402 velocity and accelaration objects.
	1 =	Motor native units in CiA402 set-velocity and set-acceleration objects which are: 0x606B - Velocity demand value 0x6081 - Profile Velocity 0x6083 - Profile Acceleration 0x6085 - Quick stop decelleration 0x6099 - Homing speeds 0x609A - Homing Acceleration 0x60FF - Target Velocity
Bit 12		
Swap databytes	0* =	No swapping of databytes. Compatible with standard PLC.
	1 =	Swap databytes. Compatible with for example KUKA robot PLC. Only applicable to Profinet.
Bit 13		
Cycle setup	0* =	Standard cyclic setup.
	1 =	Alternative cyclic setup used for MST drives with external encoder. Replaces actual position and actual velocity with external encoder position and velocity. Only applicable to Sercos.

* Factory default.

10.2.7 Register 7 - Digital outputs on module.

Only applicable for MAC00-Ex4/-Ex41
With this object the digital outputs can be controlled.
The value written to this object is directly shown on the digital outputs.

Bit	2-31	1	0
Output	Reserved	Output2* (O2)	Output1* (O1)

* The availability of the outputs depends on the actual version of the module used.
MAC00-EC4 only support Output 1 (O1).
MAC00-EC41 supports both output 1 and 2 (O1 and O2).

10.2.8 Register 8 - Poll division factor.

With this object a poll division factor can be set. This enables use of cycle times faster than the motor is capable of. If for example having a MAC050-141 and 5 cyclic write and 5 cyclic read registers, then a minimum cycle time of 20ms is needed. Instead it is possible to have a net cycle time of 1ms, and a poll division factor of 20. Then the motor internally only get updated every 20ms.

Bit	16-31	0 - 15
R/W	Reserved	Poll division factor

Only applicable for EtherCAT® in MAC050-141. Only read at power-up, or after reset. So in order to change the value, first change this value, then issue a “save in flash” command, then reset the module.

10.2.9 Register 9 - Station alias (node number).

Only applicable to EtherCAT.
With this object a station alias (node number) is set manually.

Bit	16-31	0 - 15
R/W	Reserved	Station alias

Only read at power-up, or after reset. So in order to change the value, first change this value, then issue a “save in flash” command, then reset the module.

10.2.10 Register 10 Modbus time-out

Only applicable to ModbusTCP.
The Modbus TCP protocol does not have an implementation for timeout on application layer and this may be required when controlling a drive. A supervision method has been implemented for this purpose. If modbus timeout is set to zero, this feature is disabled. The unit of the parameter is 100ms (e.g. “35” will give 3.5 seconds).

Bit	16-31	0-15
Output	Reserved	Modbus timeout in units of 100ms.

10.2.11 Register 11 - Input mask

Only applicable for MAC00-Ex4/-Ex41

Only applicable to EtherCAT® with CiA402 drive profile.

This register is used to setup input mask on the digital inputs (IN1-4).

Bit	20-31	16-19	15-12	11-8	7-4	3-0
Output	Reserved	No mask	Reserved	PL mask	Reserved	NL mask

NL mask Bit set results in that corresponding input is configured as Negative Limit switch. Bit 0-3 corresponds to IN1-4.

PL mask Bit set results in that corresponding input is configured as positive Limit switch. Bit 8-11 corresponds to IN1-4.

NO mask Bit set results in corresponding limit switch input is inverted and now acts as a normally open (NO) switch. Bit 16-19 corresponds to IN1-IN4.

10.2.12 Register 12 - Default homing method.

Only applicable to EtherCAT® with CiA402 drive profile.

This register sets a default method used for homing in CiA402 drive profile. The value of this register is copied to object 6098 - *Homing Method* at power up. This register has to be saved in module flash.

Bit	31-8	7-0
R/W	Reserved	Default homing method

10.2.13 Register 13 – Reset command delay

Specification of delay before reset, used with the commands '2', '258' and '259' to the Ethernet module command register 15 (= register 983040 when selected in Mactalk). The reset is executed the specified number of ms after receipt of one of the mentioned commands.

The default value is 500 corresponding to 0.5s. Possible range is 100 to 10000 corresponding to 0.1s to 10s.

10.2.14 Register 15 - Command register

This object is used for sending commands to the module and is write only.
Use this register instead of the MAC/MIS/MIL motor command register when used cyclic, to make sure that commands are only executed once.
If this register is placed in the cyclic list, and it is requested to execute the same command more than once, then it is required to send a "No operation" command in between.

The possible commands are listed in the three tables shown below and in the next pages.

- The first table lists commands executed by the Ethernet module it self, which is mainly common to all MAC and MIS/MIL motors.
- The second table lists commands which are redirected to the MAC motor command register and executed by the MAC motor controller, if the Ethernet module is installed in a MAC motor.
- The third table lists commands which are redirected to the MIS/MIL motor command register and executed by the MIS/MIL motor controller, if the Ethernet module is installed in a MIS/MIL motor.

Be aware that some of the commands only applies for specific protocols.

Common commands

Command no.		Command description			Active from FW build no.
Hex	Dec	MAC050 - MAC141	MAC400 to MAC4500	MIS/MILxxx	
Module only commands					
0x 0000 0000	0	No operation	< Same as	< Same as	-
0x 0000 0001	1	Reset the module	< Same as	< Same as	-
0x 0000 0002	2	Reset Ethernet module after delay. Default delay=500ms. Set delay in Ethernet module register 13 in ms. The reset is executed the specified number of ms after receipt of command.	< Same as	< Same as	-
0x 0000 0010	16	Save module parameters to flash	< Same as	< Same as	-
0x 0000 0011	17	Flash with power LED for 120 seconds. Only applicable for MAC motors. For MIS/MIL motors use commands specific for MIS/MIL.	< Same as	-	-
0x 0000 0012	18	Restore factory defaults.	< Same as	< Same as	-
0x 0000 0013	19	Copy Sync0 pulse to Out1	< Same as	No operation	-
0x 0000 0014	20	Remove Sync0 from Out1	< Same as	No operation	-
0x 0000 0015	21	Re-init cyclic data	< Same as	< Same as	10150
0x 0000 0016	22	Disable cyclic writes to motor. Used for re-initialisation of the cyclic data if they are changed and you do not want to reset the module.	< Same as	< Same as	10150
0x 0000 0017	23	Re-enable cyclic write to motor	< Same as	< Same as	10150
0x 0000 0019	25	Re-initialize internal communication between Ethernet module and motor controller, without reset. To be used if the motor controller has been reset without the module being reset.	< Same as	< Same as	-

10.2 Register Descriptions.

Command no.		Command description			Active from FW build no.
Hex	**Dec**	**MAC050 - MAC141**	**MAC400 to MAC4500**	**MIS/MILxxx**	
Synchronized commands					
0x 0000 0101	257	Simultaneous reset of the motor and the module	< Same as	< Same as	-
0x 0000 0102	258	Reset Ethernet module and motor controller after delay. Default delay=500ms. Set delay in Ethernet module register 13 in ms.			
0x 0000 0103	259	Reset Motor controller only after delay. Default delay=500 ms. Set delay in Ethernet module register 13 in ms. This command does NOT reset the Ethernet module, so Ethernet switch will not get disturbed. This command works on both MAC and MIS motors. **!** After this command the internal communication will NOT work, so a command for resseting the Ethernet module or re-initialize internal communication is required afterwards **!**			
0x 0000 0110	272	Save the motor parameters in flash memory, and do a re-sync. of internal communication afterwards.	< Same as	< Same as	-
0x 0000 0111	273	Save the motor controller parameters/ registers in flash. This will also reset the motor controller, but NOT the Ethernet module, so the Ethernet switch will not get distrurbed. This command works on both MAC and MIS motors. **!** After this command the internal communication will NOT work, so a command for resetting the Ethernet module or re-initialize the internal communication is required afterwards. **!**			

MAC motor commands" Only applicable for motors with module MAC00-Ex4/-Ex41

Use only the table below for **firmware build numbers above 1400**. If using an Ethernet module firmware with a lower build number then refer to the table on the next page.

Command no.		Command description	
Hex	**Dec**	**MAC050 - MAC141**	**MAC400 – MAC4500**
0x 0100 0001	16777217	Reset motor *(not recommended, use synchronized version instead)*.	< Same as
0x 0100 0002	16777218	Save motor parameters in flash and reset motor *(not recommended, use synchronized version instead)*.	< Same as
0x0100 00E0	16777440	No operation	< Same as
0x0100 00E1	16777441	Reset error *(Clear error bits in motor register 35)*	< Same as
0x0100 00E2	16777442	P_SOLL = 0	< Same as
0x0100 00E3	16777443	P_IST = 0	< Same as
0x0100 00E4	16777444	P_FNC = 0	< Same as
0x0100 00E5	16777445	V_SOLL = 0	< Same as
0x0100 00E6	16777446	T_SOLL = 0	< Same as
0x0100 00E7	16777447	Reset IN_POS, AC C,DEC	< Same as
0x0100 00E8	16777448	P_FNC = (FLWERR - P7) * 16	< Same as
0x0100 00E9	16777449	P_FNC = (FLWERR - P8) * 16	< Same as
0x0100 00EA	16777450	Reserved	< Same as
0x0100 00EB	16777451	Reserved	< Same as
0x0100 00EC	16777452	Activate P1,V1,A1,T1,L1,Z1	< Same as
0x0100 00ED	16777453	Activate P2,V2,A2,T2,L2,Z2	< Same as
0x0100 00EE	16777454	Activate P3,V3,A3,T3,L3,Z3	< Same as
0x0100 00EF	16777455	Activate P4,V4,A4,T4,L4,Z4	< Same as
0x0100 00F0	16777456	Start search zero	< Same as
0x0100 00F1	16777457	P_SOLL = P_IST + P7;	P_SOLL = P_IST + P7 – FLWERR;
0x0100 00F2	16777458	P_SOLL = P_IST + P8;	P_SOLL = P_IST + P8 – FLWERR;
0x0100 00F3	16777459	Reserved	< Same as
0x0100 00F4	16777460	Select absolute position mode	< Same as
0x0100 00F5	16777461	Select relative position mode using P_SOLL	< Same as
0x0100 00F6	16777462	Select relative position mode using P_FNC	< Same as
0x0100 00F7	16777463	Synchronize position manually using absolute new values. P_IST = P_NEW; P_SOLL = P_NEW; P_FUNC = P_NEW * 16;	Synchronize position manually using absolute new values. P_IST = P_NEW; P_SOLL = P_NEW; P_FNC = (P_NEW + FLWERR)*16;
0x0100 00F8	16777464	Synchronize position manually using relative new values. (basically offset the position range with the value of P_NEW). P_IST = P_IST + P_NEW; P_SOLL = P_SOLL + P_NEW; P_FUNC = P_FUNC + (P_NEW * 16);	< Same as
0x0100 00F9	16777465	No operation	< Same as
0x0100 00FA	16777466	No operation	< Same as
0x0100 00FB	16777467	No operation	< Same as
0x0100 00FC	16777468	No operation	< Same as
0x0100 00FD	16777469	Reserved	< Same as
0x0100 00FE	16777470	Reserved	< Same as
0x0100 00FF	16777471	Reserved	< Same as

Table continued next page

Use only the table below for **firmware build numbers lower than 1400**. If using an Ethernet module firmware with a higher build number then refer to the table on the previous page.

Command no.		Command description	
Hex	**Dec**	**MAC050 - MAC141**	**MAC400 – MAC4500**
0x 8000 0001	2147483649	Reset motor *(not recommended, use synchronized version instead).*	< Same as
0x 8000 0002	2147483650	Save motor parameters in flash and reset motor *(not recommended, use synchronized version instead).*	< Same as
0x8000 00E0	2147483872	No operation	< Same as
0x8000 00E1	2147483873	Reset error *(Clear error bits in motor register 35)*	< Same as
0x8000 00E2	2147483874	P_SOLL = 0	< Same as
0x8000 00E3	2147483875	P_IST = 0	< Same as
0x8000 00E4	2147483876	P_FNC = 0	< Same as
0x8000 00E5	2147483877	V_SOLL = 0	< Same as
0x8000 00E6	2147483878	T_SOLL = 0	< Same as
0x8000 00E7	2147483879	Reset IN_POS, AC C,DEC	< Same as
0x8000 00E8	2147483880	P_FNC = (FLWERR - P7) * 16	< Same as
0x8000 00E9	2147483881	P_FNC = (FLWERR - P8) * 16	< Same as
0x8000 00EA	2147483882	Reserved	< Same as
0x8000 00EB	2147483883	Reserved	< Same as
0x8000 00EC	2147483884	Activate P1,V1,A1,T1,L1,Z1	< Same as
0x8000 00ED	2147483885	Activate P2,V2,A2,T2,L2,Z2	< Same as
0x8000 00EE	2147483886	Activate P3,V3,A3,T3,L3,Z3	< Same as
0x8000 00EF	2147483887	Activate P4,V4,A4,T4,L4,Z4	< Same as
0x8000 00F0	2147483888	Start search zero	< Same as
0x8000 00F1	2147483889	P_SOLL = P_IST + P7;	P_SOLL = P_IST + P7 – FLWERR;
0x8000 00F2	2147483890	P_SOLL = P_IST + P8;	P_SOLL = P_IST + P8 – FLWERR;
0x8000 00F3	2147483891	Reserved	< Same as
0x8000 00F4	2147483892	Select absolute position mode	< Same as
0x8000 00F5	2147483893	Select relative position mode using P_SOLL	< Same as
0x8000 00F6	2147483894	Select relative position mode using P_FNC	< Same as
0x8000 00F7	2147483895	Synchronize position manually using absolute new values. P_IST = P_NEW; P_SOLL = P_NEW; P_FUNC = P_NEW * 16;	Synchronize position manually using absolute new values. P_IST = P_NEW; P_SOLL = P_NEW; P_FNC = (P_NEW + FLWERR)*16;
0x8000 00F8	2147483896	Synchronize position manually using relative new values. (basically offset the position range with the value of P_NEW). P_IST = P_IST + P_NEW; P_SOLL = P_SOLL + P_NEW; P_FUNC = P_FUNC + (P_NEW * 16);	< Same as
0x8000 00F9	2147483897	No operation	< Same as
0x8000 00FA	2147483898	No operation	< Same as
0x8000 00FB	2147483899	No operation	< Same as
0x8000 00FC	2147483900	No operation	< Same as
0x8000 00FD	2147483901	Reserved	< Same as
0x8000 00FE	2147483902	Reserved	< Same as
0x8000 00FF	2147483903	Reserved	< Same as

Table continued next page

10.2 Register Descriptions. Only MIS/MIL

Use only the table below for **firmware build numbers above 1400**. If using an Ethernet firmware with a lower build number then refer to the table on the next page.

MIS/MIL motor commands - Only applicable for MIS/MIL motors

Command no.		Command description
Motor only FastMac commands *(via module cmd register)*		
Hex	**Dec**	
0x0100 0060	16777312	No operation
0x0100 0061	16777313	Reset errors and warnings
0x0100 0062	16777314	P_SOLL = 0
0x0100 0063	16777315	P_IST = 0
0x0100 0064	16777316	Reserved (do not use)
0x0100 0065	16777317	V_SOLL = 0
0x0100 0066	16777318	Reserved (do not use)
0x0100 0067	16777319	Reset InPos, Acc, Dec
0x0100 0068	16777320	Reserved (do not use)
0x0100 0069	16777321	Reserved (do not use)
0x0100 006A	16777322	Reserved (do not use)
0x0100 006B	16777323	Reserved (do not use)
0x0100 006C	16777324	Activate P1, V1, A1, T1
0x0100 006D	16777325	Activate P2, V2, A2, T2
0x0100 006E	16777326	Activate P3, V3, A3, T3
0x0100 006F	16777327	Activate P4, V4, A4, T4
0x0100 0070	16777328	Start zero search
0x0100 0071	16777329	P_SOLL = P_IST + P7
0x0100 0072	16777330	P_SOLL = P_IST + P8
0x0100 0073	16777331	No operation
0x0100 0074	16777332	Select absolute position mode
0x0100 0075	16777333	Select relative position mode
0x0100 0076	16777334	Reserved, do not use
0x0100 0077	16777335	Copy P_NEW to both P_SOLL and P_IST
0x0100 0078	16777336	Add P_NEW to both P_SOLL and P_IST
0x0100 0079	16777337	Reserved (do not use)
0x0100 007A	16777338	Reserved (do not use)
0x0100 007B	16777339	Reserved (do not use)
0x0100 007C	16777340	Reserved (do not use)
0x0100 007D	16777341	Reserved (do not use)
0x0100 007E	16777342	Reserved (do not use)
0x0100 007F	16777343	Reserved (do not use)
Motor only normal commands *(via module cmd register)*		
0x0100 0101	16777473	Re-sync P_IST and P_ENCODER position
0x0100 010B	16777483	Reset motor *(not recommended, use synchronized version instead)*.
0x0100 010C	16777484	Save motor parameters in flash and reset motor *(not recommended, use synchronized version instead)*.
0x0100 010D	16777485	Save to flash memory, then continue normal execution. NOTE: Some registers used only during startup!
0x0100 0171	16777585	Turn on LED flashing
0x0100 0172	16777586	Turn off LED flashing

Use only the table below for **firmware build numbers below 1400**. If using an Ethernet firmware with a lower build number then refer to the table on the next page.

MIS/MIL motor commands - Only applicable for MIS/MIL motors

Command no.		Command description
Motor only FastMac commands *(via module cmd register)*		
Hex	**Dec**	
0x8000 0060	2147483744	No operation
0x8000 0061	2147483745	Reset errors and warnings
0x8000 0062	2147483746	P_SOLL = 0
0x8000 0063	2147483747	P_IST = 0
0x8000 0064	2147483748	Reserved (do not use)
0x8000 0065	2147483749	V_SOLL = 0
0x8000 0066	2147483750	Reserved (do not use)
0x8000 0067	2147483751	Reset InPos, Acc, Dec
0x8000 0068	2147483752	Reserved (do not use)
0x8000 0069	2147483753	Reserved (do not use)
0x8000 006A	2147483754	Reserved (do not use)
0x8000 006B	2147483755	Reserved (do not use)
0x8000 006C	2147483756	Activate P1, V1, A1, T1
0x8000 006D	2147483757	Activate P2, V2, A2, T2
0x8000 006E	2147483758	Activate P3, V3, A3, T3
0x8000 006F	2147483759	Activate P4, V4, A4, T4
0x8000 0070	2147483760	Start zero search
0x8000 0071	2147483761	P_SOLL = P_IST + P7
0x8000 0072	2147483762	P_SOLL = P_IST + P8
0x8000 0073	2147483763	No operation
0x8000 0074	2147483764	Select absolute position mode
0x8000 0075	2147483765	Select relative position mode
0x8000 0076	2147483766	Reserved, do not use
0x8000 0077	2147483767	Copy P_NEW to both P_SOLL and P_IST
0x8000 0078	2147483768	Add P_NEW to both P_SOLL and P_IST
0x8000 0079	2147483769	Reserved (do not use)
0x8000 007A	2147483770	Reserved (do not use)
0x8000 007B	2147483771	Reserved (do not use)
0x8000 007C	2147483772	Reserved (do not use)
0x8000 007D	2147483773	Reserved (do not use)
0x8000 007E	2147483774	Reserved (do not use)
0x8000 007F	2147483775	Reserved (do not use)
Motor only normal commands *(via module cmd register)*		
0x8000 0101	2147483905	Re-sync P_IST and P_ENCODER position
0x8000 010B	2147483915	Reset motor *(not recommended, use synchronized version instead)*.
0x8000 010C	2147483916	Save motor parameters in flash and reset motor *(not recommended, use synchronized version instead)*.
0x8000 010D	2147483917	Save to flash memory, then continue normal execution. NOTE: Some registers used only during startup!
0x8000 0171	2147484017	Turn on LED flashing
0x8000 0172	2147484018	Turn off LED flashing

10.2 Register Descriptions.

10.2.15 Register 16-23 - Register no. to place in “Cyclic Read”

These registers contain the numbers that define the registers which are in the Cyclic Read.
That is the register's, which is transmitted from slave to master cyclically. If some of these registers are changed, it is necessary to issue a "save in flash" command and to reboot the device before the changes take effect.

10.2.16 Register 24-31 - Register no. to place in “Cyclic Write”

These registers contain the numbers that define the registers which are in the Cyclic Write.
That is the register's, which is transmitted from master to slave cyclically. If some of these registers are changed, it is necessary to issue a "save in flash" command and to reboot the device before the changes take effect.

10.2.17 Register 32-38

These registers contain HW, SW and communication information of the module.

10.2.18 Register 39 CRC error count on SPI.

This register reflects the no. of CRC errors that have occurred on the internal SPI communication with the motor, since power up.

10.2.19 Register 47 - Digital inputs on module

Only applicable for MAC00-Ex4/-Ex41
With this object the status of the 4 digital inputs can be read.

Bit	4-31	3	2	1	0
Input	Reserved	IN4*	IN3*	IN2*	IN1*

* The availability of the inputs depends on the actual version of the module used.
MAC00-EC4 only support Input 1 (IN1).
MAC00-EC41 supports input 1, 2, 3 and 4 (IN1, IN2, IN3 and IN4).

10.2.20 Register 48 - Status bits

This register is used for miscellaneous information about the module.

Bit	16-31	15	14	13	12	11	10	9	8	7	0-6
Output	Do not use	PL active	NL active	Sync error	RS232 Mactalk	UDP Mactalk	Distributed Clock (DC) enabled – EtherCAT Clock synchronous enabled -Sercos	Cyclic PLC communication is running. Not applicable to ModbusTCP.	Do not use	1=No communication with the motor	Do not use

Further descriptions - see next page

BIT 7
No motor comm.No communication between motor and module.

BIT 9
CYCLIC RunningCyclic communication with PLC running. Not aplicable for the ModbusTCP protocol.

Bit 10
DC enableDistributed Clocks enabled. Only aplicable to EtherCAT.

Bit 11
UDP Mactalk. MacTalk connected via UDP

Bit 12
RS232 MacTalk. MacTalk connected via RS232

Bit 13
SYNC_ERROR. PLL synchronization error in motor

Bit 14
NL_ERROR. Negative limit switch activated

Bit 15
PL_ERROR. Positive limit switch activated.

Bit 16
SPI channel used for internal cyclic communication to motor controller.

Bit 17
Warning: Motor firmware too old to support all module features.

Bit 18
Warning: Cyclic data blocked.

10.2.21 Register 49 - Current protocol type installed in Ethernet module

0x34 = EthernetIP.
0x35 = EtherCAT®.
0x36 = Ethernet POWERLINK.
0x37 = ProfiNet.
0x38 = ModbusTCP.
0x39 = Sercos III.

11 Using MacTalk over Ethernet

11.1 Using MacTalk over Ethernet

11.1.1 Introduction

The configuration software tool MacTalk is able to connect to a motor either using a serial connection or an Ethernet based TCP/IP connection.
Please notice that there are some limitations/precautions.

- MAC00-Exx modules with a hardware version of 1.3 or older do NOT support Mactalk over Ethernet communication. See also *How to find FW/HW version at product, page 13*.

- Currently only the **Profinet**, **EthernetIP**, **Sercos**, **ModbusTCP** and **EtherCAT** protocols support MacTalk over Ethernet.
  PROFINET IO firmware version must be firmware version 3.17 Build 425 or higher.
  EthernetIP must be firmware version 3.21 Build 425 or higher.
  ModbusTCP must be firmware version 3.17 or higher.
  EtherCAT must be min. firmware version 3.25 build 1120 or higher.
  Sercos: All firmware versions support MacTalk over Ethernet.
  See also *How to find FW/HW version at product, page 13*.

- Make sure the motor has the latest firmware installed, that is V2.11 or newer for MAC400-4500 and V9.01 for MAC50-141. Ethernet connectivity is only supported in the MIS/MILxxxxxx**E**xxxxxx series of stepper motors. For the MIS/MIL motors please use firmware V1.12 or greater. All the firmwares required should be included in the install package for MacTalk or by using the internet update feature in MacTalk. See also *How to find FW/HW version at product, page 13*.

- Make sure that Mactalk is version 1.50.49 or newer.

- Firmware updates over the Ethernet channel are only possible if the Ethernet module firmware **build** number are higher than 1400. And only with a Mactalk version of 1.70.27 or newer. And if using MIS/MILxxx motors it also requires the MIS/MIL motor version to be 4.01.073 or higher. See also *How to find FW/HW version at product, page 13*.

- eRxP programming over Ethernet is only possible with MIS/MIL motors or with Ethernet module firmwares with build numbers of at least 1120.

- You should not have a serial cable connected at the same time as Mactalk over Ethernet, as it will ruin any attempts on eRxP programming, and firmware updates.

- In MIS/MIL motors with hardware version before 1.6 OR Ethernet hardware version before 1.3 disturbances of motor run will happen if Mactalk over Ethernet is connected while running in some modes. If there is an info text on the Ethernet tab telling the motor is capable of running with 1ms cycle time, then both the hardware versions are new and OK for this also.

The hardware required is the mandatory 24V supply for the motor and the Ethernet cable going either from an Ethernet switch or directly from the PC to the M12 connector on the MAC00-Exx module in the motor.
In order to establish the Ethernet connection from the PC where MacTalk is running, to the motor the PC and the motor needs to be configured to run on the same subnet.
By default the motor is configured to run on the following IP-address: 192.168.0.XX at startup where XX refers to the last 2 digits in the MAC-ID which is printed on a label.
So, if a MAC-ID has the value: 00 : 50 : C2 : D0 : C9 : 14, then the IP address is set to: 192.168.0.20. Please remark that the MAC address is in hexadecimal and that the IP address is in decimal.

11.1 Using MacTalk over Ethernet

The PC from where MacTalk is used needs to be configured for this IP range.
The in depth PC – configuration is beyond the scope of this manual since this greatly depends on the networks equipment end network connected. However a brief description on how to configure the IP address manually is discussed. This method is necessary if the motor is connected directly to the Ethernet port in the PC or if the network isn't capable of assigning IP addresses to connected equipment automatically.

NOTE!
With laptops or desktops with more than one network card, ie. a wireless one, it can be necessary to turn off the unused one, to force windows to route the requests the correct way.

11.2 Setting up the Ethernet at the PC

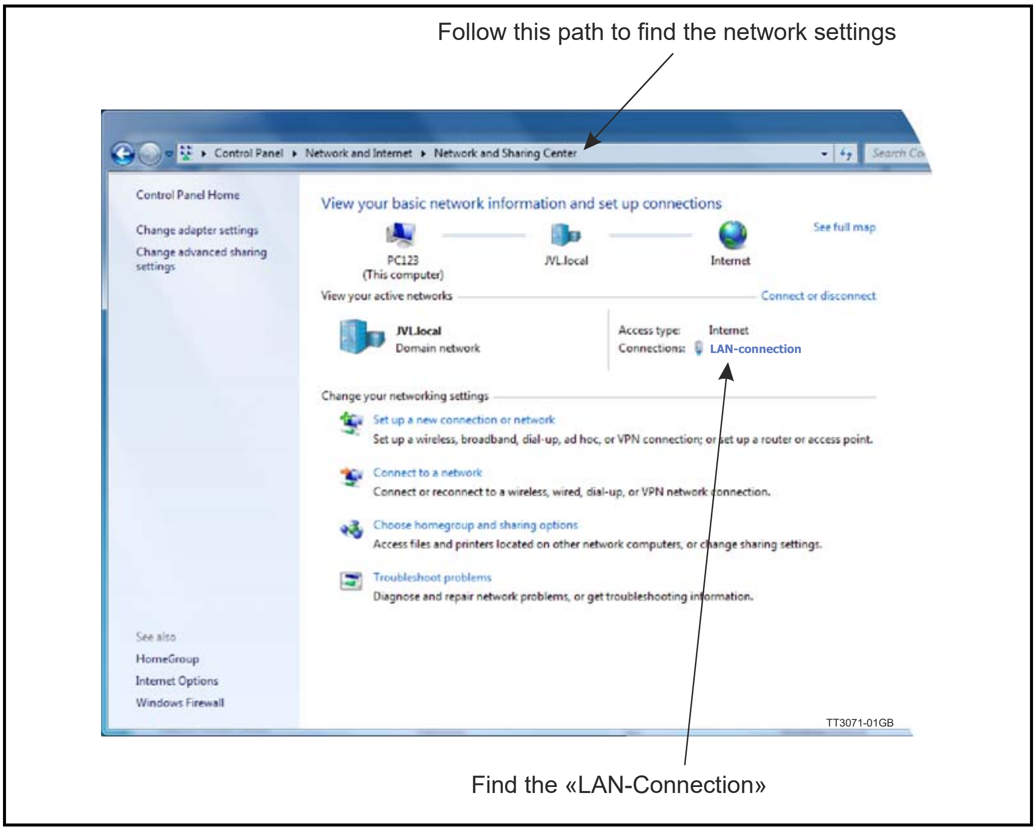
11.2.1 Setting up the PC for EthernetIP, Profinet, ModbusTCP or Sercos.

When a connection is made directly from the PC it is very important to observe the IP-settings of the PC, since the most common way is for the PC to receive the settings from a server such as a DHCP/server or similar.
Since the motor doesn't offer any DHCP service it is necessary to setup the IP-address in the PC manually.

Please note that this is taken from Windows 7, but the method is basically the same for other Windows version.

To reach the IP settings please follow this path:

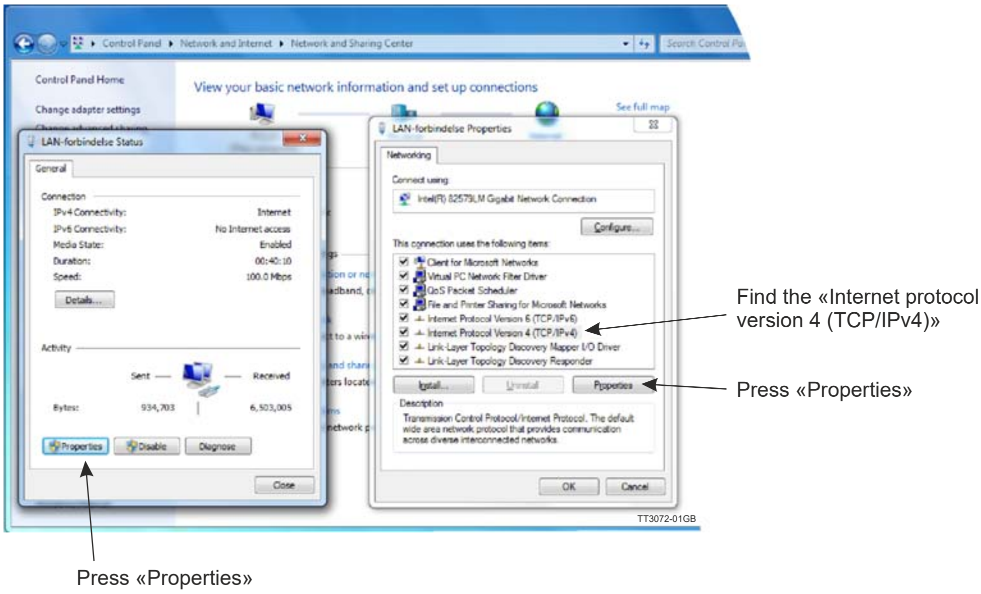
Step 1.
Press the LAN-Connection and press the "Properties".



Continued next page

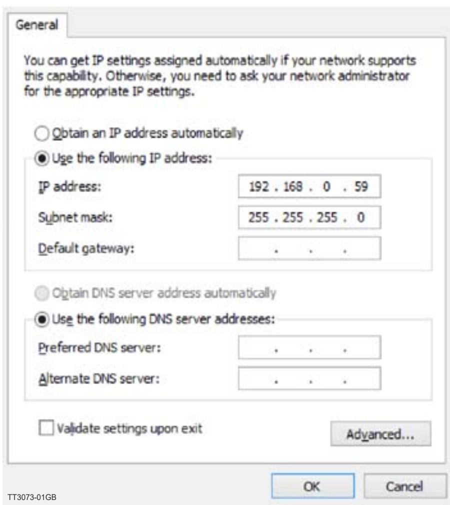
11.2 Setting up the Ethernet at the PC

Step 2.
Find the “Internet protocol version 4 (TCP/IPv4)” and press “Properties”.



Now the settings finally appears and we are able to change the IP address, subnet mask and gateway.

Step 3.
Select “Use the following” and enter a valid configuration similar to the one below.

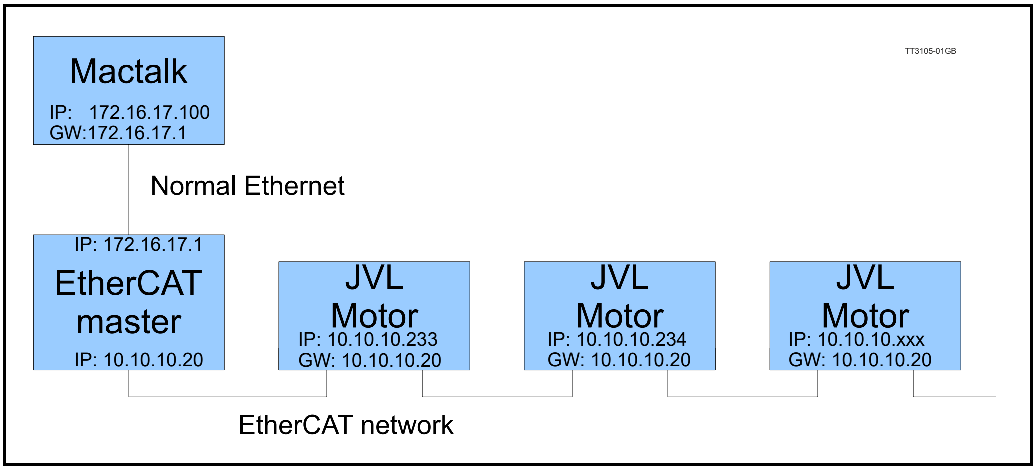


The above example is a basic settings that sets the IP address on the PC to 192.168.0.59, subnet mask to 255.255.255.0 and the gateway to 1.1.1.1.
Now the PC is configured for a fixed IP address and is ready to establish the connection to the motor.

11.2 Setting up the Ethernet at the PC

11.2.2 Setting up the Ethernet at the PC (EtherCAT).

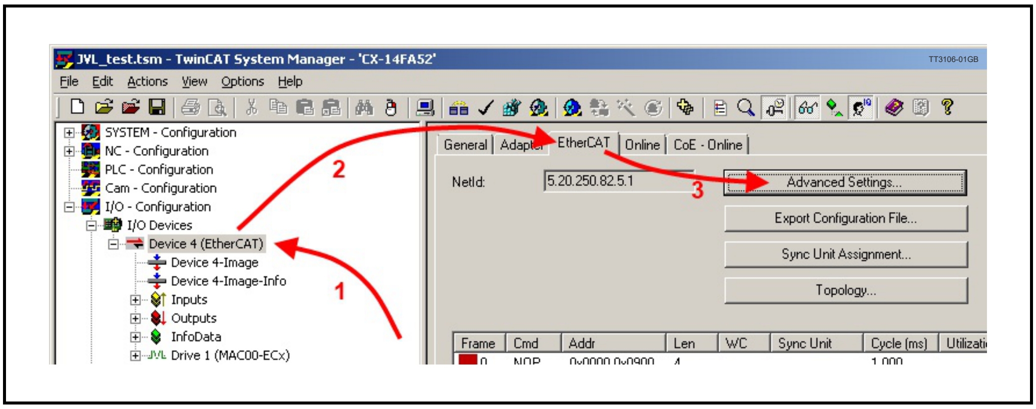
With EtherCAT it is not allowed to connect Mactalk directly to the JVL motor(s) via Ethernet. As this protocol is a master-slave protocol it is only allowed to connect Mactalk to the Master which then routes the Mactalk frames to the motor(s), and response frames back to Mactalk. This is done via a special protocol named EoE (Ethernet over EtherCAT), by which the normal Ethernet frames are send on the EtherCAT network in between the real-time frames. Please see the illustration below.



The IP addresses shown on the figure is just an example of a working configuration.

The EtherCAT master needs to be setup to act as a router between Mactalk and the slaves on the EtherCAT network. The EtherCAT master has to be setup first, to know what to setup in the Mactalk-PC and in the JVL motor slaves. Below is shown the steps for configuring a Beckhoff TwinCAT master to route normal Ethernet frames to the JVL slaves on the EtherCAT network.

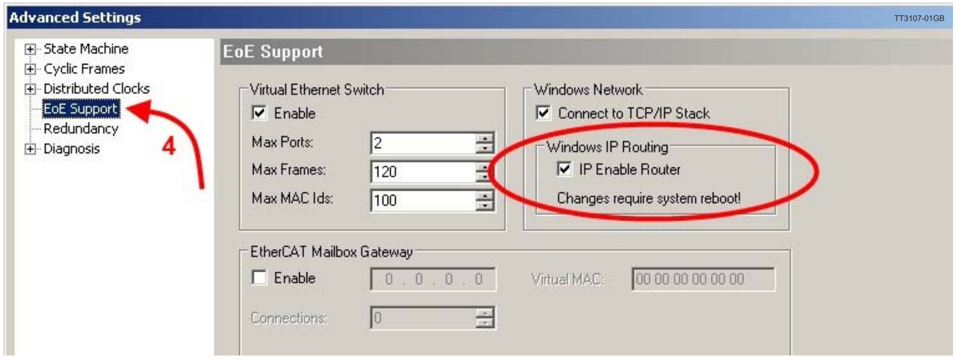
Step 1-3.
Select the I/O Device, then select the “EtherCAT” tab and press the “Advanced Settings” button.



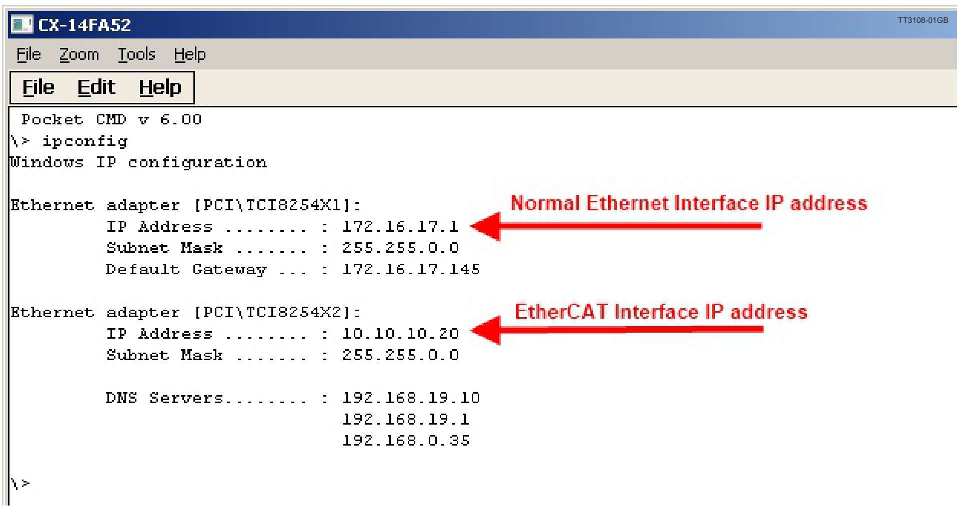
11.2 Setting up the Ethernet at the PC

Then it is possible to setup the JVL Motors for working correctly with EoE. Please follow the steps shown below.

Step 4.
Check the “IP Enable Router” in the EoE support. If not already checked, then a reboot of the EtherCAT master is necessary.

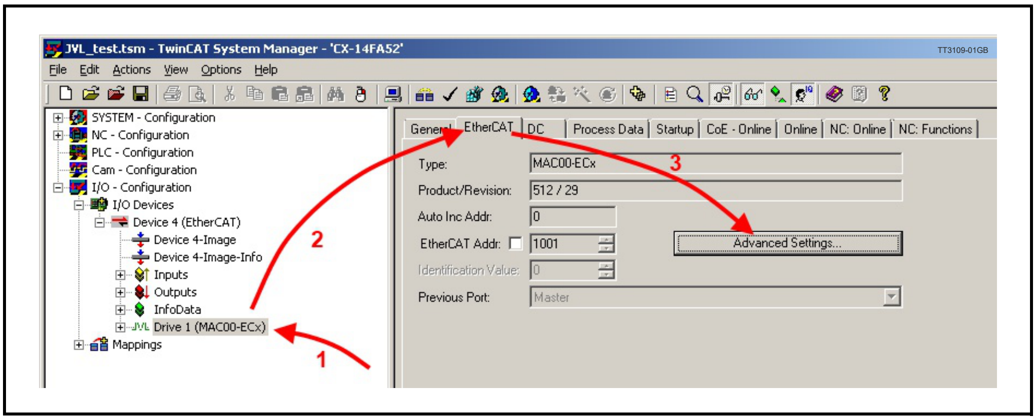


Step 5.
Find the IP addresses of the interfaces in the EtherCAT master. If necessary then change them to something suitable (beyond the scope of this manual).

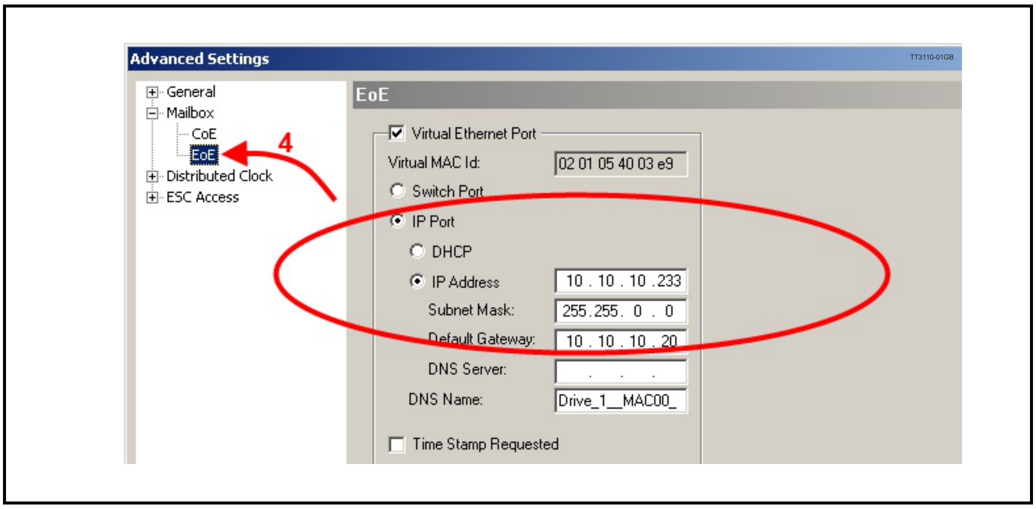


11.2 Setting up the Ethernet at the PC

Step 1-3.
Select the JVL Drive, then select the "EtherCAT" tab and press the "Advanced Settings" button.



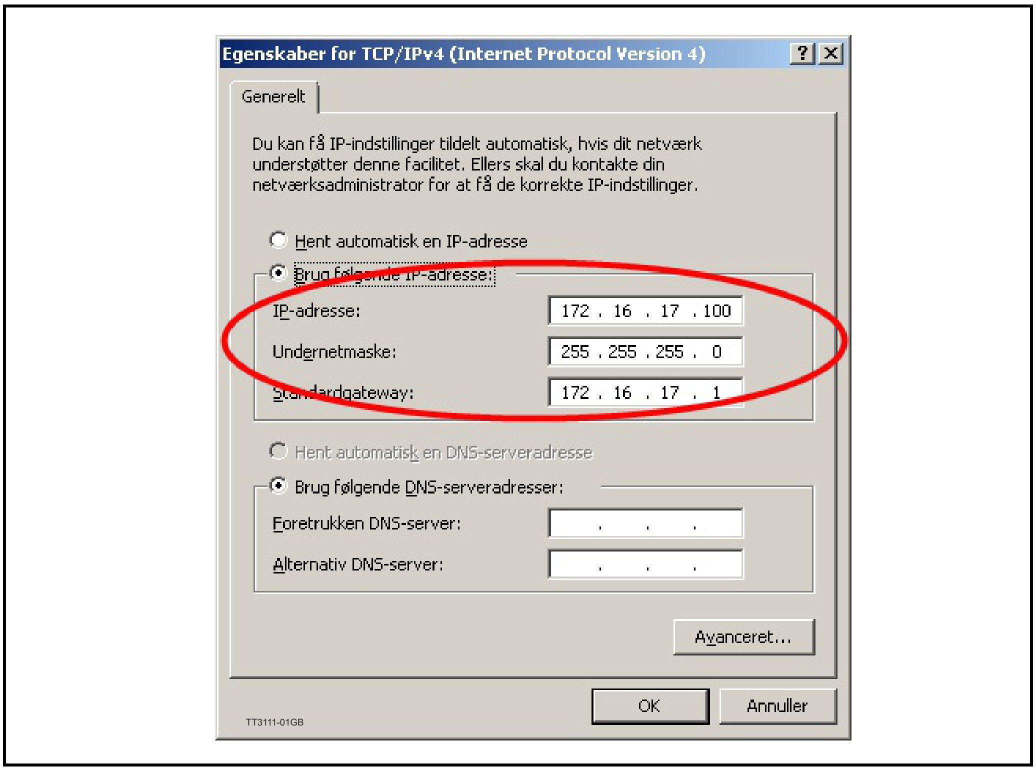
Step 4.
In the "Advanced Settings" window expand the "Mailbox" item and press the "EoE" item. Check that "IP Port" and "IP Address" checkboxes is checked. If not then check them. Make sure that the IP address, subnet mask and the default Gateway is suitable (IP address have to be same subnet as the EtherCAT interface on the EtherCAT master; the Default gateway have to be exactly the IP address of the EtherCAT interface on the EtherCAT master).



When there is a router – as is necessary in EtherCAT – in between Mactalk and the motor, the Mactalk PC has to be setup so it knows the IP address of the router to use.
So setup the Gateway address in the Mactalk PC to the IP address of the Ethernet interface of the EtherCAT master, and be sure that the Mactalk PC, IP address is in the same subnet as the EtherCAT master.

11.2 Setting up the Ethernet at the PC

Follow the steps in paragraph 9.2.1 - *Setting up the PC for EthernetIP, Profinet, ModbusTCP or Sercos., page 272*, but exchange the IP address with one in the same subnet as the Ethernet interface of the EtherCAT master, and set the default gateway to the IP address of the Ethernet interface of the EtherCAT master. With the settings used in this example the setup looks like this:



Now the connection is setup and should work. But it is advisable to test the connection stepwise first.
Open a command prompt on the Mactalk PC by pressing **start** and then **run**.
Then enter **cmd** and press **OK**.
In the command prompt first try to ping the Ethernet interface on the TwinCAT master by entering its IP address, for example:

- **ping 172.16.17.1 ← Replace IP with the TwinCAT master Ethernet interface address from your system.**

Next ping the EtherCAT interface on the EtherCAT master by entering the IP address of the master:

- **ping 10.10.10.20 ← Replace IP with the TwinCAT master EtherCAT interface address from your system.**

Last ping the JVL motor on the EtherCAT network:

- **ping 10.10.10.233 ← Replace IP address with the JVL motor IP address from your system.**

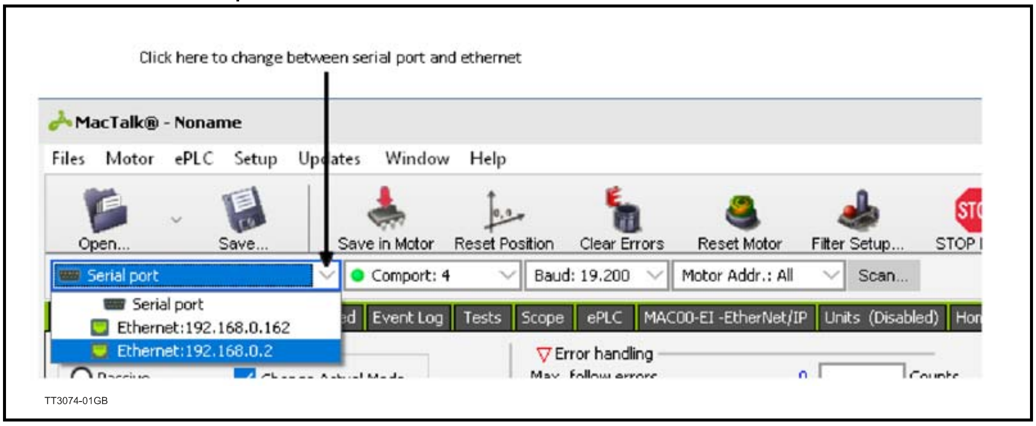
If one of the "pings" should fail, go back to the setup and check that every step is done correctly.

11.3 Setting up MacTalk for Ethernet

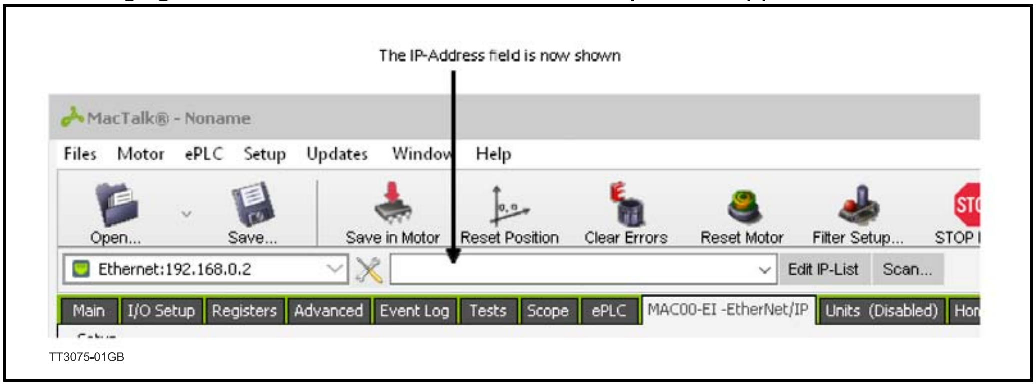
11.3.1 Setting up MacTalk for Ethernet communication

When MacTalk is opened the first time it is, by default configured for running serial RS232/RS485 connection. To change this please find the address box next to the “COM scan” in the upper tool bar and change it from “All” to “Eth”.

Step 1.
Select the Ethernet port used for communication to the PLC/motors as shown below.



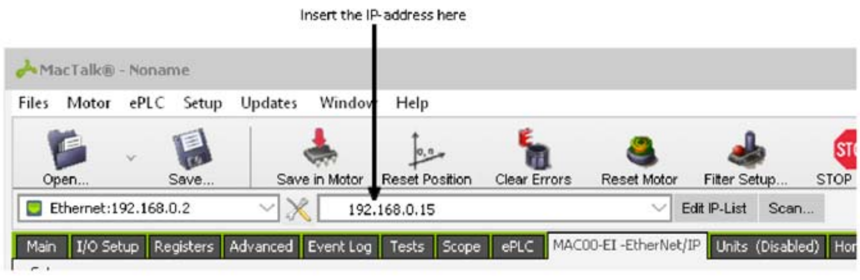
Step 2.
After changing the the Address box, the IP-address input field appears.



Step 3.
Now MacTalk is ready to connect to the motor and the next step is to enter the IP address of the motor to connect to.

11.3 Setting up MacTalk for Ethernet

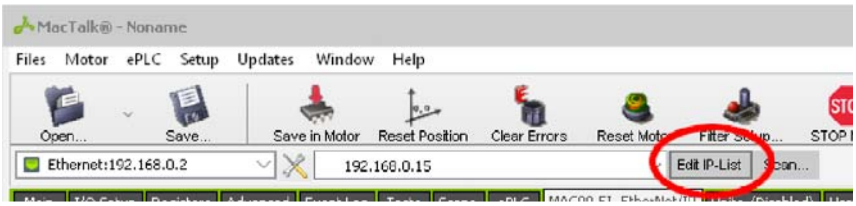
Step 4.
Lets assume that the motor with the IP address 192.168.0.58 is connected to the PC from where MacTalk is running or the same network that the PC is running, we enter the IP address.



TT3076-01GB

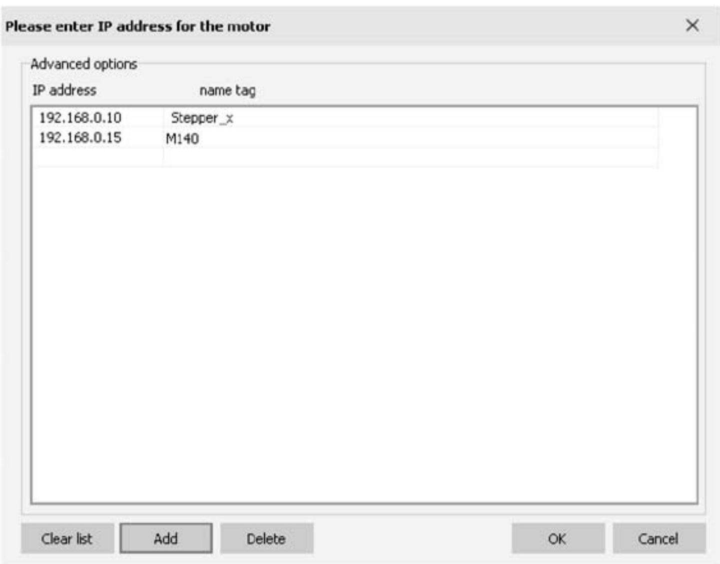
Step 5.
Since it is the first time the address is entered MacTalk offers the possibility to sign in the IP address and assign an alias name to this IP address which is stored and later be shown in the address field instead of remembering the IP address of the motor. This greatly helps managing multiple motors in a network instead of handling all the "anonymous" IP addresses.

Step 6.
The following dialog appears when a new address is entered.



TT3077-01GB

Step 7.
Pressing "Yes" will show the list of IP addresses and user composed names.

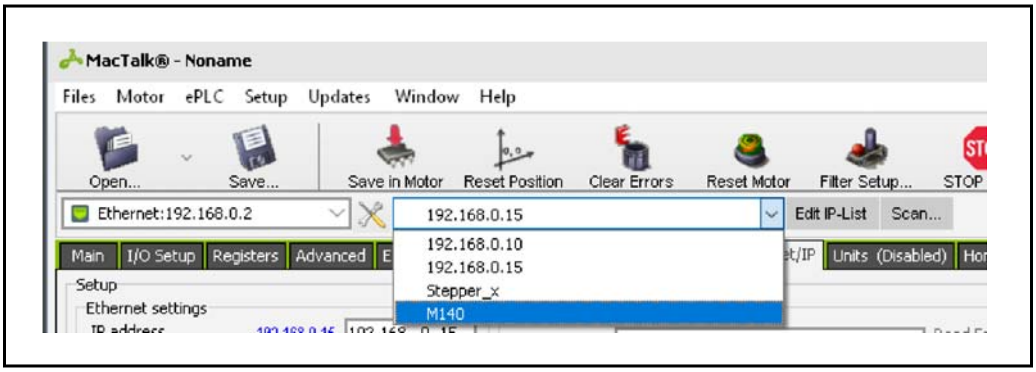


TT3078-01GB

11.3 Setting up MacTalk for Ethernet

Step 8.
In the list presented we have added a motor with the IP/address 192.168.0.56. This motor is stepper motor so we name it "Stepper_x" to be easy recognizable. We also have a MAC140 motor in the network, for this motor we have assigned the name M140.
The list is added to the address bar which automatically suggest the motor when we type in the first letters of the name. The motor can also be selected directly in the list. Please note that both the IP address and the name is added to the list and saved. The list is loaded automatically when MacTalk is started.

Step 9.
Add a name to the list in the field next to the IP address and press "Ok", Now the list is saved. The name entered can now be used to access the motor on the network. The complete list can be cleared by pressing "Clear list" or a single enty can be deleted by pressing "Delete".
When MacTalk is started this list is read and added to the address bar selections, so that either the name or the IP address can be selected.



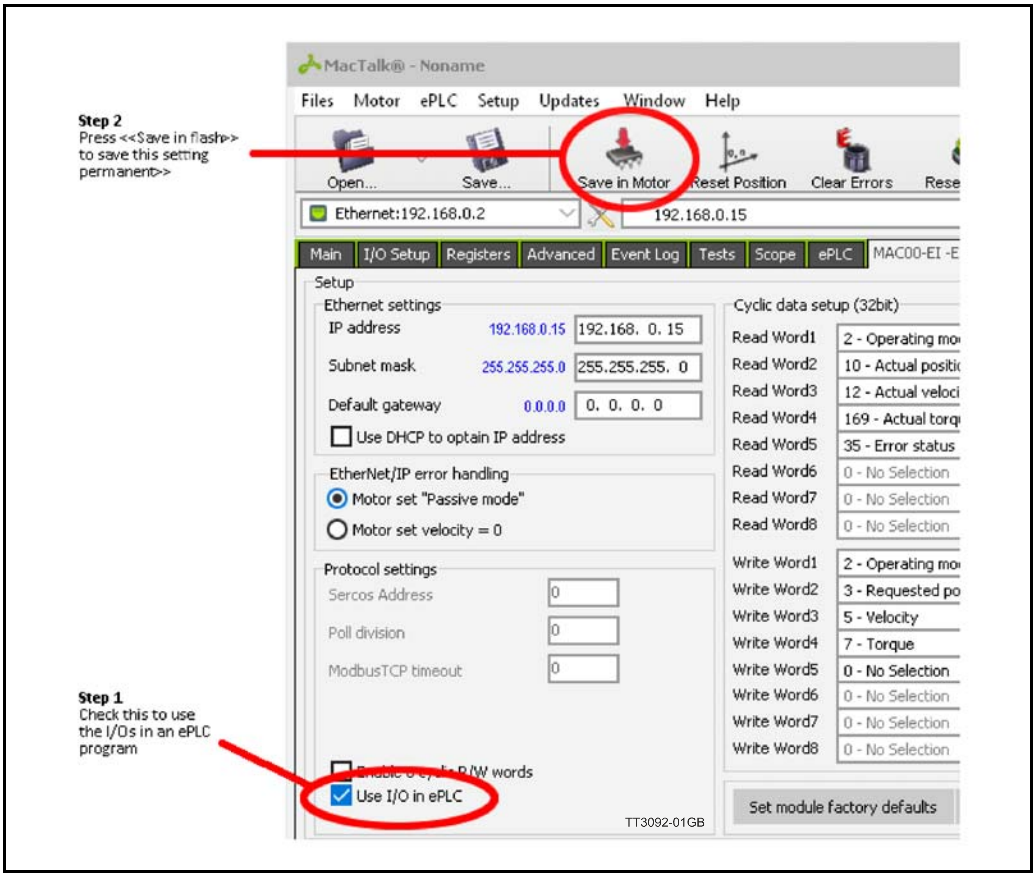
12 Examples common to all protocols

12.1 Using module I/O in embedded PLC

12.1.1 Using module I/O in embedded PLC

When using the module digital I/O's - which is opto isolated - it is necessary to use external supply to power the opto isolators to the pins IO- and O+ in the "I/O" connector. Please see chapter 2 *External signals available at the MAC00-Ex4 and Ex41., page 17* for further details.

A possible exception to this is when using MAC00-Ex41 containing extended I/O’s.
If using this module it is possible to activate the two dip switches inside the module. Then the internal power is also supplied to the opto isolators thereby eliminating the need for external supply.
The module I/O's (2 outputs and 4 inputs in MAC00-Ex41, 1 output and 1 input in MAC00-Ex4) are as default accessible from the Ethernet connected PLC by writing and reading to/from the module registers 7 and 47.
Please refer to chapter 8 for further details.
If it is required to use the digital module I/O's in embedded PLC (PLC - integrated sequential PLC), then this functionality has to be enabled first.
This is either done by manually manipulating the bits 7-9 in the module setup register (module reg. 6), from the Ethernet connected PLC. Please see chapter 8 for further details about the module registers.
Or it can be done in MacTalk in the Ethernet tab by checking the "Use I/O in eRxP" as shown below, and then pressing the "Apply and save button"



Then the module I/O's are visible from the eRxP in the motor. The module outputs O1, O2 can be activated by the bits 30 and 31 in the motor error/status register (motor reg. 35). The module inputs IN1-IN4 are read in motor register 210 in the bits 2-5.

12.1.2 Reset motor errors using cyclic I/O (JVL Profile)

Reset of motor errors are often done by issuing a command to the motor command register, but it is not recommended to place the motor command register in the cyclic list, as this will cause the Ethernet master to write the command at every net cycle causing the command to be executed several times until the command register is reset to zero.
The recommneded way to do this, is using the **Module command register.**
When a command is written to the module command register it is only sent to the motor **once**, for every change of value.

Procedure for setup:
In cyclic data setup in Mactalk in the Ethernet tab, select register "983040 - General command" (This is the module command register). Remember that there must not be any motor register after a module register in the cyclic list's. Press the "Apply and save" button. See picture below.

Procedure for using:
When a reset motor error is requested place the command 0x0100 00E1 = 16777441 dec in the write word where you have placed the register 983040. (If using a module software version with a build number below 1400 the command number 0x8000 00E1 = 2147483873 dec. must be used instead).
After the command is executed place the command 0 (zero) in the write word where you have placed the register 983040, in order to "rearm" for next command.

13 Appendix

13.1 Technical Data

13.1.1 EthernetIP for MAC or MIS - Technical specifications

Galvanic isolated, 100MBit, full duplex, 100Base-Tx, no termination necessary.
Network topology: Line, Star, Tree, Ring.
Supported Protocols:

- DHCP - Dynamic Host Configuration Protocol
- ACD - Address Conflict Detection
- DLR - Device Level Ring (ring topology on device level)

Max. 100 m cable between slaves.

Connectors (only applicable to MAC00-EIx)**:**

- “PWR” (power) M12 connector 5pin male
- “I/O” M12 connector 8pin female
- “L/A IN” and “L/A OUT” (Ethernet) M12 connector 4pin D-coded female.

Supply (only applicable to MAC00-EIx)**:**
Supply voltage (CV): 12-48VDC Nominal (absolute max. 50V)
Current rating (CV): typical 150mA, max. 250mA

User I/O (only applicable to MAC00-EIx)**:**
User inputs:
Input impedance: 4.7k
Input current @24V: 5.1mA
Digital output current (HW rev. Up to 1.2): 10mA
Digital output current (HW rev. from 1.3): 15mA

13.1.2 EtherCAT® for MAC or MIS - Technical specifications

Galvanic isolated, 100MBit, full duplex, 100Base-Tx, no termination necessary.
Network topology: Line, Star, Tree, Ring (line recommended)
Pass through delay: $<1\mu s$
Supported Protocols:

- SDO client and server side protocol
- CoE Emergency messages (CoE stack)

Max. 100 m cable between slaves.
Maximum number of slaves: 65535

Connectors (only applicable to MAC00-ECx)**:**

- “PWR” (power) M12 connector 5pin male
- “I/O” M12 connector 8pin female
- “L/A IN” and “L/A OUT” (Ethernet) M12 connector 4pin D-coded female.

Supply (only applicable to MAC00-ECx)**:**
Supply voltage (CV): 12-48VDC Nominal (absolute max. 50V)
Current rating @ 24V DC (CV): typical 150mA, max. 250mA

User I/O (only applicable to MAC00-ECx)**:**
User inputs:
Input impedance: 4.7k
Input current @24V: 5.1mA
Digital output current (HW rev. Up to 1.2): 10mA
Digital output current (HW rev. from 1.3): 15mA

13.1.3 Powerlink for MAC or MIS - Technical specifications

Galvanic isolated, 100MBit, half duplex, 100Base-Tx, no termination necessary.
Network topology: Line and tree possibly (line recommended)
Pass through delay: $<0.5\mu s$.
Acyclic data tramsfer: SDO Upload/Download
Functions: SDO over ASND and UDP
Ethernet Powerlink version: V2
Max. 100 m cable between slaves.
Maximum number of slaves (CN's) per segment: 239

Connectors (only applicable to MAC00-ELx)**:**
- "PWR" (power) M12 connector 5pin male
- "I/O" M12 connector 8pin female
- "L/A IN" and "L/A OUT" (Ethernet) M12 connector 4pin D-coded female.

Supply (only applicable to MAC00-ELx)**:**
Supply voltage (CV): 12-48VDC Nominal (absolute max. 50V)
Current rating @ 24V DC (CV): typical 150mA, max. 250mA

User I/O (only applicable to MAC00-ELx)**:**
Input impedance: 4.7k
Input current @24V: 5.1mA
Digital output current (HW rev. Up to 1.2): 10mA
Digital output current (HW rev. from 1.3): 15mA

13.1.4 PROFINET IO for MAC or MIS - Technical specifications

Galvanic isolated, 100MBit, full duplex, 100Base-Tx, no termination necessary.
Network topology: Line, ring, tree and star possibly.
Netload class I.
Forwarding delay: 3.25μs.
Netload Class I.
Minimum cycle time: 1ms (with MAC400-4500 and MIL/MIS motors).

Supported Protocols

- CL-RPC – Connection less Remote Procedure Call
- DCP – Discovery and Configuration Protocol
- LLDP – Link Layer Discovery Protocol
- RTA – Real time Acyclic Protocol
- RTC – Real time Cyclic Protocol, Class 1
- SNMP – Simple Network Management Protocol
- MRP – MRP Client is supported

Max. 100 m cable between slaves.

Connectors (only applicable to MAC00-EPx)**:**

- “PWR” (power) M12 connector 5pin male
- “I/O” M12 connector 8pin female
- “L/A IN” and “L/A OUT” (Ethernet) M12 connector 4pin D-coded female.

Supply (only applicable to MAC00-EPx)**:**
Supply voltage (CV): 12-48VDC Nominal (absolute max. 50V)
Current rating @ 24V DC (CV): typical 150mA, max. 250mA

User I/O (only applicable to MAC00-EPx)**:**
Digital input impedance: 4.7k
Digital input current @24V: 5.1mA
Digital output current (HW rev. Up to 1.2): 10mA
Digital output current (HW rev. from 1.3): 15mA

13.1.5 Modbus TCP/IP for MAC or MIS - Technical specifications

Galvanic isolated, 100MBit, full duplex, 100Base-Tx, no termination necessary.
Network topology: Line, ring, tree and star possibly.
Forwarding delay: 10-130µs.
Minimum cycle time: 2ms (with MAC400-4500).
Max. 100 m cable between slaves.
Protocol:

- Function codes supported: 3, 16, 23.
- Max. 124 modbus read registers per frame (= 62, 32bit registers).
- Max. 32 modbus write registers per frame (= 16, 32bit registers)
- 32bit support by 2x16bit registers. Only even no. of 16bit registers.
- I/O mode: Server, port 502.

Connectors (only applicable to MAC00-EMx)**:**

- “PWR” (power) M12 connector 5pin male
- “I/O” M12 connector 8pin female
- “L/A IN” and “L/A OUT” (Ethernet) M12 connector 4pin D-coded female.

Supply (only applicable to MAC00-EMx)**:**
Supply voltage (CV): 12-48VDC Nominal (absolute max. 50V)
Current rating @ 24V DC (CV): typical 150mA, max. 250mA

User I/O (only applicable to MAC00-EMx)**:**
Digital input impedance: 4.7k
Digital input current @24V: 5.1mA
Digital output current (HW rev. Up to 1.2): 10mA
Digital output current (HW rev. from 1.3): 15mA

13.1.6 Sercos® for MAC and MIS motors - Technical specifications

Galvanic isolated, 100MBit, full duplex, 100Base-Tx, no termination necessary.
Network topology: Line and ring possibly.
Forwarding delay: <1µs.
Minimum cycle time synchronized MAC400-MAC4500: 1ms
Minimum cycle time synchronized MISxxxES / SMCxxxES : 1ms
Minimum cycle time FSP IO profile / JVL profile non-synchronized: 500µs (with MAC400-MAC4500).
Max. jitter (synchronized): ±1µs.
Max. 100 m cable between slaves.

Maximum number of applicable Sercos® addresses	511
Supported Sercos® Communication Specification	Version1.1.2
SCP Sync support	yes
SCP_NRT PC support	yes
S/IP support	yes
Identification LED feature supported	yes
Hot plug	yes
Direct Cross communication	no

The sequences MDT-AT-UCC and MDT-UCC-AT (only with SCP_Sync active) are supported

(continued next page)

13.1 Technical Data

Connectors (only applicable to MAC00-ESx)**:**
- "PWR" (power) M12 connector 5pin male
- "I/O" M12 connector 8pin female
- "L/A IN" and "L/A OUT" (Ethernet) M12 connector 4pin D-coded female.

Supply (only applicable to MAC00-ESx)**:**
Supply voltage (CV): 12-48VDC Nominal (absolute max. 50V)
Current rating @ 24V DC (CV): typical 150mA, max. 250mA

User I/O (only applicable to MAC00-ESx)**:**
Digital input impedance: 4.7k
Digital input current @24V: 5.1mA
Digital output current (HW rev. Up to 1.2): 10mA
Digital output current (HW rev. from 1.3): 15mA

13.1.7 Sercos® Protocol features:

○ Supported **F**unction **S**pecific **P**rofiles:
- FSP_IO (*with JVL profile*)
  - 32 byte cyclic data in each direction (*max. 8 registers*)

○ FSP_DRIVE (*Only in MAC400-MAC4500 and MIS17x/23x/34x/43x*)
- 48 byte cyclic data in each direction (*max. 16 producer IDN's, and 5 consumer IDN's*).
- Position mode
- Velocity mode

○ Pack Profile / Sercos® Drive.
- Pack profile Basic A
- Pack profile Basic B
- Pack Profile Extended (*partly*)
  - Drive controlled homing
  - Probing Function. (*Only with MAC400 – MAC4500 motors*)

○ Supported **S**ercos **C**ommunication **P**rofile classes:
- SCP_VarCFG (*with FSP_Drive profile*)
- SCP_FixCFG (*with JVL profile under FSP_IO profile*)
- SCP_HP (=Hot plug)
- SCP_SysTime
- SCP_NRTPC
- SCP_SYNC (=synchronization of master and drive)
  - The sequences MDT-AT-UCC and MDT-UCC-AT are supported.
  - 1, 2 ms Sercos® cycle time with MAC400+ motors.
  - 1, 2, 4 ms Sercos® cycle time with MIS motors.
  - 1-32 ms motion cycle time with interpolation.
- SCP_Diag (=Diagnostic)
- SCP_SIP
- SCP_RTB (=Real Time Bits). (*Only with MAC400 – MAC4500 motors*)

○ Supported **G**eneric **D**evice **P**rofile classes:

- GDP_Basic
- GDP_PWD (=IDN Password protection)
- GDP_DiagT (=Diagnostic Trace)
- GDP_Id (=Identification)
- GDP_PrBBasic (=Probes/capture). (*Only with MAC400 – MAC4500 motors*)
  - 50µs resolution. ±2.5µs jitter.
  - Single measuring and position probing.
- GDP_BKP (=IDN Backup)
- GDP_StM (=State Machine)

13.2 Motor registers MAC050 - 141

13.2.1 Register list for MAC050, 095, 140 and 141.

The following list is only valid for the MAC50, MAC95, MAC140 and MAC141 motors.

Please notice: At the Ethernet modules all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range / Default	Size / Access	Unit	Description
0	N/A	N/A	N/A	N/A	N/A	Dummy register, do not use
1	PROG_VERSION	Displayed on bottom right status line.				Firmware version number.
2	MODE_REG	Startup mode / Change actual mode				The actual operating mode of the drive. In general, the motor will either be passive, attempt to reach a certain position, attempt to maintain a constant velocity or attempt to produce a constant torque. The various modes define the main type of operation as well as what determines the setpoint for that operation. The special cases 256..258 are used to perform a few special operations on the entire set of registers. Supported values are: 0 : **Passive mode**. The axis is not controlled by the drive, and can easily be moved by hand or external mechanics. 1 : **Velocity mode**. The drive will attempt to run the motor at a constant velocity selected by Reg5, V_SOLL, without violating the maximum torque or acceleration. 2 : **Position mode**. The drive will at all times attempt to move the actual motor position to the position selected by Reg3, P_SOLL, without violating the maximum velocity, torque or acceleration. 3 : **Gear Position mode.** 4 : **Analogue torque mode.** 5 : **Analogue velocity mode.** 6 : **Analog Velocity Gear mode.** 7 : **Manual current mode.** 8 : **Step response test mode.** 9 : **Internal test mode.** 10 : **Brake mode.** 11 : **Stop mode.** 12 : **Torque based zero search mode.** 13 : **Forward/only zero search mode.** 14 : **Forward+backward zero search mode.** 15 : **Safe mode.** 16 : **Analogue velocity with deadband mode.** 17 : **Velocity limited Analog Torque mode.** 18 : **Analogue gear mode.** 19 : **Coil mode.** 20 : **Analogue bi-position mode.** 21 : **Analogue to position mode.** 22 : **Internal test mode.** 23 : **Internal test mode.** 24 : **Gear follow mode.** 25 : **IHOME mode.** 256 257 258
3	P_SOLL, 32-bit	Position	-67M - +67M	32 bit R / W		The target position that the drive will attempt reach in position related modes.
4	(high word of P-SOLL)	-	-			-

13.2 Motor registers MAC050 - 141

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range / Default	Size / Access	Unit	Description
5	V_SOLL	Max. Velocity				The maximum velocity the motor is allowed to use.
6	A_SOLL	Acceleration			Counts/ Sample	The maximum acceleration in counts/sample^2 the drive is allowed to use during normal operation. Also note Reg32, ACC_EMERG, used during emergency stops.
7	T_SOLL	Torque	0-1023		-	The maximum torque that the drive is allowed to use. The value 1023 corresponds to 300% of nominal load, and is the absolute maximum peak torque allowed. The value 341 gives 100% (nominal load).
8	P_FNC, 32-bit (Sometimes named P_SIM)				Counts	
9	(high word of P_FNC/P_SIM)					
10	P_IST, 32-bit	Actual position				The actual motor position measured by the internal encoder. Updated every 1.9ms. Note that this register is maintained incrementally, which means that the user can update it to offset the working range. When updating when the drive is not in Passive mode, P_IST and P_SOLL should be updated together in an atomic operation, using Reg163, P_NEW, or other special measures. Also note that the firmware will change this register after a zero search operation has completed.
11	(high word of P_IST)	-	-			-
12	V_IST					Actual velocity of the drive.
13	KVOUT	Load factor				Ratio of the total inertia driven by the drive to the inertia of the motors rotor itself.
14	GEARF1					Gear factor 1, Nominator
15	GEARF2					Gear factor 2, Denominator
16	I2T					Energy dissipated in the motor windings.
17	I2TLIM					Safety limit for I2T above. Motor will set an error bit if I2T gets above I2TLIMIT.
18	UIT					Energy dissipated in the internal power dump.
19	UITLIM					Limit for Reg18, UIT. Motor will set an error bit if UIT gets above UITLIM
20	FLWERR, 32-bit					A measure of how far the drive is from its ideal regulation goal. This value is calculated differently in the various modes, and can mean things like pulses from theoretical position or difference in actual velo city to V_SOLL. Contact JVL for more detailed information for specific modes.
21	(high word of FLWERR)					
22	FLWERRMAX, 32-bit					When Reg20, FLWERR, exceeds this limit, an error bit is set in Reg35, ERR_STAT, and the motor will stop if Reg22 is non-zero. Usually this value is set experimentally to detect situations where a movement is blocked or fails.
23	(high word of FLWERRMAX)					
24	FNCERR, 32-bit					Shows how much the motor is behind the ideal movement; precise operation depends on mode. When this accumulated value exceeds Reg26, FNCERRMAX, the FNC_ERR bit is set in Reg35, ERR_STAT and the motor will stop.

TT1521GB

13.2 Motor registers MAC050 - 141

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range / Default	Size / Access	Unit	Description
25	(high word of FNCERR)					
26	FNCERRMAX, 32-bit					
27	(hi-word of FNCERRMAX)					
28	MIN_P_IST, 32-bit					
29	(hi-word of MIN_P_IST)					
30	MAX_P_IST, 32-bit					
31	(hi-word of MAX_P_IST)					
32	ACC_EMERG					
33	INPOSWIN					
34	INPOSCNT					
35	ERR_STAT					**Bit 0, I2T_ERR** Too much energy dissipated in the motor windings. Set when Reg16, I2T, exceeds Reg17, I2TLIM **Bit 1, FLW_ERR** The actual position is too far behind the ideal position. Set when FLWERRMAX is non-zero, and FLWERR exceeds FLWERRMAX. **Bit 2, FNC_ERR** The value of Reg24, FNCERR, exceeded the value of Reg26, FNCERRMAX. **Bit 3, UIT_ERR** The value of Reg18, UIT, exceeded the value of Reg19, UITLIM. **Bit 4, IN_POS** For position-related modes: The actual position was detected to be inside the InPosition window (Reg33, INPOSWIN) at least the number of times defined in Reg34, INPOSCNT. For other modes: Depends on mode; for velocity related modes, this bit means AtVelocity; for other more special modes, this bit is calculated differently, ask JVL for details. **Bit 5, ACC_FLAG** The drive is currently accelerating (the velocity is increasing). **Bit 6, DEC_FLAG** The drive is currently decelerating (the velocity is decreasing). **Bit 7, PLIM_ERR** One of the software position limits was exceeded,, drive will go into stop mode, then passive mode automatically. **Bit 8, FRAME_ERR_TX** A framing error was detected during the last reception on the FastMac protocol. Continued next page

TT1522GB

13.2 Motor registers MAC050 - 141

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range / Default	Size / Access	Unit	Description
35	ERR_STAT (cont. from last page)					**Bit 9, RELPOSPSOLL** **Bit 10, RELPOSPFNC** These two bits determine what will happen when one of the eight general purpose position registers, P1-P8 is activated through either a FastMac command (including activating s register group), through writing to Reg43, P_REG_P,on changes in bi-position mode or during manual resynchronization. If both are zero, the P register gets copied to the target register(s). If Bit 9 is set, the value of Reg3, P_SOLL, is added to the target register(s) to make a relative movement. If Bit 10 is set, the value of Reg8, P_FNC, is added to the target register(s) to make a relative movement. **Bit 11, IX_ERR** The current in at least one of the motor windings was measured to be too high, possibly because of bad current loop filter settings. Values for the current filter have been overwritten with default values. Specifically registers 106 through 111, 127 and 128. **Bit 12, UV_ERR** The motor power supply voltage (Reg151, U_SUPPLY) was measured to be below the value in Reg152, U_MIN_SUP and the drive was configured to set an error bit in case of undervoltage. **Bit 13, UV_DETECT** The motor power supply voltage (Reg151, U_SUPPLY) was measured to be below1.25 times the value in Reg152. This is a warning bit, not an error. **Bit 14, DIS_P_LIM** When this bit is set (during zero search or by the user), the drive will disable its position limits so it can move also outside the position limit range. This bit is cleared automatically when the actual position gets inside the position range again. **Bit 15, SSI_ERROR**

TT1523GB

13.2 Motor registers MAC050 - 141

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range / Default	Size / Access	Unit	Description
36	CNTRL_BITS					Bit 0, USRINTF0 Bit 1, USRINTF1 Bit 2, PULSEDIR Bit 3, INPSIGN Bit 4, HICLK Bit 5, HALL_INT Bit 6, RECORDBIT Bit 7, REWINDBIT Bit 8, RECINNERBIT Bit 9, AUTO_RESYNC Bit 10, MAN_RESYNC Bit 11, INDEX_HOME Bit 12, REL_RESYNC Bit 13, HALL_C Bit 14, HALL_B Bit 15, HALL_A
37	STARTMODE					
38	P_HOME, 32-bit					
39	(hi-word of P_HOME)					
40	V_HOME					Velocity used during Zero Search/Homing
41	T_HOME					Negative => home on falling edge of AN_INP
42	HOMEMODE					Used by FastMac commands
43	P_REG_P					
44	V_REG_P					
45	A_REG_P					
46	T_REG_P					
47	L_REG_P					
48	Z_REG_P					
49	POS0 / P1, 32-bit					
50	(hi-word of P1)					
51	POS1 / P2, 32-bit					
52	(hi-word of P2)					
53	POS2 / P3, 32-bit					
54	(hi-word of P3)					
55	POS3 / P4, 32-bit					
56	(hi-word of P4)					
57	POS4 / P5, 32-bit					
58	(hi-word of P5)					
59	POS5 / P6, 32-bit					Bit 0, COIL_START_DIR Bit 1, COIL_POS_CMD Bit 2, COIL_PWR_CMD Bit 3, COIL_POS_ACCEPT Bit 4, COIL_PWR_FLASH
60	(hi-word of P6)					
61	POS6 / P7, 32-bit					
62	(hi-word of P7)					
63	POS7 / P8, 32-bit					
64	(hi-word of P8)					
65	VEL0 / V1					
66	VEL1 / V2					
67	VEL2 / V3					
68	VEL3 / V4					
69	VEL4 / V5					
70	VEL5 / V6					
71	VEL6 / V7					
72	VEL7 / V8					
73	ACC0 / A1					
74	ACC1 / A2					
75	ACC2 / A3					
76	ACC3 / A4					
77	TQ0 / T1					
78	TQ1 / T2					
79	TQ2 / T3					
80	TQ3 / T4					

TT1524GB

13.2 Motor registers MAC050 - 141

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range / Default	Size / Access	Unit	Description
81	LOAD0 / L1					
82	LOAD1 / L2					
83	LOAD2 / L3					
84	LOAD3 / L4					
85	ZERO0 / Z1					
86	ZERO1 / Z2					
87	ZERO2 / Z3					
88	ZERO3 / Z4					
89	KFF3					
90	KFF2					
91	KFF1					
92	KFF0					
93	KVFX4					
94	KVFX3					
95	KVFX2					
96	KVFX1					
97	KVFY3					
98	KVFY2					
99	KVFY1					
100	KVFY0					
101	GEARB					
102	KVB3					
103	KVB2					
104	KVB1					
105	KVB0					
106	KIFX2					
107	KIFX1					
108	KIFY1					
109	KIFY0					
110	KIB1					
111	KIB0					
112	SAMPLE1					
113	SAMPLE2					
114	SAMPLE3					
115	SAMPLE4					
116	REC_CNT					
117	FNC_OUT					
118	FF_OUT					
119	VB_OUT					
120	V_EXT					Velocity of external encoder (Pulse In) in counts per sample
121	VF_OUT					
122	ANINP					
123	ANINP_OFFSET					
124	ELDEGN_OFFSET					
125	ELDEGP_OFFSET					
126	PHASE_COMP					
127	AMPLITUDE					
128	MAN_I_NOM					
129	MAN_ALPHA					
130	UMEAS					
131	I_NOM					
132	PHI_SOLL					
133	IA_SOLL					
134	IB_SOLL					
135	IC_SOLL					
136	IX_SELECT					
137	IA_IST					
138	IB_IST					
139	IC_IST					
140	IA_OFFSET					
141	IB_OFFSET					
142	IC_OFFSET					

TT1525GB

13.2 Motor registers MAC050 - 141

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range / Default	Size / Access	Unit	Description
143	ELDEG_IST					
144	V_ELDEG					
145	UA_VAL					
146	UB_VAL					
147	UC_VAL					
148	KIA					
149	KIB					
150	KIC					
151	U_SUPPLY					
152	MIN_U_SUP					
153	MOTORTYPE					
154	SERIALNUMBER, 32-bit					
155	(hi-word of SERIALNUMBER)					
156	MYADDR					
157	HWVERSION					
158	CHECKSUM, 32-bit					
159	(hi-word of CHECKSUM)					
160	UV_HANDLE					Bit 0, SET_UV_ERR Bit 1, UV_GO_PASSIVE Bit 2, unused Bit 3, UV_VSOLL0
161	INV_OUTPUT					Bit 0, INV_INPOSOUT Bit 1, INV_ERROROUT Bit 2, INVROTDIR Bit 3, O1USERCTRL Bit 4, O2USERCTRL
162	INDEX_OFFSET					
163	P_NEW, 32-bit					
164	(hi-word of P_NEW)					
165	FILTERID, 32-bit					
166	(hi-word of FILTERID)					
167	HARDWARELIM					Bit 0, HW_PLIM_NEG Bit 1, HW_PLIM_POS Bit 2, HW_PLIM_IN1 Bit 3, HW_PLIM_IN2 Bit 4, HW_PLIM_IN3 Bit 5, HW_PLIM_IN4 Bit 6, HW_PLIM_IN5 Bit 7, HW_PLIM_IN6 Bit 8, HW_PLIM_ANINP
168	HOMING_DONE					Bit-0 set every time a zero search has completed. Not cleared by firmware, except after reset.

TT1526GB

13.2 Motor registers MAC050 - 141

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range / Default	Size / Access	Unit	Description
169	GROUP_ID					
170	GROUP_SEQ					
171	MONITOR_CMP					
172	MONITOR_REG1					
173	MONITOR_REG2					
174	MONITOR_ACT					
175	MONITOR_SRC					
176	MONITOR_DST					
177	MONITOR_SAV					
178	SSI_BITS1					Bit 0, SSI_ENABLE Bit 1, SSI_DIR Bit 2, SSI_POS_SYNC Bit 3, SSI_RESET Bit 4, SSI_NOCHECK Bit 15, SSI_ERROR_CNTL
179	OUTPUT_CTRL					Bit 0, OUTPUT_O1 Bit 1, OUTPUT_O2
180	SETUP_BITS					Bit 0, POWERSAVE_ENABLED
181	V_IST_MAX					
182	UART1_SETUP		0, 1, 2			Selects what protocol to run on the RS422 lines that can be used for Pulse In, Pulse Out or Serial Data. The selection in this register is used only if the lowest two bits in Reg36, CNTRL_BITS are set to Serial Data. Values of Reg182, UART1_SETUP: 0: Autodetect incoming 1 Megabit Modbus telegrams for a few seconds after startup. Stay in Modbus if any valid Modbus telegrams detected, else switch to 19200 baud FastMac and stay in Fastmac. 1: Run the FastMac protocol at 19200 baud from the beginning and stay in FastMac. 2-65535: Run 1 Megabit/s Modbus from the beginning and stay in Modbus.
183	STATUS_BITS					
184	MODE0 / M1					
185	MODE1 / M2					
186	MODE2 / M3					
187	MODE3 / M4					
188	HWI0, 32-bit					
189	(hi-word of HWI0)					
190	HWI1, 32-bit					
191	(hi-word of HWI1)					
192	HWI2, 32-bit					
193	(hi-word of HWI2)					
194	HWI3, 32-bit					
195	(hi-word of HWI3)					
196	HWI4, 32-bit					
197	(hi-word of HWI4)					

TT1527GB

13.2 Motor registers MAC050 - 141

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range / Default	Size / Access	Unit	Description
198	HWI5, 32-bit					
199	(hi-word of HWI5)					
200	HWI6, 32-bit					
201	(hi-word of HWI6)					
202	HWI7, 32-bit					
203	(hi-word of HWI7)					
204	-					Reserved for future purposes
205	-					Reserved for future purposes
206	-					Reserved for future purposes
207	-					Reserved for future purposes
208	-					Reserved for future purposes
209	-					Reserved for future purposes
210	-					Reserved for future purposes
211	COMMAND					
212	FIELDBUS_ADDR					
213	FIELDBUS_SPEED					
214	-					Reserved for future purposes
215	-					Reserved for future purposes
216	-					Reserved for future purposes
217	-					Reserved for future purposes
218	-					Reserved for future purposes
219	-					Reserved for future purposes
220	-					Reserved for future purposes
221	-					Reserved for future purposes
222	-					Reserved for future purposes
223	-					Reserved for future purposes
224	-					Reserved for future purposes
225	-					Reserved for future purposes
226	-					Reserved for future purposes
227	-					Reserved for future purposes
228	-					Reserved for future purposes
229	-					Reserved for future purposes
230	-					Reserved for future purposes
231	-					Reserved for future purposes
232	-					Reserved for future purposes
233	-					Reserved for future purposes
234	-					Reserved for future purposes
235	-					Reserved for future purposes
236	-					Reserved for future purposes
237	-					Reserved for future purposes
238	-					Reserved for future purposes
239	-					Reserved for future purposes
240	-					Reserved for future purposes
241	-					Reserved for future purposes
242	-					Reserved for future purposes
243	-					Reserved for future purposes
244	-					Reserved for future purposes
245	-					Reserved for future purposes
246	-					Reserved for future purposes
247	-					Reserved for future purposes
248	-					Reserved for future purposes
249	-					Reserved for future purposes
250	-					Reserved for future purposes
251	-					Reserved for future purposes
252	-					Reserved for future purposes
253	-					Reserved for future purposes
254	-					Reserved for future purposes

TT1528GB

13.3 Motor registers MAC400 - 4500

13.3.1 Register list for MAC400, 800, 1500 and 4500

The following list is only valid for the MAC400 to MAC4500 motors.

Please notice: At the Ethernet modules all registers is transmitted as 32 bit, some of them originally derive from 16 bit in the case of MAC050-141. In those situations it is necessary to interpret them as 16 bit to get the sign correct.

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range / Default	Size / Access	Unit	Description
0	N/A	N/A	N/A	N/A	N/A	Dummy register, do not use.
1	PROG_VERSION	Displayed on bottom right status line.	-	- / R	-	Firmware version
2	MODE_REG	Startup mode / Change actual mode	0..25, 256, 257, 258 / 0 (passive)	Word / RW	-	The actual operating mode of the drive. In general, the motor will either be passive, attempt to reach a certain position, attempt to maintain a constant velocity or attempt to produce a constant torque. The various modes define the main type of operation as well as what determines the setpoint for that operation. The special cases 256..258 are used to perform a few special operations on the entire set of registers. Supported values are: 0 = **Passive mode**. The axis is not controlled by the drive, and can easily be moved by hand or external mechanics. 1 = **Velocity mode**. The drive will attempt to run the motor at a constant velocity selected by Reg5, V_SOLL, without violating the maximum torque or acceleration. 2 = **Position mode**. The drive will at all times attempt to move the actual motor position to the position selected by Reg3, P_SOLL, without violating the maximum velocity, torque or acceleration. 3 = Gear Position mode. 4 = Analogue torque mode. 5 = Analogue velocity mode. 6 = Analog Velocity Gear mode. 7 = Manual current mode. 8 = Step response test mode. 9 = Internal test mode. 10 = Brake mode. 11 = Stop mode. 12 = Torque based zero search mode. 13 = Forward/only zero search mode. 14 = Forward+backward zero search mode. 15 = Safe mode. 16 = Analogue velocity with deadband mode. 17 = Velocity limited Analog Torque mode. 18 = Analogue gear mode. 19 = Coil mode. 20 = Analogue bi-position mode. 21 = Analogue to position mode. 22 = Internal test mode. 23 = Internal test mode. 24 = Gear follow mode. 25 = IHOME mode. 256: 257: 258:

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
3	P_SOLL	Max Velocity	± 2^31 / 0	Word / RW	Encoder counts	The target position that the drive will attempt to reach in position related modes.
4	P_NEW	*(not present)*	±2^31 / 0	Word / RW	Encoder counts	Used to update both P_IST and P_SOLL in a single atomic operation to prevent motor movements during the change. P_NEW holds either an absolute position or a relative position. After writing a value to P_NEW, update both bits 8 and 6 in Reg36, CNTRL_BITS. Bit 8, SYNCPOSREL, will select a relative position update when set or an absolute update when cleared. Setting bit 6, SYNCPOSMAN, executes the P_IST+P_SOLL update, that is, either both are set equal to P_NEW, or P_NEW is added to both, using signed addition. P_FUNC is updated accordingly. The undocumented FastMac commands 23 and 24 can also be used to set these bits and perform the same absolute and relative updates. This is useful for expanding the logical position range beyond +/- 2^31.
5	V_SOLL	Max Velocity	Na / 277(100RPM)	Word / RW	Cnt's/ sample	Desired velocity 1 RPM=2.77056 counts/sample. Example: To obtain 100 RPM, V_SOLL must be set to 277.
6	A_SOLL	Acceleration	na / 18 (5003RPM/s²)	Word / RW	Cnt's/ Sample²	The desired nominal acceleration. 1000 RPM/s = 3.598133 counts/Sample² Example: To obtain 100000 RPM/s, A_SOLL must be set to 360.
7	T_SOLL	Torque	0-1023 / 1023(300%)	Word / RW	-	The maximum torque that the drive is allowed to use. The value 1023 corresponds to 300% of nominal load, and is the absolute maximum peak torque allowed. The value 341 gives 100% (nominal load).
8	P_FNC			Word / RW	Encoder counts	
9	INDEX_OFFSET	*(not present)*		Word / RW	Encoder counts	Updated after a Zero Search to show at what single-turn encoder position the zero point was detected. This is used by MacTalk on the Test tab to show if the zero search resulted in a valid zero position.
10	P_IST	Actual Position	±2^31 / 0	Word / RW	Encoder counts	The actual motor position measured by the internal encoder. Updated every 1.3ms (or every 2.6 ms with Reg157, OUTLOPDIV=2) Note that this register is maintained incrementally, which means that the user can update it to offset the working range. When updating when the drive is not in Passive mode, P_IST and P_SOLL should be updated together in an atomic operation, using Reg4, P_NEW, or other special measures. Also note that the firmware will change this register after a zero search operation has completed.
11	V_IST_16	Actual Velocity	Na / 0	Word / R	Enc.cnt's/ Sample/16	V_IST (actual velocity) measured over 16 samples. Same unit as V_SOLL (register 5).
12	V_IST	*(not present)*	Na / 0	Word / R	Enc.cnt's/ Sample	Actual velocity. 1RPM=0.17316 counts/sample.
13	KVOUT	Load	Na / 65536(1.0)	Fixed16 / RW	-	Must be set to the ratio between the total inertia driven by the motor relative to the motors own rotor inertia. So for at motor shaft that is not mechanically connected to anything, this value should be 1.0. The load factor is perhaps the single most important value of the filter setup. Always try to set this right before experimenting with filter setups.

TT1501GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
14	GEARF1	Gear factor Input	Na / 2000	Word / RW	-	The nominator used to scale / gear pulses from an external encoder/source. Used in gear modes.
15	GEARF2	Gear Output	Na / 500	Word / RW	-	The denominator used to scale / gear pulses from an external encoder/source. Used in gear modes.
16	I2T	Motor Load (mean)	Na / 0	Word / R	-	The calculated power dissipated in the motor, and thus an approximated value for the rise in temperature inside the physical motor. See also I2TLIM (Reg 17). MacTalk value is calculated as **[%]=I2T/I2TLIM x 100**
17	I2TLIM	*(not present)*	Na / 100000	Word / R	-	The limit for the value of Reg16, I2T, where bit 0, I2T_ERR, in Reg35, ERR_STAT will be set and the motor will change into passive mode.
18	UIT	Regen-Rative Load	Na / 0	Word / R		The calculated power dissipated in the internal power dump/brake resistors, and thus a way to estimate their rise in temperature. See also UITLIM (Reg 19) MacTalk value is calculated as **[%]=UIT/UITLIM x 100**
19	UITLIM	*(not present)*	Na / 2322	Word / R		The limit for the value of Reg18, UIT, where bit 3, UIT_ERR, in Reg35, ERR_STAT will be set and the motor will change into passive mode.
20	FLWERR		Na / 0	Word / RW	Encoder counts	A measure of how far the drive is from its ideal regulation goal. This value is calculated differently in the various modes, and can mean things like 'pulses from theoretical position' or 'difference in actual velocity to V_SOLL'. Contact JVL for more detailed information for specific modes.
21	U_24V		Na / 0	Word / R		The internal control voltage measured.
22	FLWERRMAX		Na / 0	Word / RW	Encoder counts	When Reg20, FLWERR, exceeds this limit, bit 1, FLW_ERR, in Reg35, ERR_STAT, is set and the motor will stop if Reg22 is non-zero. Usually this value is set experimentally to detect situations where a movement is blocked or fails.
23	UV_HANDLE	- Set error bit - Go to passive - Set velocity to 0	Na / 0	Word / RW		Bits to determine what will happen when the main supply voltage to the motor is below the threshold for motor operation. Any combination of the following bits can be set. Bit 0: Set bit 9, UV_ERR, in Reg35, ERR_STAT. Bit 1: Perform a controlled stop, then go passive. Bit 2: Set V_SOLL to zero, do not go passive.
24	FNCERR	*(not present)*	Na / 0	Word / RW	Encoder counts	Shows how much the motor is behind the ideal movement; precise operation depends on mode. When this accumulated value exceeds Reg26, FNCERRMAX, the FNC_ERR bit is set in Reg35, ERR_STAT and the motor will stop.
25	P_IST_TURNTAB	*(not present)*	Na / 0	Word / R	-	Displays the actual position, like P_IST, but is offset by N times the rotary table working range so P_IST_TURNTAB is always between MIN_P_IST and MAX_P_IST. Used mainly with the Rotary table option.
26	FNCERRMAX	*(not present)*	Na / 0	Word / RW	Encoder counts	The limit used with Reg24, FNCERR.
27	TURNTAB_COUNT	*(not present)*	Na / 0	Word / RW	-	Holds a count of the number of times the value of Reg25, P_IST, wraps around one of its limits, MIN_P_IST or MAX_P_IST. Used only with the Rotary table option. Counts up or down depending on the direction of the wrap around.
28	MIN_P_IST	*(not present)*	Na / 0	Word / RW	Encoder counts	Used to define and enable the minimum software position limit, so the motor will stop (and enter passive mode) if the value of P_IST (the actual position) gets below this value. If MIN_P_IST is zero, the low position limit will not be enabled.
29	DEGC	*(not present)*	Na / 0	Word / R	-	The temperature measured inside the drive.

TT1502GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
30	MAX_P_IST	*(not present)*	Na / 0	Word / RW	Encoder counts	Used to define and enable the maximum software position limit, so the motor will stop (and enter passive mode) if the value of P_IST (the actual position) gets above this value. If MAX_P_IST is zero, the high position limit will not be enabled. In Rotary Table operation, this limit is used as the higher wrap-around position count
31	DEGCMAX	*(not present)*	Na / 690(84'C)	Word / R	-	The maximum value of Reg29, DEGC, before the motor will set the Temperature error bit in ERR_STAT and change into Passive mode. Same scaling as Reg29, DEGC.
32	ACC_EMERG	*(not present)*	Na / 0	Word / RW	-	Acceleration to use during emergency stops.
33	INPOSWIN	*(not present)*	Na / 100	Word / RW	Encoder counts	The value of this parameter depends on the operating mode. In all cases it helps to define when the motor is InPosition and thus will set the InPosition bit in the ERR_STAT regis ter. For normal Position related modes, the motor is considered to be in position when the actual position is less than INPOSWIN encoder counts away from its target position P_SOLL and have been detected to be so at least INPOSCNT times. For Velocity rel ated modes, the concept of InPosition will instead mean AtVelocity and work in a similar way that the actual velocity V_IST must have been measured INPOSCNT consecutive times to be within INPOSWIN counts/sample before the InPosition bit is set in Reg35, ERR_STAT.
34	INPOSCNT	*(not present)*	Na / 3	Word / RW	-	The number of consecutive times the In Position condition must have been met before the InPosition bit is set in ERR_STAT. See description above for INPOSWIN.
35	ERR_STAT	*(not present)*	Na / 0	Word / RW	-	Bit 0, I2T_ERR Set when the calculated thermal energy stored in the physical motor exceeds a limit. Condition is that Reg16, I2T gets larger than Reg17, I2TLIM. Bit 1, FLW_ERR Set if the follow error in Reg20, FLWERR, gets larger than Reg22, FLWERRMAX. Never set if Reg22, FLWERRMAX is zero. Bit 2, FNC_ERR Set if the function error in Reg24, FNCERR, get slarger than Reg26, FNCERRMAX. Never set if Reg26, FNCERRMAX is zero. Bit 3, UIT_ERR Set when the calculated energy/temperature in the internal brake resistor (power dump) get dangerousl high. Bit 4, IN_POS In Position mode, status of when/whether the motor position is inside the inposition window defined by RegReg33, INPOSWIN, for the number of samples defined in Reg34, INPOSCNT. In Velocity mode, this bit means rather 'At Velocity'. For other modes, like Torque modes, see the technical manual for details of how the inposition status is calculated/maintained. Bit 5, ACC_FLAG Set when the motor is accelerating, which means that the velocity changes from a higher value to a lower value over tah latest samples. Please note that, when the velocity is negative, this flag is set when the velocity changes from a more negative value to a less negative value (closer to zero). This may not be intuitive, but can be said to be mathematically correct, and is maintained for backwards compatibility reasons.

TT1503GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacRegIO Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
33	INPOSWIN	*(not present)*	Na / 100	Word / RW	Encoder counts	The value of this parameter depends on the operating mode. In all cases it helps to define when the motor is InPosition and thus will set the InPosition bit in the ERR_STAT register. For normal Position related modes, the motor is considered to be in position when the actual position is less than INPOSWIN encoder counts away from its target position P_SOLL and have been detected to be so at least INPOSCNT times. For Velocity related modes, the concept of InPosition will instead mean AtVelocity and work in a similar way that the actual velocity V_IST must have been measured INPOSCNT consecutive times to be within INPOSWIN counts/sample before the InPosition bit is set in Reg35, ERR_STAT.
34	INPOSCNT	*(not present)*	Na / 3	Word / RW	-	The number of consecutive times the In Position condition must have been met before the InPosition bit is set in ERR_STAT. See description above for INPOSWIN.
35	ERR_STAT (continued)	*(not present)*	Na / 0	Word / RW	-	Bit 0, I2T_ERR Set when the calculated thermal energy stored in the physical motor exceeds a limit. Condition is that Reg16, I2T gets larger than Reg17, I2TLIM. Bit 1, FLW_ERR Set if the follow error in Reg20, FLWERR, gets larger than Reg22, FLWERRMAX. Never set if Reg22, FLWERRMAX is zero. Bit 2, FNC_ERR Set if the function error in Reg24, FNCERR, get slarger than Reg26, FNCERRMAX. Never set if Reg26, FNCERRMAX is zero. Bit 3, UIT_ERR Set when the calculated energy/temperature in the internal brake resistor (power dump) get dangerousl high. Bit 4, IN_POS In Position mode, status of when/whether the motor position is inside the inposition window defined by RegReg33, INPOSWIN, for the number of samples defined in Reg34, INPOSCNT. In Velocity mode, this bit means rather 'At Velocity'. For other modes, like Torque modes, see the technical manual for details of how the inposition status is calculated/maintained. Bit 5, ACC_FLAG Set when the motor is accelerating, which means that the velocity changes from a higher value to a lower value over tah latest samples. Please note that, when the velocity is negative, this flag is set when the velocity changes from a more negative value to a less negative value (closer to zero). This may not be intuitive, but can be said to be mathematically correct, and is maintained for backwards compatibility reasons. Bit 6, DEC_FLAG Set when the motor is decelerating. Please see the description for ACC_FLAG above to understand the behaviour with negative velocity. Bit 7 PLIM_ERR Set when one of the software position limits in Reg28 and Reg30 have been exceeded. Note that there is an option to temporarily disable position limits to be able to be move inside the position limit range again.

TT1533-01GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacRegIO Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
35	ERR_STAT (continued)	*(not present)*	Na / 0	Word / RW	-	Bit 8, DEGC_ERR Temperature error; set when the value in Reg29, DEGC, exceeds the value in Reg31, DEGCMAX. Bit 9, UV_ERR Under voltage error; can be set when either the AC sense detects missing 230 or 120 V AC input, or when the DC bus voltage get below a threshold voltage of approximately 80 V DC. Note that this error will only get set if bit 0 in Reg23, UV_HANDLE, is set. The AC sense input is not used with MAC402. Bit 10, UV_DETECT Reflects the current status of missing AC input or low DC bus voltage, on the same condition as UV_ERR above. The bit will be cleared when the under voltage condition is no longer present for about one second. Note that there is a hysteresis built-in, so the DC bus voltage must be 1.25 times higher than the cut-off voltage threshold before the under voltage condition is cleared. Bit 11, OV_ERR Overvoltage error; gets set when the DC bus voltage is measured to be above 450 V for MAC400 and MAC800, or 1000 V DC for MAC1500 and MAC3000, or 63 V DC for MAC402. Overvoltage often comes when the motor is braking a large inertia and the internal or external power dump/brake resistor can no longer accumulate the energy. It can also come from connecting a too high AC or DC supply voltage, perhaps by connecting 230VAC to the 120 VAC input on MAC400. Bit 12, IPEAK_ERR A much too high current was measured in one or more of the motor phases. This can happen if the supply voltage is too low relative to the rotational speed. It can also happen in more extreme short-time overload conditions. This error requires a reset or power cycling of the drive since it cannot be cleared using the normal Clear Errors mechanisms. Bit 13, SPEED_ERR The velocity was measured to be higher than a limit for an average of 16 samples. This limit is normally 3600 RPM for MAC800, MAC1500 and MAC3000, and it is 4300 RPM for MAC400 and MAC402. There is an option in later firmware to increase the speed error limit to 4000 RPM on MAC800 for shorter movements. Bit 14, DIS_P_LIM This bit is not an error or status bit, but rather a command bit. When this is set, the motor will stay in an active mode and allow moves even though one of the software position limits defined by Reg28 and Reg30 is exceeded. The bit is automatically cleared once the position gets inside the valid position range. Bit 15, INDEX_ERR The bit is set if an encoder error is detected. This can be a real encoder hardware error or excessive electrical noise. The motor must be reset or power cycled to clear this error. Bit 16, OLDFILTERR This bit is no longer used. It could detect if a previous version of the velocity filters was found, or loaded, that did not work with newer firmware. Bit 17, U24V_ERR This error bit get set if the control voltage, normally at 24VDC, is measured to be below 12 V (9V on MAC402). The motor must be reset or power cycled to clear this error. It is considered unsafe to continue, because the electronics may not work correctly, and may have lost data.

T1534-01GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
36	CNTRL_BITS	*(not present)*	Na / 32	Word / RW	-	Bit 0, RECORDBIT Set by the user to start or continue the sampling of register values, using the Classic scope system. This bit will clear itself when the sample buffer has been filled. Bit 1, REWINDBIT If set, the index into the sample buffer will be zeroed and sampling will continue if in progress. This bit is typically set together with RECORD_BIT above. Bit 2, RECINNERBIT If set, the samplinG7scope system samples at 100 microseconds between samples instead of the normal 1.3milliseconds. Normally used only for internal JVL development and service purposes. Bit 3, RELPOSPSOLL Bit 4, RELPOSPFNC These two bits select what happens if one of the general-purpose position registers, P1 through P8 is 'activated' by a FastMac command. If one of these is set, this activates a relative movement rather than the absolute position move that happens if none of these bits are set. If RELPOSPSOLL is set alone, the value of the selected P1-P8 register is added to the target position register Reg3, P_SOLL. If RELPOSPFNC is set, the value of the selected P1-P8 is added to an internal variable that will generate the movement, leaving P_SOLL unchanged. This is used for 'endless relative' movements, since it will not cause any overflow of the target position, but note that the actual position will wrap around at +/-2^31 (2,147,483,648 counts) without problems for the movement. Note that these bits also control the movements in Analogue Bi-position mode in similar ways. Bit 5, SYNCPOSAUTO If set when switching mode from Passive mode into an active mode, The follow error and the function error are zeroed, and the actual position is transferred to the P_FNC register, to avoid initial movement. Bit 6, SYNCPOSMAN Set to manually synchronize the position by copying the value of P_NEW, to P_IST, P_SOLL, and P_FNC, with proper scaling. In other words, set: P_IST = P_NEW, P_SOLL = P_NEW, P_FNC = (P_NEW + FLWERR)*16. See also bit 8 below. Note that this operation is performed as an atomic (unbreakable) operation, and is currently the only way to perform this perfect synchronization. Bit 7, MAN_NO_BRAKE Bit 8, SYNCPOSREL When set, modifies the manual synchronization performed by bit 6 above to use relative synchronization rather than absolute synchronization. In other words, set: P_IST = P_IST + P_NEW, P_SOLL = P_SOLL + P_NEW, P_FNC = P_FNC + (P_NEW + FLWERR)*16.

TT1504GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
39	HW_SETUP (continued from last page)	*(not present)*	Na / 9	Word / RW	-	Bit 13, PULSE_8000 If set, rescale the 8192 encoder pulses to 8000 for MAC800 compatibility and better Vel-filter performance Bits 14..15: reserved Bit 16, DIRCDWR Direction signal for the MultiFuncIo2 A channel (or both A and B?) Bit 17, SELINDEX Not used - prepared to select between encoder A or Index signal -> MultF. Bit 18, ALWAYS_COOL Bit 19, POSITION_CAPTURE_UP Used to enable SW position capture based on analogue input rising edge Bit 20, POSITION_CAPTURE_DN Used to enable SW position capture based on analogue input falling edge Bit 21, PULSE_8000 If set, rescale the 8192 encoder pulses to 8000 for MAC800 compatibility and better Vel-filter performance Bit 22, ENC_SCALING Reserved for freely selectable encoder scaling. Bit 23, SBUF_2048 Set to use a sample buffer length of 2048. Use 512 if not set (backwards compatible).
40	V_HOME	*(not present)*	Na / -138	Word / RW	-	Velocity to use during a zero search operation (Homing operation). After the operation has completed, the drive will go back to using the regular V_SOLL.
41	T_HOME	*(not present)*	Na / 341	Word / RW	-	Torque to use during a zero search operation (Homing operation). After the operation has completed, the drive will go back to using the regular T_SOLL.
42	HOME_MODE	*(not present)*	Na / 0	Word / RW	-	Defines if the motor should start a zero search immediately after start up, as well as the type of zero search to perform when a FastMac command is received. Bits 7..0 define the zero search mode the motor should start up in. If this value is zero, the motor will not perform a zero search at startup, but will start up in the mode selected by Reg37, START_MODE. See bits 15..8 below for an exception! Bits 15..8 define what mode the motor will set when it receives a FastMac command (96+16). NOTE that if all these bits are non-zero the motor will start up in passive mode instead of starting in START_MODE! Bit 16 is set after a zero search has completed, and can thus be used to test if the motor has performed a zero search at least once after +24V was last turned on. After a zero search has completed, the motor will always change into the mode defined by Reg 37, START_MODE (unless an error occurs that will stop the motor and set ERR_STAT bit(s)).
43	P_REG_P	*(not present)*	0-8 / 0	Word / RW	-	When set to 1..8, copies one of POS0..POS7 to P_SOLL, then resets to 0
44	V_REG_P	*(not present)*	0-8 / 0	Word / RW	-	When set to 1..8, copies one of VEL0..VEL7 to V_SOLL, then resets to 0
45	A_REG_P	*(not present)*	0-4 / 0	Word / RW	-	When set to 1..4, copies one of ACC0..ACC3 to A_SOLL, then resets to 0
46	T_REG_P	*(not present)*	0-4 / 0	Word / RW	-	When set to 1..4, copies one of TQ0..TQ3 to T_SOLL, then resets to 0
47	L_REG_P	*(not present)*	0-4 / 0	Word / RW	-	When set to 1..4, copies one of LOAD0..LOAD3 to KVOUT then resets to 0
48	Z_REG_P	*(not present)*	0-4 / 0	Word / RW	-	When set to 1..4, copies one of ZERO0..ZERO3 to INPOSWIN, then resets to 0

TT1505GE

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
49	POS0	Position1 (P1)	Na / 0	Word / RW	-	
50	CAPCOM0	*(not present)*	Na / 0	Word / RW	-	
51	POS1	Position2 (P2)	Na / 0	Word / RW	-	
52	CAPCOM1	*(not present)*	Na / 0	Word / RW	-	
53	POS2	Position3 (P3)	Na / 0	Word / RW	-	
54	CAPCOM2	*(not present)*	Na / 0	Word / RW	-	
55	POS3	Position4 (P4)	Na / 0	Word / RW	-	
56	CAPCOM3	*(not present)*	Na / 0	Word / RW	-	
57	POS4	Position5 (P5)	Na / 0	Word / RW	-	
58	CAPCOM4	*(not present)*	Na / 0	Word / RW	-	
59	POS5	Position6 (P6)	Na / 0	Word / RW	-	
60	CAPCOM5	*(not present)*	Na / 0	Word / RW	-	
61	POS6	Position7 (P7)	Na / 0	Word / RW	-	
62	CAPCOM6	*(not present)*	Na / 0	Word / RW	-	
63	POS7	Position8 (P8)	Na / 0	Word / RW	-	
64	CAPCOM7	*(not present)*	Na / 0	Word / RW	-	
65	VEL0	Velocity 1 (V1)	Na / 277(100RPM)	Word / RW	-	Velocity register V1. Used with the fastmac protocol or by the MAC00-R1/3/4 nanoPLC module. See also V_SOLL (register 5) which have the same scaling.
66	VEL1	Velocity 2 (V2)	Na / 277(100RPM)	Word / RW	-	Velocity register V8 - see also register 65.
67	VEL2	Velocity 3 (V3)	Na / 277(100RPM)	Word / RW	-	Velocity register V8 - see also register 65.
68	VEL3	Velocity 4 (V4)	Na / 277(100RPM)	Word / RW	-	Velocity register V8 - see also register 65.
69	VEL4	Velocity 5 (V5)	Na / 277(100RPM)	Word / RW	-	Velocity register V8 - see also register 65.
70	VEL5	Velocity 6 (V6)	Na / 277(100RPM)	Word / RW	-	Velocity register V8 - see also register 65.
71	VEL6	Velocity 7 (V7)	Na / 277(100RPM)	Word / RW	-	Velocity register V8 - see also register 65.
72	VEL7	Velocity 8 (V8)	Na / 277(100RPM)	Word / RW	-	Velocity register V8 - see also register 65.

TT1506GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
73	ACC0	*(not present)*	Na / 18(5003RPM/s²)	Word / RW	Enc.cnt's Per sample²	
74	ACC1	*(not present)*	Na / 18(5003RPM/s²)	Word / RW	Enc.cnt's Per sample²	
75	ACC2	*(not present)*	Na / 18(5003RPM/s²)	Word / RW	Enc.cnt's Per sample²	
76	ACC3	*(not present)*	Na / 18(5003RPM/s²)	Word / RW	Enc.cnt's Per sample²	
77	TQ0	Torque 1 (T1)	Na / 1023(300%)	Word / RW		Torque register T1. Used with the fastmac protocol or by the MAC00-R1/3/4 nanoPLC module. See also T_SOLL (register 7)
78	TQ1	Torque 2 (T2)	Na / 1023(300%)	Word / RW	-	Torque register T2 - see also register 77.
79	TQ2	Torque 3 (T3)	Na / 1023(300%)	Word / RW	-	Torque register T2 - see also register 77.
80	TQ3	Torque 4 (T4)	Na / 1023(300%)	Word / RW	-	Torque register T2 - see also register 77.
81	LOAD0	Load 1 (L1)	Na / 0	Word / RW	-	
82	LOAD1	Load 2 (L2)	Na / 0	Word / RW	-	
83	LOAD2	Load 3 (L3)	Na / 0	Word / RW	-	
84	LOAD3	Load 4 (L4)	Na / 0	Word / RW	-	
85	ZERO0	*(not present)*	Na / 0	Word / RW	-	
86	ZERO1	*(not present)*	Na / 0	Word / RW	-	
87	ZERO2	*(not present)*	Na / 0	Word / RW	-	
88	ZERO3	*(not present)*	Na / 0	Word / RW	-	
89	MODE0	*(not present)*	Na / 0	Word / RW	-	
90	MODE1	*(not present)*	Na / 0	Word / RW	-	
91	MODE2	*(not present)*	Na / 0	Word / RW	-	
92	MODE3	*(not present)*	Na / 0	Word / RW	-	
93	HWI0	*(not present)*	Na / 0	Word / RW	-	**H**ard**W**are **I**nputs Regs 93-104, HWI0-11, allow the digital inputs from Reg106 to control the values of other motor registers. The most common use is to copy one of two values to a target register. This can be used to switch between two velocities, positions or modes. For instance to switch between two target positions, set Reg49, POS0 to 1000 and Reg51, POS1 to 2000 and set the motor into position mode. Then P_SOLL can be set to receive either the value 1000 or 2000 depending on the voltage on the digital input (the Input State) The copying is executed every 1.3 ms. The digital inputs can thus be considered level-triggered rather than edge-triggered. (Continued next page)

TT1507GE

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
93	HWI0 (Continued from last page)	*(not present)*	Na / 0	Word / RW	-	Each of the HW!0-11 registers have the following bit fields: Bits [31:24]: Destination register used (only) when bits [3:0] equals 7. Bits [23:16]: Source register number 0..254 for DI=1 Bits [15:8]: Source register number 0..254 for DI=0 Bits [7:4]: Select digital input bit number in Reg106. Bits [3:0]: Target register selection. 0=None, 1=MODE_REG, 2=V_SOLL, 3=P_SOLL, 4=A_SOLL, 5=T_SOLL, 6=INPOSWIN, 7=Register number from bits [31:24]. When the value of bits [3:0] are one of 1..6, the two source registers are implicitly fixed to the corresponding group of register, and the value of bits [23:16] and bits [15:8] are used as an index into that group of registers. For instance if bits [3:0] equals 3, the values of bits [23:16] and bits [15:8] must be in the range 1..8 to select POS1 ..POS8 for source registers to copy into P_SOLL. When the value of bits [3:0] equals 7, the values of bits [23:16] and [15:8] hold the full register numbers in the range 1-254. For more advanced use, any of the source register or index values can be set to zero, which means DoNothing. This effectively means that in one of the Input States a source register will be copied to the target register, while in the other Input State no copying will happen so the target register will not be modified by the digital input. The 12 HWI functions are executed every 1.3 ms in the order from HWI0 to HWI11. NO other operations happen in between regardless of communications and other parallel operations. It is therefore safe to rely on stable register values and consistent digital input values during the execution of the 12 HWI functions. This implies that HWI function with higher numbers have higher priority because they are executed later, and that it is safe to change the same target register several times during the HWI evaluation. Note that each of the HWI function can use any of the digital inputs, and that more than one HWI function can use the same digital input. A typical HWI application is Jogging, where two pushbuttons connected to two separate digital inputs are used to move the motor position manually. This can be realized with a HWI setup like: HWI0 uses Digital Input 1: ON => MODE_REG=1 (velocity mode) OFF => MODE_REG=3 (gear mode) HWI1 also uses Digital Input 1: ON => V_SOLL=**+100**RPM OFF => V_SOLL = 3000 RPM HWI2 uses Digital Input 2: ON => MODE_REG=1 (velocity mode) OFF => MODE_REG=3 (gear mode) HWI3 also uses Digital Input 2: ON => V_SOLL=**-100**RPM OFF => V_SOLL = 3000 RPM This will keep the motor in Gear mode with a maximum velocity of 3000 RM when none of the pushbuttons are activated, and change to Velocity mode wit either +100 or -100 RPM as long as one of the pushbuttons are held active. In this setup Digital Input 2 will have higher priority than Digital Input 1, because it is evaluated later and overwrites V_SOLL in case both buttons are held down.

TT1508GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
94	HWI1	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
95	HWI2	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
96	HWI3	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
97	HWI4	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
98	HWI5	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
99	HWI6	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
100	HWI7	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
101	HWI8	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
102	HWI9	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
103	HWI10	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
104	HWI11	*(not present)*	Na / 0	Word / RW	-	See Reg93, HWI0, for description
105	MAC00_TYPE	*(not present)*	Na / 0	Word / RW	-	Identifies the Generation-2 module type autodetected at startup. 0 = No Gen2 module found, 1=MAC00-B41, 2=MAC00-P4 or MAC00-P5 found.
106	MAC00_1 / Digital Inputs	I/O management	Na / 0	Word / RW	-	The registers from 106 to 120 are used to support different interface modules with the Generation-2 connectors. The function of these registers will be different depending on which module is mounted in the motor. The Gen.2 module type is detected automatically by the motor at start up. Reg106, Digital inputs, is a bitmapped value where bits [15:8] show the status of hardware signals in the basic motor as described below, while bits [7:0] show the status of the digital inputs from the MAC00-B41 module. Be aware that bits [15:0] in Reg215, IO_POLARITY, can be set to invert the value of the corresponding bits [15:0] in this register. Bits [15:12] show the values of the four RS-422 signals. These are intended mostly for serial communications to some modules or to use Modbus RS485, but they can be used as digital inputs provided that the input voltage is kept within -7 to +12 volts. These are differential signals, so to use them as single-ended inputs, one of the differential lines must be kept at a constant voltage in between the high and low thresholds for the single-ended line. At the time of this writing, bits [15:12] are supported on MAC400, but not yet on MAC800. Bit 15: Multifunction 1, channel B Bit 14: Multifunction 1, channel A Bit 13: Multifunction 2, channel B Bit 12: Multifunction 2, channel A Bits [10:8] show the status of the analogue inputs ANINP2, ANINP1 and ANINP. Status will be high (logic 1) when the value of the analogue line is above 5.0 volts. This threshold can be adjusted by modifying the corresponding ANINPx_OFFSET registers. This way it is possible to use the analogue inputs as digital inputs with adjustable thresholds in the range -10V to +10V. Bit 10: ANINP2 (not signal conditioned) Bit 9: ANINP1 (not signal conditioned) Bit 8: ANINP (signal conditioned) To use ANINP3 (availab le on the MAC00-P4 and MAC00-P5 modules as analogue current loop 4-20 mA) use Reg222, IOSETUP to make ANINP reflect the (signal conditioned) value of this input, so the digital status will be shown in Bit 8. To use ANINP2 as a signal conditioned input, use a similar trick so IOSETUP is set to make ANINP reflect the signal conditioned value of ANINP2 in bit 8. Bits 6, 7, and 11 are unused.

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
107	MAC00_2	*(not present)*	Na / 0	Word / RW	-	Shows various status bits for the currently mounted Gen2 module. For the MAC00-B41: Bit 0: Digital Output overload. This shows the staus of the output driver chip that controls the six digital outputs. The overload status can be set if either an overcurrent condition or a too high temperature is detected. This status bit is cleared when these conditions are no longer present. Bit 1: CVO voltage detected. This bit reflects if the voltage at the CVO terminal is above a hardwired default value. CVO is the supply voltage for the digital outputs.
108	MAC00_3	*(not present)*	Na / 0	Word / RW	-	N/U
109	MAC00_4	*(not present)*	Na / 0	Word / RW	-	N/U
110	MAC00_5	*(not present)*	Na / 0	Word / RW	-	N/U
111	MAC00_6	*(not present)*	Na / 0	Word / RW	-	N/U
112	MAC00_7	*(not present)*	Na / 0	Word / RW	-	N/U
113	MAC00_8 / B41_DO / Digital outputs	I/O management	Na / 0	Word / RW	-	Bits [5:0] of this register controls the digital outputs O6..O1 on the MAC00-B41 module. Each bit that is set here will enable the corresponding PNP output. It is possible to overwrite these bits by using Registers 115-120, see below. Also Reg215, IO_POLARITY, will invert the value of these bits before there are written to the hardware.
114	MAC00_9 / B41_DOSTATUS	I/O management	Na / 0	Word / RW	-	Shows the status of each of the six digital outputs actually written to the hardware. This value will be Reg113, possibly modified by Regs115 - 120 and finally possibly having some bits inverted by Reg215.
115	MAC00_10 / B41_CONF0	*(not present)*	Na / 0	Word / RW	-	Controls **IO1** on MAC00-B41 (bit 0 in B41_DO). Each of the B41_CONF5..CONF0 registers can be used to modify the corresponding digital outputs by effectively overwriting bits [5:0] in Reg113, B41_DO. They can be set to replace the corresponding bit in B41_DO with any bit from any motor register in the range 1..254, typically status bits from Reg35, ERR_STAT, for instance bits INPOS or ANY_ERR. Bits [31:24]: reserved Bits [23:16]: Source register number, 1..254. Bits [15:5]: Reserved Bits [4:0]: Bit number in source register to use. Reg215, IO_POLARITY, will be applied after these registers to allow general inversion of each digital output bit.
116	MAC00_11 / B41_CONF1	*(not present)*	Na / 0	Word / RW	-	Controls **IO2** on MAC00-B41 (bit 1 in B41_DO). See Reg115, B41_CONF0 for description.
117	MAC00_12 / B41_CONF2	*(not present)*	Na / 0	Word / RW	-	Controls **IO3** on MAC00-B41 (bit 2 in B41_DO). See Reg115, B41_CONF0 for description.
118	MAC00_13 / B41_CONF3	*(not present)*	Na / 0	Word / RW	-	Controls **IO4** on MAC00-B41 (bit 3 in B41_DO). See Reg115, B41_CONF0 for description.
119	MAC00_14 / B41_CONF4	*(not present)*	Na / 0	Word / RW	-	Controls **IO5** on MAC00-B41 (bit 4 in B41_DO). See Reg115, B41_CONF0 for description.
120	MAC00_15 / B41_CONF5	*(not present)*	Na / 0	Word / RW	-	Controls **IO6** on MAC00-B41 (bit 5 in B41_DO). See Reg115, B41_CONF0 for description.

TT1510GE

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
121	KFF5	KFF5	Na / 0	Word / RW	-	Filter coefficients used by the velocity and position regulator loops. These values should be loaded only from MacTalk, and not modified by the user, since this can have dangerous effects.
122	KFF4	KFF4	Na / 0	Word / RW	-	
123	KFF3	KFF3	Na / 0	Word / RW	-	
124	KFF2	KFF2	Na / 0	Word / RW	-	
125	KFF1	KFF1	Na / 0	Word / RW	-	
126	KFF0	KFF0	Na / 0	Word / RW	-	
127	KVFX6	*(not present)*	Na / 0	Word / RW	-	
128	KVFX5	*(not present)*	Na / 0	Word / RW	-	
129	KVFX4	*(not present)*	Na / 0	Word / RW	-	
130	KVFX3	*(not present)*	Na / 0	Word / RW	-	
131	KVFX2	*(not present)*	Na / 0	Word / RW	-	
132	KVFX1	*(not present)*	Na / 0	Word / RW	-	
133	KVFY5	*(not present)*	Na / 0	Word / RW	-	
134	KVFY4	*(not present)*	Na / 0	Word / RW	-	
135	KVFY3	*(not present)*	Na / 0	Word / RW	-	
136	KVFY2	*(not present)*	Na / 0	Word / RW	-	
137	KVFY1	*(not present)*	Na / 0	Word / RW	-	
138	KVFY	*(not present)*	Na / 0	Word / RW	-	
139	KVB4	*(not present)*	Na / 0	Word / RW	-	
140	KVB3	*(not present)*	Na / 0	Word / RW	-	
141	KVB2	*(not present)*	Na / 0	Word / RW	-	
142	KVB1	*(not present)*	Na / 0	Word / RW	-	
143	KVB0	*(not present)*	Na / 0	Word / RW	-	
144	KIFX2	*(not present)*	Na / 0	Word / R	-	Filter coefficients used by the current loop for low-level control f the phase currents. These values are fixed and should not be modified by the user.
145	KIFX1	*(not present)*	Na / 0	Word / R	-	
146	KIFY1	*(not present)*	Na / 0	Word / R	-	
147	KIFY0	*(not present)*	Na / 0	Word / R	-	
148	KIB1	*(not present)*	Na / 0	Word / R	-	
149	KIB0	*(not present)*	Na / 0	Word / R	-	

TT1511GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
150	<reserved>	*(not present)*	-			
151	<reserved>	*(not present)*	-			
152	<reserved>	*(not present)*	-			
153	<reserved>	*(not present)*	-			
154	<reserved>	*(not present)*	-			
155	ID_RESERVED	*(not present)*	-			<reserved>
156	S_ORDER	*(not present)*	Na / 0	Word / RW	-	An S-profile can be used to modify/smooth the acceleration at the beginning and end of a change in velocity. This is useful to prevent overshoot. The value of zero disables the S-profile so the normal A_SOLL is used. Values 1..8 can be used to select a progressively smoother S-profile, with 8 being the smoothest (and slowest). The value of S_ORDER may not be changed unless the motor is in Passive mode (MODE_REG=0).
157	OUTLOOPDIV	*(not present)*	Na / 0	Word / RW	-	Divider value for the velocity loop. With the standard value of 1, the velocity loop is recalculated every 1.3 ms. With a value of 2, the loop is recalculated every 2.6 ms, which can give better performance for slow movements and/or large inertia. It is absolutely necessary to use a different set of filters in Regs121-142 when changing this value. To change this value from MacTalk, and gain access to the extended filters, open the Filter Setup window, then hold down both the Control and Shift keys and double-click on the text ‘More’ to the left of the ‘Stability’ slider (at the green end). After entering the correct password, Sample Frequency can be selected and MacTalk will use the appropriate filter set. Note that the units of all velocity-related register, measured in counts/sample will now be doubled, and all acceleration-related registers, measured in Counts/sample^2, will be four times larger.

TT1512GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
158	SAMPLE1	*(not present)*	Na / 0	Word / RW	-	SAMPLE1..4 controls the scope/sample function. Register number, bit field and min/max/average sample type for the first value in each sample.
159	SAMPLE2	*(not present)*	Na / 0	Word / RW	-	Register number, bit field and min/max/average sample type for the second value in each sample.
160	SAMPLE3	*(not present)*	Na / 0	Word / RW	-	Register number, bit field and min/max/average sample type for the third value in each sample.
161	SAMPLE4	*(not present)*	Na / 0	Word / RW	-	Register number, bit field and min/max/average sample type for the fourth value in each sample.
162	REC_CNT	*(not present)*	0-511 or 0..2047 / 0	Word / RW	-	Index into the sample buffer used for scope functionality. The length of the sample buffer, and thus the range of this parameter if determined by bit 23, SBUF_2048, in Reg39, HW_SETUP. See document/section “YY” for further information on the sample system.
163	V_EXT	*(not present)*	Na / 0	Word / R	-	Unscaled/Raw velocity of external encoder input in pulses per 1.3ms.
164	GV_EXT	*(not present)*	Na / 0	Word / R	-	Velocity of external encoder input V_EXT, after being scaled by the ratio GEARF1/GEARF2
165	G_FNC	*(not present)*	Na / 0	Word / R	-	
166	FNC_OUT	*(not present)*	Na / 0	Word / R	-	
167	FF_OUT	*(not present)*	Na / 0	Word / R	-	
168	VB_OUT	*(not present)*	Na / 0	Word / R	-	
169	VF_OUT	Actual torque	Na / 0	Word / RW	-	
170	ANINP	*(not present)*	Na / 0	Word / RW	-	
171	ANINP_OFFSET	*(not present)*	Na / 0	Word / RW	-	
172	ELDEG_OFFSET	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
173	PHASE_COMP	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
174	AMPLITUDE	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
175	MAN_I_NOM	*(not present)*	Na / 0	Word / RW	-	<used with motor current loop>
176	MAN_ALPHA	*(not present)*	Na / 0	Word / RW	-	<used with motor current loop>
177	UMEAS	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
178	I_NOM	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
179	PHI_SOLL	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
180	IA_SOLL	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
181	IB_SOLL	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
182	IC_SOLL	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>

TT1513GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
183	IA_IST	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
184	IB_IST	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
185	IC_IST	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
186	IA_OFFSET	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
187	IB_OFFSET	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
188	KIA	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
189	KIB	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
190	ELDEG_IST	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
191	V_ELDEG	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
192	UA_VAL	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
193	UB_VAL	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
194	UC_VAL	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
195	EMK_A	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
196	EMK_B	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
197	EMK_C	*(not present)*	Na / 0	Word / R	-	<used with motor current loop>
198	U_BUS	Bus voltage	Na / 0	Word / R	-	The actual voltage of the internal DC bus, updated every 100 us. One count corresponds to ~0.888V.
199	U_BUS_OFFSET	*(not present)*	-	Word / R	-	Factory offset used to calibrate the measurement of Reg198, U_BUS.
200	TC0_CV1	*(not present)*	Na / 0	Word / R	-	<used by JVL only to monitor internal timing´>
201	TC0_CV2	*(not present)*	Na / 0	Word / R	-	<used by JVL only to monitor internal timing´>

TT1514GB

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
202	MY_ADDR	*(not present)*	Na / 0	Word / RW	-	The motor address used for the MacTalk protocol. The motor will respond to telegrams with this address or the broadcast address 255. MY_ADDR can also be used for the Modbus protocol if selected in Reg213, UART1_SETUP: Further, MY_ADDR can be read and used by the fieldbus modules for CANopen, DeviceNet and Profibus to define their address on the fieldbus, if not selected by DIP-switches on the MAC00-xx module.
203	MOTOR_TYPE	*(not present)*	Na / 0	Word / R	-	Value read from factory flash memory to identify the type of motor: 12=MAC400, 13=MAC400B, 14=MAC800, 15=MAC800B.
204	SERIAL_NUMBER	*(not present)*	Na / 0	Word / R	-	Value read from factory flash memory to show the JVL serial number of the motor.
205	HW_VERSION	*(not present)*	Na / 0	Word / R	-	Bits [23:20]: Value read from factory flash memory to identify the Main version of the bootloader. Bits [19:16]: Value read from factory flash memory to identify the Minor version of the bootloader. Bits [7:4]: Value read from factory flash memory to identify the Main version of the PCB controller board hardware. Bits [3:0]: Value read from factory flash memory to identify the Minor version of the PCB controller board hardware. The remaining bits are reserved.
206	CHKSUM	*(not present)*	Na / 0	Word / R	-	Value read from factory flash memory to show the checksums of the firmware and the bootloader.
207	USEROUTVAL	*(not present)*	Na / 0	Word / RW	-	The values of bits [1:0] are output to the standard InPosition and ErrorOut hardware signals if the corresponding bits [9:8], USER_INPOS and USER_ERROR, in Reg39, HW_SETUP are set.
208	COMM_ERRS	*(not present)*	Na / 0	Word / RW	-	Counts the number of communication errors that have occurred on the MacTalk serial interface. Errors can be framing errors and protocol data errors.
209	INDEX_IST	*(not present)*	0..8191 or 0..7999	Word / R	-	Actual single-turn position of the internal encoder, valid for both incremental and absolute encoders.
210	HW_PLIM	*(not present)*	Na / 0	Word / RW	-	Hardware position limits – used by the MAC00-FSx module.
211	COMMAND_REG	*(not present)*	Na / 0	Word / RW	-	1=Reset, 2=Save to flash and reset, 128..255 = Execute FastMac commands.
212	UART0_SETUP	MacTalk Baudrate	Na / 0	Word / RW	-	0=9600, 1=19200, 2=38400, 3=57600, 4=115200, 5=230400 baud.
213	UART1_SETUP	Serial data	Na / 0	Word / RW	-	This register selects the type of protocol to use on the Serial Data interface. See section “XX”.
214	EXTENC_BITS	*(not present)*	Na / 0	Word / RW	-	Supports setup of signals used for label dispenser functionality with the MAC00-B41 module.
215	INPUT_LEVELS	*(not present)*	Na / 0	Word / RW	-	
216	ANINP1	*(not present)*	Na / 0	Word / RW	-	
217	ANINP1_OFFSET	*(not present)*	Na / 0	Word / RW	-	
218	ANINP2	*(not present)*	Na / 0	Word / RW	-	
219	ANINP2_OFFSET	*(not present)*	Na / 0	Word / RW	-	
220	ANINP3	*(not present)*	Na / 0	Word / RW	-	
221	ANINP3_OFFSET	*(not present)*	Na / 0	Word / RW	-	

TT1515GE

13.3 Motor registers MAC400 - 4500

Reg. Nr.	Firmware / MacReglo Name	MacTalk Name	Range/ Default	Size / Access	Unit	Description
222	IOSETUP	*(not present)*	Na / 0	Word / RW	-	Selects what hardware analogue input signal that goes to the main ANINP register and controls some filtering/signal conditioning.
223	ANOUT1	*(not present)*	Na / 0	Word / RW	-	The value written here by the user, or by the firmware, will be output to the 4-20 mA hardware output on the MAC00-P5/P4 modules.
224	ANOUT1_OFFSET	*(not present)*	Na / 0	Word / RW	-	Offset that is added to ANOUT1 before writing to hardware.
225	P_OFFSET	*(not present)*	Na / 0	Word / RW	-	Used to adjust the zero position for absolute multi-turn encoders.
226	P_MULTITURN	*(not present)*	Na / 0	Word / RW	-	The full multi-turn position read directly from the absolute encoder, if mounted.
227	AIFILT_MAXSLOPE	*(not present)*	Na / 0	Word / RW	-	
228	AIFILT_FILTFACT	*(not present)*	Na / 0	Word / RW	-	
229	P_QUICK	N/A	Na / 0	Word / RW	-	The actual position of the internal encoder. Much like P_IST, but updated every 100us. P_IST is updated only once every 1.3ms (or 2.6 ms for OUTLOOPDIV=2).
230	XREG_ADDR	*(not present)*	Na / 0	Word / RW	-	Address of extended registers, XREGs. A positive value will write the contents of Reg231, XREG_DATA, to that register. A negative value will cause the value of that XREG to be writen to XREG_DATA. After the reading or writing operation has completed, XREG_ADDR will be set to zero. The first NN XREGs are used for configuration of the switchboard for hardware signals that can be routed in several ways through the FPGA in MAC800 HW 1.8 and later or MAC400 HW1.? And later.
231	XREG_DATA	*(not present)*	Na / 0	Word / RW	-	Data to or from extended registers. See XREG_ADDR for description

TT1516GE

13.4 Motor registers MISxxx

13.4.1 Register list for MIS motors.

Please notice: At the Ethernet modules all registers is transmitted as 32 bit.

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
1	PROG_ VERSION	32bit	R	-	-	Major*16 + Minor + 16384 + 17*2^14	The firmware version. The Bit 14 is set to indicate that the type is a stepper motor controller, while bits [19:14] are set to the specific motor type, where 17 means SMC85xx.	"Status bar"
2	MODE_REG	32bit	R/W	0, 1, 2, 13, 14	0	-	Controls the operating mode of the motor. 0 : Passive 1 : Velocity mode 2 : Position mode 13 : Zero search type 1 14 : Zero search type 2 32: Cyclic Synchronous Position mode (Ethernet only)	Current Mode
3	P_SOLL	32bit	R/W	(-2^{31})-$(2^{31}$-1)	0	Steps	The desired position. When in position mode, the motor will move to this position. This value can be changed at any time.	Position
4	Reserved						(intended for 64-bit P_SOLL hi-word)	
5	V_SOLL	32bit	R/W	-300,000-300,000	10000	0.01 RPM	The maxium allowed velocity. When in velocity mode the motor will run constantly at this velocity. Specify a negative velocity to invert the direction. This value can be changed at any time. Example: The value 25000 selects 250 RPM	Max velocity
6	A_SOLL	32bit	R/W	1-500,000	1000	RPM/s	The acceleration/deceleration ramp to use. If this value is changed during at movement it will first be active when the motor stops or changes direction.	Acceleration
7	RUN_ CURRENT	32bit	R/W	0-1533	511	C: 5.87 mA B: 3.91 mA A: 1.96 mA	Current to use when the motor is running. The unit depends on the driver: C = 9 A, B = 6 A, A = 3 A.	Running Current
8	STANDBY_ TIME	32bit	R/W	1-65535	500	ms	Number of milliseconds before changing to standby current.	Standby Time
9	STANDBY_ CURRENT	32bit	R/W	0-1533	128	C: 5.87 mA B: 3.91 mA A: 1.96 mA	The standby current. The unit depends on the driver: C = 9 A, B = 6 A, A = 3 A.	Standby Current
10	P_IST	32bit	R/W	(-2^{31})-$(2^{31}$-1)	-	Steps	The actual position. This value can be changed at any time.	Actual position
11	Reserved						(intended for 64-bit P_IST hi-word)	
12	V_IST	32bit	R	-300,000 - 300,000	-	0.01 RPM	The current velocity.	Actual velocity
13	V_START	32bit	R/W	1-300,000	1000	0.01 RPM	The start velocity. The motor will start the acceleration at this velocity.	Start velocity
14	GEAR1	32bit	R/W	(-2^{31})-$(2^{31}$-1)	409600	Counts	The multiplier of the gear factor	Output
15	GEAR2	32bit	R/W	(-2^{31})-$(2^{31}$-1)	2048	Counts	The divider of the gear factor	Input
16	ENCODER_ POS	32bit	R/W	(-2^{31})-$(2^{31}$-1)	-	Steps	If the encoder option is installed, this show the position feedback from the encoder.	Encoder position
17	Reserved						(intended for 64-bit ENCODER_POS hi-word)	
18	INPUTS	32bit	R	-	-	Special	The current status of the digital inputs.	"Status bar"
19	OUTPUTS	32bit	R/W	-	0	Special	The current status of the digital outputs, can be written to change the outputs.	"Status bar"
20	FLWERR	32bit	R	(-2^{31})-$(2^{31}$-1)	-	Steps	When the encoder option is installed this shows encoder deviation from the calculated position (P_IST).	Follow error
21	Reserved						(intended for 64-bit FLWERR hi-word)	

13.4 Motor registers MISxxx

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
22	FLWERRMAX	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	Steps	The maximum allowed value in FLWERR before an error is triggered. If FLWERRMAX = 0, the error is disabled.	Error handling -> Follow error
23	Reserved						(intended for 64-bit FLWERRMAX hi-word)	
24	COMMAND	32bit	R/W	FastMac commands: 0-127 Other: 256-	0	-	Used to issue commands to the motor. 0-127 is the normal FastMac commands, where only a subset is implemented in SMC85/66.	Special command
25	STATUSBITS	32bit	R	-	-	Special	Status bits: Bit 0: Reserved Bit 1: AutoCorrection active Bit 2: In Physical Position Bit 3: At velocity Bit 4: In position Bit 5: Accelerating Bit 6: Decelerating Bit 7: Zero search done Bit 8: PassWord lock Bit 9: Magnetic encoder error Bits 10-13: Reserved Bit 14: Electromech. brake active (Int./Ext.) Bit 15: Closed loop lead/lag detected Bit 16: Closed loop activated Bit 17: Internal encoder calibrated (ready for closed loop) Bit 18: Standby current is used Bit 19: External memory ok Bit 20: Internal encoder ok Bit 21: Ethernet sync activated Bit 22: In target position Bit 23: STO channel A ok Bit 24: STO channel B ok Bit 25-31: Reserved	Run Status
26	TEMP	32bit	R		-	-2.27 – uses offset	Temperature measured inside the motor. See the detailed description for information on the value scaling.	Temperature
28	MIN_P_IST	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	Steps	Negative software position limit	Position limit min
29	Reserved						(intended for 64-bit MIN_P_IST hi-word)	
30	MAX_P_IST	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	Steps	Positive software position limit	Position limit max
31	Reserved						(intended for 64-bit MAX_P_IST hi-word)	
32	ACC_EMERG	32bit	R/W	1-500,000	10,000	RPM/s	Acceleration to use when performing an emergency stop when an error has occurred.	Error acceleration
33	IN_POSITION_WINDOW	32bit	R/W	$0-(2^{32}-1)$	20000	Steps	Selects how close the internal encoder position must be to P_SOLL to set the InPhysical-Position status bit and prevent further AutoCorrection.	In position window
34	IN_POSITION_COUNT	32bit	R/W	0-100	2	Counts	The number of times to attempt AutoCorrection. A value of zero disables AutoCorrection.	Max. number of retries

13.4 Motor registers MISxxx

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
35	ERR_BITS	32bit	R/W		0	Special	Error bits: Bit 0: General error (always set together with another error bit) Bit 1: Follow error Bit 2: Output driver Bit 3: Position Limit Bit 4: Low bus voltage Bit 5: Over voltage Bit 6: Temperature >90 °C Bit 7: Internal (Self diagnostics failed) Bit 8: Absolute multiturn encoder lost position Bit 9: Absolute multiturn encoder sensor counting Bit 10: No comm. to absolute multiturn encoder Bit 11: SSI encoder counting Bit 12: Closed loop Bit 13: External memory Bit 14: Absolute single turn encoder Bit 27: Safe Torque Off (STO)	Errors
36	WARN_BITS	32bit	R/W		0	Special	Warning bits: Bit 0: Positive limit active Bit 1: Negative limit active Bit 2: Positive limit has been active Bit 3: Negative limit has been active Bit 4: Low bus voltage Bit 5: Reserved Bit 6: Temperature >80 °C Bit 7: SSI encoder Bit 8: Driver overload	Warnings
37	STARTMODE	32bit	R/W	0, 1, 2, 3	0	-	The motor will change to this mode after power up. This is also the mode that is used after a zero search is completed. See MODE_REG for a list of possible modes.	Startup mode
38	P_HOME	32bit	R/W	(-2^{31})-$(2^{31}$-1)	0	Steps	The found zero point is offset with this value.	Zero search position
39	Reserved						(intended for 64-bit P_HOME hi-word)	
40	V_HOME	32bit	R/W	-300,000-300,000	-5000	0.01 RPM	The velocity to use during zero search. Set a negative velocity to search in the negative direction.	Zero search velocity
42	HOMEMODE	32bit	R/W	0,13,14	0	-	Select the zero search that should start on power up.	Zero search mode
43-45	Reserved	32bit	R/W	1-8	0		Planned - Not supported yet!	
46	AbsEncPos	32bit	R	0-409,500	0	Steps	The position last read from the internal magnetic encoder. This is the absolute single-turn position.	Abs. encoder position
47	EXTENCODER	32bit	R	(-2^{31})-$(2^{31}$-1)	0	Counts	The value from an external encoder, eg. SSI.	SSI Encoder value
48	FlexReg	32bit	R	-	0	-	A mix of 16 bits from different registers. The user can set this up.	
49-64	Pn	32bit	R/W	(-2^{31})-$(2^{31}$-1)	0	Steps	8 position registers (odd numbered registers)	Position n (Pn)

13.4 Motor registers MISxxx

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
65-72	Vn	32bit	R/W	0-300,000	10000	0.01 RPM	8 Velocity registers	Velocity n (Vn)
73-76	An	32bit	R/W	1-500,000	1000	RPM/s	4 Acceleration registers	Acceleration n (An)
77-80	Tn	32bit	R/W	0-1533	511	5.87 mA	4 Run current registers	Current n (Tn)
81-88	Analog Filtered	32bit	R	0-4095	0	1.221 mV	The voltage on inputs 1 to 8 after being filtered in firmware. See the AFZUP_xxx registers for filter parameters. 5V is equal to a value of 4095.	N/A
89-96	AnalogInput	32bit	R	0-4095	-	1.221 mV	The unfiltered voltage on inputs 1 to 8. 5V is equal to a value of 4095.	N/A
97	BUSVOL	32bit	R	0-4095	-	26.525 mV	Bus voltage	Bus voltage
98	MIN_BUSVOL	32bit	R/W	0-4095	565	26.525 mV	Trigger point for under voltage	Min bus voltage
99	ENCODER_TYPE	32bit	R	0-10	-	-	Internal encoder type 0: No encoder 1: H2 (Single turn encoder 10 bit) 2: H3 (Absolute multi turn encoder 10 bit) 3: H2 (Single turn encoder 12 bit) 4: H4 (Singleturn encoder 12 bit + absolute multi turn encoder.	"Tooltip on motor"
100	AFZUP_WriteBits	32bit	R/W	-	0	Special	Bits 0.7: Bit mask for which of the analog inputs that will use the current value of the ConfMin/Max, MaxSlope and Filter registers. Bit 15: Set when values have been copied and used.	N/A – handled on the Filter Setup screen.
101	AFZUP_ReadIndex	32bit	R/W	0, 1-8, 32768-32775	0	Special	Bits 0-7: Index (1-8) of the analog input whose ConfMin/Max, MaxSlope and filter values to load into the corresponding AFZUO_xxx registers (for read-back). Bit 15 gets set after the registers have been updated.	N/A – handled on the Filter Setup screen.
102	AFZUP_ConfMin	32bit	R/W	0-4094	0	1.221 mV	Minimum confidence limit for analog inputs.	Confidence Min
103	AFZUP_ConfMax	32bit	R/W	1-4095	4095	1.221 mV	Maximum confidence limit for analog inputs.	Confidence Max
104	AFZUP_MaxSlope	32bit	R/W	2-4095	4095	1.221 mV	Maximum slope limit for analog inputs.	Max Slope
105	AFZUP_Filter	32bit	R/W	1-64	64	64^{th} of new sample	Filter value for analog inputs.	Filter (on the Filter Setup screen)
106	FilterStatus	32bit	R	0-65535	0		Individual status bits for 50% of samples outside confidence limits (high 8 bits) and 50% of samples violated the slope limit. (low 8 bits)	N/A (shown graphically)
107	SSI_Setup1	32bit	R/W	-	-	Special	SSI setup bits: Bit 0-4: No. of data bits Bit 5-7: No. of samples Bit 8-15: SSI clk. frequency Bit 16-28: Max. sample deviation Bit 29-31: Read retries	SSI Encoder setup
110	SettlingTime	32bit	R/W	0-32676	0	ms	Number of milliseconds to wait after an AutoCorrection attempt before testing for the position being within the target window.	Settling time between retries

13.4 Motor registers MISxxx

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
111	SSI_Setup2	32bit	R/W	-	-	Special	SSI setup bits: Bit 0-7: Prepare time Bit 8: Gray to bin conversion Bit 9: Reserved Bit 10: Disable interrupts Bit 11-18: Wait time	SSI Encoder setup
112-115	SAMPLE1-4	32bit	R/W	-	0	-	Select what register(s) to sample – part of the sample/scope function.	N/A
116	REC_CNT	32bit	R/W	-	0	-	Number of samples to make – part of the scope/ sample function.	N/A
117	S_TIME	32bit	R/W	-	1	ms	Sampletime – part of the scope/sample function.	N/A
118	S_CONTROL	32bit	R/W	-	0	-	Controls the scope/sample system.	N/A
120	INDEX_ OFFSET	32bit	R	0-409600	-	Steps	The position of the zero sensor relative to the encoder index. This is set after a zero search where the index is used.	Tests tab
121	Modbus_Setup	32bit	R/W	-	0	Special	Modbus setup bits: Bit 0: Enabled Bit 1: Type Bit 2-3: Parity Bit 4: Data bits Bit 5: Stop bits	N/A
122	Zero_Search_BITS	32bit	R/W	-	0	Special	Bits to control Zero Search: Bit 0: Search for index. Bit 1: Change direction on limit. Bit 2: Search for opposite side of sensor. Bit 3: Reserved Bit 4: Ignore switch (Used for searching only for index). Bit 5: Disable the 60 s Zero Search time out.	Advanced –> Zero search
124	SETUP_BITS	32bit	R/W	-	0	Special	Bit 0: Invert motor direction. Bit 1: Don't start program after power up. Bit 2-3: External encoder input type Bit 5: Synchronize to encoder after passive Bit 6: In phys. Position update continuously Bit 10: Startup: Transfer single turn position to P_IST Bit 11: Startup: Transfer multi turn position to P_IST Bit 12: Startup: Keep External Encoder Bit 13: Startup: Keep SSI Value Bit 14: CANopen: Beckhoff mode Bit 16: External Encoder counting direction Bit 17: Disable position limit error Bit 19: Disable brake (int./ext.) temporarily Bit 20: Disable SSI encoder error Bit 21: Low bus voltage -> Error Bit 22: Low bus voltage -> Passive Bit 23: Low bus voltage -> 0 RPM Bit 24: Enable closed loop Bit 25: Enable closed loop current control Bit 28: Position limits without memory	0: Invert motor direction 1: Don't start program after power up 2-3: 0 = Disabled, 1 = Quadrature, 2 = Puls/ direction 17: No error if position limit is detected

13.4 Motor registers MISxxx

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
125	IOSETUP	32bit	R/W	-	0	Special	Bit 0-7 sets the I/O active level. Bit 8-15 enables the I/O as an output.	Inputs/Outputs
129	NL_MASK	32bit	R/W	-	0	IO Mask	Input mask for Negative limit input.	Dedicated inputs - Negative limit input
130	PL_MASK	32bit	R/W	-	0	IO Mask	Input mask for Positive limit input.	Dedicated inputs - Positive limit input
132	HOME_MASK	32bit	R/W	-	0	IO Mask	Input mask for home sensor input(s), each bit set select which I/O 1-8 to use.	Dedicated inputs - Home input
135	INPUT_FILTER_MASK	32bit	R/W	-	0	IO Mask	Input mask for the digital inputs with input filter. Bits set use the input filter time in register 136, bits clear use a fixed update time of 100 us.	IOx digital input filter enabled
136	INPUT_FILTER_CNT	32bit	R/W	-	5	ms	The number of milliseconds the filtered digital inputs must be stable before accepting a change.	Input filter time
137	INPOS_MASK	32bit	R/W	-	0	IO Mask	Output mask for In position output	Dedicated outputs - In position
138	ERROR_MASK	32bit	R/W	-	0	IO Mask	Output mask for error output.	Dedicated outputs - Error
139	ACCEPT_VOLTAGE	32-bit	R/W		2052	8.764 mV	The voltage that must be measured before the current status log is erased.	Acceptance voltage
140	ACCEPT_COUNT	32-bit	R/W		100	Counts	The number of times the ACCEPT_VOLTAGE must be measured before starting the processor	Acceptance count
141	SAVE_VOLTAGE	32-bit	R/W		1710	8.764 mV	The voltage that determines how low the CVI can be before shut down.	Save voltage
143	CVI_VOLT	32-bit-	R	-	-	8.764 mV	The measured control voltage	N/A
144	P_NEW	32bit	R/W	(-2^{31})-$(2^{31}$-1)	0	Counts	Used with FastMac commands 23 and 24 for changing both the actual and requested position in one operation either absolute or relative.	N/A
145	Reserved						(intended for 64-bit P_NEW hi-word)	
146	BAUD_RATE	32bit	R/W	0-5	1	-	The baud rate on the serial port. 0 : 9600 baud 1 : 19200 baud (default) 2 : 38400 baud 3 : 57600 baud 4 : 115200 baud 5 : 230400 baud 6 : 460800 baud 7 : 921600 baud	Baud rate
147	TX_DELAY	32bit	R/W	1-255	15	Bits	The time to wait before the response is transmitted. The unit corresponds to the time of one bit at the current baud rate.	Transmit delay
148	GROUP_ID	32bit	R/W	0-255		-	The group id of the motor – used for the GroupWrite telegram on the MacTalk protocol.	Group Id
149	GROUP_SEQ	32bit	R	0-255	-	-	The last received group write sequence – part of the MacTalk serial protocol.	N/A
150	MY_ADDR	32bit	R/W	0-254	254	-	The motor address. Used on the MacTalk serial protocol.	Motor address

13.4 Motor registers MISxxx

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
151	MOTORTYPE	32bit	R	80-254		-	The motor type. Examples: 80: SMC85, 81: MIS340, 82: MIS341, 83: MIS342 120: MIS17, 150: SMC66, 151: MIS230, 152: MIS231 250: MIL340	“Status bar”
152	SERIAL-NUMBER	32bit	R	-	-	-	The serial number of the motor.	“Status bar”
154	CHECKSUM_1	32bit	R	0-65535	-		Firmware checksum part 1	“Tooltip on motor”
155	CHECKSUM_2	32bit	R	0-65353	-		Firmware checksum part 2	“Tooltip on motor”
156	HARDWARE_REV	32bit	R	0-65535	-	Major*16 + Minor	The revision of the hardware	“Tooltip on motor”
157	MAX_VOLTAGE MAX_CURRENT	32bit	R	0-100 [VDC] 0-9000 [mARMS]	*	Volt	Bit 0-15: Max voltage on bus If the bus voltage exceeds this value, the motor will go in error. Bit 16-31: Full scale motor current in mARMS	“Tooltip on motor”
158	AVAIBLE_IO	32bit	R	-	-	IO Mask and max current from 1-1532.	Bit 0-15: Defines what IO that are available on the connector – programmed during manufacturing. Bit 16-31: The max current to the motor.	N/A
159	BOOTLOADER_VER	32bit	R	0-65535	-	Major*16 + Minor	The version of the boot loader	“Tooltip on motor”
160	NOTSAVED	32bit	R/W	0-65535	0	-	This register is not used internally, but will always be 0 after power-on. Please notice that MacTalk uses this register.	N/A
165	OPTIONS_ BITS	32bit	R	0-65535	-	-	This register contains information about what options that are available. Bit 0-7 defines the options available in the hardware (or licensed). Bit 8-15 defines the options available in the firmware. Bit 0,8 : CANopen fieldbus	“Tooltip on motor”
166	FBUS_NODEID	32bit	R/W	0-127	5	Node id	The node id on the CANopen fieldbus interface.	CANopen -> Node Id
167	FBUS_BAUD	32bit	R/W	0-8	2	-	The baudrate used on the CANopen fieldbus interface. 0 : 1000 kbit/s 2 : 500 kbit/s 3 : 250 kbit/s 4 : 125 kbit/s 5 : 100 kbit/s 6 : 50 kbit/s 7 : 20 kbit/s 8 : 10 kbit/s	CANopen -> Baud rate
168	ModuleType	32bit	R	0	0	-	Tells which type of module is connected to the internal 1Mbit/s Modbus channel. 0 = No module 0x34 = EthernetIP 0x35 = EtherCAT 0x36 = PowerLink 0x37 = Profinet 0x38 = Modbus/TCP 0x3A = Sercos III	Dedicated tab
170	EXT_ ENCODER	32bit	R/W	(-2^{31})-$(2^{31}$-1)	-	Counts	This register counts the external encoder.	External encoder
171	Reserved						(intended for 64-bit EXT_ENCODER hi-word)	

13.4 Motor registers MISxxx

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
172	EXT_ ENCODER_ VEL	32bit	R	$(-2^{31})-(2^{31}-1)$	-	Counts/16ms	This register is updated with the velocity of the external encoder input. The velocity is measured every 16ms.	External encoder Velocity
174	D_SOLL	32bit	R/W	1-500,000	1000	RPM/s	The deceleration ramp to use. If this value is changed during at movement it will first be active when the motor stops or changes direction. If 0, A_SOLL is used for deceleration.	Deceleration
175	Internal_Encoder_Setup	32bit	R/W	-	-	Special	Bit 0-1: Hysteresis (0, 0.17, 0.35, 0.70 deg) Bit 2-4: Resolution (16,15,14,13,12*,11,10*,9) Bit 5: Filter cutoff (16 kHz, 3 kHz) Bit 6: Filter time (0, 1.2 us) *Closed loop compatible	N/A
176	FW_BUILD	32bit	R	$0-(2^{32}-1)$	-	Counts	Current firmware build number.	"Status bar"
177	InTargetPositionTime	32bit	R/W	$0-(2^{32}-1)$	10	ms	Time the motor must stand still before InTargetPosition flag is set.	N/A
179	BRAKE	32bit	R/W	$0-(2^{32}-1)$	-	Special	Selects which one of the eight I/O pins to use for the external brake.	N/A

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
The following parameters are only available when the CanOpen option is installed and only used for DSP-402 **NOTE: DSP-402 is NOT supported yet!**								
180	ControlWord	32bit	R/W	0-65535	0	-	Object 6040 subindex 0	
181	StatusWord	32bit	R	0-65535	0	-	Object 6041 subindex 0	
182	ModeOf-Operation	32bit	R/W	0-255	0	-	Object 6060 subindex 0	
183	ModeOfOperationDisplay	32bit	R	0-255	0	-	Object 6061 subindex 0	
184	Target-Position	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	-	Object 607A subindex 0	
185	Reserved							
186	Actual-Position	32bit	R	$(-2^{31})-(2^{31}-1)$	0	-	Object 6064 subindex 0	
187	Reserved							
188	Target-Velocity	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	-	Object 60FF subindex 0	
189	Reserved							
190	ActualVelocity	32bit	R	$(-2^{31})-(2^{31}-1)$	0	-	Object 606C subindex 0	
191	Reserved							
192	Digital-Outputs	32bit	R/W	0-65535	0	-	Object 60FE subindex 1 (Low 16bit)	
193	Reserved							
194	DigitalInput	32bit	R	0-65535	0	-	Object 60FD subindex 1 (Low 16bit)	
195								

13.4 Motor registers MISxxx

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
202	TICKS	32bit	R/W	0-(2^{32}-1)	0	ms	Timer. Increments at a fixed rate of one count per mS. Starts at zero after the motor has been reset	N/A
212	CUR_SCALE_MAX	32bit	R/W	0-2047	2047	Counts	Closed loop: Max current in closed loop with current control. 2047 = 100 % of RUN_CURRENT.	N/A
213	CUR_SCALE_MIN	32bit	R/W	0-2047	1	Counts	Closed loop: Min current in closed loop with current control. 2047 = 100 % of RUN_CURRENT.	N/A
215	CUR_SCALE_FACTOR	32bit	R/W	1-10,000	500	Counts	Closed loop: The slope of the velocity dependent current decrement rate.	N/A
216	KPHASE	32bit	R/W	0-200	-	Counts	Closed loop: A motor dependent factor which optimizes the commutation angle at high speeds.	N/A
217	ACTUAL_TORQUE	32bit	R	0-2047	-	Counts	Closed loop: The actual motor current in closed loop with active current control. 2047 = 100 % of RUN_CURRENT.	Actual torque
218	CUR_SCALE_INC	32bit	R/W	1-100,000	2000	Counts	Closed loop: Current increment rate in closed loop with current control. (1=fastest)	N/A
219	CUR_SCALE_DEC	32bit	R/W	1-100,000	4000	Counts	Closed loop: Current decrement rate in closed loop with current control. (1=fastest)	N/A
222	XFIELD_ADDR	32bit	R/W	-	0	Special	Address for the internal switch board/cross field setup.	N/A
223	XFIELD_DATA	32bit	R/W	-	0	Special	Data for the internal switch board/cross field setup.	N/A
224-231	FlexRegSetup	32bit	R/W		0	-	Each register in this range sets up 2 bits in the FlexRegister 48 = 16 bits in total.	N/A
232	FlexLEDSetup1	32bit	R/W		0	-	Sets up LED L3 and L2 on the motor.	N/A
233	FlexLEDSetup2	32bit	R/W		0	-	Sets up LED L1 GREEN and L1 RED on the motor.	N/A
236	V_SOLL_AUTO	32bit	R/W	-300,000-300,000	0	0.01 RPM	In position mode the auto correction is run with V_SOLL, but if V_SOLL_AUTO != 0 it will be used in stead.	Auto correction velocity
237	V_IST_CALC	32bit	R	-300,000-300,000	0	0.01 RPM	The theoretical actual velocity.	Actual velocity
238	MOTOR_REV	32bit	R		0	Rev	Number of motor revolutions the motor has run since last power on.	Event log -> Motor rev
239	EX_CYCLIC_SETUP	32bit	R		0	Special	The actual cyclic setup from the Ethernet module. Bit 0-15: Cycle period (us) Bit 16-31: Sync0 offset in percent.	N/A
241	EX_CRC_ERR	32bit	R		0	Counts	CRC error counter of the internal communication between controller and Ethernet module.	N/A
242	V_HOME_CRAWL	32bit	R/W	0-300,000	0	0.01 RPM	In Zero Search type 2, the “crawl” velocity is V_HOME/64 by default. If register 242 is !=0, a user defined velocity is used.	Zero search crawl velocity
243	V_HOME_TIMEOUT	32bit	R/W		0	ms	If 0, the Zero Search time out is 60000 ms. Else the value in this register is used.	Zero search time out
244	TEMP_LIMITS	32bit	R		0	Special	The actual temperature limits in the motor: Bit 0-15: Warning limit (unit: degC) Bit 16-31: Error limit (unit: °C)	N/A
245	CL_CATCH_UP	32bit	R/W	-	0	Special	Bit 0-7: Allowable overspeed in percent (0-100) Bit 8-31: Follow error limit before overspeed is used.	Allowable overspeed Follow error before overspeed
252	LOWBUSCVI_CNT	32bit	R/W		10	Counts	Number of times in a row the voltage can be too low before error is set. Time between each measurement = 100 us.	N/A
253	V_ENCODER	32bit	R	-300,000-300,000	-	0.01 RPM	The actual internal encoder velocity.	Internal encoder velocity