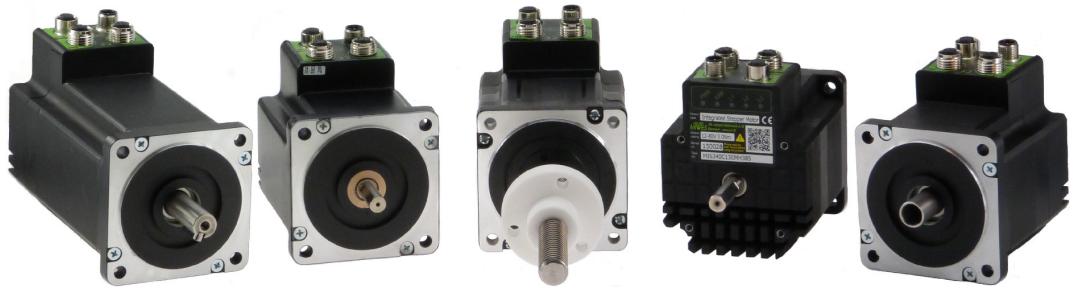


# **Integrated Step Motors, QuickStep,**

**MIS23x (Generation 2)  
MIS340, MIS341, MIS342,  
MIS430, and MIS432**

**Including Step Motor Controller  
SMC66, SMC85**

## **User Manual**



**JVL Industri Elektronik A/S**

# Important User Information



## Warning



The MIS and SMC 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 for technical assistance. Your nearest contact can be found on our web site [www.jvl.dk](http://www.jvl.dk)

Copyright 1998-2017, JVL Industri Elektronik A/S. All rights reserved.  
This user manual must not be reproduced in any form without prior written permission of JVL Industri Elektronik A/S.  
JVL Industri Elektronik A/S reserves the right to make changes to information contained in this manual without prior notice.  
Furthermore JVL Industri Elektronik A/S assumes no liability for printing errors or other omissions or discrepancies in this user manual.

*MacTalk and MotoWare are registered trademarks*

JVL Industri Elektronik A/S  
Blokken 42  
DK-3460 Birkerød  
Denmark  
Tlf. +45 45 82 44 40  
Fax. +45 45 82 55 50  
e-mail: [jvl@jvl.dk](mailto:jvl@jvl.dk)  
Internet: <http://www.jvl.dk>

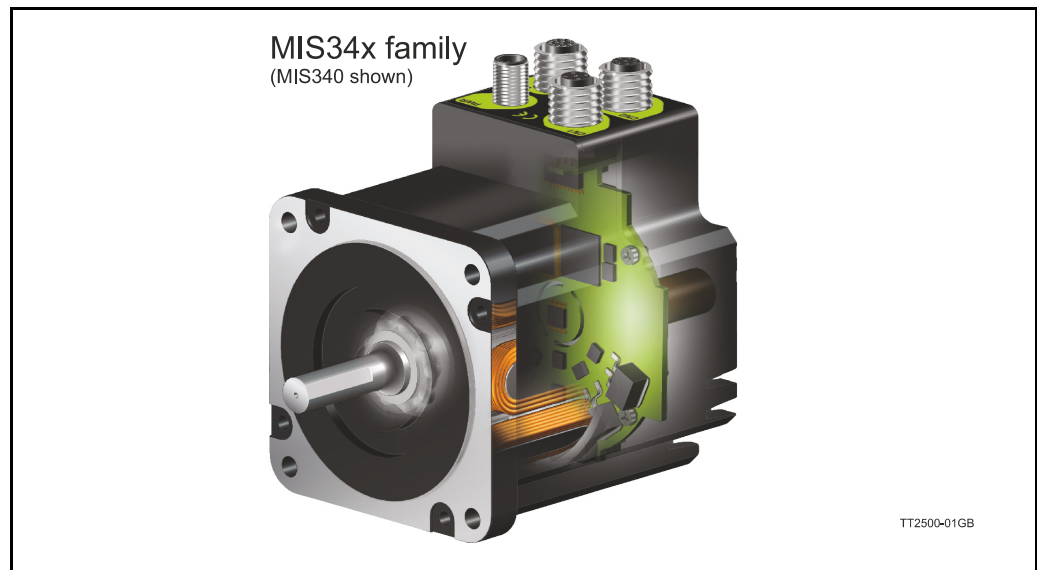
# Contents

---

<b>I</b>	<b>Introduction .....</b>	<b>5</b>
1.1	Feature overview .....	6
1.2	General description .....	9
<b>2</b>	<b>Hardware .....</b>	<b>11</b>
2.1	Power Supply .....	12
2.2	User Inputs .....	18
2.3	Analogue Inputs .....	22
2.4	User Outputs .....	28
2.5	Serial interfaces overview .....	30
2.6	RS485 Interface .....	31
2.7	EMC considerations .....	32
2.8	How to connect a MIS motor .....	34
2.9	LED indicators basic motor .....	50
2.10	LED indicators using CANopen .....	51
2.11	LED indicators using Ethernet .....	52
<b>3</b>	<b>Hardware None-intelligent products .....</b>	<b>53</b>
<b>4</b>	<b>Using MacTalk .....</b>	<b>55</b>
4.1	Using the MacTalk software .....	56
4.2	How to update MacTalk .....	64
4.3	How to update the motor firmware .....	65
4.4	How to update the encoder FW .....	66
4.5	How to get SW/HW motor info .....	67
<b>5</b>	<b>Description of functions .....</b>	<b>69</b>
5.1	Setting up the motor current .....	70
5.2	Auto Correction .....	72
5.3	Closed loop operation .....	76
5.4	Absolute position back-up .....	85
5.5	Multifunction I/O setup .....	88
5.6	Dedicated outputs .....	91
5.7	SSI encoder/sensor interface .....	92
5.8	Absolute Multi-turn Encoder .....	97
5.9	Position Limits .....	102
5.10	Under voltage Handling .....	107
5.11	Electro Mechanical brake .....	110
<b>6</b>	<b>Modes .....</b>	<b>113</b>
6.1	Passive Mode .....	114
6.2	Velocity Mode .....	115
6.3	Positioning Mode .....	116
6.4	Gear Mode .....	117
6.5	Zero search modes .....	125
<b>7</b>	<b>Error Handling .....</b>	<b>131</b>
7.1	Setup error limits .....	132
7.2	Error messages .....	133
<b>8</b>	<b>Registers .....</b>	<b>141</b>
8.1	Introduction to registers .....	142
8.2	Internal registers .....	143
<b>9</b>	<b>Building Sequential Programs .....</b>	<b>197</b>
9.1	Getting started with programming .....	198
9.2	Programming Main window .....	199
9.3	Programming menu .....	200
9.4	How to build a program .....	201
9.5	General programming hints .....	204
9.6	Command toolbox description .....	205

9.7	Graphic programming command reference .....	206
9.8	Command timing .....	225
9.9	More about program timing .....	226
<b>10</b>	<b>Ethernet protocols (optional) .....</b>	<b>227</b>
<b>11</b>	<b>CANopen (optional) .....</b>	<b>229</b>
11.1	General information about CANopen .....	230
11.2	Connection and setup of the CAN bus .....	234
11.3	Using CANopenExplorer .....	237
11.4	Objects in the DS301 standard .....	242
11.5	Objects used in the DSP-402 standard .....	252
11.6	Flexible Register setup .....	259
11.7	More details of CANopen Theory .....	260
<b>12</b>	<b>Modbus interface .....</b>	<b>271</b>
12.1	Modbus .....	272
<b>13</b>	<b>Stand alone electronics .....</b>	<b>275</b>
13.1	Step motor controllers (SMC66/85) .....	276
<b>14</b>	<b>Technical Data .....</b>	<b>279</b>
14.1	MIS23x Technical Data .....	280
14.2	MIS34x Technical Data .....	281
14.3	Torque Curves .....	282
14.4	Physical Dimensions .....	288
14.5	Life time .....	292
14.6	Trouble-shooting guide .....	293
<b>15</b>	<b>Accessories .....</b>	<b>295</b>
15.1	Cables .....	296
15.2	Power Supplies .....	297
15.3	Brakes and shaft reinforcement .....	298
15.4	Gear and brake mounting instruction .....	299
<b>16</b>	<b>Appendix .....</b>	<b>301</b>
16.1	Motor Connections .....	302
16.2	Serial communication .....	304
16.3	MIS Ordering Information .....	309
<b>17</b>	<b>Declarations .....</b>	<b>313</b>
17.1	CE Declaration of Conformity .....	314
17.2	Vibrationtest certificates MIS34x .....	316





This user manual describes the set-up and usage of the following products:

Complete motors with build-in controller or driver

- Types **MIS171**, **MIS172** and **MIS176** (NEMA17 sizes)
- Types **MIS231**, **MIS232** and **MIS234** (NEMA23 sizes)
- Types **MIS340**, **MIS341** and **MIS342** (NEMA34 sizes)
- Types **MIS43x** (NEMA43 sizes) - only limited supported in this manual.

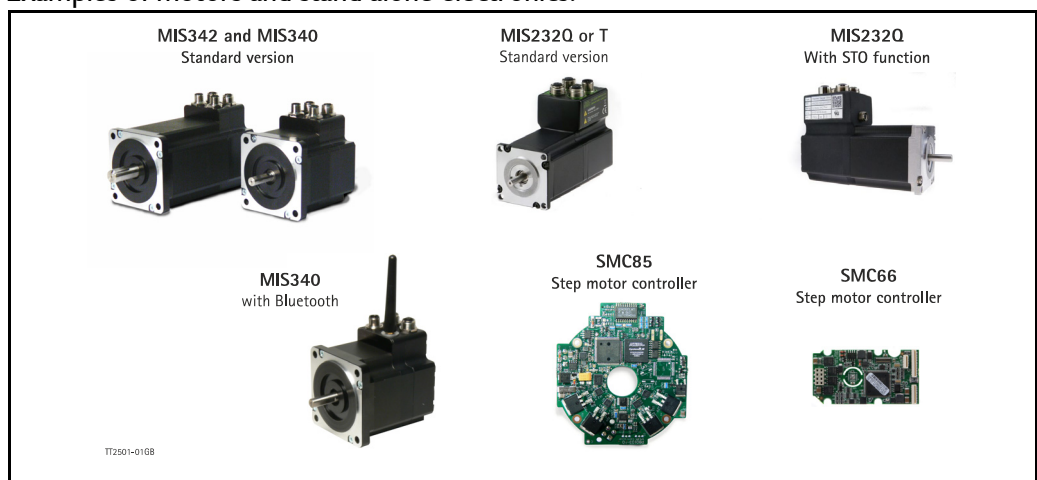
Important please notice that only the latest generation of MIS231 to 234 is supported by this user manual. The extension after MIS23x must be: S, Q, T or R.

Stand-alone electronics without motor

- Types **SMC66** and **SMC85** controller PCB with intelligence (fully programmable)

All the Quickstep motors are available as a fully programmable product with a wide range of features also covering a simple pulse and direction interface or Ethernet options.

Examples of motors and stand alone electronics.



# 1.1

# Feature overview



The compact MIS motors are designed for easy installation and high performance use. A large number of features are available and also multiple options available as listed below.

- Serial RS485 or 5V serial position controller.
- Build-in mini PLC with graphic programming.
- Option for CANbus, CANopen DS-301. Fully ISO 11898-2:2016 compliant/(DSP-402 in development).
- Options for EthernetIP, Profinet, Powerlink, ModbusTCP, SercosIII and EtherCAT.
- A dual supply facility is available so that position and parameters are maintained at emergency stop.
- Electronic Gear mode.
- MACmotor protocol so MAC servomotors and MIS stepper motors can be connected on the same RS485 bus.
- Command for easy PLC/PC setup and communication.
- Power supply 12-72 VDC.
- Extremely high torque vs speed - up to 3000 RPM with good performance.
- Fixed 409600 steps per revolution
- Built-in 32Bit  $\mu$ processor with 8 In/Out that can be configured as inputs, PNP outputs or analogue inputs. 5V serial and RS485 interface for set up and programming.
- MODBUS interface.
- 9.6kbit/sec. to 1Mb/sec. communication.

Benefits when using the MIS motors:

- De-central intelligence.
- Simple installation. No cables between motor and controller/driver.
- EMC safe. Switching noise remains within motor.
- Compact. Does not take space in the control cabinet.
- Low-cost alternative traditional systems where motor and controller is separated.
- Option: Closed loop feature by means of magnetic encoder with resolution of up to 4096 pulses/rev. (H2 or H4 option)
- Option: Absolute multi turn encoder for keeping the position permanent also during power down. (H3 or H4).
- Vibration tested at 4G in 3 axis and shock tested at 15G in 3 axis according to IEC60068.
- Interface possibilities:
- From PC/PLC with serial commands via 5V serial or RS485.
- Pulse/direction input. Encoder output.
- CANopen.
- 8 I/O, 5-28VDC that can be configured as Inputs, Outputs or analogue inputs.
- Wireless options: WiFi, Bluetooth and Zigbee.

# 1.1

## Feature overview

The MIS motors are also available with alternative options.

- Protection class IP65.
- Hollow shaft.
- Integrated ball screw or spindle for linear movement.
- Custom made design for special applications

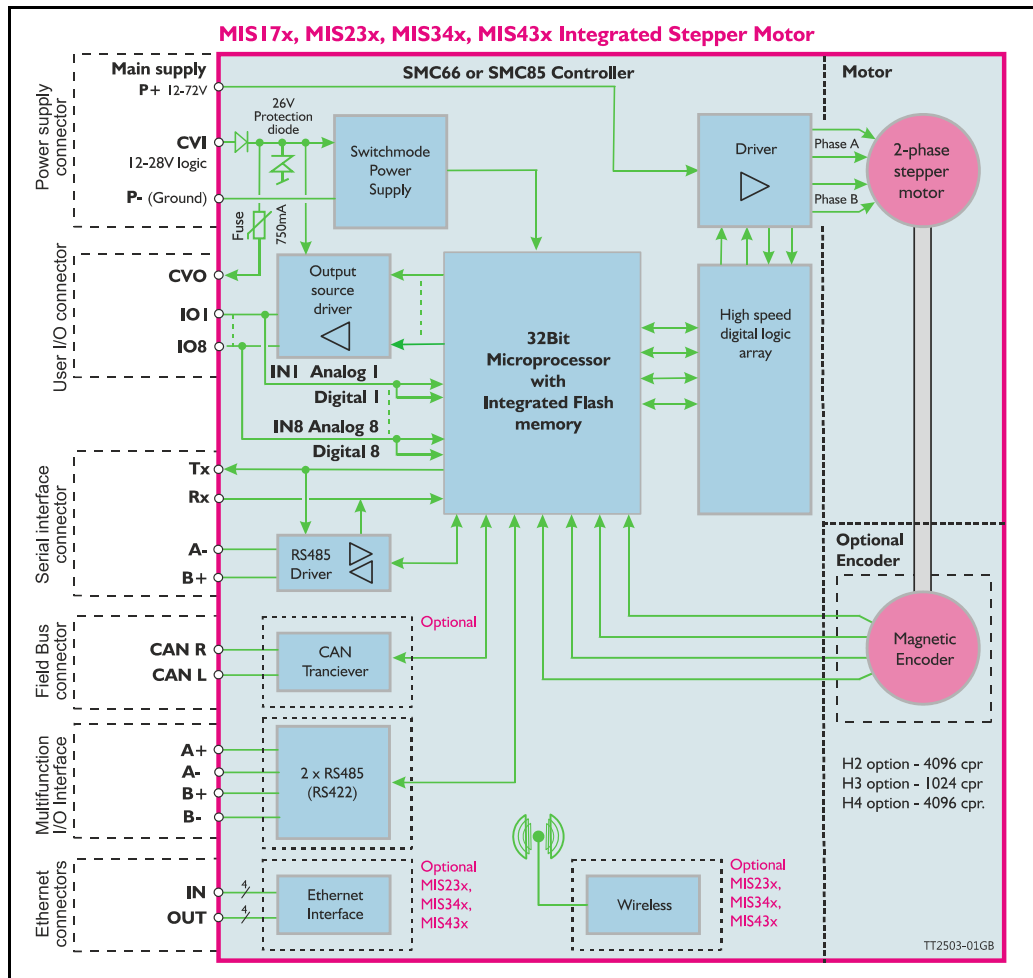


# 1.1

# Feature overview

## 1.1.1

### Block diagram, Positioning/Speed Control



## 1.2

## General description

The QuickStep motors are currently available in 6 different models divided in 2 families. NEMA23 covers: MIS231, MIS232 and MIS234, with holding torque ratings from 1.1 to 3.1 Nm and NEMA34 covers: MIS340, 341 and 342. The basic functions and I/O features are the same for all models. MIS43x models up to 25.0 Nm are under development.

Motor Type	MIS231	MIS232	MIS234	MIS340	MIS341	MIS342	Unit
Holding Torque	1.1	1.6	2.9	3.0	6.1	9.0	Nm
Inertia	0.3	0.48	0.96	1.4	2.7	4.0	kgcm <sup>2</sup>
Flange	NEMA23 (57x57 mm.)			NEMA34 (87x87 mm)			-
Length	96	118.5	154	9[3.74]	126[4.96]	156.0[6.14]	mm [Inch]
Shaft Ø	6.35	6.35	10.0	9.53	14.0	14.0	mm
Shaft radial play	Max. 0.02 (450g load)			Max. 0.02 (450g load)			mm
Shaft axial play	Max. 0.08 (450g load)			Max. 0.08 (450g load)			mm
Max radial force	7.5 (20mm from flange)			22 (20mm from flange)			kg
Max axial force	1.5			6			kg
Weight	0.9	1.2	1.8	2.7	4.2	5.8	kg

### 1.2.1

#### Basic modes/functions in the QuickStep motor

The QuickStep motor offers the following functions:

Mode	Description
Passive	The motor will be in a completely passive state but communication is active and internal registers can be setup. Motor shaft can be turned by hand.
Velocity	The motor velocity can be controlled using MacTalk software or by setting register 5 (V_SOLL) using serial or program commands.
Position	The motor position can be controlled using MacTalk or by setting register 3 (P_SOLL) using serial or program commands.
Gear	The motor position and velocity can be controlled by pulse and direction or encoder signals at the inputs "IN1" and "IN2". The gear ratio can be set to a large ratio by using register 14 (GEAR1) and register 15 (GEAR2).
CSP Mode	Cyclic Synchronous Position mode (Ethernet only)



The following pages explains how the I/O, Power supply, Interface etc. can be connected and used.

## 2.1

# Power Supply

### 2.1.1 General Aspects of Power Supply

Powering of the motor is relatively simple.

The supply input of the MIS motor family is equal for all family members.

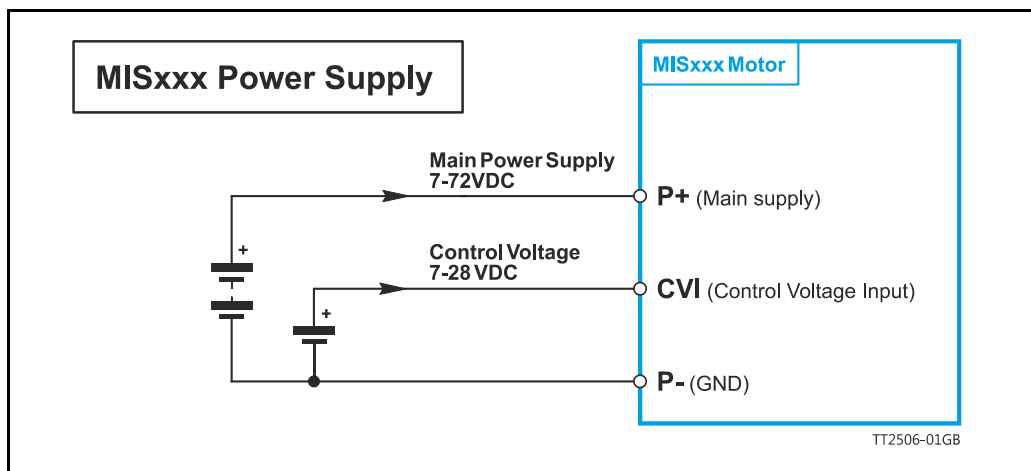
The supply consist of 2 inputs:

**CVI** The control voltage input is supplying all the internal control circuits including the user I/O circuitry. The voltage needed must be in the range 7-28 VDC which also support battery driven applications. Supply current is below 200 mA (voltage dependant and no user outputs activated).

**P+** The main power for driving the motor is supplied on this terminal and must be in the range 7-72 VDC. The voltage at this terminal will also influence torque at higher velocities. A voltage of 72 VDC will give much higher torque than using for example 24 VDC. The supply current can get as high as 6 ARMS. See also [Torque Curves](#), page 282 which shows the relation between supply voltage and the torque.

If the motor need to be stopped for safety reasons it can be done by removing P+ but keeping CVI connected. This will keep the complete motor alive including I/O's and encoder circuit (if present) except that the motor driver and motor is power less and will not rotate.

Supplying both terminals from the same power supply is not a problem but the voltage must be maximum 28 VDC in order to respect the maximum voltage for the CVI input.



**NB:** for actual connections, see drawing [How to connect a MIS motor](#), page 34



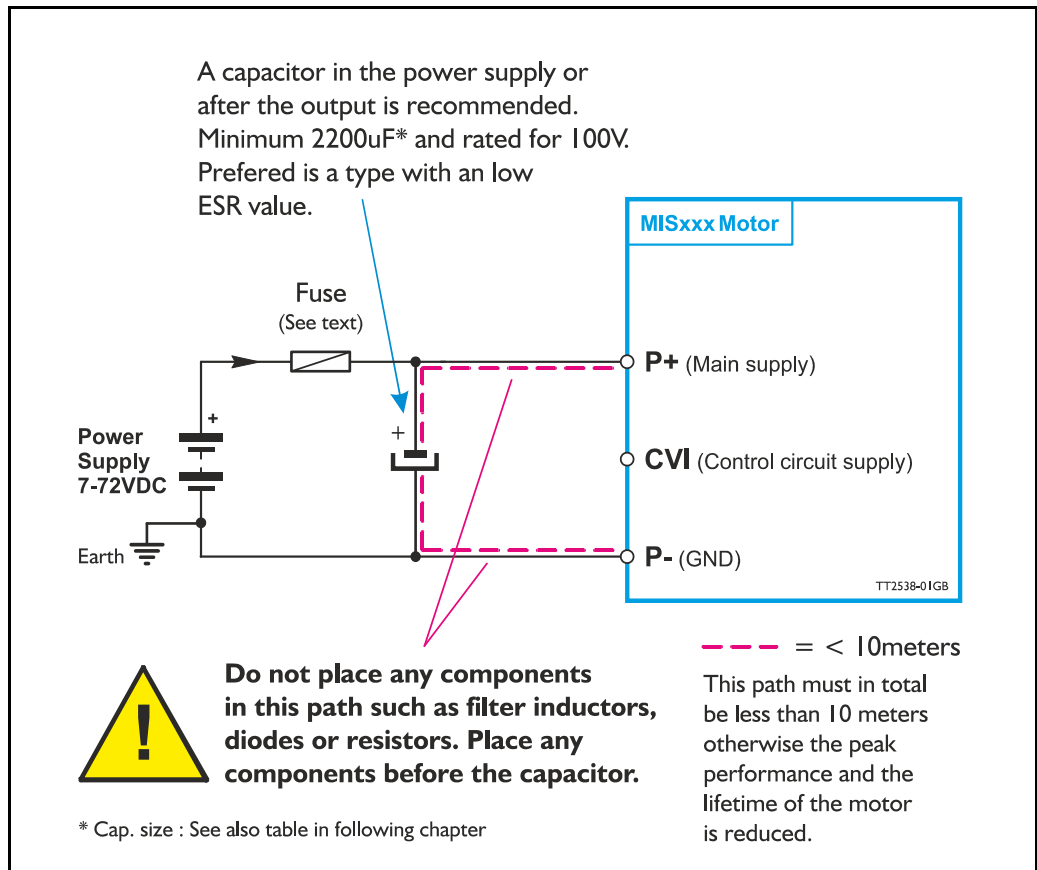
## 2.1

# Power Supply

### 2.1.2 Power Supply - Requirements and Precautions

For optimum performance and lifetime of the MIS motor, it is recommended that a capacitance of minimum  $2200\mu\text{F}$  is connected to the power supply that supply the P+ terminal. It should be mounted as close as possible to the motor.

Also, it is recommended that minimum  $0.75\text{ mm}^2$  cable is used to connect the power supply to the motor. If the supply voltage drops below  $7\text{V}$ , the internal reset circuitry will reset the driver and an under voltage error is generated. Provision should therefore be made to ensure that the supply voltage is always maintained at a minimum of  $7\text{V}$ , even in the event of a mains voltage drop.



CVI supply Precautions.

The CVI supply is not critical since the supply current is quite small ( $<200\text{ mA}$ ). Only make sure that the voltage stay at  $24\text{ VDC}$  nominal and do not exceed  $30\text{ VDC}$ . A CVI voltage down to  $7\text{ VDC}$  is also possible but a software setup is needed to allow this. See also [Setup position backup using MacTalk](#), page 85

**Warning:** A supply voltage at CVI or P+ higher than  $100\text{VDC}$  will cause permanent damages. A voltage over  $30\text{V}$  at the CVI will activate a protection circuit which shuts down the supply input. In this case CVI need to be disconnected to reset the protection state.

## 2.1

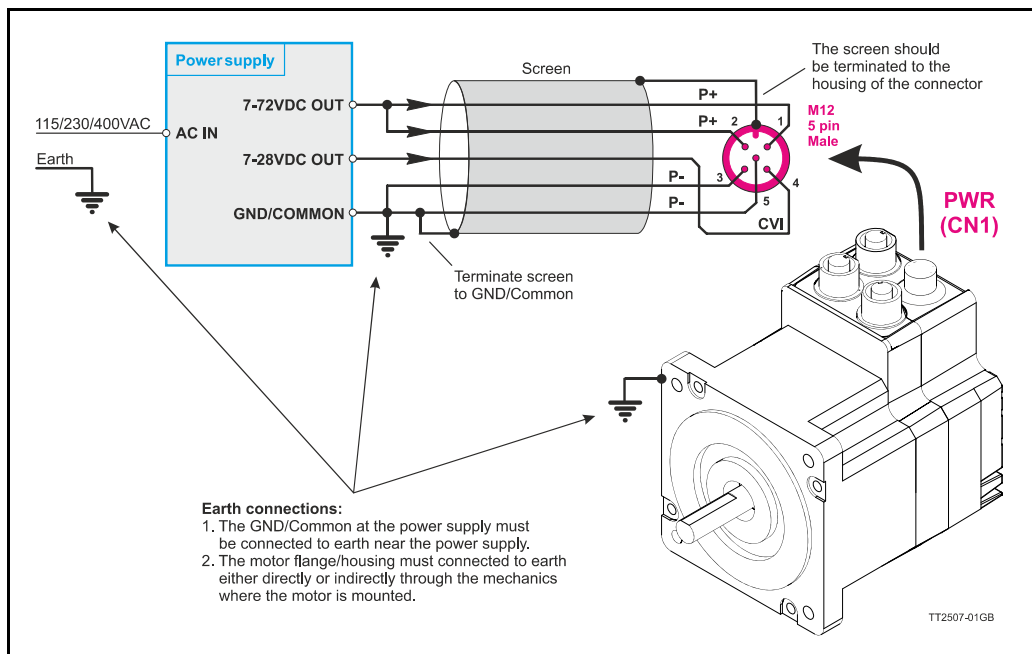
# Power Supply

### 2.1.3 Power Supply Grounding and Earthing

It is mandatory that the motor flange is connected to earth. Also it is mandatory that the earth is connected to GND/Common at a central point near the power supply.

The P- (GND/Common) is internally connected to the motor chassis/housing. Also the outside body at all M12 connectors is connected to the chassis/housing and thereby to the P- (GND/Common).

The illustration below shows how to make a good power and earth connection of the MIS motor.



## 2.1

# Power Supply

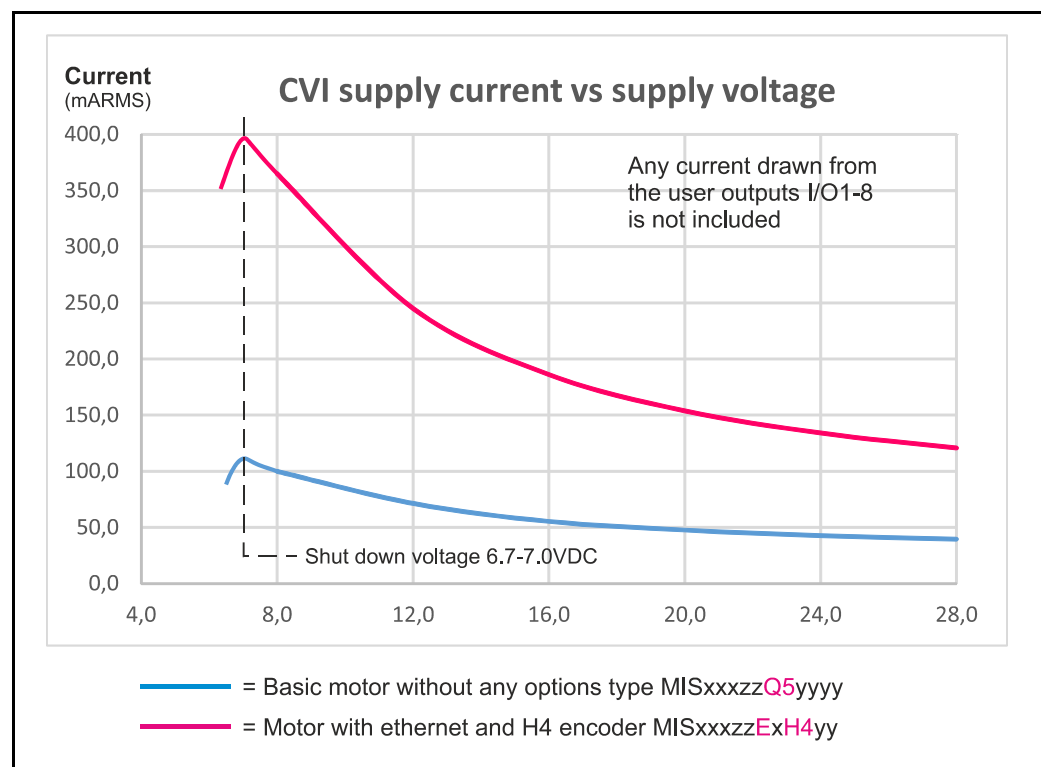
### 2.1.4 Control Voltage (CVI terminal)

The control voltage should be in the range 7-28VDC and is used to supply the microprocessor circuit, internal functions in general and the user output driver (O1-8).

To ensure that position and parameters are maintained after an emergency stop, the control voltage should be maintained under a stop situation where the P+ (main power) is disconnected.

**Warning:** a voltage at the CVI terminal higher than 30VDC can damage the controller or cause malfunction. A protection circuit will protect from damages. If this protection circuit get triggered the CVI power need to be cycled in order to reset the protection.

The figure below shows the typical relation between supply current and supply voltage at the CVI input. As shown the current is very dependant at which options is installed in the motor. Worst case is if the motor is equipped with Ethernet and H4 encoder and best case is the basic motor such without Ethernet and encoder.



**Please notice that the user I/O's are supplied from the CVI terminal. The curves shown above do NOT include any load current at the user outputs. Add the load current(s) to the current shown above. If the motor is equipped with an internal electromechanical brake this must also be added to the current consumption.**

## 2.1

# Power Supply

### 2.1.5 Dimensioning power supply and fuse - only MIS23x motors

Notice that this manual only covers MIS23x/MIL23x generation 2 motors.  
The power supply must be dimensioned according to the actual motor size.  
The size of the pre-fuse also depends on the actual model of the MIS motor.  
Use the following table to select the power supply and fuse ratings.

Desired voltage	MIS231/MIL231		MIS232/MIL232		MIS234/MIL234	
	Supply rating	Fuse size	Supply rating	Fuse size	Supply rating	Fuse size
-						
12VDC	20W	T4A	40W	T6.3A	60W	T10A
24VDC	40W	T4A	80W	T6.3A	160W	T10A
48VDC	80W	T4A	160W	T6.3A	320W	T10A
Recommended power supply	PSU24-075 PSU48-240 PSU80-4		PSU24-240 PSU48-240 PSU80-4		PSU24-240 PSU48-240 PSU80-4	

See also the appendix which shows the standard power supplies that JVL offers.

### 2.1.6 Dimensioning power supply and fuse - only MIS34x motors

The power supply must be dimensioned according to the actual motor size.  
The size of the pre-fuse also depends on the actual model of the MIS motor.  
Use the following table to select the power supply and fuse ratings.

Desired voltage	MIS340/MIL340		MIS341/MIL341		MIS342/MIL342	
	Supply rating	Fuse size	Supply rating	Fuse size	Supply rating	Fuse size
-						
24VDC	120W	T6.3A	200W	T6.3A	250W	T10A
48VDC	240W	T6.3A	350W	T6.3A	500W	T10A
72VDC	450W	T6.3A	600W	T6.3A	700W	T10A
Recommended power supply	PSU24-240 PSU48-240 PSU80-4 PSU80-1000-10		PSU24-240 PSU48-240 PSU80-4 PSU80-1000-10		PSU24-240 PSU80-4 PSU80-1000-10	

Please notice that the specified wattage values are worst case values at maximum torque.

See also the appendix which shows the standard power supplies that JVL offers.

## 2.1

# Power Supply

---

### 2.1.7 Select Your Power Supply

We recommend the use the highest possible voltage to supply the motor.

In general the motor torque from a MIS is not affected by the supply voltage at speeds below 100RPM but at higher velocities the torque will be very influenced by the supply voltage (P+ terminal).

Additionally, higher voltage gives better current and filter regulation and thereby better performance. If there is a tendency for motor resonance, a lower supply voltage can be a solution to the problem.

## 2.2

# User Inputs

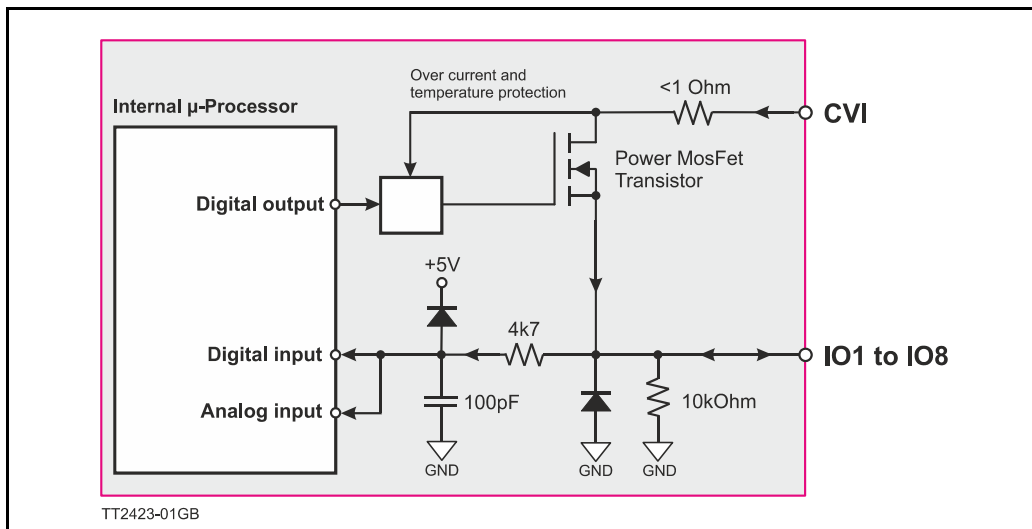
### 2.2.1 User Inputs

The MIS motors has 8 inputs/outputs (IO's) that each can be set individually to input, output or analogue input 0-5VDC via MacTalk or software commands. This makes it for example possible to have 4 inputs, 3 outputs and one analogue input.



**Please notice:** The number of available I/O terminals available may differ depending at which motor type and connector configuration you are using. Please consult the chapter [Connector overview for the MIS motors](#), page 34

**Input/output functional diagram:**

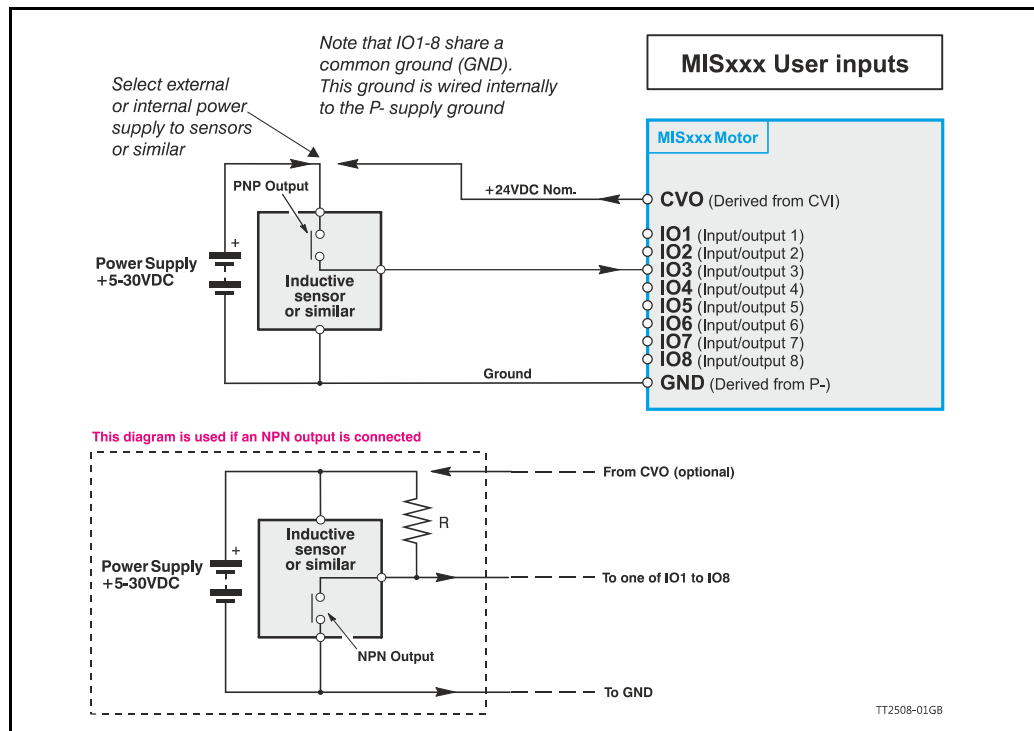


### 2.2.2 General Input features

- Inputs are TTL to 28VDC compliant.
- Over-current protection and thermal shut-down.
- 10 kOhm input resistance.
- No galvanic isolation but very robust against noise and spikes/surges.
- Zero search input can be selected to any input 1 to 8.
- Digital filter can be enabled for each input selectable from 0 to 100ms. If disabled (default), the response time is 100 $\mu$ s.
- Limit switch inputs

## 2.2

# User Inputs



### 2.2.3

#### General

The Controller is equipped with a total of 8 digital inputs. Each input can be used for a variety of purposes depending on the actual application. Each of the inputs can be detected from the actual program that has been downloaded to the Controller or via serial commands.

The Inputs are not optically isolated from other Controller circuitry. All of the Inputs have a common ground terminal, denoted *GND*. Each Input can operate with voltages in the range 5 to 30VDC. Note that the Inputs should normally be connected to a PNP output since a positive current must be applied for an input to be activated.

Note that CVO (control voltage output) is internally connected to the CVI supply terminal in the PWR connector. This provides the facility that local sensors can be supplied directly from the controller.

### 2.2.4

#### Connection of NPN Output

If an Input is connected to an NPN output, a Pull-Up resistor must be connected between the Input and the + supply. See the illustration above.

The value of the resistance used depends on the supply voltage. The following resistances are recommended:

Supply Voltage	Recommended Resistance R
5-12VDC	1kOhm / 0.25W
12-18VDC	2.2kOhm / 0.25W
18-24VDC	3.3kOhm / 0.25W
24-30VDC	4.7kOhm / 0.25W

## 2.2

# User Inputs

### 2.2.5 Digital inputs - Usage.

All of the eight I/O signals can be used as digital inputs. The sampled and possibly filtered value of each input is stored in the Input's register (reg. 18). Unlike the analogue inputs, there is only one value for each digital input, so it must be configured to be either unfiltered or filtered.

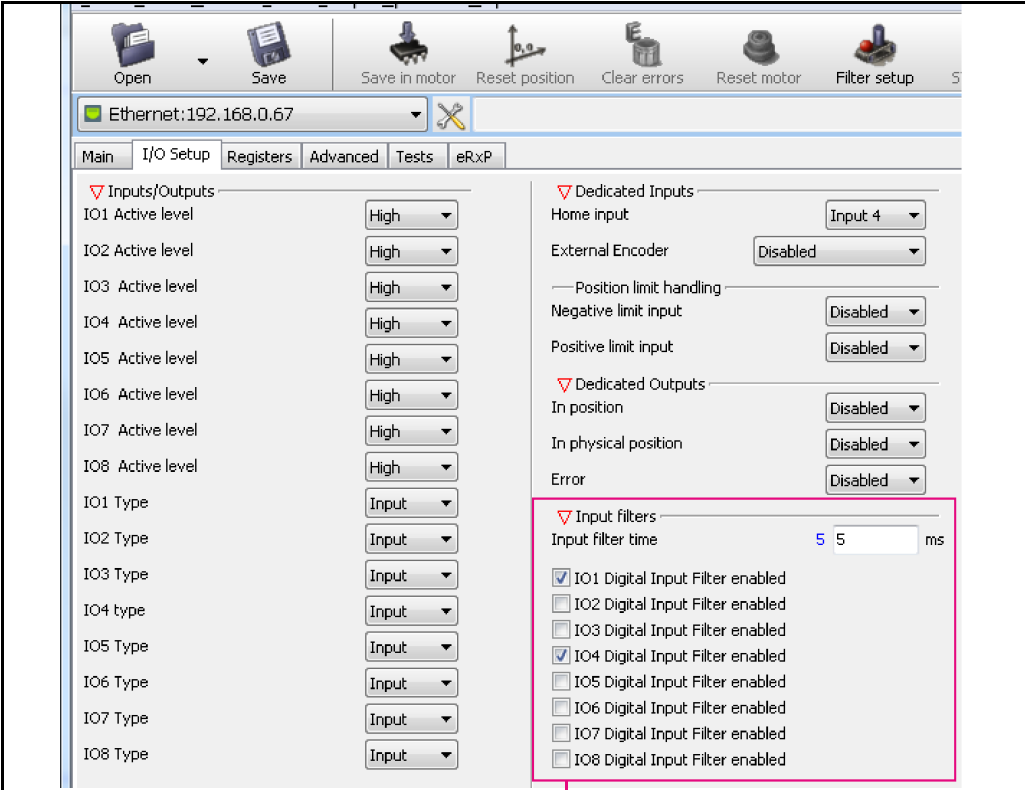
Unfiltered (high-speed) digital inputs are sampled every 100  $\mu$ S (micro-seconds). Filtered digital inputs are sampled every milli-second, and the filter value can be set in the range 1 to 100 mS, so the filtered input must be sampled to have the same logical value for that number of samples in a row. Once an input has changed state after passing the filtering, it will again take the same number of samples of the opposite logical level to change it back. For example, if the filter is set to 5 mS and the start value is 0 (zero), the input will remain at zero until three samples in succession have been read as 1 (one). If the signal immediately drops down to 0 again, it will take three samples of zero in succession before the register bit gets set to zero.

Note that enabling filtering of the digital inputs does load the micro-controller, so if filtering of the digital inputs is not needed, ALL the inputs can be selected as high-speed to optimise the available resources from the micro controller.

### 2.2.6 Digital input filter setup with MacTalk:

By default, the digital input filters are disabled and therefore the inputs are sampled every 100  $\mu$ s.

If "IOx Digital Filter enabled" is set, the specific input will use the digital filter according to the "Input filter time". The remaining digital inputs will still be updating every 100  $\mu$ s.



The screenshot shows the MacTalk software interface for configuring digital inputs. The 'Input filters' section is highlighted with a pink box. It contains a list of checkboxes for IO1 through IO8, with IO1 and IO4 checked. The 'Input filter time' is set to 5 ms. Below the screenshot, a text box explains: 'Select inputs that need filtering and set the filter time here. The filter time is common for all inputs selected.'



## 2.2

## User Inputs

---

### 2.2.7 Digital input filter setup without MacTalk:

If MacTalk is not used for setting up parameters/registers related to the digital filters it must be done as follows.

The motor contains a number of registers which can be accessed from various protocols depending at which options the motor has.

Protocols available are for example Ethernet (EthernetIP, ProfiNet etc.) and CANopen, Modbus or the MacTalk protocol.

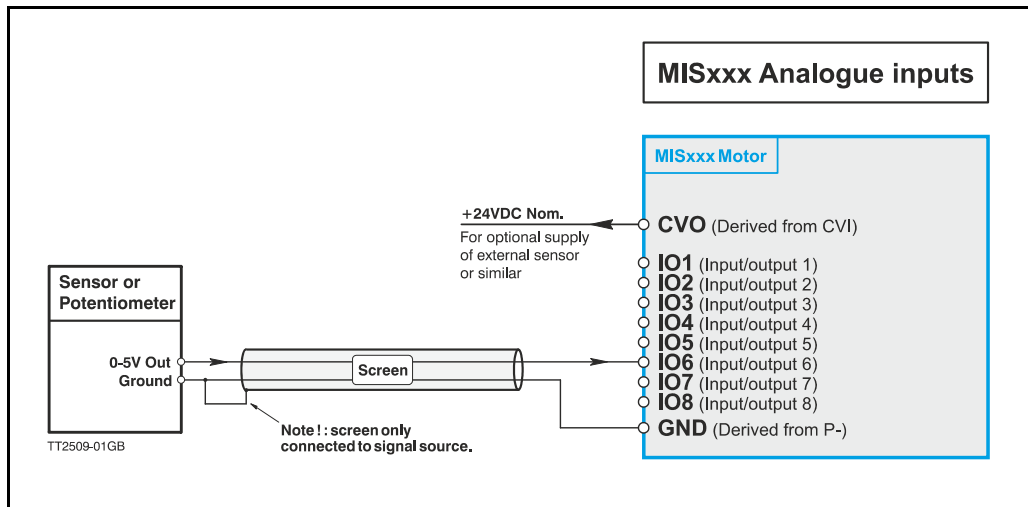
Each field in MacTalk described earlier in this chapter is accessing a register in the motor. The registers that are relevant for digital filters setup are:

**R135** INPUT\_FILTER\_MASK This register controls filtering of each of the eight I/O pins that are used as digital inputs. If the bit corresponding to the input number is set in this register, the filter will be enabled.  
See also: [Input\\_Filter\\_Mask](#), page 177

**R136** INPUT\_FILTER\_CNT The filtering of all of the eight digital inputs is controlled by the value in this register together with register 135. The input must be sampled at the same value for the specified number of milliseconds in this register to be accepted as the new filtered usable value. See also [Digital inputs - Usage.](#), page 20  
See also: [Input\\_Filter\\_Cnt](#), page 177

## 2.3

# Analogue Inputs



### 2.3.1 General

The 0-5V Analogue Inputs are used for example when the Controller is operated as a stand-alone unit. In this kind of application it can be an advantage to use a potentiometer, joystick or other device for adjusting speed, position, acceleration, etc.

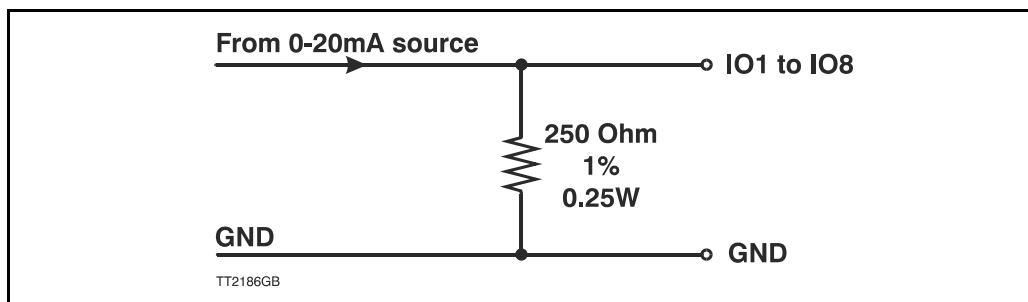
In these modes of operation, the motor is controlled to produce a velocity or position, etc., which is determined by, and proportional to, the voltage applied to the Analogue Input.

The Analogue Inputs share a common internal supply with the GND and P- terminal and are not optically isolated from all other inputs and outputs. The Analogue Inputs are protected against voltage overload up to 30V peak and have a built-in filter which removes input signal noise. See [Analogue input filters](#), page 23.

Always use shielded cable to connect the source used to control an Analogue Input since the motor, etc., can easily interfere with the analogue signal and cause instability.

The Controller is equipped with 8 analogue-to-digital converters (ADC) which convert the detected analogue signal level. The ADCs have a resolution of 12bit.

In order to use the Analogue Inputs as 0-20 mA inputs, a 250  $\Omega$ , 1% resistor must be connected between IO 1-8 and GND.



**Please notice:** The number of available I/O terminals available may differ depending at which motor type and connector configuration you are using. Please consult the chapter [Connector overview for the MIS motors](#), page 34

## 2.3

# Analogue Inputs

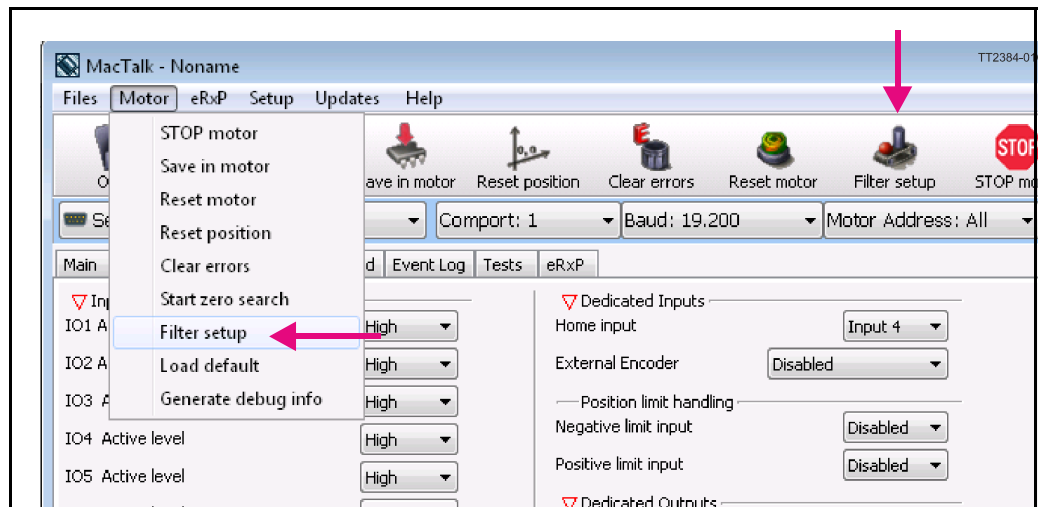
### 2.3.2 Analogue input filters

An analogue signal is not always fully stable and may fluctuate a bit. Also general noise and sudden spikes from other equipment can be a problem.

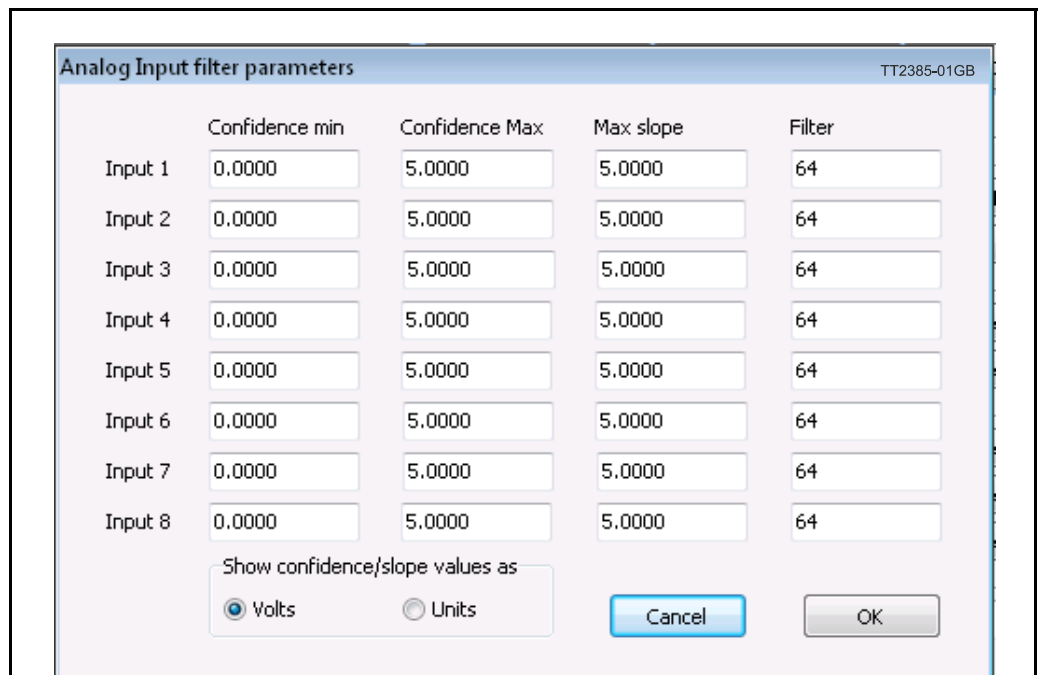
To help filtering an analogue input signal the MIS have an advanced input filter function. This can be setup as follows.

### 2.3.3 Analog filter setup with MacTalk:

It is strongly recommended to setup the analog input filtering using MacTalk. The setup dialog is found in the menu Motor -> Filter setup, or the "Filter setup" button on the toolbar.



Write the settings for each Input and click 'OK'. The parameters can afterwards be "Saved in motor".



## 2.3

# Analogue Inputs

---

### 2.3.4 Analog filter setup without MacTalk:

If MacTalk is not used for setting up parameters/registers related to the analog filters it must be done as follows.

The motor contains a number of registers which can be accessed from various protocols depending at which options the motor has.

Protocols available are for example Ethernet (EthernetIP, ProfiNet etc.) and CANopen, Modbus or the MacTalk protocol.

Each field in MacTalk described earlier in this chapter is accessing a register in the motor.

The registers that are relevant for analog filters setup are:

#### **R100 Afzup\_WriteBits**

When changing values for the analogue input filter parameters, this register is used in combination with registers 102-106. First, all of the registers 102-106 must be loaded with the values to be used for one or more analogue input filters. Then the lower eight bits in this register are set to select which inputs the parameters in registers 102-106 should control. The firmware will detect this and copy the parameter values from registers 102-106 to internal storage. Once this has been completed, the firmware sets bit 15 in this register to show that registers 102-106 are free to receive new values for programming the remaining inputs with other filter parameters. To use the same filtering for all analogue inputs, this register can be loaded with 255 (hex FF).

#### **R101 Afzup\_ReadIndex**

This register makes it possible to read back the analogue input filter parameters for 1 analogue input at a time. To select a new input, write a value of 1 to 8 to this register and wait for bit 15 to be set high. When bit 15 has been set by the firmware, the registers 102-106 have been loaded with the filter parameters currently used by that analogue input.

These registers acts as setup registers for the analogue filters, but also as the read-out of the actual settings. The setup and read-out procedures are described above.

<b>R102 Afzup_ConfMin</b>	Confidence minimum.
<b>R103 Afzup_ConfMax</b>	Confidence maximum.
<b>R104 Afzup_MaxSlope</b>	Max slope.
<b>R105 Afzup_Filter</b>	Filter.

The result of the filtered analog inputs can be read in the following registers. They are not visible in MacTalk, but can be used in an RxP program or monitored through other interfaces (Ethernet, CANopen, Modbus etc.)

#### **R81-88 Analog\_Filtered**

The voltage on inputs 1 to 8 after being filtered in firmware. See the [Afzup\\_Filter](#), page 169 for filter parameters. 5.00V is equal to a value of 4095.

#### **R89-96 Analog Input**

The unfiltered voltage on inputs 1 to 8. 5.00V is equal to a value of 4095.

## 2.3

# Analogue Inputs

---

### 2.3.5 Detailed description of the analog filter function

The MIS motors have 8 general-purpose I/Os, that can be used as both digital inputs, digital outputs and analogue inputs. When an I/O is configured to be an input, it simultaneously has both a digital value (high or low) and an analogue value in the range 0.00 to 5.00 Volts. Input voltages higher than 5.0 Volts will be internally limited and read as 5.00 Volts.

The inputs use a resolution of 12 bits, which means that in the raw motor units a value of 5.00 Volts reads out as the value 4095.

This gives a resolution of  $5.00/4095 = 1.221$  mV per count.

The eight values from the analogue inputs are maintained by the MIS firmware in the registers 89...96 as raw, unfiltered values with the fastest possible update frequency, and additionally in the registers 81...88 as filtered values. The firmware does not use any of the values for dedicated functions. It is always up to the program in the motor to read and use the values.

The analogue filtered values are typically used to suppress general noise or to define how quickly the input value is allowed to change, or in some cases to limit the input voltage range. A typical example is an analogue input that is connected to a manually controlled potentiometer, so an operator can regulate the speed of the machine by turning a knob. In many environments, this setup is subject to noise, which could make the motor run unevenly, and cause too sharp accelerations or decelerations when the knob is turned.

The filter functions supported in the MIS firmware always use three different steps.

#### Confidence check

First the raw input value is compared to two Confidence limits: Confidence Min and Confidence Max. If the new value is either smaller than the Confidence Min limit or larger than the Confidence Max limit, it is simply discarded (not used at all), and the value in its associated register is unchanged. This is done to eliminate noise spikes. Confidence limits can only be used if not all of the measurement range is used. Values of 0 for Confidence Min and 4095 for Confidence Max will effectively disable the confidence limits.

#### Slope limitation

After a new sample has passed the Confidence limit checks, its value is compared with the last filtered value in its associated register. If the difference between the old and the new value is larger than the Max Slope Limit, the new value is modified to be exactly the old value plus or minus the Max Slope Limit. This limits the speed of change on the signal. Since the samples come at fixed intervals of 10 mS, it is easy to determine the number of Volts per millisecond. A value of 4095 will effectively disable slope limitation.

#### Filtering

After a new sample has both passed the confidence limits checks and has been validated with respect to the slope limitation, it is combined with the last filtered value by taking a part of the new sample and a part of the old filtered value, adding them together and writing the result back to the final destination register – one of the registers 81...88. For instance a filter value of 14 would take 14/64 of the new sample plus 50/64 of the old value. A filter of 64 would simply copy the new sample to the rule, thus disabling the filtering. This completes the filtering of the analogue inputs.

## 2.3

# Analogue Inputs

---

### Confidence alarms

If either of the Confidence Min or Confidence Max limits is used, it may be possible that no new samples are accepted, which means that the filtered value will never change even though there is a change in the input voltage. For instance, if the Confidence Min limit is set to 2.0 V, and the actual input voltage is 1.50 V, the filtered value may continue to read out 0.00 V (or the last value it had before exceeding the confidence limits).

To help troubleshooting in cases like this, each input has a status bit that is set if at least half of the new samples during the last second lie outside either confidence limit. It is not possible to see which of the confidence limits is violated. The status bits are updated once per second.

### Slope alarms

If the Max Slope limit is used (by setting its value lower than 4095), it may be possible that many samples have their value limited. This is not necessarily an error in itself, but can be a sign of a fault causing a noisy signal, or it can be a sign that the Max Slope limit is set too low, which can have implications if the analogue voltage is used to control the motor speed, torque, etc.

To help troubleshooting in cases like this, each input has a status bit that is set if at least half of the new samples during the last second were limited by the Max Slope setting. The status bits are updated once per second.

### Example of analogue input filter operation:

Note that even though the examples use units rather than Volts, decimal values are used, since the motor uses a much higher resolution internally to store the units.

Also note that as long as the slope limitation is in effect, the result will keep a constant slope even when using a filter. When the slope limitation is no longer in effect, the filter will cause the value to approach the final result more slowly as it approaches the result.

Confidence Min = 0, Confidence Max = 500, Max Slope = 10, Filter = 8, Old filtered value = 0.

Sample 1 = 100      Confidence OK, slope limit to 0 + 10 = 10,  
result =  $100 \cdot (8/64) + 0 \cdot (56/64) = 1.25$  units.

Sample 2 = 100      Confidence OK, slope limit to 1.25 + 10 = 11.25,  
result =  $11.25 \cdot (8/64) + 1.25 \cdot (56/64) = 2.5$  units.

Sample 3 = 100      Confidence OK, slope limit to 2.5 + 10 = 12.5,  
result =  $12.5 \cdot (8/64) + 2.5 \cdot (56/64) = 3.75$  units.

Sample 4 = 800      Confidence error, keep old value, result = **3.75** units.

...and so on until the result gets  $\sim = 95.0$  units...

Sample 78 = 100      Confidence OK, no slope limitation needed,  
result =  $100 \cdot (8/64) + 95 \cdot (56/64) = 95.625$  units.

Sample 79 = 100      Confidence OK, no slope limitation needed,  
result =  $100 \cdot (8/64) + 95.625 \cdot (56/64) \sim = 96.171875$  units.

Sample 80 = 100      Confidence OK, no slope limitation needed,  
result =  $100 \cdot (8/64) + 96.171875 \cdot (56/64) \sim = 96.65$  units.

Sample 81 = 100      Confidence OK, no slope limitation needed,  
result =  $100 \cdot (8/64) + 96.65 \cdot (56/64) \sim = 97.07$  units.

## 2.3

## Analogue Inputs

---

Sample 82 = 100    Confidence OK, no slope limitation needed,  
result =  $100*(8/64)+97.07*(56/64) \sim = 97.44$  units.

Sample 83 = 100    Confidence OK, no slope limitation needed,  
result =  $100*(8/64)+97.44*(56/64) \sim = 97.76$  units.

... The following samples produce the following results ending up with the input value (100.0).

98.04, 98.28, 98.49, 98.68, 98.85, 99.00, 99.12, 99.23, 99.33, 99.41, 99.48, 99.55, 99.60,  
99.65, 99.70, 99.74, 99.77, 99.80, 99.82, 99.84, 99.86, 99.88, 99.90, 99.91, 99.92, 99.93,  
99.94, 99.95, 99.95, 99.96, 99.96, 99.97, 99.97, 99.98, 99.98, 99.98, 99.98, 99.99, 99.99,  
99.99, .....100.0

## 2.4

# User Outputs

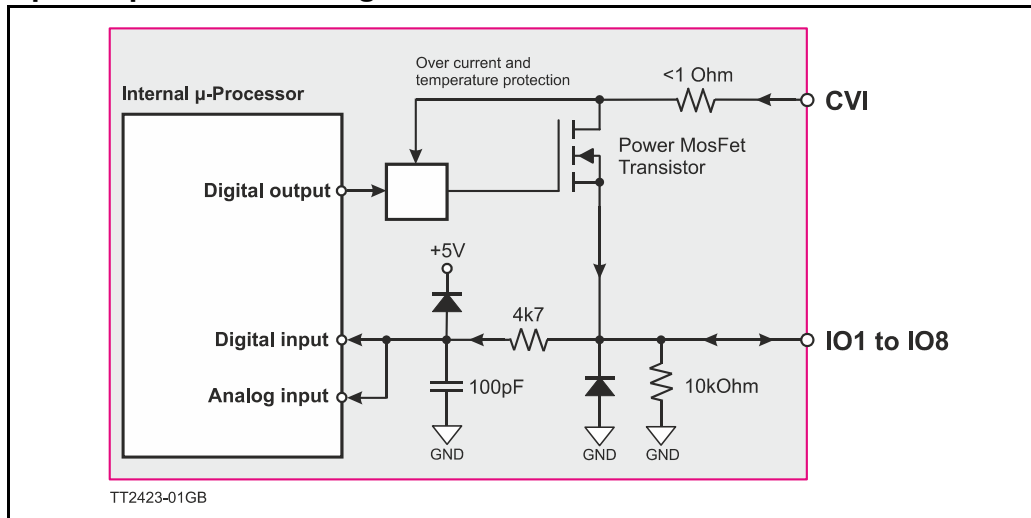
### 2.4.1 User outputs

The MIS motors has 8 inputs/outputs (IO's) that each can be set individually to input, output or analogue input 0-5V via MacTalk or software commands. This means that it for example is possible to have 4 inputs, 3 outputs and one analogue input.



**Please notice:** The number of available I/O terminals available may differ depending at which motor type and connector configuration you are using. Please consult the chapter [Connector overview for the MIS motors](#), page 34

#### Input/output functional diagram:

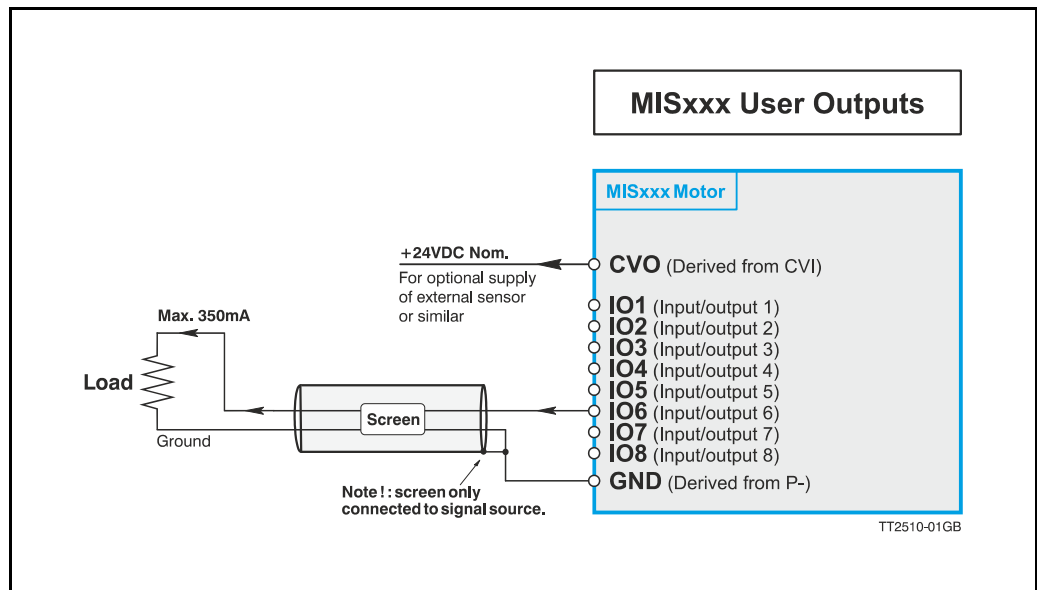


- The Outputs are Source outputs and 5-28VDC compliant
- No galvanic isolation
- Short-circuit to ground protected that shuts down all outputs and sets Error bit in software.
- Optional “In Position” and “Error” signals can be selected to be on any outputs 1 to 8
- Optional Encoder outputs
- 300 mA output current per channel even with all channels fully loaded at the same time.
- Internal ground clamp diodes to protect when inductive load is driven.



## 2.4

# User Outputs



### 2.4.2 General

The Controller is equipped with a total of 8 digital outputs. Each output can be used for a variety of purposes depending on the Controller's basic mode of operation. The Outputs are not optically isolated from other Controller circuitry. The output circuitry is powered from the internal power supply CVI. The output circuitry operates with voltages in the range 5-28VDC. Each output can supply a continuous current up to 350mA. The outputs are all source drivers, i.e. if a given output is activated, contact is made between the control voltage (CVI) and the respective output terminal. See above illustration.

### 2.4.3 Overload of User Outputs

All of the outputs are short-circuit protected, which means that the program and the motor is stopped and the output is automatically disconnected in the event of a short circuit. The output will first function normally again when the short-circuit has been removed.

**Note:** Do not connect a voltage greater than 30VDC to the CVI terminal as the output circuitry may be seriously damaged and the unit will require factory repair.

If one or more outputs are short circuited, MacTalk will show Error "Output Driver" and Bit 2 will be set in Err\_Bits.  
See also [Err\\_Bits](#), page 163.

## 2.5 Serial interfaces overview

---

### 2.5.1 Serial interfaces

The Controller has 2 serial interfaces:

- RS485 (Dual channel A and B) balanced for up to 32 units in multi-axis applications and Modbus communication. (Standard)
- CANbus -CANopen DS-301. Fully ISO 11898-2:2016 compliant
- CANbus - CANopen DSP-402 is in development but not available now.

CANbus and RS485 can be used at the same time.



**Please notice:** The number of available I/O terminals available may differ depending at which motor type and connector configuration you are using. Please consult the chapter [Connector overview for the MIS motors](#), page 34

## 2.6

# RS485 Interface

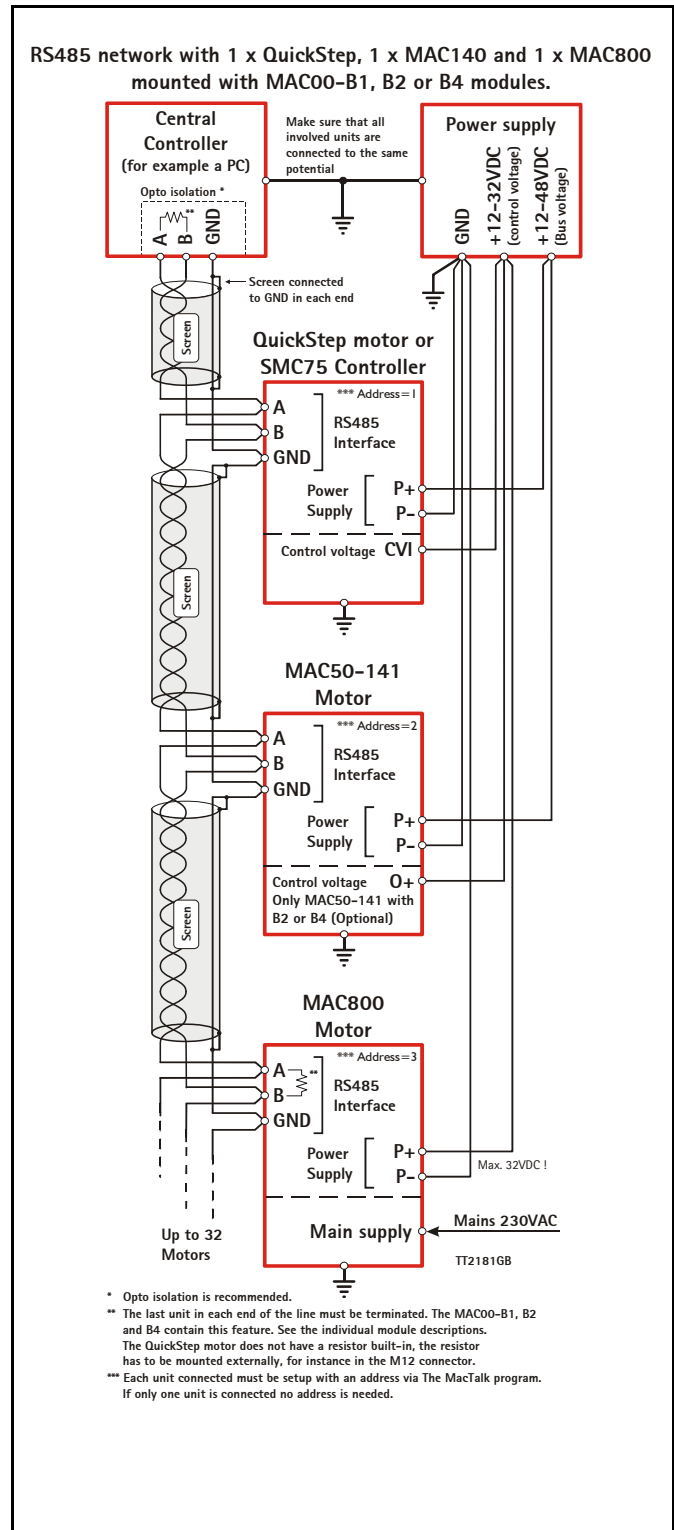
### 2.6.1 RS485 - General description when using a QuickStep motor

The RS485 interface offers more noise immune communication compared to a USB or RS232 interface. Up to 32 motors can be connected to the same interface bus.

When connecting the RS485 interface to a central controller, the following rules must be followed:

- 1 Use twisted pair cable.
- 2 Use shielded cable.
- 3 Make sure that the GND 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 last unit in each end of the network must be terminated with a 120 Ohm resistor between A and B.
- 6 Ensure that the supply lines are made individually in order to reduce the voltage drop between the motors.
- 7 Central Controller RS485 interface:  
If available, it is strongly recommended a type with optical isolation is used.

The default configuration:  
Data bits = 8  
Baud rate = 19200  
Stop bit = 1  
Parity = None



## 2.7

## EMC considerations

---

### 2.7.1 EMC considerations

The MIS family of motors eliminates the traditional problems with noise from long motor cables that emit noise and feedback cables that are sensitive to noise from external sources.

However, it is still necessary to be aware of noise problems with communications cables and the 8 general-purpose inputs and outputs.

Whenever a digital signal changes level quickly, a noise spike is generated, and is transferred to the other wires in the same cable, and to a lesser degree to wires in other cables located close to the cable with the switching signal. A typical example is when a digital output from the MIS motor changes from low to high to drive a relay. If this digital output signal is transmitted in a multi-wire cable together with the RS-485 signals, there is a high risk that the RS-485 signal will be affected to the extent that the communication will fail, and require software retries.

If communication is used during operation, and operation includes either digital input signals or digital output signals, some precautions must be taken to avoid noise problems. The following sections describe a number of measures which can be taken to solve noise problems. In most installations, no special measures will be required, but if noise problems are experienced – and/or must be avoided – it is highly recommended the instructions below are followed.

### 2.7.2 Use short cables

The shorter a cable is, the less noise problems it will induce. Be sure to keep the cables as short as possible. Instead of curling up the cables, cut them off at the minimum required length.

### 2.7.3 Use separate cables

Avoid running digital signals in the same multi-wire cables as RS-485 communication signals.

On some models of the MIS motors, the same connector contains both RS-485 signals and I/O signals – typically the I/Os 1-4.

In many applications, far from all inputs and outputs are used. If only up to four I/Os are required, consider using only I/Os 5-8 which are typically available via another connector on the motor.

### 2.7.4 Use filters

If more than 4 I/Os are needed, consider using I/Os 1-4 for inputs and I/Os 5-8 for outputs. It is normally possible to install a hardware filter on the digital input signals before they enter the cable. With such a (good) filter, noise on the RS-485 signals will not be a problem.

It is also possible to use filters on the outputs, but it is more difficult. It can be done by using short cables from the motor to the filters, and then using longer cables from the filters to the output targets. It may be easier to use a short cable from the motor to a splitter box, and then split the I/Os in one cable and the RS-485 signals in another cable.

### 2.7.5 Use termination (resistors) on the RS-485 signals

RS-485 is typically used to connect a single master PC or PLC to one or more motors in a chain. Both ends of the chain must have a 120 Ohms termination resistor connected between the A- and B+ signals. There is typically a terminating resistor in the master PC or PLC, but there is no termination inside the motors. Therefore an external resistor must be connected at the end of the cable out of the last motor in the chain. If the last motor has no connection cable, a connector with a resistor soldered between the A- and B+ pins should be used.

## 2.7

## EMC considerations

---

As an alternative, a connector with a short cable can be used with the resistor soldered between the two wires carrying A- and B+.

Use individually shielded cables.

In some installations, it will be necessary to have RS-485 signals in the same multi-wire cables as fast-switching digital signals. In addition to keeping cable lengths to a minimum and using termination resistors, high-quality cables, where each wire is shielded from the other wires in the cable, should be used. This is typically done using a metal foil wrapped around each wire. These types of cables are more expensive, but the overall cost and noise immunity requirements may justify the solution instead of splitting cables.

### 2.7.6 Use simple shielding

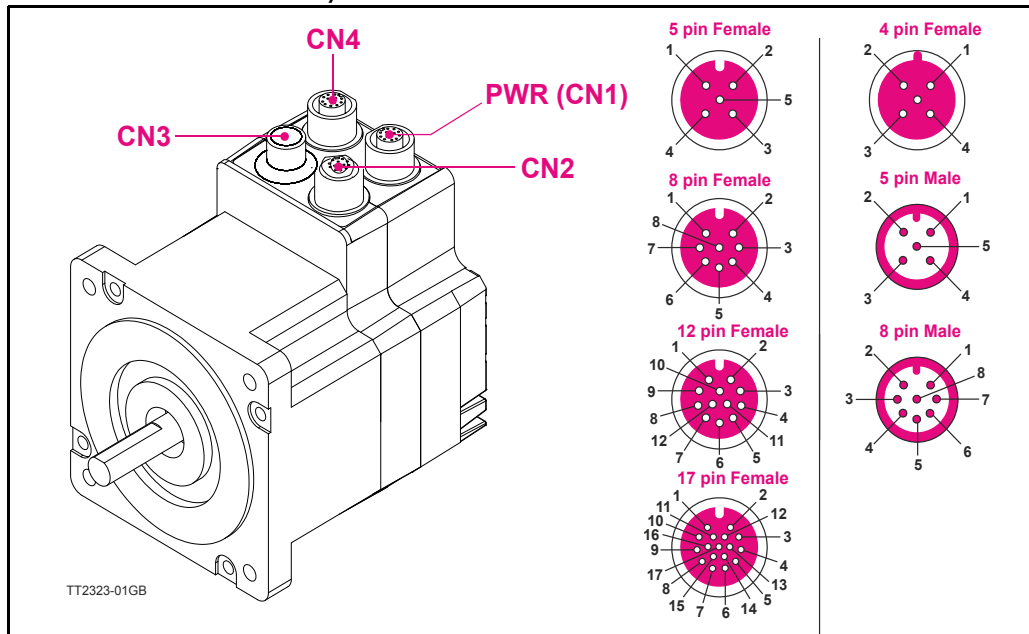
Using cables with only a single shield shared by all the signal wires will also improve noise problems to some degree, but will not guarantee completely stable operation for mixed signal cables. If a cable carries only RS-485 or only digital I/O, this simple and inexpensive form of shielding is recommended.

## 2.8 How to connect a MIS motor

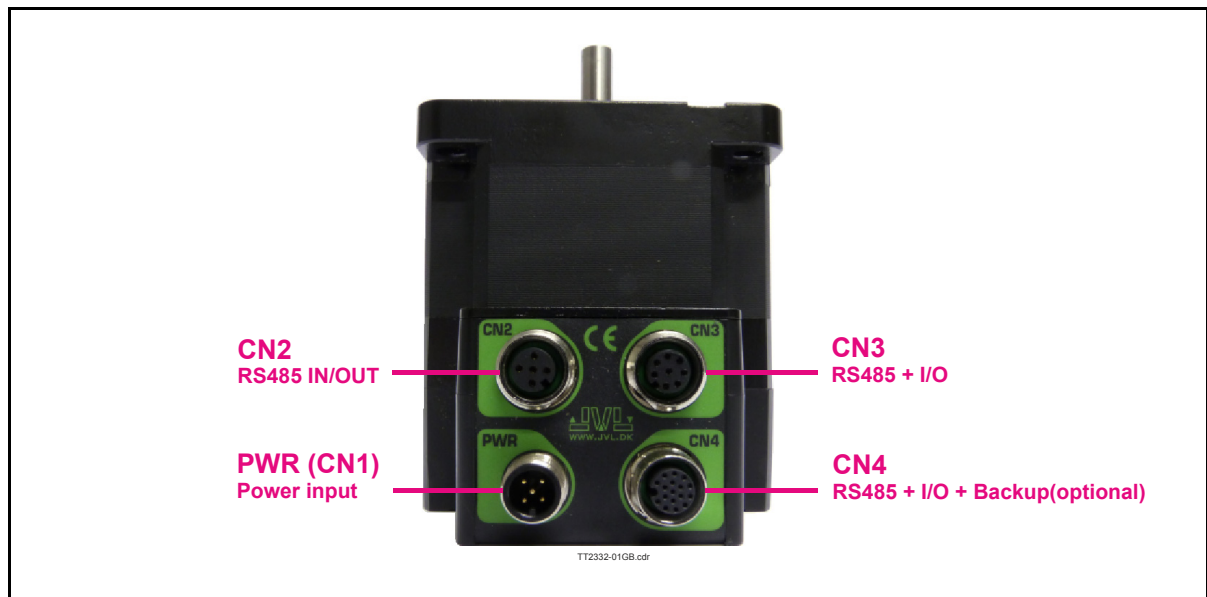
### 2.8.1 Connector overview for the MIS motors

QUICKSTEP Connector Overview	Power Male 5Pin	IO1-8, RS485, MFIO Female 17Pin	RS485 Female 5Pin	RS485 + IO1-4 Female 8Pin	RS485 + IO1-4 Female 8Pin	CANopen Female 5Pin	SSI Encoder Male 8Pin	Profibus Male 5Pin	Ethernet Female 4Pin
Connector ID	PWR (CN1)	CN4	CN2	CN3	CN2	CN2 & CN3	CN3	CN2 & CN3	CN2 & CN3
MISxxxnyy <b>Q5</b> zz85 (8IOA) <i>Preferred type</i>	x	x	x	x					
MISxxxnyy <b>P6</b> zz85 (CAN-open)	x	x				x			
MISxxxnyy <b>Q9</b> zz85 (SSI input)	x	x			x		x		
MISxxxnyy <b>Ex</b> zz85 (Ethernet)	x	x							x
MISxxxnyy <b>FB</b> zz85 (Bluetooth)	x	x		x	x				
MISxxxnyy <b>FP</b> zz85 (Profibus)	x	x						x	
M12 Pin1	P+ (12-72VDC)	IO1	B0+ (RS485)	IO1	IO1	CAN_SHLD	IO5 Zero Set	5VDC	TX0_P
M12 Pin2	P+ (12-72VDC)	GND	A0- (RS485)	IO2	IO2	Unused	IO6 CNTDIR	A-	RX0_P
M12 Pin3	P- (GND)	IO2	B0+ (RS485)	IO3	IO3	CAN_GND	A+ (Clock+)	DGND	TX0_N
M12 Pin4	CVI (12-28VDC)	IO3	A0- (RS485)	GND	GND	CAN_H	GND	B+	RX0_N
M12 Pin5	P- (GND)	B1- (RS422)	GND	B0-(RS485)	B0-(RS485)	CAN_L	B- (Data In-)	SHIELD	-
M12 Pin6	-	IO4	-	A0+(RS485)	A0+(RS485)	-	B+ (Data In+)	-	-
M12 Pin7	-	A1- (RS422)	-	IO4	IO4	-	A- (Clock -)	-	-
M12 Pin8	-	B1+ (RS422)	-	CVO (out)	CVO (out)	-	CVO (out)	-	-
M12 Pin9	-	CVO (out)	-	-	-	-	-	-	-
M12 Pin10	-	A1+ (RS422)	-	-	-	-	-	-	-
M12 Pin11	-	IO5	-	-	-	-	-	-	-
M12 Pin12	-	IO6	-	-	-	-	-	-	-
M12 Pin13	-	IO7	-	-	-	-	-	-	-
M12 Pin14	-	IO8	-	-	-	-	-	-	-
M12 Pin15	-	A0+(RS485)	-	-	-	-	-	-	-
M12 Pin16	-	GND	-	-	-	-	-	-	-
M12 Pin17	-	B0-(RS485)	-	-	-	-	-	-	-
M12 Connector solder terminals	WI1008-M12F5SS1	(not available)	WI1008-M12M5SS1	WI1008-M12M8SS1	WI1008-M12M8SS1	WI1008-M12M5SS1	WI1008-M12F8SS1	WI1028-M12F5SS1	(not available)
M12 Cables 5m	WI1000-M12F5T05N	WI1009-M12M17T05N	WI1005-M12M8V M5V03N	WI1000-M12M8T05N	WI1000-M12M8T05N	WI1006-M12F5 TM5T05N	WI1000-M12F8T05N	WI1026-M12-F5S0R	WI1046-M12M4S05R

Connector layout - The shown motor is a MIS34x motor but the connector locations are the same at other MIS family members with radial standard connectors.



## 2.8 How to connect a MIS motor



### 2.8.2 MISxxxxxxQ5xxxx connector description.

The MIS motors offers robust M12 connectors which makes it ideal for automation applications. The M12 connectors offer solid mechanical protection and are easy operate. Following scheme gives the relevant information about each connector and the pins, wire colours and a short description of the signals available.

The connector layout:

"PWR" (CN1) - Power input. M12 - 5pin male connector				
Signal name	Description	Pin no.	JVL Cable W11000-M12F5TxxN	Isolation group
P+	Main supply +12-72VDC. Connect with pin 2 *	1	Brown	1
P+	Main supply +12-72VDC. Connect with pin 1 *	2	White	1
P-	Main supply ground. Connect with pin 5 *	3	Blue	1
CVI	Control and user output supply +12-30VDC. <b>DO NOT connect &gt;30V to this terminal!</b>	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)

## 2.8

# How to connect a MIS motor

<b>“CN2” - RS485 IN/OUT. M12 - 5pin female connector.</b>				
Signal name	Description	Pin no.	JVL Cable WI1000-M12 M5TxxN	Isolation group (See note)
RS485: B+	RS485 interface. Leave open if unused	1	Brown	1
RS485: A-	RS485 interface. Leave open if unused	2	White	1
RS485: B+	RS485 interface. Leave open if unused	3	Blue	1
RS485: A-	RS485 interface. Leave open if unused	4	Black	1
GND	Ground intended to be used together with the other signals in this connector	5	Grey	1
<b>“CN3” - RS485 + I/O connector - M12 - 8pin female connector.</b>				
Signal name	Description	Pin no.	JVL Cable WI1000-M12 M8TxxN	Isolation group (See note)
IO1	I/O channel 1. Can be used as input or output	1	White	1
IO2	I/O channel 2. Can be used as input or output	2	Brown	1
IO3	I/O channel 3. Can be used as input or output	3	Green	1
GND	Ground intended to be used together with the other signals in this connector	4	Yellow	1
RS485: B+	RS485 interface. Leave open if unused	5	Grey	1
RS485: A-	RS485 interface. Leave open if unused	6	Pink	1
IO4	I/O channel 4. Can be used as input or output	7	Blue	1
CVO	Supply output. Connected internally to the <b>CVI</b> terminal in the PWR connector. <b>DO NOT connect &gt;30V to this terminal!</b> USB interface. Supply input 5VDC nominal	8	Red	1
<b>“CN4” - RS485 + I/O + Backup (option) connector - M12 - 17pin female connector</b>				
Signal name	Description	Pin no.	JVL Cable WI1009M12 M17TxxN	Isolation group (see note)
IO1	I/O channel 1. Can be used as input or output	1	Brown	1
GND	Ground intended to be used together with the other signals in this connector	2	Blue	1
IO2	I/O channel 2. Can be used as input or output	3	White	1
IO3	I/O channel 3. Can be used as input or output	4	Green	1
B1-	RS422 I/O terminal B-	5	Pink	1
IO4	I/O channel 4. Can be used as input or output	6	Yellow	1
A1-	RS422 I/O terminal A-	7	Black	1
B1+	RS422 I/O terminal B+	8	Grey	1
CVO	Supply output. Connected internally to the <b>CVI</b> terminal in the PWR connector. <b>DO NOT connect &gt;30V to this terminal!</b>	9	Red	1
A1+	RS422 I/O terminal A+	10	Violet	1
IO5	I/O channel 5. Can be used as input or output	11	Grey/pink	1
IO6	I/O channel 6. Can be used as input or output	12	Red/blue	1
IO7	I/O channel 7. Can be used as input or output	13	White/Green	1
IO8	I/O channel 8. Can be used as input or output	14	Brown/Green	1
RS485: B+	RS485 interface. Leave open if unused	15	White/Yellow	1
GND/ EXTBACKUP	Only for motors installed with the H3 option (absolute multi turn encoder). This terminal can be connected to an external supply. Connect to ground if not used.	16	Yellow/brown	1
RS485: A-	RS485 interface. Leave open if unused	17	White/grey	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.				



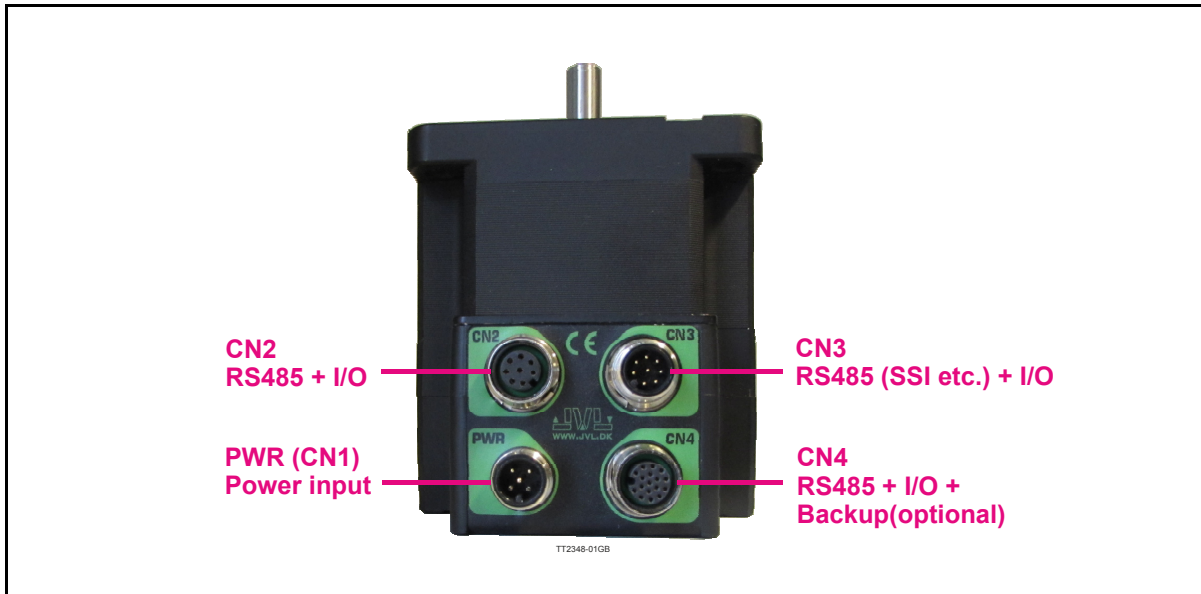
## 2.8 How to connect a MIS motor

### 2.8.3 Cables for the MISxxxxxxQ5xxxx

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

MISxxx Connectors				Description	JVL Order no.	Picture
"PWR" 5pin Male	"CN2" 5pin Female	"CN3" 8pin Female	"CN4" 17pin Female			
	X			RS485 Interface cable. Connects directly from the MIS motor to a RS485 comport. Length: 5m (197 inch)	RS485-M12-1-5-5	
		X		RS485 Interface cable. Connects directly from the MIS motor to a RS485 comport. Length: 5m (197 inch)	RS485-M12-1-5-8	
Not relevant. The RS485-USB-ATC-820 connect to CN2 through cable type RS485-M12-1-5-8				USB to RS485 Converter. To be used if no RS485 COM port is available.	RS485-USB-ATC-820	
	X			Cable (Ø5.5mm) with M12 male 5-pin connector loose wire ends 0.35mm <sup>2</sup> (22AWG) and foil screen. Length: 5m (197 inch)	WI1000-M12M5T05N	
	X			Same as above but 20m (787 inch)	WI1000-M12M5T20N	
		X		Cable with M12 male 8-pin connector loose wire ends 0.35mm <sup>2</sup> (22AWG) and screen.	WI1000-M12M8T05N	
		X		Same as above but 20m (787 inch)	WI1000-M12M8T20N	
			X	Cable with M12 male 17-pin connector loose wire ends 0.22mm <sup>2</sup> (24AWG) and screen. Length: 5m (197 inch)	WI1009-M12M17S05N	
			X	Same as above but 20m (787 inch)	WI1009-M12M17S20N	
			X	Junction box for splitting the 17 pin I/O connector into 4 independent connectors. Include also 9 LED's for monitoring the I/O status and communication. Cable length: 0,5m (20 inch)	PA0190	
<b>Protection caps. Optional if connector is not used to protect from dust / liquids.</b>						
	X	X	X	IP67 protection cap for M12 female connector.	WI1000-M12FCAP1	
X				IP67 protection cap for M12 male connector.	WI1000-M12MCAP1	

## 2.8 How to connect a MIS motor



### 2.8.4 MISxxxxxxQ9xxxx connector description.

The MIS motors offers robust M12 connectors which makes it ideal for automation applications. The M12 connectors offer solid mechanical protection and are easy to operate.

The following scheme gives the relevant information about each connector and the pins, wire colours and a short description of the signals available.

The connector layout:

"PWR" (CN1) - Power input. M12 - 5pin male connector				
Signal name	Description	Pin no.	JVL Cable WI1000- M12F5TxxN	Isolation group
P+	Main supply +12-72VDC. Connect with pin 2 *	1	Brown	1
P+	Main supply +12-72VDC. Connect with pin 1 *	2	White	1
P-	Main supply ground. Connect with pin 5 *	3	Blue	1
CVI	Control and user output supply +12-30VDC. <b>DO NOT connect &gt;30V to this terminal!</b>	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)

## 2.8

## How to connect a MIS motor

<b>"CN2" - RS485 + I/O connector - M12 - 8pin female connector.</b>				
Signal name	Description	Pin no.	JVL Cable WI1000-M12 M8TxN	Isolation group (See note)
IO1	I/O channel 1. Can be used as input or output	1	White	1
IO2	I/O channel 2. Can be used as input or output	2	Brown	1
IO3	I/O channel 3. Can be used as input or output	3	Green	1
GND	Ground intended to be used together with the other signals in this connector	4	Yellow	1
RS485: B+	RS485 interface. Leave open if unused	5	Grey	1
RS485: A-	RS485 interface. Leave open if unused	6	Pink	1
IO4	I/O channel 4. Can be used as input or output	7	Blue	1
CVO	Supply output. Connected internally to the CVI terminal in the PWR connector.	8	Red	1
<b>"CN3" - RS485 (SSI etc.) + I/O. M12 - 8pin Male connector.</b>				
Signal name	Description	Pin no.	JVL Cable WI1000-M12 F8TxN	Isolation group (See note)
IO5	Used for zero set. Leave open if unused	1	White	1
IO6	Counting direction. Leave open if unused	2	Brown	1
RS485: A+	Clock +. Leave open if unused	3	Green	1
GND	Signal ground. Leave open if unused	4	Yellow	1
RS485: B-	Data in -. Leave open if unused	5	Grey	1
RS485: B+	Data in +. Leave open if unused	6	Pink	1
RS485: A-	Clock -. Leave open if unused	7	Blue	1
CVO	Supply output. Connected internally to the CVI terminal in the PWR connector.	8	Red	1
<b>"CN4" - RS485 + I/O + Backup<sub>(option)</sub> connector - M12 - 17pin female connector</b>				
Signal name	Description	Pin no.	JVL Cable WI1009M12 M17TxN	Isolation group (see note)
IO1	I/O channel 1. Can be used as input or output	1	Brown	1
GND	Ground intended to be used together with the other signals in this connector	2	Blue	1
IO2	I/O channel 2. Can be used as input or output	3	White	1
IO3	I/O channel 3. Can be used as input or output	4	Green	1
B1-	RS422 I/O terminal B-	5	Pink	1
IO4	I/O channel 4. Can be used as input or output	6	Yellow	1
A1-	RS422 I/O terminal A-	7	Black	1
B1+	RS422 I/O terminal B+	8	Grey	1
CVO	Supply output. Connected internally to the CVI terminal in the PWR connector.	9	Red	1
A1+	RS422 I/O terminal A+	10	Violet	1
IO5	I/O channel 5. Can be used as input or output	11	Grey/pink	1
IO6	I/O channel 6. Can be used as input or output	12	Red/blue	1
IO7	I/O channel 7. Can be used as input or output	13	White/Green	1
IO8	I/O channel 8. Can be used as input or output	14	Brown/Green	1
RS485: B+	RS485 interface. Leave open if unused	15	White/Yellow	1
GND/EXTBACKUP	Only for motors installed with the H3 option (absolute multiturn encoder). This terminal can be connected to an external supply. Connect to ground if not used.	16	Yellow/brown	1
RS485: A-	RS485 interface. Leave open if unused	17	White/grey	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.				

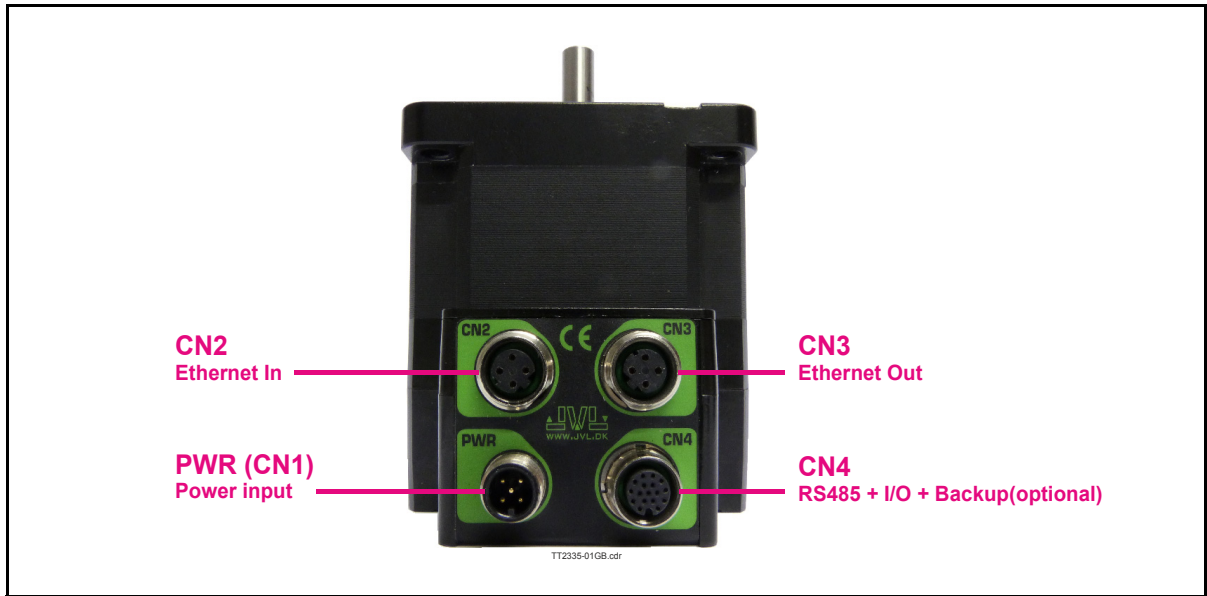
## 2.8 How to connect a MIS motor

### 2.8.5 Cables for the MISxxxxxx**Q9**xxxx

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

MISxxx Connectors				Description	JVL Order no.	Picture
"PWR" 5pin Male	"CN2" 8pin Female	"CN3" 8pin Male	"CN4" 17pin Female			
X				Cable (Ø5.5mm) with M12 <b>female</b> 5-pin connector loose wire ends 0.35mm <sup>2</sup> (22AWG) and foil screen. Length: 5m (197 inch)	WI1000-M12F5T05N	
	X			RS485 Interface cable. Connects directly from the MIS motor to a RS485 comport. Length: 5m (197 inch)	RS485-M12-1-5-8	
Not relevant. The RS485-USB-ATC-820 connect to CN2 through cable type RS485-M12-1-5-8				USB to RS485 Converter. To be used if no RS485 COM port is available. Use also RS485-M12-1-5-8	RS485-USB-ATC-820	
	X			Cable (Ø5.5mm) with M12 <b>male</b> 8-pin connector loose wire ends 0.35mm <sup>2</sup> (22AWG) and foil screen. Length: 5m (197 inch)	WI1000-M12M8T05N	
	X			Same as above but 20m (787 inch)	WI1000-M12M8T20N	
		X		Cable with M12 <b>female</b> 8-pin connector loose wire ends 0.22mm <sup>2</sup> (24AWG) and screen. Length: 5m (197 inch)	WI1000-M12F8T05N	
		X		Same as above but 20m (787 inch)	WI1000-M12F8T20N	
			X	Cable with M12 <b>male</b> 17-pin connector loose wire ends 0.22mm <sup>2</sup> (24AWG) and screen. Length: 5m (197 inch)	WI1009-M12M17S05N	
			X	Same as above but 20m (787 inch)	WI1009-M12M17S20N	
			X	Junction box for splitting the 17 pin I/O connector into 4 independent connectors. Include also 9 LED's for monitoring the I/O status and communication. Cable length: 0,5m (20 inch)	PA0190	
<b>Protection caps. Optional if connector is not used to protect from dust / liquids.</b>						
	X		X	IP67 protection cap for M12 <b>female</b> connector.	WI1000-M12FCAP1	
X		X		IP67 protection cap for M12 <b>male</b> connector.	WI1000-M12MCAP1	

## 2.8 How to connect a MIS motor



### 2.8.6 MISxxxxxxExxxxx connector description.

Hardware wise all the MIS motors with the Ethernet option are equal and offer the connectivity shown in the table below.

The following Ethernet protocols are supported in this moment:

- MISxxxxxxEPxxxx : ProfiNet
- MISxxxxxxEIxxxx : EtherNetIP
- MISxxxxxxECxxxx : EtherCAT
- MISxxxxxxELxxxx : Ethernet POWERLINK
- MISxxxxxxEMxxxx : Modbus TCP
- MISxxxxxxESxxxx : Sercos III

The MIS motors offers robust M12 connectors which makes it ideal for automation applications. The M12 connectors offer solid mechanical protection and are easy operate. Following scheme gives the relevant information about each connector and the pins, wire colours and a short description of the signals available.

The connector layout:

"PWR" (CN1) - Power input. M12 - 5pin male connector				
Signal name	Description	Pin no.	JVL Cable W11000-M12F5TxxN	Isolation group
P+	Main supply +12-72VDC. Connect with pin 2 *	1	Brown	1
P+	Main supply +12-72VDC. Connect with pin 1 *	2	White	1
P-	Main supply ground. Connect with pin 5 *	3	Blue	1
CVI	Control and user output supply +12-30VDC. <b>DO NOT connect &gt;30V to this terminal !</b>	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)

## 2.8 How to connect a MIS motor

<b>“CN2” - Ethernet In port connector - M12 - 4pin female connector “D” coded</b>				
Signal name	Description	Pin no.	JVL Cable W11046-M12M4S05R	Isolation group (See note)
Tx0_P	Ethernet Transmit channel 0 - positive terminal	1	-	2
Rx0_P	Ethernet Receive channel 0 - positive terminal	2	-	2
Tx0_N	Ethernet Transmit channel 0 - negative terminal	3	-	2
Rx0_N	Ethernet Receive channel 0 - negative terminal	4	-	2
<b>“CN3” - Ethernet Out port connector. M12 - 4 pin female connector “D” coded</b>				
Signal name	Description	Pin no.	JVL Cable W11046-M12M4S05R	Isolation group (see note)
Tx1_P	Ethernet Transmit channel 1 - positive terminal	1	-	3
Rx1_P	Ethernet Receive channel 1 - positive terminal	2	-	3
Tx1_N	Ethernet Transmit channel 1 - negative terminal	3	-	3
Rx1_N	Ethernet Receive channel 1 - negative terminal	4	-	3
<b>“CN4” - RS485 + I/O + Backup (option) connector - M12 - 17pin female connector</b>				
Signal name	Description	Pin no.	JVL Cable W11009M12 M17TxN	Isolation group (see note)
IO1	I/O channel 1. Can be used as input or output	1	Brown	1
GND	Ground intended to be used together with the other signals in this connector	2	Blue	1
IO2	I/O channel 2. Can be used as input or output	3	White	1
IO3	I/O channel 3. Can be used as input or output	4	Green	1
B1-	RS422 I/O terminal B-	5	Pink	1
IO4	I/O channel 4. Can be used as input or output	6	Yellow	1
A1-	RS422 I/O terminal A-	7	Black	1
B1+	RS422 I/O terminal B+	8	Grey	1
CVO	Supply output. Connected internally to the CVI terminal in the PWR connector. <b>DO NOT connect &gt;30V to this terminal !</b>	9	Red	1
A1+	RS422 I/O terminal A+	10	Violet	1
IO5	I/O channel 5. Can be used as input or output	11	Grey/Pink	1
IO6	I/O channel 6. Can be used as input or output	12	Red/Blue	1
IO7	I/O channel 7. Can be used as input or output	13	White/Green	1
IO8	I/O channel 8. Can be used as input or output	14	Brown/Green	1
RS485: B+	RS485 interface. Leave open if unused	15	White/Yellow	1
GND/ EXTBACKUP	Only for motors installed with the H3 option (absolute multi turn encoder). This terminal can be connected to an external supply. Connect to ground if not used.	16	Yellow/Brown	1
RS485: A-	RS485 interface. Leave open if unused	17	White/Grey	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.				

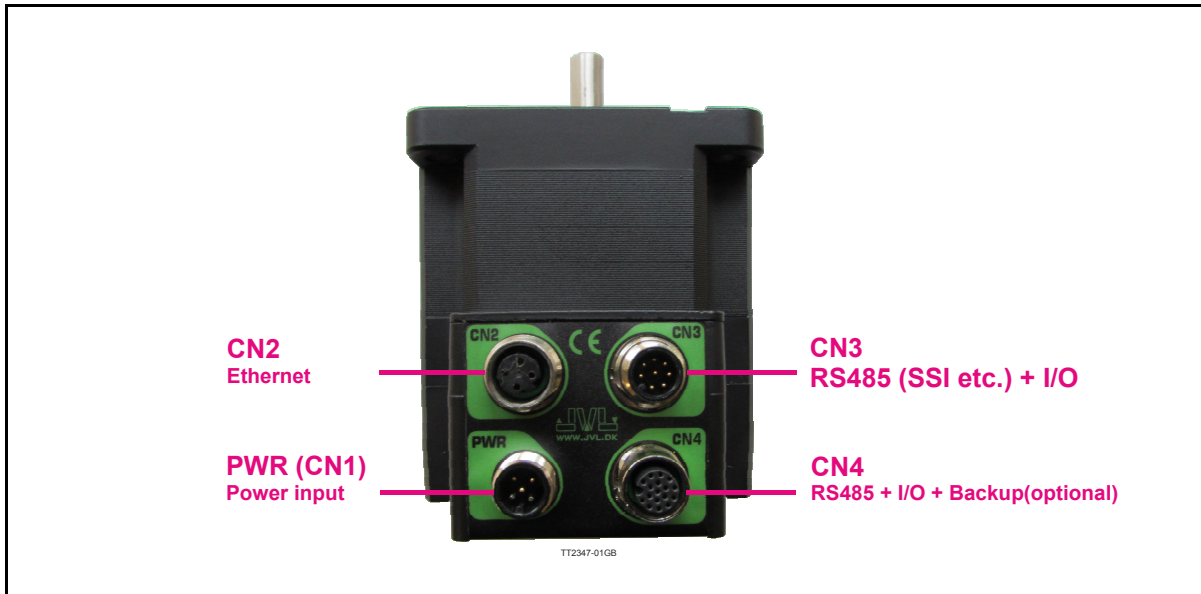
## 2.8 How to connect a MIS motor

### 2.8.7 Cables for the MISxxxxxxExxxxx

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

MIS34x Connectors				Description	JVL Order no.	Picture
"PWR" 5pin Male	"CN2" 5pin Female	"CN3" 8pin Female	"CN4" 17pin Female			
			X	RS485 Interface cable. Connects directly from The MIS motor to a RS485 comport. Length: 5m (197 inch)	RS485-M12-1-5-17S	
X				Cable (Ø5.5mm) with M12 male 5-pin connector loose wire ends 0.35mm <sup>2</sup> (22AWG) and foil screen. Length: 5m (197 inch)	WI1000-M12F5T05N	
Not relevant. The RS485-USB-ATC-820 connect to CN2 through cable type RS485-M12-1-5-8				USB to RS485 Converter. To be used if no RS485 COM port is available.	RS485-USB-ATC-820	
	X	X		Cable (Ø5.5mm) with M12 male D-coded 4-pin connector loose wire ends 0.35mm <sup>2</sup> (22AWG) and foil screen. Length: 5m (197 inch)	WI1046-M12M4S05R	
	X	X		Same as above but 15m (590 inch)	WI1046-M12M4S15R	
	X	X		Cable (Ø5.5mm) with M12 male D-coded 4-pin connector and RJ45 connector. Length: 5m (197 inch)	WI1046-M12M4S05-NRJ45	
This cable only exist in 5m length						
			X	Cable with M12 male 17-pin connector loose wire ends 0.22mm <sup>2</sup> (24AWG) and screen. Length: 5m (197 inch)	WI1009-M12M17S05N	
			X	Same as above but 20m (787 inch)	WI1009-M12M17S20N	
<b>Protection caps. Optional if connector is not used to protect from dust / liquids.</b>						
	X	X	X	IP67 protection cap for M12 female connector.	WI1000-M12FCAP1	
X				IP67 protection cap for M12 male connector.	WI1000-M12MCAP1	

## 2.8 How to connect a MIS motor



### 2.8.8 MISxxxxxxE(2-7)xxxx connector description.

Hardware wise all the MIS motors with the Ethernet option are equal and offer the connectivity shown in the table below.

The following Ethernet protocols are supported in this moment:

- MISxxxxxxE2xxxx : EtherCAT
- MISxxxxxxE3xxxx : EtherNet/IP
- MISxxxxxxE4xxxx : Ethernet POWERLINK
- MISxxxxxxE5xxxx : Modbus TCP
- MISxxxxxxE6xxxx : ProfiNet
- MISxxxxxxE7xxxx : Sercos III

The MIS motors offers robust M12 connectors which makes it ideal for automation applications. The M12 connectors offer solid mechanical protection and are easy operate. Following scheme gives the relevant information about each connector and the pins, wire colours and a short description of the signals available.

The connector layout:

"PWR" (CN1) - Power input. M12 - 5pin male connector				
Signal name	Description	Pin no.	JVL Cable WI1000-M12F5TxxN	Isolation group
P+	Main supply +12-72VDC. Connect with pin 2 *	1	Brown	1
P+	Main supply +12-72VDC. Connect with pin 1 *	2	White	1
P-	Main supply ground. Connect with pin 5 *	3	Blue	1
CVI	Control and user output supply +12-30VDC. <b>DO NOT connect &gt;30V to this terminal !</b>	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)



## 2.8

## How to connect a MIS motor

<b>“CN2” - Ethernet In port connector - M12 - 4pin female connector “D” coded</b>				
Signal name	Description	Pin no.	JVL Cable WI1046-M12M4S05R	Isolation group (See note)
Tx0_P	Ethernet Transmit channel 0 - positive terminal	1	-	2
Rx0_P	Ethernet Receive channel 0 - positive terminal	2	-	2
Tx0_N	Ethernet Transmit channel 0 - negative terminal	3	-	2
Rx0_N	Ethernet Receive channel 0 - negative terminal	4	-	2
<b>“CN3” - Ethernet Out port connector. M12 - 4 pin female connector “D” coded</b>				
Signal name	Description	Pin no.	JVL Cable WI1046-M12M4S05R	Isolation group (see note)
Tx1_P	Ethernet Transmit channel 1 - positive terminal	1	-	3
Rx1_P	Ethernet Receive channel 1 - positive terminal	2	-	3
Tx1_N	Ethernet Transmit channel 1 - negative terminal	3	-	3
Rx1_N	Ethernet Receive channel 1 - negative terminal	4	-	3
<b>“CN4” - RS485 + I/O + Backup (option) connector - M12 - 17pin female connector</b>				
Signal name	Description	Pin no.	JVL Cable WI1009M12 M17TxxN	Isolation group (see note)
IO1	I/O channel 1. Can be used as input or output	1	Brown	1
GND	Ground intended to be used together with the other signals in this connector	2	Blue	1
IO2	I/O channel 2. Can be used as input or output	3	White	1
IO3	I/O channel 3. Can be used as input or output	4	Green	1
B1-	RS422 I/O terminal B-	5	Pink	1
IO4	I/O channel 4. Can be used as input or output	6	Yellow	1
A1-	RS422 I/O terminal A-	7	Black	1
B1+	RS422 I/O terminal B+	8	Grey	1
CVO	Supply output. Connected internally to the CVI terminal in the PWR connector. <b>DO NOT connect &gt;30V to this terminal !</b>	9	Red	1
A1+	RS422 I/O terminal A+	10	Violet	1
IO5	I/O channel 5. Can be used as input or output	11	Grey/Pink	1
IO6	I/O channel 6. Can be used as input or output	12	Red/Blue	1
IO7	I/O channel 7. Can be used as input or output	13	White/Green	1
IO8	I/O channel 8. Can be used as input or output	14	Brown/Green	1
RS485: B+	RS485 interface. Leave open if unused	15	White/Yellow	1
GND/ EXTBACKUP	Only for motors installed with the H3 option (absolute multi turn encoder). This terminal can be connected to an external supply. Connect to ground if not used.	16	Yellow/Brown	1
RS485: A-	RS485 interface. Leave open if unused	17	White/Grey	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.				

## 2.8 How to connect a MIS motor

### 2.8.9 Cables for the MISxxxxxxE6xxxx

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

MIS34x Connectors				Description	JVL Order no.	Picture
"PWR" 5pin Male	"CN2" 5pin Female	"CN3" 8pin Female	"CN4" 17pin Female			
			X	RS485 Interface cable. Connects directly from The MIS motor to a RS485 comport. Length: 5m (197 inch)	RS485-M12-1-5-17S	
X				Cable (Ø5.5mm) with M12 male 5-pin connector loose wire ends 0.35mm <sup>2</sup> (22AWG) and foil screen. Length: 5m (197 inch)	WI1000-M12F5T05N	
Not relevant. The RS485-USB-ATC-820 connect to CN2 through cable type RS485-M12-1-5-8				USB to RS485 Converter. To be used if no RS485 COM port is available.	RS485-USB-ATC-820	
	X	X		Cable (Ø5.5mm) with M12 male D-coded 4-pin connector loose wire ends 0.35mm <sup>2</sup> (22AWG) and foil screen. Length: 5m (197 inch)	WI1046-M12M4S05R	
	X	X		Same as above but 15m (590 inch)	WI1046-M12M4S15R	
	X	X		Cable (Ø5.5mm) with M12 male D-coded 4-pin connector and RJ45 connector. Length: 5m (197 inch)	WI1046-M12M4S05-NRJ45	
This cable only exist in 5m length						
			X	Cable with M12 male 17-pin connector loose wire ends 0.22mm <sup>2</sup> (24AWG) and screen. Length: 5m (197 inch)	WI1009-M12M17S05N	
			X	Same as above but 20m (787 inch)	WI1009-M12M17S20N	
<b>Protection caps. Optional if connector is not used to protect from dust / liquids.</b>						
	X	X	X	IP67 protection cap for M12 female connector.	WI1000-M12FCAP1	
X				IP67 protection cap for M12 male connector.	WI1000-M12MCAP1	

## 2.8 How to connect a MIS motor

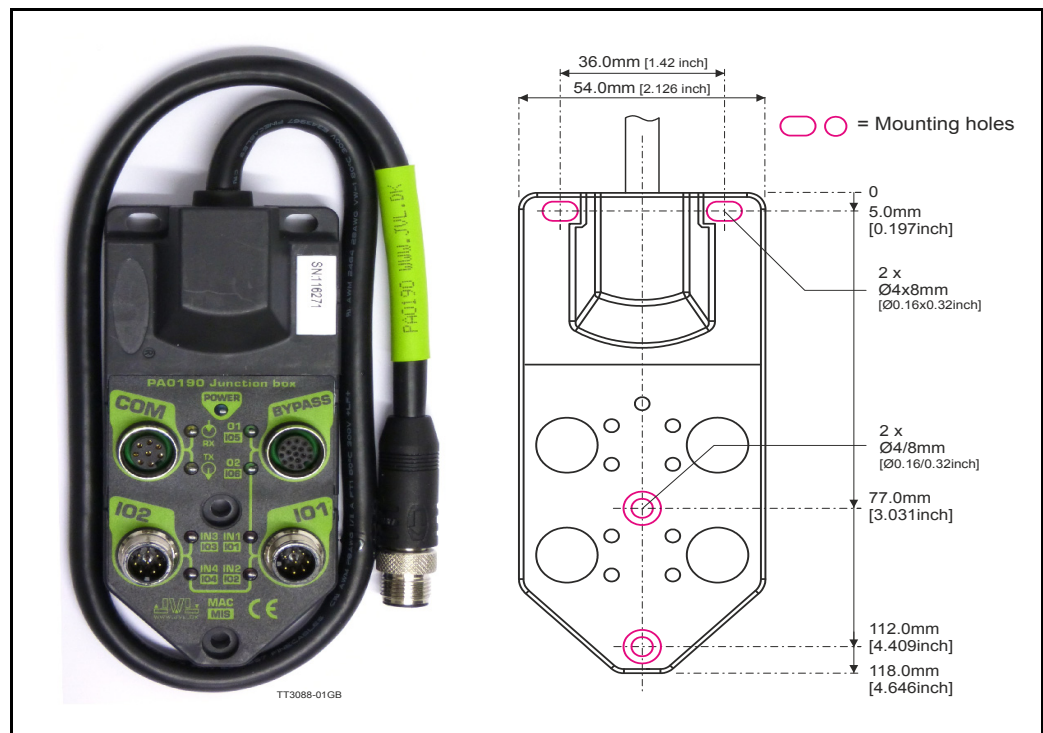
### 2.8.10 Drawing and description of PA0190

Junction box that splits the signals in the MIS motors “CN4” 17 pin I/O connector into 4 individual connectors giving an easy and more flexible installation.

#### Usage hints:

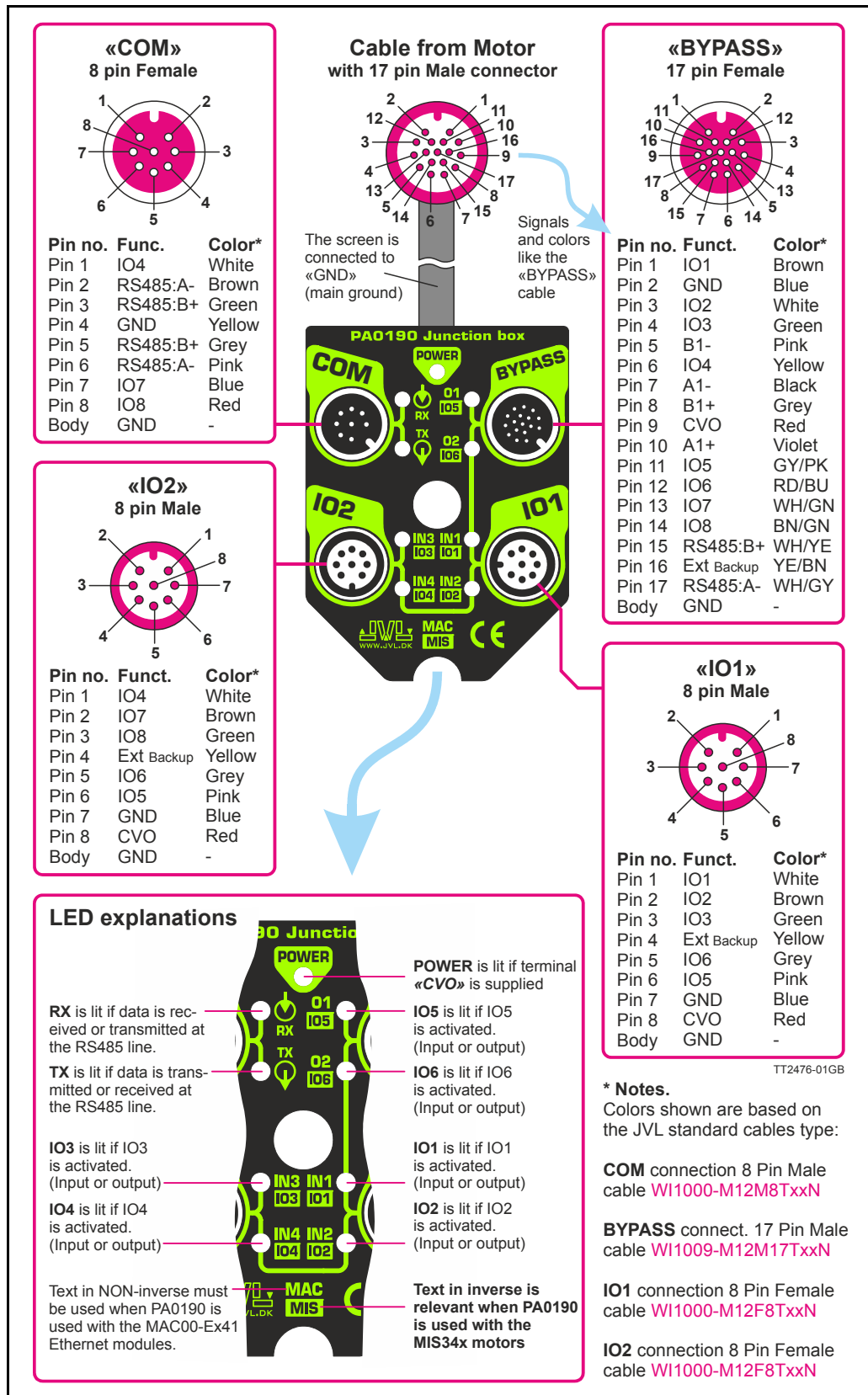
The LED's will only work with a MIS motor where the OUT+ and IO- is supplied from the CN4 connector. See also the I/O description for the module.

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.



## 2.8 How to connect a MIS motor

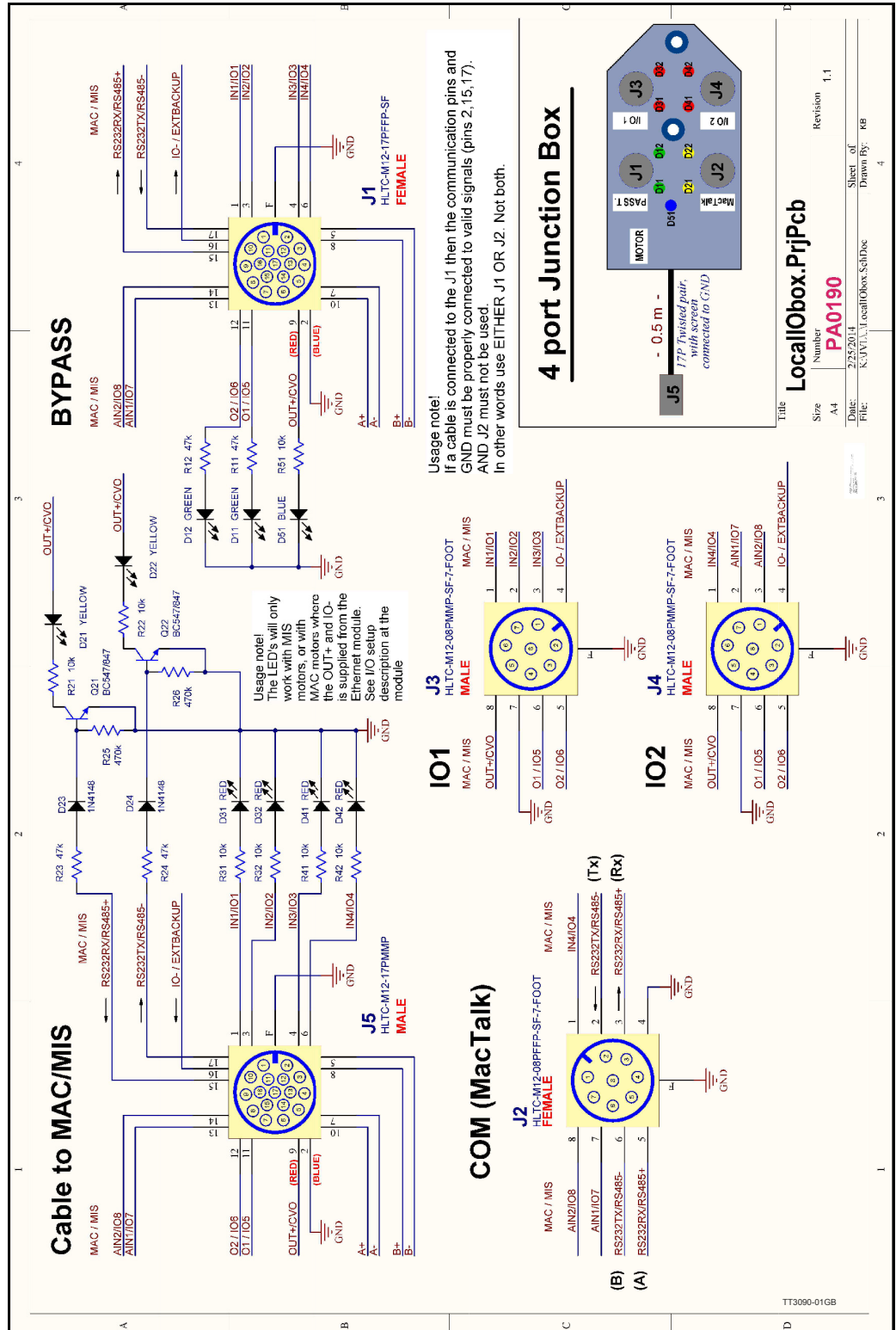
Terminal and LED description of the PA0190 Junction box.



# 2.8

# How to connect a MIS motor

Diagram of the internal details in the PA0190 Junction Box.



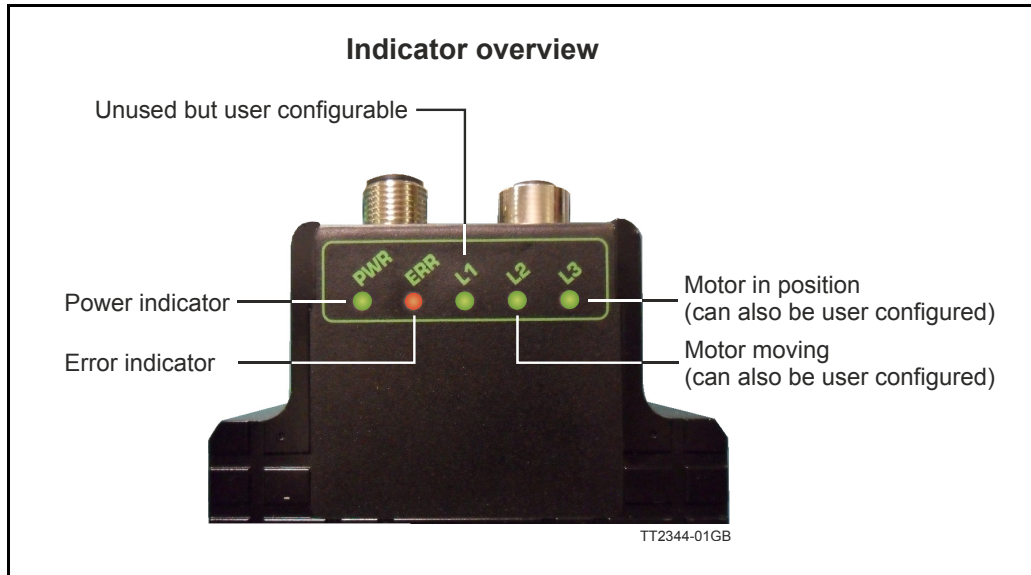
## 2.9

# LED indicators basic motor

### 2.9.1 LED's - description for products without Ethernet or CANopen.

This description covers all MIS motors with basic configuration without any Ethernet or CANopen such as MISxxxxxxQ5xxxx, MISxxxxxxQ9xxxx, MISxxxxxxFBxxxx, or MISxxxxxxEWxxxx.

The LED's are used for indicating states and faults.



#### LED indicator descriptions (default setup)

LED Text	Colour	Constant off	Constant on	Blinking
L1	Green	Default	Only when user configured	Only when user configured
L2	Green	Motor not moving	Motor moving	-
L3	Green	Motor not in position	Motor in position	-
ERR	Red	No error	-	Error
PWR	Green	Power is not applied	Power is applied to both motor and module. Only <b>MIS17x</b> and <b>MIS23x</b> : The LED will lit red constant-ly if the supply is too low.	-

L1, Red, shows by default the Status bit I5: Closed loop lead/lag detected. It tells if the motor is overloaded, which can be caused of too low Running current or too heavy load.

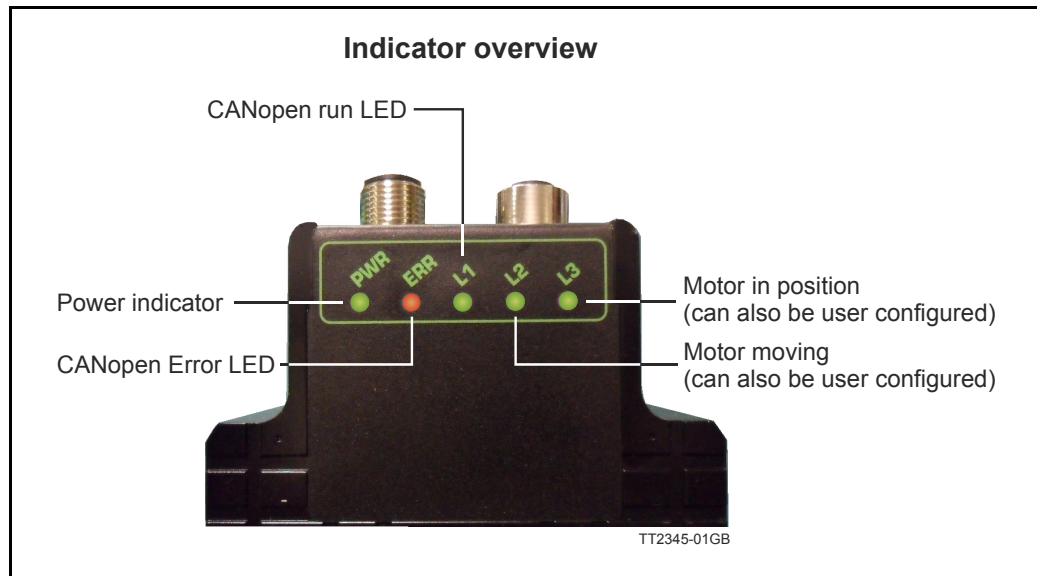
L1 to L3 can be configured to show the status of a almost any single bit from a user defined register. Please see [FlexLEDSetup1](#), page 191 for the details.

## 2.10 LED indicators using CANopen

### 2.10.1 Indicator LED's - description for products with CANopen.

This description covers all MISxxx products with build in CANopen option such as MIS34xxxxP6xxxx.

The LED's are used for indicating states and faults.



LED indicator descriptions (default setup)

LED Text	Function	Colour	Constant off	Constant on	Blinking
L1	CANopen run LED	Green	Please see below and optionally the DS303-3 standard		
L2	Motor moving	Green	Motor not moving	Motor moving	-
L3	Motor in position	Green	Motor not in position	Motor in position	-
ERR	CANopen error LED	Red	Please see below and Please see the DS303-3 standard		
PWR	Power	Green	Power is not applied.	Power is applied to both motor and module. Only <b>MIS17x</b> and <b>MIS23x</b> : The LED will lit red constantly if the supply is too low.	-

L2 to L3 can be configured to show the status of a almost any single bit from a user defined register. Please see [FlexLEDSetup1](#) , page 191 for the details.

## 2.11 LED indicators using Ethernet

---

### 2.11.1 Indicator LED's - description for products with Ethernet.

The MIS motors offers optional 6 different Ethernet protocols.  
These are:

- EtherCAT
- ModbusTCP
- Profinet
- Powerlink
- EthernetIP
- SercosIII

This manual do only cover description of how to connect.

Concerning LED indicators, software and protocol setup and usage please consult a separate manual that can be found at [www.jvl.dk](http://www.jvl.dk) using this link: [www.jvl.dk](http://www.jvl.dk)

The LED descriptions are in the chapters "Commissioning" for each protocol.



### **3 Hardware None-intelligent products**

---

JVL have a range of stepper motors with an integrated stepper driver (none programmable) that are cost effective and easy to use.

JVL also offer a range of stand alone stepper drivers in the working range 3 to 9 ARMS output current and 24 to 150VDC supply.

**Please contact your JVL distributor for further information.**





# 4.1 Using the MacTalk software

**Setup save/open**  
The complete setup can be either saved or reloaded from a file using these buttons

**Startup mode**  
The basic functionality of the unit is setup in this field.

**Profile Data**  
All the main parameters for controlling the motor behaviour are setup in this field.

**Driver Parameters**  
These fields are used to define standby and running current.

**Gear Factor**  
The gear ratio can be entered here

**Motion Parameters**  
The distance the motor has to run is entered here

**System control**  
Use these buttons to save data permanently, reset the motor etc.

**Error Handling**  
Use these fields to define error limits for the position range etc.

**Motor status**  
This field shows the actual motor load, position and speed etc.

**Run status**  
Status of the motor. The supply voltage, tempere etc.

**Inputs**  
The status of the digital and analogue inputs are shown here.

**Outputs**  
The status of the outputs are shown here and can be activated by the cursor

**Errors**  
If a fatal error occurs, information will be displayed here.

**Warnings**  
Warnings are shown here

**Help Line**  
Left area:  
If values entered are outside their normal values, errors are shown here.  
Right area:  
Here it is possible to see if a motor is connected, the type, version and serial no.

**Zero Search**  
All the parameters regarding the position zero search can be specified here.

**Under voltage handling**  
Setup how motor should react if supply voltage is too low

**Communication**  
The actual address of the motor can be entered here

TT2145GB

## 4.1.1 MacTalk introduction

The MacTalk software is the main interface for setting up the MIS motor for a specific application.

The program offers the following features:

- Selection of operating mode of the MIS motor.
- Changing main parameters such as speed, motor current, zero search type, etc.
- Monitoring in real time the actual motor parameters, such as supply voltage, input status, etc.
- Changing protection limits such as position limits.
- Saving all current parameters to PC.
- Restoring all parameters from PC.
- Saving all parameters permanently in the motor.
- Updating the motor firmware or MacTalk software from the internet or a file.

The main window of the program changes according to the selected mode, thus only showing the relevant parameters for operation in the selected mode.

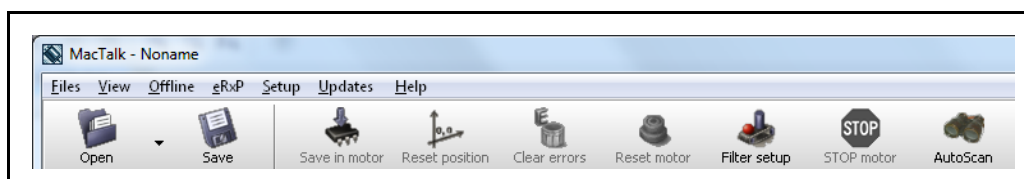
The following pages describe the actual window for each mode and how the parameters affect MIS motor operation.

## 4.1 Using the MacTalk software

---

### 4.1.2 Toolbar description

The toolbar at the top of MacTalk contains the most commonly used features.



#### **Open**

Opens a setup file from PC and downloads the setup to the motor. If no motor is connected, the setup is shown in MacTalk and can be edited and saved to the PC.

#### **Save**

Saves the actual setup from the motor to a file. If no motor is connected, the actual off-line settings (including module setups and program) are saved.

#### **Save in motor**

The complete actual setup in the basic motor will be saved permanently in the flash memory. If the motor is powered down or reset, the saved setup will be used.

#### **Reset position**

Resets the position counter to 0. The content of the position counter can be monitored in the right side of the main screen as "Actual position".

#### **Clear errors**

Clears all the errors (if any). Please note that if an error is still present, the motor will remain in the actual error state.

#### **Reset motor**

Reset the motor. Same as performing a power off / on operation.

#### **Filter Setup**

For specifying the filter setup of the analogue inputs.

#### **STOP motor**

Stops the motor immediately using a controlled deceleration ramp and puts the motor into passive mode. If a program is present this is stopped as well.

This button shall be considered a functional stop button and is available using the keyboard shortcut F8.

Pressing the "Stop" button will immediately stop the motor by changing the currently running mode to "passive" using a fast controlled deceleration curve.

Using a quickstep motor or a module that enables the user to execute RxP programs this execution is also halted to prevent the motor from starting up if a startup-mode is setup from a program.

**Warning! Do not consider this button as an appropriate Emergency stop. Always fit an Emergency stop circuitry to your motor setup.**

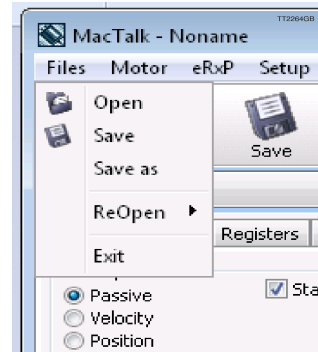
#### **Auto Scan**

If the actual COM port is not known or the motor is setup with an address different from default the Auto Scan feature can help finding the motor(s).

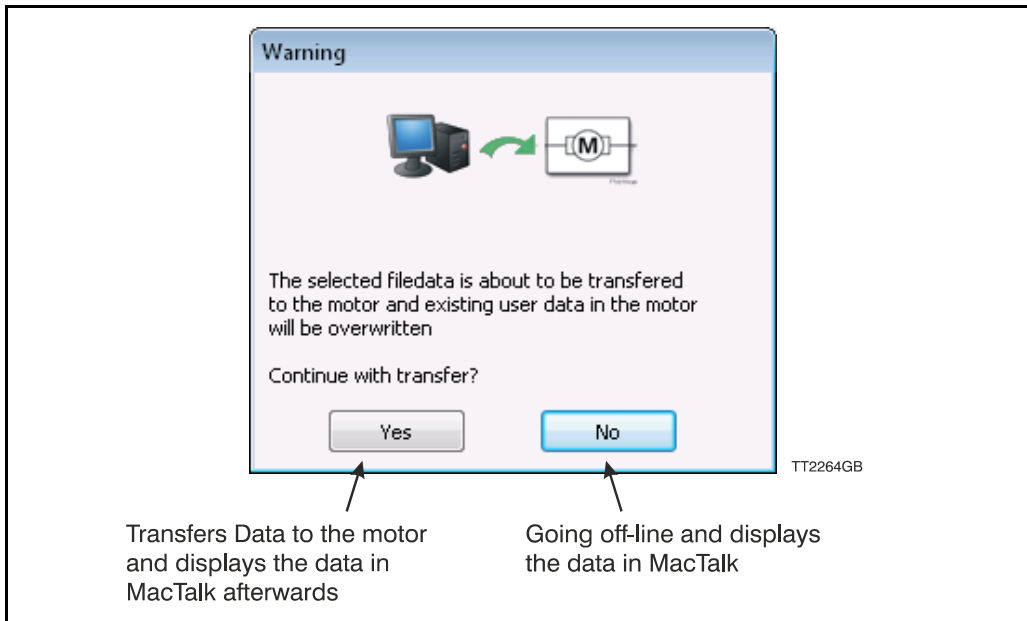
# 4.1 Using the MacTalk software

## 4.1.3 Saving or opening a setup file to/from PC

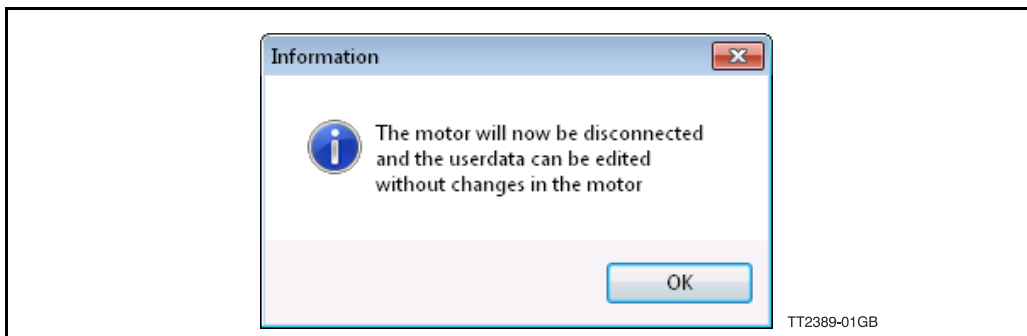
The complete motor setup can be saved to PC or opened from PC and transferred to the motor. Saving and opening a file over a network is also possible. The setup files use the extension MAC. By default, the setup files are saved in the same directory in which MacTalk itself is also installed. Other directories can be selected.



In case a motor is present and a PC file is opened the user is prompted for keeping the connection or going offline and displaying the file-content. The following message box appears.



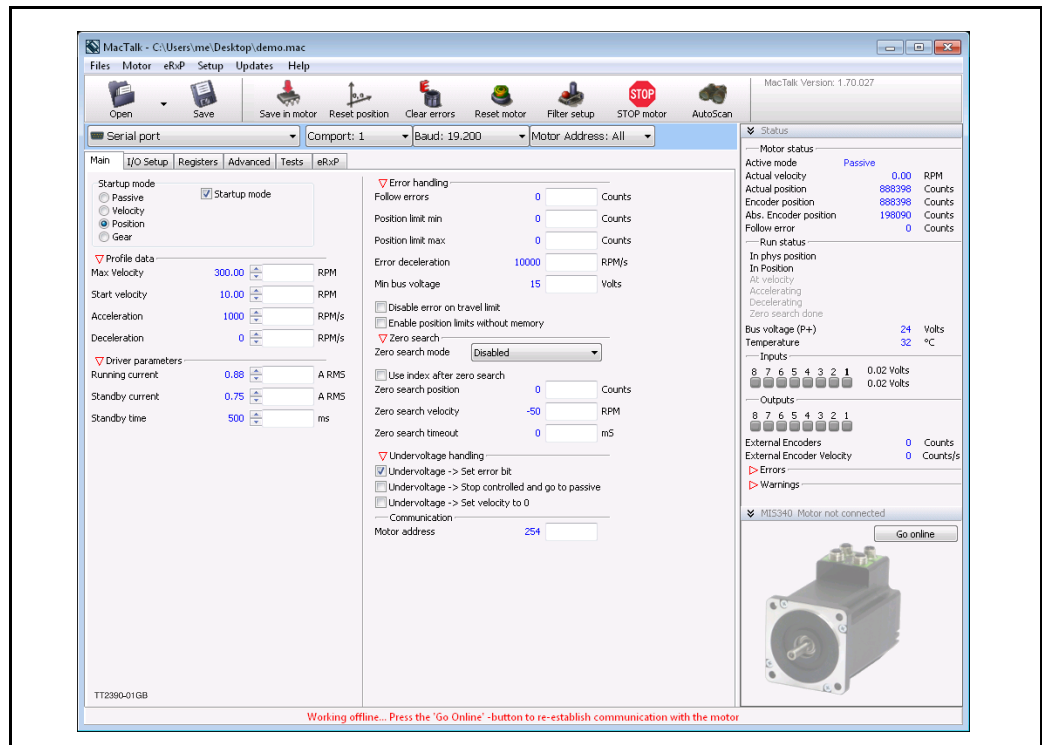
If the user decides to go offline the following text box is presented.



Pressing "OK" disconnects the motor from the PC-application and all data can be edited without any interruption in the motor.

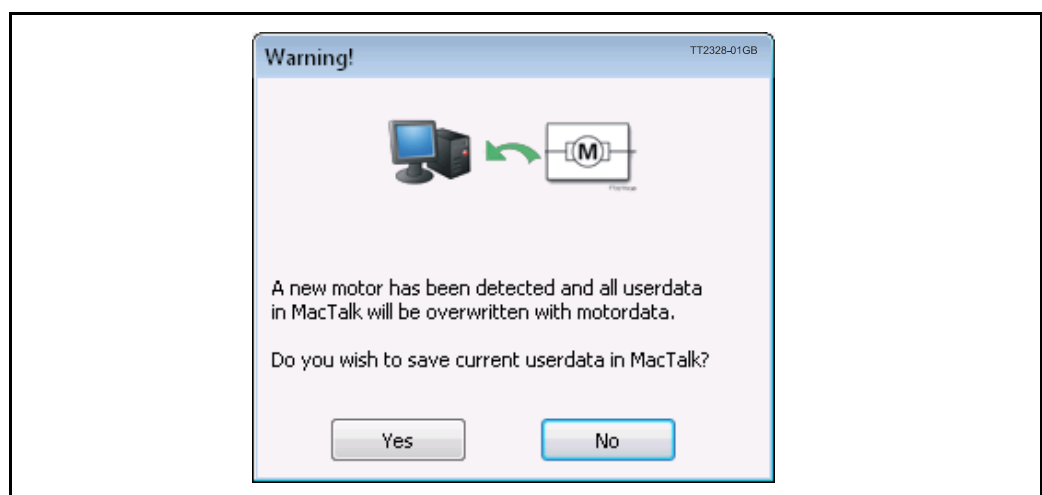
## 4.1 Using the MacTalk software

The following MacTalk view is presented.



As seen in the bottom info line, the motor is disconnected and the file data is currently present in Mactalk. To re-establish communication with the motor, simply press the "Go Online" -button and if any data has been changed a warning box appears enabling the user to save current data before re-establishing communication with the motor as this will overwrite existing data in MacTalk.

If data is changed in MacTalk the user is warned that current data in MacTalk may be overwritten and needs to be saved. The following warning box is presented.



Choosing "No" will immediately upload all motor data, pressing "yes" will save all data in the open file.

# 4.1 Using the MacTalk software

## 4.1.4 Main Screen

The screenshot shows the MacTalk software interface with various tabs and fields. The 'Main' tab is active, displaying motor parameters such as Max. Velocity (100.00 RPM), Start velocity (10.00 RPM), Acceleration (1000 RPM/s), and Deceleration (0 RPM/s). The 'Motor parameters' section shows Position (888398 Counts) and Motor address (254). The 'Status' panel on the right shows Motor status (Active mode), Actual velocity (0.00 RPM), Actual position (888398 Counts), Encoder position (888398 Counts), Abs. Encoder position (198106 Counts), and Follower error (95 Counts). A motor image is shown at the bottom right.

a) This field shows the register values in the controller

b) Here it is possible to key in new values. After pressing enter the value will be transferred to the motor and thereafter be read again from the controller and be shown at point a. Because of digitalizing of the keyed in value, the returned value in a) can be different from the value in b).

c) By pressing the unit field it is possible to change between internal unit in the controller and the unit shown to the user. E.g. If user unit for current is ARMS and the internal unit is 5.87mA (300ARMS correspond to 511 units.) Not all registers have different internal and user unit. Speed for example is always specified in RPM.

## 4.1.5 I/O Setup tab

The screenshot shows the I/O Setup tab in MacTalk software. It displays a list of inputs (I01-I08) and outputs (O01-O08) with their respective active levels and types. The 'Dedicated Inputs' section shows selection for HM, NKL, and PL. The 'Dedicated Outputs' section shows selection for 'In position', 'In Physical Position', and 'Error' output. The 'Input filters' section shows a list of digital input filters (I01-I08) and a filter time constant of 5 ms. The 'Status' panel on the right shows Motor status (Active mode), Actual velocity (0.00 RPM), Actual position (888398 Counts), Encoder position (888398 Counts), Abs. Encoder position (198106 Counts), and Follower error (95 Counts). A motor image is shown at the bottom right.

Active level for the I/Os can be chosen to high or low on inputs

Dedicated Inputs Selection for Inputs HM, NKL and PL. An external encoder can also be selected here and defined as either quadrature or pulse/direction type. Selection if it shall be Inputs or Outputs

Dedicated Outputs Selection for outputs "In position", "In Physical Position", "Error" output. It can also be selected if the pulses generated shall be used internally, externally or both and which output should be used for pulse and direction signals

Selection of Inputs for HM, NL and PL

Selection of output for In-Position and Errors

Status of the inputs \*)

Status of the outputs

Input filters Here the filter for the digital inputs can be selected.

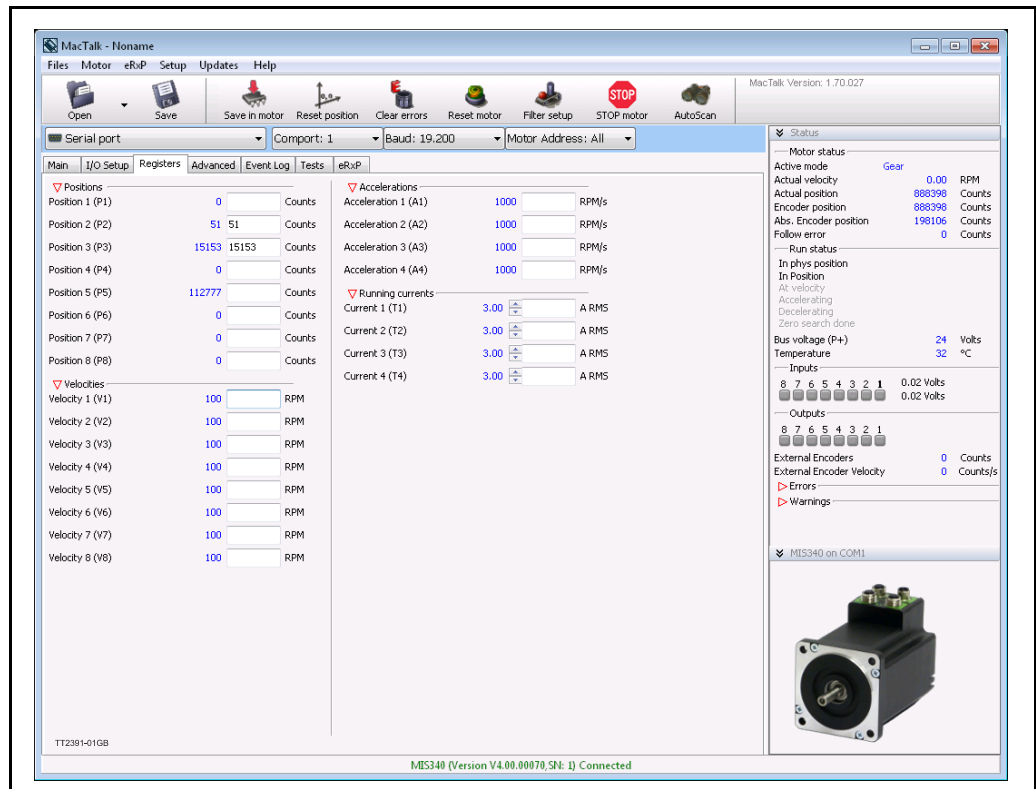
Filter time constant can be adjusted here. The same value is used for all inputs if filter inputs are enabled.

\*) The analogue value of certain inputs can be read. Click at the input lamp and the analogue value will be shown. The upper value is the actual value and the lower value the filtered value.



# 4.1 Using the MacTalk software

## 4.1.6 Register Screen



These registers can be used with FastMac commands. For example, the motor can run to position P2 using velocity V2, acceleration/deceleration A2, running current T2, using only a one byte command.

These values are not updated automatically. To update, place the cursor at the specific register value to the left of the box for new values, and click. Otherwise they only update at motor reset or power up.

# 4.1 Using the MacTalk software

## 4.1.7 Advanced Screen

If it is desired to run the motor in the opposite direction it can be done by marking "Invert motor direction"

When this field is marked the motor runs to the AP (Actual position) from the encoder position when the motor goes from passive to position mode

Remove the mark in this field and the motor will start the program at start-up

Zero search options can be selected here

It is possible to have a certain number of motors doing the same by giving them the same group id.

## 4.1.8 Test Screen

This screen is used for adjusting the Zero search sensor to the correct position when using the index pulse of an encoder. The index pulse should be in the green area. If not, the sensor has to be adjusted.

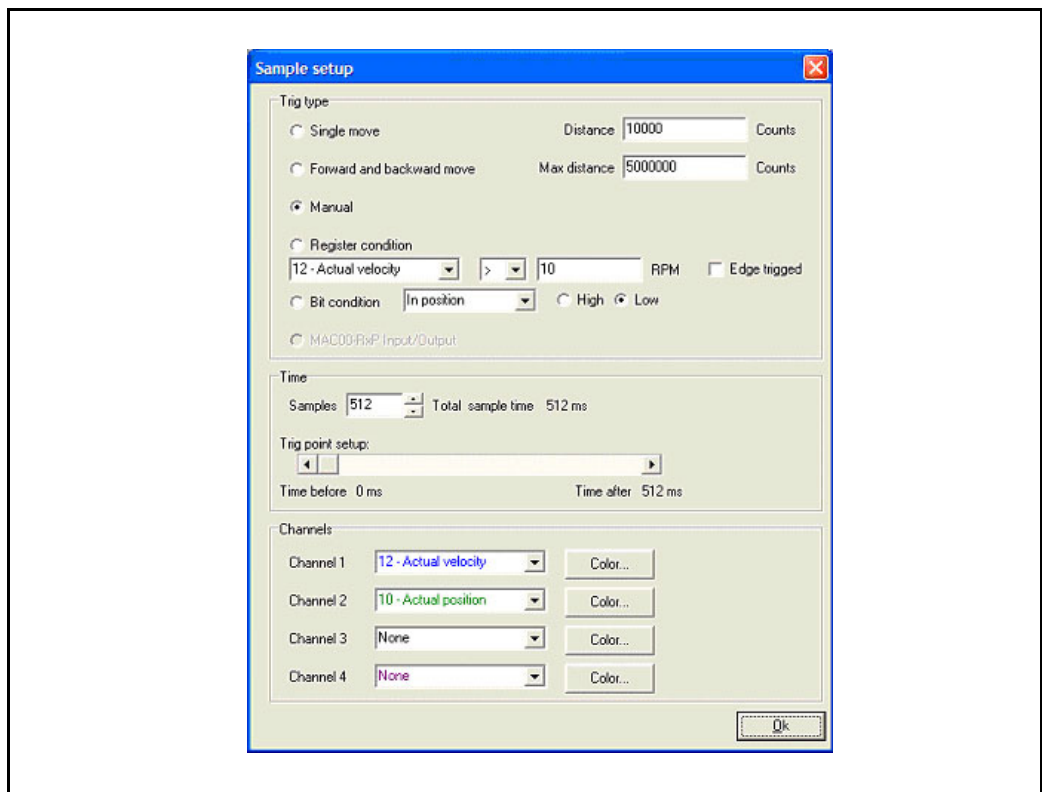
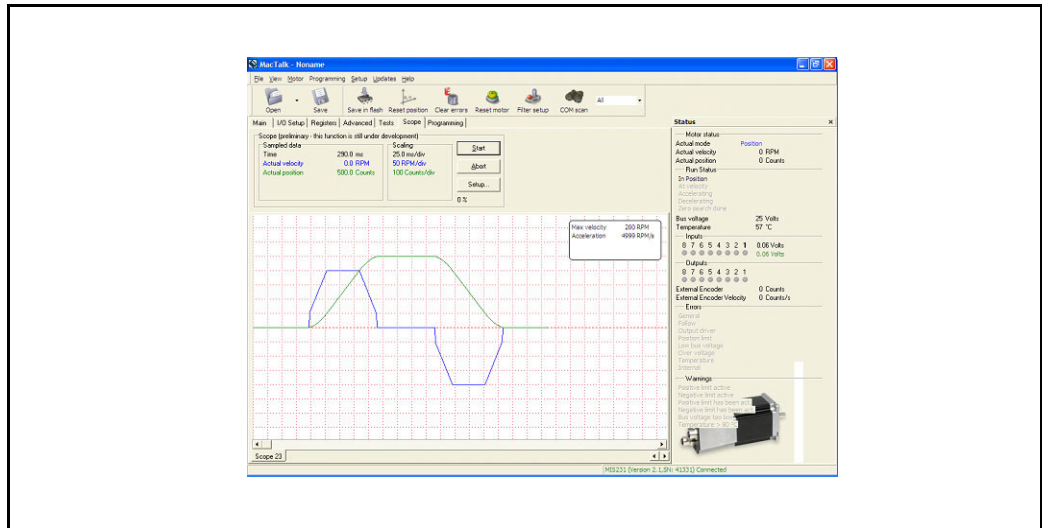
# 4.1 Using the MacTalk software

## 4.1.9 Scope Function

This function is not available yet!

The Scope function is an excellent and necessary function for testing a new application or finding errors in an existing system.

The Setup has to be selected to set up the Scope function correctly before use. Most registers in the MIS motors can be selected for viewing, different trigger functions can be selected, saving and loading scope pictures is possible, etc.

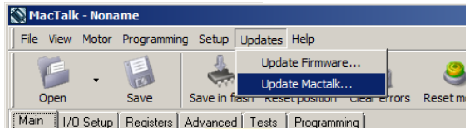


## 4.2 How to update MacTalk

MacTalk can be updated directly from the internet at any time. It is recommended always to use the latest version of MacTalk since it support the latest features and bugs may have been found and corrected. Below is shown how to make an update of MacTalk.

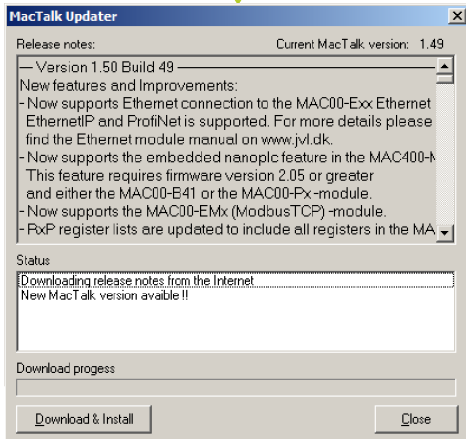
### Step 1

Choose the *Update MacTalk* in the *Updates* menu.



### Step 2

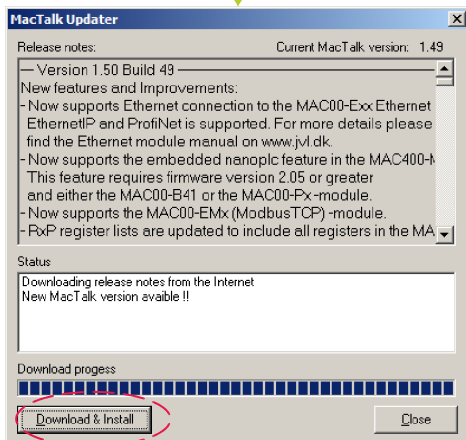
MacTalk will now check if newer version exist on the JVL server. If a newer version exist it will automatically be downloaded and the release notes can be seen in the window.



### Step 3

Press the *Download & Install* button to download the selected latest MacTalk.

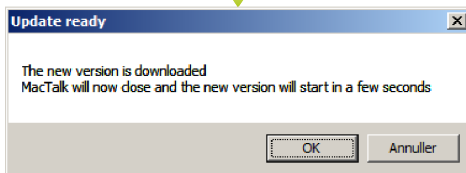
The progress counter will now rise from 0 to 100%.



### Step 4

When the download process is finished, the status shows "*Update ready*".

Press "*OK*" in order to start the new version of MacTalk.



### Step 5

After MacTalk have restarted the version number of the new MacTalk can be observed in the top of the screen.

The complete update is finished !.



TT2342-01GB

## 4.3 How to update the motor firmware

The firmware in the motor can be updated directly from the internet at any time by using MacTalk.

It is recommended always to use the latest version of the firmware available for the actual MIS motor used since it will contain the latest features and bugs may have been found and corrected. Below is shown how to make an update of the firmware. Notice that the screen dumps below is based on the update of a MIS34x but could be any other size of MIS motor.

### Step 1

The firmware update will erase the existing user setup of the motor. Use the Save button to save the existing setup before updating the motor. Then choose the *Update Firmware* in the *Updates* menu.

### Step 2

The first list shown is only the newest firmwares related to the actual motor connected.

It may also contains encoder and/or Ethernet firmware if these features are present.

To see all files also older versions enable the checkbox "Show all files".

Select the desired firmware, «SMCxxx firmware».

Press *Start* to download the selected firmware.

The progress counter will now rise from 0 to 100%.

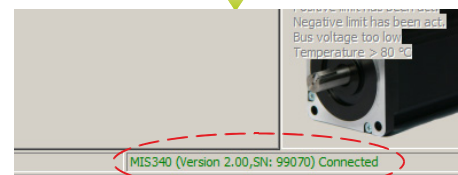
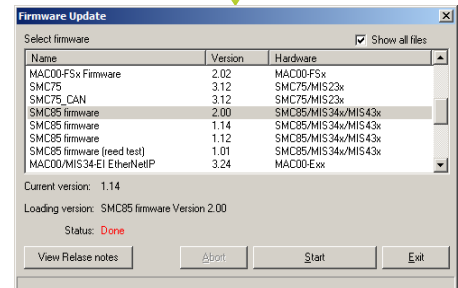
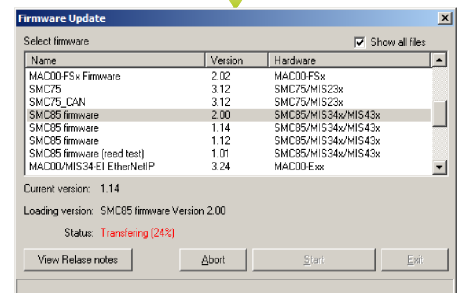
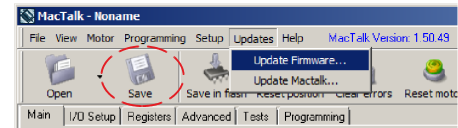
### Step 3

When the download process is finished, the status shows "Done".

### Step 4

The on-line information shown in the lower right corner of the MacTalk main window will now show the complete type of firmware and version.

The firmware update is now fully completed. Please remember that the settings of the motor is set back to default. But can be reinstalled by opening the user setup file made initially in this update sequence.



Hint!: Some older products may not start after pushing the "start" button showed above. If this is the case simply switch off power wait 5 seconds and reapply power. The update should now start.

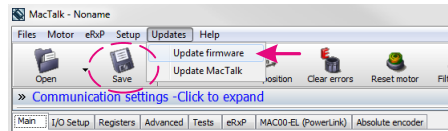
# 4.4 How to update the encoder FW

Only MISxxx---H3/H4

If the motor has the H3 or H4 (absolute multiturn encoder feature) then the firmware can be updated directly from the internet very easy at any time by using MacTalk. It is recommended always to use the latest version of the firmware available for the actual MIS motor used since it will contain the latest features and bugs may have been found and corrected. Below is shown how to make an update of the encoder firmware.

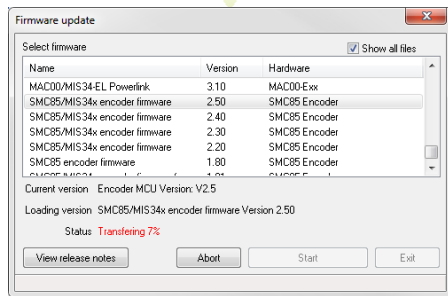
### Step 1

The encoder firmware update will erase the existing user setup of the motor. Use the *Save* button to save the existing setup before updating the motor. Then choose the *Update Firmware* in the *Updates* menu.



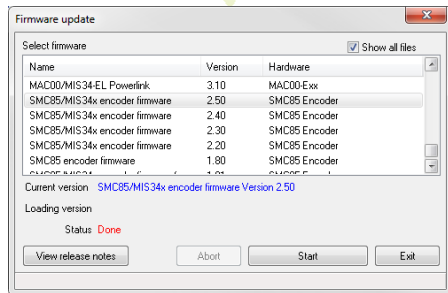
### Step 2

The first list shown is only the newest encoder firmwares related to the actual motor connected. It may also contains main and/or Ethernet firmware if these features are present. To see all files also older versions enable the checkbox "Show all files". Select the desired firmware, «SMC... encoder firmware». Press *Start* to download the selected firmware. The progress status counter will now rise from 0 to 100%.



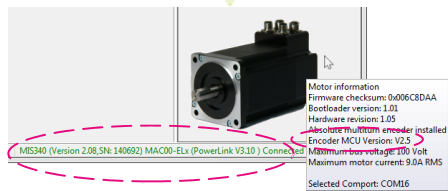
### Step 3

When the download process is finished, the status shows "Done". If the download process for some reason did NOT start/succed start from step 1 in this guide but switch off power until the «Start» botton have been activated and then switch on power.



### Step 4

The on-line information shown in the lower right corner of the MacTalk main window will now show the complete type of main firmware and optionally the ethernet firmware version (if ethernet is present) including version. The encoder firmware version is shown by placing the mouse cursor at top of the motor for a short while.



The firmware update is now fully completed. Please remember that the settings of the motor is set back to default. But can be reinstalled by opening the user setup file made initially in this update sequence.

TT2349-01GB

Hint!: Some older products may not start after pushing the "start" button showed above. If this is the case simply switch off power wait 5 seconds and re-apply power. The update should now start.

## 4.5 How to get SW/HW motor info

Place the mouse cursor in this field and the motor information box will show up

Motor information  
 Firmware checksum: 0x00DD31B7  
 FPGA Checksum: 0x0000E1E1  
 Bootloader version: 1.02  
 Hardware revision: 1.01  
 Encoder: Not installed  
 Maximum bus voltage: 100 Volt  
 Maximum motor current: 6.00A RMS  
 CanOpen: No support in hardware  
 Selected Comport: COM1

Info about which motor type and options that is connected with MacTalk - Also the serial number is monitored here

Info about SW and HW setups in the motor.

Description of contents in info box	Description												
<b>Firmware checksum</b>	Unique identifier for the installed firmware part 1. The firmware can be updated with MacTalk.												
<b>FPGA checksum</b>	Unique identifier for the installed firmware part 2. The firmware can be updated with MacTalk.												
<b>Bootloader version</b>	The program that handles firmware updates. The bootloader itself is installed during manufacturing.												
<b>Hardware version</b> Note: Internal controller SMC66 is used in MIS17x and MIS23x. Internal controller SMC85 is used in MIS34x and MIS43x.	The version of the controller/driver. All hardware versions supports the documented features in this manual except SMC85 (used in MIS34x and MIS43x) where that have been updated between version 1.05 and 1.07 (version 1.06 do not exist)												
	<b>SMC85:</b> <table border="1"> <thead> <tr> <th>1.05</th> <th>1.07</th> </tr> </thead> <tbody> <tr> <td>H2 encoder is 10 bit and does not support closed loop.</td> <td>H2 encoder is 10/12 bit and supports closed loop.</td> </tr> <tr> <td></td> <td>H4 encoder option available.</td> </tr> <tr> <td>Event log and actual position are saved in flash at shutdown. If CVI drops too rapid, the informations are not saved.</td> <td>Event log and actual position is saved every second and can be monitored in MacTalk during motor operation.</td> </tr> <tr> <td>Limited memory for future improvements.</td> <td>Increased memory for future improvements.</td> </tr> <tr> <td>Only external brake is available.</td> <td>Both external and integrated brake can be mounted.</td> </tr> </tbody> </table>	1.05	1.07	H2 encoder is 10 bit and does not support closed loop.	H2 encoder is 10/12 bit and supports closed loop.		H4 encoder option available.	Event log and actual position are saved in flash at shutdown. If CVI drops too rapid, the informations are not saved.	Event log and actual position is saved every second and can be monitored in MacTalk during motor operation.	Limited memory for future improvements.	Increased memory for future improvements.	Only external brake is available.	Both external and integrated brake can be mounted.
1.05	1.07												
H2 encoder is 10 bit and does not support closed loop.	H2 encoder is 10/12 bit and supports closed loop.												
	H4 encoder option available.												
Event log and actual position are saved in flash at shutdown. If CVI drops too rapid, the informations are not saved.	Event log and actual position is saved every second and can be monitored in MacTalk during motor operation.												
Limited memory for future improvements.	Increased memory for future improvements.												
Only external brake is available.	Both external and integrated brake can be mounted.												
	<b>SMC66</b> From version 1.01 has the same features as <b>SMC85</b> 1.07.												

## 4.5 How to get SW/HW motor info

---

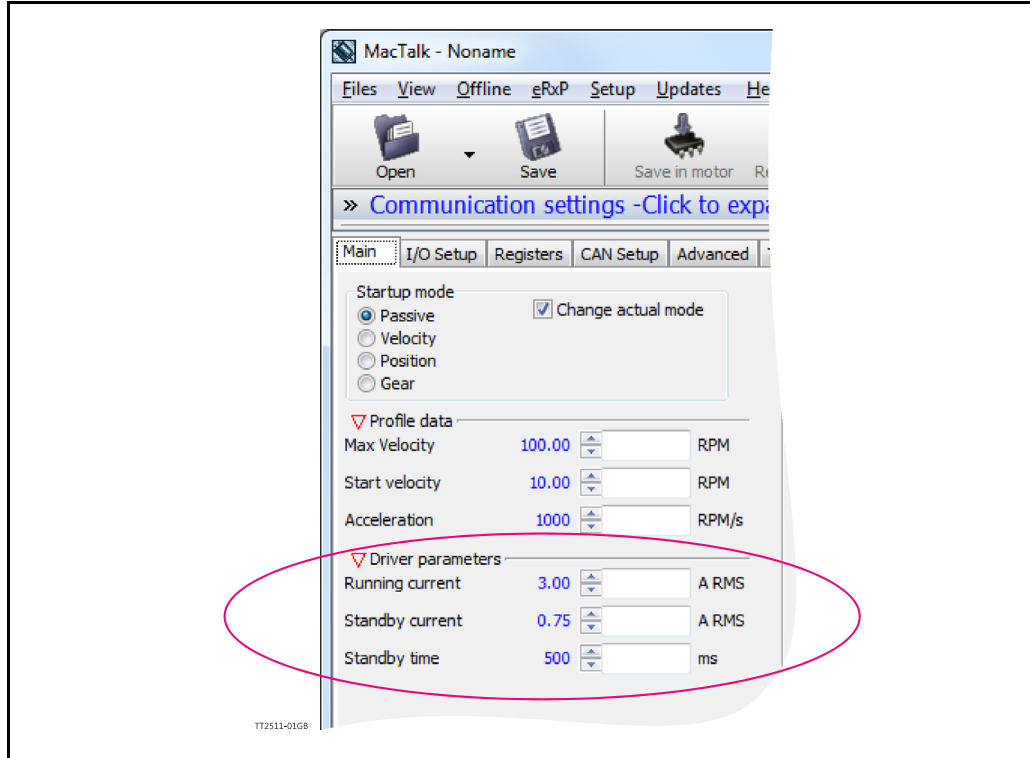
<b>Absolute multi turn encoder installed</b>	The absolute multiturn encoder option H3/H4 is installed.
<b>Encoder: Not installed</b>	An H2 encoder is not installed. Therefore the motor can only operate in open loop and will not be able to correct the position in case the motor is stalling.
<b>Encoder MCU Version</b>	The absolute multi turn encoder firmware version, can be updated with MacTalk.
<b>Maximum bus voltage</b>	The limit where the "Over voltage" error bit will be set.
<b>Maximum motor current</b>	The maximum motor Running and Standby current is defined by this value.
<b>CANopen</b>	An option which can be installed during the manufacturing.
<b>Selected Comport</b>	The actual serial com port the motor is connected to.





## 5.1 Setting up the motor current

The current supplied to each of the step motor's phases can be adjusted for standby and running currents by changing the values of standby and running currents under "Driver parameters" in the MacTalk program. The register is also accessible in general through the serial interface.



The electronics automatically switches between the two currents by detecting the presence of step-pulses. If a rising edge is detected at the step-clock, the "Move current" is selected. If no rising edge is detected during the period specified by "Standby time" at the step-clock input, the current is automatically switched back to "Standby current".

Values for the two currents are typically adjusted so that the Operating Current is significantly higher than the Standby Current, since the motor must be supplied with more power to drive its load during acceleration and constant operation than when it is stationary.

### 5.1.1 Standby current considerations

The main reason for having the Standby current setting is to optimise the heat produced by the motor.

Having a suitable standby current will make sure that the motor do no loose position but also make sure that the motor is not producing more heat than necessary.

A typical setting of the Standby Current typically is 30-40% of the Running current.

Normally the motor do not need to produce any significant torque during standby and therefore it makes sense to lower the standby current.

## 5.1 Setting up the motor current

### 5.1.2 MIS23x Current and torque relation

	Standard torque versions			High torque versions			Unit
	MIS231Q or MIS231S	MIS232Q or MIS232S	MIS234Q or MIS234S	MIS231T or MIS231R	MIS232T or MIS232R	MIS234T or MIS234R	
<b>Standby Current</b>	0-6000	0-6000	0-6000	0-6000	0-6000	0-6000	mA RMS
<b>Running Current</b>	0-6000	0-6000	0-6000	0-6000	0-6000	0-6000	mA RMS
<b>Holding Torque</b>	0-1.2[170]	0-1.9[269]	0-3.0[425]		0-2.5[xxx]		Nm [Oz-In]

### 5.1.3 MIS34x and MIS43x Current and torque relation

	MIS340	MIS341	MIS342				Unit
<b>Standby Current</b>	0-9000	0-9000	0-9000				mA RMS
<b>Running Current</b>	0-9000	0-9000	0-9000				mA RMS
<b>Holding Torque</b>	0-3.0[425]	0-6.1[863]	0-9.0[1274]				Nm [Oz-In]

If a MIS232 motor is used and the current is set to 6000 mA, the motor will be able to deliver a torque of 1.9 Nm at low speed. If the current is set to 3000 mA, the motor will be able to deliver 0.85Nm.

In other words the torque produced is proportional to the current setup.

See also [Run\\_Current](#), page 155 for information about Running Current and [Standby\\_Current](#), page 156 for information about Standby Current.

### 5.2.1 Position "Auto correction"

This feature is only active when the motor is in Position Mode. Also the function is only active if the motor is equipped with one of the encoder options -H2/H3 or H4.

The function can be called a semi-closed loop feature since it do only correct the motor position after a move have been done or tried done and not dynamically during a move like closed loop will do. See also [Introduction to closed loop operation.](#), page 76

The auto correction feature can be useful to assure that the motor reach its target position. The feature will take effect if for example the movement was physically blocked, the torque of the motor was insufficient, or a bad value for start velocity or acceleration were used. It might also be used to handle occasional mechanical oscillations.

### 5.2.2 "Auto correction" - basic function

The auto-correction feature is only used when the motor has stalled and not reached its final target position within the given position window.

Each time the motor has done a movement the "Actual position" counter and the "Encoder position" counter is compared.

If the difference without sign is within the value specified in the "In position window" as shown below no further action is taken.

If the difference is larger than the value in the "In position window" the motor will try to correct the position by doing a new motor movement. The "Max number of retries" is the number of times the motor will try to correct the position, and the "Settling time between retries" is the time the motor will wait between each retry.

The AutoCorrection system will first wait (unconditionally) for a certain time (settling time) to allow the initial movement to settle mechanically before testing for the target position. It will then attempt a normal movement, using the same values for velocities and acceleration as the movement that failed. It will continue until the encoder position is within the target window, or the selected number of retries has expired.

Note that AutoCorrection will only start after the value of the Position (P\_SOLL) register is changed. In other words, changing P\_SOLL (not just writing the same value again) will reload the maximum number of retries and set the Auto Correction Active status bit. The Auto Correction Active status bit will remain set until either the position is within the target window or the max number of retries has been exhausted.

Also note that if the motor is used to control other motors by sending out the pulse and direction signals on digital outputs, any extra movements caused by AutoCorrection will send out additional steps to the other motors.

## 5.2.3 Setup with MacTalk

Following parameters are available in MacTalk.

When selected the in position flag will realtime indicate if the motor is within the position window compared to a perfect move.

TT2338-01GB

**In position window**

Defines the window wherein the motor must be before the In Physical Position flag is set.

**Autocorrection velocity**

Defines the velocity used if a correction is done. Can be useful when Ethernet or other protocol is used since the main velocity register can be overwritten with a velocity value which is not optimal for auto correction.

**Max number of retries**

Defines how many auto correction retries that are allowed to be done

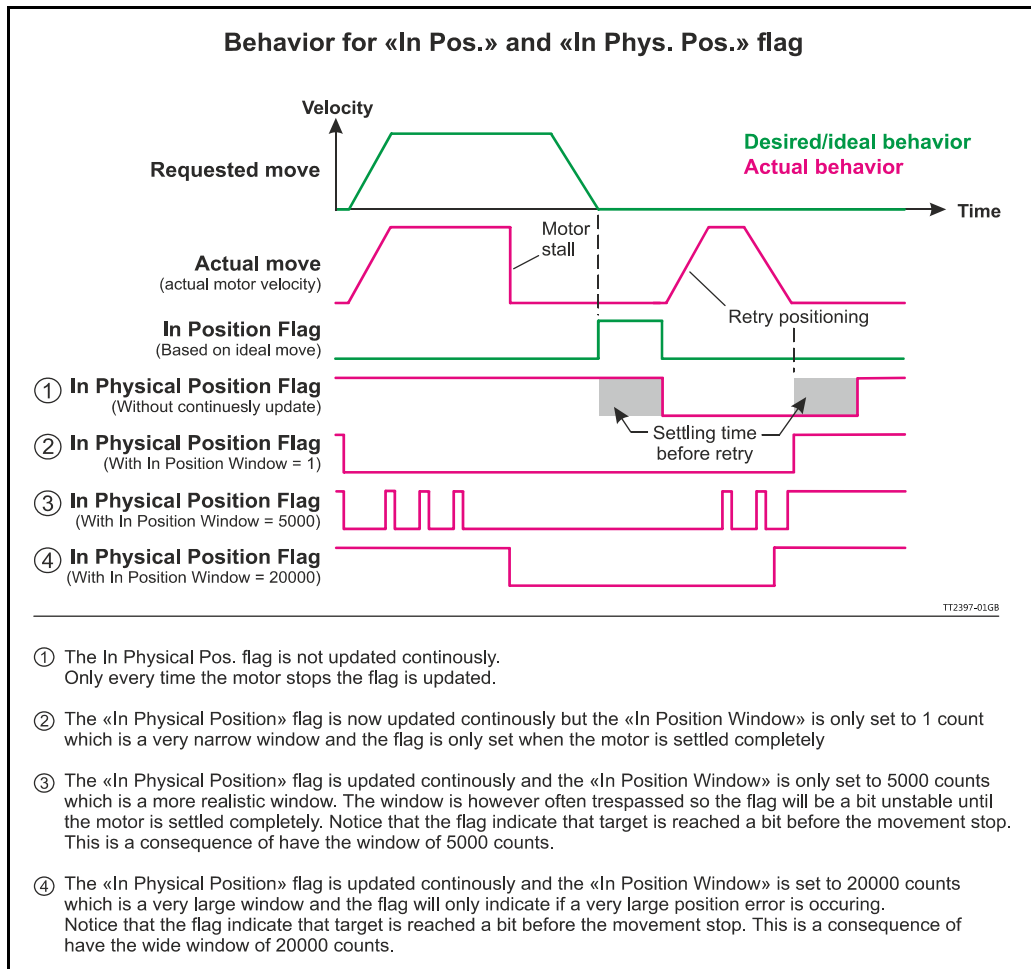
**Settling time between retries**

Defines defines the time between the auto correction retries

**Update the In Physical Position bit continuously**

Defines if the In Physical Position bit is updated continuously or only after the motor has stopped (default)

The relation between settings and behaviour of the In Physical Position flag can be seen below.



#### 5.2.4 Setup without MacTalk

If MacTalk is not used for setting up parameters/registers related to the auto correction feature it must be done as follows.

The motor contains a number of registers which can be accessed from various protocols depending at which options the motor has.

Protocols available are for example Ethernet (EthernetIP, Profinet etc.) and CANopen, Modbus or the MacTalk protocol.

Each field in MacTalk described earlier in this chapter is accessing a register in the motor. The registers that are relevant for auto correction operation are:

##### R33 - IN\_POSITION\_WINDOW

##### MacTalk name: “In Position Window”.

Selects how close the internal encoder position must be to the target Position (P\_SOLL) to set the InPhysical-Position status bit and prevent further AutoCorrection.

See also: [IN\\_POSITION\\_WINDOW](#), page 162

**R236 - V\_SOLL\_AUTO****MacTalk name: “Auto correction velocity”.**

The auto correction is done per default with the velocity specified in the general velocity register. If an alternative velocity is intended the V\_SOLL\_AUTO register can be used. If V\_SOLL\_AUTO != 0 it will be used instead of the general velocity.

See also: [V\\_SOLL\\_AUTO](#), page 192

**R34 - IN\_POSITION\_COUNT****MacTalk name: “Max. number of retries”.**

Specifies the maximum number of auto correction retries before no further attempts are done. A value of 0 (zero) effectively disables AutoCorrection.

See also: [IN\\_POSITION\\_COUNT](#), page 162

**R110 - SETTLING\_TIME****MacTalk name: “Settling time between retries”.**

When the internal encoder option is installed and register 34, InPositionCount, is none-zero so AutoCorrection is enabled, the value in this register defines how many milliseconds to wait after each movement attempt before testing whether the encoder position is within the target window as defined in register 33. This waiting time is often necessary to allow mechanical oscillations to die out.

See also: [Settling Time](#), page 171

**R124 - SETUP\_BITS****MacTalk name: “Update the In Physical Position bit continuously”.**

Bit no. 6 defines if the In Physical Position bit should be updated continuously or not.

Default: Bit 6 = 0 = only update after motor stops.

See also: [Setup\\_Bits](#), page 174

**R25 - STATUSBITS****MacTalk name: (Run status area)**

This register contains 2 bits that are relevant for the auto correction feature.

**Bit 1: AutoCorrection Active**

If set an auto correction cycle is in progress because target position was not met.

**Bit 2: In Physical Position**

If set the motor position is physically within the In\_Physical\_Position\_Window

See also: [Status bits](#), page 160

### 5.3.1 Introduction to closed loop operation.

When running a stepper motor under normal operating conditions the load torque will cause a small displacement of the rotor from its nominal position, but normally such displacements do not lead to a loss of synchronization.

However, synchronization will be lost if the load exceeds the available motor torque. This can cause the motor to stall with a position loss which must be corrected afterwards – for example with the *Auto correction* function (see also [Auto Correction](#), page 72) or by monitoring the position externally with an encoder.

The MISxxH2xx (only available from serial numbers 173000) and MISxxH4xx have the ability to run in **closed loop** and therefore always tracks the rotor displacement in real time. The control algorithm aligns the commutation angle and motor current when needed. This avoids that the motor is stalling and runs the motor at a lower current when possible with the advantage that the overall system efficiency is much better.

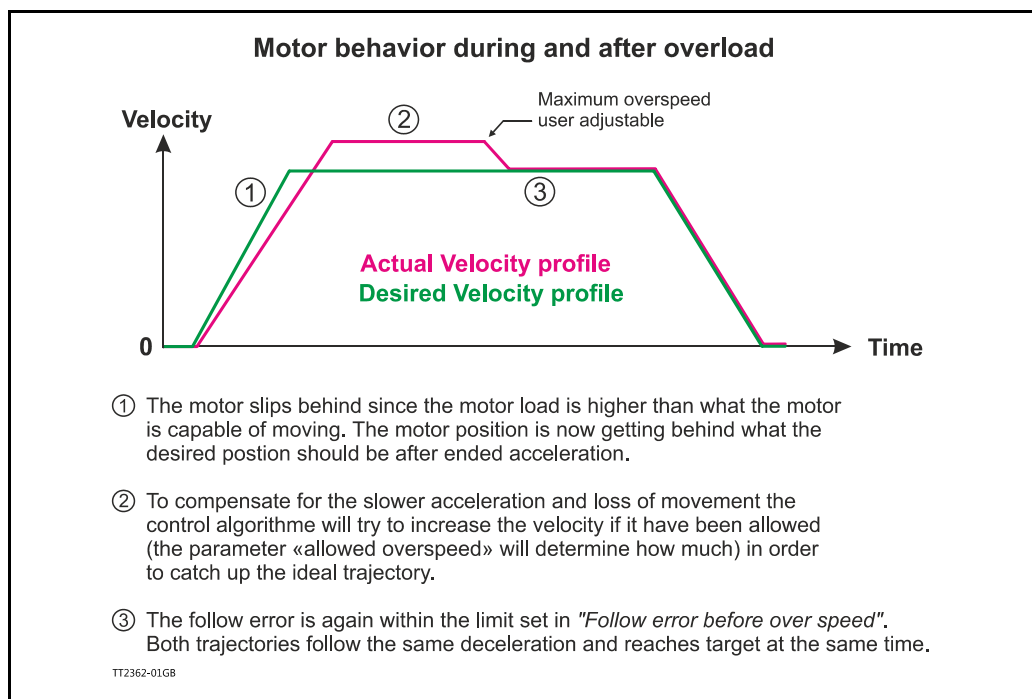
#### The closed loop offers 2 ways of operation

1. Closed loop with constant current. The current is maximum all the time regardless which load is applied to the motor.
2. Closed loop with dynamic current control. The current is adjusted real time to match the actual load. The advantage is that the motor runs more efficient (less heat) and the audible noise when running is much less.

### 5.3.2 Examples of motor behaviour in closed loop

#### Example 1:

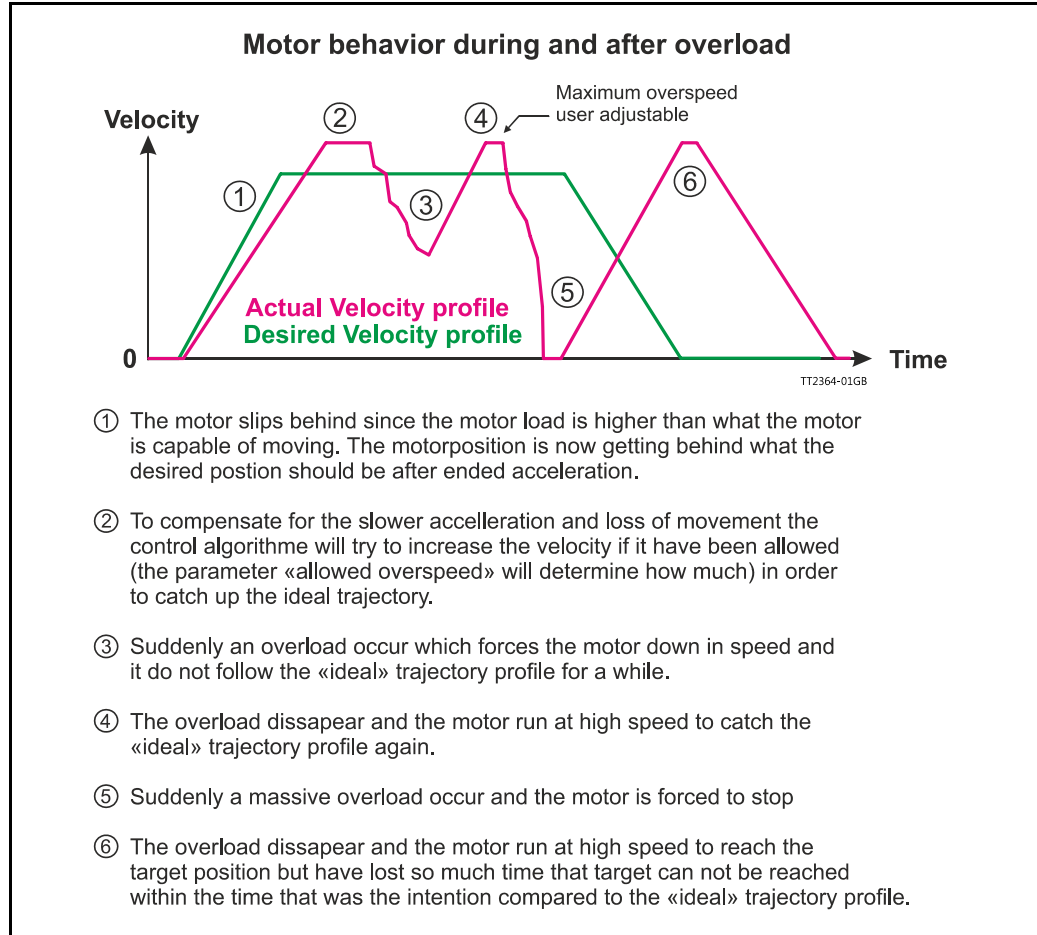
A too high acceleration has been set. The motor cannot accelerate the load fast enough and therefore a follow error will be incremented until the motor is able to maintain the right speed.





**Example 2:**

The motor is not able to follow the ideal acceleration and therefore increases the velocity to catch up the follow error. During the movement overloads also occur forcing the motor to go down in speed because lack of torque to overcome the load.

**5.3.3 Current control (optional)**

In a classic stepper motor system (not closed loop) the motor current is typically set to maximum to make sure that the motor does not stall and lose track of its position. The disadvantage is that the motor becomes hot and energy is wasted.

When running in closed loop, loss of position (motor stall) is not a concern and therefore the control algorithm's current control will (if enabled) adjust the current to a level where the motor is able to follow the requested velocity and not lose the position.

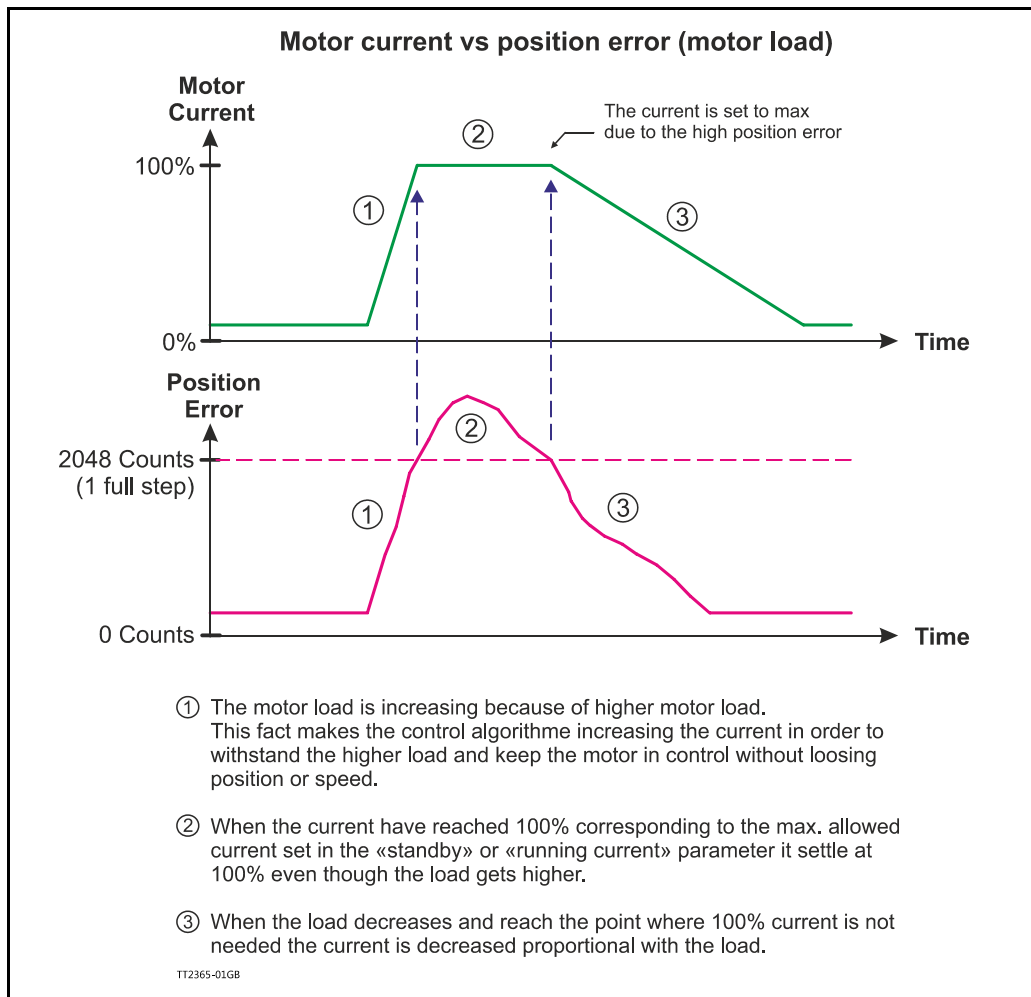
The actual running current ("Actual torque" in MacTalk) is a percentage of the user-defined "Running current". The motor can never run with a higher RMS current than the one specified in the "Running current" register, and the algorithm decreases the actual running current according to the follow error.

**Example:**

The example shows a situation where the motor is running at a steady velocity and the follow error is stable. Suddenly the motor is temporarily overloaded, motor current is increased. When the current has reached 100% it settles since the electronics can not produce more than 100% and also have to respect the user current setting. The follow error still increases because the load on the shaft exceeds the available torque.

In the last part of the cycle the load is lowered and when the position error becomes below 2048 counts (1 full step) the current is also proportionally decreased by the control algorithm and it settles at the same level as before the increased load/position error.

As illustrated on the figure, the slope of the current increase and decrease are asymmetrical. This is to stabilize the current control.



## 5.3

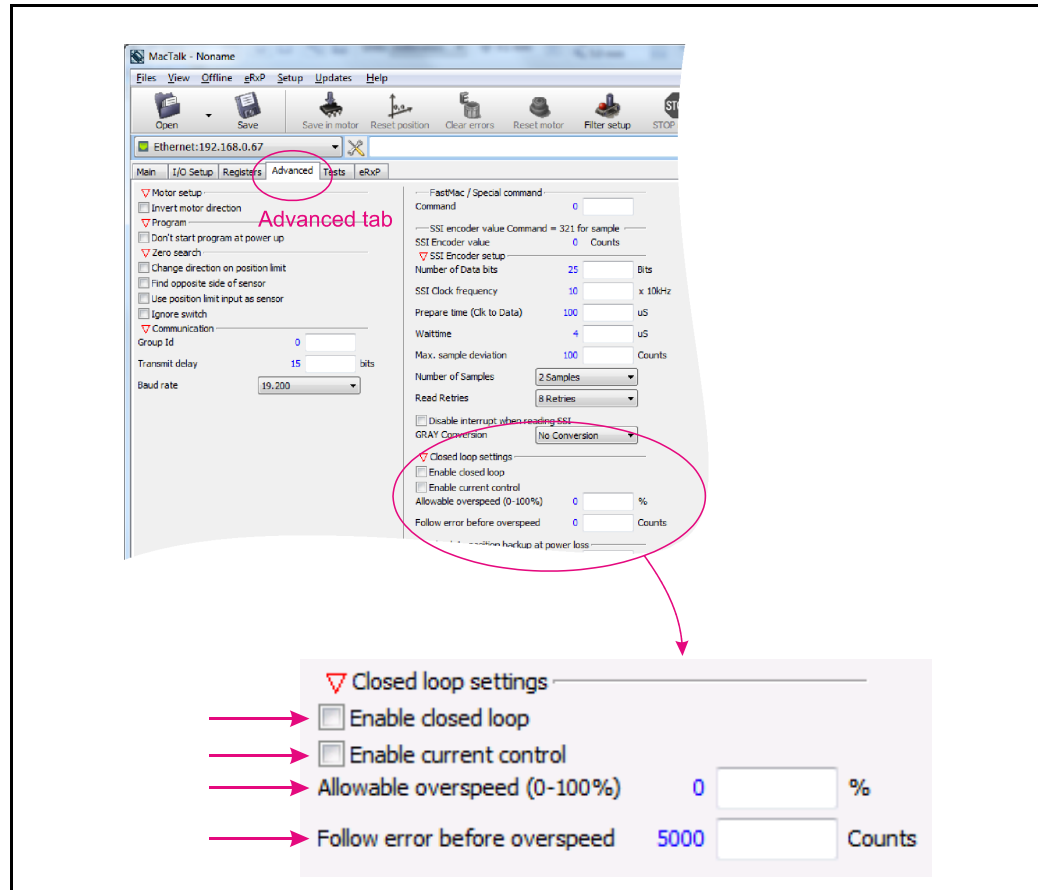
# Closed loop operation

Only MISxxx---H2/H4---

### 5.3.4 Setup with MacTalk

From firmware version 4.00 the closed loop control is available, if either the H2 or H4 encoder option is installed. Only motors with serial number 173000 or higher support the closed loop feature.

The closed loop function is by **default disabled**, also the current control is by default disabled.



Parameter explanations:

#### Enable closed loop

Check this field to activate the overall closed loop feature. Can be activated/deactivated on the fly and have immediate effect. Default: inactive.

#### Enable current control

Check this field to add current control. The current control will increase/decrease the current proportional to what is needed for driving the motor load. Can be activated/deactivated on the fly and have immediate effect. Default: inactive.

#### Allowable over speed (0-100%) and Follow error before overspeed

Allows the motor to run with a higher velocity if the encoder position deviate from the theoretical position by more than "Follow error before overspeed" counts.

The default for "Allowable overspeed..." is 0% which means that the maximum speed will never exceed the "Max velocity" setting.

The default for "Follow error before overspeed" is 5000 counts.

## 5.3

# Closed loop operation

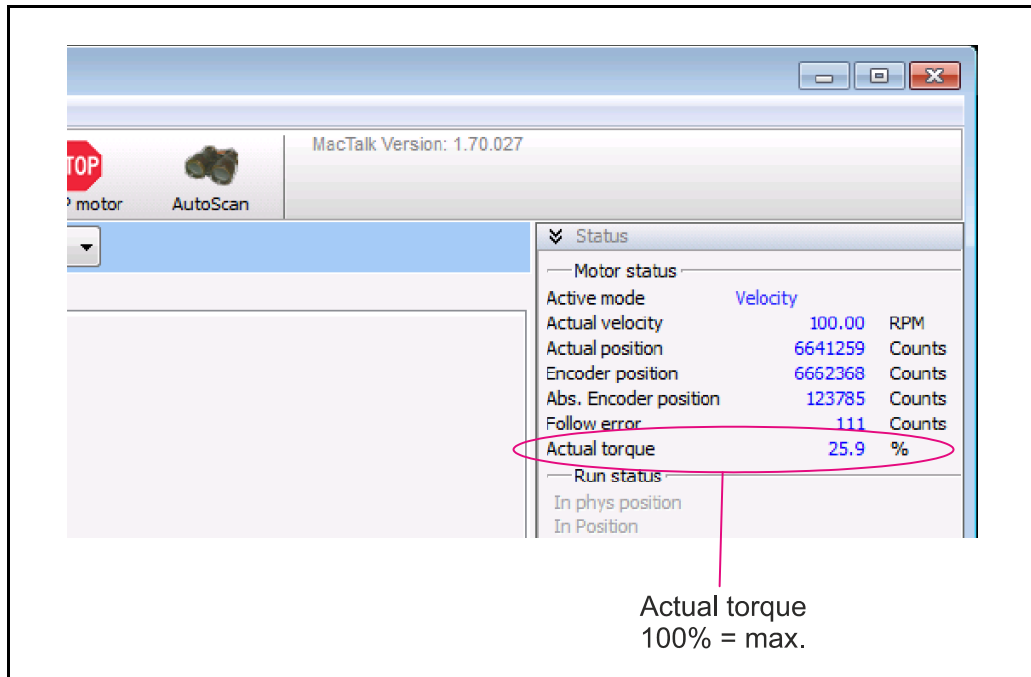
Only MISxxx---H2/H4---

### 5.3.5 Monitoring the actual motor torque

When using the closed loop feature and having the current control activated at the same time - the actual torque can be monitored.

The actual torque read-out is based on the actual motor current as a percentage of the "Running current" setting.

In MacTalk it is labelled "Actual torque". Its monitored among the other motor status values in the right side of the main screen.



The "Actual torque" is defined from how many percentage of the "Running current" that is used.

*Continued next page...*

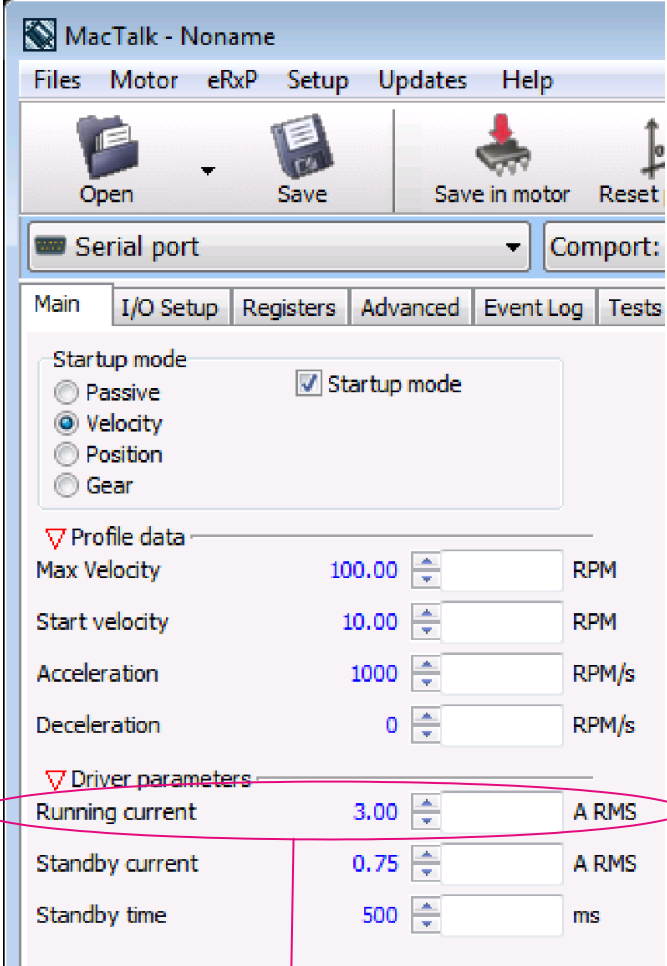
## 5.3

# Closed loop operation

Only MISxxx---H2/H4---

The allowed maximum current that can be used when the motor is running is setup in the “Running current” register which is found at the main tab in the left side. Notice that when using the closed loop current control feature only the “Running current setting is used”.

The “Standby current” and “Standby time” setting is **not** used at any time.



The screenshot shows the MacTalk software interface with the following settings:

Parameter	Value	Unit
Startup mode	Velocity	
Max Velocity	100.00	RPM
Start velocity	10.00	RPM
Acceleration	1000	RPM/s
Deceleration	0	RPM/s
Running current	3.00	A RMS
Standby current	0.75	A RMS
Standby time	500	ms

Running current can be set here  
Standby current and Standby time is not used  
when current control and closed loop is enabled.

**5.3.6 Setup without MacTalk**

If MacTalk is not used for setting up parameters/registers related to the closed loop feature it must be done as follows.

The motor contains a number of registers which can be accessed from various protocols depending at which options the motor has.

Protocols available are for example Ethernet (EthernetIP, Profibus etc.) and CAN-open, Modbus or the MacTalk protocol.

Each field in MacTalk described earlier in this chapter is accessing a register in the motor.

The registers that are relevant for closed loop operation are:

<b>R25</b>	STATUS_BITS	Overall status related to the closed loop feature. - Bit 15: Closed loop lead/lag - Bit 16: Closed loop activated - Bit 17: Internal encoder calibrated - Bit 20: Internal encoder ok See also: <a href="#">Status bits</a> , page 160
<b>R35</b>	ERR_BITS	Error status related to the closed loop feature. Following bits are available in the setup bit register. Bit 12: Closed loop error. Bit 14: Abs. single turn encoder (H2/H4) error. See also: <a href="#">Err_Bits</a> , page 163
<b>R124</b>	SETUP_BITS	This register is used to activate or deactivate the closed loop and current control feature. Following bits are available in the setup bit register. - Bit 24: Enable closed loop - Bit 25: Enable closed loop current control See also: <a href="#">Setup_Bits</a> , page 174
<b>R217</b>	ACTUAL_TORQUE	The actual motor current in closed loop with active current control. 2047 = 100% current. See also: <a href="#">ACTUAL_TORQUE</a> , page 190

**Other registers that may be relevant for the closed loop operation:**

<b>R5</b>	V_SOLL	The maximum (nominal) velocity allowed. See also: <a href="#">V_SOLL</a> , page 155
<b>R6</b>	A_SOLL	The acceleration/deceleration ramp to use. See also: <a href="#">A_SOLL</a> , page 155
<b>R7</b>	RUN_CURRENT	The maximum motor current is setup in this register. See also: <a href="#">Run_Current</a> , page 155

## 5.3.7 Special settings



**Please note:** The following parameters are optional and it is not recommended to modify them since they are optimized from factory.

The following 2 registers are relevant for the closed loop operation

**Motor commutation and encoder setup:**

1. KPHASE - register that offsets the magnetic field as function of the velocity  
See also: [Internal\\_Encoder\\_Setup](#), page 186
2. Internal\_Encoder\_Setup - register that define the encoder resolution and other encoder related parameters. See also: [Internal\\_Encoder\\_Setup](#), page 186

**Current control advanced registers:**

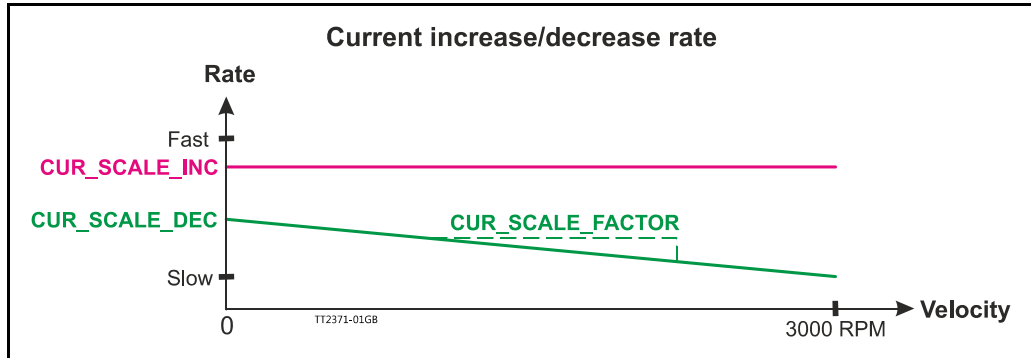
None of the below shown registers are accessible from MacTalk but only from other protocols such as Ethernet, Modbus, CANopen etc.

The registers shown below are all related to optimizing the current control performance.

<b>R212</b>	CUR_SCALE_MAX	The maximum allowable current. 2047 = 100% (of Running current). See also: <a href="#">CUR_SCALE_MAX</a> , page 188
<b>R213</b>	CUR_SCALE_MIN	The minimum allowable current. 1 = 0,05% (of Running current). See also: <a href="#">CUR_SCALE_MIN</a> , page 188
<b>R215</b>	CUR_SCALE_FACTOR	The slope of the velocity dependent current decrement rate. See also: <a href="#">CUR_SCALE_FACTOR</a> , page 189
<b>R218</b>	CUR_SCALE_INC	Current increment rate (independent of velocity). See also: <a href="#">CUR_SCALE_INC</a> , page 190
<b>R219</b>	CUR_SCALE_DEC	Current decrement rate (at 0 RPM) See also: <a href="#">CUR_SCALE_DEC</a> , page 190

*Continued next page.*

The relationship between the current control rate and velocity is illustrated on the figure. The decrement rate is inverse proportional to the velocity because the motor is much more sensitive to variations in the running current at high speeds. It gives a more stable motor behaviour if the current is decreased slower at high speeds. The increase rate must always be fast enough to detect the worst case where the motor is blocked at 3000 RPM.



The start value and the slope can be modified with the registers `CUR_SCALE_INC`, `CUR_SCALE_DEC` and `CUR_SCALE_FACTOR`. The default values are already optimised to give the best overall performance on all velocities, some other values could be better for at specific velocity.

How to calculate how the registers affect the timing at a specific velocity (RPM):

$$DEC\_CNT(RPM) = \left( \frac{RPM \times 100}{4096} \times CUR\_SCALE\_FACTOR \right) + CUR\_SCALE\_DEC$$

$$T_{DEC} = \frac{1}{36MHz} \times DEC\_CNT$$

The increment timing is independent of velocity:

$$T_{INC} = \frac{1}{36MHz} \times CUR\_SCALE\_INC$$

Timing for the default factory settings:

$$DEC\_CNT(0) = \left( \frac{0 \text{ RPM} \times 100}{4096} \times 500 \right) + 4000 = 4000$$

$$T_{DEC\_0} = \frac{1}{36MHz} \times 4000 = 111\mu s$$

$$DEC\_CNT(3000) = \left( \frac{3000 \text{ RPM} \times 100}{4096} \times 500 \right) + 4000 = 40621$$

$$T_{DEC\_3000} = \frac{1}{36MHz} \times 40621 = 1,13ms$$

$$T_{INC} = \frac{1}{36MHz} \times 2000 = 55,56\mu s$$

The current is regulated in 2048 steps from 0-100 %.

This means that it will take  $T_{INC} * 2048 = 113 \text{ ms}$  for the current to increase from 0 to 100 % - at all velocities.

It will take  $T_{DEC\_0} * 2048 = 227 \text{ ms}$  for the current to decrease from 100 % to 0 % at 0 RPM and  $T_{DEC\_3000} * 2048 = 2,3 \text{ s}$  at 3000 RPM.



## 5.4

# Absolute position back-up

Only MISxxx---H2---

### 5.4.1 Position back-up

This feature is specifically ment to be used when the motor is equipped with a H2 encoder and is ment to be a “low cost” solution compared to having the H3 or H4 encoder which is a full featured absolute multiturn encoder that takes care of keeping the motor position intact regardless if motor is powered or not.

### 5.4.2 Functional description

The position backup feature makes it possible to save the last position before power was removed. If the motor has not been moved more that half a revolution in either direction during power down.

An hardware improvement have been made on newer motors having a serial number (label at motor will tell) >SNI 73000. Before and after serial number I 73000 the MIS motor therefor have a different functionality.

### 5.4.3 Setup position backup using MacTalk

— Absolute position backup at power loss —

Acceptance Count 100 Times — Acceptance count Register 140

Acceptance voltage 18.0 Volt — Acceptance voltage Register 139

Save voltage 15.0 Volt — Save threshold voltage reg. 141

Value of P\_IST after powerup

P\_IST = 0

Absolute Singleturn Encoder — SetupRegister Register 124

Absolute Multiturn Encoder — Only valid if H2 encoder is present

Valid for H3 or H4 encoder option  
If no encoder option is present the position data stored by the backup system will be used.

TT2265GB MIS340 (Version V4.00.00070, SN: 1) Connected

The available options when using the absolute position backup feature is available in MacTalk as shown above.

The parameters available and their function are as follows:

Acceptance Count  
Acceptance Voltage  
Save Voltage

Value of P\_IST after power up

P\_IST=0  
Absolute Single turn Encoder  
Absolute Multiturn Encoder

#### 5.4.4 Setup position backup NOT using MacTalk

An hardware improvement have been made on newer motors having a serial number (label at motor will tell) >SN I73000. Before and after serial number I73000 the MIS motor therefor have a different functionality.

##### **Only MIS34x and MIS43x: Serial numbers < I73000 or HW<VI.6:**

See also [How to get SW/HW motor info](#), page 67 to obtain this information.

The absolute position backup system is activated when a voltage goes under a level, defined by SAVE\_VOLTAGE (register I41).

Then all absolute multiturn information is saved to the flash memory at once.

When power is applied again all data are recalled from the flash memory and the motor can start operating based on the actual position that it had before power down.

It is required that the supply control voltage drops relatively slowly to allow time to save the values to flash memory. This can be secured by adding, if necessary, a large capacitor on the CVI supply voltage and powering on/off the external power supply on the AC side. Beside the position information also the Event log information is saved. This is very helpful for later troubleshooting.

##### **Only MIS34x and MIS43x: Serial numbers < I73000 or HW>VI.5:**

See also [How to get SW/HW motor info](#), page 67 to obtain this information.

The absolute position backup system is active all the time and is saving the absolute multi turn position data once every second. This feature is based on a relatively new memory technology called FRAM (Ferroelectric RAM) and no battery backup is involved.

If suddenly the supply power is removed and the motor stay within half a revolution during power down the position is valid. If the motor was running during power down the position data is probably invalid.

When power is applied again all data are recalled from the memory and the motor can start operating based on the actual position that it had before power down.

Beside the position information also Event log information is saved. This is very helpful for later troubleshooting.

#### 5.4.5 Registers involved

##### **Register I41,**

Save Threshold Voltage, selects the voltage threshold, that will trigger the flash backup save operation (and stop all other motor operation).

When register I42 has the value I2, the scaling/unit of register I41 is the same as register 97, Bus Voltage (4095 = I I I.4V).

The register I42 has the values of I-8 or 8I-88, the scaling/unit of register I41 is the same as registers 8I-96 (4095 = 5.0V)

##### **Register I39,**

Acceptance Voltage, selects the voltage threshold that defines when the power supply is ready to use for erasing flash memory after power up. The scaling/unit is the same as register I41.

## 5.4 Absolute position back-up

Only MISxxx---H2---

### Register 140,

Acceptance Count, selects the number of times the Acceptance Voltage must be measured after power up before the flash erase operation is started.  
The count is in units of 1 ms.

### Register 124,

SetupBits, selects to use Flash-based Absolute Multi turn Encoder functionality when bit 11 is set.

All data storage done by the absolute position backup function can also be monitored in the event log - see also : [Reading the Event log](#), page 134

## 5.5 Multifunction I/O setup

The MIS motor contain a dual RS485 port also called the Multifunction port, since it can be setup for many purposes such as data I/O, pulses out from the internal encoder, and pulses in from an external encoder.

The setup is not yet fully integrated in MacTalk but the following guide shows how to setup the most common combinations.

**Following setups are described in this chapter:**

- 1 Internal encoder (H2 or H4 option only) is send out at multifunction channel A and B. Link to description: [Internal encoder \(H2/H4\) signals](#) , page 89
- 2 Quadrature signals from the internal pulse generator is sent to the multifunction channels. Link to description: [Quadrature signals from the internal pulse generator](#), page 89
- 3 Same as #2 but in pulse/direction format instead of quadrature. Link to description: [Pulse/direction signals from the internal pulse generator](#), page 90

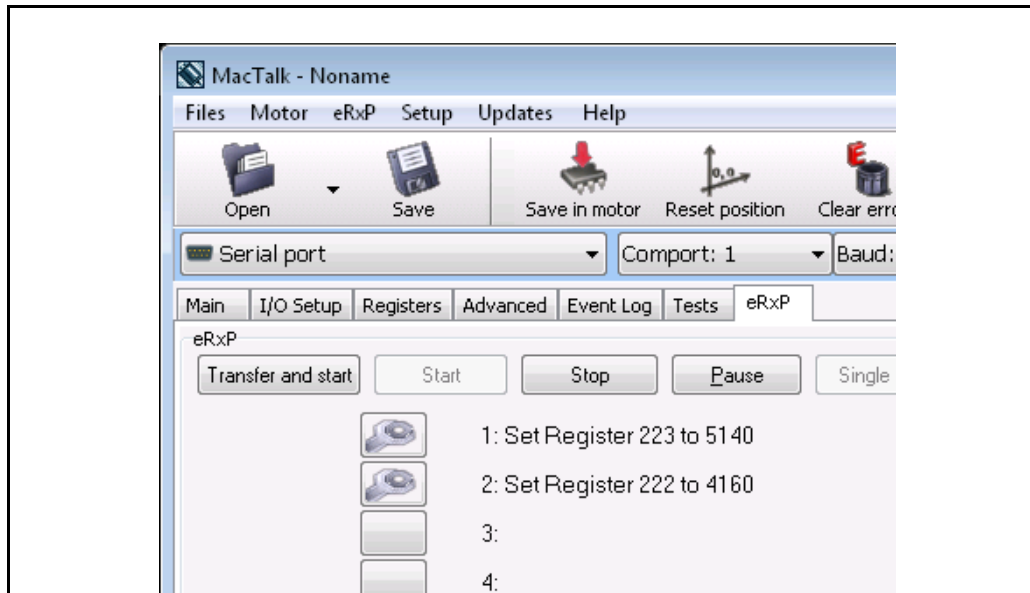
### 5.5.1 Configuration of the Multifunction port in general

The RS485 interface (dual) can be setup to output internal encoder signals, pulse/direction or quadrature signals from the internal pulse generator etc.

The registers mentioned below can be written by any supported protocol/interface, but are not visible as fields in MacTalk. By using the RxP program, the 2 registers can be written with 2 lines in the beginning of the program:

**R223 XFIELD\_DATA**  
**R222 XFIELD\_ADDR**

Data for the internal switch board/cross field setup.  
Address for the internal switch board/cross field setup.



Following text describe specifically which values that must be written into register 222 and 223 to obtain various outputs from the motor.

## 5.5 Multifunction I/O setup

### 5.5.2 Internal encoder (H2/H4) signals

The internal encoder quadrature signal generates 1024 pulses (4096 counts) per revolution. To output the signals A and B on the RS485 A and B port, write to the following registers via a little program in the RxP program:

Set register 223 to 5140

Set register 222 to 4160

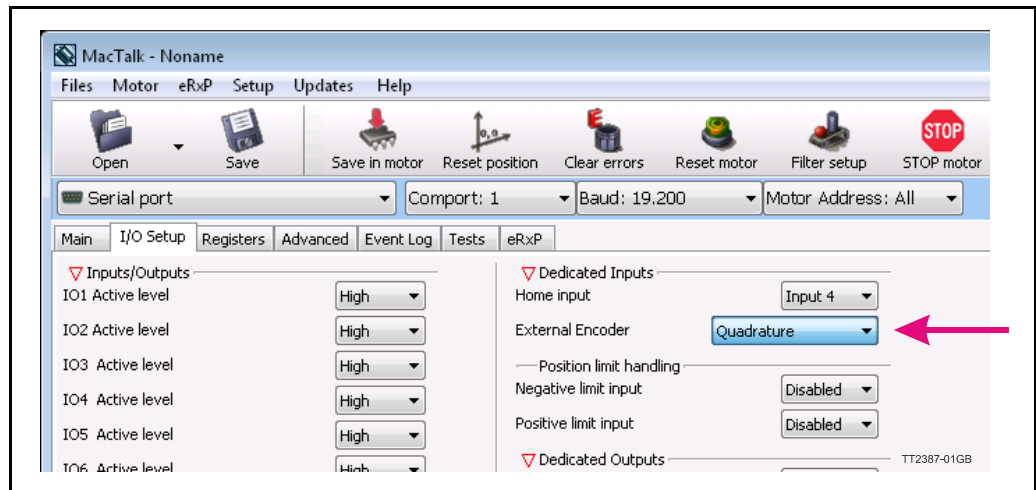
### 5.5.3 Quadrature signals from the internal pulse generator

The internal pulse generator has a resolution of 409600 cpr (20 MHz at 3000 RPM), but for the output channel it is scaled by 16 in order to limit the max output frequency to 1.28 MHz at 3000 RPM.

Set register 223 to 4626

Set register 222 to 4160

Then set the internal pulse generator to quadrature format:



Which can also be done without MacTalk:

#### RI24 SETUP\_BITS

A general setup register for many settings in the motor. Only bit 2 and 3 are used for this setting, all other bits must not be changed. This table shows how the combo box in MacTalk is setting the bits:

Bit 2	Bit 3	Setting
0	0	None
0	1	Quadrature
1	0	Pulse/direction

## 5.5 Multifunction I/O setup

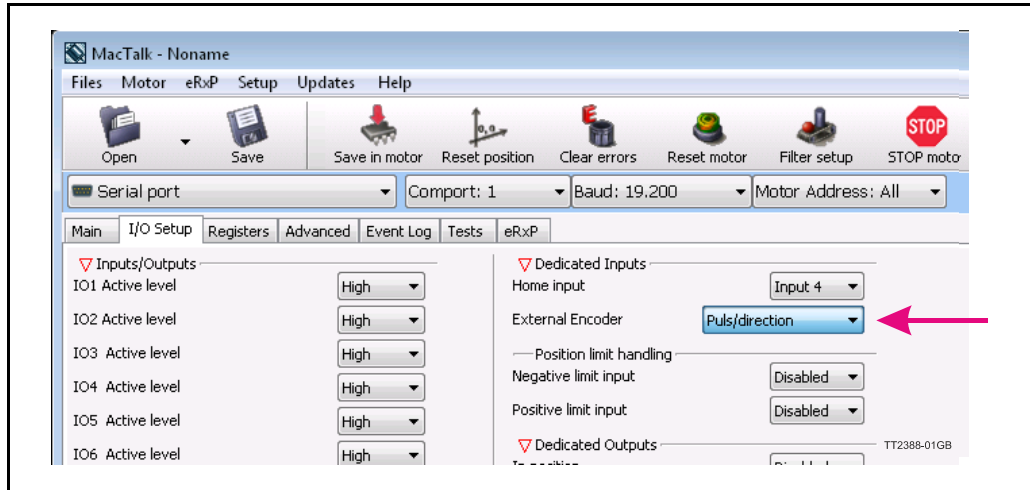
### 5.5.4 Pulse/direction signals from the internal pulse generator

The internal pulse generator has a resolution of 409600 cpr (20 MHz at 3000 RPM), but for the output channel it is scaled by 16 in order to limit the max output frequency to 1.28 MHz at 3000 RPM.

Set register 223 to 4626

Set register 222 to 4160

Then set the internal pulse generator to pulse/direction:



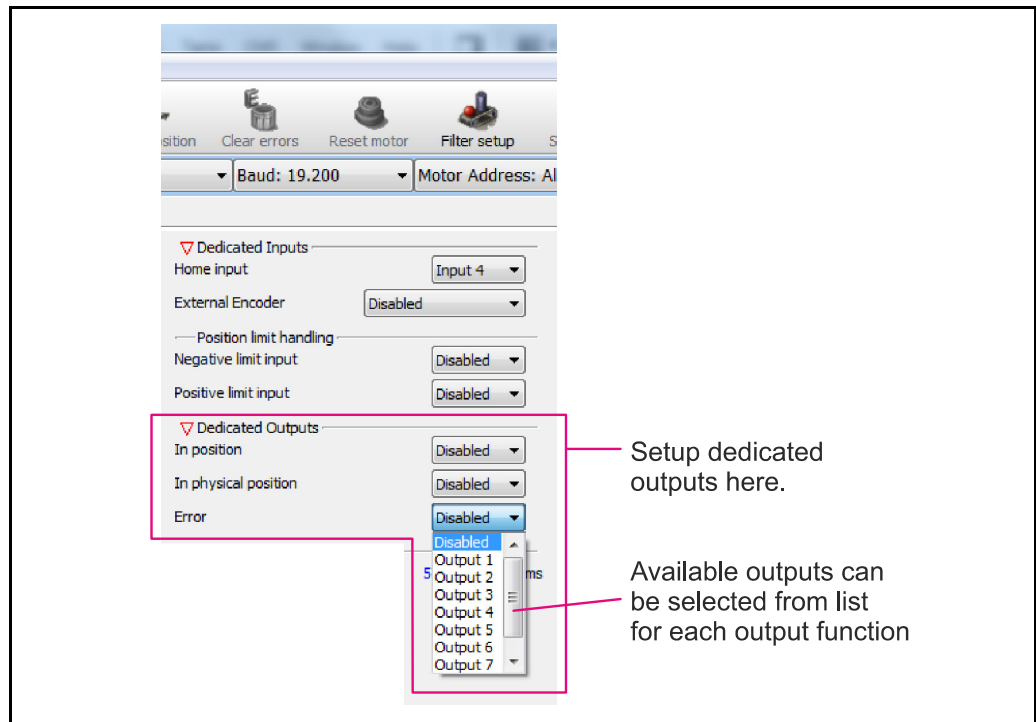
Which can also be done without MacTalk:

**R124 SETUP\_BITS** A general setup register for many settings in the motor. Only bit 2 and 3 are used for this setting, all other bits must not be changed. This table shows how the combo box in MacTalk is setting the bits:

Bit 2	Bit 3	Setting
0	0	None
0	1	Quadrature
1	0	Pulse/direction

## 5.6

# Dedicated outputs



### 5.6.1 Error Output

The internal flag that indicate when a fatal error have occurred can be copied to a physical output. This setup is done in MacTalk - please see illustration above or by setting a bit in register `Error_Mask`. See also : [Error\\_Mask](#), page 177.

This feature can be used for signalling to a PLC or other equipment in a motion control system that an error have occurred. Under normal operation the Error Output is active. If an error occur such as temperature too high the output is set to passive.

### 5.6.2 In Position Output

The internal flag that indicate when the motor has reached its target position can be copied to a physical output. This selection is done in MacTalk or by setting a bit in register 137 (bit 0-7) `InPos_Mask`, See also [Inpos\\_Mask](#), page 177.

Function at output: When the motor is running, the output will be inactive. When the motor has reached target position and is at stand-still, the output will be activated.

### 5.6.3 In Physical Position Output” (Only valid for H2, H3, H4 options)

This signal can be used only if the MIS motor is equipped with an internal encoder (H2, H3 or H4) or an external encoder for measuring the actual position of the motor.

This signal can be selected to be continuously updated and will then indicate if the motor is inside the “In Position Window” all the time.

If continuous update of the “In Physical Position” is not selected and the autocorrection is used, this signal is changed after a move and when a check has been made of the position after the “settling time between retries” if the motor is inside the “In Position Window”.

The signal can be copied to a physical output

This selection is done in MacTalk or by setting a bit in register 137 (bit 8). See also [Inpos\\_Mask](#), page 177.

## 5.7 SSI encoder/sensor interface

### 5.7.1 General information on how to connect the SSI device.

The SSI interface is based on 2 differential lines. Both lines are available in the MI2 connectors and are named AI +, AI- and BI +, BI- (4 wires) -  
In order to see the exact physical location of the signals please consult the pages:

- [Connector overview for the MIS motors](#), page 34

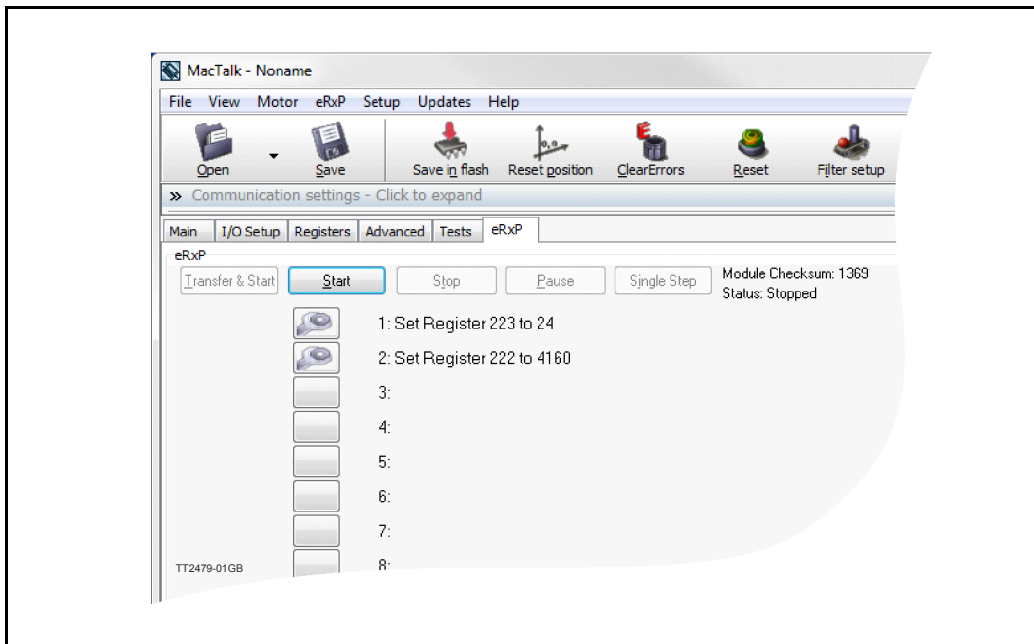
The function of the signals is as follows:

- Line **AI +** and **AI -** transmit a clock signal to the SSI device.
- Line **BI +** and **BI -** receives the data stream from the SSI device.

### 5.7.2 Setup and operation of the SSI function when using MacTalk.

When using the MacTalk Windows program supplied by JVL the following descriptions must be used.

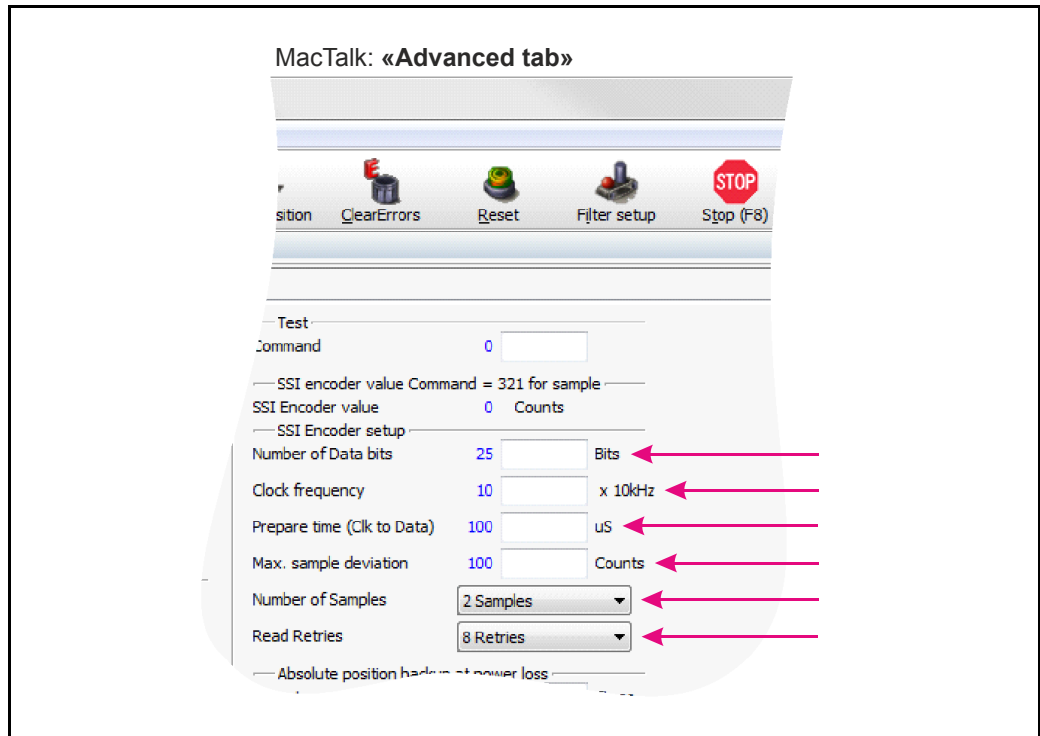
In order to setup for SSI support 2 codes need to be executed in a RxP program. These 2 commands setup the RS485 multifunction interface covering the 4 lines to the SSI device for transmitting a clock and receiving data from to/from the SSI device.





## 5.7 SSI encoder/sensor interface

From MacTalk all configurations and settings are accessible. Choosing the “**Advanced**”-tab gives access to the “SSI encoder value” and the “SSI encoder setup”.



### Field: “Number of Data bits”

Selects the number of data bits in each SSI transfer. The valid range is 1 to 31. Set this value according to the actual SSI device connected. Please consult the data sheet for the actual SSI device concerning which value to be used.

### Field: “Clock frequency”

Selects the maximum clock speed in units of 10 kHz. The valid range is 1 to 255, corresponding to 10 kHz to 2.55 MHz.

Set this value according to the actual SSI device connected. Please consult the data sheet for the actual SSI device concerning which value to be used.

### Field: “Prepare time” (Clk to Data)

A typical SSI device needs a so called prepare time to sample the position data before transfer. This field is dedicated to type in the prepare time in micro seconds at the start of an SSI transfer. The valid range is 1 to 255, corresponding to 1 to 255 micro seconds. Set this value according to the actual SSI device connected. Please consult the data sheet for the actual SSI device concerning which value to be used.

### Field “Max. sample deviation”

Selects the maximum allowed deviation between two samples. The valid range is 1 to 8191. This function is ment to be an extra safety to avoid invalid reading of position data caused by noise influencing the signal. Please bear in mind that if the external SSI device is tracking the position of something that moves the value and thereby the deviation from one sample to the next can be significant.

## 5.7 SSI encoder/sensor interface

### Field: “Number of Samples”

Selects the number of samples in each SSI measurement. If all samples stay below the “Max. sample deviation” value (described earlier in this text), no retry is required. If one pair of samples fails the whole measurement fails and a retry is attempted if allowed according to the “Read Retries” parameter.

### Field: “Read Retries”

Selects the number of retries before time out and reporting an error.

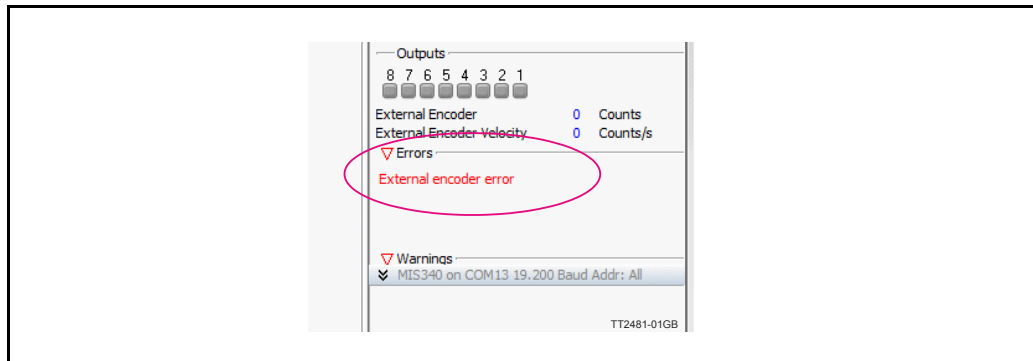
The MIS<sub>xxx</sub> / SMC66 and SMC85 has a build in data validation function which automatically compares the current sample with the previous and makes sure that the deviation is within the “Max. sample deviation” limit. “Number of samples” determines how many samples one measurement contains. If the measurement fails, a retry is attempted if number of retries has not exceeded the “Read Retries” value.

$$\begin{aligned} |\text{Sample}(x) - \text{Sample}(x+1)| < (\text{Max\_sample\_deviation}) &= \text{OK} \\ |\text{Sample}(x) - \text{Sample}(x+1)| > (\text{Max\_sample\_deviation}) &= \text{ERROR} \end{aligned}$$

If the number of retries has exceeded the “Read Retries” value and the measurement still reports an error, the reading will be discarded and “SSI Encoder value” still shows the last correct measurement.

An error bit will be set in the ERR\_BITS register (34) on position 11.

MacTalk will report this error :



### Additional hardware settings:

Some LIKA SSI encoders offer 2 additional hardware settings, for instance “Zero Setting” and “Invert Counting Direction”. These settings can be controlled by having user I/O 5 and 6 set as output. Consult the data sheet for the specific encoder to read more about the behaviour of these settings and to make sure they are available.

## 5.7 SSI encoder/sensor interface

### 5.7.3 Setup and operation of the SSI function when NOT using MacTalk.

Following lines describe how to access relevant registers when having a SSI device connected.

When reading the data from the SSI device the data will be placed as a signed 32 bit integer in register 47 shortly after the read command have been executed.

The time before data is present can be calculated after following formula:

Total time from the read command 321 is executed until valid data is present is equal:

$$\text{Time} = \text{Prepare time (timer)} + (1 / \text{Clock frequency} * (\text{Number of Data bits} + 1))$$

#### Example:

Encoder used have following setup:

Prepare time = 100uS (0.0001 sec.)

Clock frequency = 10kHz (10000Hz)

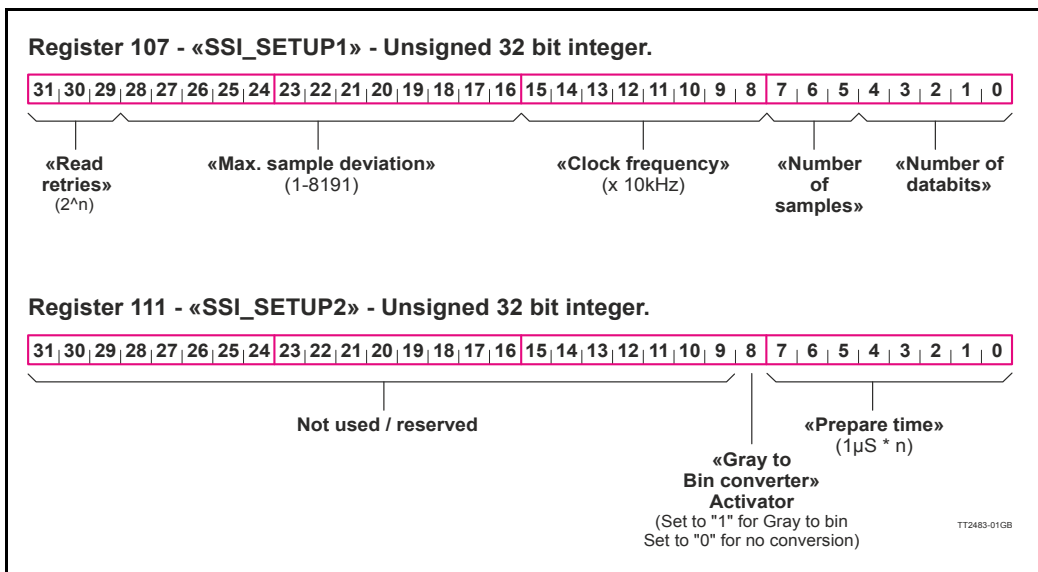
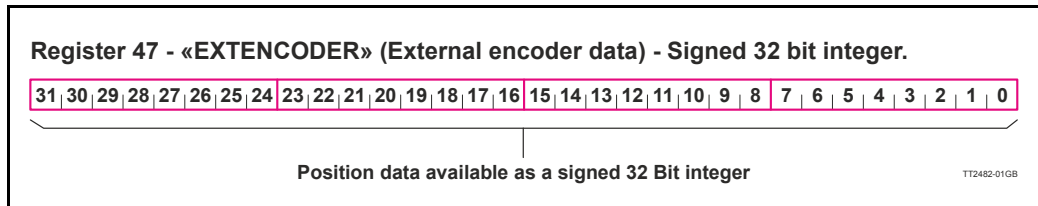
Number of data bits = 25

$$0.0001 + (1/10000 * (25 + 1)) = 0.0027 \text{ sec.} = 2.7\text{ms}$$

#### Note:

If noise have affected the signal the time before valid data is presented will be longer. Depending on the specific setting for the data validation function.

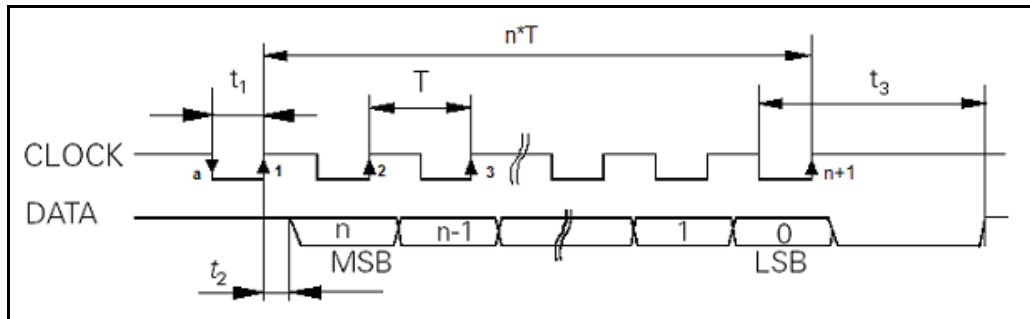
The position data from the encoder is presented in register 47 as shown below.



## 5.7 SSI encoder/sensor interface

### 5.7.4 The SSI interface principle of operation.

When the differential lines are used for SSI, the AI +/- lines work as a clock signal to the encoder, while the BI +/- signals work as a data signal from the encoder to the controller (MISxxx/SMCxx).



The figure above shows the SSI protocol principle.

The clock line is normally high. When a reading is requested, the clock goes low for  $t_1$  micro seconds (see illustration above) to allow the encoder to sample and prepare a value.

On the first rising edge of the clock (1), no sampling is done but on the second rising edge of the clock (2) the first data bit is read from the data line.

Shortly after reading the bit value, the MISxxx/SMCxx will set the clock high and execute another cycle, where the data bit is sampled just before each rising clock.

After the last data bit has been sampled, the clock stays high.

### 5.8.1 Introduction

Optional the MIS motors can be equipped with the absolute multi-turn encoder (MISxxx-zzyyH3/H4nn option) which offers the possibility to keep track of the position regardless if power is connected or not.

When this option is present a mechanical zero search is only necessary one time after installation and the system will keep track of the actual motor position at any time afterwards.

The encoder is based on a magnetic principle which do not offer the same resolution and precision as the more expensive optical based solutions. It do however offer the advantage to keep track of the position without power applied. The magnet principle is also much more tolerant to mechanical stress such as shock and extreme temperatures.

The absolute multi-turn option offers the following main features.

#### Encoder:

Resolution	409.600 counts per revolution displayed (internal 1024 cpr)
Accuracy	+/- 0,1% of full scale
Repeatability	+/- 0,1% of full scale
Position range	-5245 to +5245 motor revolutions (+/- $2^{31}$ motor counts)

#### Motor:

Resolution	409.600 counts per revolution (standard)
------------	--

As seen above the motor resolution is much higher than the encoder resolution. The encoder option is however mostly used for stall detection at the motor (the motor loose its position) and for this situation the encoder will be adequate since the motor can only stall in multiples of a 1/50 shaft revolution corresponding to 7.2 mechanical degrees.

## 5.8 Absolute Multi-turn Encoder

Only MISxxx---H3/H4--

### 5.8.2 How to Preset the encoder position.

After installing the motor it is normally desired to make an alignment of the encoder position which represent the "Actual position" compared to the actual physical position of the motor and its load.

This operation is recommended to do in the following manner.

1. Set the motor in a known position
2. Insert the corresponding position value as shown below.
3. Press the "Set position" button to preset the new position value.
4. The encoder position and all other relevant position registers are now preset with the new value. No further action is needed. The motor will remember this change also during power off.

Select the «Absolute encoder» tab

MacTalk - Noname  
Files Motor eRxP Setup Updates Help

Open Save Save in motor Reset position Clear errors Reset motor Filter setup STO

Serial port Comport: 1 Baud: 19,200 Motor Address: All

Main I/O Setup Registers Advanced Event Log Tests eRxP **Absolute encoder**

Absolute encoder position  
-2147483648 0 2147483647

Encoder position: 5694000

Change encoder position  
New encoder position:  
0 Set position

Information  
When the position reaches the limitation in the position register, the bar turns red to indicate that the position will wrap within a short distance. Special care on the position handling needs to be taken when wrapping has occurred.  
The position registers are 32bit signed and the motor resolution is 409600 counts/rev. This limits the travel distance to approx. 5243 motor revolutions in each direction.

First the new value that the encoder must be preset to must be typed here

Secondly press «Set position» in order to preset the encoder counter with the new value.

TT2339-01GB

## 5.8 Absolute Multi-turn Encoder

Only MISxxx---H3/H4--

### 5.8.3 Position "Auto-correction"

This feature is only active when the motor is in position mode.

The auto-correction feature is only used when the motor has stalled and not reached its final target position within the given position window.

Each time the motor has done a movement the "Actual position" counter and the "Encoder position" counter is compared.

If the difference without sign is within the value specified in the "In position window" as shown below no further action is taken.

If the difference is larger than the value in the "In position window" the motor will try to correct the position by doing a new motor movement. The "Max number of retries" is the number of times the motor will try to correct the position, and the "Settling time between retries" is the time the motor will wait between each retry.

When selected the in position flag will realtime indicate if the motor is within the position bus window compared to a perfect move.

TT2338-01GB

See also [Position "Auto correction"](#), page 72

## 5.8 Absolute Multi-turn Encoder

Only MISxxx---H3/H4--

### 5.8.4 Operation of the encoder when NOT using MacTalk.

The following description is useful for reading the encoder or presetting a new encoder value if MacTalk is not used for the communication.

This could for example be when using one of the Ethernet options.

#### Reading the encoder position.

The encoder position is read from the register 46 (AbsEncPos). The value is scaled to match the motor resolution which means 409600 counts per motor revolution.

#### Presetting a new encoder position.

Write the new encoder value to register 144 (P\_NEW) and afterward write 316 into the register 24 (Command reg). Notice that this value must match the motor resolution which means 409600 counts per motor revolution.

Optionally the desired position value (P\_SOLL and P\_IST) can be set to the same position by writing the value 119 to the register 24 (Command reg).

### 5.8.5 Updating the Encoder firmware

The the H3/H4 encoder contain a firmware which is programmed from factory.

This firmware normally follow the encoder through the products lifetime.

Bit if it have been lost for some reason or need an update because a newer firmware have been released having additional features it described in details how to do in [How to update the encoder FW](#), page 66.



## 5.8

# Absolute Multi-turn Encoder

Only MISxxx---H3/H4--

### 5.8.6 Position retention time

A special high temperature "Poly-carbon mono fluoride" lithium cell is used as backup for the absolute multi-turn encoder. Compared to standard cells this one has extremely good performance at high temperatures up to 125 degrees Celsius.

Normal lithium cells self-discharge very rapidly just being exposed to a temperature of 60 degree Celsius.

The retention time of the position when the motor is not supplied is better than **10 years**. This depends of how long time the motor is in use and the temperature.

The self-discharge at 85 degrees Celsius (185 degrees Fahrenheit) is 6% per year, and 2% per year at 40 degrees Celsius (104 degrees Fahrenheit).

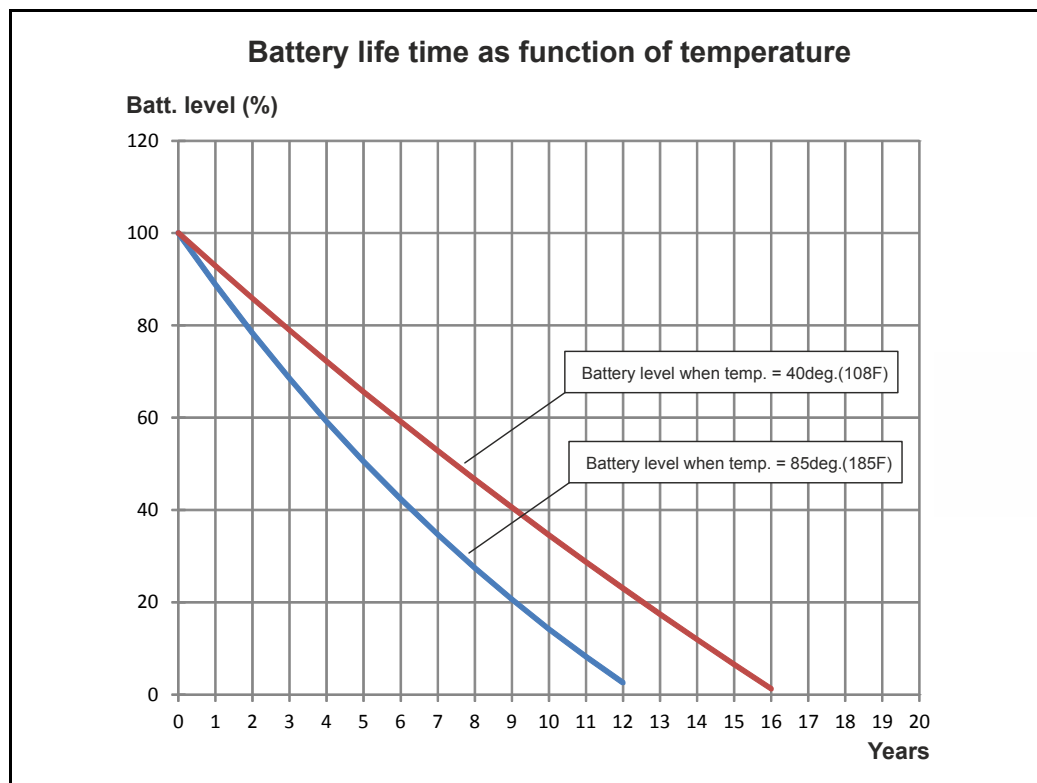
Concerning the retention time, there is no big difference if the motor is in use at a high temperature or it is powered down (low temperature).

The absolute multi-turn encoder's current consumption from the battery when the motor is not externally powered, is max.  $1.5\mu\text{A}$ .

The curves below shows what is considered as a worst case scenario for the retention time. The curves are based on 40 and 85 degree Celsius (108/185F) **without any external power applied to the motor** meaning that the internal battery have to supply current to the encoder circuitry all the time. The current consumption of the encoder circuitry is however not dominant compared to the internal leakage current in the battery.

#### Hints to optimise the battery lifetime:

1. Avoid to place the motor in an environment with high temperatures.
2. Set the running and especially the standby motor current as low as possible in order not to heat up the motor unnecessarily.
3. Keep the external power applied as much as possible.



## 5.9

# Position Limits

### 5.9.1 Position limitation features.

The MIS motor family offers 2 different methods of limiting the movement of the motor. In some applications it may be fatal or critical if the position of the motor passes a certain mechanical position range.

The 2 methods are as follows:

- **End of Travel Limit Inputs.**  
Limit switch inputs for detecting the physical position of the mechanics.
- **Software Position Limits.**  
Limits in software to prevent the motor to move outside a certain position range.

The next pages describe the function and how to use these 2 methods.

### 5.9.2 End-of Travel Limit Inputs

Any of the 8 general inputs (IO1 to IO8) can be used as limit inputs. The input can be set from MacTalk or via register *NL\_Mask*, page 175 or *PL\_Mask*, page 176.

#### Positive limit (PL)

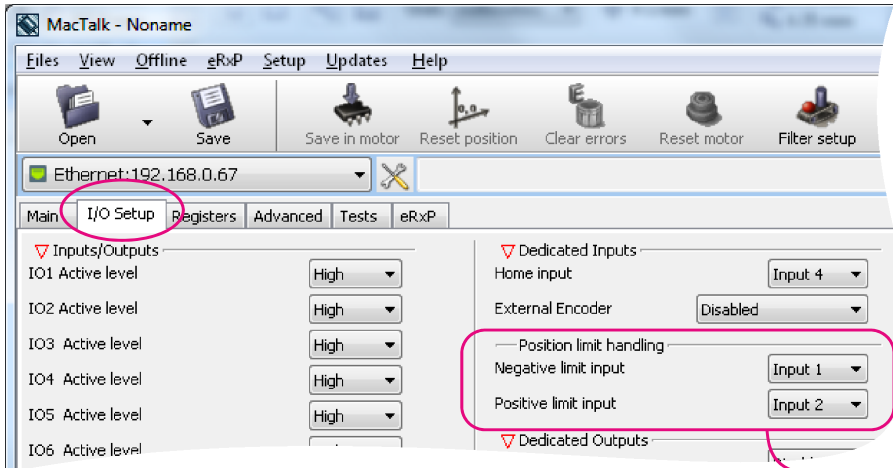
Activation of the Positive limit (PL) Input will halt motor operation if the motor is moving in a positive direction. The motor can however operate in a negative direction even if the PL Input is activated.

#### Negative limit (NL)

Activation of the Negative limit (NL) Input will halt motor operation if the motor is moving in a negative direction. The motor can however operate in a positive direction even if the NL Input is activated.

Below is shown how to select the desired input(s) to be used for the limit switch(es). Notice that the inputs default are set to disabled. Its also possible to select only one input for one of the directions and keeping the opposite input disabled. Please use the general chapter *I/O Setup tab*, page 60 for setting up the active level, optional input filter etc.

**How to select the input(s) for the End of Travel Limit**



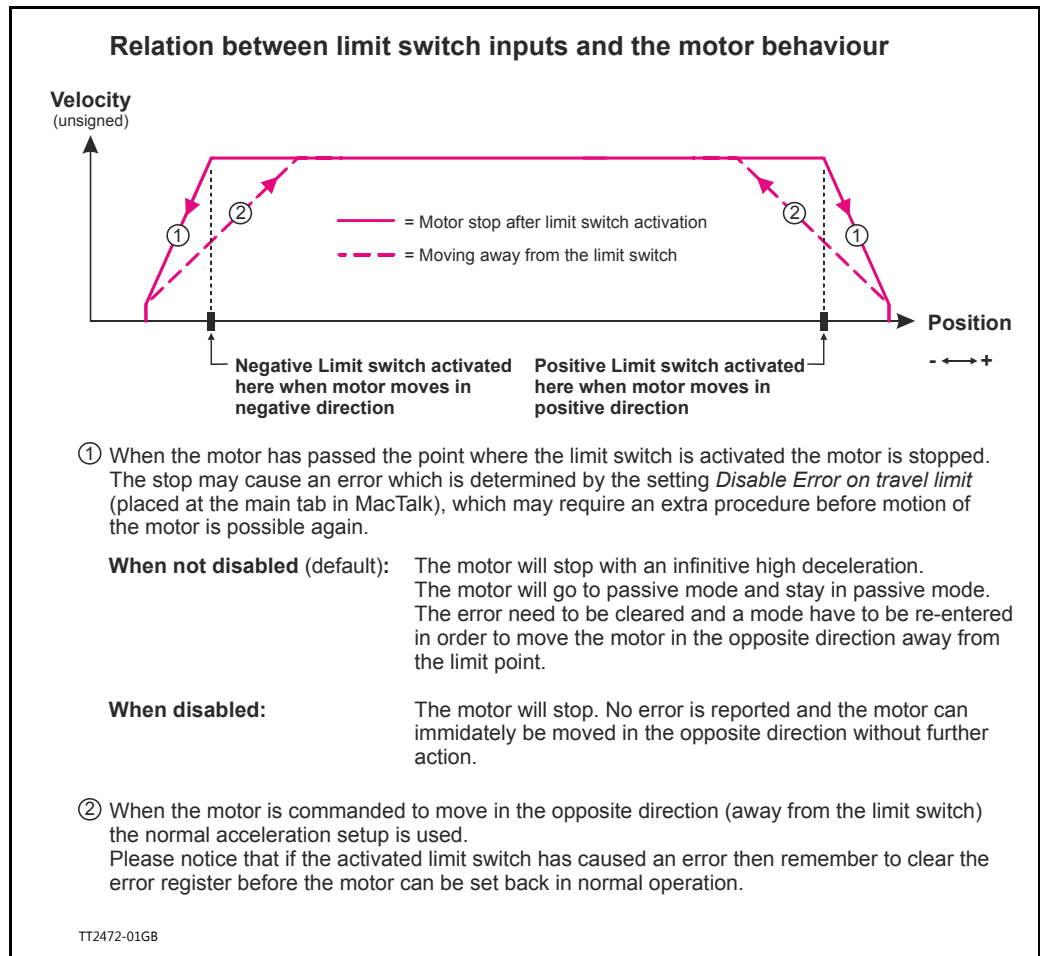
TT2471-01GB

Select the negative and/or positive limit input(s) in these fields.

## 5.9

# Position Limits

The following illustration shows the timing and motor behaviour when the limit switch inputs are activated.



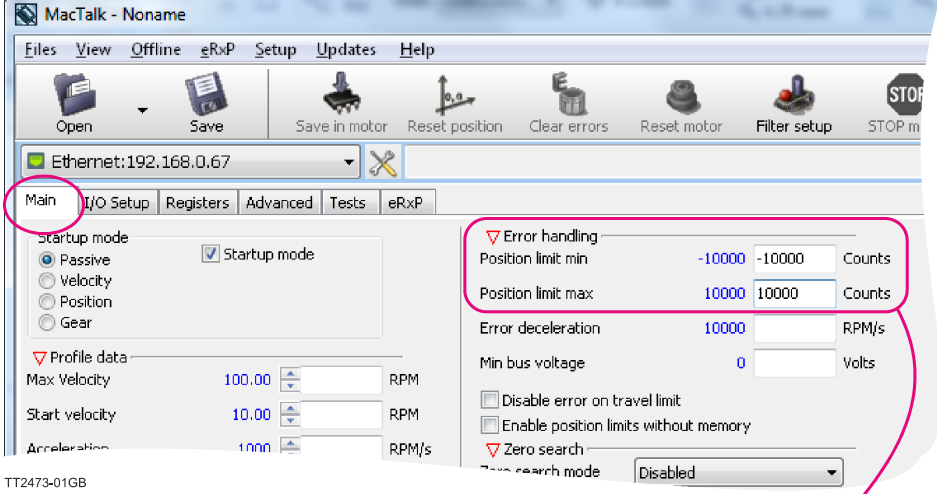
## 5.9

# Position Limits

### 5.9.3 Software Position Limits.

These limits are software limits and are can optionally be used to prevent the motor to move outside a certain position range in cases where for example faulty position commands are send to the motor or similar cases.

**How to setup the Position limits**



Select the negative and/or positive limit(s) in these fields.



**CAUTION** - Please notice that the motor may start to move if the position limits are changed after a situation where they have been passed and have forced the motor to stop. Its recommended to set the motor in passive mode before changing the limits in order to avoid personal injuries.

The limits has the same function as the physical End of travel inputs but is a pure software limitation. Default for both parameters is 0 meaning that the feature is disabled. Notice that if one of the parameters are different from 0, both values are activated.

#### **Positive Position Limit (PLS)**

When the motor is moving in a positive direction in position mode or gear mode, the motor will stop at Position Limit Max. In velocity mode the speed will internally be set to 0 when passing Position Limit Max, causing the motor to decelerate and stop.

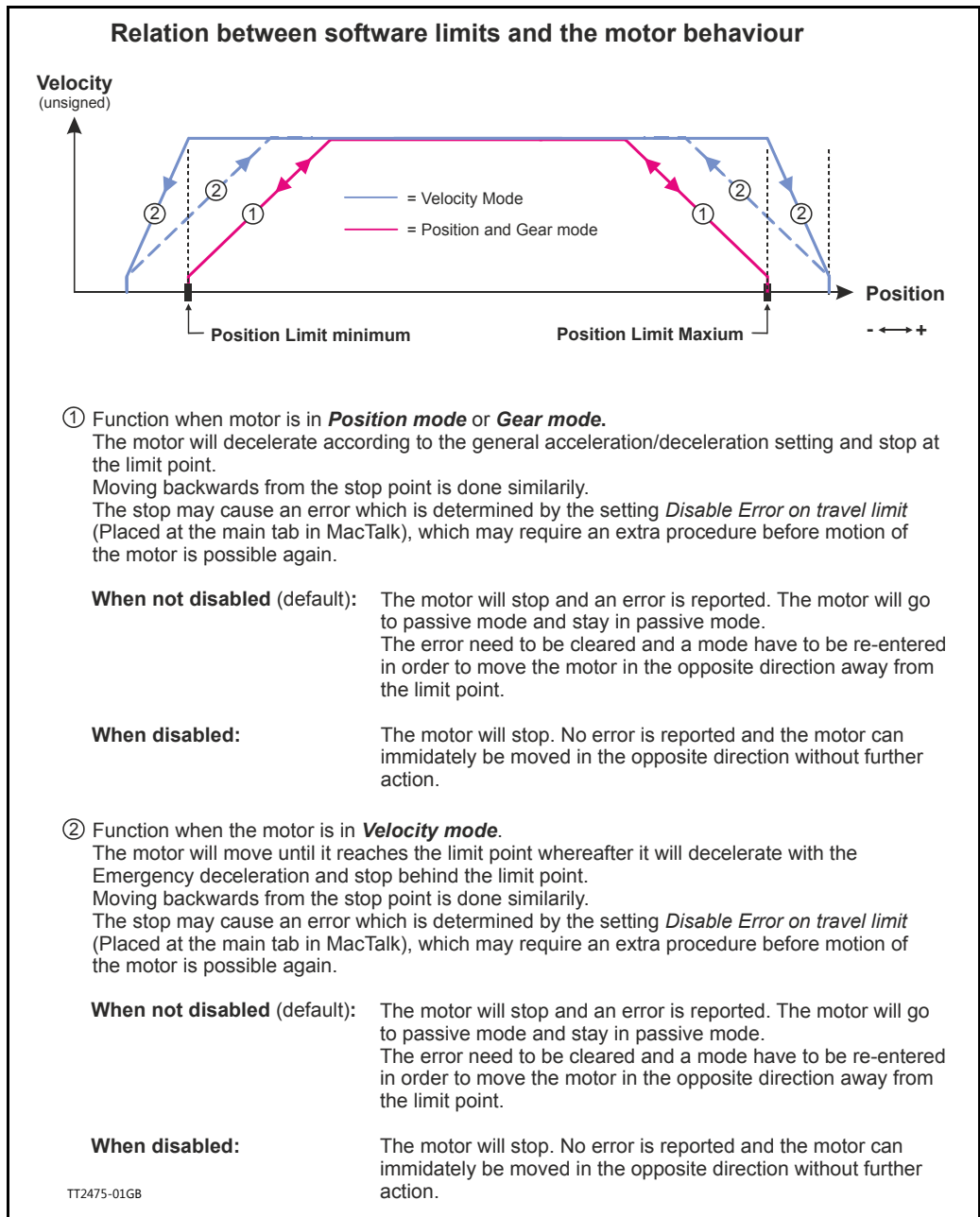
#### **Negative Position Limit (NLS)**

When the motor is moving in a negative direction in position mode or gear mode, the motor will stop at Position Limit Min. In velocity mode the speed will internally be set to 0 when passing Position Limit Min, causing the motor to decelerate and stop.

## 5.9

# Position Limits

The illustration below shows how the software limits take effect at the motor movement.



For further information about the internal registers that are behind the fields in MacTalk see also: [Min\\_P\\_IST](#), page 161 and [Max\\_P\\_IST](#), page 162.

## 5.9

## Position Limits

---

### 5.9.4 Limit Error handling

A bit will be set in the Controller's warning register if either the NL, PL, NLS or PLS has been activated or are active. See [Warn\\_Bits](#), page 163.  
Bits 0 and 2 are common for PL and PLS. Bits 1 and 3 are common for NL and NLS.

The motor will stop and activate an error, when reaching a limit. When a limit error is active the motor is forced into passive mode, and further movements are impossible. To reset the error press the "Clear Errors" button in MacTalk. Clearing errors automatically also clears warnings too.

### 5.9.5 Limit handling - optional

The MIS motors can be configured to stop and stay in the current mode when reaching a limit. This can be done by setting the *DisableErrorOnTravelLimit-bit* in the *SETUP\_BITS* register. See [Setup\\_Bits](#), page 174.

When the *NoErrorOnPositionLimit-bit* is set the motor decelerates with the "Error deceleration" on travel limits in all modes and on position limits in velocity mode. The normal acceleration is used on position limits in position and gear mode.

### 5.9.6 Simple mode: Position limits without memory

A very simple implementation of the hardware position limits sets the Error bit if limit has been reached, but is allowed to continue in the same direction when the Error has been reset and the motor has been set in an active mode.

Please be aware: When combining this mode with *DisableErrorOnTravelLimit-bit*, the motor will continue to run if it runs "over" the sensor because of too low deceleration. This can cause damage if not handled correctly by external logic (a PLC for instance).

"Position limits without memory" is enabled in the [Setup\\_Bits](#), page 174, **bit 28**.

## 5.10

## Under voltage Handling

---

The MIS motors offer the possibility to define the behaviour during and after the P+ bus voltage (main supply) disappear. This situation could for example be during an emergency stop, which causes the P+ supply to be cut while the control voltage (CVI) is still applied to the motor.

3 options available:

### **Under voltage -> Set error bit**

Default = ON.

If this option is selected an under voltage will be handled like an error situation and the corresponding error bit will be set. The motor is stopped using the "Error deceleration" before the motor is switched to Passive mode, like any other error situation. When P+ is re-applied the motor will stay in Passive mode and report an "Low bus voltage" error. To get the motor back in normal operation the error must be cleared and an operation mode must be selected. If this function is activated it will have first priority and the "Error deceleration" will be used compared to the 2 other options "Under voltage -> Stop controlled and go to passive" and "Under voltage -> Set velocity to 0" which both use normal deceleration.

### **Under voltage -> Stop controlled and go to passive**

Default = Off.

This option makes the motor decelerating according to the normal acceleration parameter and go to Passive mode when P+ is removed. When P+ is re-applied the motor stays in Passive mode. To get the motor back in normal operation an operation mode must be selected.

### **Under voltage -> Set velocity to 0**

Default = Off.

This option simply just sets the velocity to 0 causing the motor to decelerate and stay stationary when P+ is removed. The velocity setting will stay at 0 also after P+ is re-applied. A velocity value (>0 RPM) must be written into the velocity register to get the motor moving again.

*Continued next page*

# 5.10 Under voltage Handling

## 5.10.1 Setup with MacTalk

The 3 options that define the behaviour of the motor when P+ bus voltage is lower than what is set in the “Min bus voltage” field can all be accessed from MacTalk as shown below.

The screenshot shows the MacTalk software interface with the following settings:

- Startup mode:**  Passive,  Startup mode,  Velocity,  Position,  Gear
- Profile data:** Max Velocity: 100.00 RPM, Start velocity: 10.00 RPM, Acceleration: 1000 RPM/s, Deceleration: 0 RPM/s
- Driver parameters:** Running current: 3.00 A RMS, Standby current: 0.75 A RMS, Standby time: 500 ms
- Error handling:** Follow errors: 0 Counts, Position limit min: 0 Counts, Position limit max: 0 Counts, Error deceleration: 10000 RPM/s, **Min bus voltage: 15 Volts**
- Undervoltage handling:**  Undervoltage -> Set error bit,  Undervoltage -> Stop controlled and go to passive,  Undervoltage -> Set velocity to 0
- Communication:** Motor address: 254

The undervoltage level is defined in this field. Default = 15 Volt.

The 3 fields that define which action that should be taken in case of undervoltage



## 5.10 Under voltage Handling

---

### 5.10.2 Setup without MacTalk

If MacTalk is not used for setting up parameters and registers related to the under voltage feature it must be done as follows.

The motor contains a number of registers which can be accessed from various protocols depending at which options the motor has.

Protocols available are for example Ethernet (EthernetIP, Profibus etc.) and CAN-open, Modbus or the MacTalk protocol.

Each field in MacTalk described earlier in this chapter is accessing a register in the motor.

The registers that are relevant for the under voltage feature are:

<b>R98</b>	MIN_BUSVOL	The voltage level that defines when an under voltage situation is present. See also: <a href="#">Min_Busvol</a> , page 167
<b>R124</b>	SetupBits	SetupBitsThe 3 bits that define what action should be taken in case of an under voltage situation. <b>Bit 21:</b> Under voltage -> Set error bit <b>Bit 22:</b> Under voltage -> Stop controlled and go to passive <b>Bit 23:</b> Under voltage -> Set velocity to 0 See also: <a href="#">Setup_Bits</a> , page 174

# 5.11 Electro Mechanical brake

## 5.11.1 Brake Introduction

The motor can be equipped with a electro mechanical brake to hold the position in Passive mode and power off situations where the motor has no torque. This is often desired to keep mechanics in position for example if it's a vertical movement.

The brake control always takes care that the brake is activated (hold the motor) if a situation occurs where the motor is not powered and therefore can slip away from the desired position.

This will typically be in situations like when the motor is in Passive mode or an error has occurred which will cause the motor to be power less and not able to keep its position in a controlled manner.

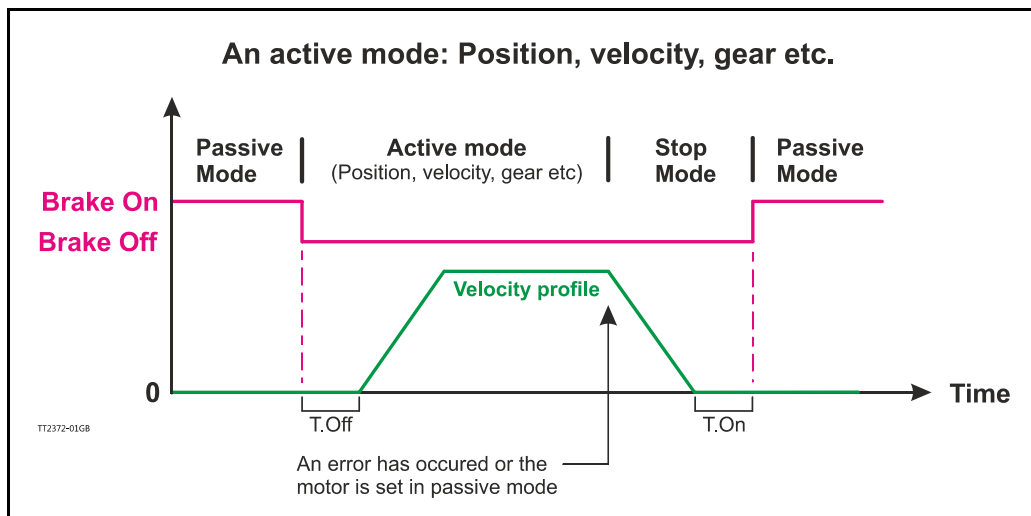
No involvement from users side is needed to activate and de-activate the brake.

It is also possible to connect an external brake via one of the user outputs I/O 1 to 8.

## 5.11.2 Brake timing

The brake is always active in Passive mode because the motor has zero torque. When changing to an active mode, the standby current is applied and the brake is disabled. The brake needs some milliseconds to release and therefore there will be a short delay (typically ~40 ms) before the motor can move. The brake is always off in active modes.

When changing to Passive mode the motor goes into Stop mode to decelerate according to the "Deceleration"-ramp. When "Actual velocity" is 0, the brake is activated and also here a short delay makes sure that the brake is active before the motor goes passive.



## 5.11.3 Connecting an external brake

An external brake can also be connected to one of the eight user outputs I/O 1 to 8.

The external brake will be operated in parallel with the internal brake without any time delay.

MacTalk do not support setup of the external brake but an internal register is available for setting up the brake. Please refer to [Access without MacTalk](#), page 112

## 5.11 Electro Mechanical brake

### 5.11.4 Brake registers

2 registers in the motor are relevant for the operation of the brake.

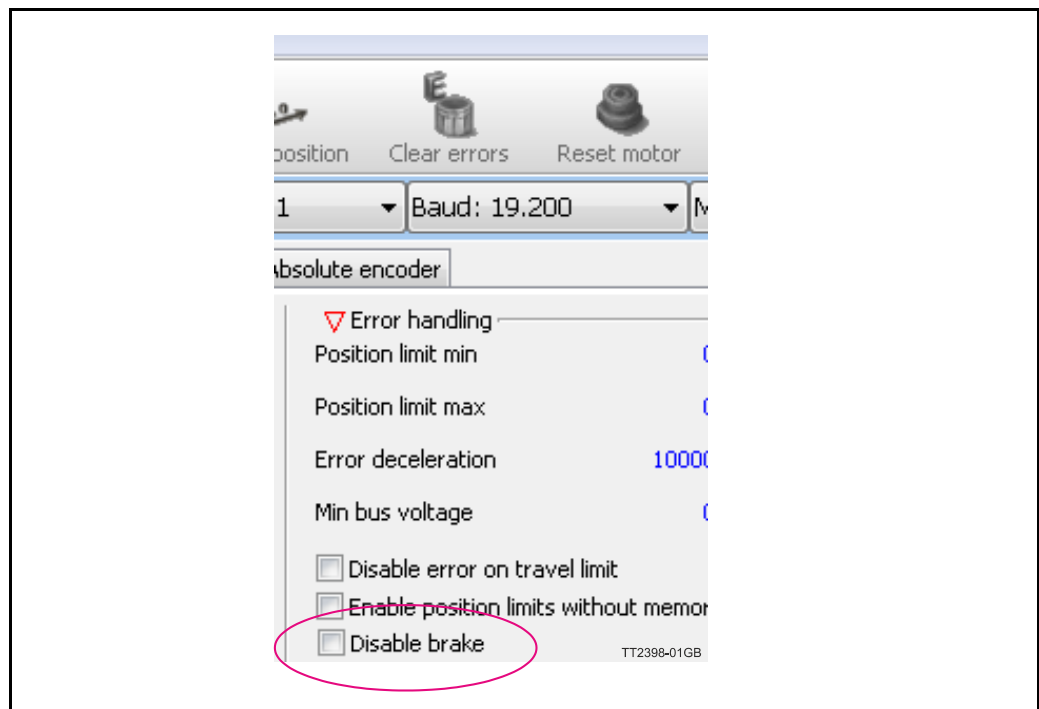
One register makes it possible to disable the brake so that the motor can run freely regardless which mode or condition the motor is in.

The other makes it possible to verify if the brake is active or passive.

### 5.11.5 Access from MacTalk

A field named "Disable brake" at the main tab in MacTalk makes it possible to disable the brake.

In the status bar at the right side its also possible to see whether the brake is active or not.



## 5.11 Electro Mechanical brake

---

### 5.11.6 Access without MacTalk

If MacTalk is not used for setting up parameters/registers related to the brake feature it must be done as follows.

The motor contains a number of registers which can be accessed from various protocols depending which options the motor has.

Protocols available are for example Ethernet (EthernetIP, Profibus etc.) and CAN-open, Modbus or the MacTalk protocol.

Each field in MacTalk described earlier in this chapter is accessing a register in the motor.

The registers that are relevant for operating the brake are:

#### **R25 STATUS\_BITS** **MacTalk name:** N/A

Bit 14 in this register monitors the actual brake status. If Bit 14 is 0 the brake is not active (motor can run freely) and if bit 14 is 1 the brake is active and keeps the motor in position.

Notice that other bits in this register are used for other purposes. See also: [Status bits](#), page 160

#### **R124 SETUP\_BITS** **MacTalk name:** Disable brake

Bit 19 in this register set to 1 will disable the brake which will allow the machine operator to move the shaft.

Notice that other bits in this register are used for other purposes. See also: [Setup\\_Bits](#), page 174

Connect an external brake

#### **R179 BRAKE** **MacTalk name:** N/A

The user outputs I/O1-8 can be used to control an external brake. The BRAKE register sets this up:

Bit 0-7: **Brake output mask**  
Defines which of the eight outputs that controls the brake.

Bit 8-15: **Brake\_T\_ON** (ms)  
Brake turn on time

Bit 16-23: **Brake\_T\_OFF** (ms)  
Brake turn off time

See also: [Status bits](#), page 160

The QuickStep motor offers the following modes of operation:

**Passive mode:**

The motor will be in a completely passive state but communication is active and internal registers can be set up.

**Velocity mode:**

The motor velocity can be controlled using MacTalk software or by setting register 5 (V\_SOLL) using serial or program commands.

**Position mode:**

The motor position can be controlled using MacTalk software or by setting register 3 (P\_SOLL) using serial or program commands.

**Gear mode:**

The motor position and velocity can be controlled by pulse and direction or encoder signals at IN1 and IN2.

The gear ratio can be set to a large ratio using register 14 (GEAR1) and register 15 (GEAR2).

**Zero search mode type 1 and type2:**

Searches for sensor to define a zero position (Reference point).

**Cyclic Synchronous Position mode (CSP):**

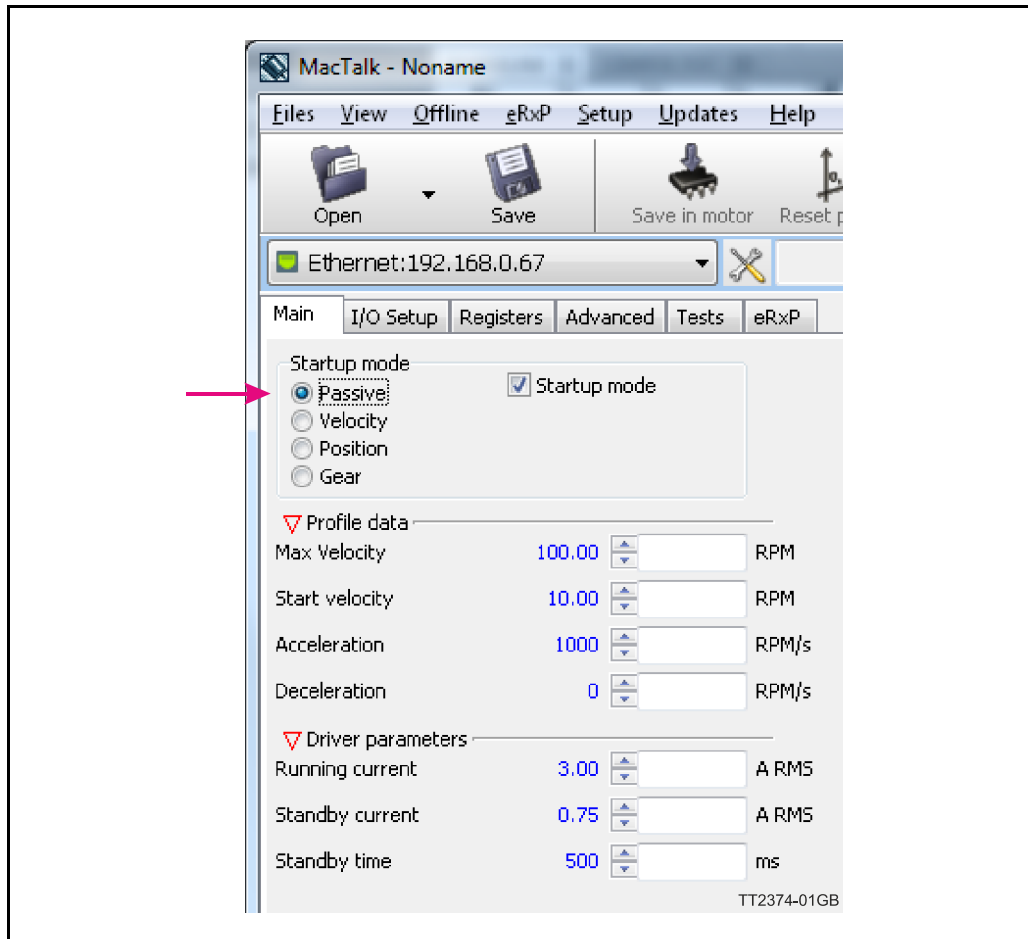
This mode is entered by the Ethernet module and enables very accurate positioning from the PLC. Cycle times down to 1 ms are supported and means that P\_SOLL is updated once every ms. Please consult the Ethernet manual for details.

## 6.1

# Passive Mode

### 6.1.1 Passive Mode

After power up, the controller will start up in passive mode. This means that it is possible to communicate and read/write to/from registers, but no current is supplied to the motor. It should thus be possible to turn the motor shaft as no voltage is connected to the motor. If there is encoder feed-back, the encoder counter will always register the correct position.

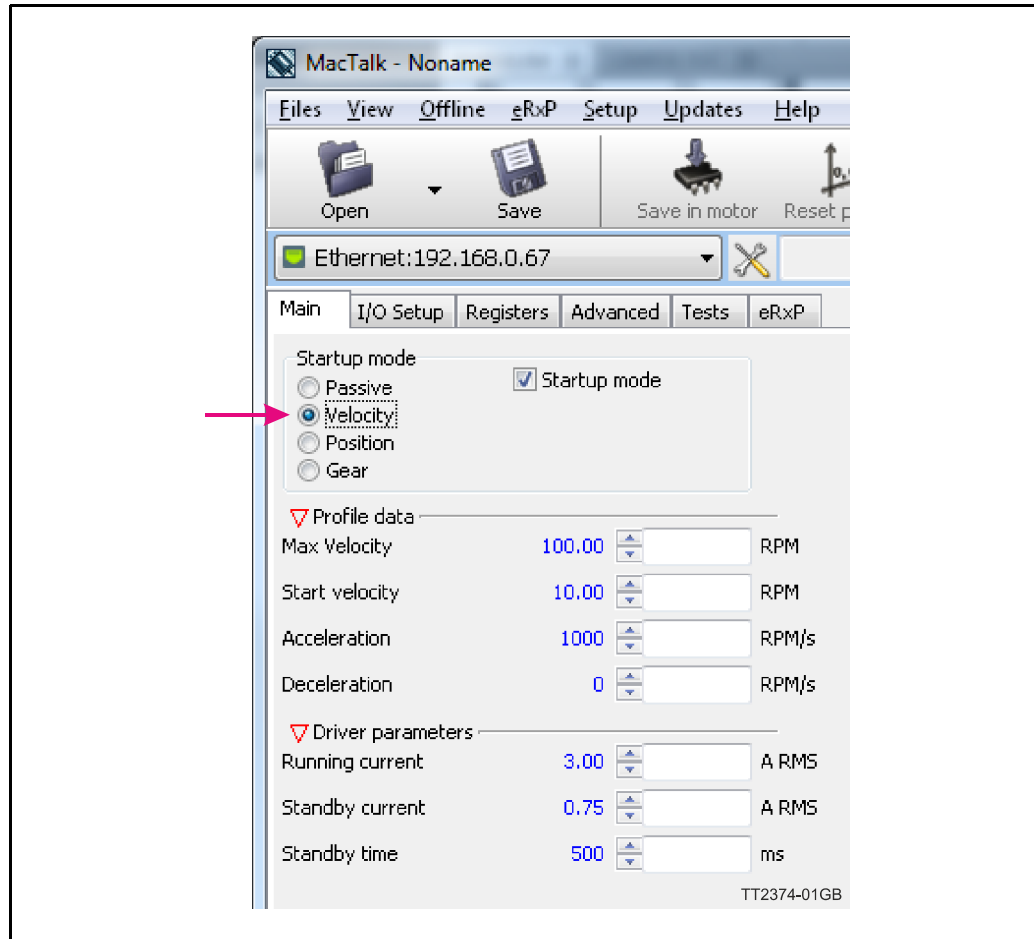


## 6.2

# Velocity Mode

### 6.2.1 Velocity Mode

In this mode, the QuickStep motor controls the motor velocity via the Max Velocity setting. This mode is typically used for simple tasks or for applications in which an overall unit, such as a PC-board or PLC, controls velocity and positioning.

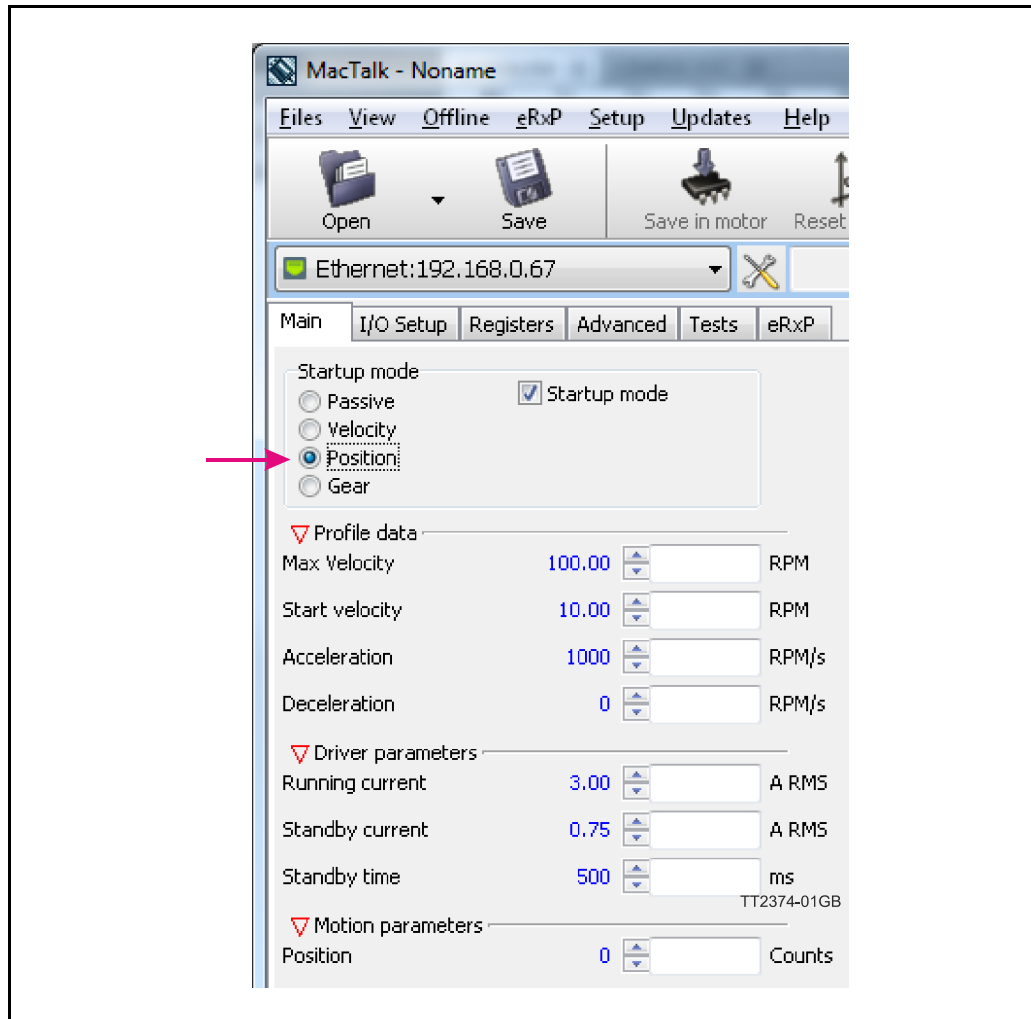


## 6.3

# Positioning Mode

### 6.3.1 Positioning Mode

In this mode, the QuickStep motor positions the motor via commands sent over the serial interface. Various operating parameters can be changed continuously while the motor is running. This mode of operation is used primarily in systems where the Controller is permanently connected to a PC/PLC via the interface. This mode is also well suited for setting up and testing systems. The mode is also used when programming is done.





## 6.4

# Gear Mode

### 6.4.1 Gear Mode.

In this mode, the QuickStep motor functions as in a step motor driver. The motor moves one step each time a pulse is applied to an input.

Velocity, acceleration and deceleration are determined by the external frequency but can be limited and controlled by the QuickStep motor. In addition, the QuickStep motor also provides a facility for electronic gearing at a keyed-in ratio in the following intervals:

- **MIS17x, 23x, 34x, and 43x:** 1/2147483647 to 2147483647/1.

**Main parameters used in Gear Mode**

Select gear mode here.

Make sure that all these parameters are set to proper values in order not to cause any limitations in the motors ability to move.

Insert the resolution for the pulse source in this field.

Insert the resolution for the motor in this field.

Notice that if other ratios than 1:1 between pulse source and motor is desired either the input and/or output value must be scaled to match the desired ratio.

This value reflect the target position (P\_SOLL) which is controlled directly from the pulse source. This value should normally match the actual position unless the motor has stalled or some of the motion parameters have been set to a limiting value.

TT2462-01GB

#### Example:

A MIS231 motor has a resolution of 409600 steps/rev. and an encoder is connected which has a resolution of 2048 counts/rev.

If one revolution of the encoder should result in one motor revolution the Input must be set to 2048 and the Output to 409600.

If the motor must run 5 revolutions because there is a gear with a reduction of 5:1 the output must be set to  $5 \times 409600 = 2048000$  instead.

## 6.4

# Gear Mode

### 6.4.2 Signal formats supported.

If gear mode is selected an external pulse source can control the position of the motor. Following 2 formats are supported in all the MIS motors:

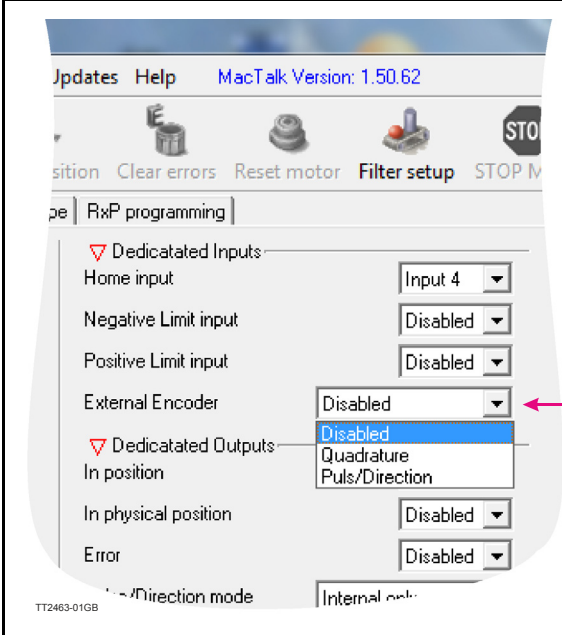
#### 1. Pulse and direction control

One input is applied with a pulse signal. Each rising edge at this input will cause the motor to move with a certain ratio (length) according to the gear registers “input” and “output” described at the previous page (see [Gear Mode.](#), page 117).  
A secondary input controls in which direction the motor moves.

#### 2. Quadrature control

When selecting this format 2 square wave signals (channel A and B) 90 degree phase shifted is applied to 2 inputs. Each transition (count) at the A or B channel will cause the motor to move with a certain ratio (length) according to the gear registers “input” and “output” described at the previous page (see [Gear Mode.](#), page 117).

The formats can be selected in MacTalk at the “I/O setup” tab.



**How to setup the input format.**

Select input format here.

- Quadrature is typically used for incremental encoders and supports a 2 channel 90 degree phase shiftet signal. The direction is defined by the polarity on the phase shift.
- Pulse/direction is typically used as format in stepper motor systems. A pulse signal is applied to one input and the direction to another input.

## 6.4

# Gear Mode

### 6.4.3 How to connect the pulse source.

The present firmware require that the external pulse source is applied to specific inputs. These inputs differ from motor family to motor family.

Other input options are under development.

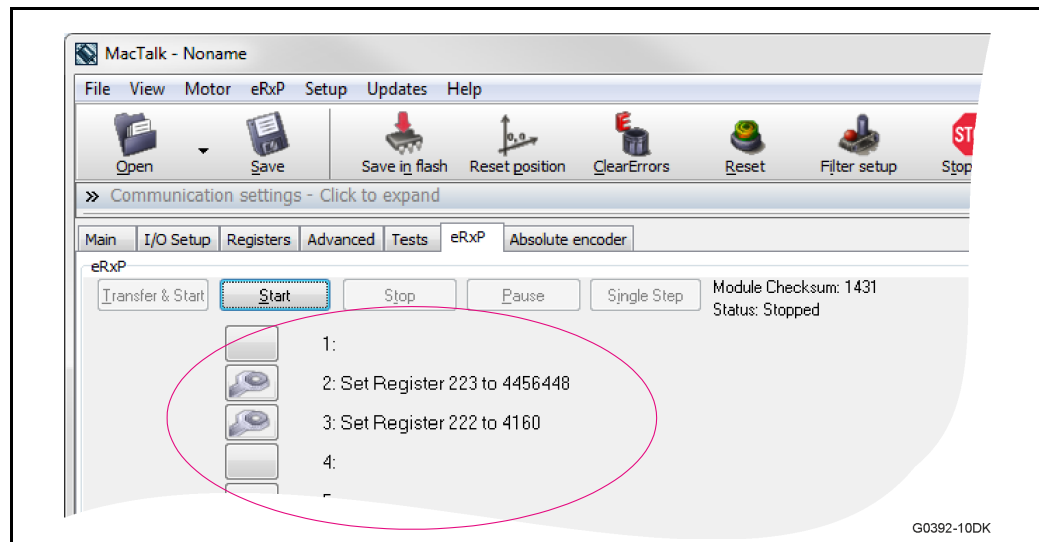
The external source must be connected to the following inputs:

Quadrature format	I/O terminal	Pulse/direction format	I/O terminal
Channel A	<b>A1+</b> and <b>A1-</b>	Pulse	<b>A1+</b> and <b>A1-</b>
Channel B	<b>B1+</b> and <b>B1-</b>	Direction	<b>B1+</b> and <b>B1-</b>

### 6.4.4 Single ended signals

If the external encoder signals are only available as single ended and with voltage levels up to CVO (typical 24V) it is possible to use IO2+3 as inputs.

To make this change, a small RxP program is required in addition to the settings in [Signal formats supported.](#), page 118 Signal formats supported:



Continued next page

## 6.4

# Gear Mode

---

After this program has been executed, the external source must be connected to the following inputs:

Quadrature format	I/O terminal	Pulse/direction format	I/O terminal
Channel A	<i>I02</i>	Pulse	<i>I02</i>
Channel B	<i>I03</i>	Direction	<i>I03</i>

Please note that the MIS motor family offers different connector configurations. For specific details about at which connectors the I/O terminals are available please consult: - [Connector overview for the MIS motors](#), page 34.

### 6.4.5 General considerations concerning cabling.

It is strongly recommended that shielded cable is always used when connecting the external pulse source to the pulse input to ensure that no noise from the surroundings affect the quality of the signal and worst case cause the motor movement to get affected. Also a good, solid ground wire between the motor and the source is recommended since any differences in the potential between the pulse source and the motor can affect the motor movement.

General guidelines concerning the I/O's are given in the following chapters:

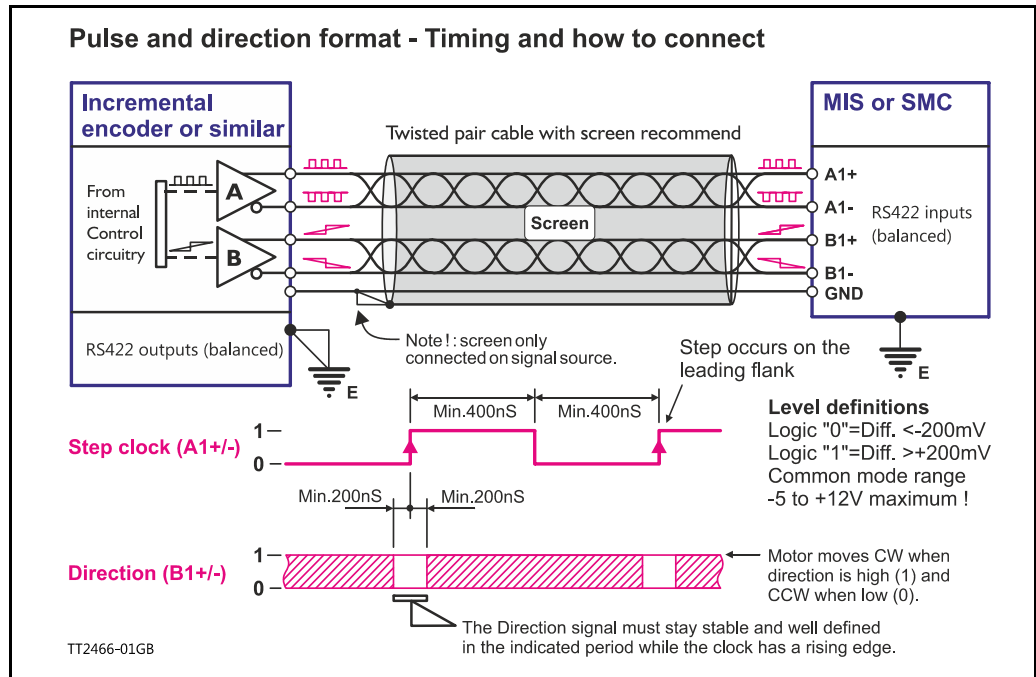
- [User Inputs](#), page 18 or [General](#), page 29.

## 6.4

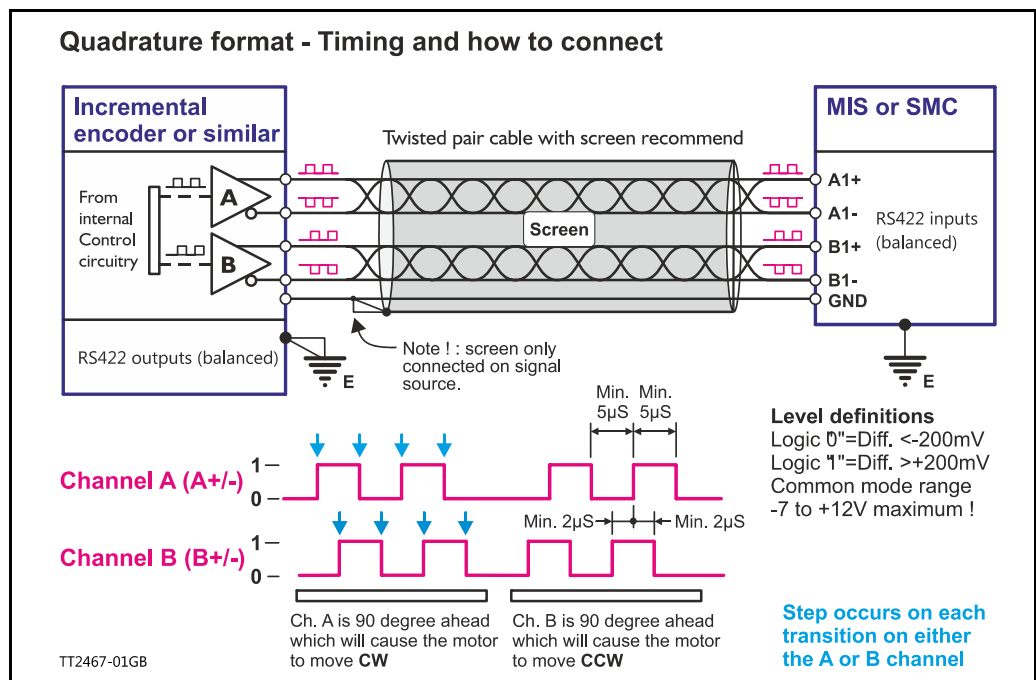
# Gear Mode

### 6.4.6 Signal function and timing.

The description below shows how to connect the pulse source when using the pulse and direction format. Also the timing is shown. Please be aware that if the indicated minimum timing is not respected the motor may lose some of the step clocks and the position of the motor will end up being out of synchronism with the pulse generator.



The description below shows how to connect the pulse source when using the quadrature format. Also the timing is shown. Please be aware that if the indicated minimum timing is not respected the motor may lose some of the step clocks and the position of the motor will end up being out of synchronism with the pulse generator.



## 6.4

# Gear Mode

### 6.4.7 Principle of gear mode.

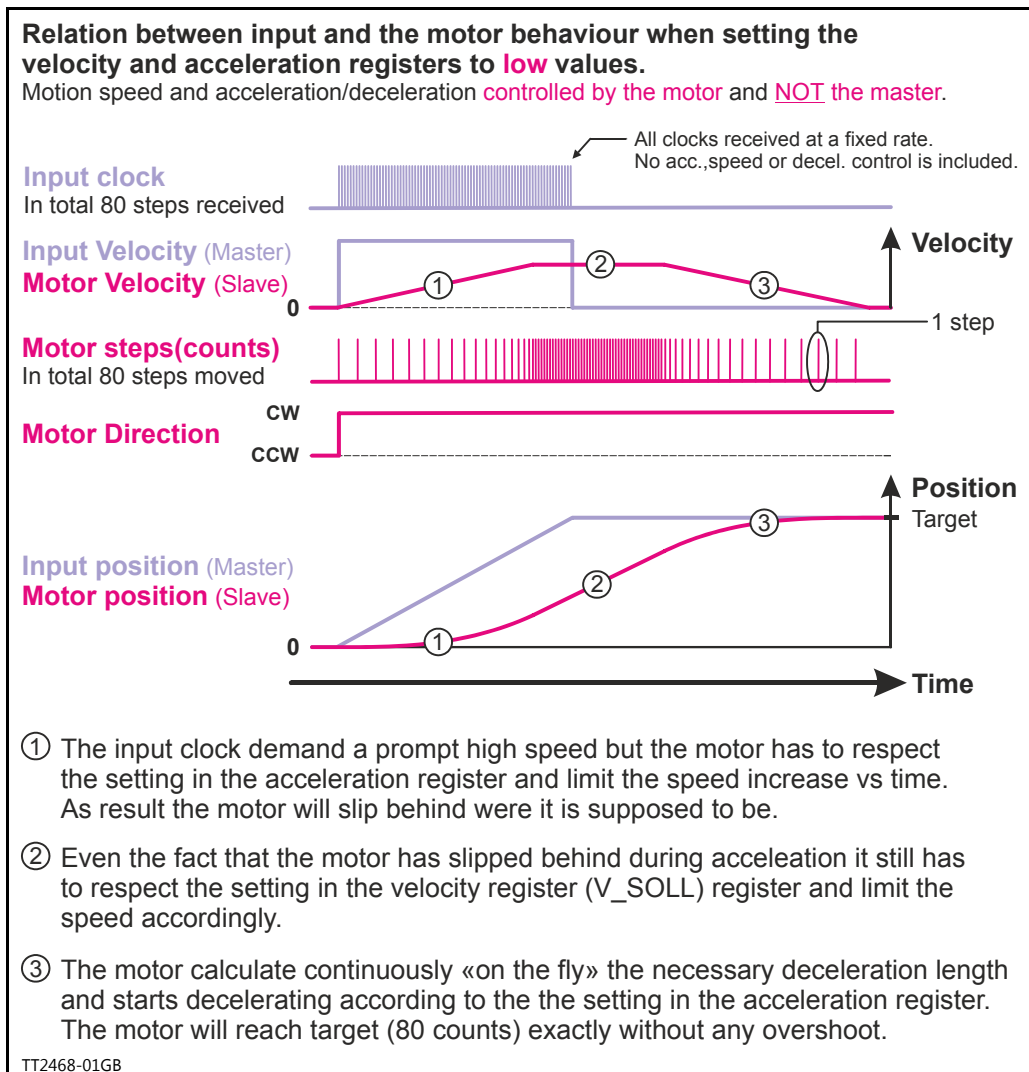
As mentioned on the previous pages the motor will follow the input signal synchronously to a certain extent according to the gear ratio setup. There are, however, a few other parameters which are vital for success in the actual application.

These parameters are:

**Velocity** The value of the velocity register will be respected at any time when the motor is in gear mode. Therefore, care must be taken if this is set lower than the equivalent speed of the external source producing clocks to the gear input since it will be a limiting factor.

**Acceleration** Similar to the velocity register the acceleration setting will be respected at any time and if set lower (slower speed rise time) than the external source producing the clocks it will start to be a limiting factor.

Below can be seen an example of the relation between applied clocks and the actual motor movement when having velocity and acceleration set to low values.

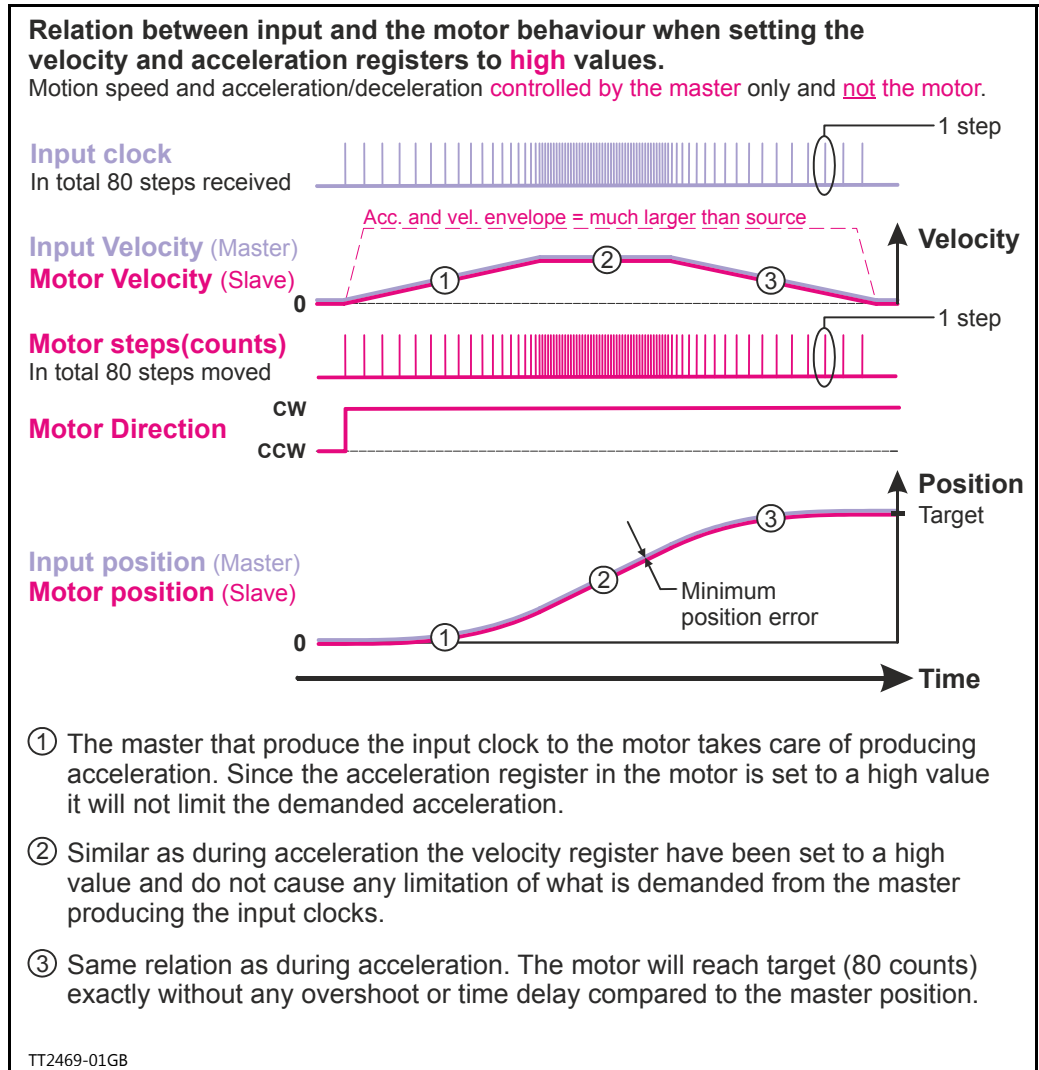


## 6.4

## Gear Mode

As an alternative to the previous illustration the full control can be done by the master by setting the velocity and acceleration registers to a significant higher value than the equivalent value of the clock source which will assure that the motor follows each clock with a very narrow timing and no delays.

This solution must be used if the master that produces the clocks do generate the full motion profile with acceleration to a desired top speed and make sure to decelerate and hit target.



## 6.4

# Gear Mode

### Example 1: Encoder (quadrature) input.

An external encoder feeds the MIS. The I/O type is set to “Pulse input” and “Input type” is set to “Quadrature” in order to decode the encoder signal. The encoder is connected to the 2 pulse input terminals.

See also [How to connect the pulse source.](#), page 119.

The resolution of the external encoder is 2048 cpr.

The MIS motor itself has 409600 cpr. If this application requires that the MIS motor rotates 1 rev. each time the external encoder has rotated 1 rev., the *Input* parameter is set to 2048

(external encoder) and the *Output* parameter is set to 409600.

Now the ratio between the external encoder and the MIS motor will be 1:1. Ensure the “Profile data” is set to proper values in order not to limit motor operation unintentionally.

### Example 2: Pulse and direction input.

A traditional step motor system with separate driver and motor is replaced by the integrated MIS motor, meaning that the MIS motor receives a pulse and direction signal which is a very common signal format in step motor applications.

The I/O type is set to “Pulse input” and “Input type” is set to “Pulse-direction” in order to decode the input signal. The pulse signal is connected to the 2 pulse input terminals. See also [How to connect the pulse source.](#), page 119.

The MAC motor is replacing a step motor system with 400 steps per revolution, which means that when the pulse source produce 400 pulses, it expects the MIS motor to rotate one revolution.

The MIS motor itself has 409600 cpr. If this application requires that the MIS motor rotates 1 revolution each time 400 pulses are received, the *Input* parameter is set to 400 since the MIS motor interpret every rising edge at the applied pulse signal as one count (step). The *Output* parameter is set to 409600 since this is the number of counts (steps) on one revolution.

Now the MIS motor will move 1 revolution for every 400 pulses that are applied to the pulse input. Ensure the “Profile data” is set to proper values in order not to limit motor operation unintentionally.

The following table can be used as guide for setting up typical gear ratios:

Pulse and direction gear ratio setup - “Commonly used ratios”		
Applied number of pulses(clocks) per desired MIS motor revolution.	“Input” register	“Output” register
200	200	409600
400	400	409600
500	500	409600
800	800	409600
1000	1000	409600
1600	1600	409600
2000	2000	409600
409600 (Equal to the MISxxx resolution)	409600	409600

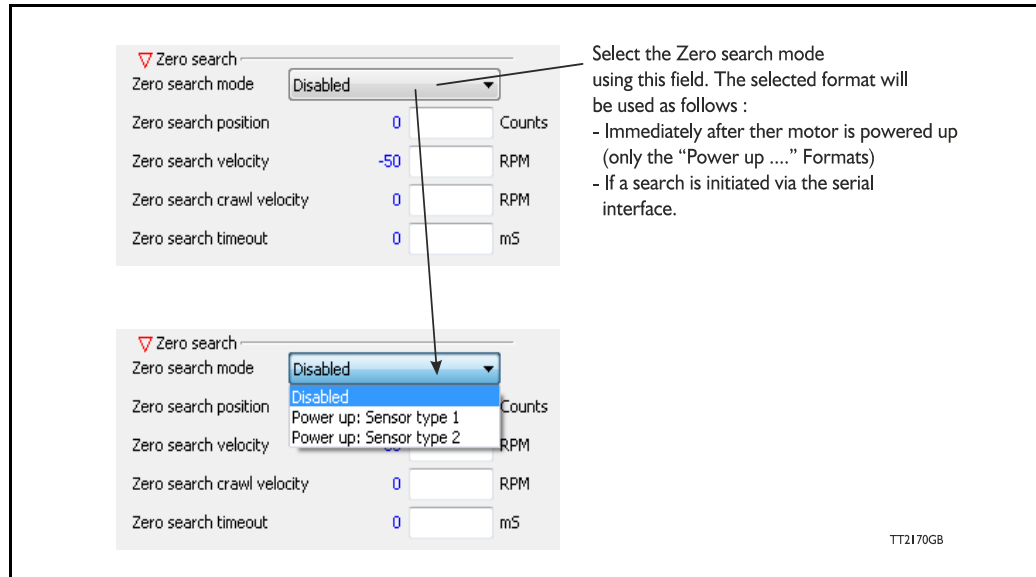


## 6.5

# Zero search modes

### 6.5.1 Mechanical zero search modes

In all positioning systems there is a requirement to be able to find a mechanical zero position after the system is powered up or at specific times during operation. For this purpose the MIS motor offers 2 different Zero search modes which can be selected from the MacTalk main window or by sending a command via one of the serial interfaces.



The menu offers 3 options:

**Disabled** (default)

The Zero search is disabled.

**Power up: Sensor type 1**

The Zero search function will start seeking for Zero until an external sensor is activated. The point at which the sensor is activated is defined as the zero.

**Power up: Sensor type 2**

Like above (Sensor type 1) but after the sensor is activated the direction of movement is reversed and the point at which the sensor is disabled is defined as zero.

The following sections explain in detail the functionality of the 2 fundamental Zero search modes.

### 6.5.2 Starting a Zero search

If the Zero search mode is set to *Disabled*, no Zero search is done at any time unless written in a program.

If one of the 2 modes *Power up: Sensor type 1* or *Sensor type 2* is selected, the respective Zero search mode will be executed every time the MIS motor is powered up if no program is started up. If a program has been made and is running, the Zero search command must be executed within the program to execute a Zero search.

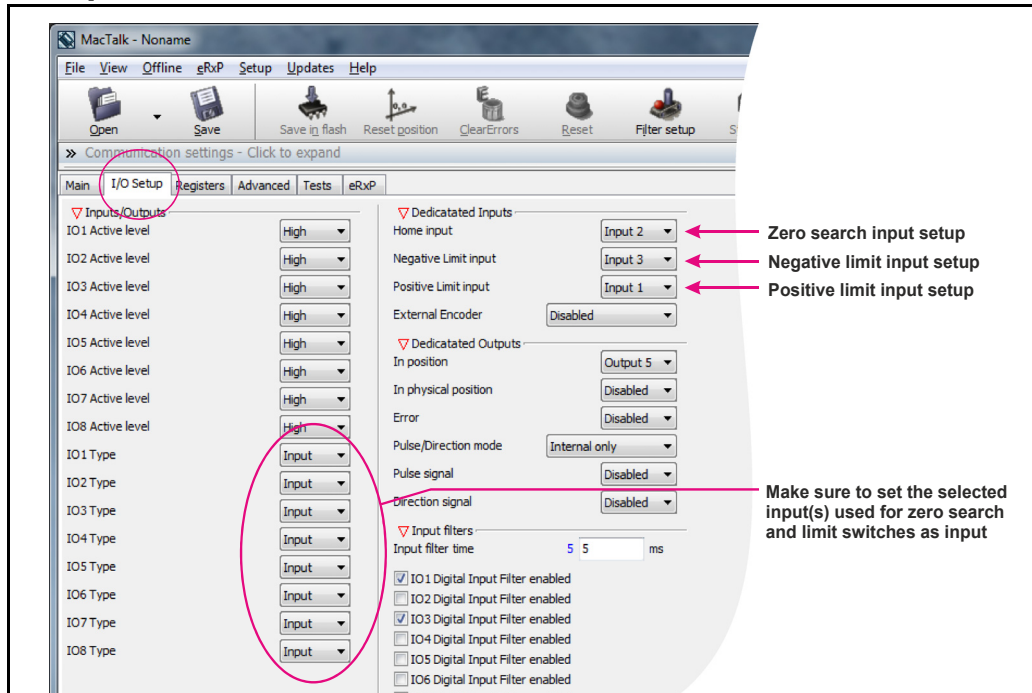
The MIS motor's zero search facility is very flexible. The inputs for reference and limit switches must be set up correctly before use.

The active levels must also be set up correctly.

## 6.5

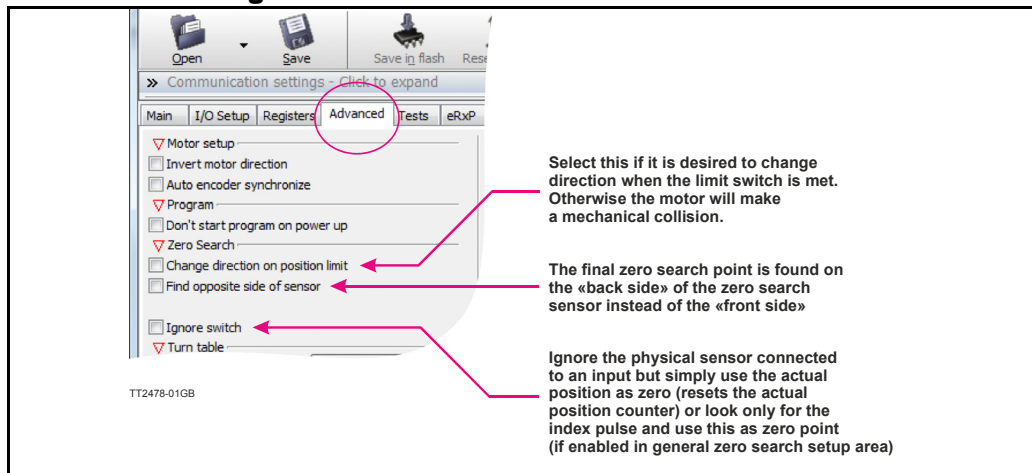
# Zero search modes

### 6.5.3 Set up the I/O's for zero search



**Important information:** Each of the 8 pins can be defined as inputs or outputs. The active digital input level for each input is also defined in the above screen. Furthermore, it is possible to set up a filter for each input to avoid noise interfering with the program. The inputs for Home, Negative Limit and Positive Limit are selected here.

### 6.5.4 Advanced settings



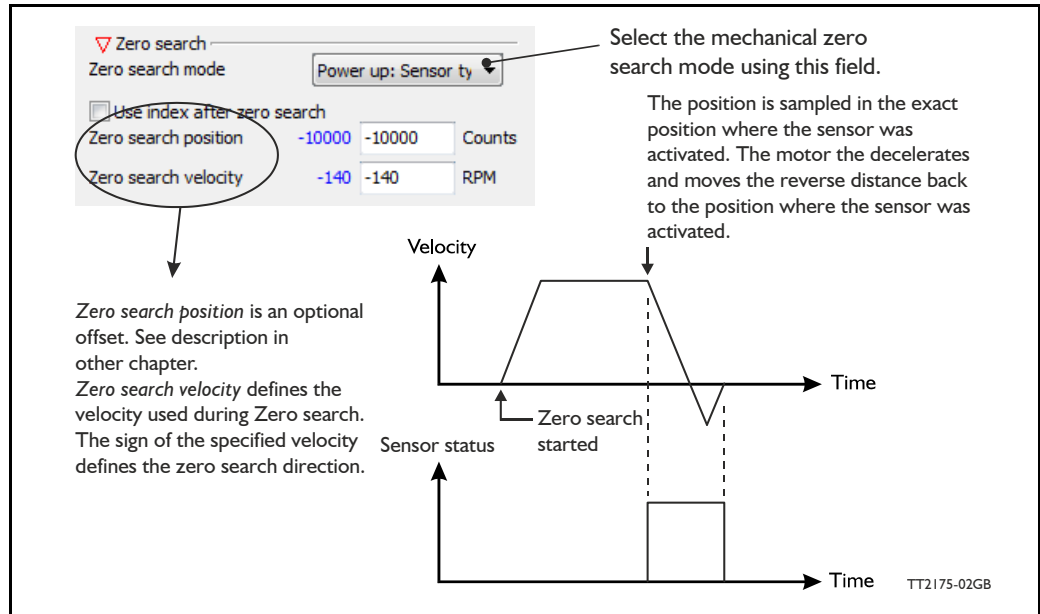
There are several ways to perform a Zero search:

- Start from both sides of the reference sensor in a system with limit switches without having position limit problems.
- to go to the opposite side of the sensor and use this position as zero position.
- use a position limit as reference position. In this case the zero search position must be different from 0 or the motor enters passive mode.
- ignore the reference switch input and use the actual position or index pulse as zero position before using the zero search position.

## 6.5 Zero search modes

### 6.5.5 "Sensor type 1" Zero search

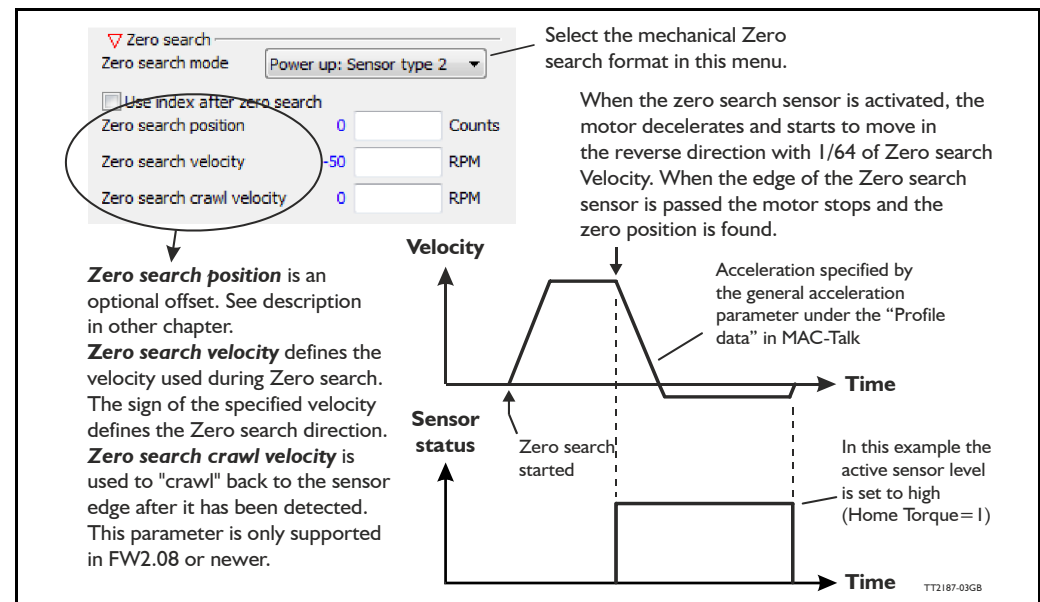
Sensor type 1 zero search is carried out according to the following illustration:



The Zero sensor must be connected to a user input  
For connection information, see [User Inputs](#), page 18

### 6.5.6 "Sensor type 2" Zero search

Sensor type 2 zero search is carried out according to the following illustration.



The Zero sensor must be connected to a user input. For connection information, see [User Inputs](#), page 18.

**Hint:** Make sure the acceleration/deceleration is set to an appropriate value which stops the motor when the Zero search switch is detected but before mechanical collision.

## 6.5

# Zero search modes

### 6.5.7 Making a Zero point offset

Common for all the zero search modes, it is possible to optionally define the zero point as a value other than zero (position 0).

When is it useful to use the zero point offset?

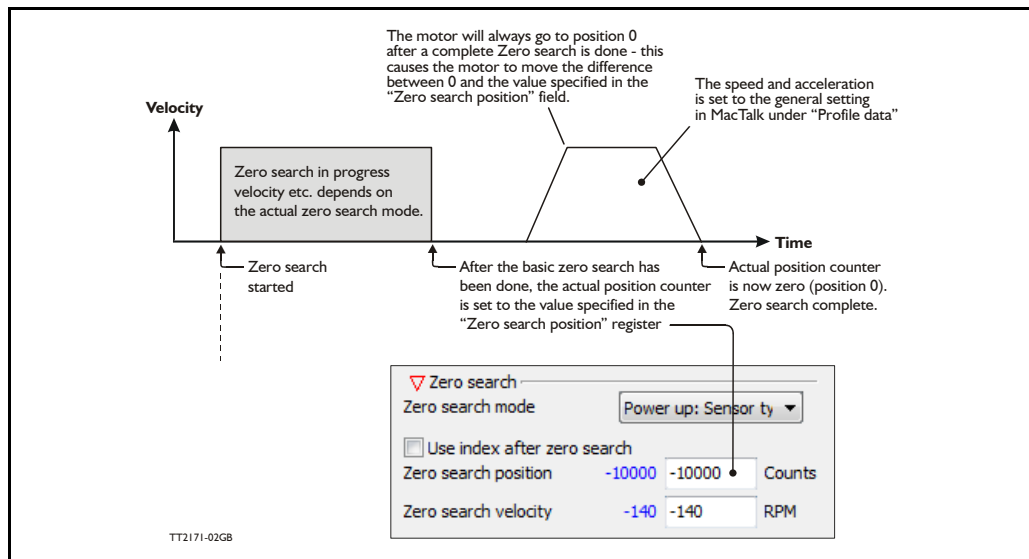
- If it is required that the position interval under normal operation is always convenient values from 0 to x instead of a mixture of negative and positive values. This can happen if the zero point sensor is placed a long distance away from the normal positioning interval or inside the normal positioning interval.
- If an automatic move to an initial position is desired after a power up zero search.

The offset value must be specified in the “Zero search position” field.

The complete zero search will be performed in the following order:

1. The zero search is started either automatically (power up) or initiated by a command from the serial interface.
2. The basic Zero search is completed and the position counter is set to the value specified in the “Zero search position” field.
3. If the zero search position value is different from position, the motor will now move to position 0.
4. The zero search is now complete and the motor will switch to normal operation, i.e. the mode selected in the “Start up mode” field in the main window.

The illustration below shows the complete zero search cycle.



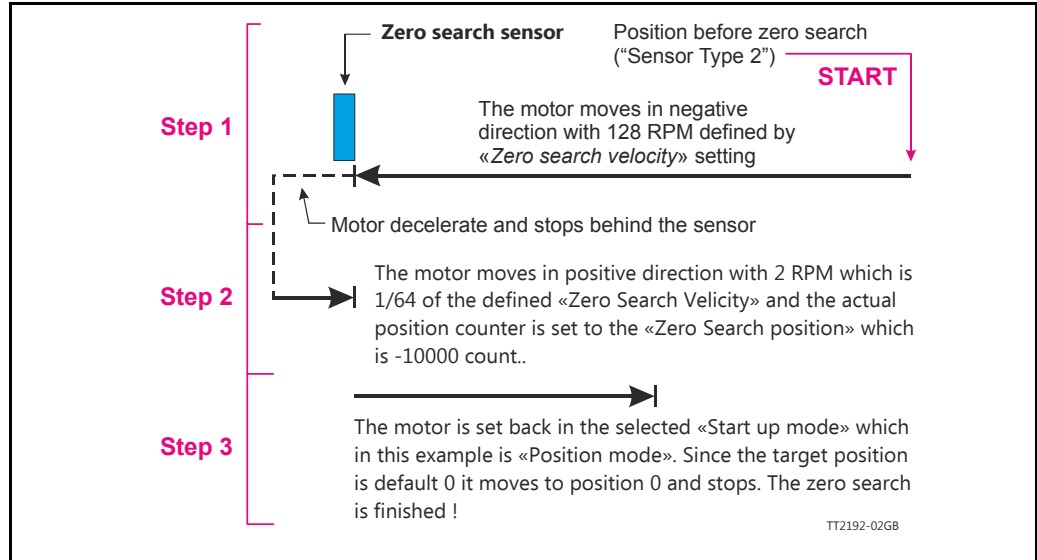
## 6.5

# Zero search modes

### Zero point offset Example.

Setup done before start:

- Zero search velocity = -128 rpm
- Zero search position = -10000 counts



### 6.5.8 Setting up zero search without MacTalk

If MacTalk is not used for setting up parameters/registers related to the zero search feature it must be done as follows.

The motor contains a number of registers which can be accessed from various protocols depending at which options the motor has.

Protocols available are for example Ethernet (EthernetIP, Profibus etc.) and CAN-open, Modbus or the MacTalk protocol.

Each field in MacTalk described earlier in this chapter is accessing a register in the motor.

The registers that are relevant for zero search operation are:

#### Zero search basic settings:

**R38 P\_HOME**      **MacTalk name: “Zero search position”**  
The found zero point is offset with this value.  
See also [P\\_Home](#), page 164

**R40 V\_HOME**      **MacTalk name: “Zero search velocity”**  
The velocity to use during zero search.  
Set a negative velocity to search in the negative direction.  
See also [V\\_Home](#), page 164

**R42 HOMEMODE**      **MacTalk name: “Zero search mode”**  
Selects the zero search type that should start on power up.  
See also [Home mode](#), page 164

## 6.5

## Zero search modes

---

- R120 INDEX\_OFFSET** **MacTalk name: N/A**  
The position of the zero sensor relative to the encoder index. This is set after a zero search where the index is used.  
See also [Index\\_Offset](#), page 172
- R242 V\_HOME\_CRAWL** **MacTalk name: “Zero search crawl velocity”**  
In Zero Search type 2, the “crawl” velocity is V\_HOME/64 by default.  
If register 242:V\_HOME\_CRAWL is <>0, a user defined velocity is used – independent of V\_HOME. Please note that overshoot can occur if this velocity is set too high.  
See also [V\\_HOME\\_CRAWL](#), page 193
- R243 V\_HOME\_TIMEOUT** **MacTalk name: N/A**  
The default zero search time out is 60 s. This register sets another time out in milliseconds. If the time out is passed the motor will return to startup mode.  
See also [V\\_HOME\\_TIMEOUT](#), page 193

### Zero search advanced settings:

- R122 Zero\_Search\_BITS** **MacTalk names:** (multiple - see below)  
“Use index after zero search”, bit 0  
“Change direction on position limit”, bit 1  
“Find opposite side of sensor”, bit 2  
“Ignore switch”, bit 4  
“Disable zero search timeout”, bit 5  
Explanation of the individual bits see [Advanced settings](#), page 126. See also [Zero\\_Search\\_Bits](#), page 173

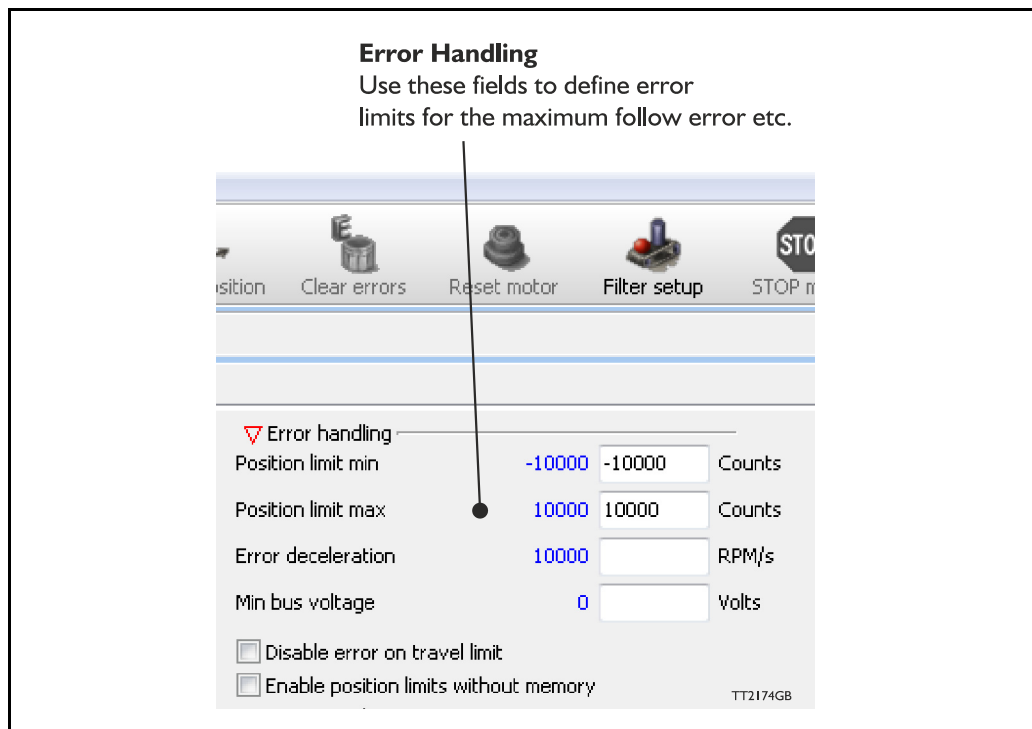
### Zero search I/O setup:

- R125 IOSETUP** **MacTalk names: “I/O Setup” tab**  
Bit 0-7 Sets the I/O active level.  
Bit 8-15 Enables the I/O as an output.  
See also [IOsetup](#), page 174
- R132 HOME\_MASK** **MacTalk name: “Home input”**  
Input mask for home sensor input(s), each bit select which of the I/O 1-8 to use as input for the home sensor signal.  
See also [Home\\_Mask](#), page 176
- R135 INPUT\_FILTER\_MASK** **MacTalk names: “Input filters”**  
Input mask for the digital inputs with input filter.  
Bits set use the input filter time in register 136, bits cleared use a fixed update time of 100 us.  
See also [Input\\_Filter\\_Mask](#), page 177
- R136 INPUT\_FILTER\_CNT** **MacTalk name: “Input filter time”**  
The number of milliseconds the filtered digital inputs must be stable before accepting a change.  
See also [Input\\_Filter\\_Cnt](#), page 177



## 7.1

# Setup error limits



The MIS motor contains 5 fundamental parameters which are used for protection related purposes. They all have effect regardless of which mode of operation the motor is set to use.

### Position limit min. and max.

Same as physical limit switches but implemented in software. Default is 0 meaning that the feature is disabled. If one parameter is different from 0, both values are activated. See also [End-of Travel Limit Inputs](#), page 102

### Error acceleration

If a fatal error occurs, it can be convenient to use a controlled deceleration instead of a sudden stop. If the inertia in the system is high and the mechanical parts are weak, a sudden stop can cause damage and unintended behaviour. Use this parameter to define the deceleration used during a fatal error. Default is 0, meaning that the feature is disabled.

### Min. bus voltage

This is the level of P+ at which the motor goes into error state “low bus voltage”.

### Disable error on travel limit

If one of the position limits (external sensors) are activated no error is generated. This will avoid that the motor enter passive mode and make the motor power less. See also [End-of Travel Limit Inputs](#), page 102

### Enable position limits without memory

See [Simple mode: Position limits without memory](#), page 106



## 7.2

# Error messages

### 7.2.1 How to monitor Errors

Any error that occur will show up in the right side of the MacTalk screen.

Some errors can be cleared by pressing the “Clear Error” button.

Other errors are fatal and need other actions.

For example will a temperature error not be possible to clear before the temperature is lower than what is accepted.

The screenshot shows the MacTalk software interface. At the top, there are several icons: 'Clear errors' (circled in red), 'Reset motor', 'Filter setup', 'STOP motor', and 'AutoScan'. A text box above the 'Clear errors' icon says 'Errors can be cleared here if they are not fatal'. Below the icons are dropdown menus for 'Baud: 19,200' and 'Motor Address: All'. The main area is divided into two columns. The left column contains 'Error handling' settings with various input fields and checkboxes. The right column is a 'Status' panel with sections for 'Motor status', 'Run status', 'Inputs', 'Outputs', 'External Encoders', and 'Errors'. The 'Errors' section is circled in red and lists: 'General', 'Follow Error', 'Output driver', and 'Position limit'. A text box below the status panel says 'Error messages are shown here if any'. The top right of the window shows 'MacTalk Version: 1.70.027'.

The next pages describe every error including cause for the error and how to solve the error situation.

# 7.2

# Error messages

## 7.2.2 Reading the Event log

In MacTalk it is possible to monitor all the saved data. This tab shows the total amount each error has occurred, the last 20 errors with time stamps, total amount of revolutions the motor has run during the entire lifetime etc.

The motor do not have a real time clock so all time stamps are based on the active time where the motor has been powered also showed as “Up time”.

The screenshot shows the MacTalk software interface. The main window is titled "MacTalk - Noname" and includes a menu bar (Files, Motor, eRXP, Setup, Updates, Help) and a toolbar with icons for Open, Save, Save in motor, Reset position, Clear errors, Reset motor, Filter setup, STOP motor, and AutoScan. The status bar at the top right indicates "MacTalk Version: 1.70.027".

The interface is divided into several sections:

- Serial port:** Comport: 1, Baud: 19,200, Motor Address: All.
- Event Log:** A table showing error types and their occurrences. The "Follow error" is highlighted.
- Up time:** A large digital display showing "28h:47m:1s".
- Status:** A panel on the right showing motor status, run status, and various parameters like velocity, position, and temperature.
- Inputs/Outputs:** A section with indicator lights for inputs and outputs.
- Warnings:** A section at the bottom right showing a warning for "MIS340 on COM1" with an image of the motor.

Error type	Number of errors	Last error time
Follow error	9	27h:49m:11s
Output driver	2	23h:37m:21s
Position limits	2	23h:37m:21s
Low bus voltage	4	23h:37m:21s
Overvoltage	2	23h:37m:21s
Temperature	2	23h:37m:21s
Internal error	4	23h:37m:21s
Encoder lost position	2	23h:37m:21s
Encoder reed sensor	2	23h:37m:21s
Encoder no communication	2	23h:37m:21s
External encoder	2	23h:37m:21s
Closed loop lost sync	2	23h:37m:21s
Saved encoder position	0	0h:0m:0s
Saved P_IJT	0	0h:0m:0s

Saved events	Saved value
Saved encoder position	190900
Saved P_IJT	000390
Powerdown count	52
Total runtime [hours:minutes:seconds]	27h:39m:27s
PLC Flash savings	2
Userdata flash savings	2
Saved SSI data	0
Saved Ext. Encoder data	0
Total amount of rotations [1000x rev]	36101

Error type	Last error time
Follow error	27h:49m:11s
Follow error	27h:11m:37s
Closed loop lost sync	23h:37m:21s
External encoder	23h:37m:21s
Encoder no communication	23h:37m:21s
Encoder reed sensor	23h:37m:21s
Encoder lost position	23h:37m:21s
Internal error	23h:37m:21s
Temperature	23h:37m:21s
Output driver	23h:37m:21s
Internal error	23h:37m:21s
External error	23h:37m:21s

Motor status:  
Active mode: Gear  
Actual velocity: 0.00 RPM  
Actual position: 888398 Counts  
Encoder position: 888398 Counts  
Abs. Encoder position: 190106 Counts  
Follow error: 55 Counts

Run status:  
In phys position  
In Position  
At velocity  
Accelerating  
Decelerating  
Zero search done

Bus voltage (P+): 24 Volts  
Temperature: 32 °C

Inputs: 8 7 6 5 4 3 2 1 (0.02 Volts)  
Outputs: 8 7 6 5 4 3 2 1 (0.02 Volts)

External Encoders: 0 Counts  
External Encoder Velocity: 0 Counts/s

Warnings:  
MIS340 on COM1

MIS340 (Version V4.00.00070, SN: 1) Connected  
TT2394-01GB

## 7.2

## Error messages

### 7.2.3 Error messages and error handling

The following list show the possible error messages, the cause of the error and possible actions to prevent the error from happening. Each error can also be monitored by reading the Error register (register 35) by using software packages like the OCX driver or MacRegIO.

### 7.2.4 Error message 'Follow error'

Message no. / Message	1 / 'Follow error'
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	The Follow error (register 20) has exceeded the value specified in "Follow error max" (register 22).
Possible cause of this error	"Follow error max" is set to a too low value and therefore the Follow error exceeds this value at normal operation. The motor has stalled because of too much load or a too low "Running current" (register 7). Faulty encoder
Solutions to avoid error	Set "Follow error max" to a "much" greater value than the average "Follow error" while running at the desired velocity (V_SOLL, register 5). Please allow some overhead in order to avoid Errors because of small spikes in the "Follow error". Please notice that 1 revolution is 409600 counts.  Make sure that the "Running current" (register 7) is set high enough to avoid step loss or stalling.  In Passive mode, the shaft can be moved freely to check that the encoder is counting properly – 1 revolution should make 409600 counts. Looking at the front of the motor, the positive counting direction is clockwise.
How to return to normal operation	Clear the error bits in register 35. Return to the desired active mode. or Reset the motor / Cycle the power
Error bit / Firmware name	Bit 1

### 7.2.5 Error message 'Output driver'

Message no. / Message	2 / 'Output driver'
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	1 or more of the 8 IO's has a hardware fault.
Possible cause of this error	An IO has been setup as an output, but 24 V is applied to that specific output.  Output pins are overloaded.
Solutions to avoid error	Make sure that the setup is correct. The easiest way to do this is by connecting to MacTalk and go to the I/O setup tab. Here the actual setting for each pin is shown. The status on the Inputs and Outputs can be monitored in the right "Status" panel.  Always take care not to load the outputs by more than 300 mA per channel. Please consult the <a href="#">User Outputs</a> , page 28 section.  Check that none of the wires are short-circuited.
How to return to normal operation	Reset the motor / Cycle the power
Error bit / Firmware name	Bit 2

## 7.2

# Error messages

### 7.2.6 Error message 'Position limit'

Message no. / Message	3 / 'Position limit'
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	This error relates to both the Hardware and software position limits. It is set when one of the 4 conditions is true: <ul style="list-style-type: none"><li>• The positive sensor has been active</li><li>• The negative sensor has been active</li><li>• Actual Position is greater or equal to Max position (register 30)</li><li>• Actual Position is less or equal to Min position (register 28)</li></ul>
Possible cause of this error	<ul style="list-style-type: none"><li>• One of the position limits is reached.</li><li>• Noise on the hardware limit input.</li><li>• Faulty limit sensor.</li></ul>
Solutions to avoid error	<ul style="list-style-type: none"><li>• When position limits are activated, make sure that the motor does not reach the limits. An internal absolute multi turn encoder can be a good help to avoid reaching the limits.</li><li>• If the motor must reach the limits, but also stay in the active mode, it is possible to disable the 'Position limit' error message by setting Setupbit 17 in register 124. Please see <a href="#">Position Limits</a>, page 102</li><li>• If noise has triggered the hardware position limits, it is recommended to add some digital filtering on the inputs. The easiest way to do this is by connecting the motor to MacTalk and setup the filter for the specific input on the I/O setup tab. See also <a href="#">Digital input filter setup with MacTalk</a>, page 20</li></ul>
How to return to normal operation	<ul style="list-style-type: none"><li>• Clear the error bits in register 35.</li><li>• Return to the desired active mode. It is now only possible to run the motor in the opposite direction of the limit.</li><li>• Reset the motor / Cycle the power</li></ul>
Error bit / Firmware name	Bit 3

### 7.2.7 Error message 'Low bus voltage'

Message no. / Message	4 / 'Low bus voltage'
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	<ul style="list-style-type: none"><li>• The measured bus voltage is lower than the level Min_Busvol (register 98).</li><li>• The voltage of the P+ bus voltage has been measured to be lower than the limit selected in the register 'Min_Busvol' (register 98). This has resulted in an error as configured in the setup of 'Undervoltage handling'. See also <a href="#">Under voltage Handling</a>, page 107.</li></ul>
Possible cause of this error	<ul style="list-style-type: none"><li>• The current rating of the external power supply is too small.</li><li>• The power supply is not able to deliver the required peak currents that the motor need. This is a typical problem when using switch mode power supply.</li><li>• The power cable is under dimensioned.</li><li>• The under voltage min. setting must be decreased.</li></ul>
Solutions to avoid error	<ul style="list-style-type: none"><li>• Use a power supply with a higher current rating.</li><li>• Use a power cable with at least 0,75mm<sup>2</sup> wires (up to cable lengths of 10m. If the power cable is longer, use 1,5mm<sup>2</sup> or use multiple wires in parallel.</li><li>• Connect a capacitor across the supply line close to the motor. Especially if using a switch mode power supply this will help.</li></ul>
How to return to normal operation	<ul style="list-style-type: none"><li>• Reset the motor, clear the error bit(s) in register 35 or cycle the power.</li></ul>
Error bit / Firmware name	Bit 4

## 7.2

## Error messages

### 7.2.8 Error message 'Over voltage'

Message no. / Message	5 / 'Over voltage'
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	A too high P+ voltage (> 100 V) has been measured.
Possible cause of this error	<ul style="list-style-type: none"><li>• A too high P+ voltage has been connected.</li><li>• The returned amount of energy from the motor has been too high. This can typically happen if:</li><li>• The motor decelerate a large load inertia too fast. - The motor is turned by an external force.</li></ul>
Solutions to avoid error	<ul style="list-style-type: none"><li>• Decrease the load inertia.</li><li>• Decrease the top speed and/or the acceleration value.</li><li>• Make sure that the supply voltage is within the limits.</li></ul>
How to return to normal operation	<ul style="list-style-type: none"><li>• Reset the motor, clear the error bit(s) in register 35 or cycle the power.</li></ul>
Error bit / Firmware name	Bit 5

### 7.2.9 Error message 'Temperature'

Message no. / Message	6 / 'Temperature'
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	The temperature has been higher than 90°C (194F) which is not allowed.
Possible cause of this error	<ul style="list-style-type: none"><li>• The ambient temperature is higher than allowed - max is +40°C/ 104°F.</li><li>• The motor is build into an environment where it can not dissipate enough heat.</li><li>• The motor is not mounted on a proper mechanical structure where heat can be dissipated.</li></ul>
Solutions to avoid error	<ul style="list-style-type: none"><li>• Make precautions to decrease the surrounding ambient temperature.</li><li>• Lower the speed and or load on the motor.</li></ul>
How to return to normal operation	<ul style="list-style-type: none"><li>• Reset the motor, clear the error bit(s) in register 35 or cycle the power.</li></ul>
Error bit / Firmware name	Bit 6

### 7.2.10 Error message 'Internal error'

Message no. / Message	7 / 'Internal error'
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	The firmware consists of 2 parts, and only one part is working.
Possible cause of this error	<ul style="list-style-type: none"><li>• Firmware update process has been interrupted.</li></ul>
Solutions to avoid error	<ul style="list-style-type: none"><li>• Use the recommended USB-RS485 converter with part number RS485-USB-ATC-820. See <a href="#">MISxxxxxQ5xxx connector description.</a>, page 35</li><li>• Let MacTalk finish the firmware update.</li></ul>
How to return to normal operation	<ul style="list-style-type: none"><li>• Try firmware updating again and follow the recommendations above.</li></ul>
Error bit / Firmware name	Bit 7

## 7.2

## Error messages

### 7.2.11 Error message 'Encoder lost position'

<b>Message no. / Message</b>	<b>8 / 'Encoder lost position'</b>
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	The absolute multi turn encoder (H3/H4) has lost the position.
Possible cause of this error	<ul style="list-style-type: none"><li>• The reset of the encoder can be caused by a firmware update.</li><li>• The battery level is low and the encoder cannot remember the position.</li></ul>
Solutions to avoid error	Hints to optimise the battery lifetime: <ol style="list-style-type: none"><li>1. Avoid to place the motor in an environment with high temperatures.</li><li>2. Set the running and especially the standby motor current as low as possible in order not to heat up the motor unnecessarily.</li><li>3. Keep the external power applied as much as possible.</li></ol> See also <a href="#">Position retention time</a> , page 101 for further info
How to return to normal operation	<ul style="list-style-type: none"><li>• Reset the position (special command 354 in register 24), clear the error bit(s) in register 35 or cycle the power.</li><li>• If the battery level is low, this error will re-appear every time after power has been off for a while. In that case, the motor must be returned for service.</li></ul>
Error bit / Firmware name	Bit 8

### 7.2.12 Error message 'Encoder Reed error'

<b>Message no. / Message</b>	<b>9 / 'Encoder Reed error'</b>
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	<ul style="list-style-type: none"><li>• The absolute multi turn encoder (H3/H4) has detected a wrong sequence in the positioning algorithm.</li><li>• This error also occurs after firmware update.</li></ul>
Possible cause of this error	<ul style="list-style-type: none"><li>• This can be caused by a mechanical shock on the shaft or an external magnetic field.</li><li>• Because the encoder has been reset during a firmware update.</li></ul>
Solutions to avoid error	<ul style="list-style-type: none"><li>• Do not place the motor inside a strong magnetic field.</li><li>• Do not expose the shaft or the motor for mechanical shocks.</li></ul>
How to return to normal operation	<ul style="list-style-type: none"><li>• Reset the position (special command 354 in register 24), clear the error bit(s) in register 35 or cycle the power.</li></ul>
Error bit / Firmware name	Bit 9

## 7.2

## Error messages

### 7.2.13 Error message 'Encoder COM error'

Message no. / Message	10 / 'Encoder COM error'
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	The internal communication to the Absolute multi turn encoder (H3/H4) does not work.
Possible cause of this error	<ul style="list-style-type: none"><li>• A firmware update of either the SMC66/85 or the Absolute multi turn encoder (H3/H4) has gone wrong.</li><li>• Hardware error.</li></ul>
Solutions to avoid error	<ul style="list-style-type: none"><li>• This error should not occur during normal operation, but can happen if something went wrong in a firmware update process.</li><li>• If a new firmware update does not clear the error: Return the motor for service.</li></ul>
How to return to normal operation	<ul style="list-style-type: none"><li>• This error will only be set during startup and can be cleared afterwards. Then the motor operation will be the same, but the multi turn encoder will not work.</li><li>• Clear the error bit(s) in register 35.</li></ul>
Error bit / Firmware name	Bit 10

### 7.2.14 Error message 'External encoder'

Message no. / Message	11 / 'External encoder'
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	An external SSI encoder has been enabled but communication with the encoder has failed.
Possible cause of this error	<ul style="list-style-type: none"><li>• The encoder is not connected correctly.</li><li>• The format chosen is not compatible with the actual encoder.</li><li>• Improper cabling have been used.</li></ul>
Solutions to avoid error	<ul style="list-style-type: none"><li>• Use proper cabling between the motor and the external SSI encoder. A screened cable with twisted pair wires is recommended.</li><li>• Make sure that the right SSI format is selected.</li></ul>
How to return to normal operation	<ul style="list-style-type: none"><li>• Reset the motor, clear the error bit(s) in register 35 or cycle the power.</li></ul>
Error bit / Firmware name	Bit 11

### 7.2.15 Error message 'Closed loop'.

Message no. / Message	12 / 'Closed loop'
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	-
Possible cause of this error	-
Solutions to avoid error	-
How to return to normal operation	-
Error bit / Firmware name	Bit 12

## 7.2

## Error messages

### 7.2.16 Error message 'External memory'

<b>Message no. / Message</b>	<b>13 / 'External memory'</b>
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	The controller has build in memory for the "Event log".
Possible cause of this error	The memory self test has failed because of faulty hardware inside the motor.
Solutions to avoid error	Return the motor for service.
How to return to normal operation	<ul style="list-style-type: none"><li>• This error will only be set during startup and can be cleared afterwards. Then the motor operation will be the same, but the "Event log" will not work.</li><li>• Clear the error bit(s) in register 35.</li></ul>
Error bit / Firmware name	Bit 13

### 7.2.17 Error message 'Single turn encoder error'

<b>Message no. / Message</b>	<b>14 / 'Singleturn encoder error'</b>
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	The absolute single turn encoder (H2/H4) has failed.
Possible cause of this error	<ul style="list-style-type: none"><li>• Wrong setup.</li><li>• The distance between the internal magnet and the internal encoder sensor is outside the limits caused by too high force at the motor shaft in forward or backward direction.</li></ul>
Solutions to avoid error	<ul style="list-style-type: none"><li>• Take care when changing settings in Internal_Encoder_Setup (register 175).</li><li>• Do not expose the shaft or the motor for mechanical shocks.</li></ul>
How to return to normal operation	<ul style="list-style-type: none"><li>• If the error appeared after changing some settings, please try to "Load factory defaults" in MacTalk.</li><li>• Cycle the power to the motor.</li></ul>
Error bit / Firmware name	Bit 14

### 7.2.18 Error message 'Safe Torque Off'

<b>Message no. / Message</b>	<b>15 / 'Safe Torque Off'</b>
Type / Motor action	Unrecoverable error / Motor is set in Passive mode.
Error condition	"Safe Torque Off" has been triggered.
Possible cause of this error	<ul style="list-style-type: none"><li>• One of the two inputs STO channel A or STO channel B has been measured less than 18 V.</li><li>• A hardware fault in the STO circuitry.</li></ul>
Solutions to avoid error	Make sure that both STO channels are connected to 24 V with a stable power supply.
How to return to normal operation	<ul style="list-style-type: none"><li>• Apply a stable 24 V to both STO channels and clear the error bit(s) in register 35.</li><li>• If the Safe Torque Off error is still present, the STO circuit has a hardware fault and the motor must be sent to JVL for inspection.</li></ul>
Error bit / Firmware name	Bit 27





## 8.1 Introduction to registers

---

All of the motor registers can be accessed either through the RS485 interface which is the standard interface in the MIS motors.

Optionally the registers can also be accessed through the optional CANopen, or Ethernet interface. A separate manual LB0056 exist for the industrial ethernet protocols.

The Ethernet manual can be found at [www.jvl.dk](http://www.jvl.dk) using this link : [www.jvl.dk](http://www.jvl.dk)

When accessing registers over CANopen, they are mapped to object indexes 2012 and 2014 (hex) with the sub-index equal to the register number 1...255. Use index 2012 for the 32-bit registers and index 2014 for the 16-bit registers.

For example to access all 32 bits of P\_SOLL, use index 2012, subindex 3. To access 16 bits of V\_SOLL use index 2014, subindex 5. This is described in more detail in [CANopen \(optional\)](#), page 229.

All of the registers can be accessed over CANopen with the same Read/Write access restrictions as when using the RS485 interface.

Some registers are tagged as R for Read-only. There are different reasons for this, such as protecting the serial number from being changed or indicating that the value in registers, such as analogue Inputs, will never be read by the motor but always overwritten using the latest sampled values.

## 8.2

# Internal registers

### 8.2.1 Register Overview

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
1	PROG_VERSION	32bit	R	-	-	Major*16 + Minor + 16384 + 17*2 <sup>14</sup>	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 <sup>31</sup> )-(2 <sup>31</sup> -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 maximum 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 <sup>31</sup> )-(2 <sup>31</sup> -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 <sup>31</sup> )-(2 <sup>31</sup> -1)	409600	Counts	The multiplier of the gear factor	Output
15	GEAR2	32bit	R/W	(-2 <sup>31</sup> )-(2 <sup>31</sup> -1)	2048	Counts	The divider of the gear factor	Input
16	ENCODER_POS	32bit	R/W	(-2 <sup>31</sup> )-(2 <sup>31</sup> -1)	-	Steps	If the encoder option is installed, this shows 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 <sup>31</sup> )-(2 <sup>31</sup> -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)	

## 8.2

## Internal registers

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

## 8.2

## Internal registers

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)

## 8.2

## Internal registers

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 <sup>th</sup> 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

## 8.2

## Internal registers

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

## 8.2

## Internal registers

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



## 8.2

## Internal registers

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	0-65535	-		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	AVAILABLE_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	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)	

## 8.2

## Internal registers

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								
<b>NOTE: DSP-402 is NOT supported yet!</b>								
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								

## 8.2

## Internal registers

Reg	Name	Size	Access	Range	Default	Unit	Description	MacTalk name
202	TICKS	32bit	R/W	0-(2 <sup>32</sup> -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 instead.	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

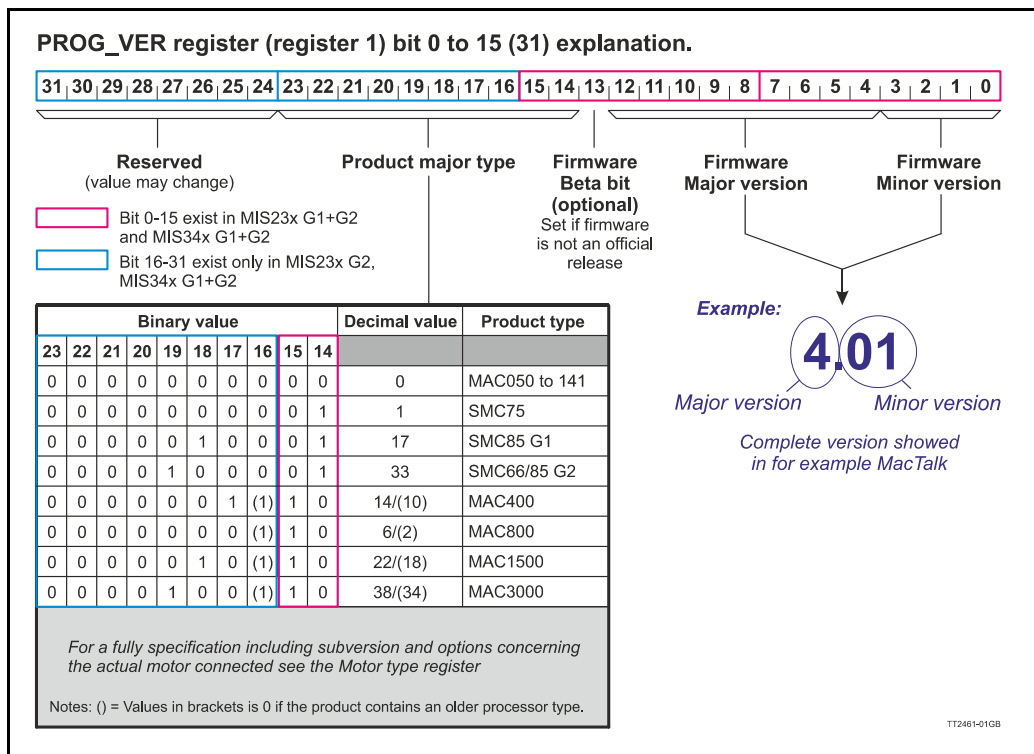
## 8.2 Internal registers

### 8.2.2 Prog\_Vers

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
1	PROG_VERSION	32bit	R	-	*	-	"Status bar"

**Description:** The firmware version. The Bit 14 is set to indicate that the type is SMC75 or SMC85. Bit 0-3 is the minor version and bit 4-12 is the major version. Bit 13 is set if the actual firmware is a beta version (not officially released). Bit 14 to 23 indicate the overall motor type. For specific motor type see also the register [Motor type](#), page 181

Detailed description of the individual bits:



## 8.2

## Internal registers

---

### 8.2.3 Mode\_Reg

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
2	Mode_Reg	32bit	R/W	0,1,2,3,11, 13,14,15	0	-	Current Mode

Description: Controls the operating mode of the motor. The following modes can be selected:

- 0: Passive
- 1: Velocity mode
- 2: Position mode
- 3: Gear mode
- 11: Stop mode
- 13: Zero search type 1
- 14: Zero search type 2
- 15: Safe mode

#### **Passive mode (0)**

In this mode, the motor current is turned off and the motor will not react to any position/velocity commands.

#### **Velocity mode (1)**

When the motor is in velocity mode, the controller accelerates the motor to the velocity in V\_SOLL. V\_SOLL can be changed at any time and the move will decelerate/accelerate accordingly.

It is permissible to change A\_SOLL and V\_START during a movement, but the changes will first take effect after the motor has stopped. Please note that if the motor needs to change direction, it will decelerate and stop, and the new A\_SOLL and V\_START will be activated.

#### **Position mode (2)**

When the motor is in position mode, the controller will always try to move until P\_IST = P\_SOLL.

The movement will follow the profile specified by V\_SOLL, A\_SOLL and V\_START.

P\_SOLL can be changed at any time and the motor will move accordingly.

V\_SOLL can also be changed during a movement.

It is permissible to change A\_SOLL and V\_START during a movement, but the changes will first take effect after the motor has stopped. Please note that if the motor needs to change direction, it will decelerate and stop, and the new A\_SOLL and V\_START will be active.

## 8.2

# Internal registers

### Gear mode (3)

The GEAR mode works as position mode, but has an additional feature. The input on the external encoder is multiplied with GEAR1/GEAR2 and added to P\_SOLL. Any remainder of the result is saved and used next time the external encoder changes.

The result is that this mode can be used as an electronic gear.

When using gear mode, it is not recommend to set V\_START below 10 rpm. This can gives problems at low speeds, because the motor will lag behind when doing the first step. It will then accelerate in order to catch up.

**NOTE:** Time from the first input pulse to the first step is typically 30-60 $\mu$ s if not on standby. 72-102 $\mu$ s if on standby.

### Stop mode (11)

When changing from an active mode (Velocity, Position, Gear) to passive mode the motor decelerates with A\_SOLL (or D\_SOLL if not = 0) before it goes passive.

### Zero search type 1 (13)

When the operation mode is set to 13, the controller will start the search for the zero point. See “[Sensor type 1](#)” *Zero search*, page 127 for details.

### Zero search type 2 (14)

When the operation mode is set to 15, the controller will start the search for the zero point. See “[Sensor type 2](#)” *Zero search*, page 127 for details.

### Safe mode (15)

This mode is similar to passive mode, but also allows the “save in flash” and “reset” commands. Safe mode cannot be entered/exited directly; this must be done using the serial commands ENTER/EXIT SAFEMODE.

#### Example:

Writing MODE\_REG=2 will set the motor in position mode. When P\_SOLL is changed, the motor will move to this position with the specified max velocity (V\_SOLL) and acceleration (A\_SOLL).

Writing MODE\_REG=13 will start a zero search for a sensor. When the search is completed, the MODE\_REG will automatically be changed to the mode specified in START\_MODE.

### 8.2.4 P\_SOLL

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
3	P_SOLL	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	Counts	Position

**Description:** The desired position. When in position mode, the motor will move to this position. This value can be changed at any time. The maximum possible position difference is  $2^{31}-1$ . If relative movement is used, the P\_SOLL will just wrap at  $2^{31}-1$  and the motor will move correctly. Please note that the turntable function changes the behaviour of P\_SOLL. See [Turntable\\_Mode](#), page 175. **Notice that the turntable feature is not supported in the present firmware version.**

The MISxxx motor family all have 409600 counts per motor revolution.

**Example:** If P\_SOLL = 0 and then P\_SOLL is set to 409600, the motor moves one revolution forward.

## 8.2 Internal registers

### 8.2.5 V\_SOLL

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
5	V_SOLL	32bit	R/W	±1-300000 (0.01-3000RPM)	10000 (100 RPM)	RPM/100	Max velocity

**Description:** The maximum velocity allowed. 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:** V\_SOLL = 25000, will limit the velocity to 250 RPM.

### 8.2.6 A\_SOLL

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
6	A_SOLL	32bit	R/W	1-500000	1000	RPM/s	Acceleration

**Description:** 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.

**Example:** A\_SOLL = 100, will set the acceleration to 100 RPM/s.

### 8.2.7 Run\_Current

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
7	RUN_CURRENT	32bit	R/W	0-1533	511	5.87mA	Running Current

**Description:** This register sets the running current for the motor. The software is made for controlling motors up to 9 ARMS per motor phase but the maximum allowed current setting is different from motor to motor size.

Motor type	Max. current	Max. Run_Current setting
MIS17x	4 ARMS	4 ARMS / 5.87 mA = <b>681</b>
MIS23x	6 ARMS	6 ARMS / 5.87 mA = <b>1022</b>
MIS34x	9 ARMS	9 ARMS / 5.87 mA = <b>1533</b>
MIS43x	9 ARMS	9 ARMS / 5.87 mA = <b>1533</b>
MIL34x	6 ARMS	6 ARMS / 5.87 mA = <b>1022</b>

The running current is active when the motor is running and after it stops until the specified standby time has elapsed. See [Standby Time](#), page 156.

When a new value is written to the RUN\_CURRENT register, the new motor current will be set instantly.

**Example:** RUN\_CURRENT = 100, will set the running current to 0.587 ARMS.

## 8.2 Internal registers

### 8.2.8 Standby\_Time

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
8	STANDBY_TIME	32bit	R/W	1-65535	500	ms	Standby Time

Description: This register sets the standby time. This time is the time from the last step has been performed until the current changes from running to standby. When a new request for a move is received the current changes from standby to running with no delay.

Example: STANDBY\_TIME = 200, will result in the controller switching to the standby current after 200ms.

### 8.2.9 Standby\_Current

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
9	STANDBY_CURRENT	32bit	R/W	0-1533	128	5.87 mA	Standby Current

Description: The current range is defined similar to the running current. Please see [Run\\_Current](#), page 155. The standby current is active when the motor has stopped and the specified Standby time has elapsed. See [Standby\\_Time](#), page 156. When the STANDBY\_CURRENT is changed, the new standby current be set instantly.

Example: STANDBY\_CURRENT = 50, will set the standby current to 0.285 ARMS.

### 8.2.10 P\_IST

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
10	P_IST	32bit	R/W	$(-2^{31})-(2^{31}-1)$	-	Counts	Actual Position

Description: This register shows the actual position of the motor. This is updated each time the motor makes a step. If P\_IST is changed when in position mode or gear mode, the motor will move until P\_IST = P\_SOLL. When P\_IST reaches  $2^{31}-1$ , it will wrap around to  $-2^{31}$ . Please note that the turntable function changes the behaviour of P\_IST. See [Turntable\\_Mode](#), page 175. **Notice that the turntable feature is not supported in the present firmware version.**

Example: P\_IST = 1000, P\_SOLL = 1000. P\_IST is set to 500. The motor will move 500 steps forward and P\_IST will again be 1000.

### 8.2.11 V\_IST

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
12	V_IST	32bit	R	$\pm 1-300000$ (0.01-3000RPM)	-	RPM/100	Actual Velocity

Description: This register shows the actual velocity of the motor. The velocity is positive when running in a positive direction and negative when running in a negative direction.

Example: If V\_SOLL = 40000 (400 RPM) and a movement of -10000 steps is done, V\_IST will be -40000 (400 RPM) during the move and when the move is complete V\_IST will be 0.



## 8.2 Internal registers

### 8.2.12 V\_START

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
13	V_START	32bit	R/W	$\pm 1-300000$ (0.01-3000RPM)	10000 (100 RPM)	RPM/100	Start Velocity

**Description:** The start velocity. The motor will start the acceleration at this velocity. It will also stop the deceleration at this velocity. If  $|V\_SOLL|$  is lower than  $V\_START$  the motor will not accelerate at all, but start to run at  $V\_SOLL$  instantly. The motor will actually start the movement with an internal  $V\_START = V\_SOLL$ .

If  $V\_START$  is changed during a movement, it will first be active when the motor stops or changes direction. This also means that if  $V\_SOLL$  is changed to a value below  $V\_START$ , while the motor is in motion, the motor will decelerate to  $V\_START$  and run at that velocity.

**Example:**  $V\_START = 10000$  (100 RPM),  $V\_SOLL = 20000$  (200 RPM),  $MODE\_REG = 1$ . The motor will accelerate from 100 RPM to 200 RPM.  
 $V\_SOLL$  is now changed to 5000 (50RPM). The motor will decelerate to 100 RPM and continue at 100 RPM.  
 $V\_SOLL$  is now changed to -50 RPM. The motor will stop and start at -50 RPM.

### 8.2.13 GEAR1

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
14	GEAR1	32bit	R/W	$(-2^{31})-(2^{31}-1)$	409600	Counts	Output

**Description:** When the gear mode is active, the input from the external encoder is multiplied by  $GEAR1$  and divided by  $GEAR2$ .

**Example:**

1.  $GEAR1 = 409600$ ,  $GEAR2 = 2048$ . If 2048 counts are applied to the input, the motor will turn 1 revolution.
2. If one step is applied, the motor will move 200 counts.

### 8.2.14 GEAR2

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
15	GEAR2	32bit	R/W	$(-2^{31})-(2^{31}-1)$	2048	Counts	Input

**Description:** The denominator of the gear factor. See  $GEAR1$  for details.

### 8.2.15 Encoder\_Pos

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
16	ENCODER_POS	32bit	R/W	$(-2^{31})-(2^{31}-1)$	-	Steps	Encoder position

**Description:** If the internal encoder option is installed, this register shows the position feedback from the encoder. This value is initialized to zero at power-up and modified by the firmware when a zero search is performed. The value can be used internally by the AutoCorrection system to retry a movement in position and gear modes.

## 8.2 Internal registers

### 8.2.16 Inputs

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
18	INPUTS	32bit	R	-	-	Special	Inputs

Description: This register shows the status of the digital inputs. Bit 0-7 shows whether IO 1-8 is active or inactive. The active level can be set using IOSETUP. See [IOsetup](#), page 174. Bits 8-15 are not used and will always be 0. The inputs can be filtered or unfiltered. See [Input\\_Filter\\_Mask](#), page 177.

Note that all of the inputs have a digital state and an analogue value at the same time. This register shows their digital state only. Note that the digital inputs can be filtered by setting bits in register 135 ([Input\\_Filter\\_Mask](#), page 177).

Bit	7	6	5	4	3	2	1	0
Input	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1

### 8.2.17 Outputs

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
19	OUTPUTS	32bit	R/W	-	0	Special	Outputs

Description: This register shows the status of the outputs. Bit 0-7 shows whether IO 1-8 is active or inactive. The active level can be set using IOSETUP. See [IOsetup](#), page 174. Please note that the output driver for each output also has to be enabled. This is also done using IOSETUP. The register can be changed in order to change the status of the outputs.

### 8.2.18 Flwerr

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
20	FLWERR	32bit	R	$(-2^{31})-(2^{31}-1)$	-	Steps	Follow Error

Description: When the encoder option is installed, this register shows the encoder deviation from the calculated position (P\_IST).

### 8.2.19 Flwerrmax

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
22	FLWERRMAX	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	Steps	Follow Error Max

Description: The maximum allowed value in FLWERR before an error is triggered. If FLWERRMAX = 0, the error is disabled. See register 35 ([Err\\_Bits](#), page 163) for a description of the error bit.

## 8.2 Internal registers

### 8.2.20 Command

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
24	COMMAND	32bit	R/W	0-127, 255-1000	0	-	Special command

**Description:** Used to issue commands to the motor. 0-128 are the normal FastMac commands. The values 128-255 are reserved. Command 256 will activate a new baud rate on the serial ports, and command 257 will synchronize the internal encoder position to the actual motor position.

- 257 Re-sync P\_IST and P\_ENCODER position
- 267 Reset the CPU.
- 268 Save to flash memory then reset the CPU.
- 269 Save to flash memory, then continue normal execution. NOTE: Some registers used only during startup!
- 316 Preset H3 encoder position (encoder opt. H3) with P\_NEW.
- 320 Set up the RS422 to support SSI encoder.
- 321 Read SSI encoder.
- 322 Read SSI encoder and convert from Gray code to binary.
- 342 Clear all flash sectors in the RXP area.
- 354 Preset encoder opt. H2, H3 and H4, P\_IST and P\_SOLL with P\_NEW, Follow error disabled temporarily to avoid errors.
- 394 Emergency stop with deceleration.
- 395 Emergency stop without deceleration.



**Please note:** Several of the commands access the flash memory. Please notice that the flash memory have restricted number of write cycles (100000 write cycles) and can be permanent damaged if this number is exceeded.

## 8.2

## Internal registers

### 8.2.21 Status bits

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
25	STATUSBITS	32bit	R	-	-	Special	Run Status

Description: This register contains a number of status bits that indicate status of various functions in the motor. The status bit are:

- 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: Reserved
- Bit 9: Internal encoder error
- Bit 10: H3 calibration data present
- Bit 11: H3 linearisation table recorded
- Bit 12: General Error (Same as register 35 bit 0).
- Bit 13: H3 calibration data locked
- Bit 14: Electromechanical brake active (Int./Ext.)
- Bit 15: Closed loop lead/lag detected. Bit also activates the LI LED if no Ethernet or CANopen option is present.
- Bit 16: Closed loop activated
- Bit 17: Internal encoder calibrated (ready for closed loop)
- Bit 18: Standby current is being used in stead of Running current
- Bit 19: Reserved
- Bit 20: Internal encoder OK
- Bit 21: Ethernet Sync is activated. Motor will only change the velocity and position when sync pulse is received.
- Bit 22: In target position if encoder position and P\_SOLL are within the window.
- Bit 23: STO channel A status
- Bit 24: STO channel B status
- Bit 25-26: External memory size: 0 = 0 kbit, 1 = 4kbit, 2 = 64kbit

## 8.2 Internal registers

---

### 8.2.22 Temp

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
26	TEMP	32bit	R	0...127	-	-2.27 - uses offset	Temperature

Description: Temperature measured inside the motor electronics.  
The approximate temperature in degrees Celsius is calculated from the value in this register using the formula:  $T_c = 2.27 * \text{Value}$ .

### 8.2.23 Min\_P\_IST

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
28	MIN_P_IST	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	Steps	Position Limit Min

Description: Position limit for movement in the negative direction. The motor can be configured to stop automatically when it reaches this position.

## 8.2 Internal registers

---

### 8.2.24 Max\_P\_IST

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
30	MAX_P_IST	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	Steps	Position Limit Max

Description: Position limit for movement in the positive direction. The motor can be configured to stop automatically when it reaches this position.

### 8.2.25 Acc\_Emerg

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
32	ACC_EMERG	32bit	R/W	1-500000	10000	RPM/s	Error Acceleration

Description: The motor will use this acceleration during an emergency stop.

### 8.2.26 IN\_POSITION\_WINDOW

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
33	IN_POSITION_WINDOW	32bit	R/W	$0 - (2^{31}-1)$	20000	Counts	In position window

Description: Selects how close the internal encoder position must be to the target Position (P\_SOLL) to set the InPhysical-Position status bit and prevent further AutoCorrection.

### 8.2.27 IN\_POSITION\_COUNT

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
33	IN_POSITION_COUNT	32bit	R/W	0 - 100	2	Counts	Max. number of retries

Description: The number of times to attempt AutoCorrection. A value of zero disables AutoCorrection.

## 8.2

## Internal registers

### 8.2.28 Err\_Bits

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
35	ERR_BITS	32bit	R/W		0	Special	Errors

Description: This register contains all information about present errors - if any.

#### Error bits:

- Bit 0: General error. Will always be set together with one of the other bits.
- Bit 1: Follow error
- Bit 2: Output driver. Bit is set if one of the user outputs is short circuited.
- Bit 3: Position Limit
- Bit 4: Low bus voltage
- Bit 5: Over voltage
- Bit 6: Temperature too high (>90°C)
- Bit 7: Internal error (Self diagnostics failed)
- Bit 8: Encoder Lost Position (Absolute Multi-turn Encoder option, H3).
- Bit 9: Encoder Reed Error (Absolute Multi-turn Encoder option, H3).
- Bit 10: Encoder Communication Error (Absolute Multi-turn Encoder option, H3).
- Bit 11: SSI encoder
- Bit 12: Closed loop
- Bit 13: External memory
- Bit 14: Single turn encoder error (H2)
- Bit 27: Safe Torque Off

If any of these bits are set, the motor is in a state of error, and will not move until all the errors have been cleared.

Some of the errors can be cleared by writing zero to this register.

Other errors will require hardware fixes or intervention, such as allowing the motor cool down or adjusting the power supply voltage.

### 8.2.29 Warn\_Bits

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
36	WARN_BITS	32bit	R/W		0	Special	Warnings

Description: Warning bits:

- Bit 0: Positive limit active. This bit will be set as long as the positive limit is active.
- Bit 1: Negative limit active. This bit will be set as long as the negative limit is 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 has been above 80°C
- Bit 7: SSI encoder
- Bit 8: Driver overload

These bits provide information on both the actual state and remembered state of the end position limits, the supply voltage and the temperature. These are used for diagnostic purposes as well as handling position limit stops, also after the motor may have left the end position mechanically.

## 8.2 Internal registers

### 8.2.30 Start mode

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
37	STARTMODE	32bit	R/W	-	0	-	Startup Mode

Description: The motor will switch to this mode after power up. This is also the mode that is used when a zero search has been completed. See [Mode\\_Reg](#), page 153 for a list of possible modes.

### 8.2.31 P\_Home

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
38	P_HOME	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	Steps	Zero Search Position

Description: The zero point found is offset with this value.

### 8.2.32 V\_Home

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
40	V_HOME	32bit	R/W	$\pm 1-300000$ (0.01-3000RPM)	5000 (50 RPM)	RPM/100	Zero Search Velocity

Description: The velocity used during zero search. Set a negative velocity to search in the negative direction.

### 8.2.33 Home mode

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
42	HOMEMODE	32bit	R/W	0,13,14	0	-	Zero Search Mode

Description: Selects the zero search that should start on power up.  
A value of 13 will use sensor type 1, while a value of 14 will use sensor type 2.  
Select 0 (default) if no automatic zero search must be done after power up.



## 8.2 Internal registers

### 8.2.34 Absolute encoder position

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
46	ABSENCODER	32bit	R	H2 (0-409500) H3 $((2^{31})-(2^{31}-1))$ H4 $((2^{31})-(2^{31}-1))$	0	-	Absolute Encoder Position

**Description:** If one of the encoder options are present in the motor this register monitors the position value. The value are shown in 2 different formats depending on which encoder option that is present.

#### **H2 encoder option:**

The register contains the absolute single turn position shown in the range 0-409500 counts.

#### **H3 + H4 encoder option:**

The register contains the absolute multi turn position for the whole 32 bit signed range.

### 8.2.35 EXTENCODER2

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
47	EXTENCODER2	32bit	R	$(-2^{31})-(2^{31}-1)$	0	-	SSI Encoder Value

**Description:** This is the actual encoder position data received from the external SSI encoder. Some SSI encoders output Gray coded values. The firmware offers the possibility to do the Gray code to binary conversion before updating the EXTENCODER2 register with the actual position.

**Example:** An SSI encoder outputs the position in binary. We want to sample, and update the EXTENCODER2 register 47 without any conversion. This can be done by use command 321.

If the SSI encoder outputs the position in Gray code, the value can be converted to binary before updating the EXTENCODER2 register by using command 322 instead.

For further description of the external encoder interface using SSI format please consult [The SSI interface principle of operation.](#), page 96

### 8.2.36 FlexRegister

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
48	FlexRegister	32bit	R	$(-2^{15})-(2^{15}-1)$	0	-	

**Description:** A register that can be set up to contain different bits from several registers. 16 bits are available. See also [Flexible Register setup](#), page 259.

## 8.2 Internal registers

### 8.2.37 Pn

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
49,51, 53,55, 57,59, 61,63	Pn	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	Steps	Position n (Pn)

Description: These eight general-purpose position registers are referred to as P1... P8 and can be used to make absolute or relative movements in several different ways, either from the user program or via the serial interfaces. See also the sections on FastMac commands, and the P\_NEW register description (*P\_New*, page 179).

### 8.2.38 Vn

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
65-72	Vn	32bit	R/W	$\pm 1-300000$ (0.01-3000RPM)	25000 (250 RPM)	RPM/100	Velocity n (Vn)

Description: These eight general-purpose Velocity registers are referred to as V1...V8 and can be used to change the velocity in several different ways, either from the user program or via the serial interfaces. See also the sections on FastMac commands.

### 8.2.39 An

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
73-76	An	32bit	R/W	1-500000	1000	RPM/s	Acceleration n (An)

Description: These four general-purpose Acceleration registers are referred to as A1... A4 and can be used to change the acceleration in several different ways, either from the user program or via the serial interfaces. See also the sections on FastMac commands.

### 8.2.40 Tn

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
77-80	Tn	32bit	R/W	0-511	511	5.87 mA	Current n (Tn)

Description: These four general-purpose Torque registers are referred to as T1...T4 and can be used to change the Running current in several different ways, either from the user program or via the serial interfaces. See also the sections on FastMac commands. They select the current in the motor windings used during movement.

## 8.2 Internal registers

### 8.2.41 Analogue Filtered

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
<b>81-88</b>	Analogue Filtered	32bit	R	0-4095	0	1.221mV	N/A

Description: These eight registers hold the software-filtered analogue value of each of the eight I/O's: IO-1 to IO-8. Their values are updated every ten milliseconds. See the AFZUP\_xx registers 100-106 for the filter parameters. Important: Also read the section on analogue filters in this manual.

To use the unfiltered values of the inputs for faster updates, but with no noise immunity, use registers 89-96 instead ([Analogue In](#), page 167).

An input voltage of 5.00 Volts corresponds to a register value of 4095.

See also : [Analogue input filters](#), page 23

### 8.2.42 Analogue In

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
<b>89-96</b>	Analogue Input	32bit	R	0-4095	-	1.221 mV	N/A

Description: These eight registers hold the unfiltered analogue value of each of the eight I/Os: IO-1 to IO-8. Their values are updated approximately every 1 ms.

To use the filtered values of the inputs for better noise immunity, use registers 81-88 instead ([Analogue Filtered](#), page 167).

An input voltage of 5.00 Volts corresponds to a register value of 4095.

See also : [Analogue input filters](#), page 23

### 8.2.43 Busvol

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
<b>97</b>	BUSVOL	32bit	R	0-4095	-	26.67 mV	Bus Voltage

Description: The supply voltage inside the motor is continually measured and stored in this register. This value is the basis for the warnings and errors of Low Bus Voltage and Over Voltage.

### 8.2.44 Min\_Busvol

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
<b>98</b>	MIN_BUSVOL	32bit	R/W	0-4095	15	26.67 mV	Min Bus Voltage

Description: Trigger point for under-voltage

## 8.2 Internal registers

### 8.2.45 Encoder\_Typ

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
99	ENCODER_TYPE	32bit	R	0-10	-	-	"Tooltip on motor"

Description: This register monitor which encoder option that is installed in the motor.

- 0 = No encoder
- 1 = Absolute single turn encoder 10 bit (H2)
- 2 = Absolute multi turn encoder (H3)
- 3 = Absolute single turn encoder 12 bit (H2)
- 4 = Absolute single turn + multi turn encoder (H4)

### 8.2.46 Afzup\_WriteBits

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
100	AFZUP_WriteBits	32bit	R/W	-	0	Special	N/A handled on the Filter Setup screen

Description: When changing values for the analogue input filter parameters, this register is used in combination with registers 102-106. First, all of the registers 102-106 must be loaded with the values to be used for one or more analogue input filters. Then the lower eight bits in this register are set to select which inputs the parameters in registers 102-106 should control.

The firmware will detect this and copy the parameter values from registers 102-106 to internal storage. Once this has been completed, the firmware sets bit 15 in this register to show that registers 102-106 are free to receive new values for programming the remaining inputs with other filter parameters. To use the same filtering for all analogue inputs, this register can be loaded with 255 (hex FF).

### 8.2.47 Afzup\_ReadIndex

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
101	AFZUP_Read Index	32bit	R/W	0, 1-8, 32768-32775	0	Special	N/A handled on the Filter Setup screen

Description: This register makes it possible to read back the analogue input filter parameters for one analogue input at a time. To select a new input, write a value of 1 to 8 to this register and wait for bit 15 to be set high. When bit 15 has been set by the firmware, the registers 102-106 have been loaded with the filter parameters currently used by that analogue input.

### 8.2.48 Afzup\_ConfMin

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
102	AFZUP Conf Min	32bit	R/W	0-4094	0	1.221 mV	Confidence Min

Description: The minimum confidence limits for analogue inputs are set and read back using this register in combination with the read and write 'command' registers 100 and 101. If a new raw sample value is less than the value in this register, it is simply discarded and the filtered input value in registers 81-88 will not change. A value of zero in this register will effectively disable the minimum confidence check.

## 8.2 Internal registers

### 8.2.49 Afzup\_ConfMax

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
103	AFZUP_Conf Max	32bit	R/W	1-4095	4095	1.221 mV	Confidence Max

Description: The maximum confidence limits for analogue inputs are set and read back using this register in combination with the read and write 'command' registers 100 and 101. If a new raw sample value is larger than the value in this register, it is simply discarded and the filtered input value in registers 81-88 will not change. A value of 4095 in this register will effectively disable the maximum confidence check.

### 8.2.50 Afzup\_MaxSlope

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
104	AFZUP_Max Slope	32bit	R/W	2-4095	4095	1.221 mV	Max Slope

Description: The maximum slopes per sample for analogue inputs are set and read back using this register in combination with the read and write 'command' registers 100 and 101. If a new raw sample value on an analogue input lies farther from the previous filtered value in registers 81-88, the new sample will be modified to lie at most MaxSlope units from the filtered value. This is used to suppress noise and limit acceleration. Note that the value is optionally filtered after being slope limited, in which case the effective slope limitation will be divided by the filter ratio. A value of 4095 will effectively disable slope limitation.

### 8.2.51 Afzup\_Filter

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
105	AFZUP_Filter	32bit	R/W	1-64	64	64 <sup>th</sup> of new sample	Filter (on the Filter setup screen)

Description: The final filtering of new samples on the analogue inputs can be selected using this register in combination with the read and write 'command' registers 100 and 101. The final filtered value results from taking Filter/64 of the new sample plus (64-Filter)/64 of the old value and storing the result in registers 81-88. A value of 64 effectively disables this filtering, so the new sample simply replaces the old value.

### 8.2.52 FilterStatus

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
106	FilterStatus	32bit	R	0-65535	0		N/A (shown graphically)

Description: This register contains status bits for the analogue input filters. The lowest eight bits hold confidence errors for each of the eight inputs, while the highest eight bits hold the status of their slope errors. The filter status is updated each second. The confidence error bit will be set if more than half of the samples within the last second fell outside either of the confidence limits. The slope errors will be set if more than half of the samples within the last second were slope limited.

## 8.2 Internal registers

---

### 8.2.53 SSI\_SETUP1

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
107	SSI_Setup1	32bit	R/W	32Bit	25bit, 100kHz frequency pre- pare time = 100µs	*	Number of data bits SSI Clock Frequency Wait time Max. sample deviation Number of samples Read retries

\* Number of data bits. Clock frequency, Disable interrupts when Reading SSI

Description: SSI encoder interface setup bits:

- Bit 0-4: Number of data bits in each SSI transfer
- Bit 5-7: Number of samples for each SSI position reading
- Bit 8-15: SSI clock frequency in units of 10 kHz
- Bit 16-28: Max. sample deviation between each sample
- Bit 29-31: Read retries

See also: [SSI encoder/sensor interface](#), page 92

## 8.2 Internal registers

### 8.2.54 Settling Time

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
110	Settling Time	32bit	R/W	0-32676	0	ms	Settling time between retries

Description: When the internal encoder option is installed and register 34, InPositionCount, is non-zero so AutoCorrection is enabled, the value in this register defines how many milliseconds to wait after each movement attempt before testing whether the encoder position is within the target window as defined in register 33. This waiting time is often necessary to allow mechanical oscillations to die out.

### 8.2.55 SSI\_SETUP2

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
111	SSI_Setup2	32bit	R/W	32 bit	25bit, 100kHz frequency prepare time=100µs	-	Prepare time GRAY conversion

Description: SSI encoder interface setup bits:  
Bit 0-7: Prepare time in milliseconds  
Bit 8: Gray to bin conversion (1=on, 0=off)

### 8.2.56 Sample 1-4

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
112-115	SAMPLE1-4	32bit	R/W	-	0	-	N/A

Description: Up to four registers can be set up to be sampled into buffers for diagnostic purposes. These registers define which registers are sampled. All of the registers 1-255 can be sampled.  
A value of zero in any of these four registers will cause the corresponding sample buffer to contain zeroes.  
See registers 116-119 for more information on the sampling system.

### 8.2.57 Rec\_Cnt

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
116	REC_CNT	32bit	R/W	-	0	-	N/A

Description: This value specifies the number of samples to take for each of the sampled registers selected in registers 112-115. This value must never be set larger than the value in the read-only register 119. Sampling will stop automatically after the specified number of samples has been taken.

## 8.2 Internal registers

### 8.2.58 S\_Time

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
117	S_TIME	32bit	R/W	-	1	-	N/A

Description: This value selects the time in milliseconds between samples of the registers selected in registers 112-115.

### 8.2.59 S\_Control

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
118	S_CONTROL	32bit	R/W	-	0	-	NA

Description: This value controls the sample system. It can assume three different values:  
A value of zero is set by the firmware after all sampling has completed.  
A value of one will initialize the sample system.  
A value of two will start a new sample sequence and set this register to zero at completion.  
The sampled values are read back using the command hex 53 *SMC\_READSAMPLE*.

### 8.2.60 Buf\_Size

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
119	BUF_SIZE	32bit	R	-	-	-	N/A

Description: This read-only register contains the maximum length of the sample buffers used to sample the registers selected in registers 112-115.  
Register 116 should never be set to a value higher than the value in this register.

### 8.2.61 Index\_Offset

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
120	INDEX_OFFSET	32bit	R	0-1599	-	Steps	Tests-

Description: This register can be selected to receive the absolute value of the internal encoder where the Zero search/home position was found during Zero Search. This is selected by bit 0, Use Index, in register 122. It requires that the internal encoder option is installed.



## 8.2 Internal registers

### 8.2.62 Modbus\_setup

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
121	Modbus_setup	32bit	R/W	-	-	-	-

Description: The traditional MacTalk channel can be setup to run Modbus protocol according to these settings:

Bit description:

Bit 0: Enabled

Bit 1: Type (0 = RTU, 1 = ASCII)

Bit 2-3: Parity (0=None, 1=Odd, 2=Even)

Bit 4: Data bits (0=7 bits, 1=8 bits)

Bit 5: Stop bits (0=1 bit, 1=2 bit)

When enabled, the motor can still be connected with the MacTalk protocol the 1.st. second after power on. This way the Modbus settings can be disabled again if necessary.

### 8.2.63 Zero\_Search\_Bits

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
122	Zero_Search_Bits	32bit	R/W	-	0	Special	Advanced-Zero Search

Description: This register contains configuration bits, that define how Zero search should be carried out.

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 zero search time out

## 8.2

## Internal registers

### 8.2.64 Setup\_Bits

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
124	SETUP_BITS	32bit	R/W	-	0	Special	Don't start program after power up. Invert motor direction. External Encoder Support Auto encoder synchronize etc. etc. (se below)

Description: Bit 0: Invert motor direction  
Bit 1: Do not start RxP program after power up.  
Bit 2-3: Select encoder input type. 0 = Disabled, 1 = Quadrature, 2 = Pulse/direction  
Bit 4: Reserved  
Bit 5: Synchronize encoder position to P\_IST after change to active mode.  
Follow error = 0  
Bit 6: InPhysPosMode (If set, recalc InPhysPos continuously. If 0, only after stop)  
Bit 7-9: Reserved  
Bit 10: EncoderToP\_IST (Automatically transfer the absolute single turn encoder position to P\_IST at power up)  
Bit 11: Multiturn (Automatically transfer the multi turn encoder position to actual P\_IST at power up)  
Bit 12: KeepExtEncoder (Do not zero the external encoder count on startup)  
Bit 13: KeepSSIValue (Do not zero the SSI data register on startup)  
Bit 14: UseBeckhoff (use the Beckhoff variant of CAN - required by TwinCAT)  
Bit 15: Reserved  
Bit 16: External Encoder counting direction (1=inverse)  
Bit 17: Disable position limit error. Motor stays in active mode on position limit.  
Bit 19: Disable brake (int./ext.) temporarily in order to move the shaft in passive mode.  
Bit 20: Disable SSI encoder error. Motor can stay in active mode even if SSI position is wrong.  
Bit 21: "Low bus voltage" gives an Error  
Bit 22: "Low bus voltage" sets the motor in Passive mode.  
Bit 23: "Low bus voltage" sets V\_SOLL to 0 RPM.  
Bit 24: Enable closed loop  
Bit 25: Enable closed loop current control  
Bit 28: Position limits without memory.

These individual bits are used to control various functions in the firmware. Bits marked in grey are not fully available - consult JVL !.

### 8.2.65 IOsetup

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
125	IOSETUP	32bit	R/W	-	0	Special	Inputs/Outputs

Description: This register controls the eight IO's: IO-1 to IO-8. These pins can be used either in input mode as combined digital and analogue inputs or used in output mode as digital outputs. The lowest eight bits in this register can be used to individually invert the active level of the digital inputs. The highest eight bits are used to select the corresponding pin as an output.

## 8.2 Internal registers

### 8.2.66 Turntable\_Mode

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
126	TURNTABLE_MODE	32bit	R/W	-	0	Special	Turn Table -Mode

**Description:** Sorry but this feature is not yet available.

In turntable mode, the motor controls the revolution of a turntable that has the number of positions specified in register 127, Turntable\_Size.

This means the same position will be reached after rotating this number of steps in either direction.

This register selects one of three modes that define how the motor should move to a new position when the P\_SOLL register is changed.

If the value of this register is zero, the motor will not operate in turntable mode.

In mode 1, the motor will always move to a new position by turning in a positive direction. So to move one step backwards, it must instead move Turntable\_Size-1 steps forward.

In mode 2, the motor will always move to a new position by turning in a negative direction.

In mode 3, the motor will move in the direction that takes the smallest number of steps to reach the new position.

Note that the motor will not move at all if the new position in register P\_SOLL is either negative or larger than the value of register 127, Turntable\_Size.

### 8.2.67 Turntable\_Size

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
127	TURNTABLE_SIZE	32bit	R/W	-	0	Steps	Turn Table - Size

**Description:** Sorry but this feature is not yet available.

If turntable mode is selected in register 126, the number of steps needed for a full revolution of the turntable is set in this register. Note that the register P\_SOLL must always have a value between zero and the value in this register minus one. Negative values are not allowed for P\_SOLL or Turntable\_Size.

### 8.2.68 NL\_Mask

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
129	NL_MASK	32bit	R/W	-	0	IO Mask	Dedicated Inputs Negative Limit Input

**Description:** Selects which one of the eight IO pins to use for the dedicated function of Negative Position Limit.

Exactly one bit must be set, and the IO pin must be configured in register 125 as an input.

**Example:** If input 7 is to be used for the Negative Input Limit, write  $2^7 = 128$  to this register.

## 8.2 Internal registers

---

### 8.2.69 PL\_Mask

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
130	PL_MASK	32bit	R/W	-	0	IO Mask	Dedicated Inputs - Positive Limit Input

Description: Selects which one of the eight IO pins to use for the dedicated function of Positive Position Limit.  
Exactly one bit must be set, and the IO pin must be configured in register 125 as an input.

Example: If input 8 is to be used for the Positive Input Limit, write 27 = 128 to this register.

### 8.2.70 Home\_Mask

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
132	HOME_MASK	32bit	R/W	-	0	IO Mask	Dedicated inputs. Home Input

Description: Selects which one of the eight IO pins to use for the dedicated function of Home Input.  
Exactly one bit must be set, and the IO pin must be configured in register 125 as an input.

Example: If input 2 is to be used for the Home Input, write 21 = 2 to this register.

## 8.2 Internal registers

### 8.2.71 Input\_Filter\_Mask

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
135	INPUT_FILTER_MASK	32bit	R/W	-	0	IO Mask	IOx digital input filter enabled

**Description:** This register controls filtering of each of the eight IO pins that are used as digital inputs. If the bit corresponding to the input number is set in this register, the input value will be filtered to a new logical level is only accepted after that level has been measured on the hardware pin for the number of milliseconds specified in register 136. If the bit is not set, the input will be updated directly from the hardware value every 100 microseconds. Please read the section on Digital Input filters in this manual.

### 8.2.72 Input\_Filter\_Cnt

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
136	INPUT_FILTER_CNT	32bit	R/W	-	5	ms	Input filter time

**Description:** The filtering of all of the eight digital inputs is controlled by the value in this register together with register 135. The input must be sampled at the same value for the specified number of milliseconds in this register to be accepted as the new filtered value. See also the section on Digital Input Filters in this manual.

### 8.2.73 Inpos\_Mask

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
137	INPOS_MASK	32bit	R/W	-	0	IO MASK	Dedicated Outputs - In Position

**Description:** Selects which one of the eight IO pins to use for the dedicated function of In Position Output. Exactly one bit must be set, and the IO pin must be configured in register 125 as an output.

The In Position output will then be set after a movement has completed.

**Example:** If output “n” is to be used for the In Position Output, write  $2^{(n-1)}$  to this register.

### 8.2.74 Error\_Mask

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
138	ERROR_MASK	32bit	R/W	-	0	IO Mask	Dedicated Outputs - Error

**Description:** Selects which one of the eight IO pins to use for the dedicated function of Error Output. Exactly one bit must be set, and the IO pin must be configured in register 125 as an output.

The Error Output will set be set when any error is set.

See register 35 (*Err\_Bits*, page 163) for more information on errors.

**Example:** If output “n” is to be used for the Error Output, write  $2^{(n-1)}$  to this register.

## 8.2 Internal registers

---

### 8.2.75 Acceptance voltage

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
139	Acceptance Voltage	32bit	R/W	32bit	2052 (18 Volt)	8.764 mV	Acceptance Voltage

Description: The acceptance Voltage, is the voltage required at the CVI supply terminal (PWR connector) for the program to start up.  
The typical and recommended nominal voltage is 24VDC but if a lower voltage is used in for example battery powered applications please make sure that the acceptance voltage also covers that the battery is much lower than if its fully charged.

### 8.2.76 Acceptance count

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
140	Acceptance Count	32bit	R/W	32bit	100	-	Acceptance Count

Description: Acceptance Count is the number of times a voltage above the acceptance voltage must have been measured before the program starts.  
The basic idea behind this register/function is to make sure that the startup is completed and the supply voltage is stable.

## 8.2 Internal registers

### 8.2.77 Save voltage

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
141	Save voltage	32bit	R/W	32bit	1710 (15 Volt)	8.764 mV	Save Voltage

Description: This register sets the voltage level where the program shuts down and all motor activity stops.

### 8.2.78 CVI\_VOLT

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
143	CVI_VOLT	32bit	R			8.764 mV	N/A

Description: The measured control voltage.

### 8.2.79 P\_New

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
144	P_NEW	32bit	R/W	$(-2^{31})-(2^{31}-1)$	0	Counts	N/A

Description: This register can be used to change both of the registers P\_SOLL and P\_IST in one operation. This can be used to correct or offset the current position without performing a movement. The register value can be copied to P\_IST and P\_SOLL using FastMac command 23, or it can be added with sign to both of these registers using FastMac command 24.

### 8.2.80 Baud\_Rate

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
146	BAUD_RATE	32bit	R/W	0-7	1	-	Baud Rate

Description: 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

The firmware will automatically update the baud rate after this value is changed over the serial interface (RS485) once the motor has finished transmitting all data bytes that are queued.

## 8.2 Internal registers

---

### 8.2.81 Tx\_Delay

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
147	TX_DELAY	32bit	R/W	1-255	15	Bits	Transmit Delay

Description: The time to wait before the response is transmitted. The unit corresponds to the time of one bit at the current baud rate.

Many PLCs and communications processors require a minimum delay after they have sent a command to the motor before they are able to receive the response.

### 8.2.82 Group\_Id

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
148	GROUP_ID	32bit	R/W	0-255	-	-	Group Id

Description: The group ID of the motor. The motor will accept data from a group write command only if the group ID number in the command matches this number. The idea is that several motors can have the same group ID so they can be updated with new register values in parallel to save transmission time.

### 8.2.83 Group\_Seq

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
149	GROUP_SEQ	32bit	R	0-255	-	-	N/A

Description: The last received group write sequence.

### 8.2.84 My\_Addr

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
150	MY_ADDR	32bit	R/W	0-254	254	-	Motor Address

Description: The motor address. Data communicated over the serial interface will only be accepted if the address byte in the command is either equal to this value or has the value 255, which means broadcast to all motors.



## 8.2

# Internal registers

### 8.2.85 Motor type

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
151	MOTORTYPE	32bit	R	64-xx		-	"Status Bar"

Description: The actual motor type or controller type. Please notice that this register is in common for all motor families and sizes from JVL including the MAC servomotor range. The list show all available types within stepper motors including versions with build in brake. The motor type register will contain the following values dependent at the actual motor type.

Motor type	Technology	Decimal value	Hex value
SMC85	Stepper motor controller	80	
MIS340	Stepper motor rotary	81	
MIS341	Stepper motor rotary	82	
MIS342	Stepper motor rotary	83	
MIS343	Stepper motor rotary	84	
MIS344	Stepper motor rotary	85	
MIS345	Stepper motor rotary	86	
MIS430	Stepper motor rotary	90	
MIS431	Stepper motor rotary	91	
MIS432	Stepper motor rotary	92	
MIS433	Stepper motor rotary	93	
MIS434	Stepper motor rotary	94	
MIS435	Stepper motor rotary	95	
MIS511	Stepper motor rotary	100	
MIS512	Stepper motor rotary	101	
MIS513	Stepper motor rotary	102	
MIS514	Stepper motor rotary	103	
MIS515	Stepper motor rotary	104	
MIS170	Stepper motor rotary	120	
MIS171	Stepper motor rotary	121	
MIS172	Stepper motor rotary	122	
MIS173	Stepper motor rotary	123	
MIS174	Stepper motor rotary	124	
MIS175	Stepper motor rotary	125	
MIS176	Stepper motor rotary	126	
SMC66	Stepper motor controller	150	
MIS230x	Stepper motor rotary	151	
MIS231x	Stepper motor rotary	152	
MIS232x	Stepper motor rotary	153	
MIS233x	Stepper motor rotary	154	
MIS234x	Stepper motor rotary	155	
MIL230x	Stepper motor linear	200	
MIL231x	Stepper motor linear	201	
MIL232x	Stepper motor linear	202	
MIL233x	Stepper motor linear	203	
MIL234x	Stepper motor linear	204	
MIL340x	Stepper motor linear	250	
MIL341x	Stepper motor linear	251	
MIL342x	Stepper motor linear	252	
MIL343x	Stepper motor linear	253	
MIL344x	Stepper motor linear	254	

This value is read-only and is programmed into the motor during manufacturing.

## 8.2 Internal registers

### 8.2.86 Serial\_Number

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
152	SERIAL-NUMBER	32bit	R	-	-	-	"Status Bar"

Description: The serial number of the motor.  
This value is read-only and is programmed into the motor during manufacturing.

### 8.2.87 Checksum

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
154-155	CHECKSUM	32bit	R	0-65535	-		"Tooltip on motor"

Description: Firmware checksum.  
This value is read-only and is programmed into the motor during firmware update.

### 8.2.88 Hardware\_Rev

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
156	HARDWARE_REV	32bit	R	0-65535	-	Major*16+ Minor +16384	"Tooltip on Motor"

Description: The revision of the hardware. This value is read-only and is programmed into the motor during manufacturing.

### 8.2.89 Max\_Voltage

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
157	MAX_VOLTAGE	32bit	R	0-100	*	Volt	"Tooltip on Motor"

Description: The maximum allowed voltage on the bus. If the bus voltage exceeds this value, the motor will enter an error state.  
This value is read-only and is programmed into the motor during manufacturing. It reflects the rating of the hardware components. Supplying a higher voltage can damage the electronics components permanently. If in doubt, it is strongly recommended to first supply 24 Volts and connect the motor to MacTalk. In MacTalk this value can be read by holding the mouse cursor over the image of the motor in the lower right of the main window.

Bit 0-15: Max voltage on bus

Bit 16-31: Full scale motor current in mARMS

## 8.2 Internal registers

### 8.2.90 Available\_IO

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
158	AVAILABLE_IO	32bit	R	-	-	IO MASK	N/A

Description: Defines what IO that are available on the connector.  
This value is read-only and is programmed into the motor during manufacturing. Service personnel may ask for this value to identify the type of connector board mounted on the motor. The values are not documented here.

Bit 0-15: Defines what IO that are available on the connector

Bit 16-31: The max current to the motor in the same units as Running current register 7

### 8.2.91 Bootloader\_Ver

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
159	BOOTLOADER_VER	32bit	R	0-65535	-	Major*16+ Minor +16384	"Tooltip on Motor"

Description: The version of the boot-loader.  
This value is read-only and is programmed into the motor during manufacturing

### 8.2.92 Not saved

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
160	NOTSAVED	32bit	R/W	0-65535	0	-	N/A

Description: This register is not used internally, but will always be 0 after power on. Please note that MacTalk uses this register

### 8.2.93 Option\_Bits

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
165	OPTION_BITS	32bit	R	0-65535	-	-	"Tooltip on motor"

Description: This register contains information about what options 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

Bit 1,9: DeviceNet fieldbus

## 8.2 Internal registers

---

### 8.2.94 Fbus\_Node\_Id

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
166	FBUS_NODE_ID	32bit	R/W	0-255	5	-	Fieldbus - Node ID

Description: The node id on the fieldbus interface.

### 8.2.95 Fbus\_Baud

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
167	FBUS_BAUD	32bit	R/W	0-8	2	-	Fieldbus - Baud Rate

Description: The baudrate used on the CANopen interface (optional).

- 0: 1000 kbit/s
- 1: 800 kbit/s (unsupported)
- 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

### 8.2.96 Module Type

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
168	MODULE_TYPE	32bit	R				(dedicated tab in MacTalk when module is present)

Description: 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

### 8.2.97 Ext\_Encoder

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
170	EXT_ENCODER	32bit	R/W	$(-2^{31})-(2^{31}-1)$	-	Counts	External Encoder

Description: This register counts the external encoder input at the multifunction I/O.

## 8.2 Internal registers

---

### 8.2.98 Ext\_Encoder\_Vel

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
172	EXT_ENCODER_VEL	32bit	R	$(-2^{31})-(2^{31}-1)$	-	Counts 16ms	External Encoder Velocity

Description: This register is updated with the velocity of the external encoder input. The velocity is measured every 16ms.

## 8.2 Internal registers

### 8.2.99 Internal\_Encoder\_Setup

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
175	Internal_Encoder_Setup	32bit	R/W	-	-	Special	N/A

The internal encoder has different settings available:

#### Hysteresis

Is used to prevent flickering of the angular position LSBs. Bit 0-1 set the hysteresis.

#### Resolution

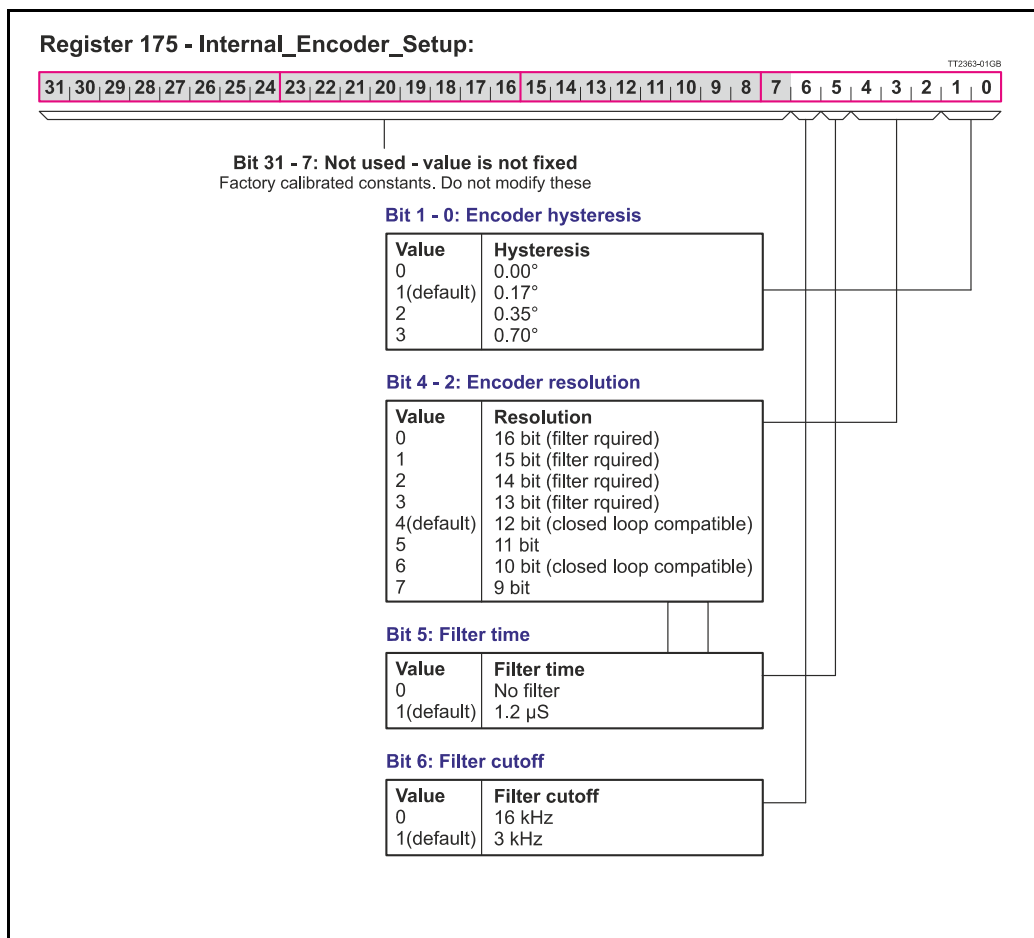
Bit 2-4 determines the resolution, i.e. number of counts in 1 revolution. If the motor is set up to output the encoder pulses, this will also be affected by changing the resolution.

#### Filter

The filter can be enabled to allow resolutions above 12 bits. Bit 5 set the filter.

#### Filter cutoff frequency.

A filter cutoff frequency on 3 kHz is recommended in the entire velocity range from 0 to 3000 RPM. The cutoff frequency is controlled by bit 6.



## 8.2 Internal registers

### 8.2.100 FW\_BUILD

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
176	FW_BUILD	32bit	R	0 - (2 <sup>32</sup> -1)	-	Counts	"Status bar"

Description: The actual firmware build number. This number is unique for each beta and released version.

### 8.2.101 InTargetPositionTime

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
177	InTargetPositionTime	32bit	R/W	0 - (2 <sup>32</sup> -1)	10	ms	N/A

Description: Time the motor must stand still before InTargetPosition (reg 25: Statusbits) flag is set.

### 8.2.102 BRAKE

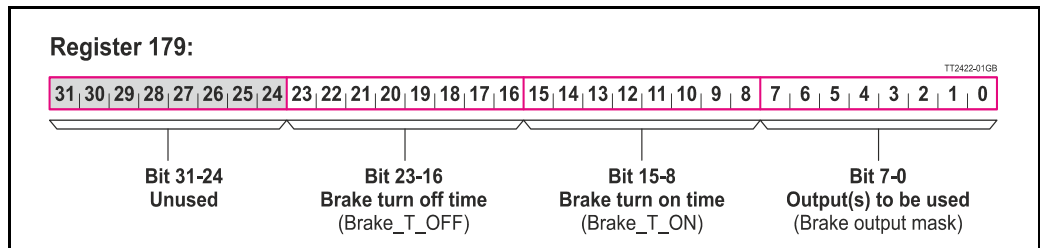
Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
179	BRAKE	32bit	R/W	0 - (2 <sup>32</sup> -1)	0	Special	N/A

Description: This register selects which one of the eight IO (IO1 to IO8) pins to use for the external brake.

Each of the first 8 bits in this register corresponds to 1 output pin. The selected IO pin must be configured in register 125 as an output.

See also [User outputs](#), page 28.

The bits have following function:



#### Example:

Output 4 is used for the Brake Output. T\_ON time is 40 ms and T\_OFF is 50 ms:

The following string will define this (shown in groups of 8 bits)

00000000 / 00110010 / 00101000 / 00001000 = in hex : 00 32 28 08

Bit 0-7: Brake output mask = 8, define that output 4 controls the brake.

Bit 8-15: Brake\_T\_ON - Time from motor is stopped until brake is activated = 40 ms

Bit 16-23: Brake\_T\_OFF - Time from the motor is activated until the brake is de-activated = 50 ms.

## 8.2 Internal registers

---

### 8.2.103 TICKS

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
202	TICKS	32bit	R/W	0 - (2 <sup>32</sup> -1)	0	ms	N/A

Description: Increments at a fixed rate of 1 count per millisecond. Starts at zero when CVI has been applied.

### 8.2.104 CUR\_SCALE\_MAX

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
212	CUR_SCALE_MAX	32bit	R/W	0 - 2047	2047	Counts	N/A

Description: Closed loop: Max running current in closed loop with "Current control" enabled. 2047 = 100 % of RUN\_CURRENT.

### 8.2.105 CUR\_SCALE\_MIN

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
213	CUR_SCALE_MIN	32bit	R	0 - 2047	1	Counts	N/A

Description: Closed loop: Minimum running current in closed loop with "Current control" enabled. 2047 = 100 % of RUN\_CURRENT. See also [Special settings](#), page 83.



## 8.2 Internal registers

### 8.2.106 CUR\_SCALE\_FACTOR

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
215	CUR_SCALE_FACTOR	32bit	R/W	1 - 10,000	500	Counts	N/A

Description: Closed loop: The slope of the velocity dependent current decrement rate. See also [Special settings](#), page 83.

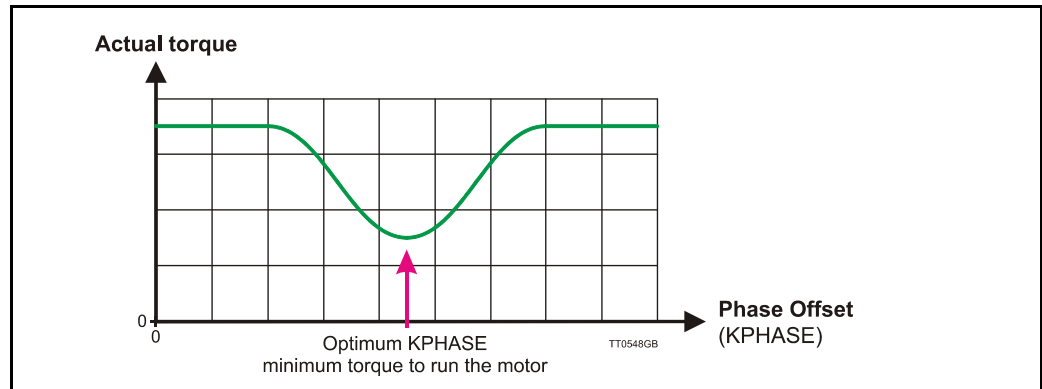
### 8.2.107 KPHASE

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
216	KPHASE	32bit	R/W	0-200	-	Counts	N/A

The KPHASE parameter is decisive for how far commutation of the motor is offset from the motor's actual position. KPHASE is velocity dependent, which means that it has increasing significance as motor velocity increases. The KPHASE parameter is factory calibrated, but can be adjusted by the user if necessary.

#### Finding the optimal KPHASE

The optimal KPHASE value is found by running the motor at high speed (2000 RPM) and observing the "Actual torque" in MacTalk. The actual torque will settle at its minimum value at the optimal KPHASE. The new KPHASE can be saved in flash and will then be used automatically after a reset.



## 8.2 Internal registers

### 8.2.108 ACTUAL\_TORQUE

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
217	ACTUAL_TORQUE	32bit	R	0-2047	-	Counts	Actual torque

Only used when the closed loop operation is active.

The register show the actual motor torque as a value from 0 to 2047 corresponding to 0-100% of the setting done in the motor "Running current" register.

- see also [Run\\_Current](#), page 155.

In passive mode = 0 %

In active mode without current control enabled = 100 %

In active mode with current control enabled = 0-100 % dependent on the load.

### 8.2.109 CUR\_SCALE\_INC

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
218	CUR_SCALE_INC	32bit	R/W	0-100000	2000	Counts	N/A

Used in closed loop operation.

The increment rate of the current determines how fast the actual torque must be increased when a rotor displacement has been measured. The rate is independent of the actual velocity. See also: [Special settings](#), page 83

### 8.2.110 CUR\_SCALE\_DEC

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
219	CUR_SCALE_DEC	32bit	R/W	0-100000	4000	Counts	N/A

Used in closed loop operation.

The increment rate of the current determines how fast the actual torque can be decreased. The rate is inverse proportional to the actual velocity, which means the it will decrease slower at higher velocities.

The slope is determined by register 215 see: [CUR\\_SCALE\\_DEC](#), page 190.

See also: [Special settings](#), page 83

### 8.2.111 XFIELD\_ADDR

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
222	XFIELD_ADDR	32bit	R/W	-	0	Special	N/A

The internal XFIELD allows the user to configure many different combinations of inputs and outputs from the motor. The RS422 interface can be used for connections like:

- External SSI encoder
- Serial communication channel
- Quadrature/pulse-direction encoder input
- Internal encoder output
- Quadrature/pulse-direction step generation output

This register controls the internal addressing for this setup. It is strongly recommended to use MacTalk as interface for the setup.

## 8.2 Internal registers

### 8.2.112 XFIELD\_DATA

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
223	XFIELD_DATA	32bit	R/W	-	0	Special	N/A

The internal XFIELD allows the user to configure many different combinations of inputs and outputs from the motor. The RS422 interface can be used for connections like:  
External SSI encoder  
Quadrature/pulse-direction encoder input  
Internal encoder output  
Quadrature/pulse-direction step generation output

This register controls the internal data for this setup. It is strongly recommended to use MacTalk as interface for the setup.

### 8.2.113 FlexRegSetup

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
224-231	FlexRegSetup	32bit	R/W	-	0	-	N/A

A set of registers that determine the address for each (0-15) bit in register 48: FlexRegister. See also [Flexible Register setup](#), page 259  
Each register in this range sets up 2 bits in the FlexRegister 48 = 16 bits in total.

### 8.2.114 FlexLEDSetup1

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
232	FlexLEDSetup1	32bit	R/W	-	102303769	-	N/A

Sets up LED "L2" and "L3" on the motor.  
If the motor has **no** Ethernet module or CANopen interface, it will be in the default configuration and the 2 LEDs "L2" and "L3" can be configured to show various conditions. In default configuration the L2 (green) and L3 (green), can be configured to display the status of a single bit in any register.

The default settings show:

L2 = "At velocity" bit from the Status register (25).

L3 = "In position" bit from the Status register (25).

**Setup:**

Bit 0-8: Register for L3

Bit 9-13: Bit for L3

Bit 16-24: Register for L2

Bit 25-29: Bit for L2

## 8.2 Internal registers

### 8.2.115 FlexLEDSetup2

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
233	FlexLEDSetup2	32bit	R/W	-	504954880	-	N/A

Sets up LED “LI” GREEN and “LI” RED on the motor.

If the motor has **no** Ethernet module or CANopen interface, it will be in the default configuration and the LED “LI” can be configured to display the status of a single bit in any register.

The default settings show:

LI (green) = OFF

LI (red) = “Closed loop lead/lag detected” bit from the Status register (25).

**Setup:**

Bit 0-8: Register for LI (green)

Bit 9-13: Bit for LI (green)

Bit 16-24: Register for LI (red)

Bit 25-29: Bit for LI (red)

### 8.2.116 V\_SOLL\_AUTO

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
236	V_SOLL_AUTO	32bit	R/W	-3,000.00 - 3,000.00	0	0.01 RPM	Auto correction velocity

In position mode the auto correction is run with V\_SOLL, but if V\_SOLL\_AUTO != 0 it will be used in stead.

### 8.2.117 V\_IST\_CALC

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
237	V_IST_CALC	32bit	R/W	-3,000.00 - 3,000.00	0	0.01 RPM	Actual velocity

The theoretical actual velocity.

### 8.2.118 MOTOR\_REV

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
238	MOTOR_REV	32bit	R/W	0-2 <sup>32</sup> -1	0	Revolutions	Event log -> Motor revolutions

Counts multiples of 409600 counts since power on. The value is added to the motor revolution counter in the Event log in order to keep the total amount of revolutions the motor has run in its entire lifetime.

## 8.2 Internal registers

### 8.2.119 EX\_CYCLIC\_SETUP

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
239	EX_CYCLIC_SETUP	32bit	R/W	-	0	Special	N/A

The actual cyclic setup from the Ethernet module.

Settings from the Ethernet module to setup the cycle period and how many percent the sync-pulse must be offset. These settings can be changed with the specific Ethernet protocol, but not directly in this register. This is read only.

Bit 0-15: Cycle period ( $\mu$ s)

Bit 16-31: Sync0 offset in percent.

### 8.2.120 EX\_CRC\_ERR

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
241	EX_CRC_ERR	32bit	R/W	0-2 <sup>32</sup> -1	0	Counts	N/A

CRC error counter of the internal communication between controller and Ethernet module.

### 8.2.121 V\_HOME\_CRAWL

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
242	V_HOME_CRAWL	32bit	R/W	0-300000 (0-3000 RPM)	0	0.01 RPM	Zero search crawl velocity

In Zero Search type 2, the “crawl” velocity is V\_HOME/64 by default.

If register 242:V\_HOME\_CRAWL is !=0, a user defined velocity is used – independent of V\_HOME. Please note that overshoot can occur if this velocity is set too high.

### 8.2.122 V\_HOME\_TIMEOUT

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
243	V_HOME_TIMEOUT	32bit	R/W	0-2 <sup>32</sup> -1	0	ms	Zero search time-out

In all Zero Search modes, the time out is by default 60 s. This delay can be changed by writing a value different from 0 to this register. The unit is milliseconds.

If 0, the Zero Search time out is 60000 ms. Else the value in this register is used.

## 8.2 Internal registers

### 8.2.123 TEMP\_LIMITS

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
244	TEMP_LIMITS	32bit	R	-	0	Special	N/A

The actual temperature limits in the motor.

A Warning will be set when the temperature exceeds this value:  
Bit 0-15: Warning limit (unit: degC)

An Error bit will be set when the temperature exceeds this value:  
Bit 16-31: Error limit (unit: °C)

The limits cannot be changed by the user. They are factory settings.

### 8.2.124 CL\_CATCH\_UP

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
245	CL_CATCH_UP	32bit	R/W	-	0	Special	Allowable over-speed. Follow error before overspeed

Bit 0-7: Allowable overspeed in percent (0-100)  
The motor can exceed the Max velocity (V\_SOLL) set in register 5 if this register is non zero. It means that the motor can catch up the follow error.  
Default 0 % = No over speed allowed.

Bit 8-31: Follow error limit before overspeed is used.  
The motor will exceed the Max velocity (V\_SOLL) if the Follow error (register 20) is higher than specified in this register. Default 5000 counts.

### 8.2.125 LOWBUSCVI\_CNT

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
252	LOWBUSCVI_CNT	32bit	R/W	-	10	Counts	N/A

Number of times in a row the voltage can be too low before error is set. Time between each measurement = 50 us.

For motors with serial numbers < 173000 this also sets the time before start saving of the internal Event log. From serial number > 173000 the Event log is saved continuously, every 1 seconds.

## 8.2 Internal registers

---

### 8.2.126 V\_ENCODER

Reg	Name	Size	Access	Range	Default	Unit	MacTalk name
253	V_ENCODER	32bit	R	-3,000.00 - 3,000.00	-	0.01 RPM	Internal encoder velocity

The actual velocity measured from the internal (H2/H4) encoder.





## **9 Building Sequential Programs**

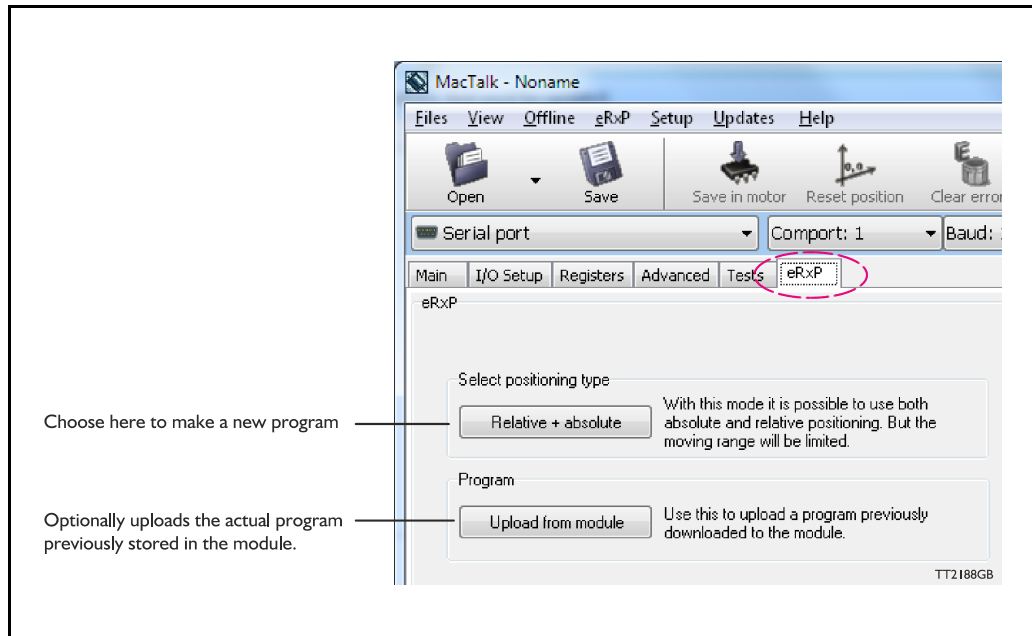
---

## 9.1 Getting started with programming

When using the MIS motors, almost any kind of program can be created using a set of user friendly icons.

Make the required choice on the eRxP Programming tab.

The name eRxP refers to the programmable module (R-module) from the MAC motor series. [e]mbedded [R]-module number [x] [P]rogramming



After making one of these 2 choices, the program window will be opened.

## 9.2

# Programming Main window

The main window for creating a new program or editing a program is shown below:

**RxP menu**  
Main menu for creating a new program, Verifying program size and other basic details for the motor.

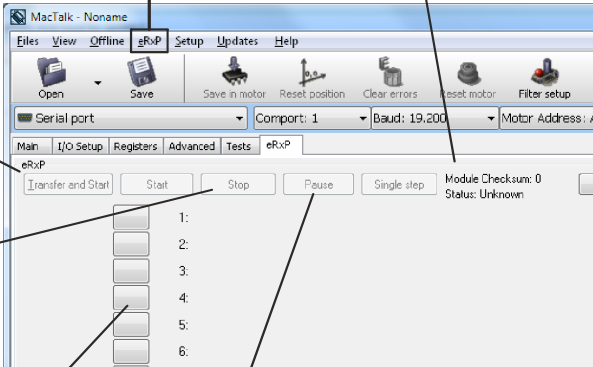
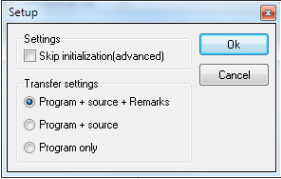
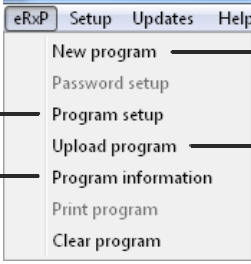
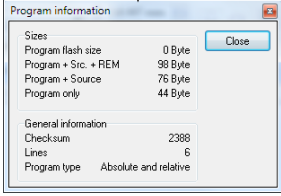
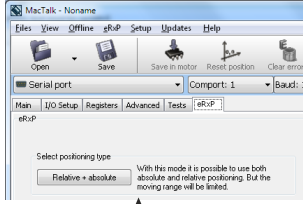
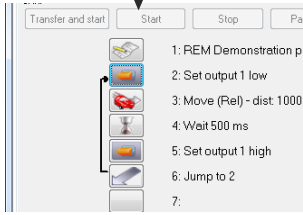
**Transfer & Start**  
Will transfer the complete program and start it. Use *Stop* or *Pause* to stop it.

**Stop**  
Use this button if the program must be stopped.

**Program lines**  
Each button represent a program line. By pushing the button a command can be entered at the program line.

**Status texts**  
The message *Program not transferred* means that there is a difference between the program seen on the screen and the actual program in the module. This can happen if the program have been edited but not transferred. *Status: Running* (or *Stopped*) refers to the program in the module.

**Pause**  
Use this button if the program must be paused. By paused means that actual program line executed is temporary paused. When paused the single step feature can be used to debug the program.

TT2189GB

## 9.3

# Programming menu

The menu found at the top of the main window gives access to the following options:

The diagram illustrates the 'eRxP' menu and its associated sub-windows. The menu items and their descriptions are as follows:

- New program**: Start to make new program.
- Password setup**: Password protection. See separate chapter for description.
- Program setup**: Upload the program from the module to MacTalk
- Upload program**: Upload the program from the module to MacTalk
- Program information**: Print the actual program
- Print program**: Print the actual program
- Clear program**: Erases the program

The 'Program information' sub-window displays the following data:

Sizes	
Program flash size	0 Byte
Program + Src. + REM	10 Byte
Program + Source	10 Byte
Program only	0 Byte

General information	
Checksum	-1
Lines	1
Program type	None

The 'Setup' sub-window shows the following settings:

- Settings**:  Skip initialization(advanced)
- Transfer settings**:
  - Program + source + Remarks
  - Program + source
  - Program only

Additional descriptions for menu options:

- Program + Source**: Shows the memory usage if the program (compiled) + source program and remarks is downloaded into the module.
- Program + Source - REM**: Same as above but without remarks.
- Program only**: Same as above but without source program and remarks.
- Checksum**: Shows the checksum of the complete program downloaded into the module. The checksum is unique and can be used to verify whether the program in the module matches the original program or not.
- Lines**: The number of program lines used in the source program (MacTalk)
- Mode**: Specify the program type actually used.
- Skip initialization (advanced)**: Bypasses internal initialization routines after powerup. (Only for very special use).
- Program + Source + Remarks**: Default. Choosing this will transfer everything down into the module memory. This can be an advantage if remarks and source program must be uploaded to MacTalk later.
- Program + Source**: Same as above but without remarks.
- Program only**: Only the compiled program is transferred.

TT2173GB

## 9.4

# How to build a program

When choosing New program in the Programming menu or entering MacTalk for the first time, programming can be started. Press the button at line 1 and a tool box will pop up.

**1** Press the first button to create the first program line. The "Select command" box will pop up.

**2** Choose the desired command. In this example it is desired to wait for an input to be activated before further program execution.

**3** Choose to wait until input 5 is high and press OK

**4** The command is inserted at the previous selected program line

TT0983GB

Continued

# 9.4

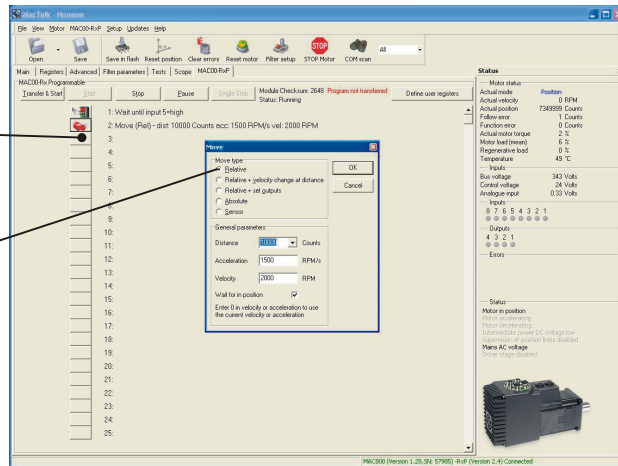
# How to build a program

5

Press the second button to create the second program line

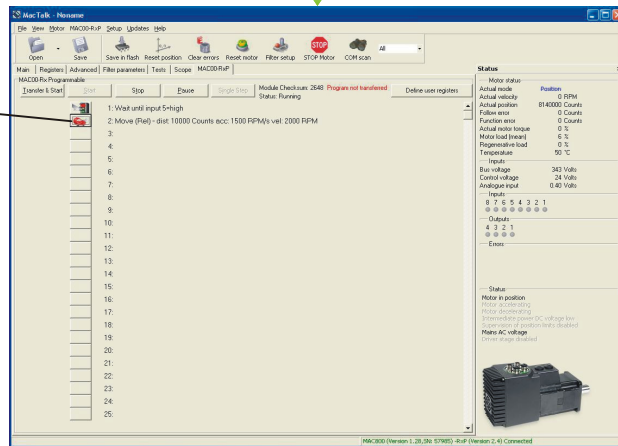
6

Choose the movement type needed.  
Relative: Move x counts forward with reference to the actual position.  
Absolute: Move to the x position with reference to the zero search position.



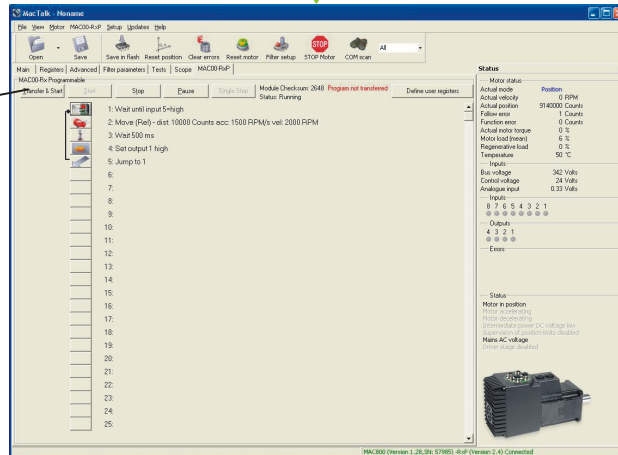
7

The relative move command just entered is converted into a program line.



8

Multiple program lines are entered by the user forming the last part of the program.



9

Now the program is finished. Press the "Transfer & Start" button. Now the program will be transferred and stored permanently in the module. The program will be executed immediately

TT0984GB

Continued

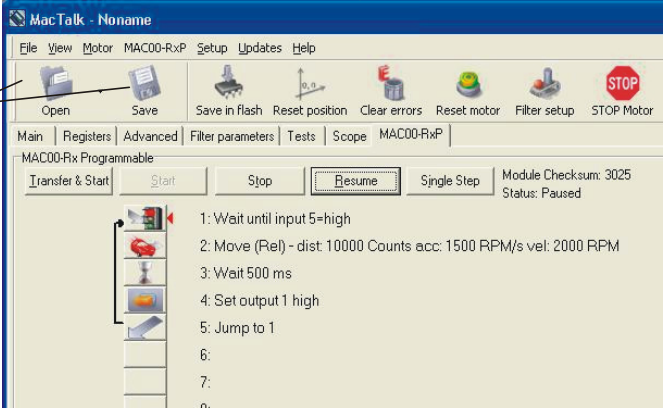
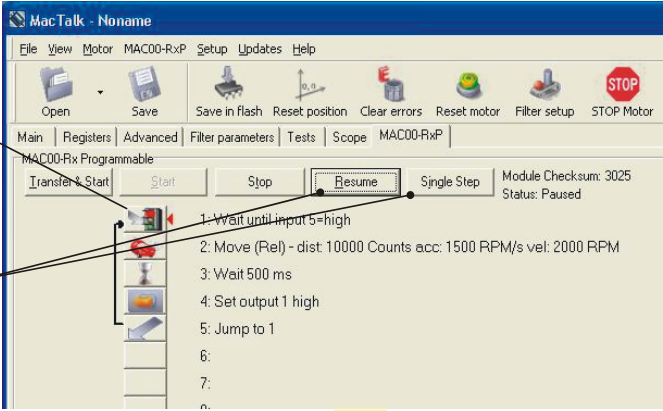
## 9.4

# How to build a program

**10** Now the program is running continuously. The actual program line which is executed is shown by the small red arrow.

**11** By choosing the "Pause" button, the program is paused. After it is paused, it is possible to single step through each program line which can be a useful feature to debug the program since the action in each line can be closely observed.

**12** When the program is finished, it can be saved on the harddisc or floppy disc. Please be aware that when saving the program it is the complete program including the overall setup of the motor such as servofilter, I/O setup etc. Everything is stored in a file with the extension .MAC. Later it can be opened and restored in the motor.



TT0985GB

## 9.5 General programming hints

---

When programming and saving programs the following hints may be useful to ensure that the program behaves as expected.

1. When transferring the program to the motor, it is saved permanently in memory and the program will be executed each time the motor is switched on.
2. Before beginning to program, ensure that the basic parameters for controlling acceleration, torque, safety limits, etc. are set to proper values. When saving the program to the PC, all of these basic parameter settings will be saved together with the program as a complete motor setup package.
3. A program line can be edited by double-clicking on the command text.
4. When the cursor is placed on top of the command icon, an edit menu will be shown by right-clicking.



## 9.6 Command toolbox description

The toolbox used for programming covers 18 different command types.

The basic idea of the commands is to provide easy access to the most common functions of the motor. Some functions may seem to be missing at first glance, but the buttons “Set register in the QuickStep motor” or “Wait for a register value before continuing” give direct access to all the 50 registers in the basic QuickStep motor, such as the gear ratio or the actual torque register.

In total, this gives a very powerful programming tool since >95% of a typical program can be built using the simple command icons, while the remaining 5% is typically achieved by accessing the basic motor registers directly.

The following gives a short description of all 18 command icons.

Use: Initiates any motor movement relative or absolute.

Use: Unconditional jump from one program line to another.

Use: Inserts a delay in the program specified in milliseconds.

Use: Write a value to almost any register in the basic MAC/MIS motor.

Use: Wait for a certain state at one or more of the digital inputs.

Use: Initiates a zero search to a sensor or a torque (no sensor).

Use: Change mode and activate register using a single command.

Use: Inserts a remark/ Comment in the program source code.

Use: Set the motor in the desired mode such as position- or velocity mode.

Use: Set a certain state at one or multiple digital outputs.

Use: Conditional jump from one program line to another. Input dependent

Use: Wait for a certain state at one or multiple digital inputs.

Use: Conditional jump from one program line to another. Register dependent

Use: Save the actual motor position to an intermediate register.

Use: Preset the position counter to a certain value.

Use: Binary format instead of graphic commands.


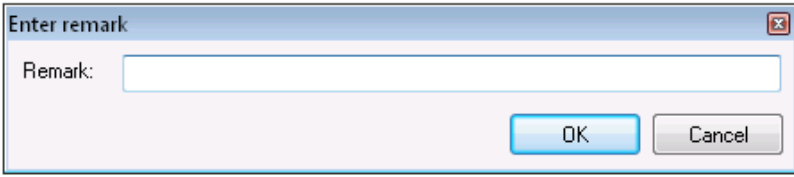
Use: Performs a calculation using register values and constants.

Use: Compares two registers to each other before jumping or moving in the program.


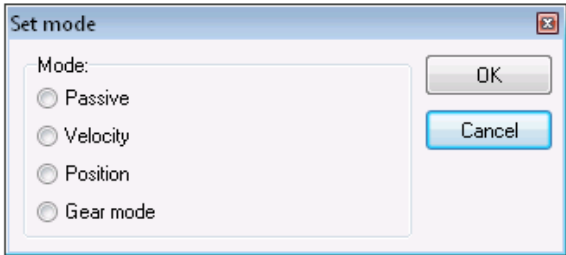
TT1103-02GB

## 9.7 Graphic programming command reference


### 9.7.1 Enter your own remarks

Icon:	
Dialogue:	
Function:	Inserts a remark/comment in the source code. The program line will not do anything, but can make the source code easier to read. This can be very important if other programmers have to review or work on the code, or if the program is only worked on infrequently.

### 9.7.2 Set operation mode


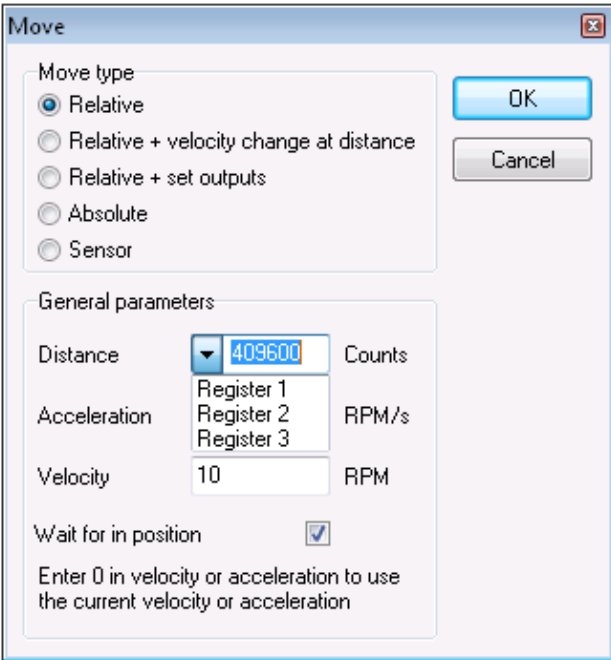
Icon:	
Dialogue:	
Function:	Sets the operating mode of the motor. When the program encounters a program line with this command, the motor's operating mode will be set to the specified mode. This allows you to use different operating modes in different parts of the program. For a detailed description of the individual operating modes, refer to section 1.3.1., <a href="#">Basic modes/functions in the QuickStep motor</a> , page 9.

### 9.7.3 Move operations

Icon:	
Function:	The Move commands are very flexible, with five different operating modes. Each mode is described in its own section below.


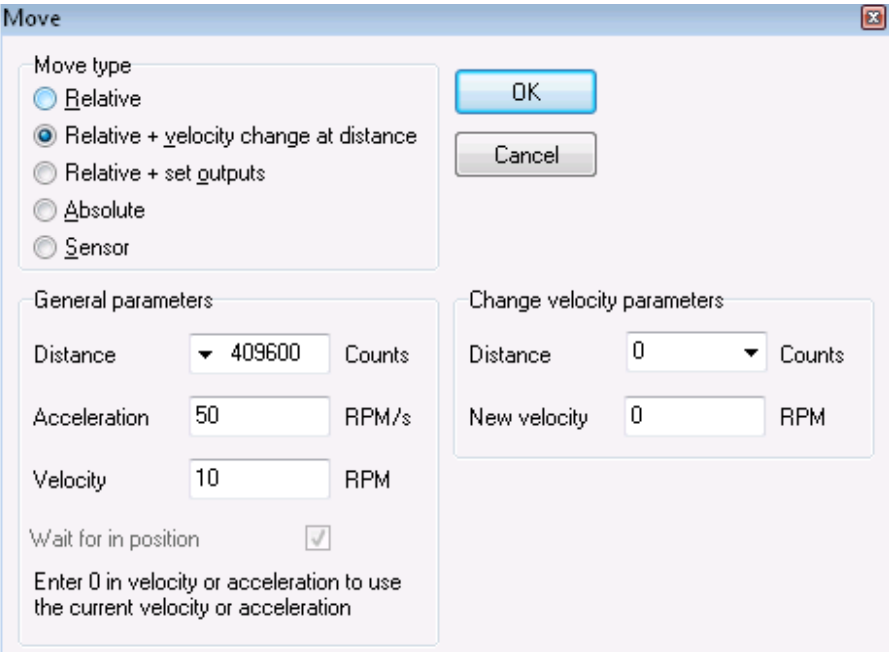
## 9.7 Graphic programming command reference

### 9.7.4 Move (Relative)

Icon:	
Dialogue:	
Function:	<p>Performs a movement relative to the current position. The distance moved is measured in encoder counts, and can either be entered directly or taken from three registers in the user memory area. For further information on using these memory registers, refer to the sections on the 'Save position' and 'Set position' commands.</p> <p>Note that if you specify a velocity, motor register no. 5 (V_SOLL) will be overwritten with this velocity value. Also, if you specify an acceleration, motor register no. 6 (A_SOLL) will be overwritten with the acceleration value specified. Register no. 49 (PI) is always overwritten by this command.</p> <p>If the 'Wait for in position' option is checked, the program will wait until the motor has finished the movement, before proceeding to the next program line. If this option is not checked, the program will start the movement, then immediately start executing the next command. The motor will finish the movement on its own, unless given other instructions by the program.</p>


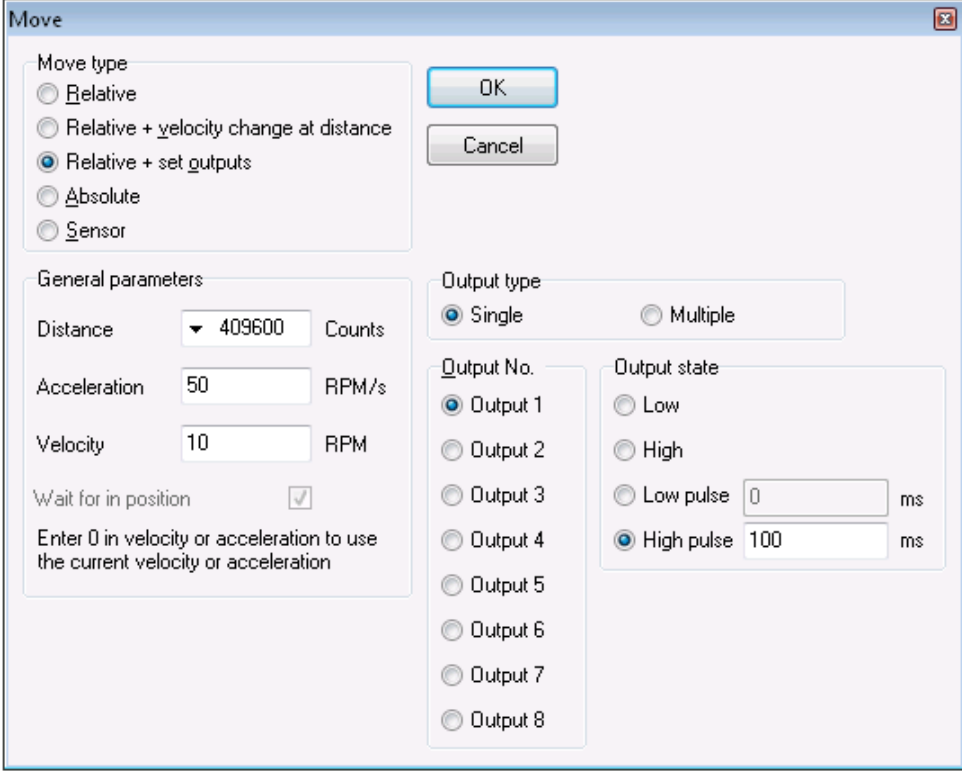
## 9.7 Graphic programming command reference

### 9.7.5 Move (Relative + velocity change at a distance)

Icon:	
Dialogue:	
Function:	<p>Performs a relative movement, and changes velocity at a specified distance before reaching the new position. The distances are measured in encoder counts and can either be entered directly, or taken from three memory registers in the RxP module. For further information on using these memory registers, refer to the sections on the 'Save position' and 'Set position' commands.</p> <p>Note that motor register no. 5 (V_SOLL) will always be overwritten with the value specified in the 'New velocity' field. Also, if you specify an acceleration, motor register no. 6 (A_SOLL) will be overwritten with the acceleration value specified. Register no. 49 (PI) is always overwritten by this command.</p> <p>This command always waits until the movement is finished, before proceeding to the next line in the program.</p>


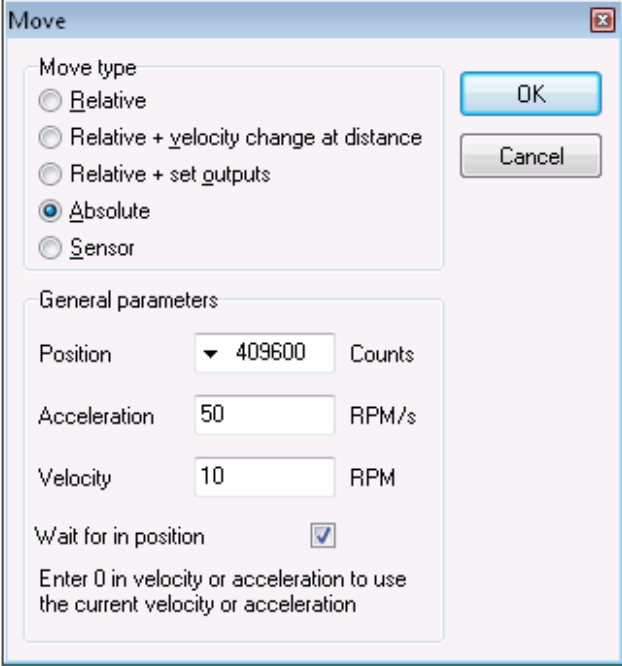
## 9.7 Graphic programming command reference

### 9.7.6 Move (Relative + set outputs)

Icon:	
Dialogue:	
Function:	<p>Performs a movement relative to the current position, and sets one or more outputs when the operation is completed. The distance moved is given in encoder counts and can either be entered directly, or can be taken from one of three memory registers in the user memory area. For further information on using these memory registers, refer to the sections on the 'Save position' and 'Set position' commands.</p> <p>Note that if you specify a velocity, motor register no. 5 (V_SOLL) will be overwritten with this velocity value. Also, if you specify an acceleration, motor register no. 6 (A_SOLL) will be overwritten with the acceleration value specified. Register no. 49 (PI) is always overwritten by this command.</p> <p>This command always waits until the movement is finished, before proceeding to the next line in the program.</p>


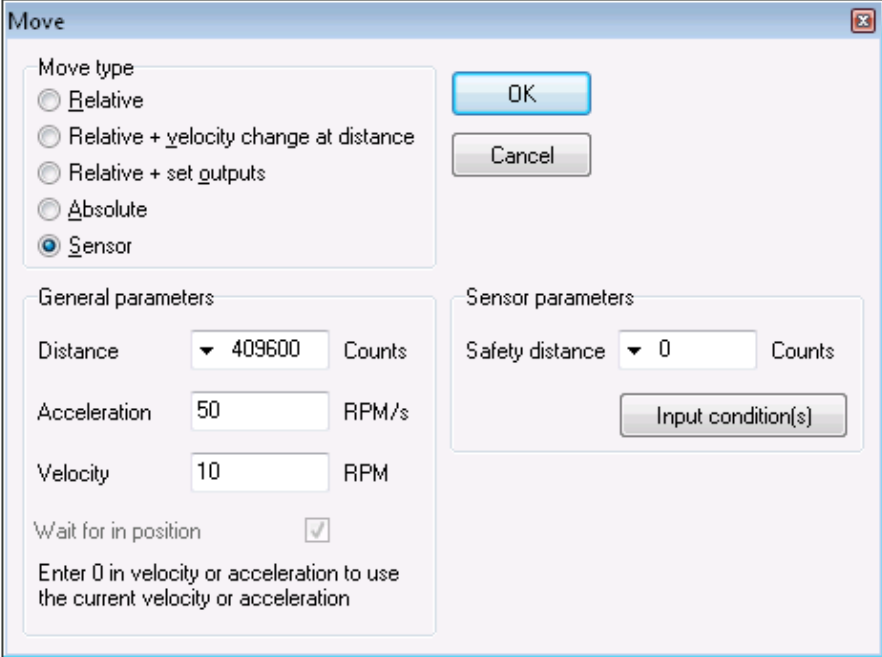
## 9.7 Graphic programming command reference

### 9.7.7 Move (Absolute)

Icon:	
Dialogue:	
Function:	<p>Moves to an absolute, non-relative position. The position is given in encoder counts and can either be entered directly, or can be taken from one of three memory registers in the user memory area. For further information on using these memory registers, refer to the sections on the 'Save position' and 'Set position' commands.</p> <p>Note that if you specify a velocity, motor register no. 5 (V_SOLL) will be overwritten with this velocity value. Also, if you specify an acceleration, motor register no. 6 (A_SOLL) will be overwritten with the acceleration value specified.</p> <p>If the 'Wait for in position' option is checked, the program will wait until the motor has finished the movement before proceeding to the next program line. If this option is not checked, the program will start the movement, then immediately start executing the next command. The motor will finish the movement on its own, unless given other instructions by the program.</p>


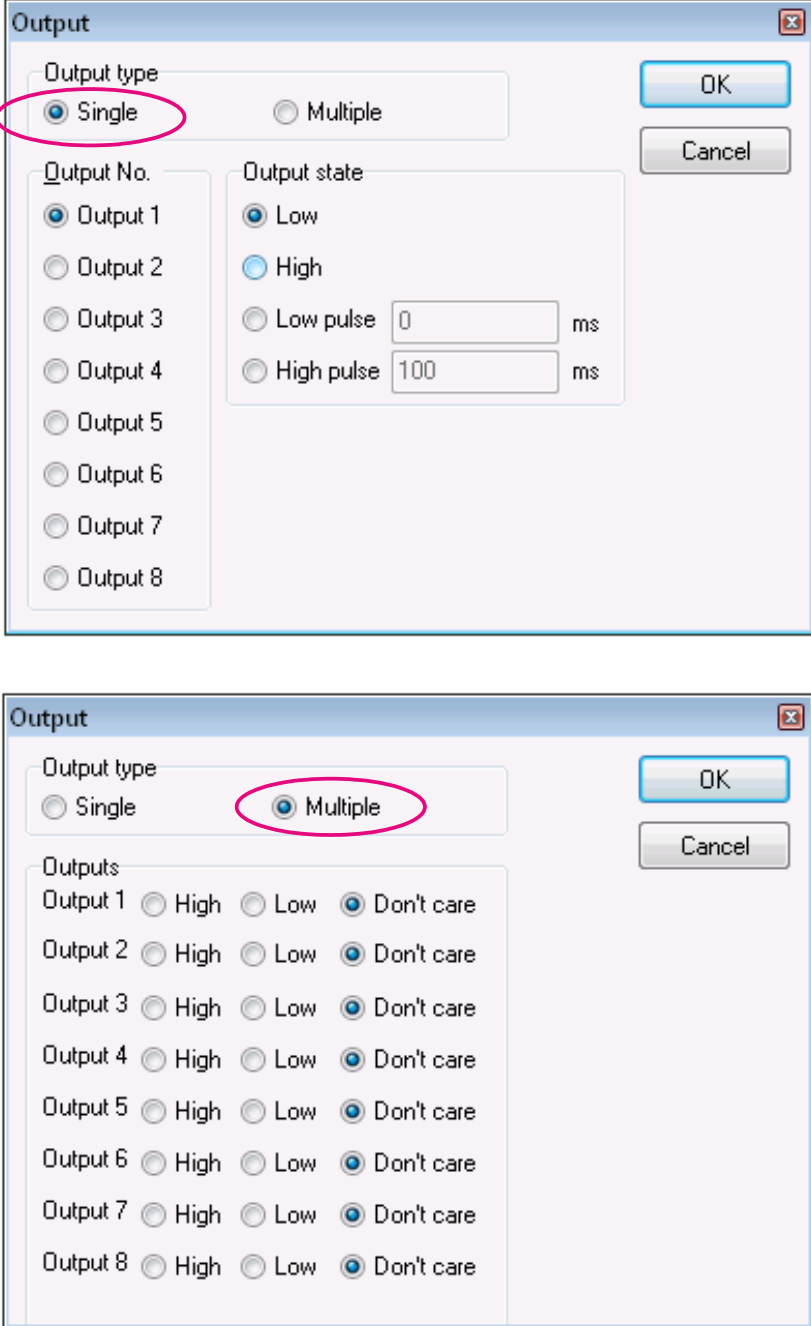
## 9.7 Graphic programming command reference

### 9.7.8 Move (Sensor)

Icon:	
Dialogue:	
Function:	<p>Performs a movement in the direction specified until an input condition is satisfied. The motor then moves the distance specified before stopping. The motor will not move farther than the Safety distance specified, regardless of whether the input condition is satisfied. The distances are measured in encoder counts and can either be entered directly, or taken from three memory registers in the user memory area. For further information on using these memory registers, refer to the sections on the 'Save position' and 'Set position' commands.</p> <p>Note that if you specify a velocity, motor register no. 5 (V_SOLL) will be overwritten with this velocity value. Also, if you specify an acceleration, motor register no. 6 (A_SOLL) will be overwritten with the acceleration value specified. Register no. 49 (PI) is always overwritten by this command.</p> <p>This command always waits until the movement is finished before proceeding to the next line in the program.</p>

## 9.7 Graphic programming command reference


### 9.7.9 Set outputs

Icon:	
Dialogue:	 <p>The first screenshot shows the 'Output' dialog box with 'Single' selected under 'Output type'. Under 'Output state', 'Low' is selected. There are input fields for 'Low pulse' (0 ms) and 'High pulse' (100 ms). The second screenshot shows 'Multiple' selected under 'Output type'. Under 'Outputs', 'Don't care' is selected for all outputs from Output 1 to Output 8.</p>
Function:	Sets one or more outputs. When setting a single output, you can set it to high, low, or you can specify the length (in milliseconds) of a pulse to send out on that output. When setting multiple outputs, you can specify whether to set each output high, low, or leave it in its current state.


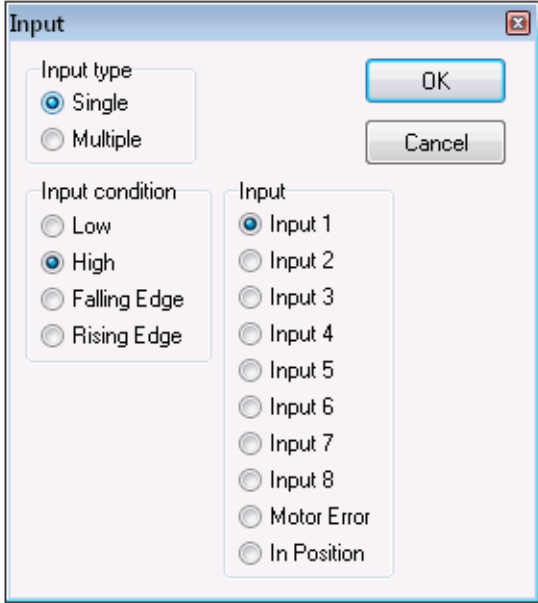


## 9.7 Graphic programming command reference

### 9.7.10 Unconditional jump


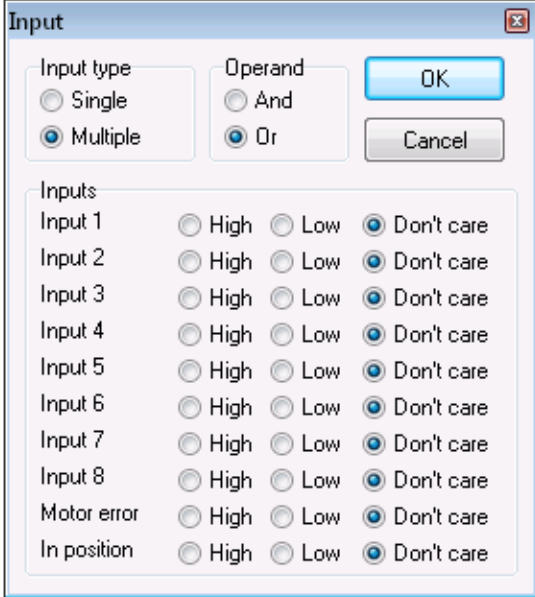
Icon:	
Dialogue:	None. After selecting this command, the mouse cursor changes. The next program line that you click on will become the destination for the jump.
Function:	Jumps to another line in the program.

### 9.7.11 Conditional jump (single input)

Icon:	
Dialogue:	
Function:	<p>Tests for an input condition before either jumping to another line in the program or moving on to the next line in the program. If the condition is met, the command jumps to the specified program line. If the condition is not met, the program proceeds to execute the next line in the program.</p> <p>When 'Input type' is set to 'Single', the command can test a single input for one of four possible conditions: the input is low, the input is high, the input has transitioned to low (Falling Edge), or the input has transitioned to high (Rising Edge). If transitions are tested for, the transition must have taken place during the last 30 microseconds.</p> <p>After pressing the OK button, the dialogue will disappear, and the mouse cursor will change. The next program line that you click on will then become the destination of the jump command.</p>


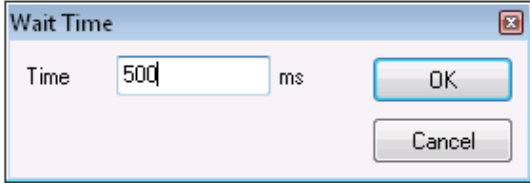
## 9.7 Graphic programming command reference

### 9.7.12 Conditional jump (multiple inputs)


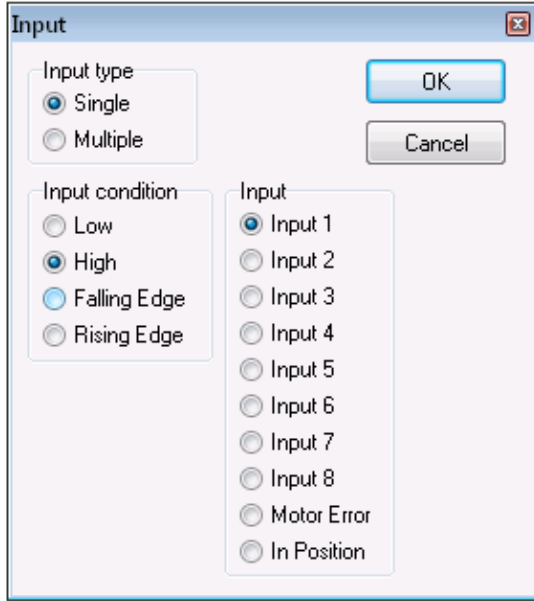
Icon:	
Dialogue:	
Function:	<p>Tests for an input condition before either jumping to another line in the program or moving on to the next line in the program. If the condition is met, the command jumps to the specified program line. If the condition is not met, the program proceeds to execute the next line in the program.</p> <p>When 'Input type' is set to 'Multiple', multiple inputs can be tested for being either high or low. The 'Operand' setting determines whether one or all of the inputs must meet their test criterion. If set to 'And', all inputs must match their test settings. If set to 'Or', only one input need match its test setting. Inputs that are set to 'Don't care' are not tested.</p> <p>After pressing the OK button, the dialogue will disappear, and the mouse cursor will change. The next program line that you click on will then become the destination of the jump command.</p>

## 9.7 Graphic programming command reference

### 9.7.13 Wait for (x) ms before continuing


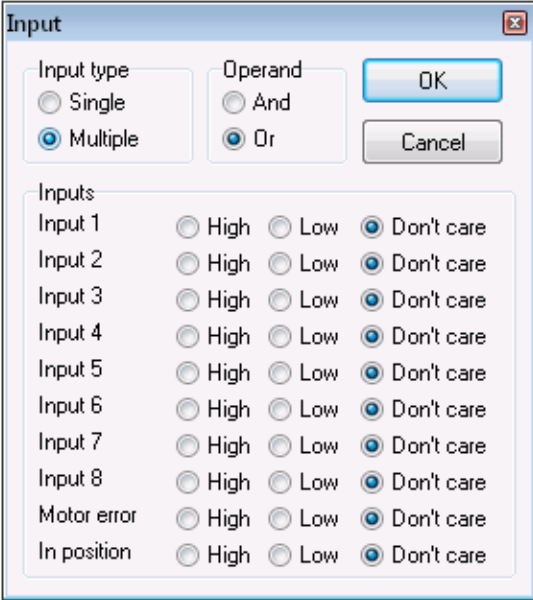
Icon:	
Dialogue:	
Function:	<p>Causes the program to pause for a number of milliseconds before continuing. The maximum pause that can be specified is 32767 milliseconds. The minimum pause that can be specified is 0 milliseconds.</p> <p>Note that this command overwrites Timer 1 in the RxP module's memory.</p>

### 9.7.14 Wait for an input combination before continuing (single input)

Icon:	
Dialogue:	
Function:	<p>Waits for a specified input condition to occur. The next line in the program will not be executed until the input condition has been met.</p> <p>If 'Input type' is set to 'Single', the command will wait for one of four things to happen on the specified input: that the input tests as high, that the input tests as low, that the input transitions from high to low (Falling Edge), or that the input transitions from low to high (Rising Edge). The input is tested with 30 microsecond intervals.</p>


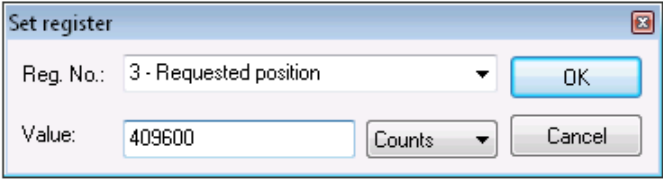
## 9.7 Graphic programming command reference

### 9.7.15 Wait for an input combination before continuing (multiple inputs)


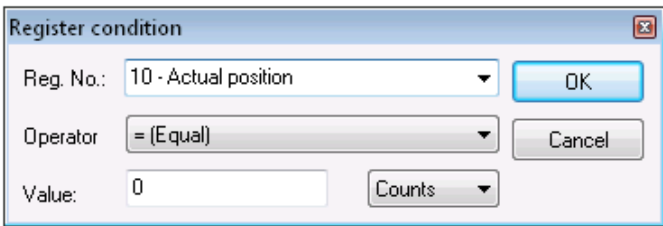
Icon:	
Dialogue:	
Function:	<p>Waits for a specified input condition to occur. The next line in the program will not be executed until the input condition has been met.</p> <p>If 'Input type' is set to 'Multiple', multiple inputs can be tested for being either high or low. The 'Operand' setting determines whether one or all of the inputs must meet their test criterion. If set to 'And', all inputs must match their test settings. If set to 'Or', only one input need match its test setting. Inputs that are set to 'Don't care' are not tested. The inputs are tested with 30 microsecond intervals.</p>

## 9.7 Graphic programming command reference

### 9.7.16 Set a register in the MIS motor


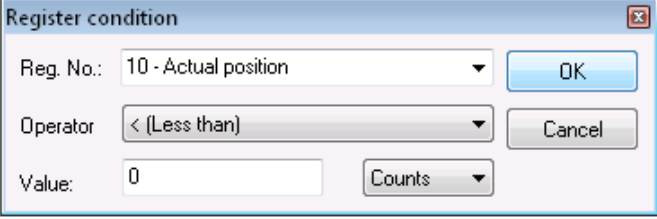
Icon:	
Dialogue:	
Function:	<p>Sets a register in the motor to a specified value. The register is selected from a list of known, user-accessible registers. The value can either be entered as native motor units or it can be entered as generic engineering units.</p> <p>The dialogue above provides an example: register no. 3 (P_SOLL, or Requested position, depending on your preference) can either be set to an integer number of encoder counts, or it can be set to a non-integer number of revolutions.</p>

### 9.7.17 Jump according to a register in the MAC motor


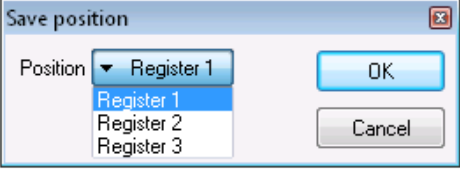
Icon:	
Dialogue:	
Function:	<p>Tests a register in the motor against a specified value before either jumping to another line in the program or moving on to the next line in the program. If the condition is met, the command jumps to the specified program line. If the condition is not met, the program proceeds to execute the next line in the program. The value can either be entered as native motor units, or it can be entered as generic engineering units. The dialogue above provides an example: register no. 10 (P_IST, or Actual position, depending on your preference) must be equal to 0 revolutions if the jump is to be executed. The position that the register is tested against can be specified as an integer number of encoder counts or can be specified as a non-integer number of revolutions.</p> <p>After pressing the OK button, the dialogue will disappear and the mouse cursor will change. The next program line that you click on will then become the destination of the jump command.</p>

## 9.7 Graphic programming command reference

### 9.7.18 Wait for a register value before continuing


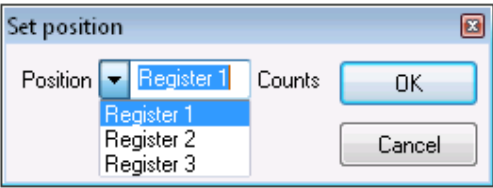
Icon:	
Dialogue:	
Function:	<p>Tests a register in the motor against a specified value and waits until the specified condition is met. The value can either be entered as native motor units or can be entered as generic engineering units.</p> <p>The dialogue above provides an example: register no. 10 (P_IST, or Actual position, depending on your preference) must be less than 0 revolutions, before the program will continue. The position that the register is tested against can be specified as an integer number of encoder counts, or can be specified as a non-integer number of revolutions.</p>

### 9.7.19 Save position


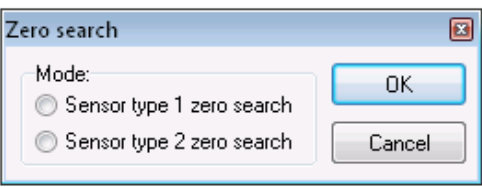
Icon:	
Dialogue:	
Function:	<p>Saves the current position from register no. 10 (P_IST) to one of three locations in the user memory area. The saved position(s) can then be used whenever a position or distance is needed in a move command.</p>

## 9.7 Graphic programming command reference

### 9.7.20 Set position


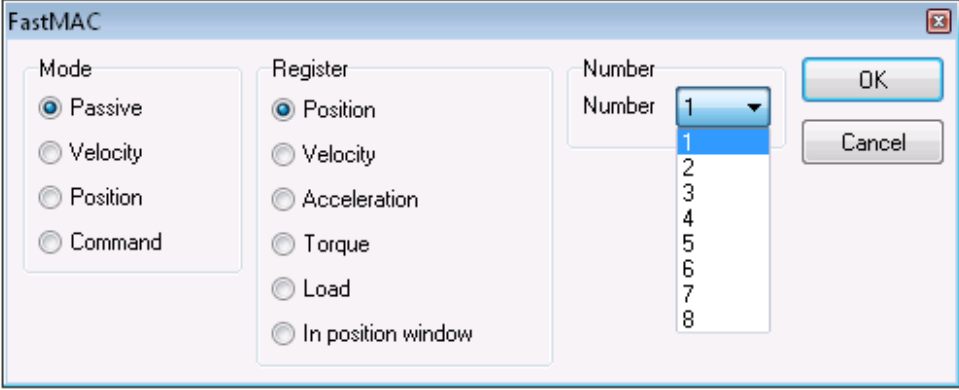
Icon:	
Dialogue:	
Function:	Sets the current position stored in register no. 10 (P_IST) to one of three position values stored in the user memory area. This is the reverse of the 'Save position' command.

### 9.7.21 Zero search

Icon:	
Dialogue:	
Function:	Initiates a zero search. The program waits until the zero search has completed before proceeding to the next command. For a detailed description of how to set up a zero search, refer to <a href="#">Zero search modes</a> , page 125

## 9.7 Graphic programming command reference


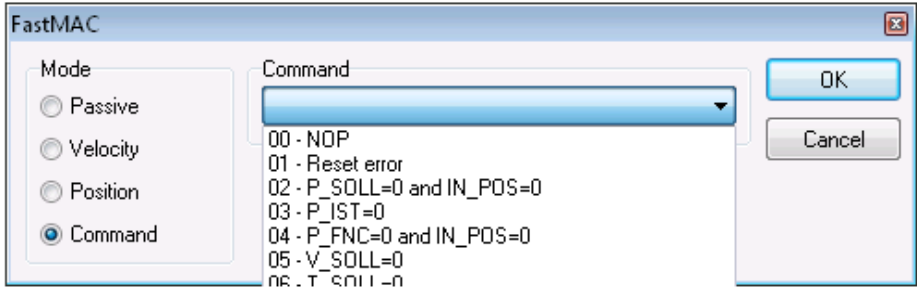
### 9.7.22 Send FastMAC command (change mode and activate register)

Icon:	
Dialogue:	
Function:	<p>FastMAC commands are also sometimes referred to as FlexMAC commands. The advantage of these commands is a very low communication overhead. FastMAC/FlexMAC commands are described in detail in section 4.5.7 of the MAC user manual, JVL publication no. LB0047-20GB. However, a brief summary is in order.</p> <p>If 'Mode' is set to 'Passive', 'Velocity', or 'Position', the motor will switch to that mode. Also, one of the passive motor registers will be activated, in the sense that its value will be written to the corresponding active motor register, which actually controls motor behaviour. In the example above, the value in register no. 65 (VI) will be written to register no. 5 (V_SOLL). Move operations will then take place at that velocity.</p>


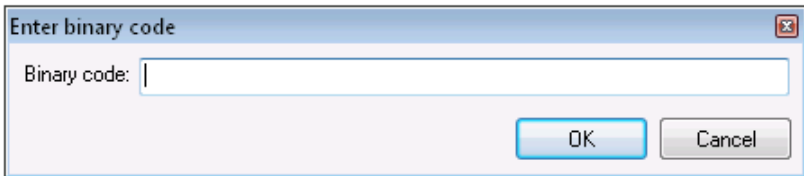


## 9.7 Graphic programming command reference

### 9.7.23 Send FastMAC command (macro command)


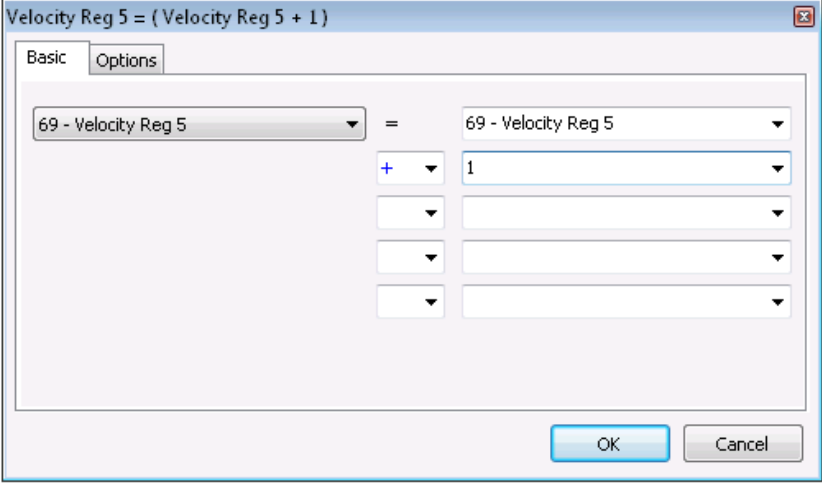
Icon:	
Dialogue:	
Function:	<p>If 'Mode' is set to 'Command', the motor does not necessarily change mode but it can be commanded to carry out a series of predetermined operations. Describing all of the FastMAC commands is beyond the scope of this section but for example, using a single command it is possible to activate four different sets of registers, each controlling position, velocity, acceleration, torque, load factor, and in-position window. For further details, refer to section 4.5.7 of the MAC user manual.</p>

### 9.7.24 Binary command

Icon:	
Dialogue:	
Function:	<p>MacTalk programs are sent to the motor in a compact, binary format, which is then interpreted by the motor's firmware. The existing set of graphic commands covers most situations, but when special needs arise, anything that can be done with programs can be done with a binary command. If special needs arise that are not covered by the other commands, contact JVL for assistance.</p>


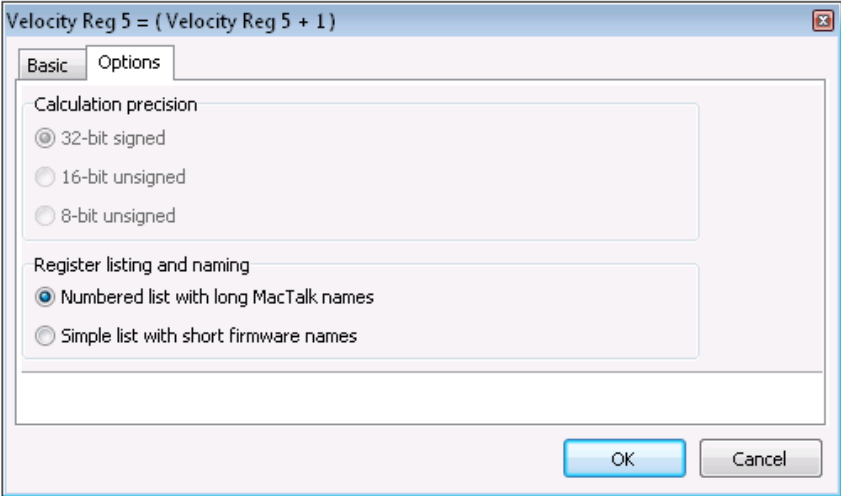
## 9.7 Graphic programming command reference

### 9.7.25 Calculator (basic)

Icon:	
Dialogue:	
Function:	<p>Performs a calculation using register values, constants, and the four basic arithmetic operations: +, -, * and /. The result is stored in a register. Arithmetic operations take place in the order that they are specified. Operands/arguments can be either integer constants or registers. The caption of the dialogue box shows the resulting expression in traditional infix format. It is continuously updated as you type in the expression.</p> <p>Note that if you write a value to a register using this command, that value is always measured in native motor units. Conversion from generic engineering units is only supported for the commands 'Set a register', 'Jump according to a register', and 'Wait for a register value before continuing'.</p>


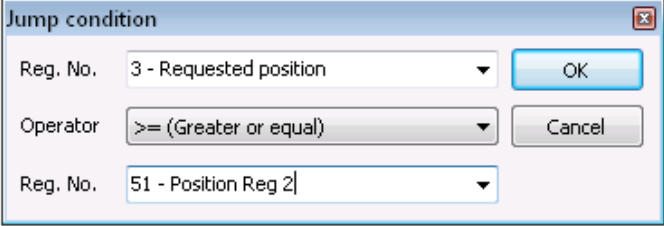
## 9.7 Graphic programming command reference

### 9.7.26 Calculator (options)

Icon:	
Dialogue:	
Function:	<p>The options tab contains various settings that affect the operation of the Calculator command. 'Calculation precision' is currently preset to 32-bit precision and cannot be changed. This is not an error, and should not be reported.</p> <p>'Register listing and naming' provides an alternative method of entering data into the dialogue by selecting 'Simple list with short firmware names'. Instead of selecting, for example, '3 – Requested position' to access register no. 3, you can simply type 'P_SOLL'. If you wish to enter a constant, you simply enter the digits – the dialogue will not mistake the constant for a register number.</p> <p>If you are in doubt about a register name, look at the expression in the caption of the dialogue box. A recognized register name will appear in the expression. An unrecognizable register name will appear as a zero. You can switch between the two methods of data entry at any time.</p>

## 9.7 Graphic programming command reference



















### 9.7.27 Jump according to a comparison

Icon:	
Dialogue:	
Function:	<p>Compares two registers with each other before either jumping to another line in the program or moving on to the next line in the program. If the condition is met, the command jumps to the specified program line. If the condition is not met, the program proceeds to execute the next line in the program.</p> <p>Any two registers can be compared with each other but the command does not do anything beyond comparing the registers numerical values measured in native motor units. To ensure that comparisons are meaningful, it is preferable to compare registers that hold the same type of information in the same binary format.</p> <p>In the example above, two position registers are compared. Both hold position information, both are 32-bit wide, and both measure position in encoder counts. Such a comparison will always yield meaningful, predictable results.</p> <p>For other types of registers, see the relevant register sections.</p>

## 9.8

## Command timing

Each command has a certain execution time. The specified execution time in the following table is the maximum execution time if not using CANopen, serial communication and the motor is disabled. The actual execution may be faster.

Icon	Name	Execution time [ $\mu$ s]
	Remarks	0
	Set operation mode	60
	Move relative (no velocity, no acceleration) <sup>1</sup>	90
	Move relative + set velocity (no acceleration) <sup>1</sup>	150
	Move relative + set velocity + set acceleration <sup>1</sup>	210
	Move absolute (no velocity, no acceleration) <sup>1</sup>	60
	Move absolute + set velocity (no acceleration) <sup>1</sup>	120
	Move absolute + set velocity + set acceleration <sup>1</sup>	180
	Set single output (high/low)	30
	Set multiple outputs	30*number of outputs
	Unconditional jump	30
	Conditional jump (inputs)	60
	Set a register	60
	Conditional jump (register)	120
	Save position	60
	Set position	90
	Send fastMAC command	30
	Binary command	30

1) The time for all move commands is shown without waiting for in position

## 9.9 More about program timing

---

The firmware is structured so that one program instruction is executed for each pass of the main loop, which takes approximately 30 microseconds ( $\mu\text{s}$ ) without CANopen, without serial communications and when the motor is not running. The Main Loop Time is termed MLT in the following text.

A single program line in MacTalk can generate more than one instruction. For example, assigning a constant value to a register uses two instructions: First load the value to the internal stack and then Store from the stack to the target register. The above table in [Motor Connections](#), page 302 reflects this operation.

The main loop time will vary depending on a number of factors: The serial communications speed and load, whether CANopen is installed, and the CANopen communications speed and load.

Serial communications on the RS-485 line can load the motor up to 1% at 19.200 baud, which is insignificant, but at the maximum baud rate of 921.600 the communications can load the motor up to 45%, which would result in an MLT of  $\sim 60 \mu\text{s}$ .

When CANopen firmware is installed, the basic MLT will change from 30 to 90  $\mu\text{s}$  with no communications.

When loading the CANbus with communications, the MLT can rise significantly. For example, when using seven transmit PDOs with an event timer value of 1 ms and a CANbus link speed of 500 kbits/s, the MLT can rise to 150-200  $\mu\text{s}$ . Also using RS-485 communications at high baud rates can result in even longer MLT values. However, this scenario is very unlikely.

**Note: In applications where program timing is critical, tests must be performed to ensure that timing is satisfactory when communication is running according to conditions used in production!**

## 10 Ethernet protocols (optional)

---

The MIS motors offers optional 6 different Ethernet protocols.  
These are:

- **EtherCAT**
- **ModbusTCP**
- **Profinet**
- **Powerlink**
- **EthernetIP**
- **SercosIII**

This manual do only cover description of how to connect.  
Concerning software and protocol setup and usage please consult a separate manual that can be found at [www.jvl.dk](http://www.jvl.dk) using this link: [www.jvl.dk](http://www.jvl.dk)





This chapter covers the JVL Stepper motor controllers SMC66 and SMC85 which are used with the MIS17x, 23x, 34x and 43x motor series on a CANopen network.

The chapter covers the following main topics:

- General introduction: a section with general information about CANopen. See [section 11.1.1](#) to [section 11.1.5](#).
- Setting up the Baud-rate, node-id and termination of the CAN bus. Covers also the wiring of the CAN bus. See [section 11.2.1](#) to [section 11.2.5](#).
- Using CANopenExplorer.  
See [section 11.3.1](#) to [section 11.3.3](#).
- Survey of Communication specific objects and manufacturer specific objects in the DS301 standard. Communication objects consist of the general information about the settings in the module, while the Manufacturer specific objects consist of the settings of input/output and the motor parameters. This section also covers the settings of the transmit and receive PDOs in the module. See [section 11.4.1](#) to [section 11.4.7](#).
- Survey of objects which are used in the DSP-402 standard. See [section 11.5.1](#) to [section 11.5.7](#).
- Section with more detailed explanations of the CANopen theory, particularly DS-301.  
See [section 11.7.1](#) to [section 11.7.7](#).

# 11.1 General information about CANopen

---

## 11.1.1 Introduction

A CANopen option is available for all the MIS motors.

When this option is installed, the controllers include a CANopen slave. Through the CANopen slave, all the registers of the controller can be accessed. Both implement object dictionaries that follow the CiA DS-301 standard.

The controllers are designed to be used on a CANbus, CANopen DS-301 and CANopen DSP-402. Do not use the modules together with CANKingdom or DeviceNet.

The MIS motors supports dynamic mapping of all objects in both 16- and 32-bit. The data field length of a PDO can be between 1 and 8 bytes so the user can decide how to combine these objects to obtain the most efficient packed amount of information in order to decrease bus load. Please consult the section [Dynamic Mapping](#), page 245 to learn how to construct these PDO's. Also supports default mapping is supported.

Furthermore it is possible to implement a flexible register, which is fully user customizable so that the user can select single bits to monitor by a TxPDO. Read more about this in the section [Flexible Register setup](#), page 259.

## 11.1.2 CiA membership

CiA (CAN in Automation) is a non-profit society. The object of the society is to promote CAN (Controller-Area-Network) and to provide a path for future developments of the CAN protocol. CiA specifications cover physical layer definitions as well as application layer and device profile descriptions.

In order to receive the CAN standard, it is necessary to obtain CiA membership. The membership fee depends on a company's number of employees. Membership runs from January 1<sup>st</sup> until December 31<sup>st</sup> and is renewed automatically unless cancelled in writing by the end of a calendar year. Companies applying for membership after July 1<sup>st</sup> pay 50% of annual membership.

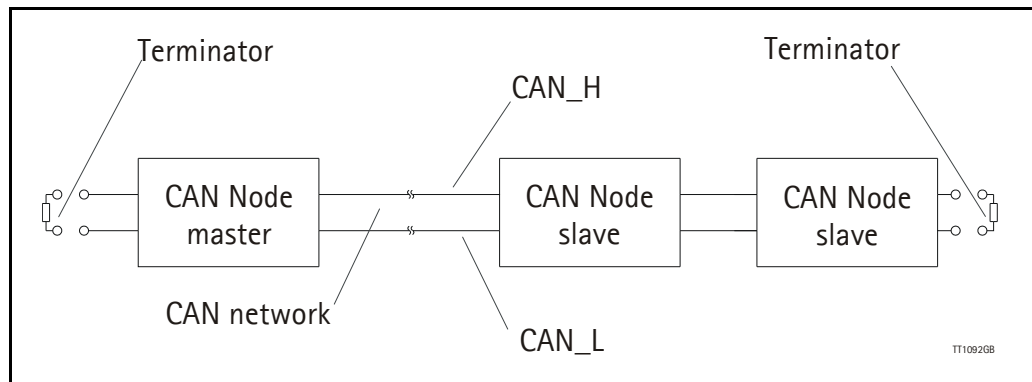
A PDF application form can be downloaded from <http://www.can-cia.org/cia/application.html>.

Note: Once you have received a license from CIA, standards will be sent on a CD and are downloadable via member login. All of the CiA specifications can be ordered from the following URL: [www.can-cia.org/downloads/ciaspecifications/](http://www.can-cia.org/downloads/ciaspecifications/)

## 11.1.3 CANopen network

The CAN bus is a serial bus with multi-master capabilities where different products from different manufacturers can communicate with each other. These include, for example, devices such as PLCs, motors, sensors and actuators. Some message types have higher priority and are sent first, for time-critical applications. New devices can easily be integrated on an existing bus, without the need to reconfigure the entire network. The devices are connected through a 2-wire bus cable with ground, and data is transmitted serially.

## 11.1 General information about CANopen



### 11.1.4 CANopen, general information

CANopen is a CAN-based, higher-level protocol. The purpose of CANopen is to give an understandable and unique behaviour on the CAN network. The CAN network is the hardware level of the system, and CANopen is the software level. CANopen is based on the communication profile described in CiA DS-301, and specifies all of the basic communication mechanisms.

CiA DS-301 contains message types on the lowest software level. The DSP-402 CANopen standard defines the device profile and the functional behaviour for servo drive controllers, frequency inverters and stepper motors. The DSP-402 constitutes a higher software level, and it uses the DS-301 communication, but makes the device independent of the manufacturer. Not all JVL functionality is available.

The CANbus with real-time capabilities works in accordance with the ISO 11898 standard. The major performance features and characteristic of the CANopen protocol are described below:

#### Message-oriented protocol:

The CANopen protocol does not exchange data by addressing the recipient of the message, but rather marks each transmitted message with a message identifier. All nodes in the network check the identifier when they receive a message to see whether it is relevant for them. Messages can therefore, be accepted by none, one, several or all participants.

#### Prioritisation of messages:

As the identifier in a message also determines its priority for accessing the bus, it is possible to specify a correspondingly rapid bus access for messages according to their importance. Especially important messages can thus gain access to the bus without a prolonged wait-time, regardless of the loading on the bus at any instant.

This characteristic means that important messages are transmitted with high priority even in exceptional situations, thereby ensuring proper functioning of a system even during phases of restricted transmission capacity.

# 11.1 General information about CANopen

### Multi-Master capability:

Bus access rights are not issued by a mean-level control unit (bus master) per network. Instead, each network node can start to send a message with equal rights as soon as the bus has become free. If several participants access the bus at the same time, an arbitration process allocates each participant the bus access right in line with the priority of the message they want to send at that particular moment. Each participant can therefore communicate directly with every other participant. As the transmission of a message can be initiated by the message source itself, then in the case of event-controlled transmission of messages, the bus is only occupied when a new message is on-hand.

### No-loss bus arbitration:

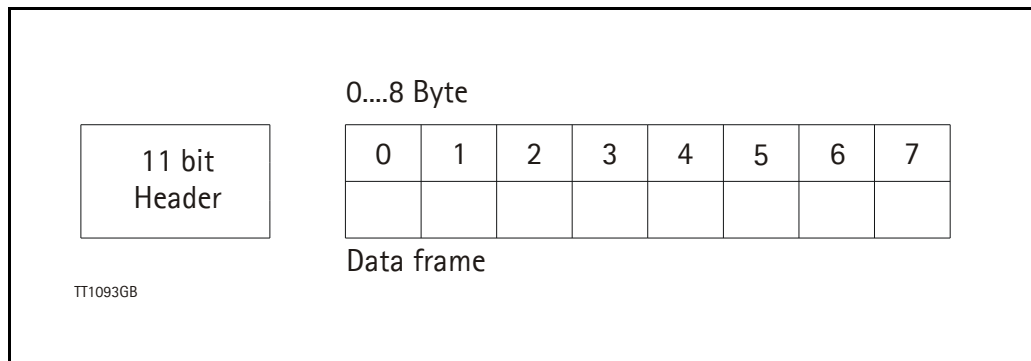
As the bus is accessed at random under the CANopen protocol, it is possible that several participants try to occupy the bus at the same time. In other random bus access routines, this causes the destruction of suppressed messages. In order to solve such a bus access conflict, a repeated occupation of the bus is required using an appropriate triggering strategy. The CANopen protocol therefore deploys a routine to ensure that the message with the highest priority at any given time is sent without any destruction of message contents.

### Short block length:

The maximum data length of a CAN message is limited to 8 bytes. This data length is usually sufficient to transmit the information occurring in the lowest field area in a CAN message.

## 11.1.5 Header

A CAN message transmits the communications object and a variety of management and control information. The management and control information bits are used to ensure error-free data transmission, and are automatically removed from the received message and inserted before a message is sent. A simplified CANopen message could be as in the figure below:



The two bit fields "Header" and "Data" form the simplified CANopen message. The 11-bit Header is also designated as the identifier or as the COB-ID (Communication Object identifier).

## 11.1 General information about CANopen

---

JVL uses the 11-bit format type CAN A, but not the 29-bit format type CAN B.

The COB-ID carries out two tasks for the controller communications object.

- Bus arbitration: Specification of transmission priorities.
- Identification of communications objects.

The COB-ID comprises two sections:

- Function code, 4 bits in size (0...15)
- Node address (Node ID), 7 bits in size (0...127).

The function code classifies the communications objects, and controls the transmission priorities. Objects with a small function code are transmitted with high priority. For example, in the case of simultaneous bus access an object with the function code "1" is sent before an object with the function code "3".

### **Node address:**

Every device is configured before network operation with a unique 7-bit long node address between 1 and 127. The device address "0" is reserved for broadcast transmissions, in which messages are sent simultaneously to all devices.

PDO, SDO, EMCY, NMT and heartbeat use the header frame for communication on the CANopen bus.

## 11.2 Connection and setup of the CAN bus

---

### 11.2.1 Connecting the motor to the CAN bus

Before you connect the motor to the CAN-bus, the Baud-rate, the Node-ID and the termination must be selected.

On the serial bus it is possible to set a transmission speed (Baud-rate) of max. 1000 Kbit/s and a min. of 10 Kbit/s. The Baud-rate depends on the cable length, and the wire cross-section. The table below gives some recommendations for networks with less than 64 nodes. Recommended bus cable cross-sections are according to CIA.

:

Bus Distance (m)	Cross-section (mm <sup>2</sup> )	Terminator (Ohms)	Baud-rate (Kbit/s)
25	0.25-0.34	120	1000
100	0.34-0.6	150-300	500
250	0.34-0.6	150-300	250
500	0.5-0.6	150-300	125
500	0.5-0.6	150-300	100
1000	0.75-0.8	150-300	50

The bus wires may be routed in parallel, twisted and/or shielded, depending on EMC requirements. The layout of the wiring should be as close as possible to a single line structure in order to minimize reflections. The cable stubs for connection of the bus node must be as short as possible, especially at high bit rates. The cable shielding in the housing must have a large contact area. For a drop cable, a wire cross-section of 0.25 to 0.34 mm<sup>2</sup> would be an appropriate choice in many cases.

For bus lengths greater than 1 km, a bridge or repeater device is recommended. Galvanic isolation between the bus nodes is optional.

### 11.2.2 Necessary accessories:

The EDS file for the MIS motors is available for download at JVL's web-site, <http://www.jvl.dk>, under the downloads menu, Field bus Interface Specifications Files. EDS means Electronic Data Sheet. This file contains the information about the motor settings that are required to configure the setup and program in the master. The MIS motor is a slave module on the CAN-bus. The master can, for example, be a PLC or a PC.

If you are using a PLC as master, then make sure it is provided with a CANopen communications module, and that the correct programming tools are available. For support of the PLC master, the PLC vendor is recommended.

If you are using a PC as master, JVL provides some tools that can help when installing and using the MIS motors.

## 11.2 Connection and setup of the CAN bus

---

The latest firmware for the MIS motors is available at JVL's web-site under the menu downloads/firmware. In the site's programs menu, the software CANopenExplorer is also available, but note that this is not a free-ware program. Please contact your JVL representative for further information.

CANopenExplorer can be used to load the EDS file and operate with the motor. The CANopenExplorer software must use a special dongle for communication with the PC. For further information about the dongle, see [An overall method for communication test](#), page 237. The PC must be provided with a CANopen communications module.

### 11.2.3 EDS (Electronic data Sheet)

In order to give the user of CANopen more support, the device description is available in a standardised way, and gives the opportunity to create standardised tools for configuration of CANopen devices, designing networks with CANopen devices, and managing project information on different platforms. The EDS file are ASCII-coded.

### 11.2.4 Setting the node id and baud rate

The node id is set using MacTalk. It is located in register 162. The baud rate is also set using MacTalk and is located in register 163.  
See also [Baud\\_Rate](#), page 179

## 11.2 Connection and setup of the CAN bus

---

### 11.2.5 Bus termination

In order to guarantee correct operation of the CAN bus, bus terminating resistors must be provided at both ends of the bus cable.

See the general connection guide for connecting CAN bus to the MIS motors.

[How to connect a MIS motor, page 34](#)



## 11.3 Using CANopenExplorer

---

### 11.3.1 The CANopenExplorer program

The CANopenExplorer is a program that was developed for internal use only, especially in production, but the program offers features that are very convenient and which make it very easy to start up the MIS motor when this is supplied with the CANopen option.

The program can write and send SDOs, PDOs, SYNC and heartbeat messages, and also can read EDS files.

### 11.3.2 An overall method for communication test

Depending on the type of master and software solution available, the following components must be available:

PLC: PLC with a CANopen module and software that can communicate with this module.  
The CANopen module must be connected to a CAN bus, as shown in [section 11.2.5](#). To set up the master, download the EDS file from the JVL web site (see [section 11.2.2](#)). This file contains all register set-up data for the MIS motors. For details of the node-ID and the Baud-rate, see [section 11.2.4](#). The power supply must be connected to the motor as shown in [How to connect a MIS motor](#), page 34.

PC: PC with a CAN adaptor and software that can communicate with this module, or if the CANopenExplorer software is used, the PCAN-USB Dongle from Peak-system that is connected to a USB port on the PC. The Peak systems web site address is <http://www.peak-system.com>. This includes a list of distributors. To set up the master, download the EDS file from the JVL web-page, see [section 11.2.2](#). This file contains all register set-up data for the MIS motors. For details of the node-ID and the Baud-rate, see [Setting the node id and baud rate](#), page 235. The power supply must be connected to the motor as shown in [How to connect a MIS motor](#), page 34.

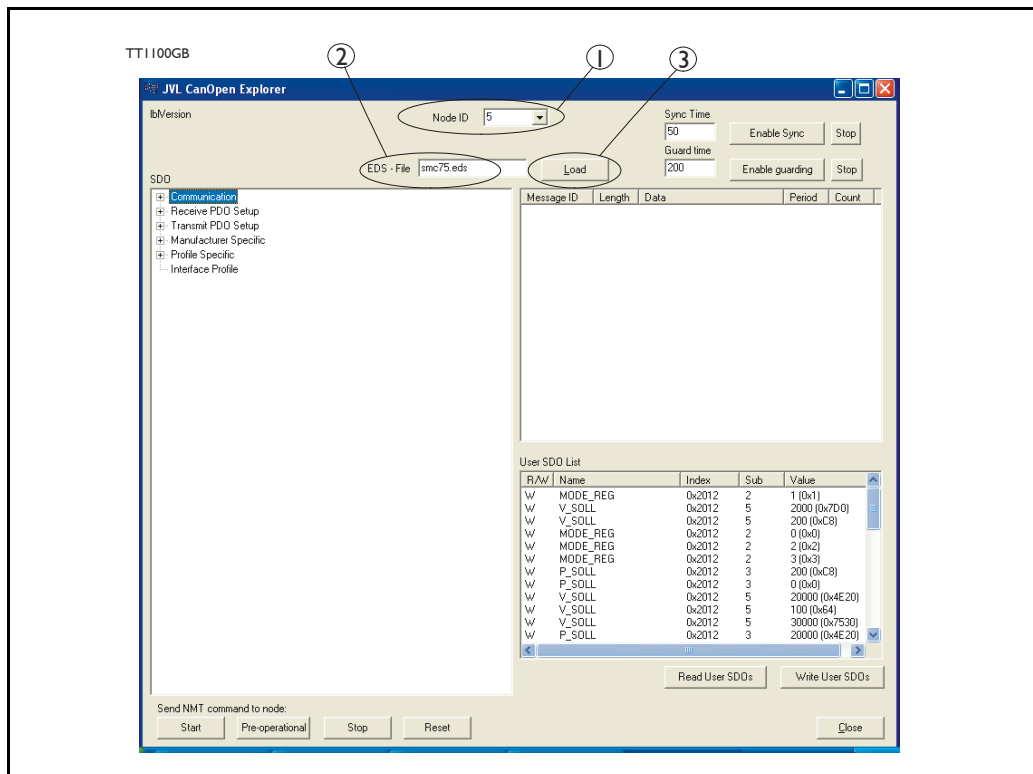
If CANopenExplorer is used, see the following method for testing the motor communication:

- Install CANopenExplorer
- Connect the motor to the USB port via the Dongle.
- Connect power supply, see [section How to connect a MIS motor](#), page 34.
- Run the CANopenExplorer program on the PC.

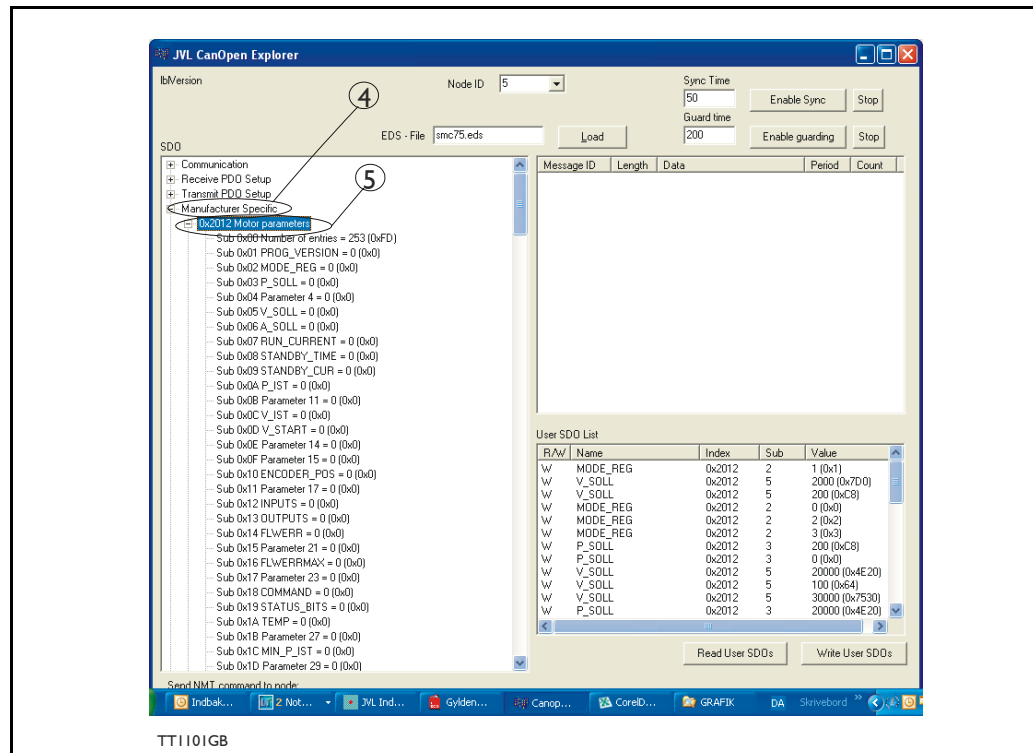
- 1: Select the correct node ID in the slave using MacTalk. See [Setting the node id and baud rate](#), page 235.
- 2: Select the EDS file. For all the MIS motors this file is SMC85\_V1\_00\_S.eds or newer.
- 3: Load the EDS file by pressing load.

# 11.3

# Using CANopenExplorer



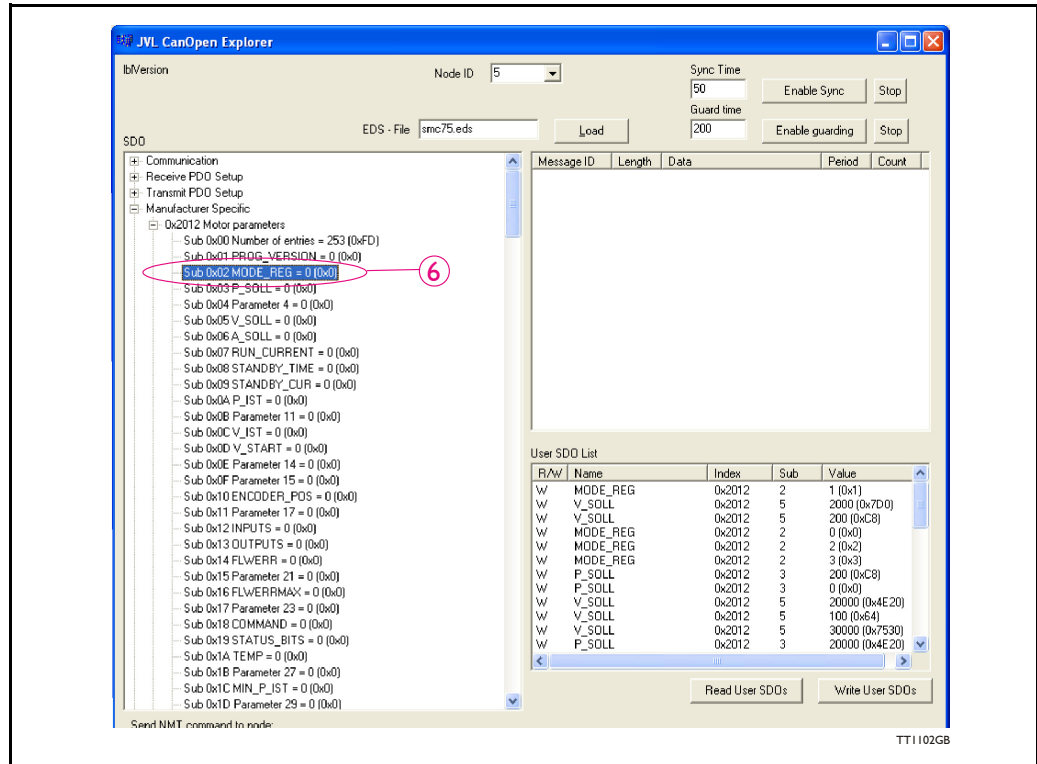
- 4: Select here on the + the manufacturer specific register.
- 5: Select thereafter the object 0x2012. Object 0x2012 contains the motor parameters.



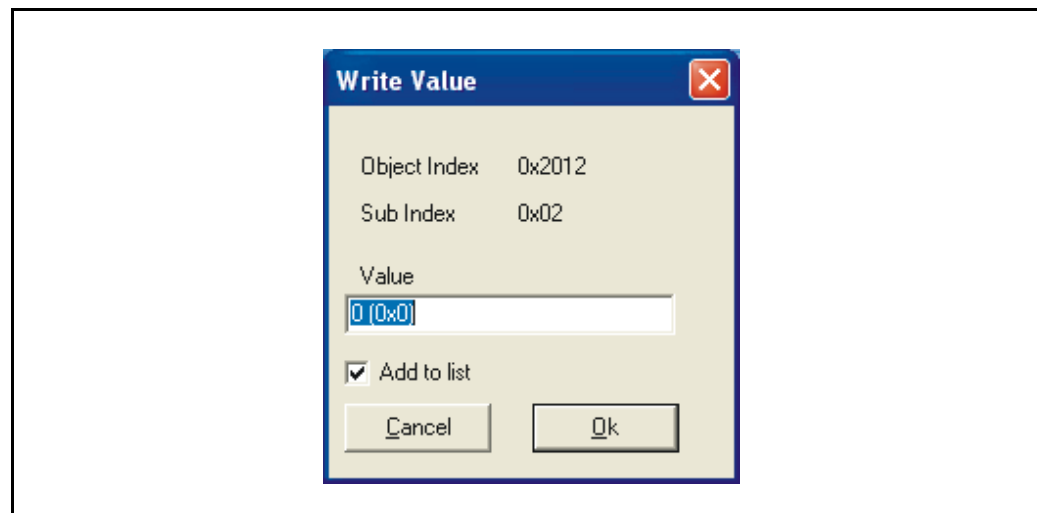
## 11.3

# Using CANopenExplorer

- 6: Point to the sub register 0x02, which is the register that determines in which mode the motor will operate.



Press W on the keyboard. The following screen appears:



- 7: Type 02 in the window, and press OK.
- 8: Click on the sub register 0x05, which is the register to choose the velocity the motor will use. Press W on the keyboard, type 100 in the window, and press OK. The value 100 is in RPM.
- 9: Click on the sub register 0x03, which is the register to choose the distance the motor will run. Press W on the keyboard, type 20000 in the window, and type OK. The value 20000 is in Steps

## 11.3

# Using CANopenExplorer

Now the motor shaft will rotate slowly, until the motor has counted 20000 Encoder pulses. If you want to stop the motor, then click on sub register 0x02 and write 0 in the window, and the motor will switch to passive mode. If using other software, the test could be described as, (using object 2012h):

Sub-register	Name	Width	Unit	Operation	Value
02h	Mode_Reg	16 bit		Set up the motor in position mode	02h
05h	V_SOLL	16 bit	RPM	Sets up the desired velocity	100h
03h	P_SOLL	32 bit	Steps	The motor rotates the desired numbers of encoder pulses	20000
02h	Mode_Reg	16 bit		Sets the motor to passive mode	00h
Returning the motor with higher velocity					
02h	Mode_Reg	16 bit		Set up the motor in position mode	02h
05h	V_SOLL	16 bit	RPM	Sets up the desired velocity	200h
03h	P_SOLL	32 bit	Steps	The motor rotates the desired numbers of Steps	-20000
02h	Mode_Reg	16 bit		Sets the motor in passive mode	00h

### 11.3.3 How to use CANopenExplorer

After startup, the name and details of the HW-interface, such as PCAN\_USB should appear upper left.

When you turn on a motor/CAN node after having started CANopenExplorer, the Data Window (large centre right), will contain a message with the number 0x7xx, where xx is the node ID. For example: 0x704 will indicate node 4. Set the Node ID field top centre to that value (4).

Ensure that the correct EDS\_file is loaded. The program loads a hard-coded default file - either SMC85\_V1\_00\_S.eds (or newer). It is also possible to load another EDS file by writing the file name in the "EDS file" field, top centre, and pressing the load button. Note that the EDS view (large centre left panel) will add the new file at the bottom but will not clear any existing file(s) that are loaded.

Normal operation will be to select an object in the EDS view pane, and press either R for read or W for write. Pressing R should read the value (successful if no error pops up). Pressing W for write will pop up a small window in which the present value is displayed in both decimal and hex. It is then possible to write a new value either in decimal or hex using a 0x prefix, such as 0x185 to enable the first TPDO on node 5 (by clearing the high bit). If the "Add to list" checkbox is checked, the object will be added to the user SDO list as a write SDO. Pressing A performs a read and adds it to the user SDO list pane (lower right) as a read SDO.

The SDOs in the user SDO pane can be rearranged by dragging them with the mouse. Double-clicking on a user SDO list will execute the operation, either reading or writing. The bus state can be changed using the NMT buttons, lower left, e.g. to Operational to enable PDOs.

## 11.3

# Using CANopenExplorer

---

The button Read User SDOs will read all of the “R” type objects in the user SDO list. This is useful for updating a large number of values in the EDS view.

The button Write User SDOs will write all of the “W” type objects in the user SDO list. This is useful for automated testing.

Entries can be deleted from the user SDO list by selecting them with the mouse and pressing the delete key.

The sync Time field (top right) sets the time in milliseconds for the SYNC messages to be sent out. SYNCs can be started and stopped using the buttons Enable Sync and the Stop button to the right.

The Guard Time field below the Sync Time field works like SYNC - just for the Guarding message.

The close button exits the program after saving the list of user SDOs, which will be automatically reloaded at the next program start.

## 11.4 Objects in the DS301 standard

### 11.4.1 DS301 specified Communications objects

The DS301 specified Communications objects are shown in the table below. To obtain the default value in CANopenExplorer, press R on the keyboard, and the actual value will be shown.

Name	Index (hex)	Sub Index	Data Type	Read only	Default	Description
Device type	1000		UNSIGNED32	X	0x40192	Contains information about the device type. See note at top of next page. Mandatory.
Error Register	1001		UNSIGNED8	X		This is the mapping error register, and it is part of the emergency object. If any of the sub indices are high, an error has occurred. See also section 11.4.2. Mandatory
		0			Generic error. Mandatory	
		1			Current	
		2			Voltage	
		3			Temperature	
		4			Communication (Overrun)	
		5			Device profile specific	
		6			Reserved	
	7	Manufacturer specific				
Reservation register	1004					Reservation of PDOs
		0		X		Reserved numbers of PDOs
		1		X		Reserved numbers of syncPDOs
		2		X		Reserved numbers of asyncPDOs
Manufacturer device name	1008		VISIBLE STRING	X	JVL A/S	
Manufacturer hardware version	1009		VISIBLE STRING	X		
Manufacturer software version	100A		VISIBLE STRING	X		Example: Version x.x

## 11.4 Objects in the DS301 standard

Name	Index (hex)	Sub Index	Data Type	Read only	Default	Description
Guard time	100C		UNSIGNED16			Informs about the Guard time in milliseconds. Is only mandatory if the module does not support heartbeat
Life time factor	100D		UNSIGNED8			Is the factor that guard time is multiplied with to give the life time for the node quarding protocol
Heartbeat time	1017		UNSIGNED8			If the Heartbeat timer is not 0, Heartbeat is used.
Identity object	1018		IDENTITY	X		Contain general information about the module
		0	1 to 4	X	4h	Number of entries. Mandatory
		1	UNSIGNED32	X	0x0117	Vendor ID, contains a unique value allocated to each manufacturer. 117h is JVLs vendor ID. Mandatory.
		2	UNSIGNED32	X	0x0200	Product Code, identifies a specific device version. MISxxx has the product code 300H.
		3	UNSIGNED32	X		Revision number.
		4	UNSIGNED32	X		Serial number

### 11.4.2 Emergency object

The EMCY (emergency) object is used to transfer an error message to the CANopen master, or also to another node which can process the error message. The reaction on the emergency object is not specified. An emergency object is transmitted only once per “error event”.

The MIS motor (or SMC66/85) supports the EMC object (Emergency). The following error codes can be generated:

Error code 1001h: Generic error - Motor error  
 Error code 1002h: Generic error - Position error  
 Error code 1003h: Generic error - Follow error  
 Error code 1004h: Generic error - Low

Transmit PDO25:

Use Transmit PDO25 in asynchronous mode to read the status of the error.

In the MIS motor (or SMC66/85), no error control is enabled when the modules are started up because if there is any fault in the system, it is impossible to get in contact with the module. After the module has started up and there is communication between the master and the slave, turn on the required error control mechanism in the communication objects, see [DS301 specified Communications objects](#), page 242.

## 11.4 Objects in the DS301 standard

---

### 11.4.3 Object dictionary

Name	Index (hex)	Sub Index	Type	Read only	Default	Description
Motor parameters	2012	0	Unsigned8	x	254	Subindex count
		n	Unsigned32			Access to the 32 bit motor register, n
Motor parameters	2014	0	Unsigned8	x	254	Subindex count
		n	Unsigned16			Access to the motor register n, but as 16bit

Writing to these objects in CANopenExplorer is done by pressing W on the keyboard when the register in folder Manufacturer is selected. Reading is done by pressing R.

#### Object 2012h – Motor parameters

With this object, all the registers of the MIS motor can be accessed. All the registers are accessed as 32 bit. When reading and writing to 16-bit registers, the values are automatically converted in the module.

#### Object 2014h – Motor parameters (16 bit)

Works as 2012h, but the parameters are accessed as 16-bit. If writing to a 32bit parameter, the 16-bit value will be treated as signed.

### 11.4.4 Enable and Disable PDOs

In the CANopen profile, it is only possible to have four transmit and four receive PDOs enabled at the same time. In the MIS motor (or SMC66/85), all PDOs are disabled when the module is booted up. The user must choose which PDOs the application will use and enable these.

To enable or disable a PDO, it is necessary to write to the MSB (bit 31) in the PDO COB-ID entry in the PDO communication parameter Record. The COB-ID register is sub-index 1h, and the value range of this register is UNSIGNED32.

The PDOs are enabled when bit 31 is 0, and is disabled when bit 31 is 1.



# 11.4 Objects in the DS301 standard

## 11.4.5 Dynamic Mapping

All motor registers are available in the "Manufacturer Specific" objects.

Example: Dynamic Mapping in CANopenExplorer

CANopenExplorer shows that TxPDO1 is mapped to transmit object 0x2012, sub 0x02, 32 bit.

If the user wants to map to another object, following procedure must be followed:

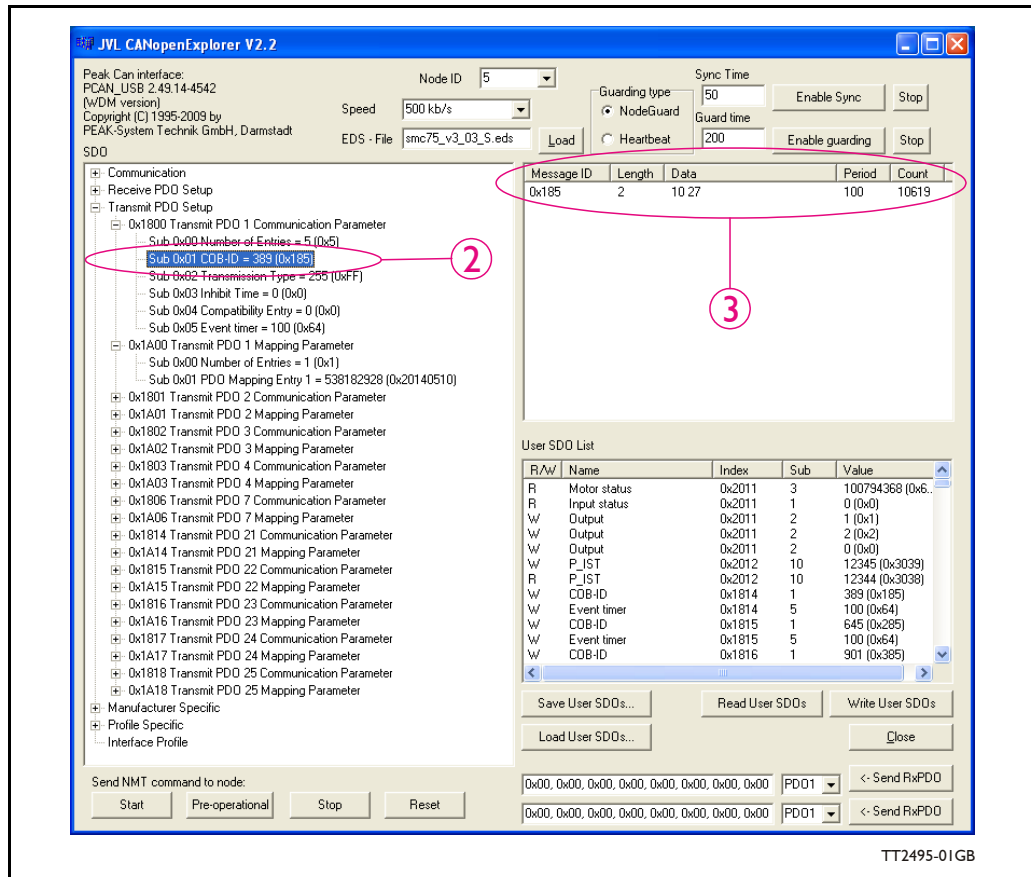
- De-activate the PDO by writing 1 to MSB of the COB-ID: 0x8000185 - see #1 at the illustration below.
- Write 0 in Number of Entries.
- Remap to another object by writing to PDO Mapping Entry 1: 0x20140510.
- Write 1 in Number of Entries.

The screenshot shows the JVL CANopenExplorer V2.2 interface. The 'SDD' tree on the left is expanded to '0x1800 Transmit PDO 1 Communication Parameter'. A red circle highlights the entry 'Sub 0x01 COB-ID = 389 (0x185)', with a vertical bar icon next to it. The 'User SDO List' table at the bottom right shows the following data:

R/W	Name	Index	Sub	Value
R	Motor status	0x2011	3	100794368 (0x6...)
R	Input status	0x2011	1	0 (0x0)
W	Output	0x2011	2	1 (0x1)
W	Output	0x2011	2	2 (0x2)
W	Output	0x2011	2	0 (0x0)
W	P_IST	0x2012	10	12345 (0x3039)
R	P_IST	0x2012	10	12344 (0x3038)
W	COB-ID	0x1814	1	389 (0x185)
W	Event timer	0x1814	5	100 (0x64)
W	COB-ID	0x1815	1	645 (0x285)
W	Event timer	0x1815	5	100 (0x64)
W	COB-ID	0x1816	1	901 (0x385)

## 11.4 Objects in the DS301 standard

- Activate the PDO by writing 0 to MSB of the COB-ID: 0x185 - see #2 at the illustration below.
- Now we receive the lower 16 bit of the V\_SOLL register (0x05): 0x2710 = 10000 = 100 RPM - see #3 at the illustration below.



### 11.4.6 Receive PDOs

The PDO 1-20 are reserved for use with DSP-402. The following receive PDOs are available:

Receive PDO 21:

This PDO can be used to update the position, velocity and acceleration. The data in the PDO is written directly to the position register and if the motor is in position mode, it will start moving to that position.

## 11.4 Objects in the DS301 standard

The table below shows default values of the COB-ID:

PDO	Sub-index	Type	Description	Default	Access type
21	1	Receive	COB-ID	Nodeid+0x80000200	r/w
	1	Transmit	COB-ID	Nodeid+0x80000180	r/w
22	1	Receive	COB-ID	Nodeid+0x80000300	r/w
	1	Transmit	COB-ID	Nodeid+0x80000280	r/w
23	1	Receive	COB-ID	Nodeid+0x80000400	r/w
	1	Transmit	COB-ID	Nodeid+0x80000380	r/w
24	1	Receive	COB-ID	Nodeid+0x80000500	r/w
	1	Transmit	COB-ID	Nodeid+0x80000480	r/w
25	1	Transmit	COB-ID	Nodeid+0x80000480	r/w

### Receive PDO 21:

With this PDO it is possible to update the target position (P\_SOLL), the maximum velocity (V\_SOLL) and the acceleration (A\_SOLL).

Byte	0	1	2	3	4	5	6	7
Data	P_SOLL			V_SOLL		A_SOLL		
Object	2012h, sub 3			2014h, sub 5		2014h, sub 6		

### Receive PDO 22:

With this PDO it is possible to update the running current and operating mode.

Byte	0	1	2	3	4	5	6	7
Data	RUN_CURRENT		MODE_REG					
Object	2014h, sub 7		2014h, sub 2					

### Receive PDO 23:

This PDO can be used to issue a Motor command.

Byte	0	1	2	3	4	5	6	7
Data	Motor Command		Reserved	Reserved	Reserved	Res.	Res.	Res.
Object	2014h, sub 24							

### Receive PDO 24:

This PDO updates the outputs.

Byte	0	1	2	3	4	5	6	7
Data	Output data		Reserved	Reserved	Reserved	Res.	Res.	Res.
Object	2014h, sub 19							

## 11.4 Objects in the DS301 standard

---

### 11.4.7 Transmit PDOs

The PDOs 1-20 are reserved for use with DSP-402.

All of the transmit PDOs support synchronous transmission. PDO 25 also supports asynchronous transmission.

Transmit PDO 21:

With this PDO the actual position can be read.

Byte	0	1	2	3	4	5	6	7
Data	P_IST				V_IST		Motor error	
Object	2012h, sub 10				2014h, sub 12		2014h, sub 35	

Transmit PDO 22:

With this PDO the actual velocity can be read.

Byte	0	1	2	3	4	5	6	7
Data	V_IST		Unused	Unused	Unused	Unused	Unused	Unused
Object	2014h, sub 12							

Transmit PDO 23:

With this PDO the value of the analogue inputs 1-4 can be read.

Byte	0	1	2	3	4	5	6	7
Data	ANALOGUE1		ANALOGUE2		ANALOGUE3		ANALOGUE4	
Object	2014h, sub 89		2014h, sub 90		2014h, sub 91		2014h, sub 92	

## 11.4 Objects in the DS301 standard

### Transmit PDO 24:

With this PDO the value of the analogue inputs 4-8 can be read.

Byte	0	1	2	3	4	5	6	7
Data	ANALOGUE5		ANALOGUE6		ANALOGUE7		ANALOGUE8	
Object	2014h, sub 93		2014h, sub 94		2014h, sub 95		2014h, sub 96	

### Transmit PDO 25:

With this PDO the motor status, inputs and last error can be read.

This PDO also supports asynchronous transmission. If this PDO is in asynchronous mode, it will be transmitted every time the run status or inputs are changed.

Byte	0	1	2	3	4	5	6	7
Data	Inputs		Motor error		Unused	Unused	Unused	Unused
Object	2014h, sub 18		2014h, sub 35					

### 11.4.8 Beckhoff support

The MIS motors (or SMC66/85) supports running CAN with Beckhoff PLC.

In this mode, 4 receive and transmit PDO's are enabled from startup and are configured as PDO I-4.

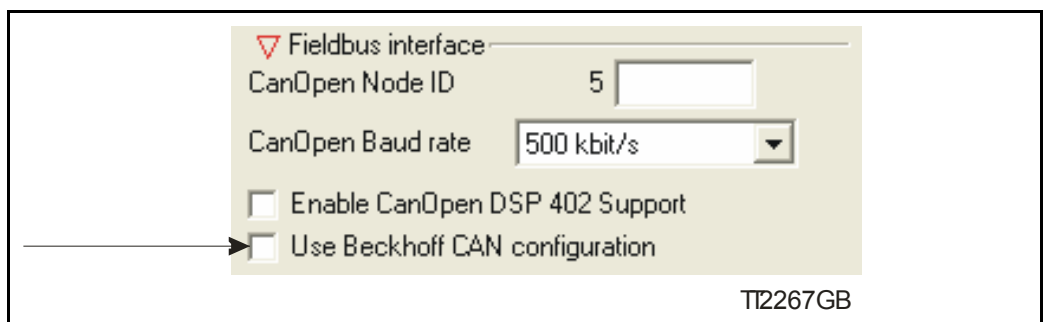
COB\_ID = 0x800000xxx: NOT ENABLED

COB\_ID = 0x000000xxx: ENABLED

### 11.4.9 PDO setup in Beckhoff mode

Normally each selected PDO needs to be enabled after power up and initialization but in Beckhoff mode PDO I-4 is automatically enabled at power up.

To setup and use the Beckhoff mode, enable the Beckhoff support from MacTalk and press the Save in flash -button.



## 11.4 Objects in the DS301 standard

### 11.4.10 Beckhoff receive PDO setup

The table below shows default values of the COB-ID:

PDO	Sub-index	Type	Description	Default	Access type
1	1	Receive	COB-ID	Nodeid+0x00000200	r/w
	1	Transmit	COB-ID	Nodeid+0x00000180	r/w
2	1	Receive	COB-ID	Nodeid+0x00000300	r/w
	1	Transmit	COB-ID	Nodeid+0x00000280	r/w
3	1	Receive	COB-ID	Nodeid+0x00000400	r/w
	1	Transmit	COB-ID	Nodeid+0x00000380	r/w
4	1	Receive	COB-ID	Nodeid+0x00000500	r/w
	1	Transmit	COB-ID	Nodeid+0x00000480	r/w

Receive PDO 1

Byte	0	1	2	3	4	5	6	7
Data	P_SOLL			V_SOLL		A_SOLL		
Object	2012h, sub 3			2014h, sub 5		2014h, sub 6		

Receive PDO 2:

With this PDO it is possible to update the running current and operating mode.

Byte	0	1	2	3	4	5	6	7
Data	RUN_CURRENT		MODE_REG					
Object	2014h, sub 7		2014h, sub 2					

Receive PDO 3:

This PDO can be used to issue a Motor command.

Byte	0	1	2	3	4	5	6	7
Data	Motor Command		Reserved	Reserved	Reserved	Res.	Res.	Res.
Object	2014h, sub 24							

Receive PDO 4:

This PDO updates the outputs.

Byte	0	1	2	3	4	5	6	7
Data	Output data		Reserved	Reserved	Reserved	Res.	Res.	Res.
Object	2014h, sub 19							

## 11.4 Objects in the DS301 standard

### 11.4.11 Beckhoff transmit PDO setup

Transmit PDO 1:

With this PDO the actual position can be read.

Byte	0	1	2	3	4	5	6	7
Data	P_IST				V_IST		Motor error	
Object	2012h, sub 10				2014h, sub 12		2014h, sub 35	

Transmit PDO 2:

With this PDO the value of the analogue inputs 1-4 can be read.

Byte	0	1	2	3	4	5	6	7
Data	ANALOGUE1		ANALOGUE2		ANALOGUE3		ANALOGUE4	
Object	2014h, sub 89		2014h, sub 90		2014h, sub 91		2014h, sub 92	

Transmit PDO 3:

With this PDO the value of the analogue inputs 4-8 can be read.

Byte	0	1	2	3	4	5	6	7
Data	ANALOGUE5		ANALOGUE6		ANALOGUE7		ANALOGUE8	
Object	2014h, sub 93		2014h, sub 94		2014h, sub 95		2014h, sub 96	

Transmit PDO 4:

With this PDO the actual velocity can be read.

Byte	0	1	2	3	4	5	6	7
Data	V_IST		Reserved	Reserved	Reserved	Res.	Res.	Res.
Object	2014h, sub 18		User selectable 16-bit register exc. STATUSBITS (register 25)		User selectable 32-bit register exc. ENCODER_POS (register16)			

## 11.5 Objects used in the DSP-402 standard

---

### 11.5.1 DSP-402 Support

**This feature is not released yet - consult JVL to get more info.**

#### **Introduction**

The MIS23x (SMC75) supports the DSP-402 standard from CiA (<http://www.can-cia.com/>).

Please refer to this standard for details of the functions.

The DSP-402 is only a standard proposal and might be changed in the future. JVL therefore reserves 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 mandatory functions are supported.

The following operation modes are supported:

- Profile position mode
- Velocity mode
- Zero Search mode

#### **Preconditions**

The start mode of the motor must be set to passive.

No power up zero searches must be selected.

When using the DSP-402 mode, manipulating parameters with object 2012h or 2014h can corrupt the behaviour of the DSP-402 functions. Also be aware that manipulating parameters in MacTalk should be avoided when using DSP-402.



## 11.5 Objects used in the DSP-402 standard

### Supported objects

The following table gives the additional object dictionary defined for DSP-402 support.

Name	Index (hex)	Sub Index	Type	Read only	Default
Device data					
Motor_type	6402	0	UNSIGNED16	X	9
Motor_catalog_number	6403	0	VISIBLE_STRING	X	MIS23x (SMC75)
Motor_manufacturer	6404	0	VISIBLE_STRING	X	JVL A/S
http_motor_catalog_address	6405	0	VISIBLE_STRING	X	www.jvl.dk
Supported_drive_modes	6502	0	UNSIGNED32	X	37
Drive_catalog_number	6503	0	VISIBLE_STRING	X	MIS23x (SMC75)
Drive_manufacturer	6504	0	VISIBLE_STRING	X	JVL A/S
http_drive_catalog_address	6505	0	VISIBLE_STRING	X	www.jvl.dk
Digital I/O					
Digital_inputs	60FD	0	UNSIGNED32	X	
Digital_outputs	60FE	0	UNSIGNED8	X	
Digital_outputs_Physical_outputs	60FE	1	UNSIGNED32		
Digital_outputs_Bit_mask	60FE	2	UNSIGNED32		
Device Control					
Abort_connection_option_code	6007	0	INTEGER16		
Error_code	603F	0	UNSIGNED16		
Control word	6040	0	UNSIGNED16		
Status word	6041	0	UNSIGNED16	X	
Quick_stop_option_code	605A	0	INTEGER16		
Modes_of_operation	6060	0	INTEGER8		
Modes_of_operation_display	6061	0	INTEGER8	X	
Profile Position parameters					
Position_actual_value	6064	0	INTEGER32	X	
Target_position	607A	0	INTEGER32		
Software_position_limit	607D	0	UNSIGNED8	X	2
Software_position_limit_Min_position_limit	607D	1	INTEGER32		
Software_position_limit_Max_position_limit	607D	2	INTEGER32		
Max_motor_speed	6080	0	UNSIGNED32		
Profile_velocity	6081	0	UNSIGNED32		
Profile_acceleration	6083	0	UNSIGNED32		

## 11.5 Objects used in the DSP-402 standard

Name	Index (hex)	Sub Index	Type	Read only	Default
Quick_stop_deceleration	6085	0	UNSIGNED32		
Motion_profile_type	6086	0	INTEGER16		
Profile velocity mode					
Velocity_sensor_actual_value	6069	0	INTEGER32	X	
Velocity_demand_value	606B	0	INTEGER32	X	
Velocity_actual_value	606C	0	INTEGER32	X	
Velocity_window	606D	0	UNSIGNED16		
Velocity_window_time	606E	0	UNSIGNED16		
Target_velocity	60FF	0	INTEGER32		
Max_torque	6072	0	UNSIGNED16		
Zero Search mode					
Home_offset	607C	0	INTEGER32		
Homing_method	6098	0	INTEGER8		
Homing_speeds	6099	0	UNSIGNED8	X	2
Homing_speeds_Speed_during_search_for_switch	6099	1	UNSIGNED32		
Homing_speeds_Speed_during_search_for_zero	6099	2	UNSIGNED32		
Homing_acceleration	609A	0	UNSIGNED32		
Factors					
Position_notation_index	6089	0	INTEGER8		
Position_dimension_index	608A	0	UNSIGNED8		
Velocity_notation_index	608B	0	INTEGER8		
Velocity_dimension_index	608C	0	UNSIGNED8		
Acceleration_notation_index	608D	0	INTEGER8		
Acceleration_dimension_index	608E	0	UNSIGNED8		
Position_encoder_resolution	608F	0	UNSIGNED8	X	2
Position_encoder_resolution_Encoder_increments	608F	1	UNSIGNED32		
Position_encoder_resolution_Motor_revolutions	608F	2	UNSIGNED32		
Velocity_encoder_resolution	6090	0	UNSIGNED8	X	2
Velocity_encoder_resolution_Encoder_increments_per_second	6090	1	UNSIGNED32		
Velocity_encoder_resolution_Motor_revolutions_per_second	6090	2	UNSIGNED32		
Gear_ratio	6091	0	UNSIGNED8	X	2
Gear_ratio_Motor_revolutions	6091	1	UNSIGNED32		
Gear_ratio_Shaft_revolutions	6091	2	UNSIGNED32		
Feed_constant	6092	0	UNSIGNED8	X	2

## 11.5 Objects used in the DSP-402 standard

Name	Index (hex)	Sub Index	Type	Read only	Default
Feed_constant_Feed	6092	1	UNSIGNED32		
Feed_constant_Shaft_revolutions	6092	2	UNSIGNED32		
Position_factor	6093	0	UNSIGNED8	X	2
Position_factor_Numerator	6093	1	UNSIGNED32		
Position_factor_Feed_constant	6093	2	UNSIGNED32		
Velocity_encoder_factor	6094	0	UNSIGNED8	X	2
Velocity_encoder_factor_Numerator	6094	1	UNSIGNED32		
Velocity_encoder_factor_Divisor	6094	2	UNSIGNED32		
Acceleration_factor	6097	0	UNSIGNED8	X	2
Acceleration_factor_Numerator	6097	1	UNSIGNED32		
Acceleration_factor_Divisor	6097	2	UNSIGNED32		
Polarity	607E	0	UNSIGNED8		

### 11.5.2 Factors

#### Position factor

The position factor is the relation between the user unit and the internal position unit (steps).

The position factor is automatically calculated when the feed constant (Object 6092h) and gear ratio (Object 6091h) are set.

Example:

A MIS232 Motor with a 3.5:1 gear box is connected to a belt drive. The diameter of the drive wheel is 12.4 cm.

The unit of position is required to be in millimetres.

The perimeter of the drive wheel is 389.56mm (124mm\*pi)

The parameters should be set as follows:

Object	Name	Value
6091 <sub>h</sub> subindex 1	Gear ratio - Motor revolutions	35
6091 <sub>h</sub> subindex 2	Gear ratio - Shaft revolutions	10
6092 <sub>h</sub> subindex 1	Feed constant - Feed	38956
6092 <sub>h</sub> subindex 2	Feed constant - Shaft revolutions	100

## 11.5 Objects used in the DSP-402 standard

---

### Velocity encoder factor

This factor is used to convert the user unit into the internal unit (RPM).  
The factor is adjusted with the object 6094h.

Example 1:

An MIS232 has 1600 counts/revolution.

We want the user unit of velocity to be in RPM. This is the same as the internal unit.

The parameters should be set as follows:

Object	Name	Value
6094 <sub>h</sub> subindex 1	Velocity encoder factor - Numerator	1600
6094 <sub>h</sub> subindex 2	Velocity encoder factor – Divisor	1600

Example 2:

We have an MIS232 that uses RPM as the internal velocity and the same belt drive as in the above Position factor example.

We want the user unit of velocity to be in mm/s.

The parameters should be set as follows:

Object	Name	Calculated value	Value
6094 <sub>h</sub> subindex 1	Velocity encoder factor - Numerator	$(60 \cdot 3.5) / 389.56$ = 0.53907	53907
6094 <sub>h</sub> subindex 2	Velocity encoder factor – Divisor	1	100000

### Acceleration factor

This factor is used to convert the user unit into the internal unit (9.54 RPM/s).  
The factor is adjusted with the object 6097h.

Example 1:

We have an MIS232 with 1600 counts/revolution.

We want the user unit of acceleration to be in RPM/s.

The parameters should be set as follows:

Object	Name	Value
6097 <sub>h</sub> subindex 1	Acceleration encoder factor - Numerator	100
6097 <sub>h</sub> subindex 2	Acceleration encoder factor – Divisor	954

## 11.5 Objects used in the DSP-402 standard

---

Example 2:

We have an MIS232 with 1600 counts/revolution and the same belt drive as in the above Position factor example.

We want the user unit of acceleration to be in mm/s<sup>2</sup>.

The parameters should be set as follows:

Object	Name	Calculated value	Value
6097 <sub>h</sub> subindex 1	Acceleration factor- Numerator	$(3.5*60) / 389.56$ = 0.53907	53907
6097 <sub>h</sub> subindex 2	Acceleration factor - Divisor	9.54	954000

### 11.5.3 Changing operation mode

Change of operation mode is only possible when the operation mode is not enabled.

There is one exception and that is when changing from Zero Search mode to profile position mode. This is possible when the Zero Search sequence is completed and can be done even though the operation mode is enabled.

### 11.5.4 Profile position mode

This mode can be used for positioning in which a move profile can be set up. The acceleration and maximum velocity can be programmed.

In this mode both absolute and relative movement is supported. This is selected using bit 6 (absolute/relative) in the status word. It is also possible to select different movement modes. This is done with 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 until 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.

### 11.5.5 Velocity mode

In this mode the motor runs at a selected velocity. A new velocity can be selected and the motor will then accelerate/decelerate to this velocity.

The maximum slippage error is not supported in this mode.

### 11.5.6 Zero Search mode

Using this mode, different Zero Search sequences can be initiated. The standard Zero Search modes from 1-34 are supported. Before starting the Zero Search, the inputs must be configured properly using MacTalk or parameters 125, 129, 130, 132.

## 11.5 Objects used in the DSP-402 standard

---

### 11.5.7 Supported PDOs

#### Receive PDOs

PDO no.	Mapping object index	Mapping object name	Comment
1	6040 <sub>h</sub>	Control word	Controls the state machine
2	6040 <sub>h</sub> 6060 <sub>h</sub>	Control word Modes of operation	Controls the state machine and modes of operation
3	6040 <sub>h</sub> 607A <sub>h</sub>	Control word Target position	Controls the state machine and the target position (pp)
4	6040 <sub>h</sub> 60FF <sub>h</sub>	Control word Target velocity (pv)	Controls the state machine and the target velocity (pv)
7	6040 <sub>h</sub> 60FE <sub>h</sub>	Control word Digital outputs	Controls the state machine and the digital outputs

#### Transmit PDOs

PDO no.	Mapping object index	Mapping object name	Event driven
1	6041h	Status word	Yes
2	6041h 6061h	Status word Modes of operation display	Yes
3	6041h 6064h	Status word Position actual value	No
4	6041h 606Ch	Status word Velocity actual value	No
7	6041h 60FDh	Status word Digital inputs	Yes

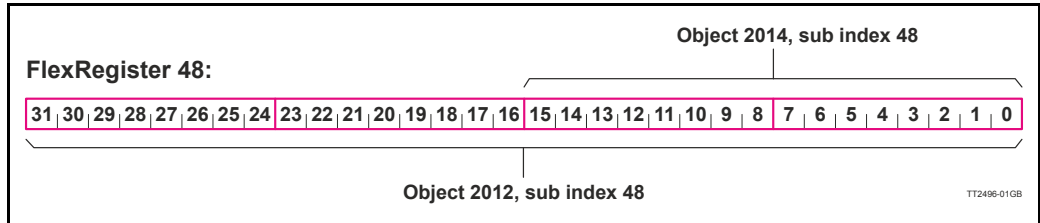
# 11.6 Flexible Register setup

## 11.6.1 Introduction.

Register 48 is a so-called "FlexRegister".

It can be built bit-by-bit from other registers and is therefore fully user customizable. This makes it possible to pack the data more efficient and thereby keeping the CAN bus load at a minimum.

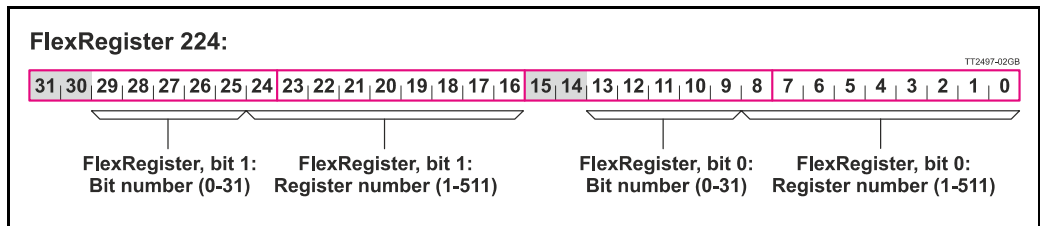
The register is available as 16- (object 2014, sub 48) and 32-bit (object 2012, sub 48).



## 11.6.2 How to setup.

To setup this register use the register 224 to 231.

Register 224 will setup FlexRegister bit 0 and 1 this way:



And the rest setup registers follow the same pattern up to 231, which represents FlexRegister bit 14 and 15.

The setup can be done manually with SDO access or directly in MacRegIO.

Please note: The function will only be activated if register 224 (FlexRegister bit 0 and 1) is set.

## 11.7 More details of CANopen Theory

### 11.7.1 CANopen DS-301 device profiles

Standardized devices in CANopen have their characteristics described in a device profile. For each device profile, particular data and parameters are strictly defined. Data and parameters are known as objects in CANopen. Objects perform all processes in CANopen; they can perform various tasks, either as communications objects or as device-specific objects where they are directly related to the device. A communication object can transport data to the bus control and establish connection, or supervise the network devices.

The application layer makes it possible to exchange meaningful real-time-data across the CAN network. The format of this data and its meaning must be known by the producer and the consumer(s). There are encoding rules that define the representation of values of data types and the CAN network transfer syntax for the representations. Values are represented as bit sequences. Bit sequences are transferred in sequences of octets (byte). For numerical data types, the encoding is with the lowest byte first.

Every object is described and classified in the object dictionary (or index) and is accessible via the network. Objects are addressed using a 16-bit index so that the object dictionary may contain a maximum of 65536 entries.

Index (Hex)	Object	Supported
0000-	Not used	
0001-001F	Static data types	
0020-003F	Complex data types	
0040-005F	Manufacturer specific Data Types	
0060-0FFF	Reserved for further use	
1000-1FFF	Communication Profile area DS301	Yes
2000-5FFF	Manufacturer specific profile area	Yes
6000-9FFF	Standardised Device Profile area (DSP-402)	Yes
A000-FFFF	Reserved for further use	

#### Index 0001-001F:

Static data types contain type definitions for standard data types like boolean, integer, floating point, etc. These entries are included for reference only, they cannot be read or written.

#### Index 0020-003F:

Complex data types are predefined structures that are composed out of standard data types and are common to all devices.

#### Index 0040-005F:

Manufacturer-specific data types are also structures composed of standard data types but are specific to a particular device.

#### Index 1000-1FFF:

The communication Profile area contains the parameters for the communication profile on the CAN network. These entries are common to all devices.

#### Index 2000-5FFF:

The manufacturer-specific profile area, for truly manufacturer-specific functionality.



## 11.7 More details of CANopen Theory

Index 6000-9FFF:

The standardised device profile area contains all data objects common to a class of devices that can be read or written via the network. The drives profile uses entries from 6000h to 9FFFh to describe the drive parameters and the drive functionality. Within this range, up to 8 devices can be described. In such a case, the devices are denominated Multi Device Modules. Multi Device Modules are composed of up to 8 device profile segments. Using this feature it is possible to build devices with multiple functionality. The different device profile entries are shifted with 800h.

A 16-bit index is used to address all entries within the object dictionary. In the case of a simple variable, this index references the value of the variable directly. In the case of records and arrays however, the index addresses the whole data structure. To allow individual elements of structures of data to be accessed via the network, a sub-index has been defined. For single object dictionary entries such as Unsigned8, Boolean, Integer32, the value of the sub-index is always zero. For complex object dictionary entries such as arrays or records with multiple data fields, the sub-index refers to fields within a data-structure pointed to by the main index. Index counting starts with one.

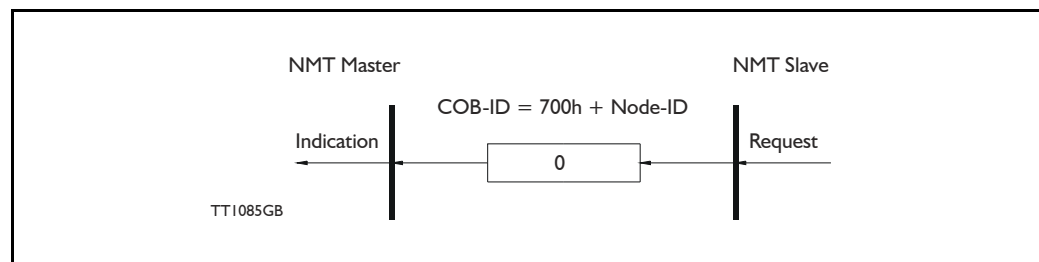
The DS-301 standard constitutes the application and the communications profile for a CANopen bus, and is the interface between the devices and the CAN bus. It defines the standard for common data and parameter exchange between other bus devices, and it controls and monitors the devices in the network. The table below lists some of the communications profile objects:

Data Transfer	Parameter Transfer	Special functions	
PDO			Process Data Objects
	SDO		Service Data Objects
		SYNC	Synchronisation
		EMCY	Emergency

The access from the CAN network is done through data objects PDO (Process Data Object) and SDO (Service Data Object).

### 11.7.2 Boot up telegram

After the initialization phase, a CANopen slave logs on with a boot up message. The node address of the slave is contained in this. This allows a CANopen master to know which slaves are connected to the network. The protocol uses the same identifier as the error control protocols. See the figure below:



One data byte is transmitted with value 0.

# 11.7 More details of CANopen Theory

## 11.7.3 PDO (Process Data Object)

PDO: Performs real-time transfers, and the transfer of PDOs is performed without a protocol. PDOs are used in two ways: for data transmission and for data reception. PDOs can bundle all objects from the object data directory, and a PDO can handle max 8 bytes of data in the same PDO. The PDO can consist of multiple objects. Another PDO characteristic is that it does not reply when it is receiving data, in order to make data transfer fast. It has a high priority identifier.

PDO connections follow the Producer/Consumer model, whereby a normal PDO connection follows the Push model and an RTR connection the Pull model.

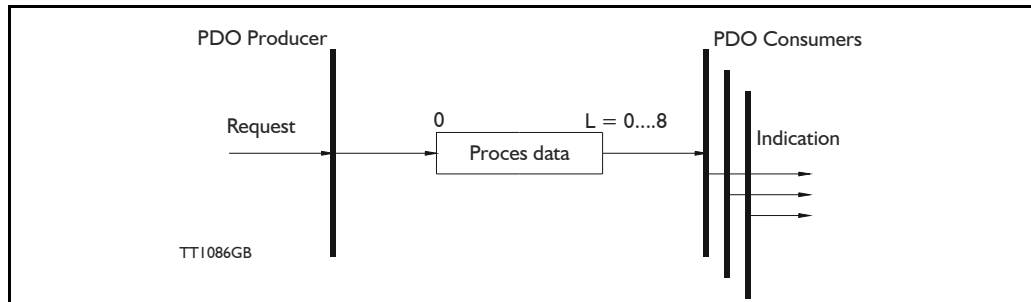
Objects are mapped in a PDO. This mapping is an agreement between the sender and receiver regarding which object is located at which position in the PDO. This means that the sender knows at which position in the PDO it should write data and the receiver knows where it should transfer the data to that is received.

The PDOs correspond to entries in the Device Object Dictionary and provide the interface to the application objects. Data type and mapping of application objects into a PDO are determined by a corresponding PDO mapping structure within the Device object Dictionary. The number and length of PDOs of a device are application specific and must be specified within the device profile

Write PDO service:

The Write PDO service is unacknowledged. A PDO producer sends its PDO to the PDO consumer. There can be 0 or more consumers in the network. For receive PDOs the MIS23x (SMC75) is the consumer and for Transmit PDOs, the producer.

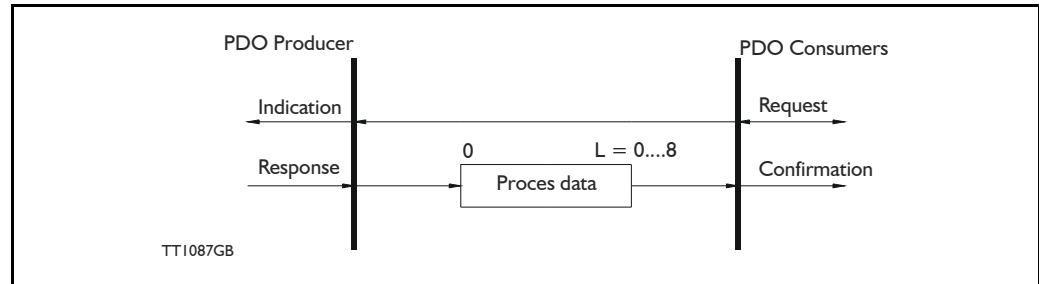
The following figure shows a Write PDO service:



## 11.7 More details of CANopen Theory

Read PDO service:

The read PDO service is an acknowledged service. One of the several PDO consumers send an RTR message to the network. After it has received the RTR message, the PDO producer sends the requested PDO. This service is used for RTR queries. Using this service, an actual value can be interrogated independently of the selected cycle time. The following figure shows a read PDO service:



PDO identifier:

In the CANopen profile, it is only possible to have four transmit and four receive PDOs enabled at the same time. In the MIS motors (or SMC66/85), all PDOs are disabled when the module is booted up. The user must choose which PDOs the application will use and enable these.

The PDO configuration can be seen either in the EDS-file or in the CANopenExplorer program, where the communication and the mapping parameters are shown.

There are two standard methods to map the PDOs in CANopen: static mapping and dynamic mapping. In static PDO mapping all PDOs are mapped in accordance with some fixed, non-modifiable setting in the relevant PDO. In dynamic PDO mapping, the setting of a PDO can be modified. It is also allowable to have a flexible combination of different process data during operation.

### 11.7.4 SDO (Service Data Objects)

SDO: can access all entries in the object directory but they are normally used in the initialization during the boot up procedure. Some SDOs characteristics are:

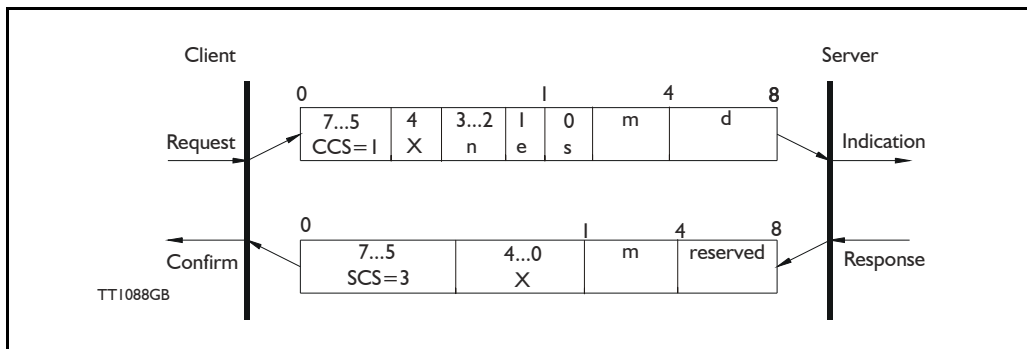
- Confirmed transfer of objects
- Data transfer/exchange is always non-synchronous
- Values greater than 4 bytes are transferred (Normal transfer)
- Values not more than 4 bytes are transferred (Expedited transfer)

Basically an SDO is transferred as a sequence of segments. Prior to transferring the segment, there is an initialization phase where client and server prepare themselves for transferring the segment. For SDOs, it is also possible to transfer a dataset of up to four bytes during the initialization phase. This mechanism is called an expedited transfer.

Download SDO protocol:

The download SDO protocol is used to write the values of the object directory into the drive.

# 11.7 More details of CANopen Theory



Upload SDO protocol:  
The upload SDO protocol is used to read the values in the object directory of the drive.

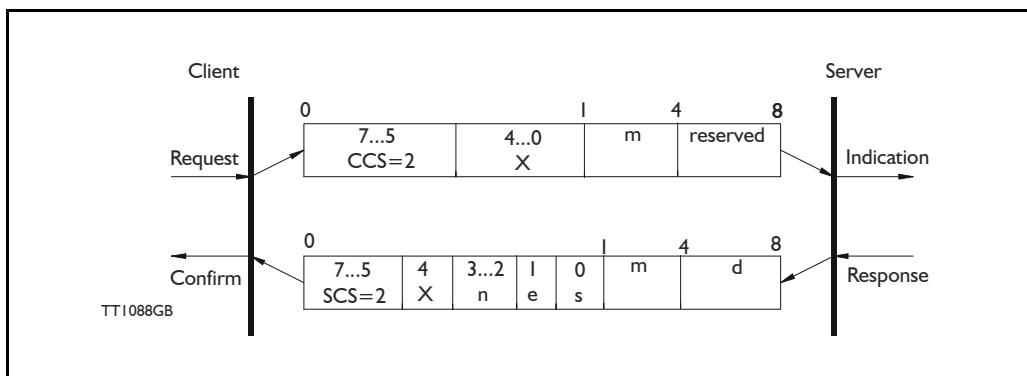


Table for upload and download SDO protocol.

	CCS:	SCS:	n:	e:	s:	m:
Down-load	1: Initiate down-load request	3: Initiate download response	Only valid if e=1 and s=1 otherwise 0. If valid it indicates the number of bytes in d that do not contain data. Bytes [8-n,7] do not contain data	Transfer type: 0=normal transfer 1=expedited transfer	Size indicator: 0=data set size is not indicated 1=data set size is indicated	Multiplexer. It represents the index/sub-index of the data to be transfer by the SDO
Upload	2: Initiate upload request	2: Initiate upload response	Only valid if e=1 and s=1 otherwise 0. If valid it indicates the number of bytes in d that do not contain data. Bytes [8-n,7] do not contain data	Transfer type: 0=normal transfer 1=expedited transfer	Size indicator: 0=data set size is not indicated 1=data set size is indicated	Multiplexer. It represents the index/sub-index of the data to be transfer by the SDO

CCS: Client command specified.  
SCS: Server commander specified.

## 11.7 More details of CANopen Theory

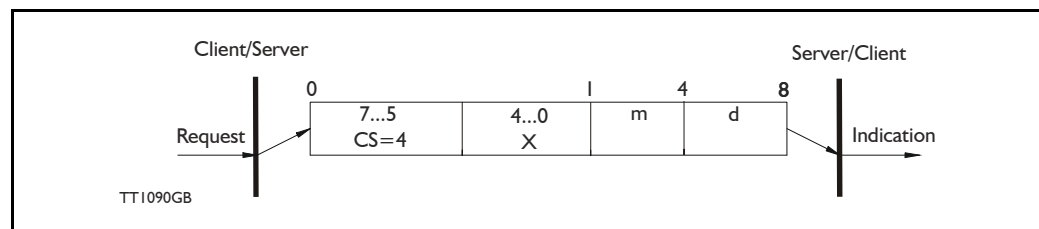
Table for upload and download SDO protocol (continued)

	<b>d:</b>	<b>X:</b>	<b>Reserved:</b>
Download	<p><b>e=0, s=0:</b> d is reserved for further use</p> <p><b>e=0, s=1:</b> d contains the number of bytes to be downloaded. Byte 4 contains the lsb and byte 7 contains the msb</p> <p><b>e=1, s=1:</b> d contains the data of length 4-n to be downloaded, the encoding depends on the type of the data referenced by index and sub-index.</p>	not used, always 0	Reserved for further use, always 0
Upload	<p><b>e=0, s=0:</b> d is reserved for further use</p> <p><b>e=0, s=1:</b> d contains the number of bytes to be uploaded. Byte 4 contains the lsb and byte 7 contains the msb</p> <p><b>e=1, s=1:</b> d contains the data of length 4-n to be uploaded, the encoding depends on the type of the data referenced by index and sub-index.</p>	not used, always 0	Reserved for further use, always 0

Abort SDO transfer protocol:

SDO tasks which the MIS motors (or SMC66/85) cannot process are responded to using an abort SDO protocol. If the module does not respond in the expected time, the CANopen master also sends an abort SDO.

The following figure shows an abort SDO transfer protocol:



There are various abort codes in CANopen. These are listed in the table below:

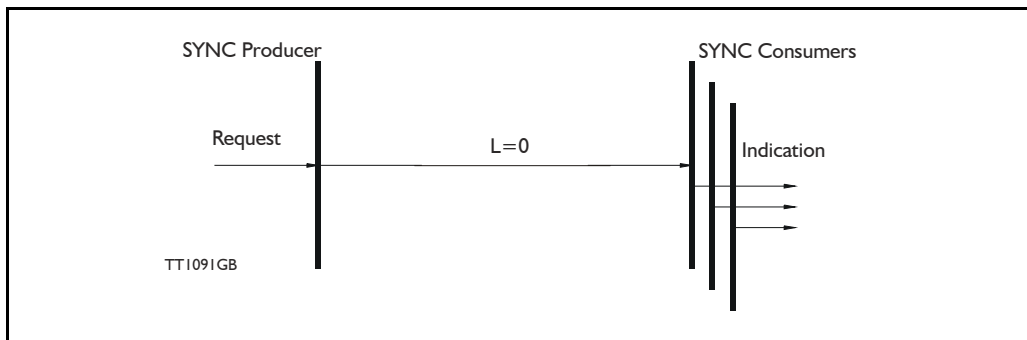
<b>Abort code</b>	<b>Description</b>
0503 0000h	Toggle bit not alternated
0504 0000h	SDO Protocol timed out
0504 0001h	Client/server command specified not valid or unknown
0504 0002h	Invalid block size (block mode only)
0504 0003h	Invalid sequence number (block mode only)
0504 0004h	CRC error (block mode only)
0504 0005h	Out of memory
0601 0000h	Unsupported access to an object
0601 0001h	Attempt to read a write-only object
0601 0002h	Attempt to write a read-only object
0602 0000h	Object does not exist in the object dictionary
0604 0041h	Object cannot be mapped to the PDO

## 11.7 More details of CANopen Theory

Abort code	Description
0604 0042h	The number and length of the objects to be mapped would exceed PDO length
0604 0043h	General parameter incompatibility reason
0606 0000h	Access failed due to a hardware error
0607 0010h	Data type does not match, length of service parameter does not match
0607 0012h	Data type does not match, length of service parameter too high
0607 0013h	Data type does not match, length of service parameter too low
0609 0011h	Sub-index does not exist
0609 0030h	Value range of parameter exceeded (only for write access)
0609 0031h	Value of parameter written too high
0609 0032h	Value of parameter written too low
0609 0036h	Maximum value is less than minimum value
0800 0000h	General error
0800 0020h	Data cannot be transferred or stored to the application
0800 0021h	Data cannot be transferred or stored to the application because of local control
0800 0022h	Data cannot be transferred or stored to the application because of the present device state
0800 0023h	Object dictionary dynamic generation fails or no object dictionary is present (e.g. object dictionary is generated from file and generation fails because of a file error).

### 11.7.5 SYNC (Synchronisation Object)

A SYNC producer sends the synchronization object cyclically a broadcast telegram. The SYNC telegram defines the basic clock cycle of the network. The time interval of the SYNC telegram is set using the object Communication Cycle period (1006h). In order to obtain a precise (accurate) cycle between the SYNC signals, the SYNC telegram is sent with a high-priority identifier. This can be modified using the object (1005h). The SYNC transfer applies the producer/consumer push model and is non-confirmed.



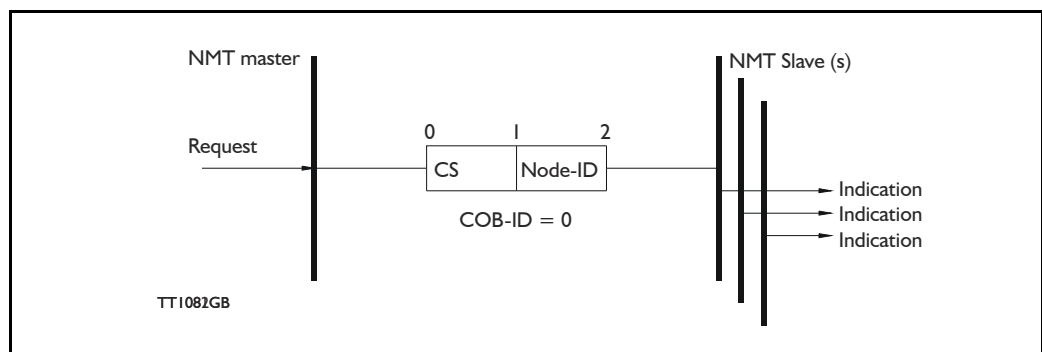
The SYNC does not carry any data ( $L=0$ ). The identifier of the SYNC object is located at object 1005h.

# 11.7 More details of CANopen Theory

## 11.7.6 NMT (Network Management services)

The Network Management is structured according to nodes and follows a master-slave structure. NMT objects are used for executing NMT services. Through NMT services, nodes are initialised, started, monitored, reset or stopped. All nodes are regarded as NMT slaves. An NMT slave is uniquely identified in the network by its Node-ID. NMT requires that one device in the network fulfils the function of the NMT master. The NMT master controls the state of the NMT slaves. The state attribute is one of the values (Stopped, Pre-operational, Operational, Initialising). The module control services can be performed with a certain node or with all nodes simultaneously. The NMT master controls its own NMT state machine via local services which are implementation dependent. The Module Control Service, except Start Remote Node, can be initiated by the local application.

A general NMT protocol:



Where **CS** is the NMT command specified. The Node-ID of the NMT slave is assigned by the NMT master in the Node Connect protocol, or 0. If 0, the protocol addresses all NMT slaves.

CS =	Operation
1	Start Remote Node
2	Stop Remote Node
128	Enter Pre Operational
129	Reset Node
130	Reset Communication

### Start Remote Node:

This is an instruction for transition from the Pre-Operational to Operational communications state. The drive can only send and receive process data when it is in the Operational state.

### Stop Remote Node:

This is an instruction for transition from either Pre-Operational to stopped or from Operational to Stopped. In the stopped state, the nodes can only process NMT instructions.

### Enter Pre Operational:

This is an instruction for transition from either Operational or Stopped state to Pre-Operational. In the Pre-Operational state, the node cannot process any PDOs. However, it can be parameterized or operated via SDO. This means set point can also be entered.

## 11.7 More details of CANopen Theory

---

### Reset Node:

This is an instruction for transition from the Operational, Pre-Operational or Stopped states to Initialization. After the Reset Node instruction, all objects (1000h-9FFFh) are re-set to the Voltage On stage.

### Reset Communication:

This is an instruction for transition from Operational or Stopped to Initialization. After the Reset Communication instruction, all communication objects (1000h-1FFFh) are re-set to the initial state.

In the various communication states, nodes can only be accessed via CANopen using specific communication services. Further, the nodes in the various states only send specific telegrams. This is clearly shown in the following table:

	Initializing	Pre-Operational	Operational	Stopped
PDO			X	
SDO		X	X	
Synchronization Object		X	X	
Emergency Object		X	X	
Boot-Up Object	X			
Network Management object		X	X	X

### 11.7.7 Error Control Services

Two possibilities exist for performing Error Control:

- Node Guarding/Life Guarding
- Heartbeat

#### Node Guarding/Life Guarding

With Node Guarding, the CANopen master sends each slave an RTR telegram (Remote Transmit request) with the COB-ID 1792 (700h) + node-ID.

Using the same COB-ID, the slave responds with its communications state, i.e. either Pre-Operational, Operational or stopped.

The CANopen slave also monitors the incoming RTR telegram from the master.

The cycle of the incoming RTR telegrams is set using the Guard Time Object.

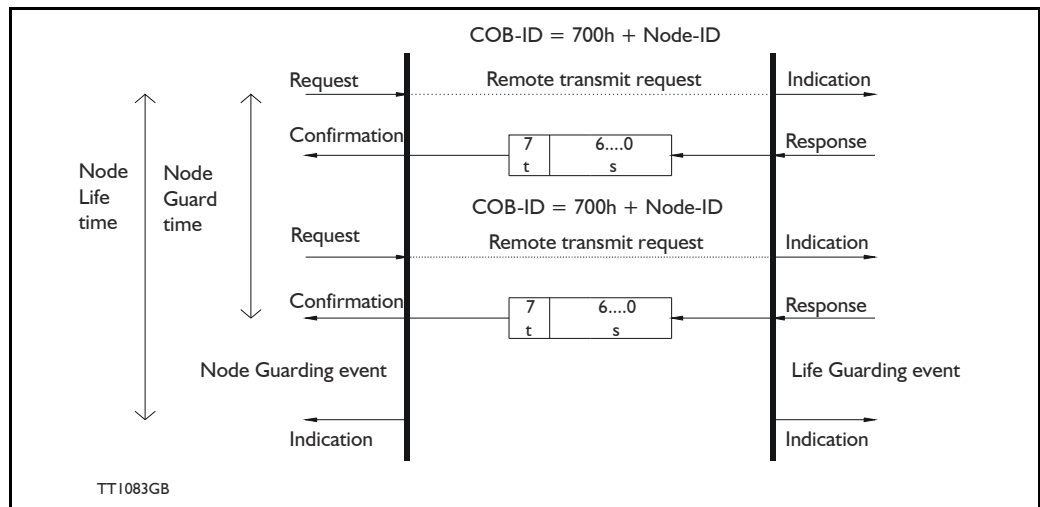
The number of RTR telegrams which can fail (at a maximum) before the slave initiates a Life Guarding event is defined using the Life time factor object.

The Node Life Time is calculated from the product of the Guard Time and Life Time Factor. This is the maximum time that the slave waits for an RTR telegram.

The figure below shows a Node Guarding/Life Guarding protocol.



## 11.7 More details of CANopen Theory



Where s is the state of the NMT slave:

s	NMT state
4	Stopped
5	Operational
7	Pre-operational

**t:** is the toggle bit. It alternates between 2 consecutive responses from the NMT Slave. The value of the toggle-bit of the first response after the guarding protocol becomes active is 0.

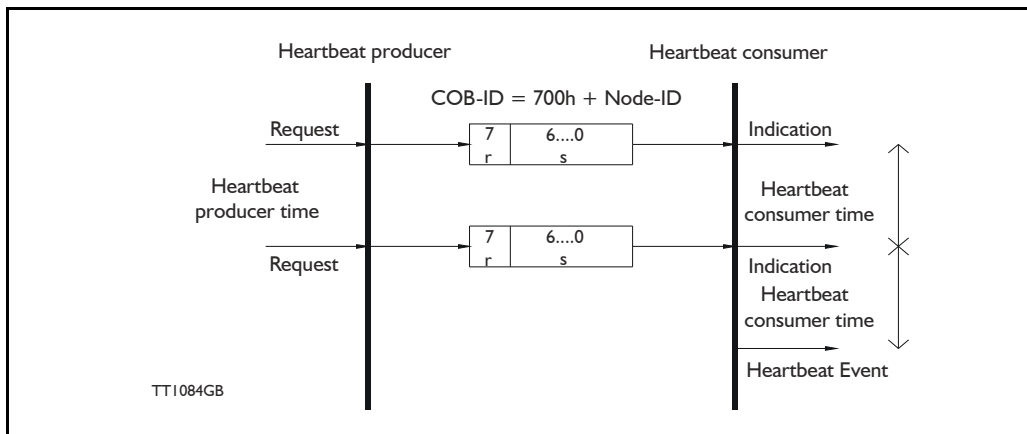
The Toggle Bit in the guarding protocol is only reset to 0 when the NMT message Reset Communication is passed (no other change of state resets the toggle bit).

If a response is received with the same value of the toggle-bit as in the preceding response, then the new response is handled as if it was not received.

### Heartbeat:

With the Heartbeat protocol, a Heartbeat Producer cyclically sends its communications state to the CAN bus. One or more Heartbeat Consumers receive the indication. The relationship between producer and consumer is configurable via the object dictionary. The Heartbeat Consumer guards the reception of the Heartbeat within the Heartbeat Consumer time. If the Heartbeat is not received within the Heartbeat Consumer Time, a Heartbeat Event will be generated.

# 11.7 More details of CANopen Theory



Where r is reserved (always 0).  
s: is the state of the Heartbeat producer:

s	NMT state
0	Boot up
4	Stopped
5	Operational
7	Pre-operational

Only one communication monitoring service may be activated. This is either Node Guarding/Life Guarding or Heartbeat. If the Heartbeat Producer Time is configured on a device, the Heartbeat Protocol begins immediately. If a device starts with a value of the Heartbeat Producer Time different from 0, the Heartbeat Protocol starts with the state transition from Initialising to Pre-operational. In this case the Bootup Message is regarded as the first heartbeat message. If the Heartbeat producer time is not 0, the heartbeat protocol is used.

In the MIS motors (or SMC66/85), none of the error control mechanisms is enabled when the modules are started up, because if there is any fault in the system it is impossible to contact the module. After the module has started up and there is communication between the master and the slave, activate the required error control mechanism in the object Dictionary. See [DS301 specified Communications objects](#), page 242.



The Modbus implementation is a subset of the Modbus Specification V1.1b. This standard can be downloaded free of charge from the website [www.modbus.org](http://www.modbus.org).

Also you may want to download and read the [Modbus Serial Line Protocol and Implementation Guide V1.02](#), that describes many aspects of the signals, and the details of using and inter-connecting two-wire RS-485.

The serial communications lines normally used for communications between the motor and MacTalk can be configured to use the Modbus protocol instead of the standard MacTalk protocol.

The MISxxx firmware supports the two command types Read Holding Registers (3) and Write Multiple Register (0x10). All other commands will result in Exception replies (exception type 1, Illegal Function).  
Use firmware version 4.00 or later.

All registers can be read as well as written over Modbus, but the number of registers per transfer is limited to 16 16-bit registers or 8 32-bit registers.  
Contact JVL if more registers are needed in a single transfer.

JVL recommends reading and writing as few 32-bit values as possible, normally 4 or maximum 8 at a time. Whenever possible, split long commands into smaller commands. Also, only write the absolutely necessary values to the motor.

All registers in the MISxxx are 32-bits. 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.

The address space is mapped to offset 40000, though it is also possible to write to register 3, P\_SOLL using the address 40006. It is not necessary to do anything to choose between offset 0 and offset 40000, just read or write to desired address.

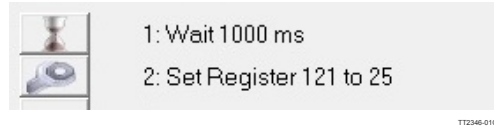
The setup of the Modbus protocol is done by writing to register 121, called ModbusSetup.  
The motor always starts up with the MacTalk protocol enabled. This is to always make it possible for a user to re-connect with MacTalk, if MacTalk is running at the time of a motor power up cycle.

The standard way of switching to Modbus is to write a value to Register 121, Modbus-Setup, that has bit 0 set to one. After a delay where there is no MacTalk communications, the motor will switch to Modbus. Note that MacTalk cannot use the Modbus protocol, and so cannot be used to configure the motor, after Modbus has been activated.  
That is, until the next power cycle, as described above.

## 12.1

# Modbus

Often the graphical program in the motor is used to write to Register 121, ModbusSetup, possibly after an initial delay, or as the result of a hardware input.



Set Register 121 to 25 (8 data bits, Even Parity, 1 stop bit).

The baudrate is unchanged after the switch from MacTalk to Modbus. It is recommended to first configure the baudrate using MacTalk, and save this to flash memory.

Register 121, ModbusSetup, supports the following bit-fields (default values marked with \*) - see also [Modbus\\_setup](#), page 173

Bits	Values	Description
0	0=Disabled *, 1=Enabled	When set to non-zero, selects to start the switch away from the MacTalk protocol and start the Modbus protocol.
1	0=Modbus RTU*, 1=Modbus ASCII	This field is not current used. The motor always uses the binary Modbus RTU format.
3:2	0=No parity, 1=Odd parity, 2=Even parity	Select the parity bit value. Select the same format as the Modbus client PLC, HMI or PC uses. (3:2 is read as 'bits 3 through 2')
4	0=7 data bits, 1=8 data bits	Number of data bits in a byte. Modbus RTU always uses 8 bits per byte.
5	0=1 stop bit, 1=2 stop bits	Select the number of stop bits. Select the same format as the Modbus client PLC, HMI or PC uses
31:6	Reserved	Reserved for future use. Please set all of these bits to zero.

Below is the basic data format for the two supported operations 3 and 16 (0x10 hexadecimal):

Read Holding operation:

Request: <adr>, 0x03, RegHi, RegLo, CountHi, CountLo, CRC1, CRC2

Offset: [0] [1] [2] [3] [4] [5] [6] [7]

Reply: <adr>, 0x03, #Bytes, Reg0Hi, Reg0Lo, Reg1Hi, Reg1Lo, ..... CRC1, CRC2

Example to read P\_IST (register 10) from a motor with address 254, values in decimal: 254, 3, 0, 20, 0, 2, NN, MM (NN and MM are the CRC-16 bytes)

Write Multiple Register operation:

Request: <adr>, 0x10, RegHi, RegLo, CountHi, CountLo, NBytes, Val0Hi, Val0Lo, ..., CRC1, CRC2

Offset: [0] [1] [2] [3] [4] [5] [6] [7] [8]

Reply: <adr>, 0x10, RegHi, RegLo, CountHi, CountLo, CRC1, CRC2

Example to write P\_SOLL (register 3) to motor with address 254, values in decimal: 254, 16, 0, 6, 0, 2, 4, bb, aa, dd, cc, NN, MM (NN and MM are the CRC-16 bytes)

## 12.1

## Modbus

---

This would write a 32-bit hexadecimal value of ddcbbbaa - note the byte-packing.

Example to write the value 999888 to P\_SOLL (register 3) using offset 40000, to the motor with address 254. Values in decimal:

Value = 999888,

High Word = 15, High Word - High byte = 0, High Word - Low byte = 15

Low Word = 16848, Low Word - High byte = 65, Low Word - Low byte = 208

Address =  $40000 + 2 * 3 = 40006$

High byte = 156, Low byte = 70

Note, that some implementations of the Modbus requires an offset added to the address, eg. to write to P\_SOLL (register 3) use the address  $2 * 3 + 1 = 7$ , or  $40000 + 2 * 3 + 1 = 40007$ . This refers only to the master. It should generate the same command, as below.

The word order for 32-bit values is Low Word, High Word, and the byte order for 16-bit values is High byte, Low byte.

Command:

254, 16, 156, 70, 0, 2, 4, 65, 208, 0, 15, 232, 101

Response:

254, 16, 156, 70, 0, 2, 154, 66



## 13.1 Step motor controllers (SMC66/85)

Sorry - Pictures soon coming

The compact step motor controller SMC66 and SMC85 are designed for positioning and speed control of stepper motors. SMC66 is a PCB with dimensions 34x65 mm and SMC85 is 78x86mm.

Both PCB's are used in the MIS motors, forming a complete integrated step motor. It may also be used with other types of step motors according to customers requirements. The basic features of the controller are overall similar to the MIS motors feature list:

- Serial RS485 or 5V serial position controller.
- Build-in mini PLC with graphic programming.
- Option for CANbus, CANopen DS-301. Fully ISO 11898-2:2016 compliant/(DSP-402 in development).
- Options for EthernetIP, Profinet, Powerlink, ModbusTCP, SercosIII and EtherCAT.
- A dual supply facility is available so that position and parameters are maintained at emergency stop.
- Electronic Gear mode.
- MACmotor protocol so MAC servomotors and MIS stepper motors can be connected on the same RS485 bus.
- Command for easy PLC/PC setup and communication.
- Power supply 12-72 VDC.
- Extremely high torque vs speed - up to 3000 RPM with good performance.
- Fixed 409600 steps per revolution
- Built-in 32Bit  $\mu$ processor with 8 In/Out that can be configured as inputs, PNP outputs or analogue inputs. 5V serial and RS485 interface for set up and programming.
- MODBUS interface.
- 9.6kbit/sec. to 1Mb/sec. communication.

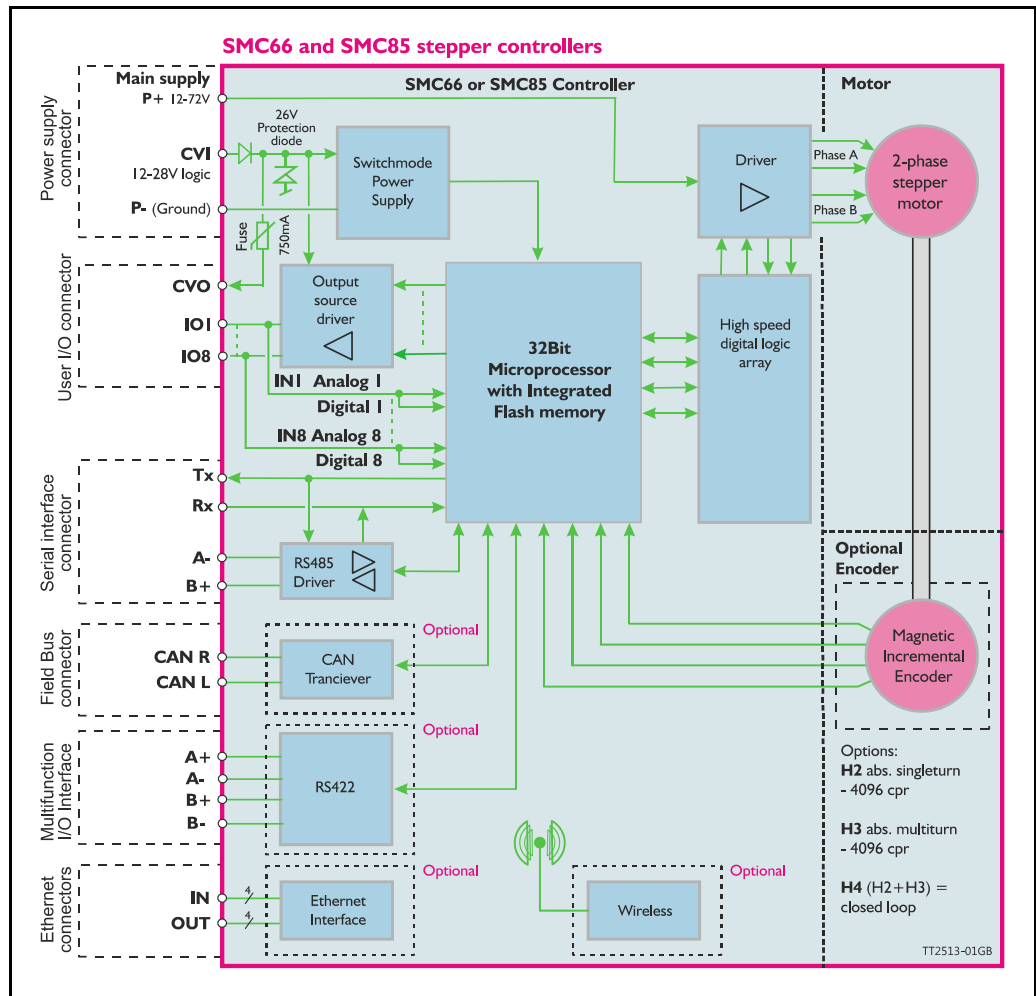
Benefits when using the SMC66 and SMC85 controllers:

- De-central intelligence.
- Simple installation. No cables between motor and controller/driver.
- EMC safe. Switching noise remains within motor.
- Compact. Does not take space in the control cabinet.
- Option: Closed loop feature by means of magnetic encoder with resolution of up to 4096 pulses/rev. (H2 or H4 option)
- Option: Absolute multi turn encoder for keeping the position permanent also during power down. (H3 or H4).
- Vibration tested at 4G in 3 axis and shock tested at 15G in 3 axis according to IEC60068.
- Interface possibilities:
- From PC/PLC with serial commands via 5V serial or RS485.
- Pulse/direction input. Encoder output.
- CANopen.
- 8 I/O, 5-28VDC that can be configured as Inputs, Outputs or analogue inputs.
- Wireless options: WiFi, Bluetooth and Zigbee.



# 13.1 Step motor controllers (SMC66/85)

## 13.1.1 Block diagram, Positioning/Speed Control



# 13.1 Step motor controllers

## (SMC66/85)

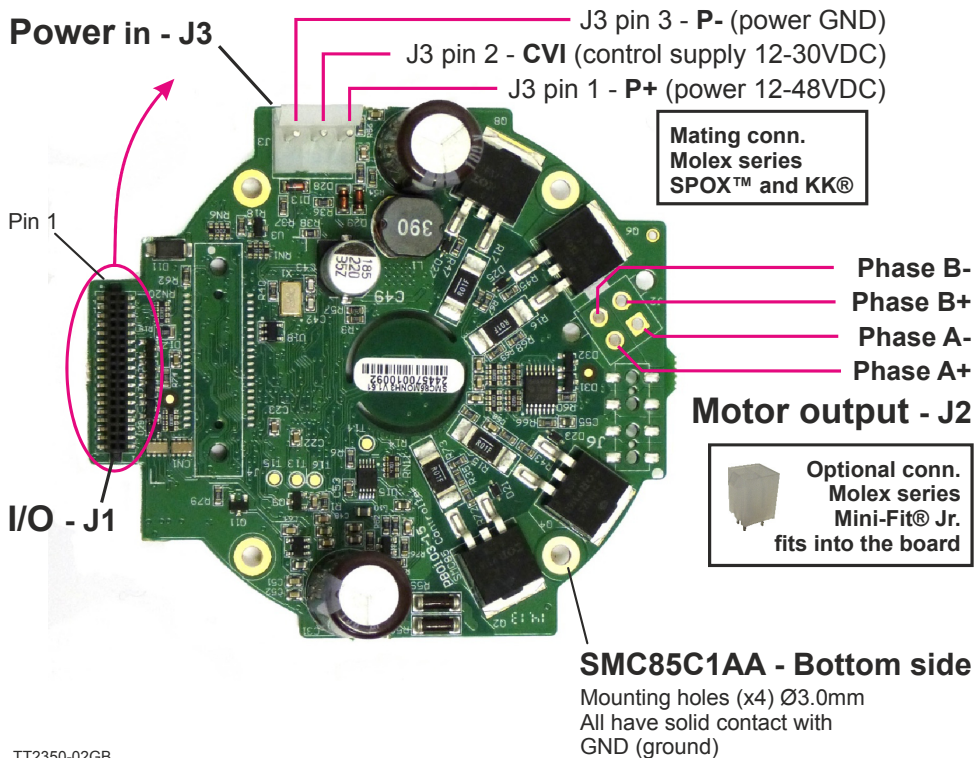
Only SMC85

### 13.1.2 SMC85 Connector overview

The connections to the various connectors of the SMC85 PCB board is shown below. Note that GND and P- are connected together internally.

#### Connector J1 - pin description

<b>GND</b> (ground for I/O's same as P-)	1	●	2	<b>Do not connect !</b>
<b>GND</b> (ground for I/O's same as P-)	3	●	4	<b>Do not connect !</b>
<b>GND</b> (ground for I/O's same as P-)	5	●	6	<b>Do not connect !</b>
<b>GND</b> (ground for I/O's same as P-)	7	●	8	<b>Do not connect !</b>
<b>GND</b> (ground for I/O's same as P-)	9	●	10	<b>Do not connect !</b>
<b>Not used - do not connect</b>	11	●	12	(Conn. internally to CVI) <b>CVO</b>
<b>+5V out - max. 50mA!</b>	13	●	14	(I/O channel 1) <b>IO1</b>
<b>RX</b> (RS232 receive - 3.3V !)	15	●	16	(I/O channel 2) <b>IO2</b>
<b>TX</b> (RS232 transmit - 3.3V !)	17	●	18	(I/O channel 3) <b>IO3</b>
<b>CAN_H</b> (optional)	19	●	20	(I/O channel 4) <b>IO4</b>
<b>CAN_L</b> (optional)	21	●	22	(I/O channel 5) <b>IO5</b>
<b>RS485 A-</b> (RS485 setup interface)	23	●	24	(I/O channel 6) <b>IO6</b>
<b>RS485 B+</b> (RS485 setup interface)	25	●	26	(I/O channel 7) <b>IO7</b>
<b>A1+</b> (RS422)	27	●	28	(I/O channel 8) <b>IO8</b>
<b>A1-</b> (RS422)	29	●	30	<b>Do not connect !</b>
<b>B1+</b> (RS422)	31	●	32	(H2 enc. output optional) <b>EA</b>
<b>B1-</b> (RS422)	33	●	34	(H2 enc. output optional) <b>EB</b>
<b>GND</b> (ground for I/O's same as P-)	35	●	36	(H2 enc. output optional) <b>EI</b>





# 14.1

# MIS23x Technical Data

Only MIS23x

<b>Main Supply Voltage (P+)</b>	Voltage Range	Nominal +12-72VDC (absolute max. = 90VDC ripple free). Min. voltage 8VDC (without ripple)				
	Current consumption	Power supply current requirements = minimum 3A recommended. Please refer to the power supply chapter. The actual power supply currents will depend on voltage and load.				
<b>Control Voltage (CVI)</b>	Range	+12 to +28VDC. Supply current 50-130mA@24VDC (depends which options installed) Supply for the internal control circuitry, the output driver (IO1-8), and feed-back circuits (if present). If the motor connected or passive mode: 100mA. Note: Battery supply 12VDC is also possible. The function of the motor is not affected before supply is below VDC. Please make sure that no voltages below this point is present since the processor will reset/restart if "dips" exist at the supply.				
<b>Mechanical</b>	Motor type:	<b>MIS231S/Q</b>	<b>MIS231T/R</b>	<b>MIS232S/Q</b>	<b>MIS232T/R</b>	<b>MIS234S/Q</b>
	Holding torque Nm [Oz-In]	0.97 [137.4]	1.16 [164.3]	1.97 [279.0]	2.53 [358.3]	3.08 [436.2]
	Inertia kgcm <sup>2</sup> [Oz-In-Sec <sup>2</sup> ]	0.3 [0.00423]	0.3 [0.00423]	0.48 [0.00677]	0.48 [0.00677]	0.65 [0.0092]
	Weight - kg [lb]	1.1 [2.43]	1.1 [2.43]	1.4 [3.09]	1.4 [3.09]	2.0 [4.41]
	Max. axial shaft force N	15	15	15	15	15
	Max. radial shaft force (N) (applied 20 mm from flange)	75	75	75	75	75
<b>Analogue Input</b>	Resolution	12 Bit				
	Voltage Range	0 to +5.00 VDC				
<b>General Purpose I/O</b>	Number/Type	8 Sources of output or input				
	Logic Range	Inputs and Outputs tolerant to +24VDC. Inputs TTL level compatible				
	Output Source Current	Up to 300 mA per output.				
	Protection	Over Temp. Short Circuit. Transient. Over Voltage. Inductive Clamp.				
	Input Filter	0.1 or 1 to 100 ms				
<b>Communication</b>	Type (Standard)	RS485				
	Type (Optional)	RS422				
	Baud Rate	9.6 to 921.6 kbps				
	Type (Optional)	CANopen DS301 (VS3.0), 2.0A Active. Ethernet: Powerlink, EthernetIP, Profinet, EtherCAT, SercosIII, ModbusTCP				
	Isolation	RS485/RS422/CANopen : None / Ethernet : Yes - withstand up to 500VDC in potential difference.				
	Features	Node Guarding, heartbeat, SDOs, PDOs (Dynamic mapping)				
<b>Motion</b>	Open Loop operation	Operation modes	Passive, Position, Gear, Velocity			
		Resolution per rev.	409600 counts			
	Internal Encoder (option: <b>H2</b> )	Type	Internal, magnetic, absolute 1 rev. Closed loop ready.			
		Resolution per rev.	4096 counts / 1024 lines (quadrature output)			
	Internal Encoder (option: <b>H3</b> )	Type	Internal, magnetic, absolute multiturn.			
		Resolution per rev.	Displayed: 409600 counts - internal: 1024 counts			
	Internal Encoder (option: <b>H4</b> )	Type	Internal, magnetic, absolute multiturn Closed loop ready.			
		Resolution per rev.	Displayed: 409600 counts - internal: 4096 counts			
	Counters	Type	Position, Encoder / 32 Bit			
		Edge Rate (Max.)	12.0 MHz			
	Velocity	Range	-3000.00 to +3000.00 RPM			
		Resolution	0.01 RPM			
		Precision	±50 ppm			
Accel./Deceleration.	Range	1 - 500000 RPM/s				
	Resolution	1 RPM/s				
Electronic Gearing	Range/Resolution (External Clock/encoder In)	Input (ext.) / Output (int.) = 1/409600 up to 409600/1				
<b>Software</b>	Program Storage	Type/Size	Flash 3072 Bytes			
	User Registers	2248 Bytes/32 bits				
	User Program Variables	Up to 224				
	Math Functions	+, -, x, /, >, <, =, <=, >=, AND, OR, XOR, NOT, !, &, ^.				
	Branch Functions	Branch & Call				
	General Purpose I/O Functions	Inputs	Home, Limit Plus, Limit Minus, Analogue In, General Purpose			
		Outputs	Moving, Fault, general Purpose			
	Party Mode Addresses	254				
Encoder Functions (options)	Stall Detection, Position maintenance, Find Index, Closed loop, Absolute Multiturn encoder					
<b>Thermal</b>	Operating/storage temp.	Ambient 0 to +40°C (32-104°F)/ -20 to +85°C. (-4 to 185 °F) (Humidity 90%). A warning message is generated if the internal temperature passes 80°C The motor is set in passive mode if the temperature passes 90°C and an error message is generated.				

# 14.2

# MIS34x Technical Data

Only MIS34x

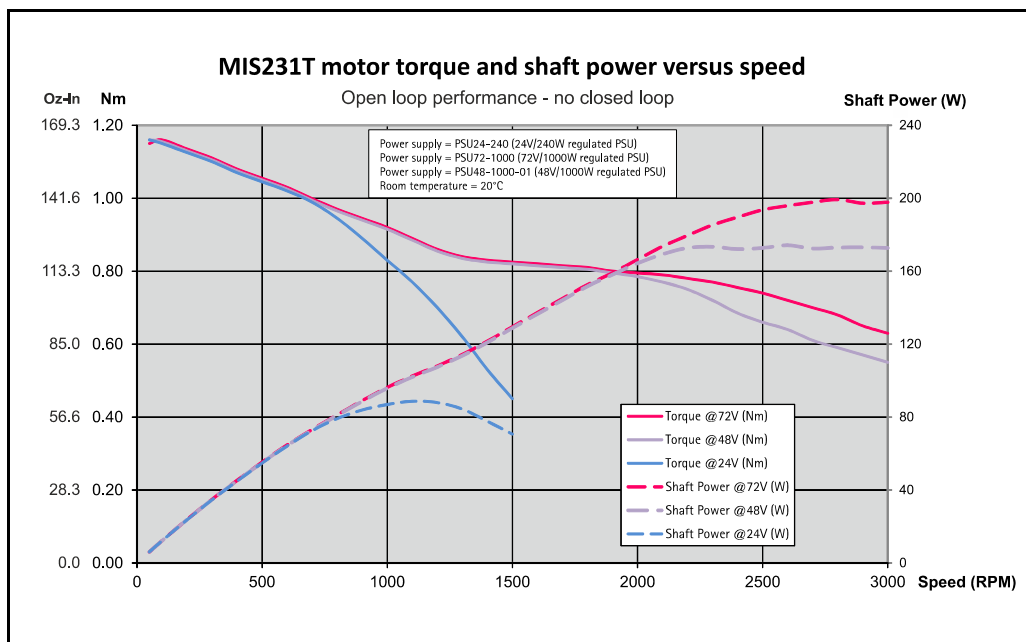
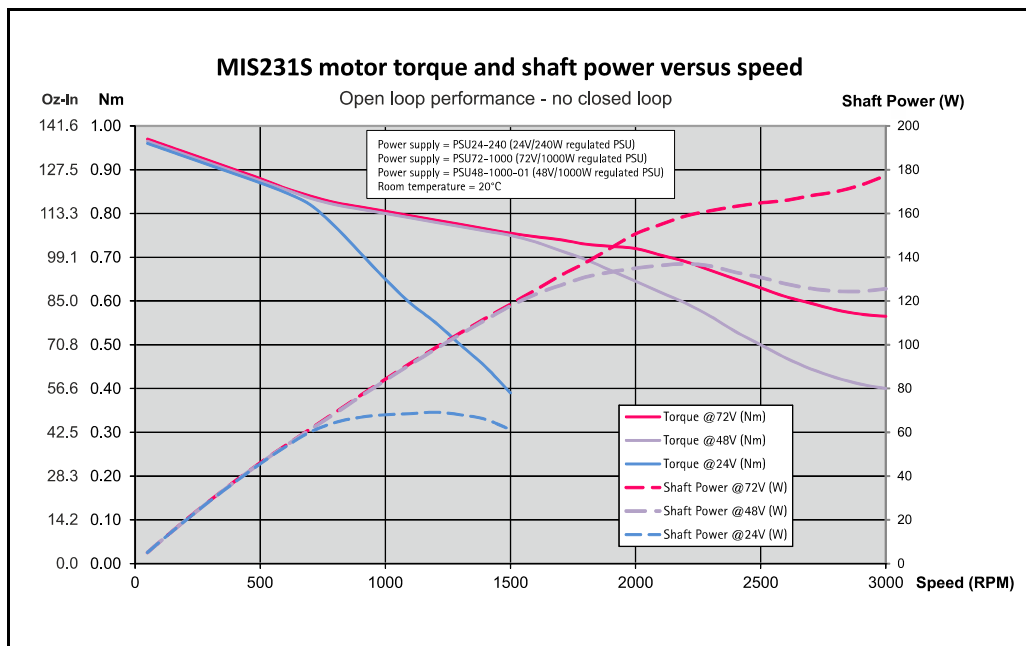
<b>Main Supply Voltage (P+ terminal)</b>	Voltage Range	Nominal +12-72VDC (absolute max. = 90VDC ripple free). Min. voltage 8VDC (without ripple)		
	Current consumption	Power supply current requirements = 10ARMS (max.). Actual power supply currents will depend on voltage and load. When motor is in "Passive" mode the current consumption is < 10mARMS@24VDC supply		
<b>Control Voltage (CVI terminal)</b>	Voltage Range	Control voltage input to maintain power to the internal control circuitry including output driver and feed-back circuits and interface (all except the motor power for turning the motor). Nominal voltage range 12-28VDC. Minimum 8VDC and Maximum 32VDC.		
	Current consumption	Typical 45mA@24.0VDC when motor is in passive mode and none of the I/O's are connected.		
<b>Mechanical</b>		<b>MIS340</b>	<b>MIS341</b>	<b>MIS342</b>
	Holding torque - Nm [Oz-In]	3.0 [424]	6.1 [863]	9.0 [1274]
	Inertia - kgcm <sup>2</sup> [Oz-In-Sec]	1.4 [0.0198]	2.7 [0.0381]	4.0 [0.0564]
	Weight - kg [lb]	2.05 [4.52]	3.13 [6.9]	4.2 [9.26]
	Max. axial shaft force	115N		
	Max. radial shaft force	180N applied 12.5mm from shaft end		
<b>Analogue Input</b>	Resolution	12 Bit		
	Voltage Range	0 to +5VDC		
<b>General Purpose I/O</b>	Number/Type	8 Sources of output or input		
	Logic Range	Inputs and Outputs tolerant to +24VDC. Inputs TTL level compatible		
	Output Source Current	Up to 300 mA per output.		
	Protection	Over Temp. Short Circuit. Transient. Over Voltage. Inductive Clamp.		
	Input Filter	0.1 or 1 to 100 ms		
<b>Communication</b>	Type (Standard)	RS485		
	Type (Optional)	RS422		
	Baud Rate	9.6 to 921.6 kbps		
	Type (Optional)	CANopen DS301 (VS3.0), 2.0A Active. Ethernet: Powerlink, EthernetIP, Profinet, EtherCAT, SercosIII, ModbusTCP		
	Isolation	RS485/RS422/CANopen : None / Ethernet : Yes - withstand up to 500VDC in potential difference.		
	Features	Node Guarding, heartbeat, SDOs, PDOs (Dynamic mapping)		
<b>Motion</b>	Open Loop operation	Operation modes	Passive, Position, Gear, Velocity	
		Resolution per rev.	409600 counts	
	Internal Encoder (option: <b>H2</b> )	Type	Internal, magnetic, absolute 1 rev. Closed loop ready.	
		Resolution per rev.	4096 counts / 1024 lines (quadrature output)	
	Internal Encoder (option: <b>H3</b> )	Type	Internal, magnetic, absolute multiturn.	
		Resolution per rev.	Displayed: 409600 counts - internal: 1024 counts	
	Internal Encoder (option: <b>H4</b> )	Type	Internal, magnetic, absolute multiturn Closed loop ready.	
		Resolution per rev.	Displayed: 409600 counts - internal: 4096 counts	
	Counters	Type	Position, Encoder/32 Bit	
		Edge Rate (Max.)	12.0 MHz	
	Velocity	Range	0.01 to 3000.00 RPM	
		Resolution	0.01 RPM	
		Precision	±50ppm	
Accel./Deceleration	Range	1 - 500000 RPM/s		
	Resolution	1 RPM/s		
Electronic Gearing	Range/Resolution (External Clock In)	Input (ext.) / Output (int.) = 1/409600 up to 409600/1		
<b>Software</b>	Program Storage	Type/Size	Flash 3072 Bytes	
	User Registers	2248 Bytes/32 bits		
	User Program Variables	Up to 224		
	Math Functions	+, -, x, /, >, <, =, <=, >=, AND, OR, XOR, NOT, !, &, ^ .		
	Branch Functions	Branch & Call		
	General Purpose I/O Functions	Inputs	Home, Limit Plus, Limit Minus, Analogue In, General Purpose	
		Outputs	Moving, Fault, general Purpose	
	Party Mode Addresses	254		
Encoder Functions (options)	Stall Detection, Position maintenance, Find Index, Closed loop, Absolute Multiturn encoder			
<b>Thermal</b>	Operating/storage temp.	Ambient 0 to +40°C (32-104°F)/ -20 to +85°C. (-4 to 185 °F) (Humidity 90%). A warning message is generated if the internal temperature passes 80°C The motor is set in passive mode if the temperature passes 90°C and an error message is generated.		

# 14.3

# Torque Curves

## 14.3.1 MIS231S and MIS231T Torque and power curves

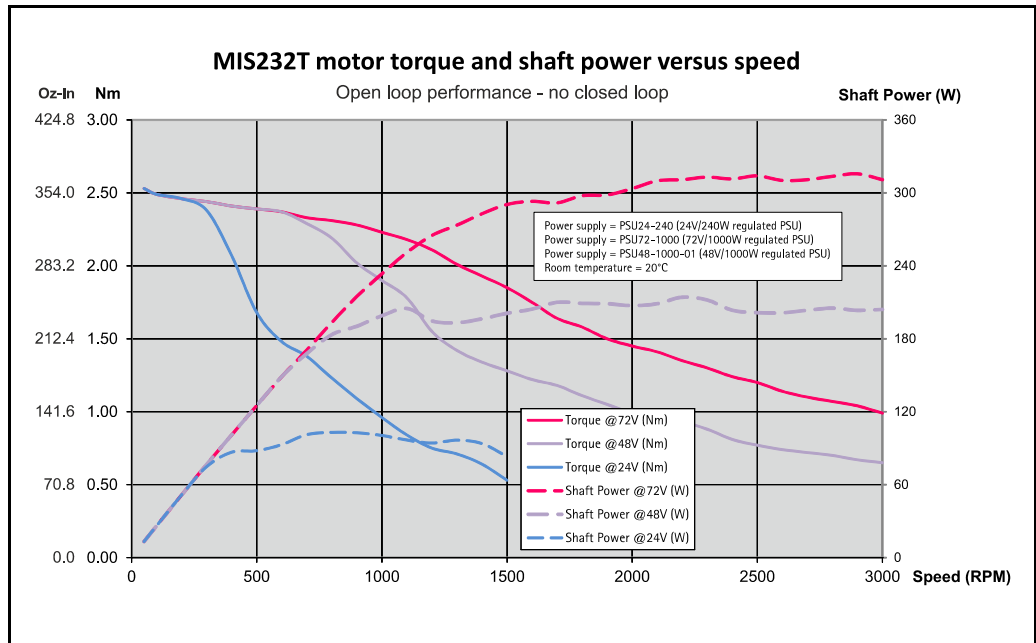
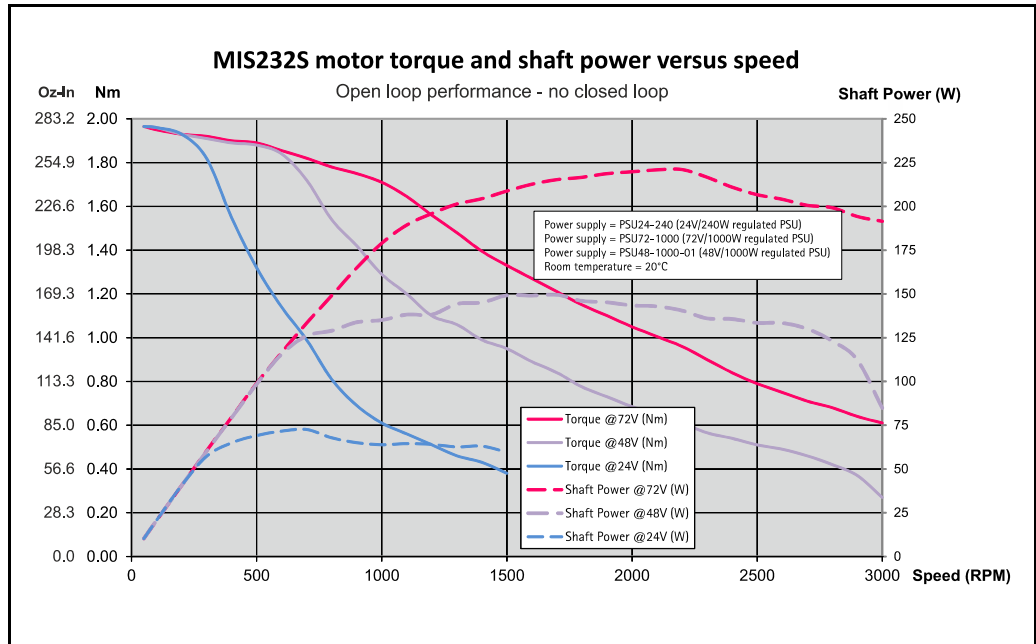
Below the torque performance for both motor families is shown. As it can be seen the supply voltage have a significant influence at the torque performance at higher speeds. Please make sure to use a supply voltage which is appropriate for the actual application. Also make sure that the supply voltage is stable without too much ripple since voltage dips can cause the motor to stall and loose position. Also shaft power (mechanical power at motor shaft) is shown.



# 14.3

# Torque Curves

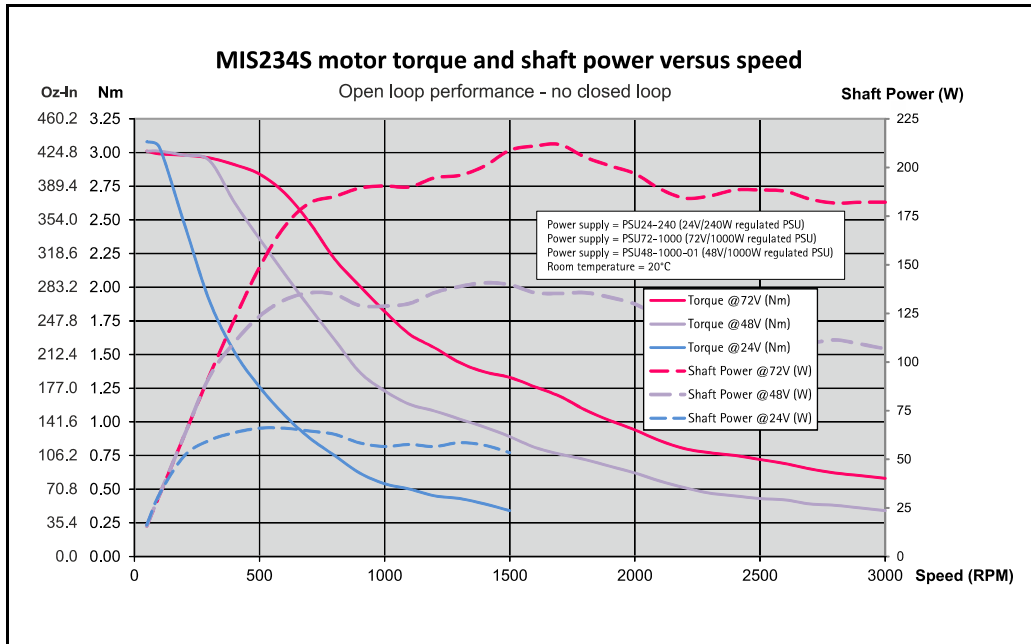
## 14.3.2 MIS232S and MIS232T Torque and power curves



# 14.3

# Torque Curves

## 14.3.3 MIS234S Torque and power curves



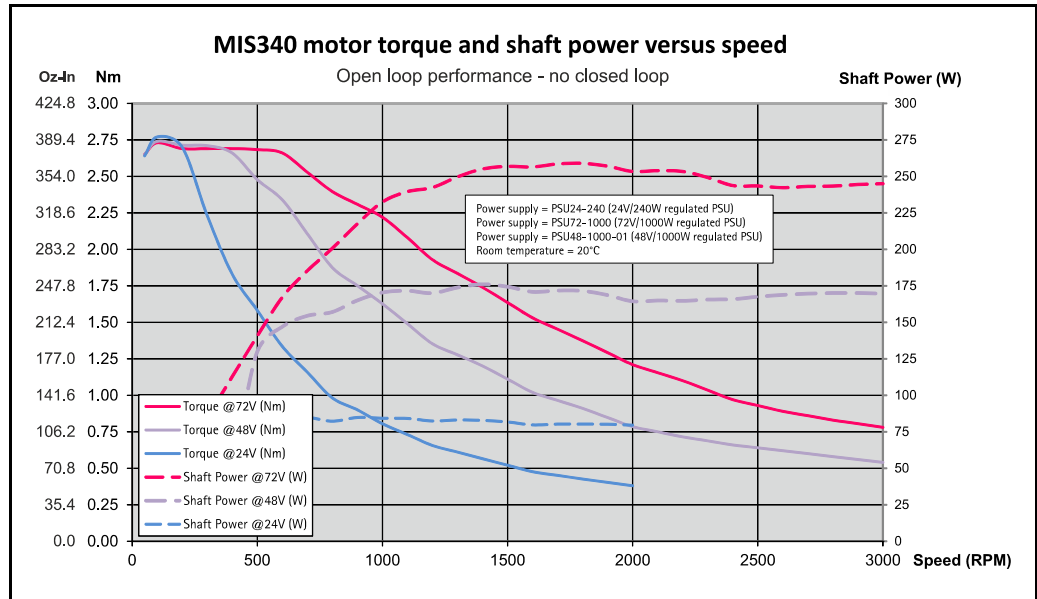


# 14.3 Torque Curves

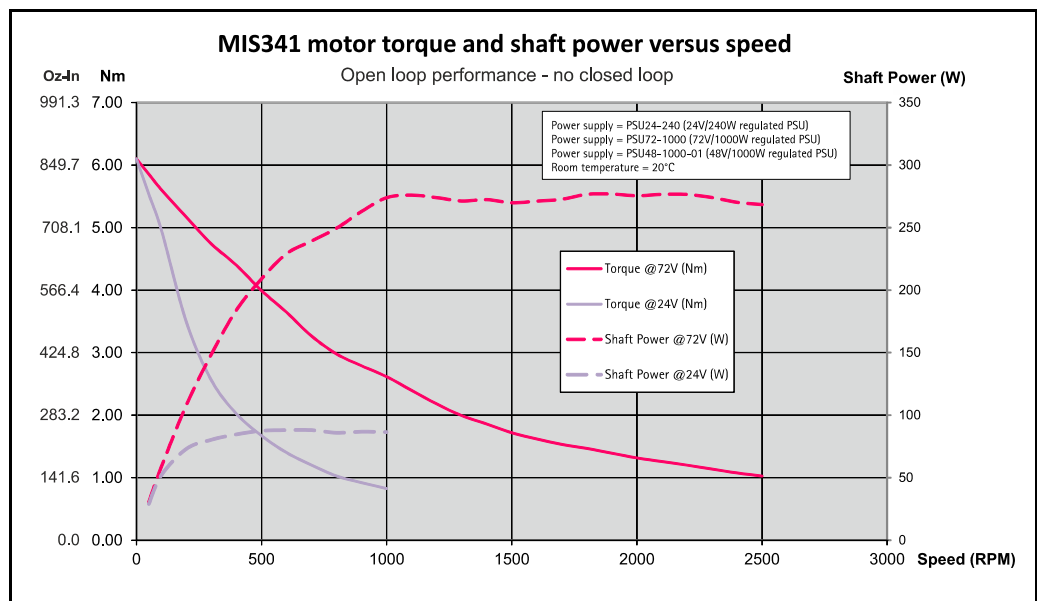
## 14.3.4 MIS34x Torque curves

Below the torque performance for both motor families is shown. As it can be seen the supply voltage have a significant influence at the torque performance at higher speeds. Please make sure to use a supply voltage which is appropriate for the actual application. Also make sure that the supply voltage is stable without too much ripple since voltage dips can cause the motor to stall and loose position.

## 14.3.5 MIS340 Torque curve



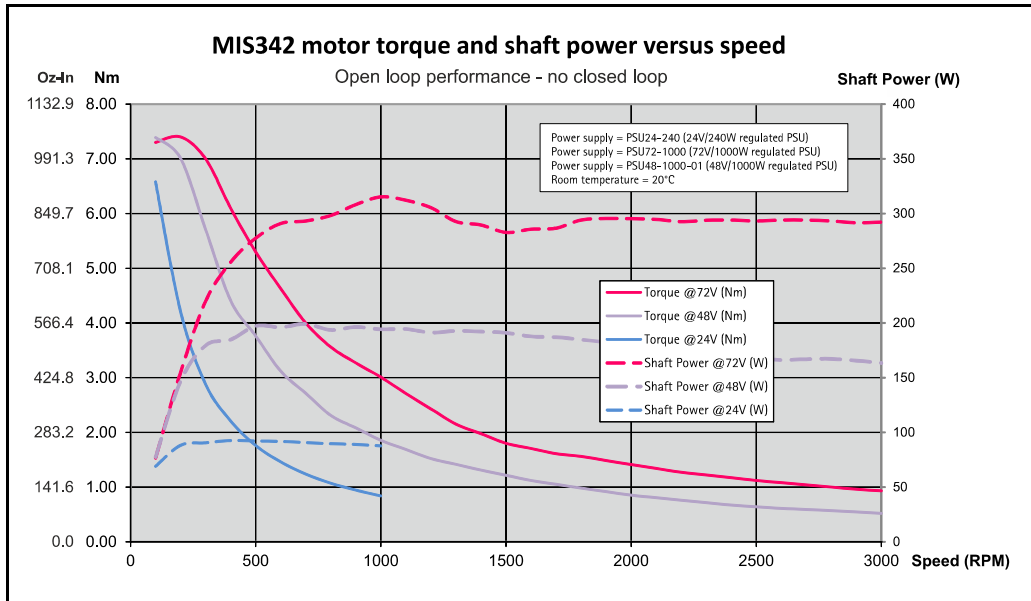
## 14.3.6 MIS341 Torque curve



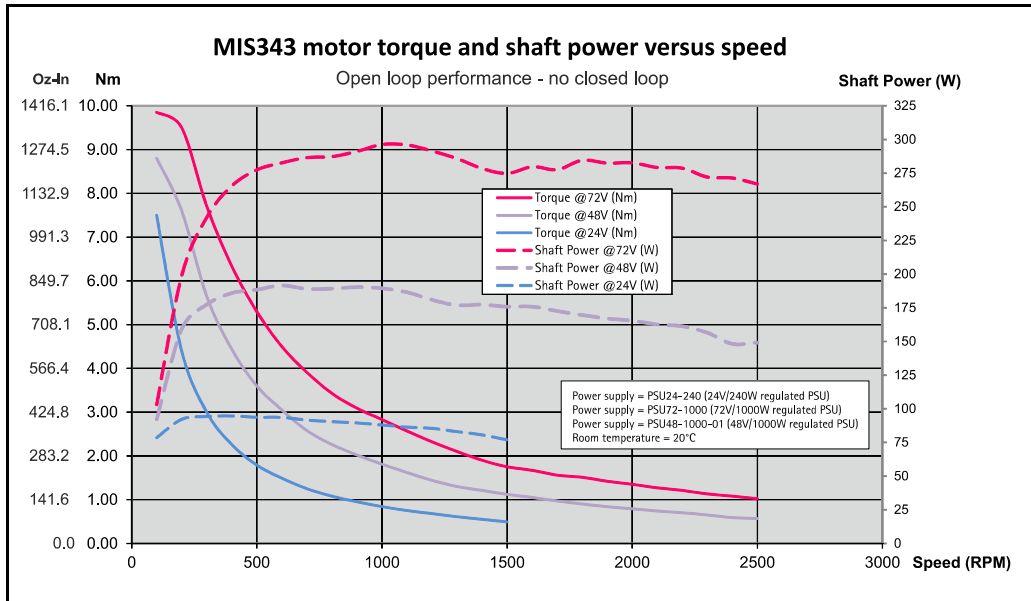
# 14.3

# Torque Curves

## 14.3.7 MIS342 Torque curve



## 14.3.8 MIS343 Torque curve

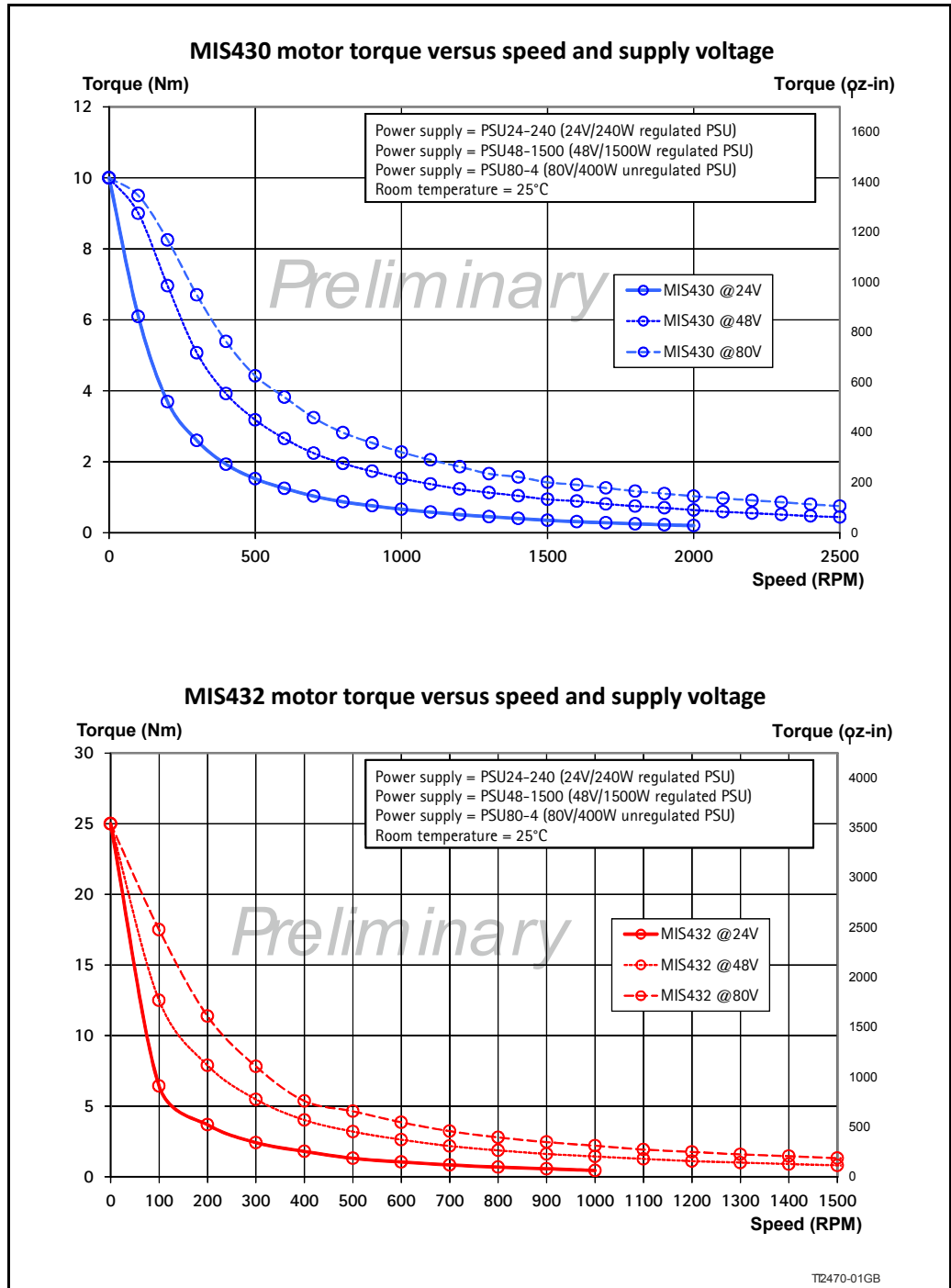


# 14.3

# Torque Curves

## 14.3.9 MIS430 and MIS432 Torque curves

Below the torque performance for the MIS43x families is shown. As it can be seen the supply voltage has a significant influence at the torque performance at higher speeds. Please make sure to use a supply voltage which is appropriate for the actual application. Also make sure that the supply voltage is stable without too much ripple since voltage dips can cause the motor to stall and loose position.



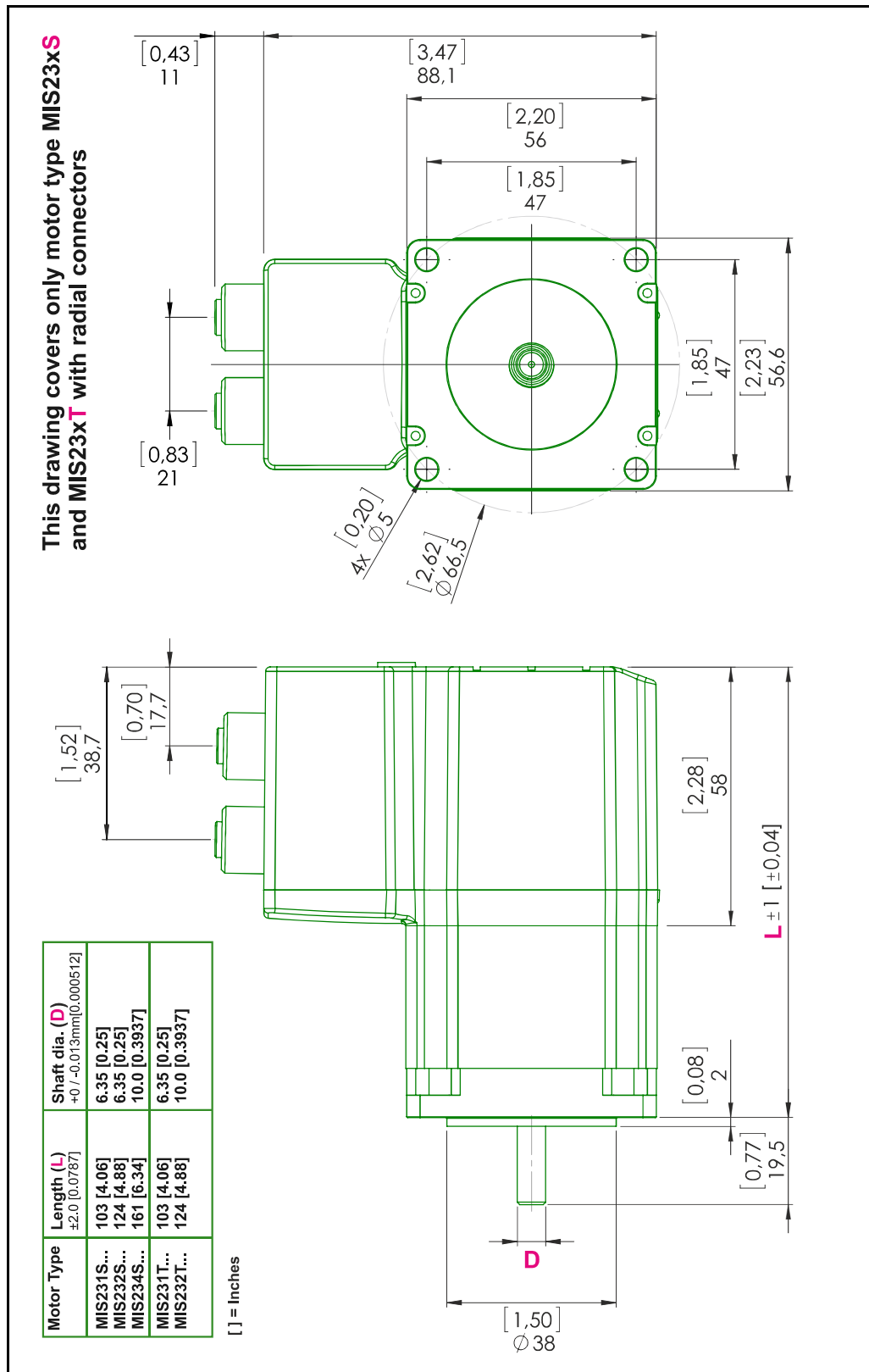
# 14.4

# Physical Dimensions

Only MIS23x

## 14.4.1 Physical dimensions MIS231S/T, MIS232S/T and MIS234S/T

Notice that this chapter only covers the MIS23x family generation 2.

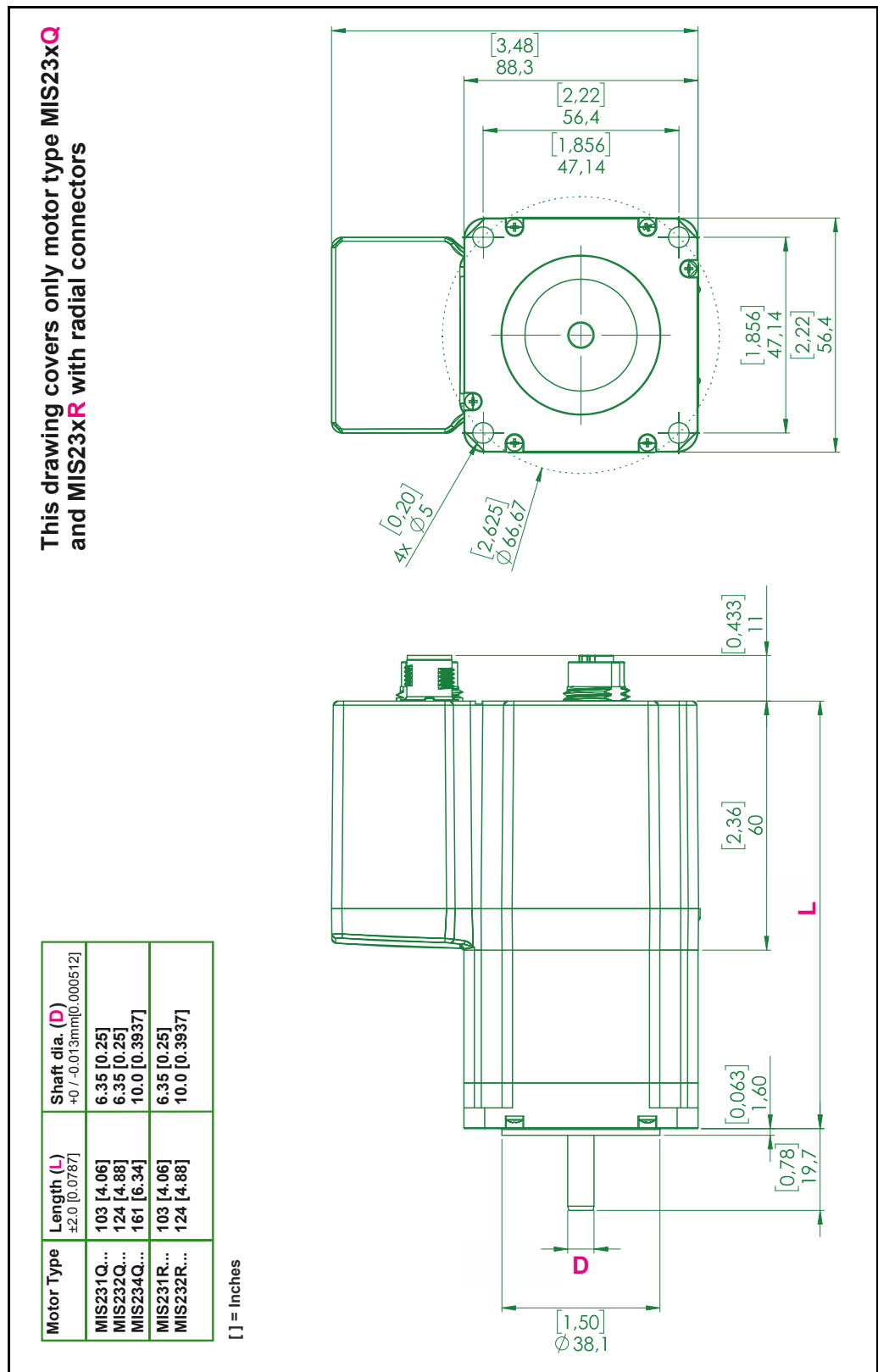


## 14.4

## Physical Dimensions

### 14.4.2 Physical dimensions MIS231Q/R, MIS232Q/R and MIS234Q/R

Notice that this chapter only covers the MIS23x family generation 2.

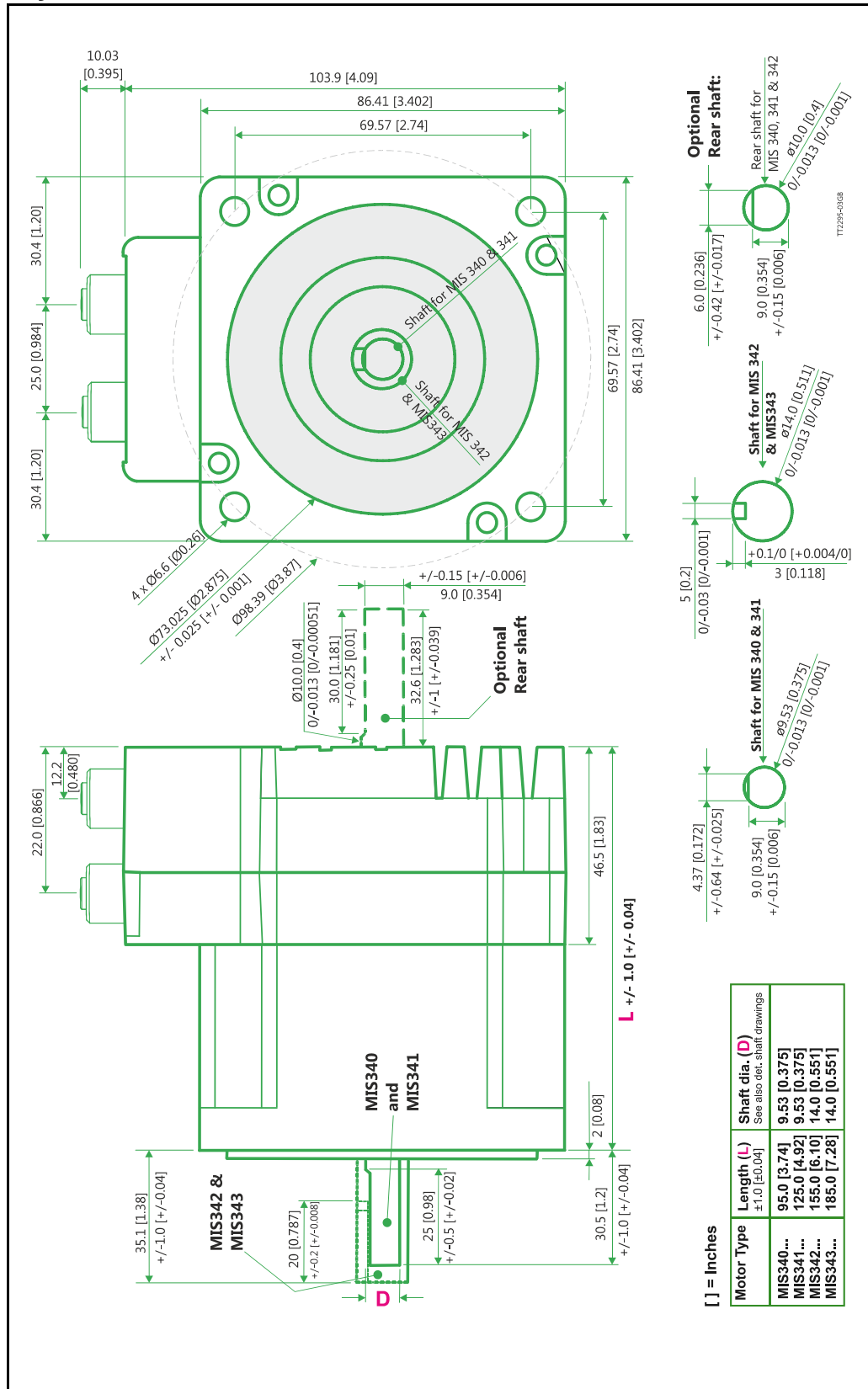


# 14.4

# Physical Dimensions

Only MIS34x

## 14.4.3 Physical dimensions MIS340 - MIS343

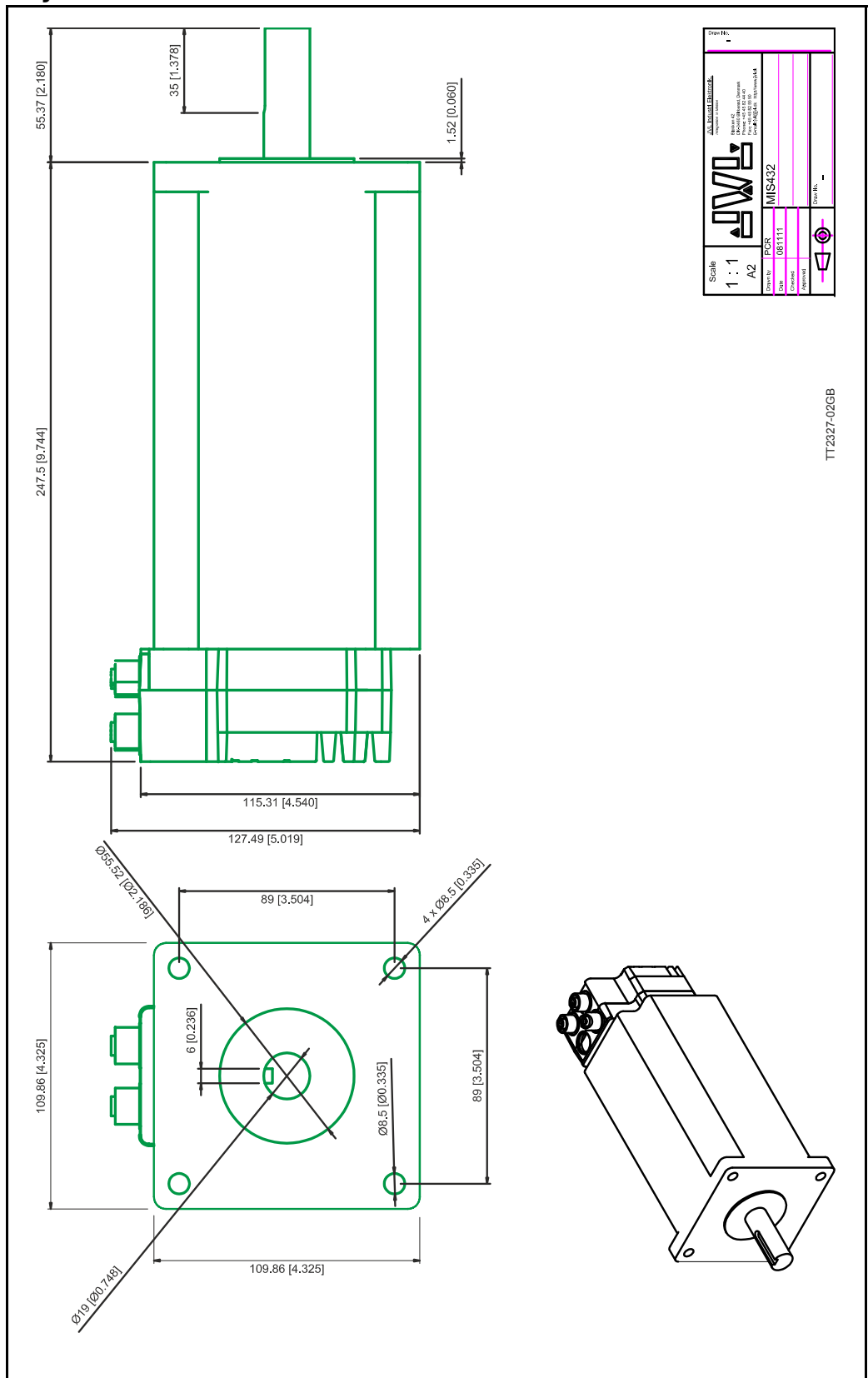


# 14.4

# Physical Dimensions

Only MIS43x

## 14.4.4 Physical dimensions MIS432



## 14.5.1 Life time of ball bearings in MIS34x

The curve below can be used to determine the relation between the radial load at the motor output shaft and where the load is placed at the shaft with reference to the flange of the motor.

The curves are based on a continuous speed of 3000 RPM.

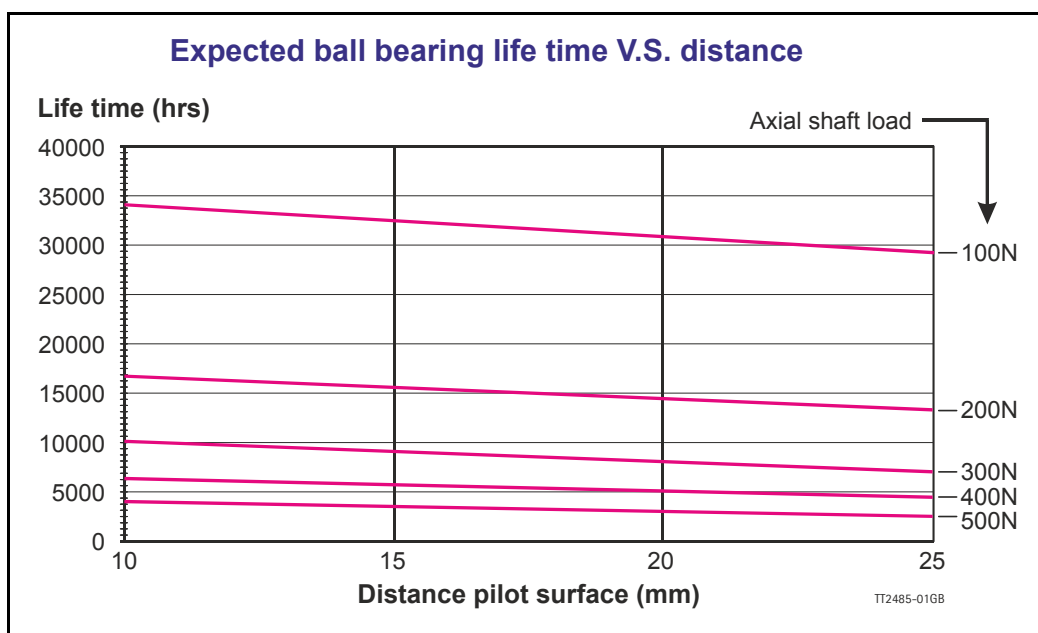
If the speed is lowered the lifetime will increase inversely proportional.

### Example:

A motor is having a radial load of 200N placed with centre 15 mm from the flange.

According to the curve the lifetime will be 15.050 hours at 3000 RPM.

If the speed is lowered to 300 RPM (10 times lower than the curve is specified at) the lifetime will increase 10 times giving a total of 150.500 hours of operation.





## **14.6**                      **Trouble-shooting guide**

---



The following accessories are available for the MIS motor series.



## 15.2

## Power Supplies

---

### 15.2.1 PSU00-PD1

Combined power dump, resistor, and capacitor unit. For a complete power supply system, only a transformer with a secondary winding supplying 32VAC is required.

For systems with up to 5-8 QuickStep motors, this unit can serve as a central power dump unit.

The capacitor offers an efficient and economical way of storing the energy returned from the motors during deceleration of high inertias. See also [www.jvl.dk](http://www.jvl.dk)

### 15.2.2 PSU48-240

A compact switch-mode power supply with 240W output power at 48VDC.

The power supply is UL and CSA approved. It is protected against overvoltage, overtemperature and short-circuit or overload of the output. The power supply can either be mounted on a DIN rail or “wall” mounted. See also the data-sheet LD0047 which can be downloaded from [www.jvl.dk](http://www.jvl.dk)

### 15.2.3 Other power supplies

JVL offers a wide range of power supplies in the power range 45W to 1.5kW with output voltages 24 and 48VDC. They all use switch-mode technology in order to minimize physical dimensions and for easy adaptation to mains voltages in the range 90 to 240VAC.

The product range covers the following types: PSU05-045, PSU24-075, PSU24-240, PSU48-240, PSU48-800, PSU48-1000, PSU48-1500.

See also the data-sheet LD0058 (overview) or LD0053 (detailed) which can be downloaded from [www.jvl.dk](http://www.jvl.dk).

## 15.3 Brakes and shaft reinforcement

---

A family of electromechanical brakes for external mounting is available for the MIS motors.

All brake types can be mounted directly on all the MIS motors and require 24VDC applied to release the motor

Further data for adding a brake to the MIS motors can be found using following links:

MIS23x: [www.jvl.dk](http://www.jvl.dk).

MIS34x: [www.jvl.dk](http://www.jvl.dk).

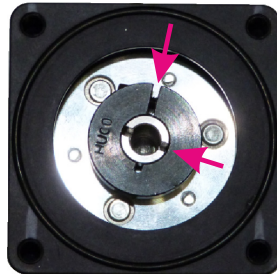
## 15.4 Gear and brake mounting instruction

### 15.4.1 Mounting a gear or a brake at the motor

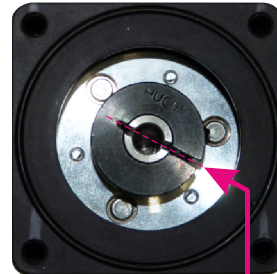
When a gear or a brake is to be mounted on the front end of a motor it is very important that this is done in the right way since a wrong way of mounting may have fatal influence at lifetime of the motor or gear/brake and performance.

Please follow this instruction step by step to make sure that the mounting is done with a good result.

- ① Step 1 - Make sure that the shaft collar is oriented correctly in order to assure that the right tension around the motor shaft is possible.  
Hint: Tighten the shaft collar gently just to keep it in the right position.

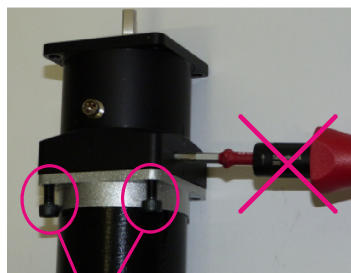


The inner and outer slit is NOT aligned. Make sure they are aligned as shown at right illustration



The inner and outer slit is aligned as they should.

- ② Step 2 - Mount the gear or brake at the motor but make sure to fasten the 4 shaft bolt first before fastening the shaft collar.  
Its recommended to use Loctite 278 in the threads to make sure that the bolts stay in place.



Do NOT tighten the shaft collar before the flange bolts are tightend

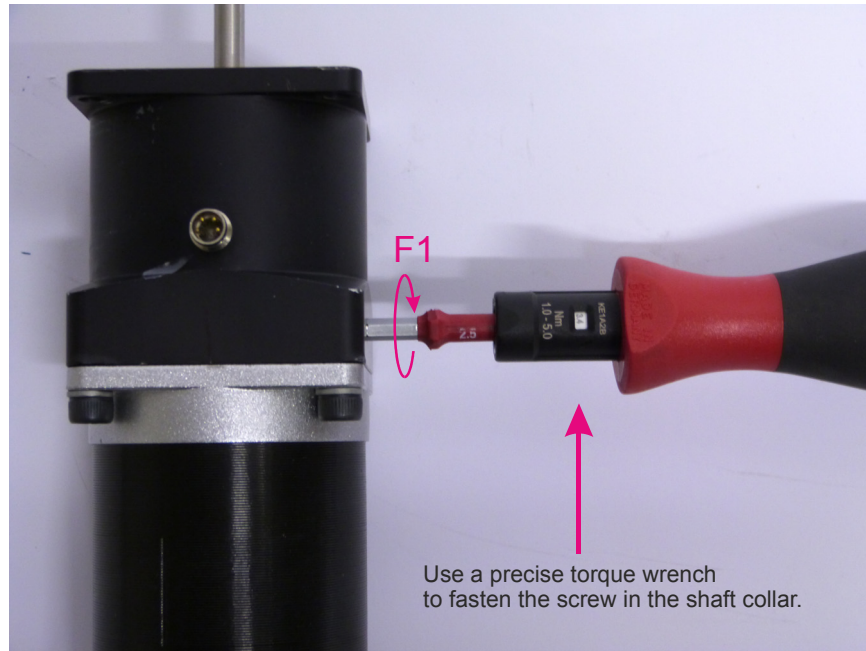


Flange bolts properly mounted and tightend.

TT1536-01GB

## 15.4 Gear and brake mounting instruction

- 3** Step 3 - Final stage. Fasten the shaft collar with a torque of according to the scheme below.  
 Please notice that it can be fatal not to use the specified torque since the shaft may slip over time and cause a position offset.



### Gears (Product type to be mounted)

Series	Used with motor type	Tool	Torque (F1)
HTRG05	MAC050 to MAC141 (Ø6.35 shaft)	Hex size 3	5Nm
HTRG05	MIS230-233 (Ø6.35 shaft)	Hex size 3	5Nm
HTRG06	MAC050 to MAC141 (Ø6.35 shaft)	Hex size 3	5Nm
HTRG06	MAC400-402 (Ø14 shaft)	Hex size 3	11Nm
HTRG08	MIS340-341 (Ø9.53 shaft)	Hex size 4	5Nm
HTRG08	MIS342 (Ø14 shaft)	Hex size 5	8Nm
HTRG08	MAC800 (Ø19 shaft)	Hex size 5	11Nm
HTRG10	MAC800 (Ø19 shaft)	Hex size 5	11Nm

### Brakes (Product type to be mounted)

Series	Used with motor type	Tool	Torque (F1)
MAB23x	MAC050 to MAC141 (Ø6.35 shaft)	Hex size 2.5	2Nm
MAB23x	MIS230-233 (Ø6.35 shaft)	Hex size 2.5	2Nm
MAB34x	MIS340-341 (Ø9.53 shaft)	Hex size 3	5Nm
			TT1537-01GB

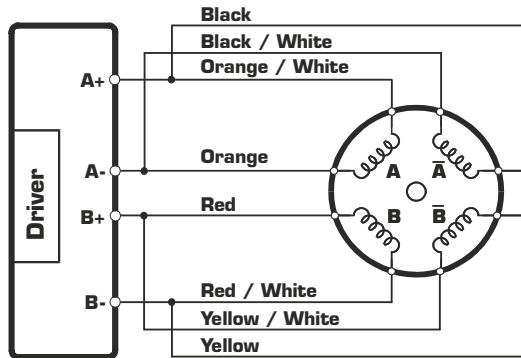




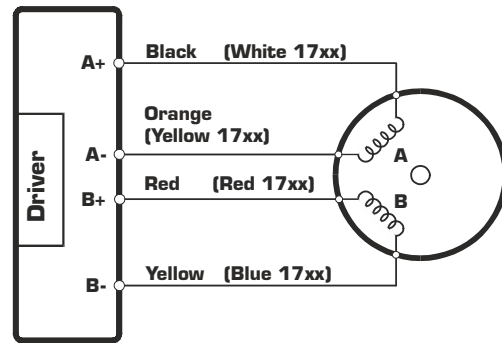
# 16.1

# Motor Connections

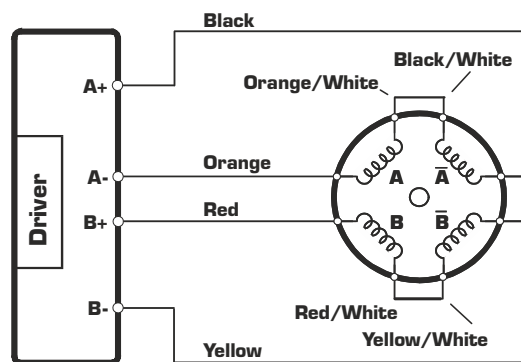
**Connection of JVL and MAE motors (parallel). Type MST23x/ MST34x and HY200-xxxx-xxx-x8**



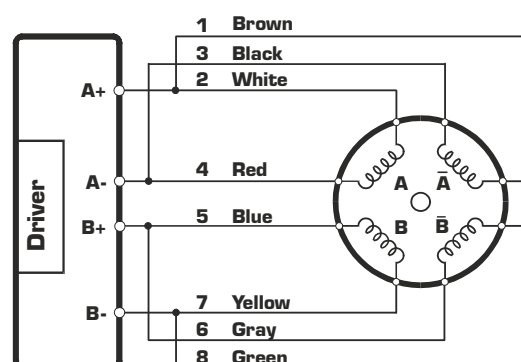
**Connection of JVL and MAE 4 wire motors. Type MST17x and HY200-xxxx-xxx-x4**



**Connection of JVL and MAE motors (serial). Type MST23x/ MST34x and HY200-xxxx-xxx-x8**



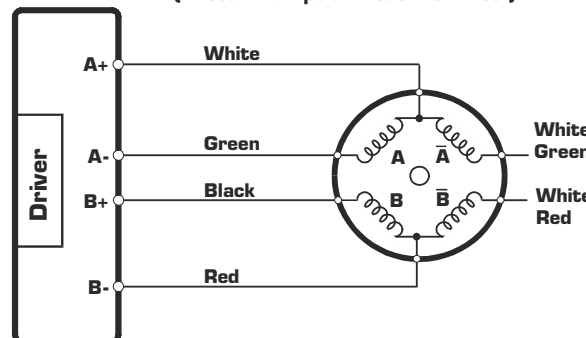
**Connection of Zebotronics motor Type : SMxxx.x.xx.x (8 terminals)**



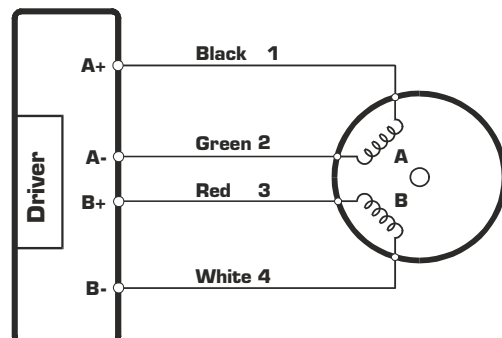
SM87/SM107/168.x.xx ↑ ↑ SM56.x.xx

**Connection of MAE motor (unipol.) Type HY200-1xxx-xxxxx6**

( Motor in unipolar model - 6 wires )



**Connection of Zebotronics motor Type : SMxxx.x.xx.x (4 terminals)**

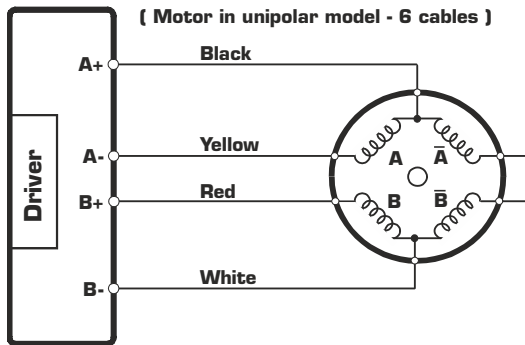


TT0005

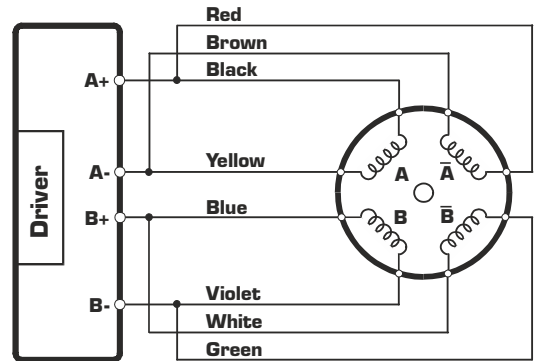
# 16.1

# Motor Connections

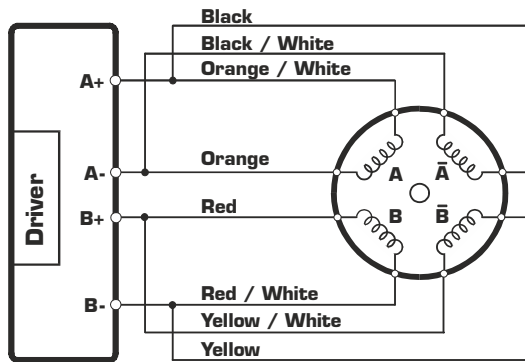
## Connection of Vexta motor Type PH2xx.xxx



## Connection of Phytron motor Type ZSx.xxx.x,x



## Connection of Vexta stepper motor Type : PH2xx-xxx



TT0006

## 16.2

## Serial communication

This section describes control of the MIS motor (or SMC66/85) via the serial interface (RS485).

The interface is RS485 compatible and uses 8 data bits, 1 stop bit and no parity.

The MIS motors (or SMC66/85) are completely controlled by reading and writing to registers.

The registers are numbered 1-255. The width of the registers is 32 bits.

To protect communication from errors, the data is transmitted twice.

First the data byte is transmitted and then an inverted version (255-x) is transmitted.

The easiest way to become familiar with the registers and MacTalk communication is to use the MacRegIO program. This program lists all of the registers, and the serial commands sent and received can be monitored.

### 16.2.1 Supported commands

Sync	Response Sync	Description
0x50	0x52	Read register
0x51	0x52	Read register block
0x52	0x11 (Acknowledge)	Write register
0x54	0x11 (Acknowledge)	Enter safe mode
0x55	0x11 (Acknowledge)	Exit safe mode
0x56	0x11 (Acknowledge)	Write to flash
0x57	None	Reset controller
0x59	None	Group write register
0x61	0x61	Program status and command
0x62	0x11 (Acknowledge)	Write program flash
0x63	0x63	Read program flash

### 16.2.2 Read register

This command can read a register. All registers are read as 32-bit.

Master sends	MIS/SMC Response
<Read><Address><RegNum><End>	<Write><MAddress><RegNum><Len><Data><End>

### Block description

Block name	Protected	Example	Description
<Read>	No	50h,50h,50h	Read command
<Address>	Yes	07h,F8h (Address 7)	The address of the MIS or SMC
<RegNum>	Yes	05h,FAh (RegNum 5)	The register number to read
<End>	No	AAh, AAh	Command termination
<Write>	No	52h,52h,52h	Write command
<MAddress>	Yes	00h,FFh (Address 0)	This will always be 0, because this is the address of the master
<RegNum>	Yes	05h,FAh (RegNum 5)	This will always be the same as requested
<Len>	Yes	04h,FBh (Len = 4)	The length will always be 4
<Data>	Yes	E8h,17h, 03h,FBh, 00h, FFh, 00h,FFh (Data = 1000)	The data read from the register
<End>	No	AAh, AAh	Command termination

## 16.2 Serial communication

### 16.2.3 Read register block

Using this command it is possible to read 64 consecutive registers at once.

Master sends	MIS/SMC Response
<ReadB><Address><RegNum><End>	<Write><MAddress><RegNum><Len><Data><End>

#### Block description

Block name	Protected	Example	Description
<ReadB>	No	51h,51h,51h	Read block command
<Address>	Yes	07h,F8h (Address 7)	The address of the MIS or SMC
<RegNum>	Yes	05h,FAh (RegNum 5)	The first register to read
<End>	No	AAh, AAh	Command termination
<Write>	No	52h,52h,52h	Write command
<MAddress>	Yes	00h,FFh (Address 0)	This will always be 0, because this is the Address of the master
<RegNum>	Yes	05h,FAh (RegNum 5)	This will always be the same as requested
<Len>	Yes	80h,7Fh (Len = 128)	The length will always be 128, so 64 registers is read in each block.
<Data>	Yes	E8h,17h, ..., 03h,FCh	The data read from the registers

### 16.2.4 Write Register

Using this command, a register can be written.

Controller sends	MIS/SMCResponse
<Write><Address><RegNum><Len><Data><End>	<Accept>

#### Block description

Block Name	Protected	Example	Description
<Write>	No	52h,52h,52h	Write command
<Address>	Yes	07h,F8h (Address 7)	The address of the MIS/SMC
<RegNum>	Yes	05h,FAh (RegNum 5)	The register number to write to
<Len>	Yes	02h,FDh (Len = 2)	The number of data bytes
<Data>	Yes	E8h,17h, 03h,FCh (Data = 1000)	The data to write to the register
<End>	No	AAh, AAh	Command termination
<Accept>	No	11h, 11h,11h	Accept from MIS/SMC

### 16.2.5 Enter safe mode

When this command is sent, the MIS/SMC switches to safe mode. In safe mode, no program or commands can enable the motor. The mode can only be exited using either an "Exit safe mode" or "Reset" command.

Controller sends	MIS/SMC response
<EntSafe><Address><End>	<Accept>

#### Block description

Block Name	Protected	Example	Description
<EntSafe>	No	54h,54h,54h	Enter safe mode command
<Address>	Yes	07h,F8h (Address 7)	The address of the MIS/SMC
<End>	No	AAh, AAh	Command termination
<Accept>	No	11h, 11h,11h	Accept from MIS/SMC

## 16.2 Serial communication

### 16.2.6 Exit safe mode

When this command is sent, the MIS/SMC switches back to normal mode.

Controller sends	MIS/SMC response
<ExitSafe><Address><End>	<Accept>

#### Block description

Block Name	Protected	Example	Description
<ExitSafe>	No	55h,55h,55h	Exit safe mode command
<Address>	Yes	07h,F8h (Address 7)	The address of the MIS/SMC
<End>	No	AAh, AAh	Command termination
<Accept>	No	11h, 11h,11h	Accept from MIS/SMC

### 16.2.7 Write to flash

This command writes the register values to flash memory. The values will then be retained after a power down. The command will only work if the motor is in "Safe mode" After the command is executed, the motor will reset. The response will only be transmitted if the command failed, e.g. if the motor is not in safe mode.

Controller sends	MIS/SMC response
<WriteFlash><Address><End>	<Accept>

#### Block description

Block Name	Protected	Example	Description
<WriteFlash>	No	56h,56h,56h	Write to flash command
<Address>	Yes	07h,F8h (Address 7)	The address of the MIS/SMC
<End>	No	AAh, AAh	Command termination
<Accept>	No	11h, 11h,11h	Accept from MIS/SMC

### 16.2.8 Reset controller

This command resets the MIS/SMC. No response will be transmitted from the MIS/SMC.

Controller sends	MIS/SMC response
<Reset><Address><End>	None

#### Block description

Block Name	Protected	Example	Description
<Reset>	No	57h,57h,57h	Reset command
<Address>	Yes	07h,F8h (Address 7)	The address of the MIS/SMC
<End>	No	AAh, AAh	Command termination

## 16.2 Serial communication

### 16.2.9 Group write register

Using this command it is possible to write a register in several MIS/SMCs with one command.

The command includes a sequence number which must be changed for each write. This is used so that the same command can be written several times, to ensure that all controllers received it. The last received sequence id can be read in register I48.

Controller sends	MIS/SMC Response
<GWrite><Group><Sequence><RegNum><Len><Data><End>	None

#### Block description

Block Name	Protected	Example	Description
<GWrite>	No	59h,59h,59h	Group write command
<Group>	Yes	07h,F8h (Address 7)	The group id of the MIS/SMCs to write to.
<Sequence>	Yes	04h,FBh (Sequence 4)	The sequence number of the write.
<RegNum>	Yes	05h,FAh (RegNum 5)	The register number to write to
<Len>	Yes	02h,FDh (Len = 2)	The number of data bytes
<Data>	Yes	E8h,17h, 03h,FC (Data = 1000)	The data to write to the register
<End>	No	AAh, AAh	Command termination

### 16.2.10 Program status and command

Using this command, different actions can be executed. The command also returns some information about the program state.

The table below shows the possible commands:

Com- mand	Data 1	Data 2	Description
0	-	-	No operation
1	-	-	Start program execution
2	-	-	Stop program execution
3	-	-	Pause program execution
4	Start Address (16bit)	End Address (16bit)	Run the program until the program pointer is outside the area [Start Address End Address] Then the program is paused
5	Set outputs (8bit)	Clear outputs (8bit)	Modifies the outputs. The bits set in the "Set outputs" data will be set and cleared for "Clear outputs". Example: The data 0x06,0x01 sets output 2+3 and clears output 1
6			Reserved
7	Size (16 bit)		Prepare the flash for a new program. Data 1 specifies the size of the program in bytes.

The command number is placed in the first command data byte. Data 1 + Data 2 are placed in the following command data bytes.

Controller sends	MIS/SMC Response
<PStat><Address><Len1><Data1><End>	<PStat><MAddress><Len2><Data2><End>

## 16.2

## Serial communication

### Block description

Block Name	Protected	Example	Description
<PStat>	No	61h,61h,61h	Program status command
<Address>	Yes	07h,F8h (Address 7)	The address of the MIS/SMC's to write to.
<Len1>	Yes	01h,FEh (Len = 1)	Length of the command data
<Data1>	Yes	01h,FEh (Start)	Command data
<MAddress>	Yes	00h,FFh (Address 0)	This will always be 0, because this is the address of the master
<Len2>	Yes	08h,F7h (Len = 8)	The length of the return data
<Data2>	Yes	09h,F6h, (Program state) 00h,FFh, 00h,FFh, (Program pointer) 00h,FFh, (Stack pointer) 00h,FFh, 00h,FFh, (Program checksum) 80h,7Fh, (Inputs) 00h,FFh (Outputs)	Data returned from MIS/SMC
<End>	No	AAh, AAh	Command termination

The returned data has the following format:

Data offset	Size	Description
0	8 bit	Program state. See table below for states.
1	16 bit	Program pointer. The current location of the program pointer.
3	8 bit	Stack pointer
4	16 bit	Program checksum. This checksum is calculated when the program is started.
6	8 bit	Input status.
7	8 bit	Output status

Program states:

Program state	Name	Description
0	Passive	The program execution is stopped. This state is only entered shortly at power-up.
1	Running	The program execution is running
2	Single Step	A single step is in progress. The program will run until the selected program position is reached.
3	Paused	The program execution is paused, but can be resumed again.
4	Stack Overflow	The stack pointer has overflowed
5	Program Overflow	The program pointer has overflowed.
6	Invalid Ins.	An invalid instruction is encountered in the program.
7	Stopped	The program execution is stopped.
8	Com. Error	Internal communication error has occurred. This cannot happen on MIS/SMC.
9	Starting Program	Program execution is being prepared. After this is completed the state will change to running.
10	Flash Error	The program data is corrupted.
11	Flash Checksum Error	The program data checksum is incorrect.



# 16.3

# MIS Ordering Information

QuickStep		MISxxx Motor Integrated Stepper motor - Part number system										
-	228	#	###	72	75	###	-	#/T				
Motor type	Size	Generation	IP and shaft	Connection	Feedback	Driver Technology	Coating	Step Resolution	mA in driver	Input format	Standby current ratio	
MIS	232	A	2	M5	H2	75		#	30	D	3	
												B001 custom made for customer. See special folder
												01 to
												31 Standby current ratio(03 = 1/3 standby current) #
												D 24V NPN inputs
												E 24V PNP inputs
												F 5V inputs
												Only if SMD73 or SMD74 driver
												yy Only non SMD73/74 Special length e.g. 01 for special screw coating.
												xx xx specify mA*100/phase. See SMD73 / SMD74 datasheet
												C Only non SMD73/74 (C = Custom length or other....two following digits after C, determines e.g. Special length of screw
												x For ML see "Special ML" tab
												Only if SMD73 or SMD74 driver
												1 1/1 step (with 200step/rev motor 200 pulses/rev)
												2 1/2 step (with 200step/rev motor 400 pulses/rev)
												4 1/4 step (with 200step/rev motor 800 pulses/rev)
												5 1/5 step (with 200step/rev motor 1000 pulses/rev)
												8 1/8 step (with 200step/rev motor 1600 pulses/rev)
												Below only for ML linear motors. Specify stroke movement in mm
												Ex Nx Bx Cx
												A 50 50 25.70
												B 75 75 32.00
												C 100 100 38.40
												D 125 125 44.70
												E 150 150 51.10
												F 200 200 63.80
												G 250 250 76.50
												H 300 300
												I 350 350
												J 400 400
												Below other optional digits.
												- Normal, No coating, Standard #
												KIT Kit for MIS23xxM5 with all cables and PA0160 test IO boks
												73 SMD73 driver 15-28VDC. Pulse and direction driver. 200,400,800,1000 or 1600step/rev (Only orders more than 10 pcs. See note1)
												C3 SMD73 driver , Coated PCB
												74 SMD74 Driver 12-48VDC based on SMD73 technology but up to 48VDC supply voltage. 200,400,800,1000 or 1600step/rev
												C4 SMD74 driver , Coated PCB
												75 SMC75 controller with MAC protocol. 12-48VDC and optional encoder. Fixed stepresolution 1600step/rev #
												C5 SMC75 controller , Coated PCB
												66 SMC66 NEW controller 6A 12-72VDC, optional H2/H3 and IE. Stepresolution up to 409600step/rev#
												C6 SMC66 NEW controller , Coated PCB
												S6 STO SMC66 NEW controller 6A 12-72VDC, optional H2/H3 and IE. Stepresolution up to 409600step/rev#
												T6 STO SMC66 NEW controller , Coated PCB
												85 SMC85 controller 12-80VDC and new high resolution driver. Stepresolution up to 409600 step/rev#
												C8 SMC85 controller, Coated PCB
												S8 STO SMC85 controller 12-80VDC and new high resolution driver. Stepresolution up to 409600 step/rev#
												T8 STO SMC85 controller, Coated PCB
												N0 No feedback (with controller pcb, no encoder option)
												H2 Magnetic encoder feedback. 256x4 pulses/rev. Only SMC75, SMC85, MIS23x and MIS34x
												H3 Absolute multiturn encoder magnetic feedback. Only SMC85 and MIS34x
												H4 H2 12bit enc + H3 encoder. Closed loop Aolute multiturn encoder magnetic feedback. Only MIS34x after 2017 and MIS23xQRST

Continued next page

# 16.3

# MIS Ordering Information

QuickStep		MISxxx Motor Integrated Stepper motor - Part number system																					
Motor type	Size	Generation	IP and shaft	Connection	Feedback	Driver Technology	Coating	StepResolution	mA in driver	Input format	Standby current ratio												
-	228	#	###	72	75	###	-	#VT															
MIS	232	A	2	M5	H2	75		#	30	D	3												
M1	M12	1 pcs.	5 pin male.	Only SMD73/74 pulse/direction driver. (Not SMC75/85)																			
M2	M12	2 pcs.	5 pin male (power).	8 pin female (RS485, 4IOA).	SMC75																		
M3	M12	3 pcs.	5 pin male (power).	8 pin female (RS485, IOA 1-4).	5 pin female (RS485).	SMC75																	
M4	M12	3 pcs.	5 pin male (power).	8 pin female (RS485, IOA 1-4).	8 pin female (5V serial, IOA5-8).	SMC75																	
M5	M12	4 pcs.	5 pin male (power).	8 pin female (RS485, IOA 1-4).	5 pin female (RS485).	8 pin female (5V serial, IOA 5-8).	SMC75																
M6	M12	4 pcs.	CANopen: 5 pin male (power).	8 pin female (RS485, IOA 1-4).	8 pin female (5V serial, IOA 5-8).	5 pin male (CAN).	SMC75																
M7	M12	4 pcs.	Device Net: 5 pin male (power).	8 pin female (RS485, IOA 1-4).	8 pin female (5V serial, IOA 5-8).	5 pin male (DEVICE).	SMC75																
M8	M12	4 pcs.	SSI + CANopen: 5 pin male (power).	8 pin female (RS485, IOA 1-4).	8 pin female (IOA 5-6).	5 pin male (CANopen).	SMC75																
M9	M12	4 pcs.	SSI: 5 pin male (power).	8 pin female (RS485, IOA 1-4).	8 pin female SSI (IOA 5-6).	5 pin female RS485.	SMC75																
MA	M12	3 pcs.	5 pin male (power).	8 pin female (RS485, IOA 1-4).	5 pin male (CAN).	SMC75																	
MB	M12	4 pcs.	5 pin male (power).	8 pin female (RS485, IOA 1-4).	5 pin male (CAN).	5 pin female (CAN).	SMC75																
MC	M12	3 pcs.	3m power cable PG12.	8 pin female (RS485, IOA 1-4).	5 pin male (CAN).	5 pin female (CAN).	SMC75.	Obsolete															
MD	M12	3 pcs.	3m power cable PG12.	8 pin female (RS485, IOA 1-4).	5 pin male (CAN).	5 pin female (CAN).	SMC75.	Obsolete															
ME	M12	2 pcs.	PWR+CAN in	5 pin male.	8 pin female (RS485, IOA 1-4)																		
MF	M12	1 pcs.	PWR+I <sup>-</sup> +3+C1	8 pin male (UDGAR)																			
MG	M12	1 pcs.	PWR, RS485 IOA 1-4	in same	M12 12 pin male.	SMC75/MIS23x																	
MH	M12	3 pcs.	PWR, CAN IO	in 5 pin male.	8 pin female (RS485, IOA 1-4)																		
ML	M12	1 pcs.	PWR, RS485 CAN IO	in 17 pin male.	SMC75/MIS23x																		
N1			No backcover and no cables.	Only PCB mounted in housing.	Open in rear end																		
R1	Radial connection.	M12 2 pcs.	5 pin male (power).	8 pin female (RS485, 4IOA)	on 2 sides.	High volume.	ONLY MIS23x																
R2	Radial connection.	M12 2 pcs.	5 pin male (power).	8 pin female (RS485, 4IOA)	on 2 sides.	1-50 pcs	Only MIS23x																
C1	2 pcs PG12 cable (Cands M12x).	5 and 1 pin cable	mounted (side mounted only)	MIS23x																			
C2	2 pcs PG12 cable (Cands M12x).	5 and 5m power and IO cable	with shield mounted (Side mounted)	Only MIS23x																			
C3	2 pcs PG12 cable (Cands M12x).	5 and 1m power and IO cable	with shield mounted (Side mounted)	Only MIS23x																			
C6	CANIOPEIN + 2 pcs PG12 cable (Cands M12x).	1,5 and 2m power and IO cable	with shield mounted (Side mounted)	Only MIS23x (Obsolete)																			
I1	Interconnector dual connector	and 1 pcs M12 for RS485	(Only MIS23x)																				
W0	2 pcs PG12 cable (Cands M12x).	5 and 1 pin cable	mounted (Rear end mounted)	Only MIS23x																			
W1	2 pcs PG12 cable (Cands M12x).	5 and 1m power and 1m IO cable	with shield mounted (Rear end mounted)	Only MIS23x																			
W2	2 pcs PG12 cable (Cands M12x).	5 and 5m power and 1m IO cable	with shield mounted (Rear end mounted)	Only MIS23x																			
W6	CANIOPEIN + 2 pcs PG12 cable (Cands M12x).	1,5 and 2m power and IO cable	with shield mounted (Rear end mounted)	Only MIS23x (Obsolete)																			
Z1	1 pcs.	Rubber sealing on top of motor.	1 cable 1m long (MSW17, MST17)																				
Z3	2 pcs	rubber sealing in side of motor.	2 cables 1m long	Only MIS23x																			
FP	M12.	Profibus 4 pcs M12.	5 pin male (power).	8 pin female (RS485).	17 female (IO).	5 pin male (B) Profibus DP (obsolete)																	
P6	M12.	CANOPEN 4 pcs M12.	5 pin male (power).	17 female (IO).	5 pin female (A) (CANopen).	5 pin female (B) (CANopen).	Only MIS34x/43x/SMC85/66																
Q9	M12.	SSI 4 pcs M12.	5 pin male (power).	8 pin female (RS485).	17 female (IO).	8 pin male (SSI + IO5-6).	Only MIS34x/43x/SMC85/66																
Q5	M12.	RS485/PLC 4 pcs M12.	5 pin male (power).	8 pin female (RS485).	17 female (IO).	5 pin female (RS485).	Only MIS34x/43x/SMC85/66																
ES	M12.	Sercos III 4 pcs M12.	5 pin male (power).	17 female (IO).	2x 4 pin male (D) Ethernet MODBUS TCP.	Only MIS34/43/SMC85/66																	
EC	M12.	Ethercat 4 pcs M12.	5 pin male (power).	17 female (IO).	2x 4 pin male (D) Ethernet Ethercat.	Only MIS34/43/SMC85/66																	
EL	M12.	Powerlink 4 pcs M12.	5 pin male (power).	17 female (IO).	2x 4 pin male (D) Ethernet Powerlink.	Only MIS34/43/SMC85/66																	
EI	M12.	Ethernet IP 4 pcs M12.	5 pin male (power).	17 female (IO).	2x 4 pin male (D) Ethernet Ethernet IP.	Only MIS34/43/SMC85/66																	
EP	M12.	Profinet 4 pcs M12.	5 pin male (power).	17 female (IO).	2x 4 pin male (D) Ethernet Profinet.	Only MIS34/43/SMC85/66																	
EM	M12.	MODBUS TCP 4 pcs M12.	5 pin male (power).	17 female (IO).	2x 4 pin male (D) Ethernet MODBUS TCP.	Only MIS34/43/SMC85/66																	
E1	NOT TO BE USED (BECAUSE CONFLICT WITH E1)	(M12, RS422(SS)I+SERCOS III 4 pcs	M12.	5 pin male (power).	17 female (IO).	1x 4 pin male (D) Ethernet																	
E2	M12.	RS422(SS)I+Ethercat 4 pcs	M12.	5 pin male (power).	17 female (IO).	1x 4 pin male (D) Ethernet Ethercat.	8 pin male (SS+IO5-6).	Only MIS34/43/SMC85/66															
E3	M12.	RS422(SS)I+Ethernet/IP 4 pcs	M12.	5 pin male (power).	17 female (IO).	1x 4 pin male (D) Ethernet/IP	8 pin male (SS+IO5-6).	Only MIS34/43/SMC85/66															
E4	M12.	RS422(SS)I+Powerlink 4 pcs	M12.	5 pin male (power).	17 female (IO).	1x 4 pin male (D) Ethernet Powerlink.	8 pin male (SS+IO5-6).	Only MIS34/43/SMC85/66															
E5	M12.	RS422(SS)I+MODBUS TCP 4 pcs	M12.	5 pin male (power).	17 female (IO).	1x 4 pin male (D) Ethernet MODBUS TCP	8 pin male (SS+IO5-6).	Only MIS34/43/SMC85/66															
E6	M12.	RS422(SS)I+Profinet 4 pcs	M12.	5 pin male (power).	17 female (IO).	1x 4 pin male (D) Ethernet Profinet	8 pin male (SS+IO5-6).	Only MIS34/43/SMC85/66															
E7	M12.	RS422(SS)I+SERCOS III 4 pcs	M12.	5 pin male (power).	17 female (IO).	1x 4 pin male (D) Ethernet Ethercat.	8 pin male (SS+IO5-6).	Only MIS34/43/SMC85/66															
EW	M12.	Wireless WLAN 4 pcs	M12.	5 pin male (power).	8 pin female (RS485).	17 female (IO).	Antenna Wireless Bluetooth																
WI	M12	Wireless Ethernet/Bluetooth or WLAN.	3 pcs M12.	5 pin male (power).	17 female (IO).	1x 4 pin male (D) Ethernet.	1x antenna.	Only MIS23QRS															
WM	M12	Wireless Modbus TCP, Bluetooth or WLAN.	3 pcs M12.	5 pin male (power).	17 female (IO).	1x 4 pin male (D) Ethernet.	1x antenna.	Only MIS23QRS															
WP	M12	Wireless Profinet, Bluetooth or WLAN.	3 pcs M12.	5 pin male (power).	17 female (IO).	1x 4 pin male (D) Ethernet.	1x antenna.	Only MIS23QRS															
HP	M23	Hummel Profinet.	P can be replaced with S: Sercos, C: Ethercat, L: Powerlink, I: ethernetIP, P: Profinet, M: ModbusTCP.																				
H5	M23	Hummel.	Functions like Q5 and Q9																				
HC	M23	Hummel.	Functions like Q5 and Q9 with CAN																				
JP	M23	JVL connector.	P can be replaced with S: Sercos, C: Ethercat, L: Powerlink, I: ethernetIP, P: Profinet, M: ModbusTCP.																				
J5	M23	JVL connector.	Functions like Q5 and Q9																				
JC	M23	JVL connector.	Functions like Q5 and Q9 with CAN																				

# 16.3

# MIS Ordering Information

QuickStep		MISxxx Motor Integrated Stepper motor - Part number system											
Motor type	Size	Generation	IP and shaft	Connection	Feedback	Driver Technology	Coating	Step/Resolution	mA in driver	Input format	Standby current ratio		
-	228	#	###	72	75	###	-	#VT					
MIS	232	A	2	M5	H2	75		#	30	D	3		
1	6.35mm shaft and IP42. Flange shaft.												
2	6.35mm shaft and IP65 (motor shaft and body) IP66 (Rear end and connector) and special painting												
3	10.0 mm shaft and IP42												
4	10.0mm shaft and IP65 (motor shaft and body) IP66 (Rear end and connector) and special painting. Stainless steel flange.												
5	14mm shaft and IP42												
6	14mm shaft round and IP35 (motor shaft and body) IP66 (Rear end and connector) and special painting												
7	8mm shaft 52mm long for HFOS worm gear. IP42												
8	6.35mm shaft with D-cut and IP42												
9	5.00 mm shaft with D-cut and IP42												
10	7.00mm shaft 45.5 mm long for Dunker flange and IP42. Only MIS23x												
11	6.35mm shaft. Black painted and rubber sealing in rear end IP65. Shaft end IP42.												
12	9.53mm shaft D shape. Black painted. Shaft end IP42. Only MIS34x												
13	9.53mm shaft D shape. Black painted. Shaft end IP42. Rear end shaft ø10mm 30mm long D shape. Only MIS34x												
14	14mm with 5x5 key shaft. Black painted. Shaft end IP42. Only MIS34x												
15	14mm with 5x5 key shaft. Black painted. Shaft end IP42. Rear end shaft ø10mm 30mm long D shape. Only MIS34x												
16	5.00 mm round shaft IP42 (not MIS34)												
17	9.53mm shaft D shape. Black painted. Shaft and rear end IP65. Only MIS34x												
18	9.53mm shaft D shape. Black painted. Shaft and rear end IP65. Rear end shaft ø10mm 30mm long D shape. Only MIS34x												
19	14mm with 5x5 key shaft. Black painted. Shaft and rear end IP35. Only MIS34x												
20	14mm with 5x5 key shaft. Black painted. Shaft and rear end IP65. Rear end shaft ø10mm 30mm long D shape. Only MIS34x												
21	19mm with key 5x20mm IP65. Shaft sealing and painted. (only for MIS43x/ MST42x ) (OLD) !!! 16mm with key 5x9mm (only for MIS34x )												
22	19mm with key 5x20mm (only for MIS43x/ MST42x )												
23	8mm Shaft: IP67, motor and housing and rear end IP67 (only MSW23/24)												
24	14 mm Shaft with D shape, Shaft IP67 motor and housing and rear end IP65 (only NEAM34 IP65 motors ) Only MSW34												
25	6.35mm Shaft IP65, motor and housing and rear end IP67. (Not stainless steel flange). Only MIS23 and MSW23												
26	10mm Shaft IP65, motor and housing and rear end IP65 (Not stainless steel flange). MIS23 / 34												
27	9.53mm shaft D shape. Potted. High G. Black painted. Shaft end IP42. Only MIS34x												
28	14mm with 5x5 key shaft. Potted. High G. Black painted. Shaft end IP65. Only MIS34x												
29	5mm shaft. Standard single (No doubleshaft) according to datasheet. With or without gear. Eg MST11x/MST14x												
30	With double shaft according to datasheet. With or without gear. Eg MST11x*MST14x. Ø5mm Only MST11/14/23												
31	Hollow shaft OD=15/ID=12. Only MIS34												
32	Hollow shaft OD=15/ID=12. With keyway 4mm. Only MIS34. New december 2016												
33	10.0 mm shaft. Black painted and rubber sealing in rear end IP35. Shaft end IP42. Only OEM +25pcs												
34	IP65. Hollow shaft OD=15/ID=12. MIS34/341 Black painted. Teflon sealing in front and rear end												
35	8mm shaft with keyway 3mm*10mm. Only OEM +25pcs. Only MIS232												
36	MIS34x/ML34x with rear end brake mounted 3Nm. Open brake. IP00, Motor IP42 (Note for ML, add B36 to the end of ML partnumber)												
37	MIS34x/ML34x with rear end brake mounted 3Nm. Rear end plastic cover. IP55, Motor IP42 (Note for ML, add B37 to the end of ML partnumber)												
38	MIS34x/ML34x with rear end brake mounted 3Nm. Rear end plastic cover. IP65, Motor IP65 (Note for ML, add B38 to the end of ML partnumber)												
43	8mm Shaft: IP67, motor and housing and rear end IP67 (only MSW23/24). Painted Thick Black												
44	14 mm Shaft with D shape, Shaft IP67 motor and housing and rear end IP65 (only NEAM34 IP65 motors ) Only MSW34. Painted Thick Black												
45	6.35mm Shaft IP65, motor and housing and rear end IP67. (Not stainless steel flange). Only MIS23 and MSW23. Painted Thick Black												
46	6.35mm Shaft, motor and housing and rear end IP67. (Not stainless steel flange). Only MIS23 and MSW23/24. Painted WHITE epoxy												
47	14mm Shaft, motor and housing and rear end IP67. (Not stainless steel flange). Only MIS34 and MSW34. Painted WHITE epoxy												
48	10.0 mm shaft length 20.6 D shape 9mm/15mm. IP42 shaft and housing. Only MIS234												
49	5.00 mm shaft with D-cut, Double shaft 5mm D-shape, and IP42												
50	14mm shaft with keyway, length: 30mm threaded hole in front shaft												
51	10mm D-shape, Double shaft. IP42.												
52	10mm Hollow shaft Ø7mm special. Front shaft SW9 thread. Rear end shaft G1/8 thread. Only MIS231*36*												
53	6.35mm with D-shape. IP57 (MSW17x)												
54	MIS34x with special ø3.5/ø8.0 mm shaft. Shaft length 8,15/27,8 mm. Only MIS34x. Only OEM 25 pcs (340/341)												
55	MIS34x with special ø3.5/ø10.0 mm shaft. Shaft length 7.0/45.7 mm. Only MIS34x. Only OEM 25 pcs (340/341)												
56	6.35mm shaft IP65 (sealing), Motor and housing IP65, no painting												
57	6.35 mm shaft, rear mounted brake, IP42												
58	6.35 mm shaft, rear mounted brake + cover. IP65												
Ex	Linear stepmotor External nut. x can be A to Z. Pitch from 0.6 to 25.4mm/rev. See jvLdk												
Nx	Linear stepmotor Non-Captive. External nut.(Not ML). x can be A to Z. Pitch from 0.6 to 25.4mm/rev. See jvLdk												
Cx	Linear stepmotor captive with intern guider. x can be A to Z. Pitch from 0.6 to 25.4mm/rev. See jvLdk (Not available for ML34)												
Bx	Linear stepmotor External nut with ball screw. x can be A to Z. Pitch typical 5.00mm See jvLdk												
Ax	Linear stepmotor External anti backlash nut (small screw diameter, default) (Not available for ML34). More information.												
Fx	Linear stepmotor External bronze nut. (small screw diameter, default)												
Gx	Linear stepmotor External plastic nut. (big screw diameter)												
Hx	Linear stepmotor External bronze nut. (big screw diameter)												
Rx	Linear stepmotor External nut with rolled ball screw and steel balls. (Only for +25 pcs. order)												

# 16.3

# MIS Ordering Information

QuickStep	MISxxx Motor Integrated Stepper motor - Part number system										
Motor type	Size	Generation	IP and shaft	Connection	Feedback	Driver Technology	Coating	StepResolution mA in driver	Input format	Standby current ratio	
-	228	#	###	72	75	###	-	#VT			
MIS	232	A	2	M5	H2	75		#	30	D	3
	A	Driver 3.0A/phase, Motor 3Amp, for 200step/rev. See driver technology for actual stepresolution.									
	B	Driver 6.0A/phase, Motor 6Amp and 200step/rev. (Eg ML34xxx). See driver technology for actual stepresolution.									
	C	Driver 9.0A/phase, Motor 9Amp and 200step/rev. See driver technology for actual stepresolution.									
	D	Driver 12.0A/phase, Motor 12Amp and 200step/rev. See driver technology for actual stepresolution.									
	E	Reserved									
	F	Driver 3.0A/phase, Motor 3Amp and double stepresolution (400step/rev). See driver technology for actual stepresolution.									
	G	Driver 6.0A/phase, Motor 6Amp and double stepresolution (400step/rev). See driver technology for actual stepresolution.									
	H	Driver 9.0A/phase, Motor 9Amp and double stepresolution (400step/rev). See driver technology for actual stepresolution.									
	I	Driver 12.0A/phase, Motor 12Amp and double stepresolution (400step/rev). See driver technology for actual stepresolution.									
	J	Reserved									
	K	Driver 4.6A/phase, Motor 6Amp and 200step/rev. See driver technology for actual stepresolution. (only MIS23x).									
	L	Driver 4.0A/phase, Motor 6Amp and 200step/rev. See driver technology for actual stepresolution. (only MIS23x)									
	M	Driver 3.0A/phase, Motor 3Amp and 200step/rev. See driver technology for actual stepresolution. IP65									
	N	Driver 6.0A/phase, Motor 6Amp and 200step/rev. IP65. See driver technology for actual stepresolution. (eg ML34xxx)									
	O	Driver 9.0A/phase, Motor 9Amp and 200step/rev. See driver technology for actual stepresolution. IP65									
	P	Driver 12.0A/phase, Motor 12Amp and 200step/rev. See driver technology for actual stepresolution. IP65									
	Q	Axial connector, Standard Torque, For new driver technology 6A N17/23 and new housing									
	R	Axial Connector, Extreme high Torque +40%, For new driver technology 6A N17/23 and new housing									
	S	Radial connector, Standard Torque, For new driver technology 6A N17/23 and new housing									
	T	Radial connector, Extreme high torque +40%, For new driver technology 6A N17/23 and new housing									
	230	NEMA23 stepmotor									
	231	NEMA23 stepmotor, 1 stack									
	232	NEMA23 stepmotor, 2 stack									
	234	NEMA23 stepmotor									
	340	NEMA34 stepmotor 1/2 stack									
	341	NEMA34 stepmotor 1 stack									
	342	NEMA34 stepmotor 2 stack									
	430	NEMA43 stepmotor 1/2 stack									
	431	NEMA43 stepmotor 1 stack									
	432	NEMA43 stepmotor 2 stack									
ML	Like MISxxx Motor Integrated Stepper but with linear motor, Captive, in and external nut.										
MIS	MISxxx Motor Integrated Stepper										
<b>Examples</b>											
-	215	#	-	16	#	###	-	#			
MIS	231	A	1	R1	N0	75		8	25	D	Motor 6.35 shaft, flying leads, SMD73 driver
-	228	#	-	-	#	###	-	#			
MIS	232	A	3	M1	N0	73		2	30	D	Motor 10mm shaft, M12, SMD73
-	228	#	-	40	#	###	-	#VT			
MIS	232	A	1	M3	N0	75					Motor 6.35mm shaft, SMC75, 3 pcs M12 connectors
-	252	#	-	###	#	###	-	#VT			
MIS	234	A	3	M6	N0	75					Motor 10mm shaft, SMC75, 4 pcs M12 connectors, CANopen
MIS	232	A	1	M7	H2	75					Motor 6.35mm shaft, SMC75, 4 pcs M12 connectors, DeviceNet, Encoder H2 option
MIS	340	B	5	M1	N0	41					Motor 14,0 mm shaft, 1 pcs M12 connector, 80V driver
MIS	342	B	5	M7	N0	76					Motor 14,0 mm shaft, 4 pcs M12 connectors, 80V controller, DeviceNet, Encoder H2 option

See also JVL's product part number builder using the following link:

<http://www.jvl.dk/ppnb/ppnb.htm>



## 17.1 CE Declaration of Conformity

### ***EU - Declaration of Conformity***

#### **Manufacturer**

Company Name: JVL Industri Elektronik A/S  
Address: Bregnerødvej 127  
DK-3460 Birkerød  
Denmark  
Telephone: +45 45 82 44 40  
E-mail: jvl@jvl.dk  
Web: www.jvl.dk

**Hereby declares that**

#### **Product**

No.: MIS231, 232 and 234  
Name: Integrated Hybrid stepper motor  
Type: Main no. followed by R, S, T or Q incl. subversions

- is in conformity with:

DIRECTIVE 2014/30/EU OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 26 February 2014 on the harmonisation of the laws of the Member States relating to electromagnetic compatibility

and

DIRECTIVE 2014/35/EU OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 26 February 2014 on the harmonisation of the laws of the Member States relating to the making available on the market of electrical equipment designed for use within certain voltage limits

- is manufactured in accordance with the following standards:

*EN 61800-3 Adjustable speed electrical power drives systems - part 3:  
EMC product standard including specific test methods..*

July 2017



Bo V. Jessen  
Technical Director  
JVL Industri Elektronik A/S

LX0027-01GB

## 17.1 CE Declaration of Conformity

### ***EU - Declaration of Conformity***

#### **Manufacturer**

Company Name: JVL Industri Elektronik A/S  
Address: Bregnerødvej 127  
DK-3460 Birkerød  
Denmark  
Telephone: +45 45 82 44 40  
E-mail: jvl@jvl.dk  
Web: www.jvl.dk

#### **Hereby declare that:**

#### **Product**

No.: MIS340, MIS341, MIS342, and MIS343  
Name: Integrated Stepper Motor  
Sub-types: -C12wwnnnyx85, -C12wwnnnyx85,  
-C14wwnnnyx85, -C17wwnnnyx85,  
-C27wwnnnyx85, and -C31wwnnnyx85,  
(ww=connector configuration,  
nnn=internal option module,  
yx=optional encoder options)

- is in conformity with:

DIRECTIVE 2014/30/EU OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of  
26 February 2014 on the harmonisation of the laws of the Member States relating to electromagnetic  
compatibility

and

DIRECTIVE 2014/35/EU OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL  
of 26 February 2014 on the harmonisation of the laws of the Member States relating to the making  
available on the market of electrical equipment designed for use within certain voltage limits

- is manufactured in accordance with the following standards:

*EN 61800-3 Adjustable speed electrical power drives systems - part 3:  
EMC product standard including specific test methods..*

Maj 2016



Bo V. Jessen  
Technical Director  
JVL Industri Elektronik A/S

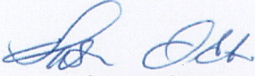
LX0023-02GB



## 17.2 Vibrationtest certificates MIS34x



### Mechanical assessment sheet no. 1336

<b>DELTA client</b> JVL Industri Elektronik A/S Blokken 42 3460 Birkerød Denmark	<b>DELTA project no.</b> T207608
<b>Product identification</b> Series MIS34x: MIS340 MIS341 MIS342	
<b>DELTA report(s)</b> DELTA project no. T207608, DANAK-19/13991 Revision 1	
<b>Other document(s)</b> “The MIS34x integrated stepper motor family”, 20140214, JVL Industri Elektronik A/S – BVJ 12.5.2 Physical dimensions MIS340, MIS341 and MIS342, JVL Industri Elektronik A/S – User Manual – Integrated Stepper Motors MIS23x, 34x, 43x 12.2 MIS34x Technical Data, , JVL Industri Elektronik A/S – User Manual – Integrated Stepper Motors MIS23x, 34x, 43x	
<b>Conclusion</b> The MIS342C14EPH385 has been tested according to the below listed standards. The test results are given in the DELTA report listed above. The tests were carried out as specified and neither malfunctions nor mechanical damages were detected.  IEC 60068-2-6: 2007, Test Fc; 5 – 25 Hz: ±1.6 mm, 25 - 500 Hz: 4 g, 1.0 oct./min., 3 x 10 sweep cycles IEC 60068-2-27, Test Ea, Shock; 15 g, 30 ms, 6 x 1000 shocks  The MIS34x integrated stepper motor family covers a number of family members ie. MIS340, MIS341, and MIS342. The tested MIS342C14EPH385 is worst case since the motor length of this family member is larger than the rest of the family and it is equipped with all extra options available and relevant for the vibration and shock tests i.e. ethernet and absolute multiturn encoder option.  Based on the documentation mentioned above, the test results are considered to be representative for smaller family members, MIS340, MIS341, and MIS342 models without options.	
<b>Date</b>  Hørsholm, 7 October 2014	<b>Assessor</b>   Susanne Otto B.Sc.E.E., B.Com (Org)



**A**

**A\_SOLL** 155, 207–211, 247, 250  
**Abort SDO** 265  
**Acc\_Emerg** 162  
**Acceleration factor** 256  
**Address, CANopen** 233  
**Address, MacTalk** 57  
**Afzup\_ConfMax** 169  
**Afzup\_ConfMin** 168  
**Afzup\_Filter** 169  
**Afzup\_MaxSlope** 169  
**Afzup\_ReadIndex** 168  
**Afzup\_WriteBits** 168  
**An** 166  
**Analog input filters** 23  
**AnalogFiltered** 167  
**AnalogIn** 167  
**Analogue inputs** 22  
**Auto correction** 72  
**Available\_IO** 183  
**B**  
**Baud rate** 159, 179, 226, 235, 280–281  
**Binary command** 221  
**Boot up telegram** 261  
**Bootloader\_Ver** 183  
**Brakes and shaft reinforcement** 298  
**Busvol** 167  
**C**  
**Cables** 37, 40, 43, 46, 296  
**Cabling** 234  
**Calculator (basic)** 222  
**Calculator (options)** 223  
**CAN A** 233  
**CAN B** 233  
**CAN bus connectors** 236  
**CANopen** 30, 142, 183, 226, 229–244, 247–249, 251–258, 260–270  
**CAN bus connectors** 236  
**CanOpen Explorer** 235, 237–241  
**Communication test** 237  
**Connecting the SMC75 Controller to the CAN bus** 234  
**DS-301 device profiles** 260  
**Node id and baud rate** 235  
**CANopen network** 230  
**Capacitor** 13  
**CE requirements** 313  
**Checksum** 182  
**CiA membership** 230  
**Clear errors** 57

**COB-ID** 233, 247, 250  
**Command** 159  
**Command timing** 302  
**Conditional jump (multiple inputs)** 214  
**Conditional jump (single input)** 213  
**Confidence alarms** 26  
**Confidence check** 25  
**Connecting the SMC75 Controller to the CAN bus** 234  
**Connections**  
**MIS23x** 34  
**SMC75** 278  
**Connectors** 35–46  
**MI2** 35–46  
**Control voltage** 15  
**CVI control voltage** 15  
**D**  
**Declaration of Conformity** 313–314  
**Digital inputs** 20  
**Dimensions** 288  
**Direction inputs** 118  
**Download SDO** 263  
**DS-301** 242  
**DS301 specified Communications objects** 242  
**DSP-402 Support** 252  
**E**  
**EDS file** 234  
**EMCY** 243  
**Emergency object** 243  
**Enable and Disable PDOs** 244  
**Encoder\_Pos** 157  
**Encoder\_Type** 168  
**End-of-travel inputs** 102  
**Enter safe mode** 305  
**Err\_Bits** 29, 163  
**Error acceleration** 132  
**Error Control Services** 268  
**Error handling** 131  
**Error output** 91  
**Error\_Mask** 177  
**Errors, clearing** 57  
**Exit safe mode** 306  
**Expansion modules**  
**MAC00-B1/B2/B4** 36–37, 39–40, 42–43, 45–46  
**Ext\_Encoder** 184  
**Ext\_Encoder\_Vel** 185–195  
**F**  
**Factors** 255  
**Fbus\_Baud** 184

- 
- Fbus\_Node Id 184
  - Filtering 25
  - Filters 32, 57
  - Filters, analog input 23
  - FilterStatus 169
  - Flash 57
  - Flwerr 158
  - Flwerrmax 158
  - Function description 124
  - Fuse dimensioning 16
  - G**
  - Galvanic isolation 19, 22, 29
  - Gear mode 117, 124
  - GEAR1 9, 113, 154, 157
  - GEAR2 9, 154, 157
  - Ground 19
  - Grounding 35, 38, 41, 44
  - Grounding, power supply 14
  - Group write register 307
  - Group\_Id 180
  - Group\_Seq 180
  - H**
  - Hardware\_Rev 182
  - Heartbeat 268–269
  - Home sensor 127
  - Home\_Bits 173
  - Homemode 164
  - Homing mode 257
  - I**
  - In physical position output 91
  - In position output 91
  - Index\_Offset 172–173
  - Inpos\_Mask 177
  - Input\_Filter\_Cnt 177
  - Input\_Filter\_Mask 177
  - Inputs 158
    - Analogue 22
    - Digital 20
    - End-of-travel 102
    - Multifunction I/O 124
    - Quadrature input 124
    - SMC75 18
    - Step pulse and direction 118
  - Interface
    - RS485 31
  - losetup 158, 174
  - IP67 37, 40, 43, 46
  - J**
  - Jump 213
  - Jump according to a comparison 224
  - Jump according to a register in the MAC motor 217
  - Jumps 213–214, 217, 224
  - L**
  - Life Guarding 268
  - M**
  - M12 35–46
  - MAB23x-01 298
  - MAB23x-02 298
  - MAC00-B1/B2/B4 Expansion Modules 36–37, 39–40, 42–43, 45–46
    - MAC00-B4 cables 37, 40, 43, 46
  - MacTalk 55–56, 60–63
  - Main Loop Time 226
  - Max\_P\_Ist 162
  - Max\_Voltage 182
  - Min bus voltage 132
  - Min\_Busvol 167
  - Min\_P\_Ist 161
  - MIS23x connections 34
  - MLT 226
  - MODE\_REG 247, 250
  - Mode\_Reg 153, 240, 247, 250
  - Modes of operation 9, 113, 206, 257
    - Gear mode 117
    - Passive mode 114
    - Positioning mode 116
    - Velocity mode 115
    - Zero search mode 125–129
  - Motor Connections 302
  - Motortype 181
  - Move (Absolute) 210
  - Move (Relative + set outputs) 209
  - Move (Relative + velocity change at a distance) 208
  - Move (Relative) 207
  - Move (Sensor) 211
  - Move current 70
  - Move operations 206
  - Multi-Master capability 232
  - My\_Addr 180
  - N**
  - Negative limit 102
  - NL\_Mask 175
  - NL, negative limit 102
  - NMT (Network Management services) 267
  - Node address 233
  - Node Guarding/Life Guarding 268
  - Node id 235
  - No-loss bus arbitration 232
-

- 
- Notsaved 183
  - NPN output 19
  - O**
  - Object dictionary 244
  - Object dictionary defined for DSP-402
    - support 253
  - Opening a file 58
  - Operating modes 9, 113–117, 125–129, 206, 257
    - Gear mode 124
  - Optical isolation 19, 22, 29
  - Option\_Bits 183
  - Ordering Information 309
  - Outputs 158
    - Error output 91
    - In position 91
    - In pyhsical position 91
    - SMC75 user outputs 28
  - P**
  - P- terminal 13
  - P\_Home 164
  - P\_Ist 156, 158, 179, 248, 251
  - P\_New 166, 179
  - P\_Soll 9, 72, 113, 223, 247, 250
  - P+ terminal 13
  - Passive mode 114
  - PDOs 244, 246, 248, 258, 262–263
  - PL, positive limit 102
  - PLC systems 91
  - Pn 166
  - PNP output 19
  - Position factor 255
  - Position limit min and max 132
  - Position mode 9
  - Positioning mode 116
  - Positioning-Speed Control 6, 8, 276–277
  - Positive limit 102
  - Power Supplies 297
  - Power Supply
    - Capacitor 13
  - Power supply
    - Grounding 14
  - Power supply,
    - SMC75 12
  - Profile position mode 257
  - Prog\_Vers 152, 197
  - Program comments 206
  - Program jumps 213–214, 217, 224
  - Program status and command 307
  - Programming 197–224
    - PSU05-045 297
    - PSU24-075 297
    - PSU24-240 297
    - PSU48-1000 297
    - PSU48-1500 297
    - PSU48-240 297
    - PSU48-800 297
    - Pull-up resistor 19
  - Q**
  - QuickStep motors 9
  - R**
  - Read register 304
  - Read register block 305
  - Receive PDOs 246, 258
  - Register overview 142
  - Registers 141–142, 154–159, 162–163, 168, 171, 178
    - A\_Soll 155, 207–211, 247, 250
    - Acc\_Emerg 162
    - Afzup\_ConfMax 169
    - Afzup\_ConfMin 168
    - Afzup\_MaxSlope 169
    - Afzup\_ReadIndex 168
    - Afzup\_WriteBits 168
    - An 166
    - AnalogFiltered 167
    - AnalogIn 167
    - Available\_IO 183
    - Bootloader\_Ver 183
    - Busvol 167
    - Checksum 182
    - Command 159
    - Encoder\_Pos 157
    - Encoder\_Type 168
    - Err\_Bits 29, 163
    - Error\_Mask 177
    - Ext\_Encoder 184
    - Ext\_Encoder\_Vel 185–195
    - Fbus\_Baud 184
    - Fbus\_Node Id 184
    - FilterStatus 169
    - Flwerr 158
    - Flwerrmax 158
    - GEAR1 9, 113, 154, 157
    - GEAR2 9, 154, 157
    - Group\_Id 180
    - Group\_Seq 180
    - Hardware\_Rev 182
-

- 
- Home\_Bits 173
  - Homemode 164
  - Index\_Offset 172–173
  - Inpos\_Mask 177
  - Input\_Filter\_Cnt 177
  - Input\_Filter\_Mask 177
  - Inputs 158
  - losetup 158, 174
  - Max\_P\_Ist 162
  - Max\_Voltage 182
  - Min\_Busvol 167
  - Min\_P\_Ist 161
  - Mode\_Reg 153, 240, 247, 250
  - Motortype 181
  - My\_Addr 180
  - NL\_Mask 175
  - Notsaved 183
  - Option\_Bits 183
  - Outputs 158
  - P\_Home 164
  - P\_Ist 156, 158, 179, 248, 251
  - P\_New 166, 179
  - P\_Soll 9, 72, 113, 223, 247, 250
  - Pn 166
  - Prog\_vers 152, 197
  - Register descriptions 154–159, 162–163, 168, 171, 178
  - Run\_Current 155, 247, 250
  - Serial\_Number 182
  - Setup\_Bits 174
  - Standby\_Current 156
  - Standby\_Time 156
  - Startmode 164
  - Statusbits 160
  - Temp 161
  - Tn 166
  - Turntable\_Mode 175
  - V\_Home 164
  - V\_Ist 156, 248, 251
  - V\_Soll 9, 113, 155, 207–211, 220, 240, 247, 250
  - V\_Start 153–154, 157
  - Vn 166
  - Remarks 206
  - Reset controller 306
  - Reset motor 57
  - Reset position 57
  - Resistors, termination 32
  - RS232/RS485 304
  - RS485 interface 31
  - Run\_Current 155, 247, 250
  - S**
  - Save in flash 57
  - Save position 218
  - Saving a file 58
  - Scope function 63
  - SDO (Service Data Objects) 263
  - Send FastMAC command 220–221
  - Serial communication 304
  - Serial\_Number 182
  - Set a register in the MIS motor 217
  - Set operation mode 206
  - Set outputs 212
  - Set position 219
  - Setup\_Bits 174
  - Short block length 232
  - Slope alarms 26
  - Slope limitation 25
  - SMC75 6, 8, 276–277
    - Inputs 18
  - SMC75 analogue inputs 22
  - SMC75 connector 278
  - SMC75 Power Supply 12
  - SMC75 user outputs 28
  - Specifications 280, 288
  - Standby current 70
  - Standby time 70
  - Standby\_Current 156
  - Standby\_Time 156
  - Startmode 164
  - Statusbits 160
  - Step pulse and direction inputs 118
  - Step pulse inputs 118
  - SYNC (Synchronisation Object) 266
  - T**
  - Technical Data 280, 288
  - Temp 161
  - Temperature protection 91
  - Termination 234, 236
  - Termination resistors 32
  - Tn 166
  - Torque 71
  - Transmit PDOs 248, 258
  - Trouble-shooting 293
  - Turntable\_Mode 175
  - U**
  - Unconditional jump 213
  - Upload SDO protocol 264
  - User outputs 28
-

**V**

V\_Home 164  
V\_1st 156, 248, 251  
V\_SOLL 220  
V\_Soll 9, 113, 155, 207–  
211, 220, 240, 247, 250  
V\_Start 153–154, 157  
Velocity accuracy 302  
Velocity encoder factor 256  
Velocity mode 9, 115, 257  
Vn 166  
Voltage Overload 22

**W**

Wait for (x) ms before continuing 215  
Wait for a register value before  
continuing 218  
Wait for an input combination before continuing  
(multiple inputs) 216  
Wait for an input combination before continuing  
(single input) 215  
Write Register 305  
Write to flash 306

**Z**

Zero search 219  
Zero search mode 125–129