IM232-MA
IM233-MA

Intelligent Brushless DC Motor

TECHNOSOFT

Intelligent Motor

Technical Reference

TECHNOSOFT

IM232-MA
IM233-MA
Technical Reference

P091.042.IM23x-MA.UM.1007

Technosoft S.A.
Buchaux 38
CH-2022 Bevaix, NE
Switzerland
Tel.: +41 (0) 32 732 5500
Fax: +41 (0) 32 732 5504
e-mail: contact@technosoftmotion.com
http://www.technosoftmotion.com/

Read This First

About This Manual

This book is a technical reference manual for the **IM23x-MA** family of intelligent servo intelligent motors, including the following products:

IM232-MA, CAN (p/n P042.001.E203) – Multi Axis Motor. Standard execution using Technosoft TMLCAN protocol on CANbus
IM232-MA, CANopen (p/n P042.001.E213) – Multi Axis Motor using CANopen protocol on CANbus
IM233-MA, CAN (p/n P042.001.E303) – Multi Axis Motor. Standard execution using Technosoft TMLCAN protocol on CANbus
IM233-MA CANopen (p/n P042.001.E313) – Multi Axis Motor using CANopen protocol on CANbus

There are currently two versions (revisions) of the afore-mentioned motors:

- IM23x-MA v1.1
- IM23x-MA v2.0

Both versions are fully electrical-, mechanical- and software-compatible. The changes between the two versions consist only in a different placement of the connectors on the backside of the motor.

In order to operate the **IM23x-MA** intelligent motors, you need to pass through 3 steps:

- ❑ **Step 1 Hardware installation**
- ❑ **Step 2 Drive setup** using Technosoft **EasySetUp** software for intelligent motor commissioning
- ❑ **Step 3 Motion programming** using one of the options:
  - ❑ A **CANopen master** (for the IM23x-MA CANopen version)

- ❑ The intelligent motor **built-in motion controller** executing a Technosoft Motion Language (**TML**) program developed using Technosoft **EasyMotion Studio** software
- ❑ A **TML_LIB motion library for PCs** (Windows or Linux)
- ❑ A **TML_LIB motion library for PLCs**
- ❑ A **distributed control** approach which combines the above options, like for example a host calling motion functions programmed on the intelligent motors in TML

This manual covers **Step 1** in detail. It describes the **IM23x-MA** hardware including the technical data, the connectors and the wiring diagrams needed for installation. The manual also presents an overview of the following steps, and includes the scaling factors between the real SI units and the intelligent motor internal units. For detailed information regarding the next steps, refer to the related documentation.

Notational Conventions

This document uses the following conventions:

- **TML** – Technosoft Motion Language
- **SI units** – International standard units (meter for length, seconds for time, etc.)
- **IU units** – Internal units of the intelligent motor
- **IM23x-MA** – all products described in this manual
- **IM23x-MA CANopen** – all CANopen executions from IM23x-MA family
- **IM23x-MA CAN** – IM23x-MA CAN standard executions

Related Documentation

MotionChip™ II TML Programming (part no. P091.055.MCII.TML.UM.xxxx) describes in detail TML basic concepts, motion programming, functional description of TML instructions for high level or low level motion programming, communication channels and protocols. Also give a detailed description of each TML instruction including syntax, binary code and examples.

MotionChip II Configuration Setup (part no. P091.055.MCII.STP.UM.xxxx) describes the MotionChip II operation and how to setup its registers and parameters starting from the user application data. This is a technical reference manual for all the MotionChip II registers, parameters and variables.

Help of the EasySetUp software – describes how to use **EasySetUp** to quickly setup any Technosoft intelligent motor for your application using only 2 dialogues. The output of EasySetUp is a set of setup data that can be downloaded into the intelligent motor EEPROM or saved on a PC file. At power-on, the intelligent motor is initialized with the setup data read from its EEPROM. With EasySetUp it is also possible to retrieve the complete setup information from a intelligent motor previously programmed. EasySetUp includes a firmware programmer with allows you to update your intelligent motor firmware to the latest revision. **EasySetUp can be downloaded free of charge from Technosoft web page**

CANopen Programming **(part no. P091.063.UM.xxxx)** – explains how to program the Technosoft intelligent intelligent motors using **CANopen** protocol and describes the associated object dictionary for the **DS-301** communication profile and the **DSP-402** device profile

Help of the EasyMotion Studio software – describes how to use the EasyMotion Studio to create motion programs using in Technosoft Motion Language (TML). EasyMotion Studio platform includes **EasySetUp** for the intelligent motor/motor setup, and a **Motion Wizard** for the motion programming. The Motion Wizard provides a simple, graphical way of creating motion programs and automatically generates all the TML instructions. *With EasyMotion Studio you can fully benefit from a key advantage of Technosoft intelligent motors – their capability to execute complex motions without requiring an external motion controller, thanks to their built-in motion controller.* **A demo version of EasyMotion Studio (with EasySetUp part fully functional) can be downloaded free of charge from Technosoft web page**

TML_LIB v2.0 **(part no. P091.040.v20.UM.xxxx)** – explains how to program in **C, C++,C#, Visual Basic or Delphi Pascal** a motion application for the Technosoft intelligent intelligent motors using TML_LIB v2.0 motion control library for PCs. The TML_lib includes ready-to-run examples that can be executed on **Windows** or **Linux** (x86 and x64).

TML_LIB_LabVIEW v2.0 **(part no. P091.040.LABVIEW.v20.UM.xxxx)** – explains how to program in **LabVIEW** a motion application for the Technosoft intelligent intelligent motors using TML_LIB_Labview v2.0 motion control library for PCs. The TML_Lib_LabVIEW includes over 40 ready-to-run examples.

TML_LIB_S7 **(part no. P091.040.S7.UM.xxxx)** – explains how to program in a PLC **Siemens series S7-300 or S7-400** a motion application for the Technosoft intelligent intelligent motors using TML_LIB_S7 motion control library. The TML_LIB_S7 library is **IEC61131-3 compatible**.

TML_LIB_CJ1 **(part no. P091.040.CJ1.UM.xxxx)** – explains how to program a PLC Omron series CJ1 a motion application for the Technosoft intelligent intelligent motors using TML_LIB_CJ1 motion control library for PCs. The TML_LIB_CJ1 library is **IEC61131-3 compatible**.

TechnoCAN **(part no. P091.063.TechnoCAN.UM.xxxx)** – presents TechnoCAN protocol – an extension of the CANopen communication profile used for TML commands

If you Need Assistance …

If you want to …	Contact Technosoft at …
Visit Technosoft online	World Wide Web: http://www.technosoftmotion.com/
Receive general information or assistance (see Note)	World Wide Web: http://www.technosoftmotion.com/ Email: contact@technosoftmotion.com
Ask questions about product operation or report suspected problems (see Note)	Fax: (41) 32 732 55 04 Email: hotline@technosoftmotion.com
Make suggestions about, or report errors in documentation (see Note)	Mail: Technosoft SA Buchaux 38 CH-2022 Bevaix, NE Switzerland

Contents

This page is empty

1. Safety information

Read carefully the information presented in this chapter before carrying out the intelligent motor installation and setup! It is imperative to implement the safety instructions listed hereunder.

This information is intended to protect you, the intelligent motor and the accompanying equipment during the product operation. Incorrect handling of the intelligent motor can lead to personal injury or material damage.

Only qualified personnel may install, setup, operate and maintain the intelligent motor. A "qualified person" has the knowledge and authorization to perform tasks such as transporting, assembling, installing, commissioning and operating intelligent motors.

The following safety symbols are used in this manual:

WARNING!	***SIGNALS A DANGER TO THE OPERATOR WHICH MIGHT CAUSE BODILY INJURY. MAY INCLUDE INSTRUCTIONS TO PREVENT THIS SITUATION***

CAUTION!	***SIGNALS A DANGER FOR THE DRIVE WHICH MIGHT DAMAGE THE PRODUCT OR OTHER EQUIPMENT. MAY INCLUDE INSTRUCTIONS TO AVOID THIS SITUATION***

CAUTION!	***INDICATES AREAS SENSITIVE TO ELECTROSTATIC DISCHARGES (ESD) WHICH REQUIRE HANDLING IN AN ESD PROTECTED ENVIRONMENT***

1.1. Warnings

WARNING!	***THE VOLTAGE USED IN THE DRIVE MIGHT CAUSE ELECTRICAL SHOCKS. DO NOT TOUCH LIVE PARTS WHILE THE POWER SUPPLIES ARE ON***

WARNING!	***TO AVOID ELECTRIC ARCING AND HAZARDS, NEVER CONNECT / DISCONNECT WIRES FROM THE INTELLIGENT MOTOR WHILE THE POWER SUPPLIES ARE ON***

WARNING!	**THE DRIVE MAY HAVE HOT SURFACES DURING OPERATION.**

WARNING!	**DURING INTELLIGENT MOTOR OPERATION, THE CONTROLLED MOTOR WILL MOVE. KEEP AWAY FROM ALL MOVING PARTS TO AVOID INJURY**

1.2. Cautions

CAUTION!	**THE POWER SUPPLIES CONNECTED TO THE DRIVE MUST COMPLY WITH THE PARAMETERS SPECIFIED IN THIS DOCUMENT**

CAUTION!	**TROUBLESHOOTING AND SERVICING ARE PERMITTED ONLY FOR PERSONNEL AUTHORISED BY TECHNOSOFT**

CAUTION!	***THE DRIVE CONTAINS ELECTROSTATICALLY SENSITIVE COMPONENTS WHICH MAY BE DAMAGED BY INCORRECT HANDLING. THEREFORE THE DRIVE SHALL BE REMOVED FROM ITS ORIGINAL PACKAGE ONLY IN AN ESD PROTECTED ENVIRONMENT***

To prevent electrostatic damage, avoid contact with insulating materials, such as synthetic fabrics or plastic surfaces. In order to discharge static electricity build-up, place the intelligent motor on a grounded conductive surface and also ground yourself.

2. Product Overview

2.1. Introduction

The **IM23x-MA** is a family of fully digital intelligent brushless motors, based on the latest DSP technology and they offer unprecedented intelligent motor performance combined with an embedded motion controller.

All intelligent motors perform position, speed or torque control and work in either single-, multi-axis or stand-alone configurations. Thanks to the embedded motion controller, the IM23x-MA intelligent motors combine controller, intelligent motor and PLC functionality in a single compact unit and are capable to execute complex motions without requiring intervention of an external motion controller. Using the high-level Technosoft Motion Language (**TML**) the following operations can be executed directly at intelligent motor level:

- Setting various motion modes (profiles, PVT, PT, electronic gearing[1] or camming[1], etc.)
- Changing the motion modes and/or the motion parameters
- Executing homing sequences[2]
- Controlling the program flow through:
  - Conditional jumps and calls of TML functions
  - TML interrupts generated on pre-defined or programmable conditions (protections triggered, transitions on limit switch or capture inputs, etc.)
  - Waits for programmed events to occur
- Handling of digital I/O and analogue input signals
- Executing arithmetic and logic operations
- Performing data transfers between axes
- Controlling motion of an axis from another one via motion commands sent between axes
- Sending commands to a group of axes (multicast). This includes the possibility to start simultaneously motion sequences on all the axes from the group
- Synchronizing all the axes from a network

Using **EasyMotion Studio** for TML programming you can really distribute the intelligence between the master and the intelligent motors in complex multi-axis applications, reducing both the development time and the overall communication requirements. For example, instead of trying to command each movement of an axis, you can program the intelligent motors using TML to execute complex motion tasks and inform the master when these tasks are done. Thus, for each axis control the master job may be reduced at: calling TML functions stored in the intelligent motor EEPROM (with possibility to abort their execution if needed) and waiting for a message, which confirms the TML functions execution.

Apart from a CANopen master, the IM23x-MA intelligent motors can also be controlled from a PC or PLC using the family of **TML_LIB** motion libraries.

For all motion programming options, the IM23x-MA commissioning for your application is done using **EasySetUp**.

[1] Optional for IM23x-MA CANopen executions
[2] Available only for IM23x-MA CAN executions

2.2. Key Features

- Fully digital intelligent brushless motor with built-in controller and high-level TML motion language
- Low system cost due to compactness and reduced wiring
- Available in 2 motor lengths, offering from 0.2 to 0.3 Nm of continuous torque (models IM232, IM233)
- Position, speed or torque control
- Various motion programming modes:
  - Position profiles with trapezoidal or S-curve speed shape
  - Position, Velocity, Time (PVT) 3rd order interpolation
  - Position, Time (PT) 1st order interpolation
  - Electronic gearing and camming[1]
  - External analogue or digital reference[1]
  - 33 Homing modes
- 9 Isolated digital I/Os (0…24V):
  - 3 general-purpose inputs / outputs:
    - 2 inputs
    - 1 output
  - 6 dedicated inputs / outputs:
    - ENABLE input
    - Positive Limit Switch input
    - Negative Limit Switch input
    - READY output
    - Master encoder outputs / Slave encoder inputs / Pulse & Direction inputs, differential signals
- 1 analog input, 0-5 V
- Integrated 2'000 bits/revolution quadrature encoder
- Integrated Protections for over current, over temperature, $i^2 t$, control error
- RS-232 serial interface (up to 115200 bps)
- CAN-bus 2.0B up to 1Mbit/s, with communication protocol:
  - CANopen[2] – compatible with CiA standards: DS301 and DSP402
  - TMLCAN[3] – compatible with all Technosoft intelligent motors with CANbus interface
- 1.5K × 16 internal SRAM memory

---

[1] Optional for IM23x-MA CANopen executions
[2] Available only for IM23x-MA CANopen executions
[3] Available only for IM23x-MA CAN executions

- 8K × 16 E^2ROM to store TML programs and data
- Nominal PWM switching frequency: 20 kHz
- Logic and Motor power supply: 12-48VDC;
- Operating ambient temperature: 0-40°C
- Hardware Protections:
    - All I/Os are ESD protected

2.3. IM23x-MA Dimensions

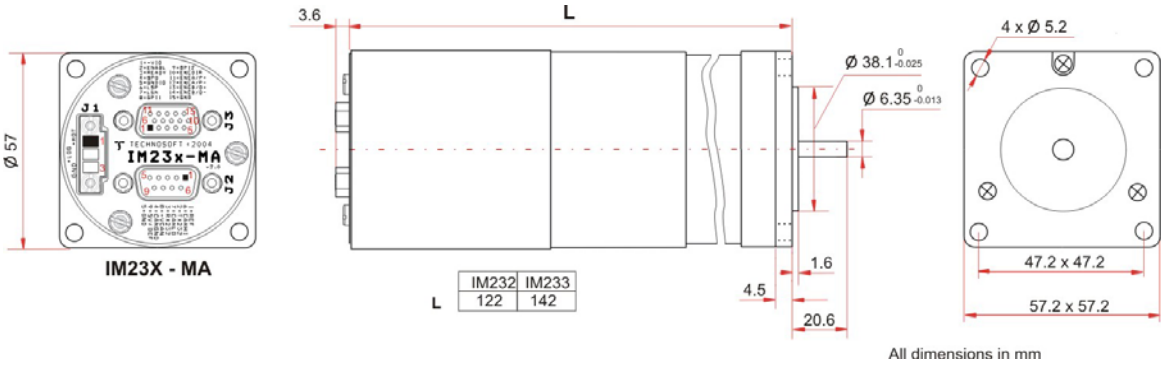


Figure 2.1. *IM23x-MA intelligent motor dimensions*

2.4. Electrical Specifications

All parameters were measured under the following conditions (unless otherwise specified):
T_{amb} = 25°C, logic supply (V_{LOG}) = 24V_{DC}, motor supply (V_{MOT}) =48V_{DC} ;
Supplies start-up / shutdown sequence: *-any-* ;
Shaft torque = 0.22Nm (IM232-MA) / 0.31Nm (IM233-MA); shaft speed = 4000rpm

Logic Supply Input

	Measured between +V_{LOG} and GND.	Min.	Typ.	Max.	Units
Supply voltage	Nominal values	12	24	48	V_{DC}
	Absolute maximum values, continuous †	0		55	V_{DC}
	Absolute maximum values, surge (duration ≤ 10mS) †	-0.5		65	
Supply current	Idle		80		mA
	Operating			120	mA
ESD Rating			±30		KV

Motor Supply Input

	Measured between +V_{MOT} and GND.	Min.	Typ.	Max.	Units
Supply voltage	Nominal values	12		48	V_{DC}
	Absolute maximum values, continuous, including ripple & braking-induced over-voltage	0		55	V_{DC}
	Absolute maximum values, surge (duration ≤ 10mS) †	-0.5		65	V
Supply current	Idle		0.5	1	mA
	Operating	-16.5	±5.5	+16.5	A
ESD Rating			±30		KV

I/O Supply Input

	All voltages referenced to GND_IO.	Min.	Typ.	Max.	Units
Supply voltage	Nominal values	12		24	V_{DC}
	Absolute maximum values, continuous †	-30		+30	V_{DC}
	Absolute maximum values, surge (duration ≤ 10mS) †	-50		35	V
Supply current	Normal operation		100	150	mA
Isolation voltage	Between GND and GND_IO			500	V_{RMS}

CAN-Bus Supply Input

	All voltages referenced to GND_CAN.	Min.	Typ.	Max.	Units
Supply voltage	Nominal values	12		24	V_{DC}
	Absolute maximum values, continuous †	-30		+30	V_{DC}
	Absolute maximum values, surge (duration ≤ 10mS) †	-50		35	V
Supply current	Normal operation		40	80	mA
Isolation voltage	Between GND and GND_IO			500	V_{RMS}

Digital Inputs (5V/DEF, ENCDIR)

	All voltages referred to *GND*	Min.	Typ.	Max.	Units
Input voltage	Logic "LOW"	0		1	V
	Logic "HIGH"	2		5	
	Absolute maximum, surge (duration ≤ 1S) †	-0.5		+5.5	
Input current	Logic "HIGH"; Internal pull-up to +5V	0	0	0	mA
	Logic "LOW" (strapped to GND)	8	10	12	

Isolated Digital Inputs (ENABL, LSP, LSN, GPI1, GPI2)

	All voltages referred to *GND_IO*	Min.	Typ.	Max.	Units
Input voltage	Logic "LOW"	-5	0	0.5	V
	Logic "HIGH"	12	24	30	
	Absolute maximum, surge (duration ≤ 1S) †	-12		+65	
Input current	Logic "HIGH"	4	10	25	mA
	Logic "LOW"			0.5	
Input impedance	Referred to GND_IO		2.5		KΩ
Input frequency		0		5	KHz
Pulse width	0→1→0 or 1→0→1	100			µS

Isolated Digital Outputs (READY, GPO)

	All voltages referred to *GND_IO*	Min.	Typ.	Max.	Units
Output voltage	Logic "HIGH"; Output crt. = 50mA	V_{LOG}-3	V_{LOG}-1		V
	Absolute maximum, continuous †	-0.5		V_{LOG} + 0.5V	
Output current	Logic "HIGH"; Output voltage ≥V_{LOG}-3V	50	80		mA
	Logic "LOW" (leakage current)			0.1	
Clamp diodes current	Output voltage ≤ (-0.5V) or ≥ (V_{LOG}+0.5V); Absolute maximum, continuous †	-100		+100	
Output frequency	External load 1KΩ to GND_IO	0		5	KHz
Pulse width	0→1→0 or 1→0→1; no external load ($5V_{PP}$)	100			µS

Differential Inputs / Outputs (ENCA/P+, ENCA/P-, ENCB/D+, ENCB/D-)

	All voltages referred to *GND*	Min.	Typ.	Max.	Units
Standards compliance		TIA/EIA-422			
Differential input voltage hysteresis		±0.1	±0.2	±0.5	V
Input voltage common mode range		-7		+12	V
	Absolute maximum, surge (duration ≤ 1S) †	-25		+25	
Input impedance			3		KΩ
Differential output voltage	External 120Ω load	±1.5	±3		V
Output current	External 120Ω load	±60			mA
	Absolute maximum, surge (duration ≤ 1S) †			±250	
Input / Output frequency		0		500	KHz

Analog Inputs (REF, DIR)

	All voltages referred to *GND*	Min.	Typ.	Max.	Units
Resolution			10		bits
Differential linearity	Guaranteed 10-bits no-missing-codes			0.09	% FS[1]
Offset error			±0.5	±2	% FS[1]
Gain error			±0.5	±3	% FS[1]
Bandwidth (-3dB)			250		Hz
Input voltage Operating range	REF	0		5	V
Input voltage Absolute Maximum, continuous †	REF	-5		30	V
Input impedance	REF		30		KΩ
External potentiometer	Recommended resistance	10	10	20	KΩ

RS-232

	All voltages referred to *GND*	Min.	Typ.	Max.	Units
Standards compliance		TIA/EIA-232-C			
Bit rate	Depending on software settings	9600		115200	Baud
ESD Protection	Human Body Model (100pF, 1.5 KΩ)			±15	KV
Input voltage	RX232 input	-25	-	+25	V
Output short-circuit withstand	TX232 output to GND	Guaranteed			

CAN-Bus

	All voltages referred to *GND_CAN*	Min.	Typ.	Max.	Units
Standards compliance		CAN-Bus 2.0B error active; ISO 11898-2			
Recommended transmission line impedance	Measured at 1MHz	90	120	150	Ω
Bit rate	Depending on software settings	125K		1M	Baud
Number of network nodes	Depending on software settings			64	-
ESD Protection	Human Body Model			±15	KV

Others

		Min.	Typ.	Max.	Units
Temperature	Operating	0		40	°C
	Storage (not powered)	-40		85	°C
Humidity (Non-condensing)	Operating	0		90	%RH
	Storage	0		100	%RH
Altitude / pressure	Altitude (referenced to sea level)		0 ÷ 1[6]	+4	Km
	Ambient Pressure	0.64	0.9 ÷ 1	4.0	atm
Weight	IM232-MA		0.9		kg
	IM233-MA		1.3		
Protection degree		IP31 (according to IEC529)			

Motor parameters

		IM232- MA	IM233- MA	Units
N° of pole		4	4	-
N° of phase		3	3	-
Rated voltage		36	36	V
Rated speed		4000	4000	rpm
Rated torque		0,22	0,31	Nm
Max peak torque		0,68	0,925	Nm
Torque constant		0,063	0,0561	Nm/A
Terminal Resistance		0,58	0,5	ohm
Line to line inductance		2,1	1,65	mH
B.E.M.F. at nominal speed		21	20,3	Vrms
Max peak current		9,8	15	A
Rotor Inertia		11,9	17,3	Kgmm

[1] "FS" stands for "Full Scale"

† Stresses beyond values listed under "absolute maximum ratings" may cause permanent damage to the device. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

T.B.D. = To be determined

[6] At altitudes over 1,000m, current and power rating are reduced due to thermal dissipation efficiency at higher altitudes. See Figure 2.6 – De-rating with altitude

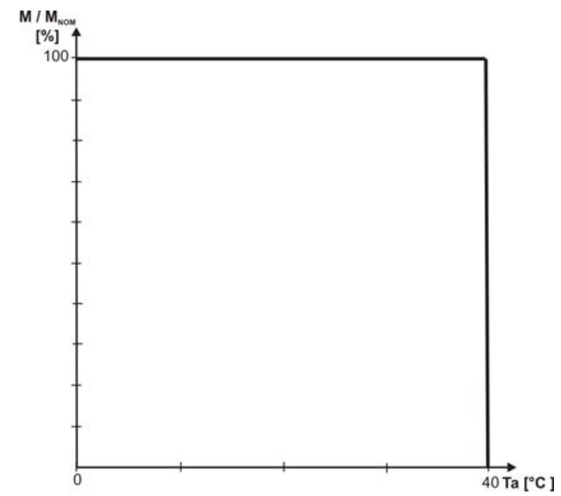


Figure 2.2. *De-rating with ambient temperature[7] [8] for IM232-MA*

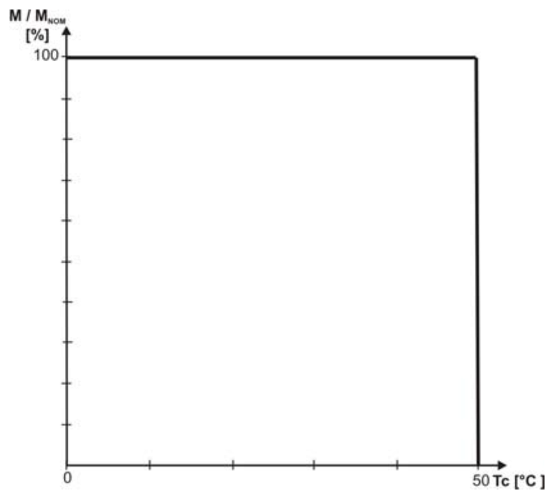


Figure 2.3. *De-rating with case temperature[9] for IM232-MA*

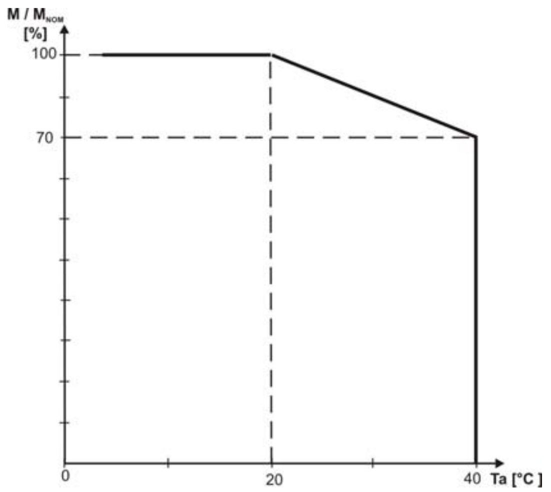


Figure 2.4. De-rating with ambient temperature for IM233-MA

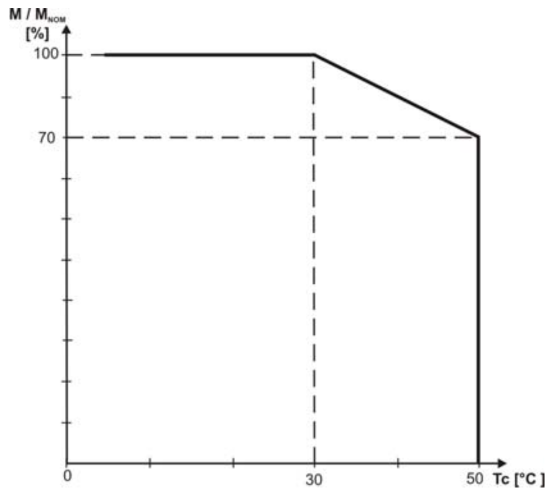


Figure 2.5. *De-rating with case temperature for IM233-MA*

[7] M_{NOM} – the nominal torque value
[8] Stand-alone operation, vertical mounting
[9] Fixed on metallic surface, vertical mounting. Temperature is measured at the contact area between the IDMx40 and the heat sink.

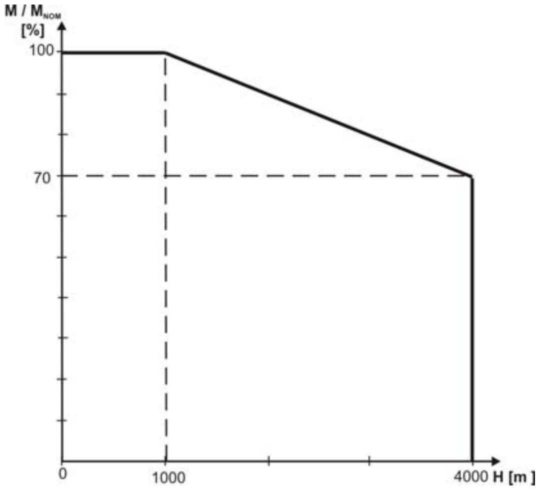


Figure 2.6. *De-rating with altitude*

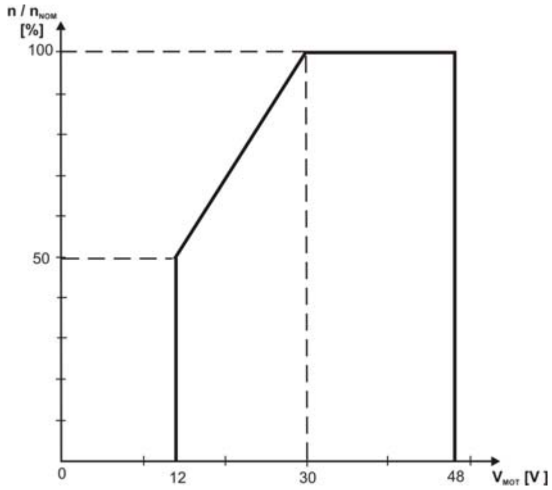


Figure 2.7. *Speed de-rating with the supply voltage*[1]

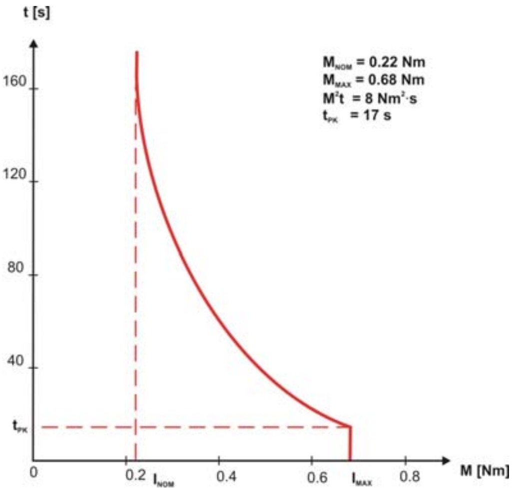


Figure 2.8. *Over-current diagram for IM232*

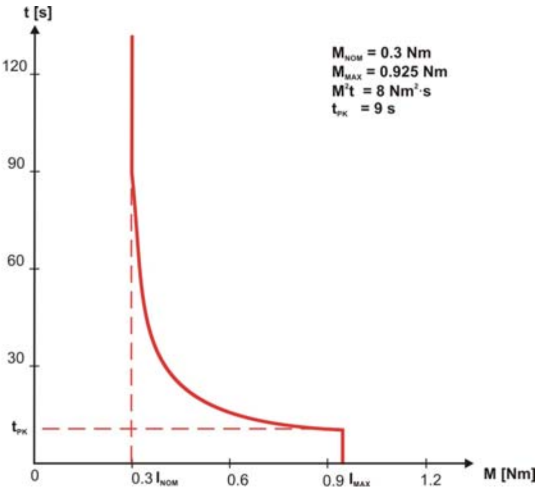


Figure 2.9. *Over-current diagram for IM233*

[1] n_{NOM} - the nominal speed

2.5. Torque-Speed Diagrams

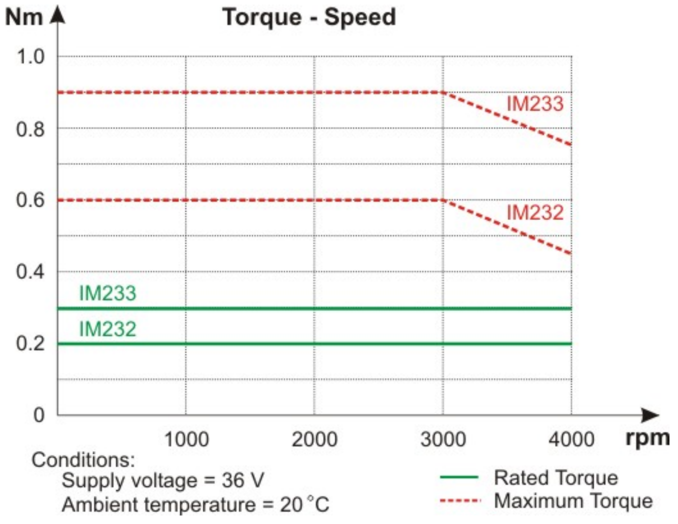


Figure 2.10. *Torque – speed diagrams*

3. Step 1. Hardware Installation

3.1. Mounting

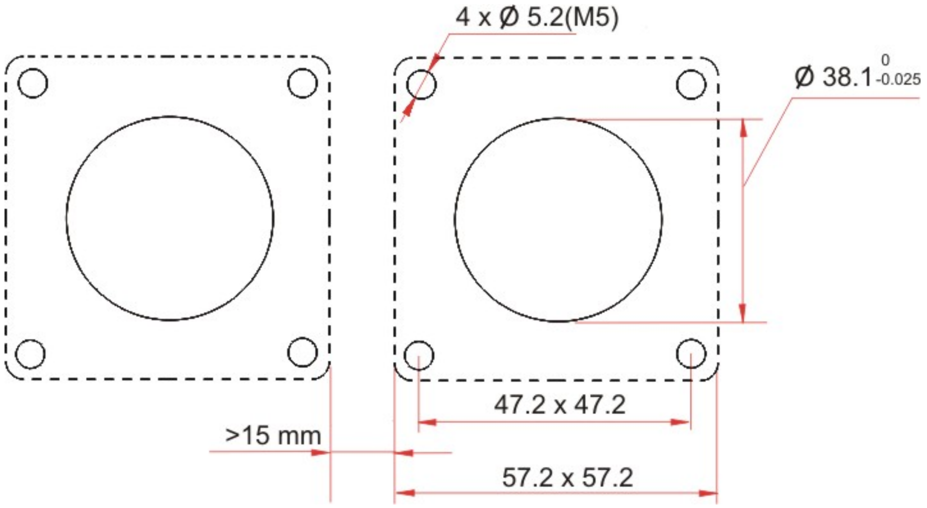


Figure 3.1. *Panel cut-out*

The IM23x-MA intelligent motor was designed to be cooled by natural convection. It can be mounted horizontally or vertically (see Figure 3.1). Leave at least 15mm between the intelligent motor and surrounding walls/intelligent motors, to allow for free air circulation.

3.2. Connectors and Connection Diagrams

3.2.1. Connectors Layout

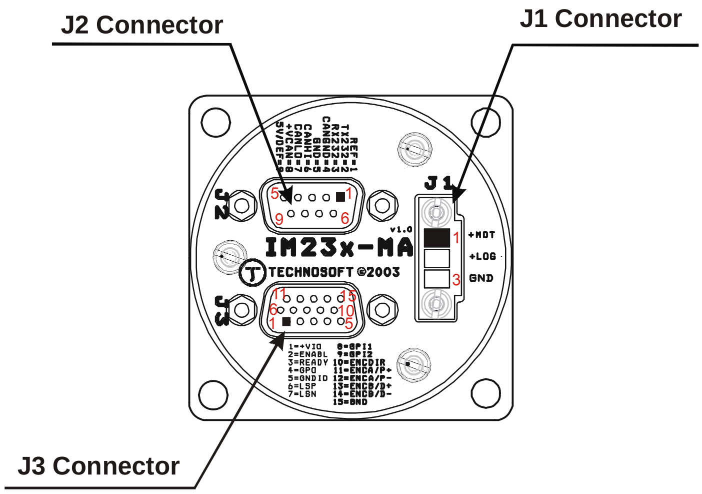


Figure 3.2. *IM23x-MA v1.1 intelligent motor drawing*

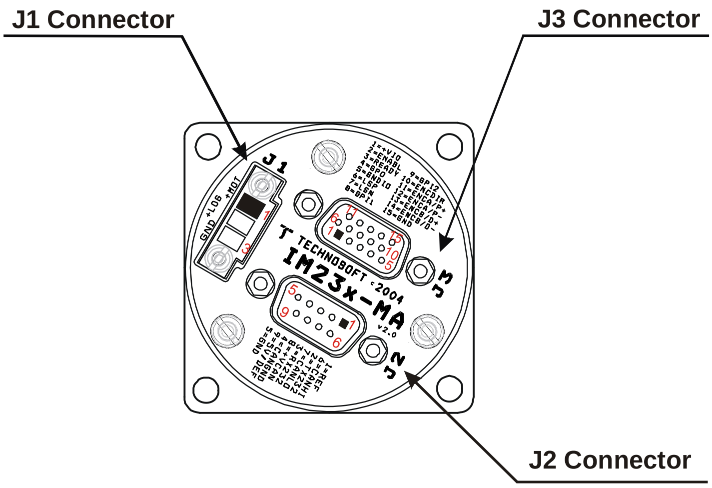


Figure 3.3. *IM23x-MA v2.0 intelligent motor drawing*

3.2.2. Identification Labels

Figure 3.4. *IM232-MA Identification Label*

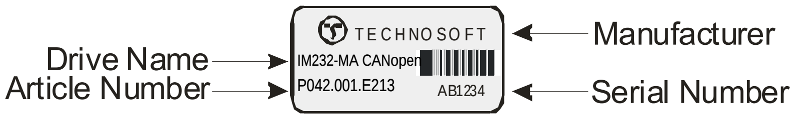


Figure 3.5. *IM232-MA CANopen Identification Label*

Figure 3.6. *IM233-MA Identification Label*

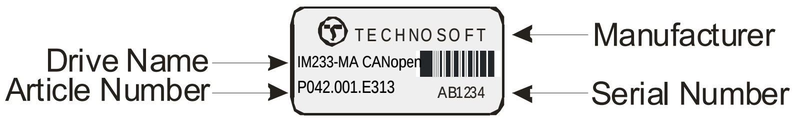


Figure 3.7. *IM233-MA CANopen Identification Label*

3.2.3. Supply - J1 Connector pinout

Pin	Pin name	TML name	Type	Function / Alternate function / Comments
1	**+MOT**	-	I	**Motor supply** - $12V_{MIN}$ … $48V_{MAX}$
2	**+LOG**	-	I	**Logic supply** - $24V_{TYP}$, $12V_{MIN}$ … $48V_{MAX}$
3	**GND**	-	-	**Ground**

3.2.4. Serial Communication - J2 Connector pinout

Pin	Pin name	TML name	Type	Function / Alternate function / Comments
1	**REF**	AD5	I	**Analog reference input** - 0…5 V
2	**Tx232**	-	O	**RS232 Transmit**
3	**Rx232**	-	I	**RS232 Receive**
4	**CANGND**	-	I	**CAN-Bus isolated ground**
5	**GND**	-	-	**Ground**
6	**CANHI**	-	-	**CAN-Bus Positive**; isolated - referred to "CANGND"
7	**CANLO**	-	-	**CAN-Bus Negative**; isolated - referred to "CANGND"
8	**+VCAN**	-	-	**CAN-Bus isolated supply**; $24V_{TYP}$, $12V_{MIN}$ … $36V_{MAX}$
9	**5V/DEF**	-	-	**+5V output** for potentiometer (5V @ 1mA) Not-Autorun - Strap to GND and reset, in order to stop automatical execution of TML program

3.2.5. I/O & Feedback - J3 Connector pinout

Pin	**Pin name**	TML name	Type	Function / Alternate function / Comments
1	**+VIO**	-	-	**Input / output isolated supply**
2	**ENABLe**	IN#16	I	**Enable digital** input
3	**READY**	OUT#12	O	**Ready digital** output; isolated; PNP output from "+VIO"
4	**GPO**	OUT#26	O	**General purpose** output; isolated; PNP output from "+VIO"
5	**GNDIO**	-	-	**Input/output isolated ground**
6	**LSP**	IN#2	I	**Limit switch "P"** input; isolated; 0 … 24V referred to "GNDIO"
7	**LSN**	IN#24	I	**Limit switch "N"** input; isolated; 0…24V referred to "GNDIO"
8	**GPI1**	IN#29	I	**General purpose** input; isolated; 0 … 24V referred to "GNDIO"
9	**GPI2**	IN#30	I	**General purpose** input; isolated; 0…24V referred to "GNDIO"
10	**ENCDIR**	-	-	**Select Encoder Output** or Master Encoder (Pulse) input; Strap to GND to select input
11	**ENCA/P+**	-	I/O	**Encoder A+** input / output (pulse+); RS422 (differential 0 … 5V)
12	**ENCA/P-**	-	I/O	**Encoder A-** input / output (pulse-); RS422 (differential 0 … 5V)
13	**ENCB/D+**	-	I/O	**Encoder B+** input / output (dir+); RS422 (differential 0 … 5V)
14	**ENCB/D-**	-	I/O	**Encoder B-** input . output (dir-); RS422 (differential 0 … 5V)
15	**GND**	-	-	**Ground**

3.2.6. Digital I/O connection

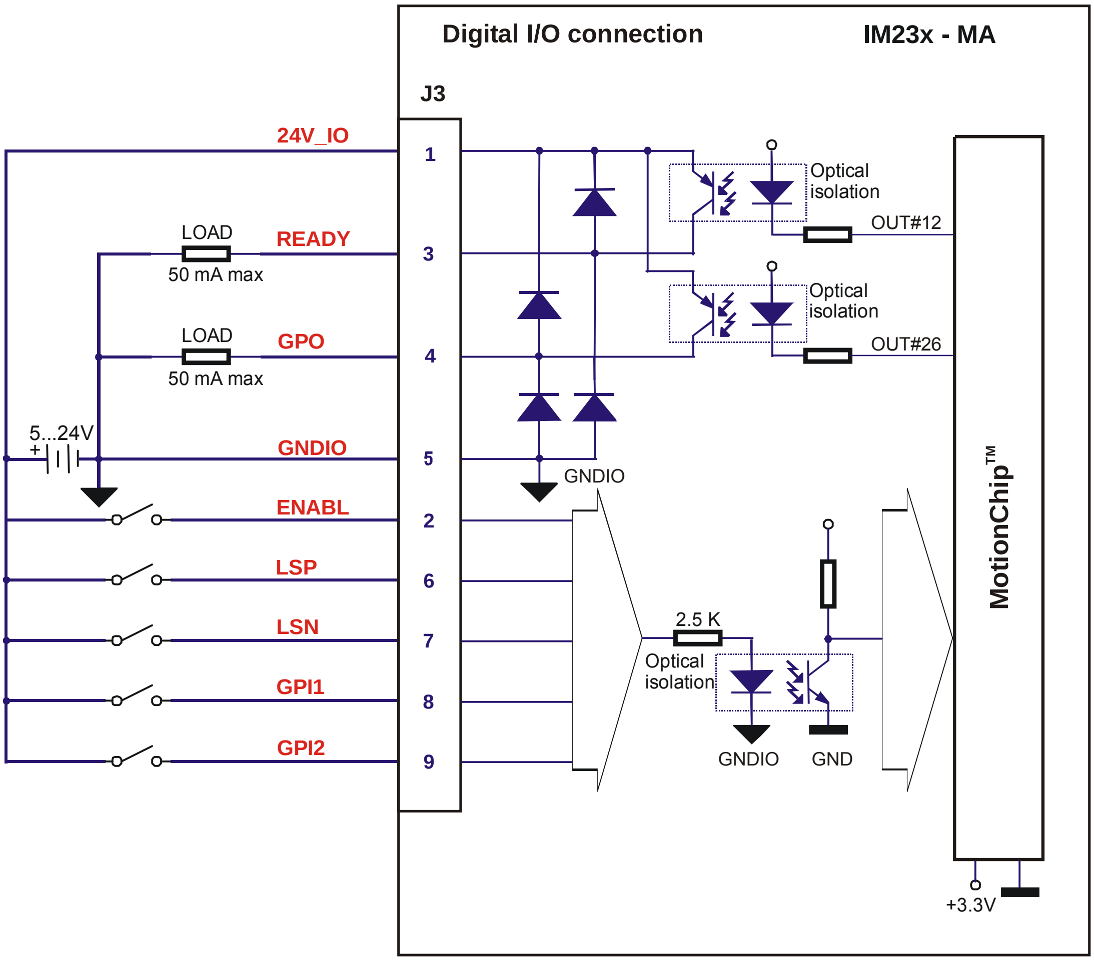


Figure 3.8. *Digital I/O connection*

3.2.7. Pulse & Directions inputs connection

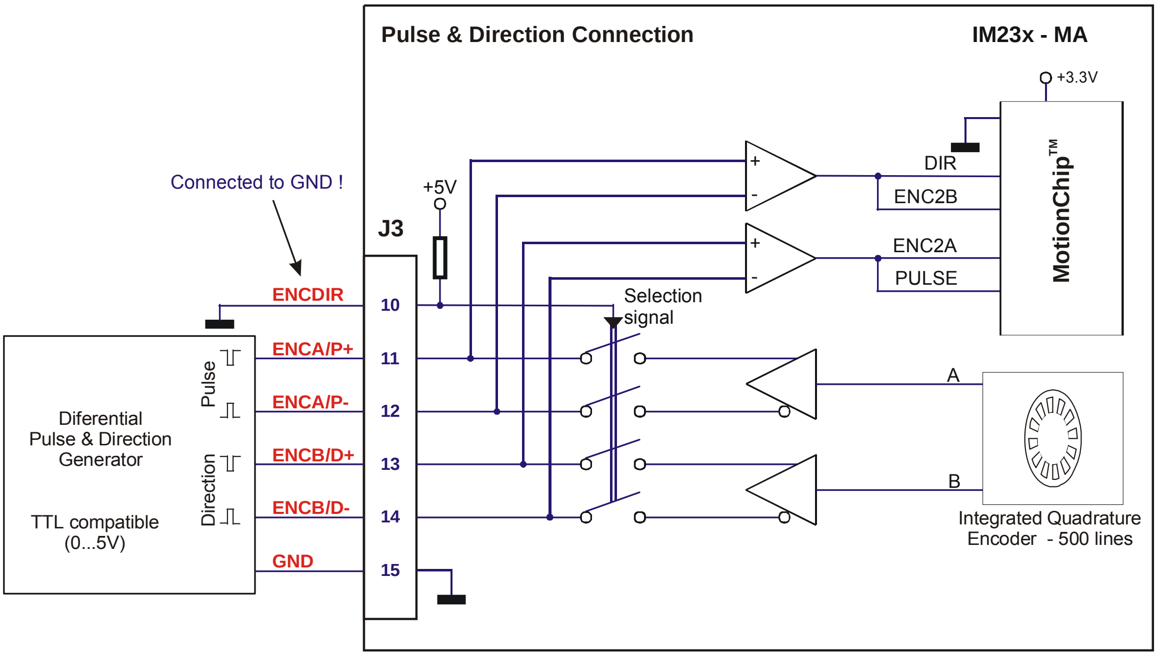


Figure 3.9. *Pulse & Direction inputs connection*

CAUTION !	**CONNECT ENCDIR PIN TO GND! ELSE SEVER DAMAGE WILL OCCUR ON THE IM23X-MA INTELLIGENT MOTOR AND PULSE & DIRECTION GENERATOR!**

3.2.8. Analog inputs connection

3.2.8.1 Analog inputs connection

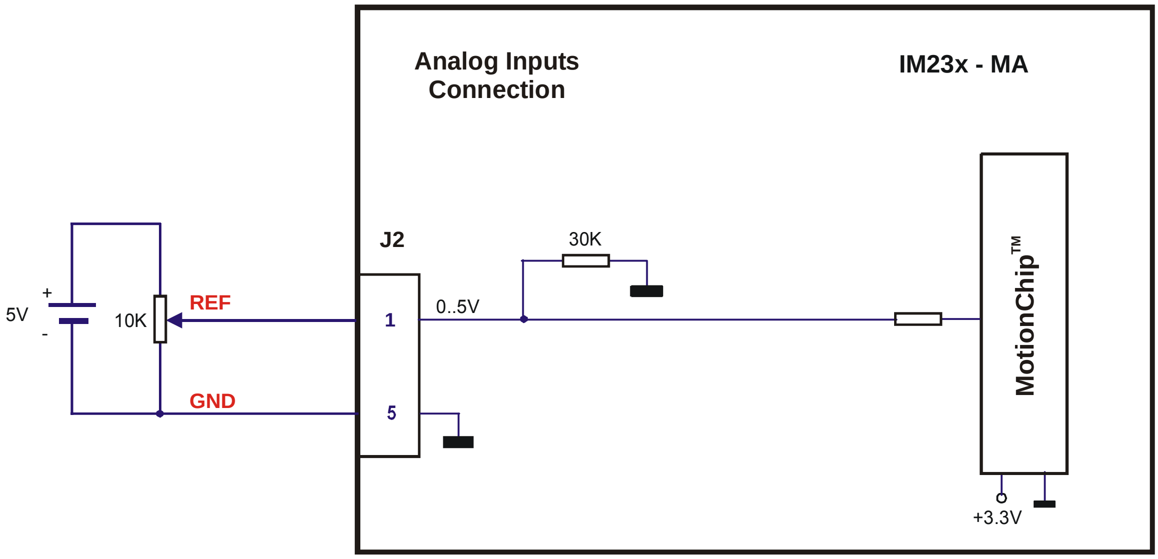


Figure 3.10. *Analog inputs connection*

3.2.8.2 Recommendation for wiring

Use a 2-wire shielded cable as follows: 1st wire connects the live signal to the intelligent motor positive input (+); 2nd wire connects the signal ground to the intelligent motor negative input (-). Connect the shield to ground at the IM23x-MA side, at pin 5 of connector J2, or using the Sub-D shield case.

3.2.9. Supply connection

3.2.9.1 Supply connection

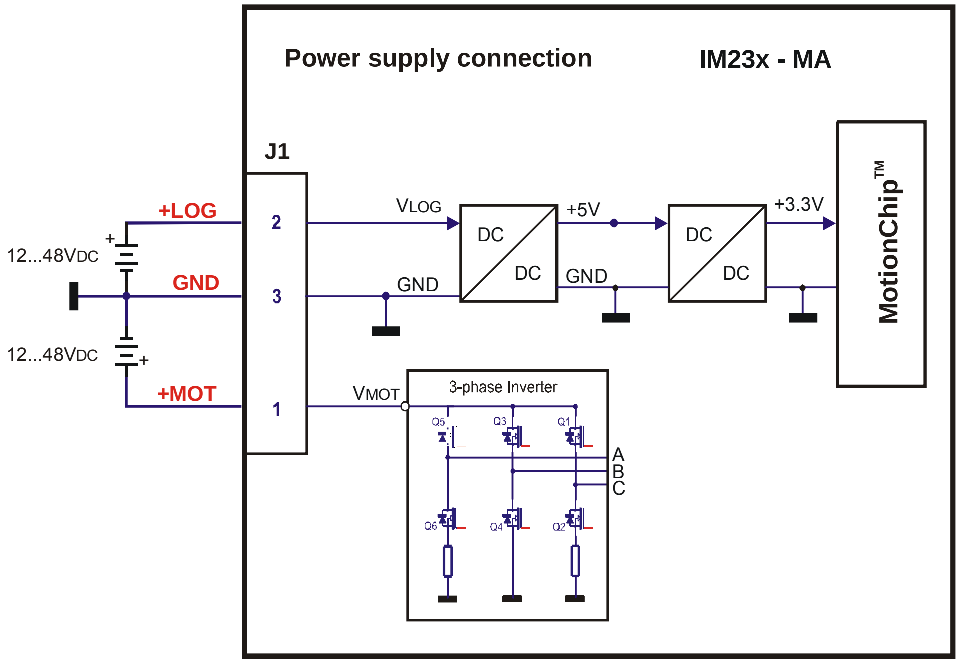


Figure 3.11. *Supply connection*

3.2.9.2 Recommendations for Supply Wiring

Use short, thick wires between the IM23x-MA and the motor power supply. If the wires are longer than 2 meters, use twisted wires for the supply and ground return. For wires longer than 20 meters, add a capacitor of at least 1000 μF (rated at an appropriate voltage) right on the terminals of the IM23x-MA.

3.2.9.3 Recommendations to limit over-voltage during braking

During abrupt motion brakes or reversals the regenerative energy is injected into the motor power supply. This may cause an increase of the motor supply voltage (depending on the power supply characteristics). If the voltage bypasses the U_{MAX} value, the intelligent motor over-voltage protection is triggered and the intelligent motor power stage is disabled.

In order to avoid this situation ***add a capacitor on the motor supply*** big enough to absorb the overall energy flowing back to the supply. The capacitor must be rated to a voltage equal or bigger than the maximum expected over-voltage and can be sized with the formula:

$$C \geq \frac{2 \times E_M}{U_{MAX}^2 - U_{NOM}^2} - C_{Drive}$$

where:

U_{MAX} - is the over-voltage protection limit expressed in [V]. You can read this value in the “Drive Info” dialogue, which can be opened from the “Drive Setup”.

C_{Drive} - is the intelligent motor internal capacitance (220 μF)

U_{NOM} - is nominal motor supply voltage expressed in [V]. You can read this value in the “Drive Info” dialogue, which can be opened from the “Drive Setup”.

E_M - the overall energy flowing back to the supply in Joules. In case of a rotary motor and load,

E_M can be computed with the formula:

$$E_M = \underbrace{\frac{1}{2}(J_M + J_L)\varpi_M}_{\text{Kinetic energy}} + \underbrace{(m_M + m_L)g(h_{initial} - h_{final})}_{\text{Potential energy}} - \underbrace{3I_M^2 R_{Ph} t_d}_{\text{Copper losses}} - \underbrace{\frac{t_d \varpi_M}{2} T_F}_{\text{Friction losses}}$$

where:

J_M – total rotor inertia = $119 \cdot 10^{-7}$ kgm^2 (IM232-MA) / $173 \cdot 10^{-7}$kgm^2 (IM233-MA)

J_L – total load inertia as seen at motor shaft after transmission [kgm^2]

ϖ_M – motor angular speed before deceleration [rad/s]

m_M – motor mass [kg] – when motor is moving in a non-horizontal plane

m_L – load mass [kg] – when load is moving in a non-horizontal plane

g – gravitational acceleration i.e. 9.8 [m/s^2]

$h_{\mathbf{initial}}$ – initial system altitude [m]

$h_{\mathbf{final}}$ – final system altitude [m]

I_M – motor current during deceleration [A_{RMS}/phase]

R_{Ph} – motor phase resistance [Ω]

$t_{\mathbf{d}}$ – time to decelerate [s]

T_F – total friction torque as seen at motor shaft [Nm] – includes load and transmission

In case of a linear motor and load, the motor inertia J_M and the load inertia J_L will be replaced by the motor mass and the load mass measured in [kg], the angular speed ϖ_M will become linear speed measured in [m/s] and the friction torque T_F will become friction force measured in [N].

Remark: *If the above computation of* E_M *can't be done due to missing data, a good starting value for the capacitor can be 10 000 µF / 100V.*

3.2.10. Serial RS-232 connection

3.2.10.1 Serial RS-232 connection

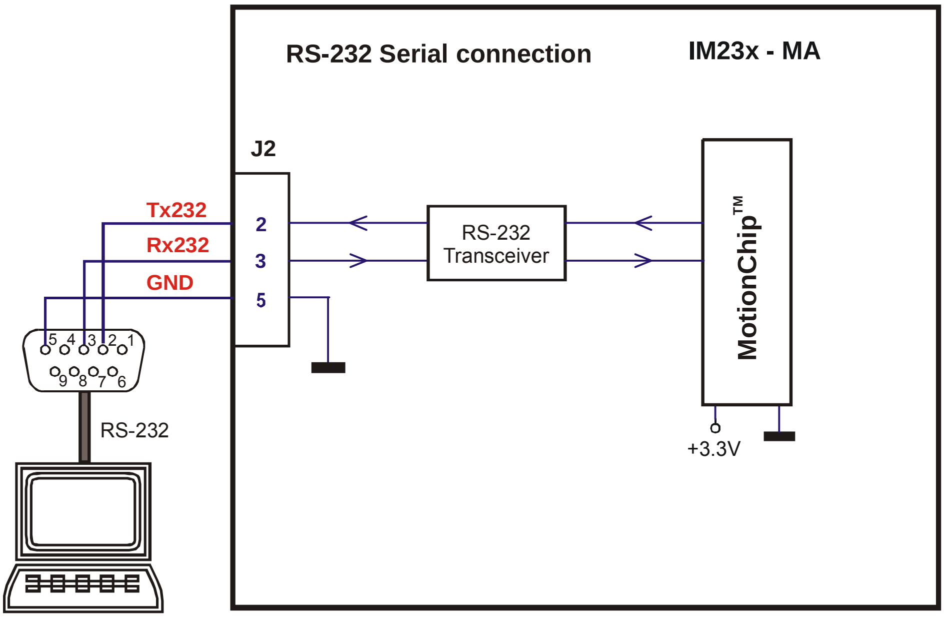


Figure 3.12. *Serial RS-232 connection*

3.2.10.2 Recommendation for wiring

a) If you build the serial cable, you can use a 3-wire shield cable with shield connected to BOTH ends. Do not use the shield as GND. The ground wire (pin 14 of J1) must be included inside the shield, like the RxD and TxD signals

b) Do not rely on an earthed PC to provide the IM23x-MA GND connection! The intelligent motor must be earthed through a separate circuit. Most communication problems are caused by the lack of such connection

c) Always power-off all the IM23x-MA supplies before inserting/removing the RS-232 serial connector.

3.2.11. CAN connection

3.2.11.1 CAN connection

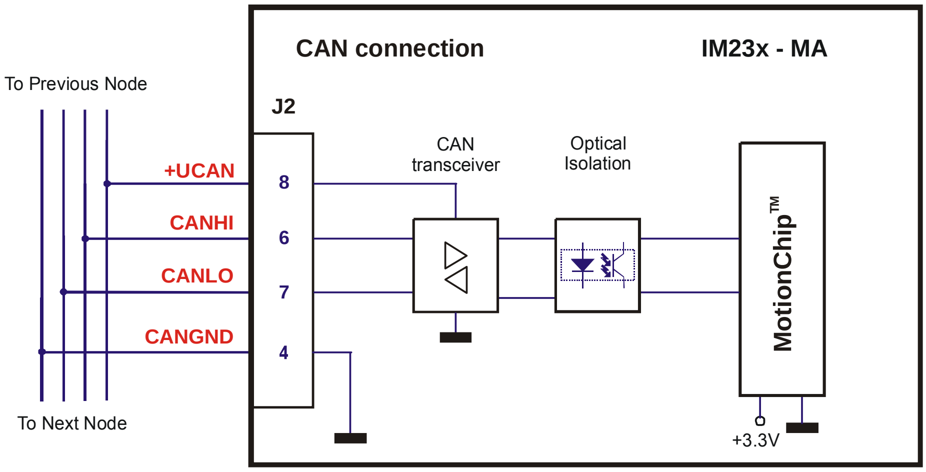


Figure 3.13. *CAN connection*

Remarks:

1. *The CAN network requires a 120-Ohm terminator. This is not included on the board. See Figure 3.22.*
2. *CAN signals are insulated from other IM23x-MA circuits. Thus, it requires an external supply to operate the CAN transceiver.*

3.2.11.2 Recommendation for wiring

a) Build CAN network using cables with 2-pairs of twisted wires (2 wires/pair) as follows: one pair for CAN_H with CAN_L and the other pair for CAN_V+ with CAN_GND. The cable impedance must be 105 ... 135 ohms (120 ohms typical) and a capacitance below 30pF/meter.

b) Whenever possible, use daisy-chain links between the CAN nodes. Avoid using stubs. A stub is a "T" connection, where a derivation is taken from the main bus. When stubs can't be avoided keep them as short as possible. For 1 Mbit/s (worst case), the maximum stub length must be below 0.3 meters.

c) The 120Ω termination resistors must be rated at 0.2W minimum. Do not use winded resistors, which are inductive.

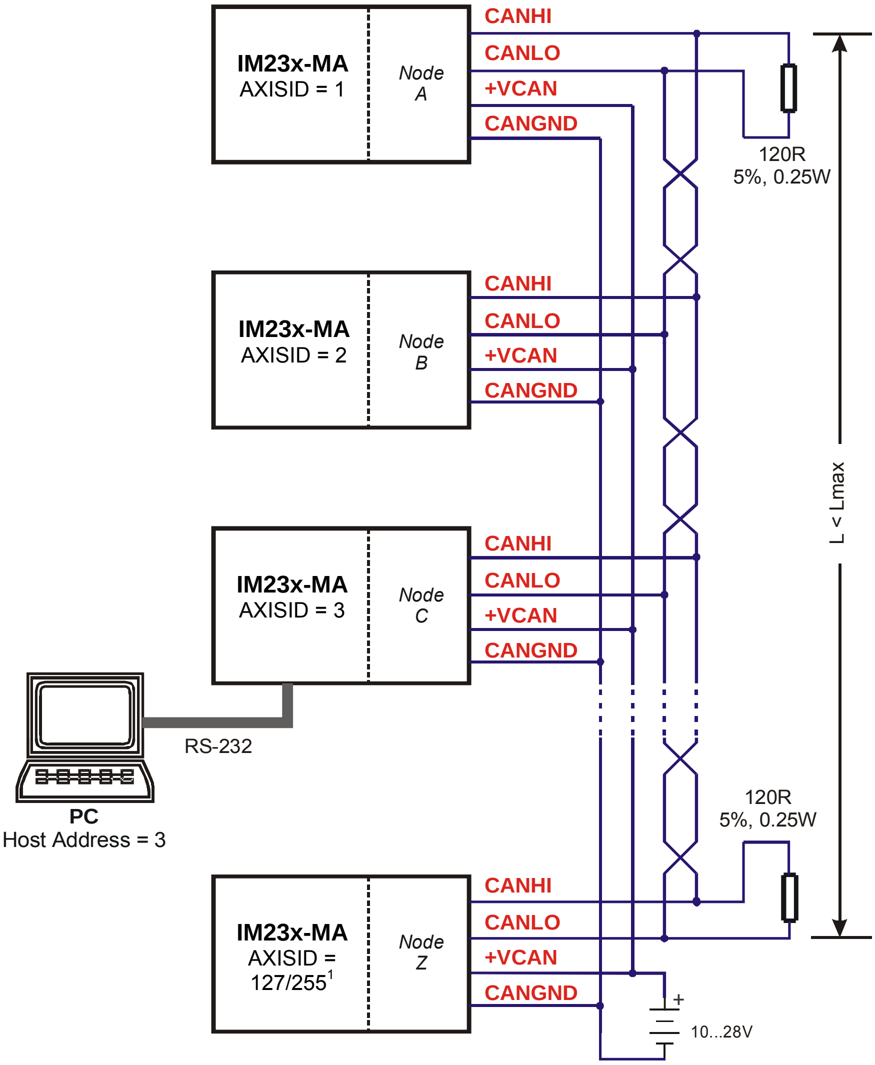


Figure 3.14. *Multiple-Axis CAN network*[1]

Remark: *The AxisID must be set by software, using instruction AXISID number.*

[1] The maximum value of the AXISID is 127 for the ISCMxx05 CANopen executions and 255 for ISCMxx05 CAN executions

3.2.12. Master - Slave encoder connection

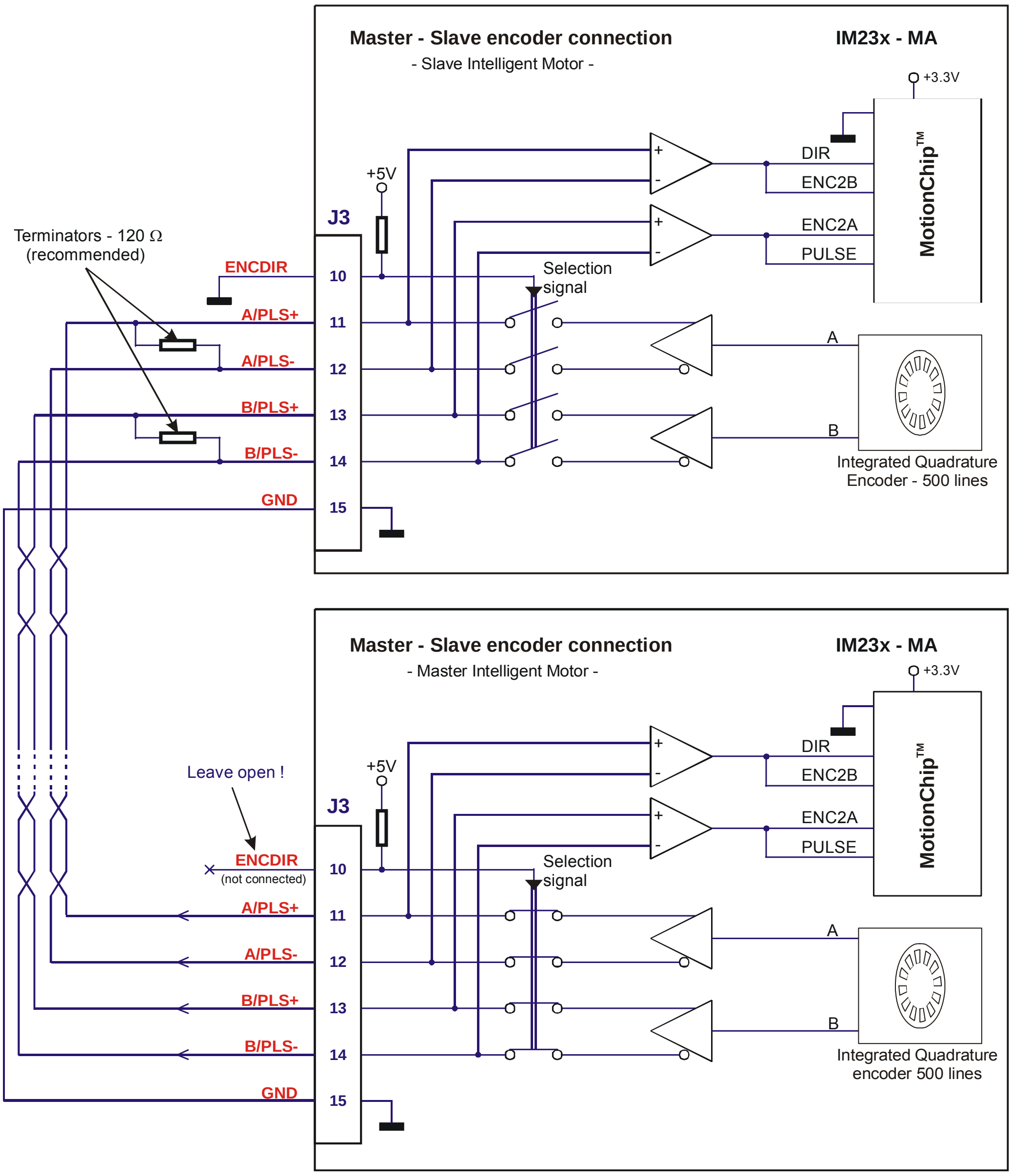


Figure 3.15. *Master – Slave encoder connection using second encoder input*

3.2.13. Connectors Type and Mating Connectors

Connector	Function	Producer	Board connector
J1	Supply	Phoenix Contact	MC1,5/3-STF-3,81[12]
J2	Serial Communication	Fischer Elektronik	DS-09-L
J3	I/O & Feedback	W+P Products	108-15-1-3-0

[12] The mating connector accepts wires of 0.14 … 1.5 mm^2 (AWG28 … AWG16)

4. Step 2. Drive Setup

4.1. Installing EasySetUp

EasySetUp is a PC software platform for the setup of the Technosoft intelligent motors. It can be downloaded ***free of charge*** from Technosoft web page. EasySetUp comes with an ***Update via Internet tool*** through which you can check if your software version is up-to-date, and when necessary download and install the latest updates. EasySetUp includes a firmware programmer through which you can update your intelligent motor firmware to the latest revision.

EasySetUp can be installed independently or together with **EasyMotion Studio** platform for motion programming using TML. You will need EasyMotion Studio only if you plan to use the advance features presented in Section 5.3 Combining CANopen /or other host with TML. A **demo version of EasyMotion Studio** including the **fully functional version of EasySetUp** can be downloaded free of charge from Technosoft web page.

On request, EasySetUp can be provided on a CD too. In this case, after installation, use the update via internet tool to check for the latest updates. Once you have started the installation package, follow its indications.

4.2. Getting Started with EasySetUp

Using EasySetUp you can quickly setup a intelligent motor for your application. The intelligent motor can be:

- directly connected with your PC via a serial RS 232 link
- any intelligent motor from a CANbus network where the PC is serially linked with one of the other intelligent motors.

The output of EasySetUp is a set of *setup data*, which can be downloaded into the intelligent motor EEPROM or saved on your PC for later use.

EasySetUp includes a set of evaluation tools like the Data Logger, the Control Panel and the Command Interpreter which help you to quickly measure, check and analyze your intelligent motor commissioning.

EasySetUp works with **setup** data. A **setup** contains all the information needed to configure and parameterize a Technosoft intelligent motor. This information is preserved in the intelligent motor EEPROM in the *setup table*. The setup table is copied at power-on into the RAM memory of the intelligent motor and is used during runtime. With EasySetUp it is also possible to retrieve the complete setup information from a intelligent motor previously programmed.

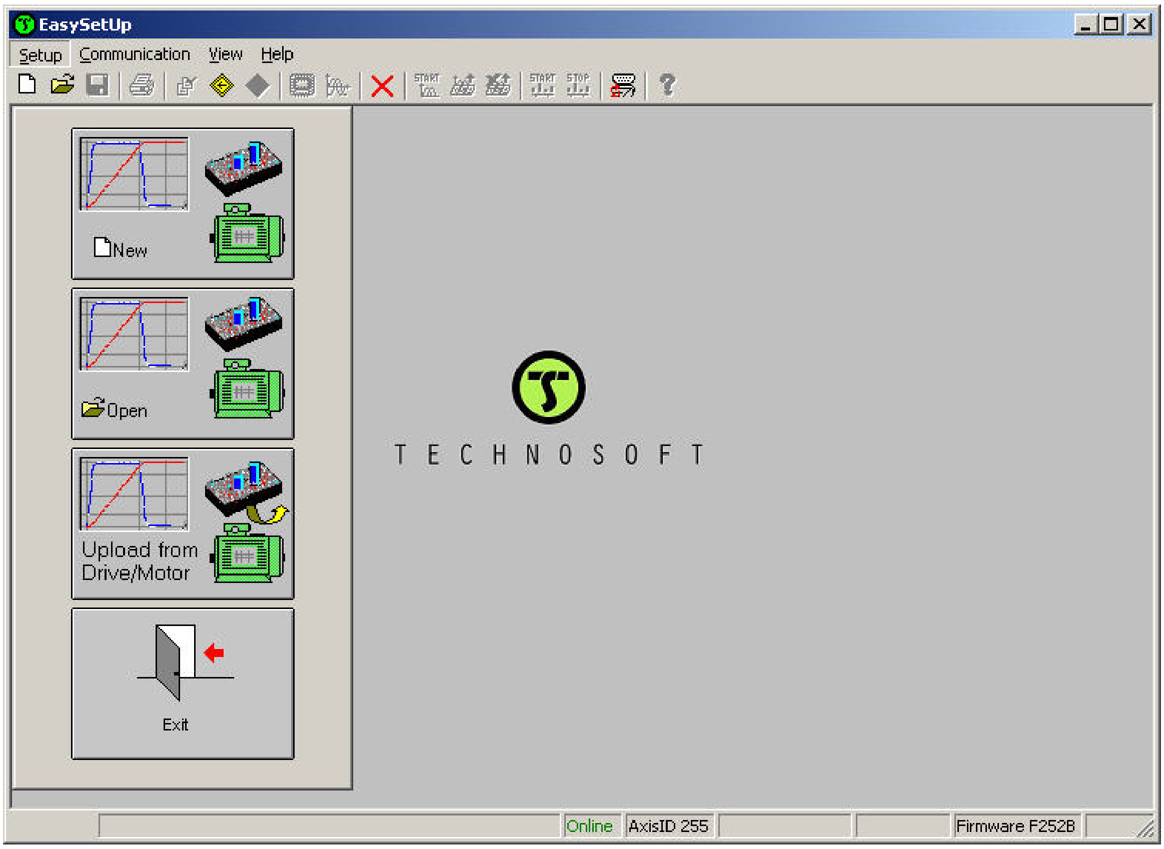
Note that with EasySetUp you do only your intelligent motor/motor commissioning. For motion programming you have the following options:

- Use a **CANopen** master (for IM23x-MA CANopen executions)
- Use **EasyMotion Studio** to create and download a **TML** program into the intelligent motor/motor memory

- Use one of the **TML_LIB** motion libraries to control the intelligent motors/motors from your host/master. If your host is a **PC**, TML_LIB offers a collection of high level motion functions which can be called from applications written in C/C++, Visual Basic, Delphi Pascal or LabVIEW. If your host is a **PLC**, TML_LIB offers a collection of function blocks for motion programming, which are **IEC61131-3 compatible** and can be integrated in your PLC program.
- **Implement** on your master the TML commands you need to send to the intelligent motors/motors using one of the supported communication channels. The implementation must be done according with Technosoft communication protocols.
- **Combine** TML programming at intelligent motor level with one of the other options (see Section 5.3)

4.2.1. Establish communication

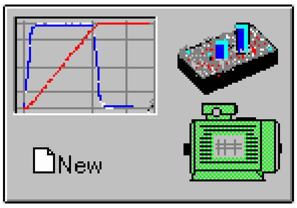
EasySetUp starts with an empty window from where you can create a **New** setup, **Open** a previously created setup which was saved on your PC, or **Upload** the setup from the intelligent motor/motor.



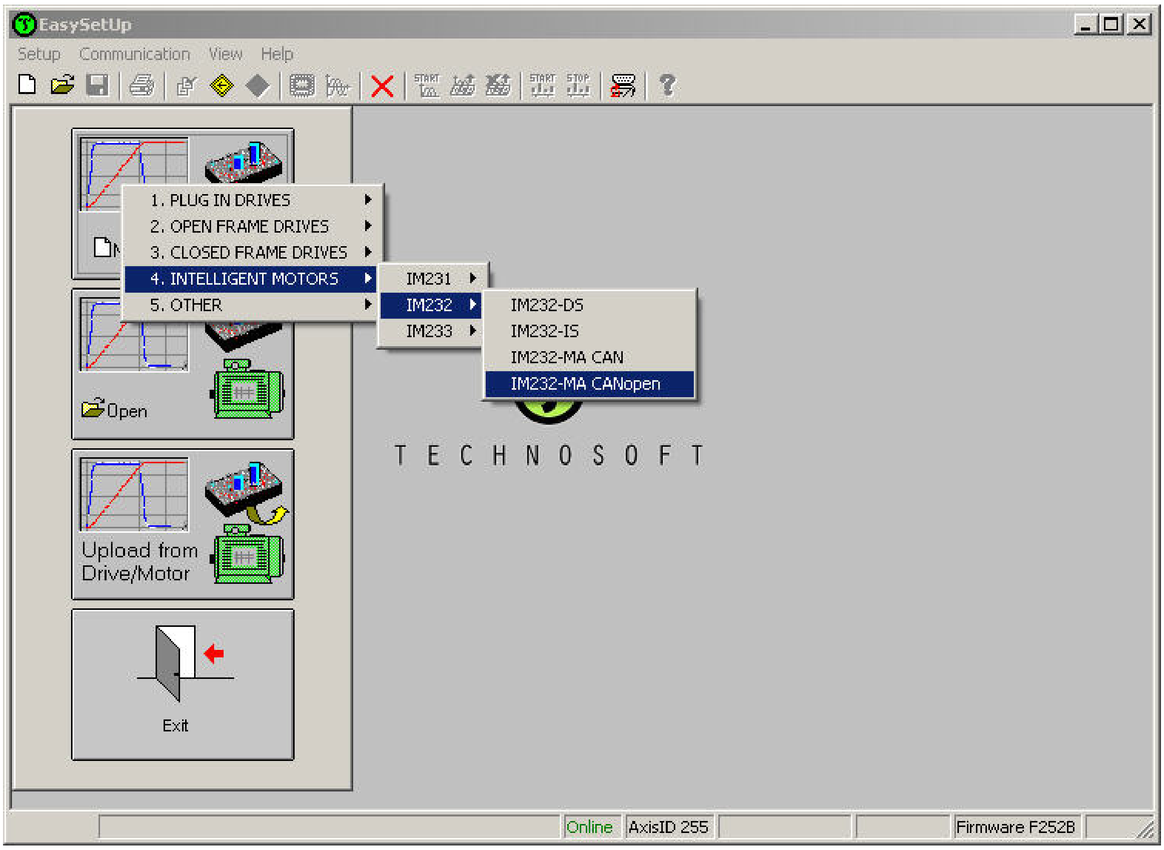
Before selecting one of the above options, you need to establish the communication with the intelligent motor you want to commission. Use menu command **Communication | Setup** to check/change your PC communication settings. Press the **Help** button of the dialogue opened. Here you can find detailed information about how to setup your intelligent motor and do the connections. Power on the intelligent motor, then close the Communication | Setup dialogue with OK. If the communication is established, EasySetUp displays in the status bar (the bottom line) the text "**Online**" plus the axis ID of your intelligent motor/motor and its firmware version. Otherwise the text displayed is "**Offline**" and a communication error message tells you the error type. In this case, return to the Communication | Setup dialogue, press the Help button and check troubleshoots

Remark: *When first started, EasySetUp tries to communicate via RS-232 and COM1 with a intelligent motor having axis ID=255 (default communication settings). If your intelligent motor is powered with all the DIP switches OFF and it is connected to your PC port COM1 via an RS-232 cable, the communication shall establish automatically. If the intelligent motor has a different axis ID and you don't know it, select in the Communication | Setup dialogue at "Axis ID of intelligent motor/motor connected to PC" the option* ***Autodetected****.*

4.2.2. Setup intelligent motor/motor

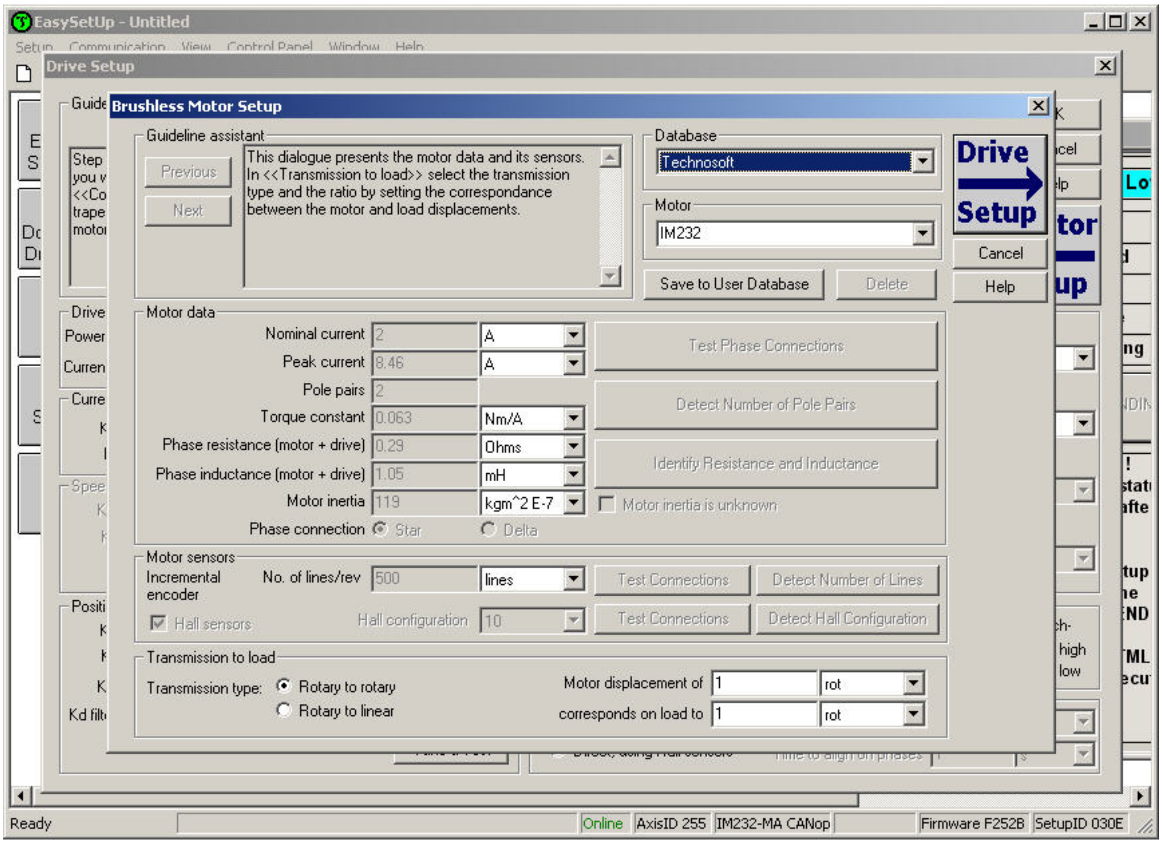
Press **New** button 

and select your intelligent motor type.



The selection continues with the type of the intelligent motor (for example: the IM232-MA CANopen).

The selection opens 2 setup dialogues: for **Motor Setup** and for **Drive setup** through which you can configure and parameterize a Technosoft intelligent motor, plus several predefined control panels customized for the product selected.



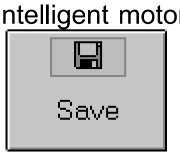
In the **Motor setup** dialogue you can see the data of the brushless motor and the associate sensor (incremental encoder).

In the **Drive setup** dialogue you can configure and parameterize the intelligent motor for your application. In each dialogue you will find a **Guideline Assistant**, which will guide you through the whole process of introducing and/or checking your data. Close the Drive setup dialogue with **OK** to keep all the changes regarding the motor and the intelligent motor setup.

4.2.3. Download setup data to intelligent motor/motor

Press the **Download to Drive/Motor** button 

to download your setup data in the intelligent motor/motor EEPROM memory in the *setup table*. From now on, at each power-on, the setup data is copied into the intelligent motor/motor RAM memory which is used during runtime.

It is also possible to **Save** 

the setup data on your PC and use it in other applications.

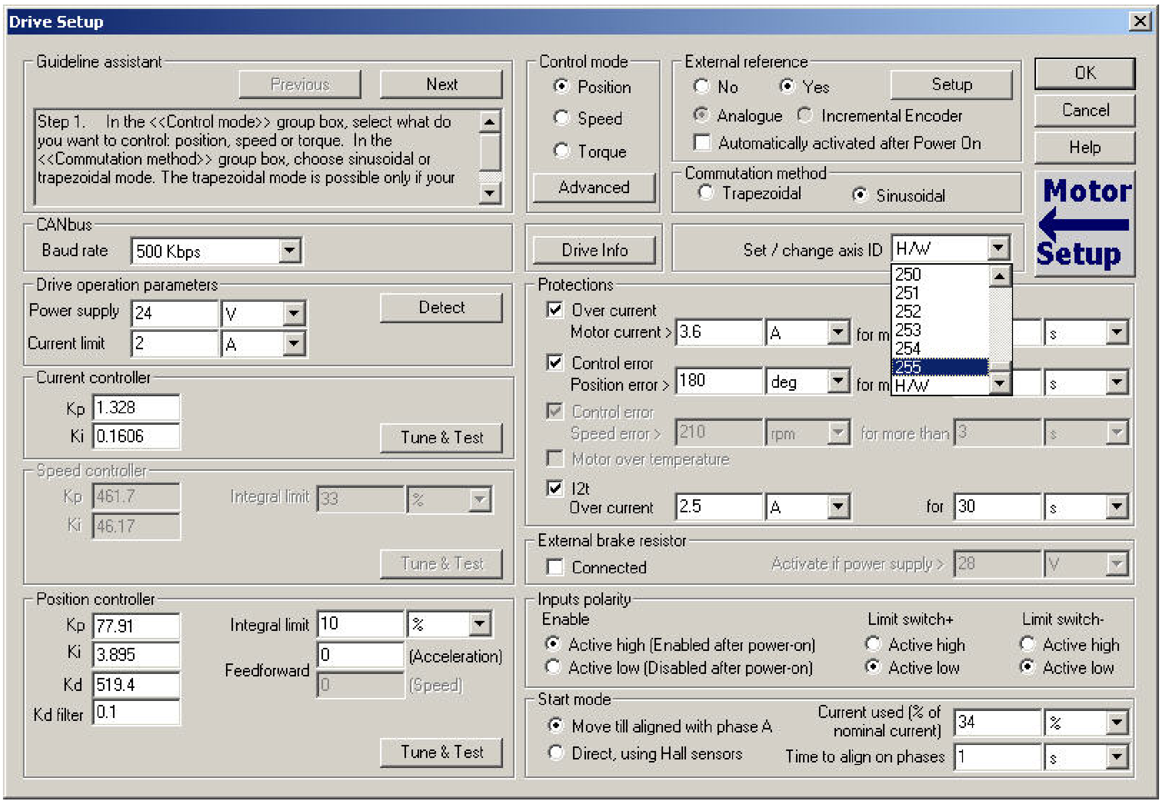
To summarize, you can define or change the setup data in the following ways:

- create a new setup data by going through the motor and intelligent motor dialogues
- use setup data previously saved in the PC
- upload setup data from a intelligent motor/motor EEPROM memory

4.2.4. Evaluate intelligent motor/motor behaviour (optional)

You can use the **Data Logger** or the **Control Panel** evaluation tools to quickly measure and analyze your application behavior. In case of errors like protections triggered, use the Drive Status control panel to find the cause.

4.3. Changing the intelligent motor Axis ID



The axis ID of an IM23x-MA intelligent motor can be set software – any value between 1 and 255, stored in the setup table.

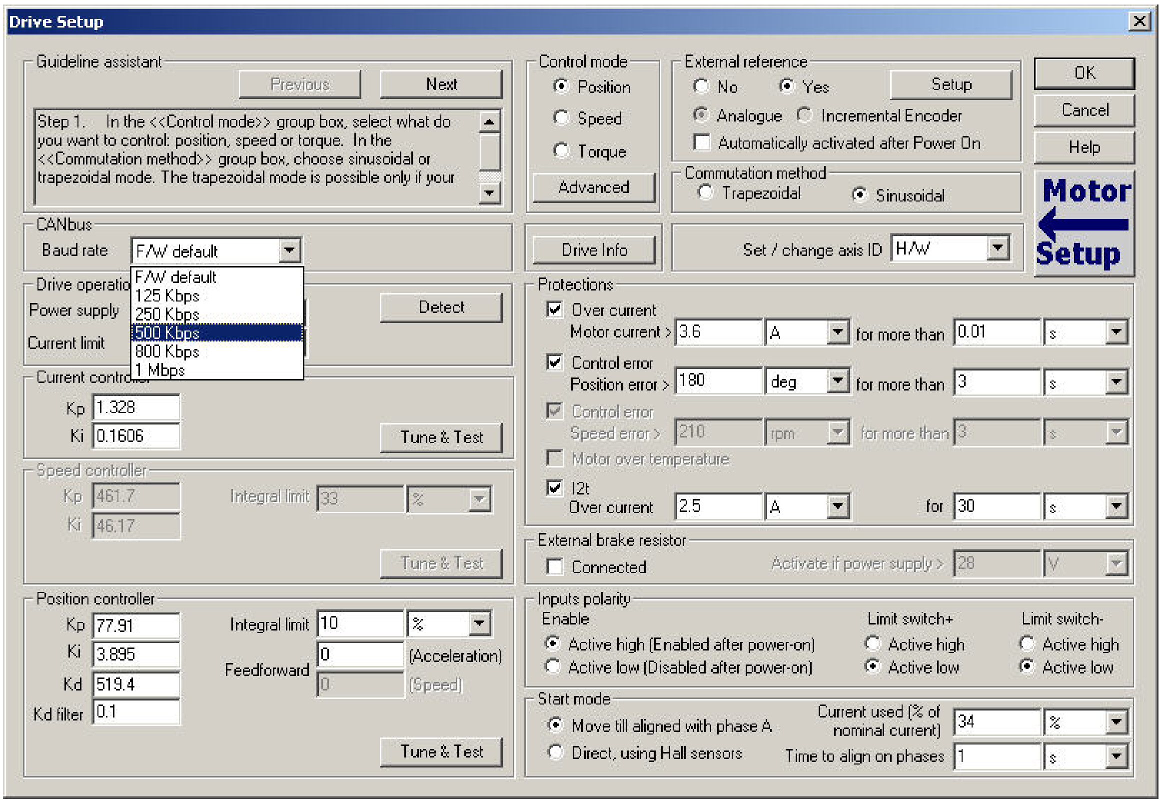
The axis ID is initialized at power on, using the following algorithm:

a) If a valid setup table exists, with the value read from it. This value can be an axis number 1 to 255

b) If the setup table is invalid, with the last value set with a valid setup table. This value can be an axis number 1 to 255

Remark: *If a intelligent motor axis ID was previously set by software and its value is not anymore known, you can find it by selecting in the Communication | Setup dialogue at "Axis ID of intelligent motor/motor connected to PC" the option* ***Autodetected****. Apply this solution only if this intelligent motor is connected directly with your PC via an RS-232 link. If this intelligent motor is part of a CANbus network and the PC is serially connected with another intelligent motor, use the menu command* ***Communication | Scan Network***

4.4. Setting CANbus rate



The IM23x-MA intelligent motors can work with the following rates on the CAN: 125kHz, 250kHz, 500KHz, 1MHz. In the Drive Setup dialogue you can choose the initial CAN rate after power on. This information is stored in the setup table. The CAN rate is initialized using the following algorithm:

If a valid setup table exists, with the CAN rate value read from it. This can be any of the supported rates or can indicate to use the firmware default (F/W default) value, which is 500kHz

If the setup table is invalid, with the last CAN rate value set with a valid setup table. This can be any of the supported rates or can indicate to use the firmware default (F/W default) value.

If there is no CAN rate value set by a valid setup table, with the firmware default value i.e. 500kHz

4.5. Creating an Image File with the Setup Data

Once you have validated your setup, you can create with the menu command **Setup | Create EEPROM Programmer File** a software file (with extension **.sw**) which contains all the setup data to write in the EEPROM of your intelligent motor.

A software file is a text file that can be read with any text editor. It contains blocks of data separated by an empty raw. Each block of data starts with the block start address, followed by data values to place in ascending order at consecutive addresses: first data – to write at start address, second data – to write at start address + 1, etc. All the data are hexadecimal 16- bit values (maximum 4 hexadecimal digits). Each raw contains a single data value. When less then 4 hexadecimal digits are shown, the value must be right justified. For example 92 represent 0x0092.

The **.sw** file can be programmed into a intelligent motor:

- from a CANopen master, using the communication objects for writing data into the intelligent motor EEPROM
- from a host PC or PLC, using the TML_LIB functions for writing data into the intelligent motor EEPROM
- using the EEPROM Programmer tool, which comes with EasySetUp but may also be installed separately. The EEPROM Programmer was specifically designed for repetitive fast and easy programming of **.sw** files into the Technosoft intelligent motors during production.

5. Step 3. Motion Programming

5.1. Using a CANopen Master (for IM23x-MA CANopen executions)

The IM23x-MA intelligent motor supports the CiA draft standard **DS-301 v4.02** CANopen Application Layer and Communication Profile. It also conforms with the CiA draft standard proposal **DSP-402 v2.0** CANopen Device Profile for Drives and Motion Control. For details see CANopen Programming manual (part no. P091.063.UM.xxxx)

5.1.1. DS-301 Communication Profile Overview

The IM23x-MA intelligent motor accepts the following basic services and types of communication objects of the CANopen communication profile DS 301 v4.02:

- **Service Data Object (SDO)**

Service Data Objects (SDOs) are used by CANopen master to access any object from the intelligent motor's Object Dictionary. Both expedited and segmented SDO transfers are supported (see DS301 v4.02 for details). SDO transfers are confirmed services. The SDOs are typically used for intelligent motor configuration after power-on, for PDOs mapping and for infrequent low priority communication between the CANopen master with the intelligent motors.

- **Process Data Object (PDO)**

Process Data Objects (PDO) are used for high priority, real-time data transfers between CANopen master and the intelligent motors. The PDOs are unconfirmed services which are performed with no protocol overhead. Transmit PDOs are used to send data from the intelligent motor, and receive PDOs are used to receive on to the intelligent motor. The IM23x-MA accepts 4 transmit PDOs and 4 receive PDOs. The contents of the PDOs can be set according with the application needs using the dynamic PDO-mapping. This operation can be done during the intelligent motor configuration phase using SDOs.

- **Synchronization Object (SYNC)**

The SYNC message provides the basic network clock, as the SYNC producer broadcasts the synchronization object periodically. The service is unconfirmed. The IM23x-MA supports both SYNC consumer and producer.

- **Time Stamp Object (TIME)**

The Time Stamp Object is not supported by the IM23x-MA device.

- **Emergency Object (EMCY)**

Emergency objects are triggered by the occurrence of a intelligent motor internal error situation. An emergency object is transmitted only once per 'error event'. As long as no new errors occur, the intelligent motor will not transmit further emergency objects.

- **Network Management Objects (NMT)**

The Network Management is node oriented and follows a master-slave structure. NMT objects are used for executing NMT services. Through NMT services the intelligent motor can be initialized, started, monitored, reset or stopped. The IM23x-MA is a NMT slave in a CANopen network.

- **Module Control Services** – through these unconfirmed services, the NMT master controls the state of the intelligent motor. The following services are implemented: Start Remote Node, Stop Remote Node, Enter Pre-Operational, Reset Node, Reset Communication
- **Error Control Services** – through these services the NMT master detects failures in a CAN-based network. Both error control services defined by DS301 v4.02 are supported by the IM23x-MA: Node Guarding (including Life Guarding) and Heartbeat
- **Bootup Service** - through this service, the intelligent motor indicates that it has been properly initialized and is ready to receive commands from a master

5.1.2. TechnoCAN Extension (for IM23x-MA CAN execution)

In order to take full advantage of the powerful Technosoft Motion Language (TML) built into the IM23x-MA, Technosoft has developed an extension to CANopen, called TechnoCAN through which TML commands can be exchanged with the intelligent motors. Thanks to TechnoCAN you can inspect or reprogram any of the Technosoft intelligent motors from a CANopen network using EastSetUp or EasyMotion Studio and an RS-232 link between your PC and anyone of the intelligent motors.

TechnoCAN uses only identifiers outside of the range used by the default by the CANopen predefined connection set (as defined by CiA DS301 v4.02). Thus, TechnoCAN protocol and CANopen protocol can co-exist and communicate simultaneously on the same physical CAN bus, without disturbing each other.

5.1.3. DSP-402 and Manufacturer Specific Device Profile Overview

The IM23x-MA supports the following CiA DSP402 v2.0 modes of operation:

- **Profile position mode**
- **Profile velocity mode**
- **Homing mode**
- **Interpolated position mode**

Additional to these modes, there are also several manufacturer specific modes defined:

- **External reference modes (position, speed or torque)**
- **Electronic gearing position mode**

5.1.4. Checking Setup Data Consistency

During the configuration phase, a CANopen master can quickly verify using the checksum objects and a reference **.sw** file (see 4.5 and 5.2.4 for details) whether the non-volatile EEPROM memory of an IM23x-MA intelligent motor contains the right information. If the checksum reported by the intelligent motor doesn't match with that computed from the **.sw** file, the CANopen master can download the entire **.sw** file into the intelligent motor EEPROM using the communication objects for writing data into the intelligent motor EEPROM.

5.2. Using the built-in Motion Controller and TML

One of the key advantages of the Technosoft intelligent motors is their capability to execute complex motions without requiring an external motion controller. This is possible because Technosoft intelligent motors offer in a single compact package both a state of art digital intelligent motor and a powerful motion controller.

5.2.1. Technosoft Motion Language Overview

Programming motion directly on a Technosoft intelligent motor requires creating and downloading a TML (Technosoft Motion Language) program into the intelligent motor memory. The TML allows you to:

- Set various motion modes (profiles, PVT, PT, electronic gearing or camming[13], etc.)
- Change the motion modes and/or the motion parameters
- Execute homing sequences[14]
- Control the program flow through:
  - Conditional jumps and calls of TML functions
  - TML interrupts generated on pre-defined or programmable conditions (protections triggered, transitions on limit switch or capture inputs, etc.)
  - Waits for programmed events to occur
- Handle digital I/O and analogue input signals
- Execute arithmetic and logic operations
- Perform data transfers between axes
- Control motion of an axis from another one via motion commands sent between axes
- Send commands to a group of axes (multicast). This includes the possibility to start simultaneously motion sequences on all the axes from the group
- Synchronize all the axes from a network

In order to program a motion using TML you need EasyMotion Studio software platform.

5.2.2. Installing EasyMotion Studio

EasyMotion Studio is an integrated development environment for the setup and motion programming of Technosoft intelligent intelligent motors. It comes with an ***Update via Internet tool*** through which you can check if your software version is up-to-date, and when necessary download and install the latest updates.

A **demo version of EasyMotion Studio** including the **fully functional version of EasySetUp** can be downloaded free of charge from Technosoft web page.

[13] Optional for the IM23x-MA CANopen executions
[14] The customization of the homing routines is available only for IM23x-MA CAN executions

EasyMotion Studio is delivered on a CD. Once you have started the installation package, follow its indications. After installation, use the update via internet tool to check for the latest updates. Alternately, you can first install the demo version and then purchase a license. By introducing the license serial number in the menu command **Help | Enter registration info…**, you can transform the demo version into a fully functional version.

5.2.3. Getting Started with EasyMotion Studio

Using EasyMotion Studio you can quickly do the setup and the motion programming of a Technosoft a intelligent motor according with your application needs. The intelligent motor can be:

- directly connected with your PC via a serial RS 232 link
- any intelligent motor from a CANbus network where the PC is serially linked with one of the other intelligent motors.

The output of the EasyMotion Studio is a set of setup data and a motion program, which can be downloaded to the intelligent motor/motor EEPROM or saved on your PC for later use.

EasyMotion Studio includes a set of evaluation tools like the Data Logger, the Control Panel and the Command Interpreter which help you to quickly develop, test, measure and analyze your motion application.

EasyMotion Studio works with **projects**. A project contains one or several **Applications**.

Each application describes a motion system for one axis. It has 2 components: the **Setup** data and the **Motion** program and an associated axis number: an integer value between 1 and 255. An application may be used either to describe:

1. One axis in a multiple-axis system
2. An alternate configuration (set of parameters) for the same axis.

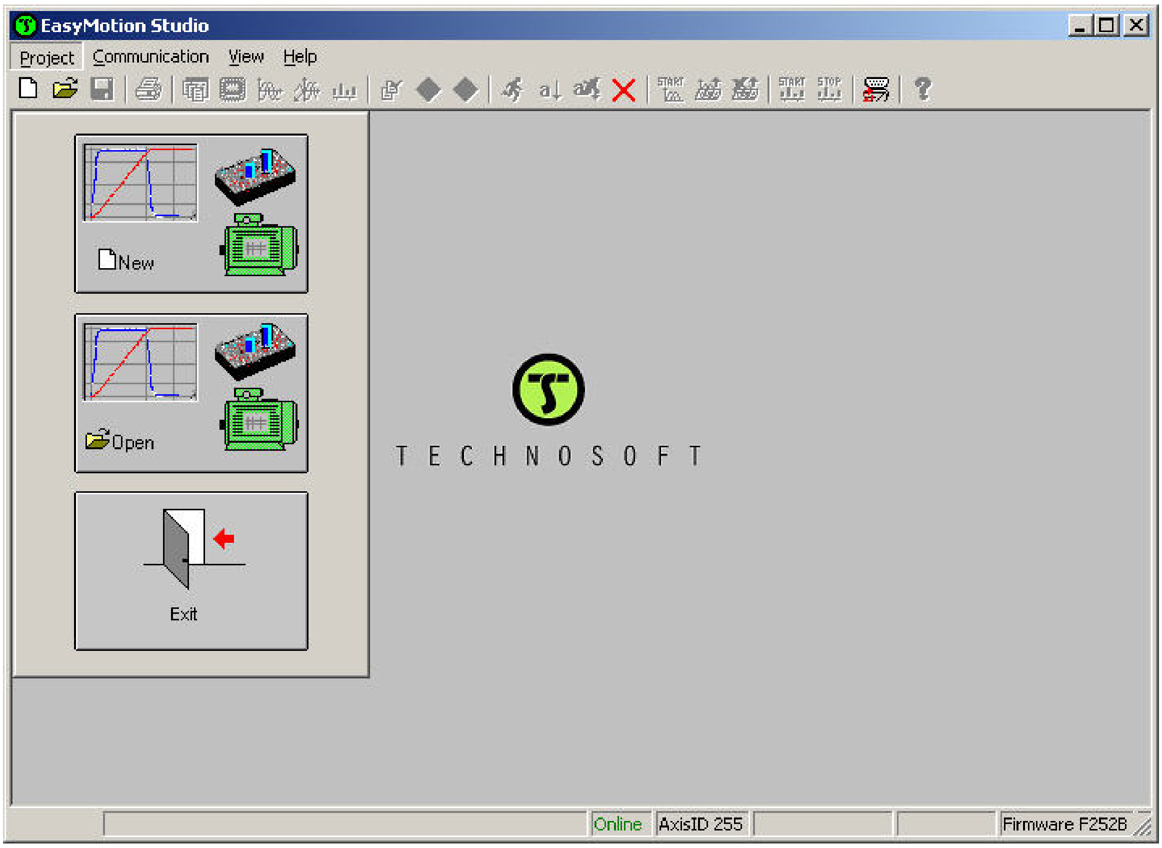
In the first case, each application has a different axis number corresponding to the axis ID of the intelligent motors/motors from the network. All data exchanges are done with the intelligent motor/motor having the same address as the selected application. In the second case, all the applications have the same axis number.

The setup component contains all the information needed to configure and parameterize a Technosoft intelligent motor. This information is preserved in the intelligent motor/motor EEPROM in the *setup table*. The setup table is copied at power-on into the RAM memory of the intelligent motor/motor and is used during runtime.

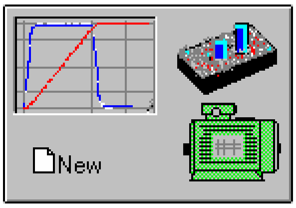
The motion component contains the motion sequences to do. These are described via a TML (Technosoft Motion Language) program, which is executed by the intelligent motors/motors built-in motion controller.

5.2.3.1 Create a new project

EasyMotion Studio starts with an empty window from where you can create a new project or open a previously created one.

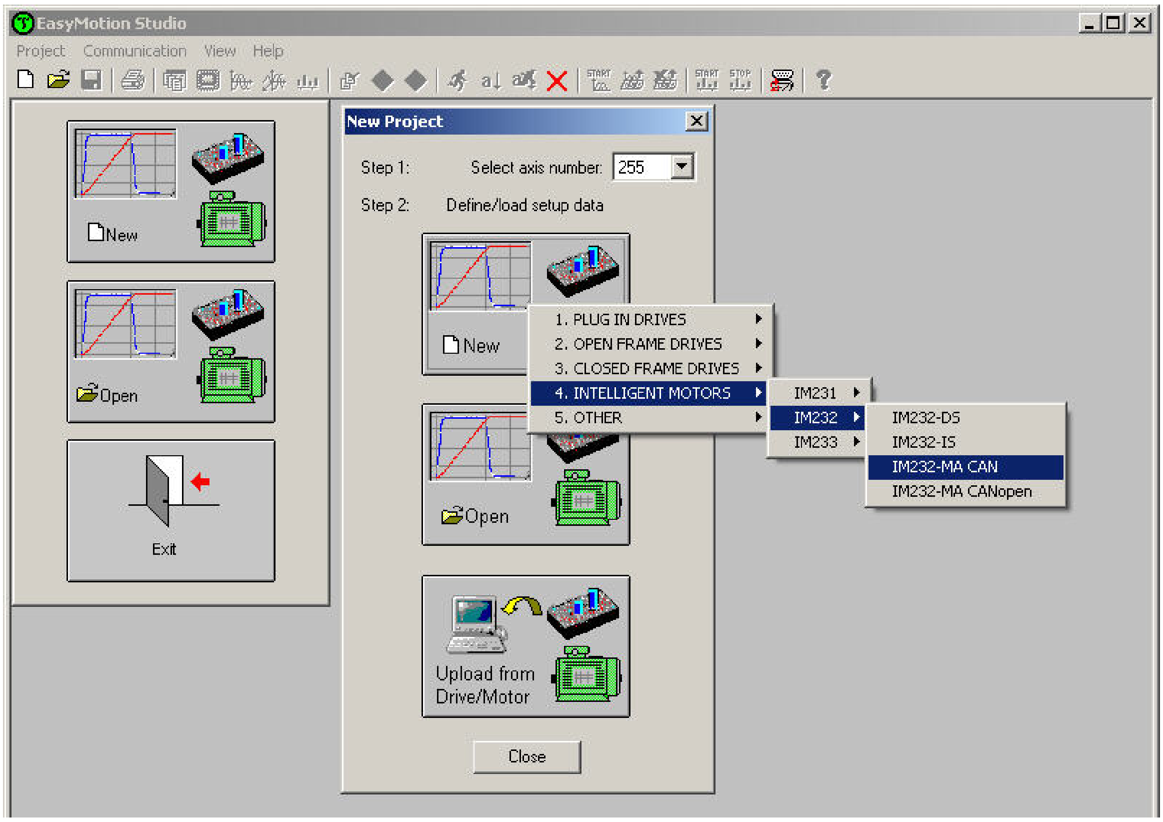


When you start a new project, EasyMotion Studio automatically creates a first application. Additional applications can be added later. You can duplicate an application or insert one defined in another project.

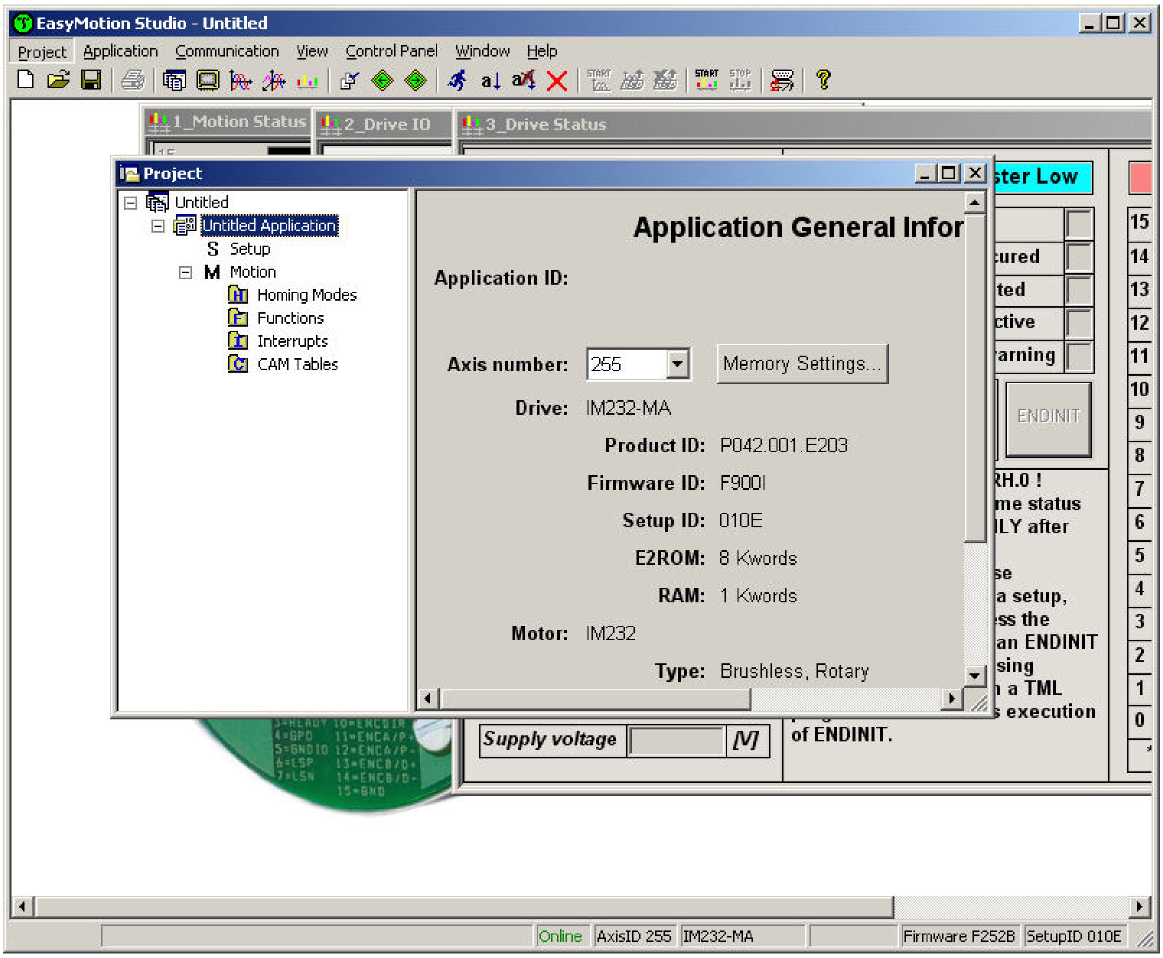
Press **New** button 

to open the "New Project" dialogue. Set the axis number for your first application equal with your intelligent motor/motor axis ID. The initial value proposed is 255 which is the default axis ID of the intelligent motors.

Press **New** button and select your intelligent motor type (for example the IM232-MA CAN product).



Click on your selection. EasyMotion Studio opens the Project window where on the left side you can see the structure of a project. At beginning both the new project and its first application are named "Untitled". The application has 2 components: **S** Setup and **M** Motion (program).



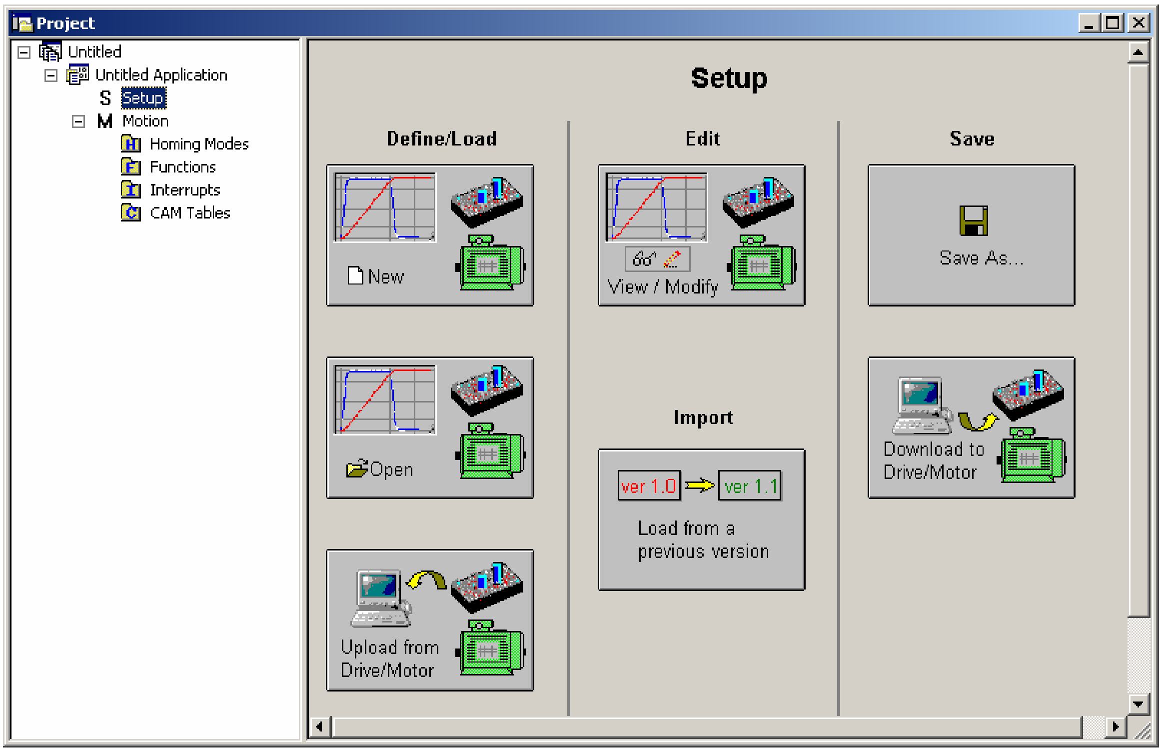
5.2.3.2 Step 2 Establish communication

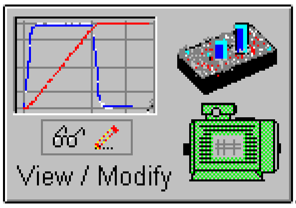
If you have a intelligent motor/motor connected with your PC, now its time to check the communication. Use menu command **Communication | Setup** to check/change your PC communication settings. Press the **Help** button of the dialogue opened. Here you can find detailed information about how to setup your intelligent motor/motor and the connections. Power on the intelligent motor, then close the Communication | Setup dialogue with OK. If the communication is established, EasyMotion Studio displays in the status bar (the bottom line) the text "**Online**" plus the axis ID of your intelligent motor/motor and its firmware version. Otherwise the text displayed is "**Offline**" and a communication error message tells you the error type. In this case, return to the Communication | Setup dialogue, press the Help button and check troubleshoots.

Remark: *When first started, EasyMotion Studio tries to communicate via RS-232 and COM1 with a intelligent motor having axis ID=255 (default communication settings). If your intelligent motor is powered with all the DIP switches OFF and it is connected to your PC port COM1 via an RS-232 cable, the communication shall establish automatically.*

5.2.3.3 Setup intelligent motor/motor

In the project window left side, select “**S** Setup”, to access the setup data for your application.



Press **View/Modify** button 

. This opens 2 setup dialogues: for **Motor Setup** and for **Drive Setup** (same like on EasySetUp) through which you can configure and parameterize a Technosoft intelligent motor. In the **Motor setup** dialogue you can introduce the data of your motor and the associated sensors. Data introduction is accompanied by a series of tests having as goal to check the connections to the intelligent motor and/or to determine or validate a part of the motor and sensors parameters. In the **Drive setup** dialogue you can configure and parameterize the intelligent motor for your application. In each dialogue you will find a **Guideline Assistant**, which will guide you through the whole process of introducing and/or checking your data.

Press the **Download to Drive/Motor** button 

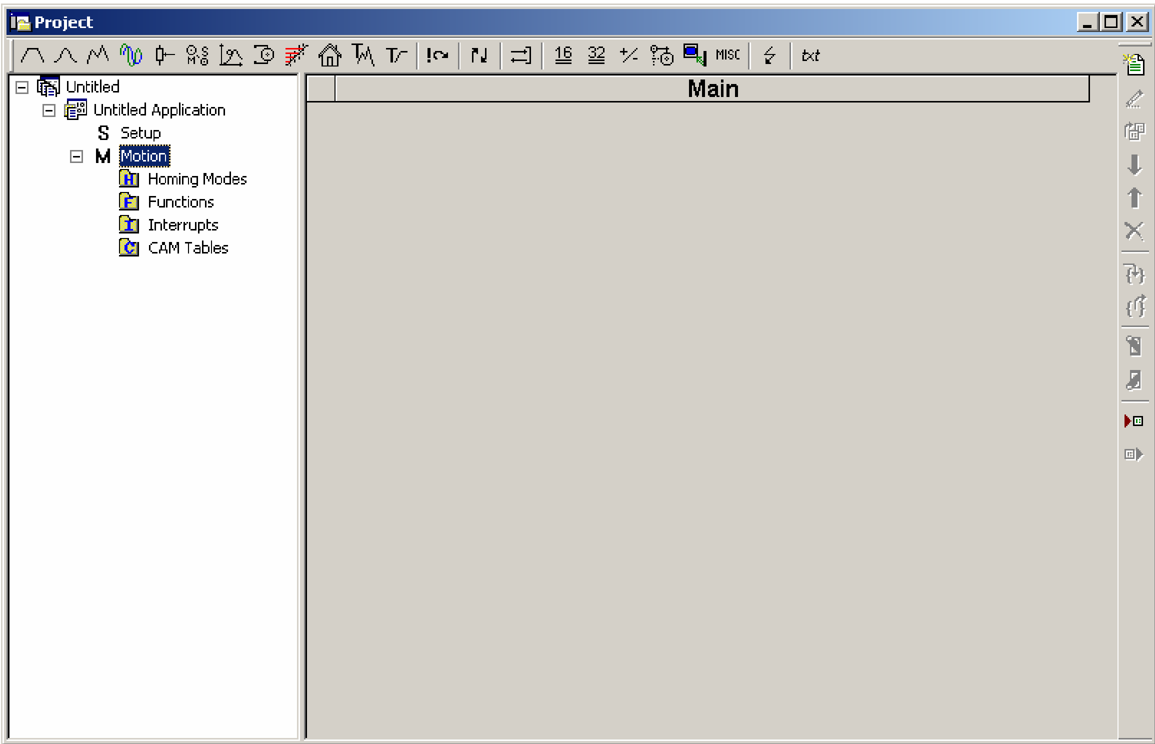
to download your setup data in the intelligent motor/motor EEPROM memory in the *setup table*. From now on, at each power-on, the setup data is copied into the intelligent motor/motor RAM memory which is used during runtime. It is also possible to save the setup data on your PC and use it in other applications. Note that you can upload the complete setup data from a intelligent motor/motor.

To summarize, you can define or change the setup data of an application in the following ways:

- create a new setup data by going through the motor and intelligent motor dialogues
- use setup data previously saved in the PC
- upload setup data from a intelligent motor/motor EEPROM memory

5.2.3.4 Program motion

In the project window left side, select "**M** Motion", for motion programming. This automatically activates the **Motion Wizard**.



The Motion Wizard offers you the possibility to program all the motion sequences using high level graphical dialogues which automatically generate the corresponding TML instructions. Therefore with Motion Wizard you can develop motion programs using almost all the TML instructions without needing to learn them. A TML program includes a main section, followed by the subroutines used: functions, interrupt service routines and homing procedures[15]. The TML program may also include cam tables used for electronic camming applications[16].

When activated, Motion Wizard adds a set of toolbar buttons in the project window just below the title. Each button opens a programming dialogue. When a programming dialogue is closed, the associated TML instructions are automatically generated. Note that, the TML instructions generated are not a simple text included in a file, but a motion object. Therefore with Motion Wizard you define your motion program as a collection of motion objects.

The major advantage of encapsulating programming instructions in motion objects is that you can very easily manipulate them. For example, you can:

- Save and reuse a complete motion program or parts of it in other applications
- Add, delete, move, copy, insert, enable or disable one or more motion objects
- Group several motion objects and work with bigger objects that perform more complex functions

As a starting point, push for example the leftmost Motion Wizard button – Trapezoidal profiles, and set a position or speed profile. Then press the **Run** button. At this point the following operations are done automatically:

- A TML program is created by inserting your motion objects into a predefined template
- The TML program is compiled and downloaded to the intelligent motor/motor
- The TML program execution is started

For learning how to send TML commands from your host/master, using one of the communication channels and protocols supported by the intelligent motors use menu command **Application | Binary Code Viewer…** Using this tool, you can get the exact contents of the messages to send and of those expected to be received as answers.

5.2.3.5 Evaluate motion application performances

EasyMotion Studio includes a set of evaluation tools like the **Data Logger**, the **Control Panel** and the **Command Interpreter** which help you to quickly measure and analyze your motion application.

5.2.4. Creating an Image File with the Setup Data and the TML Program

Once you have validated your application, you can create with the menu command **Application | Create EEPROM Programmer File** a software file (with extension **.sw**) which contains all the data to write in the EEPROM of your intelligent motor. This includes both the setup data and the motion program. For details regarding the **.sw** file format and how it can be programmed into a intelligent motor, see paragraph 4.5

[15] The customization of the interrupt service routines and homing routines is available only for IM23x-MA CAN executions
[16] Optional for the IM23x-MA CANopem executions

5.3. Combining CANopen /or other host with TML

Due to its embedded motion controller, a IM23x-MA offers many programming solutions that may simplify a lot the task of a CANopen master. This paragraph overviews a set of advanced programming features which arise when combining TML programming at intelligent motor level with CANopen master control. A detailed description of these advanced programming features is included in the ***CANopen Programming*** **(part no. P091.063.UM.xxxx)** manual. All features presented below require usage of EasyMotion Studio as TML programming tool

Remark: *If you don't use the advanced features presented below you don't need EasyMotion Studio. In this case the IM23x-MA is treated like a standard CANopen intelligent motor, whose setup is done using EasySetUp.*

5.3.1. Using TML Functions to Split Motion between Master and Drives

With Technosoft intelligent intelligent motors you can really distribute the intelligence between a CANopen master and the intelligent motors in complex multi-axis applications. Instead of trying to command each step of an axis movement, you can program the intelligent motors using TML to execute complex tasks and inform the master when these are done. Thus for each axis, the master task may be reduced at: calling TML functions (with possibility to abort their execution) stored in the intelligent motors EEPROM and waiting for a message, which confirms the finalization of the TML functions execution.

5.3.2. Executing TML programs

The distributed control concept can go one step further. You may prepare and download into a intelligent motor a complete TML program including functions, homing procedures[17], etc. The TML program execution can be started by simply writing a value in a dedicated object,

5.3.3. Loading Automatically Cam Tables Defined in EasyMotion Studio

The IM23x-MA offers others motion modes like[18]: electronic gearing, electronic camming, external modes with analogue or digital reference etc. When electronic camming is used, the cam tables can be loaded in the following ways:

a) The master downloads the cam points into the intelligent motor active RAM memory after each power on;

b) The cam points are stored in the intelligent motor EEPROM and the master commands their copy into the active RAM memory

c) The cam points are stored in the intelligent motor EEPROM and during the intelligent motor initialization (transition to Ready to Switch ON status) are automatically copied from EEPROM to the active RAM

For the last 2 options the cam table(s) are defined in EasyMotion Studio and are included in the information stored in the EEPROM together with the setup data and the TML programs/functions.

---

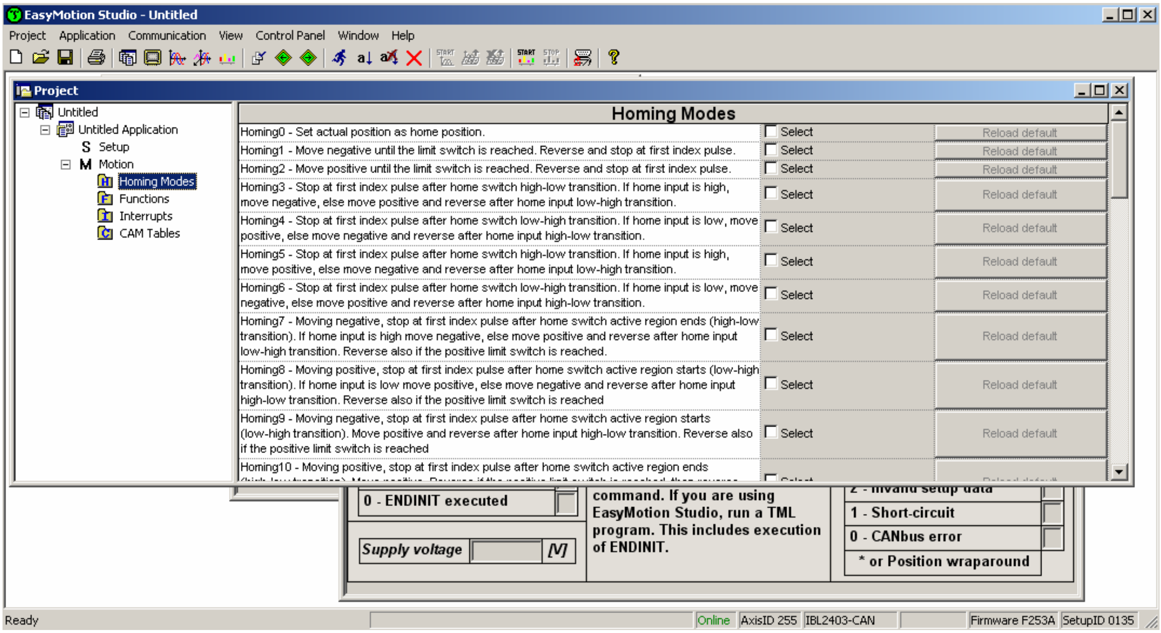
[17] The customization of the interrupt service routines and homing routines is available only for IM23x-MA CAN executions

[18] Optional for IM23x-MA CANopen executions

Remark: The cam tables are included in the **.sw** file generated with EasyMotion Studio. Therefore, the intelligent motors can check the cam presence in the intelligent motor EEPROM using the same procedure as for testing of the setup data.

5.3.4. Customizing the Homing Procedures (for IM23x-MA CAN executions)

The IM23x-MA supports all homing modes defined in DSP-402 device profile. If needed, any of these homing modes can be customized. In order to do this you need to select the Homing Modes from your EasyMotion Studio application and in the right side to set as “User defined” one of the Homing procedures. Following this operation the selected procedure will occur under Homing Modes in a subtree, with the name *HomeX* where X is the number of the selected homing.



If you click on the *HomeX* procedure, on the right side you’ll see the TML function implementing it. The homing routine can be customized according to your application needs. It’s calling name and method remain unchanged.

5.3.5. Customizing the Drive Reaction to Fault Conditions (for IM23x-MA CAN executions)

Similarly to the homing modes, the default service routines for the TML interrupts can be customized according to your application needs. However, as most of these routines handle the intelligent motor reaction to fault conditions, it is mandatory to keep the existent functionality while adding your application needs, in order to preserve the correct protection level of the intelligent motor. The procedure for modifying the TML interrupts is similar with that for the homing modes.

5.4. Using Motion Libraries for PC-based Systems

A **TML Library for PC** is a collection of high-level functions allowing you to control from a PC a network of Technosoft intelligent intelligent motors. It is an ideal tool for quick implementation on PCs of motion control applications with Technosoft products.

With the TML Motion Library functions you can: communicate with a intelligent motor / motor via any of its supported channels (RS-232, CAN-bus, etc.), send motion commands, get automatically or on request information about intelligent motor / motor status, check and modify its setup parameters, read inputs and set outputs, etc.

The TML Motion Library can work under a **Windows** or **Linux** operating system. Implemented as a .dll/.so, it can be included in an application developed in **C/C++/C#, Visual Basic, Delphi Pascal** or **Labview**.

Using a TML Motion Library for PC, you can focus on the main aspects of your application, while the motion programming part can be reduced to calling the appropriate functions and getting the confirmation when the task was done.

All Technosoft's TML Motion Libraries for PCs are provided with EasySetUp.

5.5. Using Motion Libraries for PLC-based Systems

A **TML Motion Library for PLC** is a collection of high-level functions and function blocks allowing you to control from a PLC the Technosoft intelligent intelligent motors. The motion control function blocks are developed in accordance with the **PLC IEC61131-3 standard** and represent an ideal tool for quick implementation on PLCs of motion control applications with Technosoft products.

With the TML Motion Library functions you can: communicate with a intelligent motor/motor via any of its supported channels, send motion commands, get automatically or on request information about intelligent motor/motor status, check and modify its setup parameters, read inputs and set outputs, etc. Depending on the PLC type, the communication is done either directly with the CPU unit, or via a CANbus or RS-232 communication module.

Using a TML Motion Library for PLC, you can focus on the main aspects of your PLC application, while the motion programming part can be reduced to calling the appropriate functions and monitoring the confirmations that the task was done.

As Technosoft intelligent motors can execute a greater number of motion modes than those introduced by the PLC IEC61131-3 standard, the TML Motion Libraries for PLCs include additional function blocks to support them. All these blocks have been designed using the guidelines described in the PLC IEC61131-3 standards, so they can be used on any developmemnt platform that is **IEC 61136 compliant.**

All Technosoft's TML Motion Libraries for PLC are provided with EasySetUp.

6. Scaling factors

Technosoft intelligent motors work with parameters and variables represented in the intelligent motor internal units (IU). These correspond to various signal types: position, speed, current, voltage, etc. Each type of signal has its own internal representation in IU and a specific scaling factor. This chapter presents the intelligent motor internal units and their relation with the international standard units (SI).

In order to easily identify them, each internal unit has been named after its associated signal. For example the **position units** are the internal units for position, the **speed units** are the internal units for speed, etc.

6.1. Position units

The internal position units are encoder counts. The correspondence with the load **position in SI units**[19] is:

$$\text{Load\_Position[SI]} = \frac{2 \times \pi}{4 \times \text{No\_encoder\_lines} \times \text{Tr}} \times \text{Motor\_Position[IU]}$$

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

Encoder_accuracy – is the linear encoder accuracy i.e. distance in [m] between 2 pulses

Tr – transmission ratio between the motor displacement in SI units and load displacement in SI units

6.2. Speed units

The internal speed units are encoder counts / (slow loop sampling period). The correspondence with the load **speed in SI units** is:

$$\text{Load\_Speed[SI]} = \frac{2 \times \pi}{4 \times \text{No\_encoder\_lines} \times \text{Tr} \times \text{T}} \times \text{Motor\_Speed[IU]}$$

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

Encoder_accuracy – is the linear encoder accuracy i.e. distance in [m] between 2 pulses

Tr – transmission ratio between the motor displacement in SI units and load displacement in SI units

[19] SI units for position are: [rad] for a rotary movement, [m] for a linear movement

T – is the slow loop sampling period expressed in [s]. You can read this value in the “Advanced” dialogue, which can be opened from the “Drive Setup”

6.3. Acceleration units

The internal acceleration units are encoder counts / (slow loop sampling period)[2]. The correspondence with the load **acceleration in SI units** is:

$$\text{Load\_Acceleration[SI]} = \frac{2 \times \pi}{4 \times \text{No\_encoder\_lines} \times \text{Tr} \times \text{T}^2} \times \text{Motor\_Acceleration[IU]}$$

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

Encoder_accuracy – is the linear encoder accuracy i.e. distance in [m] between 2 pulses

Tr – transmission ratio between the motor displacement in SI units and load displacement in SI units

T – is the slow loop sampling period expressed in [s]. You can read this value in the “Advanced” dialogue, which can be opened from the “Drive Setup”

6.4. Jerk units

The internal jerk units are encoder counts / (slow loop sampling period)[3]. The correspondence with the load **jerk in SI units**[20] is:

$$\text{Load\_Jerk[SI]} = \frac{2 \times \pi}{4 \times \text{No\_encoder\_lines} \times \text{Tr} \times \text{T}^3} \times \text{Motor\_Jerk[IU]}$$

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

Encoder_accuracy – is the linear encoder accuracy i.e. distance in [m] between 2 pulses

Tr – transmission ratio between the motor displacement in SI units and load displacement in SI units

T – is the slow loop sampling period expressed in [s]. You can read this value in the “Advanced” dialogue, which can be opened from the “Drive Setup”

6.5. Current units

The internal current units refer to the motor phase currents. The correspondence with the motor currents in [A] is:

$$\text{Current[A]} = \frac{2 \times \text{Ipeak}}{65520} \times \text{Current[IU]}$$

[20] SI units for jerk are [rad/s^3] for a rotary movement, [m/s^3] for a linear movement

where Ipeak – is the intelligent motor peak current expressed in [A]. You can read this value in the “Drive Info” dialogue, which can be opened from the “Drive Setup”.

6.6. Voltage command units

The internal voltage command units refer to the voltages applied on the motor. The significance of the voltage commands as well as the scaling factors, depend on the motor type and control method used.

In case of **brushless motors** intelligent motorn in **sinusoidal** mode, a field oriented vector control is performed. The voltage command is the amplitude of the sinusoidal phase voltages. In this case, the correspondence with the motor phase voltages in SI units i.e. [V] is:

$$\text{Voltage command[V]} = \frac{1.1 \times \text{Vdc}}{65534} \times \text{Voltage command[IU]}$$

where Vdc – is the intelligent motor power supply voltage expressed in [V].

In case of **brushless** motors intelligent motorn in **trapezoidal** mode, the voltage command is the voltage to apply between 2 of the motor phases, according with Hall signals values. In this case, the correspondence with the voltage applied in SI units i.e. [V] is:

$$\text{Voltage command[V]} = \frac{\text{Vdc}}{32767} \times \text{Voltage command[IU]}$$

This correspondence is akso available for **DC brushed** motors which have the voltage command internal units as the brushless motors intelligent motorn in trapezoidal mode.

6.7. Voltage measurement units

The internal voltage measurement units refer to the intelligent motor V_{MOT} supply voltage. The correspondence with the supply voltage in [V] is:

$$\text{Voltage\_measured[V]} = \frac{\text{VdcMaxMeasurable}}{65520} \times \text{Voltage\_measured[IU]}$$

where VdcMaxMeasurable – is the maximum measurable DC voltage expressed in [V]. You can read this value in the “Drive Info” dialogue, which can be opened from the “Drive Setup”.

Remark: *the voltage measurement units occur in the scaling of the over voltage and under voltage protections and the supply voltage measurement*

6.8. Time units

The internal time units are expressed in slow loop sampling periods. The correspondence with the time in [s] is:

$$\text{Time[s]} = \text{T} \times \text{Time[IU]}$$

where T – is the slow loop sampling period expressed in [s]. You can read this value in the “Advanced” dialogue, which can be opened from the “Drive Setup”. For example, if T = 1ms, one second = 1000 IU.

6.9. Drive temperature units

The intelligent motor includes a temperature sensor. The correspondence with the temperature in [°C] is:

Drive temperature [°C] = $\frac{3.3[V] \times DriveTemperature[IU]}{65520 \times Sensor\_gain[V/°C]} - \frac{Sensor\_output\_0°C[V]}{Sensor\_gain[V/°C]}$

where:

Sensor_gain – is the temperature sensor gain

Sensor_output_0°C – is the temperature sensor output at 0°C. You can read these values in the "Drive Info" dialogue, which can be opened from the "Drive Setup"

6.10. Master position units

When the master position is sent via a communication channel or via pulse & direction signals, the master position units depend on the type of position sensor present on the master axis.

When the master position is an encoder the correspondence with the international standard (SI) units is:

$$Master\_position[rad] = \frac{2 \times \pi}{4 \times No\_encoder\_lines} \times Master\_position[IU]$$

where:

No_encoder_lines – is the master number of encoder lines per revolution

6.11. Master speed units

The master speed is computed in internal units (IU) as master position units / slow loop sampling period i.e. the master position variation over one position/speed loop sampling period.

When the master position is an encoder, the correspondence with the international standard (SI) units is:

$$Master\_speed[rad/s] = \frac{2 \times \pi}{4 \times No\_encoder\_lines \times T} \times Master\_speed[IU]$$

where:

No_encoder_lines – is the master number of encoder lines per revolution

T – is the slave slow loop sampling period, expressed in [s]. You can read this value in the "Advanced" dialogue, which can be opened from the "Drive Setup".

6.12. Motor position units

The internal motor position units are encoder counts. The correspondence with the motor **position in SI units**[21] is:

$$Motor\_Position[SI] = \frac{2 \times \pi}{4 \times No\_encoder\_lines} \times Motor\_Position[IU]$$

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

Encoder_accuracy – is the linear encoder accuracy i.e. distance in [m] between 2 pulses

6.13. Motor speed units

The internal motor speed units are encoder counts / (slow loop sampling period). The correspondence with the motor **speed in SI units** is:

$$Motor\_Speed[SI] = \frac{2 \times \pi}{4 \times No\_encoder\_lines \times T} \times Motor\_Speed[IU]$$

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

Encoder_accuracy – is the linear encoder accuracy i.e. distance in [m] between 2 pulses

T – is the slow loop sampling period expressed in [s]. You can read this value in the “Advanced” dialogue, which can be opened from the “Drive Setup”

[21]SI units for motor position are: [rad] for a rotary motor, [m] for a linear motor

7. Memory Map

The intelligent motor has 2 types of memory: a 1.5K×16 SRAM (internal) memory and an 8K×16 serial E^2ROM (external) memory.

The SRAM memory is mapped both in the program space (from 8270h to 87FFh) and in the data space (from 0A70h to 0FFFh). The data memory can be used for real-time data acquisition and to temporarily save variables during a TML program execution. The program space can be used to download and execute TML programs. It is the user's choice to decide how to split the 1.5-K SRAM into data and program memory.

The E^2ROM is seen as 8K×16 program memory mapped in the address range 4000h to 5FBEh. It offers the possibility to keep TML programs in a Non-volatile memory. Read and write accesses to the E^2ROM locations, as well as TML programs downloading and execution, are done from the user's point of view similarly to those in the SRAM program memory. The E^2ROM SPI serial access is completely transparent to the user.

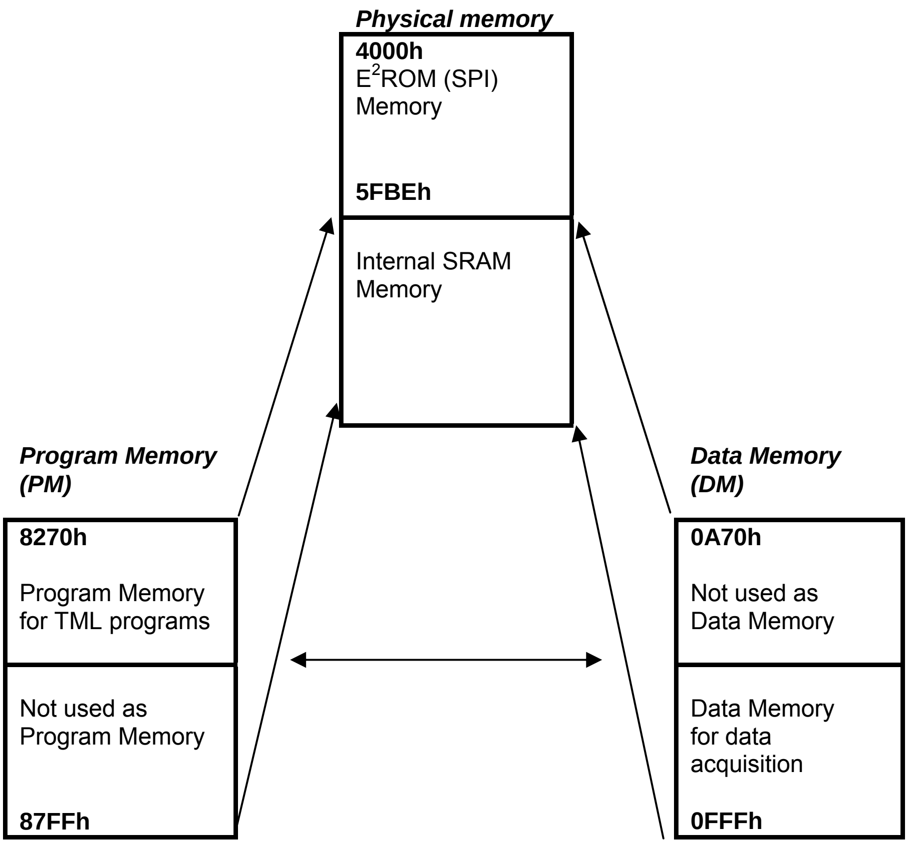


Figure 7.1. *IM23x-MA Memory Map*

This page is empty

T E C H N O S O F T