iPOS4808 VX

Intelligent Servo Drive for Step, DC, Brushless DC and AC Motors

TECHNOSOFT

Intelligent Servo Drive

Technical Reference

TECHNOSOFT

iPOS4808 VX

Technical Reference

P091.027.iPOS4808.VX.UM.0818

Technosoft S.A.
Avenue des Alpes 20
CH-2000 NEUCHATEL
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

The information in this document is subject to change without notice.

About This Manual

This book is a technical reference manual for the **iPOS4808 VX** family of intelligent servo drives, including the following product:

iPOS4808 VX-CAN (p/n P027.014.E001)

In order to operate the **iPOS4808** drives, you need to pass through 3 steps:

- ❑ **Step 1 Hardware installation**
- ❑ **Step 2 Drive setup** using Technosoft **EasySetUp** software for drive commissioning
- ❑ **Step 3 Motion programming** using one of the options:

  - ❑ A **CANopen master**[1]
  - ❑ The drive's **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 drives in TML

This manual covers **Step 1** in detail. It describes the **iPOS4808** 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 drive internal units. For detailed information regarding the next steps, refer to the related documentation.

Notational Conventions

This document uses the following conventions:

- **iPOS4808**– all products described in this manual
- **IU units** – Internal units of the drive
- **SI units** – International standard units (meter for length, seconds for time, etc.)
- **TML** – Technosoft Motion Language
- **TMLCAN** – Technosoft protocol for exchanging TML commands via CAN-bus

Related Documentation

Help of the EasySetUp software – describes how to use **EasySetUp** to quickly setup any Technosoft drive for your application using only 2 dialogues. The output of EasySetUp is

---

[1] when iPOS480x is set in CANopen mode

a set of setup data that can be downloaded into the drive EEPROM or saved on a PC file. At power-on, the drive is initialized with the setup data read from its EEPROM. With EasySetUp it is also possible to retrieve the complete setup information from a drive previously programmed. **EasySetUp can be downloaded free of charge from Technosoft web page**

iPOS CANopen Programming **(part no. P091.063.iPOS.UM.xxxx)** – explains how to program the iPOS family of intelligent drives using **CANopen** protocol and describes the associated object dictionary for **CiA 301 v.4.2** application layer and communication profile and **CiA DSP 402 v3.0** device profile for drives and motion control now included in IEC 61800-7-1 Annex A, IEC 61800-7-201 and IEC 61800-7-301 standards

Motion Programming using EasyMotion Studio (part no. P091.034.ESM.UM.xxxx) – describes how to use the EasyMotion Studio to create motion programs using in Technosoft Motion Language (TML). EasyMotion Studio platform includes **EasySetUp** for the drive/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 drives – 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 drives 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 drives 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 drives 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 in a PLC **Omron series CJ1** a motion application for the Technosoft intelligent drives using TML_LIB_CJ1 motion control library for PLCs. The TML_LIB_CJ1 library is **IEC61131-3 compatible**.

TML_LIB_X20 **(part no. P091.040.X20.UM.xxxx)** – explains how to program in a PLC **B&R series X20** a motion application for the Technosoft intelligent drives using TML_LIB_X20 motion control library for PLCs. The TML_LIB_X20 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

IO-iPOS4808 (part no. P091.027.IO-iPOS4808.UM.xxxx) – describes the IO iPOS4808 I/O extension board included in the **iPOS4808**Starter Kits.

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 Avenue des Alpes 20 CH-2000, Neuchatel Switzerland

Contents

1. Safety information

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

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

Only qualified personnel may install, set up, operate and maintain the drive. A "qualified person" has the knowledge and authorization to perform tasks such as transporting, assembling, install-ing, commissioning and operating drives.

The following safety symbols are used in this manual:

	WARNING!	*SIGNALS A DANGER THAT MIGHT CAUSE BODILY INJURY TO THE OPERATOR. 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 DIS-CHARGES (ESD), WHICH REQUIRE HANDLING IN AN ESD PROTECTED ENVIRONMENT*

1.1. Warnings

	WARNING!	*TO AVOID ELECTRIC ARCING AND HAZARDS, NEVER PLUG / UNPLUG THE iPOS4808 FROM IT'S SOCKET WHILE THE POWER SUPPLIES ARE ON !*
	WARNING!	**THE DRIVE MAY HAVE HOT SURFACES DURING OPERATION.**
	WARNING!	**DURING DRIVE 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 iPOS4808 IS AN ELECTROSTATICALLY SENSITIVE DEVICE, WHICH WILL 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 drive on a grounded conductive surface and also ground yourself.

1.3. Quality system, conformance and certifications

quality austria Succeed with Quality — IQNet — THE INTERNATIONAL CERTIFICATION NETWORK	**IQNet** and **Quality Austria** certification about the implementation and maintenance of the Quality Management System which fulfills the requirements of Standard **ISO 9001:2015**.
	Quality Austria Certificate about the application and further development of an effective **Quality Management System** complying with the requirements of Standard **ISO 9001:2015**
REACH COMPLIANT · COMPLIANT	**REACH Compliance - TECHNOSOFT hereby confirms that this product comply with the legal obligations regarding Article 33 of the European REACH Regulation 1907/2006 (Registration, Evaluation, Authorization and Restriction of Chemicals), which came into force on 01.06.2007.**
ROHS COMPLIANT · COMPLIANT	**RoHS Compliance - Technosoft SA here with declares that this product is manufactured in compliance with the RoHS directive 2002/95/EC on the restriction of the use of certain hazardous substances in electrical and electronic equipment (RoHS)**
CE	**Technosoft SA hereby declares that this product conforms to the following European applicable directives:** **2014/30/EU Electromagnetic Compatibility (EMC) Directive** **2014/35/EU Low Voltage Directive (LVD)** **93/68/EEC CE Marking Directive**
CONFLICT MINERALS	**Conflict minerals statement - Technosoft declares that the company does not purchase 3T&G (tin, tantalum, tungsten & gold) directly from mines or smelters...** **We have no indication that Technosoft products contain minerals from conflict mines or smelters in and around the DRC.**

For other certifications visit: http://technosoftmotion.com/en/quality-system

2. Product Overview

2.1. Introduction

The **iPOS4808** is a family of fully digital intelligent servo drives, based on the latest DSP technology and they offer unprecedented drive performance combined with an embedded motion controller.

Suitable for control of brushless DC, brushless AC (vector control), DC brushed motors and step motors, the iPOS4808 drives accept as position feedback incremental encoders (quadrature or sine/cosine) and linear Halls signals.

All drives perform position, speed or torque control and work in single, multi-axis or stand-alone configurations. Thanks to the embedded motion controller, the iPOS4808 drives combine controller, drive 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 drive 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
- ❑ 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

By implementing motion sequences directly at drive level you can really distribute the intelligence between the master and the drives 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 drives 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 drive EEPROM and waiting for a message, which confirms the TML functions execution completion.

All iPOS4808 drives are equipped with a serial RS232 and a CAN 2.0B interface and can be set (hardware, via an AxisID input level) to operate in 2 modes:

- ❑ **CANopen**
- ❑ **TMLCAN**

When **CANopen** mode is selected, the iPOS4808 conforms to **CiA 301 v4.2** application layer and **CiA DSP 402 v3.0** device profile for drives and motion control, now included in IEC 61800-7-1 Annex A, IEC 61800-7-201 and IEC 61800-7-301 standards. In this mode, the iPOS4808 may be controlled via a CANopen master. As a bonus, iPOS4808 offers a CANopen master the option to call motion sequences, written in TML and stored in the drive EEPROM, using manufacturer specific objects (see for details par. 5.3).

---

[1] Available if the master axis sends its position via a communication channel

When **TMLCAN** mode is selected, the iPOS4808 behaves as standard Technosoft intelligent drive and conforms to Technosoft protocol for exchanging TML commands via CAN-bus. When TMLCAN protocol is used, it is not mandatory to have a master. Any iPOS4808 can be set to operate standalone, and may play the role of a master to coordinate both the network communication/synchronization and the motion application via TML commands sent directly to the other drives.

When higher level coordination is needed, apart from a CANopen master, the iPOS4808 drives can also be controlled via a PC or a PLC using one of the **TML_LIB** motion libraries.

For iPOS4808 commissioning **EasySetUp** or **EasyMotion Studio** PC applications may be used.

EasySetUp is a subset of EasyMotion Studio, including only the drive setup part. The output of EasySetUp is a set of setup data that can be downloaded into the drive EEPROM or saved on a PC file. At power-on, the drive is initialized with the setup data read from its EEPROM. With EasySetUp it is also possible to retrieve the complete setup information from a drive previously programmed. EasySetUp shall be used for drive setup in all cases where the motion commands are sent exclusively from a master. Hence neither the iPOS4808 TML programming capability nor the drive camming mode are used. **EasySetUp can be downloaded free of charge from Technosoft web page.**

EasyMotion Studio platform includes EasySetUp for the drive 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 drives – their capability to execute complex motions without requiring an external motion controller, thanks to their built-in motion controller*s. EasyMotion Studio, shall be used to program motion sequences in TML. This is the iPOS4808 typical operation mode when TMLCAN protocol is selected. EasyMotion Studio shall also be used with CANopen protocol, if the user wants to call TML functions stored in the drive EEPROM or to use the camming mode. With camming mode, EasyMotion Studio offers the possibility to quickly download and test a cam profile and also to create a **.sw** file (see par. 5.2.4) with the cam data. The .sw file can be afterwards stored in a master and downloaded to the drive, wherever needed. **A demo version of EasyMotion Studio (with EasySetUp part fully functional) can be downloaded free of charge from Technosoft web page.**

2.2. Key Features

- Fully digital servo drive suitable for the control of rotary or linear brushless, DC brush, and step motors
- Very compact design
- Standard PCIe 8x mating connector
- Sinusoidal (FOC) or trapezoidal (Hall-based) control of brushless motors
- Open or closed-loop control of 2 and 3-phase steppers
- Various modes of operation, including: torque, speed or position control; position or speed profiles, external analogue reference or sent via a communication bus
- Technosoft Motion Language (TML) instruction set for the definition and execution of motion sequences
- Standalone operation with stored motion sequences
- 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]

---

[1] Available if the master axis sends its position via a communication channel

- 35 Homing modes
- Single-ended and RS-422 differential encoder interface
- Single-ended, open collector Digital Hall sensor interface
- Linear Hall sensor interface
- Analogue Sin/Cos encoder interface (differential 1V_{pp})
- SSI, EnDAT, BiSS absolute encoders and resolver feedback possible with an additional feedback extension module
- Drive temperature sensor
- 2 analogue inputs: 12-bit, 0-5V (Reference, Feedback) or general-purpose
- 8 digital inputs: 5-36V, NPN: Enable, Limit switch +, Limit switch –, 5 general-purpose
- 5 digital outputs : 5-36V, 0.5A, NPN open-collector: Ready, Error, 3 general-purpose
- RS-232 serial interface (up to 115200 bps)
- CAN-bus 2.0B up to 1Mbit/s with hardware axis ID selection[1]
- Two operation modes:
  - **CANopen** – conforming with **CiA 301 v4.2** and **CiA DSP 402 v3.0**
  - **TMLCAN** – intelligent drive conforming with Technosoft protocol for exchanging TML commands via CAN-bus
- 2K $\times$ 16 internal SRAM memory for data acquisition
- 4K $\times$ 16 E^{2}ROM to store TML programs and data
- PWM switching frequency up to 100kHz
- Motor supply: 11-48V
- Logic supply: 9-36V.Separate supply is optional
- Output current:
  - iPOS4808 VX-CAN: 8A[2] continuous; 20A peak
- Operating ambient temperature: 0-40°C (over 40°C with derating)
- Hardware Protections:
  - Short-circuit between motor phases
  - Short-circuit from motor phases to ground
  - Over-voltage
  - Under-voltage
  - Over-current
  - Drive over-temperature

---

[1] 127 addresses in CANopen mode, 196 addresses in TMLCAN mode
[2] 8A cont. with DC, step and BLDC motors (trapezoidal)

2.3. Supported Motor-Sensor Configurations

The position and/or speed are controlled using one feedback sensor.

Sensor type	Brushless PMSM	Brushless BLDC	DC Brush	Stepper 2 phase	Stepper 3 phase
Incr. encoder	Yes	-	Yes	Yes	-
Incr. encoder + Digital Hall	Yes	Yes	-	-	-
Digital halls only	Yes	-	-	-	-
Linear Halls	Yes				
Analogue Sin/Cos encoder	Yes				
Tacho	-	-	Yes	-	-
Open-loop (no sensor)	-	-	-	Yes	Yes
Open-loop (with step loss detection using incr. Encoder)	-	-	-	Yes	Yes
Open-loop with incr. encoder on load	-	-	-	Yes	Yes

2.4. iPOS4808 VX Drive Dimensions
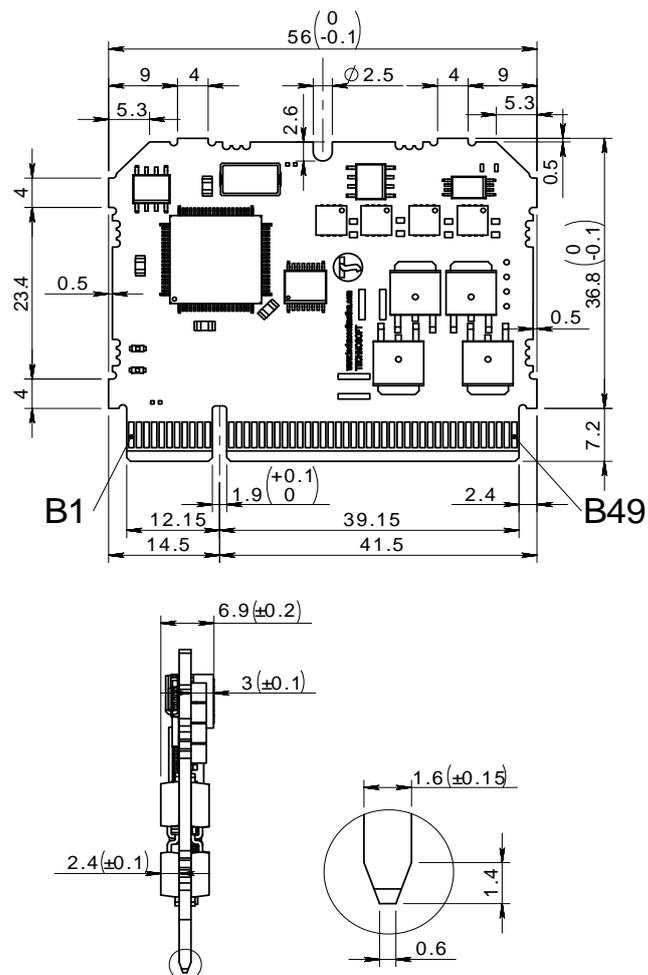
Figure 2.1. *iPOS4808 VX drive dimensions*

All dimensions are in mm. The drawings are not to scale.

2.5. Electrical Specifications

All parameters measured under the following conditions (unless otherwise specified):

T_{amb} = 0…40°C, V_{LOG} = 24 V_{DC}; V_{MOT} = 48V_{DC}; Supplies start-up / shutdown sequence: -*any*-
Load current (sinusoidal amplitude / continuous BLDC,DC,stepper) = 8A iPOS4808

2.5.1. Operating Conditions

		Min.	Typ.	Max.	Units
Ambient temperature[1]		0		+40	ºC
Ambient humidity	Non-condensing	0		90	%Rh
Altitude / pressure[2]	Altitude (referenced to sea level)	-0.1	0 ÷ 2.5	[2]	Km
	Ambient Pressure	0 [2]	0.75 ÷ 1	10.0	atm

2.5.2. Storage Conditions

		Min.	Typ.	Max.	Units
Ambient temperature		-40		TBD	ºC
Ambient humidity	Non-condensing	0		100	%Rh
Ambient Pressure		0		10.0	atm
ESD capability (Human body model)	Not powered; applies to any accessible part			±0.5	kV
	Original packaging			±15	kV

2.5.3. Mechanical Mounting

		Min.	Typ.	Max.	Units
Airflow		natural convection[3], closed box			
Spacing required for vertical mounting (see *Figure 3.2*)	Between adjacent drives	30			mm
	Between drives and nearby walls	30			mm
	Between drives and roof-top	20			mm
Spacing required for horizontal mounting (see *Figure 3.5*)	Between adjacent drives	4			mm
	Between drives and nearby walls	5			mm
	Space needed for drive removal	10			mm
	Between drives and roof-top	15			mm
Insertion force	Using recommended mating connectors		31	55	N
Extraction force		8	16		N

---

[1] Operating temperature at higher temperatures is possible with reduced current and power ratings
[2] iPOS480x can be operated in vacuum (no altitude restriction), but at altitudes over 2,500m, current and power rating are reduced due to thermal dissipation efficiency.
[3] In case of forced cooling (conduction or ventilation) the spacing requirements may drop down to mechanical tolerances as long as the ambient temperature is kept below the maximum operating limit

2.5.4. Environmental Characteristics

		Min.	Typ.	Max.	Units
Size (Length x Width x Height)	Without mating connector		56 x 44 x 6.9		mm
			~2.2 x 1.73 x 0.27		inch
	With recommended mating vertical connector. Height above PCB surface.		56 x 48.1 x 8.9		mm
			~2.2 x 1.89 x 0.35		inch
	With recommended mating horizontal connector. Height above PCB surface.		56 x 56.4 x 9.4		mm
			~2.2 x 2.22 x 0.37		inch
Weight			16		g
Power dissipation	Idle (no load)		1.7		W
	Operating		4.3		
Cleaning agents	Dry cleaning is recommended		Only Water- or Alcohol- based		
Protection degree	According to IEC60529, UL508		IP00		-

2.5.5. Logic Supply Input (+V$_{LOG}$)

		Min.	Typ.	Max.	Units
Supply voltage	Nominal values	9		36	V$_{DC}$
	Absolute maximum values, drive operating but outside guaranteed parameters	8		40	V$_{DC}$
	Absolute maximum values, continuous	-0.6		42	V$_{DC}$
	Absolute maximum values, surge (duration $\leq$ 10ms) [†]	-1		+45	V
Supply current	+V$_{LOG}$ = 7V		125	320	mA
	+V$_{LOG}$ = 12V		85	220	
	+V$_{LOG}$ = 24V		50	145	
	+V$_{LOG}$ = 40V		40	120	

2.5.6. Motor Supply Input (+V$_{MOT}$)

		Min.	Typ.	Max.	Units
Supply voltage	Nominal values	11		50	V$_{DC}$
	Absolute maximum values, drive operating but outside guaranteed parameters	9		52	V$_{DC}$
	Absolute maximum values, continuous	-0.6		54	V$_{DC}$
	Absolute maximum values, surge (duration $\leq$ 10ms) [†]	-1		TBD	V

		Min.	Typ.	Max.	Units
	Idle		1	5	mA
	Operating	-20	±8	+20	A
	Absolute maximum value, short-circuit condition (duration $\leq$ 10ms) [†]			26	A

2.5.7. Motor Outputs (A/A+, B/A-, C/B+, BR/B-)

			Min.	Typ.	Max.	Units
Nominal output current, continuous	for DC brushed, steppers and BLDC motors with Hall-based trapezoidal control	iPOS4808			8	A
	for PMSM motors with FOC sinusoidal control (sinusoidal amplitude value)	iPOS4808			8	
	for PMSM motors with FOC sinusoidal control (sinusoidal effective value)	iPOS4808			5.67	
Motor output current, peak	maximum 2.5s	iPOS4808	-20		+20	A
Short-circuit protection threshold	measurement range	iPOS4808		±26	TBD	A
Short-circuit protection delay			5	10		µS
On-state voltage drop	Nominal output current; including typical mating connector contact resistance			±0.3	±0.5	V
Off-state leakage current				±0.5	±1	mA
Motor inductance (phase-to-phase)	Recommended value, for ripple ±5% of measurement range; +V_{MOT} = 48 V	F_{PWM} = 20 kHz	330			µH
		F_{PWM} = 40 kHz	150			
		F_{PWM} = 60 kHz	120			
		F_{PWM} = 80 kHz	80			
		F_{PWM} = 100 kHz	60			
	Absolute minimum value, limited by short-circuit protection; +V_{MOT} = 48 V	F_{PWM} = 20 kHz	120			µH
		F_{PWM} = 40 kHz	40			
		F_{PWM} = 60 kHz	30			
		F_{PWM} = 80 kHz	15			
		F_{PWM} = 100 kHz	8			
Motor electrical time-constant (L/R)	Recommended value, for ±5% current measurement error due to ripple	F_{PWM} = 20 kHz	250			µs
		F_{PWM} = 40 kHz	125			
		F_{PWM} = 60 kHz	100			
		F_{PWM} = 80 kHz	63			
		F_{PWM} = 100 kHz	50			
Current measurement accuracy	FS = Full Scale			±4	±8	%FS

2.5.8. Digital Inputs (IN0, IN1, IN2/LSP, IN3/LSN, IN4/Enable, IN5, IN6, IN7)

		Min.	Typ.	Max.	Units
Mode compliance			TTL / CMOS / LVTTL (3.3V) / Open-collector / NPN / 24V outputs		
Default state	Input floating (wiring disconnected)		Logic HIGH		
Input voltage	Logic "LOW"		0	0.8	V
	Logic "HIGH"	2	5÷24		
	Floating voltage (not connected)		3		
	Absolute maximum, continuous	-10		+30	
	Absolute maximum, surge (duration ≤ 1s) [†]	-20		+40	
Input current	Logic "LOW"; Pulled to GND		0.6	1	mA
	Logic "HIGH"; Internal 4.7KΩ pull-up to +3.3	0	0	0	
	Logic "HIGH"; Pulled to +5V		0.15	0.2	
	Logic "HIGH"; Pulled to +24V		2	2.5	
Input frequency		0		150	kHz
Minimum pulse width		3.3			µS
ESD protection	Human body model	±5			kV

2.5.9. Digital Outputs (OUT0, OUT1, OUT2/Error, OUT3/ Ready, OUT5)

		Min.	Typ.	Max.	Units
Mode compliance	All outputs (OUT0, OUT1, OUT2/Error, OUT3/Ready)	TTL / CMOS / Open-collector / NPN 24V			
	Ready, Error	Same as above + LVTTL (3.3V)			
Default state	Not supplied (+V_{LOG} floating or to GND)	High-Z (floating)			
	Immediately after power-up — OUT0, OUT1, OUT5	Logic "HIGH"			
	Immediately after power-up — OUT2/Error, OUT3/ Ready	Logic "LOW"			
	Normal operation — OUT0, OUT1, OUT2/Error, OUT5	Logic "HIGH"			
	Normal operation — OUT3/Ready	Logic "LOW"			
Output voltage	Logic "LOW"; output current = 0.5A		0.2	0.8	V
	Logic "HIGH"; output current = 0, no load — OUT2/Error, OUT3/ Ready	2.9	3	3.3	
	Logic "HIGH"; output current = 0, no load — OUT0, OUT1	4	4.5	5	
	Logic "HIGH", external load to +V_{LOG}		V_{LOG}		
	Absolute maximum, continuous	-0.5		V_{LOG}+0.5	
	Absolute maximum, surge (duration $\leq$ 1s) [†]	-1		V_{LOG}+1	
Output current	Logic "LOW", sink current, continuous			0.5	A
	Logic "LOW", sink current, pulse ≤ 5 sec.			1	A
	Logic "HIGH", source current; external load to GND; V_{OUT} >= 2.0V — OUT2/Error, OUT3/ Ready			2	mA
	Logic "HIGH", source current; external load to GND; V_{OUT} >= 2.0V — OUT0, OUT1			4	mA
	Logic "HIGH", leakage current; external load to +V_{LOG}; V_{OUT} = V_{LOG} max = 40V		0.1	0.2	mA
Minimum pulse width		TBD			µS
ESD protection	Human body model	±5			kV

2.5.10. Digital Hall Inputs (Hall1, Hall2, Hall3)

		Min.	Typ.	Max.	Units
Mode compliance		TTL / CMOS / Open-collector			
Default state	Input floating (wiring disconnected)	Logic HIGH			
Input voltage	Logic "LOW"		0	0.8	V
	Logic "HIGH"	2	5		
	Floating voltage (not connected)		4.4		
	Absolute maximum, surge (duration $\leq$ 1s) [†]	-10		+15	
Input current	Logic "LOW"; Pull to GND			5	mA
	Logic "HIGH"; Internal 1KΩ pull-up to +5	0	0	0	
Minimum pulse width		2			µS
ESD protection	Human body model	±5			kV

2.5.11. Encoder Inputs (A/A+, A-, B/B+, B-, Z/Z+, Z-)

		Min.	Typ.	Max.	Units
Single-ended mode compliance	Leave negative inputs disconnected	TTL / CMOS / Open-collector			
Input voltage, single-ended mode A/A+, B/B+	Logic "LOW"			1.6	V
	Logic "HIGH"	1.8			
	Floating voltage (not connected)		4.5		
Input voltage, single-ended mode Z/Z+	Logic "LOW"			1.2	V
	Logic "HIGH"	1.4			
	Floating voltage (not connected)		4.7		
Input current, single-ended mode A/A+, B/B+, Z/Z+	Logic "LOW"; Pull to GND		2.5	3	mA
	Logic "HIGH"; Internal 2.2KΩ pull-up to +5	0	0	0	
Differential mode compliance	For full RS422 compliance, see [1]	TIA/EIA-422-A			
Input voltage, differential mode	Hysteresis	±0.06	±0.1	±0.2	V
	Common-mode range (A+ to GND, etc.)	-7		+7	
Input impedance, differential	A+ to A-, B+ to B-	4.2	4.7		kΩ
	Z+ to Z-	6.1	7.2		
Input frequency	Single-ended mode, Open-collector / NPN	0		500	kHz

		Min.	Typ.	Max.	Units
Input frequency	Differential mode, or Single-ended driven by push-pull (TTL / CMOS)	0		10	MHz
Minimum pulse width	Single-ended mode, Open-collector / NPN	1			µS
	Differential mode, or Single-ended driven by push-pull (TTL / CMOS)	50			nS
Input voltage, any pin to GND	Absolute maximum values, continuous	-7		+7	V
	Absolute maximum, surge (duration ≤ 1s) [†]	-11		+14	
ESD protection	Human body model	±1			kV

2.5.12. Linear Hall Inputs (LH1, LH2, LH3)

		Min.	Typ.	Max.	Units
Input voltage	Operational range	0	0.5÷4.5	4.9	V
	Absolute maximum values, continuous	-7		+7	
	Absolute maximum, surge (duration ≤ 1s) [†]	-11		+14	
Input current	Input voltage 0…+5V	-1	±0.9	+1	mA
Interpolation Resolution	Depending on software settings			11	bits
Frequency		0		1	kHz
ESD protection	Human body model	±1			kV

---

[1] For full RS-422 compliance, 120Ω termination resistors must be connected across the differential pairs, as close as possible to the drive input pins. See *Figure 3.21. Differential incremental encoder connection*

2.5.13. Sin-Cos Encoder Inputs (Sin+, Sin-, Cos+, Cos-)

		Min.	Typ.	Max.	Units
Input voltage, differential	Sin+ to Sin-, Cos+ to Cos-		1	1.25	V_{PP}
Input voltage, any pin to GND	Operational range	-1	2.5	4	V
	Absolute maximum values, continuous	-7		+7	
	Absolute maximum, surge (duration $\leq$ 1s) [†]	-11		+14	
Input impedance	Differential, Sin+ to Sin-, Cos+ to Cos- [1]	4.2	4.7		kΩ
	Common-mode, to GND		2.2		kΩ
Interpolation Resolution	Software selectable, for one sine/cosine period	2		10	bits
Frequency	Sin-Cos interpolation	0		450	kHz
	Quadrature, no interpolation	0		10	MHz
ESD protection	Human body model	±1			kV

2.5.14. Analog 0…5V Inputs (REF, FDBK)

		Min.	Typ.	Max.	Units
Input voltage	Operational range	0		4.95	V
	Absolute maximum values, continuous	-12		+18	
	Absolute maximum, surge (duration $\leq$ 1s) [†]			±36	
Input impedance	To GND		7		kΩ
Resolution			12		bits
Integral linearity				±2	bits
Offset error			±2	±10	bits
Gain error			±1%	±3%	% FS[2]
Bandwidth (-3dB)	Depending on software settings	0		0.8	kHz
ESD protection	Human body model	±5			kV

2.5.15. Axis ID Inputs (AxisID 0, AxisID 1, Axis ID 2)

		Min.	Typ.	Max.	Units
External connections	7 levels		Not connected; Strap to GND; Strap to +5V; 4.7kΩ to GND; 4.7kΩ to +5V; 22kΩ to GND; 22kΩ to +5V;		
Pin current	Use to size PCB tracks			±0.5	mA
4.7kΩ/22kΩ resistor rating		3			mW
4.7kΩ/22kΩ resistor tolerance				±5	%
ESD protection	Human body model	±5			kV

2.5.16. RS-232

		Min.	Typ.	Max.	Units
Standards compliance			TIA/EIA-232-C		
Bit rate	Depending on software settings	9600		115200	Baud
Short-circuit protection	232TX short to GND		Guaranteed		
ESD protection	Human body model	±2			kV

---

[1] For many applications, an 120Ω termination resistor should be connected across SIN+ to SIN-, and across COS+ to COS-. Please consult the feedback device datasheet for confirmation.

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

2.5.17. CAN-Bus

		Min.	Typ.	Max.	Units
Standards compliance		ISO11898, CiA-301v4.2 & 402v3.0			
Bit rate	Depending on software settings	125		1000	Kbps
Bus length	1Mbps			25	m
	800Kbps			50	
	500Kbps			100	
	≤ 250Kbps			250	
Termination resistor	Between CAN-Hi, CAN-Lo	none on-board			
Node addressing	Strapping option (AxisID0,1,2)	1 ÷ 127 (CANopen); 1-195 & 255 (TMLCAN)			-
Voltage, CAN-Hi or CAN-Lo to GND		-26		-26	V
ESD protection	Human body model	±15			kV

2.5.18. Supply Output (+5V)

		Min.	Typ.	Max.	Units
+5V output voltage	Current sourced = 250mA	4.8	5	5.2	V
+5V output current		600	650		mA
Short-circuit protection		NOT protected			
Over-voltage protection		NOT protected			
ESD protection	Human body model	±1			kV

† 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.

2.5.19. **iPOS4808 de-rating curves**
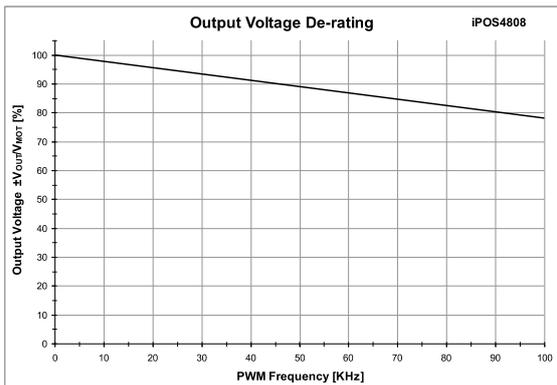
Figure 2.2. *iPOS4808 VX-CAN Output Voltage De-rating with PWM frequency[1]*
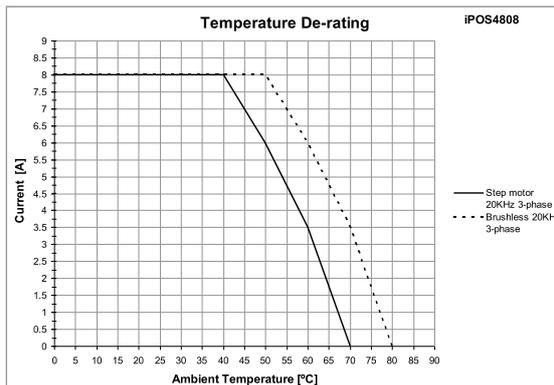
Figure 2.3. *iPOS4808 VX-CAN De-rating with ambient temperature*
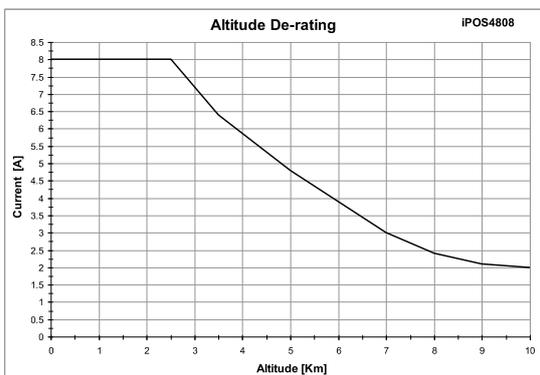
Figure 2.4. *iPOS4808 VX-CAN De-rating with altitude*
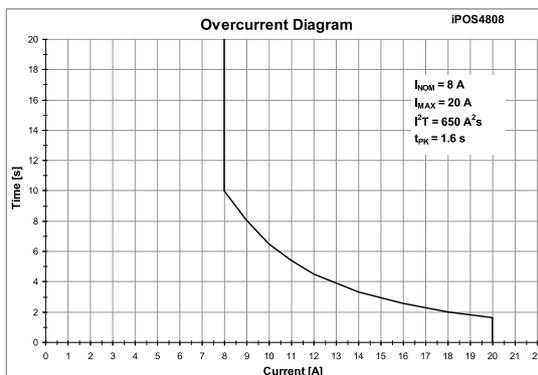
Figure 2.5. *iPOS4808 VX-CAN Over-current diagram*

⚠	**CAUTION!**	*FOR PWM FREQUENCIES LESS THAN 20 KHZ, CORRELATE THE PWM FREQUENCY WITH THE MOTOR PARAMETERS IN ORDER TO AVOID POSSIBLE MOTOR DAMAGE.*

---

[1] V_{OUT} – the output voltage, V_{MOT} – the motor supply voltage

3. Step 1. Hardware Installation

3.1. Mechanical Mounting

The iPOS4808 drive is intended to be mounted vertically or horizontally on a motherboard equipped with the recommended mating connectors, as specified in chapter 3.3. Several iPOS4808 drives can be hosted by a single motherboard.

3.1.1. Vertical Mounting

When iPOS4808 is mounted vertically, its overall envelope (size) including the recommended mating connector is shown in *Figure 3.1.* The use of a fixing case holding the drive is strongly recommended, to avoid vibration and shock problems.
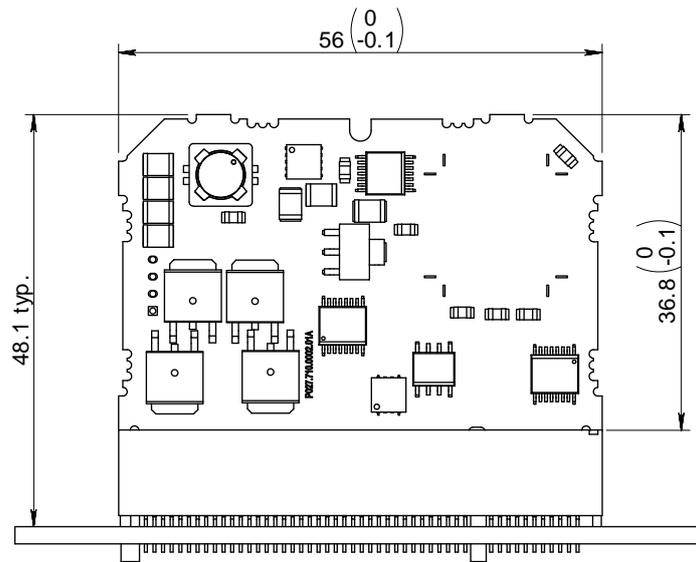
Figure 3.1 *Overall dimensions using recommended mating connector*

The ensemble motherboard + iPOS4808 drive(s) can be cooled by natural convection. The motherboard can be mounted horizontally or vertically. In both cases, the air temperature must not exceed the limits indicated.

Figure 3.2 shows the recommended spacing to assure proper airflow by natural convection, *in the worst case* – closed box done from a plastic (non-metallic) material with no ventilation openings.

Whenever possible, ventilation openings shall be foreseen on the top side wall or the box and at the bottom of the lateral walls. When using a horizontal motherboard considerably larger than the size of the hosted iPOS4808 drives, it is recommended to provide ventilation holes also in the motherboard PCB.

Remark: *In case of using a metallic box, with ventilation openings, all spacing values may be reduced substantially. With proper ventilation, keeping the air surrounding the iPOS4808 inside the limits indicated. The spacing values may be reduced down to the mechanical tolerance limits of Figure 3.1.*
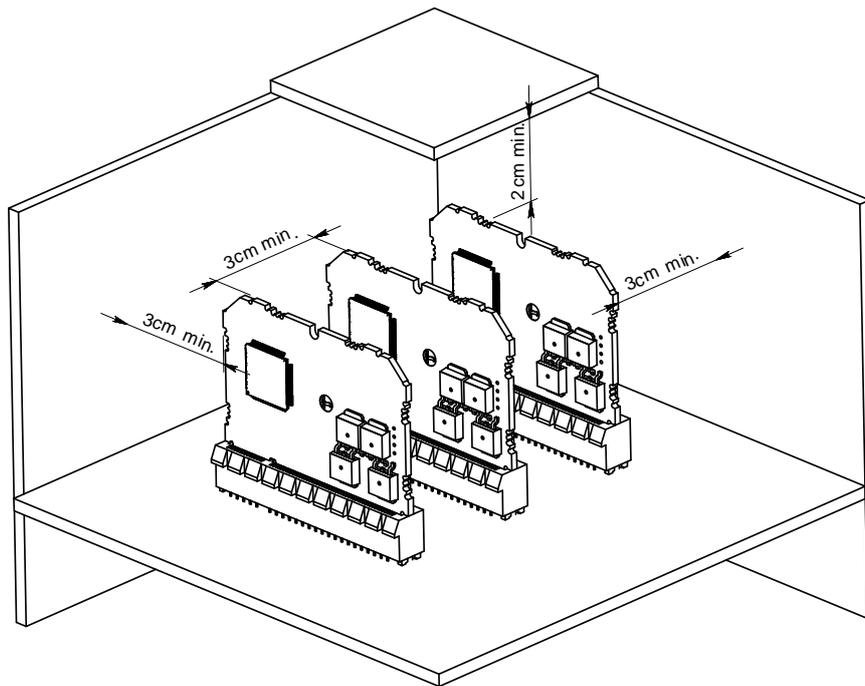
Figure 3.2 Recommended spacing for vertical mounting, <u>worst case</u>: non-metallic, closed box

For ventilation holes / slots that pierce a metallic wall also used for EMI shielding, the maximum slot length *d* should be less than λ/16, where λ is the wavelength corresponding to maximum filtered frequency (f_f):

$$d < \frac{\lambda}{16} = \frac{3 \cdot 10^8}{16 \cdot f_f}$$

If ventilation driven by natural convection is not enough to maintain the temperature surrounding the iPOS4808 drive(s) inside the limits indicated, then alternate forced cooling methods must be applied.

For thermal calculations, each iPOS4808 drive can be assumed to generate 1.5 Watt (= 5 BTU/hour) at idle, and up to TBD Watt worst case while driving a motor.

3.1.2. Horizontal Mounting

When iPOS4808 is mounted horizontally, its overall envelope (size) including the recommended mating connector is shown in *Figure 3.3*. The iPOS4808 has 6 fixing tabs and one groove that may be used to secure the drive. *Figure 3.4* details the fixing tabs and groove position shown with red arrows.
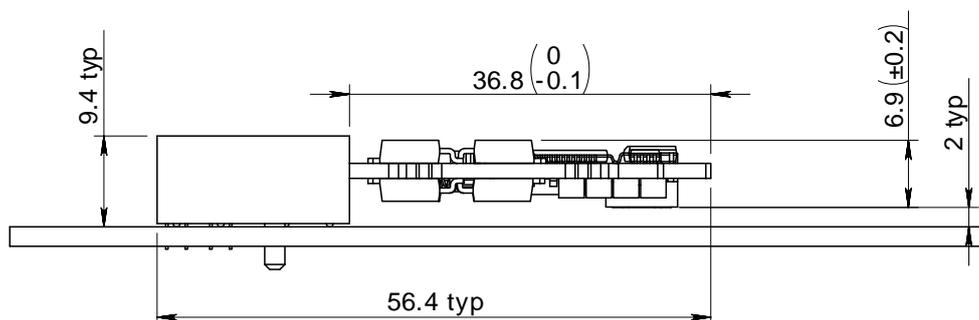
Figure 3.3 Overall dimensions for horizontal mounting
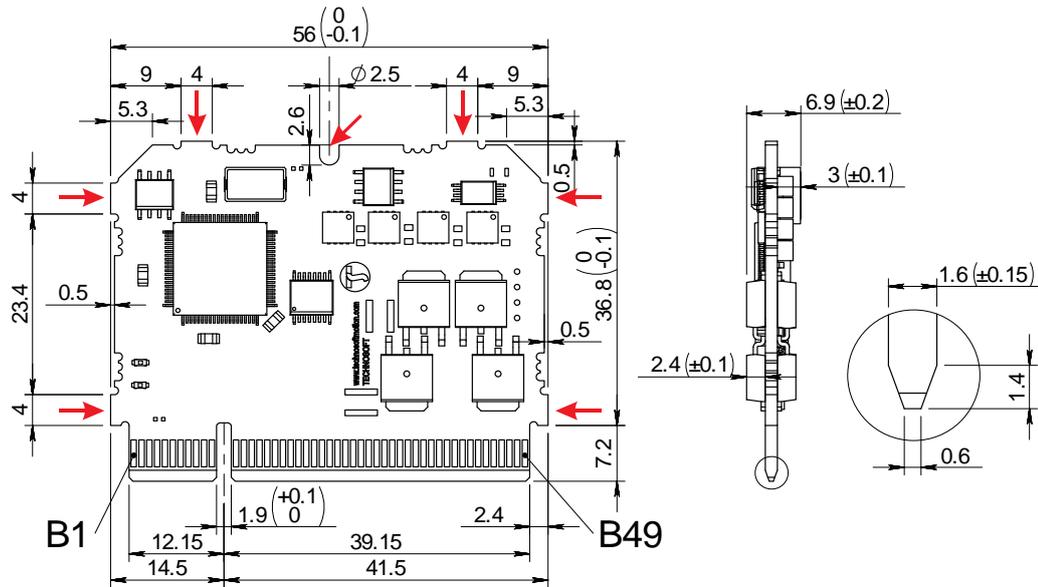
Figure 3.4 *Fixing tabs and groove position*

Figure 3.5 shows the recommended spacing to assure proper airflow by natural convection, *in the worst case* – closed box done from a plastic (non-metallic) material with no ventilation openings.

Whenever possible, ventilation openings shall be foreseen. Recommendations for their size and position are presented in par.3.1.2.

Remark: *In case of using a metallic box, with ventilation openings, all spacing values may be reduced substantially. With proper ventilation, keep the air surrounding the iPOS4808 inside the limits indicated. The spacing values may be reduced down to the mechanical tolerance limits of Figure 3.1.*

3.2. Motherboard PCB Design

The iPOS4808 drive, when plugged vertically into the recommended mating connector, has the following PCB footprint:
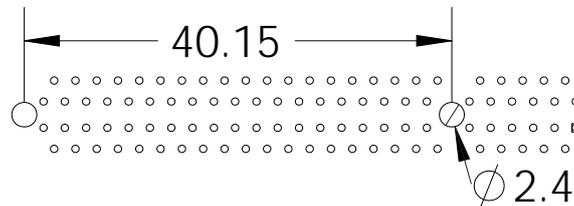
Figure 3.6 PCB footprint of mating connector

It is recommended to use a multi-layer PCB for the motherboard, in order to have enough room for routing all the 98 pins of the iPOS4808. Using a 2-layer PCB is possible when some of the iPOS4808 pins remain un-connected.

Below is a list of recommendations for the PCB design of the motherboard:

- Motor supply and motor outputs: use islands / areas of copper to escape connector area; this will maximize current capability. When using simple tracks, use at least 200mil cross section (150mil track width for 1oz/ft^2 copper thickness) – for iPOS4808.

- Motor supply and ground return tracks between iPOS4808 and the nearby V_{MOT} decoupling capacitor are to be considered as EMI sources, and kept to a minimum length.

- Place the decoupling capacitors on V_{MOT} and V_{LOG} (see also 0. Power Supply Connection) as close as physically possible to the iPOS4808, to minimize EM radiated emissions. For un-shielded applications (no metallic box) and typical EMC regulations, the spacing between iPOS4808 and capacitors must be less than 3 centimeters.

- In multi-axis applications (multiple iPOS4808 drives on the same motherboard), it is preferable to have a separate decoupling capacitor for each drive's V_{MOT}. For V_{LOG} it is acceptable to share one decoupling capacitor for two drives.

- For stringent EMI requirements, it may be necessary to add common-mode filtering on the motor and/or logic supply inputs. Be sure to use 3-phase EMC filters, not 2-phase filters, in order to fulfill the basic requirement of zero common-mode current through the filter. This is necessary because the ground negative return is shared between V_{MOT} and V_{LOG}.

- Motor outputs shall be routed with parallel traces, and minimizing the loop area between these tracks. Avoid placing components above or below the motor output tracks, as these components may become effective antennas radiating EMI. If possible, route all 4 motor outputs in strip-line configuration (above or below a ground plane).

- For stringent EMI requirements, it may be necessary to add common-mode inductors on the motor outputs. Place these filters near the iPOS4808, not near the external connector, to reduce radiation from the PCB tracks.

- Motor outputs must be separated from any nearby track (on the same layer) by a guard ring / track / area connected to ground. It is recommended to use the same guarding precaution also for tracks on nearby layers, i.e. use intermediate guard layer(s) connected to ground. The motor outputs must be treated as first source of noise on the motherboard. Second source of noise is the current flow between each iPOS4808 and it's decoupling V_{MOT} capacitor.

- For best EMC performance, it is strongly recommended to provide an un-interrupted ground plane on one of the inner layers.

- All GND pins of the iPOS4808 are galvanically connected together on-board the iPOS4808. If the motherboard provides an uninterrupted ground plane, it is recommended to connect all GND pins to the ground plane, and use the ground plane to distribute GND wherever needed. If the motherboard does not provide an uninterrupted ground plane, it is best to use each GND pin for it's intended purpose, as described in par. 3.4.3. This will create local "star point" ground connection on-board each iPOS4808. For a multi-axis motherboard with one common power supply for all motors, each motor power supply return track shall be routed separately for each iPOS4808, and star-point connected at the power supply terminal.

- The following signal pairs must be routed differentially, i.e. using parallel tracks with minimal loop area: A+/Sin+ , A-/Sin- ; B+/Cos+ , B-/Cos- ; Z+ , Z- ; CAN-Hi , CAN-Lo.
- CAN-Bus tracks must be routed with a bus topology, without branches / bifurcations, in a daisy-chain fashion. The bus ends must be at the termination resistor(s) and/or external connectors.
- When using +5V$_{OUT}$ as supply for external devices (like encoders, Hall sensors, etc.) provide extra filtering and protection: use series resettable (PTC) fuses to add short-circuit protection; use transient absorbers to protect against ESD and over-voltage; add high-frequency filtering to protect against external noise injected on +5V$_{OUT}$.
- The outer box / case / cabinet must be connected to the motherboard ground either galvanically (directly) or through high-frequency decoupling capacitors, rated at an appropriate voltage.

⚠	**CAUTION !**	*WHEN iPOS4808 IS SET IN TMLCAN MODE, IT STARTS TO EXECUTE AUTOMATICALLY AT POWER ON THE TML APPLICATION FROM ITS EEPROM. ADD ON THE MOTHERBOARD THE POSSIBILITY TO DISABLE THIS FEATURE AS SHOWN PAR. 3.4.14. THIS MIGHT BE NEEDED DURING DEVELOPMENT PHASE IN CASE THE EEPROM CONTENT IS ACCIDENTALLY CORRUPTED.*

	CAUTION!	*THE iPOS4808 IS AN ELECTROSTATICALLY SENSITIVE DEVICE, WHICH WILL BE DAMAGED BY INCORRECT HANDLING. THEREFORE THE DRIVE SHALL BE REMOVED FROM ITS ORIGINAL PACKAGE ONLY IN AN ESD PROTECTED ENVIRONMENT !*

3.3. Mating Connector

Connector	Description	Manufacturer	Part Number
J1	PCIe 8x vertical card edge connector,1.0mm pitch, 2x49 contacts	Tyco	3-1775801-4
		Samtec	PCIE-098-02-F-D-TH
	PCIe 8x horizontal card edge connector,1.0 mm pitch, 2x49 contacts	Tyco	1761465-3
		Samtec	PCIE-098-02-F-D-RA

3.4. Connector and Connection Diagrams
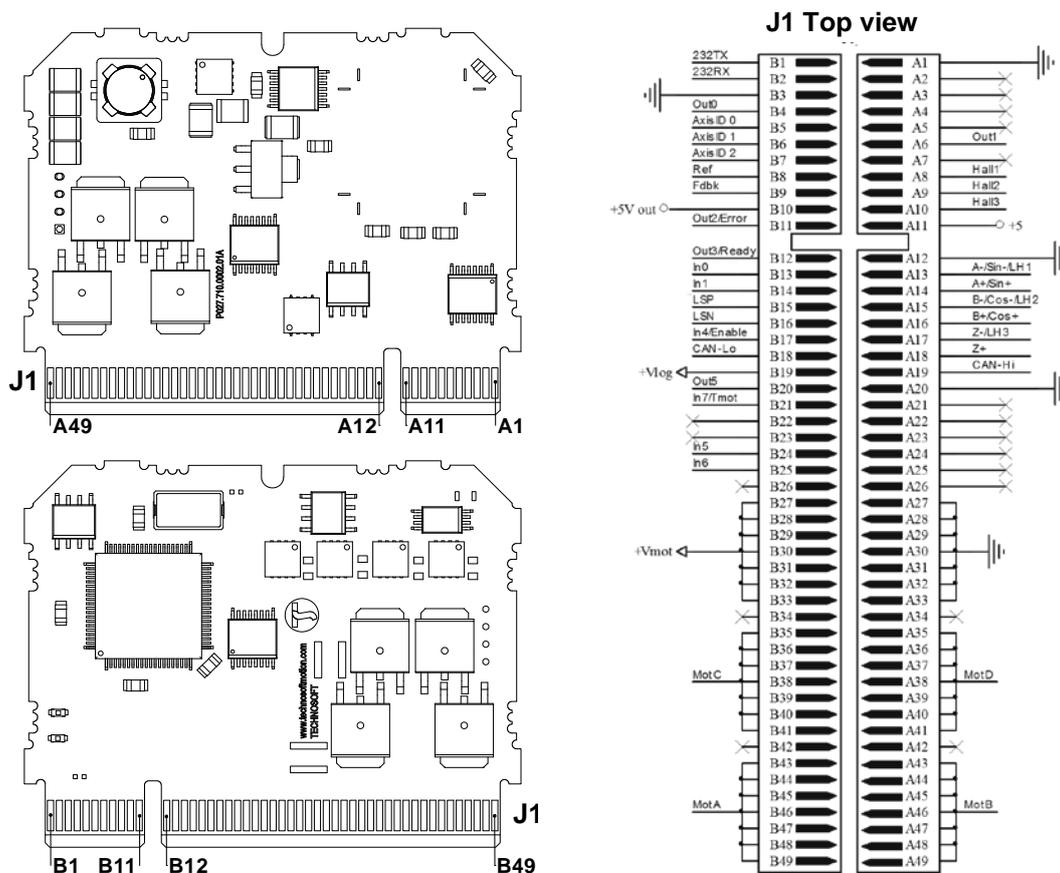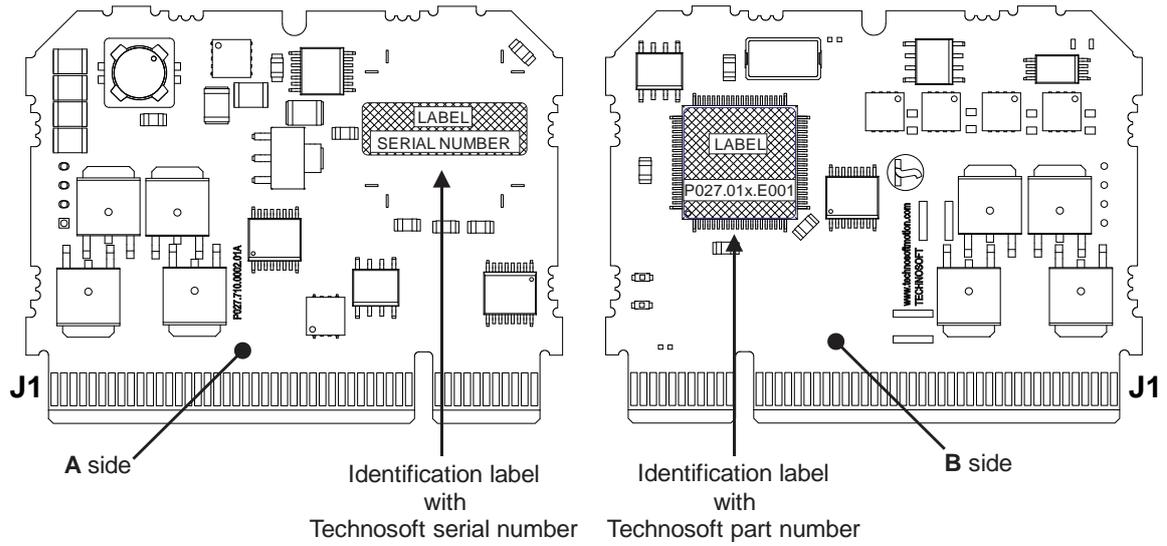
3.4.1. Connector Layout

***Figure 3.7.** iPOS4808 VX drive connector J1*

3.4.2. Identification Labels

Figure 3.8. *iPOS4808 VX-CAN Identification Labels*

3.4.3. J1 Connector pinout (A Side)

	Pin	Name	Type	Description
J1	A1	GND	-	Return ground for extension bus
	A2	reserved	O	Reserved
	A3	reserved	O	Reserved
	A4	reserved	I	Reserved
	A5	reserved	I/O	Reserved
	A6	OUT1	O	5-36V 0.5A digital output, NPN O.C. / TTL pull-up
	A7	reserved	I/O	Reserved
	A8	Hall 1	I	Digital input Hall 1 sensor
	A9	Hall 2	I	Digital input Hall 2 sensor
	A10	Hall 3	I	Digital input Hall 3 sensor
	A11	+5V$_{OUT}$	O	5V supply for sensors - internally generated
	A12	GND	-	Return ground for sensors supply
	A13	A- /Sin- /LH1	I	Incr. encoder A- diff. input, or analogue encoder Sin- diff. input, or linear Hall 1 input
	A14	A/A+/Sin+	I	Incr. encoder A single-ended, or A+ diff. input, or analogue encoder Sin+ diff. input

	Pin	Name	Type	Description
J1	A15	B-/ Cos/LH2	I	Incr. encoder B- diff. input, or analogue encoder Cos- diff. input, or linear Hall 2 input
	A16	B+/ Cos+	I	Incr. encoder B single-ended, or B+ diff. input, or analogue encoder Cos+ diff. input
	A17	Z- /LH3	I	Incr. encoder Z- diff. input, or linear Hall 3 input
	A18	Z+	I	Incr. encoder Z (index) single-ended, or Z+ diff. input
	A19	Can-Hi	I/O	CAN-Bus positive line(dominant high)
	A20	GND	-	Negative return (ground) of the logic supply
	A21-22	reserved	O	Reserved for interface extensions[†]
	A23-26	reserved	I	Reserved
	A27-33	GND	-	Negative return (ground) of the motor supply
	A34	reserved	-	Reserved, not connected
	A35-41	Br/B-	O	Brake resistor / Phase B- for 2-ph steppers
	A42	reserved	-	Reserved, not connected
	A43-49	B/A-	O	Phase B for 3-ph motors, A- for 2-ph steppers, Motor- for DC brush motors

3.4.4. J1 Connector pinout (B Side)

Connector description

	Pin	Name	Type	Description
J1	B1	232TX	O	RS-232 Data Transmission
	B2	232RX	I	RS-232 Data Reception
	B3	GND	-	Return ground for CAN-Bus and RS-232 pins
	B4	OUT0	O	5-36V 0.5A general-purpose digital output, NPN open-collector / TTL pull-up
	B5	AxisID 0	I	Axis ID/Address input. 7 states: floating, strap to GND or +5V, resistor 4K7 or 22K to GND or +5V
	B6	AxisID 1	I	Axis ID/Address input. 7 states: floating, strap to GND or +5V, resistor 4K7 or 22K to GND or +5V
	B7	AxisID 2	I	Axis ID/Address input. 7 states: floating, strap to GND or +5V, resistor 4K7 or 22K to GND or +5V
	B8	REF	I	Analogue input, 12-bit, 0-5V. Used to read an analog position, speed or torque reference, or used as general purpose analogue input
	B9	FDBK	I	Analogue input, 12-bit, 0-5V. Used to read an analogue position or speed feedback (as tacho), or used as general purpose analogue input
	B10	+5V$_{OUT}$	O	5V output supply for I/O usage
	B11	OUT2/ Error	O	5-36V 0.5A drive error output, active low, NPN open-collector/TTL pull-up. Also drives the red LED
	B12	OUT3/ Ready	O	5-36V 0.5A drive ready output, active low, NPN open-collector/TTL pull-up. Also drives the green LED.

	Pin	Name	Type	Description
J1	B13	IN0	I	5-36V digital input General-purpose
	B14	IN1	I	5-36V digital input
	B15	IN2/ LSP	I	5-36V digital input Positive limit switch input
	B16	IN3/ LSN	I	5-36V digital input. Negative limit switch input
	B17	IN4/ Enable	I	5-36V digital input. Drive enable input
	B18	Can-Lo	I/O	CAN-Bus negative line (dominant low)
	B19	+V$_{LOG}$	I	Positive terminal of the logic supply: 9 to 36V$_{DC}$
	B20	OUT5	O	5-36V 0.5A digital output, NPN O.C. / TTL pull-up
	B21	IN7	I	5-36V digital input
	B22	reserved	-	Do not connect
	B23	reserved	I	Reserved
	B24	IN5	I	5-36V digital input General-purpose
	B25	IN6	I	5-36V digital input General-purpose
	B26	reserved	-	Reserved, not connected
	B27-33	+V$_{MOT}$	I	Positive terminal of the motor supply: 11 to 50V$_{DC}$
	B34	reserved	-	Reserved, not connected
	B35-41	C/B+	O	Phase C for 3-ph motors, B+ for 2-ph steppers
	B42	reserved	-	Reserved, not connected
	B43-49	A/A+	O	Phase A for 3-ph motors, A+ for 2-ph steppers, Motor+ for DC brush motors

3.4.5. Operation Mode and Axis ID Selection

3.4.5.1 Selection Levels on Axis ID Inputs

On iPOS4808 the selection of the operation mode: CANopen or TMLCAN as also of the axis ID number is done by setting different voltage levels on the inputs AxisID 0 (pin B5), AxisID 1(pin B6) and AxisID 2 (pin B7). Each input can be set to one of the following 7 levels:

Level	Connection needed
L0	Connect input directly to ground (for example pin B3)
L1	Connect input through a 4.7kΩ resistor to ground
L2	Connected input through a 22kΩ resistor to ground
L3	Nothing connected – leave input open
L4	Connect input through a 22kΩ resistor to +5Vdc (for example pin B10)
L5	Connect input through a 4.7Kohm resistor to +5Vdc
L6	Connect input directly to +5V

The operation mode selection is done via AxisID 2:

- CANopen mode, if the input levels are: L0, L1 or L2
- TMLCAN mode, if the input levels are L3, L4, L5, L6

Figure 3.9 shows how to set the 7 levels on the AxisID 0 input and the resulting axis ID values when AxisID 2 input level is set for **CANopen** operation. *Table 3.1* shows how to set all possible values for axis ID in this mode of operation.

Remarks:

1. *AxisID value is computed with formula: 49 x AxisID2 + 7 x AxisID1 + AxisID0, where each AxisID can have one of the integer values: 0 to 6 (0 for L0, 1 for L1, 2 for L2, etc.)*
2. *If the resulting AxisID value is 0 (all 3 inputs are connected to GND), the axis ID will be set to 127. If the resulting AxisID is greater than 127, the axis ID will be set to 255.*

Figure 3.10 shows how to set the 7 levels on the AxisID 0 input and the resulting axis ID values when AxisID 2 input level is set for **TMLCAN** operation. *Table 3.2. Axis ID setting in TMLCAN mode* shows how to set all possible values for axis ID in this mode of operation.

Remarks:

1. *AxisID is computed with formula: 49 x (AxisID2 - 3) + 7 x AxisID1 + AxisID0, where each AxisID can have one of the integer values: 0 to 6 (0 for L0, 1 for L1, 2 for L2, etc.)*
2. *If the resulting AxisID value is 0, the axis ID will be set to 255*
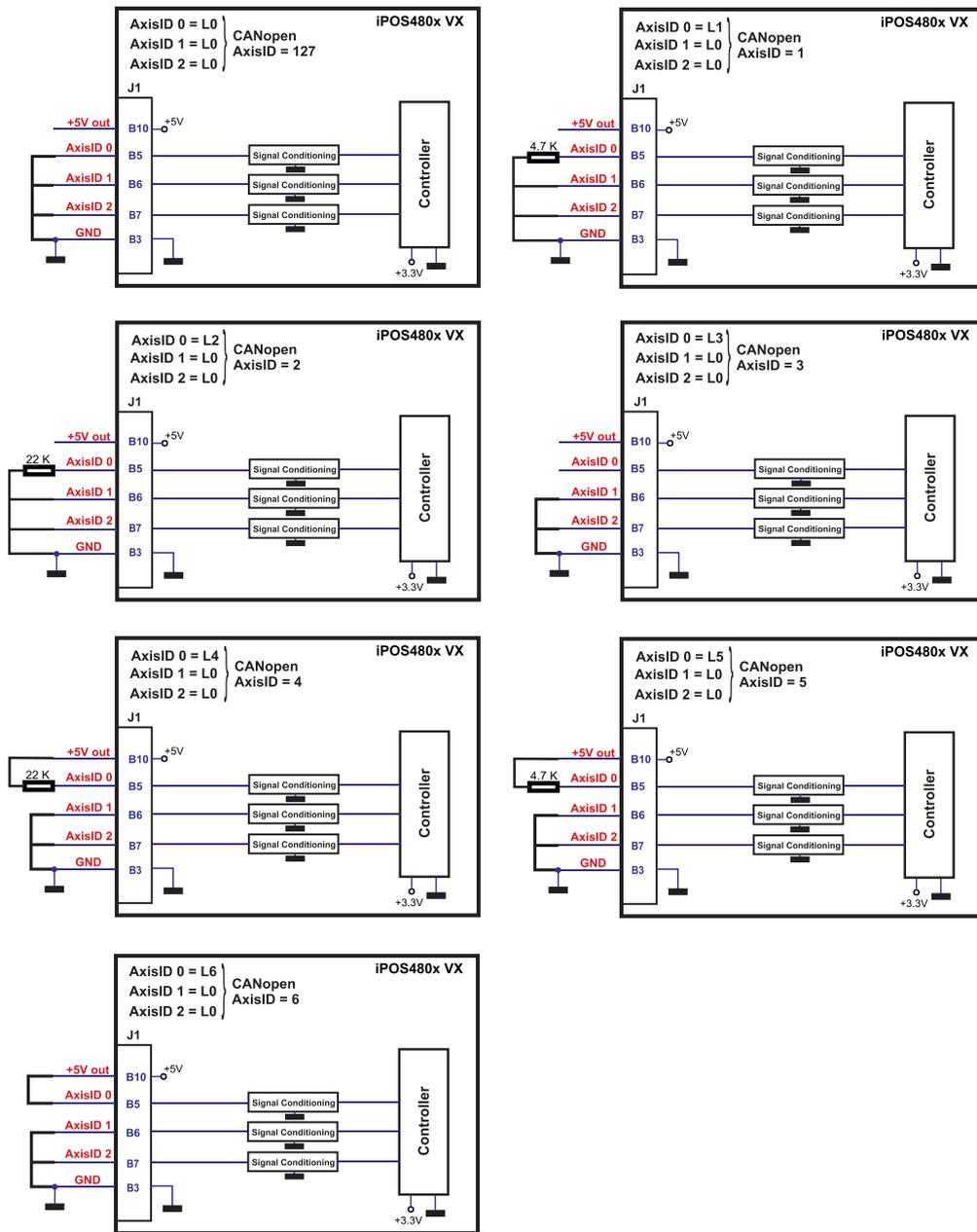
Figure 3.9 *Axis ID Setting Examples. CANopen mode*
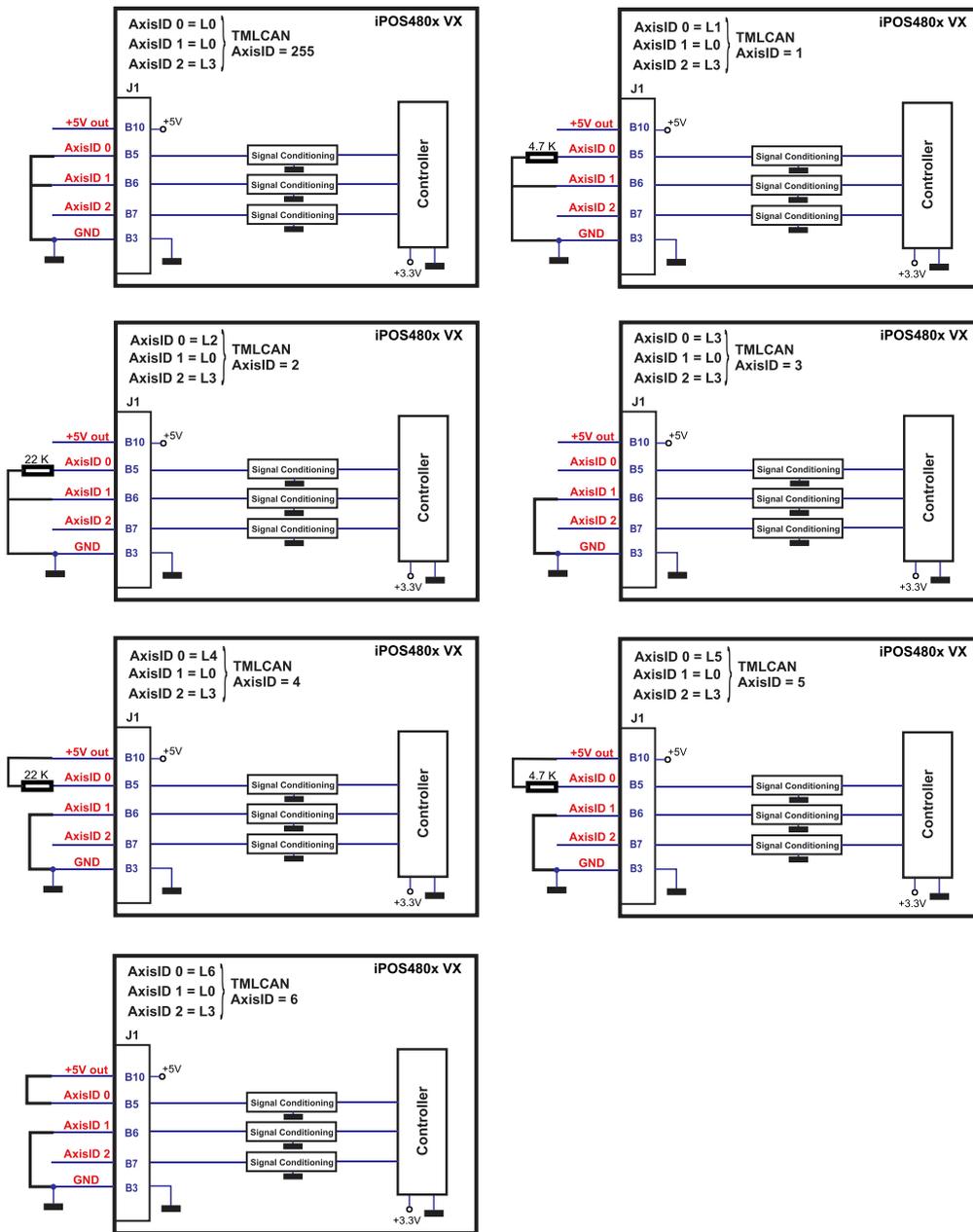
Figure 3.10 *Axis ID Setting Examples. TMLCAN mode*

3.4.5.2 Axis ID Settings

Table 3.1. *Axis ID setting in CANopen mode*

Axis ID 2	Axis ID 1	Axis ID 0	ID CANopen		Axis ID 2	Axis ID 1	Axis ID 0	ID CANopen
L0	L0	L0	127		L0	L5	L1	36
L0	L0	L1	1		L0	L5	L2	37
L0	L0	L2	2		L0	L5	L3	38
L0	L0	L3	3		L0	L5	L4	39
L0	L0	L4	4		L0	L5	L5	40
L0	L0	L5	5		L0	L5	L6	41
L0	L0	L6	6		L0	L6	L0	42
L0	L1	L0	7		L0	L6	L1	43
L0	L1	L1	8		L0	L6	L2	44
L0	L1	L2	9		L0	L6	L3	45
L0	L1	L3	10		L0	L6	L4	46
L0	L1	L4	11		L0	L6	L5	47
L0	L1	L5	12		L0	L6	L6	48
L0	L1	L6	13		L1	L0	L0	49
L0	L2	L0	14		L1	L0	L1	50
L0	L2	L1	15		L1	L0	L2	51
L0	L2	L2	16		L1	L0	L3	52
L0	L2	L3	17		L1	L0	L4	53
L0	L2	L4	18		L1	L0	L5	54
L0	L2	L5	19		L1	L0	L6	55
L0	L2	L6	20		L1	L1	L0	56
L0	L3	L0	21		L1	L1	L1	57
L0	L3	L1	22		L1	L1	L2	58
L0	L3	L2	23		L1	L1	L3	59
L0	L3	L3	24		L1	L1	L4	60
L0	L3	L4	25		L1	L1	L5	61
L0	L3	L5	26		L1	L1	L6	62
L0	L3	L6	27		L1	L2	L0	63
L0	L4	L0	28		L1	L2	L1	64
L0	L4	L1	29		L1	L2	L2	65
L0	L4	L2	30		L1	L2	L3	66
L0	L4	L3	31		L1	L2	L4	67
L0	L4	L4	32		L1	L2	L5	68
L0	L4	L5	33		L1	L2	L6	69
L0	L4	L6	34		L1	L3	L0	70
L0	L5	L0	35		L1	L3	L1	71

Axis ID 2	Axis ID 1	Axis ID 0	ID CANopen
L1	L3	L2	72
L1	L3	L3	73
L1	L3	L4	74
L1	L3	L5	75
L1	L3	L6	76
L1	L4	L0	77
L1	L4	L1	78
L1	L4	L2	79
L1	L4	L3	80
L1	L4	L4	81
L1	L4	L5	82
L1	L4	L6	83
L1	L5	L0	84
L1	L5	L1	85
L1	L5	L2	86
L1	L5	L3	87
L1	L5	L4	88
L1	L5	L5	89
L1	L5	L6	90
L1	L6	L0	91
L1	L6	L1	92
L1	L6	L2	93
L1	L6	L3	94
L1	L6	L4	95
L1	L6	L5	96
L1	L6	L6	97
L2	L0	L0	98
L2	L0	L1	99
L2	L0	L2	100
L2	L0	L3	101
L2	L0	L4	102
L2	L0	L5	103
L2	L0	L6	104
L2	L1	L0	105
L2	L1	L1	106
L2	L1	L2	107
L2	L1	L3	108
L2	L1	L4	109

Axis ID 2	Axis ID 1	Axis ID 0	ID CANopen
L2	L1	L5	110
L2	L1	L6	111
L2	L2	L0	112
L2	L2	L1	113
L2	L2	L2	114
L2	L2	L3	115
L2	L2	L4	116
L2	L2	L5	117
L2	L2	L6	118
L2	L3	L0	119
L2	L3	L1	120
L2	L3	L2	121
L2	L3	L3	122
L2	L3	L4	123
L2	L3	L5	124
L2	L3	L6	125
L2	L4	L0	126
L2	L4	L1	255
L2	L4	L2	255
L2	L4	L3	255
L2	L4	L4	255
L2	L4	L5	255
L2	L4	L6	255
L2	L5	L0	255
L2	L5	L1	255
L2	L5	L2	255
L2	L5	L3	255
L2	L5	L4	255
L2	L5	L5	255
L2	L5	L6	255
L2	L6	L0	255
L2	L6	L1	255
L2	L6	L2	255
L2	L6	L3	255
L2	L6	L4	255
L2	L6	L5	255
L2	L6	L6	255

Table 3.2. *Axis ID setting in TMLCAN mode*

Axis ID 2	Axis ID 1	Axis ID 0	ID TMLCAN
L3	L0	L0	255
L3	L0	L1	1
L3	L0	L2	2
L3	L0	L3	3
L3	L0	L4	4
L3	L0	L5	5
L3	L0	L6	6
L3	L1	L0	7
L3	L1	L1	8
L3	L1	L2	9
L3	L1	L3	10
L3	L1	L4	11
L3	L1	L5	12
L3	L1	L6	13
L3	L2	L0	14
L3	L2	L1	15
L3	L2	L2	16
L3	L2	L3	17
L3	L2	L4	18
L3	L2	L5	19
L3	L2	L6	20
L3	L3	L0	21
L3	L3	L1	22
L3	L3	L2	23
L3	L3	L3	24
L3	L3	L4	25
L3	L3	L5	26
L3	L3	L6	27
L3	L4	L0	28
L3	L4	L1	29
L3	L4	L2	30
L3	L4	L3	31
L3	L4	L4	32
L3	L4	L5	33
L3	L4	L6	34
L3	L5	L0	35
L3	L5	L1	36

Axis ID 2	Axis ID 1	Axis ID 0	ID TMLCAN
L3	L5	L2	37
L3	L5	L3	38
L3	L5	L4	39
L3	L5	L5	40
L3	L5	L6	41
L3	L6	L0	42
L3	L6	L1	43
L3	L6	L2	44
L3	L6	L3	45
L3	L6	L4	46
L3	L6	L5	47
L3	L6	L6	48
L4	L0	L0	49
L4	L0	L1	50
L4	L0	L2	51
L4	L0	L3	52
L4	L0	L4	53
L4	L0	L5	54
L4	L0	L6	55
L4	L1	L0	56
L4	L1	L1	57
L4	L1	L2	58
L4	L1	L3	59
L4	L1	L4	60
L4	L1	L5	61
L4	L1	L6	62
L4	L2	L0	63
L4	L2	L1	64
L4	L2	L2	65
L4	L2	L3	66
L4	L2	L4	67
L4	L2	L5	68
L4	L2	L6	69
L4	L3	L0	70
L4	L3	L1	71
L4	L3	L2	72
L4	L3	L3	73

Axis ID 2	Axis ID 1	Axis ID 0	ID TMLCAN
L4	L3	L4	74
L4	L3	L5	75
L4	L3	L6	76
L4	L4	L0	77
L4	L4	L1	78
L4	L4	L2	79
L4	L4	L3	80
L4	L4	L4	81
L4	L4	L5	82
L4	L4	L6	83
L4	L5	L0	84
L4	L5	L1	85
L4	L5	L2	86
L4	L5	L3	87
L4	L5	L4	88
L4	L5	L5	89
L4	L5	L6	90
L4	L6	L0	91
L4	L6	L1	92
L4	L6	L2	93
L4	L6	L3	94
L4	L6	L4	95
L4	L6	L5	96
L4	L6	L6	97
L5	L0	L0	98
L5	L0	L1	99
L5	L0	L2	100
L5	L0	L3	101
L5	L0	L4	102
L5	L0	L5	103
L5	L0	L6	104
L5	L1	L0	105
L5	L1	L1	106
L5	L1	L2	107
L5	L1	L3	108
L5	L1	L4	109
L5	L1	L5	110
L5	L1	L6	111
L5	L2	L0	112
L5	L2	L1	113

Axis ID 2	Axis ID 1	Axis ID 0	ID TMLCAN
L5	L2	L2	114
L5	L2	L3	115
L5	L2	L4	116
L5	L2	L5	117
L5	L2	L6	118
L5	L3	L0	119
L5	L3	L1	120
L5	L3	L2	121
L5	L3	L3	122
L5	L3	L4	123
L5	L3	L5	124
L5	L3	L6	125
L5	L4	L0	126
L5	L4	L1	127
L5	L4	L2	128
L5	L4	L3	129
L5	L4	L4	130
L5	L4	L5	131
L5	L4	L6	132
L5	L5	L0	133
L5	L5	L1	134
L5	L5	L2	135
L5	L5	L3	136
L5	L5	L4	137
L5	L5	L5	138
L5	L5	L6	139
L5	L6	L0	140
L5	L6	L1	141
L5	L6	L2	142
L5	L6	L3	143
L5	L6	L4	144
L5	L6	L5	145
L5	L6	L6	146
L6	L0	L0	147
L6	L0	L1	148
L6	L0	L2	149
L6	L0	L3	150
L6	L0	L4	151
L6	L0	L5	152

Axis ID 2	Axis ID 1	Axis ID 0	ID TMLCAN		Axis ID 2	Axis ID 1	Axis ID 0	ID TMLCAN
L6	L0	L6	153		L6	L4	L0	175
L6	L1	L0	154		L6	L4	L1	176
L6	L1	L1	155		L6	L4	L2	177
L6	L1	L2	156		L6	L4	L3	178
L6	L1	L3	157		L6	L4	L4	179
L6	L1	L4	158		L6	L4	L5	180
L6	L1	L5	159		L6	L4	L6	181
L6	L1	L6	160		L6	L5	L0	182
L6	L2	L0	161		L6	L5	L1	183
L6	L2	L1	162		L6	L5	L2	184
L6	L2	L2	163		L6	L5	L3	185
L6	L2	L3	164		L6	L5	L4	186
L6	L2	L4	165		L6	L5	L5	187
L6	L2	L5	166		L6	L5	L6	188
L6	L2	L6	167		L6	L6	L0	189
L6	L3	L0	168		L6	L6	L1	190
L6	L3	L1	169		L6	L6	L2	191
L6	L3	L2	170		L6	L6	L3	192
L6	L3	L3	171		L6	L6	L4	193
L6	L3	L4	172		L6	L6	L5	194
L6	L3	L5	173		L6	L6	L6	195
L6	L3	L6	174					

3.4.6. 24V Digital I/O Connection

Figure 3.11. *24V Digital I/O connection*

Remarks:
1. *Connect the external load to the external supply +V_{LOG} for using the 24V outputs*
2. *The maximum sink current is 0.5A continuous, up to 1A pulsed for less than 5 seconds*
3. *The inputs are compatible with NPN type outputs (input must be pulled to GND to change it's default state)*
4. *The outputs are compatible with NPN type inputs (load is tied to common +V_{LOG}, output pulls to GND when active and is floating when inactive)*

3.4.7. 5V Digital I/O Connection
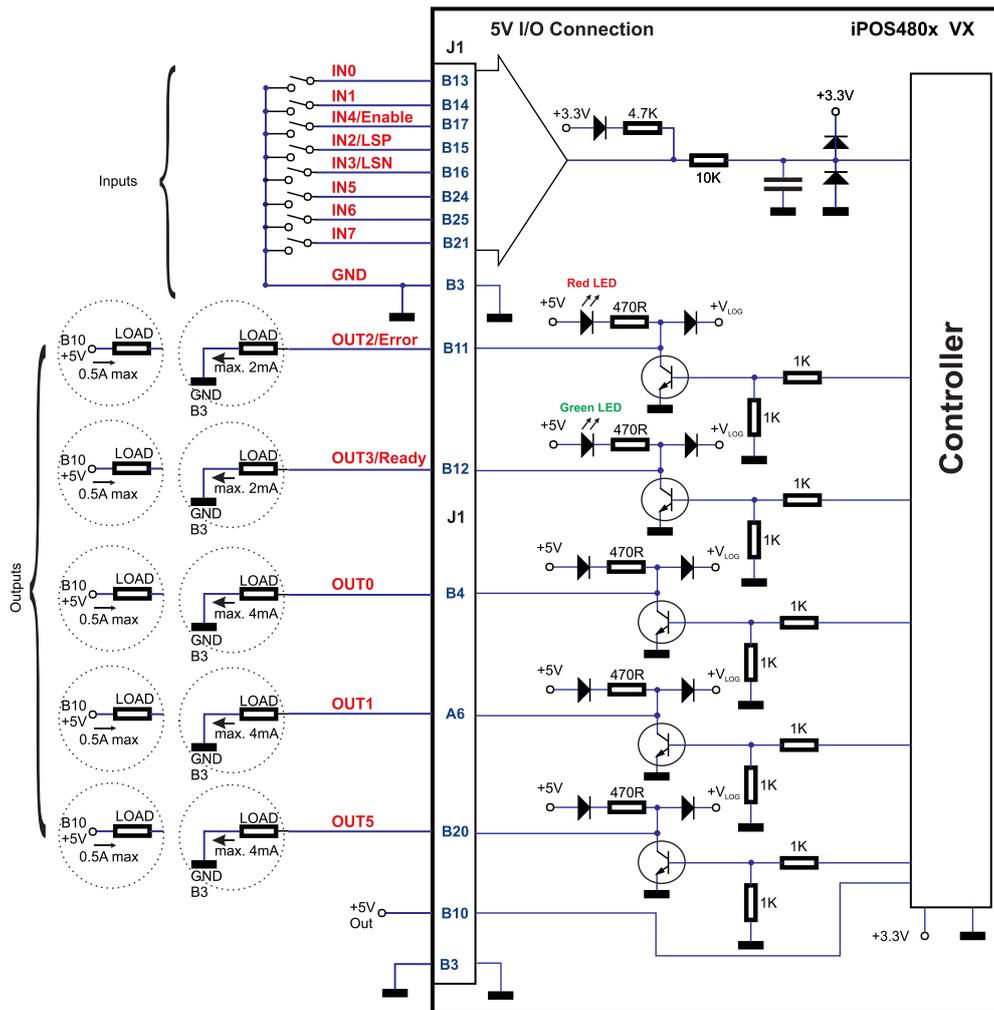
Figure 3.12. *5V Digital I/O connection*

Remarks:
1. *The inputs are compatible with TTL(5V), LVTTL(3.3V), CMOS and open collector outputs*
2. *The outputs are compatible with TTL (5V) inputs*
3. *The output loads can be individually and independently connected to +5V or to GND.*

3.4.8. Analog Inputs Connection

3.4.8.1 0-5V Input Range
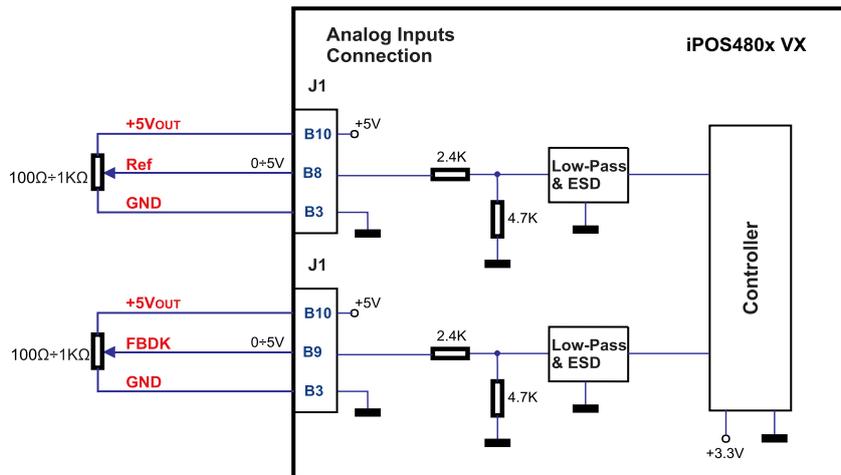
Figure 3.13. *Analog inputs connection*

Remark: *Default input range for analog inputs is 0÷5 V for REF and FBDK. For a +/-10 V range, see Figure 3.14.*

3.4.8.2 +/- 10V to 0-5V Input Range Adapter
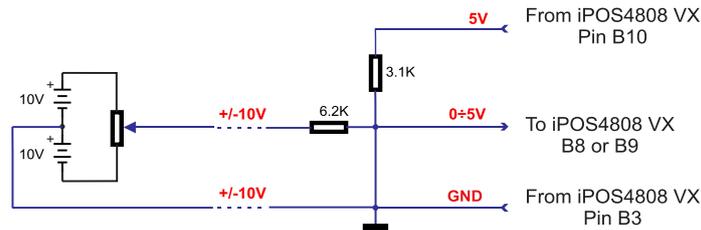
Figure 3.14. *+/-10V to 0-5V adapter*

3.4.8.3 Recommendation for wiring

a) If the analogue signal source is single-ended, use a 2-wire twisted shielded cable as follows: 1st wire connects the live signal to the drive input; 2nd wire connects the source ground to the drive ground; shield will be connected to the drive ground terminal.

b) If the analogue signal source is differential and the signal source ground is isolated from the drive GND, use a 2-wire twisted shielded cable as follows: 1st wire connects the source plus (positive, in-phase) to the drive analogue input; 2nd wire connects the source minus (negative, out-of-phase) to the drive ground (GND). Shield is connected only at the drive side, to the drive GND, and is left unconnected at the source side.

c) If the analogue signal source is differential and the signal source ground is common with the drive GND, use a 2-wire shielded cable as follows: 1st wire connects the source plus (positive, in-phase) to the drive analogue input; 2nd wire connects the source ground to the drive ground (GND); shield is connected only at the drive side, to the drive GND, and is left unconnected at the source side. The source minus (negative, out-of-phase) output remains unconnected.

3.4.9. Motor connections

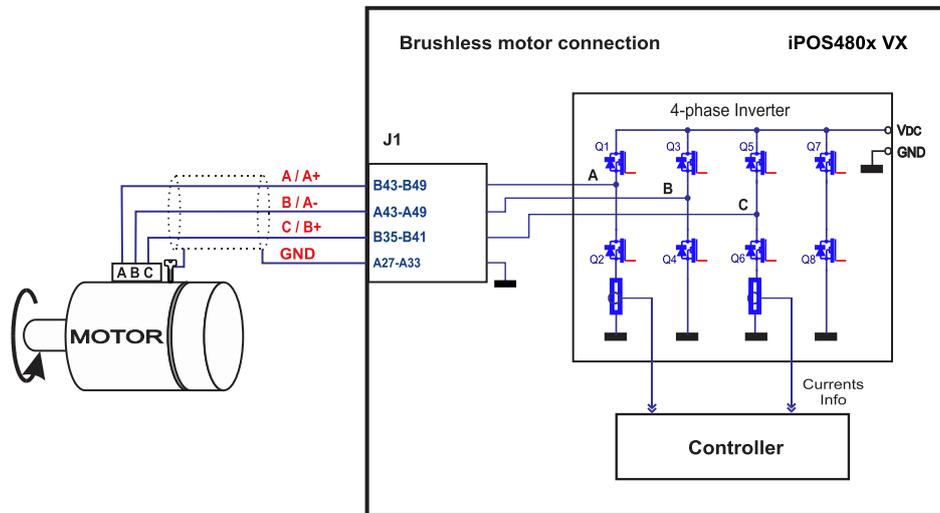#### 3.4.9.1 Brushless Motor connection

Figure 3.15. *Brushless motor connection*

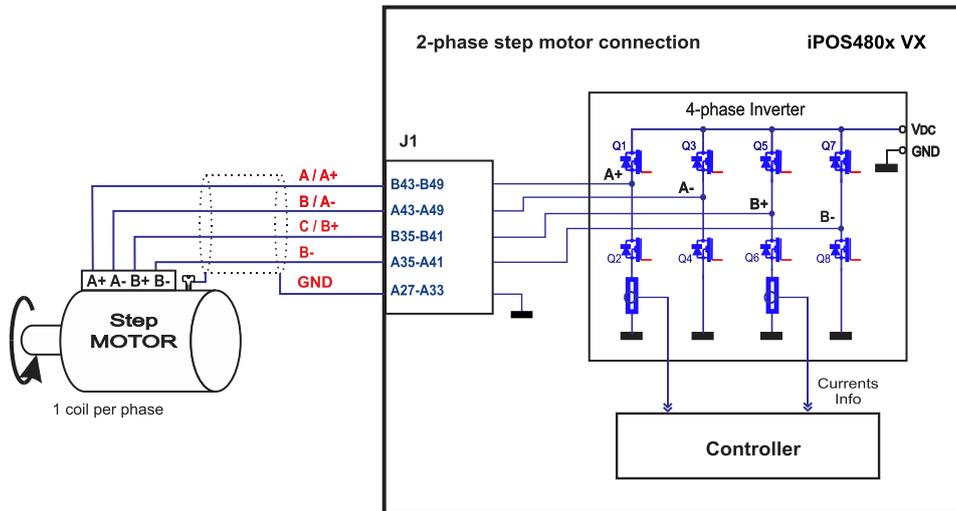#### 3.4.9.2 2-phase Step Motor connection

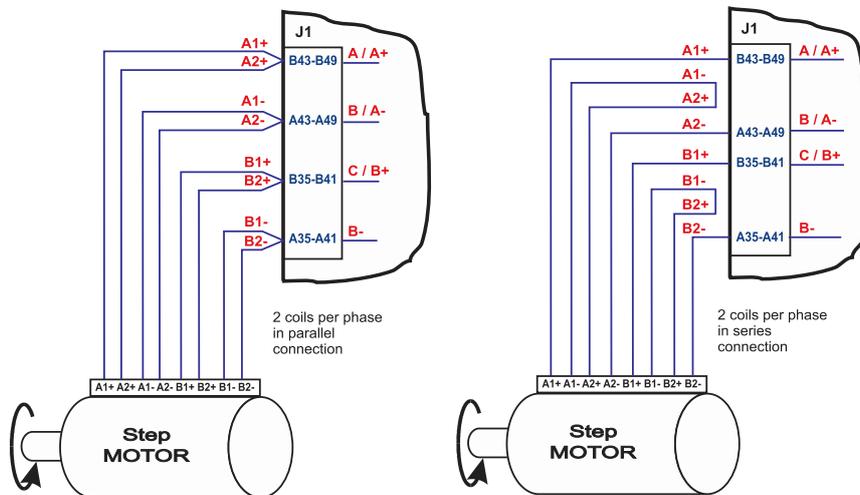**Figure 3.16.** *2-phase step motor connection, one coil per phase*

Figure 3.17. *2-phase step motor connection, two coils per phase*
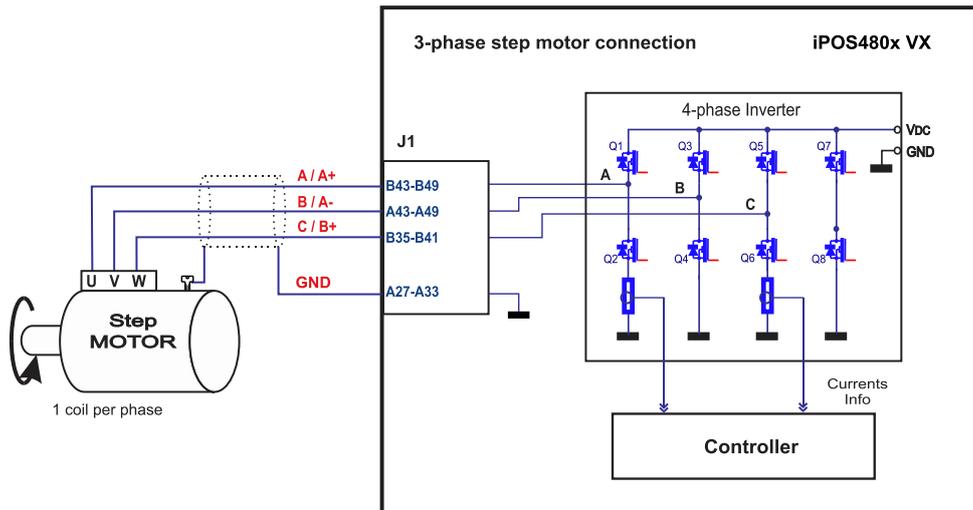

3.4.9.3   3-Phase Step Motor connection

Figure 3.18. *3-phase step motor connection*
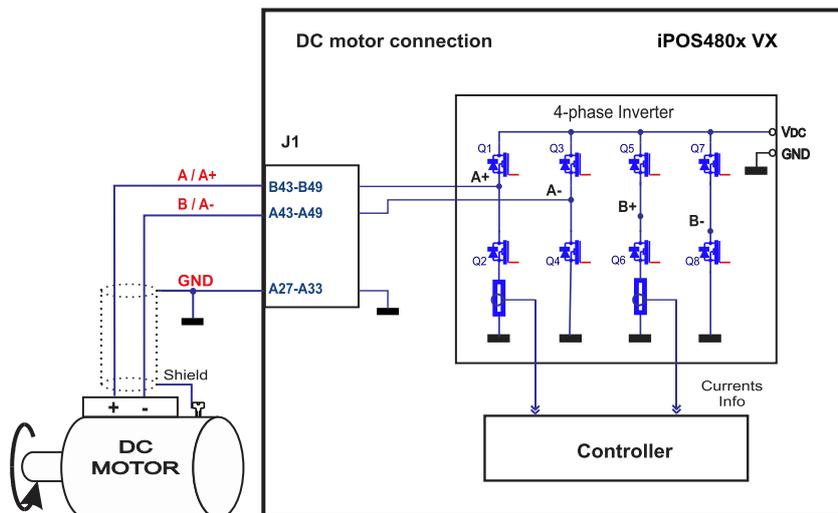
3.4.9.4   DC Motor connection

Figure 3.19. *DC Motor connection*

3.4.9.5   Recommendations for motor wiring

a) Avoid running the motor wires in parallel with other wires for a distance longer than 2 meters. If this situation cannot be avoided, use a shielded cable for the motor wires. Connect the cable shield to the iPOS4808 GND pin. Leave the other end disconnected.

b) The parasitic capacitance between the motor wires must not bypass 10nF. If very long cables (tens of meters) are used, this condition may not be met. In this case, add series inductors between the iPOS4808 outputs and the cable. The inductors must be magnetically shielded (toroidal, for example), and must be rated for the motor surge current. Typically the necessary values are around 100 µH.

c) A good shielding can be obtained if the motor wires are running inside a metallic cable guide.

3.4.10. Feedback connections

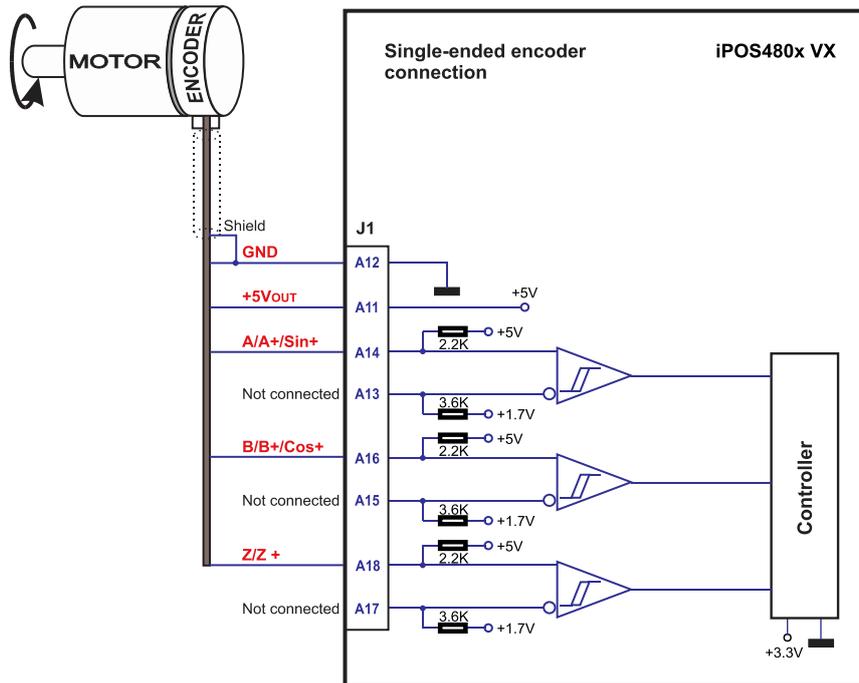#### 3.4.10.1 Single-ended Incremental Encoder Connection

Figure 3.20. *Single-ended incremental encoder connection*

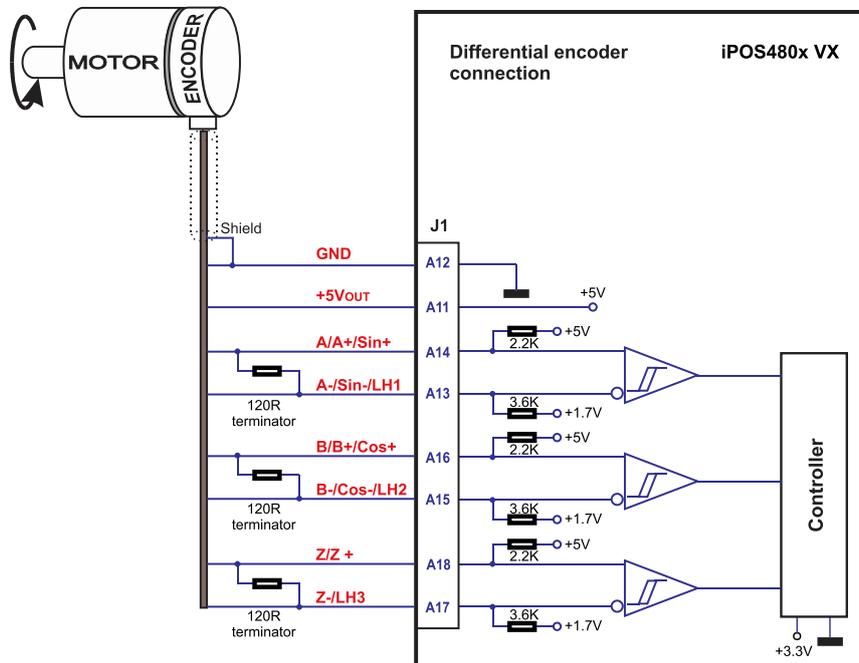#### 3.4.10.2 Differential Incremental Encoder Connection

Figure 3.21. *Differential incremental encoder connection*

Remark: *120Ω (0.25W) terminators are required for long encoder cables, or noisy environments.*
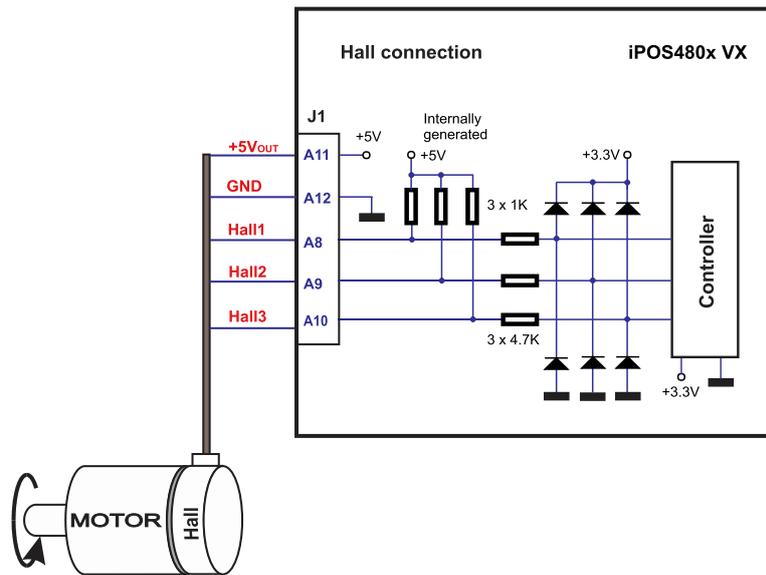
3.4.10.3 Digital Hall Connection

Figure 3.22. *Digital Hall connection*
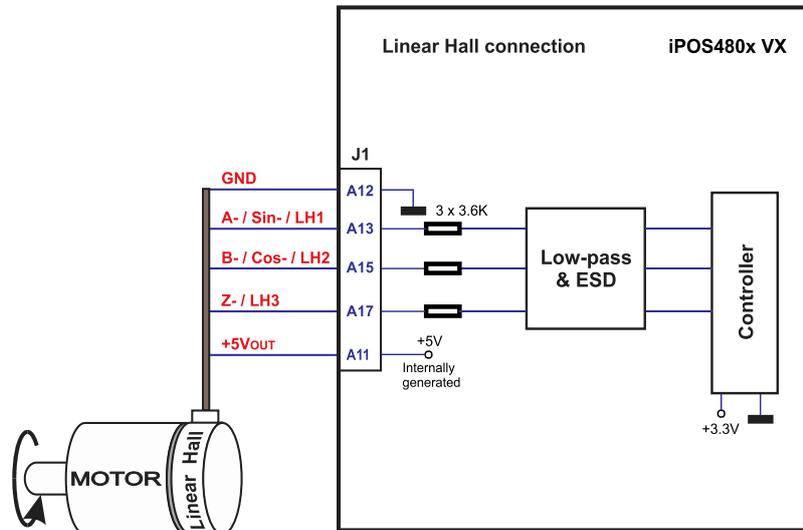
3.4.10.4 Linear Hall Connection

Figure 3.23. *Linear Hall connection*

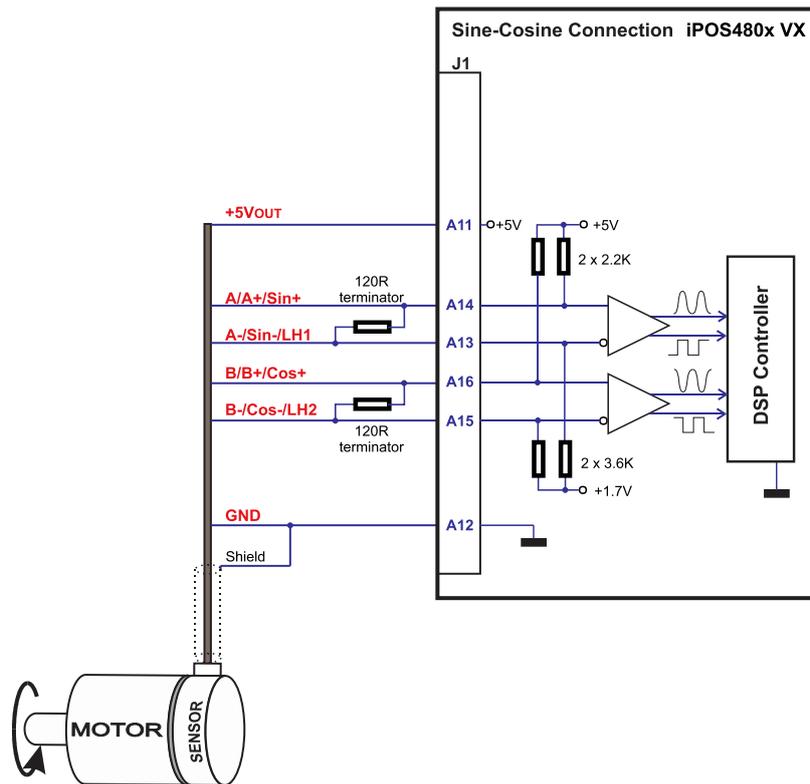### 3.4.10.5 Sine-Cosine Analog Encoder Connection

Figure 3.24. *Sine-Cosine analogue encoder connection*

3.4.10.6 Recommendations for wiring

a) Always connect both positive and negative signals when the position sensor is differential and provides them. Use one twisted pair for each differential group of signals as follows: A+/Sin+ with A-/Sin-/LH1, B+/Cos+ with B-/Cos-/LH2, Z+ with Z-/LH3. Use another twisted pair for the 5V supply and GND.

b) Always use shielded cables to avoid capacitive-coupled noise when using single-ended encoders or Hall sensors with cable lengths over 1 meter. Connect the cable shield to the GND, at only one end. This point could be either the iPOS4808 (using the GND pin) or the encoder / motor. Do not connect the shield at both ends.

c) If the iPOS4808 5V supply output is used by another device (like for example an encoder) and the connection cable is longer than 5 meters, add a decoupling capacitor near the supplied device, between the +5V and GND lines. The capacitor value can be 1...10 µF, rated at 6.3V.

3.4.11. **Power Supply Connection**

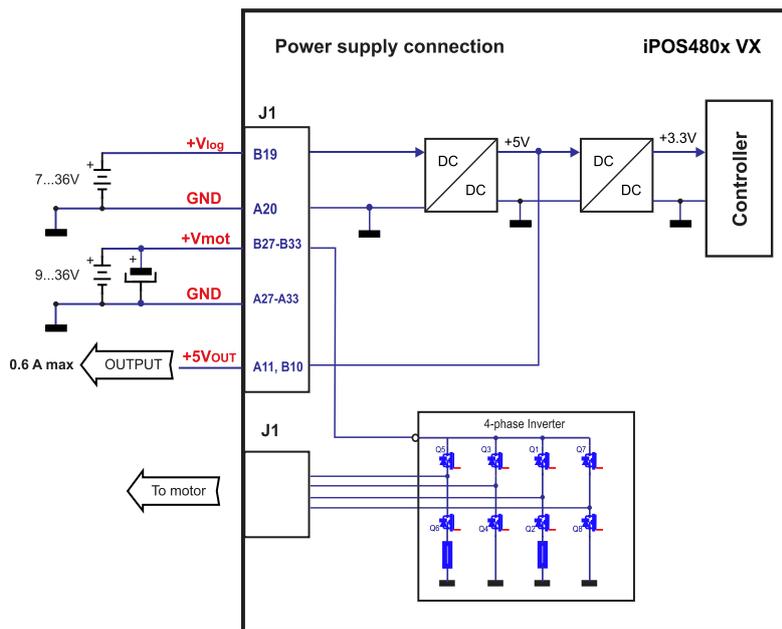#### 3.4.11.1 Supply Connection

Figure 3.25. *Supply connection*

3.4.11.2 Recommendations for Supply Wiring

Always provide a nearby capacitor on the motor supply lines. The capacitor should be located within 10cm of the iPOS4808 edge connector, max. 20cm. The minimum recommended capacitance is 330µF for iPOS4808, always rated at the appropriate voltage.

Use short, thick wires between the iPOS4808 and the motor power supply. Connect power supply wires to all the indicated pins. 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 2,200µF (rated at an appropriate voltage) right on the terminals of the iPOS4808.

3.4.11.3 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 53V, the drive over-voltage protection is triggered and the drive power stage is disabled. In order to avoid this situation you have 2 options:

Option 1. 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:

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

where:

   U_{MAX} = 53V is the over-voltage protection limit

   U_{NOM} is the nominal motor supply voltage

   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:

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

where:

  J_M – total rotor inertia [kgm^2]
  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]
  $h_{initial}$ – initial system altitude [m]
  h_{final} – final system altitude [m]
  I_M – motor current during deceleration [A$_{RMS}$/phase]
  R_{Ph} – motor phase resistance [Ω]
  t_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].

Option 2. Connect a brake resistor R_{BR} between phase BR/B- (pins A35-A41) and ground (pins A23+A24+B23+B24), and activate the software option of dynamic braking (see below).

This option is not available when the drive is used with a step motor.

The dynamic braking option can be found in the Drive Setup dialogue within EasyMotion / EasySetup. The braking will occur when DC bus voltage increases over U_{BRAKE}. This parameter (U_{BRAKE}) should be adjusted depending on the nominal motor supply. Optimally (from a braking point of view), U_{BRAKE} should be a few volts above the maximum nominal supply voltage. This setting will activate the brake resistor earlier, before reaching dangerous voltages – when the over-voltage protection will stop the drive. Of course, U_{BRAKE} must always be less than U_{MAX} – the over-voltage protection threshold.

Remark: *This option can be combined with an external capacitor whose value is not enough to absorb the entire regenerative energy E_M but can help reducing the brake resistor size.*

Brake resistor selection
The brake resistor value must be chosen to respect the following conditions:
1. to limit the maximum current below the drive peak current I_{PEAK} = 20A

$$R_{BR} > \frac{U_{MAX}}{I_{PEAK}}$$

2. to sustain the required *braking power*:

$$P_{BR} = \frac{E_M - \frac{1}{2}C(U_{MAX}^2 - U_{brake}^2)}{t_d}$$

where C is the capacitance on the motor supply (external), i.e:

$$R_{BR} < \frac{U_{BRAKE}^2}{2 \times P_{BR}}$$

3. to limit the average current below the drive nominal current I_{NOM}=10A

$$R_{BR} > \frac{P_{BR} \times t_d}{t_{CYCLE} \times I_{NOM}^2}$$

where t_{CYCLE} is the time interval between 2 brakes in case of repetitive moves.

4. to be rated for an average power $P_{AV} = \frac{P_{BR} \times t_d}{t_{CYCLE}}$ and a peak power $P_{PEAK} = \frac{U_{MAX}^2}{R_{BR}}$

Remarks:

1. If $\dfrac{U_{MAX}}{I_{PEAK}} > \dfrac{U_{BRAKE}^2}{2 \times P_{BR}}$ the braking power P_{BR} must be reduced by increasing either t_d – the

   time to decelerate or C – the external capacitor on the motor supply

2. If $\dfrac{P_{BR} \times t_d}{t_{CYCLE} \times I_{NOM}^2} > \dfrac{U_{BRAKE}^2}{2 \times P_{BR}}$ either the braking power must be reduced (see Remark 1)

   or t_{CYCLE} – the time interval between braking cycles must be increased

⚡	**WARNING!**	*THE BRAKE RESISTOR MAY HAVE HOT SURFACES DURING OPERATION.*

3.4.12. Serial RS-232 connection

3.4.12.1 Serial RS-232 connection
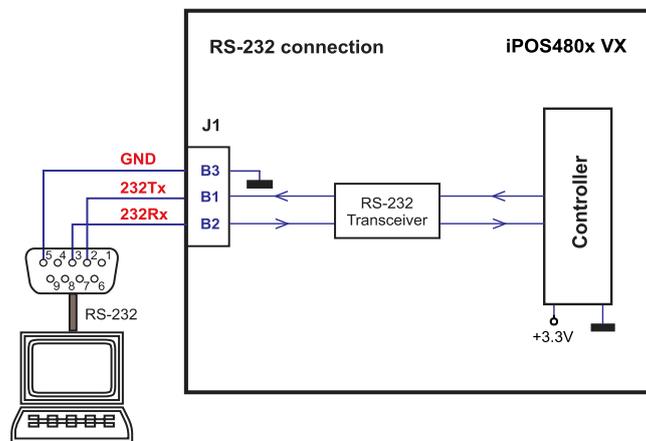
Figure 3.26. *Serial RS-232 connection*

3.4.12.2 Recommendation for wiring

a) If you build the serial cable, you can use a 3-wire shielded cable with shield connected to BOTH ends. Do not use the shield as GND. The ground wire (pin B3 of J1) must be included inside the shield, like the 232Rx and 232Tx signals

b) Always power-off all the iPOS4808 supplies before inserting/removing the RS-232 serial connector

c) Do not rely on an earthed PC to provide the iPOS4808 GND connection! The drive must be earthed through a separate circuit. Most communication problems are caused by the lack of such connection

3.4.13. CAN-bus connection
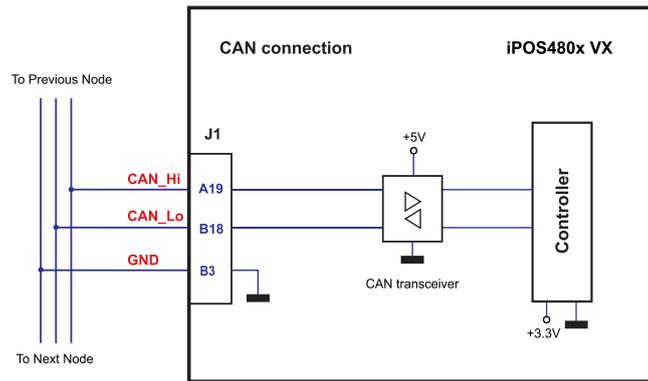
3.4.13.1 CAN connection

Figure 3.27. *CAN connection*

Remarks:
1. *The CAN network requires a 120-Ohm terminator. This is not included on the board. Figure 3.28 shows how to connect it on your network*
2. *CAN signals are not insulated from other iPOS4808 circuits.*

3.4.13.2 Recommendation for wiring

a) Build CAN network using cables with twisted wires (2 wires/pair), with CAN-Hi twisted together with CAN-Lo. It is recommended but not mandatory to use a shielded cable. If so, connect the shield to GND. The cable impedance must be 105 ... 135 ohms (120 ohms typical) and a capacitance below 30pF/meter.

b) When using a printed circuit board (PCB) motherboard based on FR-4 material, build the CAN network using a pair of 12mil (0.012") tracks, spaced 8 to 10mils (0.008"…0.010") apart, placed over a local ground plane (microstrip) which extends at least 1mm left and right to the tracks.

c) 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.

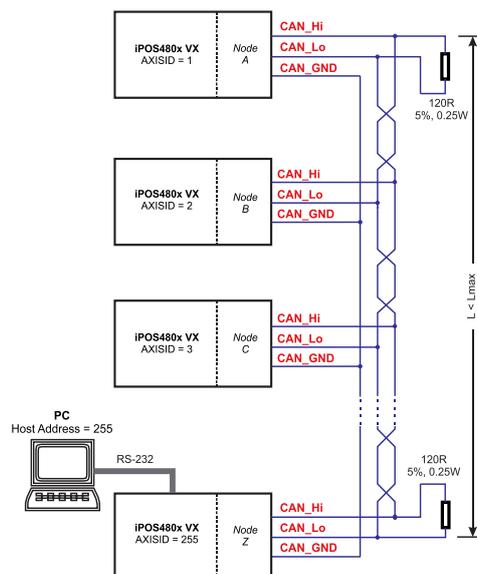d) The 120Ω termination resistors must be rated at 0.2W minimum. Do not use winded resistors, which are inductive.

***Figure 3.28.*** *Multiple-Axis CAN network*

3.4.14. Removal from Autorun Mode

When iPOS4808 is set in TMLCAN operation mode, it enters by default after power on in *Autorun* mode, if the drive has in its local EEPROM a valid TML application (motion program), this is automatically executed as soon as the motor supply V_{MOT} is turned on.

In order to remove the drive from *Autorun*, you have 2 ways:

a) Software - by writing value 0x0001 in first EEPROM location, from address 0x4000;

b) Hardware – by temporary connecting all digital Hall inputs to GND, during the power on for about 1s (until the green led is turned on), as shown in *Figure 3.29*. This option is particularly useful when it is not possible to communicate with the drive.

After the drive is set in *non-Autorun/slave* mode using 2nd method, the 1st method may be used to invalidate the TML application from the EEPROM. On next power on, in absence of a valid TML application, the drive enters in the *non-Autorun/slave* mode independently of the digital Hall inputs status.
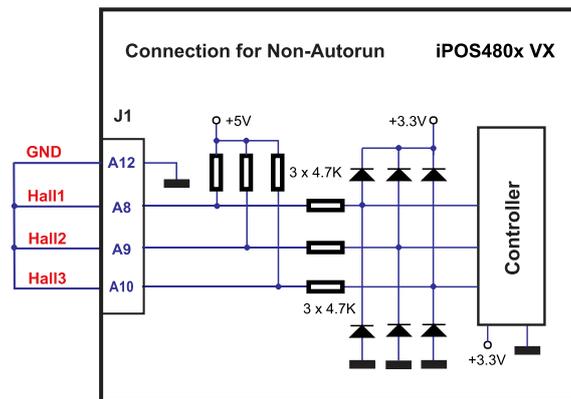
Figure 3.29. *Temporary connection during power-on to remove the drive from **Autorun** mode*

4. Step 2. Drive Setup

4.1. Installing EasySetUp

EasySetUp is a PC software platform for the setup of the Technosoft drives. 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 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 advanced features presented in Section 5.3. 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 drive for your application. The drive can be connected with your PC in one of the following ways:

1. Via an RS232 link, directly connected to the PC, or via an USB to RS232 adapter or using Technosoft Ethernet to RS232 adapter, function of your PC communication interfaces;
2. Via a CAN-bus link, directly connected to the PC through a PC-CAN interface, or using Technosoft Ethernet to CAN adapter
3. Via another drive from the same CAN-bus network connected to the PC via one of the above options from point 1.

The output of EasySetUp is a set of *setup data*, which can be downloaded into the drive 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 drive commissioning.

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

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

- ▪ Use a **CANopen** master to control the iPOS4808 as a standard CANopen drive
- ▪ Use **EasyMotion Studio** to create and download a **TML** program into the drive/motor memory
- ▪ Use one of the **TML_LIB** motion libraries to control the drives/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 drives/motors using one of the supported communication channels. The implementation must be done according with Technosoft communication protocols.
- **Combine** TML programming at drive 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 drive/motor.
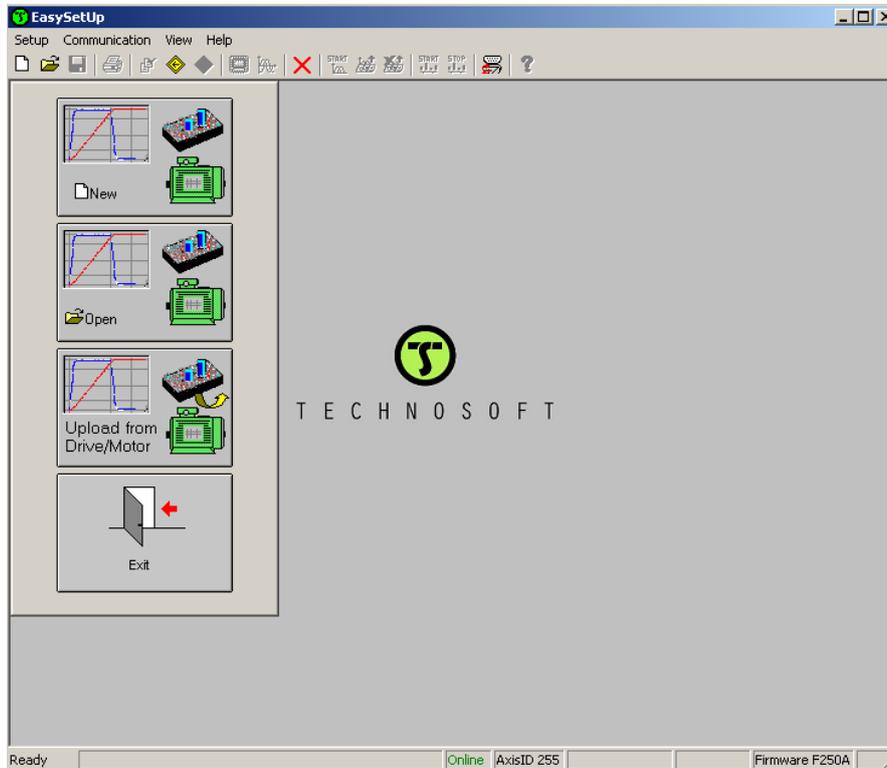

Before selecting one of the above options, you need to establish the communication with the drive you want to commission. Use menu command **Communication | Setup** to check/change your PC communication settings. Press the **Help** button of the dialogue opened. Here you can find detailed information about how to setup your drive and do the connections. Power on the drive, 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 drive/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 drive having axis ID=255 (default communication settings). If the drive has a different axis ID and you don't know it, select in the Communication | Setup dialogue at "Axis ID of drive/motor connected to PC" the option **Autodetected**. If this drive is part of a CANbus network,, use the menu command **Communication | Scan Network***

4.2.2. **Setup drive/motor**
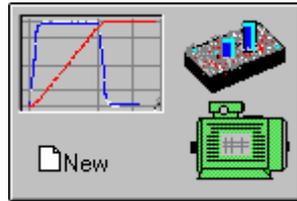
Press **New** button and select your drive type.
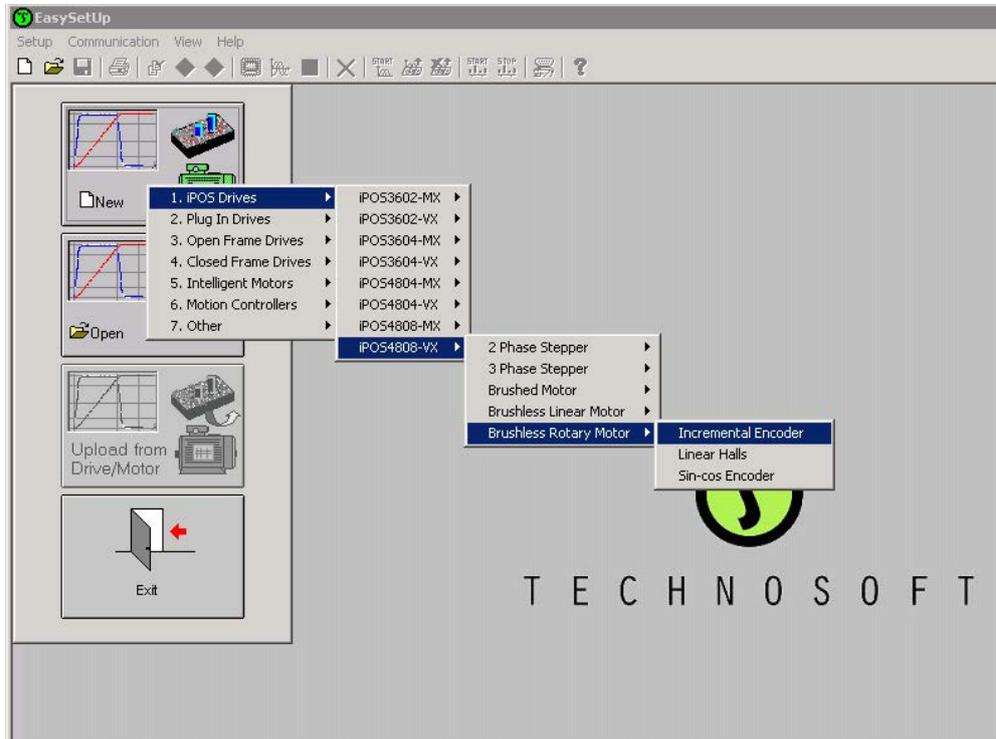
The selection continues with the motor technology (for example: brushless or brushed) and type of feedback device (for example: Incremental encoder, Linear Halls).

The selection opens 2 setup dialogues: for **Motor Setup** and for **Drive setup** through which you can configure and parameterize a Technosoft drive, plus several predefined control panels customized for the product selected.
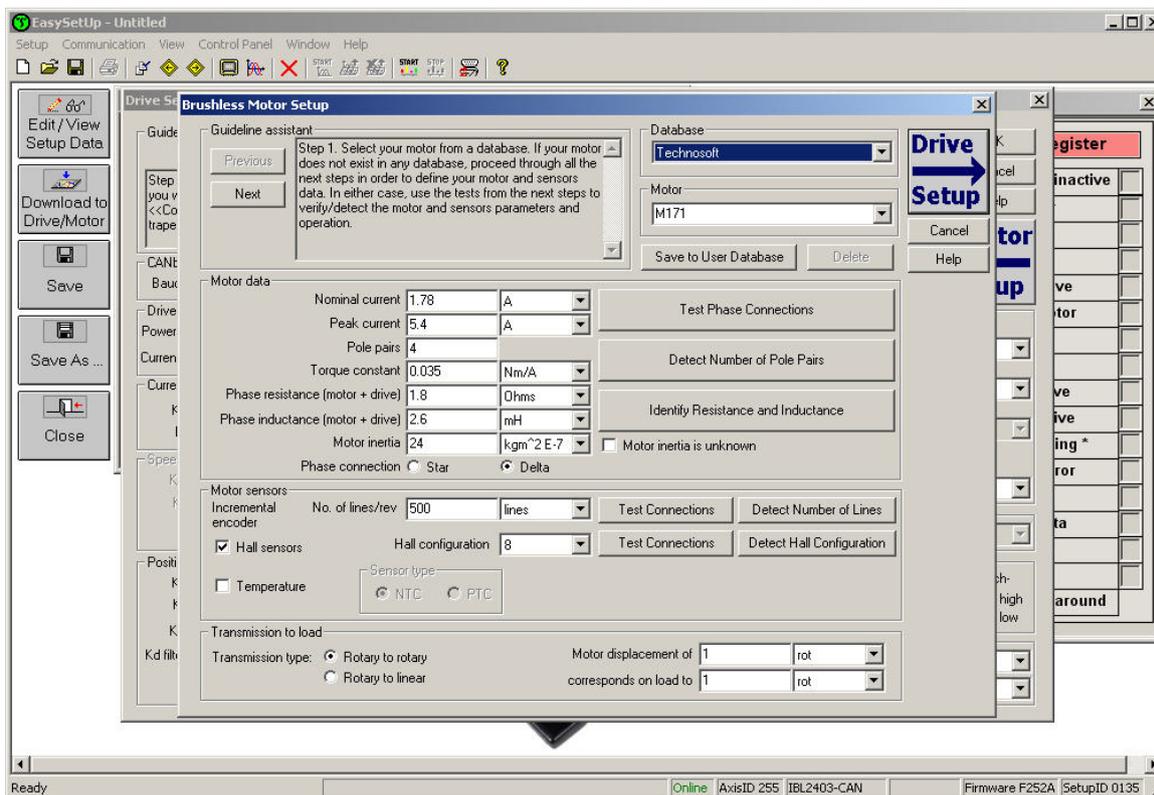
In the **Motor setup** dialogue you can introduce the data of your motor and the associated sensors. Data introduction is accompanied by a series of tests having as goal to check the connections to the drive and/or to determine or validate a part of the motor and sensors parameters. In the **Drive setup** dialogue you can configure and parameterize the drive 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 drive setup.

4.2.3. **Download setup data to drive/motor**

Press the **Download to Drive/Motor** button to download your setup data in the drive/motor EEPROM memory in the *setup table*. From now on, at each power-on, the setup data is copied into the drive/motor RAM memory which is used during runtime. It is also possible to
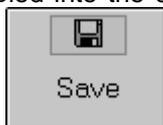
Save the setup data on your PC and use it in other applications.

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 drive dialogues
- use setup data previously saved in the PC
- upload setup data from a drive/motor EEPROM memory

4.2.4. **Evaluate drive/motor behavior (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 drive Axis ID
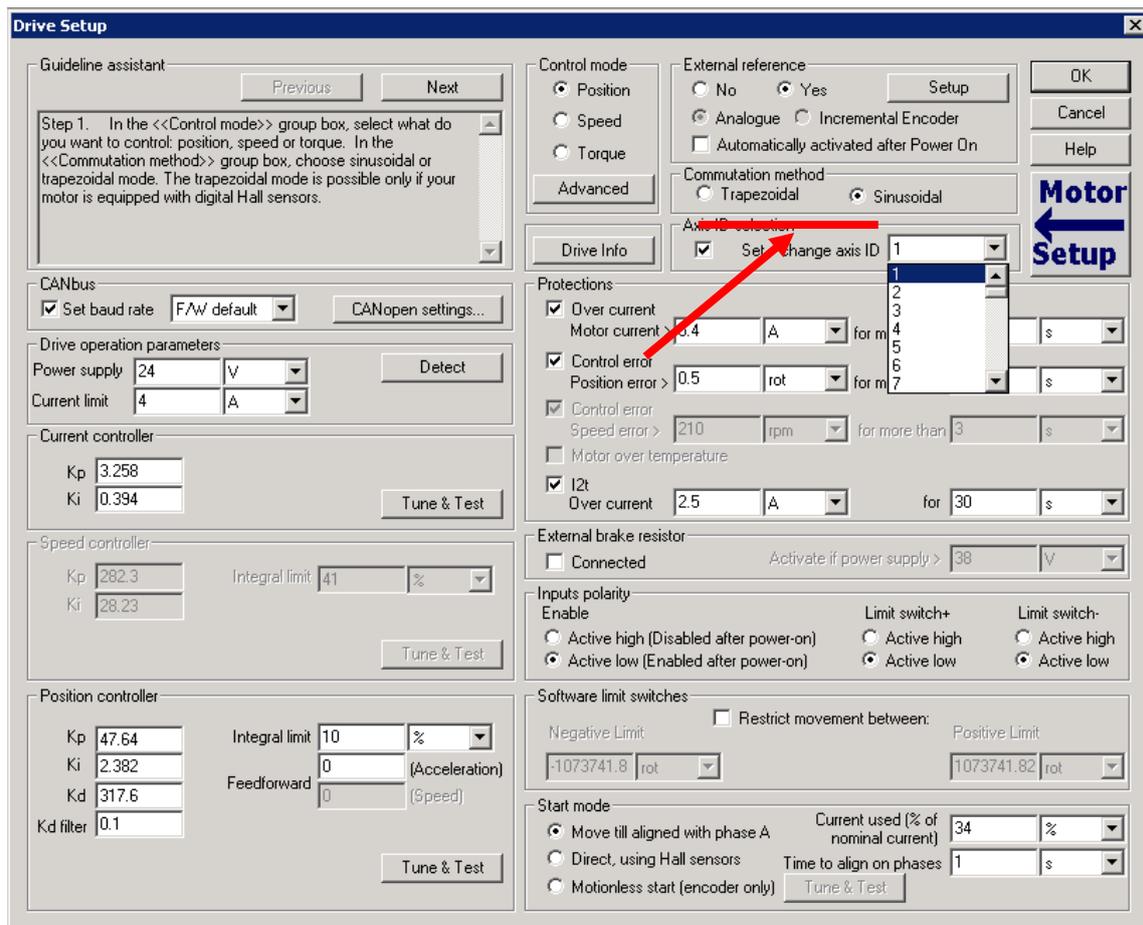
The axis ID of an iPOS4808 drive can be set in 3 ways:

- Hardware (H/W) – according with AxisID inputs levels as described in par. 3.4.5)
- Software (via Setup) – any value between 1 and 255, stored in the setup table. If the drive is in CANopen mode, a Node ID value above 127 is automatically converted into 255 and the drive is set with CAN communication "non-configured" mode waiting for a CANopen master to configure it using CiA-305 protocol. A "non-configured" drive answers only to CiA-305 commands. All other CANopen commands are ignored and transmission of all other CANopen messages (including boot-up) is disabled. In absence of a CANopen master, you can get out a drive from "non-configured" mode, by setting another axis ID between 1 and 127, from above dialogue using a serial link between the drive and the PC.
- Software (via CANopen master) – using CiA-305 protocol

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

a) If a valid setup table exists, and this setup table was created with the *Axis ID Selection* checkbox <u>checked</u> in the Drive Setup dialogue (see above) – with the value read from the setup table. This value can be an axis number 1 to 255 or can indicate that axis ID will be set according with the AxisID inputs levels. If the drive I set in CANopen mode and the Axis ID is over 127 it is converted into 255 and the drive enters in CAN communication "non-configured" mode

b) If a valid the setup table exists, and this was created with the *Axis ID Selection* checkbox <u>unchecked</u> in the Drive Setup dialogue (see above) – with the last value set either from a valid setup table or by a CANopen master via CiA-305 protocol. This value can be an axis number 1 to 255 for TMLCAN, 1 to 127 for CANopen, or can indicate that axis ID will be set according with the AxisID inputs levels

c) If the setup table is invalid, with the last value set either from a valid setup table or by a CANopen master via CiA-305 protocol. This value can be an axis number 1 to 255 for TMLCAN, 1 to 127 for CANopen, or can indicate that axis ID will be set according with the AxisID inputs levels

d) If the setup table is invalid, there is no previous axis ID set from a valid setup table or by a CANopen master, according with the AxisID inputs levels

Remark: If you don't know the axis ID set in a drive, you can find it in the following way:

a) *Connect the drive via a serial RS232 link to a PC where EasySetUp or EasyMotion Studio are installed*

b) *With the drive powered, open EasySetUp or EasyMotion Studio and check the status bar. If communication with the drive is established, the status bar displays* **Online** *in green and nearby the drive's Axis ID. If the status bar displays* **Offline** *in red, execute menu command "Communication|Setup…" and in the dialogue opened select at "Channel Type"* **RS232** *and at "Axis ID of drive/motor connected to PC" the option* **Autodetected***. After closing the dialogue with OK, communication with the drive shall be established and the status bar shall display the drive's Axis ID*

c) *If the access to the drive with the unknown Axis ID is difficult, but this drive is connected via CANbus with other Technosoft drives having an easier access, connect your PC serially to one of the other drives. Use EasySetUp or EasyMotion Studio menu command* **Communication | Scan Network** *to find the axis IDs of all the Technosoft drives present in the network.*

4.4. Setting CANbus rate
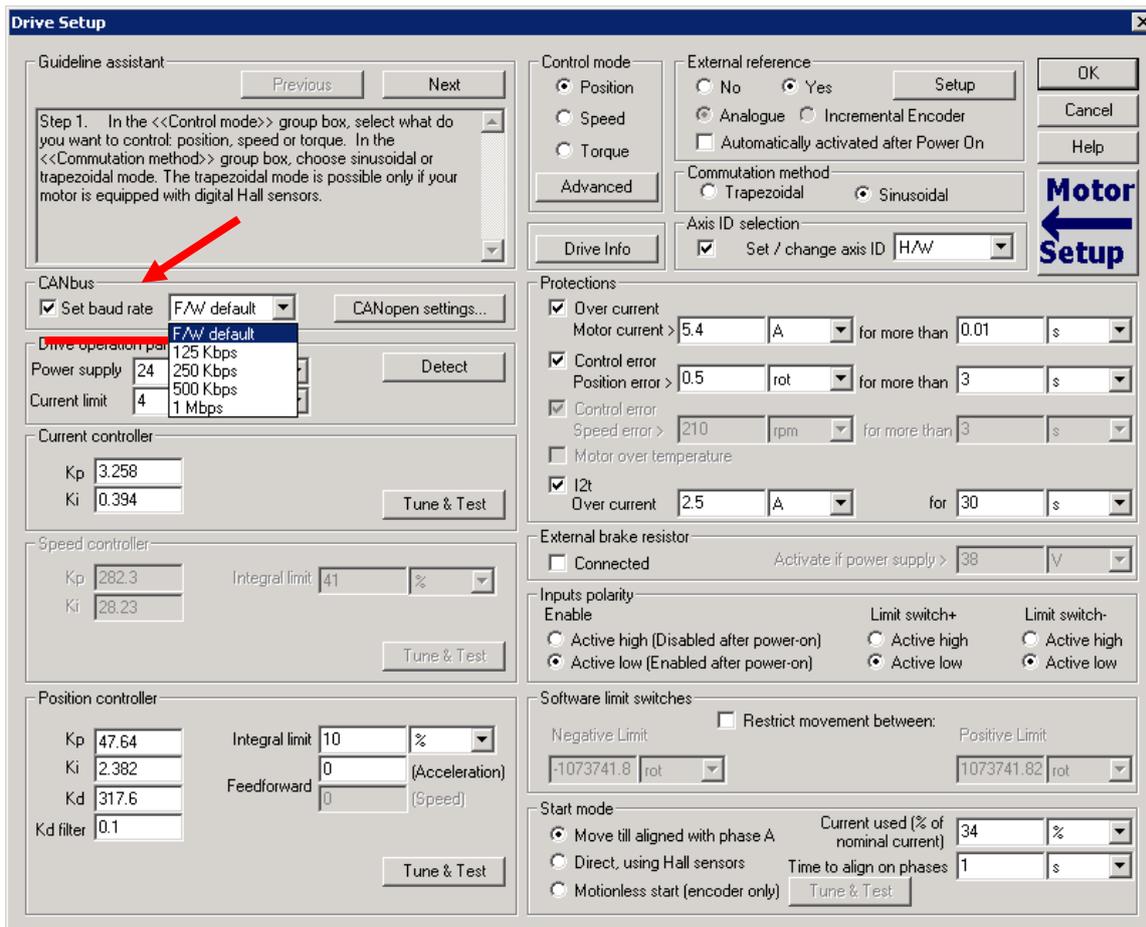
The iPOS4808 drives accept the following CAN rates: 125Kbps, 250 Kbps, 500kbps and 1Mbps. Using the Drive Setup dialogue you can choose the initial CAN rate after power on. This information is stored in the setup table The CAN rate is initialized using the following algorithm:

a) If a valid setup table exists, and this setup table was created with the *Set baud rate* checkbox <u>checked</u> in the Drive Setup dialogue (see above) – with the value read from the setup table. This value can be one of the above 4 values or the firmware default (F/W default) which is 500kbs

b) If a valid setup table exists, and this setup table was created with the *Set baud rate* checkbox <u>unchecked</u> in the Drive Setup dialogue (see above) – with the last value set either from a valid setup table or by a CANopen master via CiA-305 protocol

c) If the setup table is invalid, with the last value set either from a valid setup table or by a CANopen master via CiA-305 protocol.

d) If the setup table is invalid, there is no previous CAN rate set from a valid setup table or by a CANopen master, with f/w default value which is 500kbs

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 drive.

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 drive:

- from a CANopen master, using the communication objects for writing data into the drive EEPROM
- from a host PC or PLC, using the TML_LIB functions for writing data into the drive 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 drives during production.

5. Step 3. Motion Programming

5.1. Using a CANopen Master (for iPOS4808 CANopen execution)

The iPOS4808 drive conforms to **CiA 301 v.4.2** application layer and communication profile, **CiA WD 305 v.2.2.13** layer settings services and protocols and **CiA DSP 402 v3.0** device profile for drives and motion control the now included in IEC 61800-7-1 Annex A, IEC 61800-7-201 and IEC 61800-7-301 standards. For details see ***iPOS CANopen Programming* (part no. P091.063.iPOS.UM.xxxx)**.

5.1.1. CiA-301 Application Layer and Communication Profile Overview

The iPOS4808 drive accepts the following basic services and types of communication objects of the CANopen communication profile CiA301 v4.2:

- **Service Data Object (SDO)**

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

- **Process Data Object (PDO)**

Process Data Objects (PDO) are used for high priority, real-time data transfers between CANopen master and the drives. The PDOs are unconfirmed services which are performed with no protocol overhead. Transmit PDOs are used to send data from the drive, and receive PDOs are used to receive on to the drive. The iPOS4808 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 drive 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 iPOS4808 supports both SYNC consumer and producer.

- **Time Stamp Object (TIME)**

The Time Stamp Object is supported by the iPOS4808 device.

- **Emergency Object (EMCY)**

Emergency objects are triggered by the occurrence of a drive internal error situation. An emergency object is transmitted only once per 'error event'. As long as no new errors occur, the drive 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 drive can be initialized, started, monitored, reset or stopped. The iPOS4808 is a NMT slave in a CANopen network.

- **Module Control Services** – through these unconfirmed services, the NMT master controls the state of the drive. 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 iPOS4808: Node Guarding (including Life Guarding) and Heartbeat
- **Bootup Service** - through this service, the drive indicates that it has been properly initialized and is ready to receive commands from a master

5.1.2. CiA-402 and Manufacturer Specific Device Profile Overview

The iPOS4808 supports the following CiA 402 modes of operation:

- **Profile position and velocity modes**
- **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 and camming position mode**

5.1.3. TechnoCAN Extension

In order to take full advantage of the powerful Technosoft Motion Language (TML) built into the iPOS4808, Technosoft has developed an extension to CANopen, called TechnoCAN through which TML commands can be exchanged with the drives. Thanks to TechnoCAN you can inspect or reprogram any of the Technosoft drives from a CANopen network using EastSetUp or EasyMotion Studio and an RS-232 link between your PC and anyone of the drives.

TechnoCAN uses only identifiers outside of the range used by the default by the CANopen predefined connection set (as defined by CiA 301). Thus, TechnoCAN protocol and CANopen protocol can co-exist and communicate simultaneously on the same physical CAN bus, without disturbing each other.

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 0 and 5.2.4 for details) whether the non-volatile EEPROM memory of an iPOS4808 drive contains the right information. If the checksum reported by the drive doesn't match with that computed from the **.sw** file, the CANopen master can download the entire **.sw** file into the drive EEPROM using the communication objects for writing data into the drive EEPROM.

5.2. Using the built-in Motion Controller and TML

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

5.2.1. Technosoft Motion Language Overview

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

- Set various motion modes (profiles, PVT, PT, electronic gearing or camming[1], etc.)
- Change the motion modes and/or the motion parameters
- Execute homing sequences[2]
- Control the program flow through:

---

[1] Optional for iPOS480x CANopen execution
[2] The customization of the homing routines is available only for iPOS480x CAN execution

- 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 drives. 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.

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 drive according with your application needs. The drive can be connected with your PC in one of the following ways:

1. Via an RS232 link, directly connected to the PC, or via an USB to RS232 adapter or using Technosoft Ethernet to RS232 adapter, function of your PC communication interfaces;
2. Via a CAN-bus link, directly connected to the PC through a PC-CAN interface, or using Technosoft Ethernet to CAN adapter
3. Via another drive from the same CAN-bus network, which is connected to the PC via one of the above options from point 1.

The output of the EasyMotion Studio is a set of setup data and a motion program, which can be downloaded to the drive/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 drives/motors from the network. All data exchanges are done with the drive/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 drive. This information is preserved in the drive/motor EEPROM in the *setup table*. The setup table is copied at power-on into the RAM memory of the drive/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 drives/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.
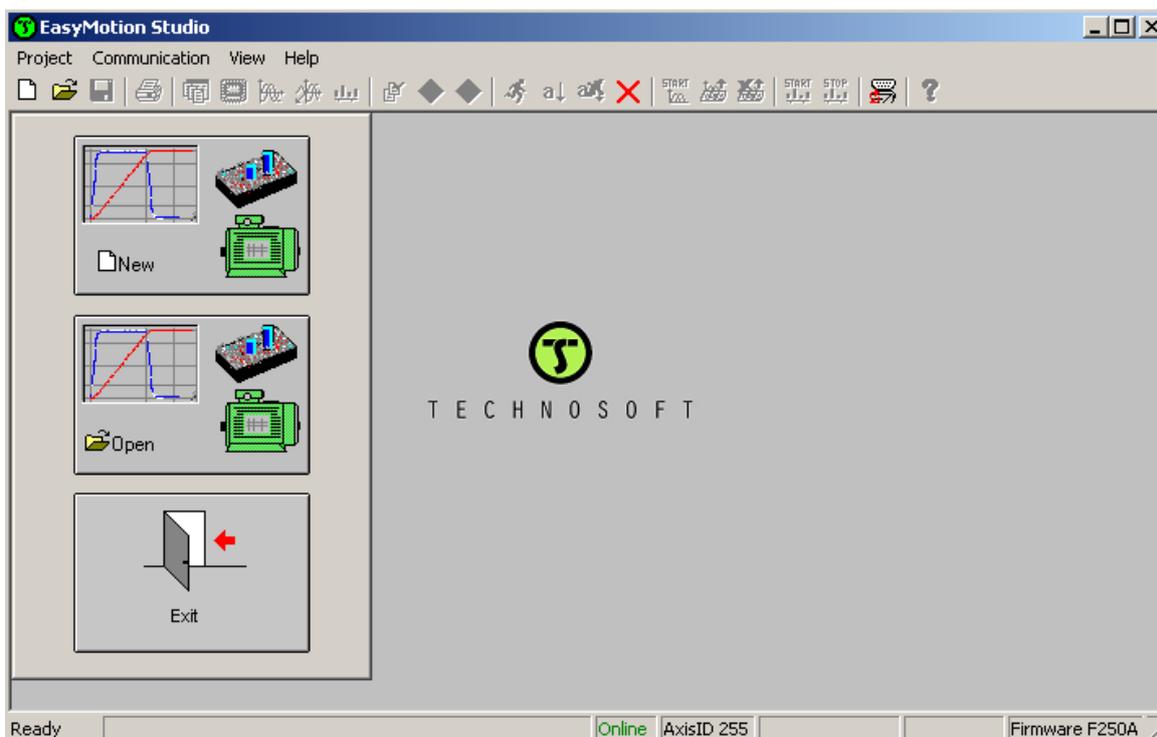

When you start a new project, EasyMotion Studio automatically creates a first application. Additional applications can be added later. You can duplicate an application or insert one defined in another project.
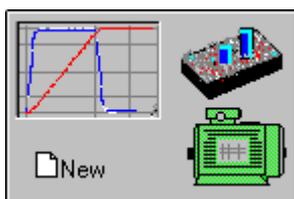
Press **New** button to open the "New Project" dialogue. Set the axis number for your first application equal with your drive/motor axis ID. The initial value proposed is 255 which is the default axis ID of the drives. Press **New** button and select your drive type. Depending on the

product chosen, the selection may continue with the motor technology (for example: brushless or brushed) and the type of feedback device (for example: incremental encoder).
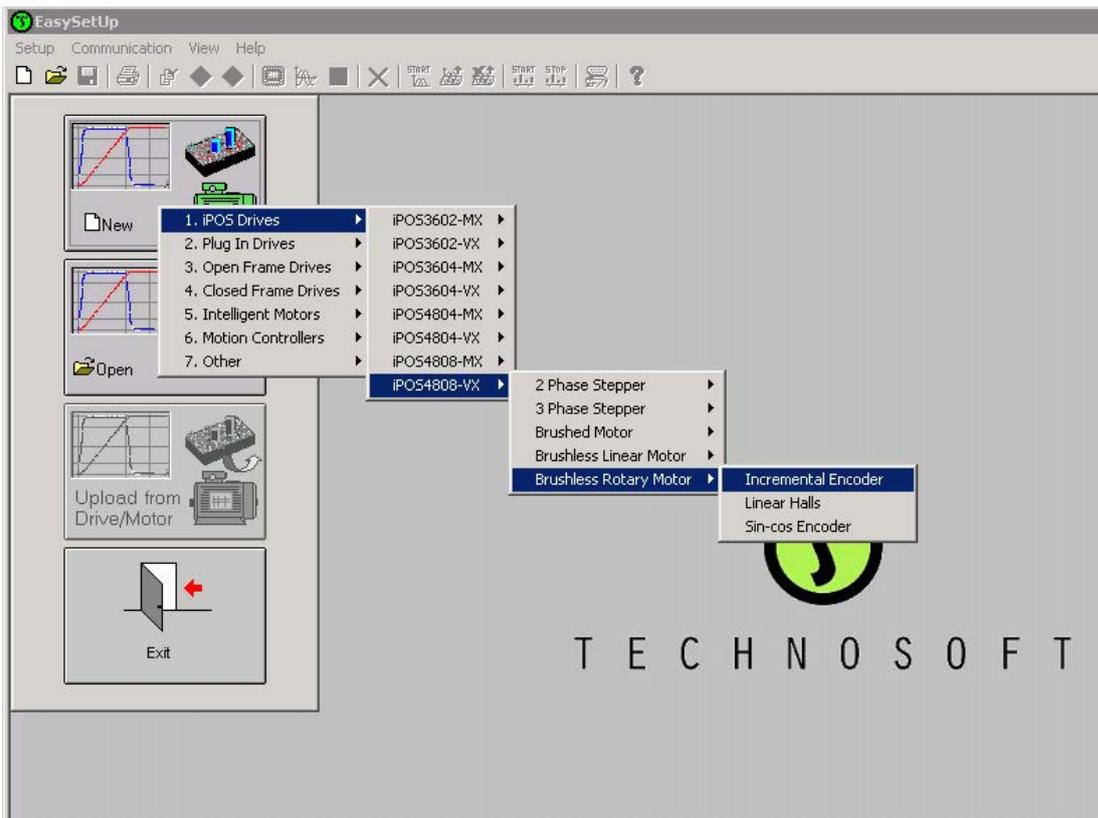
Click on your selection. EasyMotion Studio opens the Project window where on the left side you can see the structure of a project. At beginning both the new project and its first application are named "Untitled". The application has 2 components: **S** Setup and **M** Motion (program).
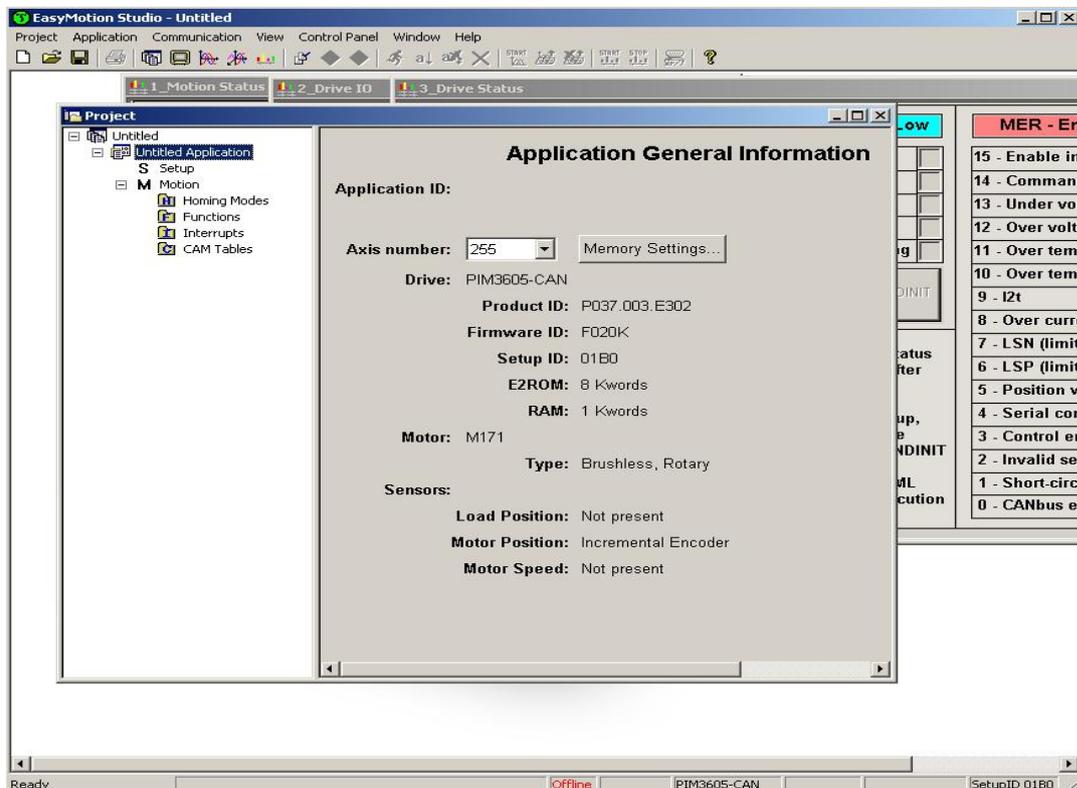
5.2.3.2   Step 2 Establish communication

If you have a drive/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 drive/motor and the connections. Power on the drive, 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 drive/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 drive having axis ID=255 (default communication settings).).   If the drive has a different axis ID and you don't know it, select in the Communication | Setup dialogue at "Axis ID of drive/motor connected to PC" the option **Autodetected.** If this drive is part of a CANbus network and the PC is serially connected with another drive, use the menu command **Communication | Scan Network***

5.2.3.3   Setup drive/motor

In the project window left side, select "**S** Setup", to access the setup data for your application.
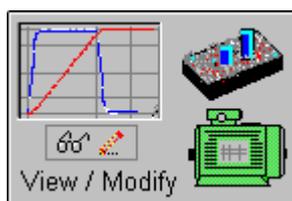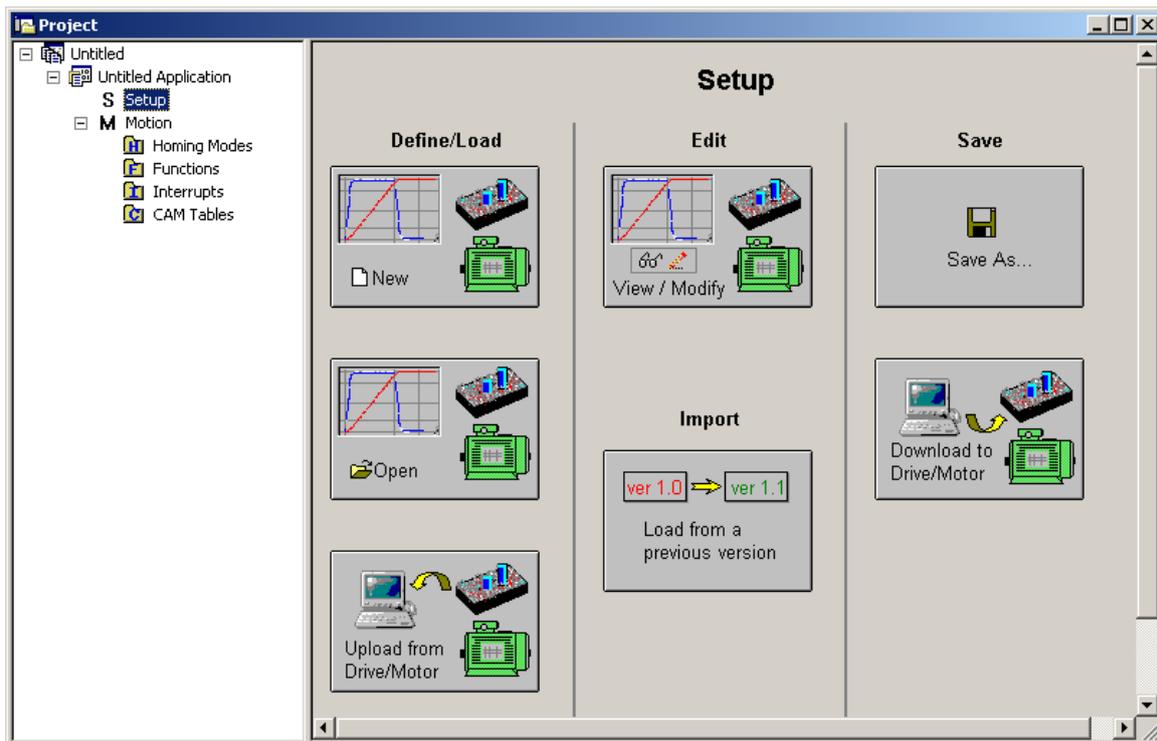
Press **View/Modify** button ![View / Modify]. This opens 2 setup dialogues: for **Motor Setup** and for **Drive Setup** (same like on EasySetUp) through which you can configure and parameterize a Technosoft drive. In the **Motor setup** dialogue you can introduce the data of your motor and the associated sensors. Data introduction is accompanied by a series of tests having as

goal to check the connections to the drive 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 drive 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 drive/motor EEPROM memory in the *setup table*. From now on, at each power-on, the setup data is copied into the drive/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 drive/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 drive dialogues

- use setup data previously saved in the PC

- upload setup data from a drive/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**.
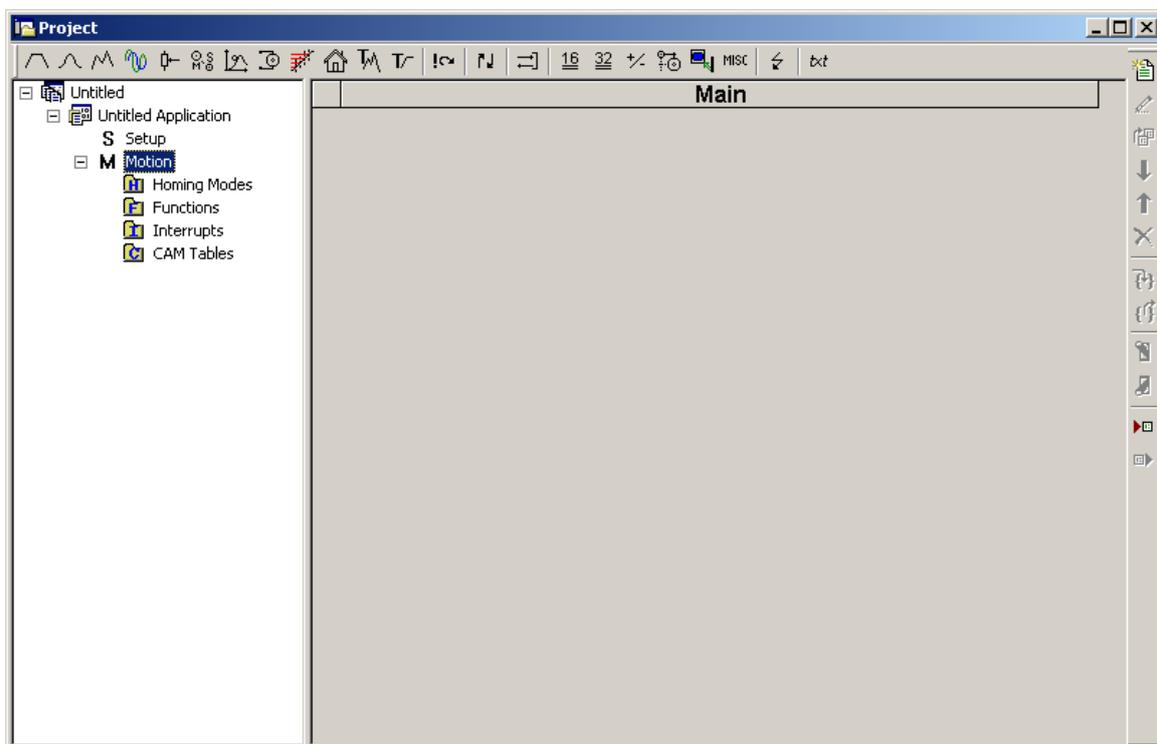
The Motion Wizard offers you the possibility to program all the motion sequences using high level graphical dialogues which automatically generate the corresponding TML instructions. Therefore with Motion Wizard you can develop motion programs using almost all the TML instructions without needing to learn them. A TML program includes a main section, followed by the subroutines used:

functions, interrupt service routines and homing procedures[1]. The TML program may also include cam tables used for electronic camming applications[2].

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 drive/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 drives 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 drive. 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 drive, see paragraph 0

5.3. Combining CANopen /or other host with TML

Due to its embedded motion controller, an iPOS4808 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 drive 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

---

[1] The customization of the interrupt service routines and homing routines is available only for iPOS480x CAN execution
[2] Optional for iPOS480x CANopen execution

Remark: If you don't use the advanced features presented below you don't need EasyMotion Studio. In this case the iPOS4808 is treated like a standard CANopen drive, whose setup is done using EasySetUp.

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

With Technosoft intelligent drives you can really distribute the intelligence between a CANopen master and the drives in complex multi-axis applications. Instead of trying to command each step of an axis movement, you can program the drives 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 drives 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 drive a complete TML program including functions, homing procedures[1], 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 iPOS4808 offers others motion modes like[2]: 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 drive active RAM memory after each power on;
   b) The cam points are stored in the drive EEPROM and the master commands their copy into the active RAM memory
   c) The cam points are stored in the drive EEPROM and during the drive 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. **Remark:** The cam tables are included in the **.sw** file generated with EasyMotion Studio. Therefore, the drives can check the cam presence in the drive EEPROM using the same procedure as for testing of the setup data.
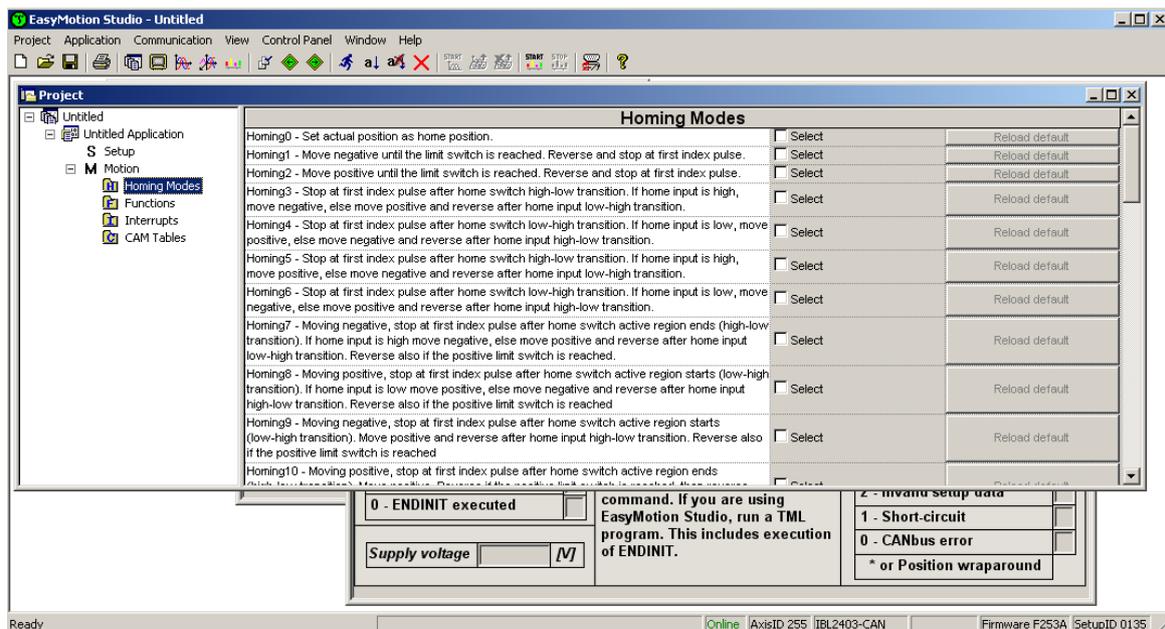
5.3.4. Customizing the Homing Procedures

The iPOS4808 supports all homing modes defined in CiA402 device profile, plus 4 custom based on hard stop. 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 sub tree, with the name *HomeX* where X is the number of the selected homing.

---

[1] The customization of the interrupt service routines and homing routines is available only for iPOS480x CAN executions
[2] Optional for the iPOS480x CANopen execution

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

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 drive 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 drive. 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 drives. 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 drive / motor via any of its supported channels (RS-232, CAN-bus, etc.), send motion commands, get automatically or on request information about drive / 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 drives. 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 drive/motor via any of its supported channels, send motion commands, get automatically or on request information about drive/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.

All these blocks have been designed using the guidelines described in the PLC standards, so they can be used on any development platform that is **IEC 61136 compliant.**

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

6. Scaling factors

Technosoft drives work with parameters and variables represented in the drive 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 drive 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

6.1.1. Brushless / DC brushed motor with quadrature encoder on motor

The internal position units are encoder counts. The correspondence with the load **position in SI units**[1] is:

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

6.1.2. Brushless motor with linear Hall signals

The internal position units are counts. The motor is rotary. The resolution i.e. number of counts per revolution is programmable as a power of 2 between 512 and 8192. By default it is set at 2048 counts per turn. The correspondence with the load position in SI units is:

---

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

For rotary motors:
$$Load\_Position[SI] = \frac{2 \times \pi}{resolution \times Tr} \times Motor\_Position[IU]$$

For linear motors:
$$Load\_Position[SI] = \frac{Pole\_Pitch}{Tr} \times Motor\_Position[IU]$$

where:

resolution – is the motor position resolution

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

Pole_Pitch – is the magnetic pole pitch NN (distance expressed in [m])

6.1.3. DC brushed motor with quadrature encoder on load and tacho on motor

The internal position units are encoder counts. The motor is rotary and the transmission is rotary-to-rotary. The correspondence with the load position in SI units is:

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

where:

No_encoder_lines – is the encoder number of lines per revolution

6.1.4. Step motor open-loop control. No feedback device

The internal position units are motor µsteps. The correspondence with the load **position in SI units** is:

$$Load\_Position[SI] = \frac{2 \times \pi}{No\_\mu steps \times No\_steps \times Tr} \times Motor\_Position[IU]$$

where:

No_steps – is the number of motor steps per revolution

No_µsteps – is the number of microsteps per step. You can read/change this value in the "Drive Setup" dialogue from EasySetUp.

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

Step motor closed-loop control. Incremental encoder on motor

The internal position units are motor encoder counts. The correspondence with the load **position in SI units** [1] is:

---

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

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

where:

No_encoder_lines – is the motor encoder number of lines per revolution

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

6.1.5. **Step motor open-loop control. Incremental encoder on load**

The internal position units are load encoder counts. The transmission is rotary-to-rotary. The correspondence with the load position in SI units is:

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

6.1.6. **Brushless motor with sine/cosine encoder on motor**

The internal position units are interpolated encoder counts. The correspondence with the load position in SI units is:

For rotary motors:

$$Load\_Position[SI] = \frac{2 \times \pi}{4 \times Enc\_periods \times Interpolation \times Tr} \times Motor\_Position[IU]$$

For linear motors:

$$Load\_Position[SI] = \frac{Encoder\_accuracy}{Interpolation \times Tr} \times Motor\_Position[IU]$$

where:
        Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution
        Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation
        Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period
        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 internal position units / (slow loop sampling period) i.e. the position variation over one slow loop sampling period

6.2.1. **Brushless / DC brushed motor with quadrature encoder on motor**

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

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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"

6.2.2. Brushless motor with linear Hall signals

The internal speed units are counts / (slow loop sampling period). The motor is rotary. The position resolution i.e. number of counts per revolution is programmable as a power of 2 between 512 and 8192. By default it is set at 2048 counts per turn. The correspondence with the load speed in SI units is:

For rotary motors:
$$Load\_Speed[SI] = \frac{2 \times \pi}{resolution \times Tr \times T} \times Motor\_Speed[IU]$$

For linear motors:
$$Load\_Speed[SI] = \frac{Pole\_Pitch}{resolution \times Tr \times T} \times Motor\_Speed[IU]$$

where:

resolution – is the motor position resolution

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"

Pole_Pitch – is the magnetic pole pitch NN (distance expressed in [m])

6.2.3. DC brushed motor with quadrature encoder on load and tacho on motor

The internal speed units are encoder counts / (slow loop sampling period). The motor is rotary and the transmission is rotary-to-rotary. The correspondence with the load speed in SI units is:

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

where:

No_encoder_lines – is the encoder number of lines per revolution

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"

6.2.4. **DC brushed motor with tacho on motor**

When only a tachometer is mounted on the motor shaft, the internal speed units are A/D converter bits. The correspondence with the load **speed in SI units**[1] is:

$$Load\_Speed[SI] = \frac{Analogue\_Input\_Range}{4096 \times Tacho\_gain \times Tr} \times Motor\_Speed[IU]$$

where:

Analogue_Input_Range – is the range of the drive analogue input for feedback, expressed in [V]. You can read this value in the "Drive Info" dialogue, which can be opened from the "Drive Setup"

Tacho_gain – is the tachometer gain expressed in [V/rad/s]

6.2.5. **Step motor open-loop control. No feedback device**

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

$$Load\_Speed[SI] = \frac{2 \times \pi}{No\_\mu steps \times No\_steps \times Tr \times T} \times Motor\_Speed[IU]$$

where:

No_steps – is the number of motor steps per revolution

No_µsteps – is the number of microsteps per step. You can read/change this value in the "Drive Setup" dialogue from EasySetUp.

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"

Step motor open-loop control. Incremental encoder on load

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

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

Tr – transmission ratio between the motor displacement in [rad] and load displacement in [rad] or [m]

---

[1] SI units for speed are [rad/s] for a rotary movement, [m/s] 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".

6.2.6. Step motor closed-loop control. Incremental encoder on motor

The internal speed units are motor encoder counts / (slow loop sampling period). The correspondence with the load **speed in SI units**[1] is:

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

where:

No_encoder_lines – is the motor encoder number of lines per revolution

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".

6.2.7. Brushless motor with sine/cosine encoder on motor

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

For rotary motors:

$$Load\_Speed[SI] = \frac{2 \times \pi}{4 \times Enc\_periods \times Interpolation \times Tr \times T} \times Motor\_Speed[IU]$$

For linear motors:

$$Load\_Speed[SI] = \frac{Encoder\_accuracy}{Interpolation \times Tr \times T} \times Motor\_Speed[IU]$$

where:
        Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution
        Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period
        Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation
        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"

6.3. Acceleration units

The internal acceleration units are internal position units / (slow loop sampling period)[2] i.e. the speed variation over one slow loop sampling period.

6.3.1. Brushless / DC brushed motor with quadrature encoder on motor

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

---

[1] SI units for speed are [rad/s] for a rotary movement , [m/s] for a linear movement

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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"

6.3.2. Brushless motor with linear Hall signals

The internal acceleration units are counts / (slow loop sampling period)[2]. The motor is rotary. The position resolution i.e. number of counts per revolution is programmable as a power of 2 between 512 and 8192. By default it is set at 2048 counts per turn. The correspondence with the load **acceleration in SI units**[1] is:

For rotary motors:

$$Load\_Acceleration[SI] = \frac{2 \times \pi}{resolution \times Tr \times T^2} \times Motor\_Acceleration[IU]$$

For linear motors:

$$Load\_Acceleration[SI] = \frac{Pole\_Pitch}{resolution \times Tr \times T^2} \times Motor\_Acceleration[IU]$$

where:

resolution – is the motor position resolution

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"

Pole_Pitch – is the magnetic pole pitch NN (distance expressed in [m])

6.3.3. DC brushed motor with quadrature encoder on load and tacho on motor

The internal acceleration units are encoder counts / (slow loop sampling period)[2]. The motor is rotary and the transmission is rotary-to-rotary. The correspondence with the load acceleration in SI units is:

---

[1] SI units for acceleration are [rad/s^2] for a rotary movement, [m/s^2] for a linear movement

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

where:

No_encoder_lines – is the encoder number of lines per revolution

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"

6.3.4. Step motor open-loop control. No feedback device

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

$$Load\_Acceleration[SI] = \frac{2 \times \pi}{No\_\mu steps \times No\_steps \times Tr \times T^2} \times Motor\_Acceleration[IU]$$

where:

No_steps – is the number of motor steps per revolution

No_µsteps – is the number of microsteps per step. You can read/change this value in the "Drive Setup" dialogue from EasySetUp.

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"

6.3.5. Step motor open-loop control. Incremental encoder on load

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

      For rotary-to-rotary transmission:

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

      For rotary-to-linear transmission:

$$Load\_Acceleration[m/s^2] = \frac{Encoder\_accuracy}{T^2} \times Load\_Acceleration[IU]$$

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".

6.3.6. Step motor closed-loop control. Incremental encoder on motor

The internal acceleration units are motor encoder counts / (slow loop sampling period)^2. The transmission is rotary-to-rotary. The correspondence with the load **acceleration in SI units** is:

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

where:

No_encoder_lines – is the motor encoder number of lines per revolution

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"

6.3.7. Brushless motor with sine/cosine encoder on motor

The internal acceleration units are interpolated encoder counts / (slow loop sampling period)^2. The correspondence with the load **acceleration in SI units**[1] is:

For rotary motors:

$$Load\_Acceleration[SI] = \frac{2 \times \pi}{4 \times Enc\_periods \times Interpolation \times Tr \times T^2} \times Motor\_Acceleration[IU]$$

For linear motors:

$$Load\_Acceleration[SI] = \frac{Encoder\_accuracy}{Interpolation \times Tr \times T^2} \times Motor\_Acceleration[IU]$$

where:
    Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution
    Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period
    Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation
    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 internal position units / (slow loop sampling period)^3 i.e. the acceleration variation over one slow loop sampling period.

6.4.1. Brushless / DC brushed motor with quadrature encoder on motor

The internal jerk units are encoder counts / (slow loop sampling period)^3. The correspondence with the load **jerk in SI units**[2] is:

---

[1] SI units for acceleration are [rad/s^2] for a rotary movement, [m/s^2] for a linear movement
[2] SI units for jerk are [rad/s^3] for a rotary movement, [m/s^3] for a linear movement

---

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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"

6.4.2. Brushless motor with linear Hall signals

The internal jerk units are counts / (slow loop sampling period)[3]. The motor is rotary. The position resolution i.e. number of counts per revolution is programmable as a power of 2 between 512 and 8192. By default it is set at 2048 counts per turn. The correspondence with the load acceleration in SI units is:

For rotary motors: $Load\_Jerk[SI] = \dfrac{2 \times \pi}{resolution \times Tr \times T^3} \times Motor\_Jerk[IU]$

For linear motors: $Load\_Jerk[SI] = \dfrac{Pole\_Pitch}{resolution \times Tr \times T^3} \times Motor\_Jerk[IU]$

where:

resolution – is the motor position resolution

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"

Pole_Pitch – is the magnetic pole pitch NN (distance expressed in [m])

6.4.3. DC brushed motor with quadrature encoder on load and tacho on motor

The internal jerk units are encoder counts / (slow loop sampling period)[3]. The motor is rotary and the transmission is rotary-to-rotary. The correspondence with the load jerk in SI units is:

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

where:

No_encoder_lines – is the encoder number of lines per revolution

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"

6.4.4. Step motor open-loop control. No feedback device

The internal jerk units are motor µsteps / (slow loop sampling period)[3]. The correspondence with the load **jerk in SI units**[1] is:

$$Load\_Jerk[SI] = \frac{2 \times \pi}{No\_\mu steps \times No\_steps \times Tr \times T^3} \times Motor\_Jerk[IU]$$

where:

No_steps – is the number of motor steps per revolution

No_µsteps – is the number of microsteps per step. You can read/change this value in the "Drive Setup" dialogue from EasySetUp.

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"

6.4.5. **Step motor open-loop control. Incremental encoder on load**

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

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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".

6.4.6. **Step motor closed-loop control. Incremental encoder on motor**

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

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

where:

No_encoder_lines – is the motor encoder number of lines per revolution

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".

6.4.7. **Brushless motor with sine/cosine encoder on motor**

---

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

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

For rotary motors:

$$Load\_Jerk[SI] = \frac{2 \times \pi}{4 \times Enc\_periods \times Interpolation \times Tr \times T^3} \times Motor\_Jerk[IU]$$

For linear motors:

$$Load\_Jerk[SI] = \frac{Encoder\_accuracy}{Interpolation \times Tr \times T^3} \times Motor\_Jerk[IU]$$

where:

Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution
Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period
Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation
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:

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

where   Ipeak – is the drive 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** driven 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:

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

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

In case of **brushless** motors driven 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:

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

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

6.7. Voltage measurement units

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

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

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:

$$Time[s] = T \times Time[IU]$$

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. Master position units

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

6.10. 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.

6.11. Motor position units

6.11.1. Brushless / DC brushed motor with quadrature encoder on motor

The internal motor position units are encoder counts. The correspondence with the motor **position in SI units**[1] is:

---

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

---

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

6.11.2. Brushless motor with linear Hall signals

The internal motor position units are counts. The motor is rotary. The resolution i.e. number of counts per revolution is programmable as a power of 2 between 512 and 8192. By default it is set at 2048 counts per turn. The correspondence with the motor position in SI units is:

For rotary motors:
$$Motor\_Position[SI] = \frac{2 \times \pi}{resolution} \times Motor\_Position[IU]$$

For linear motors:
$$Motor\_Position[SI] = \frac{Pole\_Pitch}{resolution} \times Motor\_Position[IU]$$

where:

resolution – is the motor position resolution

Pole_Pitch – is the magnetic pole pitch NN (distance expressed in [m])

6.11.3. DC brushed motor with quadrature encoder on load and tacho on motor

The motor position is not computed.

6.11.4. Step motor open-loop control. No feedback device

The internal motor position units are motor µsteps. The correspondence with the motor **position in SI units**[1] is:

$$Motor\_Position[SI] = \frac{2 \times \pi}{No\_\mu steps \times No\_steps} \times Motor\_Position[IU]$$

where:

No_steps – is the number of motor steps per revolution

No_µsteps – is the number of microsteps per step. You can read/change this value in the "Drive Setup" dialogue from EasySetUp.

6.11.5. Step motor open-loop control. Incremental encoder on load

In open-loop control configurations with incremental encoder on load, the motor position is not computed.

6.11.6. Step motor closed-loop control. Incremental encoder on motor

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

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

where:

No_encoder_lines – is the motor encoder number of lines per revolution

6.11.7. Brushless motor with sine/cosine encoder on motor

The internal motor position units are interpolated encoder counts. The correspondence with the motor position in SI units is:

For rotary motors:

$$Motor\_Position[SI] = \frac{2 \times \pi}{4 \times Enc\_periods \times Interpolation} \times Motor\_Position[IU]$$

For linear motors:

$$Motor\_Position[SI] = \frac{Encoder\_accuracy}{Interpolation} \times Motor\_Position[IU]$$

where:

Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution
Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation
Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period

6.12. Motor speed units

6.12.1. Brushless / DC brushed motor with quadrature encoder on motor

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

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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"

6.12.2. Brushless motor with linear Hall signals

The internal motor speed units are counts / (slow loop sampling period). The motor is rotary. The position resolution i.e. number of counts per revolution is programmable as a power of 2 between 512 and 8192. By default it is set at 2048 counts per turn. The correspondence with the motor speed in SI units is:

For rotary motors:
$$Motor\_Speed[SI] = \frac{2 \times \pi}{resolution \times T} \times Motor\_Speed[IU]$$

For linear motors:
$$Motor\_Speed[SI] = \frac{Pole\_Pitch}{resolution \times T} \times Motor\_Speed[IU]$$

where:

resolution – is the motor position resolution

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"

Pole_Pitch – is the magnetic pole pitch NN (distance expressed in [m])

6.12.3. DC brushed motor with quadrature encoder on load and tacho on motor

The internal motor speed units are A/D converter bits. The correspondence with the motor **speed in SI units**[1] is:

$$Motor\_Speed[SI] = \frac{Analogue\_Input\_Range}{4096 \times Tacho\_gain} \times Motor\_Speed[IU]$$

where:

Analogue_Input_Range – is the range of the drive analogue input for feedback, expressed in [V]. You can read this value in the "Drive Info" dialogue, which can be opened from the "Drive Setup"

Tacho_gain – is the tachometer gain expressed in [V/rad/s]

6.12.4. DC brushed motor with tacho on motor

The internal motor speed units are A/D converter bits. The correspondence with the motor speed in SI units is:

$$Motor\_Speed[SI] = \frac{Analogue\_Input\_Range}{4096 \times Tacho\_gain} \times Motor\_Speed[IU]$$

where:

Analogue_Input_Range – is the range of the drive analogue input for feedback, expressed in [V]. You can read this value in the "Drive Info" dialogue, which can be opened from the "Drive Setup"

Tacho_gain – is the tachometer gain expressed in [V/rad/s]

6.12.5. Step motor open-loop control. No feedback device or incremental encoder on load

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

---

[1] SI units for motor speed are [rad/s] for a rotary motor, [m/s] for a linear motor

$$Motor\_Speed[SI] = \frac{2 \times \pi}{No\_\mu steps \times No\_steps \times T} \times Motor\_Speed[IU]$$

where:

No_steps – is the number of motor steps per revolution

No_µsteps – is the number of microsteps per step. You can read/change this value in the "Drive Setup" dialogue from EasySetUp.

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"

6.12.6. Step motor closed-loop control. Incremental encoder on motor

The internal motor speed units are motor encoder counts / (slow loop sampling period). The correspondence with the load 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 motor encoder number of lines per revolution
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".

6.12.7. Brushless motor with sine/cosine encoder on motor

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

For rotary motors:

$$Motor\_Speed[SI] = \frac{2 \times \pi}{4 \times Enc\_periods \times Interpolation \times T} \times Motor\_Speed[IU]$$

For linear motors:

$$Motor\_Speed[SI] = \frac{Encoder\_accuracy}{Interpolation \times T} \times Motor\_Speed[IU]$$

where:
 Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution
 Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period
 Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation
 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"
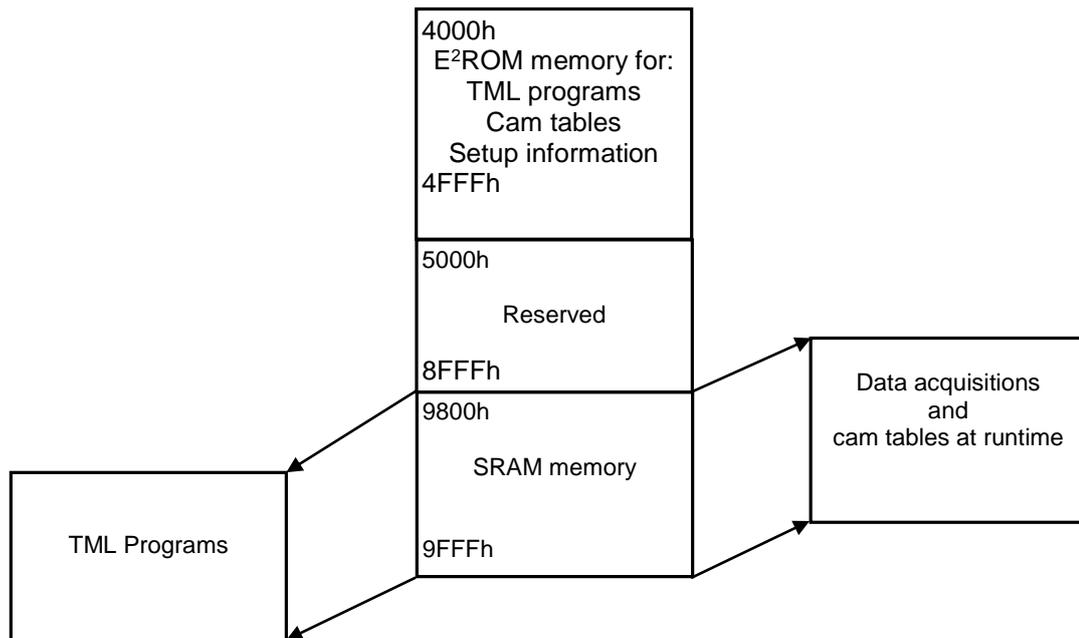
7. Memory Map

iPOS4808 VX has 2 types of memory available for user applications: 2K×16 SRAM and 4K×16 serial E²ROM.

The SRAM memory is mapped in the address range: 9800h to 9FFFh. It can be used to download and run a TML program, to save real-time data acquisitions and to keep the cam tables during run-time.

The E²ROM is mapped in the address range: 4000h to 4FFFh. It is used to keep in a non-volatile memory the TML programs, the cam tables and the drive setup information.

Remark: *EasyMotion Studio handles automatically the memory allocation for each motion application. The memory map can be accessed and modified from the main folder of each application*

Figure 7.1. *iPOS4808 VX Memory Map*