IBL2401-RS232
IBL2401-CAN

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

T E C H N O S O F T

Intelligent Servo Drive

Technical Reference

TECHNOSOFT

IBL2401-RS232
IBL2401-CAN

Technical Reference

P091.035.IBL2401.UM.0710

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

Read This First

About This Manual

This book is a technical reference manual for the **IBL2401** family of intelligent servo drives, including the following products:

IBL2401-RS232 (p/n P035.001.E001) - Universal Drive for Brushless, DC and step motors.
IBL2401-CAN (p/n P035.001.E002) **-** Universal Drive for Brushless, DC and step motors. Standard execution using Technosoft TMLCAN protocol on CANbus
IBL2401-CAN, CANopen (BL) (p/n P035.001.E012) **-** Servo Drive for Brushless and DC motors using CANopen protocol on CANbus
IBL2401-CAN, CANopen (ST) (p/n P035.001.E013) - Stepper Drive using CANopen protocol on CANbus
IBL2401-CAN, CANopen (LH) (p/n P035.001.E014) - Servo Drive for Brushless motors with Linear Hall sensors using CANopen protocol on CANbus

In order to operate the **IBL2401** 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** (for the IBL2401 CANopen version)
  - ❑ The drive **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 **IBL2401** 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:

- **TML** – Technosoft Motion Language
- **SI units** – International standard units (meter for length, seconds for time, etc.)
- **IU units** – Internal units of the drive
- **IBL2401** – all products described in this manual
- **IBL2401 CANopen –** the CANopen executions from the IBL2401 family
- **IBL2401 CAN –** the CAN standard executions

Related Documentation

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

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

Help of the EasySetUp software – describes how to use **EasySetUp** to quickly setup any Technosoft drive for your application using only 2 dialogues. 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 includes a firmware programmer with allows you to update your drive firmware to the latest revision. **EasySetUp can be downloaded free of charge from Technosoft web page**

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

Help of the EasyMotion Studio software – describes how to use the EasyMotion Studio to create motion programs using in Technosoft Motion Language (TML). EasyMotion Studio platform includes **EasySetUp** for the 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 a PLC Omron series CJ1 a motion application for the Technosoft intelligent drives using TML_LIB_CJ1 motion control library for PCs. The TML_LIB_CJ1 library is **IEC61131-3 compatible**.

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

If you Need Assistance …

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

Contents

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, setup, operate and maintain the drive. A “qualified person” has the knowledge and authorization to perform tasks such as transporting, assembling, installing, commissioning and operating drives.

The following safety symbols are used in this manual:

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

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

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

1.1. Warnings

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

WARNING!	***TO AVOID ELECTRIC ARCING AND HAZARDS, NEVER CONNECT / DISCONNECT WIRES FROM THE DRIVE 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 DRIVE CONTAINS ELECTROSTATICALLY SENSITIVE COMPONENTS WHICH MAY BE DAMAGED BY INCORRECT HANDLING. THEREFORE THE DRIVE SHALL BE REMOVED FROM ITS ORIGINAL PACKAGE ONLY IN AN ESD PROTECTED ENVIRONMENT***

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

2. Product Overview

2.1. Introduction

The **IBL2401** 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 IBL2401 drives accept as position feedback incremental encoders (quadrature) and linear Halls signals.

All drives perform position, speed or torque control and work in either single-, multi-axis or stand-alone configurations. Thanks to the embedded motion controller, the IBL2401 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 or camming[1], etc.)
- Changing the motion modes and/or the motion parameters
- Executing homing sequences[2]
- Controlling the program flow through:
  - Conditional jumps and calls of TML functions
  - TML interrupts generated on pre-defined or programmable conditions (protections triggered, transitions on limit switch or capture inputs, etc.)
  - Waits for programmed events to occur
- Handling of digital I/O and analogue input signals
- Executing arithmetic and logic operations
- Performing data transfers between axes
- Controlling motion of an axis from another one via motion commands sent between axes
- Sending commands to a group of axes (multicast). This includes the possibility to start simultaneously motion sequences on all the axes from the group
- Synchronizing all the axes from a network

Using **EasyMotion Studio** for TML programming you can really distribute the intelligence between the master and the 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 (with possibility to abort their execution if needed) and waiting for a message, which confirms the TML functions execution.

Apart from a CANopen master, the IBL2401 drives can also be controlled from a PC or PLC using the family of **TML_LIB** motion libraries.

For all motion programming options, the IBL2401 commissioning for your application is done using **EasySetUp**.

---

[1] Optional for the IBL2401 CANopen execution
[2] Available only for the IBL2401 CAN executions

2.2. Key Features

- Digital drives for control of brushless DC, brushless AC , DC brushed and step motors with built-in controller and high-level TML motion language
- Position, speed or torque control
- Various motion programming modes:
  - Position profiles with trapezoidal or S-curve speed shape
  - Position, Velocity, Time (PVT) 3rd order interpolation
  - Position, Time (PT) 1st order interpolation
  - Electronic gearing and camming[3]
  - External analogue or digital reference[3]
  - 33 Homing modes
- Single-ended, differential and/or open-collector encoder interface
- Single-ended, open collector Hall sensor interface
- Linear Hall sensor interface[4]
- 7 dedicated digital input-output lines (5V and 24V compatible):
  - 5 digital input lines
  - 2 digital output lines
- RS-232 serial interface (up to 115200 bps)
- CAN-bus 2.0B up to 1Mbit/s, with communication protocol:
  - CANopen[5] – compatible with CiA standards: DS301 and DSP402
  - TMLCAN[6] – compatible with all Technosoft drives with CANbus interface
- 1.5K × 16 internal SRAM memory
- 8K × 16 E^2ROM to store TML programs and data
- Nominal PWM switching frequency: 40 kHz[7]
- Power supply: 6-27 V; 1A; 3.6 A PEAK
- Minimal motor inductance: 25 µH at 12V; 50 µH at 24V
- Operating ambient temperature: 0-40°C
- Hardware Protections:
  - All I/Os are ESD protected

[3] Optional for the IBL2401 CANopen execution
[4] Available only for the IBL2401 CAN executions
[5] Available only for the IBL2401 CANopen executions
[6] Available only for the IBL2401- CAN execution
[7] Nominal values cover all cases. Higher values may be programmed for configurations with brushless DC, DC brush and step motors.

2.3. Supported Motor-Sensor Configurations

IBL2401 supports the following configurations:

1. Position, speed or torque control of a **brushless AC rotary motor** with an **incremental quadrature encoder** on its shaft. The brushless motor is vector controlled like a permanent magnet synchronous motor. It works with **sinusoidal** voltages and currents. Scaling factors take into account the transmission ratio between motor and load (rotary or linear). Therefore, the motion commands (for position, speed and acceleration) expressed in SI units (or derivatives) refer to the load[8], while the same commands, expressed in IU units, refer to the motor.

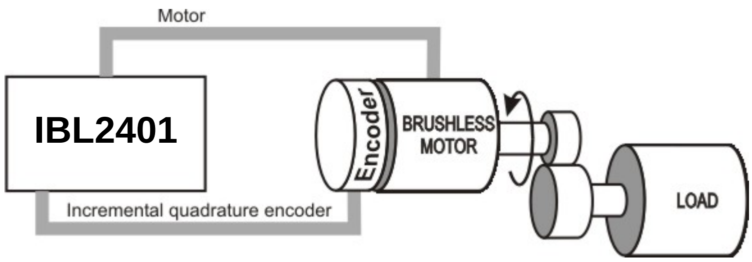


Figure 2.1. *Brushless AC rotary motor. Position/speed/torque control. Quadrature encoder on motor.*

2. Position, speed or torque control of a **brushless DC rotary motor** with **digital Hall sensors** and an **incremental quadrature encoder** on its shaft. The brushless motor is controlled using Hall sensors for commutation. It works with rectangular currents and **trapezoidal** BEMF voltages. Scaling factors take into account the transmission ratio between motor and load (rotary or linear). Therefore, the motion commands (for position, speed and acceleration) expressed in SI units (or derivatives) refer to the load, while the same commands, expressed in IU units, refer to the motor.

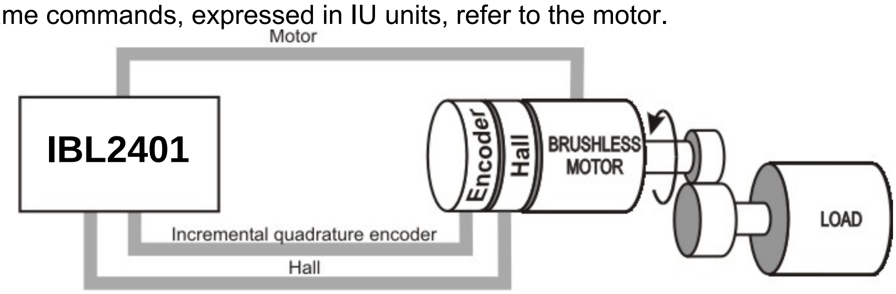


Figure 2.2. *Brushless DC rotary motor. Position/speed/torque control. Hall sensors and quadrature encoder on motor*

[8] Motion commands can be referred to the motor by setting in EasySetUp a rotary to rotary transmission with ratio 1:1

3. Position, speed or torque control of a **brushless AC linear motor** with an **incremental quadrature linear encoder** on the track. The brushless motor is vector controlled like a permanent magnet synchronous motor. It works with **sinusoidal** voltages and currents. Scaling factors take into account the transmission ratio between motor and load (linear or rotary). Therefore, the motion commands (for position, speed and acceleration) expressed in SI units (or derivatives) refer to the load, while the same commands, expressed in IU units, refer to the motor.

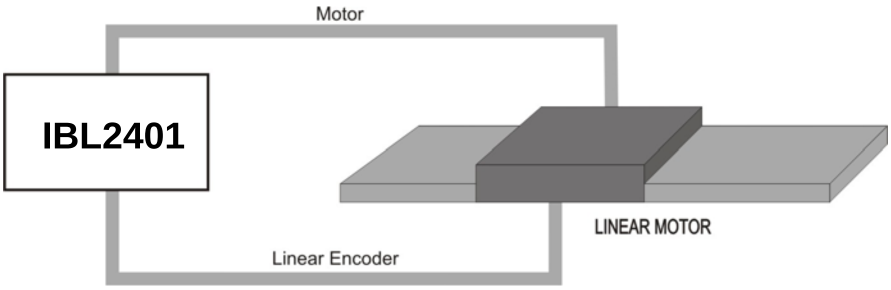


Figure 2.3. *Brushless AC linear motor with incremental quadrature linear encoder signals. Position/speed/torque control*

4. Position, speed or torque control of a **brushless AC rotary motor** with **linear Hall signals**[2]. The brushless motor is vector controlled like a permanent magnet synchronous motor. It works with **sinusoidal** voltages and currents. Scaling factors take into account the transmission ratio between motor and load (rotary or linear). Therefore, the motion commands (for position, speed and acceleration) expressed in SI units (or derivatives) refer to the load[9], while the same commands, expressed in IU units, refer to the motor.

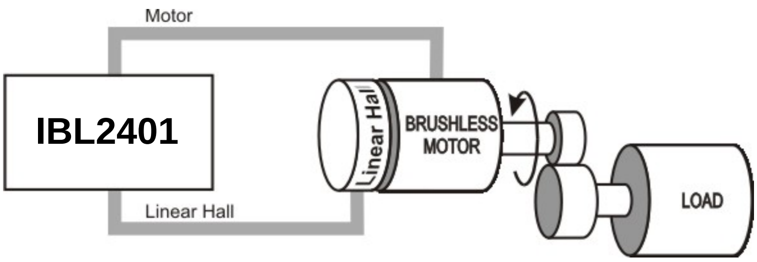


Figure 2.4. *Brushless AC rotary motor with linear Hall signals.. Position/speed/torque control*

5. Position, speed or torque control of a **brushless AC linear motor** with **linear Hall signals**[10]. The brushless motor is vector controlled like a permanent magnet synchronous motor. It works with **sinusoidal** voltages and currents. Scaling factors take into account the transmission ratio between motor and load (rotary or linear). Therefore,

[9] Motion commands can be referred to the motor by setting in EasySetUp a rotary to rotary transmission with ratio 1:1
[10] Available only for the IBL2401 CAN execution

the motion commands (for position, speed and acceleration) expressed in SI units (or derivatives) refer to the load, while the same commands, expressed in IU units, refer to the motor.

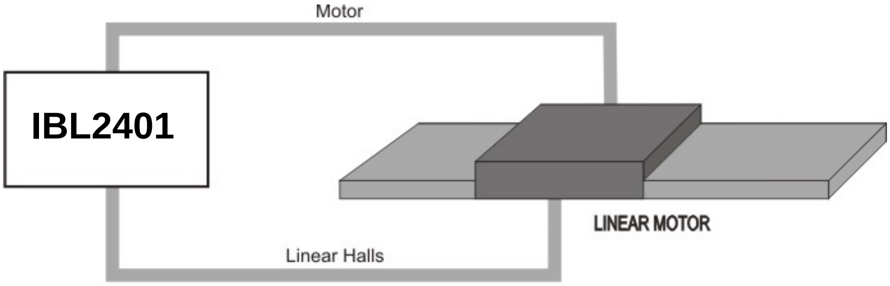


Figure 2.5. *Brushless AC linear motor with linear Hall signals.. Position/speed/torque control*

6. Position, speed or torque control of a **DC brushed rotary motor** with an **incremental quadrature encoder** on its shaft. Scaling factors take into account the transmission ratio between motor and load (rotary or linear). Therefore, the motion commands (for position, speed and acceleration) expressed in SI units (or derivatives) refer to the load[1], while the same commands, expressed in IU units, refer to the motor.

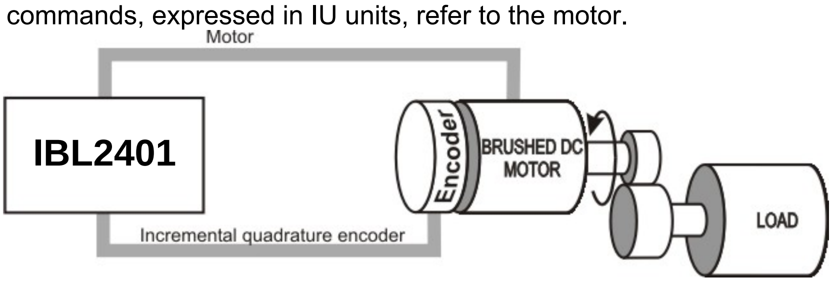


Figure 2.6. *DC brushed rotary motor. Position/speed/torque control. Quadrature encoder on motor*

7. Load position control using an **incremental quadrature encoder** on load, combined with speed control of a **DC brushed rotary motor** having a **tachometer** on its shaft. The motion commands (for position, speed and acceleration) in both SI and IU units refer to the load

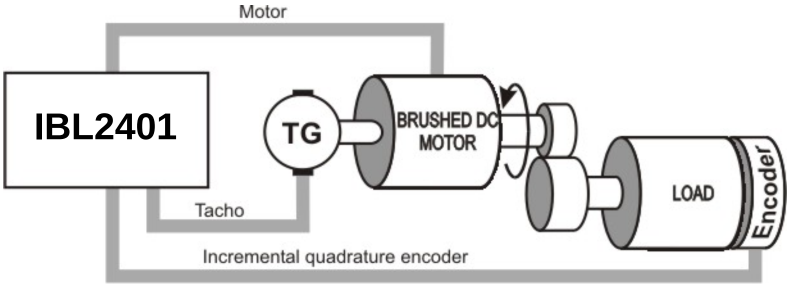


Figure 2.7*. DC brushed rotary motor. Position/speed/torque control. Quadrature encoder on load plus tachometer on motor*

8. Speed or torque control of a **DC brushed rotary motor** with a **tachometer** on its shaft. Scaling factors take into account the transmission ratio between motor and load (rotary or linear). Therefore, the motion commands (for speed and acceleration) expressed in SI units (or derivatives) refer to the load[1], while the same commands, expressed in IU units, refer to the motor

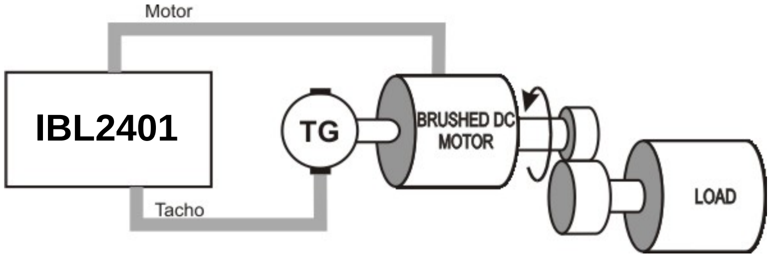


Figure 2.8. *DC brushed rotary motor. Speed/torque control. Tachometer on motor*

9. Open-loop control of a 2 or 3-phase **step motor** in position or speed. Scaling factors take into account the transmission ratio between motor and load (rotary or linear). Therefore, the motion commands (for position, speed and acceleration) expressed in SI units (or derivatives) refer to the load, while the same commands, expressed in IU units, refer to the motor.

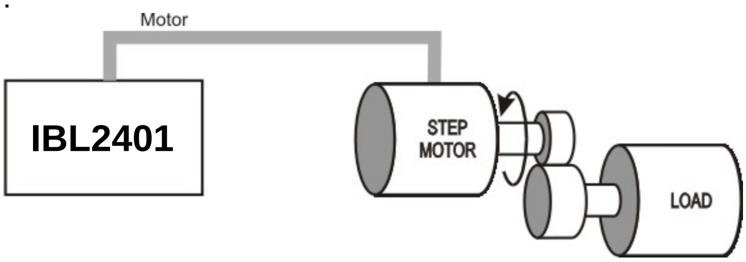


Figure 2.9. *No position or speed feedback. Open-loop control: motor position or speed .*

10. Closed-loop control of **load position using an encoder on load**, combined with open-loop control of **a 2 phase step motor** in speed, with speed reference provided by the position controller. The motion commands in both SI and IU units refer to the load.

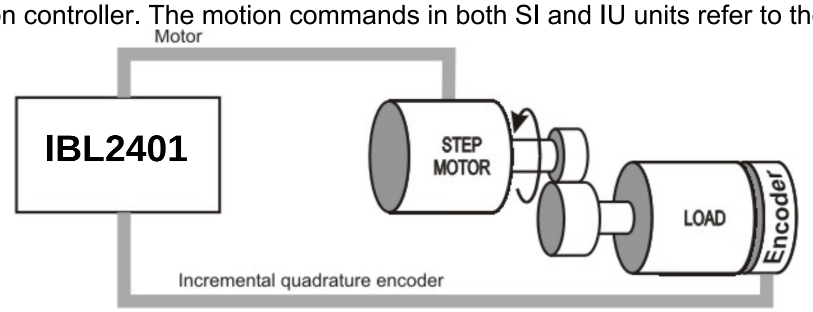


Figure 2.10. *Encoder on load. Closed-loop control: load position, open-loop control: motor speed*

11. Closed-loop control of **a 2-phase step motor** in position, speed or torque. Scaling factors take into account the transmission ratio between motor and load (rotary or linear). Therefore, the motion commands expressed in SI units (or derivatives) refer to the load[11], while the same commands, expressed in IU units refer to the motor.

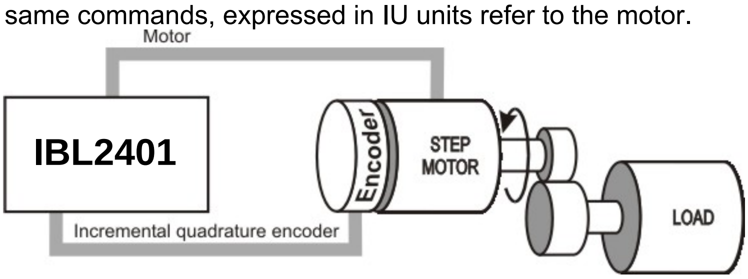


Figure 2.11. *Encoder on motor shaft. Closed-loop control: motor position, speed or torque*

[11] Motion commands can be referred to the motor by setting in EasySetUp a rotary to rotary transmission with ratio 1:1

2.4. IBL2401 Dimensions

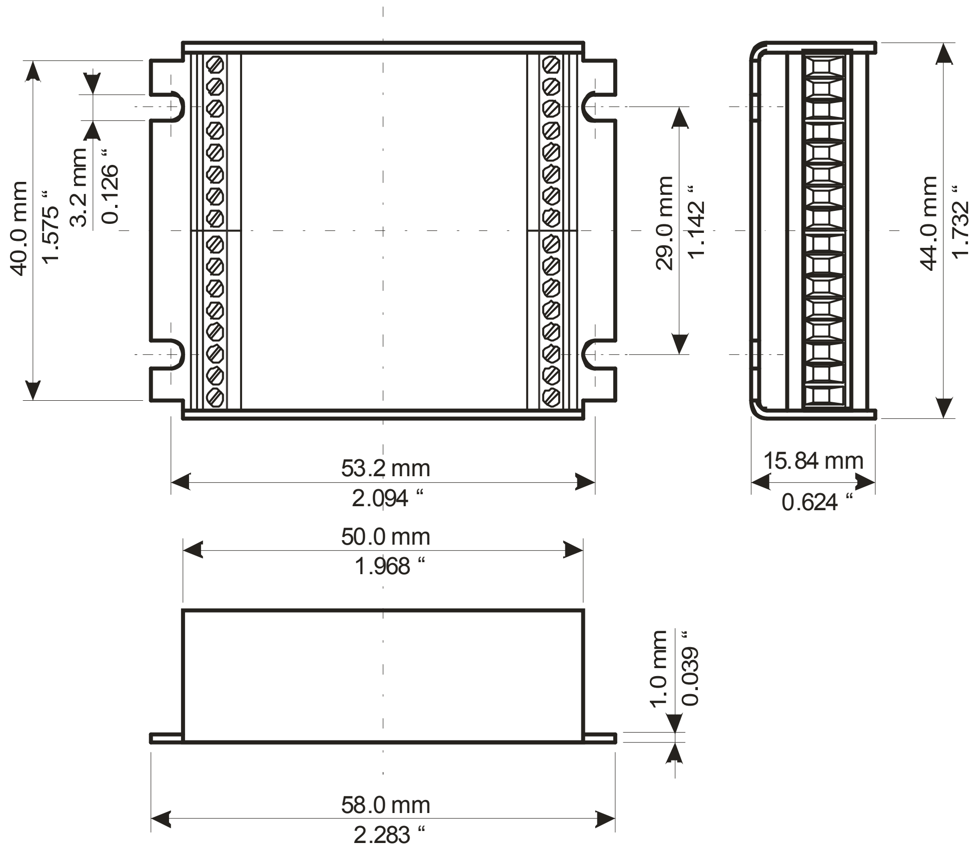


Figure 2.12. *IBL2401 drive dimensions*

2.5. Electrical Specifications

All parameters were measured under the following conditions (unless otherwise specified):
T_{amb} = 25°C, power supply (V_{DC}) = 24V_{DC};
Supplies start-up / shutdown sequence: *-any-* ;
Load current 1 A_{RMS} .

Supply Input

	Measured between +V_{DC} and GND.	Min.	Typ.	Max.	Units
Supply voltage	Nominal values	6	24	27	V_{DC}
	Absolute maximum values, continuous †	-0.5		30	V_{DC}
Supply current	Idle		100	250	mA
	Operating	-3.7	±1	+3.7	A

Motor Outputs

	All voltages referenced to GND.	Min.	Typ.	Max.	Units
Motor output current	Continuous operation, +V_{DC} = 24 V, F_{PWM} = 40 kHz	-1		+1	A_{RMS}
Motor output current, peak	Thermal limited to <= 0.5 s	-3.63		+3.63	A
H/W short-circuit protection threshold			4.8		A
H/W short-circuit protection delay			22		μs
On-state voltage drop	Output current = ±1 A	-900	±250	+300	mV
Off-state leakage current		-1	±0.1	+1	mA
Motor inductance	+V_{DC} = 12 V, F_{PWM} = 40 kHz	25			μH
	+V_{DC} = 24 V, F_{PWM} = 40 kHz	50			μH

Digital Inputs

	All voltages referenced to GND.	Min.	Typ.	Max.	Units
Input voltage	Logic "LOW"	-0.5	0	0.8	V
	Logic "HIGH"	2	5÷24	28	
	Absolute maximum, surge (duration ≤ 1S) †	-25		+30	
Input current	Logic 'HIGH'; Internal 4.7 KΩ pull-up to +5V	0	0	0	mA
	Logic "LOW"	0.8	1	1.3	
Input frequency		0		1	MHz
Minimum pulse width		0.5			μS

Digital Outputs

All voltages referenced to GND.		Min.	Typ.	Max.	Units
Output voltage	Logic "LOW"	-0.5	0	0.2	V
	Logic "HIGH" ; Output current = 0	2.4	4.4	$+V_{DC}$	
	Absolute maximum, duration < 1 ms	-1		$+V_{DC}$ + 0.5	
Output current	Logic "HIGH"; Load connected to GND			10	mA
	Logic "LOW"			50	
ESD Protection	Human Body Model (100 pF, 1.5 kΩ)			±25	KV

Encoder Inputs

		Min.	Typ.	Max.	Units
Standards compliance		Differential / TTL / CMOS / open-collector			
Low level input current	Internal 470 Ω pull-ups to +5 V_{DC}		10	12	mA
Input threshold voltage	In single-ended mode (TTL / CMOS / / open-collector)	1.8	1.9	2	V
Input hysteresis		0.1	0.2	0.5	V
Input frequency	Differential (RS422)	0		3.3	MHz
	Single-ended (TTL/CMOS/open-collector)	0		1	

Analog Inputs (Ref, Tacho)

	Referenced to GND	Min.	Typ.	Max.	Units
Voltage range		0		+5	V
Input impedance			16		KΩ
Resolution			10		bits
Differential linearity	Guaranteed 10-bit no-missing-codes			0.09	% FS[1]
Offset error				±0.3	% FS[1]
Gain error				±5	% FS[1]
Bandwidth (-3 dB)			250		Hz

Linear Hall Inputs (LH1, LH2, LH3)

	Referenced to GND	Min.	Typ.	Max.	Units
Voltage range	Maximum range	0		+5	V
	Operating range	Programmable			
Input current		-0.5		+0.5	mA
Bandwidth (-3 dB)			1		KHz

Hall Inputs (digital)

	All voltages referenced to GND.	Min.	Typ.	Max.	Units
Input voltage	Logic "LOW"	-0.5	0	0.8	V
	Logic "HIGH"	2	5	5.5	
	Absolute maximum, surge (duration ≤ 1ms)†	-8		+8	
Low level input current	Internal 1 kΩ pull-ups to +5 V_{DC}		5	6	mA

RS-232

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

CAN-Bus

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

Supply Output

		Min.	Typ.	Max.	Units
$+5V_{OUT}$ Voltage		4.75	5	5.25	V
$+5V_{OUT}$ available current				150	mA

Others

		Min.	Typ.	Max.	Units
Temperature	Operating	0		40	°C
	Storage (not powered)	-40		85	°C
Humidity (Non-condensing)	Operating	0		90	%RH
	Storage	0		100	%RH
Altitude / pressure[12]	Altitude (referenced to sea level)		0 ÷ 2.5	+7	Km
	Ambient Pressure	0.4	0.74 ÷ 1	4.0	atm
Dimensions	Length x Width x Height	58 x 44 x 16			mm
Weight			40		g
Storage temperature	Not powered	-40	85	°C	
Humidity	Non-condensing	0	90	%RH	
Protection degree		IP20 (according to IEC529)			

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

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

T.B.D. = To be determined

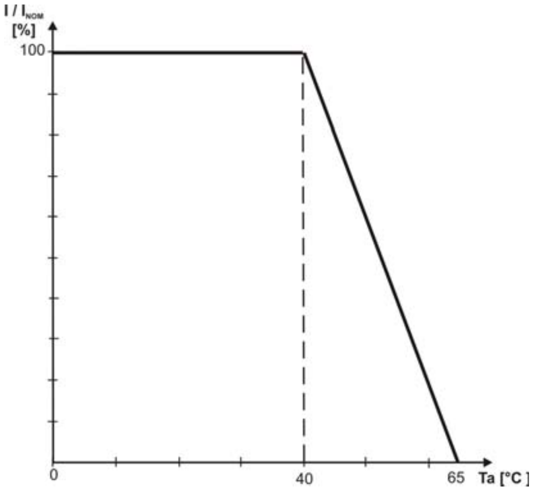


Figure 2.13. *De-rating with ambient temperature*[13] [14]

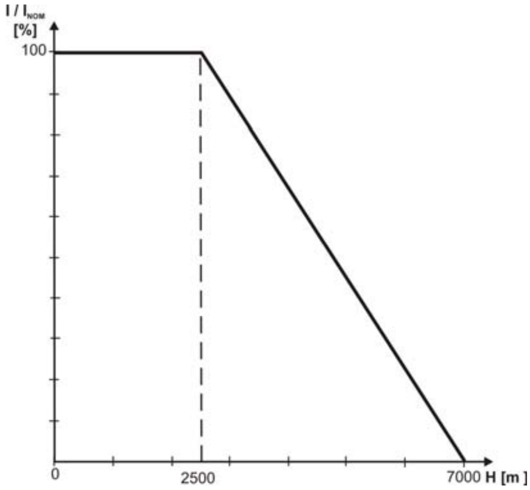


Figure 2.14. *De-rating with altitude*

[12] At altitudes over 1,000m, current and power rating are reduced due to thermal dissipation efficiency at higher altitudes. See Figure 2.13 – De-rating with altitude

[13] I_{NOM} – the nominal current

[14] Stand-alone operation, vertical mounting

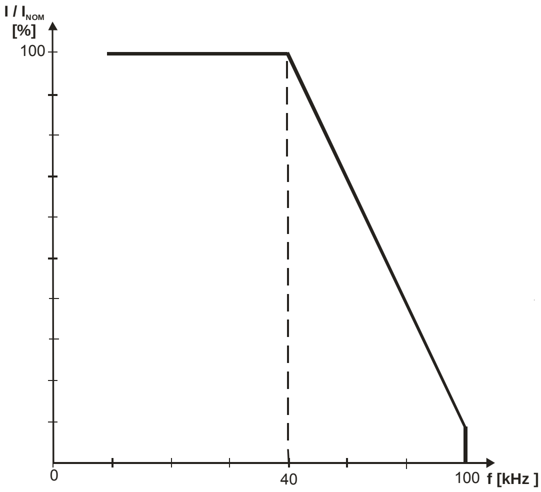


Figure 2. 15. Current De-rating with PWM frequency

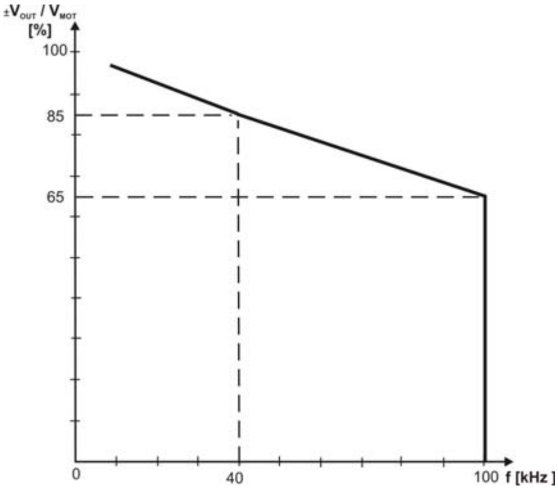


Figure 2.16. Output Voltage De-rating with PWM frequency[15]

CAUTION!	***For PWM frequencies less than 20kHz, correlate the PWM frequency with the motor parameters in order to avoid possible motor damage.***

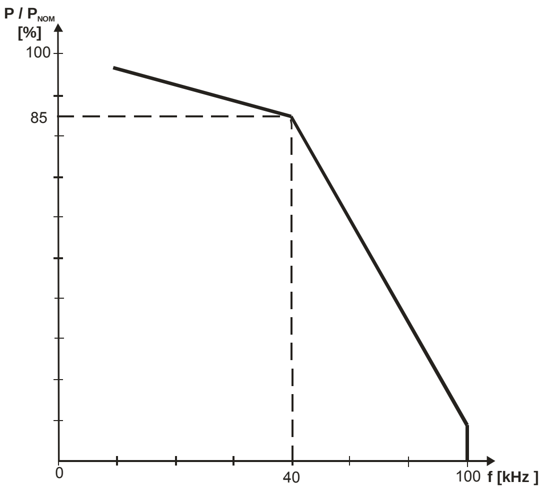


Figure 2.17. Power De-rating with PWM frequency[16]

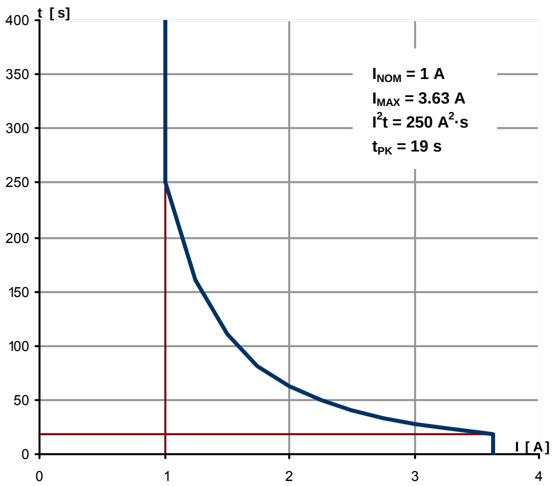


Figure 2.18. Over-current diagram

[15] V_{OUT} – the output voltage, V_{MOT} – the motor supply voltage
[16] P_{NOM} – the nominal power

3. Step 1. Hardware Installation

3.1. Mounting

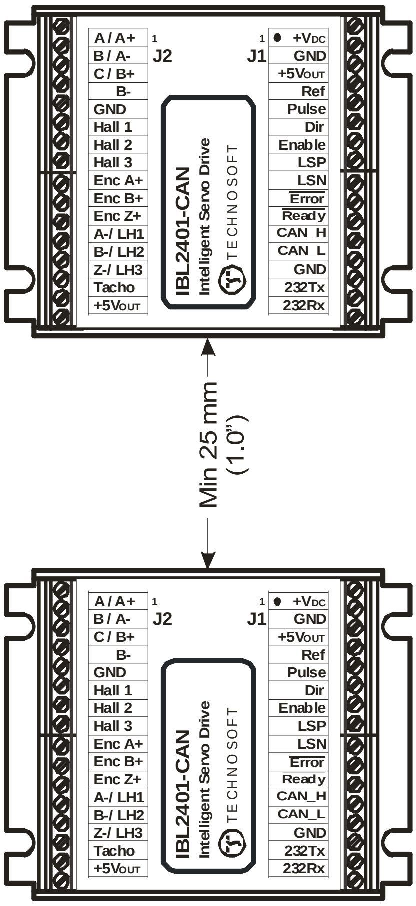


Figure 3.1. *IBL2401 drive connectors*

The IBL2401 drive was designed to be cooled by natural convection. It can be mounted horizontally (with label upwards) or vertically inside a cabinet (see Figure 3.2). In both cases, leave at least 25mm between the drive and surrounding walls/drives, to allow for free air circulation.

CAUTION !	**Before connecting the motor, be sure you have the right application programmed to E2ROM, else you can damage the motor and drive. At power-on, the TML application is automatically executed. See paragraph 3.2.13 to disable this feature.**

3.2. Connectors and Connection Diagrams

3.2.1. Connectors Layout

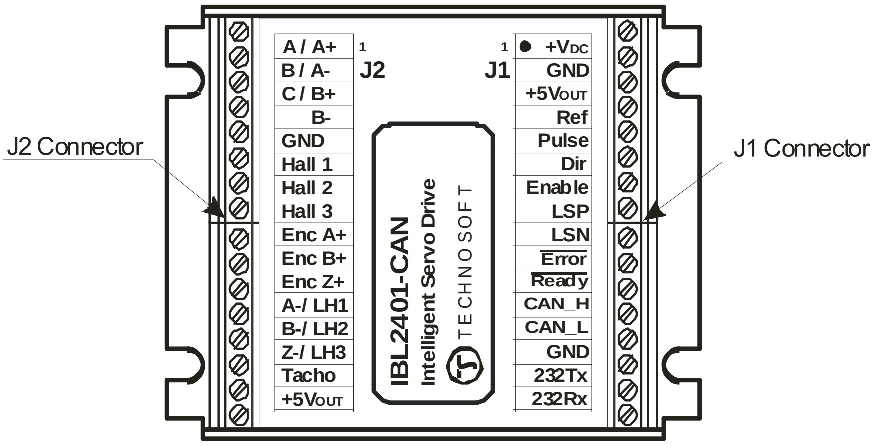


Figure 3.2. *IBL2401 drive connectors*

3.2.2. Identification Labels

Figure 3.3. *IBL2401-RS232 Identification Label*

Figure 3.4. *IBL2401-CAN (CAN execution) Identification Label*

Figure 3.5. *IBL2401-CAN (CANopen execution for Brushless and DC motors with incremental encoder) Identification Label*

Figure 3.6. *IBL2401-CAN (CANopen execution for Step motors with incremental encoder) Identification Label*

3.2.3. J1 Connector pinout

Pin	Pin name	TML name	Type	Function/Alternate function/ Comments
1	**+V$_{DC}$**	-	I	• Positive terminal of the **motor supply**: 12 to 27V$_{DC}$
2	**GND**	-	-	• **Ground**
3	**+5V$_{OUT}$**	-	O	• **5V** output (internally generated)
4	**Ref**	AD5	I	• Unipolar 0 V…+5 V analog input. May be used as analog position, speed or torque reference.
5	**Pulse**	IN#38 / PULSE	I	• **5V** or **24V** compatible digital input • Can be used as PULSE input in Pulse & Direction motion mode • Can be used as second **encoder A** signal, for single-ended encoder
6	**Dir**	IN#37 / DIR	I	• **5V** or **24V** compatible digital input • Can be used as DIRECTION input in Pulse & Direction motion mode • Can be used as second **encoder B** signal, for single-ended encoder
7	**Enable**	IN#16 / ENABLE	I	• **5V** or **24V** compatible digital input • **Enable**. Connect to high to disable PWM outputs
8	**LSP**	IN#2 / LSP	I	• **5V** or **24V** compatible digital input • **Positive limit switch**
9	**LSN**	IN#24 / LSN	I	• **5V** or **24V** compatible digital input • **Negative limit switch**
10	**/ Error**	OUT#13	O	• **5V** or **24V** compatible digital output • **Error**
11	**/ Ready**	OUT#25	O	• **5V** or **24V** compatible digital output • **Ready**
12	**CAN_H**	-	I/O	• Can-Bus positive line (positive during dominant bit) • **Not connected** on the no-CAN execution of the IBL2401 drive (P035.001.E001)
13	**CAN_L**	-	I/O	• CAN-Bus negative line (negative during dominant bit) • **Not connected** on the no-CAN execution of the IBL2401 drive (P035.001.E001)
14	**GND**	-	-	• **Ground**
15	**232Tx**	-	O	• RS-232 Data Transmission
16	**232Rx**	-	I	• RS-232 Data Reception

3.2.4. J2 Connector pinout

Pin	Pin name	TML name	Type	Function/Alternate function/ Comments
1	**A / A+**	-	O	• **Phase A** for brushless motors • **Phase A+** for step motors • **Motor+** for DC brush motors
2	**B / A-**	-	O	• **Phase B** for brushless motors • **Phase A-** for step motors • **Motor-** for DC brush motors
3	**C / B+**	-	O	• **Phase C** for brushless motors • **Phase B+** for step motors
4	**B-**	-	-	• **Phase B-** for step motors
5	**GND**	-	-	• **Ground**
6	**Hall 1**	-	I	• **Hall 1** signal for digital Hall sensor • **Not-autorun**. Connect all 3 Hall signals to GND in order to disable the Autorun
7	**Hall 2**	-	I	• **Hall 2** signal for digital Hall sensor • **Not-autorun**. Connect all 3 Hall signals to GND in order to disable the Autorun
8	**Hall 3**	-	I	• **Hall 3** signal for digital Hall sensor • **Not-autorun**. Connect all 3 Hall signals to GND in order to disable the Autorun
9	**Enc A+**	-	I	• Single-ended encoder A signal • Differential encoder positive A input
10	**Enc B+**	-	I	• Single-ended encoder B signal • Differential encoder positive B input
11	**Enc Z+**	-	I	• Single-ended encoder Z signal • Differential encoder positive Z input
12	**A- / LH1**	-	I	• Differential encoder **negative A** signal • **Linear Hall 1** signal
13	**B- / LH2**	-	I	• Differential encoder **negative B** signal • **Linear Hall 2** signal
14	**Z- / LH3**	-	I	• Differential encoder **negative Z** signal • **Linear Hall 3** signal
15	**Tacho**	AD2	I	• Unipolar 0 V…+5 V analog input. May be used as analog position or speed feedback (from a tachometer)
16	**+5 V_{OUT}**	-	O	• **5V** logic supply (internally generated)

3.2.5. 24V Digital I/O connection

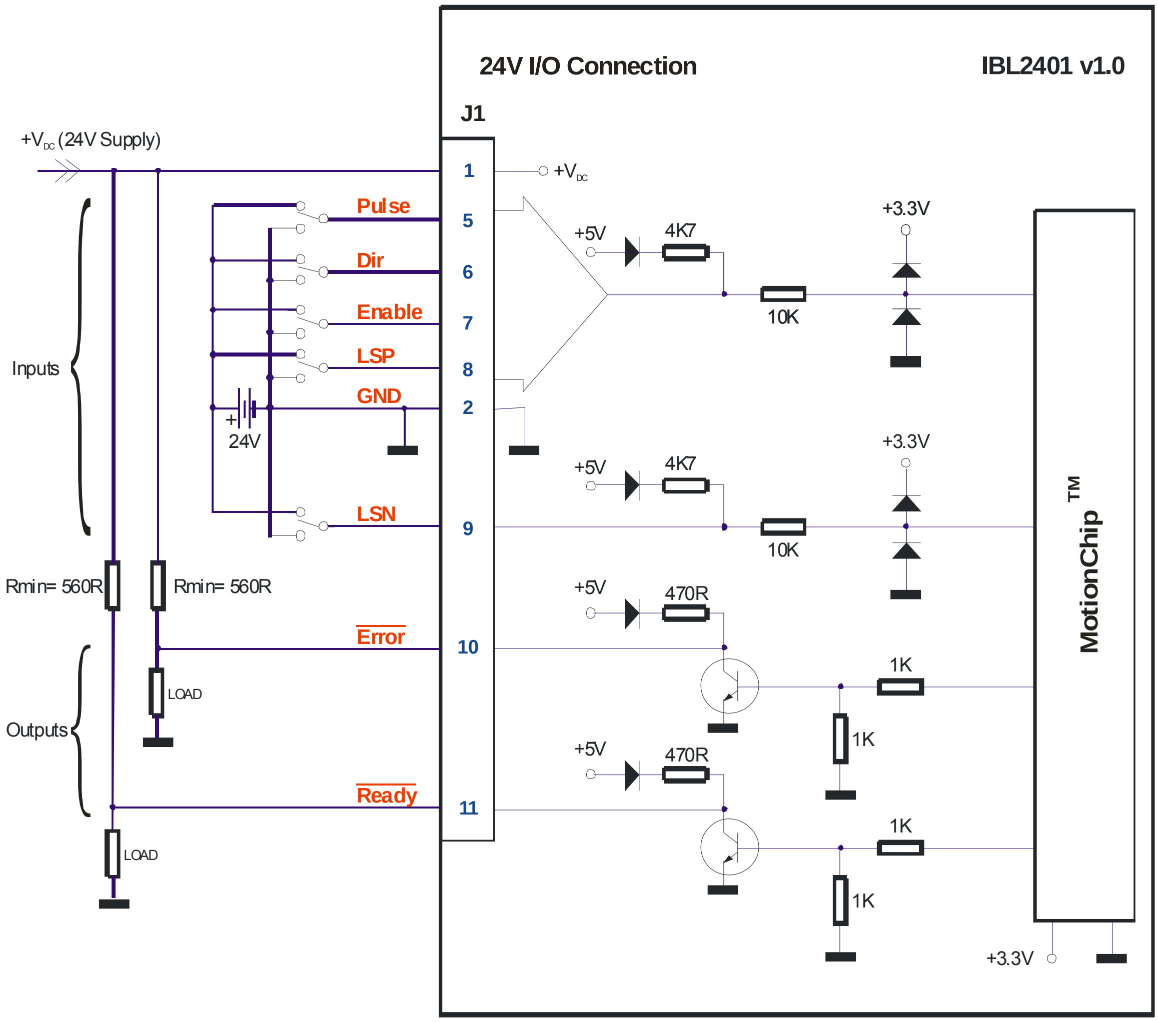


Figure 3.7. *24V Digital I/O connection*

Remarks:

1. *In order to use 24V outputs, an external resistor needs to be connected to a supply of* $+V_{DC}$
2. *The minimum value of external resistors must be 560 Ω.*
3. *The inputs are compatible with NPN type outputs.*
4. *The outputs are compatible with TTL (5V) and NPN (24V) inputs.*

3.2.6. 5V Digital I/O connection

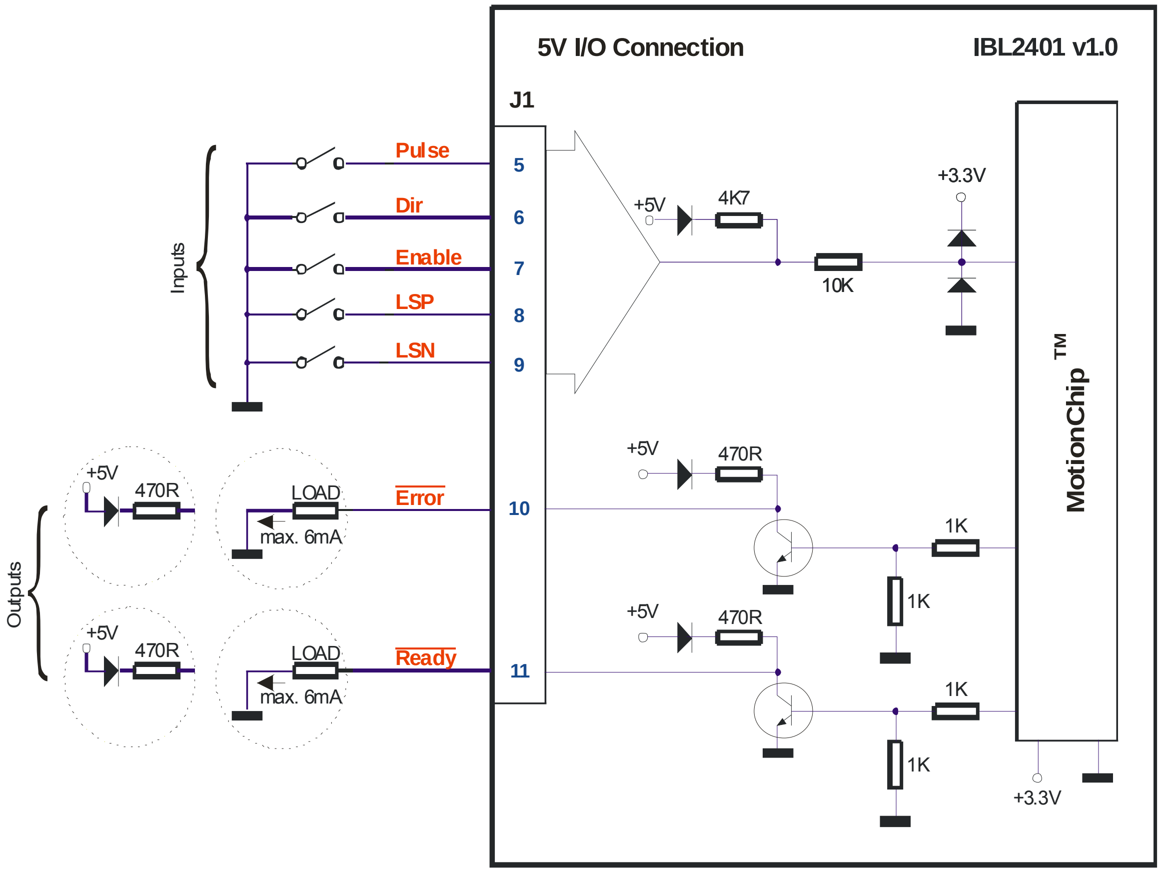


Figure 3.8. *5V Digital I/O connection*

Remarks:

1. *The inputs are compatible with NPN type outputs.*
2. *The outputs are compatible with TTL (5V) and NPN (24V) inputs.*

3.2.7. Analog inputs connection

3.2.7.1 Analog inputs connection

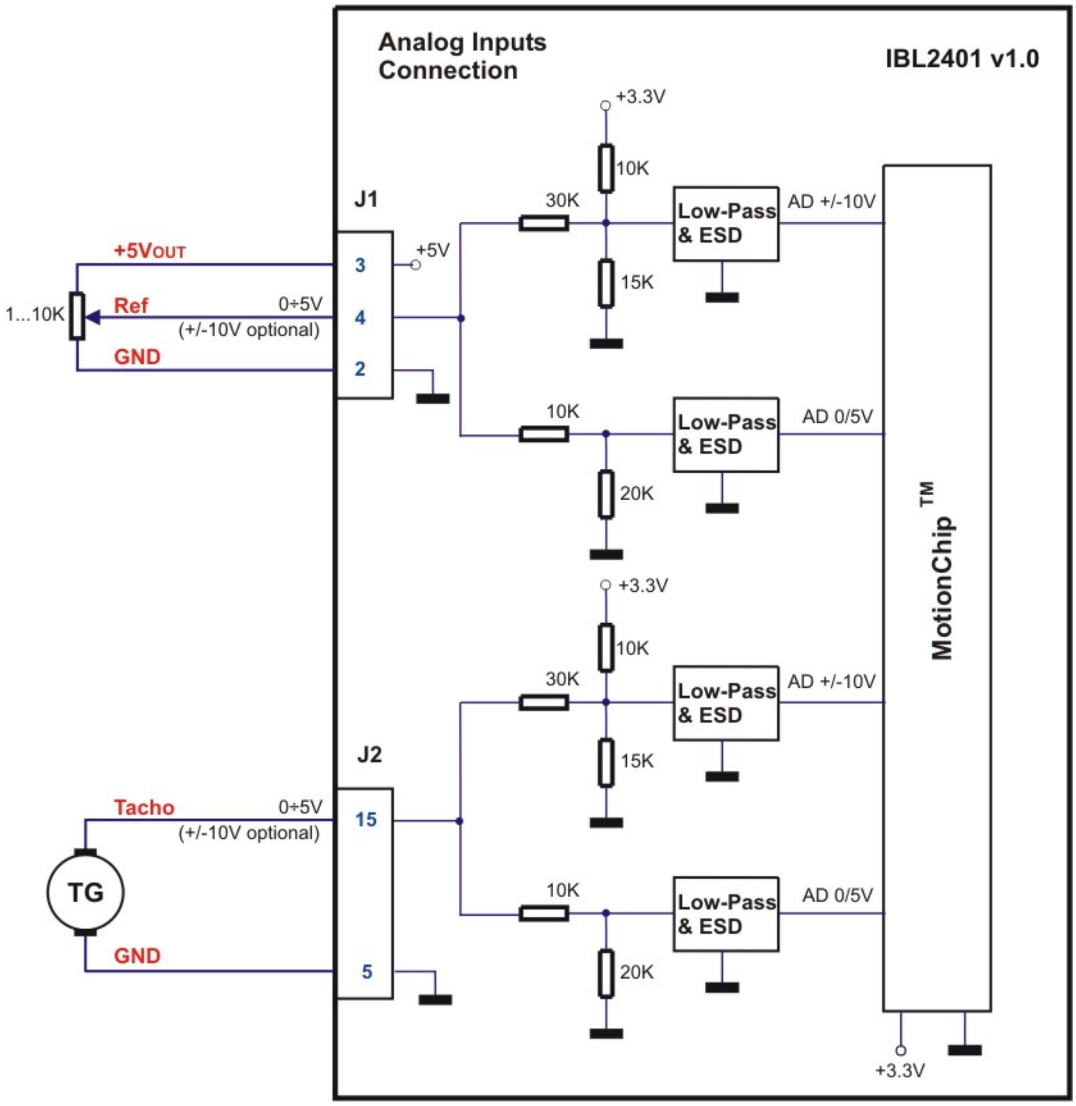


Figure 3.9. *Analog inputs connection*

Remark: *Default input range for analog inputs is 0÷5 V. For a +/-10 V range, please contact Technosoft.*

3.2.7.2 Recommendation for wiring

a) If the analogue signal source is single-ended, use a 2-wire shielded cable as follows: 1st wire connects the live signal to the drive positive input (+); 2nd wire connects the signal ground to the drive negative input (-).

b) If the analogue signal source is differential and the signal source ground is isolated from the drive GND, use a 3-wire shielded cable as follows: 1st wire connects the signal plus to the drive positive input (+); 2nd wire connects the signal minus to the drive negative input (-) and 3rd wire connects the source ground to the drive GND

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 signal plus to the drive positive input (+); 2nd wire connects the signal minus to the drive negative input (-).

3.2.8. Motor connections

3.2.8.1 Brushless Motor connection

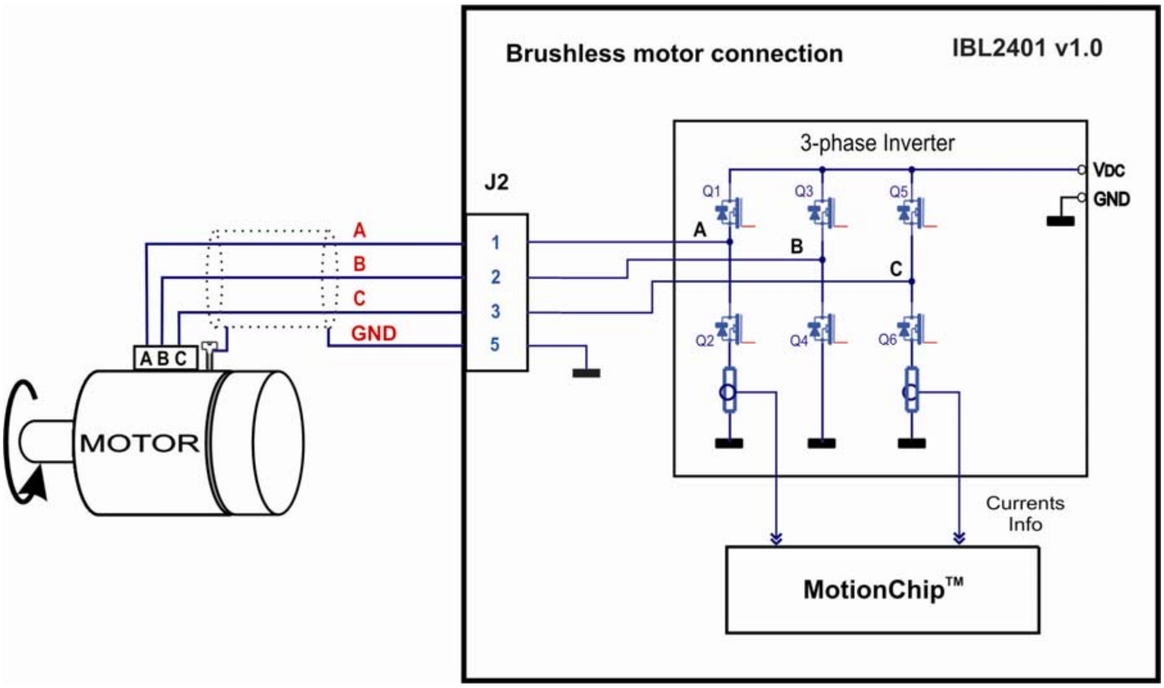


Figure 3.10. Brushless Motor connection

⚠	**CAUTION !**	**Before connecting the motor, be sure you have the right application programmed to E2ROM, else you can damage the motor and drive. At power-on, the TML application is automatically executed. See paragraph 3.2.13 to disable this feature.**

3.2.8.2 2-phase Step Motor connection

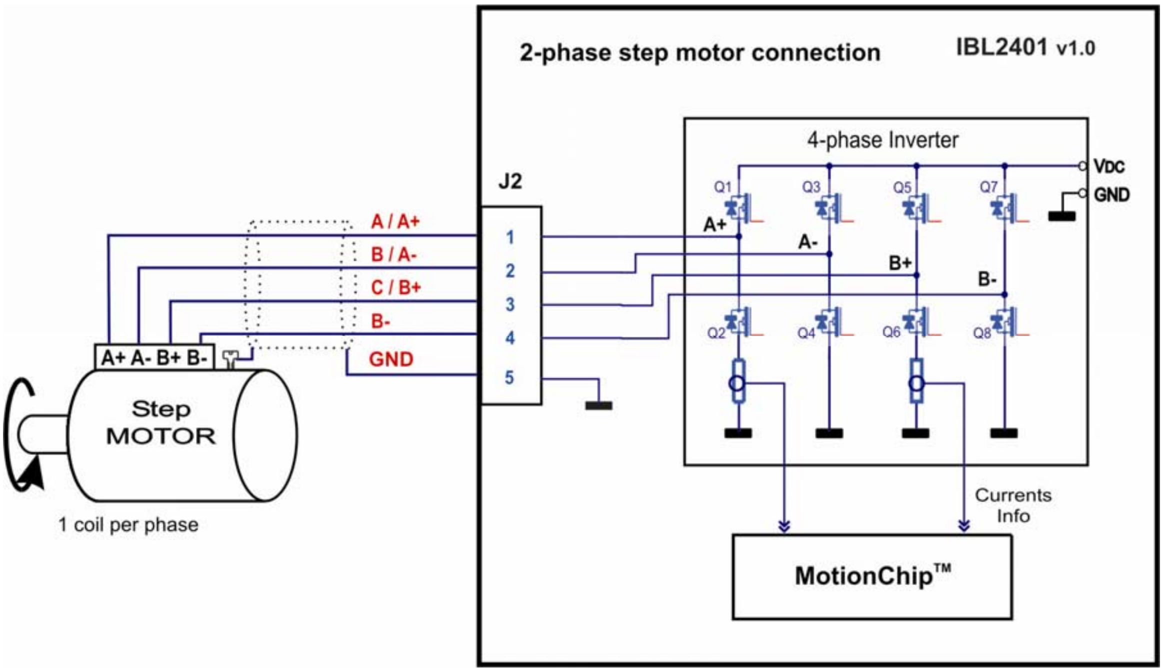


Figure 3.11. *Step Motor connection*

⚠	**CAUTION !**	**Before connecting the motor, be sure you have the right application programmed to E2ROM, else you can damage the motor and drive. At power-on, the TML application is automatically executed. See paragraph 3.2.13 to disable this feature.**

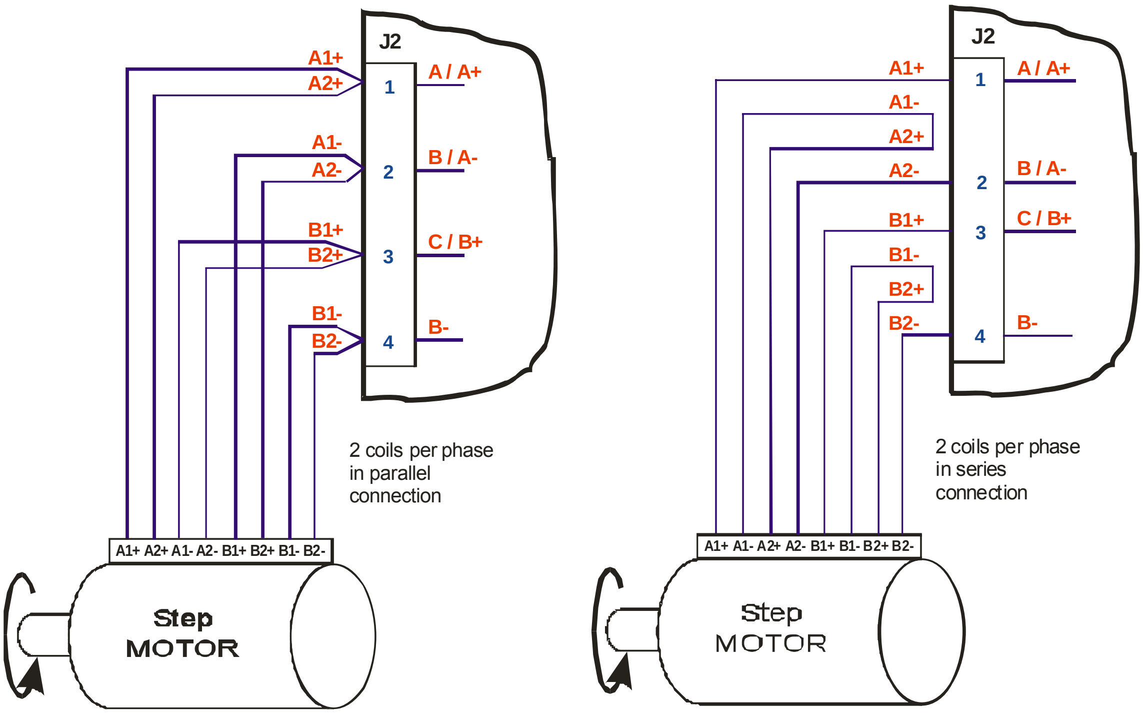


Figure 3.12. *Step Motor connection*

CAUTION !	**Before connecting the motor, be sure you have the right application programmed to E2ROM, else you can damage the motor and drive. At power-on, the TML application is automatically executed. See paragraph 3.2.13 to disable this feature.**

3.2.8.3 3-phase Step Motor connection

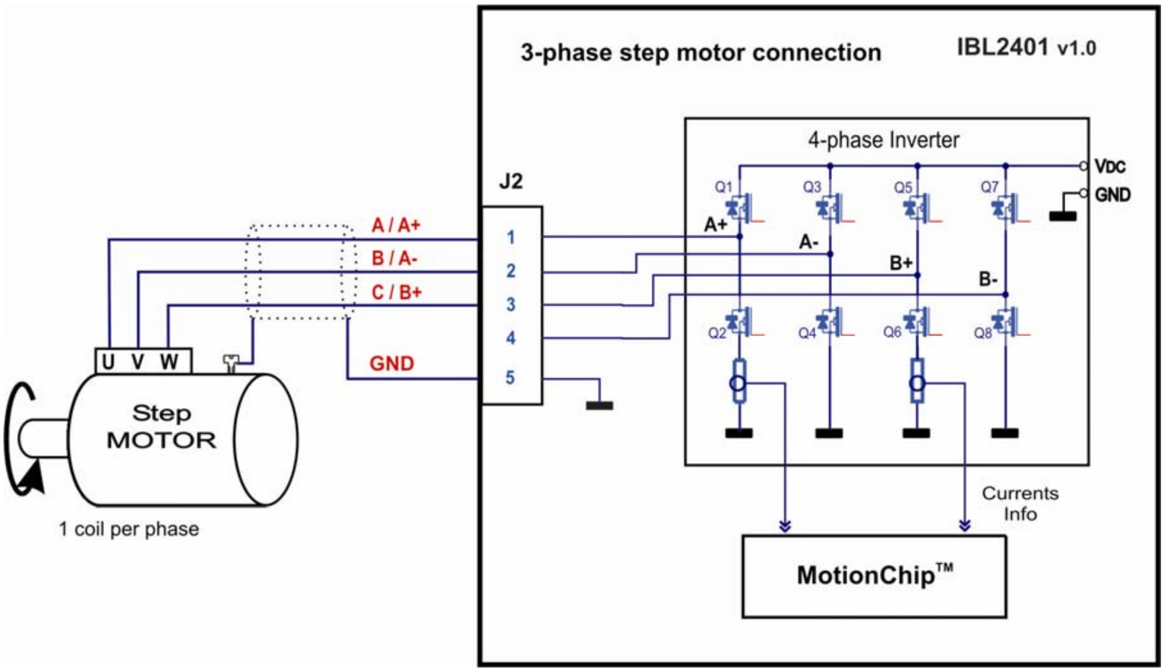


***Figure 3.13**. 3-phase Step Motor connection*

CAUTION !	**Before connecting the motor, be sure you have the right application programmed to E2ROM, else you can damage the motor and drive. At power-on, the TML application is automatically executed. See paragraph 3.2.13 to disable this feature.**

3.2.8.4 DC Motor connection

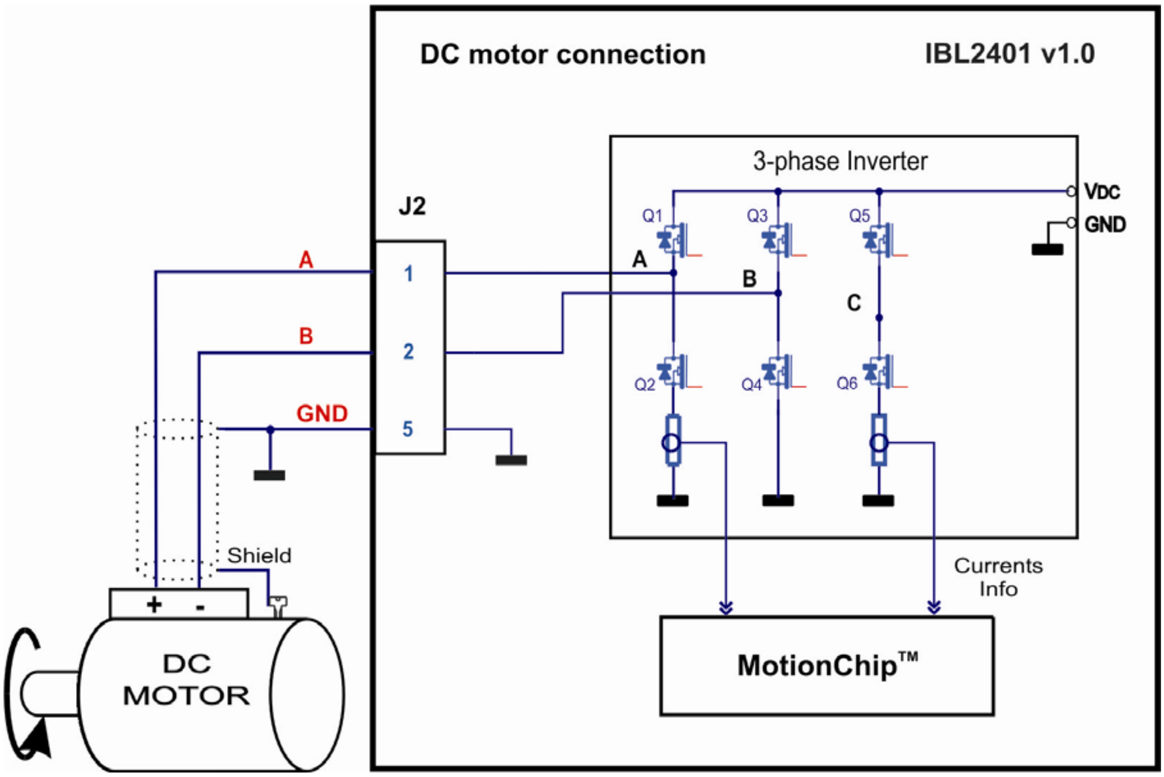


Figure 3.14. *DC Motor connection*

CAUTION !	**Before connecting the motor, be sure you have the right application programmed to E2ROM, else you can damage the motor and drive. At power-on, the TML application is automatically executed. See paragraph 3.2.13 to disable this feature.**

3.2.8.5 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 IBL2401 GND pin. Leave the other end disconnected.

b) The parasitic capacitance between the motor wires must not bypass 100nF. If very long cables (hundreds of meters) are used, this condition may not be met. In this case, add series inductors between the IBL2401 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.

3.2.9. Feedback connections

3.2.9.1 Single-ended encoder connection

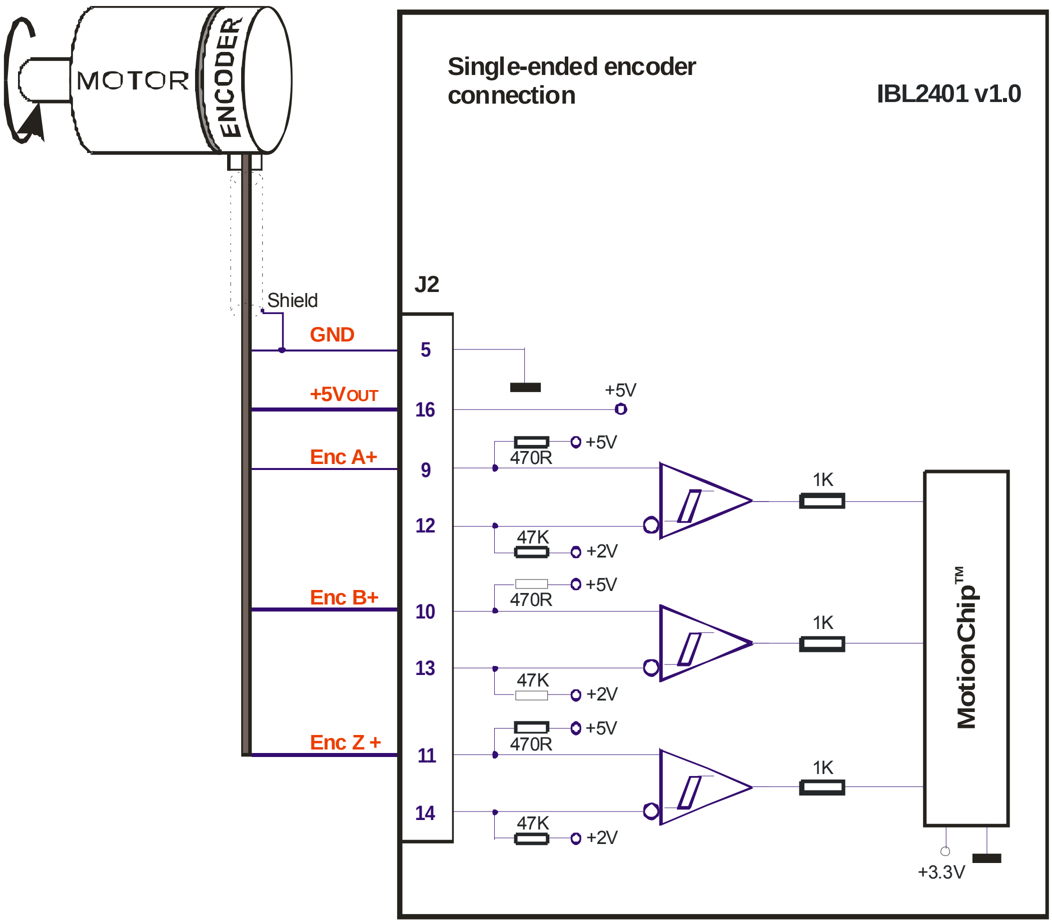


Figure 3.15. *Single-ended encoder connection*

3.2.9.2 Differential encoder connection

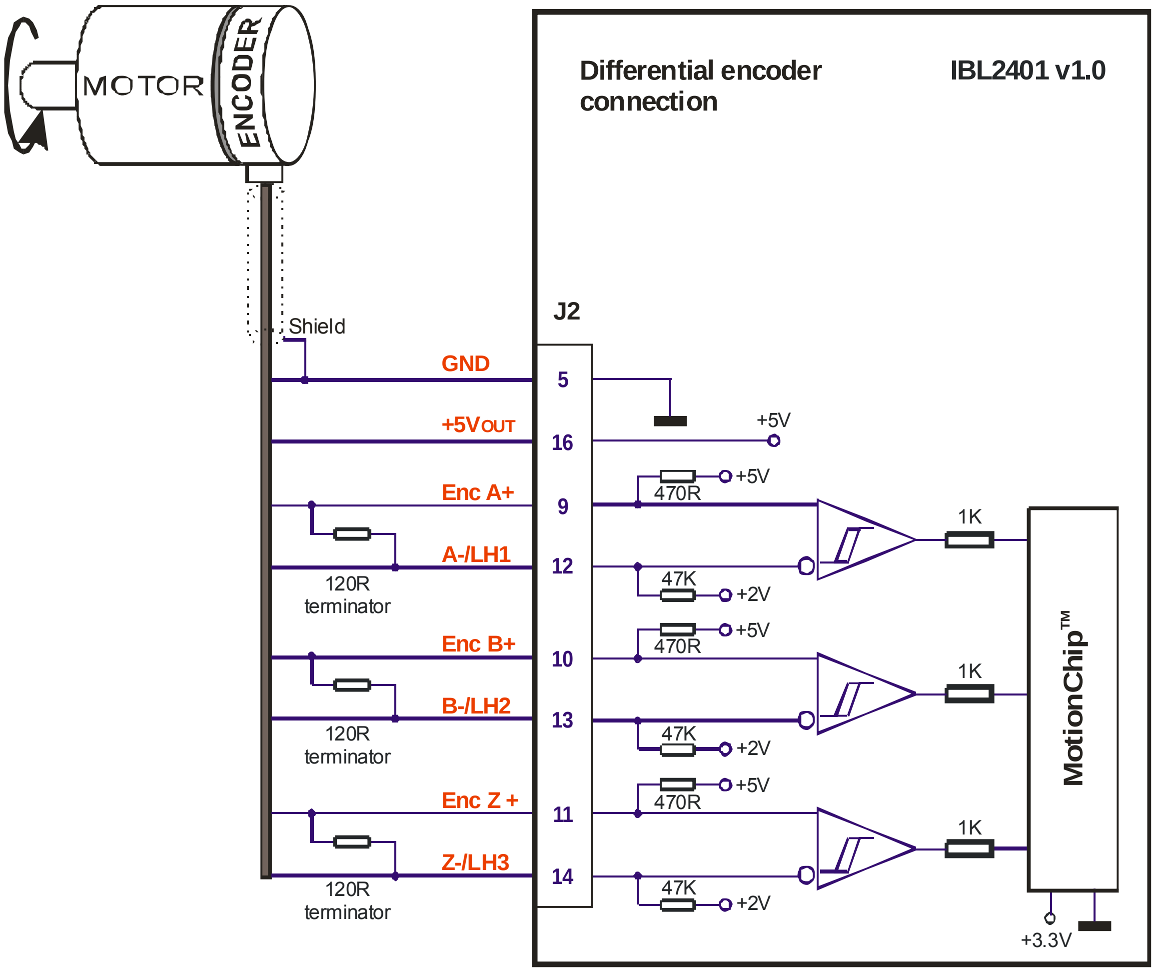


Figure 3.16. *Differential encoder connection*

Remark: *120-Ω (0.25-W) terminators are required for long encoder cables, or noisy electromagnetic environments.*

3.2.9.3 Hall connection

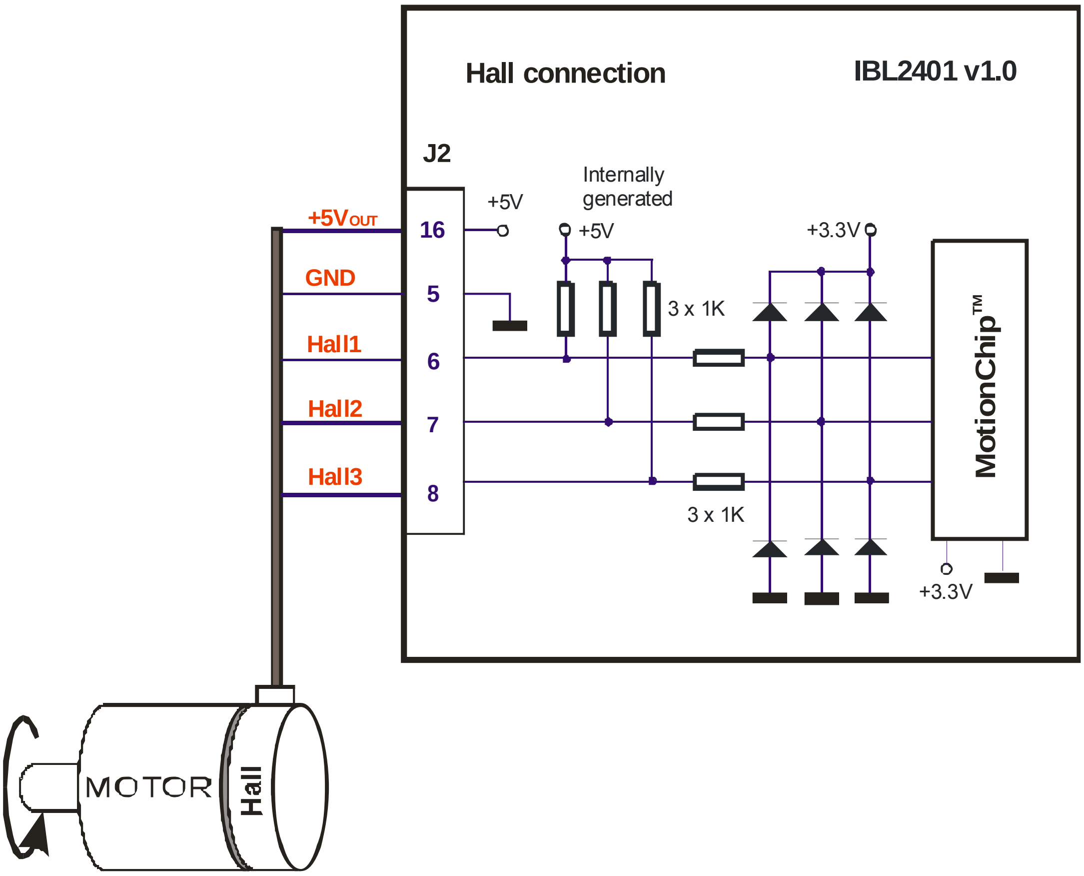


Figure 3.17. *Hall connection*

3.2.9.4 Linear Hall connection

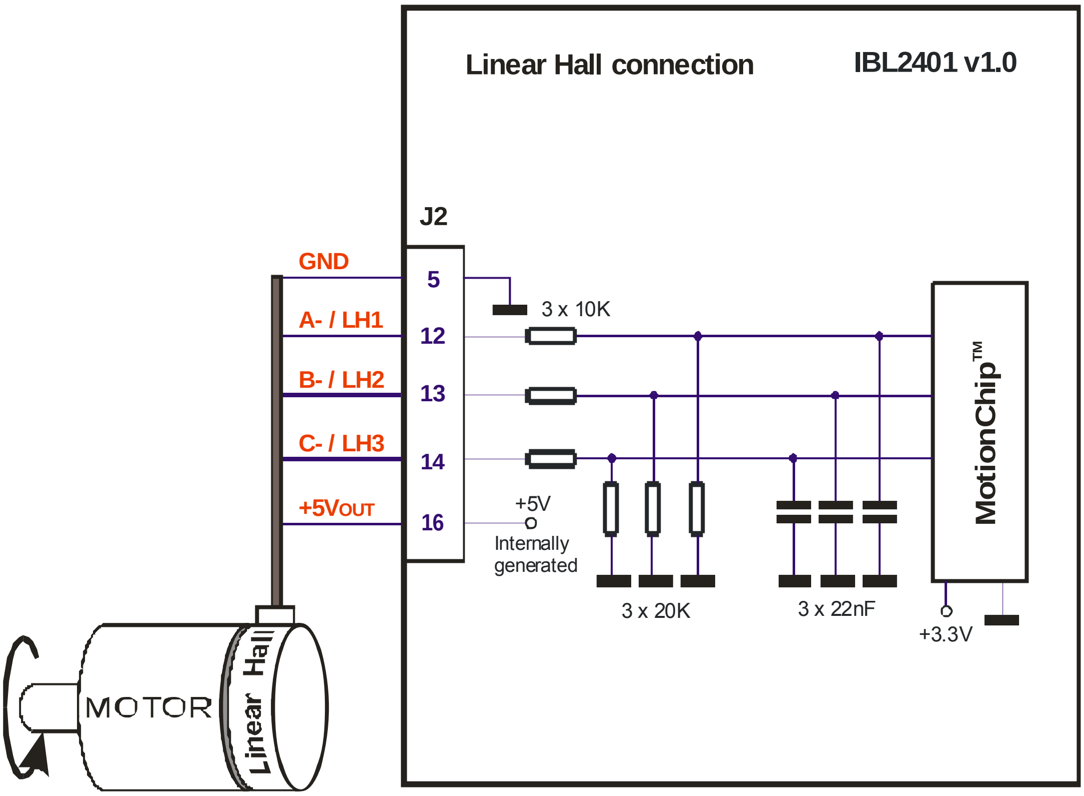


Figure 3.18. *Linear Hall connection*

3.2.9.5 Linear Hall Auto-Setup connection

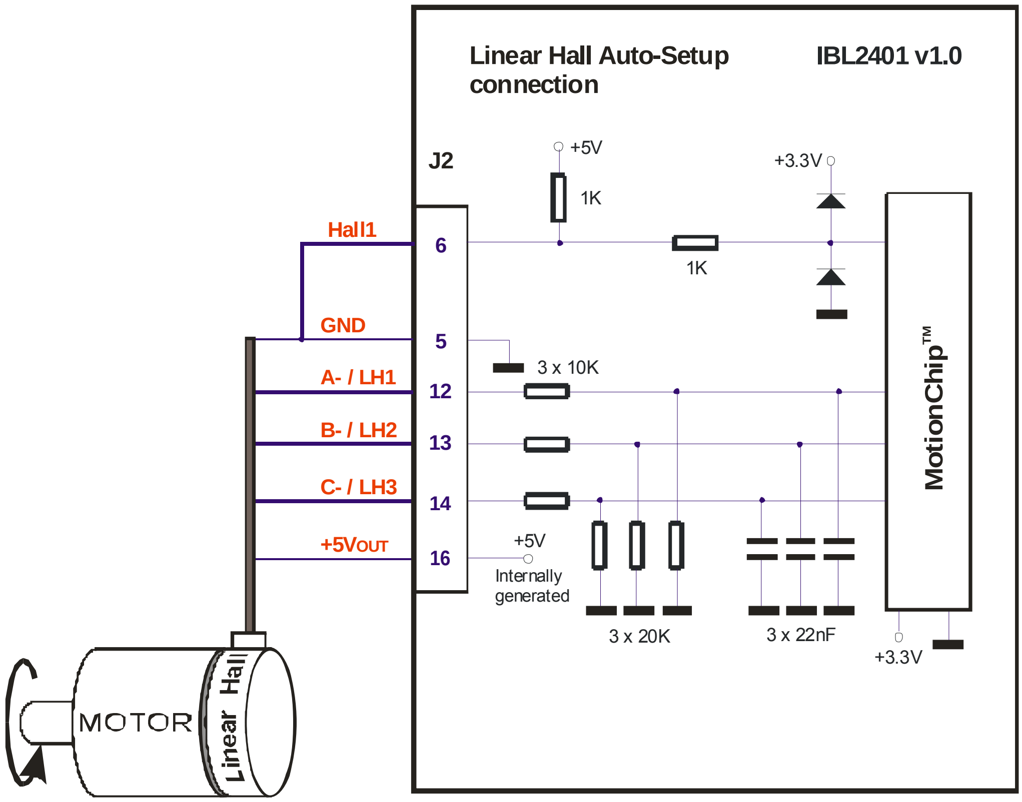


Figure 3.19. *Linear Hall Auto-Setup connection*

3.2.9.6 Recommendations for wiring

a) Always connect both positive and negative signals when the encoder or the Hall sensors are differential and provides them. Use one twisted pair for each differential group of signals as follows: Enc A+ with A-/LH1, Enc B+ with B-/LH2, Enc 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 IBL2401 (using the GND pin) or the encoder / motor. Do not connect the shield at both ends.

c) If the IBL2401 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.

3.2.10. Supply connection

3.2.10.1 Supply connection

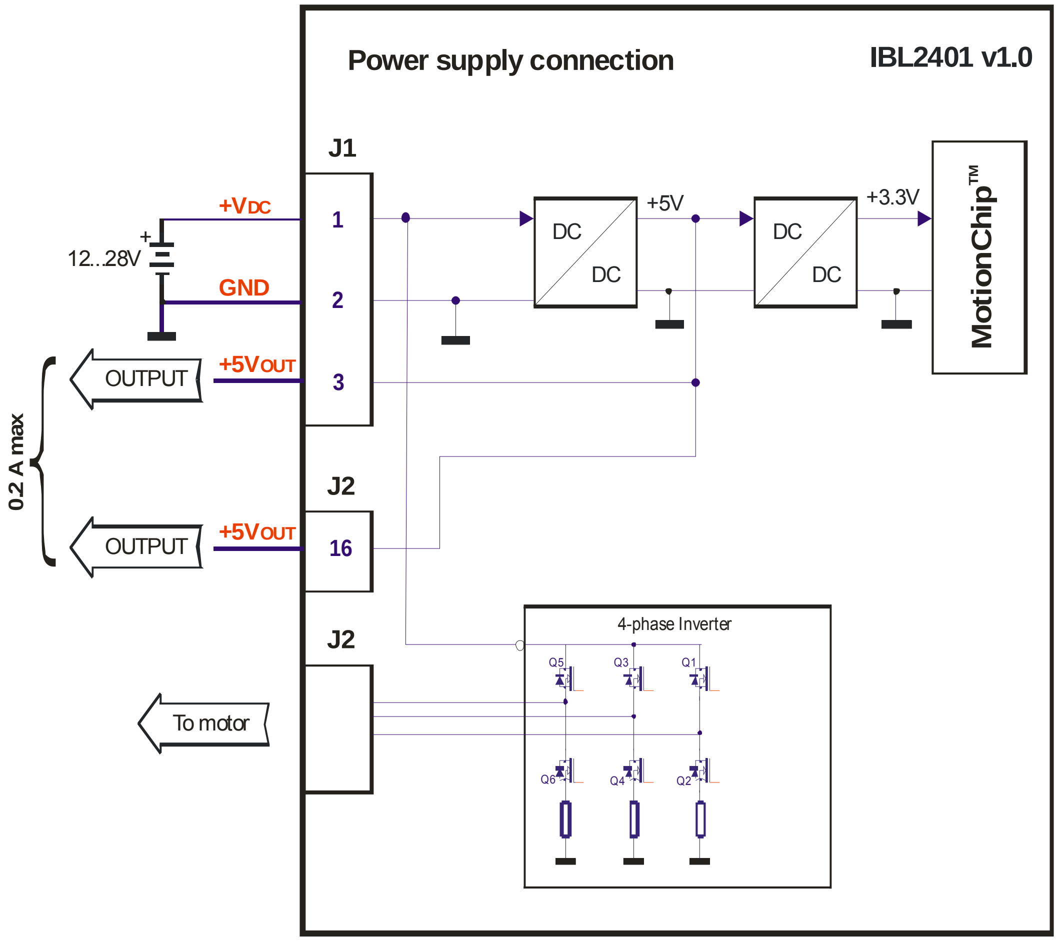


Figure 3.20. *Supply connection*

3.2.10.2 Recommendations for Supply Wiring

Use short, thick wires between the IBL2401 and the motor power supply. If the wires are longer than 2 meters, use twisted wires for the supply and ground return. For wires longer than 20 meters, add a capacitor of at least 1000 μF (rated at an appropriate voltage) right on the terminals of the IBL2401.

3.2.10.3 Recommendations to limit over-voltage during braking

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

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

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

where:

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

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

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

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

E_M can be computed with the formula:

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

where:

J_M – total rotor inertia [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].

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

3.2.11. Serial RS-232 connection

3.2.11.1 Serial RS-232 connection

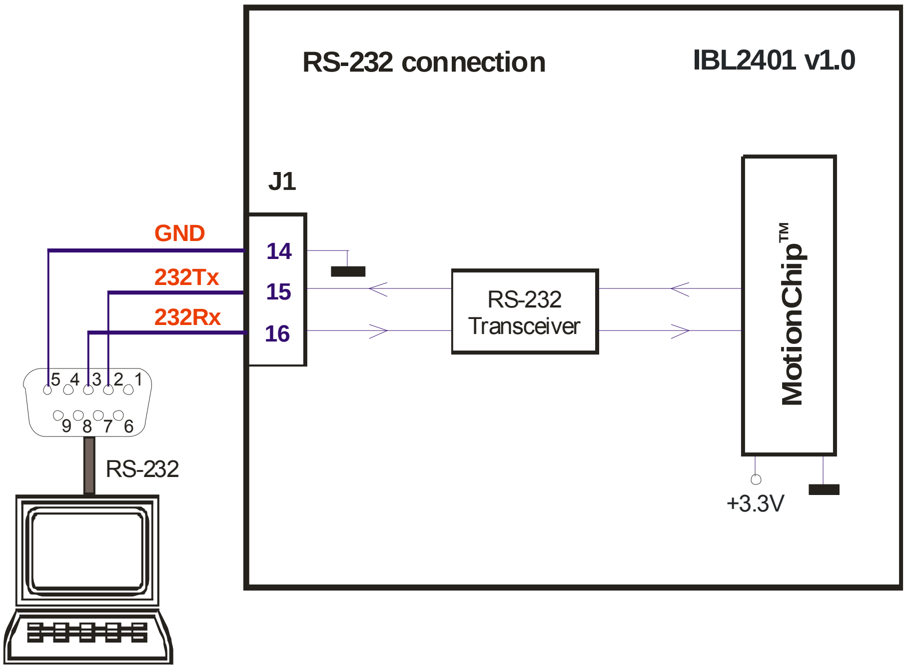


Figure 3.21. *Serial RS-232 connection*

3.2.11.2 Recommendation for wiring

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

b) Do not rely on an earthed PC to provide the IBL2401 GND connection! The drive must be earthed through a separate circuit. Most communication problems are caused by the lack of such connection

c) Always power-off all the IBL2401 supplies before inserting/removing the RS-232 serial connector.

3.2.12. CAN connection (IBL2401-CAN drives)

3.2.12.1 CAN connection (IBL2401-CAN drives)

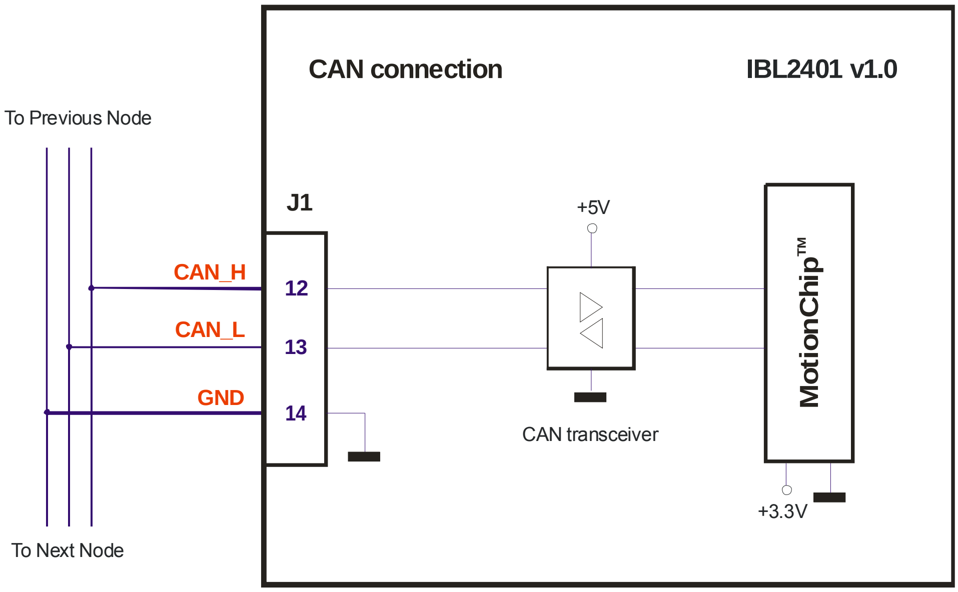


Figure 3.22. *CAN connection*

Remarks:

1. *The CAN network requires a 120-Ohm terminator. This is not included on the board. See Figure 4.14.*
2. *CAN signals are not insulated from other IBL2401 circuits.*
3. *CAN signals (CAN_H and CAN_L pins of J1 connector) are* ***not connected*** *pins on the IBL2401-RS232 drive*

3.2.12.2 Recommendation for wiring

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

b) When total CAN bus length is below 5 meters, it is possible to use a standard phone straight-through cable (with parallel wires)

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.

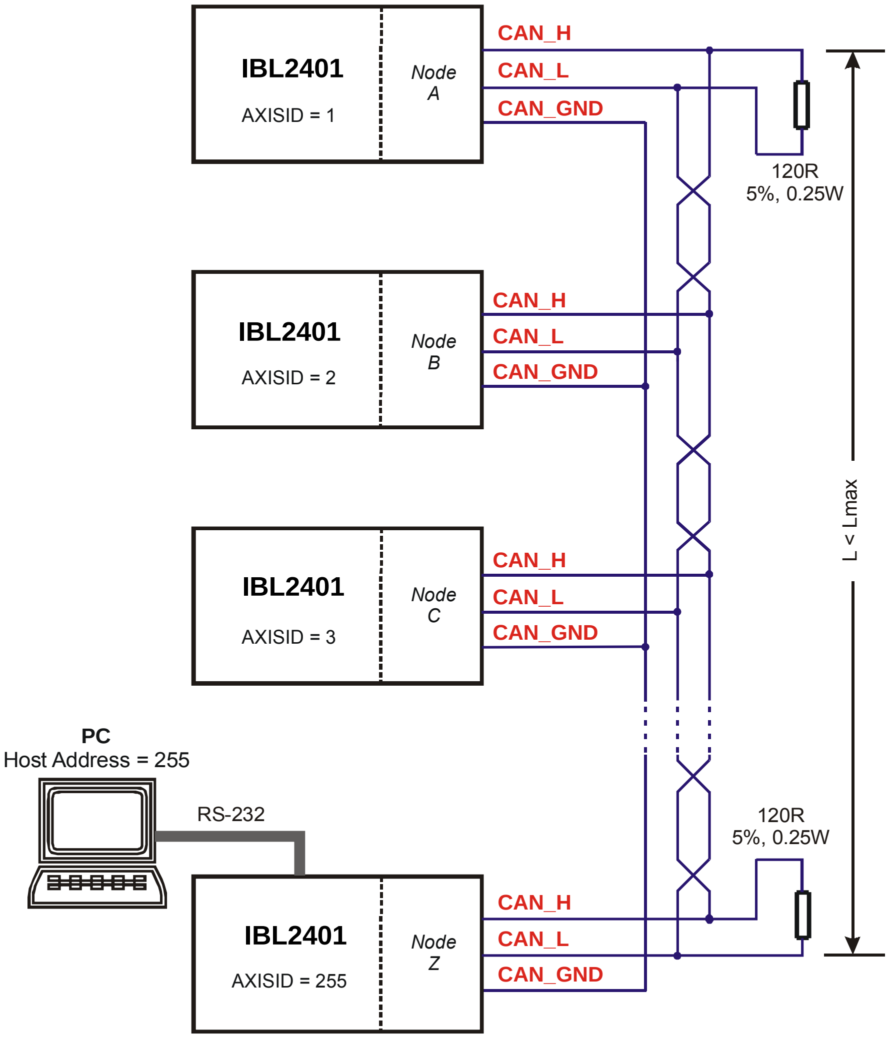


Figure 3.23. *Multiple-Axis CAN network*

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

3.2.13. Special connection (Non-Autorun)

If the drive contains in the E2ROM a valid TML application, when power-on this application is automatically executed (the drive is by default in the *autorun* mode).

To disable this feature in case that the application in the E2ROM is corrupted and the RS232 communication is lost, make the following connections:

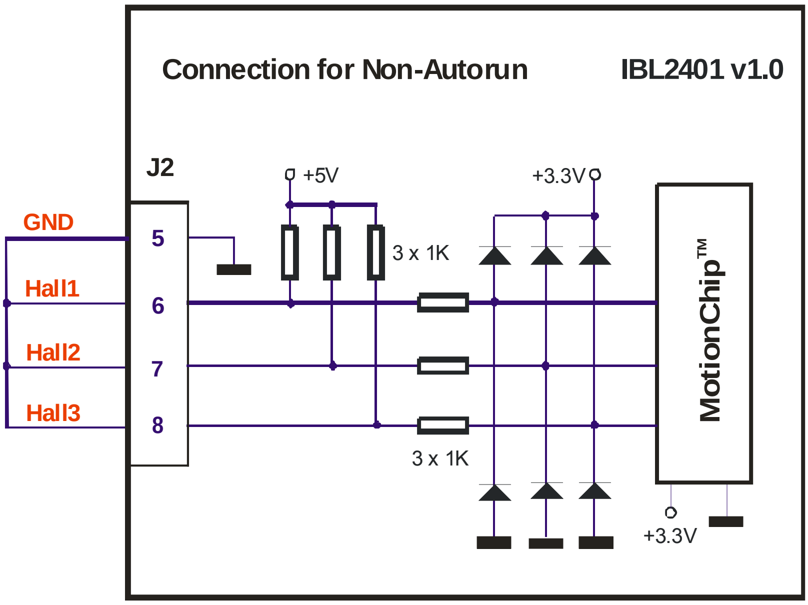


Figure 3.24. *Connection for Non-Autorun*

3.2.14. Master - Slave encoder connection

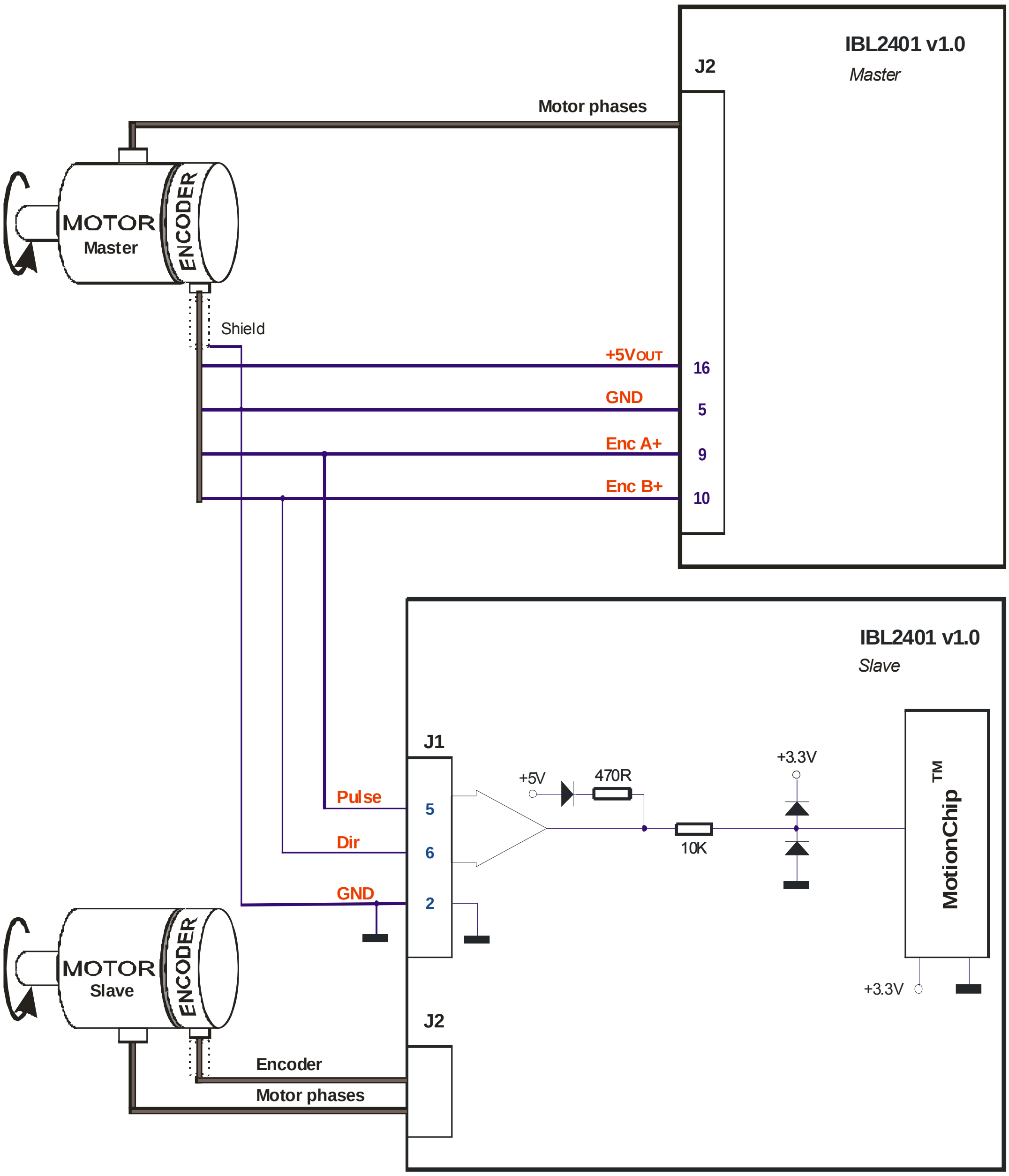


Figure 3.25. *Master – Slave encoder connection using second encoder input*

3.2.15. Connectors Type and Mating Connectors

Connector	Function	Producer	Board connector
J1	Motor & Feedback	Phoenix Contact	MPT 0,5/8 – 2,54[17]
J2	Supply, I/O & Serial	Phoenix Contact	MPT 0,5/8 – 2,54

[17] The mating connector accepts wires of 0.14 … 0.5 mm^2 (AWG26 … AWG20)

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 includes a firmware programmer through which you can update your drive firmware to the latest revision.

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

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

4.2. Getting Started with EasySetUp

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

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

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.

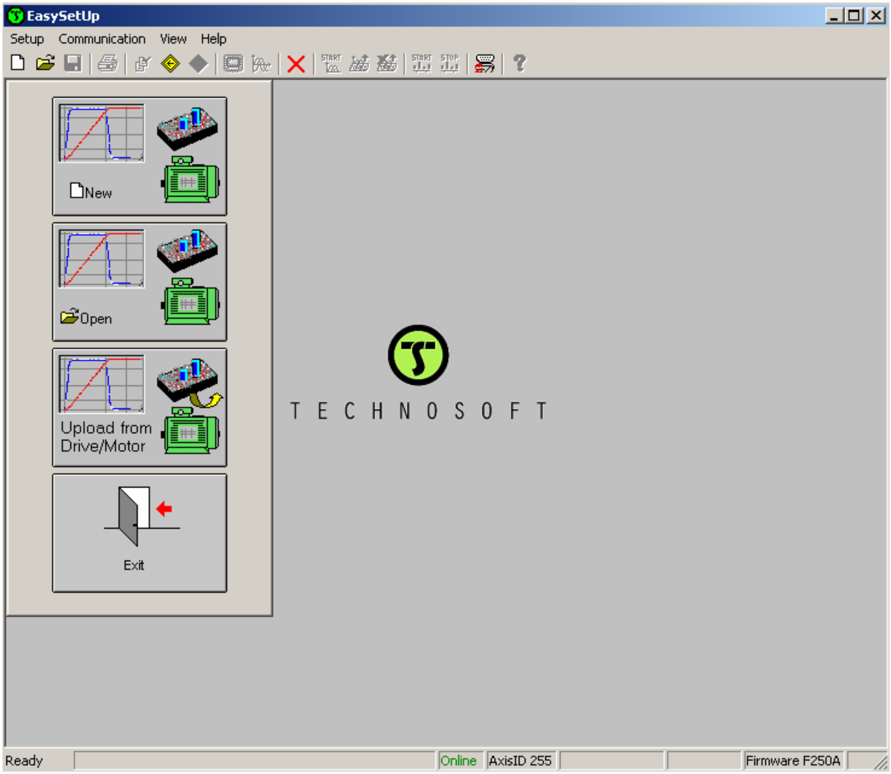
Note that with EasySetUp you do only your drive/motor commissioning. For motion programming you have the following options:

- Use a **CANopen** master (for IBL2401 CANopen)
- 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

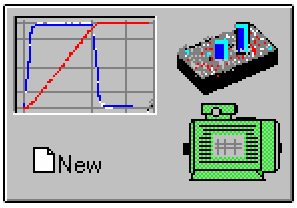
EasySetUp starts with an empty window from where you can create a **New** setup, **Open** a previously created setup which was saved on your PC, or **Upload** the setup from the drive/motor.



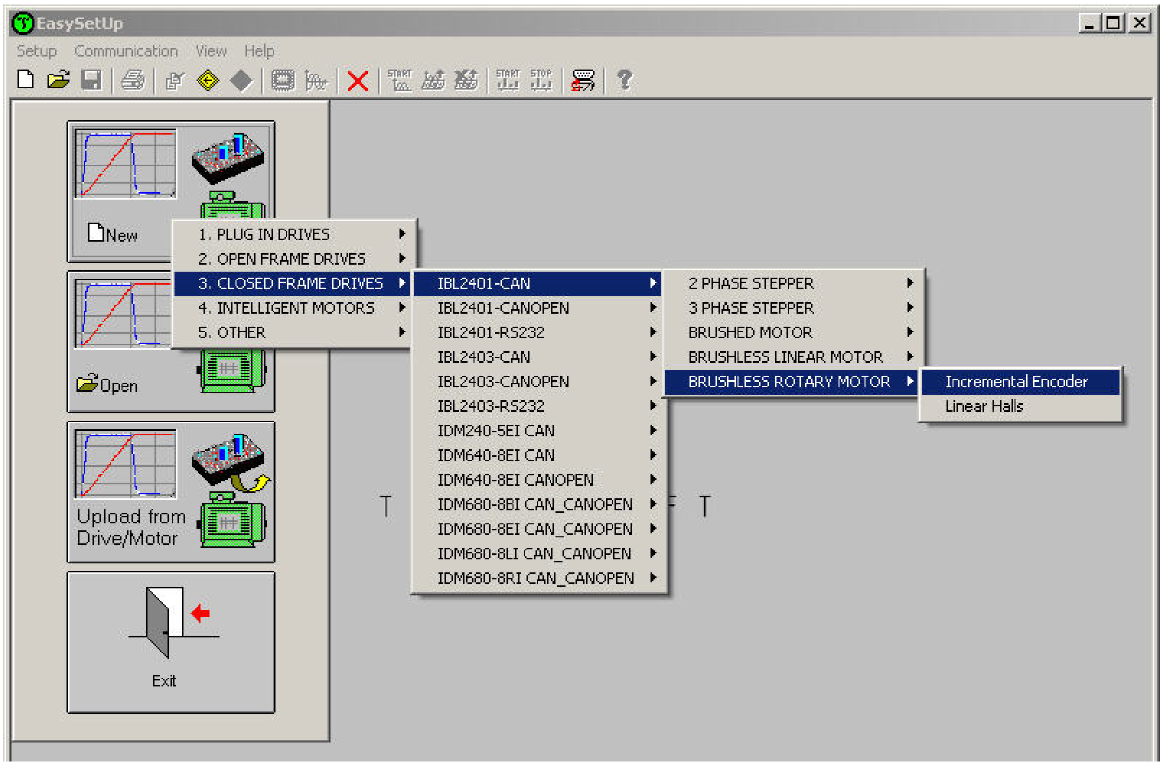
Before selecting one of the above options, you need to establish the communication with the drive you want to commission. Use menu command **Communication | Setup** to check/change your PC communication settings. Press the **Help** button of the dialogue opened. Here you can find detailed information 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 your drive is powered with all the DIP switches OFF and it is connected to your PC port COM1 via an RS-232 cable, the communication shall establish automatically. If the 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****.*

4.2.2. Setup drive/motor

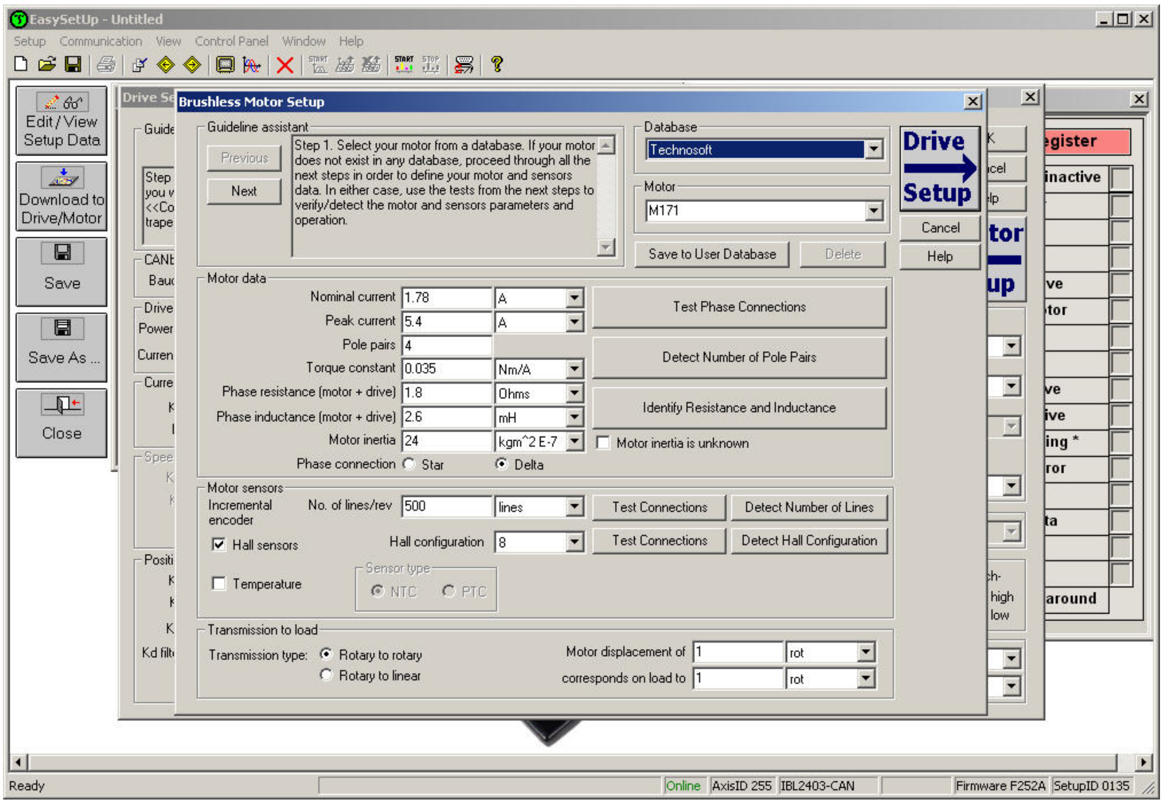
Press **New** button 

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).

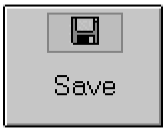
The selection opens 2 setup dialogues: for **Motor Setup** and for **Drive setup** through which you can configure and parameterize a Technosoft drive, plus several predefined control panels customized for the product selected.



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.

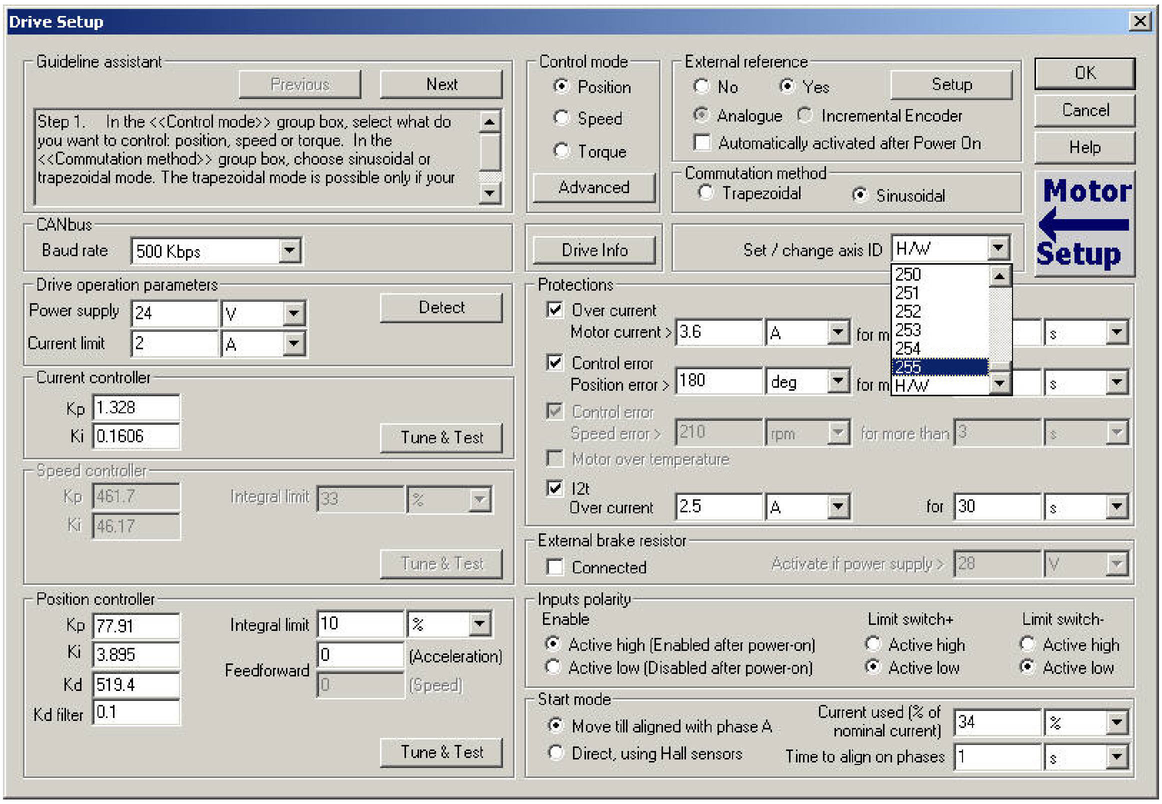
To summarize, you can define or change the setup data in the following ways:

- 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 behaviour (optional)

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

4.3. Changing the drive Axis ID



The axis ID of an IBL2401 drive can be set software – any value between 1 and 255, stored in the setup table.

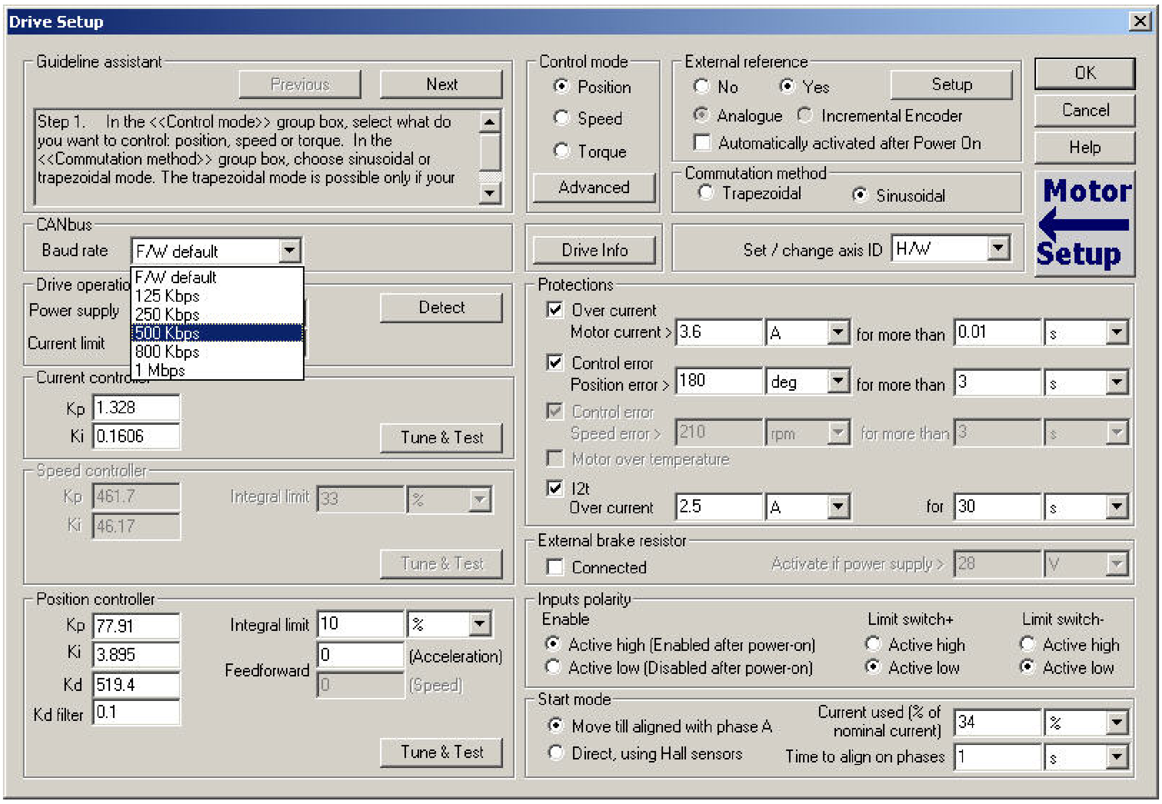
The axis ID is initialized at power on, using the following algorithm:

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

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

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

4.4. Setting CANbus rate



The IBL2401 drives can work with the following rates on the CAN: 125kHz, 250kHz, 500KHz, 1MHz. In the Drive Setup dialogue you can choose the initial CAN rate after power on. This information is stored in the setup table. The CAN rate is initialized using the following algorithm:

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

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

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

4.5. Creating an Image File with the Setup Data

Once you have validated your setup, you can create with the menu command **Setup | Create EEPROM Programmer File** a software file (with extension **.sw**) which contains all the setup data to write in the EEPROM of your 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 IBL2401 CANopen execution)

The IBL2401 drive supports the CiA draft standard **DS-301 v4.02** CANopen Application Layer and Communication Profile. It also conforms with the CiA draft standard proposal **DSP-402 v2.0** CANopen Device Profile for Drives and Motion Control. For details see CANopen Programming manual (part no. P091.063.UM.xxxx)

5.1.1. DS-301 Communication Profile Overview

The IBL2401 drive accepts the following basic services and types of communication objects of the CANopen communication profile DS 301 v4.02:

- **Service Data Object (SDO)**

Service Data Objects (SDOs) are used by CANopen master to access any object from the drive's Object Dictionary. Both expedited and segmented SDO transfers are supported (see DS301 v4.02 for details). SDO transfers are confirmed services. The SDOs are typically used for drive configuration after power-on, for PDOs mapping and for infrequent low priority communication between the CANopen master with 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 IBL2401 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 IBL2401 supports both SYNC consumer and producer.

- **Time Stamp Object (TIME)**

The Time Stamp Object is not supported by the IBL2401 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 IBL2401 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 IBL2401: 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. TechnoCAN Extension (for IBL2401 CAN execution)

In order to take full advantage of the powerful Technosoft Motion Language (TML) built into the IBL2401, 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 DS301 v4.02). Thus, TechnoCAN protocol and CANopen protocol can co-exist and communicate simultaneously on the same physical CAN bus, without disturbing each other.

5.1.3. DSP-402 and Manufacturer Specific Device Profile Overview

The IBL2401 supports the following CiA DSP402 v2.0 modes of operation:

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

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

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

5.1.4. Checking Setup Data Consistency

During the configuration phase, a CANopen master can quickly verify using the checksum objects and a reference **.sw** file (see 4.5 and 5.2.4 for details) whether the non-volatile EEPROM memory of an IBL2401 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[18], etc.)
- Change the motion modes and/or the motion parameters
- Execute homing sequences[19]
- Control the program flow through:
  - Conditional jumps and calls of TML functions
  - TML interrupts generated on pre-defined or programmable conditions (protections triggered, transitions on limit switch or capture inputs, etc.)
  - Waits for programmed events to occur
- Handle digital I/O and analogue input signals
- Execute arithmetic and logic operations
- Perform data transfers between axes
- Control motion of an axis from another one via motion commands sent between axes
- Send commands to a group of axes (multicast). This includes the possibility to start simultaneously motion sequences on all the axes from the group
- Synchronize all the axes from a network

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

5.2.2. Installing EasyMotion Studio

EasyMotion Studio is an integrated development environment for the setup and motion programming of Technosoft intelligent 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.

[18] Optional for IBL2401 CANopen execution
[19] The customization of the homing routines is available only for IBL2401 CAN execution

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:

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

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.

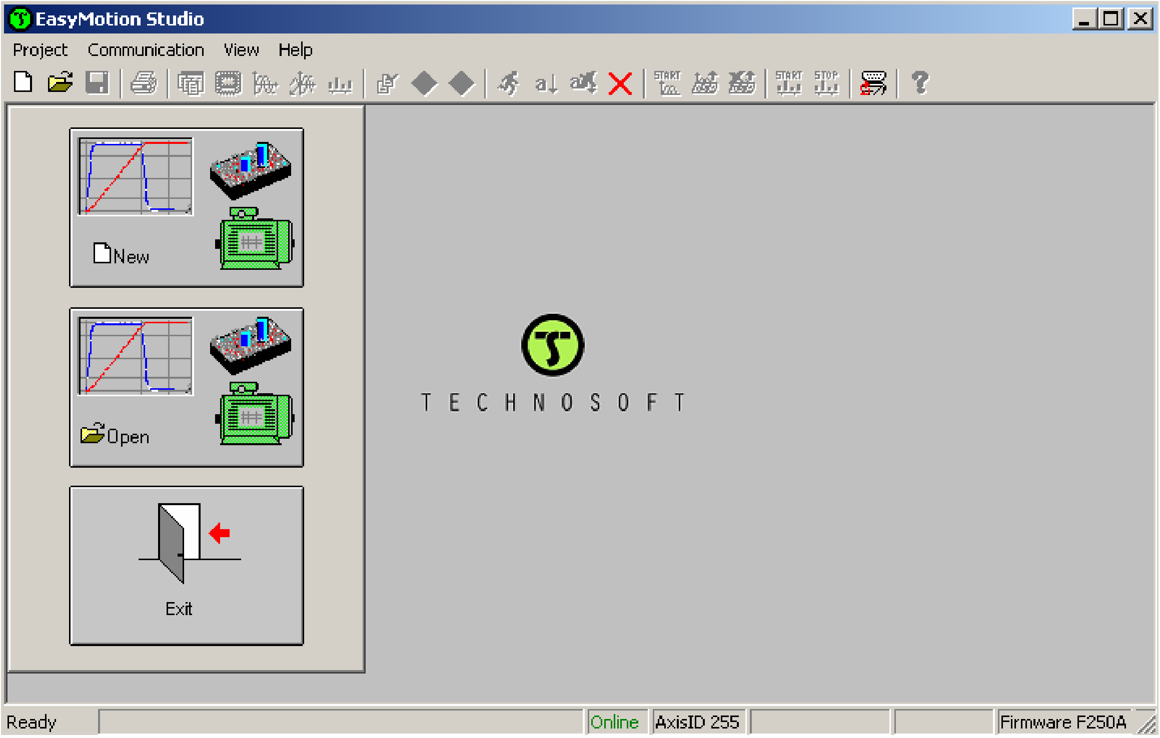
In the first case, each application has a different axis number corresponding to the axis ID of the drives/motors from the network. All data exchanges are done with the drive/motor having the same address as the selected application. In the second case, all the applications have the same axis number.

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.

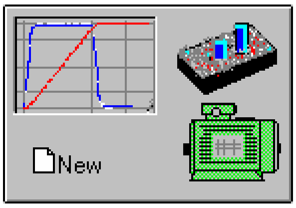
The motion component contains the motion sequences to do. These are described via a TML (Technosoft Motion Language) program, which is executed by the drives/motors built-in motion controller.

5.2.3.1 Create a new project

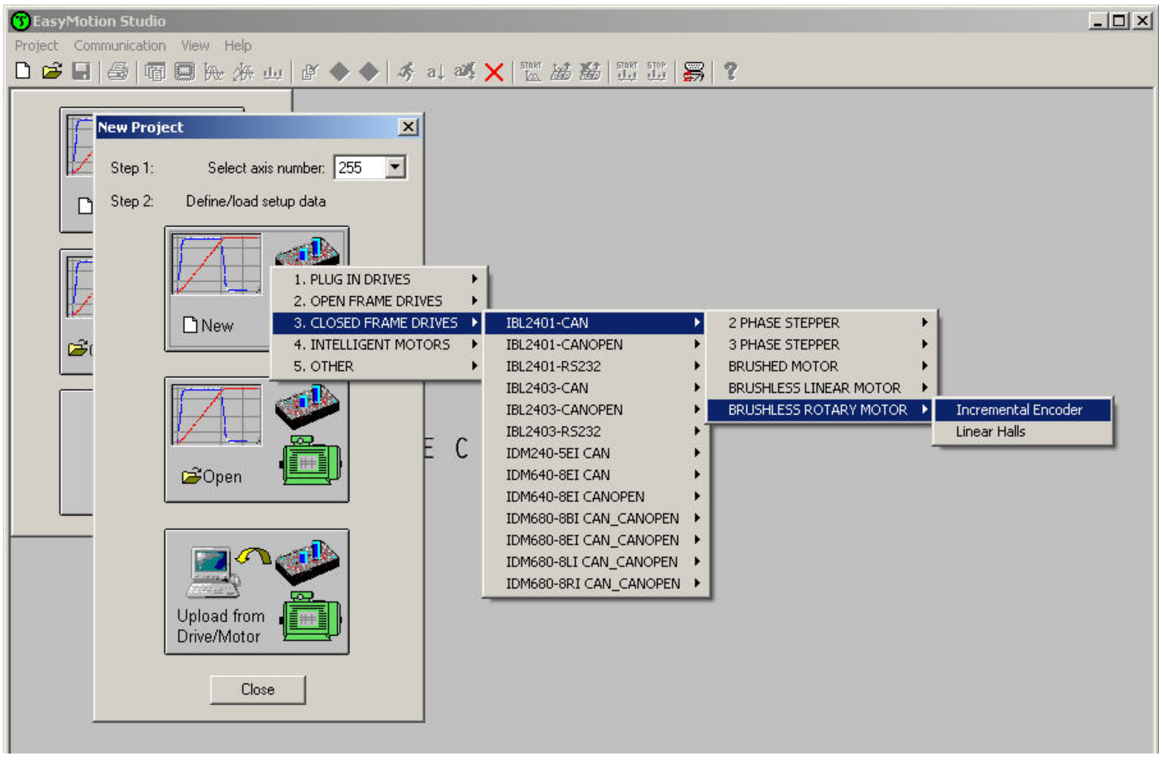
EasyMotion Studio starts with an empty window from where you can create a new project or open a previously created one.



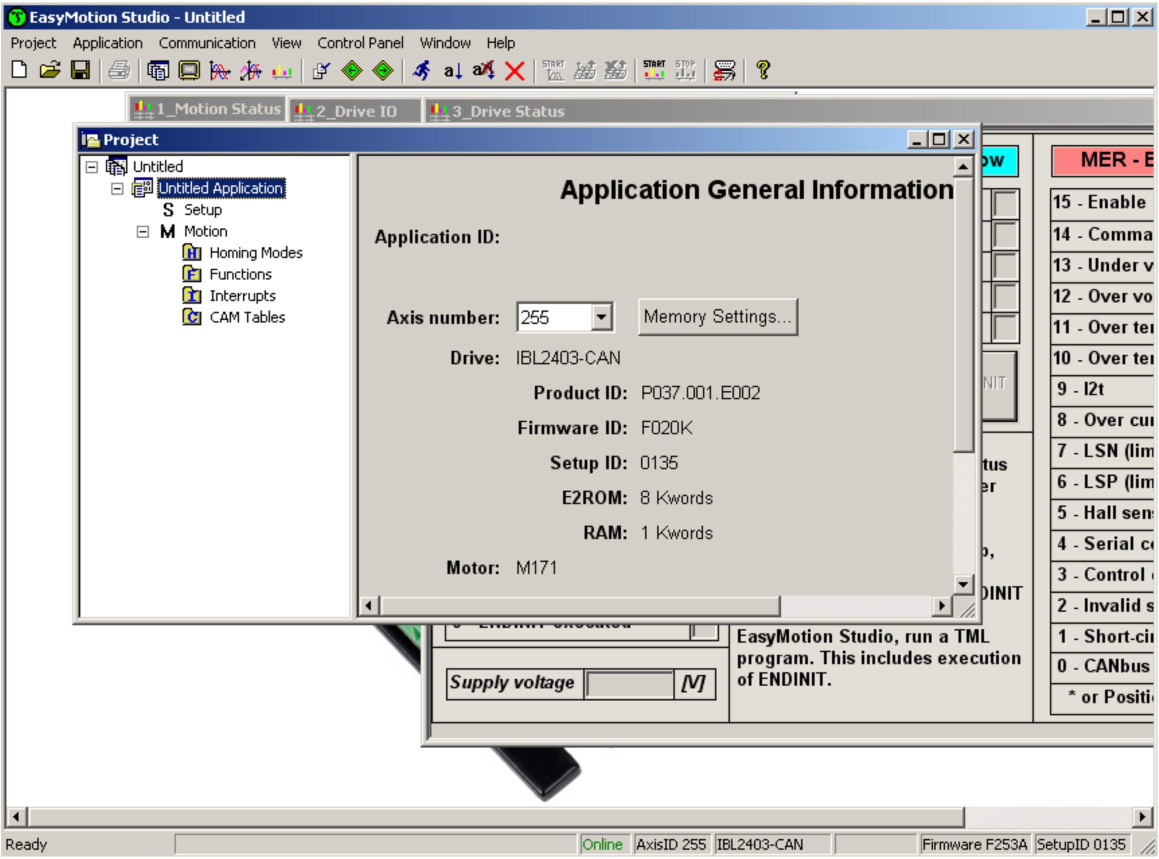
When you start a new project, EasyMotion Studio automatically creates a first application. Additional applications can be added later. You can duplicate an application or insert one defined in another project.

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).



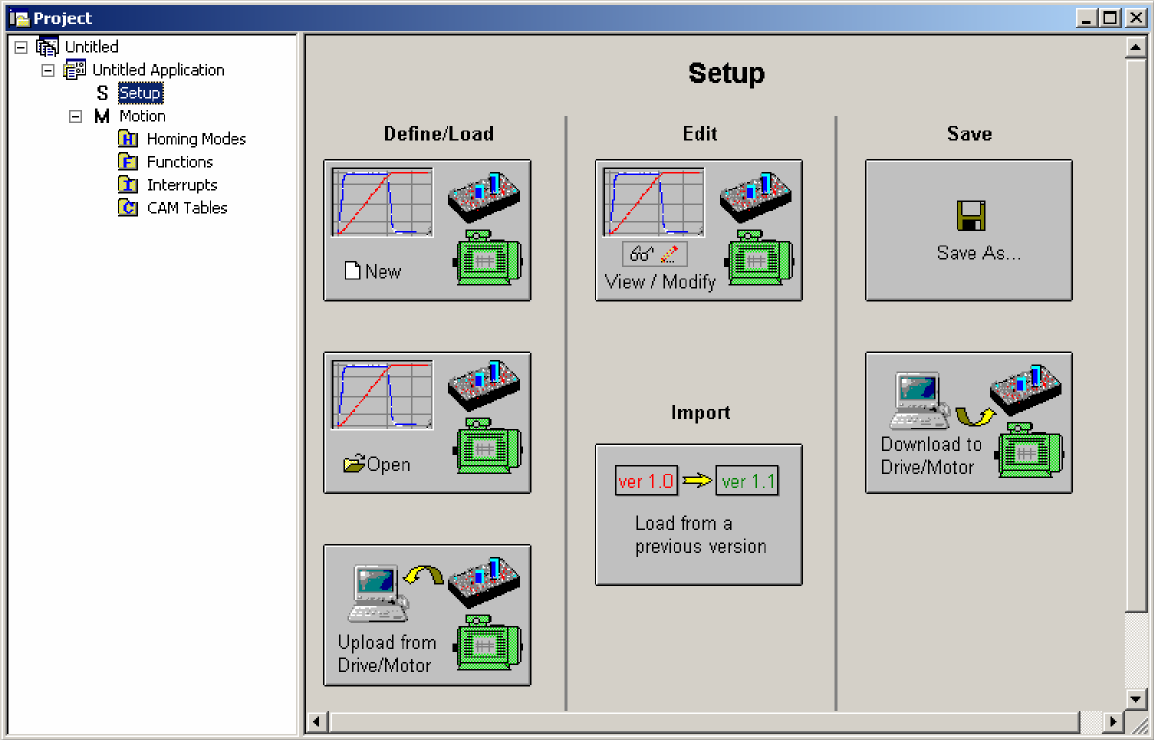
5.2.3.2 Step 2 Establish communication

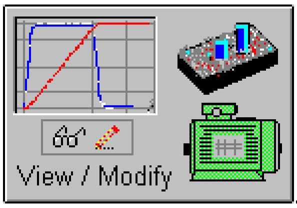
If you have a drive/motor connected with your PC, now its time to check the communication. Use menu command **Communication | Setup** to check/change your PC communication settings. Press the **Help** button of the dialogue opened. Here you can find detailed information about how to setup your drive/motor 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 your drive is powered with all the DIP switches OFF and it is connected to your PC port COM1 via an RS-232 cable, the communication shall establish automatically.*

5.2.3.3 Setup drive/motor

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



Press **View/Modify** button 

. This opens 2 setup dialogues: for **Motor Setup** and for **Drive Setup** (same like on EasySetUp) through which you can configure and parameterize a Technosoft 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 

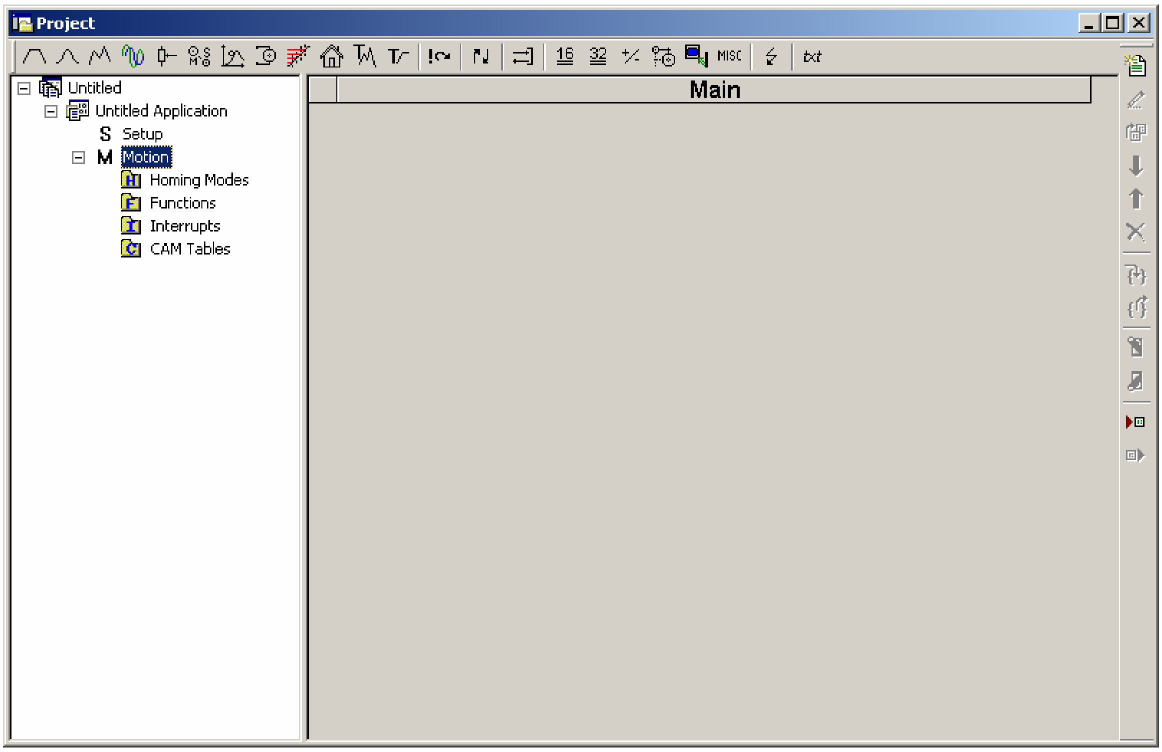
to download your setup data in the drive/motor EEPROM memory in the *setup table*. From now on, at each power-on, the setup data is copied into the drive/motor RAM memory which is used during runtime. It is also possible to save the setup data on your PC and use it in other applications. Note that you 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[20]. The TML program may also include cam tables used for electronic camming applications[21].

[20] The customization of the interrupt service routines and homing routines is available only for IBL2401 CAN executions
[21] Optional for IBL2401 CANopen execution

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 4.5

5.3. Combining CANopen /or other host with TML

Due to its embedded motion controller, an IBL2401 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

Remark: *If you don't use the advanced features presented below you don't need EasyMotion Studio. In this case the IBL2401 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 on step further. You may prepare and download into a drive a complete TML program including functions, homing procedures[22], 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 IBL2401 offers others motion modes like[23]: 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.

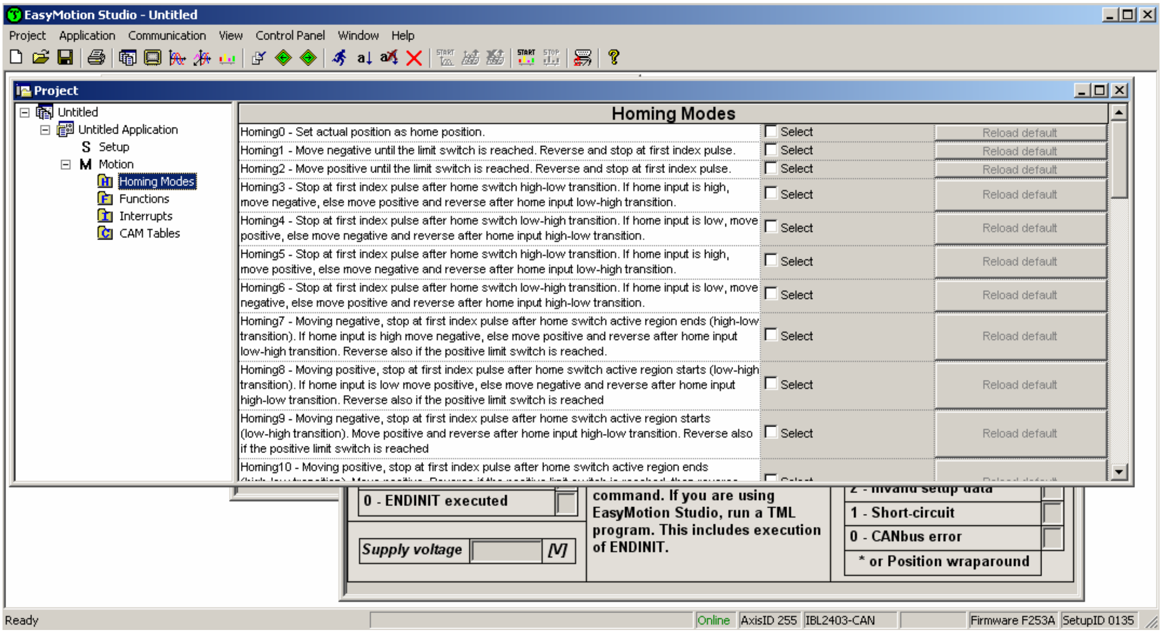
5.3.4. Customizing the Homing Procedures (for IBL2401 CAN executions)

The IBL2401 supports all homing modes defined in DSP-402 device profile. If needed, any of these homing modes can be customized. In order to do this you need to select the Homing

[22] The customization of the interrupt service routines and homing routines is available only for IBL2401 CAN executions

[23] Optional for the IBL2401 CANopen execution

Modes from your EasyMotion Studio application and in the right side to set as "User defined" one of the Homing procedures. Following this operation the selected procedure will occur under Homing Modes in a subtree, with the name *HomeX* where X is the number of the selected homing.



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

5.3.5. Customizing the Drive Reaction to Fault Conditions (for IBL2401 CAN executions)

Similarly to the homing modes, the default service routines for the TML interrupts can be customized according to your application needs. However, as most of these routines handle the 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 developmemnt 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**[24] is:

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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:

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

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

where:

resolution – is the motor position resolution

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

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:

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

where:

No_encoder_lines – is the encoder number of lines per revolution

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

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

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

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

Stepper 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** [25] is:

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

where:

No_encoder_lines – is the motor encoder number of lines per revolution

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

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

6.1.5. Stepper 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:

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

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

6.2. 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:

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

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**[26] 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. Stepper 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:

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

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

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”

Stepper 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]

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. Stepper 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**[27] 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”.

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

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:

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

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**[28] 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

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

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:

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

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. Stepper 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:

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

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. Stepper 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:

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

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. Stepper 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”

6.4. 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**[29] 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 rotary encoder number of lines per revolution

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

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: $$\text{Load\_Jerk[SI]} = \frac{2 \times \pi}{\text{resolution} \times \text{Tr} \times \text{T}^3} \times \text{Motor\_Jerk[IU]}$$

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

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:

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

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. Stepper 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**[30] is:

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

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. Stepper 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:

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

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. Stepper 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:

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

where:

No_encoder_lines – is the motor encoder number of lines per revolution

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

[30] SI units for jerk are [rad/s^3] for a rotary movement, [m/s^3] 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”.

6.5. Current units

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

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

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:

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

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:

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

This correspondence is akso available for **DC brushed** motors which have the voltage command internal units as the brushless motors 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:

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

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

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

6.8. Time units

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

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

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

6.9. Drive temperature units

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

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

where:

Sensor_gain – is the temperature sensor gain

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

6.10. Master position units

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

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

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

where:

No_encoder_lines – is the master number of encoder lines per revolution

6.11. Master speed units

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

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

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

where:

No_encoder_lines – is the master number of encoder lines per revolution

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

6.12. Motor position units

6.12.1. 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**[31] is:

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

6.12.2. 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])

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

The motor position is not computed.

6.12.4. Stepper 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:

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

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

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.

6.12.5. Stepper motor open-loop control. Incremental encoder on load

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

6.12.6. Stepper 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

6.13. Motor speed units

6.13.1. 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:

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

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”

6.13.2. 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])

6.13.3. 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**[32] is:

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

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]

6.13.4. 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:

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

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]

6.13.5. Stepper 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:

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

where:

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

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”

6.13.6. Stepper 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”.

7. Memory Map

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

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

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

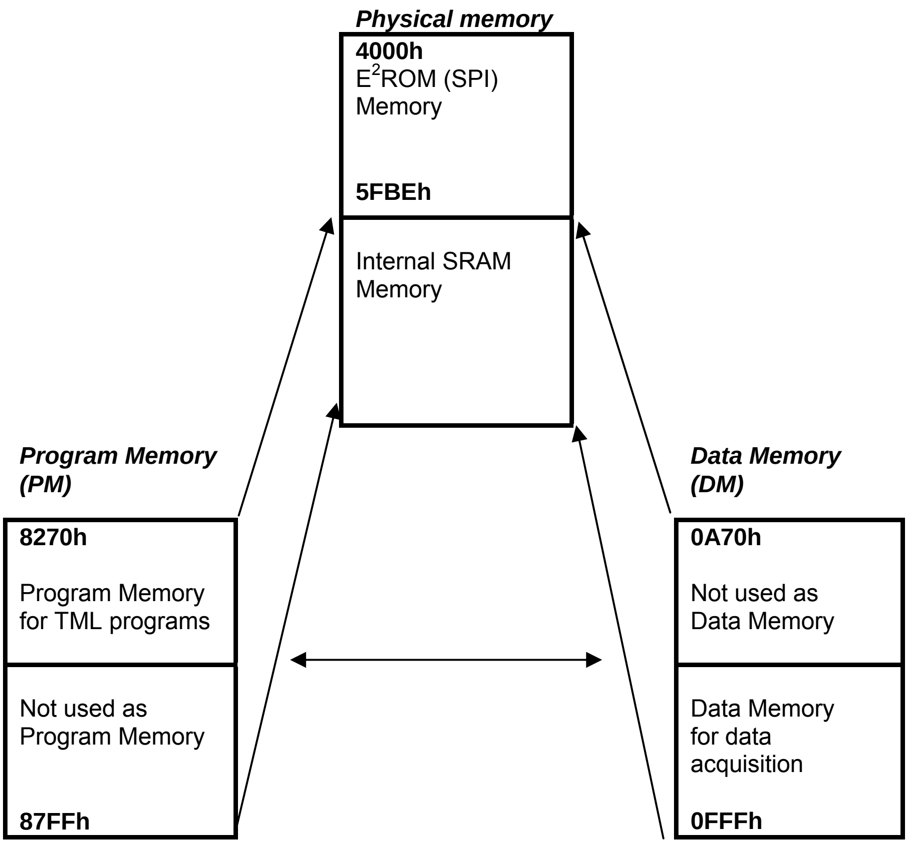


Figure 8.1. *IBL2401 / IBL2401-CAN Memory Map*

This page is empty

T E C H N O S O F T