IDM680

Intelligent
Servo Drive

TECHNOSOFT

Intelligent Drives

Technical Reference

TECHNOSOFT

IDM680

Technical Reference

P091.048.IDM680.UM.1211

Technosoft S.A.

Avenue des Alpes 20

CH-2000 NEUCHATEL

Switzerland

Tel.: +41 (0) 32 732 5500

Fax: +41 (0) 32 732 5504

contact@technosoftmotion.com

www.technosoftmotion.com

Read This First

Whilst Technosoft believes that the information and guidance given in this manual is correct, all parties must rely upon their own skill and judgment when making use of it. Technosoft does not assume any liability to anyone for any loss or damage caused by any error or omission in the work, whether such error or omission is the result of negligence or any other cause. Any and all such liability is disclaimed.

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

About This Manual

This book is a technical reference manual for the **IDM680** family of intelligent servo drives, including the following products: **IDM680-8EI**, **IDM680-8LI**, **IDM680-8RI, IDM680-8BI**. In order to operate the IDM680 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**
  - 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 IDM680 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

IDM680 – all products described in this manual

Related Documentation

Help of the EasySetUp software – describes how to use **EasySetUp** to quickly setup any Technosoft drive for your application using only 2 dialogues. The output of EasySetUp is 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.IDM680.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.	Mail: Technosoft SA Buchaux 38 CH-2022 Bevaix, NE Switzerland

This page is empty

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 **IDM680** drives are the new members of the IDM family of fully digital intelligent servo drives. Based on the latest DSP technology, they offer unprecedented performance combined with a *CANopen communication interface*.

Suitable for control of brushless DC, brushless AC (vector control), DC brushed motors and step motors, the IDM680 drives accept as position feedback incremental encoders (quadrature or sine/cosine), absolute encoders (SSI for brushless AC or DC brushed motors; BiSS or sine/cosine with EnDat for brushless AC motors), linear Halls signals and resolver (for brushless motors).

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 IDM680 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, etc.)
- Changing the motion modes and/or the motion parameters
- Executing homing sequences
- Controlling the program flow through:
  - Conditional jumps and calls of TML functions
  - TML interrupts generated on pre-defined or programmable conditions (protections triggered, transitions on limit switch or capture inputs, etc.)
  - Waits for programmed events to occur
- Handling of digital I/O and analogue input signals
- Executing arithmetic and logic operations
- Performing data transfers between axes
- Controlling motion of an axis from another one via motion commands sent between axes
- Sending commands to a group of axes (multicast). This includes the possibility to start simultaneously motion sequences on all the axes from the group
- Synchronizing all the axes from a network

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 IDM680 drives can also be controlled from a PC or PLC using the family of **TML_LIB** motion libraries.

For all motion programming options, the IDM680 commissioning for your application is done using **EasySetUp**.

2.2. Key Features

- Digital drives for control of brushless DC, brushless AC, DC brushed and step motors with built-in motion 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
  - External analogue or digital reference
  - 33 Homing modes
- Incremental encoder and digital Hall sensors interfaces: 5V single-ended, open-collector or RS-422 differential (IDM680-8EI)
- Absolute SSI encoder interface: RS-422 differential (IDM680-8EI)
- Absolute BiSS (sensor mode) encoder interface: RS-422 differential (IDM680-8BI)
- Linear Hall sensors interface: 4Vp-p (IDM680-8LI)
- Incremental or absolute sine/cosine encoder: 1Vp-p (IDM680-8LI)
- Resolver interface (IDM680-8RI)
- Second incremental encoder / pulse & direction interface (5V or 24V single-ended, open-collector or RS-422 differential) for external (master) digital reference
- Digital I/Os:
  - 6 inputs 24V, opto-isolated, common I/O ground: 2 general-purpose, 2 for limit switches, 2 for Reset and Enable (emergency shutdown)
  - 2 inputs 24V / 5V compatible (shared with second encoder / pulse & direction)
  - 6 digital outputs, opto-isolated, 24V PNP-type, 80/160 mA, short-circuit protected: 4 general-purpose, 2 for Ready and Error
- 2 differential analog inputs +/-10 V, for reference and feedback
- Compact design: 136 x 95 x 26 mm
- RS-232 serial communication up to 115kbaud
- CAN-bus 2.0A / 2.0B up to 1Mbit/s, opto-isolated, with selectable communication protocol:
  - CANopen – compatible with CiA standards: DS301 and DSP402
  - TMLCAN – compatible with all Technosoft drives with CANbus interface
- Motor temperature sensor interface
- 4K×16 SRAM for data acquisitions and 8K×16 E^2ROM for setup data and TML programs

- Nominal PWM switching frequency[1]: 20 kHz
- Nominal update frequency for torque loop[1]: 10 kHz
- Update frequency for speed/position loop[2]: 1-10 kHz
- Continuous output current: $8A_{RMS}$
- Peak output current: 16.5A
- Logic power supply: 12÷48 VDC
- Motor power supply: 12÷80 VDC
- Minimal load inductance: 50μH @12V, 200 μH @ 48 V, 330 μH @80V
- Operating ambient temperature[3]: 0-40°C

2.3. Supported Motor-Sensor Configurations

2.3.1. IDM680-8EI

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[4], while the same commands, expressed in IU units, refer to the motor.

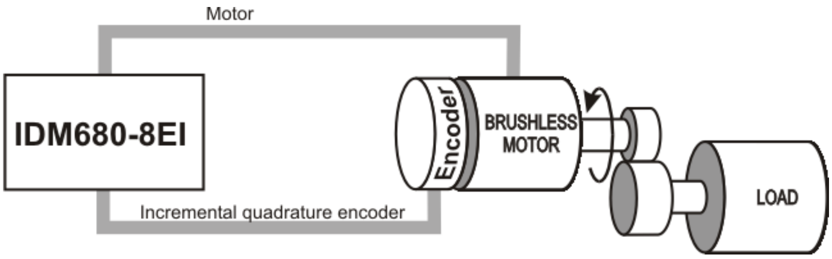


Figure 2.1. *Brushless AC rotary motor. Position/speed/torque control. Quadrature encoder on motor.*

2. Position, speed or torque control of a **brushless AC linear motor** with an **incremental quadrature encoder**. 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, while the same commands, expressed in IU units, refer to the motor.

---

[1] Nominal values cover all cases. Higher values are possible in specific configurations. For details contact Technosoft
[2] 1-2kHz cover all cases. Higher values equal with torque loop update frequency are possible with quadrature encoders
[3] For higher ambient temperatures, contact Technosoft to get derating information
[4] Motion commands can be referred to the motor by setting in EasySetUp a rotary to rotary transmission with ratio 1:1

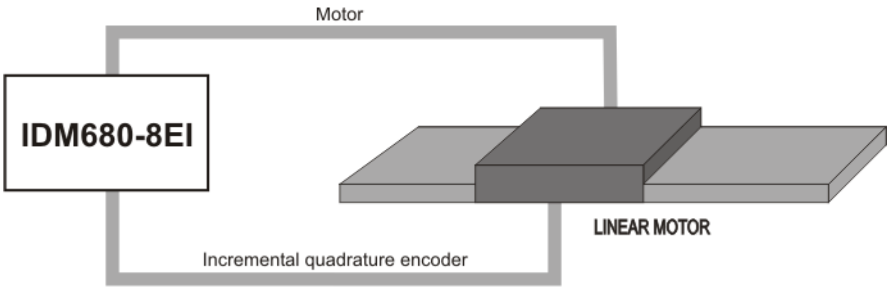


Figure 2.2. *Brushless AC linear motor. Position/speed/torque control.*
Quadrature encoder on motor.

3. 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[1], while the same commands, expressed in IU units, refer to the motor.

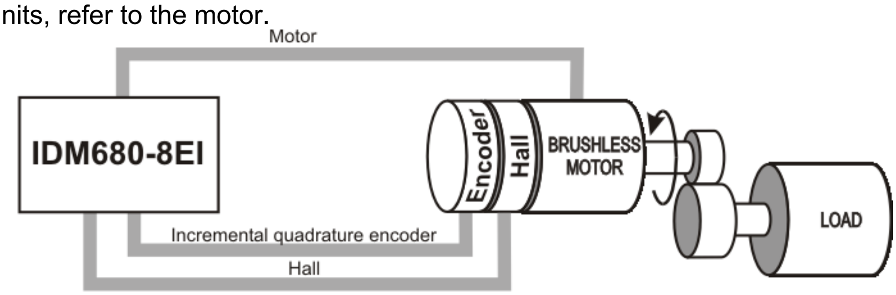


Figure 2.3. *Brushless DC rotary motor. Position/speed/torque control.*
Hall sensors and quadrature encoder on motor.

4. Position, speed or torque control of a **brushless DC linear motor** with **digital Hall sensors** and an **incremental quadrature encoder**. 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.

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

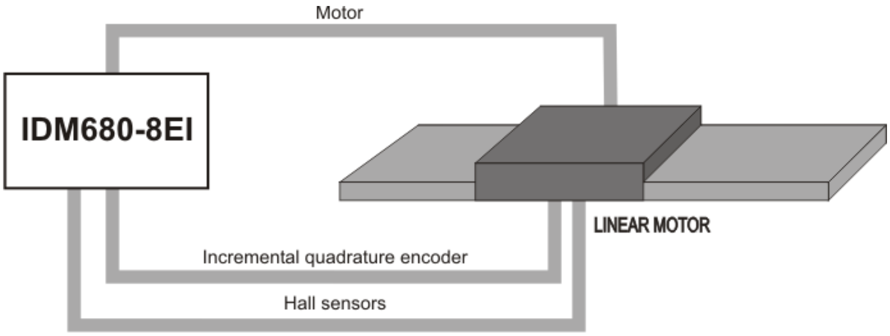


Figure 2.4. *Brushless DC linear motor. Position/speed/torque control.*
Hall sensors and quadrature encoder on motor.

5. Position, speed or torque control of a **brushless AC rotary motor** with an **absolute SSI 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[1], while the same commands, expressed in IU units, refer to the motor.

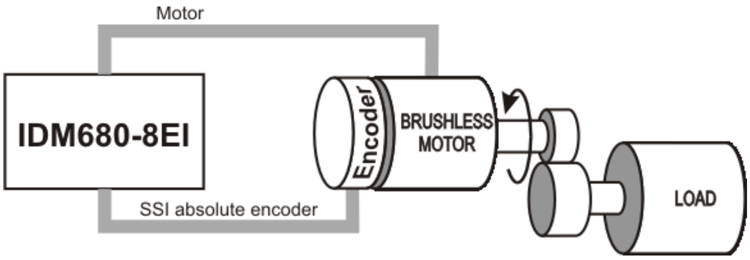


Figure 2.5. *Brushless AC rotary motor. Position/speed/torque control.*
SSI encoder on motor.

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.

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

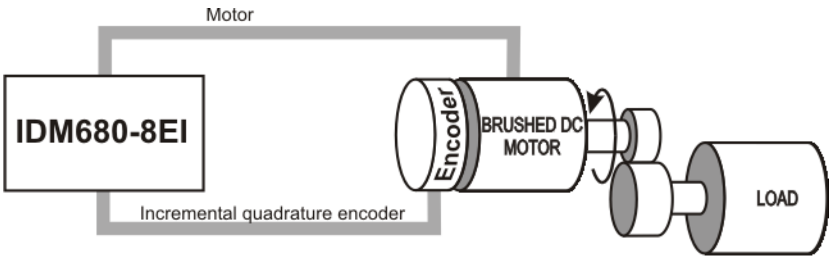


Figure 2.6. *DC brushed rotary motor. Position/speed/torque control.*
Quadrature encoder on motor.

7. 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

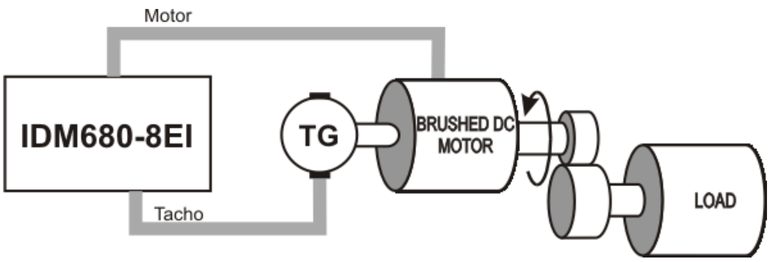


Figure 2.7. *DC brushed rotary motor. Speed/torque control.*
Tachometer on motor.

8. 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

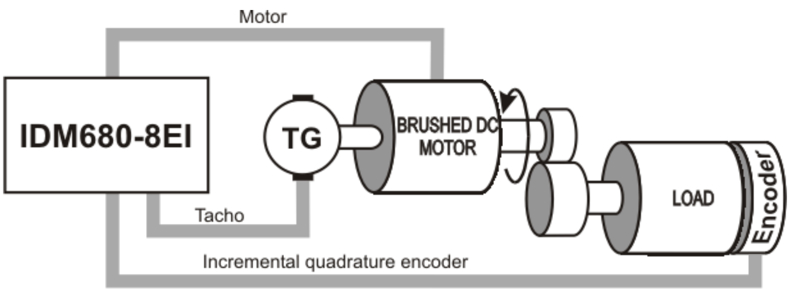


Figure 2.8. *DC brushed rotary motor. Position/speed/torque control.*
Quadrature encoder on load plus tachometer on motor.

9. Load position control using an **absolute SSI 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

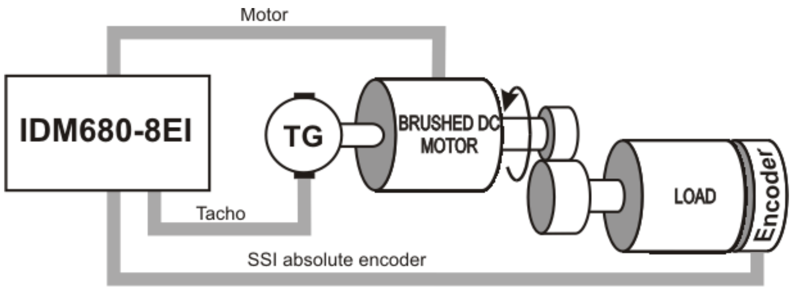


Figure 2.9. *DC brushed rotary motor. Position/speed/torque control. Absolute SSI encoder on load plus tachometer on motor.*

10. 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.

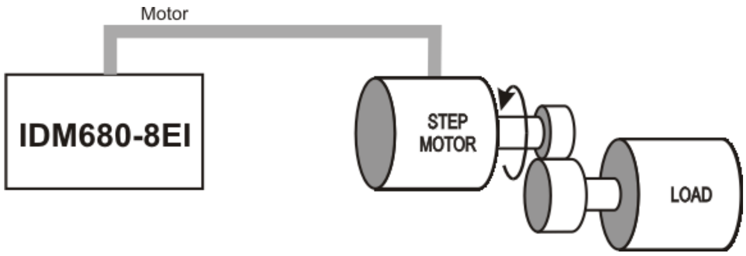


Figure 2.10. *No position or speed feedback. Open-loop control: motor position or speed.*

11. 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.

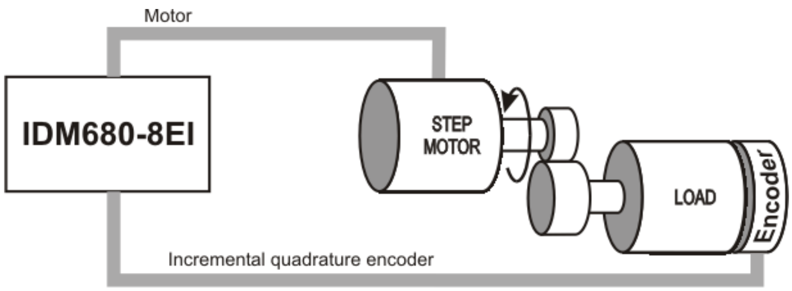


Figure 2.11. *Encoder on load. Closed-loop control: load position, open-loop control: motor speed.*

12. 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[1], while the same commands, expressed in IU units refer to the motor.

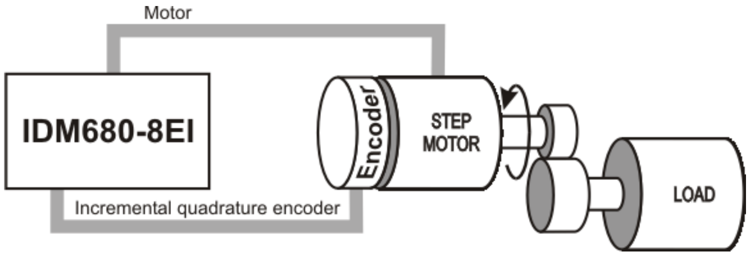


Figure 2.12. *Encoder on motor shaft.*
Closed-loop control: motor position, speed or torque.

2.3.2. IDM680-8LI

1. Position, speed or torque control of a **brushless AC rotary motor** with **linear Hall signals**.

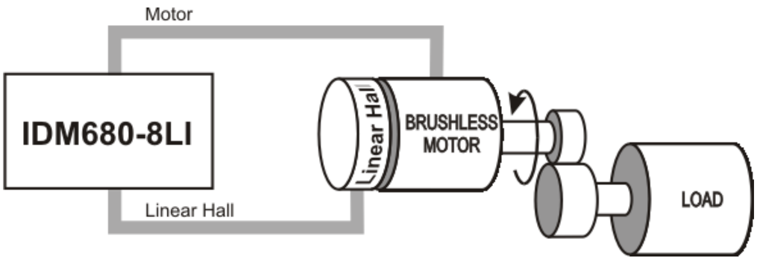


Figure 2.13. *Brushless AC rotary motor with linear Hall signals.*
Position/speed/torque control.

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[1], while the same commands, expressed in IU units, refer to the motor.

2. Position, speed or torque control of a **brushless AC rotary motor** with an **incremental sine/cosine 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[1], while the same commands, expressed in IU units, refer to the motor.

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

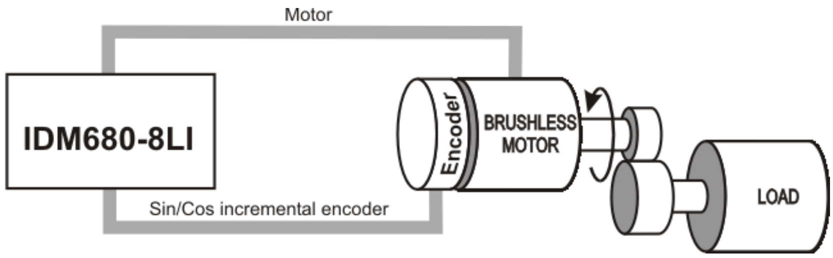


Figure 2.14. *Brushless AC rotary motor. Position/speed/torque control.*
Sine/cosine incremental encoder on motor.

3. Position, speed or torque control of a **brushless AC linear motor** with an **incremental sine/cosine encoder**. 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, while the same commands, expressed in IU units, refer to the motor.

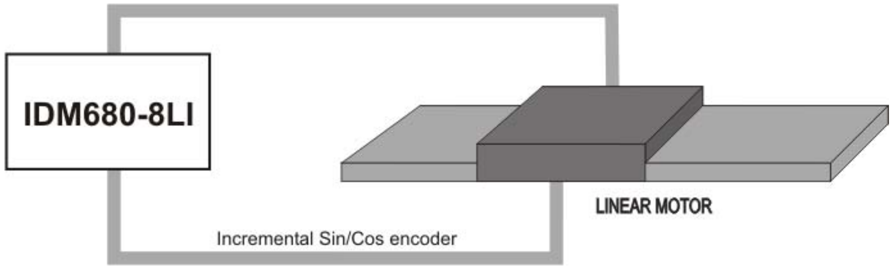


Figure 2.15. *Brushless AC linear motor. Position/speed/torque control.*
Sine/cosine incremental encoder on motor.

4. Position, speed or torque control of a **brushless AC rotary motor** with an **EnDat absolute sine/cosine 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[1], while the same commands, expressed in IU units, refer to the motor.

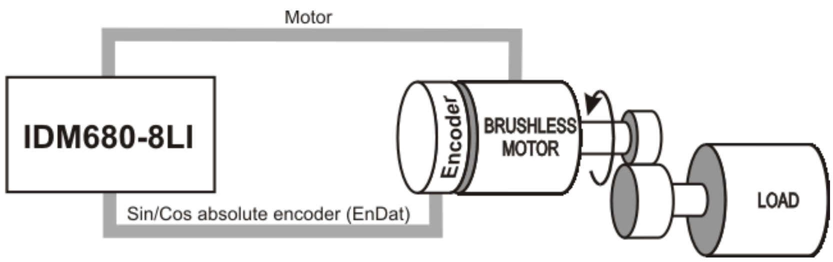


Figure 2.16. *Brushless AC rotary motor. Position/speed/torque control.*
EnDat absolute sine/cosine encoder on motor.

2.3.3. IDM680-8RI

1. Position, speed or torque control of a **brushless AC rotary motor** with a **resolver** 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[1], while the same commands, expressed in IU units, refer to the motor.

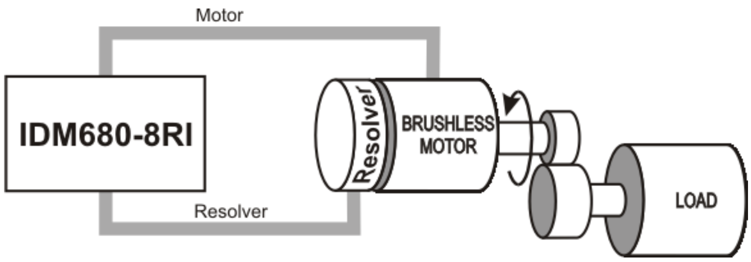


Figure 2.17. *Brushless AC rotary motor.*
Position/speed/torque control. Resolver on motor.

2.3.4. IDM680-8BI

1. Position, speed or torque control of a **brushless AC rotary motor** with an **absolute BiSS 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, while the same commands, expressed in IU units, refer to the motor.

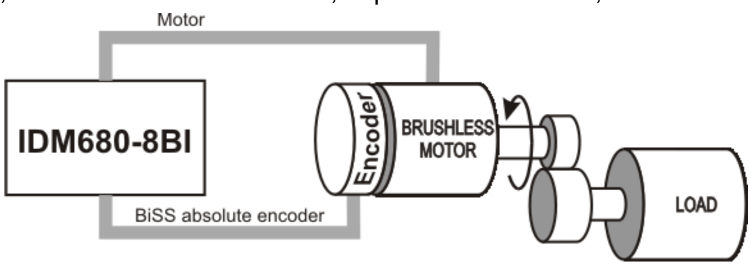


Figure 2.18. *Brushless AC rotary motor.*
Position/speed/torque control BiSS encoder on motor.

2.4. IDM680 Dimensions

The next figure presents the IDM680 drives dimensions.

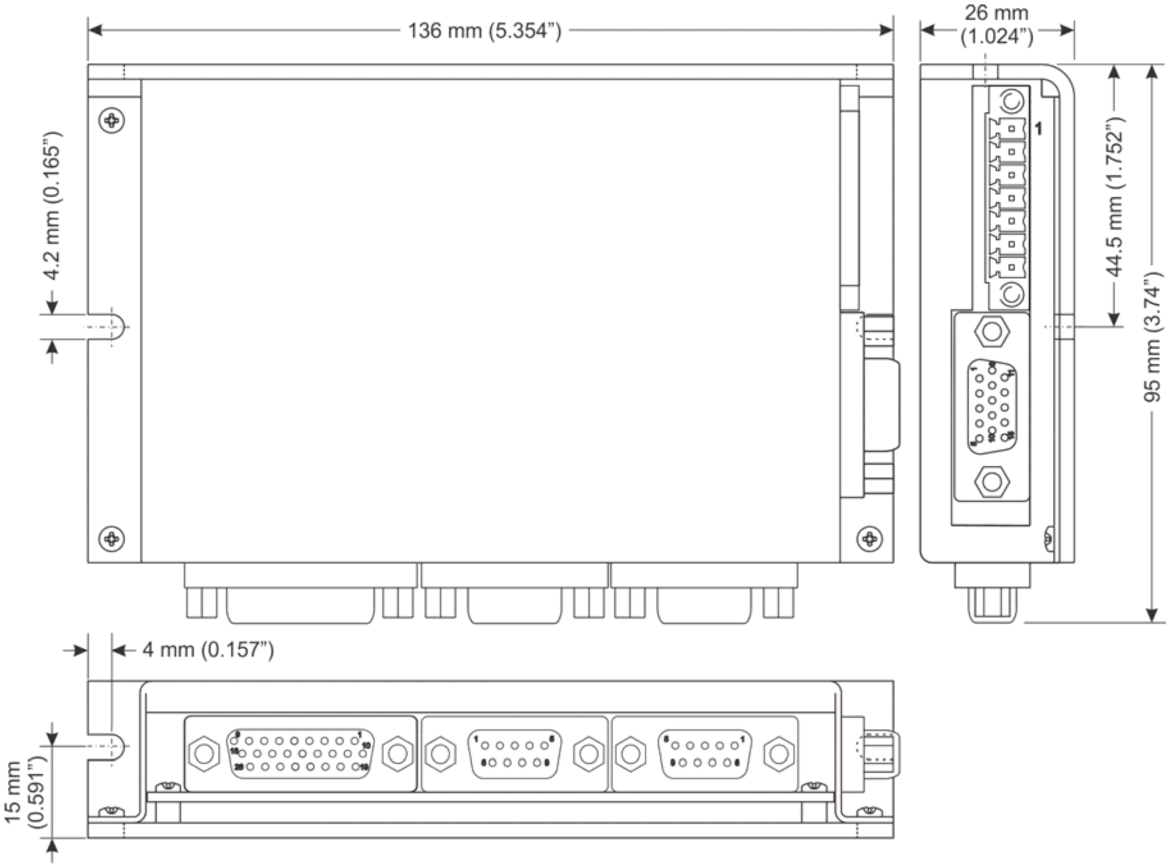


Figure 2.19. *IDM680 drives dimensions*

2.5. Electrical Specifications

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

T_{amb} = 0…40°C, V_{LOG} = 24 V_{DC}, $V_{24\ VPLC}$ = 24 V_{DC}, CAN_V+ = 24 V_{DC}; V_{MOT} = 80V_{DC}; Load current = 8A_{RMS}; Supplies start-up / shutdown sequence: -*any*-;

Logic Supply Input

	Measured between +V_{LOG} and GND.	Min.	Typ.	Max.	Units
Supply voltage	Nominal values, including ripple	12	24	48	V_{DC}
	Absolute maximum values, continuous	8		+51	V_{DC}
	Absolute maximum values, surge (duration ≤ 10ms) †	-100		+60	V
Supply current	+V_{LOG} = 12V		350	400	mA
	+V_{LOG} = 24 V		150	250	mA
	+V_{LOG} = 48 V		80	150	mA

Motor Supply Input

	Measured between +V_{MOT} and GND.	Min.	Typ.	Max.	Units
Supply voltage	Nominal values, including ripple & braking-induced over-voltage up to ±25%	12		80	V_{DC}
	Absolute maximum values, continuous	0		100	V_{DC}
	Absolute maximum values, surge (duration ≤ 10ms) †	-0.5		105	V
Supply current	Idle		0.5	1.5	mA
	Operating			16.5	A
	Absolute maximum values, surge (duration ≤ 10ms) †			100	A

I/O Supply Input (isolated)

	Measured between +24 V_{PLC} and 0V_{PLC}.	Min.	Typ.	Max.	Units
Supply voltage	Nominal values	8	24	30	V_{DC}
	Absolute maximum values, surge (duration ≤ 10ms) †	-100		32	V
Supply current	All inputs and outputs disconnected		20	30	mA
	All inputs tied to +24 V_{PLC}; all outputs sourcing simultaneously their nominal current into external load(s)		700	1000	mA
Isolation voltage rating	Between 0V_{PLC} and GND			200	V_{RMS}

CAN-Bus Supply Input (isolated)

	Measured between CAN_V+ and CAN_GND.	Min.	Typ.	Max.	Units
Supply voltage	Nominal values	8	24	30	V_{DC}
	Absolute maximum values, surge (duration ≤ 10ms) †	-75		32	V
Supply current	CAN-Bus idle		12	25	mA
	CAN-Bus operating at 1Mbit/s		60	180	mA
Isolation voltage rating	Between CAN_GND and drive GND			200	V_{RMS}

Motor Outputs

	All voltages referenced to GND.	Min.	Typ.	Max.	Units
Motor output current	Continuous operation	-8		+8	A_{RMS}
Motor output current, peak		-16.5		+16.5	A
Short-circuit protection threshold		±26	±25	±29	A
Short-circuit protection delay		12	15		μS
On-state voltage drop	Output current = ±8 A	-1100	±250	+600	mV
Off-state leakage current		-1	±0.1	+1	mA
Motor inductance	F_{PWM} = 20 kHz, $+V_{MOT}$ = 12 V	50			μH
	F_{PWM} = 20 kHz, $+V_{MOT}$ = 48 V	200			μH
	F_{PWM} = 20 kHz, $+V_{MOT}$ = 80 V	400			μH

24 V Digital Inputs (opto-isolated)

	All voltages referenced to $0V_{PLC}$.	Min.	Typ.	Max.	Units
Input voltage	Logic "LOW"	-5	0	1.2	V
	Logic "HIGH"	8	24	30	
	Absolute maximum, surge (duration ≤ 1s) †	-30		+80	
Input current	Logic "HIGH"	2.5	10	15	mA
	Logic "LOW"	0		0.2	
Input frequency		0		5	kHz
Minimum pulse width	Pulse "LOW"-"HIGH"-"LOW"	10			μs
	Pulse "HIGH"-"LOW"-"HIGH"	100			μs

Pulse / Direction / Master Encoder Inputs

		Min.	Typ.	Max.	Units
Single-Ended mode compliance	IN+; Leave IN- disconnected	TTL / CMOS / open-collector			
	IN-; Leave IN+ disconnected	24V referenced to GND			

Differential Mode Compliance	Both IN+, IN- driven; for full RS-422 compliance, see [1]	TIA / EIA – 422			
Input voltage	IN+; Logic “LOW”	-7	0	1.2	V
	IN+; Logic “HIGH”	1.8	5	12	
	IN-; Logic “LOW”	-7	0	4.6	
	IN-; Logic “HIGH”	5.4	24	30	
	Absolute maximum, surge (duration ≤ 1s) †	-12		32	
	Differential input hysteresis	±0.1	±0.2	±0.4	
	Common-mode range (differential input mode)	-12	-7… …12	30	
Input impedance	IN+		1		kΩ
	IN-		0.77		
	Differential impedance[1]	1.5			
Input frequency	Single-ended mode	0		1	MHz
	Differential mode	0		8	MHz
ESD protection	Human body model			±2	kV

24 V Digital Outputs (opto-isolated)

	All voltages referenced to $0V_{PLC}$.	Min.	Typ.	Max.	Units
Output voltage	Logic “HIGH”; $+24\ V_{PLC} = 24\ V_{DC}$; External load = 330Ω	22	23	24.5	V
	Absolute maximum, surge (duration ≤ 1s) †	-0.5		35	
Output current	Logic “HIGH”; $[+24\ V_{PLC} - V_{OUT}] \le 2$ V; all outputs except $\overline{\text{OUT5}}$ /RD and $\overline{\text{OUT4}}$ /ER			80	mA
	Logic “HIGH”; $[+24\ V_{PLC} - V_{OUT}] \le 2$ V; outputs $\overline{\text{OUT5}}$ /RD and $\overline{\text{OUT4}}$ /ER			160	mA
	Logic “LOW” (leakage crt.)		0.05	0.2	mA
	Absolute maximum, surge (duration ≤ 1s) †	-350		350	mA

Linear Hall

Applicable to IDM680-8LI		Min.	Typ.	Max.	Units
Linear Hall Voltage excursion			4	4.5	V_{PP}
Linear Hall Input voltage		0.25	0.5… …4.5	4.75	V

Input impedance			4.7		kΩ

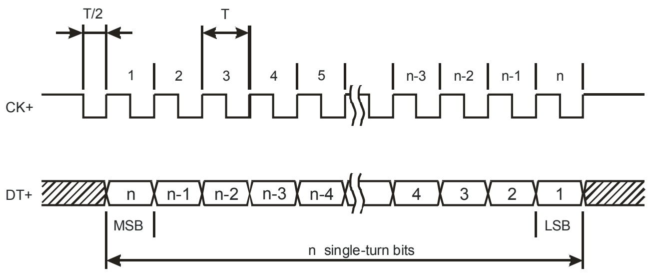
Encoder / Hall Inputs

		Min.	Typ.	Max.	Units
Single-ended mode compliance	Leave negative inputs disconnected	TTL / CMOS / open-collector			
Input threshold voltage	Single-ended mode	1.4	1.5	1.6	V
Differential mode compliance	For full RS422 compliance, see [1]	TIA/EIA-422			
Input hysteresis	Differential mode	±0.1	±0.2	±0.5	V
Input common mode range	Referenced to GND	-7		+12	V
	Absolute maximum, surge (duration ≤ 1s) †	-25		+25	
Input impedance	Single-ended mode		4.7		kΩ
	Differential mode (see [1])		120		Ω
Input Frequency		0		8	MHz
ESD Protection	Human Body Model			±2	kV

SSI Encoder Interface

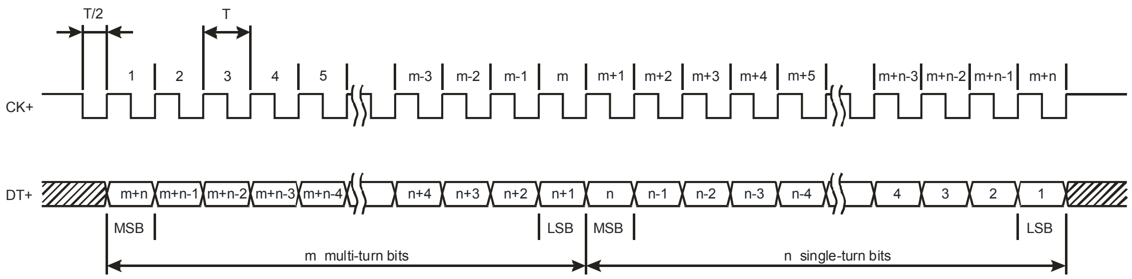
		Min.	Typ.	Max.	Units
Differential mode compliance (CLOCK, DATA) [1]	For full RS422 compliance, see **[1]**	TIA/EIA-422			
CLOCK Output voltage	Differential; 50Ω differential load	2.0	2.5	5.0	V
	Common-mode, referenced to GND	2.3	2.5	2.7	
CLOCK frequency	Software selectable	400 to 1500, in 100 increment			kHz
DATA Input hysteresis	Differential mode	±0.1	±0.2	±0.5	V
DATA Input common mode range	Referenced to GND	-7		+12	V
	Absolute maximum, surge (duration ≤ 1s) †	-25		+25	
DATA format	Software selectable	Binary / Gray			
		Single-turn / Multi-turn			
		Counting direction			
DATA resolution	Total resolution			30	bit
	Single-turn			15	bit
	Multi-turn			15	bit

Single-turn frame



CK- and DT- signals have the same form with CK+ and DT+, but with opposite polarity.

Multi-turn frame



CK- and DT- signals have the same form with CK+ and DT+, but with opposite polarity.

BiSS Encoder Interface

Applicable to IDM680-8BI		Min.	Typ.	Max.	Units
Differential mode compliance (CLOCK, DATA) [1]	For full RS422 compliance, see [1]	TIA/EIA-422			
CLOCK Output voltage	Differential; 50Ω differential load	2.0	2.5	5.0	V
	Common-mode, referenced to GND	2.3	2.5	2.7	
CLOCK frequency	Software selectable	400 to 2500, in 100 increment			kHz
DATA Input hysteresis	Differential mode	±0.1	±0.2	±0.5	V
DATA Input common mode range	Referenced to GND	-7		+12	V
	Absolute maximum, surge (duration ≤ 1s) [†]	-25		+25	
	Software selectable	Single-turn / Multi-turn			
		Counting direction			
DATA resolution	Single-turn			19	bit
	Multi-turn and single-turn			31	

Analog Inputs

		Min.	Typ.	Max.	Units
Differential voltage range			±10		V
Common-mode voltage range	Referenced to GND	-12	0… …10	+50	V
Input impedance	Differential		40		KΩ
Common-mode impedance	Referenced to GND		20		KΩ
Resolution			12		bits
Integral linearity				0.036	% FS[2]
Offset error	Common-mode voltage = 0…10 V		±0.2	±0.5	% FS[2]
Gain error	Common-mode voltage = 0…10 V		±10	±12	% FS[2]
Bandwidth (-3dB)	Depending on software settings		1.5		kHz

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			±15	kV

CAN-Bus

	All voltages referenced to CAN_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	Bit rate = 125kbps …250kbps			64	-
	Bit rate = 500kbps			50	-
	Bit rate = 1Mbps			32	-
ESD Protection	Human Body Model			±15	kV

Supply Outputs

		Min.	Typ.	Max.	Units
+5 V_{DC} voltage	Current sourced = 350 mA	4.8	5	5.2	V
+5 V_{DC} available current		400	500		mA

Other

		Min.	Typ.	Max.	Units
Operating temperature		0		40	°C
Dimensions	Length x Width x Height	136 x 95 x 26			mm
Weight			0.30		kg
Frame Insulation voltage withstand	GND to SHIELD (connected to frame)			250	V
Storage temperature	Not powered	-40		85	°C
Humidity	Non-condensing	0		90	%RH
Altitude	Referenced to sea-level	0		4000	m
Dust & humidity protection	According to IEC-60925		IP20		

[1] Differential input impedance is ≥1.5KΩ. For full RS-422 compliance, 120Ω termination resistors must be connected across the differential pairs, as close as possible to the drive input pins.

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

† Stresses beyond those 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.

3. Step 1. Hardware Installation

3.1. Mounting

The IDM680 drive was designed to be cooled by natural convection. It can be mounted horizontally (with label upwards) or vertically inside a cabinet (see Figure 3.1), with motor wires going down. In both cases, leave at least 25mm between the drive and surrounding walls/drives, to allow for free air circulation.

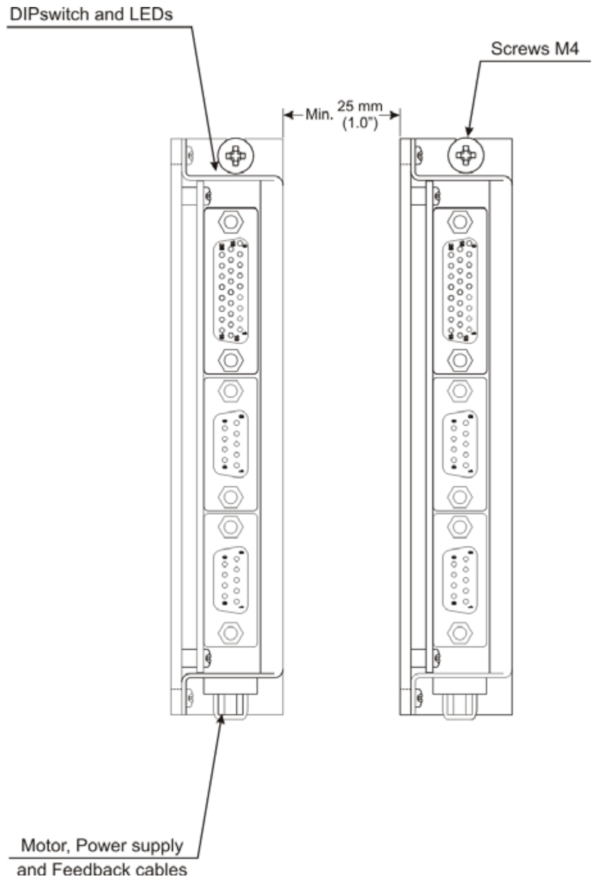


Figure 3.1. *Recommended mounting of IDM680 in a cabinet*

3.2. Connectors and Connection Diagrams

3.2.1. Connectors Layout

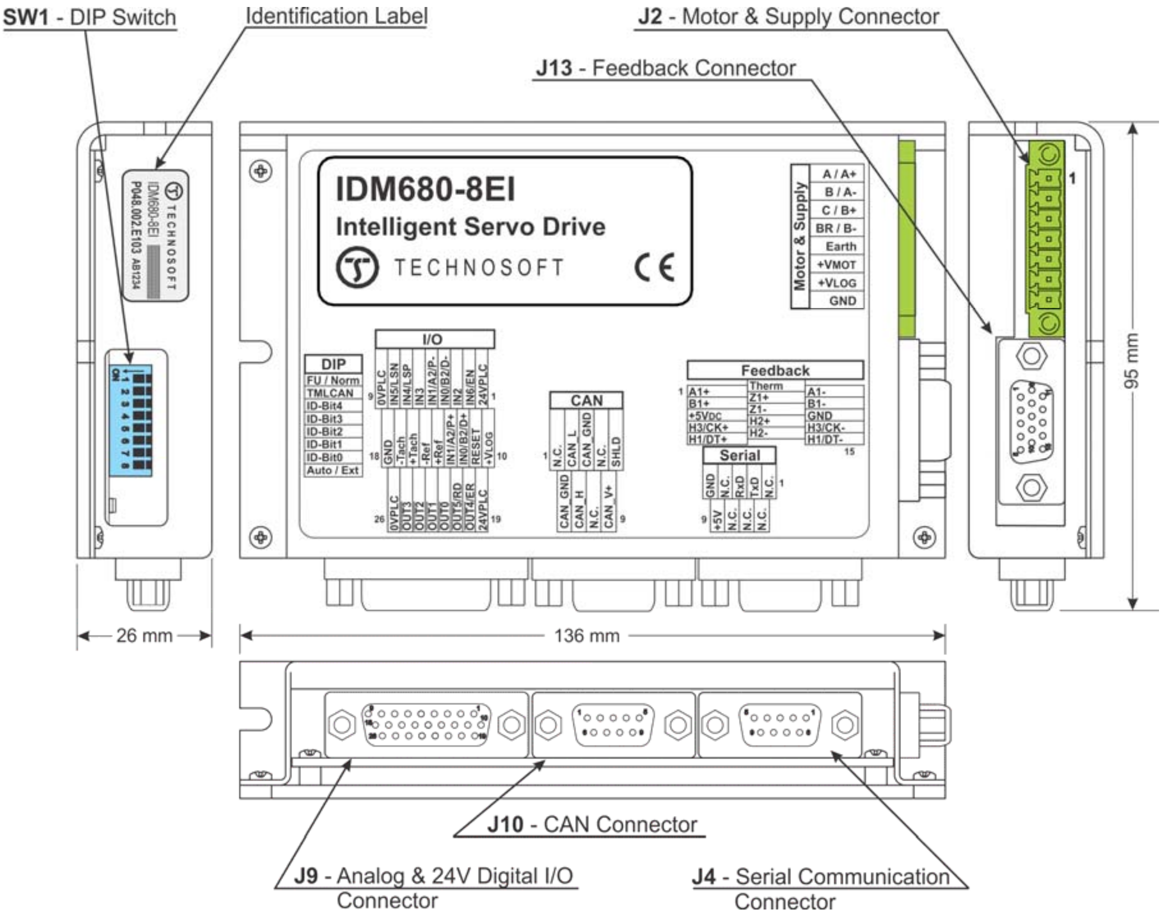


Figure 3.2. *IDM680-8EI connectors layout*

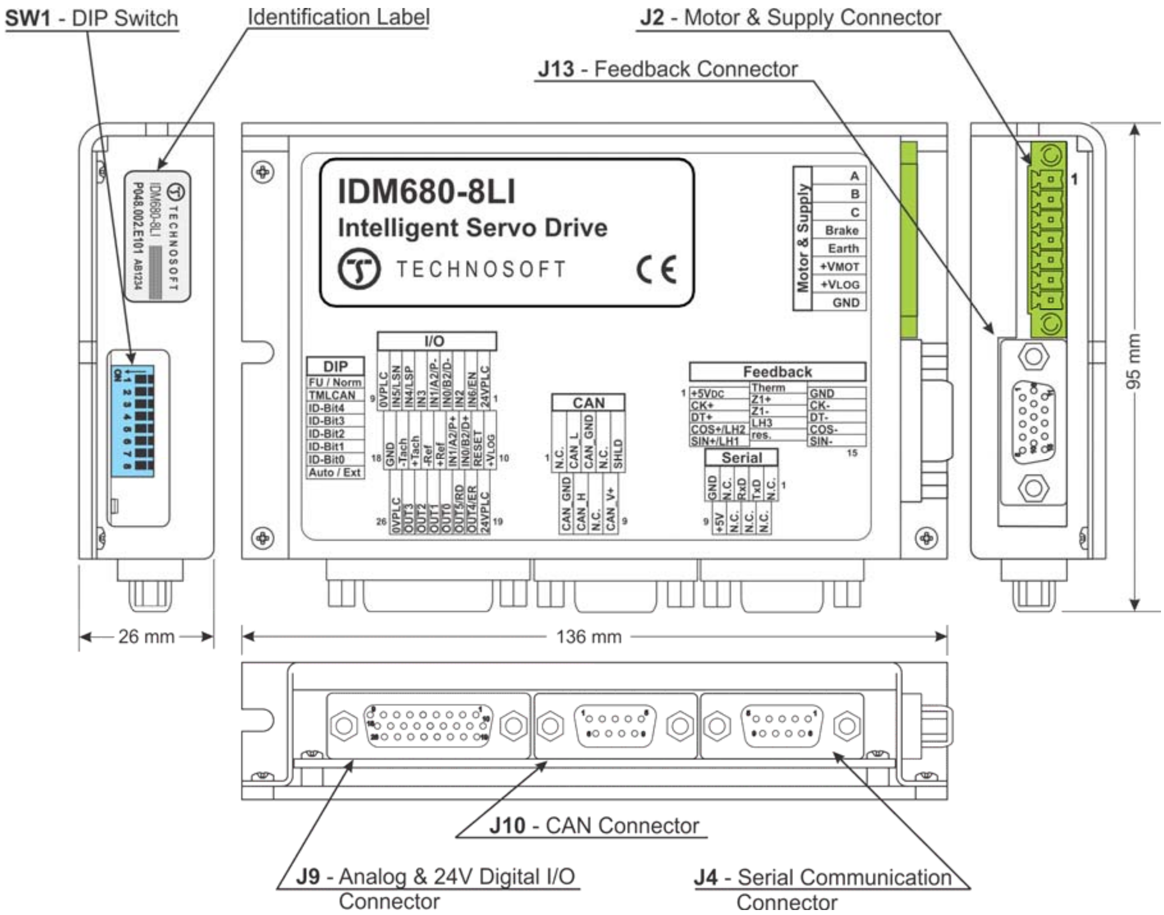


Figure 3.3. *IDM680-8LI connectors layout*

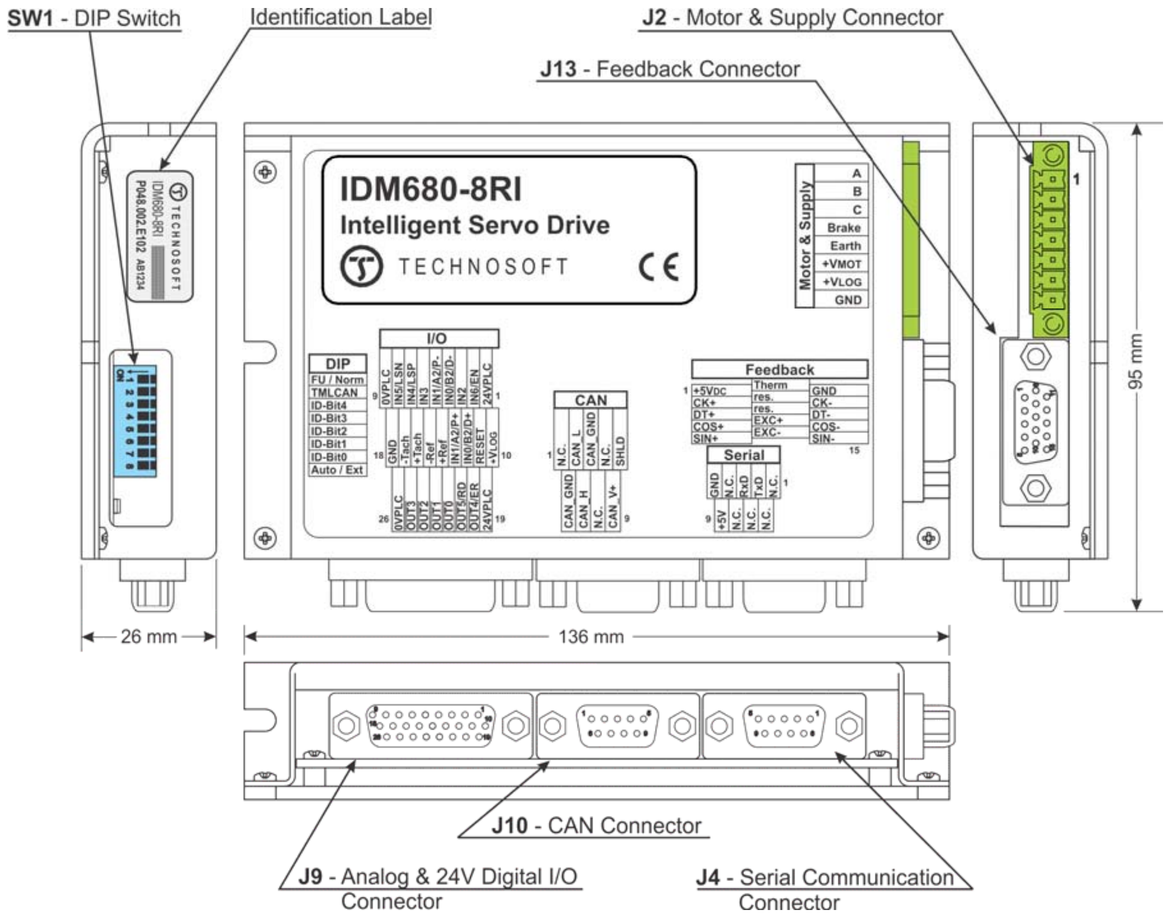


Figure 3.4. *IDM680-8RI connectors layout*

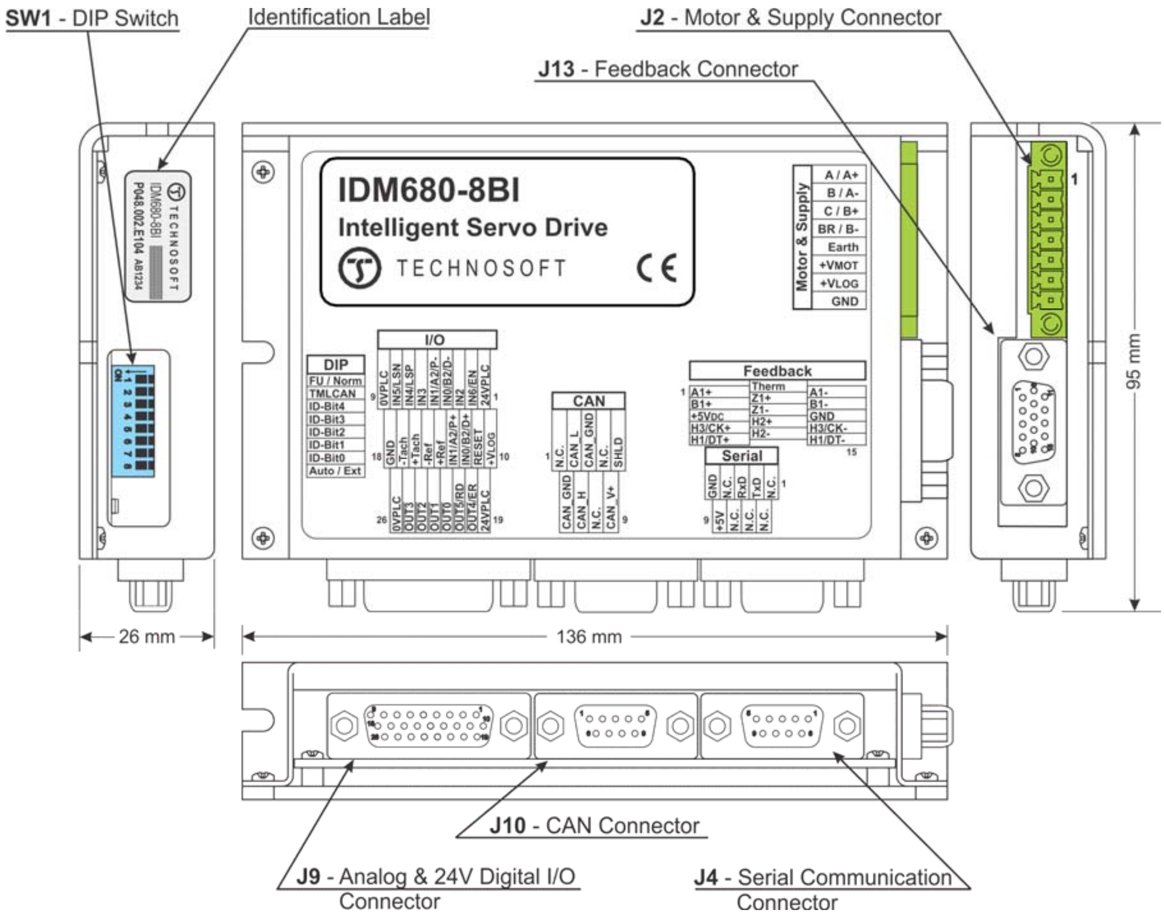


Figure 3.5. *IDM680-8BI connectors layout*

3.2.2. Identification Labels

Figure 3.6. *IDM680-8EI Identification Label*

Figure 3.7. *IDM680-8LI Identification Label*

Figure 3.8. *IDM680-8RI Identification Label*

Figure 3.9. *IDM680-8BI Identification Label*

3.2.3. Motor & Supply – J2 Connector

Pin	Name	Type	Function
1	A / A+	O	• Brushless motor or step motor (3-phase): **Phase A** • Step motor (2-phase): Phase **A+** • DC brush motor: **+** (positive terminal)
2	B / A-	O	• Brushless motor or step motor (3-phase): **Phase B** • Step motor (2-phase): **Phase A-** • DC brush motor: **-** (negative terminal)
3	C / B+	O	• Brushless motor or step motor (3-phase): **Phase C** • Step motor (2-phase): **Phase B+** • DC brush motor: not connected
4	BR / B-	O	• **Brake** output for external brake resistor (only when the drive is used with brushless or DC brushed motors) • Step motor (2-phase): **Phase B-** • DC brush motor: not connected
5	Earth	-	Earth connection
6	$+V_{MOT}$	I	Positive terminal of the motor supply: 12 to 80 V_{DC}
7	$+V_{LOG}$	I	Positive terminal of the logic supply: 12 to 48 V_{DC}
8	GND	-	Negative terminal of the $+V_{MOT}$ and $+V_{LOG}$ external power supplies

Remark: The stepper connections are not present on IDM680-8LI, IDM680-8RI and IDM680-8BI. On thes drives the J2 pins 1, 2, 3, and 4 are named:

Pin	Name
1	A
2	B
3	C
4	BRAKE

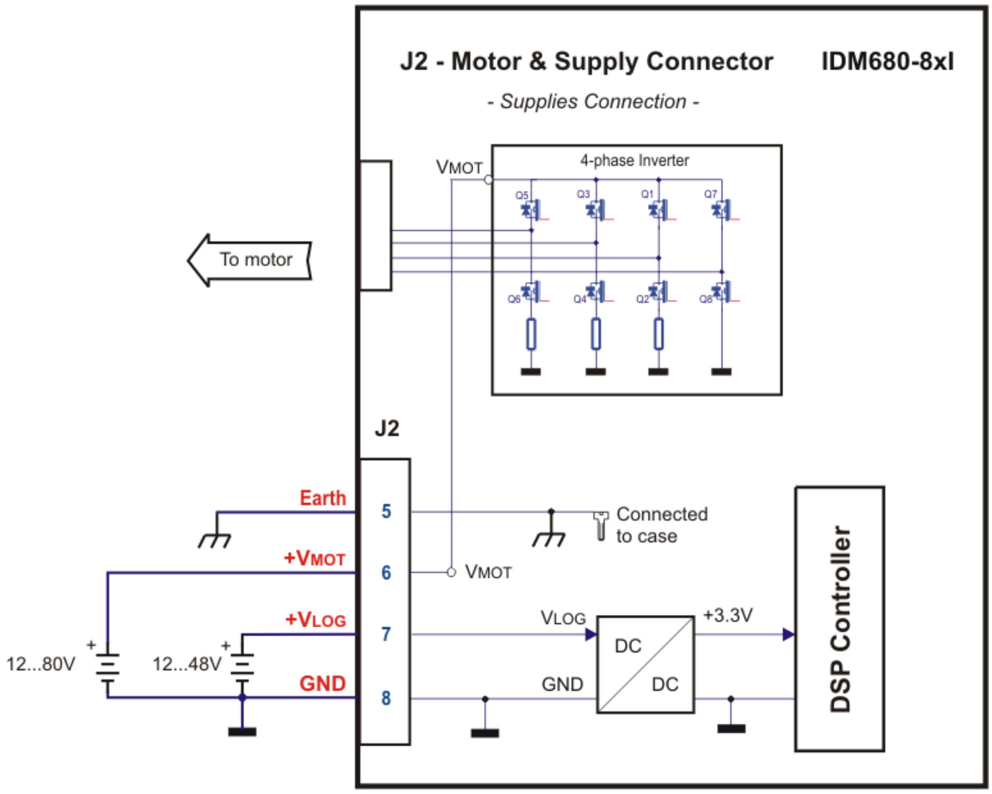


Figure 3.10. *J2 – Supplies connection*

Remark: *The EARTH signal is connected internally to the metal case and to all SHIELD signals. It is completely insulated from all electric signals of IDM680. This feature may facilitate avoiding ground loops. It is recommended that Earth be connected to GND at only one point, preferably close to the V_{MOT} supply output.*

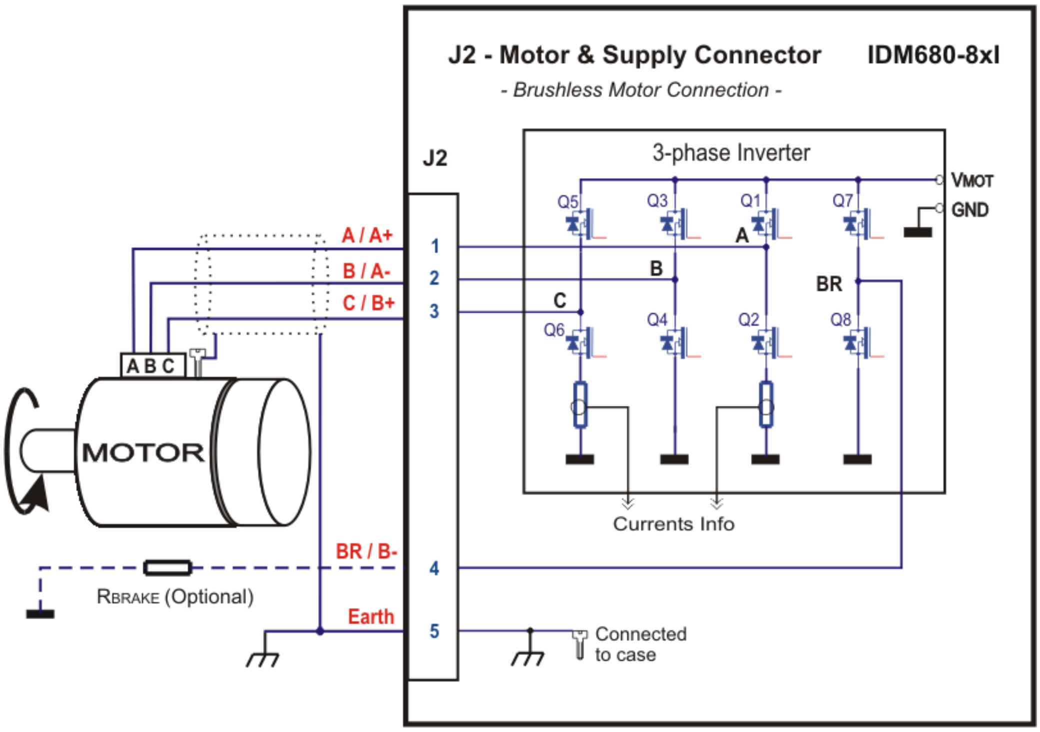


Figure 3.11. *J2 – Brushless motor connection*

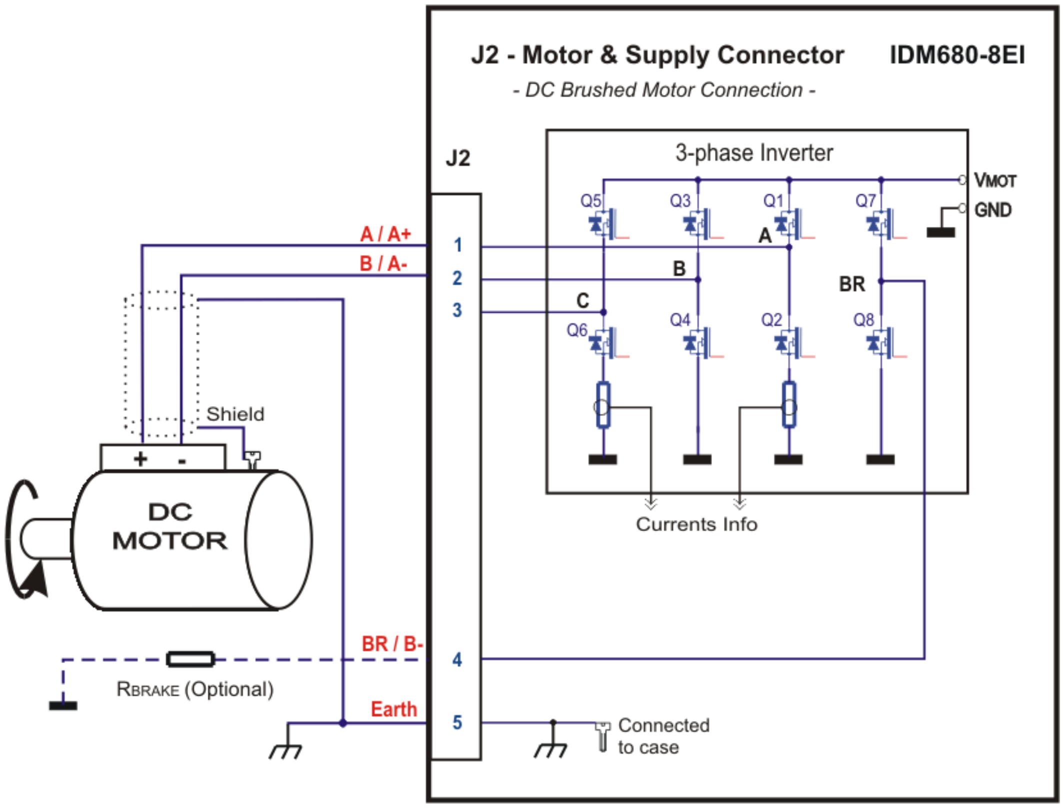


Figure 3.12. *J2 – DC brushed motor connection*

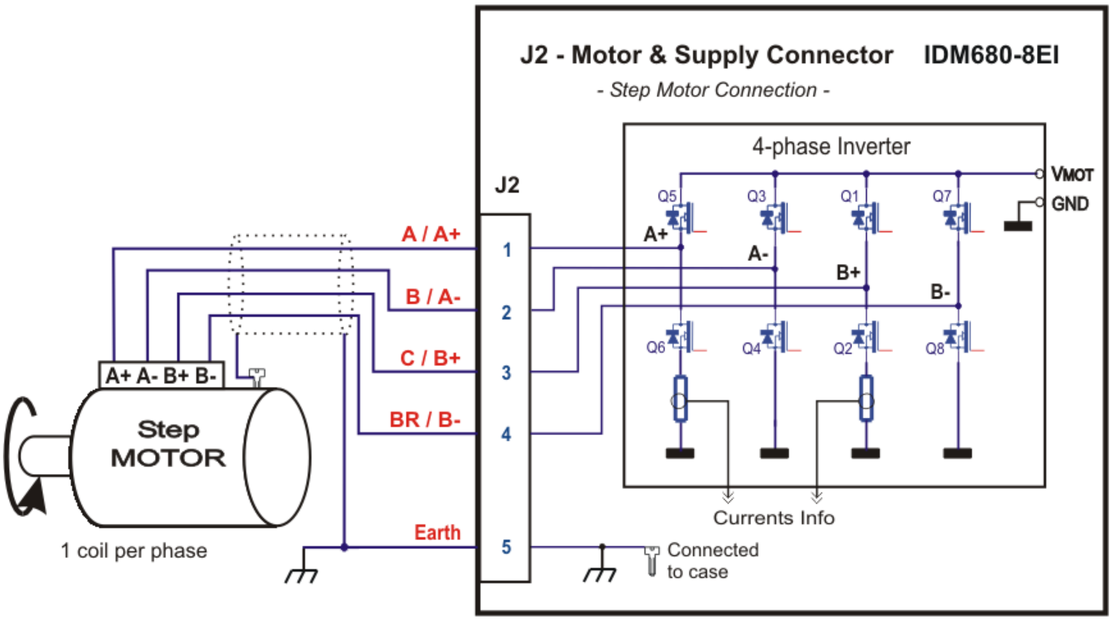


Figure 3.13. *J2 – Step motor connection – 2-phase motor with 1 coil per phase*

Remark: *The EARTH signal is connected internally to the metal case and to all SHIELD signals. It is completely insulated from all electric signals of IDM680-8EI this feature may facilitate avoiding ground loops. It is recommended that Earth be connected to GND at only one point, preferably close to the* V_{MOT} *supply output.*

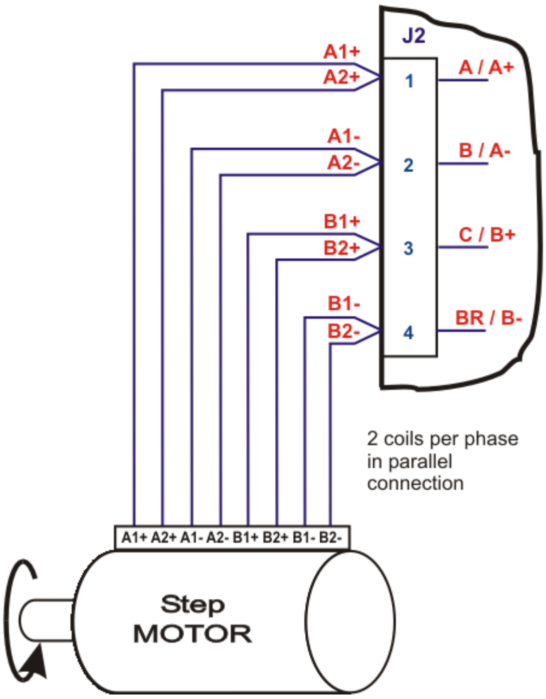


Figure 3.14 *J2 – Connection of a 2-phase motor with 2 coils per phase in parallel*

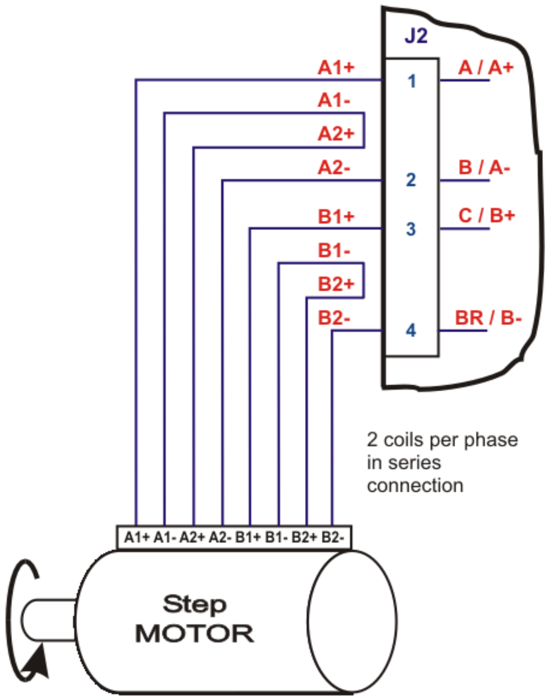


Figure 3.15. *J2 – Connection of a 2-phase motor with 2 coils per phase in series*

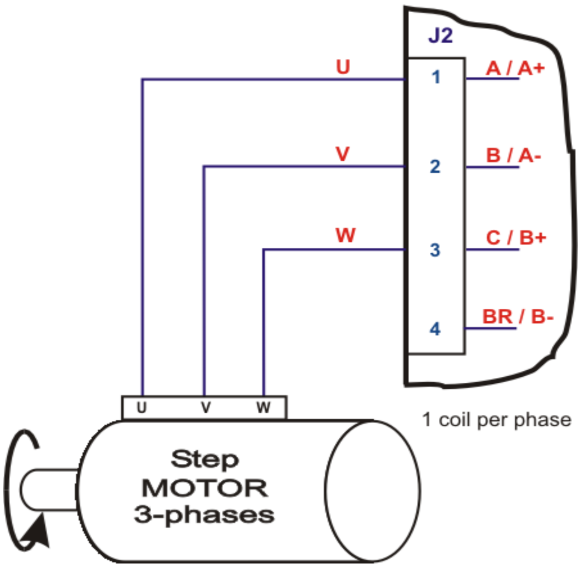


Figure 3.16. *J2 – Connection of a 3-phase motor*

3.2.3.1 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 IDM680 earth/shield 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 IDM680 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.

3.2.3.2 Recommendations for Power Supply On-Off Switch and Wiring

a) If motor supply V_{MOT} is switched on abruptly, the in-rush (start-up) current can reach very high values that can damage the drive. In order to limit the in-rush current, it is preferable to use the inherent soft-start provided by the power supplies when are turned on. Therefore, it is recommended to locate the switch for the motor supply at the INPUT of the power supply (see Figure 3.17) and NOT at the output i.e. between the supply and drive.

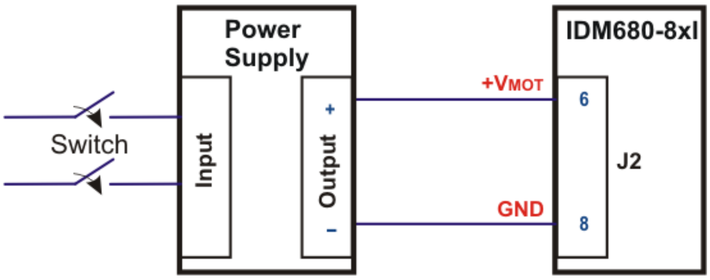


Figure 3.17. *J2 – Motor supply connection – Recommended in-rush current limitation*

b) When the above solution is not possible (as in the case of uninterruptible power supplies, or batteries/accumulators), connect an external capacitor of minimum 470μF between the switch and the drive, to reduce the slew-rate rising slope of the motor supply voltage.

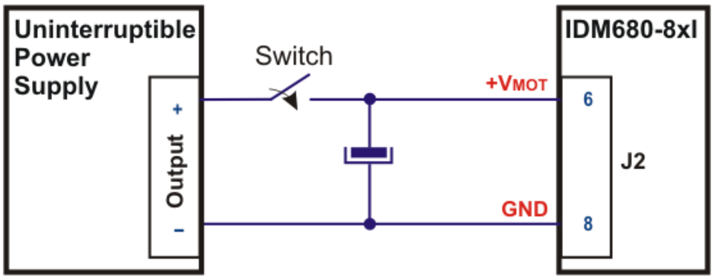


Figure 3.18. *J2 – Motor supply connection – Alternative in-rush current limitation*

⚠	**WARNING!**	***ALWAYS PROVIDE AN EXTERNAL MEAN TO SWITCH OFF THE POWER SUPPLIES! ALWAYS TURN OFF SUPPLIES BEFORE INSTALLING THE DRIVE***

⚠	**CAUTION!**	***ALWAYS LIMIT THE IN-RUSH (START-UP) CURRENT OF THE MOTOR SUPPLY, OTHERWISE IT CAN DAMAGE THE DRIVE***

3.2.3.3 Recommendations for Supply Wiring

1. Use short, thick wires between the IDM680 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 1,000 μF (rated at an appropriate voltage) right on the terminals of the IDM680.
2. When the same motor power supply is used for multiple drives, do a "star" connection centered (electrically) around the supply outputs. Connect each drive to the common motor supply using separate wires for plus and return.
3. Always connect the IDM680 earth / shield pin to a good quality earth point. The IDM680 generates electromagnetic disturbances when it's case is not grounded. Use a short and thick connection from the earth pin of the drive to the earth point. Whenever possible, mount the IDM680 drive on a metallic surface connected to earth. For mechanical fixing, use good quality plated screws that won't oxidize during the expected lifetime.

3.2.3.4 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 92V, the drive over-voltage protection is triggered and the drive power stage is disabled. In order to avoid this situation you have 2 options:

Option 1. Add a capacitor on the motor supply big enough to absorb the overall energy flowing back to the supply. The capacitor must be rated to a voltage equal or bigger than the maximum expected over-voltage and can be sized with the formula:

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

where:

U_{MAX} = 92V is the over-voltage protection limit

C_{Drive} = 200 μF is the drive internal capacitance

U_{NOM} = 80V is nominal motor supply voltage

E_M = the overall energy flowing back to the supply in Joules. In case of a rotary motor and load, E_M can be computed with the formula:

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

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.*

Option 2. Connect a brake resistor R_{BR} between pin 4 and pin 8 of the Motor & Supply connector J2 and activate the drive braking circuit from EasySetUp when motor supply voltage exceeds: U_{BRAKE} = 87V. This option is not available when the drive is used with a step motor.

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

Brake resistor selection

The brake resistor value must be chosen to respect the following conditions:

1. to limit the maximum current below the drive peak current I_{PEAK} = 16.5A

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

2. to sustain the required *braking power*:

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

where C = C_{EXT} + C_{DRIVE} is the overall capacitance on the motor supply (external + drive), i.e:

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

3. to limit the average current below the drive nominal current I_{NOM}=8A

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

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

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

Remarks:

1. *If* $\frac{U_{MAX}}{I_{PEAK}} > \frac{U_{BRAKE}^2}{2 \times P_{BR}}$ *the braking power* P_{BR} *must be reduced by increasing either* t_d *– the time to decelerate or* C_{EXT} *– the external capacitance on the motor supply*
2. *If* $\frac{P_{BR} \times t_d}{t_{CYCLE} \times I_{NOM}^2} > \frac{U_{BRAKE}^2}{2 \times P_{BR}}$ *either the braking power must be reduced (see Remark 1) or* t_{CYCLE} *– the time interval between braking cycles must be increased*

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

3.2.4. Feedback – J13 Connector (IDM680-8EI)

Pin	Name on the Drive cover	Type	Function / Comments
1	A1+	I	**Positive A** for differential encoder or **A** for single-ended encoder
2	B1+	I	**Positive B** for differential encoder or **B** for single-ended encoder
3	+5 V_{DC}	O	+5 V_{DC} Supply (generated internally)
4	H3/CK+	I/O	**Positive Hall 3** input for differential Hall or **Hall 3** for single-ended Hall **Positive Clock** output signal for differential SSI encoder
5	H1/DT+	I	**Positive Hall 1** for differential Hall or **Hall 1** for single-ended Hall **Positive Data** signal for differential SSI encoder
6	Therm	I	Analog input from motor thermal sensor
7	Z1+	I	**Positive Z** for differential encoder or **Z** for single-ended encoder [1]*)
8	Z1-	I	**Negative Z** for differential encoder
9	H2+	I	**Positive Hall 2** for differential Hall or **Hall 2** for single-ended Hall [2]*)
10	H2-	I	**Negative Hall 2** for differential Hall
11	A1-	I	**Negative A** for differential encoder
12	B1-	I	**Negative B** for differential encoder
13	GND	-	Ground of the encoder supply
14	H3/CK-	I/O	**Negative Hall 3** input for differential Hall; **Negative Clock** output signal for differential SSI encoder
15	H1/DT-	I	**Negative Hall 1** for differential Hall **Negative Data** signal for differential SSI encoder
case	SHIELD	-	Shield; Connected to frame

CAUTION! ***CHECK CURRENT CONSUMPTION FROM +5VDC SUPPLY! BYPASSING THE MAXIMUM ALLOWED CURRENT MIGHT LEAD TO DRIVE MALFUNCTION***

CAUTION! ***THE FEEDBACK CONNECTOR SIGNALS ARE ELECTROSTATICALLY SENSITIVE AND SHALL BE HANDLED ONLY IN AN ESD PROTECTED ENVIRONMENT***

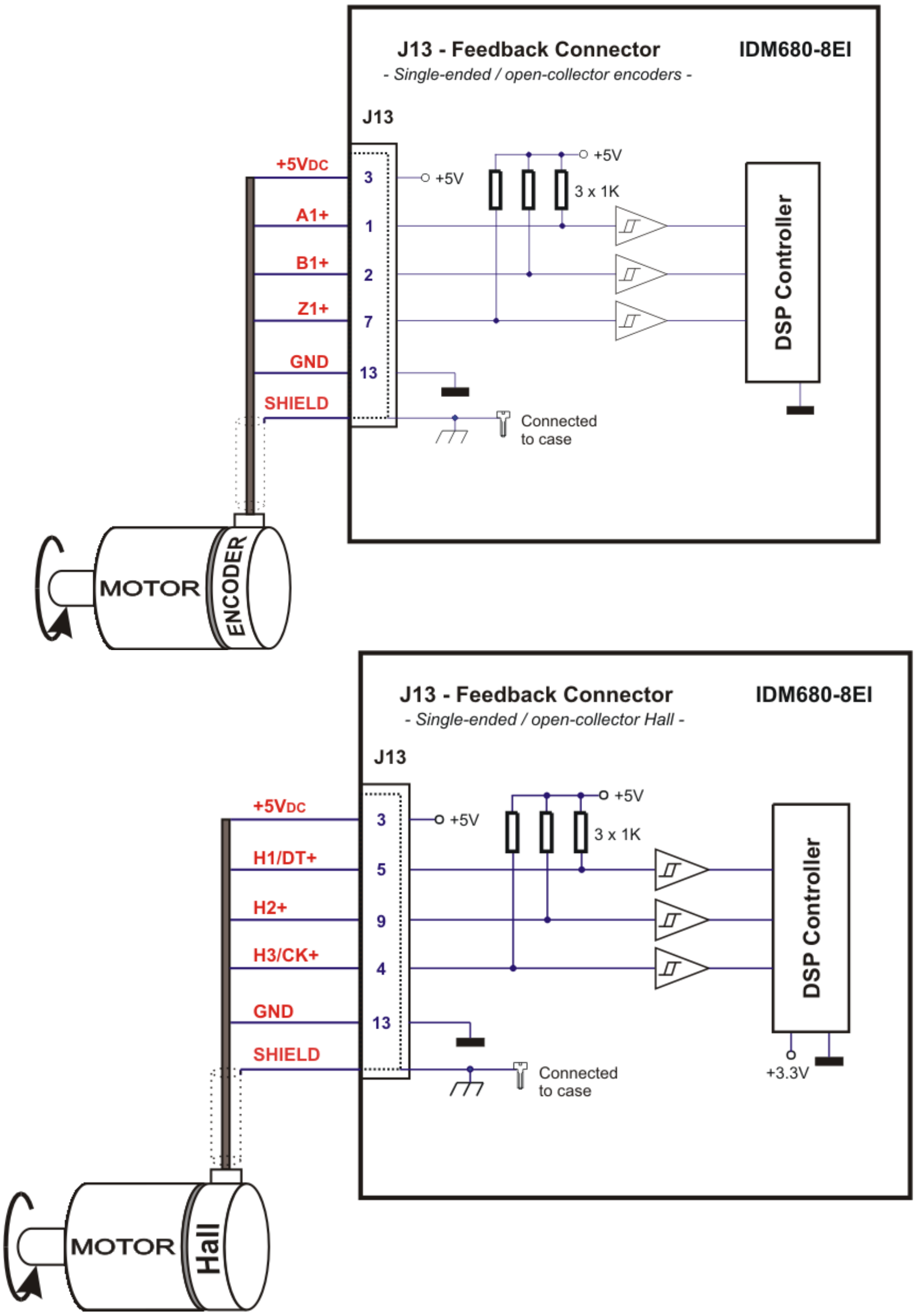


Figure 3.19. *J13 – Single-ended / open-collector encoder and Hall connection*

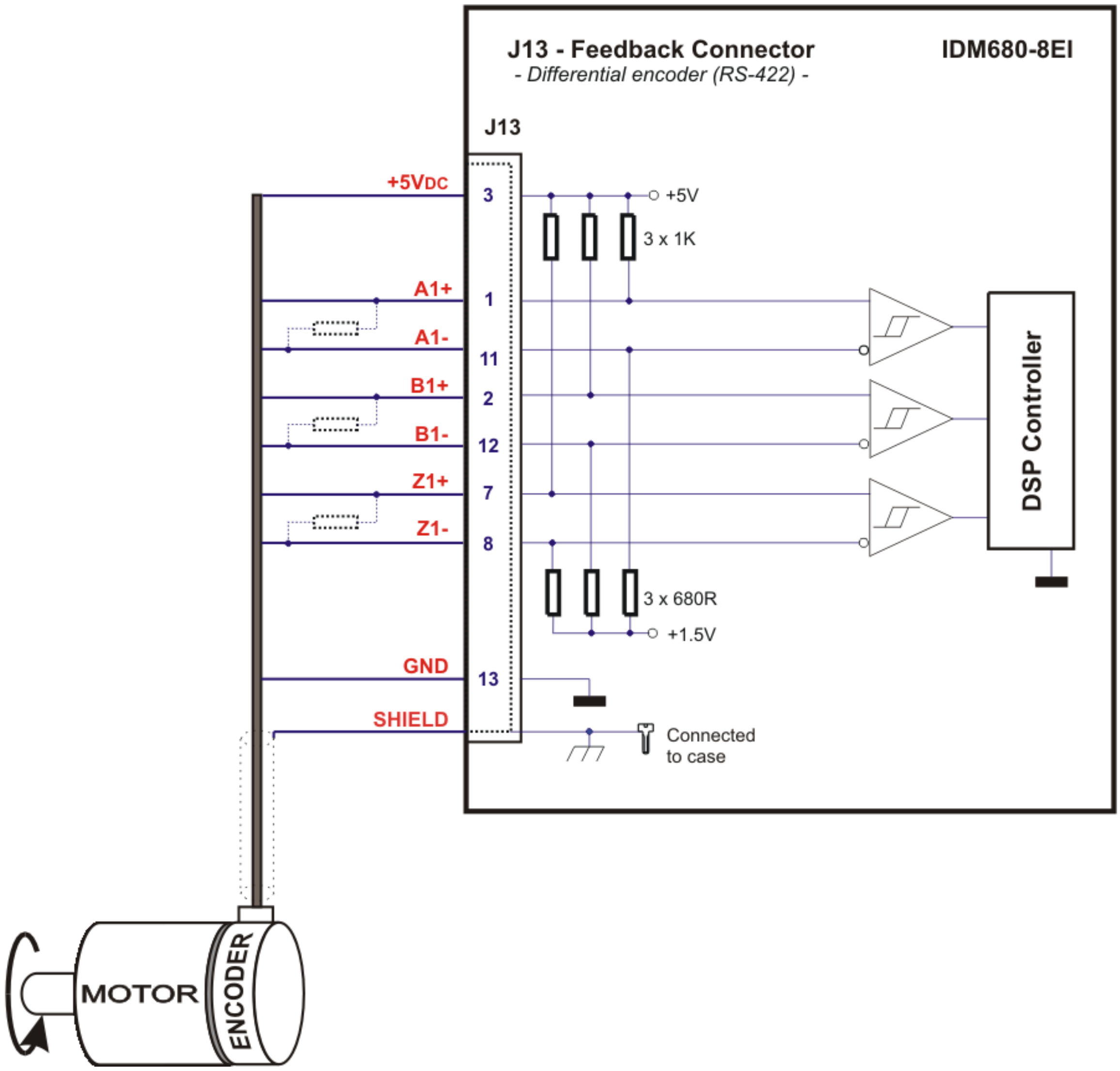


Figure 3.20. *J13 – Differential (RS-422) encoder connection*

Remark: *For long (>10 meters) encoder lines add 120Ω termination resistors close to the drive.*

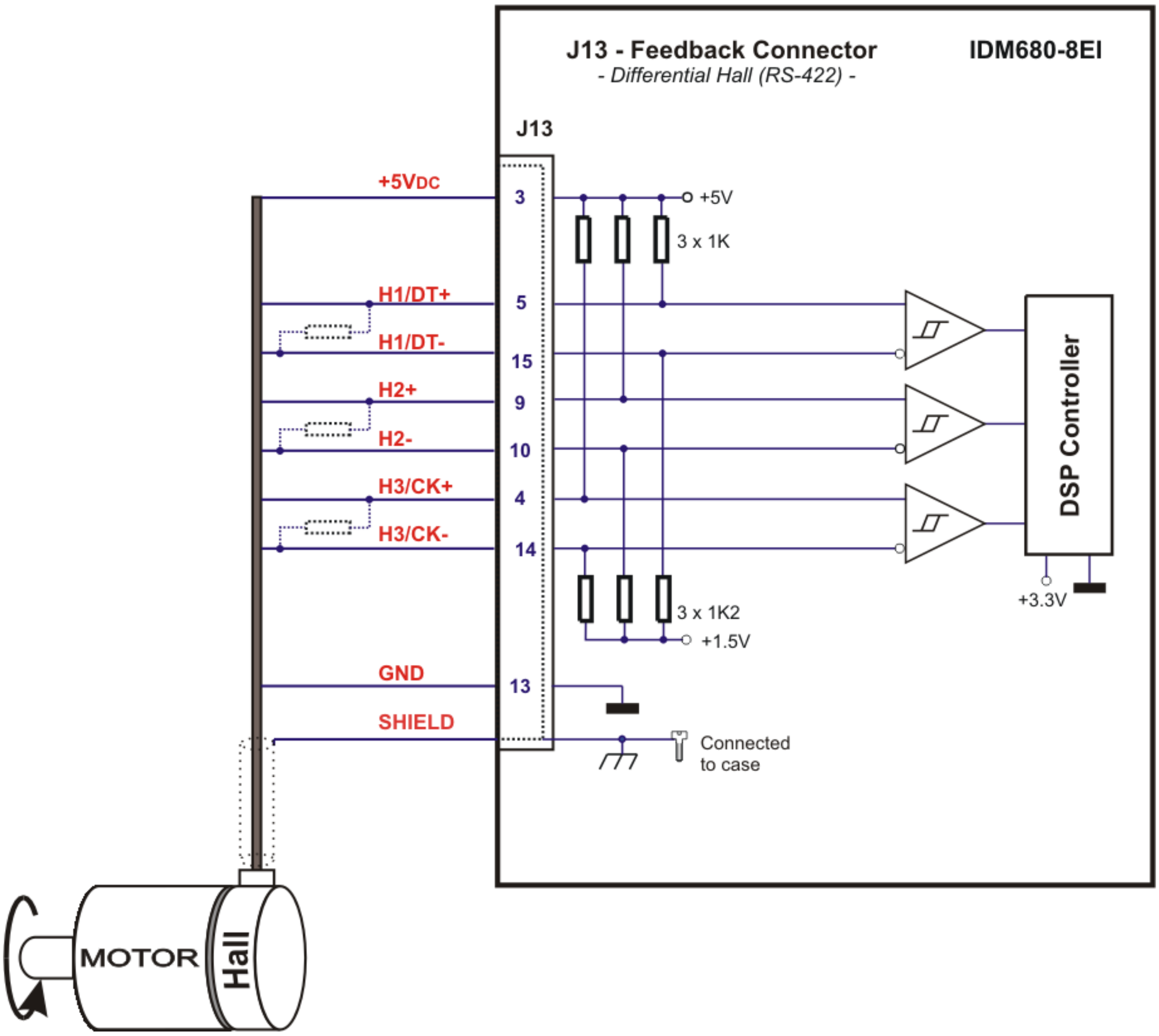


Figure 3.21. *J13 – Differential (RS-422) Hall connection*

Remark: *For long (>10 meters) Hall lines add 120Ω termination resistors close to the drive.*

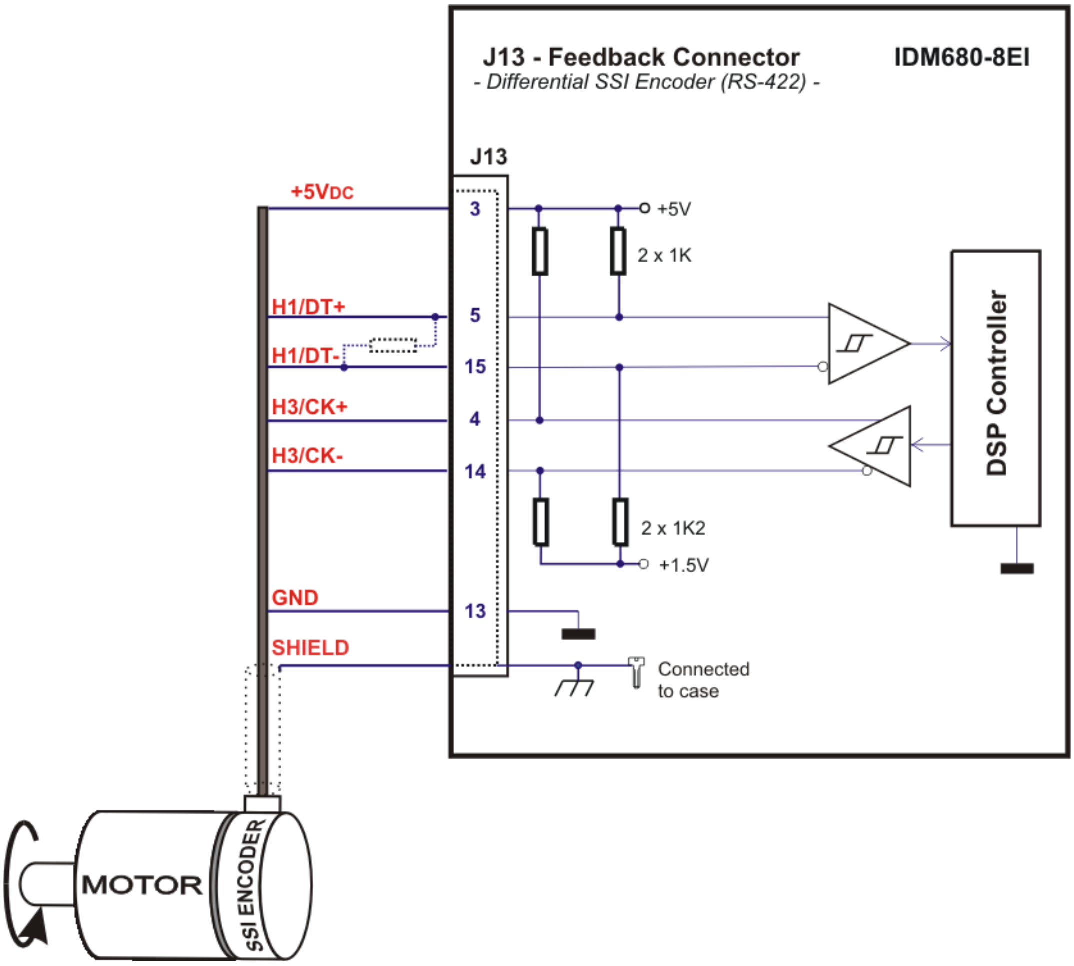


Figure 3.22. *J13 – Differential (RS-422) SSI encoder connection*

Remarks:

1.For long (>10 meters) SSI encoder lines add 120Ω termination resistors close to the drive.

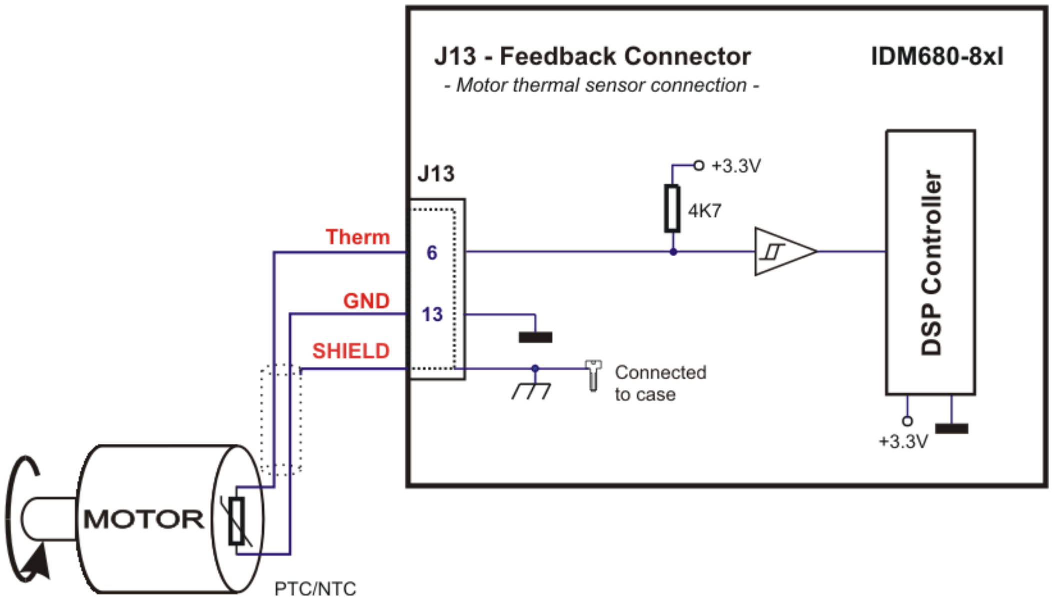


Figure 3.23. *J13 – Motor thermal sensor connection*

3.2.4.1 Recommendations for Feedback Devices 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: A+ with A-, B+ with B-, Z+ with Z-, H1/DT+ with H1/DT-, H2+ with H2-, H3/CK+ with H3/CK-. Use another twisted pair for the 5V supply and GND.

b) Keep the ground connection between an encoder and the IDM680-8EI even if the encoder supply is not provided by the drive. When using shielded cable, connect the cable shield to the earth at the encoder side. Leave the shield unconnected at the IDS side. **Never use the shield as a conductor caring a signal, for example as a ground line!**. This situation can lead to a worse behavior than a non-shielded cable

c) 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 earth potential, at only one end. This point could be either the IDM680-8EI (using the earth/shield pin(s)) or the encoder / motor. Do not connect the shield at both ends.

d) If the IDM680 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.

3.2.5. Feedback – J13 Connector (IDM680-8LI)

Pin	***Name on the Drive cover***	Type	Function / Comments
1	+5 V_{DC}	O	+5 V_{DC} Supply (generated internally)
2	OutB/CK+	O	**Positive Clock** output signal for differential EnDat protocol
3	OutA/DT+	I/O	**Positive Data** input/output signal for differential EnDat protocol
4	COS+/LH2	I	**Positive Cosine** input of the sine/cosine encoder **Linear Hall 2** input
5	SIN+/LH1	I	**Positive Sine** input of the sine/cosine encoder **Linear Hall 1** input
6	Therm	I	Analog input from motor thermal sensor
7	Z1+	I	**Positive Z** for differential encoder or **Z** for single-ended encoder[1]
8	Z1-	I	**Negative Z** for differential encoder
9	LH3	I	**Linear Hall 3** input signal
10	res.	-	Reserved
11	GND	-	Ground of the 5 V_{DC} supply
12	OutB/CK-	O	**Negative Clock** output signal for differential EnDat protocol
13	OutA/DT-	I/O	**Negative Data** input/output signal for differential EnDat protocol
14	COS-	I	**Negative Cosine** input of the sine/cosine encoder
15	SIN-	I	**Negative Sine** input of the sine/cosine encoder
case	SHIELD		Shield; Connected to frame

[1] Can capture the master position and also the motor position if an incremental or absolute sine/cosine encoder is used

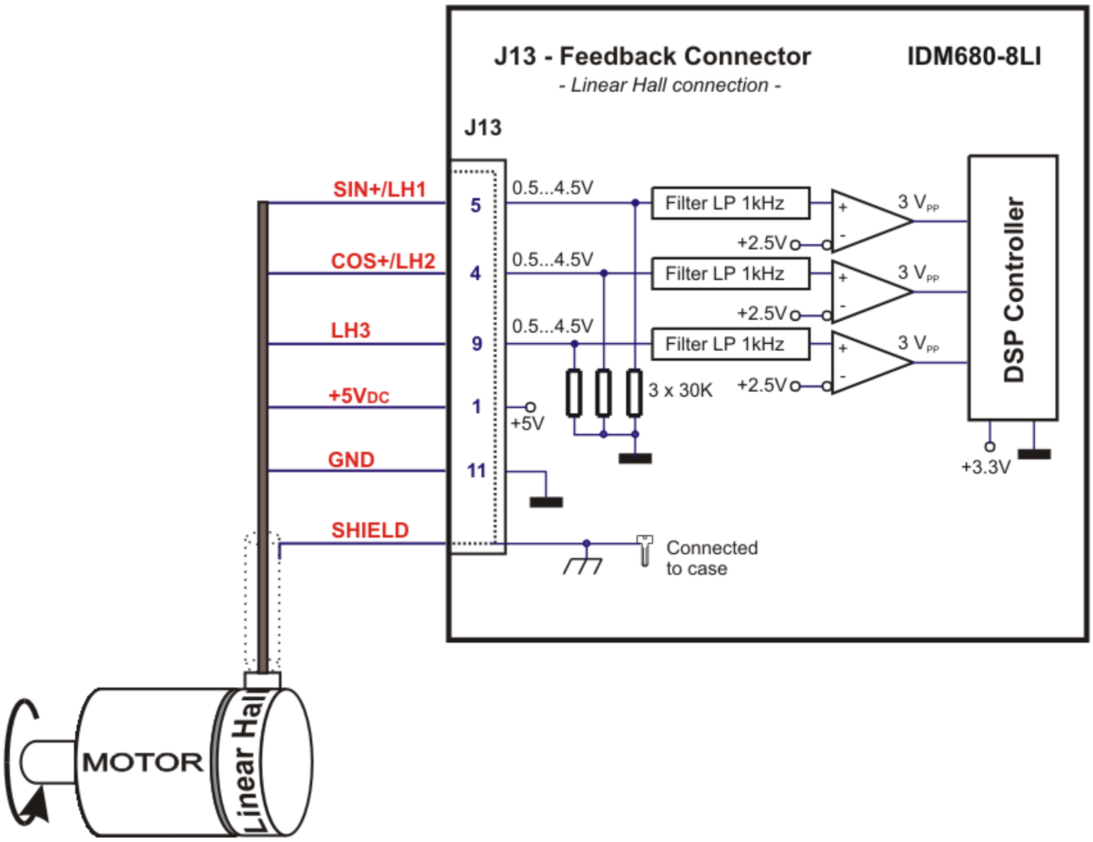


Figure 3.24. *J13 – Linear Hall sensor connection*

Remark: *Motor thermal sensor connection is presented in Figure 3.23*

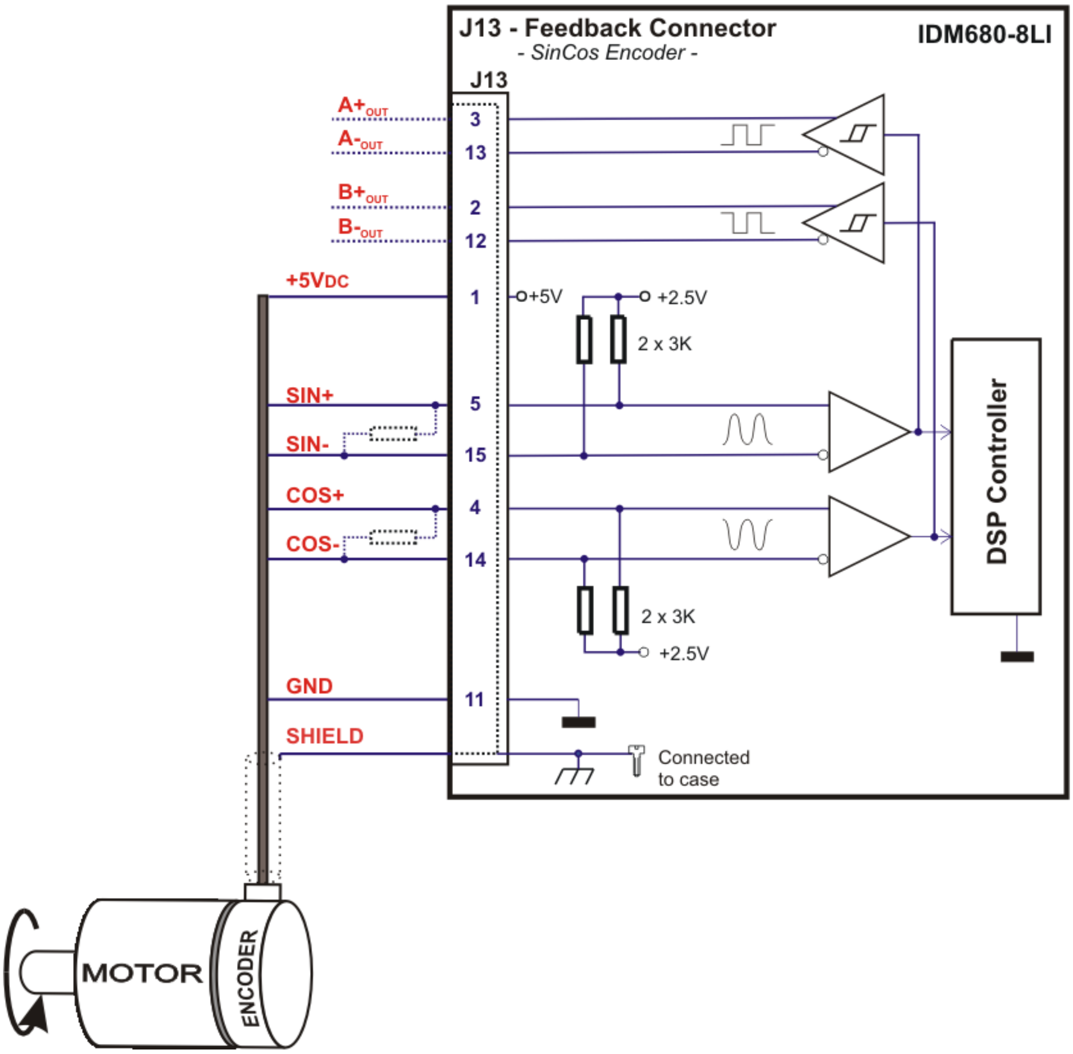


Figure 3.25. *J13 – Incremental sine/cosine encoder connection*

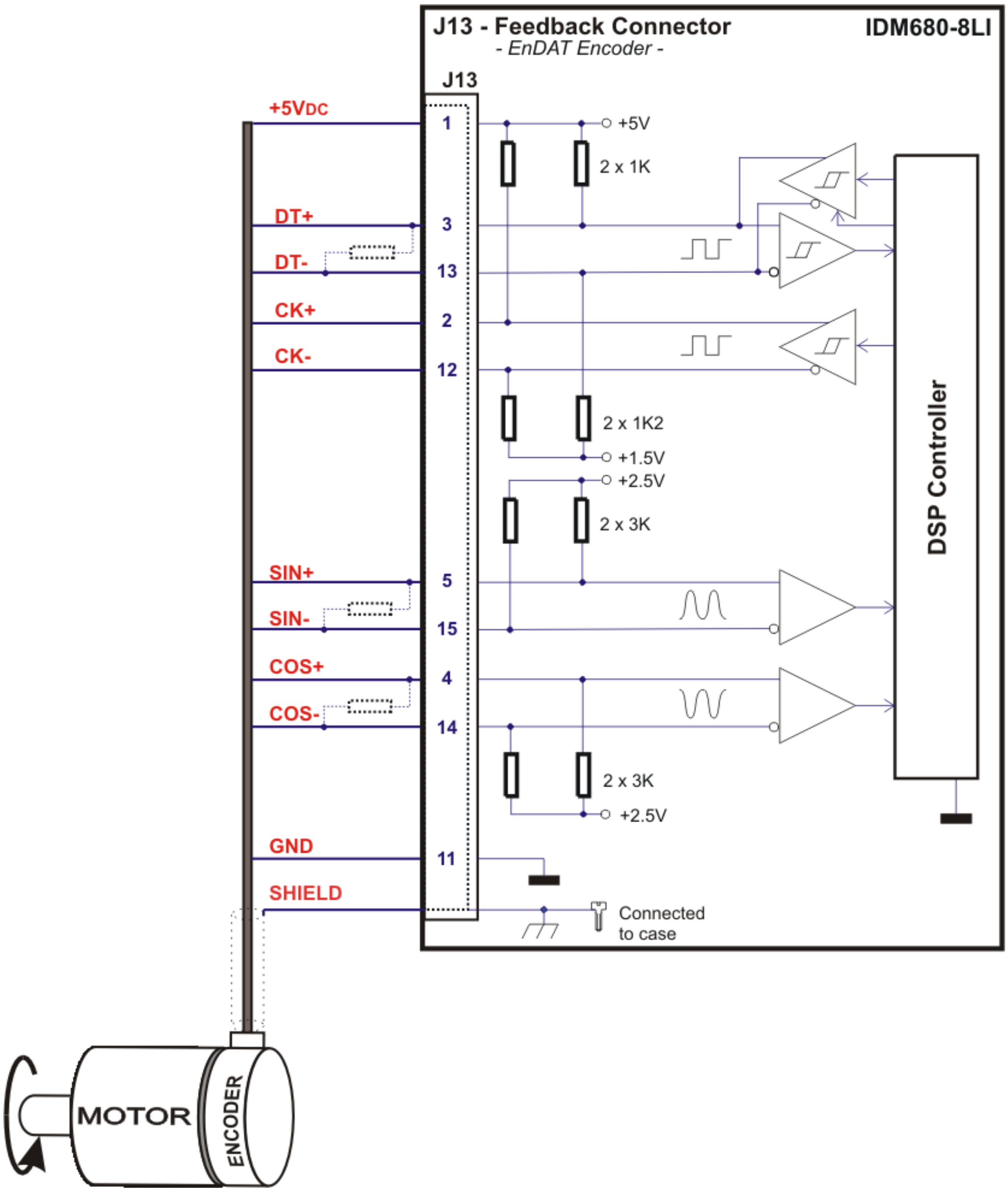


Figure 3.26. *J13 – Absolute sine/cosine encoder connection with EnDat communication protocol*

3.2.6. Feedback – J13 Connector (IDM680-8RI)

Pin	Name on the Drive cover	Type	Function / Comments
1	+5 V_{DC}	O	+5 V_{DC} Supply (generated internally)
2	CK+	O	**Positive Clock** output signal for differential SSI encoder[1]
3	DT+	I	**Positive Data** signal for differential SSI encoder
4	COS+	I	**Positive Cosine** input from the resolver
5	SIN+	I	**Positive Sine** input from the resolver
6	Therm	I	Analog input from motor thermal sensor
7	res.	-	Reserved
8	res.	-	Reserved
9	EXC+	O	**Positive Excitation** output signal to the resolver
10	EXC-	O	**Negative Excitation** output signal to the resolver
11	GND	-	Ground of the 5 V_{DC} supply
12	CK-	O	**Negative Clock** output signal for differential SSI encoder
13	DT-	I	**Negative Data** signal for differential SSI encoder
14	COS-	I	**Negative Cosine** input from the resolver
15	SIN-	I	**Negative Sine** input from the resolver
case	SHIELD		Shield; Connected to frame

[1] IDM680-8RI includes an SSI encoder interface. This is reserved for future developments. For motor-sensor configurations with SSI encoders, use IDM680-8EI. For dual loop operation with resolver on motor and SSI encoder on load, contact Technosoft

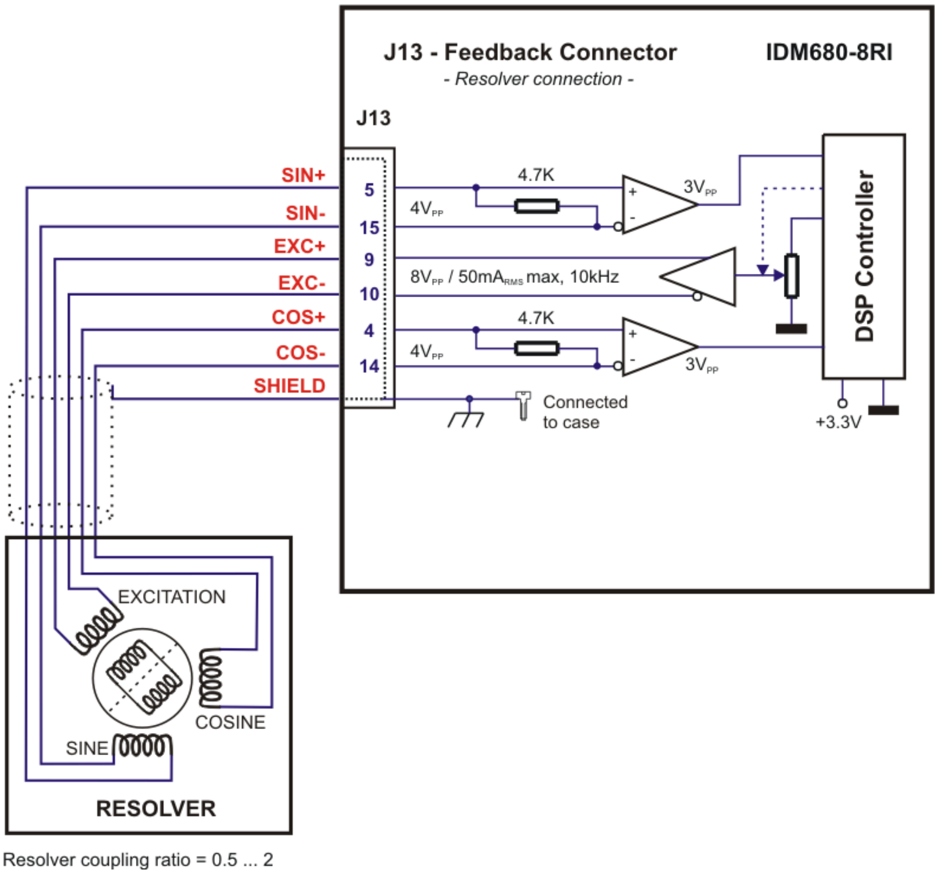


Figure 3.27. *J13 – Resolver connection*

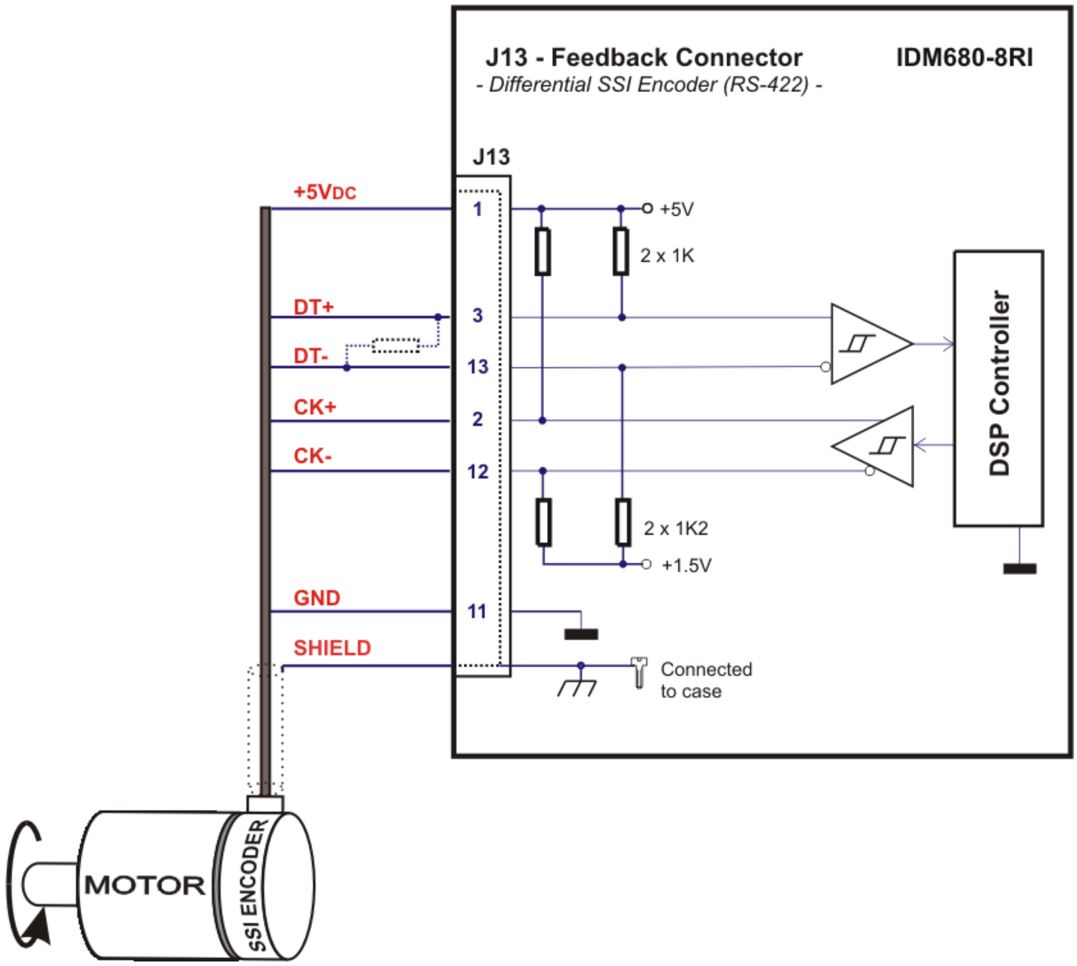


Figure 3.28. *J13 – Differential (RS-422) SSI encoder connection*

3.2.7. Feedback – J13 Connector (IDM680-8BI)

Pin	Name on the Drive cover	Type	Function / Comments
1	A1+	I	**Positive A** for differential encoder or **A** for single-ended encoder
2	B1+	I	**Positive B** for differential encoder or **B** for single-ended encoder
3	+5 V_{DC}	O	+5 V_{DC} Supply (generated internally)
4	H3/CK+	I/O	**Positive Hall 3** input for differential Hall or **Hall 3** for single-ended Hall **Positive Clock** output signal for differential BiSS/SSI encoder
5	H1/DT+	I	**Positive Hall 1** for differential Hall or **Hall 1** for single-ended Hall **Positive Data** signal for differential BiSS/SSI encoder
6	Therm	I	Analog input from motor thermal sensor
7	Z1+	I	**Positive Z** for differential encoder or **Z** for single-ended encoder [1]*)
8	Z1-	I	**Negative Z** for differential encoder
9	H2+	I	**Positive Hall 2** for differential Hall or **Hall 2** for single-ended Hall [2]*)
10	H2-	I	**Negative Hall 2** for differential Hall
11	A1-	I	**Negative A** for differential encoder
12	B1-	I	**Negative B** for differential encoder
13	GND	-	Ground of the encoder supply
14	H3/CK-	I/O	**Negative Hall 3** input for differential Hall; **Negative Clock** output signal for differential BiSS encoder
15	H1/DT-	I	**Negative Hall 1** for differential Hall **Negative Data** signal for differential BiSS encoder
case	SHIELD	-	Shield; Connected to frame

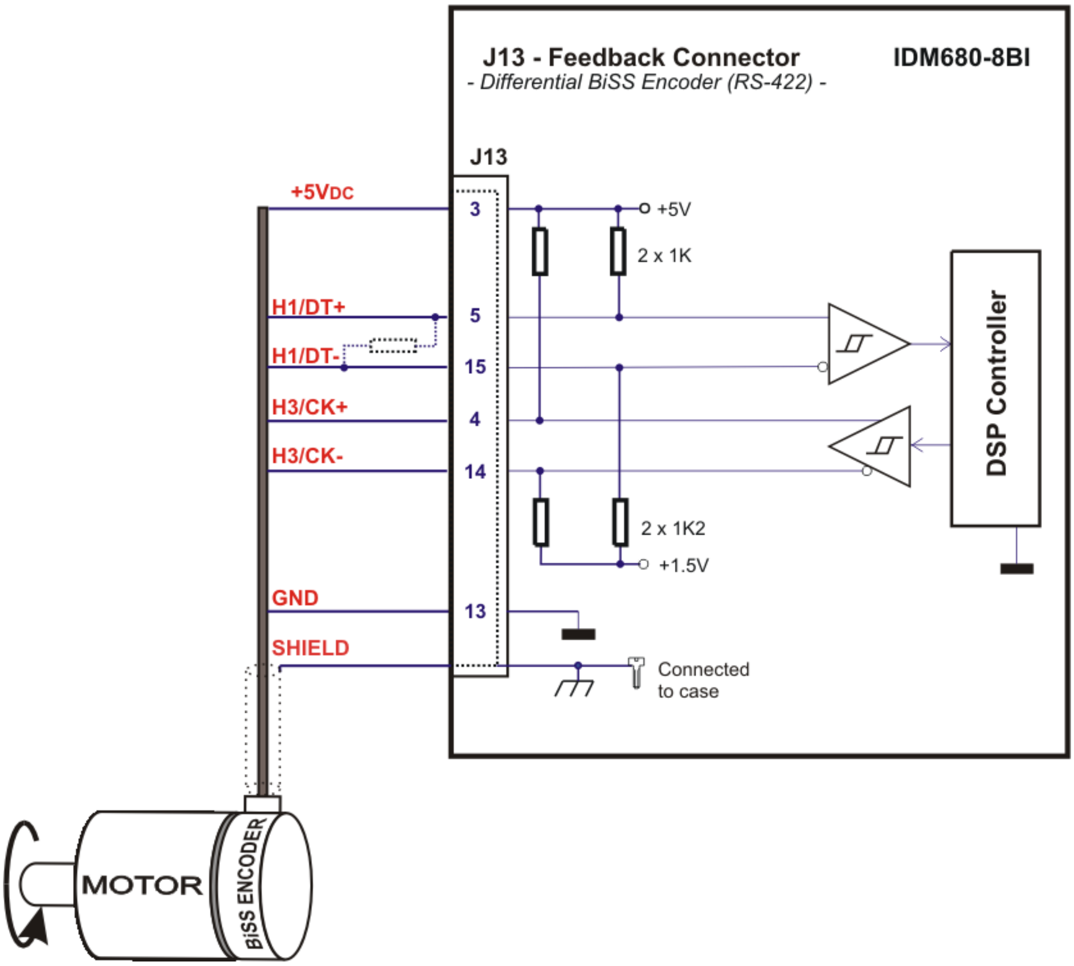


Figure 3.29. *J13 – Differential (RS-422) BiSS encoder connection*

Remarks:

1.For long (>10 meters) BiSS encoder lines add 120Ω termination resistors close to the drive.

2.For BiSS encoders that need more than $5V_{DC}$, the supply voltage should be provided from an external source.

3.2.8. Analog & Digital I/O – J9 Connector

Pin	Name on the Drive cover	TML name	Type	Function / Alternate function / Comments
1, 19	24VPLC	-	I	• 24 V power supply (+) terminal for all opto-isolated I/O
2	IN6/EN	IN(6)	I	• 24V Enable input, read as **In(6)**. On inactive level disables the drive operation similarly to AXISOFF command (power stage is turned off). Read high (1 logic) when 24VPLC are applied on IN6/EN pin • Opto-isolated • Programmable polarity / active level
3	IN2/HOME	IN(2)	I	• 24V General-purpose input **In(2)**. Read high (1 logic) when 24VPLC are applied on IN2/HOME pin • 24V Home input in homing sequences. Can be set to capture on transitions both motor and master position • Opto-isolated
4	IN0/B2/D-	IN(0)	I	• RS-422 differential **B-** / 24V single-ended **B** input when external reference is 2nd (master) encoder • RS-422 differential **Dir-** / 24V single-ended **Dir** input when external reference is Pulse & Direction • 24V General-purpose input **In(0)**. Read low (0 logic) when 24VPLC are applied on IN0/B2/D- pin • Compatible RS-422 and 24V single-ended
5	IN1/A2/P-	IN(1)	I	• RS-422 differential **A-** / 24V single-ended **A** input when external reference is 2nd (master) encoder • RS-422 differential **Puls-** / 24V single-ended **Puls** input when external reference is Pulse & Direction • 24V General-purpose input **In(1)**. Read low (0 logic) when 24VPLC are applied on IN1/A2/P- pin • Compatible RS-422 and 24V single-ended
6	IN3	IN(3)	I	• 24V General-purpose input **In(3).** Read high (1 logic) when 24VPLC are applied on IN3 pin • Opto-isolated
7	IN4/LSP	IN(4)	I	• 24V Positive limit switch input. On active level stops motion in positive direction • 24V General-purpose input **In(4)** if limit switches are disabled. Read high (1 logic) when 24VPLC are applied on IN4/LSP pin • Opto-isolated • Programmable polarity / active level

8	IN5/LSN	IN(5)	I	• 24V Negative limit switch input. On active level stops motion in negative direction • 24V General-purpose input **In(5)** if limit switches are disabled. Read high (1 logic) when 24VPLC are applied on IN5/LSN pin • Opto-isolated • Programmable polarity / active level
9, 26	0VPLC	-	-	• Ground (-) terminal for all opto-isolated I/O
10	$+V_{LOG}$	-	O	• + V_{LOG}. Logic supply voltage (as applied on J2, pin 7)
11	RESET	-	I	• RESET pin – connect to 24VPLC to reset the drive
12	IN0/B2/D+	IN(0)	I	• 24V general-purpose input **In(0).** Read high (1 logic) when 24VPLC are applied on IN0/B2/D+ pin • RS-422 differential **B+** / 5V single-ended **B** input when external reference is 2nd (master) encoder • RS-422 differential **Dir+** / 5V single-ended **Dir** input when external reference is Pulse & Direction • Compatible RS-422, 5V and 24V single-ended
13	IN1/A2/P+	IN(1)	I	• 24V general-purpose input **In(1)**. Read high (1 logic) when 24VPLC are applied on IN1/A2/P+ pin • RS-422 differential **A+** / 5V single-ended **A** input when external reference is 2nd (master) encoder • RS-422 differential **Puls+** / 5V single-ended **Puls** input when external reference is Pulse & Direction, or • Compatible RS-422, 5V and 24V single-ended
14	+Ref	AD5	I	• Analogue position, speed or torque reference input • +/-10 V differential • 12-bit resolution
15	-Ref		I	
16	+Tach	AD2	I	• Analogue speed feedback (tachometer input) • +/-10 V differential • 12-bit resolution
17	- Tach		I	
18	GND	-	O	• Ground terminal for all non-isolated I/O
20	$\overline{OUT4}$ /ER	OUT(4)	O	• 24 V Error output, seen as **Out(4)**. When Out(4) is commanded low (0 logic), $\overline{OUT4}$ /ER pin is set to +24VPLC and lights the red led • Opto-isolated • Short-circuit protected
21	$\overline{OUT5}$ /RD	OUT(5)	O	• 24 V Ready output, seen as **Out(5)**. When Out(5) is commanded low (0 logic), $\overline{OUT5}$ /RD pin is set to +24VPLC and lights the green LED • Opto-isolated • Short-circuit protected

22	$\overline{\text{OUT0}}$	OUT(0)	O	• 24 V General-purpose output **Out(0)**. When Out(0) is commanded low (0 logic), $\overline{\text{OUT0}}$ pin is set to +24VPLC • Opto-isolated • Short-circuit protected
23	$\overline{\text{OUT1}}$	OUT(1)	O	• 24 V General-purpose output **Out(1)**. When Out(1) is commanded low (0 logic), $\overline{\text{OUT1}}$ pin is set to +24VPLC • Opto-isolated • Short-circuit protected
24	$\overline{\text{OUT2}}$	OUT(2)	O	• 24 V General-purpose output **Out(2)**. When Out(2) is commanded low (0 logic), $\overline{\text{OUT2}}$ pin is set to +24VPLC • Opto-isolated • Short-circuit protected
25	$\overline{\text{OUT3}}$	OUT(3)	O	• 24 V General-purpose output **Out(3)**. When Out(3) is commanded low (0 logic), $\overline{\text{OUT3}}$ pin is set to +24VPLC • Opto-isolated • Short-circuit protected
case	SHIELD	-	-	Shield; Connected to frame

CAUTION! ***THE I/O CONNECTOR SIGNALS ARE ELECTRO-STATICALLY SENSITIVE AND SHALL BE HANDLED ONLY IN AN ESD PROTECTED ENVIRONMENT.***

Remarks:

3. *The 24V opto-isolated I/O signals are referenced to the isolated ground 0VPLC, which shall be common to all the devices sharing these signals.*
4. *The 24V opto-isolated inputs have a typical threshold of 8 Volts, therefore will not accept TTL levels.*
5. *The isolated 24VPLC supply is required only for operation of the outputs. Hence, if your application uses only opto-isolated inputs, the 24VPLC supply connection is not necessary.*
6. *The inputs In(0) and In(1) accept both TTL (5V) and 24V signals and are not opto-isolated. These inputs are referenced to the drive logic ground GND*

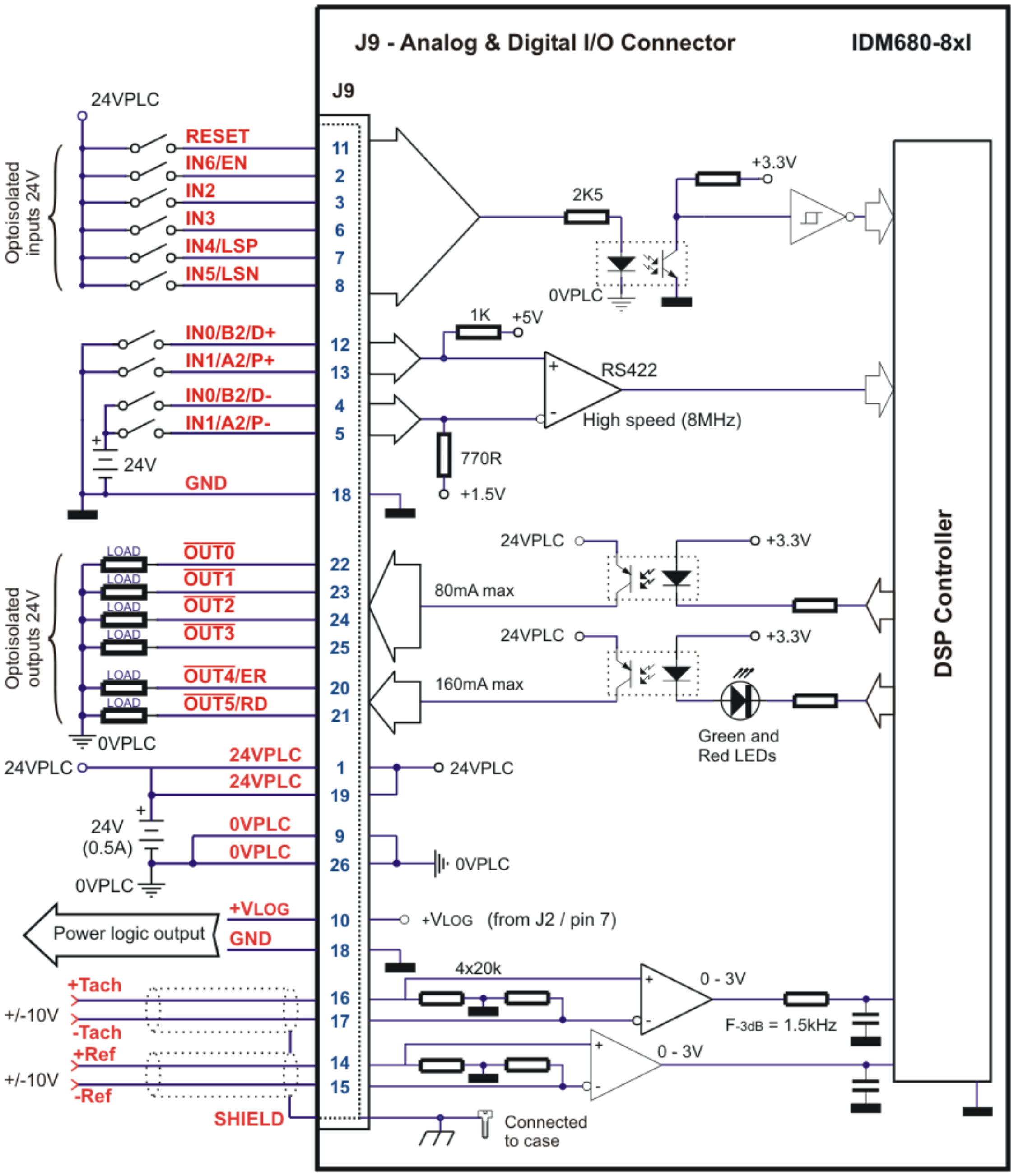


Figure 3.30 *J9 – Analogue & Digital I/O connections*

3.2.8.1 Recommendations for Analogue Signals 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 (-)

d) For all of the above cases, connect the cable shield to the drive I/O connector frame and leave the other shield end unconnected to the signal source. To further increase the noise protection, use a double shielded cable with inner shield connected to drive GND and outer shield connected to the drive I/O connector frame. Leave both shields unconnected on the signal source side

e) If the signal source output voltage is larger than +/-10V, use a 3-resistor differential divider, located near the IDM680 I/O connector. Choose the divider resistances as low as possible, close to the signal source output current limit, to minimize the noise

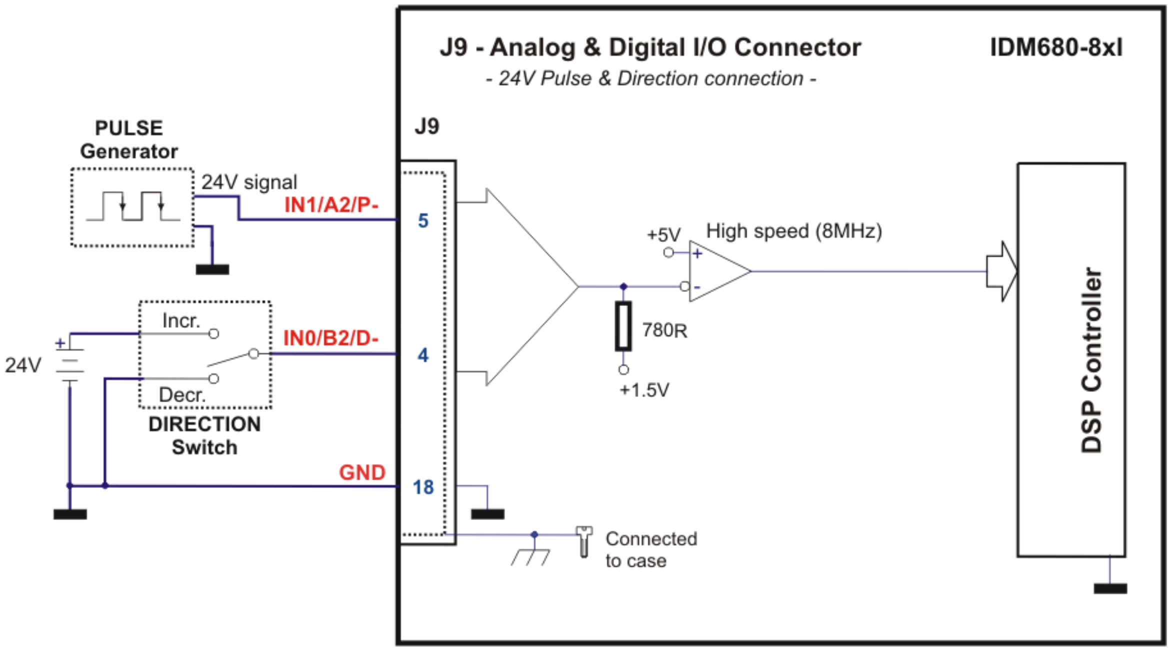


Figure 3.31 *J9 – 24 V Pulse & Direction connection*

Remarks:

1. *When using 24 V Pulse & Direction connection, leave Pins 12 – IN0/B2/D+ and 13 – IN1/A2/P+ open.*
2. *When IN1/A2/P- is used as PULSE input in Pulse & Direction motion mode, on each falling edge the reference (or feedback) is incremented / decremented.*
3. *When IN0/B2/D- is used as DIRECTION input in Pulse & Direction motion mode, the reference (or feedback) is incremented if this pin is pulled high.*

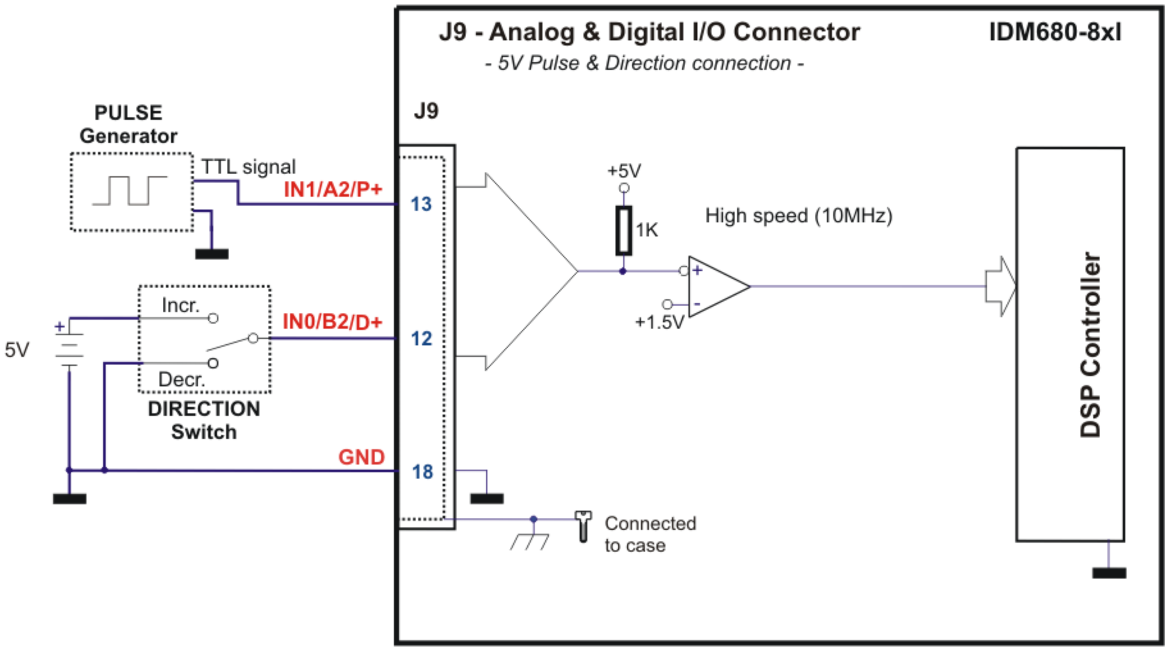


Figure 3.32. *J9 – 5V Pulse & Direction connection*

Remarks:

1. *When using 5 V Pulse & Direction connection, leave Pins 4 – IN0/B2/D- and 5–IN1/A2/P- open.*
2. *When IN1/A2/P+ is used as PULSE input in Pulse & Direction motion mode, on each rising edge the reference (or feedback) is incremented / decremented.*
3. *When IN0/B2/D+ is used as DIRECTION input in Pulse & Direction motion mode, the reference (or feedback) is incremented if this pin is pulled low.*

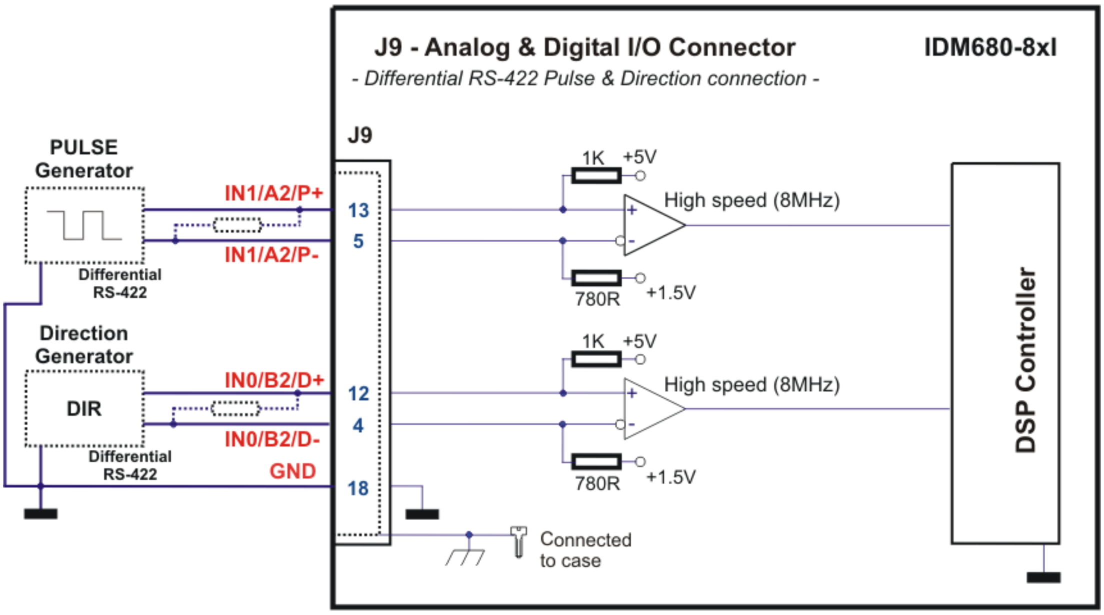


Figure 3.33. *J9 – Differential (RS-422) Pulse & Direction connection*

Remark: *For long (>10 meters) encoder lines add termination resistors (120Ω) close to the drive.*

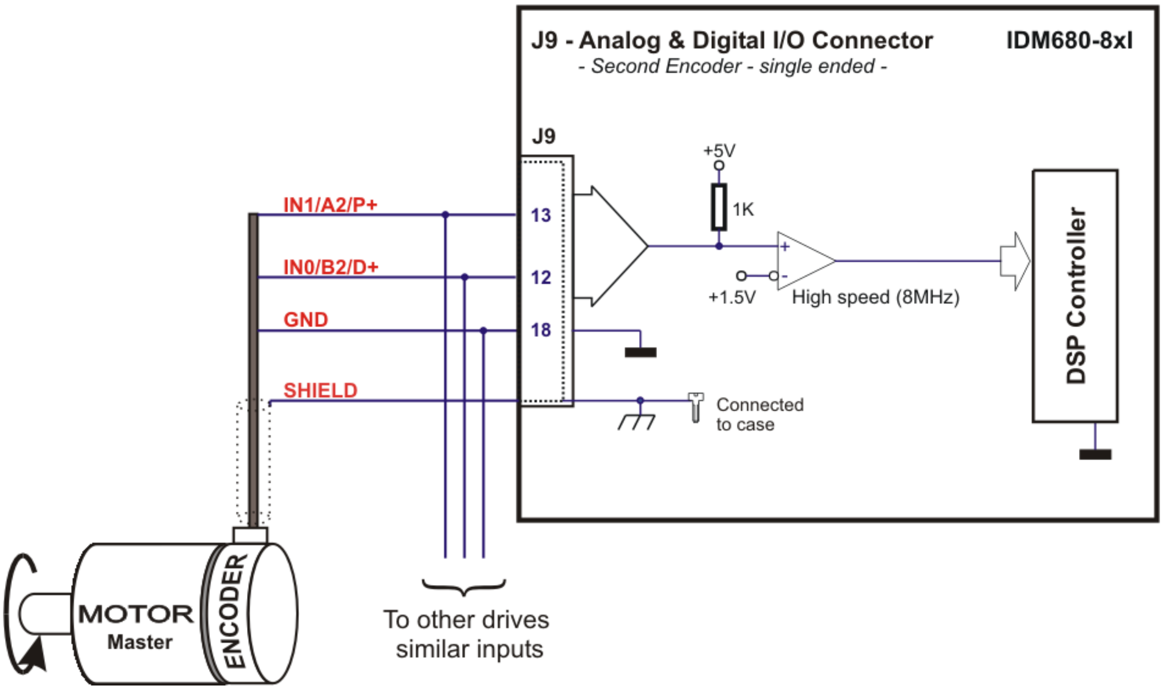


Figure 3.34. *J9 – Second encoder – single ended connection*

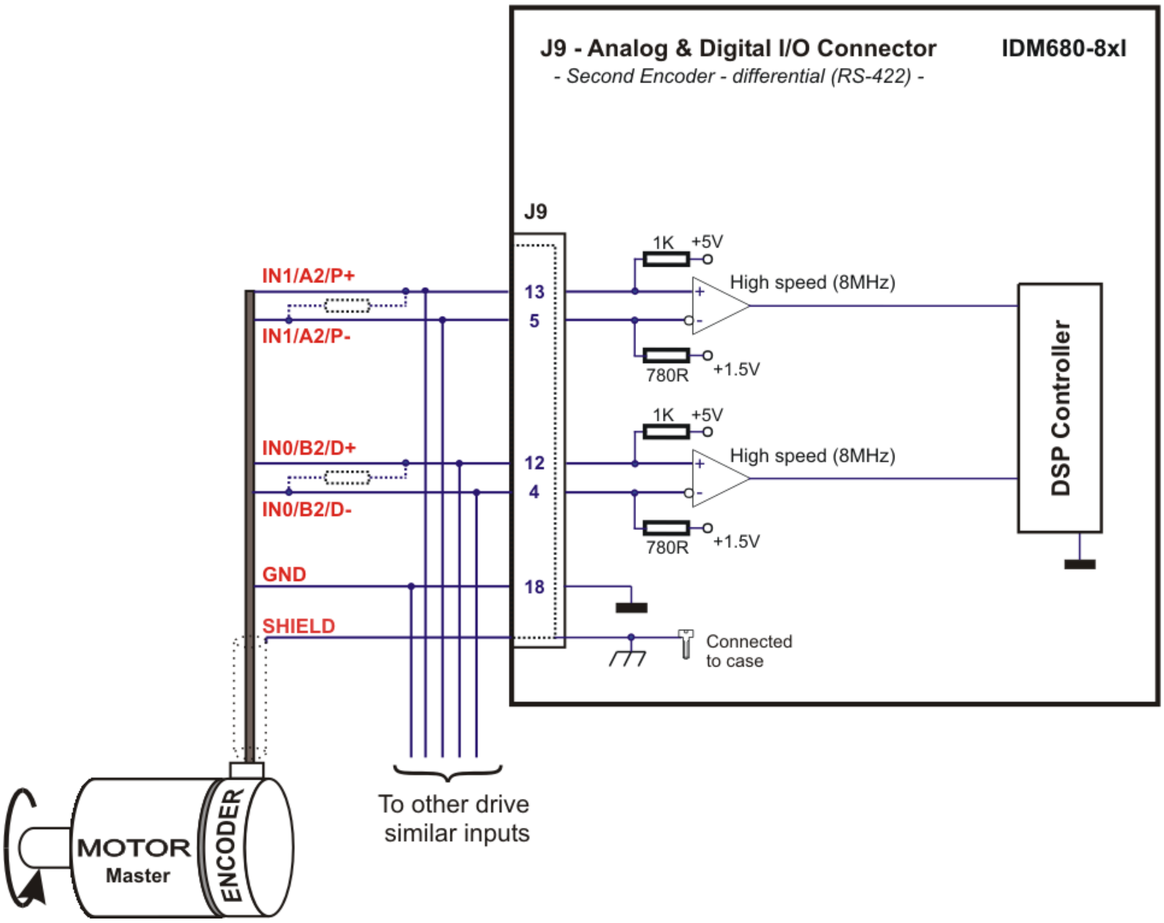


Figure 3.35. *J9 – Second encoder – differential (RS-422) connection*

Remark:

1. *For long (>10 meters) encoder lines add termination resistors (120Ω) close to the drive.*
2. *The master encoder may be supplied with +5V_{DC} from one of the drives. See connector J13 for details.*

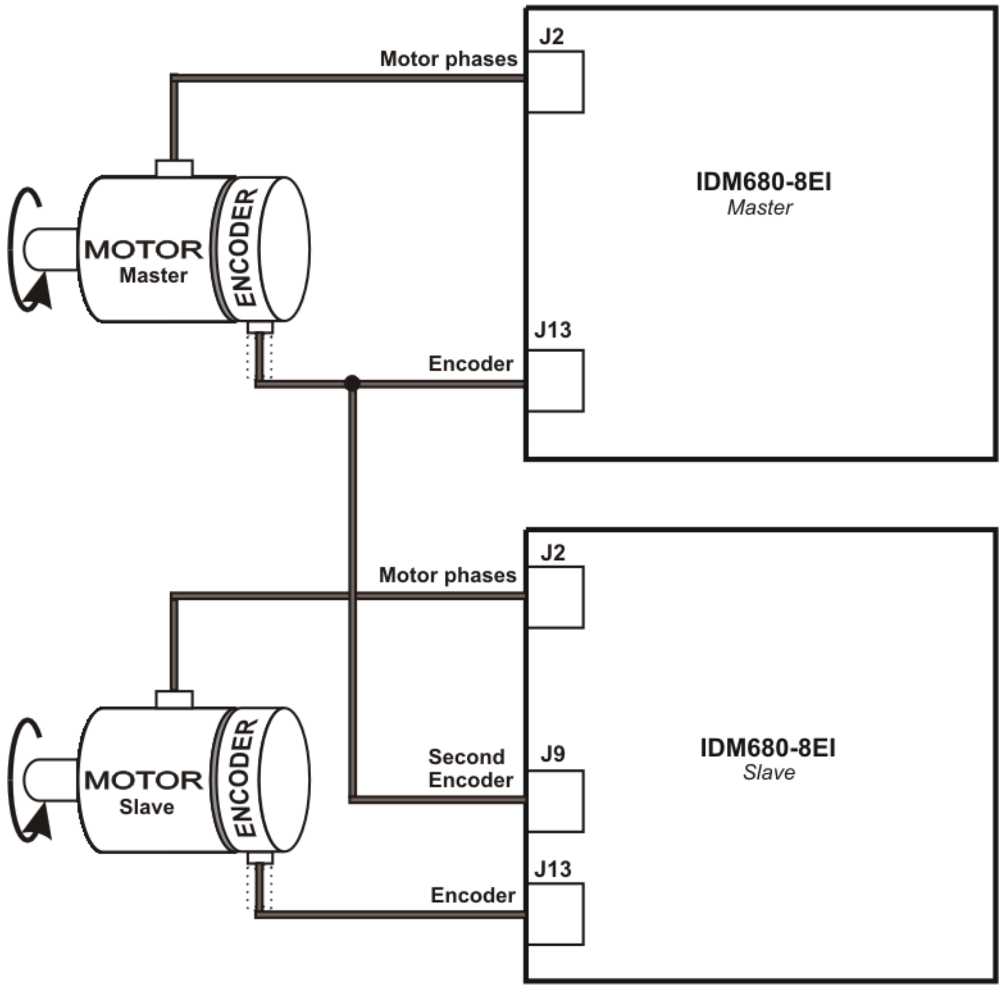


Figure 3.36. *J9 – Master – Slave connection using second encoder input*

3.2.9. Serial Communication – J4 Connector

Pin	Name	Type	Function
2	TxD	O	RS-232 Data Transmission
3	RxD	I	RS-232 Data Reception
5	GND	-	Ground
1,4,6,7,8	n.c.	-	Not Connected
9	+5V	O	Optional supply for handheld terminal (internally generated)

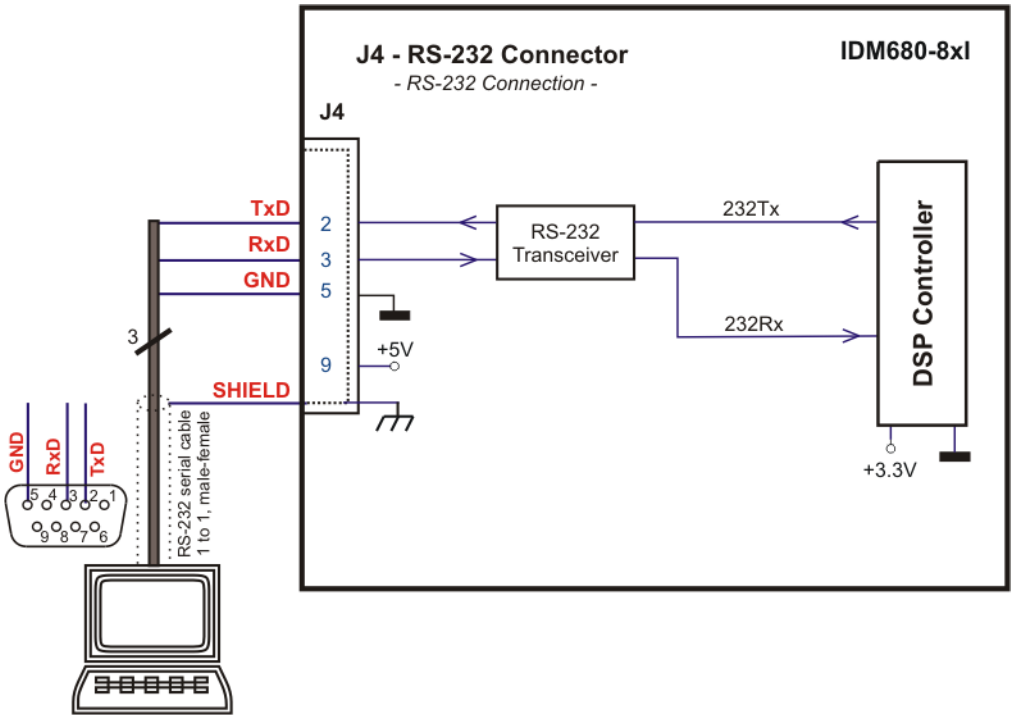


Figure 3.37. *J4 – Serial RS-232 connection*

Remarks:

1. *Use a 9-wire standard 1-to-1 (non-inverting) shielded cable, preferable with metallic or metallized shells (casings)*
2. *On IDM680 drive the electrical ground (GND) and the earth/shield are isolated*

3.2.9.1 Recommendations for RS-232 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 5 of Sub-D 9) must be included inside the shield, like the RxD and TxD signals

b) Do not rely on an earthed PC to provide the IDM680 earth 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 IDS supplies before inserting/removing the RS-232 serial connector.

CAUTION!	***DO NOT CONNECT/DISCONNECT THE RS-232 CABLE WHILE THE DRIVE IS POWERED ON. THIS OPERATION CAN DAMAGE THE DRIVE***

3.2.10. CAN Communication – J10 Connector

Pin	Name	Type	Function
2	CAN_L	I/O	CAN-Bus negative line (negative during dominant bit)
3,6	CAN_GND	-	Reference ground for LO, HI and CAN_V+ signals
5	SHLD	-	Shield; Connected to frame
7	CAN_H	I/O	CAN-Bus positive line (positive during dominant bit)
1,4,8	n.c.	-	Not connected
9	CAN_V+	I	+24 V_{DC} isolated supply input

CAUTION!	***THE CANBUS CONNECTOR SIGNALS ARE ELECTRO-STATICALLY SENSITIVE AND SHALL BE HANDLED ONLY IN AN ESD PROTECTED ENVIRONMENT.***

Remarks:

a) The CAN network requires two 120Ω termination resistors even for short cables. These resistors are not included on the drive.

b) All 4 CAN signals are fully insulated from all other IDM680 circuits (system ground – GND, IO ground – 0VPLC and Earth). Therefore, the CAN network requires a separate supply

3.2.10.1 Recommendations for CAN 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) When total CAN bus length is over 40 meters, it is mandatory to use shielded twisted cables. Connect the cable shield to J10 pin 5 (SHLD)

d) 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.

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

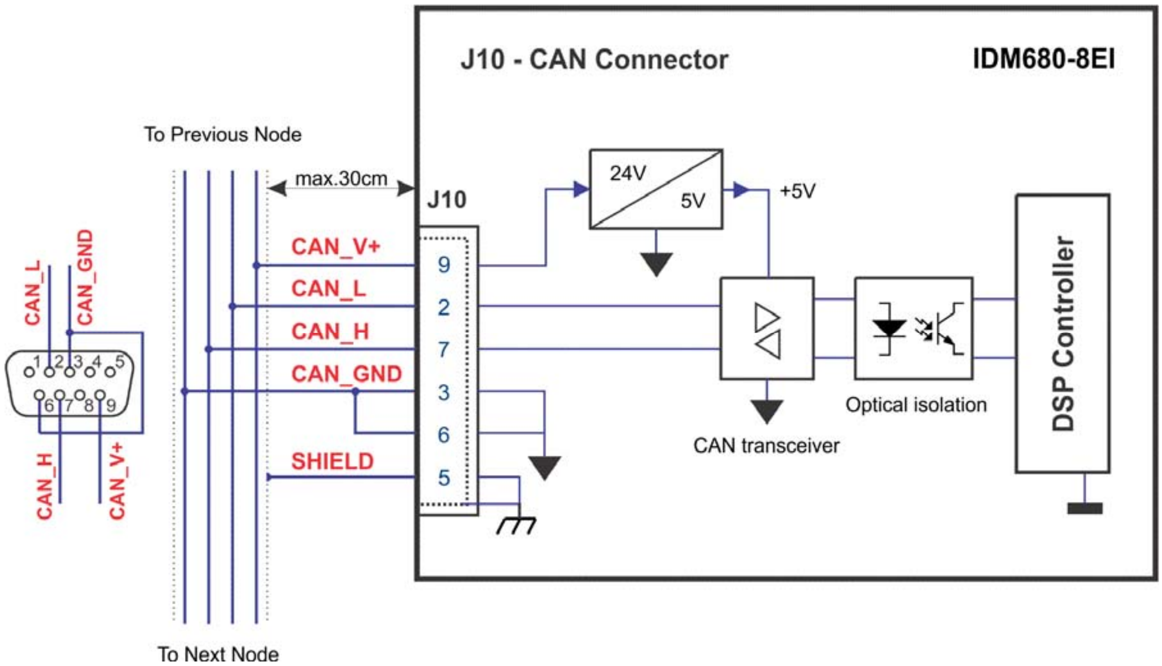


Figure 3.38. *J10 – CAN Connector*

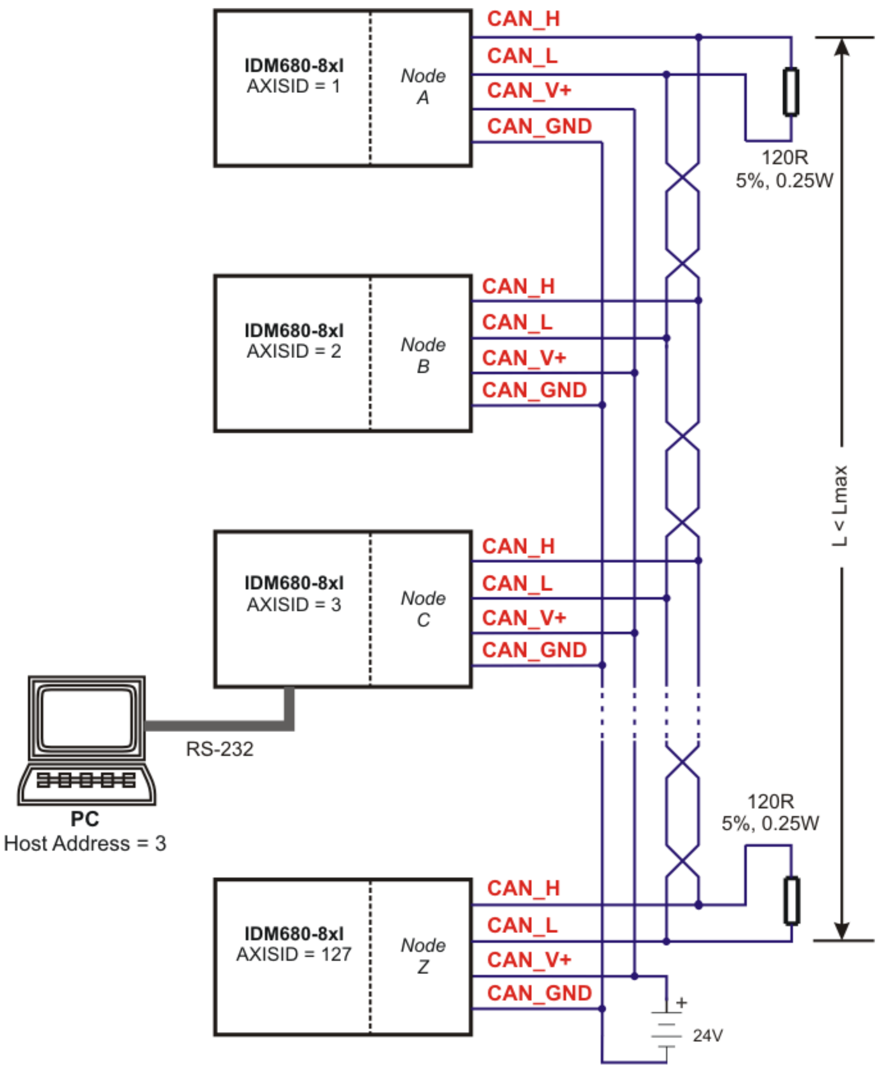


Figure 3.39. *Multiple-Axis CAN network*

3.2.11. Connectors Type and Mating Connectors

Connector	Function	Mating connector
J2	Motor & supply	Phoenix Contact MC 1.5/8-STF-3.5 [1]
J4	Serial	generic 9-pin Sub-D male
J10	CAN	generic 9-pin Sub-D female
J13	Feedback	generic 15-pin High Density Sub-D male
J9	Analog & 24 V digital I/O	generic 26-pin High Density Sub-D male

[1]. The mating connector accepts wires of 0.14 … 1.5 mm^2 (AWG35 … AWG16)

3.3. DIP-Switch Settings

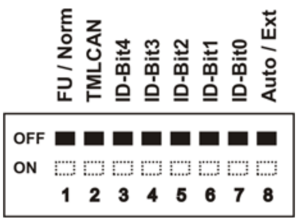


Figure 3.40. *SW1 – DIP Switch*

- **Position 1**: FU / Norm
  - ON: Enable **F**irmware **U**pdate
  - OFF: **Norm**al operation
- **Position 2**: TMLCAN
  - ON: The drive communicates via CAN using TMLCAN protocol (CAN 2.0B)
  - OFF: The drive communicates via CAN using CANopen protocol (CAN 2.0A)
- **Positions 3 … 7**: ID-Bitx.
  Axis ID switches The drive axis/address number is set according with Table 3.1
- **Position 8**: Auto / Ext
  - ON: Sets the drive in AUTORUN mode (only with TMLCAN protocol). After power-on, the drive automatically executes a TML program from its internal E^2ROM.
  - OFF: Sets the drive in External (slave) mode. After power-on, the drive waits for commands from an external device. With CANopen protocol, the drive is always in external mode independently of the switch position

Remark: All switches are sampled at power-up, and the drive is configured accordingly

Table 3.1. *Axis ID / Address configuration*

DIP Switch position					Axis ID
3	**4**	**5**	**6**	**7**	
ID – Bit4	**ID – Bit3**	**ID – Bit2**	**ID – Bit1**	**ID – Bit0**	
OFF	OFF	OFF	OFF	OFF	255
OFF	OFF	OFF	OFF	ON	1
OFF	OFF	OFF	ON	OFF	2
OFF	OFF	OFF	ON	ON	3
OFF	OFF	ON	OFF	OFF	4
OFF	OFF	ON	OFF	ON	5
OFF	OFF	ON	ON	OFF	6
OFF	OFF	ON	ON	ON	7
OFF	ON	OFF	OFF	OFF	8
OFF	ON	OFF	OFF	ON	9
OFF	ON	OFF	ON	OFF	10
OFF	ON	OFF	ON	ON	11
OFF	ON	ON	OFF	OFF	12
OFF	ON	ON	OFF	ON	13
OFF	ON	ON	ON	OFF	14
OFF	ON	ON	ON	ON	15
ON	OFF	OFF	OFF	OFF	16
ON	OFF	OFF	OFF	ON	17
ON	OFF	OFF	ON	OFF	18
ON	OFF	OFF	ON	ON	19
ON	OFF	ON	OFF	OFF	20
ON	OFF	ON	OFF	ON	21
ON	OFF	ON	ON	OFF	22
ON	OFF	ON	ON	ON	23
ON	ON	OFF	OFF	OFF	24
ON	ON	OFF	OFF	ON	25
ON	ON	OFF	ON	OFF	26
ON	ON	OFF	ON	ON	27
ON	ON	ON	OFF	OFF	28
ON	ON	ON	OFF	ON	29
ON	ON	ON	ON	OFF	30
ON	ON	ON	ON	ON	31

Technosoft drives can be set with axis ID values from 1 to 255. In CANopen protocol the maximum axis number is 127. When CANopen protocol is used, the CAN communication sees the drives axis ID modulo 128. The correspondence is given in Table 3.2. In order to avoid having multiple devices with the same Axis ID, do not use in the same CANopen network drives having the same Axis ID in modulo 128. Put in other words, the difference between any two Axis ID values should not be 128.

Remark: *The Axis ID modulo 128 applies only for CAN communication with CANopen protocol. The serial communication and the TMLCAN protocol use the complete axis ID value.*

Table 3.2. *Axis ID modulo 128 seen in CANopen communication*

Real axis ID of the drive	Axis ID seen in CANopen communication
129	1
130	2
…	…
140	12
…	…
200	72
…	…
255	127

When CANopen protocol is selected, the drives can also communicate using *TechnoCAN* protocol – an extension of the CANopen. The TechnoCAN protocol is used to get/send TML commands. TechnoCAN protocol can coexist with CANopen protocol on the same physical network, because it uses ID areas not covered by CANopen. TechnoCAN protocol offers the possibility to inspect the status of ALL Technosoft drives connected on a CANopen network. This operation is done using EasySetUp or EasyMotion Studio and a single RS-232 link with any of the drives from the CANopen network. The inspection / data acquisition can be done while the main application is running.

In TechnoCAN protocol the maximum axis number is 31. When TML commands are exchanged using TechnoCAN protocol, the CAN communication sees the drives axis ID modulo 32. The correspondence is given in Table 3.3. In order to avoid having multiple devices with the same Axis ID, do not use TechnoCAN in a CANopen network with drives having the same Axis ID in modulo 32. Put in other words, the difference between any two Axis ID values should not be a multiple of 32. Note that this restriction applies only when EasySetUp or EasyMotion Studio are used for inspection/debugging. During normal CANopen operation the modulo 32 restriction do not apply.

Table 3.3. *Axis ID modulo 32 seen in TechnoCAN communication*

Real axis ID of the drive	Axis ID seen in CANopen communication
33	1
34	2
…	…
200	8
…	…
255	31

3.4. LED Indicators

LED Color	Function
Green	Lit after power-on when the drive initialization ends. Turned off when an error occurs
Red	Turned on when the power stage error signal is generated or when OUT4 is set low

3.5. First Power-Up

In order to setup the drive for your application you need to communicate with it. The easiest way is via an RS-232 serial link between your PC and the drive. Therefore, before the first power-up, check the following:

- Power supply connections and their voltage levels
- Motor connections
- Serial cable connections
- DIP switch positions: all shall be OFF (not pressed)
- EasySetUp is installed on the PC which is serially connected with the drive (see chapter Step 2. Drive Setup

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 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:

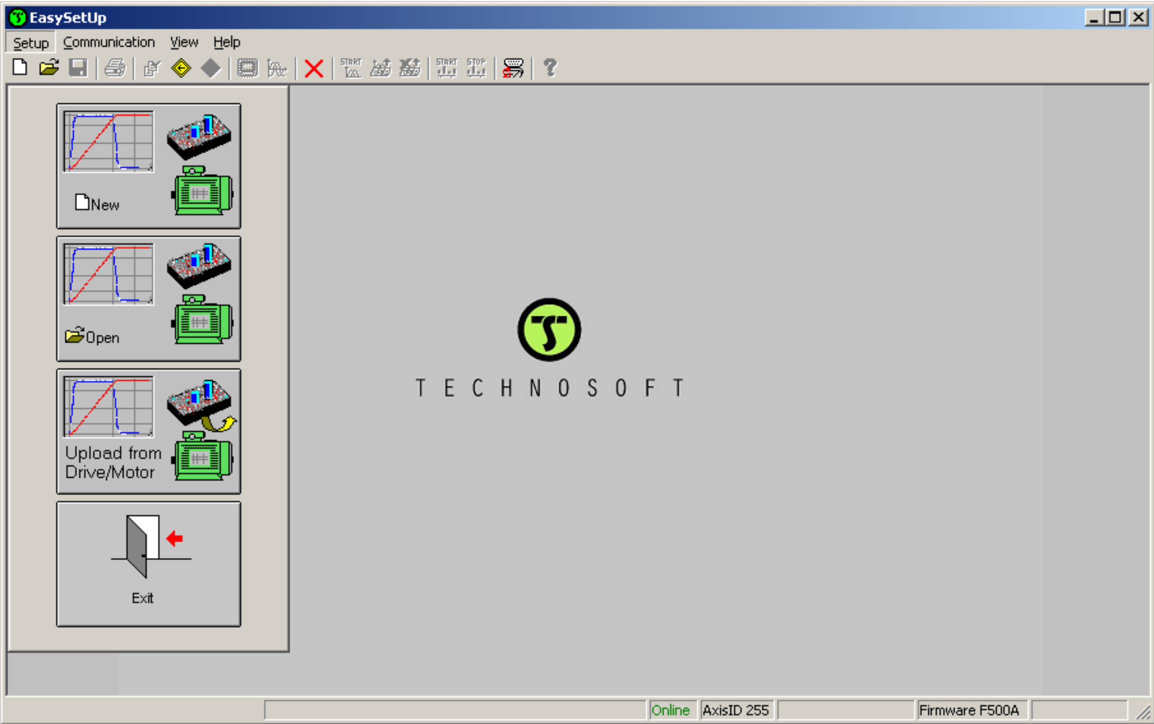
- Use a **CANopen** master
- Use **EasyMotion Studio** to create and download a **TML** program into the drive/motor memory
- Use one of the **TML_LIB** motion libraries to control the drives/motors from your host/master. If your host is a **PC**, TML_LIB offers a collection of high level motion functions 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

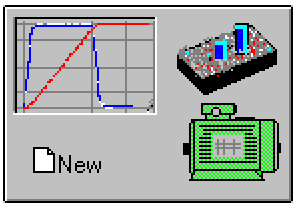
EasySetUp starts with an empty window from where you can create a **New** setup, **Open** a previously created setup which was saved on your PC, or **Upload** the setup from the drive/motor.



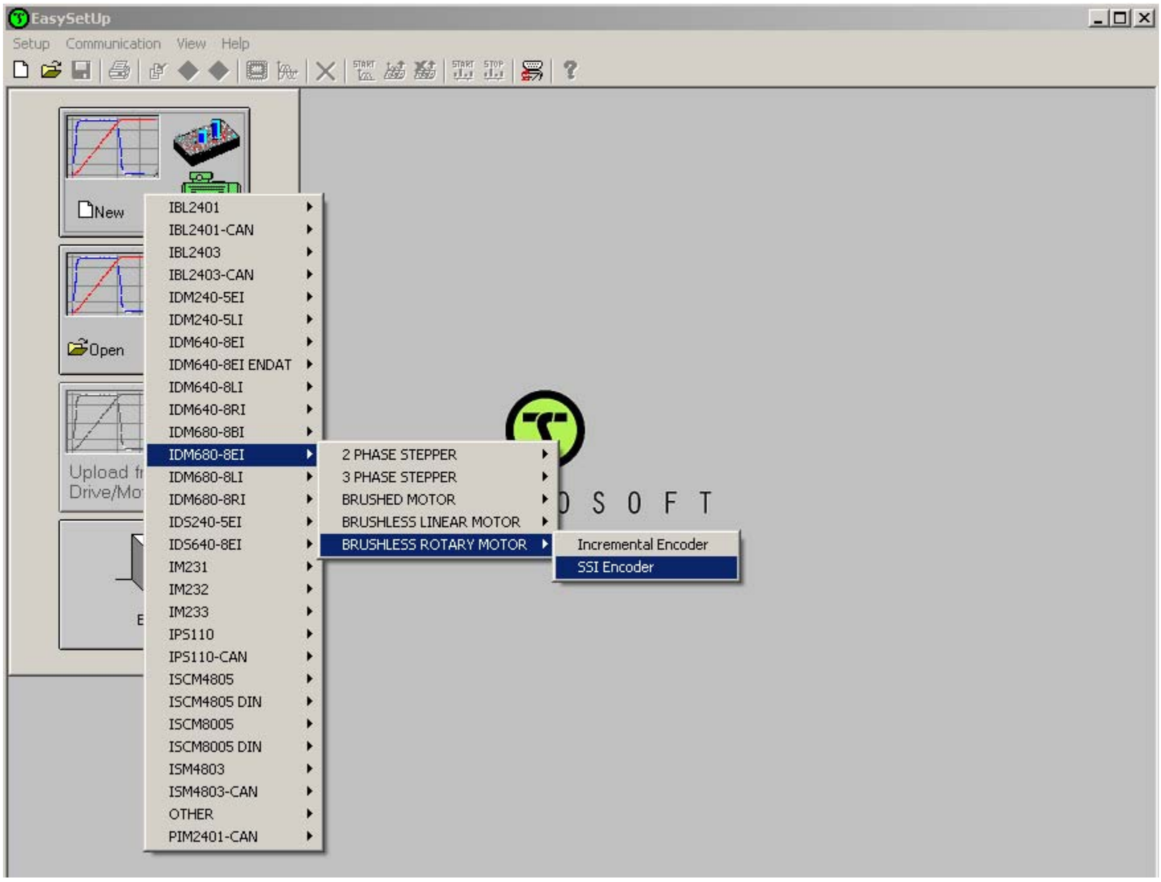
Before selecting one of the above options, you need to establish the communication with the drive you want to commission. Use menu command **Communication | Setup** to check/change your PC communication settings. Press the **Help** button of the dialogue opened. Here you can find detailed information 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

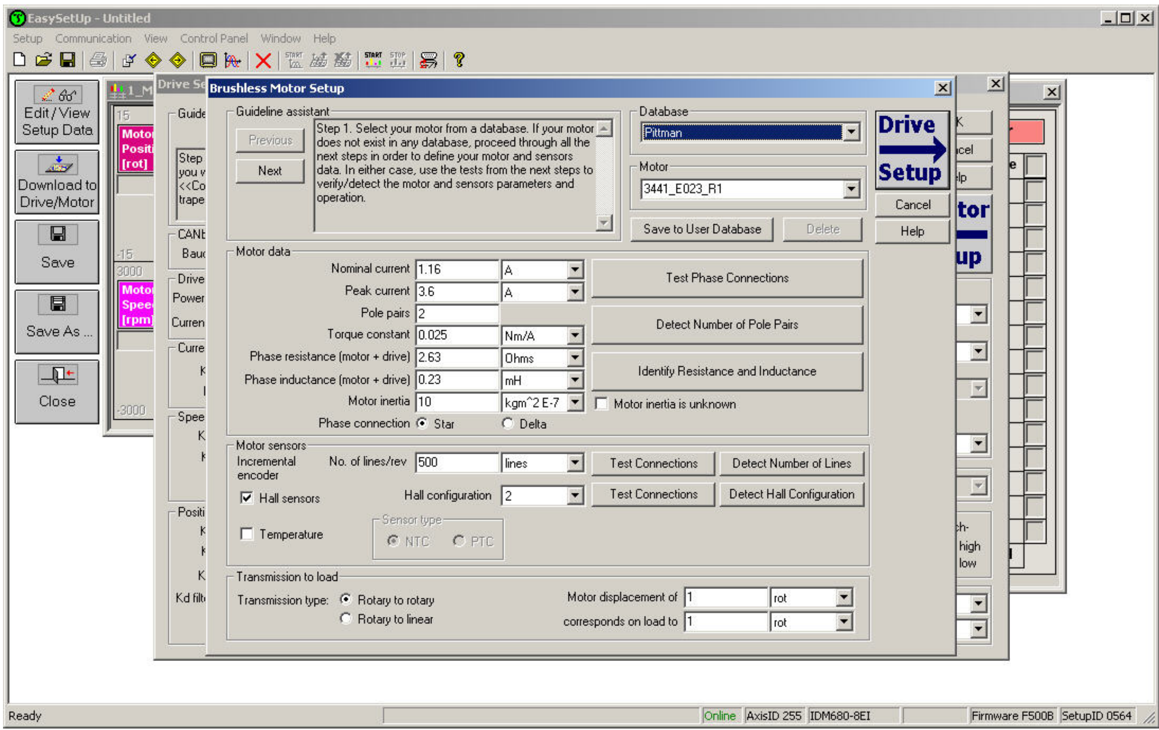
Press **New** button 

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, SSI encoder).

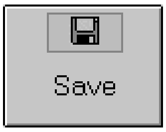
The selection opens 2 setup dialogues: for **Motor Setup** and for **Drive setup** through which you can configure and parameterize a Technosoft drive, plus several predefined control panels customized for the product selected.



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 IDM680 drive can be set in 2 ways:

- Hardware (H/W) – according with the DIP switch selection in the range 1 to 31 or 255 (see 3.3 DIP-Switch Settings)
- Software – any value between 1 and 255, stored in the setup table

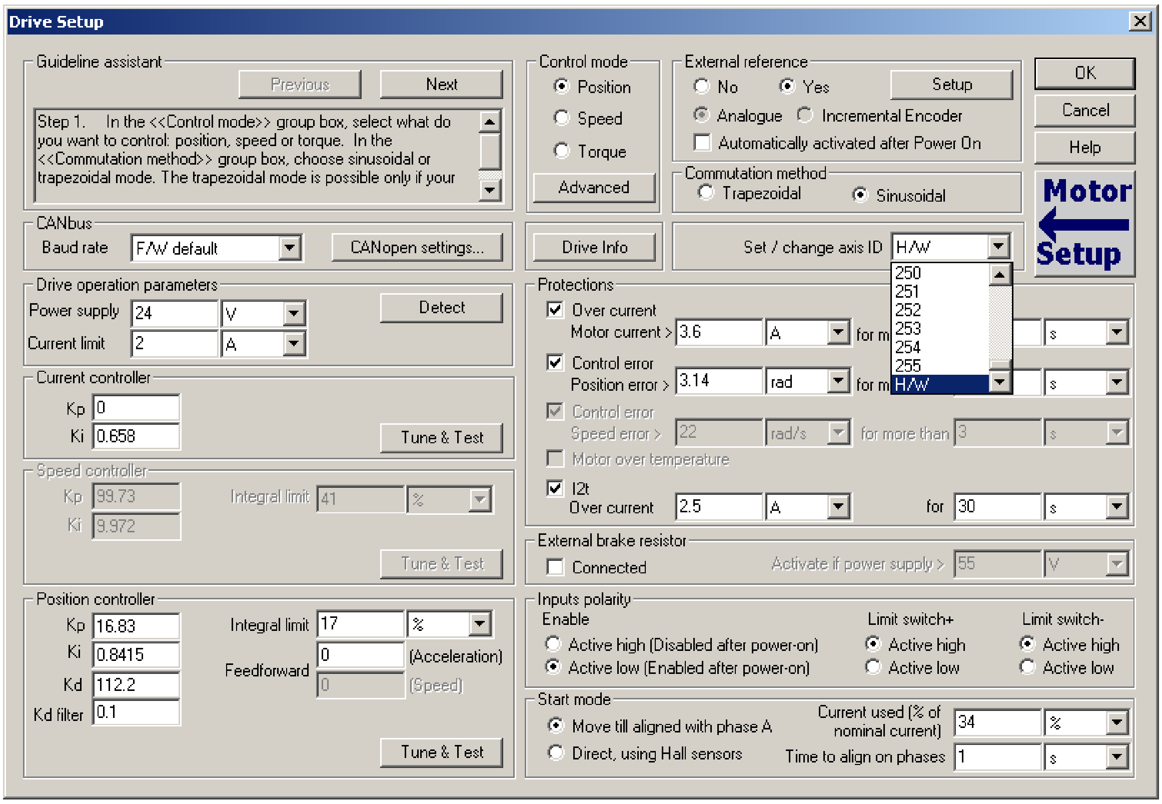
The axis ID is initialized at power on, using the following algorithm:

a) If a valid setup table exists, with the value read from it. This value can be an axis number 1 to 255 or can indicate that axis ID will be set according with DIP switch selection

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 or can indicate that axis ID will be set according with DIP switch selection

c) If there is no axis ID set by a valid setup table, according with DIP switch selection

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 and factor group scaling factors

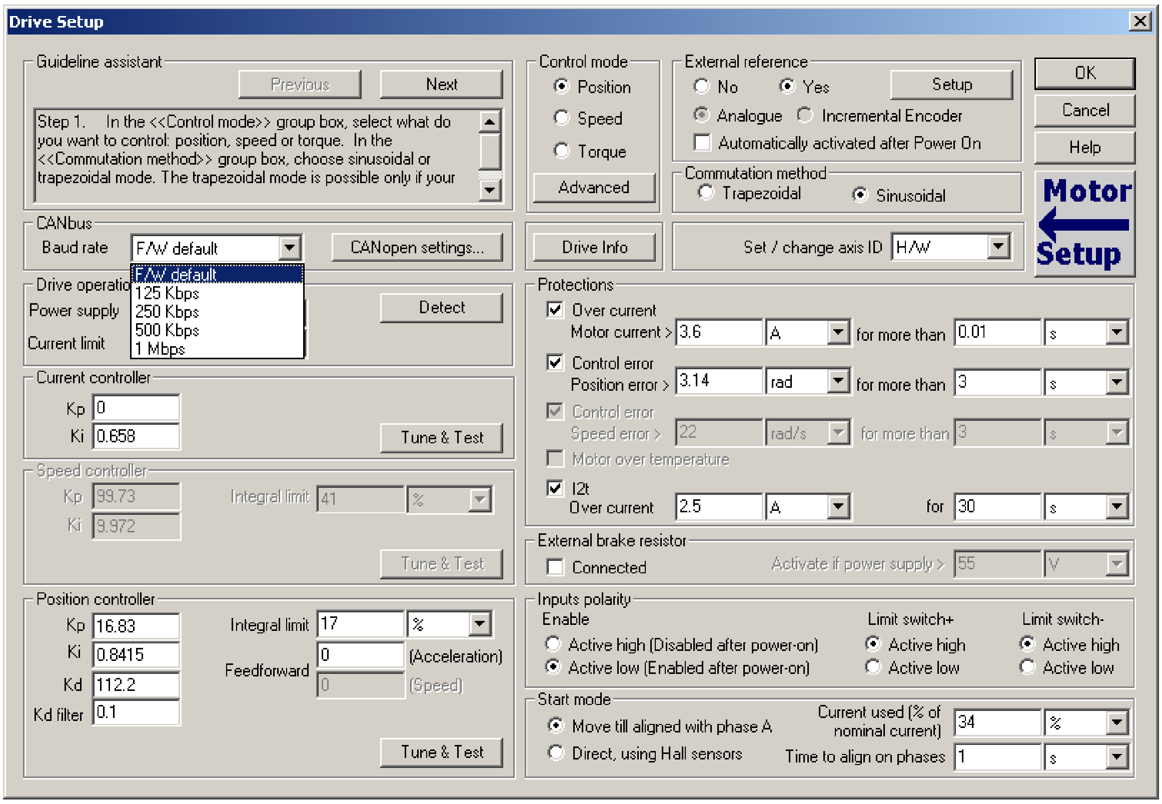
The IDM680 drives can work with the following rates on the CAN: 125kHz, 250kHz, 500KHz, 1MHz. In the Drive Setup dialogue you can choose the initial CAN rate after power on. This information is stored in the setup table. The CAN rate is initialized using the following algorithm:

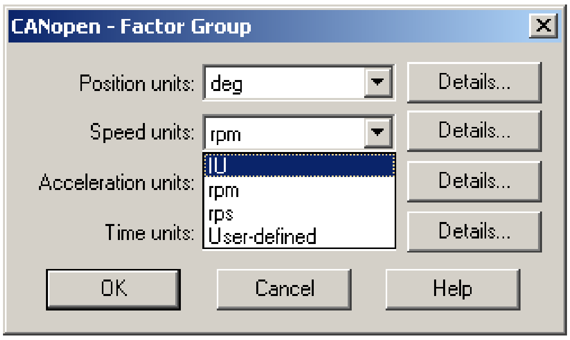
If a valid setup table exists, with the CAN rate value read from it. This can be any of the supported rates or can indicate to use the firmware default (F/W default) value, which is 500kHz

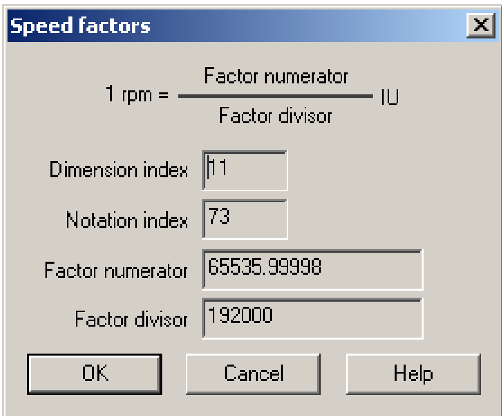
If the setup table is invalid, with the last CAN rate value set with a valid setup table. This can be any of the supported rates or can indicate to use the firmware default (F/W default) value

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

By pressing the CANopen Settings button, you can choose the initial values after power on for the CANopen factor group settings. The factor group settings describe the scaling factors for position, speed, acceleration and time objects. In the factor group dialogue you can select the units to use when writing to these objects or reading them. You can either choose one of the standard units defined in the CANopen standard DSP-402 or define your own unit.







In the last case, it is your responsibility to set the factor numerator and divisor as well as its dimension and notation index. The factor group settings are stored in the setup table. By default the drive uses its internal units. The correspondence between the drive internal units and the SI units is presented in chapter 6 Scaling Factors.

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

The IDM680 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.IDM680.UM.xxxx)

5.1.1. DS-301 Communication Profile Overview

The IDM680 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 IDM680 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 IDM680 supports both SYNC consumer and producer.

- **Time Stamp Object (TIME)**

The Time Stamp Object is not supported by the IDM680 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 IDM680 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 IDM680: 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

In order to take full advantage of the powerful Technosoft Motion Language (TML) built into the IDM680, 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 IDM680 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**
- **Electronic camming 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 IDM680 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 to create and download 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, etc.)
- Change the motion modes and/or the motion parameters
- Execute homing sequences
- 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.

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.

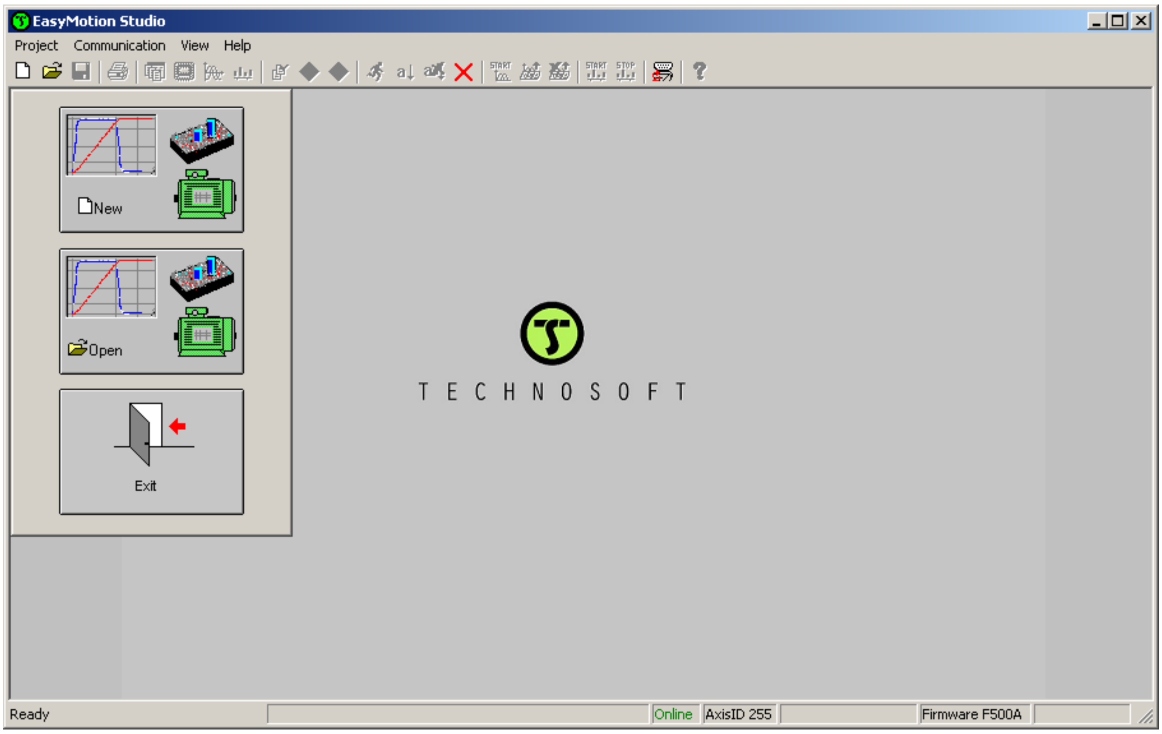
In the first case, each application has a different axis number corresponding to the axis ID of the drives/motors from the network. All data exchanges are done with the drive/motor having the same address as the selected application. In the second case, all the applications have the same axis number.

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.

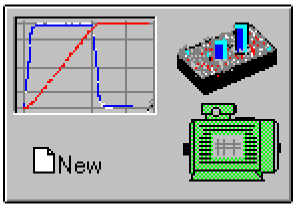
The motion component contains the motion sequences to do. These are described via a TML (Technosoft Motion Language) program, which is executed by the drives/motors built-in motion controller.

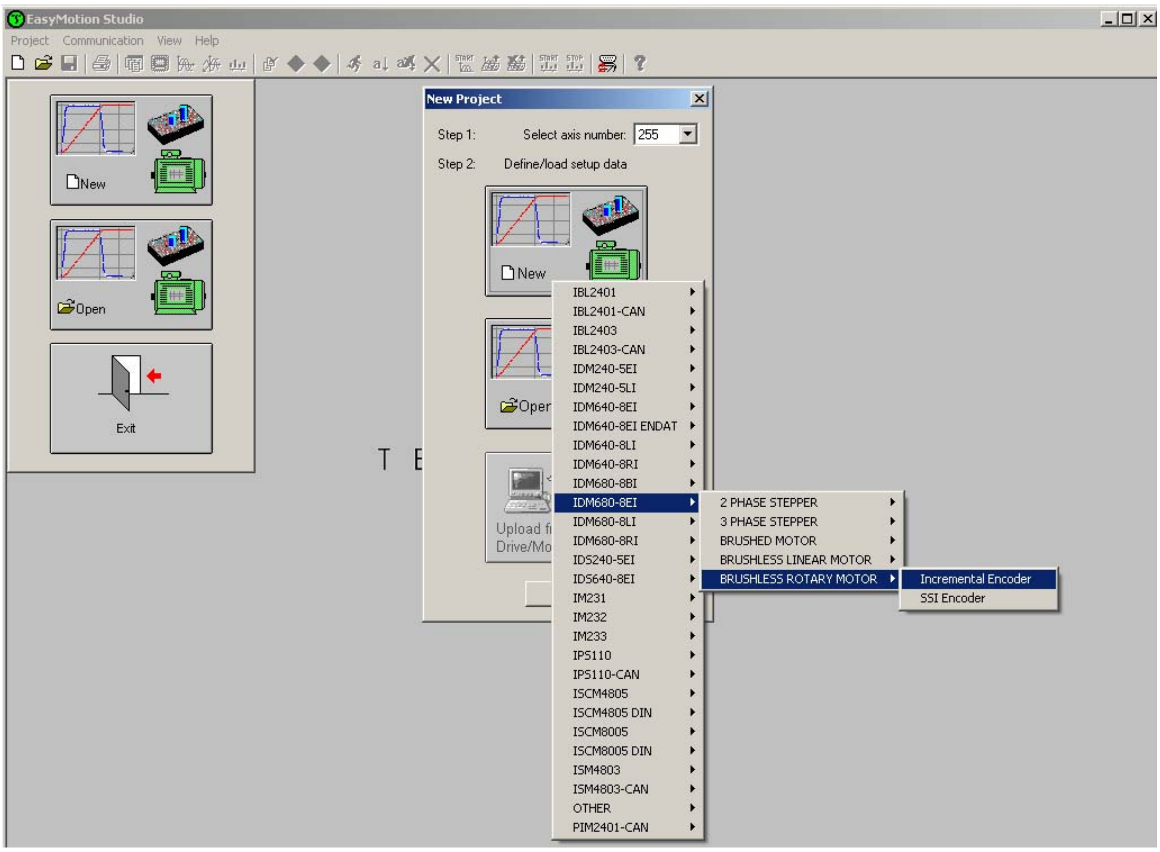
5.2.3.1 Create a new project

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

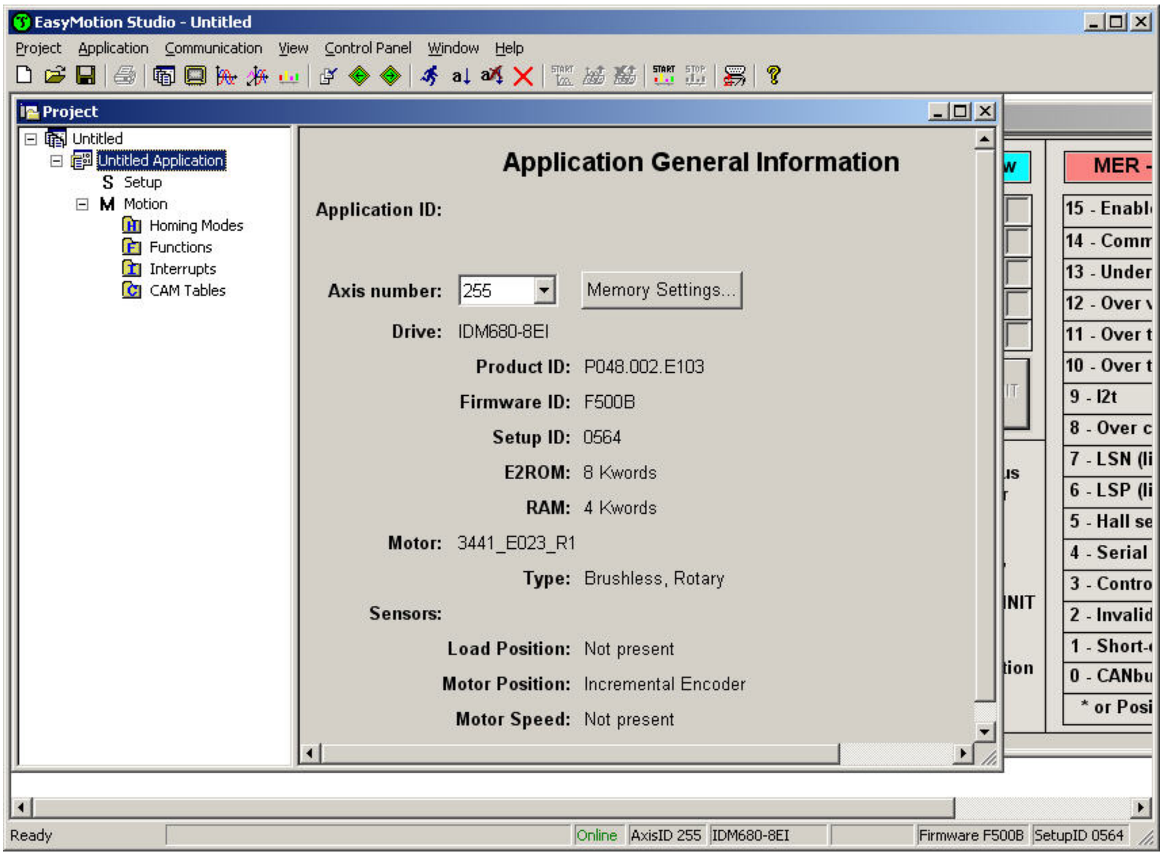


When you start a new project, EasyMotion Studio automatically creates a first application. Additional applications can be added later. You can duplicate an application or insert one defined in another project.

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 having all the axis ID switches OFF (see 3.3 DIP-Switch Settings). 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: SSI encoder, 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).



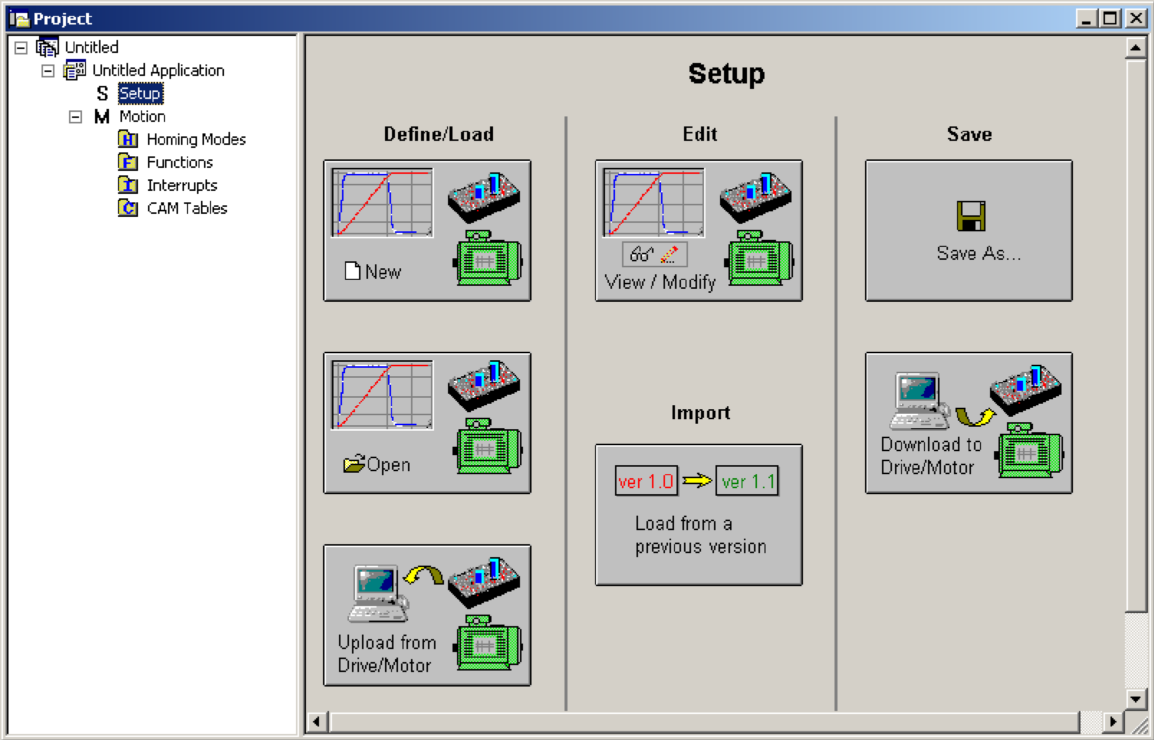
5.2.3.2 Step 2 Establish communication

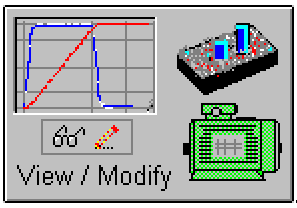
If you have a drive/motor connected with your PC, now its time to check the communication. Use menu command **Communication | Setup** to check/change your PC communication settings. Press the **Help** button of the dialogue opened. Here you can find detailed information about how to setup your drive/motor 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:

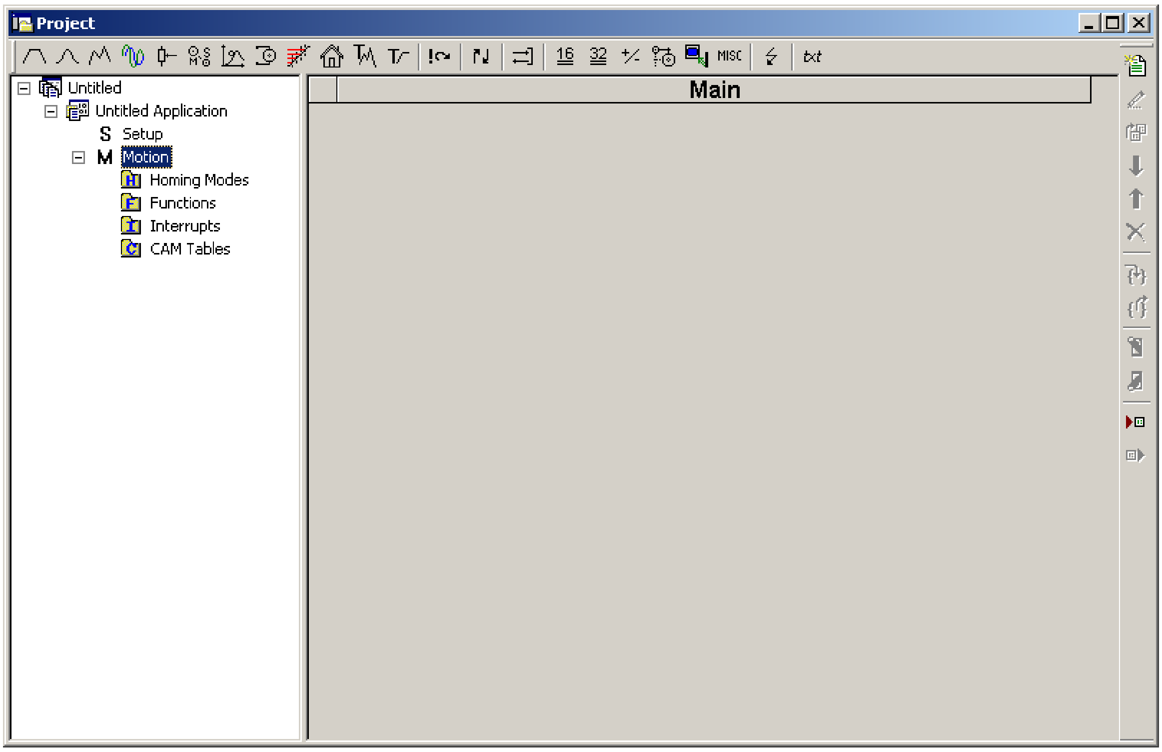
create a new setup data by going through the motor and drive dialogues

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. The TML program may also include cam tables used for electronic camming applications.

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 with TML

Due to its embedded motion controller, an IDM680 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.CANopen.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 IDM680 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, 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

Apart from the standard modes of operation of DSP-402, the IDM680 offers others like: electronic gearing, electronic camming, external modes with analogue or digital reference etc. When electronic camming is used, the cam tables can be loaded in the following ways:

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

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

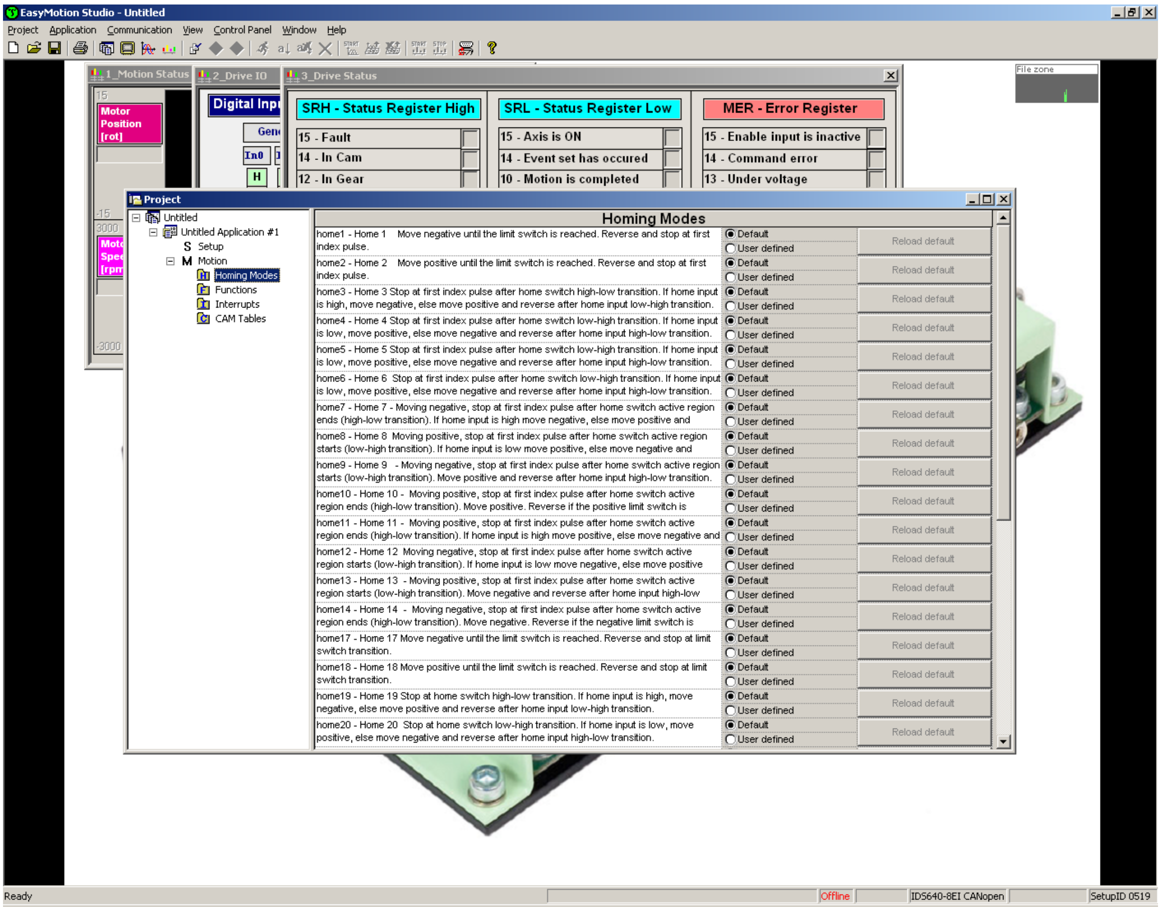
c) The cam points are stored in the drive EEPROM and during the drive initialization (transition to Ready to Switch ON status) are automatically copied from EEPROM to the active RAM

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

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

5.3.4. Customizing the Homing Procedures

The IDM680 supports all homing modes defined in DSP-402 device profile. If needed, any of these homing modes can be customized. In order to do this you need to select the Homing Modes from your EasyMotion Studio application and in the right side to set as "User defined" one of the Homing procedures. Following this operation the selected procedure will occur under Homing Modes in a subtree, with the name *HomeX* where X is the number of the selected homing.



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

5.3.5. Customizing the Drive Reaction to Fault Conditions

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++, 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 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**[1] is:

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

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

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

6.1.2. Brushless motor with sine/cosine encoder on motor

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

For rotary motors:

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

For linear motors:

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

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

where:

Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution

Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation

Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period

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

6.1.3. Brushless motor with absolute SSI/BiSS encoder on motor

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

$$Load\_Position[SI] = \frac{2 \times \pi}{2^{No\_bits\_resolution} \times Tr} \times Motor\_Position[IU]$$

where:

No_bits_resolution – is the SSI/BiSS encoder resolution in bits per revolution

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

6.1.4. 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:

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

where:

resolution – is the motor position resolution

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

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

6.1.5. Brushless motor with resolver

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 4096 counts per turn. The correspondence with the load **position in SI units**[1] is:

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

where:

resolution – is the motor position resolution

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

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

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

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

where:

No_encoder_lines – is the encoder number of lines per revolution

6.1.7. DC brushed motor with absolute SSI encoder on load and tacho on motor

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

$$Load\_Position[SI] = \frac{2 \times \pi}{2^{No\_bits\_resolution}} \times Load\_Position[IU]$$

where:

No_bits_resolution – is the SSI encoder resolution in bits per revolution

6.1.8. 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:

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

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

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

6.1.9. 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** [1] is:

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

where:

No_encoder_lines – is the motor encoder number of lines per revolution

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

6.1.10. 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:

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

6.2. 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**[1] is:

---

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

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

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

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 sine/cosine encoder on motor

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

For rotary motors:

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

For linear motors:

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

where:

Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution

Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period

Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation

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

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

---

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

6.2.3. Brushless motor with absolute SSI/BiSS encoder on motor

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

$$\text{Load\_Speed[SI]} = \frac{2 \times \pi}{2^{\text{No\_bits\_resolution}} \times \text{Tr} \times \text{T}} \times \text{Motor\_Speed[IU]}$$

where:

No_bits_resolution – is the SSI/BiSS encoder resolution in bits 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”

6.2.4. 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:

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

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”

6.2.5. Brushless motor with resolver

The internal speed units are counts / (slow loop sampling period). 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 4096 counts per turn. The correspondence with the load **speed in SI units**[2] is:

---

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

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

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

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"

6.2.6. 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"

6.2.7. DC brushed motor with absolute SSI 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:

$$Load\_Speed[SI] = \frac{2 \times \pi}{2^{No\_bits\_resolution} \times T} \times Load\_Speed[IU]$$

where:

No_bits_resolution – is the SSI encoder resolution in bits 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"

6.2.8. DC brushed motor with tacho on motor

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

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

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

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]

6.2.9. 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:

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

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”

6.2.10. 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”.

6.2.11. 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**[1] is:

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

where:

No_encoder_lines – is the motor encoder number of lines per revolution

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

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

6.3. 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**[2] is:

For rotary motors:

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

For linear motors:

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

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

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

6.3.2. Brushless motor with sine/cosine encoder on motor

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

For rotary motors:

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

For linear motors:

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

where:

Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution

Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period

Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation

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

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

6.3.3. Brushless motor with absolute SSI/BiSS encoder on motor

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

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

where:

No_bits_resolution – is the SSI/BiSS encoder resolution in bits per revolution

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

6.3.4. Brushless motor with linear Hall signals

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

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

where:

resolution – is the motor position resolution

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

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

6.3.5. Brushless motor with resolver

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 4096 counts per turn. The correspondence with the load **acceleration in SI units** is:

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

where:

resolution – is the motor position resolution

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

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

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

6.3.6. 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:

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

where:

No_encoder_lines – is the encoder number of lines per revolution

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

6.3.7. DC brushed motor with absolute SSI 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**[1] is:

$$Load\_Acceleration[SI] = \frac{2 \times \pi}{2^{No\_bits\_resolution} \times T^2} \times Load\_Acceleration[IU]$$

where:

No_bits_resolution – is the SSI encoder resolution in bits 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"

6.3.8. DC brushed motor with tacho on motor

When only a tachometer is mounted on the motor shaft, the internal acceleration units are A/D converter bits / (slow loop sampling period). The correspondence with the load acceleration in SI units is:

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

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]

[1] 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"

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

6.3.9. 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**[1] is:

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

where:

No_steps – is the number of motor steps per revolution

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

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

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

6.3.10. 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:

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

For rotary-to-linear transmission:

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

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

---

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

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

6.3.11. 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**[1] is:

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

where:

No_encoder_lines – is the 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**[2] is:

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

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

---

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

[2] 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 sine/cosine encoder on motor

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

For rotary motors:

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

For linear motors:

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

where:

Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution

Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period

Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation

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

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

6.4.3. Brushless motor with absolute SSI/BiSS encoder on motor

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

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

where:

No_bits_resolution – is the SSI/BiSS encoder resolution in bits per revolution

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

[1] 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”

6.4.4. 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:

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

where:

resolution – is the motor position resolution

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

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

6.4.5. Brushless motor with resolver

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 4096 counts per turn. The correspondence with the load **jerk in SI units**[1] is:

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

where:

resolution – is the motor position resolution

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

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

6.4.6. 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:

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

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

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"

6.4.7. DC brushed motor with absolute SSI 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:

$$\text{Load\_Jerk[SI]} = \frac{2 \times \pi}{2^{\text{No\_bits\_resolution}} \times T^2} \times \text{Load\_Jerk[IU]}$$

where:

No_bits_resolution – is the SSI encoder resolution in bits 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"

6.4.8. 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**[1] is:

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

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"

6.4.9. 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:

---

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

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

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”.

6.4.10. 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:

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

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

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:

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

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

6.9. Drive temperature units

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

$$\text{Drive temperature [°C]} = \frac{3\text{[V]} \times \text{DriveTemperature[IU]}}{65520 \times \text{Sensor\_gain[V/°C]}} - \frac{\text{Sensor\_output\_0°C[V]}}{\text{Sensor\_gain[V/°C]}}$$

where:

Sensor_gain – is the temperature sensor gain

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

6.10. Master position units

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

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

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

where:

No_encoder_lines – is the master number of encoder lines per revolution

6.11. Master speed units

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

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

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

where:

No_encoder_lines – is the master number of encoder lines per revolution

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

6.12. Motor position units

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**[1] is:

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

For linear motors: $$Motor\_Position[SI] = Encoder\_accuracy \times Motor\_Position[IU]$$

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

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

6.12.2. Brushless motor with sine/cosine encoder on motor

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

For rotary motors:

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

For linear motors:

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

where:

Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution

Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation

Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period

6.12.3. Brushless motor with absolute SSI/BiSS encoder on motor

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

$$Motor\_Position[SI] = \frac{2 \times \pi}{2^{No\_bits\_resolution}} \times Motor\_Position[IU]$$

where:

No_bits_resolution – is the SSI/BiSS encoder resolution in bits per revolution

6.12.4. 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:

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

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

where:

resolution – is the motor position resolution

6.12.5. Brushless motor with resolver

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 4096 counts per turn. The correspondence with the motor position in SI units is:

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

where:

resolution – is the motor position resolution

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

The motor position is not computed.

6.12.7. DC brushed motor with absolute SSI encoder on load & tacho on motor

The motor position is not computed.

6.12.8. 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:

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

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.

6.12.9. 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.

---

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

6.12.10. 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**[1] is:

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

For linear motors: $$Motor\_Speed[SI] = \frac{Encoder\_accuracy}{T} \times Motor\_Speed[IU]$$

where:

No_encoder_lines – is the rotary encoder number of lines per revolution

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

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

6.13.2. Brushless motor with sine/cosine encoder on motor

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

For rotary motors:

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

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

For linear motors:

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

where:

Enc_periods – is the rotary encoder number of sine/cosine periods or lines per revolution

Encoder_accuracy – is the linear encoder accuracy in [m] for one sine/cosine period

Interpolation – is the interpolation level inside an encoder period. Its a number power of 2 between 1 an 256. 1 means no interpolation

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

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

6.13.3. Brushless motor with absolute SSI/BiSS encoder on motor

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

$$Motor\_Speed[SI] = \frac{2 \times \pi}{2^{No\_bits\_resolution} \times T} \times Motor\_Speed[IU]$$

where:

No_bits_resolution – is the SSI/BiSS encoder resolution in bits 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"

6.13.4. 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:

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

where:

resolution – is the motor position resolution

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

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"

6.13.5. Brushless motor with resolver

The internal motor speed units are counts / (slow loop sampling period). 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 4096 counts per turn. The correspondence with the motor speed in SI units is:

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

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"

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

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

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

where:

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

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

6.13.7. DC brushed motor with absolute SSI encoder on load & tacho on motor

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

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

where:

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

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

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

6.13.8. 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]

6.13.9. 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**[1] is:

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

where:

No_steps – is the number of motor steps per revolution

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

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

6.13.10. 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:

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

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”.

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

7. Memory Map

IDM680 has 2 types of memory available for user applications: 4K×16 SRAM and 8K×16 serial E^2ROM.

The SRAM memory is mapped in the address range: 9000h to 9FFFh. It can be used to download and run a TML program, to save real-time data acquisitions and to keep the cam tables during run-time.

The E^2ROM is mapped in the address range: 4000h to 5FFFh. It is used to keep in a non-volatile memory the TML programs, the cam tables and the drive setup information.

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

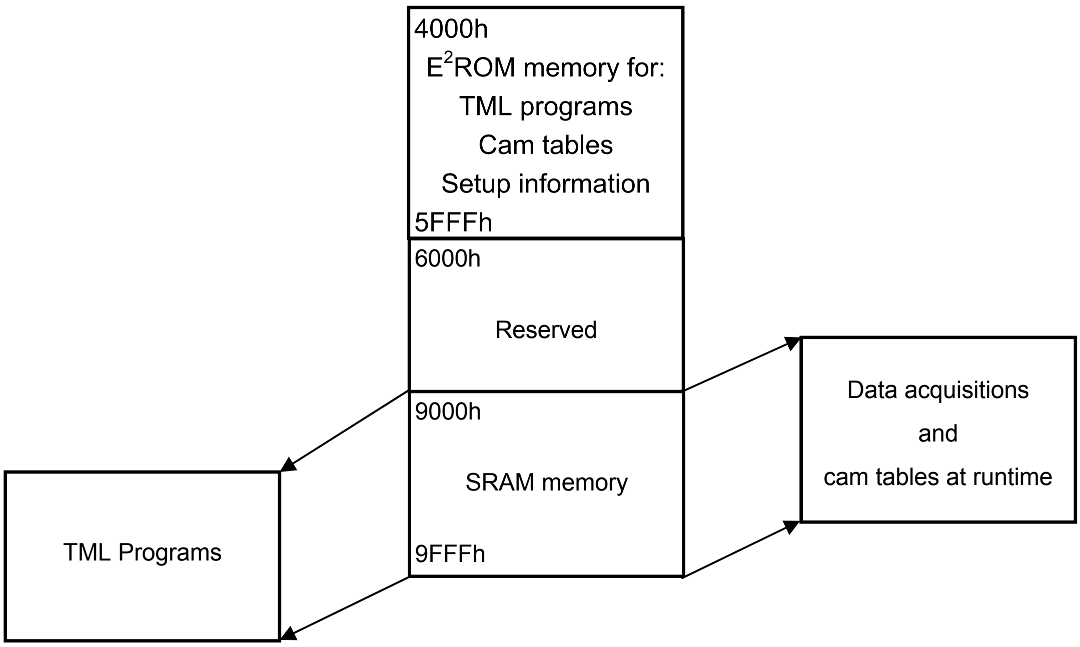


Figure 7.1. *IDM680 Memory Map*

T E C H N O S O F T