192-300303N5
23.07.2024

GVI

Global Vehicle Inverter

Configuration Manual

ENGINEERING YOUR SUCCESS.

Non-warranty clause

The contents of this manual have been verified against the associated hardware and software. Although every effort has been taken to ensure the accuracy of this document it may be necessary, without notice, to make amendments or correct omissions. Parker Hannifin Manufacturing cannot accept responsibility for damage, injury, or expenses resulting therefrom.

Production site:

Germany

Parker Hannifin Manufacturing Germany GmbH & Co. KG
Electromechanical & Drives Division Europe [EMDE]
Robert-Bosch-Strasse 22
77656 Offenburg (Germany)
Tel.: + 49 (0781) 509-0
Fax: + 49 (0781) 509-98176
Internet: **www.parker.com/eme http://www.parker.com/eme**
E-mail: **EM-Motion@parker.com**

Certified according to ISO 9001:2015

Parker Hannifin Manufacturing Germany GmbH & Co KG - Sitz: Bielefeld - Amtsgericht: Bielefeld HRA 15699
Partner liable to unlimited extent: Parker Hannifin GmbH, Sitz Bielefeld, Amtsgericht Bielefeld HRB 35489
Geschäftsführung der PARKER Hannifin GmbH: Dr.-Ing. Hans-Jürgen Haas, Kees Veraart, Chairman of the board: Dr.-Ing. Gerd Scheffel

Table of Contents

1 Introduction

1.1 About this document

1.1.1 Definitions

Throughout this documentation the product Global Vehicle Inverter is referred to as “The motor controller” or GVI.

GVI is a family of motor controllers for use in systems with 24-650 DC (nominal) supply and power levels from 4.4 to 398 kVA. GVI frame sizes C, D, E are referred to as Low Voltage (LV) devices, frame sizes G and H are considered as High Voltage (HV) Devices. The GVI is suitable for most electric vehicle applications.

1.1.2 Terms and abbreviations

Application	A customer specific use of GVI hardware and software
Application software	Customer-specific configuration of GVI parameters
CAN	Controller Area Network
EMC	Electromagnetic compatibility
EMF	Electromotive force
ESD	Electrostatic discharge
GVI	Motor controller / Drive / Inverter / Controller
HV	High voltage
HVIL	Hazard/High Voltage Interlock Loop
HVL	High Voltage Level
HW	Hardware
HW ID	Hardware ID
I^2t	Overload protection based on a thermal model
Integration	Software integration is the implementation of all necessary functions to control the system for a customer-specific application. Hardware integration is the physical installation of the GVI in a customer’s equipment
MCU	Motor control unit
MTO	Motor Torque Off
NMT	Network Management
OD	Object Dictionary
OEM	Original equipment manufacturer
OVP	Overvoltage Protection
PMAC	Permanent magnet motor
RMA	Return of Material Authorization
SSC	Safe Short Circuit
SW	Software
VCU/VMC	Vehicle Control Unit / Vehicle Master Controller
WEG	Water Ethylene Glycol mixture

1.1.3 This revision

This revision replaces all previous revisions of this document. Parker has made every effort to ensure that this document is complete and accurate at the time of printing. In accordance with our policy of continuous product improvement, all data in this document is subject to change or correction without prior notice.

1.1.4 Scope

The GVI is a software configurable device. In a CAN based system, the GVI setup and operation can be managed by a vehicle master controller communicating over the CAN Bus.

This document presents the general procedure for the startup and verification of a GVI following installation in an operational system (vehicle).

Before continuing with the configuration, ensure the Start-up and Commissioning section from the hardware manual (see chapter 1.1.6) has been completed and is fully understood. It is also helpful to have the **OD**, the list of all parameters and variables the GVI has accessible via the CAN bus, available when reading through this manual.

1.1.5 Warning, caution and information notices

Special attention must be paid to the information presented in warning, caution and information notices when they appear in this manual. Definitions of caution, warning and information notices are shown below:

WARNING

This section describes the risk of the hazard, for example High voltage - risk of personnel injury.

A Warning informs the user of a hazard or potential hazard that could result in serious or fatal injury and damage to the equipment if the precautions or instructions given in the warning notice are not observed/followed.

CAUTION

This section describes the risk of the hazard, for example Risk of damage to equipment.

A Caution informs the user of a hazard or potential hazard that could result in damage to the equipment if the precautions or instructions given in the caution notice are not observed/followed.

NOTE

A note contains supplemental information or references to supplemental information on a topic.

1.1.6 Related documents

For more information about the inverter, see the following related documents.

#	Document	Description
1	GVI Object Dictionary	Available from Parker as an HTML file
2	Product Manual for GVI-C D E	Parker Reference 192-300300Nx
3	Product Manual for GVI-G-H	Parker Reference 192-300302Nx
4	GVI Configuration Tool Quick Start	Parker Reference 192-300306Nx
5	Application Note GVI I/O Control Mode	Parker EMPD Reference 192-300304Nx
6	Application Note Integrate GVI with IQAN	Parker EMPD Reference 192-300305Nx
7	GVI CAN Message Database	Parker EMPD Reference 192-300301Nx
8	GVI Flux Weakening	Application Note – Contact EMPD Support
9	GVI LV encoder offset	Application Note – Contact EMPD Support
10	GVI Motor Data	Application Note – Contact EMPD Support
11	GVI Bus DC Generator	Application Note – Contact EMPD Support
12	GVI/GVM Rotation direction	Application Note – Contact EMPD Support
13	GVI/GVM Tool Motor Setting	Application Note – Contact EMPD Support

Table 1 References

1.2 Configuring the GVI for the application

As shipped, the GVI is pre-configured with a set of default parameters appropriate for the GVI. A necessary step in the GVI start-up procedure is to update those default GVI parameters that are adjustable by the user to the correct values for the selected motor and application. The most common user-adjustable parameters are specified in this document.

The vehicle master controller or the Parker GVI Configuration Tool downloads parameter values that are dependent on the vehicle configuration chosen in production and/or in the field (for example performance modes). In order to retain a configuration, the parameters have to be stored in the GVI’s non-volatile memory (EEPROM). Thereafter the configuration is not changed until another configuration is downloaded and stored in the same manner. Consult the vehicle manufacturer’s maintenance information for specific instructions for performing this task.

2 Commissioning of GVI

As shipped, the GVI is pre-configured with a set of default parameters appropriate for the GVI. A necessary step in the GVI start-up procedure is to update those default GVI parameters that are adjustable by the user to the correct values for the selected motor and application. This is normally done by the Parker GVI Configuration Tool or by the Vehicle Master Controller. The most common user-adjustable parameters are specified in this document, for a comprehensive list is see **OD** (see 1.1.6). In order to change the values of the parameters, it is necessary to connect it to the CAN network according to the appropriate hardware manual as listed in chapter 1.1.6. The GVI uses the CANopen SDO protocol for configuration as shown in Figure 1 (Details in chapter 4.1). Some aspects are also configured by digital inputs (see chapter 3.2).

In order to retain a configuration, the parameters have to be stored in the GVI’s non-volatile memory (EEPROM). Thereafter the configuration is not changed until another configuration is downloaded and stored in the same manner.

WARNING

Testing vehicle parameters – risk of personnel injury/damage to equipment

All GVI settings and functionality must be verified and validated by the OEM.

The complete range of parameter values that are updated by the machine/vehicle controller (or any other device) must also be verified and validated before use in the field by the end user.

During the process of establishing parameter values it is of great importance to take proper safety precautions when testing, since incorrect parameter values may jeopardize the operation of the machine/vehicle’s safety critical functions.

It is the OEMs responsibility to ensure that the machine/vehicle is configured and set up to conform to applicable safety regulations.

The VCU may need to detect a situation where a GVI´s configuration (set of parameter values) is no longer correct for the application. One way of detecting such an issue could be by reading the checksum for all parameters stored in the GVI´s EEPROM (related parameter EepromPlasmaSegmentCheckSum, 0x2009:5). The typical reaction to an invalid configuration is to prohibit vehicle operation until the GVI has been correctly configured.

On the CAN network the vehicle master controller functions as master and GVI and other nodes such as display, joystick etc. as slave.

For control during operation, it is possible to select between CANopen PDO protocol, J1939 protocol or I/O control (only LV inverter), details in chapter 4.3.

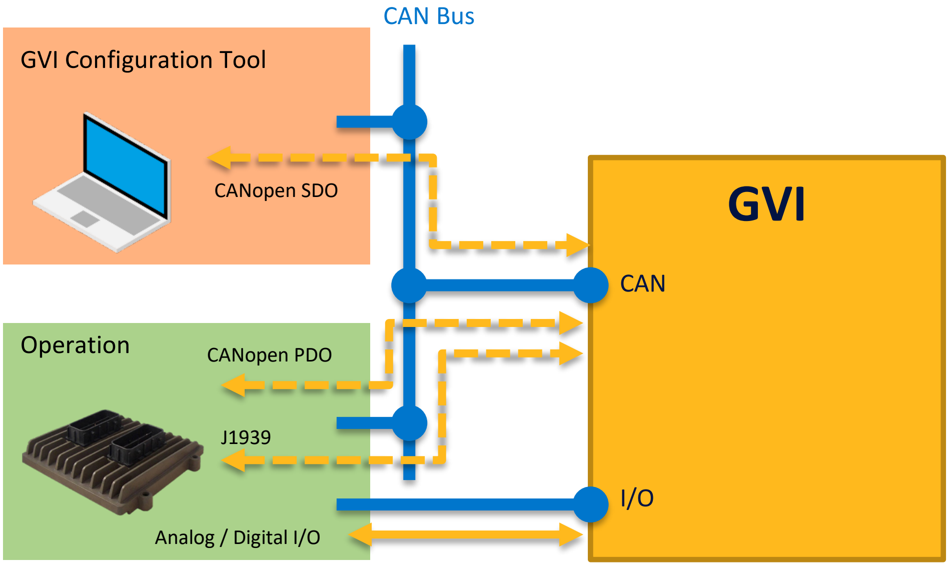
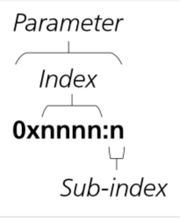


Figure 1 GVI Communication Interface for configuration and operation

3 Device Configuration

3.1 Parameter syntax

All parameters in the Object Dictionary have a name, an index and a sub-index:



3.2 Hardware ID

In order to differentiate multiple GVI on one CAN bus, they must have different HW-IDs. One of 4 HW-IDs may be selected by applying a jumper to the digital inputs (HV drives) or connect the inputs to BAT+ (LV drives) of the device as described in Table 2.

Other ID numbers (From 1 to 31) can be allocated with Parker GVI software tool. A restart of GVI must be performed.

Hardware ID	Digital Input Wiring	CANopen Node ID	J1939 Source Address
0	None	6h	200d
1	ID Pin 1 (18) to B+	7h	201d
2	ID Pin 2 (30) to B+	8h	202d
3	ID Pin 1 (18) and 2 (30) to B+	9h	203d

Table 2 GVI Low Voltage CAN ID Connections

Hardware ID	Digital Input Wiring	CANopen Node ID	J1939 Source Address
0	None	6h	200d
1	ID Pin 1 (28) to GND (31)	7h	201d
2	ID Pin 2 (15) to GND (12)	8h	202d
3	ID Pin 1 (28) and 2 (15) to GND	9h	203d

Table 2 GVI High Voltage CAN ID Connections

3.3 Internal Parameter Handling

3.3.1 General

Parameters are stored in EEPROM memory, which provides non-volatile storage and the capability to change parameter values. Figure 2 shows the organization of GVI memory and the paths over which parameters flow.

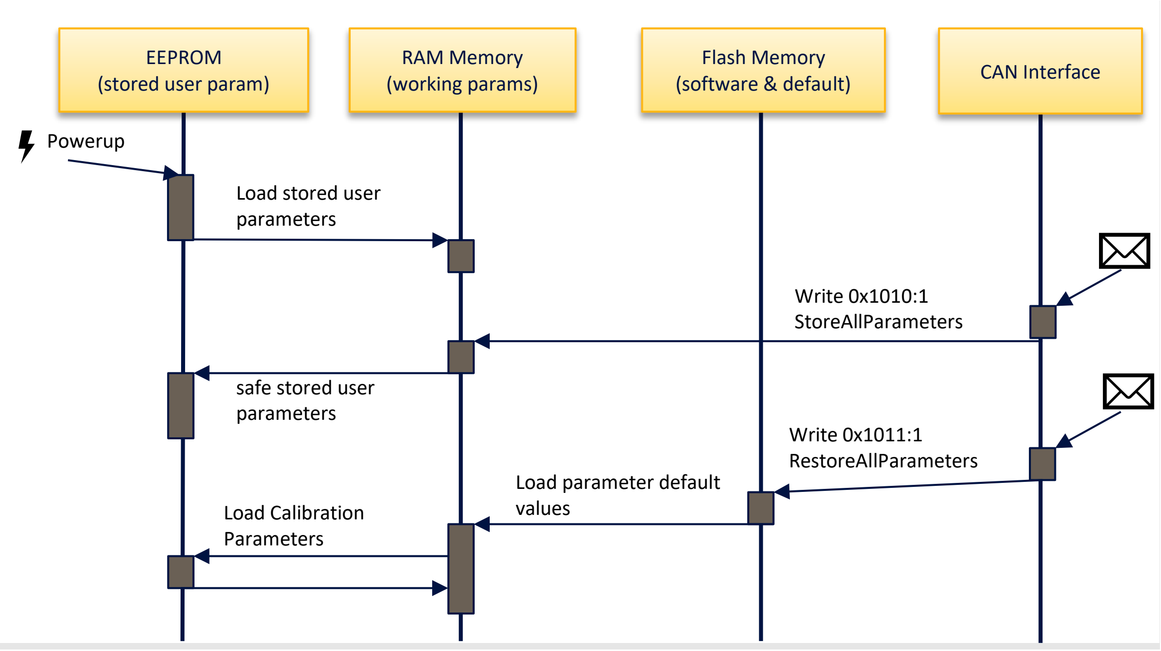


Figure 2 Parameter Access and Flow

3.3.2 Writing parameters

It should be observed that some objects are password protected, i.e. access/write to RAM is only possible if the correct password has been written to Password Object (4200h) prior to the attempted access. The relevant passwords are available on request from Parker.

WARNING

Wrong Parameter Values - risk of unpredictable equipment behavior

Where applicable the **OD** specifies permissible Min and Max values, however there are no firmware checks that a parameter has been set within the range. It should also be observed that the default value of an entry may not be correctly listed for all applications/configurations. The Unit/Scale gives either the unit of the value and/or the scaling used where applicable.

3.3.3 Parametrization Complete

A GVI that has not been correctly parameterized will have the object 0x2029:1 ParametersIsLoaded set 0 (FALSE) by default during production. This setting indicates that parameters have not been loaded. Object 0x2029:1 ParametersIsLoaded should be changed to 1 (TRUE) when parameters for the application have been loaded so that the inverter can be enabled.

CAUTION

If the firmware is updated or the Hardware ID is changed then the parameter (0x2029:1) is again set to 0 to prevent accidental operation.

3.3.4 Parameter verification at startup

On power up (power on the KEY input for LV drives or Logic Supply + DigInput Unit Enable for HV) the drive software copies parameter values from EEPROM memory into a segment of RAM memory reserved for parameters. The operating software then performs several checks to verify the validity of the parameter set in RAM. The parameter set is considered invalid whenever any of the following discrepancies are detected.

- GVI HW ID (read from HW ID pins, see chapter 3.2) ≠ GVI ID (read from EEPROM)
- Application SW Build Time (read from Flash Memory) ≠ Application SW Build Time (read from EEPROM)
- Parameter checksum error

When an invalid set of parameters is detected, the GVI operating software performs the following actions:

- Copies default parameters from flash memory into RAM
- Updates GVI ID from reading of HW ID pins
- Updates Application SW Build Time stored with parameters
- Computes checksum
- Saves parameters to EEPROM
- Sets Event 1402, Default parameters, Sets the warning bit 12

The vehicle master controller may then permit default parameters to be used or download different parameters to the GVI.

During operation, all parameter read and write operations access the "working parameters" in RAM memory. The content of RAM memory is lost whenever power is removed from the Enable input. The vehicle controller must take specific actions to permanently save parameter value changes.

3.3.5 Saving parameters

In order to update the permanent set of parameter values in EEPROM memory, the vehicle master controller must initiate EEPROM save by writing the value 65766173h to CANopen object Store All Parameters (0x1010:01). Saving parameters must be performed with the vehicle at standstill. Reading of (0x1010:01) returns 1 if no storing is in progress (IDLE STATE). Read returns 0 if any storing is in progress (BUSY STATE).

3.3.6 Duplicating a configuration

The fact that the operating characteristics of a GVI are determined by a set of numbers (parameters) combined with the digital nature of the drive assures that two GVI identically configured behave identically in a given application.

In a production environment any of the following methods may be employed to configure GVI:

- The user may download the configuration to the drive as a step in their vehicle assembly process.
  This requires a PC with CAN Bus interface and the clone functionality of the “GVI Config Tool” to perform the procedure.
- The user may download the configuration to the drive from the vehicle master controller.

3.3.7 Reverting to default parameters

The user may revert to default parameters (stored in flash memory along with the GVI software) using a procedure similar to saving parameters, i.e. by writing the value 64616F6Ch to CANopen object Restore All Default Parameters (0x1011:01). The power stage needs to be disabled when restoring. This procedure does not store the defaulted parameters to EEPROM, see section 3.3.5 for how to store parameters to EEPROM.

3.4 System overview

⚠

WARNING

Equipment testing - risk of unpredictable equipment behavior

- All GVI parameter values established must be verified and validated prior to that the parameter values are used in the field by the end user
- This must be done in order to establish that all safety critical functions in the vehicle, for example braking, are working properly
- The complete range of parameter values that are updated by the vehicle master controller (or any other device) must be verified and validated prior to those values being used in the field by the end user.
- During the process when the parameter values are established it is of utmost importance to take proper precautions when testing since wrong parameter values may jeopardize the operation of the vehicle’s safety critical functions

The functionality of the GVI comes largely from its operating software, which controls nearly every aspect of its operation. Parameters and variables passed to the operating software utilize the internal modules and define the operating envelope of the motor/drive combination.

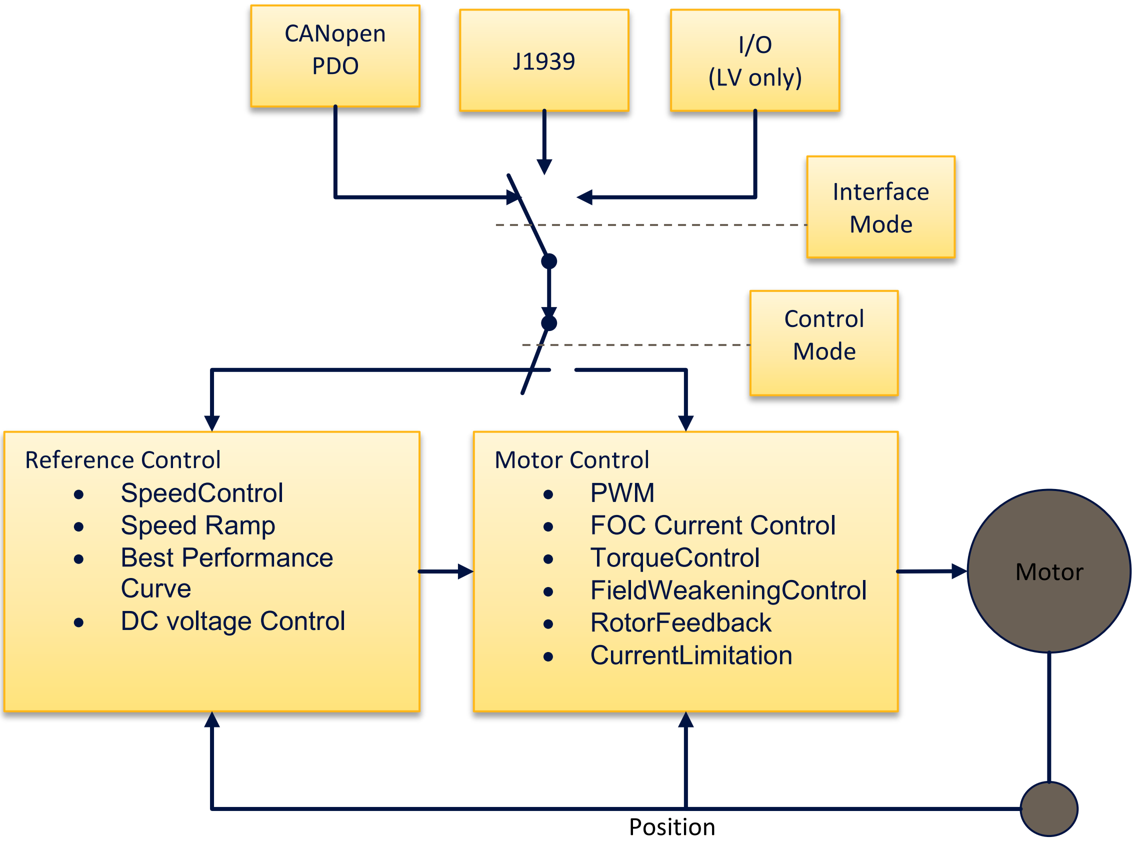


Figure 3 Overview of the GVI control structure

4 Communication

4.1 Interface communication device

To communicate via CANopen or J1939 with Parker GVI Configuration Tool and GVI, two dongles are compatible:

IXXAT® => USB-to-CAN v²

KVASER® => Leaf Light v2

IXXAT®=> USB-to-CAN v²

KVASER® => Leaf Light v2

4.2 Introduction to CANopen

CANopen is the main CAN protocol of the GVI. Commands, motion variables and parameters exchanged between the VCU and a GVI are embedded in the CANopen communication objects. In the CANopen Communications protocol implemented in the GVI, the following five types of objects (messages) are utilized:

Object (message) type	Purpose
Service data object (SDO)	Exchange of non-real time data and control, Device Configuration
Process data object (PDO)	Exchange of real time data and control, Device control
Network management (NMT)	Transmitted by the master to control network boot up, initialization and state transitions. Network management messages may be directed to individual devices or broadcast to all network nodes
Emergency (EMCY)	Transmitted by slaves to signal internal error conditions to the master

Table 2 Message types used in the GVI

Each GVI requires a periodic Receive PDO containing motion and related control variables when in CANopen interface mode (see chapter 4.3). The GVI can be configured to use up to four Transmit/Receive PDO´s.

Messages intended for specific devices on the CAN network contain a COB-ID (Communications Object - Identifier) field which identifies the target node (device) for the message. Each GVI on the CAN bus determines its CANopen ID via the wiring configuration of two pins on the interface mating connector. Each GVI on the CAN-bus must have a unique CANopen ID. See more details about how to setup CANopen ID using the connector interface in the product manual, see chapter 1.1.6. or 3.2

4.3 Baud Rate

The default Baud Rate of the CAN interface is 250 kbit/s. It can be changed via the Parameter *Bitrate* Object 0x5F04:01. You must then restart the drive and scan with Parker GVI Configuration Tool.

4.4 Interface Mode for Device Control

The drive can be controlled in operational state using either CANopen protocol, J1939 protocol or digital/analog inputs (LV drives only). CANopen protocol provides the most flexible interface of the three.

Selecting interface mode is done by changing the Object RequestedInterface 0x2029:02 according to Table 4.

Value	Description
0	(Default) LV: Hardware Controlled, HV: CANopen Controlled
1	CANopen Controlled
2	J1939 Controlled
3	IO Controlled (Not available for HV)
4	Invalid

Table 3 Interface Mode selection for LV and HV GVI in Object RequestedInterface 0x2029:2 and ActualInterface 0x2029:3

For Low Voltage inverters (LV) the interface mode can also be changed by jumpers on digital inputs, see Table 5. To be I/O switchable RequestedInterface 0x2029 must be set to zero (=*Hardware Controlled)*, which is the Default setting.

Digital Input 3 (pin # 23)	Digital Input 4 (pin #11)	Interface Mode
0	0	CAN Open
1	0	CAN Open
0	1	J1939
1	1	I/O

Table 4 Interface Mode selection by digital inputs for LV GVI only

To apply the switch between Interface Modes, a restart of GVI is necessary.

Details on the different Interface Modes are in following chapters.

4.5 CAN Port supported protocols

NOTE

High voltage GVI, frames G and H, are equipped with two CAN ports. The functionality of these CAN ports differs with firmware version. Ensure that the GVI is wired to correctly support the required protocol.

GVI frame size H and H (HV) have two CAN interface ports, CAN_Main and CAN_Aux. In previous versions of the hardware installation manual these were referred to as CAN_J1939 and CAN_CANopen.

The functionality of these two ports is dependent on the firmware version. GVI Frames C, D and E have only one port which supports all protocols, as described in section 4.4 above.

Protocol	Firmware C03x04 and earlier	Firmware C04x01 and later	Used For
J1939	CAN_Main	CAN_Main	J1939 Communication
CANOpen SDO1	CAN_Aux	CAN_Main	CANOpen Communication
CANOpen SDO2	CAN_Aux	CAN_Aux	GVI Configuration Tool
CANOpen PDO	CAN_Aux	CAN_Main	CANOpen Communication
NMT	CAN_Aux	CAN_Main	Configuration Tool NMT
CANTalk	CAN_Aux	CAN_Aux	Firmware Installation

It is important that the GVI is integrated with the vehicle CAN using the correct port for the protocol required. With firmware version C04x01 or later, all vehicle protocols are supported on CAN_Main, with CAN_Aux being reserved for programming and firmware updates using the GVI configuration software tool. Prior to version C04x01, CAN_Main (CAN_J1939) was reserved exclusively for J1939 vehicle communications, and all other functions were supported only on CAN_Aux (CAN_CANopen).

4.6 CANopen Interface Mode

The GVI is controlled via the CAN bus by the vehicle master controller. By passing a PDO regularly to the GVI with a constant rate, the speed of the motor and the status of the GVI can be controlled.

The default configuration of GVI has four receive and four transmit PDOs with a default mapping described in CAN Message Database in chapter 1.1.6.

In its basic configuration the Command Word and the Command Speed are sent to the GVI via this Receive PDO. The GVI then returns a Transmit PDO regularly that at least includes the Status Word and the Actual Speed.

The Command Word is a bit field and by clearing/setting bits, the operation of the GVI is controlled. The Status Word is also a bit field and it reflects the actual status of the GVI.

The sequence to initiate the CANopen communication and to enable the operation of the GVI is described in chapter 10.6, where a detailed example of CAN communication during operation between VCU and GVI in a system is shown.

The protocol is based on CANopen with 11 bit identifier, based on CiA Draft Standard 301, version 4.01.

4.6.1 Error Handling

When an event classified as an error occurs, a CANopen standard error code is sent out in an EMCY message. The emergency message and its contents are shown in Table 6. The emergency message follows reference CiA Draft Standard 301, Version4.0. Parts of the manufacturer specific Error Field (i.e. bytes 4-7) are used for a more specified error bit field Extended Errors.

Byte#	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Content	Emergency error code		Error register	Not used	EventID		System Reaction	Not used

Table 5 Emergency message

4.6.2 Emergency Error Code

Each detected event is configured according to the table given at the end of the **OD**. When an event is detected several actions might be taken.

A group of events can be connected to a linear current reduction when active. Linear reductions are connected to events set when Temperature, DC Bus Voltage or Motor Speed is too high or too low. The actual current reduction can be read using AbilityAccelerationCurrent, 0x2095:09 and AbilityBrakeCurrent, 0x2095:10. If the reason for the linear reduction event no longer exists the event will be cleared and full current will be allowed. See more about current limitations in chapter 6.3.3.

Any event that does result in a trip or a current reduction will only be cleared when CommandEnable is not set. The main reason for this is to prevent unintended current output if an event has healed. All other events will be cleared when the reason for setting the event no longer exist. Events with a linear current reduction will be cleared when the reason for setting the event no longer exist.

The currently active events can be read using ActiveEvents, 0x3011:01.

4.6.3 Drive Reset command

Reset a GVI via CANopen or J1939 is possible, following this procedure.

- Via CANopen command

  First => CommandAll.Enable = 0

  Then => UNIT_ENABLE = 0 **OR** NMT reset

- Via J1939

  First => CommandAll.Enable = 0

  Then => UNIT_ENABLE = 0

4.6.4 Default PDO definition

The default PDO definition is given in the document *GVI CAN Message Database*, see chapter 1.1.6. The mapping can be changed according to application needs. More about how to map the PDOs and setting up the Tx and Rx members, can be read in chapter 4.4.4.

NOTE

Starting a motor requires a certain sequence of setting bits on CommandAll Object. Details in chapter 10.6.

4.6.5 Change default PDO communication

4.6.5.1 General

The following instruction shows how to make changes to the PDO communication setup dynamically.

1. set drive in NMT state PRE_OPERATIONAL.
2. The PDO to be edited is then disabled by setting the PDO_NOT_VALID bit in the Id field. i.e. write 0x80000000 to the CobId for the PDO.
   When a PDO is disabled it is possible to edit the contents of the PDO map. The number of parameters in the map can also be changed by writing the required number of members in the LastSubIndex field.
3. After finished editing, the PDO is enabled again by writing 0 (zero) to the CobId (subindex 1) for this particular PDO.

Example of used syntax for the PDO map:

- 3010h:1 (32-bit variable) gives value=30100120h where index=3010h subindex=1h datatype=20h (→32-bit variable).
- 2001h:2 (16-bit variable) gives value=20010210h where index=2001h subindex=2h datatype=10h (→16-bit variable).
- 2071h:8 (8-bit variable) gives value=20710808h where index=2071h subindex=8h datatype=08h (→8-bit variable)

4.6.5.2 Example: Changing a Member in an existing PDO1Rx

Example

The aim is to change map member 2 in PDO1rx from [0x2000:02] (S16) *CommandSpeed* to [0x2000:04] (S16) *CommandTorque*.

First set NMT state to Pre-Operational.

The CobInPDO1rx is located at [0x1400:01] *ReceivePDO1ComParameters*. Write value 0x80000000 to this parameter. This will disable the PDO1rx.

Then go to 0x1600 *ReceivePDO1MapParameters*. Member 2 in this map is located at [0x1600:02] The value that can be read is 20000210h which corresponds to [0x2000:02] (16-bit variable). For setting the new PDO map for member 2, write 20000410h to [0x1600:02]. The new setting is stored to eeprom automatically and ready to be used.
Now the PDO1rx needs to be enabled. This is achieved by writing 0 (zero) to parameter 0x1400:01.

4.6.5.3 Example: Adding a Member in PDO2Tx

The aim is to add 2 members [0x2077:01] *Iq* and [0x2078:02] *IqRef* to the PDO map PDO2tx. As an assumption there are already 2 members in the PDO2Tx map. Hence the added members shall be implemented at [0x1A01:3 and 4]

Set the drive into state Pre-Operational.
The CobIdPDO2Tx is located at [0x1801:1], write 0x80000000 to this parameter to disable PDO2tx.
At 0x1A01 *(TransmitPDO2MapParameters)* subindex 0, the current number of members in the map can be read. Increase this number by 2, i.e change the value for 0x1A01:0 from 2 to 4.
The mapping is then set up by writing 20770110h to [0x1A01:3] and 20780210h to [0x1A01:4].

The PDO2tx then needs to be enabled again. This is achieved by writing 0 (zero) to parameter [0x1801:1].

4.7 J1939 Interface Mode

This chapter contains the specification of the GVI J1939 protocol. The detailed definition of the J1939 messages are given in CAN Message Database in chapter 1.1.6.

In order to decode the J1939 messages on the CAN bus please use ParkerGVI_J1939.dbc file.

Address Claiming is not supported.

Unlike the CANopen PDO messages, the J1939 message mapping to OD parameters cannot be changed.

4.7.1 J1939 Parameters

All parameters relating to J1939 communication can be found under indexes 0x2700, 0x2701, 0x2702 & 0x2704. To modify and read these parameters communication with the drive using SDO is needed, it is not possible to modify these parameters using J1939 communication.

Under 0x2700 parameters related to timeout for receive messages are located. The source address for both the GVI and the external controller can be changed here. The ComState (0x2700:01) can also be monitored to see if the GVI is in a timeout state etc.

Under 0x2704 the transmit periods for the GVI transmitted values are located. They can be changed to lower the busload etc. If this parameter is set to 0 the message will not be transmitted.

4.7.2 Timeout

Timeout handling of the Command Messages is described in Figure 4. At start-up, the J1939 initializes a start-up timer and starts to look for received messages. If no message is received before a predefined start-up time located in OD index ComStartupTimeoutMs (0x2700:02), the unit will enter the timeout state and send an error event until a command messages is received.

If a command1 messages is received and it is a valid signal, the unit will enter CANCOM_OK. A command1 message needs to be received periodically at a predefined time located in OD index CommandMessageTimeoutMs (0x2700:03). If this is not the case, the unit will enter the timeout state and send an error event until a command1 messages is received.

If the command1 message is not valid, the unit will enter the timeout state and set an error event until a valid command1 messages is received. The definition for a valid message is explained in Section 4.5.3

Same applies for command2 message.

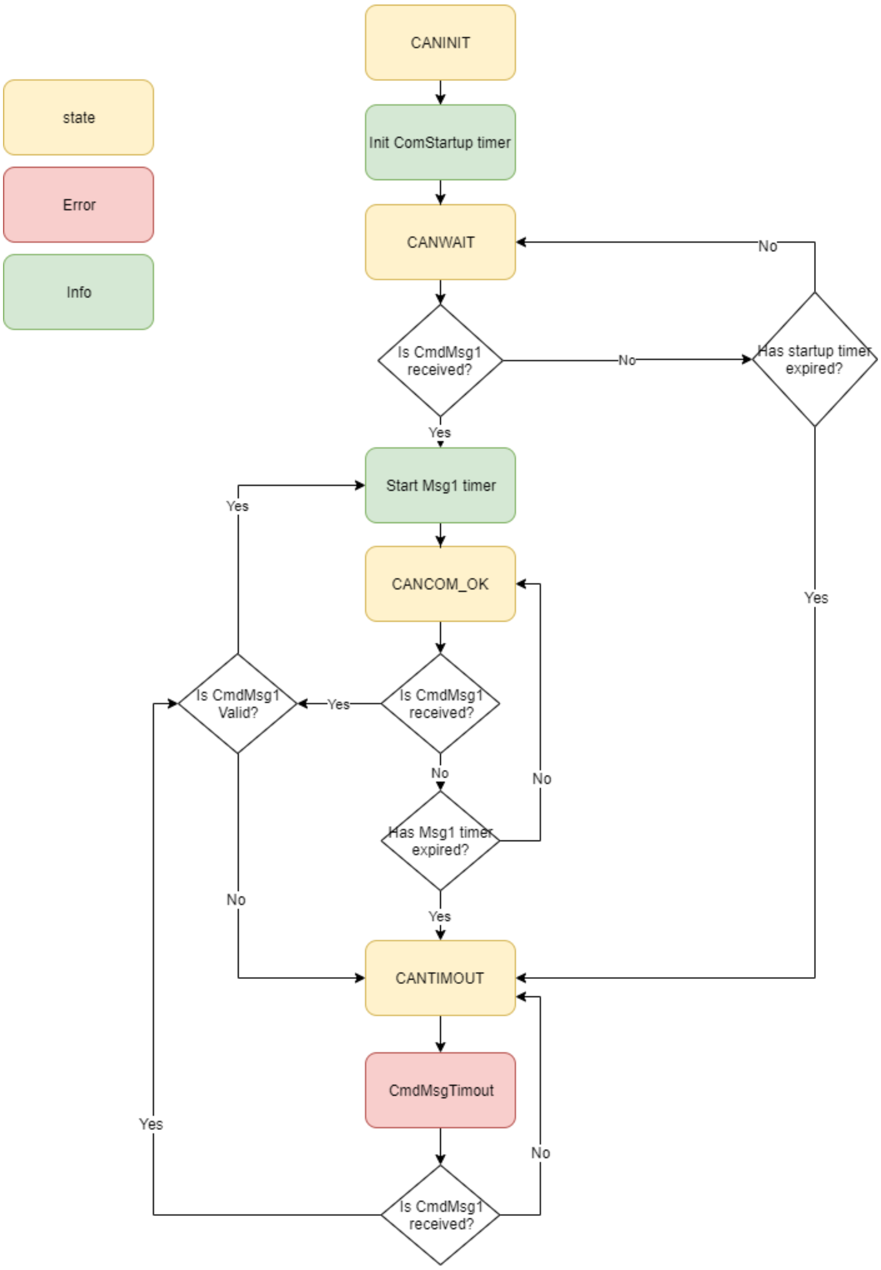


Figure 4. Flowchart J1939 protocol.

4.7.3 Valid Command Message

A command messages is valid if:

1. A signal in the messages is inside of the valid range according SAEJ1939.
2. All critical signals are defined in the messages.
3. CommandWord is an exception since all 16 bits in the bit field are used.

The valid signal range for SAEJ1939 is defined in Table 7.

Type	Range
UINT8	0xFA
UINT16	0xFAFF

Table 6: Valid signal range according to SAEJ1939

All command messages should be sent at 10ms period time as default. The default timeout is 500ms and for the limit message the default timeout is 5000ms.

4.7.4 Default Source Addresses

Source address (SA) for the GVI is given in chapter 3.2. The unit listens to messages sent from an external controller (denoted EC) at SA = 0xC7 by default. It is possible to change the source address for both the EC and the inverter with an OD parameter. The address for the inverter is 0x2700:11 and the EC 0x2700:12.

4.7.5 Error Handling

In order to detect an error, the *status message* Bit 6 must be monitored (see chapter 4.5.7.1). DM1 functionality is not supported. *Event Message* provides the information about which event caused the error (see chapter 4.5.7.5).

4.7.6 Receive messages

Listed below are all the receive messages that the GVI is expecting to receive. These messages are vital for the GVI to be able to operate and if no message is received the inverter will be set to an error state until the correct command message is received.

For details please see *GVI CAN Message Database*, chapter 1.1.6.

4.7.6.1 Command1 Message

The command1 message is the main communication message to communicate with the GVI over J1939. The message contents are the SPNs

- CommandAll
- CommandSpeed
- CommandAccelerationChange
- CommandDecelerationChange.

NOTE

Starting a motor requires a certain sequence of setting bits on CommandAll Object. Details see in chapter 10.6.

The CommandAll corresponds to the same CommandAll bitfield that is available in the CommandAll SDO parameter 0x2000:01. For this signal there is no out of range check since all 16 bits are used for control actions.

The command speed is the reference speed when the GVI is in speed control mode. If this signal is above 0xFAFF (64255) the signal is considered invalid and the cmd1msg timeout event is set. This is automatically cleared if the signal is again within the valid range. The command speed transmitted via J1939 will be the set value for 0x2000:02 in the **OD**. If this signal is above/below MaxCommandSpeed 0x2020:11 the value will be limited to the max command speed internally.

CommandAccelerationChange and CommandDecelerationChange are the acceleration and deceleration ramps when the GVI is in SPEED_PDO_RAMP_MODE. The valid range for these signals are 0xFA (250) and if they are above this value they are considered invalid and the GVI is set into the same timeout event as for the command speed. Values transmitted over J1939 will be set to 0x2000:05 & 0x2000:06 in the **OD**.

The last 16 bits in the J1939 message are reserved and unused.

4.7.6.2 Command2 message

The command2 message contains control signals related to all different control modes except speed control mode. The signals included in the message are

- CommandTorque
- CommandAcCurrent
- CommandVoltage
- RequestedControlMode

The last 8 bits of the command 2 message are unused.

Command torque is the commanded torque when the GVI is in Torque mode. The command torque can be set regardless of which control mode the GVI is in but the reference is only used when the GVI is in torque mode. If the command torque signal is higher than 0xFAFF, an event will be set “cmdmsg2Timout”. This event will be cleared once the signal is within the valid range.

Command AC current is the current reference when the GVI is in AC current mode. The same event as for the torque mode will be set if the signal is out of range. The event is cleared once the signal is within the valid range.

CommandVoltage is the reference value when the GVI is in Voltage mode. If this signal is out of the valid range 0xFAFF an event will be set. The event is cleared once the signal is within valid range.

Requested control mode sets the control mode to the requested control mode. If an invalid control mode is set the last valid control mode will be kept until a new valid control mode is selected.

4.7.6.3 Limit message

The limit message can be used to limit the available torque and available current in both acceleration and brake quadrants and also positive and negative DC current. Note that the measurement of DC current is not possible, the DC current is estimated from the measured AC current and the DC power is calculated using the measured AC current. For these limits to be active the 2020:10:Bit8 and Bit 9 have to be set to 1 (one) in order for the limits to have effect.

The signals included in the message are

- AccTorqueLimit
- BrakeTorqueLimit
- PosDcCurrentLimit
- NegDcCurrentLimit

AccTorqueLimit sets the maximum allowed acceleration torque. If this signal is out of range the latest received valid limit is kept and no event is set.

BrakeTorqueLimit set the maximum allowed brake torque. If this signal is out of range the latest valid signal is kept.

PosDcCurrentLimit limits the max DC current draw into the inverter. If invalid signal is received the limit is set to the maximum possible limit.

NegDcCurrentLimit limits the max DC current that can be fed from the inverter into the DC power source. If invalid signal is received the limit is set to the maximum possible limit.

4.7.7 Transmit messages

The GVI transmits 5 messages, these messages contains vital data from the GVI.

For details please see *GVI CAN Message Database*, chapter 1.1.6.

4.7.7.1 Status message

Status message contains following SPNs

- StatusAll
- CanSignalRotorSpeedInRpm
- CanSignalRmsMotorCurrent
- CanSignalFilteredVoltage

StatusAll is the equivalent of the 0x2001:01 (StatusAll) bit field in the **OD**. This message bits are all used for providing status information regarding the controller.

CanSignalRotorSpeedInRpm is the measured speed of the motor. If speed sensor is disconnected the raw value on the bus will be set to 0.

CanSignalRmsMotorCurrent transmits the RMS motor current from the motor.

CanSignalFilteredVoltage is the DC bus voltage.

In Table 8 the resulting J1939 identifier for the Status message can be seen. The default transmission rate is 10ms.

HW-ID	Priority	PGN	Source Address Default (Controller)	J1939 Identifier
0	6	0xFF3C	0xC8	0x18FF3CC8
1	6	0xFF3C	0xC9	0x18FF3CC9
2	6	0xFF3C	0xCA	0x18FF3CCA
3	6	0xFF3C	0xCB	0x18FF3CCB

Table 7: Resulting J1939 identifier for the Status message.

4.7.7.2 Diagnostics1 message

- CanSignalActTorque
- DCBusCurrent
- DigitalInStatus
- AbilityAccelerationCurrent

4.7.7.3 Diagnostics2 message

- ActualLimitationType
- RegulatorStatus
- SensorAngle
- Iq
- Id

4.7.7.4 Diagnostics3 message

- CanSignalMotorTemp
- CanSignalInverterTemp
- ActualControlMode
- ActiveEvents

4.7.7.5 Event message

This message contains following signals

- Event 1
- Event 2
- Event 3
- Event 4

The event message will transmit the ID of the active events in the GVI. The **OD** file can then be used to obtain information regarding the active events. Max number of active events that can be transmitted is 4. If more than four events are active the most recent 4 will be transmitted.

4.8 IO Interface Mode

GVI with frame size C, D and E (LV) are capable of being controlled by digital and analog input and output signals. For details see the *Application Note GVI I/O Control Mode* referenced in chapter 1.1.6.

5 Motor Control

NOTE

All following GVI/MOTOR tuning sequences assume unknown motor data (third party motor supplier).

For GVM (full range) and NX8HxM refer to PARKER motor specifications and all APPLICATION NOTES mentioned in this manual. You will find the full process for GVI / MOTOR parameter setting and how to use different tools to fully create a correct GVI configuration file. You can also contact local your local Parker Mobile Application engineer or your local distributor for any assistance.

The GVI is used to drive permanent magnet motor (PMAC) ONLY. For example, Parker’s GVM (Global Vehicle Motor) or NX8xHM (LV only). For this purpose, the GVI uses the field-oriented control method, see Figure 5. In this method the physical quantities such as stator voltages and currents are transformed into orthogonal vectors oriented to the rotor flux. This allows have two independent components to be controlled, the d-axis for flux control and q-axis for torque control. To achieve that there are two PI current controllers.

Every GVI control mode GVI (Torque, Speed, Current, DC Voltage control mode), requires a properly configured and tuned motor control loop.

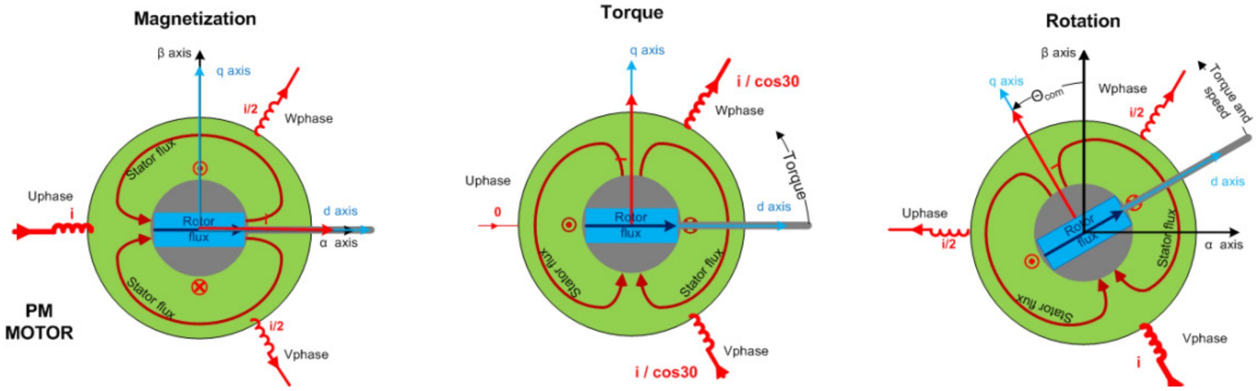


Figure 5 Field Oriented Control for PMAC motors

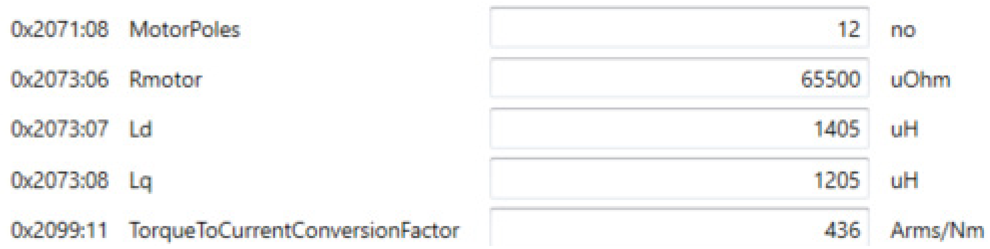
5.1 Motor Winding Parameters

In order to setup the drive it is required to configure the following Motor parameters. See application notes referenced to in chapter 1.1.6 (GVI Motor Data)

Prior to the configuration, the following motor data shall preferably be obtained from the motor supplier/manufacturer:

- Number of motor poles
- Stator resistance (phase to neutral)
- Ld, inductance in d-direction (phase to neutral)

- Lq, inductance in q-direction (phase to neutral)
- TorqueToCurrentConversionFactor (Arms/Nm) = 1024/Motor Kt



0x2071:08	MotorPoles	12	no
0x2073:06	Rmotor	65500	uOhm
0x2073:07	Ld	1405	uH
0x2073:08	Lq	1205	uH
0x2099:11	TorqueToCurrentConversionFactor	436	Arms/Nm

Figure 6 Motor Winding Parameters

If the motor parameters are not known, please refer to chapter 10.4 in the appendix.

Rmotor value needs to be at a temperature of 20°C. The motor resistance is calculated for the actual motor temperature with a fixed temperature coefficient of $0.0039\frac{1}{°C}$. The resistance can be read at RmotorTempComp 0x2073.15.

5.2 Current Controller Tuning

A PI-controller is used for d- and q-current using reference voltage as input. The proportional part correlates to inductance and the integral part correlates to resistance as well as inductance. Nominal rotor flux and inductances are used to de-couple the two controllers.

This document describes the procedure used to verify that the current step response with GVI is OK and the actions that must be taken if it is not OK. A responsive current controller is required since it must be faster than the input term in order to prevent instability. Other regulators (e.g. speed, torque and voltage) are higher up in the control strategy and used as input to the current controller.

WARNING

- Tuning of the current controller can lead to uncontrolled movement of the motor if the control parameters are wrong!
- Never use FeedbackMode=0 (Physical Sensor) if the SensorAngleOffset is not set properly, otherwise the motor will accelerate with control!

5.2.1 Id PI-Controller

Following formulas give stable current controllers for most motors, when Ld, Lq and R of the motor is known.

$$\mathbf{Pgain}_{\mathrm{D/Qpart}_{\mathrm{CurrentController}}} = 940,8 \cdot \frac{L_{d/q}\,[\mu H]}{U_{DC,nom}[V]}$$

$$\mathbf{Igain}_{\mathrm{D/Qpart}_{\mathrm{CurrentController}}} = 3,609 \cdot \frac{R\,[\mu\Omega]}{U_{DC,nom}[V]}$$

$U_{DC,nom}$ is the nominal GVI DC voltage. Do not use the battery voltage used in the application, but the nominal voltage of the device. This is part of the product code, for Example GVI-C024 for 24V. Ld, Lq and R are the phase to neutral motor's values.

If detailed motor data is not known (or to check if theoretical values are good enough) use the procedure below.

General suggested approaches to the current response tuning:

- Start with a low I-gain and gradually increase P-gain until ActualCurrents.Id reaches approximately 80% of ReferenceCurrents.IdRef without losing stability, then gradually increase I-gain until desired response is achieved.
- Set or reference a requirement on rise time and maximum overshoot. Tune P-gain to achieve the wanted rise time. Tune I-gain to achieve zero steady state error fast enough, but avoiding overshoot.

1. Write the following to the drive:
    - RequestedControlMode (0x2020:12) = 3 (AC current control mode)
    - FeedbackMode (0x2052:10) = 1
    - ExternalIdRefActive (0x2078:13) = 1
    - ExternalIqRefActive (0x2078:14) = 1
    - ExternalIqRef (0x2078:12) = 0
    - ExternalIdRef (0x2078:11) = 50% of maximum current
    - MaxIdRefChange (0x2072:12) = 0
    - MaxIqRefChange (0x2072:13) = 0
2. Setup the transient recorder (see chapter 1.1.6) with Id (0x2077:2) as first item.
3. Change trigger mode to Greater than and Trig value to e.g. 200.
4. Set Interval to 1, i.e. one sample equals 250 s.
5. Arm trigger.
6. Enable the drive.
7. Disable the drive.
8. Plot and zoom in on response.
9. If necessary adjust trigger value and delay as needed.
10. Repeat the step at least one time (first time the rotor may not be aligned with the assumed d-direction).

The response time is application dependent but can typically be between 0.5 to 2.5 ms (2 - 10 samples) and is defined as the time it takes between 10% to 90% of the final value. The overshoot shall be less than 20% and there shall be no oscillations (i.e. overshoot followed by undershoot with significant amplitude). See figures below for an example of a step response that is approved and responses that are not approved.

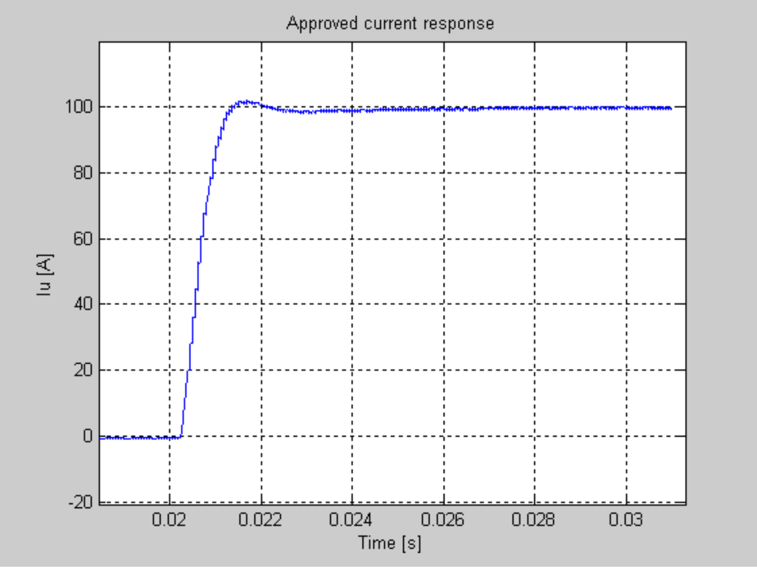


Figure 7 Current step response that is approved. The rise time is adequate and the overshoot is only a few percent.

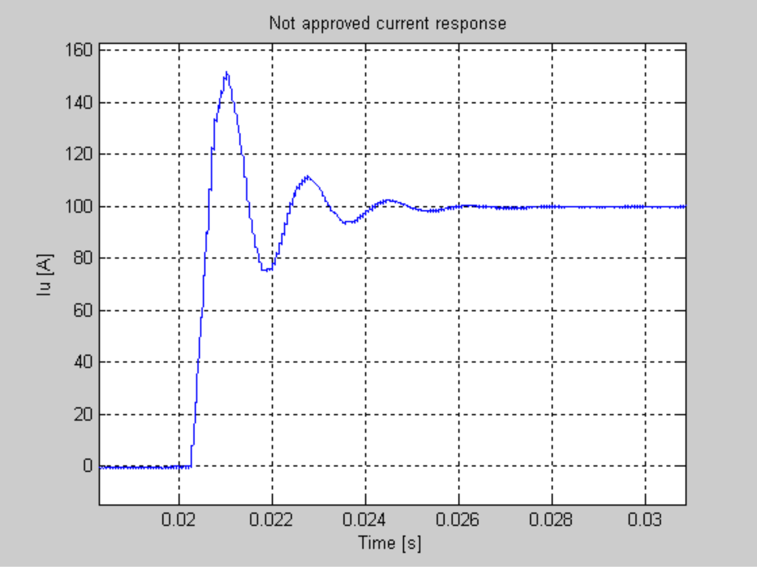


Figure 8 Current step response that is not approved. The rise time is too low, the overshoot is around 50% and is also followed by a significant undershoot.

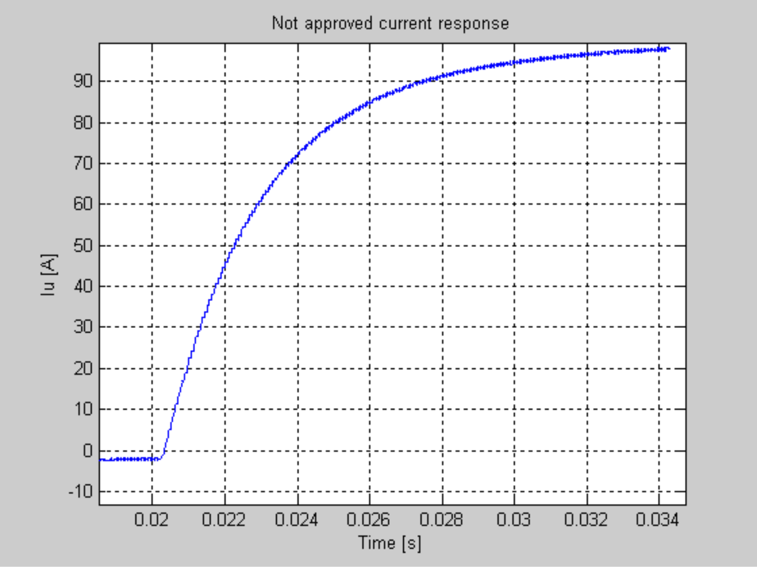


Figure 9 Current step response that is not approved. The rise time is too long.

If the initial part of the response is too slow, then redo the step response with PgainDpart_CurrentController (0x2072:05) increased.

If the response time is too short then redo the step response with PgainDpart_CurrentController (0x2072:05) decreased.

If the overshoot is too high (oscillations), then redo the step response with IgainDpart_CurrentController (0x2072:06) decreased.

5.2.2 Iq PI-Controller

This is usually not necessary, the gains found in d-direction can be used here directly, but tuning d- and q-current control individually can give improved current control in some PM motor applications.

The procedure is similar to the one in d-direction, but with q instead of d. Between each repetition of the step response, toggle FeedbackMode (0x2052:10) to 0 and then back to 1 in order to reset the sensor angle to the rotor's actual position (requires sensor to be correctly aligned).

5.3 Feedback Sensor

For FOC to work properly, the correct offset between feedback sensor and motor magnets must be known. While a slightly misaligned offset will lead to decreased efficiency a large error can lead to uncontrolled motor movement.

Feedback sensors of Parker GVM motors are aligned at production to specific values given in the table below

	SensorAngleOffset 0x2052:03
LV GVI	44° (GVM 12 poles)
LV GVI	53° (NX8xHM)
HV GVI	30° (12 poles) 22.5° (16 poles)

Table 8 Default Sensor Angle Offset values for GVM Motors

If the SensorAngleOffset is not known, or needs fine tuning, please refer to chapter 10.3.

In order to reduce noise in the stator angle from the sensor a Phase Locked Loop filter is implemented (0x2071:18&19). This filter doesn't have angle delay at constant speed but setting the filter too slow can introduce angle error at high accelerations.

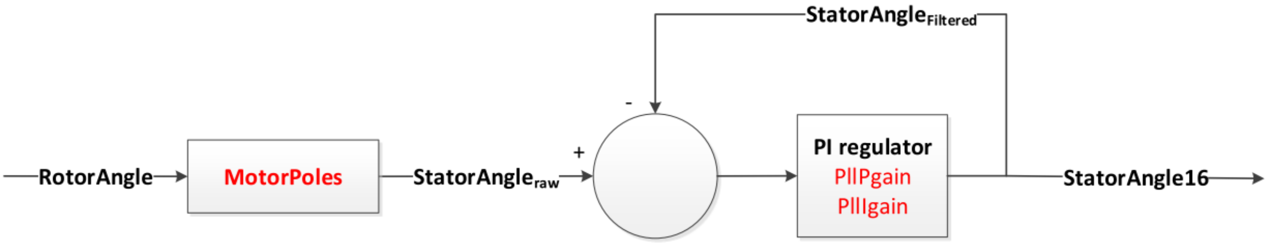


Figure 10 PLL filter for Stator Angle

5.4 Field weakening control

The induced voltage of a PMAC motor is proportional to the speed. At a certain speed, it reaches the available DC link voltage, so that no further speed increase can take place. To avoid this, the stator flux can be weakened. Refer to specific application note (see chapter 1.1.6) for settings.

In field weakening mode, the PI speed and current regulators are inactive, and field weakening control generates the reference voltages by controlling the VoltageAngle via a single PI-Controler with ActTorque and RefTorque as input.

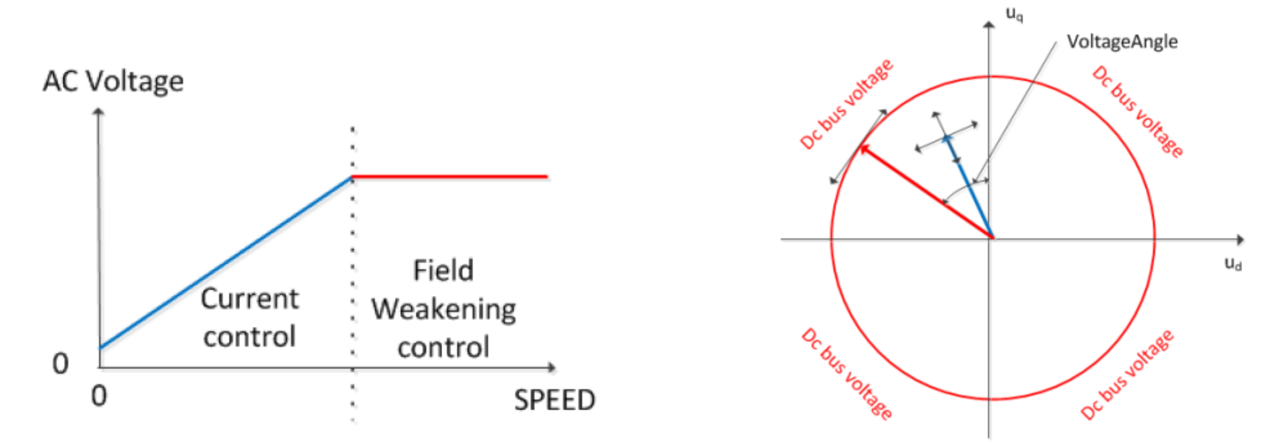


Figure 11 Field weakening control mode

The advantage of field weakening control via VoltageAngle is that no voltage reserve is required. The disadvantage is a lower bandwidth due to the reduced degree of freedom.

A critical point is where to enter or leave the field weakening control.

The used AC voltage is calculated based on regulator output UdRef and UqRef

UPeak = sqrt(UdRef^2 + UqRef^2)

UPeakNormalized= UPeak * NominalVoltage / DcVoltage

- Entering Field Weakening
    - UPeakNormalized >= UPeakLimit

- Leaving Field Weakening: There are three criteria's that can be selected

  1. IdFiltered > IdRef

  2. abs(IqFiltered) > abs(IqRef)

  3. VoltageAngle not at MaxVoltageAngle

Which of the criteria's to use is selected with the parameter TransitionRequired. The corresponding status is readable at TransitionStatus

Required NumberOfSamplesAboveLimit can be added to delay the leaving of field weakening.

IdLimitOffset and IqLimitOffset can be added to IdFiltered and/or IqFiltered to delay the leaving of field weakening.

5.4.1 Controller response tuning with dyno

This PI-controller needs to be tuned to be faster than the speed controller so that speed control does not become unstable, but not too fast so that the field weakening becomes unstable. It should in most cases be enough to have PgainVoltageAngle (0x2075:09) set to 0 and only use IgainVoltageAngle (0x2075:04) to tune the response.

1. Write the following values in the drive:
   - RequestedControlMode (0x2020:12) = 3 (AC current mode)
   - CommandAcCurrent (0x2000:04) = 0
   - MaxCurrentChange (0x2095:03) = 0
2. Setup the transient recorder with the following signals:
   - RefTorque (0x2076:01)
   - ActTorque (0x2076:02)
   - VoltageAngle (0x2075:07)
3. Set Trig Mode to Greater than, Trig Value to 100, and increase Trig Delay.
4. Enable the drive.
5. Use the dyno to spin the motor into the field weakening area. You can check that field weakening is active by checking that FieldWeakeningControl: RegulatorStatus (0x2075:02) > 0.
6. Arm transient recorder trigger.
7. Do a step up in CommandAcCurrent (0x2000:04).
8. Set CommandAcCurrent (0x2000:04) = 0.
9. Plot the response. The response time should be around 100ms. If VoltageAngle gets saturated, use smaller current steps.
10. Adjust IgainVoltageAngle (0x2075:04) and repeat the step to obtain the desired response time.
11. Change back the temporary changes and then save to EEPROM:
    - RequestedControlMode (0x2020:12)
    - MaxCurrentChange (0x2095:03)

5.4.2 Max voltage angle tuning with dyno

The torque/power normally increases with increasing voltage angle but only up to a certain point, after which the torque and power decreases with increased angle and current, and this point shall never be exceeded.

For a PM-motor, setting this to 90 degrees is safe. It may be possible to get some more torque at high speeds if this is increased.

1. Set RequestedControlMode (0x2020:12) = 3 (AC current mode).
2. Enable the drive.
3. Use the dyno to spin the motor to max speed.
4. Set CommandAcCurrent (0x2000:04) = max current.
5. If RegulatorStatus (0x2075:02) = 3 then voltage angle has reached the limit. Try adjusting MaxVoltageAngle (0x2075:11) until you find a local maximum on DcBusPower (0x2073:03).

5.5 Current angle calculation (MTPA)

Although for PMAC motors the quadrature current Iq is creating the torque, an extra amount of torque can be produced by id current. This is due to reluctance effects. A PMAC motor without any reluctance torque (Ld = Lq) should always have a current angle of zero for best performance, i.e. all current goes in Iq and no current in Id. If the motor has reluctance torque (Ld < Lq), the difference in inductances and rotor flux will determine the current angle for maximum torque. Algorithms which exploit this effect are called “Maximum Torque Per Amp”. (MTPA) algorithms.

The current angle is defined as zero when all current is in the q-axis and increases with negative d-current.

The figure below shows an example of how the torque constant (Nm/A) could differ with different current angles. The task is to fit this line as well as possible by determining the best current angle for two (or more) load points. It is recommended that these load points are chosen as the continuous torque and the maximum torque unless other load points are more important. The measurements should be conducted at motor temperatures normally seen in the application (usually 80 to 120 ° C).

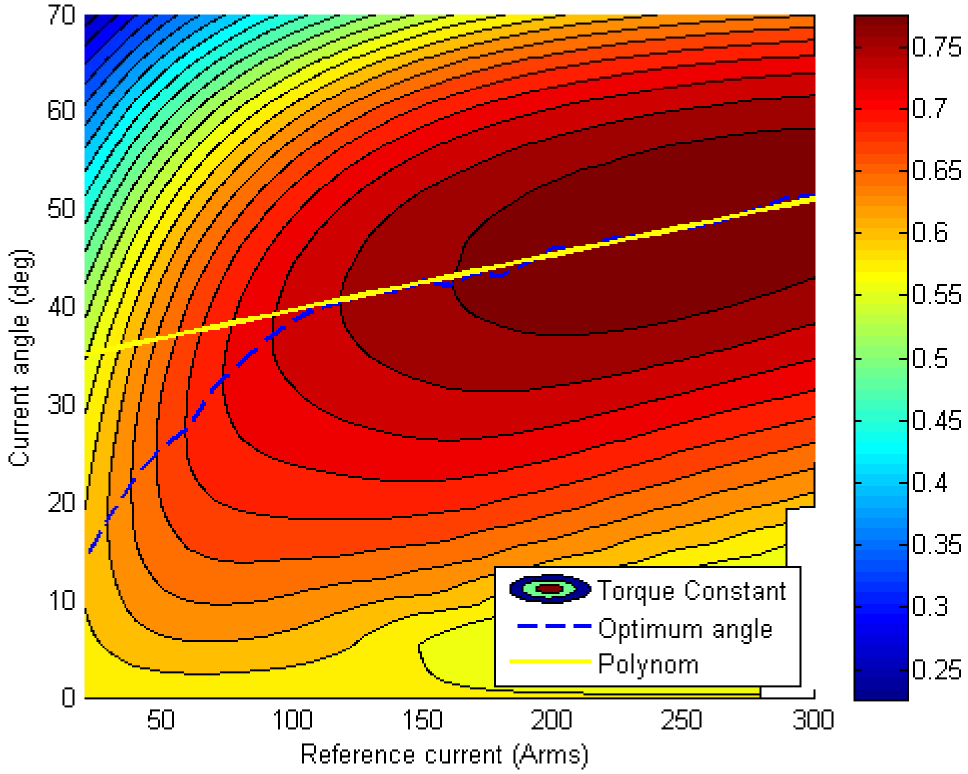


Figure 12 Optimum current angle as function of total current for a PMAC motor

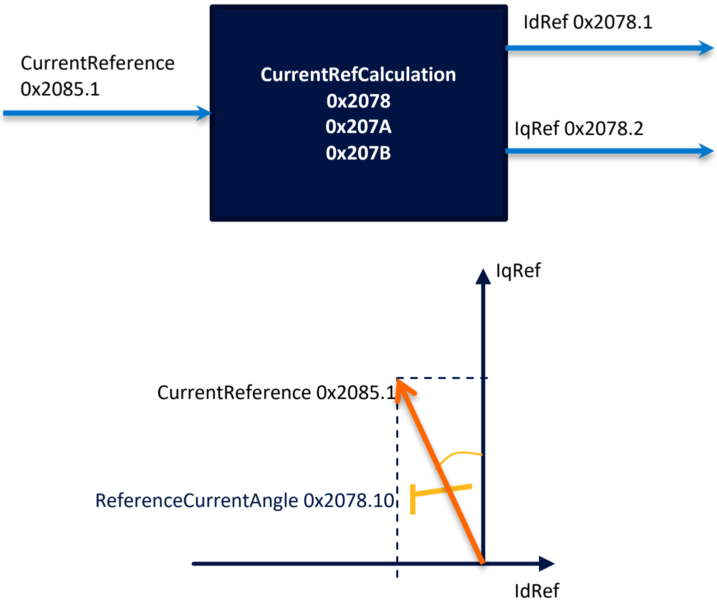

CurrentReference 0x2085.1
CurrentRefCalculation
0x2078
0x207A
0x207B
IdRef 0x2078.1
IqRef 0x2078.2
IqRef
CurrentReference 0x2085.1
ReferenceCurrentAngle 0x2078.10
IdRef

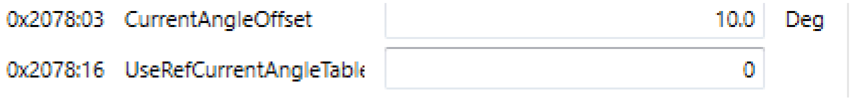
Figure 13 Reference Current calculation for current controller using MTPA

GVI provides two implementations of MTPA algorithms in the module CurrentRefCalculation shown in Figure 13.

Offset - Gain -Method

- Current Angle Interpolation Tables

For Parker’s GVM and NX8HMx Motors the optimum is usually a constant offset of 10°, so it is the best to set the parameters as follows



0x2078:03	CurrentAngleOffset	10.0	Deg
0x2078:16	UseRefCurrentAngleTabl	0	

Figure 14 Current Angle Offset

If the optimal angle is unknown, please refer to chapter 10.1 for identification procedures.

5.6 Flux calculation and Torque estimation

In order to decouple the Q and D controllers and accurately estimate torque, it is necessary to estimate the magnetic flux in the motor.

The FluxCalculationMethod should be set on CALCULATED_DQ_FLUX.

There are different calculation methods used for the flux, depending on motor speed

#	Condition	Flux Calculation Method

1	DeltaStatorAngle > MinDeltaStatorAngleForFluxCalc	ActualDFlux = (UqRef - R*iq) / ω_{el}
		ActualQFlux = -(UdRef - R*id) / ω_{el}

2	DeltaStatorAngle < 0.5*MinDeltaStatorAngleForFluxCalc	ActualDFlux = DFluxInterpolation(MotorCurrentUnfiltered)
		ActualQFlux = QFluxInterpolation(MotorCurrentUnfiltered)

3	DeltaStatorAngle between condition #1 and #2	Interpolate between flux calculation method #1 and #2

Table 9 Different flux calculation method depending on the speed (DeltaStatorAngle) of the motor. DFluxInterpolation / QFluxInterpolation are 8-point lookup tables.

The value for the DFluxInterpolation and QFluxInterpolation tables are available on request for Parker’s GVM motors (see application note chapter 1.1.6). For other motors they must be tuned according to chapter 10.5.

The torque is calculated from DFlux and QFlux according to following equations.

ActTorque = 3/2 * MotorPoles/2 * (ActualDFlux * Iq - ActualQFlux * Id)

RefTorque = 3/2 * MotorPoles/2 * (ActualDFlux * IqRef - ActualQFlux * IdRef)

6 Application

For the Application Layer to work as expected it must be ensured that bit 15 is set in the ApplicationSetupWord (0x2010:10)

6.1 Control modes

The control modes in the GVI are explained in this chapter.

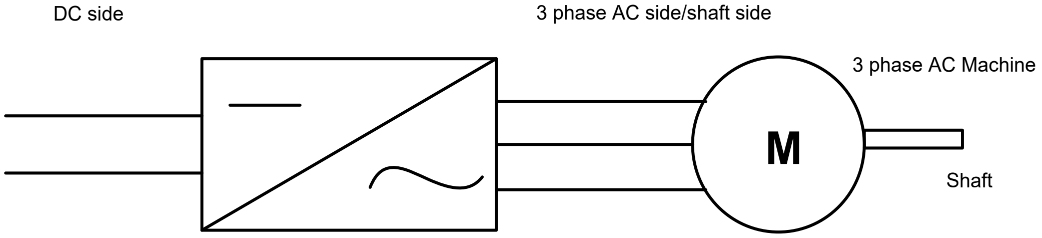


Figure 15. Principal layout of inverter.

The appropriate control mode for the GVI must be chosen depending on the application. This is selected by Object 0x2020:12 *RequestedControlMode* as shown in Table 11.

ActualControlMode (0x2020:13) shows the currently active control mode and can be seen in TxPDO4 or Diagnostics 3 Message (see CAN Message Database, chapter 1.1.6).

Value	Description
0	Speed Mode
3	AC Current Mode
5	Torque Mode
8	DC Voltage Mode

Table 10: Values of supported control modes for RequestedControlMode and Actual Control Mode

6.1.1.1 Speed control mode

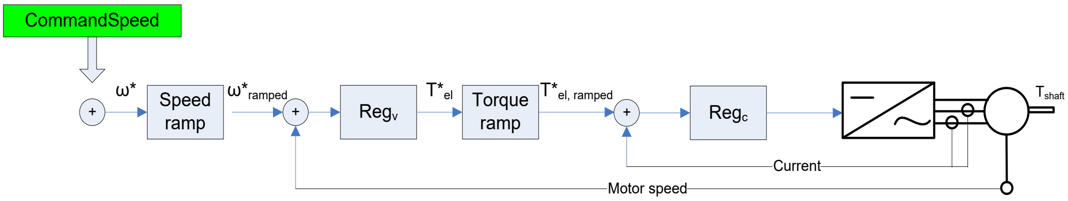


Figure 16. Controller structure in speed control mode.

In speed mode, the speed on the shaft (see Figure 15 & 16) is controlled by a PI controller. The reference value for the speed is set by *CommandSpeed*. Details in chapter Speed mode.

Related parameters to speed control can be found in OD index 0x2000 (ApplicationCommands), 0x2090 (SpeedRegulatorPgain),0x2091 (SpeedRegulatorIgainSmall), 0x2092 (SpeedRegulatorIgainMedium), 0x2093 (SpeedRegulatorIgainLarge).

See **OD** document for explanation of each parameter (chapter 1.1.6).

6.1.1.2 AC current control mode

In current mode, the current, see Figure 15 is controlled by a PI controller. The reference value for the current is *CommandAcCurrent.*

Parameters related to current control can be found in OD index 0x2072 (CurrentController), 0x2075 (FieldWeakeningControl) and 0x2078 (ReferenceCurrents). See **OD** document for explanation of each parameter (chapter 1.1.6).

6.1.1.3 Torque control mode

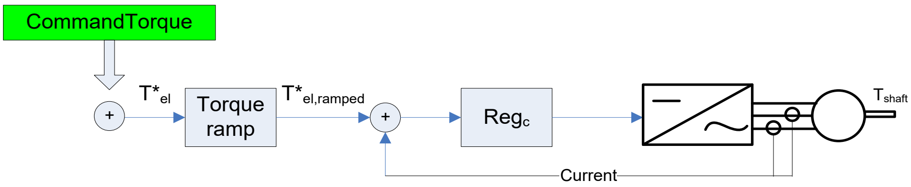


Figure 17.Controller structure in torque control mode.

In torque mode, the electrodynamic torque, see Figure 15 & 17 is controlled by a PI controller. The reference value for the torque is set by *CommandTorque*.

Parameters related to torque control require further tuning.

6.1.1.4 DC voltage control mode

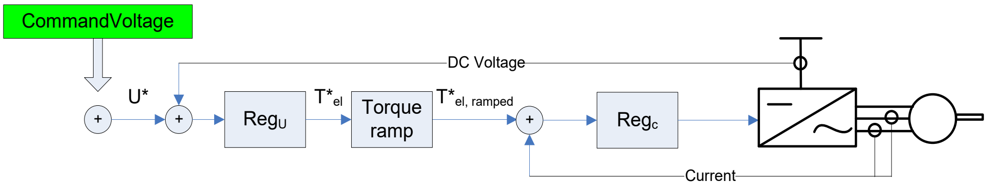


Figure 18. Controller structure in DC voltage control mode

In DC Voltage Control Mode, the voltage on the DC side, see Figure 15 & 18, is controlled by a lead lag controller by default. The reference value for the voltage is set by *CommandVoltage*.

Related parameters to voltage control can be found in OD index 0x2640 (DCVoltageControl). See **OD** document and application note for explanation of each parameter (chapter 1.1.6).

6.2 Speed mode

The speed controller is shown in Figure 19 and consists of three main blocks, namely Speed Ramp, Best Performance Curve and Speed PI Control. The Command Speed is the input to the speed controller and the limited Current reference is the output. This output is fed to the Current Regulator. It is also possible to disable the speed controller and use AC Current control instead. The input is then the AC Command Current.

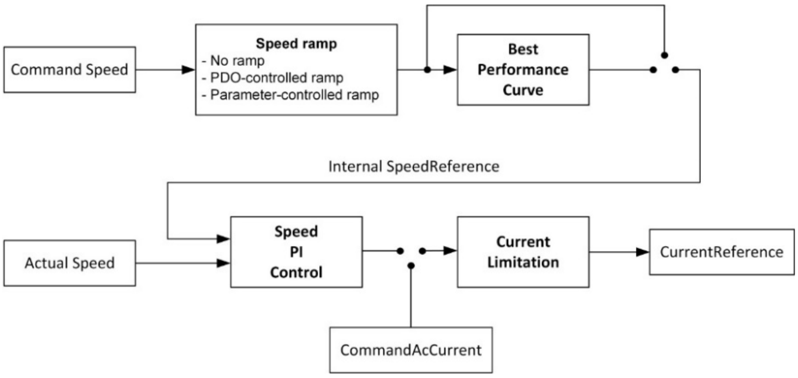


Figure 19. Speed control

Common parameters and variables speed control mode are shown in Table 12. (for AC Current Mode it is only the parameter RequestedControlMode that is of interest).

Index	Sub-index	Name	Description
0x2020	12	RequestedControlMode	0 = Speed Mode 3 = AC Current Mode 5 = Torque Mode 8 = DC Voltage Mode
0x2020	1	Speed Ramp Mode	1 = Speed control with no ramp 2 = Not valid 3 = Not valid 4 = Speed control with PDO controlled ramp, see 6.2.1.4 for how to setup the PDO 5 = Speed control with parameter controlled ramp
0x2020	6	Best Performance Curve Mode	0 = No BPC mode (BPC disabled) 1 = BPC mode 2 = BPC PDO mode 3 = BPC Reduction Mode
0x2020	10	Application Setup Word	Bit 3 Inverted Rotational Direction False or True
0x2090	1:1-4	Speed PI Controller: P-gain Set 0-3	P-gain for the speed regulator
0x2091	1:1-4	Speed PI Controller: I-gain Small Set 0-3	Igain (Small) for the speed regulator When used: Speed Ramp Mode = 1: Always used Speed Ramp Mode = 4 or 5: During constant speed and acceleration
0x2092	1:1-4	Speed PI Controller: I-gain Medium Set 0-3	Igain (Medium) for the speed regulator When used: Speed Ramp Mode = 1: Not used Speed ramp Mode = 4 or 5: During rollback
0x2093	1:1-4	Speed PI Controller: I-gain Large Set 0-3	Igain (Large) for the speed regulator When used: Speed Ramp Mode = 1: Not used Speed Ramp Mode = 4 or 5: During standing and deceleration
The different sets of Speed Regulator P- and I-gain in 0x2090-0x2093 can be altered by setting bit 4 in CommandWord, 0x2000:1. With this bit set Set 1 is used, if unset Set 0 is used. Normally 2 sets covers most applications.			
2096h	1	Internal Speed Reference	Read only variable. Speed reference input to speed PI controller (internal SW variable). Differs from Command Speed when ramps and/or BPC are used

Table 11 Speed Control – Parameters and Variables

NOTE

Inverted Rotational Direction does not invert the feedback for the actual torque: 0x2076 subindex 8, the Iq: 0x2077 subindex 1 or the Id: 0x2077 subindex 2.

6.2.1 Speed ramp

The Speed Ramp functionality is mainly described for the traction application, but of course the ramp functionality can be utilized also for other applications.

There are three different speed ramp modes:

- No ramp
- PDO controlled ramp
- Parameter Controlled ramp

6.2.1.1 Parameter controlled ramp General

The purpose of ramping the Command Speed is to get a smooth acceleration/deceleration, save battery lifetime and give the truck equal stop distance regardless of load and slopes when decelerating. The speed ramp functionality generates set values for the speed PI controller. See Figure 19. Several parameters can be set for controlling of the behavior of the ramp:

Index	Sub-index	Name	Description
0x2020	1	Speed Ramp Mode	1 = No ramp 4 = Speed control with PDO controlled ramp 5 = Speed control with parameter controlled ramp
0x2020	7	Rollback Speed	0-8000 rpm Controls if the truck stands still or starts rolling when slowing down to zero speed in a slope. Setting the speed to zero means stand still (hillhold). Only used in speed ramp modes with ramps (Speed Ramp Mode = 4 or 5)
0x2020	10	Application Setup Word	Bit 0 Adjust Speed During Acceleration If set to true, the truck may accelerate faster than the ramp says if accelerating in a steep downhill slope. Bit 1 Adjust Speed During Deceleration If set to true, the truck may decelerate faster than the ramp says if decelerating in a steep uphill slope. Bit 2 Enter Rollback Mode At Zero Speed If set to true, the truck stops before it eventually starts to roll. Only used if Rollback Speed is NOT zero

Table 12 Speed Ramp – Parameters and Variables

6.2.1.2 Basics

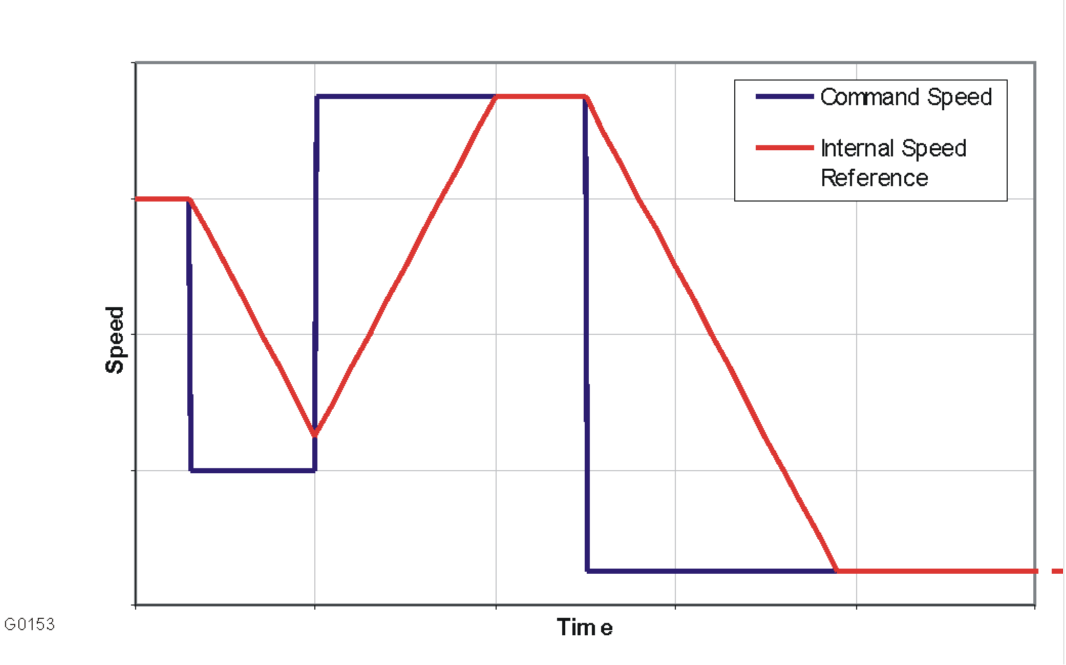


Figure 20 The basic speed ramp

The basic speed ramp functionality is shown in Figure 20 . When a step change in the Command Speed is sent to the GVI. The Internal Speed Reference value is ramped from the previous Internal Speed Reference speed value to the new Internal Speed Reference.

6.2.1.3 No ramp

In the No Ramp mode, the speed ramp controller is not used and only the speed PI controller is used. Only *CommandAll* and *CommandSpeed* are used for control.

6.2.1.4 PDO-controlled ramp

In the PDO Controlled Ramp mode, all ramp information is sent inside the PDO. See GVI CAN Message Database, chapter 1.1.6.

The *CommandAccelerationChange* (0x2000:01) and *CommandDecelerationChange* (0x2000:06) are the ramp values. The Command Word uses the same bits as for No Ramp mode to control the parameter set in the speed PI controller (bit 4 in the *CommandAll)*. The differences compared to using *Parameter Ramp Mode* are that *reverse brake* (plug brake), *neutral brake* and *foot pedal brake* are to be handled within the VCU.

6.2.1.5 Parameter-controlled ramp

In the Parameter Controlled Ramp mode the actual ramp values are given by parameters (accessible via SDOs) preset in GVI. This setting is often used if no PDO´s are used, for instance in a standalone application, where the speed reference is given from an Analog input.

Bit	Description
1-2	Speed Neutral Brake Ramp Parameter Set (0-3). See object indexes 2104h - 2105h
11-12	Speed Ramp Parameter Set (0-3). See object indexes 2100h - 2103h
13	Foot Brake Active, 1 = True

Table 13 Parameter Controlled Ramp - Command Word 0x2000:1

There are a total of four groups of speed ramp parameters, namely Forward Acceleration, Forward Deceleration, Reverse Acceleration and Reverse Deceleration.

Index	Sub-index	Name	Description
Forward Acceleration (0x2100) and Reverse Acceleration (0x2102)	1	Parameter Set 0	Used according to bits 11-12 in the Command Word. Used if Command Speed > Actual Speed and Command Speed ≠ 0 and has the same sign as the Actual Speed
	2	Parameter Set 1	
	3	Parameter Set 2	
	4	Parameter Set 3	
Forward Deceleration (0x2101) and Reverse Deceleration (0x2103)	1	Parameter Set 0	Used according to bits 11-12 in the Command Word if Command Speed < Actual Speed and Command Speed ≠ 0 and has the same sign as the Actual Speed. Also called Reduction brake
	2	Parameter Set 1	
	3	Parameter Set 2	
	4	Parameter Set 3	

Table 14 Parameter Controlled Ramp - Ramp Parameters

Additionally, there are also a total of six special speed ramp parameters setup in two groups, namely Special Forward Deceleration and Special Reverse Deceleration.

Index	Sub-index	Name	Description
Special Forward Deceleration (0x2104) and Special Reverse Deceleration (0x2105)	1	Foot Brake	Used if bit 13 in the Command Word is set
	2	Reverse Brake	Used if Command Speed has the opposite sign of Actual Speed. UsedDeceleration = NeutralBrake + (ReverseBrake - NeutralBrake) *CommandSpeed / UsedMaxSpeedInRpm Often also called plugging, plug braking or Switch back braking
	3	Neutral Brake Set 0	Used according to bits 1-2 in the Command Word if Command Speed = 0, or direction is set to neutral. Also called coasting
	4	Neutral Brake Set 1	
	5	Neutral Brake Set 2	
	6	Neutral Brake Set 3	

Table 15 Special speed ramp parameters

These speed ramp parameters are used during accelerations and decelerations according to descriptions in Table 15 & 16 and also in a graphical view in Figure 21.

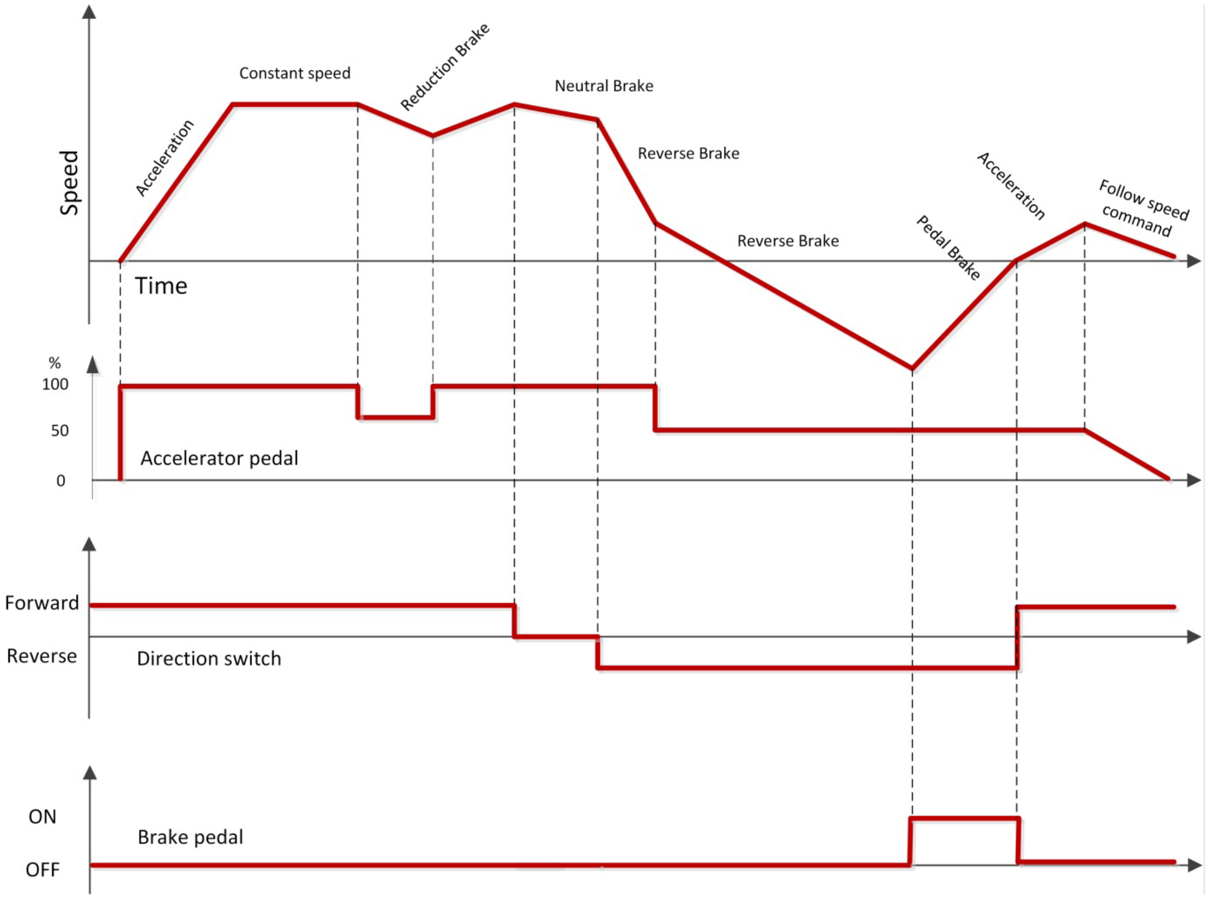


Figure 21 Graphical view describing where the different modes of ramping can occur in a Parameter Controlled Ramp setup. The accelerator pedal input can also be seen as the commanded speed value from a PDO

In order to improve or easily adjust the ramp functionality while using Parameter Controlled Ramp it is also possible to include the parameters 0x2000:07 and 0x2000:08 described in Table 17 in a PDO. See chapter 4.4.4 to see how to change the default PDO mapping.

Index	Sub-index	Name	Description

0x2000	7	Acceleration Reduction Factor	UsedAcceleration = Acceleration *AccelerationReductionFactor / 128
			AccelerationReductionFactor = 128 means no reduction

0x2000	8	Max Speed Reduction Factor	Reduction of parameter Max Speed that is used during Reverse Brake
			UsedMaxSpeed= MaxSpeedInRpm*(MaxSpeedReductionFactor+1)/256
			MaxSpeedReductionFactor = 255 means no reduction

0x2120	1	Max Speed	Used during Reverse Brake

Table 16 Parameter Controlled Ramp – Parameters

6.2.1.6 Slopes

In case of braking or slowing down in an uphill slope the gravity will contribute to the deceleration. If the slope is steep the truck will slow down faster than the Internal Speed Reference. In order to allow the truck to decelerate faster than the ramp, the ramp is adjusted to the actual speed when decelerating in a steep slope as can be seen in Figure 22. This functionality is possible to enable or disable with bit 1 (Adjust Speed During Deceleration) in Application Setup Word (0x2020:10).

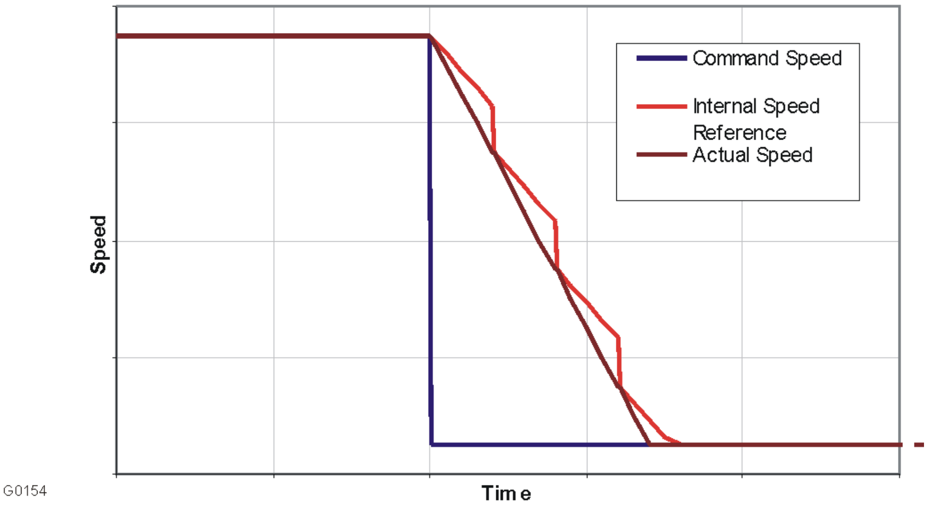


Figure 22 Adjustment of deceleration ramp when braking in an uphill slope

In case of acceleration in a downhill slope the gravitation will contribute to the acceleration. If the slope is steep the truck will accelerate faster than the Internal Speed Reference. In order to allow the truck to accelerate faster than the ramp, the ramp is adjusted to the actual speed when accelerating in a steep slope. This functionality is possible to enable or disable with bit 0 (Adjust Speed During Acceleration) in Application Setup Word (0x2020:10).

These two bits, bit 0 and bit 1 in 0x2020:10 can only be used when utilizing Speed Ramp Mode = Speed Control with PDO Controlled Ramp or Speed Control with Parameter Controlled ramp (4 or 5) , hence not when No Ramp Mode is chosen.

6.2.1.7 Rollback

In order to apply this feature the Speed Ramp Mode, 0x2020:1, needs to be 4 or 5, so the ramp functionality is used.

When slowing down to zero speed on a slope the truck can either continue to stand still or start rolling. The parameter used to control the behavior is called the Rollback Speed (0x2020:7 in **OD**). Setting this parameter to zero means that the truck will stand still after slowing down. If the Rollback Speed is set > 0 the truck will continue rolling after slowing down. In a downhill slope the truck will either reach zero speed or not depending on bit 2 (Enter Rollback Mode at Zero Speed) in Application Setup Word (0x2020:10), see Table 13. After slowing down, the truck will continue rolling down with a speed larger than zero but less than or equal to the Rollback Speed. Since an accelerating torque is not allowed the speed will be lower than the Rollback Speed if the slope is flat. In an uphill slope the truck will reach zero speed and then start rolling backwards down the slope with a speed larger than zero but less than or equal to the Rollback Speed value.

The bit 9 (Rollback disable) in the Command Word can be used to override the Rollback Speed parameter, i.e. the rollback functionality becomes disabled if the bit is set. This can be used to switch between hill-hold and rollback in real time, i.e. if you would like to stop the rollback when the driver pushes the brake pedal.

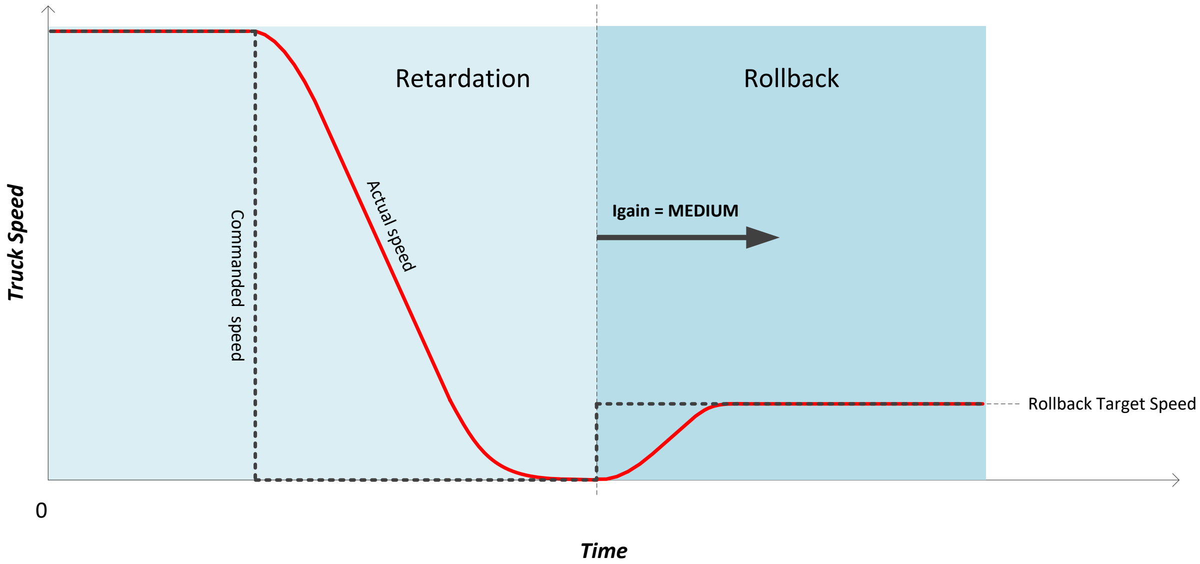


Figure 23 Principle of how Rollback works, Igain used is IgainMedium. This shows typical behavior if stopping in a downhill slope and rollback speed is downwards

6.2.2 Speed PID Controller

The Speed PID controller is a standard PI controller with two inputs, the Internal Speed Reference and the Actual Speed, and one output, the Current reference. Depending if speed ramps are used or not the PI controller works a bit differently. If no ramps are used, there is only one I-gain used, I-gain Small. If ramps are used, there are a total of three I-gains, I-gain Small, Medium and Large. This is summarized in Table 18 along with the other parameters for the Speed PID Controller.

In order to dynamically quickly change performance of the vehicle response, it is possible to predefine 2 different parameter sets, each of them including Pgain, Dgain, IgainSmall, IgainMedium and IgainLarge.

D-gain (0X208F) is set to 0 by default (disabled)

Index	Sub-index	Name	Description
0x208F	1-4*	Speed PID Controller: D-gain Set 0-3	D-gain for the speed regulator
0x2090	1-4*	Speed PID Controller: P-gain Set 0-3	P-gain for the speed regulator
0x2091	1-4*	Speed PID Controller I-gain Small Set 0-3	Igain (Small) for the speed regulator When used: Speed Ramp Mode = 1: Always used Speed Ramp Mode = 4 or 5: During constant speed and acceleration
0x2092	1-4*	Speed PID Controller I-gain Medium Set 0-3	Igain (Medium) for the speed regulator When used: Speed Ramp Mode = 1: Not used Speed ramp Mode = 4 or 5: During rollback
0x2093	1-4*	Speed PID Controller I-gain Large Set 0-3	Igain (Large) for the speed regulator When used: Speed Ramp Mode = 1: Not used Speed Ramp Mode = 4 or 5: During standing and deceleration

*Only 2 sets can be used in GVI (set 0 and 1). The different sets of Speed Regulator D-, P- and I-gain in 0x208F-0 0x2090-0x2093 can be altered by setting bit 4 in CommandWord, 0x2000:1. With this bit set Set 1 is used, if unset Set 0 is used. Normally 2 sets cover most applications.

Table 17 PID controller – Parameters

6.2.3 Tuning of speed control

Tuning of the parameters in the Speed Control determines the response characteristics for the GVI's Speed Control.

WARNING

Wrong parameter tuning can result in oscillation and unstable control.

6.2.3.1 Tuning in parameter or PDO controlled ramp mode

With Speed Control Mode equal to Parameter or PDO Controlled Ramp mode, the optimum response for various operating conditions (i.e. slow speed, high speed, braking, holding) is mainly tuned with the acceleration and deceleration values, i.e.

- PDO Controlled Ramp: Command Acceleration and Command Deceleration that are set by the truck controller.
- Parameter Controlled Ramp: Ramp Parameters in Table15.

The values to be used are normally determined by the operating requirements for the vehicle.

Usually it is not necessary to utilize the two sets of Speed PID Controller parameters, which means that Set 0 can be used at all times. This means that bit 4 in the Command Word should be equal to zero. The default Speed PID Controller parameters (D-gain, P-gain, I-gain (Small, Medium and Large) are satisfactory for the majority of applications and normally do not require further adjustment. However, optimum values can be established empirically by alternately adjusting a parameter and operating the vehicle until the desired “feel” or response is achieved. Some general tuning guidelines follow.

Speed PID Controller P-gain Set 0-3 (2090h:1-4): Should be increased gradually from default value until the vehicle's response to operator controls is acceptable.

- Excessively high P-gain can cause instability and excessive power consumption.
- Speed PI Controller I-gain (Small, Medium and Large) Set 0-3 (2091h-2093h:1-4): The I-gain is typically used to increase sensitivity to small changes in operator controls when at or near zero speed. Igain, in combination with P-gain, is beneficial for holding zero speed or to overcome friction when starting. Excessive I-gain can cause the motor/load to break into oscillation. I-gain should not be introduced until P-gain has been adjusted and the motor/load exhibits stable operation. It should be observed that there are three I-gains (Small, Medium and Large) that are used in PDO and Parameter Controlled Ramp modes. Table 12 explains during which operating conditions the Small, Medium and Large I-gain, respectively, are used. In Figure 24 it is visualized how the different types of I-gains (Small and Large) are working during a typical acceleration and deceleration phase. Figure 24 shows how the ramping of internal speed reference and changing speed controller integral gains typically is done during acceleration and deceleration.
- D gain Set 0-3 (208h:1-4) is set to 0 (disable) by default. This parameter is to be used if speed overshoot occurs when releasing load (pump application).

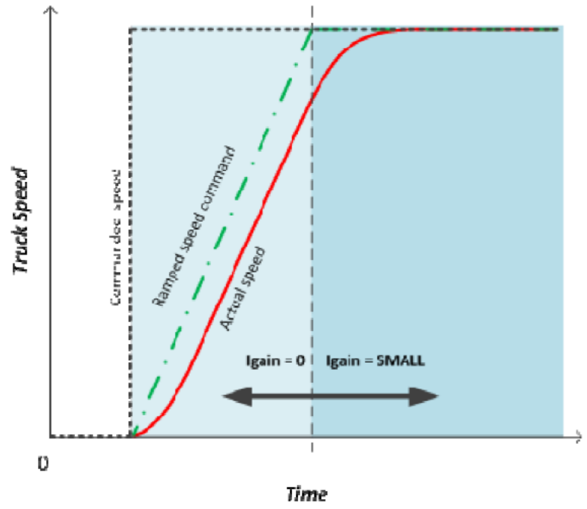
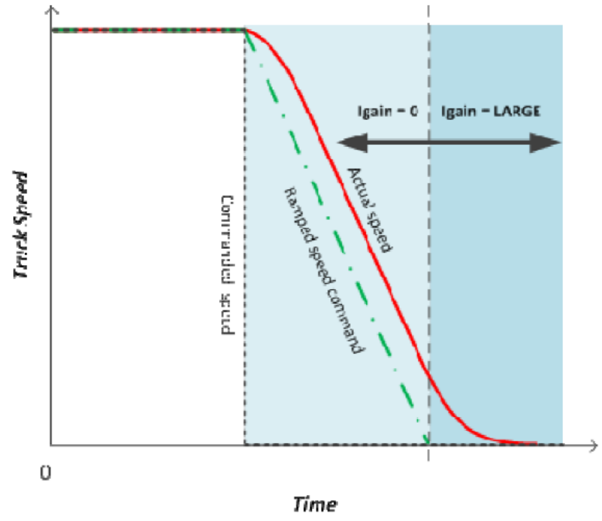
Acceleration	
Deceleration	

Figure 24 Use of I-gain during a typical acceleration and deceleration phase. This is valid only for SpeedRamp mode set to 4 or 5

The Current Limitation parameters Table 20 also have to be set to appropriate values so that optimum response for various operating conditions is achieved. It is also possible to get dynamic current limit adjustability during operation by including the parameter Max Acceleration Torque Current in the PDO.

6.2.3.2 Tuning in No ramp mode

In Speed Ramp Mode = 1 (Speed Control with no ramp) the two sets of Speed PI Controller parameters that are provided can be utilized. The vehicle master controller can dynamically select the optimum response for various operating conditions (i.e. slow speed, high speed, braking, holding). The bit 4 in Command Word is used to choose 1 out of the 2 possible parameter sets that are to be used in the speed PI control.

Optimum values for the Speed PI Controller parameters are usually established empirically by alternately adjusting a parameter and operating the vehicle until the desired “feel” or response is achieved (see tuning guidelines in chapter 6.2.3.1).

It should be observed that in the No Ramp mode only the I-gain Small is utilized (see Table 18).

6.2.4 Best performance curve

The Best Performance Curve functionality is implemented after the Speed Ramp and before the Speed PI controller, as can be seen in the Figure 19. The purpose of the Best Performance Curve, or the 8-point curve as it also is called, is to save battery lifetime and get an optimal efficiency out of the GVI -motor system. The Best Performance Curve limits the motor current as a function of the motor speed (Actual Speed).

An example of a Best Performance Curve is shown in Figure 25. The curve is specified by 8 points and a linear interpolation is used in between the points. One value for Motor current and one value for Speed define each point. The Motor current value of the first point will be valid down to zero Speed and the Motor current value of the 8th point will be valid for Speeds higher than the 8th speed point. The speed points must be incremental.

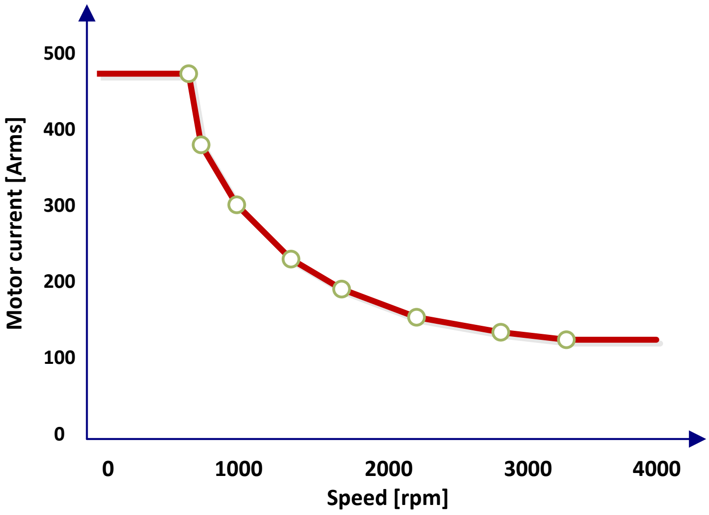


Figure 25 Example of Best Performance Curve (BPC).Speed and Motor Current points are set in 0x2140:1-8 and 0x2141:1-8

The BPC functionality uses a PI controller for limitation of the current which gives smoother operation than with an instant current limitation. To allow over-shoot in the PI controller there needs to be a certain margin between the curve and the maximum available motor current. The gains of the controller can be adjusted in order to give suitable response time for the system. BPC is available both for traction and hydraulic applications.

Index	Sub-index	Name	Description
0x2020	6	Best Performance Curve Mode	0 = No BPC mode (BPC disabled) 1 = BPC mode 2 = BPC PDO mode 3 = BPC Reduction Mode
0x2140	1-8	BestPerformanceCurveSpeed	Speed Table, speed values in incremental order starting from index 1
0x2141	1-8	BestPerformanceCurveCurrent	Current Table, limited current for corresponding index in speed table
0x2142	2	BPCIgain	Best performance curve PI-controller I-gain, default value is 100 P-gain is set to 0
0x2142	3	InternalCurrentReference	Internal SW variable, input current reference to the best performance curve in PI-controller. In BPC_PDO_MODE the absolute value of the written value is used. (Only in this mode is it possible to write to this index)
0x2142	4	ReductionFactor	In BPC Reduction Mode this value is used to reduce the current in the BPC-curve InternalTorqueCurrentReference = InternalTorqueCurrentReference *Reductionfactor/256

Table 18 Best Performance Curve Parameters

6.3 Limits

6.3.1 Supervision

With IndependentSupervision 0x2500 trip levels for different speed, current and voltages can be set. Also, this module allows to see the currently active limitations for speed, current and voltages.

6.3.2 Absolute current limits

The amplitude and derivate of the unlimited Current, which is the output of the Speed PI controller, is checked and if necessary limited in the Current Limitation block (see Figure 19).

The speed and the current have the same sign if power is fed from the GVI to the motor. If the speed and the current have opposite signs, the power is fed from the motor to the GVI (regeneration). From that follows that if the Actual Speed and the Current has the same sign, the maximum amplitude is given by the parameter Max Acceleration Current (0x2095:1) in Table 20. With opposite signs on the Actual Speed and the Current, the maximum amplitude is given by the parameter Max Brake Current (0x2095:2). It might be necessary to have different current limits, not only dependent on acceleration and brake state, but also dependent on the speed direction see objects used in Figure 26.

The derivate of the Current is limited in order to avoid noise and unnecessary wear in couplings and gearboxes. The maximum allowed Current derivate is controlled with the parameter Max Current Change.

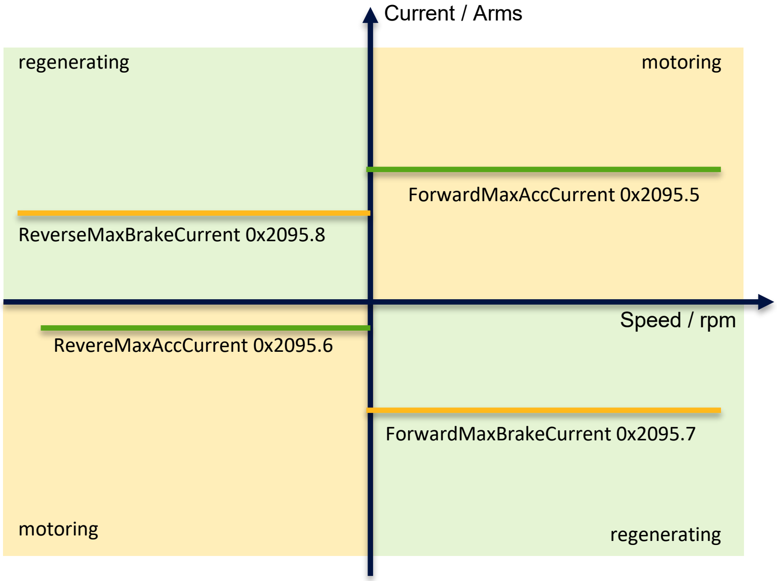

Current / Arms
regenerating
motoring
ForwardMaxAccCurrent 0x2095.5
ReverseMaxBrakeCurrent 0x2095.8
Speed / rpm
RevereMaxAccCurrent 0x2095.6
ForwardMaxBrakeCurrent 0x2095.7
motoring
regenerating

Figure 26 Absolute Current Limitation Parameters (depending on direction/acceleration)

Index	Sub-index	Name	Description
0x2095	1	Max Acceleration Current	Sets both Forward and Reverse Max Acceleration Torque Currents to the same value. Hence, this setting changes both 0x2095:5 and 6 Unit: ARMS
0x2095	2	Max Brake Current	Sets both Forward and Reverse Max Brake Torque Currents to the same value. Hence, this setting changes both 0x2095:7 and 8 Unit: ARMS
0x2095	3	Max Current Change	Unit : deciARMS/ms
0x2095	5	Forward Max Acceleration Current	Unit : ARMS
0x2095	6	Reverse Max Acceleration Current	Unit : ARMS
0x2095	7	Forward Max Brake Current	Unit : ARMS
0x2095	8	Forward Max Brake Current	Unit : ARMS

Table 19 Current Limitation – Parameters

6.3.3 Linear Current Reductions

6.3.3.1 General

The available Current can be reduced for a number of reasons, including:

- Over Temperature on Power Stage or Motor (For Power Stage internal temperature supervision the values are fixed, for details see Product Manual section 7.2)
- Over Voltage
- Under Voltage
- Over Speed
- Current Limitation sent via CAN
- Event driven, often customized

If the Current is reduced due to internal reasons the warning bit in the StatusWord should be lit. See chapter “Emergency Error Code” in **OD** for explanation of warning bits and when they are lit.

The actual limitation can be read from **OD** objects 0x2095:19. Type of limitation can be read from **OD** objects 0x2095:20 and 21.

Index	Sub-index	Name	Description
0x2095	19 20 21	ActualLimitationType AccLimitationType BrakeLimitationType	0 = No limitation 1 = Event limitation 2 = Motor temperature limitation 3 = Low DC voltage limitation 4 = High DC voltage limitation 5 = Low Speed limitation 6 = High Speed limitation 8 = DC Power limitation 9 = External current limitation 10 = External torque limitation 11 = Heatsink temperature 12 = Energy 13 = Max current 15 = Switching frequency

Table 20 Current limitation types

6.3.3.2 Parameters and variables related to Current Reductions

The maximum Motor Current can be read and written in **OD** objects 0x2095:5,6,7,8.

Index	Sub-index	Name	Description
0x2095	5	Forward Max Acceleration Current	Unit : ARMS
0x2095	6	Reverse Max Acceleration Current	Unit : ARMS
0x2095	7	Forward Max Brake Current	Unit : ARMS
0x2095	8	Forward Max Brake Current	Unit : ARMS

Table 21 Nominal maximum current limit in the four quadrants

If the Motor Current is reduced the momentarily available Motor Current can be read in **OD** objects 0x2095:9, 10.

Index	Sub-index	Name	Description
0x2095	9	AbilityAccelerationCurrent	Unit : ARMS
0x2095	10	AbilityBrakeCurrent	Unit : ARMS

Table 22 Ability Current (read only), resulting current availability after reduction

The available current is always the minimum of the AbilityCurrent from 0x2095: 9, 10 and CurrentLimit set in 0x2095: 5,6,7,8.

In order to temporarily limit the motor current it is also possible by via CAN using **OD** objects 0x2095:15, 16. The limited current will then be the minimum of the AbilityCurrent from 0x2095: 9, 10 and TorqueCurrentLimit in 0x2095:15,16. Remember that acceleration current and brake current are treated separately.

Index	Sub-index	Name	Description
0x2095	15	AccCurrentLimit	Unit : ARMS
0x2095	16	BrakeCurrentLimit	Unit : ARMS

Table 23 Externally (CAN) set motor current limitation.

Current-limiting parameters dependent on power stage temperature

These parameters cannot be adjusted via the **OD**. The drive temperature can be read in **OD** 0x2041:3.

The derating output starts when the power stage temperature is at 85°C (100% of available current) to 110°C (40% of available current). If temperature continues to increase, the power stage is disabled.

The same applies for negative temperature value, from -20°C to -50°C (low voltage GVI only)

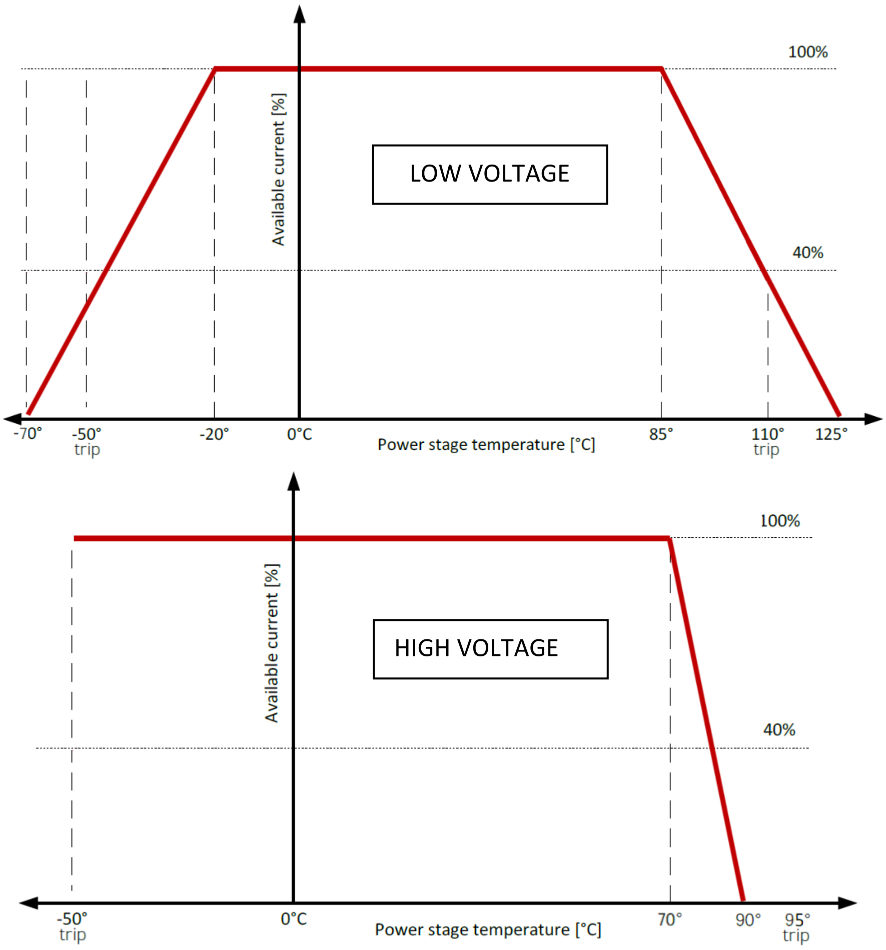


Figure 27 Graphical view of how power stage temperature limits the available motor current (low and high voltage GVI)

NOTE

This only applies to current used for acceleration not deceleration.

Current-limiting parameters dependent on DC bus voltage

These parameters can be adjusted via the **OD**, according to battery limits (refer to battery supplier)

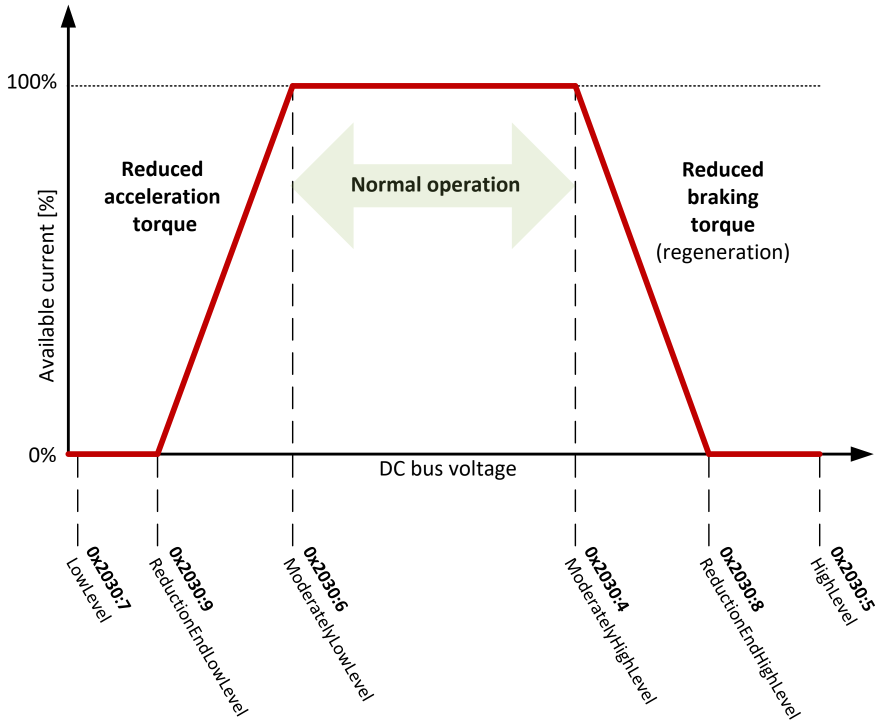


Figure 28 Graphical view of how DC bus voltage level could affect the available motor current.

Index	Sub-index	Name	Description
0x2030	4	ModeratelyHighLevel	If DC Voltage exceeds this level, a warning is set, and currents starts to linearly reduce to 0.
0x2030	5	HighLevel	If DC Voltage exceeds this level, the application trips.
0x2030	8	ReductionEndHighLevel	When DC Voltage reaches this level current is reduced to 0. Set to 0 to disable this feature. In order to allow full current for braking this must be set to 0
0x2030	6	ModeratelyLowLevel	If DC Voltage goes below this level, a warning is set, and currents starts to linearly reduce to 0. Warning level
0x2030	9	ReductionEndLowLevel	When DC Voltage reaches this level current is reduced to 0. Set to 0 to disable this feature
0x2030	7	ReductionEndLowLevel	If DC Voltage exceeds this level, the application trips. Set to 0 to disable this feature

Table 24 Parameters to adjust motor current limitation due to DC bus voltage.

Current-limiting parameters dependent on motor temperature.

These parameters are by default set to correct values in the object dictionary (for GVM/NX8HM series).

High temperature derating starts at 140°C (0x2040:5) and the drive is disable when temperature limit of 150°C (2040:7) is reached.

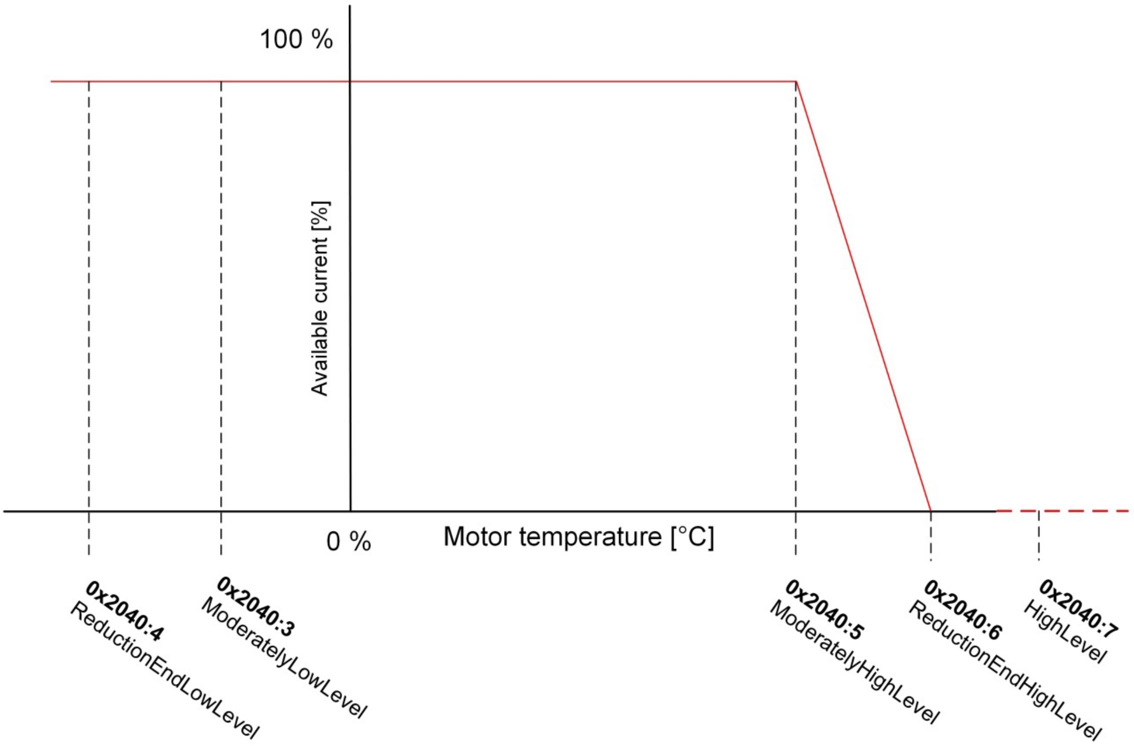


Figure 29 Motor temperature depending on motor current availability.

Index	Sub-index	Name	Description
0x2040	3	ModeratelyLowLevel	Level at which a Low Temp Status will become active.
0x2040	4	ReductionEndLowLevel	When motor temperature reaches this level current is reduced to 0. Only accelerating current is reduced. Set to 0 to disable this feature
0x2040	5	ModeratelyHighLevel	Motor temperature level when current starts to linearly reduce towards 0. Warning level
0x2040	6	ReductionEndHighLevel	When motor temperature reaches this level current is reduced to 0. Only accelerating current is reduced. Set to 0 to disable this feature
0x2040	7	HighLevel	Level at which a High Temp Status will become active. The application trips.

Table 25 Parameters to adjust motor current limitation due to motor temperature.

Current-limiting parameters depending on motor speed.

This limitation is also available if using AC Current Mode.

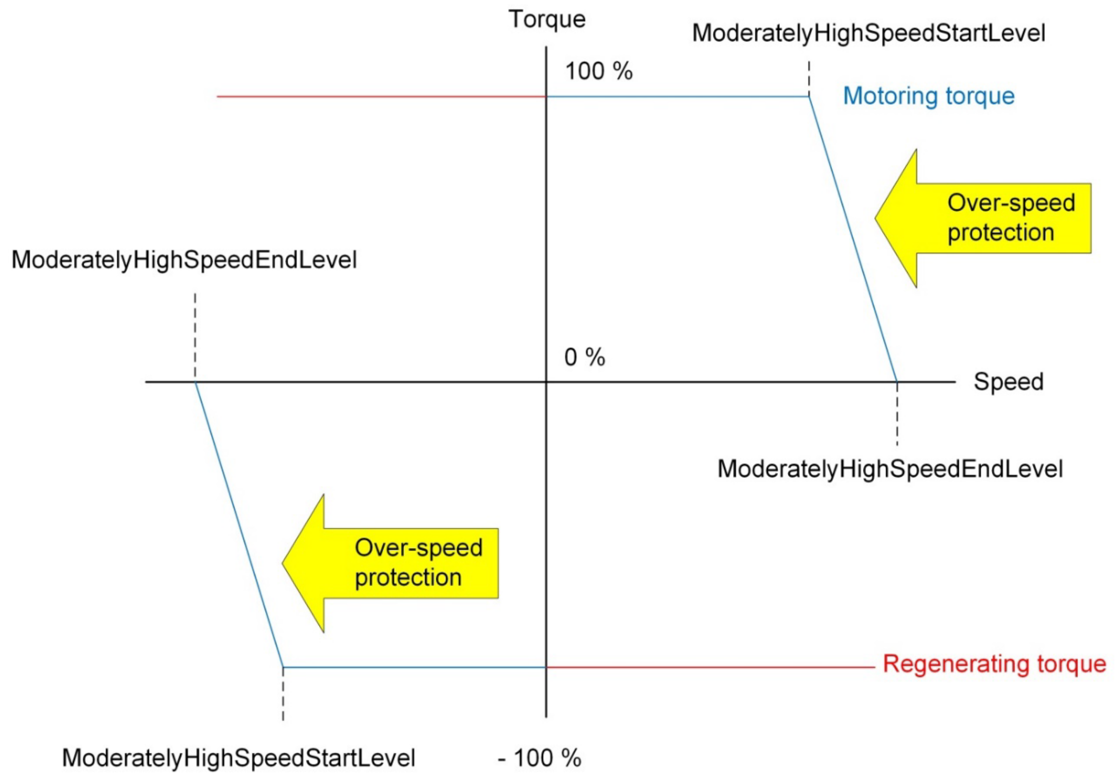

Torque
ModeratelyHighSpeedStartLevel
100 %
Motoring torque
Over-speed protection
ModeratelyHighSpeedEndLevel
0 %
Speed
ModeratelyHighSpeedEndLevel
Over-speed protection
Regenerating torque
ModeratelyHighSpeedStartLevel
- 100 %

Figure 30 Motor current limitation due to motor speed (full torques at reverse speed)

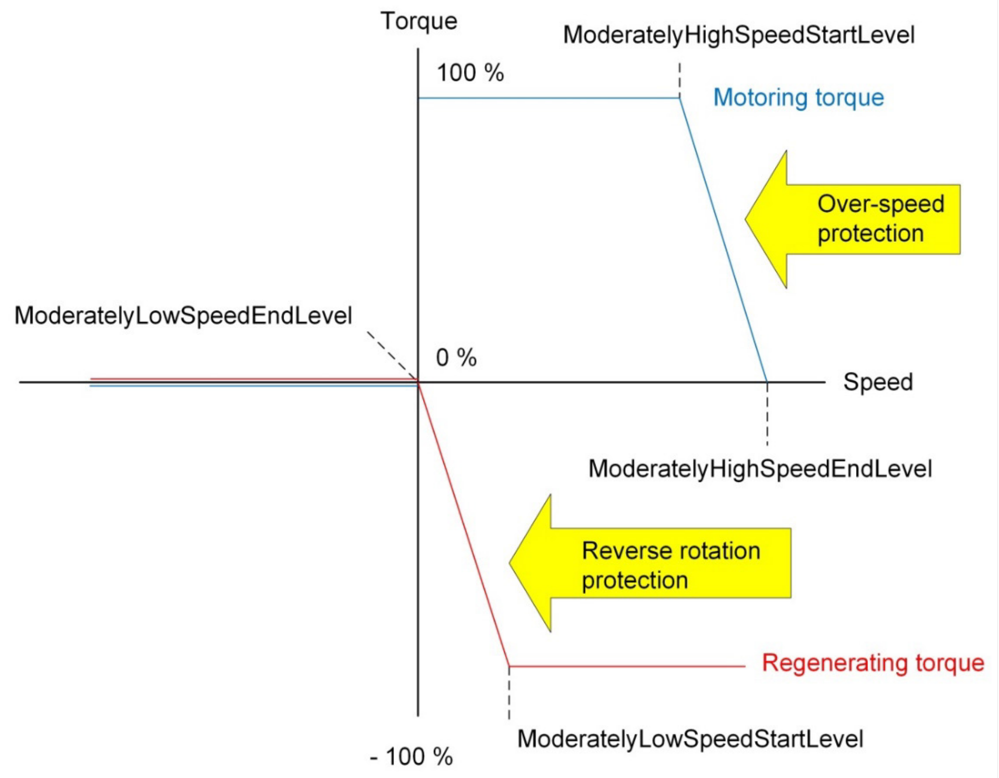

Torque
ModeratelyHighSpeedStartLevel
100 %
Motoring torque
Over-speed protection
ModeratelyLowSpeedEndLevel
0 %
Speed
ModeratelyHighSpeedEndLevel
Reverse rotation protection
Regenerating torque
- 100 %
ModeratelyLowSpeedStartLevel

Figure 31 Motor current limitation due to motor speed (no torques at reverse speed)

Index	Sub-index	Name	Description
0x2095	11	ModeratelyHighSpeedStartLevel	At this speed level linear reduction of current starts towards 0. Set to 0 disable this feature.
0x2095	12	ModeratelyHighSpeedEndLevel	Level for end of reduction current, zero current. Set to 0 to disable this feature
0x2095	13	ModeratlyLowStartSpeed	Linear low speed protection. Braking current starts to be reduced when speed is less than ModeratelyLowStartSpeed and 0 current is reached at ModeratelyLowEndSpeed. Mainly used with combustion engines to prevent rotation of engine in wrong direction. Set to 0 to disable this feature.
0x2095	14	ModeratlyLowEndSpeed	Level for end of reduction current, zero current. Set to 0 to disable this feature

Table 26 Parameters to adjust motor current limitation due to motor speed.

6.3.4 Speed Limit

Value for limitation of commanded speed can be set with MaxCommandSpeed 0x2020:11

6.3.5 Over Speed

Values for over speed limit can be set in forward and reverse directions. When the speed value is higher than the limit set, the application trips. Over speed limits are active for all control modes.

Index	Sub-index	Name	Description
0x2500	01	OverSpeedLimitForward	At this speed level, the application trips.
0x2500	02	OverSpeedLimitReverse	At this speed level, the application trips.

Table 27 Parameters to adjust motor over speed limits

6.3.6 DC Power Limit

The power taken from DC bus or regenerated into the DC bus can be limited by using the objects

- PosDcPowerLimit 0x2094:1
- NegDcPowerLimit 0x2094:2

These parameters are mapped into the CANopen PDO / J1939 command messages, not saved to EEPROM. Function must be activated in ApplicationSetupWord (0x2020:10).

It is not possible to simultaneously use PosDcCurrentLimit and NegDcCurrentLimit, since the above-mentioned DC power objects are calculated from DC current.

6.3.7 Current limits Vs PMW

Depending on output frequency and PWM, maximum current is limited. To reduce noise, PWM can be adjusted, ensure to set max current according figure 32/33

3 PWM level parameters =>

- 0x2074:2 PWMFrequencyAtLowSpeed
- 0x2074:3 PWMFrequencyAtMidSpeed
- 0x2074:4 PWMFrequencyAtHighSpeed.

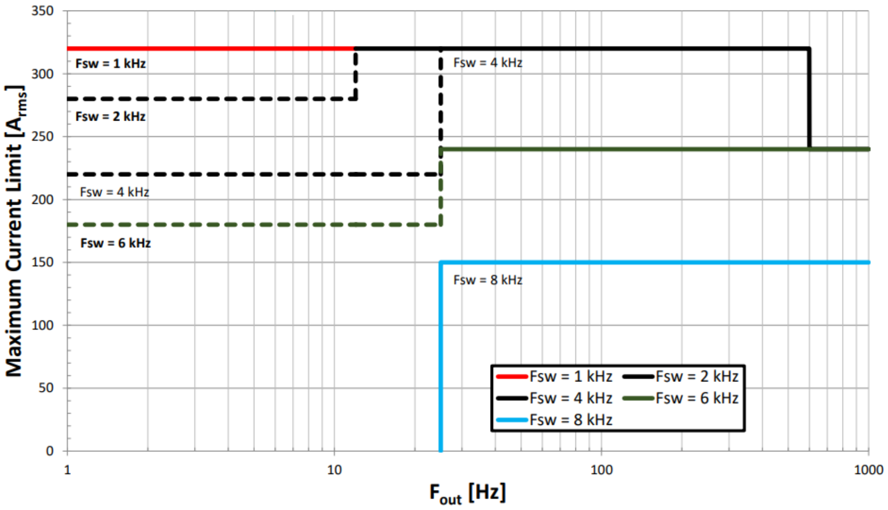


Figure 32 Current limits Vs PWM GVI size G (300A).

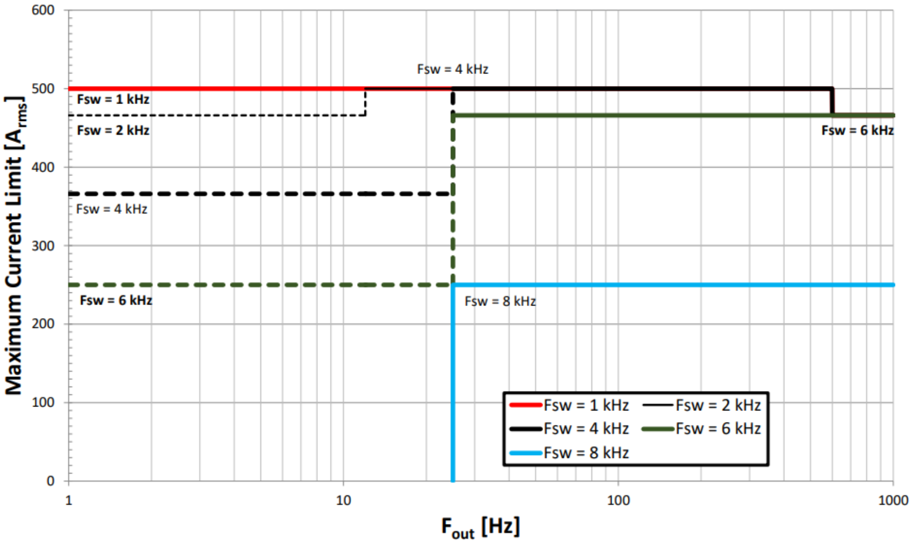


Figure 33 Current limits Vs PWM GVI size H (500A).

7 Open drain output control (LV only)

For the GVI LV version six open drain outputs are available. The output can be either in General Output (On/Off), Voltage Control, Current Control or Open-loop PWM mode. Not every mode is possible with every output. For details, please see the *Product Manual for GVI-C D E* in section 1.1.6. Mode is selected with the parameter 0x2061:1 OutputMode.

It is possible to dedicate an Open Drain output as a Main Contactor output. The dedicated output is then enabled when GVI is switched on (DC bus is charged, bit 1 in Status Word). This setting is done at 0x2060:6.

Coils connected to Open Drain output 1,2 and 3 must be fed by the HIGH_SIDE_OUT. HIGH_SIDE_OUT switch is automatically enabled at startup of the device.

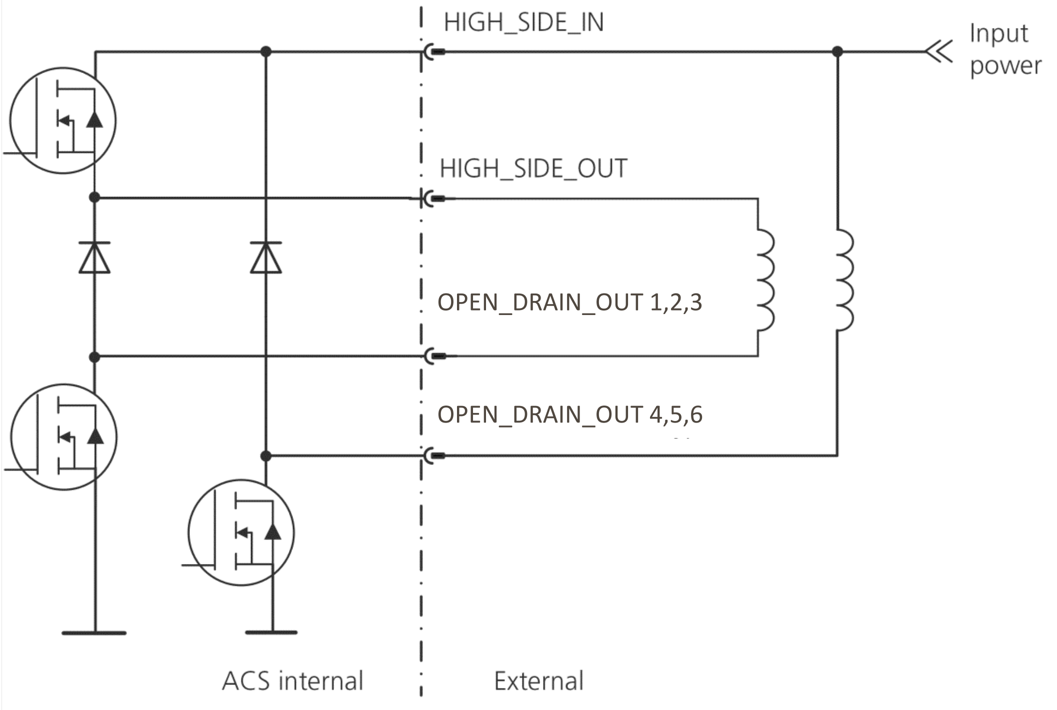


Figure 34 Open Drain output schematic.

On PDO error OPEN_DRAIN 1, 3 and 5 are switched off, the others are not affected by PDO errors.

The main contactor close as soon as "Switch on" bit is 1 and key (pin 3) is on.
Switch on is 1 when charging level is OK so no possibility to control main contactor with switch on in commandall (2000:01 bit 1)

7.1 Voltage Control

The Open Drain Outputs are voltage controlled in both modes and the voltage is defined as the voltage between B+ and the Open Drain channel output, Pin 2 (OD1) and Pin 10 (OD2) in K1. In order to realize the commanded voltage, PWM is used. The switching frequency is by default equal to 1 kHz but can be chosen also as 2, 4, 8 or 16kHz with parameter 2060:1.

The Open Drain Output voltage control is implemented as shown in Figure 35. The Output can be set up for a Pull Voltage (0x2063:1) when the output is enabled and a Hold Voltage (0x2064:1) that follows upon the Pull Voltage. The control of the function runs with 500Hz, which means that the smallest Pull Time (0x2064:1) is 2ms. If the Pull Time is set to zero that part of the output is removed. For example, if the Pull Time is set to zero, no Pull Voltage will be generated.

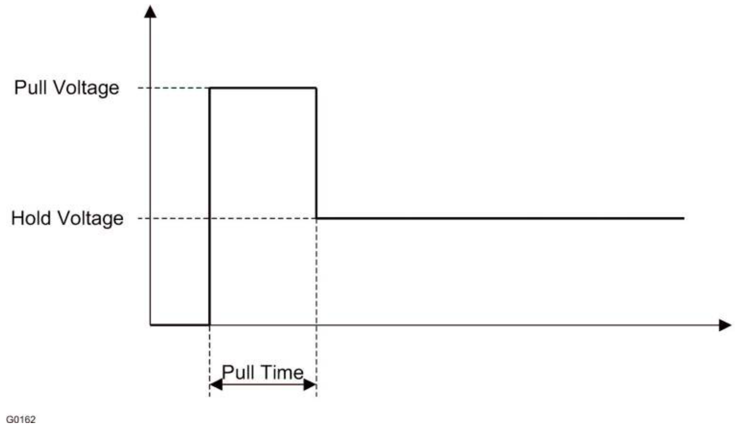


Figure 35 Open drain output functionality

In order to generate an Open Drain Output voltage that is equal to the Dc Bus Voltage, the Pull Time is set to zero and the Hold Voltage is set to a voltage that always is above the Dc Bus Voltage (for example, 100V in a 48V system).

Pull-Hold is used to reduce the strain on components and to save energy. It is needed due to the fact that it takes more power to first energize the coil than it takes to keep it continuously energized.

The voltage is controlled by varying the PWM quota of the open drain output. This also makes it possible to use 24V and 36V coils in a 48V system.

In order for it to work, the DC bus voltage must be above 1/8 of the nominal voltage.

Object NotConnectedLevel (0x206C:1) is active in voltage control, be sure that value is correctly set according to the attached contactor.

7.2 Current Control

The outputs can be current controlled. The benefit of current control is that the electromagnetic force of the coil is independent of coil temperature and battery voltage. By using the PWM of the voltage and current measurement feedback the current can be regulated. Control tuning can be achieved using the P- and I-gain parameters. Remember that in order to work the DC bus voltage must be above 1/8 of the nominal voltage.

7.3 Parameters

The parameters and variables for the Open Drain Output Control are summarized in the following table.

Index	Sub-index	Name	Description
0x2060	1	Switching Frequency	Open Drain Outputs Switching Frequency (1, 2, 4, 8 or 16 kHz). Same frequencies for all Open Drain outputs.
0x2060	6	MainContactor	Sets which Open Drain output to be used as main Contactor. 0 means none, 1 means Open Drain output 1, etc
0x2061	1, 2	Open Drain Output Mode	0 = Output not used, output is disabled. 1 = General output. Open Drain is on/off controlled. 2 = Voltage control 3 = Current control
0x2062	1, 2	Open Drain Output 1, 2 : Pull Time	During this time in ms the Pull Voltage is generated
0x2063	1, 2	Open Drain Output 1, 2 : Pull Voltage	Reference voltage in V between B+ and output when output is enabled
0x2064	1, 2	Open Drain Output 1, 2 : Hold Voltage	Reference voltage in V that follows upon the Pull Voltage
0x2065	1, 2	CurrentReference	Current reference for Open Drain output when in Current control mode
0x2066	1, 2	Pgain	Pgain for Open Drain Output. Used at Current control mode
0x2067	1, 2	Igain	Igain for Open Drain Output. Used at Current control mode.
0x206A	1, 2	Open Drain Output 1, 2: Current	Current measured through open drain channel
0x206C	1,2	NotConnectedLevel	Low Current limit in mA for not connected alarm

Table 28 Open Drain Output Control – Parameters and Variables. The denotation 1, 2 means Open Drain Output 1 resp Open Drain Output 2

8 HVIL Configuration (HV GVI only)

The hazardous voltage interlock (HVIL) is a separate circuit that supervises all access points in the vehicle where high voltage live parts can be exposed and be potentially dangerous for an operator. The GVI requires a 15mA DC current source to be supplied by the main power supply controller in the vehicle.

HVIL may be configured within the application software such that HVIL events leads to device power down or to show only as an active event:

Disabled HVIL Reactions (default!)**:**	**Enabled HVIL Reactions**
HvilApplicationReactionsActual (0x2029:8) = 0	HvilApplicationReactionsActual (0x2029:8) = 1
The HVIL circuit continues to be monitored, and an event logged on HVIL break, however the GVI will NOT take any action should the circuit be broken.	The HVIL circuit within the GVI will expect to see a current source connected and will react should the circuit be broken.
An open circuit HVIL can be monitored on Active Events ID 450,451,452,453 so that the master controller may issue a disable command and disconnect the battery.	An open circuit HVIL can be monitored on Active Events ID 450,451,452,453 and on 460,461,362,463 so that the master controller may issue a disable command and disconnect the battery.
The power stage stays enabled.	The power stages will be disabled.
This mode allows the use of HVIL methods other than a DC current source. If no HVIL loop is present the HVIL events should be disabled.	Only external 15mA DC current source is allowed

Table 30 HVIL reactions

To request a change in reactions taken by the application regarding HVIL, a password must be written to parameter HvilApplicationReactionsRequested (0x2029:7). If the password is correct the value of this parameter will be updated to 0 or 1 depending on if the password sent is the one for activating or disabling the reactions. The actual reaction used is only updated during a restart of the inverter and can be read in HvilApplicationReactionsActual (0x2029:8).

Action	**Password**	**HvilApplicationReactionsRequested**
Activate HvilApplicationReactionsRequested	54321	1
Disable HvilApplicationReactionsRequested	43210	0

Table 31 HVIL Password

WARNING

- When HVIL reactions are disabled, be sure to provide external measures to switch off all high voltage active parts in case of an HVIL loop interrupt.
- IF internal HVIL reaction is enabled, make sure ApplicationSetupWord (0x2020:10) Bit 15 is set to TRUE (green button).

9 Error Handling

The inverter may enter an error state following an event driven shutdown of the power stage (a “trip”), e.g. over current, over voltage, over speed, disconnected speed sensor, CAN message timeout etc. To resume operation, the error must not be active anymore and CommandAll Enable bit needs to be toggled (see GVI CAN Message Database, section 1.1.6).

Some events may attempt to restart the power stage a number of times, e.g. at over current glitches or over voltage protection. If this is not successful, a trip event will eventually occur.

For a complete list of events and their actions, see **OD** Appendix A.

10 Appendix

10.1 Motor Control Method

The motor control method is flux-oriented vector control. Current angle control with gain and offset is used below base speed to optimize mix of d- and q-current.

When full utilization of dc voltage is detected, the controller automatically changes mode to voltage angle control. A specific required reference current (from the speed regulator for instance) will automatically result in a specific optimal stator frequency with full dc voltage.

10.1.1 Current control and SPWM-sym

The dq-plane implemented current controller (so called vector control) controls the magnetization current and the torque producing current independently of each other. The software implemented Current Control is able to compensate for temperature and frequency related changes in motor winding impedance as well as variations in the DC-supply voltage, thereby providing precise control of motor flux and torque over a wide range of operating conditions. The Current Control computes the required motor voltage, which is then realized by SPWM-sym (Sinusoidal Pulse Width Modulation with summarization). The output from the SPWM-sym block is the MOSFET (LV) or IGBT (HV) gate pulses.

10.1.2 Power conversion section

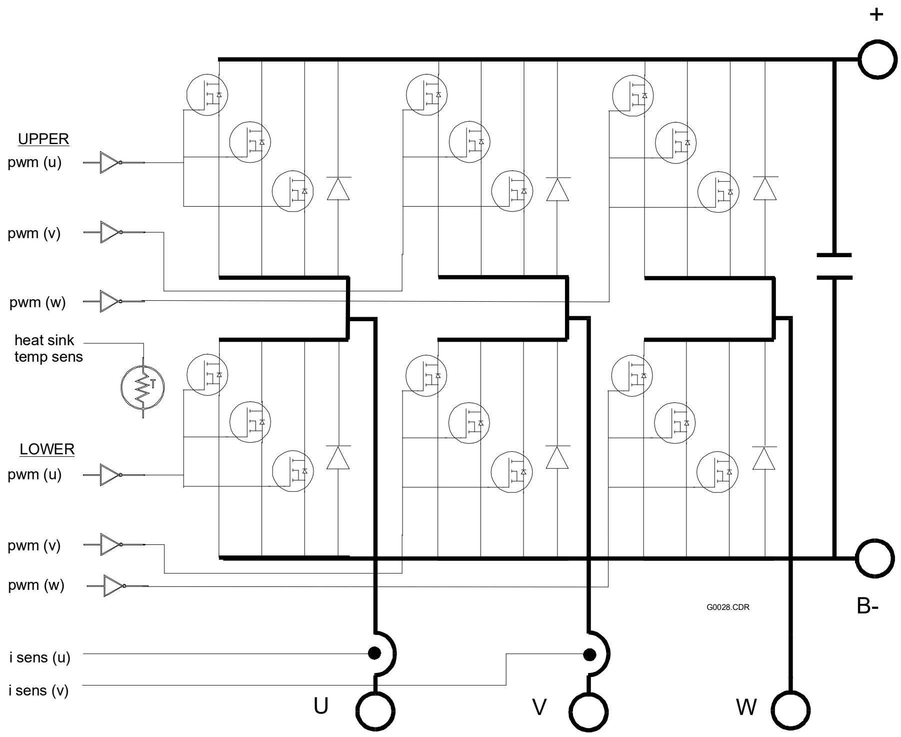


Figure 36 Power Conversion Section, example layout

The principal drive output, a variable frequency, variable amplitude, three phase current, is produced by the Power Conversion section from a DC power source. Figure 36 shows the general circuit configuration. Depending on its current rating, the GVI may employ more or less transistors than illustrated in Figure 36. During braking, regenerated energy from the motor is returned to the battery.

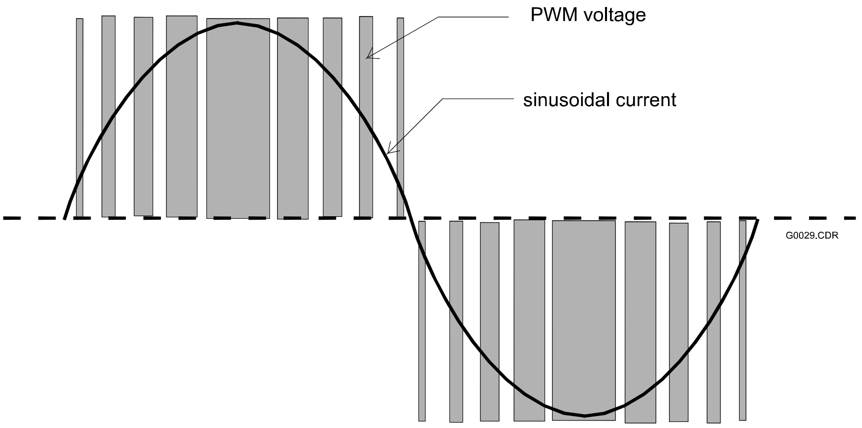


Figure 37 Pulse Width Modulation

All power components are mounted and thermally bonded to a large heat sink that forms one surface of the drive. Heat sink temperature and DC supply voltage are sensed and monitored for control and protection purposes.

Utilizing efficient power transistors, the Power Conversion section amplifies three PWM (Pulse Width Modulated) current commands supplied by the Current Regulator producing three PWM voltages (see Figure 37). These PWM voltage waveforms, when applied to the inductance of the stator, produce currents in the motor, which approximate sine waves.

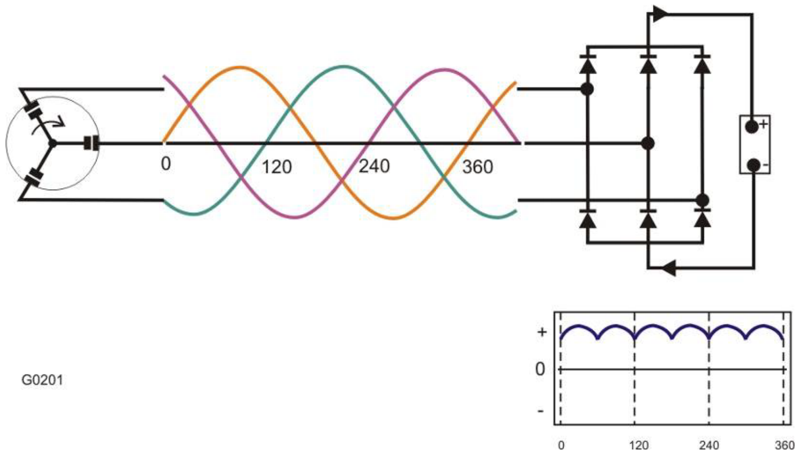


Figure 38 Current Flow during Regeneration

During braking, the rotor runs at higher speed than the speed of the synchronous flux vector and the motor functions as a generator, supplying power to the battery. The change from motor to generator is smooth and instant when rotor speed exceeds synchronous speed. The GVI supplies a magnetizing current at the proper frequency for optimum regeneration performance. The generated power is dependent on the slip (which the GVI controls). The transistors functions as a 6-pulse rectifier (see Figure 38), converting the 3-phase AC current to DC-current that charges the battery.

10.2 Current Angle Tuning

If the motor data for the optimum current angle is not known, the following chapters can be used to identify the Current Angle. For GVM/NX refer to Parker motor technical data.

10.2.1 Offset and Gain Method

The tuning requires measurements at two load points (as a minimum).

The test can be performed in various ways. One of the simplest is to optimize internally estimated power for a given current. The transient recorder can be used to speed up the sweeping of a wide range of current angles via Parker GVI Configuration Tool.

1. Write to the following entries in the drive:
    - RequestedControlMode (0x2020:12) = 3 (AC current mode)
    - CurrentAngleGain (0x2078:4) = 0
    - MaxCurrentAngle (0x2078:5) = 90° (in external scaling)
2. Set up the transient recorder with the following signals:
    1. ReferenceCurrentAngle (0x2078:10)
    2. DcBusPower (0x2073:3)
3. Enable the drive.
4. Use the dyno to spin the shaft to approximately half to 2/3 of base speed. The speed shall be such that UPeakNormalized (0x2079:9) is about 10000.
5. Set CommadAcCurrent (0x2000:4) to first load point (I1).
6. Start transient recorder
7. Sweep CurrentAngleOffset (0x2078:3) slowly around the expected maximum.
8. Plot and look at which ReferenceCurrentAngle the DcBusPower is at maximum. Convert the internal angle value to degrees by multiplying with 0.0055.
9. If a maximum is not found the procedure must be repeated with different range of CurrentAngleOffset or a higher interval on the transient recorder.
10. Set current (0x2000:4) to second load point (I2).
11. Repeat sweeping of CurrentAngleOffset and plotting in Transient Recorder.
12. Disable the drive.
13. Change back RequestedControlMode (0x2020:12) to its original value.

These values should be written to:

- CurrentAngleOffset (0x2078:3) (using external scaling)
- CurrentAngleGain (0x2078:4)

10.2.2 Current Angle Interpolation Tables

Where the above mentioned “Offset and Gain” method approximates the optimal angle, the interpolation table will ensure near optimal current angle for all currents (if tuned correctly).

The arrays will typically contain 8 elements. There is an AC current array and a current angle array. The first element in the AC current array will be 0. The last element will be a current representing the maximum normal AC current for the application in question. The 6 elements in between should be selected to make the interpolated lines follow the optimal torque line as closely as possible. The elements in the current angle array will dictate the current angle for the AC current will the corresponding array index.

NOTE

In theory, the first value in the current angle array should be 0.

Tuning the current angle array is quite straight forward. It is easiest to perform on a software built without current angle interpolation, since we can then use the CurrentAngleOffset parameter.

1. Write to the following entries in the drive:
   - RequestedControlMode (0x2020:12) = 3 (AC current mode)
   - CurrentAngleGain (0x2078:4) = 0
   - MaxCurrentAngle (0x2078:5) = 90° (in external scaling)
2. Set up the transient recorder with the following signals:
   - ReferenceCurrentAngle (0x2078:10)
   - DcBusPower (0x2073:3)
3. Enable the drive.
4. Use the dyno to spin the shaft to approximately half to 2/3 of base speed. The speed shall be such that UPeakNormalized (0x2079:9) is about 10000.

NOTE

At no point during the tuning should the inverter go into field weakening

5. Set CommandAcCurrent (0x2000:4) to first non-zero current point, i.e. corresponding to array index 1.
6. Start the transient recorder
7. Sweep CurrentAngleOffset (0x2078:3) slowly around the expected maximum.

NOTE

Staying too long in high current regions will cause an increase in electric machine temperature, which in turn will cause a drop in torque and consequently also power. Try to be fast enough so that the temperature change does not affect your torque/power reading

8. Plot and look at which ReferenceCurrentAngle the DcBusPower is at maximum. Convert the internal angle value to degrees.

9. If a maximum is not found the procedure must be repeated with a different range of CurrentAngleOffset or a higher interval on the transient recorder.
10. Set current (0x2000:4) to the next current point (I2) and repeat the above procedure while noting the optimal current angles.
11. Disable the drive.
12. Change back RequestedControlMode (0x2020:12) to its original value.

An alternative to the above method, should an external torque measurement exist, is to control the reference angle with GVI Config Tool and plot the torque response with any low latency plotting program available. Quickly sweeping the CurrentAngleOffset, the angle of maximum torque is noted for each current point.

10.3 Alignment of absolute position sensor

An absolute position sensor must have a correct offset in relation to the magnets in order for the motor to run at best efficiency. It is also possible that the sensor may be connected incorrectly leading to no movement.

These instructions will identify incorrect connections and find the correct offset. For this to work the motor must be able to spin freely in both directions. First the motor will spin in an open loop (no sensor used for commutation) where the rotor will follow the current vector. Then the motor will spin closed loop and the back-EMF will be used to find a more accurate offset value.

NOTE

The offset may have multiple solutions depending on ratio between sensor and motor poles. For example, if there are 10 motor pole pairs and 5 sensor pole pairs, there are two solutions to the proper SensorAngleOffset.

10.3.1 Automatic offset measurement with rotation

The drive can attempt to find the offset automatically if the rotor can spin freely in both directions.

1. Start calibration routine by writing 1 to State (0x2051:1).
2. Enable the drive.
3. If the drive goes into error state, check the reason and repeat:
    - If the NoRotation event occurs, check if motor is able to spin without much friction or inertia. If the motor is able to accelerate but too slowly, increase AccAndBrakeCurrent (0x2051:4) or increase AccelerationTimeout (0x2051:5).
    - If the WrongDirection event occurs, check that motor phase cables are connected correctly or the position sensor wiring is correct. If it is, set SensorDirectionInverted (0x2052:18) = 1.
    - If the LowDcBusVoltage event occurs, check that the DC bus voltage is at nominal voltage and that the battery or power supply is able to provide enough current to keep the voltage steady.
    - If the DC Bus - High event occurs, perhaps due to the use of a power supply instead of a battery, set MaxBrakeCurrent (0x2095:2) = 0.

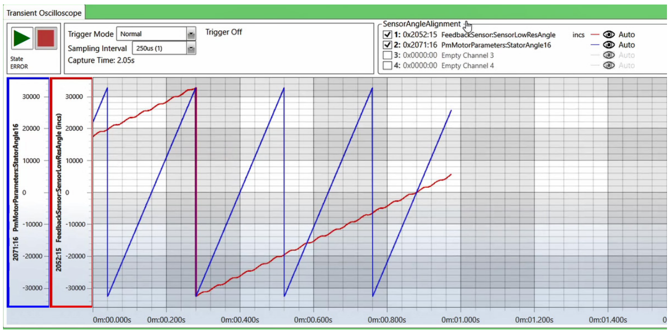
4. If the calibration routine finishes (State reads 254), SensorAngleOffset (0x2052:3) has been updated with the found value. These steps can be repeated to check that results are consistent. A manual fine alignment may be used to verify the results.

10.3.2 Manual rough alignment

If only a fine tuning or a verification of the alignment is needed. Go directly to fine-alignment.

The axis of the motor to be aligned must NOT be connected any device that mechanically loads the shaft.

1. Write to the following entries:
   - RequestedControlMode (0x2020:12) = Speed Mode
   - FeedbackMode (0x2052:10) = Simulated rotor speed
   - ExternalIdRef (0x2078:11) to approximately 50% of max current
   - ExternalIqRef (0x2078:12) = 0
   - ExternalIdRefActive (0x2078:13) = 1
   - ExternalIqRefActive (0x2078:14) = 1
   - CommandAccelerationChange (0x2000:5) = 1
2. Setup the transient recorder with the following signals:
   - SensorAngle (0x2052:15)
   - StatorAngle16 (0x2071:16)
3. Set transient recorder Trig Mode to Normal.
4. Enable the drive.
5. Increase CommandSpeed (0x2000:2) until motor spins with a steady speed (around 100rpm)
6. Trigger the transient recorder and plot the results. The following should be true:
   - The two signals should increase in the same direction. If not, check sensor and phase connections. If they are incorrect, set SensorDirectionInverted (0x2052:18) = 1.
   - The number of stator turns in one sensor turn should agree with the ratio between motor poles and sensor poles
   - The two signals should cross zero at approximately the same time. If not, adjust SensorAngleOffset (0x2052:3).



Trigger Mode: Normal
Sampling Interval: 250us (1)
Capture Time: 2.05s
Trigger Off
State
ERROR

SensorAngleAlignment
☑ 1: 0x2052:15 FeedbackSensor:SensorLowResAngle incs Auto
☑ 2: 0x2071:16 PmMotorParameters:StatorAngle16 Auto
□ 3: 0x0000:00 Empty Channel 3 Auto
□ 4: 0x0000:00 Empty Channel 4 Auto

Figure 39 Manual rough sensor alignment

7. Disable the drive.

8. Change back the temporary values and then save to EEPROM:
    - RequestedControlMode (0x2020:12)
    - FeedbackMode (0x2052:10) = 0
    - ExternalIqRefActive (0x2078:14) = 0
    - ExternalIdRefActive (0x2078:13) = 0

10.3.3 Manual fine-alignment with shaft un-coupled

1. Set RequestedControlMode (0x2020:12) = Speed Control.
2. Enable the drive.
3. Spin the motor forwards with CommandSpeed (0x2000:2). The speed shall be such that UPeakNormalized (0x2079:9) is about 10000.
4. Note the speed, UqRefFiltered (2079:11) and UdRefFiltered (0x2079:12).
5. Spin the motor backwards with the same speed. Note the speed, UqReference and UdReference.
6. Imagine Ud and Uq as voltage vectors in the DQ-plane as shown in figure 40 below. The voltage vectors shall be symmetrical around the D-axis. If not, SensorAngleOffset (0x2052:3) must be adjusted.

NOTE

Calculate the average value of Ud forwards and backwards. Tune until you get this average value.

7. Repeat until a correct alignment has been achieved.
8. Disable the drive.
9. Change back RequestedControlMode (0x2020:12).

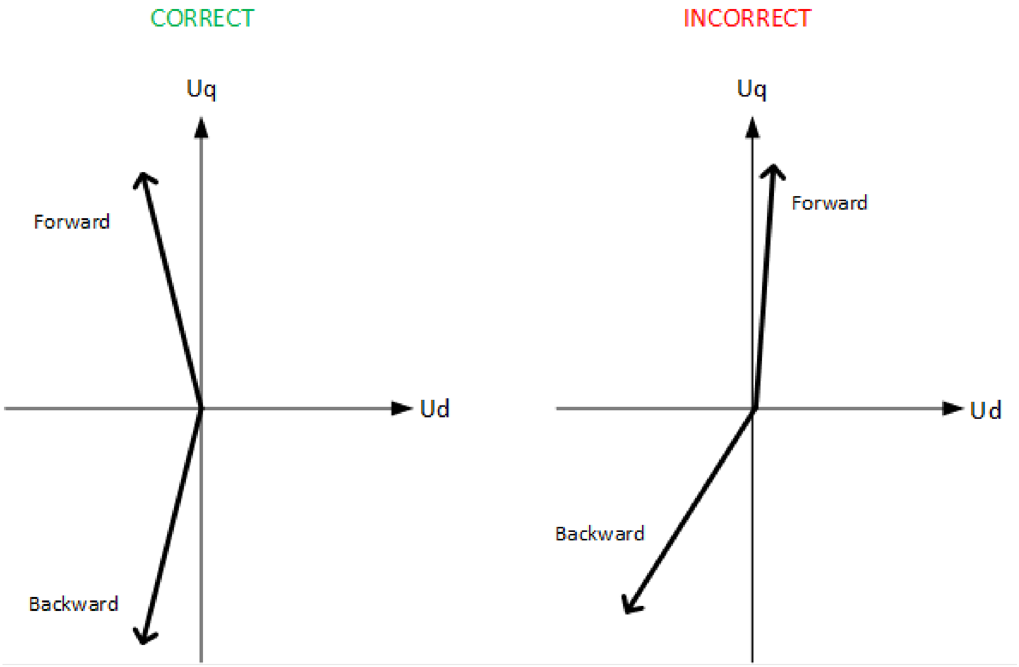


Figure 40 Examples of correct and incorrect alignment

Examples of good values:

Speed	Uq	Ud	VoltageAngle
2000	10000	-200	1.2
-2000	-10000	-200	1.2

Table 32 Examples of correct alignment

10.3.4 Manual fine-alignment with shaft coupled

Use these instructions if the motor is coupled to a diesel engine for example.

1. Set drive in AC control mode (0x2020:12 = 3).
2. Enable the drive.
3. Spin the other motor, preferably to at least half of base speed.
4. Set CommandAcCurrent (0x2000:4) to some positive value.
5. Read ActualVoltageAngle (0x2079:14).
6. Change sign on CommandAcCurrent but keep the same amplitude.
7. Read ActualVoltageAngle (0x2079:14). It should be about the same as before but with different sign. If not, adjust SensorAngleOffset (0x2052:3) and repeat.
8. Change back RequestedControlMode (0x2020:12).

NOTE

ActualVoltageAngle can be filtered more by increasing ActualVoltageAngleFilterShiftSteps (0x2079:17)

10.3.5 PLL filtering stator angle

The PLL filter serves the purpose of smoothing out a bad feedback signals without introducing delay in the signals. This can be adjusted in case of very bad and perturbed signal.

PLL Pgain 0x2071:18 is set to 640 by default, increase to 1280 in case of bad signal quality.

PLL Igain 0x2071:19 is set to 32 by default, increase to 64 in case of bad signal quality.

Make new test (use transient scope recorder).

10.3.6 Absolute position sensor sample time delay

Some position sensors (sine/cosine sensors, resolvers) have a slight delay in the signal sampling. The delay primarily causes a speed dependency in the torque estimation, typically with a clear breakpoint at some certain speed where the torque suddenly changes. If the control is optimized, the. Torque will typically drop. To adjust for this, the parameter SinCosSensor.SampleTimeDelay (0x2053:41) can be tuned using the following steps. Preferably, this should be one of the last steps of the overall tuning.

1. Enable the BRAKE drive and pick a point around 50% of base speed.
2. Enable the DUT in torque or current mode.
3. Apply a fairly high torque, enough to get good signal-to-noise ratio.
4. Ramp speed towards the maximum speed of the application
5. If there's a sudden shift in the torque estimation, increase SampleTimeDelay to see if it makes a difference.

NOTE

A typical value seen for most resolvers is around 9500 ns.

10.4 Motor data

The motor data is usually used to calculate rotor flux and torque. The auto-tuning will find values for resistance and inductance that will work in most cases. For a PM motor, the rotor flux must be determined manually. If a more accurate torque estimation is needed, R and L can be determined using external torque measurement.

Refer to GVI-Motor-Data application note.

10.4.1 Nominal rotor flux

The nominal rotor flux value is used to be able to enable the drive at high speed, as a feed-forward to the current controller and to calculate torque.

In order to determine rotor flux in a test bench, the motor can be run at speed and with little or no current.

1. Set RequestedControlMode (0x2020:12) = Speed Mode.
2. Enable the drive.
3. Increase CommandSpeed (0x2000:2) until UPeakNormalized (0x2079:9) is about 5000 - 10000.
4. Read RotorFluxCalc (0x2073:9) (in external scaling).

5. Disable the drive.
6. Change back RequestedControlMode (0x2020:12) to its original value.

Update the following values:

- NominalRotorFlux (0x2076:3) (in external scaling) to same value as RotorFluxCalc.

10.4.2 Motor resistance

The resistance is used to calculate the rotor flux ψr, which in turn is used for the torque calculation. These instructions are only applicable if a high accuracy torque estimation is needed, otherwise the resistance found during auto-tuning should be sufficiently accurate.

An external torque measurement is needed for these instructions.

1. Set RequestedControlMode (0x2020:12) = 3 (AC current control mode).
2. Set CommandAcCurrent (0x2000:4) = 0.
3. Set ExternalIqRefActive (0x2078:14) = 1.
4. Enable the drive.
5. Use the dyno to spin the motor to around half of base speed.
6. Set ExternalIqRef (0x2078:12) = 50% of max current.
7. Adjust Rmotor (0x2073:6) until ActTorque (0x2076:2) corresponds to the externally measured torque.
8. Disable the drive.
9. Change back ExternalIqRefActive (0x2078:14) to 0.

Update the following values:

- Rmotor (0x2073:6)

10.4.3 Inductances Ld and Lq

The inductances are used to de-couple the current control and in the torque calculation.

1. Set RequestedControlMode (0x2020:12) = AC current control mode.
2. Set ExternalIqRefActive (0x2078:14) = 1 and ExternalIdRefActive (0x2078:13) = 1.
3. Set ExternalIqRef (0x2078:12) = 0 and ExternalIdRef (0x2078:11) = 0.
4. Enable the drive.
5. Use the dyno to spin the motor so that UPeakNormalized (0x2079:9) is about 10000.
6. Write down the value of DeltaStatorAngle16 (0x2071:17).
7. Set ExternalIqRef (0x2078:12) = 50% of max current. Write this value down.
8. Write down the value of UdRefFiltered (0x2079:12).
9. Set ExternalIqRef (0x2078:12) = 0.
10. Write down the value of RotorFluxCalc (0x2073:9).
11. Set ExternalIdRef (0x2078:12) = negative 25% of max current. Write this value down.

12. Write down the value of UqRefFiltered (0x2079:12).
13. Set ExternalIdRef (0x2078:12) = 0.
14. Disable the drive.
15. Change back ExternalIqRefActive (0x2078:14) and ExternalIdRefActive (0x2078:13) to 0.

Use the following equations to calculate Lq and Ld. All values should be in SI units so make sure to take the factor into consideration when reading values from CAN!

Unom = NominalVoltage (0x2030:15)

Uq = UqRefFiltered / 16384 * Unom /√3

Ud = UdRefFiltered / 16384 * Unom /√3

ω = DeltaStatorAngle16 * 2 * π * 4000 / 65536

Iq = ExternalIqRef * √2

Id = ExternalIdRef * √2

ψr = RotorFluxCalc / 1000

$$Lq = -\frac{Ud}{\omega * Iq}$$

$$Ld = \frac{\frac{Uq}{\omega} - \psi\mathrm{r}}{Id}$$

Update the following values:

- Ld (0x2073:7)
- Lq (0x2073:8)

10.5 DQ Flux Table Tuning

For PARKER MOTORS, refer to application note (see 1.1.6)

Since DQFlux tables is only 8 points in relation to the current the look-up for the CurrentAngle need to be set before tuning DQFlux. The look-up is either based on CurrentAngleOffset and CurrentAngleGain or CurrentAngleTable depending on choosen method. If the look-up for the CurrentAngle is changed also DQFlux table need to be updated.

DQFlux Calculation is based on following equations.

- ActualDFlux = (UqRef - R*iq) / ω_{el}
- ActualQFlux = -(UdRef - R*id) / ω_{el}
- UdRef = R*Id - ω_{el}*QFlux = R*Id - ω_{el}*Lq*Iq
- UqRef = R*Iq + ω_{el}*DFlux = R*Iq + ω_{el}*Ld*Id + ω_{el}* MagnetFlux

If nominal R is wrong, the ActualDFlux will not be correct.

1. First tune the MinDeltaStatorForFluxCalc
    - run in speed model

- set MinDeltaStatorForFluxCalc to a really low value so that the tables for DQflux is not used.
- Do
    - Either check at which speed (actually DeltaStatorAngle16) to note when the ActualDQflux is stable
    - Or Check when at which speed gives good enough DQflux estimation for the torque accuracy to be within tolerance.

2. The goal should be to tune MinDeltaStatorAngleForFluxCalc as low as possible since when the table is used at low speed the flux is not compensated by temperature changes as it is when above MinDeltaStatorAngleForFluxCalc.
3. Run in current mode with the load motor controlling it at a speed slightly higher than the speed which was found to give stable Fluxcalculations and make sure that DeltaStatorAngle16 is strictly higher than MinDeltaStatorAngleForFluxCalc, this can be done by temporarily setting MinDeltaStatorAngleForFluxCalc to 0.
4. Go through all current points (8 points) according to DFluxInterpolationInput and QFluxInterpolationInput accordingly.
5. For every current point in the table take a note of the ActualDFlux and ActualQFlux and input these values at the correct place in DFluxInterpolationOutput and QFluxInterpolationOutput
6. Reset the MinDeltaStatorAngleForFluxCalc to the found value.

10.6 CANopen Operation example

In the example below one CAN node Master, node ID 1, controls three CAN slave nodes, node ID´s 3, 4 and 5, see Figure 41. As can be seen each slave node uses two receive PDO´s and two transmit PDO´s. For PDO1, each COB-ID is given by 0x180 + Node ID for transmit PDO and 0x200 + Node ID for receive PDO. For PDO2 the base is instead 0x280 + Node ID for receive and 0x300 + Node ID for transmit.

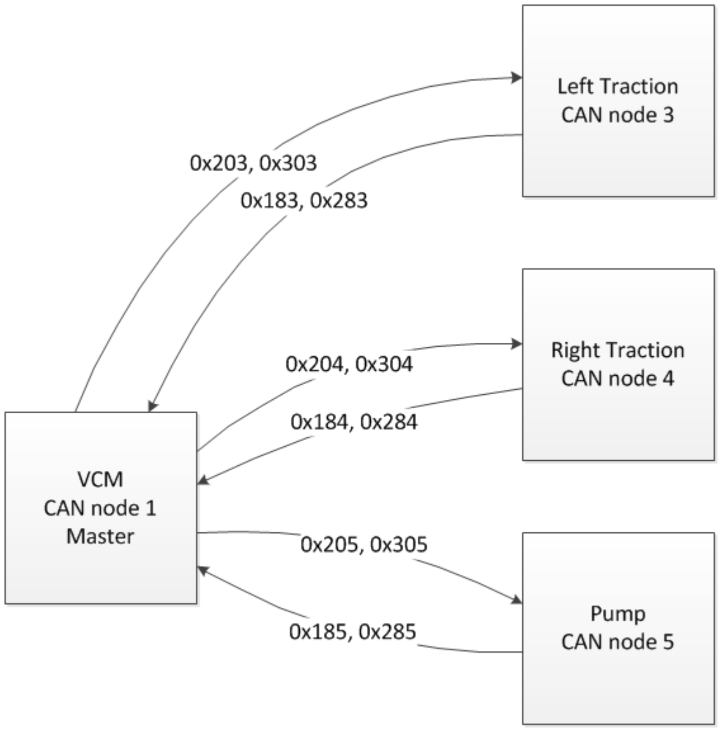


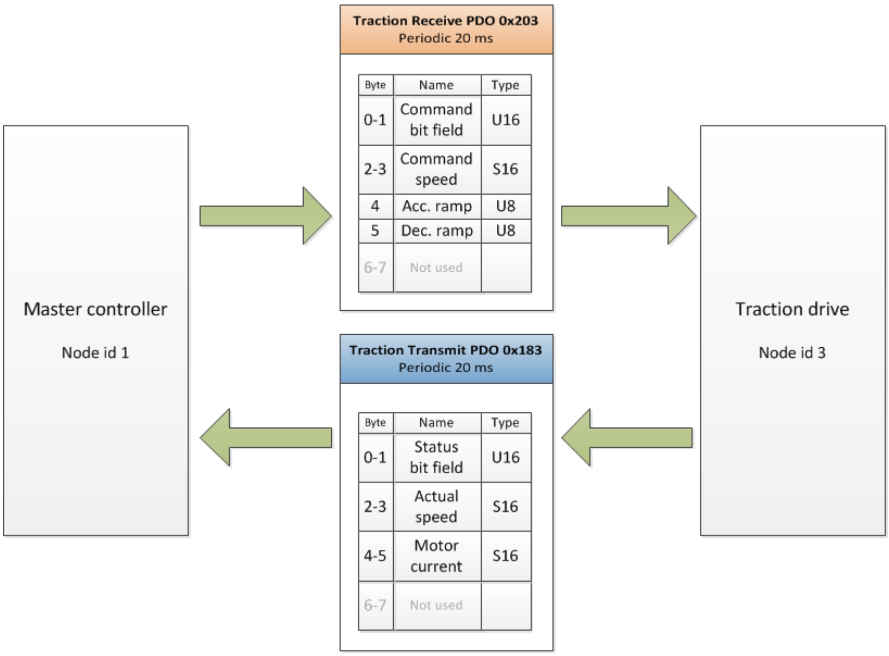
Figure 41 Example CAN system, one Master is exchanging CAN communication with three CAN slaves

Communication object	COB-ID(s) hex	Slave nodes
NMT commands	000	Receive only
Sync	080	Receive only
Emergency	080 + NodeID	Transmit
PDO	180 + NodeID	transmit PDO 1
	200 + NodeID	receive PDO 1
	280 + NodeID	transmit PDO 2
	300 + NodeID	receive PDO 2
	380 + NodeID	transmit PDO 3
	400 + NodeID	receive PDO 3
	480 + NodeID	transmit PDO 4
	500 + NodeID	receive PDO 4
SDO	580 + NodeID	Transmit
	600 + NodeID	Receive
NMT node monitoring, heartbeat	700 + NodeID	Transmit

Table 33 CAN Object description

The sequence to initiate the CANopen communication and to enable the operation of the GVI is described in the following:

1. KEY On, hence apply KEY start voltage by connecting B+ to input pin KEY.
2. *Hint: NMT status can be observed at parameter location 0x5F04:8 CanOpenState.*
3. Time from KEY On to NMT state Pre-Operational is less than 500ms.
4. Use the SDO in order to configure GVI parameters or Parker GVI Configuration Tool.



Traction Receive PDO 0x203
Periodic 20 ms

Byte	Name	Type
0-1	Command bit field	U16
2-3	Command speed	S16
4	Acc. ramp	U8
5	Dec. ramp	U8
6-7	Not used	

Traction Transmit PDO 0x183
Periodic 20 ms

Byte	Name	Type
0-1	Status bit field	U16
2-3	Actual speed	S16
4-5	Motor current	S16
6-7	Not used	

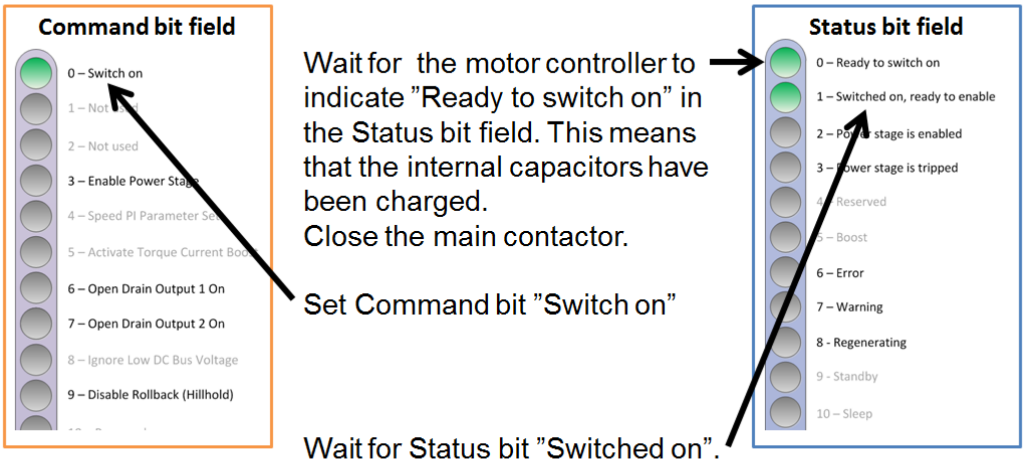
Figure 42 PDO communication between Master (Node 1) and (Traction Node 3) has started. Node 3 is used as an example.

5. Send the NMT message Start Remote Node to the GVI. The CANopen state of the GVI will become OPERATIONAL.
   Start up the PDO objects, i.e. start sending the PDO to the GVI (PDO1Rx) and start receiving the PDO from the GVI (PDO1Tx). If the SYNC object is used then also start to send that. The transmit period of PDO1Rx and SYNC, if used, shall be between 8 to 20 ms. The transmit period of the PDO1Tx in the GVI depends on the Transmission Type configured in the software. Standard setup are that PDO from GVI are transmitted upon receive of PDO from vehicle master controller.

   If a second, third or fourth PDO are used, i.e. PDO2, 3 or 4Rx and/or PDO2, 3 or 4Tx, this service shall also be started.

   Note: Once the CANopen state of the GVI becomes OPERATIONAL (via the NMT message Start Remote Node), PDO communication must start within one second. Once the first PDORx is received, subsequent PDORx's must be received at an interval not longer than specified by the PDO Timeout parameters (Index 5F04h:2-5). The SYNC object, if used, must occur at an interval not longer than specified by the Sync Timeout parameter (Index 5F04h:6).

   It is assumed that Requested Control Mode is set to Speed Mode (= 0), see section 0 and Speed Ramp Mode is set to Speed PDO Ramp Mode (= 4), see more in section 6.2.1.4.


Command bit field
0 – Switch on
1 – Not used
2 – Not used
3 – Enable Power Stage
4 – Speed PI Parameter Set
5 – Activate Torque Current Boost
6 – Open Drain Output 1 On
7 – Open Drain Output 2 On
8 – Ignore Low DC Bus Voltage
9 – Disable Rollback (Hillhold)
Status bit field
0 – Ready to switch on
1 – Switched on, ready to enable
2 – Power stage is enabled
3 – Power stage is tripped
4 – Reserved
5 – Boost
6 – Error
7 – Warning
8 - Regenerating
9 - Standby
10 – Sleep

Wait for the motor controller to indicate "Ready to switch on" in the Status bit field. This means that the internal capacitors have been charged.
Close the main contactor.

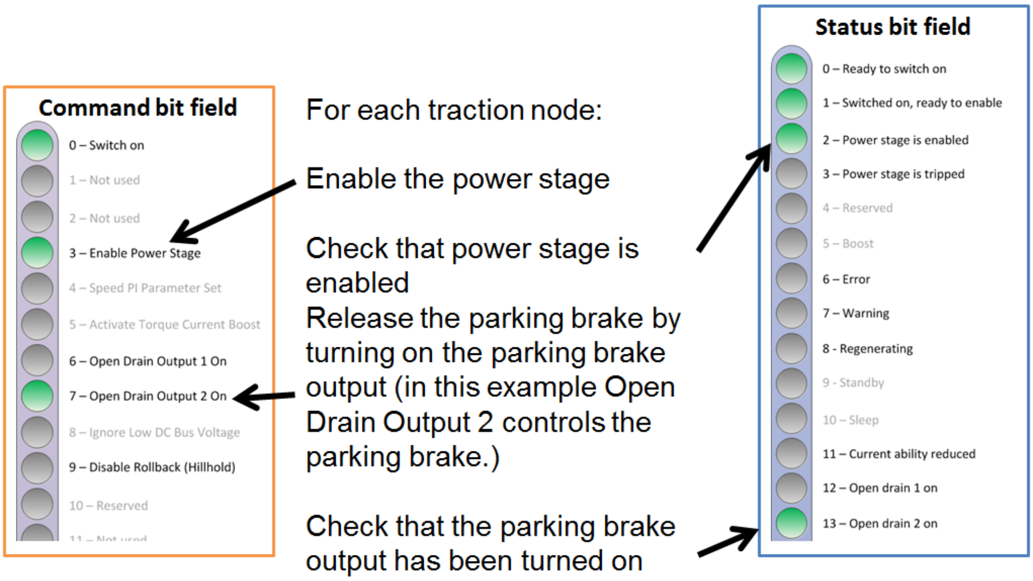
Set Command bit "Switch on"

Wait for Status bit "Switched on".

Figure 43 Switch on procedure

6. Wait until the bit Ready To Switch On (Bit 0) is set in the Status Word, i.e. the Dc Bus is charged. Then the bit Switch On (Bit 0) in the Command Word shall be set. The GVI confirms this by setting the bit Switched On (Bit 1) in the Status Word.

   This step (6) is appropriate if main-contactor is integrated with the GVI, for instance predefined to one of the GVI Open Drain outputs. If main-contactor is instead connected and controlled by the master controller the charging level and when to close the main-contactor has to be handled from the master controller and switch on procedure for the GVI has no effect.


Command bit field
0 – Switch on
1 – Not used
2 – Not used
3 – Enable Power Stage
4 – Speed PI Parameter Set
5 – Activate Torque Current Boost
6 – Open Drain Output 1 On
7 – Open Drain Output 2 On
8 – Ignore Low DC Bus Voltage
9 – Disable Rollback (Hillhold)
10 – Reserved
11 – Not used
Status bit field
0 – Ready to switch on
1 – Switched on, ready to enable
2 – Power stage is enabled
3 – Power stage is tripped
4 – Reserved
5 – Boost
6 – Error
7 – Warning
8 - Regenerating
9 - Standby
10 – Sleep
11 – Current ability reduced
12 – Open drain 1 on
13 – Open drain 2 on

For each traction node:

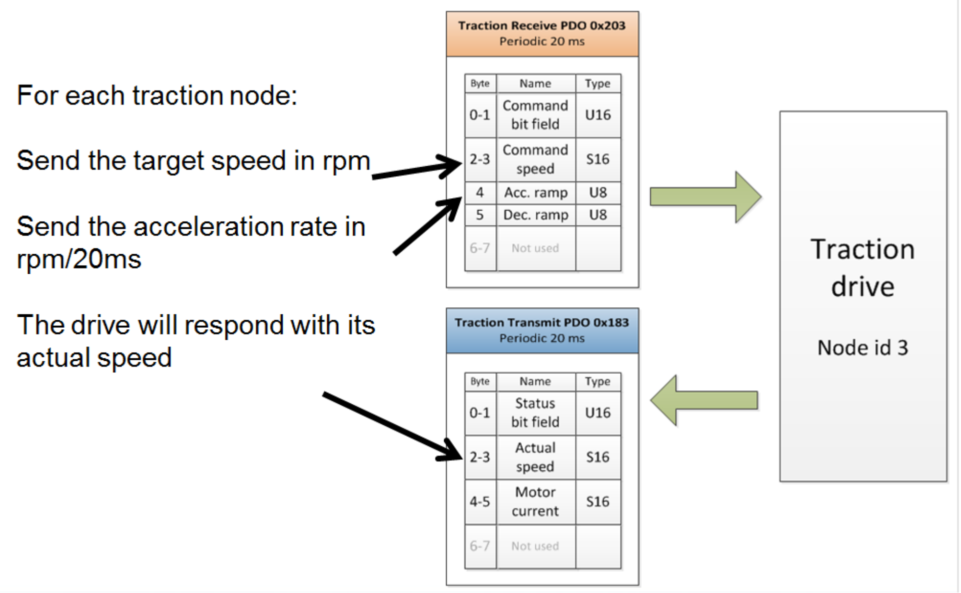
Enable the power stage

Check that power stage is enabled
Release the parking brake by turning on the parking brake output (in this example Open Drain Output 2 controls the parking brake.)

Check that the parking brake output has been turned on

Figure 44 Enabling of power stage and (if used) releasing the parking brake

7. After the bit Switch On is set or other means to secure main contactor being closed; it is possible to enable the GVI by setting the bit Enable Operation (Bit 3) in the Command Word. The GVI confirms this by setting the bit Operation Enabled (Bit 2) in the Status Word.
   The GVI can be disabled at any time by clearing the bit Enable Operation in the Command Word. The GVI confirms this by resetting the Operation Enabled bit in the Status Word.



For each traction node:

Send the target speed in rpm

Send the acceleration rate in rpm/20ms

The drive will respond with its actual speed

Traction Receive PDO 0x203
Periodic 20 ms

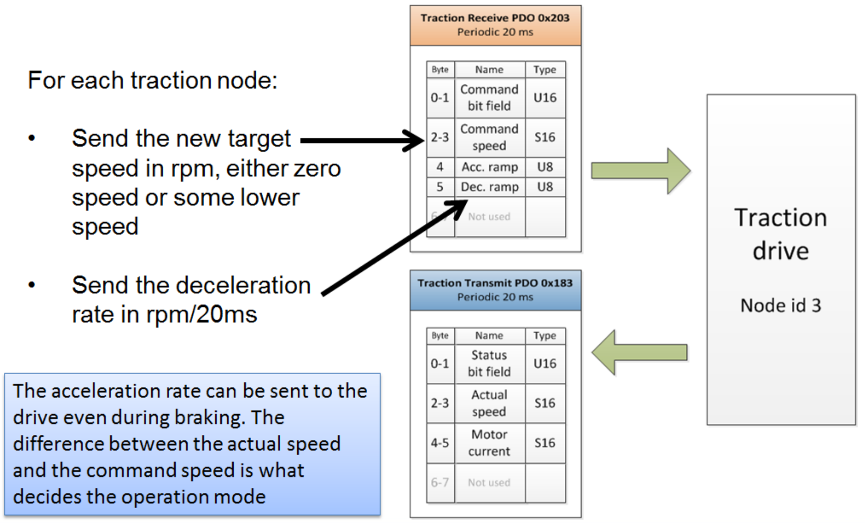
Byte	Name	Type
0-1	Command bit field	U16
2-3	Command speed	S16
4	Acc. ramp	U8
5	Dec. ramp	U8
6-7	Not used	

Traction Transmit PDO 0x183
Periodic 20 ms

Byte	Name	Type
0-1	Status bit field	U16
2-3	Actual speed	S16
4-5	Motor current	S16
6-7	Not used	

Figure 45 Speed reference command, accelerating

8. After enabling of power stage has been verified a Speed Command can now be set in PDO received by CAN slave.



For each traction node:

- Send the new target speed in rpm, either zero speed or some lower speed
- Send the deceleration rate in rpm/20ms

The acceleration rate can be sent to the drive even during braking. The difference between the actual speed and the command speed is what decides the operation mode

Traction Receive PDO 0x203
Periodic 20 ms

Byte	Name	Type
0-1	Command bit field	U16
2-3	Command speed	S16
4	Acc. ramp	U8
5	Dec. ramp	U8
6-7	Not used	

Traction Transmit PDO 0x183
Periodic 20 ms

Byte	Name	Type
0-1	Status bit field	U16
2-3	Actual speed	S16
4-5	Motor current	S16
6-7	Not used	

Figure 46 Speed reference command, braking

9. In order to reduce the vehicle the speed command has to be altered to zero speed or some lower speed.
   For switch back braking, or plugging, both Dec rate and Acc rate are used. Initially the decal rate down to zero speed and then accel rate for accelerating to speed in the opposite direction.
   For the Pump node the procedure is the same but here there is no parking brake to consider before setting the speed command. Instead there might be some hydraulic valves to activate in order to accomplish successful operation.

Parker Worldwide

Europe, Middle East, Africa

AE – United Arab Emirates,
Dubai
Tel: +971 4 8127100
parker.me@parker.com

AT – Austria, St. Florian
Tel: +43 (0)7224 66201
parker.austria@parker.com

AZ – Azerbaijan, Baku
Tel: +994 50 2233 458
parker.azerbaijan@parker.com

BE/NL/LU – Benelux,
Hendrik Ido Ambacht
Tel: +31 (0)541 585 000
parker.nl@parker.com

BG – Bulgaria, Sofia
Tel: +359 2 980 1344
parker.bulgaria@parker.com

BY – Belarus, Minsk
Tel: +48 (0)22 573 24 00
parker.poland@parker.com

CH – Switzerland, Etoy
Tel: +41 (0)21 821 87 00
parker.switzerland@parker.com

CZ – Czech Republic, Klecany
Tel: +420 284 083 111
parker.czechrepublic@parker.com

DE – Germany, Kaarst
Tel: +49 (0)2131 4016 0
parker.germany@parker.com

DK – Denmark, Ballerup
Tel: +45 43 56 04 00
parker.denmark@parker.com

ES – Spain, Madrid
Tel: +34 902 330 001
parker.spain@parker.com

FI – Finland, Vantaa
Tel: +358 (0)20 753 2500
parker.finland@parker.com

FR – France, Contamine s/Arve
Tel: +33 (0)4 50 25 80 25
parker.france@parker.com

GR – Greece, Piraeus
Tel: +30 210 933 6450
parker.greece@parker.com

HU – Hungary, Budaörs
Tel: +36 23 885 470
parker.hungary@parker.com

IE – Ireland, Dublin
Tel: +353 (0)1 466 6370
parker.ireland@parker.com

IL – Israel
Tel: +39 02 45 19 21
parker.israel@parker.com

IT – Italy, Corsico (MI)
Tel: +39 02 45 19 21
parker.italy@parker.com

KZ – Kazakhstan, Almaty
Tel: +7 7273 561 000
parker.easteurope@parker.com

NO – Norway, Asker
Tel: +47 66 75 34 00
parker.norway@parker.com

PL – Poland, Warsaw
Tel: +48 (0)22 573 24 00
parker.poland@parker.com

PT – Portugal
Tel: +351 22 999 7360
parker.portugal@parker.com

RO – Romania, Bucharest
Tel: +40 21 252 1382
parker.romania@parker.com

RU – Russia, Moscow
Tel: +7 495 645-2156
parker.russia@parker.com

SE – Sweden, Borås
Tel: +46 (0)8 59 79 50 00
parker.sweden@parker.com

SK – Slovakia, Banská Bystrica
Tel: +421 484 162 252
parker.slovakia@parker.com

SL – Slovenia, Novo Mesto
Tel: +386 7 337 6650
parker.slovenia@parker.com

TR – Turkey, Istanbul
Tel: +90 216 4997081
parker.turkey@parker.com

UA – Ukraine, Kiev
Tel: +48 (0)22 573 24 00
parker.poland@parker.com

UK – United Kingdom, Warwick
Tel: +44 (0)1926 317 878
parker.uk@parker.com

ZA – South Africa, Kempton Park
Tel: +27 (0)11 961 0700
parker.southafrica@parker.com

North America

CA – Canada, Milton, Ontario
Tel: +1 905 693 3000

US – USA, Cleveland
Tel: +1 216 896 3000

Asia Pacific

AU – Australia, Castle Hill
Tel: +61 (0)2-9634 7777

CN – China, Shanghai
Tel: +86 21 2899 5000

HK – Hong Kong
Tel: +852 2428 8008

IN – India, Mumbai
Tel: +91 22 6513 7081-85

JP – Japan, Tokyo
Tel: +81 (0)3 6408 3901

KR – South Korea, Seoul
Tel: +82 2 559 0400

MY – Malaysia, Shah Alam
Tel: +60 3 7849 0800

NZ – New Zealand, Mt Wellington
Tel: +64 9 574 1744

SG – Singapore
Tel: +65 6887 6300

TH – Thailand, Bangkok
Tel: +662 186 7000

TW – Taiwan, Taipei
Tel: +886 2 2298 8987

South America

AR – Argentina, Buenos Aires
Tel: +54 3327 44 4129

BR – Brazil, Sao Jose dos Campos
Tel: +55 800 727 5374

CL – Chile, Santiago
Tel: +56 2 623 1216

MX – Mexico, Toluca
Tel: +52 72 2275 4200

EMEA Product Information Centre
Free phone: 00 800 27 27 5374
(from AT, BE, CH, CZ, DE, DK, EE, ES, FI, FR, IE, IL,
IS, IT, LU, MT, NL, NO, PL, PT, RU, SE, SK, UK, ZA)

US Product Information Centre
Toll-free number: 1-800-27 27 537

www.parker.com

Your local authorized Parker distributor