Reference Manual

TMC9660 Parameter Mode

TMC9660 Parameter Mode Reference Manual

GENERAL DESCRIPTION

The TMC9660 is a highly integrated monolithic gate driver and motor controller IC with buck converter. It includes a smart gate driver, a high-performance motion controller with hardware-based field-oriented control (FOC) and servo controller (velocity, position, ramp generator), motor position feedback interfaces (2xA/B/N encoder, HALL, SPI), an analog signal processing block for bottom shunt current measurement (programmable current-sense amplifiers [CSAs] and analog-to-digital converters [ADCs]).

The TMC9660 supports two modes of operation: one mode to directly access the hardware registers, and the enhanced and simplified parameter mode. This document is the reference manual for the TMC9660 Parameter Mode. All the necessary information needed to configure and operate the device in parameter mode can be found in this document. For general information on the IC, refer to the main TMC9660 data sheet.

Related Documents

- TMC9660 Data Sheet

APPLICATIONS

- Robotics
- Power Tools
- Gardening
- Automated Guided Vehicles (AGV)/Warehouse Automation
- Pump (e.g., Peristaltic)
- Industrial 3D Printing
- Factory Automation
- Desktop Manufacturing
- E-Bike/Light Electric Vehicles or LEV

TMC9660 FEATURES

- Three-Phase Permanent Magnet Synchronous Motors (PMSM)/Brushless DC (BLDC), Two-Phase Stepper Motor, and Brushed DC Motor Support
- 7.7V to 70V Single-Supply Operating Voltage Range
- Smart Gate Driver with Adjustable Strength up to 1A/2A Source/Sink
- Field-Oriented Controller/FOC in Hardware for Wide Bandwidth Current Control Loop
- Position, Velocity, and Torque Controller in Hardware for Fast and Precise Control
- 8-Point Ramp Generator with Ramp Calculation in Real Time in Hardware
- Fast Space Vector Pulse Width Modulation (SVPWM) Engine (2kHz …100kHz) with 120MHz Clock
- Feedback Position Sensor Support (Hall, 2xA/B/N, SPI)
- Bottom Shunt Current Measurement (Programmable CSA and ADCs)
- Charge Pump with Voltage Doubler
- Trickle Charge Pump for 100% PWM Duty Cycle

PARAMETER MODE FEATURES

- SPI, UART Interfaces for Communication with Main/Application Controller
- SPI, I2C Interfaces for Flash/EEPROM for Parameter Storage
- Watchdog with Separate Internal Oscillator
- Low-Power Hibernation Mode with Wake-Up Button and Timer Support
- IIT Motor Overload Protection
- Automatic Gate Driver Startup
- Advanced Gate Driver Fault Detection and Fault Handling

One Analog Way, Wilmington, MA 01887-2356, U.S.A. | Tel: 781.329.4700

- Support for Electromechanical Brakes
- Brake Chopper Overvoltage Protection
- Advance Script Execution Capabilities for Standalone Operation
- Integrated Preprogrammed 32-Bit/40MHz Microcontroller Supporting Initial Configuration (OTP) of the Device

SIMPLIFIED BLOCK DIAGRAM

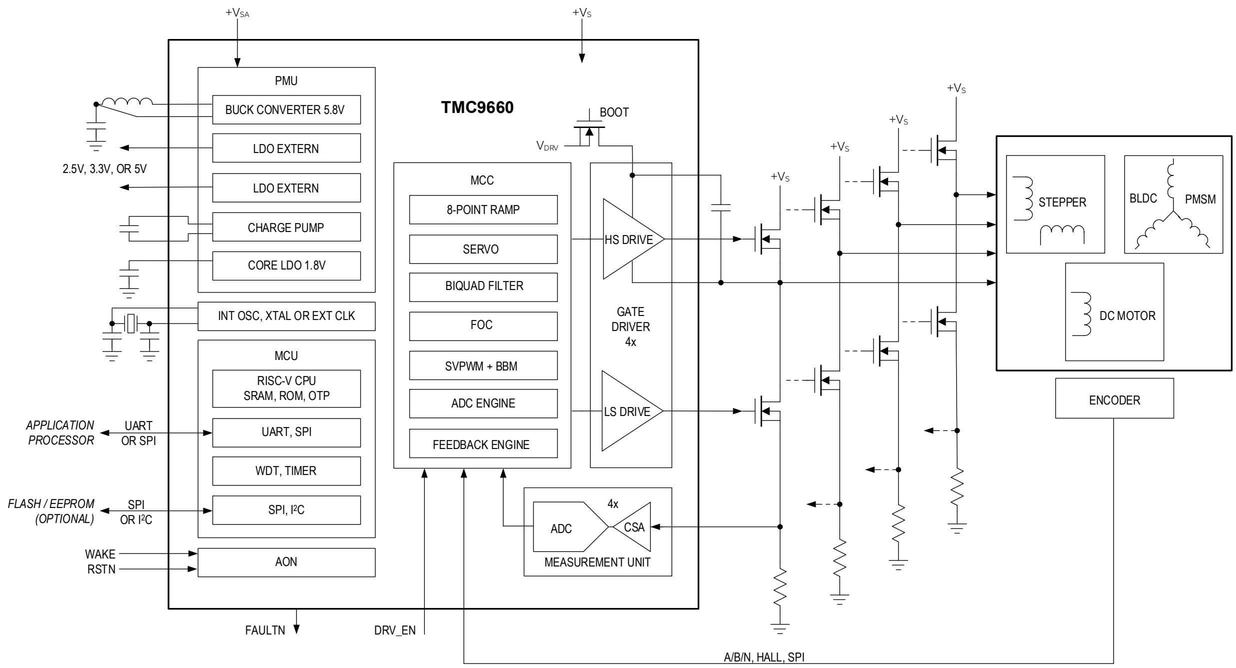

Figure 1. TMC9660 simplified block diagram

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

COMMUNICATION INTERFACES

For parameter read / write access either UART or SPI interface may be used. Both interfaces share the same command structure based on the TMCL protocol. During the initial system and IO-configuration the desired interfaces must be specified.

UART Interface

The UART interface uses two signals/pins – UART_TXD (transmit data out) and UART_RXD (receive data in). For bus communication – e.g., RS485 – an additional signal/pin UART_TXEN is available for switching an external transceiver between transmit and receive mode in hardware.

The communication protocol itself follows a strict command/reply order. Thus, new commands should not be sent from the attached microcontroller before the reply for the previous command has been received.

The TMC9660 Parameter Mode does not send out any reply data before receiving a command first to avoid any collision on the bus interface.

Every command consists of nine bytes sent with the least-significant-bit (LSB) first. It starts with a one-byte address, a one-byte command field, a one-byte type field, a one-byte motor/bank field and a four-byte value field.

Table 1. Command format for parameter read/write access through UART

BYTE	**0**		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Bit	0	1-7	8-15	16-27	28-31	32-63				64-71
Desc.	Sync bit	Module Address	Operation	Type	Motor/Bank	Data (big-endian)				Checksum

The sync bit is always 1. It is used for automatic baud rate detection.

The module address reuses the upper 7 bits of the bootloader device address.

The checksum is calculated by adding up the first eight bytes using 8-bit addition. Here is an example for checksum calculation using C code:

```
unsigned char i, nChecksum;
unsigned char nCommand[9];

// nCommand[0..7] should contain the command
// Calculate checksum from command bytes
nChecksum = nCommand[0];
for (i = 1; i < 8; i++)
{
  nChecksum += nCommand[i];
}
// Insert checksum as last byte of the command
nCommand[8] = nChecksum;
```

Figure 2. Checksum calculation for command

For almost all commands, a reply is be sent back from the TMC9660 Parameter Mode. The reply also consists of nine bytes.

Table 2. Reply format for parameter read/write access through UART

BYTE	0	1		2	3	4	5	6	7	8
Bit	0-7	8	9-15	16-23	24-31	32-63				64-71
Desc.	Host Address	Sync bit	Module Address	TMCL Status	Operation	Data (big-endian)				Checksum

The sync bit is still present in the reply datagram and is still 1.

The checksum calculation is the same as for the command format.

Serial-Peripheral Interface

The SPI for communication with an external microcontroller uses the SPI peripheral device of the TMC9660 Parameter Mode and requires 4 pins for communication. The SPI interface operates in mode 3.

The external microcontroller operates as an SPI controller. Every SPI command from the external microcontroller to the TMC9660 Parameter Mode is expected to have a length of 64 bit. A reply for this command has the same length and is sent from the TMC9660 Parameter Mode back to the external microcontroller with the next SPI command. All data is sent with most significant bit (MSB) first. The data is sent in big-endian.

The *Table 3* and *Table 4* show the command and reply format. The response always matches the previously requested datagram. For the initial datagram, only zeros are returned, and the status code indicates that it is the first datagram.

The *Table 5* lists the status codes. A not ready status Indicates that the system is still busy processing the command. The reply sent in this datagram is not the reply to the processed command, and the command sent in this datagram is ignored. Simply resend the datagram until this status no longer appears.

Table 3. Command format for parameter read/write access through SPI

BYTE	0	1	2	3	4	5	6	7
Bit	0-7	8-19	20-23	24-55				56-63
Desc.	Operation	Type	Motor/Bank	Data				Checksum

Table 4. Reply format for parameter read/write access through SPI

BYTE	0	1	2	3	4	5	6	7
Bit	0-7	8-19	20-23	24-55				56-63
Desc.	SPI Status	TMCL Status	Operation	Data				Checksum

Table 5. SPI status codes

VALUE	STATUS CODE	DESCRIPTION
0xFF	SPI_STATUS_OK	Indicates that the operation was successful
0x00	SPI_STATUS_CHECKSUM_ERROR	Indicates that there was a checksum error
0x0C	SPI_STATUS_FIRST_CMD	Indicates the initial response after initialization
0xF0	SPI_STATUS_NOT_READY	Indicates that the system is not ready for a new command

BOOT CONFIGURATION

This section lists configuration settings for setting up the TMC9660 for parameter mode operation. For details on how the configuration mechanism is used, refer to the *Bootloader Configuration* section in the product data sheet for TMC9660. The tables in the following sections list the options that can be configured for the parameter mode features.

ABN Encoder 1

Table 6. Boot configuration options for ABN 1 encoder

NAME	OFFSET	BITS	DESCRIPTION
ABN1_ENABLE	32	1	Enables the usage of ABN1. When enabled, the following other ABN1 settings take effect, otherwise they are ignored. Default: 0
ABN1_A	32	10-11	Selects which pin to use for A input: **0: GPIO5** 1: GPIO8 2: GPIO17 3: RESERVED
ABN1_B	32	12-13	Selects which pin to use for B input: **0: GPIO1** 1: GPIO13 2: GPIO18 3: RESERVED
ABN1_N	32	14-15	Selects which pin to use for N input: **0: N channel disabled** 1: GPIO14 2: GPIO16 3: RESERVED

Hall

Table 7. Boot configuration options for Hall encoder

NAME	OFFSET	BITS	DESCRIPTION
HALL_ENABLE	32	0	Enables the usage of the Hall encoder. When enabled, the following other Hall settings take effect, otherwise they are ignored. Default: 0
HALL_U	32	4-5	Selects which pin to use for U input: GPIO2 GPIO7 GPIO9 RESERVED
HALL_V	32	6-7	Selects which pin to use for V input: GPIO3 GPIO15 RESERVED RESERVED

NAME	OFFSET	BITS	DESCRIPTION
HALL_W	32	8-9	Selects which pin to use for W input: GPIO4 GPIO8 GPIO10 RESERVED

SPI Encoder

Table 8. Boot configuration for SPI encoder

NAME	OFFSET	BITS	DESCRIPTION
SPI_ENC_ENABLE	38	0	Enables the usage of the SPI encoder. When enabled, the following other SPI encoder settings take effect, otherwise they are ignored. Default: 0
SPI_ENC_BLOCK	38	1	Selects the SPI encoder to use: **0: SPI0** 1: SPI1 Note: The SPI encoder cannot share an SPI interface with the SPI flash and the SPI subordinate features.
SPI_ENC_MODE	38	2-3	The SPI mode to use to communicate with the SPI encoder
SPI_ENC_FREQ	38	4-7	Selects the SPI encoder frequency: $$f_{SPIEncoder} = \frac{f_{System}}{SPI\_ENC\_FREQ + 4}$$ The system frequency (f_{System}) is 40 MHz. Default: 0
SPI_ENC_CS_PIN	38	8-9	Selects which pin to use for SPI encoder chip select. The pin selection depends on the selected SPI block: SPI0: **0: GPIO8** 1: GPIO12 2: GPIO13 3: GPIO16 SPI1: **0: GPIO15** 1: RESERVED 2: RESERVED 3: RESERVED
SPI_ENC_CS_POL	38	19	Selects the polarity of the chip select signal: 0: **Active high** 1: Active low

ABN Encoder 2

Note: The ABN2 encoder does not support usage of an N channel signal.

Table 9. Boot configuration for ABN 2 encoder

NAME	OFFSET	BITS	DESCRIPTION
ABN2_ENABLE	34	12	Enables the usage of the ABN2 encoder. When enabled, the following other ABN2 settings take effect, otherwise they are ignored. Default: 0
ABN2_A	34	13	Selects which pin to use for A input: **0: GPIO6** 1: GPIO15
ABN2_B	32	14-15	Selects which pin to use for B input: **0: GPIO7** 1: GPIO11 2: GPIO16 3: RESERVED

REF Switches

Table 10. Boot configuration for reference switches

NAME	OFFSET	BIT	DESCRIPTION
REF_L_PIN	34	0-1	Selects which pin to use for REF_L input: **0: REF_L input is disabled** 1: GPIO2 2: GPIO12 3: GPIO16
REF_R_PIN	34	2-3	Selects which pin to use for REF_R input: **0: REF_R input is disabled** 1: GPIO3 2: GPIO18 3: RESERVED
REF_H_PIN	34	4-6	Selects which pin to use for REF_H input: **0: REF_H input is disabled** 1: GPIO4 2: GPIO7 3: GPIO15 4: GPIO17 5: RESERVED 6: RESERVED 7: RESERVED

StepDir Interface

Table 11. Boot configuration for step direction interface

NAME	OFFSET	BITS	DESCRIPTION
STEPDIR_ENABLE	34	8	Enables the usage of the STEP/DIR interface. When enabled, the following other STEP/DIR settings take effect, otherwise they are ignored. Default: 0
STEP_PIN	34	9-10	Selects which pin to use for STEP input: **0: GPIO7** 1: GPIO11 2: GPIO16 3: RESERVED
DIR_PIN	34	11	Selects which pin to use for Dir input: **0: GPIO6** 1: GPIO15

Mechanical Brake

Table 12. Boot configuration for mechanical break

NAME	OFFSET	BITS	DESCRIPTION
MECH_BRAKE_ENABLE	36	12	Enables the usage of the mechanical brake. When enabled, the following other mechanical brake settings take effect, otherwise they are ignored. Default: 0
MECH_BRAKE_OUTPUT	36	13-14	Selects which pin to use for mechanical brake output: **0: GPIO8** 1: GPIO10 2: GPIO18 3: Y2_LS Note: Y2_LS is a gatedriver output, able to drive a low-site FET. The GPIO outputs are digital signals not meant to directly drive the gate of an FET. Note: Y2_LS is not available when using stepper motor.

Brakechopper

Table 13. Boot configuration for brake chopper

NAME	OFFSET	BITS	DESCRIPTION
BRAKECHOPPER_ENABLE	36	4	Enables the usage of the brakechopper. When enabled, the following other brakechopper settings take effect, otherwise they are ignored. Default: 0

NAME	OFFSET	BITS	DESCRIPTION
BRAKECHOPPER_OUTPUT	36	5-9	Selects which pin to use for brakechopper output: - 0-18: GPIO0 – GPIO18 - 19: Y2_HS (driving a low-side FET) - 20-31: RESERVED Default: 0 Note: Y2_HS is a gatedriver output, able to drive a low-site FET. The GPIO outputs are digital signals not meant to directly drive the gate of an FET. Note: Y2_LS is not available when using stepper motor.

External Memory Storage Selection

Table 14. Boot configuration for external memory storage

NAME	OFFSET	BITS	DESCRIPTION
MEM_TMCL_SCRIPT	40	0-1	Selects which external memory to use for storing TMCL script: **0: TMCL Script disabled** 1: External SPI flash 2: External I2C EEPROM 3: RESERVED
MEM_PARAMETERS	40	2-3	Selects which external memory to use for storing parameter settings: **0: Parameter storage disabled** 1: External SPI flash 2: External I2C EEPROM 3: RESERVED

Note: Any external memory used must be partitioned and have a partition with the correct type available to enable using these features.

SUPPORTED PARAMETER MODE COMMANDS

The TMCL protocol is used to communicate with the TMC9660 Parameter Mode. *Table 18* provides a comprehensive list of all available operations along with their corresponding optional parameters. The reply codes for these operations are listed in *Table 19*. Scripts based on TMCL commands can be executed autonomously by TMC9660 Parameter Mode. For a more detailed description of the scripting functionality and related commands, see the section *Script*.

SET and GET Parameters and Global Parameters

The primary configuration of TMC9660 Parameter Mode is achieved using either parameters or global parameters. System-related settings can be located within the global parameters. Subsets of parameters related to specific features can be found in their respective feature sections.

To set and get a parameter, the operations "SAP" and "GAP" must be used, respectively. The motor number must be zero. For a list of all available parameters, see the section *Parameters*.

For global parameters, the operations "SGP" and "GGP" are used. The available banks are 0, 2, and 3. For a list of all available global parameters, see the section *Global Parameters*.

If the write access was successful, the status code "REPLY_OK" should be replied.

Flags

The TMC9660 Parameter Mode includes three distinct flag parameters, each representing various status and error conditions. These parameters are identified as GENERAL_STATUS_FLAGS, GENERAL_ERROR_FLAGS, and GDRV_ERROR_FLAGS. Within these parameters, individual bits correspond to unique flags.

Certain flags are designated as read-only, whereas others are clearable. They can be cleared by writing the specified bit to one or in other words writing the flags mask value to the parameter. Occasionally, clearing specific flags is necessary to initiate an event or to rectify a fault state. Detailed explanations regarding this process can be found in the respective sections dedicated to each feature within this document. All the available flags are listed in *Table 58*, *Table 59,* and *Table 60* of section *All flags*.

Storing System Settings in External Memory

The TMC9660 Parameter Mode supports storing a system configuration to automatically restore it upon system restart. All parameters that can be stored are marked as Read / Write / External memory (RWE). The external memory must be configured in the boot configuration and a valid partition table must be written to the external memory device. Note that the CONFIG_ERROR flag is set if a valid partition table has been flashed but no configuration written to it yet.

To store the current system configuration, the operation STAP with type 0xFFF, bank 0xF, and value 0xFFFFFFFF must be sent. The successful storage can be verified by reading the status flag CONFIG_STORED in the GENERAL_STATUS_FLAGS parameter. Note that it may take a short amount of time to complete writing all settings to external storage depending on the external memory. If the flag CONFIG_ERROR persists after writing the configuration, verify the external memory configuration and the partition table on the external memory.

The system automatically checks for stored settings in external memory on startup. If settings are found, the system automatically restores them. The flag CONFIG_LOADED in the GENERAL_STATUS_FLAGS parameter indicates that the restore was successful.

To clear all settings previously written to memory, send the operation code "FactoryDefault". The values in external memory are erased. After a reboot, the system restores the default values.

GPIO Control

The GPIO pins can be read and controlled using the SIO and GIO operations. The GPIO should be configured in the boot configuration. The global parameters IO_DIRECTION_MASK, IO_INPUT_PULLUP/PULLDOWN_ENABLE_MASK and IO_INPUT_PULLUP/PULLDOWN_DIRECTION_MASK allow modification during operation but should be used with care. To set a digital output, the SIO operation must be sent. The port number must be handed over as type and the desired setting as value. To get an IO value, the GIO operation is utilized. The port number is handed over as type. To read a digital pin, the motor/bank value must be zero; for an analog one, it must be one.

RamDebug

The RamDebug feature is intended for debugging and monitoring. It allows the collection of samples on up to four channels with a sampling rate of up to 25kHz and 4096 samples. The feature offers a trigger system similar to an oscilloscope, which allows triggering on events to capture them. The feature can be accessed using the operation code “RamDebug”. The type values for this command are listed in *Table 16*.

Reset and Initialize RamDebug

To configure a sample capture, the system first needs to be initialized by using the type code zero. After that, the channels can be configured.

Set Prescaler and Sample Count

To configure the sampling frequency and the number of samples, the prescaler and the sample count can be specified. The maximum number of samples available depends on the number of channels to be used. Note that the RAMDebug sample count must be set to the total amount of samples, not the number of samples per channel.

The sampling frequency is a value derived from the RAMDebug frequency.

$$f_{sampling} = \frac{f_{RAMDebug}}{value_{downsampling} + 1}$$

The RAMDebug frequency is derived from the PWM frequency. The RAMDebug frequency is limited to 25kHz. If the PWM frequency is set to a frequency higher than 25kHz, RAMDebug automatically prescales the PWM frequency to stay at or below 25kHz. See the section *Control Loop Frequencies* to find more information.

Table 15. PWM frequency vs. RAMDebug frequency

PWM FREQUENCY	RAMDEBUG FREQUENCY
$100\ kHz \geq f_{PWM} > 75\ kHz$	$f_{RAMDebug} = f_{PWM}/4$
$75\ kHz \geq f_{PWM} > 50\ kHz$	$f_{RAMDebug} = f_{PWM}/3$
$50\ kHz \geq f_{PWM} > 25\ kHz$	$f_{RAMDebug} = f_{PWM}/2$
$25\ kHz \geq f_{PWM}$	$f_{RAMDebug} = f_{PWM}$

Set Number of Pretrigger Samples

Pretrigger samples allow to observe the channel states slightly before the trigger event. The number of pretrigger samples that is desired can be specified using the type code 13. The number of pretrigger samples specified is in total and not per channel.

Set Up the Channels

To set a channel, the RamDebug state must be idle. To set a channel, the type code 4 is used. The motor/bank values specify the channel type, and the value field specifies the address of the desired value. For global parameter access, the first byte of the address specifies the bank number. The parameter/global parameter number is specified by the last three bytes. The channel setup must be repeated for every channel that is to be sampled. The channel number is defined by the order in which the channels are written.

Set a Trigger

The configuration of the trigger channel is identical to a capture channel. Additionally, a mask and shift value can be specified by type code 6. The mask and shift are applied to the channel value before checking the trigger. The trigger criterion is specified using type code 7, with the motor/bank value specifying the trigger type and the value specifying the trigger threshold. Type code 7 starts the measurement.

Get Status

To get the current status, type code 10 is used. This allows us to check if the system has already triggered and if the capture is done. For a full list of status codes see *Table 17*.

Get Samples

As soon as the sampling is done, samples can be downloaded. The sampled values are requested using type code 9. The motor/bank value specifies the requested sample. The samples are ordered by acquisition time and channel. For a four-channel system, index zero would return the first element of channel zero, index 1 the first element of channel one, etc. Index 4 returns the second value of channel zero.

Table 16. List of RAMDebug type commands

NUMBER	TYPE	MOTOR/BANK	VALUE	DESCRIPTION
0	Initialize and reset	-	-	Initialize and reset
1	Set sample count	-	Number	Number of samples to collect in total (not per-channel)
3	Set prescaler	-	Prescale Value	Sets divider for sampling rate. Dividing maximum frequency by value+1
4	Set channel	Type: 0: Disabled 1: Parameter 3: Global parameter	Motor/Bank: 0xFF000000 AP/GP number: 0x00000FFF	Set channel
5	Set trigger channel	Type: 0: Disabled 1: Parameter 3: Global parameter	Motor/Bank: 0xFF000000 AP/GP number: 0x00000FFF	Specify source of the data to be triggered on
6	Set trigger mask shift	Shift	Mask	Specify a mask and shift value to be applied to the trigger value.

NUMBER	TYPE	MOTOR/BANK	VALUE	DESCRIPTION
7	Enable trigger	Type: 0: Unconditional 1: Rising edge signed 2: Falling edge signed 3: Dual edge signed 4: Rising edge unsigned 5: Falling edge unsigned 6: Both edge unsigned	Threshold	Start the measurement by enabling the trigger.
8	Get state	-	-	Request state of RamDebug. 0: Idle 1: Trigger 2: Capture 3: Complete 4: Pretrigger See *Table 17*.
9	Read sample	index	-	Returns the sampled values.
10	Get info	-	0: Max number of channels 1: Buffer size in samples 2: RAMDebug frequency 3: Captured sample count 4: RAMDebug prescaler value on trigger event.	Readout general information
11	Get channel type	index	-	Readout channel type information
12	Get channel address	Index	-	Readout channel address
13	Set pretrigger sample count	-	Number of samples	Set the total number of pretrigger samples (not per-channel)
14	Get pretrigger sample count	-	-	Get the total number of pretrigger samples

Table 17. List of RAMDebug states

NUMBER	NAME	DESCRIPTION
0	Idle	RAMDebug is not running and can be configured. Use type code 0 to enter this state.
1	Trigger	RAMDebug is waiting for the trigger event to happen. When updating a value that RAMDebug is triggering on, ensure this state is reached before updating.

2	Capture	RAMDebug has been triggered and is capturing samples.
3	Complete	RAMDebug has finished capturing samples. The data can now be downloaded using type code 9.
4	Pretrigger	RAMDebug is capturing samples for the pretrigger.

All TMCL Operation and Reply Codes

Table 18 shows a list of all the TMCL commands available. A more detailed description of the scripting related command can be found in the section *Script*. All possible reply codes are listed in *Table 19*.

Table 18. TMCL commands

NUMBER	OPERATION	TYPE	MOTOR/ BANK	VALUE	DESCRIPTION
3	MST	-	-	-	Stops motor movement
5	SAP	parameter	0	value	Sets parameter (motion control specific settings)
6	GAP	parameter	0	-	Gets parameter (read out motion control specific settings)
7	STAP	0xFFF	0xF	0xFFFFFFFF	Stores all storable parameters to external memory.
9	SGP	parameter	0,2,3	value	Sets global parameter (module specific settings e.g.\ communication settings or TMCL user variables)
10	GGP	parameter	-	-	Gets global parameter (read out module specific settings e.g.\ communication settings or TMCL user variables)
13	RFS	START| STOP| STATUS	0	-	Reference search
14	SIO	port number	0	0,1	Sets digital output to specified value
15	GIO	port number	0: digital 1: analog	-	Gets value of analog/digital input
19	CALC	operation	-	value	Arithmetical operation between accumulator and direct value
20	COMP	-	-	value	Compares accumulator with value
21	JC	condition		address	Jump conditional
22	JA	-	-	address	Jump absolute

NUMBER	OPERATION	TYPE	MOTOR/ BANK	VALUE	DESCRIPTION
23	CSUB	-	-	address	Calls subroutine
24	RSUB	-	-		Returns from subroutine
25	EI	-	-	Interrupt number	Enables interrupt
26	DI	-	-	Interrupt number	Disables interrupt
27	WAIT	condition	-	ticks	Waits with further program execution
28	STOP	-	-	-	Stops program execution
33	CALCX	type	-	-	Arithmetical operation between accumulator and X-register
34	AAP	parameter	0	-	Accumulator to axis parameter
35	AGP	parameter	0,2,3	-	Accumulator to global parameter
36	CLE	flag	-	-	Clears an error flag
37	VECT	interrupt number	-	address	Defines interrupt vector
38	RETI	-	-	-	Returns from interrupt
40	CALCVV	type	user variable 1	user variable 2	Arithmetical operation between two user variables
41	CALCVA	type	user variable	-	Arithmetical operation between user variable and accumulator
42	CALCAV	type	user variable	-	Arithmetical operation between accumulator and user variable
43	CALCVX	type	user variable	-	Arithmetical operation between user variable and X register
44	CALCXV	type	user variable	-	Arithmetical operation between X register and user variable
45	CALCV	type	-	value	Arithmetical operation between user variable and direct value
48	RST	-	-	address	Restarts the program from the given address
49	DJNZ	user variable	-	address	Decrement and jump if not zero
55	SIV	-	-	value	Sets indexed variable
56	GIV	-	-	-	Gets indexed variable

NUMBER	OPERATION	TYPE	MOTOR/ BANK	VALUE	DESCRIPTION
57	AIV	-	-	-	Accumulator to indexed variable
128	ApplStop	-	-	-	Stops a running TMCL program
129	ApplRun	0: from current address 1: from specific address	-	address	Starts or continue execution of a TMCL program
130	ApplStep	-	-	-	Executes only the next TMCL command
131	ApplReset	-	-	-	Stops a running TMCL program and resets program counter to start of program.
132	DownloadStart	-	-	-	Allows all TMCL commands except Reset and Stop to be executed. After command DownloadStart is sent, all following commands are not executed but stored into nonvolatile memory until “DownloadEnd” is sent.
133	DownloadEnd	-	-	-	Ends the TMCL download mode.
134	ReadMem	-	-	address	Returns script command at address.
135	GetStatus	*Script*	-	-	Gets status of script.
136	GetVersion	-	-	-	Gets version of system
137	FactoryDefault	-	-	-	Clears settings stored in external memory.
141	Breakpoint	0: Add breakpoint 1: Del. breakpoint 2: Del. all breakpoints 3. Get max. number of breakpoints	-	address	Adds and deletes breakpoints.

NUMBER	OPERATION	TYPE	MOTOR/ BANK	VALUE	DESCRIPTION
142	RAMDebug	See *Table 16*	See *Table 16*	See *Table 16*	Access to RAMDebug control.
157	GetInfo	0: ID 1: Version	-	-	Gets ID and version info
242	Boot	0x81	0x92	0xA3B4C5D6	Returns the system to the bootloader. The system must be manually set to "System off" commutation mode before executing this command, see the section *Erratum 3: Exiting to bootloader with ongoing motor commutation* for details.

Table 19. TMCL status codes

NUMBER	TMCL STATUS	DESCRIPTION
100	REPLY_OK	Command executed successfully
101	REPLY_CMD_LOADED	Command loaded successfully
1	REPLY_CHKERR	Check error occurred
2	REPLY_INVALID_CMD	Invalid command received
3	REPLY_WRONG_TYPE	Wrong type of data received
4	REPLY_INVALID_VALUE	Invalid value received
6	REPLY_CMD_NOT_AVAILABLE	Command not available
7	REPLY_CMD_LOAD_ERROR	Error occurred while loading command
9	REPLY_MAX_EXCEEDED	Maximum limit exceeded
10	REPLY_DOWNLOAD_NOT_POSSIBLE	Download operation not possible

GATE DRIVER

The gate driver is responsible for providing the necessary signals to the gates of the power transistors, ensuring efficient and reliable motor control. It supports various PWM modes, protection mechanisms, and configuration options to adapt to different motor types and applications. For further information, refer to the TMC9660 data sheet.

Configuration

The gate driver controls four half-bridges, each of which is responsible for driving a phase of the motor. The general configuration for the TMC9660 Parameter Mode is located in the boot configuration. The system automatically selects the enabled phases based on the boot configuration and the MOTOR_TYPE parameter. Many off the TMC9660 Parameter Mode gate driver settings are separated into the three phases always used for the motor UX1, VX2, WY1 and the phase Y2 that can have alternate functions for mechanical brake and brake chopper.

After startup, the gate driver is disabled and can be activated by both asserting the DRV_EN pin and changing the COMMUTATION_MODE parameter to a value other than “System off”. The IDLE_MOTOR_PWM_BEHAVIOR parameter determines whether the gate driver is disabled again when switching back to “System off”. By default, the gate driver is disabled again.

The gate driver uses bootstrap capacitors to provide the necessary voltage for driving the high-side transistors. The BOOTSTRAP_CURRENT_LIMIT parameter allows you to define the maximum charge current for these capacitors.

An undervoltage protection for the bootstrap capacitors is activated by default and can be disabled using the parameters UVP_BST_UVW_ENABLE and UVP_BST_Y2_ENABLE. The flags U_BOOTSTRAP_UNDERVOLTAGE, V_BOOTSTRAP_UNDERVOLTAGE, W_BOOTSTRAP_UNDERVOLTAGE and Y2_BOOTSTRAP_UNDERVOLTAGE indicate a fault condition related to the corresponding phase. By default, these errors are handled as described in the section *Fault Handling*.

Currents and Timing

The gate driver unit allows to control the charge times and currents of the MOSFETs. The drive times are defined by the parameters DRIVE_TIME_SINK_UVW, DRIVE_TIME_SINK_Y2 and DRIVE_TIME_SOURCE_UVW, DRIVE_TIME_SOURCE_Y2. The discharge currents are defined by the parameters UVW_SINK_CURRENT, Y2_SINK_CURRENT. To set the charge currents UVW_SOURCE_CURRENT, Y2_SOURCE_CURRENT must be configured. A timing diagram for the gate driver can be found in *Figure 3*.

The adaptive mode is a feature designed to optimize MOSFET gate driver performance by dynamically adjusting the MOSFET's discharge time. The feature can be switched on using the parameter ADAPTIVE_DRIVE_TIME_UVW and ADAPTIVE_DRIVE_TIME_Y2. The gate driver continuously monitors the gate voltage during the discharge cycle. If the gate is fully discharged early, the discharge cycle is shortened by terminating the gate drive current before the full DRIVE_TIME_SINK time elapses. In this operation mode, the DRIVE_TIME_SINK parameters act as an upper limit for the discharge time. The adaptive mode has no effect on the DRIVE_TIME_SOURCE.

The Break Before Make (BBM) time, also referred to as dead time defines the interval between the deactivation of one MOSFET and the activation of the complementary MOSFET. This delay is essential to prevent a short circuit, often termed "shoot-through," which can lead to significant power dissipation and potentially damage the device. The BBM time is configurable through the parameters BREAK_BEFORE_MAKE_TIME_LOW_UWV and BREAK_BEFORE_MAKE_TIME_HIGH_UVW for U, V, W and BREAK_BEFORE_MAKE_TIME_LOW_Y2 and BREAK_BEFORE_MAKE_TIME_HIGH_Y2 for Y2. It is generally recommended to set the BBM values to zero and rely on the timing parameters.

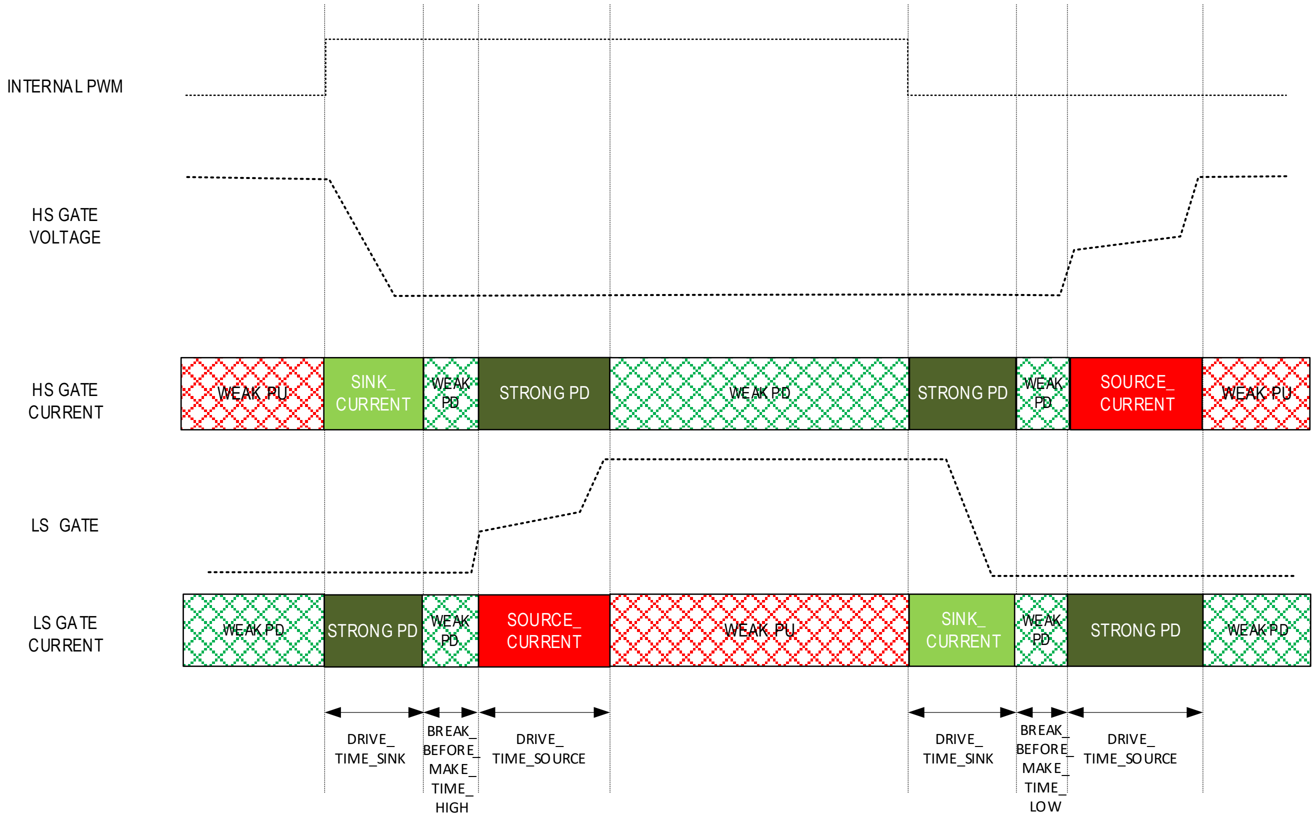

Figure 3. Gate driver timing diagram

Table 20. Gate driver timer and current settings

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
233	PWM_L_OUTPUT_POLARITY PWM_L output polarity. False: ACTIVE_HIGH True: ACTIVE_LOW	0, 1	0	RWE
234	PWM_H_OUTPUT_POLARITY PWM_H output polarity. False: ACTIVE_HIGH True: ACTIVE_LOW	0, 1	0	RWE
235	BREAK_BEFORE_MAKE_TIME_LOW_UVW [8.33ns] Break before make time for the low side gates of the UVW phases. Applied before switching from high to low.	0 … 255	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
236	BREAK_BEFORE_MAKE_TIME_HIGH_UVW [8.33ns] Break before make time for the high side gates of the UVW phases. Applied before switching from low to high.	0 … 255	0	RWE
237	BREAK_BEFORE_MAKE_TIME_LOW_Y2 [8.33ns] Break before make time for the low side gate of the Y2 phase. Applied before switching from high to low.	0 … 255	0	RWE
238	BREAK_BEFORE_MAKE_TIME_HIGH_Y2 [8.33ns] Break before make time for the high side gate of the Y2 phase. Applied before switching from low to high.	0 … 255	0	RWE
239	USE_ADAPTIVE_DRIVE_TIME_UVW If enabled, the discharge cycle of the low- and high-side gates for the UVW phases is shortened by monitoring the gate voltages. If enabled, the value on T_DRIVE_SINK acts as an upper bound instead of a fixed time. False: DISABLED True: ENABLED	0, 1	1	RWE
240	USE_ADAPTIVE_DRIVE_TIME_Y2 If enabled, the discharge cycle of the low- and high-side gates for the Y2 phase is shortened by monitoring the gate voltages. If enabled, the value on T_DRIVE_SINK acts as an upper bound instead of a fixed time. False: DISABLED True: ENABLED	0, 1	1	RWE
241	DRIVE_TIME_SINK_UVW Discharge time for the low and high side gates of the UVW phases. During this time, the full sink current is applied. The applied time is defined as (1s / 120MHz) x (2 x DRIVE_TIME_SINK_UVW + 3).	0 … 255	255	RWE
242	DRIVE_TIME_SOURCE_UVW Charge time for the low and high side gates of the UVW phases. During this time, the full source current is applied. The applied time is defined as (1s / 120MHz) x (2 x DRIVE_TIME_SOURCE_UVW + 3).	0 … 255	255	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
243	DRIVE_TIME_SINK_Y2 Discharge time for the low and high side gates of the Y2 phase. During this time, the full sink current is applied. The applied time is defined as (1s / 120MHz) x (2 x DRIVE_TIME_SINK_Y2 + 3).	0 ... 255	255	RWE
244	DRIVE_TIME_SOURCE_Y2 Charge time for the low and high side gates of the Y2 phase. During this time, the full source current is applied. The applied time is defined as (1s / 120MHz) x (2 x DRIVE_TIME_SOURCE_Y2 + 3).	0 ... 255	255	RWE
245	UVW_SINK_CURRENT Limit the maximum sink current for the low and high side gates of the UVW phases. 0: CUR_50_MILLIAMP 1: CUR_100_MILLIAMP 2: CUR_160_MILLIAMP 3: CUR_210_MILLIAMP 4: CUR_270_MILLIAMP 5: CUR_320_MILLIAMP 6: CUR_380_MILLIAMP 7: CUR_430_MILLIAMP 8: CUR_580_MILLIAMP 9: CUR_720_MILLIAMP 10: CUR_860_MILLIAMP 11: CUR_1000_MILLIAMP 12: CUR_1250_MILLIAMP 13: CUR_1510_MILLIAMP 14: CUR_1770_MILLIAMP 15: CUR_2000_MILLIAMP	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	4	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
246	UVW_SOURCE_CURRENT Limit the maximum source current for the low and high side gates of the UVW phases. 0: CUR_25_MILLIAMP 1: CUR_50_MILLIAMP 2: CUR_80_MILLIAMP 3: CUR_105_MILLIAMP 4: CUR_135_MILLIAMP 5: CUR_160_MILLIAMP 6: CUR_190_MILLIAMP 7: CUR_215_MILLIAMP 8: CUR_290_MILLIAMP 9: CUR_360_MILLIAMP 10: CUR_430_MILLIAMP 11: CUR_500_MILLIAMP 12: CUR_625_MILLIAMP 13: CUR_755_MILLIAMP 14: CUR_855_MILLIAMP 15: CUR_1000_MILLIAMP	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	4	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
247	Y2_SINK_CURRENT Limit the maximum sink current for the low and high side gates of the Y2 phase. 0: CUR_50_MILLIAMP 1: CUR_100_MILLIAMP 2: CUR_160_MILLIAMP 3: CUR_210_MILLIAMP 4: CUR_270_MILLIAMP 5: CUR_320_MILLIAMP 6: CUR_380_MILLIAMP 7: CUR_430_MILLIAMP 8: CUR_580_MILLIAMP 9: CUR_720_MILLIAMP 10: CUR_860_MILLIAMP 11: CUR_1000_MILLIAMP 12: CUR_1250_MILLIAMP 13: CUR_1510_MILLIAMP 14: CUR_1770_MILLIAMP 15: CUR_2000_MILLIAMP	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	4	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
248	Y2_SOURCE_CURRENT Limit the maximum source current for the low and high side gates of the Y2 phase. 0: CUR_25_MILLIAMP 1: CUR_50_MILLIAMP 2: CUR_80_MILLIAMP 3: CUR_105_MILLIAMP 4: CUR_135_MILLIAMP 5: CUR_160_MILLIAMP 6: CUR_190_MILLIAMP 7: CUR_215_MILLIAMP 8: CUR_290_MILLIAMP 9: CUR_360_MILLIAMP 10: CUR_430_MILLIAMP 11: CUR_500_MILLIAMP 12: CUR_625_MILLIAMP 13: CUR_755_MILLIAMP 14: CUR_855_MILLIAMP 15: CUR_1000_MILLIAMP	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	4	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
249	BOOTSTRAP_CURRENT_LIMIT Bootstrap current limit. 0: CUR_45_MILLIAMP 1: CUR_91_MILLIAMP 2: CUR_141_MILLIAMP 3: CUR_191_MILLIAMP 4: CUR_267_MILLIAMP 5: CUR_292_MILLIAMP 6: CUR_341_MILLIAMP 7: CUR_391_MILLIAMP	0, 1, 2, 3, 4, 5, 6, 7	7	RWE

Protections

The gate driver incorporates several protections features to ensure the safe and reliable operation of the motor drive system. These features are designed to detect and respond to fault conditions, preventing damage to the power stage. Each fault gets reported in the flag parameter GDRV_ERROR_FLAGS. The fault handling is described in the section *Fault Handling*.

Overcurrent and Short-Circuit Protection (OCP)

Overcurrent protection (OCP) is a critical safety feature that monitors the current flowing through the power MOSFETS. If the current exceeds a predefined threshold, the OCP mechanism triggers, typically by disabling the gate driver. This helps to prevent excessive current from damaging the MOSFETS.

The high-side OCP is always be measured using the voltage drop over the MOSFET, while the low-side OCP can also use the shunt. This and the OCP thresholds for the low-side and high-side MOSFETS can be configured through the parameters listed in *Table 21*. These parameters also allow you to set the deglitch time, blanking time, and threshold level for each channel. The deglitch time determines the duration for which the overcurrent condition must persist before it is recognized as a fault. The configurable blanking prevents the OCP detection after charge and discharge cycle to filter out any transient spikes or noise. The current threshold level is programmable and represents the current at which the OCP mechanism is triggered. The OCP mechanism includes an automatic retry feature, specified in the parameters listed in *Table 40*. After a specified number of retries, if the overcurrent condition persists, the gate driver remains disabled until the fault is cleared.

All related parameters are listed in *Table 21*.

Table 21. Overcurrent protection parameters

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
254	OVERCURRENT_PROTECTION_UVW_LOW_SIDE_ENABLE Enable the overcurrent protection on the low side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
255	OVERCURRENT_PROTECTION_UVW_HIGH_SIDE_ENABLE Enable the overcurrent protection on the high side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
256	OVERCURRENT_PROTECTION_Y2_LOW_SIDE_ENABLE Enable the overcurrent protection on the low side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
257	OVERCURRENT_PROTECTION_Y2_HIGH_SIDE_ENABLE Enable the overcurrent protection on the high side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
258	OVERCURRENT_PROTECTION_UVW_LOW_SIDE_THRESHOLD Overcurrent protection threshold for the low side of the UVW phases (uses second list if OVERCURRENT_PROTECTION_UVW_LOW_SIDE_USE_VDS=true). 0: V_80_OR_63_MILLIVOLT 1: V_165_OR_125_MILLIVOLT 2: V_250_OR_187_MILLIVOLT 3: V_330_OR_248_MILLIVOLT 4: V_415_OR_312_MILLIVOLT 5: V_500_OR_374_MILLIVOLT 6: V_582_OR_434_MILLIVOLT 7: V_660_OR_504_MILLIVOLT 8: V_125_OR_705_MILLIVOLT 9: V_250_OR_940_MILLIVOLT 10: V_375_OR_1180_MILLIVOLT 11: V_500_OR_1410_MILLIVOLT 12: V_625_OR_1650_MILLIVOLT 13: V_750_OR_1880_MILLIVOLT 14: V_875_OR_2110_MILLIVOLT 15: V_1000_OR_2350_MILLIVOLT	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
259	OVERCURRENT_PROTECTION_UVW_HIGH_SIDE_THRESHOLD Overcurrent protection threshold for the high side of the UVW phases. 0: V_63_MILLIVOLT 1: V_125_MILLIVOLT 2: V_187_MILLIVOLT 3: V_248_MILLIVOLT 4: V_312_MILLIVOLT 5: V_374_MILLIVOLT 6: V_434_MILLIVOLT 7: V_504_MILLIVOLT 8: V_705_MILLIVOLT 9: V_940_MILLIVOLT 10: V_1180_MILLIVOLT 11: V_1410_MILLIVOLT 12: V_1650_MILLIVOLT 13: V_1880_MILLIVOLT 14: V_2110_MILLIVOLT 15: V_2350_MILLIVOLT	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
260	OVERCURRENT_PROTECTION_Y2_LOW_SIDE_THRESHOLD Overcurrent protection threshold for the low side of the Y2 phase (uses second list if OVERCURRENT_PROTECTION_Y2_LOW_SIDE_USE_VDS= true). 0: V_80_OR_63_MILLIVOLT 1: V_165_OR_125_MILLIVOLT 2: V_250_OR_187_MILLIVOLT 3: V_330_OR_248_MILLIVOLT 4: V_415_OR_312_MILLIVOLT 5: V_500_OR_374_MILLIVOLT 6: V_582_OR_434_MILLIVOLT 7: V_660_OR_504_MILLIVOLT 8: V_125_OR_705_MILLIVOLT 9: V_250_OR_940_MILLIVOLT 10: V_375_OR_1180_MILLIVOLT 11: V_500_OR_1410_MILLIVOLT 12: V_625_OR_1650_MILLIVOLT 13: V_750_OR_1880_MILLIVOLT 14: V_875_OR_2110_MILLIVOLT 15: V_1000_OR_2350_MILLIVOLT	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
261	OVERCURRENT_PROTECTION_Y2_HIGH_SIDE_THRESHOLD Overcurrent protection threshold for the high side of the Y2 phase. 0: V_63_MILLIVOLT 1: V_125_MILLIVOLT 2: V_187_MILLIVOLT 3: V_248_MILLIVOLT 4: V_312_MILLIVOLT 5: V_374_MILLIVOLT 6: V_434_MILLIVOLT 7: V_504_MILLIVOLT 8: V_705_MILLIVOLT 9: V_940_MILLIVOLT 10: V_1180_MILLIVOLT 11: V_1410_MILLIVOLT 12: V_1650_MILLIVOLT 13: V_1880_MILLIVOLT 14: V_2110_MILLIVOLT 15: V_2350_MILLIVOLT	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
262	OVERCURRENT_PROTECTION_UVW_LOW_SIDE_BLANKING Overcurrent protection blanking time for the low side of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	2	RWE
263	OVERCURRENT_PROTECTION_UVW_HIGH_SIDE_BLANKING Overcurrent protection blanking time for the high side of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	2	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
264	OVERCURRENT_PROTECTION_Y2_LOW_SIDE_BLANKING Overcurrent protection blanking time for the low side of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	2	RWE
265	OVERCURRENT_PROTECTION_Y2_HIGH_SIDE_BLANKING Overcurrent protection blanking time for the high side of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	2	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
266	OVERCURRENT_PROTECTION_UVW_LOW_SIDE_DEGLITCH Overcurrent protection deglitch time for the low side of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	6	RWE
267	OVERCURRENT_PROTECTION_UVW_HIGH_SIDE_DEGLITCH Overcurrent protection deglitch time for the high side of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	6	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
268	OVERCURRENT_PROTECTION_Y2_LOW_SIDE_DEGLITCH Overcurrent protection deglitch time for the low side of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	6	RWE
269	OVERCURRENT_PROTECTION_Y2_HIGH_SIDE_DEGLITCH Overcurrent protection deglitch time for the high side of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	6	RWE
270	OVERCURRENT_PROTECTION_UVW_LOW_SIDE_USE_VDS Use the VDS measurement for the overcurrent protection on the low side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
271	OVERCURRENT_PROTECTION_Y2_LOW_SIDE_USE_VDS Use the VDS measurement for the overcurrent protection on the low side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE

Undervoltage Events (UVLO)

Undervoltage lockout (UVLO) protects the gate driver and power transistors from operating at insufficient voltage levels. Three different types of UVLO are implemented.

The VS UVLO detects an insufficient motor voltage. The threshold is set using another parameter listed in *Table 22*. If the fault occurs and the protection is enabled, the gate driver is disabled.

The VDRV UVLO detects an insufficient gate driver voltage. The threshold for the VDRV UVLO is fixed at a VDRV UVLO Threshold. If the fault occurs and the protection is enabled, the gate driver is disabled.

The BST UVLOs detect an insufficient voltage on any of the connected bootstrap capacitors. If the fault occurs and the protection is enabled, the respective channel is disabled. It is the user's responsibility to disable the other channels if desired.

All related parameters are listed in *Table 22*.

Table 22. Undervoltage protection parameter

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
250	UNDERVOLTAGE_PROTECTION_SUPPLY_LEVEL Undervoltage protection level for VS (Supply voltage). 0 disables the comparator. 1-16 are mapped to 0-15 HW values, with the comparator enabled.	0 … 16	0	RWE
251	UNDERVOLTAGE_PROTECTION_VDRV_ENABLE Enable the undervoltage protection for VDRV (Driver voltage). False: DISABLED True: ENABLED	0, 1	1	RWE
252	UNDERVOLTAGE_PROTECTION_BST_UVW_ENABLE Enable the undervoltage protection on the bootstrap capacitor of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
253	UNDERVOLTAGE_PROTECTION_BST_Y2_ENABLE Enable the undervoltage protection on the bootstrap capacitor of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE

Gate Short Protection (VGS)

The gate driver also includes protection against gate-to-source voltage shorts. This is achieved by monitoring the voltage between the gate and source terminals of the power transistors. If a short circuit is detected, the gate driver is disabled to prevent damage. The VGS protection parameters, such as deglitch and blanking times, can be configured through the parameters listed in *Table 23*. The deglitch time determines the minimum duration of a short-circuit condition before it is recognized as a fault. The blanking time prevents the VGS fault detection after a charge and discharge cycle to filter out any transient spikes or noise. The gate driver can be configured to automatically retry enabling the channel after a VGS fault. The number of retries is specified in the parameters listed in *Table 40*. If the fault persists after the retries, the gate driver remains disabled until the fault is cleared.

Table 23. Gate short protection parameters

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
272	VGS_SHORT_ON_PROTECTION_UVW_LOW_SIDE_ENABLE Enable the gate-source short protection for the ON transition of the low side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
273	VGS_SHORT_OFF_PROTECTION_UVW_LOW_SIDE_ENABLE Enable the gate-source short protection for the OFF transition of the low side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
274	VGS_SHORT_ON_PROTECTION_UVW_HIGH_SIDE_ENABLE Enable the gate-source short protection for the ON transition of the high side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
275	VGS_SHORT_OFF_PROTECTION_UVW_HIGH_SIDE_ENABLE Enable the gate-source short protection for the OFF transition of the high side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
276	VGS_SHORT_ON_PROTECTION_Y2_LOW_SIDE_ENABLE Enable the gate-source short protection for the ON transition of the low side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
277	VGS_SHORT_OFF_PROTECTION_Y2_LOW_SIDE_ENABLE Enable the gate-source short protection for the OFF transition of the low side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
278	VGS_SHORT_ON_PROTECTION_Y2_HIGH_SIDE_ENABLE Enable the gate-source short protection for the ON transition of the high side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
279	VGS_SHORT_OFF_PROTECTION_Y2_HIGH_SIDE_ENABLE Enable the gate-source short protection for the OFF transition of the high side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
280	VGS_SHORT_PROTECTION_UVW_BLANKING Gate-source short protection blanking time for the low and high sides of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC	0, 1, 2, 3	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
281	VGS_SHORT_PROTECTION_Y2_BLANKING Gate-source short protection blanking time for the low and high sides of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC	0, 1, 2, 3	1	RWE
282	VGS_SHORT_PROTECTION_UVW_DEGLITCH Gate-source short protection deglitch time for the low and high sides of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	1	RWE
283	VGS_SHORT_PROTECTION_Y2_DEGLITCH Gate-source short protection deglitch time for low and high sides of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	1	RWE

MOTOR CONFIGURATION

Depending on the motor, some basic parameters must be specified as one of the first steps during configuration. The setup differs depending on the motor used. It is also critical to configure the current measurement ADCs based on the selected motor type and the current shunt resistors used. The PWM frequency should also be adapted to the setup. A detailed description on the PWM setup is described in section *PWM Frequency Configuration*.

Motor Type Setup

Some motor settings depend on the motor type. The type of motor that is connected must be configured using the parameter MOTOR_TYPE. All motors can be inverted in their turning direction by setting the MOTOR_DIRECTION parameter.

DC

If the DC motor is selected, terminals UX1 and VX2 of the power stage are used to control the motor. No further motor specific configuration is needed for DC motors. For DC motors only, the torque values are relevant, and all flux-related values have no influence. To specify a maximum motor current, set the parameter MAX_TORQUE.

Stepper

For stepper motors, all four terminals are used. UX1 and VX2 control the first phase named X, while terminals WY1 and Y2 are used to control the 2nd phase named Y. The number of pole pairs is a motor specific parameter that must be set through the parameter MOTOR_POLE_PAIRS. It is necessary to calculate mechanical shaft angles out of the feedback signals. It can be calculated by dividing 90 degrees by the step angle.

The phase current can be assembled out of the torque and flux values. The maximum values allowed to be applied to the motor can be defined by the parameters MAX_TORQUE and MAX_FLUX and should be configured according to the maximum motor current. Unless the field weakening feature is used only torque or flux current is applied to the motor at a time. More on the modes of operation and the applied currents in the section *Commutation Modes*.

$$I_{PH} = \sqrt{{I_T}^2 + {I_F}^2}$$

BLDC

If the BLDC motor is selected terminals UX1, VX2, and WY1 are used to control the motor phases U, V, and W, respectively. The number of pole pairs is a motor specific parameter that must be set through the parameter MOTOR_POLE_PAIRS. It is necessary to calculate mechanical shaft angles out of the feedback signals.

The phase current can be assembled out of the torque and flux values. The maximum values allowed to be applied to the motor can be defined by the parameters MAX_TORQUE and MAX_FLUX and should be configured according to the maximum motor current. When the field weakening feature is not active only torque or flux current is applied to the motor at a time. More on the modes of operation and the applied currents in the section *Commutation Modes*.

$$I_{PH} = \sqrt{{I_T}^2 + {I_F}^2}$$

ADC Setup for Motor Current Measurement

It is critical to configure the motor current measurement correctly depending on the motor and system configuration. This includes the current sense amplifiers, ADC scaling, and inversion.

The CSAs must be configured for the system. The gain of the CSA can be adjusted using the parameters CSA_GAIN_ADC_IO_TO_ADC_I2 and CSA_GAIN_ADC_I3. If the CSA bypass is selected and an external CSA is used, the input is inverted. Therefore, the ADC inversion setting, described in the next paragraph, must be adjusted accordingly.

It is crucial to configure the ADCs for the motor phases correctly. This includes mapping the correct ADC to the corresponding motor phase and specifying whether the ADC signals need to be inverted. The current measurements of the ADCs must be mapped to the correct motor phase using the phase mapping parameters. Inverting the ADCs can be done by setting the ADC-specific inverted values. It is essential that the ADC values are inverted correctly to ensure proper behavior. *Table 24* shows the default mapping for all motor types with enabled internal CSA. To verify if the ADCs are mapped and inverted correctly, a motor can be slowly turned in open-loop voltage mode. The corresponding phase voltages and currents should be in phase.

During startup, the ADCs are automatically offset calibrated. A successful calibration is indicated by the ADC_OFFSET_CALIBRATED flag. If this flag is cleared, the ADCs are recalibrated as soon as possible. For calibration to be possible, the commutation mode must be off. The user must also ensure that the motor is not rotating during calibration. The parameters ADC_I0_SCALE to ADC_I3_SCALE provide the scaling factors for the ADCs to account for potential tolerances in the shunt resistors.

The torque and flux values can be used in real-world units if the current scaling value is set correctly. It can be calculated from the resistance of the current shunt resistor and the CSA gain using the formulas below. The current can be calculated as root mean squared (RMS) or peak (P). For DC motors peak must be used. The calculated value must be written to the parameter CURRENT_SCALING_FACTOR. If the correctly calculated factor is written to the TMC9660 Parameter Mode, the target torque and flux as well as the actual torque and flux are in mA.

$$Factor_{mA\,RMS} = \frac{1024 \times 2.5 \times 1000}{\sqrt{2} \times (2^{16} - 1) \times G_{CSA} \times R_{Shunt}} = \frac{27.62}{G_{CSA} \times R_{Shunt}}$$

$$Factor_{mA\,P} = \frac{1024 \times 2.5 \times 1000}{(2^{16} - 1) \times G_{CSA} \times R_{Shunt}} = \frac{39.06}{G_{CSA} \times R_{Shunt}}$$

Table 24. Standard ADC inversion table for usage with internal CSA

MOTOR	UX1	VX2	WY1	Y2
DC	Inverted	Non inverted	-	-
BLDC	Inverted	Inverted	Inverted	-
Stepper	Inverted	Non inverted	Inverted	Non inverted

Table 25. ADC configuration parameters

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
12	ADC_SHUNT_TYPE Shunt type used for ADC measurements. 0: INLINE_UVW 1: INLINE_VW 2: INLINE_UW 3: INLINE_UV 4: BOTTOM_SHUNTS	0, 1, 2, 3, 4	4	RWE
13	ADC_I0_RAW Raw ADC measurement of the ADC I0 shunt.	-32768 ... 32767	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
14	ADC_I1_RAW Raw ADC measurement of the ADC I1 shunt.	-32768 ... 32767	0	R
15	ADC_I2_RAW Raw ADC measurement of the ADC I2 shunt.	-32768 ... 32767	0	R
16	ADC_I3_RAW Raw ADC measurement of the ADC I3 shunt.	-32768 ... 32767	0	R

PWM Frequency Configuration

The parameter MOTOR_PWM_FREQUENCY allows you to set the PWM frequency between 10kHz and 100kHz. This frequency is derived from a 120MHz clock. There are also multiple other frequencies derived from the PWM frequency, further described in the next section.

The PWM frequency should match the motor's winding time constant T_S, calculated as $T_S = L_S/R_S$. The PWM period $1/f_{PWM}$ should be at least five times smaller than T_S. To ensure optimal performance, it is recommended to start with specific loop frequencies based on the motor type. For a typical high inductance stepper motor, a 20kHz loop frequency is suggested. For a typical BLDC motor, a 25kHz loop frequency is recommended. For fast-spinning BLDC motors (n > 10,000rpm), loop frequencies like 50kHz and 100kHz should be used. Using a PWM frequency that is too low can cause high current ripple and low efficiency, while a frequency that is too high can result in higher switching losses. Additionally, PWM frequencies below 20kHz might create audible switching noise and should be avoided.

Control Loop Frequencies

The PWM frequency also determines the speed of multiple other critical system loops. The torque and flux controllers operate at PWM frequency. The TMC9660 Parameter Mode specific features, fault handling, IIT and the GDRV setup and monitoring are handled at a down sampled PWM frequency. The frequency can be calculated by dividing the PWM frequency with a division factor depending on the frequency shown in *Table 26*.

The velocity control loop can be slowed down by a frequency divider configurable through the parameter VELOCITY_LOOP_DOWNSAMPLING. This includes the complete structure shown in *Figure 4*. The position loop frequency can be further decreased by configuring the POSTION_LOOP_DOWNSAMPLING parameter. Since the integrator speeds depend on the PWM frequency and the down sampling factors the control loop parameters change depending on the frequency configuration. *Figure 4* illustrates the structure.

Some management functionalities in the system are run at a 1kHz frequency.

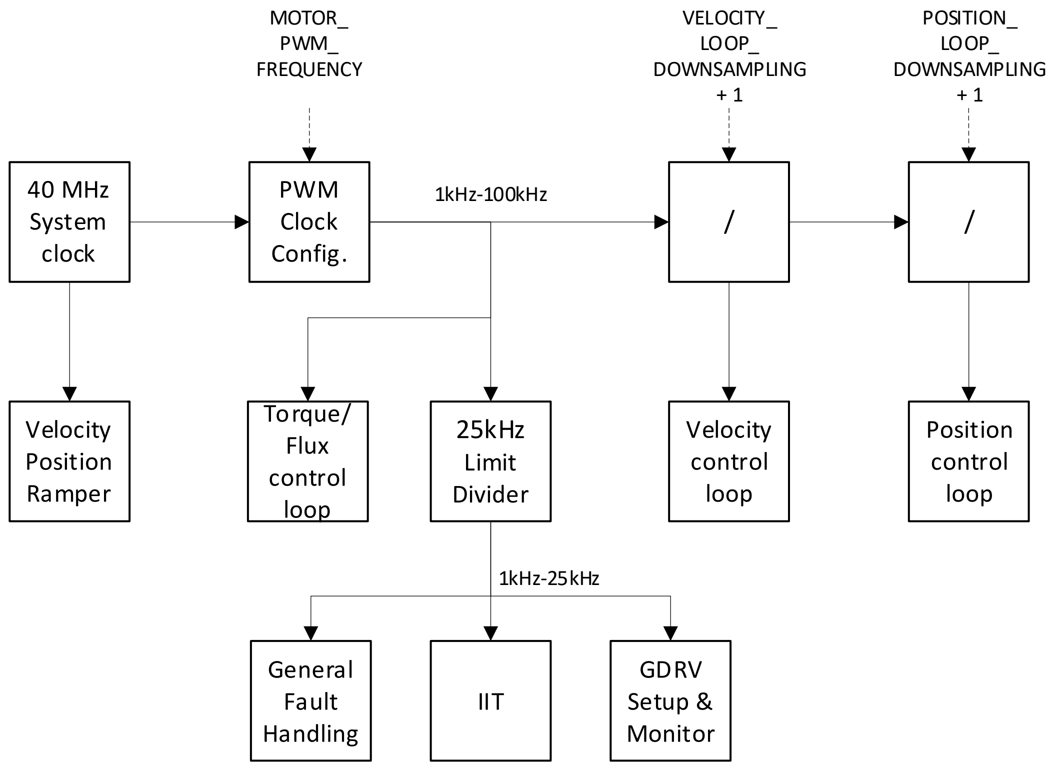

Figure 4. System frequency diagram

Table 26. Process frequency depending on PWM frequency

PWM FREQUENCY	DIVIDER	PROCESS FREQUENCY
1000Hz - 25000Hz	1	1000Hz – 25000Hz
25001Hz – 50000Hz	2	12500Hz-25000Hz
50001Hz – 75000Hz	3	16667Hz-25000Hz
75001Hz – 100000Hz	4	18750Hz-25000Hz

PWM Switching Scheme

The parameter PWM_SWITCHING_SCHEME is used to change the PWM pattern generation to enable full voltage utilization for BLDC motors. Depending on the motor the recommended configuration changes.

BLDC

For three phase BLDC motors four different PWM modulation modes are available. *Figure 5* shows the duty cycle depending on the motor angles for the different PWM switching schemes. By default, a space vector modulation is applied. This modulation helps to generate higher peak voltage effectively. The third option is the flat bottom modulation. Depending on the specified PWM scheme the maximum duty cycle value changes. The maximum values depending on the mode can be found in *Table 27*.

While the space vector mode results in a pattern where high-side and low-side switches are equally loaded, the Flat Bottom option puts more load on low-side switches for an extended current measurement time window.

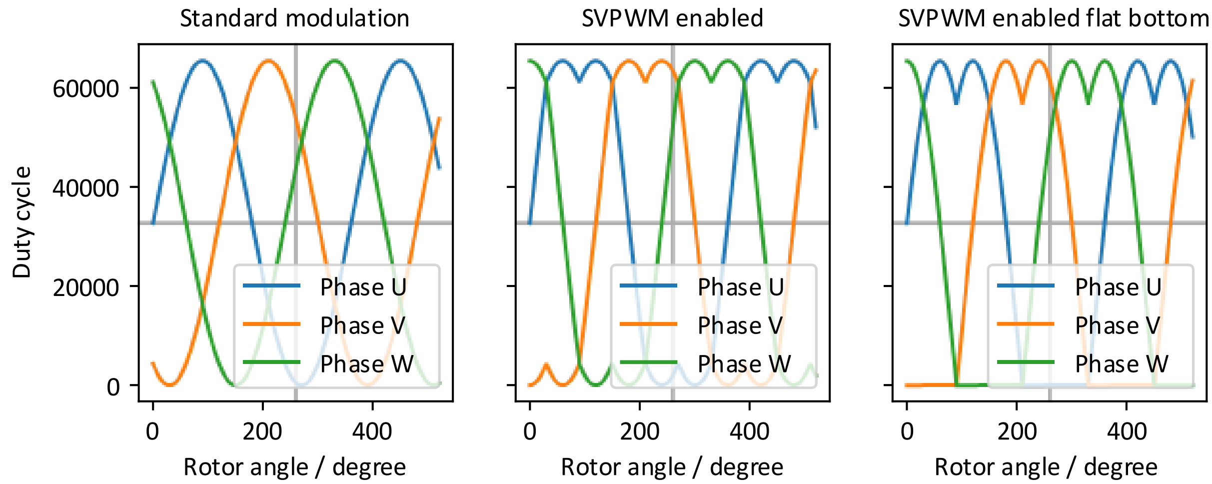

Figure 5. PWM modes for three-phase BLDC motor

Table 27. Maximum duty cycle value based on PWM switching scheme

MOTOR TYPE	PWM_SWITCHING_SCHEME	MAXIMUM OUTPUT VOLTAGE	EFF. MAXIMUM DUTY CYCLE [%]
BLDC	0: Standard	16383	86%
BLDC	1: SVPWM / 2: Flat Bottom	18900	100%
Stepper/DC	--	16383	100%

STEPPER

For stepper motors only a standard sinus modulation and a flat bottom modulation are available shown in *Figure 6*
In both modes the maximum output voltage limit is 16383.

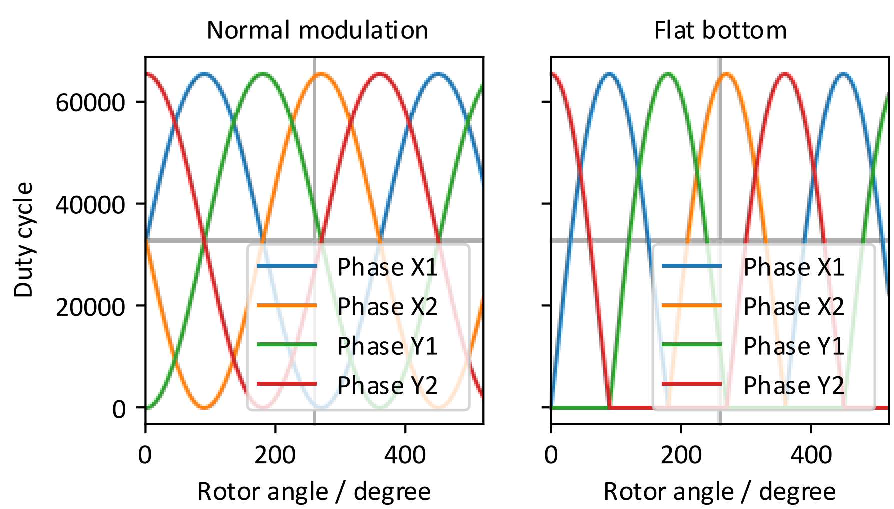

Figure 6. PWM modes for stepper motors

DC

For DC motors same PWM modulations as for steppers are available. The maximum output voltage limit is 16383.

Table 28. PWM parameters setup parameters

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
3	MOTOR_PWM_FREQUENCY [Hz] Set the frequency of the motor PWM.	10000 ... 100000	25000	RWE
8	PWM_SWITCHING_SCHEME PWM switching scheme. 0: STANDARD 1: SVPWM 2: FLAT_BOTTOM	0, 1, 2	1	RWE

FEEDBACK SENSOR CONFIGURATION

To enable closed-loop operation with the FOC engine, correct feedback configuration is critical. Supported feedback types, in the TMC9660 Parameter Mode, include ABN, digital hall, and SPI-Encoder. Different sensors can be used for angle determination needed for FOC commutation, velocity, and position measurement and control. ABN2 cannot be used for the FOC angle calculation, only for velocity and position feedback. These sensors must be configured as described below before use. All sensors must be configured prior to setup in the boot configuration.

ABN Encoder

To use an ABN encoder the steps and inversion must be configured. The system automatically initializes the sensor as soon as needed. To check ABN setup the motor can be turned in one of the open loop commutation modes with constant velocity. To validate the correct setup the parameters OPENLOOP_ANGLE and ABN_1_PHI_E can be monitored in parallel. Both values must increment by the same rate.

ABN Initialization Methods

Because the ABN signal does not have absolute position information, the goal of ABN initialization is to align the internal ABN encoder angle with the absolute position of a motor's rotor. There are four initialization methods available, each with its advantages and disadvantages. The mode can be selected with the parameter ABN_1_INIT_METHOD. The current state of initialization can be read out using the parameter ABN_1_INIT_STATE.

Note: ABN_1_INIT_STATE DONE signals that the initialization process is completed, it does not guarantee that initialization was successful. A successful initialization must be verified manually by starting the motor afterwards and observing the behavior.

0. Forced phi_e zero with active swing
    - Forces the motors rotor into a phi_e zero position using the OPENLOOP_CURRENT, and then zeros the encoder angle. It uses active swinging around phi_e zero to prevent stall at exactly 180 degrees offset.
    - Pro: No need for digital hall.
    - Con: Fails if the OPENLOOP_CURRENT is not high enough to move the motor.
1. Forced phi_e 90/zero
    - Forces the motors rotor into a phi_e 90-degree position for a moment using the OPENLOOP_CURRENT and then into a phi_e zero position where the internal encoder angle is zeroed. This is very similar to the previous method and also prevents a stall at a 180 degree offset.
    - Pro: No need for digital hall.
    - Con: Fails if the OPENLOOP_CURRENT is not high enough to move the motor.
2. Use hall
    - Rotates the motor in digital hall commutation until there is a transition in the hall signal, and this position is then used to align the ABN angle with the digital hall angle.
    - It is required that digital hall is configured correctly beforehand.
    - Pro: More smooth startup behavior. And, the method is independent of OPENLOOP_CURRENT.
    - Con: Needs a motor with integrated digital hall.
3. Use N-channel offset
    - Rotates the motor in open loop commutation and as soon as there is an N-channel event the ABN_1_N_CHANNEL_PHI_E_OFFSET is applied to internal ABN angle for the alignment.

- Pro: More reproducible alignment compared to the two forced phi_e methods.
- Con: Needs the N-channel offset to be determined first either by one of the two forced phi_e methods or manually.
- Con: Fails if the OPENLOOP_CURRENT is not high enough to move the motor.

The initialization method is executed automatically on the transition to ABN encoder commutation.

Table 29. Parameters for ABN encoder feedback

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
89	ABN_1_PHI_E Phi_e calculated from abn feedback.	-32768 ... 32767	0	R
90	ABN_1_STEPS ABN 1 encoder steps per rotation (CPR).	0 ... 16777215	65536	RWE
91	ABN_1_DIRECTION ABN 1 encoder rotation direction. False: NOT_INVERTED True: INVERTED	0, 1	0	RWE
92	ABN_1_INIT_METHOD Select an ABN encoder initialization method that fits best to your motor's sensors. 0: FORCED_PHI_E_ZERO_WITH_ACTIVE_SWING Forces the rotor into phi_e zero using the open loop current but actively swings the rotor. 1: FORCED_PHI_E_90_ZERO Forces the rotor into phi_e 90-degree position and then into zero position using the open loop current. 2: USE_HALL Turns the motor slightly in hall commutation mode until a hall signal change gives a new absolute position, which then the ABN phi_e is aligned to. 3: USE_N_CHANNEL_OFFSET Turns the motor slightly in open loop commutation mode until a N-channel is reached and gives an absolut position, which then the ABN phi_e is aligned to.	0, 1, 2, 3	0	RWE
93	ABN_1_INIT_STATE Actual state of ABN encoder initialization. 0: IDLE 1: BUSY 2: WAIT 3: DONE	0, 1, 2, 3	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
94	ABN_1_INIT_DELAY [ms] When one of the "Forced phi_e" initialization methods is used, this value defines the wait time until the phi_e ABN angle is set to zero. This parameter should be set in a way, that the motor has stopped mechanical oscillations after the specified time.	1000 ... 10000	1000	RWE
95	ABN_1_INIT_VELOCITY Init velocity for ABN encoder initialization with N-channel offset.	-200000 ... 200000	5	RWE
96	ABN_1_N_CHANNEL_PHI_E_OFFSET Offset between phi_e zero and the ABN encoders index pulse position. This value is updated asynchronously on any ABN initialization other than the "Use-N channel offset" method. The value can then be used for the "Use N-channel offset" based initialization.	-32768 ... 32767	0	RWE
97	ABN_1_N_CHANNEL_INVERTED ABN 1 encoder N-channel is inverted. False: ACTIVE_HIGH True: ACTIVE_LOW	0, 1	0	RWE
98	ABN_1_N_CHANNEL_FILTERING ABN 1 encoder N-channel filtering. Useful for imprecise encoders with the index pulses lasting multiple A/B steps. 0: FILTERING_OFF 1: N_EVENT_ON_A_HIGH_B_HIGH 2: N_EVENT_ON_A_HIGH_B_LOW 3: N_EVENT_ON_A_LOW_B_HIGH 4: N_EVENT_ON_A_LOW_B_LOW	0, 1, 2, 3, 4	0	RWE
99	ABN_1_CLEAR_ON_NEXT_NULL Clear the actual position on the next ABN 1 encoder N-channel event. False: DISABLED True: ENABLED	0, 1	0	RW
100	ABN_1_VALUE Raw ABN encoder internal counter value.	0 ... 16777215	0	R

Hall Encoder

To use a Hall feedback the HALL_SECTOR_OFFSET and HALL_INVERT_DIRECTION are the minimum set of parameters that must be configured. To validate the correct setup the parameters OPENLOOP_ANGLE and HALL_PHI_E can be monitored in parallel while turning the motor in an open loop commutation mode. Both values must be synchronized.

Filtering

Hall sensor signals can be digitally filtered. The parameter HALL_FILTER_LENGTH defines the length of the filter. Each hall input signal must be stable for the number of clock cycles specified by this register value before new values are accepted.

Hall Offset Compensation

As typical hall sensors can have quite high mounting tolerances the hall sensor position can be calibrated. Therefore, the offset positions can be written to the respective HALL_POSITION_X_OFFSET parameters. In these parameters, the exact position of the sensor in regard to the electrical angle PHI_E is saved.

Extrapolation

The discrete hall sensor positions can be extrapolated to generate a higher resolution position signal. The extrapolation can be enabled when the corresponding bit is set, and the velocity is higher than 60rpm electrical. It deactivates automatically below 60rpm. It is recommended to calibrate hall sensor offsets to achieve maximum performance.

Hall Error

A HALL_ERROR flag is triggered within the GENERAL_ERROR_FLAGS if either all Hall input signals are High or Low. This indicates that no Hall encoder is connected or damaged. While bootup the flag gets triggered and is therefore always on. A manual reset is needed to detect following errors. The error is not processed automatically but requires external handling.

Table 30. Parameters for HALL encoder

Nr.	Parameter/Description	Range	Default	R/W
74	HALL_PHI_E Phi_e calculated from hall feedback.	-32768 ... 32767	0	R
75	HALL_SECTOR_OFFSET Hall sensor 60-degree/sector offset composed of 120-degree offset (order) and 180-degree offset (polarity). 0: DEG_0 1: DEG_60 2: DEG_120 3: DEG_180 4: DEG_240 5: DEG_300	0, 1, 2, 3, 4, 5	0	RWE
76	HALL_FILTER_LENGTH Filter length of the hall sensor input signal filters.	0 ... 255	0	RWE
77	HALL_POSITION_0_OFFSET Hall offset compensation for 0-degree hall position.	-32768 ... 32767	0	RWE
78	HALL_POSITION_60_OFFSET Hall offset compensation for 60-degree hall position.	-32768 ... 32767	10922	RWE
79	HALL_POSITION_120_OFFSET Hall offset compensation for 120-degree hall position.	-32768 ... 32767	21845	RWE

Nr.	Parameter/Description	Range	Default	R/W
80	HALL_POSITION_180_OFFSET Hall offset compensation for 180-degree hall position.	-32768 ... 32767	-32768	RWE
81	HALL_POSITION_240_OFFSET Hall offset compensation for 240-degree hall position.	-32768 ... 32767	-21846	RWE
82	HALL_POSITION_300_OFFSET Hall offset compensation for 300-degree hall position.	-32768 ... 32767	-10923	RWE
83	HALL_INVERT_DIRECTION Inverts the hall angle direction. False: NOT_INVERTED True: INVERTED	0, 1	0	RWE
84	HALL_EXTRAPOLATION_ENABLE Allows the activation of the hall extrapolation to generate a higher resolution position signal. When enabled, the extrapolation is only active at speeds higher than 60rpm. False: DISABLED True: ENABLED	0, 1	0	RWE
85	HALL_PHI_E_OFFSET Use this parameter to compensate hall sensor mounting tolerances.	-32768 ... 32767	0	RWE

SPI Encoder

The TMC9660 Parameter Mode provides generic support for SPI-Encoders. The interface enables the use of a position sensor with SPI interface for FOC commutation, velocity, and position feedback. The generic approach supports a large variety of sensors. The parameters must be configured sensor specific depending on the parameters in *Table 31*.

Single Transfer Mode

The single transfer mode is utilized for tunneling data to the sensor, allowing configuration commands to be sent before the sensor is used. To send an SPI frame, you need to set the SPI_ENCODER_MAIN_TRANSFER_CMD_SIZE and write the TX data to the SPI_ENCODER_TRANSFER_DATA_X parameters. A single transfer is triggered by setting SPI_ENCODER_TRANSFER to 1. Optionally, you can read the RX data from the SPI_ENCODER_TRANSFER_DATA_X parameters. See *Figure 7* to understand how the SPI_ENCODER_TRANSFER_DATA_3_0 maps into an SPI frame.

When the SPI_ENCODER_MAIN_TRANSFER_CMD_SIZE is set to 2, note that the hexadecimal integer value 0xD3D2D1D0 inside SPI_ENCODER_TRANSFER_DATA_3_0 is mirrored compared to the timing diagram representation. On the bit level, the most significant bit (MSB) is transmitted first, as detailed in *Figure 8*.

If more than 4 bytes need to be transferred, SPI_ENCODER_TRANSFER_DATA_7_4 to SPI_ENCODER_TRANSFER_DATA_15_12 can be used to transfer up to 16 bytes at once. With a SPI_ENCODER_MAIN_TRANSFER_CMD_SIZE of 5, the data maps into the SPI frame as shown in *Figure 9*.

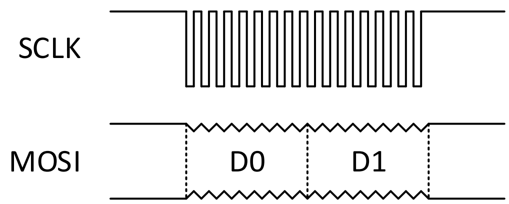

Figure 7. Timing with a SPI_ENCODER_MAIN_TRANSFER_CMD_SIZE of 2

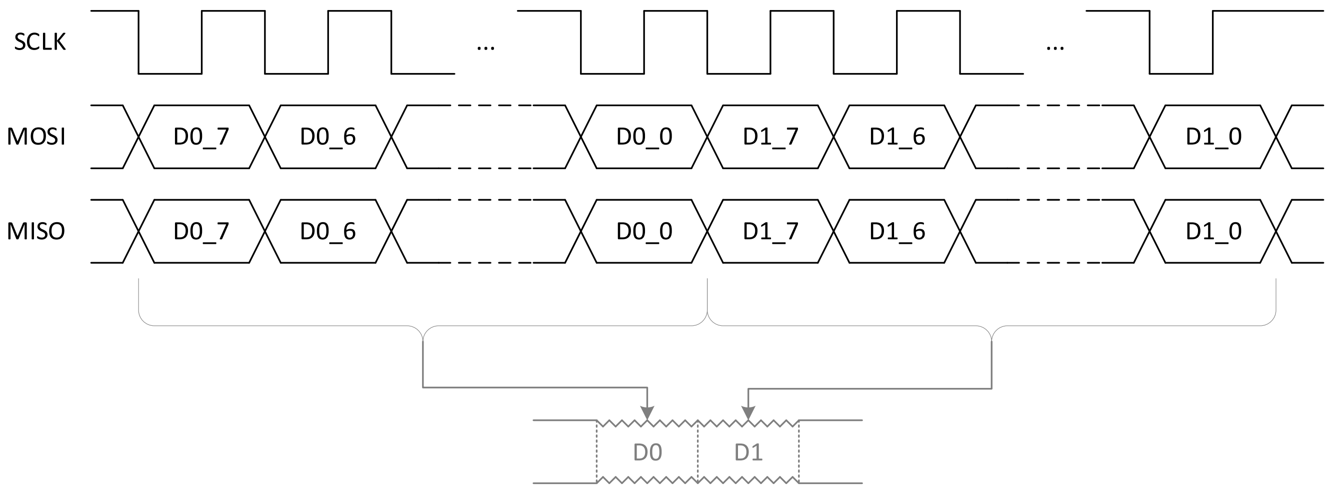

Figure 8. SPI frame bit level detail

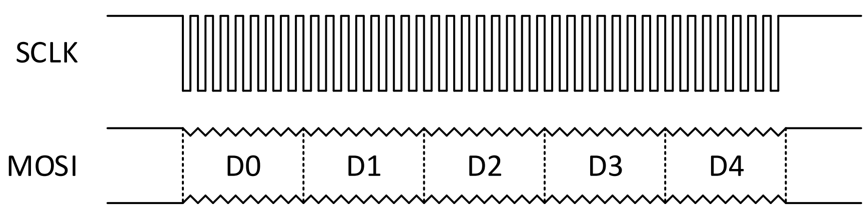

Figure 9. Timing with a SPI_ENCODER_MAIN_TRANSFER_CMD_SIZE of 5

Continuous Transfer Mode

Continuous transfer is used to continuously acquire a motor's rotor position for the FOC or to use the sensor's position as velocity or position feedback. To set up continuous transfer, you need to follow these steps:

First, set the SPI_ENCODER_MAIN_TRANSFER_CMD_SIZE to the request frame size and write the request frame data to the SPI_ENCODER_TRANSFER_DATA_X parameters. Then, set the SPI_ENCODER_POSITION_COUNTER_SHIFT to bit shift the position counter to the right and the SPI_ENCODER_POSITION_COUNTER_MASK to mask out the position. Enable continuous transfer by setting SPI_ENCODER_TRANSFER to 2.

After setting up continuous transfer, check if SPI_ENCODER_DIRECTION needs to be modified before using the SPI encoder for position feedback. If the SPI encoder position is used for commutation, consider updating the SPI_ENCODER_INITIALIZATION_METHOD.

Example without Shift

In this example, the position inside the response frame is left-aligned. Using the following settings:

- SPI_ENCODER_TRANSFER_DATA_3_0 = 0x0000FFFF
- SPI_ENCODER_MAIN_TRANSFER_CMD_SIZE = 2
- SPI_ENCODER_POSITION_COUNTER_SHIFT = 0
- SPI_ENCODER_POSITION_COUNTER_MASK = 0x00003FFF

With this configuration, the SPI transfer appears as shown in *Figure 10*.

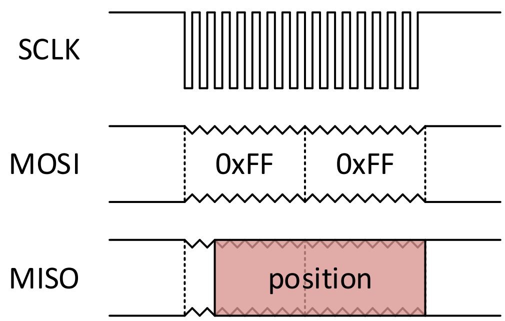

Figure 10. Example SPI frame without shift

Example with Shift

In this example, the position is shifted by 10 bits to the left. Using the following settings:

- SPI_ENCODER_TRANSFER_DATA_3_0 = 0x000000A6
- SPI_ENCODER_MAIN_TRANSFER_CMD_SIZE = 4
- SPI_ENCODER_POSITION_COUNTER_SHIFT = 10
- SPI_ENCODER_POSITION_COUNTER_MASK = 0x0003FFF

With this configuration, the SPI transfer appears as shown in *Figure 11*.

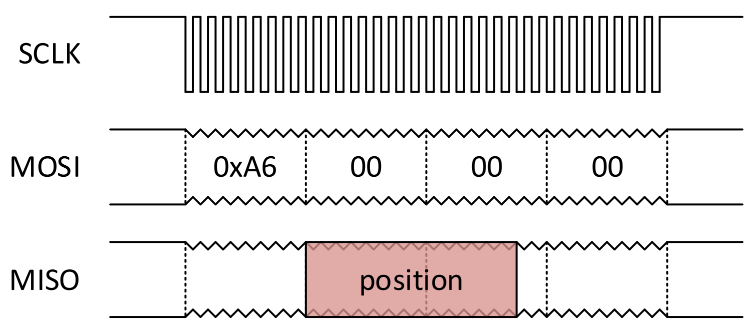

Figure 11. Example SPI frame with shift

SPI Encoder Initialization Method

When an SPI encoder is used for commutation, the SPI_ENCODER_INITIALIZATION_METHOD parameter is used to select the desired initialization method. For the “Forced PHI_E” methods, the OPENLOOP_CURRENT is used to force the motor into a known position. Ensure this current is high enough to provide a smooth motor movement, ideally with no load on the motor during alignment. Especially BLDC-Motors tend to show increased cogging which requires higher currents for smooth rotations. The alignment starts as soon as the COMMUTATION_MODE is changed to “FOC (SPI enc)”.

If the “Use offset” method is selected, alignment is achieved using the SPI_ENCODER_OFFSET parameter. This offset must be determined first, for example, through one of the “Forced PHI_E” methods, as “Forced PHI_E” calculates the offset and writes it into SPI_ENCODER_OFFSET.

Lookup Table Correction Support

Hall effect-based rotor position sensors can have offset and gain errors. To compensate for these errors, a lookup table (LUT) based correction can be used. The LUT consists of 256 entries, each being an 8-bit signed integer correction value. Additionally, there is a common shift factor that allows the 8-bit value to be shifted to compensate for larger gain errors. With the shift factor, the maximum error that can be compensated ranges from -2048 to 2037.

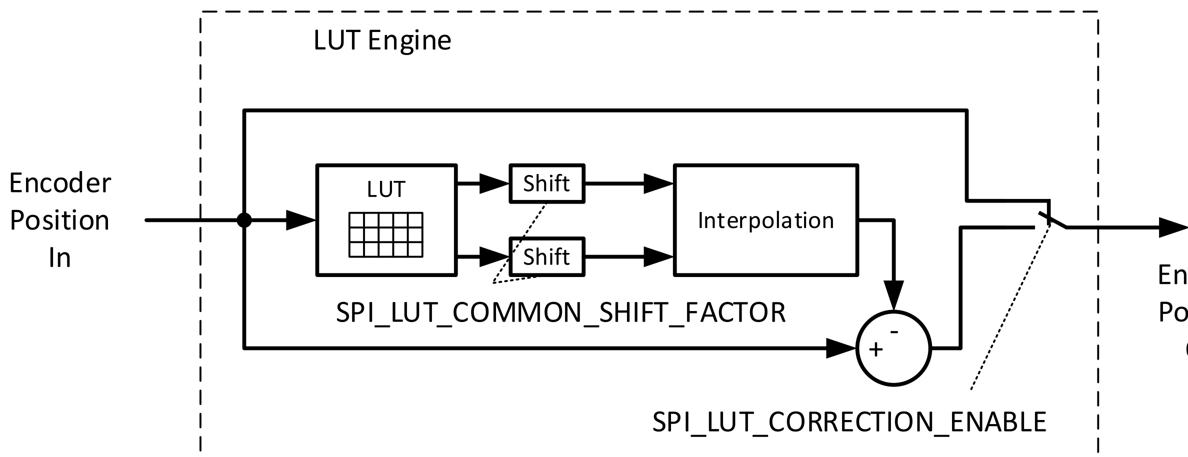

Figure 12. Basic LUT engine block diagram

LUT Table Creation

One method of creating the LUT is by rotating the motor slowly in open-loop commutation while no load is applied to the motor. During the rotation, the ACTUAL_POSITION and the SPI_ENCODER_POSITION_COUNTER_VALUE are sampled. Ensure that the LUT correction is disabled during this rotation.

LUT Upload and Nonvolatile Storage

The LUT is uploaded through the parameters SPI_LUT_ADDRESS_SELECT and SPI_LUT_DATA. Each 8-bit LUT entry is uploaded individually by first writing the index into SPI_LUT_ADDRESS_SELECT and then writing the 8-bit value into SPI_LUT_DATA. The LUT table is part of the parameter storage mechanism described in the section *Storing System Settings in External Memory*.

Table 31. Parameters for SPI encoder

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
181	SPI_ENCODE_CS_SETTLE_DELAY_TIME [ns] Add a delay from CS going low to first SCLK edge.	0 … 6375	0	RWE
182	SPI_ENCODER_CS_IDLE_DELAY_TIME [us] Extend CS idle time between SPI message frames.	0 … 102	0	RWE
183	SPI_ENCODER_MAIN_TRANSFER_CMD_SIZE Size of the first SPI transfer frame.	1 … 16	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
184	SPI_ENCODER_SECONDARY_TRANSFER_CMD_SIZE Size of the optional secondary SPI transfer frame. If set to zero, no secondary SPI transfer.	0 … 15	0	RWE
185	SPI_ENCODER_TRANSFER_DATA_3_0 Used to set the transmit data and read out the received data.	0 … 4294967295	0	RWE
186	SPI_ENCODER_TRANSFER_DATA_7_4 Used to set the transmit data and read out the received data.	0 … 4294967295	0	RWE
187	SPI_ENCODER_TRANSFER_DATA_11_8 Used to set the transmit data and read out the received data.	0 … 4294967295	0	RWE
188	SPI_ENCODER_TRANSFER_DATA_15_12 Used to set the transmit data and read out the received data.	0 … 4294967295	0	RWE
189	SPI_ENCODER_TRANSFER SPI interface setting, polarity and phase. 0: OFF 1: TRIGGER_SINGLE_TRANSFER 2: CONTINUOUS_POSITION_COUNTER_READ	0, 1, 2	0	RWE
190	SPI_ENCODER_POSITION_COUNTER_MASK Mask to be used to collect the position counter value from the continuous received data.	0 … 4294967295	0	RWE
191	SPI_ENCODER_POSITION_COUNTER_SHIFT Right bit shift for the position counter value before mask is applied.	0 … 127	0	RWE
192	SPI_ENCODER_POSITION_COUNTER_VALUE Actual SPI encoder position value.	0 … 4294967295	0	R
193	SPI_ENCODER_COMMUTATION_ANGLE Actual absolute encoder angle value.	-32768 … 32767	0	R
194	SPI_ENCODER_INITIALIZATION_METHOD Select the used absolute encoder initialization mode 0: FORCED_PHI_E_ZERO_WITH_ACTIVE_SWING Forces the rotor into PHI_E zero using the open-loop current but actively swings the rotor. 1: FORCED_PHI_E_90_ZERO Forces the rotor into PHI_E 90 degree position and then into zero position using the open-loop current. 2: USE_OFFSET	0, 1, 2	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
195	SPI_ENCODER_DIRECTION SPI encoder direction. False: NOT_INVERTED True: INVERTED	0, 1	0	RWE
196	SPI_ENCODER_OFFSET This value represents the internal commutation offset. (0...max. encoder steps per rotation).	0 ... 4294967295	0	RWE
197	SPI_LUT_CORRECTION_ENABLE Enable the lookup table-based encoder correction. False: DISABLED True: ENABLED	0, 1	0	RWE
198	SPI_LUT_ADDRESS_SELECT Address to read or write the lookup table.	0 ... 255	0	RW
199	SPI_LUT_DATA Data to read or write to a lookup table address.	-128 ... 127	0	RW
201	SPI_LUT_COMMON_SHIFT_FACTOR All LUT table entries are multiplied with 2^SHIFT_FACTOR to compensate for larger erros if needed.	0 ... 4	0	RW

ABN 2 Encoder

The TMC9660 Parameter Mode supports a second ABN encoder for position and velocity feedback. This sensor can for example be mounted behind a gearbox. It can not be used to determine the electrical angle for commutation. The encoder needs similar configuration to the configuration of the primary ABN encoder. The second encoder can not be used for commutation. N-Channel is not supported. If used for velocity and position calculation the sensor generates a PHI_M signal. The parameter ABN_2_GEAR_RATIO can be used to sync the velocity to the velocity measured at the shaft.

Table 32. Parameters for ABN 2 encoder

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
174	ABN_2_STEPS ABN 2 encoder steps per rotation (CPR).	0 ... 16777215	1024	RWE
175	ABN_2_DIRECTION ABN 2 encoder rotation direction. False: NORMAL True: INVERTED	0, 1	0	RWE
176	ABN_2_GEAR_RATIO ABN 2 encoder gear ratio.	1 ... 255	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
177	ABN_2_ENABLE Enable the ABN 2 encoder. Disabling resets the counted steps. False: DISABLED True: ENABLED	0, 1	0	RWE
178	ABN_2_VALUE Raw ABN2 encoder internal counter value.	0 ... 4294967295	0	R

COMMUTATION MODES

To turn a motor the commutation mode must be selected using the parameter COMMUTATION_MODE. This specifies what feedback system is used to determine the motors shaft angle. It is also used to enable/disable the system or switching to a state with shorted motor coils. The subsections below list the commutation modes and explain their behavior.

System Off

This is the default state after power-on and reset events. For safety reasons, certain operations like changing the motor type won't work unless this mode is active.

The behavior of the PWM signals when this state is active is determined by the IDLE_MOTOR_PWM_BEHAVIOR parameter. The PWM signals are switched off if IDLE_MOTOR_PWM_BEHAVIOR is set to 1, leaving the motor electrically floating. This is the default behavior. The PWM stays active if IDLE_MOTOR_PWM_BEHAVIOR is set to 0.

Note: as described in the section *Erratum 3: Exiting to bootloader with ongoing motor commutation*, this state must be manually selected if exiting from parameter mode to the bootloader is needed.

System Off, Low-Side FETs On

This mode charges all low-side motor FETs. The motor coils are practically shorted to ground. This mode is not used during normal operation.

System Off, High-Side FETs On

This mode charges all high-side motor FETs. The motor coils are practically shorted to supply voltage. This mode is not used during normal operation.

FOC (Openloop, Voltage Mode)

This mode applies a constant duty cycle to the motor. The voltage is defined by the parameter OPENLOOP_VOLTAGE that specifies the PWM duty cycle. Thus, it is relative to the supply voltage. This mode is intended for initial setup purposes only. The current is not limited in this mode thus the open-loop voltage parameter must be configured with care.

Stepper/BLDC

For stepper and BLDC motors, the commutation angle is calculated by the internal hardware ramper block. After selecting the mode and specifying the OPENLOOP_VOLTAGE velocity target must be set using the parameter

TARGET_VELOCITY. Setting the velocity target applies the configured voltage that stays on even if the velocity target is set to zero. For more details on the velocity and ramper functionality, see the section *Velocity Mode*.

DC

For DC motors, no velocity input is needed or possible in this mode. To reverse the direction of the motor, use the MOTOR_DIRECTION parameter.

FOC (Openloop, Current Mode)

This mode applies a constant current to the motor. The correct ADC setup is critical for this and all following modes. The current is defined by the parameter OPENLOOP_CURRENT. The current is regulated by the flux control loop described in the section *Torque and Flux Control* For DC motors the torque loop is used.

Stepper/BLDC

For stepper and BLDC motors, the commutation angle is calculated by the internal hardware ramper block. The maximum current is additionally limited by the parameter MAX_FLUX. After selecting the mode and specifying the OPENLOOP_CURRENT a velocity target must be set using the parameter TARGET_VELOCITY. For more details on the velocity and ramper functionality, see the section *Velocity Mode*.

DC

For DC motors, no velocity input is needed or possible in this mode. The maximum current is additionally limited by the parameter MAX_TORQUE.

FOC (ABN), FOC (Hall Sensor), FOC (SPI Enc)

This mode uses sensor feedback to calculate the motor shaft position. The feedback source must be configured prior to activating this mode. In case of ABN selecting the commutation mode triggers an automatic encoder initialization. For more information, see the section *ABN Initialization Methods*.

Stepper/BLDC

For stepper and BLDC motors, the commutation angle is calculated from the selected feedback method. In this mode, torque, velocity, and position control are available. Only torque is applied to the motor and the flux value can be assumed zero unless field weakening is active. The setting MAX_TORQUE is applied to the motor limiting the current. If field weakening is active a flux component is added. Find more information on this in the section *Field Weakening and Flux Control*. If ABN is selected as feedback, the system tries to automatically initialize the sensor. For more information, see the section *ABN Encoder*.

DC

To use a DC motor with velocity or position control, the according commutation mode must also be selected. The setting MAX_TORQUE is applied to limit the motor current.

Table 33. Parameters to set up commutation mode

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
4	COMMUTATION_MODE Selected FOC operation mode depending on feedback used for commutation. 0: SYSTEM_OFF 1: SYSTEM_OFF_LOW_SIDE_FETS_ON 2: SYSTEM_OFF_HIGH_SIDE_FETS_ON 3: FOC_OPENLOOP_VOLTAGE_MODE 4: FOC_OPENLOOP_CURRENT_MODE 5: FOC_ABN 6: FOC_HALL_SENSOR 7: RESERVED 8: FOC_SPI_ENC	0, 1, 2, 3, 4, 5, 6, 7, 8	0	RW
9	IDLE_MOTOR_PWM_BEHAVIOR Configure if the PWM should be off (high-z) or on (all motor phases same voltage) in commutation mode "System Off". False: PWM_ON_WHEN_MOTOR_IDLE True: PWM_OFF_WHEN_MOTOR_IDLE	0, 1	1	RWE

TORQUE AND FLUX CONTROL

The fundamentals of motion control are controlling the motor currents. These can be separated into torque and flux values. The regulation is performed by the hardware torque and flux PI controllers.

Control Loop Configuration

The system controls the torque and flux value applied to a motor. The structure of the control loop is shown in *Figure 13*. The controllers must be configured. By default, the torque PI parameters are applied to torque and flux PI. They can however be separated by setting the parameter SEPARATE_TORQUE_FLUX_PI_PARAMTERS to true. To access the controller gains the parameters TORQUE_P, TORQUE_I, FLUX_P and FLUX_I exist. Using the parameter CURRENT_NORM_P and CURRENT_NORM_I the PI controllers' range can be extended by shifting the controller output by either 8 or 16 bits right. The setup of the biquad is described in the section *Biquad filter setup*.

For DC motors only, torque control is applicable. The flux control structure is inactive.

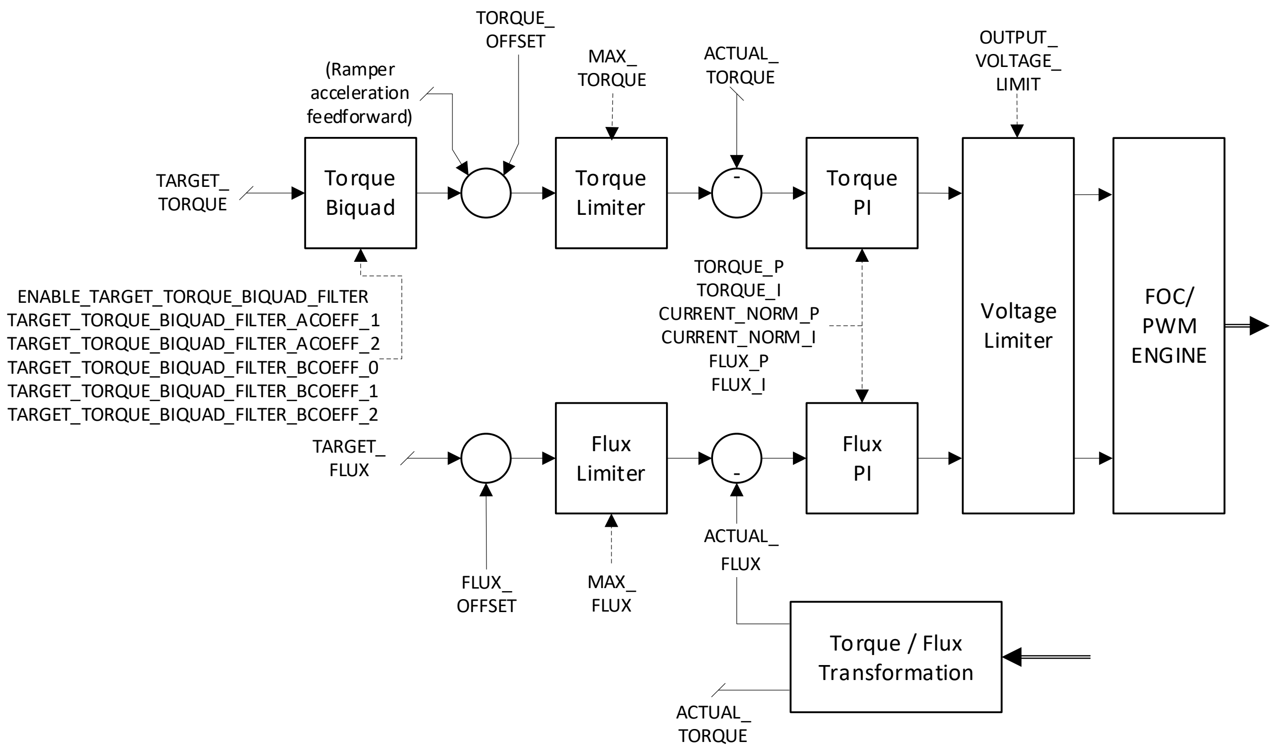

Figure 13. Torque and flux control loop

Drive a Motor in Torque Mode

The system can be operated in torque mode. In this mode of operation, a torque target is set through the parameter TARGET_TORQUE. To active the mode it is sufficient to write the TARGET_TORQUE parameter. Note that this mode is only available in commutation modes with sensor feedback. The status flag REGULATION_TORQUE indicates that torque mode is active. The system tries to regulate the applied torque to match the target. The actual value can be read using the parameter ACTUAL_TORQUE.

Field Weakening and Flux Control

Since the motor back EMF is proportional to the motor speed, there is a motor specific speed limit at which the supply voltage is compensated by the back EMF and the motor speed cannot be increased any further since no more torque can be produced. This is shown in *Figure 14*. The voltage limiter limits the output PWM duty cycle to OUTPUT_VOLTAGE_LIMIT. The inputs to the voltage limiter are UD and UQ, while its outputs are UD_LIMITED and UQ_LIMITED. For BLDC and stepper motors the absolute voltage needs to be limited.

To accelerate the motor beyond this velocity the back EMF needs to be reduced. This can be done by inducing a negative flux current to reduce the flux that couples into the coil windings due to the rotor permanent magnets. As the total current needs to stay constant due to thermal limitations the torque current drops, and the output torque is reduced compared to non-field-weakening operation. The field weakening feature is implemented as an I controller (see *Figure 14*) and is enabled by setting the corresponding parameter FIELDWEAKENING_I. If the actual motor voltage is below FIELDWEAKENING_VOLTAGE_THRESHOLD the flux target output is always 0. Once the actual motor voltage exceeds FIELDWEAKENING_VOLTAGE_THRESHOLD a negative flux target is generated. The user needs to carefully monitor the total current during field-weakening operation when FLUX_ACTUAL is a significant part of the total current that is thermally relevant.

$$U_{Max} > \sqrt{U_Q^2 + U_D^2}$$

U_D is granted its maximum value while U_Q is limited to the remaining voltage.

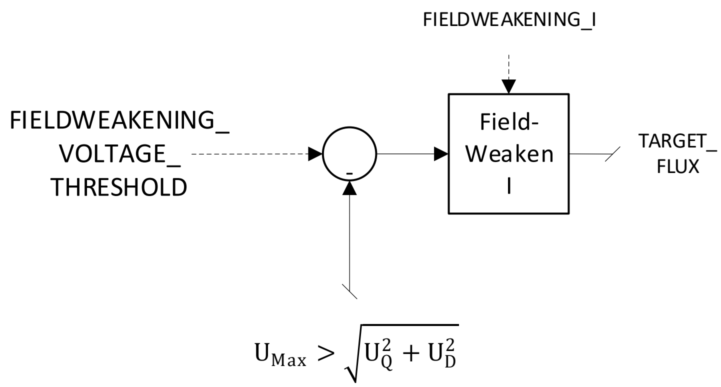

Figure 14. Field weakening controller structure

Table 34. Parameters for torque and flux control loop

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
6	MAX_TORQUE [mA] Maximum motor torque. Note: This value can be temporarily exceeded marginally due to the operation of the current regulator.	0 ... 65535	2000	RWE
7	MAX_FLUX [mA] Max. motor flux. Note: This value can be temporarily exceeded marginally due to the operation of the current regulator.	0 ... 65535	2000	RWE
104	TARGET_TORQUE [mA] Target torque value. Write to activate torque regulation.	-32768 ... 32767	0	RW
105	ACTUAL_TORQUE [mA] Actual motor torque value.	-32767 ... 32768	0	R
106	TARGET_FLUX [mA] Target flux value.	-10000 ... 10000	0	RW
107	ACTUAL_FLUX [mA] Actual motor flux value.	-2147483648 ... 2147483647	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
108	TORQUE_OFFSET [mA] (peak) Offset applied to torque value.	-4700 … 4700	0	RW
109	TORQUE_P P parameter for torque PI regulator. Also controls flux P parameter unless separate torque/flux loops are enabled.	0 … 32767	50	RWE
110	TORQUE_I I parameter for torque PI regulator. Also controls flux I parameter unless separate torque/flux loops are enabled.	0 … 32767	100	RWE
111	FLUX_P P parameter for flux PI regulator. Only available when separated torque/flux loops are enabled.	0 … 32767	50	RWE
112	FLUX_I I parameter for flux PI regulator. Only available when separated torque/flux loops are enabled.	0 … 32767	100	RWE
113	SEPARATE_TORQUE_FLUX_PI_PARAMTERS Enable to configure separate PI values for the torque and flux current control loops. False: TORQUE_FLUX_PI_COMBINED True: TORQUE_FLUX_PI_SEPARATED	0, 1	0	RWE
114	CURRENT_NORM_P P parameter normalization format for current PI regulator. 0: SHIFT_8_BIT 1: SHIFT_16_BIT	0, 1	0	RWE
115	CURRENT_NORM_I I parameter normalization format for current PI regulator. 0: SHIFT_8_BIT 1: SHIFT_16_BIT	0, 1	1	RWE
116	TORQUE_PI_ERROR Torque PI regulator error.	-2147483648 … 2147483647	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
117	FLUX_PI_ERROR Flux PI regulator error.	-2147483648 ... 2147483647	0	R
118	TORQUE_PI_INTEGRATOR Integrated error of torque PI regulator.	-2147483648 ... 2147483647	0	R
119	FLUX_PI_INTEGRATOR Integrated error of flux PI regulator.	-2147483648 ... 2147483647	0	R
120	FLUX_OFFSET [mA] (peak) Offset applied to flux value.	-4700 ... 4700	0	RW
308	FIELDWEAKENING_I I parameter for field weakening controller.	0 ... 32767	0	RWE
310	FIELDWEAKENING_VOLTAGE_THRESHOLD Maximum motor voltage allowed for field weakening.	0 ... 32767	32767	RWE

VELOCITY MODE

To activate velocity control, it is sufficient to write a TARGET_VELOCITY. Based on the target and actual velocity, and if configured the velocity ramping parameters, a target torque is calculated and applied to the torque control structure described in the previous section. The general velocity control structure is drawn in *Figure 15*. If the system is operated in velocity mode the status flag REGULATION_VELOCITY is active.

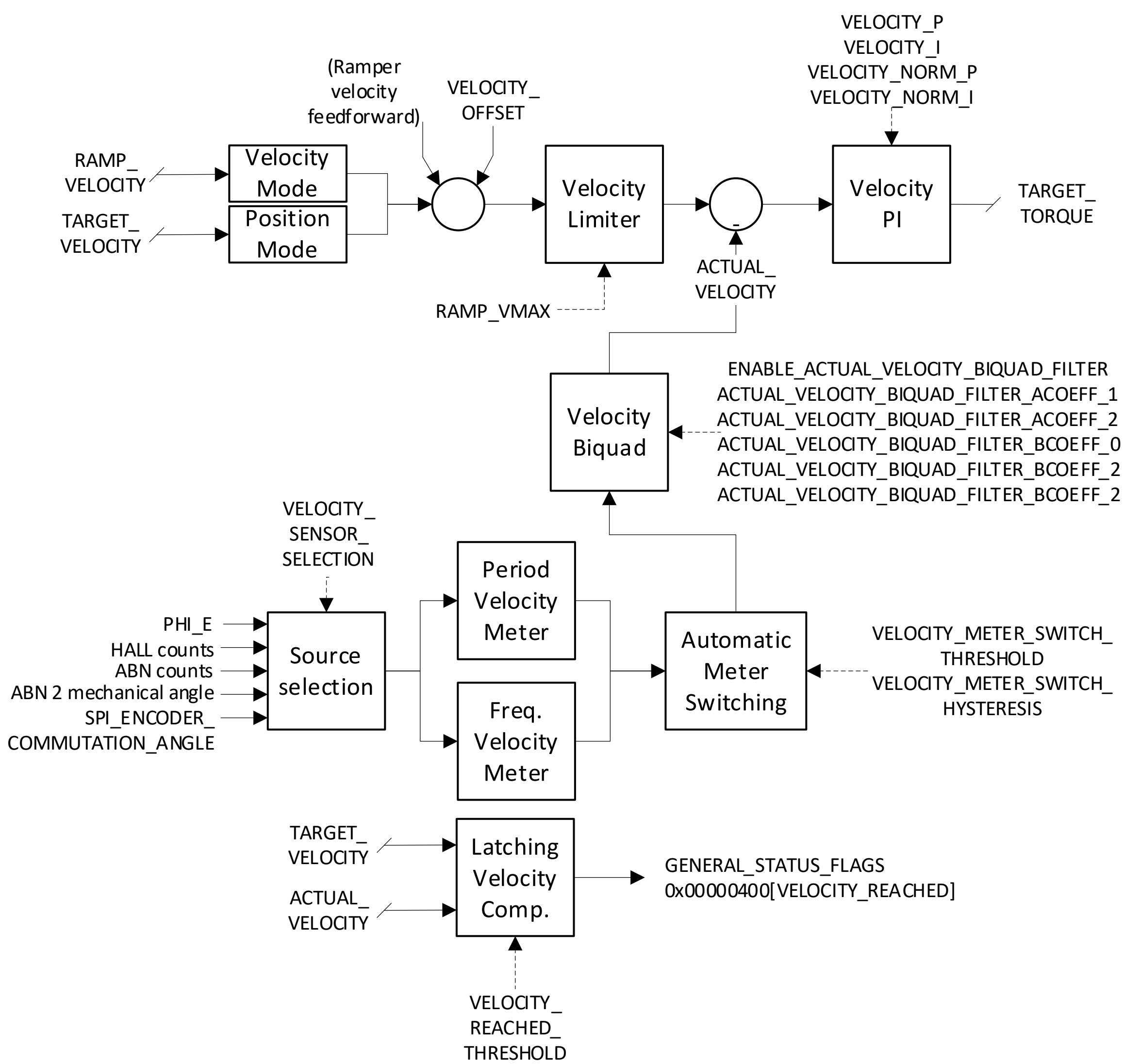

Figure 15. Velocity control loop

For velocity and position control a sophisticated ramper structure is available. When using the TMC9660 Parameter Mode the ramper block is always active to allow additional functionality. This functionality includes reacting to limit switch inputs, protection features when the system can not follow the target ramp as well as feedforward calculation.

Disabling the parameter RAMP_ENABLE configures the hardware ramper for maximum available acceleration. All ramper-based features are thus still available. The general ramper block integration is illustrated in *Figure 16*. More details follow in the sections *Position Mode* and *Ramper Stop Conditions and Reference Switches*.

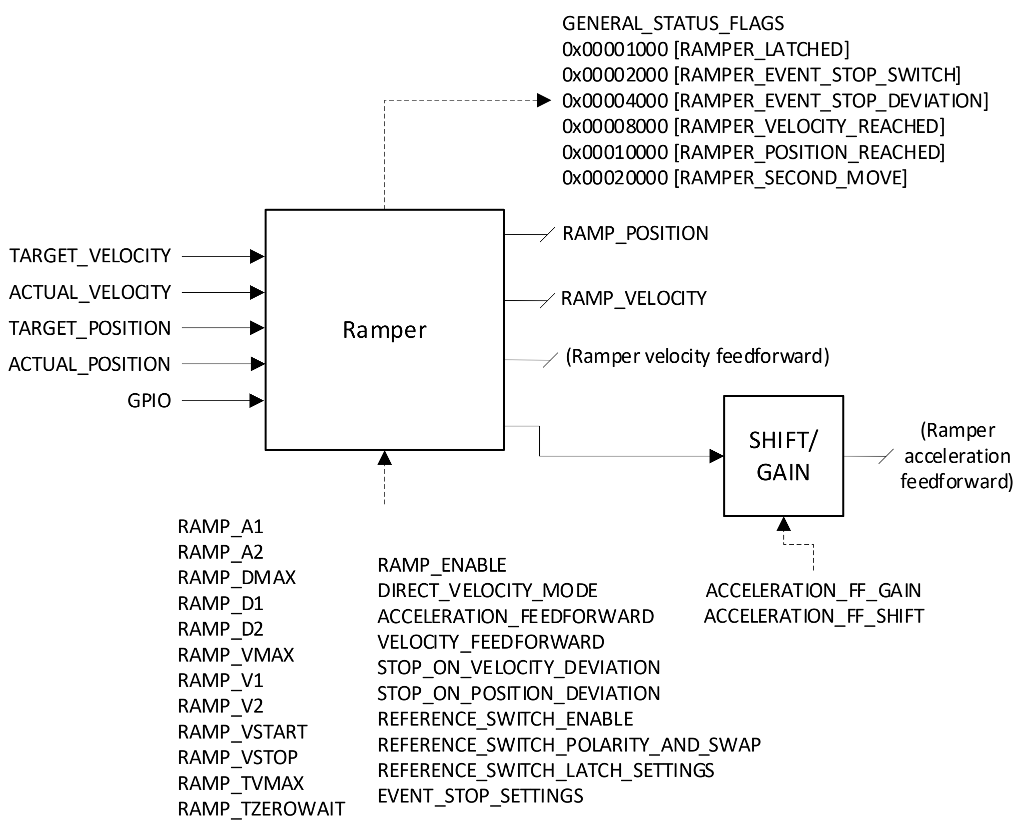

Figure 16. Ramper block integration

Velocity Feedback System and Scaling

The velocity control system's feedback must be selected using the parameter VELOCITY_SENSOR_SELECTION. Depending on the interface, the feedback system may need additional configuration. See the section *Feedback Sensor Configuration*.

The frequency of the velocity PI controller, the feedback system, and the biquad filter depends on the PWM frequency configuration and the velocity loop down sampler. For more details, see the section *Control Loop Frequencies*.

Calculating real-world velocity values from the target and actual values depends on the selected velocity feedback configuration. The conversion factor from internal units to mechanical RPM (k_{RPM}) can generally be calculated based on the number of counts per mechanical revolution (CPR). The formulas below show how the internal velocity can be converted to mechanical RPM:

$$v_{RPM_m} = \frac{v_{Internal}}{k_{RPM}}$$

$$k_{RPM} = CPR \times \frac{2^{24}}{40MHz \times 60}$$

The CPR value depends on the selected feedback method defined by the parameter VELOCITY_SENSOR_SELECTION.

- For SAME_AS_COMMUTATION the value is dependent on the pole pair count of the motor that should also be defined in the parameter MOTOR_POLE_PAIRS.

$$CPR = 2^{16} \times MOTORS\_POLE\_PAIRS$$

- For DIGITAL_HALL the value is dependent on the pole pair count of the motor that should also be defined in the parameter MOTOR_POLE_PAIRS.

$$CPR = 6 \times MOTORS\_POLE\_PAIRS$$

- For ABN1_ENCODER and ABN2_ENCODER and SPI_ENCODER, the value of counts per mechanical revolution must be determined from the encoder data sheet.

Example:

For a BLDC motor with a pole pair count of 4 and VELOCITY_SENSOR_SELECTION set to SAME_AS_COMMUTATION, the resulting scaling factor is 1832.519. If an ABN encoder with 4096 steps is used, a scaling factor of 28.633 results. To determine the mechanical revolutions per minute, the internal unit must be divided by the calculated scaling factor.

Additionally, the parameter VELOCITY_SCALING_FACTOR can be set to enable internal scaling of the parameters. To enable this, the calculated scaling value (k) must be written to the parameter. Note that it is advised to leave the parameter at default and handle scaling externally. Internal scaling factors can only be set with integer precision, and scaling reduces the resolution of the target and actual values.

The TMC9660 Parameter Mode offers two separate velocity calculation engines and an automatic switchover. For low velocities, a period-based measurement engine is available. This engine evaluates the period between two encoder increments to calculate the actual velocity. A second frequency-based meter is available and beneficial for higher motor velocities. This meter evaluates the velocity feedback increments per velocity loop period. The TMC9660 Parameter Mode supports automatic switchover between both feedback systems based on a threshold velocity. The parameter VELOCITY_METER_SWITCH_THRESHOLD defines the threshold for the switchover. Additionally, a hysteresis can be configured. The system switches back to the period meter as soon as the actual velocity falls below the threshold minus the hysteresis. The switchover between the metering systems is not noticeable and does not need further configurations.

The optimal point for switching from period-based to frequency-based velocity measurement can be calculated. At this point, the noise performance of the frequency-based velocity measurement becomes superior to the period-based measurement. The TMC9660 Parameter Mode supports an automatic switchover between both meter systems. This feature is particularly relevant for applications that need precise velocity control in a broad range of velocities.

The mechanical revolutions per minute at which the switchover should happen can be calculated. This value then must be converted to the scaled target and actual values and written to the parameter VELOCITY_METER_SWITCH_THRESHOLD. The formula to calculate the value is shown below.

There are two threshold values that can be calculated. The first velocity value is the limit of the period meter (v_{PerLim_{RPM}}). At this point, the period meters hit a maximum input frequency where the noise performance significantly decreases. The formula for this threshold includes a margin factor of 0.9. The second velocity value is the crossover point (v_{COP_{RPM}}), which calculates the point where the frequency meter generally shows better performance. *Figure 17* illustrates the thresholds. To calculate the values, the encoder counts per revolution (CPR) and the velocity loop frequency (f_{Velo}) are needed (see the section *Control Loop Frequencies*). The smaller of the two calculated velocities should be chosen.

$$v_{PerLim_{RPM}} = 0.9 \times \frac{40MHz}{CPR} \times \frac{60}{53}$$

$$v_{COP_{RPM}} = 60 \times \frac{f_{Velo} + \sqrt{f_{Velo}^2 + f_{Velo} \times 40MHz \times 8}}{4 \times CPR}$$

$$v_{THR_{RPM}} = \min\{v_{COP_{RPM}}, v_{PerLim_{RPM}}\}$$

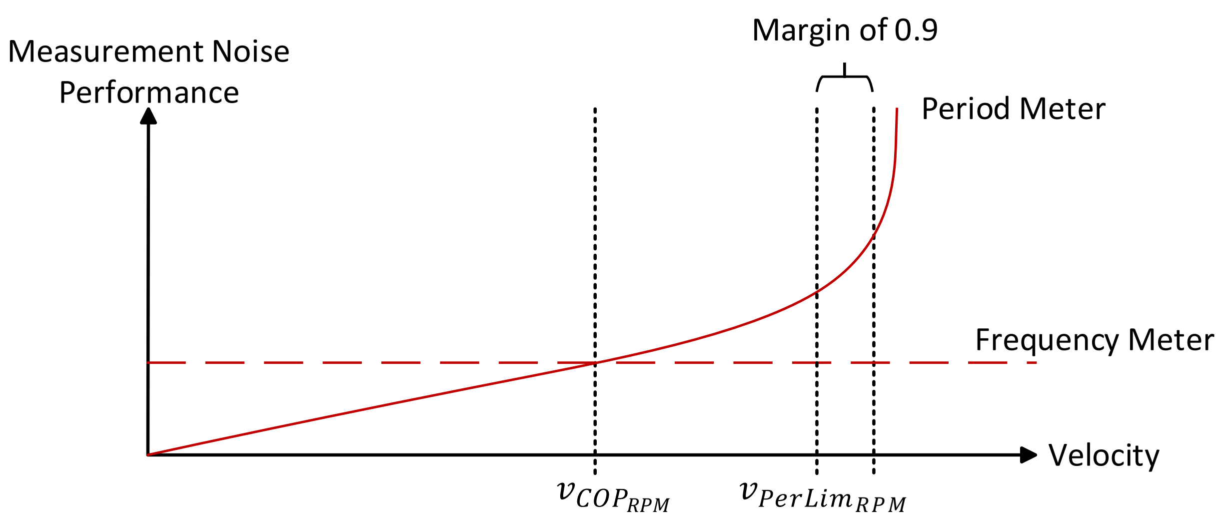

Figure 17. Noise performance of frequency and period velocity meter

This value must then be converted back to the scaled velocity used as target and actual values. This calculation is shown in the formula below. The parameter VELOCITY_METER_SWITCH_THRESHOLD should be set according to the calculated value. Additionally, the parameter VELOCITY_METER_SWITCH_HYSTERESIS can be set to suppress switching due to sensor noise in the border region.

$$\text{VELOCITY\_METER\_SWITCH\_THRESHOLD} = \frac{v_{THR_{RPM}} \times k_{RPM}}{\text{VELOCITY\_SCALING\_PARAMETER}}$$

Note: When using the unscaled internal velocity unit, the 16-bit resolution of VELOCITY_METER_SWITCH_HYSTERESIS might not be sufficient, and the switchover might be triggered by noise in the velocity signal. In that case, it is recommended to scale the velocity to real-world units to obtain a much higher velocity range.

The velocity feedback can be filtered using a biquad filter running at velocity loop frequency. The setup is described in the section *Biquad filter setup*.

Table 35. Parameters for velocity loop

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
123	VELOCITY_SENSOR_SELECTION Feedback source for the velocity PI regulator. 0: SAME_AS_COMMUTATION 1: DIGITAL_HALL 2: ABN1_ENCODER 3: ABN2_ENCODER 4: SPI_ENCODER	0, 1, 2, 3, 4	0	RWE
124	TARGET_VELOCITY Target velocity value. Write to activate velocity regulation.	-134217728 ... 134217727	0	RW
125	ACTUAL_VELOCITY Actual velocity value.	-2147483648 ... 2147483647	0	R
127	VELOCITY_P P parameter for velocity PI regulator.	0 ... 32767	800	RWE
128	VELOCITY_I I parameter for velocity PI regulator.	0 ... 32767	1	RWE
129	VELOCITY_NORM_P P parameter normalization format for velocity PI regulator. 0: NO_SHIFT 1: SHIFT_8_BIT 2: SHIFT_16_BIT 3: SHIFT_24_BIT	0, 1, 2, 3	2	RWE
130	VELOCITY_NORM_I I parameter normalization format for velocity PI regulator. 0: SHIFT_8_BIT 1: SHIFT_16_BIT 2: SHIFT_24_BIT 3: SHIFT_32_BIT	0, 1, 2, 3	2	RWE
131	VELOCITY_PI_INTEGRATOR Integrated error of velocity PI regulator.	-2147483648 ... 2147483647	0	R
132	VELOCITY_PI_ERROR Velocity PI regulator error.	-2147483648 ... 2147483647	0	R
133	VELOCITY_SCALING_FACTOR Scaling factor to convert internal velocity to real-world units.	1 ... 2047	1	RWE
135	VELOCITY_LOOP_DOWNSAMPLING Downsampling factor for velocity controller.	0 ... 127	5	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
137	VELOCITY_METER_SWITCH_THRESHOLD Velocity threshold switching from period to frequency velocity meter.	0 ... 134217727	2000	RWE
138	VELOCITY_METER_SWITCH_HYSTERESIS Velocity hysteresis for switching back from frequency to period velocity meter.	0 ... 65535	500	RWE
139	VELOCITY_METER_MODE Currently used velocity meter mode. 0: PERIOD_METER Measurement of velocity by time measurement between position changes. 1: FREQUENCY_METER Velocity Meter running at PWM frequency. Calculates the velocity using the difference of the angle in one clock cycle. 2: SOFTWARE_METER Measurement of velocity by software.	0, 1, 2	0	R

Velocity Ramp Functionality

The TMC9660 Parameter Mode supports a sophisticated hardware ramper functionality, with a configurable velocity profile illustrated in *Figure 18*. It consists of three different acceleration values (RAMP_A1, RAMP_A2, and RAMP_AMAX) that are also applied during deceleration, depending on the velocity thresholds RAMP_V1 and RAMP_V2. Three different sets of acceleration and deceleration can be freely combined and configured using dedicated parameters. The transition velocities RAMP_V1 and RAMP_V2 allow for velocity-dependent switching between three acceleration and deceleration settings. Typically, high-velocity applications use lower acceleration and deceleration values at higher velocities due to the motor's torque decline at higher speeds.

The acceleration can be calculated using the formula below. Depending on the amount of velocity units incremented over a time in seconds. The real-world acceleration is depending on the used velocity feedback method. The acceleration value does not take additional velocity scaling into account. If a scaler value is used, as described in the previous section, the scaler must be added to the formula as a factor.

$$A_X = \frac{\Delta V_{\text{counts}} \times 2^{17}}{\Delta t_s \times 40 \times 10^6} \times VELOCITY\_SCALING\_FACTOR$$

To enhance the controller's performance, an acceleration feedforward system can be configured. The feedforward signal must be enabled with the parameter ACCELERATION_FEEDFORWARD and gain and shift factors need to be defined using the parameters ACCELERATION_FF_GAIN and ACCELERATION_FF_SHIFT. The offset applied to the torque controller can be calculated as:

$$Offset = \frac{(A_{ramper} \times ACCELERATION\_FF\_GAIN)}{2^{ACCELERATION\_FF\_SHIFT \times 4}}$$

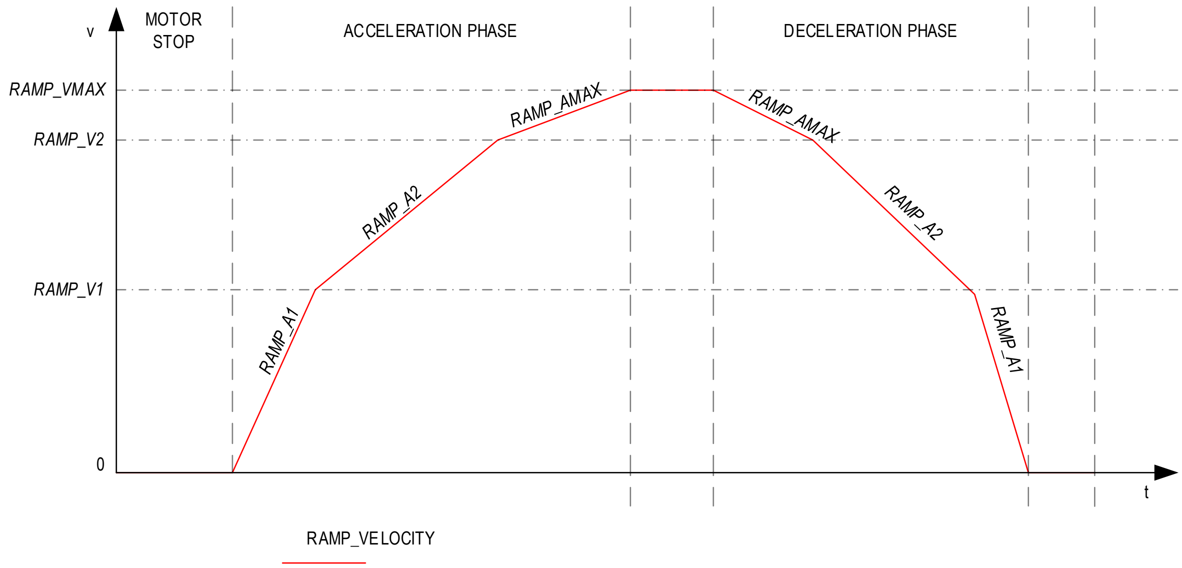

Figure 18. Velocity ramper profile

Table 36. Parameters for ramper function block

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
45	OPENLOOP_ANGLE Phi_e calculated by the ramper hardware. Used for commutation in openloop modes.	-32768 … 32767	0	R
50	ACCELERATION_FF_GAIN Gain applied to acceleration feedforward.	0 … 65535	8	RWE
51	ACCELERATION_FF_SHIFT Shift applied to acceleration feedforward. 0: NO_SHIFT 1: SHIFT_4_BIT 2: SHIFT_8_BIT 3: SHIFT_12_BIT 4: SHIFT_16_BIT 5: SHIFT_20_BIT 6: SHIFT_24_BIT	0, 1, 2, 3, 4, 5, 6	4	RWE
52	RAMP_ENABLE Enable the application of acceleration and deceleration ramps. False: DISABLED True: ENABLED	0, 1	0	RWE
53	DIRECT_VELOCITY_MODE Specify the control loop structure for velocity mode. Directly regulating the velocity or regulating on a constantly calculated target position. False: DISABLED True: ENABLED	0, 1	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
54	RAMP_AMAX [internal] Acceleration in top part of eight-point ramp.	1 ... 8388607	1000	RWE
55	RAMP_A1 [internal] First acceleration in eight-point ramp.	1 ... 8388607	8000	RWE
56	RAMP_A2 [internal] Second acceleration in eight-point ramp.	1 ... 8388607	4000	RWE
57	RAMP_DMAX [internal] Deceleration in top part of eight-point ramp.	1 ... 8388607	1000	RWE
58	RAMP_D1 [internal] Second deceleration in eight-point ramp.	1 ... 8388607	8000	RWE
59	RAMP_D2 [internal] First deceleration in eight-point ramp.	1 ... 8388607	8000	RWE
60	RAMP_VMAX [internal] Maximum velocity of eight-point ramp.	0 ... 134217727	134217727	RWE
61	RAMP_V1 [internal] Velocity threshold to switch from A1/D1 to A2/D2.	0 ... 134217727	0	RWE
62	RAMP_V2 [internal] Velocity threshold to switch from A2/D2 to AMAX/DMAX.	0 ... 134217727	0	RWE
63	RAMP_VSTART [internal] Start velocity of ramp.	0 ... 8388607	0	RWE
64	RAMP_VSTOP [internal] Stop velocity of ramp. Needs to be greater than 0.	1 ... 8388607	1	RWE
65	RAMP_TVMAX [internal] Minimum time at VMAX to start deceleration.	0 ... 65535	0	RWE
66	RAMP_TZEROWAIT [internal] Wait time at end of ramp to signal stop.	0 ... 65535	0	RWE
67	ACCELERATION_FEEDFORWARD_ENABLE Enable the acceleration feedforward feature. False: DISABLED True: ENABLED	0, 1	0	RWE
68	VELOCITY_FEEDFORWARD_ENABLE Enable the velocity feedforward feature. False: DISABLED True: ENABLED	0, 1	0	RWE
69	RAMP_VELOCITY Target velocity calculated by ramp controller.	-134217727 ... 134217727	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
70	RAMP_POSITION Target position calculated by ramp controller.	-2147483648 ... 2147483647	0	R

POSITION MODE

To use the system in position mode the commutation mode must be configured first. After that the position control mode gets activated by writing a target position to the parameter TARGET_POSITION. This automatically activates the position control mode. The status flag REGULATION_POSITION indicates that the system is in position control mode.

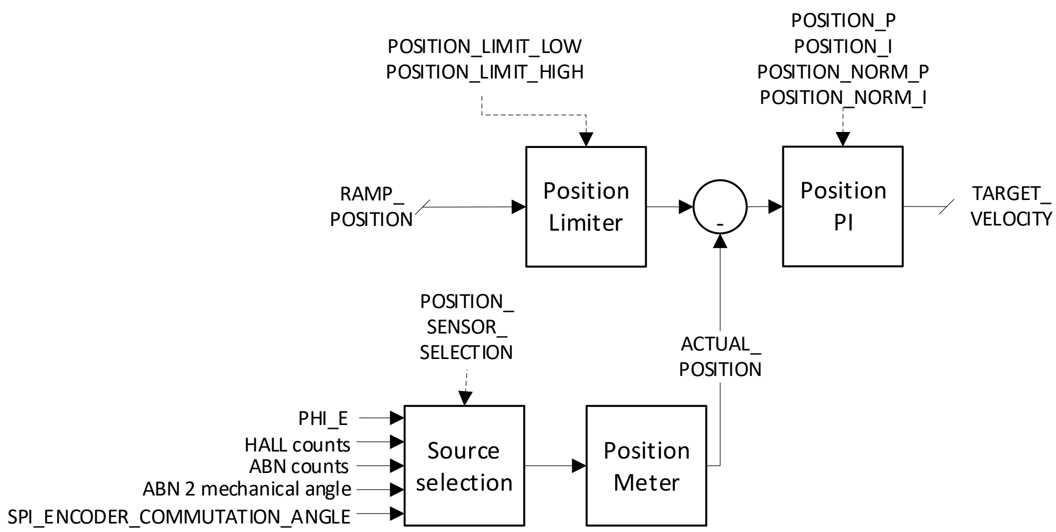

Figure 19. Position loop structure

Position Feedback System and Scaling

The feedback system for the position control must be selected using the parameter POSITION_SENSOR_SELECTION. Depending on the interface, the feedback system needs additional configuration first; see the section *Feedback Sensor Configuration* for details.

The scaling between the internal position values and real-world units such as degrees, radiants or rotations depend on the configured feedback method. If "Same as commutation" is configured 65536 position steps are applied for each pole pair. For a full rotation the range must be multiplied by the number of motor pole pairs.

In the TMC9660 Parameter Mode, an optional position scaling is available with the parameter POSITION_SCALING_FACTOR. This scales the internal units and can be used to scale the target and actual values to real-world units. If maximum position resolution is desired, it is recommended to leave the position scaling factor at the default value of 1024. If an encoder is configured the scaling depends on the encoder resolution.

The frequency of the position PI controller and the feedback system depend on the PWM frequency configuration as well as the velocity and position loop down sampler. For more details see the section *Control Loop Frequencies*.

Position Ramp Functionality

For position control, a sophisticated ramp generator is available. It delivers three-phase acceleration and three-phase deceleration ramps with additional programmable start and stop velocities. The general ramp profile is shown in *Figure 20*.

Three different sets of acceleration and deceleration can be combined freely and configured using the dedicated parameters. The transition velocities RAMP_V1 and RAMP_V2 allow for velocity-dependent switching between three acceleration and deceleration settings. The acceleration values are the same that are used for velocity ramps. The deceleration values can be set by the parameters RAMP_D1, RAMP_D2 and RAMP_DMAX. The deceleration values can be calculated the same as the velocity values described in the section *Velocity Ramp Functionality*.

A typical high-velocity application uses lower acceleration and deceleration values at higher velocities, as the motor's torque declines at higher velocity. When considering friction in the system, it becomes clear that the deceleration capability of the system is quicker than the acceleration capability. Thus, deceleration values can be set higher in many applications. This way, the operation speed of the motor in time-critical applications is maximized.

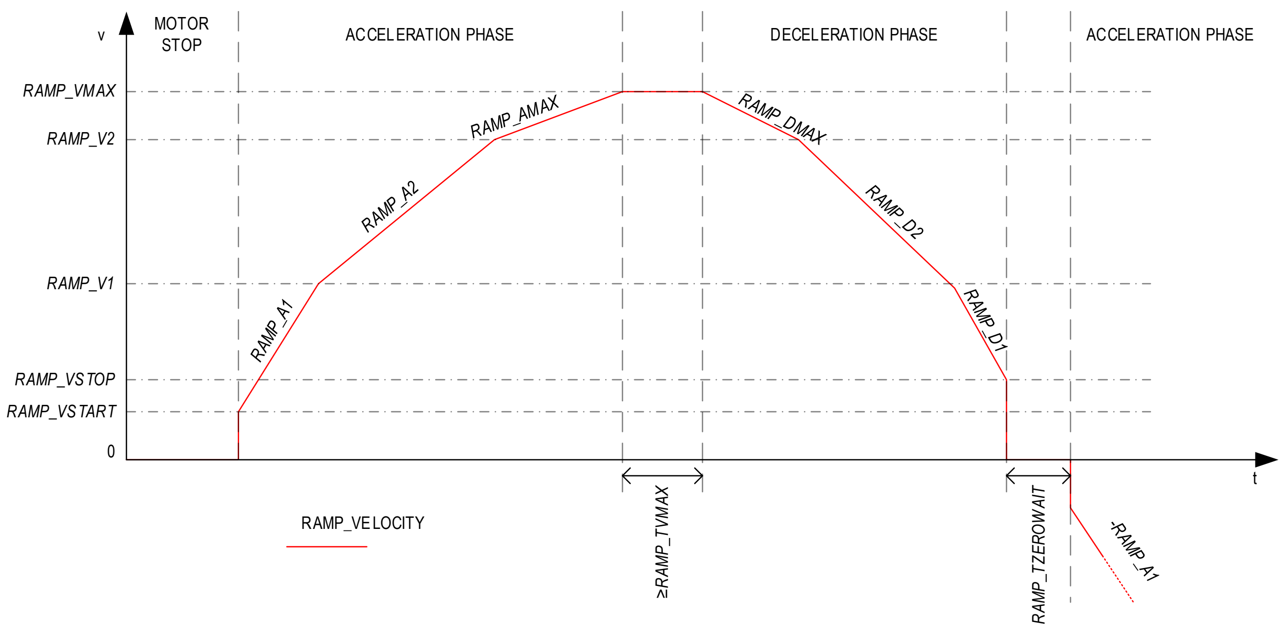

Figure 20. General position ramp profile

As target positions and ramp parameters may be changed at any time during the motion, the motion controller always uses the optimum (fastest) way to reach the target while sticking to the acceleration constraints set by the user. The *Figure 21* illustrates various scenarios in which the ramper dynamically adjusts to a new target position. This way, it might happen that the motion becomes automatically stopped, crosses zero, and drives back again. For example, during the final deceleration phase, the target position might be changed and "pulled-in" to a closer position, and the configured deceleration value does not allow reaching the new target position directly. This case is flagged by the flag RAMPER_SECOND_MOVE in the parameter GENERAL_STATUS_FLAGS.

The ramp generator further supports automatic jerk reduction by avoiding the direct transition from an acceleration phase into a deceleration phase, and from a deceleration phase to an acceleration phase. These situations are smoothed by enforcing a constant velocity segment of a minimum duration (TVMAX). Configure

RAMP_TVMAX as required by mechanical jerk response. Set RAMP_TVMAX to zero to disable jerk reduction. The parameter value can be multiplied by 12.8 µs to get the real would unit.

When using increased levels of start- and stop velocity, it becomes clear, that a subsequent move into the opposite direction would provide a jerk identical to RAMP_VSTART+RAMP_VSTOP, rather than only RAMP_VSTART. As the motor probably is not able to follow this, you can set a time delay for a subsequent move by setting RAMP_TZEROWAIT. Once the target position is reached, the generic status flag RAMPER_POSITION_REACHED becomes active.

The set of three acceleration and deceleration segments can be used in two ways. Either for adaptation to the motor torque curve, by using higher acceleration values at lower velocity, or to reduce the jerk (change of acceleration) when transitioning from one acceleration segment to the next. For jerk optimized ramps, typically RAMP_A1, RAMP_D1, RAMP_AMAX, and RAMP_DMAX are set to lower values than RAMP_A2 and RAMP_D2. The most critical points with regards to jerk are the transition from acceleration to deceleration with no constant velocity segment, as well as the transition from deceleration to acceleration in case of on-the-fly change of target position.

To address both, the 8-point motion profile generator allows to enforce a constant velocity segment based on a minimum segment duration (RAMP_TVMAX). In case this duration could not be kept due to insufficient distance, a reduced maximum velocity becomes calculated and is used for the constant velocity segment. Minimum maximum velocity is identical to RAMP_VSTOP.

Note that the start velocity can be set to zero if not used. The stop velocity can be set to a low value (1000 or down to a few 10) if not used. <u>If RAMP_VSTOP = 0, the position might not precisely reach the configured target position.</u> Also take care to always set RAMP_VSTOP identical to or above RAMP_VSTART. This ensures that even a short motion can be terminated successfully at the target position.

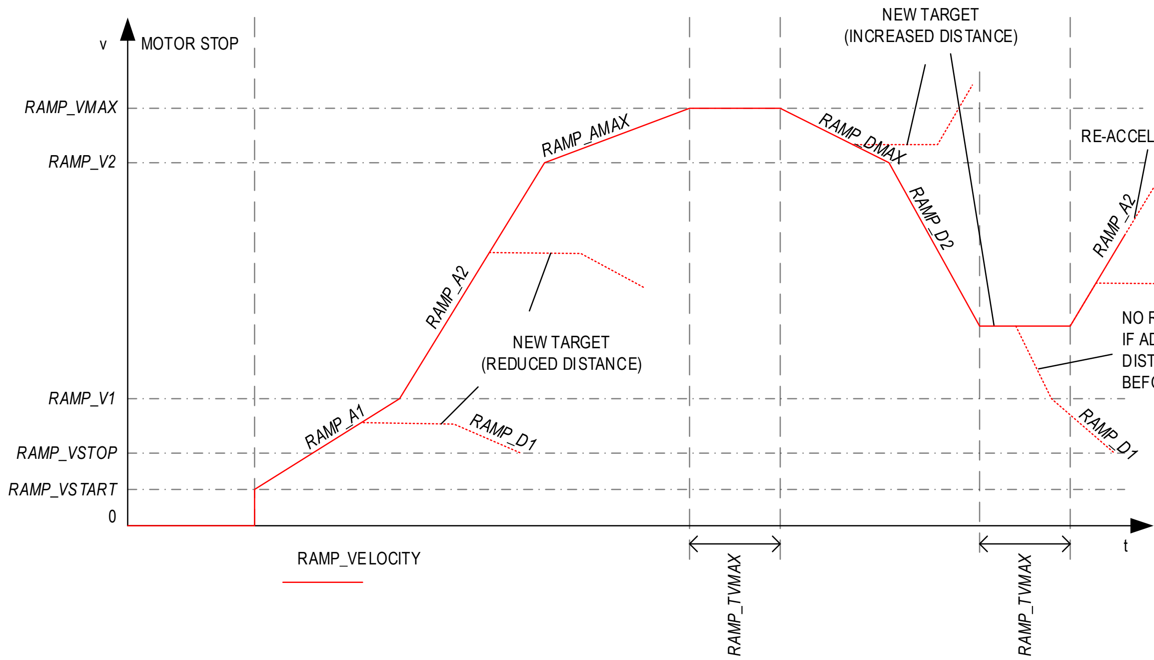

Figure 21. Position ramp profile with on-the-fly target changes

If VSTOP is not reached due to a too short travel distance, there is only a very short linear acceleration using A1 and the ramp terminates immediately when XTARGET is reached.

In cases where users can interact with a system, some applications require terminating a motion by ramping down to zero velocity before the target position has been reached.

Options to Terminate Motion using Acceleration Settings:

a) Switch to velocity mode, set RAMP_VMAX=0 and RAMP_AMAX (plus A1, A2, if used) to the desired deceleration value. This stops the motor using a linear ramp.

b) For a stop in positioning mode, set RAMP_VSTART=0 and RAMP_VMAX=0. RAMP_VSTOP is not used in this case. The driver uses RAMP_AMAX, RAMP_A1, and RAMP_A2 (as determined by V1 and V2) for decelerating to zero velocity.

c) Activate a stop switch. See the section *Ramper Stop Conditions and Reference Switches*.

d) Use the position or velocity deviation feature. See the section *Ramper Stop Conditions and Reference Switches*.

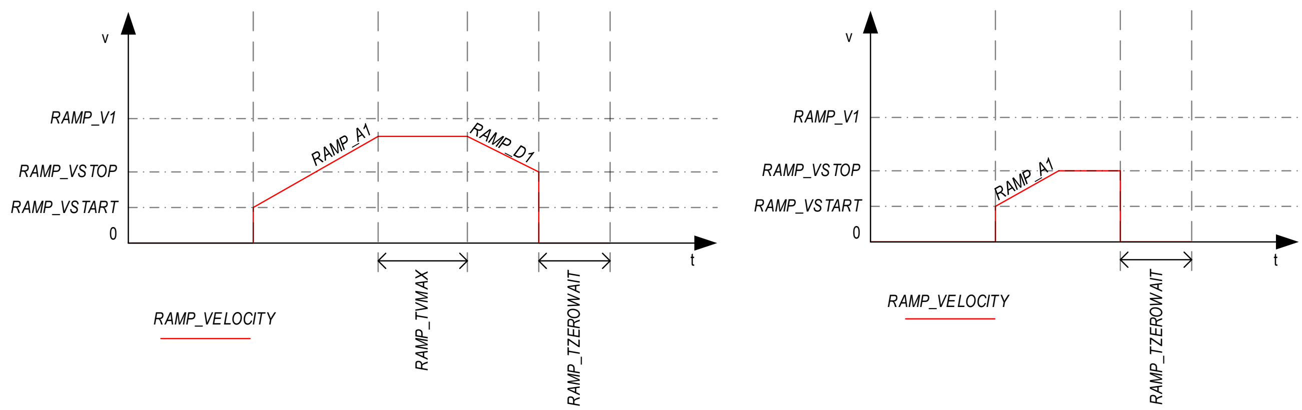

Figure 22. Position ramp profile with early ramp termination

To enhance the controller performance velocity feedforward can be activated using the parameter VELOCITY_FEEDFORWARD. The acceleration feedforward, described in the section *Velocity Ramp Functionality*, is also available.

Velocity Target with Position Control

An alternative control loop configuration for velocity control is to generate an internal position target based on an external velocity target. To reconfigure the control loop, enable the parameter DIRECT_VELOCITY_MODE. If the target velocity cannot be reached, the motor continues to move until the internal target position is reached. To protect against unexpected behavior a stop on position or velocity deviation can be configured, see the section *Ramper Stop Conditions and Reference Switches*. <u>To avoid unexpected behavior in this mode, it is necessary to ensure that the internal position does not overflow.</u>

Table 37. Parameters for position mode

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
142	POSITION_SENSOR_SELECTION Feedback source for the position PI regulator. 0: SAME_AS_COMMUTATION 1: DIGITAL_HALL 2: ABN1_ENCODER 3: ABN2_ENCODER 4: SPI_ENCODER	0, 1, 2, 3, 4	0	RWE
143	TARGET_POSITION Target position value. Write to activate position regulation.	-2147483648 ... 2147483647	0	RW
144	ACTUAL_POSITION Actual position value.	-2147483648 ... 2147483647	0	RW
145	POSITION_SCALING_FACTOR Scaling factor to convert internal position to real-world units.	1024 ... 65535	1024	RWE
146	POSITION_P P parameter for position PI regulator.	0 ... 32767	5	RWE
147	POSITION_I I parameter for position PI regulator.	0 ... 32767	0	RWE
148	POSITION_NORM_P P parameter normalization format for position PI regulator. 0: NO_SHIFT 1: SHIFT_8_BIT 2: SHIFT_16_BIT 3: SHIFT_24_BIT	0, 1, 2, 3	1	RWE
149	POSITION_NORM_I I parameter normalization format for position PI regulator. 0: SHIFT_8_BIT 1: SHIFT_16_BIT 2: SHIFT_24_BIT 3: SHIFT_32_BIT	0, 1, 2, 3	1	RWE
150	POSITION_PI_INTEGRATOR Integrated error of position PI regulator.	-2147483648 ... 2147483647	0	R
151	POSITION_PI_ERROR Error of position PI regulator.	-2147483648 ... 2147483647	0	R

RAMPER STOP CONDITIONS AND REFERENCE SWITCHES

The ramper system additionally offers the ability to stop the ramper based on events. These events can be triggered by external limit switches or when the controller is unable to follow the set ramp sufficiently. The parameter EVENT_STOP_SETTINGS uses bits to configure the behavior. The zero bit allows you to configure whether the system performs a hard or soft stop when a stop condition occurs. A soft stop means that the system decelerates with the configured ramp parameters. In the case of a limit switch, the system decelerates according to the configured ramp upon detecting a limit switch press. However, this could potentially lead to overdriving the limit switch. Conversely, if a soft stop is not configured, the system halts immediately without any deceleration upon detecting a limit switch press. The same applies to deviation events. For all available stop options, see *Table 38*.

The first and second bits of the EVENT_STOP_SETTINGS parameter enable the stop on position and stop on velocity features. If the feature is active, the system stops when the motor cannot follow the ramp. For position deviation, the deviation of the parameters RAMP_POSITION and ACTUAL_POSITION is compared against the value of the parameter STOP_ON_POSITION_DEVIATION. For velocity deviation, the parameters RAMP_VELOCITY, ACTUAL_VELOCITY, and STOP_ON_VELOCITY_DEVIATION are used. As soon as a deviation triggers a stop, the general status flag RAMPER_EVENT_STOP_DEVIATION indicates the status.

The system supports left, right, and home switch inputs. These inputs can be utilized to automatically halt the motor or to execute a reference search. To make these switches available, they must be configured in the BOOT_CONFIG. The parameter REFERENCE_SWITCH_POLARITY_AND_SWAP allows for the inversion of switch polarity and the swapping of left and right switches. An automatic halt can be configured for the system when a reference switch is triggered. This configuration is achieved using the REFERENCE_SWITCH_ENABLE parameter. When a limit switch leads to a stop, the general status flag RAMPER_EVENT_STOP_SWITCH indicates this. The REFERENCE_SWITCH_LATCH_SETTINGS parameter allows the configuration of latch behavior, useful for logging a switch position. When configured, the position at the latch event is logged in the LATCH_POSITION parameter. The RAMPER_LATCHED flag in GENERAL_STATUS_FLAGS is activated, indicating the event. To log a new position in the latch register, RAMPER_LATCHED must be cleared by writing the bit to 1.

Table 38.Parameters for ramper stop conditions.

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
134	STOP_ON_VELOCITY_DEVIATION Maximum of velocity deviation tolerated before stop event is triggered (if activated).	0 … 200000	0	RW
152	STOP_ON_POSITION_DEVIATION Maximum of position deviation tolerated before stop event is triggered (if activated).	0 … 2147483647	0	RWE
154	LATCH_POSITION Position switch latched.	-2147483648 … 2147483647	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
161	REFERENCE_SWITCH_ENABLE Bitwise enable for stopping when reference switch input is triggered. Bit 2: Stop on reference input home. Bit 1: Stop on reference input right. Bit 0: Stop on reference input left. 0: NO_STOP_ON_SWITCH_TRIGGERED 1: STOP_ON_L 2: STOP_ON_R 3: STOP_ON_R_AND_L 4: STOP_ON_H 5: STOP_ON_H_AND_L 6: STOP_ON_H_AND_R 7: STOP_ON_H_R_AND_L	0, 1, 2, 3, 4, 5, 6, 7	0	RWE
162	REFERENCE_SWITCH_POLARITY_AND_SWAP Bitwise configuration of reference switch configuration. Options to swap left and right input and invert switch polarity. Bit 3: Swap left and right switch. Bit 2: Invert polarity of home switch. Bit 1: Invert polarity of right switch. Bit 0: Invert polarity of left switch. 0: NOT_SWAPPED_NOT_INVERTED 1: L_INVERTED 2: R_INVERTED 3: R_AND_L_INVERTED 4: H_INVERTED 5: H_AND_L_INVERTED 6: H_AND_R_INVERTED 7: H_R_AND_L_INVERTED 8: L_R_SWAPPED_L_INVERTED 9: L_R_SWAPPED_R_INVERTED 10: L_R_SWAPPED_R_AND_L_INVERTED 11: L_R_SWAPPED_H_INVERTED 12: L_R_SWAPPED_H_AND_L_INVERTED 13: L_R_SWAPPED 14: L_R_SWAPPED_H_AND_R_INVERTED 15: L_R_SWAPPED_H_R_AND_L_INVERTED	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
163	REFERENCE_SWITCH_LATCH_SETTINGS Bitwise configuration of reference switch latch configuration. Writing position to latch position parameter. Bit 3: Trigger latch on falling home signal. Bit 2: Trigger latch on rising home signal. Bit 1: Trigger latch on falling left and right signal. Bit 0: Trigger latch on rising left and right signal. 0: NO_TRIGGER 1: L_R_RISING_EDGE 2: L_R_FALLING_EDGE 3: L_R_BOTH_EDGES 4: H_RISING_EDGE 5: H_L_R_RISING_EDGE 6: H_RISING_L_R_FALLING_EDGE 7: H_RISING_L_R_BOTH_EDGES 8: H_FALLING_EDGE 9: H_FALLING_L_R_RISING_EDGE 10: H_L_R_FALLING_EDGE 11: H_FALLING_L_R_BOTH_EDGES 12: H_BOTH_EDGES 13: H_BOTH_L_R_RISING_EDGE 14: H_BOTH_L_R_FALLING_EDGE 15: H_L_R_BOTH_EDGES	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE
164	EVENT_STOP_SETTINGS Bitwise configuration of stop configuration. Bit 2: Stop if the velocity loop deviation is larger then the parameter "Stop on velocity deviation". Bit 1: Stop if the position loop deviation is larger then the parameter "Stop on position deviation". Bit 0: If enabled the system ramps down to zero if a stop condition rises. Doing a soft and not a hard stop. 0: DO_HARD_STOP 1: DO_SOFT_STOP 2: STOP_ON_POS_DEVIATION 3: STOP_ON_POS_DEVIATION_SOFT_STOP 4: STOP_ON_VEL_DEVIATION 5: STOP_ON_VEL_DEVIATION_SOFT_STOP 6: STOP_ON_POS_VEL_DEVIATION 7: STOP_ON_POS_VEL_DEVIATION_SOFT_STOP	0, 1, 2, 3, 4, 5, 6, 7	0	RWE

BIQUAD FILTER SETUP

The TMC9660 Parameter Mode includes biquad filters to enhance the performance of its velocity and torque control loops. The biquad filter, essentially a second-order digital filter, operates by processing the incoming signal through a combination of current and past input samples, along with previous output samples. The filter's behavior, such as its role as a low-pass, high-pass, band-pass, or notch filter, is determined by a set of coefficients that are stored in dedicated register. The filter's output Y(n) at a given time step n is calculated using the following equation:

$$Y(n) = X(n) \times b_0 + X(n-1) \times b_1 + X(n-2) \times b_2 + Y(n-1) \times a_1 + Y(n-2) \times a_2$$

where:

- X(n) represents the current input sample.
- X(n-1) and X(n-2) are the previous input samples.
- Y(n-1) and Y(n-2) are the previous output samples.
- a_1, a_2, b_0, b_1, and b_2 are the filter coefficients.

The filter coefficients are 24-bit values normalized to a Q4.20 format.

The velocity biquad filter is used to filter the actual velocity of the motor that is used as input for the velocity controller. The exact location in the control loop can be seen in *Figure 15* . The filter coefficients can be set using the parameters named ACTUAL_VELOCITY_BIQUAD_FILTER_ in *Table 39*. This filter is enabled by default because the measured velocity is usually quite noisy. The biquad can be disabled with the parameter ENABLE_ACTUAL_VELOCITY_BIQUAD_FILTER.

The torque biquad filter is used to filter the input target value of the torque controller. This can be helpful when using the velocity or position control. In that case the output of the velocity controller is used as target torque. The exact location in the control loop can be seen in *Figure 13*. The filter coefficients can be set using the parameters named TARGET_TORQUE_BIQUAD_FILTER_ in *Table 39*. The biquad can be disabled with the parameter ENABLE_TARGET_TORQUE_BIQUAD_FILTER.

Table 39. Biquad filter configuration parameters

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
318	TARGET_TORQUE_BIQUAD_FILTER_ENABLE Enable the target torque biquad filter. False: DISABLED True: ENABLED	0, 1	0	RWE
319	TARGET_TORQUE_BIQUAD_FILTER_ACOEFF_1 Target torque biquad filter aCoeff_1.	-2147483648 ... 2147483647	0	RWE
320	TARGET_TORQUE_BIQUAD_FILTER_ACOEFF_2 Target torque biquad filter aCoeff_2.	-2147483648 ... 2147483647	0	RWE
321	TARGET_TORQUE_BIQUAD_FILTER_BCOEFF_0 Target torque biquad filter bCoeff_0.	-2147483648 ... 2147483647	1048576	RWE
322	TARGET_TORQUE_BIQUAD_FILTER_BCOEFF_1 Target torque biquad filter bCoeff_1.	-2147483648 ... 2147483647	0	RWE
323	TARGET_TORQUE_BIQUAD_FILTER_BCOEFF_2 Target torque biquad filter bCoeff_2.	-2147483648 ... 2147483647	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
324	ACTUAL_VELOCITY_BIQUAD_FILTER_ENABLE Enable the actual velocity biquad filter. False: DISABLED True: ENABLED	0, 1	1	RWE
325	ACTUAL_VELOCITY_BIQUAD_FILTER_ACOEFF_1 Actual velocity biquad filter aCoeff_1.	-2147483648 ... 2147483647	1849195	RWE
326	ACTUAL_VELOCITY_BIQUAD_FILTER_ACOEFF_2 Actual velocity biquad filter aCoeff_2.	-2147483648 ... 2147483647	15961938	RWE
327	ACTUAL_VELOCITY_BIQUAD_FILTER_BCOEFF_0 Actual velocity biquad filter bCoeff_0.	-2147483648 ... 2147483647	3665	RWE
328	ACTUAL_VELOCITY_BIQUAD_FILTER_BCOEFF_1 Actual velocity biquad filter bCoeff_1.	-2147483648 ... 2147483647	7329	RWE
329	ACTUAL_VELOCITY_BIQUAD_FILTER_BCOEFF_2 Actual velocity biquad filter bCoeff_2.	-2147483648 ... 2147483647	3665	RWE

FAULT HANDLING

The TMC9660 Parameter Mode supports advanced handling of fault conditions. Depending on the fault, the system can be configured to react in various ways to reduce the risk of damage, ensuring optimal protection and reliability. This fault handling is applied to overtemperature conditions, IIT violations, and gate driver faults. When a fault occurs, the system switches off in all cases. For all faults, four different switch-off behavior options are available. It can be selected if all MOSFETs get discharged or the system shorts the motor phases. In this case, the high-side MOSFETs are switched and shorted if possible. Shorting the motor phases ensures that the motor cannot generate high voltages due to back-EMF; however, high motor currents will result if the motor was turning before the fault condition occurred. Open loop behavior can generate significant voltages. Both options are available with or without engagement of the mechanical brake. The mechanical brake must be configured as described in the section *Mechanical Brake*.

For gate driver-related faults, an additional retry mechanism is in place to prevent a false trigger from causing a system stop. Depending on the specific gate driver fault, the system's response options may be limited. The system attempts to switch off and restart. The maximum number of retries is defined by the FAULT_HANDLER_NUMBER_OF_RETRIES parameter, where a value of zero means no retries and 255 means infinite retries. The behavior between fault and retry is governed by the GDRV_RETRY_BEHAVIOUR parameter. If all retries fail, the system applies the standard fault behavior defined by the DRIVE_FAULT_BEHAVIOUR parameter. If electrical braking is selected, the system shorts the low side if the high side is not possible due to the fault. If this is also not possible, it defaults to open-circuit behavior. The flag FAULT_RETRY_HAPPENED indicates that a retry occurred, and FAULT_RETRIES_FAILED indicates that the maximum number of retries was reached without success.

Table 40. Parameters for fault handling

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
286	GDRV_RETRY_BEHAVIOUR This value defines the state the system goes to after a fault condition on a motor phase occurs. 0: OPEN_CIRCUIT The system switches off and discharges the gates. The motor can spin freely. 1: ELECTRICAL_BRAKING The system switches off and, if possible and depending on fault, enables the LS or HS gates, braking the motor electrically.	0, 1	0	RWE
287	DRIVE_FAULT_BEHAVIOUR This value defines the state the system goes to after a fault condition on a motor phase occurs and all retries failed. 0: OPEN_CIRCUIT The system switches off and discharges the LS and HS gates, letting the motor spin freely. 1: ELECTRICAL_BRAKING The system switches off and, if possible and depending on fault, enables the LS or HS gates, braking the motor electrically. 2: MECHANICAL_BRAKING_AND_OPEN_CIRCUIT The system switches off, discharges the LS and HS gates, and, if correctly configured, engages the mechanical brake. 3: MECHANICAL_AND_ELECTRICAL_BRAKING The system switches off, if possible and depending on fault, enables the LS or HS gates, and, if correctly configured, engages the mechanical brake.	0, 1, 2, 3	0	RWE
288	FAULT_HANDLER_NUMBER_OF_RETRIES Maximum number of retries that are performed for every fault that is detected.	0 ... 255	5	RWE

IIT

The IIT monitor is responsible for monitoring energy flowing into the motor. This allows to protect against over temperature situations in sustained load conditions. It continuously records the total current used by the motor, which is determined by both torque and flux. This total motor current value is accessible through the parameter ACTUAL_TOTAL_MOTOR_CURRENT.

The system supports two separate IIT monitoring windows. Within each window, the squared value of the total current is accumulated during the specified winding time. The duration of these winding times is set by the parameters THERMAL_WINDING_TIME_CONSTANT_1 and THERMAL_WINDING_TIME_CONSTANT_2. The cumulative squared current for each window can be checked through the parameters IIT_SUM_1 and IIT_SUM_2.

With the parameters IIT_LIMIT_1 and IIT_LIMIT_2 protective thresholds can be set for each window. If not set to maximum the protections are active indicated by the status flags IIT_ACTIVE_1 and IIT_ACTIVE_2. If the sum exceeds the threshold, the motor performs an emergency stop. The behavior is defined by the parameter DRIVE_FAULT_BEHAVIOUR. If an IIT event occurs, it is indicated by the flags IIT_EXCEEDED_1 or IIT_EXCEEDED_2. *Figure 23* illustrates the feature for two different time windows.

The IIT monitor's update rate is depending on the MOTOR_PWM_FREQUENCY. For more details, see the section *PWM Frequency Configuration*. When setting the THERMAL_WINDING_TIME_CONSTANT values the PWM frequency divider from *Table 26* must be accounted as shown in the formula bellow. The actual time const where desired time constant $t_{ActualTimeConst}$ must be scaled with the frequency divider $d_{freq.}$.

$$\text{THERMAL\_WINDING\_TIME\_CONSTANT[ms]} = t_{ActualTimeConst}[ms] \;/\; d_{freq.}$$

An approximate value for the limit can be calculated with the time constant and maximum continuous current I_c $[A]$ for the window using the formula bellow.

$$IIT\_LIMIT[A^2ms] = I_c^2[A] \times THERMAL\_WINDING\_TIME\_CONSTANT[ms].$$

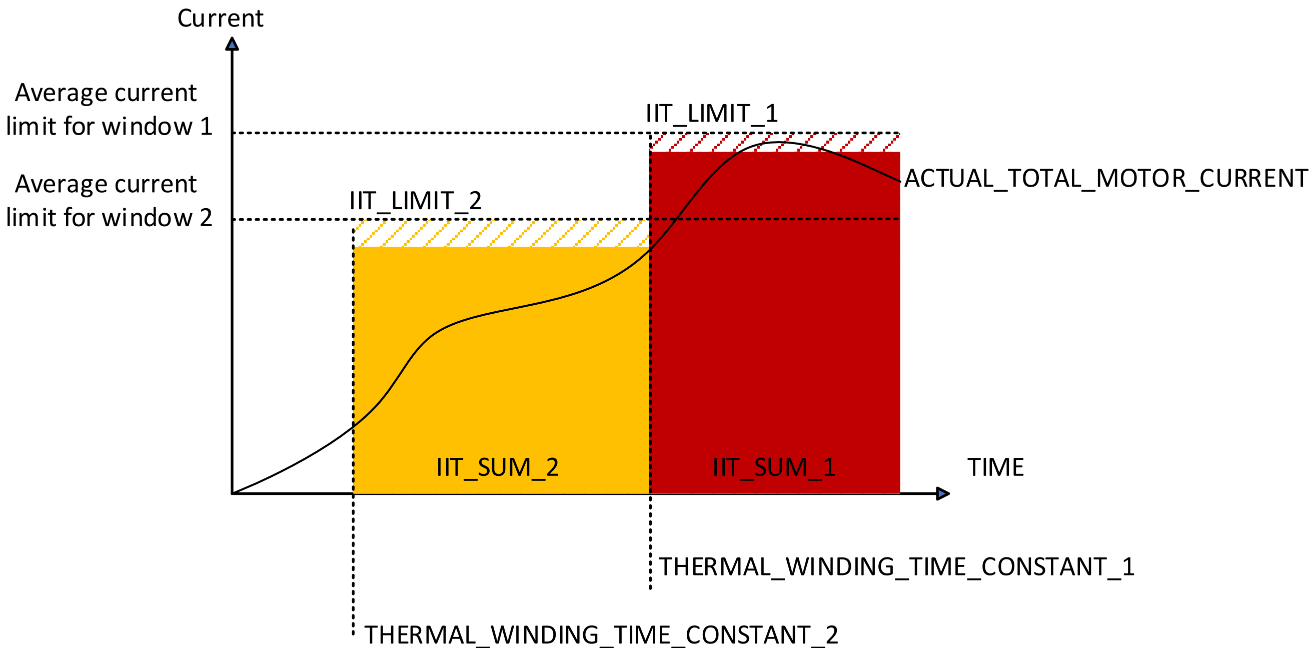

Figure 23. Illustration of the IIT windows

Table 41. Parameters for IIT

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
224	THERMAL_WINDING_TIME_CONSTANT_1 [ms] Thermal winding time constant for the used motor. Used for IIT monitoring. Setting a new value restarts the IIT monitoring.	1000 ... 60000	3000	RWE
225	IIT_LIMIT_1 [A^2 x ms] An actual IIT sum that exceeds this limit leads to trigger the IIT_1_EXCEEDED.	0 ... 4294967295	4294967295	RWE
226	IIT_SUM_1 [A^2 x ms] Actual sum of the IIT monitor 1.	0 ... 4294967295	0	R
227	THERMAL_WINDING_TIME_CONSTANT_2 [ms] Thermal winding time constant for the used motor. Used for IIT monitoring. Setting a new value restarts the IIT monitoring.	1000 ... 60000	6000	RWE
228	IIT_LIMIT_2 [A^2 x ms] An actual IIT sum that exceeds this limit leads to trigger the IIT_2_EXCEEDED.	0 ... 4294967295	4294967295	RWE
229	IIT_SUM_2 [A^2 x ms] Actual sum of the IIT monitor 2.	0 ... 4294967295	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
230	RESET_IIT_SUMS Reset both IIT sums.	0 … 0	0	W
231	ACTUAL_TOTAL_MOTOR_CURRENT [mA] Total current through the motor windings.	0 … 65535	0	R

TEMPERATURE PROTECTIONS

The TMC9660 Parameter Mode incorporates a temperature protection mechanism that mitigates the risk of thermal overrun by integrating readings from both external and internal temperature sensors.

External Temperature Sensor

The TMC9660 Parameter Mode features temperature protections using an external analog temperature sensor that can be connected to AIN3. To convert the parameter EXTERNAL_TEMPERATURE into a voltage use the following formula.

$$\text{Voltage [Volts]} = \frac{\text{par(EXTERNAL\_TEMPERATURE)} \times 2.5V}{2^{16} - 1}$$

Depending on the external sensor this voltage can be converted into a temperature.

In addition to monitoring the value of the external temperature sensor, two threshold values can be configured. The EXTERNAL_TEMPERATURE_WARNING_THRESHOLD parameter sets a threshold that, when exceeded, triggers a temperature warning (EXTERNAL_TEMP_WARNING) in the GENERAL_ERROR_FLAGS. Similarly, the EXTERNAL_TEMPERATURE_SHUTDOWN_THRESHOLD parameter establishes a threshold that initiates a motor shutdown when exceeded, which is then indicated as EXTERNAL_TEMP_EXCEEDED in the GENERAL_ERROR_FLAGS. The DRIVE_FAULT_BEHAVIOUR parameter specifies the motor's shutdown response upon surpassing this threshold.

To reset a motor after an external temperature shutdown, the EXTERNAL_TEMP_EXCEEDED flag must be cleared before attempting to restart the motor.

On-Chip Temperature Sensor

The TMC9660 Parameter Mode also features monitoring the internal chip temperature. The parameter CHIP_TEMPERATURE can be converted into temperature in degree Celsius using the formula below.

$$\text{Temperature [°C]} = \text{par(CHIP\_TEMPERATURE)} \times 0.01615 - 268.15$$

In addition to monitoring the value of the chip temperature, two threshold values can be configured. The CHIP_TEMPERATURE_WARNING_THRESHOLD parameter sets a threshold that, when exceeded, triggers a temperature warning (CHIP_TEMP_WARNING) in the GENERAL_ERROR_FLAGS. Similarly, the CHIP_TEMPERATURE_SHUTDOWN_THRESHOLD parameter establishes a threshold that initiates a motor shutdown when exceeded, which is then indicated as CHIP_TEMP_EXCEEDED in the GENERAL_ERROR_FLAGS. The DRIVE_FAULT_BEHAVIOUR parameter specifies the motor's shutdown response upon surpassing this threshold.

To reset a motor after an external temperature shutdown, the CHIP_TEMP_EXCEEDED flag must be cleared before attempting to restart the motor.

Table 42. Parameters for temperature protection

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
293	EXTERNAL_TEMPERATURE The actual temperature at the external temperature sensor.	0 ... 65535	0	R
294	EXTERNAL_TEMPERATURE_SHUTDOWN_THRESHOLD The temperature threshold at which the motor driver is shut down.	0 ... 65535	65535	RWE
295	EXTERNAL_TEMPERATURE_WARNING_THRESHOLD The temperature threshold above which the warning flag is set.	0 ... 65535	65535	RWE
296	CHIP_TEMPERATURE The actual temperature of the chip.	0 ... 65535	0	R
297	CHIP_TEMPERATURE_SHUTDOWN_THRESHOLD The temperature threshold at which the motor driver is shut down.	0 ... 65535	65535	RWE
298	CHIP_TEMPERATURE_WARNING_THRESHOLD The temperature threshold above which the warning flag is set.	0 ... 65535	65535	RWE

HEARTBEAT MONITORING

The TMC9660 Parameter Mode supports a heartbeat monitor that can be configured using the global parameters HEARTBEAT_MONITORING_CONFIG and HEARTBEAT_MONITORING_TIMEOUT in bank 0. This feature monitors the communication interfaces specified in the configuration parameter. If no datagram is received within the timeout period, the system initiates a motor shutdown. The behavior of the shutdown is defined by the parameter DRIVE_FAULT_BEHAVIOUR. The status flag HEARTBEAT_STOPPED is raised.

Table 43. Parameters in global bank 0 for heartbeat monitoring

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
3	HEARTBEAT_MONITORING_CONFIG Configuration to enable heartbeat monitoring. In mode 3 the heartbeat is considered stopped if both UART and SPI communication is stopped. While at least one of them is beating no safe stop is issued. 0: DISABLED 1: TMCL_UART_INTERFACE 2: SPI_INTERFACE 3: TMCL_UART_AND_SPI_INTERFACE	0, 1, 2, 3	0	RWE
4	HEARTBEAT_MONITORING_TIMEOUT [ms] Timeout above which a heartbeat is consider stopped.	1 ... 4294967295	100	RWE

BRAKE CHOPPER

The brake chopper can be used to dissipate energy over a brake resistor in case of an overvoltage condition. See *Figure 24* for an application example. For this feature to be available it must be configured in the boot configuration and an external MOSFET as well as the brake resistor must be wired up accordingly.

The feature can be enabled using the parameter ENABLE_BRAKE_CHOPPER. When the supply voltage rises above the BRAKE_CHOPPER_VOLTAGE_LIMIT the MOSFET gets activated and the excess energy gets dissipated by current flowing through the brake resistor. As soon as the supply voltage drops below the voltage limit minus the hysteresis the MOSFET gets switched off again.

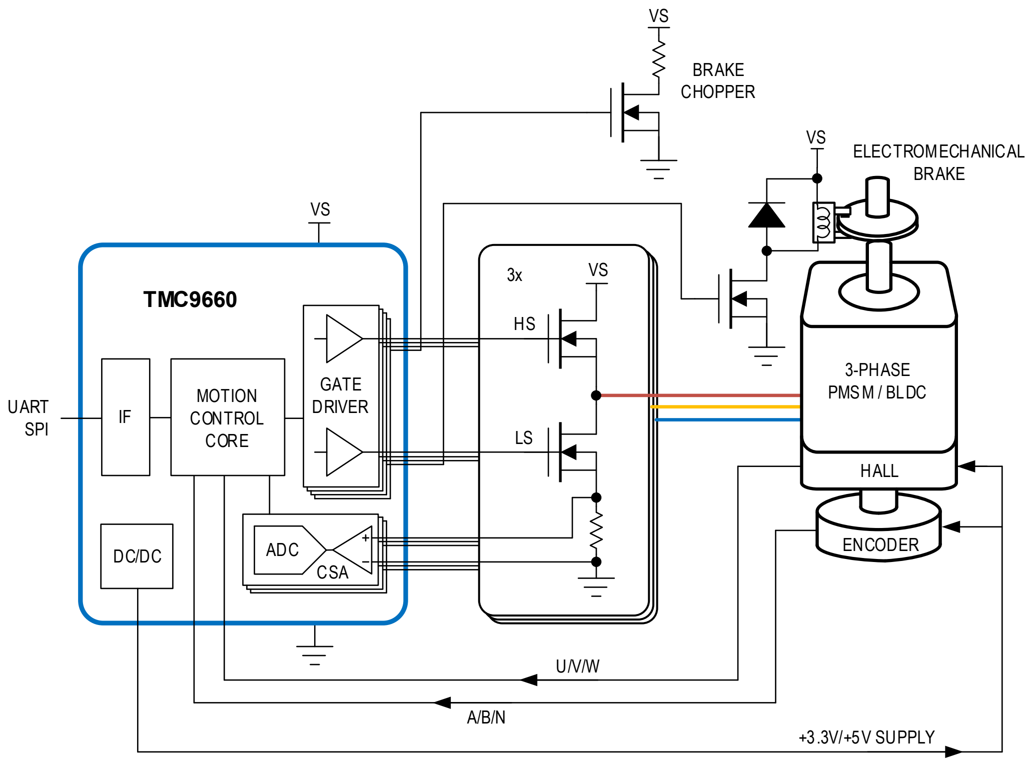

Figure 24. Application example with BLDC and Brake Chopper as well as external Electromechanical Brake

Table 44. Parameters for brake chopper

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
212	BRAKE_CHOPPER_ENABLE Enable the brake chopper functionality. False: DISABLED True: ENABLED	0, 1	0	RWE
213	BRAKE_CHOPPER_VOLTAGE_LIMIT [0.1V] If the brake chopper is enabled and supply voltage exceeds this value, the brake chopper output is activated.	50 ... 1000	260	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
214	BRAKE_CHOPPER_HYSTERESIS [0.1V] An activated brake chopper is deactivated if the actual supply voltage is lower than BRAKE_CHOPPER_VOLTAGE_LIMIT - BRAKE_CHOPPER_HYSTERESIS.	0 ... 50	5	RWE

MECHANICAL BRAKE

If configured in the boot configuration, controlling an external brake is supported as shown in *Figure 24*. The external MOSFET and brake need to be wired up accordingly. A GPIO output option with external Gate-Driver is available as well in case all half-bridges are used with a Stepper Motor. The external brake is controlled through a PWM signal from 0-99% duty cycle. By default, the brake output has zero percent duty cycle. This should result in a locked state for most brake systems. However, the brake output can be inverted if needed by setting the parameter INVERT_BRAKE_OUTPUT.

INVERT_BRAKE_OUTPUT needs the following setting, depending on the used output option:

- Output on GPIO: NORMAL (0)
- Output on Y2-LS: INVERTED (1)

To release the mechanical brake, the parameter RELEASE_BRAKE must be set to 1. When a brake release is triggered, a PWM with BRAKE_RELEASING_DUTY_CYCLE is applied to the brake output pin for at least the BRAKE_RELEASING_DURATION. Afterwards, a PWM with BRAKE_HOLDING_DUTY_CYCLE is applied as long as the RELEASE_BRAKE parameter is set. This release cycle is shown in *Figure 25*.

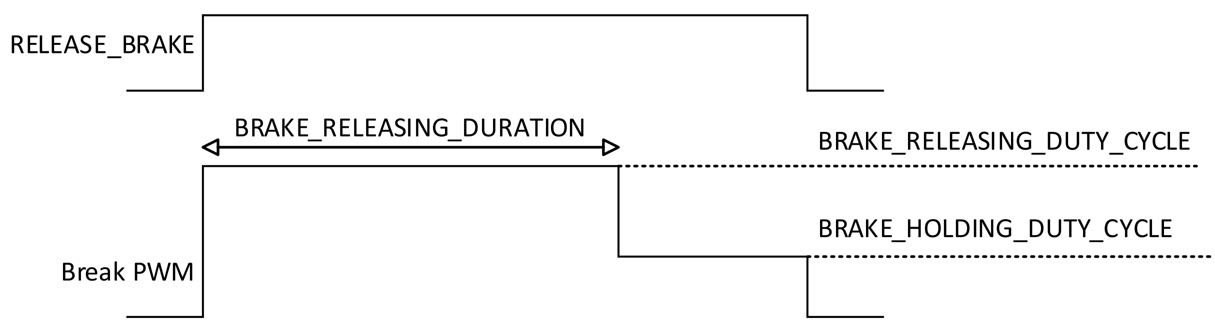

Figure 25. Mechanical brake release sequence

Therefore, the mechanical brake feature is always in one of the following three internal states:

- BRAKE_OFF
- BRAKE_RELEASING
- BRAKE_HOLDING

In general, the PWM polarity, the release duration and both duty cycles can be written at any time, but the newly written values only come into effect on the transition to another state, with the following exceptions:

- While in the state BRAKE_HOLDING, the hold duty cycle gets updated if the user overwrites it.
- While in the state BRAKE_OFF, the polarity gets updated if the user overwrites it.

Table 45. Mechanical brake parameters

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
216	RELEASE_BRAKE Release the external brake by applying a PWM signal. False: BRAKE_PWM_DEACTIVATED True: BRAKE_PWM_ACTIVATED	0, 1	0	RWE
217	BRAKE_RELEASING_DUTY_CYCLE [%] Set the duty cycle of the first PWM phase for releasing the brake.	0 … 99	75	RWE
218	BRAKE_HOLDING_DUTY_CYCLE [%] Set the duty cycle of the second PWM phase to hold the brake.	0 … 99	11	RWE
219	BRAKE_RELEASING_DURATION [ms] Set the duration the brake PWM uses the first duty cycle.	0 … 65535	80	RWE
221	INVERT_BRAKE_OUTPUT Invert the brake output. False: NORMAL True: INVERTED	0, 1	0	RWE

AUTOMATIC HOMING

The automatic homing/reference search routine allows to automatically find the limit switches and use them as reference for the actual position. It can be configured using the parameters in *Table 47*.

TMC9660 Parameter Mode supports eight different search patterns. These can be configured using the parameter REFERENCE_SEARCH_MODE and are illustrated in *Figure 26*. Two different search speeds are supported. The velocity REFERENCE_SEARCH_SPEED is used to find the reference switch. The parameter REFERENCE_SWITCH_SPEED should be configured to be slower and is used to increase accuracy when finding the exact switch position.

The parameters RIGHT_LIMIT_SWITCH_POSITION, HOME_SWITCH_POSITION, LAST_REFERENCE_POSITION allow the corresponding switch positions to be read out.

If a left limit switch is found, the switch position is set as actual position zero. The reference positions for the left and right switches are always the points at which a switch press was detected first. This is even true for the modes where the left switch is approached from both sides and the motor is driven to the center of the left switch afterwards. In that case the actual motor position at the center of the left switch is at an offset to the zero position. If the zero position is meant to be the center position of the left switch, the actual position must be manually set to zero after the reference search is finished.

In contrast, in the modes where the search for a home switch is performed, the motor is driven to the center position of the home switch and the actual position is automatically set to zero at that home switch center position.

After configuration, a reference search can be triggered. This is done by using the TMCL command RFS (13). The type argument specifies the desired operation. It can be used to start or stop the search, or to request the current state. See *Table 46* for the usage of the command.

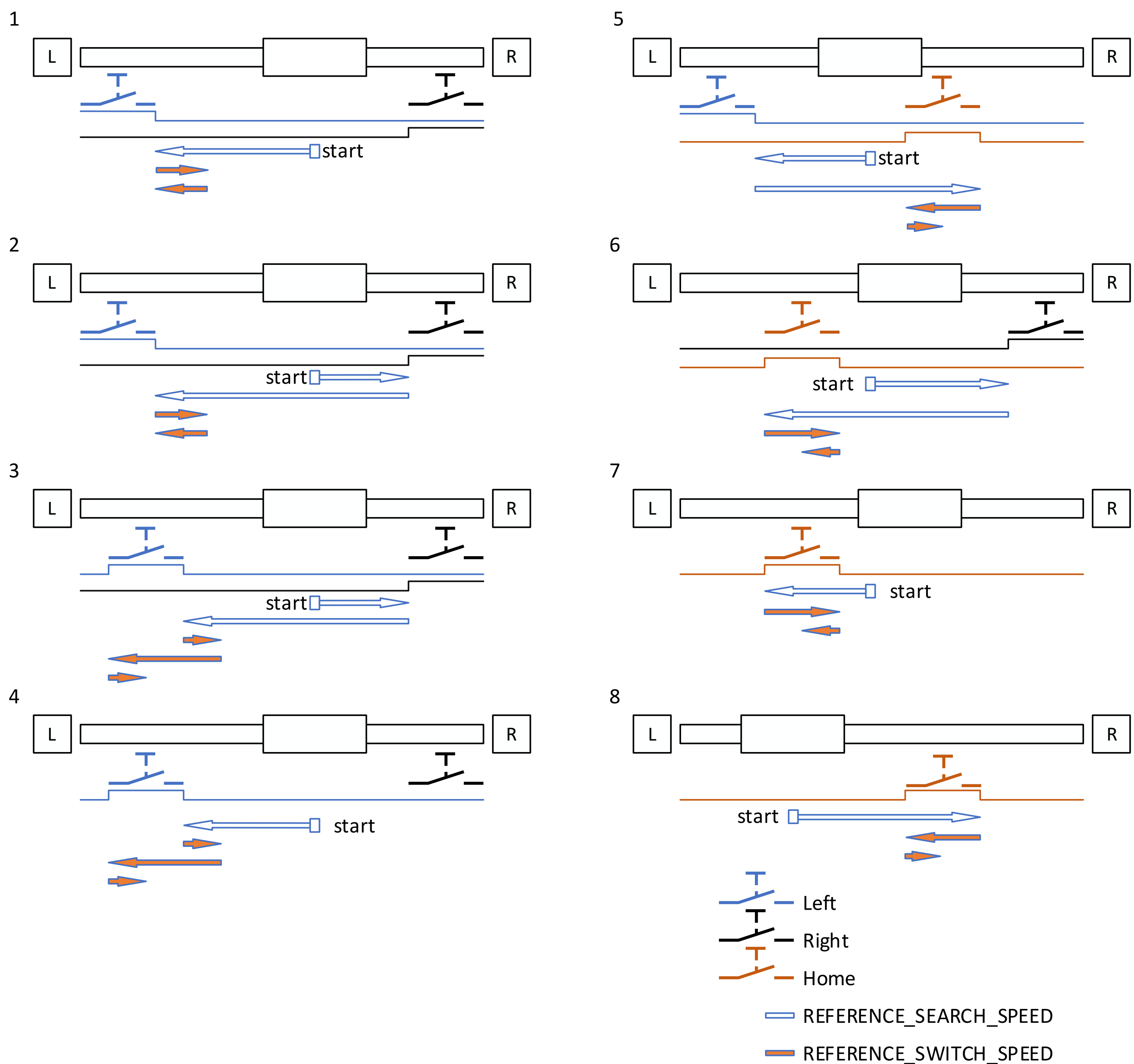

Figure 26. Reference search modes configured by the parameter REFERENCE_SEARCH_MODE

Table 46. TMCL command RFS structure

TYPE	COMMAND
0	Start reference search.
1	Stop reference search.
2	Return reference search status. 0: Idle 1: Start reference drive. 2: Start drive to right limit switch (fast). 3: Wait until right switch was reached. 4: Start drive to left limit switch (fast). 5: Wait until left switch was reached. 6: Drive out of left switch (slowly). 7: Wait until left switch was exited then drive slowly into again. 8: Wait until left switch was reached again then drive to switch position. 9: Wait until position was reached then set position to zero. 10: Wait until switch is pushed again. 11: Wait until the other side of the switch was reached. 12: Reserved 13: Wait until center of switch was reached. 14: Reference drive finished: restore settings. 15: Stop reference drive.

Table 47. Parameters for homing/reference search

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
165	REFERENCE_SWITCH_SEARCH_MODE Determine the reference switch search sequence. 1: LEFT_SWITCH Search for left limit switch. 2: RIGHT_SWITCH_LEFT_SWITCH Search for right limit switch then search for left limit switch. 3: RIGHT_SWITCH_LEFT_SWITCH_BOTH_SIDES Search for right limit switch then approach left limit switch from both sides. 4: LEFT_SWITCH_BOTH_SIDES Approach left limit switch from both sides. 5: HOME_SWITCH_NEG_DIR_LEFT_END_SWITCH Search for home switch in negative direction, turn around if left end switch detected. 6: HOME_SWITCH_POS_DIR_RIGHT_END_SWITCH Search for home switch in positive direction, turn around if right end switch detected. 7: HOME_SWITCH_NEG_DIR_IGNORE_END_SWITCH Search for home switch in negative direction, ignore end switch. 8: HOME_SWITCH_POS_DIR_IGNORE_END_SWITCH Search for home switch in positive direction, ignore end switch.	1, 2, 3, 4, 5, 6, 7, 8	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
166	REFERENCE_SWITCH_SEARCH_SPEED Speed used for the reference switch search sequence.	-134217728 ... 134217727	0	RWE
167	REFERENCE_SWITCH_SPEED Lower speed used e.g. for positioning the motor at a reference switch position.	-134217728 ... 134217727	0	RWE
168	RIGHT_LIMIT_SWITCH_POSITION Right limit switch position.	-2147483648 ... 2147483647	0	R
169	HOME_SWITCH_POSITION Home switch position.	-2147483648 ... 2147483647	0	R
170	LAST_REFERENCE_POSITION Last reference position.	-2147483648 ... 2147483647	0	R

STEP/DIR

As a target movement interface, a STEP/DIR interface can be configured. The feature and the corresponding GPIO pins must be configured in the boot configuration.

The STEP input is a pulse signal where each pulse influences the target position based on the number of step signal pulses detected, the micro-step configuration and the state of the DIR input.

The DIR input determines the direction of the motor rotation. If the DIR input is low, the target position gets incremented; if it is high, the target position gets decremented.

The TMC9660 Parameter Mode supports micro-stepping, step extrapolation, and velocity feedforward for STEP/DIR. The features are described below, and a full list of parameters can be found in *Table 48*.

To improve the resolution, micro-stepping can be configured. A full step corresponds to a quarter electrical revolution. Micro-stepping applies only a partial step for each detected step input. The micro-step size is determined by the STEP/DIR_STEP_DIVIDER_SHIFT parameter. The smallest supported step size is 1/1024 of a full step.

An optional extrapolation between two consecutive step pulses can be enabled. This feature can improve the smoothness of the movement. If enabled, a step signal is divided into up to 1024 micro-steps and applied over the period of the step. The applied micro-steps are always 1/1024 of a full step, thus the division depends on the selected micro-step configuration. The extrapolation leads to overdriving the target after the last step occurred. After a timeout, defined by the parameter STEP/DIR_STEP_SIGNAL_TIMEOUT_LIMIT, the target position gets corrected. For high velocities, the extrapolation has a diminishing effect on the smoothness of rotation. A maximum velocity for the extrapolation can be configured using the parameter STEP/DIR_MAXIMUM_EXTRAPOLATION_VELOCITY. Beyond this velocity, the extrapolation is not applied anymore. The influence of the extrapolation and the correction behavior after timeout are illustrated in *Figure 27*.

For higher tracking accuracy, a velocity feedforward can be enabled. It is activated by the VELOCITY_FEEDFORWARD parameter. If activated, a velocity signal is calculated based on the step input and pre-fed to the velocity controller.

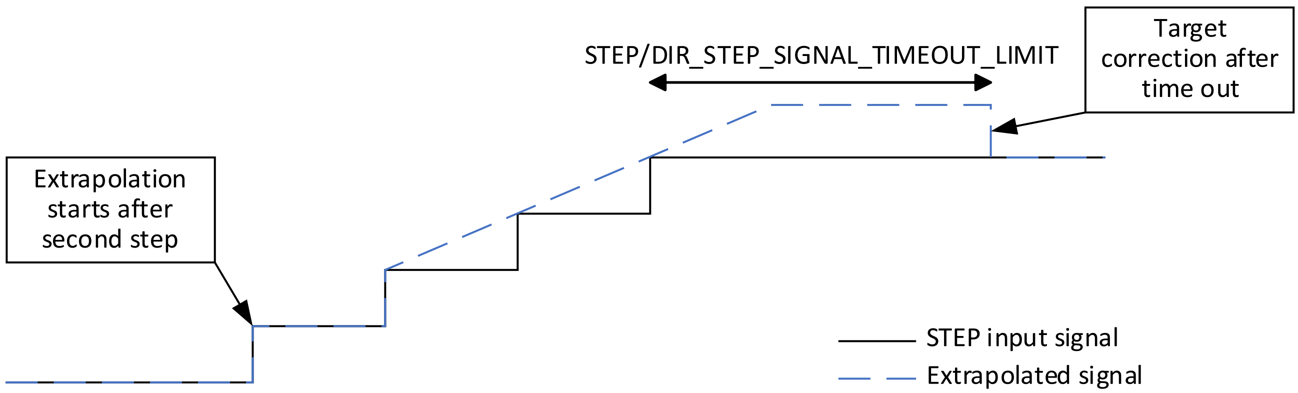

Figure 27. STEP/DIR extrapolation behavior

Table 48. Parameters for Step/Dir

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
68	VELOCITY_FEEDFORWARD_ENABLE Enable the velocity feedforward feature. False: DISABLED True: ENABLED	0, 1	0	RWE
205	STEP_DIR_STEP_DIVIDER_SHIFT Configure step/dir to use between 1 and 1024 microsteps per full step. 0: STEP_MODE_FULL 1: STEP_MODE_HALF 2: STEP_MODE_QUARTER 3: STEP_MODE_1_8TH 4: STEP_MODE_1_16TH 5: STEP_MODE_1_32ND 6: STEP_MODE_1_64TH 7: STEP_MODE_1_128TH 8: STEP_MODE_1_256TH 9: STEP_MODE_1_512TH 10: STEP_MODE_1_1024TH	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10	0	RWE
206	STEP_DIR_ENABLE Enable the Step/Dir input functionality. False: DISABLED True: ENABLED	0, 1	0	RW
207	STEP_DIR_EXTRAPOLATION_ENABLE Enable the Step/Dir extrapolation feature. False: DISABLED True: ENABLED	0, 1	0	RW

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
208	STEP_DIR_STEP_SIGNAL_TIMEOUT_LIMIT [ms] Step signal timeout limit.	1 … 2000	1000	RW
209	STEP_DIR_MAXIMUM_EXTRAPOLATION_VELOCITY [eRPM] Maximum velocity up to which extrapolation is used.	0 … 2147483647	2147483647	RW

SCRIPT

To use the scripting feature, an external memory device must be configured in the boot configuration. Additionally, a valid partition table must be written on the memory device. For a comprehensive list of all the commands and their corresponding numbers, see *Table 18* in the section *All TMCL Operation and Reply Codes*. It is strongly recommended to use the script feature in combination with the TMCL-IDE.

Download and Execute a Script

The first step to program a script is to enter the download mode by sending the operation "DownloadStart". All scriptable operations after this are stored to memory instead of being executed. Starting the download automatically erases the previous content of the script memory. To end the download, send the operation "DownloadEnd". The script addresses are in the order the storable operations are received, starting from zero.

A programmed script can be read out using the "ReadMem" operation with the command address as value. The address corresponds to the previous order of writing the script operations.

To start script execution, the operation "ApplRun" must be sent. The type and address fields allow to specify the address the application starts from. The script execution can be stopped with the operation "ApplStop". When the operation "ApplReset" is sent, the script execution is stopped, and the program counter gets reset to zero.

If a script should be started automatically, the global parameter AUTO_START_MODE must be set and the system state must be stored afterwards as described in the section *Storing System Settings in External Memory*. After the next startup, the script execution starts automatically.

The current state of script execution can be read out by using the operation "GetStatus". Depending on the type field different status information can be read. *Table 49* shows how the reply value has to be interpreted depending on the type value.

Table 49. Bytes returned by the "GetStatus" operation based on the type

TYPE NR.	BYTE 0	BYTE 1	BYTE 2	BYTE 3
1	Program counter		Reserved	Current script mode
2	Accumulator value			
3	X-Register value			

Table 50. Script states

NR.	SCRIPT MODE	DESCRIPTION
0	Idle	Idle state. No script execution.
1	Run	Script is running.
2	Step	Run single command, then switching to debug mode.
3	Reserved	-

NR.	SCRIPT MODE	DESCRIPTION
4	Download	Every command sent in gets stored to external memory instead of execution
5	Debug	Pause execution. Occurs after TMCL_STEP or a breakpoint was hit.

Breakpoints and Step Operation

For debugging scripts, up to three breakpoints can be inserted. Sending the “Breakpoint” operation with type zero allows setting a breakpoint at a specific address. The breakpoint can be removed by using type one and the same address. By sending type two, all breakpoints can be removed. Additionally, it is possible to step through the script. If the script is halted, using the “ApplStep” operation allows executing only the next command.

Parameter Commands

To set and get parameters, the normal “SAP” and “GAP” operations are used. For global parameters, “GGP” and “SGP” are used.

When the get commands “GAP” and “GGP” are executed from a script, the internal accumulator value gets updated. This value can be used in comparison and calculation commands. To write the accumulator value to a parameter or a global parameter, the operations “AAP” and “AGP” must be used.

Wait for Events

The script execution can be paused to wait for an event. This feature can be scripted using the “WAIT” operations. The type of the wait operations specifies the condition that must be met to continue script execution. The available conditions are listed in *Table 51*. With the value of the operation a timeout value must be specified. This time is measured in 10ms per tick. If a wait times out the execution proceeds, and the error time out flag gets raised. A conditional jump allows to branch based on the timeout.

Table 51. Event conditions

NR.	CONDITION	DESCRIPTION
0	TICKS	Accumulator value is zero
4	RFS	Accumulator value is not zero
5	LATCHED	Is equal
6	RAMPER_VELOCITY_REACHED	Is not equal
7	RAMPER_POSITION_REACHED	Is greater than
8	STOPLEFT	Is greater equal
9	STOPRIGHT	Is lower than
10	STOPHOME	I lower equal

Branch Commands

Branching is supported in TMCL script to enable complex script structures. Branching can happen conditional or unconditional.

Jump to Address

The operation “JA” allows for an unconditional jump to an address specified by the operation’s value. This can also be used to implement a loop. Conditional jumps can be implemented using the operation “JC”. The condition for a jump must be specified as the type. If the jump condition depends on a comparison, the “COMP” operation must be executed before the “JC” operation. The “COMP” operation is used to provide the value to compare against.

Table 52. Conditional jump “JC” conditions

NR.	CONDITION	DESCRIPTION
0	ZE	Accumulator value is zero
1	NE	Accumulator value is not zero
2	EQ	Is equal
3	NE	Is not equal
4	GT	Is greater than
5	GE	Is greater equal
6	LT	Is lower than
7	LE	I lower equal
8	ETO	Error time out flag. Wait command timed out previously. The error flag can be cleared with operation "CLE".

Interrupts on Events

Interrupts allow the normal code execution to be stopped and branched off based on an event. Interrupts can occur on three separate timers, the stop inputs, and all GPIO inputs. Each interrupt must be configured with a vector specifying the target address to jump using the "VECT" operation. Any interrupt needs a separate activation, and interrupts must be set active globally by enabling the global interrupt using the operation "EI". The interrupts itself must be configured using the parameters in the global parameter bank 3. See *Table 53* for a complete list.

Table 53. Interrupt numbers

NR.	TYPE	DESCRIPTION
0	Timer 0	Interrupt on end of timer 0.
1	Timer 1	Interrupt on end of timer 1.
2	Timer 2	Interrupt on end of timer 2.
3	RAMPER_LATCHED	React to latched flag goes.
4	RAMPER_EVENT_STOP_DEVIATION	React to stop on deviation.
5	VELOCITY_REACHED	React to velocity reached flag.
6	POSITION_REACHED	React to position reached flag.
7	DRIVER_TEMP_EXEEDED	React to timer interrupt.
8	IIT_EXEEDED_1	React to IIT window 1 flag.
9	IIT_EXEEDED_2	React to IIT window 2 flag.
10	STOP_LEFT	React to stop left input
11	STOP_RIGHT	React to stop right input
12	STOP_HOME	React to stop home input
13	INPUT_0	React to GPIO 0
14	INPUT_1	React to GPIO 1
15	INPUT_2	React to GPIO 2
16	INPUT_3	React to GPIO 3
17	INPUT_4	React to GPIO 4
18	INPUT_5	React to GPIO 5
19	INPUT_6	React to GPIO 6
20	INPUT_7	React to GPIO 7
21	INPUT_8	React to GPIO 8
22	INPUT_9	React to GPIO 9
23	INPUT_10	React to GPIO 10

24	INPUT_11	React to GPIO 11
25	INPUT_12	React to GPIO 12
26	INPUT_13	React to GPIO 13
27	INPUT_14	React to GPIO 14
28	INPUT_15	React to GPIO 15
29	INPUT_16	React to GPIO 16
30	INPUT_17	React to GPIO 17
31	INPUT_18	React to GPIO 18
32	Global	Enable / Disable interrupt globally

Call Subroutine

Calling subroutines is also an option to realize subroutines. Subroutines can be called using the operation "CSUB" to call a subroutine and "RSUB" to return from a subroutine.

Calculation Commands

Multiple commands are available that allow calculations to be performed. All calculation commands are listed in *Table 54*. Depending on the operation, the values that are used in the calculation operation vary. The types that can be used with the operations are listed in *Table 55*.

Table 54. All TMCL calculation commands

NUMBER	OPERATION	TYPE	MOTOR/ BANK	VALUE	DESCRIPTION
33	CALCX	type	-	-	Arithmetical operation between accumulator and X-register
40	CALCVV	type	user variable 1	user variable 2	Arithmetical operation between two user variables
41	CALCVA	type	user variable	-	Arithmetical operation between user variable and accumulator
42	CALCAV	type	user variable	-	Arithmetical operation between accumulator and user variable
43	CALCVX	type	user variable	-	Arithmetical operation between user variable and X register
44	CALCXV	type	user variable	-	Arithmetical operation between X register and user variable
45	CALCV	type	-	value	Arithmetical operation between user variable and direct value

Table 55. Calculation operations available

NUMBER	TYPE	DESCRIPTION	FORMULA
0	ADD	Addition	Value_1 = Value_1 + Value_2
1	SUB	Subtract	Value_1 = Value_1 - Value_2
2	MUL	Multiply	Value_1 = Value_1 x Value_2
3	DIV	Divide	Value_1 = Value_1 / Value_2
4	MOD	Modulo	Value_1 = Value_1 % Value_2
5	AND	Bitwise and	Value_1 = Value_1 & Value_2
6	OR	Bitwise or	Value_1 = Value_1 | Value_2
7	XOR	Bitwise xor	Value_1 = Value_1 ^ Value_2
8	NOT	Bitwise not	Value_1 ~= Value_2
9	LOAD	Load value	Value_1 = Value_2

HIBERNATION AND WAKEUP

The TMC9660 Parameter Mode can be sent to a low-power hibernation state. In this state, all memory not saved to external nonvolatile memory is lost, including the boot configuration if it is not burned into the part. Using the global parameter ENABLE_WAKE_PIN_CONTROL sends the system into a deep sleep state. A wake pin configured in the boot configuration wakes up the system when toggled. The global parameter GO_TO_TIMEOUT_POWER-DOWN_STATE can send the part into the hibernation state for a predefined time.

Table 56. Global parameters in bank 0 for hibernation and wakeup

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
10	WAKE_PIN_CONTROL_ENABLE Enable TMC9660 WAKE pin functionality to be able to be put the chip into power-down state and wake the chip up again later. False: DISABLED True: ENABLED	0, 1	0	RWE
11	GO_TO_TIMEOUT_POWER_DOWN_STATE Use this parameter to put TMC9660 into power-down state for a given time period. Note that if Pin wakeup is configured, the WAKE pin must be pulled to GND in order to power down and pulling the WAKE pin back up before the time elapses, the TMC9660 already wakes up. 0: T_250_MILLISEC 1: T_500_MILLISEC 2: T_1_SEC 3: T_2_SEC 4: T_4_SEC 5: T_8_SEC 6: T_16_SEC 7: T_32_SEC	0, 1, 2, 3, 4, 5, 6, 7	0	W

PARAMETERS

Table 57 shows a full list of all parameters that can be set using the commands SAP and read using GAP. A “RWE” in the read/write collum indicates storability in nonvolatile external memory.

Table 57. Full list of parameters

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
0	MOTOR_TYPE Selected motor type. PWM must be turned off to change this. 0: NO_MOTOR 1: DC_MOTOR 2: STEPPER_MOTOR 3: BLDC_MOTOR	0, 1, 2, 3	0	RWE
1	MOTOR_POLE_PAIRS Pole pair count of the motor.	0 ... 127	1	RWE
2	MOTOR_DIRECTION Motor direction bit. Inverts the motor direction. False: NOT_INVERTED True: INVERTED	0, 1	0	RWE
3	MOTOR_PWM_FREQUENCY [Hz] Set the frequency of the motor PWM.	10000 ... 100000	25000	RWE
4	COMMUTATION_MODE Selected FOC operation mode depending on feedback used for commutation. 0: SYSTEM_OFF 1: SYSTEM_OFF_LOW_SIDE_FETS_ON 2: SYSTEM_OFF_HIGH_SIDE_FETS_ON 3: FOC_OPENLOOP_VOLTAGE_MODE 4: FOC_OPENLOOP_CURRENT_MODE 5: FOC_ABN 6: FOC_HALL_SENSOR 7: RESERVED 8: FOC_SPI_ENC	0, 1, 2, 3, 4, 5, 6, 7, 8	0	RW
5	OUTPUT_VOLTAGE_LIMIT PID UQ/UD output limit for circular limiter.	0 ... 32767	8000	RWE
6	MAX_TORQUE [mA] Maximum motor torque. Note: This value can be temporarily exceeded marginally due to the operation of the current regulator.	0 ... 65535	2000	RWE
7	MAX_FLUX [mA] Max. motor flux. Note: This value can be temporarily exceeded marginally due to the operation of the current regulator.	0 ... 65535	2000	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
8	PWM_SWITCHING_SCHEME PWM switching scheme. 0: STANDARD 1: SVPWM 2: FLAT_BOTTOM	0, 1, 2	1	RWE
9	IDLE_MOTOR_PWM_BEHAVIOR Configure if the PWM should be off (high-z) or on (all motor phases same voltage) in commutation mode "System Off". False: PWM_ON_WHEN_MOTOR_IDLE True: PWM_OFF_WHEN_MOTOR_IDLE	0, 1	1	RWE
12	ADC_SHUNT_TYPE Shunt type used for ADC measurements. 0: INLINE_UVW 1: INLINE_VW 2: INLINE_UW 3: INLINE_UV 4: BOTTOM_SHUNTS	0, 1, 2, 3, 4	4	RWE
13	ADC_I0_RAW Raw ADC measurement of the ADC I0 shunt.	-32768 ... 32767	0	R
14	ADC_I1_RAW Raw ADC measurement of the ADC I1 shunt.	-32768 ... 32767	0	R
15	ADC_I2_RAW Raw ADC measurement of the ADC I2 shunt.	-32768 ... 32767	0	R
16	ADC_I3_RAW Raw ADC measurement of the ADC I3 shunt.	-32768 ... 32767	0	R
17	CSA_GAIN_ADC_I0_TO_ADC_I2 Current sense amplifier gain for ADC I0, I1 and I2. 0: GAIN_5X 1: GAIN_10X 2: GAIN_20X 3: GAIN_40X 4: GAIN_1X_BYPASS_CSA	0, 1, 2, 3, 4	1	RWE
18	CSA_GAIN_ADC_I3 Current sense amplifier gain for ADC I3. 0: GAIN_5X 1: GAIN_10X 2: GAIN_20X 3: GAIN_40X 4: GAIN_1X_BYPASS_CSA	0, 1, 2, 3, 4	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
19	CSA_FILTER_ADC_I0_TO_ADC_I2 Current sense amplifier filter for ADC I0, I1 and I2. 0: T_0_55_MICROSEC 1: T_0_75_MICROSEC 2: T_1_0_MICROSEC 3: T_1_35_MICROSEC	0, 1, 2, 3	0	RWE
20	CSA_FILTER_ADC_I3 Current sense amplifier filter for ADC I3. 0: T_0_55_MICROSEC 1: T_0_75_MICROSEC 2: T_1_0_MICROSEC 3: T_1_35_MICROSEC	0, 1, 2, 3	0	RWE
21	CURRENT_SCALING_FACTOR Current scaling factor converting internal units to real-world units.	1 ... 65535	520	RWE
22	PHASE_UX1_ADC_MAPPING Mapping ADC to UX1. 0: ADC_I0 1: ADC_I1 2: ADC_I2 3: ADC_I3	0, 1, 2, 3	0	RWE
23	PHASE_VX2_ADC_MAPPING Mapping ADC to VX2. 0: ADC_I0 1: ADC_I1 2: ADC_I2 3: ADC_I3	0, 1, 2, 3	1	RWE
24	PHASE_WY1_ADC_MAPPING Mapping ADC to WY1. 0: ADC_I0 1: ADC_I1 2: ADC_I2 3: ADC_I3	0, 1, 2, 3	2	RWE
25	PHASE_Y2_ADC_MAPPING Mapping ADC to Y2. 0: ADC_I0 1: ADC_I1 2: ADC_I2 3: ADC_I3	0, 1, 2, 3	3	RWE
26	ADC_I0_SCALE Scaling applied to ADC I0.	1 ... 32767	1024	RWE
27	ADC_I1_SCALE Scaling applied to ADC I1.	1 ... 32767	1024	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
28	ADC_I2_SCALE Scaling applied to ADC I2.	1 ... 32767	1024	RWE
29	ADC_I3_SCALE Scaling applied to ADC I3.	1 ... 32767	1024	RWE
30	ADC_I0_INVERTED Invert the reading of ADC I0. False: NOT_INVERTED True: INVERTED	0, 1	1	RWE
31	ADC_I1_INVERTED Invert the reading of ADC I1. False: NOT_INVERTED True: INVERTED	0, 1	1	RWE
32	ADC_I2_INVERTED Invert the reading of ADC I2. False: NOT_INVERTED True: INVERTED	0, 1	1	RWE
33	ADC_I3_INVERTED Invert the reading of ADC I3. False: NOT_INVERTED True: INVERTED	0, 1	1	RWE
34	ADC_I0_OFFSET Offset applied to ADC I0 measurement.	-32768 ... 32767	0	RWE
35	ADC_I1_OFFSET Offset applied to ADC I1 measurement.	-32768 ... 32767	0	RWE
36	ADC_I2_OFFSET Offset applied to ADC I2 measurement.	-32768 ... 32767	0	RWE
37	ADC_I3_OFFSET Offset applied to ADC I3 measurement.	-32768 ... 32767	0	RWE
38	ADC_I0 Scaled and offset compensated ADC I0 measurement.	-32768 ... 32767	0	R
39	ADC_I1 Scaled and offset compensated ADC I1 measurement.	-32768 ... 32767	0	R
40	ADC_I2 Scaled and offset compensated ADC I2 measurement.	-32768 ... 32767	0	R
41	ADC_I3 Scaled and offset compensated ADC I3 measurement.	-32768 ... 32767	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
45	OPENLOOP_ANGLE Phi_e calculated by the ramper hardware. Used for commutation in openloop modes.	-32768 ... 32767	0	R
46	OPENLOOP_CURRENT [mA] Openloop current applied in openloop, current mode.	0 ... 65535	1000	RWE
47	OPENLOOP_VOLTAGE Openloop voltage applied in openloop, voltage mode.	0 ... 16383	0	RWE
50	ACCELERATION_FF_GAIN Gain applied to acceleration feedforward.	0 ... 65535	8	RWE
51	ACCELERATION_FF_SHIFT Shift applied to acceleration feedforward. 0: NO_SHIFT 1: SHIFT_4_BIT 2: SHIFT_8_BIT 3: SHIFT_12_BIT 4: SHIFT_16_BIT 5: SHIFT_20_BIT 6: SHIFT_24_BIT	0, 1, 2, 3, 4, 5, 6	4	RWE
52	RAMP_ENABLE Enable the application of acceleration and deceleration ramps. False: DISABLED True: ENABLED	0, 1	0	RWE
53	DIRECT_VELOCITY_MODE Specify the control loop structure for velocity mode. Directly regulating the velocity or regulating on a constantly calculated target position. False: DISABLED True: ENABLED	0, 1	1	RWE
54	RAMP_AMAX [internal] Acceleration in top part of eight-point ramp.	1 ... 8388607	1000	RWE
55	RAMP_A1 [internal] First acceleration in eight-point ramp.	1 ... 8388607	8000	RWE
56	RAMP_A2 [internal] Second acceleration in eight-point ramp.	1 ... 8388607	4000	RWE
57	RAMP_DMAX [internal] Deceleration in top part of eight-point ramp.	1 ... 8388607	1000	RWE
58	RAMP_D1 [internal] Second deceleration in eight-point ramp.	1 ... 8388607	8000	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
59	RAMP_D2 [internal] First deceleration in eight-point ramp.	1 ... 8388607	8000	RWE
60	RAMP_VMAX [internal] Maximum velocity of eight-point ramp.	0 ... 134217727	134217727	RWE
61	RAMP_V1 [internal] Velocity threshold to switch from A1/D1 to A2/D2.	0 ... 134217727	0	RWE
62	RAMP_V2 [internal] Velocity threshold to switch from A2/D2 to AMAX/DMAX.	0 ... 134217727	0	RWE
63	RAMP_VSTART [internal] Start velocity of ramp.	0 ... 8388607	0	RWE
64	RAMP_VSTOP [internal] Stop velocity of ramp. Needs to be greater than 0.	1 ... 8388607	1	RWE
65	RAMP_TVMAX [internal] Minimum time at VMAX to start deceleration.	0 ... 65535	0	RWE
66	RAMP_TZEROWAIT [internal] Wait time at end of ramp to signal stop.	0 ... 65535	0	RWE
67	ACCELERATION_FEEDFORWARD_ENABLE Enable the acceleration feedforward feature. False: DISABLED True: ENABLED	0, 1	0	RWE
68	VELOCITY_FEEDFORWARD_ENABLE Enable the velocity feedforward feature. False: DISABLED True: ENABLED	0, 1	0	RWE
69	RAMP_VELOCITY Target velocity calculated by ramp controller.	-134217727 ... 134217727	0	R
70	RAMP_POSITION Target position calculated by ramp controller.	-2147483648 ... 2147483647	0	R
74	HALL_PHI_E Phi_e calculated from hall feedback.	-32768 ... 32767	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
75	HALL_SECTOR_OFFSET Hall sensor 60-degree/sector offset composed of 120 offset (order) and 180 degree offset (polarity). 0: DEG_0 1: DEG_60 2: DEG_120 3: DEG_180 4: DEG_240 5: DEG_300	0, 1, 2, 3, 4, 5	0	RWE
76	HALL_FILTER_LENGTH Filter length of the hall sensor input signal filters.	0 ... 255	0	RWE
77	HALL_POSITION_0_OFFSET Hall offset compensation for 0 degree hall position.	-32768 ... 32767	0	RWE
78	HALL_POSITION_60_OFFSET Hall offset compensation for 60 degree hall position.	-32768 ... 32767	10922	RWE
79	HALL_POSITION_120_OFFSET Hall offset compensation for 120 degree hall position.	-32768 ... 32767	21845	RWE
80	HALL_POSITION_180_OFFSET Hall offset compensation for 180 degree hall position.	-32768 ... 32767	-32768	RWE
81	HALL_POSITION_240_OFFSET Hall offset compensation for 240 degree hall position.	-32768 ... 32767	-21846	RWE
82	HALL_POSITION_300_OFFSET Hall offset compensation for 300 degree hall position.	-32768 ... 32767	-10923	RWE
83	HALL_INVERT_DIRECTION Invert the hall angle direction. False: NOT_INVERTED True: INVERTED	0, 1	0	RWE
84	HALL_EXTRAPOLATION_ENABLE Enable the hall extrapolation to generate a higher resolution position signal. The extrapolation is only active at speeds higher than 60 rpm. False: DISABLED True: ENABLED	0, 1	0	RWE
85	HALL_PHI_E_OFFSET Use this parameter to compensate hall sensor mounting tolerances.	-32768 ... 32767	0	RWE
89	ABN_1_PHI_E Phi_e calculated from abn feedback.	-32768 ... 32767	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
90	ABN_1_STEPS ABN 1 encoder steps per rotation (CPR).	0 ... 16777215	65536	RWE
91	ABN_1_DIRECTION ABN 1 encoder rotation direction. False: NOT_INVERTED True: INVERTED	0, 1	0	RWE
92	ABN_1_INIT_METHOD Select an ABN encoder initialization method that fits best to your motor's sensors. 0: FORCED_PHI_E_ZERO_WITH_ACTIVE_SWING Forces the rotor into phi_e zero using the open loop current but actively swings the rotor. 1: FORCED_PHI_E_90_ZERO Forces the rotor into phi_e 90 degree position and then into zero position using the open loop current. 2: USE_HALL Turns the motor slightly in hall commutation mode until a hall signal change gives a new absolut position, which then the ABN phi_e is aligned to. 3: USE_N_CHANNEL_OFFSET Turns the motor slightly in open loop commutation mode until a N-channel is reached and gives an absolut position, which then the ABN phi_e is aligned to.	0, 1, 2, 3	0	RWE
93	ABN_1_INIT_STATE Actual state of ABN encoder initialization. 0: IDLE 1: BUSY 2: WAIT 3: DONE	0, 1, 2, 3	0	R
94	ABN_1_INIT_DELAY [ms] When one of the "Forced phi_e" initialization methods is used, this value defines the wait time until the phi_e ABN angle is set to zero. This parameter should be set in a way, that the motor has stopped mechanical oscillations after the specified time.	1000 ... 10000	1000	RWE
95	ABN_1_INIT_VELOCITY Init velocity for ABN encoder initialization with N-channel offset.	-200000 ... 200000	5	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
96	ABN_1_N_CHANNEL_PHI_E_OFFSET Offset between phi_e zero and the ABN encoders index pulse position. This value is updated asynchronously on any ABN initialization other than the "Use-N channel offset" method. The value can then be used for the "Use N-channel offset" based initialization.	-32768 ... 32767	0	RWE
97	ABN_1_N_CHANNEL_INVERTED ABN 1 encoder N-channel is inverted. False: ACTIVE_HIGH True: ACTIVE_LOW	0, 1	0	RWE
98	ABN_1_N_CHANNEL_FILTERING ABN 1 encoder N-channel filtering. Useful for imprecise encoders with the index pulses lasting multiple A/B steps. 0: FILTERING_OFF 1: N_EVENT_ON_A_HIGH_B_HIGH 2: N_EVENT_ON_A_HIGH_B_LOW 3: N_EVENT_ON_A_LOW_B_HIGH 4: N_EVENT_ON_A_LOW_B_LOW	0, 1, 2, 3, 4	0	RWE
99	ABN_1_CLEAR_ON_NEXT_NULL Clear the actual position on the next ABN 1 encoder N-channel event. False: DISABLED True: ENABLED	0, 1	0	RW
100	ABN_1_VALUE Raw ABN encoder internal counter value.	0 ... 16777215	0	R
104	TARGET_TORQUE [mA] Target torque value. Write to activate torque regulation.	-32768 ... 32767	0	RW
105	ACTUAL_TORQUE [mA] Actual motor torque value.	-32767 ... 32768	0	R
106	TARGET_FLUX [mA] Target flux value.	-10000 ... 10000	0	RW
107	ACTUAL_FLUX [mA] Actual motor flux value.	-2147483648 ... 2147483647	0	R
108	TORQUE_OFFSET [mA] (peak) Offset applied to torque value.	-4700 ... 4700	0	RW

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
109	TORQUE_P P parameter for torque PI regulator. Also controls flux P parameter unless separate torque/flux loops are enabled.	0 ... 32767	50	RWE
110	TORQUE_I I parameter for torque PI regulator. Also controls flux I parameter unless separate torque/flux loops are enabled.	0 ... 32767	100	RWE
111	FLUX_P P parameter for flux PI regulator. Only available when separated torque/flux loops are enabled.	0 ... 32767	50	RWE
112	FLUX_I I parameter for flux PI regulator. Only available when separated torque/flux loops are enabled.	0 ... 32767	100	RWE
113	SEPARATE_TORQUE_FLUX_PI_PARAMTERS Enable to configure separate PI values for the torque and flux current control loops. False: TORQUE_FLUX_PI_COMBINED True: TORQUE_FLUX_PI_SEPARATED	0, 1	0	RWE
114	CURRENT_NORM_P P parameter normalization format for current PI regulator. 0: SHIFT_8_BIT 1: SHIFT_16_BIT	0, 1	0	RWE
115	CURRENT_NORM_I I parameter normalization format for current PI regulator. 0: SHIFT_8_BIT 1: SHIFT_16_BIT	0, 1	1	RWE
116	TORQUE_PI_ERROR Torque PI regulator error.	-2147483648 ... 2147483647	0	R
117	FLUX_PI_ERROR Flux PI regulator error.	-2147483648 ... 2147483647	0	R
118	TORQUE_PI_INTEGRATOR Integrated error of torque PI regulator.	-2147483648 ... 2147483647	0	R
119	FLUX_PI_INTEGRATOR Integrated error of flux PI regulator.	-2147483648 ... 2147483647	0	R
120	FLUX_OFFSET [mA] (peak) Offset applied to flux value.	-4700 ... 4700	0	RW

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
123	VELOCITY_SENSOR_SELECTION Feedback source for the velocity PI regulator. 0: SAME_AS_COMMUTATION 1: DIGITAL_HALL 2: ABN1_ENCODER 3: ABN2_ENCODER 4: SPI_ENCODER	0, 1, 2, 3, 4	0	RWE
124	TARGET_VELOCITY Target velocity value. Write to activate velocity regulation.	-134217728 ... 134217727	0	RW
125	ACTUAL_VELOCITY Actual velocity value.	-2147483648 ... 2147483647	0	R
126	VELOCITY_OFFSET [rpm] Offset applied to velocity value.	-200000 ... 200000	0	RW
127	VELOCITY_P P parameter for velocity PI regulator.	0 ... 32767	800	RWE
128	VELOCITY_I I parameter for velocity PI regulator.	0 ... 32767	1	RWE
129	VELOCITY_NORM_P P parameter normalization format for velocity PI regulator. 0: NO_SHIFT 1: SHIFT_8_BIT 2: SHIFT_16_BIT 3: SHIFT_24_BIT	0, 1, 2, 3	2	RWE
130	VELOCITY_NORM_I I parameter normalization format for velocity PI regulator. 0: SHIFT_8_BIT 1: SHIFT_16_BIT 2: SHIFT_24_BIT 3: SHIFT_32_BIT	0, 1, 2, 3	2	RWE
131	VELOCITY_PI_INTEGRATOR Integrated error of velocity PI regulator.	-2147483648 ... 2147483647	0	R
132	VELOCITY_PI_ERROR Velocity PI regulator error.	-2147483648 ... 2147483647	0	R
133	VELOCITY_SCALING_FACTOR Scaling factor to convert internal velocity to real-world units.	1 ... 2047	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
134	STOP_ON_VELOCITY_DEVIATION Maximum of velocity deviation tolerated before stop event is triggered (if activated).	0 … 200000	0	RW
135	VELOCITY_LOOP_DOWNSAMPLING Downsampling factor for velocity controller.	0 … 127	5	RWE
136	VELOCITY_REACHED_THRESHOLD Deviation between target and actual velocity below which the velocity reached flag goes active and latches. If a new target velocity is set the flag is reset.	0 … 2000000000	0	RWE
137	VELOCITY_METER_SWITCH_THRESHOLD Velocity threshold switching from period to frequency velocity meter.	0 … 134217727	2000	RWE
138	VELOCITY_METER_SWITCH_HYSTERESIS Velocity hysteresis for switching back from frequency to period velocity meter.	0 … 65535	500	RWE
139	VELOCITY_METER_MODE Currently used velocity meter mode. 0: PERIOD_METER Measurement of velocity by time measurement between position changes. 1: FREQUENCY_METER Velocity Meter running at PWM frequency. Calculates the velocity using the difference of the angle in one clock cycle. 2: SOFTWARE_METER Measurement of velocity by software.	0, 1, 2	0	R
142	POSITION_SENSOR_SELECTION Feedback source for the position PI regulator. 0: SAME_AS_COMMUTATION 1: DIGITAL_HALL 2: ABN1_ENCODER 3: ABN2_ENCODER 4: SPI_ENCODER	0, 1, 2, 3, 4	0	RWE
143	TARGET_POSITION Target position value. Write to activate position regulation.	-2147483648 … 2147483647	0	RW
144	ACTUAL_POSITION Actual position value.	-2147483648 … 2147483647	0	RW

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
145	POSITION_SCALING_FACTOR Scaling factor to convert internal position to real-world units.	1024 … 65535	1024	RWE
146	POSITION_P P parameter for position PI regulator.	0 … 32767	5	RWE
147	POSITION_I I parameter for position PI regulator.	0 … 32767	0	RWE
148	POSITION_NORM_P P parameter normalization format for position PI regulator. 0: NO_SHIFT 1: SHIFT_8_BIT 2: SHIFT_16_BIT 3: SHIFT_24_BIT	0, 1, 2, 3	1	RWE
149	POSITION_NORM_I I parameter normalization format for position PI regulator. 0: SHIFT_8_BIT 1: SHIFT_16_BIT 2: SHIFT_24_BIT 3: SHIFT_32_BIT	0, 1, 2, 3	1	RWE
150	POSITION_PI_INTEGRATOR Integrated error of position PI regulator.	-2147483648 … 2147483647	0	R
151	POSITION_PI_ERROR Error of position PI regulator.	-2147483648 … 2147483647	0	R
152	STOP_ON_POSITION_DEVIATION Maximum of position deviation tolerated before stop event is triggered (if activated).	0 … 2147483647	0	RWE
153	POSITION_LOOP_DOWNSAMPLING Downsampling factor for position controller.	0 … 127	0	RWE
154	LATCH_POSITION Position switch latched.	-2147483648 … 2147483647	0	R
155	POSITION_LIMIT_LOW Position limit low.	-2147483648 … 2147483647	-2147483648	RWE
156	POSITION_LIMIT_HIGH Position limit high.	-2147483648 … 2147483647	2147483647	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
157	POSITION_REACHED_THRESHOLD Deviation between target and actual position below which the position reached flag goes active and latches. If a new target position is set the flag is reset.	0 … 2000000000	0	RWE
161	REFERENCE_SWITCH_ENABLE Bitwise enable for stopping when reference switch input is triggered. Bit 2: Stop on reference input home. Bit 1: Stop on reference input right. Bit 0: Stop on reference input left. 0: NO_STOP_ON_SWITCH_TRIGGERED 1: STOP_ON_L 2: STOP_ON_R 3: STOP_ON_R_AND_L 4: STOP_ON_H 5: STOP_ON_H_AND_L 6: STOP_ON_H_AND_R 7: STOP_ON_H_R_AND_L	0, 1, 2, 3, 4, 5, 6, 7	0	RWE
162	REFERENCE_SWITCH_POLARITY_AND_SWAP Bitwise configuration of reference switch configuration. Options to swap left and right input and invert switch polarity. Bit 3: Swap left and right switch. Bit 2: Invert polarity of home switch. Bit 1: Invert polarity of right switch. Bit 0: Invert polarity of left switch. 0: NOT_SWAPPED_NOT_INVERTED 1: L_INVERTED 2: R_INVERTED 3: R_AND_L_INVERTED 4: H_INVERTED 5: H_AND_L_INVERTED 6: H_AND_R_INVERTED 7: H_R_AND_L_INVERTED 8: L_R_SWAPPED_L_INVERTED 9: L_R_SWAPPED_R_INVERTED 10: L_R_SWAPPED_R_AND_L_INVERTED 11: L_R_SWAPPED_H_INVERTED 12: L_R_SWAPPED_H_AND_L_INVERTED 13: L_R_SWAPPED 14: L_R_SWAPPED_H_AND_R_INVERTED 15: L_R_SWAPPED_H_R_AND_L_INVERTED	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
163	REFERENCE_SWITCH_LATCH_SETTINGS Bitwise configuration of reference switch latch configuration. Writing position to latch position parameter. Bit 3: Trigger latch on falling home signal. Bit 2: Trigger latch on rising home signal. Bit 1: Trigger latch on falling left and right signal. Bit 0: Trigger latch on rising left and right signal 0: NO_TRIGGER 1: L_R_RISING_EDGE 2: L_R_FALLING_EDGE 3: L_R_BOTH_EDGES 4: H_RISING_EDGE 5: H_L_R_RISING_EDGE 6: H_RISING_L_R_FALLING_EDGE 7: H_RISING_L_R_BOTH_EDGES 8: H_FALLING_EDGE 9: H_FALLING_L_R_RISING_EDGE 10: H_L_R_FALLING_EDGE 11: H_FALLING_L_R_BOTH_EDGES 12: H_BOTH_EDGES 13: H_BOTH_L_R_RISING_EDGE 14: H_BOTH_L_R_FALLING_EDGE 15: H_L_R_BOTH_EDGES	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE
164	EVENT_STOP_SETTINGS Bitwise configuration of stop configuration. Bit 2: Stop if the velocity loop deviation is larger then the parameter "Stop on velocity deviation". Bit 1: Stop if the position loop deviation is larger then the parameter "Stop on position deviation". Bit 0: If enabled the system ramps down to zero if a stop condition rises. Doing a soft and not a hard stop. 0: DO_HARD_STOP 1: DO_SOFT_STOP 2: STOP_ON_POS_DEVIATION 3: STOP_ON_POS_DEVIATION_SOFT_STOP 4: STOP_ON_VEL_DEVIATION 5: STOP_ON_VEL_DEVIATION_SOFT_STOP 6: STOP_ON_POS_VEL_DEVIATION 7: STOP_ON_POS_VEL_DEVIATION_SOFT_STOP	0, 1, 2, 3, 4, 5, 6, 7	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
165	REFERENCE_SWITCH_SEARCH_MODE Determine the reference switch search sequence. 1: LEFT_SWITCH Search for left limit switch. 2: RIGHT_SWITCH_LEFT_SWITCH Search for right limit switch then search for left limit switch. 3: RIGHT_SWITCH_LEFT_SWITCH_BOTH_SIDES Search for right limit switch then approach left limit switch from both sides. 4: LEFT_SWITCH_BOTH_SIDES Approach left limit switch from both sides. 5: HOME_SWITCH_NEG_DIR_LEFT_END_SWITCH Search for home switch in negative direction, turn around if left end switch detected. 6: HOME_SWITCH_POS_DIR_RIGHT_END_SWITCH Search for home switch in positive direction, turn around if right end switch detected. 7: HOME_SWITCH_NEG_DIR_IGNORE_END_SWITCH Search for home switch in negative direction, ignore end switch. 8: HOME_SWITCH_POS_DIR_IGNORE_END_SWITCH Search for home switch in positive direction, ignore end switch.	1, 2, 3, 4, 5, 6, 7, 8	0	RWE
166	REFERENCE_SWITCH_SEARCH_SPEED Speed used for the reference switch search sequence.	-134217728 ... 134217727	0	RWE
167	REFERENCE_SWITCH_SPEED Lower speed used e.g. for positioning the motor at a reference switch position.	-134217728 ... 134217727	0	RWE
168	RIGHT_LIMIT_SWITCH_POSITION Right limit switch position.	-2147483648 ... 2147483647	0	R
169	HOME_SWITCH_POSITION Home switch position.	-2147483648 ... 2147483647	0	R
170	LAST_REFERENCE_POSITION Last reference position.	-2147483648 ... 2147483647	0	R
174	ABN_2_STEPS ABN 2 encoder steps per rotation (CPR).	0 ... 16777215	1024	RWE
175	ABN_2_DIRECTION ABN 2 encoder rotation direction. False: NORMAL True: INVERTED	0, 1	0	RWE
176	ABN_2_GEAR_RATIO ABN 2 encoder gear ratio.	1 ... 255	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
177	ABN_2_ENABLE Enable the ABN 2 encoder. Disabling resets the counted steps. False: DISABLED True: ENABLED	0, 1	0	RWE
178	ABN_2_VALUE Raw ABN2 encoder internal counter value.	0 … 4294967295	0	R
181	SPI_ENCODE_CS_SETTLE_DELAY_TIME [ns] Add a delay from CS going low to first SCLK edge.	0 … 6375	0	RWE
182	SPI_ENCODER_CS_IDLE_DELAY_TIME [us] Extend CS idle time between SPI message frames.	0 … 102	0	RWE
183	SPI_ENCODER_MAIN_TRANSFER_CMD_SIZE Size of the first SPI transfer frame.	1 … 16	1	RWE
184	SPI_ENCODER_SECONDARY_TRANSFER_CMD_SIZE Size of the optional secondary SPI transfer frame. If set to zero, no secondary SPI transfer.	0 … 15	0	RWE
185	SPI_ENCODER_TRANSFER_DATA_3_0 Used to set the transmit data and read out the received data.	0 … 4294967295	0	RWE
186	SPI_ENCODER_TRANSFER_DATA_7_4 Used to set the transmit data and read out the received data.	0 … 4294967295	0	RWE
187	SPI_ENCODER_TRANSFER_DATA_11_8 Used to set the transmit data and read out the received data.	0 … 4294967295	0	RWE
188	SPI_ENCODER_TRANSFER_DATA_15_12 Used to set the transmit data and read out the received data.	0 … 4294967295	0	RWE
189	SPI_ENCODER_TRANSFER SPI interface setting, polarity and phase. 0: OFF 1: TRIGGER_SINGLE_TRANSFER 2: CONTINUOUS_POSITION_COUNTER_READ	0, 1, 2	0	RWE
190	SPI_ENCODER_POSITION_COUNTER_MASK Mask to be used to collect the position counter value from the continuous received data.	0 … 4294967295	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
191	SPI_ENCODER_POSITION_COUNTER_SHIFT Right bit shift for the position counter value before mask is applied.	0 … 127	0	RWE
192	SPI_ENCODER_POSITION_COUNTER_VALUE Actual SPI encoder position value.	0 … 4294967295	0	R
193	SPI_ENCODER_COMMUTATION_ANGLE Actual absolute encoder angle value.	-32768 … 32767	0	R
194	SPI_ENCODER_INITIALIZATION_METHOD Select the used absolute encoder initialization mode 0: FORCED_PHI_E_ZERO_WITH_ACTIVE_SWING Forces the rotor into PHI_E zero using the open loop current but actively swings the rotor. 1: FORCED_PHI_E_90_ZERO Forces the rotor into PHI_E 90 degree position and then into zero position using the open loop current. 2: USE_OFFSET	0, 1, 2	0	RWE
195	SPI_ENCODER_DIRECTION SPI encoder direction. False: NOT_INVERTED True: INVERTED	0, 1	0	RWE
196	SPI_ENCODER_OFFSET This value represents the internal commutation offset. (0...max. encoder steps per rotation).	0 … 4294967295	0	RWE
197	SPI_LUT_CORRECTION_ENABLE Enable the lookup table based encoder correction. False: DISABLED True: ENABLED	0, 1	0	RWE
198	SPI_LUT_ADDRESS_SELECT Address to read or write the lookup table.	0 … 255	0	RW
199	SPI_LUT_DATA Data to read or write to a lookup table address.	-128 … 127	0	RW
201	SPI_LUT_COMMON_SHIFT_FACTOR All LUT table entries are multiplied with 2^SHIFT_FACTOR to compensate for larger erros if needed.	0 … 4	0	RW

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
205	STEP_DIR_STEP_DIVIDER_SHIFT Configure step/dir to use between 1 and 1024 microsteps per full step. 0: STEP_MODE_FULL 1: STEP_MODE_HALF 2: STEP_MODE_QUARTER 3: STEP_MODE_1_8TH 4: STEP_MODE_1_16TH 5: STEP_MODE_1_32ND 6: STEP_MODE_1_64TH 7: STEP_MODE_1_128TH 8: STEP_MODE_1_256TH 9: STEP_MODE_1_512TH 10: STEP_MODE_1_1024TH	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10	0	RWE
206	STEP_DIR_ENABLE Enable the Step/Dir input functionality. False: DISABLED True: ENABLED	0, 1	0	RW
207	STEP_DIR_EXTRAPOLATION_ENABLE Enable the Step/Dir extrapolation feature. False: DISABLED True: ENABLED	0, 1	0	RW
208	STEP_DIR_STEP_SIGNAL_TIMEOUT_LIMIT [ms] Step signal timeout limit.	1 ... 2000	1000	RW
209	STEP_DIR_MAXIMUM_EXTRAPOLATION_VELOCITY [eRPM] Maximum velocity up to which extrapolation is used.	0 ... 2147483647	2147483647	RW
212	BRAKE_CHOPPER_ENABLE Enable the brake chopper functionality. False: DISABLED True: ENABLED	0, 1	0	RWE
213	BRAKE_CHOPPER_VOLTAGE_LIMIT [0.1V] If the brake chopper is enabled and supply voltage exceeds this value, the brake chopper output is activated.	50 ... 1000	260	RWE
214	BRAKE_CHOPPER_HYSTERESIS [0.1V] An activated brake chopper is deactivated if the actual supply voltage is lower than BRAKE_CHOPPER_VOLTAGE_LIMIT - BRAKE_CHOPPER_HYSTERESIS.	0 ... 50	5	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
216	RELEASE_BRAKE Release the external brake by applying a PWM signal. False: BRAKE_PWM_DEACTIVATED True: BRAKE_PWM_ACTIVATED	0, 1	0	RWE
217	BRAKE_RELEASING_DUTY_CYCLE [%] Set the duty cycle of the first PWM phase for releasing the brake.	0 ... 99	75	RWE
218	BRAKE_HOLDING_DUTY_CYCLE [%] Set the duty cycle of the second PWM phase to hold the brake.	0 ... 99	11	RWE
219	BRAKE_RELEASING_DURATION [ms] Set the duration the brake PWM uses the first duty cycle.	0 ... 65535	80	RWE
221	INVERT_BRAKE_OUTPUT Invert the brake output. False: NORMAL True: INVERTED	0, 1	0	RWE
224	THERMAL_WINDING_TIME_CONSTANT_1 [ms] Thermal winding time constant for the used motor. Used for IIT monitoring. Setting a new value restarts the IIT monitoring.	1000 ... 60000	3000	RWE
225	IIT_LIMIT_1 [A^2 x ms] An actual IIT sum that exceeds this limit leads to trigger the IIT_1_EXCEEDED.	0 ... 4294967295	4294967295	RWE
226	IIT_SUM_1 [A^2 x ms] Actual sum of the IIT monitor 1.	0 ... 4294967295	0	R
227	THERMAL_WINDING_TIME_CONSTANT_2 [ms] Thermal winding time constant for the used motor. Used for IIT monitoring. Setting a new value restarts the IIT monitoring.	1000 ... 60000	6000	RWE
228	IIT_LIMIT_2 [A^2 x ms] An actual IIT sum that exceeds this limit leads to trigger the IIT_2_EXCEEDED.	0 ... 4294967295	4294967295	RWE
229	IIT_SUM_2 [A^2 x ms] Actual sum of the IIT monitor 2.	0 ... 4294967295	0	R
230	RESET_IIT_SUMS Reset both IIT sums.	0 ... 0	0	W

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
231	ACTUAL_TOTAL_MOTOR_CURRENT [mA] Total current through the motor windings.	0 … 65535	0	R
233	PWM_L_OUTPUT_POLARITY PWM_L output polarity. False: ACTIVE_HIGH True: ACTIVE_LOW	0, 1	0	RWE
234	PWM_H_OUTPUT_POLARITY PWM_H output polarity. False: ACTIVE_HIGH True: ACTIVE_LOW	0, 1	0	RWE
235	BREAK_BEFORE_MAKE_TIME_LOW_UVW [8.33ns] Break before make time for the low side gates of the UVW phases. Applied before switching from high to low.	0 … 255	0	RWE
236	BREAK_BEFORE_MAKE_TIME_HIGH_UVW [8.33ns] Break before make time for the high side gates of the UVW phases. Applied before switching from low to high.	0 … 255	0	RWE
237	BREAK_BEFORE_MAKE_TIME_LOW_Y2 [8.33ns] Break before make time for the low side gate of the Y2 phase. Applied before switching from high to low.	0 … 255	0	RWE
238	BREAK_BEFORE_MAKE_TIME_HIGH_Y2 [8.33ns] Break before make time for the high side gate of the Y2 phase. Applied before switching from low to high.	0 … 255	0	RWE
239	USE_ADAPTIVE_DRIVE_TIME_UVW If enabled, the discharge cycle of the low- and high-side gates for the UVW phases is shortened by monitoring the gate voltages. If enabled, the value on T_DRIVE_SINK acts as an upper bound instead of a fixed time. False: DISABLED True: ENABLED	0, 1	1	RWE
240	USE_ADAPTIVE_DRIVE_TIME_Y2 If enabled, the discharge cycle of the low- and high-side gates for the Y2 phase is shortened by monitoring the gate voltages. If enabled, the value on T_DRIVE_SINK acts as an upper bound instead of a fixed time. False: DISABLED True: ENABLED	0, 1	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
241	DRIVE_TIME_SINK_UVW Discharge time for the low and high side gates of the UVW phases. During this time, the full sink current is be applied. The applied time is defined as (1s / 120MHz) x (2 x DRIVE_TIME_SINK_UVW + 3).	0 … 255	255	RWE
242	DRIVE_TIME_SOURCE_UVW Charge time for the low and high side gates of the UVW phases. During this time, the full source current is applied. The applied time is defined as (1s / 120MHz) x (2 x DRIVE_TIME_SOURCE_UVW + 3).	0 … 255	255	RWE
243	DRIVE_TIME_SINK_Y2 Discharge time for the low and high side gates of the Y2 phase. During this time, the full sink current is applied. The applied time is defined as (1s / 120MHz) x (2 x DRIVE_TIME_SINK_Y2 + 3).	0 … 255	255	RWE
244	DRIVE_TIME_SOURCE_Y2 Charge time for the low and high side gates of the Y2 phase. During this time, the full source current is applied. The applied time is defined as (1s / 120MHz) x (2 x DRIVE_TIME_SOURCE_Y2 + 3).	0 … 255	255	RWE
245	UVW_SINK_CURRENT Limit the maximum sink current for the low and high side gates of the UVW phases. 0: CUR_50_MILLIAMP 1: CUR_100_MILLIAMP 2: CUR_160_MILLIAMP 3: CUR_210_MILLIAMP 4: CUR_270_MILLIAMP 5: CUR_320_MILLIAMP 6: CUR_380_MILLIAMP 7: CUR_430_MILLIAMP 8: CUR_580_MILLIAMP 9: CUR_720_MILLIAMP 10: CUR_860_MILLIAMP 11: CUR_1000_MILLIAMP 12: CUR_1250_MILLIAMP 13: CUR_1510_MILLIAMP 14: CUR_1770_MILLIAMP 15: CUR_2000_MILLIAMP	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	4	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
246	UVW_SOURCE_CURRENT Limit the maximum source current for the low and high side gates of the UVW phases. 0: CUR_25_MILLIAMP 1: CUR_50_MILLIAMP 2: CUR_80_MILLIAMP 3: CUR_105_MILLIAMP 4: CUR_135_MILLIAMP 5: CUR_160_MILLIAMP 6: CUR_190_MILLIAMP 7: CUR_215_MILLIAMP 8: CUR_290_MILLIAMP 9: CUR_360_MILLIAMP 10: CUR_430_MILLIAMP 11: CUR_500_MILLIAMP 12: CUR_625_MILLIAMP 13: CUR_755_MILLIAMP 14: CUR_855_MILLIAMP 15: CUR_1000_MILLIAMP	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	4	RWE
247	Y2_SINK_CURRENT Limit the maximum sink current for the low and high side gates of the Y2 phase. 0: CUR_50_MILLIAMP 1: CUR_100_MILLIAMP 2: CUR_160_MILLIAMP 3: CUR_210_MILLIAMP 4: CUR_270_MILLIAMP 5: CUR_320_MILLIAMP 6: CUR_380_MILLIAMP 7: CUR_430_MILLIAMP 8: CUR_580_MILLIAMP 9: CUR_720_MILLIAMP 10: CUR_860_MILLIAMP 11: CUR_1000_MILLIAMP 12: CUR_1250_MILLIAMP 13: CUR_1510_MILLIAMP 14: CUR_1770_MILLIAMP 15: CUR_2000_MILLIAMP	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	4	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
248	Y2_SOURCE_CURRENT Limit the maximum source current for the low and high side gates of the Y2 phase. 0: CUR_25_MILLIAMP 1: CUR_50_MILLIAMP 2: CUR_80_MILLIAMP 3: CUR_105_MILLIAMP 4: CUR_135_MILLIAMP 5: CUR_160_MILLIAMP 6: CUR_190_MILLIAMP 7: CUR_215_MILLIAMP 8: CUR_290_MILLIAMP 9: CUR_360_MILLIAMP 10: CUR_430_MILLIAMP 11: CUR_500_MILLIAMP 12: CUR_625_MILLIAMP 13: CUR_755_MILLIAMP 14: CUR_855_MILLIAMP 15: CUR_1000_MILLIAMP	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	4	RWE
249	BOOTSTRAP_CURRENT_LIMIT Bootstrap current limit. 0: CUR_45_MILLIAMP 1: CUR_91_MILLIAMP 2: CUR_141_MILLIAMP 3: CUR_191_MILLIAMP 4: CUR_267_MILLIAMP 5: CUR_292_MILLIAMP 6: CUR_341_MILLIAMP 7: CUR_391_MILLIAMP	0, 1, 2, 3, 4, 5, 6, 7	7	RWE
250	UNDERVOLTAGE_PROTECTION_SUPPLY_LEVEL Undervoltage protection level for VS (Supply voltage). 0 disables the comparator. 1-16 are mapped to 0-15 HW values, with the comparator enabled.	0 … 16	0	RWE
251	UNDERVOLTAGE_PROTECTION_VDRV_ENABLE Enable the undervoltage protection for VDRV (Driver voltage). False: DISABLED True: ENABLED	0, 1	1	RWE
252	UNDERVOLTAGE_PROTECTION_BST_UVW_ENABLE Enable the undervoltage protection on the bootstrap capacitor of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
253	UNDERVOLTAGE_PROTECTION_BST_Y2_ENABLE Enable the undervoltage protection on the bootstrap capacitor of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
254	OVERCURRENT_PROTECTION_UVW_LOW_SIDE_ENABLE Enable the overcurrent protection on the low side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
255	OVERCURRENT_PROTECTION_UVW_HIGH_SIDE_ENABLE Enable the overcurrent protection on the high side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
256	OVERCURRENT_PROTECTION_Y2_LOW_SIDE_ENABLE Enable the overcurrent protection on the low side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
257	OVERCURRENT_PROTECTION_Y2_HIGH_SIDE_ENABLE Enable the overcurrent protection on the high side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
258	OVERCURRENT_PROTECTION_UVW_LOW_SIDE_THRESHOLD Overcurrent protection threshold for the low side of the UVW phases (uses second list if OVERCURRENT_PROTECTION_UVW_LOW_SIDE_USE_VDS=true). 0: V_80_OR_63_MILLIVOLT 1: V_165_OR_125_MILLIVOLT 2: V_250_OR_187_MILLIVOLT 3: V_330_OR_248_MILLIVOLT 4: V_415_OR_312_MILLIVOLT 5: V_500_OR_374_MILLIVOLT 6: V_582_OR_434_MILLIVOLT 7: V_660_OR_504_MILLIVOLT 8: V_125_OR_705_MILLIVOLT 9: V_250_OR_940_MILLIVOLT 10: V_375_OR_1180_MILLIVOLT 11: V_500_OR_1410_MILLIVOLT 12: V_625_OR_1650_MILLIVOLT 13: V_750_OR_1880_MILLIVOLT 14: V_875_OR_2110_MILLIVOLT 15: V_1000_OR_2350_MILLIVOLT	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE
259	OVERCURRENT_PROTECTION_UVW_HIGH_SIDE_THRESHOLD Overcurrent protection threshold for the high side of the UVW phases. 0: V_63_MILLIVOLT 1: V_125_MILLIVOLT 2: V_187_MILLIVOLT 3: V_248_MILLIVOLT 4: V_312_MILLIVOLT 5: V_374_MILLIVOLT 6: V_434_MILLIVOLT 7: V_504_MILLIVOLT 8: V_705_MILLIVOLT 9: V_940_MILLIVOLT 10: V_1180_MILLIVOLT 11: V_1410_MILLIVOLT 12: V_1650_MILLIVOLT 13: V_1880_MILLIVOLT 14: V_2110_MILLIVOLT 15: V_2350_MILLIVOLT	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
260	OVERCURRENT_PROTECTION_Y2_LOW_SIDE_THRESHOLD Overcurrent protection threshold for the low side of the Y2 phase (uses second list if OVERCURRENT_PROTECTION_Y2_LOW_SIDE_USE_VDS= true). 0: V_80_OR_63_MILLIVOLT 1: V_165_OR_125_MILLIVOLT 2: V_250_OR_187_MILLIVOLT 3: V_330_OR_248_MILLIVOLT 4: V_415_OR_312_MILLIVOLT 5: V_500_OR_374_MILLIVOLT 6: V_582_OR_434_MILLIVOLT 7: V_660_OR_504_MILLIVOLT 8: V_125_OR_705_MILLIVOLT 9: V_250_OR_940_MILLIVOLT 10: V_375_OR_1180_MILLIVOLT 11: V_500_OR_1410_MILLIVOLT 12: V_625_OR_1650_MILLIVOLT 13: V_750_OR_1880_MILLIVOLT 14: V_875_OR_2110_MILLIVOLT 15: V_1000_OR_2350_MILLIVOLT	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE
261	OVERCURRENT_PROTECTION_Y2_HIGH_SIDE_THRESHOLD Overcurrent protection threshold for the high side of the Y2 phase. 0: V_63_MILLIVOLT 1: V_125_MILLIVOLT 2: V_187_MILLIVOLT 3: V_248_MILLIVOLT 4: V_312_MILLIVOLT 5: V_374_MILLIVOLT 6: V_434_MILLIVOLT 7: V_504_MILLIVOLT 8: V_705_MILLIVOLT 9: V_940_MILLIVOLT 10: V_1180_MILLIVOLT 11: V_1410_MILLIVOLT 12: V_1650_MILLIVOLT 13: V_1880_MILLIVOLT 14: V_2110_MILLIVOLT 15: V_2350_MILLIVOLT	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
262	OVERCURRENT_PROTECTION_UVW_LOW_SIDE_BLANKING Overcurrent protection blanking time for the low side of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	2	RWE
263	OVERCURRENT_PROTECTION_UVW_HIGH_SIDE_BLANKING Overcurrent protection blanking time for the high side of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	2	RWE
264	OVERCURRENT_PROTECTION_Y2_LOW_SIDE_BLANKING Overcurrent protection blanking time for the low side of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	2	RWE
265	OVERCURRENT_PROTECTION_Y2_HIGH_SIDE_BLANKING Overcurrent protection blanking time for the high side of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	2	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
266	OVERCURRENT_PROTECTION_UVW_LOW_SIDE_DEGLITCH Overcurrent protection deglitch time for the low side of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	6	RWE
267	OVERCURRENT_PROTECTION_UVW_HIGH_SIDE_DEGLITCH Overcurrent protection deglitch time for the high side of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	6	RWE
268	OVERCURRENT_PROTECTION_Y2_LOW_SIDE_DEGLITCH Overcurrent protection deglitch time for the low side of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	6	RWE
269	OVERCURRENT_PROTECTION_Y2_HIGH_SIDE_DEGLITCH Overcurrent protection deglitch time for the high side of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	6	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
270	OVERCURRENT_PROTECTION_UVW_LOW_SIDE_USE_VDS Use the VDS measurement for the overcurrent protection on the low side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
271	OVERCURRENT_PROTECTION_Y2_LOW_SIDE_USE_VDS Use the VDS measurement for the overcurrent protection on the low side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
272	VGS_SHORT_ON_PROTECTION_UVW_LOW_SIDE_ENABLE Enable the gate-source short protection for the ON transition of the low side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
273	VGS_SHORT_OFF_PROTECTION_UVW_LOW_SIDE_ENABLE Enable the gate-source short protection for the OFF transition of the low side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
274	VGS_SHORT_ON_PROTECTION_UVW_HIGH_SIDE_ENABLE Enable the gate-source short protection for the ON transition of the high side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
275	VGS_SHORT_OFF_PROTECTION_UVW_HIGH_SIDE_ENABLE Enable the gate-source short protection for the OFF transition of the high side of the UVW phases. False: DISABLED True: ENABLED	0, 1	1	RWE
276	VGS_SHORT_ON_PROTECTION_Y2_LOW_SIDE_ENABLE Enable the gate-source short protection for the ON transition of the low side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
277	VGS_SHORT_OFF_PROTECTION_Y2_LOW_SIDE_ENABLE Enable the gate-source short protection for the OFF transition of the low side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
278	VGS_SHORT_ON_PROTECTION_Y2_HIGH_SIDE_ENABLE Enable the gate-source short protection for the ON transition of the high side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
279	VGS_SHORT_OFF_PROTECTION_Y2_HIGH_SIDE_ENABLE Enable the gate-source short protection for the OFF transition of the high side of the Y2 phase. False: DISABLED True: ENABLED	0, 1	1	RWE
280	VGS_SHORT_PROTECTION_UVW_BLANKING Gate-source short protection blanking time for the low and high sides of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC	0, 1, 2, 3	1	RWE
281	VGS_SHORT_PROTECTION_Y2_BLANKING Gate-source short protection blanking time for the low and high sides of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC	0, 1, 2, 3	1	RWE
282	VGS_SHORT_PROTECTION_UVW_DEGLITCH Gate-source short protection deglitch time for the low and high sides of the UVW phases. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	1	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
283	VGS_SHORT_PROTECTION_Y2_DEGLITCH Gate-source short protection deglitch time for low and high sides of the Y2 phase. 0: OFF 1: T_0_25_MICROSEC 2: T_0_5_MICROSEC 3: T_1_MICROSEC 4: T_2_MICROSEC 5: T_4_MICROSEC 6: T_6_MICROSEC 7: T_8_MICROSEC	0, 1, 2, 3, 4, 5, 6, 7	1	RWE
286	GDRV_RETRY_BEHAVIOUR This value defines the state the system goes to after a fault condition on a motor phase occurs. 0: OPEN_CIRCUIT The system switches off and discharges the gates. The motor can spin freely. 1: ELECTRICAL_BRAKING The system switches off and, if possible and depending on fault, enables the LS or HS gates, braking the motor electrically.	0, 1	0	RWE
287	DRIVE_FAULT_BEHAVIOUR This value defines the state the system goes to after a fault condition on a motor phase occurs and all retries failed. 0: OPEN_CIRCUIT The system switches off and discharges the LS and HS gates, letting the motor spin freely. 1: ELECTRICAL_BRAKING The system switches off and, if possible and depending on fault, enables the LS or HS gates, braking the motor electrically. 2: MECHANICAL_BRAKING_AND_OPEN_CIRCUIT The system switches off, discharges the LS and HS gates, and, if correctly configured, engages the mechanical brake. 3: MECHANICAL_AND_ELECTRICAL_BRAKING The system switches off, if possible and depending on fault, enables the LS or HS gates, and, if correctly configured, engages the mechanical brake.	0, 1, 2, 3	0	RWE
288	FAULT_HANDLER_NUMBER_OF_RETRIES Maximum number of retries that are performed for every fault that is detected.	0 … 255	5	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
289	GENERAL_STATUS_FLAGS Actual status flags. 0x00000001: REGULATION_STOPPED 0x00000002: REGULATION_TORQUE 0x00000004: REGULATION_VELOCITY 0x00000008: REGULATION_POSITION 0x00000010: CONFIG_STORED 0x00000020: CONFIG_LOADED 0x00000040: CONFIG_READ_ONLY 0x00000080: TMCL_SCRIPT_READ_ONLY 0x00000100: BRAKE_CHOPPER_ACTIVE 0x00000200: POSITION_REACHED 0x00000400: VELOCITY_REACHED 0x00000800: ADC_OFFSET_CALIBRATED 0x00001000: RAMPER_LATCHED 0x00002000: RAMPER_EVENT_STOP_SWITCH 0x00004000: RAMPER_EVENT_STOP_DEVIATION 0x00008000: RAMPER_VELOCITY_REACHED 0x00010000: RAMPER_POSITION_REACHED 0x00020000: RAMPER_SECOND_MOVE 0x00040000: IIT_1_ACTIVE 0x00080000: IIT_2_ACTIVE 0x00100000: REFSEARCH_FINISHED 0x00200000: Y2_USED_FOR_BRAKING 0x00800000: STEPDIR_INPUT_AVAILABLE 0x01000000: RIGHT_REF_SWITCH_AVAILABLE 0x02000000: HOME_REF_SWITCH_AVAILABLE 0x04000000: LEFT_REF_SWITCH_AVAILABLE 0x08000000: ABN2_FEEDBACK_AVAILABLE 0x10000000: HALL_FEEDBACK_AVAILABLE 0x20000000: ABN1_FEEDBACK_AVAILABLE 0x40000000: SPI_FLASH_AVAILABLE 0x80000000: I2C_EEPROM_AVAILABLE	-	-	
290	SUPPLY_VOLTAGE [0.1V] The actual supply voltage.	0 ... 1000	0	R
291	SUPPLY_OVERVOLTAGE_WARNING_THRESHOLD [0.1V] The supply overvoltage warning threshold.	0 ... 1000	480	RWE
292	SUPPLY_UNDERVOLTAGE_WARNING_THRESHOLD [0.1V] The supply undervoltage warning threshold.	0 ... 1000	50	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
293	EXTERNAL_TEMPERATURE The actual temperature at the external temperature sensor.	0 … 65535	0	R
294	EXTERNAL_TEMPERATURE_SHUTDOWN_THRESHOLD The temperature threshold at which the motor driver is shut down.	0 … 65535	65535	RWE
295	EXTERNAL_TEMPERATURE_WARNING_THRESHOLD The temperature threshold above which the warning flag is set.	0 … 65535	65535	RWE
296	CHIP_TEMPERATURE The actual temperature of the chip.	0 … 65535	0	R
297	CHIP_TEMPERATURE_SHUTDOWN_THRESHOLD The temperature threshold at which the motor driver is shut down.	0 … 65535	65535	RWE
298	CHIP_TEMPERATURE_WARNING_THRESHOLD The temperature threshold above which the warning flag is set.	0 … 65535	65535	RWE
299	GENERAL_ERROR_FLAGS Actual error flags. 0x00000001: CONFIG_ERROR 0x00000002: TMCL_SCRIPT_ERROR 0x00000004: HOMESWITCH_NOT_FOUND 0x00000020: HALL_ERROR 0x00000200: WATCHDOG_EVENT 0x00002000: EXT_TEMP_EXCEEDED 0x00004000: CHIP_TEMP_EXCEEDED 0x00010000: ITT_1_EXCEEDED 0x00020000: ITT_2_EXCEEDED 0x00040000: EXT_TEMP_WARNING 0x00080000: SUPPLY_OVERVOLTAGE_WARNING 0x00100000: SUPPLY_UNDERVOLTAGE_WARNING 0x00200000: ADC_IN_OVERVOLTAGE 0x00400000: FAULT_RETRY_HAPPEND 0x00800000: FAULT_RETRIES_FAILED 0x01000000: CHIP_TEMP_WARNING 0x04000000: HEARTBEAT_STOPPED	-	-	

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
300	GDRV_ERROR_FLAGS Gate driver error flags. 0x00000001: U_LOW_SIDE_OVERCURRENT 0x00000002: V_LOW_SIDE_OVERCURRENT 0x00000004: W_LOW_SIDE_OVERCURRENT 0x00000008: Y2_LOW_SIDE_OVERCURRENT 0x00000010: U_LOW_SIDE_DISCHARGE_SHORT 0x00000020: V_LOW_SIDE_DISCHARGE_SHORT 0x00000040: W_LOW_SIDE_DISCHARGE_SHORT 0x00000080: Y2_LOW_SIDE_DISCHARGE_SHORT 0x00000100: U_LOW_SIDE_CHARGE_SHORT 0x00000200: V_LOW_SIDE_CHARGE_SHORT 0x00000400: W_LOW_SIDE_CHARGE_SHORT 0x00000800: Y2_LOW_SIDE_CHARGE_SHORT 0x00001000: U_BOOTSTRAP_UNDERVOLTAGE 0x00002000: V_BOOTSTRAP_UNDERVOLTAGE 0x00004000: W_BOOTSTRAP_UNDERVOLTAGE 0x00008000: Y2_BOOTSTRAP_UNDERVOLTAGE 0x00010000: U_HIGH_SIDE_OVERCURRENT 0x00020000: V_HIGH_SIDE_OVERCURRENT 0x00040000: W_HIGH_SIDE_OVERCURRENT 0x00080000: Y2_HIGH_SIDE_OVERCURRENT 0x00100000: U_HIGH_SIDE_DISCHARGE_SHORT 0x00200000: V_HIGH_SIDE_DISCHARGE_SHORT 0x00400000: W_HIGH_SIDE_DISCHARGE_SHORT 0x00800000: Y2_HIGH_SIDE_DISCHARGE_SHORT 0x01000000: U_HIGH_SIDE_CHARGE_SHORT 0x02000000: V_HIGH_SIDE_CHARGE_SHORT 0x04000000: W_HIGH_SIDE_CHARGE_SHORT 0x08000000: Y2_HIGH_SIDE_CHARGE_SHORT 0x20000000: GDRV_UNDERVOLTAGE 0x40000000: GDRV_LOW_VOLTAGE 0x80000000: GDRV_SUPPLY_UNDERVOLTAGE	-	-	

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
301	ADC_STATUS_FLAGS ADC channel clipped. Flags do latch, write 1 to clear. 0x00000001: I0_CLIPPED 0x00000002: I1_CLIPPED 0x00000004: I2_CLIPPED 0x00000008: I3_CLIPPED 0x00000010: U0_CLIPPED 0x00000020: U1_CLIPPED 0x00000040: U2_CLIPPED 0x00000080: U3_CLIPPED 0x00000100: AIN0_CLIPPED 0x00000200: AIN1_CLIPPED 0x00000400: AIN2_CLIPPED 0x00000800: AIN3_CLIPPED 0x00001000: VM_CLIPPED 0x00002000: TEMP_CLIPPED	-	-	
304	MCC_INPUTS_RAW Raw inputs for ABN, hall, reference switches, driver enabled, hall filtered and ABN2 or Step/Dir.	0 ... 32767	0	R
305	FOC_VOLTAGE_UX Interim result of the FOC for phase U (X in case of motor type is a stepper motor).	-32768 ... 32767	0	R
306	FOC_VOLTAGE_WY Interim result of the FOC for phase W (Y in case of motor type is a stepper motor).	-32768 ... 32767	0	R
307	FOC_VOLTAGE_V Interim result of the FOC for phase V (BLDC motor only).	-32768 ... 32767	0	R
308	FIELDWEAKENING_I I parameter for field weakening controller.	0 ... 32767	0	RWE
310	FIELDWEAKENING_VOLTAGE_THRESHOLD Maximum motor voltage allowed for field weakening.	0 ... 32767	32767	RWE
311	FOC_CURRENT_UX Interim measurement of the FOC for phase UX.	-32768 ... 32767	0	R
312	FOC_CURRENT_V Interim measurement of the FOC for phase V.	-32768 ... 32767	0	R
313	FOC_CURRENT_WY Interim measurement of the FOC for phase WY.	-32768 ... 32767	0	R

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
314	FOC_VOLTAGE_UQ Interim measurement of the FOC for Uq.	-32768 ... 32767	0	R
315	FOC_CURRENT_IQ Interim measurement of the FOC for Iq.	-32768 ... 32767	0	R
318	TARGET_TORQUE_BIQUAD_FILTER_ENABLE Enable the target torque biquad filter. False: DISABLED True: ENABLED	0, 1	0	RWE
319	TARGET_TORQUE_BIQUAD_FILTER_ACOEFF_1 Target torque biquad filter aCoeff_1.	-2147483648 ... 2147483647	0	RWE
320	TARGET_TORQUE_BIQUAD_FILTER_ACOEFF_2 Target torque biquad filter aCoeff_2.	-2147483648 ... 2147483647	0	RWE
321	TARGET_TORQUE_BIQUAD_FILTER_BCOEFF_0 Target torque biquad filter bCoeff_0.	-2147483648 ... 2147483647	1048576	RWE
322	TARGET_TORQUE_BIQUAD_FILTER_BCOEFF_1 Target torque biquad filter bCoeff_1.	-2147483648 ... 2147483647	0	RWE
323	TARGET_TORQUE_BIQUAD_FILTER_BCOEFF_2 Target torque biquad filter bCoeff_2.	-2147483648 ... 2147483647	0	RWE
324	ACTUAL_VELOCITY_BIQUAD_FILTER_ENABLE Enable the actual velocity biquad filter. False: DISABLED True: ENABLED	0, 1	1	RWE
325	ACTUAL_VELOCITY_BIQUAD_FILTER_ACOEFF_1 Actual velocity biquad filter aCoeff_1.	-2147483648 ... 2147483647	1849195	RWE
326	ACTUAL_VELOCITY_BIQUAD_FILTER_ACOEFF_2 Actual velocity biquad filter aCoeff_2.	-2147483648 ... 2147483647	15961938	RWE
327	ACTUAL_VELOCITY_BIQUAD_FILTER_BCOEFF_0 Actual velocity biquad filter bCoeff_0.	-2147483648 ... 2147483647	3665	RWE
328	ACTUAL_VELOCITY_BIQUAD_FILTER_BCOEFF_1 Actual velocity biquad filter bCoeff_1.	-2147483648 ... 2147483647	7329	RWE
329	ACTUAL_VELOCITY_BIQUAD_FILTER_BCOEFF_2 Actual velocity biquad filter bCoeff_2.	-2147483648 ... 2147483647	3665	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
330	TORQUE_FLUX_COMBINED_TARGET_VALUES Raw (unscaled) torque and flux target values combined into one 32 bit value. This is only used for simplified compact measurement during tuning operations.	0 ... 4294967295	0	R
331	TORQUE_FLUX_COMBINED_ACTUAL_VALUES Raw (unscaled) torque and flux actual values combined into one 32 bit value. This is only used for simplified compact measurement during tuning operations.	0 ... 4294967295	0	R
332	VOLTAGE_D_Q_COMBINED_ACTUAL_VALUES Raw (unscaled) voltage actual values combined into one 32 bit value. This is only used for simplified compact measurement during tuning operations.	0 ... 4294967295	0	R
333	INTEGRATED_ACTUAL_TORQUE_VALUE Periodically summed up actual torque value. This is only used for simplified measurement with low measurement frequency during tuning operations.	0 ... 4294967295	0	R
334	INTEGRATED_ACTUAL_VELOCITY_VALUE Periodically summed up actual velocity value. This is only used for simplified measurement with low measurement frequency during tuning operations.	0 ... 4294967295	0	R

ALL FLAGS

Table 58, *Table 59,* and *Table 60* in this section show all flag values in the three flag parameters with descriptions and access. An access of RWC indicates write to clear.

Table 58. All flags in the parameter GENERAL_STATUS_FLAGS

MASK	FLAG	DESCRIPTION	ACCESS
0x00000001	REGULATION_STOPPED	System does not regulate motion	R
0x00000002	REGULATION_TORQUE	System is regulating mode torque.	R
0x00000004	REGULATION_VELOCITY	System is regulating mode velocity.	R
0x00000008	REGULATION_POSITION	System is regulating mode position.	R
0x00000010	CONFIG_STORED	Config was stored successfully.	RWC
0x00000020	CONFIG_LOADED	Config was loaded successfully.	RWC
0x00000040	CONFIG_READ_ONLY	Memory for config is read only.	R
0x00000080	TMCL_SCRIPT_READ_ONLY	Memory for tmcl-script is read only.	R
0x00000100	BREAK_CHOPPER_ACTIVE	Brake chopper is active.	R

0x00000200	POSITION_REACHED	Actual velocity and target velocity are bellow POSITION_REACHED_THRESHOLD.	R
0x00000400	VELOCITY_REACHED	Actual velocity and target velocity are bellow VELOCITY_REACHED_THRESHOLD.	R
0x00000800	ADC_OFFSET_CALIBRATED	The ADC offset was calibrated automatically. Clear to recalibrate.	RWC
0x00001000	RAMPER_LATCHED	The ramper latched a position.	RWC
0x00002000	RAMPER_EVENT_STOP_SWITCH	Ramper had a switch stop event	R
0x00004000	RAMPER_EVENT_STOP_DEVIATION	Ramper had a deviation stop event	RWC
0x00008000	RAMPER_VELOCITY_REACHED	The ramper reached its velocity target	R
0x00010000	RAMPER_POSITION_REACHED	The ramper reached its position target	R
0x00020000	RAMPER_SECOND_MOVE	The ramper needed a second move to reach target	RWC
0x00040000	IIT_1_ACTIVE	IIt 1 active	R
0x00080000	IIT_2_ACTIVE	IIt 2 active	R
0x00100000	REFSEARCH_FINISCHED	Reference search finished	R
0x00200000	Y2_USED_FOR_BRAKING	Fourth phase used for braking	R
0x00800000	STEPDIR_INPUT_AVAILABLE	Signals that StepDir is available	R
0x01000000	RIGHT_REF_SWITCH_AVAILABLE	Signals that REF_R is available	R
0x02000000	HOME_REF_SWITCH_AVAILABLE	Signals that REF_H is available	R
0x04000000	LEFT_REF_SWITCH_AVAILABLE	Signals that REF_L is available	R
0x08000000	ABN2_FEEDBACK_AVAILABLE	Signals that ABN2 feedback is available	R
0x10000000	HALL_FEEDBACK_AVAILABLE	Signals that hall feedback is available	R
0x20000000	ABN1_FEEDBACK_AVAILABLE	Signals that ABN1 feedback is available	R
0x40000000	SPI_FLASH_AVAILABLE	Signals that an external SPI flash is available	R
0x80000000	I2C_EEPROM_AVAILABLE	Signals that an external I2C EEPROM is available	R

Table 59. All flags in the parameter GENERAL_ERROR_FLAGS

MASK	FLAG	DESCRIPTION	ACCESS
0x00000001	CONFIG_ERROR	Verification of config storage failed	R
0x00000002	TMCL_SCRIPT_ERROR	TMCL Script not available	R
0x00000004	HOMESWITCH_NOT_FOUND	Reference search for home switch failed	R
0x00000020	HALL_ERROR	Signals an invalid hall state	RWC
0x00000200	WATCHDOG_EVENT	Watchdog reset indication	RWC
0x00002000	EXT_TEMP_EXCEEDED	External temperature exceeded	RWC
0x00004000	CHIP_TEMP_EXCEEDED	Chip temperature threshold exceeded	RWC
0x00010000	I2T_1_EXCEEDED	Signals that I2t limit 1 was exceeded	RWC
0x00020000	I2T_2_EXCEEDED	Signals that I2t limit 2 was exceeded	RWC
0x00040000	EXT_TEMP_WARNING	External temperature warning threshold exceeded	RWC

MASK	FLAG	DESCRIPTION	ACCESS
0x00080000	SUPPLY_OVERVOLTAGE_WARNING	Supply overvoltage warning threshold exceeded	RWC
0x00100000	SUPPLY_UNDERVOLTAGE_WARNING	Supply voltage below undervoltage warning threshold	RWC
0x00200000	ADC_IN_OVERVOLTAGE	ADC IN over 2V while ADC enabled	RWC
0x00400000	FAULT_RETRY_HAPPEND	The set number of max. retries was exceeded without recovering	RWC
0x00800000	FAULT_RETRIES_FAILED	All retries of a detected fault failed	RWC
0x01000000	CHIP_TEMP_WARNING	Chip temperature warning threshold exceeded	RWC
0x04000000	HEARTBEAT_STOPPED	Heartbeat stopped	RWC

Table 60. All flags in the parameter GDRV_ERROR_FLAGS

MASK	FLAG	DESCRIPTION	ACCESS
0x00000001	U_LOW_SIDE_OVERCURRENT	U low side overcurrent	RWC
0x00000002	V_LOW_SIDE_OVERCURRENT	V low side overcurrent	RWC
0x00000004	W_LOW_SIDE_OVERCURRENT	W low side overcurrent	RWC
0x00000008	Y2_LOW_SIDE_OVERCURRENT	Y2 low side overcurrent	RWC
0x00000010	U_LOW_SIDE_DISCHARGE_SHORT	U low side discharge short	RWC
0x00000020	V_LOW_SIDE_DISCHARGE_SHORT	V low side discharge short	RWC
0x00000040	W_LOW_SIDE_DISCHARGE_SHORT	W low side discharge short	RWC
0x00000080	Y2_LOW_SIDE_DISCHARGE_SHORT	Y2 low side discharge short	RWC
0x00000100	U_LOW_SIDE_CHARGE_SHORT	U low side charge short	RWC
0x00000200	V_LOW_SIDE_CHARGE_SHORT	V low side charge short	RWC
0x00000400	W_LOW_SIDE_CHARGE_SHORT	W low side charge short	RWC
0x00000800	Y2_LOW_SIDE_CHARGE_SHORT	Y2 low side charge short	RWC
0x00001000	U_BOOTSTRAP_UNDERVOLTAGE	U bootstrap undervoltage	RWC
0x00002000	V_BOOTSTRAP_UNDERVOLTAGE	V bootstrap undervoltage	RWC
0x00004000	W_BOOTSTRAP_UNDERVOLTAGE	W bootstrap undervoltage	RWC
0x00008000	Y2_BOOTSTRAP_UNDERVOLTAGE	Y2 bootstrap undervoltage	RWC
0x00010000	U_HIGH_SIDE_OVERCURRENT	U high side overcurrent	RWC
0x00020000	V_HIGH_SIDE_OVERCURRENT	V high side overcurrent	RWC
0x00040000	W_HIGH_SIDE_OVERCURRENT	W high side overcurrent	RWC
0x00080000	Y2_HIGH_SIDE_OVERCURRENT	Y2 high side overcurrent	RWC
0x00100000	U_HIGH_SIDE_DISCHARGE_SHORT	U high side discharge short	RWC
0x00200000	V_HIGH_SIDE_DISCHARGE_SHORT	V high side discharge short	RWC
0x00400000	W_HIGH_SIDE_DISCHARGE_SHORT	W high side discharge short	RWC
0x00800000	Y2_HIGH_SIDE_DISCHARGE_SHORT	Y2 high side discharge short	RWC
0x01000000	U_HIGH_SIDE_CHARGE_SHORT	U high side charge short	RWC
0x02000000	V_HIGH_SIDE_CHARGE_SHORT	V high side charge short	RWC
0x04000000	W_HIGH_SIDE_CHARGE_SHORT	W high side charge short	RWC

0x08000000	Y2_HIGH_SIDE_CHARGE_SHORT	Y2 high side charge short	RWC
0x20000000	GDRV_UNDERVOLTAGE	Gate driver undervoltage	RWC
0x40000000	GDRV_LOW_VOLTAGE	Gate driver low voltage	RWC
0x80000000	GDRV_SUPPLY_UNDERVOLTAGE	Supply undervoltage	RWC

GLOBAL PARAMETERS

Global parameters provide the capability to set and get not motion related parameters from TMC9660 Parameter Mode. These parameters are divided into the three banks 0, 2, and 3. To store the settings in nonvolatile memory, see the section *Storing System Settings in External Memory*.

For a comprehensive list of global parameters, see the *Table 61*, *Table 62,* and *Table 63* in the following section.

Bank 0

Bank 0 is designed for the configuration of general system settings. This includes communication details, IO-configurations, stimulus, and script start configurations.

Table 61. Full list of global parameters in bank 0

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
1	SERIAL_ADDRESS The module (target) address for RS485 and UART. Changes take effect after system restart if stored to ext memory. Only odd values are allowed (1, 3, 5, .. 255).	1 … 255	1	RWE
2	SERIAL_HOST_ADDRESS The Host address for RS485 and UART. Changes take effect after system restart.	1 … 255	2	RWE
3	HEARTBEAT_MONITORING_CONFIG Configuration to enable heartbeat monitoring. In mode 3 the heartbeat is considered stopped if both UART and SPI communication is stopped. While at least one of them is beating no safe stop is issued. 0: DISABLED 1: TMCL_UART_INTERFACE 2: SPI_INTERFACE 3: TMCL_UART_AND_SPI_INTERFACE	0, 1, 2, 3	0	RWE
4	HEARTBEAT_MONITORING_TIMEOUT [ms] Timeout above which a heartbeat is consider stopped.	1 … 4294967295	100	RWE
5	IO_DIRECTION_MASK Mask for setting the GPIOs to input/output. Setting a bit to 1 configures the corresponding GPIO as an output.	0 … 524287	0	RWE
6	IO_INPUT_PULLUP_PULLDOWN_ENABLE_MASK Mask for enabling pullup/pulldown for input pins. Setting a bit to 1 enables configuring a pull resistor on the corresponding GPIO.	0 … 524287	0	RWE
7	IO_INPUT_PULLUP_PULLDOWN_DIRECTION_MASK Mask for setting direction pullup/pulldown for input pins. Setting a bit to 1 sets the pull direction for the corresponding GPIO to PULLUP.	0 … 524287	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
10	WAKE_PIN_CONTROL_ENABLE Enable TMC9660 WAKE pin functionality to be able to be put the chip into power-down state and wake the chip up again later. False: DISABLED True: ENABLED	0, 1	0	RWE
11	GO_TO_TIMEOUT_POWER_DOWN_STATE Use this parameter to put TMC9660 into power-down state for a given time period. Note that if Pin wakeup is configured, the WAKE pin must be pulled to GND in order to power down and pulling the WAKE pin back up before the time elapses, the TMC9660 already wakes up. 0: T_250_MILLISEC 1: T_500_MILLISEC 2: T_1_SEC 3: T_2_SEC 4: T_4_SEC 5: T_8_SEC 6: T_16_SEC 7: T_32_SEC	0, 1, 2, 3, 4, 5, 6, 7	0	W
12	MAIN_LOOPS [1/s] Main loops per second.	0 ... 4294967295	0	R
13	TORQUE_LOOPS [1/s] Torque loops per second.	0 ... 4294967295	0	R
14	VELOCITY_LOOPS [1/s] Velocity loops per second.	0 ... 4294967295	0	R
77	AUTO_START_ENABLE Use automatic TMCL application start after power up. False: DISABLED True: ENABLED	0, 1	1	RWE
85	CLEAR_USER_VARIABLES Clear user variables on startup False: TRY_LOAD_FROM_STORAGE True: CLEAR	0, 1	0	RWE

Bank 2

Bank 2 provides access to User Variables, which are utilized by the scripting feature.

Table 62. Full list of global parameters in bank 2

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
0	USER_VARIABLE_0 User variable 0	-2147483648 ... 2147483647	0	RWE

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
1	USER_VARIABLE_1 User variable 1	-2147483648 ... 2147483647	0	RWE
2	USER_VARIABLE_2 User variable 2	-2147483648 ... 2147483647	0	RWE
3	USER_VARIABLE_3 User variable 3	-2147483648 ... 2147483647	0	RWE
4	USER_VARIABLE_4 User variable 4	-2147483648 ... 2147483647	0	RWE
5	USER_VARIABLE_5 User variable 5	-2147483648 ... 2147483647	0	RWE
6	USER_VARIABLE_6 User variable 6	-2147483648 ... 2147483647	0	RWE
7	USER_VARIABLE_7 User variable 7	-2147483648 ... 2147483647	0	RWE
8	USER_VARIABLE_8 User variable 8	-2147483648 ... 2147483647	0	RWE
9	USER_VARIABLE_9 User variable 9	-2147483648 ... 2147483647	0	RWE
10	USER_VARIABLE_10 User variable 10	-2147483648 ... 2147483647	0	RWE
11	USER_VARIABLE_11 User variable 11	-2147483648 ... 2147483647	0	RWE
12	USER_VARIABLE_12 User variable 12	-2147483648 ... 2147483647	0	RWE
13	USER_VARIABLE_13 User variable 13	-2147483648 ... 2147483647	0	RWE
14	USER_VARIABLE_14 User variable 14	-2147483648 ... 2147483647	0	RWE
15	USER_VARIABLE_15 User variable 15	-2147483648 ... 2147483647	0	RWE

Bank 3

Bank 3 is dedicated to the configuration of interrupt options in the scripting feature.

Table 63. Full list of global parameters in bank 3

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
0	TIMER_0_PERIOD [ms] Time between two interrupts.	0 ... 2147483647	0	RW
1	TIMER_1_PERIOD [ms] Time between two interrupts.	0 ... 2147483647	0	RW
2	TIMER_2_PERIOD [ms] Time between two interrupts.	0 ... 2147483647	0	RW
10	STOP_LEFT_TRIGGER_TRANSITION Stop left trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
11	STOP_RIGHT_TRIGGER_TRANSITION Stop right trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
12	HOME_RIGHT_TRIGGER_TRANSITION Stop home trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
13	INPUT_0_TRIGGER_TRANSITION Input 0 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
14	INPUT_1_TRIGGER_TRANSITION Input 1 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
15	INPUT_2_TRIGGER_TRANSITION Input 2 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
16	INPUT_3_TRIGGER_TRANSITION Input 3 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
17	INPUT_4_TRIGGER_TRANSITION Input 4 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
18	INPUT_5_TRIGGER_TRANSITION Input 5 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
19	INPUT_6_TRIGGER_TRANSITION Input 6 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
20	INPUT_7_TRIGGER_TRANSITION Input 7 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
21	INPUT_8_TRIGGER_TRANSITION Input 8 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
22	INPUT_9_TRIGGER_TRANSITION Input 9 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
23	INPUT_10_TRIGGER_TRANSITION Input 10 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
24	INPUT_11_TRIGGER_TRANSITION Input 11 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
25	INPUT_12_TRIGGER_TRANSITION Input 12 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
26	INPUT_13_TRIGGER_TRANSITION Input 13 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
27	INPUT_14_TRIGGER_TRANSITION Input 14 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
28	INPUT_15_TRIGGER_TRANSITION Input 15 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
29	INPUT_16_TRIGGER_TRANSITION Input 16 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW

NR.	PARAMETER/DESCRIPTION	RANGE	DEFAULT	R/W
30	INPUT_17_TRIGGER_TRANSITION Input 17 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW
31	INPUT_18_TRIGGER_TRANSITION Input 18 trigger transition. 0: OFF 1: RISING 2: FALLING 3: BOTH	0, 1, 2, 3	0	RW

ERRATA

This section describes known Parameter Mode issues, their restrictions, and their workarounds.

Erratum 1: TMCL Script GPIO input

Within the scripting feature the return value of the GPIO input (GIO command) is not passed into the accumulator. Therefore, no further processing of the read analog values and getting the digital inputs is possible.

As a workaround one analog input AIN3 can still be used as EXTERNAL_TEMPERATURE through the GAP.

For digital inputs the interrupts are available to trigger at least on their behavior given by global parameter bank 3.

Erratum 2: IIT calculation

The internal IIT sum calculation can overflow when $(I_{measured}[mA])^2 \cdot \frac{t_{max}[ms]}{100} > 0xFFFFFFFF$, where $I_{measured}$ is the measured total motor current at any given time and t_{max} is the larger of the two configured time windows.

This overflow can lead to IIT sums being built up drastically slower and triggering the motor shutdown accordingly late and non-deterministic, which would break the functionality of the IIT feature as a protection mechanism.

There are ways to mitigate this problem manually:

Make sure the larger time window does not exceed $t_{max}[ms] = \frac{0xFFFFFFFF}{I_{max}[mA]^2} \cdot 100$, where $I_{max}[mA]$ is the maximum expected motor current (e.g. for a maximum expected motor current of 8000 mA, $t_{max}[ms] = 6710$).
The maximum target motor current can be limited by setting the maxTorque and maxFlux parameters. A safety margin in the maximum time window is highly recommended, as the max. currents can be temporarily exceeded by the control loops.

If a high maximum current is expected and the resulting maximum time window is too short for the desired application, an overflow can be prevented by lowering the current resolution from mA to e.g., 0.01A through the currentScalingFactor, by dividing it by, in this case, 10. All real-world currents would then be interpreted in units of 0.01A.

Furthermore, the two IIT sums use a shared ring buffer but a bug in the logic leads to the ring buffer sizes not always being calculated correctly.

Two ensure correct functionality:

- THERMAL_WINDING_TIME_CONSTANT_1 and THERMAL_WINDING_TIME_CONSTANT_2 should be written explicitly, not relying on the configured default values and
- THERMAL_WINDING_TIME_CONSTANT_1 and THERMAL_WINDING_TIME_CONSTANT_2 should be written twice in a row to trigger a recalculation in the software which fixes the issue.

Writing a new THERMAL_WINDING_TIME_CONSTANT_1 and/or THERMAL_WINDING_TIME_CONSTANT_2 when IIT is active might spontaneously trigger one of the flags that indicate a limit being exceeded. After clearing the flags, IIT operates normally again.

Erratum 3: Exiting to bootloader with ongoing motor commutation

The system does not automatically turn off the motor commutation when exiting from parameter mode to the bootloader, for instance, when using the Boot TMCL command described in *Table 18*.

Exiting the parameter mode while a motor movement is ongoing could lead to unexpected and potentially dangerous behavior. For this reason, it is very important to make sure the system is manually set to “System off” through the COMMUTATION_MODE parameter before exiting to the bootloader.

Erratum 4: Control loop target delays

When writing a target torque in torque mode to the corresponding parameter, the update to the target torque register is running in a 1kHz loop, which leads to the target torque update being delayed up to 1ms. The same is true for writing a target flux.

Position and velocity mode only have this same delay on the very first written target value when transitioning to position/velocity regulation mode from another regulation mode.

In position mode the target velocity parameter is synched in a 1kHz loop. This leads to the software parameter being heavily quantized. The actual internal target is not though, so that behavior does not influence the actual velocity controller performance but only the display to the user.

Erratum 5: SPI subordinate not being disabled

SPI1 is configured as SPI subordinate when it should be disabled, making SPI1 unusable for e.g. SPI encoder.

SPI subordinate can be manually disabled by sending a TMCL command with the *Operation* code **146**, *Type* **4**, *Motor/Bank* **9** and *Data* **0** (see the section *Communication Interfaces*).

Erratum 6: When using SPI encoder, the direction must be specified.

When using an SPI encoder, the direction parameter SPI_ENCODER_DIRECTION must be set. The default value is not applied correctly. Otherwise, the parameter SPI_ENCODER_COMMUTATION_ANGLE is not calculated correctly.

REVISION HISTORY

Table 64. Revision History

Revision Number	Revision Date	Change(s)
0	02/25	Initial release