TMC5221/TMC5222

16V, 1.4ARMS, Ultra-Small Stepper Driver and Controller

General Description

The TMC5221/TMC5222 are miniaturized, smart, high-performance, single-axis stepper motor controller and driver ICs optimized for mobile battery-operated equipment and space-constrained applications. These come with serial interfaces (SPI for TMC5221 and I^2C for TMC5222), and extensive safety, protection and diagnostic capabilities. Highest integration, sensorless load feedback, high energy efficiency, and a number of low-power modes enable miniaturized and cost-effective solutions. StealthChop2™ ensures absolutely noiseless operation combined with maximum efficiency and optimal motor torque control. TriCoder allows full-scale current detecting and monitoring of the external force moving the motor.

The TMC5221/TMC5222 combine a flexible, eight-point ramp generator with reduced jerk for automatic positioning with a smart 256 microstep 16V, $1.4A_{RMS}$ stepper motor driver with non-dissipative integrated current sensing (ICS).

The TMC5221/TMC5222 integrate safety functions like automatic safe position motion triggered by two hardware inputs and OTP-based limiting values to prevent any hazardous motion or out-of-specification motor current.

The TMC5221/TMC5222 feature abundant diagnostics and protection functions such as adaptive short-circuit protection and overcurrent protection, ADC supply voltage and temperature monitoring.

Applications

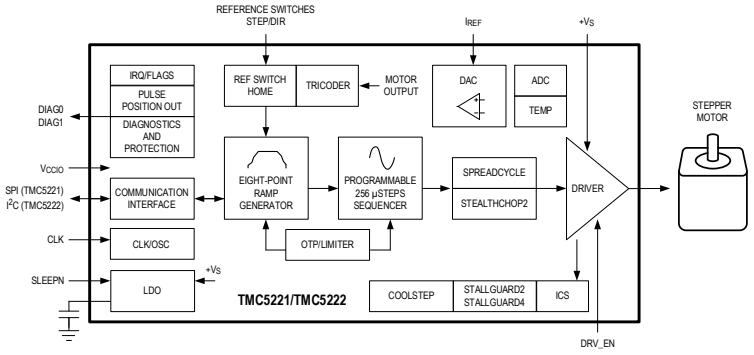
- Wearables and Personal Portable Devices
- Optical Systems, Lens Control
- CCTV, Surveillance, and Conference Systems
- Patch, Insulin Pumps, and Liquid Handling
- Small Printing and Scanning Devices
- Lab and Office Automation

Benefits and Features

- 2.1V to 16V DC Supply Voltage Range
- 1.1V to 2V Interface Level Fits Mobile CPUs
- Low $R_{DS(ON)}$ (HS + LS): 0.17Ω Typical (T_A = +25°C)
- Up to $1.4A_{RMS}$ Coil Current (2A Sine Peak)
- Fully-Integrated Lossless Current Sensing (ICS)
- Eight-Point Motion Controller for Minimum Jerk
- STEP/DIR Interface with MicroPlyer™ Interpolation
- SPI (TMC5221) or I^2C (TMC5222) Interface
- TriCoder Sensorless Standstill Steploss Detection
- High Resolution with 256 Microsteps per Full Step
- StealthChop2™ Silent Motor Operation
- SpreadCycle™ Highly Dynamic Chopper Mode
- StallGuard2™ and StallGuard4™ Sensorless Motor Load Detection
- CoolStep™ Current Control Saves Energy up to 75%
- ADC for Supply Voltage Supervision
- Passive Braking and Freewheeling Mode
- Bridge Disable, Reset, and Low-Power Mode
- OTP Preconfiguration and Safe Parameter Limits
- Motor Phase and Chip Temperature Measurement
- 2.3mm × 2.49mm, 30-Pin, WLCSP with 0.4mm Pitch

Ordering Information appears at end of data sheet.

Simplified Block Diagram



19-102154; Rev.0, 07/26

© 2026 Analog Devices, Inc. All rights reserved. Trademarks and registered trademarks are the property of their respective owners.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Absolute Maximum Ratings

V_S to GND ..... -0.3V to 18V
V_{DD1V8} to GND ..... -0.3V to 2.2V
PGND to GND ..... -0.3V to +0.3V
OA1, OA2, OB1, OB2 ..... -0.3V to V_S + 0.3V
SLEEPN ..... -0.3V to V_{CC_IO} + 0.3V
I_{REF} to GND ..... -0.3V to V_{DD} + 0.3V
V_{CC_IO} to GND ..... -0.3V to 2.2V
Logic I/O Voltage to GND ..... -0.3V to V_{CC_IO} + 0.3V
Operating Temperature Range ..... -40°C to +125°C
Junction Temperature ..... +160°C
Storage Temperature Range ..... -65°C to +150°C
Soldering Temperature (Reflow) ..... +260°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Package Information

30 WLCSP	
Package Code	W302V2Z+1
Outline Number	*21-100779*
Land Pattern Number	Refer to *Application Note 1891*
Thermal Resistance, Four-Layer Board: JEDEC PCB, 2 × 2 Via Array, M1 Traces, M2/M3 90% Cu, M4 25% Cu	
Junction-to-Ambient (θ_{JA})	49.38°C/W

For the latest package outline information and land patterns (footprints), go to *https://www.analog.com/en/design-center/packaging-quality-symbols-footprints/package-index.html*. Note that a "+", "#", or "-" in the package code indicates RoHS status only. Package drawings may show a different suffix character, but the drawing pertains to the package regardless of RoHS status.

Package thermal resistances were obtained using the method described in JEDEC specification JESD51-7, using a four-layer board. For detailed information on package thermal considerations, refer to *https://www.analog.com/en/technical-articles/thermal-characterization-of-ic-packages.html*.

Electrical Characteristics

(V_S = 2.1V to 16V, R_{REF} = 60.4kΩ 0.1%, typical values assume T_A = +25°C, V_S = 12V, V_{CC_IO} = 1.2V, SLEEPN = V_{CC_IO} and DRV_EN = V_{CC_IO}. Limits are 100% tested at T_A = +25°C. Limits cover the operating temperature range and relevant supply voltage range, and are guaranteed by design and characterization.)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
POWER SUPPLY							
Supply Voltage Range	V_S			2.1		16	V
Sleep Mode Current Consumption	I_{VS}	V(SLEEPN) = 0, HV outputs pulled up	25°C		0.024	1	µA
			85°C		0.3		µA
Low Power Quiescent Current Consumption	I_{VS}	V(SLEEPN) = 1, QSC_STS_ENA = 1, ADC_EN = 0, QSC_TRICODER_EN = 1	25°C		50	70	µA
			85°C		60		
Nap Mode 2 Current Consumption	I_{VS}	V(SLEEPN) = 1, QSC_STS_ENA = 1, ADC_EN = 0, QSC_TRICODER_EN = 0	25°C		23	40	µA
			85°C		30		
Quiescent Current Consumption	I_{VS}	Driver off			1.9	2.9	mA
Logic I/O Supply	V_{CC_IO}			1.1		2	V
Sleep Mode Current Consumption	I_{VCC}	V(SLEEPN) = 0	25°C			1	µA
			85°C		0.2		
Quiescent Current Consumption	I_{VCC}	V(SLEEPN) = 1 No external clk			16		µA
LOGIC LEVEL INPUTS/OUTPUTS							
Input Voltage Level (High)	V_{IH}			0.65x V_{CC_IO}			V
Input Voltage Level (Low)	V_{IL}					0.35x V_{CC_IO}	V
Input Hysteresis	V_{HYS}				0.08x V_{CC_IO}		mV
Pull-Down Resistor	R_{PD}	To GND		60	100	140	kΩ
Pull-Up Resistor	R_{PU}	To V_{CC_IO}		60	100	140	kΩ
Open-Drain Output Logic-Low Voltage	V_{OL}	ILOAD = 2mA				0.2	V
Push-Pull Output Logic-High Voltage	V_{OH}	I_{LOAD} = 2mA		V_{CC_IO} - 0.2			V
Open-Drain Output Logic-High Leakage Current	I_{OH}	V_{CC_IO} = 1.8V		-1		+1	µA

(V_S = 2.1V to 16V, R_{REF} = 60.4kΩ 0.1%, typical values assume T_A = +25°C, V_S = 12V, V_{CC_IO} = 1.2V, SLEEPN = V_{CC_IO} and DRV_EN = V_{CC_IO}. Limits are 100% tested at T_A = +25°C. Limits cover the operating temperature range and relevant supply voltage range, and are guaranteed by design and characterization.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
OUTPUT SPECIFICATIONS						
Output On-Resistance Low Side	R_{ONLS}	FS_SENSE = 0b00		0.2	0.35	Ω
		FS_SENSE = 0b01		0.1	0.18	
		FS_SENSE = 0b10		0.07	0.12	
		FS_SENSE = 0b11		0.05	0.09	
Output On-Resistance High Side	R_{ONHS}			0.12	0.24	Ω
Output Leakage (Driver Off)	I_{LEAK}	25°C	-5		+5	µA
		125°C	-35		+35	
Output Slew Rate	SR			200		V/µs
PROTECTION CIRCUITS						
Overcurrent Protection Threshold Low Side	OCP, LS	FS_SENSE = 0b00	0.825			A
		FS_SENSE = 0b01	1.65			
		FS_SENSE = 0b10	2.475			
		FS_SENSE = 0b11	3.3			
Overcurrent Protection Threshold High Side	OCP, HS	FS_SENSE = 0b00	1			A
		FS_SENSE = 0b01	2.2			
		FS_SENSE = 0b10	3.6			
		FS_SENSE = 0b11	4			
Overcurrent Protection Blanking Time	T_{OCP}		0.9	1.6	2.6	µs
UVLO Threshold on V_S	UVLO	V_S rising	1.7	1.8	1.9	V
UVLO Threshold on V_S Hysteresis	UVLOHYS			0.11		V
UVLO Threshold on V_{CC_IO}	UVLOVCC	V_{CC_IO} rising		0.82	1.05	V
UVLO Threshold on V_{CC_IO} Hysteresis	UVLOVCCHY S			30		mV
Thermal-Protection Threshold Temperature	T_{SD}			+155		°C
Thermal-Protection Temperature Hysteresis				20		°C
CURRENT REGULATION						
I_{REF} Pin Resistor	R_{REF}			60.4		kΩ
I_{REF} Pin Resistor Threshold Low	R_{REF_TL}		47		57	kΩ

(V_S = 2.1V to 16V, R_{REF} = 60.4kΩ 0.1%, typical values assume T_A = +25°C, V_S = 12V, V_{CC_IO} = 1.2V, SLEEPN = V_{CC_IO} and DRV_EN = V_{CC_IO}. Limits are 100% tested at T_A = +25°C. Limits cover the operating temperature range and relevant supply voltage range, and are guaranteed by design and characterization.)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
I_{REF} Pin Resistor Threshold High	R_{REF_TH}			63		78	kΩ
I_{REF} Output Voltage	V_{REF}			0.882	0.9	0.918	V
Full-Scale Current Constant	KIFSSNS	FS_SENSE= 0b11, FS_GAIN = 0b11			0.033		A/kΩ
Current Regulation Accuracy with External Reference	I_{ACC_EXT}	I_{TARGET} from 35% to 100% FS_SENSE= 0b11, FS_GAIN = 0b11, the typical value is 1 sigma at 25°C. (Note 1)	External Reference	-5	0.6	5	%
Current Regulation Accuracy with Internal Reference	I_{ACC_INT}	I_{TARGET} from 35% to 100% FS_SENSE= 0b11, FS_GAIN = 0b11, the typical value is 1 sigma at 25°C. (Note 1)	Internal Reference	-7	0.6	7	%
FUNCTIONAL TIMING							
Sleep Time	Tsleep	SLEEPN = 1 to OUT_x three-state.				50	µs
Wake-Up Time from Sleep	Twake	SLEEPN = 0 to normal operation.			500	1000	µs
Switch from Quiescent Mode to On-State	Ton	Assuming typical values on oscillators and internal oscillator used.			230		µs
Enable Time	Ten	Time from DRV_EN pin rising edge to driver on.				1	µs
Disable Time	Tdis	Time from DRV_EN pin falling edge to driver off.				10	µs
CLOCK OSCILLATOR AND INPUT							
Internal Clock Frequency	Fclkosc			11.9	12.5	13.2	MHz
External Clock Frequency	Fclk			8	16	20	MHz
External Clock Duty Cycle	Tclkl			40		60	%
External Clock Detection in Cycles				4		8	
External Clock Timeout Detection in				12		16	Cycles

(V_S = 2.1V to 16V, R_{REF} = 60.4kΩ 0.1%, typical values assume T_A = +25°C, V_S = 12V, V_{CC_IO} = 1.2V, SLEEPN = V_{CC_IO} and DRV_EN = V_{CC_IO}. Limits are 100% tested at T_A = +25°C. Limits cover the operating temperature range and relevant supply voltage range, and are guaranteed by design and characterization.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Cycles of Internal Fclkosc						
External Clock Detection Lower Frequency Threshold			4			MHz
Internal Clock Frequency in Low Power Quiescent Mode	Fclkqsc	V(SLEEPN) = 1, QSC_STS_ENA = 1	533	609	665	kHz
SPI TIMING (TMC5221 ONLY)						
SCK Valid Before or After Change of CSN	t_{CC}		t_{SCK}			ns
CSN High Time	t_{CSH}	Active system clock (depending on IC-state: qsc mode, active, external clock supplied).	4 × t_{CLK}			ns
SCK Low Time	t_{CL}		20			ns
SCK High Time	t_{CH}		20			ns
SCK Frequency	f_{SCK}	V_{CC_IO} = 1.8V			10	MHz
		V_{CC_IO} = 1.2V			8	
SDI Setup Time Before SCK Rising Edge	t_{DU}		10			ns
SDI Hold Time After SCK Rising Edge	t_{DH}		10			ns
Data Out Valid Time After SCK Falling Edge	t_{DO}	V_{CC_IO} = 1.8V		27	40	ns
		V_{CC_IO} = 1.2V		36	51	
SPI Input Filter	t_{FILT}	Rising and falling edge		10		ns
I^2C TIMING (TMC5222 ONLY)						
SCL Clock Frequency	f_{SCL}				1	MHz
Bus Free Time Between a STOP and START Condition	t_{BUF}		0.5			µs
Low Period of the SCL Clock	t_{LOW}		0.5			µs
High Period of the SCL Clock	t_{HIGH}		0.26			µs
Data Setup Time	t_{SU}		50			ns
Data Hold Time	t_{HD}		0			ns
Setup time for Repeated START Condition	$t_{SU:STA}$				260	ns

(V_S = 2.1V to 16V, R_{REF} = 60.4kΩ 0.1%, typical values assume T_A = +25°C, V_S = 12V, V_{CC_IO} = 1.2V, SLEEPN = V_{CC_IO} and DRV_EN = V_{CC_IO}. Limits are 100% tested at T_A = +25°C. Limits cover the operating temperature range and relevant supply voltage range, and are guaranteed by design and characterization.)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Hold Time for (Repeated) START Condition	$t_{HD:STA}$					260	ns
Setup Time for STOP Condition	$t_{SU:STO}$					260	ns
STEP/DIRECTION TIMING							
STEP Frequency	Fstep	Maximum microstep resolution	DEDGE = 1			Fclk/4	MHz
			DEDGE = 0			Fclk/2	
Full-Step Frequency	Fps					Fclk/512	MHz
STEP High Time	Tsh			Tclk + 20			ns
STEP Low Time	Tsl			Tclk + 20			ns
DIR to STEP Setup Time	Tdsu			Tclk + 20			ns
DIR to STEP Hold Time	Tdsh			Tclk + 20			ns
DIR/STEP to CLK Setup Time	Tsu			10			ns
DIR/STEP to CLK Hold Time	Tsh			10			ns
ADC/TEMPERATURE AND V_S MEASUREMENT							
ADC Resolution					9		Bit
Temperature Measurement Resolution	TMEAS				1.0		°C
Temperature Measurement Accuracy	TSIGMA				1.5		°C
V_S Measurement Accuracy	VSSIGMA	V_S < 5V		-10		+10	%
		V_S ≥ 5V		-5		+5	
ADC Sample Rate	Fsample,ADC	Fclk,INT = 12.5MHz			Fclk,INT/3750		MHz
BEMF ENCODER							
Pull-Down Resistance on OA1/OB1	RPDBEMF	BEMF encoder activated			345	600	Ω
BEMF Encoder Hysteresis	BEMFHYS	BEMF_HYST = 0		8	10	12	mV
		BEMF_HYST = 1		23	25	27	
		BEMF_HYST = 2		47	50	53	

(V_S = 2.1V to 16V, R_{REF} = 60.4kΩ 0.1%, typical values assume T_A = +25°C, V_S = 12V, V_{CC_IO} = 1.2V, SLEEPN = V_{CC_IO} and DRV_EN = V_{CC_IO}. Limits are 100% tested at T_A = +25°C. Limits cover the operating temperature range and relevant supply voltage range, and are guaranteed by design and characterization.)

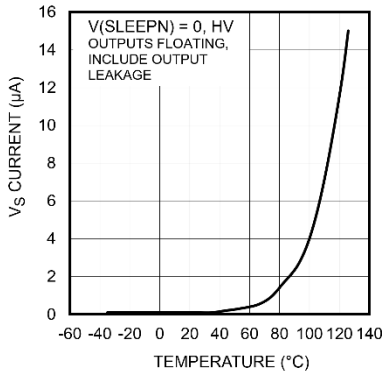
PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
		BEMF_HYST = 3	71	75	79	
		BEMF_HYST = 4	95	100	105	
		BEMF_HYST = 5	142	150	158	
		BEMF_HYST = 6	190	200	210	
		BEMF_HYST = 7	236	250	264	

Note 1: Characterized at bench and tested in production with wider limits.

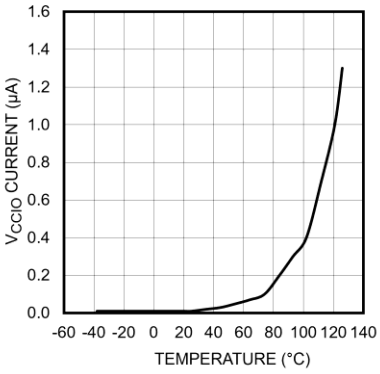
Typical Operating Characteristics

$V(V_S)$ = 16V, $V(V_{CCIO})$ = 2V

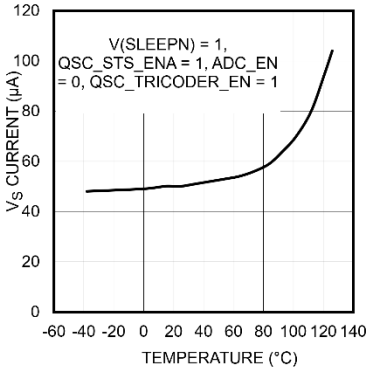
SLEEP MODE CURRENT CONSUMPTION vs. TEMPERATURE



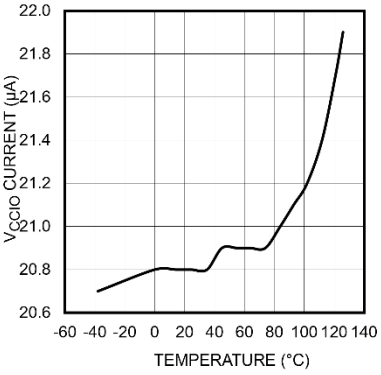
SLEEP MODE CURRENT CONSUMPTION vs. TEMPERATURE



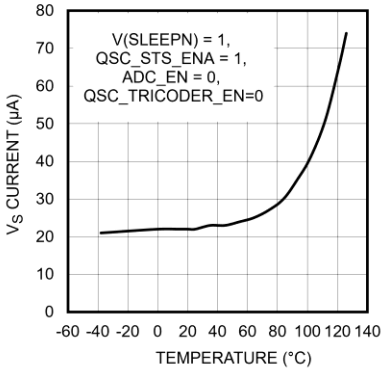
L. P. QUIESCENT CURRENT CONSUMPTION vs. TEMPERATURE



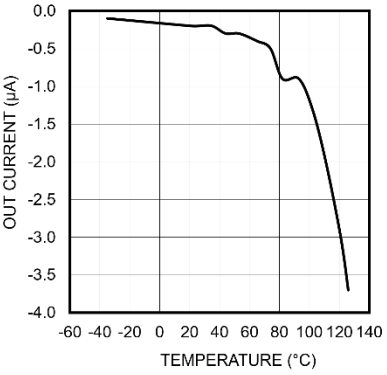
L. P. QUIESCENT CURRENT CONSUMPTION vs. TEMPERATURE



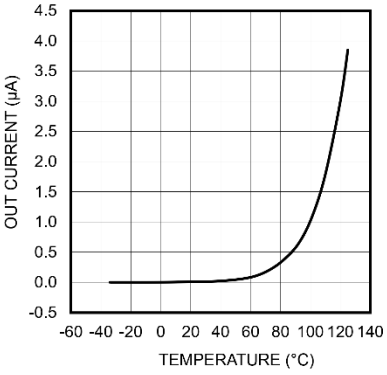
NAP MODE 2 CURRENT CONSUMPTION vs. TEMPERATURE



OUTPUT LEAKAGE AT 16V vs. TEMPERATURE



OUTPUT LEAKAGE AT 0V vs. TEMPERATURE



Timing Diagrams

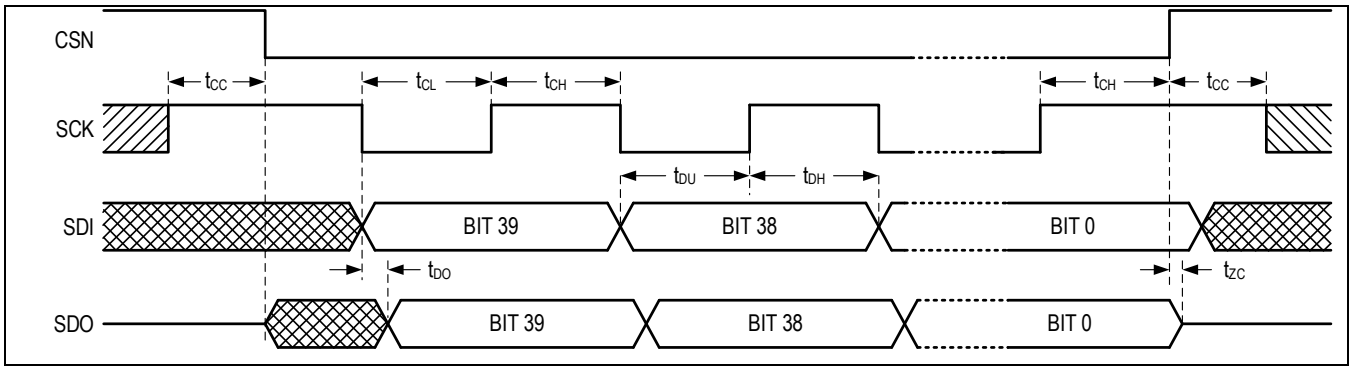


Figure 1. SPI Communication Interface Timing Diagram SPI Mode 3 (TMC5221 Only)

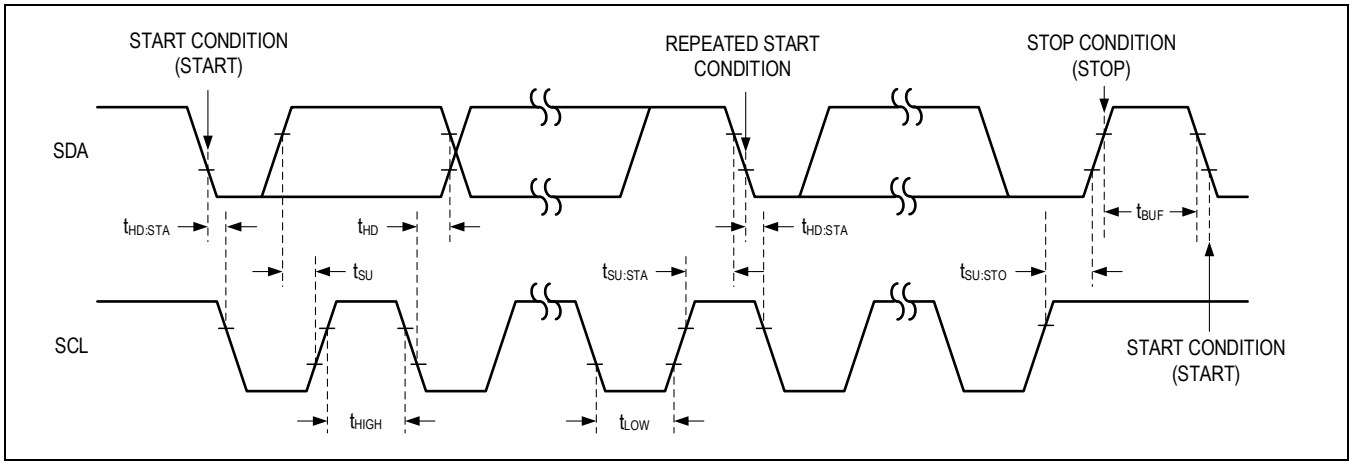
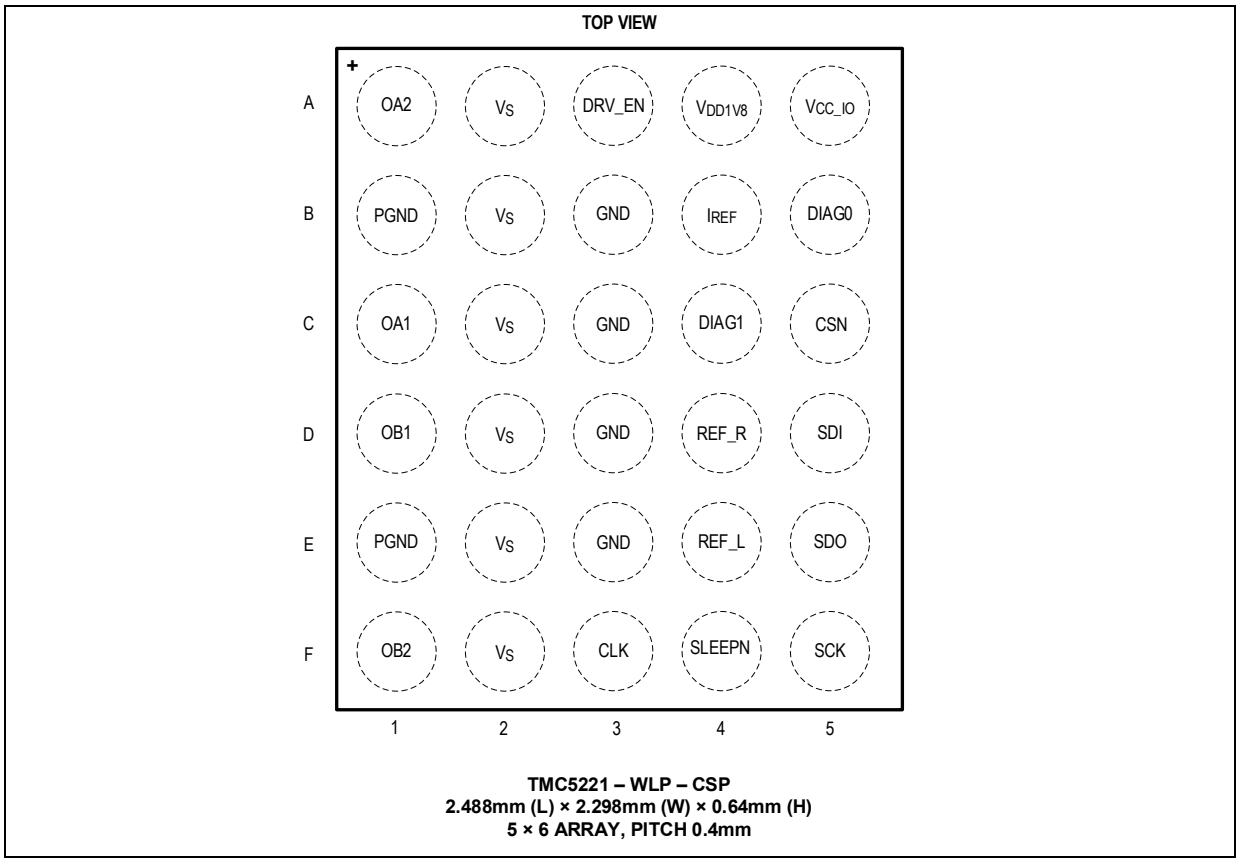
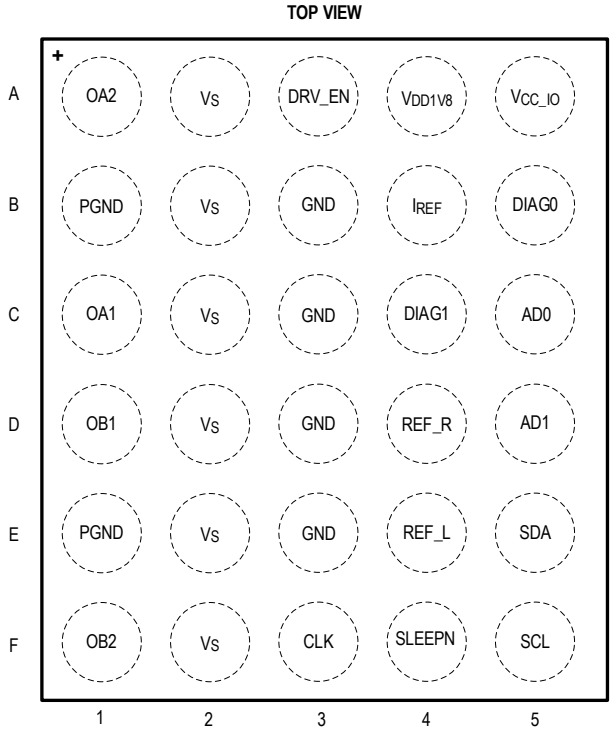


Figure 2. I^2C Communication Interface Timing Diagram (TMC5222 Only)

Pin Configurations



TMC5221 – WLP – CSP
2.488mm (L) × 2.298mm (W) × 0.64mm (H)
5 × 6 ARRAY, PITCH 0.4mm



TMC5222 – WLP – CSP
2.488mm (L) × 2.298mm (W) × 0.64mm (H)
5 × 6 ARRAY, PITCH 0.4mm

Pin Descriptions

PIN		NAME	FUNCTION	REF SUPPLY	Type
TMC5221	TMC5222				
A2, B2, C2, D2, E2, F2	A2, B2, C2, D2, E2, F2	V_S	Motor Supply Voltage. Provide filtering capacitor near pin with short loop to nearest PGND pin (or using the GND plane).	—	Supply
A4	A4	V_{DD1V8}	Output of Internal 1.8V Regulator. Attach a 2.2µF or larger ceramic capacitor to GND near to pin for best performance.	—	Supply
B4	B4	I_{REF}	Analog Reference Current for Current Scaling. Provide external resistor R_{REF} to GND if required.	VDD1V8	Analog Input
A5	A5	V_{CC_IO}	Digital IO Supply Voltage Provided from External Source to Define Circuit IO Level.	—	Supply
F3	F3	CLK	CLK Input. Tie to GND using short wire for internal clock or supply external clock. Internal clock-fail over circuit protects against loss of external clock signal.	VCC_IO	Digital Input (Pull-Down)
C5		CSN	SPI Chip Select Input (Active Low)	VCC_IO	Digital Input (Pull-Up)
F5		SCK	SPI Serial Clock Input	VCC_IO	Digital Input (Pull-Up)

D5		SDI	SPI Data Input	VCC_IO	Digital Input (Pull-Up)
E5		SDO	SPI Data Output (Three-State) in SPI Mode	VCC_IO	Digital Output
	E5	SDA	I^2C Serial Data Input/Output	VCC_IO	Digital Input (Pull-Up)
	F5	SCL	I^2C Serial Clock Input	VCC_IO	Digital Input (Pull-Up)
	C5	AD0	Hardwired Input Pin 0 Input for I^2C address	VCC_IO	Digital Input
	D5	AD1	Hardwired Input Pin 1 Input for I^2C address	VCC_IO	Digital Input
B5	B5	DIAG0	Configurable Diagnostics Output DIAG0. Use external pull-up resistor with 4.7kΩ or less in open-drain mode.	VCC_IO	Digital Output
A3	A3	DRV_EN	Driver Enable Pin. Drive this pin logic high to enable the output stage. Drive this pin logic low to tristate the output stage.	VCC_IO	Digital Input (Pull Down)
F4	F4	SLEEPN	Active-Low Power-Down Input/Reset Input. A short pulse to GND at SLEEPN pin resets the device and disables the power drivers. Apply a low level to bring the device to sleep mode. Once the IC returns from sleep mode/reset, it must be reconfigured before being used again. The latest register contents before going into sleep mode are not stored. While reconfiguring the IC, it is advised to hold the driver disabled with the drv_en_sw bit in the GCONF register. If not used, connect to V_{CC_IO}. **Note:** Do not use during high motor velocity to prevent from hard stops and potential back-EMF spikes.	VCC_IO	Digital Input (Pull-Down)
C1	C1	OA1	Motor Coil A Output 1, Motor 1	VS	Power Output
A1	A1	OA2	Motor Coil A Output 2, Motor 1	VS	Power Output
D1	D1	OB1	Motor Coil B Output 1, Motor 1	VS	Power Output
F1	F1	OB2	Motor Coil B Output 2, Motor 1	VS	Power Output
B1, E1, B3, C3, D3, E3	B1, E1, B3, C3, D3, E3	GND/PGND	Supply and System Ground. Connect to the GND plane. Provide a solid connection to GND plane for best heat transfer.	—	GND
C4	C4	DIAG1	Configurable Diagnostic Output DIAG1. Use external pull-up resistor with 4.7kΩ or less in open-drain mode.	VCC_IO	Digital Output
D4	D4	REF_R/DIR	Right Reference Switch Input REFR or Direction Input DIR in S/D Mode. or	VCC_IO	Digital Input (Pull-Down)

			External Input Trigger for Hard Emergency Stop. or Interrupt Input for Automatic Safe Position Event.		
E4	E4	REF_L/STEP	Left Reference Switch Input REFL or STEP Input in S/D Mode. or External Input Trigger for Hard Emergency Stop. or Interrupt Input for Automatic Safe Position Event.	VCC_IO	Digital Input (Pull-Down)

Functional Diagrams

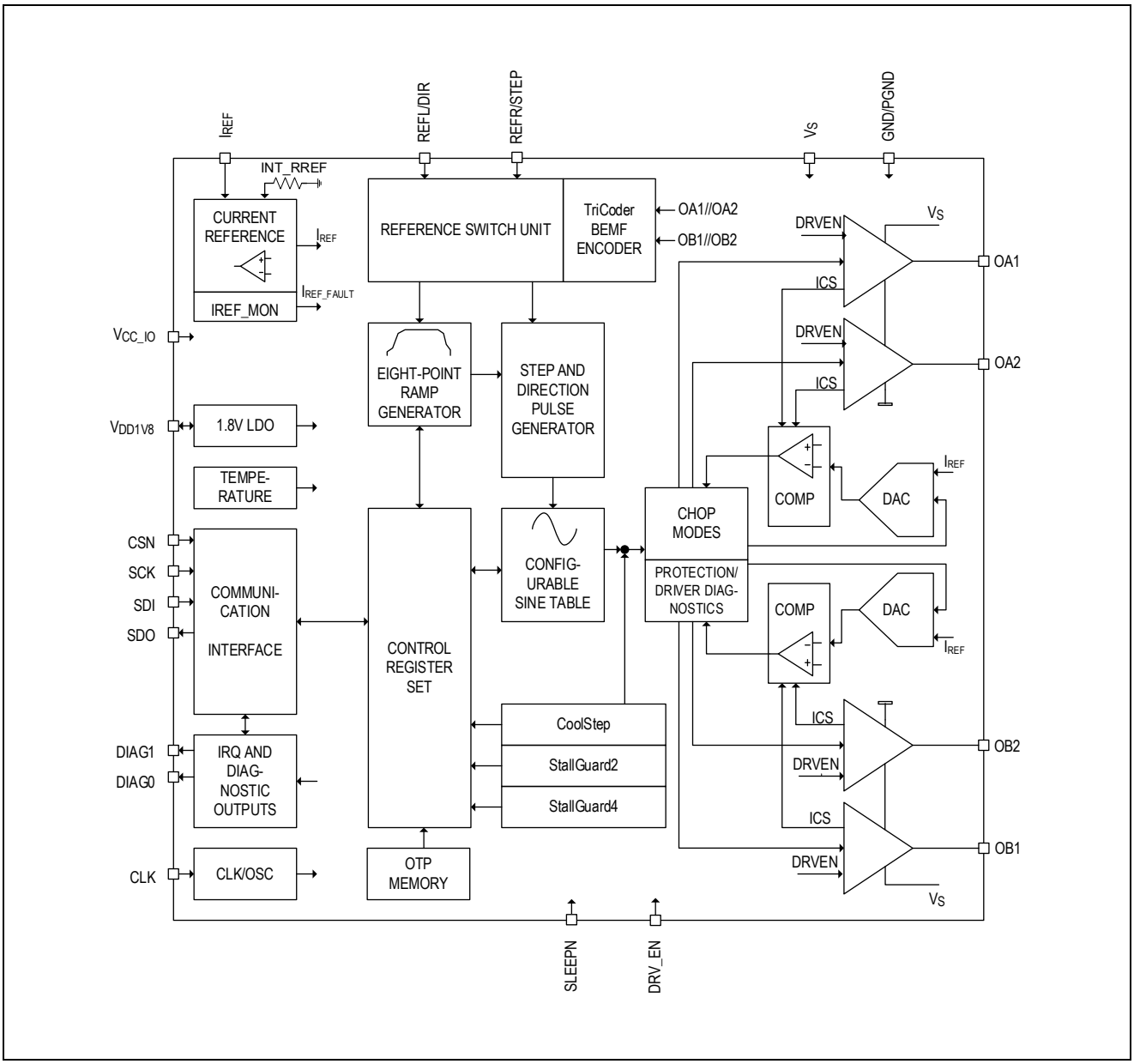


Figure 3. TMC5221 Block Diagram

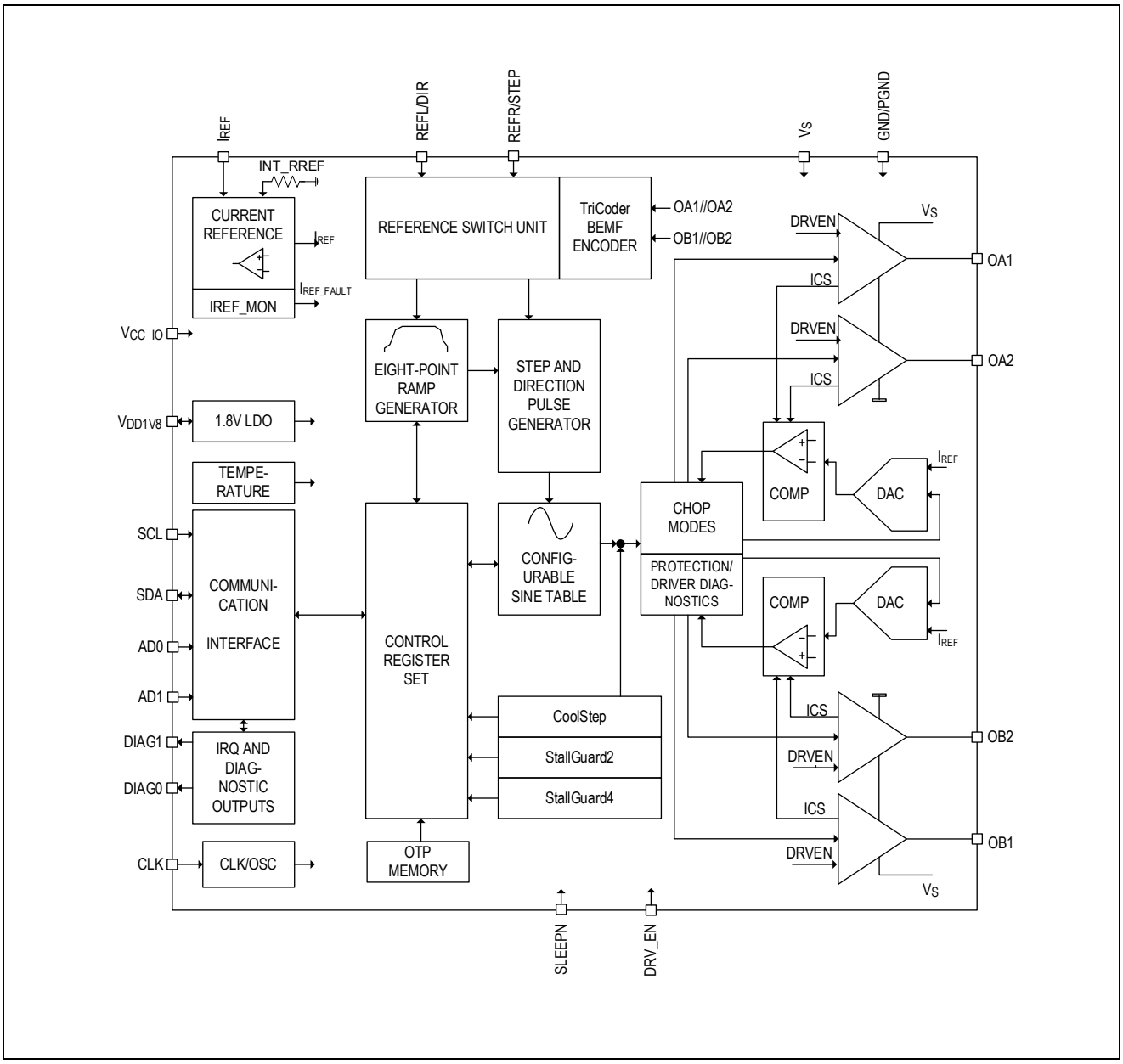


Figure 4. TMC5222 Block Diagram

Detailed Description

Principles Of Operation

The TMC5221/TMC5222 are intelligent stepper motor motion controller and driver chips (cDrivers). This smart power conversion component directly drives a two-phase stepper motor and interfaces to a CPU for configuration, control, and diagnostic feedback. All the functionality and logic to control and drive a stepper motor is integrated. No software is required to control the motor – only configuration and target positions or velocities must be provided. The TMC5221/TMC5222 offer numerous unique functions, which are enabled by the system-on-chip integration of the driver and controller.

Single-Axis Motion Controller and Driver

The TMC5221/TMC5222 include everything to control and drive a stepper motor. No external motion controller or step-pulse generator is required to control the motor. Just provide target positions and other ramp parameters through the serial interface. In addition, reference switches can be connected, example, for homing and monitoring. The two variants differ from the serial interface only (SPI and I^2C for the TMC5221 and TMC5222, respectively). The motion-controller mode is the default mode of operation. *Figure 5* and *Figure 6* show high-level block diagrams for both the variants.

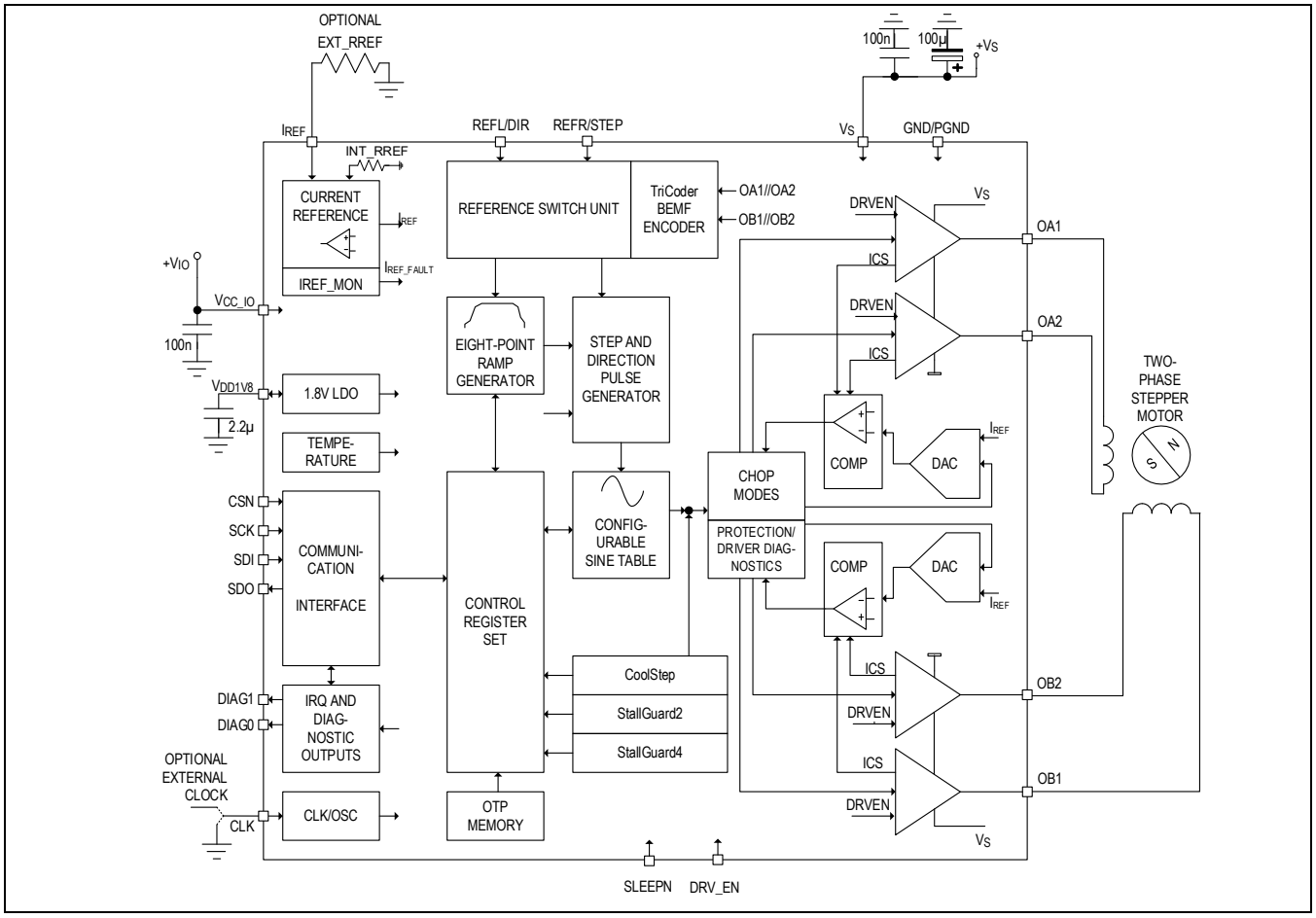


Figure 5. TMC5221 High-Level Block Diagram

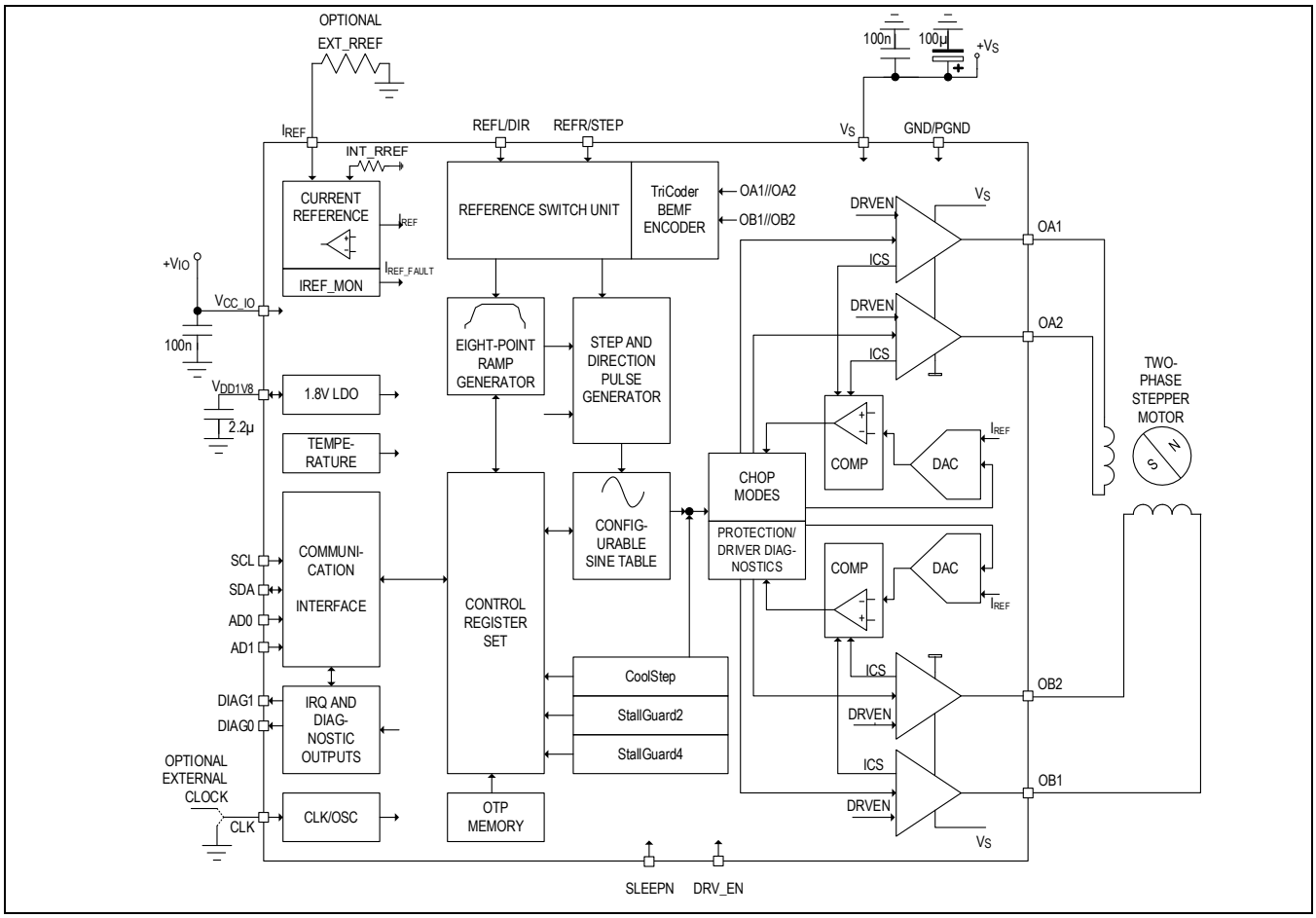


Figure 6. TMC5222 High-Level Block Diagram

Key Concepts

The TMC5221/TMC5222 implement advanced features exclusive to products from Analog Devices, Inc. *Table 1* highlights the key features. These features contribute toward greater precision, greater energy efficiency, higher reliability, smoother motion, and cooler operation in stepper-motor applications.

Table 1. TMC5221/TMC5222 Key Concepts

KEY CONCEPT	DESCRIPTION	FURTHER DETAILS
StealthChop2	No-noise, high-precision current-control algorithm for inaudible motion and inaudible standstill of the motor. Allows faster motor acceleration and deceleration than StealthChop™, and extends StealthChop to low standstill motor currents.	See the *StealthChop2* section for more information.
SpreadCycle	High-precision, cycle-by-cycle current control for highest dynamic movements.	See the *SpreadCycle Chopper* section for more information.
StallGuard2/4	Sensorless stall detection and mechanical load measurement. Sensorless homing saves end switches and warns in case of motor overload.	See the *StallGuard2* and *StallGuard4* sections for more information.
CoolStep	Uses StallGuard2/StallGuard4 measurement to adapt the motor current for best efficiency and lowest heat-up of the motor and driver.	See the *CoolStep* section for more information.

MicroPlyer	Microstep interpolator for obtaining full 256-microstep smoothness with lower resolution step inputs starting from full step.	See the *MicroPlyer* section for more information.
TriCoder	A sensorless standstill steploss detection feature making use of the motor back-EMF (BEMF) to detect motor motion in applications where the motor is disabled and a holding current cannot be applied.	See the *TriCoder (Back-EMF Sensorless Standstill Steploss* Detection section for more information.

In addition to these performance enhancements, Analog Devices' motor drivers offer safeguards to detect and protect against shorted outputs, output open-circuit, overtemperature, and undervoltage conditions for enhancing safety and recovery from equipment malfunctions.

Control Interfaces

The TMC5221 supports an SPI.

The SPI is a bit-serial interface synchronous to a bus clock. For every bit sent from the bus controller to the bus peripheral, another bit is sent simultaneously from the peripheral back to the controller. Communication between an SPI controller (example, an MCU) and the TMC5221 peripheral always consists of sending one 40-bit command word and receiving one 40-bit status word.

The TMC5222 supports an I^2C interface with partially configurable seven bit device address (OTP, AD0, AD1). Sequential write and read operations are possible with the option of an automatic register address increment. 10-bit addressing is not supported.

Moving and Controlling the Motor

Integrated Eight-Point Motion Controller

The integrated 32-bit motion controller automatically drives the motor to target positions, or accelerates to target velocities. All motion parameters can be changed on-the-fly. The motion controller recalculates immediately. A minimum set of configuration data consists of acceleration and deceleration values, and the maximum motion velocity. A start and a stop velocity are supported, as well as a second and a third acceleration and deceleration setting selected by velocity thresholds resulting in an eight-point velocity profile. These settings allow the adaptation of the motion profile to the motor torque profile as well as jerk reduction for near S-ramp performance. The integrated motion controller supports immediate reaction to mechanical reference switches, and to the sensorless stall detection StallGuard2 and StallGuard4.

The benefits of the integrated eight-point motion controller are:

- Flexible ramp shaping.
- Efficient use of motor torque for acceleration and deceleration allows higher machine throughput.
- Pseudo S-ramp for jerk reduction.
- Immediate reaction to stop and stall conditions.

Step and Direction Mode

If S-shaped motion profiles are needed, the internal motion controller can be disabled, and step and direction signals can be fed directly from an external motion controller like the TMC4361A (refer to the TMC4361A data sheet) or a CPU. In this case, the TMC5221/TMC5222 take care of the current control and deliver feedback on the state of the driver and motor. MicroPlyer automatically smoothens motion. This mode can be enabled by configuration through the serial interface. Active edges on the STEP input can be rising edge, or rising and falling edge. Using both edges cuts the toggle rate of the STEP signal in half, which is useful for control over slow interfaces such as optically-isolated couplers. The DIR input determines whether to step forward or back.

Automatic Standstill Power-Down and Power-Up

The automatic current reduction drastically reduces application power dissipation and cooling requirements. *Figure 7* shows the behavior of the current levels versus time.

Even a reduction to half of the run current reduces standstill power dissipation to approximately 25%. Standstill current, delay time, and decay parameters can be configured through the serial control interfaces.

Automatic freewheeling and passive motor braking are provided as an option for standstill.

Passive braking reduces motor standstill power consumption to zero, while still providing effective dampening and braking.

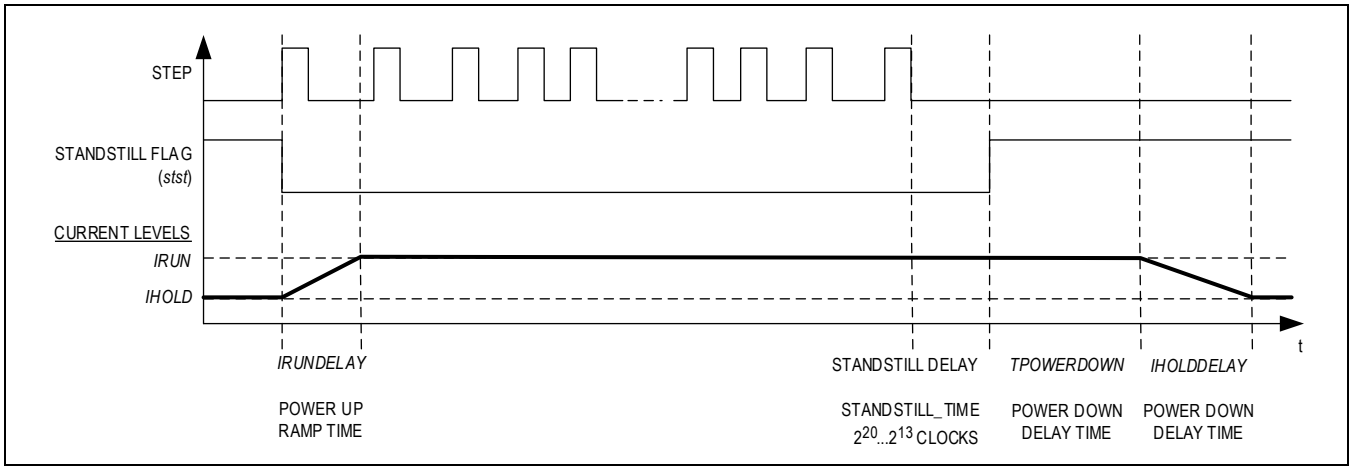


Figure 7. Automatic Motor Current Control at Standstill and Ramp-Up

StealthChop2 and SpreadCycle Driver

StealthChop2 is a voltage-chopper-based current controller. It guarantees absolute quiet motors, except for noise generated by ball bearings, especially in standstill and slow motion.

Unlike other voltage mode choppers, StealthChop2 does not require any configuration and is also able to control the motor current. It automatically learns the best settings during the first motion after power-up and further optimizes the settings in subsequent motions.

An initial homing sequence is sufficient for learning. Optionally, initial learning parameters can be loaded to the register set. StealthChop2 allows high motor dynamics, by reacting at once to a change of motor velocity.

For highest velocity applications, SpreadCycle is an alternative option to StealthChop2. StealthChop2 and SpreadCycle are intended for use in a combined configuration to leverage the benefits of both; StealthChop2 for no-noise standstill, silent and smooth performance; SpreadCycle at higher velocity for high dynamics and highest peak velocity at low vibration.

SpreadCycle is an advanced cycle-by-cycle chopper mode. It offers smooth operation and good resonance dampening over a wide range of speed and load. The SpreadCycle chopper scheme automatically integrates and tunes fast decay cycles to guarantee smooth zero-crossing performance.

The benefits of StealthChop2 and SpreadCycle driver are:

- Significantly improved microstepping with low-cost motors.
- Motor runs smoothly and quietly.
- Absolutely no standby noise.
- Reduced mechanical resonance improves torque output.

StallGuard2 and StallGuard4 (Mechanical Load Sensing)

StallGuard2 and StallGuard4 provide an accurate measurement of the load on the motor. These can be used for stall detection as well as other uses at loads below those that stall the motor, such as CoolStep load-adaptive current reduction.

This gives more information on the drive, allowing functions like sensorless homing and diagnostics of the drive mechanics. While StallGuard2 combines with SpreadCycle chopper, StallGuard4 uses a different principle to combine with StealthChop2.

The benefits of StallGuard2 and StallGuard4 are:

- Reference search without additional sensors.
- Use as a mechanical health indicator.
- React to defined load conditions.

CoolStep (Load Adaptive Current Control)

CoolStep drives the motor at the optimum current. It uses the StallGuard2 or StallGuard4 load measurement information to adjust the motor current to the minimum amount required in the actual load situation.

CoolStep results in energy savings and keeps the components cool. CoolStep increases the motor efficiency compared to standard operation with approximately 50% torque reserve because the motor is driven with optimum current.

The benefits of CoolStep are:

- Highest energy efficiency, power consumption decreased by up to 75%.
- Motor generates less heat.
- Improved mechanical precision.
- Less or no cooling.
- Improved reliability.
- Use of smaller motor is possible, less torque reserve required.
- Less motor noise due to less energy exciting motor resonances.

Standstill Steploss Detection

The sensorless standstill steploss detection feature, TriCoder, is based on a BEMF decoder. The BEMF decoder allows the detection of motor motion while the motor is disabled (that is, no active current is driven into the motor coils). This feature is especially useful for devices where a holding current cannot be applied due to power-saving requirements, but a certain amount of cogging torque or friction normally keeps the motor in position. In case the motor is turned by an external force, its motion can be tracked. The result can be used to trigger a new homing sequence in case the motor is moved, or to keep track of the number of steps done and correcting for it later.

LOW POWER MODES

Low-Power Quiescent Mode

The low-power quiescent mode is a special TMC5221/TMC5222 mode that reduces power consumption to a minimum while keeping the communication interface alive and maintaining the register contents. The chip power consumption is reduced to 23µA, respectively, to 50µA, when TriCoder remains active.

This mode can be entered by setting the QSC_STS_ENA bit in the GCONF (0x0) register to 1. Similar to the other stop and power-down modes, the motor driver stage is completely disabled when entering this mode by automatically clearing DRV_EN_SW in GCONF (0x0). When waking up by clearing QSC_STS_ENA, it is possible to set the DRV_EN_SW in the same write access. When using the automatic safe position interrupt procedure, the DRV_EN_SW is automatically enabled again.

By default, the TriCoder unit is still enabled during this mode to allow for position tracking. Using the QSC_TRICODER_EN bit in the BEMF_CONF (0x31) register, it can be switched off if not used to save additional energy.

External Reset and Sleep Mode

The reset and sleep mode are controlled with the SLEEPN pin. A short pulse on SLEEPN with a duration > 35µs (typical) results in a chip reset (also visible at the diagnostics outputs). Very short pulses below this duration are filtered out and do not have an effect on operation. If SLEEPN is kept at GND, the IC goes into the low-power standby state (sleep mode). All internal supplies are switched off.

In both cases, reset and standby, all internal register values and configurations are cleared and set to their defaults, and power bridges are off. After power-up or leaving the sleep mode and reset condition, the registers must be reconfigured. While reconfiguring the IC, it is advised to keep the bridge drivers disabled using the GCONF register bit DRV_EN_SW.

Do not pull down SLEEPN during high motor velocity, as energy fed back from the motor can boost V_S supply above the abs max, damaging the chip. If not used, connect SLEEPN to V_{CC_IO}.

Note: Pull-up and pull-down resistors are disconnected during sleep mode.

DEVICE INTERFACES

SPI (TMC5221 Only)

SPI Datagram Structure

The TMC5221 uses 40-bit SPI datagrams for communication with a microcontroller. Microcontrollers with hardware SPI are typically able to communicate using integer multiples of 8 bits. The CSN line of the device must stay active (= low) for the complete duration of the datagram transmission. For daisy-chained devices, the CSN line must stay active for the whole transmission that supports 40 + n × 8 bit (n == 0,1,2,3 …).

Each datagram sent to the device is composed of an address byte followed by four data bytes. This allows direct 32-bit data word communication with the register set. Each register is accessed by 32 data bits even if it uses less than 32 data bits. *Figure 8* shows the datagram structure. For simplification, each register is specified by a one-byte address:

- For a read access, the most significant bit of the address byte is 0.
- For a write access, the most significant bit of the address byte is 1.

All registers are readable. Most of them are read/write. Some registers are read-only and some write 1 to clear (example, GSTAT registers).

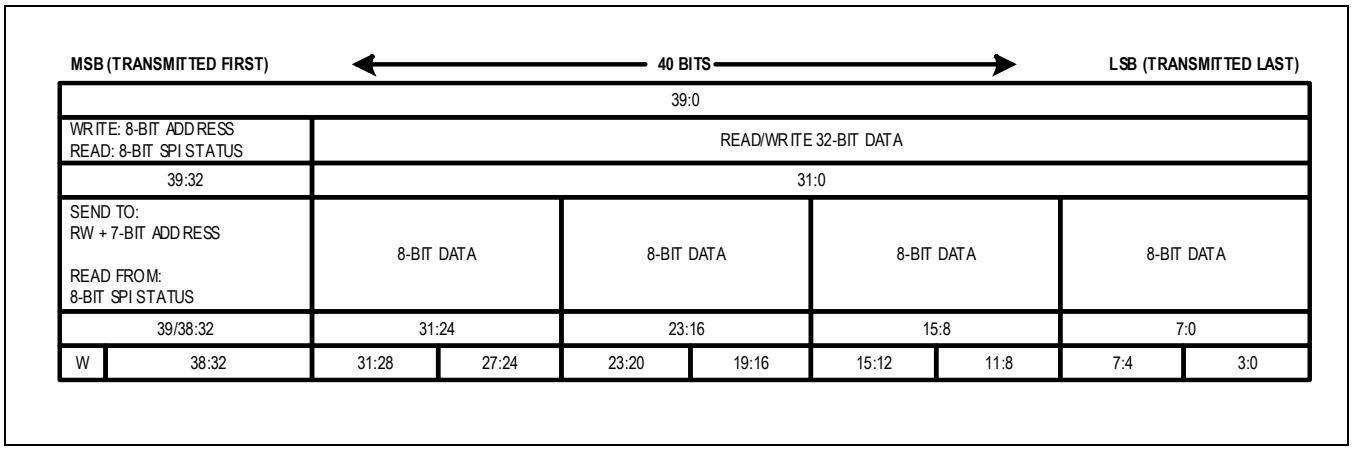


Figure 8. SPI Datagram Structure

Selection of Write/Read (WRITE_notREAD)

The read and write selection is controlled by the MSB of the address byte (bit-39 of the SPI datagram). This bit is 0 for read access and 1 for write access. The bit named W is a WRITE_notREAD control bit. The active-high write bit is the MSB of the address byte. Therefore, 0x80 must be added to the address for a write access. The SPI always delivers data back to the controller, independent of the W bit. The data transferred back is the data read from the address, which is transmitted with the previous datagram, if the previous access is a read access. If the previous access is a write access, the data read back mirrors the previously received write data. Therefore, the difference between a read and a write access is that the read access does not transfer data to the addressed register. It only transfers the address and its 32 data bits are dummies. The following read or write access delivers back the data read from the address transmitted in the preceding read cycle.

A read access request datagram uses dummy write data. Read data is transferred back to the controller with the subsequent read or write access. Hence, multiple registers can be read in a pipelined fashion.

Whenever data is read from or written to the TMC5221, the MSBs delivered back contain the SPI status. The SPI_STATUS is a number of eight selected status bits.

For an example of the SPI read/write flow, see *Table 2*.

For a read access to the register (XACTUAL) with the address 0x18, the address byte must be set to 0x18 in the access preceding the read access. For a write access to the register (VACTUAL), the address byte must be set to 0x80 + 0x19 = 0x99. For read access, the data bit might have any value (-). Therefore, these can be set to 0.

Table 2. SPI Read/Write Example Flow

ACTION	DATA SENT TO TMC5221	DATA RECEIVED FROM TMC5221
Read XACTUAL	0x1800000000	0xSS and unused data*
Read XACTUAL	0x1800000000	0xSS and XACTUAL
Write VMAX = 0x00ABCDEF	0xA100ABCDEF	0xSS and XACTUAL
Write VMAX = 0x00123456	0xA100123456	0xSS00ABCDEF

*SS is a placeholder for the status bits SPI_STATUS.

SPI Status Bits Transferred with Each Datagram Read Back

New status information is latched at the end of each access and is available with the next SPI transfer. *Table 3* lists these status flags.

Table 3. SPI_STATUS (Status Flags Transmitted with Each SPI Access in Bits 39:32)

BIT	NAME	NOTES
7	STATUS_STOP_R	Refswitch r status
6	STATUS_STOP_L	Refswitch l status
5	POSITION_REACHED	Signals motor target position reached (motion controller only).
4	VELOCITY_REACHED	1: signals mcc velocity VMAX reached (motion controller only).
3	Standstill (STST)	STST flag from DRV_STATUS register.
2	STALLGUARD	STALLGUARD flag from DRV_STATUS register.
1	gstat_error	GSTAT[1] DRV_ERR || GSTAT[2] UV_LDO || GSTAT[4] VS_UVLO || GSTAT[5] VCCIO_RST || GSTAT [6] RREF_ERR || GSTAT [7] VDD_UVLO
0	RESET	GSTAT[0] – 1: Signals a reset occurred (clear by reading GSTAT) (cleared by clearing bit in GSTAT register, write 1 to the bit position).

Data Alignment

All data are right aligned. Some registers represent unsigned (positive) values, some represent integer values (signed) as two's complement numbers. Single bits or groups of bits are represented as single bits, respectively, as integer groups.

SPI Signals

The SPI bus on the TMC5221 has four signals:

- SCK: bus clock input
- SDI: serial data input
- SDO: serial data output
- CSN:chip select input (active low)

The SPI peripheral is enabled for an SPI transaction by a low on-the-chip-select input CSN. Bit transfer is synchronous to the bus clock SCK, with the peripheral latching the data from SDI on the rising edge of SCK and driving data to SDO following the falling edge. The most significant bit is sent first. A minimum of 40 SCK clock cycles is required for a bus transaction with the TMC5221.

If more than 40 clocks are driven, the additional bits shifted into SDI are shifted out on SDO after a 40-clock delay through an internal shift register. This can be used for daisy chaining multiple chips.

The CSN must be low during the whole bus transaction. When CSN goes high, the contents of the internal shift register are latched into the internal control register and recognized as a command from the SPI controller to the SPI peripheral. If more than 40 bits are sent, only the last 40 bits received before the rising edge of CSN are recognized as the command.

SPI Timing

Figure 9 shows the SPI timing diagram. The SPI uses SPI mode 3. The SPI maximum frequency is at 10MHz. SCK is independent from the clock frequency of the system, while the only parameter depending on the clock frequency is the minimum CSN high time. All SPI inputs are internally filtered to avoid triggering on pulses shorter than 10ns. The figure shows the timing parameters of an SPI bus transaction. Timing values are given in the section. In qsc mode, there must be a break between datagrams once the field EXT_CLK switches to the internal slow oscillator.

After the conditions for entering qsc mode are fulfilled, there is a switch in internal clock frequencies within 130µs. It makes sense to not transmit any write data during this duration. After this, the break between datagrams (CSN high time) must be at least 9.5µs - 8 × t_{sck} long.

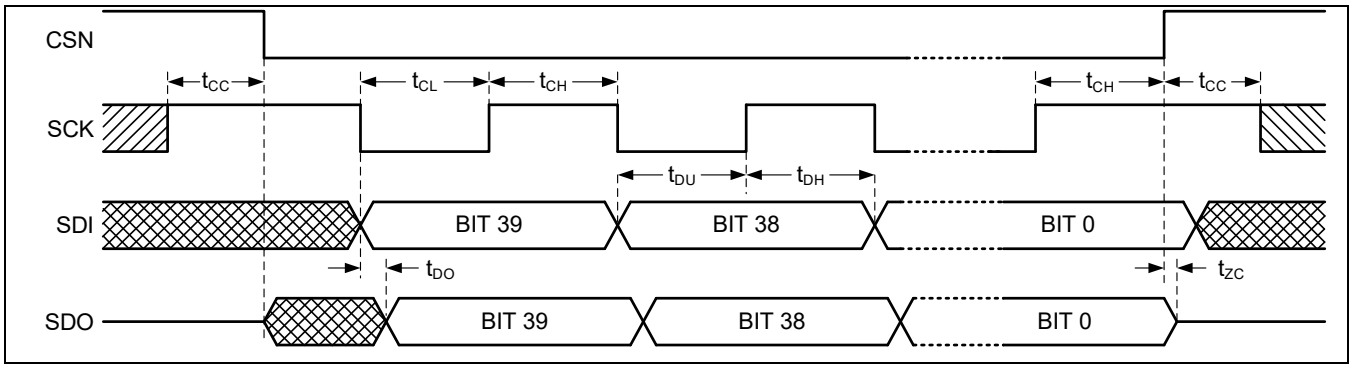


Figure 9. SPI Timing Diagram

I²C Interface (TMC5222 Only)

The TMC5222 integrates a two-wire I²C serial interface, which can operate up to 1MHz.

Device Address

The circuit must be provided with a physical address to discriminate it from the other ones on the serial bus. This address is coded on seven bits (two bits are internally hardwired to '1'), yielding the theoretical possibility of 32 different circuits on the same bus. It is a combination of three OTP memory bits (I2C_NODE_ADDRESS) that default to 0b000 and two hardwired address bit (pin AD0, AD1). AD0/AD1 must either be connected to ground or V_{CCIO}. When AD0/AD1 is not connected and left floating, the correct functionality of the serial interface is not guaranteed.

With AD1 and AD0 tied to GND, and no memory bits burned, the device address defaults to 0b1100000.

AD6	AD5	AD4	AD3	AD2	AD1	AD0
1	1	AD4 (OTP)	AD3 (OTP)	AD2 (OTP)	AD1 (Pin)	AD0 (Pin)

Attention: I²C addresses 11110xx reserved for 10-bit-addressing and 11111xx reserved for future use.

Write an I²C Register

A complete datagram consists of the following:

S	Device Address	'0'	A	Register Address	A	Register Byte (3)	A	Register Byte (2)	A	Register Byte (1)	A	Register Byte (0)	A	P

in which,

S is the I²C start condition.

P is the stop condition.

A is the acknowledge bit.

The acknowledge bit signals the correct reception of the preceding byte to the transmitter. In this case, the TMC5222 pulls the SDA line low. The TMC5222 reads the incoming data at SDA with every rising edge of the SCL line. To finish the transmission, the peripheral must transmit a stop condition. There is the option to automatically increase the address used for register access (I2C_AUT_INC in register NODECONF). In this case, I²C write access to consecutive registers can be chained without intermediate I²C register address write access.

Read an I²C Register

A read sequence consists of two steps. First, the register address must be written with a write operation. Secondary, the read command can be issued. If the first step is omitted, the current register address (namely, the address of the last written register) is assumed for the read operation. There is the option to automatically increase the address used for register access (I2C_AUT_INC in register NODECONF). In this case, I²C read access to consecutive registers can be chained without intermediate I²C register address write access.

S	Device Address	'0'	A	Register Address	A	P

S	Device Address	'1'	A	Register Byte (3)	A	Register Byte (2)	A	Register Byte (1)	A	Register Byte (0)	NA	P

in which,

S is the I²C start condition.

P is the I²C stop condition.

A is ACK.

NA is NACK/No ACK.

Step/Direction Interface

The STEP and DIR inputs provide a simple, standard interface compatible with many existing motion controllers. The MicroPlyer step pulse interpolator brings the smooth motor operation of high-resolution microstepping to applications originally designed for coarser stepping.

To enable the step/direction mode, the bit *SD* in the *GCONF* register must be set to 1. The internal motion controller is then switched off. The only motion controller registers remaining active in this case are the current level settings in the *IHOLD_IRUN* register. In step/direction mode, the reference switch inputs are not available.

Note: In step/direction mode, the motion controller is completely disabled.

Timing

Figure 10 and *Figure 11* show the timing parameters and input filter structure for the STEP and DIR signals. When the dedge mode bit in the *CHOPCONF* register is set, both edges of STEP are active. If *DEDGE* is cleared, only rising edges are active. STEP and DIR are sampled and synchronized to the system clock. An internal analog filter of approximately 10ns removes glitches on the signals, such as those caused by long PCB traces. If the signal source is far from the chip, and especially if the signals are carried on cables, the signals should be filtered or transmitted differentially.

See the **Electrical Characteristics** section for the specified timing parameters.

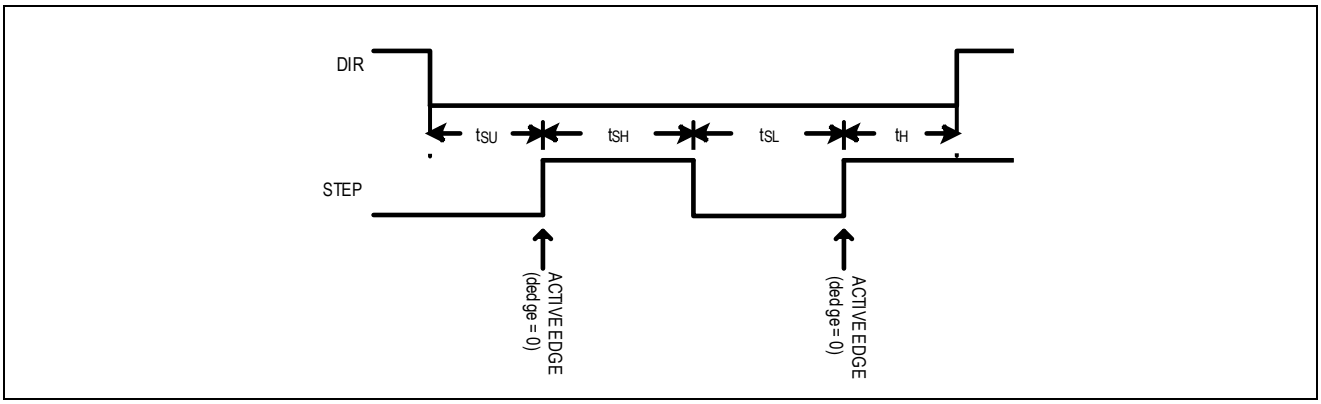


Figure 10. STEP/DIR Signal Timing

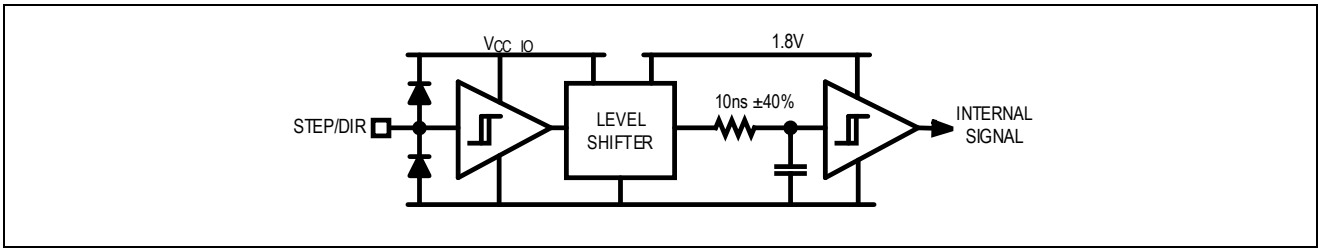


Figure 11. STEP/DIR Signal Input Filter Structure

Changing Resolution

A reduced microstep resolution allows the limitation of the step frequency for the step and direction interface, or compatibility to an older, less performing driver. The internal microstep table with 1024 sine-wave entries generates sinusoidal motor coil currents. These 1024 entries correspond to one electrical revolution or four full steps. The microstep-resolution setting determines the step width taken within the table. Depending on the DIR input, the microstep counter is increased (DIR = 0) or decreased (DIR = 1) with each step pulse by the step width. The microstep resolution determines the increment or decrement. At maximum resolution, the sequencer advances one step for each step pulse. At half resolution, it advances two steps. The increment is up to 256 steps for full stepping. The sequencer has a special provision to allow seamless switching between different microstep rates at any time. When switching to a lower microstep resolution, it calculates the nearest step within the target resolution and reads the current vector at that position. This behavior is especially important for low resolutions like full step and half step because any failure in the step sequence leads to an asymmetrical run when comparing a motor running clockwise and counterclockwise. *Table 4* shows the coil current values for half stepping and full stepping.

Examples:

Full step: Cycles through table positions: 128, 384, 640, and 896 (45°, 135°, 225°, and 315° electrical position, both coils on at identical current). The coil current in each position corresponds to the RMS value (0.71 × amplitude). The step size is 256 (90° electrical).

Half step: The first table position is 64 (22.5° electrical). The step size is 128 (45° steps).

Quarter step: The first table position is 32 (90°/8 = 11.25° electrical). The step size is 64 (22.5° steps). This way, equidistant steps result and they are identical in both rotation directions. Some older drivers also use zero current (table entry 0, 0°) as well as full current (90°) within the step tables. This kind of stepping is avoided because it provides less torque and has a worse power dissipation in driver and motor.

Table 4. Full-Step/Half-Step Lookup Table Values for Phase A/B Coil Currents

STEP POSITION	TABLE POSITION	CURRENT COIL A	CURRENT COIL B
Half step 0	64	38.3%	92.4%
Full step 0	128	70.7%	70.7%
Half step 1	192	92.4%	38.3%
Half step 2	320	92.4%	-38.3%
Full step 1	384	70.7%	-70.7%
Half step 3	448	38.3%	-92.4%
Half step 4	576	-38.3%	-92.4%
Full step 2	640	-70.7%	-70.7%
Half step 5	704	-92.4%	-38.3%
Half step 6	832	-92.4%	38.3%
Full step 3	896	-70.7%	70.7%
Half step 7	960	-38.3%	92.4%

MicroPlyer Step Interpolator and Standstill Detection

For each active edge on the STEP input, MicroPlyer produces microsteps at 256x resolution. It interpolates the time between the two step impulses at the step input based on the last step interval. This way, from 2 microsteps (128 microsteps to 256 microsteps interpolation) up to 256 microsteps (full step input to 256 microsteps) are driven for a single step pulse.

The MicroPlyer function is enabled by setting the bit *INTPOL* in the register *CHOPCONF*.

The step rate for the interpolated 2 microsteps to 256 microsteps is determined by measuring the time interval of the previous step period and dividing it into up to 256 equal parts. The maximum time between 2 microsteps corresponds to the configured standstill time in STANDSTILL_TIME (default: 2^{20}, which is approximately one million system clock cycles), for an even distribution of 256 microsteps. At 12.5MHz system clock frequency, this results in a minimum step input frequency of 12.5Hz for MicroPlyer operation. A lower step rate causes the *STST* bit to be set, which indicates a standstill event. At this frequency, microsteps occur at a rate of $f\_clk/2^{16}$ ~ 191Hz (with internal oscillator). When a standstill is detected, the driver automatically switches the motor to holding current *IHOLD*.

Note: MicroPlyer only works perfectly with a stable step frequency. Do not use the *DEDGE* option if the step signal at the STEP pin does not have a 50% duty cycle.

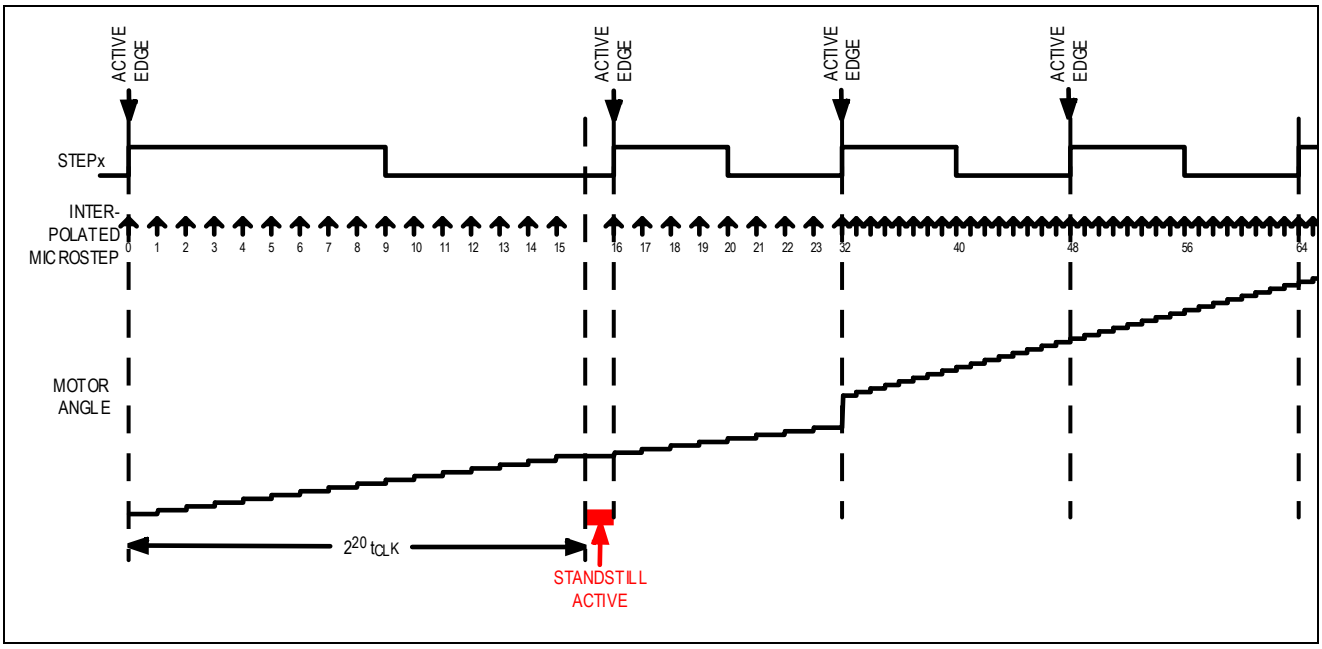


Figure 12. MicroPlyer Microstep Interpolation with Rising Step Signal Frequency (Example: 16 to 256)

In *Figure 13*, the first STEP signal cycle is long enough to set the standstill bit *STST*. This bit is cleared on the next active STEP edge. Then, the external frequency of the STEP signal increases. After one cycle at the higher rate, MicroPlyer adapts the interpolated microstep rate to the higher frequency. During the last cycle at the slower rate, MicroPlyer does not generate all 16 microsteps. So, there is a small jump in motor angle between the first and second cycles at the higher rate.

Note: The TMC5221/TMC5222 provide a configurable standstill detection timer length. *STANDSTILL_TIME* in the register *DRV_CONF* allows configuration with eight different options.

OPERATION DETAILS

This section details the basic steps to parameterize the driver fitting for the motor, usage of sensorless features, and shows the options for motion control and application-specific adaptations. Once these basic features are familiar, the flow charts for basic parameterization tasks provide a guideline.

Motor Current Control

Integrated Current Sensing (ICS)

Non-dissipative current sensing is integrated in the TMC5221/TMC5222. This feature eliminates the bulky external power resistors, which are normally required with external current sensing. The ICS results in dramatic space and power saving compared to conventional architectures based on external sense resistors. For optimum performance, the ICS individually measures $R_{DS(ON)}$ for each of the power MOSFETs, taking into account the individual MOSFET temperature to yield the best results.

Setting the Motor Current

Table 5 shows the parameters related to the motor current setting in the TMC5221/TMC5222.

Table 5. Parameters Controlling the Motor Current

PARAMETER	DESCRIPTION	SETTING	NOTES
IRUN	Current scale when motor is running. Scales coil current values as taken from the internal sine-wave table. For high-precision motor operation, work with a current scaling factor in the range 16 to 31 because digitally scaling down the current values reduces the effective microstep resolution by making the microsteps coarser. This setting also controls the maximum current value set by CoolStep. Default = 31.	0 to 31	Scaling factor (IRUN + 1)/32 for IRUN and (IHOLD + 1)/32 for IHOLD Hint: Do not use values below 7 in combination with StealthChop, as automatic current regulations require a minimum CS_ACTUAL of 7 during motor operation. With CoolStep active, therefore, go for at least 16, respectivel, 31
IHOLD	Identical to IRUN, but for motor in standstill. IHOLD typically is much lower than IRUN and can be scaled down to the minimum value required to maintain motor position. Default = 8.		
IHOLDDELAY	Allows smooth current reduction from run current to hold current. IHOLDDELAY controls the number of clock cycles for motor power-down after TZEROWAIT in increments of 2^{18} clocks. 0 = Instant power-down. 1 to 15: Current reduction delay per current step in multiple of 2^{18} clocks. Example: When using IRUN = 31 and IHOLD = 16, 15 current steps are required for hold current reduction. An IHOLDDELAY setting of 4, thus, results in a power-down time of $4 \times 15 \times 2^{18}$ clock cycles, example, approximately 1.25 seconds at 12.5MHz. Default = 1.	0	Instant power-down to IHOLD.
		1 to 15	IHOLDDELAY × 2^{18} Tclk per current decrement.
IRUNDELAY		0	Instant power-up to IRUN

	Controls the number of clock cycles for motor power-up after start is detected. Allows smooth current increment upon start of a motion from hold current (IHOLD) to run current (IRUN). While a quick power-up is important to establish full motor torque, a small delay time helps to reduce acoustic noise and avoids a peak current on the power supply resulting from energy required to build up the motor's magnet field. Default = 4.	1 to 15	IRUNDELAY × 512 Tclk per current increment.
TPOWERDOWN	TPOWERDOWN sets the delay time after standstill (stst) of the motor-to-motor current power-down. The time range is about 0 seconds to 4 seconds. Note: A minimum setting of 2 is required to allow the automatic tuning of StealthChop2 PWM_OFFS_AUTO. Default = 10.	0 to 255	TPOWERDOWN × 2^{18} Tclk
FS_GAIN	FS_GAIN sets the GAIN of the current sense amplifier in front of the DAC. Use this together with FS_SENSE for fine-tuning the current scaling.	0 to 3	See *IFS Full-Scale Range Settings* for further details.
FS_SENSE	FS_SENSE sets the output stage (SENSE-FET) resistance to allow for a certain voltage drop that gives the best measurement range. This is the basic adaptation of the drivers $R_{DS(ON)}$ current sensing to the motor current range. Select the lowest fitting range for best current precision. These bits also define the overcurrent protection threshold (this value fits the peak current setting).	0 to 3	See *IFS Full-Scale Range Settings* for further details.
GLOBALSCALER_A	Global scaling of motor current. This value is multiplied to the current scaling to fine-tune and adapt to a certain motor. Scales Phase A of the motor to allow the fine-tuning of the motor current. Tip: Values > 128 are recommended for best results. This value should be chosen before tuning the other settings because it also influences chopper hysteresis. For normal operation, set identical to GLOBALSCALER_B. Differing values can only be used with SpreadCycle.	0 to 255	0 = Full-scale (or write 256). 1 to 31 = Not allowed for operation. 32 to 255 = 32/256 to 255/256 of full-scale current.
GLOBALSCALER_B	Global scaling of motor current. This value is multiplied to the current scaling to fine-tune and adapt to a certain motor.	0 to 255	0 = Full-scale (or write 256). 1 to 31 = Not allowed for

	Scales Phase B of motor to allow the fine-tuning of the motor current. Tip: Values > 128 are recommended for best results. This value should be chosen before tuning the other settings, because it also influences chopper hysteresis. For normal operation, set identical to GLOBALSCALER_A. Differing values can only be used with SpreadCycle.		operation. 32 to 255 = 32/256 to 255/256 of full-scale current.

Setting the Full-Scale Current

The full-scale current IFS is a maximum absolute current level setting and is defined by a reference resistor R_{REF} and two parameters in the DRV_CONF register: FS_GAIN and FS_SENSE. Adapting IFS is needed to benefit from a best possible current control resolution. Up to 16 different full-scale current ranges can be configured to adapt to different motor sizes and applications.

The reference resistor R_{REF} can be either internal or external depending on the S_IREF_INTCUR_EN setting.

For very precise absolute current control, an external reference resistor is recommended. In this case, a 60.4KΩ R_{REF} resistor with better than 1% accuracy must be connected between the pin I_{REF} and GND. If the internal reference is used, connect the pin I_{REF} to GND. Accuracy data are shown in both the cases in the respective table.

The two 2-bit parameters, FS_SENSE and FS_GAIN, in the DRV_CONF register define the typical on-resistance of the driver low-side FET and the gain of the current-sense amplifier, allowing to find the optimal setting depending on the desired current range.

The following equations show the full-scale current IFS as a function of the RREF resistor and the DRV_CONF parameter settings.

The total proportionality constant KIFS is the product of the K_{IFSSNS} and $K_{IFSGAIN}$, which depends on the FS_SENSE and FS_GAIN parameters, as shown in *Table 6*.

$$I_{FS\_RMS} = \frac{(K_{IFSSNS}(kV) \times K_{IFSGAIN})}{R_{REF}(k\Omega)}$$

$$I_{FS\_PEAK} = \frac{(K_{IFSSNS}(kV) \times K_{IFSGAIN})}{R_{REF}(k\Omega)} \times \sqrt{2}$$

Notes:

- Some of the combinations of FS_SENSE and FS_GAIN result in similar full-scale ranges. It is always better to choose FS_GAIN as large as possible for best current control accuracy.
- For optimal performance of the current regulation and proper current levels, the TMC5221/TMC5222 require a symmetrical board layout.

Table 6. IFS Full-Scale Range Settings

REGISTER CONFIGURATION		$K_{IFSGAIN}$	K_{IFSSNS} (kV)	IFS_RMS (A)	IFS_PEAK (A)	TYPICAL LOW SIDE R_{DSON} (mΩ)
FS_SENSE [1:0]	FS_GAIN [1:0]					
11	11	1.0	85,4	1.41	2	50
	10	0.75		1.06	1.5	
	01	0.5		0.71	1	
	00 (Default)	0.25		0.35	0.5	
10	11	1.0	64,1	1.06	1.50	67
	10	0.75		0.8	1.13	
	01	0.5		0.53	0.75	
	00 (Default)	0.25		0.27	0.38	
01	11	1.0	42,7	0.71	1	100
	10	0.75		0.53	0.75	
	01	0.5		0.35	0.5	
	00 (Default)	0.25		0.18	0.25	
00 (default)	11	1.0	21,4	0.35	0.5	200
	10	0.75		0.27	0.38	
	01	0.5		0.18	0.25	
	00 (Default)	0.25		0.09	0.13	

StealthChop2

StealthChop2 is an extremely quiet mode of operation for stepper motors. It is based on a voltage-mode PWM. When at standstill and at low velocities, the motor is absolutely noiseless. Thus, StealthChop2-operated stepper motor applications are very suitable for office or home use. The motor operates absolutely free of vibration at low velocities. With StealthChop, the motor current is applied by driving a certain effective voltage into the coil, using a voltage-mode PWM. With the enhanced StealthChop2, the driver automatically adapts to the application parameters for best performance. No more configurations are required. Optional configuration allows for tuning the setting in special cases, or for setting initial values for the automatic adaptation algorithm. For high-velocity drives, SpreadCycle should be considered in combination with StealthChop2.

Operate the motor within the application when exploring StealthChop2. Motor performance often is better with a mechanical load because it prevents the motor from stalling due to mechanical oscillations, which can occur without load.

Automatic Tuning

StealthChop2 integrates an automatic tuning (AT) procedure, which adapts the most important operating parameters to the motor automatically. This way, StealthChop2 allows high motor dynamics and supports powering down the motor to very low currents. Just two steps must be considered for best results (see *Table 7*). Start with the motor in standstill, but powered with nominal run current (AT#1). Move the motor at a medium velocity, example, as part of a homing procedure (AT#2). The flowchart in *Figure 13* shows the tuning procedure.

Table 7. Constraints and Requirements for StealthChop2 Autotuning AT#1 and AT#2

STEP	PARAMETER	CONDITIONS	REQUIRED DURATION
AT#1	PWM_ OFS_AUTO	• Motor in standstill and actual current scale (CS) is identical to run current (IRUN). • If standstill reduction is enabled, an initial step pulse switches the drive back to run current or sets IHOLD to IRUN. • V_S pin at operating level.	≤ 2^{20} + 2 × 2^{18} tCLK, ≤ 130ms (with internal clock).
AT#2	PWM_ GRAD_AUTO	• Move motor at a velocity where a significant amount of back EMF is generated and where the full run current can be reached. • 1.5 × PWM_OFS_AUTO × (IRUN + 1)/32 < (PWM_SCALE_SUM/4) < 4 × PWM_OFS_AUTO × (IRUN + 1)/32 • PWM_SCALE_SUM < 1023 **Tip:** A typical range is 60RPM to 300RPM for a standard 50-pole stepper motor.	Eight full steps are required for a change of ±1. For a typical motor with PWM_GRAD_AUTO optimum at 50 or less, up to 400 full steps are required when starting from default value 0.

Tip: Determine the best conditions for automatic tuning with the TMC5221/TMC5222-EVKIT. Use application-specific parameters for PWM_GRAD and PWM_OFS for initialization in firmware or preload registers with OTP to provide the initial tuning parameters. Monitor PWM_SCALE_AUTO going down to zero during the constant velocity phase in AT#2 tuning. This indicates a successful tuning. The **Velocity-Based Scaling** section gives an overview on how these parameters can be calculated based on motor parameters.

Caution: Operating in StealthChop2 without proper tuning can lead to high motor currents during a deceleration ramp, especially with low resistive motors and fast deceleration settings. Follow the automatic tuning process and check the optimum tuning conditions using the TMC5221/TMC5222-EVKIT. It is recommended to use an initial value for the PWM_OFS and PWM_GRAD settings determined per motor type.

Modifying the current range using FS_GAIN, FS_SENSE, GLOBALSCALER_A/B, or V_S voltage invalidates the result of the automatic tuning process. Motor current regulation cannot compensate the significant changes until the next AT#1 phase. Automatic tuning adapts to changed conditions whenever the AT#1 and AT#2 conditions are fulfilled in the later operation.

As automatic tuning determines an offset for minimum PWM duty cycle to reach the target current, it cannot properly react to a rising supply voltage during operation. Therefore, significant rise of the supply voltage in the application should be avoided. It adapts with the beginning of the next standstill phase. Other than that, a falling supply voltage is compensated by regulation at any time.

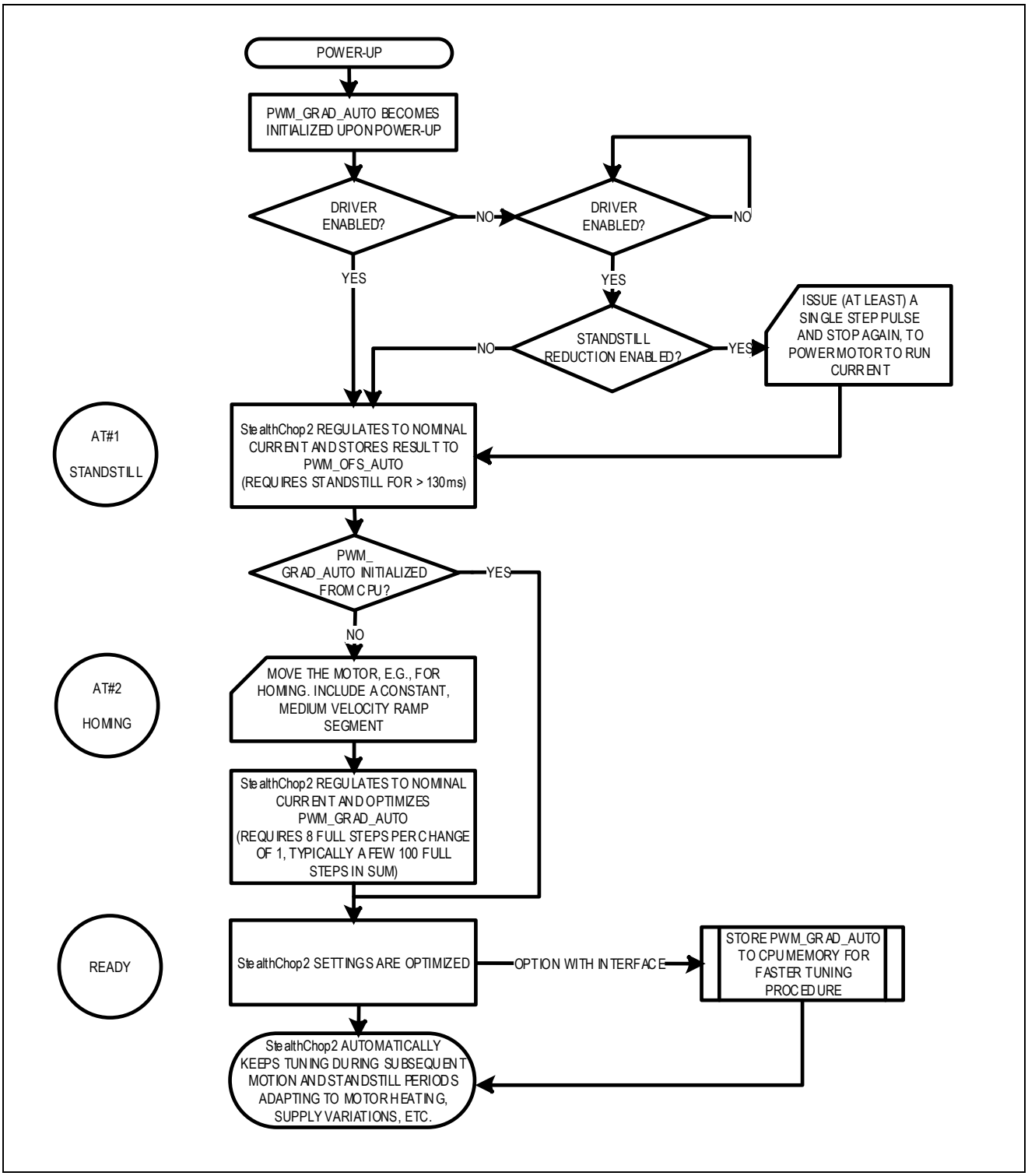


Figure 13. StealthChop2 Automatic Tuning Procedure

StealthChop2 Options

To match the motor current to a certain level, the effective PWM voltage is scaled depending on the actual motor velocity. Several additional factors influence the required voltage level to drive the motor at the target current; the motor resistance, its back-EMF (example, directly proportional to its velocity), as well as the actual level of the supply voltage. Two modes

of PWM regulation are provided: automatic tuning mode (AT) using current feedback (*PWM_AUTOSCALE* = 1, *PWM_AUTOGRAD* = 1) and a feed-forward velocity-controlled mode (*PWM_AUTOSCALE* = 0). The feed-forward velocity-controlled mode does not react to a change of the supply voltage or to events like a motor stall, but it provides very stable amplitude. It does not use nor require any means of current measurement. This is perfect when the motor type and supply voltage are well known. Therefore, the automatic mode is recommended, unless current regulation is not satisfying in the given operating conditions.

It is recommended to use application-specific initial tuning parameters, fitting the motor type and supply voltage. Additionally, operate in the automatic tuning mode to respond to a parameter change, example, due to motor heat-up or change of supply voltage.

Non-automatic mode (*PWM_AUTOSCALE* = 0) should be taken into account only with well-known motor and operating conditions. In this case, careful programming through the interface is required. The operating parameters PWM_GRAD and PWM_OFS can be determined in the automatic tuning mode initially.

The StealthChop2 PWM frequency can be chosen in four steps to adapt the frequency divider to the frequency of the clock source. A setting in the range of 20kHz to 50kHz is good for most applications. It balances low current ripple and good higher velocity performance vs. dynamic power dissipation. *Table 8* provides the example frequencies and settings, as well as the recommended values.

Table 8. Choice of PWM Frequency for StealthChop2

CLOCK FREQUENCY fCLK (MHz)	PWM_FREQ = 00 fPWM = 2/1024 fCLK (kHz)	PWM_FREQ = 01 fPWM = 2/683 fCLK (kHz)	PWM_FREQ = 10 fPWM = 2/512 fCLK (kHz)	PWM_FREQ = 11 fPWM = 2/410 fCLK (kHz)
20	39.1*	58.1	78.1	97.6
18	35.2*	52.7	70.3	87.8
16	31.3*	46.9*	62.5	78.0
12.5 (Internal)	24.4*	36.6*	48.8*	61.0
12	23.5*	36.1*	46.9*	58.4
8	15.7	25.3*	31.3*	38.8*

*Recommended values.

StealthChop2 Current Regulator

In StealthChop2 voltage PWM mode, the autoscaling function (*PWM_AUTOSCALE* = 1, *PWM_AUTOGRAD* = 1) regulates the motor current to the desired current setting. Automatic scaling is used as part of the AT process, and for subsequent tracking of changes within the motor parameters. The driver measures the motor current during the chopper on-time and uses a proportional regulator to regulate PWM_SCALE_AUTO to match the motor current to the target current. PWM_REG is the proportionality coefficient for this regulator. Basically, the proportionality coefficient should be as small as possible to get a stable and soft regulation behavior, but it must be large enough to allow the driver to quickly react to changes caused by the variation of the motor target current. During the initial tuning step AT#2, PWM_REG also compensates for the change of motor velocity. Therefore, a high acceleration during AT#2 requires a higher setting of PWM_REG. With careful selection of homing velocity and acceleration, a minimum setting of the regulation gradient often is sufficient (PWM_REG = 1). The PWM_REG setting should be optimized for the fastest required acceleration and deceleration ramp. *Figure 14* and *Figure 15* show examples of motor phase currents for a good setting and setting too small for PWM_REG.

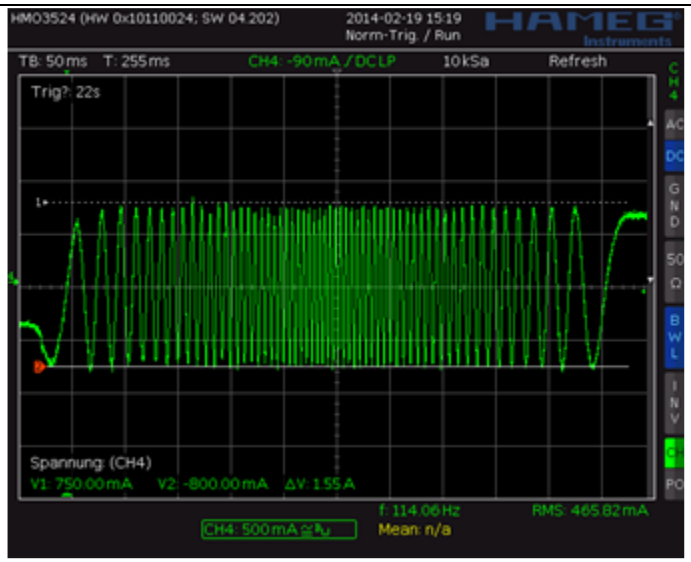


Figure 14. StealthChop2: Good Setting for PWM_REG

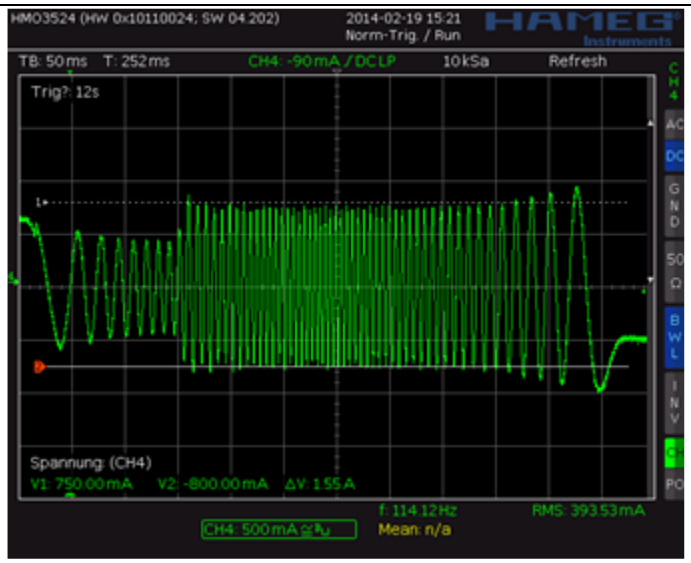


Figure 15. StealthChop2: Setting Too Small for PWM_REG During AT#2

The quality of the PWM_REG setting in phase AT#2 and the finished automatic tuning procedure (or non-automatic settings for PWM_OFS and PWM_GRAD) can be examined when monitoring the motor current during an acceleration phase, as shown in *Figure 16* and *Figure 17*.

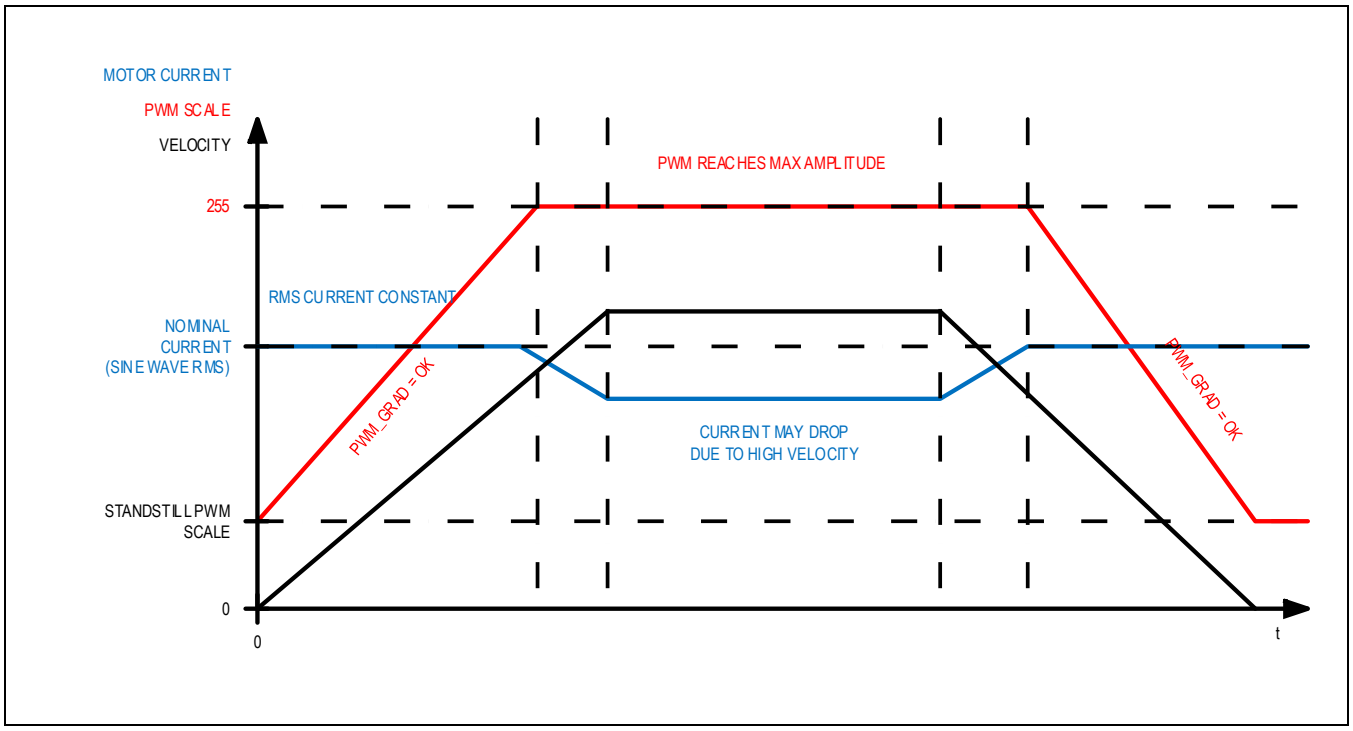


Figure 16. Successfully Determined PWM_GRAD(_AUTO) and PWM_OFS(_AUTO)

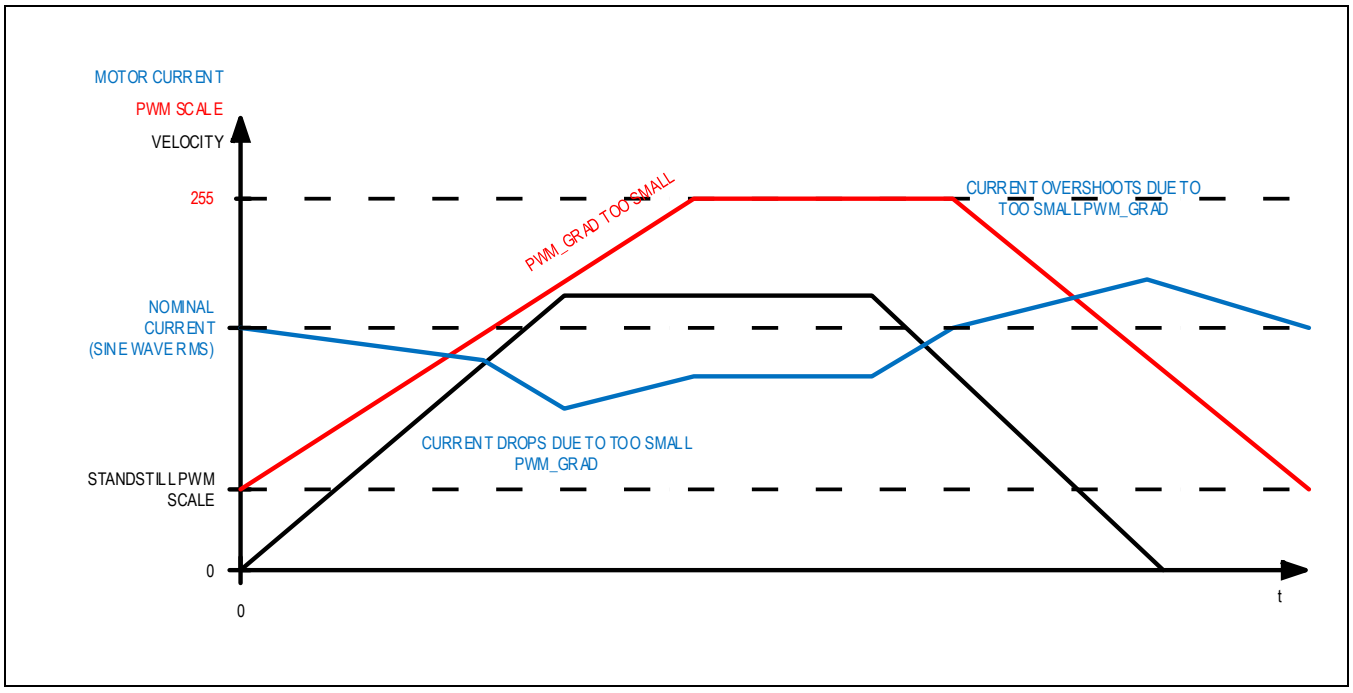


Figure 17. Example of Setting Too Small for PWM_GRAD

Lower Current Limit

Depending on the setting of PWM_MEAS_SD_ENABLE, the StealthChop2 current regulator principle imposes a lower limit for motor current regulation. As the coil current is measured during the chopper on-phase only (PWM_MEAS_SD_ENABLE = 0), a minimum chopper duty cycle allowing the coil current regulation is given by the blank time as set by TBL and the chopper frequency setting. Therefore, the motor-specific minimum coil current in the

StealthChop2 autoscaling mode rises with the supply voltage and chopper frequency. A lower blanking time allows a lower current limit. It is important for the correct determination of PWM_OFS_AUTO, that in AT#1, the run current, GLOBALSCALER_A/B, and IRUN is well within the regulation range. Lower currents (example, for standstill power down) are automatically realized based on PWM_OFS_AUTO and PWM_GRAD_AUTO based on PWM_OFS and PWM_GRAD, respectively, with non-automatic current scaling. The freewheeling option allows going to zero motor current.

The lower motor coil current limit for StealthChop2 automatic tuning (PWM_MEAS_SD_ENABLE = 0) is:

$$I_{LOWERLIMIT} = t_{BLANK} \times f_{PWM} \times \frac{V_S}{R_{COIL}}$$

where, V_S is the motor supply voltage and R_{COIL} is the motor coil resistance.

$I_{LOWERLIMIT}$ can be treated as a rule-of-thumb value for the minimum nominal IRUN motor current setting. When the lower limit is not sufficient to reach the desired setting, be sure to set PWM_MEAS_SD_ENABLE = 1. fPWM is the chopper frequency, as determined by the PWM_FREQ setting.

Example: A motor has a coil resistance of 5Ω and the supply voltage is 16V. With TBL = 0b01 and PWM_FREQ = 0b00, t_{BLANK} is 24 clock cycles and fPWM is 2/(1024 clock cycles):

$$I_{LOWERLIMIT} = 24t_{CLK} \times \frac{2}{1024t_{CLK}} \times \frac{16V}{5\Omega} = \frac{24}{512} \times \frac{16V}{5\Omega} = 150mA$$

This means the motor target current for automatic tuning must be 150mA or more, taking into account all relevant settings. This lower current limit also applies for the modification of the motor current by the GLOBALSCALER_A/B.

Note: For automatic tuning, a lower coil current limit applies.

IRUN ≥ 8: Current settings for IRUN below 8 do not work with automatic tuning.

$I_{LOWERLIMIT}$: Depending on the setting of bit PWM_MEAS_SD_ENABLE (in register PWM_CONF[22]) for automatic tuning, a lower coil current limit applies. The motor current in the automatic tuning phase AT#1 must exceed this lower limit. Calculate $I_{LOWERLIMIT}$ or measure it using a current probe. Setting the motor run-current or hold-current below the lower current limit during operation by modifying IRUN and IHOLD is possible after successful automatic tuning. The lower current limit also limits the capability of the driver to respond to the changes of GLOBALSCALER_A/B.

The lower current limit also limits the capability of the driver to respond to the changes of GLOBALSCALER_A/B.

To overcome the lower limit, set PWM_MEAS_SD_ENABLE = 1. This allows the IC to additionally measure coil current in the slow decay phase.

Velocity-Based Scaling

For very precise, even motor motion, the current regulation can be switched off using purely velocity-based feed-forward control. *Figure 18* shows the principle.

Velocity-based scaling is an optional mode and scales the StealthChop2 amplitude based on the time between every two steps, example, based on TSTEP, measured in clock cycles. Thus, the current is scaled based only on the velocity.

This function is available when setting PWM_AUTOSCALE = 0. The basic idea is to have a linear approximation of the voltage required to drive the target current into the motor. The stepper motor has a certain coil resistance, and thus, needs a certain voltage amplitude to yield a target current based on the basic formula I = U/R, with R being the coil resistance and U being the supply voltage scaled by the PWM value.

The initial value for PWM_OFS can be calculated by:

$$PWM\_OFS = \frac{374 \times R_{COIL} \times I_{COIL}}{V_S}$$

where, V_S is the motor supply voltage and I_{COIL} is the target RMS current.

The effective PWM voltage UPWM (1/SQRT(2) × peak value) results, considering the 10-bit resolution and 248 sine-wave peak for the actual PWM amplitude shown as PWM_SCALE_SUM:

$$U_{PWM} = V_S \times \frac{PWM\_SCALE\_SUM}{256} \times \frac{248}{256} \times \frac{1}{\sqrt{2}} = V_S \times \frac{PWM\_SCALE\_SUM}{374}$$

With rising motor velocity, the motor generates an increasing back-EMF voltage. The back-EMF voltage is proportional to the motor velocity. It reduces the PWM voltage effective at the coil resistance, and thus, current decreases. The TMC5221/TMC5222 provide a second velocity dependent factor (PWM_GRAD) to compensate for this. The overall effective PWM amplitude (PWM_SCALE_SUM) in this mode is automatically calculated, independent of the microstep frequency, as:

$$\text{PWM\_SCALE\_SUM} = \text{PWM\_OFS} \times \left(\frac{\text{CS\_ACTUAL} + 1}{32}\right) + \text{PWM\_GRAD} \times \frac{256}{\text{TSTEP}}$$

CS_ACTUAL takes into account the actual current scaling, as defined by IHOLD and IRUN, or by CoolStep.

TSTEP represents the microstep frequency for the 256-microstep resolution equivalent and fCLK is the clock frequency supplied to the driver or the actual internal frequency.

As a first approximation, the back-EMF subtracts from the supply voltage, and thus, the effective current amplitude decreases. This way, a first approximation for the PWM_GRAD setting can be calculated as follows:

$$\text{PWM\_GRAD} = C_{BEMF} \left| \frac{V}{\frac{rad}{s}} \right| \times 2\pi \times \frac{f_{clk} \times 1.46}{V_S \times \text{MSPR}}$$

where, CBEMF is the back-EMF constant of the motor in volts per radian/second.

MSPR is the number of microsteps per rotation related to the 1/256 microstep resolution, example, 51200 = 256 microsteps multiplied by 200 full steps for a 1.8° motor.

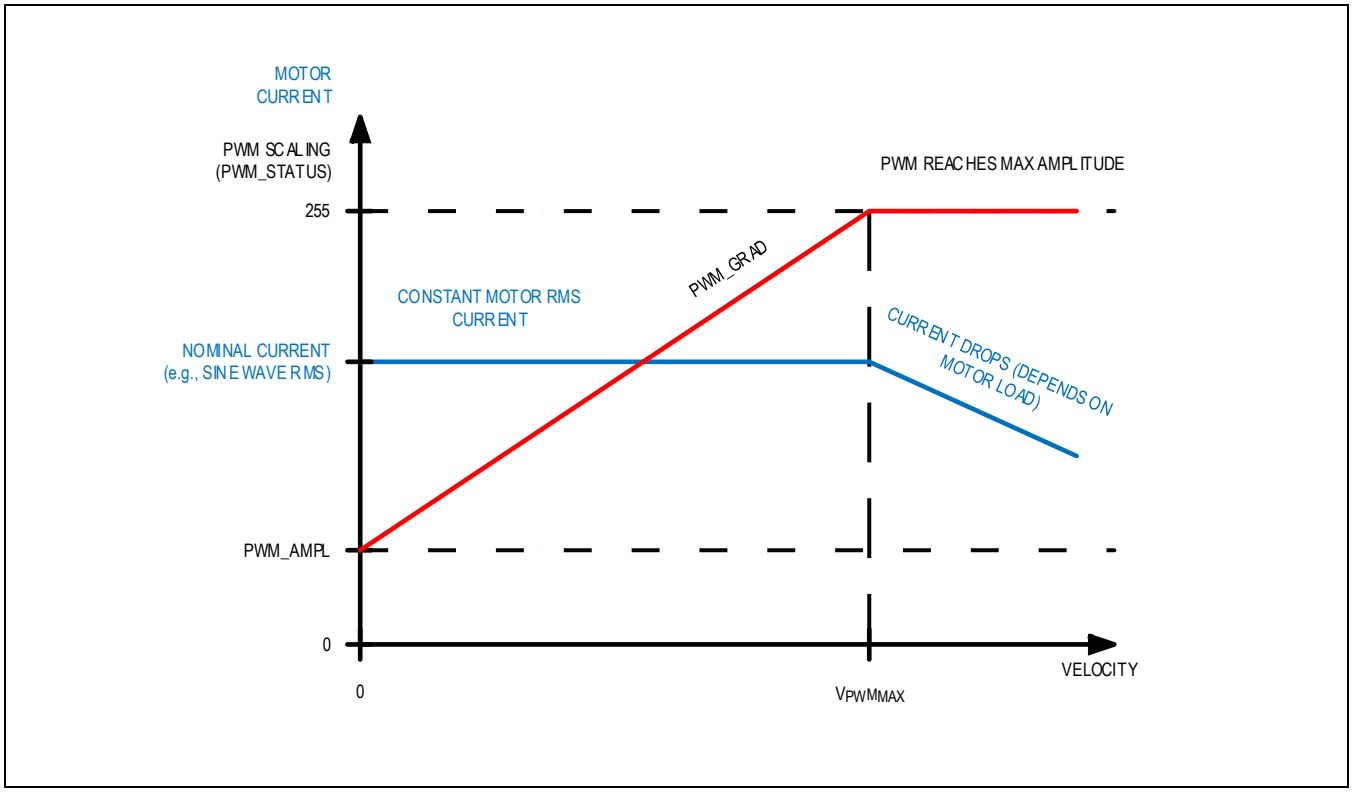


Figure 18. Velocity-Based PWM Scaling (PWM_AUTOSCALE = 0)

The values for PWM_OFS and PWM_GRAD can easily be optimized by tracing the motor current with a current probe on the oscilloscope. Alternatively, automatic tuning determines these values and they can be read out from PWM_OFS_AUTO and PWM_GRAD_AUTO.

Understanding the Back-EMF (BEMF) Constant of a Motor

The BEMF constant is the voltage a motor generates when turned with a certain velocity. Often motor data sheets do not specify this value because it can be deducted from the motor torque and coil current rating. Within SI units, the numeric

value of the BEMF constant C_{BEMF} has the same numeric value as the numeric value of the torque constant. For example, a motor with a torque constant of 1Nm/A would have a C_{BEMF} of 1V/rad/s. Turning such a motor with 1rps (1rps = 1 revolution per second = 6.28rad/s) generates a BEMF voltage of 6.28V. Thus, the BEMF constant can be calculated as:

$$C_{BEMF}\left|\frac{V}{\frac{rad}{s}}\right| = \frac{\text{Holding TORQUE[Nm]}}{2 \times I_{COILNOM}[A]}$$

where, $I_{COILNOM}$ is the motor's rated RMS phase current for the specified holding torque.

Holding torque is the motor-specific holding torque, example, the torque reached at $I_{COILNOM}$ on both coils. The torque unit is [Nm], where 1Nm = 100Ncm = 1000mNm. The BEMF voltage is valid as RMS voltage per coil, thus, the nominal current is multiplied by 2 in this formula, since the nominal current assumes a full step position, with two coils operating.

Combining StealthChop2 and SpreadCycle

For applications requiring high-velocity motion, SpreadCycle can bring more stable operation in the upper velocity range. To combine no-noise operation with highest dynamic performance, the TMC5221/TMC5222 allow combining StealthChop2 and SpreadCycle based on a velocity threshold. With this, StealthChop2 is only active at low velocities, as shown in *Figure 19*.

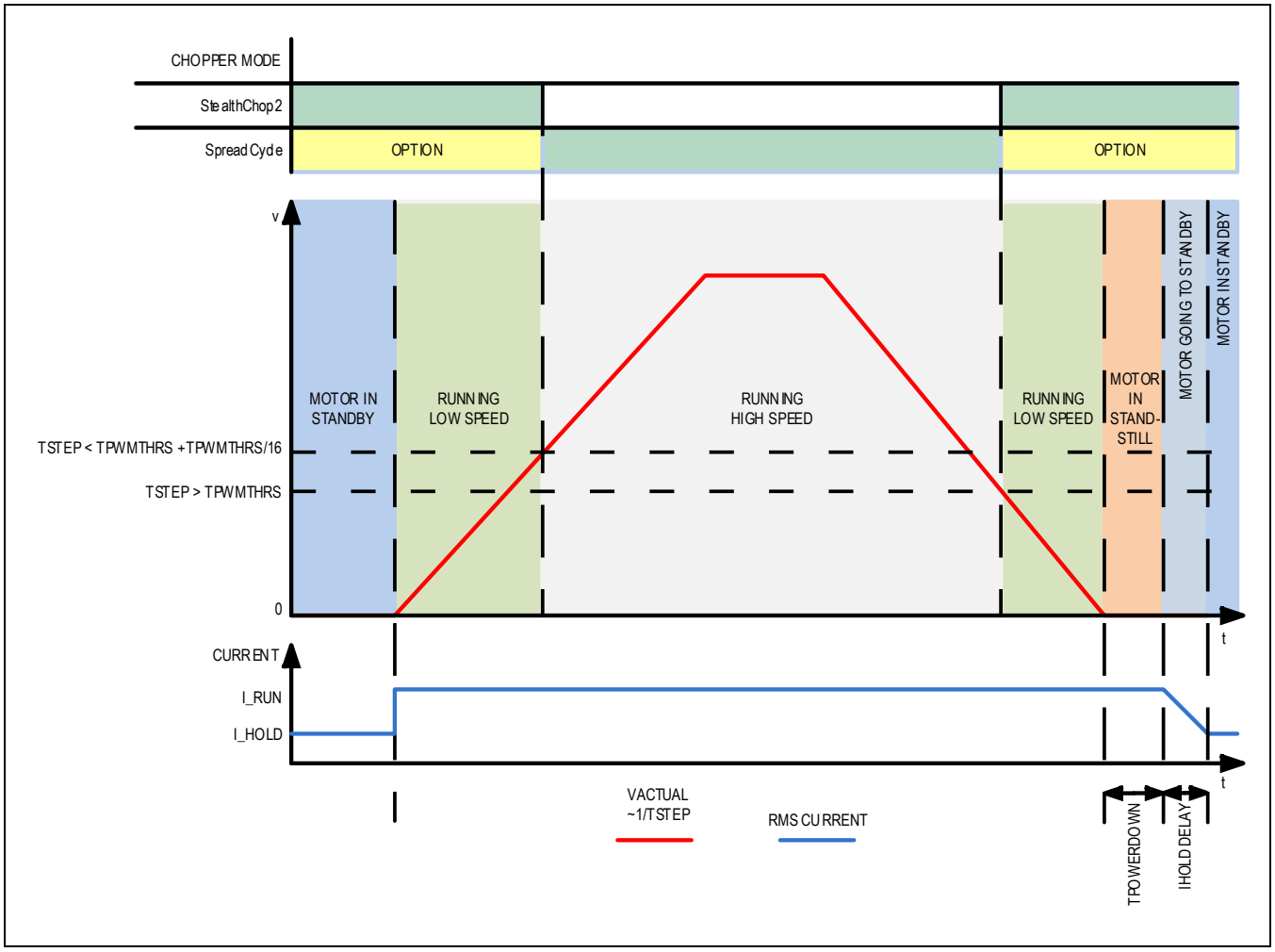


Figure 19. TPWMTHRS for Optional Switching to SpreadCycle

As a first step, both chopper principles StealthChop2 and SpeadCycle should be parameterized and optimized individually (see the *StealthChop2*, and *SpreadCycle and Classic Chopper* sections).

The next step is to define the switchover velocity. For example, StealthChop2 operation is used for precise, low-speed positioning, while SpreadCycle is used for highly dynamic motion. TPWMTHRS determines this transition velocity. Read out TSTEP when moving at the desired velocity and program the resulting value to TPWMTHRS. Use a low transfer velocity to avoid a jerk at the switching point.

Jerkless Switching to SpreadCycle

Without automatic compensation, a jerk occurs when switching at higher velocities, because the back-EMF of the motor (which rises with the velocity) causes a phase shift of up to 90° between the motor voltage and motor current. Therefore, when switching at higher velocities between voltage PWM and current PWM mode, this jerk occurs with increased intensity. A high jerk may even produce a temporary overcurrent condition (depending on the motor coil resistance). At low velocities (example, 1RPM to a few 10RPM), it can be completely neglected for most motors. With automatic switching controlled by TPWMTHRS, the driver can automatically eliminate the jerk using StallGuard4 to determine the phase shift. It applies the same phase shift to SpreadCycle until the velocity falls back below the switching threshold.

Set TPWMTHRS to 0 to work with StealthChop2 only.

When enabling the StealthChop2 mode the first time using automatic current regulation, the motor must be at standstill to allow a proper current regulation. When the drive switches to SpreadCycle at a higher velocity, StealthChop2 logic stores the last current regulation setting until the motor returns to a lower velocity again. This way, the regulation has a known starting point when returning to a lower velocity, where StealthChop2 is re-enabled. Therefore, neither the velocity threshold nor the supply voltage must be considerably changed during the phase while the chopper is switched to a different mode, because otherwise, the motor can lose steps, or the instantaneous current might be too high or too low.

A motor stall or a sudden change in the motor velocity may lead to the driver detecting a short circuit or it may lead to a state of automatic current regulation, from which it cannot recover. Clear the error flags and restart the motor from zero velocity to recover from this situation.

Start the motor from standstill when switching on StealthChop2 the first time and keep it stopped for at least 128 chopper periods to allow StealthChop2 to do the initial standstill current control.

Flags in StealthChop2

As StealthChop2 uses voltage-mode driving, status flags based on current measurement respond slower, and the driver has a delayed reaction to sudden changes of back-EMF, like a motor stall.

A motor stall, or abrupt stop of the motion during operation in StealthChop2, can lead to an overcurrent condition. Depending on the previous motor velocity, and on the coil resistance of the motor, it significantly increases motor current for a time of several 10ms. With low velocities, where the back-EMF is just a fraction of the supply voltage, there is no danger of triggering the short detection.

Switch the driver stage to the lowest current range (FS_SENSE) supporting the motor. This automatically adapts the overcurrent threshold in four steps, and thus, reduces peak currents when there is a sudden motor stall.

Open Load Flags

Analog Devices motor driver ICs offer advanced diagnostic features to ensure safe and reliable motor operation. Among these features are open load A (OLA) and open load B (OLB), which are specifically designed to detect a phase loss (a condition where one of the motor phases is disconnected or fails to carry current). When the motor is supposed to be energized but no current is flowing through one of the phases (A or B), the respective flag (OLA or OLB) is triggered.

In StealthChop2 mode, the status information is different from the cycle-by-cycle regulated SpreadCycle mode for the OLA and OLB flags.

- If OLA and OLB are not set, this indicates the current regulation is reaching the nominal current on both coils.
- If OLA and OLB flags are constantly set, this indicates an interrupted motor coil.
- If OLA and OLB are flickering, this indicates differences in motor-coil resistance exceeding approximately 5%.
- One or both flags are active if the current regulation does not succeed in scaling up to the full target current within the last few full steps (because no motor is attached or a high velocity exceeds the PWM limit).

If desired, do an on-demand, open-load test using the SpreadCycle chopper, as it directly tests if the motor coils are connected. With StealthChop2, PWM_SCALE_SUM can be checked to detect the correct coil resistance.

Note: When there is an open-load situation on one coil, the current regulation can exceed the target current on the other coil up to the overcurrent detection trip point. This is because the current regulation in certain situations due to measurement restriction only regulates the current on the coil with higher target current. In critical applications, check for open load in SpreadCycle, first.

Information on Motor State from PWM_SCALE_SUM

Information about the motor state is available with automatic scaling by reading out PWM_SCALE_SUM. As this parameter reflects the actual voltage required to drive the target current into the motor, it depends on several factors: motor load, coil resistance, supply voltage, and current setting. Therefore, an evaluation of the PWM_SCALE_SUM value allows checking the motor operation point. When reaching the limit (1023), the current regulator cannot sustain the full motor current, example, due to a permanent or temporary drop in supply voltage.

Freewheeling and Passive Braking

StealthChop2 provides different options for motor standstill. These options can be enabled by setting the standstill current IHOLD to 0 and choosing the desired option using the PWM_FREEWHEEL setting. The desired option is enabled after a time period specified by TPOWERDOWN and IHOLDDELAY. Current regulation is frozen once the motor target current is at zero current to ensure a quick start-up. With the freewheeling options, both freewheeling and passive braking can be realized. Passive braking is an effective eddy current motor braking, which consumes a minimum amount of energy because no active current is driven into the coils. However, passive braking allows slow turning of the motor when a continuous torque is applied.

Parameters Controlling StealthChop2

Table 9 contains all the parameters related to the StealthChop2 chopper mode.

Table 9. Parameters Controlling StealthChop2

PARAMETER	DESCRIPTION	SETTING	NOTES
EN_PWM_MODE	General enable for use of StealthChop2 (GCONF register). Default = 0.	0	StealthChop2 disabled. SpreadCycle active.
		1	StealthChop2 enabled (depending on velocity thresholds). When initially enabling, the motor should be at standstill. Also see the automatic tuning flow chart in *Figure 47*.
PWM_MEAS_SD_ENABLE	Control of current measurement during slow decay phase. 1 is recommended for most universal use. Default = 0.	0	Current measured during on-phases only. Lower current limit applies.
		1	Current measured during slow decay phases in addition to overcoming lower current limit.
PWM_DIS_REG_STST	This option disables current regulation and eliminates any regulation noise during standstill. If disabled, the motor current can change during extended periods of standstill time in case the supply voltage changes or the motor cools down. Default = 0.	0	Current regulation always on.
		1	Disable current regulation when motor is in standstill and current is reduced (less than IRUN).
TPWMTHRS	Specifies the upper velocity for operation in StealthChop2. Enter the TSTEP reading (time between two microsteps) when operating at the desired threshold	0 to 1048575	StealthChop2 is disabled if TSTEP falls under TPWMTHRS.

	velocity. Default = 0.		
PWM_LIM	This register clips the PWM amplitude to a lower value when switching from SpreadCycle to StealthChop2. The driver stores the previously used StealthChop2 amplitude during SpreadCycle operation. For best results with automatic switching between both modes, set to 15 and set the SG4_THRS.sg_angle_offset bit. Default = 12.	0 to 15	Upper 4 bits of 8-bit amplitude limit.
PWM_AUTOSCALE	Enable automatic current scaling using current measurement. If off, use forward-controlled, velocity-based mode. Default = 1.	0	Forward-controlled mode.
		1	Automatic scaling with current regulator.
PWM_AUTOGRAD	Enable automatic tuning of PWM_GRAD_AUTO. Default = 1.	0	Disable. Use PWM_GRAD from register instead.
		1	Enable.
PWM_FREQ	PWM frequency selection. Use the lowest setting that gives good results to have the best current regulation (highest PWM resolution). The frequency measured at each of the chopper outputs is half of the effective chopper frequency fPWM. Default = 0.	0	fPWM = 2/1024 fCLK.
		1	fPWM = 2/683 fCLK.
		2	fPWM = 2/512 fCLK.
		3	fPWM = 2/410 fCLK.
PWM_REG	User-defined PWM amplitude regulation loop P-coefficient. A higher value leads to a higher adaptation speed when pwm_autoscale = 1. However, values that are too high can lead to regulation noise and overshoots. Default = 4.	1 to 15	Results in 0.5 to 7.5 steps for PWM_SCALE_AUTO regulator per full step.
PWM_OFS	User-defined PWM amplitude (offset) for velocity-based scaling and initialization value for automatic tuning of PWM_OFFS_AUTO. Default = 0x1D.	0 to 255	PWM_OFS = 0 disables linear current scaling based on current setting.
PWM_GRAD	User defined PWM amplitude (gradient) for velocity-based scaling and initialization value for automatic tuning of PWM_GRAD_AUTO allows for preloading of previously determined PWM_OFS. Default = 0.	0 to 255	—
PWM_SCALE_SUM	Actual PWM scaling, as determined by the actual settings. This value is shown in higher precision (10-bit) compared to 8-bit for PWM_GRAD/OFS_AUTO values.	0 to 1023	—
PWM_FREEWHEEL	Standstill option when motor current setting is zero (I_HOLD = 0). Only available when StealthChop2 is enabled. The freewheeling option makes the motor easily movable, while both the coil short options realize a passive brake. Default = 0.	0	Normal operation.
		1	Freewheeling.
		2	Coil short through LS drivers.
		3	Coil short through HS drivers.

PWM_SCALE_AUTO	Readback of the actual StealthChop2 voltage PWM scaling correction, as determined by the current regulator. Regulates close to 0 during tuning.	-255 to +255	Read-only. Scaling value is frozen when operating in SpreadCycle.
PWM_GRAD _AUTO PWM_OFS _AUTO	Allow monitoring of the automatic tuning and determination of initial values for PWM_OFS and PWM_GRAD.	0 to 255	Read-only.
TOFF	General enable for the motor driver; the actual value does not influence StealthChop2. Default = 0.	0	Driver off.
		1 to 15	Driver enabled.
TBL	Comparator blank time. Choose a setting of 1 or 2 for typical applications. For higher capacitive loads, 3 may be required. Lower settings allow StealthChop2 to regulate down to lower coil current values. A higher setting leads to a higher minimum current due to lower duty-cycle limitation in StealthChop2 unless setting PWM_MEAS_SD_ENABLE. Default = 2.	0	20 tCLK
		1	28 tCLK
		2	36 tCLK
		3	54 tCLK

SpreadCycle and Classic Chopper

While StealthChop2 is a voltage-mode, PWM-controlled chopper, SpreadCycle is a cycle-by-cycle current control. Therefore, it can react extremely fast to changes in motor velocity or motor load. The currents through both the motor coils are controlled using choppers. The choppers work independent of each other. *Figure 20* shows the different chopper decay phases.

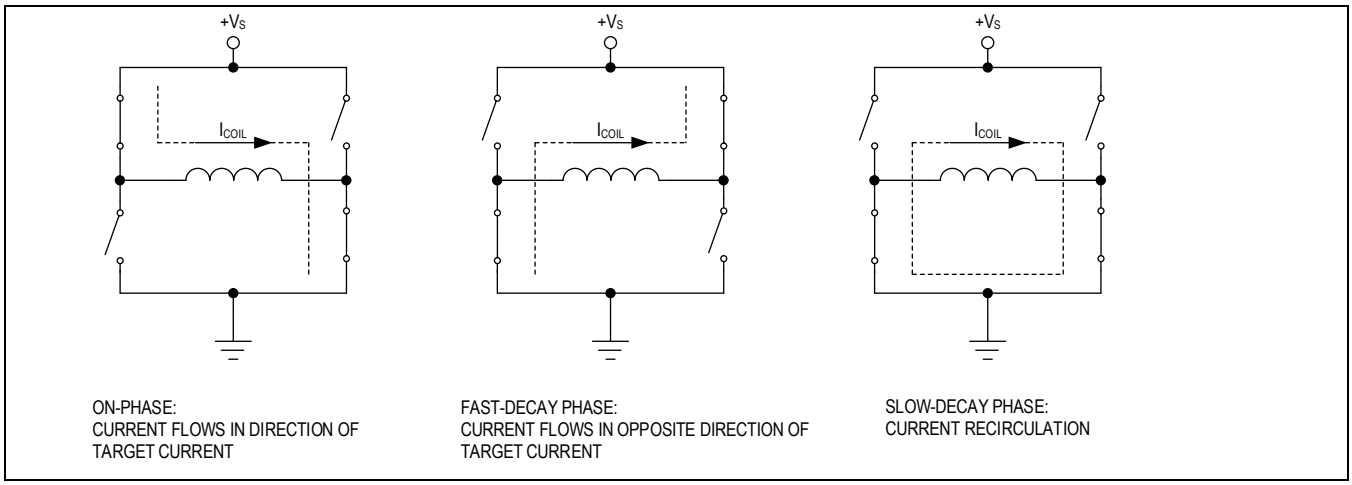


Figure 20. Typical Chopper Decay Phases

Although the current can be regulated using only on-phases and fast-decay phases, insertion of the slow-decay phase is important to reduce electrical losses and current ripple in the motor. The duration of the slow-decay phase is specified in a control parameter and sets an upper limit on the chopper frequency. The current comparator measures the coil current during phases when the current flows through exactly one low-side transistor, but not during the slow-decay phase. The slow-decay phase is terminated by a timer. The on-phase is terminated by the comparator when the current through the coil reaches the target current. The fast-decay phase can be terminated by either the comparator or another timer.

When the coil current is switched, spikes in the $R_{DS(ON)}$-based current measurement occur due to charging and discharging parasitic capacitance. During this time, typically, one or two microseconds, the current cannot be measured. Blanking is the time when the input to the comparator is masked to block these spikes.

There are two cycle-by-cycle chopper modes available: the high-performance chopper algorithm SpreadCycle (CHM = 0) and a proven constant off-time chopper mode (CHM = 1). The constant off-time mode cycles through three phases: on,

fast decay, and slow decay. The SpreadCycle mode cycles through four phases: on, slow decay, fast decay, and a second slow decay. *Table 10* shows the parameters related to these two chopper modes.

The chopper frequency is an important parameter for a chopped motor driver. A frequency too low might generate audible noise. A higher frequency reduces current ripple in the motor, but with a frequency too high magnetic losses may rise. Also, power dissipation in the driver rises with increasing frequency due to the increased influence of switching slopes causing dynamic dissipation. Therefore, a compromise must be found. Most motors work optimally in a frequency range of 25kHz to 40kHz. The chopper frequency is influenced by a number of parameter settings, as well as by the motor inductivity and supply voltage.

Tip: A chopper frequency in the range of 25kHz to 40kHz gives a good result for most motors when using SpreadCycle. A higher frequency leads to increased switching losses.

Table 10. Parameters Controlling SpreadCycle and Classic Constant Off-Time Chopper

PARAMETER	DESCRIPTION	SETTING	NOTES
TOFF	Sets the slow-decay time (off time). This setting also limits the maximum chopper frequency. For operation with StealthChop2, this parameter is not used, but it is required to enable the motor. When operating with StealthChop2 only, any setting can be used. Setting this parameter to zero completely disables all driver transistors and the motor can freewheel. Default = 0.	0	Chopper off.
		1 to 15	Off time setting NCLK = 24 + 32 × TOFF (1 works with minimum blank time TBL of 28 clocks).
TBL	Selects the comparator blank time. This time must safely cover the switching event and duration of the ringing. For most applications, a setting of 1 or 2 is appropriate. For highly capacitive loads, example, when filter networks are used, a setting of 2 or 3 is required. Default = 2.	0	20 tCLK Restriction: Use this setting only in combination with an external clock up to 10MHz
		1	28 tCLK
		2	36 tCLK
		3	54 tCLK.
CHM	Selection of the chopper mode. Default = 0.	0	SpreadCycle
		1	Classic constant off-time chopper.

SpreadCycle Chopper

The SpreadCycle chopper algorithm is a precise and simple-to-use chopper mode, which automatically determines the optimum length for the fast-decay phase. The SpreadCycle provides superior microstepping quality even with default settings. Several parameters are available to optimize the chopper to the application.

Each chopper cycle comprises an on-phase, a slow-decay phase, a fast-decay phase, and a second slow-decay phase. The two slow-decay phases and the two blank times per chopper cycle put an upper limit to the chopper frequency. The slow-decay phases typically make up for about 30% to 70% of the chopper cycle in standstill, and are important for low motor and driver power dissipation.

Example Calculation of TOFF:

Target chopper frequency: 25kHz

⇨ TOFF = 1/25kHz × 50/100 × 1/2 = 10µs

Assumption: Two slow-decay cycles make up for 50% of the overall chopper cycle time.

For the TOFF setting, this means: TOFF = (tOFF × fCLK - 24)/32

With 12.5MHz clock, this results in TOFF = 3.16, which requires setting TOFF = 3.

With 16MHz clock, this results in TOFF = 4.25, which requires setting of TOFF = 4.

The hysteresis start setting forces the driver to introduce a minimum amount of current ripple into the motor coils. The current ripple must be higher than the current ripple caused by resistive losses in the motor to give best microstepping results. This allows the chopper to precisely regulate the current for both the rising and falling target current. The time required to introduce the current ripple into the motor coil also reduces the chopper frequency. Therefore, a higher hysteresis setting leads to a lower chopper frequency. The motor inductance limits the ability of the chopper to follow a changing motor current. Also, the duration of the on-phase and fast decay must be longer than the blanking time because the current comparator is disabled during blanking.

It is easiest to find the best setting by starting from a low hysteresis setting (example, HSTRT = 0, HEND = 0) and increasing HSTRT, until the motor runs smoothly at low-velocity settings. This can best be checked when measuring the motor current with a current probe. Checking the sine-wave shape near the zero transition shows a small ledge between both the half waves in case the hysteresis setting is too small. At medium velocities (example, 100 full steps to 400 full steps per second), a hysteresis setting too low leads to increased humming and vibration of the motor. A hysteresis setting too high leads to reduced chopper frequency and increased chopper noise but does not yield any benefit for the wave shape.

The setting is independent of the motor because higher current motors typically also have a lower coil resistance. Therefore, choosing a low-to-medium default value for the hysteresis (for example, effective hysteresis = 4) normally fits most applications. The setting can be optimized by experimenting with the motor. A setting too low results in reduced microstep accuracy, while a setting too high leads to more chopper noise and motor power dissipation. When the fast-decay time is slightly longer than the blanking time, the setting is optimum. If this is hard to obtain, the off-time setting can be reduced.

The hysteresis principle can in some cases lead to the chopper frequency becoming too low, example, when the coil resistance is high compared to the supply voltage. This is avoided by splitting the hysteresis setting into a start setting (HSTRT + HEND) and an end setting (HEND). An internal hysteresis decrementer (HDEC) interpolates between both the settings, by decrementing the hysteresis value stepwise each 16 system clocks. At the beginning of each chopper cycle, the hysteresis begins with a value that is the sum of the start and end values (HSTRT + HEND), and decrements during the cycle, until either the chopper cycle ends or the hysteresis end value (HEND) is reached, as shown in *Figure 21*. This way, the chopper frequency is stabilized at high amplitudes and low supply voltage situations, if the frequency gets too low. This prevents the frequency from reaching the audible range. *Table 11* lists the related hysteresis parameters.

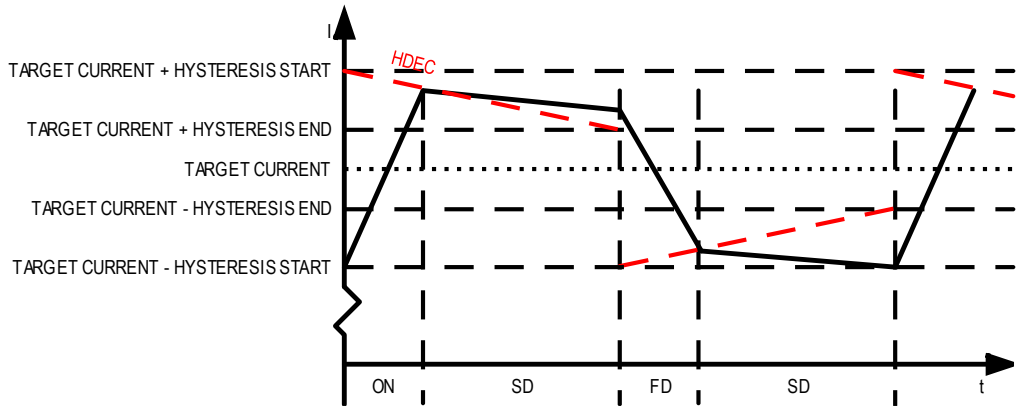


Figure 21. SpreadCycle Chopper Scheme Showing Coil Current During a Chopper Cycle

Table 11. SpreadCycle Mode Hysteresis Parameters

PARAMETER	DESCRIPTION	SETTING	NOTES
HSTRT	Hysteresis start setting. This value is an offset from the hysteresis end value HEND.	0 to 7	Hysteresis start = 1 to 8. This value adds to hysteresis end.

	Default = 5.		
HEND	Hysteresis end setting. Sets the hysteresis end value after a number of decrements. The sum HSTRT + HEND must be ≤ 16. At a current setting of maximum 30 (amplitude reduced to 240), the sum is not limited. Default = 2.	0 to 2	-3 to -1: Negative hysteresis end.
		3	0: Zero hysteresis end.
		4 to 15	1 to 12: Positive hysteresis end.

Example for Hysteresis Setting:

A hysteresis of 4 is desired. If a hysteresis decrement is not used, set as follows:

HEND = 6 (sets an effective end value of 6 - 3 = 3).

HSTRT = 0 (sets minimum hysteresis of 1: 3 + 1 = 4).

To take advantage of the variable hysteresis, most of the value can be set to the HSTRT, example, 4, and the remaining 1 to hysteresis end. The resulting configuration register values are as follows:

HEND = 4 (sets an effective end value of 1).

HSTRT = 2 (sets an effective start value of hysteresis end +3: 1 + 3 = 4).

Classic Constant Off-Time Chopper

The classic constant off-time chopper is an alternative to SpreadCycle. The constant off-time chopper uses a fixed-time fast decay following each on-phase. While the duration of the on-phase is determined by the chopper comparator, the fast-decay time must be long enough for the driver to follow the falling slope of the sine-wave, but it should not be so long that it causes excess motor current ripple and power dissipation. *Figure 22* shows the basic principle. This can be tuned using an oscilloscope or evaluating the motor smoothness at different velocities. A good starting value is a fast-decay time setting similar to the slow-decay time setting.

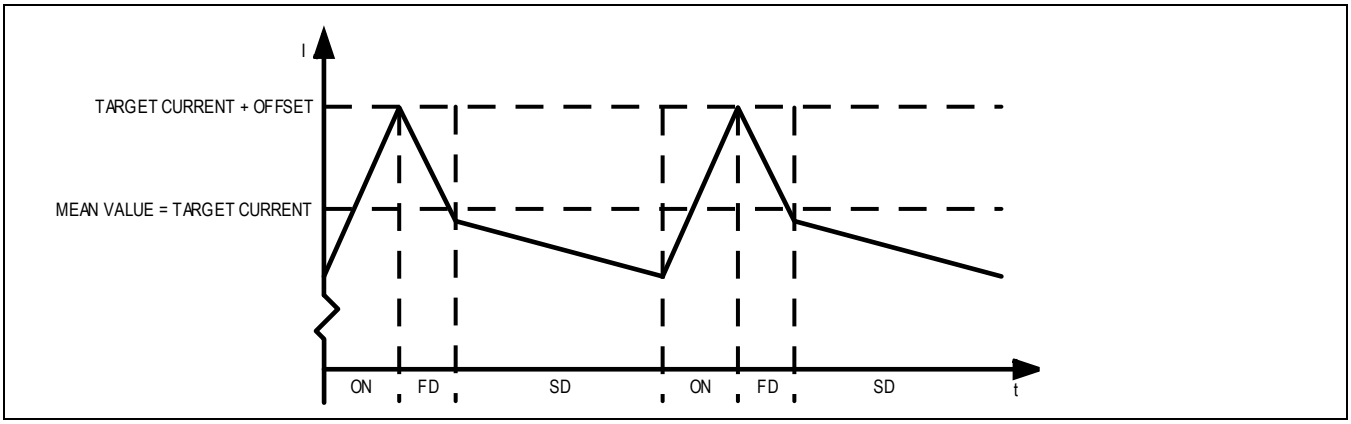


Figure 22. Classic Constant Off-Time Chopper with Offset Showing Coil Current

After tuning the fast-decay time, the offset should be tuned for a smooth zero crossing. This is necessary because the fast-decay phase makes the absolute value of the motor current lower than the target current, as shown in *Figure 23*. If the zero offset is too low, the motor stands still for a short moment during current zero crossing. If it is set too high, it makes a larger microstep. Typically, a positive offset setting is required for the smoothest operation. *Table 12* shows the related parameters for this mode.

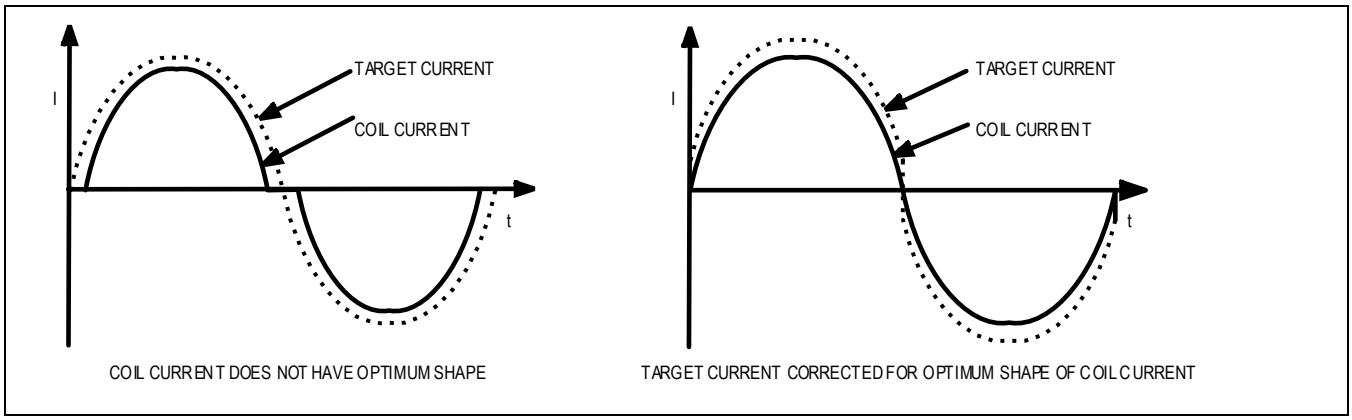


Figure 23. Zero Crossing with Classic Chopper and Correction Using Sine-Wave Offset

Table 12. Parameters Controlling Constant Off-Time Chopper Mode

PARAMETER	DESCRIPTION	SETTING	NOTES
TFD (FD3 and HSTRT)	Fast-decay time setting. With CHM = 1, these bits control the portion of fast decay for each chopper cycle. Default = 5.	0	Slow decay only.
		1 to 15	Duration of fast-decay phase.
OFFSET (HEND)	Sine-wave offset. With CHM = 1, these bits control the sine-wave offset. A positive offset corrects for zero crossing error. Default = 2.	0 to 2	Negative offset: -3 to -1
		3	No offset: 0
		4 to 15	Positive offset 1 to 12.
DISFDCC	Selects usage of the current comparator for termination of the fast-decay cycle. If current comparator is enabled, it terminates the fast-decay cycle in case the current reaches a higher negative value than the actual positive value. Default = 0.	0	Enable comparator termination of fast-decay cycle.
		1	End by time only.

Velocity-Based Mode Control

The TMC5221/TMC5222 allow the configuration of different chopper modes and modes of operation for optimum motor control. Depending on the motor load, the different modes can be optimized for lowest noise and high precision, highest dynamics, or maximum torque at highest velocity. Some of the features like CoolStep or StallGuard2 are useful in a limited velocity range. A number of velocity thresholds allow combining the different modes of operation within an application requiring a wide velocity range.

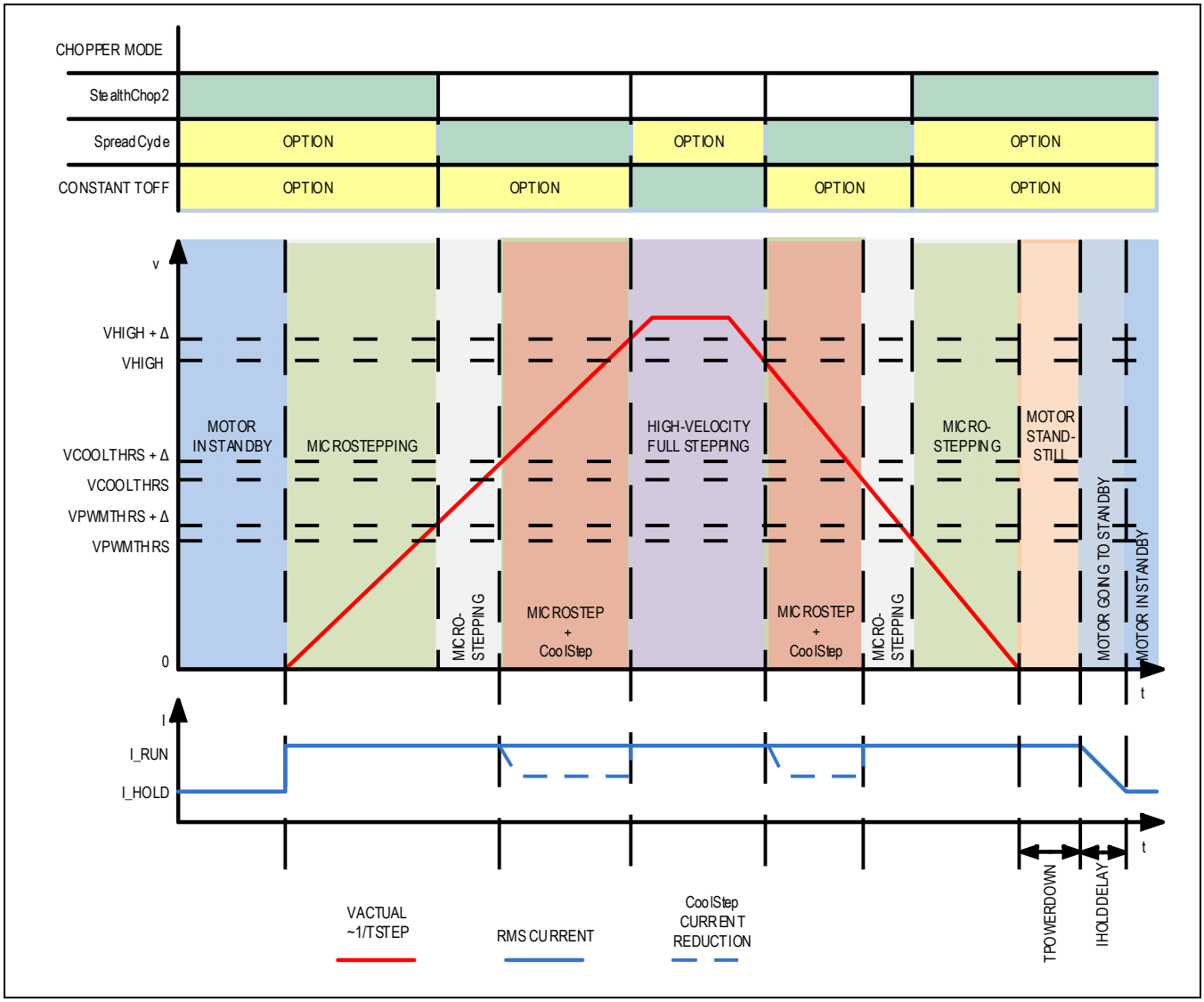


Figure 24. Choice of Velocity-Dependent Modes

Figure 24 shows all the available thresholds and the required ordering. VPWMTHRS, VHIGH, and VCOOLTHRS are determined by the settings TPWMTHRS, THIGH, and TCOOLTHRS. The velocity is described by the time interval TSTEP between each two step pulses. This allows determination of the velocity when an external step source is used. TSTEP always is normalized to 256 microstepping. This way, the thresholds do not have to be adapted when the microstep resolution is changed. The thresholds represent the same motor velocity, independent of the microstep settings. TSTEP is compared to these threshold values. A hysteresis of 1/16 TSTEP or 1/32 TSTEP is applied to avoid continuous toggling of the comparison results when a jitter in the TSTEP measurement occurs. The upper switching velocity is higher by 1/16 or 1/32 of the value set as the threshold (can be selected with configuration bit small_hysteresis in the GCONF register). The motor current can be programmed to a run and a hold level depending on the standstill flag STST.

Using automatic velocity thresholds allows tuning the application for different velocity ranges. Features like CoolStep integrate completely transparently in the setup. This way, once parameterized, they do not require any activation or deactivation by software.

Table 13. Velocity-Based Mode Control Parameters

PARAMETER	DESCRIPTION	SETTING	NOTES
STST	Indicates motor standstill in each operation mode. Time is $2^{20\text{-STANDSTILL\_TIME}}$ clocks after the last step pulse. This can be configured by STANDSTILL_TIME. Default/reset: 0.	0/1	Status bit, read-only.
TPOWER DOWN	This is the delay time after standstill (stst) of the motor-to-motor current power-down. Time range is approximately 0 seconds to 5 seconds (with fCLK = 12.5MHz). Setting 0 is no delay, 1 is one clock cycle delay. Further increment is in discrete steps of 2^{18} clock cycles. Default: 0xA.	0 to 255	Time in multiples of 2^{18} × tCLK.
TSTEP	Actual measured time between two 1/256 microsteps derived from the step input frequency in units of 1/fCLK. Measured value is (2^{20}) - 1 in case of overflow or standstill. Default/reset: 0.	0 to 1048575	Status register, read-only. Actual measured step time in multiples of tCLK.
TPWMTHRS	TSTEP ≥ TPWMTHRS: • StealthChop2 PWM mode is enabled, if configured. • DcStep™ is disabled. Default: 0.	0 to 1048575	Setting to control the upper velocity threshold for operation in StealthChop2.
TCOOLTHRS	TCOOLTHRS ≥ TSTEP ≥ THIGH: • StallGuard2/4 and CoolStep are enabled, if configured. • StealthChop2 voltage PWM mode is disabled. TCOOLTHRS ≥ TSTEP: • StallGuard2 stall output signal is enabled (if configured) for use with the external controller. Default: 0.	0 to 1048575	Setting to control the lower velocity threshold for operation with CoolStep and StallGuard2/4.
THIGH	TSTEP ≤ THIGH: • CoolStep is disabled (motor runs with normal current scale). • StealthChop2 voltage PWM mode is disabled. • If vhighchm is set, the chopper switches to chm = 1 with TFD = 0 (constant off-time with slow decay only). • Chopper sync is switched off (SYNC = 0). • If vhighfs is set, the motor operates in full-step mode and the stall detection is switched over to DcStep stall detection. Default: 0.	0 to 1048575	Setting to control the upper threshold for operation with CoolStep and StallGuard2 as well as optional high velocity step mode.
SMALL_HYSTERESIS	Hysteresis for step frequency comparison based on TSTEP (lower velocity threshold) and (TSTEP × 15/16) - 1 or (TSTEP × 31/32) - 1 (upper velocity threshold). Default: 0.	0	Hysteresis is 1/16.
		1	Hysteresis is 1/32. Recommended when used with internal motion controller.
VHIGHFS		0	No switch to full step.

	This bit enables switching to full step when VHIGH is exceeded. Switching takes place only at 45° position. The full-step target current uses the current value from the microstep table at the 45° position. Default: 0.	1	Full step at high velocities.
VHIGHCHM	This bit enables switching to chm = 1 and fd = 0 when VHIGH is exceeded. This way, a higher velocity can be achieved. Can be combined with vhighfs = 1. If set, the TOFF setting automatically is doubled during high-velocity operation to avoid doubling of the chopper frequency. Default: 0.	0	No change of chopper mode.
		1	Classic constant time off chopper at high velocities.
EN_PWM_MODE	StealthChop2 voltage PWM enable flag (depending on velocity thresholds). Switch from off to on state while in standstill only. Default: 0.	0	No StealthChop2.
		1	StealthChop2 active if configured and TSTEP > TPWMTHRS.

Ramp Generator

The ramp generator allows motion based on the target position or target velocity. It automatically calculates the motion profile taking into account acceleration and velocity settings. The TMC5221/TMC5222 integrates a new type of ramp generator, which offers faster machine operation compared to the classical linear acceleration ramps. The eight-point ramp generator allows adapting the acceleration ramps to the torque curves of a stepper motor. In position mode, it uses three different acceleration settings each for the acceleration and deceleration phases to allow for jerk-minimized ramps.

Real-World Unit Conversion

The TMC5221/TMC5222 use their internal or external clock signal as a time reference for all internal operations. Thus, all time, velocity, and acceleration settings are referenced to fCLK. *Table 14* shows all the ramp generator parameters and their units, and their dependency on fCLK. For best stability and reproducibility, it is recommended to use an external quartz oscillator as a time base, or to provide a clock signal from a microcontroller. [V_REG] and [A_REG] are internal units of the TMC5221/TMC5222. These are the values to be written to the velocity/acceleration registers of the TMC5221/TMC5222. Calculator tools and evaluation tools are available from the product website.

Table 14. Ramp Generator Parameters vs. Units

PARAMETER/SYMBOL	UNITS	DESCRIPTION
fCLK	Hz	Clock frequency of the TMC5221/TMC5222.
s	s	Second.
µS	Microstep	—
FS	Full step	—
USC (microstep count)	—	Microstep resolution in number of microsteps (that is, the number of microsteps between two full steps, normally 256).
FSC (full-step count)	—	Motor full steps per rotation example, 200.
µstep velocity v[Hz]	microsteps/s	$v[Hz] = [V\_REG] \times fCLK[Hz]/2^{24}$
µstep acceleration a[Hz/s]	microsteps/s^2	$a[Hz/s] = [A\_REG] \times fCLK[Hz]^2/2^{41}$
Rotations per second v[rps]	rotations/s	v[rps] = v[microsteps/s]/USC/FSC
rps acceleration a[rps/s^2]	rotations/s^2	a[rps/s^2] = a[microsteps/s^2]/USC/FSC
Ramp steps[microsteps] = rs	microsteps	$rs = ([V\_REG])^2/[A\_REG]/2^8$ microsteps during linear acceleration ramp (assuming acceleration from 0 to V).
TSTEP, Txxx_THRS	—	$TSTEP = f_{CLK}/f_{256STEP} = f_{CLK}/(f_{STEP} \times 256/USC)$ $= 2^{24}/(VACTUAL \times 256/USC)$.

		The time reference for velocity thresholds is referred to the actual 1/256 microstep frequency ($f_{256STEP}$) of the step input, respectively, velocity v[Hz].
Ramp generator update rate	Hz	$f_{UPDATE} = f_{CLK}/512$. VACTUAL updates with this frequency.

In rare cases, the upper acceleration limit can impose a limitation to the application, example, when working with a reduced clock frequency or high gearing and low load on the motor. To increase the effective acceleration possible, the microstep resolution of the sequencer input can be decreased. Setting the CHOPCONF options intpol = 1 and MRES = %0001 doubles the motor velocity for the same speed setting, and thus, also doubles effective acceleration and deceleration. The motor has the same smoothness but half-position resolution with this setting.

Motion Profiles

Ramp Mode

The ramp generator supports three acceleration phases and three deceleration phases with additional programmable start and stop velocities when in positioning mode.

Three different sets of acceleration and deceleration can be combined freely. The transition velocities V1 and V2 allow for velocity-dependent switching between the three acceleration and deceleration settings. A typical high-velocity application uses lower acceleration and deceleration values at higher velocities as the torque of a stepper motor declines at higher velocity. When considering friction in the system, it is clear that the deceleration capability of the system is quicker than the acceleration capability. Thus, the deceleration values can be set higher in many applications. This allows the operation speed of the motor in time-critical applications to be maximized. As target positions and ramp parameters can 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.

As a result, it is possible the motion is automatically stopped, crosses zero, and drives back again. For example, during the final deceleration phase, the target position is again changed and pulled into a closer position, and the configured deceleration value is not allowed to reach the new target position directly. This case, overshooting the target position and a second motion into the opposite direction to reach the target, is flagged by the SECOND_MOVE flag.

The ramp generator further supports automatic jerk reduction by smoothening the transition from the acceleration phase to deceleration phase and from the deceleration phase to acceleration phase by enforcing a constant velocity segment of a minimum duration (TVMAX), as required by the mechanical jerk response.

Figure 25 to *Figure 31* give some examples of typical (corner) cases.

Note:

- The start velocity can be set to zero, if not used.
- The stop velocity must always be set to a low value (1000 or down to 10), if not used. Never set VSTOP = 0 because the position might not precisely reach the configured microstep target position.
- Take care to always set VSTOP identical to or above VSTART. This ensures that even a short motion can be terminated successfully at the target position.
- Set TVMAX to 0 to disable jerk reduction.

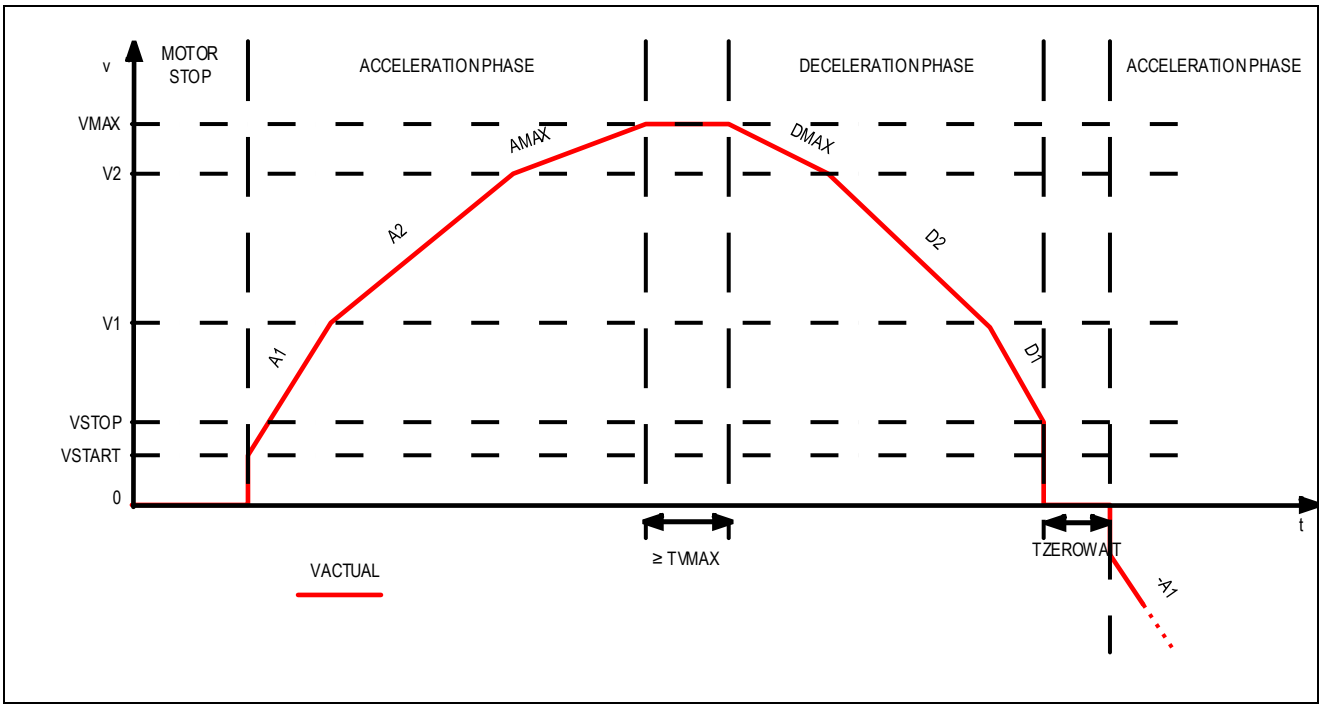


Figure 25. Ramp-Generator Velocity Trace Showing Second Move into Negative Direction

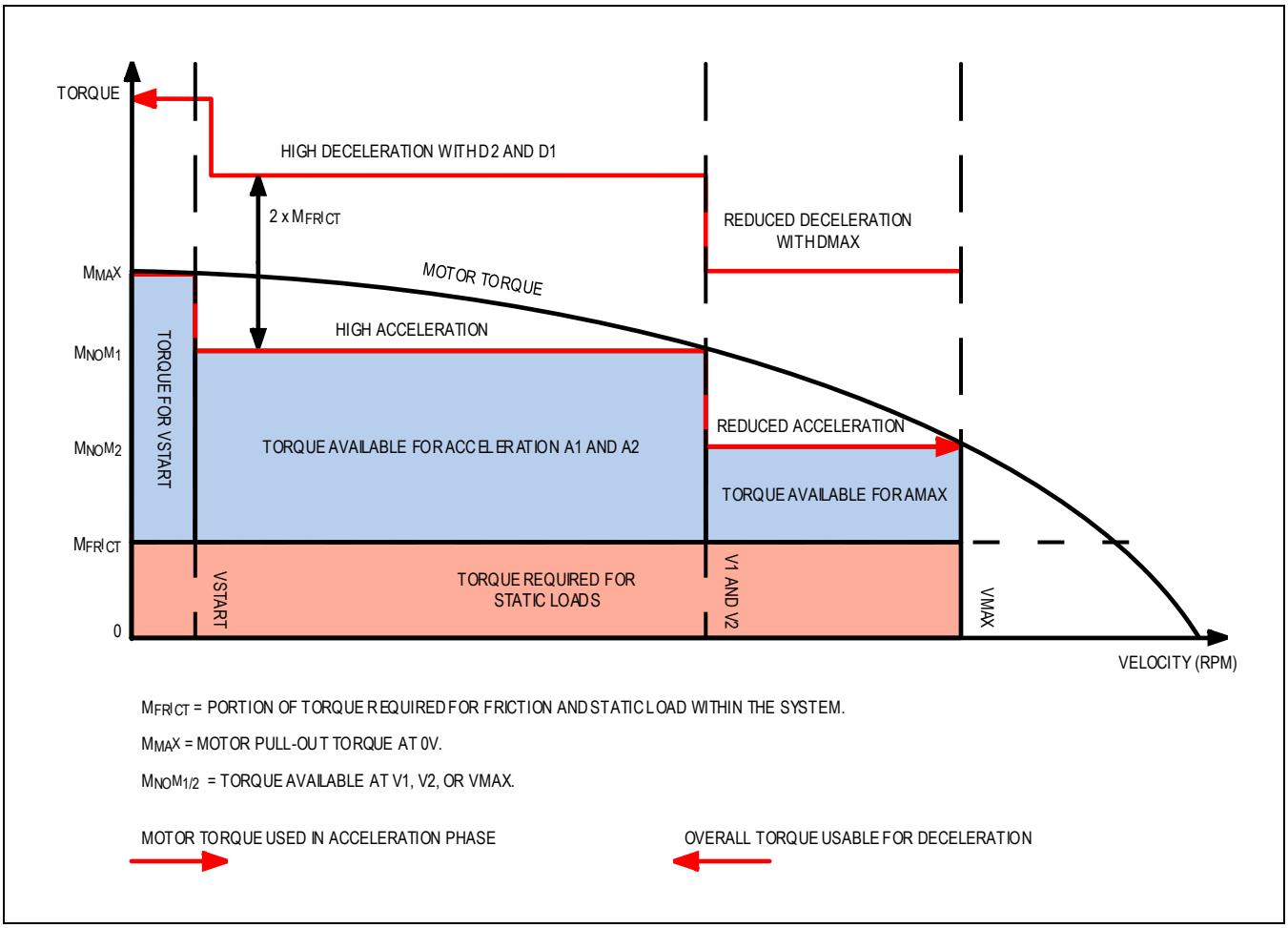


Figure 26. Optimized Motor Torque Usage with the Ramp Generator

Eight-Point Ramp (Start and Stop Velocity)

When needed, it is possible to skip the lower velocity range of the motor to accelerate short moves. This can be done using an increased VSTART and VSTOP level. Typical ranges are up to a few 10Hz full steps.

When using increased levels of start and stop velocity, it is clear that a subsequent move into the opposite direction provides a jerk identical to VSTART + VSTOP, rather than only VSTART. Given that the motor is unlikely to be able to follow this, a time delay can be set for a subsequent move by setting TZEROWAIT (only available in positioning mode). An active delay time is flagged by the T_ZEROWAIT_ACTIVE flag. Once the target position is reached, the position_reached flag is 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, A1, D1, AMAX, and DMAX are set to lower values than A2 and D2. The most critical points with regard to jerk are the transition from acceleration to deceleration with no constant velocity segment, as well as the transition from deceleration to acceleration in the case of an on-the-fly change of target position.

To address both ways, the eight-point motion profile generator allows to enforce a constant velocity segment based on a minimum segment duration (TVMAX). In case this duration cannot be kept due to insufficient distance, a reduced VMAX (VMAX´) is calculated and is used for the constant velocity segment. The minimum VMAX´ is identical to VSTOP. *Figure 27* to *Figure 31* show the resulting velocity profiles based on the different position distances for a pseudo-S-shaped configuration.

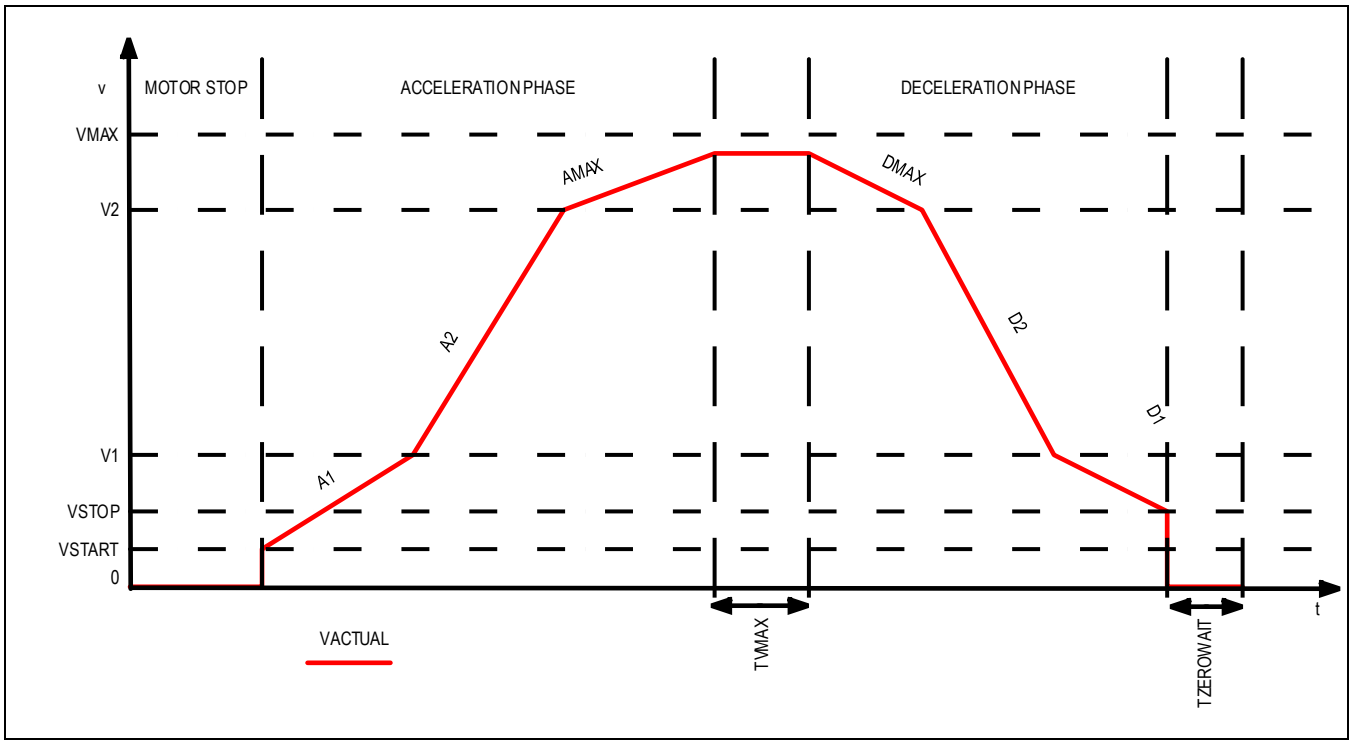


Figure 27. Eight-Point Ramp with VMAX Not Reached Due to Distance Too Low

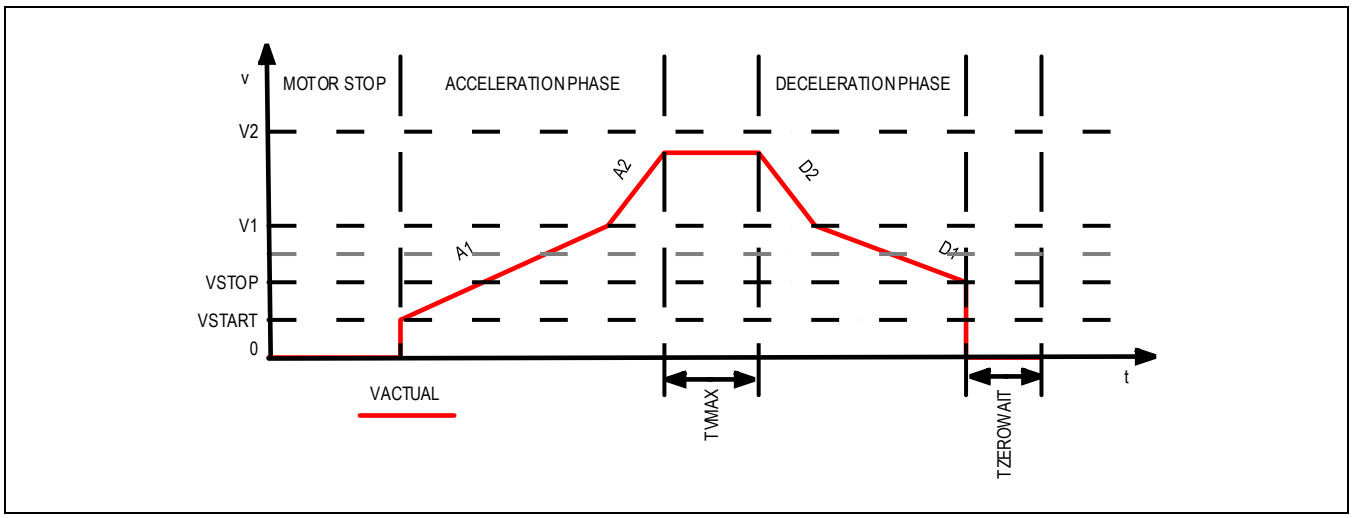


Figure 28. V2 Not Reached and No AMAX and DMAX Phase Due to Low Distance

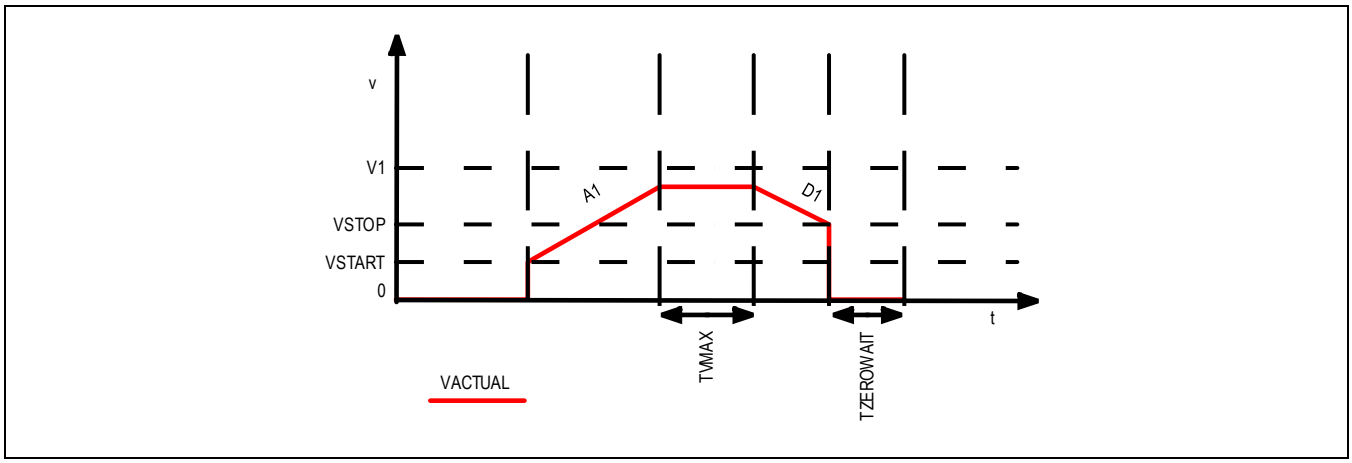


Figure 29. V1 Not Reached Due to Low Distance

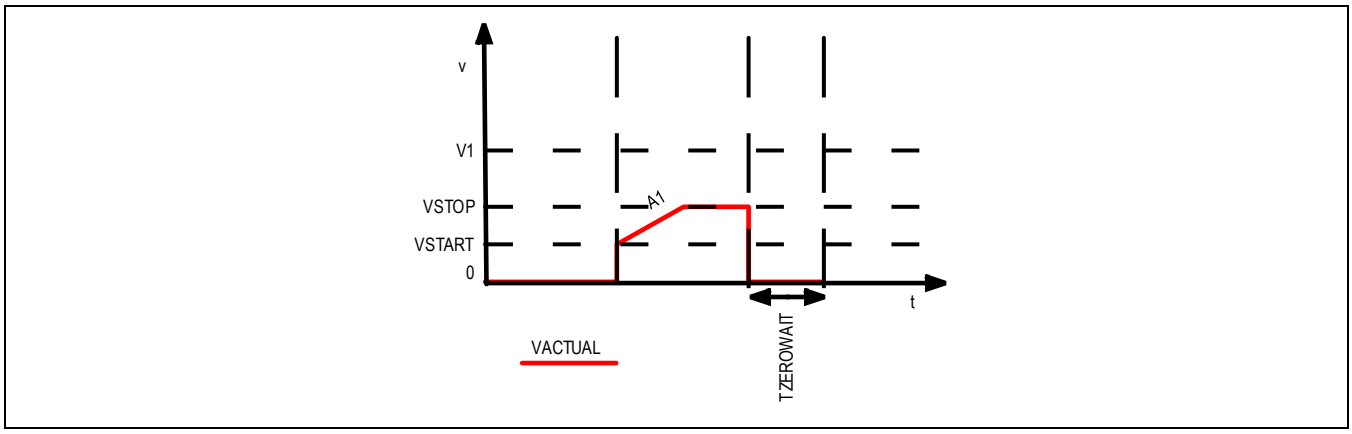


Figure 30. TVMAX Not Kept Due to Low Distance

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

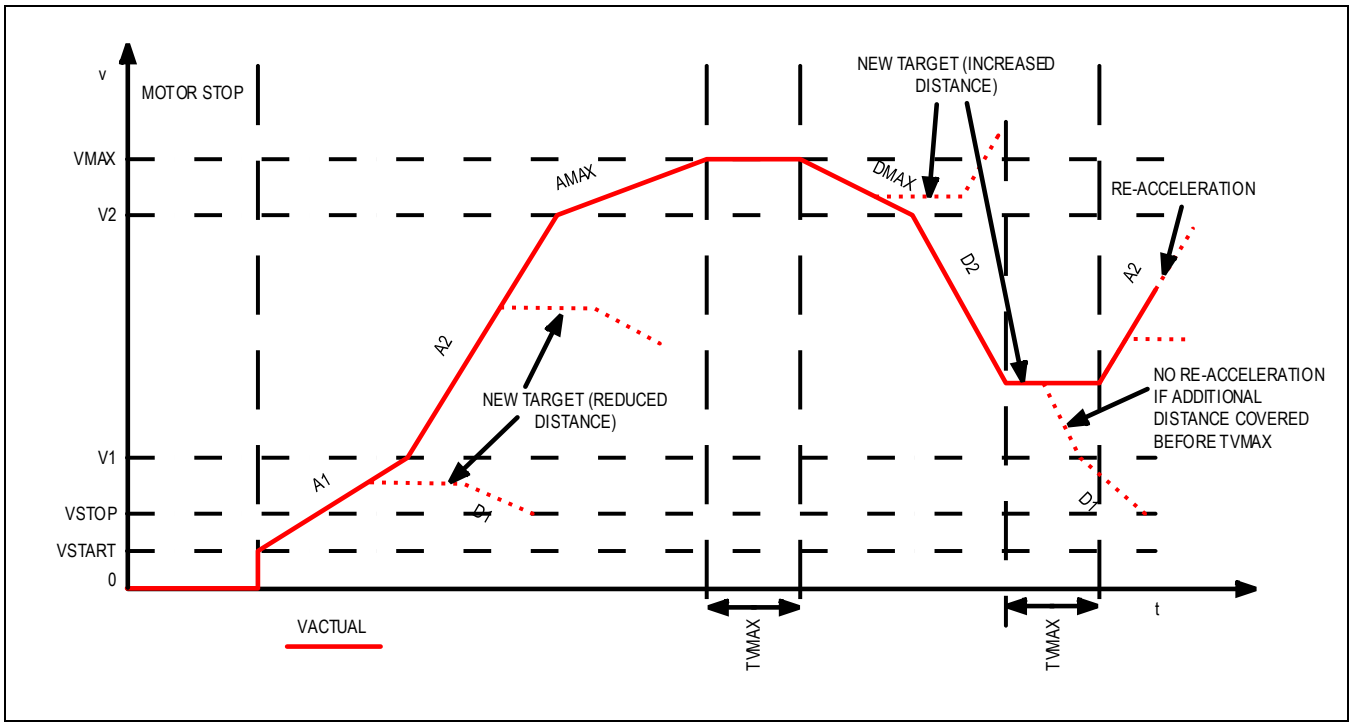


Figure 31. Eight-Point Ramp Examples with On-the-Fly Target Position Change

Velocity Mode

For ease-of-use, velocity-mode movements do not use the different acceleration and deceleration settings. For velocity mode, A1, A2, AMAX, V1, V2, and VMAX are relevant, while only using AMAX and VMAX is also fine.

To decelerate the motor to standstill, it is sufficient to set VMAX to zero. The VZERO flag signals the standstill of the motor. The VELOCITY_REACHED flag always signals the target velocity is reached.

Early Ramp Termination

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 is reached.

The options to terminate motion using the acceleration settings are:

1. Switch to velocity mode. Set VMAX = 0 and AMAX to the desired deceleration value. This stops the motor using a linear ramp. If not already on VMAX, this can trigger a TVMAX phase, if configured. Using softstop instead ignores the TVMAX setting.
2. For a stop in positioning mode, it is recommended to use softstop with virtual switches.
3. For a stop using DMAX, D1, D2, and VSTOP, trigger the deceleration phase by copying XACTUAL to XTARGET. Set TZEROWAIT sufficiently to allow the CPU to interact during this time. The driver decelerates and eventually comes to a stop. Poll the actual velocity to terminate motion during the TZEROWAIT time using option 1 or 2.
4. Activate a stop switch. This can be done with the hardware input, example, using a wired 'OR' to the stop switch input. If the hardware input is not used, and the REFL and REFR are tied to a fixed level, enable the stop function (STOP_L_ENABLE, STOP_R_ENABLE), and use the inverting function (POL_STOP_L, POL_STOP_R) to simulate the switch activation.
5. Utilize the virtual stop switches (VIRTUAL_STOP_L and VIRTUAL_STOP_R). The position comparison (X_ACTUAL vs. VIRTUAL_STOP_L/R) then triggers a stop accordingly.

Application Example: Joystick Control

Applications like surveillance cameras can be optimally enhanced using the motion controller. While joystick commands operate the motor at a user-defined velocity, the target ramp generator ensures the valid motion range is never left.

For joystick control, follow these steps:

- Use the positioning mode to control the motion direction and to set the motion limit(s). Additionally, mechanical limits can be enforced by the virtual stop switches (VIRTUAL_STOP_L and VIRTUAL_STOP_R).
- Modify VMAX depending on the joystick input at any time in the range of VSTART to the maximum value. With VSTART = 0, the stop motion can be stopped by setting VMAX = 0. The motion controller uses A1, A2, and AMAX as determined by V1 and V2 to adapt the velocity for ramping up and down.
- In case the acceleration settings are not modified, it is not necessary to rewrite XTARGET; simply modify VMAX.
- DMAX, D1, D2, and VSTOP are only used when the ramp controller slows down due to reaching the target position, or when using a softstop.

Velocity Thresholds

The ramp generator provides several velocity thresholds coupled with the actual velocity VACTUAL as shown in Figure 32. The different ranges allow programming the motor to the optimum step mode, coil current, and acceleration settings. Most applications do not require all of the thresholds, but in principle all modes can be combined. VHIGH and VCOOLTHRS are determined by the THIGH and TCOOLTHRS settings to allow determination of the velocity when an external step source is used. TSTEP is compared to these threshold values. A hysteresis of 1/16 TSTEP or 1/32 TSTEP (see the SMALL_HYSTERESIS bit in GCONF register) is applied to avoid continuous toggling of the comparison results when a jitter in the TSTEP measurement occurs. The upper switching velocity is higher by 1/16 or 1/32 of the value set as threshold. The StealthChop2 threshold TPWMTHRS is not shown. VCOOLTHRS can either be used in the StealthChop2 velocity range, or in the SpreadCycle velocity range.

The velocity thresholds for the different chopper modes and sensorless operation features are coupled to the time between each two microsteps (= TSTEP).

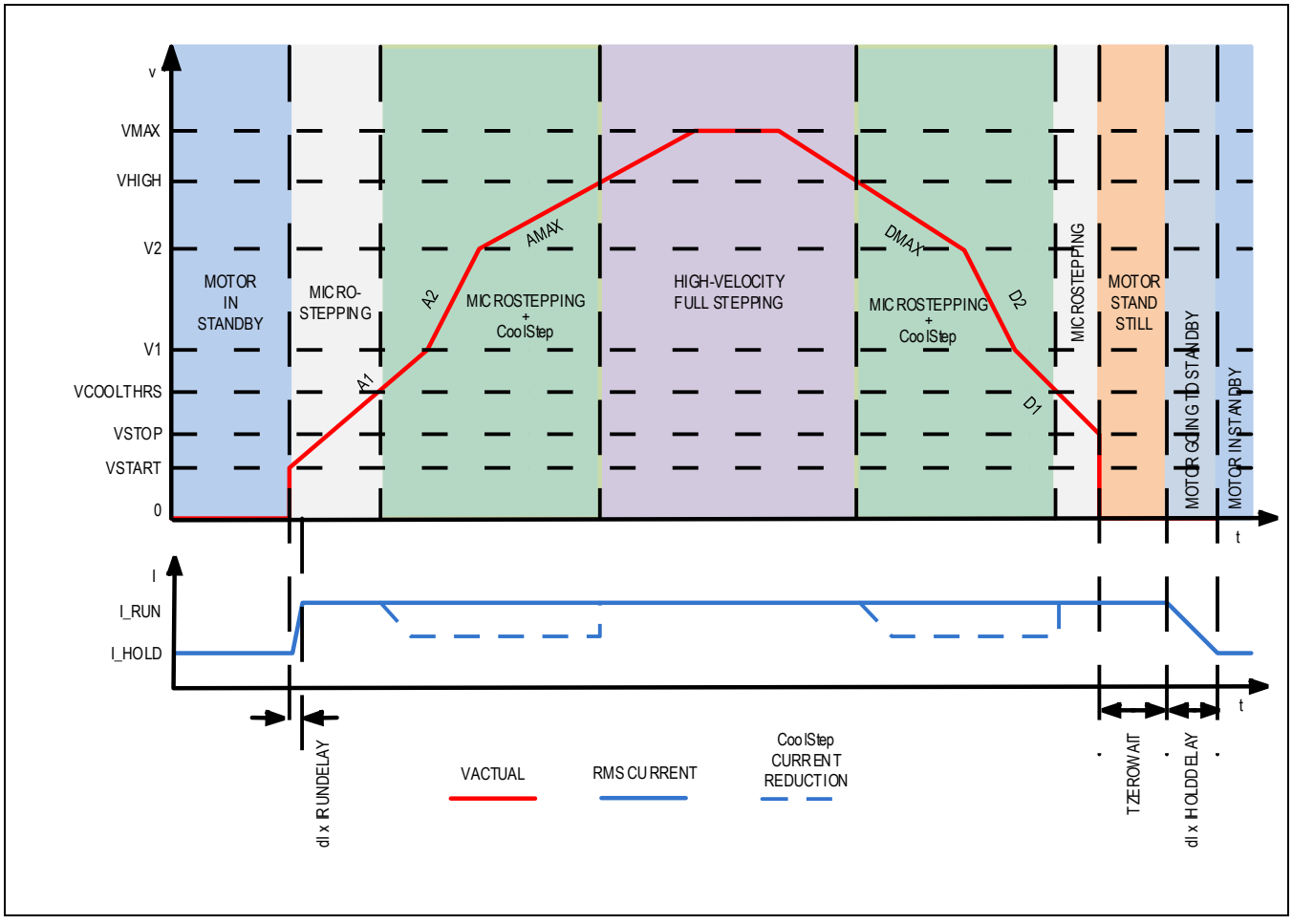


Figure 32. Ramp-Generator, Velocity-Dependent Motor Control

Reference Switches

Prior to the normal operation of the drive, an absolute reference position must be set. The reference position can be found using a physical hard stop, which can be detected by StallGuard2, StallGuard4, or by an electromechanical reference switch.

In case of a linear drive, the mechanical motion range must not be exceeded, as shown in *Figure 33*. This can also be ensured for abnormal situations by enabling the stop switch functions for the left and right reference switches. Therefore, the ramp generator responds to a number of stop events, as configured in the SW_MODE register. There are two ways to stop the motor:

- Abruptly using a switch. This is useful in an emergency and for StallGuard2/StallGuard4-based homing.
- The motor can be softly decelerated to zero with the deceleration settings (DMAX, V2, D2, V1, and D1) using the softstop function (bit EN_SOFTSTOP = 1).

Tip: Latching of the ramp position XACTUAL to the holding register XLATCH upon a switch event gives a precise snapshot of the position of the reference switch.

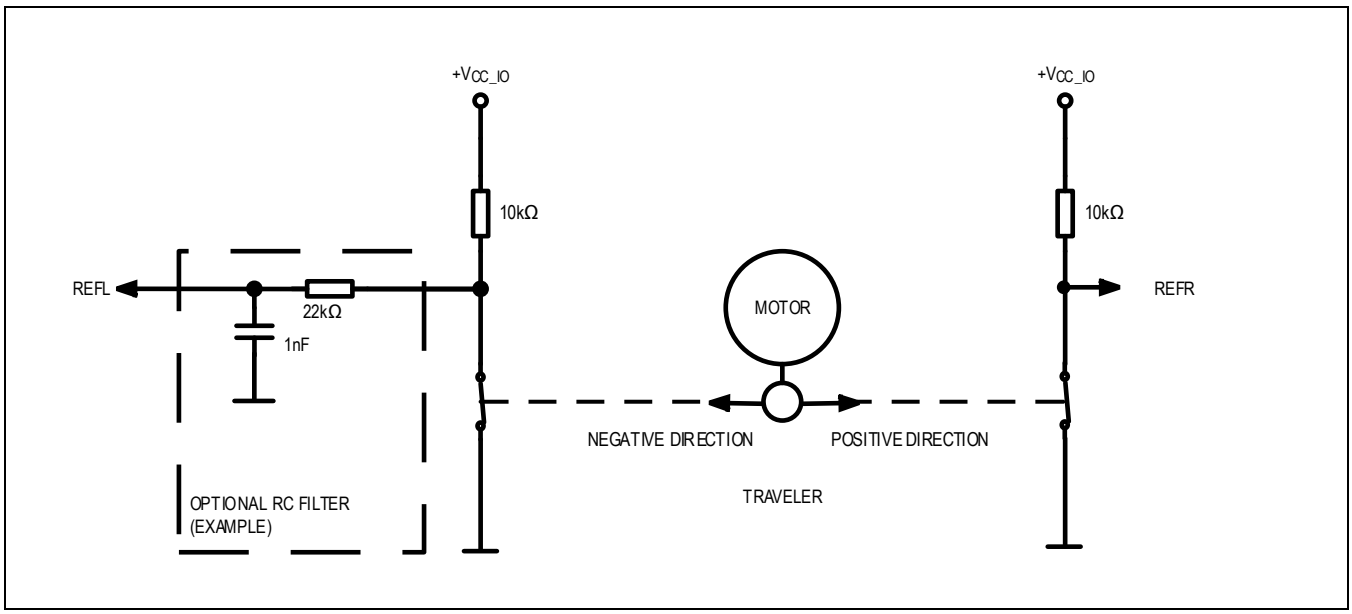


Figure 33. Using Reference Switches (Example)

Normally open or normally closed switches can be used by programming the switch polarity or selecting the pull-up or pull-down resistor configuration. A normally closed switch is failsafe with respect to an interrupt of the switch connection. Switches that can be used are:

- Mechanical switches
- Photo interrupters
- Hall sensors

Be careful to select reference switch resistors matching the switch requirements. In the case of long cables, additional RC filtering might be required near the TMC5221/TMC5222 reference inputs. Adding an RC filter also reduces the danger of destroying the logic level inputs by wiring faults, but it adds a certain delay that should be considered with respect to the application.

To implement a homing procedure, do the following:

- Make sure the home switch is not pressed, example, by moving away from the switch.
- Activate the position latching upon the desired switch event and activate the motor (soft) stop upon active switch. StallGuard2/StallGuard4-based homing is always a hard stop regardless of the EN_SOFTSTOP setting.
- Start a motion ramp into the direction of the switch. (Move to a more negative position for a left switch and a more positive position for a right switch). This motion can be timed out using a position ramping command.

- As soon as the switch is hit, the position is latched and the motor is stopped. Wait until the motor is in standstill again by polling the actual velocity VACTUAL or checking vzero or the standstill flag.
- Switch the ramp generator to hold mode, and calculate the difference between the latched position and actual position. For StallGuard2/StallGuard4-based homing or when using hard stop, XACTUAL stops exactly at the home position so there is no difference (0).
- Write the calculated difference into the actual position register. Homing is now finished. A move to position 0 brings back the motor exactly to the switching point. If StallGuard2/StallGuard4 is used for homing, read and write back RAMP_STAT to clear the StallGuard2/StallGuard4 stop event event_stop_sg and release the motor from the stop condition.

Virtual Reference Switches

The TMC5221/TMC5222 feature virtual reference switches to support applications that only have a single or no reference switch (StallGuard2/StallGuard4 homing) to safely limit the physical motion range. *Figure 34* illustrates this. The virtual stop switches are active when the actual motor position (XACTUAL) exceeds VIRTUAL_STOP_R during a movement into a positive direction or falls below VIRTUAL_STOP_L during a movement in a negative direction. Enable virtual stop switches by setting EN_VIRTUAL_STOP_L or EN_VIRTUAL_STOP_R. Each virtual stop switch only blocks motion into the respective direction.

Set the values of the virtual stops (VIRTUAL_STOP_R and VIRTUAL_STOP_L) with sufficient distance to the overflow/ underflow ranges of the signed 32-bit motion range to allow motor deceleration if soft deceleration is used.

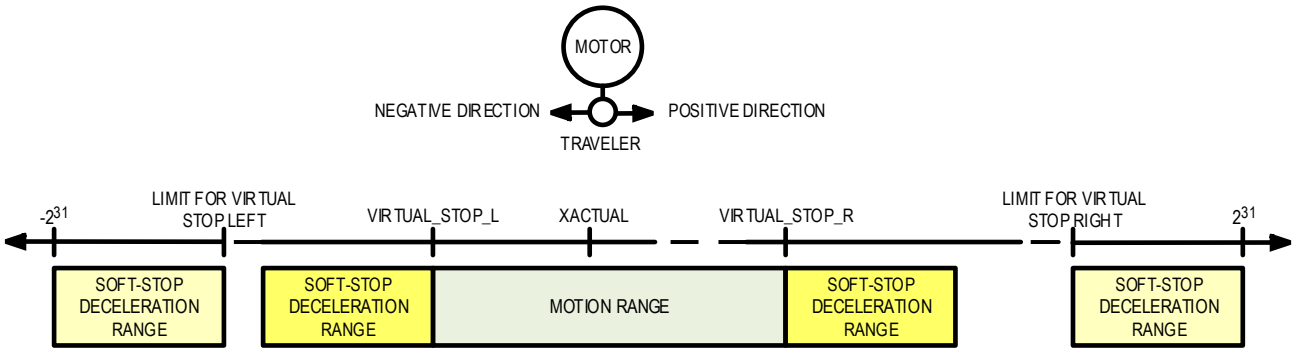


Figure 34. Virtual Stop Switches and Limit Visualization

Automatic Cyclic Motion

This feature can be used to enable continuous movement between two previously stored positions.

To prepare, write a left and a right target position into the registers VIRTUAL_STOP_L and VIRTUAL_STOP_R. Configure the ramp generator for positioning mode and set RAMPMODE to positioning mode.

Note: Virtual stops should be disabled.

To start motion, move to one of the two positions stored in VIRTUAL_STOP_L/_R as a starting point. Once this position is reached (example. POSITION_REACHED in RAMP_STAT), set bit EN_AUTO_FEATURE.

Starting from this, the motion controller continuously moves between the positions stored in the VIRTUAL_STOP_L and VIRTUAL_STOP_R registers. Clear the bit EN_AUTO_FEATURE to disable the cyclic update of the target positions and stop the function. The bit also is cleared when writing to XTARGET using the communication interface or when using the automatic safe position interrupt.

Ramp-Generator Response Time

The ramp generator is implemented in hardware and executes commands in less than a microsecond, switching over to the desired mode and target values taking effect.

A velocity accumulator updates the velocities each 512 clock cycles, based on the actual acceleration setting, to give a smooth acceleration.

However, at low motion velocities and low acceleration settings, example, at the start of positioning ramp (VSTART) or stop (VSTOP), the actual step pulse rate is comparatively low.

Therefore, a significant delay can add to the execution of the first and last steps, as determined by the selected microstep velocity.

For example, at a microstep velocity of 10Hz, a time of 100ms expires in between each two steps. Given that (at least a part of) the last microstep of a ramp is executed with a velocity equal to VSTOP. This can cause significant delay to reach the target position. Set VSTOP in the minimum range of 100 to 1000 for quick ramp termination (100 yields approximately < 10ms and 1000 yields approximately < 1ms).

Position-Compare Functions

The position-compare function allows the triggering of external events synchronously to the motor motion. The function outputs an active-high level whenever XACTUAL = X_COMPARE. The duration of the pulse, thus, corresponds to the time that XACTUAL matches X_COMPARE, that is, the duration of one microstep at the actual velocity.

A repetition function allows triggering a periodic compare pulse. Use X_COMPARE_REPEAT to program the desired period (microstep distance) with up to 2^{24} - 1 steps. *Table 15* lists the options.

Table 15. X_COMPARE_REPEAT Options and Periodic Pulse Behavior

VALUE	DESCRIPTION
0	No repetition.
1	The output goes active when reaching X_COMPARE for the first time. It disables for one clock-cycle when leaving the position and reactivates right away with the next step. To disable, set X_COMPARE_REPEAT to 0.
> 1	Results in a continuous pulse train, once the first compare position has been passed until the motion direction is changed. Distance between compare pulses: 2 to 224 - 1 microsteps.

Whenever the position-compare condition XACTUAL = X_COMPARE goes from a true to a false state, X_COMPARE is automatically incremented or decremented by the value programmed to X_COMPARE_REPEAT. The decision to increment or decrement X_COMPARE is based on the actual motion direction. This way, the first compare event in a motion is given by the content of X_COMPARE and the distance of subsequent events is programmed by X_COMPARE_REPEAT. When changing motion direction, or whenever the next pulse position is modified by software, be sure to first disable the repetition mechanism by setting X_COMPARE_REPEAT = 0. Next, reprogram X_COMPARE with the next desired pulse position because the previously automatically generated next position still lies in the previous motion direction and is not reached. Following the write access to X_COMPARE, the repetition mechanism can be enabled once again. The step to first disable the repetition mechanism is required in case X_COMPARE is identical to X_ACTUAL during the write access to X_COMPARE, as this also triggers the repetition mechanism.

StallGuard2 Load Measurement

To fit different motor control schemes, the TMC5221/TMC5222 offer two types of StallGuard® sensorless load detection schemes, covering the two basic chopper modes. StallGuard2 works in SpreadCycle operation, while StallGuard4 is optimized for StealthChop2 operation.

StallGuard2 provides an accurate measurement of the load on the motor. It can be used for stall detection as well as other uses at loads below those which stall the motor, such as CoolStep load-adaptive current reduction. The StallGuard2 measurement value changes linearly over a wide range of load, velocity, and current settings. As the load on the motor increases, the StallGuard2 value (SG_RESULT) decreases, as shown in *Figure 35*. Tuning is required to properly detect stalls. Set the StallGuard2 threshold (SGTHRS) so that SG_RESULT reaches 0 (or near to 0) when the motor is overloaded/stalls. *Table 16* lists the related parameters.

Tip: To use StallGuard2 and CoolStep, the StallGuard2 sensitivity should first be tuned using the SGT setting.

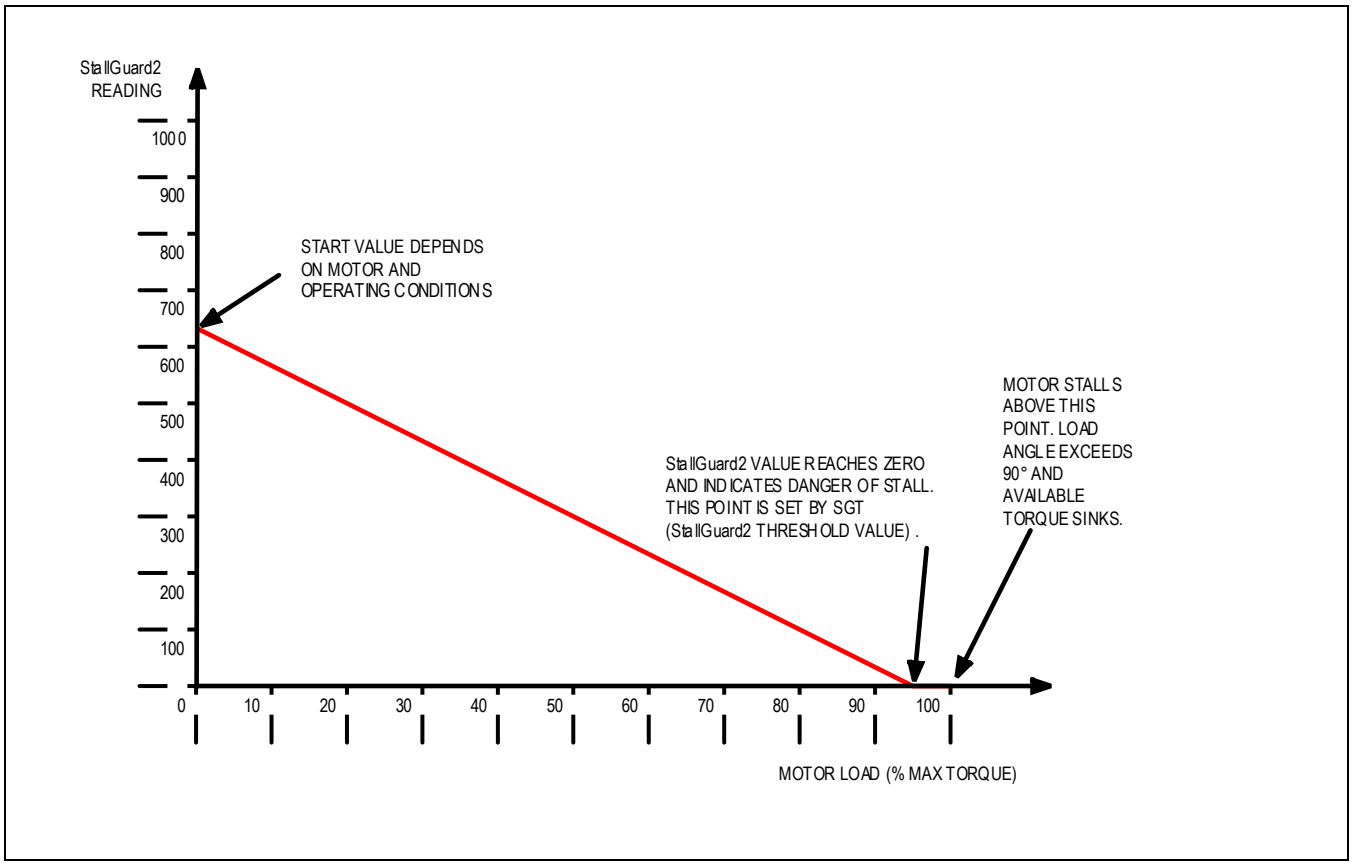


Figure 35. Function Principle of StallGuard2

Table 16. StallGuard2-Related Parameters

PARAMETER	DESCRIPTION	SETTING	NOTES
SGT	With SGT, a positive or negative offset can be added to the StallGuard2 result. A higher value increases the allowed dynamic range of the StallGuard2 result and decreases the sensitivity. A typical goal of StallGuard2 tuning is to shift the StallGuard2 result in a way that allows for a maximum dynamic range but still ensures the StallGuard2 result safely reaches a value of zero before the motor stalls.	0	Indifferent value
		+1 to +63	Less sensitivity
		-1 to -64	Higher sensitivity
SFILT		0	Standard mode

	Enables the StallGuard2 filter for more precision of the measurement. If set, reduces the measurement frequency to one measurement per electrical period of the motor (four full steps).	1	Filtered mode
SG_RESULT	This is the StallGuard2 result. A higher reading indicates less mechanical load. A lower reading indicates a higher load, and thus, a higher load angle. Tune the SGT setting to show a SG_RESULT reading of approximately 0 to 100 at maximum load before motor stall.	0 to 1023	0: Highest load Low value: high load High value: less load

Tuning StallGuard2 Threshold SGT

The StallGuard2 value SG_RESULT is affected by motor-specific characteristics and application-specific demands on load, velocity, supply voltage, and current level. Therefore, the easiest way to tune the StallGuard2 threshold SGT for a specific motor type and operating conditions is interactive tuning in the actual application.

Initial procedure for tuning StallGuard2 SGT:

- Operate the motor at the normal operation velocity, supply voltage, and current setting for the application and monitor SG_RESULT.
- Apply a slowly increasing mechanical load to the motor. If the motor stalls before SG_RESULT reaches zero, decrease SGT. If SG_RESULT reaches zero before the motor stalls, increase SGT. A good SGT starting value is zero. SGT is signed. Therefore, it can have negative or positive values.
- Next, enable sg_stop and make sure the motor is safely stopped whenever it is stalled. Increase SGT if the motor is stopped before a stall occurs. Restart the motor by disabling SG_STOP or by clearing EVENT_STOP_SG in the RAMP_STAT register (write 1 to clear).
- The optimum setting is reached when SG_RESULT is between 0 and approximately 100 at an increasing load shortly before the motor stall. SG_RESULT increases by 100 or more without a load. In most cases, SGT can be tuned for a certain motion velocity or a velocity range. Make sure the setting works reliably in a certain range (example, 80% to 120% of desired velocity) and also under extreme motor conditions (lowest and highest applicable temperature).

Optional procedure allowing automatic tuning of SGT:

The SGT setting is a factor, which compensates the StallGuard2 measurement for resistive losses inside the motor. At standstill and very low velocities, resistive losses are the main factor for the balance of energy in the motor because the mechanical power is zero or near to zero. Consequently, SGT can be set to an optimum at near zero velocity. This algorithm is especially useful for tuning SGT within the application to give the best result independent of environment conditions, motor stray, and more.

- Operate the motor at low velocity < 10RPM (that is, a few to a few full steps per second), and target operation current and supply voltage. In this velocity range, there is not much dependence of the SG_RESULT on the motor load because the motor does not generate significant back-EMF. Therefore, the mechanical load does not make a big difference on the result.
- Switch on SFILT. Next, increase SGT starting from 0 to a value where SG_RESULT starts rising. With a high SGT, SG_RESULT rises up to the maximum value. Reduce again to the highest value where SG_RESULT stays at 0. Now, the SGT value is set as sensitively as possible. With the SG_RESULT increasing at higher velocities, there is useful stall detection.
- SG_RESULT goes to zero when the motor stalls and the ramp generator can be programmed to stop the motor upon a stall event by enabling SG_STOP in SW_MODE. Set TCOOLTHRS to match the lower velocity threshold, where StallGuard2 delivers a good result to use SG_STOP.

The upper velocity for the stall detection with this setting is determined by the velocity, where the motor back-EMF approaches the supply voltage, and the motor current starts dropping when further increasing velocity.

The system clock frequency affects SG_RESULT. An external crystal-stabilized clock should be used for applications that demand the highest performance. The power-supply voltage also affects SG_RESULT. Therefore, tighter regulation results in more accurate values. The SG_RESULT measurement has a high resolution and there are a few ways to enhance its accuracy, as described in the following sections.

Variable Velocity Limits TCOOLTHRS and THIGH

The SGT setting chosen as a result of the previously described SGT tuning can be used for a certain velocity range. Outside this range, a stall may not be detected safely and CoolStep might not give the optimum result, as shown in *Figure 36*.

In many applications, operation at or near a single operation point is used most of the time and a single setting is sufficient. The driver provides a lower and an upper velocity threshold to match this. The stall detection is disabled outside the determined operation point, example, during he acceleration phases preceding a sensorless homing procedure when setting TCOOLTHRS to a matching value. An upper limit can be specified by THIGH. The velocity limits VHIGH and VCOOLTHRS are determined by the settings THIGH and TCOOLTHRS.

In some applications, a velocity-dependent tuning of the SGT value can be expedient, using a small number of support points and linear interpolation.

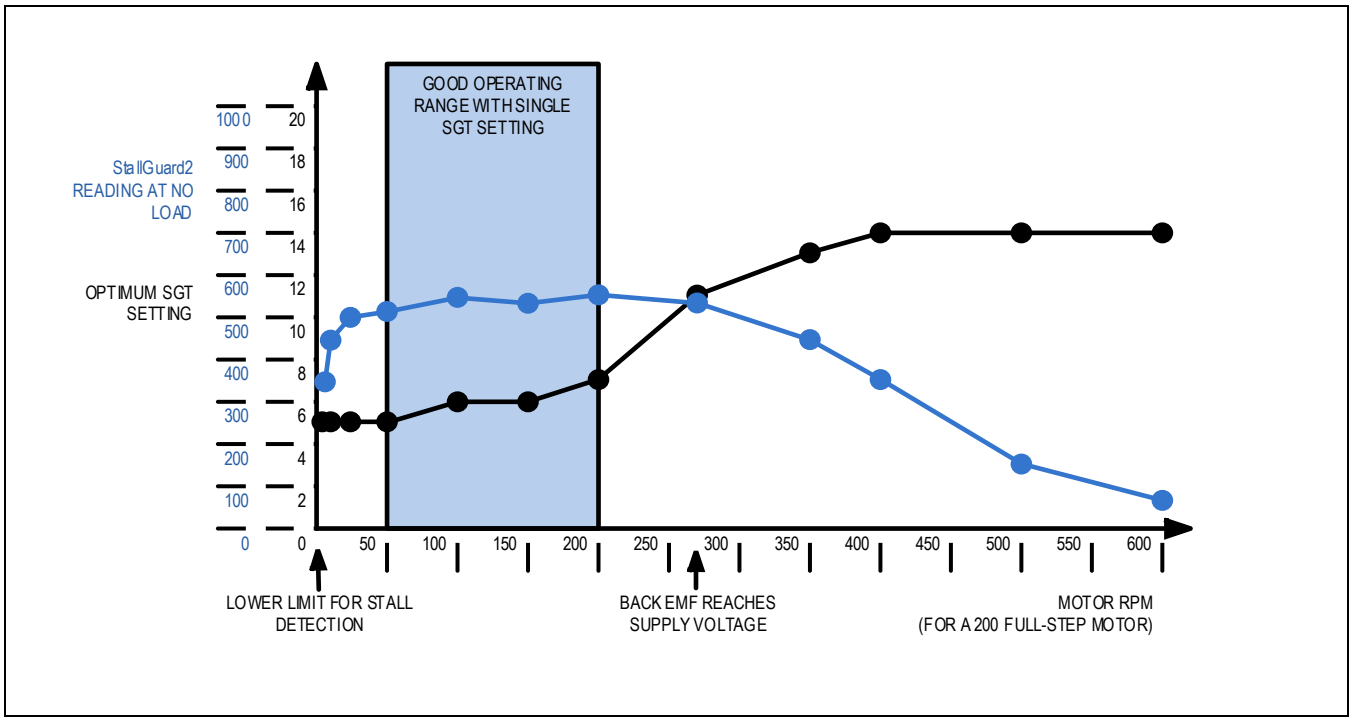


Figure 36. Optimum SGT Setting and StallGuard2 Reading with an Example Motor

Small Motors with High Torque Ripple and Resonance

Motors with a high detent torque show an increased variation of the StallGuard2 measurement value SG_RESULT with varying motor currents, especially at low currents. For these motors, the current dependency should be checked for best results.

Temperature Dependence of Motor Coil Resistance

Motors working over a wide temperature range may require temperature correction because motor-coil resistance increases with rising temperature. This can be corrected as a linear reduction of SG_RESULT at increasing temperature as motor efficiency is reduced.

Accuracy and Reproducibility of StallGuard2 Measurement

In a production environment, it may be desirable to use a fixed SGT value within an application for one motor type. Most of the unit-to-unit variation in StallGuard2 measurements results from manufacturing tolerances in motor construction. The measurement error of StallGuard2, provided that all other parameters remain stable, can be as low as:

StallGuard2 measurement error = ± max(1, |SGT|)

StallGuard2 Update Rate and Filter

The StallGuard2 measurement value SG_RESULT is updated with each full step of the motor. This is enough to safely detect a stall because a stall always means the loss of four full steps. In a practical application, especially when using

CoolStep, a more precise measurement might be more important than an update for each full step because the mechanical load never changes instantaneously from one step to the next. For these applications, the SFILT bit enables a filtering function over four load measurements. The filter should always be enabled when high-precision measurement is required. It compensates for variations in motor construction, for example, due to the misalignment of the phase A to phase B magnets. The filter should be disabled when rapid response to increasing load is required and for best results of sensorless homing using StallGuard2.

Detecting a Motor Stall

For best stall detection, work without StallGuard2 filtering (SFILT = 0). To safely detect a motor stall, the stall threshold must be determined using a specific SGT setting. Therefore, the maximum load must be determined, which the motor can drive without stalling. At the same time, monitor the SG_RESULT value at this load, example, some value within the range 0 to 100. The stall threshold must be a value safely within the operating limits to allow for parameter stray. The response at an SGT setting at or near 0 gives some idea on the quality of the signal: Check the SG_RESULT value without load and with maximum load. They should show a difference of at least 100 or a few 100, which is largely compared to the offset. If the SGT value is set so that a reading of 0 occurs at maximum motor load, the stall can be automatically detected to issue a motor stop. In the moment of the step resulting in a step loss, the lowest reading is visible. After the step loss, the motor vibrates and shows a higher SG_RESULT reading.

Homing with StallGuard2

The homing of a linear drive requires moving the motor into the direction of a hard stop. As StallGuard2 needs a certain velocity to work (as set by TCOOLTHRS), make sure the start point is far enough from the hard stop to provide the distance required for the acceleration phase. After setting up SGT and the ramp generator registers, start a motion into the direction of the hard stop and activate the stop-on-stall function (set SG_STOP in SW_MODE). Once a stall is detected, the ramp generator stops the motion and sets VACTUAL to zero, stopping the motor. The stop condition is also indicated by the STALLGUARD flag in DRV_STATUS. After setting up new motion parameters to prevent the motor from restarting right away, StallGuard2 can be disabled, or the motor can be re-enabled by clearing the EVENT_STOP_SG flag in the RAMP_STAT register. Clearing the EVENT_STOP_SG flag in RAMP_STAT restarts the motor after the expiration of TZEROWAIT when the motion parameters are not modified.

Limits of StallGuard2 Operation

StallGuard2 does not operate reliably at extreme motor velocities. Very low motor velocities (for many motors, less than 1rps) generate a low back-EMF and make the measurement unstable and dependent on environment conditions (example, temperature). The automatic tuning procedure described above can help with compensating for variations in conditions like motor temperature. Other conditions also lead to extreme settings of SGT and poor response of the measurement value SG_RESULT to the motor load.

Very high motor velocities, in which the full sinusoidal current is not driven into the motor coils, also leads to poor response. These velocities are typically characterized by the motor back-EMF reaching the supply voltage.

StallGuard4 Load Measurement

StallGuard4 is optimized for operation with StealthChop2, while its predecessor StallGuard2 works with SpreadCycle. The function of both is similar. Both deliver a load value, going from a high value at low load to a low value at high load. While StallGuard2 is tuned to show a zero reading for stall detection, StallGuard4 uses a comparison value to trigger stall detection, rather than shifting the measurement result by applying an offset.

StallGuard4 provides an accurate measurement of the load on the motor and can be used for stall detection, load estimation, as well as for CoolStep load-adaptive current reduction. The StallGuard4 measurement value changes linearly over a wide range of load, velocity, and current settings, as shown in *Figure 37*. When approaching maximum motor load, the value goes down to a motor-specific lower value. This corresponds to a load angle of 90° between the magnetic field of the coils and magnets in the rotor. This is also the most energy-efficient point of operation for the motor.

To use StallGuard4, check the sensitivity of the motor at edge conditions. *Table 17* lists the related parameters of StallGuard4.

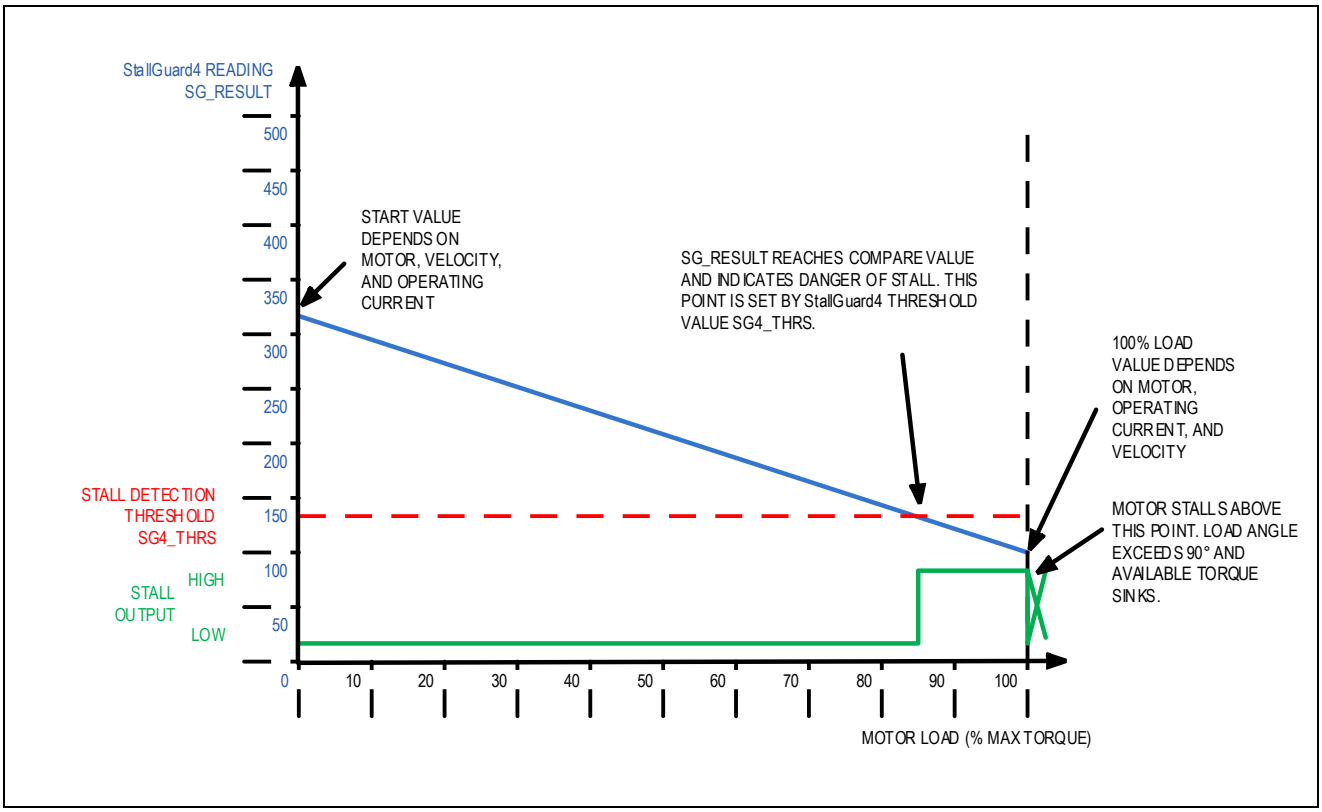


Figure 37. StallGuard4 Mode of Operation

Table 17. StallGuard4-Related Parameters

PARAMETER	DESCRIPTION	SETTING	NOTES
SG4_THRS	This value controls the StallGuard4 threshold level for stall detection. It compensates for motor-specific characteristics and controls sensitivity. A higher value gives a higher sensitivity. A higher value makes StallGuard4 more sensitive and requires less torque to indicate a stall. Default = 0.	0 to 511	This value is compared to SG4_RESULT. The stall output is active if SG4_RESULT falls below this value.
SG4_RESULT	This is the StallGuard4 result. A higher reading indicates less mechanical load. A lower reading indicates a higher load, and thus, a higher load angle. This value is generated independent of the enabling conditions like the actual chopper mode and velocity thresholds like VCOOLTHRS. The result is calculated from SG4_IND_x measurements, adding one bit for higher precision, and has a similar range as StallGuard2.	0 to 510	Low value: highest load. High value: low/no load.
SG4_IND_3 SG4_IND_2 SG4_IND_1 SG4_IND_0	Individual measurements for motor phase A falling (SG4_IND_0)/rising (SG4_IND_1) transition or phase B falling (SG4_IND_2)/rising (SG4_IND_3) transition. Individual measurements are available in filtered mode only (SG4_FILT_EN = 1). SG4_IND_0 covers all cases in unfiltered mode (SG4_FILT_EN = 0).	0 to 255	Low value: highest load. High value: low/no load.

SG4_FILT_EN	0: Unfiltered operation. SG4_RESULT updates with each full step. 1: Filtered operation. SG4_IND_x available. SG4_RESULT gives the average of last four SG4_IND_x measurements.	0/1	0: Filter off (default). 1: Filtered operation. SG4_IND values available.

Tuning StallGuard4

The StallGuard4 value SG4_RESULT is affected by motor-specific characteristics and application-specific demands on load, coil current, and velocity. Therefore, the easiest way to tune the StallGuard4 threshold SG4_THRS for a specific motor type and operating conditions is interactive tuning in the actual application.

The initial procedure for tuning StallGuard4 SG4_THRS is as follows:

- Operate the motor at the normal operation velocity for the application and monitor SG4_RESULT.
- Apply a slowly increasing mechanical load to the motor. Check the lowest value of SG4_RESULT before the motor stalls. Use this value as the starting value for SG4_THRS.
- Next, monitor the StallGuard4 output signal through the DIAGx output (also set TCOOLTHRS to match the lower velocity limit for operation) and stop the motor when a pulse is seen on the respective output. Make sure the motor is safely stopped whenever it is stalled. Increase SG4_THRS if the motor is stopped before a stall occurs.
- The optimum setting is reached when a stall is safely detected and leads to a pulse at DIAGx at the moment where the stall occurs. In most cases, SG4_THRS can be tuned for a certain motion velocity or velocity range. Make sure the setting works reliably in a certain range (example, 80% to 120% of the desired velocity) and also under extreme motor conditions (lowest and highest applicable temperature).

The diagnostic output DIAGx is pulsed by StallGuard4 when SG4_RESULT falls below SG4_THRS. It is only enabled in the StealthChop2 mode and when TCOOLTHRS ≥ TSTEP > TPWMTHRS.

The external motion controller should react to a single pulse by stopping the motor, if desired. Set TCOOLTHRS to match the lower velocity threshold where StallGuard4 delivers a good result.

The SG4_RESULT measurement has a high resolution and there are a few ways to enhance its accuracy, as described in the following sections.

StallGuard4 Update Rate

The StallGuard4 measurement value SG4_RESULT is updated with each full step of the motor. This is enough to safely detect a stall because a stall always means the loss of four full steps.

StallGuard4 provides two options for measurement:

- SG4_FILT_EN = 0: A single measurement, updated after each full step and valid for each full step. This measurement allows he quickest reaction to load variations as SG4_RESULT is fully updated with each zero transmission of a coil voltage. Therefore, it is optimum for stall detection with a hard obstacle.
- SG4_FILT_EN = 1: In this mode, four individual signals are generated: SG4_IND_0, upon the falling zero transition of the cosine wave (coil A); SG4_IND_1, upon the rising zero transition of the cosine wave; SG4_IND_2, upon the falling zero transition of the sine-wave (coil B); SG4_IND_3, upon the rising zero transition of the sine-wave. The actual value for SG4_RESULT is the mean value of all four measurements and is updated once each full step. With this, each full step has an influence of only 25% on the overall result. This mode is perfect for the detection of soft obstacles or for the use of CoolStep on imprecise motors. In filtered mode, sensitivity to a sudden load increase (hard motor blockage) is reduced.

Detecting a Motor Stall

To safely detect a motor stall, the stall threshold must be determined using a specific SG4_THRS setting and a specific motor velocity or velocity range. Furthermore, the motor current setting has a certain influence and should not be modified once the optimum values are determined. Therefore, the maximum load must be determined, which the motor can drive without stalling for the given application. At the same time, monitor SG4_RESULT at this load. The stall threshold should be a value safely within the operating limits to allow for parameter stray. More refined evaluation may also react to a change of SG4_RESULT rather than comparing to a fixed threshold. This rules out certain effects that influence the absolute value.

Limits of StallGuard4 Operation

StallGuard4 does not operate reliably at extreme motor velocities. Very low motor velocities (for many motors, less than 1rps) generate a low back-EMF and make the measurement unstable and dependent on environment conditions such as temperature. Other conditions also lead to a poor response of the measurement value SG4_RESULT to the motor load. Very high motor velocities, in which the full sinusoidal current is not driven into the motor coils, also leads to poor response. These velocities are typically characterized by the motor back-EMF exceeding the supply voltage.

Filtered StallGuard Value

In addition to StallGuard2/4 measurement results, there is the option to filter these results further and set a separate output signal threshold for this filter result. This PT1 low-pass filter takes the SG_RESULT register value as input and has a programmable length of 0/2/4/8/16/32/64 input updates (SG_PREWARN_FILTER in register SG_PREWARN_CONF). The filtered output value is available as SG_PREWARN_RESULT in register SG_PREWARN_CONF. A dedicated threshold value for this filtered StallGuard signal can be set with SG_PREWARN_THRSH in the same register. The resulting flag can be selected for output on DIAG0/1 (DIAG0/1_STALL_PREWARN in DIAG_CONF) and is available in the status register (STALLGUARD_PREWARN in DRV_STATUS).

The flag SG_PREWARN_RESULT_VALID informs if the filter value is valid after switching algorithms or after changing the filter length.

CoolStep Load Adaptive Current Scaling

CoolStep is an automatic smart-energy optimization for stepper motors based on the motor mechanical load, making them environmentally friendly. Depending on the actual chopper mode, CoolStep automatically uses the StallGuard4 load measurement result in StealthChop2, or StallGuard2 in SpreadCycle. Coolstep requires that either StallGuard2 or StallGuard4 (depending on the chopper mode being used) be tuned before use. A single tuning does not cover all the operating points.

Setting Up for CoolStep

CoolStep is controlled by several parameters, but the two parameters listed in *Table 18* are critical for understanding how it works. *Table 19* gives the additional parameters related to CoolStep.

Table 18. CoolStep Critical Parameters

PARAMETER	DESCRIPTION	RANGE	NOTES
SEMIN	4-bit unsigned integer that sets a lower threshold. If SG_RESULT goes below this threshold (indicating a large load), CoolStep increases the current to both coils. The 4-bit SEMIN value is scaled by 32 to cover the lower half of the range of the 10-bit SG_RESULT value.	0	Disable CoolStep (default).
		1 to 15	Threshold is SEMIN × 32.
SEMAX	4-bit unsigned integer that controls an upper threshold. If SG_RESULT is sampled equal to or above this threshold enough times (indicating a light load), CoolStep decreases the current to both coils. The upper threshold is (SEMIN + SEMAX + 1) × 32. Default = 0.	0 to 15	Threshold is (SEMIN + SEMAX + 1) × 32.

Figure 38 shows the operating regions of CoolStep:

- The black line represents the SG_RESULT measurement value.
- The blue line represents the mechanical load applied to the motor.
- The red line represents the current into the motor coils.

When the load increases, SG_RESULT falls below SEMIN × 32 and CoolStep increases the current. When the load decreases, SG_RESULT rises above (SEMIN + SEMAX + 1) × 32, and the current is reduced, as shown in *Figure 38*.

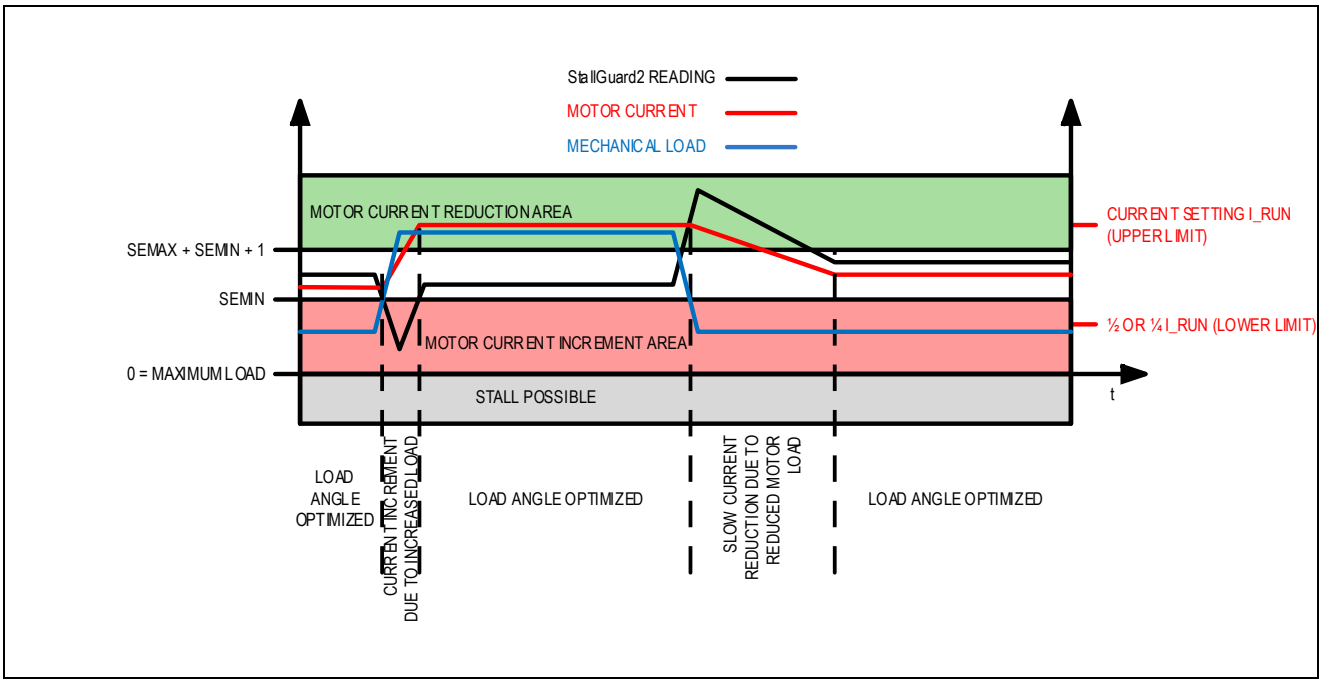


Figure 38. CoolStep Adapts Motor Current to the Load

Table 19. Additional CoolStep Parameters and Status Information

PARAMETER	DESCRIPTION	RANGE	NOTES
SEUP	Sets the current increment step. The motor current is incremented by this setting whenever a new StallGuard2 or StallGuard4 value is measured that lies below the lower threshold as set by SEMIN. Default = 0.	0 to 3	Step width of CS value CS_ACTUAL is 1, 2, 4, 8.
SEDN	Sets the number of StallGuard2/4 readings above the upper threshold necessary for each current decrement of the motor current. Default = 0.	0 to 3	Number of StallGuard2 measurements per decrement: 32, 8, 2, 1.
SEIMIN	Sets the lower motor current limit for CoolStep operation by scaling the IRUN current setting. When using StealthChop2, make sure to operate well above the minimum motor current as determined for StealthChop2 current regulation, especially when a reduction down to 25% is desired. Default = 0.	0	0: 1/2 of IRUN (when used with StealthChop2 requires IRUN ≥ 16).
		1	1: 1/4 of IRUN (when used with StealthChop2 requires IRUN ≥ 28).
TCOOLTHRS	Lower velocity threshold for switching on CoolStep. Below this velocity, CoolStep is disabled. Adapt to the lower limit of the velocity range where StallGuard2 gives a stable result. Tip: Can be adapted to disable CoolStep during the acceleration and deceleration phase by setting VCOOLTHRS identical to VMAX.	1 to 2^20 - 1	Specifies lower CoolStep velocity by comparing the threshold value to TSTEP.

	Default = 0.		
THIGH	Upper velocity threshold value for CoolStep. Above this velocity, CoolStep is disabled. Adapt to the velocity range where StallGuard2/4 gives a stable result. Default = 0.	1 to 2^20 - 1	Also controls additional functions like switching to full stepping.
CS_ACTUAL	This status value provides the actual motor current scale as controlled by CoolStep. The value goes up to the IRUN value and down to the portion of IRUN, as specified by SEIMIN.	0 to 31	1/32, 2/32, to 32/32, read-only.

Tuning CoolStep

Before tuning CoolStep in conjunction with SpreadCycle, first tune the StallGuard2 threshold level SGT, which affects the range of the load measurement value SG_RESULT. CoolStep uses SG_RESULT to operate the motor near the optimum load angle of +90°. In conjunction with StealthChop2, CoolStep uses SG4_RESULT. In this mode, the leveling is done through SEMIN.

The current increment speed is specified in SEUP and the current decrement speed is specified in SEDN. These can be tuned separately because these are triggered by different events that may need different responses. The encodings for these parameters allow the coil currents to be increased much more quickly than decreased because crossing the lower threshold is a more serious event that may require a faster response. If the response is too slow, the motor may stall. In contrast, a slow response to crossing the upper threshold does not risk anything more serious than missing an opportunity to save power.

CoolStep operates between the limits controlled by the current scale parameter IRUN and SEIMIN bit.

Response Time

For fast response to increasing motor load, use a high current increment step SEUP. If the motor load changes slowly, a lower current increment step can be used to avoid motor oscillations. If the filter controlled by SFILT is enabled, the measurement rate and regulation speed are cut by a factor of four.

Tip: The most common and most beneficial use is to adapt CoolStep for operation at the typical system target operation velocity, and to set the velocity thresholds accordingly. As the acceleration and deceleration values normally show high/dynamic values, the full motor current is required in these phases while having only a small contribution to the overall power consumption due to their short duration.

Low Velocity and Standby Operation

Because CoolStep is unable to measure the motor load in standstill and at very low RPM, a lower velocity threshold is provided. This threshold should be set to an application-specific default value. Below this threshold, the normal current setting using IRUN or IHOLD is valid. An upper threshold is provided by the VHIGH setting. The velocity limits VHIGH and VCOOLTHRS are determined by the settings THIGH and TCOOLTHRS.

Both thresholds can be set as a result of the StallGuard2 and StallGuard4 tuning process.

Sine-Wave Lookup Table

The TMC5221/TMC5222 provides a programmable lookup table for storing the microstep current waveform. By default (reset condition), the table is preprogrammed with a sine-wave, which is a good starting point for most stepper motors. Reprogramming the table to a motor-specific wave allows drastically-improved microstepping, especially with low-cost motors. The benefits are:

- Microstepping: Extremely improved with low-cost motors.
- Motor: Runs smoothly and quietly.
- Torque: Reduced mechanical resonances yield improved torque.
- Low-frequency motor noise: Reduced by adapting the sine and cosine wave shift for the actual motor's manufacturing tolerance.

The TMC5221/TMC5222 allow independent configuration of the sine-wave for each motor. For each motor, the phase shift between phase A and B is adjustable in the range of ±45°. In addition, there is an independent current scaler per motor phase for each motor (see the *Setting the Motor Current* section).

Microstep Table

To minimize the required memory and the amount of data to be programmed, only a quarter of the wave is stored. The internal microstep table maps the microstep wave from 0° to 90°. It is symmetrically extended to 360°. When reading out the table, the 10-bit microstep counter, MSCNT, addresses the fully-extended wave table. The table is stored in an incremental fashion using each 1-bit per entry. Therefore, only 256 bits (ofs00 to ofs255) are required to store the quarter wave. These bits are mapped to eight 32-bit registers. Each ofs bit controls the addition of an inclination Wx or Wx + 1 when advancing one step in the table. When Wx is 0, a 1-bit in the table at the actual microstep position means “add one” when advancing to the next microstep. As the wave can have a higher inclination than 1, the base inclinations Wx can be programmed to -1, 0, 1, or 2 using up to four flexible programmable segments within the quarter wave. This way, even a negative inclination can be realized. The four inclination segments are controlled by the position registers X1 to X3. Inclination segment 0 goes from microstep position 0 to X1 - 1 and its base inclination is controlled by W0; segment 1 goes from X1 to X2 - 1 with its base inclination controlled by W1, and more.

When modifying the wave, care must be taken to ensure a smooth and symmetrical zero transition when the quarter wave is expanded to a full wave. The maximum resulting swing of the wave should be adjusted to a range of -248 to +248 to give the best possible resolution while leaving headroom for the hysteresis-based chopper to add an offset. *Figure 39* gives an example.

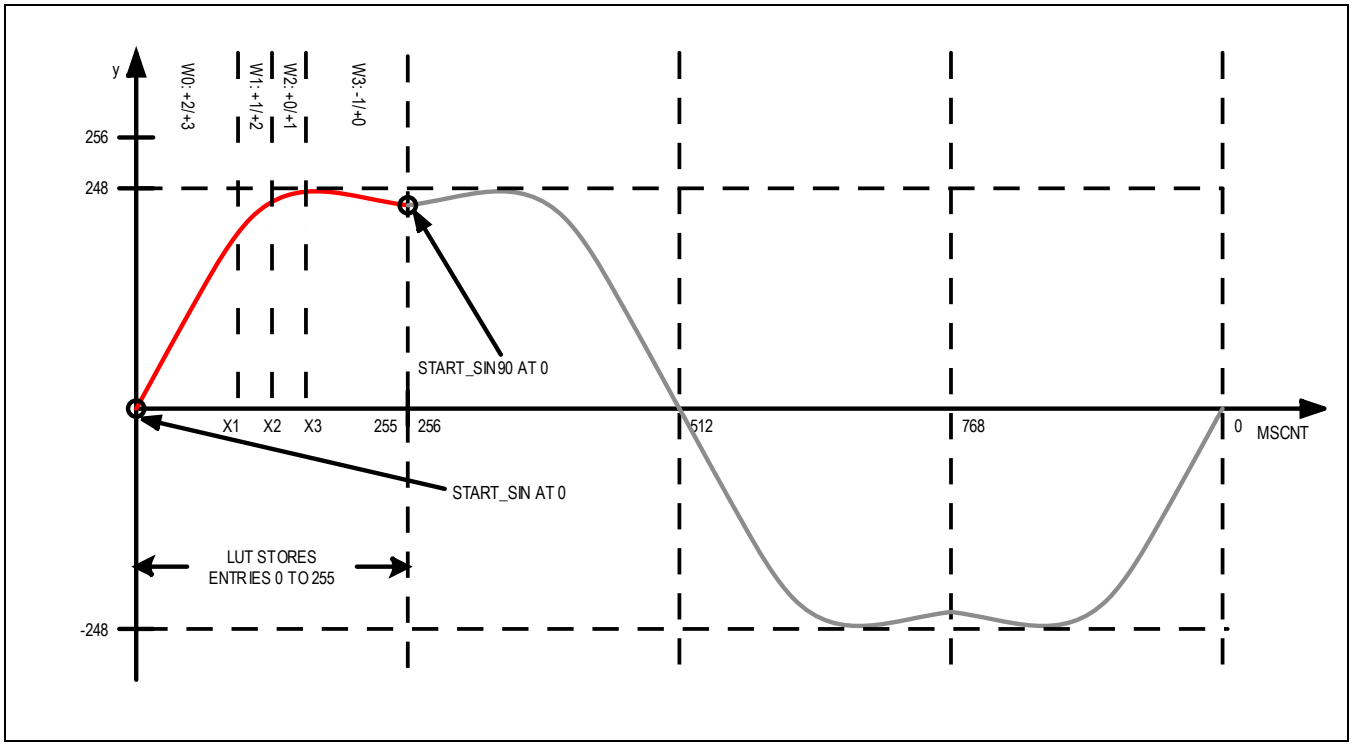


Figure 39. LUT Programming Example

When the microstep sequencer advances within the table, it calculates the actual current values for the motor coils with each microstep and stores them to the registers CUR_A and CUR_B. However, the incremental coding requires an absolute initialization, especially when the microstep table is modified. Therefore, CUR_A and CUR_B are initialized whenever MSCNT passes zero.

Matching the phase shift to the motor:

Two registers control the starting values of the tables, as shown in *Figure 40*.

- As the starting value at zero is not necessarily 0 (it might be 1 or 2), it can be programmed into the starting point register START_SIN.

- Similarly, the start of the second wave for the second motor coil must be stored in START_SIN90. This register stores the resulting table entry for a phase shift of 90° for a two-phase motor. To adapt for motor tolerances, the phase shift can be modified from 90° (256 microsteps) to anywhere between 45° and 135° by adding a microstep offset in the range of -127 to +127 (register OFFSET_SIN90). Motor tolerance requires moderate adaptations of a few steps to tens of steps, maximum. The required correction offset can be found out using StallGuard4 individual values SG4_IND and trimming the offset until both coils give a symmetrical result.

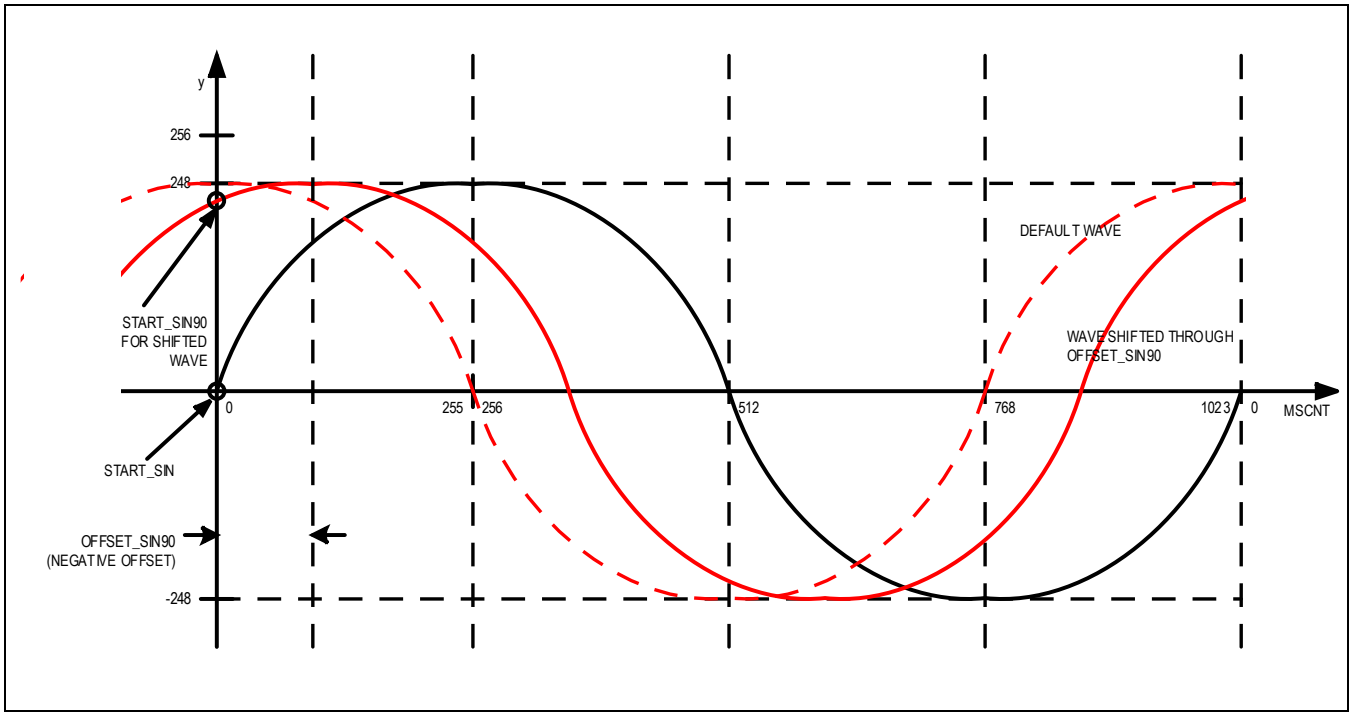


Figure 40. Shifting the Cosine Wave Through OFFSET_SIN90

The default table is a good base for configuring a microstep table. This is an initialization example for the reset default microstep table:

MSLUT[0] = %10101010101010101011010101010100 = 0xAAAAB554

MSLUT[1] = %01001010100101010101010010101010 = 0x4A9554AA

MSLUT[2] = %00100100010010010010100100101001 = 0x24492929

MSLUT[3] = %00010000000100000100001000100010 = 0x10104222

MSLUT[4] = %11111011111111111111111111111111 = 0xFBFFFFFF

MSLUT[5] = %10110101101110110111011101111101 = 0xB5BB777D

MSLUT[6] = %01001001001010010101010101010110 = 0x49295556

MSLUT[7] = %00000000010000000100001000100010 = 0x00404222

MSLUTSEL = 0xFFFF8056:

X1 = 128, X2 = 255, X3 = 255

W3 = %01, W2 = %01, W1 = %01, W0 = %10

MSLUTSTART = 0x00F70000:

START_SIN = 0, START_SIN90 = 247

To optimize the motor phase shift, run the motor at a medium velocity in StealthChop2 and set sg4_filt_en = 1. Adapt the phase offset to match the StallGuard4 results for phase A (SG4_IND_0 + SG4_IND_1) to phase B (SG4_IND_2 + SG4_IND_3).

If the phase A value is > phase B value, increment OFFSET_SIN90, otherwise decrement. Repeat until the best match is found.

Be sure to enter the correct value for START_SIN90. For an offset of -10 to +9, use START_SIN90 = 247. For an offset up to -17 or +17, use START_SIN90 = 246. START_SIN is always 0.

TriCoder (Back-EMF Sensorless Standstill Steploss Detection)

The TriCoder function is a sensorless standstill steploss detection feature making use of the motor back-EMF (BEMF). The TriCoder allows the detection of motor motion while the motor is disabled (that is, no active current is driven into the motor coils). This feature is especially useful for devices where a holding current cannot be applied due to power saving requirements, but a certain amount of cogging torque or friction normally keeps the motor in position. If the motor is turned by an external force, its motion can be tracked. The result can be used to trigger a new homing sequence if the motor is moved, or to keep track of the number of steps done, and correcting for it later.

An alternative use is to employ the motor as an input device, example, for manual teach-in or for user-interaction.

The system allows the following precision:

- The principle requires a certain minimum BEMF voltage generated by the motor to detect motion. Depending on the motor, this required BEMF level is easily reached in the range of a few RPM, and thus, the occurrence of motion can be easily detected.
- During the start or end of the motion, one or a few steps can be missed or counted incorrectly. The relevance of this must be checked with the actual motor and application mechanics.
- The system counts in multiples of one full step. By setting a scaling factor, the modified motor position can be directly tracked.

TriCoder BEMF Decoder Principle of Operation

The TriCoder BEMF decoder allows tracking a motor motion while the motor is disabled, or when the motor is stopped using passive braking through low-side drivers. To detect motor motion, the driver IC checks the voltage on the motor coils and compares it to a set of programmable hysteresis thresholds. *Figure 41* shows the principal mode of operation.

The detector circuit uses a combination of programmable hysteresis thresholds as well as bandwidth limiting to avoid false triggering, example, due to voltage spikes coupled into the motor lines.

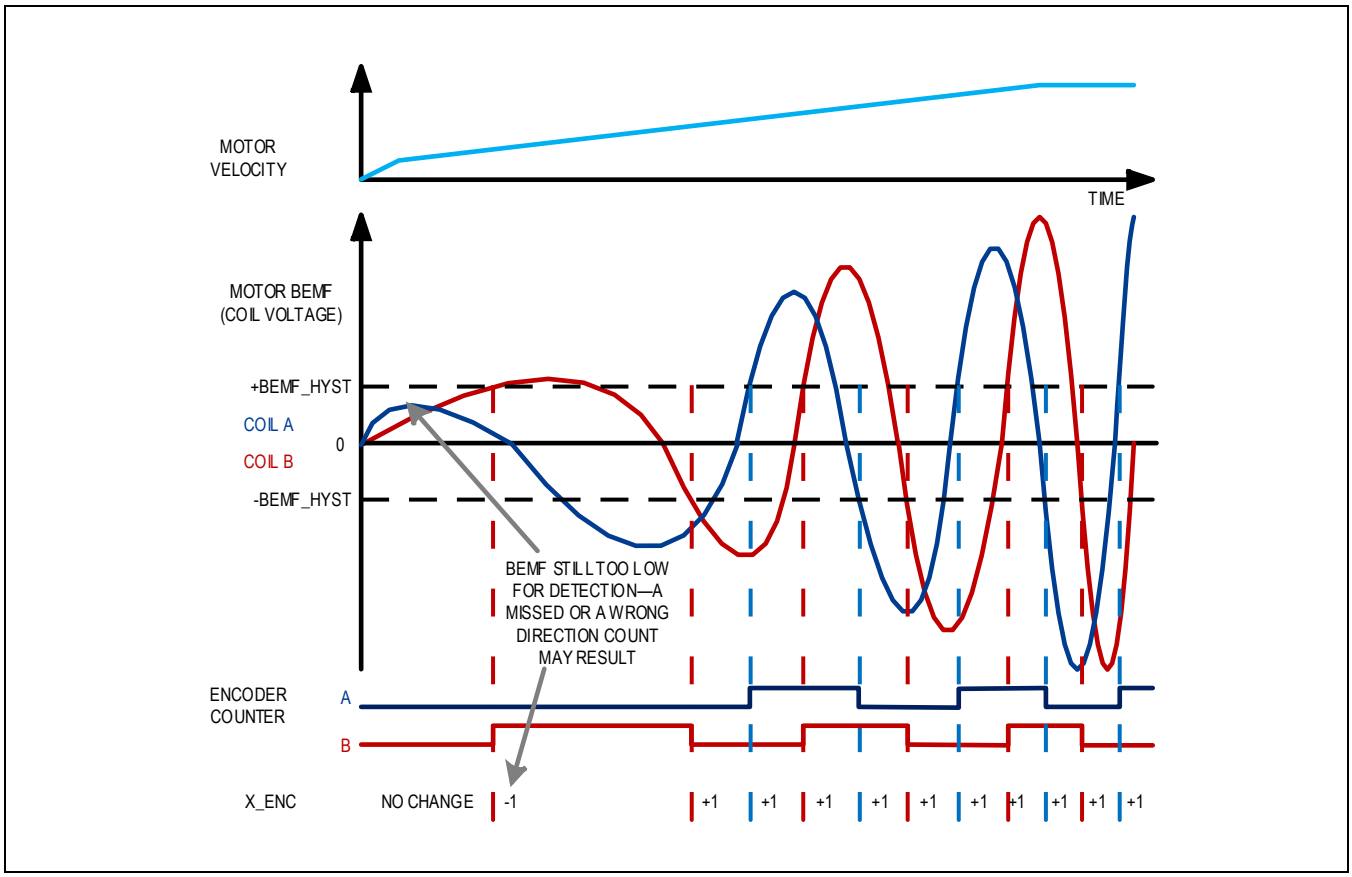


Figure 41. Detection of Motor Movement

Motor and Velocity Requirements for TriCoder Operation

Check the motor back-EMF voltage to determine the lowest velocity that can be detected using a certain hysteresis setting.

$$C_{BEMF}\left|\frac{V}{\frac{rad}{s}}\right| = \frac{\text{Holding Torque[Nm]}}{2 \times \text{ICOILNOM[A]}}$$

$$U_{BMEopen}[V] = \frac{\text{HoldingTorque[Nm]}}{2 \times \text{ICOILNOM[A]}} \times \frac{2\pi \times \text{Velocity[RPM]}}{60}$$

In passive braking mode, the motor is electrically dampened by shorting out its coils.

Consequently, TriCoder sensitivity to a certain motor velocity is reduced, as the generated back-EMF voltage first has to be converted into a motor current resulting from the motor coil resistance.

This current again is transferred back into a voltage in the $R_{DS(ON)}$ of the low-side MOSFET switches within the driver. For best results in this mode, choose the lowest power stage current setting to increase $R_{DS(ON)}$ vs. coil resistance.

$$U_{BEMFbrake}[V] = \frac{\text{Holding Torque[Nm]}}{2 \times \text{ICOILNOM[A]}} \times \frac{2\pi \text{Velocity[RPM]}}{60} \times \frac{R_{DS(ON)}}{R_{DS(ON)}+R_{COIL}}$$

Time to Enable

To operate the motor, the motor must be in the freewheeling or passive braking mode. Depending on the preceding operation, a certain delay time might be required to ensure the remaining motor coil current is deceased. Following the detection of a motor stop, the motor current ramps down to 0 (controlled by IHOLDDELAY settings). Set the standstill detection time STANDSTILL_TIME as required by the lowest operation velocity. In the case of a fast IHOLDDELAY setting (0 = instantaneous power-down) and a highly inductive motor, an additional time of a few hundred microseconds to a few

milliseconds may be required to decay the remaining motor current (BEMF_BLANK_TIME) by feeding it back to the power supply. Any remaining current might look like a motor BEMF and can otherwise lead to a step detection.

TriCoder Settings

Table 20. Settings Required for TriCoder Operation

CONFIGURATION	REGISTER	SETTINGS
Settings required to automatically enable the TriCoder, whenever the motor is in the freewheeling mode.	PWMCONF. PWM_FREEWHEEL	Set to 1 (freewheel) or 2 (passive braking).
	With StealthChop: IHOLD	Set to 0, to enable freewheeling options during standstill. CS_ACTUAL must reach 0 to enable these options.
	With SpreadCycle: TOFF	Set TOFF = 0 to disable the motor and allow freewheeling. **Hint:** This setting is not automatical, but must be done whenever the motor is in standstill and TriCoder is desired.
	In Low-Power Quiescent Mode: BEMF_CONF. QSC_TRICODER_EN	Set QSC_TRICODER_EN = 1 to enable TriCoder during the low-power quiescent mode.
Control of TriCoder Operation	BEMF_CONF. BEMF_BLANK_TIME	Set this time safely to allow the motor coil current resulting from the previous motor operation to decay to 0 before the motor BEMF is monitored. Increase if a false motion detection occurs directly after going to motor standstill. Counts in multiple of 2^{14} clock cycles.
	BEMF_CONF. BEMF_HYST	(0 to 7): Hysteresis 10, 25, 50, 75, 100, 150, 200, and 250mV (typ).
	DRV_CONF. STANDSTILL_TIME	Shorten the time to standstill detection to speed up the ramp down of the motor current to 0 as controlled by the IHOLD, IRUN, and IHOLDDELAY settings. The default is 2^{20} clock cycles.
	BEMF_CONF. BEMF_FILTER_SEL	This setting limits the input bandwidth for A and B channels, and with this, the encoder count rate to the quadruple of the resulting frequency (when using the internal f_{CLK} with ~12.5MHz: 2kHz, 4kHz, 8kHz, 16kHz). Set well above the expected maximum count rate. Digital filter time for decoded BEMF signals. 0: f_{CLK}/24600, 1: f_{CLK}/12300, 2: f_{CLK}/6150, 3: f_{CLK}/2870
	BEMF_CONF. BEMF_INCREMENT	Set to match the X_BEMF counter to the motor microstep resolution, example, 8 for 256 microstep setting. With this setting, each motor full step increases/decreases X_BEMF by 256. Counting direction can be changed by BEMF_DIR, if needed.
	X_BEMF	Can be set to 0 or X_ACTUAL before TriCoder is activated. Read out the step difference or updated motor position while TriCoder is in operation. In case BEMF_OVERWRITE is used, it makes sense to keep the value at 0 instead of X_ACTUAL. This value is cleared on an automatic safe position interrupt and stored in X_BEMF_LATCH for readback. This is done to avoid multiple additions of detected position displacement with multiple interrupts.

Diagnostic and Status Outputs

Based on their configuration, the two diagnostics outputs, DIAG0 and DIAG1, deliver a range of status and interrupt signals to the host to monitor and react on certain driver and motion controller conditions. With the DIAG_CONF bits, DIAG0_NOD_PP and DIAG1_NOD_PP, either an open-drain (OD, active-low, default setting) output signal can be chosen or an active-high, push-pull (PP) output signal. When using the open-drain output, an external pull-up resistor in

the 4.7kΩ to 33kΩ range is required. When the DIAG pins are configured for push-pull operation, the DIAG_CONF bits, DIAG0_INVPP and DIAG1_INVPP, can be used to select the polarity of the push-pull output signals from active high to active low. They do not affect the OD-mode.

Figure 42 shows the DIAG0/DIAG1 circuit. DIAG0/DIAG1 is active upon a reset condition. The register DIAG_CONF is used to select the signals and flags related to the internal motion controller, driver status, and error flags for the DIAG0/DIAG1 outputs. The signals available for selection are the same for the DIAG0 and DIAG1 output, and can be enabled individually and separately for each output for maximum flexibility.

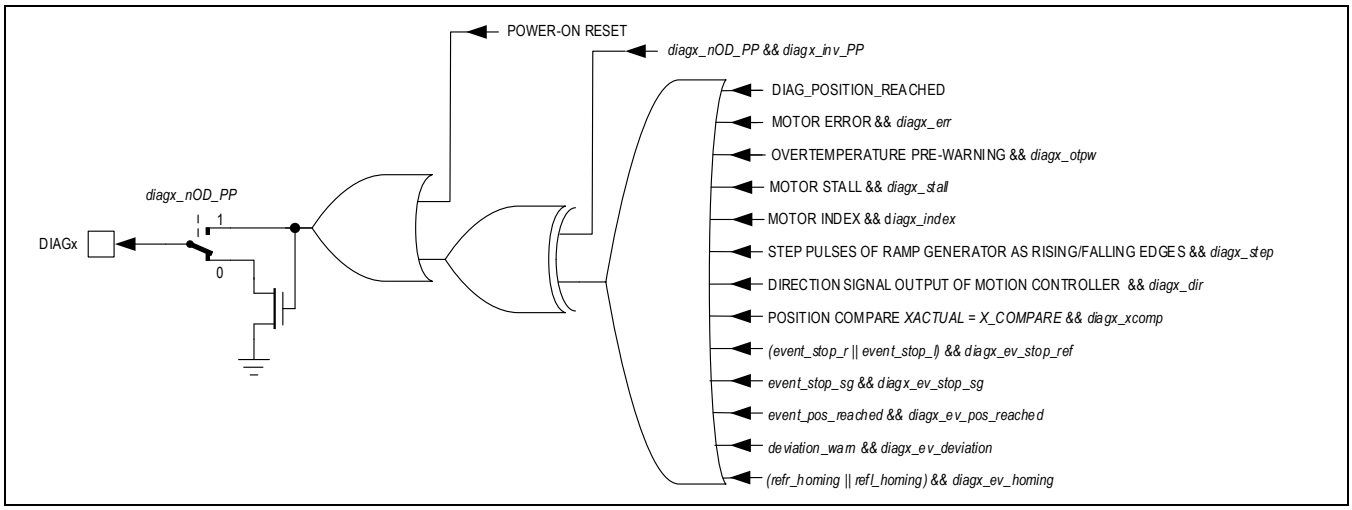


Figure 42. DIAGx (DIAG0/DIAG1) Output Circuit

Attention: The duration of the index pulse corresponds to the duration of the micro step. When working without interpolation at less than 256 micro steps, the index time goes down to two CLK clock cycles

Chip Emergency Stop and Automatic Safe Position Interrupt

The TMC5221/TMC5222 come with different functions, allowing direct response to a condition signaled by an external source, by stopping the motor (emergency stop), or positioning to a certain target position (automatic safe position).

Table 21 gives an overview on the operation modes and available emergency stop modes. The maximum IC power consumption refers only to the IC itself and does not take into account the actual motor current.

Table 21. Chip Mode Comparison

MODE	ENTRY CONDITION	EXIT CONDITION	REGISTER CONTENT	TYP IC POWER CONSUMPTION	SPI	MOTOR POWER
Emergency stop over SPI/I²C.	GCONF register (0x0): DRV_EN_SW = 0	GCONF register (0x0): DRV_EN_SW = 1	No change	**1.9mA**	Active	Driver disabled immediately.
Configurable external emergency stop with standstill options.	To enable this mode: GCONF register (0x0): STOP_ENABLE = 1. To trigger and keep this mode, pin REFL or REFR(depends on configuration in GCONF [5]) in pulled to and held at 1. To configure this mode, IHOLD_IRUN register (0x12):	Pin REFR or REFL pulled to 0 (depends on configuration).	No change	**Depending on IHOLD**	Active	Driver active but motor stops with configured hold current or StealthChop2 freewheel option.

	IHOLD and IHOLD_DELAY for current reduction. PWMCONF register (0x3C): PWM_FREEWHEEL.					
Low-power quiescence mode	To enable this mode, GCONF register (0x0): QSC_STS_ENA = 1. To trigger this mode, once the motor driver is in standstill, the IC goes into quiescence mode. Mode active indicator (IOIN register) (0x4): Check if bit-25 QSC = 1. Device must be in standstill (check bit-31 STST in DRV_STATUS register (0x3B)). To further configure the quiescence mode, BEMF_CONF register (0x31): QSC_TRICODER_EN = 1 to enable encoder during quiescence mode.	GCONF register (0x0): QSC_STS_ENA = 0	No change	23µA	Active, but longer break between datagrams (~10 us of CSn high time).	Driver disabled.
Reset	Low pulse > 30µs on SLEEPN.	SLEEPN = 1	Reset to defaults.	~1µA (while in reset).	Not active	Driver disabled after SLEEPN input filter delay (~50µs).
Sleep mode	SLEEPN pulled low permanently.	SLEEPN = 1	Reset to defaults.	~1µA	Not active	Driver disabled after SLEEPN input filter delay (~50µs).

Emergency Stop Using SPI (TMC5221) or I²C (TMC5222)

The TMC5221/TMC5222 provide a register-controlled option to switch off all power MOSFETs of the driver stage. This allows putting the motor into freewheeling. This can be done by setting the GCONF register bit DRV_EN_SW to 0.

Emergency Stop Using DRV_EN pin

The TMC5221/TMC5222 output stage can be tristated at any time by driving the DRV_EN pin low. The DRV_EN asynchronous signal bypasses all the digital blocks of the TMC5221/TMC5222 and goes directly to the driver output stage. Driving this pin low switches off the motor driver so that the torque is immediately removed from the motor. The DRV_EN pin can be used to implement a hardware-based safety channel, which does not rely on the MCU software and serial interface communication. When the DRV_EN pin is driven low, the content of the registers is preserved, making possible for the external controller to read the TMC5221/TMC5222 registers through the serial interface. It is not recommended to drive the DRV_EN pin low, while the motor is in rotation.

Emergency Stop With Standstill Options

Some applications may require the driver to be put into a state with active holding current or with a passive braking mode. This is possible by programming the input pin REFL, respectively, REFR, to act as a motor stop and disable function. Set the GCONF bit STOP_ENABLE to activate this option. Choose the reference pin with GCONF REFR_STOP.

Whenever the pin REFL/REFR is pulled high and as long as it stays high, the motor stops and goes to the power-down state, as configured by IHOLD and IHOLD_DELAY (in the IHOLD_IRUN register) and StealthChop2 standstill options (PWM_FREEWHEEL field in the PWMCONF register) when StealthChop2 is in use. When the pin REFL/REFR is pulled low again, the motor driver returns to the default operational mode.

Automatic Safe Position Function

The TMC5221/TMC5222 feature an automatic safe position interrupt procedure. It allows positioning the motor to one of two stored positions, whenever an external signal triggers this function. Once parameterized, no CPU interaction is required to react to the external trigger signal. This reduces the reaction time and reduces the dependency on software functions.

- Respond to emergency signal, example, from the accelerometer (mobile system falling), temperature sensor (high/low threshold), and more, and bring the system to safe state.
- React to system supply detection, example, when the power supply drops below the critical level.
- Respond to system reset or watchdog timeout detection.
- Respond to external inputs while the CPU is in the sleep mode to bridge the wake-up time.

The safe position function is triggered by a positive edge on the REF_L or REF_R pin. It sets the ramp generator to the position mode using the positions stored in VIRTUAL_STOP_L for REF_L, respectively, VIRTUAL_STOP_R for REF_R as a target position. The ramping parameters are not modified. Any onging motion is overwritten and does not finish.

Additionally, the integration of the TriCoder feedback allows to automatically take into account position displacement caused by the external force, that occurred while the motor is not powered. In this case, the controller automatically updates the internal actual position XACTUAL by adding X_BEMF whenever starting the automatic safe position motion. This option is enabled by BEMF_OVERWRITE.

DIAG output pins can be configured to signal an active safety position interrupt event (REFL_HOMING/REFR_HOMING) to the controller. With both target positions enabled, even a simple back-and-forward movement can be implemented using the REFL/REFR inputs without com interface processor intervention. This is activated by enabling [16] of SW_MODE after driving to the VIRTUAL_STOP_L or VIRTUAL_STOP_R position.

Preparation

To prepare operation, configure the ramp generator for motion in the position mode. At least, VMAX, AMAX, DMAX, and VSTOP must be set. Enable the REF_L pin interrupt response by bit EN_REFL_HOMING in the register SW_MODE. Enable the REF_R pin interrupt response using EN_REFR_HOMING.

Hint: The safe position function cannot be used in combination with the Step and Direction operation (ensure SD = 0), active stop switch function, or the virtual stop functionality (EN_VIRTUAL_STOP_L and EN_VIRTUAL_STOP_R).

Internal Execution

Once a rising edge is detected on the configured REF_L/REF_R pin, the device switches the register RAMPMODE to position mode. In case the driver is in qsc mode, the controller exits standby and waits until CS_ACTUAL reaches IHOLD. After ensuring the motor is powered, register XTARGET is overwritten with the value contained in VIRTUAL_STOP_L/R, whichever triggered the interrupt. This initiates motion to the stored target position. During exectution, any further interrupts are ignored until XTARGET reaches the preprogrammed position!

The interrupt execution can be checked in the register RAMP_STAT. The write1toClear-bits REFL_HOMING and REFR_HOMING display an interrupt is triggered by the REF_L or REF_R pin. Clear these bits after an interrupt is issued to track any new safe position action.

While the device is in low-power standby (qsc mode), an interrupt event wakes the device by clearing the bit QSC_STS_ENA as well as enabling the driver DRV_EN_SW. This assumes that pin DRV_EN is kept at the V_{CCIO} level as the driver is continuously disabled otherwise. (See also section LOW POWER MODES).

As CS_ACTUAL is 0 during the qsc mode, XTARGET is updated after CS_ACTUAL ≥ IHOLD again. Maximum wake-up time can be calculated from 186µs + 732 × t_clk (internal or external) + IRUN_DELAY setting. This means there are at least 230µs (typ.) for wake-up + IRUN_DELAY setting (512 × t_{clk} × IRUN_DELAY × (IHOLD-1)) (if IHOLD_DELAY≠0) to wait until the position is updated. If IHOLD_DELAY = 0, then IHOLD is reached immediately after the wake-up time and XTARGET is updated after that.

Early Termination

Writing a new value to XTARGET during automatic safe position action terminates the intterupt and initiates motion to the new target position. Optionally, terminate the interrupt prior to reaching the target position by disabling EN_REFL_HOMING, respectively, EN_REFR_HOMING. Only when correctly terminated, a new interrupt can be issued.

Note: The automatic safe position interrupt does not enforce positioning mode during its execution. The change to positioning mode is a single event.

Hint: Do not change the target positions VIRTUAL_STOP_L and VIRTUAL_STOP_R while an interrupt is executed as this hinders the interrupt from terminating. To ensure this, disable EN_REFL_HOMING and EN_REFR_HOMING whenever setting new target positions.

Safe Limits for Motor Current and Velocity

The TMC5221/TMC5222 offer registers that limit the maximum value for current and velocity settings. As with all registers, these can be preprogrammed to application-specific limits and locked using the OTP. When trying to change a limited register with a value exceeding the limit, the write access data is clipped to the value of the limit, effectively writing the limiting value to the register.

- Add a safety layer for wearables, robotics, and delicate mechanical machines.
- Protect the hardware, PCB, and mechanics against misconfiguration due to software malfunction.
- Protect against hacking and tuning.

Table 22 gives an overview of the available limiting registers.

Table 22. Available Limit Values

REGISTER	ADDRESS	USAGE	FIXED/LIMIT (DEFAULT)	RELATED REGISTER/ REGISTER VALUE
VMAX_LIMIT	0x44	Limit the highest possible velocity for the eight-point ramp generator to this value. Example: VMAX_LIMIT = 10000 In case the software writes a higher value to VMAX, VMAX is set to VMAX_LIMIT = 10000 instead.	0x7FFFFF (maximum value)	VMAX (0x24)
LIMIT_VALUES [7:0]: GLOBAL_SCALER_LIMIT	0x45	Limits the current setting of GLOBAL_SCALER_A and GLOBAL_SCALER_B values. With 0, the limit is disabled.	0x0 (maximum value)	GLOBAL_SCALER (0x6)
LIMIT_VALUES [9:8]: RREF_LIMIT	0x45	Limit coarse current setting.	0x3 (maximum value)	FS_SENSE in DRV_CONF (0x5)
LIMIT_VALUES [11:10]: GAIN_SCALER_LIMIT	0x45	Limit coarse current setting.	0x3 (maximum value)	FS_GAIN in DRV_CONF (0x5)

LIMIT_VALUES [20:16]: IRUN_LIMIT	0x45	Limits IRUN value. IRUN cannot be set above the IRUN_LIMIT.	0x1F (maximum value)	Limit for IRUN register IHOLD_IRUN (0x14).
LIMIT_VALUES [28:24]: IHOLD_LIMIT	0x45	Limits IHOLD value. IHOLD cannot be set above the IHOLD_LIMIT.	0x1F (maximum value)	Limit for IHOLD register IHOLD_IRUN (0x14).
VIRTUAL_STOP_L VIRTUAL_STOP_R		Motion limits for XACTUAL position.		

Using OTP

The TMC5221/TMC5222 come with a field-programmable OTP to preset the register values of bits and bitfields so there is no need to initially configure them using the communication interface.

Benefits

- Shorten power-up time by reducing or eliminating the demand for paramerization, especially for multi-node systems.
- Protect against the change of critical parameters by locking and limiting registers.
- Reduce CPU overhead and required memory space.
- Set device-address for mutiple nodes on a I^2C bus.

The preset values stored in OTP automatically are loaded into the register bank directly after device power-up or device reset. They overwrite default reset values. In addition, it is possible to lock most register bits, registers, and fields from write access through the interface. This adds protection against incidental later change. Further, bits 2,3, and 4 of the I^2C node address can be configured using the OTP to extend the system address range for up to 32 nodes.

The OTP memory is organized in a way that allows multiple changes to the initialization values in device life, until all OTP bits are used up. Therefore, it is also called a multiple times programmable (MTP) memory. For example, a configuration of up to 15 registers still leaves up to 64 OTP entries free, allowing four times fixing the complete parameter set, or several ten times fixing a partial set of registers. Further, OTP configuration can be tried out (OTP prototyping) before actually burning a new parameter set. However, typically, OTP cannot be programmed within the application, but in a defined programming environment, as certain conditions must be fulfilled for programming.

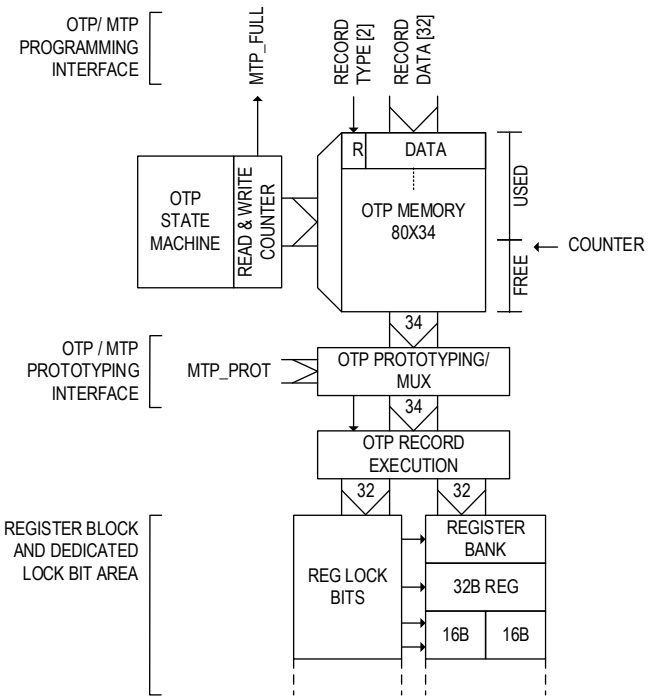


Figure 43. MTP Memory Organization

OTP Programming

OTP programming requires a defined supply voltage and clock frequency. An external voltage of 8.7V ± 0.13V (1.5%) must be supplied to the pin V_S. The voltage level can be checked by reading CP_CMPOUT with enabled window comparators. In addition, if an external clock is used, it must be set to 12.5 MHz (+-2.5%) during the programming procedure. Using the internal oscillator is fine in a temperature range from 15°C to 35°C. Programming must not be done in the low-power quiescent mode. To ensure sufficient stability and low ripple of the supply voltage, and a low device self-heating, it is recommended to disable the motor driver before programming.

Attention: Violation of the correct OTP programming requirements can lead to unsuccessful programming or device failure. Do not program while the motor is in operation.

Enter the OTP mode by writing password 0x12A7 into the register 0x7D. This enables the OTP register map and disables the functional register map. Leaving the OTP mode can be done by writing register 0x7D to 0 or any 16-bit value different from the password.

The OTP can be sequentially programmed with a number of records. Each record is identified by a 2-bit record type and contains up to 32-bit data. These sequentially are read and executed. This way, a later record can overwrite data written by a previous record, allowing modifications to the previously programmed content.

When in the OTP mode, MTP_RECORD is used to set a record type and data to be burnt. *Table 23* gives an overview of the address and function.

Table 23. MTP Record Types

RECORD TYPE	FTP REGISTER	LOCATION IN RECORD DATA [31:0]	DESCRIPTION
0x0	REGISTER_ADDRESS	Bits [6:0]	Define the TMC5221/TMC5222 (core) register address for the following record. Note that the address is incremented after each REGISTER_DATA and REGISTER_LOCK_BITS record to enable block wise access to multiple registers without writing a new REGISTER_ADDRESS record.
	REGISTER_WRITE_LOCK	Bit [7]	If this bit is set, the write access is blocked for all registers where the data is overwritten using the REGISTER_DATA records following this instruction. This protects all the registers written in the following REGISTER_DATA accesses against change through interface write access.
0x1	REGISTER_DATA	Bits [31:0]	Set the data to write to the TMC5221/TMC5222 register addressed by REGISTER_ADDRESS. REGISTER_ADDRESS auto increments to the next address after evaluating REGISTER_DATA.
0x2	REGISTER_LOCK_BITS	Bits [31:0]	Allows locking registers or parts of registers (bits, repectively, bit fields) against future access. Only valid for TMC5221/TMC5222 (core) registers that support the write lock option for each bit. If any bit in this record is set to 1, the corresponding bit or bit field of the TMC5221/TMC5222 core register selected with REGISTER_ADDRESS is set to read-only. For bit fields, set the corresponding LSB in REGISTER_LOCK_BIT to protect these against change. Do not use in parallel to REGISTER_WRITE_LOCK, as this protects the whole register with all bits and bit fields against a write access.
0x3	I2C_NODE_ADDRESS	Bits [2:0]	Set the bits [4:2] of the I^2C device address (only applicable for TMC5222).
	DISABLE_IREF_FAULT	Bit [3]	0: IREF fault disables the driver. 1: IREF fault is ignored by the IC.
	BLOCK_MTP_ACCESS	Bit [31]	Lock MTP. Burning this record with this bit set to 1, every further record burned to the OTP is ignored. Setting this bit to 1 effectively locks the MTP for any further changes.

Totally, the IC can store up to 80 of these records. For a new device, the memory space is empty. The internal state machine keeps track of the used and free entries. The counter is not accessible through the interface! When all OTP entries are used, the bit MTP_FULL signals that no further write access is possible.

Once a record is written into the MTP_RECORD field, it can be burnt into the OTP by setting the bit SRT_PROG in register MTP_Control to 1. The register MTP_STATUS signals a successfull result of the burning procedure by the bit DONE, otherwise MTP_STATUS provides additional information on why burning failed. The burnt values do not automatically apply directly after the burn procedure. Either a power cycle is required to activate them, or using bit MTP_RESTORE in MTP_Control to soft-reload the new values from OTP.

If the motor is operatated after a soft-reload, the window comparator in CP_CONTROL_2 must be deacitvated.

The TMC5221/TMC5222 allow testing OTP settings before burning these in a prototyping mode.

Example for changing the power-up default values of some MSLUT registers:

Ensure burning voltage 8.7V ± 0.13V (1.5%) on VS pin;
Write(0x7D, 0x12A7); // set the password to enter OTP mode
Write(0x21, 0x1B); //enable the window comparator and supply voltage checks
Read(0x22); //Check bits [2] ==0 and [0] == 1. This means the burning voltage is in the correct range.

//Writing the record at record type 0
Write(0x30, 0x8); //set MSLUT0 as starting address for consecutive writing with no write lock
Write(0x31, 0x0);
Write(0x32, 0x0);
Write(0x33, 0x0);
Write(0x34, 0x0); //virtual address 0x0 for the address offset

Write(0x10, 0x1); //Burn the record above in OTP
Read(0x11) until return is 0x80; //DONE bit is set and no other bit is set.

Write(0x30, 0x44); //set the default value of MSLUT0 to 0x11223344
Write(0x31, 0x33);
Write(0x32, 0x22);
Write(0x33, 0x11);
Write(0x34, 0x1); //virtual address 0x1 for data starting with the register address

Write(0x10, 0x1); //Burn the record above in OTP
Read(0x11) until return is 0x80; //DONE bit is set and no other bit is set.

//no need to rewrite the address here as it is the register following directly after MSLUT0
Write(0x30, 0x88); //set the default value of MSLUT1 to 0x55667788
Write(0x31, 0x77);
Write(0x32, 0x66);
Write(0x33, 0x55);
Write(0x34, 0x1); //virtual address 0x1 for data starting with the register address

Write(0x10, 0x1); //Burn the record above in OTP
Read(0x11) until return is 0x80; //DONE bit is set and no other bit is set.

//continue with MSLUT2 without writing virtual address 0x0 again OR choose a different address offset in order to set up
//non consecutive registers.

OTP Prototyping

Simulating the OTP entry is a way to test the determined values before burning them into the OTP. This kind of prototyping allows direct test of the intended sequence. It is not stored permamently and is lost after a power cycle, device reset, or restoring the OTP content. The protyping OTP sequence is applied immediately for each data or lock record written. In contrast to OTP programming, there are no specific requirements concerning supply voltage, clock frequency, or temperature during OTP prototyping.

Operate the device in active mode (no low-power quiescent mode). Prototyping is enabled by writing the OTP access password 0x12A7 into the register 0x7D and setting MTP_PROT_EN (bit-4) of the register MTP_CONTROL (0x10).

Other than OTP programming access, prototyping uses a bytewise protocol to simulate OTP record access. Therefore, more addresses are required to set all data to fill each of the four record types. The register MTP_PROT_ADDR (0x12) allows bytewise access to the MTP records, while data can be written to the selected MTP record register by writing MTP_PROT_WDATA (0x13). Once the MSBs of register data or lock map are written, the record is executed. Prototyping data is not written into the OTP memory.

Attention: Once the bit BLOCK_MTP_ACCESS is set, prototyping cannot overwrite values.

MTP_PROT_ADDR VALUE	RECORD TYPE	MTP_REGISTER [SELECTED BITS] WRITABLE BY MTP_PROT_WDATA	HINT
0x0	0x0	REGISTER_ADDRESS [6:0]	
		REGISTER_WRITE_LOCK [7]	
0x4	0x1	REGISTER_DATA[7: 0]	Write access to MSBs (0x7) starts prototyping access to register map.
0x5		REGISTER_DATA[15: 8]	
0x6		REGISTER_DATA[23:16]	
0x7		REGISTER_DATA[31:24]	
0x8	0x2	REGISTER_LOCK_BITS[7: 0]	Write access to MSBs (0xB) starts prototyping access to register lock map.
0x9		REGISTER_LOCK_BITS[15: 8]	
0xA		REGISTER_LOCK_BITS[23:16]	
0xB		REGISTER_LOCK_BITS[31:24]	
0xC	0x3	I2C_NODE_ADDRESS [2:0]	Set the bits [4:2] of the I^2C device address (only applicable for TMC5222). **Hint**: Adress changes immediately during prototyping!
		DISABLE_IREF_FAULT [3]	Setting directly is valid.
0xF		BLOCK_MTP_ACCESS[7]	Setting directly is valid: once this bit is set, further prototyping cannot be executed until the next chip reset.

Protoyping example:

//Prototype USER_DATA_3 (0x49) and USER_DATA_4(0x4A)

Write(0x7D, 0x12A7); // set the password to enter OTP mode

Write(0x10, 0x10); //Enable prototyping

Write(0x12, 0x0); //Set MTP_PROT_ADDR to point to FTP Register: REGISTER_WRITE_LOCK, REGISTER_ADDRESS

Write(0x13, 0x49); //Set the internal register offset to point to USER_DATA_3 (0x49) without Write Lock

Write(0x12, 0x4); //Set MTP_PROT_ADDR to point to FTP Register: REGISTER_DATA[7:0]

Write(0x13, 0x10); //Set bits [7:0] of USER_DATA_3 to 0x10

Write(0x12, 0x5); //Set MTP_PROT_ADDR to point to FTP Register: REGISTER_DATA[15:8]

Write(0x13, 0x32); //Set bits [15:8] of USER_DATA_3 to 0x32

Write(0x12, 0x6); //Set MTP_PROT_ADDR to point to FTP Register: REGISTER_DATA[23:16]

Write(0x13, 0x54); //Set bits [23:16] of USER_DATA_3 to 0x54

Write(0x12, 0x7); //Set MTP_PROT_ADDR to point to FTP Register: REGISTER_DATA[31:24]

Write(0x13, 0x76); //Set bits [31:24] of USER_DATA_3 to 0x76. Internal address counter +1.

//As the address counter now already points to 0x4A there is no need to set MTP_PROT_ADDR again

Write(0x12, 0x4); //Set MTP_PROT_ADDR to point to FTP Register: REGISTER_DATA[7:0]

Write(0x13, 0xFE); //Set bits [7:0] of USER_DATA_4 to 0xFE

Write(0x12, 0x5); //Set MTP_PROT_ADDR to point to FTP Register: REGISTER_DATA[15:8]

Write(0x13, 0xDC); //Set bits [15:8] of USER_DATA_4 to 0xDC

Write(0x12, 0x6); //Set MTP_PROT_ADDR to point to FTP Register: REGISTER_DATA[23:16]

Write(0x13, 0xBA); //Set bits [23:16] of USER_DATA_4 to 0xBA

Write(0x12, 0x7); //Set MTP_PROT_ADDR to point to FTP Register: REGISTER_DATA[31:24]

Write(0x13, 0x98); //Set bits [31:24] of USER_DATA_4 to 0x98. Internal address counter +1.

Write(0x10, 0x0); //Disable prototyping

Write(0x7D, 0x0123); // Leave OTP mode

DcStep

DcStep is an automatic commutation mode for the stepper motor. It allows the stepper to run with its target velocity as commanded by the ramp generator as long as it can cope with the load. In case the motor is overloaded, it slows down to a velocity where the motor can still drive the load. This way, the stepper motor never stalls and can drive heavy loads as fast as possible. Its higher torque available at lower velocity, plus dynamic torque from its flywheel mass, allow compensating for mechanical torque peaks. In case the motor is completely blocked, the stall flag is set.

This mode is only available when the internal motion controller is used. In the external step and direction mode, DcStep is not available.

Designing-In DcStep

In a classical application, the operation area is limited by the maximum torque required at maximum application velocity. A safety margin of up to 50% torque is required to compensate for unforeseen load peaks, torque loss due to resonance, and aging of mechanical components. DcStep allows using up to the full available motor torque, as shown in *Figure 44*. Even higher short-time dynamic loads can be overcome using motor and application flywheel mass without the danger of a motor stall. With DcStep, the nominal application load can be extended to a higher torque only limited by the safety margin near the holding torque area (the highest torque the motor can provide). Additionally, maximum application velocity can be increased up to the actually reachable motor velocity.

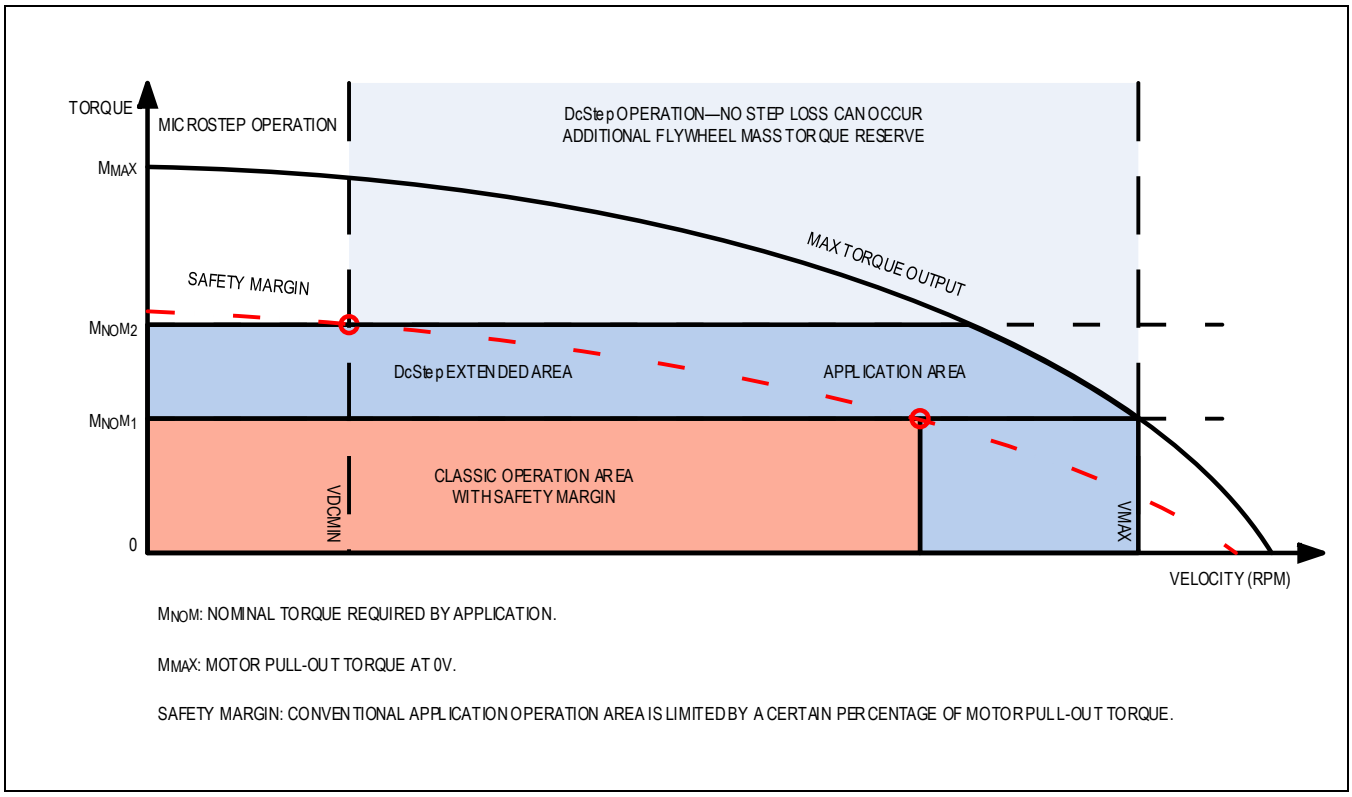


Figure 44. DcStep Extended Application Operation Area

DcStep Integration with the Motion Controller

DcStep requires only a few settings. It directly feeds back motor motion to the ramp generator so that it seamlessly integrates into the motion ramp even if the motor is overloaded with respect to the target velocity. DcStep operates the motor using the constant TOFF chopper in full-step mode at the ramp generator target velocity VMAX or at reduced velocity if the motor is overloaded. *Figure 45* illustrates an overload situation. It requires setting the minimum operation velocity VDCMIN. VDCMIN must be set to the lowest operating velocity where DcStep gives a reliable detection of motor operation. The motor never stalls unless it is braked down to a velocity below VDCMIN. If the velocity falls below this value, the motor restarts once its load is released, unless the stall detection is enabled (set SG_STOP). Stall detection is covered by StallGuard2/StallGuard4 when the velocity goes below VDCMIN.

Note: DcStep requires the phase polarity of the sine-wave is positive within the MSCNT range 768 to 255 and negative within 256 to 767. The cosine polarity must be positive from 0 to 511 and negative from 512 to 1023. Any phase shift disturbs the DcStep operation. Therefore, it is advised to work with the default wave. See the **Enabling DcStep Operation** section for an initialization with the default table.

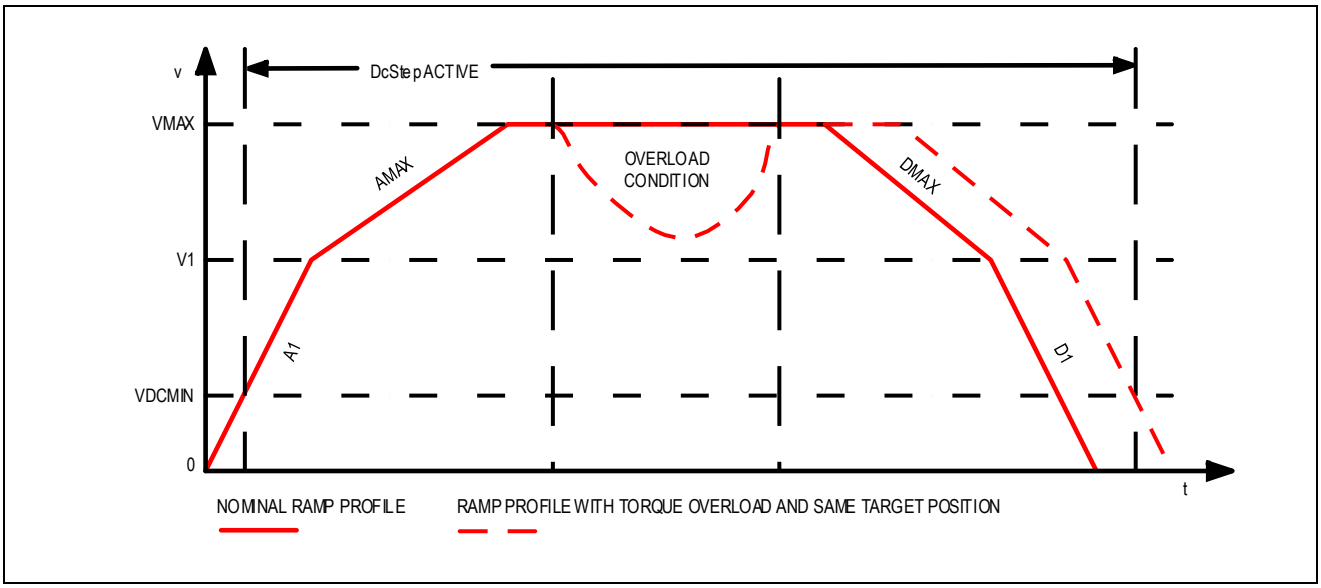


Figure 45. DcStep Velocity Profile with Overload Condition

Stall Detection in DcStep Mode

While DcStep is able to decelerate the motor upon overload, it cannot avoid a stall in every operational situation. Once the motor is blocked, or it is decelerated below a motor-dependent minimum velocity where the motor operation cannot safely be detected anymore, the motor may stall and lose steps. To safely detect a step loss and avoid restarting of the motor, the stop-on-stall can be enabled (set the SG_STOP flag). In this case, VACTUAL is set to zero once the motor is stalled. It remains stopped until reading the RAMP_STAT status flags. The flag event_stop_sg shows the active stop condition. It remains stopped until resetting the EVENT_SG_STOP flag or disabling stop-on-stall. A StallGuard2 load value is also available during the DcStep operation. The range of values is limited to 0 to 255 as DcStep only sees a load angle of 0° to 90°, which is mapped to 0 to 255. In certain situations, the load angle might slip temporarily to the 90° to 180° range, which is not stable but gives a read-out of up to 511. To enable StallGuard2, also set TCOOLTHRS corresponding to a velocity slightly above VDCMIN or up to VMAX. *Table 24* shows the parameters related to DcStep stall detection.

Stall detection in this mode can trigger falsely due to resonances when flywheel loads are loosely coupled to the motor axis.

Table 24. Parameters for Stall Detection in DcStep Mode

PARAMETER	DESCRIPTION	RANGE	NOTES
VHIGHFS and VHIGHCHM	These chopper configuration flags in CHOPCONF must be set for DcStep operation. As soon as VDCMIN is exceeded, the chopper is switched to constant TOFF chopper and full stepping.	0/1	Set to 1 for DcStep.
TOFF	DcStep often benefits from an increased off-time value in CHOPCONF. Settings > 2 are preferred.	2 to 15	Settings 8 to 15 do not make any difference to setting 8 for DcStep operation.
VDCMIN	This is the lower threshold for DcStep operation when using an internal ramp generator. Below this threshold, the motor operates in normal microstep mode. In DcStep operation, the motor operates at minimum VDCMIN, even when it is completely blocked. Tune together with DC_TIME setting. Activation of StealthChop2 also disables DcStep.	0 to 2^{22}	0: Disable DcStep. Set to the lower velocity limit for DcStep operation.
DC_TIME	This setting controls the reference pulse width for DcStep load measurement. It must be optimized for robust	0 to 1023	Lower limit for the setting is: tBLANK (as defined by TBL) in

	operation with maximum motor torque. A higher value allows higher torque and higher velocity. A lower value allows operation down to a lower velocity as set by VDCMIN. Check the best setting under nominal operating conditions and recheck under extreme operating conditions (example, lowest operation supply voltage, highest motor temperature, and highest supply voltage, lowest motor temperature).		clock cycles + n with n in the range of 1 to 100 (for a typical motor).
DC_SG	This setting controls stall detection in DcStep mode. Increase for higher sensitivity. A stall can be used as an error condition by issuing a hard stop for the motor. Enable the SG_STOP flag for stopping the motor upon a stall event. This way the motor is stopped once it stalls.	0 to 255	Set slightly higher than DC_TIME/16.

Measuring Actual Motor Velocity in DcStep Operation

DcStep has the ability to reduce motor velocity when the motor is slower than the target velocity due to mechanical load. VACTUAL shows the ramp generator target velocity. It is not influenced by DcStep. Measuring DcStep velocity is possible based on the position counter XACTUAL.

Therefore, take two snapshots of the position counter with a known time difference:

$$\mathrm{VACTUAL_{DCSTEP}} = \frac{\mathrm{XACTUAL(TIME2) - XACTUAL(TIME1)}}{\mathrm{TIME2 - TIME1}} \times \frac{2^{24}}{f_{CLK}}$$

An example of actual motor velocity in DcStep operation is:

At 16.0MHz clock frequency, a 0.954s measurement delay directly yields in the velocity value; a 9.54ms delay yields in 1/100 of the actual DcStep velocity.

To get the time interval as precise as possible, snapshot a timer each time the transmission of XACTUAL from the IC starts or ends. The rising edge of the CSN pin for SPI transmission provides the most exact time reference.

Clock Oscillator and Clock Input

Using the Internal Clock

If the internal clock oscillator is used, directly tie the CLK input pin to GND close to the IC. The internal clock typically runs at a frequency of 12.5MHz in on-state and 609khz in low-power quiescent mode. Additional information is given in the section.

Using an External Clock

Using a dedicated external clock allows to precisely calculate the time base for all velocity and acceleration computations inside the TMC5221/TMC5222. When an external clock is available, a frequency of 8MHz to 20MHz is recommended for optimum performance. The required minimum and maximum duty cycle of the clock signal is defined in the section. The clock's duty cycle requirements must be satisfied, especially at clock frequencies close to 20MHz.

Make sure the clock source supplies clean CMOS output logic levels and steep slopes when using a high clock frequency. The external clock input is enabled as soon as an external clock is provided at the CLK pin. Reading out EXT_CLK in the IOIN register gives feedback on which clock source (external/internal/internal_slow) is currently in use.

When the external clock fails or is switched off, the internal clock takes over seamlessly and automatically to protect the driver from damage, and the motion continues. This might lead to differences in velocity when using the internal motion controller and f_clk_int != f_clk_ext. While the TMC5221/TMC5222 are in quiescence power-down mode, the clock source switches to the internal low-power clock (609kHz).

Protections and Driver Diagnostics

The TMC5221/TMC5222 drivers provide a complete set of diagnostics, status information, and protection capabilities, example, short-to-GND protection and undervoltage detection. A detection of an open-load condition allows testing if a motor-coil connection is interrupted. See the DRV_STATUS register table for details.

Thermal Protection and Shutdown

The TMC5221/TMC5222 have internal thermal protection. If the die temperature exceeds +155°C (typical value), a fault indication (the OT fault flag in DRV_STATUS) is raised and the driver is three-stated until the junction temperature drops below approximately +135°C (typical value). After that, the driver is re-enabled automatically.

In addition, the internal ADC senses the chip average temperature (although the driver stages may be at a much higher temperature), which can be read out from TEMPERATURE in ADC_SUPPLY_TEMP. This measurement function can be disabled with the ADC_EN control bit in the same register to reduce energy consumption if temperature measurement is not needed. The ADC_TEMPERATURE value can be converted into degree Celsius with the following formula:

T(°C) = (1.017 °C × ADC_TEMPERATURE) - 259,2 °C.

The overtemperature prewarning flag (OTPW in DRV_STATUS) is set when TEMPERATURE reaches ~120°C and may be propagated to any DIAG pin.

The ADC measurement is disabled in the low-power quiescent mode regardless of the ADC_EN bit.

Heat is mainly generated by the motor driver stages. Most critical situations, where the driver MOSFETs can be overheated, are avoided by a correctly set current range, which in turn influences short protection sensitivity. For many applications, the overtemperature prewarning indicates an abnormal operation and can be used to initiate user warnings or power-reduction measures like motor current reduction. The thermal shutdown is an emergency measure only. Temperature rising to the shutdown level should be prevented by design.

Motor Temperature Measurement

The PWM_SCALE register shows the actual duty cycle in StealthChop2 operation. For a given motor current, the duty cycle depends on the phase resistance of the motor. As the phase resistance is temperature dependent, PWM_SCALE_SUM values can be used to estimate the actual motor temperature and monitor changes in the motor temperature over time. This measurement is preferably done during motor standstill or slow movements. Typically, the motor temperature does not change quickly.

Short Protection

The TMC5221/TMC5222 power stages are protected against a short-circuit condition to GND and to V_S by an additional measurement of the current flowing through the high-side MOSFETs. This is important as most short-circuit conditions result from a motor cable insulation defect, example, when touching the conducting parts connected to the system ground.

The short detection is protected against spurious triggering, example, by ESD (by retrying thrice before switching off the motor).

Once a short condition is safely detected, both the driver bridges are switched off, and the s2ga or s2gb flag is set. To restart the motor, intervene by disabling and re-enabling the driver. This can be done by either toggling the DRV_EN pin, toggling DRV_EN_SW, or changing TOFF to 0 and back to its previous value. Note that the short-to-GND protection cannot protect the system and the power stages for all possible short events as a short event is undefined, and a complex network of external components may be involved. Therefore, short circuits should be avoided.

Depending on the full-scale current setting, the low-side short protection triggers at different overcurrent protection thresholds, as shown in the **Electrical Characteristics** table.

Open Load Diagnostics

Interrupted cables are a common cause for systems failing, example, when connectors are not firmly plugged. The TMC5221/TMC5222 detect open load conditions by checking if these can reach the desired motor-coil current. Undervoltage conditions, high motor velocity settings, or short and overtemperature conditions can also cause triggering of the open-load flag, and inform that motor torque may suffer. In motor standstill, open load cannot be measured as the coils might eventually have zero current.

To safely detect an interrupted coil connection, operate in SpreadCyle, and check the open load flags following a motion of a minimum of four times the selected microstep resolution (= 4 full steps) into a single direction using low or nominal motor velocity operation only. However, the OLA and OLB flags are informational and do not cause any driver action.

Undervoltage Lockout Protection

The TMC5221/TMC5222 feature a UVLO protection for $+V_S$ and a POR protection for $+V_{CC\_IO}$. The UVLO condition on $+V_S$ is triggered below 1.9V (maximum). The POR condition on $+V_{CC\_IO}$ is triggered below 1.05V (maximum).

A $+V_S$ UVLO condition can be read from the GSTAT register as the vm_uvlo flag. This is a write-clear flag. It must be actively set to 1 to clear. The UVLO condition is also shown at the DIAG0/1 pin, depending on the configured pin settings (DIAGx_ERROR).

During a $+V_{CC\_IO}$ POR, no communication with the IC is possible. The DIAG0 pin is active-low (open-drain).

ESD Protection

The chip has internal ESD protection on every pin. The TMC5221/TMC5222 motor phase output pins are protected up to 8kV HBM in the application when using a bypass capacitor of at least 1µF on the positive voltage supply (V_S pin). This is not a protection against hot plugging of a motor.

Quick Configuration Guides

The following guides in *Figure 46* to *Figure 55* are meant as practical tools to set up an initial configuration, take a minimum set of measurements, and make decisions for tuning the driver. They do not cover all the advanced functionalities and options, instead concentrating on the basic function set to make a motor run smoothly. Once the motor runs, there is the option to explore additional features and further functionalities in more detail. A current probe on one motor coil is a good aid to find the best settings.

Current Setting

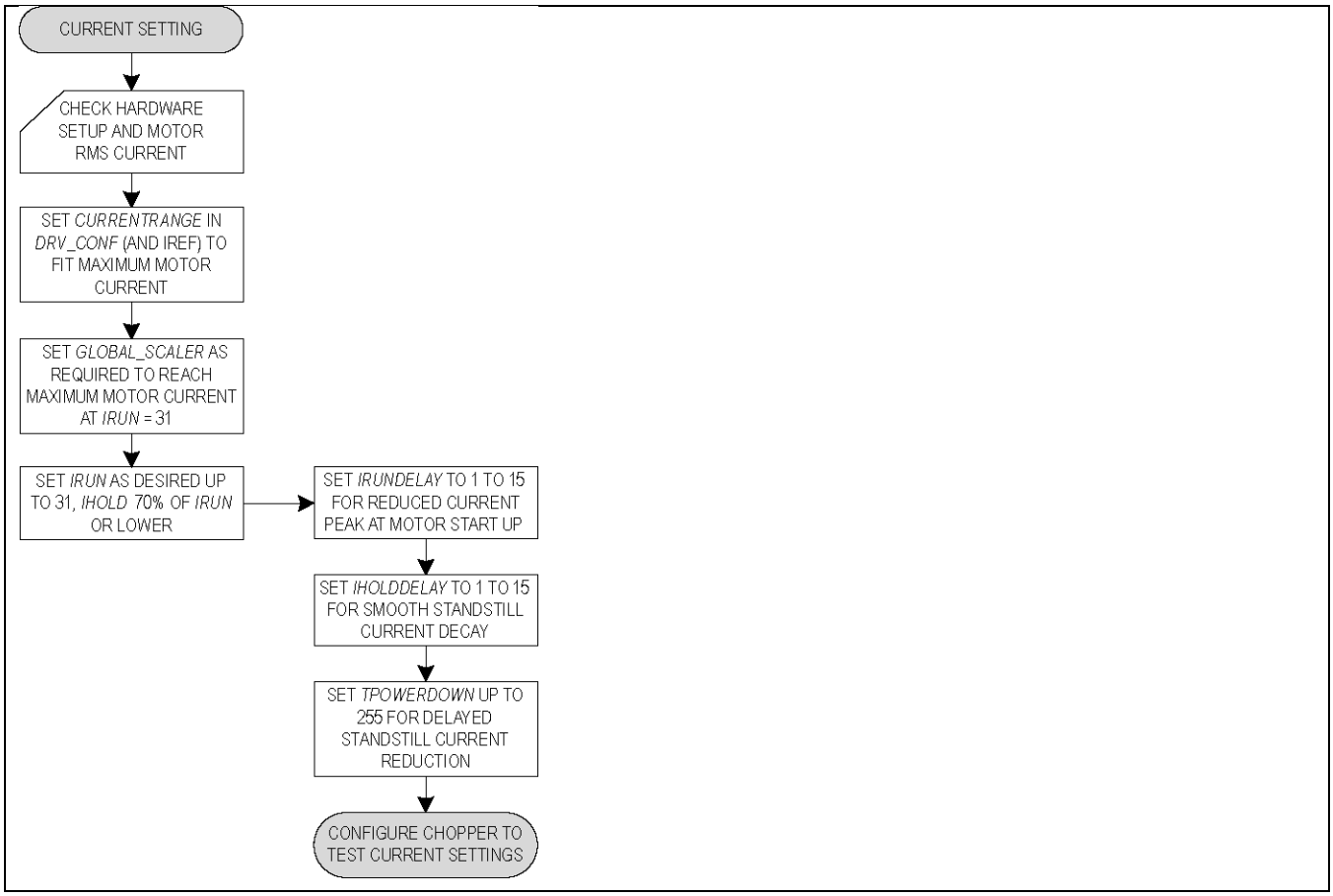


Figure 46. Current Setting

StealthChop2 Configuration

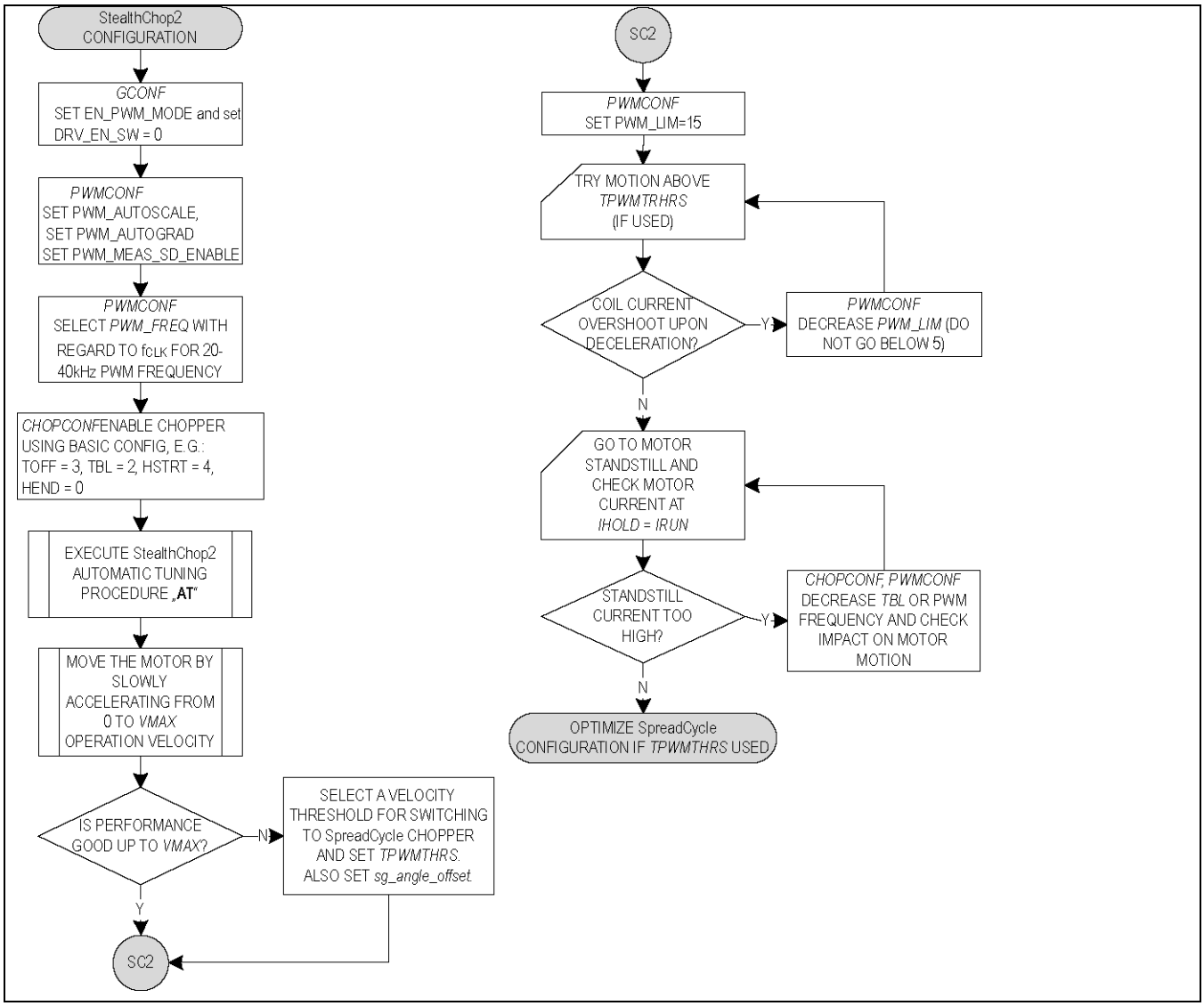


Figure 47. StealthChop2 Configuration

SpreadCycle Configuration

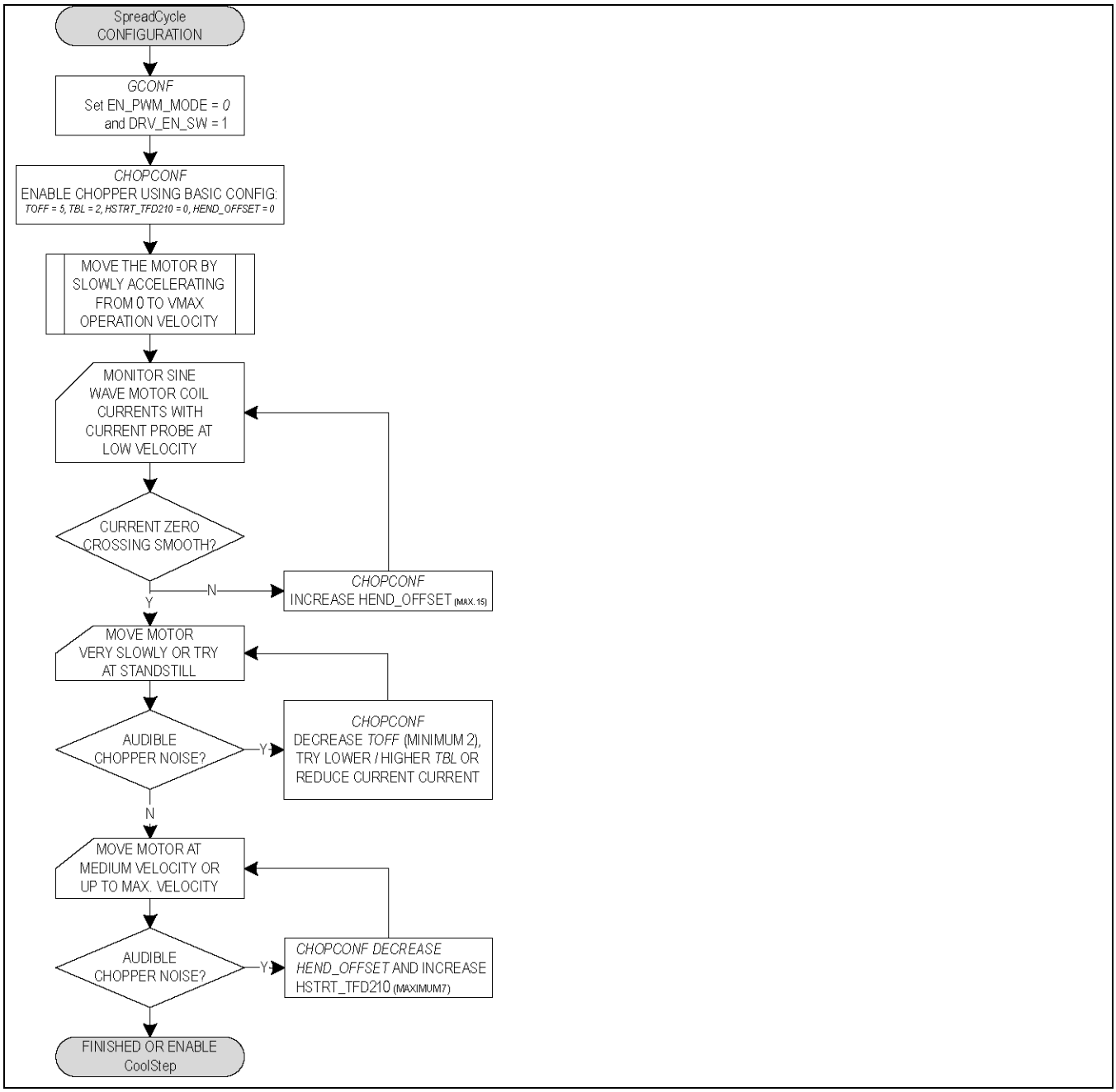


Figure 48. SpreadCycle

Enabling CoolStep in Combination with StealthChop2

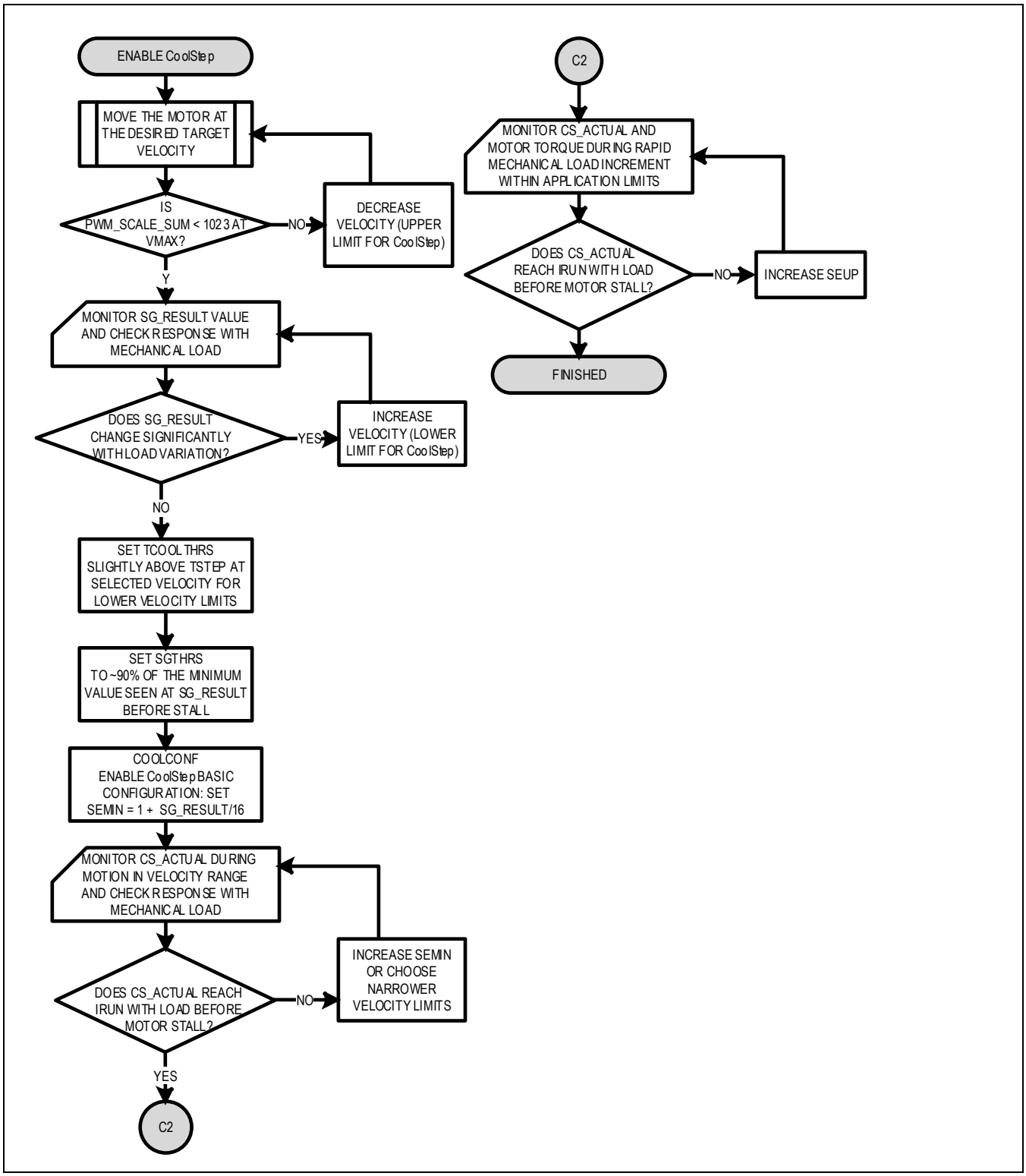


Figure 49. CoolStep with StealthChop2

Enabling CoolStep in Combination with SpreadCycle

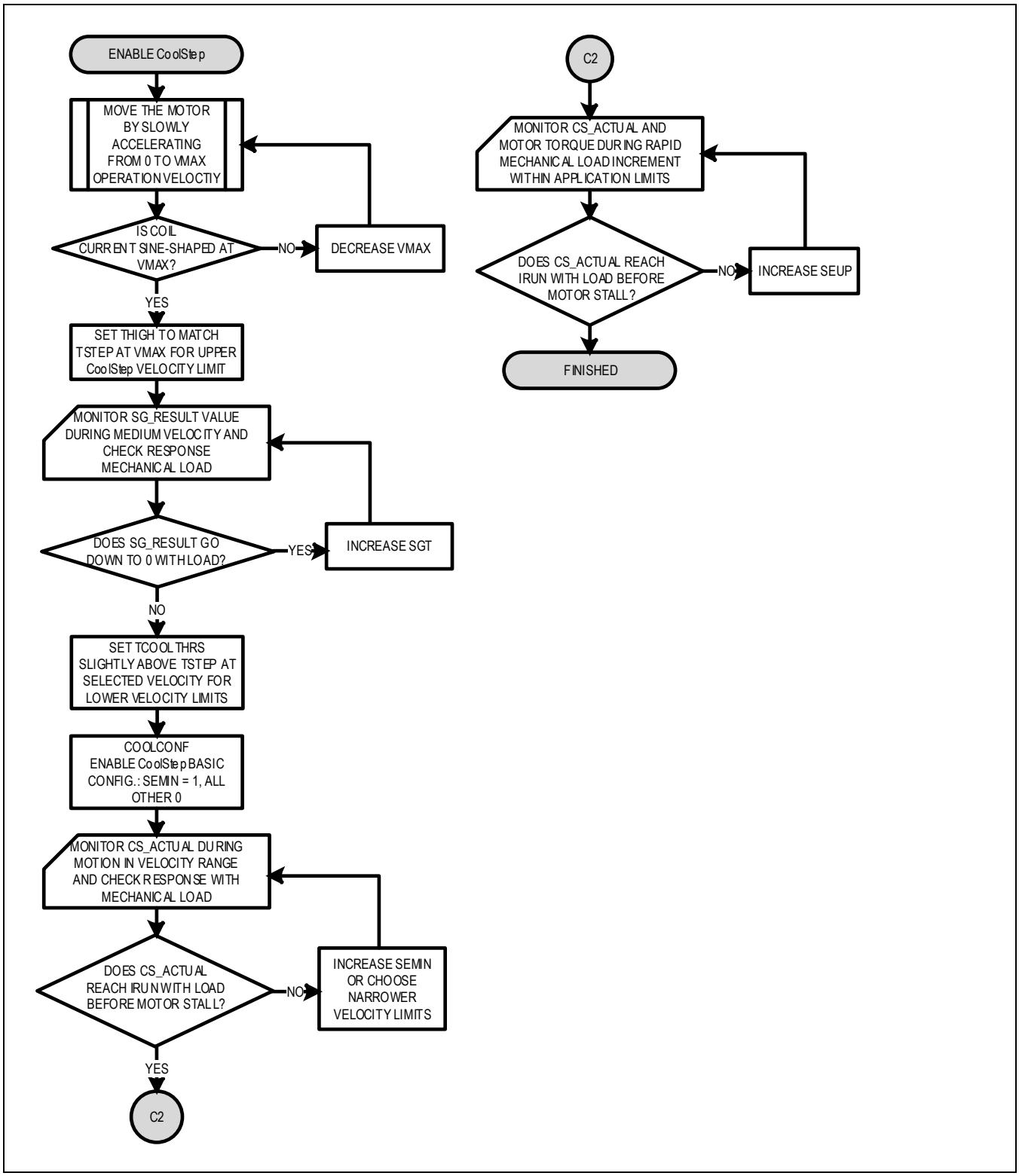


Figure 50. CoolStep with SpreadCycle

Moving the Motor Using the Motion Controller

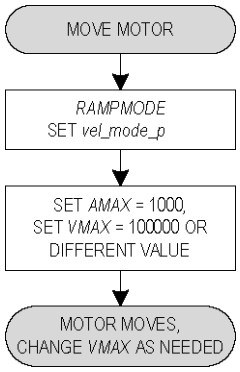


Figure 51. Moving a Motor in Velocity Mode

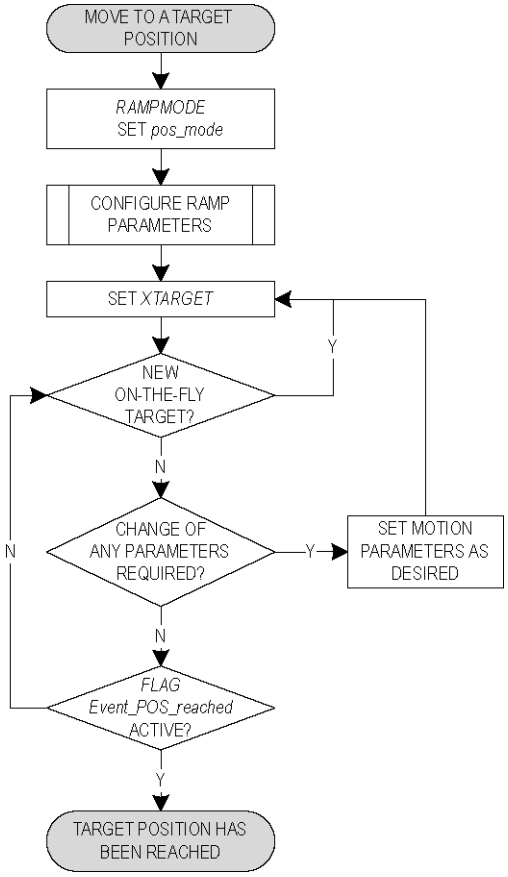


Figure 52. Moving a Motor to a Target Position

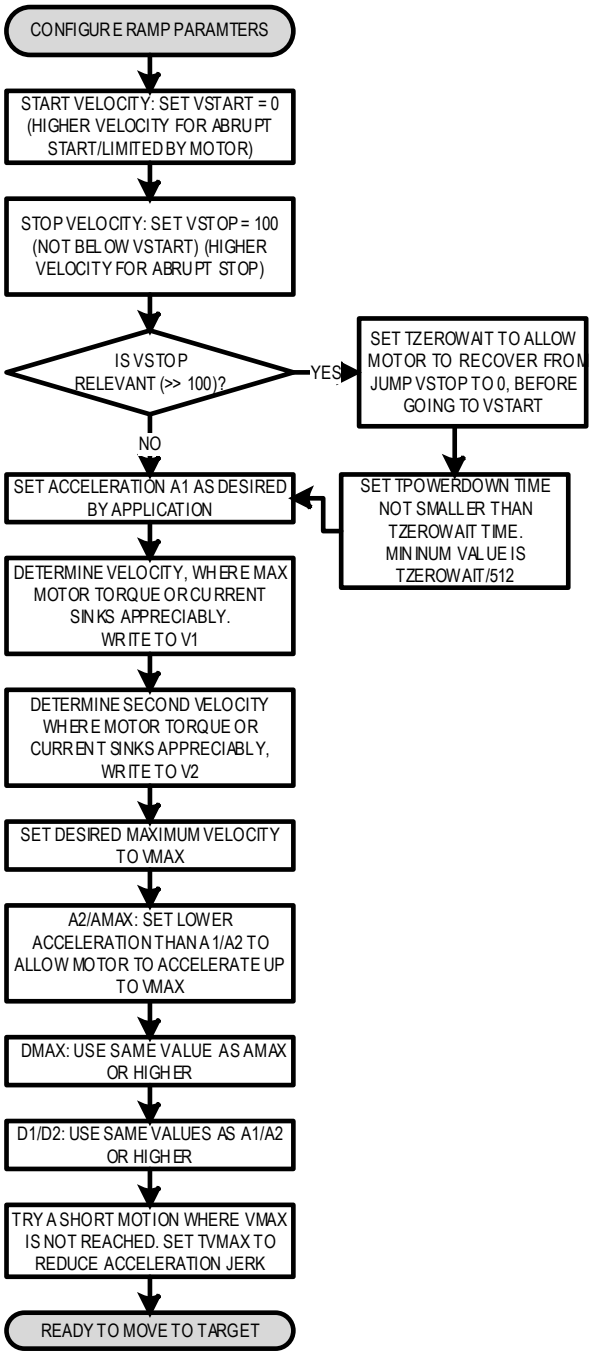


Figure 53. Motion Ramp Parameter Setting

Enabling DcStep Operation

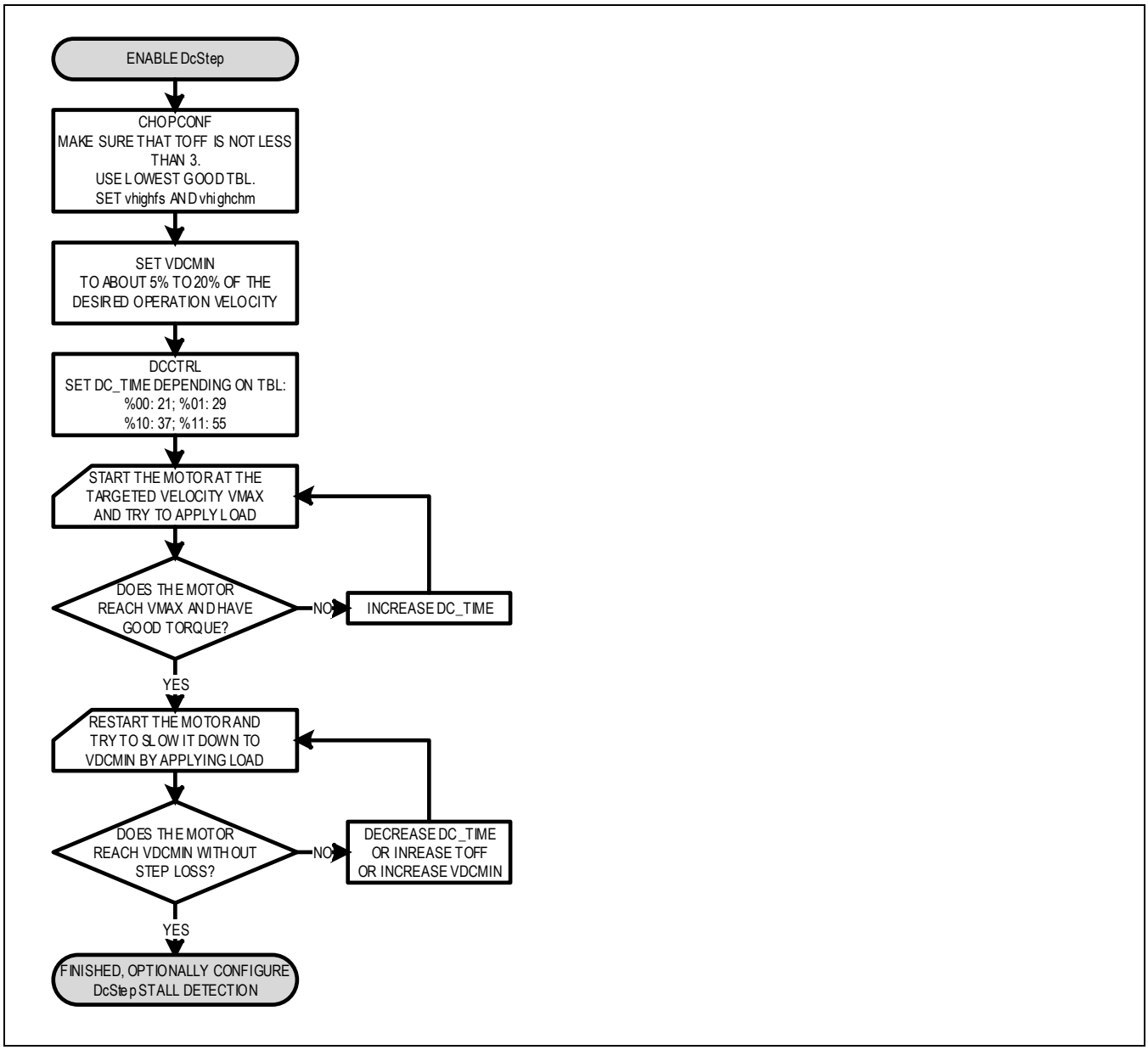


Figure 54. DcStep

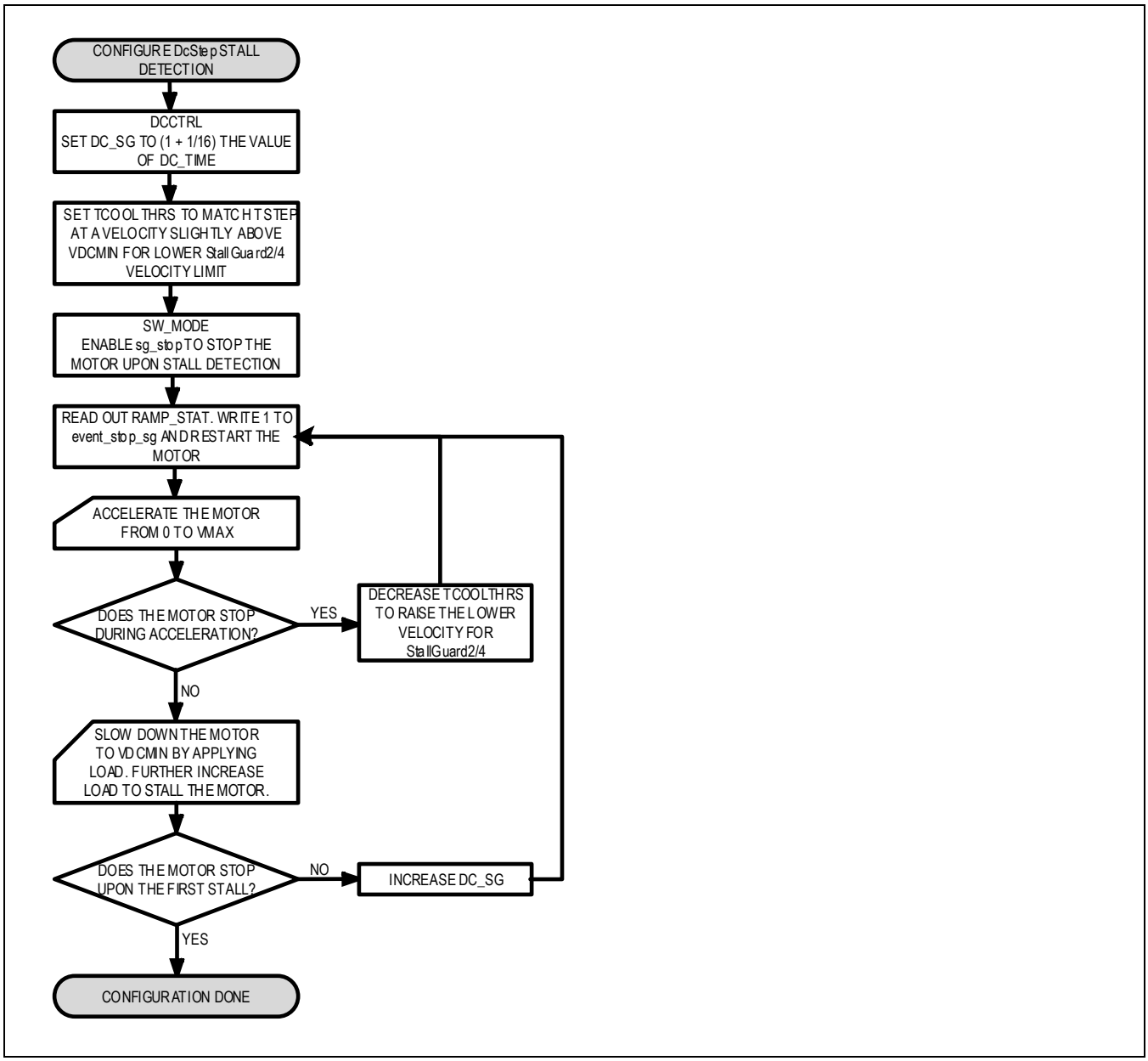


Figure 55. Using Stall Detection with DcStep

Fitting the Motor

The motor should be carefully selected to give a good fit to the application's mechanics, as well as available supply voltage and current, especially for low-voltage operation. Therefore, it is important to understand the supply-voltage requirement for a given motor. Both the generation of a certain torque and the ability to provide this torque at a desired velocity require a motor-specific voltage. These two components add up.

The main relevant parameters for a stepper motor are:

- Nominal (RMS) coil current ICOILNOM [A]
- Nominal coil resistance RCOIL [Ω]
- Rated coil voltage UN = RCOIL × ICOILNOM [V] (sometimes specified instead of ICOILNOM).
- Holding torque at ICOILNOM HoldingTorque [Nm].
- Back-EMF (BEFM) constant CBEMF [V/rad/s]. More information on the motor BEMF is also given in the *Understanding the Back-EMF (BEMF) Constant of a Motor* section.

The specified motor torque is reached with the RMS I_{COIL} current in both the motor coils to build up the required magnetic field strength. Essentially, a lower current proportionally generates a lower torque, example, 70% of torque at 70% current. Even a reduction to 70% saves a lot of energy because power dissipation goes with the square of the current. Thus, a motor with more reserves can offer better efficiency.

With this, calculate the required supply voltage UBAT for motor standstill and slow motion, considering the driver's power stage resistance (LS + HS):

$$U_{BAT} = (R_{COIL} + 0.32\Omega) \times I_{COIL} \times \sqrt{2}$$

I_{COIL} is the RMS motor current that gives the desired torque.

For higher velocity operation (more than a few electrical rotations per second), the motor-specific back-EMF constant CBEMF should be additionally considered (see the following explanation). Given this, the lowest feasible supply voltage for a given motor and a maximum velocity [RPM] calculates to:

$$U_{BAT} = ((R_{COIL} + 0.32\Omega) \times I_{COIL} + \frac{\text{Holding Torque[Nm]}}{2 \times I_{COILNOM}} \times \frac{2\pi \times \text{Velocity[RPM]}}{60}) \times \sqrt{2}$$

Adapt the motor to battery operation: With most motor suppliers, the coil winding can be adapted. This allows trading a lower motor voltage for battery operation versus higher motor current. A motor with a short, thick coil wire can work at a lower voltage than the same motor with a long, thin coil wire, but it needs a higher current for the same torque.

Typical Application Circuits

Standard Application Circuit

The standard application circuit uses a minimum set of additional components, as shown in *Figure 56*. Use low ESR capacitors for filtering the power supply V_S and depend on the actual supply voltage used. The capacitors must handle the current ripple caused by the chopper operation. A capacitor of 10µF near the driver is recommended for best performance, depending on the required motor voltage and current. Current ripple in the supply capacitors also depends on the power-supply internal resistance and cable length (if applicable). V_{CC_IO} must be supplied from an external source, for example a 1.8V LDO.

Place all the filter capacitors as close as possible to the related IC pins. Use a solid common GND layer for all GND connections. Connect the filtering capacitor directly to the V_{DD1V8} pin.

Note: For optimal performance of the current regulation and proper current levels, the TMC5221/TMC5222 require a symmetrical board layout.

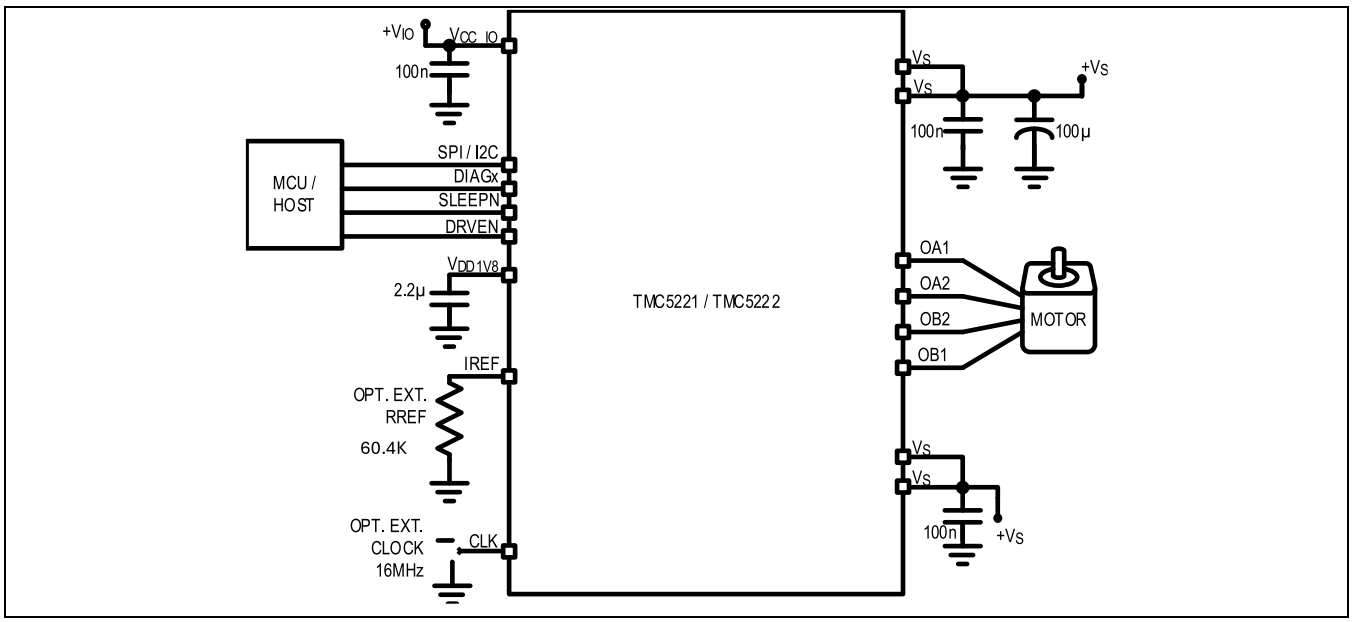


Figure 56. Standard Application Circuit of the TMC5221/TMC5222

All TMC5221/TMC5222 signals are referenced to their respective GND. Directly connect all the GND pins/balls to a common ground. For thermal reasons, the PCB top layer is connected to a larger GND plane to spread heat within the PCB.

Driver Protection and EME Circuitry

Some applications must handle ESD events caused by the motor operation or external influence. Despite the integrated ESD protection circuitry within the driver chips, ESD events occurring during operation can cause a reset or even a destruction of the motor driver, depending on their energy. In particular, plastic housings and belt drive systems tend to cause ESD events of several kV. It is best practice to avoid ESD events by attaching all the conductive parts, especially the motors themselves, to PCB ground, or to apply electrically-conductive plastic parts. In addition, the driver can be protected to a certain extent against ESD events or live plugging/pulling the motor, which also causes high voltages and high currents into the motor connector terminals.

Since the TMC5221/TMC5222 are very small components and target low-voltage operation, the use of additional external protection circuity should be carefully planned to avoid unnecessarily inflating the BOM and required board space.

Figure 57 shows a simple scheme and uses capacitors at the driver outputs to reduce the dV/dt caused by ESD events. Larger capacitors are beneficial to ESD suppression but cause additional current flow in each chopper cycle, and thus, increase driver power dissipation. This effect increases with the supply voltage. The values shown are example values and can be varied between 100pF and 1nF. The capacitors also dampen the high-frequency noise injected from the digital parts of the application PCB circuitry, and thus, reduce electromagnetic emission.

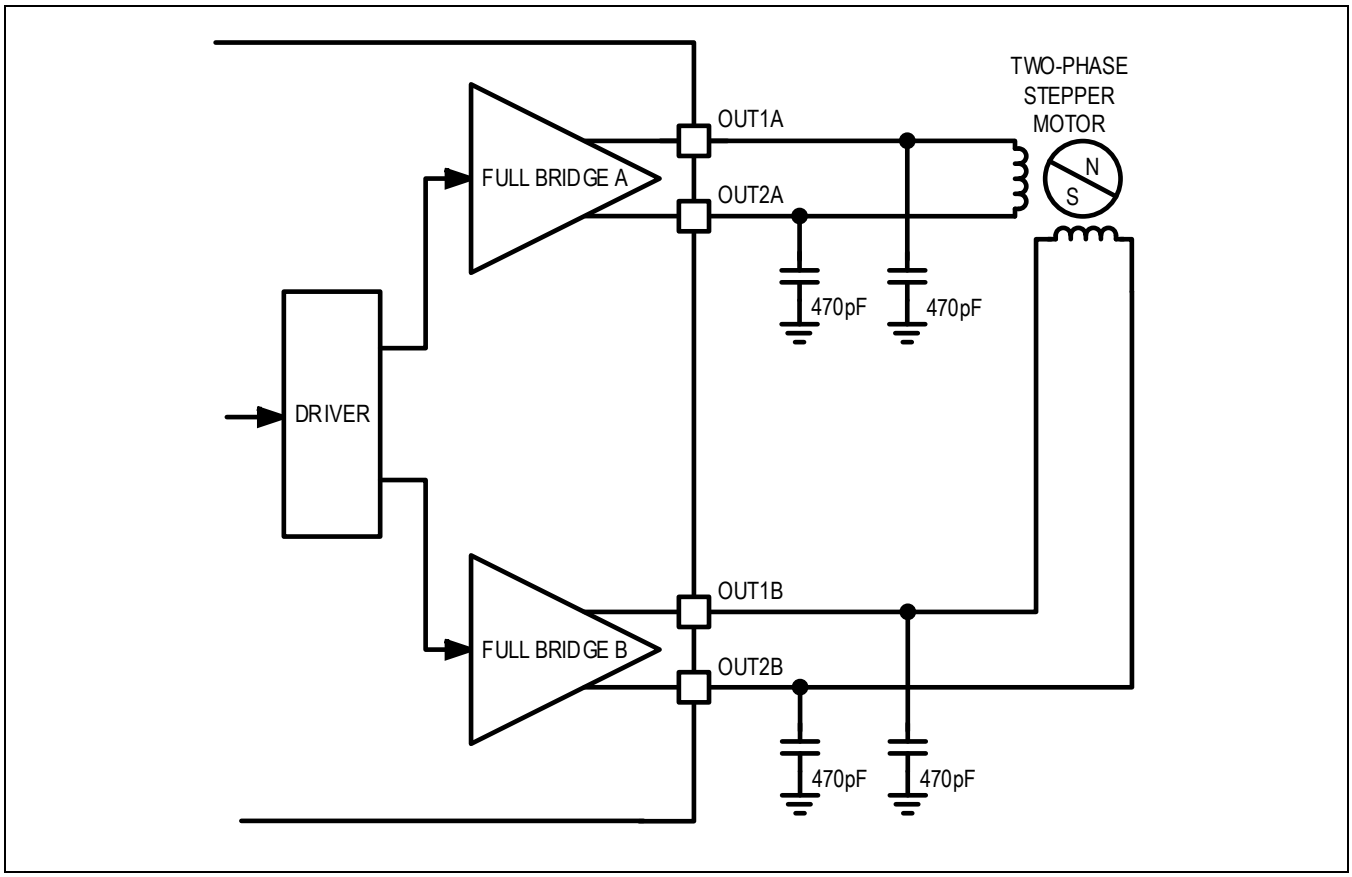


Figure 57. Simple ESD Enhancement

Figure 58 shows a more detailed scheme and uses LC filters to decouple the driver outputs from the motor connector. Varistors V1 and V2 between the coil terminals eliminate the coil overvoltage caused by live plugging. Another option is to protect all the outputs by a varistor (V1A, V1B, V2A, V2B) against the ESD voltage. Fit the varistors to the supply voltage rating. The SMD inductivities conduct full motor coil current and must be selected accordingly.

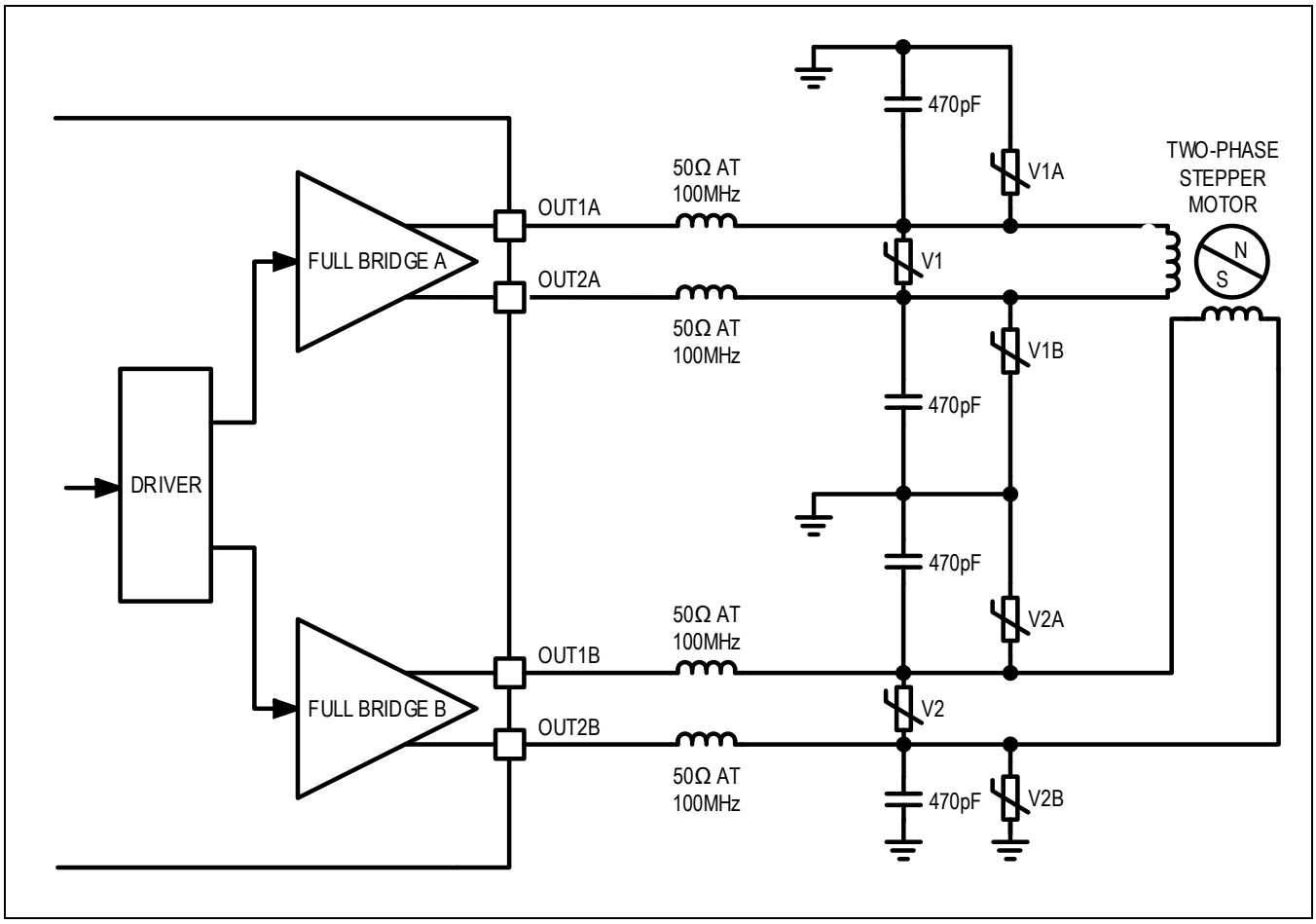


Figure 58. Extended Motor Output Protection

Register Map

GCR Register Overview

Shows all the related registers of the GCR block.

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)						
0x00 GCONF								
				DISABLE_ DRIVER_ ON_UVLO	S_IREF_ INTCUR_EN	SW_RESET	DRV_EN_ SW	QSC_STS_ ENA
	SD	DIRECT_ MODE	REFR_ STOP	STOP_ ENABLE	SMALL_ HYSTERESIS	SHAFT	MULTISTEP_FILT	EN_PWM_ MODE
0x01 GSTAT								
	VDD_UVLO	RREFF_ ERR	VCCIO_RST	VM_UVLO	REGISTER_ RESET	UV_LDO	DRV_ERR	RESET
0x03 NODECONF								
								I2C_AUT_ INC
0x04 IOIN	VERSION					ONSTATE	QSC	EXT_CLK[1]
	EXT_CLK[0]	SILICON_RV						
0x05 DRV_CONF								
						STANDSTILL_TIME		
					FS_GAIN		FS_SENSE	
0x06 GLOBAL_SCALER								
	GLOBALSCALER_B							
	GLOBALSCALER_A							
0x07 DIAG_CONF	DIAG1_ INVPP	DIAG1_ NOD_PP	DIAG0_ INVPP	DIAG0_ NOD_PP	DIAG1_ POSITION_ REACHED	DIAG1_EV_ HOMING	DIAG1_EV_ DEVIATION	DIAG1_EV_ POS_ REACHED
	DIAG1_EV_ STOP_SG	DIAG1_EV_ STOP	DIAG1_ XCOMP	DIAG1_DIR	DIAG1_ STEP	DIAG1_ INDEX	DIAG1_ STALL	DIAG1_ STALL_ PREWARN
	DIAG1_ OTPW	DIAG1_ ERROR	DIAG0_ POSITION_ REACHED	DIAG0_EV_ HOMING	DIAG0_EV_ DEVIATION	DIAG0_EV_ POS_ REACHED	DIAG0_EV_ STOP_SG	DIAG0_EV_ STOP_REF
	DIAG0_ XCOMP	DIAG0_DIR	DIAG0_ STEP	DIAG0_ INDEX	DIAG0_ STALL	DIAG0_ STALL_ PREWARN	DIAG0_ OTPW	DIAG0_ ERROR
0x08 MSLUT0	MSLUT_0[31:24]							
	MSLUT_0[23:16]							
	MSLUT_0[15:8]							
	MSLUT_0[7:0]							
0x09 MSLUT1	MSLUT_1[31:24]							
	MSLUT_1[23:16]							
	MSLUT_1[15:8]							
	MSLUT_1[7:0]							

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)							
0x0A MSLUT2	MSLUT_2[31:24]								
	MSLUT_2[23:16]								
	MSLUT_2[15:8]								
	MSLUT_2[7:0]								
0x0B MSLUT3	MSLUT_3[31:24]								
	MSLUT_3[23:16]								
	MSLUT_3[15:8]								
	MSLUT_3[7:0]								
0x0C MSLUT4	MSLUT_4[31:24]								
	MSLUT_4[23:16]								
	MSLUT_4[15:8]								
	MSLUT_4[7:0]								
0x0D MSLUT5	MSLUT_5[31:24]								
	MSLUT_5[23:16]								
	MSLUT_5[15:8]								
	MSLUT_5[7:0]								
0x0E MSLUT6	MSLUT_6[31:24]								
	MSLUT_6[23:16]								
	MSLUT_6[15:8]								
	MSLUT_6[7:0]								
0x0F MSLUT7	MSLUT_7[31:24]								
	MSLUT_7[23:16]								
	MSLUT_7[15:8]								
	MSLUT_7[7:0]								
0x10 MSLUT_START	OFFSET_SIN90								
	START_SIN90								
	START_SIN								
0x11 MSLUT_SEL	X3								
	X2								
	X1								
	W3		W2		W1		W0		
0x12 X_COMPARE	X_COMPARE[31:24]								
	X_COMPARE[23:16]								
	X_COMPARE[15:8]								
	X_COMPARE[7:0]								
0x13 X_COMPARE_REPEAT									
	X_COMPARE_REPEAT[23:16]								
	X_COMPARE_REPEAT[15:8]								
	X_COMPARE_REPEAT[7:0]								
0x14 IHOLD_IRUN					IRUNDELAY				
					IHOLDDELAY				
				IRUN					
				IHOLD					
0x15 TPOWERDOWN									
	TPOWERDOWN								
0x16 TSTEP									
					TSTEP[19:16]				
	TSTEP[15:8]								
	TSTEP[7:0]								

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)
0x17 TPWMTHRS		
		TPWMTHRS[19:16]
	TPWMTHRS[15:8]	
	TPWMTHRS[7:0]	
0x18 TCOOLTHRS		
		TCOOLTHRS[19:16]
	TCOOLTHRS[15:8]	
	TCOOLTHRS[7:0]	
0x19 THIGH		
		THIGH[19:16]
	THIGH[15:8]	
	THIGH[7:0]	
0x1A RAMPMODE		
		RAMPMODE
0x1B XACTUAL	XACTUAL[31:24]	
	XACTUAL[23:16]	
	XACTUAL[15:8]	
	XACTUAL[7:0]	
0x1C VACTUAL		
	VACTUAL[23:16]	
	VACTUAL[15:8]	
	VACTUAL[7:0]	
0x1D AACTUAL		
	AACTUAL[23:16]	
	AACTUAL[15:8]	
	AACTUAL[7:0]	
0x1E VSTART		
		VSTART[17:16]
	VSTART[15:8]	
	VSTART[7:0]	
0x1F A1		
		A1[17:16]
	A1[15:8]	
	A1[7:0]	
0x20 V1		
		V1[19:16]
	V1[15:8]	
	V1[7:0]	
0x21 A2		
		A2[17:16]
	A2[15:8]	
	A2[7:0]	
0x22 V2		
		V2[19:16]
	V2[15:8]	
	V2[7:0]	
0x23 AMAX		
		AMAX[17:16]
	AMAX[15:8]	
	AMAX[7:0]	

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)						
0x24 VMAX								
		VMAX[22:16]						
	VMAX[15:8]							
	VMAX[7:0]							
0x25 DMAX								
							DMAX[17:16]	
	DMAX[15:8]							
	DMAX[7:0]							
0x26 D2								
							D2[17:16]	
	D2[15:8]							
	D2[7:0]							
0x27 D1								
							D1[17:16]	
	D1[15:8]							
	D1[7:0]							
0x28 VSTOP								
							VSTOP[17:16]	
	VSTOP[15:8]							
	VSTOP[7:0]							
0x29 TZEROWAIT								
	TZEROWAIT[15:8]							
	TZEROWAIT[7:0]							
0x2A TVMAX								
	TVMAX[15:8]							
	TVMAX[7:0]							
0x2B XTARGET	XTARGET[31:24]							
	XTARGET[23:16]							
	XTARGET[15:8]							
	XTARGET[7:0]							
0x2C VDCMIN								
		VDCMIN[22:16]						
	VDCMIN[15:8]							
	VDCMIN[7:0]							
0x2D SW_MODE								
								EN_AUTO_ FEATURE
	EN_REFR_ HOMING	EN_REFL_ HOMING	EN_ VIRTUAL_ STOP_R	EN_ VIRTUAL_ STOP_L	EN_ SOFTSTOP	SG_STOP		LATCH_R_ INACTIVE
	LATCH_R_ ACTIVE	LATCH_L_ INACTIVE	LATCH_L_ ACTIVE	SWAP_LR	POL_ STOP_R	POL_ STOP_L	STOP_R_ ENABLE	STOP_L_ ENABLE
0x2E RAMP_STAT								
							REFR_ HOMING	REFL_ HOMING
	STATUS_ VIRTUAL_ STOP_R	STATUS_ VIRTUAL_ STOP_L	STATUS_ SG	SECOND_ MOVE	T_ ZEROWAIT _ACTIVE	VZERO	POSITION_ REACHED	VELOCITY_ REACHED
	EVENT_ POS_ REACHED	EVENT_ STOP_SG	EVENT_ STOP_R	EVENT_ STOP_L	STATUS_ LATCH_R	STATUS_ LATCH_L	STATUS_ STOP_R	STATUS_ STOP_L

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)							
0x2F XLATCH	XLATCH[31:24]								
	XLATCH[23:16]								
	XLATCH[15:8]								
	XLATCH[7:0]								
0x30 X_BEMF	X_BEMF[31:24]								
	X_BEMF[23:16]								
	X_BEMF[15:8]								
	X_BEMF[7:0]								
0x31 BEMF_CONF									
			BEMF_OVERWRITE	BEMF_DIR	BEMF_INCREMENT				
			BEMF_FILTER_SEL		BEMF_BLANK_TIME[7:4]				
	BEMF_BLANK_TIME[3:0]				QSC_TRICODER_EN	BEMF_HYST			
0x32 BEMF_DEVIATION								DEVIATION_WARN	
					DEVIATION[19:16]				
	DEVIATION[15:8]								
	DEVIATION[7:0]								
0x33 VIRTUAL_STOP_L	VIRTUAL_STOP_L[31:24]								
	VIRTUAL_STOP_L[23:16]								
	VIRTUAL_STOP_L[15:8]								
	VIRTUAL_STOP_L[7:0]								
0x34 VIRTUAL_STOP_R	VIRTUAL_STOP_R[31:24]								
	VIRTUAL_STOP_R[23:16]								
	VIRTUAL_STOP_R[15:8]								
	VIRTUAL_STOP_R[7:0]								
0x35 X_BEMF_LATCH	X_BEMF_LATCH[31:24]								
	X_BEMF_LATCH[23:16]								
	X_BEMF_LATCH[15:8]								
	X_BEMF_LATCH[7:0]								
0x36 MSCNT									
							MSCNT[9:8]		
	MSCNT[7:0]								
0x37 MSCURACT								CUR_B[8]	
	CUR_B[7:0]								
								CUR_A[8]	
	CUR_A[7:0]								
0x38 CHOPCONF			DEDGE	INTPOL	MRES				
	TPFD				VHIGHCHM	VHIGHFS		TBL[1]	
	TBL[0]	CHM		DISFDCC	FD3	HEND_OFFSET[3:1]			
	HEND_OFFSET[0]	HSTRT_TFD210			TOFF				
0x39 COOLCONF								SFILT	
		SGT							
	SEIMIN	SEDN			SEMAX				
		SEUP			SEMIN				
0x3A DCCTRL									
	DC_SG								
							DC_TIME[9:8]		
	DC_TIME[7:0]								

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)						
0x3B DRV_STATUS	STST	OLB	OLA	S2GB	S2GA	OTPW	OT	STALLGUARD
	STALLGUARD_PREWARN	RES_DET		CS_ACTUAL				
	FSACTIVE	STEALTH	S2VSB	S2VSA			SG_RESULT[9:8]	
	SG_RESULT[7:0]							
0x3C PWMCONF	PWM_LIM				PWM_REG			
	PWM_DIS_REG_STST	PWM_MEAS_SD_ENABLE	PWM_FREEWHEEL		PWM_AUTOGRAD	PWM_AUTOSCALE	PWM_FREQ	
	PWM_GRAD							
	PWM_OFS							
0x3D PWM_SCALE								PWM_SCALE_AUTO[8]
	PWM_SCALE_AUTO[7:0]							
							PWM_SCALE_SUM[9:8]	
	PWM_SCALE_SUM[7:0]							
0x3E PWM_AUTO								
	PWM_GRAD_AUTO							
	PWM_OFS_AUTO							
0x3F SG4_THRS								
							SG4_FILT_EN	SG4_THRS[8]
	SG4_THRS[7:0]							
0x40 SG4_RESULT								
							SG4_RESULT[9:8]	
	SG4_RESULT[7:0]							
0x41 SG4_IND	SG4_IND_3							
	SG4_IND_2							
	SG4_IND_1							
	SG4_IND_0							
0x42 SG_PREWARN_CONF		SG_PREWARN_FILTER						SG_PREWARN_THRSH[8]
	SG_PREWARN_THRSH[7:0]							
				SG_PREWARN_RESULT_VALID			SG_PREWARN_RESULT[9:8]	
	SG_PREWARN_RESULT[7:0]							
0x43 ADC_SUPPLY_TEMP								SUPPLY[8]
	SUPPLY[7:0]							
				ADC_EN				TEMPERATURE[8]
	TEMPERATURE[7:0]							
0x44 VMAX_LIMIT								
		VMAX_LIMIT[22:16]						
	VMAX_LIMIT[15:8]							
	VMAX_LIMIT[7:0]							

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)
0x45 LIMIT_VALUES		IHOLD_LIMIT
		IRUN_LIMIT
		GAIN_SCALER_LIMIT, RREF_LIMIT
		GLOBAL_SCALER_LIMIT
0x46 USER_DATA_0		USER_DATA_0[31:24]
		USER_DATA_0[23:16]
		USER_DATA_0[15:8]
		USER_DATA_0[7:0]
0x47 USER_DATA_1		USER_DATA_1[31:24]
		USER_DATA_1[23:16]
		USER_DATA_1[15:8]
		USER_DATA_1[7:0]
0x48 USER_DATA_2		USER_DATA_2[31:24]
		USER_DATA_2[23:16]
		USER_DATA_2[15:8]
		USER_DATA_2[7:0]
0x49 USER_DATA_3		USER_DATA_3[31:24]
		USER_DATA_3[23:16]
		USER_DATA_3[15:8]
		USER_DATA_3[7:0]
0x4A USER_DATA_4		USER_DATA_4[31:24]
		USER_DATA_4[23:16]
		USER_DATA_4[15:8]
		USER_DATA_4[7:0]
0x4B USER_DATA_5		USER_DATA_5[31:24]
		USER_DATA_5[23:16]
		USER_DATA_5[15:8]
		USER_DATA_5[7:0]
0x4C USER_DATA_6		USER_DATA_6[31:24]
		USER_DATA_6[23:16]
		USER_DATA_6[15:8]
		USER_DATA_6[7:0]
0x4D USER_DATA_7		USER_DATA_7[31:24]
		USER_DATA_7[23:16]
		USER_DATA_7[15:8]
		USER_DATA_7[7:0]
0x7D OTP_MODE		
		OTP_MODE_PASSWORD[15:8]
		OTP_MODE_PASSWORD[7:0]

0x00: GCONF

Default: 0x00000A02

Global Configuration Flags

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[12] DISABLE_DRIVER_ON_UVLO	RW 0x0	1: Disable driver (drv_en_sw is set to 0) as long as vdd_uvlo, vm_uvlo, uv_ldo in gstat are set. 0: Driver is enabled when returning from UVLO condition. Hint: Before enabling, check that gstat flags are cleared. Otherwise, the driver is immediately disabled.
	0: DISABLED 1: ENABLED	Driver reactivates after clearing UVLO condition. Driver is disabled after clearing UVLO condition.
[11] S_IREF_INTCUR_EN	RW 0x1	Selection of the reference resistor (default: internal). To change this bit, disable driver either by pin DRV_EN or drv_en_sw, wait 432 × ((1/f_clk)), then write protection is disabled, and the bit can safely be written.
	0: EXT_RES 1: INT_RES	External resistor is used. Internal resistor is used (default).
[10] SW_RESET	RW 0x0	Trigger a chip reset. Same functionality as pulse on nSLEEP. MTP burn: Do not change reset value to 1. Hint: Clears itself
	0: FUNCTIONAL 1: SW_RESET	Functional Reset
[9] DRV_EN_SW	RW 0x1	Software gating of the DRV_EN pin. Can be used to disable the driver. (Default enabled). Condition for enabling the driver: DRV_EN pin is 1. TOFF setting in CHOPCONF register != 0. drv_en_sw is set. This bit is automatically disabled when switchting to the qsc state. Write this bit to 1 when disabling qsc_sts_en as well to immediately enable the driver after the wake-up procedure. Is automatically set on a homing trigger on REFL or REFR. If the homing feature is enabled for REFR and/or REFL, this bit cannot be cleared as long as REFL and/or REFR remain at active level.
	0: DISABLED 1: ENABLED	Driver disabled. Driver enabled.
[8] QSC_STS_ENA	RW 0x0	Enable quiescent mode for low power consumption. This disables the driver by automatically setting drv_en_sw to 1 after the standstill condition is met. Is only enabled once the driver is in standstill. Check bit qsc (0x4 [15]) to see if the device is in the low-power quiescent mode. The mode is only entered once standstill is detected (see example 0x3B DRV_STATUS stst [31]). If the homing feature is enabled for REFR and/or REFL this bit cannot be set as long as REFL and/or REFR remain at active level.
	0: DISABLED 1: ENABLED	Disable low-power quiescent mode. Enable low-power quiescent mode.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7] SD	RW 0x0	When set to 1, switches off internal ramp generator and uses external S/D input.
	0: INT_MCC	Using internal motion controller.
	1: EXT_SD	Switches off internal motion controller and using external STEP/DIR inputs.
[6] DIRECT_MODE	RW 0x0	Enable direct motor phase current control using serial interface.
	0: NORMAL_MODE	Normal operation.
	1: DIRECT_MODE	Motor coil currents and polarity directly programmed using serial interface: Register XTARGET (0x2B) specifies signed coil A current (bits 8..0) and coil B current (bits 24..16). In this mode, the current is scaled by IHOLD setting (0x12). Velocity-based current regulation of StealthChop is not available in this mode. The automatic StealthChop current regulation works only for low stepper motor velocities.
[5] REFR_STOP	RW 0x0	1: Use REFR pin for sequencer stop. 0: Use REFL pin for sequencer stop. When [4] stop_enable is 1
	0: REFL_EM_STOP	REFL used for sequencer stop.
	1: REFR_EM_STOP	REFR used for sequencer stop.
[4] STOP_ENABLE	RW 0x0	Motor hard stop function enable. This leaves the motor powered on but motion is disabled.
	0: NORMAL	Normal operation.
	1: EM_STOP	Emergency stop: PIN REFx stops the sequencer depending on [5] refr_stop.
[3] SMALL_HYSTERESIS	RW 0x0	Hysteresis selection for step frequency comparison. S. TSTEP and TPWMTHRS, TCOOLTHRS, THIGH.
	0: HYST_1_16	Hysteresis for step frequency comparison is 1/16.
	1: HYST_1_32	Hysteresis for step frequency comparison is 1/32.
[2] SHAFT	RW 0x0	Change motor direction/direction sign.
	0: DIR	Default motor direction.
	1: DIR_INV	Inverse motor direction.
[1] MULTISTEP_FILT	RW 0x1	Enable step input filtering for StealthChop2.
	0: FILT_DIS	Step input filtering disabled.
	1: FILT_EN	Enable step input filtering for StealthChop2 optimization with external step source (default = 1).
[0] EN_PWM_MODE	RW 0x0	Enable the StealthChop2 mode.
	0: SPREADCYCLE	Spreadcycle
	1: STEALTHCHOP	StealthChop2 voltage PWM mode enabled (depending on velocity thresholds). Switch from off to on state while in standstill and at IHOLD = nominal IRUN current only.

0x01: GSTAT

Default: 0x00000029

Global Status Flags
(Rewrite with 1-bit to clear respective flags).

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7] VDD_UVLO	RW, W1C 0x0	V_{DD} undervoltage occurred since last time clearing this bit.
	0: OPERATIONAL	Normal operation.
	1: VDD_UVLO_DET	V_{DD} undervoltage occurred.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[6] RREFF_ERR	RW, W1C 0x0	Indicates there is an issue with the external reference resistor since the last time this bit is cleared.
	0: OPERATIONAL	Normal operation.
	1: RREF_ERR_DET	Reference resistor issue since last clearing this bit.
[5] VCCIO_RST	RW, W1C 0x1	A V_{CCIO} undervoltage occurred. Clear after initial power-up.
	0: OPERATIONAL	Normal operation.
	1: VCCIO_UVLO_DET	V_{CCIO} undervoltage happened since clearing this bit.
[4] VM_UVLO	RW, W1C 0x0	1: VM undervoltage occured since last reset. Clear after initial power-up.
	0: OPERATIONAL	Normal operation.
	1: VM_UVLO_DET	VM undervoltage occured since last reset.
[3] REGISTER_RESET	RW, W1C 0x1	Register map default reset flag. Clear after initial power-up.
	0: OPERATIONAL	Normal operation.
	1: REG_RES_DET	Indicates the register map is reset. All registers are cleared to reset values.
[2] UV_LDO	RW, W1C 0x0	LDO undervoltage condition flag. Clear after initial power-up. #type = COW
	0: OPERATIONAL	Normal operation.
	1: UV_LDO	Indicates an undervoltage on the LDO . The driver is disabled during undervoltage. This flag is latched for information. (Is also one when returning from qsc due to resetting the LDO).
[1] DRV_ERR	RW, W1C 0x0	Driver error flag. Clear after initial power-up. #type = COW
	0: OPERATIONAL	Normal operation.
	1: DRV_DIS_DET	Indicates the driver is shut down due to overtemperature, short-circuit detection, or issues with an external reference resistor. Read DRV_STATUS for details. The flag can only be cleared when the error conditions are cleared.
[0] RESET	RW, W1C 0x1	Reset flag. Clear after initial power-up. #type = COW
	0: NO_RESET	Normal operation.
	1: RESET_DET	Indicates the IC is reset.

0x03: NODECONF

Default: 0x00000001

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[0] I2C_AUT_INC	RW 0x1	1: Autoincrement of regmap address after 32-bit access. Use this bit for burst reading/writing consecutive registers. (example MSLUT_x). Hint: After a successful 32 -bit write, the address is internally incremented, while on a read the address is incremented while the first byte is sent. This register is for TMC5222 only.
	0: DISABLED	Register map address stays the same after successful read or write access.
	1: ENABLED	Register map address is incremented after successful read or write access.

0x04: IOIN

Default: 0x00000000

Returns information on IC revision, current operational state, and utilized clock.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:28] VERSION	R, unsigned 0x0	0x3 (Identical numbers mean full digital compatibility).
[26] ONSTATE	R 0x0	1: Device is fully operational and the driver can be enabled. 0: Driver is disabled. If the device is not in qsc mode, then it is currently processing an error condition or booting.
	0: DISABLED	Device is booting, in quiescent mode, or in fault condition.
	1: FUNCTIONAL	Device fully operational.
[25] QSC	R 0x0	1: Device is in low-power quiescent mode.
	0: OPERATIONAL	Device is not in low-power quiescent mode.
	1: QSC	Device is in low-power quiescent mode.
[24:23] EXT_CLK	R 0x0	Clock source indicator flag.
	0: CLK_INT	The internal oscillator is used for generating the clock-signal (12.5MHz typ).
	1: CLK_EXT	The external oscillator is used for generating the clock-signal.
	2: CLK_INT_QSC	The second internal oscillator is used for generating the slow clock-signal (609 kHz typ) in the low-power quiescent mode.
	3: NOT_USED	Not used.
[22:20] SILICON_RV	R, unsigned 0x0	Silicon revision number

0x05: DRV_CONF

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[18:16] STANDSTILL_TIME	RW 0x0	Determines how many CLK cycles must pass until the driver signals standstill (s. 0x3B DRV_STATUS bit 31 stst) after a STEP pulse or the motion controller has stopped. Time until standstill: (1/f_clk) × 2^((20 - STANDSTILL_TIME))
	0: T_STST_20	Time until standstill: t_clk × 220
	1: T_STST_19	Time until standstill: t_clk × 219
	2: T_STST_18	Time until standstill: t_clk × 218
	3: T_STST_17	Time until standstill: t_clk × 217
	4: T_STST_16	Time until standstill: t_clk × 216
	5: T_STST_15	Time until standstill: t_clk × 215
	6: T_STST_14	Time until standstill: t_clk × 214
	7: T_STST_13	Time until standstill: t_clk × 213
[3:2] FS_GAIN	RW 0x0	Scales the reference current I_{REF}. Use this together with FS_SENSE for fine-tuning the current scaling. Selected current: 0.125 A × [(FS_SENSE + 1)] × [(FS_GAIN + 1)]
	0: GAIN_025	25% I_{REF}
	1: GAIN_05	50% I_{REF}
	2: GAIN_075	75% I_{REF}
	3: GAIN_100	100% I_{REF}

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[1:0] FS_SENSE	RW 0x0	Full-Scale Range The full-scale bits allow for a basic adaptation of the drivers $R_{DS(ON)}$ current sensing to the motor current range. Select the lowest fitting range for best current precision. The full-scale bits also define the overcurrent protection threshold (this value is the peak current setting). The $R_{DS(ON)}$ and overcurrent protection threshold values are given in the EC table under output specifications and protection circuits. Selected current: 0.125 A × [(FS_SENSE + 1)] × [(FS_GAIN + 1)]
	0: RDSON_200 1: RDSON_100 2: RDSON_67 3: RDSON_50	KIFS = 4.10, typ. $R_{DS(ON),LS}$ = 0.2Ω KIFS = 8.16, typ. $R_{DS(ON),LS}$ = 0.1Ω KIFS = 12.06, typ. $R_{DS(ON),LS}$ = 0.067Ω KIFS = 16.00, typ. $R_{DS(ON),LS}$ = 0.05Ω

0x06: GLOBAL_SCALER

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[15:8] GLOBALSCALER_B	RW, unsigned 0x00	Global scaling of motor current. This value is multiplied to the current scaling to adapt a drive to a certain motor type. This value should be chosen before tuning other settings, because it also influences chopper hysteresis. Scales Phase B. 0: Full-scale (or write 256). 1 to 31: Not allowed for operation. 32 to 255: 32/256 to 255/256 of maximum current. Hint: Values >128 recommended for best results Hint: Different values for GLOBALSCALER_B and GLOBALSCALER_A are only supported in SpreadCycle mode.
[7:0] GLOBALSCALER_A	RW, unsigned 0x00	Global scaling of motor current. This value is multiplied to the current scaling to adapt a drive to a certain motor type. This value should be chosen before tuning other settings, because it also influences chopper hysteresis. Scales Phase A. 0: Full-scale (or write 256). 1 to 31: Not allowed for operation. 32 to 255: 32/256 to 255/256 of maximum current. Hint: Values >128 recommended for best results. Hint: Different values for GLOBALSCALER_B and GLOBALSCALER_A are only supported in SpreadCycle mode.

0x07: DIAG_CONF

Default: 0x00000000

DOx pin configuration. Each selected signal is combined by logic or function on the DOx pin.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31] DIAG1_INVPP	RW 0x0	Set the active polarity of DIAG1 in push-pull mode.
	0: HI_ACTIVE 1: LO_ACTIVE	Push-pull is high active. Push-pull is low active.
[30] DIAG1_NOD_PP	RW 0x0	Configures DIAG1 pin as open-drain (low active, HiZ) or push-pull.
	0: OPEN_DRAIN 1: PUSH_PULL	Open_drain Push_pull
[29] DIAG0_INVPP	RW 0x0	Set the active polarity of DIAG0 in push-pull mode.
	0: HI_ACTIVE 1: LO_ACTIVE	Push-pull is high active. Push-pull is low active.
[28] DIAG0_NOD_PP	RW 0x0	Configures DIAG0 pin as open-drain (low active, HiZ) or push-pull.
	0: OPEN_DRAIN 1: PUSH_PULL	Open_drain Push_pull
[27] DIAG1_POSITION_REACHED	RW 0x0	Position_reached of 0x2E RAMP_STAT.
	0: DISABLED 1: ENABLED	Disabled Position_reached on DIAG1.
[26] DIAG1_EV_HOMING	RW 0x0	Enables refr_homing | refl_homing of 0x2E RAMP_STAT.
	0: DISABLED 1: ENABLED	Disabled refx_homing on DIAG1.
[25] DIAG1_EV_DEVIATION	RW 0x0	Enables deviation_warn of 0x32 BEMF_DEVIATION.
	0: DISABLED 1: ENABLED	Disabled Deviation warn on DIAG1.
[24] DIAG1_EV_POS_REACHED	RW 0x0	Enables event_pos_reached of 0x2E RAMP_STAT.
	0: DISABLED 1: ENABLED	Disabled event_pos_reached on DIAG1.
[23] DIAG1_EV_STOP_SG	RW 0x0	Enables event_stop_sg of 0x2E RAMP_STAT
	0: DISABLED 1: ENABLED	Disabled event_stop_sg on DIAG1.
[22] DIAG1_EV_STOP	RW 0x0	Enables (event_stop_r | event_stop_l) of 0x2E RAMP_STAT.
	0: DISABLED 1: ENABLED	Disabled event_stop_x on DIAG1.
[21] DIAG1_XCOMP	RW 0x0	Enables x_compare pulse: if(X_COMPARE == XACTUAL) DIAG1 = active else, DIAG1 = inactive
	0: DISABLED 1: ENABLED	Disabled x_compare pulse on DIAG1.
[20] DIAG1_DIR	RW 0x0	Enables dir output of internal ramp generator on DIAG1.
	0: DISABLED 1: ENABLED	Disabled Dir on DIAG1.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[19] DIAG1_STEP	RW 0x0	Enables step output of internal ramp generator on DO1.
	0: DISABLED 1: ENABLED	Disabled Step on DIAG1.
[18] DIAG1_INDEX	RW 0x0	Enables index on DIAG1: When the microstep table crosses its entry on position 0, this is 1.
	0: DISABLED 1: ENABLED	Disabled Index pulse on DIAG1.
[17] DIAG1_STALL	RW 0x0	Enables StallGuard of 0x3B on DIAG1. TCOOLTHRS must be reached to see the stall on DIAG1.
	0: DISABLED 1: ENABLED	Disabled StallGuard on DIAG1.
[16] DIAG1_STALL_PREWARN	RW 0x0	Enables stallguard_prewarn on DIAG1.
	0: DISABLED 1: ENABLED	Disabled StallGuard_prewarn on DIAG1.
[15] DIAG1_OTPW	RW 0x0	Propagate overtemperature prewarning (otpw) to DIAG1 once the ADC measures values above 120°C.
	0: DISABLED 1: ENABLED	Disabled otpw on DIAG1
[14] DIAG1_ERROR	RW 0x0	Indicates the driver is shut down due to overtemperature, short circuit detection, or issues with an external reference resistor.
	0: DISABLED 1: ENABLED	Disabled Error condition on DIAG1.
[13] DIAG0_POSITION_REACHED	RW 0x0	Position_reached of 0x2E RAMP_STAT.
	0: DISABLED 1: ENABLED	Disabled Position_reached on DIAG0.
[12] DIAG0_EV_HOMING	RW 0x0	Enables refr_homing | refl_homing of 0x2E RAMP_STAT.
	0: DISABLED 1: ENABLED	Disabled refx_homing on DIAG0.
[11] DIAG0_EV_DEVIATION	RW 0x0	Enables deviation_warn of 0x32 BEMF_DEVIATION.
	0: DISABLED 1: ENABLED	Disabled Deviation warn on DIAG0.
[10] DIAG0_EV_POS_REACHED	RW 0x0	Enables event_pos_reached of 0x2E RAMP_STAT.
	0: DISABLED 1: ENABLED	Disabled event_pos_reached on DIAG0.
[9] DIAG0_EV_STOP_SG	RW 0x0	Enables event_stop_sg of 0x2E RAMP_STAT.
	0: DISABLED 1: ENABLED	Disabled event_stop_sg on DIAG0.
[8] DIAG0_EV_STOP_REF	RW 0x0	Enables (event_stop_r | event_stop_l) of 0x2E RAMP_STAT.
	0: DISABLED 1: ENABLED	Disabled event_stop_x on DIAG0.
[7] DIAG0_XCOMP	RW 0x0	Enables x_compare pulse: if(X_COMPARE == XACTUAL) DIAG0 = active else, DIAG0 = inactive
	0: DISABLED 1: ENABLED	Disabled x_compare pulse on DIAG0.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[6] DIAG0_DIR	RW 0x0	Enables dir output of internal ramp generator on DIAG0.
	0: DISABLED 1: ENABLED	Disabled dir on DIAG0.
[5] DIAG0_STEP	RW 0x0	Enables step output of internal ramp generator on DIAG0.
	0: DISABLED 1: ENABLED	Disabled step on DIAG0
[4] DIAG0_INDEX	RW 0x0	Enables index on DIAG0: When the microstep table crosses its entry on position 0, this is 1.
	0: DISABLED 1: ENABLED	Disabled index pulse on DIAG0.
[3] DIAG0_STALL	RW 0x0	Enables StallGuard of 0x3B on DIAG0. TCOOLTHRS must be reached to see the stall on DIAG0.
	0: DISABLED 1: ENABLED	Disabled stall on DIAG0
[2] DIAG0_STALL_PREWARN	RW 0x0	Enables stallguard_prewarn on DIAG0.
	0: DISABLED 1: ENABLED	Disabled StallGuard_prewarn on DIAG0.
[1] DIAG0_OTPW	RW 0x0	Propagate overtemperature prewarning (otpw) to DO0 once the ADC measures values above 120°C.
	0: DISABLED 1: ENABLED	Disabled otpw propagated to DO0.
[0] DIAG0_ERROR	RW 0x0	Indicates the driver is shut down due to overtemperature, short-circuit detection, or issues with an external reference resistor.
	0: DISABLED 1: ENABLED	Disabled Error condition propagated to DO0.

0x08: MSLUT0

Default: 0xAAAAB554

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] MSLUT_0	RW, unsigned 0xAAAAB554	Each bit gives the difference between entry x and entry x + 1 when combined with the corresponding MSLUTSEL W bits: 0: W = %00: -1 %01: +0 %10: +1 %11: +2 1: W = %00: +0 %01: +1 %10: +2 %11: +3 This is the differential coding for the first quarter of a wave. Start values for CUR_A and CUR_B are stored for MSCNT position 0 in START_SIN and START_SIN90. ofs31, ofs30, …, ofs01, ofs00 … ofs255, ofs254, …, ofs225, ofs224 Reset default = sine-wave table

0x09: MSLUT1

Default: 0x4A9554AA

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] MSLUT_1	RW, unsigned 0x4A9554AA	Each bit gives the difference between entry x and entry x + 1 when combined with the corresponding MSLUTSEL W bits: 0: W = %00: -1 %01: +0 %10: +1 %11: +2 1: W = %00: +0 %01: +1 %10: +2 %11: +3 This is the differential coding for the first quarter of a wave. Start values for CUR_A and CUR_B are stored for MSCNT position 0 in START_SIN and START_SIN90. ofs31, ofs30, …, ofs01, ofs00 … ofs255, ofs254, …, ofs225, ofs224 Reset default = sine-wave table

0x0A: MSLUT2

Default: 0x24492929

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] MSLUT_2	RW, unsigned 0x24492929	Each bit gives the difference between entry x and entry x + 1 when combined with the corresponding MSLUTSEL W bits: 0: W = %00: -1 %01: +0 %10: +1 %11: +2 1: W = %00: +0 %01: +1 %10: +2 %11: +3 This is the differential coding for the first quarter of a wave. Start values for CUR_A and CUR_B are stored for MSCNT position 0 in START_SIN and START_SIN90. ofs31, ofs30, …, ofs01, ofs00 … ofs255, ofs254, …, ofs225, ofs224 Reset default = sine-wave table

0x0B: MSLUT3

Default: 0x10104222

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] MSLUT_3	RW, unsigned 0x10104222	Each bit gives the difference between entry x and entry x + 1 when combined with the corresponding MSLUTSEL W bits: 0: W = %00: -1 %01: +0 %10: +1 %11: +2 1: W = %00: +0 %01: +1 %10: +2 %11: +3 This is the differential coding for the first quarter of a wave. Start values for CUR_A and CUR_B are stored for MSCNT position 0 in START_SIN and START_SIN90. ofs31, ofs30, …, ofs01, ofs00 … ofs255, ofs254, …, ofs225, ofs224 Reset default = sine-wave table

0x0C: MSLUT4

Default: 0xFBFFFFFF

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] MSLUT_4	RW, unsigned 0xFBFFFFFF	Each bit gives the difference between entry x and entry x + 1 when combined with the corresponding MSLUTSEL W bits: 0: W = %00: -1 %01: +0 %10: +1 %11: +2 1: W = %00: +0 %01: +1 %10: +2 %11: +3 This is the differential coding for the first quarter of a wave. Start values for CUR_A and CUR_B are stored for MSCNT position 0 in START_SIN and START_SIN90. ofs31, ofs30, …, ofs01, ofs00 … ofs255, ofs254, …, ofs225, ofs224 Reset default = sine-wave table

0x0D: MSLUT5

Default: 0xB5BB777D

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] MSLUT_5	RW, unsigned 0xB5BB777D	Each bit gives the difference between entry x and entry x + 1 when combined with the corresponding MSLUTSEL W bits: 0: W = %00: -1 %01: +0 %10: +1 %11: +2 1: W = %00: +0 %01: +1 %10: +2 %11: +3 This is the differential coding for the first quarter of a wave. Start values for CUR_A and CUR_B are stored for MSCNT position 0 in START_SIN and START_SIN90. ofs31, ofs30, …, ofs01, ofs00 … ofs255, ofs254, …, ofs225, ofs224 Reset default = sine-wave table

0x0E: MSLUT6

Default: 0x49295556

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] MSLUT_6	RW, unsigned 0x49295556	Each bit gives the difference between entry x and entry x + 1 when combined with the corresponding MSLUTSEL W bits: 0: W = %00: -1 %01: +0 %10: +1 %11: +2 1: W = %00: +0 %01: +1 %10: +2 %11: +3 This is the differential coding for the first quarter of a wave. Start values for CUR_A and CUR_B are stored for MSCNT position 0 in START_SIN and START_SIN90. ofs31, ofs30, …, ofs01, ofs00 … ofs255, ofs254, …, ofs225, ofs224 Reset default = sine-wave table

0x0F: MSLUT7

Default: 0x00404222

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] MSLUT_7	RW, unsigned 0x00404222	Each bit gives the difference between entry x and entry x + 1 when combined with the corresponding MSLUTSEL W bits: 0: W = %00: -1 %01: +0 %10: +1 %11: +2 1: W = %00: +0 %01: +1 %10: +2 %11: +3 This is the differential coding for the first quarter of a wave. Start values for CUR_A and CUR_B are stored for MSCNT position 0 in START_SIN and START_SIN90. ofs31, ofs30, …, ofs01, ofs00 … ofs255, ofs254, …, ofs225, ofs224 Reset default = sine-wave table

0x10: MSLUT_START

Default: 0x00F70000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:24] OFFSET_SIN90	RW, signed 0x00	Signed offset for CUR_B ±127 microsteps. Adapt START_SIN90 to match the sine-wave table at position 0.
[23:16] START_SIN90	RW, unsigned 0xF7	START_SIN90 sets the absolute current for the microstep table entry at positions 256.
[7:0] START_SIN	RW, unsigned 0x00	START_SIN sets the absolute current at microstep table entry 0.

0x11: MSLUT_SEL

Default: 0xFFFF8056

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:24] X3	RW, unsigned 0xFF	LUT segment 3 start. The sine-wave lookup table can be divided into up to four segments using an individual step width control entry Wx . The segment borders are selected by X1, X2, and X3. Segment 0 goes from 0 to X1 - 1. Segment 1 goes from X1 to X2 - 1. Segment 2 goes from X2 to X3 - 1. Segment 3 goes from X3 to 255. For a defined response, the values shall satisfy: 0 < X1 < X2 < X3

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[23:16] X2	RW, unsigned 0xFF	LUT segment 2 start. The sine-wave lookup table can be divided into up to four segments using an individual step width control entry Wx . The segment borders are selected by X1, X2, and X3. Segment 0 goes from 0 to X1 - 1. Segment 1 goes from X1 to X2 - 1. Segment 2 goes from X2 to X3 - 1. Segment 3 goes from X3 to 255. For a defined response, the values shall satisfy: 0 < X1 < X2 < X3
[15:8] X1	RW, unsigned 0x80	LUT segment 1 start. The sine-wave lookup table can be divided into up to four segments using an individual step width control entry Wx . The segment borders are selected by X1, X2, and X3. Segment 0 goes from 0 to X1 - 1. Segment 1 goes from X1 to X2 - 1. Segment 2 goes from X2 to X3 - 1. Segment 3 goes from X3 to 255. For a defined response, the values shall satisfy: 0 < X1 < X2 < X3
[7:6] W3	RW 0x1	LUT width select from ofs (X3) to ofs 255. Width control bit coding W0 to W3: %00: MSLUT entry 0, 1 select: -1, +0 %01: MSLUT entry 0, 1 select: +0, +1 %10: MSLUT entry 0, 1 select: +1, +2 %11: MSLUT entry 0, 1 select: +2, +3
	0: W3_SUB1_ADD0 Current MSLUT entry 0 (1): -1 (+0) to sine-wave. 1: W3_ADD0_ADD1 Current MSLUT entry 0 (1): +0 (+1) to sine-wave. 2: W3_ADD1_ADD2 Current MSLUT entry 0 (1): +1 (+2) to sine-wave. 3: W3_ADD2_ADD3 Current MSLUT entry 0 (1): +2 (+3) to sine-wave.	
[5:4] W2	RW 0x1	LUT width select from ofs (X2) to ofs (X3 - 1). Width control bit coding W0 to W3: %00: MSLUT entry 0, 1 select: -1, +0 %01: MSLUT entry 0, 1 select: +0, +1 %10: MSLUT entry 0, 1 select: +1, +2 %11: MSLUT entry 0, 1 select: +2, +3
	0: W2_SUB1_ADD0 Current MSLUT entry 0 (1): -1 (+0) to sine-wave. 1: W2_ADD0_ADD1 Current MSLUT entry 0 (1): +0 (+1) to sine-wave. 2: W2_ADD1_ADD2 Current MSLUT entry 0 (1): +1 (+2) to sine-wave. 3: W2_ADD2_ADD3 Current MSLUT entry 0 (1): +2 (+3) to sine-wave.	
[3:2] W1	RW 0x1	LUT width select from ofs (X1) to ofs (X2 - 1). Width control bit coding W0 to W3: %00: MSLUT entry 0, 1 select: -1, +0 %01: MSLUT entry 0, 1 select: +0, +1 %10: MSLUT entry 0, 1 select: +1, +2 %11: MSLUT entry 0, 1 select: +2, +3
	0: W1_SUB1_ADD0 Current MSLUT entry 0 (1): -1 (+0) to sine-wave. 1: W1_ADD0_ADD1 Current MSLUT entry 0 (1): +0 (+1) to sine-wave. 2: W1_ADD1_ADD2 Current MSLUT entry 0 (1): +1 (+2) to sine-wave. 3: W1_ADD2_ADD3 Current MSLUT entry 0 (1): +2 (+3) to sine-wave.	

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[1:0] W0	RW 0x2	LUT width select from ofs00 to ofs (X1 - 1). Width control bit coding W0 to W3: %00: MSLUT entry 0, 1 select: -1, +0 %01: MSLUT entry 0, 1 select: +0, +1 %10: MSLUT entry 0, 1 select: +1, +2 %11: MSLUT entry 0, 1 select: +2, +3
	0: W0_SUB1_ADD0 1: W0_ADD0_ADD1 2: W0_ADD1_ADD2 3: W0_ADD2_ADD3	Current MSLUT entry 0 (1): -1 (+0) to sine-wave. Current MSLUT entry 0 (1): +0 (+1) to sine-wave. Current MSLUT entry 0 (1): +1 (+2) to sine-wave. Current MSLUT entry 0 (1): +2 (+3) to sine-wave.

0x12: X_COMPARE

Default: 0xFFFFFFFF

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] X_COMPARE	RW, signed 0xFFFFFFFF	Position comparison register for motion controller position strobe. X_COMPARE is an absolute position. The position pulse is available on output DOx when enabled in DIAG_CONF (0x7). XACTUAL = X_COMPARE: Output signal (position pulse) is high. It returns to a low state, if the positions mismatch. If X_COMPARE_REPEAT is >1, X_COMPARE is the position reference for the periodic position strobe trigger output.

0x13: X_COMPARE_REPEAT

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[23:0] X_COMPARE_REPEAT	RW, unsigned 0x000000	This register defines a relative distance in microsteps (based on MRES confguration). If set to >1, the position compare pulse is raised every time a multiple of X_COMPARE_REPEAT µsteps is made. Thereby, the X_COMPARE register defines the base position for the modulo calculation of X_COMPARE_REPEAT µsteps are made into positive or negative direction.

0x14: IHOLD_IRUN

Default: 0x04011F08

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[27:24] IRUNDELAY	RW 0x4	Controls the number of clock cycles for motor power-up before a motion is started. The smooth transition avoids a motor jerk upon power-up. 0: Instant power-up. 1 to 15: Delay per current increase step in multiples of IRUNDELAY × 512 clocks. When powering up in standtill, IRUNDELAY is used until IHOLD is reached in case IHOLDDELAY = 0.
	0: INSTANT	Instant jump to IRUN on start of a motion.
	1: DELAY_512	Delay of 512 × t_clk per current increment.
	2: DELAY_1024	Delay of 1024 × t_clk per current increment.
	3: DELAY_1536	Delay of 1536 × t_clk per current increment.
	4: DELAY_2048	Delay of 2048 × t_clk per current increment.
	5: DELAY_2560	Delay of 2560 × t_clk per current increment.
	6: DELAY_3072	Delay of 3072 × t_clk per current increment.
	7: DELAY_3584	Delay of 3584 × t_clk per current increment.
	8: DELAY_4096	Delay of 4096 × t_clk per current increment.
	9: DELAY_4608	Delay of 4608 × t_clk per current increment.
	10: DELAY_5120	Delay of 5120 × t_clk per current increment.
	11: DELAY_5632	Delay of 5632 × t_clk per current increment.
	12: DELAY_6144	Delay of 6144 × t_clk per current increment.
	13: DELAY_6656	Delay of 6656 × t_clk per current increment.
	14: DELAY_7168	Delay of 7168 × t_clk per current increment.
	15: DELAY_7680	Delay of 7680 × t_clk per current increment.
[19:16] IHOLDDELAY	RW 0x1	Controls the number of clock cycles for motor power-down after a motion as soon as standstill is detected (stst = 1) and TPOWERDOWN has expired. The smooth transition avoids a motor jerk upon power-down. 0: Instant power-down. 1 to 15: Delay per current reduction step in multiples of 2^(18) clocks.
	0: INSTANT	Instant reduction to IHOLD current after TPOWERDOWN has expired.
	1: DELAY_1_218	One decrement every 2^18 × t_clk.
	2: DELAY_2_218	One decrement every 2 × 2^18 × t_clk.
	3: DELAY_3_218	One decrement every 3 × 2^18 × t_clk.
	4: DELAY_4_218	One decrement every 4 × 2^18 × t_clk.
	5: DELAY_5_218	One decrement every 5 × 2^18 × t_clk.
	6: DELAY_6_218	One decrement every 6 × 2^18 × t_clk.
	7: DELAY_7_218	One decrement every 7 × 2^18 × t_clk.
	8: DELAY_8_218	One decrement every 8 × 2^18 × t_clk.
	9: DELAY_9_218	One decrement every 9 × 2^18 × t_clk.
	10: DELAY_10_218	One decrement every 10 × 2^18 × t_clk.
	11: DELAY_11_218	One decrement every 11 × 2^18 × t_clk.
	12: DELAY_12_218	One decrement every 12 × 2^18 × t_clk.
	13: DELAY_13_218	One decrement every 13 × 2^18 × t_clk.
	14: DELAY_14_218	One decrement every 14 × 2^18 × t_clk.
	15: DELAY_15_218	One decrement every 15 × 2^18 × t_clk.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[12:8] IRUN	RW, unsigned 0x1F	Motor run current. Device uses this current when a motion is detected and when not in standstill. IRUNDELAY is used to ramp up the current to the value of this register. (Current scaled to (IRUN + 1)/32)). 0 = 1/32 1 = 2/32 to 31 = 32/32 Hint: A setting from 16 to 31 is preferred for best microstep performance.
[4:0] IHOLD	RW, unsigned 0x08	Standstill current Once the device detects standstill, the current is reduced to this value using the IHOLDDELAY setting. (Current scaled to (IHOLD + 1)/32)). 0 = 1/32 1 = 2/32 to 31 = 32/32 In combination with StealthChop mode, setting IHOLD = 0 allows to choose freewheeling or coil short circuit for motor standstill.

0x15: TPOWERDOWN

Default: 0x0000000A

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7:0] TPOWERDOWN	RW, unsigned 0x0A	TPOWERDOWN sets the delay time after standstill (stst) of the motor to motor current powering down to IHOLD. Time range is about 0 seconds to 4 seconds. Attention: A minimum setting of 2 is required to allow the automatic tuning of StealthChop PWM_OFFS_AUTO. Reset default = 10 (0 to ((2^(8)) - 1)) × 2^(18) tCLK

0x16: TSTEP

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[19:0] TSTEP	R, unsigned 0x00000	Actual measured time between two 1/256 microsteps derived from the step input frequency in units of 1/f CLK. Measured value is (2^(20)) - 1 in case of overflow or standstill. All TSTEP-related thresholds (Txxx) use a hysteresis of 1/16 of the compare value to compensate for jitter in the clock or the step frequency. The flag small_hysteresis in GCONF (0x0) modifies the hysteresis to a smaller value of 1/32. (Txxx x 15/16) – 1, or, (Txxx x 31/32) - 1 is used as a second compare value for each comparison value. This means the lower switching velocity equals the calculated setting, but the upper switching velocity is higher, as defined by the hysteresis setting. When working with the motion controller, the measured TSTEP for a given velocity V is in the range: (2^(24)/V) ≤ TSTEP ≤ 2^(24)/V - 1. In DcStep mode, TSTEP does not show the mean velocity of the motor, but the velocities for each microstep, which may not be stable, and thus, does not represent the real motor velocity in case it runs slower than the target velocity.

0x17: TPWMTHRS

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[19:0] TPWMTHRS	RW, unsigned 0x00000	This is the upper velocity for StealthChop voltage PWM mode. TSTEP ≥ TPWMTHRS StealthChop PWM mode is enabled, if configured. DcStep is disabled.

0x18: TCOOLTHRS

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[19:0] TCOOLTHRS	RW, unsigned 0x00000	This is the lower threshold velocity for switching on smart energy CoolStep and StallGuard feature. (Unsigned). Set this parameter to disable CoolStep at low speeds, where it cannot work reliably. The stop-on-stall function (enable with sg_stop when using internal motion controller) and the stall output signal isenabled when exceeding this velocity. In non-DcStep mode, it is disabled again once the velocity falls below this threshold. TCOOLTHRS ≥ TSTEP ≥ THIGH: CoolStep is enabled, if configured. TCOOLTHRS ≥ TSTEP Stop-on-stall is enabled, if configured. Stall output signal (DO0/1) is enabled, if configured.

0x19: THIGH

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[19:0] THIGH	RW, unsigned 0x00000	This velocity setting allows velocity dependent switching into a different chopper mode and fullstepping to maximize torque. (Unsigned). The stall detection feature is switched off for 2 to 3 electrical periods whenever passing the THIGH threshold to compensate for the effect of switching modes. TSTEP ≤ THIGH: CoolStep is disabled (motor runs with normal current scale). StealthChop voltage PWM mode is disabled. If vhighchm is set, the chopper switches to chm = 1 with TFD = 0 (constant off-time with slow decay only). If vhighfs is set, the motor operates in fullstep mode and the stall detection is switched over to DcStep stall detection.

0x1A: RAMPMODE

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[1:0] RAMPMODE	RW 0x0	Motion controller ramping mode
	0: POS_MODE	Positioning mode (using all A, D, and V parameters, as well as XTARGET).
	1: VEL_POS	Velocity mode to positive VMAX (using A1, A2, AMAX acceleration if V1, V2 are configured).
	2: VEL_NEG	Velocity mode to negative VMAX (using A1, A2, AMAX acceleration if V1, V2 are configured).
	3: HOLD	Hold mode (velocity remains unchanged, unless stop event occurs).

0x1B: XACTUAL

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] XACTUAL	RW, signed 0x00000000	Actual motor position (signed). Hint: This value normally should only be modified, when homing the drive. In positioning mode, modifying the register content starts a motion when XACTUAL does not equal XTARGET and ramp parameters allow for it. To manipulate XACTUAL in position mode without motion, make sure the motor is stopped and set VMAX = 0 and VSTART = 0 before changing XACTUAL, or change the XACTUAL after selecting hold mode in the register RAMPMODE.

0x1C: VACTUAL

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[23:0] VACTUAL	R, signed 0x000000	Actual motor velocity from ramp generator (signed). The sign matches the motion direction. A negative sign means motion to lower XACTUAL . ±(2^(23)) - 1 [µsteps/t]

0x1D: AACTUAL

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[23:0] AACTUAL	R, signed 0x000000	Current acceleration used by the ramp generator. 0 to (2^(24)) - 1 [µsteps/ta²]

0x1E: VSTART

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[17:0] VSTART	RW, unsigned 0x00000	Motor start velocity (unsigned). For universal use, set VSTOP ≥ VSTART. This is not required if the motion distance is sufficient to ensure deceleration from VSTART to VSTOP. Only used in position mode. 0 to (2^(18)) - 1 [µsteps/t]

0x1F: A1

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[17:0] A1	RW, unsigned 0x00000	First acceleration between VSTART and V1 (unsigned). 0 to (2^(18)) - 1 [µsteps/ta²]

0x20: V1

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[19:0] V1	RW, unsigned 0x00000	First acceleration/deceleration phase threshold velocity (unsigned). 0: Disables A1 and D1 phase, use AMAX, DMAX only. 0 to (2^(20)) - 1 [µsteps/t]

0x21: A2

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[17:0] A2	RW, unsigned 0x00000	Acceleration between V1 and V2 (unsigned). 0 to (2^(18)) - 1 [µsteps/ta²]

0x22: V2

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[19:0] V2	RW, unsigned 0x00000	Velocity difference from VMAX for activation of acceleration segments with A2 and D2. 0: Disables A2 and D2 phase, use AMAX, DMAX only. 0 to (2^(20)) - 1 [µsteps/t]

0x23: AMAX

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[17:0] AMAX	RW, unsigned 0x00000	Second acceleration between V2 and VMAX (unsigned). This is the acceleration and deceleration value for velocity mode if V1 and V2 are 0 0 to (2^(18)) - 1 [µsteps/ta²]

0x24: VMAX

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[22:0] VMAX	RW, unsigned 0x000000	Motion ramp target velocity (for positioning, ensure VMAX ≥ VSTART (unsigned) This is the target velocity in velocity mode. It can be changed any time during a motion. 0 to (2^(23)) - 512 [µsteps/t]

0x25: DMAX

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[17:0] DMAX	RW, unsigned 0x00000	Deceleration between VMAX and V2 (unsigned) in position mode or when using soft-stop. 0 to (2^(18)) - 1 [µsteps/ta²]

0x26: D2

Default: 0x00000010

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[17:0] D2	RW, unsigned 0x00010	Deceleration between V2 and V1 (unsigned) in position mode or when using soft-stop. Attention: Do not set 0 in positioning mode, even if V2 = 0. 1 to (2^(18)) - 1 [µsteps/ta²] Reset default = 10

0x27: D1

Default: 0x0000000A

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[17:0] D1	RW, unsigned 0x0000A	Deceleration between V1 and VSTOP (unsigned) in position mode or when using soft-stop. Attention: Do not set 0 in positioning mode, even if V1 = 0. 1 to (2^(16)) - 1 [µsteps/ta²] Reset default = 10

0x28: VSTOP

Default: 0x0000000A

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[17:0] VSTOP	RW, unsigned 0x0000A	Motor stop velocity (unsigned) in position mode. Hint: Set VSTOP ≥ VSTART to allow positioning for short distances. Attention: Do not set 0 in positioning mode, minimum 10 recommended. 1 to (2^(18)) - 1 [µsteps/t] Reset default = 10

0x29: TZEROWAIT

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[15:0] TZEROWAIT	RW, unsigned 0x0000	Defines the waiting time after ramping down to zero velocity before next movement or direction inversion can start. Time range is about 0 seconds to 2 seconds. This setting avoids excess acceleration, example, from VSTOP to VSTART. This setting is not used in velocity mode. (0 to (2^(16) - 1)) × 512 tCLK

0x2A: TVMAX

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[15:0] TVMAX	RW, unsigned 0x0000	Minimum time for constant velocity segments in multiples of 512 clocks. 0: Disables minimum duration setting for constant velocity phase. >0: A minimum duration of constant velocity is inserted in between any change from acceleration to deceleration or vice versa to reduce jerk. (0 to (2^(16)- 1)) × 512 t CLK

0x2B: XTARGET

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] XTARGET	RW, signed 0x00000000	Target position for ramp mode (signed). Write a new target position to this register to activate the ramp generator positioning in RAMPMODE = 0. Initialize all velocity, acceleration, and deceleration parameters before. The position is allowed to wrap around, thus, XTARGET value optionally can be treated as an unsigned number. The maximum possible displacement is ±(2^(31) - 1). When increasing V1, V2, D1, D2, or DMAX during a motion, rewrite XTARGET afterwards to trigger a second acceleration phase, if desired. -2^(31) to (+2^(31) - 1)

0x2C: VDCMIN

Default: 0x00000000

dcStep start velocity.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[22:0] VDCMIN	RW, unsigned 0x000000	DcStep automatic commutation is enabled above velocity VDCMIN (unsigned). DcStep is only available when using the internal ramp generator. DcStep is not available in step and direction mode. In this mode, the actual position is determined by the sensorless motor commutation and is fed back to XACTUAL. In case the motor is heavily loaded, VDCMIN also is used as the minimum step velocity. Activate stop-on-stall (sg_stop) to detect step loss. 0: Disable, DcStep off |VACT| ≥ VDCMIN ≥ 256: Triggers the same actions as exceeding THIGH setting. Switches on automatic commutation DcStep. Remember to also set DCCTRL parameters to operate DcStep. (Only bits 22:8 are used for value and comparison).

0x2D: SW_MODE

Default: 0x00000000

Switch mode configuration.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[16] EN_AUTO_FEATURE	RW 0x0	Enables movement between the two positions defined in VIRTUAL_STOP_L (0x33) and VIRTUAL_STOP_R (0x34) using the configured ramp parameters. The device updates the register X_TARGET with VIRTUAL_STOP_L content when reaching VIRTUAL_STOP_R position and vice versa. Procedure: Drive to one of the positions defined in the VIRTUAL_STOP_x registers. After reaching the position, enable this bit to start the movement. If left alone, this continues until disabling the feature. Writing to X_TARGET using the COM-Interface clears this bit and disables the procedure. When used in combination with the homing interrupt, this bit is cleared on an occurring interrupt and the procedure is disabled.
	0: DISABLED 1: ENABLED	Disabled. Last automatic position update drive is finished. Continuous movement between positions stored in VIRTUAL_STOP_L/R.
[15] EN_REFR_HOMING	RW 0x0	With rising edge on REFR X_TARGET is set to the value of VIRTUAL_STOP_R. If the device is in qsc mode, qsc_sts_ena is cleared and drv_en_sw is set (both 0x0 GCONF). After the device reaches hold current IHOLD (0x14), the movement is triggered by writing the value of VIRTUAL_STOP_R to X_TARGET.
	0: DISABLED 1: ENABLED	Disabled. Active REFL interrupts are cleared, while the position drive continues. Triggers a homing event on the rising edge of REFR.
[14] EN_REFL_HOMING	RW 0x0	With rising edge on REFL X_TARGET is set to the value of VIRTUAL_STOP_L. If the device is in qsc mode, qsc_sts_ena is cleared and drv_en_sw is set (both 0x0 GCONF). After the device reaches hold current IHOLD (0x14), the movement is triggered by writing the value of VIRTUAL_STOP_L to X_TARGET.
	0: DISABLED 1: ENABLED	Disabled. Active REFL interrupts are cleared, while the position drive continues. Triggers a homing event on rising edge of REFL.
[13] EN_VIRTUAL_STOP_R	RW 0x0	Automatic motor stop during active right virtual stop condition.
	0: DISABLED 1: ENABLED	Disabled Enabled
[12] EN_VIRTUAL_STOP_L	RW 0x0	Enables automatic motor stop during active left virtual stop condition.
	0: DISABLED 1: ENABLED	Disabled Enabled

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[11] EN_SOFTSTOP	RW 0x0	The soft-stop mode always uses the deceleration ramp settings DMAX, V1, D1, V2, D2, VSTOP, and TZEROWAIT for stopping the motor. A stop occurs when the velocity sign matches the reference switch position (REFL, VIRTUAL_STOP_L for negative velocities, REFR, VIRTUAL_STOP_R for positive velocities), and the respective switch stop function is enabled. A hard stop also uses TZEROWAIT before the motor is released. Attention: Do not use soft-stop in combination with StallGuard. Use soft-stop for StealthChop operation at high velocity. In this case, hard stop must be avoided, as it can result in severe overcurrent.
	0: HARD_STOP 1: SOFT_STOP	Hard stop forces velocity to 0. Soft-stop using deceleration configuration.
[10] SG_STOP	RW 0x0	Enable stop by StallGuard. Disable to release motor after stop event. Program TCOOLTHRS for velocity threshold. The ramp generator is hard stopped when a stall is detected. (VACTUAL is set immediately to 0). Do not enable during motor spin-up, wait until the motor velocity exceeds a certain value, where StallGuard delivers a stable result. This velocity threshold should be programmed using TCOOLTHRS.
	0: DISABLED 1: ENABLED	Disabled Enabled
[8] LATCH_R_INACTIVE	RW 0x0	Activates latching of the position XACTUAL to XLATCH upon an inactive going edge on status_stop_r. The active level is defined by pol_stop_r.
	0: DISABLED 1: ENABLED	Disabled Enable latching on falling edge of status_stop_r.
[7] LATCH_R_ACTIVE	RW 0x0	Activates latching of the position XACTUAL to XLATCH upon an active going edge on status_stop_r. Activate latch_r_active to detect any spurious stop event by reading status_latch_r.
	0: DISABLED 1: ENABLED	Disabled Enable latching on rising edge of status_stop_r.
[6] LATCH_L_INACTIVE	RW 0x0	Activates latching of the position XACTUAL to XLATCH upon an inactive going edge on status_stop_l. The active level is defined by pol_stop_l.
	0: DISABLED 1: ENABLED	Disabled Enable latching on falling edge of status_stop_l.
[5] LATCH_L_ACTIVE	RW 0x0	Activates latching of the position XACTUAL to XLATCH upon an active going edge on the left reference switch input REFL. Activate latch_l_active to detect any spurious stop event by reading status_latch_l.
	0: DISABLED 1: ENABLED	Disabled Enable latching on rising edge of status_stop_l.
[4] SWAP_LR	RW 0x0	Swap the left and right reference switch input REFL and REFR for stop events.
	0: DEFAULT 1: SWAPPED	Default assignments. REFL and REFR switched.
[3] POL_STOP_R	RW 0x0	Sets the active polarity of the right reference switch input.
	0: NON_INV 1: INV	Non-inverted, high active: a high level on REFR stops the motor. Inverted, low active: a low level on REFR stops the motor.
[2] POL_STOP_L	RW 0x0	Sets the active polarity of the left reference switch input.
	0: NON_INV 1: INV	Non-inverted, high active: a high level on REFL stops the motor. inverted, low active: a low level on REFL stops the motor.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[1] STOP_R_ENABLE	RW 0x0	Enables automatic motor stop during active right reference switch input. The motor restarts in case the stop switch is released.
	0: DISABLED 1: ENABLED	Disabled Enabled
[0] STOP_L_ENABLE	RW 0x0	Enables automatic motor stop during active left reference switch input. The motor restarts in case the stop switch is released.
	0: DISABLED 1: ENABLED	Disabled Enabled

0x2E: RAMP_STAT

Default: 0x0000C000

Ramp status and switch event status.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[17] REFR_HOMING	RW, W1C 0x0	Gives information on if a homing procedure using REFR switch is triggered in the past. Write 1 to clear to check for future homing events on REFR.
	0: INACTIVE 1: ACTIVE	No homing event on REFR detected until now. A homing event on REFR is detected.
[16] REFL_HOMING	RW, W1C 0x0	1: Homing procedure using REFL switch is triggered.
	0: INACTIVE 1: ACTIVE	No homing event on REFL is detected until now. A homing event on REFL is detected.
[15] STATUS_VIRTUAL_STOP_R	R 0x1	Virtual reference switch right status (1 = active). To actually stop the ramp, this feature must be enabled in the register SW_MODE.
	0: INACTIVE 1: ACTIVE	Virtual stop r inactive. Virtual stop r detected.
[14] STATUS_VIRTUAL_STOP_L	R 0x1	Virtual reference switch left status (1 = active). To actually stop the ramp, this feature must be enabled in the register SW_MODE.
	0: INACTIVE 1: ACTIVE	Virtual stop l inactive. Virtual stop l active.
[13] STATUS_SG	R 0x0	1: Signals an active StallGuard input from the CoolStep driver or DcStep unit, if enabled. When polling this flag, stall events may be missed. Activate sg_stop to be sure not to miss the stall event.
	0: NO_STALL 1: STALL	No stall detected. Stall detected.
[12] SECOND_MOVE	RW, W1C 0x0	1: Signals the automatic ramp required moving back in the opposite direction, example, due to on-the-fly parameter change.
	0: INACTIVE 1: ACTIVE	No second move needed to reach target. Direction change is needed to reach a target position.
[11] T_ZEROWAIT_ACTIVE	R 0x0	1: Signals that TZEROWAIT is active after a motor stop. During this time, the ramp is not moving.
	0: INACTIVE 1: ACTIVE	Inactive TZEROWAIT break between motions is active.
[10] VZERO	R 0x0	1: Signals the actual velocity is 0.
	0: VEL_NOT_0 1: VEL_0	VACTUAL != 0 VACTUAL == 0
[9] POSITION_REACHED	R 0x0	1: Signals the target position is reached. This flag is set while XACTUAL and XTARGET match, and VACTUAL is 0.
	0: POS_NOT_REACHED 1: POS_REACHED	Target position not reached yet. Motion to target position is completed.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[8] VELOCITY_REACHED	R 0x0	1: Signals the target velocity is reached. This flag is set while abs (VACTUAL) and VMAX match.
	0: VEL_NOT_VMAX 1: VEL_VMAX	VMAX not reached. VMAX is reached.
[7] EVENT_POS_REACHED	RW, W1C 0x0	1: Signals the target position is reached (position_reached becoming active).
	0: INACTIVE 1: ACTIVE	Target position is not reached. Target position is reached since last clearing this flag.
[6] EVENT_STOP_SG	RW, W1C 0x0	1: Signals an active StallGuard stop event. Resetting the register clears the stall condition and the motor may restart motion, unless the motion controller is stopped.
	0: INACTIVE 1: ACTIVE	No active stall event. Motor is stopped due to detected stall.
[5] EVENT_STOP_R	R 0x0	1: Active stop right condition due to stop switch or virtual stop. The stop condition and interrupt condition can be removed by setting RAMP_MODE to hold mode or by commanding a move to the opposite direction. In soft_stop mode, the condition remains active until the motor has stopped motion into the direction of the stop switch. Disabling the stop switch or stop function also clears the flag, but the motor continues motion.
	0: INACTIVE 1: ACTIVE	No stop event. Active stop event right direction.
[4] EVENT_STOP_L	R 0x0	1: Active stop left condition due to stop switch or virtual stop. The stop condition and interrupt condition can be removed by setting RAMP_MODE to hold mode or by commanding a move to the opposite direction. In soft_stop mode, the condition remains active until the motor has stopped motion into the direction of the stop switch. Disabling the stop switch or stop function also clears the flag, but the motor continues motion. This bit is ORed to the interrupt output signal.
	0: INACTIVE 1: ACTIVE	No stop event. Active stop event left direction.
[3] STATUS_LATCH_R	RW, W1C 0x0	1: Latch right ready (enable position latching using SW_MODE settings latch_r_active or latch_r_inactive).
	0: INACTIVE 1: ACTIVE	No value latched. XACTUAL latched to XLATCH.
[2] STATUS_LATCH_L	RW, W1C 0x0	1: Latch left ready (enable position latching using SW_MODE settings latch_l_active or latch_l_inactive).
	0: INACTIVE 1: ACTIVE	No value latched. XACTUAL latched to XLATCH.
[1] STATUS_STOP_R	R 0x0	Reference switch right status (1 = active) after considering polarity and possible swap of REFL/REFR inputs (s. SW_MODE swap_lr). This bit just gives the status of the reference switch. To see a possible stop event, check event_stop_r.
	0: INACTIVE 1: ACTIVE	Inactive Reference switch active
[0] STATUS_STOP_L	R 0x0	Reference switch left status (1 = active) after considering polarity and possible swap of REFL/REFR inputs (s. SW_MODE SWAP_LR). This bit just gives the status of the reference switch. To see a possible stop event, check event_stop_l.
	0: INACTIVE 1: ACTIVE	Inactive Reference switch active

0x2F: XLATCH

Default: 0x00000000

Ramp generator latch position.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] XLATCH	R, signed 0x00000000	Ramp generator latch position, latches XACTUAL upon a programmable switch event (see SW_MODE).

0x30: X_BEMF

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] X_BEMF	RW, signed 0x00000000	Gives information on how many fullsteps the tricoder feature has detected since its enabling. Value depends on BEMF_INCREMENT.

0x31: BEMF_CONF

Default: 0x00080108

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[21] BEMF_OVERWRITE	RW 0x0	Update XACTUAL with X_BEMF (XACTUAL = XACTUAL + XBEMF) on refr_homing/refl_homing to compensate motor movement during QSC mode. After updating XACTUAL, X_BEMF is cleared to 0. The X_BEMF value used for updating is stored to X_BEMF_LATCH.
	0: DISABLED 1: ENABLED	Disabled Update XACTUAL with X_BEMF on homing event.
[20] BEMF_DIR	RW 0x0	This changes the counting direction of the Tricoder for X_BEMF register.
	0: NON_INV 1: INV	No inversion of counting direction. Invert counting direction.
[19:16] BEMF_INCREMENT	RW 0x8	Select the value X_BEMF. It is incremented/decremented by when a fullstep is detected by the Tricoder.
	0: FS_1 1: FS_2 2: FS_4 3: FS_8 4: FS_16 5: FS_32 6: FS_64 7: FS_128 8: FS_256	±1 per detected fullstep ±2 per detected fullstep ±4 per detected fullstep ±8 per detected fullstep ±16 per detected fullstep ±32 per detected fullstep ±64 per detected fullstep ±128 per detected fullstep ±256 per detected fullstep
[13:12] BEMF_FILTER_SEL	RW 0x0	Digital low-pass filter for filtering the back-emf signals of the Tricoder. Increase cutoff frequency if motion is too fast to detect.
	0: F_DIV24600 1: F_DIV12300 2: F_DIV6150 3: F_DIV2870	fctoff = fCLK/24600 fctoff = fCLK/12300 fctoff = fCLK/6150 fctoff = fCLK/2870
[11:4] BEMF_BLANK_TIME	RW, unsigned 0x10	Time until the Tricoder is enabled after reaching Tricoder enable conditions. (BEMF_BLANK_TIME × ((2^(14)) - 1) × t CLK)
[3] QSC_TRICODER_EN	RW 0x1	Tricoder enable setting during low-power quiescenct mode.
	0: QSC_OFF 1: QSC_ON	Tricoder is inactive during low-power quiescent mode to reduce power consumption. Tricoder is active during low-power quiescent mode (default).

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[2:0] BEMF_HYST	RW 0x0	Set hysteresis for Tricoder comparator. Increase if there is noise on the signal when the motor is not moved.
	0: HYST_10MV 10mV 1: HYST_25MV 25mV 2: HYST_50MV 50mV 3: HYST_75MV 75mV 4: HYST_100MV 100mV 5: HYST_150MV 150mV 6: HYST_200MV 200mV 7: HYST_250MV 250mV	

0x32: BEMF_DEVIATION

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[24] DEVIATION_WARN	RW, W1C 0x0	Tricoder deviation warning flag. Checks whether X_BEMF has left the range given by DEVIATION.
	0: INACTIVE No warning (disabled or within configured deviation range). 1: ACTIVE Deviation warning. X_BEMF is not within the range configured in DEVIATION. Cannot be cleared while a warning still persists. Set DEVIATIONto zero to disable.	
[19:0] DEVIATION	RW, unsigned 0x00000	Maximum value abs(X_BEMF) may deviate from 0 until deviation_warn is set. Result in flag deviation_warn 0 = function is off.

0x33: VIRTUAL_STOP_L

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] VIRTUAL_STOP_L	RW, signed 0x00000000	Virtual stop switch based on internal position XACTUAL. A stop is raised, based on the signed comparison: XACTUAL ≤ VIRTUAL_STOP_L -2^(31) to (+2^(31) - 1) May also be used as XTARGET buffer for homing feature and autotarget.

0x34: VIRTUAL_STOP_R

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] VIRTUAL_STOP_R	RW, signed 0x00000000	Virtual stop switch based on internal position XACTUAL. A stop is raised, based on the signed comparison: XACTUAL ≥ VIRTUAL_STOP_R -2^(31) to (+2^(31) - 1) May also be used as XTARGET buffer for homing feature and autotarget.

0x35: X_BEMF_LATCH

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] X_BEMF_LATCH	R, signed 0x00000000	If BEMF_OVERWRITE is enabled, X_BEMF is cleared on a homing event and its value latched into this register.

0x36: MSCNT

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[9:0] MSCNT	R, unsigned 0x000	Microstep counter. Indicates actual position in the microstep table for CUR_B. CUR_A uses an offset of 256 (two-phase motor). Move to a position where MSCNT is zero before reinitializing MSLUTSTART or MSLUT and MSLUTSEL .

0x37: MSCURACT

Default: 0x000000F7

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[24:16] CUR_B	R, signed 0x000	Actual microstep current for motor phase B as read from MSLUT (not scaled by current).
[8:0] CUR_A	R, signed 0x0F7	Actual microstep current for motor phase A as read from MSLUT (not scaled by current).

0x38: CHOPCONF

Default: 0x10410150

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[29] DEDGE	RW 0x0	Enable double edge step pulses.
	0: DISABLED 1: ENABLED	Only rising edge on the step input is interpreted as step. Enable step impulse at each step edge to reduce step frequency requirement.
[28] INTPOL	RW 0x1	The actual microstep resolution (MRES) can be extrapolated to 256 microsteps for smoothest motor operation (useful when not using 256 microstep resolution) by setting this bit.
	0: DISABLED 1: ENABLED	Interpolation disabled. Interpolation enabled with 256 microsteps.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[27:24] MRES	RW 0x0	Microstep resolution selection. The resolution gives the number of microstep entries per sine quarter wave. The driver automatically uses microstep positions, which result in a symmetrical wave, when choosing a lower microstep resolution. step width = 2^(MRES) [microsteps]. Native 256 microstep setting. Normally use this setting with the internal motion controller.
	0: RES_256 1: RES_128 2: RES_64 3: RES_32 4: RES_16 5: RES_8 6: RES_4 7: RES_HS 8: RES_FS	256 steps per fullstep 128 steps per fullstep 64 steps per fullstep 32 steps per fullstep 16 steps per fullstep 8 steps per fullstep 4 steps per fullstep 2 steps per fullstep Fullstep
[23:20] TPFD	RW 0x4	Passive fast decay time. TPFD allows dampening of motor mid-range resonances. Passive fast decay time setting controls duration of the fast-decay phase inserted after bridge polarity change NCLK = 128 × TPFD %0000: Disable %0001 to %1111: 1 to 15
	0: TPFD_OFF 1: TPFD_128 2: TPFD_256 3: TPFD_384 4: TPFD_512 5: TPFD_640 6: TPFD_768 7: TPFD_896 8: TPFD_1024 9: TPFD_1152 10: TPFD_1280 11: TPFD_1408 12: TPFD_1536 13: TPFD_1664 14: TPFD_1792 15: TPFD_1920	Disabled Passive fast decay time: 128 × t_clk Passive fast decay time: 256 × t_clk Passive fast decay time: 384 × t_clk Passive fast decay time: 512 × t_clk Passive fast decay time: 640 × t_clk Passive fast decay time: 768 × t_clk Passive fast decay time: 896 × t_clk Passive fast decay time: 1024 × t_clk Passive fast decay time: 1152 × t_clk Passive fast decay time: 1280 × t_clk Passive fast decay time: 1408 × t_clk Passive fast decay time: 1536 × t_clk Passive fast decay time: 1664 × t_clk Passive fast decay time: 1792 × t_clk Passive fast decay time: 1920 × t_clk
[19] VHIGHCHM	RW 0x0	High velocity chopper mode. This bit enables switching to CHM =1 and FD =0, when VHIGH is exceeded. This way, a higher velocity can be achieved. Can be combined with VHIGHFS = 1. If set, the TOFF setting automatically is doubled during high velocity operation to avoid doubling of the chopper frequency.
	0: CTOFF_THIGH_DIS 1: CTOFF_THIGH_EN	Disabled Switch to constant TOFF chopper when reaching THIGH.
[18] VHIGHFS	RW 0x0	High velocity fullstep selection. This bit enables switching to fullstep, when VHIGH is exceeded. Switching takes place only at 45° position. The fullstep target current uses the current value from the microstep table at the 45° position.
	0: FS_THIGH_DIS 1: FS_THIGH_EN	Disabled Switches from microstep to fullstep on reaching THIGH.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[16:15] TBL	RW 0x2	TBL/blank time select. %00 to %11: Set comparator blank time to 20, 28, 36, or 54 clocks. %01 or %10 is recommended for most applications. Hint: Reduce only if the blanking time is relevant for ending the on-phase or fast-decay phase in SpreadCycle instead of the comparator event.
	0: TBL_20 1: TBL_28 2: TBL_36 3: TBL_54	20 clock cycles 28 clock cycles 36 clock cycles 54 clock cycles
[14] CHM	RW 0x0	Chopper mode: Switch between SpreadCycle and constant TOFF chopper.
	0: SPREADCYCLE 1: CLASSIC_CHOP	Standard mode (SpreadCycle). Constant off-time with fast-decay time. Fast-decay time is also terminated when the negative nominal current is reached. Fast decay is after on-time.
[12] DISFDCC	RW 0x0	Fast decay mode. with CHM = 1: DISFDCC = 1 disables current comparator usage for termination of the fast decay cycle.
	0: DISABLED 1: ENABLED	Current comparator disables fast decay. Fixed fast decay time.
[11] FD3	RW 0x0	TFD[3] with CHM =1: MSB of fast decay time setting TFD.
	0: TFD3_0 1: TFD3_1	MSB of TFD setting: 0 MSB of TFD setting: 1
[10:7] HEND_OFFSET	RW, unsigned 0x2	With CHM = 0: HEND hysteresis low value. %0000 to %1111: Hysteresis is -3, -2, -1, 0, 1 to 12 (1/512 of this setting adds to current setting). This is the hysteresis value used for the hysteresis chopper. With CHM = 1: OFFSET sine-wave offset. %0000 to %1111: Offset is -3, -2, -1, 0, 1 to 12 This is the sine-wave offset and 1/512 of the value is added to the absolute value of each sine-wave entry.
[6:4] HSTRT_TFD210	RW, unsigned 0x5	With CHM = 0: HSTRT hysteresis start value added to HEND. %000 to %111: Add 1 to 8 to hysteresis. Low value HEND (1/512 of this setting adds to current setting). Attention: Effective HEND + HSTRT ≤ 16. Hysteresis decrement is done each 16 clocks. With CM = 1: TFD [2..0] fast decay time setting. Fast decay time setting (MSB: FD3): %0000 to %1111: Fast decay time setting TFD with NCLK = 32 × TFD (%0000: slow decay only)

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[3:0] TOFF	RW 0x0	TOFF off-time and driver enable. Off-time setting controls duration of slow-decay phase. NCLK = 24 + 32 × TOFF %0000: Driver disable, all bridges off. %0001: 1 – use only with TBL ≥ 2. %0010 to %1111: 2 to 15
	0: DRIVER_OFF 1: TOFF_56 2: TOFF_88 3: TOFF_120 4: TOFF_152 5: TOFF_184 6: TOFF_216 7: TOFF_248 8: TOFF_280 9: TOFF_312 10: TOFF_344 11: TOFF_376 12: TOFF_408 13: TOFF_440 14: TOFF_472 15: TOFF_504	Driver disabled and all bridges off. Slow decay phase duration: 56 × t_clk. Use with TBL >= 2. Slow decay phase duration: 88 × tclk. Slow decay phase duration: 120 × tclk. Slow decay phase duration: 152 × tclk. Slow decay phase duration: 184 × tclk. Slow decay phase duration: 216 × tclk. Slow decay phase duration: 248 × tclk. Slow decay phase duration: 280 × tclk. Slow decay phase duration: 312 × tclk. Slow decay phase duration: 344 × tclk. Slow decay phase duration: 376 × tclk. Slow decay phase duration: 408 × tclk. Slow decay phase duration: 440 × tclk. Slow decay phase duration: 472 × tclk. Slow decay phase duration: 504 × tclk.

0x39: COOLCONF

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[24] SFILT	RW 0x0	StallGuard2 filter enable.
	0: FILT_DISABLED 1: FILT_ENABLED	Standard mode, high time resolution for StallGuard2. Filtered mode, StallGuard2 signal updated for each four fullsteps only to compensate for motor pole tolerances.
[22:16] SGT	RW, signed 0x00	StallGuard2 This signed value controls StallGuard level for stall output and sets the optimum measurement range for readout. A lower value gives a higher sensitivity. Zero is the starting value working with most motors. -64 to +63: A higher value makes StallGuard2 less sensitive and requires more torque to indicate a stall.
[15] SEIMIN	RW 0x0	Minimum current for smart current control.
	0: IRUN_DIV2 1: IRUN_DIV4	1/2 of current setting (IRUN) (when used with StealthChop requires IRUN ≥ 16). 1/4 of current setting (IRUN) (when used with StealthChop requires IRUN ≥ 28).
[14:13] SEDN	RW 0x0	Current down step speed.
	0: STEP_DOWN_EACH_32 1: STEP_DOWN_EACH_8 2: STEP_DOWN_EACH_2 3: STEP_DOWN_EACH_1	For each 32 StallGuard2/StallGuard4, values decrease by one. For each 8 StallGuard2/StallGuard4, values decrease by one. For each 2 StallGuard2/StallGuard4, values decrease by one. For each StallGuard2/StallGuard4, values decrease by one.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[11:8] SEMAX	RW, unsigned 0x0	StallGuard2/StallGuard4 hysteresis value for smart current control. If the StallGuard result is equal to or above (SEMIN + SEMAX + 1) × 32, the motor current is decreased to save energy. %0000 to %1111: 0 to 15
[6:5] SEUP	RW 0x0	Current up step width. Current increment steps per measured StallGuard2/StallGuard4 value.
	0: STEP_UP_1 1: STEP_UP_2 2: STEP_UP_4 3: STEP_UP_8	1 increment per StallGuard2/StallGuard4 value. 2 increments per StallGuard2/StallGuard4 value. 4 increments per StallGuard2/StallGuard4 value. 8 increments per StallGuard2/StallGuard4 value.
[3:0] SEMIN	RW, unsigned 0x0	Minimum StallGuard2/StallGuard4 value for smart current control and enable. If the StallGuard result falls below SEMIN × 32, the motor current is increased to reduce motor load angle. %0000: Smart current control CoolStep off %0001 to %1111: 1 to 15

0x3A: DCCTRL

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[23:16] DC_SG	RW, unsigned 0x00	Maximum PWM on-time for step loss detection using DcStep StallGuard2 in DcStep mode (DC_SG × 16/fCLK). Set slightly higher than DC_TIME/16. 0 = disable.
[9:0] DC_TIME	RW, unsigned 0x000	Upper PWM on-time limit for commutation (DC_TIME × 1/f CLK). Set slightly above effective blank time TBL .

0x3B: DRV_STATUS

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31] STST	R 0x0	Standstill indicator. This flag indicates motor standstill in each operation mode. This occurs depending on the standstill timer configuration. between 2^(13) to 2^(20) clock cycles after the last step pulse.
	0: INACTIVE 1: ACTIVE	Motor moving. Motor in standstill.
[30] OLB	R 0x0	Open load indicator phase B. This is just an informative flag. The driver takes no action upon it. False detection may occur in fast motion and standstill. Check during slow motion only.
	0: OPERATIONAL 1: OPEN_LOAD	Normal operation. Open load detected on phase B.
[29] OLA	R 0x0	Open load indicator phase A. This is just an informative flag. The driver takes no action upon it. False detection may occur in fast motion and standstill. Check during slow motion only.
	0: OPERATIONAL 1: OPEN_LOAD	Normal operation. Open load detected on phase A.

BITS AND NAME	TYPE AND RESET		DESCRIPTION
[28] S2GB	R 0x0		Short-to-ground indicator phase B. The driver is disabled. The flags stay active, until the driver is disabled by software (TOFF = 0, setting DRV_EN = 0 or setting drv_en_sw = 0).
	0: 1:	OPERATIONAL ERROR	Normal operation. Short-to-GND detected on phase B.
[27] S2GA	R 0x0		Short-to-ground indicator phase A. The driver is disabled. The flags stay active, until the driver is disabled by software (TOFF = 0, setting DRV_EN = 0 or setting drv_en_sw = 0).
	0: 1:	OPERATIONAL ERROR	Normal operation. Short-to-GND detected on phase A.
[26] OTPW	R 0x0		Overtemperature prewarning flag. Overtemperature prewarning threshold is exceeded. The overtemperature prewarning flag is common for both bridges.
	0: 1:	INACTIVE ACTIVE	Normal operation. Configured threshold exceeded.
[25] OT	R 0x0		Overtemperature flag.
	0: 1:	OPERATIONAL ERROR	Normal operation. Overtemperature limit is reached. Drivers are disabled due to cooling down of the IC. The overtemperature flag is common for both bridges.
[24] STALLGUARD	R 0x0		StallGuard2/StallGuard4 status.
	0: 1:	INACTIVE ACTIVE	Normal operation. Motor stall detected or DcStep stall in DcStep mode.
[23] STALLGUARD_PREWARN	R 0x0		1: SG_PREWARN_RESULT_VALID = 1 && (SG_PREWARN_THRSH >= SG_PREWARN_RESULT) This bit can also be propagated to a DOx pin.
	0: 1:	INACTIVE ACTIVE	StallGuard prewarning not triggered. StallGuard prewarning detected.
[22] RES_DET	R 0x0		Depending on GCONF[11], this shows the status of the used resistor.
	0: 1:	NO_RES_DET RES_DET	No resistor detected. Resistor detected.
[20:16] CS_ACTUAL	R, unsigned 0x00		Actual motor current/smart energy current. Actual current control scaling, for monitoring smart energy current scaling controlled using the settings in register COOLCONF, or for monitoring the function of the automatic current scaling. (Current is scaled by (CS_ACTUAL + 1)/32).
[15] FSACTIVE	R 0x0		Fullstep active indicator.
	0: 1:	USTEP FSTEP	Microstepping active. Indicates the driver has switched to fullstep, as defined by chopper mode settings and velocity thresholds,
[14] STEALTH	R 0x0		StealthChop indicator.
	0: 1:	SPREADCYCLE_CTOFF STEALTHCHOP	StealthChop not active. Driver operates in StealthChop mode.
[13] S2VSB	R 0x0		Short-to-supply indicator phase B. The driver is disabled. The flags stay active, until the driver is disabled by software (TOFF = 0, setting pin DRV_EN = 0 or setting drv_en_sw = 0).
	0: 1:	OPERATIONAL ERROR	No error. Short-to-supply detected on phase B.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[12] S2VSA	R 0x0	Short-to-supply indicator phase A. The driver is disabled. The flags stay active, until the driver is disabled by software (TOFF = 0, setting pin DRV_EN = 0 or setting drv_en_sw = 0).
	0: OPERATIONAL 1: ERROR	No error. Short-to-supply detected on phase A.
[9:0] SG_RESULT	R, unsigned 0x000	StallGuard2 result, respectively, PWM on-time for coil A in standstill for motor temperature detection. Mechanical load measurement: The StallGuard result helps to measure mechanical motor load. A higher value means lower mechanical load. A value of 0 signals highest load. With optimum SGT setting, this is an indicator for a motor stall. The stall detection compares SG_RESULT to 0 to detect a stall. SG_RESULT is used as a base for CoolStep operation, by comparing it to a programmable upper and lower limit. StallGuard2 works best with microstep operation or DcStep. Temperature measurement: In standstill, no StallGuard2 result can be obtained. SG_RESULT shows the chopper on-time for motor coil A instead. Move the motor to a determined microstep position at a certain current setting to get a rough estimation of motor temperature by reading the chopper on-time. As the motor heats up, its coil resistance rises and the chopper on-time increases. When in StealthChop, this register displays SG4_RESULT.

0x3C: PWMCONF

Default: 0xC40C001D

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:28] PWM_LIM	RW, unsigned 0xC	PWM automatic scale amplitude limit when switching on. Limit for PWM_SCALE_AUTO when switching back from SpreadCycle to StealthChop. This value defines the upper limit for bits 7 to 4 of the automatic current control when switching back. It can be set to reduce the current jerk during mode change back to StealthChop. It does not limit PWM_GRAD or PWM_GRAD_AUTO offset. (Default = 12)
[27:24] PWM_REG	RW 0x4	Regulation loop gradient. User-defined maximum PWM amplitude change per half-wave when using pwm_autoscale = 1 .
	0: RESERVED 1: INC_05 2: INC_10 3: INC_15 4: INC_20 5: INC_25 6: INC_30 7: INC_35 8: INC_40 9: INC_45 10: INC_50 11: INC_55 12: INC_60 13: INC_65 14: INC_70 15: INC_75	Do not use 0.5 increments (slowest regulation) 1 increment 1.5 increments 2 increments 2.5 increments 3 increments 3.5 increments 4 increments 4.5 increments 5 increments 5.5 increments 6 increments 6.5 increments 7 increments 7.5 increments (fastest regulation)

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[23] PWM_DIS_REG_STST	RW 0x0	Standstill current regulation control.
	0: ACTIVE	Current regulation active.
	1: INACTIVE	Disable current regulation when motor is in standstill and current is reduced (less than IRUN). This option eliminates any regulation noise during standstill.
[22] PWM_MEAS_SD_ENABLE	RW 0x0	Slow-decay phase low-side current measurement control.
	0: DISABLED	Slow-decay low-side measurement disabled (default).
	1: ENABLED	Uses slow-decay phases on low-side to measure the motor current to reduce the lower current limit.
[21:20] PWM_FREEWHEEL	RW 0x0	Standstill mode configuration. Standstill option when motor current setting is zero (I_HOLD = 0).
	0: NORMAL	Normal operation.
	1: FREEWHEEL	Freewheeling.
	2: LS_SHORT	Coil shorted using LS drivers.
	3: HS_SHORT	Coil shorted using HS drivers.
[19] PWM_AUTOGRAD	RW 0x1	PWM automatic gradient adaptation.
	0: FIXED	Fixed value for PWM_GRAD (PWM_GRAD_AUTO = PWM_GRAD).
	1: AUTO	Automatic tuning (only with pwm_autoscale= 1) (reset default). PWM_GRAD_AUTO is initialized with PWM_GRAD while pwm_autograd= 0 and is optimized automatically during motion. Preconditions PWM_OFS_AUTO is automatically initialized. This requires standstill at IRUN for >130ms to a) detect standstill b) wait > 128 chopper cycles at IRUN, and c) regulate PWM_OFS_AUTO so that -1 PWM_SCALE_AUTO. Motor running and 1.5 × PWM_OFS_AUTO PWM_SCALE_SUM PWM_OFS_AUTO and PWM_SCALE_SUM. Time required for tuning PWM_GRAD_AUTO. About 8 fullsteps per change of ±1. Also enables use of reduced chopper frequency for tuning PWM_OFS_AUTO.
[18] PWM_AUTOSCALE	RW 0x1	PWM automatic amplitude scaling.
	0: USER	User-defined feed-forward PWM amplitude. The current settings IRUN and IHOLD have no influence. The resulting PWM amplitude (limited to 0 to 255) is: PWM_OFS × ((CS_ACTUAL + 1)/32) + PWM_GRAD × 256/TSTEP.
	1: AUTO	Enable automatic current control (reset default).
[17:16] PWM_FREQ	RW 0x0	PWM frequency selection depending on used clock frequency.
	0: FCLK_2DIV1024	fPWM = 2/1024 fCLK (reset default)
	1: FCLK_2DIV683	fPWM = 2/683 fCLK
	2: FCLK_2DIV512	fPWM = 2/512 fCLK
	3: FCLK_2DIV410	fPWM = 2/410 fCLK
[15:8] PWM_GRAD	RW, unsigned 0x00	Velocity-dependent gradient for PWM amplitude: PWM_GRAD × 256/TSTEP This value is added to PWM_AMPL to compensate for the velocity-dependent motor back-EMF. Use PWM_GRAD as initial value for automatic scaling to speed up the automatic tuning process. To do this, set PWM_GRAD to the determined, application-specific value, with pwm_autoscale = 0. Only afterwards, set pwm_autoscale = 1. Enable StealthChop when finished. After initial tuning, the required initial value can be read out from PWM_GRAD_AUTO.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7:0] PWM_OFS	RW, unsigned 0x1D	User-defined PWM amplitude offset (0 to 255) related to full motor current (CS_ACTUAL = 31) in standstill. Use PWM_OFS as initial value for automatic scaling to speed up the automatic tuning process. To do this, set PWM_OFS to the determined, application-specific value, with pwm_autoscale = 0. Only afterwards, set pwm_autoscale = 1. Enable StealthChop when finished. PWM_OFS = 0 disables scaling down motor current below a motor-specific lower measurement threshold. This setting should only be used under certain conditions, that is, when the power supply voltage can vary up and down by a factor of two or more. It prevents the motor going out of regulation, but it also prevents power down below the regulation limit. PWM_OFS > 0 allows automatic scaling to low PWM duty cycles even below the lower regulation threshold. This allows low (standstill) current settings based on the actual (hold) current scale (register IHOLD_IRUN).

0x3D: PWM_SCALE

Default: 0x00000000

Results of StealthChop amplitude regulator. These values can be used to monitor automatic PWM amplitude scaling (255 = maximum voltage).

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[24:16] PWM_SCALE_AUTO	R, unsigned 0x000	Automatically determined current scaling value.
[9:0] PWM_SCALE_SUM	R, unsigned 0x000	Bits: 9:0 [0 to 1023] PWM_SCALE_SUM: Actual PWM duty cycle. This value is used for scaling the values CUR_A and CUR_B read from the sine-wave table. 1023: Maximum duty cycle. This value has a higher precision for the duty cycle read out. Bits 9:2 correspond to the 8-bit values in other PWM duty cycle-related registers.

0x3E: PWM_AUTO

Default: 0x00000000

These automatically generated values can be read out to determine a default/power-up setting for PWM_GRAD and PWM_OFS.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[23:16] PWM_GRAD_AUTO	R, unsigned 0x00	Automatically determined gradient value.
[7:0] PWM_OFS_AUTO	R, unsigned 0x00	Automatically determined offset value.

0x3F: SG4_THRS

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[9] SG4_FILT_EN	RW 0x0	SG4 filter control. 1: SG4_RESULT is averaged using sg4_ind_0, _1, _2, _3.
	0: FILT_DISABLE 1: FILT_ENABLE	Disable SG4 filter. Enable SG4 filter.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[8:0] SG4_THRS	RW, unsigned 0x000	Detection threshold for stall. The StallGuard4 value SG4_RESULT is compared to this threshold. A stall is signaled with: SG_RESULT ≤ SG4_THRS

0x40: SG4_RESULT

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[9:0] SG4_RESULT	R, unsigned 0x000	StallGuard result for StallGuard4 only. SG4_RESULT is updated with each fullstep, independent of TCOOLTHRS and SG4THRS. A higher value signals a lower motor load and more torque headroom. Intended for StealthChop mode only. Bits 9 and 0 always show 0. Scaling to 10-bit is for compatibility to StallGuard2.

0x41: SG4_IND

Default: 0x00000000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:24] SG4_IND_3	R, unsigned 0x00	When SG4_filt_en = 1: Displays SG4 measurement 3 used as filter input.
[23:16] SG4_IND_2	R, unsigned 0x00	When SG4_filt_en = 1: Displays SG4 measurement 2 used as filter input.
[15:8] SG4_IND_1	R, unsigned 0x00	When SG4_filt_en = 1: Displays SG4 measurement 1 used as filter input.
[7:0] SG4_IND_0	R, unsigned 0x00	Displays SG4 measurement. When SG4_filt_en = 1: Displays SG4 measurement 0 used as filter input.

0x42: SG_PREWARN_CONF

Default: 0x00001000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[30:28] SG_PREWARN_FILTER	RW 0x0	Additional low-pass filter using SG_RESULT as input. Output is SG_PREWARN_RESULT. (0: Bypass) Filter length: 2^SG_PREWARN_FILTER (maximum 64).
	0: DISABLED — Uses SG_RESULT. 1: FILT_2 — Filter over 2 SG_RESULT values. 2: FILT_4 — Filter over 4 SG_RESULT values. 3: FILT_8 — Filter over 8 SG_RESULT values. 4: FILT_16 — Filter over 16 SG_RESULT values. 5: FILT_32 — Filter over 32 SG_RESULT values. 6: FILT_64 — Filter over 64 SG_RESULT values. 7: NOT_USED — Not used.	
[24:16] SG_PREWARN_THRSH	RW, unsigned 0x000	StallGuard prewarning threshold compared to SG_PREWARN_RESULT. if SG_PREWARN_RESULT_VALID = 1 && (SG_PREWARN_THRSH >= SG_PREWARN_RESULT), stallguard_prewarn in DRV_STATUS is set.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[12] SG_PREWARN_RESULT_VALID	R 0x1	1: Signals that SG_PREWARN_RESULT can be evaluated and the automatic comparison with the threshold SG_PREWARN_THRSH is enabled. Is enabled after 2^(SG_PREWARN_FILTER + 4 × stallguard_filter_enabled)(fullsteps). Is set to 0 when TSTEP > TCOOLTHRS. Enabled when TSTEP < TCOOLTHRS - hysteresis setting (see small_hysteresis setting in register 0x0). Is set to 0 when switching between sg4 and sg2 or the filter setting has changed. With default settings after power-up, this value is 1
	0: INVALID 1: VALID	Result is not yet valid as filter is not ready. Result is valid.
[9:0] SG_PREWARN_RESULT	R, unsigned 0x000	Current value of the filtered StallGuard value. Is 0 and disabled when TSTEP > TCOOLTHRS. Enabled when TSTEP < TCOOLTHRS - hysteresis setting (see small_hysteresis setting in register 0x0). The value can only be trusted when SG_PREWARN_RESULT_VALID is 1. Is cleared when switching chopper modes.

0x43: ADC_SUPPLY_TEMP

Default: 0x00001000

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[24:16] SUPPLY	R, unsigned 0x000	VSupply [V] = 0.0364 V × SUPPLY Internal temperature measurement: Temperature = 2.03 × ADC_TEMPERATURE - 259 °C
[12] ADC_EN	RW 0x1	1: Enable measurements of ADC (default). 0: Disable ADC when measurements are not needed to reduce power consumption. Hint: The ADC is automatically disabled in qsc mode.
	0: ADC_DISABLED 1: ADC_ENABLED	ADC measurement disabled. ADC measurement enabled.
[8:0] TEMPERATURE	R, unsigned 0x000	T [°C] = 1.017°C × TEMPERATURE -259,2°C Internal temperature measurement: Temperature = 2.03 × ADC_TEMPERATURE - 259 °C

0x44: VMAX_LIMIT

Default: 0x007FFFFF

Limits register field VMAX. VMAX can never be higher than this value. Reset value is set to not affect behaviour if limit is not used.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[22:0] VMAX_LIMIT	RW, unsigned 0x7FFFFF	Limits register field VMAX. VMAX can never be higher than this value.

0x45: LIMIT_VALUES

Default: 0x1F1F0F00

Register holds multiple limit values for other registers. Reset value is set to not affect behaviour if limit is not used.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[28:24] IHOLD_LIMIT	RW, unsigned 0x1F	Limits IHOLD value. IHOLD can never be higher than this limit value.
[20:16] IRUN_LIMIT	RW, unsigned 0x1F	Limits IRUN value. IRUN can never be higher than this limit value.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[11:10] GAIN_SCALER_LIMIT	RW, unsigned 0x3	Limits FSR_IREF value in register DRV_CONF. FSR_IREF can never be higher than this limit value.
[9:8] RREF_LIMIT	RW, unsigned 0x3	Limits FSR value in register DRV_CONF. FSR can never be higher than this limit value.
[7:0] GLOBAL_SCALER_LIMIT	RW, unsigned 0x00	Limits register GLOBALSCALER_A and GLOBALSCALER_B. Limit is disabled if GLOBAL_SCALER_LIMIT = 0. Otherwise, if GLOBALSCALER_A/B is greater than the limit or 0, the limit value is used.

0x46: USER_DATA_0

Default: 0x00000000

General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] USER_DATA_0	RW, unsigned 0x00000000	General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

0x47: USER_DATA_1

Default: 0x00000000

General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] USER_DATA_1	RW, unsigned 0x00000000	General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

0x48: USER_DATA_2

Default: 0x00000000

General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] USER_DATA_2	RW, unsigned 0x00000000	General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

0x49: USER_DATA_3

Default: 0x00000000

General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] USER_DATA_3	RW, unsigned 0x00000000	General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

0x4A: USER_DATA_4

Default: 0x00000000

General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] USER_DATA_4	RW, unsigned 0x00000000	General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

0x4B: USER_DATA_5

Default: 0x00000000

General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] USER_DATA_5	RW, unsigned 0x00000000	General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

0x4C: USER_DATA_6

Default: 0x00000000

General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] USER_DATA_6	RW, unsigned 0x00000000	General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

0x4D: USER_DATA_7

Default: 0x00000000

General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[31:0] USER_DATA_7	RW, unsigned 0x00000000	General purpose register for user data. Register is not used internally. Register value can be set by OTP to store configuration data.

0x7D: OTP_MODE

Default: 0x0000ED58

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[15:0] OTP_MODE_PASSWORD	RW, unsigned 0xED58	Enable OTP mode by setting this register to 0x12A7

MTP_CTRL Register Overview

The register map switches from the functional registers to this register map when the device is set to the OTP mode. Functional registers are not available in the OTP mode.

ADDRESS AND NAME	FIELDS						MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)	
0x10 MTP_CONTROL	MTP_ RESTORE			MTP_ PROT_EN			STOP_ PROG	SRT_ PROG
0x11 MTP_STATUS	DONE	ECC_ ERR_2BIT	ECC_ ERR_1BIT	OV_ DURING_ BURN_ PULSE	VPP_INIT_ FAIL	MTP_ FULL	VERI_FAIL	
0x12 MTP_PROT_ADDR	MTP_PROT_ADDR							
0x13 MTP_PROT_WDATA	MTP_PROT_WDATA							
0x14 MTP_PROT_RDATA	MTP_PROT_RDATA							
0x21 CP_CONTROL_2				SLP_EN_ STG_SEL	SLP_EN_ WINCMP	SLP_EN_ VRPCP	SLP_MAX_STAGES	
0x22 CP_STATUS				PROG_ ERR	CP_ERR	CP_CMPOUT		
0x30 MTP_DATA0	MTP_RECORD[7:0]							
0x31 MTP_DATA1	MTP_RECORD[15:8]							
0x32 MTP_DATA2	MTP_RECORD[23:16]							
0x33 MTP_DATA3	MTP_RECORD[31:24]							
0x34 MTP_ADDR		MTP_RECORD[38:32]						

0x10: MTP_CONTROL

Default: 0x00

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7] MTP_RESTORE	RW 0x0	Restore the MTP after a burn as taken place to load the latest state into the MTP flops. This bit is auto-clearing.
[4] MTP_PROT_EN	RW 0x0	Set the controller into prototype mode so that MTP_PROT_WDATA/RDATA/ADDR can be used to interogate the MTP registers without having to execute a burn.
[1] STOP_PROG	RW 0x0	At any time, writing this bit to 1 can STOP/abandon a field program burn.
[0] SRT_PROG	RW 0x0	Start executing the OTP write procedure.

0x11: MTP_STATUS

Default: 0x00

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7] DONE	R 0x0	Programming complete.
[6] ECC_ERR_2BIT	R 0x0	One of the previously burned records contains an uncorrectible 2-bit error. The record is not loaded, and the registers it is intended to load are not written. If an erroneous record is overwritten by a correct record, this status is not cleared.
[5] ECC_ERR_1BIT	R 0x0	One of the actively loaded records contains a single-bit error that is corrected. If an erroneous record is overwritten by a correct record, this status is cleared for each MTP register.
[4] OV_DURING_BURN_PULSE	R 0x0	When the memory is actively executing a burn pulse, the high-voltage V_{PP} voltage has gone above the abs-max specified in the data sheet. This is considered a critical failure, and device operation is not guaranteed thereafter.
[3] VPP_INIT_FAIL	R 0x0	The programming voltage does not reach the desired value before burning. Therefore, the burn i abandoned and not attempted.
[2] MTP_FULL	R 0x0	The OTP is full, and no more records can be burned.
[1] VERI_FAIL	R 0x0	OTP record is attempted to be burned thrice, and verification of each attempt produces an incorrect result.

0x12: MTP_PROT_ADDR

Default: 0x00

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7:0] MTP_PROT_ADDR	RW, unsigned 0x00	The address of the MTP register to be read/written to for prototyping purposes.

0x13: MTP_PROT_WDATA

Default: 0x00

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7:0] MTP_PROT_WDATA	RW, unsigned 0x00	Writing to this register triggers a write to the register with the MTP_PROT_ADDR address. Example, setting MTP_PROT_ADDR to 0xC selects MTP_CONFIG_0 as target. Then, writing, example, 0x08 to MTP_PROT_WDATA sets bit DISABLE_IREF_FAULT.

0x14: MTP_PROT_RDATA

Default: 0x00

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7:0] MTP_PROT_RDATA	R, unsigned 0x00	Writing to the MTP_PROT_ADDR register triggers a read to be sent to the MTP register at the address MTP_PROT_ADDR. This register can be used to read the contents of the register at that address.

0x21: CP_CONTROL_2

Default: 0x00

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[4] SLP_EN_STG_SEL	RW 0x0	Set to 1.
[3] SLP_EN_WINCMP	RW 0x0	Enable the window comparator IP. This is required before field programming can start.
[2] SLP_EN_VRPCP	RW 0x0	Enables the controller that monitors the quality of the external voltage using the window comparator.
[1:0] SLP_MAX_STAGES	RW, unsigned 0x0	The maximum number of charge pump stages the charge pump should use (when SLVHS_OVERRIDE = 1). When SLVHS_OVERRIDE = 0, the maximum stages are 3 (all enabled).

0x22: CP_STATUS

Default: 0x00

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[4] PROG_ERR	RW 0x0	This bit asserts when the burn voltage is not within range when the power switch is enabled. This bit is updated in real-time. If the burn voltage is marginal, this bit likely dithers. If the burn voltage is out of range, this bit stays high. Once the power switch is disabled, this bit retains its state.
[3] CP_ERR	RW 0x0	After the external voltage is enabled, a failsafe timer waits for the voltage to be in range of the window comparators before starting the burn. If the voltage never comes in range, the failsafe timer expires and this bit is asserted.
[2:0] CP_CMPOUT	R, unsigned 0x0	This bit stores a live readout of the window comparator values at all times. The burning voltage is in range when this field has the value 0b011. CP_CMPOUT[0] = lower threshold. - If 0, the voltage is too low. - If 1, the voltage is above the lower threshold. CP_CMPOUT[1] = middle threshold. same as CP_CMPOUT[0]. CP_CMPOUT[2] = upper threshold. - If 0, the voltage is below the upper threhsold. - If 1, the voltage is too high.

0x30: MTP_DATA0

Default: 0x00
Data byte 0 for OTP write.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7:0] MTP_RECORD[7:0]	RW, unsigned 0x00	

0x31: MTP_DATA1

Default: 0x00
Data byte 1 for OTP write.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7:0] MTP_RECORD[15:8]	RW, unsigned 0x00	

0x32: MTP_DATA2

Default: 0x00
Data byte 2 for OTP write.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7:0] MTP_RECORD[23:16]	RW, unsigned 0x00	

0x33: MTP_DATA3

Default: 0x00
Data byte 3 for OTP write.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[7:0] MTP_RECORD[31:24]	RW, unsigned 0x00	

0x34: MTP_ADDR

Default: 0x00
Virtual address for OTP write.

BITS AND NAME	TYPE AND RESET	DESCRIPTION
[6:0] MTP_RECORD[38:32]	RW, unsigned 0x00	

Ordering Information

PART NUMBER	TEMPERATURE RANGE	PIN-PACKAGE
TMC5221AWV+	-40°C to +125°C	30 WLCSP (2.3mm × 2.49mm)
TMC5222AWV+	-40°C to +125°C	30 WLCSP (2.3mm × 2.49mm)
TMC5221AWV+T	-40°C to +125°C	30 WLCSP (2.3mm × 2.49mm)
TMC5222AWV+T	-40°C to +125°C	30 WLCSP (2.3mm × 2.49mm)

+Denotes a lead(Pb)-free/RoHS-compliant package.

T = Tape and reel.

Revision History

REVISION NUMBER	REVISION DATE	DESCRIPTION	PAGES CHANGED
0	07/26	Initial release	—

Information furnished by Analog Devices is believed to be accurate and reliable. However, no responsibility is assumed by Analog Devices for its use, nor for any infringements of patents or other rights of third parties that may result from its use. Specifications subject to change without notice. No license is granted by implication or otherwise under any patent or patent rights of Analog Devices. Trademarks and registered trademarks are the property of their respective owners. All Analog Devices products contained herein are subject to release and availability.