*Click **here** to ask an associate for production status of specific part numbers.*

TMC5271

2.1V to 20V, 1.6A_{RMS} Stepper Driver and Controller

General Description

The TMC5271 is a miniaturized smart high-performance single-axis stepper motor controller and driver IC with serial communication interfaces (SPI and UART) and extensive diagnostic capabilities. Highest integration, high energy efficiency, and a small form factor enable miniaturized and scalable systems for cost-effective solutions.

StealthChop2™ ensures absolutely noiseless operation combined with maximum efficiency and the optimal motor torque control.

The TMC5271 combines a flexible 8-point ramp generator with reduced jerk for automatic positioning with a smart stepper motor driver based on a 256 microstep built-in indexer and two fully integrated 20V, 2.24A_{MAX} H-bridges plus non-dissipative integrated current sensing (ICS), which eliminates bulky external sense resistors resulting in space and power savings.

The integrated power MOSFETs' low impedance results in high driving efficiency and minimal heat generation.

The typical total $R_{DS(ON)}$ (high side plus low side) is 0.155Ω with a maximum RMS current per H-bridge of I_{RMS} = 1.6A_{RMS} and a maximum output current per H-Bridge at I_{MAX} = 3.0A_{MAX} (limited by overcurrent protection).

The TMC5271 features abundant diagnostics and protection functions such as short-circuit protection/overcurrent protection, thermal shutdown, and undervoltage lockout.

The TMC5271 is available in a tiny 2.97mm x 3.13mm, 36-pin WLCSP package.

Applications

- Wearables
- Personal Portable Devices
- Optical Systems, Lens Control
- CCTV, Surveillance, Conference Systems Focus/Zoom and Pan/Tilt Control
- Patch and Insulin Pumps
- Pipettes and Liquid Handling
- Small Printing and Scanning Devices
- Lab and Office Automation
- Space-Constrained Stepper Motor Applications

Benefits and Features

- 2.1V to 20V DC Voltage Range
- Two 20V H-Bridges
- Low $R_{DS(ON)}$ (HS Plus LS): 0.155Ω Typical (T_A = +25°C)
- Current Ratings per H-Bridge (Typical at +25°C):
  - Bridge Peak Current I_{MAX} = 3.0A_{MAX}
  - Full-Scale Current IFS = 1.6A_{RMS} (2.24A Sine-Wave Peak)
- Fully Integrated Lossless Current Sensing (ICS)
- Eight-Point Motion Controller for Minimum Jerk
- Step/Direction Interface with MicroPlyer™ Step Interpolation
- SPI or Single-Wire UART Interface
- Quadrature Encoder Interface
- Reference Switch Inputs
- TriCoder Sensorless Standstill Steploss Detection and Full-Step Encoder
- Highest 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 for Energy Savings up to 75%
- Passive Braking and Freewheeling Mode
- Bridge Disable, Reset, and Low-Power Mode
- Motor Phase and Chip Temperature Measurement
- Full Protection and Diagnostics
- 2.97mm x 3.13mm, 36-Pin WLCSP

Ordering Information appears at end of data sheet.

19-101709; Rev 0; 6/23

Simplified Block Diagram

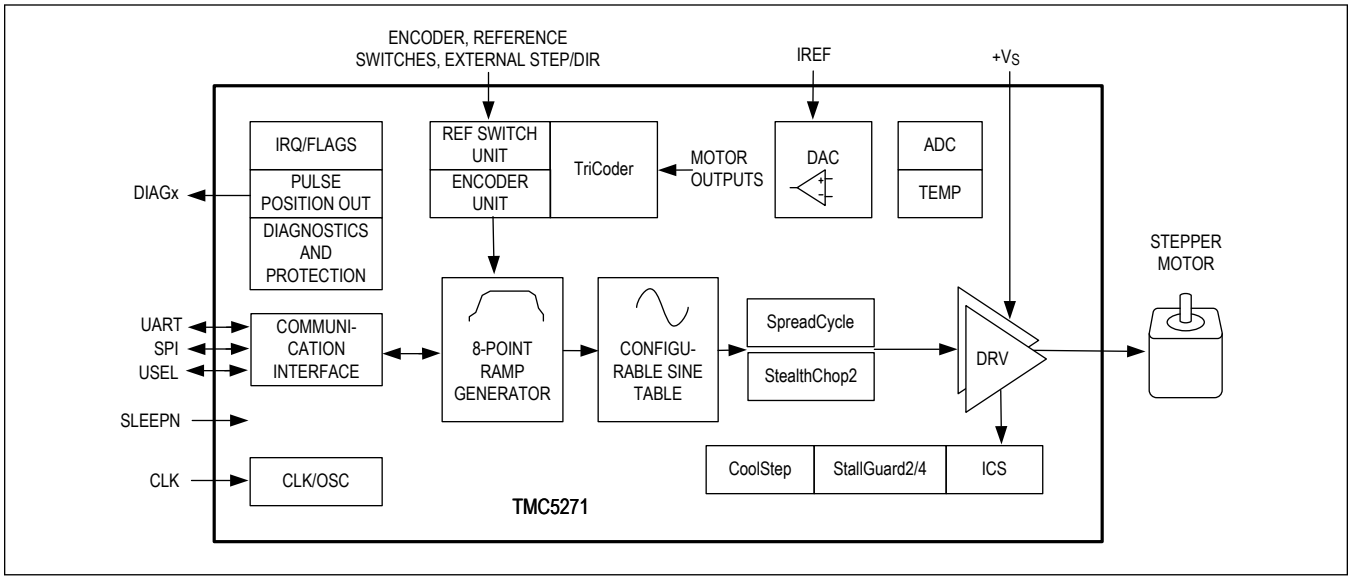


TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

LIST OF FIGURES

LIST OF FIGURES (CONTINUED)

LIST OF TABLES

Absolute Maximum Ratings

V_S to GND ..... -0.3V to 22V
V_{DD1V8} to GND ..... -0.3V to min(2.2, V_S + 0.3)V
PGND to GND ..... -0.3V to +0.3V
OA1, OA2, OB1, OB2 ..... -0.3V to V_S + 0.3V
SLEEPN ..... -0.3V to V_S + 0.3V
IREF to GND ..... -0.3V to V_{DD} + 0.3V
V_{CC_IO} to GND ..... -0.3V to 5.5V
Logic Input Voltage to GND ..... -0.3V to V_{CC_IO} + 0.5V
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

36 WLCSP

Package Code	W362A3+1
Outline Number	21-100643
Land Pattern Number	*Refer to Application Note 1891*
Thermal Resistance, Four-Layer Board: JEDEC PCB, 2 x 2 Via Array, M1 Traces, M2/M3 90% Cu, M4 25% Cu	
Junction to Ambient (θ_{JA})	33.22°C/W
Junction to Board (θ_{JB})	4.13°C/W
Junction to Case (θ_{JC})	0.19°C/W
Junction-to-Board Characterization Parameter (ψ_{JB})	3.89°C/W
Junction-to-Top (of Package) Characterization Parameter (ψ_{JT})	0.59°C/W

For the latest package outline information and land patterns (footprints), go to *www.maximintegrated.com/packages*. 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 *www.maximintegrated.com/thermal-tutorial*.

Electrical Characteristics

(V_S = 2.1V to 20V, R_{REF} = from 10kΩ to 60kΩ, typical values assume T_A = +25°C, V_S = 16V, SLEEPN = V_S, and V_{CC_IO} = 3.3V, limits are 100% tested at T_A = +25°C. Limits over the operating temperature range and relevant supply voltage range are guaranteed by design and characterization.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
POWER SUPPLY						
Supply Voltage Range	V_S		2.1		20	V
Sleep Mode Current Consumption	I_{VS}	V(SLEEPN) = 0, HV outputs pulled up		28n	2u	A
Quiescent Current Consumption	I_{VS}	V(SLEEPN) = 1, qsc_sts_ena = 1, adc_en = 0, qsc_enc_en = 0		480	800	µA
Driver Off Current Consumption	I_{VS}	V(SLEEPN) = 1, qsc_sts_ena = 0		3.4	5	mA
1.8V Regulator Output Voltage	V_{VDD}	Internally short-circuit protected, V_S = 2.1V, I_{LOAD} = 20mA		1.8		V
V_{DD} Current Limit	$IV18_{LIM}$	Internally short-circuit protected	20			mA

Electrical Characteristics (continued)

(V_S = 2.1V to 20V, R_{REF} = from 10kΩ to 60kΩ, typical values assume T_A = +25°C, V_S = 16V, SLEEPN = V_S, and V_{CC_IO} = 3.3V, limits are 100% tested at T_A = +25°C. Limits over the operating temperature range and relevant supply voltage range are guaranteed by design and characterization.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Logic I/O Supply	V_{CC_IO}	V_{CC_IO} can be either 1.8V, 3.3V, or 5V level and is defined by the user application circuit	1.6		5.5	V
Sleep Mode Current Consumption	I_{VCC}	V(SLEEPN) = 0			1	µA
Quiescent Current Consumption	I_{VCC}	V(SLEEPN) = 1		36	70	µA
	I_{SLEEPN}			32	60	
LOGIC LEVEL INPUTS/OUTPUTS						
Input Voltage Level—High	V_{IH}		1.2			V
Input Voltage Level—Low	V_{IL}				0.65	V
Input Hysteresis	V_{HYS}			85		mV
Pull-Down Current	I_{PD}	To GND	6	13	22	µA
Pull-Up Current	I_{PU}	To V_{DD}	8	15	23	µA
Open-Drain Output Logic-Low Voltage	V_{OL}	I_{LOAD} = 5mA			0.4	V
Push-Pull Output Logic-High Voltage	V_{OH}	I_{LOAD} = 5mA	V_{CC_IO} - 400mV			V
Open-Drain Output Logic-High Leakage Current	I_{OH}	V(PIN) = 5.5V, V_{CCIO} = 5.5V	-1		+1	µA
SLEEPN Voltage Level High	VIH_{SLEEPN}		1.3			V
SLEEPN Voltage Level Low	VIL_{SLEEPN}				0.4	V
OUTPUT SPECIFICATIONS						
Output On-Resistance Low Side	$R_{DS(ON),LS}$	Full-scale bits = 00		0.19	0.375	Ω
		Full-scale bits = 01		0.1	0.195	
		Full-scale bits = 10		0.07	0.135	
		Full-scale bits = 11		0.055	0.11	
Output On-Resistance High Side	$R_{DS(ON),HS}$			0.1	0.19	Ω
Output Leakage	I_{LEAK}		-10		+10	µA
Output Slew Rate	SR			200		V/µs
PROTECTION CIRCUITS						
Overcurrent Protection Threshold	OCP	Full-scale bits = 00	0.75			A
		Full-scale bits = 01	1.5			
		Full-scale bits = 10	2.25			
		Full-scale bits = 11	3			
Overcurrent Protection Blanking Time	TOCP		0.9	1.6	2.6	µs

Electrical Characteristics (continued)

(V_S = 2.1V to 20V, R_{REF} = from 10kΩ to 60kΩ, typical values assume T_A = +25°C, V_S = 16V, SLEEPN = V_S, and V_{CC_IO} = 3.3V, limits are 100% tested at T_A = +25°C. Limits over the operating temperature range and relevant supply voltage range are guaranteed by design and characterization.)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
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				1.3	V
UVLO Threshold on V_S Hysteresis	UVLOVCCHYS				65		mV
Thermal-Protection Threshold Temperature	TSD				+165		°C
Thermal-Protection Temperature Hysteresis					+20		°C
CURRENT REGULATION							
IREF Pin Resistor Range	R_{REF}			10		60	kΩ
IREF Output Voltage	V_{REF}			0.882	0.9	0.918	V
Full-Scale Current Constant	K_{IFS}	Value is in RMS	FSR_M = 0, FSR_IREF_M = 3		4.1		A x kΩ
			FSR_M = 1, FSR_IREF_M = 3		8.16		
			FSR_M = 2, FSR_IREF_M = 3		12.06		
			FSR_M = 3, FSR_IREF_M = 3		16		
Current Trip Regulation Accuracy	DITRIP1	ITRIG from 15% to 100% FSR, R_{REF} = 10kΩ, FSR_IREF_M = 3		-4		+4	%
FUNCTIONAL TIMING							
Sleep Time	t_{SLEEP}	SLEEPN = 1 to OUT_x three-state				50	µs
Wake-Up Time from Sleep	t_{WAKE}	SLEEPN = 0 to normal operation			300	500	µs
Enable Time	t_{EN}	Time from CSN/AD2 pin rising edge to driver on				1	µs
Disable Time	t_{DIS}	Time from CSN/AD2 pin rising edge to driver off				1	µs
CLOCK OSCILLATOR AND INPUT							
Internal Clock Frequency	f_{CLKOSC}			11.9	12.5	13.2	MHz
External Clock Frequency	f_{CLK}			8	16	20	MHz
External Clock Duty Cycle	t_{CLKL}			40		60	%
External Clock Detection in Cycles				4		8	ns

Electrical Characteristics (continued)

(V_S = 2.1V to 20V, R_{REF} = from 10kΩ to 60kΩ, typical values assume T_A = +25°C, V_S = 16V, SLEEPN = V_S, and V_{CC_IO} = 3.3V, limits are 100% tested at T_A = +25°C. Limits over the operating temperature range and relevant supply voltage range are guaranteed by design and characterization.)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
External Clock Timeout Detection in Cycles of Internal f_{CLKOSC}				12		16	Cycles
External Clock Detection Lower Frequency Threshold				4			MHz
SPI TIMING							
SCK Valid Before or After Change of CSN	t_{CC}			t_{CLK}			ns
CSN High Time	t_{CSH}			4 x t_{CLK}			ns
SCK Low Time	t_{CL}			20			ns
SCK High Time	t_{CH}			20			ns
SCK Frequency	f_{SCK}					10	MHz
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, SPI_FLT_SEL = 00			27	40	ns
SPI Input Filter	t_{FILT}	Rising and falling edge, SPI_FLT_SEL = 10			10		ns
STEP/DIRECTION TIMING							
STEP Frequency	f_{STEP}	Maximum microstep resolution	Dedge = 1			f_{CLK}/4	MHz
			Dedge = 0			f_{CLK}/2	
Full-Step Frequency	f_{PS}					f_{CLK}/512	MHz
STEP High Time	t_{SH}			t_{CLK} + 20			ns
STEP Low Time	t_{SL}			t_{CLK} + 20			ns
DIR to STEP Setup Time	t_{DSU}			20			ns
DIR to STEP Hold Time	t_{DSH}			20			ns
DIR/STEP to CLK Setup Time	t_{SU}			10			ns
DIR/STEP to CLK Hold Time	t_{SH}			10			ns
ENCODER TIMING							
Encoder Counting Frequency	f_{CNT}				< 2/3 f_{CLK}	f_{CLK}	MHz
ABN Input Low Time	t_{ABNL}			3 x t_{CLK} + 20			ns

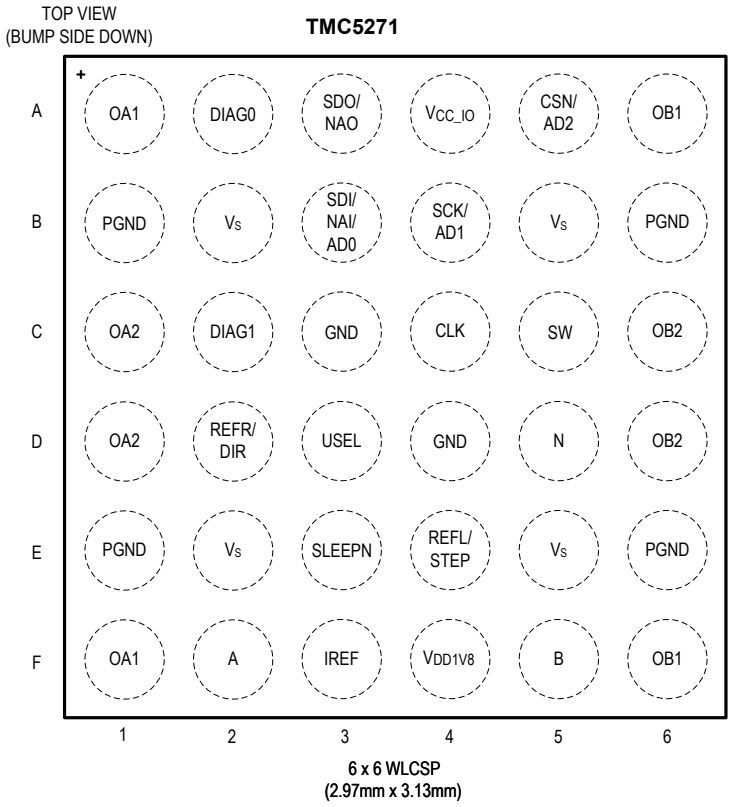
Electrical Characteristics (continued)

(V_S = 2.1V to 20V, R_{REF} = from 10kΩ to 60kΩ, typical values assume T_A = +25°C, V_S = 16V, SLEEPN = V_S, and V_{CC_IO} = 3.3V, limits are 100% tested at T_A = +25°C. Limits over the operating temperature range and relevant supply voltage range are guaranteed by design and characterization.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
ABN Input High Time	t_{ABNH}		3 x t_{CLK} + 20			ns
ABN Spike Filtering Time	$t_{FILTABN}$	Rising and falling edge		3 x t_{CLK}		ns
ADC/TEMPERATURE MEASUREMENT						
ADC Resolution				8		Bit
Temperature Measurement Resolution	T_{MEAS}			2.5		°C
Temperature Measurement Accuracy	T_{SIGMA}			1.5		°C
ADC Sample Rate	$f_{SAMPLE,ADC}$	$f_{CLK,EXT}$ = 20MHz, $f_{CLK,INT}$ = 12.5MHz		$f_{CLK,INT}$/ 396		MHz
BEMF ENCODER						
Pull-Down Resistance on OA1/OB1	R_{PDBEMF}	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	
		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	

Pin Configuration

36 WLCSP



Pin Description

PIN	NAME	FUNCTION	REF SUPPLY	TYPE
E2, E5, B2, B5	V_S	Motor Supply Voltage. Provide filtering capacitor near pin with short loop to nearest PGND pin (or using the GND plane).	—	Supply
F4	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
F3	IREF	Analog Reference Current for Current Scaling. Provide external resistor R_{REF} to GND.	V_{DD1V8}	Analog Input
A4	V_{CC_IO}	Digital IO Supply Voltage Provided from External Source to Define Circuit IO Level	—	Supply
C4	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.	—	Digital Input (Pull-Down)
A5	CSN/AD2	SPI Chip Select Input (Active Low) (USEL = 0) or Address Input 2 (+4) in UART Mode (USEL = 1)	—	Digital Input (Pull-Up)

Pin Description (continued)

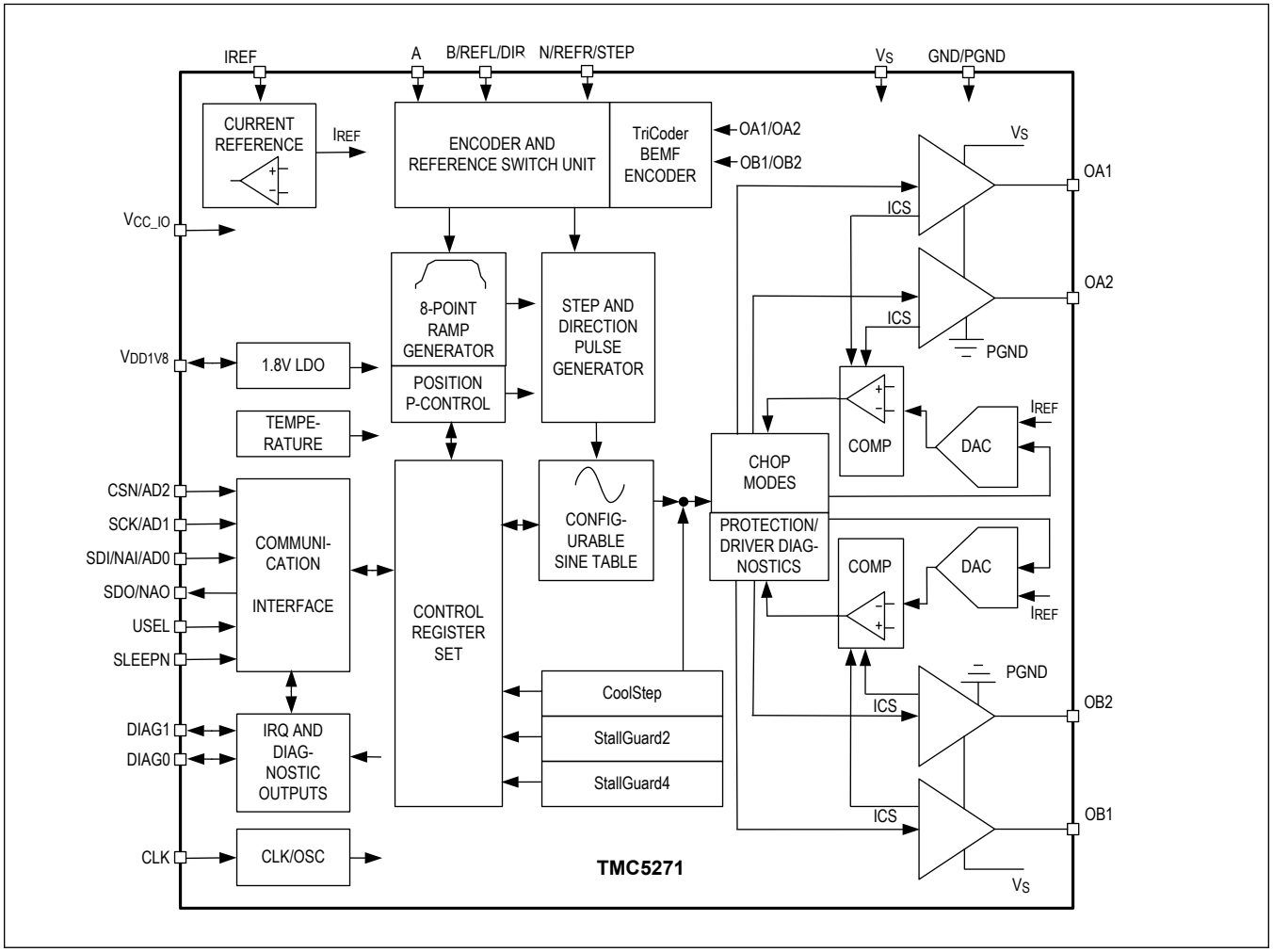
PIN	NAME	FUNCTION	REF SUPPLY	TYPE
B4	SCK/AD1	SPI Serial Clock Input (USEL = 0) or Address Input 1 (+2) in UART Mode (USEL = 0)	—	Digital Input (Pull-Up)
B3	SDI/NAI/AD0	SPI Data Input (USEL = 0) or Next Address Input in Daisy Chain UART Mode or Address Input 0 (+1) in Default UART Mode (USEL = 1)	—	Digital Input (Pull-Up)
A3	SDO/NAO	SPI Data Output (Three-State) in SPI Mode (USEL = 0). Next Address Output (NAO) in UART Mode (USEL = 1). UART Data Output in UART Ring Mode (AD0 = AD1 = AD2 = 1).	V_{CC_IO}	Digital Output
D3	USEL	Interface Selection Pin. When tied low, the SPI interface is enabled. When tied high, the UART interface is enabled.	—	Digital Input (Pull-Down)
A2	DIAG0	Configurable Diagnostics Output DIAG0. Use external pull-up resistor with 4.7kΩ or less in open-drain mode.	V_{CC_IO}	Digital Output
E3	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_enn bit in the GCONF register. If not used, connect to V_{CC_IO} or V_S. (This is a high-voltage pin.) **Note:** Do not use during high motor velocity to prevent from hard stops and potential back-EMF spikes.	V_S	Analog input (Pull-Down)
A1, F1	OA1	Motor Coil A Output 1, Motor 1	V_S	Analog Output
C1, D1	OA2	Motor Coil A Output 2, Motor 1	V_S	Analog Output
A6, F6	OB1	Motor Coil B Output 1, Motor 1	V_S	Analog Output
C6, D6	OB2	Motor Coil B Output 2, Motor 1	V_S	Analog Output
B1, B6, C3, D4, E1, E6	GND/PGND	Supply and System Ground. Connect to the GND plane. Provide a solid connection to GND plane for best heat transfer.	—	GND
C2	DIAG1	Configurable Diagnostic Output DIAG1. Use external pull-up resistor with 4.7kΩ or less in open-drain mode.	V_{CC_IO}	Digital Output

Pin Description (continued)

PIN	NAME	FUNCTION	REF SUPPLY	TYPE
F2	A	Encoder A-Channel Input or External Input Trigger for Controlled Emergency Stop		Digital Input (Pull-Down)
F5	B	Encoder B-Channel Input		Digital Input (Pull-Down)
D5	N	Encoder N-Channel Input		Digital Input (Pull-Down)
C5	SW	Single-Wire UART Data Bus In/Out in UART Communication Mode	V_{CC_IO}	Digital Input/ Output
D2	REFR/DIR	Right Reference Switch Input REFR or Direction Input DIR in S/D Mode or External Input Trigger (with REFL) for Hard Emergency Stop		Digital Input (Pull-Down)
E4	REFL/STEP	Left Reference Switch Input REFL or STEP Input in S/D Mode or External Input Trigger (with REFR) for Hard Emergency Stop		Digital Input (Pull-Down)

Functional Diagrams

TMC5271



Detailed Description

Principles of Operation

The TMC5271 motion controller and stepper motor driver chip is a smart power-conversion component. It is interfaced from an MCU and no additional software is required to control the motors. Apart from basic chip configuration only acceleration parameters, velocity parameters, and target positions must be provided through one of the available interfaces. The TMC5271 offers numerous unique functions, which are enabled by the system-on-chip integration of driver and controller.

Single-Axis Motion Controller and Driver

The TMC5271 includes 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 or an incremental encoder can be connected, e.g., for homing and monitoring. The motion-controller mode is the default mode of operation. Figure 1 shows a high-level block diagram.

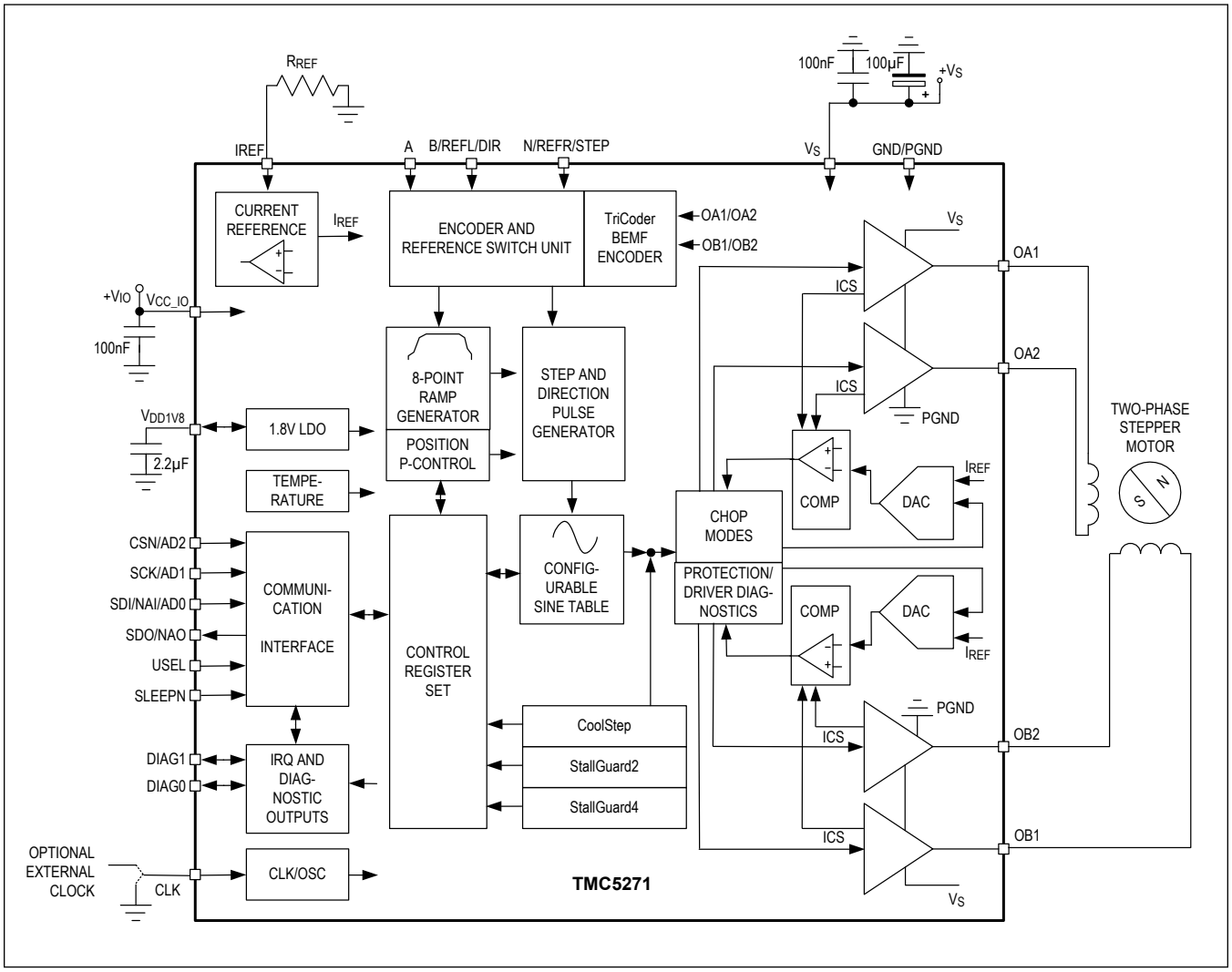


Figure 1. Single-Axis Motion Controller and Driver

Key Concepts

The TMC5271 implements advanced features which are exclusive to products from Analog Devices, Inc. The key features are highlighted in Table 1. These features contribute toward greater precision, greater energy efficiency, higher reliability, smoother motion, and cooler operation in stepper-motor applications.

Table 1. TMC5271 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.

Table 1. TMC5271 Key Concepts (continued)

KEY CONCEPT	DESCRIPTION	FURTHER DETAILS
SpreadCycle	High-precision cycle-by-cycle current control for highest dynamic movements.	See the *Spreadcycle and Classic 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 Load Measurement* and *StallGuard4 Load Measurement* sections for more information.
CoolStep	Uses StallGuard2/4 measurement to adapt the motor current for best efficiency and lowest heat-up of motor and driver.	See the *CoolStep Load Adaptive Current Scaling* section for more information.
MicroPlyer	Microstep interpolator for obtaining full 256-microstep smoothness with lower resolution step inputs starting from full step.	See the *Step/Direction Interface* 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 TMC5271 supports both an SPI interface and a UART-based single-wire interface with CRC checking. Selection of the actual interface combination is done through the USEL pin, which can be hardwired to GND or V_{CC_IO} depending on the desired interface selection.

The SPI interface 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 (e.g., an MCU) and the TMC5271 peripheral always consists of sending one 40-bit command word and receiving one 40-bit status word.

The single-wire interface allows for a direct bidirectional single-wire interface between the host and a TMC5271. It can be driven by any standard UART. No baud rate configuration is required.

In addition, ring mode and daisy chaining are supported for multiple axis control using a single host.

Moving and Controlling the Motor

Integrated 8-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 8-point velocity profile. These settings allow 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 8-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 TMC5271 takes care of the current control and delivers feedback on the state of the driver and motor. MicroPlyer automatically smoothens motion. This mode can be enabled by configuration through SPI or UART. Active edges on the STEP input can be rising or falling. 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 2 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.

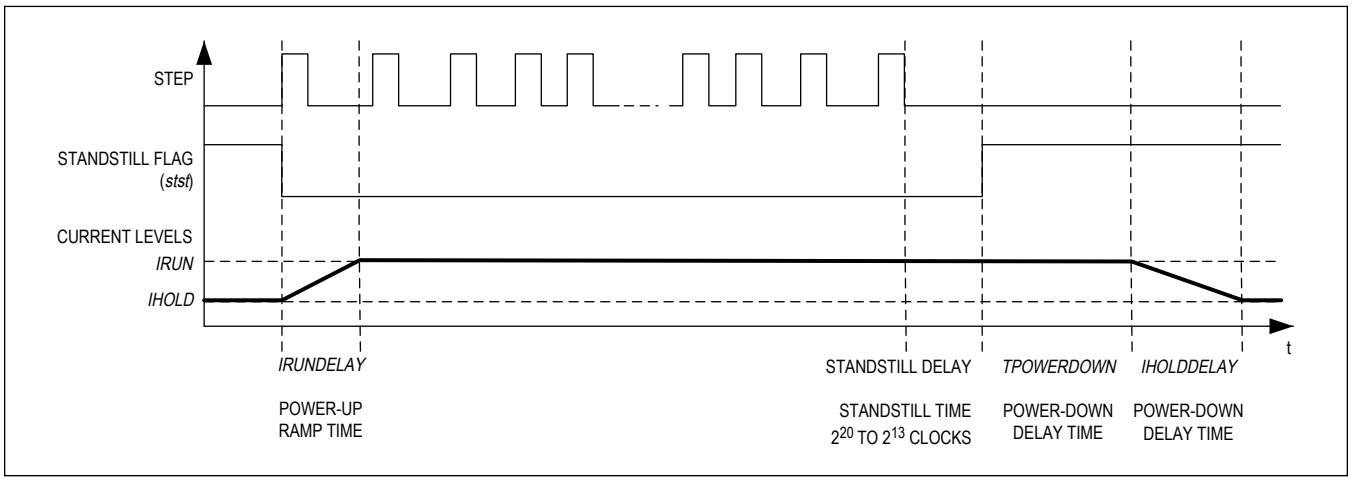


Figure 2. Automatic Motor Current Control at Standstill and Ramp-Up

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 to be used 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. 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.

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

Encoder Interface

The TMC5271 provides an encoder interface for external incremental encoders. The encoder can be used for homing of the motion controller (alternatively to reference switches) and for consistency checks on-the-fly between encoder position and ramp generator position. The encoder interface is also used for a special P-regulator mode for closed-loop position control. A programmable prescaler allows the adaptation of the encoder resolution to the motor resolution. A 32-bit encoder counter is provided.

Standstill Steploss Detection

The sensorless standstill steploss detection feature is based on a BEMF decoder. The BEMF decoder allows the detection of motor motion while the motor is disabled (i.e., 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 becomes 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 has been moved, or to keep track of the number of steps done and correcting for it later.

SPI Interface

SPI Datagram Structure

The TMC5271 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.

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. The datagram structure is shown in Figure 3.

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 (e.g., GSTAT registers).

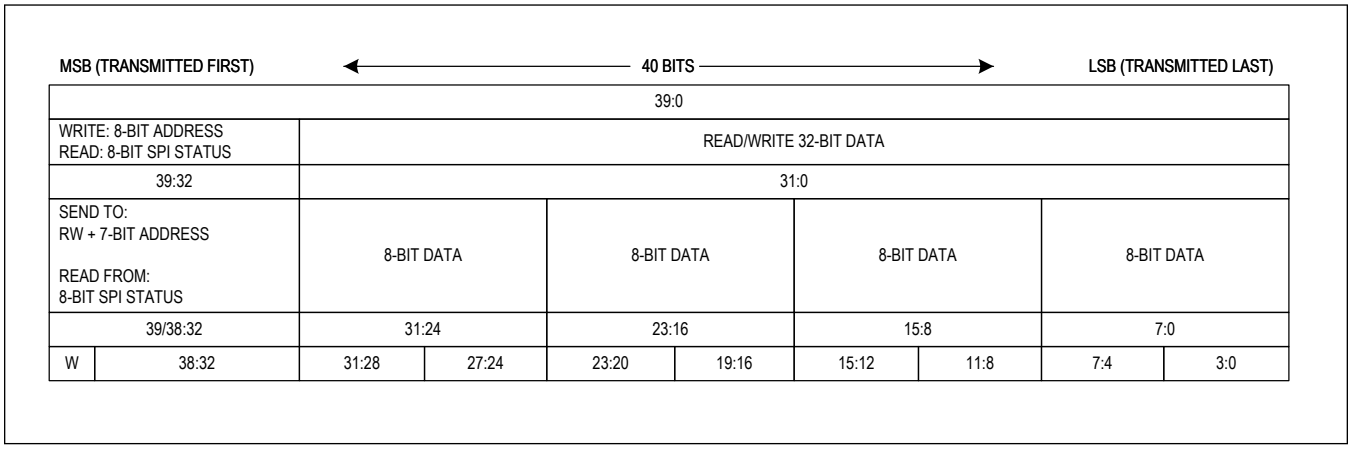


Figure 3. SPI Datagram Structure

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 has to be added to the address for a write access. The SPI interface always delivers data back to the controller, independent of the W bit. The data transferred back is the data read from the address which was transmitted with the previous datagram, if the previous access was a read access. If the previous access was a write access, then 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, reading multiple registers can be done in a pipelined fashion.

Whenever data is read from or written to the TMC5271, 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 has to be set to 0x18 in the access preceding the read access. For a write access to the register (VACTUAL), the address byte has to be set to 0x80 + 0x19 = 0x99. For read access, the data bit might have any value (-). Therefore, they can be set to 0.

Table 2. SPI Read/Write Example Flow

ACTION	DATA SENT TO TMC5271	DATA RECEIVED FROM TMC5271
Read XACTUAL	0x1800000000	0xSS and unused data*

Table 2. SPI Read/Write Example Flow (continued)

ACTION	DATA SENT TO TMC5271	DATA RECEIVED FROM TMC5271
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 becomes latched at the end of each access and is available with the next SPI transfer. [[SPI_STATUS—Status Flags Transmitted With Each SPI Access In Bits 39 To 32]] lists these status flags.

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

BIT	NAME	NOTES
7	Not used	Always 0
6	status_stop	1: Signals stop on motion controller (stop_l, stop_r, stop_sg, virtual_stop_l, virtual_stop_r)
5	Not used	Always 0
4	position_reached	RAMP_STAT[9] = 1: Signals motor target position reached (motion controller only)
3	Not used	Always 0
2	velocity_reached	RAMP_STAT[8] = 1: Signals motor target velocity reached (motion controller only)
1	driver_error	GSTAT[1] = 1: Signals driver error (cleared by clearing bit in GSTAT register, write 1 to the bit position)
0	reset_flag	GSTAT[0] – 1: Signals that a reset has 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 TMC5271 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 TMC5271.

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 4 shows the SPI timing diagram. The SPI interface uses SPI mode 3. The SPI max 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 *Electrical Characteristics* section.

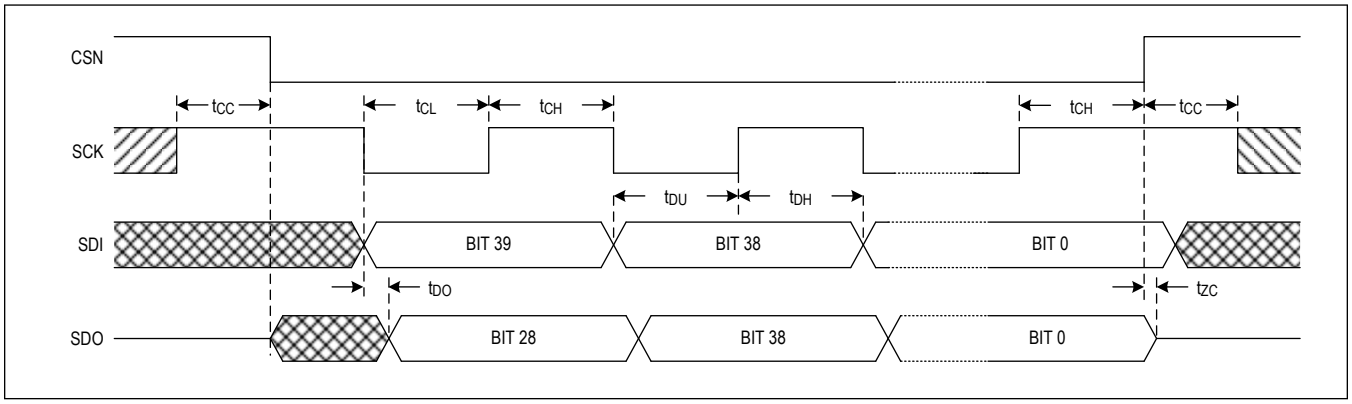


Figure 4. SPI Timing Diagram

UART Single-Wire Interface

The UART single-wire interface allows control of the TMC5271 with any microcontroller UART. It shares the transmit and receive line like an RS485-based interface. Data transmission is secured using a cyclic redundancy check, so that increased interface distances (e.g., over cables between two PCBs) can be bridged without danger of wrong or missed commands even in the event of electromagnetic disturbance. The automatic baud rate detection makes this interface easy to use. It offers different wiring and addressing schemes.

UART Datagram Structure

UART Write Access

EACH BYTE IS LSB TO MSB, HIGHEST BYTE TRANSMITTED FIRST

0:63											
SYNC + RESERVED								8 BIT NODE ADDR	RW + 7 BIT REGISTER ADDR.		32 BIT DATA	CRC
1	0	1	0	RESERVED / DON'T CARES, BUT INCLUDED IN CRC				*NODEADDR*	REGISTER ADDRESS	1	DATA BYTES 3, 2, 1, 0 (HIGH TO LOW BYTE)	CHECKSUM
0	1	2	3	4	5	6	7	8:15	16:23		24:55	56:63

Figure 5. UART Write-Access Datagram Structure

Figure 5 shows the UART write-access datagram structure. A sync nibble precedes each transmission to and from the TMC5271 and is embedded into the first transmitted byte, followed by an addressing byte. Each transmission allows a synchronization of the internal baud rate divider to the UART host clock. The actual baud rate is adapted and variations of the internal clock frequency are compensated. Thus, the baud rate can be freely chosen within the valid range. Each transmitted byte starts with a start bit (logic 0, low level on DIAG1) and ends with a stop bit (logic 1, high level on DIAG1). The bit time is calculated by measuring the time from the beginning of the start bit (1 to 0 transition) to the end of the sync frame (1 to 0 transition from bit 2 to bit 3). All data is transmitted byte wise. The 32-bit data words are transmitted with the highest byte first.

A minimum baud rate of 9000 baud is permissible, assuming a 20MHz clock (worst case for low baud rate). Maximum baud rate is $f_{CLK}/16$ due to the required stability of the baud clock.

When switching to low-power mode the clock source is switched to the internal clock. This results in the maximum baud rate being reduced to 1/16 of the internal oscillator.

NODEADDR is an 8-bit address, which can be configured in the NODECONF register (0x3).

After power-up or reset, the initial peripheral address is selected by the three address signals CSN_AD2, SCK_AD1, and

SDI_AD0 in the range 0 to 7, which then becomes added to the value of NODEADDR.

The peripheral address is determined by the sum of the register NODEADDR and the pin selection given above.

Example: A high level on SDI (with CSN low and SCK low) increments the NODEADDR setting by one.

Bit 7 of the register address identifies a read (0) or a write (1) access.

Example: register address 0x10 is changed to 0x90 for a write access.

The communication becomes reset if a pause time of longer than 63-bit times between the start bits of two successive bytes occurs. This timing is based on the last correctly received datagram. In this case, the transmission needs to be restarted after a failure recovery time of minimum 12-bit times of bus idle time. This scheme allows the UART host to reset communication in case of transmission errors. Any pulse on an idle data line below 16 clock cycles is treated as a glitch and leads to a timeout of 12-bit times, for which the data line must be idle. Other errors, like wrong CRC, are also treated the same way. This allows a safe resynchronization of the transmission after any error conditions. It should be noted that due to this mechanism an abrupt reduction of the baud rate to less than 15% of the previous value is not possible.

Each accepted write datagram becomes acknowledged by the receiver by incrementing an internal cyclic datagram counter (8 bits). Reading out the datagram counter allows the UART host to check the success of an initialization sequence or single write accesses. Read accesses do not modify the counter.

UART Read Access

EACH BYTE IS LSB TO MSB, HIGHEST BYTE TRANSMITTED FIRST

<table>
<tr><td colspan="12">0:31</td></tr>
<tr><td colspan="8">SYNC + RESERVED</td><td colspan="2">8-BIT NODE ADDRESS</td><td colspan="2">7-BIT REGISTER ADDR + RW</td><td>CRC</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td colspan="4">RESERVED/DON'T CARES, BUT INCLUDED IN CRC</td><td colspan="2">NODEADDR</td><td>REGISTER ADDRESS</td><td>0</td><td>CHECKSUM</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td colspan="2">8:15</td><td colspan="2">16:23</td><td>24:31</td></tr>
</table>

Figure 6. UART Read-Access Request Datagram Structure

The read access request datagram structure as shown in Figure 6 is identical to the write-access datagram structure but uses a lower number of user bits. Its function is the addressing of the UART node and the transmission of the desired register address for the read access. The TMC5271 responds with the same baud rate as the UART host uses for the read request.

To ensure a clean bus transition from the host to the node, the TMC5271 does not immediately send the reply to a read access, but it uses a programmable delay time after which the first reply byte becomes sent following a read request. This delay time can be set in multiples of 8-bit times using the SENDDELAY time setting (default = 8-bit times) according to the needs of the UART host. In a multinode system, set SENDDELAY to a minimum of 2 for all nodes. Otherwise, a nonaddressed node might detect a transmission error upon read access to a different node.

EACH BYTE IS LSB TO MSB, HIGHEST BYTE TRANSMITTED FIRST

0:63											
SYNC + RESERVED								8-BIT NODE ADDRESS	7-BIT REGISTER ADDR + RW		32-BIT DATA	CRC
1	0	1	0	RESERVED (0)				0xFF	REGISTER ADDRESS	0	DATA BYTES 3, 2, 1, 0 (HIGH TO LOW BYTE)	CHECKSUM
0	1	2	3	4	5	6	7	8:15	16:23		24:55	56:63

Figure 7. UART Read-Access Reply Datagram Structure

The read response as shown in Figure 7 is sent to the UART host using address code %11111111. The transmitter is switched inactive 4-bit times after the last bit is sent.

Address %11111111 is reserved for read accesses going to the UART host. A node must not use this address.

UART Signals

The UART interface on the TMC5271 comprises the five signals listed in Table 4. In UART mode, each node checks the single-wire pin DIAG1 for correctly received datagrams with its own address continuously. The pin is switched as input during this time. It adapts to the baud rate based on the sync nibble, as described in the UART Write Access section. In case of a read access, it switches on its output driver on DIAG1 and sends its response using the same baud rate.

Table 4. TMC5271 UART Interface Signals

SIGNAL	DESCRIPTION
DIAG1	Data input and output in default UART mode. Data input in UART ring mode.
CSN/AD2	Bit 2 of UART address increment (+4)
SCK/AD1	Bit 1 of UART address increment (+2)
SDI/NAI/AD0	Bit 0 of UART address increment (+1) Tie to NAO of previous IC in chain for sequential addressing scheme.
SDO/NAO	Next address output (NAO) pin for chained sequential addressing scheme (reset default = high). Data output in UART ring mode.

CRC Calculation

An 8-bit CRC polynomial is used for checking both read and write access. It allows detection of up to eight single bit errors. The CRC8-ATM polynomial with an initial value of zero is applied LSB to MSB, including the sync and addressing byte. The sync nibble is assumed to always be correct. The TMC5271 responds only to correctly transmitted datagrams containing its own node address. It increases its datagram counter for each correctly received write-access datagram.

$CRC = x^8 + x^2 + x^1 + x^0$

Serial calculation example:

CRC = (CRC << 1) OR (CRC.7 XOR CRC.1 XOR CRC.0 XOR [new incoming bit])

C-Code Example for CRC Calculation

```
void swuart_calcCRC(uint8_t* datagram, uint8_t datagramLength)
{
  int i,j;
  uint8_t* crc = datagram + (datagramLength-1); // CRC located in last byte of message
  uint8_t currentByte;
```

```
  *crc = 0;

  for (i = 0; i<(datagramLength-1); i++) {      // Execute for all bytes of a message
    currentByte = datagram[i];                  // Retrieve a byte to be sent from array
    for (j = 0; j<8; j++) {
      if ((*crc >> 7) ^ (currentByte&0x01))   // Update CRC based result of XOR operation
      {
        *crc = (*crc << 1) ^ 0x07;
      }
      else
      {
        *crc = (*crc << 1);
      }
      currentByte = currentByte >> 1;
    } // for CRC bit
  } // for message byte
}
```

Addressing Multiple Nodes

There are three options to connect multiple nodes:

1. Direct node addressing if only one or up to seven nodes are to be connected
2. Daisy chaining for more than seven nodes (see the *UART Daisy Chaining* section)
3. Ring mode for more than seven nodes and when trace lengths between nodes is short (see the *UART Ring Mode* section)

If only one or up to seven TMC5271 devices are addressed by a host using a single UART bus interface, a simple hardware-address selection can be used. The individual UART node addresses are set by connecting the UART address pins (SDI, SCK, and CSN) to V_{CC_IO} and GND.

Note: If all three address pins (AD0, AD1, and AD2) are set to 1 or connected to V_{CC_IO}, the UART is working in ring mode (see the *UART Ring Mode* section).

UART Daisy Chaining

If more than seven nodes need to be connected to the same UART bus, a different approach must be used. This approach can address up to 255 nodes by using the output NAO (SDO) as a selection pin for the bit 0 address pin of the next device as shown in Figure 8 and Table 5. Proceed as follows:

- Tie all address pins as well as SDI/NAI/AD0 of the first TMC5271 to GND.
- Connect the SDO/NAO output of the first TMC5271 to the next node's address[0] pin (SDI/AD0). Connect further nodes in the same fashion.
- Now, the first node responds to address 0. The following nodes are set to address 1.
- Program the first TMC5271 to its specific node address. Note: Once a node is initialized with its node address, its SDO/NAO output which is tied to the next nodes's address[0] pin (SDI/AD0), has to be programmed to logic 0 to differentiate the next node from all following nodes.
- Now, the second node is accessible and can get its specific node address. Further nodes can be programmed to their specific node addresses sequentially.

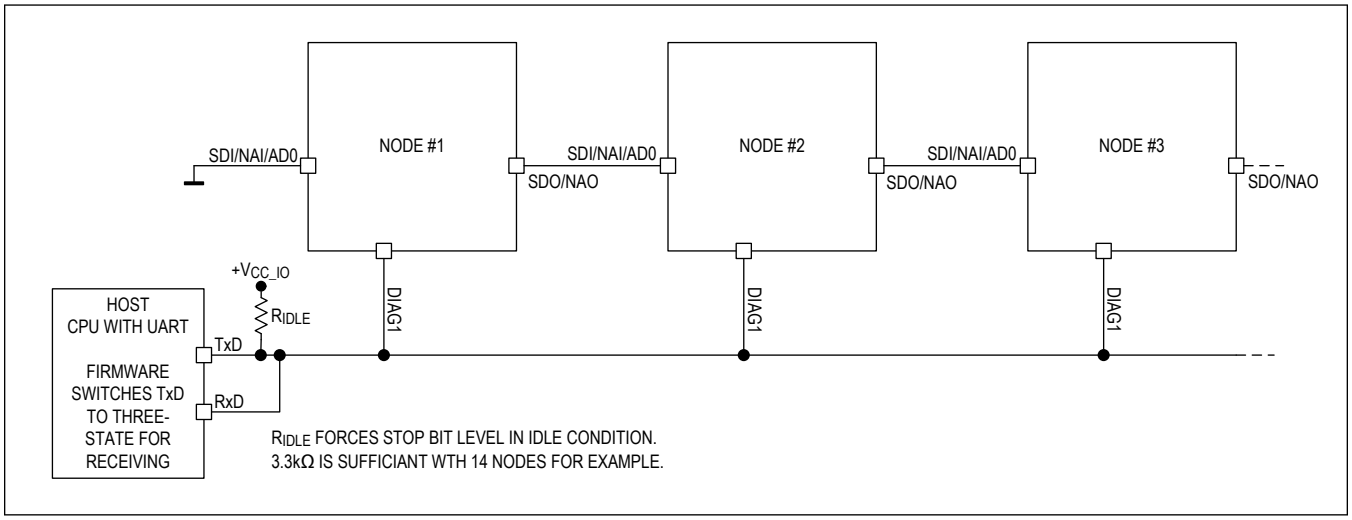


Figure 8. UART Daisy-Chaining Example

Table 5. UART Example for Addressing up to 255 Nodes

PHASE	NODE #1	NODE #2	NODE #3
Addressing phase 1	Address 0, NAO is high	Address 1	Address 1
Addressing phase 2	Program to address 254 and set NAO low	Address 0, NAO is high	Address 1
Addressing phase 3	Address 254	Program to address 253 and set NAO low	Address 0
Addressing phase 4	Address 254	Address 253	Program to address 252 and set NAO low
Addressing phase x	Continue procedure	—	—

UART Ring Mode

In the UART ring mode, all nodes are cascaded in a ring structure. Ring mode is enabled by setting all three UART addressing pins SDI/NAI/AD0, SCK/AD1, and CSN/AD2 to 1 (connect to V_{CC_IO} or keep open as they have internal pull-up resistors). The DIAG1 pin is used as a UART data input. The SDO/NAO pin is used as UART data output. SDO/NAO stays high (idle state) and does not forward data incoming at the SW pin until NODEADDR has been programmed. All data is forwarded from DIAG1 to SDO/NAO after NODEADDR has been programmed to an address value different from 0. The structure is shown in Figure 9.

The intention is to allow for a simple flexible addressing method without the need for using the address pins SDI/NAI/AD0, SCK/AD1, and CSN/AD2, or the chained sequential addressing scheme. At the same time, the distance between each two nodes can be kept short due to the chain structure and the load on each line is only a single input. Therefore, the logical ring is optimal when wiring cost/PCBA-level routing cost is critical. If the physical structure on the board resembles a line rather than a ring, the nodes can be cascaded in an interleaved way so that each second node is connected in a left-to-right direction and back. This way, the distance from the last node's data output to the host is no longer than the distance between three nodes.

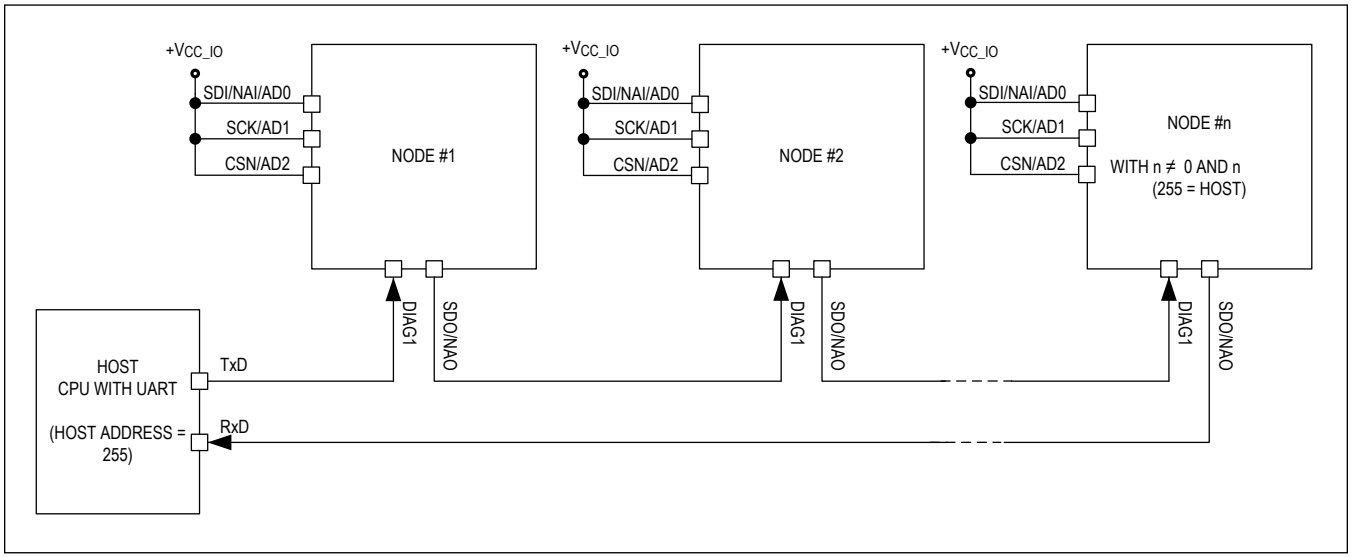


Figure 9. UART Ring-Mode Wiring Example

Be aware that data can only pass from any node through the ring structure back to the host after all nodes have been assigned a unique address different from 0. Do not assign two identical addresses.

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 SD bit in the GCONF register needs to 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 incremental encoder option as well as the reference switch inputs are also not available.

Note: In step/direction mode, the motion controller is not just switched off but completely disabled and not clocked to reduce overall power consumption.

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.

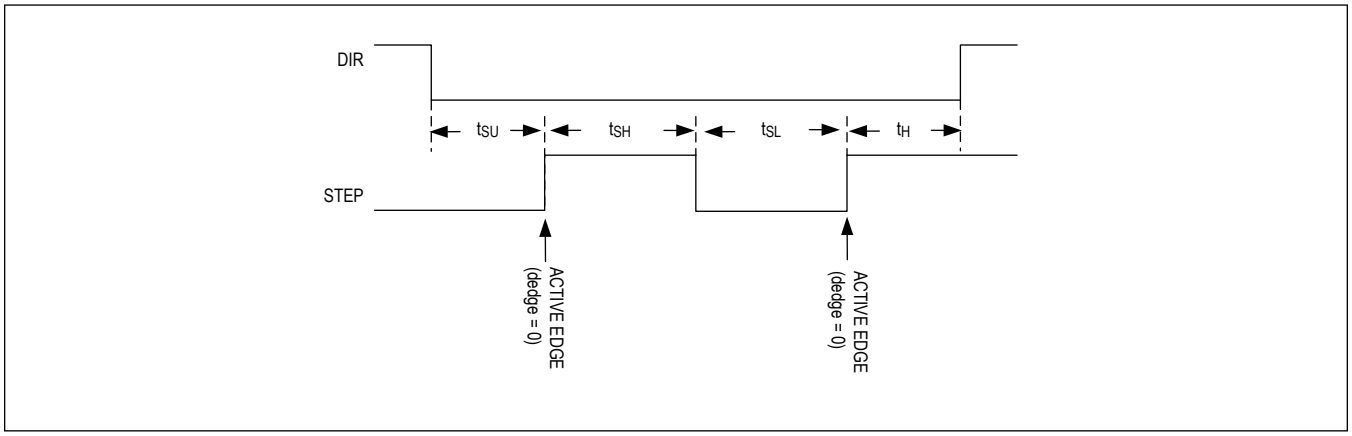


Figure 10. STEP/DIR Signal Timing

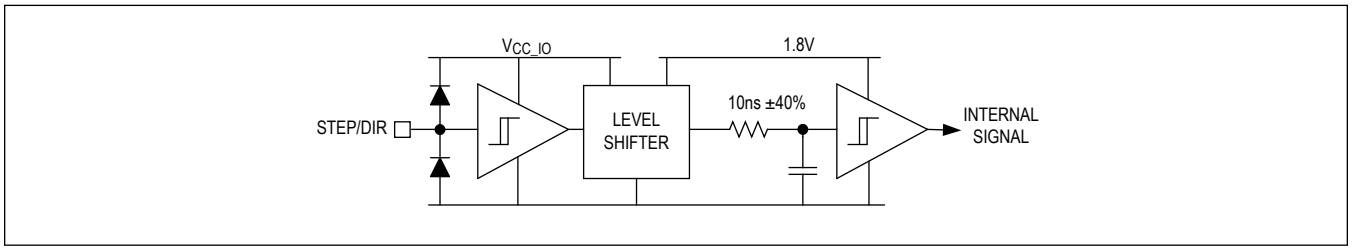


Figure 11. STEP/DIR Signal Input Filter Structure

Changing Resolution

A reduced microstep resolution allows 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 the 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 would lead to an asymmetrical run when comparing a motor running clockwise and counterclockwise. Table 6 shows 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 x 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 6. 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 intpol bit in the CHOPCONF register.

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 2^{20} (approximately one million system clock cycles), for an even distribution of 256 microsteps. At 16MHz system clock frequency, this results in a minimum step input frequency of 16Hz for MicroPlyer operation. A lower step rate causes the STST bit to be set, which indicates a standstill event. At that frequency, microsteps occur at a rate of (system clock frequency)/2^{16} ~ 256Hz. 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.

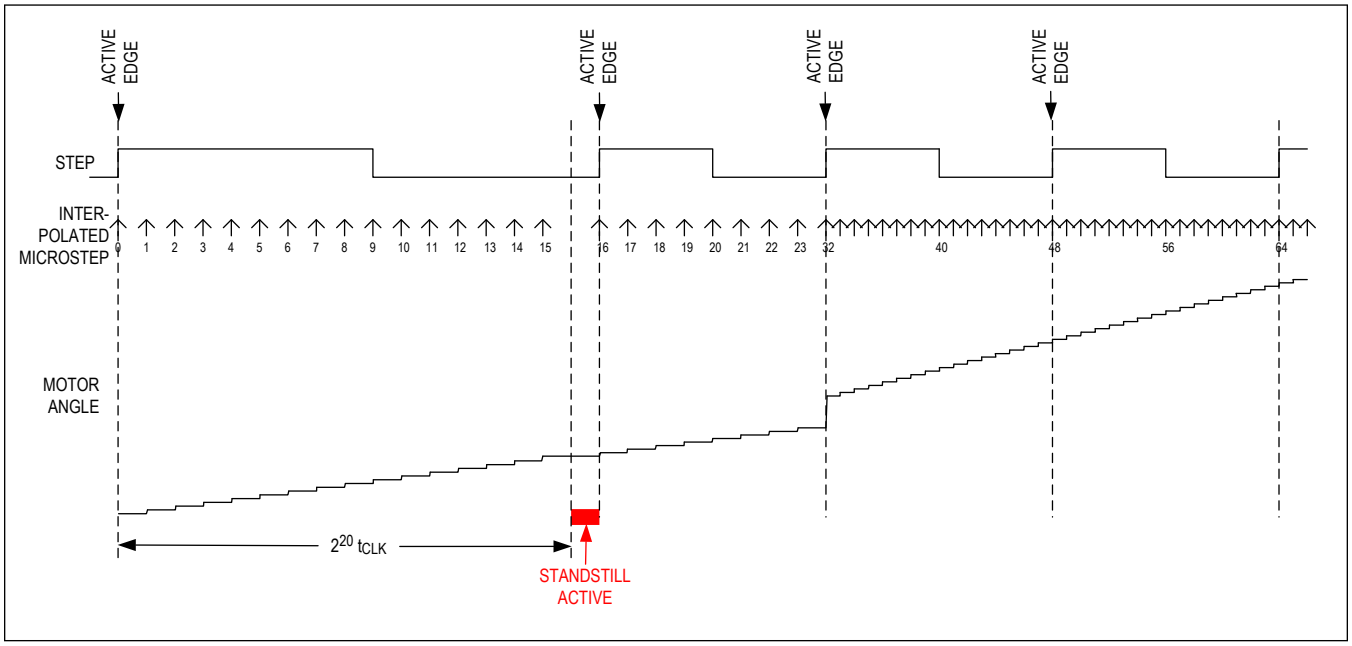


Figure 12. MicroPlyer Microstep Interpolation with Rising Step Signal Frequency (Example: 16 to 256)

In Figure 12, 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 did 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: TMC5271 provides a configurable standstill detection timer length. STANDSTILL_TIME in the DRV_CONF register allows configuration with eight different options.

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 indoor 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 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 have to be taken into account 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, e.g., 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 x 2^{18} t_{CLK}, ≤ 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 x PWM_OFS_AUTO x (IRUN + 1)/32 < (PWM_SCALE_SUM/4) < 4 x PWM_OFS_AUTO x (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 best conditions for automatic tuning with the TMC5271-EVAL-KIT. Use application-specific parameters for PWM_GRAD and PWM_OFS for initialization in firmware to provide 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.

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 optimum tuning conditions using the TMC5271-EVAL-KIT kit. 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 FSR, FSR_IREF, GLOBALSCALER, or V_S voltage invalidates the result of the automatic tuning process. Motor current regulation cannot compensate significant changes until the next AT#1 phase. Automatic tuning adapts to changed conditions whenever AT#1 and AT#2 conditions are fulfilled in the later operation.

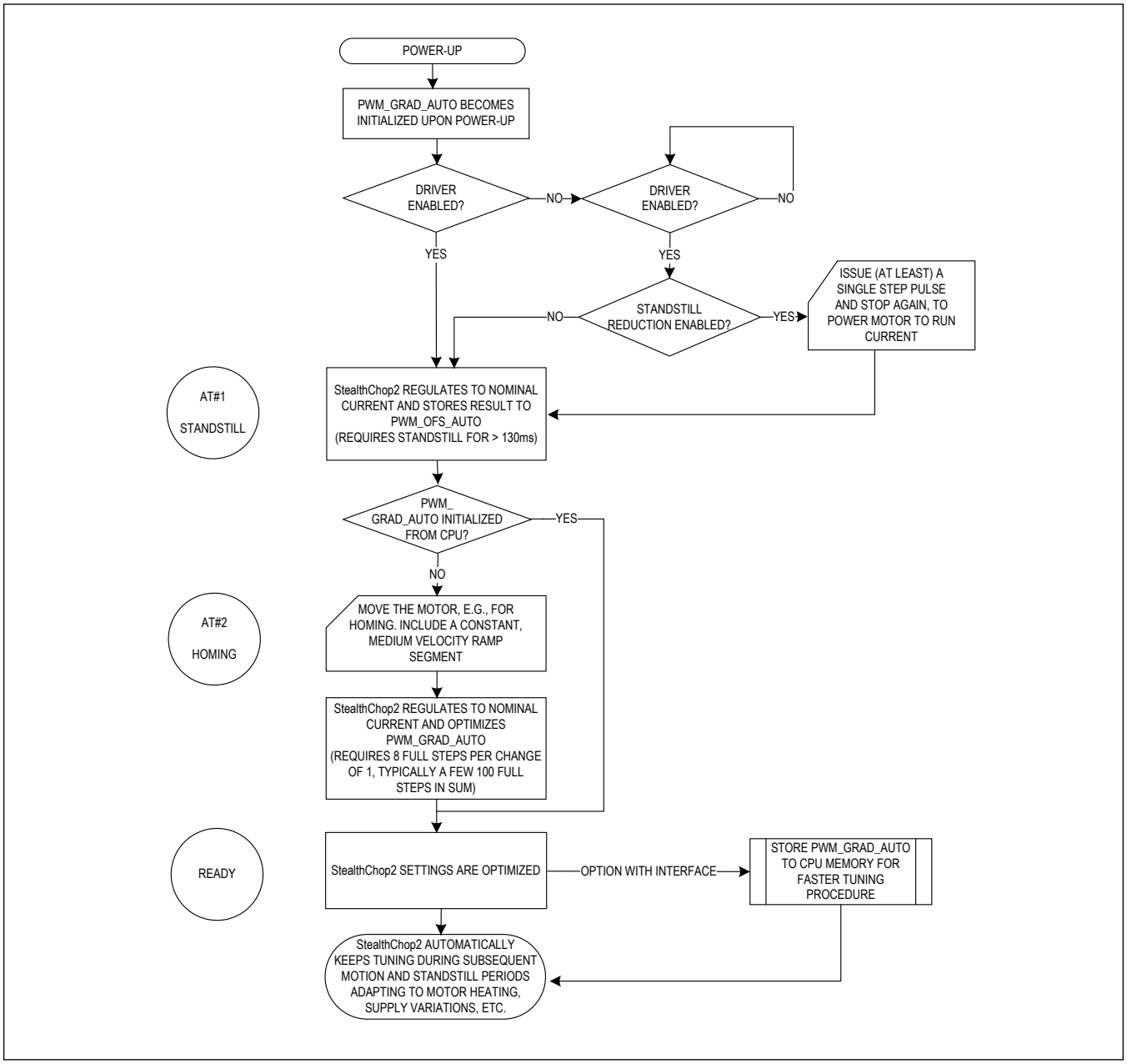


Figure 13. StealthChop2 Automatic Tuning Procedure

StealthChop2 Options

To match the motor current to a certain level, the effective PWM voltage becomes 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 (e.g., 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 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 automatic tuning mode to respond to a parameter change, e.g., due to motor heat-up or change of supply voltage.

Nonautomatic 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 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–50kHz is good for most applications. It balances low current ripple and good higher velocity performance vs. dynamic power dissipation. Example frequencies and settings are given in Table 8 as well as recommended values.

Table 8. Choice of PWM Frequency for StealthChop2

CLOCK FREQUENCY f_{CLK} (MHz)	PWM_FREQ = %00 f_{PWM} = 2/1024 f_{CLK} (kHz)	PWM_FREQ = %01 f_{PWM} = 2/683 f_{CLK} (kHz)	PWM_FREQ = %10 f_{PWM} = 2/512 f_{CLK} (kHz)	PWM_FREQ = %11 f_{PWM} = 2/410 f_{CLK} (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_auto_grad = 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 variation of the motor target current (e.g., change of V_{REF}). During 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 good setting and setting too small for PWM_REG.

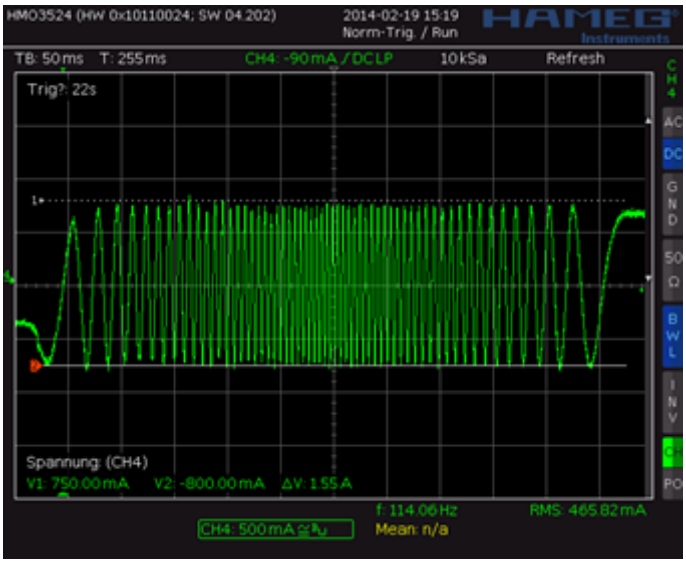


Figure 14. StealthChop2: Good Setting for PWM_REG

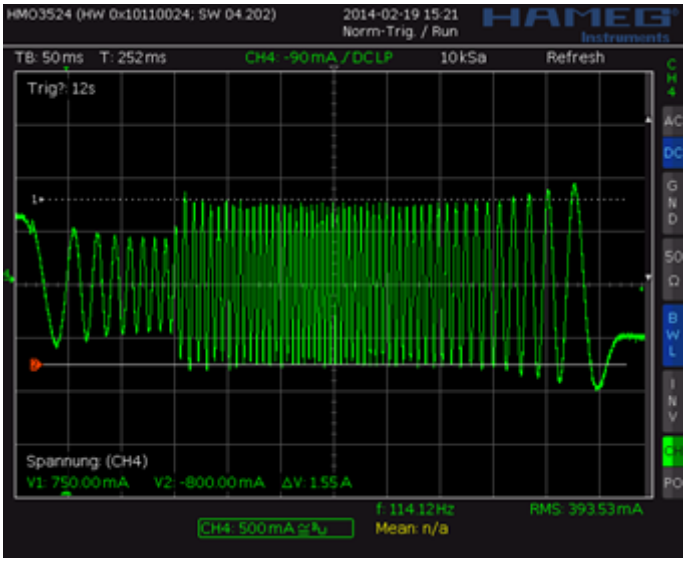


Figure 15. StealthChop2: Setting Too Small for PWM_REG During AT#2

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

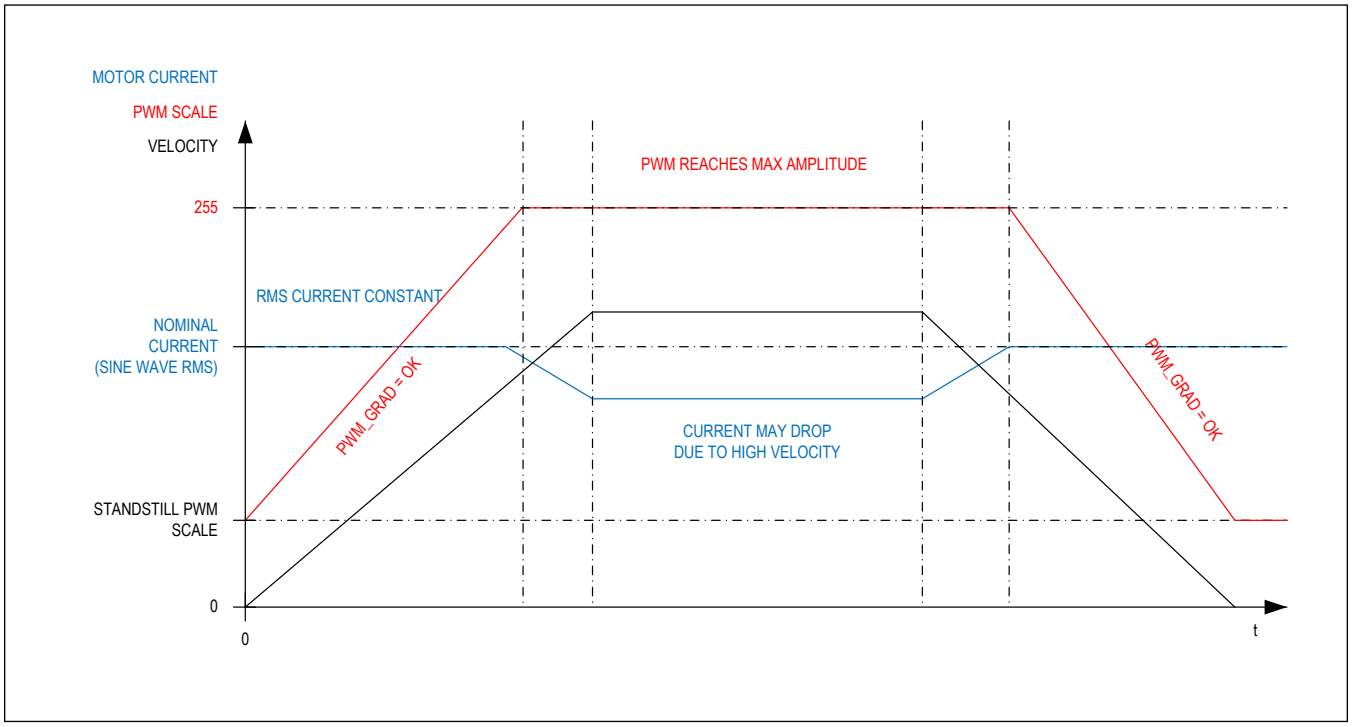


Figure 16. Successfully Determined PWM_GRAD(_AUTO) and PWM_OFS(_AUTO)

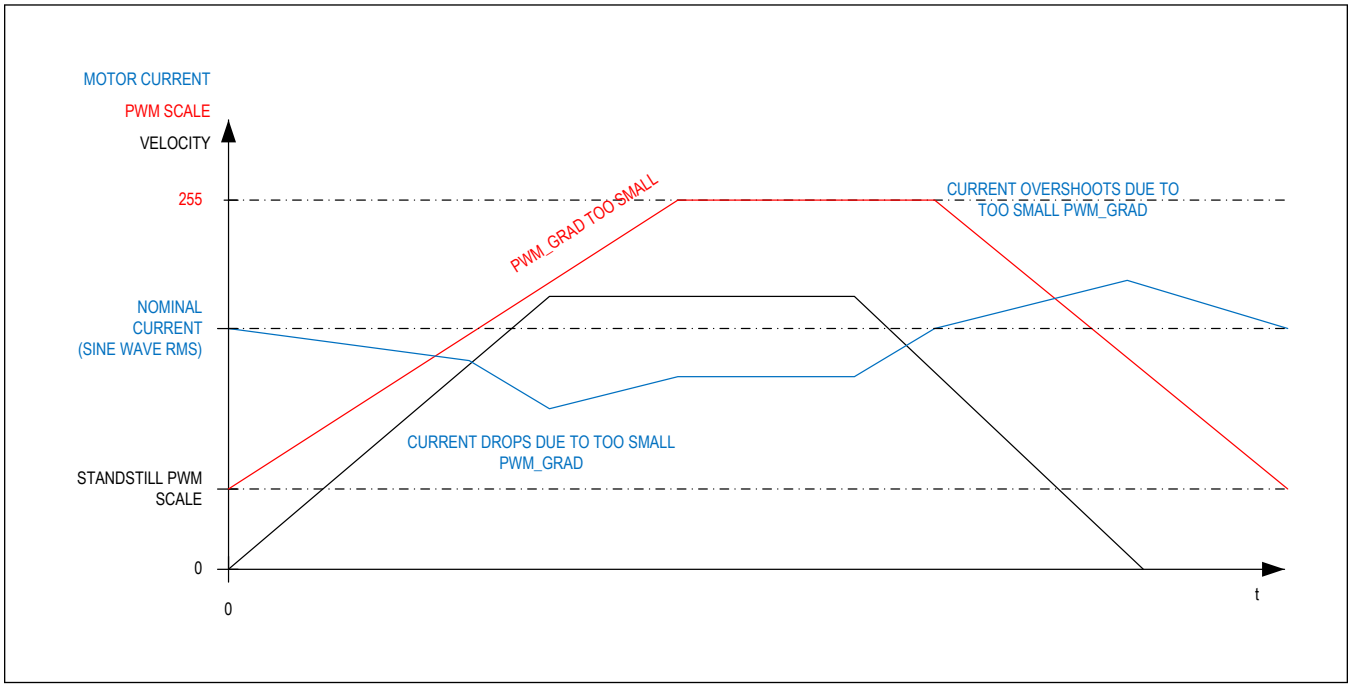


Figure 17. Example of Setting Too Small for PWM_GRAD

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 chopper on phase only (pwm_meas_sd_enable = 0), a minimum chopper duty cycle allowing coil current regulation is given by the blank time as set by TBL and by the chopper frequency setting. Therefore, the motor-specific minimum coil current in StealthChop2 autoscaling mode rises with the supply voltage and with the 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, and IRUN is well within the regulation range. Lower currents (e.g., 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 nonautomatic 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.

f_{PWM} is the chopper frequency as determined by the PWM_FREQ setting.

Example: A motor has a coil resistance of 5Ω, the supply voltage is 16V. With TBL = %01 and PWM_FREQ = %00, t_{BLANK} is 24 clock cycles and f_{PWM} 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 modification of the motor current by the GLOBALSCALER.

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 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 changes of GLOBALSCALER.

The lower current limit also limits the capability of the driver to respond to changes of GLOBALSCALER.

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 by using purely velocity-based feed-forward control. The principle is shown in Figure 18.

Velocity-based scaling is an optional mode and scales the StealthChop2 amplitude based on the time between every two steps, e.g., 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 U_{PWM} (1/SQRT(2) x 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 TMC5271 provides 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:

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

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

With TSTEP representing the microstep frequency for the 256-microstep resolution equivalent and f_{CLK} 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 PWM_GRAD setting can be calculated as follows:

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

Where C_{BEMF} 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, e.g., 51200 = 256 microsteps multiplied by 200 full steps for a 1.8° motor.

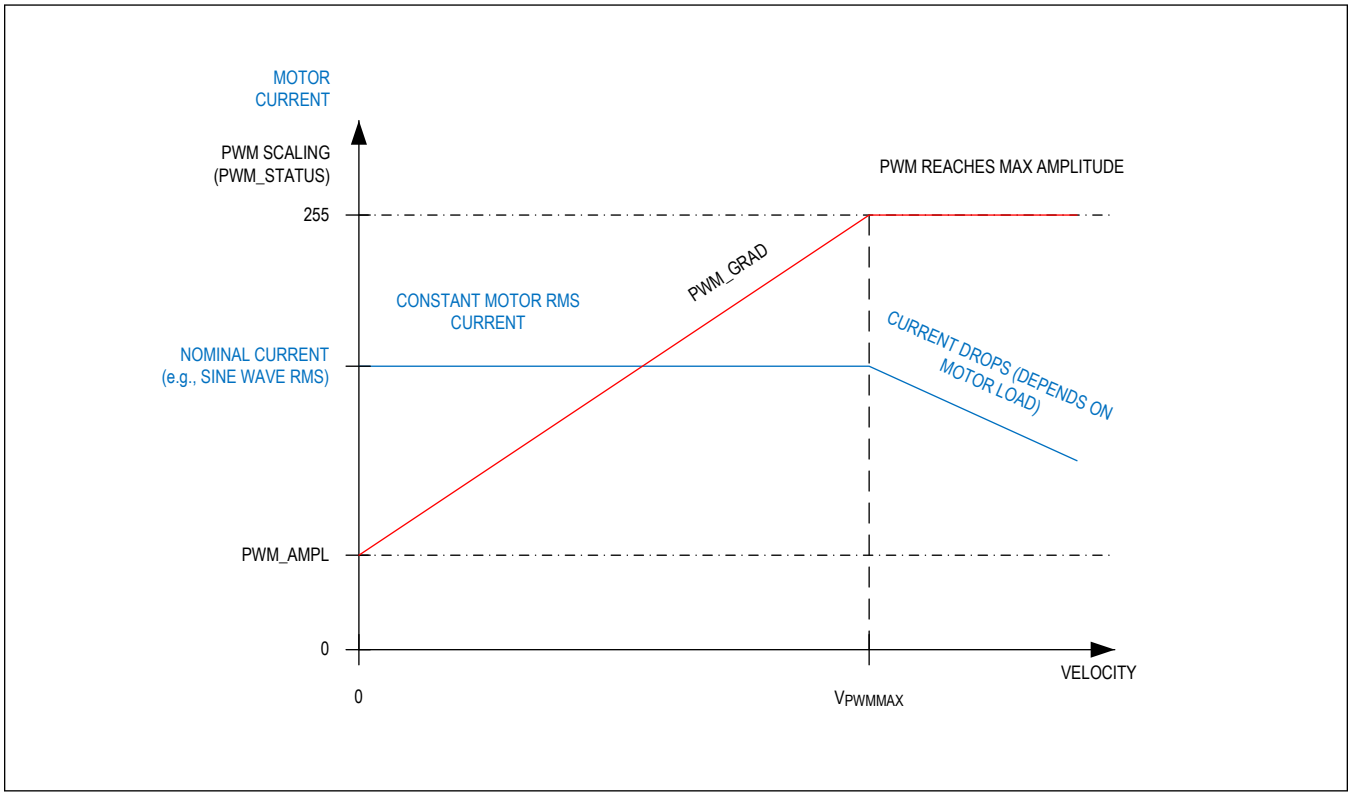


Figure 18. Velocity-Based PWM Scaling (pwm_autoscale = 0)

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 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:

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

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

Holding torque is the motor-specific holding torque, e.g., 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 TMC5271 allows combining StealthChop2 and

SpreadCycle based on a velocity threshold. With this, StealthChop2 is only active at low velocities as shown in Figure 19.

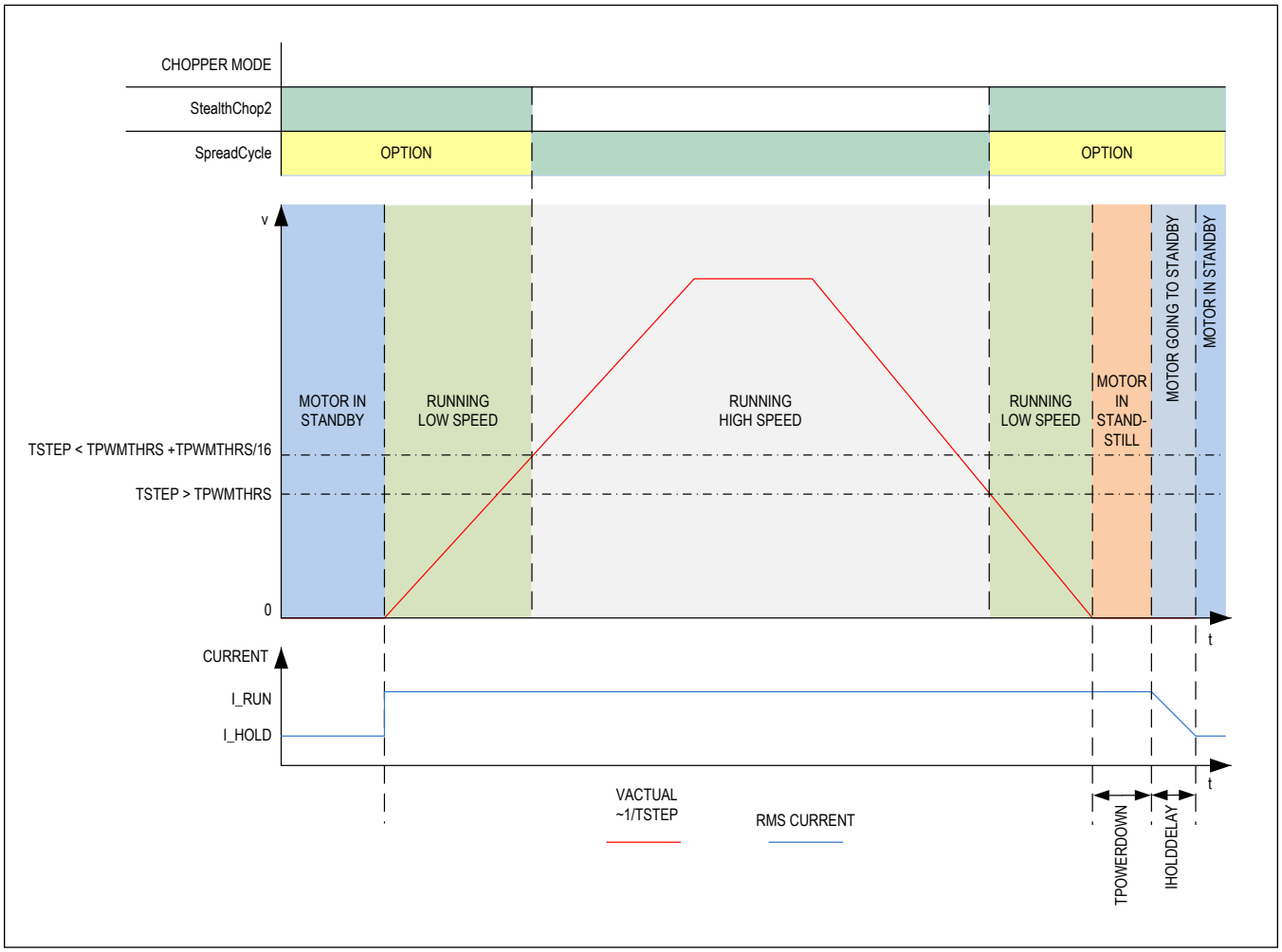


Figure 19. TPWMTHRS for Optional Switching to SpreadCycle

As a first step, both chopper principles StealthChop2 and SpeadCycle should be parameterized and optimized individually (see the *StealthChop2* and *SpreadCycle* 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 (sg_angle_offset), a jerk would occur 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 motor voltage and motor current. Therefore, when switching at higher velocities between voltage PWM and current PWM mode, this jerk would occur with increased intensity. A high jerk may even produce a temporary overcurrent condition (depending on the motor coil resistance). At low velocities (e.g., 1 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 by using StallGuard4

to determine the phase shift. It will apply the same phase shift to SpreadCycle until the velocity falls back below the switching threshold. Set flag SG4_THRS.sg_angle_offset to enable this function.

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 becomes 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 could 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 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 (DRV_CONF.current_range) supporting the motor. This automatically adapts the overcurrent threshold in three steps and thus reduces peak currents when there is a sudden motor stall.

Open Load Flags

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 that the current regulation is reaching the nominal current on both coils.
- If OLA and OLB flags are constant, 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 did 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 delivers the safest result. 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, e.g., 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 FREEWHEEL setting. The desired option becomes enabled after

a time period specified by TPOWERDOWN and IHOLDDELAY. Current regulation becomes frozen once the motor target current is at zero current to ensure a quick startup. 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 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 48.
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 velocity. Default = 0.	0 to 1048575	StealthChop2 is disabled if TSTEP falls under TPWMTHRS.
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.

Table 9. Parameters Controlling StealthChop2 (continued)

PARAMETER	DESCRIPTION	SETTING	NOTES
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 f_{PWM}. Default = 0.	0	f_{PWM} = 2/1024 f_{CLK}.
		1	f_{PWM} = 2/683 f_{CLK}.
		2	f_{PWM} = 2/512 f_{CLK}.
		3	f_{PWM} = 2/410 f_{CLK}.
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	—
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 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. Will regulate close to 0 during tuning.	-255 to 255	Read-only. Scaling value becomes 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.

Table 9. Parameters Controlling StealthChop2 (continued)

PARAMETER	DESCRIPTION	SETTING	NOTES
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	16 t_{CLK}
		1	24 t_{CLK}
		2	36 t_{CLK}
		3	54 t_{CLK}

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 motor coils are controlled using choppers. The choppers work independent of each other. In Figure 20, the different chopper decay phases are shown.

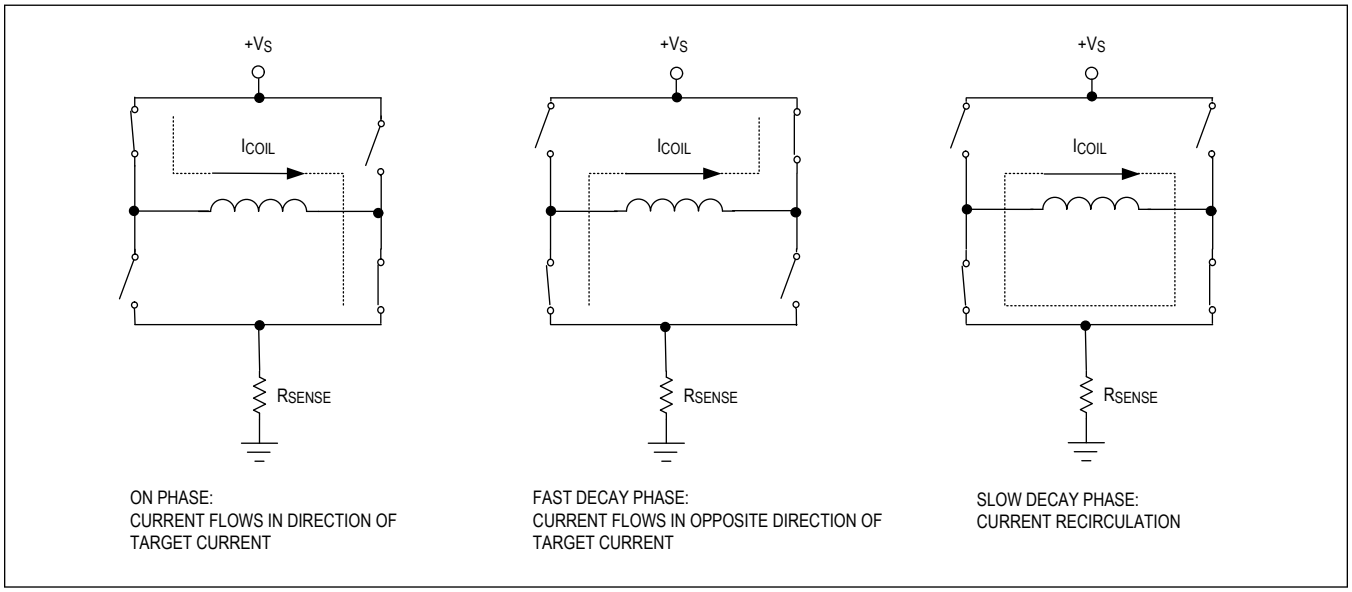


Figure 20. Typical Chopper Decay Phases

Although the current could 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 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 that is too low might generate audible noise. A higher frequency reduces current ripple in the motor, but with a frequency that is 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 needs to 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 N_{CLK} = 24 + 32 x TOFF (1 works with minimum blank time TBL of 28 clocks)
TBL	Selects the comparator blank time. This time needs to safely cover the switching event and the duration of the ringing. For most applications, a setting of 1 or 2 is appropriate. For highly capacitive loads, e.g., when filter networks are used, a setting of 2 or 3 is required. Default = 2.	0	20 t_{CLK} Restriction: Use this setting only in combination with an external clock up to 10MHz
		1	28 t_{CLK}
		2	36 t_{CLK}
		3	54 t_{CLK}.
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%–70% of the chopper cycle in standstill and are important for low motor and driver power dissipation.

The following is an example of the calculation of a starting value for the slow-decay time TOFF:

- Target chopper frequency: 25kHz
  - t_{OFF} = 1/25kHz x 50/100 x 1/2 = 10µs
  - Assumption: Two slow-decay cycles make up for 50% of overall chopper cycle time.
- For the TOFF setting, this means: TOFF = (t_{OFF} x f_{CLK} - 12)/32
- With 12MHz clock, this results in TOFF = 3.4, which would require a setting of TOFF = 3 or 4.
- With 16MHz clock, this results in TOFF = 4.6, which would require a setting of TOFF = 4 or 5.

Tip: The highest motor velocities sometimes benefit from setting TOFF to 1 or 2 and a short TBL setting.

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 which is caused by resistive losses in the motor to give best microstepping results. This allows the chopper to precisely regulate the current for both 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 the 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 (e.g., 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 half waves in case the hysteresis setting is too small. At medium velocities (e.g., 100 full steps to 400 full steps per second), a hysteresis setting that is too low leads to increased humming and vibration of the motor. A hysteresis setting that is 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 moto. A setting that is too low results in reduced microstep accuracy, while a setting that is too high leads to more chopper noise and motor power dissipation. When the fast-decay time becomes 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 could in some cases lead to the chopper frequency becoming too low e.g., when the coil resistance is high when 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 settings, by decrementing the hysteresis value stepwise each 16 system clocks. At the beginning of each chopper cycle, the hysteresis begins with a value which is the sum of the start and the 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. The related hysteresis parameters are listed in Table 11.

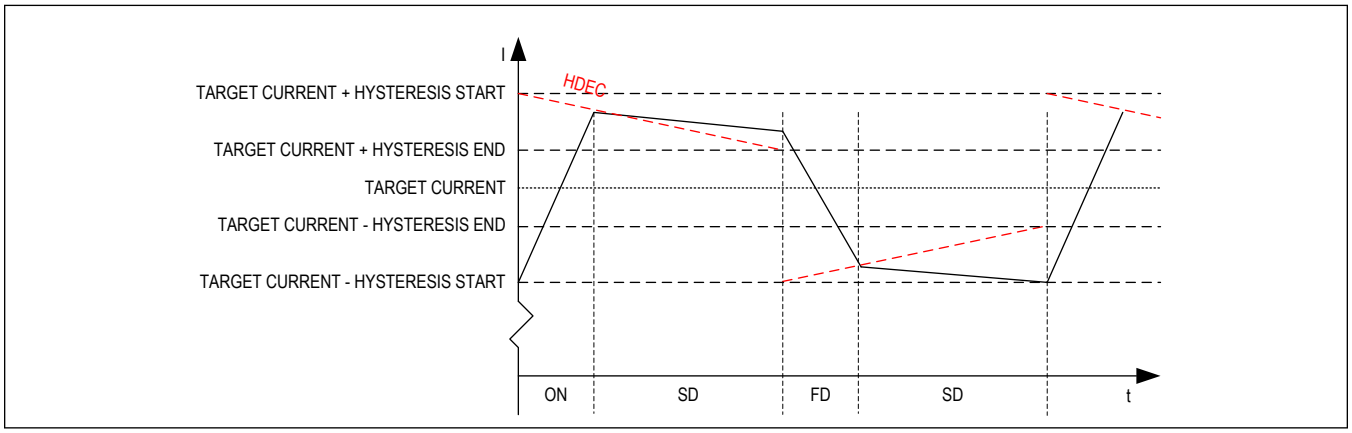


Figure 21. SpreadCycle Chopper Scheme Showing Coil Current During a Chopper Cycle

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. Default = 5.	0 to 7	HSTRT = 1 to 8. This value adds to HEND.
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 max. 30 (amplitude reduced to 240), the sum is not limited. Default = 2.	0 to 2	-3 to -1: Negative HEND.
		3	0: Zero HEND.
		4 to 15	1 to 12: Positive HEND.

Even at HSTRT = 0 and HEND = 0, the TMC5271 sets a minimum hysteresis through analog circuitry.

The following example shows minimum hysteresis:

A hysteresis of 4 has been chosen. 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, e.g., 1: 3 + 1 = 4).

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

HEND = 0 (sets an effective end value of -3).

HSTRT = 6 (sets an effective start value of hysteresis end +7: 7 - 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 needs to 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. The basic principle is shown in Figure 22. This can be tuned using an oscilloscope or evaluating motor smoothness at different velocities. A good starting value is a fast-decay time setting similar to the slow-decay time setting.

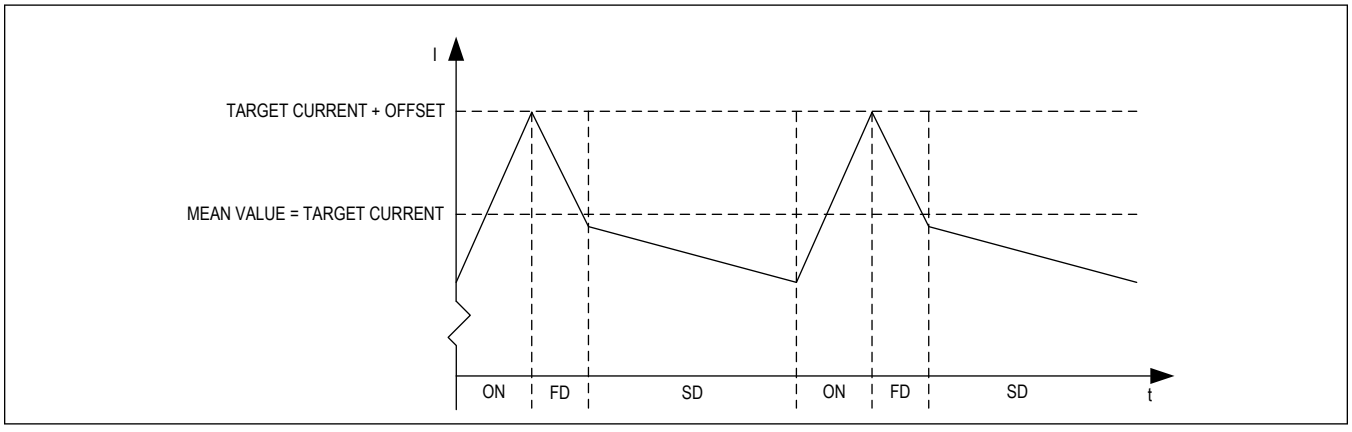


Figure 22. Classic Constant Off-Time Chopper with Offset Showing Coil Current

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 smoothest operation. Table 12 shows the related parameters for this mode.

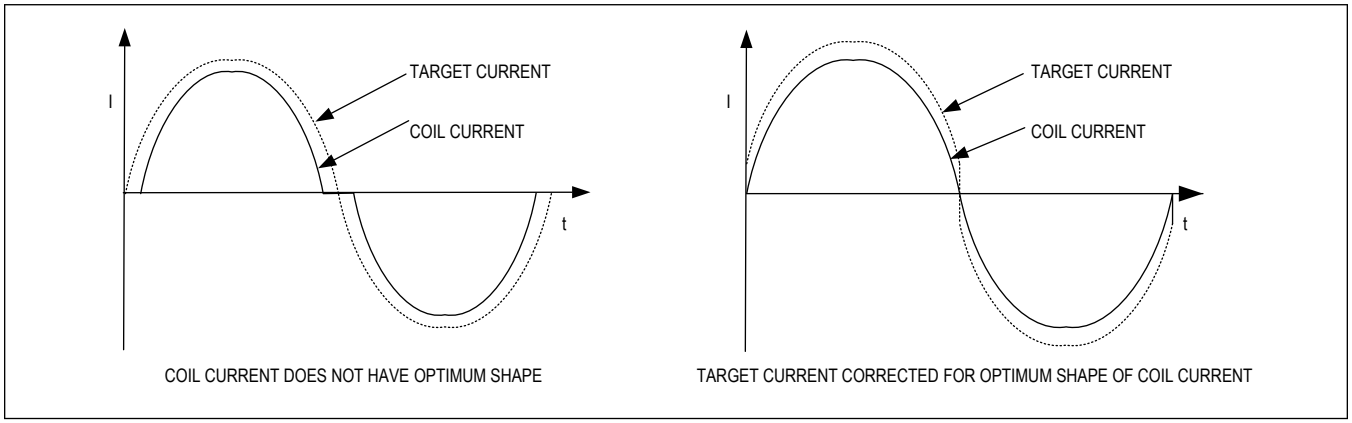


Figure 23. Zero Crossing with Classic Chopper and Correction Using Sine-Wave Offset

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

Integrated Current Sensing (ICS)

Nondissipative current sensing is integrated in the TMC5271. 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 individual MOSFET temperature to yield the best results.

Setting the Motor Current

Table 13 shows the parameters related to the motor current setting in TMC5271.

Table 13. 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 microsteps coarser. This setting also controls the maximum current value set by CoolStep. Default = 31.	0 to 31	Scaling factor 1/32, 2/32, to 32/32.
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 x 15 x 2^{18} clock cycles, e.g., approximately one second at 16MHz. Default = 1.	0	Instant power-down to IHOLD.
		1 to 15	1 x 2^{18} to 15 x 2^{18} clocks per current decrement.
IRUNDELAY	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.	0	Instant power-up to IRUN when position has been reached.
		1 to 15	Delay per current increment step in multiple of IRUNDELAY x 512 clocks.
TPOWERDOWN	TPOWERDOWN sets the delay time after standstill (stst) of the motor to motor current power-down. Time range is about 0 to 4 seconds. **Note:** A minimum setting of 2 is required to allow automatic tuning of StealthChop2 PWM_OFFS_AUTO Default = 10.	0 to 255	0 to ((2^8) - 1) x 2^{18} t_{CLK}.
FSR_IREF	Scales the reference current I_{REF} (based on R_{REF}). This is like scaling the external resistance R_{REF}. Use this together with FSR_Mx for finetuning the current scaling.	0 to 3	0: 25% I_{REF}. 1: 50% I_{REF}. 2: 75% I_{REF}. 3: 100% I_{REF}.

Table 13. Parameters Controlling the Motor Current (continued)

PARAMETER	DESCRIPTION	SETTING	NOTES
FSR	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 over current protection threshold (this value is the peak current setting).	0 to 3	The $R_{DS(ON)}$ and overcurrent protection threshold values are given in the *EC Characteristics* section under output specifications and protection circuits.
GLOBALSCALER_A	Global scaling of motor current. This value is multiplied to the current scaling to finetune and adapt to a certain motor. Scales Phase A. **Tip:** Values > 128 recommended for best results. This value should be chosen before tuning other settings because it also influences chopper hysteresis. Differing values can only be used when using only 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 finetune and adapt to a certain motor. Scales Phase B. **Tip:** Values > 128 recommended for best results. This value should be chosen before tuning other settings because it also influences chopper hysteresis. Differing values can only be used when using only 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.

Setting the Full-Scale Current

The full-scale current I_{FS} is a maximum absolute current level setting and is defined by the external reference resistor R_{REF} and two parameters in the DRV_CONF register. Adapting I_{FS} 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 with the same external resistor R_{REF}.

A standard low-power resistor with 1% accuracy is sufficient for R_{REF}. The external resistor R_{REF} can range between 10kΩ and 60kΩ.

Therefore, connect a resistor from I_{REF} to GND to set the full-scale chopping current I_{FS}.

The two 2-bit parameters FSR and FSR_IREF in the DRV_CONF register define the typical on-resistance of the driver stage (FSR) and further scale I_{REF} (FSR_IREF) to control the full-scale range based on R_{REF}.

FSR_IREF allows for four different settings to define the KIFS_IREF scaling factor as given in Table 14.

Table 14. KIFS_IREF Scaling Factor based on FSR_IREF Setting

FSR_IREF BITS 1:0	KIFS_IREF
11	1.00
10	0.75
01	0.50
00 (default)	0.25

The equations below shows the full-scale current I_{FS} as a function of the R_{REF} resistor and the DRV_CONF parameter settings.

The proportionality constant K_{IFS} depends on the selected full-scale range setting FSR.

The I_{REF} scaling factor depends on FSR_IREF.

$$I_{FS}(RMS) = \frac{\left(K_{IFS}(KV) \times KIFS\_IREF\right)}{R_{REF}(k\Omega)}$$

$$I_{FS}(PEAK) = \frac{\left(K_{IFS}(KV) \times KIFS\_IREF\right)}{R_{REF}(k\Omega)} \times \sqrt{2}$$

Using the two equations Table 15 and Table 16 show the values for the full-scale range for for R_{REF} = 10kΩ and for R_{REF} = 36kΩ.

Notes:

- Some of the combinations of FSR and FSR_IREF result in similar full-scale ranges. It is always better to choose FSR_IREF as large as possible for best current control accuracy.
- For optimal performance of the current regulation and proper current levels, the TMC5271 requires a symmetrical board layout.

Table 15. I_{FS} Full-Scale Range Settings (Standard R_{REF} = 10kΩ)

REGISTER CONFIGURATION		KIFS_IREF	K_{IFS} (A x kΩ)	I_{FS} (A RMS)	I_{FS} (A PEAK)	TYPICAL $R_{DS(ON)}$ (LS) (Ω)
FSR BITS 1:0	FSR_IREF BITS 1:0					
11	11	1.0	16	1.60	2.26	0.055
	10	0.75		1.20	1.69	
	01	0.5		0.80	1.13	
	00 (default)	0.25		0.40	0.56	
10	11	1.0	12.06	1.21	1.70	0.070
	10	0.75		0.9	1.28	
	01	0.5		0.60	0.85	
	00 (default)	0.25		0.30	0.43	
01	11	1.0	8.16	0.82	1.15	0.100
	10	0.75		0.61	0.86	
	01	0.5		0.41	0.58	
	00 (default)	0.25		0.20	0.29	
00 (default)	11	1.0	4.1	0.41	0.58	0.185
	10	0.75		0.31	0.43	
	01	0.5		0.21	0.29	
	00 (default)	0.25		0.10	0.14	

Table 16. I_{FS} Full-Scale Range Settings (Example for R_{REF} = 36kΩ)

REGISTER CONFIGURATION		KIFS_IREF	K_{IFS} (A x kΩ)	I_{FS} (A RMS)	I_{FS} (A PEAK)	TYPICAL $R_{DS(ON)}$ (LS) (Ω)
FSR BITS 1:0	FSR_REF BITS 1:0					
11	11	1.0	16	0.444	0.628	0.055
	10	0.75		0.333	0.471	
	01	0.5		0.222	0.314	
	00 (default)	0.25		0.111	0.157	
10	11	1.0	12.06	0.335	0.474	0.070
	10	0.75		0.251	0.355	
	01	0.5		0.168	0.237	

Table 16. I_{FS} Full-Scale Range Settings (Example for R_{REF} = 36kΩ) (continued)

	00 (default)	0.25		0.084	0.118	
01	11	1.0	8.16	0.227	0.321	0.100
	10	0.75		0.170	0.240	
	01	0.5		0.113	0.160	
	00 (default)	0.25		0.057	0.080	
00 (default)	11	1.0	4.1	0.114	0.161	0.185
	10	0.75		0.085	0.120	
	01	0.5		0.057	0.080	
	00 (default)	0.25		0.028	0.040	

Velocity-Based Mode Control

The TMC5271 allows 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.

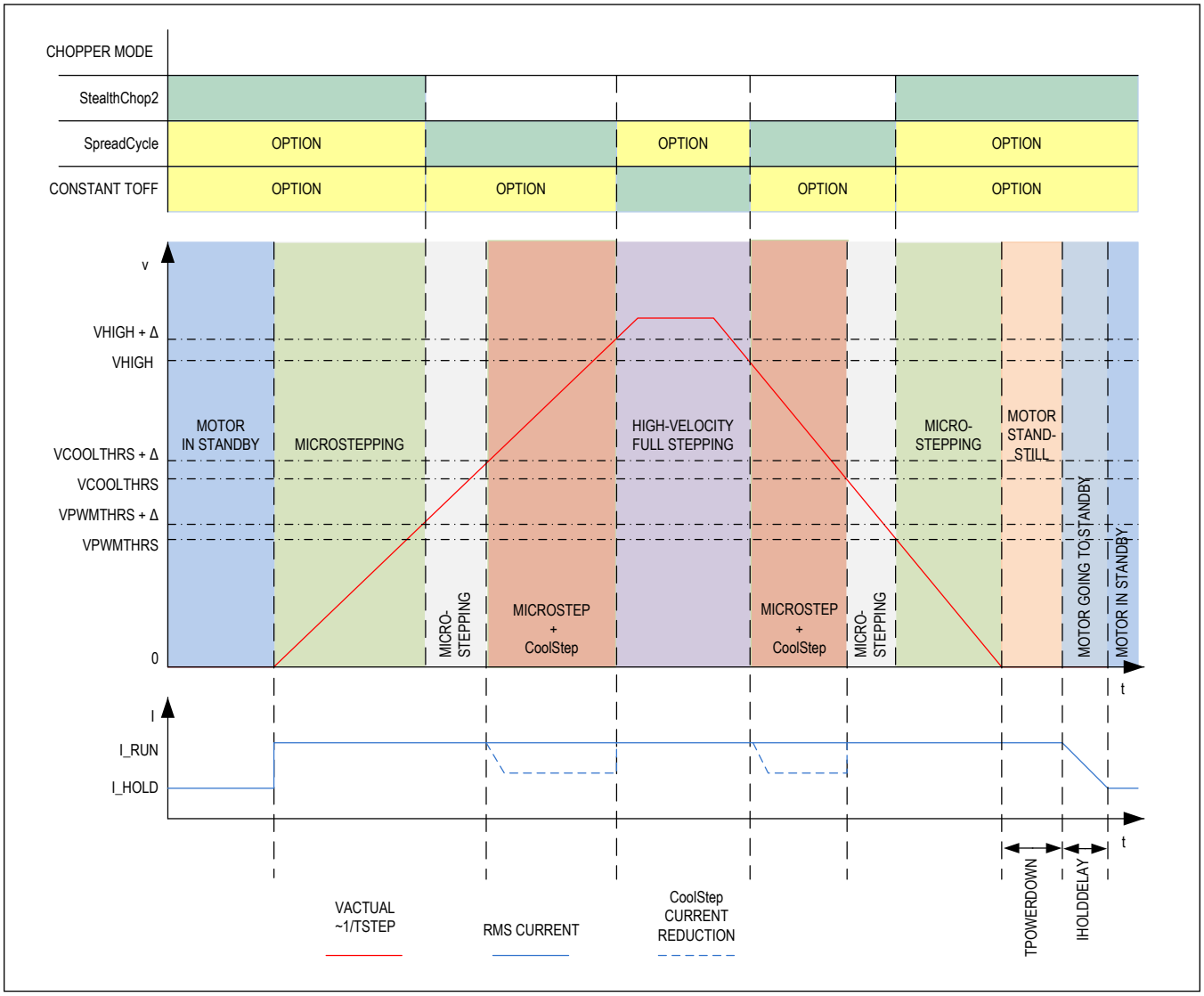


Figure 24. Choice of Velocity-Dependent Modes

Figure 24 shows all 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 becomes 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 becomes 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 17. Velocity-Based Mode Control Parameters

PARAMETER	DESCRIPTION	SETTING	NOTES
stst	Indicates motor standstill in each operation mode. Time is 2^{20} clocks after the last step pulse. 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 to 4 seconds (with f_{CLK} = 16MHz). 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} x t_{CLK}.
TSTEP	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. Default/reset: 0.	0 to 1048575	Status register, read-only. Actual measured step time in multiple of t_{CLK}.
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 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 becomes 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 x 15/16) - 1 or (TSTEP x 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	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.	0	No switch to full step.
		1	Full step at high velocities.

Table 17. Velocity-Based Mode Control Parameters (continued)

PARAMETER	DESCRIPTION	SETTING	NOTES
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 becomes 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 target position or target velocity. It automatically calculates the motion profile taking into account acceleration and velocity settings. The TMC5271 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 phase and for the deceleration phase to allow for jerk-minimized ramps.

Real Word Unit Conversion

The TMC5271 uses its internal or external clock signal as a time reference for all internal operations. Thus, all time, velocity, and acceleration settings are referenced to f_{CLK}. Table 18 shows all ramp generator parameters and their units and their dependency on f_{CLK}. 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[TMC5271] and a[TMC5271] are internal units of the TMC5271. These are the values that need to be written to the velocity/acceleration registers of the TMC5271. Calculator tools and evaluation tools are available from the product website.

Table 18. Ramp Generator Parameters vs. Units

PARAMETER/ SYMBOL	UNITS	DESCRIPTION
f_{CLK}	Hz	Clock frequency of the TMC5271.
s	s	Second.
µS	Microstep	—
FS	Full step	—
USC—microstep count	—	Microstep resolution in number of microsteps (i.e., the number of microsteps between two full steps, normally 256).
FSC—full-step count	—	Motor full steps per rotation e.g., 200.
µstep velocity v[Hz]	microsteps/s	v[Hz] = v[TMC5271] x (f_{CLK}[Hz]/ 2^{24}).
µstep acceleration a[Hz/s]	microsteps/s^2	a[Hz/s] = a[TMC5271] x f_{CLK}[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.

Table 18. Ramp Generator Parameters vs. Units (continued)

PARAMETER/ SYMBOL	UNITS	DESCRIPTION
Ramp steps[microsteps] = rs	microsteps	rs = (v[TMC5271])^2/a[TMC5271]/2^8 microsteps during linear acceleration ramp (assuming acceleration from 0 to V).
TSTEP, Txxx_THRS	—	TSTEP = $f_{CLK}/f_{256STEP}$ = $f_{CLK}/(f_{STEP}$ x 256/USC) = 2^{24}/(VACTUAL x 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, e.g., 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 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 becomes clear that deceleration capability of the system is quicker than acceleration capability. Thus, deceleration values can be set higher in many applications. This allows the operation speed of the motor in time-critical applications is 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 by the user. As a result, it is possible that the motion becomes 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 is flagged by the second_move flag.

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

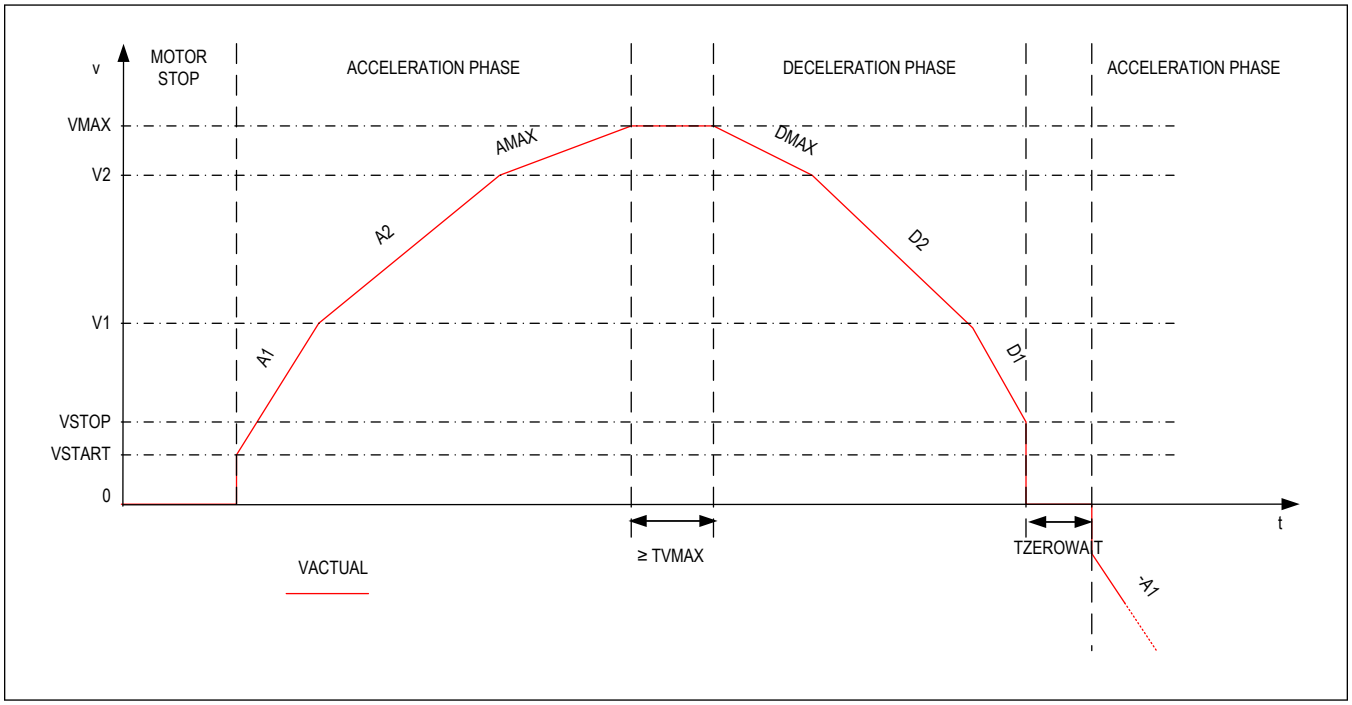


Figure 25. Ramp-Generator Velocity Trace Showing Second Move into Negative Direction

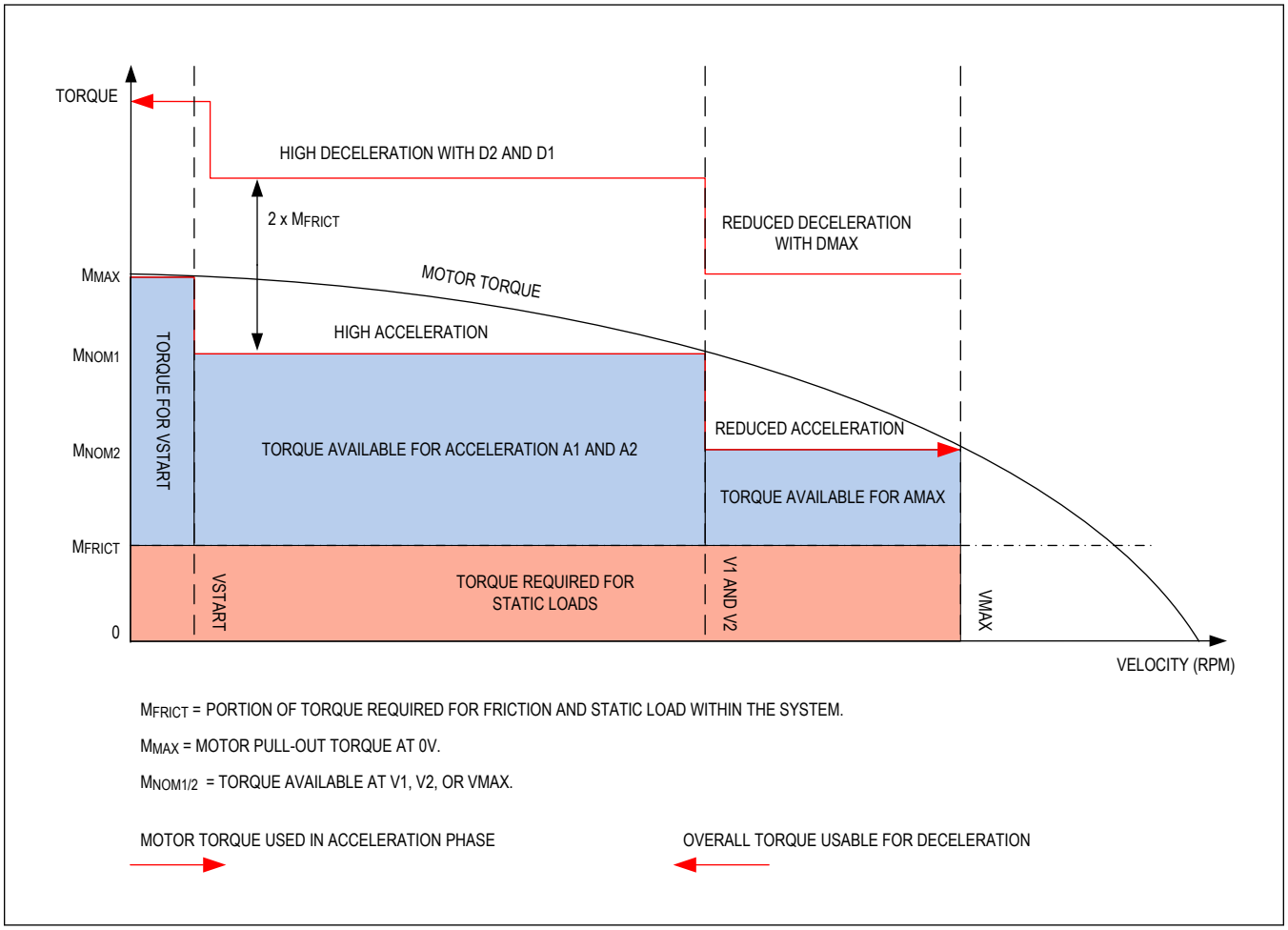


Figure 26. Optimized Motor Torque Usage with the Ramp Generator

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 by 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 becomes clear that a subsequent move into the opposite direction would provide 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 becomes active.

The set of three acceleration and deceleration segments can be used in two ways; either for adaptation to the motor torque curve by using higher acceleration values at lower velocity or to reduce the jerk (change of acceleration) when transitioning from one acceleration segment to the next. For jerk-optimized ramps, typically 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 8-point motion profile generator allows the user 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 different position distances for a pseudo-S-shaped configuration.

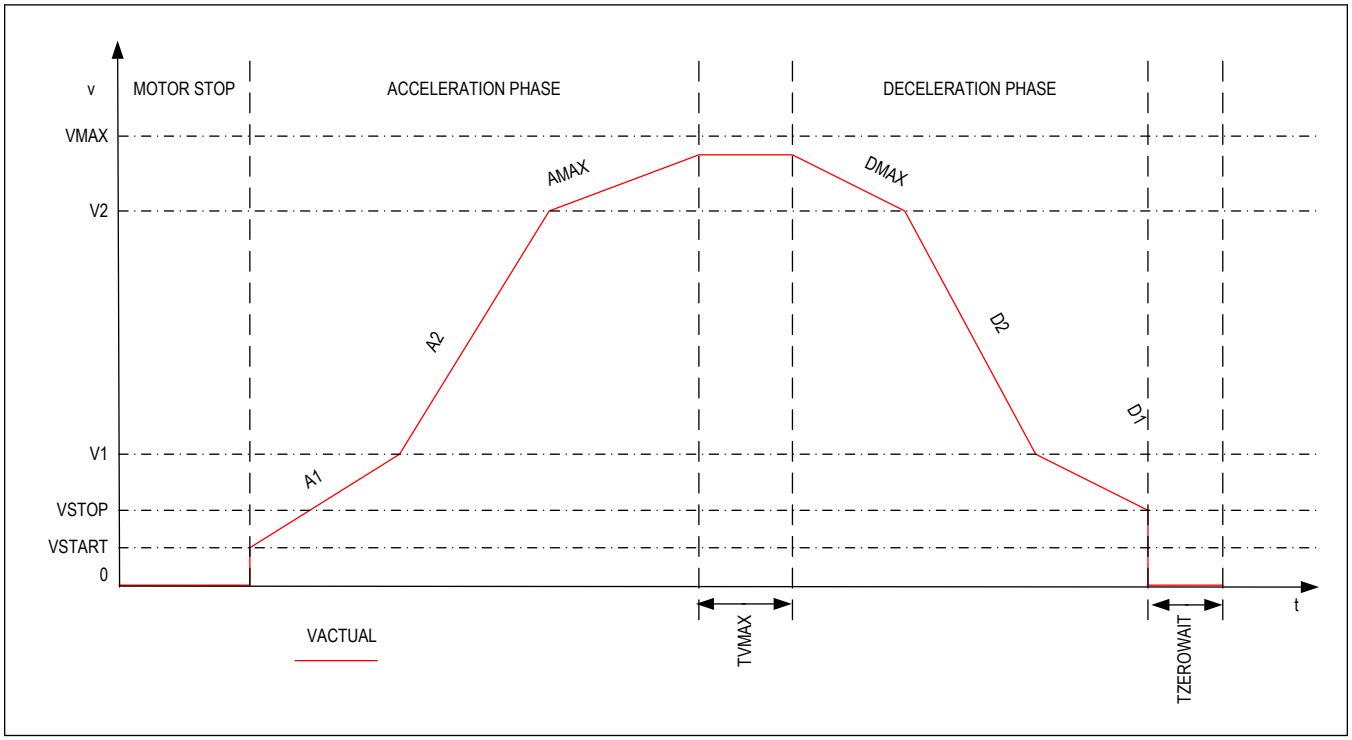


Figure 27. 8-Point Ramp with VMAX Not Reached Due to Distance Too Low

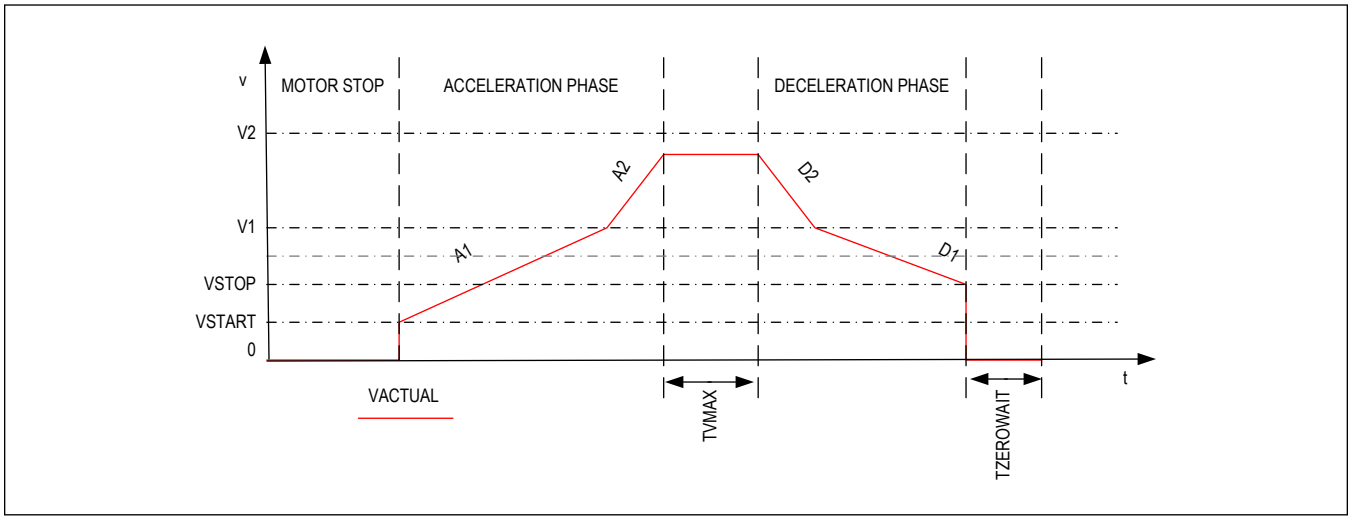


Figure 28. V2 Not Reached and No AMAX and DMAX Phase Due to Low Distance

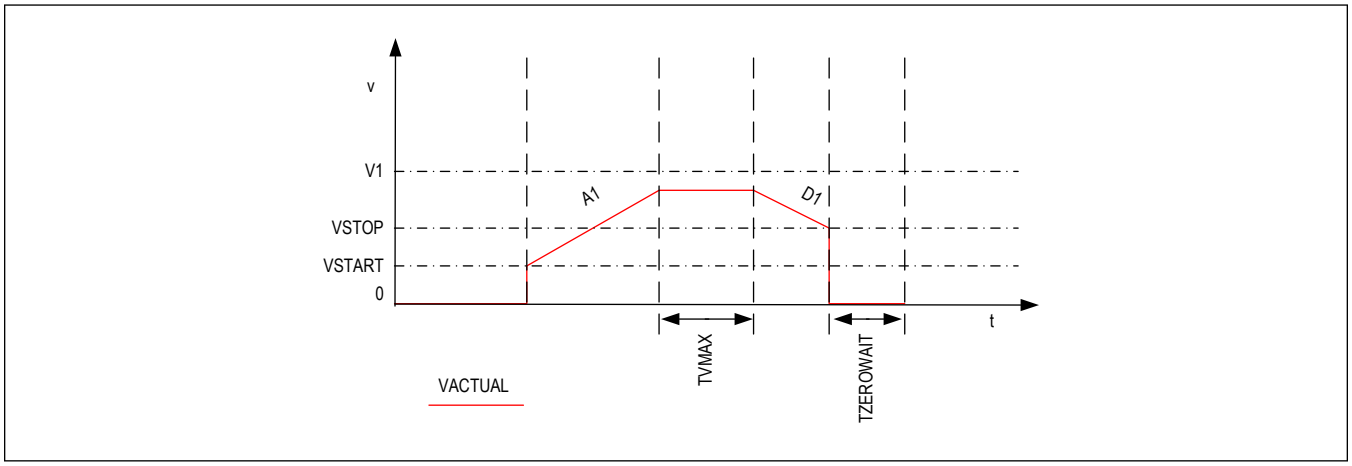


Figure 29. V1 Not Reached Due to Low Distance

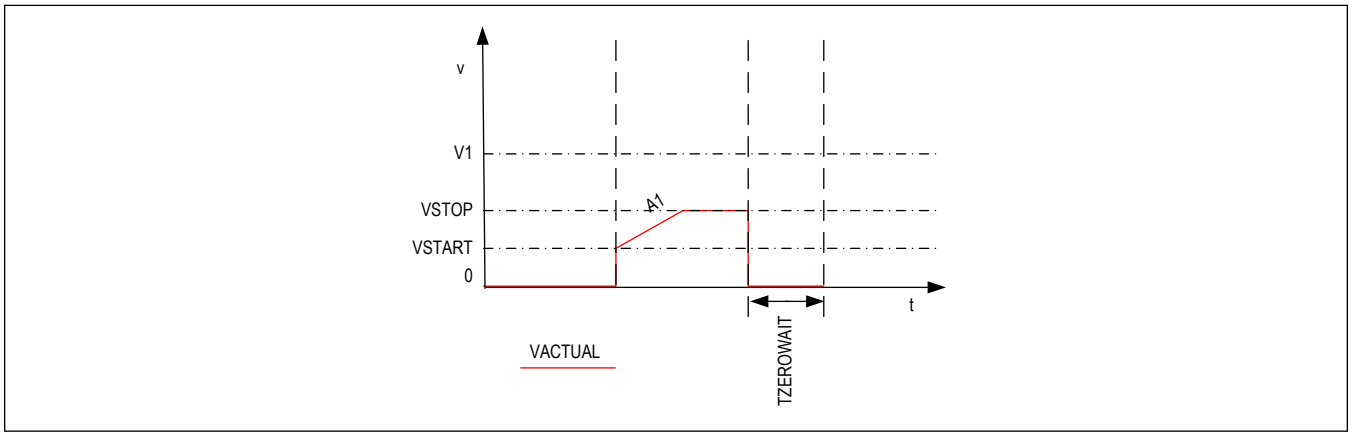


Figure 30. TVMAX Not Kept Due to Low Distance

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

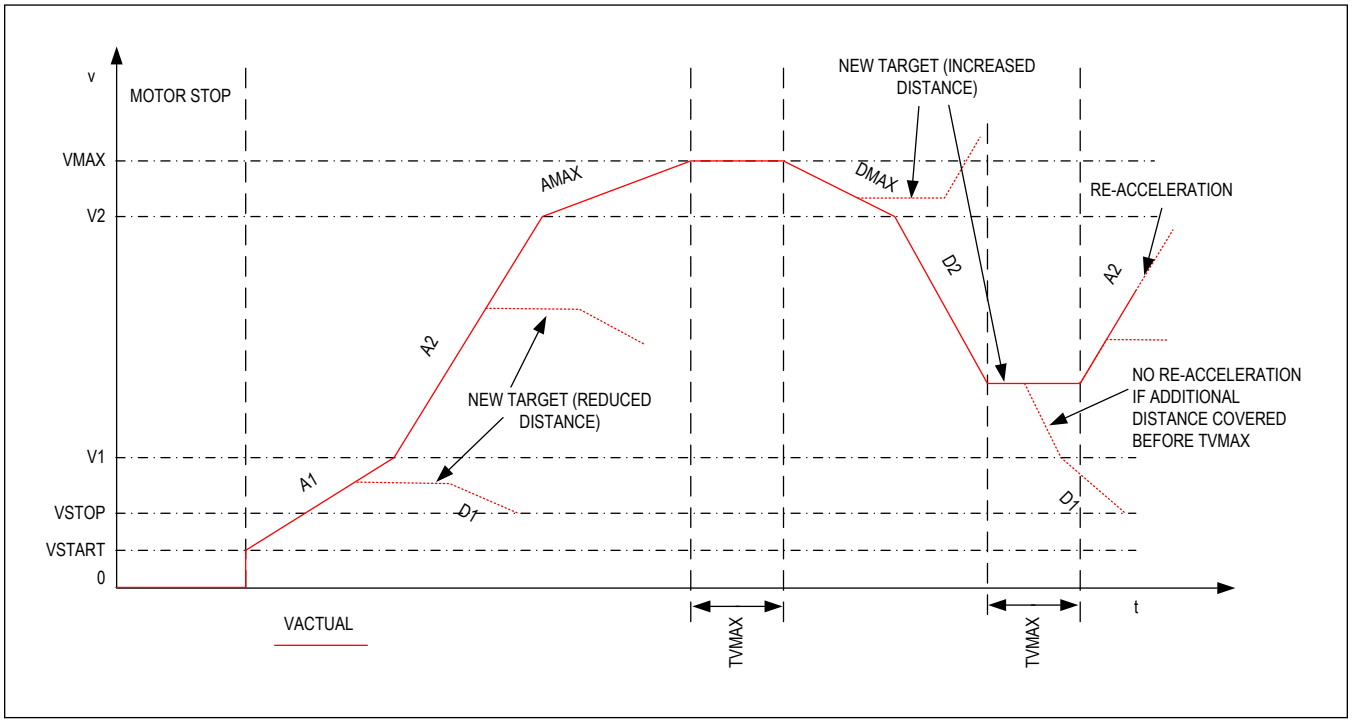


Figure 31. 8-Point Ramp Examples with On-the-Fly Target Position Change

Velocity Mode

For ease of use, velocity-mode movements do not use the different acceleration and deceleration settings. For velocity mode, only AMAX and VMAX are relevant. The ramp generator always uses AMAX to accelerate or decelerate to VMAX in this mode.

To decelerate the motor to standstill, it is sufficient to set VMAX to zero. The vzero flag signals standstill of the motor. The velocity_reached flag always signals that the target velocity has been 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 has been reached.

The options to terminate motion using 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 TZEROWAIT time using option 1 or 2.
4. Activate a stop switch. This can be done by means of the hardware input, e.g., 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 that the valid motion range is never left.

For joystick control, follow these steps:

1. 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).
2. Modify VMAX depending on the joystick input at any time in the range of VSTART to the maximum value. With VSTART = 0, 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 velocity for ramping up and ramping down.
3. In case the acceleration settings are not modified, it is not necessary to rewrite XTARGET; simply modify VMAX.
4. 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 a number of 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).

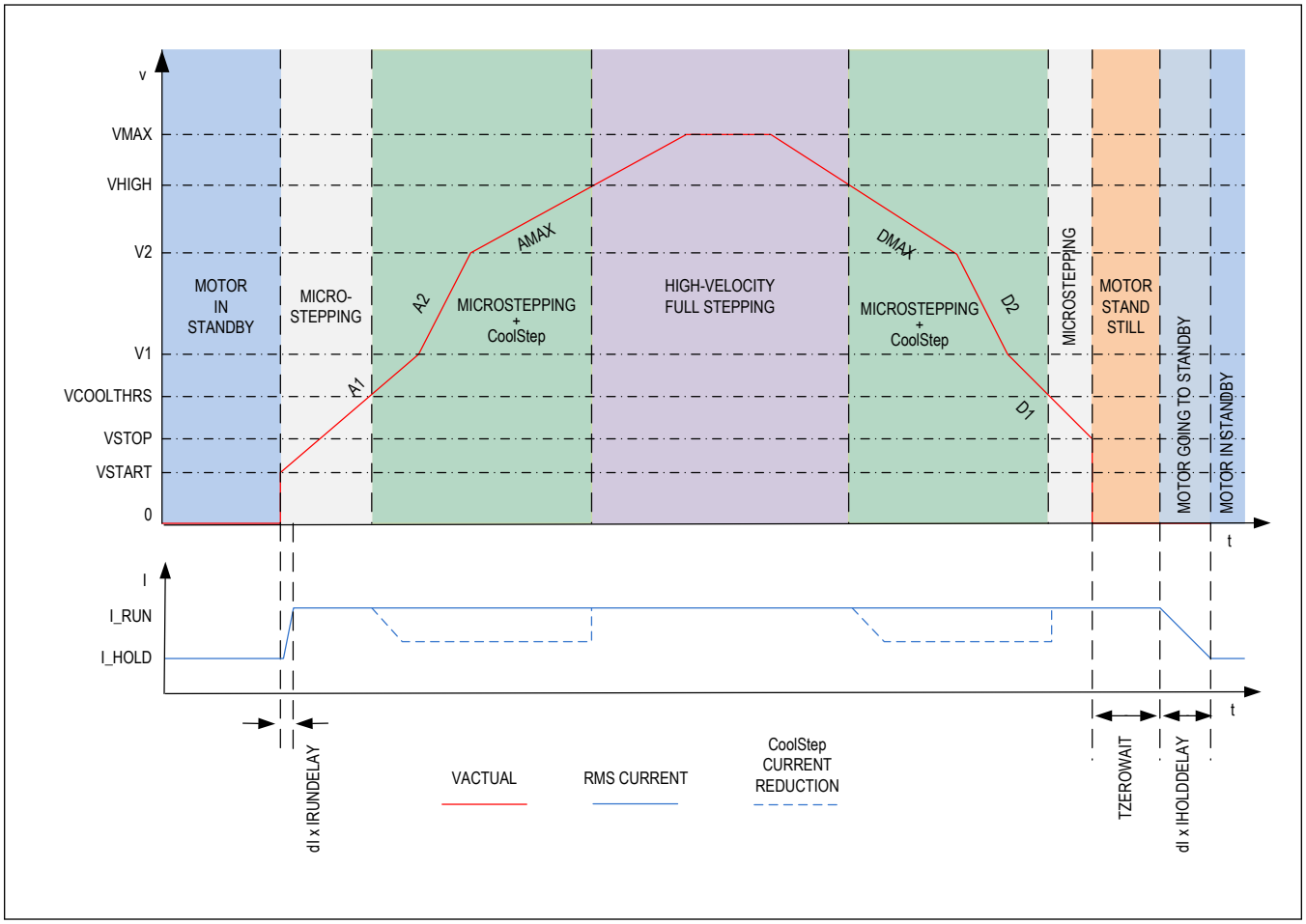


Figure 32. Ramp-Generator, Velocity-Dependent Motor Control

Reference Switches

Prior to 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 a 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 the right reference switch. 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:

1. It can be stopped abruptly using a switch. This is useful in an emergency and for StallGuard2/4-based homing.
2. The motor can be softly decelerated to zero with the deceleration settings (DMAX, V2, D2, V1, and D1) using the soft-stop 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.

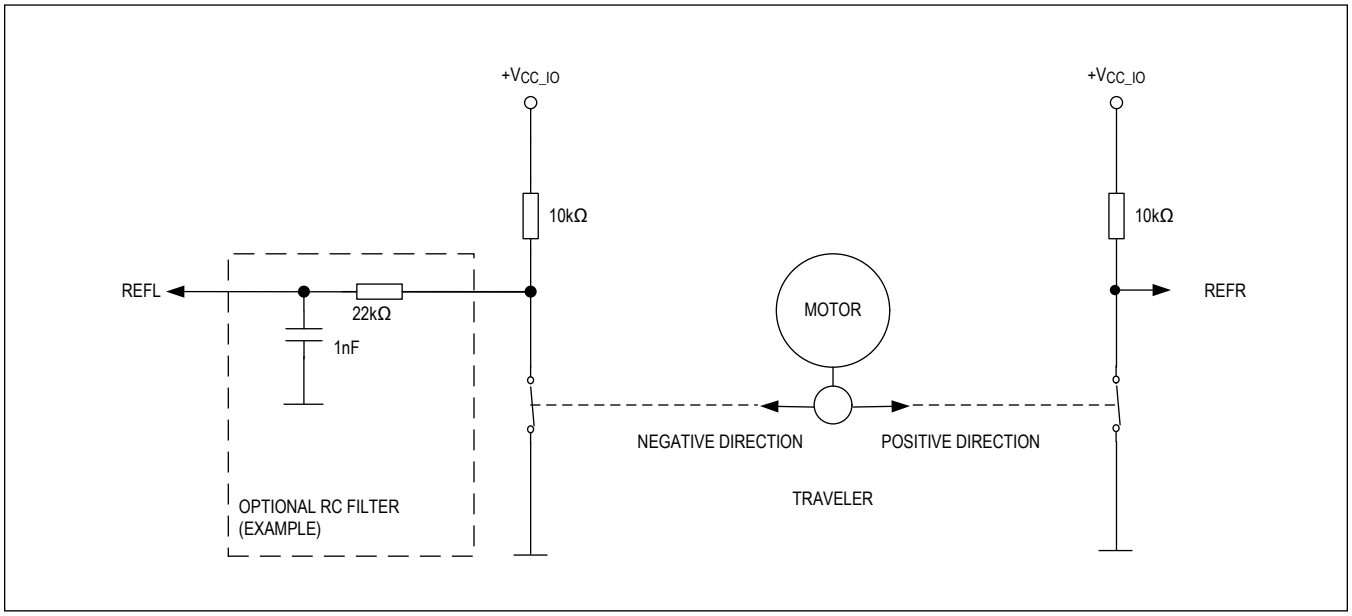


Figure 33. Using Reference Switches (Example)

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 which 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 TMC5271 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 which should be considered with respect to the application.

To implement a homing procedure, do the following:

1. Make sure that the home switch is not pressed, e.g., by moving away from the switch.
2. Activate position latching upon the desired switch event and activate motor (soft) stop upon active switch. StallGuard2/4-based homing requires using a hard stop (en_softstop = 0).
3. Start a motion ramp into the direction of the switch. (Move to a more negative position for a left switch and to a more positive position for a right switch). This motion can be timed out by using a position ramping command.
4. As soon as the switch is hit, the position becomes 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.
5. Switch the ramp generator to hold mode and calculate the difference between the latched position and the actual position. For StallGuard2/4-based homing or when using hard stop, XACTUAL stops exactly at the home position so there is no difference (0).
6. 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 StallGuard2/4 was used for homing, read and write back RAMP_STAT to clear the StallGuard2/4 stop event event_stop_sg and release the motor from the stop condition.

For homing with a third switch, proceed as follows.

Some applications use an additional home switch, which operates independently of the mechanical limit switches. The encoder functionality of the TMC5271 provides an additional source for position latching. It allows using the N channel input to snapshot XACTUAL with a rising or falling edge event, or both. This function also provides an interrupt output.

1. Activate the latching function (ENCMODE: Set ignoreAB, clr_cont, neg_edge or pos_edge and latch_x_act). The latching function can then trigger the interrupt output (check by reading n_event in ENC_STATUS when an interrupt is signaled at DIAG0).
2. Move to the direction where the N channel switch should be. If the motor hits a stop switch (REFL or REFR) before the home switch is detected, reverse the motion direction.
3. Read out XLATCH once the switch has been triggered. It gives the position of the switch event.
4. After detection of the switch event, stop the motor, and subtract XLATCH from the actual position. (A detailed description of the required steps is in the homing procedure above.)

Virtual Reference Switches

The TMC5271 supports virtual reference switches to support applications that only have a single or no reference switch (StallGuard2/4 homing) to safely limit the physical motion range. This is illustrated in Figure 34. The virtual stop switches become 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.

Optionally, the virtual stops can be switched to monitor the encoder position (X_ENC). To do this, set virtual_stop_enc to 1.

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 was used.

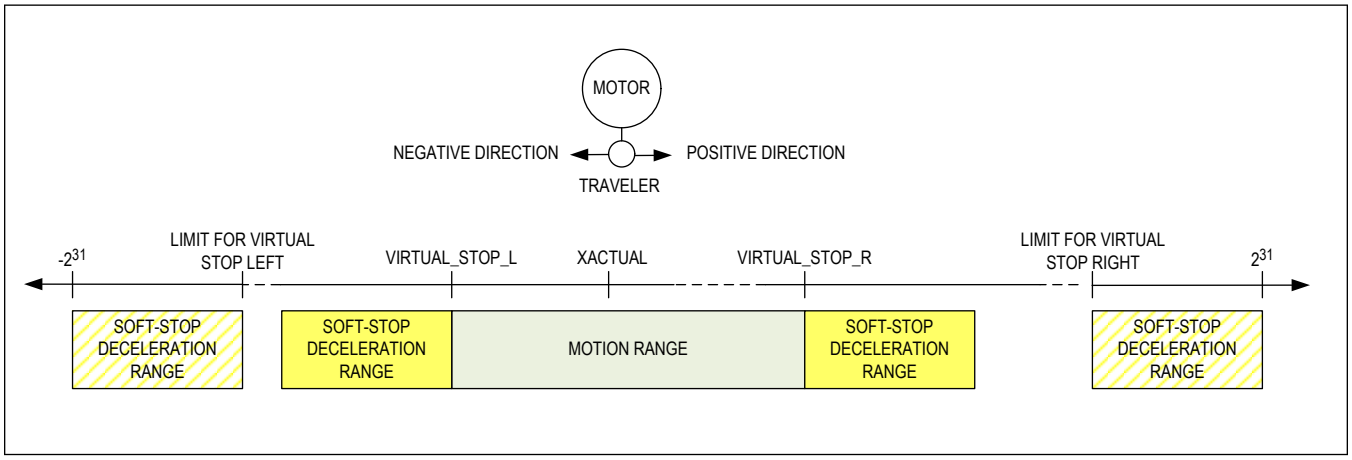


Figure 34. Virtual Stop Switches and Limit Visualization

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, e.g., 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 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 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, i.e., 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. The options are listed in Table 19.

Table 19. 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 2^{24} - 1 microsteps.

Whenever the position-compare condition XACTUAL = X_COMPARE goes from a true to a false state, X_COMPARE becomes 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 would not be 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 would also trigger the repetition mechanism.

Closed-Loop Position Control with Encoder

This function is a simple closed-loop position regulation based on the encoder position feedback. It is a P-regulator. This function is enabled by setting the proportional factor P > 0. With P = 0 this function is disabled. Also, the motion controller must be enabled and not in step/direction mode (GCONF[13] SD = 0).

The error is calculated as the deviation between X_ACTUAL and X_ENC:

error = X_ACTUAL − X_ENC

The tolerance parameter relates to the error. Errors below the tolerance value are ignored. The limit of the P-regulator output is set by the VMAX parameter.

The en_tol_on_pos_reached bit controls the behavior of the P-regulator when the position-reached flag, pos_reached in the RAMP_STAT register, is set.

When set to 0, the P-regulator is always active. Relevant parameters are located in the POSITION_P_CTRL register (0x2D) and given in Table 20.

Table 20. Closed-Loop Position Control Related Parameters

PARAMETER	DESCRIPTION	RANGE	NOTES
en_tol_on_pos_reached	Enable tolerance once the motor reaches the target position. 0: The P-controller ignores absolute errors below the tolerance. 1: The P-controller ignores absolute errors below tolerance only when the pos_reached flag (RAMP_STAT[9]) of the motion controller is set to 1.	0/1	Default = 1

Table 20. Closed-Loop Position Control Related Parameters (continued)

PARAMETER	DESCRIPTION	RANGE	NOTES
tolerance	P-controller error tolerance setting Errors < tolerance are ignored. Default = 0.	0 to 255	—
P	Proportional parameter for position P-regulator. P = 0 disables the P-regulator function. P > 0 enables the P-regulator function. A factor of 1 means that a position deviation of +5 or -5 leads to a corrective velocity of +5 or -5. Default = 0.	UINT 10.0	No fractional part

StallGuard2 Load Measurement

To fit different motor control schemes, the TMC5271 offers 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 becomes overloaded/stalls. Table 21 lists related parameters.

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

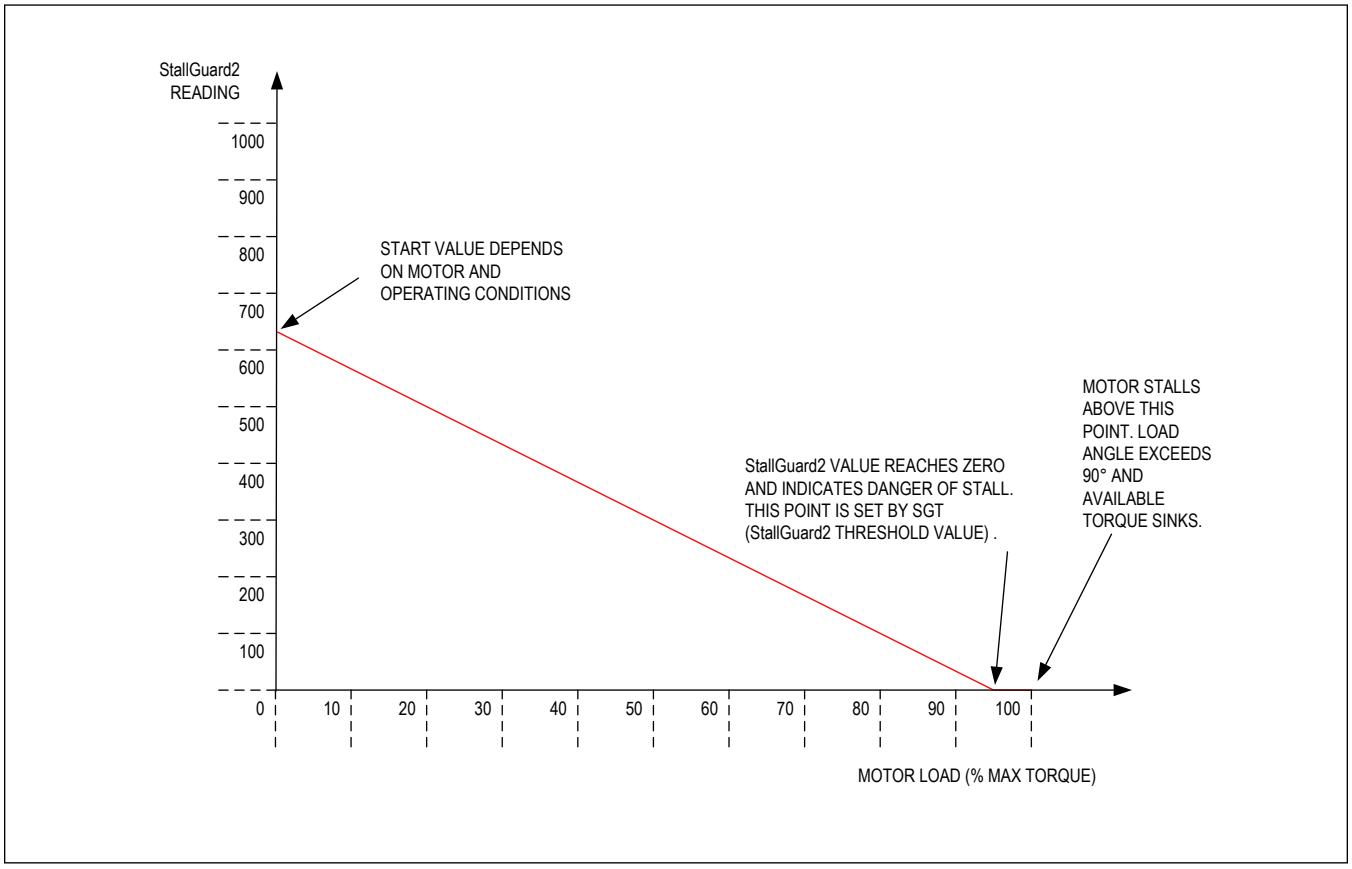


Figure 35. Function Principle of StallGuard2

Table 21. 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 that 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	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).	0	Standard mode
		1	Filtered mode

Table 21. StallGuard2-Related Parameters (continued)

PARAMETER	DESCRIPTION	SETTING	NOTES
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:

1. Operate the motor at the normal operation velocity, supply voltage, and current setting for the application and monitor SG_RESULT.
2. 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.
3. Next, enable sg_stop and make sure that 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).
4. 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 that the setting works reliably in a certain range (e.g., 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 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, etc.

1. Operate the motor at low velocity < 10RPM (i.e., 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, mechanical load does not make a big difference on the result.
2. 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.
3. 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, e.g., during 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.

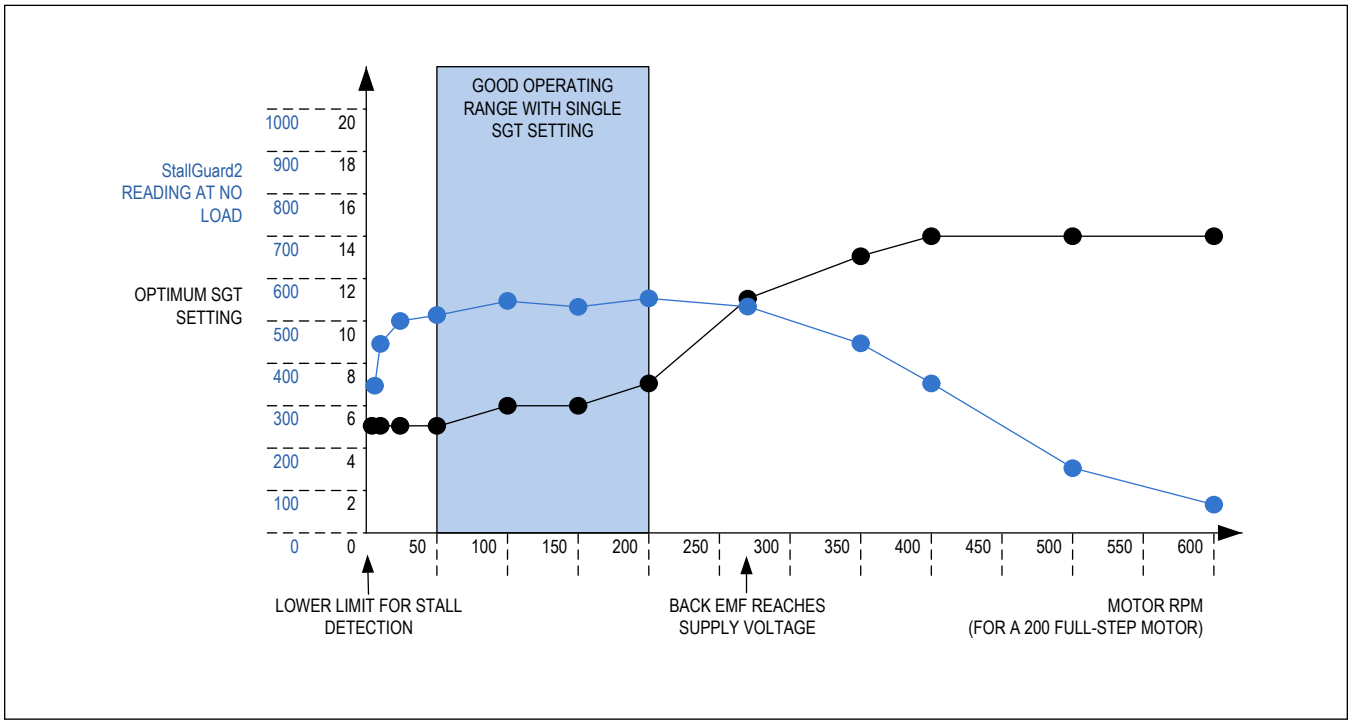


Figure 36. Optimum SGT Setting and StallGuard2 Reading with an Example Motor

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 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 needs to be determined, which the motor can drive without stalling. At the same time, monitor the SG_RESULT value at this load, e.g., 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 that the start point is far enough away 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 motion and sets VACTUAL zero, stopping the motor. The stop condition is also indicated by the StallGuard2 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 expiration of TZEROWAIT when the motion parameters have not been 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 (e.g., 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 22 lists related parameters of StallGuard4.

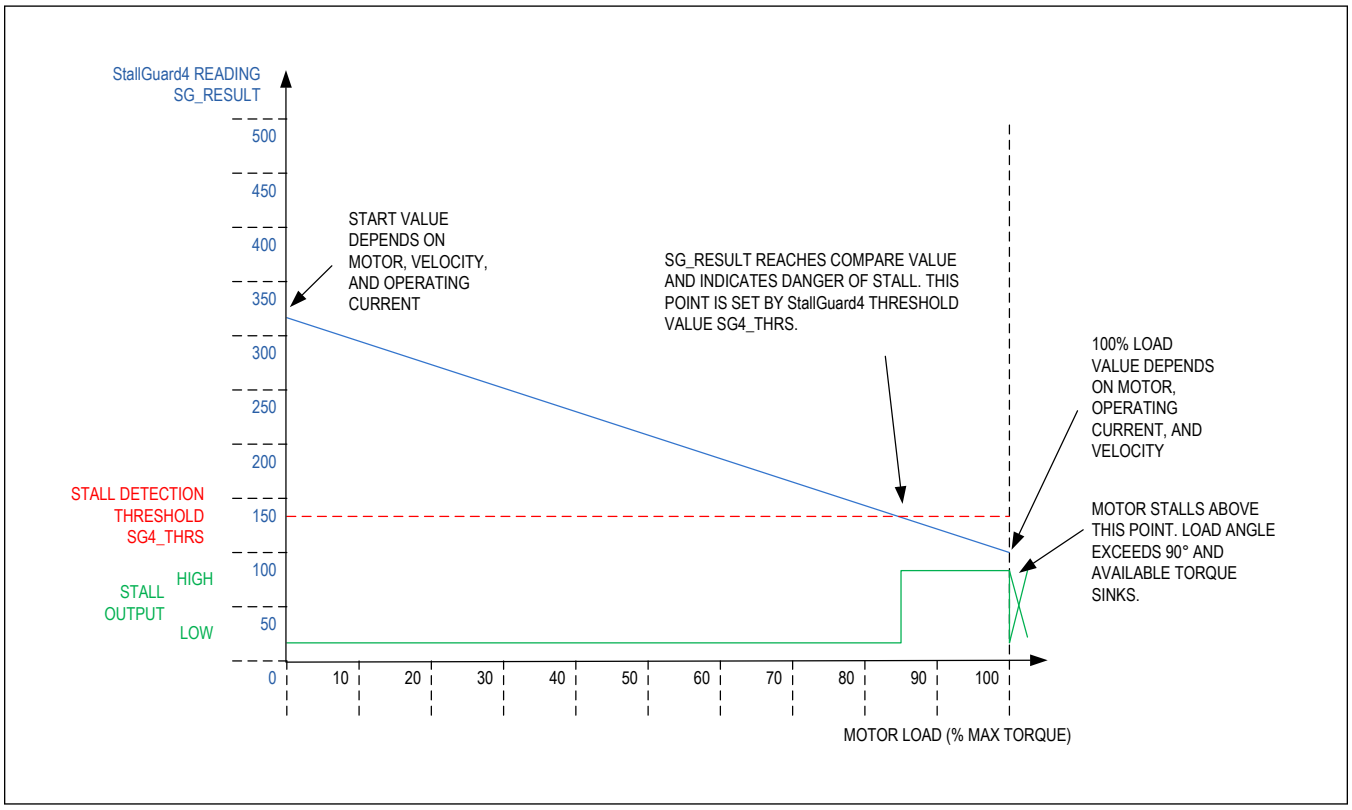


Figure 37. StallGuard4 Mode of Operation

Table 22. 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 255	This value is compared to SG4_RESULT. The stall output becomes 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 becomes 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.

Table 22. StallGuard4-Related Parameters (continued)

PARAMETER	DESCRIPTION	SETTING	NOTES
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.
sg_angle_offset	This flag enables optimized switching between StealthChop2 and SpreadCycle by using the SG4_RESULT to determine the phase lag in StealthChop2 and compensate for the phase jump when switching from voltage-controlled to current-controlled operation in SpreadCycle. The phase offset is stored and is subtracted again when switching back to StealthChop2.	0/1	0: No angle correction (default). 1: Optimized switching between StealthChop2 and SpreadCycle.

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:

1. Operate the motor at the normal operation velocity for the application and monitor SG4_RESULT.
2. 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 (apply half of the value).
3. 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 that the motor is safely stopped whenever it is stalled. Increase SG4_THRS if the motor is stopped before a stall occurs.
4. 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 a velocity range. Make sure that the setting works reliably in a certain range (e.g., 80% to 120% of 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 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:

1. sg4_filt_en = 0: A single measurement, updated after each full step and valid for each full step. This measurement allows 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.

2. sg4_filt_en = 1: In this mode, four individual signals are generated: SG4_IND_0 upon falling zero transition of the cosine wave (coil A); SG4_IND_1 upon rising zero transition of the cosine wave; SG4_IND_2 upon falling zero transition

of the sine wave (coil B); SG4_IND_3 upon 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 detection of soft obstacles or for 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 optimum values are determined. Therefore, the maximum load needs to 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 which 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.

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 prior to use. A single tuning does not cover all operating points.

Setting up for CoolStep

CoolStep is controlled by several parameters but the two parameters listed in Table 23 are critical for understanding how it works. Additional parameters related to CoolStep are given in Table 24.

Table 23. 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 x 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) x 32. Default = 0.	0 to 15	Threshold is (SEMIN + SEMAX + 1) x 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 x 32 and CoolStep increases the current. When the load decreases, SG_RESULT rises above (SEMIN + SEMAX + 1) x 32, and the current is reduced as shown in Figure 38.

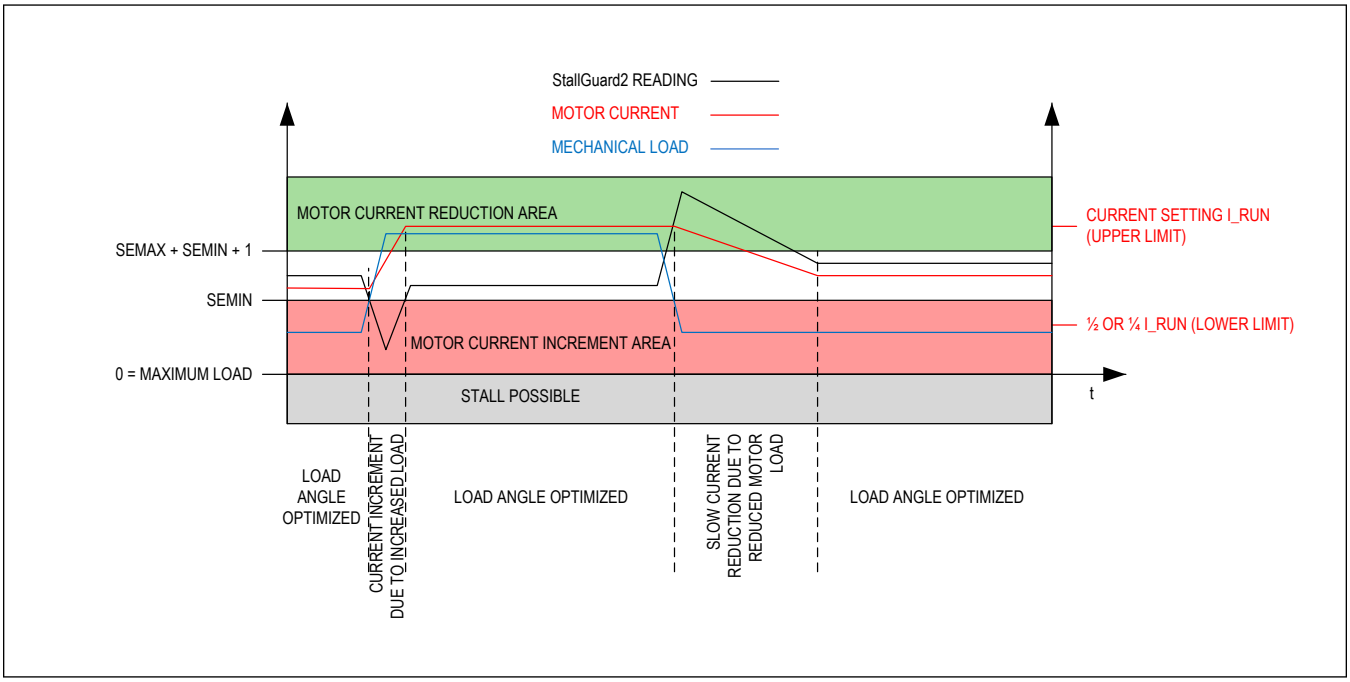


Figure 38. CoolStep Adapts Motor Current to the Load

Table 24. Additional CoolStep Parameters and Status Information

PARAMETER	DESCRIPTION	RANGE	NOTES
SEUP	Sets the current increment step. The motor current becomes 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 becomes disabled. Adapt to the lower limit of the velocity range where StallGuard2 gives a stable result. **Tip:** Can be adapted to disable CoolStep during acceleration and deceleration phase by setting VCOOLTHRS identical to VMAX. Default = 0.	1 to 2^{20} - 1	Specifies lower CoolStep velocity by comparing the threshold value to TSTEP.

Table 24. Additional CoolStep Parameters and Status Information (continued)

PARAMETER	DESCRIPTION	RANGE	NOTES
THIGH	Upper velocity threshold value for CoolStep. Above this velocity CoolStep becomes 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. They can be tuned separately because they 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 limits controlled by the current scale parameter IRUN and the 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 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 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.

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 GCONF bits diag0_int_pushpull and diag1_poscomp_pushpull, either an open collector (active-low) output signal can be chosen (default) or an active-low push-pull output signal. When using the open collector output, an external pull-up resistor in the 4.7kΩ to 33kΩ range is required.

Figure 39 shows the DIAG0 circuit. DIAG0 is driven low upon a reset condition. With GCONF bit diag0_sel_nError_Ramp, a first selection can be made between signals and flags related to the internal motion controller or driver status and error flags. Various configuration bits allow enabling specific flags and interrupt signals to DIAG0. GCONF bit diag0_stall_step allows mapping of the internal step signal of the motion controller as a debug output to DIAG0. This step signal shows an active edge (rising as well as falling) per each microstep generated by the motion controller.

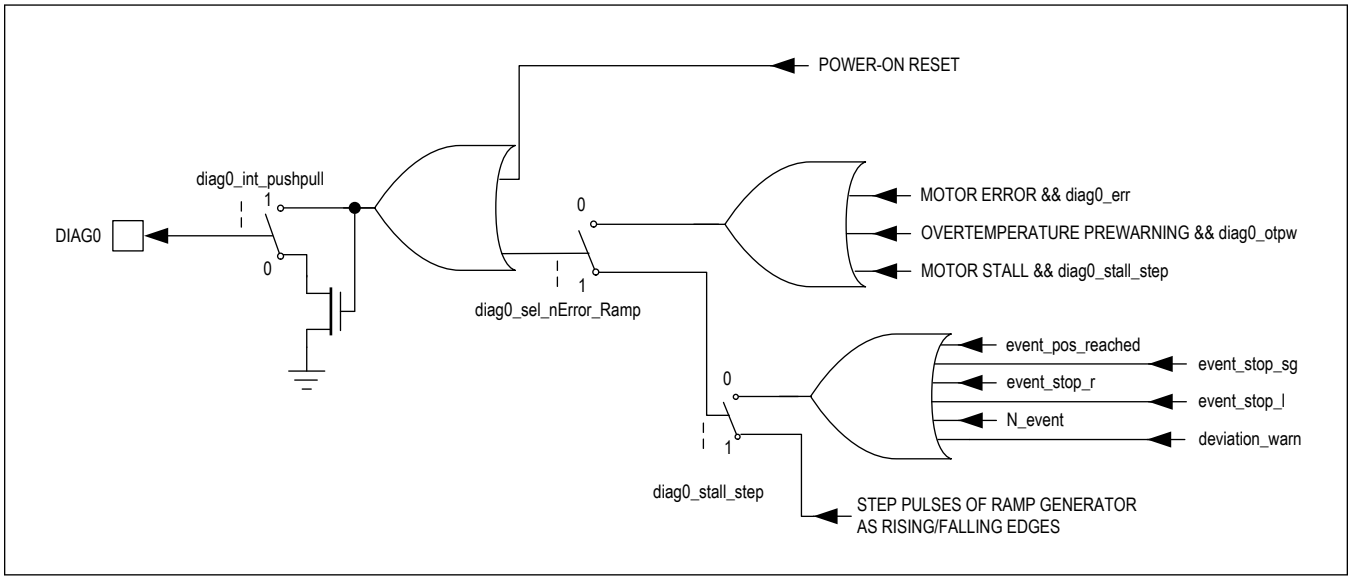


Figure 39. DIAG0 Output Circuit

Figure 40 shows the DIAG1 circuit structure. For DIAG1 with GCONF bit diag1_sel_nStallIndex_xcomp, a first selection can be made between signals and flags related to the internal motion controller's position-compare function or motor driver stall and index flags. Configuration bits allow enabling specific flags and interrupt signals to DIAG1. GCONF bit diag1_stall_dir allows mapping of the internal direction signal of the motion controller as a debug output to DIAG1.

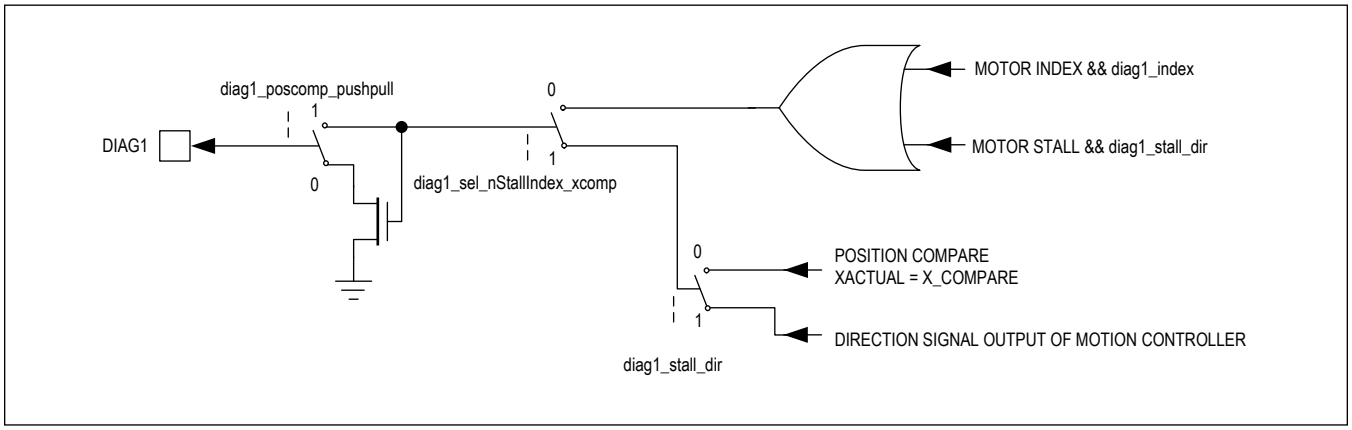


Figure 40. DIAG1 Output Circuit

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 becomes 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 becomes completely blocked, the stall flag becomes set.

This mode is only available when the internal motion controller is used. In 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 41. 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 (which is the highest torque the motor can provide). Additionally, maximum application velocity can be increased up to the actually reachable motor velocity.

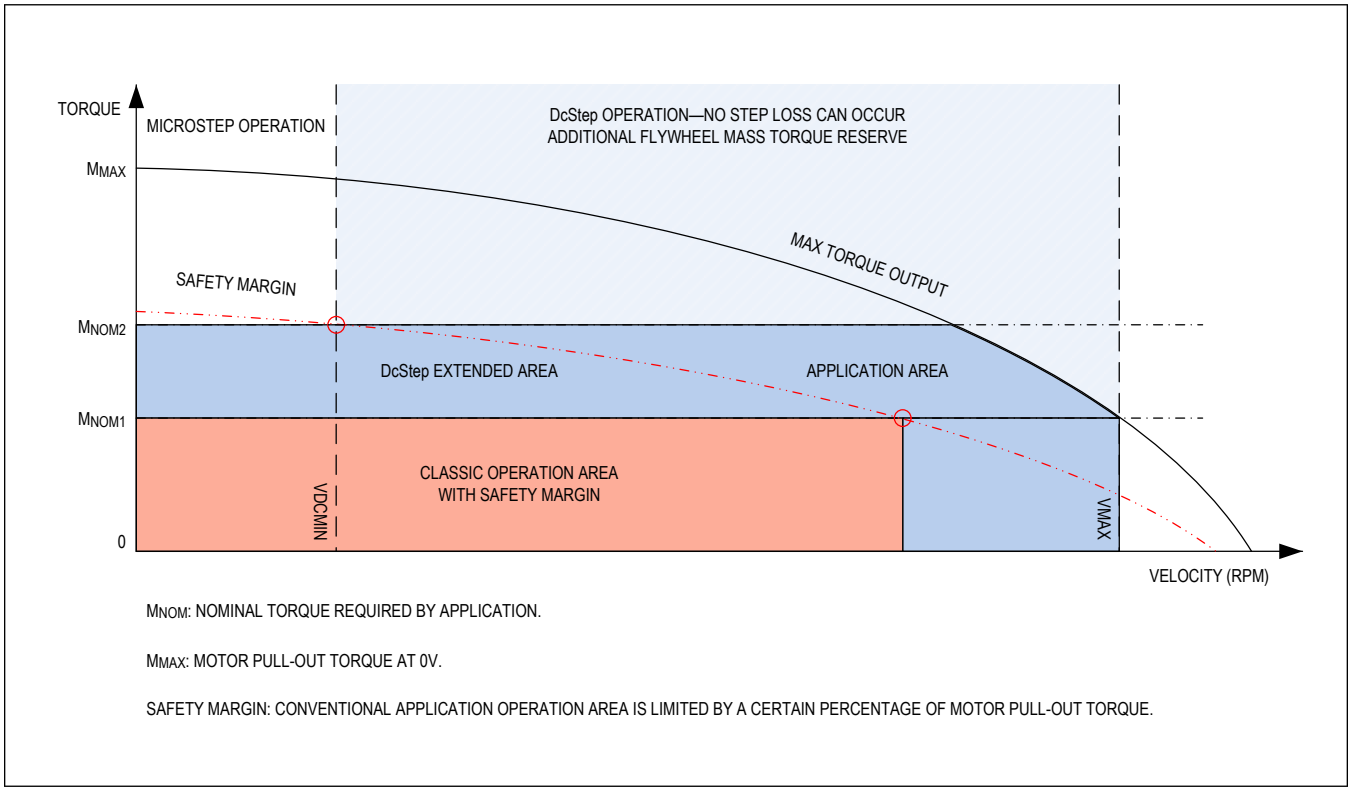


Figure 41. DcStep Extended Application Operation Area

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 becomes seamlessly integrated into the motion ramp even if the motor becomes overloaded with respect to the target velocity. DcStep operates the motor using the constant T_{OFF} chopper in full-step mode at the ramp generator target velocity VMAX or at reduced velocity if the motor becomes overloaded. Figure 42 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 becomes 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 becomes enabled (set sg_stop). Stall detection is covered by StallGuard2/4 when the velocity goes below VDCMIN.

Note: DcStep requires that 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. A phase shift by 1 disturbs DcStep operation. Therefore, it is advised to work with the default wave. See the *Sine-Wave Lookup Table* section for an initialization with the default table.

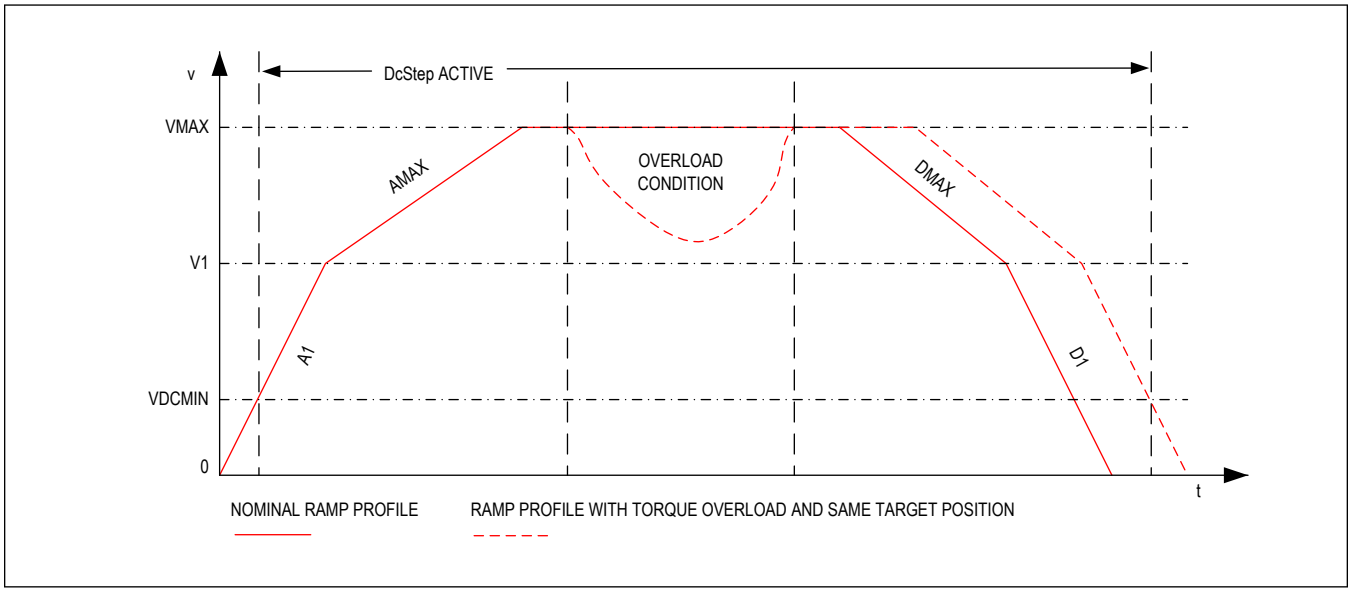


Figure 42. DcStep Velocity Profile with Overload Condition

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 becomes decelerated below a motor-dependent minimum velocity where the motor operation cannot safely be detected anymore, the motor may stall and loose 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 becomes 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 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 25 shows 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 25. Parameters for Stall Detection in DcStep Mode

PARAMETER	DESCRIPTION	RANGE	NOTES
vhighfs and vhighchm	These chopper configuration flags in CHOPCONF need to be set for DcStep operation. As soon as VDCMIN is exceeded, the chopper is switched to constant T_{OFF} 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.

Table 25. Parameters for Stall Detection in DcStep Mode (continued)

PARAMETER	DESCRIPTION	RANGE	NOTES
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 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 (e.g., lowest operation supply voltage, highest motor temperature, and highest supply voltage, lowest motor temperature).	0 to 1023	Lower limit for the setting is: t_{BLANK} (as defined by TBL) in 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 becomes 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:

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

An example of actual motor velocity in DcStep operation is:

At 16.0MHz clock frequency, a 0.954s measurement delay would directly yield in the velocity value; a 9.54ms delay would yield 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 NCS for SPI transmission provides the most exact time reference.

Sine-Wave Lookup Table

The TMC5271 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 user benefits are:

- Microstepping—extremely improved with low-cost motors
- Motor—runs smoothly and quietly
- Torque—reduced mechanical resonances yields improved torque
- Low-frequency motor noise—reduced by adapting the sine and cosine wave shift for the actual motor's manufacturing tolerance

The TMC5271 allows 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 *Integrated Current Sensing* section).

Microstep Table

To minimize required memory and the amount of data to be programmed, only a quarter of the wave becomes stored. The internal microstep table maps the microstep wave from 0° to 90°. It becomes 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, etc.

When modifying the wave, care must be taken to ensure a smooth and symmetrical zero transition when the quarter wave becomes 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. An example is given in Figure 43.

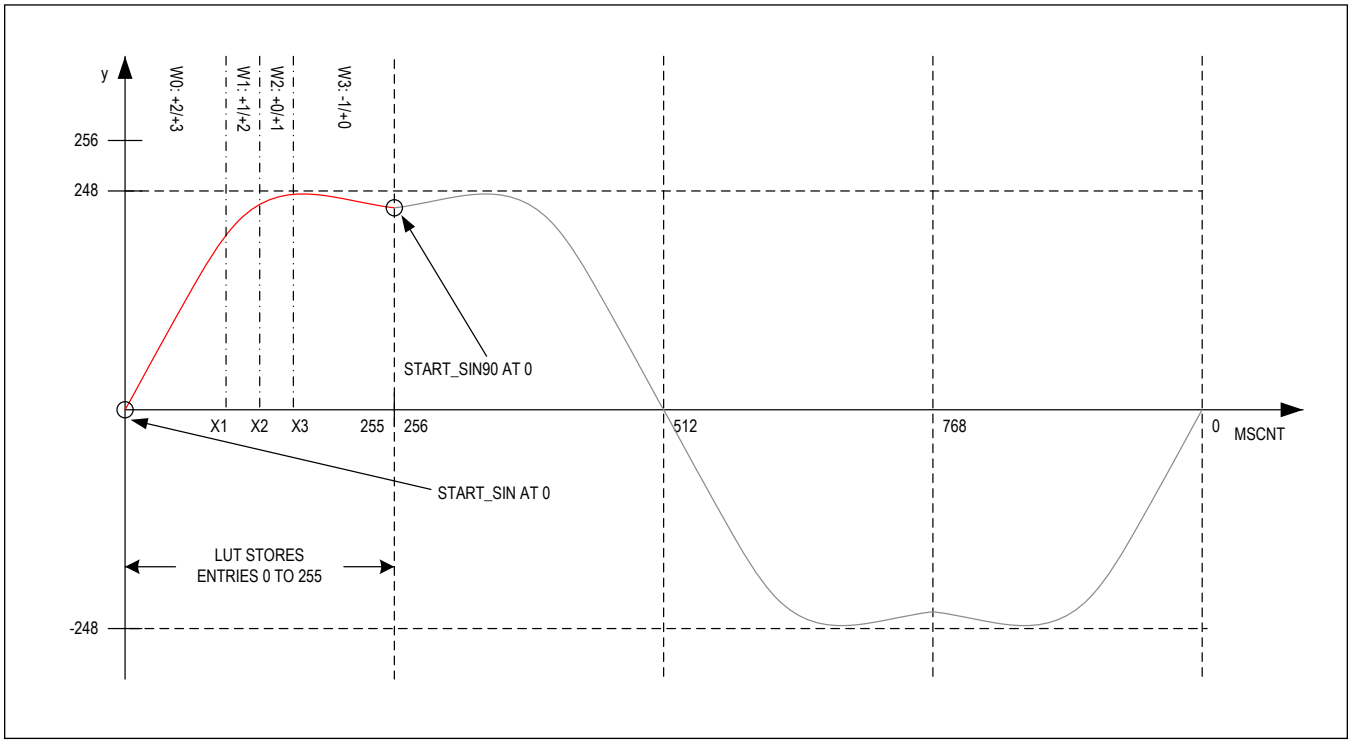


Figure 43. LUT Programming Example

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 becomes modified. Therefore, CUR_A and CUR_B become 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 44.

- 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 needs to 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.

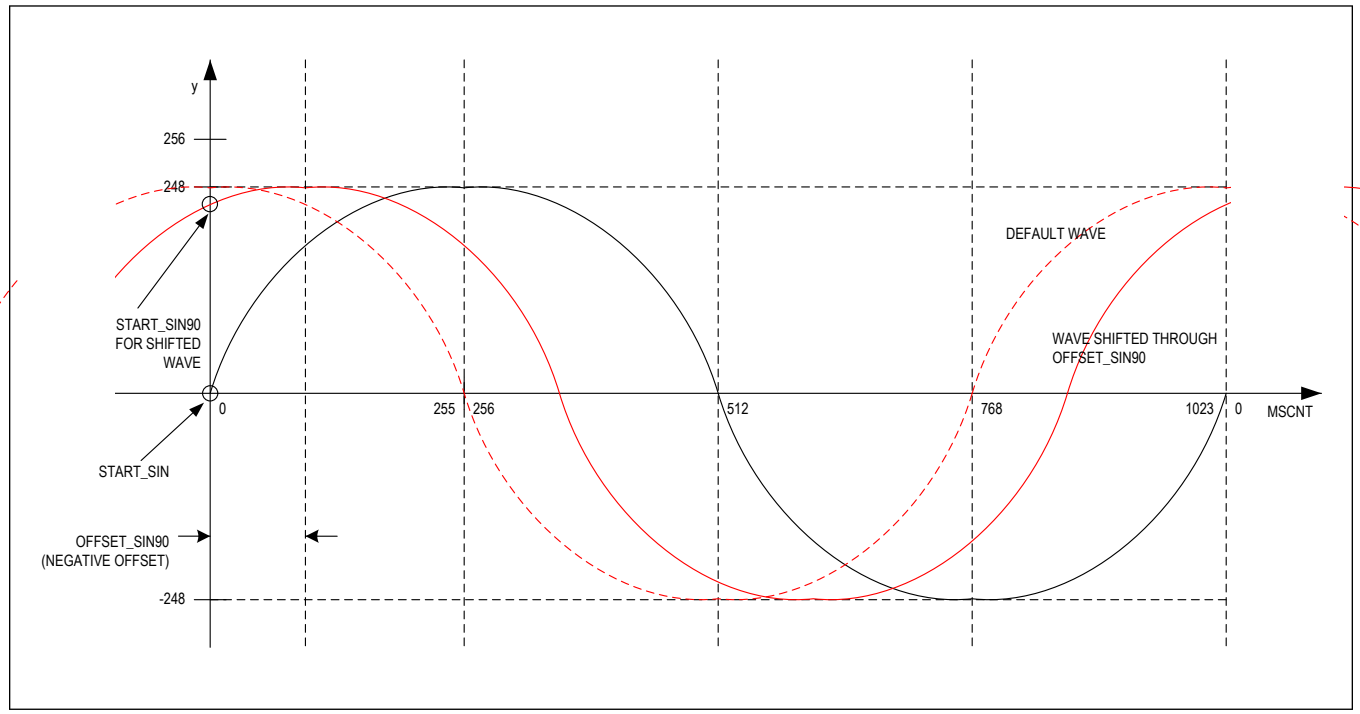


Figure 44. Shifting the Cosine Wave Through OFFSET_SIN90

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 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 BEMF decoder allows the detection of motor motion while the motor is disabled (i.e., 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 becomes turned by an external force, its motion can be tracked. The result can be used to trigger a new homing sequence if the motor has been 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, e.g., 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 has to be checked with the actual motor and application mechanics.
- The system counts in multiples of one full step. By setting an adapted encoder 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. The principal mode of operation is shown in Figure 45.

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

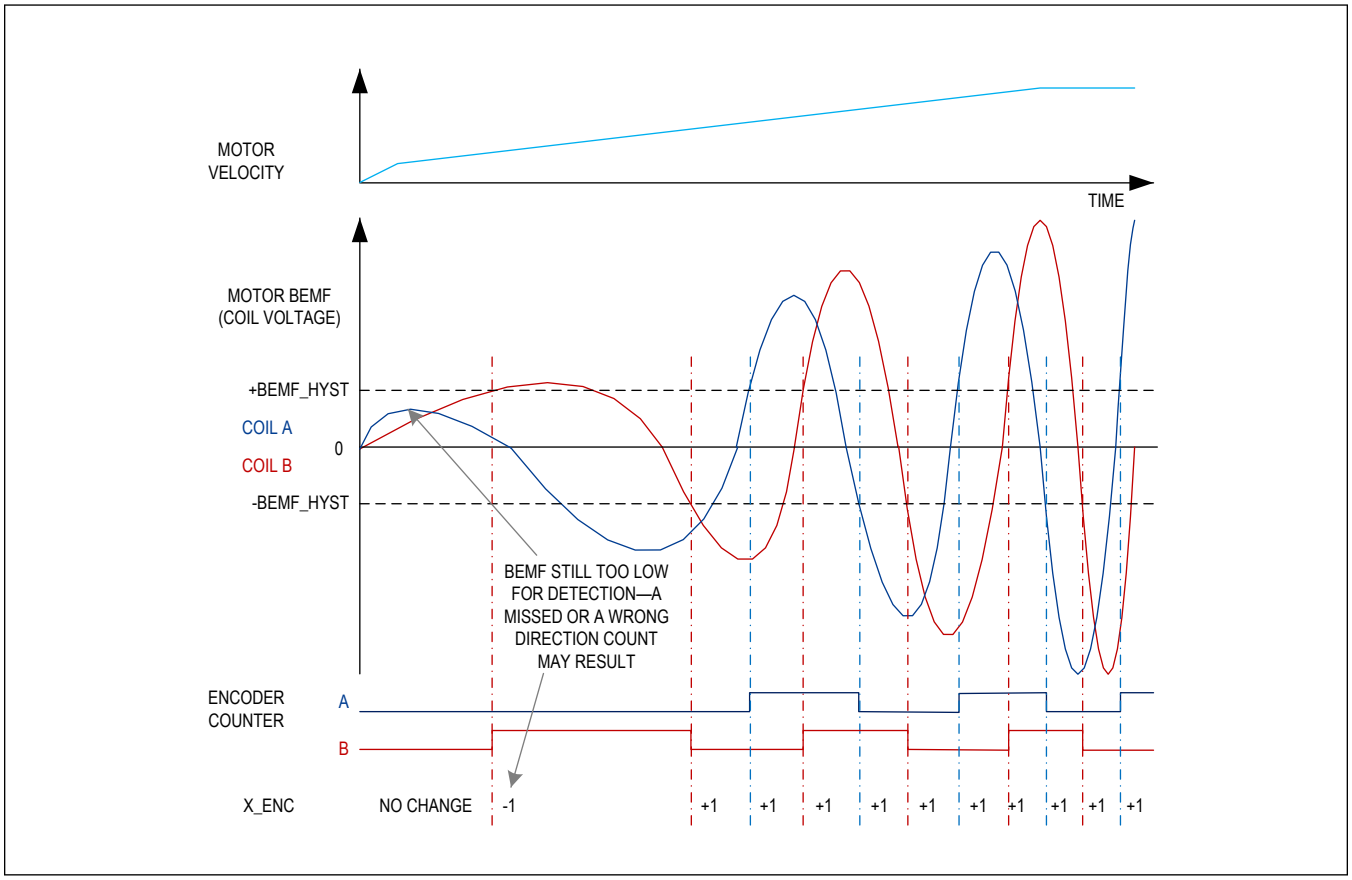


Figure 45. Detection of Motor Movement

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.

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

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

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

Consequently, encoder 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.

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

Time to Enable

To operate the motor, the motor has to be in freewheeling or passive braking mode. Depending on the preceding operation, a certain delay time might be required to ensure that the remaining motor coil current has deceased. Following 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 remaining motor current (BEMF_BLANK_TIME) by feeding it back to the power supply. Any remaining current might look like a motor BEMF and could otherwise lead to a step detection.

Relevant Settings

1. Configuration: Enable BEMF decoder

These settings are required to automatically enable the BEMF decoder, whenever the motor is in freewheeling mode:

- ENCMODE_NBEMF_ABN_SEL: Set to 0 to disable external encoder inputs. Motor must be at standstill (stst = 1).
- PWM_CONF_FREEWHEEL: Set to 1 (freewheel) or 2 (passive braking).

StealthChop2:

- IHOLD: Set to 0, to enable freewheeling options during standstill. CS_ACTUAL has to reach 0.

SpreadCycle:

- Set TOFF = 0 to disable motor and allow freewheeling.

Low-power mode:

- In low-power mode, it is sufficient to set ENCMODE_NBEMF_ABN_SEL = 0 and ENCMODE_QSC_ENC_EN = 1 to enable BEMF encoder during low-power mode.

2. Operational Settings

- ENCMODE_BEMF_BLANK_TIME: Has to be expired before encoder feature becomes active.
- ENCMODE_BEMF_HYST (0 to 7): Hysteresis 10, 25, 50, 75, 100, 150, 200, and 250mV (typ).
- ENCMODE_BEMF_BLANK_TIME: Counts in multiple of 2^{14} clock cycles. This time allows the motor coil current resulting from 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.
- DRV_CONF_Mx_STANDSTILL_TIME: Shorten the time to standstill detection to speed up the ramp down of motor current to 0 as controlled by the IHOLD_IRUN_IHOLDDELAY setting. The default is 2^{20} clock cycles.
- ENCMODE_BEMF_FILTER_SEL: Digital filter time for decoded BEMF signals.
  - 0: f_{CLK}/24600
  - 1: f_{CLK}/12300
  - 2: f_{CLK}/6150
  - 3: f_{CLK}/2870
  - 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.
- ENC_CONST: Set to match X_ENC counter to the motor microstep resolution e.g., 256 for 256 microstep setting. With this setting, each motor full step increases/decreases X_ENC by 256.
- X_ENC: Can be set to 0 or to X_ACTUAL before encoder becomes activated. Read out the step difference or updated motor position while encoder is in operation.

ABN Incremental Encoder Interface

The TMC5271 comes with an incremental encoder interface for ABN encoders. The encoder gives positions by digital incremental quadrature signals (usually named A and B) and an index signal (usually named N for null, Z for zero, or I for index).

To use the incremental encoder interface, it must be actively enabled. After reset and power-up, the sensorless full-step encoder is chosen by default. To enable the incremental encoder interface, set the nBEMF_ABN_SEL bit in ENCMODE register.

N Signal

The N signal can be used to clear the position counter or to take a snapshot of XACTUAL. To continuously monitor the N channel and trigger clearing of the encoder position or latching of the position, where the N channel event has been detected, set the clr_cont flag. Alternatively, it is possible to react to the next encoder N channel event only, and automatically disable the clearing or latching of the encoder position after the first N signal event (clr_once flag). This might be desired because the encoder gives this signal once for each revolution.

To check for an encoder latched event:

- Option 1: Check ENC_LATCH for change. It starts up with 0 and shows the encoder count where the N event occurred after starting motion for the first time. For consecutive rotations, it shows increased/decreased values and thus always changes.
- Option 2: Check for the interrupt output active and read the flag only following active interrupt output. The DIAG0 pin needs to be configured for the interrupt lines using the diag0_nint_step bit from the GCONF register.

Some encoders require a validation of the N signal by a certain configuration of A and B polarity. This can be controlled by the pol_A and pol_B flags in the ENCMODE register. For example, when both pol_A and pol_B are set, an active N event is only accepted during a high polarity of both A and B channel.

For clearing the encoder position ENC_POS with the next active N event, set clr_enc_x = 1 and clr_once = 1 or clr_cont = 1.

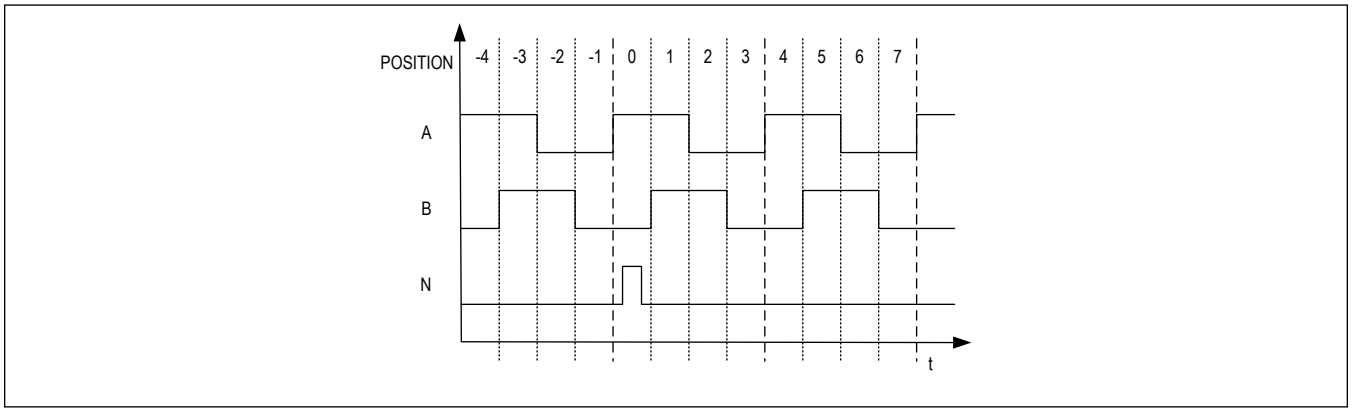


Figure 46. Outline of ABN Signals of an Incremental Encoder

X_ENC Encoder Counter

The X_ENC encoder counter holds the current encoder position ready for read-out. Different modes of handling of the signals A, B, and N take into account active-low and active-high signals found with different types of encoders as shown in Figure 46.

ENC_STATUS Register

The ENC_STATUS register holds the status of the event of an encoder clear upon an N channel signal. The ENC_LATCH register always stores the actual encoder position on an N signal event.

ENC_CONST Encoder Constant

The encoder constant (or encoder factor) ENC_CONST is added to or subtracted from the encoder counter on each polarity change of the quadrature signals AB of the incremental encoder. The encoder constant ENC_CONST represents a signed fixed point number (16.16) to facilitate the generic adaption between motors and encoders. In decimal mode, the lower 16 bits represent a number between 0 and 9999. For stepper motors equipped with incremental encoders, the fixed number representation allows very comfortable parameterization. Additionally, mechanical gearing can easily be taken into account. Negating the sign of ENC_CONST allows inversion of the counting direction to match motor and encoder direction.

Examples:

- Encoder factor of 1.0: *ENC_CONST* = 0x0001.0x0000 = FACTOR.FRACTION
- Encoder factor of -1.0: ENC_CONST = 0xFFFF.0x0000. This is the two's complement of 0x00010000. It equals (2^{16} - (FACTOR + 1)).(2^{16} - FRACTION)
- Decimal mode encoder factor 25.6: 00025.6000 = 0x0019.0x1770 = FACTOR.DECIMALS (DECIMALS = first 4 digits of fraction)
- Decimal mode encoder factor -25.6: (2^{16} – (25 + 1)).(10000 - 6000) = (2^{16} – 26).(4000) = 0xFFE6.0x0FA0.
- A negative encoder constant is calculated using the following equation: (2^{16} - (FACTOR + 1)).(10000 - DECIMALS)

Setting the Encoder to Match Motor Resolution

In Table 26, the encoder example setting for motor parameters are:

1. USC = 256 microsteps.
2. FSC = 200 full-step motor.
3. Factor = FSC x USC/encoder resolution.

Table 26. Encoder Example Settings for a 200 Full-Step Motor with 256 Microsteps

ENCODER RESOLUTION	REQUIRED ENCODER FACTOR	NOTES
200	256	Use this encoder factor for the BEMF encoder to match motor steps to 1/256 microstepping.
360	142.2222 = 9320675.5555/2^{16} = 1422222.2222/10000	No exact match possible.
500	102.4 = 6710886.4/2^{16} = 1024000/10000	Exact match with decimal setting.
1000	51.2	Exact match with decimal setting.
1024	50	—
4000	12.8	Exact match with decimal setting.
4096	12.5	—
16384	3.125	—

Example:

The encoder constant register must be programmed to 51.2 in decimal mode. Therefore, set

ENC_CONST = 51 × 2^{16} + 0.2 × 10000

Chip Emergency Stop and Power Down Modes

The TMC5271 comes with different emergency stop modes and power-down modes that can be controlled by pins and registers.

Table 27 gives an overview on these modes. The maximum IC power consumption refers only to the IC itself and does not take into account the actual motor current.

Table 27. Chip Mode Comparison

MODE	ENTRY CONDITION	EXIT CONDITION	REGISTER CONTENT	MAX IC POWER CONSUMPTION	SPI/ UART	MOTOR POWER
Emergency stop over SPI/UART	GCONF register (0x0): drv_enn = 1	GCONF register (0x0): drv_enn = 0	No change	~6mA	Active	Driver disabled immediately

Table 27. Chip Mode Comparison (continued)

MODE	ENTRY CONDITION	EXIT CONDITION	REGISTER CONTENT	MAX IC POWER CONSUMPTION	SPI/ UART	MOTOR POWER
Configurable external emergency stop with standstill options	To enable this mode: GCONF register (0x0): stop_enable = 1 To trigger and keep this mode, pin A1 pulled to and held at 1. To configure this mode, IHOLD_IRUN register (0x12): IHOLD and IHOLD_DELAY. PWMCONF register (0x3C): FREEWHEEL.	Pin A1 pulled to 0	No change	~6mA	Active	Driver active but motor stops with configured hold current or StealthChop2 freewheel option
Hard external emergency stop	To enable this mode, DRV_CONF register (0x5): en_em_disable = 1. To configure this mode, DRV_CONF register (0x5): sel_em_stop_src. To trigger this mode, pins REFL and REFR = 1.	Pins (REFL and REFR = 0)	No change	~6mA	Active	Driver disabled immediately
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 15 qsc = 1. To further configure the quiescence mode, ENCMODE register (0x2F): qsc_enc_en = 1 to enable encoder during quiescence mode.	GCONF register (0x0): qsc_sts_ena = 0	No change	~750µA	Active	Driver disabled immediately
Reset	Low pulse > 30µs on SLEEPN	SLEEPN = 1	Reset to default	~1µA (while in reset)	Not active	Driver disabled after SLEEPN input filter delay (~50µs)
Sleep mode	SLEEPN pulled low permanently	SLEEPN = 1	Reset to default	~1µA	Not active	Driver disabled after SLEEPN input filter delay (~50µs)

Emergency Stop Using SPI or UART

The TMC5271 provides 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 GCONF register bit drv_enn to 1.

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 A to act as a motor stop and disable function. Set the GCONF bit stop_enable to activate this option.

Whenever pin A becomes 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 (FREEWHEEL parameter in the PWMCONF register) when StealthChop2 is in use. When pin A is pulled low again, the motor driver returns to default operational mode.

Hard Emergency Stop

This mode is a special hard stop mode to switch off the motor driver immediately when it is triggered by the external control pins. Care needs to be taken since a hard stop at high velocities can lead to a high BEMF voltage that can destroy the device.

The mode is enabled and configured using the en_em_disable and sel_em_stop_src bits in the DRV_CONF register (0x5). The mode is then triggered using the REFL and REFR pins. Both pins REFL and REFR must be shorted to high (1) to trigger the hard stop.

Low-Power Quiescence Mode

The low-power quiescence mode is a special TMC5271 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 approximately 750µA.

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 becomes completely disabled when entering this mode. It must be re-enabled by setting the drv_enn bit in the GCONF register with a second register access after qsc_sts_ena has been set to 0 again.

By default, the incremental encoder unit is still enabled during this mode to allow for position tracking. Using the qsc_enc_en bit in the ENCMODE (0x2F) 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 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 sleep mode and reset condition, the registers need to be reconfigured. While reconfiguring the IC, it is advised to keep the bridge drivers disabled using the GCONF register bit drv_enn.

Do not use during high motor velocity as energy fed back from the motor can damage the chip. If not used, connect SLEEPN to V_S (this is a high-voltage pin).

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. Additional information is given in the *Electrical Characteristics* section.

Using an External Clock

Using a dedicated external clock allows the user to precisely calculate the time base for all velocity and acceleration computations inside the TMC5271. 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 *Electrical Characteristics* section. The clock's duty cycle requirements need to be satisfied especially at clock frequencies close to

20MHz.

Make sure that 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 the ext_clk bit in the IOIN register gives feedback on which clock source is currently in use (1 = external clock).

When the external clock fails or is switched off, the internal clock takes over seamlessly and automatically to protect the driver from damage. While the TMC5271 is in quiescence power-down mode, the clock source switches to the internal clock.

Protections and Driver Diagnostics

The TMC5271 driver provides a complete set of diagnostics, status information, and protection capabilities e.g., 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 TMC5271 has an internal thermal protection. If the die temperature exceeds +165°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 +145°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 the IOIN[8:1] register as the ADC_TEMPERATURE parameter. This measurement function can be disabled with the ADC_EN control bit in the IOIN[9] register to reduce energy consumption if temperature measurement is not needed. The ADC_TEMPERATURE value can be converted into degrees Celsius with the following formula:

T(°C) = (2.03 x ADC_TEMPERATURE) - 259

Heat is mainly generated by the motor driver stages. Most critical situations where the driver MOSFETs could be overheated are avoided when enabling the short-to-GND protection. 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 TMC5271 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 e.g., when touching the conducting parts connected to the system ground. The short detection is protected against spurious triggering e.g., by ESD discharges (by retrying three times before switching off the motor).

Once a short condition is safely detected, the corresponding driver bridge is switched off and the s2ga or s2gb flag is set. To restart the motor, the user must intervene by disabling and re-enabling the driver. It should be noted 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 Table 28.

Table 28. Overcurrent Protection Thresholds Based on the Full-Scale Current Setting

FULL-SCALE CURRENT SETTING FSR (BITS)	OVERCURRENT PROTECTION THRESHOLD (A)
11	3.00
10	2.25
01	1.50
00	0.75

Open-Load Diagnostics

Interrupted cables are a common cause for systems failing e.g., when connectors are not firmly plugged. The TMC5271 detects open-load conditions by checking if it 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 the user 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 TMC5271 features a UVLO protection for +V_S and +V_{CC_IO}. The UVLO condition on +V_S is triggered below 1.9V (max). The UVLO condition on +V_{CC_IO} is triggered below 1.3V (max).

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 pin depending on the configured pin settings.

During a +V_{CC_IO} UVLO, 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 TMC5271 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 protection against hot plugging of a motor.

Quick Configuration Guides

The following guides in Figure 47 to Figure 56 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 advanced functionalities and options, instead concentrating on the basic function set to make a motor run smoothly. Once the motor runs, the user can decide to explore additional features and further functionality in more detail. A current probe on one motor coil is a good aid to find the best settings.

Current Setting

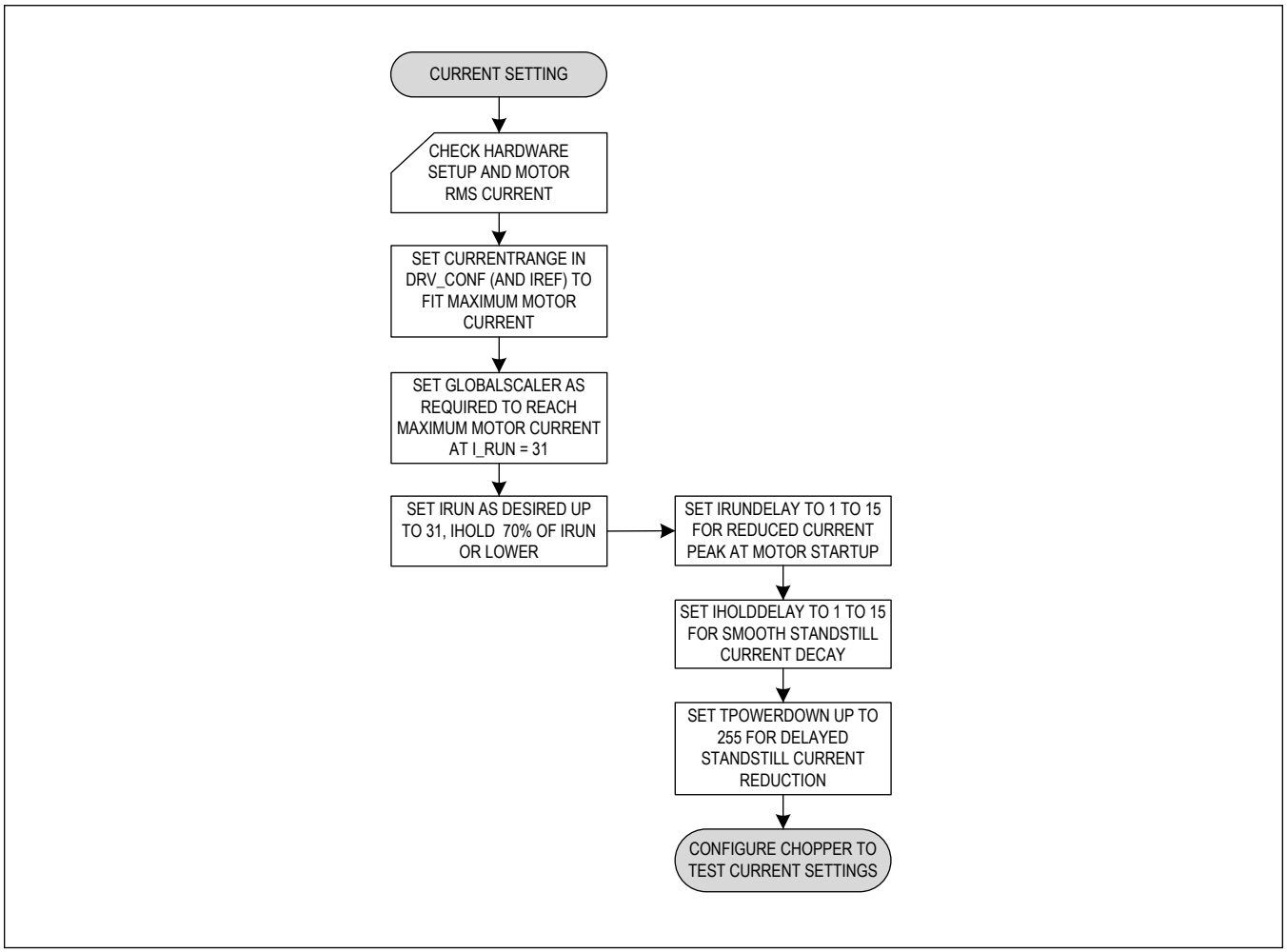


Figure 47. Quick Configuration Guide—Current Setting

StealthChop2 Configuration

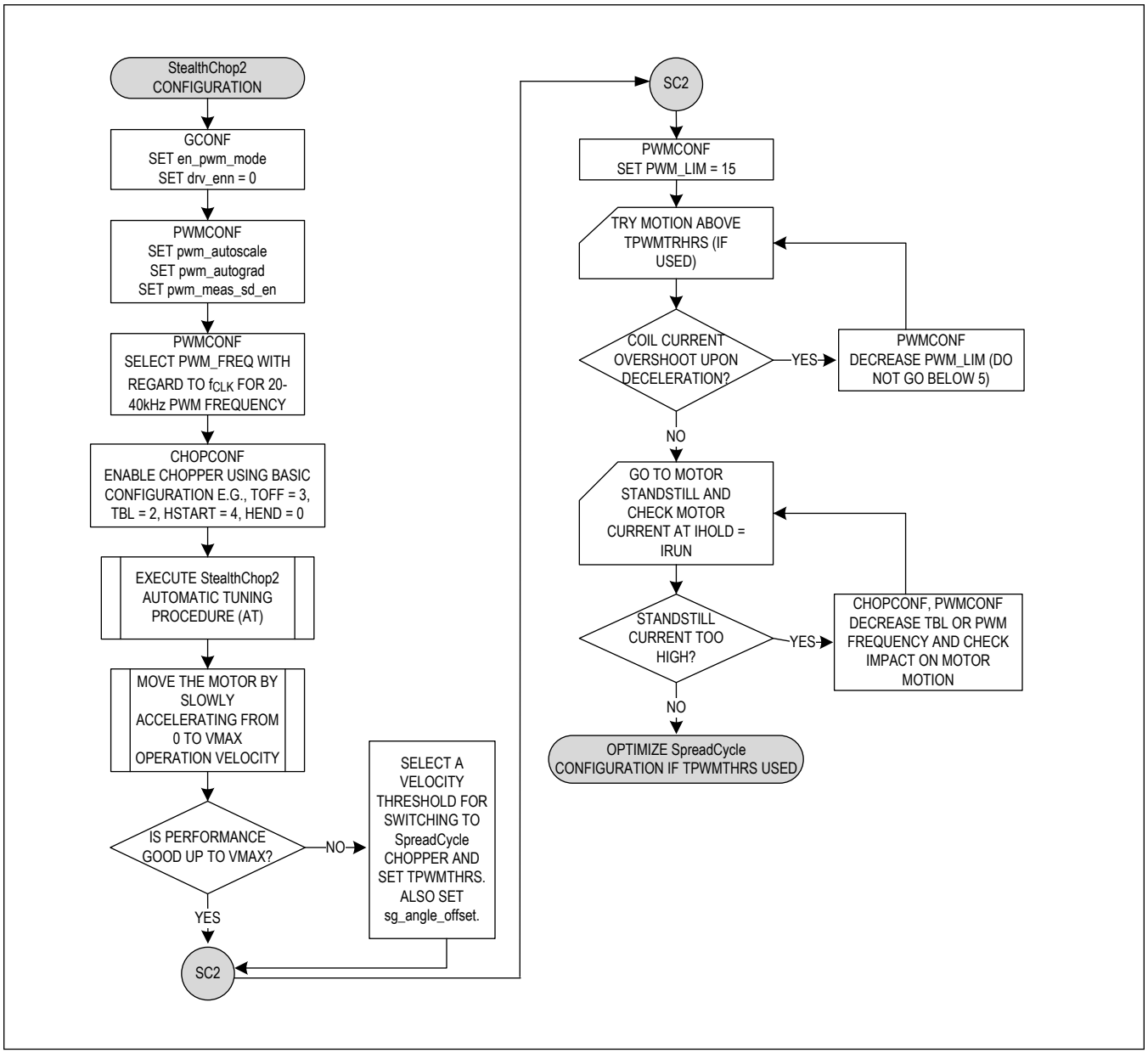


Figure 48. Quick Configuration Guide—StealthChop2 Configuration

SpreadCycle Configuration

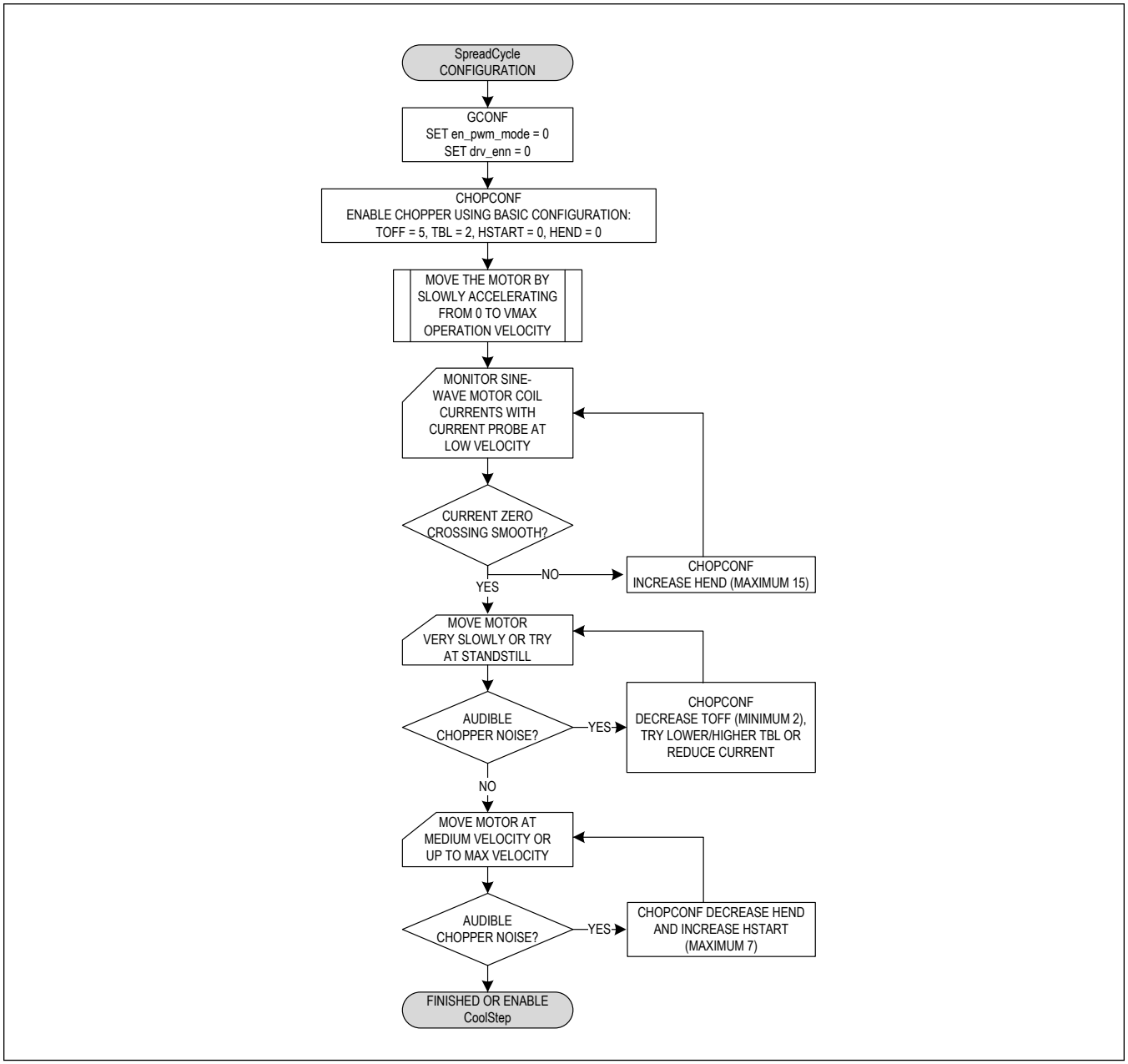


Figure 49. Quick Configuration Guide—SpreadCycle

Enabling CoolStep in Combination with StealthChop2

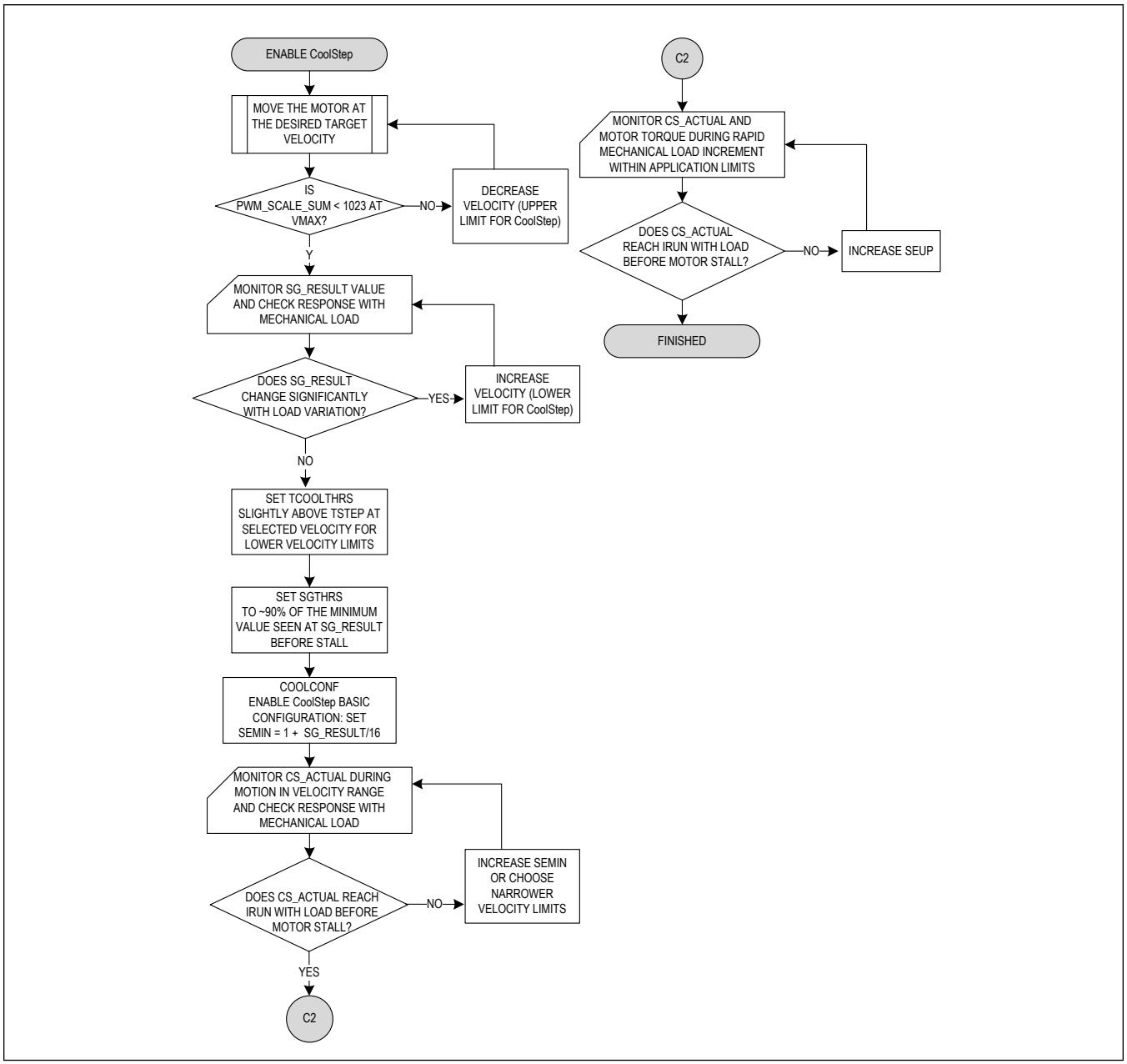


Figure 50. Quick Configuration Guide—CoolStep with StealthChop2

Enabling CoolStep in Combination with SpreadCycle

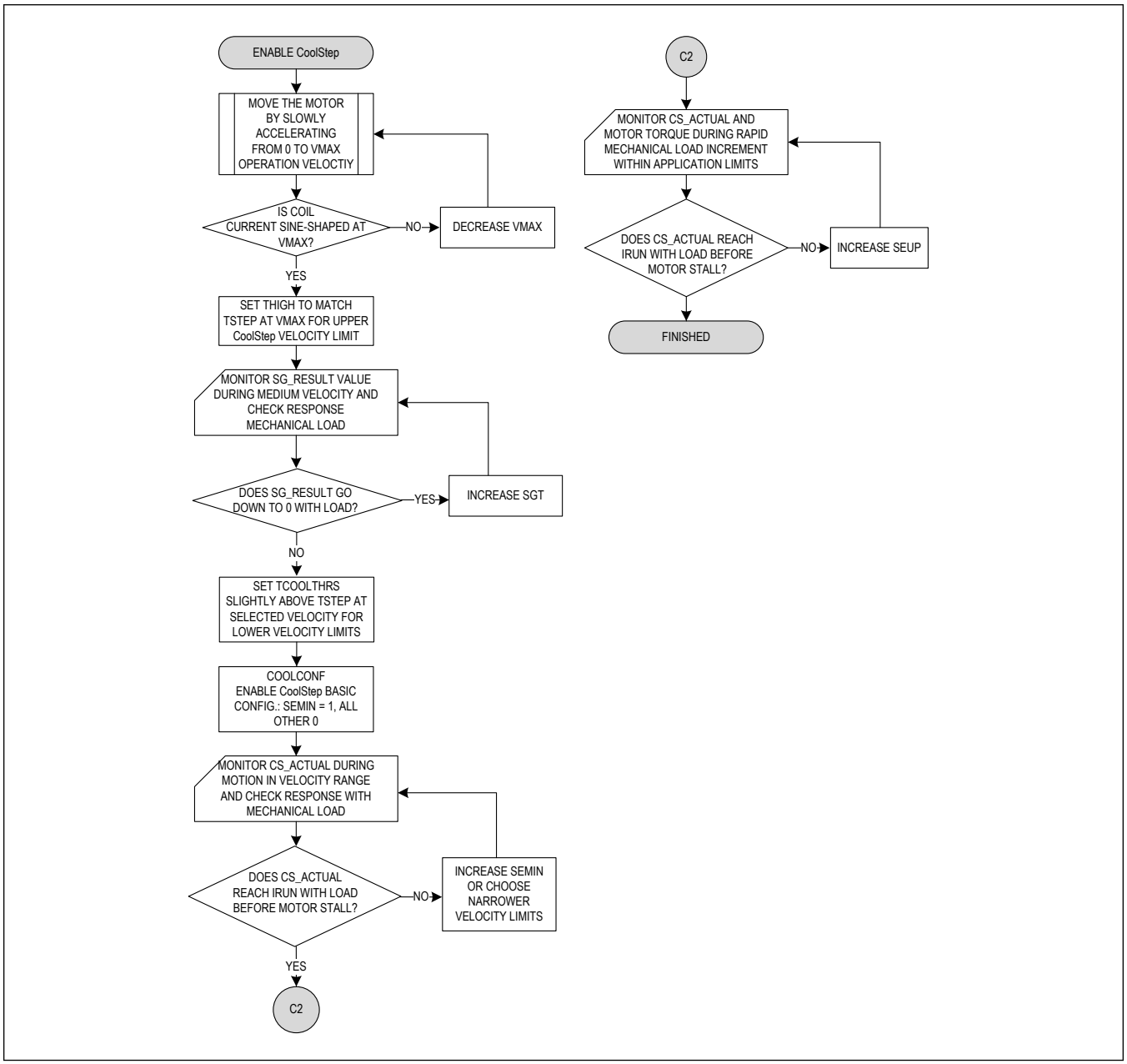


Figure 51. Quick Configuration Guide—CoolStep with SpreadCycle

Moving the Motor Using the Motion Controller

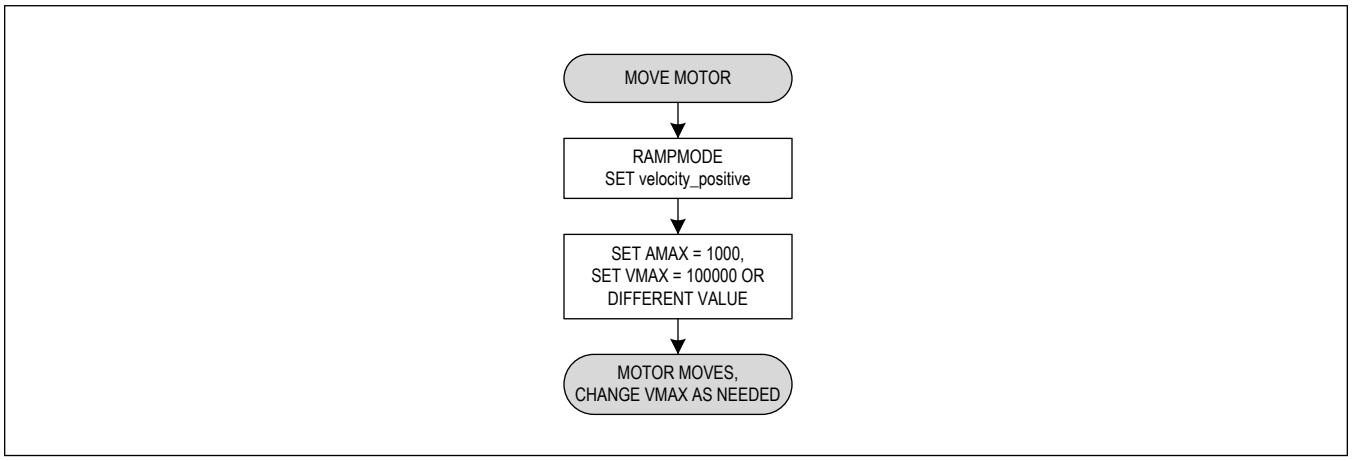


Figure 52. Quick Configuration Guide—Moving a Motor in Velocity Mode

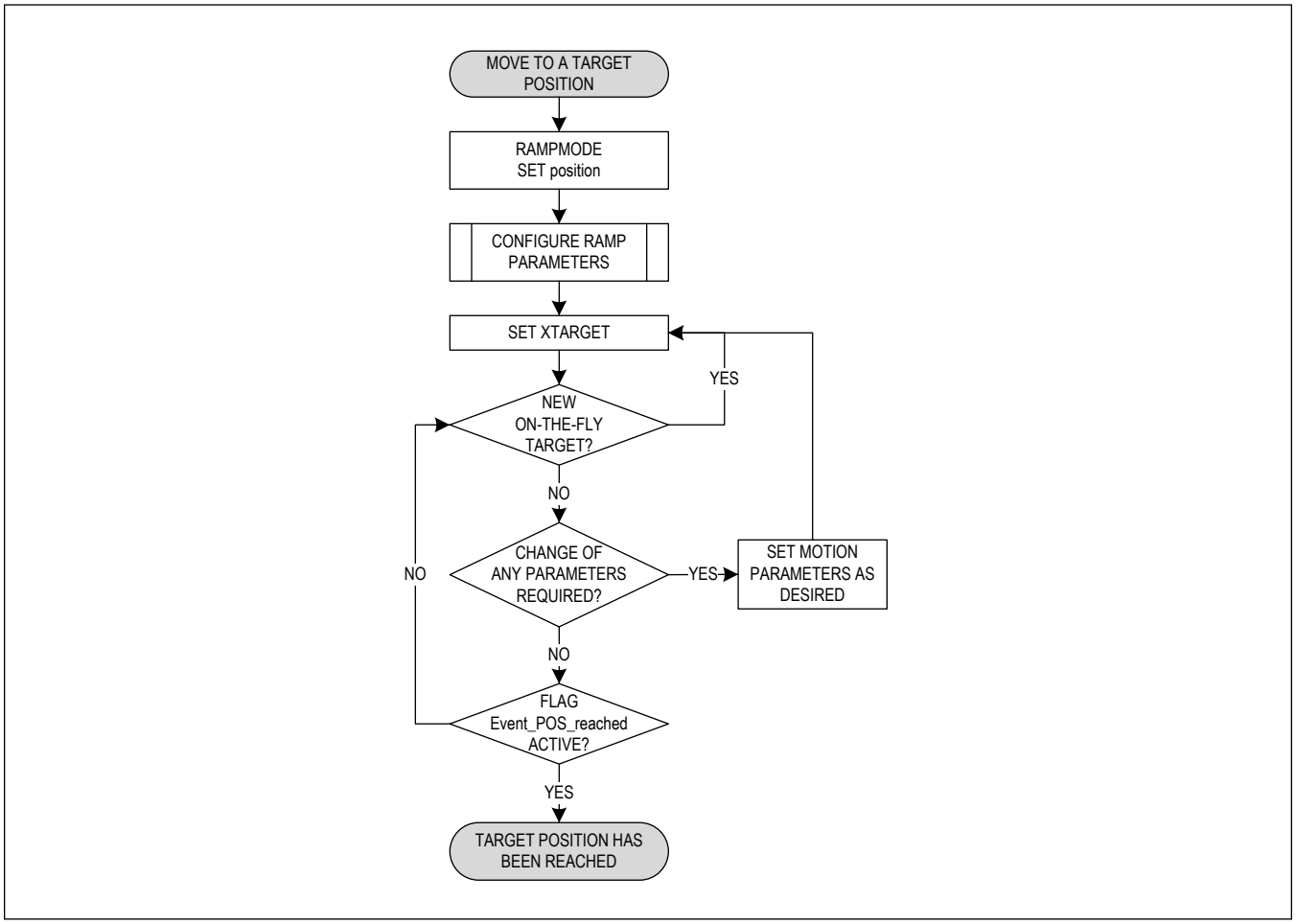


Figure 53. Quick Configuration Guide—Moving a Motor to a Target Position

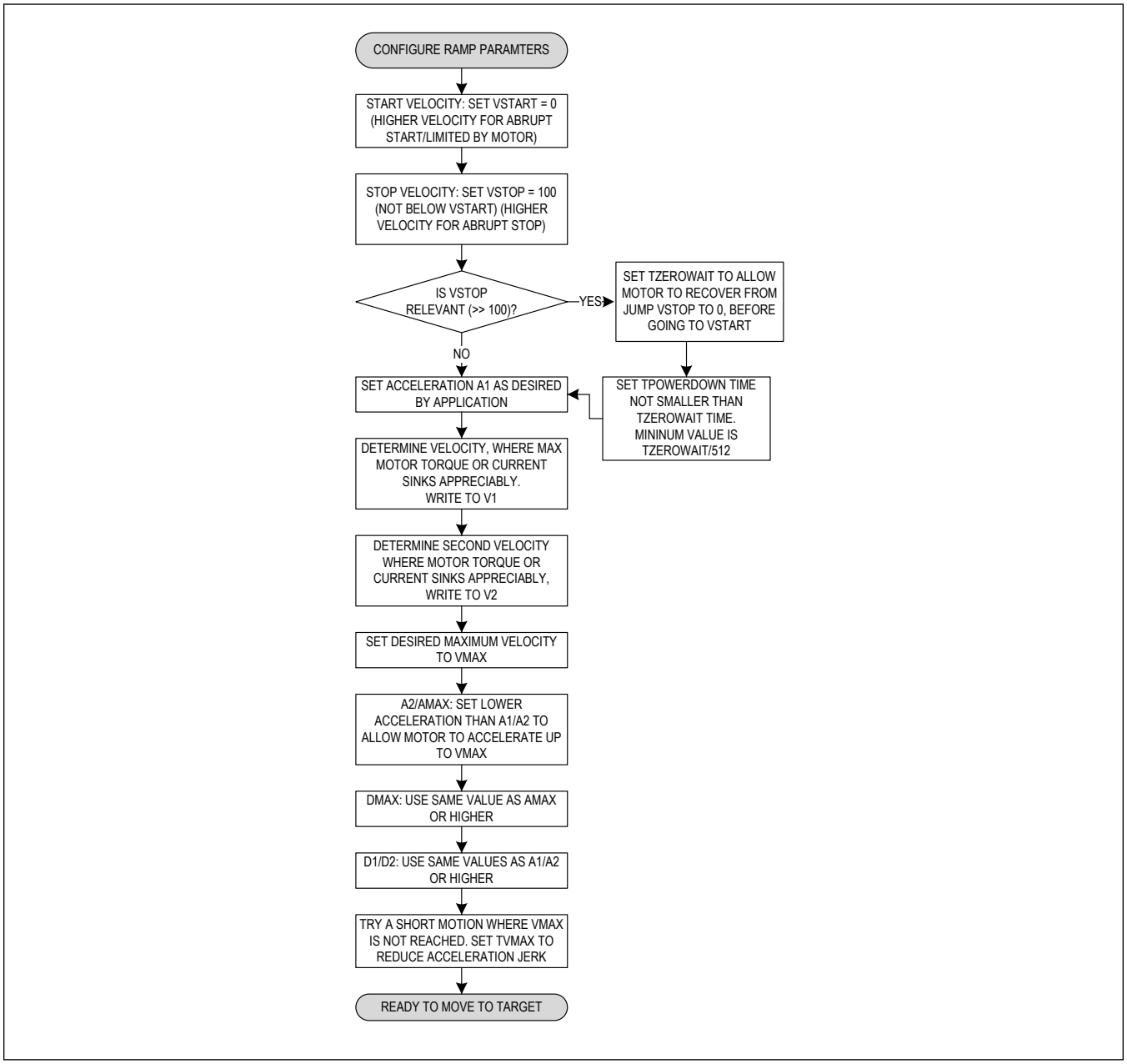


Figure 54. Quick Configuration Guide—Motion Ramp Parameter Setting

Enabling DcStep Operation

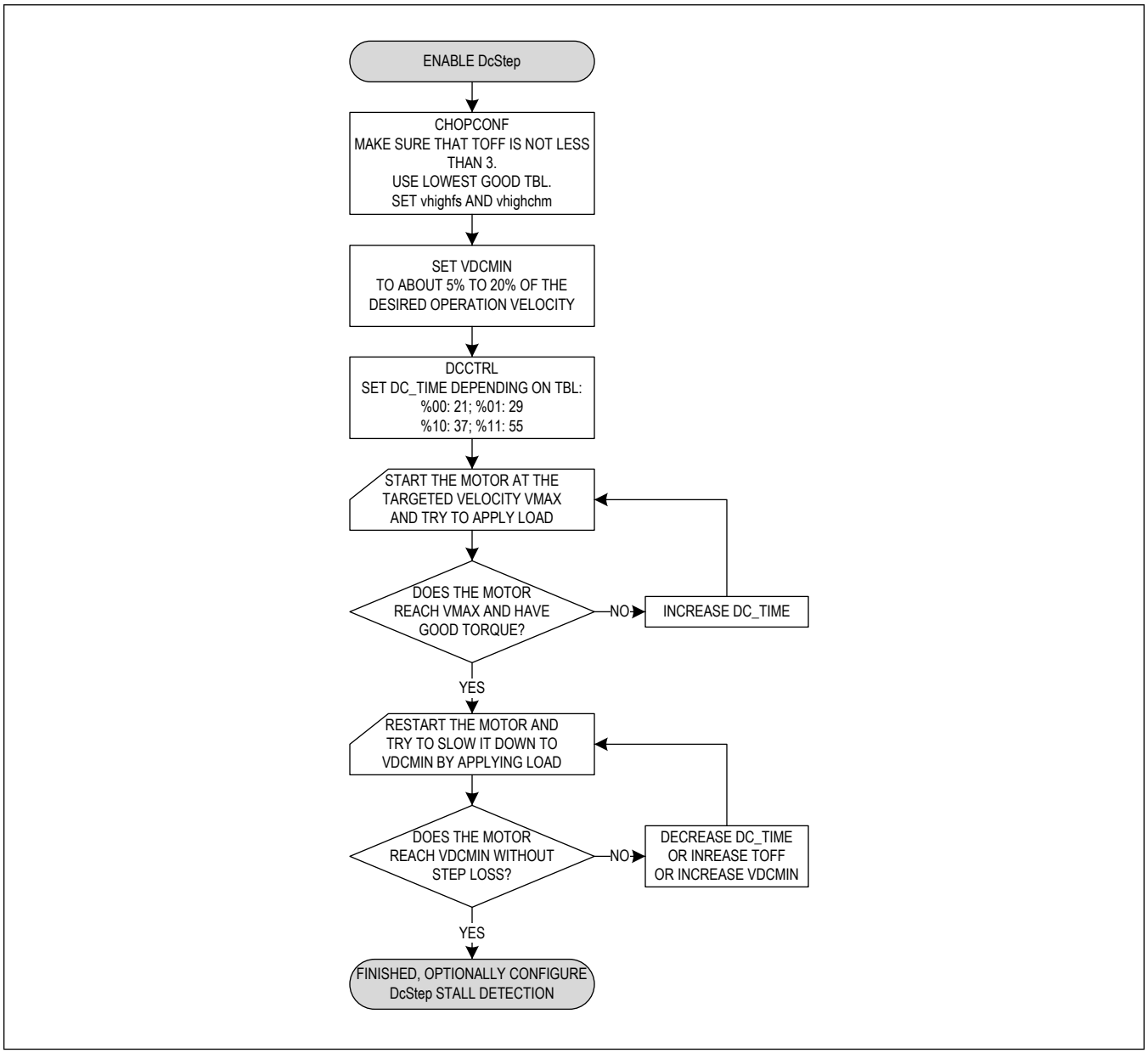


Figure 55. Quick Configuration Guide—DcStep

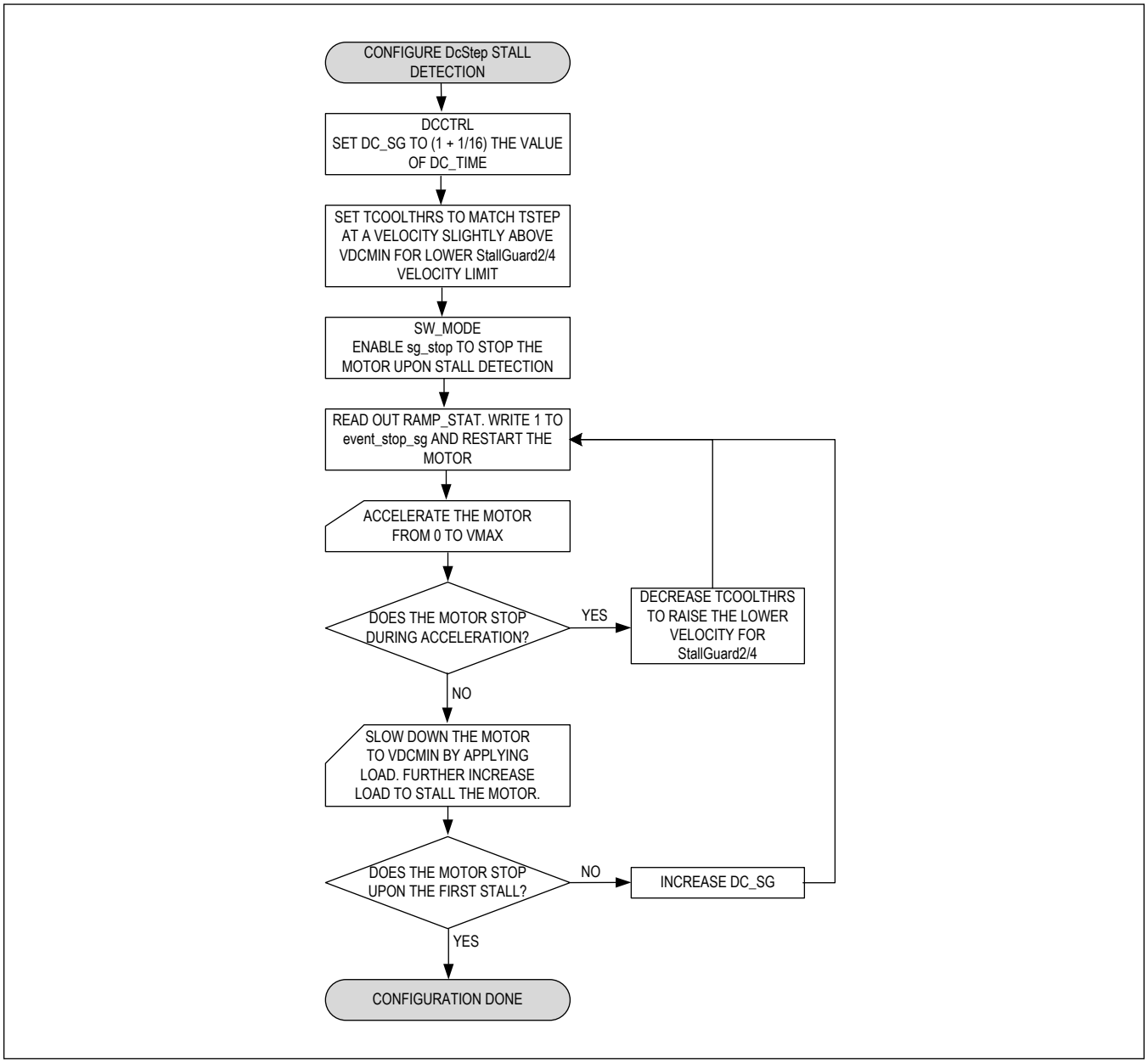


Figure 56. Quick Configuration Guide—Using Stall Detection with DcStep

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 $I_{COILNOM}$ [A]

- Nominal coil resistance R_{COIL} [Ω]
- Rated coil voltage $U_N = R_{COIL} \times I_{COILNOM}$ [V] (sometimes specified instead of $I_{COILNOM}$)
- Holding torque at $I_{COILNOM}$ HoldingTorque [Nm]
- Back EMF (BEFM) constant C_{BEMF} [V/rad/s]. More information on the motor BEMF is also given in the *Understanding the Back EMF Constant of a Motor* section.

The specified motor torque is reached with the RMS I_{COIL} current in both motor coils to build up the required magnetic field strength. Essentially, a lower current proportionally generates a lower torque, e.g., 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 U_{BAT} for motor standstill and slow motion considering the driver's power stage resistance (LS + HS):

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

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

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

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

Adapt the motor to battery operation: With most motor suppliers the user can adapt the coil winding. 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.

Register Map

TMC5271

ADDRESS	NAME	MSB							LSB
General Configuration Registers									
0x00	GCONF[31:24]	–	diag1_sel_nStallIndex_xcomp	diag0_sel_nError_Ramp	–	–	–	–	–
	GCONF[23:16]	–	–	–	–	–	–	–	–
	GCONF[15:8]	qsc_sts_ena	drv_enn	SD	direct_mode	stop_enable	small_hysteresis	diag1_poscomp_pushpull	diag0_int_pushpull
	GCONF[7:0]	diag1_index	diag1_stall_dir	diag0_stall_step	diag0_otpw	diag0_error	shaft	multistep_filt	en_pwm_mode
0x01	GSTAT[31:24]	–	–	–	–	–	–	–	–
	GSTAT[23:16]	–	–	–	–	–	–	–	–
	GSTAT[15:8]	–	–	–	–	–	–	–	–
	GSTAT[7:0]	–	–	–	vm_uvlo	register_reset	uv_ldo	drv_err	reset
0x02	IFCNT[31:24]	–	–	–	–	–	–	–	–
	IFCNT[23:16]	–	–	–	–	–	–	–	–
	IFCNT[15:8]	–	–	–	–	–	–	–	–
	IFCNT[7:0]	IFCNT[7:0]							
0x03	NODECONF[31:24]	–	–	–	–	–	–	–	–
	NODECONF[23:16]	–	–	–	–	–	–	–	–
	NODECONF[15:8]	–	–	–	–	SENDDELAY[3:0]			
	NODECONF[7:0]	NODEADDR[7:0]							
0x04	IOIN[31:24]	VERSION[7:0]							
	IOIN[23:16]	–	–	SPI_FLT_SEL[1:0]		–	SILICON_RV[2:0]		
	IOIN[15:8]	qsc	–	OUTPUT	EXT_RES_DET	EXT_CLK	–	ADC_EN	ADC_TEMPERATURE[7]
	IOIN[7:0]	ADC_TEMPERATURE[6:0]							Reserved
0x05	DRV_CONF[31:24]	–	–	–	–	–	–	–	–
	DRV_CONF[23:16]	–	–	–	–	–	STANDSTILL_TIME[2:0]		
	DRV_CONF[15:8]	–	–	–	–	–	–	–	–
	DRV_CONF[7:0]	–	–	–	en_em_disable	FSR_IREF[1:0]		FSR[1:0]	
0x06	GLOBAL SCALER[31:24]	–	–	–	–	–	–	–	–
	GLOBAL SCALER[23:16]	–	–	–	–	–	–	–	–
	GLOBAL SCALER[15:8]	GLOBALSCALER_B[7:0]							
	GLOBAL SCALER[7:0]	GLOBALSCALER_A[7:0]							

ADDRESS	NAME	MSB							LSB
0x07	RAMPMODE[31:24]	–	–	–	–	–	–	–	–
	RAMPMODE[23:16]	–	–	–	–	–	–	–	–
	RAMPMODE[15:8]	–	–	–	–	–	–	–	–
	RAMPMODE[7:0]	–	–	–	–	–	–	RAMPMODE[1:0]	
0x08	MSLUT_ADDR[31:24]	–	–	–	–	–	–	–	–
	MSLUT_ADDR[23:16]	–	–	–	–	–	–	–	–
	MSLUT_ADDR[15:8]	–	–	–	–	–	–	–	–
	MSLUT_ADDR[7:0]	–	–	–	MSLUT_ADDR[4:0]				
0x09	MSLUT_DATA[31:24]	MSLUT_DATA[31:24]							
	MSLUT_DATA[23:16]	MSLUT_DATA[23:16]							
	MSLUT_DATA[15:8]	MSLUT_DATA[15:8]							
	MSLUT_DATA[7:0]	MSLUT_DATA[7:0]							
Position Compare									
0x10	X_COMPARE[31:24]	X_COMPARE[31:24]							
	X_COMPARE[23:16]	X_COMPARE[23:16]							
	X_COMPARE[15:8]	X_COMPARE[15:8]							
	X_COMPARE[7:0]	X_COMPARE[7:0]							
0x11	X_COMPARE_REPEAT[31:24]	–	–	–	–	–	–	–	–
	X_COMPARE_REPEAT[23:16]	X_COMPARE_REPEAT[23:16]							
	X_COMPARE_REPEAT[15:8]	X_COMPARE_REPEAT[15:8]							
	X_COMPARE_REPEAT[7:0]	X_COMPARE_REPEAT[7:0]							
Velocity Dependent Configuration Registers									
0x12	IHOLD_IRUN[31:24]	–	–	–	–	IRUNDELAY[3:0]			
	IHOLD_IRUN[23:16]	–	–	–	–	IHOLDDELAY[3:0]			
	IHOLD_IRUN[15:8]	–	–	–	IRUN[4:0]				
	IHOLD_IRUN[7:0]	–	–	–	IHOLD[4:0]				
0x13	TPOWERDOWN[31:24]	–	–	–	–	–	–	–	–
	TPOWERDOWN[23:16]	–	–	–	–	–	–	–	–
	TPOWERDOWN[15:8]	–	–	–	–	–	–	–	–
	TPOWERDOWN[7:0]	TPOWERDOWN[7:0]							
0x14	TSTEP[31:24]	–	–	–	–	–	–	–	–
	TSTEP[23:16]	–	–	–	–	TSTEP[19:16]			
	TSTEP[15:8]	TSTEP[15:8]							
	TSTEP[7:0]	TSTEP[7:0]							
0x15	TPWMTHRS[31:24]	–	–	–	–	–	–	–	–
	TPWMTHRS[23:16]	–	–	–	–	TPWMTHRS[19:16]			
	TPWMTHRS[15:8]	TPWMTHRS[15:8]							
	TPWMTHRS[7:0]	TPWMTHRS[7:0]							
0x16	TCOOLTHRS[31:24]	–	–	–	–	–	–	–	–
	TCOOLTHRS[23:16]	–	–	–	–	TCOOLTHRS[19:16]			

ADDRESS	NAME	MSB							LSB
	TCOOLTHRS[15:8]	TCOOLTHRS[15:8]							
	TCOOLTHRS[7:0]	TCOOLTHRS[7:0]							
0x17	THIGH[31:24]	–	–	–	–	–	–	–	–
	THIGH[23:16]	–	–	–	–	THIGH[19:16]			
	THIGH[15:8]	THIGH[15:8]							
	THIGH[7:0]	THIGH[7:0]							
Ramp Generator Registers									
0x18	XACTUAL[31:24]	XACTUAL[31:24]							
	XACTUAL[23:16]	XACTUAL[23:16]							
	XACTUAL[15:8]	XACTUAL[15:8]							
	XACTUAL[7:0]	XACTUAL[7:0]							
0x19	VACTUAL[31:24]	–	–	–	–	–	–	–	–
	VACTUAL[23:16]	VACTUAL[23:16]							
	VACTUAL[15:8]	VACTUAL[15:8]							
	VACTUAL[7:0]	VACTUAL[7:0]							
0x1A	AACTUAL[31:24]	–	–	–	–	–	–	–	–
	AACTUAL[23:16]	AACTUAL[23:16]							
	AACTUAL[15:8]	AACTUAL[15:8]							
	AACTUAL[7:0]	AACTUAL[7:0]							
0x1B	VSTART[31:24]	–	–	–	–	–	–	–	–
	VSTART[23:16]	–	–	–	–	–	–	VSTART[17:16]	
	VSTART[15:8]	VSTART[15:8]							
	VSTART[7:0]	VSTART[7:0]							
0x1C	A1[31:24]	–	–	–	–	–	–	–	–
	A1[23:16]	–	–	–	–	–	–	A1[17:16]	
	A1[15:8]	A1[15:8]							
	A1[7:0]	A1[7:0]							
0x1D	V1[31:24]	–	–	–	–	–	–	–	–
	V1[23:16]	–	–	–	–	V1[19:16]			
	V1[15:8]	V1[15:8]							
	V1[7:0]	V1[7:0]							
0x1E	A2[31:24]	–	–	–	–	–	–	–	–
	A2[23:16]	–	–	–	–	–	–	A2[17:16]	
	A2[15:8]	A2[15:8]							
	A2[7:0]	A2[7:0]							
0x1F	V2[31:24]	–	–	–	–	–	–	–	–
	V2[23:16]	–	–	–	–	V2[19:16]			
	V2[15:8]	V2[15:8]							
	V2[7:0]	V2[7:0]							
0x20	AMAX[31:24]	–	–	–	–	–	–	–	–
	AMAX[23:16]	–	–	–	–	–	–	AMAX[17:16]	
	AMAX[15:8]	AMAX[15:8]							

ADDRESS	NAME	MSB							LSB
	AMAX[7:0]	AMAX[7:0]							
0x21	VMAX[31:24]	–	–	–	–	–	–	–	–
	VMAX[23:16]	–	VMAX[22:16]						
	VMAX[15:8]	VMAX[15:8]							
	VMAX[7:0]	VMAX[7:0]							
0x22	DMAX[31:24]	–	–	–	–	–	–	–	–
	DMAX[23:16]	–	–	–	–	–	–	DMAX[17:16]	
	DMAX[15:8]	DMAX[15:8]							
	DMAX[7:0]	DMAX[7:0]							
0x23	D2[31:24]	–	–	–	–	–	–	–	–
	D2[23:16]	–	–	–	–	–	–	D2[17:16]	
	D2[15:8]	D2[15:8]							
	D2[7:0]	D2[7:0]							
0x24	D1[31:24]	–	–	–	–	–	–	–	–
	D1[23:16]	–	–	–	–	–	–	D1[17:16]	
	D1[15:8]	D1[15:8]							
	D1[7:0]	D1[7:0]							
0x25	VSTOP[31:24]	–	–	–	–	–	–	–	–
	VSTOP[23:16]	–	–	–	–	–	–	VSTOP[17:16]	
	VSTOP[15:8]	VSTOP[15:8]							
	VSTOP[7:0]	VSTOP[7:0]							
0x26	TVMAX[31:24]	–	–	–	–	–	–	–	–
	TVMAX[23:16]	–	–	–	–	–	–	–	–
	TVMAX[15:8]	TVMAX[15:8]							
	TVMAX[7:0]	TVMAX[7:0]							
0x27	TZEROWAIT[31:24]	–	–	–	–	–	–	–	–
	TZEROWAIT[23:16]	–	–	–	–	–	–	–	–
	TZEROWAIT[15:8]	TZEROWAIT[15:8]							
	TZEROWAIT[7:0]	TZEROWAIT[7:0]							
0x28	XTARGET[31:24]	XTARGET[31:24]							
	XTARGET[23:16]	XTARGET[23:16]							
	XTARGET[15:8]	XTARGET[15:8]							
	XTARGET[7:0]	XTARGET[7:0]							
Ramp Generator Driver Feature Control Registers									
0x29	VDCMIN[31:24]	–	–	–	–	–	–	–	–
	VDCMIN[23:16]	–	VDCMIN[14:8]						
	VDCMIN[15:8]	VDCMIN[7:0]							
	VDCMIN[7:0]	Reserved[7:0]							
0x2A	SW_MODE[31:24]	–	–	–	–	–	–	–	–
	SW_MODE[23:16]	–	–	–	–	–	–	–	–
	SW_MODE[15:8]	–	virtual_Step_enc	en_virtual_stop_r	en_virtual_stop_l	en_softstop	sg_stop	en_latch_encoder	latch_r_inactive

ADDRESS	NAME	MSB							LSB
	SW_MODE[7:0]	latch_r_active	latch_l_inactive	latch_l_active	swap_lr	pol_stop_r	pol_stop_l	stop_r_enable	stop_l_enable
0x2B	RAMP_STAT[31:24]	–	–	–	–	–	–	–	–
	RAMP_STAT[23:16]	–	–	–	–	–	–	–	–
	RAMP_STAT[15:8]	status_virtual_stop_r	status_virtual_stop_l	status_sg	second_move	t_zerowait_active	vzero	position_reached	velocity_reached
	RAMP_STAT[7:0]	event_pos_reached	event_stop_sg	event_stop_r	event_stop_l	status_latch_r	status_latch_l	status_stop_r	status_stop_l
0x2C	XLATCH[31:24]	XLATCH[31:24]							
	XLATCH[23:16]	XLATCH[23:16]							
	XLATCH[15:8]	XLATCH[15:8]							
	XLATCH[7:0]	XLATCH[7:0]							
0x2D	POSITION_P_CTRL[31:24]	–	–	–	en_tol_on_pos_reached	–	–	–	–
	POSITION_P_CTRL[23:16]	tolerance[7:0]							
	POSITION_P_CTRL[15:8]	–	–	–	–	–	–	P[9:8]	
	POSITION_P_CTRL[7:0]	P[7:0]							
Encoder Registers									
0x2E	X_ENC[31:24]	X_ENC[31:24]							
	X_ENC[23:16]	X_ENC[23:16]							
	X_ENC[15:8]	X_ENC[15:8]							
	X_ENC[7:0]	X_ENC[7:0]							
0x2F	ENCMODE[31:24]	–	–	BEMF_FILTER_SEL[1:0]		–	–	–	–
	ENCMODE[23:16]	BEMF_BLANK_TIME[7:0]							
	ENCMODE[15:8]	qsc_enc_en	bemf_hyst[2:0]			nBEMF_ABN_SEL	enc_sel_decimal	latch_x_act	clr_enc_x
	ENCMODE[7:0]	pos_neg_edge[1:0]		clr_once	clr_cont	ignore_AB	pol_N	pol_B	pol_A
0x30	ENC_CONST[31:24]	ENC_CONST[31:24]							
	ENC_CONST[23:16]	ENC_CONST[23:16]							
	ENC_CONST[15:8]	ENC_CONST[15:8]							
	ENC_CONST[7:0]	ENC_CONST[7:0]							
0x31	ENC_STATUS[31:24]	–	–	–	–	–	–	–	–
	ENC_STATUS[23:16]	–	–	–	–	–	–	–	–
	ENC_STATUS[15:8]	–	–	–	–	–	–	–	–
	ENC_STATUS[7:0]	–	–	–	–	–	–	deviation_warn	n_event
0x32	ENC_LATCH[31:24]	ENC_LATCH[31:24]							
	ENC_LATCH[23:16]	ENC_LATCH[23:16]							

ADDRESS	NAME	MSB							LSB
	ENC_LATCH[15:8]	ENC_LATCH[15:8]							
	ENC_LATCH[7:0]	ENC_LATCH[7:0]							
0x33	ENC_DEVIATION[31:24]	–	–	–	–	–	–	–	–
	ENC_DEVIATION[23:16]	–	–	–	–	ENC_DEVIATION[19:16]			
	ENC_DEVIATION[15:8]	ENC_DEVIATION[15:8]							
	ENC_DEVIATION[7:0]	ENC_DEVIATION[7:0]							
0x34	VIRTUAL_STOP_L[31:24]	VIRTUAL_STOP_L[31:24]							
	VIRTUAL_STOP_L[23:16]	VIRTUAL_STOP_L[23:16]							
	VIRTUAL_STOP_L[15:8]	VIRTUAL_STOP_L[15:8]							
	VIRTUAL_STOP_L[7:0]	VIRTUAL_STOP_L[7:0]							
0x35	VIRTUAL_STOP_R[31:24]	VIRTUAL_STOP_R[31:24]							
	VIRTUAL_STOP_R[23:16]	VIRTUAL_STOP_R[23:16]							
	VIRTUAL_STOP_R[15:8]	VIRTUAL_STOP_R[15:8]							
	VIRTUAL_STOP_R[7:0]	VIRTUAL_STOP_R[7:0]							
Motor Driver Registers									
0x36	MSCNT[31:24]	–	–	–	–	–	–	–	–
	MSCNT[23:16]	–	–	–	–	–	–	–	–
	MSCNT[15:8]	–	–	–	–	–	–	MSCNT[9:8]	
	MSCNT[7:0]	MSCNT[7:0]							
0x37	MSCURACT[31:24]	–	–	–	–	–	–	–	CUR_B[8]
	MSCURACT[23:16]	CUR_B[7:0]							
	MSCURACT[15:8]	–	–	–	–	–	–	–	CUR_A[8]
	MSCURACT[7:0]	CUR_A[7:0]							
0x38	CHOPCONF[31:24]	diss2vs	diss2g	dedge	intpol	MRES[3:0]			
	CHOPCONF[23:16]	TPFD[3:0]				vhighchm	vhighfs	–	TBL[1]
	CHOPCONF[15:8]	TBL[0]	chm	–	disfdcc	fd3	HEND_OFFSET[3:1]		
	CHOPCONF[7:0]	HEND_OFFSET[0]	HSTRT_TFD210[2:0]			TOFF[3:0]			
0x39	COOLCONF[31:24]	–	–	–	–	–	–	–	sfilt
	COOLCONF[23:16]	–	sgt[6:0]						
	COOLCONF[15:8]	seimin	sedn[1:0]		–	semax[3:0]			
	COOLCONF[7:0]	–	seup[1:0]		–	semin[3:0]			
0x3A	DCCTRL[31:24]	–	–	–	–	–	–	–	–
	DCCTRL[23:16]	DC_SG[7:0]							

ADDRESS	NAME	MSB							LSB
	DCCTRL[15:8]	–	–	–	–	–	–	DC_TIME[9:8]	
	DCCTRL[7:0]	DC_TIME[7:0]							
0x3B	DRV_STATUS[31:24]	stst	olb	ola	s2gb	s2ga	otpw	ot	stallguard
	DRV_STATUS[23:16]	–	–	–	CS_ACTUAL[4:0]				
	DRV_STATUS[15:8]	fsactive	stealth	s2vsb	s2vsa	–	–	SG_RESULT[9:8]	
	DRV_STATUS[7:0]	SG_RESULT[7:0]							
0x3C	PWMCONF[31:24]	PWM_LIM[3:0]				PWM_REG[3:0]			
	PWMCONF[23:16]	pwm_dis_reg_stst	pwm_meas_sd_enable	FREEWHEEL[1:0]		pwm_autograd	pwm_autoscale	PWM_FREQ[1:0]	
	PWMCONF[15:8]	PWM_GRAD[7:0]							
	PWMCONF[7:0]	PWM_OFS[7:0]							
0x3D	PWM_SCALE[31:24]	–	–	–	–	–	–	–	PWM_SCALE_AUTO[8]
	PWM_SCALE[23:16]	PWM_SCALE_AUTO[7:0]							
	PWM_SCALE[15:8]	–	–	–	–	–	–	PWM_SCALE_SUM[9:8]	
	PWM_SCALE[7:0]	PWM_SCALE_SUM[7:0]							
0x3E	PWM_AUTO[31:24]	–	–	–	–	–	–	–	–
	PWM_AUTO[23:16]	PWM_GRAD_AUTO[7:0]							
	PWM_AUTO[15:8]	–	–	–	–	–	–	–	–
	PWM_AUTO[7:0]	PWM_OFS_AUTO[7:0]							
0x3F	SG4_THRS[31:24]	–	–	–	–	–	–	–	–
	SG4_THRS[23:16]	–	–	–	–	–	–	–	–
	SG4_THRS[15:8]	–	–	–	–	–	–	SG_ANGLE_OFFSET	SG4_filt_en
	SG4_THRS[7:0]	SG4_THRS[7:0]							
0x40	SG4_RESULT[31:24]	–	–	–	–	–	–	–	–
	SG4_RESULT[23:16]	–	–	–	–	–	–	–	–
	SG4_RESULT[15:8]	–	–	–	–	–	–	SG4_RESULT[9:8]	
	SG4_RESULT[7:0]	SG4_RESULT[7:0]							
0x41	SG4_IND[31:24]	sg4_ind_3[7:0]							
	SG4_IND[23:16]	sg4_ind_2[7:0]							
	SG4_IND[15:8]	sg4_ind_1[7:0]							
	SG4_IND[7:0]	sg4_ind_0[7:0]							
Microstep Look-Up Table									
0x80	MSLUT0[31:24]	MSLUT_0[31:24]							
	MSLUT0[23:16]	MSLUT_0[23:16]							
	MSLUT0[15:8]	MSLUT_0[15:8]							
	MSLUT0[7:0]	MSLUT_0[7:0]							
0x81	MSLUT1[31:24]	MSLUT_1[31:24]							
	MSLUT1[23:16]	MSLUT_1[23:16]							

ADDRESS	NAME	MSB							LSB
	MSLUT1[15:8]	MSLUT_1[15:8]							
	MSLUT1[7:0]	MSLUT_1[7:0]							
0x82	MSLUT2[31:24]	MSLUT_2[31:24]							
	MSLUT2[23:16]	MSLUT_2[23:16]							
	MSLUT2[15:8]	MSLUT_2[15:8]							
	MSLUT2[7:0]	MSLUT_2[7:0]							
0x83	MSLUT3[31:24]	MSLUT_3[31:24]							
	MSLUT3[23:16]	MSLUT_3[23:16]							
	MSLUT3[15:8]	MSLUT_3[15:8]							
	MSLUT3[7:0]	MSLUT_3[7:0]							
0x84	MSLUT4[31:24]	MSLUT_4[31:24]							
	MSLUT4[23:16]	MSLUT_4[23:16]							
	MSLUT4[15:8]	MSLUT_4[15:8]							
	MSLUT4[7:0]	MSLUT_4[7:0]							
0x85	MSLUT5[31:24]	MSLUT_5[31:24]							
	MSLUT5[23:16]	MSLUT_5[23:16]							
	MSLUT5[15:8]	MSLUT_5[15:8]							
	MSLUT5[7:0]	MSLUT_5[7:0]							
0x86	MSLUT6[31:24]	MSLUT_6[31:24]							
	MSLUT6[23:16]	MSLUT_6[23:16]							
	MSLUT6[15:8]	MSLUT_6[15:8]							
	MSLUT6[7:0]	MSLUT_6[7:0]							
0x87	MSLUT7[31:24]	MSLUT_7[31:24]							
	MSLUT7[23:16]	MSLUT_7[23:16]							
	MSLUT7[15:8]	MSLUT_7[15:8]							
	MSLUT7[7:0]	MSLUT_7[7:0]							
0x88	MSLUT_START[31:24]	OFFSET_SIN90[7:0]							
	MSLUT_START[23:16]	START_SIN90[7:0]							
	MSLUT_START[15:8]	–	–	–	–	–	–	–	–
	MSLUT_START[7:0]	START_SIN[7:0]							
0x89	MSLUT_SEL[31:24]	X3[7:0]							
	MSLUT_SEL[23:16]	X2[7:0]							
	MSLUT_SEL[15:8]	X1[7:0]							
	MSLUT_SEL[7:0]	W3[1:0]		W2[1:0]		W1[1:0]		W0[1:0]	

Register Details

GCONF (0x0)

BIT	31	30	29	28	27	26	25	24
Field	–	diag1_sel_nStallIndex_xcomp	diag0_sel_nError_Ramp	–	–	–	–	–
Reset	–	0x0	0x0	–	–	–	–	–
Access Type	–	Write, Read	Write, Read	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	qsc_sts_ena	drv_enn	SD	direct_mode	stop_enable	small_hysteresis	diag1_poscomp_pushpull	diag0_int_pushpull
Reset	0x0	0x1	0x0	0x0	0x0	0x0	0x0	0x0
Access Type	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	diag1_index	diag1_stall_dir	diag0_stall_step	diag0_otpw	diag0_error	shaft	multistep_filt	en_pwm_mode
Reset	0x0	0x0	0x0	0x0	0x0	0x0	0x1	0x0
Access Type	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
diag1_sel_nStallIndex_xcomp	30	DIAG1 output configuration	0x0: DIAG1 provides selected information on stall or index pulse as configured 0x1: DIAG1 is signaling X_COMPARE = X_ACTUAL or direction of motor 0 motion controller as configured
diag0_sel_nError_Ramp	29	DIAG0 output configuration	0x0: DIAG0 provides selected information on the driver status 0x1: DIAG0 provides motion controller status information
qsc_sts_ena	15	Enable quiescent mode for low-power consumption. This disables the driver by resetting drv_enn to 1. Only enabled once the driver is in standstill. Check the qsc bit (0x4 [15]) to see whether the device is in quiescent mode.	0x0: disable quiescent mode 0x1: enable quiescent mode
drv_enn	14	Driver stage control	0x0: Enable driver 0x1: Disable driver

BITFIELD	BITS	DESCRIPTION	DECODE
SD	13	When set to 1 switches off internal ramp generator and use external S/D input.	0x0: Using internal motion controller 0x1: Switches off internal motion controller and using external STEP/DIR inputs
direct_mode	12	Enable direct motor phase current control through the serial interface	0x0: Normal operation 0x1: Motor coil currents and polarity directly programmed through the serial interface. Register XTARGET (0x28) specifies signed coil A current (bits 8:0) and coil B current (bits 24:16). In this mode, the current is scaled by the IHOLD setting (0x12). Velocity-based current regulation of StealthChop2 is not available in this mode. The automatic StealthChop2 current regulation only works for low stepper motor velocities.
stop_enable	11	Motor hard stop function enable	0x0: Normal operation 0x1: Emergency stop. ENC_A1 stops the sequencer when tied high (no steps are executed by the sequencer: motor goes to standstill state).
small_hysteresis	10	Hysteresis selection for step frequency comparison	0x0: Hysteresis for step frequency comparison is 1/16 0x1: Hysteresis for step frequency comparison is 1/32
diag1_poscomp_pushpull	9	DIAG1 output type configuration	0x0: DIAG1 is open collector output (active low) 0x1: Enable DIAG1 push-pull output (active low)
diag0_int_pushpull	8	DIAG0 output type configuration.	0x0: DIAG0 is open collector output (active low) 0x1: Enable DIAG0 push-pull output (active low)
diag1_index	7	DIAG1 output configuration	0x0: Disable DIAG1 active on index position 0x1: diag1_index Enable DIAG1 active on index position (microstep look-up table position 0)
diag1_stall_dir	6	DIAG1 output configuration depending on diag1_sel_nStallIndex_xcomp (0x0 [30])	0x0: diag1_sel_nStallIndex_xcomp = 0 disables motor stall signal at DIAG1 Motor stall not indicated at DIAG1 diag1_sel_nStallIndex_xcomp = 1 enables DIAG1 to output position compare signal DIAG1 outputs position compare signal 0x1: diag1_sel_nStallIndex_xcomp = 0 enables DIAG1 active on motor stall (set TCOOLTHRS before using this feature) Enable DIAG1 active on motor stall (set TCOOLTHRS before using this feature) diag1_sel_nStallIndex_xcomp = 1 enables DIAG1 as direction output for external STEP/DIR driver using the direction signal of ramp generator 0 Enable DIAG1 as DIR output for external STEP/DIR driver using DIR of ramp generator 0

BITFIELD	BITS	DESCRIPTION	DECODE
diag0_stall_step	5	DIAG0 output configuration depending on diag0_sel_nError_Ramp (0x0 [29])	0x0: diag0_sel_nError_Ramp = 0 disables motor stall signal at DIAG0 diag0_sel_nError_Ramp = 1 enables motion controller interrupt flags at DIAG0 0x1: diag0_sel_nError_Ramp = 0 enables DIAG0 active on motor stall diag0_sel_nError_Ramp = 1 enables DIAG0 as step output for external STEP/DIR driver using the direction signal of ramp generator 0
diag0_otpw	4	DIAG0 output configuration	0x0: Disable DIAG0 active on driver overtemperature prewarning 0x1: Enable DIAG0 active on driver overtemperature prewarning (otpw)
diag0_error	3	DIAG0 output configuration DIAG0 always shows the reset status, i.e., it is active-low during reset condition	0x0: Disable DIAG0 active on driver errors 0x1: Enable DIAG0 active on driver errors Over temperature (ot), short to GND (s2g)
shaft	2	Change motor direction/direction sign	0x0: Default motor direction 0x1: Inverse motor direction
multistep_filt	1	Enable step input filtering for StealthChop2	0x0: Step input filtering disabled 0x1: Enable step input filtering for StealthChop2 optimization with external step source (default = 1)
en_pwm_mode	0	Enable the StealthChop2 mode	0x0: No StealthChop2 0x1: StealthChop2 voltage PWM mode enabled (depending on velocity thresholds). Switch from off to on state while in stand-still and at IHOLD = nominal IRUN current, only.

GSTAT (0x1)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

BIT	7	6	5	4	3	2	1	0
Field	–	–	–	vm_uvlo	register_reset	uv_ldo	drv_err	reset
Reset	–	–	–	0b0	0b1	0b0	0b0	0b1
Access Type	–	–	–	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read

BITFIELD	BITS	DESCRIPTION	DECODE
vm_uvlo	4	V_S undervoltage detection flag	0: Normal operation 1: V_S undervoltage has occured since last reset
register_reset	3	Register-map default reset flag	0: Normal operation 1: Indicates that the register map has been reset. All registers have been cleared to reset values.
uv_ldo	2	LDO undervoltage condition flag	0: Normal operation 1: Indicates an undervoltage on the LDO. The driver is disabled during undervoltage. This flag is latched for information. This flag is also set when returning from quiescence mode due to resetting the LDO.
drv_err	1	Driver error flag	0: Normal operation 1: Indicates that the driver has been shut down due to overtemperature or short-circuit detection. Read DRV_STATUS for details. The flag can only be cleared when the temperature is below the limit again.
reset	0	Reset flag	0: Normal operation 1: Indicates that the IC has been reset

IFCNT (0x2)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	IFCNT[7:0]							
Reset	0x0							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
IFCNT	7:0	Interface transmission counter. This register becomes incremented with each successful UART interface write access. It can be read out to check the serial transmission for lost data. Read accesses do not change the content. Disabled in SPI operation. The counter wraps around from 255 to 0.

NODECONF (0x3)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	SENDDELAY[3:0]			
Reset	–	–	–	–	0x3			
Access Type	–	–	–	–	Write, Read			
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	NODEADDR[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
SENDDELAY	11:8	SWUART node configuration	0x0: 8-bit times (not allowed with multiple nodes) 0x1: 8-bit times (not allowed with multiple nodes) 0x2: 24-bit times 0x3: 24-bit times 0x4: 40-bit times 0x5: 40-bit times 0x6: 56-bit times 0x7: 56-bit times 0x8: 72-bit times 0x9: 72-bit times 0xA: 88-bit times 0xB: 88-bit times 0xC: 104-bit times 0xD: 104-bit times 0xE: 120-bit times 0xF: 120-bit times
NODEADDR	7:0	These 8 bits set the UART address of this node. The address becomes incremented by 1–6 as defined by AD0, AD1, and AD2. NODEADDR + ADx must be < 255.	

IOIN (0x4)

BIT	31	30	29	28	27	26	25	24
Field	VERSION[7:0]							
Reset								
Access Type	Read Only							

BIT	23	22	21	20	19	18	17	16
Field	–	–	SPI_FLT_SEL[1:0]		–	SILICON_RV[2:0]		
Reset	–	–	0x2		–	0b000		
Access Type	–	–	Write, Read		–	Read Only		

BIT	15	14	13	12	11	10	9	8
Field	qsc	–	OUTPUT	EXT_RES_DET	EXT_CLK	–	ADC_EN	ADC_TEMPERATURE[7]
Reset		–	0b1	0b0	0b0	–	0x1	
Access Type	Read Only	–	Write, Read	Read Only	Read Only	–	Write, Read	Read Only

BIT	7	6	5	4	3	2	1	0
Field	ADC_TEMPERATURE[6:0]							Reserved
Reset								
Access Type	Read Only							Read Only

BITFIELD	BITS	DESCRIPTION	DECODE
VERSION	31:24	0x60 (Identical numbers mean full digital compatibility.)	
SPI_FLT_SEL	21:20	Filter configuration for SPI interface	0x0: No filter is applied 0x1: No filter is applied 0x2: 10ns filter 0x3: 20ns filter
SILICON_RV	18:16	Silicon revision number	0x0: Engineering samples 0x1: Mass production silicon 0x2 0x3 0x4 0x5 0x6 0x7
qsc	15	Quiescence mode flag	0x0: Normal mode 0x1: Device is in quiescence mode
OUTPUT	13	Output polarity of SDO pin when UART is enabled by the USEL pin. The main purpose is to use SDO as the next address output (NAO) signal for daisy-chain addressing of multiple nodes. **Note:** Reset value is 1 for use as NAO to next IC in single-wire chain. No use in UART ring mode.	0x0: Low level, 0 0x1: High level, 1 (default reset value)

BITFIELD	BITS	DESCRIPTION	DECODE
EXT_RES_DET	12	External resistor detection	0x0: No external resistor detected 0x1: External resistor between IREF and GND
EXT_CLK	11	Clock-source indicator flag	0x0: The internal oscillator is used for generating the clock signal (12.5MHz) 0x1: The external oscillator is used for generating the clock signal
ADC_EN	9	ADC enable control	0x0: Disable ADC when temperature measurement is not needed to reduce power consumption 0x1: Enable temperature measurement of ADC (default)
ADC_TEMPERATURE	8:1	T = (2.03 x ADC_TEMPERATURE - 259)°C	
Reserved	0		

DRV_CONF (0x5)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	–	STANDSTILL_TIME[2:0]		
Reset	–	–	–	–	–	0x0		
Access Type	–	–	–	–	–	Write, Read		
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	–	–	en_em_disable	FSR_IREF[1:0]		FSR[1:0]	
Reset	–	–	–	0x0	0x0		0x0	
Access Type	–	–	–	Write, Read	Write, Read		Write, Read	

BITFIELD	BITS	DESCRIPTION	DECODE
STANDSTILL_TIME	18:16	Determines how many CLK cycles need to pass until the driver signals standstill. $2^{(20-STANDSTILL\_TIME)}$	0x0: 2^{20} 0x1: 2^{19} 0x2: 2^{18} 0x3: 2^{17} 0x4: 2^{16} 0x5: 2^{15} 0x6: 2^{14} 0x7: 2^{13}

BITFIELD	BITS	DESCRIPTION	DECODE
en_em_disable	4	Emergency driver off enable	0x0: Emergency stop disabled 0x1: Enables driver disable when REFL and REFR inputs are both triggered
FSR_IREF	3:2	Scales the reference current I_{REF} (based on R_{REF}). This is like scaling the external resistance R_{REF}. Use this together with FSR for finetuning the current scaling.	0x0: 25% I_{REF} 0x1: 50% I_{REF} 0x2: 75% I_{REF} 0x3: 100% I_{REF}
FSR	1:0	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 Electrical Characteristics table under output specifications and protection circuits.	0x0: K_{IFS} = 4.10, typ. $R_{DS(ON),LS}$ = 0.19Ω. 0x1: K_{IFS} = 8.16, typ. $R_{DS(ON),LS}$ = 0.1Ω. 0x2: K_{IFS} = 12.06, typ. $R_{DS(ON),LS}$ = 0.07Ω. 0x3: K_{IFS} = 16.00, typ. $R_{DS(ON),LS}$ = 0.055Ω.

GLOBAL SCALER (0x6)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	GLOBALSCALER_B[7:0]							
Reset	0x0							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	GLOBALSCALER_A[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
GLOBALSCALER_B	15:8	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 **Tip:** Values > 128 recommended for best results. Different values for GLOBALSCALER_B and GLOBALSCALER_A are only supported in SpreadCycle mode.
GLOBALSCALER_A	7:0	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 **Tip:** Values > 128 recommended for best results. Different values for GLOBALSCALER_B and GLOBALSCALER_A are only supported in SpreadCycle mode.

RAMPMODE (0x7)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	–	–	–	–	–	RAMPMODE[1:0]	
Reset	–	–	–	–	–	–	0x0	
Access Type	–	–	–	–	–	–	Write, Read	

BITFIELD	BITS	DESCRIPTION	DECODE
RAMPMODE	1:0	Motion controller ramping mode	0x0: Positioning mode (using all A, D, and V parameters) 0x1: Velocity mode to positive VMAX (using AMAX acceleration) 0x2: Velocity mode to negative VMAX (using AMAX acceleration) 0x3: Hold mode (velocity remains unchanged unless stop event occurs)

MSLUT_ADDR (0x8)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	–	–	–	MSLUT_ADDR[4:0]				
Reset	–	–	–	0x0				
Access Type	–	–	–	Write, Read				

BITFIELD	BITS	DESCRIPTION
MSLUT_ADDR	4:0	Selects which MSLUT register is accessible for RW operations in MSLUT_DATA (0x9) register

MSLUT_DATA (0x9)

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_DATA[31:24]							
Reset	0xAAAAB554							
Access Type	Write, Read							

BIT	23	22	21	20	19	18	17	16
Field	MSLUT_DATA[23:16]							
Reset	0xAAAAB554							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	MSLUT_DATA[15:8]							
Reset	0xAAAAB554							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	MSLUT_DATA[7:0]							
Reset	0xAAAAB554							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MSLUT_DATA	31:0	Select with MSLUT_ADDR 0x00: MSLUT_0 0x01: MSLUT_1 0x02: MSLUT_2 0x03: MSLUT_3 0x04: MSLUT_4 0x05: MSLUT_5 0x06: MSLUT_6 0x07: MSLUT_7 0x08: MSLUT_START 0x09: MSLUT_SEL

X_COMPARE (0x10)

BIT	31	30	29	28	27	26	25	24
Field	X_COMPARE[31:24]							
Reset	0xFFFFFFFF							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	X_COMPARE[23:16]							
Reset	0xFFFFFFFF							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	X_COMPARE[15:8]							
Reset	0xFFFFFFFF							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	X_COMPARE[7:0]							
Reset	0xFFFFFFFF							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
X_COMPARE	31:0	Position comparison register for motion controller position strobe. X_COMPARE is an absolute position. The position pulse is available on output DIAG1 when enabled in GCONF (0x0). XACTUAL = X_COMPARE: Output signal (position pulse) becomes 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.

X_COMPARE_REPEAT (0x11)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	X_COMPARE_REPEAT[23:16]							
Reset	0x0							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	X_COMPARE_REPEAT[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	X_COMPARE_REPEAT[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
X_COMPARE_REPEAT	23:0	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 microsteps have been made. Therefore, the X_COMPARE register defines the base position for the modulo calculation of X_COMPARE_REPEAT microsteps that have been made into positive or negative direction.

IHOLD_IRUN (0x12)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	IRUNDELAY[3:0]			
Reset	–	–	–	–	0x4			
Access Type	–	–	–	–	Write, Read			
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	IHOLDDELAY[3:0]			
Reset	–	–	–	–	0x1			
Access Type	–	–	–	–	Write, Read			
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	IRUN[4:0]				
Reset	–	–	–	0b11111				
Access Type	–	–	–	Write, Read				
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	–	–	IHOLD[4:0]				
Reset	–	–	–	0b01000				
Access Type	–	–	–	Write, Read				

BITFIELD	BITS	DESCRIPTION
IRUNDELAY	27:24	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 multiple of IRUNDELAY x 512 clocks
IHOLDDELAY	19:16	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 multiple of 2^{18} clocks

BITFIELD	BITS	DESCRIPTION
IRUN	12:8	Motor run current 0 = 1/32 1 = 2/32 to 31 = 32/32 **Tip:** A setting from 16 to 31 is preferred for best microstep performance.
IHOLD	4:0	Standstill current 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.

TPOWERDOWN (0x13)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	TPOWERDOWN[7:0]							
Reset	0x0A							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
TPOWERDOWN	7:0	TPOWERDOWN sets the delay time after standstill (stst) of the motor-to-motor current power-down. Time range is about 0 to 4 seconds. **Attention**: A minimum setting of 2 is required to allow automatic tuning of StealthChop PWM_OFFS_AUTO. Reset default = 10 (0 to ((2^8)-1)) x 2^{18} t_{CLK}

TSTEP (0x14)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	TSTEP[19:16]			
Reset	–	–	–	–	0x0			
Access Type	–	–	–	–	Read Only			
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	TSTEP[15:8]							
Reset	0x0							
Access Type	Read Only							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	TSTEP[7:0]							
Reset	0x0							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
TSTEP	19:0	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 stand still. 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 small_hysteresis flag 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 that 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: $(2^{24} / V) \leq TSTEP \leq 2^{24}/V - 1$. 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 do not represent the real motor velocity if it runs slower than the target velocity.

TPWMTHRS (0x15)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	TPWMTHRS[19:16]			
Reset	–	–	–	–	0x0			
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	TPWMTHRS[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	TPWMTHRS[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
TPWMTHRS	19:0	This is the upper velocity for StealthChop2 voltage PWM mode. TSTEP ≥ TPWMTHRS • StealthChop PWM mode is enabled, if configured • DcStep is disabled

TCOOLTHRS (0x16)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	TCOOLTHRS[19:16]			
Reset	–	–	–	–	0x0			
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	TCOOLTHRS[15:8]							
Reset	0x0							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	TCOOLTHRS[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
TCOOLTHRS	19:0	This is the lower threshold velocity for switching on the smart energy CoolStep and StallGuard2/4 features (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 become enabled when exceeding this velocity. In non-DcStep mode, it becomes 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 (DIAG0/1) is enabled, if configured

THIGH (0x17)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	THIGH[19:16]			
Reset	–	–	–	–	0x0			
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	THIGH[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	THIGH[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
THIGH	19:0	This velocity setting allows velocity-dependent switching into a different chopper mode and full stepping to maximize torque (unsigned). The stall detection feature becomes switched off for 2 to 3 electrical periods whenever passing 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 full-step mode and the stall detection becomes switched over to DcStep stall detection

XACTUAL (0x18)

BIT	31	30	29	28	27	26	25	24
Field	XACTUAL[31:24]							
Reset	0x0							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	XACTUAL[23:16]							
Reset	0x0							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	XACTUAL[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	XACTUAL[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
XACTUAL	31:0	Actual motor position (signed) **Tip:** 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 changing to hold mode.

VACTUAL (0x19)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	VACTUAL[23:16]							
Reset	0x0							
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	VACTUAL[15:8]							
Reset	0x0							
Access Type	Read Only							
BIT	7	6	5	4	3	2	1	0
Field	VACTUAL[7:0]							
Reset	0x0							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
VACTUAL	23:0	Actual motor velocity from ramp generator (signed) The sign matches the motion direction. A negative sign means motion to lower XACTUAL. $\pm(2^{23}) - 1$ [µsteps/t]

AACTUAL (0x1A)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	AACTUAL[23:16]							
Reset	0x000000							
Access Type	Read Only							

BIT	15	14	13	12	11	10	9	8
Field	AACTUAL[15:8]							
Reset	0x000000							
Access Type	Read Only							
BIT	7	6	5	4	3	2	1	0
Field	AACTUAL[7:0]							
Reset	0x000000							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
AACTUAL	23:0	Current acceleration used by the ramp generator 0 to (2^{24}) - 1 [µsteps/ta²]

VSTART (0x1B)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	VSTART[17:16]	
Reset	–	–	–	–	–	–	0x0	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	15	14	13	12	11	10	9	8
Field	VSTART[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	VSTART[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
VSTART	17:0	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]

A1 (0x1C)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	A1[17:16]	
Reset	–	–	–	–	–	–	0x0	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	15	14	13	12	11	10	9	8
Field	A1[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	A1[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
A1	17:0	First acceleration between VSTART and V1 (unsigned) 0 to (2^{18}) - 1 [µsteps/ta²]

V1 (0x1D)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	V1[19:16]			
Reset	–	–	–	–	0x0			
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	V1[15:8]							
Reset	0x0							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	V1[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
V1	19:0	First acceleration/deceleration phase threshold velocity (unsigned) 0: Disables A1 and D1 phase; use AMAX, DMAX only 0 to (2^{20}) - 1 [µsteps/t]

A2 (0x1E)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	A2[17:16]	
Reset	–	–	–	–	–	–	0b000000000000000000	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	15	14	13	12	11	10	9	8
Field	A2[15:8]							
Reset	0b000000000000000000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	A2[7:0]							
Reset	0b000000000000000000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
A2	17:0	Acceleration between V1 and V2 (unsigned) 0 to (2^{18}) - 1 [µsteps/ta²]

V2 (0x1F)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	V2[19:16]			
Reset	–	–	–	–	0x0			
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	V2[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	V2[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
V2	19:0	Velocity difference from VMAX for activation of acceleration segments with AMAX/2 and DMAX/2. 0: Disables AMAX/2 and DMAX/2 phase, use AMAX, DMAX only 0 to (2^{20}) - 1 [µsteps/t]

AMAX (0x20)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	AMAX[17:16]	
Reset	–	–	–	–	–	–	0x0	
Access Type	–	–	–	–	–	–	Write, Read	

BIT	15	14	13	12	11	10	9	8
Field	AMAX[15:8]							
Reset	0x0							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	AMAX[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
AMAX	17:0	Second acceleration between V2 and VMAX (unsigned) This is the acceleration and deceleration value for velocity mode. 0 to (2^{18}) - 1 [µsteps/ta²]

VMAX (0x21)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

BIT	23	22	21	20	19	18	17	16
Field	–	VMAX[22:16]						
Reset	–	0x0						
Access Type	–	Write, Read						

BIT	15	14	13	12	11	10	9	8
Field	VMAX[15:8]							
Reset	0x0							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	VMAX[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
VMAX	22:0	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]

DMAX (0x22)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	DMAX[17:16]	
Reset	–	–	–	–	–	–	0x0	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	15	14	13	12	11	10	9	8
Field	DMAX[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	DMAX[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
DMAX	17:0	Deceleration between VMAX and V2 (unsigned) 0 to (2^{18}) - 1 [µsteps/ta^2]

D2 (0x23)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	D2[17:16]	
Reset	–	–	–	–	–	–	0x10	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	15	14	13	12	11	10	9	8
Field	D2[15:8]							
Reset	0x10							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	D2[7:0]							
Reset	0x10							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
D2	17:0	Deceleration between V2 and V1 (unsigned) **Note:** Do not set 0 in positioning mode even if V2 = 0 1 to (2^{18}) - 1 [µsteps/ta²] Reset default = 10

D1 (0x24)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	D1[17:16]	
Reset	–	–	–	–	–	–	0d10	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	15	14	13	12	11	10	9	8
Field	D1[15:8]							
Reset	0d10							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	D1[7:0]							
Reset	0d10							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
D1	17:0	Deceleration between V1 and VSTOP (unsigned) **Note**: Do not set 0 in positioning mode even if V1 = 0 1 to (2^{16}) - 1 [µsteps/ta²] Reset default = 10

VSTOP (0x25)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	VSTOP[17:16]	
Reset	–	–	–	–	–	–	0d10	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	15	14	13	12	11	10	9	8
Field	VSTOP[15:8]							
Reset	0d10							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	VSTOP[7:0]							
Reset	0d10							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
VSTOP	17:0	Motor stop velocity (unsigned) **Tip:** Set VSTOP ≥ VSTART to allow positioning for short distances **Attention**: Do not set 0 in positioning mode, minimum 10 recommend. 1 to (2^{18}) - 1 [µsteps/t] Reset default = 10

TVMAX (0x26)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	TVMAX[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	TVMAX[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
TVMAX	15:0	Minimum time for constant velocity segments in multiple of 512 clocks. 0: Disables minimum duration setting for constant velocity phase > 0: A minimum duration of constant velocity is inserted between any change from acceleration to deceleration or vice versa to reduce jerk (0 to (2^{16} - 1)) x 512 t_{CLK}

TZEROWAIT (0x27)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

BIT	15	14	13	12	11	10	9	8
Field	TZEROWAIT[15:8]							
Reset	0x0							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	TZEROWAIT[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
TZEROWAIT	15:0	Defines the waiting time after ramping down to zero velocity before next movement or direction inversion can start. Time range is about 0 to 2 seconds. This setting avoids excess acceleration, e.g., from VSTOP to -VSTART. This setting is not used in velocity mode. (0 to (2^{16} - 1)) x 512 t_{CLK}

XTARGET (0x28)

BIT	31	30	29	28	27	26	25	24
Field	XTARGET[31:24]							
Reset	0x0							
Access Type	Write, Read							

BIT	23	22	21	20	19	18	17	16
Field	XTARGET[23:16]							
Reset	0x0							
Access Type	Write, Read							

BIT	15	14	13	12	11	10	9	8
Field	XTARGET[15:8]							
Reset	0x0							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	XTARGET[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
XTARGET	31:0	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. Therefore, the XTARGET value 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)

VDCMIN (0x29)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	VDCMIN[14:8]						
Reset	–	0b000000000000000						
Access Type	–	Write, Read						
BIT	15	14	13	12	11	10	9	8
Field	VDCMIN[7:0]							
Reset	0b000000000000000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	Reserved[7:0]							
Reset	0x00							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
VDCMIN	22:8	DcStep automatic commutation becomes 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. If the motor becomes 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 for comparison)
Reserved	7:0	reads always 0

SW_MODE (0x2A)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	virtual_Step_enc	en_virtual_stop_r	en_virtual_stop_l	en_softstop	sg_stop	en_latch_encoder	latch_r_inactive
Reset	–	0x0	0x0	0x0	0x0	0x0	0x0	0x0
Access Type	–	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read
BIT	7	6	5	4	3	2	1	0
Field	latch_r_active	latch_l_inactive	latch_l_active	swap_lr	pol_stop_r	pol_stop_l	stop_r_enable	stop_l_enable
Reset	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0
Access Type	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
virtual_Step_enc	14	Source for virtual stop (VIRTUAL_STOP_L and VIRTUAL_STOP_R)	0x0: Virtual stop relates to ramp generator position XACTUAL 0x1: Virtual stop relates to encoder position X_ENC

BITFIELD	BITS	DESCRIPTION	DECODE
en_virtual_stop_r	13	Automatic motor stop during active right virtual stop condition	0x0: Disabled 0x1: Enabled
en_virtual_stop_l	12	Enables automatic motor stop during active left virtual stop condition	0x0: Disabled 0x1: Enabled
en_softstop	11	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 becomes released. **Caution:** Do not use soft stop in combination with StallGuard2/4. Use soft stop for StealthChop2 operation **at high velocity**. In this case, a hard stop must be avoided, as it could result in severe overcurrent.	0x0: Hard stop 0x1: Soft stop
sg_stop	10	Enable stop by StallGuard2/4. Disable to release motor after stop event. Program TCOOLTHRS for velocity threshold. Do not enable during motor spin-up. Wait until the motor velocity exceeds a certain value where StallGuard2/4 delivers a stable result. This velocity threshold should be programmed using TCOOLTHRS.	0x0: Disabled 0x1: Enabled
en_latch_encoder	9	Latch encoder position to ENC_LATCH upon reference switch event	0x0: Disabled 0x1: Enabled
latch_r_inactive	8	Activates latching of the position to XLATCH upon an inactive edge on the right reference switch input REFR. The active level is defined by pol_stop_r.	0x0: Disabled 0x1: Enabled
latch_r_active	7	Activates latching of the position to XLATCH upon an active edge on the right reference switch input REFR. Activate latch_r_active to detect any spurious stop event by reading status_latch_r.	0x0: Disabled 0x1: Enabled
latch_l_inactive	6	Activates latching of the position to XLATCH upon an inactive edge on the left reference switch input REFL. The active level is defined by pol_stop_l.	0x0: Disabled 0x1: Enabled
latch_l_active	5	Activates latching of the position to XLATCH upon an active edge on the left reference switch input REFL. Activate latch_l_active to detect any spurious stop event by reading status_latch_l.	0x0: Disabled 0x1: Enabled
swap_lr	4	Swap the left and the right reference switch input REFL and REFR	0x0: Default assignments 0x1: REFL and REFR switched

BITFIELD	BITS	DESCRIPTION	DECODE
pol_stop_r	3	Sets the active polarity of the right reference switch input	0x0: Not inverted, active high: a high level on REFR stops the motor 0x1: Inverted, active low: a low level on REFR stops the motor
pol_stop_l	2	Sets the active polarity of the left reference switch input	0x0: Non-inverted, active high: a high level on REFL stops the motor 0x1: Inverted, active low: a low level on REFL stops the motor
stop_r_enable	1	Enables automatic motor stop during active right reference switch input. The motor restarts in case the stop switch becomes released.	0x0: Disabled 0x1: Enabled
stop_l_enable	0	Enables automatic motor stop during active left reference switch input. The motor restarts in case the stop switch becomes released.	0x0: Disabled 0x1: Enabled

RAMP_STAT (0x2B)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	status_virtual_stop_r	status_virtual_stop_l	status_sg	second_move	t_zerowait_active	vzero	position_reached	velocity_reached
Reset	0x1	0x1	0x0	0x0	0x0	0x0	0x0	0x0
Access Type	Read Only	Read Only	Read Only	Write 1 to Clear, Read	Read Only	Read Only	Read Only	Read Only
BIT	7	6	5	4	3	2	1	0
Field	event_pos_reached	event_stop_sg	event_stop_r	event_stop_l	status_latch_r	status_latch_l	status_stop_r	status_stop_l
Reset	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0
Access Type	Write 1 to Clear, Read	Write 1 to Clear, Read	Read Only	Read Only	Write 1 to Clear, Read	Write 1 to Clear, Read	Read Only	Read Only

BITFIELD	BITS	DESCRIPTION
status_virtual_stop_r	15	Virtual reference switch right status (1 = active)
status_virtual_stop_l	14	Virtual reference switch left status (1 = active)

BITFIELD	BITS	DESCRIPTION
status_sg	13	1: Signals an active StallGuard2 input from the CoolStep driver or from the 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.
second_move	12	1: Signals that the automatic ramp required moving back in the opposite direction e.g., due to on-the-fly parameter change. Write 1 to clear.
t_zerowait_active	11	1: Signals that TZEROWAIT is active after a motor stop. During this time, the motor is in standstill.
vzero	10	1: Signals that the actual velocity is 0
position_reached	9	1: Signals that the target position is reached. This flag becomes set while XACTUAL and XTARGET match.
velocity_reached	8	1: Signals that the target velocity is reached. This flag becomes set while VACTUAL and VMAX match.
event_pos_reached	7	1: Signals that the target position has been reached (position_reached becoming active). Write 1 to clear flag and interrupt condition. This bit is ORed to the interrupt output signal.
event_stop_sg	6	1: Signals an active StallGuard2/4 stop event. Resetting the register clears the stall condition and the motor may restart motion unless the motion controller has been stopped. Write 1 to clear flag and interrupt condition. This bit is ORed to the interrupt output signal.
event_stop_r	5	1: Active stop right condition due to stop switch or virtual stop. The stop condition and the 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 the stop function also clears the flag but the motor continues motion. This bit is ORed to the interrupt output signal.
event_stop_l	4	1: Active stop left condition due to stop switch or virtual stop. The stop condition and the 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 the stop function also clears the flag but the motor continues motion. This bit is ORed to the interrupt output signal.
status_latch_r	3	1: Latch right ready (enable position latching using SW_MODE settings latch_r_active or latch_r_inactive). Write 1 to clear.
status_latch_l	2	1: Latch left ready (enable position latching using SW_MODE settings latch_l_active or latch_l_inactive). Write 1 to clear.
status_stop_r	1	Reference switch right status (1 = active)
status_stop_l	0	Reference switch left status (1 = active)

XLATCH (0x2C)

BIT	31	30	29	28	27	26	25	24
Field	XLATCH[31:24]							
Reset	0x0							
Access Type	Read Only							
BIT	23	22	21	20	19	18	17	16
Field	XLATCH[23:16]							
Reset	0x0							
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	XLATCH[15:8]							
Reset	0x0							
Access Type	Read Only							
BIT	7	6	5	4	3	2	1	0
Field	XLATCH[7:0]							
Reset	0x0							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
XLATCH	31:0	Ramp generator latch position. Latches XACTUAL upon a programmable switch event (see SW_MODE description). The encoder position can be latched to ENC_LATCH together with XLATCH to allow consistency checks.

POSITION_P_CTRL (0x2D)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	en_tol_on_pos_reached	–	–	–	–
Reset	–	–	–	0x1	–	–	–	–
Access Type	–	–	–	Write, Read	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	tolerance[7:0]							
Reset	0x0							
Access Type	Write, Read							

BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	P[9:8]	
Reset	–	–	–	–	–	–	0x0	
Access Type	–	–	–	–	–	–	Write, Read	

BIT	7	6	5	4	3	2	1	0
Field	P[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
en_tol_on_pos_reached	28	Enable tolerance once the motion controller reached the target position	0x0: The P-controller ignores absolute errors below the tolerance 0x1: The P-controller ignores absolute errors below tolerance only when the pos_reached flag (RAMP_STAT[9]) of the motion controller is set to 1
tolerance	23:16	P-controller error tolerance setting Errors < tolerance are ignored	
P	9:0	Proportional parameter for position P-regulator P = 0 disables the P-regulator function P > 0 enables the P-regulator function UINT10.0, no fractional part	

X_ENC (0x2E)

BIT	31	30	29	28	27	26	25	24
Field	X_ENC[31:24]							
Reset	0x00000000							
Access Type	Write, Read							

BIT	23	22	21	20	19	18	17	16
Field	X_ENC[23:16]							
Reset	0x00000000							
Access Type	Write, Read							

BIT	15	14	13	12	11	10	9	8
Field	X_ENC[15:8]							
Reset	0x00000000							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	X_ENC[7:0]							
Reset	0x00000000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
X_ENC	31:0	Actual encoder position (signed)

ENCMODE (0x2F)

BIT	31	30	29	28	27	26	25	24
Field	–	–	BEMF_FILTER_SEL[1:0]		–	–	–	–
Reset	–	–	0x0		–	–	–	–
Access Type	–	–	Write, Read		–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	BEMF_BLANK_TIME[7:0]							
Reset	0x10							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	qsc_enc_en	bemf_hyst[2:0]			nBEMF_ABN_SEL	enc_sel_decimal	latch_x_act	clr_enc_x
Reset	0x1	0x0			0x0	0x0	0x0	0x0
Access Type	Write, Read	Write, Read			Write, Read	Write, Read	Write, Read	Write, Read
BIT	7	6	5	4	3	2	1	0
Field	pos_neg_edge[1:0]		clr_once	clr_cont	ignore_AB	pol_N	pol_B	pol_A
Reset	0x0		0x0	0x0	0x0	0x0	0x0	0x0
Access Type	Write, Read		Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

BITFIELD	BITS	DESCRIPTION	DECODE
BEMF_FILTER_SEL	29:28	Digital lowpass filter for filtering the back EMF signals of the decoder	0x0: $f_{CTOFF} = f_{CLK}/24600$ 0x1: $f_{CTOFF} = f_{CLK}/12300$ 0x2: f_{CTOFF} = fCLK/6150 0x3: $f_{CTOFF} = f_{CLK}/2870$
BEMF_BLANK_TIME	23:16	Blank time is BEMF_BLANK_TIME x ((2^{14}) - 1) x t_{CLK}	
qsc_enc_en	15	Encoder enable setting during quiescence mode	0x0: Incremental decoder/back EMF decoder is inactive during quiescence to reduce power consumption 0x1: Incremental decoder/back EMF decoder is active during quiescence (default)

BITFIELD	BITS	DESCRIPTION	DECODE
bemf_hyst	14:12	Set hysteresis for BEMF encoder comparator	0x0: 10mV 0x1: 25mV 0x2: 50mV 0x3: 75mV 0x4: 100mV 0x5: 150mV 0x6: 200mV 0x7: 250mV
nBEMF_ABN_SEL	11	Encoder source selector	0x0: Back-EMF full-step encoder 0x1: Incremental encoder interface
enc_sel_decimal	10	Encoder prescaler mode selection	0x0: Encoder prescaler divisor binary mode: Counts in ENC_CONST (fractional part)/65536 0x1: Encoder prescaler divisor decimal mode: Counts in ENC_CONST (fractional part)/10000
latch_x_act	9	Position latch configuration	0x0: Disabled 0x1: Also latch XACTUAL position together with X_ENC. Allows latching the ramp generator position upon an N channel event as selected by pos_edge and neg_edge.
clr_enc_x	8	Encoder latch configuration	0x0: Upon N event, X_ENC becomes latched to ENC_LATCH only 0x1: Latch and additionally clear encoder counter X_ENC at N event
pos_neg_edge	7:6	N channel event sensitivity	0x0: N channel event is active during an active N event level 0x1: N channel is valid upon active N event 0x2: N channel is valid upon inactive N event 0x3: N channel is valid upon active and inactive N event
clr_once	5	Position latch configuration	0x0: Disabled 0x1: Latch or latch and clear X_ENC on the next N event following the write access
clr_cont	4	Position latch configuration	0x0: Disabled 0x1: Always latch or latch and clear X_ENC upon an N event. (Once per revolution, it is recommended to combine this setting with edge sensitive N event.)
ignore_AB	3	N event configuration	0x0: An N event occurs only when polarities given by pol_N, pol_A, and pol_B match 0x1: Ignore A and B polarity for N channel event
pol_N	2	Defines active polarity of N	0x0: Low active 0x1: High active
pol_B	1	Required B polarity for an N channel event	0x0: Negative 0x1: Positive
pol_A	0	Required A polarity for an N channel event	0x0: Negative 0x1: Positive

ENC_CONST (0x30)

BIT	31	30	29	28	27	26	25	24
Field	ENC_CONST[31:24]							
Reset	0x00010000							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	ENC_CONST[23:16]							
Reset	0x00010000							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	ENC_CONST[15:8]							
Reset	0x00010000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	ENC_CONST[7:0]							
Reset	0x00010000							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
ENC_CONST	31:0	Accumulation constant (signed) 16-bit integer part, 16-bit fractional part X_ENC accumulates ±ENC_CONST/(2^{16} x X_ENC) (binary) or ±ENC_CONST/(10^4 x X_ENC) (decimal) ENCMODE bit enc_sel_decimal switches between decimal and binary setting. Use the sign to match rotation direction. Binary: ±[µsteps/2^{16}] ±(0 to 32767.999847) Decimal: ±(0.0 to 32767.9999) Reset default = 1.0 (= 65536)

ENC_STATUS (0x31)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	deviation_warn	n_event
Reset	–	–	–	–	–	–	0x0	0x0
Access Type	–	–	–	–	–	–	Write 1 to Clear, Read	Write 1 to Clear, Read

BITFIELD	BITS	DESCRIPTION	DECODE
deviation_warn	1	Encoder deviation warning flag	0x0: No warning 0x1: Deviation warning. Cannot be cleared while a warning still persists. Set ENC_DEVIATION to zero to disable.
n_event	0	Encoder N event flag Write 1 to clear	0x0: No event 0x1: Event detected.

ENC_LATCH (0x32)

BIT	31	30	29	28	27	26	25	24
Field	ENC_LATCH[31:24]							
Reset	0x0							
Access Type	Read Only							
BIT	23	22	21	20	19	18	17	16
Field	ENC_LATCH[23:16]							
Reset	0x0							
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	ENC_LATCH[15:8]							
Reset	0x0							
Access Type	Read Only							

BIT	7	6	5	4	3	2	1	0
Field	ENC_LATCH[7:0]							
Reset	0x0							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
ENC_LATCH	31:0	Encoder position X_ENC latched on N event

ENC_DEVIATION (0x33)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	ENC_DEVIATION[19:16]			
Reset	–	–	–	–	0x0			
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	ENC_DEVIATION[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	ENC_DEVIATION[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
ENC_DEVIATION	19:0	Maximum number of steps deviation between encoder counter and XACTUAL for deviation warning. For BEMF-encoder feature, deviation is compared to 0 instead of XACTUAL. Result in deviation_warn flag in ENC_STATUS register. 0 = Function is off.

VIRTUAL_STOP_L (0x34)

BIT	31	30	29	28	27	26	25	24
Field	VIRTUAL_STOP_L[31:24]							
Reset	0x0							
Access Type	Write, Read							

BIT	23	22	21	20	19	18	17	16
Field	VIRTUAL_STOP_L[23:16]							
Reset	0x0							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	VIRTUAL_STOP_L[15:8]							
Reset	0x0							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	VIRTUAL_STOP_L[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
VIRTUAL_STOP_L	31:0	Virtual stop switch based on encoder or internal position. A stop is raised based on the signed comparison: virtual_stop_enc = 1: X_ENC ≤ VIRTUAL_STOP_L virtual_stop_enc = 0: XACTUAL ≤ VIRTUAL_STOP_L -2^{31} to $(+2^{31} - 1)$

VIRTUAL_STOP_R (0x35)

BIT	31	30	29	28	27	26	25	24
Field	VIRTUAL_STOP_R[31:24]							
Reset	0x0							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	VIRTUAL_STOP_R[23:16]							
Reset	0x0							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	VIRTUAL_STOP_R[15:8]							
Reset	0x0							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	VIRTUAL_STOP_R[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
VIRTUAL_STOP_R	31:0	Virtual stop switch based on encoder or internal position. A stop is raised, based on the signed comparison: virtual_stop_enc = 1: X_ENC ≥ VIRTUAL_STOP_R virtual_stop_enc = 0: XACTUAL ≥ VIRTUAL_STOP_R -2^{31} to $(+2^{31} - 1)$

MSCNT (0x36)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	MSCNT[9:8]	
Reset	–	–	–	–	–	–	0b0000000000	
Access Type	–	–	–	–	–	–	Read Only	
BIT	7	6	5	4	3	2	1	0
Field	MSCNT[7:0]							
Reset	0b0000000000							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
MSCNT	9:0	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 re-initializing MSLUTSTART or MSLUT and MSLUTSEL.

MSCURACT (0x37)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	CUR_B[8]
Reset	–	–	–	–	–	–	–	0x0
Access Type	–	–	–	–	–	–	–	Read Only
BIT	23	22	21	20	19	18	17	16
Field	CUR_B[7:0]							
Reset	0x0							
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	CUR_A[8]
Reset	–	–	–	–	–	–	–	0xF7
Access Type	–	–	–	–	–	–	–	Read Only
BIT	7	6	5	4	3	2	1	0
Field	CUR_A[7:0]							
Reset	0xF7							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
CUR_B	24:16	Actual microstep current for motor phase B as read from MSLUT (not scaled by current)
CUR_A	8:0	Actual microstep current for motor phase A as read from MSLUT (not scaled by current)

CHOPCONF (0x38)

BIT	31	30	29	28	27	26	25	24
Field	diss2vs	diss2g	dedge	intpol	MRES[3:0]			
Reset	0x0	0x0	0x0	0x1	0x0			
Access Type	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read			
BIT	23	22	21	20	19	18	17	16
Field	TPFD[3:0]				vhighchm	vhighfs	–	TBL[1]
Reset	0x4				0x0	0x0	–	0b10
Access Type	Write, Read				Write, Read	Write, Read	–	Write, Read
BIT	15	14	13	12	11	10	9	8
Field	TBL[0]	chm	–	disfdcc	fd3	HEND_OFFSET[3:1]		
Reset	0b10	0x0	–	0x0	0x0	0x2		
Access Type	Write, Read	Write, Read	–	Write, Read	Write, Read	Write, Read		

BIT	7	6	5	4	3	2	1	0
Field	HEND_OFFSET[0]	HSTRT_TFD210[2:0]			TOFF[3:0]			
Reset	0x2	0b101			0x0			
Access Type	Write, Read	Write, Read			Write, Read			

BITFIELD	BITS	DESCRIPTION	DECODE
diss2vs	31	Short to supply protection disable	0x0: Short to V_S protection is on 0x1: Short to V_S protection is disabled
diss2g	30	Short to GND protection disable	0x0: Short to GND protection is on 0x1: Short to GND protection is disabled
dedge	29	Enable double edge step pulses	0x0: Disabled 0x1: Enable step impulse at each step edge to reduce step frequency requirement
intpol	28	Interpolation to 256 microsteps	0x0: No interpolation 0x1: The actual microstep resolution (MRES) becomes extrapolated to 256 microsteps for smoothest motor operation (useful for STEP/DIR operation only)
MRES	27:24	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]	0x0: Native 256 microstep setting. Normally use this setting with the internal motion controller. 0x1: 128 0x2: 64 0x3: 32 0x4: 16 0x5: 8 0x6: 4 0x7: 2, half step 0x8: 1, full step 0x9: Unused 0xA: Unused 0xB: Unused 0xC: Unused 0xD: Unused 0xE: Unused 0xF: Unused
TPFD	23:20	Passive fast decay time TPFD allows dampening of motor midrange resonances. Passive fast decay time setting controls duration of the fast decay phase inserted after bridge polarity change. N_{CLK} = 128 x TPFD %0000: Disable %0001 to %1111: 1 to 15	
vhighchm	19	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 becomes doubled during high-velocity operation to avoid doubling of the chopper frequency.	

BITFIELD	BITS	DESCRIPTION	DECODE
vhighfs	18	High-velocity full-step selection This bit enables switching to full step when VHIGH is exceeded. Switching takes place only at the 45° position. The full-step target current uses the current value from the microstep table at the 45° position.	
TBL	16:15	TBL/blank time select %00 to %11: Set comparator blank time to 16, 24, 36, or 54 clocks %01 or %10 is recommended for most applications	0x0: 16 0x1: 24 0x2: 36 0x3: 54
chm	14	Chopper mode	0x0: Standard mode (SpreadCycle) 0x1: 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.
disfdcc	12	Fast decay mode configuration (with chm = 1)	0x0: Enables current comparator usage for termination of the fast decay cycle 0x1: Disables current comparator usage for termination of the fast decay cycle
fd3	11	TFD[3] With chm = 1: MSB of fast decay time setting TFD	
HEND_OFFSET	10:7	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 which is 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 becomes added to the absolute value of each sine-wave entry.	

BITFIELD	BITS	DESCRIPTION	DECODE
HSTRT_TFD 210	6:4	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) **Note:** Effective HEND + HSTRT ≤ 16. Hysteresis decrement is done each 16 clocks. With chm = 1: TFD [2:0] fast decay time setting (MSB: fd3): %0000 to %1111: Fast decay time setting TFD with NCLK = 32 x TFD (%0000: slow decay only)	
TOFF	3:0	TOFF off time and driver enable Off time setting controls duration of slow decay phase N_{CLK} = 24 + 32 x TOFF %0000: Driver disable, all bridges off %0001: 1 (use only with TBL ≥ 2) %0010 to %1111: 2 to 15	

COOLCONF (0x39)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	sfilt
Reset	–	–	–	–	–	–	–	0x0
Access Type	–	–	–	–	–	–	–	Write, Read
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	sgt[6:0]						
Reset	–	0x0						
Access Type	–	Write, Read						
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	seimin	sedn[1:0]		–	semax[3:0]			
Reset	0x0	0x0		–	0x0			
Access Type	Write, Read	Write, Read		–	Write, Read			
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	seup[1:0]		–	semin[3:0]			
Reset	–	0x0		–	0x0			
Access Type	–	Write, Read		–	Write, Read			

BITFIELD	BITS	DESCRIPTION	DECODE
sfilt	24	StallGuard2/4 filter enable	0x0: Standard mode, high time resolution for StallGuard2/4 0x1: Filtered mode, StallGuard2/4 signal updated for each four full steps only to compensate for motor pole tolerances
sgt	22:16	StallGuard2/4 threshold value This signed value controls StallGuard2/4 level for stall output and sets the optimum measurement range for read-out. A lower value gives a higher sensitivity. Zero is the starting value working with most motors. -64 to +63: A higher value makes StallGuard2/4 less sensitive and requires more torque to indicate a stall.	
seimin	15	Minimum current for smart current control	0x0: 1/2 of current setting (IRUN) (when used with StealthChop2 requires IRUN ≥ 16) 0x1: 1/4 of current setting (IRUN) (when used with StealthChop2 requires IRUN ≥ 28)
sedn	14:13	Current down step speed	0x0: For each 32 StallGuard2/4 values, decrease by one 0x1: For each 8 StallGuard2/4 values, decrease by one 0x2: For each 2 StallGuard2/4 values, decrease by one 0x3: For each StallGuard2/4 values, decrease by one
semax	11:8	StallGuard2/4 hysteresis value for smart current control If the StallGuard2/4 result is equal to or above (SEMIN + SEMAX + 1) x 32, the motor current is decreased to save energy. %0000 to %1111: 0 to 15	
seup	6:5	Current up step width. Current increment steps per measured StallGuard2/4 value	0x0: 1 0x1: 2 0x2: 4 0x3: 8
semin	3:0	Minimum StallGuard2/4 value for smart current control and smart current enable. If the StallGuard2/4 result falls below SEMIN x 32, the motor current is increased to reduce motor load angle. %0000: Smart current control CoolStep off %0001 to %1111: 1 to 15	

DCCTRL (0x3A)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

BIT	23	22	21	20	19	18	17	16
Field	DC_SG[7:0]							
Reset	0x0							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	DC_TIME[9:8]	
Reset	–	–	–	–	–	–	0x0	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	7	6	5	4	3	2	1	0
Field	DC_TIME[7:0]							
Reset	0x0							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
DC_SG	23:16	Maximum PWM on-time for step loss detection using DcStep StallGuard2 in DcStep mode (DC_SG x 16/f_{CLK}). Set slightly higher than DC_TIME/16. 0 = Disable.
DC_TIME	9:0	Upper PWM on-time limit for commutation (DC_TIME x 1/f_{CLK}). Set slightly above effective blank time TBL.

DRV_STATUS (0x3B)

BIT	31	30	29	28	27	26	25	24
Field	stst	olb	ola	s2gb	s2ga	otpw	ot	stallguard
Reset	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0
Access Type	Read Only	Read Only	Read Only	Read Only	Read Only	Read Only	Read Only	Read Only
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	CS_ACTUAL[4:0]				
Reset	–	–	–	0x0				
Access Type	–	–	–	Read Only				
BIT	15	14	13	12	11	10	9	8
Field	fsactive	stealth	s2vsb	s2vsa	–	–	SG_RESULT[9:8]	
Reset	0x0	0x0	0x0	0x0	–	–	0x0	
Access Type	Read Only	Read Only	Read Only	Read Only	–	–	Read Only	
BIT	7	6	5	4	3	2	1	0
Field	SG_RESULT[7:0]							
Reset	0x0							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION	DECODE
stst	31	Standstill indicator This flag indicates motor standstill in each operation mode. This occurs depending on standstill timer configuration between 2^{13} to 2^{20} clock cycles after the last step pulse.	0x0: Motor moving 0x1: Motor in standstill
olb	30	Open-load indicator phase B	0x0: Normal operation 0x1: Open load detected on phase B This is only an informative flag. The driver takes no action based on it. False detection may occur in fast motion and standstill. Check during slow motion only.
ola	29	Open load indicator phase A	0x0: Normal operation 0x1: Open load detected on phase A This is only an informative flag. The driver takes no action based on it. False detection may occur in fast motion and standstill. Check during slow motion only.
s2gb	28	Short to ground indicator phase B	0x0: Normal operation 0x1: Short to GND detected on phase B. The driver becomes disabled. The flags stay active until the driver is disabled by software (TOFF = 0 or drv_enn = 1).
s2ga	27	Short to ground indicator phase A	0x0: Normal operation 0x1: Short to GND detected on phase A. The driver becomes disabled. The flags stay active until the driver is disabled by software (TOFF = 0 or drv_enn = 1).
otpw	26	Overtemperature pre-warning flag	0x0: Normal operation 0x1: Overtemperature prewarning threshold is exceeded. The overtemperature prewarning flag is common for both bridges.
ot	25	Overtemperature flag	0x0: Normal operation 0x1: Overtemperature limit has been reached. Drivers become disabled until otpw is also cleared due to cooling down of the IC. The overtemperature flag is common for both bridges.
stallguard	24	StallGuard2/4 status	0x0: Normal operation 0x1: Motor stall detected (SG_RESULT = 0) or DcStep stall in DcStep mode
CS_ACTUAL	20:16	Actual motor current/smart energy current Actual current control scaling for monitoring smart energy current scaling controlled by the settings in the COOLCONF register or for monitoring the function of the automatic current scaling	
fsactive	15	Full-step active indicator	0x0: Microstepping active 0x1: Indicates that the driver has switched to full step as defined by chopper mode settings and velocity thresholds

BITFIELD	BITS	DESCRIPTION	DECODE
stealth	14	StealthChop2 indicator	0x0: StealthChop2 not active 0x1: Driver operates in StealthChop2 mode
s2vsb	13	Short to supply indicator phase B	0x0: No error 0x1: Short to supply detected on phase B. The driver becomes disabled. The flags stay active until the driver is disabled by software (TOFF = 0 or drv_enn = 1).
s2vsa	12	Short to supply indicator phase A	0x0: No error 0x1: Short to supply detected on phase A. The driver becomes disabled. The flags stay active until the driver is disabled by software (TOFF = 0 or drv_enn = 1).
SG_RESULT	9:0	StallGuard2/4 result with respect to PWM on-time for coil A in standstill for motor temperature detection. Mechanical load measurement: The StallGuard2/4 result gives a means 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 a lower limit. It is not applicable in StealthChop2 mode. StallGuard2/4 works best with microstep operation or DcStep. Temperature measurement: In standstill, no StallGuard2/4 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.	

PWMCONF (0x3C)

BIT	31	30	29	28	27	26	25	24
Field	PWM_LIM[3:0]				PWM_REG[3:0]			
Reset	0xC				0x4			
Access Type	Write, Read				Write, Read			
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	pwm_dis_reg_stst	pwm_meas_sd_enable	FREEWHEEL[1:0]		pwm_autograd	pwm_autoscale	PWM_FREQ[1:0]	
Reset	0b0	0b0	0x0		0x1	0x1	0x0	
Access Type	Write, Read	Write, Read	Write, Read		Write, Read	Write, Read	Write, Read	

BIT	15	14	13	12	11	10	9	8
Field	PWM_GRAD[7:0]							
Reset	0x0							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	PWM_OFS[7:0]							
Reset	0x1D							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
PWM_LIM	31:28	PWM automatic scale amplitude limit when switching on. Limit for PWM_SCALE_AUTO when switching back from SpreadCycle to StealthChop2. 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 StealthChop2. It does not limit PWM_GRAD or PWM_GRAD_AUTO offset. (Default = 12)	
PWM_REG	27:24	Regulation loop gradient. User-defined maximum PWM amplitude change per half wave when using pwm_autoscale = 1.	0x0: Do not use 0x1: 0.5 increments (slowest regulation) 0x2: 1 increment 0x3: 1.5 increments 0x4: 2 increments (reset default) 0x5: 2.5 increments 0x6: 3 increments 0x7: 3.5 increments 0x8: 4 increments 0x9: 4.5 increments 0xA: 5 increments 0xB: 5.5 increments 0xC: 6 increments 0xD: 6.5 increments 0xE: 7 increments 0xF: 7.5 increments (fastest regulation)
pwm_dis_reg_stst	23	Standstill current regulation control	0x0: Current regulation active 0x1: Disable current regulation when motor is in standstill and current is reduced (less than IRUN). This option eliminates any regulation noise during standstill.
pwm_meas_sd_enable	22	Slow decay phase low-side current measurement control	0x0: Slow decay low-side measurement disabled (default) 0x1: Uses slow decay phases on low side to measure the motor current to reduce the lower current limit
FREEWHEEL	21:20	Standstill mode configuration. Standstill option when motor current setting is zero (I_HOLD = 0).	0x0: Normal operation 0x1: Freewheeling 0x2: Coil shorted using LS drivers 0x3: Coil shorted using HS drivers

BITFIELD	BITS	DESCRIPTION	DECODE
pwm_autograd	19	PWM automatic gradient adaptation	0x0: Fixed value for PWM_GRAD (PWM_GRAD_AUTO = PWM_GRAD) 0x1: Automatic tuning (only with pwm_autoscale = 10 (reset default) PWM_GRAD_AUTO is initialized with PWM_GRAD while pwm_autograd = 0 and becomes optimized automatically during motion. **Preconditions:** 1. PWM_OFS_AUTO has been 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 < 1 2. Motor running and 1.5 x PWM_OFS_AUTO < PWM_SCALE_SUM < 4 x PWM_OFS_AUTO and PWM_SCALE_SUM < 255 Time required for tuning PWM_GRAD_AUTO: Approximately 8 full steps per change of ±1. Also enables use of reduced chopper frequency for tuning PWM_OFS_AUTO.
pwm_autoscale	18	PWM automatic amplitude scaling	0x0: 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 x ((CS_ACTUAL + 1)/32) + PWM_GRAD x 256/TSTEP 0x1: Enable automatic current control (reset default)
PWM_FREQ	17:16	PWM frequency selection	0x0: f_{PWM} = 2/1024 f_{CLK} (reset default) 0x1: f_{PWM} = 2/683 f_{CLK} 0x2: f_{PWM} = 2/512 f_{CLK} 0x3: f_{PWM} = 2/410 f_{CLK}
PWM_GRAD	15:8	Velocity-dependent gradient for PWM amplitude: PWM_GRAD x 256/TSTEP This value is added to PWM_AMPL to compensate for the velocity-dependent motor back-EMF. Use PWM_GRAD as the 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 StealthChop2 when finished. After initial tuning, the required initial value can be read out from PWM_GRAD_AUTO.	

BITFIELD	BITS	DESCRIPTION	DECODE
PWM_OFS	7:0	User-defined PWM amplitude offset (0 to 255) related to full motor current (CS_ACTUAL = 31) in standstill. Use PWM_OFS as the 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 StealthChop2 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 i.e., 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).	

PWM_SCALE (0x3D)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	PWM_SCALE_AUTO[8]
Reset	–	–	–	–	–	–	–	0b000000000
Access Type	–	–	–	–	–	–	–	Read Only
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	PWM_SCALE_AUTO[7:0]							
Reset	0b000000000							
Access Type	Read Only							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	–	–	PWM_SCALE_SUM[9:8]	
Reset	–	–	–	–	–	–	0b0000000000	
Access Type	–	–	–	–	–	–	Read Only	

BIT	7	6	5	4	3	2	1	0
Field	PWM_SCALE_SUM[7:0]							
Reset	0b0000000000							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
PWM_SCALE_AUTO	24:16	Automatically determined current scaling value
PWM_SCALE_SUM	9:0	[0 to 1023: 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.

PWM_AUTO (0x3E)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	PWM_GRAD_AUTO[7:0]							
Reset	0x00							
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	PWM_OFS_AUTO[7:0]							
Reset	0x0							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
PWM_GRAD_AUTO	23:16	Automatically determined gradient value
PWM_OFS_AUTO	7:0	Automatically determined offset value

SG4_THRS (0x3F)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	SG_ANGLE_OFFSET	SG4_filt_en
Reset	–	–	–	–	–	–	0b1	0b0
Access Type	–	–	–	–	–	–	Write, Read	Write, Read
BIT	7	6	5	4	3	2	1	0
Field	SG4_THRS[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION	DECODE
SG_ANGLE_OFFSET	9	Control of automatic phase shift compensation based on StallGuard4 when switching from StealthChop2 to SpreadCycle controlled by TPWMTHRS	0x0: Disabled 0x1: Enabled
SG4_filt_en	8	SG4 filter control	0x0: Disable SG4 filter 0x1: Enable SG4 filter
SG4_THRS	7:0	Detection threshold for stall. The StallGuard4 value SG4_RESULT is compared to double this threshold. A stall is signaled with SG_RESULT ≤ SG4_THRS.	

SG4_RESULT (0x40)

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	SG4_RESULT[9:8]	
Reset	–	–	–	–	–	–	0x0	
Access Type	–	–	–	–	–	–	Read Only	
BIT	7	6	5	4	3	2	1	0
Field	SG4_RESULT[7:0]							
Reset	0x0							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
SG4_RESULT	9:0	StallGuard result for StallGuard4,only. SG4_RESULT becomes updated with each full step, independent of TCOOLTHRS and SG4THRS. A higher value signals a lower motor load and more torque headroom. Intended for StealthChop2 mode only. Bits 9 and 0 always show 0. Scaling to 10 bit is for compatibility to StallGuard2.

SG4_IND (0x41)

BIT	31	30	29	28	27	26	25	24
Field	sg4_ind_3[7:0]							
Reset	0x00							
Access Type	Read Only							
BIT	23	22	21	20	19	18	17	16
Field	sg4_ind_2[7:0]							
Reset	0x00							
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	sg4_ind_1[7:0]							
Reset	0x00							
Access Type	Read Only							
BIT	7	6	5	4	3	2	1	0
Field	sg4_ind_0[7:0]							
Reset	0x00							
Access Type	Read Only							

BITFIELD	BITS	DESCRIPTION
sg4_ind_3	31:24	When SG4_filt_en = 1, displays SG4 measurement 3 used as filter input
sg4_ind_2	23:16	When SG4_filt_en = 1, displays SG4 measurement 2 used as filter input
sg4_ind_1	15:8	When SG4_filt_en = 1, displays SG4 measurement 1 used as filter input
sg4_ind_0	7:0	Displays SG4 measurement. When SG4_filt_en = 1, displays SG4 measurement 0 used as filter input.

MSLUT0 (0x80)

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_0[31:24]							
Reset	0xAAAAB554							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	MSLUT_0[23:16]							
Reset	0xAAAAB554							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	MSLUT_0[15:8]							
Reset	0xAAAAB554							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	MSLUT_0[7:0]							
Reset	0xAAAAB554							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MSLUT_0	31:0	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.

MSLUT1 (0x81)

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_1[31:24]							
Reset	0x4A9554AA							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	MSLUT_1[23:16]							
Reset	0x4A9554AA							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	MSLUT_1[15:8]							
Reset	0x4A9554AA							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	MSLUT_1[7:0]							
Reset	0x4A9554AA							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MSLUT_1	31:0	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.

MSLUT2 (0x82)

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_2[31:24]							
Reset	0x24492929							
Access Type	Write, Read							

BIT	23	22	21	20	19	18	17	16
Field	MSLUT_2[23:16]							
Reset	0x24492929							
Access Type	Write, Read							

BIT	15	14	13	12	11	10	9	8
Field	MSLUT_2[15:8]							
Reset	0x24492929							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	MSLUT_2[7:0]							
Reset	0x24492929							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MSLUT_2	31:0	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.

MSLUT3 (0x83)

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_3[31:24]							
Reset	0x10104222							
Access Type	Write, Read							

BIT	23	22	21	20	19	18	17	16
Field	MSLUT_3[23:16]							
Reset	0x10104222							
Access Type	Write, Read							

BIT	15	14	13	12	11	10	9	8
Field	MSLUT_3[15:8]							
Reset	0x10104222							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	MSLUT_3[7:0]							
Reset	0x10104222							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MSLUT_3	31:0	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.

MSLUT4 (0x84)

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_4[31:24]							
Reset	0xFBFFFFFF							
Access Type	Write, Read							

BIT	23	22	21	20	19	18	17	16
Field	MSLUT_4[23:16]							
Reset	0xFBFFFFFF							
Access Type	Write, Read							

BIT	15	14	13	12	11	10	9	8
Field	MSLUT_4[15:8]							
Reset	0xFBFFFFFF							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	MSLUT_4[7:0]							
Reset	0xFBFFFFFF							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MSLUT_4	31:0	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.

MSLUT5 (0x85)

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_5[31:24]							
Reset	0xB5BB777D							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	MSLUT_5[23:16]							
Reset	0xB5BB777D							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	MSLUT_5[15:8]							
Reset	0xB5BB777D							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	MSLUT_5[7:0]							
Reset	0xB5BB777D							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MSLUT_5	31:0	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.

MSLUT6 (0x86)

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_6[31:24]							
Reset	0x49295556							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	MSLUT_6[23:16]							
Reset	0x49295556							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	MSLUT_6[15:8]							
Reset	0x49295556							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	MSLUT_6[7:0]							
Reset	0x49295556							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MSLUT_6	31:0	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.

MSLUT7 (0x87)

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_7[31:24]							
Reset	0x00404222							
Access Type	Write, Read							
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	MSLUT_7[23:16]							
Reset	0x00404222							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	MSLUT_7[15:8]							
Reset	0x00404222							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	MSLUT_7[7:0]							
Reset	0x00404222							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
MSLUT_7	31:0	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.

MSLUT_START (0x88)

BIT	31	30	29	28	27	26	25	24
Field	OFFSET_SIN90[7:0]							
Reset	0x00							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	START_SIN90[7:0]							
Reset	0xF7							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	START_SIN[7:0]							
Reset	0x00							
Access Type	Write, Read							

BITFIELD	BITS	DESCRIPTION
OFFSET_SIN90	31:24	Signed offset for CUR_B ±127 microsteps. Adapt START_SIN90 to match the sine-wave table at position 0.
START_SIN90	23:16	START_SIN90 gives the absolute current for microstep table entry at positions 256
START_SIN	7:0	START_SIN gives the absolute current at microstep table entry 0

MSLUT_SEL (0x89)

BIT	31	30	29	28	27	26	25	24
Field	X3[7:0]							
Reset	0xFF							
Access Type	Write, Read							
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	X2[7:0]							
Reset	0xFF							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	X1[7:0]							
Reset	0x80							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	W3[1:0]		W2[1:0]		W1[1:0]		W0[1:0]	
Reset	0b01		0b01		0b01		0b10	
Access Type	Write, Read		Write, Read		Write, Read		Write, Read	

BITFIELD	BITS	DESCRIPTION
X3	31:24	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 defined response, the values must satisfy: 0 < X1 < X2 < X3
X2	23:16	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 defined response, the values must satisfy: 0 < X1 < X2 < X3

BITFIELD	BITS	DESCRIPTION
X1	15:8	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 defined response, the values must satisfy: 0 < X1 < X2 < X3
W3	7:6	LUT width select from ofs(X3) to ofs255. 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
W2	5:4	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
W1	3:2	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
W0	1:0	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

Typical Application Circuits

Standard Application Circuit

The standard application circuit uses a minimum set of additional components as shown in Figure 57. Use low ESR capacitors for filtering the power supply V_S and depend on the actual supply voltage used. The capacitors need to handle the current ripple caused by 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 3.3V LDO.

Place all 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 TMC5271 requires a symmetrical board layout.

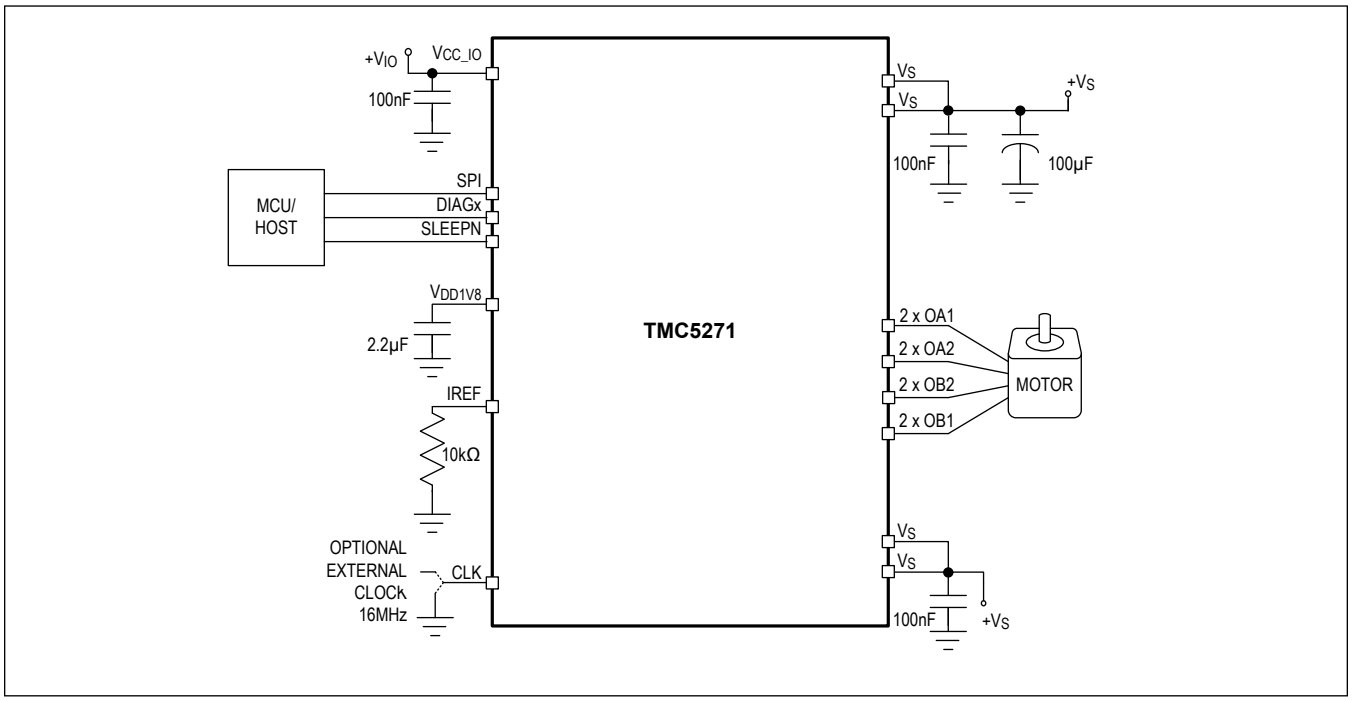


Figure 57. Standard Application Circuit of the TMC5271

All TMC5271 signals are referenced to their respective GND. Directly connect all 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 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 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

Typical Application Circuits (continued)

high currents into the motor connector terminals.

Since the TMC5271 is a very small component and targets low-voltage operation, the use of additional external protection circuity should be carefully planned to avoid unnecessarily inflating the BOM and required board space.

A simple scheme is shown in Figure 58 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 high-frequency noise injected from digital parts of the application PCB circuitry and thus reduce electromagnetic emission.

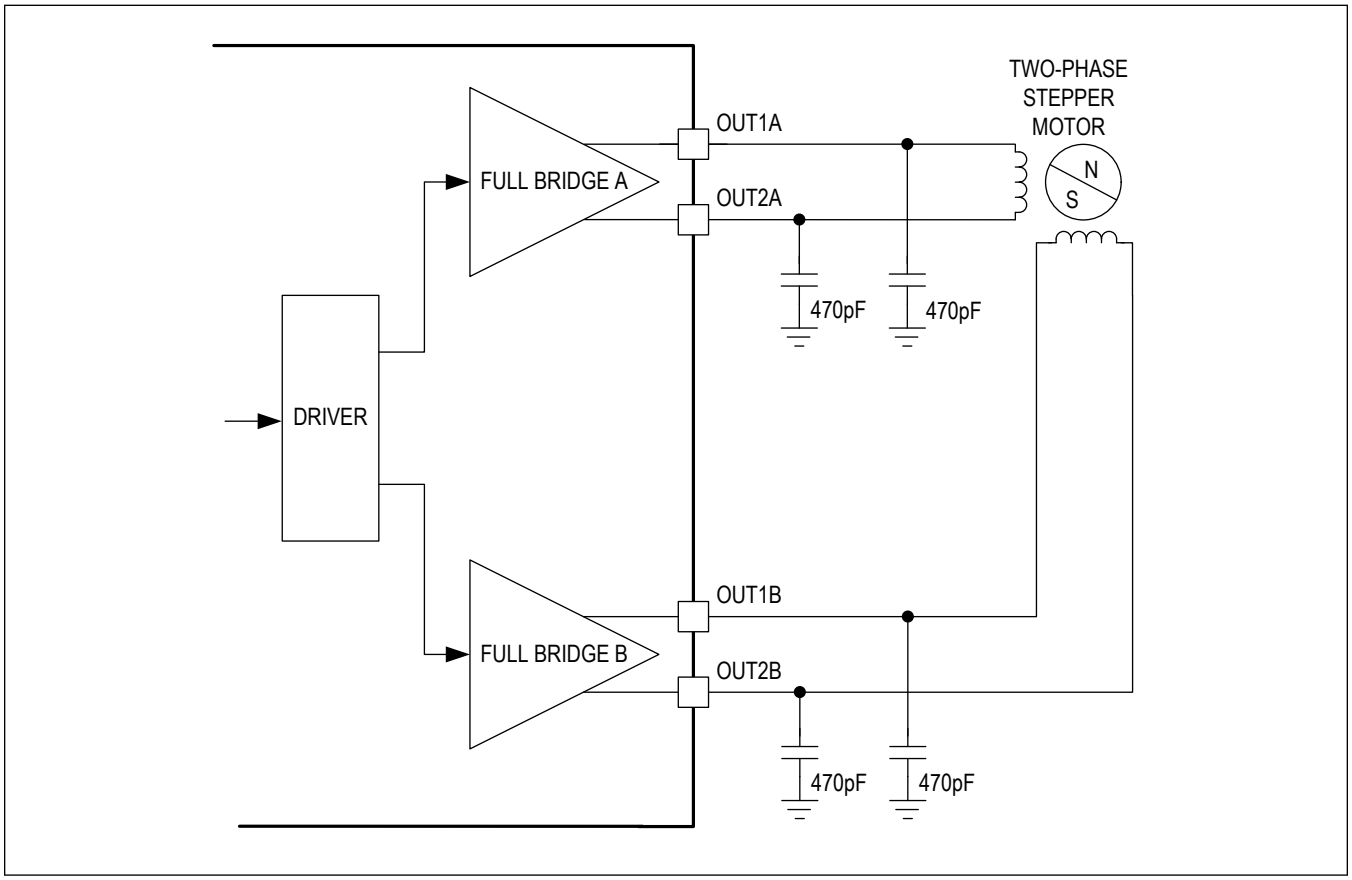


Figure 58. Simple ESD Enhancement

A more detailed scheme is shown in Figure 59 and uses LC filters to decouple the driver outputs from the motor connector. Varistors V1 and V2 between the coil terminals eliminate coil overvoltage caused by live plugging. Another option is to protect all 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 need to be selected accordingly.

Typical Application Circuits (continued)

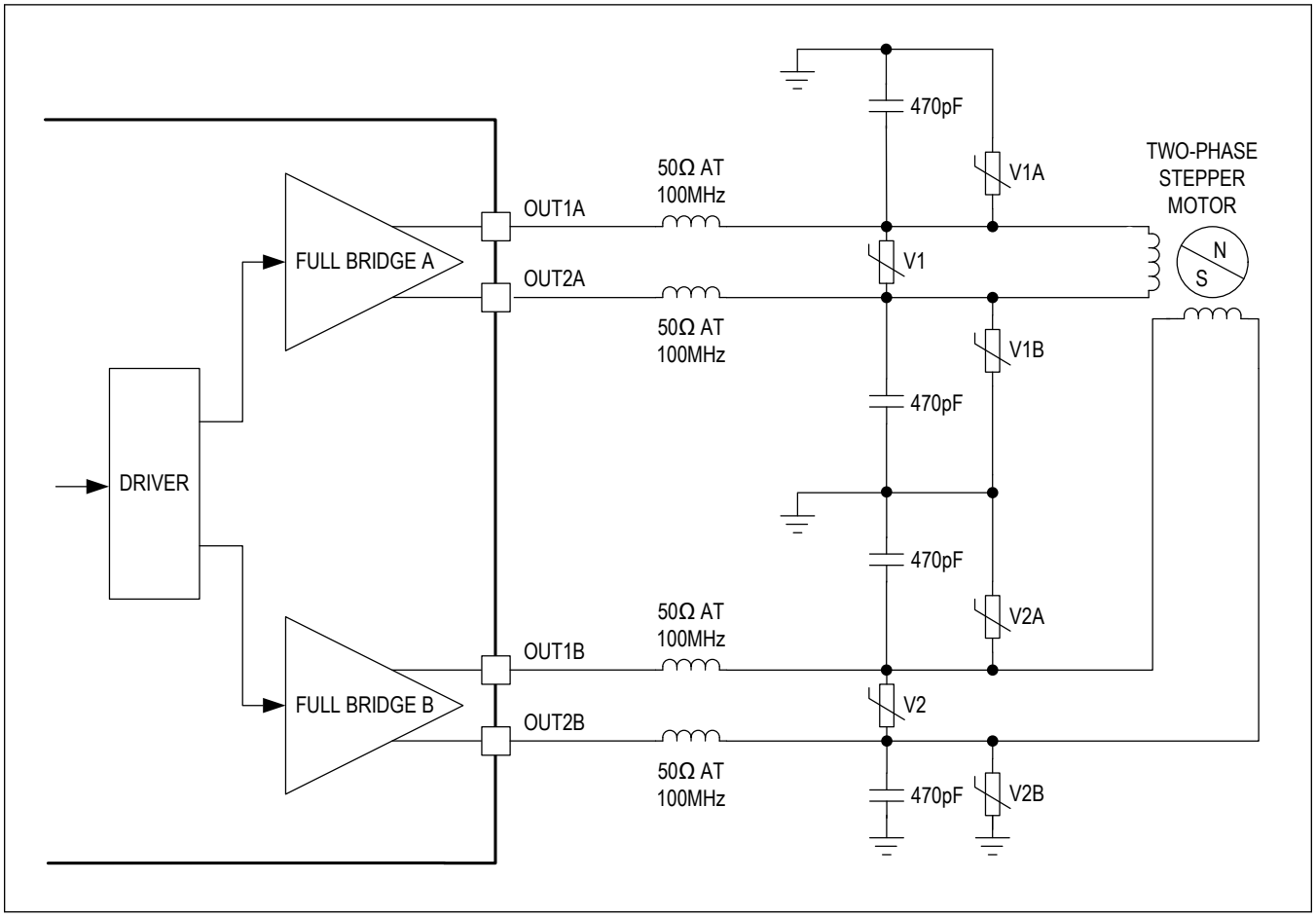


Figure 59. Extended Motor Output Protection

Ordering Information

PART NUMBER	TEMPERATURE RANGE	PIN-PACKAGE
TMC5271AWX+	-40°C to +125°C	36 WLCSP (2.97mm x 3.13mm)
TMC5271AWX+T	-40°C to +125°C	36 WLCSP (2.97mm x 3.13mm)

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

T = Tape and reel.

Revision History

REVISION NUMBER	REVISION DATE	DESCRIPTION	PAGES CHANGED
0	6/23	Initial release	—

StealthChop2 is a trademark of Maxim Integrated Products, Inc.
MicroPlyer is a trademark of Maxim Integrated Products, Inc.
SpreadCycle is a trademark of Maxim Integrated Products, Inc.
StallGuard2 is a trademark of Maxim Integrated Products, Inc.
StallGuard4 is a trademark of Maxim Integrated Products, Inc.
CoolStep is a registered trademark of Maxim Integrated Products, Inc.
StealthChop is a registered trademark of Maxim Integrated Products, Inc.
StallGuard is a registered trademark of Maxim Integrated Products, Inc.
DcStep is a trademark of Maxim Integrated Products, Inc.