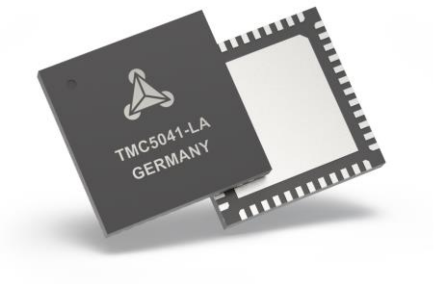
TMC5041 DATASHEET

Dual controller/driver for up to two 2-phase bipolar stepper motors. StealthChop™ no-noise stepper operation. Integrated motion controller with SPI interface.



APPLICATIONS

CCTV, Security
Office Automation
Antenna Positioning
Battery powered applications
ATM, Cash recycler, POS
Lab Automation
Liquid Handling
Medical
Printer and Scanner
Pumps and Valves

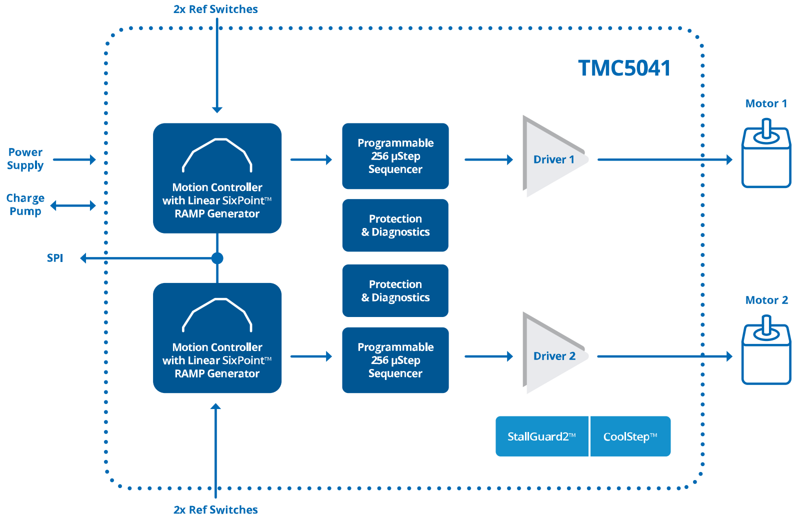
FEATURES AND BENEFITS

Two 2-phase stepper motors
Drive Capability up to 2x 1.1A coil current (2x 1.5A peak)
Motion Controller with **SixPoint™** ramp
Voltage Range 4.75… 26V DC
SPI Interface
2x **Ref.-Switch input** per axis
Highest Resolution up to 256 microsteps per full step
StealthChop™ for extremely quiet operation and smooth motion
SpreadCycle™ highly dynamic motor control chopper
StallGuard2™ high precision sensorless motor load detection
CoolStep™ current control for energy savings up to 75%
Passive Braking and freewheeling mode
Full Protection & Diagnostics
Compact Size 7x7mm^2 QFN48 package

DESCRIPTION

The TMC5041 is a dual high performance stepper motor controller and driver IC with serial communication interfaces. It combines flexible ramp generators for automatic target positioning with industries' most advanced stepper motor drivers. Based on TRINAMICs sophisticated StealthChop chopper, the driver ensures noiseless operation, maximum efficiency, and best motor torque. High integration, energy saving functions like CoolStep, and a small form factor enable miniaturized and scalable systems. The complete solution-on-a-chip reduces learning curve to a minimum while giving best user-experience and a high cost-efficiency.

BLOCK DIAGRAM

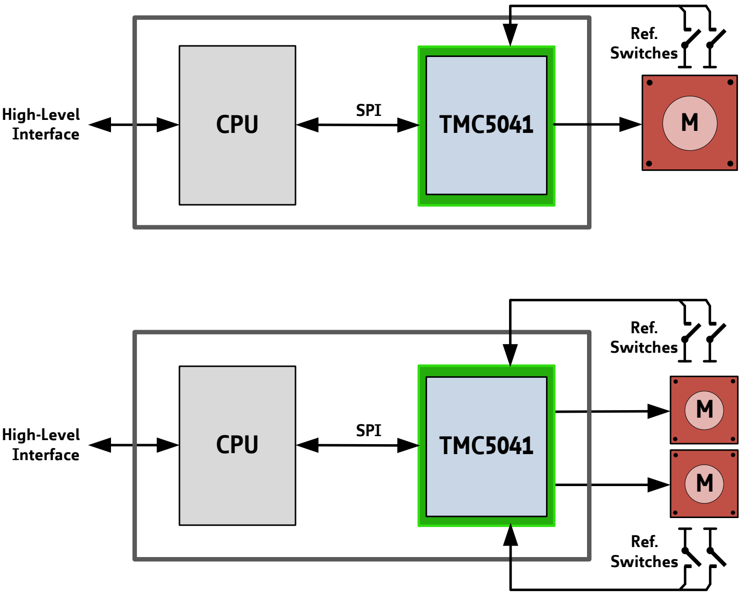


TRINAMIC Motion Control GmbH & Co. KG
Hamburg, Germany

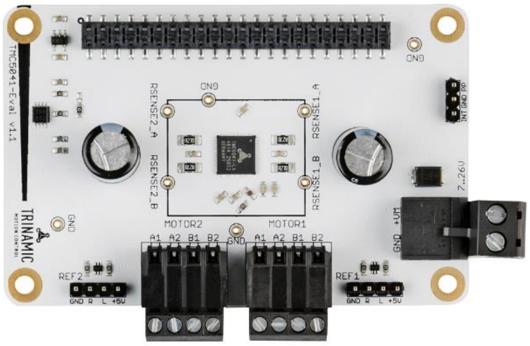
APPLICATION EXAMPLES: HIGH FLEXIBILITY – MULTIPURPOSE USE

The TMC5041 scores with power density, complete motion controlling features and integrated power stages. It offers a versatility that covers a wide spectrum of applications from battery systems up to embedded applications with 1.5A motor current per coil. The small form factor keeps costs down and allows for miniaturized layouts. Extensive support at the chip, board, and software levels enables rapid design cycles and fast time-to-market with competitive products. High energy efficiency and reliability deliver cost savings in related systems such as power supplies and cooling.

MINIATURIZED DESIGN FOR UP TO TWO STEPPER MOTORS



Two reference switch inputs can be used for each motor. A single CPU controls the whole system, which is highly economical and space saving, because the TMC5041 covers all functionality required to drive the motor.



Layout for Evaluation

TMC5041-EVAL EVALUATION BOARD

EVALUATION & DEVELOPMENT PLATFORM

The TMC5041-EVAL is part of TRINAMICs universal evaluation board system which provides a convenient handling of the hardware as well as a user-friendly software tool for evaluation. The TMC5041 evaluation board system consists of three parts: STARTRAMPE (base board), ESELSBRÜCKE (connector board including several test points), and TMC5041-EVAL.

ORDER CODES

Order code	Description	Size [mm^2]
TMC5041-LA	2-Axis Stepper Motor Controller/Driver IC, SPI, 5-26V Supply, 1.1A, QFN48, Tray	7 x 7
TMC5041-LA-T	2-Axis Stepper Motor Controller/Driver IC, SPI, 5-26V Supply, 1.1A, QFN48, Tape & Reel	7 x 7
TMC5041-EVAL-KIT	Full Evaluation Kit for TMC5041	126 x 85
TMC5041-EVAL	Evaluation Board for TMC5041 (excl. Landungsbrücke and Eselsbrücke)	85 x 55
TMC5041-BOB	Breakout board for simple prototyping	25 x 36

TABLE OF CONTENTS

1 Principles of Operation

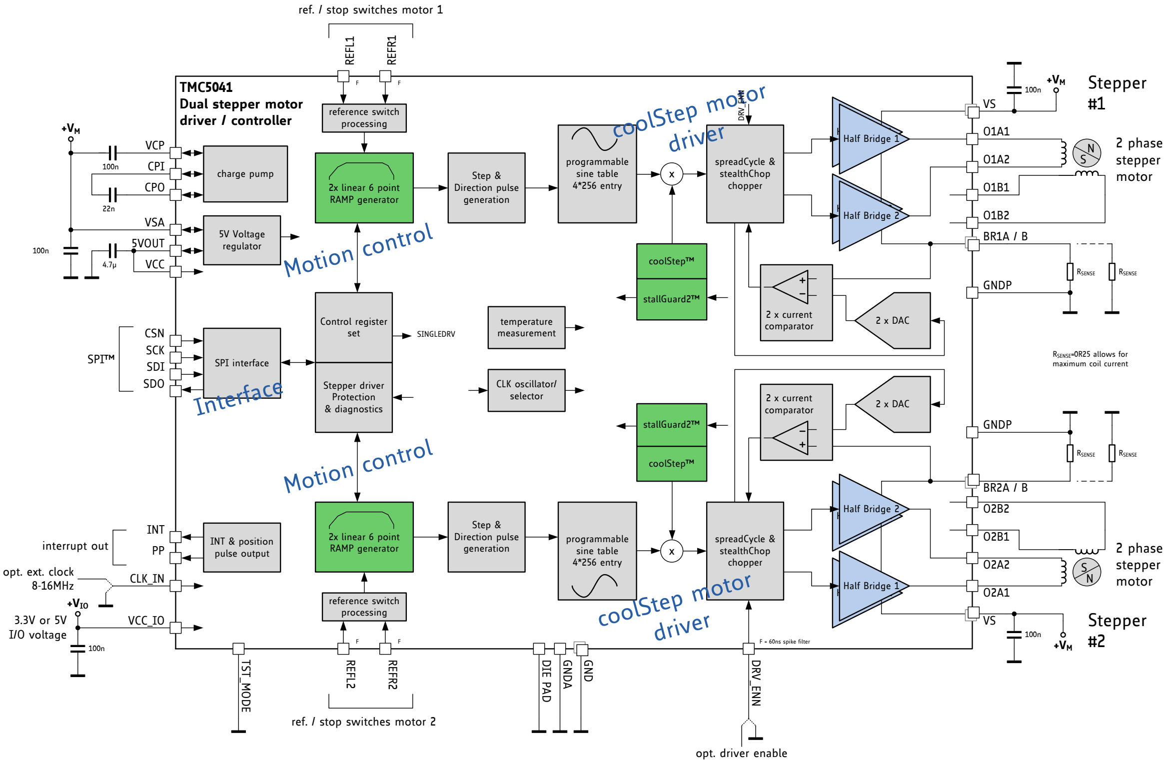


Figure 1.1 Basic application and block diagram

The TMC5041 motion controller and driver chip is an intelligent power component interfacing between the CPU and one or two stepper motors. All stepper motor logic is completely within the TMC5041. No software is required to control the motor – just provide target positions. The TMC5041 offers several unique enhancements which are enabled by the system-on-chip integration of driver and controller. The SixPoint ramp generator of the TMC5041 uses StealthChop, CoolStep, and StallGuard2 automatically to optimize every motor movement. The clear concept and the comprehensive solution save design time.

1.1 Key Concepts

The TMC5041 implements several advanced features which are exclusive to TRINAMIC products. These features contribute toward greater precision, greater energy efficiency, higher reliability, smoother motion, and cooler operation in many stepper motor applications.

StealthChop™	No-noise, high-precision chopper algorithm for inaudible motion and inaudible standstill of the motor.
StallGuard2™	High-precision load measurement using the back EMF on the motor coils.
CoolStep™	Load-adaptive current control which reduces energy consumption by as much as 75%.
SpreadCycle™	High-precision chopper algorithm available as an alternative to the traditional constant off-time algorithm.
SixPoint™	Fast and precise positioning using a hardware ramp generator with a set of four acceleration / deceleration settings. Quickest response due to dedicated hardware.

In addition to these performance enhancements, TRINAMIC 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.

1.2 SPI Control Interface

The SPI interface is a bit-serial interface synchronous to a bus clock. For every bit sent from the bus master to the bus slave another bit is sent simultaneously from the slave to the master. Communication between an SPI master and the TMC5041 slave always consists of sending one 40-bit command word and receiving one 40-bit status word.

The SPI command rate typically is a few commands per complete motor motion.

1.3 Software

From a software point of view the TMC5041 is a peripheral with a number of control and status registers. Most of them can either be written only or read only. Some of the registers allow both read and write access. In case read-modify-write access is desired for a write only register, a shadow register can be realized in master software.

1.4 Moving and Controlling the Motor

1.4.1 Integrated 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 stop velocity are supported as well as a second acceleration and deceleration setting. The integrated motion controller supports immediate reaction to mechanical reference switches and to the sensorless stall detection StallGuard2.

Benefits are:

- Flexible ramp programming
- Efficient use of motor torque for acceleration and deceleration allows higher machine throughput
- Immediate reaction to stop and stall conditions

1.5 StealthChop Driver with Programmable Microstepping Wave

Current into the motor coils is controlled using a cycle-by-cycle chopper mode. Up to three chopper modes are available: a traditional constant off-time mode and the SpreadCycle mode as well as the unique StealthChop. The constant off-time mode provides higher torque at highest velocity, while SpreadCycle mode offers smoother operation and greater power efficiency over a wide range of speed and load. The SpreadCycle chopper scheme automatically integrates a fast decay cycle and guarantees smooth zero crossing performance. In contrast to the other chopper modes, StealthChop is a voltage chopper-based principle. It guarantees that the motor is quiet in standstill and in slow motion, except for noise generated by ball bearings. The extremely smooth motion is beneficial for many applications.

Programmable microstep shapes allow optimizing the motor performance.

Benefits of using StealthChop:

- Significantly improved microstepping with low-cost motors
- Motor runs smooth and quiet
- Absolutely no standby noise
- Reduced mechanical resonances yields improved torque

1.6 StallGuard2 – Mechanical Load Sensing

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. This gives more information on the drive allowing functions like sensorless homing and diagnostics of the drive mechanics.

1.7 CoolStep – Load Adaptive Current Control

CoolStep drives the motor at the optimum current. It uses the StallGuard2 load measurement information to adjust the motor current to the minimum amount required in the actual load situation. This saves energy and keeps the components cool.

Benefits are:

-	*Energy efficiency*	power consumption decreased up to 75%
-	*Motor generates less heat*	improved mechanical precision
-	*Less or no cooling*	improved reliability
-	*Use of smaller motor*	less torque reserve required → cheaper motor does the job

Figure 1.2 shows the efficiency gain of a 42mm stepper motor when using CoolStep compared to standard operation with 50% of torque reserve. CoolStep is enabled above 60RPM in the example.

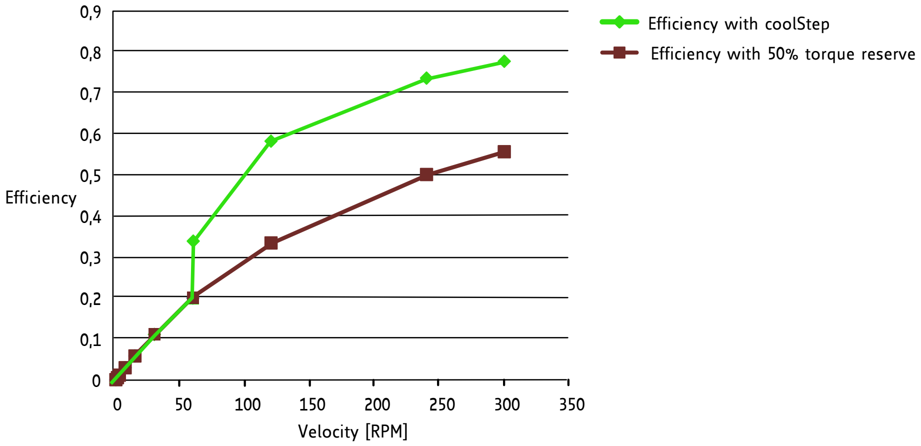


Figure 1.2 Energy efficiency with CoolStep (example)

2 Pin Assignments

2.1 Package Outline

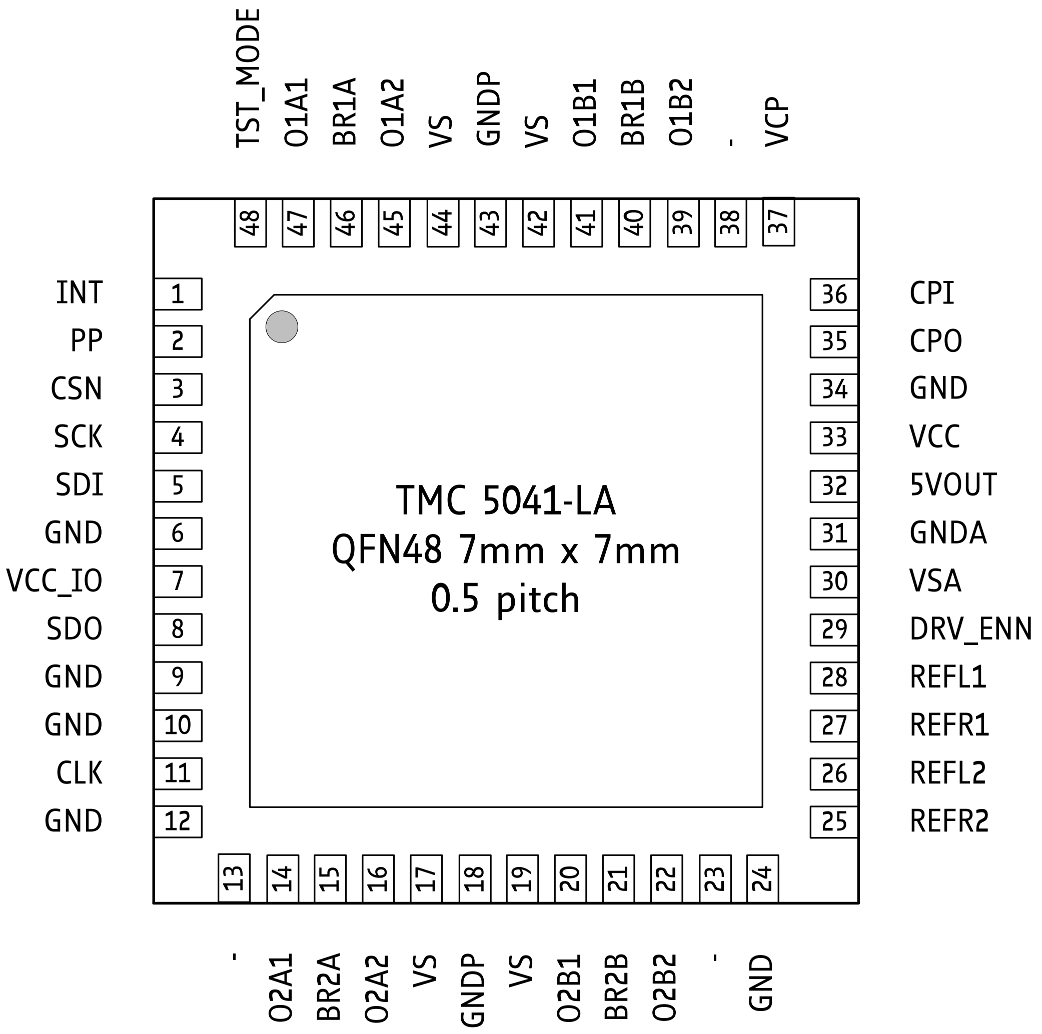


Figure 2.1 TMC5041 pin assignments.

2.2 Signal Descriptions

Pin	Number	Type	Function
GND	6, 9, 10, 12, 24, 34	GND	Digital ground pin for IO pins and digital circuitry.
VCC_IO	7		3.3V or 5V I/O supply voltage pin for all digital pins. Does not supply digital circuitry!
VSA	30		Analog supply voltage for 5V regulator – typically supplied with driver supply voltage. An additional 100nF capacitor to GND (GND plane) is recommended for best performance.
GNDA	31	GND	Analog GND. Tie to GND plane.
5VOUT	32		Output of internal 5V regulator. Attach 2.2µF or larger ceramic capacitor to GNDA near to pin for best performance. May be used to supply VCC of chip.

Pin	Number	Type	Function
VCC	33		5V supply input for digital circuitry within chip and charge pump. Attach 470nF capacitor to GND (GND plane). May be supplied by 5VOUT. A 2.2Ω resistor is recommended for decoupling noise from 5VOUT. When using an external supply, make sure, that VCC comes up before or in parallel to 5VOUT or VCC_IO, whichever comes up later!
DIE_PAD	-	GND	Connect the exposed die pad to a GND plane. Provide as many as possible vias for heat transfer to GND plane.

Table 2.1 Low voltage digital and analog power supply pins

Pin	Number	Type	Function
CPO	35	O(VCC)	Charge pump driver output. Outputs 5V (GND to VCC) square wave with 1/16 of internal oscillator frequency.
CPI	36	I(VCP)	Charge pump capacitor input: Provide external 22nF to 33nF / 50V capacitor to CPO.
VCP	37		Output of charge pump. Provide external 100nF capacitor to VS.

Table 2.2 Charge pump pins

Pin	Number	Type	Function
INT	1	O (Z)	Tristate interrupt output based on ramp generator flags *RAMP_STAT* bits 4, 5, 6 & 7. Outputs positive active interrupt signal (enable by *poscmp_enable*=1). Set *poscmp_enable*=1 to avoid floating pin, also in case the pin is not connected.
PP	2	O (Z)	Tristate position compare output for motor 1 (*poscmp_enable*=1). Set *poscmp_enable*=1 to avoid floating pin, also in case the pin is not connected.
CSN	3	I	Chip select input of SPI interface
SCK	4	I	Serial clock input of SPI interface
SDI	5	I	Data input of SPI interface
SDO	8	O (Z)	Data output of SPI interface (Tristate, enabled with CSN=0)
CLK	11	I	Clock input. Tie to GND using short wire for internal clock or supply external clock. The first high signal disables the internal oscillator until power down.
REFR2	25	I	Right reference switch input for motor 2
REFL2	26	I	Left reference switch input for motor 2
REFR1	27	I	Right reference switch input for motor 1
REFL1	28	I	Left reference switch input for motor 1
DRV_ENN	29	I	Enable input for motor drivers. The power stage becomes switched off (all motor outputs floating) when this pin becomes driven to a high level. Tie to GND for normal operation.
TST_MODE	48	I	Test mode input. Tie to GND using short wire.
-	13, 23, 38	N.C.	Unused pins – no internal electrical connection. Leave open or tie to GND for compatibility with future devices.

Table 2.3 Digital I/O pins (all related to VCC_IO supply)

Pin	Number	Type	Function
O2A1	14	O (VS)	Motor 2 coil A output 1
BR2A	15		Sense resistor connection for motor 2 coil A. Place sense resistor to GND near pin.
O2A2	16	O (VS)	Motor 2 coil A output 2
VS	17, 19		Motor supply voltage. Provide filtering capacity near pin with shortest loop to nearest GNDP pin (respectively via GND plane).
GNDP	18	GND	Power GND. Connect to GND plane near pin.
O2B1	20	O (VS)	Motor 2 coil B output 1
BR2B	21		Sense resistor connection for motor 2 coil B. Place sense resistor to GND near pin.
O2B2	22	O (VS)	Motor 2 coil B output 2
O1B2	39	O (VS)	Motor 1 coil B output 2
BR1B	40		Sense resistor connection for motor 1 coil B. Place sense resistor to GND near pin.
O1B1	41	O (VS)	Motor 1 coil B output 1
VS	42, 44		Motor supply voltage. Provide filtering capacity near pin with shortest loop to nearest GNDP pin (respectively via GND plane).
GNDP	43	GND	Power GND. Connect to GND plane near pin.
O1A2	45	O (VS)	Motor 1 coil A output 2
BR1A	46		Sense resistor connection for motor 1 coil A. Place sense resistor to GND near pin.
O1A1	47	O (VS)	Motor 1 coil A output 1

Table 2.4 Power driver pins

3 Sample Circuits

The sample circuits show the connection of the external components in different operation and supply modes. The connection of the bus interface and further digital signals is left out for clarity.

3.1 Standard Application Circuit

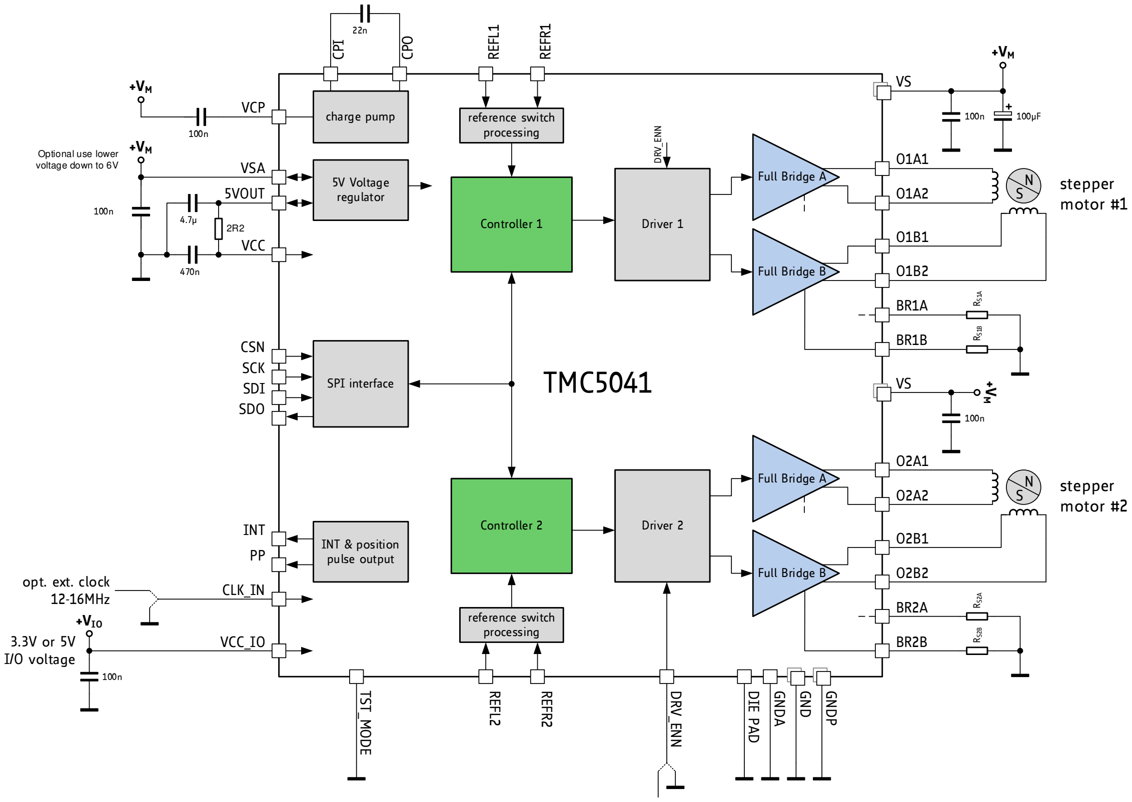


Figure 3.1 Standard application circuit

The standard application circuit uses two sense resistors to set the motor coil current. Use low ESR capacitors for filtering the power supply. The capacitors need to cope with the current ripple cause by chopper operation. A minimum capacity of 100µF near the driver is recommended for best performance. Current ripple in the supply capacitors also depends on the power supply internal resistance and cable length. VCC_IO can be supplied from 5VOUT, or from an external source, e.g., a low drop 3.3V regulator. To minimize linear voltage regulator power dissipation of the internal 5V voltage regulator in applications where VM is high, a different (lower) supply voltage can be used for VSA, if available. For example, many applications provide a 12V supply in addition to a higher supply voltage like 24V. Using the 12V supply for VSA will reduce the power dissipation of the internal 5V regulator to about 37% of the dissipation caused by supply with the full motor voltage. For best motor chopper performance, an optional R/C-filter de-couples 5VOUT from digital noise cause by power drawn from VCC.

Basic layout hints

Place sense resistors and all filter capacitors as close as possible to the related IC pins. Use a solid common GND for all GND connections, also for sense resistor GND. Connect 5VOUT filtering capacitor directly to 5VOUT and GNDA pin. See layout hints for more details. Low ESR electrolytic capacitors are recommended for VS filtering.

Attention

In case VSA is supplied by a different voltage source, make sure that VSA does not exceed VS by more than one diode drop, especially also upon power up or power down.

Attention
Ensure sufficient capacity on VS to limit supply ripple, and to keep power slopes below 1V/µs. Failure to do so could result in destructive currents via the charge pump capacitor. Provide overvoltage protection in case the motor could be manually turned at a high velocity, or in case the driver could become cut off from the main supply capacitors. Significant energy can be fed back from motor coils to the power supply in the event of quick deceleration, or when the driver becomes disabled.

3.2 5 V Only Supply

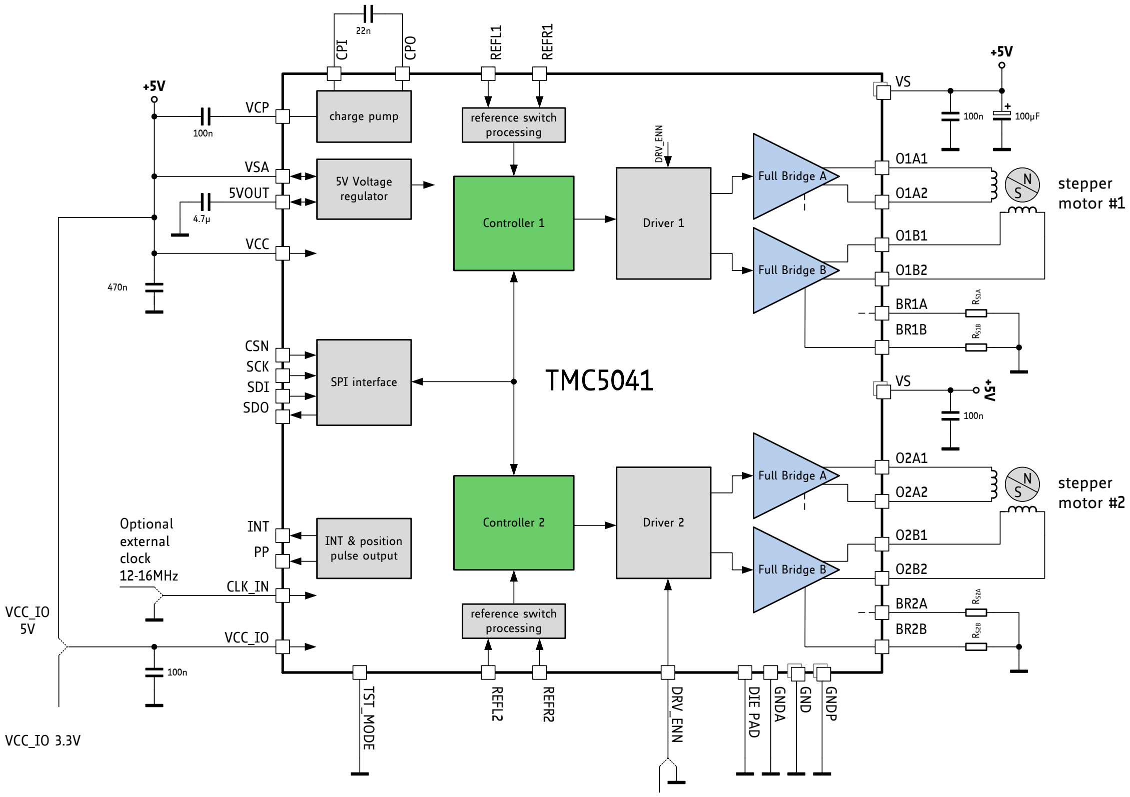


Figure 3.2 5V only operation

While the standard application circuit is limited to roughly 5.5 V lower supply voltage, a 5 V only application lets the IC run from a normal 5 V +/-5% supply. In this application, linear regulator drop must be minimized. Therefore, the major 5 V load is removed by supplying VCC directly from the external supply. In order to keep supply ripple away from the analog voltage reference, 5VOUT should have an own filtering capacity and the 5VOUT pin does not become bridged to the 5V supply.

3.3 External 5V Power Supply

When a clean external 5V power supply is available, the power dissipation caused by the internal linear regulator can be eliminated by directly supplying the analog and digital part (Figure 3.3) of the driver. This especially is beneficial in high voltage applications, and when thermal conditions are critical. Make sure, that the 5V supply does not exceed VS level by more than a diode drop.

The circuit will benefit from a well-regulated supply, e.g., when using a +/-1% regulator. A precise supply guarantees increased motor current precision because the voltage at 5VOUT directly is the reference voltage for all internal units of the driver, especially for motor current control. For best performance, the power supply should have low ripple to give a precise and stable supply at 5VOUT pin with remaining ripple well below 5mV. Some switching regulators have a higher remaining ripple, or different loads on the supply may cause lower frequency ripple. In this case, increase capacity attached to 5VOUT. In case the external supply voltage has poor stability or low frequency ripple, this would affect the precision of the motor current regulation as well as add chopper noise.

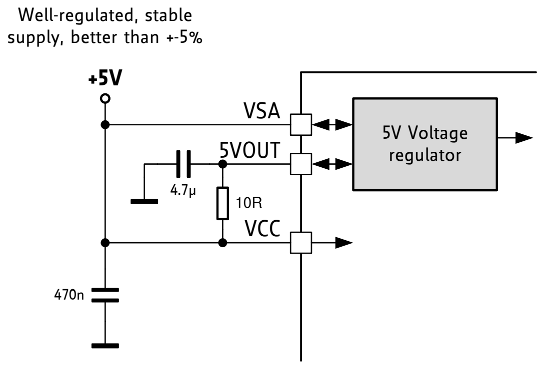


Figure 3.3 Using an external 5V supply to bypass internal regulator

3.4 Optimizing Analog Precision

The 5VOUT pin is used as an analog reference for operation of the TMC5041. Performance will degrade when there is voltage ripple on this pin. Most of the high frequency ripple in a TMC5041 design results from the operation of the internal digital logic. The digital logic switches with each edge of the clock signal. Further, ripple results from operation of the charge pump, which operates with roughly 1 MHz and draws current from the VCC pin. To keep this ripple as low as possible, an additional filtering capacitor can be put directly next to the VCC pin with vias to the GND plane giving a short connection to the digital GND pins (pin 6 and pin 34). Analog performance is best, when this ripple is kept away from the analog supply pin 5VOUT, using an additional series resistor of 2.2 Ω. The voltage drop on this resistor will be roughly 100 mV (I_{VCC} * R).

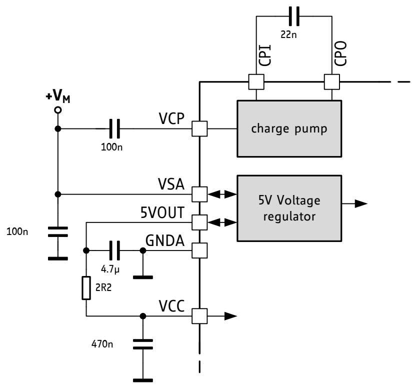
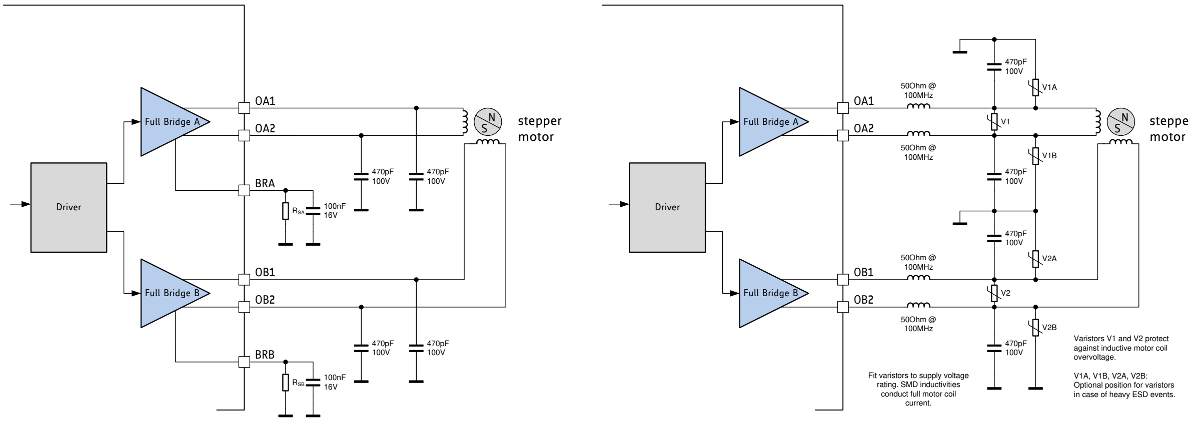


Figure 3.4 RC-Filter on VCC for reduced ripple

3.5 Driver Protection and EME Circuitry

Some applications have to cope with ESD events caused by motor operation or external influence. Despite ESD 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. Especially 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 up to a certain degree against ESD events or live plugging / pulling the motor, which also causes high voltages and high currents into the motor connector terminals. A simple scheme uses capacitors at the driver outputs to reduce the dV/dt caused by ESD events. Larger capacitors will bring more benefit concerning ESD suppression, but cause additional current flow in each chopper cycle, and thus increase driver power dissipation, especially at high supply voltages. The values shown are example values – they might 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. A more elaborate scheme uses LC filters to de-couple the driver outputs from the motor connector. Varistors in between of the coil terminals eliminate coil overvoltage caused by live plugging. Optionally protect all outputs by a varistor against ESD voltage.



Fit varistors to supply voltage rating. SMD inductivities conduct full motor coil current.

Varistors V1 and V2 protect against inductive motor coil overvoltage.

V1A, V1B, V2A, V2B: Optional position for varistors in case of heavy ESD events.

Figure 3.5 Simple ESD enhancement and more elaborate motor output protection

4 SPI Interface

4.1 SPI Datagram Structure

The TMC5041 uses 40-bit SPI™ (Serial Peripheral Interface, SPI is Trademark of Motorola) datagrams for communication with a microcontroller. Microcontrollers which are equipped with hardware SPI are typically able to communicate using integer multiples of 8 bit. The NCS line of the TMC5072 must be handled in a way, that it stays 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 via 32 data bits even if it uses less than 32 data bits.

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.

Most registers are write-only registers, some can be read additionally, and there are also some read only registers.

<table>
<tr><th colspan="40">SPI DATAGRAM STRUCTURE</th></tr>
<tr><td colspan="13">MSB (transmitted first)</td><td colspan="14">40 bit</td><td colspan="13">LSB (transmitted last)</td></tr>
<tr><td colspan="20">39 …</td><td colspan="20">… 0</td></tr>
<tr><td colspan="8">**→ 8 bit address**
← 8 bit SPI status</td><td colspan="32">**← → 32 bit data**</td></tr>
<tr><td colspan="8">39 … 32</td><td colspan="32">31 … 0</td></tr>
<tr><td colspan="8">→ to TMC5041:
RW + 7 bit address
← from TMC5041:
8 bit SPI status</td><td colspan="8">8 bit data</td><td colspan="8">8 bit data</td><td colspan="8">8 bit data</td><td colspan="8">8 bit data</td></tr>
<tr><td colspan="8">39 / 38 … 32</td><td colspan="8">31 … 24</td><td colspan="8">23 … 16</td><td colspan="8">15 … 8</td><td colspan="8">7 … 0</td></tr>
<tr><td>W</td><td colspan="7">38…32</td><td colspan="4">31…28</td><td colspan="4">27…24</td><td colspan="4">23…20</td><td colspan="4">19…16</td><td colspan="4">15…12</td><td colspan="4">11…8</td><td colspan="4">7…4</td><td colspan="4">3…0</td></tr>
<tr><td>39</td><td>38</td><td>37</td><td>36</td><td>35</td><td>34</td><td>33</td><td>32</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
</table>

4.1.1 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. So, the bit named W is a WRITE_notREAD control bit. The active high write bit is the MSB of the address byte. So, 0x80 has to be added to the address for a write access. The SPI interface always delivers data back to the master, 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. So, the difference between a read and a write access is that the read access does not transfer data to the addressed register, but it transfers the address only and its 32 data bits are dummies, and, further 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 master 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 TMC5041, the MSBs delivered back contain the SPI status, *SPI_STATUS*, a number of eight selected status bits.

Example:

For a read access to the register (*XACTUAL*) with the address 0x21, the address byte has to be set to 0x21 in the access preceding the read access. For a write access to the register (*VACTUAL*), the address byte has to be set to 0x80 + 0x22 = 0xA2. For read access, the data bit might have any value (-). So, one can set them to 0.

action	data sent to TMC5041	data received from TMC5041
read *XACTUAL*	→ `0x2100000000`	← `0xSS & unused data`
read *XACTUAL*	→ `0x2100000000`	← `0xSS & XACTUAL`
write *VMAX*:= `0x00ABCDEF`	→ `0xA700ABCDEF`	← `0xSS & XACTUAL`
write *VMAX*:= `0x00123456`	→ `0xA700123456`	← `0xSS00ABCDEF`

`*)S:` is a placeholder for the status bits *SPI_STATUS*

4.1.2 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

Bit	Name	Comment
7	-	reserved (0)
6	*status_stop_l(2)*	*RAMP_STAT2*[0] – 1: Signals motor 2 stop left switch status
5	*status_stop_l(1)*	*RAMP_STAT1*[0] – 1: Signals motor 1 stop left switch status
4	*velocity_reached(2)*	*RAMP_STAT2*[8] – 1: Signals motor 2 has reached its target velocity
3	*velocity_reached(1)*	*RAMP_STAT1*[8] – 1: Signals motor 1 has reached its target velocity
2	*driver_error(2)*	*GSTAT*[2] – 1: Signals driver 2 driver error (clear by reading *GSTAT*)
1	*driver_error(1)*	*GSTAT*[1] – 1: Signals driver 1 driver error (clear by reading *GSTAT*)
0	*reset_flag*	*GSTAT*[0] – 1: Signals, that a reset has occurred (clear by reading *GSTAT*)

4.1.3 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.

4.2 SPI Signals

The SPI bus on the TMC5041 has four signals:

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

The slave 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 slave 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 TMC5041.

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.

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 master to the slave. If more than 40 bits are sent, only the last 40 bits received before the rising edge of CSN are recognized as the command.

4.3 Timing

The SPI interface is synchronized to the internal system clock, which limits the SPI bus clock SCK to half of the system clock frequency. If the system clock is based on the on-chip oscillator, an additional 10% safety margin must be used to ensure reliable data transmission. All SPI inputs as well as the ENN input are internally filtered to avoid triggering on pulses shorter than 20ns. Figure 4.1 shows the timing parameters of an SPI bus transaction, and the table below specifies their values.

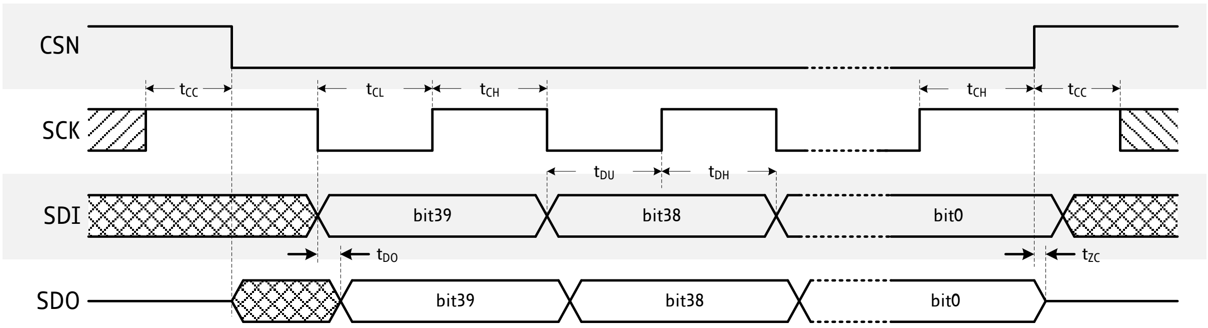


Figure 4.1 SPI timing

Hint
Usually, this SPI timing is referred to as SPI MODE 3

SPI interface timing	**AC-Characteristics** clock period: t_{CLK}					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
SCK valid before or after change of CSN	t_{CC}		10			ns
CSN high time	t_{CSH}	*) Min time is for synchronous CLK with SCK high one t_{CH} before CSN high only	t_{CLK} *)	>2t_{CLK}+10		ns
SCK low time	t_{CL}	*) Min time is for synchronous CLK only	t_{CLK} *)	>t_{CLK}+10		ns
SCK high time	t_{CH}	*) Min time is for synchronous CLK only	t_{CLK} *)	>t_{CLK}+10		ns
SCK frequency using internal clock	f_{SCK}	assumes minimum OSC frequency			4	MHz
SCK frequency using external 16MHz clock	f_{SCK}	assumes synchronous CLK			8	MHz
SDI setup time before rising edge of SCK	t_{DU}		10			ns
SDI hold time after rising edge of SCK	t_{DH}		10			ns
Data out valid time after falling SCK clock edge	t_{DO}	no capacitive load on SDO			t_{FILT}+5	ns
SDI, SCK and CSN filter delay time	t_{FILT}	rising and falling edge	12	20	30	ns

5 Register Mapping

This chapter gives an overview of the complete register set. Some of the registers bundling a number of single bits are detailed in extra tables. The functional practical application of the settings is detailed in dedicated chapters.

> *Note*
> - All registers become reset to 0 upon power up, unless otherwise noted.
> - Add 0x80 to the address **Addr** for write accesses!

NOTATION OF HEXADECIMAL AND BINARY NUMBERS	
0x	precedes a hexadecimal number, e.g. 0x04
%	precedes a multi-bit binary number, e.g. %100

NOTATION OF R/W FIELD	
R	Read only
W	Write only
R/W	Read- and writable register
R+C	Clear upon read

OVERVIEW REGISTER MAPPING

REGISTER	DESCRIPTION
General Configuration Registers	These registers contain - global configuration - global status flags
Ramp Generator Motion Control Register Set	This register set offers registers for - choosing a ramp mode - choosing velocities - homing - acceleration and deceleration - target positioning
Ramp Generator Driver Feature Control Register Set	This register set offers registers for - driver current control - setting thresholds for CoolStep operation - setting thresholds for different chopper modes - reference switch and StallGuard2 event configuration - a ramp and reference switch status register
Motor Driver Register Set	This register set offers registers for - setting / reading out microstep table and counter - chopper and driver configuration - CoolStep and StallGuard2 configuration - reading out StallGuard2 values and driver error flags

5.1 General Configuration Registers

GENERAL CONFIGURATION REGISTERS (0X00…0X0F)

R/W	**Addr**	**n**	***Register***	**Description / bit names**	
RW	0x00	11	*GCONF*	**Bit**	***GCONF* – Global configuration flags**
				0..2	Reserved, set to 0
				3	*poscmp_enable* 0: Outputs INT and PP are tristated. 1: Position compare pulse (PP) and interrupt output (INT) are available *Attention – do not leave the outputs floating in tristate condition, provide an external pull-up or set poscmp_enable=1*
				4..6	Reserved, set to 0
				7	*test_mode* 0: Normal operation 1: Enable analog test output on pin REFR2 *TEST_SEL* selects the function of REFR2: 0…4: T120, DAC1, VDDH1, DAC2, VDDH2 *Attention: Not for user, set to 0 for normal operation!*
				8	*shaft1* 1: Inverse motor 1 direction
				9	*shaft2* 1: Inverse motor 2 direction
				10	*lock_gconf* 1: GCONF is locked against further write access.
				11	Reserved, set to 0
R+C	0x01	4	*GSTAT*	**Bit**	***GSTAT* – Global status flags**
				0	*reset* 1: Indicates that the IC has been reset since the last read access to *GSTAT.* All registers have been cleared to reset values.
				1	*drv_err1* 1: Indicates, that driver 1 has been shut down due to overtemperature or short circuit detection since the last read access. Read *DRV_STATUS1* for details. The flag can only be reset when all error conditions are cleared.
				2	*drv_err2* *1:* Indicates, that driver 2 has been shut down due to overtemperature or short circuit detection since the last read access. Read *DRV_STATUS2* for details. The flag can only be reset when all error conditions are cleared.
				3	*uv_cp* 1: Indicates an undervoltage on the charge pump. The driver is disabled in this case.
W	0x03	4	*TEST_SEL*	Select test mode output *Attention: Not for user, set to 0 for normal operation!*	
R	0x04	9 + 8	*INPUT*	**Bit**	***INPUT***
				0..6	Unused, ignore these bits
				7	*drv_enn_in*: DRV_ENN pin polarity
				8	Unused, ignore this bit
				31.. 24	*VERSION*: 0x10=version of the IC Identical numbers mean full digital compatibility.

GENERAL CONFIGURATION REGISTERS (0X00…0X0F)				
R/W	**Addr**	**n**	***Register***	**Description / *bit names***
W	0x05	32	*X_COMPARE*	Position comparison register for motor 1 position strobe. Activate *poscmp_enable* to get position pulse on output PP. *XACTUAL* = *X_COMPARE*: - Output PP becomes high. It returns to a low state if the positions mismatch.

5.2 Ramp Generator Registers

*Addresses **Addr** are specified for motor 1 (upper value) and motor 2 (second address).*

5.2.1 Ramp Generator Motion Control Register Set

RAMP GENERATOR MOTION CONTROL REGISTER SET (MOTOR 1: 0X20…0X2D, MOTOR 2: 0X40…0X4D)

R/W	Addr	n	*Register*	Description / *bit names*	Range [Unit]
RW	0x20 0x40	2	*RAMPMODE*	*RAMPMODE:* 0: Positioning mode (using all A, D and V parameters) 1: Velocity mode to positive *VMAX* (using *AMAX* acceleration) 2: Velocity mode to negative *VMAX* (using *AMAX* acceleration) 3: Hold mode (velocity remains unchanged, unless stop event occurs)	0…3
RW	0x21 0x41	32	*XACTUAL*	Actual motor position (signed) *Hint:* This value normally should only be modified, when homing the drive. In positioning mode, modifying the register content will start a motion.	-2^31… +(2^31)-1
R	0x22 0x42	24	*VACTUAL*	Actual motor velocity from ramp generator (signed) The sign matches the motion direction. A negative sign means motion to lower XACTUAL.	+-(2^23)-1 [µsteps / t]
W	0x23 0x43	18	*VSTART*	Motor start velocity (unsigned) Normally, set *VSTOP* ≥ *VSTART!* VSTART may be set to a higher value, when motion distance is sufficient to allow deceleration to *VSTOP.*	0…(2^18)-1 [µsteps / t]
W	0x24 0x44	16	*A1*	First acceleration between *VSTART* and *V1* (unsigned)	0…(2^16)-1 [µsteps / ta^2]
W	0x25 0x45	20	*V1*	First acceleration / deceleration phase threshold velocity (unsigned) 0: Disables *A1* and *D1* phase, use *AMAX*, *DMAX* only	0…(2^20)-1 [µsteps / t]
W	0x26 0x46	16	*AMAX*	Second acceleration between *V1* and *VMAX* (unsigned) This is the acceleration and deceleration value for velocity mode.	0…(2^16)-1 [µsteps / ta^2]
W	0x27 0x47	23	*VMAX*	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…(2^23)-512 [µsteps / t]
W	0x28 0x48	16	*DMAX*	Deceleration between *VMAX* and *V1* (unsigned)	0…(2^16)-1 [µsteps / ta^2]

RAMP GENERATOR MOTION CONTROL REGISTER SET (MOTOR 1: 0X20…0X2D, MOTOR 2: 0X40…0X4D)					
R/W	**Addr**	**n**	***Register***	**Description / *bit names***	**Range [Unit]**
W	0x2A 0x4A	16	*D1*	Deceleration between *V1* and *VSTOP* (unsigned) *Attention: Do not set 0 in positioning mode, even if V1=0!*	1…(2^16)-1 [µsteps / ta²]
W	0x2B 0x4B	18	*VSTOP*	Motor stop velocity (unsigned) *Attention: Set VSTOP ≥ VSTART!* *Attention: Do not set 0 in positioning mode, minimum 10 recommended!*	1…(2^18)-1 [µsteps / t]
W	0x2C 0x4C	16	*TZEROWAIT*	Waiting time after ramping down to zero velocity before next movement or direction inversion can start and before motor power down starts. Time range is about 0 to 2 seconds. This setting avoids excess acceleration e.g. from *VSTOP* to -*VSTART*.	0…(2^16)-1 * 512 t_{CLK}
RW	0x2D 0x4D	32	*XTARGET*	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. *Hint:* The position is allowed to wrap around, thus, *XTARGET* value optionally can be treated as an unsigned number. *Hint:* The maximum possible displacement is +/-((2^31)-1). *Hint:* When increasing *V1, D1 or DMAX during* a motion, rewrite *XTARGET* afterwards to trigger a second acceleration phase, if desired.	-2^31… +(2^31)-1

5.2.2 Ramp Generator Driver Feature Control Register Set

RAMP GENERATOR DRIVER FEATURE CONTROL REGISTER SET (MOTOR 1: 0X30…0X36, MOTOR 2: 0X50…0X56)

R/W	Addr	n	*Register*	Description / *bit names*	
W	0x30 0x50	5 + 5 + 4	*IHOLD_IRUN*	**Bit**	***IHOLD_IRUN* – Driver current control**
				4..0	*IHOLD* Standstill current (0=1/32…31=32/32) In combination with StealthChop mode, setting *IHOLD*=0 allows to choose freewheeling or coil short circuit for motor stand still.
				12..8	*IRUN* Motor run current (0=1/32…31=32/32) *Hint:* Choose sense resistors in a way, that normal *IRUN* is 16 to 31 for best microstep performance.
				19..16	*IHOLDDELAY* Controls the number of clock cycles for motor power down after a motion as soon as *TZEROWAIT* has expired. The smooth transition avoids a motor jerk upon power down. 0: instant power down 1..15: Delay per current reduction step in multiple of 2^18 clocks
W	0x31 0x51	23	*VCOOLTHRS*		This is the lower threshold velocity for switching on smart energy CoolStep and StallGuard feature. Further it is the upper operation velocity for StealthChop. (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) becomes enabled when exceeding this velocity. It becomes disabled again once the velocity falls below this threshold. This allows for homing procedures with StallGuard by blanking out the StallGuard signal at low velocities (will not work in combination with StealthChop). *VHIGH* ≥ |*VACT*| ≥ VCOOLTHRS: - CoolStep and stop on stall are enabled, if configured - Voltage PWM mode StealthChop is switched off, if configured (Only bits 22..8 are used for value and for comparison)
W	0x32 0x52	23	*VHIGH*		This velocity setting allows velocity dependent switching into a different chopper mode and fullstepping to maximize torque. (unsigned) |*VACT*| ≥ *VHIGH*: - CoolStep is disabled (motor runs with normal current scale) - If *vhighchm* is set, the chopper switches to *chm*=1 with *TFD*=0 (constant off time with slow decay, only). - If *vhighfs* is set, the motor operates in fullstep mode. - Voltage PWM mode StealthChop is switched off, if configured (Only bits 22..8 are used for value and for comparison)

RAMP GENERATOR DRIVER FEATURE CONTROL REGISTER SET (MOTOR 1: 0X30…0X36, MOTOR 2: 0X50…0X56)				
R/W	**Addr**	**n**	***Register***	**Description / *bit names***
RW	0x34 0x54	12	***SW_MODE***	Switch mode configuration *See separate table!*
R+C	0x35 0x55	14	***RAMP_STAT***	Ramp status and switch event status *See separate table!*
R	0x36 0x56	32	*XLATCH*	Ramp generator latch position, latches *XACTUAL* upon a programmable switch event (see *SW_MODE*).

Time reference t for velocities: $t = 2^{24} / f_{CLK}$

Time reference ta² for accelerations: $ta^2 = 2^{41} / (f_{CLK})^2$

6.2.2.1 *SW_MODE* – Reference Switch & StallGuard2 Event Configuration Register

0X34, 0X54: *SW_MODE* – REFERENCE SWITCH AND STALLGUARD2 EVENT CONFIGURATION REGISTER		
Bit	**Name**	**Comment**
11	*en_softstop*	0: Hard stop 1: Soft stop The soft stop mode always uses the deceleration ramp settings *DMAX*, V1, *D1*, *VSTOP* and *TZEROWAIT* for stopping the motor. A stop occurs when the velocity sign matches the reference switch position (REFL for negative velocities, REFR for positive velocities) and the respective switch stop function is enabled. A hard stop also uses *TZEROWAIT* before the motor becomes released. *Attention: Do not use soft stop in combination with StallGuard2.*
10	*sg_stop*	1: Enable stop by StallGuard2. Disable to release motor after stop event. *Attention: Do not enable during motor spin-up, wait until the motor velocity exceeds a certain value, where StallGuard2 delivers a stable result, or set VCOOLTHRS to a suitable value.*
9	-	Unused, set to 0
8	*latch_r_inactive*	1: Activates latching of the position to *XLATCH* upon an inactive going edge on the right reference switch input REFR. The active level is defined by *pol_stop_r*.
7	*latch_r_active*	1: Activates latching of the position to *XLATCH* upon an active going edge on the right reference switch input REFR. *Hint:* Activate *latch_r_active* to detect any spurious stop event by reading *status_latch_r*.
6	*latch_l_inactive*	1: Activates latching of the position to *XLATCH* upon an inactive going edge on the left reference switch input REFL. The active level is defined by *pol_stop_l*.
5	*latch_l_active*	1: Activates latching of the position to *XLATCH* upon an active going edge on the left reference switch input REFL. *Hint:* Activate *latch_l_active* to detect any spurious stop event by reading *status_latch_l*.
4	*swap_lr*	1: Swap the left and the right reference switch input REFL and REFR
3	*pol_stop_r*	Sets the active polarity of the right reference switch input 0=non-inverted, high active: a high level on REFR stops the motor 1=inverted, low active: a low level on REFR stops the motor
2	*pol_stop_l*	Sets the active polarity of the left reference switch input 0=non-inverted, high active: a high level on REFL stops the motor 1=inverted, low active: a low level on REFL stops the motor
1	*stop_r_enable*	1: Enables automatic motor stop during active right reference switch input *Hint:* The motor restarts in case the stop switch becomes released.
0	*stop_l_enable*	1: Enables automatic motor stop during active left reference switch input *Hint:* The motor restarts in case the stop switch becomes released.

6.2.2.2 *RAMP_STAT* – Ramp and Reference Switch Status Register

0x35, 0x55: *RAMP_STAT* – RAMP AND REFERENCE SWITCH STATUS REGISTER

R/W	Bit	Name	Comment
R	13	*status_sg*	1: Signals an active StallGuard2 input from the CoolStep driver, if configured. *Hint:* When polling this flag, stall events may be missed – activate *sg_stop* to be sure not to miss the stall event.
R+C	12	*second_move*	1: Signals that the automatic ramp required moving back in the opposite direction, e.g. due to on-the-fly parameter change (Flag is cleared upon reading)
R	11	*t_zerowait_active*	1: Signals, that *TZEROWAIT* is active after a motor stop. During this time, the motor is in standstill.
R	10	*vzero*	1: Signals, that the actual velocity is 0.
R	9	*position_reached*	1: Signals, that the target position is reached. This flag becomes set while *XACTUAL* and *XTARGET* match.
R	8	*velocity_reached*	1: Signals, that the target velocity is reached. This flag becomes set while *VACTUAL* and *VMAX* match.
R+C	7	*event_pos_reached*	1: Signals, that the target position has been reached (*position_reached* becoming active). (Flag and interrupt condition are cleared upon reading) This bit is ORed to the *interrupt output* signal.
R+C	6	*event_stop_sg*	1: Signals an active StallGuard2 stop event. Reading the register will clear the stall condition and the motor may re-start motion, unless the motion controller has been stopped. (Flag and interrupt condition are cleared upon reading) This bit is ORed to the *interrupt output* signal.
R	5	*event_stop_r*	1: Signals an active stop right condition due to stop switch. 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 will remain 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 will continue motion. This bit is ORed to the *interrupt output* signal.
	4	*event_stop_l*	1: Signals an active stop left condition due to stop switch. 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 will remain 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 will continue motion. This bit is ORed to the *interrupt output* signal.
R+C	3	*status_latch_r*	1: Latch right ready (enable position latching using *SWITCH_MODE* settings *latch_r_active* or *latch_r_inactive*) (Flag is cleared upon reading)
	2	*status_latch_l*	1: Latch left ready (enable position latching using *SWITCH_MODE* settings *latch_l_active* or *latch_l_inactive*) (Flag is cleared upon reading)
R	1	*status_stop_r*	Reference switch right status (1=active)
	0	*status_stop_l*	Reference switch left status (1=active)

5.3 Microstep Table Registers

COMMON MICROSTEP TABLE REGISTERS (MOTOR 1/2: 0X60…0X69)

R/W	Addr	n	*Register*	Description / *bit names*	Range [Unit]
W	0x60	32	*MSLUT[0]* microstep table entries 0…31	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*	32x 0 or 1 *reset default= sine wave table*
W	0x61 … 0x67	7 x 32	*MSLUT[1...7]* microstep table entries 32…255		7x 32x 0 or 1 *reset default= sine wave table*
W	0x68	32	***MSLUTSEL***	This register defines four segments within each quarter *MSLUT* wave. Four 2-bit entries determine the meaning of a 0 and a 1 bit in the corresponding segment of *MSLUT*. *See separate table!*	*0<X1<X2<X3* *reset default= sine wave table*
W	0x69	8 + 8	*MSLUTSTART*	bit 7… 0: *START_SIN* bit 23… 16: *START_SIN90* *START_SIN* gives the absolute current at microstep table entry 0. *START_SIN90* gives the absolute current for microstep table entry at positions 256. Start values are transferred to the microstep registers *CUR_A* and *CUR_B*, whenever the reference position *MSCNT*=0 is passed.	*START_SIN* *reset default =0* *START_SIN90* *reset default =247*

MIRCOSTEP TABLE CALCULATION FOR A SINE WAVE EQUIVALENT TO THE POWER ON DEFAULT:

$$round\left(248 * sin\left(2 * PI * \frac{i}{1024} + \frac{PI}{1024}\right)\right) - 1$$

- *i*:[0… 255] is the table index
- The amplitude of the wave is 248. The resulting maximum positive value is 247 and the maximum negative value is -248.
- The round function rounds values from 0.5 to 1.4999 to 1

5.3.1 *MSLUTSEL* – Look up Table Segmentation Definition

0X68: *MSLUTSEL* – LOOK UP TABLE SEGMENTATION DEFINITION

Bit	Name	Function	Comment
31	*X3*	LUT segment 3 start	The sine wave look-up 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 shall satisfy: 0<*X1*<*X2*<*X3*
30			
29			
28			
27			
26			
25			
24			
23	*X2*	LUT segment 2 start	
22			
21			
20			
19			
18			
17			
16			
15	*X1*	LUT segment 1 start	
14			
13			
12			
11			
10			
9			
8			
7	*W3*	LUT width select from *ofs(X3)* to *ofs255*	Width control bit coding *W0…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
6			
5	*W2*	LUT width select from *ofs(X2)* to *ofs(X3-1)*	
4			
3	*W1*	LUT width select from *ofs(X1)* to *ofs(X2-1)*	
2			
1	*W0*	LUT width select from *ofs00* to *ofs(X1-1)*	
0			

5.4 Motor Driver Registers

MOTOR DRIVER REGISTER SET (MOTOR 1: 0X6A…0X6F, MOTOR 2: 0X7A…0X7F)					
R/W	**Addr**	**n**	***Register***	**Description / *bit names***	**Range [Unit]**
R	0x6A 0x7A	10	*MSCNT*	Microstep counter. Indicates actual position in the microstep table for *CUR_B*. *CUR_A* uses an offset of 256. *Hint:* Move to a position where *MSCNT* is zero before re-initializing *MSLUTSTART* or *MSLUT* and *MSLUTSEL*.	0…1023
R	0x6B 0x7B	9 + 9	*MSCURACT*	bit 8… 0: Sine *CUR_B* (signed): Actual microstep current for motor phase B as read from MSLUT (not scaled by current) bit 24… 16: Cosine *CUR_A* (signed): Actual microstep current for motor phase A as read from *MSLUT* (not scaled by current)	+/-0...255
RW	0x6C 0x7C	32	***CHOPCONF***	chopper and driver configuration *See separate table!*	
W	0x6D 0x7D	25	***COOLCONF***	CoolStep smart current control register and StallGuard2 configuration *See separate table!*	
R	0x6F 0x7F	32	***DRV_ STATUS***	StallGuard2 value and driver error flags *See separate table!*	

5.4.1 *CHOPCONF* – Chopper Configuration

0x6C, 0x7C: *CHOPCONF* – Chopper Configuration

Bit	Name	Function	Comment
31	-	reserved	set to 0
30	*diss2g*	short to GND protection disable	0: Short to GND protection is on 1: Short to GND protection is disabled
29	-	reserved	set to 0
28	-	reserved	set to 0
27	*mres3*	*MRES* micro step resolution	%0000: Native 256 microstep setting. Use this setting for normal operation.
26	*mres2*		
25	*mres1*		
24	*mres0*		%0001 … %1000: 128, 64, 32, 16, 8, 4, 2, FULLSTEP Reduced microstep resolution (only for special use!). The resolution gives the number of microstep entries per sine quarter wave. Especially when switching to a low resolution of 8 microsteps and below, take care to switch at certain microstep positions. The switching position determines the sequence of patterns. step width=2^*MRES* [microsteps] *Hint: Different microstep resolutions are only intended for special cases to extend the acceleration or position range*
23	-	reserved	set to 0
22	-		
21	-		
20	-		
19	*vhighchm*	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 in order to avoid doubling of the chopper frequency.
18	*vhighfs*	high velocity fullstep selection	This bit enables switching to fullstep, when *VHIGH* is exceeded. Switching takes place only at 45° position. The fullstep target current uses the current value from the microstep table at the 45° position.
17	*vsense*	sense resistor voltage based current scaling	0: Low sensitivity, high sense resistor voltage 1: High sensitivity, low sense resistor voltage
16	*tbl1*	*TBL* blank time select	%00 … %11: Set comparator blank time to 16, 24, 36 or 54 clocks *Hint*: %01 or %10 recommended for most applications
15	*tbl0*		
14	*chm*	chopper mode	0 Standard mode (SpreadCycle) 1 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.
13	*rndtf*	random *TOFF* time	0 Chopper off time is fixed as set by *TOFF* 1 Random mode, *TOFF* is random modulated by dN_{CLK}= -12 … +3 clocks.
12	*disfdcc*	fast decay mode	*chm*=1: *disfdcc*=1 disables current comparator usage for termination of the fast decay cycle

0x6C, 0x7C: *CHOPCONF* – Chopper Configuration

Bit	Name	Function	Comment	
11	*fd3*	*TFD [3]*	*chm*=1: MSB of fast decay time setting *TFD*	
10	*hend3*	*HEND* hysteresis low value *OFFSET* sine wave offset	*chm*=0	%0000 … %1111: Hysteresis is -3, -2, -1, 0, 1, …, 12 (1/512 of this setting adds to current setting) This is the hysteresis value which becomes used for the hysteresis chopper.
9	*hend2*			
8	*hend1*			
7	*hend0*		*chm*=1	%0000 … %1111: Offset is -3, -2, -1, 0, 1, …, 12 This is the sine wave offset and 1/512 of the value becomes added to the absolute value of each sine wave entry.
6	*hstrt2*	*HSTRT* hysteresis start value added to *HEND*	*chm*=0	%000 … %111: Add 1, 2, …, 8 to hysteresis low value *HEND* (1/512 of this setting adds to current setting) *Attention: Effective HEND+HSTRT ≤ 16.* *Hint:* Hysteresis decrement is done each 16 clocks
5	*hstrt1*			
4	*hstrt0*			
		TFD [2..0] fast decay time setting	*chm*=1	Fast decay time setting (MSB: *fd3*): %0000 … %1111: Fast decay time setting *TFD* with N_{CLK}= 32**HSTRT* (%0000: slow decay only)
3	*toff3*	*TOFF* off time and driver enable	Off time setting controls duration of slow decay phase N_{CLK}= 24 + 32**TOFF* %0000: Driver disable, all bridges off %0001: 1 – use only with *TBL* ≥ 36 clocks %0010 … %1111: 2 … 15	
2	*toff2*			
1	*toff1*			
0	*toff0*			

5.4.2 *COOLCONF* – Smart Energy Control CoolStep and StallGuard2

0x6D, 0x7D: *COOLCONF* – Smart Energy control CoolStep and StallGuard2

Bit	Name	Function	Comment
…	-	reserved	set to 0
24	*sfilt*	StallGuard2 filter enable	0 Standard mode, high time resolution for StallGuard2 1 Filtered mode, StallGuard2 signal updated for each four fullsteps only to compensate for motor pole tolerances
23	-	reserved	set to 0
22	*sgt6*	StallGuard2 threshold value	This signed value controls StallGuard2 level for stall output and sets the optimum measurement range for readout. A lower value gives a higher sensitivity. Zero is the starting value working with most motors. -64 to +63: A higher value makes StallGuard2 less sensitive and requires more torque to indicate a stall.
21	*sgt5*		
20	*sgt4*		
19	*sgt3*		
18	*sgt2*		
17	*sgt1*		
16	*sgt0*		
15	*seimin*	minimum current for smart current control	0: 1/2 of current setting (*IRUN*) 1: 1/4 of current setting (*IRUN*)
14	*sedn1*	current down step speed	%00: For each 32 StallGuard2 values decrease by one %01: For each 8 StallGuard2 values decrease by one %10: For each 2 StallGuard2 values decrease by one %11: For each StallGuard2 value decrease by one
13	*sedn0*		
12	-	reserved	set to 0
11	*semax3*	StallGuard2 hysteresis value for smart current control	If the StallGuard2 result is equal to or above (*SEMIN*+*SEMAX*+1)*32, the motor current becomes decreased to save energy. %0000 … %1111: 0 … 15
10	*semax2*		
9	*semax1*		
8	*semax0*		
7	-	reserved	set to 0
6	*seup1*	current up step width	Current increment steps per measured StallGuard2 value %00 … %11: 1, 2, 4, 8
5	*seup0*		
4	-	reserved	set to 0
3	*semin3*	minimum StallGuard2 value for smart current control and smart current enable	If the StallGuard2 result falls below *SEMIN**32, the motor current becomes increased to reduce motor load angle. %0000: smart current control CoolStep off %0001 … %1111: 1 … 15
2	*semin2*		
1	*semin1*		
0	*semin0*		

5.4.3 *DRV_STATUS* – StallGuard2 Value and Driver Error Flags

0x6F, 0x7F: DRV_STATUS – StallGuard2 value and driver error flags			
Bit	**Name**	**Function**	**Comment**
31	*stst*	standstill indicator	This flag indicates motor stand still in each operation mode. It is especially useful for step & dir mode.
30	*olb*	open load indicator phase B	1: Open load detected on phase A or B. *Hint:* This is just an informative flag. The driver takes no action upon it. False detection may occur in fast motion and standstill. Check during slow motion or after a motion, only.
29	*ola*	open load indicator phase A	
28	*s2gb*	short to ground indicator phase B	1: Short to GND detected on phase A or B. The driver becomes disabled. The flags stay active, until the driver is disabled by software (*TOFF*=0) or by the ENN input.
27	*s2ga*	short to ground indicator phase A	
26	*otpw*	overtemperature pre-warning flag	1: Overtemperature pre-warning threshold is exceeded. The overtemperature pre-warning flag is common for both drivers.
25	*ot*	overtemperature flag	1: 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 drivers.
24	*StallGuard*	StallGuard2 status	1: Motor stall detected (*SG_RESULT*=0)
23	-	reserved	Ignore these bits
22			
21			
20	*CS ACTUAL*	actual motor current / smart energy current	Actual current control scaling, for monitoring smart energy current scaling controlled via settings in register *COOLCONF*, or for monitoring the function of the automatic current scaling.
19			
18			
17			
16			
15	*fsactive*	full step active indicator	1: Indicates that the driver has switched to fullstep as defined by chopper mode settings and velocity thresholds.
14	-	reserved	Ignore these bits
13			
12			
11			
10			
9	*SG_ RESULT*	StallGuard2 result respectively PWM on time for coil A in standstill for motor temperature detection	Mechanical load measurement: The StallGuard2 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 StealthChop mode. StallGuard2 works best with microstep operation. Temperature measurement: In standstill, no StallGuard2 result can be obtained. *SG_RESULT* shows the chopper on-time for motor coil A instead. If the motor is moved to a determined microstep position at a certain current setting, a comparison of the chopper on-time can help to get a rough estimation of motor temperature. As the motor heats up, its coil resistance rises, and the chopper on-time increases.
8			
7			
6			
5			
4			
3			
2			
1			
0			

5.5 Voltage PWM mode StealthChop

Motor driver PWM register set (Motor 1: 0x10…0x17, Motor 2: 0x18…0x1F)

R/W	Addr	n	*Register*	Description / *bit names*	Range [Unit]
W	0x10 0x18	22	***PWMCONF***	Voltage PWM mode chopper configuration *See separate table!*	
R	0x11 0x19	8	*PWM_ STATUS*	Actual PWM scaler (255=max. Voltage)	0…255

5.5.1 *PWMCONF* – Voltage PWM mode StealthChop

0x10, 0x18: *PWMCONF* – Voltage mode PWM

Bit	Name	Function	Comment	
…	-	reserved	Set to 0	
21 20	*freewheel1* *freewheel0*	Allows different standstill modes	Stand still option when motor current setting is zero (*I_HOLD*=0). %00: Normal operation %01: Freewheeling %10: Coil shorted using LS drivers %11: Coil shorted using HS drivers	
19	-	reserved	Set to 0	
18	*pwm_ autoscale*	PWM automatic amplitude scaling	0	User defined PWM amplitude. The current settings have no influence.
			1	Enable automatic current control *Attention: When using a user defined sine wave table, the amplitude of this sine wave table should not be less than 244. Best results are obtained with 247 to 252 as peak values.*
17 16	*pwm_freq1* *pwm_freq0*	PWM frequency selection	%00: $f_{PWM}=2/1024\ f_{CLK}$ %01: $f_{PWM}=2/683\ f_{CLK}$ %10: $f_{PWM}=2/512\ f_{CLK}$ %11: $f_{PWM}=2/410\ f_{CLK}$	
15 14 13 12 11 10 9 8	*PWM_ GRAD*	User defined regulation loop gradient (bits 15…12 currently unused, set to 0)	pwm_ autoscale=0	0: StealthChop disabled 1…15: StealthChop enabled (the actual value is not used)
			pwm_ autoscale=1	0: StealthChop disabled 1…15: User defined maximum PWM amplitude change per half wave (1 to 15)
7 6 5 4 3 2 1 0	*PWM_ AMPL*	User defined amplitude	pwm_ autoscale=0	User defined PWM amplitude The resulting amplitude (0…255) is set by this value.
			pwm_ autoscale=1	User defined maximum PWM amplitude when switching back from current chopper mode to voltage PWM mode (switch over velocity defined by TPWMTHRS). Do not set too low values, as the regulation cannot measure the current when the actual PWM value goes below a setting specific value. Settings above 0x40 recommended.

6 Current Setting

The internal 5V supply voltage available at the pin 5VOUT is used as a reference for the coil current regulation based on the sense resistor voltage measurement. The desired maximum motor current is set by selecting an appropriate value for the sense resistor. The sense resistor voltage range can be selected by the *vsense* bit in *CHOPCONF*. The low sensitivity setting (high sense resistor voltage, *vsense*=0) brings best and most robust current regulation, while high sensitivity (low sense resistor voltage, *vsense*=1) reduces power dissipation in the sense resistor. The high sensitivity setting reduces the power dissipation in the sense resistor by nearly half.

After choosing the *vsense* setting and selecting the sense resistor, the currents to both coils are scaled by the 5-bit current scale parameters (*IHOLD*, *IRUN*). The sense resistor value is chosen so that the maximum desired current (or slightly more) flows at the maximum current setting (*IRUN* = %11111).

Using the internal sine wave table, which has the amplitude of 248, the RMS motor current can be calculated by:

$$I_{RMS} = \frac{CS+1}{32} * \frac{V_{FS}}{R_{SENSE}+20m\Omega} * \frac{1}{\sqrt{2}}$$

The momentary motor current is calculated by:

$$I_{MOT} = \frac{CUR_{A/B}}{248} * \frac{CS+1}{32} * \frac{V_{FS}}{R_{SENSE}+20m\Omega}$$

CS is the current scale setting as set by the *IHOLD* and *IRUN* and CoolStep.
V_{FS} is the full-scale voltage as determined by *vsense* control bit (please refer to electrical characteristics, V_{SRTL} and V_{SRTH}).
$CUR_{A/B}$ is the actual value from the internal sine wave table.

The internal resistance of 20mΩ will be increased by external trace resistance, 5mΩ are realistic.

CHOICE OF R		

Parameter	Description	Setting	Comment
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 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.	0 … 31	scaling factor 1/32, 2/32, … 32/32
IHOLD	Identical to *IRUN*, but for motor in stand still.		
IHOLD DELAY	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..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. A *IHOLDDELAY* setting of 4 thus results in a power down time of 4*15*2^18 clock cycles, i.e., roughly one second at 16MHz.	0	instant *IHOLD*
		1 …15	$1*2^{18}$ … $15*2^{18}$ clocks per current decrement
vsense	Allows control of the sense resistor *voltage range* for full scale current.	0	V_{FS} = 0.32 V
		1	V_{FS} = 0.18 V

6.1 Sense Resistors

Sense resistors should be carefully selected. The full motor current flows through the sense resistors. They also see the switching spikes from the MOSFET bridges. A low-inductance type such as film or composition resistors is required to prevent spikes causing ringing on the sense voltage inputs leading to unstable measurement results. A low-inductance, low-resistance PCB layout is essential. Any common GND path for the two sense resistors must be avoided, because this would lead to coupling between the two current sense signals. A massive ground plane is best. Please also refer to layout considerations in chapter 20.

The sense resistor needs to be able to conduct the peak motor coil current in motor standstill conditions unless standby power is reduced. Under normal conditions, the sense resistor conducts less than the coil RMS current, because no current flows through the sense resistor during the slow decay phases.

The peak sense resistor power dissipation is:

$$P_{RSMAX} = {I_{COIL}}^2 * R_{SENSE}$$

For high current applications, power dissipation is halved by using the low *vsense* setting and using an adapted resistance value. Please be aware, that in this case any voltage drop in PCB traces has a larger influence on the result. A compact layout with massive ground plane is best to avoid parasitic resistance effects.

7 StealthChop™

StealthChop is an extremely quiet mode of operation for low and medium velocities. It is based on a voltage mode PWM. In case of standstill and at low velocities, the motor is absolutely noiseless. Thus, StealthChop operated stepper motor applications are very suitable for indoor or home use. The motor operates 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. There are no more configurations required except for the regulation of the PWM voltage to yield the motor target current. Two algorithms are provided, a manual and an automatic mode.

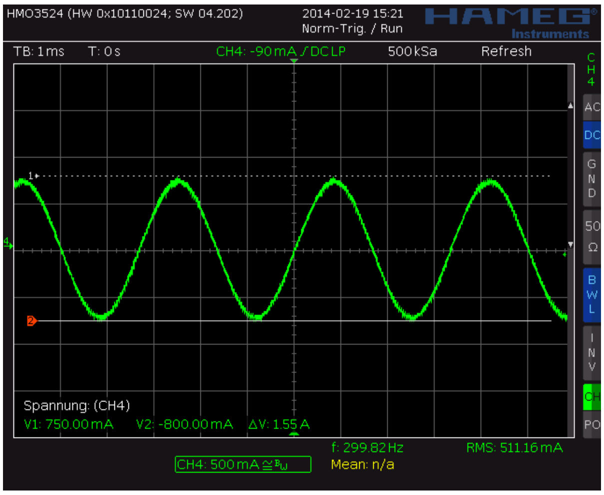


Figure 7.1 Motor coil sine wave current with StealthChop (measured with current probe)

7.1 Two Modes for Current Regulation

In order to match the motor current to a certain level, the voltage mode PWM voltage must be 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 (directly proportional to its velocity) as well as actual level of the supply voltage. For the ease of use, two modes of PWM regulation are provided: An automatic mode using current feedback (*pwm_autoscale* = 1) and a fixed scale mode (*pwm_autoscale* = 0). The fixed scale mode will not react to a change of operating conditions like 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, velocity and supply voltage are well known. Since this mode does not measure the actual current, it will not respond to modification of the current setting, like stand still current reduction. Therefore, we recommend the automatic mode, unless current regulation is not satisfying in the given operating conditions.

The PWM frequency can be chosen in a range in four steps to adapt the frequency divider to the frequency of the clock source. A setting in the range of 30-50kHz is good for many applications. It balances low current ripple and good higher velocity performance vs. dynamic power dissipation.

CHOICE OF PWM FREQUENCY FOR STEALTHCHOP

Clock frequency f_{CLK}	**PWM_FREQ=%00** f_{PWM}=2/1024 f_{CLK}	**PWM_FREQ=%01** f_{PWM}=2/683 f_{CLK}	**PWM_FREQ=%10** f_{PWM}=2/512 f_{CLK}	**PWM_FREQ=%11** f_{PWM}=2/410 f_{CLK}
18MHz	35.2kHz	52.7kHz	70.3kHz	87.8kHz
16MHz	31.3kHz	46.9kHz	62.5kHz	78.0kHz
(internal)	~26kHz	~38kHz	~52kHz	~64kHz
12MHz	23.4kHz	35.1kHz	46.9kHz	58.5kHz
10MHz	19.5kHz	29.3kHz	39.1kHz	48.8kHz
8MHz	15.6kHz	23.4kHz	31.2kHz	39.0kHz

Table 7.1 Choice of PWM frequency – green: recommended

7.2 Automatic Scaling

In StealthChop voltage PWM mode, the autoscaling function (*pwm_autoscale* = 1) regulates the motor current to the desired current setting. The driver measures the motor current during the chopper on time and uses a proportional regulator to regulate the *PWM_SCALE* in order match the motor current to the target current. *PWM_GRAD* 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, the motor velocity or effects resulting from changes of the supply voltage. As the supply voltage level and motor temperature normally change only slowly, a minimum setting of the regulation gradient often is sufficient (*PWM_GRAD*=1). If StealthChop operation is desired for a higher velocity range, variations of the motor back EMF caused by motor acceleration and deceleration may require a quicker regulation. *PWM_GRAD* setting should be optimized for the fastest required acceleration and deceleration ramp (see Figure 7.4). The quality of a given setting can be examined when monitoring *PWM_SCALE* and motor velocity. Just as in the acceleration phase, during a deceleration phase the voltage PWM amplitude must be adapted to keep the motor coil current constant. When the upper acceleration and the upper deceleration used in the application are identical, the value determined for the acceleration phase will already be optimum for both.

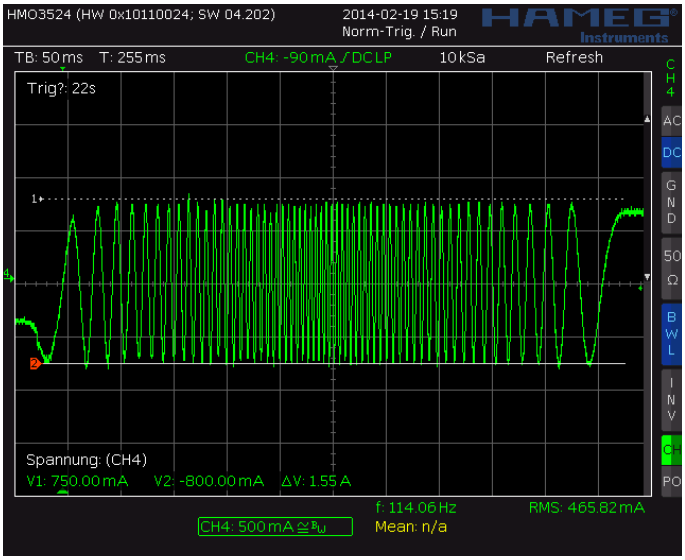


Figure 7.2 Scope shot: good setting for PWM_GRAD

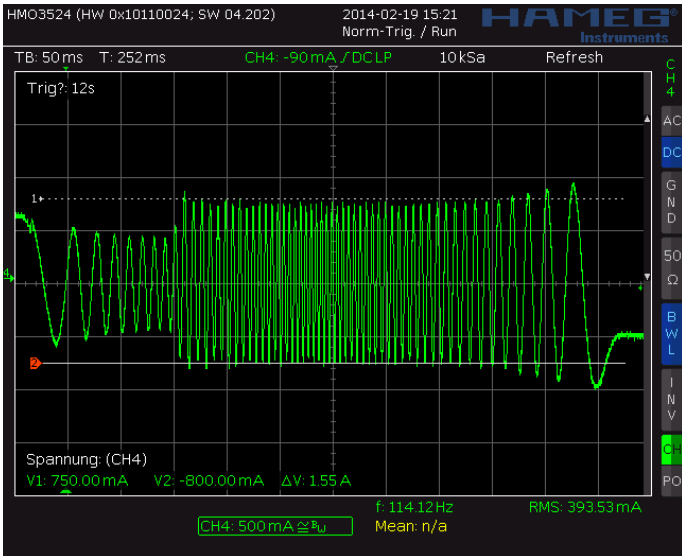


Figure 7.3 Scope shot: too small setting for PWM_GRAD

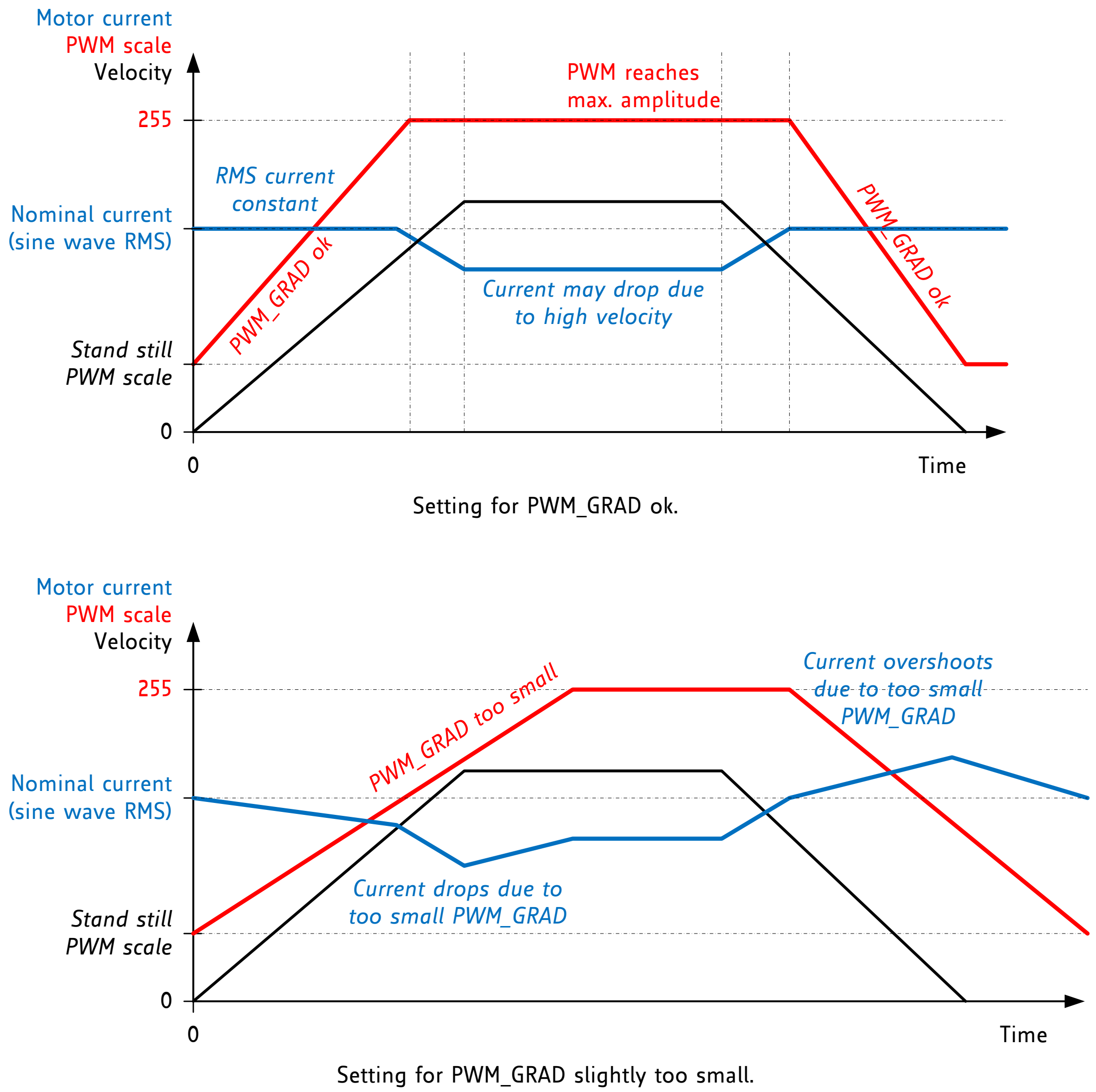


Figure 7.4 Good and too small setting for PWM_GRAD

Be sure to use a symmetrical sense resistor layout and sense resistor traces of identical length and well matching sense resistors for best performance.

Quick Start
For a quick start, see the Quick Configuration Guide in chapter 14.

7.2.1 Lower Current Limit

The autoscaling function imposes a lower limit for motor current regulation. As the coil current can be measured in the shunt resistor during chopper on phase, only, 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 StealthChop autoscaling mode rises with the supply voltage and the chopper frequency. A lower blanking time allows a lower current limit. Extremely low currents (e.g., for standstill power down) can be realized with the non-automatic current scaling or with the freewheeling option, only. The run current setting needs to be kept above the lower limit: In case the *PWM_SCALE* drops to a too low value, e.g., because the current scale was too low, the regulator may not be able to recover. The regulator will recover once the motor is in standstill. The freewheeling option allows going to zero motor current.

The lower motor coil current limit can be calculated from motor parameters and chopper settings:

$$I_{Lower\ Limit} = t_{BLANK} * f_{PWM} * \frac{V_M}{R_{COIL}}$$

With V_M the motor supply voltage and R_{COIL} the motor coil resistance.
$I_{Lower\ Limit}$ can be treated as a thumb value for the minimum possible motor current setting.

Example:

A motor has a coil resistance of 5Ω, the supply voltage is 24V. With *TBL*=%01 and *PWM_FREQ*=%00, t_{BLANK} is 24 clock cycles, f_{PWM} is 2/(1024 clock cycles):

$$I_{Lower\ Limit} = 24\ t_{CLK} * \frac{2}{1024\ t_{CLK}} * \frac{24V}{5\Omega} = \frac{24}{512} * \frac{24V}{5\Omega} = 225mA$$

Attention
For *pwm_autoscale* mode, a lower coil current limit applies. This limit can be calculated or measured using a current probe. Keep the motor run-current setting *IRUN* well above this lower current limit.

7.2.2 PWM_AMPL for Using StealthChop and SpreadCycle

When combining StealthChop with SpreadCycle or constant off time classic PWM, a switching velocity can be chosen using *VCOOLTHRS*. With this, StealthChop is only active at low velocities. Often, a very low velocity in the range of 1 to a few 10 RPM fits best. In case a high switching velocity is chosen, special care should be taken for switching back to StealthChop during deceleration, because the phase jerk can produce a short time overcurrent. (Refer to chapter 7.4 for more details about combining StealthChop with other chopper modes.)

To avoid a short time overcurrent and to minimize the jerk, the initial amplitude for switching back to StealthChop at sinking velocity can be determined using the setting *PWM_AMPL*. Tune *PWM_AMPL* to a value which gives a smooth and safe transition back to StealthChop within the application. As a thumb rule, ½ to ¾ of the last *PWM_SCALE* value which was valid after the switching event at rising velocity can be used. For high resistive steppers as well as for low transfer velocities (as set by *VCOOLTHRS*), *PWM_AMPL* can be set to 255 as most universal setting.

Note
The autoscaling function only starts up regulation during motor standstill. After enabling StealthChop and setting all parameters, be sure to wait until *PWM_SCALE* has reached a stable state before starting a motion. Failure to do so will result in zero motor current!

In case the automatic scaling regulation is instable at your desired motion velocity, try modifying the chopper frequency divider *PWM_FREQ*. Also adapt the blank time *TBL* and motor current for best result.

7.2.3 Acceleration

In automatic current regulation mode (*pwm_autoscale* = 1), the *PWM_GRAD* setting should be optimized for the fastest required acceleration ramp. Use a current probe and check the motor current during (quick) acceleration. A setting of 1 may result in a too slow regulation, while a setting of 15 responds very quickly to velocity changes but might produce regulation instabilities in some constellations. A setting of 4 is a good starting value.

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

7.3 Fixed Scaling

Non-automatic, fixed scaling scales the StealthChop amplitude based on the user defined value *PWM_AMPL*. 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, U the supply voltage scaled by the PWM value, the current I results. The initial value for PWM_AMPL at low velocities can be calculated:

$$PWM\_AMPL = \frac{374 * R_{COIL} * I_{COIL}}{V_M}$$

With V_M the motor supply voltage and I_{COIL} the target RMS current

The effective PWM voltage U_{PWM} (1/SQRT(2) x peak value) results considering the 8 bit resolution and 248 sine wave peak for the actual PWM amplitude shown as *PWM_SCALE*:

$$U_{PWM} = V_M * \frac{PWM\_SCALE}{256} * \frac{248}{256} * \frac{1}{\sqrt{2}} = V_M * \frac{PWM\_SCALE}{374}$$

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. A higher scale value is necessary to compensate for this. When a higher value is chosen, it should be made sure, that the maximum driver current is not exceeded during the acceleration phase. This can be checked with the above formula. A short time excess current will not do harm to the motor.

Hint

The setting for *PWM_AMPL* can easily be optimized by tracing the motor current with a current probe on the oscilloscope. It is not even necessary to calculate the formulas if you carefully start with a low setting for both.

7.4 Combining StealthChop with other Chopper Modes

The TMC5041 allows combining StealthChop and different chopper modes based on velocity thresholds. This way, the optimum chopper principle can be chosen for different velocity ranges. As a first step, both chopper principles should be parameterized and optimized individually. In a next step, a transfer velocity has to be fixed. For example, StealthChop operation is used for precise low speed positioning, while SpreadCycle shall be used for highly dynamic motion. *VCOOLTHRS* determines the transition velocity. Use a low transfer velocity to avoid a jerk at the switching point. A jerk occurs when switching at higher velocities, because the back-EMF of the motor (which rises with the velocity) causes a phase shift of up to 90° between motor voltage and motor current. So, when switching at higher velocities between voltage PWM and current PWM mode, this jerk will occur with increased intensity. At low velocities (e.g., 1 to a few 10 RPM), it can be completely neglected for most motors. Therefore, the *VCOOLTHRS* should be set to a low velocity, to eliminate any jerk in case an automatic switching between two chopper modes is desired. Set *VCOOLTHRS* and *VHIGH* to high values (above *VMAX*) if you want to work with StealthChop only.

When enabling the StealthChop mode the first time using automatic current regulation, the motor must be at stand still to allow a proper current regulation. When the drive switches to a different chopper mode at a higher velocity, StealthChop 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 StealthChop 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 might 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 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.

Hint

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

7.5 Flags in StealthChop

7.5.1 Open Load Flags

In StealthChop mode, status information is different from the cycle-by-cycle regulated chopper modes. OLA and OLB show if the current regulation sees that the nominal current can be reached on both coils.

- A flickering OLA or OLB can result from tiny asymmetries in the sense resistors or in the motor coils.
- An interrupted motor coil leads to a continuously active open load flag for the coil.
- Both flags are active, if the current regulation did not succeed in scaling up to the full target current within the last few fullsteps (because no motor is attached, or a high acceleration required a quick action of the current regulator).
  With automatic scaling and *PWM_GRAD* > 1, the current regulation tries to increase the current quickly to reach the target current in the interrupted motor coil. At the same time but a bit slower the current regulation tries to decrease the motor current due to the other motor coil seeing too high current.

Therefore, it is recommended to do an on-demand open load test using the SpreadCycle or classic chopper prior to operation in StealthChop, and not to switch on StealthChop in case of open load failure. Alternatively, PWM_SCALE can be checked for plausible values.

7.5.2 PWM_SCALE Informs about the Motor State

Information about the motor state is available with automatic scaling by reading out *PWM_SCALE*. 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* value allows seeing the motor load (similar to StallGuard2) and finding out if the target current can be reached. It even gives an idea on the motor temperature (evaluate at a well-known state of operation).

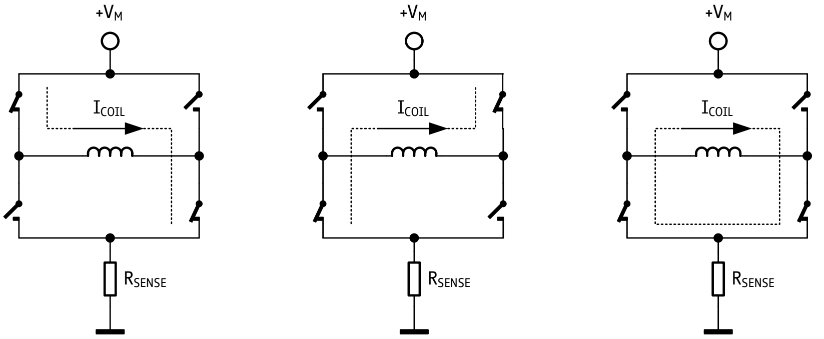
7.6 Freewheeling and Passive Motor Braking

StealthChop provides different options for motor standstill. These options can be enabled by setting the standstill current *IHOLD* to zero and choosing the desired option using the *FREEWHEEL* setting. The desired option becomes enabled after a time period specified by *TZEROWAIT* and *IHOLD_DELAY*. The *PWM_SCALE* regulation becomes frozen once the motor target current is at zero current to ensure a quick startup.

Parameter	Description	Setting	Comment
VCOOLTHRS *VHIGH*	Whichever is lower, specifies the upper velocity for operation in StealthChop voltage PWM mode.	0 … 2^23-1	
pwm_ autoscale	Enable automatic current scaling using current measurement or use fixed scaling mode.	0	Fixed mode
		1	Automatic scaling with current regulator
PWM_FREQ	PWM frequency selection. StealthChop uses a fixed PWM frequency by dividing the system clock frequency using a programmable divider. Use the lowest setting giving good results.	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_GRAD	Global enable and regulation loop gradient when *pwm_autoscale*=1.	0	Do not use StealthChop
		1 … 15	StealthChop enabled
PWM_AMPL	User defined PWM amplitude for fixed scaling or amplitude limit for re-entry into StealthChop mode when *pwm_autoscale*=1.	0 … 255	
FREEWHEEL	Stand still option when motor current setting is zero (*I_HOLD*=0). Only available with StealthChop enabled. The freewheeling option makes the motor easy movable, while both coil short options realize a passive brake. Mode 2 will brake more intensely than mode 3, because low side drivers (LS) have lower resistance than high side drivers.	0	Normal operation
		1	Freewheeling
		2	Coil shorted using LS drivers
		3	Coil shorted using HS drivers
PWM_SCALE	Read back of the actual StealthChop voltage PWM scaling as determined by the current regulation. Can be used to detect motor load and stall when *autoscale*=1.	0 … 255 (read-only)	The scaling value becomes frozen when operating in a different chopper mode
TOFF	General enable for the motor driver, the actual value does not influence StealthChop	0	Driver off
		1 … 15	Driver enabled
TBL	Selects the comparator *blank time*. This time needs to safely cover the switching event and the duration of the ringing on the sense resistor. For most applications, a setting of 1 or 2 is good. For highly capacitive loads, e.g. when filter networks are used, a setting of 2 or 3 will be required. A lower setting allows StealthChop to regulate down to lower coil current values.	0	16 t_{CLK}
		1	24 t_{CLK}
		2	36 t_{CLK}
		3	54 t_{CLK}
IRUN *IHOLD*	Run and hold current setting for stealth Chop operation – only used with *pwm_autoscale*=1		See chapter on current setting for details

8 SpreadCycle and Classic Chopper

While StealthChop 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 independently of each other. In Figure 8.1 the different chopper phases are shown.



On Phase:
current flows in
direction of target
current

Fast Decay Phase:
current flows in
opposite direction
of target current

Slow Decay Phase:
current re-circulation

Figure 8.1 Chopper 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 can measure coil current during phases when the current flows through the sense resistor, but not during the slow decay phase, so 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 may be terminated by either the comparator or another timer.

When the coil current is switched, spikes at the sense resistors occur due to charging and discharging parasitic capacitances. 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: a new high-performance chopper algorithm called SpreadCycle and a proven constant off-time chopper mode. 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.

The chopper frequency is an important parameter for a chopped motor driver. A too low frequency might generate audible noise. A higher frequency reduces current ripple in the motor, but with a too high frequency 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 are optimally working in a frequency range of 16 kHz to 30 kHz. The chopper frequency is influenced by a number of parameter settings as well as by the motor inductivity and supply voltage.

Hint

A chopper frequency in the range of 16 kHz to 30 kHz gives a good result for most motors when using SpreadCycle. A higher frequency leads to increased switching losses. It is advised to check the resulting frequency and to work below 50 kHz.

Three parameters are used for controlling both chopper modes:

Parameter	Description	Setting	Comment
TOFF	Sets the slow decay time (*off time*). This setting also limits the maximum chopper frequency. For operation with StealthChop, this parameter is not used, but it is required to enable the motor. In case of operation with StealthChop only, any setting is OK. Setting this parameter to zero completely disables all driver transistors and the motor can free-wheel.	0	chopper off
		1…15	off time setting N_{CLK}= 24 + 32**TOFF* (1 will work with minimum blank time of 24 clocks)
TBL	Selects the comparator *blank time*. This time needs to safely cover the switching event and the duration of the ringing on the sense resistor. For most applications, a setting of 1 or 2 is good. For highly capacitive loads, e.g., when filter networks are used, a setting of 2 or 3 will be required.	0	16 t_{CLK}
		1	24 t_{CLK}
		2	36 t_{CLK}
		3	54 t_{CLK}
chm	Selection of the *chopper mode*	0	SpreadCycle
		1	classic const. off time

8.1 SpreadCycle Chopper

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

Each chopper cycle is comprised of an on phase, a slow decay phase, a fast decay phase and a second slow decay phase (see Figure 8.3). 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.

Calculation of a starting value for the slow decay time *TOFF*:

Assumptions:
Target Chopper frequency: 25kHz
Two slow decay cycles make up for 50% of overall chopper cycle time

$$t_{OFF} = \frac{1}{25kHz} * \frac{50}{100} * \frac{1}{2} = 10\mu s$$

For the *TOFF* setting this means:

$$TOFF = (t_{OFF} * f_{CLK} - 24)/32$$

With 12 MHz clock this gives a setting of TOFF=3.0, i.e. 3.
With 16 MHz clock this gives a setting of TOFF=4.25, i.e. 4 or 5.

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 will allow the chopper to precisely regulate the current both for rising and for 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 will lead to a lower chopper frequency. The motor inductance limits the ability of the chopper to follow a changing motor current. Further 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 either with a current probe or by probing the sense resistor voltages (see Figure 8.2). Checking the sine wave shape near zero transition will show a small ledge between both half waves in case the hysteresis setting is too small. At medium velocities (i.e., 100 to 400 fullsteps per second), a too low hysteresis setting will lead to increased humming and vibration of the motor.

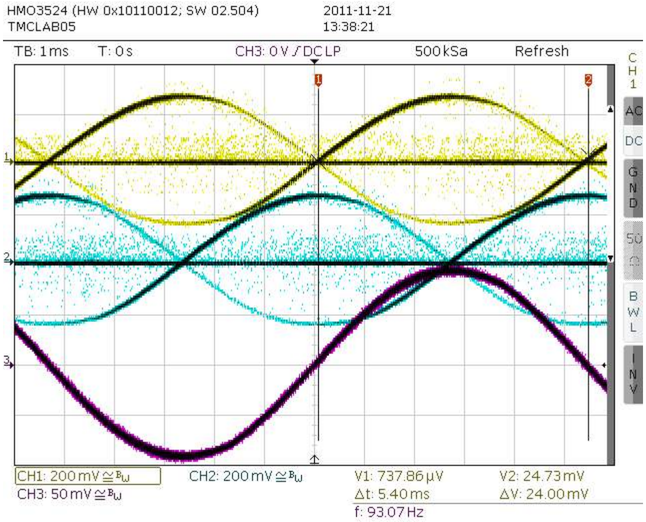


Figure 8.2 No ledges in current wave with sufficient hysteresis (magenta: current A, yellow & blue: sense resistor voltages A and B)

A too high hysteresis setting will lead to reduced chopper frequency and increased chopper noise but will not yield any benefit for the wave shape.

Quick Start
For a quick start, see the Quick Configuration Guide in chapter 14.
For detail procedure see Application Note AN001 - *Parameterization of SpreadCycle*

As experiments show, the setting is quite independent of the motor, because higher current motors typically also have a lower coil resistance. Therefore, choosing a low to medium default value for the hysteresis (for example, effective hysteresis = 4) normally fits most applications. The setting can be optimized by experimenting with the motor: A too low setting will result in reduced microstep accuracy, while a too high setting will lead to more chopper noise and motor power dissipation. When measuring the sense resistor voltage in motor standstill at a medium coil current with an oscilloscope, a too low setting shows a fast decay phase not longer than the blanking time. When the fast decay time becomes slightly longer than the blanking time, the setting is optimum. You can reduce the off-time setting, if this is hard to reach.

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 automatic 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. This way, the chopper frequency is stabilized at high amplitudes and low supply voltage situations, if the frequency gets too low. This avoids the frequency reaching the audible range.

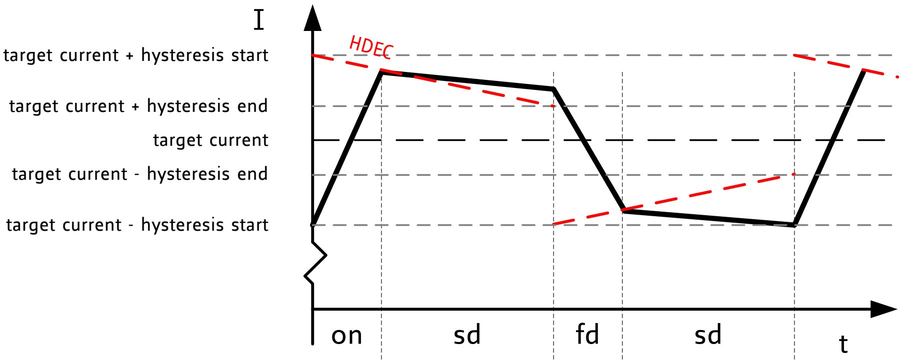


Figure 8.3 SpreadCycle chopper scheme showing coil current during a chopper cycle

Two parameters control SpreadCycle mode:

Parameter	Description	Setting	Comment
HSTRT	*Hysteresis start* setting. This value is an offset from the hysteresis end value *HEND*.	0…7	*HSTRT*=1…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.	0…2	-3…-1: negative HEND
		3	0: zero HEND
		4…15	1…12: positive HEND

Even at HSTRT=0 and HEND=0, the TMC5041 sets a minimum hysteresis via analog circuitry.

Example:

In the example a hysteresis of 4 has been chosen. You might decide to not use hysteresis decrement. In this case set:

HEND=6 (sets an effective end value of 6-3=3)
HSTRT=0 (sets minimum hysteresis, i.e. 1: 3+1=4)

In order to take advantage of the variable hysteresis, we can set most of the value to the HSTRT, i.e. 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)

Hint
Highest motor velocities sometimes benefit from setting TOFF to 1, 2 or 3 and a short TBL of 1 or 0.

8.2 Classic Constant Off Time Chopper

The classic constant off time chopper is an alternative to SpreadCycle. Perfectly tuned, it also gives good results. Also, the classic constant off time chopper is beneficial when used in combination with fullstepping, i.e., at high velocity.

The classic 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. 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.

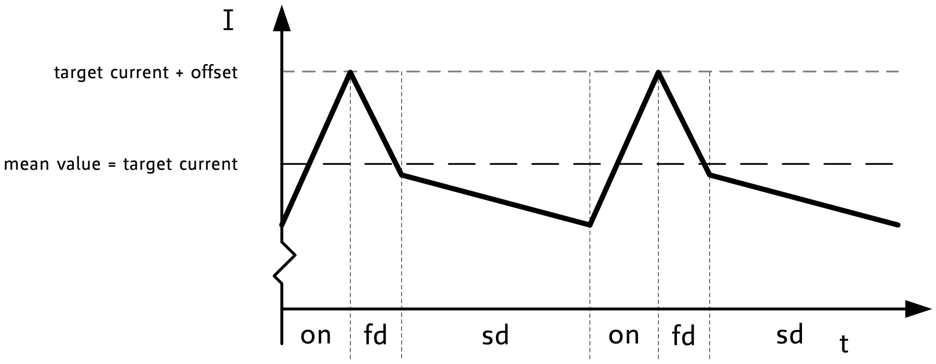
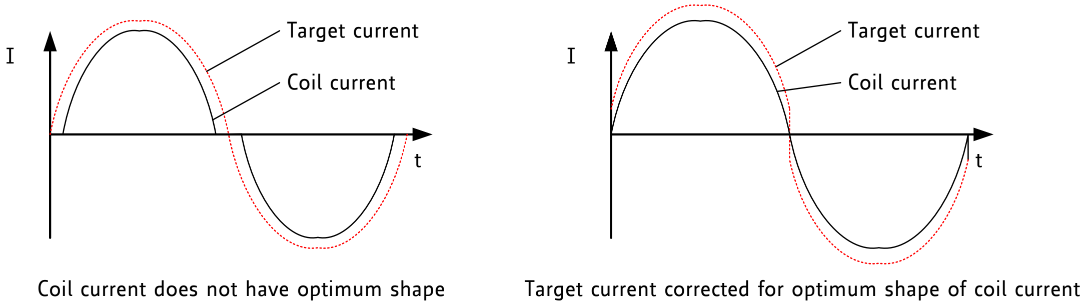


Figure 8.4 Classic const. off time chopper with offset showing coil current

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 (see Figure 8.5). 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.



Coil current does not have optimum shape

Target current corrected for optimum shape of coil current

Figure 8.5 Zero crossing with classic chopper and correction using sine wave offset

Three parameters control constant off-time mode:

Parameter	Description	Setting	Comment
TFD (*fd3* & *HSTRT*)	*Fast decay time* setting. With CHM=1, these bits control the portion of fast decay for each chopper cycle.	0	slow decay only
		1…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.	0…2	negative offset: -3…-1
		3	no offset: 0
		4…15	positive offset 1…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.	0	enable comparator termination of fast decay cycle
		1	end by time only

8.3 Random Off Time

In the constant off-time chopper mode, both coil choppers run freely without synchronization. The frequency of each chopper mainly depends on the coil current and the motor coil inductance. The inductance varies with the microstep position. With some motors, a slightly audible beat can occur between the chopper frequencies when they are close together. This typically occurs at a few microstep positions within each quarter wave. This effect is usually not audible when compared to mechanical noise generated by ball bearings, etc. Another factor which can cause a similar effect is a poor layout of the sense resistor GND connections.

Hint
A common factor, which can cause motor noise, is a bad PCB layout causing coupling of both sense resistor voltages (please refer layouts hint in chapter 20).

To minimize the effect of a beat between both chopper frequencies, an internal random generator is provided. It modulates the slow decay time setting when switched on by the *rndtf* bit. The *rndtf* feature further spreads the chopper spectrum, reducing electromagnetic emission on single frequencies.

Parameter	Description	Setting	Comment
rndtf	This bit switches on a *random off time* generator, which slightly modulates the off-time *TOFF* using a random polynomial.	0	disable
		1	random modulation enable

9 Driver Diagnostic Flags

The TMC5041 drivers supply a complete set of diagnostic and protection capabilities, like 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* table for details.

9.1 Temperature Measurement

The driver integrates a two-level temperature sensor (120°C pre-warning and 150°C thermal shutdown) for diagnostics and for protection of the IC against excess heat. Heat is mainly generated by the motor driver stages, and, at increased voltage, by the internal voltage regulator. Most critical situations, where the driver MOSFETs could be overheated, are avoided when enabling the short to GND protection. For many applications, the overtemperature pre-warning will indicate an abnormal operation situation and can be used to initiate user warning or power reduction measures like motor current reduction. The thermal shutdown is just an emergency measure and temperature rising to the shutdown level should be prevented by design.

After triggering the overtemperature sensor (*ot* flag), the driver remains switched off until the system temperature falls below the pre-warning level (*otpw*) to avoid continuous heating to the shutdown level.

9.2 Short to GND Protection

The TMC5041 power stages are protected against a short circuit condition 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 becomes switched off, and the *s2ga* or *s2gb* flag becomes 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 rather undefined, and a complex network of external components may be involved. Therefore, short circuits should basically be avoided.

9.3 Open Load Diagnostics

Interrupted cables are a common cause for systems failing, e.g., when connectors are not firmly plugged. The TMC5041 detects open load conditions by checking if it can reach the desired motor coil current. This way, also undervoltage conditions, high motor velocity settings or short and overtemperature conditions may cause triggering of the open load flag, and inform the user, that motor torque may suffer. In motor stand still, open load cannot be measured, as the coils might eventually have zero current.

Open load detection is provided for system debugging.
To safely detect an interrupted coil connection, operate in SpreadCyle, and check the open load flags following a motion of minimum four times the selected microstep resolution into a single direction using low or nominal motor velocity operation, only. However, the *ola* and *olb* flags have just informative character and do not cause any action of the driver.

10 Ramp Generator

The ramp generator allows motion based on target position or target velocity. It automatically calculates the optimum motion profile taking into account acceleration and velocity settings. The TMC5041 integrates a new type of ramp generator, which offers faster machine operation compared to the classical linear acceleration ramps. The SixPoint ramp generator allows adapting the acceleration ramps to the torque curves of a stepper motor and uses two different acceleration settings each for the acceleration phase and for the deceleration phase. See Figure 10.2.

10.1 Real World Unit Conversion

The TMC5072 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}. 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.

The units of a TMC5041 register content are written as register[5041].

PARAMETER VS. UNITS

Parameter / Symbol	Unit	calculation / description / comment
f_{CLK}[Hz]	[Hz]	clock frequency of the TMC5041 in [Hz]
s	[s]	second
US	µstep	
FS	fullstep	
µstep velocity v[Hz]	µsteps / s	v[Hz] = v[5041] * (f_{CLK}[Hz]/2 / 2^23)
µstep acceleration a[Hz/s]	µsteps / s^2	a[Hz/s] = a[5041] * f_{CLK}[Hz]^2 / (512*256) / 2^24
USC microstep count	counts	microstep resolution in number of microsteps (i.e. the number of microsteps between two fullsteps – normally 256)
rotations per second v[rps]	rotations / s	v[rps] = v[µsteps/s] / USC / FSC FSC: motor fullsteps per rotation, e.g. 200
rps acceleration a[rps/s^2]	rotations / s^2	a[rps/s^2] = a[µsteps/s^2] / USC / FSC
ramp steps[µsteps] = rs	µsteps	rs = (v[5041])^2 / a[5041] / 2^8 microsteps during linear acceleration ramp (assuming acceleration from 0 to v)

In rare cases, the upper acceleration limit might 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 may be decreased. Setting the *CHOPCONF* option *MRES*=%0001 will double the motor velocity for the same speed setting and thus also double effective acceleration and deceleration. The motor will have half position resolution with this setting.

Quick Start
For a quick start, see the Quick Configuration Guide in chapter 14.

10.2 Motion Profiles

For the ramp generator register set, please refer to the chapter 5.2.

10.2.1 Ramp Mode

The ramp generator delivers two phase acceleration and two-phase deceleration ramps with additional programmable start and stop velocities (see Figure 10.1).

> *Note*
> The start velocity can be set to zero, if not used.
>
> The stop velocity can be set to ten (or down to one), if not used.
>
> 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.

The two different sets of acceleration and deceleration can be combined freely. *A common transition speed V1 allows for velocity dependent switching between both acceleration and deceleration settings*. A typical use case will use lower acceleration and deceleration values at higher velocities, as the motors torque declines at higher velocity. When considering friction in the system, it becomes clear, that typically deceleration of the system is quicker than acceleration. Thus, deceleration values can be higher in many applications. This way, operation speed of the motor in time critical applications can be maximized.

As target positions and ramp parameters may be changed any time during the motion, the motion controller will always use the optimum (fastest) way to reach the target, while sticking to the constraints set by the user. This way it might happen, that the motion becomes automatically stopped, crosses zero and drives back again. This case is flagged by the special flag *second_move.*

10.2.2 Start and Stop Velocity

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*. As the motor probably is not able to follow this, you can set a time delay for a subsequent move by setting *TZEROWAIT*. An active delay time is flagged by the flag *t_zerowait_active*. Once the target position is reached, the flag *position_reached* becomes active.

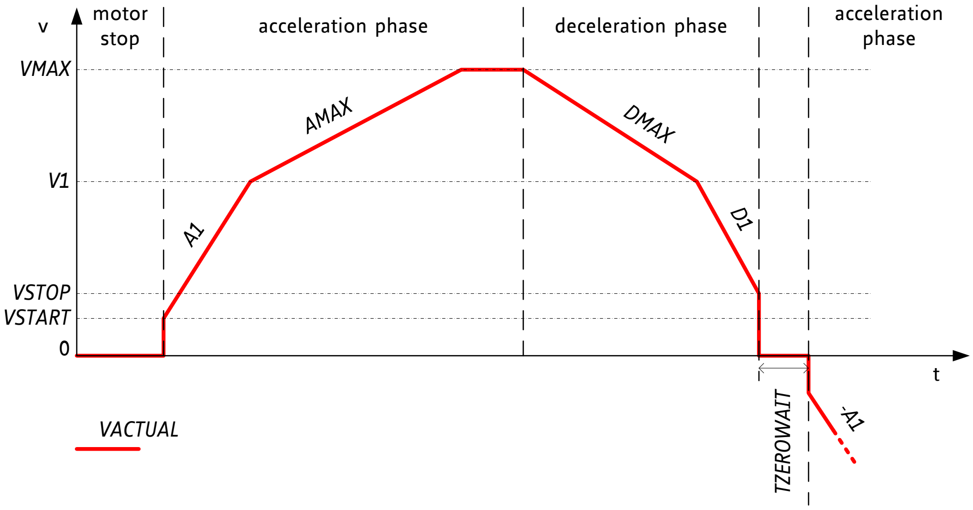


Figure 10.1 Ramp generator velocity trace showing consequent move in negative direction

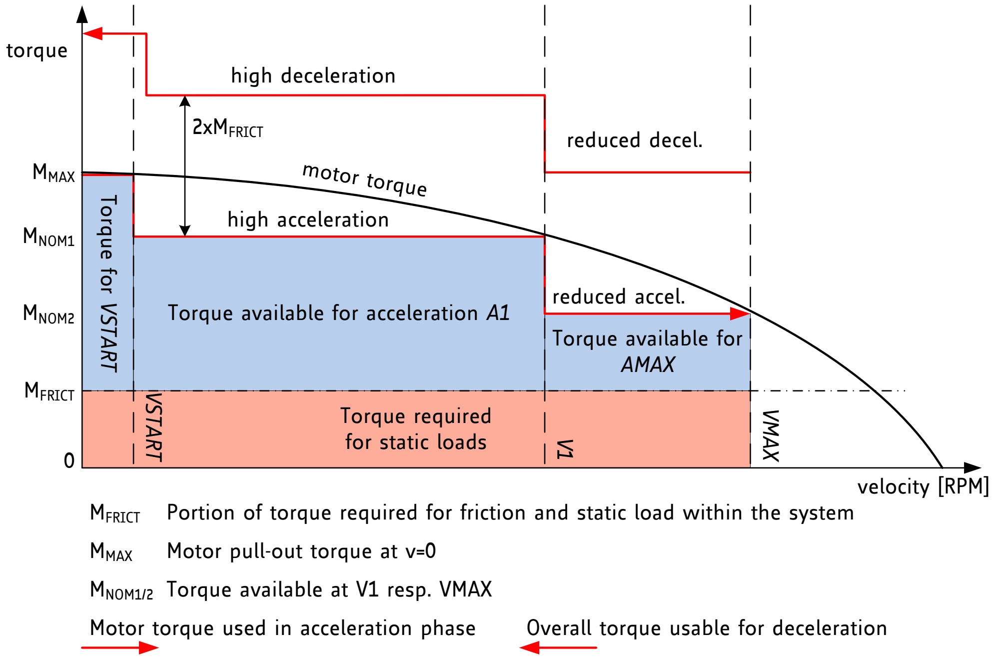


Figure 10.2 Illustration of optimized motor torque usage with TMC5041 ramp generator

10.2.3 Velocity Mode

For the ease of use, velocity mode movements do not use the different acceleration and deceleration settings. You need to set *VMAX* and *AMAX* only for velocity mode. The ramp generator always uses *AMAX* to accelerate or decelerate to *VMAX* in this mode.

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

10.2.4 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.

OPTIONS TO TERMINATE MOTION USING ACCELERATION SETTINGS:

a) Switch to velocity mode, set *VMAX*=0 and *AMAX* to the desired deceleration value. This will stop the motor using a linear ramp.
b) For a stop in positioning mode, set *VSTART*=0 and *VMAX*=0. *VSTOP* is not used in this case. The driver will use *AMAX* and *A1* (as determined by *V1*) for going to zero velocity.
c) For a stop using *D1*, *DMAX* 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 will decelerate and eventually come to a stop. Poll the actual velocity to terminate motion during *TZEROWAIT* time using option a) or b).
d) Activate a stop switch. This can be done by means of the hardware input, e.g., using a wired 'OR' to the REF switch input. If you do not use the hardware input and have tied the REFL and REFR 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.

10.2.5 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 never is left.

REALIZE JOYSTICK CONTROL

1. Use positioning mode to control the motion direction and to set the motion limit(s).
2. Modify *VMAX* at any time in the range *VSTART* to your maximum value. With *VSTART*=0, you can also stop motion by setting *VMAX*=0. The motion controller will use *A1* and *AMAX* as determined by *V1* to adapt velocity for ramping up and ramping down.
3. In case you do not modify the acceleration settings, you do not need to rewrite *XTARGET*, just modify *VMAX*.
4. *DMAX*, *D1* and *VSTOP* only become used when the ramp controller slows down due to reaching the target position, or when the target position has been modified to point to the other direction.

10.3 Interrupt Handling

The motion controllers provide the capability to issue an interrupt to the microcontroller, e.g., to react on a position reached event. In case more than one interrupt source is possible, it is necessary to carefully check for the actual event, without risking losing an event.

INTERRUPT HANDLING FOR 2 AXIS (EXAMPLE FOR TARGET_REACHED):

1. Read *RAMP_STAT1* to clear the interrupt flags. This will turn off the interrupt source.
2. Check *XACTUAL1* for reaching of the target position (and any other conditions you want to check for ramp 1).
3. Do the same for *RAMP_STAT2* and *XACTUAL2*.

This way, you are sure that you will not miss any *target_reached* condition, because you first clear the flags, and afterwards read out the condition.

10.4 Velocity Thresholds

The ramp generator provides a number of velocity thresholds coupled to the actual velocity *VACTUAL*. The different ranges allow programming the motor to the optimum step mode, coil current and acceleration settings. For the range labeled "microstepping" in Figure 10.3, either StealthChop or SpreadCycle can be used, if enabled.

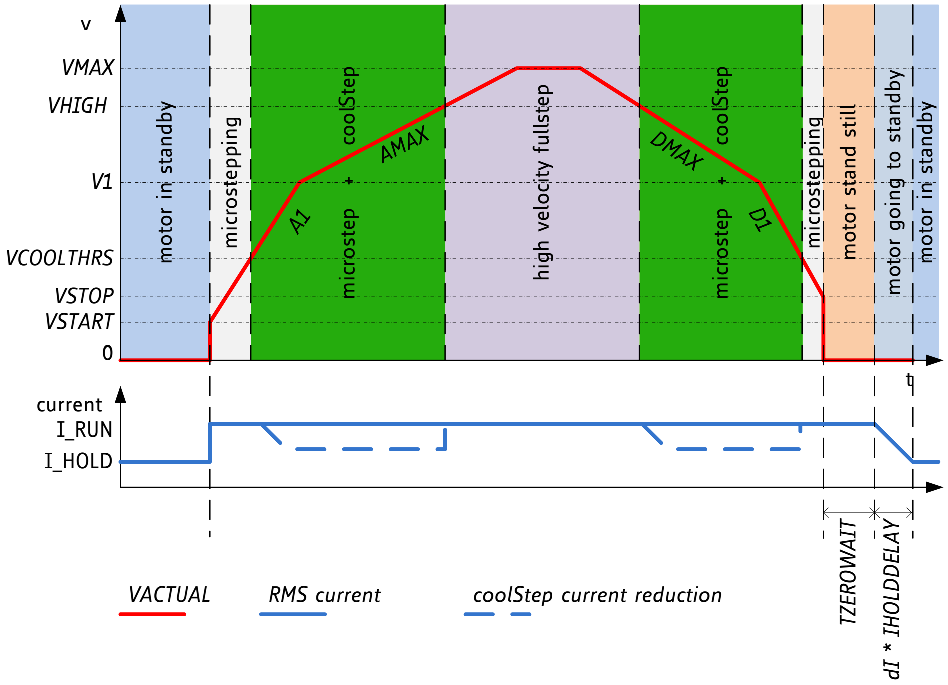


Figure 10.3 Ramp generator velocity dependent motor control

Note
Since it is not necessary to differentiate the velocity to the last detail, the velocity thresholds use a reduced number of bits for comparison and the lower eight bits of the compare values become ignored.

10.5 Reference Switches

Prior to normal operation of the drive an absolute reference position must be set. The reference position can be found using a mechanical stop which can be detected by stall detection, or by a reference switch.

In case of a linear drive, the mechanical motion range must not be left. This can be ensured also 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:

- it can be stopped abruptly when a switch is hit. This is useful in an emergency case and for StallGuard based homing.
- Or the motor can be softly decelerated to zero using deceleration settings (DMAX, V1, D1).

Hint
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.

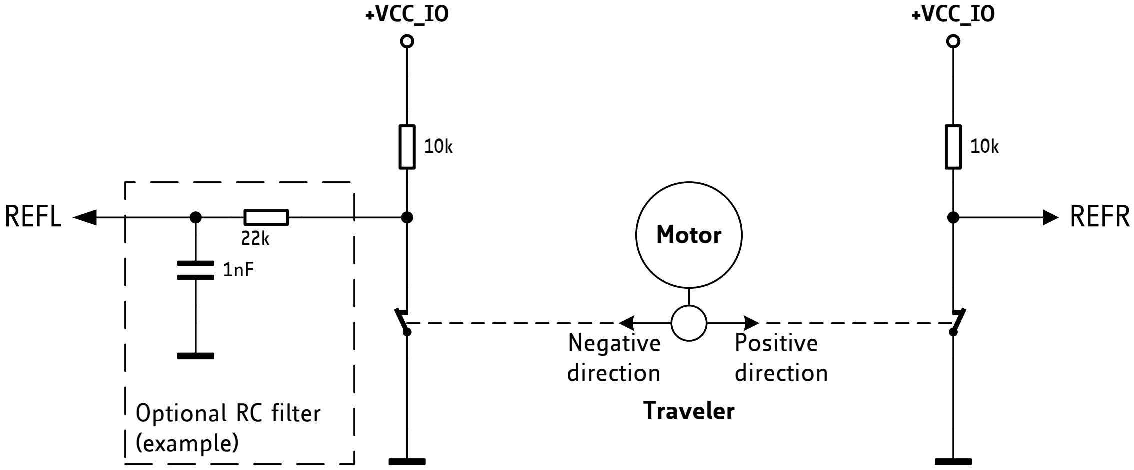


Figure 10.4 Using reference switches (example)

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, or
- hall sensors.

Be careful to select reference switch resistors matching your switch requirements!
In case of long cables additional RC filtering might be required near the TMC5041 reference inputs. Adding an RC filter will also reduce the danger of destroying the logic level inputs by wiring faults, but it will add a certain delay which should be considered with respect to the application.

IMPLEMENTING A HOMING PROCEDURE

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. StallGuard 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, to a more positive position for a right switch). You may timeout this motion 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. Please be aware that reading *RAMP_STAT* may clear flags (e.g., *sg_stop*) and thus the motor may restart after expiration of *TZEROWAIT*. In case the stop condition might be reset by the read and clear (R+C) function, be sure to execute step 5 within the time range set by *TZEROWAIT*.
5. Switch the ramp generator to hold mode and calculate the difference between the latched position and the actual position. For StallGuard 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. Now, homing is finished. A move to position 0 will bring back the motor exactly to the switching point. In case StallGuard was used for homing, a read access to *RAMP_STAT* clears the StallGuard stop event *event_stop_sg* and releases the motor from the stop condition.

10.6 Ramp Generator Response Time

The ramp generator is realized in hardware and executes commands within less than a microsecond, switching over to the desired mode and target values taking effect. The 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 it's stop (VSTOP), the actual step pulse rate is very low. Therefore, a significant delay can add for execution of the first and last steps, as determined by the selected microstep velocity. For example, a microstep velocity of 10Hz means, that 100ms expire in between of each two steps. As (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 a range of minimum 100 to 1000 for quick ramp termination (100 yields roughly <10ms, 1000 roughly <1ms).

11 StallGuard2 Load Measurement

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 shown in Figure 11.1. At maximum motor load, the value goes to zero or near to zero. This corresponds to a load angle of 90° between the magnetic field of the coils and magnets in the rotor. This also is the most energy-efficient point of operation for the motor.

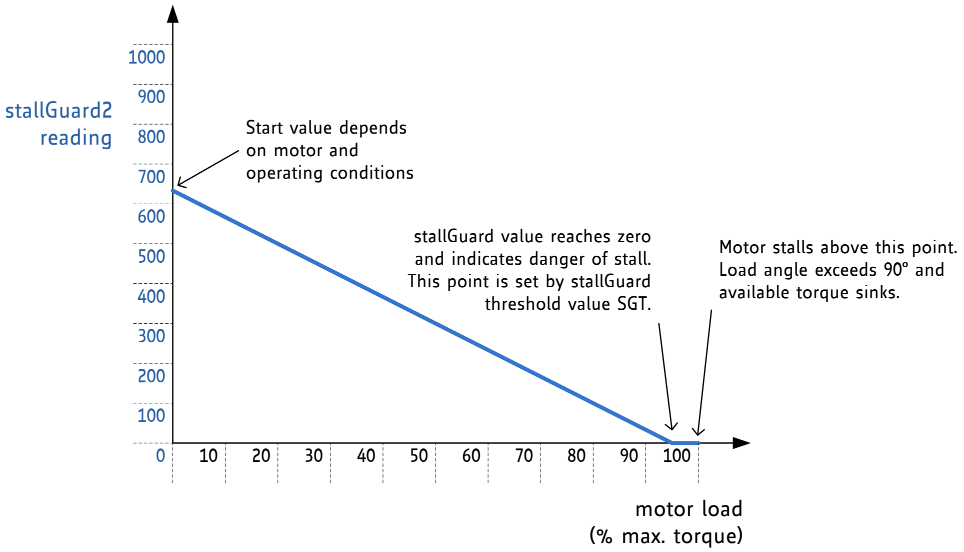


Figure 11.1 Function principle of StallGuard2

Parameter	**Description**	**Setting**	**Comment**
SGT	This signed value controls the StallGuard2 threshold level for stall detection and sets the optimum measurement range for readout. A lower value gives a higher sensitivity. Zero is the starting value working with most motors. A higher value makes StallGuard2 less sensitive and requires more torque to indicate a stall.	0	indifferent value
		+1… +63	less sensitivity
		-1… -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 (4 fullsteps).	0	standard mode
		1	filtered mode
Status word	**Description**	**Range**	**Comment**
SG	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* reading of roughly 0 to 100 at maximum load before motor stall.	0… 1023	0: highest load low value: high load high value: less load

In order to use StallGuard2 and CoolStep, the StallGuard2 sensitivity should first be tuned using the SGT setting!

11.1 Tuning StallGuard2 Threshold SGT

The StallGuard2 value *SG* is affected by motor-specific characteristics and application-specific demands on load and velocity. 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 STALLGUARD SGT

1. Operate the motor at the normal operation velocity for your application and monitor *SG*.
2. Apply slowly increasing mechanical load to the motor. If the motor stalls before *SG* reaches zero, decrease *SGT*. If *SG* reaches zero before the motor stalls, increase *SGT*. A good *SGT* starting value is zero. *SGT* is signed, so it can have negative or positive values.
3. Now enable *sg_stop* and make sure, that the motor is safely stopped whenever it is stalled. Increase *SGT* if the motor becomes stopped before a stall occurs. Restart the motor by disabling *sg_stop* or by reading the *RAMP_STAT* register (read and clear function).
4. The optimum setting is reached when *SG* is between 0 and roughly 100 at increasing load shortly before the motor stalls, and *SG* increases by 100 or more without load. *SGT* in most cases can be tuned for a certain motion velocity or a velocity range. Make sure, that the setting works reliable 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 basic idea behind the *SGT* setting is a factor, which compensates the StallGuard 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. This way, *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 < 10 RPM (i.e. a few to a few fullsteps per second) and target operation current and supply voltage. In this velocity range, there is not much dependence of *SG* on the motor load, because the motor does not generate significant back EMF. Therefore, mechanical load will not make a big difference on the result.
2. Switch on *sfilt*. Now increase *SGT* starting from 0 to a value, where *SG* starts rising. With a high *SGT*, *SG* will rise up to the maximum value. Reduce again to the highest value, where *SG* stays at 0. Now the *SGT* value is set as sensibly as possible. When you see *SG* increasing at higher velocities, there will be useful stall detection.

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.

SG 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 *VCOOLTHRS* to match the lower velocity threshold where StallGuard delivers a good result to use *sg_stop*.

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

Quick Start
For a quick start, see the Quick Configuration Guide in chapter 14.
For detail procedure see Application Note AN002 - *Parameterization of StallGuard2 & CoolStep*

11.1.1 Variable Velocity Limits *VCOOLTHRS* and *VHIGH*

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.

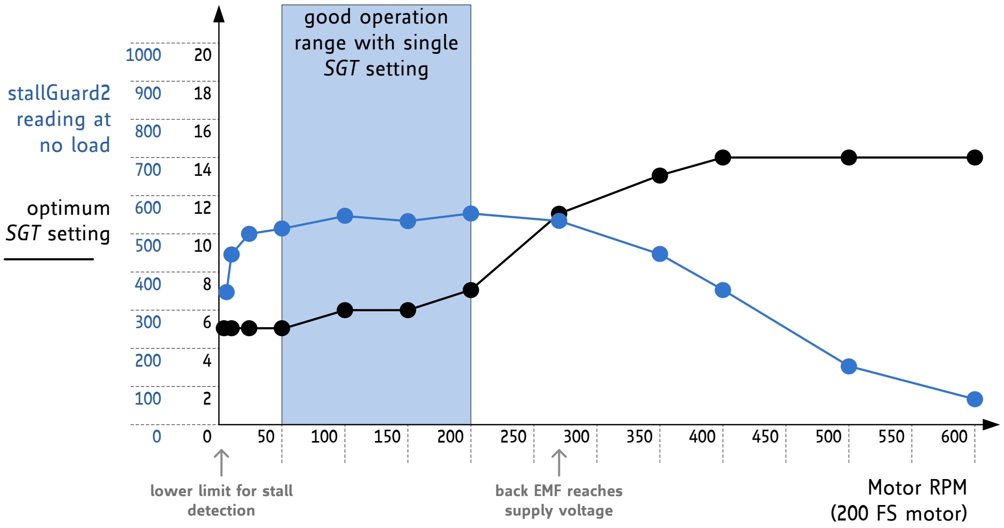


Figure 11.2 Example: Optimum SGT setting and StallGuard2 reading with an example motor

In many applications, operation at or near a single operation point is used most of the time and a single setting is sufficient. The ramp generator provides a lower (*VCOOLTHRS*) and an upper velocity threshold (*VHIGH*) to match this. The stall detection is automatically disabled outside the determined operation point, e.g., during acceleration phases preceding a sensorless homing procedure when setting *VCOOLTHRS* to a matching value. An upper limit can be specified by *VHIGH*.

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

11.1.2 Small Motors with High Torque Ripple and Resonance

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

11.1.3 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* at increasing temperature, as motor efficiency is reduced.

11.1.4 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:

$$stallGuard\ measurement\ error = \pm max(1, |SGT|)$$

11.2 StallGuard2 Update Rate and Filter

The StallGuard2 measurement value *SG* 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 fullstep 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 StallGuard.

11.3 Detecting a Motor Stall

For best stall detection, work without StallGuard 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* value at this load, e.g., some value within the range 0 to 100. The stall threshold should 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* value without load and with maximum load. They should show a difference of at least 100 or a few 100, which shall be large compared to the offset. If you set the *SGT* value in a way, that a reading of 0 occurs at maximum motor load, the stall can be automatically detected by the motion controller to issue a motor stop. In the moment of the step resulting in a step loss, the lowest reading will be visible. After the step loss, the motor will vibrate and show a higher *SG* reading.

11.4 Homing with StallGuard

The homing of a linear drive requires moving the motor into the direction of a hard stop. As StallGuard needs a certain velocity to work (as set by *VCOOLTHRS*), 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 also is indicated by the flag *StallGuard* in *DRV_STATUS*. After setting up new motion parameters to prevent the motor from restarting right away, StallGuard can be disabled, or the motor can be re-enabled by reading *RAMP_STAT*. The read and clear function of the *event_stop_sg* flag in *RAMP_STAT* would restart the motor after expiration of *TZEROWAIT* in case the motion parameters have not been modified.

11.5 Limits of StallGuard2 Operation

StallGuard2 does not operate reliably at extreme motor velocities: Very low motor velocities (for many motors, less than one revolution per second) generate a low back EMF and make the measurement unstable and dependent on environment conditions (temperature, etc.). The automatic tuning procedure described above will compensate for this. Other conditions will also lead to extreme settings of *SGT* and poor response of the measurement value *SG* 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.

12 CoolStep Operation

CoolStep is an automatic smart energy optimization for stepper motors based on the motor mechanical load, making them "green".

12.1 User Benefits

Energy efficiency – consumption decreased up to 75%
Motor generates less heat – improved mechanical precision
Less cooling infrastructure – for motor and driver
Cheaper motor – does the job!

CoolStep allows substantial energy savings, especially for motors which see varying loads or operate at a high duty cycle. Because a stepper motor application needs to work with a torque reserve of 30% to 50%, even a constant-load application allows significant energy savings because CoolStep automatically enables torque reserve when required. Reducing power consumption keeps the system cooler, increases motor life, and allows reducing cost in the power supply and cooling components.

Reducing motor current by half results in reducing power by a factor of four.

12.2 Setting up for CoolStep

CoolStep is controlled by several parameters, but two are critical for understanding how it works:

Parameter	Description	Range	Comment
SEMIN	4-bit unsigned integer that sets a *lower threshold*. If *SG* goes below this threshold, 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* value. (The name of this parameter is derived from smartEnergy, which is an earlier name for CoolStep.)	0	disable CoolStep
		1…15	threshold is *SEMIN**32
SEMAX	4-bit unsigned integer that controls an *upper threshold*. If SG is sampled equal to or above this threshold enough times, CoolStep decreases the current to both coils. The upper threshold is (*SEMIN* + *SEMAX* + 1)*32.	0…15	threshold is (*SEMIN*+*SEMAX*+1)*32

Figure 12.1 shows the operating regions of CoolStep:

- The black line represents the *SG* 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* falls below *SEMIN*, and CoolStep increases the current. When the load decreases, *SG* rises above (*SEMIN* + *SEMAX* + 1) * 32, and the current is reduced.

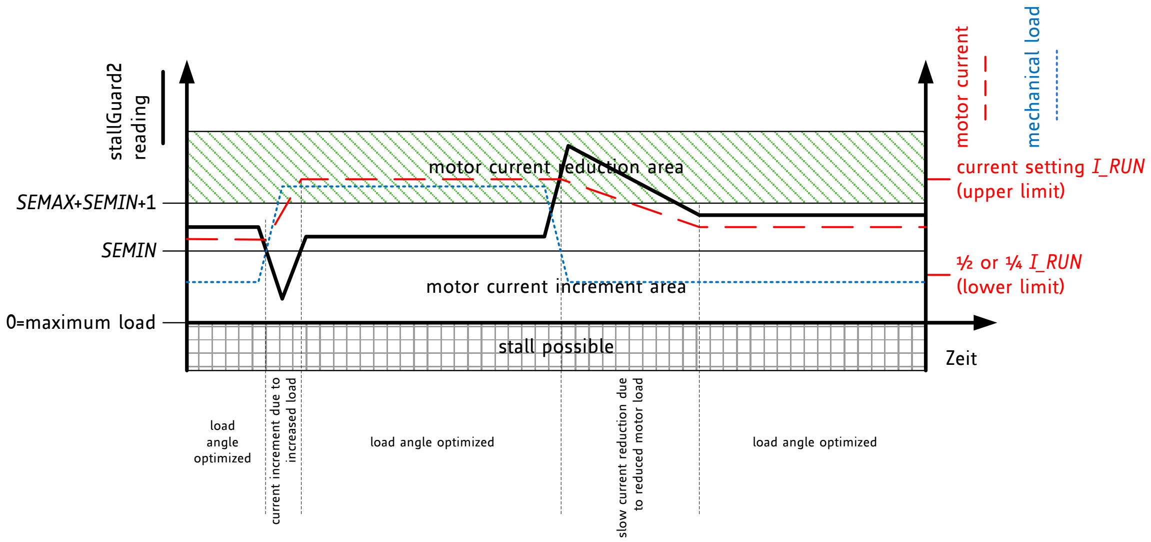


Figure 12.1 CoolStep adapts motor current to the load

Five more parameters control CoolStep and one status value is returned:

Parameter	**Description**	**Range**	**Comment**
SEUP	Sets the *current increment step*. The current becomes incremented for each measured StallGuard2 value below the lower threshold.	0…3	step width is 1, 2, 4, 8
SEDN	Sets the number of StallGuard2 readings above the upper threshold necessary for each *current decrement* of the motor current.	0…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.	0 1	0: 1/2 of IRUN 1: 1/4 of IRUN
VCOOL THRS	Lower ramp generator velocity threshold. Below this velocity CoolStep becomes disabled (not used in Step/Dir mode). Adapt to the lower limit of the velocity range where StallGuard2 gives a stable result. *Hint:* May be adapted to disable CoolStep during acceleration and deceleration phase by setting identical to *VMAX*.	1… 2^23	
VHIGH	Upper ramp generator velocity threshold value. Above this velocity CoolStep becomes disabled (not used in Step/Dir mode). Adapt to the velocity range where StallGuard2 gives a stable result.	1… 2^23	Also controls additional functions like switching to fullstepping.
Status word	**Description**	**Range**	**Comment**
CSACTUAL	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…31	1/32, 2/32, … 32/32

12.3 Tuning CoolStep

Before tuning CoolStep, first tune the StallGuard2 threshold level *SGT*, which affects the range of the load measurement value *SG*. CoolStep uses *SG* to operate the motor near the optimum load angle of +90°.

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.

12.3.1 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.

Hint

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 according. As acceleration and decelerations normally shall be quick, they will require the full motor current, while they have only a small contribution to overall power consumption due to their short duration.

12.3.2 Low Velocity and Standby Operation

Because CoolStep is not able to measure the motor load in standstill and at very low RPM, a lower velocity threshold is provided in the ramp generator. It should be set to an application specific default value. Below this threshold the normal current setting via *IRUN* respectively *IHOLD* is valid. An upper threshold is provided by the *VHIGH* setting. Both thresholds can be set as a result of the StallGuard2 tuning process.

13 Sine-Wave Look-up Table

The TMC5041 driver provides a programmable look-up table for storing the microstep current wave. It is common to both drivers. As a default, the table is pre-programmed 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.

13.1 User Benefits

Microstepping	–	extremely improved with low-cost motors
Motor	–	runs smooth and quiet
Torque	–	reduced mechanical resonances yields improved torque

13.2 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 one 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 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.

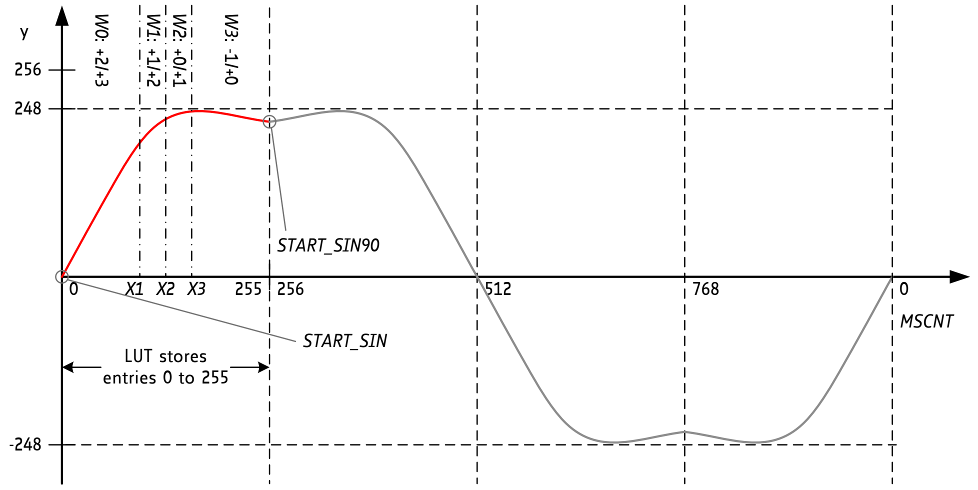


Figure 13.1 LUT programming example

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.

Two registers control the starting values of the tables:

- 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*.
- In the same way, 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 2-phase motor.

Hint
Refer chapter 5.3 for the register set and for the default table function stored in the drivers. The default table is a good base for realizing an own table.
The TMC5041-EVAL comes with a calculation tool for own waves.

Initialization example for the 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= 0, *START_SIN90*= 247

13.3 Changing Resolution

Reduced microstep resolution might be desired in some cases, e.g., for testing purpose or to increase the effective acceleration or position range. The internal microstep table uses 1024 sine wave entries to generate the wave. The step width taken within the table depends on the microstep resolution setting. Depending on the motion direction, the microstep counter is increased or decreased with each internal step pulse by the step width. In principle, the microstep resolution can be changed at any time. The microstep resolution determines the increment respectively the decrement, the TMC5041 uses for advancing in the microstep table. At maximum resolution, it advances one step for each step pulse. At half resolution, it advances two steps and so on. This way, a change of resolution is possible transparently at each time. When switching to a low resolution, the effective current wave may become asymmetrical. This will be avoided when first moving to a certain position before switching. E.g., when switching to 16 microsteps, it is optimum to first position to *MSCNT*=8. This way, the microsteps -7 and +8 are nearly symmetrical to the current zero crossing.

14 Quick Configuration Guide

This guide is meant as a practical tool to come to a first configuration and do a minimum set of measurements and decisions for tuning the driver. It does not cover all advanced functionalities but concentrates on the basic function set to make a motor run smoothly. Once the motor runs, you may decide to explore additional features and functionality in more detail. A current probe on one motor coil is a good aid to find the best settings, but it is not a must.

CURRENT SETTING AND FIRST STEPS WITH STEALTHCHOP

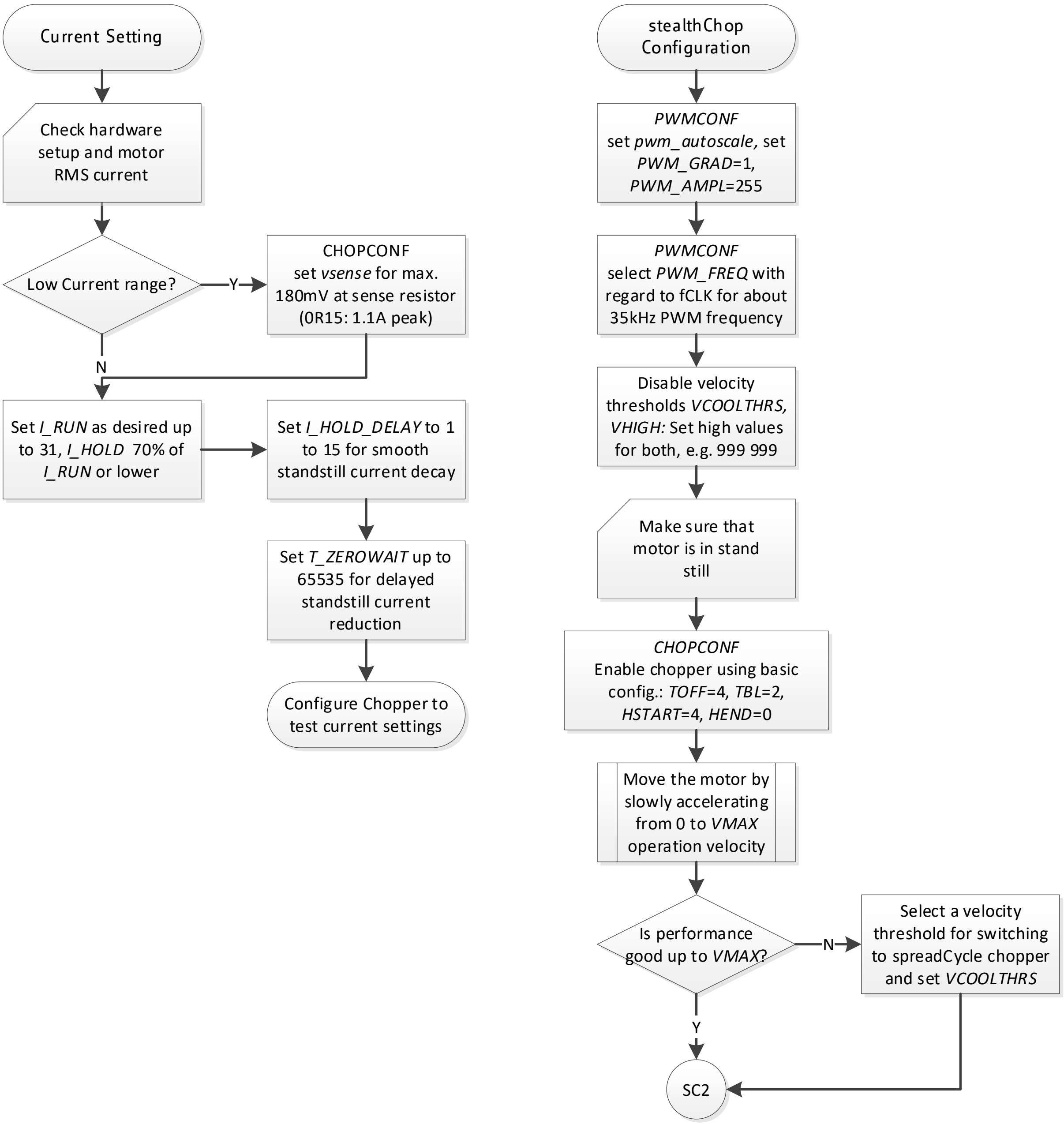


Figure 14.1 Current setting and first steps with StealthChop

TUNING STEALTHCHOP AND SPREADCYCLE

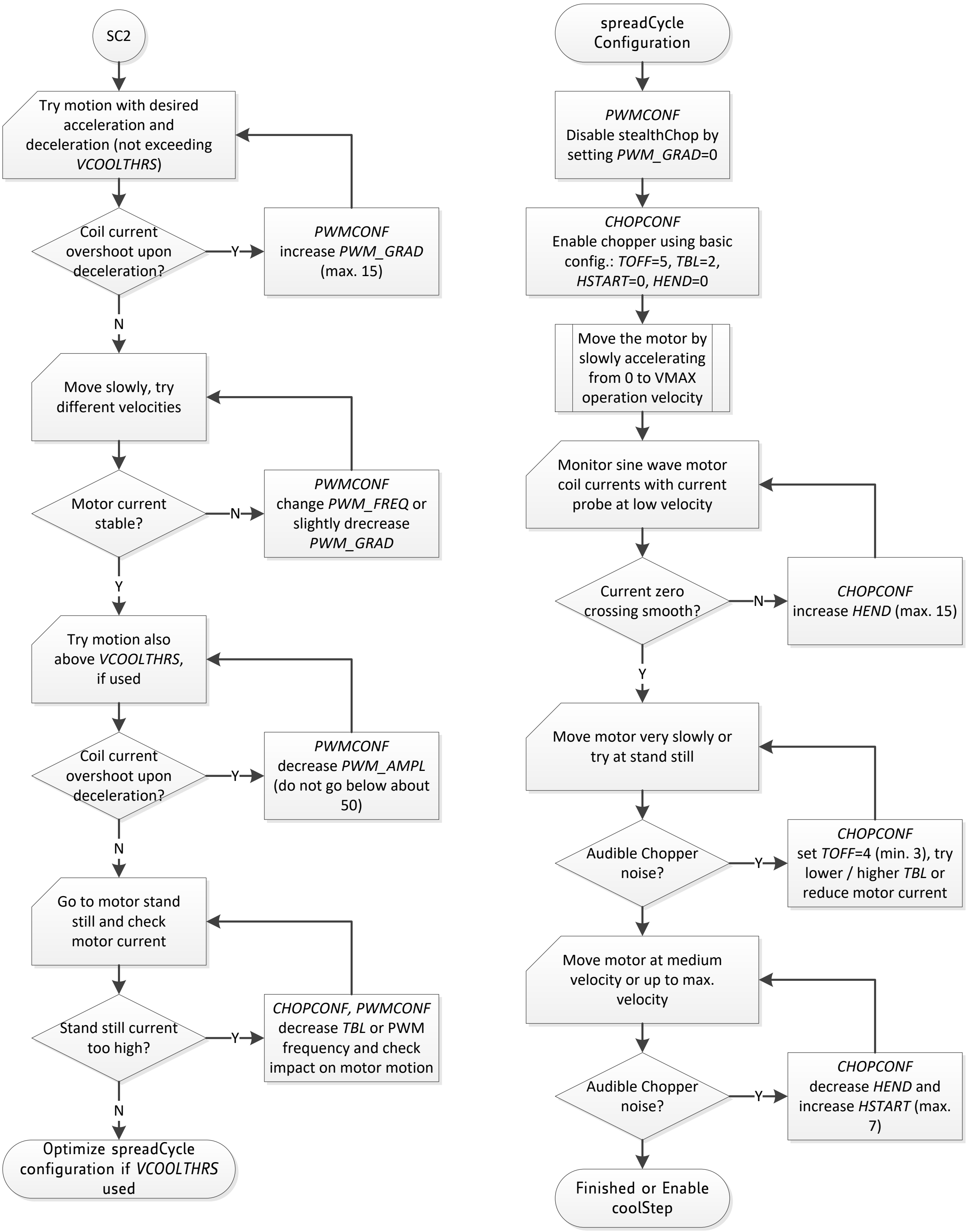


Figure 14.2 Tuning StealthChop and SpreadCycle

MOVING THE MOTOR USING THE MOTION CONTROLLER

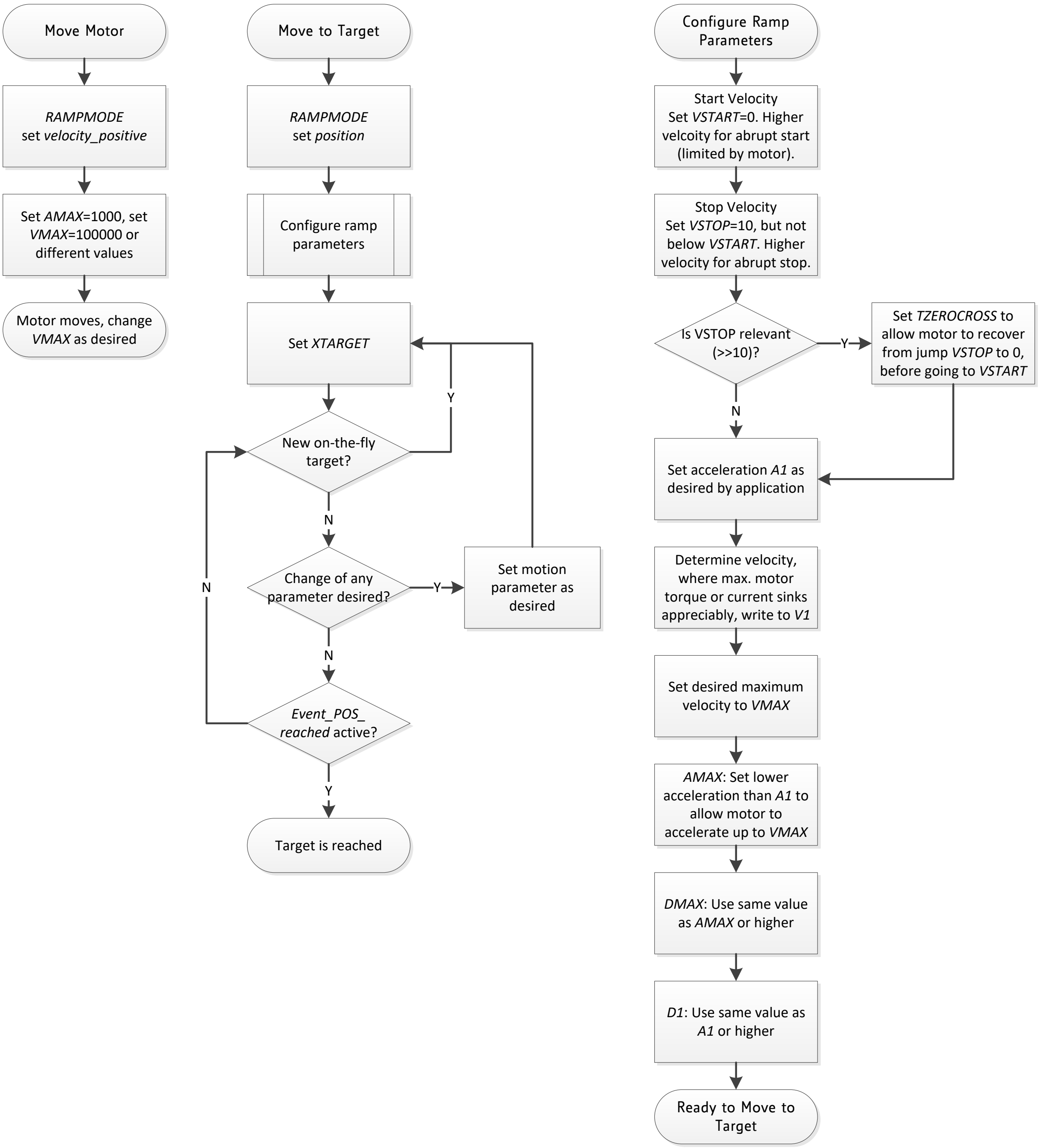


Figure 14.3 Moving the motor using the motion controller

ENABLING COOLSTEP (ONLY IN COMBINATION WITH SPREADCYCLE)

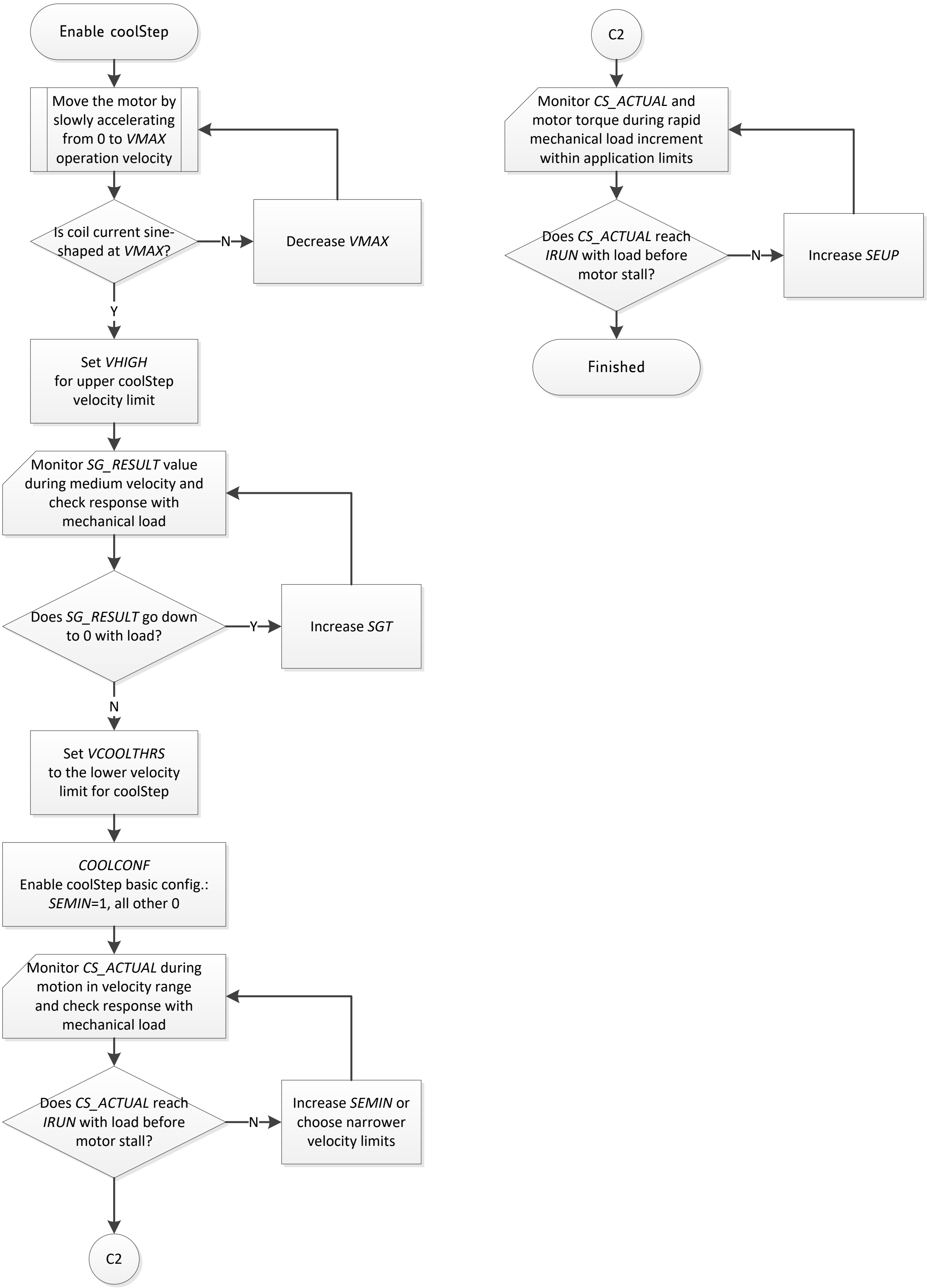


Figure 14.4 Enabling CoolStep (only in combination with SpreadCycle)

15 Getting Started

Please refer to the TMC5041 evaluation board to allow a quick start with the device, and in order to allow interactive tuning of the device setup in your application. Chapter 14 will guide you through the process of correctly setting up all registers.

15.1 Initialization Examples

SPI datagram example sequence to enable and initialize driver 1 for SpreadCycle operation combined with StealthChop at low velocities. Ramp generator 1 moves the motor in velocity mode. Additional read access to the position register:

```
SPI send: 0x8000000008;  // GCONF=8: Enable PP and INT outputs
SPI send: 0xEC000100C5;  // CHOPCONF: TOFF=5, HSTRT=4, HEND=1, TBL=2, CHM=0 (SpreadCycle)
SPI send: 0xB000011F05;  // IHOLD_IRUN: IHOLD=5, IRUN=31 (max. current), IHOLDDELAY=1
SPI send: 0xAC00002710;  // TZEROWAIT=10000
SPI send: 0x90000401C8;  // PWM_CONF: AUTO=1, 2/1024 Fclk, Switch amplitude limit=200, Grad=1
SPI send: 0xB200061A80;  // VHIGH=400 000: Set VHIGH to a high value to allow StealthChop
SPI send: 0xB100007530;  // VCOOLTHRS=30000: Set upper limit for StealthChop to about 30RPM
SPI send: 0xA600001388;  // AMAX=5000
SPI send: 0xA700004E20;  // VMAX=20000
SPI send: 0xA000000001;  // RAMPMODE=1 (positive velocity)

// Now motor 1 should start rotating

SPI send: 0x2100000000;  // Query X Actual – The next read access delivers X Actual
SPI read;                // Read X Actual
```

Initialization SPI datagram example sequence to enable and initialize the motion controller and then move one rotation (51200 microsteps) using the ramp generator.

```
SPI send: 0xA4000003E8; // A1 = 1 000 First acceleration
SPI send: 0xA50000C350; // V1 = 50 000 Acceleration threshold velocity V1
SPI send: 0xA6000001F4; // AMAX = 500 Acceleration above V1
SPI send: 0xA7000304D0; // VMAX = 200 000
SPI send: 0xA8000002BC; // DMAX = 700 Deceleration above V1
SPI send: 0xAA00000578; // D1 = 1400 Deceleration below V1
SPI send: 0xAB0000000A; // VSTOP = 10 Stop velocity (Near to zero)
SPI send: 0xA000000000; // RAMPMODE = 0 (Target position move)
// Ready to move!
SPI send: 0xADFFFF3800; // XTARGET = -51200 (Move one rotation left (200*256 microsteps))
```

Hint
Tune the configuration parameters for your motor and application for optimum performance.

16 Power-Up Reset

The chip is loaded with default values during power-up via its internal power-on reset. It will also reset to power-up defaults in case any of the supply voltages monitored by internal reset circuitry (VSA, +5VOUT or VCC_IO) falls below the undervoltage threshold. VCC is not monitored. Therefore, VCC must not be lost during operation of the chip. In case of a microcontroller software re-boot, disable the driver (*TOFF*=0), re-initialize all registers used by the software and stop any motion in progress by slowing down the ramp generator. A hardware reset requires cycling VCC_IO while keeping all digital inputs at a low level at the same time. Actively drive VCC_IO to a low level to ensure that it falls below the lower reset threshold. Current consumed from VCC_IO is low and therefore it has simple driving requirements. Due to the input protection diodes not allowing the digital inputs to rise above VCC_IO level, any active high input would hinder VCC_IO from going down.

In case, VCC becomes supplied by an external source, make sure that VCC is at a stable value above the lower operation limit once the reset ends. This normally is satisfied when generating a 3.3V VCC_IO from the +5V supply supplying the VCC pin, because it will then come up with a certain delay.

17 Clock Oscillator and Clock Input

The clock is the timing reference for all functions: the chopper, the velocity, the acceleration control, etc. Many parameters are scaled with the clock frequency. Thus, a precise reference allows a more deterministic result. The on-chip clock oscillator provides timing in case no external clock is easily available.

17.1 Using the Internal Clock

Directly tie the CLK input to GND near to the TMC5041 if the internal clock oscillator is to be used. The internal clock can be calibrated by driving the ramp generator at a certain velocity setting. Reading out position values via the interface and comparing the resulting velocity to the remote masters' clock gives a time reference. This allows scaling acceleration and velocity settings as a result. The temperature dependency and ageing of the internal clock is comparatively low.

IMPLEMENTING FREQUENCY DEPENDENT SCALING

Frequency dependent scaling allows using the internal clock for a motion control application. The time reference of the external microcontroller is used to calculate a scaler for all velocity settings. The following steps are required:

1. You may leave the motor driver disabled during the calibration.
2. Start motor in velocity mode, with *VMAX*=10000 and *AMAX*=60000 (for quick acceleration). The acceleration phase is ended after a few ms.
3. Read out *XACTUAL* twice, at time point t1 and time point t2, e.g., 100ms later (dt=0.1s). The time difference between both read accesses shall be exactly timed by the external microcontroller.
4. Stop the motion ramp by setting *VMAX*=0.
5. The number of steps done in between of t1 and t2 now can be used to calculate the factor
$$f = \frac{VMAX \, * \, dt}{XACTUAL(t2) - XACTUAL(t1)} = \frac{1000}{XACTUAL(t2) - XACTUAL(t1)}$$
6. Now multiply each velocity value with this factor f, to normalize the velocity to steps per second. At a nominal value of the internal clock frequency, 780 steps will be done in 100ms.

Hint
In case well defined velocity settings and precise motor chopper operation are desired, it is supposed to work with an external clock source.

17.2 Using an External Clock

When an external clock is available, a frequency of 10 MHz to 16 MHz is recommended for optimum performance. The duty cycle of the clock signal is uncritical, as long as minimum high or low input time for the pin is satisfied (refer to electrical characteristics). Up to 18 MHz can be used, when the clock duty cycle is 50%. 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 with the first positive polarity seen on the CLK input.

Attention
Switching off the external clock frequency prevents the driver from operating normally. Therefore, be careful to switch off the motor drivers before switching off the clock (e.g. using the enable input), because otherwise the chopper would stop and the motor current level could rise uncontrolled. The short to GND detection stays active even without clock, if enabled.

17.3 Considerations on the Frequency

A higher frequency allows faster step rates, faster SPI operation and higher chopper frequencies. On the other hand, it may cause more electromagnetic emission of the system and causes more power dissipation in the TMC5041 digital core and voltage regulator. Generally, a frequency of 10 MHz to 16 MHz should be sufficient for most applications. For reduced requirements concerning the motor dynamics, a clock frequency of down to 8 MHz can be considered.

18 Absolute Maximum Ratings

The maximum ratings may not be exceeded under any circumstances. Operating the circuit at or near more than one maximum rating at a time for extended periods shall be avoided by application design.

Parameter	Symbol	Min	Max	Unit
Supply voltage operating with inductive load ($V_{VS} \geq V_{VSA}$)	V_{VS}, V_{VSA}	-0.5	27	V
Supply and bridge voltage max. *)	V_{VS}	-0.5	28	V
VSA when different from to VS	V_{VSA}	-0.5	V_{VS}+0.5	V
I/O supply voltage	V_{VIO}	-0.5	5.5	V
digital VCC supply voltage (if not supplied by internal regulator)	V_{VCC}	-0.5	5.5	V
Logic input voltage	V_I	-0.5	V_{VIO}+0.5	V
Maximum current to / from digital pins and analog low voltage I/Os	I_{IO}		+/-10	mA
5V regulator output current (internal plus external load)	I_{5VOUT}		50	mA
5V regulator continuous power dissipation (V_{VM}-5V) * I_{5VOUT}	P_{5VOUT}		1	W
Power bridge repetitive output current	I_{Ox}		2.0	A
Junction temperature	T_J	-50	150	°C
Storage temperature	T_{STG}	-55	150	°C
ESD-Protection for interface pins (Human body model, HBM)	V_{ESDAP}		4 (tbd.)	kV
ESD-Protection for handling (Human body model, HBM)	V_{ESD}		1 (tbd.)	kV

*) Stray inductivity of GND and VS connections will lead to ringing of the supply voltage when driving an inductive load. This ringing results from the fast switching slopes of the driver outputs in combination with reverse recovery of the body diodes of the output driver MOSFETs. Even small trace inductivities as well as stray inductivity of sense resistors can easily generate a few volts of ringing leading to temporary voltage overshoot. This should be considered when working near the maximum voltage.

19 Electrical Characteristics

19.1 Operational Range

Parameter	Symbol	Min	Max	Unit
Junction temperature	T_J	-40	125	°C
Supply voltage (using internal +5V regulator)	V_{VS}	5.5	26	V
Supply voltage (internal +5V regulator bridged: V_{VCC}=V_{VSA})	V_{VS}	4.7	5.4	V
I/O supply voltage	V_{VIO}	3.00	5.25	V
VCC voltage when using optional external source (supplies digital logic and charge pump)	V_{VCC}	4.6	5.25	V
RMS motor coil current per coil (value for design guideline)	I_{RMS}		0.8	A
Peak output current per motor coil output (sine wave peak)	I_{Ox}		1.1	A
Peak output current per motor coil output (sine wave peak) Limit $T_J \leq 105$°C, e.g. for 100ms short time acceleration phase below 50% duty cycle.	I_{Ox}		1.5	A

19.2 DC Characteristics and Timing Characteristics

DC characteristics contain the spread of values guaranteed within the specified supply voltage range unless otherwise specified. Typical values represent the average value of all parts measured at +25°C. Temperature variation also causes stray to some values. A device with typical values will not leave Min/Max range within the full temperature range.

Power supply current	**DC-Characteristics** V_{VS} = 24.0V					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Total supply current, driver disabled I_{VS} + I_{VSA} + I_{VCC}	I_S	f_{CLK}=16MHz		25	40	mA
Total supply current, operating, I_{VS} + I_{VSA} + I_{VCC}	I_S	f_{CLK}=16MHz, 40kHz chopper, no load		28		mA
Static supply current	I_{VS0}	f_{CLK}=0Hz	3	4.5	7	mA
Supply current, driver disabled, dependency on CLK frequency	I_{VSX}	f_{CLK} variable, additional to I_{VS0}		1.3		mA/MHz
Internal current consumption from 5V supply on VCC pin	I_{VCC}	f_{CLK}=16MHz, 40kHz chopper		25	40	mA
IO supply current	I_{VIO}	no load on outputs, inputs at V_{IO} or GND		15	30	µA

Motor driver section	**DC- and Timing-Characteristics** V_{VS} = 24.0V					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
RDS_{ON} lowside MOSFET	R_{ONL}	measure at 100mA, 25°C, static state		0.4	0.5	Ω
RDS_{ON} highside MOSFET	R_{ONH}	measure at 100mA, 25°C, static state		0.5	0.6	Ω
slope, MOSFET turning on	t_{SLPON}	measured at 700mA load current	50	120	250	ns
slope, MOSFET turning off	t_{SLPOFF}	measured at 700mA load current	50	220	450	ns
Current sourcing, driver off	I_{OIDLE}	O_{XX} pulled to GND	120	180	250	µA

Charge pump	**DC-Characteristics**					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Charge pump output voltage	V_{VCP}-V_{VS}	operating, typical f_{chop}<40kHz	4.0	V_{5VOUT} - 0.4	V_{5VOUT}	V
Charge pump voltage threshold for undervoltage detection	V_{VCP}-V_{VS}	using internal 5V regulator voltage	3.1	3.6	3.9	V
Charge pump frequency	f_{CP}			1/16 f_{CLKOSC}		

Linear regulator	**DC-Characteristics**					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Output voltage	V_{5VOUT}	I_{5VOUT} = 0mA T_J = 25°C	4.75	5.0	5.25	V
Output resistance	R_{5VOUT}	Static load		3		Ω
Deviation of output voltage over the full temperature range	$V_{5VOUT(DEV)}$	I_{5VOUT} = 30mA T_J = full range		30	100	mV

Clock oscillator and input	**Timing-Characteristics**					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Clock oscillator frequency	f_{CLKOSC}	t_J=-50°C	9	12.4		MHz
Clock oscillator frequency	f_{CLKOSC}	t_J=50°C	10.1	13.2	17.2	MHz
Clock oscillator frequency	f_{CLKOSC}	t_J=150°C		13.4	18	MHz
External clock frequency (operating)	f_{CLK}		4	10-16	18	MHz
External clock high / low level time	t_{CLKL} / t_{CLKH}	CLK driven to 0.1 V_{VIO} / 0.9 V_{VIO}	10			ns
External clock first cycle triggering switching to external clock source	t_{CLK1}	CLK driven high	30	25		ns

Detector levels	**DC-Characteristics**					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
V_{VSA} undervoltage threshold for RESET	V_{UV_VSA}	V_{VS} rising	3.8	4.2	4.6	V
V_{5VOUT} undervoltage threshold for RESET	V_{UV_5VOUT}	V_{5VOUT} rising		3.5		V
V_{VCC_IO} undervoltage threshold for RESET	V_{UV_VIO}	V_{VCC_IO} rising	1.9	2.55	3.0	V
V_{VCC_IO} undervoltage detector hysteresis	$V_{UV\_VIOHYST}$		0.1	0.3	0.5	V
Short to GND detector threshold (V_{VSP} - V_{Ox})	V_{OS2G}		1.5	2.2	3	V
Short to GND detector delay (high side switch on to short detected)	t_{S2G}	High side output clamped to V_{SP}-3V	0.8	1.3	2	µs
Overtemperature prewarning	t_{OTPW}	Temperature rising	100	120	140	°C
Overtemperature shutdown	t_{OT}	Temperature rising	135	150	170	°C

Sense resistor voltage levels	**DC-Characteristics**					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Sense input peak threshold voltage (low sensitivity)	V_{SRTL}	*vsense*=0 *csactual*=31 *sin_x*=248 Hyst.=0; I_{BRxy}=0		320		mV
sense input peak threshold voltage (high sensitivity)	V_{SRTH}	*vsense*=1 *csactual*=31 *sin_x*=248 Hyst.=0; I_{BRxy}=0		180		mV
Sense input tolerance / motor current full-scale tolerance	I_{COIL}	*vsense*=0	-5		+5	%
Internal resistance from pin BRxy to internal sense comparator (additional to sense resistor)	R_{BRxy}			20		mΩ

Digital pins	**DC-Characteristics**					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Input voltage low level	V_{INLO}		-0.3		0.3 V_{VIO}	V
Input voltage high level	V_{INHI}		0.7 V_{VIO}		V_{VIO}+0.3	V
Input Schmitt trigger hysteresis	V_{INHYST}			0.12 V_{VIO}		V
Output voltage low level	V_{OUTLO}	I_{OUTLO} = 2mA			0.2	V
Output voltage high level	V_{OUTHI}	I_{OUTHI} = -2mA	V_{VIO}-0.2			V
Input leakage current	I_{ILEAK}		-10		10	µA
Digital pin capacitance	C			3.5		pF

19.3 Thermal Characteristics

The following table shall give an idea on the thermal resistance of the QFN-48 package. The thermal resistance for a four-layer board will provide a good idea on a typical application. The single layer board example is kind of a worst-case condition, as the typical application will require a 4-layer board. Actual thermal characteristics will depend on the PCB layout, PCB type and PCB size.

A thermal resistance of 23°C/W for a typical board means, that the package is capable of continuously dissipating 4W at an ambient temperature of 25°C with the die temperature staying below 125°C.

Parameter	Symbol	Conditions	Typ	Unit
Typical power dissipation One motor active, one motor in standby at low current	P_D	One motor 1.00A RMS 115°C (125°C) One motor 0.71A RMS 85°C (93°C) Surface temperature at package center (peak surface temperature), board 55mm x 85mm, 25°C environment StealthChop or SpreadCycle, sinewave, 40 or 20kHz chopper, 24V, 16MHz, internal supply for VCC Motors: QSH4218-035-10-027	3.7 2.4	W W
Typical power dissipation Two motors active	P_D	Two motors 0.71A RMS 113°C (119°C) Two motors 0.35A RMS 64°C (68°C)	3.7 1.4	W W
Thermal resistance junction to ambient on a single layer board	R_{TJA}	Single signal layer board (1s) as defined in JEDEC EIA JESD51-3 (FR4, 76.2mm x 114.3mm, d=1.6mm)	80	K/W
Thermal resistance junction to ambient on a multilayer board	R_{TMJA}	Dual signal and two internal power plane board (2s2p) as defined in JEDEC EIA JESD51-5 and JESD51-7 (FR4, 76.2mm x 114.3mm, d=1.6mm)	23	K/W
Thermal resistance junction to ambient on a multilayer board with air flow	R_{TMJA1}	Identical to R_{TMJA}, but with air flow 1m/s	20	K/W
Thermal resistance junction to board	R_{TJB}	PCB temperature measured within 1mm distance to the package	10	K/W
Thermal resistance junction to case	R_{TJC}	Junction temperature to heat slug of package	3	K/W

The thermal resistance in an actual layout can be tested by checking for the heat up caused by the standby power consumption of the chip. When no motor is attached, all power seen on the power supply is dissipated within the chip.

Note
A spreadsheet for calculating TMC5041 power dissipation is available on www.trinamic.com.

20 Layout Considerations

20.1 Exposed Die Pad

The TMC5041 uses its die attach pad to dissipate heat from the drivers and the linear regulator to the board. For best electrical and thermal performance, use a reasonable amount of solid, thermally conducting vias between the die attach pad and the ground plane. The printed circuit board should have a solid ground plane spreading heat into the board and providing for a stable GND reference.

20.2 Wiring GND

All signals of the TMC5041 are referenced to their respective GND. Directly connect all GND pins under the TMC5041 to a common ground area (GND, GNDP, GNDA and die attach pad). The GND plane right below the die attach pad should be treated as a virtual star point. For thermal reasons, the PCB top layer shall be connected to a large PCB GND plane spreading heat within the PCB.

Attention
Especially, the sense resistors are susceptible to GND differences and GND ripple voltage, as the microstep current steps make up for voltages down to 0.5 mV. No current other than the sense resistor current should flow on their connections to GND and to the TMC5041. Optimally place them close to the TMC5041, with one or more vias to the GND plane for each sense resistor. The two sense resistors for one coil should not share a common ground connection trace or vias, as also PCB traces have a certain resistance.

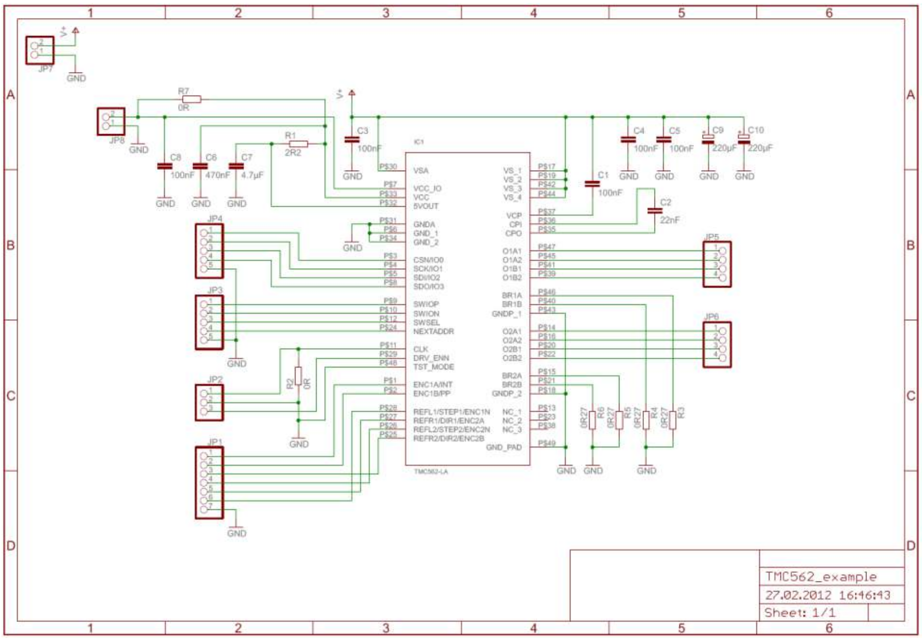
20.3 Supply Filtering

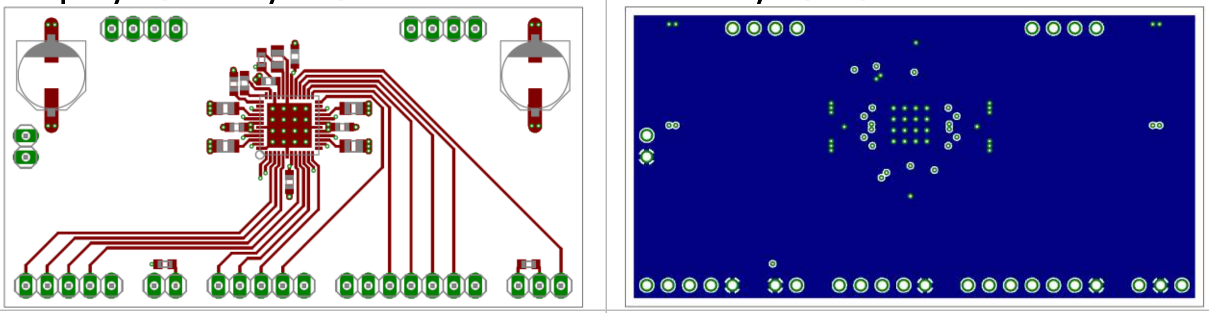
The 5VOUT output voltage ceramic filtering capacitor (4.7 µF recommended) should be placed as close as possible to the 5VOUT pin, with its GND return going directly to the GNDA pin. Use as short and as thick connections as possible. For best microstepping performance and lowest chopper noise an additional filtering capacitor can be used for the VCC pin to GND, to avoid charge pump and digital part ripple influencing motor current regulation. Therefore, place a ceramic filtering capacitor (470nF recommended) as close as possible (1-4mm distance) to the VCC pin with GND return going to the ground plane. VCC can be coupled to 5VOUT using a 2.2 Ω resistor to supply the digital logic from 5VOUT while keeping ripple away from this pin.

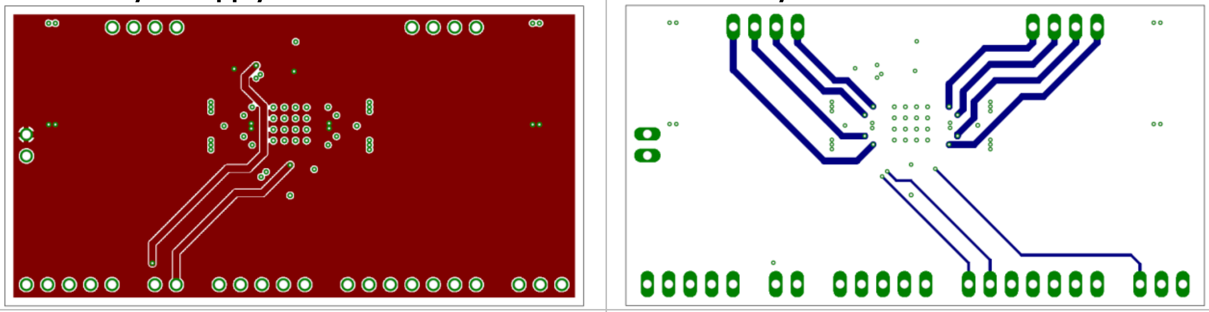
A 100 nF filtering capacitor should be placed as close as possible to the VSA pin to ground plane. The motor supply pins VS should be decoupled with an electrolytic capacitor (47 µF or larger is recommended) and a ceramic capacitor, placed close to the device.

Consider that the switching motor coil outputs have a high dV/dt. Thus, capacitive stray into high resistive signals can occur if the motor traces are near other traces over longer distances.

20.4 Layout Example


Schematic


1- Top Layer (assembly side) **2- Inner Layer (GND)**


3- Inner Layer (supply VS) **4- Bottom Layer**

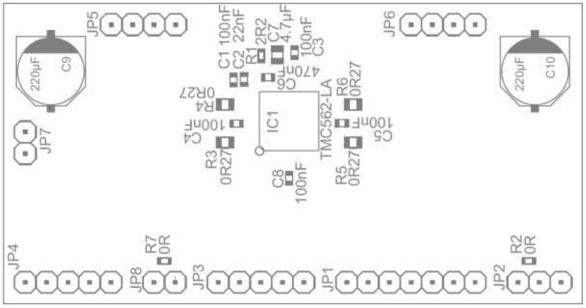

Components

Figure 20.1 Layout example

21 Package Mechanical Data

21.1 Dimensional Drawings

Attention: Drawings not to scale.

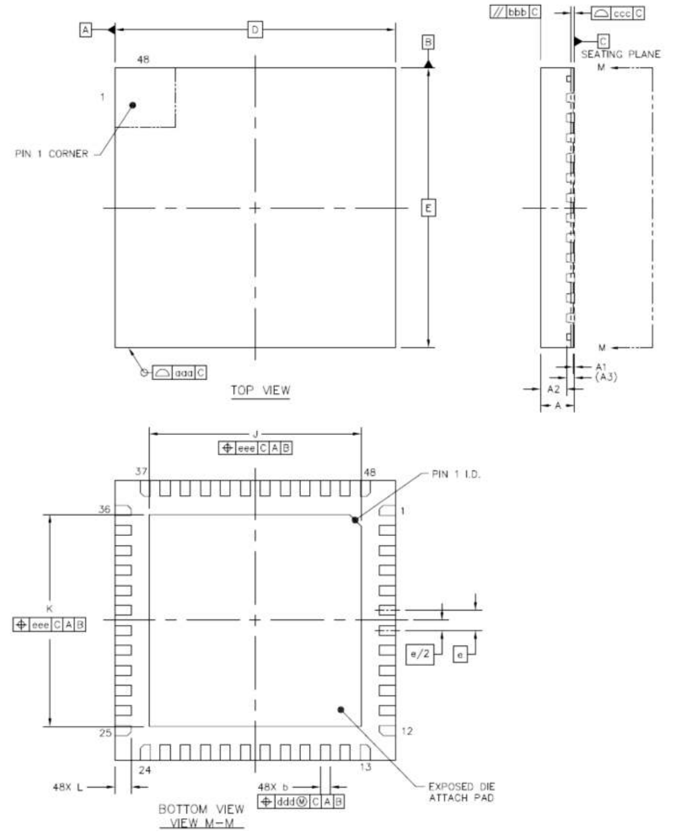


Figure 21.1 Dimensional drawings

Parameter [mm]	Ref	Min	Nom	Max
total thickness	A	0.80	0.85	0.90
stand off	A1	0.00	0.035	0.05
mold thickness	A2	-	0.65	0.67
lead frame thickness	A3		0.203	
lead width	b	0.2	0.25	0.3
body size X	D		7.0	
body size Y	E		7.0	
lead pitch	e		0.5	
exposed die pad size X	J	5.2	5.3	5.4
exposed die pad size Y	K	5.2	5.3	5.4
lead length	L	0.35	0.4	0.45
package edge tolerance	aaa			0.1
mold flatness	bbb			0.1
coplanarity	ccc			0.08
lead offset	ddd			0.1
exposed pad offset	eee			0.1

21.2 Package Codes

Type	Package	Temperature range	Code & marking	MSL level
TMC5041-LA	QFN48 (RoHS)	-40°C … +125°C	TMC5041-LA	MSL 3 / 160h
TMC5041-LA-T	Tape on reel packed products			

22 Design Philosophy

We feel that this is one of the coolest chips which we did within the last years. The TMC50XX and TMC5130 family brings premium functionality, reliability and coherence previously reserved to costly motion control units to smart applications. Integration at street level cost was possible by squeezing know-how into a few mm² of layout using one of the most modern smart power processes. The IC comprises all the knowledge gained from designing motion controller and driver chips and complex motion control systems for more than 20 years. We are often asked if our motion controllers contain software – they definitely do not. The reason is that sharing resources in software leads to complex timing constraints and can create interrelations between parts which should not be related. This makes debugging of software so difficult. Therefore, the IC is completely designed as a hardware solution, i.e., each internal calculation uses a specially designed dedicated arithmetic unit. The basic philosophy is to integrate all real-time critical functionality in hardware, and to leave additional starting points for highest flexibility. Parts of the design go back to previous ICs, starting from the TMC453 motion controller developed in 1997. Our deep involvement, practical testing and the stable team ensure a high level of confidence and functional safety.

Bernhard Dwersteg, former Trinamic CTO and founder

23 Disclaimer

TRINAMIC Motion Control GmbH & Co. KG does not authorize or warrant any of its products for use in life support systems, without the specific written consent of TRINAMIC Motion Control GmbH & Co. KG. Life support systems are equipment intended to support or sustain life, and whose failure to perform, when properly used in accordance with instructions provided, can be reasonably expected to result in personal injury or death.

Information given in this data sheet is believed to be accurate and reliable. However, no responsibility is assumed for the consequences of its use nor for any infringement of patents or other rights of third parties which may result from its use.

Specifications are subject to change without notice.

All trademarks used are property of their respective owners.

24 ESD Sensitive Device

The TMC5041 is an ESD sensitive CMOS device sensitive to electrostatic discharge. Take special care to use adequate grounding of personnel and machines in manual handling. After soldering the devices to the board, ESD requirements are more relaxed. Failure to do so can result in defect or decreased reliability.

25 Designed for Sustainability

Sustainable growth is one of the most important and urgent challenges today. We at Trinamic try to contribute by designing highly efficient IC products, to minimize energy consumption, ensure best customer experience and long-term satisfaction by smooth and silent run, while minimizing the demand for external resources, e.g., for power supply, cooling infrastructure, reduced motor size and magnet material by intelligent control interfaces and advanced algorithms.

Please help and design efficient and durable products made for a sustainable world.

26 Table of Figures

27 Revision History

Version	**Date**	**Author** **BD – Bernhard Dwersteg** **SD – Sonja Dwersteg**	**Description**
1.04	2013-NOV-06	SD	First version of preliminary TMC5041 datasheet based on TMC5031 datasheet V1.07
1.05	2015-JAN-20	BD	Full TMC5041 datasheet based on preliminary datasheet and TMC5072 V1.17 datasheet (see revision history in TMC5072 datasheet for full history)
1.06	2015-JAN-23	BD	Added StealthChop initialization example
1.09	2015-MAR-10	BD	Removed Encoder References
1.10	2015-MAR-25	BD	Removed preliminary, slight corrections in wording
1.11	2015-OCT-13	BD	Correct SPI write access example, SPI mode 3, added TCLK1 data, corrected TOFF calculation example, comments in GSTAT, comment on SPI_STATUS, 5V only +-5%, X1=128 in microstep table defaults
1.12	2016-APR-22	BD	More details on: Setting negative encoder factors, StealthChop lower current limit, Ramp generator Joystick control, Terminate Ramp, Adaptation to internal fCLK, Interrupt handling Corrected: effective StealthChop PWM frequency is 2*divider setting, Wording V1 and VMAX register, ESD schematic w. varistors instead of snubber
1.13	2017-MAY-16	BD	Minor corrections
1.14	2020-JUN-12	BD	Updated front page, minor corrections, flowchart update for StealthChop
1.15	2022-FEB-03	BD	Updated logo & order codes; minor re-wording; Added chapter on ramp generator response time; Improved several text segments

Table 27.1 Documentation revisions

28 References

[AN001] Trinamic Application Note 001 - Parameterization of SpreadCycle™, www.trinamic.com
[AN002] Trinamic Application Note 002 - Parameterization of StallGuard2™ & CoolStep™, www.trinamic.com
Calculation sheet TMC50XX_Calculations.xlsx