TMC2130-LA DATASHEET

Universal high voltage driver for two-phase bipolar stepper motor. stealthChop™ for quiet movement. Integrated MOSFETs for up to 1.7A motor current per coil. With Step/Dir Interface and SPI.

dcStep™

coolStep™

stallGuard2™

APPLICATIONS

Textile, Sewing Machines
Factory Automation
Lab Automation
Liquid Handling
Medical
Office Automation
CCTV, Security
ATM, Cash recycler
POS
Pumps and Valves

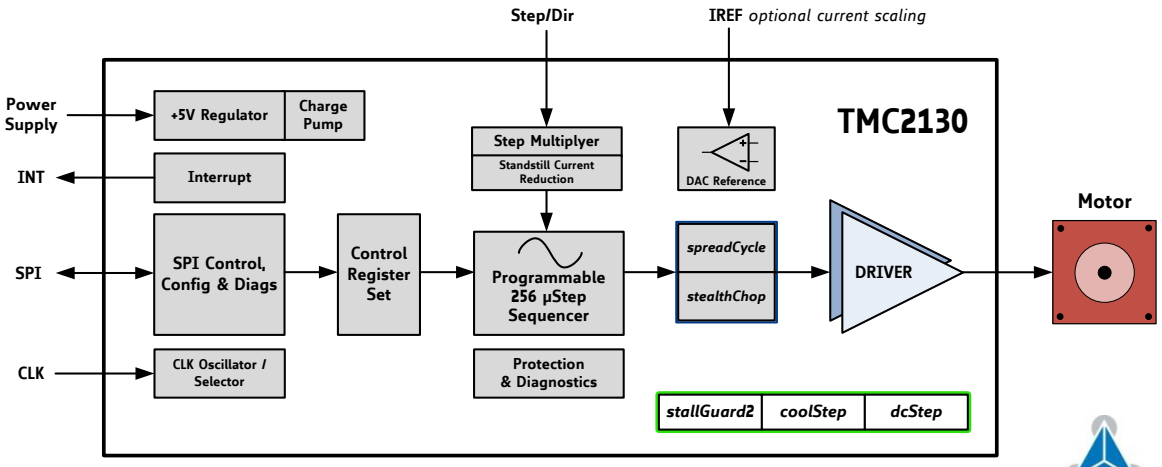
FEATURES AND BENEFITS

2-phase stepper motors
Drive Capability up to 1.7A coil current (2.5A peak)
Step/Dir Interface with microstep interpolation **microPlyer™**
SPI Interface
Voltage Range 4.75... 46V DC
Highest Resolution 256 microsteps per full step
stealthChop™ for extremely quiet operation and smooth motion
spreadCycle™ highly dynamic motor control chopper
dcStep™ load dependent speed control
stallGuard2™ high precision sensorless motor load detection
coolStep™ current control for energy savings up to 75%
Integrated Current Sense Option
Passive Breaking and freewheeling mode
Full Protection & Diagnostics
Small Size 5x6mm² QFN36 package

DESCRIPTION

The TMC2130 is a high performance driver IC for two phase stepper motors. Standard SPI and Step/Dir simplify communication. TRINAMICs sophisticated stealthChop chopper ensures noiseless operation combined with maximum efficiency and best motor torque. Due to features like stallGuard2 and coolStep energy consumption can be reduced by up to 75%. Moreover, using the dcStep feature of the TMC2130 high loads can be moved as fast as possible without steploss. Integrated power MOSFETs handle motor currents up to 1.2A RMS continuously or 2.5A short time peak current per coil. Protection and diagnostic features support robust and reliable operation. Industries' most advanced stepper motor driver enables miniaturized designs with low external component count for cost-effective and highly competitive solutions.

BLOCK DIAGRAM



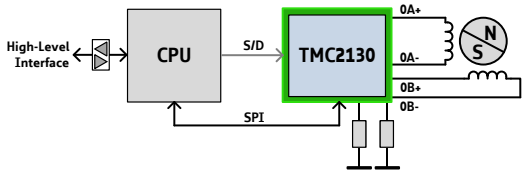
TRINAMIC Motion Control GmbH & Co. KG
Hamburg, Germany

TRINAMIC
MOTION CONTROL

APPLICATION EXAMPLES: HIGH VOLTAGE – MULTIPURPOSE USE

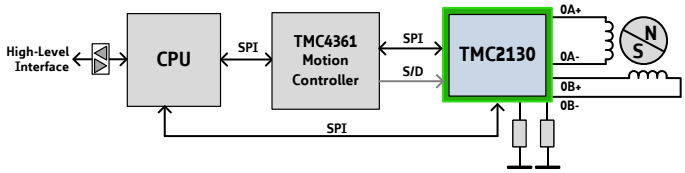
The TMC2130 scores with power density, integrated power MOSFETs, and a versatility that covers a wide spectrum of applications from battery systems up to embedded applications with 1.7A motor current per coil. Based on stallGuard2, coolStep, dcStep, spreadCycle, and stealthChop, the TMC2130 optimizes drive performance and keeps costs down. It considers velocity vs. motor load, realizes energy savings, smoothness of the drive and noiselessness. Extensive support at the chip, board, and software levels enables rapid design cycles and fast time-to-market with competitive products.

MINIATURIZED DESIGN FOR ONE STEPPER MOTOR



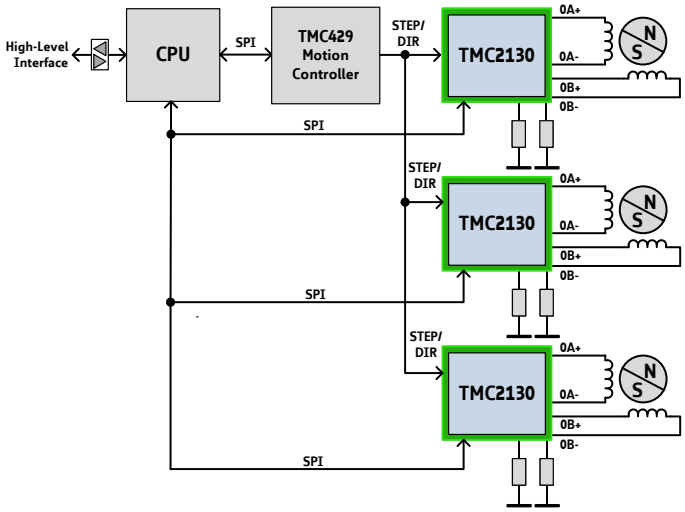
In this application, the CPU initializes the TMC2130 motor driver via SPI interface and controls motor movement by sending step and direction signals. A real time software realizes motion control.

DESIGN FOR DEMANDING APPLICATIONS WITH S-SHAPED RAMP PROFILES



The CPU initializes the TMC4361 motion controller and the TMC2130. Thereafter, it sends target positions to the TMC4361. Now, the TMC4361 takes control over the TMC2130. Combining the TMC4361 and the TMC2130 offers diverse possibilities for demanding applications including servo drive features.

COMPACT DESIGN FOR UP TO THREE STEPPER MOTORS



Here, an application with up to three stepper motors is shown. A single CPU combined with a TMC429 motion controller manages the whole stepper motor driver system. This design is highly economical and space saving if more than one stepper motor is needed.

ORDER CODES

Order code	Description	Size [mm²]
TMC2130-LA	1-axis dcStep, coolStep, and stealthChop driver; QFN32	5 x 6
TMC2130-EVAL	Evaluation board for TMC2130 two phase stepper motor controller/driver	85 x 55
TMC4361-EVAL	Motion controller board (part of evaluation board system)	85 x 55
STARTRAMPE	Baseboard for TMC2130-EVAL and further evaluation boards	85 x 55
ESELSBRÜCKE	Connector board for plug-in evaluation board system	61 x 38

Table of Contents

1 Principles of Operation

THE TMC2130 OFFERS THREE BASIC MODES OF OPERATION:

In *Step/Direction Driver Mode*, the TMC2130 is the microstep sequencer and power driver between a motion controller and a two phase stepper motor. Configuration of the TMC2130 is done via SPI. A dedicated motion controller IC or the CPU sends step and direction signals to the TMC2130. The TMC2130 provides the related motor coil currents to operate the motor. In *Standalone Mode*, the TMC2130 can be configured using pins. In this mode of operation CPU interaction is not necessary.
The third mode of operation is the *SPI Driver Mode*, which is used in combination with TRINAMICs TMC4361 motion controller chip. This mode of operation offers several possibilities for sophisticated applications.

OPERATION MODE 1: Step/Direction Driver Mode

An external motion controller is used or a central CPU generates step and direction signals. The motion controller (e.g. TMC429) controls the motor position by sending pulses on the STEP signal while indicating the direction on the DIR signal. The TMC2130 provides a microstep counter and a sine table to convert these signals into the coil currents which control the position of the motor. The TMC2130 automatically takes care of intelligent current and mode control and delivers feedback on the state of the motor. The microPlyer automatically smoothens motion. To optimize power consumption and heat dissipation, software may also adjust coolStep and stallGuard2 parameters in real-time, for example to implement different tradeoffs between speed and power consumption.

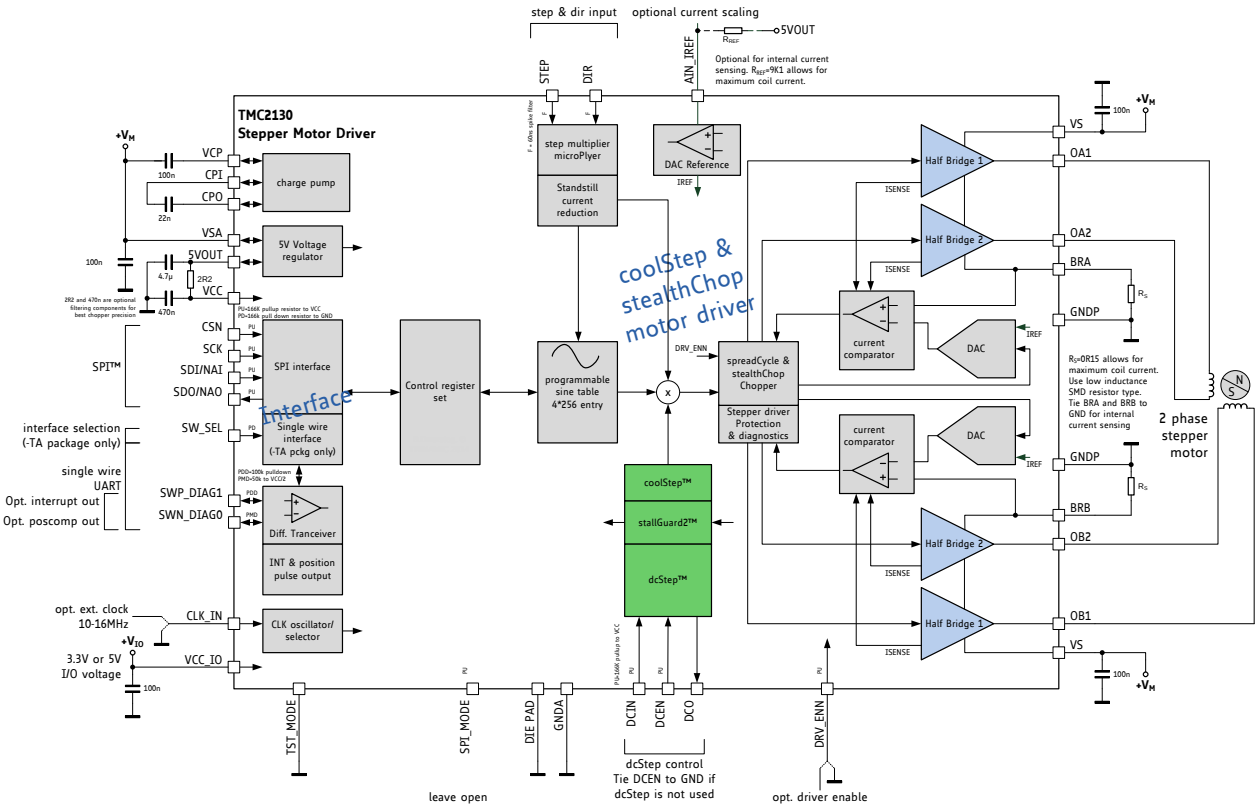


Figure 1.1 TMC2130 STEP/DIR application diagram

OPERATION MODE 2: Standalone Mode

The TMC2130 positions the motor based on step and direction signals. The microPlyer automatically smoothens motion. No CPU interaction is required. Configuration is done by hardware pins. Basic standby current control can be done by the TMC2130. Optional feedback signals allow error detection and synchronization.

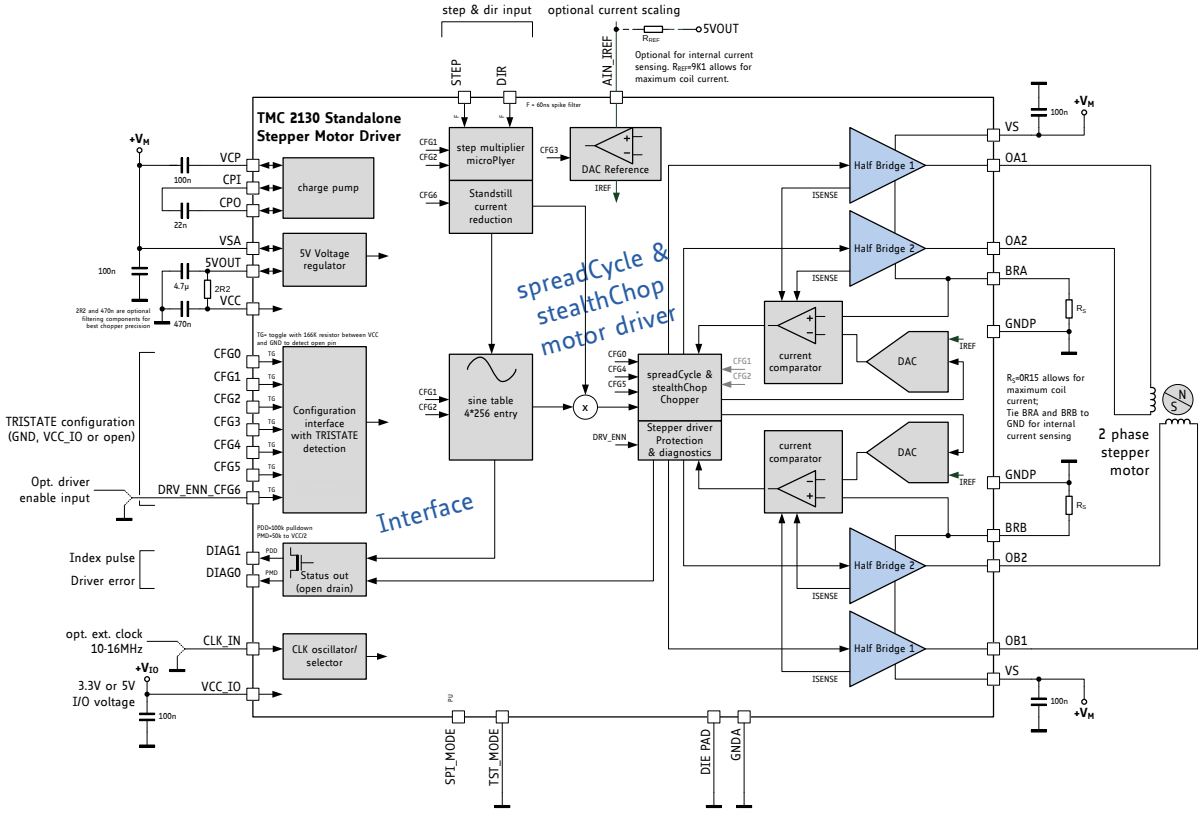


Figure 1.2 TMC2130 standalone driver application diagram

OPERATION MODE 3: SPI Driver Mode

Together with the TMC4361 high-performance S-ramp motion controller the whole communication with the TMC2130 stepper motor driver can be done via SPI. Combining these two ICs offers several possibilities for demanding applications including servo features. Please refer to Figure 1.1 for more information about the pinning, which is identical to step/direction driver mode, except that the STEP & DIR pins are not required for operation.

1.1 Key Concepts

The TMC2130 implements 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.

spreadCycle™ High-precision chopper algorithm available as an alternative to the traditional constant off-time algorithm.

dcStep™ Load dependent speed control. The motor moves as fast as possible and never loses a step.

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%.

microPlyer™ Microstep interpolator for obtaining increased smoothness of microstepping when using the STEP/DIR interface.

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 TMC2130 slave always consists of sending one 40-bit command word and receiving one 40-bit status word.

The SPI command rate typically is a single initialization after power-on.

1.3 Software

From a software point of view the TMC2130 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 the Motor

1.4.1 STEP/DIR Interface

The motor can be controlled by a step and direction input. Active edges on the STEP input can be rising edges or both rising and falling edges as controlled by a mode bit (DEDGE). Using both edges cuts the toggle rate of the STEP signal in half, which is useful for communication over slow interfaces such as optically isolated interfaces. On each active edge, the state sampled from the DIR input determines whether to step forward or back. Each step can be a fullstep or a microstep, in which there are 2, 4, 8, 16, 32, 64, 128, or 256 microsteps per fullstep. During microstepping, a step impulse with a low state on DIR increases the microstep counter and a high decreases the counter by an amount controlled by the microstep resolution. An internal table translates the counter value into the sine and cosine values which control the motor current for microstepping.

1.4.2 SPI Direct Mode

The direct mode allows control of both motor coil currents and polarity via SPI. It mainly is intended for use with a dedicated external motion controller IC with integrated sequencer. The sequencer applies sine and cosine waves to the motor coils. This mode also allows control of DC motors, etc.

1.5 stealthChop and 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 absolutely 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 1.3 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.

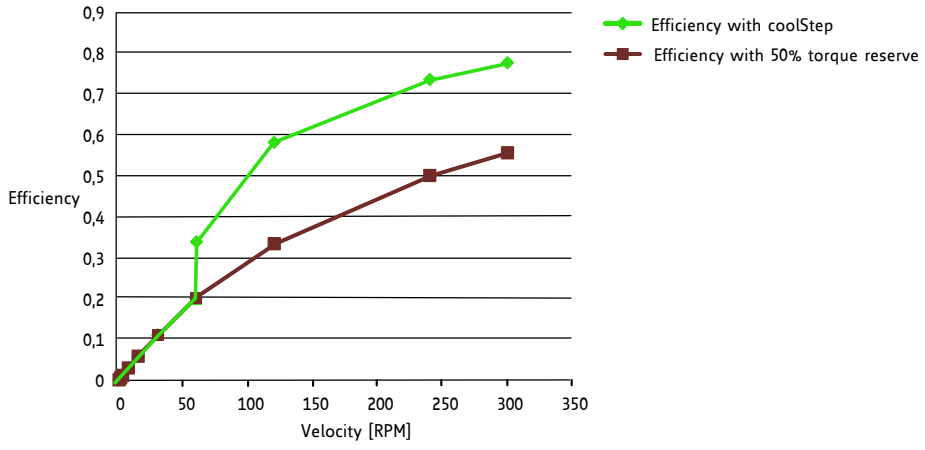


Figure 1.3 Energy efficiency with coolStep (example)

1.8 dcStep – Load Dependent Speed Control

dcStep allows the motor to run near its load limit and at its velocity limit without losing a step. If the mechanical load on the motor increases to the stalling load, the motor automatically decreases velocity so that it can still drive the load. With this feature, the motor will never stall. In addition to the increased torque at a lower velocity, dynamic inertia will allow the motor to overcome mechanical overloads by decelerating. dcStep feeds back status information to the external motion controller or to the system CPU, so that the target position will be reached, even if the motor velocity needs to be decreased due to increased mechanical load. A dynamic range of up to factor 10 or more can be covered by dcStep without any step loss. By optimizing the motion velocity in high load situations, this feature further enhances overall system efficiency.

Benefits are:

- Motor does not loose steps in overload conditions
- Application works as fast as possible
- Highest possible acceleration automatically
- Highest energy efficiency at speed limit
- Highest possible motor torque using fullstep drive
- Cheaper motor does the job

2 Pin Assignments

2.1 Package Outline

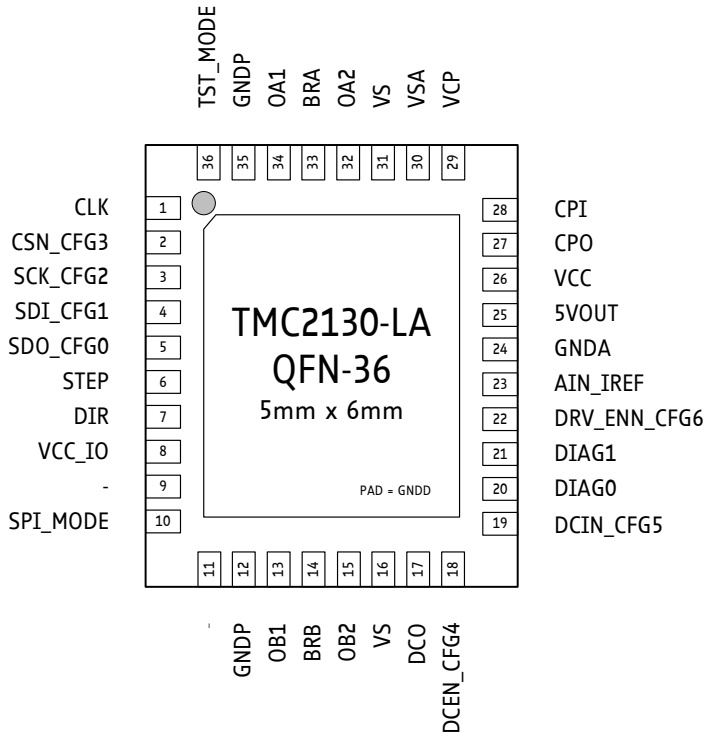


Figure 2.1 TMC2130-LA package and pinning QFN36 (5x6mm body)

2.2 Signal Descriptions

Pin	Number	Type	Function
CLK	1	DI	CLK input. Tie to GND using short wire for internal clock or supply external clock.
CSN_CFG3	2	DI	SPI chip select input (negative active) or configuration input
SCK_CFG2	3	DI	SPI serial clock input and configuration input
SDI_CFG1	4	DI	SPI data input and configuration input
SDO_CFG0	5	DIO	SPI data output (tristate) or configuration input
STEP	6	DI	STEP input
DIR	7	DI	DIR input
VCC_IO	8		3.3V to 5V IO supply voltage for all digital pins.
DNC	9	-	Do not connect. Leave open!
SPI_MODE	10	DI (pu)	Mode selection input with pullup resistor. When tied low, the chip is in standalone mode and pins have their CFG functions. When tied high, the SPI and UART interface are available for control. Integrated pull-up resistor.
N.C.	11		Unused pin, connect to GND for compatibility to future versions.
GNDP	12, 35		Power GND. Connect to GND plane near pin.
OB1	13		Motor coil B output 1
BRB	14		Sense resistor connection for coil B. Place sense resistor to GND near pin. An additional 100nF capacitor to GND (GND plane) is recommended for best performance.
OB2	15		Motor coil B output 2

Pin	Number	Type	Function
VS	16, 31		Motor supply voltage. Provide filtering capacity near pin with short loop to nearest GNDP pin (respectively via GND plane).
DCO	17	DIO	dcStep ready output
DCEN_CFG4	18	DI	dcStep enable input or configuration input. In SPI-mode, tie to GND for normal operation (no dcStep).
DCIN_CFG5	19	DI	dcStep gating input for axis synchronization or configuration input
DIAG0	20	DIO	Diagnostics output DIAG0. Use external pull-up resistor with 47k or less in open drain mode.
DIAG1	21	DIO	Diagnostics output DIAG1. Use external pull-up resistor with 47k or less in open drain mode.
DRV_ENN_ CFG6	22	DI	Enable input or configuration / Enable input. The power stage becomes switched off (all motor outputs floating) when this pin becomes driven to a high level.
AIN_IREF	23	AI	Analog reference voltage for current scaling (optional mode) or reference current for use of internal sense resistors
GNDA	24		Analog GND. Tie to GND plane.
5VOUT	25		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.
VCC	26		5V supply input for digital circuitry within chip and charge pump. Attach 470nF capacitor to GND (GND plane). May be supplied by 5VOUT. A 2.2 or 3.3 Ohm 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!
CPO	27		Charge pump capacitor output.
CPI	28		Charge pump capacitor input. Tie to CPO using 22nF 50V capacitor.
VCP	29		Charge pump voltage. Tie to VS using 100nF capacitor.
VSA	30		Analog supply voltage for 5V regulator. Normally tied to VS. Provide a 100nF filtering capacitor.
OA2	32		Motor coil A output 2
BRA	33		Sense resistor connection for coil A. Place sense resistor to GND near pin. An additional 100nF capacitor to GND (GND plane) is recommended for best performance.
OA1	34		Motor coil A output 1
TST_MODE	36	DI	Test mode input. Tie to GND using short wire.
Exposed die pad	-		Connect the exposed die pad to a GND plane. Provide as many as possible vias for heat transfer to GND plane. Serves as GND pin for digital circuitry.

3 Sample Circuits

The sample circuits show the connection of 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

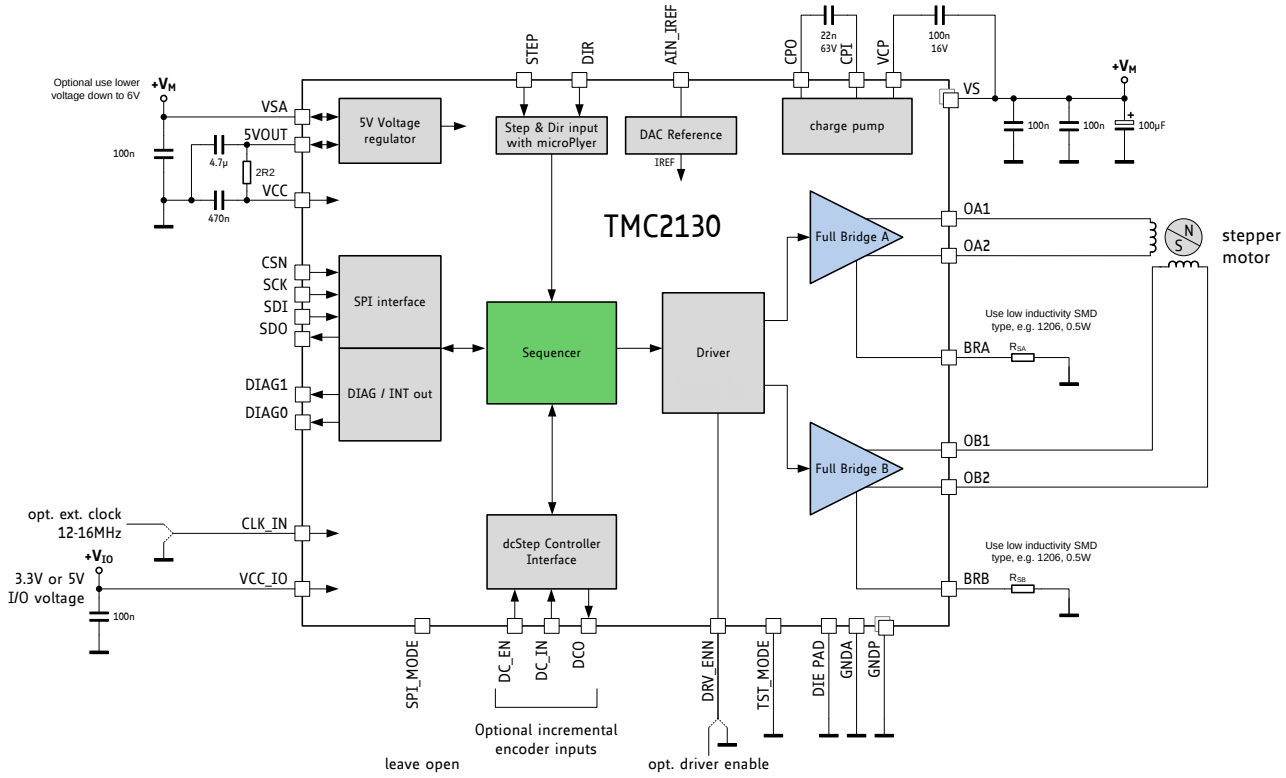


Figure 3.1 Standard application circuit

The standard application circuit uses a minimum set of additional components in order to operate the motor. Use low ESR capacitors for filtering the power supply which are capable to cope with the current ripple. The current ripple often depends on the power supply and cable length. The VCC_IO voltage can be supplied from 5VOUT, or from an external source, e.g. a low drop 3.3V regulator. In order 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 or 36V. Using the 12V supply for VSA will reduce the power dissipation of the internal 5V regulator to about 37% resp. 23% of the dissipation caused by supply with the full motor voltage.

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 upon power up or power down.

3.2 Reduced Number of Components

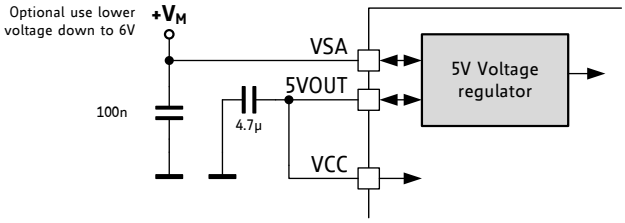


Figure 3.2 Reduced number of filtering components

The standard application circuit uses RC filtering to de-couple the output of the internal linear regulator from high frequency ripple caused by digital circuitry supplied by the VCC input. For cost sensitive applications, the RC-Filtering on VCC can be eliminated. This leads to more noise on 5VOUT caused by operation of the charge pump and the internal digital circuitry. There is a slight impact on microstep vibration and chopper noise performance.

3.3 Internal RDSon Sensing

For cost critical or space limited applications, it may be desired to eliminate the sense resistors. Further, this slightly reduces power dissipation, because the effective resistance of the driver bridge is reduced. In this application, a reference current set by a tiny external resistor programs the output current. For calculation of the reference resistor, refer chapter 10.

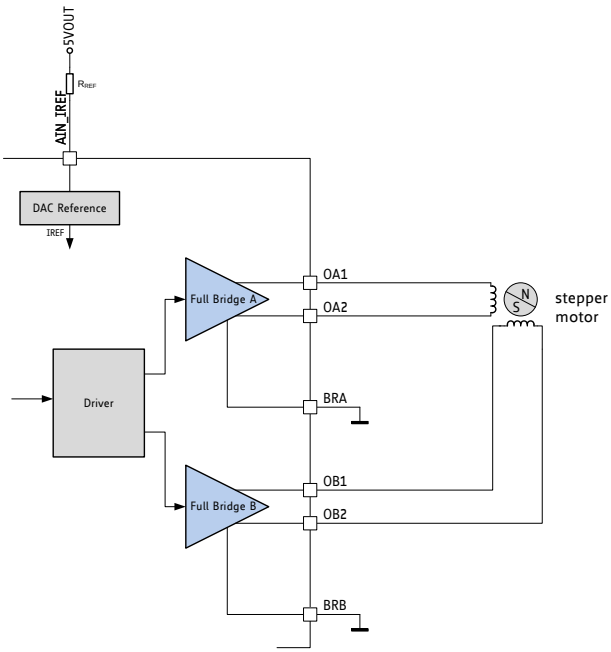


Figure 3.3 RDSon based sensing eliminates high current sense resistors

3.4 External 5V Power Supply

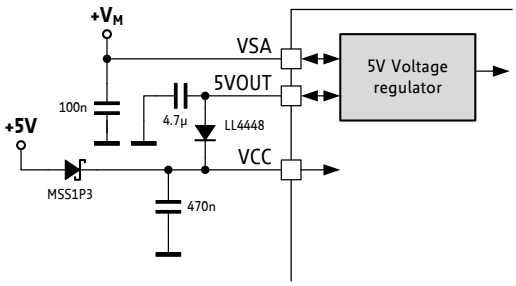
When an external 5V power supply is available, the power dissipation caused by the internal linear regulator can be eliminated. This especially is beneficial in high voltage applications, and when thermal conditions are critical. There are two options for using this external 5V source: either the external 5V source is used to support the digital supply of the driver by supplying the VCC pin or the complete internal voltage regulator becomes bridged and is replaced by the external supply voltage.

3.4.1 Support for the VCC Supply

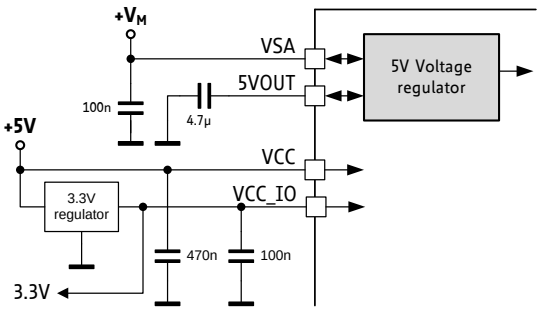
This scheme uses an external supply for all digital circuitry within the driver (Figure 3.4). As the digital circuitry makes up for most of the power dissipation, this way the internal 5V regulator sees only low remaining load. The precisely regulated voltage of the internal regulator is still used as the reference for the motor current regulation as well as for supplying internal analog circuitry.

When cutting pin VCC from 5VOUT, make sure that the VCC supply comes up before or synchronously with the 5VOUT supply, because otherwise the power-up reset event may be missed by the internal logic. A simple schematic uses two diodes forming an OR of the internal and the external power supplies to VCC to ensure this. In order to prevent the chip from drawing part of the power from its internal regulator, a low drop 1A Schottky diode is used for the external 5V supply path, while a silicon diode is used for the 5VOUT path. An enhanced solution uses a dual PNP transistor as an active switch. It minimizes voltage drop and thus gives best performance.

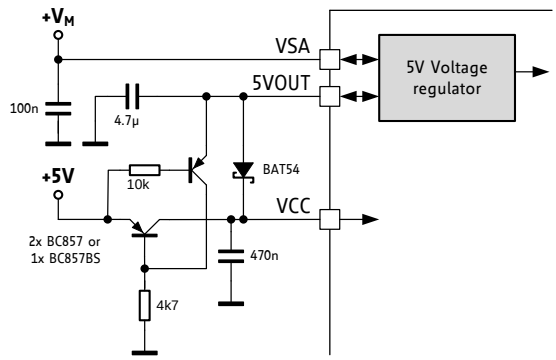
Using a 3.3V VCC_IO additional switching can be eliminated given that a safe reset condition is ensured by the 3.3V VCC_IO coming up synchronously with or delayed to VCC. This is true, when a linear regulator is used to generate a 3.3V VCC_IO from the external 5V VCC source. This 3.3V regulator will cause a certain voltage drop. A voltage drop in the regulator of 0.9V or more (e.g. LD1117-3.3) ensures that the 5V supply already has reached a lower limit of more than about 3.0V once the reset conditions ends. The reset condition ends earliest, when VCC_IO exceeds the undervoltage limit of minimum 2.1V. Make sure that the power-down sequence also is safe. Undefined states can result when VCC drops well below 4V without safely triggering a reset condition. Triggering a reset upon power-down can be ensured when VSA goes down synchronously with or before VCC.



VCC supplied from external 5V. 5V or 3.3V IO voltage.



VCC supplied from external 5V. 3.3V IO voltage generated from same source.



VCC supplied from external 5V using active switch. 5V or 3.3V IO voltage.

Figure 3.4 Using an external 5V supply for digital circuitry of driver (different options)

3.4.2 Internal Regulator Bridged

In case a clean external 5V supply is available, it can be used for complete supply of analog and digital part (Figure 3.5). 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.

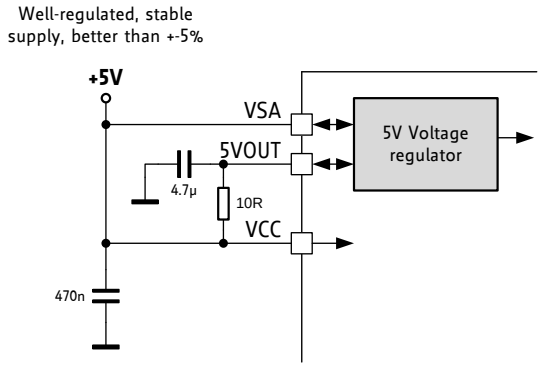


Figure 3.5 Using an external 5V supply to bypass internal regulator

3.5 Pre-Regulator for Reduced Power Dissipation

When operating at supply voltages up to 46V for VS and VSA, the internal linear regulator will contribute with up to 1W to the power dissipation of the driver. This will reduce the capability of the chip to continuously drive high motor current, especially at high environment temperatures. When no external power supply in the range 5V to 24V is available, an external pre-regulator can be built with a few inexpensive components in order to dissipate most of the voltage drop in external components. Figure 3.6 shows different examples. In case a well-defined supply voltage is available, a single 1W or higher power zener diode also does the job.

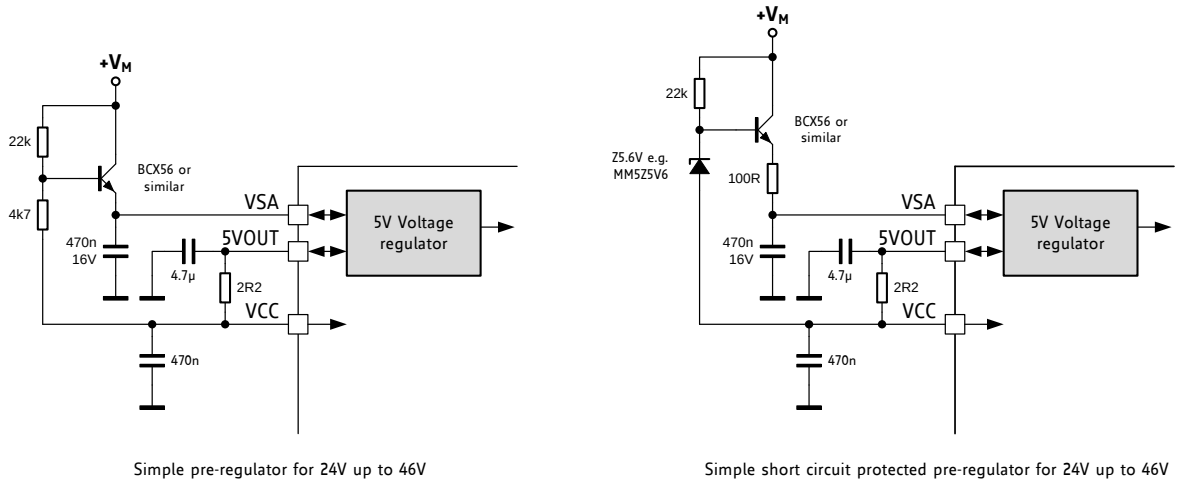


Figure 3.6 Examples for simple pre-regulators

3.6 5V Only Supply

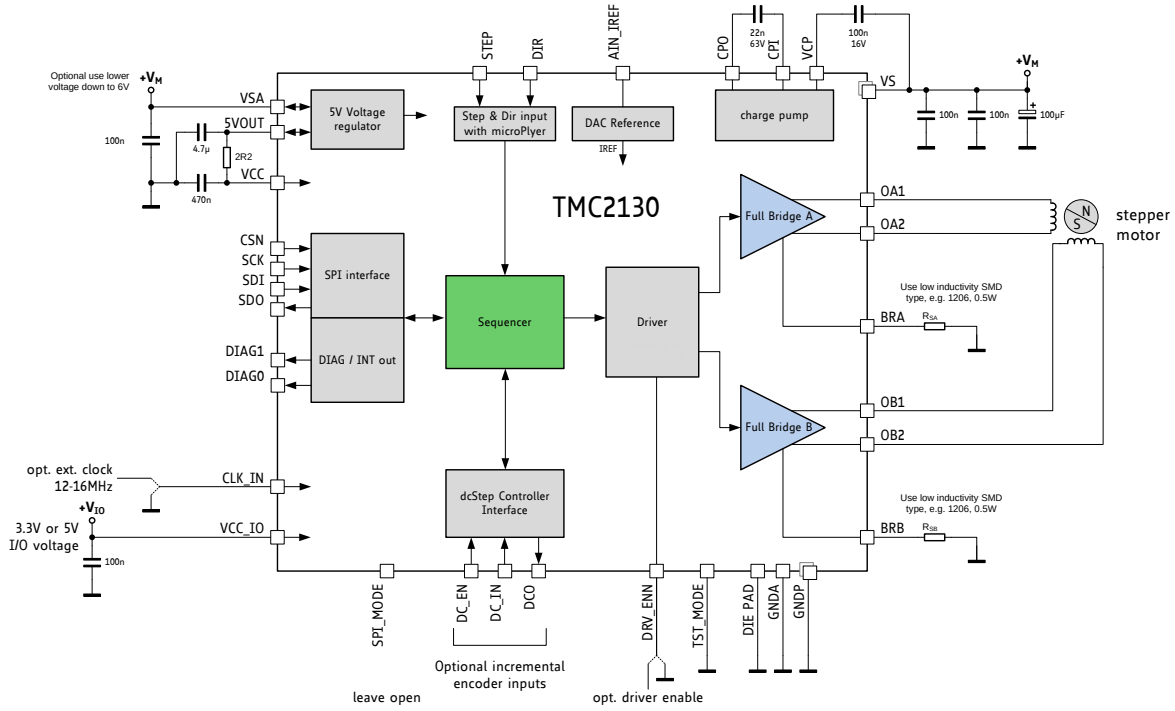


Figure 3.7 5V only operation

While the standard application circuit is limited to roughly 5.5V lower supply voltage, a 5V only application lets the IC run from a normal 5V +/-10% supply. In this application, linear regulator drop must be minimized. Therefore, the major 5V 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.

3.7 High Motor Current

When operating at a high motor current, the driver power dissipation due to MOSFET switch on-resistance significantly heats up the driver. This power dissipation will heat up the PCB cooling infrastructure also, if operated at an increased duty cycle. This in turn leads to a further increase of driver temperature. An increase of temperature by about 100°C increases MOSFET resistance by roughly 50%. This is a typical behavior of MOSFET switches. Therefore, under high duty cycle, high load conditions, thermal characteristics have to be carefully taken into account, especially when increased environment temperatures are to be supported. Refer the thermal characteristics and the layout hints for more information. As a thumb rule, thermal properties of the PCB design become critical for the QFN-36 at or above about 1000mA RMS motor current for increased periods of time. Keep in mind that resistive power dissipation raises with the square of the motor current. On the other hand, this means that a small reduction of motor current significantly saves heat dissipation and energy.

An effect which might be perceived at medium motor velocities and motor sine wave peak currents above roughly 1.2A peak is an increasing negative impact of increased internal diode conduction on the duration of the fast decay cycle of the spreadCycle chopper. This is, because the current measurement does not see the full coil current during this phase of the sine wave, because an increasing part of the current flows directly from the power MOSFETs' drain to GND and does not flow through the sense resistor. This in turn under some conditions may lead to a slight sine distortion of the current wave when using spreadCycle. This effect with most motors does not negatively influence the smoothness of operation, as it does not impact the critical current zero transition. It does not occur with stealthChop and with classic chopper.

3.7.1 Reduce Linear Regulator Power Dissipation

When operating at high supply voltages, as a first step the power dissipation of the integrated 5V linear regulator can be reduced, e.g. by using an external 5V source for supply. This will reduce overall heating. It is advised to reduce motor stand still current in order to decrease overall power dissipation. If applicable, also use coolStep. A decreased clock frequency will reduce power dissipation of the internal logic. Further a decreased chopper frequency also can reduce power dissipation.

3.7.2 Operation near to / above 2A Peak Current

The driver can deliver up to 2.5A motor peak current. Considering thermal characteristics, this only is possible in duty cycle limited operation. When a peak current up to 2.5A is to be driven, the driver chip temperature is to be kept at a maximum of 105°C. Linearly derate the design peak temperature from 125°C to 105°C in the range 2A to 2.5A output current (see Figure 3.8). Exceeding this may lead to triggering the short circuit detection.

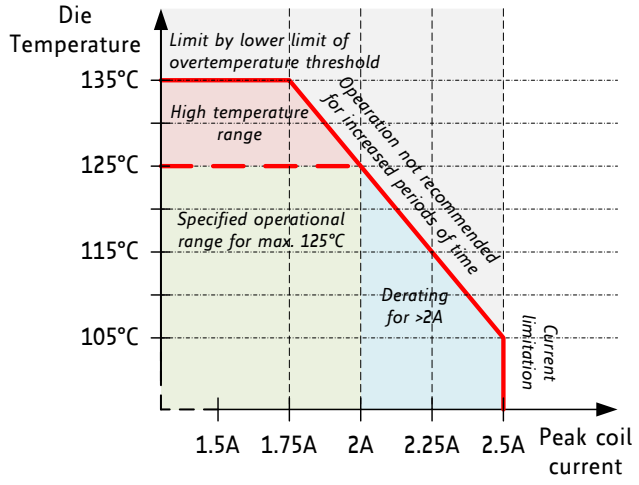


Figure 3.8 Derating of maximum sine wave peak current at increased die temperature

3.7.3 Reduction of Resistive Losses by Adding Schottky Diodes

Schottky Diodes can be added to the circuit to reduce driver power dissipation when driving high motor currents (see Figure 3.9). The Schottky diodes have a conduction voltage of about 0.5V and will take over more than half of the motor current during the negative half wave of each output in slow decay and fast decay phases, thus leading to a cooler motor driver. This effect starts from a few percent at 1.2A and increases with higher motor current rating up to roughly 20%. As a 30V Schottky diode has a lower forward voltage than a 50V or 60V diode, it makes sense to use a 30V diode when the supply voltage is below 30V. The diodes will have less effect when working with stealthChop due to lower times of diode conduction in the chopper cycle. At current levels below 1.2A coil current, the effect of the diodes is negligible.

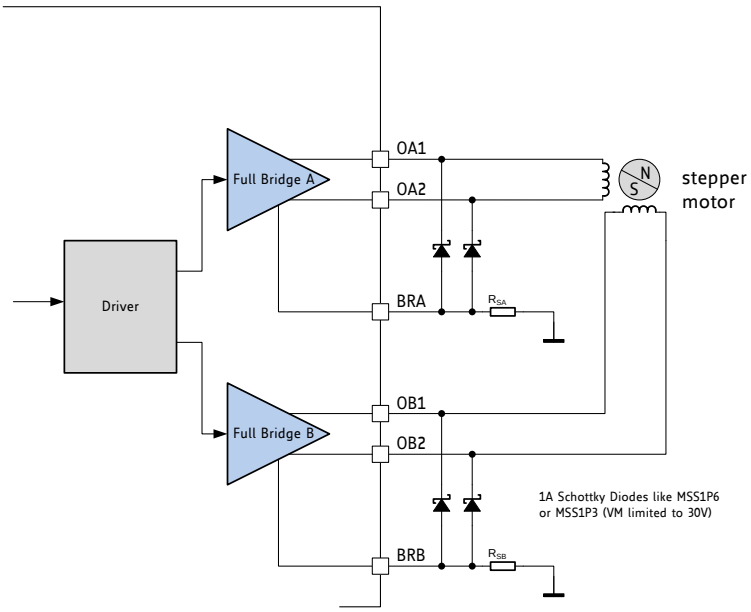


Figure 3.9 Schottky diodes reduce power dissipation at high peak currents up to 2A (2.5A)

3.8 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. 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 circuit and thus reduce electromagnetic emission. A more elaborate scheme uses LC filters to de-couple the driver outputs from the motor connector. Varistors eliminate coil overvoltage caused by live plugging. As LC filters tend to oscillate, additional snubber elements have been added. The drawback of this scheme is that it increases power dissipation significantly, especially at high supply voltages. A dampening resistor in parallel to the ferrite inductivity would be an option to the snubbers.

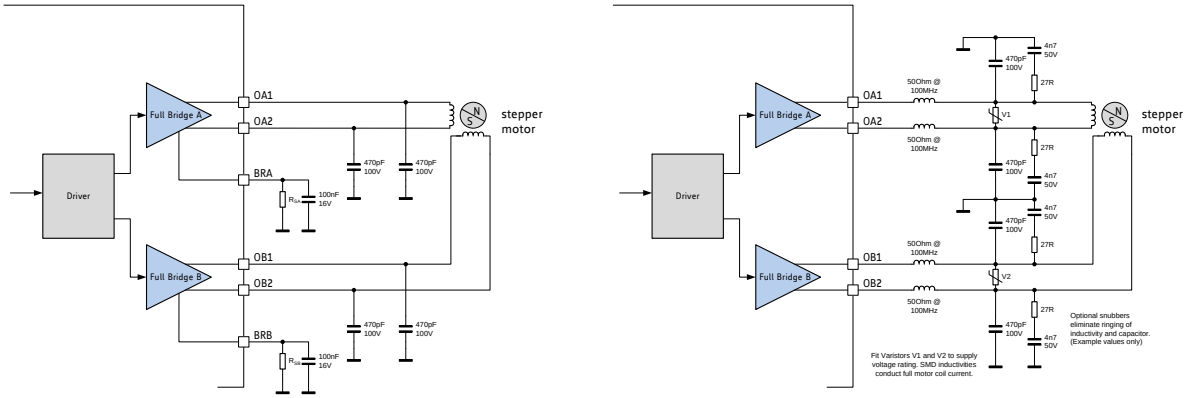


Figure 3.10 Simple ESD enhancement and more elaborate motor output protection

4 SPI Interface

4.1 SPI Datagram Structure

The TMC2130 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 device 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.

SPI DATAGRAM STRUCTURE

MSB (transmitted first)	40 bit	LSB (transmitted last)
39 …		… 0
→ 8 bit address **← 8 bit SPI status**	**← → 32 bit data**	
39 … 32	31 … 0	
→ to TMC2130 RW + 7 bit address ← from TMC2130 8 bit SPI status	8 bit data	8 bit data	8 bit data	8 bit data
39 / 38 … 32	31 … 24	23 … 16	15 … 8	7 … 0

w	38…32							31…28				27…24				23…20				19…16				15…12				11…8				7…4				3…0			
39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

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 TMC2130, 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 (*DRV_STATUS*) with the address 0x6F, the address byte has to be set to 0x6F in the access preceding the read access. For a write access to the register (*CHOPCONF*), the address byte has to be set to 0x80 + 0x6C = 0xEC. For read access, the data bit might have any value (-). So, one can set them to 0.

action	data sent to TMC2130	data received from TMC2130
read *DRV_STATUS*	→ 0x6F00000000	← 0xSS & unused data
read *DRV_STATUS*	→ 0x6F00000000	← 0xSS & DRV_STATUS
write *CHOPCONF*:= 0x00ABCDEF	→ 0xEC00ABCDEF	← 0xSS & DRV_STATUS
write *CHOPCONF*:= 0x00123456	→ 0xEC00123456	← 0xSS00ABCDEF

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

4.1.2 SPI Status Bits Transferred with Each Datagram Read Back

***SPI_STATUS* – status flags transmitted with each SPI access in bits 39 to 32**		
Bit	**Name**	**Comment**
7	-	*unused*
6	-	*unused*
5	-	*unused*
4	-	*unused*
3	*standstill*	*DRV_STATUS*[31] – 1: Signals motor stand still
2	*sg2*	*DRV_STATUS*[24] – 1: Signals stallguard flag active
1	*driver_error*	*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 TMC2130 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 TMC2130.

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.

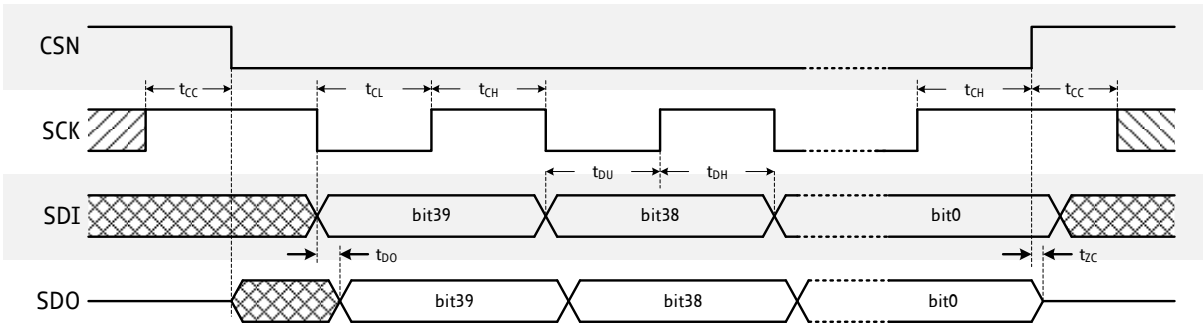


Figure 4.1 SPI timing

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 - interface configuration - and I/O signal configuration
Velocity Dependent 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 - setting thresholds for dcStep operation
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 - dcStep configuration - reading out stallGuard2 values and driver error flags
dcStep Minimum Velocity	Setting for minimum dcStep velocity

5.1 General Configuration Registers

GENERAL CONFIGURATION REGISTERS (0X00…0X0F)					
R/W	**Addr**	**n**	***Register***	**Description */ bit names***	
RW	0x00	17	*GCONF*	**Bit**	***GCONF* – Global configuration flags**
				0	*I_scale_analog* 0: Normal operation, use internal reference voltage 1: Use voltage supplied to AIN as current reference
				1	*internal_Rsense* 0: Normal operation 1: Internal sense resistors. Use current supplied into AIN as reference for internal sense resistor
				2	*en_pwm_mode* 1: stealthChop voltage PWM mode enabled (depending on velocity thresholds). Switch from off to on state while in stand still, only.
				3	*enc_commutation* 1: Enable commutation by full step encoder (DCIN_CFG5 = ENC_A, DCEN_CFG4 = ENC_B)
				4	*shaft* 1: Inverse motor direction
				5	*diag0_error* 1: Enable DIAG0 active on driver errors: Over temperature (*ot*), short to GND (*s2g*), undervoltage chargepump (*uv_cp*) DIAG0 always shows the reset-status, i.e. is active low during reset condition.
				6	*diag0_otpw* 1: Enable DIAG0 active on driver over temperature prewarning (*otpw*)
				7	*diag0_stall* 1: Enable DIAG0 active on motor stall (set *TCOOLTHRS* before using this feature)
				8	*diag1_stall* 1: Enable DIAG1 active on motor stall (set *TCOOLTHRS* before using this feature)
				9	*diag1_index* 1: Enable DIAG1 active on index position (microstep look up table position 0)
				10	*diag1_onstate* 1: Enable DIAG1 active when chopper is on (for the coil which is in the second half of the fullstep)
				11	*diag1_steps_skipped* 1: Enable output toggle when steps are skipped in dcStep mode (increment of *LOST_STEPS*). Do not enable in conjunction with other DIAG1 options.
				12	*diag0_int_pushpull* 0: DIAG0 is open collector output (active low) 1: Enable DIAG0 push pull output (active high)
				13	*diag1_pushpull* 0: DIAG1 is open collector output (active low) 1: Enable DIAG1 push pull output (active high)
				14	*small_hysteresis* 0: Hysteresis for step frequency comparison is 1/16 1: Hysteresis for step frequency comparison is 1/32

GENERAL CONFIGURATION REGISTERS (0X00…0X0F)					
R/W	**Addr**	**n**	***Register***	**Description */ bit names***	
				15	*stop_enable* 0: Normal operation 1: Emergency stop: DCIN stops the sequencer when tied high (no steps become executed by the sequencer, motor goes to standstill state).
				16	*direct_mode* 0: Normal operation 1: Motor coil currents and polarity directly programmed via serial interface: Register *XDIRECT* (0x2D) specifies signed coil A current (bits 8..0) and coil B current (bits 24..16). In this mode, the current is scaled by *IHOLD* setting. Velocity based current regulation of voltage PWM is not available in this mode. The automatic voltage PWM current regulation will work only for low stepper motor velocities.
				17	*test_mode* 0: Normal operation 1: Enable analog test output on pin DCO. *IHOLD*[1..0] selects the function of DCO: 0…2: T120, DAC, VDDH *Attention: Not for user, set to 0 for normal operation!*
R+C	0x01	3	*GSTAT*	**Bit**	***GSTAT* – Global status flags**
				0	*reset* 1: Indicates that the IC has been reset since the last read access to *GSTAT.*
				1	*drv_err* 1: Indicates, that driver 1 has been shut down due to an error since the last read access.
				2	*uv_cp* 1: Indicates an undervoltage on the charge pump. The driver is disabled in this case.
R	0x04	8 + 8	*IOIN*	**Bit**	***INPUT***
					Reads the state of all input pins available
				0	STEP
				1	DIR
				2	DCEN_CFG4
				3	DCIN_CFG5
				4	DRV_ENN_CFG6
				5	DCO
				6	This bit always shows 1.
				7	Don't care.
				31..24	*VERSION*: 0x11=first version of the IC Identical numbers mean full digital compatibility.

5.2 Velocity Dependent Driver Feature Control Register Set

<table>
<tr><th colspan="5">VELOCITY DEPENDENT DRIVER FEATURE CONTROL REGISTER SET (0X10…0X1F)</th></tr>
<tr><th>R/W</th><th>Addr</th><th>n</th><th>Register</th><th>Description / bit names</th></tr>
<tr><td rowspan="4">W</td><td rowspan="4">0x10</td><td rowspan="4">5
+
5
+
4</td><td rowspan="4">IHOLD_IRUN</td><td>Bit</td><td>IHOLD_IRUN – Driver current control</td></tr>
<tr><td>4..0</td><td>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.</td></tr>
<tr><td>12..8</td><td>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.</td></tr>
<tr><td>19..16</td><td>IHOLDDELAY
Controls the number of clock cycles for motor power down after a motion as soon as standstill is detected (stst=1) and TPOWERDOWN has expired. The smooth transition avoids a motor jerk upon power down.

0: instant power down
1..15: Delay per current reduction step in multiple of 2^18 clocks</td></tr>
<tr><td>W</td><td>0x11</td><td>8</td><td>TPOWER DOWN</td><td colspan="2">TPOWERDOWN sets the delay time after stand still (stst) of the motor to motor current power down. Time range is about 0 to 4 seconds.
0…((2^8)-1) * 2^18 t_{CLK}</td></tr>
<tr><td>R</td><td>0x12</td><td>20</td><td>TSTEP</td><td colspan="2">Actual measured time between two 1/256 microsteps derived from the step input frequency in units of 1/fCLK. Measured value is (2^20)-1 in case of overflow or stand still.

All TSTEP related thresholds use a hysteresis of 1/16 of the compare value to compensate for jitter in the clock or the step frequency. The flag small_hysteresis modifies the hysteresis to a smaller value of 1/32.
(Txxx*15/16)-1 or
(Txxx*31/32)-1 is used as a second compare value for each comparison value.
This means, that the lower switching velocity equals the calculated setting, but the upper switching velocity is higher as defined by the hysteresis setting.

In dcStep mode TSTEP will not show the mean velocity of the motor, but the velocities for each microstep, which may not be stable and thus does not represent the real motor velocity in case it runs slower than the target velocity.</td></tr>
<tr><td>W</td><td>0x13</td><td>20</td><td>TPWMTHRS</td><td colspan="2">This is the upper velocity for stealthChop voltage PWM mode.
TSTEP ≥ TPWMTHRS
- stealthChop PWM mode is enabled, if configured
- dcStep is disabled</td></tr>
</table>

VELOCITY DEPENDENT DRIVER FEATURE CONTROL REGISTER SET (0x10...0x1F)				
R/W	**Addr**	**n**	***Register***	**Description */ bit names***
W	0x14	20	*TCOOLTHRS*	This is the lower threshold velocity for switching on smart energy coolStep and stallGuard feature. (unsigned) Set this parameter to disable coolStep at low speeds, where it cannot work reliably. The stall detection and stallGuard output signal becomes enabled when exceeding this velocity. In non-dcStep mode, it becomes disabled again once the velocity falls below this threshold. *TCOOLTHRS* ≥ *TSTEP* ≥ *THIGH*: - coolStep is enabled, if configured - stealthChop voltage PWM mode is disabled *TCOOLTHRS* ≥ *TSTEP* - Stop on stall and stall output signal is enabled, if configured
W	0x15	20	*THIGH*	This velocity setting allows velocity dependent switching into a different chopper mode and fullstepping to maximize torque. (unsigned) The stall detection feature becomes switched off for 2-3 electrical periods whenever passing *THIGH* threshold to compensate for the effect of switching modes. *TSTEP* ≤ *THIGH*: - coolStep is disabled (motor runs with normal current scale) - stealthChop voltage PWM mode is disabled - If *vhighchm* is set, the chopper switches to *chm*=1 with *TFD*=0 (constant off time with slow decay, only). - chopSync2 is switched off (*SYNC*=0) - If *vhighfs* is set, the motor operates in fullstep mode and the stall detection becomes switched over to dcStep stall detection.

microstep velocity time reference t for velocities: TSTEP = f_{CLK} / f_{STEP}

5.3 SPI Mode Register

This register cannot be used in Step/Dir mode.

SPI MODE REGISTER (0X2D)					
R/W	**Addr**	**n**	***Register***	**Description */ bit names***	**Range [Unit]**
RW	0x2D	32	*XDIRECT*	*direct_mode* 0: Normal operation 1: Motor coil currents and polarity directly programmed via serial interface: Register *XDIRECT* (0x2D) specifies signed coil A current (bits 8..0) and coil B current (bits 24..16). In this mode, the current is scaled by *IHOLD* setting. Velocity based current regulation of voltage PWM is not available in this mode. The automatic voltage PWM current regulation will work only for low stepper motor velocities.	±255 for both coils

5.4 dcStep Minimum Velocity Register

DCSTEP MINIMUM VELOCITY REGISTER (0X33)				
R/W	**Addr**	**n**	***Register***	**Description */ bit names***
W	0x33	23	*VDCMIN*	The automatic commutation dcStep becomes enabled by the external signal DCEN. *VDCMIN* is used as the minimum step velocity when the motor is heavily loaded. *Hint:* Also set *DCCTRL* parameters in order to operate dcStep.

time reference t for VDCMIN: t = 2^24 / f_{CLK}

5.5 Motor Driver Registers

MICROSTEPPING CONTROL REGISTER SET (0X60…0X6B)					
R/W	**Addr**	**n**	***Register***	**Description *I 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	32x 0 or 1 *reset default= sine wave table*
W	0x61 … 0x67	7 x 32	*MSLUT[1…7]* microstep table entries 32…255	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*	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_SIN 90* *reset default =247*
R	0x6A	10	*MSCNT*	Microstep counter. Indicates actual position in the microstep table for *CUR_A*. *CUR_B* uses an offset of 256 (2 phase motor). *Hint:* Move to a position where *MSCNT* is zero before re-initializing *MSLUTSTART* or *MSLUT* and *MSLUTSEL*.	0…1023
R	0x6B	9 + 9	*MSCURACT*	bit 8… 0: *CUR_A* (signed): Actual microstep current for motor phase A as read from MSLUT (not scaled by current) bit 24… 16: *CUR_B* (signed): Actual microstep current for motor phase B as read from *MSLUT* (not scaled by current)	+/-0…255

DRIVER REGISTER SET (0X6C…0X7F)					
R/W	**Addr**	**n**	***Register***	**Description */ bit names***	**Range [Unit]**
RW	0x6C	32	***CHOPCONF***	chopper and driver configuration *See separate table!*	
W	0x6D	25	***COOLCONF***	coolStep smart current control register and stallGuard2 configuration *See separate table!*	
W	0x6E	24	*DCCTRL*	dcStep (*DC*) automatic commutation configuration register (enable via pin DCEN or via *VDCMIN*): bit 9… 0: *DC_TIME*: Upper PWM on time limit for commutation (*DC_TIME* * $1/f_{CLK}$). Set slightly above effective blank time *TBL*. bit 23… 16: *DC_SG*: Max. PWM on time for step loss detection using dcStep stallGuard2 in dcStep mode. (*DC_SG* * $16/f_{CLK}$) Set slightly higher than *DC_TIME*/16 0=disable *Attention: Using a higher microstep resolution or interpolated operation, dcStep delivers a better stallGuard signal.* *DC_SG is also available above VHIGH if vhighfs is activated. For best result also set vhighchm.*	
R	0x6F	32	***DRV_ STATUS***	stallGuard2 value and driver error flags *See separate table!*	
W	0x70	22	***PWMCONF***	Voltage PWM mode chopper configuration *See separate table!*	*reset default= 0x00050480*
R	0x71	8	*PWM_SCALE*	Actual PWM amplitude scaler (255=max. Voltage) In voltage mode PWM, this value allows to detect a motor stall.	0…255
W	0x72	2	*ENCM_CTRL*	Encoder mode configuration Bit 0: *inv*: Invert encoder inputs Bit 1: *maxspeed*: Ignore Step input. If set, the hold current *IHOLD* determines the motor current, unless a step source is activated. The direction in this mode is determined by the *shaft* bit in *GCONF* or by the *inv* bit.	
R	0x73	20	*LOST_STEPS*	Number of input steps skipped due to higher load in dcStep operation, if step input does not stop when DC_OUT is low. This counter wraps around after 2^20 steps. Counts up or down depending on direction. Only with SDMODE=1.	

MICROSTEP 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

5.5.1 *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			

5.5.2 *CHOPCONF* – Chopper Configuration

0x6C: *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	*dedge*	enable double edge step pulses	1: Enable step impulse at each step edge to reduce step frequency requirement.	
28	*intpol*	interpolation to 256 microsteps	1: The actual microstep resolution (*MRES*) becomes extrapolated to 256 microsteps for smoothest motor operation.	
27	*mres3*	*MRES* micro step resolution	%0000: Native 256 microstep setting.	
26	*mres2*			
25	*mres1*			
24	*mres0*		%0001 … %1000: 128, 64, 32, 16, 8, 4, 2, FULLSTEP Reduced microstep resolution for Step/Dir operation. The resolution gives the number of microstep entries per sine quarter wave. The driver automatically uses microstep positions which result in a symmetrical wave, when choosing a lower microstep resolution. step width=2^*MRES* [microsteps]	
23	*sync3*	*SYNC* PWM synchronization clock	This register allows synchronization of the chopper for both phases of a two phase motor in order to avoid the occurrence of a beat, especially at low motor velocities. It is automatically switched off above *VHIGH*. %0000: Chopper sync function chopSync off %0001 … %1111: Synchronization with $f_{SYNC} = f_{CLK}/(sync*64)$ *Hint:* Set *TOFF* to a low value, so that the chopper cycle is ended, before the next sync clock pulse occurs. Set for the double desired chopper frequency for *chm*=0, for the desired base chopper frequency for *chm*=1.	
22	*sync2*			
21	*sync1*			
20	*sync0*			
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 is 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.

0x6C: *CHOPCONF* – CHOPPER CONFIGURATION				
Bit	**Name**	**Function**	**Comment**	
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	
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}= 12 + 32**TOFF* %0000: Driver disable, all bridges off %0001: 1 – use only with *TBL* ≥ 2 %0010 … %1111: 2 … 15	
2	*toff2*			
1	*toff1*			
0	*toff0*			

5.5.3 *COOLCONF* – Smart Energy Control coolStep and stallGuard2

0x6D: *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 (resp. six fullsteps for 3 phase motor) 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*		

5.5.4 *PWMCONF* – Voltage PWM Mode stealthChop

0x70: *PWMCONF* – VOLTAGE MODE PWM STEALTHCHOP				
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	*pwm_ symmetric*	Force symmetric PWM	0	The PWM value may change within each PWM cycle (standard mode)
			1	A symmetric PWM cycle is enforced
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}=1/1024 f_{CLK} %01: f_{PWM}=1/683 f_{CLK} %10: f_{PWM}=1/512 f_{CLK} %11: f_{PWM}=1/410 f_{CLK}	
15 14 13 12	*PWM_ GRAD*	User defined amplitude (gradient) or regulation loop gradient	pwm_ autoscale=0	Velocity dependent gradient for PWM amplitude: *PWM_GRAD* * 256 / *TSTEP* is added to *PWM_AMPL*
11 10 9 8			pwm_ autoscale=1	User defined maximum PWM amplitude change per half wave (1 to 15)
7 6 5 4	*PWM_ AMPL*	User defined amplitude (offset)	pwm_ autoscale=0	User defined PWM amplitude offset (0-255) The resulting amplitude (limited to 0…255) is: *PWM_AMPL* + *PWM_GRAD* * 256 / *TSTEP*
3 2 1 0			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.

5.5.5 DRV_STATUS – stallGuard2 Value and Driver Error Flags

0x6F: 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. This occurs 2^20 clocks after the last step pulse.
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, 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 bridges.
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 bridges.
24	*stallGuard*	stallGuard2 status	1: Motor stall detected (*SG_RESULT*=0) or dcStep stall in dcStep mode.
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 stand still 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 in order 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. *SG_RESULT* is ALSO applicable when dcStep is active. 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			

6 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 absolutely free of vibration at low velocities. With stealthChop, the motor current is applied by driving a certain effective voltage into the coil, using a voltage mode PWM. 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.

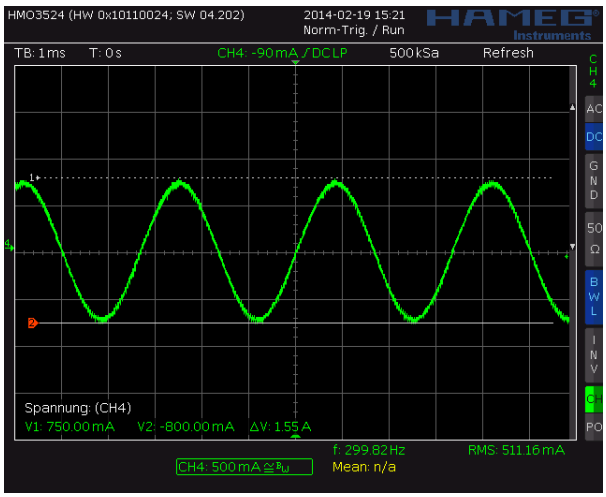


Figure 6.1 Motor coil sine wave current with stealthChop (measured with current probe)

6.1 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 (i.e. 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 feed forward velocity controlled mode (*pwm_autoscale* = 0). The feed forward velocity controlled mode will not react to a change of the supply voltage or to events like a motor stall, but it provides very stable amplitude. It does not use nor require any means of current measurement. This is perfect when motor type and supply voltage are well known. 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 in order to adapt the frequency divider to the frequency of the clock source. An optimum setting is slightly above the audible frequency range. A slightly higher value might bring a benefit for some applications.

CHOICE OF PWM FREQUENCY FOR STEALTHCHOP				
Clock frequency f_{CLK}	**PWM_FREQ=%00** f_{PWM}=1/1024 f_{CLK}	**PWM_FREQ=%01** f_{PWM}=1/683 f_{CLK}	**PWM_FREQ=%10** f_{PWM}=1/512 f_{CLK}	**PWM_FREQ=%11** f_{PWM}=1/410 f_{CLK}
18MHz	17.6kHz	26.3kHz	35.2kHz	
16MHz	15.6kHz	23.4kHz	31.2kHz	39.0kHz
(internal)	~13kHz	~19kHz	~26kHz	~32kHz
12MHz	11.7kHz	17.6kHz	23.4kHz	29.3kHz
10MHz		14.6kHz	19.5kHz	24.4kHz
8MHz		11.7kHz	15.6kHz	19.5kHz

Table 6.1 Choice of PWM frequency – green: recommended

6.2 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 in order 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 6.4). 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 in order 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.

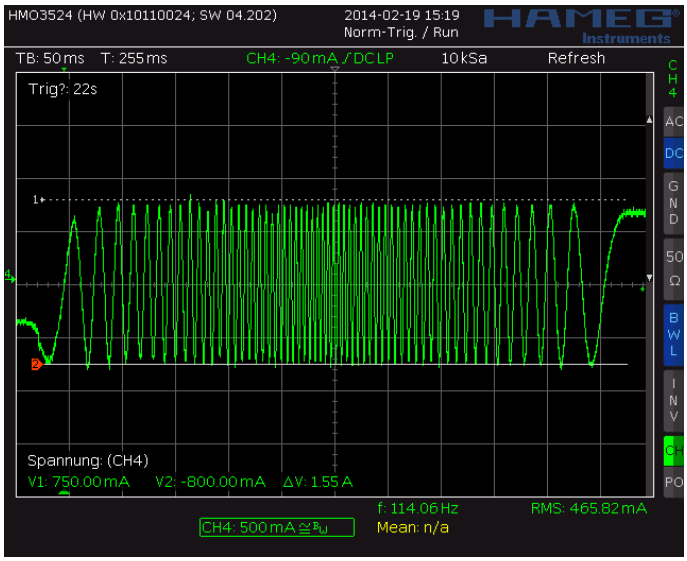


Figure 6.2 Scope shot: good setting for PWM_GRAD

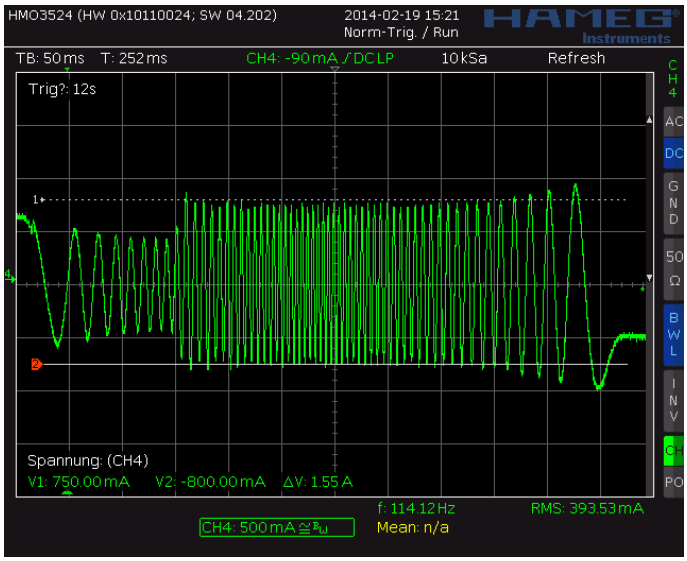


Figure 6.3 Scope shot: too small setting for PWM_GRAD

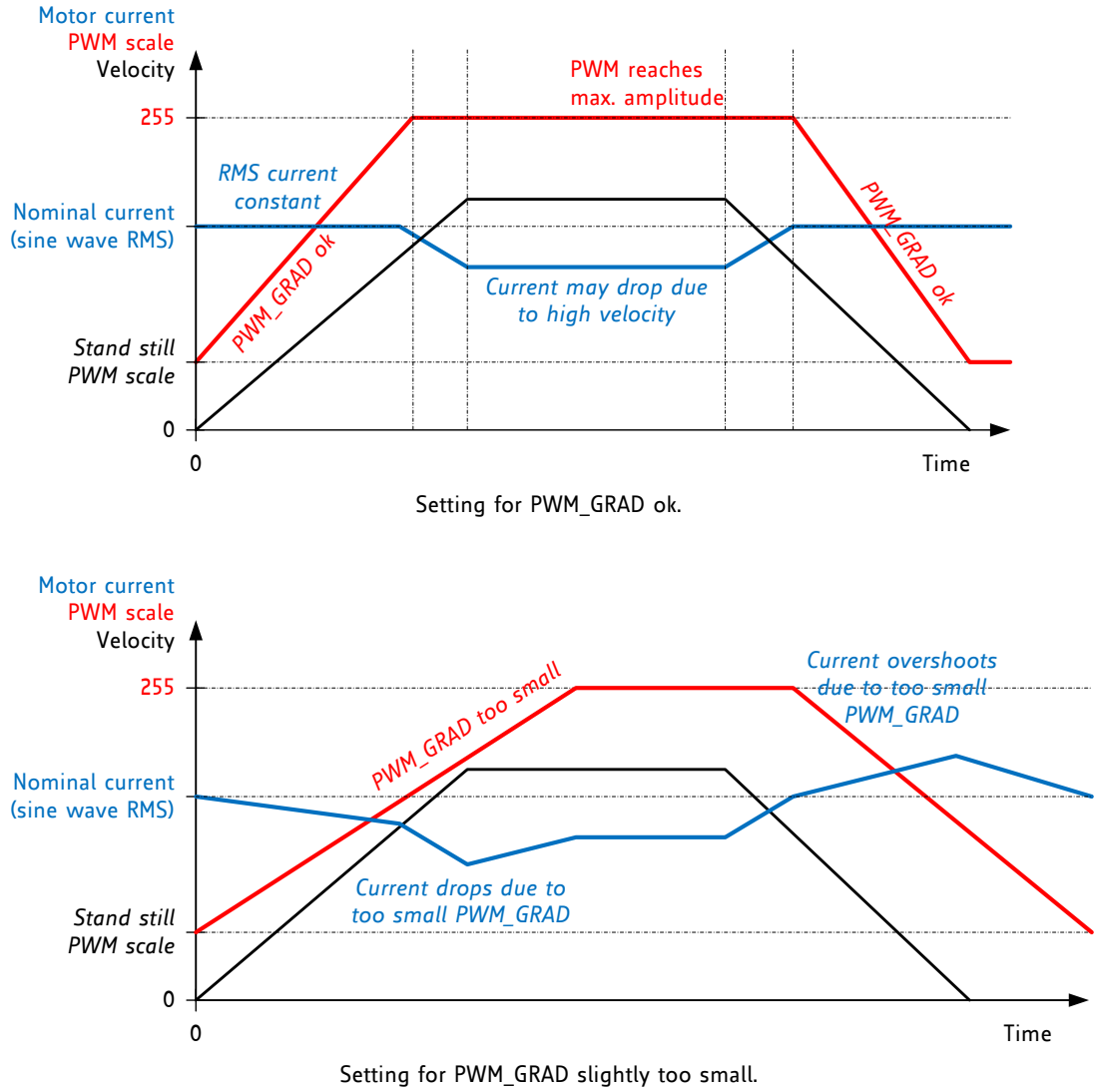


Figure 6.4 Good and too small setting for PWM_GRAD

However, the autoscaling function cannot measure low RMS current settings with high supply voltages and low inductive motors, because chopper on time may become too short for successfully measuring the motor current. Especially at higher PWM frequency settings the lower current limit rises. This happens when the PWM on time required for reaching the necessary coil current falls below the comparator blank time. Try a lower blanking time in order to reduce the 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. 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.

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 22.

6.2.1 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 *TPWMTHRS*. 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 6.4 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 *TPWMTHRS*), *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.

6.3 Velocity Based Scaling

Velocity based scaling scales the stealthChop amplitude based on the time between each two steps, i.e. based on *TSTEP*. The basic idea is to have a linear approximation of the voltage required to drive the target current into the motor. The stepper motor has a certain coil resistance and thus needs a certain voltage amplitude to yield a target current based on the basic formula I=U/R. With R being the coil resistance, U the supply voltage scaled by the PWM value, the current I results. The initial value for PWM_AMPL 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. The TMC2130 provides a second velocity dependent factor (*PWM_GRAD*) to compensate for this. The overall effective PWM amplitude (*PWM_SCALE*) in this mode automatically is calculated in dependence of the microstep frequency as:

$$PWM\_SCALE = PWM\_AMPL + PWM\_GRAD * 256 * \frac{f_{STEP}}{f_{CLK}}$$

With f_{STEP} being the microstep frequency for 256 microstep resolution equivalent and f_{CLK} the clock frequency supplied to the driver or the actual internal frequency

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

$$PWM\_GRAD = C_{BEMF}\left[\frac{V}{\frac{rad}{s}}\right] * 2\pi * \frac{f_{CLK} * 1.46}{V_M * MSPR}$$

C_{BEMF} is the back EMF constant of the motor in Volts per radian/second
MSPR is the number of microsteps per rotation, e.g. 51200 = 256µsteps multiplied by 200 fullsteps for a 1.8° motor.

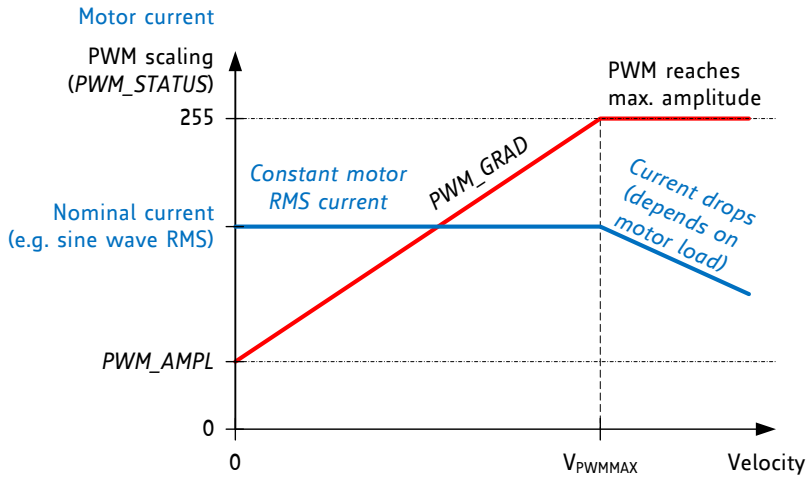


Figure 6.5 Velocity based PWM scaling (pwm_autoscale=0)

Hint
The values for *PWM_AMPL* and *PWM_GRAD* 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.

UNDERSTANDING THE BACK EMF CONSTANT OF A MOTOR

The back EMF constant is the voltage a motor generates when turned with a certain velocity. Often motor datasheets do not specify this value, as it can be deducted from motor torque and coil current rating. Within SI units, the numeric value of the back EMF constant C_{BEMF} has the same numeric value as the numeric value of the torque constant. For example, a motor with a torque constant of 1 Nm/A would have a C_{BEMF} of 1V/rad/s. Turning such a motor with 1 rps (1 rps = 1 revolution per second = 6.28 rad/s) generates a back EMF voltage of 6.28V. Thus, the back EMF constant can be calculated as:

$$C_{BEMF}\left[\frac{V}{rad/s}\right] = \frac{HoldingTorque[Nm]}{2 * I_{COILNOM}[A]}$$

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

6.3.1 Acceleration

At higher velocities with a motor not driving any load, the voltage PWM mode might not be able to dampen motor oscillations which are intrinsic to the motor. Therefore it is advised to operate the motor within the application when evaluating this chopper mode. 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.

6.4 Combining stealthChop with other Chopper Modes

The TMC2130 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. *TPWMTHRS* 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 *TPWMTHRS* should be set to a low velocity, in order to eliminate any jerk in case an automatic switching between two chopper modes is desired. Set *TPWMTHRS* zero 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 in order 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.

6.5 Flags in stealthChop

6.5.1 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.

6.5.2 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).

6.6 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 *TPOWERDOWN* and *IHOLD_DELAY*. The *PWM_SCALE* regulation becomes frozen once the motor target current is at zero current in order to ensure a quick startup.

Parameter	Description	Setting	Comment
en_pwm_mode	General enable for use of stealthChop (register *GCONF*)	0	Do not use stealthChop
		1	stealthChop enabled
TPWMTHRS	Specifies the upper velocity for operation in stealthChop voltage PWM mode. Entry the *TSTEP* reading (time between two microsteps) when operating at the desired threshold velocity.	0 … 1048575	stealthChop also is disabled if *TSTEP* falls below *TCOOLTHRS* or *THIGH*
pwm_autoscale	Enable automatic current scaling using current measurement or use forward controlled velocity based mode.	0	Forward controlled 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}=1/1024 f_{CLK}
		1	f_{PWM}=1/683 f_{CLK}
		2	f_{PWM}=1/512 f_{CLK}
		3	f_{PWM}=1/410 f_{CLK}
PWM_GRAD	User defined PWM amplitude (gradient) for velocity based scaling or regulation loop gradient when *pwm_autoscale*=1.	1 … 15	With *pwm_autoscale*=1
		0 … 255	With *pwm_autoscale*=0
PWM_AMPL	User defined PWM amplitude (offset) for velocity based scaling or amplitude limit for re-entry into stealthChop mode when *pwm_autoscale*=1.	0 … 255	
pwm_symmetric	Activate to force a symmetric PWM for each cycle. Reduces the number of updates to the PWM cycle. Special use only.	0	Normal operation
		1	A symmetric PWM cycle is enforced
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

7 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 7.1 the different chopper phases are shown.

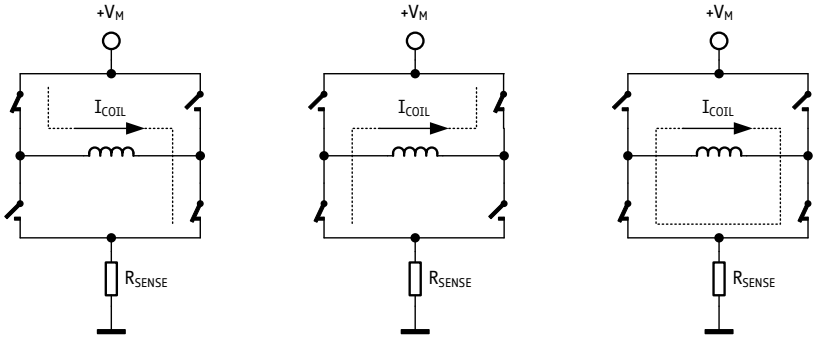


Figure 7.1 Chopper phases

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}= 12 + 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

7.1 spreadCycle Chopper

The patented spreadCycle chopper algorithm is a precise and simple to use chopper mode which automatically determines the optimum length for the fast-decay phase. The spreadCycle 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 7.3). 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 50%-75% 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 68% of overall chopper cycle time

$$t_{OFF} = \frac{1}{30kHz} * \frac{50}{100} * \frac{1}{2} = 8.3\mu s$$

For the *TOFF* setting this means:

$$TOFF = (t_{OFF} * f_{CLK} - 12)/32$$

With 12 MHz clock this gives a setting of TOFF=4.75, i.e. 4 or 5.
With 16 MHz clock this gives a setting of TOFF=6.43, i.e. 6 or 7.

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 in order 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 7.2). 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.

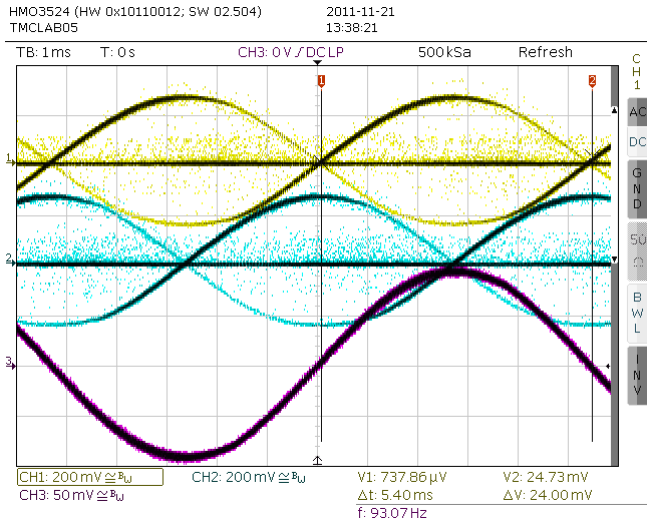


Figure 7.2 No ledges in current wave with sufficient hysteresis (magenta: current A, yellow & blue: sense resistor voltages A and B)

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 22**Fehler! Verweisquelle konnte nicht gefunden werden.**.
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.

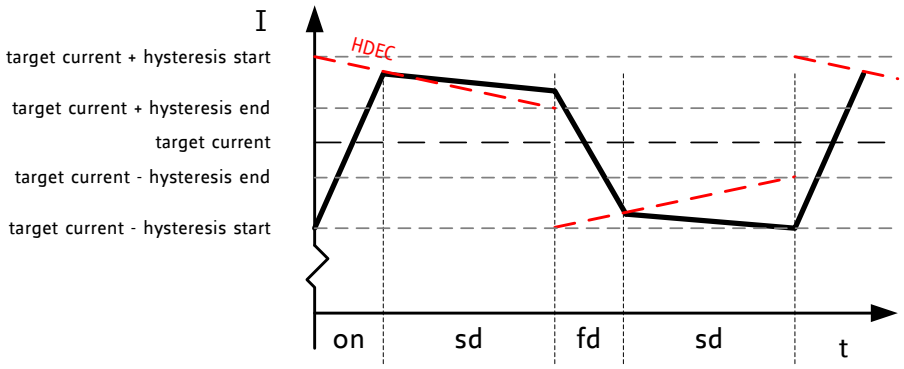


Figure 7.3 spreadCycle chopper scheme showing coil current during a chopper cycle

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 TMC2130 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 or 2 and a short TBL of 1 or 0.

7.2 Classic Constant Off Time Chopper

The classic constant off time chopper is an alternative to spreadCycle. Perfectly tuned, it also gives good results. In combination with RDSon current sensing without external sense resistors, this chopper mode can bring a benefit with regard to audible high-pitch chopper noise. Also, the classic constant off time chopper (automatically) is used in combination with fullstepping in dcStep operation.

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.

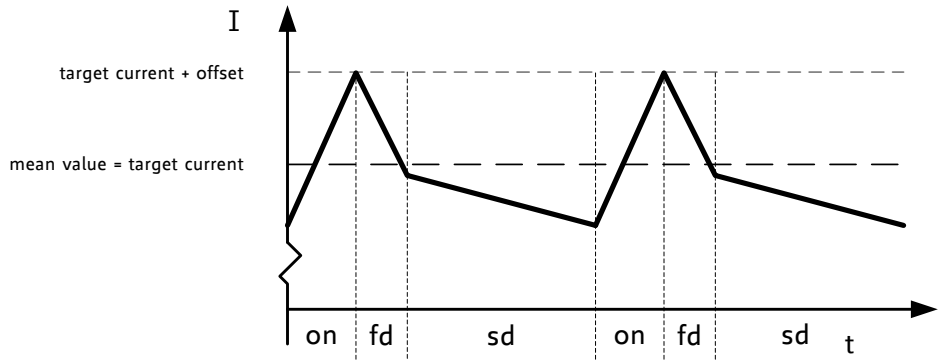


Figure 7.4 Classic const. off time chopper with offset showing coil current

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 7.5). 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.

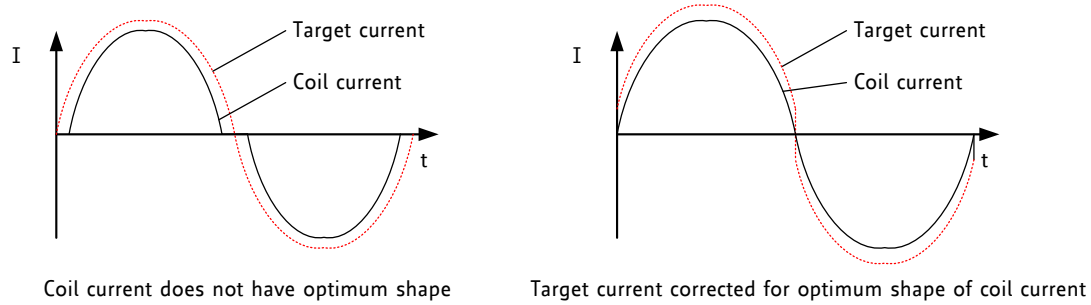


Figure 7.5 Zero crossing with classic chopper and correction using sine wave offset

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

7.3 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 29).

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

7.4 chopSync2 for Quiet 2-Phase Motor

chopSync2 is an alternative add-on concept for spreadCycle chopper and constant off time chopper to optimize motor noise at low velocities. When using stealthChop for low velocity operation, chopSync2 is not applicable.

While a frequency adaptive chopper like spreadCycle provides excellent high velocity operation, in some applications, a constant frequency chopper is preferred rather than a frequency adaptive chopper. This may be due to chopper noise in motor standstill, or due to electro-magnetic emission. chopSync2 provides a means to synchronize the choppers for both coils with a common clock, by extending the off time of the coils. It integrates with both chopper principles. However, a careful set up of the chopper is necessary, because chopSync2 can just increment the off times, but not reduce the duration of the chopper cycles themselves. Therefore, it is necessary to test successful operation best with an oscilloscope. Set up the chopper as detailed above, but take care to have chopper frequency higher than the chopSync2 frequency. As high motor velocities take advantage of the normal, adaptive chopper style, chopSync2 becomes automatically switched off using the *VHIGH* velocity limit programmed within the motion controller.

A suitable chopSync2 *SYNC* value can be calculated as follows:

$$SYNC = \left\lfloor \frac{f_{CLK}}{64 * f_{SYNC}} \right\rfloor$$

Example:

The motor is operated in spreadCycle mode (*chm*=0). The minimum chopper frequency for standstill and slow motion (up to *VHIGH*) has been determined to be 25 kHz under worst case operation conditions (hot motor, low supply voltage). The standstill noise needs to be minimized by using chopSync. The IC uses an external 16 MHz clock.

Considering the chopper mode 0, *SYNC* has to be set for the closest value resulting in or below the double frequency, e.g. 50 kHz. Using above formula, a value of 5 results exactly and can be used. Trying a value of 6, a frequency of 41.7 kHz results, which still gives an effective chopper frequency of slightly above 20 kHz, and thus would also be a valid solution. A value of 7 might still be good, but could already give high frequency noise.

In chopper mode 1, *SYNC* could be set to any value between 10 and 13 to be within the chopper frequency range of 19.8 kHz to 25 kHz.

Parameter	Description	Setting	Comment
SYNC	This register allows synchronization of the chopper for both phases of a two phase motor in order to avoid the occurrence of a beat, especially at low motor velocities. It is automatically switched off above *VHIGH*. *Hint:* Set *TOFF* to a low value, so that the chopper cycle is ended, before the next sync clock pulse occurs. Set *SYNC* for the double desired chopper frequency for *chm*=0, for the desired base chopper frequency for *chm*=1.	0	chopSync off
		1...15	$f_{CLK}/64$ ... $f_{CLK}/(15*64)$

8 Analog Current Control AIN

When a high flexibility of the output current scaling is desired, the analog input of the driver can be enabled for current control, rather than choosing a different set of sense resistors or scaling down the run current via *IRUN* parameter. This way, a simple voltage divider can be used for the adaptation of a board to different motors.

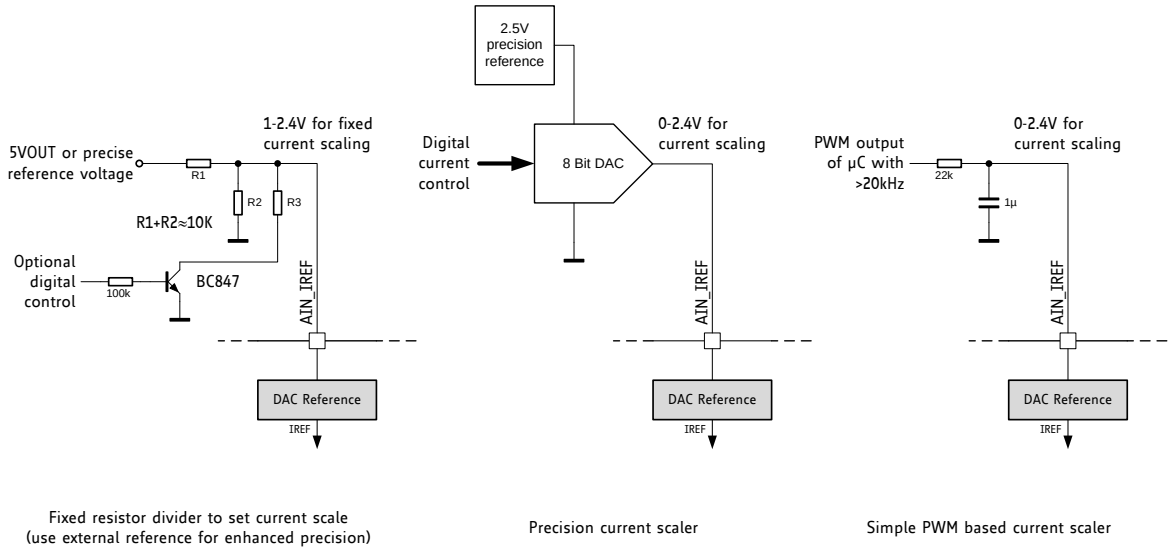
AIN Scales the Motor Current

The TMC2130 provides an internal reference voltage for current control, directly derived from the 5VOUT supply output. Alternatively, an external reference voltage can be used. This reference voltage becomes scaled down for the chopper comparators. The chopper comparators compare the voltages on BRA and BRB to the scaled reference voltage for current regulation. When *I_scale_analog* in *GCONF* is enabled, the external voltage on AIN is amplified and filtered and becomes used as reference voltage. A voltage of 2.5V (or any voltage between 2.5V and 5V) gives the same current scaling as the internal reference voltage. A voltage between 0V and 2.5V linearly scales the current between 0 and the current scaling defined by the sense resistor setting. It is not advised to work with reference voltages below about 0.5V to 1V, because relative analog noise caused by digital circuitry has an increased impact on the chopper precision at low AIN voltages. For best precision, choose the sense resistors in a way that the desired maximum current is reached with AIN in the range 2V to 2.4V. Be sure to optimize the chopper settings for the normal run current of the motor.

Driving AIN

The easiest way to provide a voltage to AIN is to use a voltage divider from a stable supply voltage or a microcontroller's DAC output. A PWM signal can also be used for current control. The PWM becomes transformed to an analog voltage using an additional R/C low-pass at the AIN pin. The PWM duty cycle controls the analog voltage. Choose the R and C values to form a low pass with a corner frequency of several milliseconds while using PWM frequencies well above 10 kHz. AIN additionally provides an internal low-pass filter with 3.5kHz bandwidth. The integration of an NTC into the voltage divider feeding AIN allows the realization of temperature dependent motor current scaling. When a precise reference voltage is available (e.g. from TL431A), the precision of the motor current regulation can be improved when compared to the internal voltage reference.

Using a low reference voltage (e.g. below 1V), for adaptation of a high current driver to a low current motor will lead to reduced analog performance. Adapting the sense resistors to fit the desired motor current gives a better result.



Fixed resistor divider to set current scale (use external reference for enhanced precision)

Precision current scaler

Simple PWM based current scaler

Figure 8.1 Scaling the motor current using the analog input

9 Current Setting

The internal 5 V 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.

When *I_scale_analog* is enabled for analog scaling of V_{FS}, the resulting voltage V_{FS}' is calculated by:

$$V'_{FS} = V_{FS} * \frac{V_{AIN}}{2.5V}$$

with V_{AIN} the voltage on pin AIN_IREF in the range 0V to V_{5VOUT}/2

CHOICE OF R_{SENSE} AND RESULTING MAX. MOTOR CURRENT		
R_{SENSE} [Ω]	***RMS current [A] (CS=31, vsense=0)***	***RMS current [A] (CS=31, vsense=1)***
1.00	0.23	0.12
0.82	0.27	0.15
0.75	0.30	0.17
0.68	0.33	0.18
0.50	0.44	0.24
0.47	0.47	0.26
0.33	0.66	0.36
0.27	0.79	0.44
0.22	0.96	0.53
0.15	1.35	0.75
0.12	1.64	0.91
0.10	1.92*)	1.06

*) Value exceeds upper current rating.

Hint
For best precision of current setting, it is advised to measure and fine tune the current in the application.

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 *TPOWERDOWN* 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

9.1 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 29.

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.

10 Using RDSon to Eliminate Sense Resistors

The TMC2130 provides the option to eliminate external sense resistors. In this mode the external sense resistors become omitted (shorted) and the internal on-resistance of the power MOSFETs is used for current measurement (see Figure 3.3). As MOSFETs are both, temperature dependent and subject to production stray, a tiny external resistor connected from +5VOUT to AIN/IREF is used to provide a precise absolute current reference. This resistor converts the 5V voltage into a reference current. Be sure to directly attach BRA and BRB pins to GND in this mode near the IC package. The mode is enabled by setting *internal_Rsense* in *GCONF*.

10.1 Limitations of RDSon Sensing

While the RDSon based measurements bring benefits concerning cost and size of the driver, it gives slightly less precise current setting when compared to external sense resistors. External sense resistors provide the possibility to adapt the driver to a wide range of motor operation currents without trading in more noise due to scaling down chopper comparator reference voltage. Therefore the use of RDSon based measurement should be considered for motor currents between 0.4A RMS and 1.2A RMS. For lower and higher current motors, the performance should be evaluated first. Low current motors (below 0.4A RMS) also perform well when using RDSon measurement in combination with the stealthChop mode, because digital scaling does not directly increase chopper noise and reduce effective microstep performance. RDSon sensing may suffer from increased chopper noise and reduced microstep precision in combination with spreadCycle, because the current measurement required for spreadCycle does not guarantee the same precision and symmetry as a sense resistor can deliver. Therefore, consider using classic constant off time chopper instead of spreadCycle in case audible high pitch chopper noise appears.

10.2 Dimensioning of Reference Resistor

For RDSon measurement, up to 1.5A (2A max.) peak current can be driven into the motor (with *vsense*=0). An external reference current into the AIN/IREF pin is used as a reference current. AIN/IREF input resistance is about 1kOhm. In order to realize a certain current a single resistor (R_{REF}) can be connected between 5VOUT and AIN/IREF (pls. refer the table for the choice of the resistor). The resulting current into AIN/IREF is amplified 3000 times. Thus, a current of 0.5mA yields a motor current of 1.5A peak. When using reference currents above 0.5mA resulting in higher theoretical current settings of up to 2A, the resulting current decreases linearly when chip temperature exceeds a certain maximum temperature. For a 2A setting it decreases from 2A at up to 100°C down to about 1.5A at 150°C. The resulting curve limits the maximum current setting in this mode. For calculation of the reference resistor, the internal resistance of AIN/RREF needs to be considered additionally.

vsense=1 allows a lower peak current setting of about 55% of the value yielded with *vsense*=0 (as specified by V_{SRTH} / V_{SRTL}). For fine tuning use the current scale *CS*.

CHOICE OF R_{REF} FOR OPERATION WITHOUT SENSE RESISTORS		
R_{REF} [Ω]	***Peak current [A] (CS=31, vsense=0)***	***Peak current [A] (CS=31, vsense=1)***
6k8	1.92	1.06
7k5	1.76	0.97
8k2	1.63	0.90
9k1	1.49	0.82
10k	1.36	0.75
12k	1.15	0.63
15k	0.94	0.52
18k	0.79	0.43
22k	0.65	0.36
27k	0.60	0.33
33k	0.54	0.29

In RDSon measurement mode, connect the BRA and BRB pins to GND using the shortest possible path (i.e. lowest possible PCB resistance). In a realistic setup, the effective current will be slightly lower than expected. RDSon based measurement gives best results when combined with classic constant off time chopper or with the voltage PWM stealthChop. When using spreadCycle with RDSon based current measurement, slightly asymmetric current measurement for positive currents (on phase) and negative currents (fast decay phase) can result in chopper noise. This especially occurs at increased die temperature and increased motor current.

Note
The absolute current levels achieved with RDSon based current sensing may depend on PCB layout exactly like with external sense resistors, because trace resistance on BR pins will add to the effective sense resistance. Therefore we recommend to measure and calibrate the current setting within the application.

Thumb rule
RDSon based current sensing works best for motors with up to 1.2A RMS current. The best results are yielded with stealthChop operation in combination with RDSon based current sensing. Consider using classic chopper rather than spreadCycle.
For most precise current control and best results with spreadCycle, it is recommended to use external 1% sense resistors rather than RDSon based current control.

11 Velocity Based Mode Control

The TMC2130 allows the configuration of different chopper modes and modes of operation for optimum motor control. Depending on the motor load, the different modes can be optimized for lowest noise & high precision, highest dynamics, or maximum torque at highest velocity. Some of the features like coolStep or stallGuard2 are useful in a limited velocity range. A number of velocity thresholds allow combining the different modes of operation within an application requiring a wide velocity range.

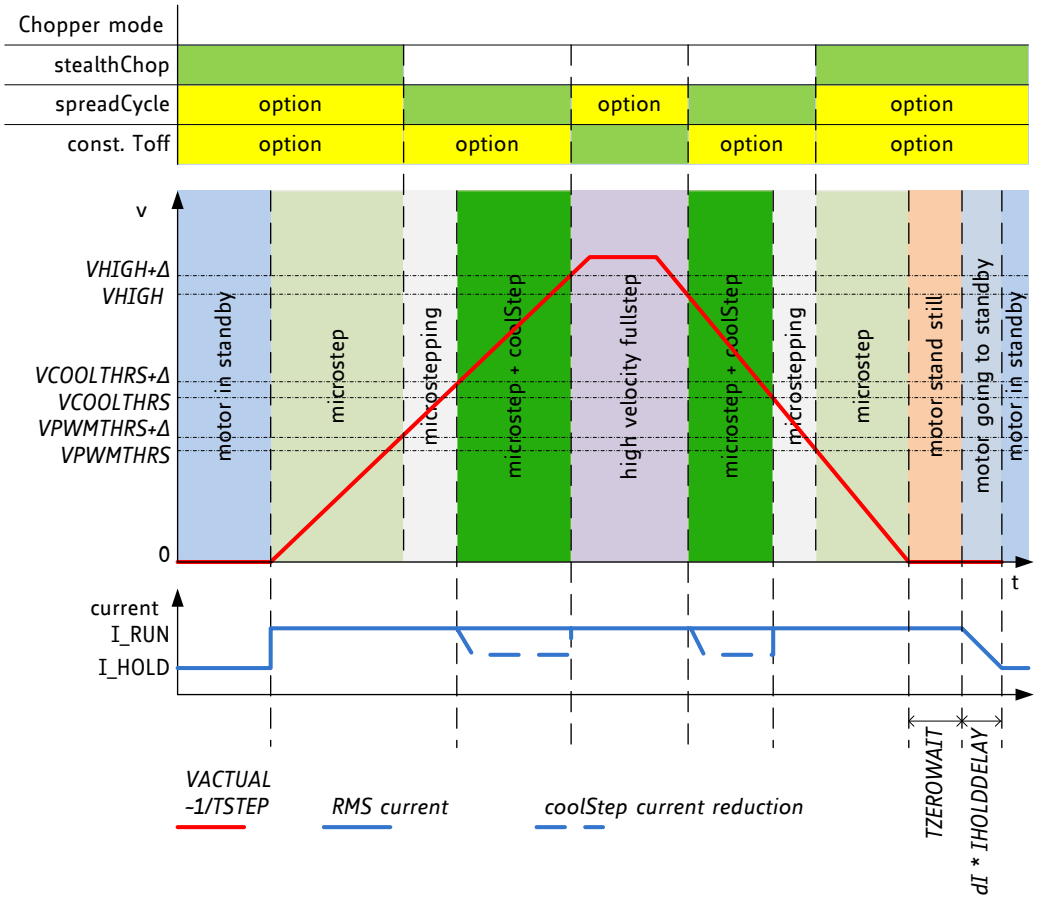


Figure 11.1 Choice of velocity dependent modes

Figure 11.1 shows all available thresholds and the required ordering. VPWMTHRS, VHIGH and VCOOLTHRS are determined by the settings *TPWMTHRS*, *THIGH* and *TCOOLTHRS*. The velocity is described by the time interval *TSTEP* between each two step pulses. This allows determination of the velocity when an external step source is used. *TSTEP* always becomes normalized to 256 microstepping. This way, the thresholds do not have to be adapted when the microstep resolution is changed. The thresholds represent the same motor velocity, independent of the microstep settings. *TSTEP* becomes compared to these threshold values. A hysteresis of 1/16 *TSTEP* resp. 1/32 *TSTEP* is applied to avoid continuous toggling of the comparison results when a jitter in the *TSTEP* measurement occurs. The upper switching velocity is higher by 1/16, resp. 1/32 of the value set as threshold. The stealthChop threshold *TPWMTHRS* is not shown. It can be included with VPWMTHRS < VCOOLTHRS. The motor current can be programmed to a run and a hold level, dependent on the standstill flag *stst*.

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

Parameter	Description	Setting	Comment
stst	This flag indicates motor stand still in each operation mode. This occurs 2^20 clocks after the last step pulse.	0/1	Status bit, read only
TPOWER DOWN	This is the delay time after stand still (*stst*) of the motor to motor current power down. Time range is about 0 to 4 seconds.	*0…255*	Time in multiples of 2^18 t_{CLK}
TSTEP	Actual measured time between two 1/256 microsteps derived from the step input frequency in units of 1/fCLK. Measured value is (2^20)-1 in case of overflow or stand still.	0… 1048575	Status register, read only. Actual measured step time in multiple of t_{CLK}
TPWMTHRS	*TSTEP* ≥ *TPWMTHRS* - stealthChop PWM mode is enabled, if configured - dcStep is disabled	0… 1048575	Setting to control the upper velocity threshold for operation in stealthChop
TCOOLTHRS	*TCOOLTHRS* ≥ *TSTEP* ≥ *THIGH*: - coolStep is enabled, if configured - stealthChop voltage PWM mode is disabled *TCOOLTHRS* ≥ *TSTEP* - Stop on stall and stall output signal is enabled, if configured	0… 1048575	Setting to control the lower velocity threshold for operation with coolStep and stallGuard
THIGH	*TSTEP* ≤ *THIGH*: - coolStep is disabled (motor runs with normal current scale) - stealthChop voltage PWM mode is disabled - If *vhighchm* is set, the chopper switches to *chm*=1 with *TFD*=0 (constant off time with slow decay, only). - chopSync2 is switched off (*SYNC*=0) - If *vhighfs* is set, the motor operates in fullstep mode and the stall detection becomes switched over to dcStep stall detection.	0… 1048575	Setting to control the upper threshold for operation with coolStep and stallGuard as well as optional high velocity step mode
small_ hysteresis	Hysteresis for step frequency comparison based on *TSTEP* (lower velocity threshold) and (*TSTEP**15/16)-1 respectively (*TSTEP**31/32)-1 (upper velocity threshold)	0 1	Hysteresis is 1/16 Hysteresis is 1/32
vhighfs	This bit enables switching to fullstep, when *VHIGH* is exceeded. Switching takes place only at 45° position. The fullstep target current uses the current value from the microstep table at the 45° position.	0 1	No switch to fullstep Fullstep at high velocities
vhighchm	This bit enables switching to *chm*=1 and *fd*=0, when *VHIGH* is exceeded. This way, a higher velocity can be achieved. Can be combined with *vhighfs*=1. If set, the *TOFF* setting automatically becomes doubled during high velocity operation in order to avoid doubling of the chopper frequency.	0 1	No change of chopper mode Classic const. Toff chopper at high velocities
en_pwm_ mode	stealthChop voltage PWM enable flag (depending on velocity thresholds). Switch from off to on state while in stand still, only.	0 1	No stealthChop StealthChop below VPWMTHRS

12 Driver Diagnostic Flags

The TMC2130 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.

12.1 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. The 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.

12.2 Short to GND Protection

The TMC2130 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. In order 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.

12.3 Open Load Diagnostics

Interrupted cables are a common cause for systems failing, e.g. when connectors are not firmly plugged. The TMC2130 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.

In order to safely detect an interrupted coil connection, read out the open load flags at 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.

14 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 14.1. 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.

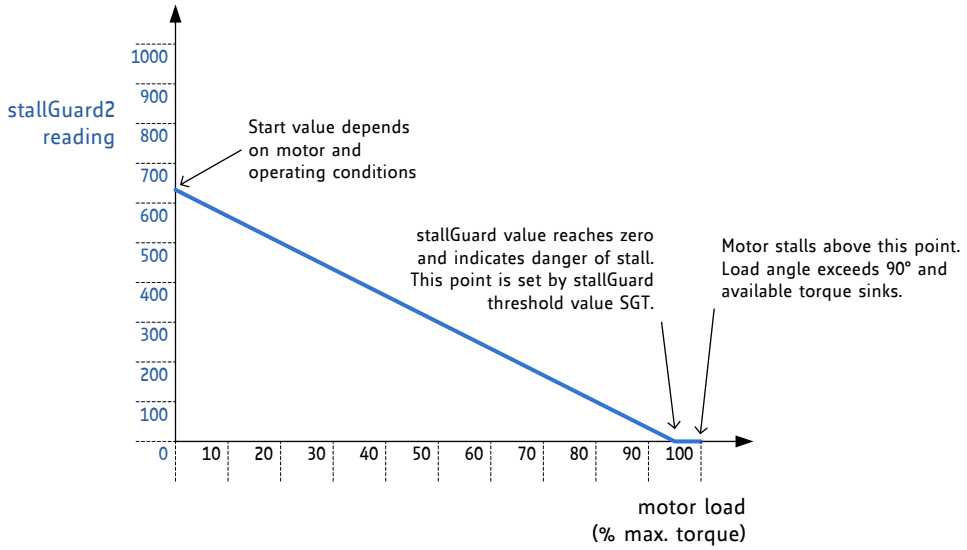


Figure 14.1 Function principle of stallGuard2

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!

14.1 Tuning the 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 monitor the stallGuard output signal via DIAG0 or DIAG1 output (configure properly) and stop the motor when a pulse is seen on the respective output. Make sure, that the motor is safely stopped whenever it is stalled. Increase *SGT* if the motor becomes stopped before a stall occurs.
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 sg_filt. 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 stall signal is activated. The external motion controller should react to a single pulse by stopping the motor if desired. Set *TCOOLSTEP* to match the lower velocity threshold where stallGuard delivers a good result.

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.

Note
Application Note 002 *Parameterization of stallGuard2 & coolStep* is available on www.trinamic.com.

14.1.1 Variable Velocity Limits *TCOOLTHRS* and *THIGH*

The *SGT* setting chosen as a result of the previously described *SGT* tuning (chapter 0) 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.

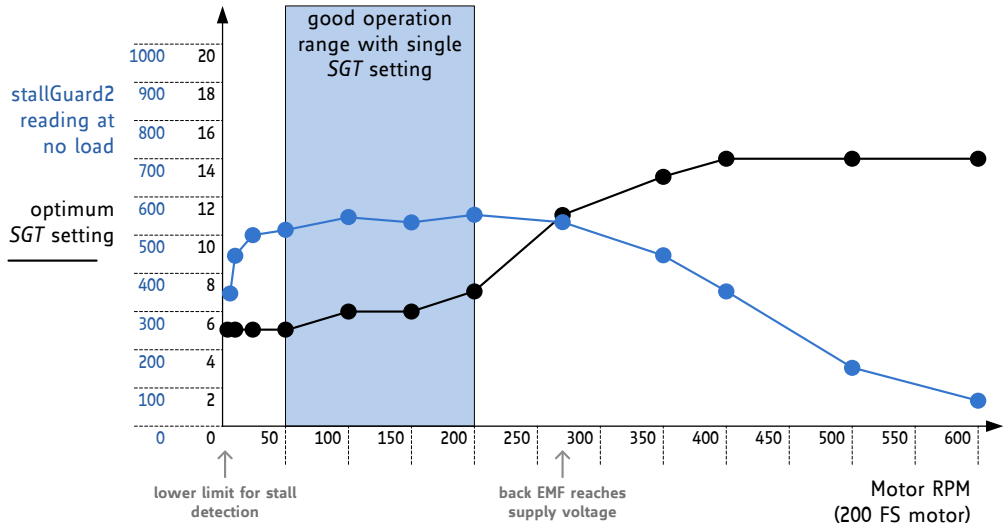


Figure 14.2 Example: optimum SGT setting and stallGuard2 reading with an example motor

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

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

14.1.2 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.

14.1.3 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.

14.1.4 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:

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

14.2 stallGuard2 Update Rate and Filtering

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.

14.3 Detecting a Motor Stall

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 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.

14.4 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.

15 coolStep Operation

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

15.1 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.

15.2 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 15.1 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.

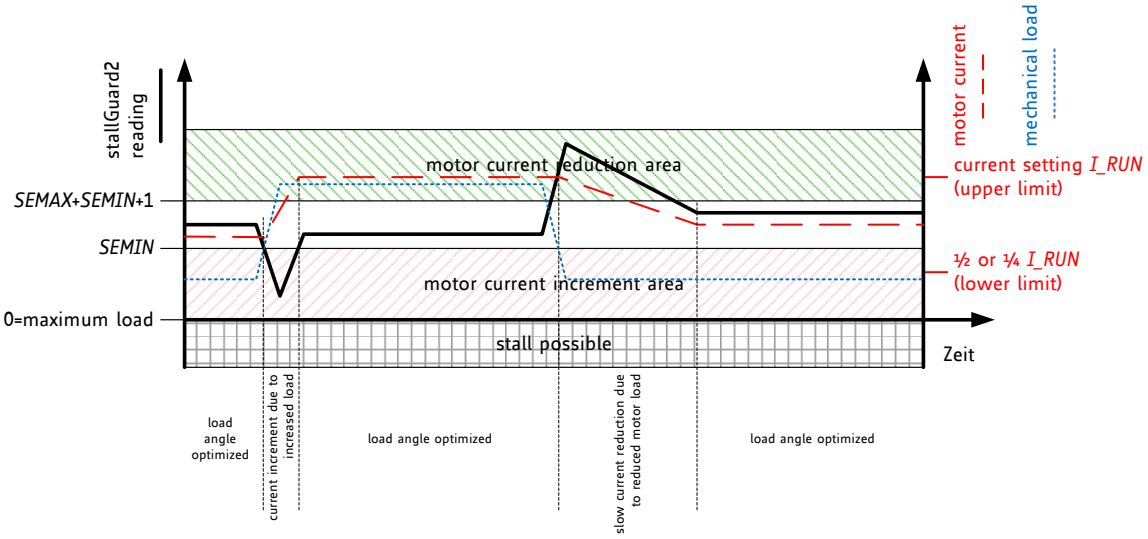


Figure 15.1 coolStep adapts motor current to the load

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	0: 1/2 of IRUN
		1	1: 1/4 of IRUN
TCOOLTHRS	Lower velocity threshold for switching on coolStep and stop on stall. 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... 2^20-1	Specifies lower coolStep velocity by comparing the threshold value to *TSTEP*
THIGH	Upper velocity threshold value for coolStep and stop on stall. Above this velocity coolStep becomes disabled. Adapt to the velocity range where stallGuard2 gives a stable result.	1... 2^20-1	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

15.3 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.

15.3.1 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.

15.3.2 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 for enabling coolStep. 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.

16 STEP/DIR Interface

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

16.1 Timing

Figure 16.1 shows the timing parameters for the STEP and DIR signals, and the table below gives their specifications. When the DEDGE mode bit in the DRVCTRL register is set, both edges of STEP are active. If DEDGE is cleared, only rising edges are active. STEP and DIR are sampled and synchronized to the system clock. An internal analog filter removes glitches on the signals, such as those caused by long PCB traces. If the signal source is far from the chip, and especially if the signals are carried on cables, the signals should be filtered or differentially transmitted.

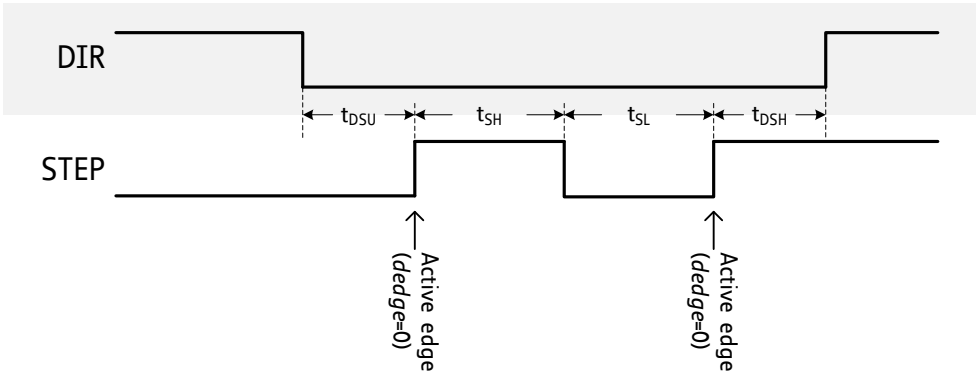


Figure 16.1 STEP and DIR timing

STEP and DIR interface timing	**AC-Characteristics** clock period is t_{CLK}					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
step frequency (at maximum microstep resolution)	f_{STEP}	*dedge*=0			½ f_{CLK}	
		dedge=1			¼ f_{CLK}	
fullstep frequency	f_{FS}				f_{CLK}/512	
STEP input low time	t_{SL}		max(t_{FILTSD}, t_{CLK}+20)			ns
STEP input high time	t_{SH}		max(t_{FILTSD}, t_{CLK}+20)			ns
DIR to STEP setup time	t_{DSU}		20			ns
DIR after STEP hold time	t_{DSH}		20			ns
STEP and DIR spike filtering time	t_{FILTSD}	rising and falling edge	36	60	85	ns
STEP and DIR sampling relative to rising CLK input	$t_{SDCLKHI}$	before rising edge of CLK input		t_{FILTSD}		ns

16.2 Changing Resolution

Sometimes operation of a motor in reduced microstep resolution is desired, in order to stay compatible to an older, less performing driver, or, when using motion controllers with limited frequency capabilities for the STEP/DIR interface. 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 DIR input, the microstep counter is increased (DIR=0) or decreased (DIR=1) with each 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 sequencer 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.

The sequencer has special provision to allow seamless switching between different microstep rates. When the microstep resolution becomes switched to a lower resolution, it calculates the nearest step within the target resolution and reads the current vector at that position. This behavior is especially important for low resolutions like fullstep and halfstep, because any failure in the step sequence would lead to asymmetrical run when comparing a motor running clockwise and counterclockwise.

Generally, different microstep resolutions are realized by stepping through the internal 256 entry microstep table in more coarse steps. In 256 microstep resolution, 1024 steps are done for a full electrical revolution using an increment of one. The increment is higher for lower resolutions, up to 256 for fullstep. When a lower resolution, each calculated table pointer becomes modified as follows, in order to point to the nearest valid microstep table address in the target resolution:

Fullstep: The first valid table position is 128 (45° electrical position, i.e. both coils on identical current). This value is the RMS-Value of 0.7 * sine wave amplitude. Step size is 256 (90° electrical)

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

Quarter step: The first valid table position is 32 (i.e. 90°/8=11.25° electrical), Step size is 64 (22.5° steps)

etc.

This way equidistant steps result and they are identical in both rotation directions. Some older drivers also use zero current (table entry 0, 0°) as well as full current (90°) within the step tables. This kind of stepping is avoided because it provides less torque and has a worse power dissipation in driver and motor. Especially for full stepping the condition one coil at maximum current and one coil off should be avoided, because in this condition only one coil contributes to the motion at each point of time.

Step position	*MSCNT* value	current coil A	current coil B
Half step 0	0	0%	100%
Full step 0	128	70.7%	70.7%
Half step 1	256	100%	0%
Full step 1	384	70.7%	-70.7%
Half step 2	512	0%	-100%
Full step 2	640	-70.7%	-70.7%
Half step 3	768	-100%	0%
Full step 3	896	-70.7%	70.7%

16.3 microPlyer Step Interpolator and Stand Still Detection

For each active edge on STEP, microPlyer produces microsteps at 256x resolution, as shown in Figure 16.2. It interpolates the time in between of two step impulses at the step input based on the last step interval. This way, from 2 microsteps (128 microstep to 256 microstep interpolation) up to 256 microsteps (full step input to 256 microsteps) are driven for a single step pulse.

Enable microPlyer by setting the *intpol* bit in the *CHOPCONF* register.

The step rate for the interpolated 2 to 256 microsteps is determined by measuring the time interval of the previous step period and dividing it into up to 256 equal parts. The maximum time between two microsteps corresponds to 2^{20} (roughly one million system clock cycles), for an even distribution of 256 microsteps. At 16 MHz system clock frequency, this results in a minimum step input frequency of 16 Hz for microPlyer operation. A lower step rate causes the *STST* bit to be set, which indicates a standstill event. At that frequency, microsteps occur at a rate of (system clock frequency)/2^{16} ~ 256 Hz. When a stand still is detected, the driver automatically switches the motor to holding current *IHOLD*.

Attention
microPlyer only works perfectly with a stable STEP frequency. Do not use the *dedge* option if the STEP signal does not have a 50% duty cycle.

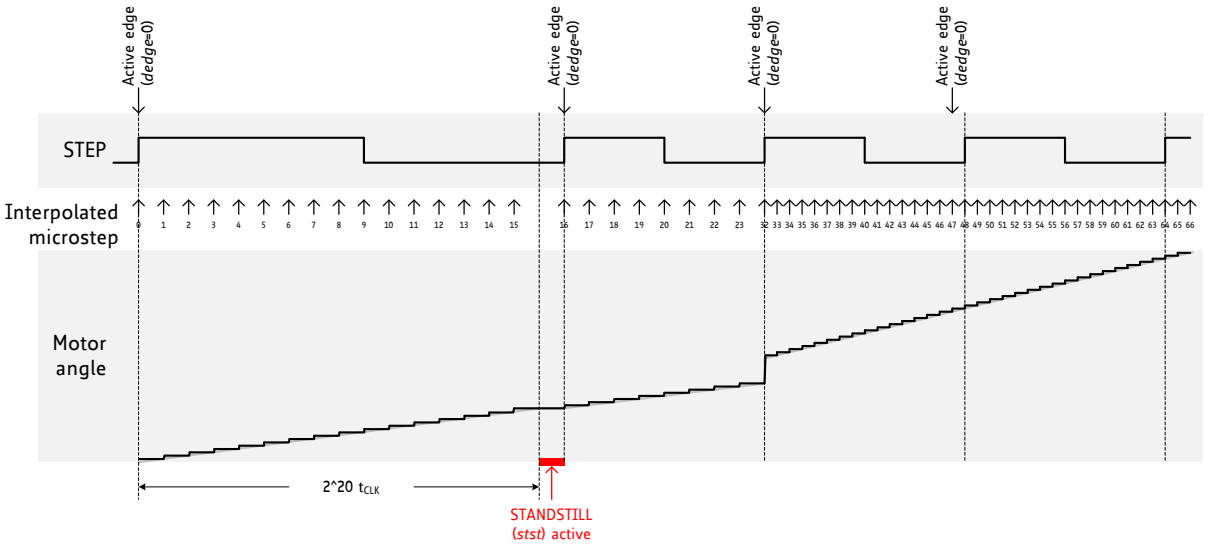


Figure 16.2 microPlyer microstep interpolation with rising STEP frequency (Example: 16 to 256)

In Figure 16.2, the first STEP cycle is long enough to set the standstill bit *stst*. This bit is cleared on the next STEP active edge. Then, the external STEP frequency increases. After one cycle at the higher rate microPlyer adapts the interpolated microstep rate to the higher frequency. During the last cycle at the slower rate, microPlyer did not generate all 16 microsteps, so there is a small jump in motor angle between the first and second cycles at the higher rate.

17 DIAG Outputs

Operation with a motion controller often requires quick reaction to certain states of the stepper motor driver. Therefore, the DIAG outputs supply a configurable set of different real time information complementing the STEP/DIR interface.

Both, the information available at DIAG0 and DIAG1 can be selected as well as the type of output (low active open drain – default setting, or high active push-pull). In order to determine a reset of the driver, DIAG0 always shows a power-on reset condition by pulling low during a reset condition. Figure 17.1 shows the available signals and control bits.

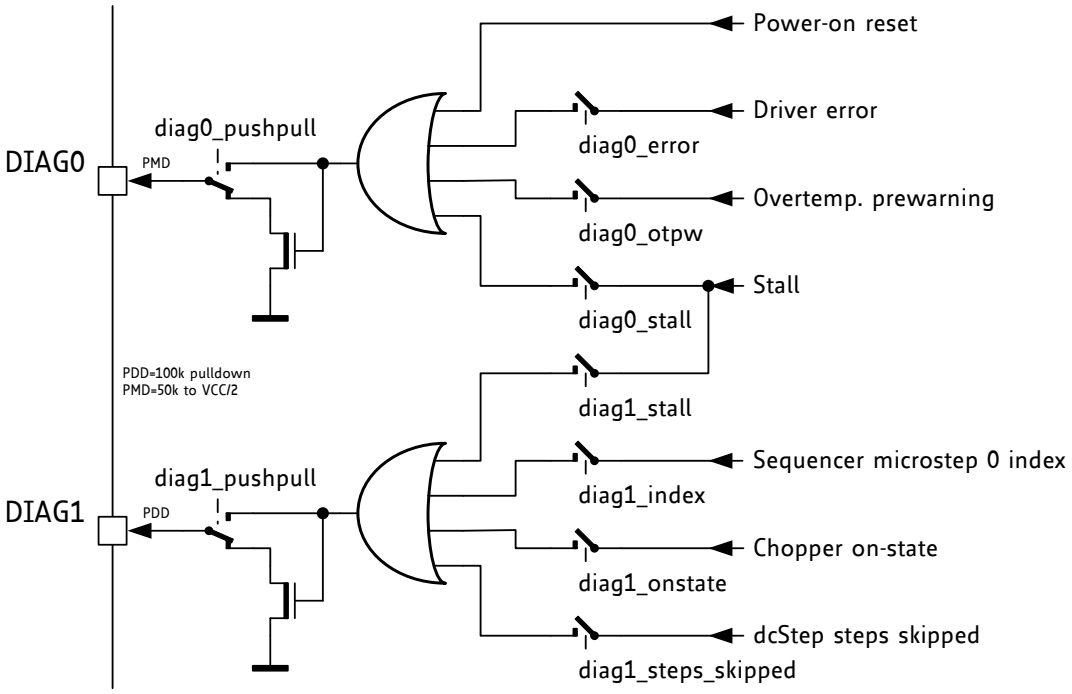


Figure 17.1 DIAG outputs in STEP/DIR mode

The stall output signal allows stallGuard2 to be handled by the external motion controller like a stop switch. The index output signals the microstep counter zero position, to allow the application to reference the drive to a certain current pattern. Chopper on-state shows the on-state of both coil choppers (alternating) when working in spreadCycle or constant off time in order to determine the duty cycle. The dcStep skipped information is an alternative way to find out when dcStep runs with a velocity below the step velocity. It toggles with each step not taken by the sequencer.

18 dcStep

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

18.1 User Benefits

dcStep™

Motor	–	never loses steps
Application	–	works as fast as possible
Acceleration	–	automatically as high as possible
Energy efficiency	–	highest at speed limit
Cheaper motor	–	does the job!

18.2 Designing-In dcStep

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

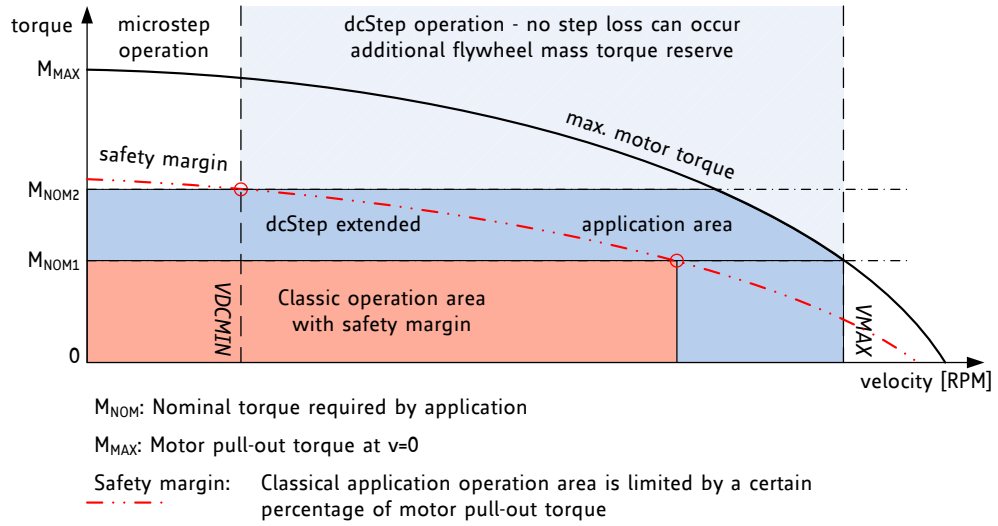


Figure 18.1 dcStep extended application operation area

Quick Start
For detail configuration procedure see Application Note AN003 - *dcStep*

dcStep requires only a few settings. It feeds back motor motion to the external ramp generator, so that it becomes seamlessly integrated into the motion ramp, even if the motor becomes overloaded with respect to the target velocity. dcStep operates the motor in fullstep mode at the target velocity or at reduced velocity if the motor becomes overloaded. It requires enforcing a minimum operation velocity either by the ramp generator or by *VDCMIN*. It shall be set to the lowest operating velocity where dcStep gives a reliable detection of motor operation. The motor never stalls unless it becomes braked to a velocity below *VDCMIN*. In case the velocity should fall below this value, the motor would restart once its load is released, unless the stall detection is used to stop the motor in this case. Stall detection is covered by stallGuard2.

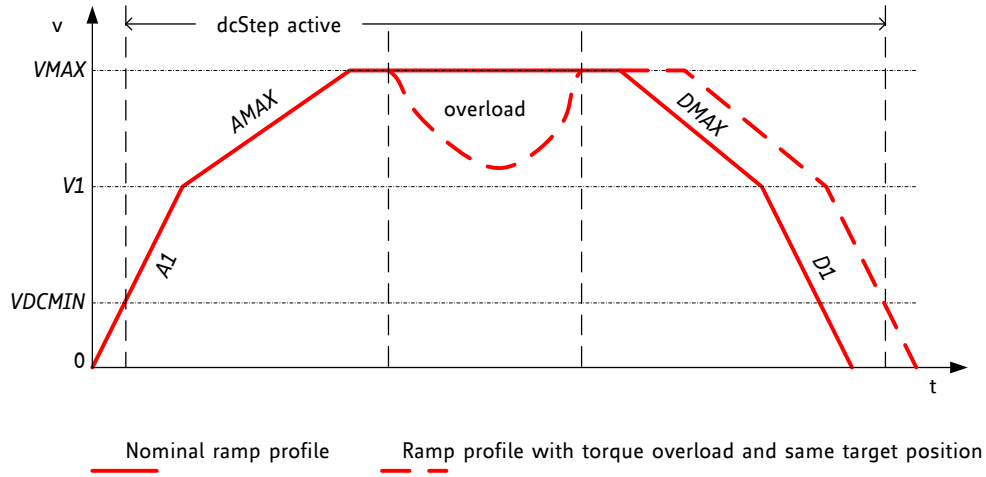


Figure 18.2 Velocity profile with impact by overload situation (example)

Attention
dcStep requires that the phase polarity of the sine wave is positive within the *MSCNT* range 768 to 255 and negative within 256 to 767. The cosine polarity must be positive from 0 to 511 and negative from 512 to 1023. A phase shift by 1 would disturb dcStep operation. Therefore it is advised to work with the default wave. Please refer chapter 19.2 for an initialization with the default table.

18.3 dcStep with STEP/DIR Interface

The TMC2130 provides two ways to use dcStep when interfaced to an external motion controller. The first way gives direct control of the dcStep step execution to the external motion controller, which must react to motor overload and is allowed to override a blocked motor situation. The second way assumes that the external motion controller cannot directly react to dcStep signals. The TMC2130 automatically reduces the motor velocity or stops the motor upon overload. In order to allow the motion controller to react to the reduced real motor velocity in this mode, the counter *LOST_STEPS* gives the number of steps which have been commanded, but not taken by the motor controller. The motion controller can later on read out *LOST_STEPS* and drive any missing number of steps. In case of a blocked motor it tries moving it with the minimum velocity as programmed by *VDCMIN*.

Enabling dcStep automatically sets the chopper to constant TOFF mode with slow decay only. This way, no re-configuration is required when switching from microstepping mode to dcStep and back.

dcStep operation in STEP/DIR mode is controlled by three pins:

- DCEN – Forces the driver to dcStep operation if high. A velocity based activation of dcStep is controlled by *TPWMTHRS* when using stealthChop operation for low velocity settings. In this case, dcStep is disabled while in stealthChop mode, i.e. at velocities below the stealthChop switching velocity.

- DCO – Informs the motion controller when motor is not ready to take a new step (low level). The motion controller shall react by delaying the next step until DCO becomes high. The sequencer can buffer up to the effective number of microsteps per fullstep to allow the motion controller to react to assertion of DCO. In case the motor is blocked this wait situation can be terminated after a timeout by providing a long > 1024 clock STEP input, or via the internal *VDCMIN* setting.
- DCIN – Commands the driver to wait with step execution and to disable DCO. This input can be used for synchronization of multiple drivers operating with dcStep.

18.3.1 Using *LOST_STEPS* for dcStep Operation

This is the simplest possibility to integrate dcStep with a dedicated motion controller: the motion controller enables dcStep using DCEN or the internal velocity threshold. The TMC2130 tries to follow the steps. In case it needs to slow down the motor, it counts the difference between incoming steps on the STEP signal and steps going to the motor. The motion controller can read out the difference and compensate for the difference after the motion or on a cyclic basis. Figure 18.3 shows the principle (simplified).

In case the motor driver needs to postpone steps due to detection of a mechanical overload in dcStep, and the motion controller does not react to this by pausing the step generation, *LOST_STEPS* becomes incremented or decremented (depending on the direction set by DIR) with each step which is not taken. This way, the number of lost steps can be read out and executed later on or be appended to the motion. As the driver needs to slow down the motor while the overload situation persists, the application will benefit from a high microstepping resolution, because it allows more seamless acceleration or deceleration in dcStep operation. In case the application is completely blocked, *VDCMIN* sets a lower limit to the step execution. If the motor velocity falls below this limit, however an unknown number of steps is lost and the motor position is not exactly known any more. DCIN allows for step synchronization of two drivers: it stops the execution of steps if low and sets DCO low.

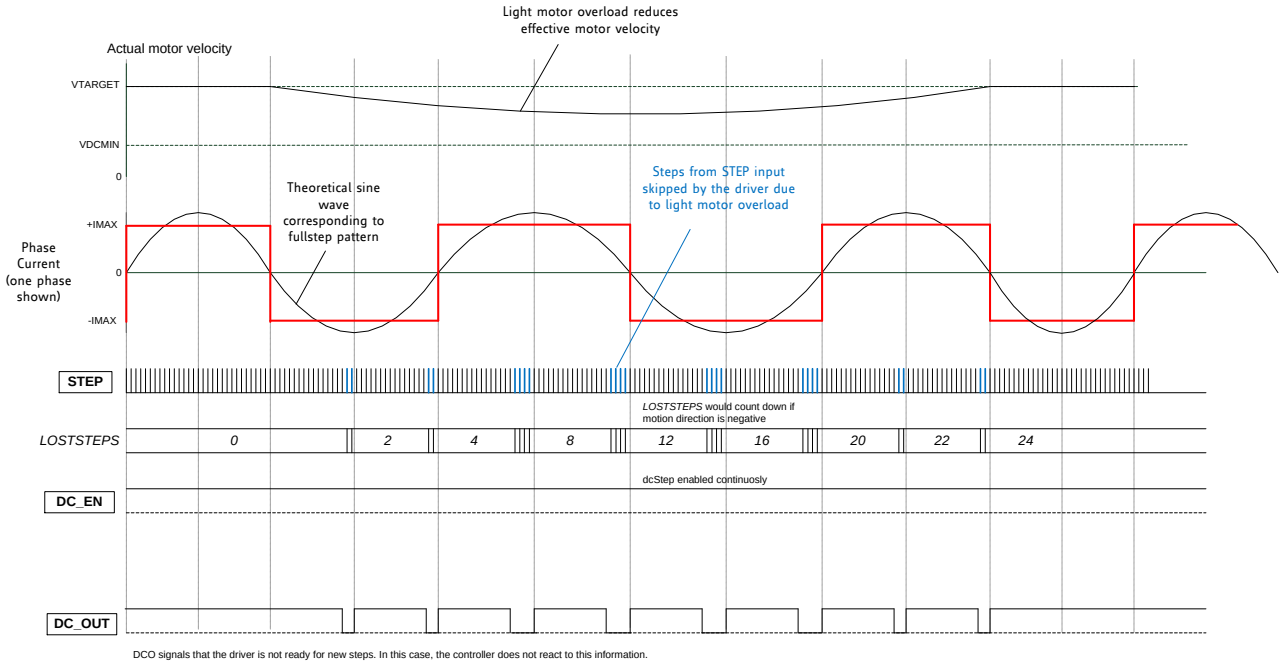


Figure 18.3 Motor moving slower than STEP input due to light overload. LOSTSTEPS incremented

18.3.2 DCO Interface to Motion Controller

DCEN enables dcStep. It is up to the connected motion controller to enable dcStep either, once a minimum step velocity is exceeded within the motion ramp, or to use the automatic threshold *VDCMIN* for dcStep enable.

The STEP/DIR interface works in microstep resolution, even if the internal step execution is based on fullstep. This way, no switching to a different mode of operation is required within the motion controller. The dcStep output DCO signals if the motor is ready for the next step based on the dcStep measurement of the motor. If the motor has not yet mechanically taken the last step, this step cannot be executed, and the driver stops automatically before execution of the next fullstep. This situation is signaled by DCO. The external motion controller shall stop step generation if DCOUT is low and wait until it becomes high again. Figure 18.5 shows this principle. The driver buffers steps during the waiting period up to the number of microstep setting minus one. In case, DCOUT does not go high within the lower step limit time e.g. due to a severe motor overload, a step can be enforced: override the stop status by a long STEP pulse with min. 1024 system clocks length. When using internal clock, a pulse length of minimum 125µs is recommended.

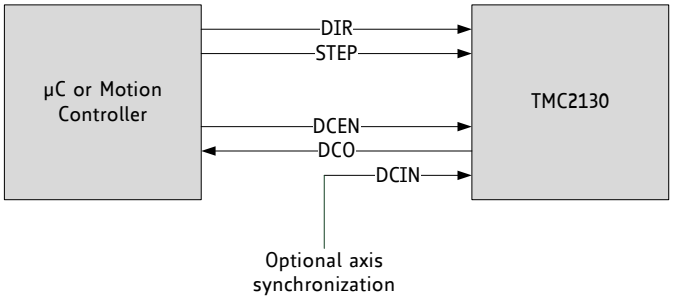


Figure 18.4 Full signal interconnection for dcStep

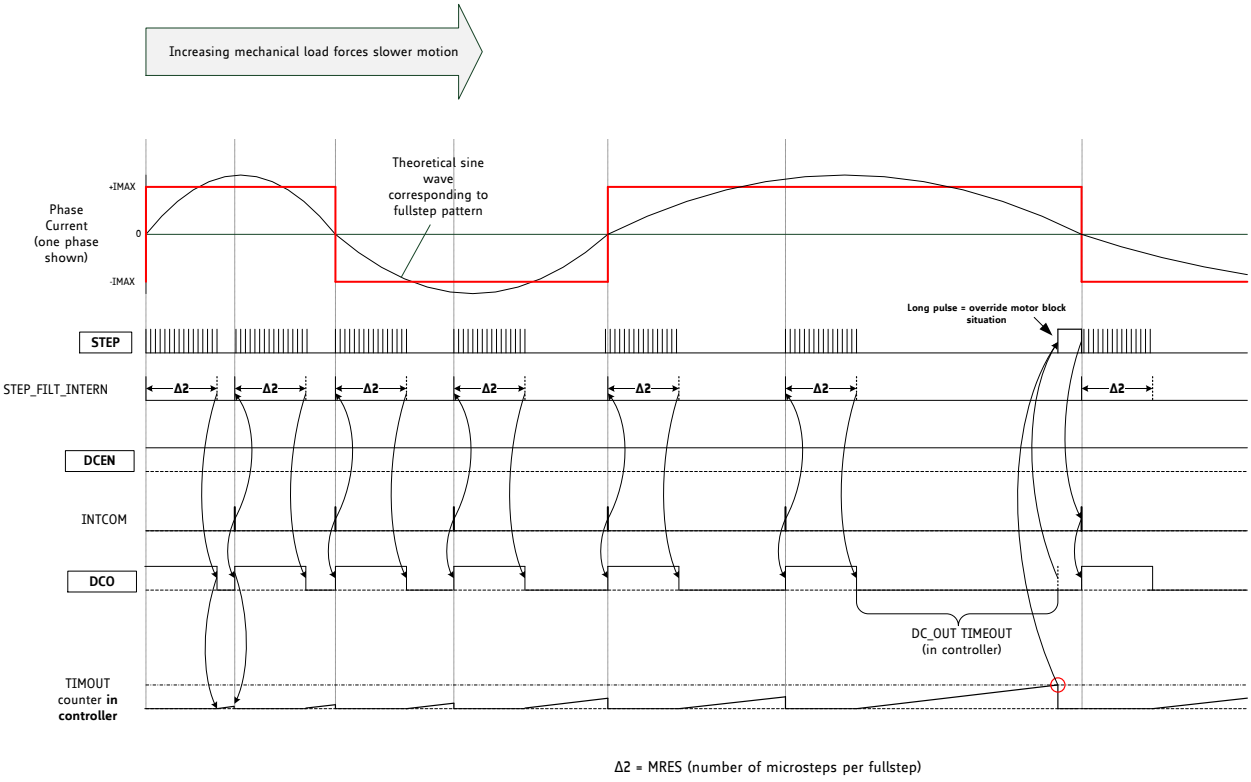


Figure 18.5 DCO Interface to motion controller – step generator stops when DCO is asserted

18.4 Stall Detection in dcStep Mode

While dcStep is able to decelerate the motor upon overload, it cannot avoid a stall in every operation situation. Once the motor is blocked, or it becomes decelerated below a motor dependent minimum velocity where the motor operation cannot safely be detected any more, the motor may stall and loose steps. In order to safely detect a step loss and avoid restarting of the motor, monitor the stall output signal for stall detection. A stallGuard2 load value also is available during dcStep operation. The range of values is limited to 0 to 255, in certain situations up to 511 will be read out. In order to enable stallGuard, also set *TCOOLTHRS* corresponding to a velocity slightly above *VDCMIN* or up to *VMAX*.

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

Parameter	Description	Range	Comment
vhighfs & *vhighchm*	These chopper configuration flags in *CHOPCONF* need to be set for dcStep operation. As soon as *VDCMIN* becomes exceeded, the chopper becomes switched to fullstepping.	0 / 1	set to 1 for dcStep
TOFF	dcStep often benefits from an increased off time value in *CHOPCONF*. Settings >2 should be preferred.	2… 15	Settings 8…15 do not make any difference to setting 8 for dcStep operation.
VDCMIN	In case the external motion controller cannot provide the lower dcStep velocity, this register may be used to enforce start/restart of a blocked motor. In dcStep operation, the motor operates at minimum *VDCMIN* even when it is completely blocked. Tune together with *DC_TIME* setting. Activation of stealthChop also disables dcStep.	0… 2^22	0: Disable Set to the low velocity limit for dcStep operation if desired.
DC_TIME	This setting controls the reference pulse width for dcStep. It needs to be set to a value higher than the effective blank time set by *TBL*. Check best setting under nominal operation conditions, and re-check under extreme operating conditions (e.g. lowest operation supply voltage, highest motor temperature, and highest supply voltage, lowest motor temperature).	0… 255	t_{BLANK} (as defined by *TBL*) in clock cycles + *n* with *n* in the range 1 to 100 (for a typical motor)
DC_SG	This setting controls stall detection in dcStep mode. A stall can be used as an error condition by issuing a hard stop for the motor. Check best setting under nominal operation conditions, and re-check under extreme operating conditions (e.g. lowest operation supply voltage, highest motor temperature, and highest supply voltage, lowest motor temperature). The stall detection is available as a pulse on DIAG0 or DIAG1 output.	0… 255	Set slightly higher than *DC_TIME*/16

19 Sine-Wave Look-up Table

The TMC2130 provides a programmable look-up table for storing the microstep current wave. 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.

19.1 User Benefits

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

19.2 Microstep Table

In order 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 a negative inclination can be realized. The four inclination segments are controlled by the position registers *X1* to *X3*. Inclination segment 0 goes from microstep position 0 to *X1*-1 and its base inclination is controlled by *W0*, segment 1 goes from *X1* to *X2*-1 with its base inclination controlled by *W1*, etc.

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

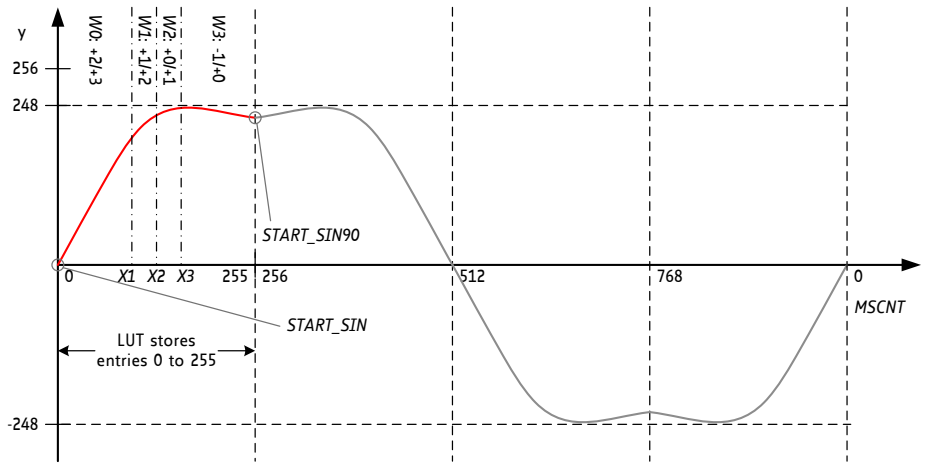


Figure 19.1 LUT programming example

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 5.5 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 TMC2130-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]= %00010000000010000010000010000010 = 0x10104222
MSLUT[4]= %11111011111111111111111111111111 = 0xFBFFFFFF
MSLUT[5]= %10110101101110110111011101111101 = 0xB5BB777D
MSLUT[6]= %01001001001010010101010101010110 = 0x49295556
MSLUT[7]= %00000000010000000100001000100010 = 0x00404222

MSLUTSEL= 0xFFFF8056:
X1=255, *X2*=255, *X3*=128
W3=%01, *W2*=%01, *W1*=%01, *W0*=%10

MSLUTSTART= 0x00F70000:
START_SIN_0= 0, *START_SIN90*= 247

20 Emergency Stop

The driver provides a negative active enable pin ENN to safely switch off all power MOSFETs. This allows putting the motor into freewheeling. Further, it is a safe hardware function whenever an emergency stop not coupled to software is required. Some applications may require the driver to be put into a state with active holding current or with a passive braking mode. This is possible by programming the pin DCIN to act as a step disable function. Set GCONF flag *stop_enable* to activate this option. Whenever DCIN becomes pulled low, the motor will stop abruptly and go to the power down state, as configured via *IHOLD*, *IHOLD_DELAY* and stealthChop standstill options. Please be aware, that disabling the driver via ENN will require three clock cycles to safely switch off the driver. In case the external CLK fails, it is not safe to disable ENN. In this case, the driver should be reset, i.e. by switching off VCC_IO.

21 DC Motor or Solenoid Operation

The TMC2130 can drive one or two DC motors using one coil output per DC motor. For DC motor operation, either a torque limited operation, or a voltage based velocity control with torque limit is possible. In order to operate DC motors, set *direct_mode* active. The motors' direction and torque become controlled by register *XDIRECT* (0x2D). Specify signed motor A current (bits 8..0) and motor B current (bits 24..16). Additionally, in this mode, the current is scaled by *IHOLD* setting. The STEP/DIR inputs and the motion controller are not used in this mode.

Using spread cycle chopper, the motor will operate in a torque limited mode. The upper motor velocity is only limited by the supply voltage (or by the load on the motor). In order to operate the motor at different velocities, use the voltage PWM mode. In this mode the motor behaves like in a DC voltage driven mode using a 4-quadrant power supply. In voltage PWM mode, the current setting directly influences motor voltage, and thus controls the motor velocity. If you want to additionally take advantage of the motor current limitation (and thus torque controlled operation) in this mode, you can use automatic current scaling. In case two DC motors are driven in voltage PWM mode, note that the automatic voltage PWM current regulation will work only for the motor which has the higher current setting. The PWM of the slower motor also will be scaled down in case the faster motor is heavier loaded. In case stand-still freewheeling is desired, you can switch on the freewheeling function.

21.1 Solenoid Operation

The same way, one or two solenoids (i.e. magnetic coils switching for example valves, etc.) can be operated by a single chip. For solenoids, it is often desired to have an increased current for a short time after switching on, and reduce the current once the magnetic element has switched. This is automatically possible by taking advantage of the automatic current scaling (*IRUN*, *IHOLD*, *IHOLDDELAY* and *TPOWERDOWN*). The current scaling in *direct_mode* is still active, but will not be triggered if no step impulse is supplied. Therefore, a step impulse must be given to the STEP input whenever one of the coils shall be switched on. This will increase the current for both coils at the same time. A simple and signal saving means to realize this would be to tie the STEP input to the CSN signal. The drawback of this solution is that the current increase will be triggered with each SPI access.

22 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 TMC2130. 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, e.g. freewheeling and further 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

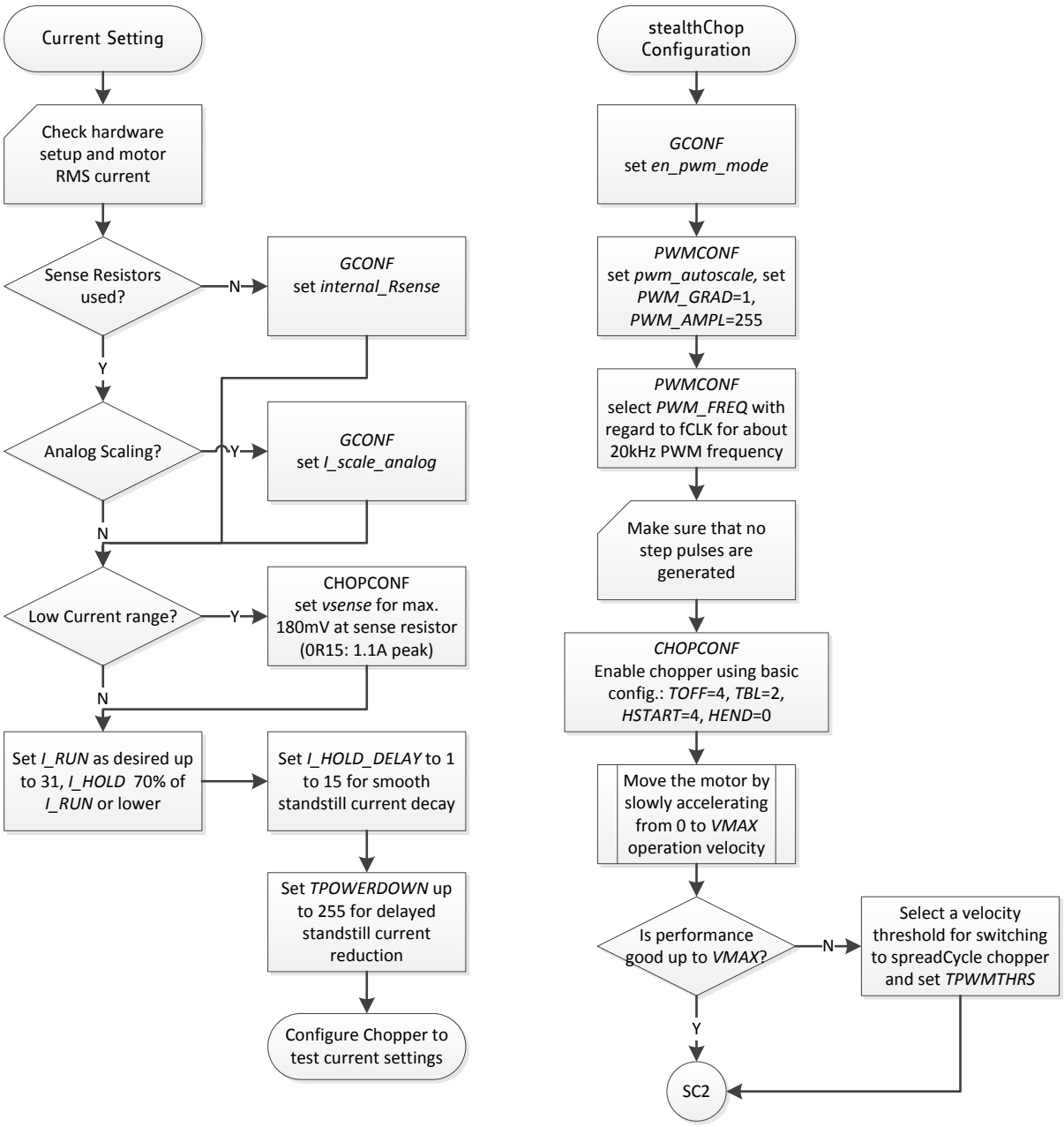


Figure 22.1 Current setting and first steps with stealthChop

TUNING STEALTHCHOP AND SPREADCYCLE

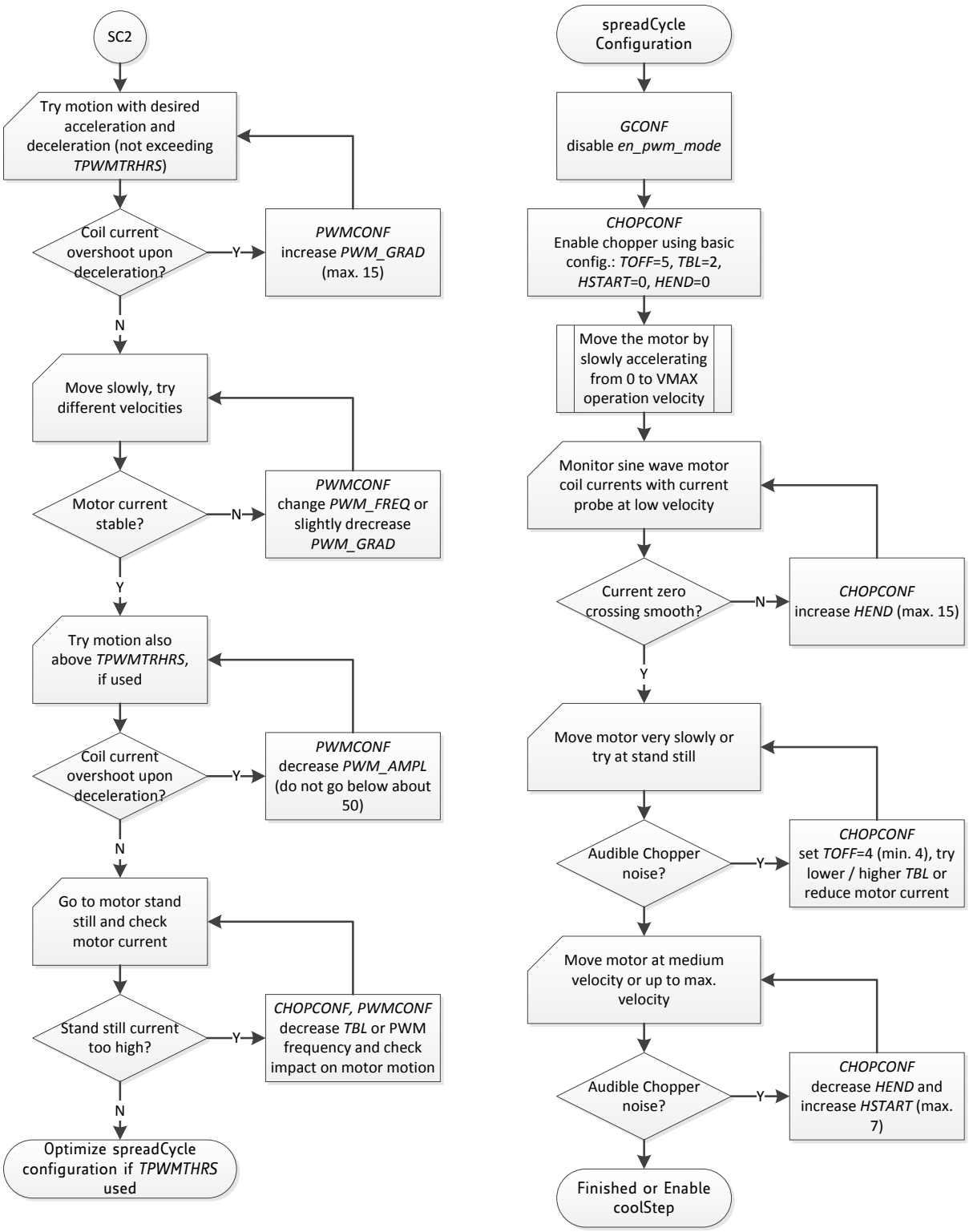


Figure 22.2 Tuning stealthChop and spreadCycle

ENABLING COOLSTEP (ONLY IN COMBINATION WITH SPREADCYCLE)

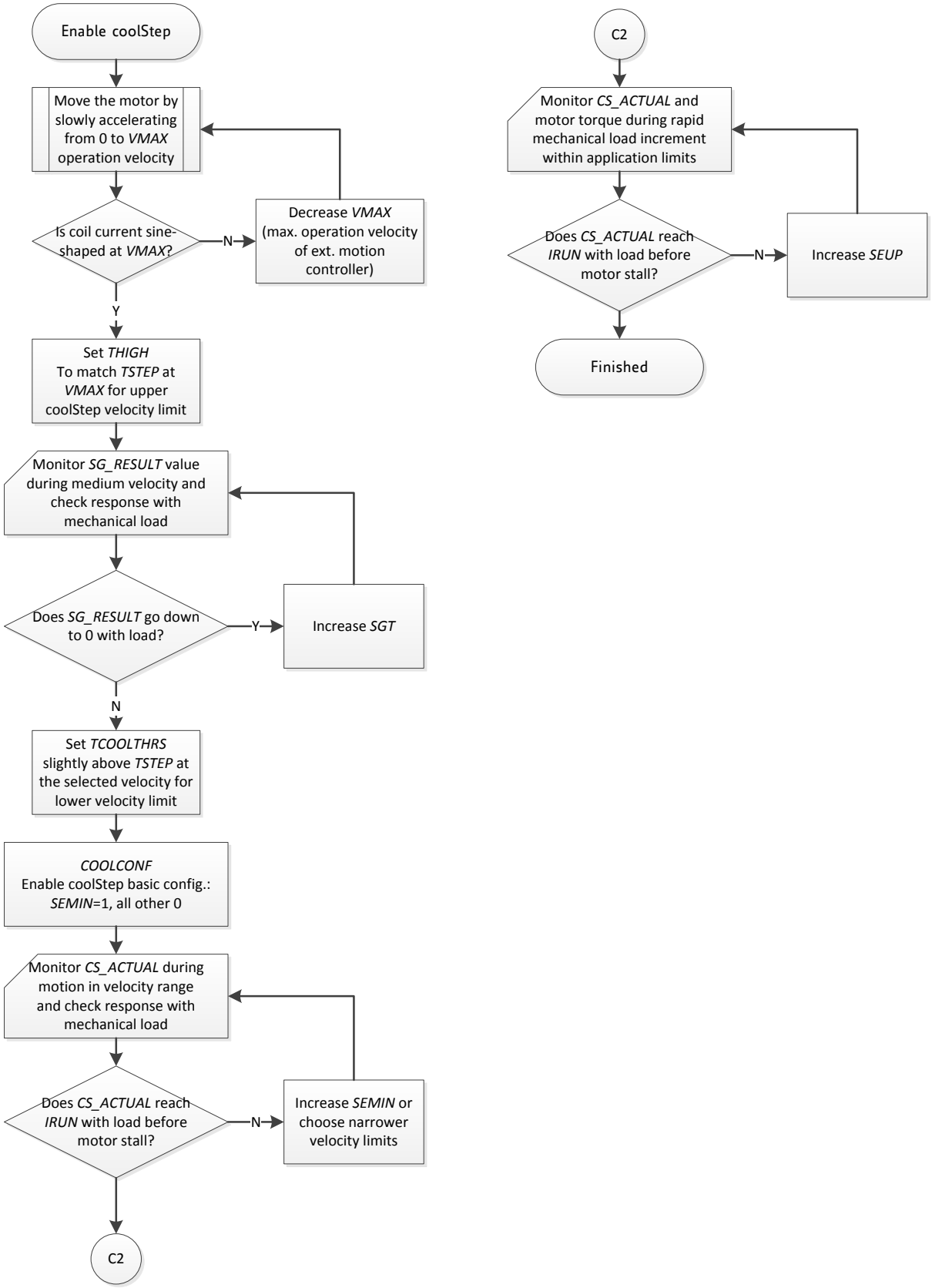


Figure 22.3 Enabling coolStep (only in combination with spreadCycle)

23 Getting Started

Please refer to the TMC2130 evaluation board to allow a quick start with the device and in order to allow interactive tuning of the device setup in your application. It will guide you through the process of correctly setting up all registers. The following example gives a minimum set of accesses allowing moving a motor.

23.1 Initialization Example

SPI datagram example sequence to enable the driver for step and direction operation and initialize the chopper for spreadCycle operation and for stealthChop at <60 RPM:

```
SPI send: 0xEC000100C5;  // CHOPCONF: TOFF=5, HSTRT=4, HEND=1, TBL=2, CHM=0 (spreadCycle)
SPI send: 0x9000061F0A;  // IHOLD_IRUN: IHOLD=10, IRUN=31 (max. current), IHOLDDELAY=6
SPI send: 0x910000000A;  // TPOWERDOWN=10: Delay before power down in stand still
SPI send: 0x8000000004;  // EN_PWM_MODE=1 enables stealthChop (with default PWM_CONF)
SPI send: 0x93000001F4;  // TPWM_THRS=500 yields a switching velocity about 35000 = ca. 30RPM
SPI send: 0xF0000401C8;  // PWM_CONF: AUTO=1, 1/1024 Fclk, Switch amplitude limit=200, Grad=1
```

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

24 Standalone Operation

For standalone operation, no SPI interface is required to configure the TMC2130. All pins with suffix CFG0 to CFG6 have a special meaning in this mode. They are evaluated using tristate detection, in order to differentiate between

- CFG pin tied to GND
- CFG pin open (no connection)
- CFG pin tied to VCC_IO

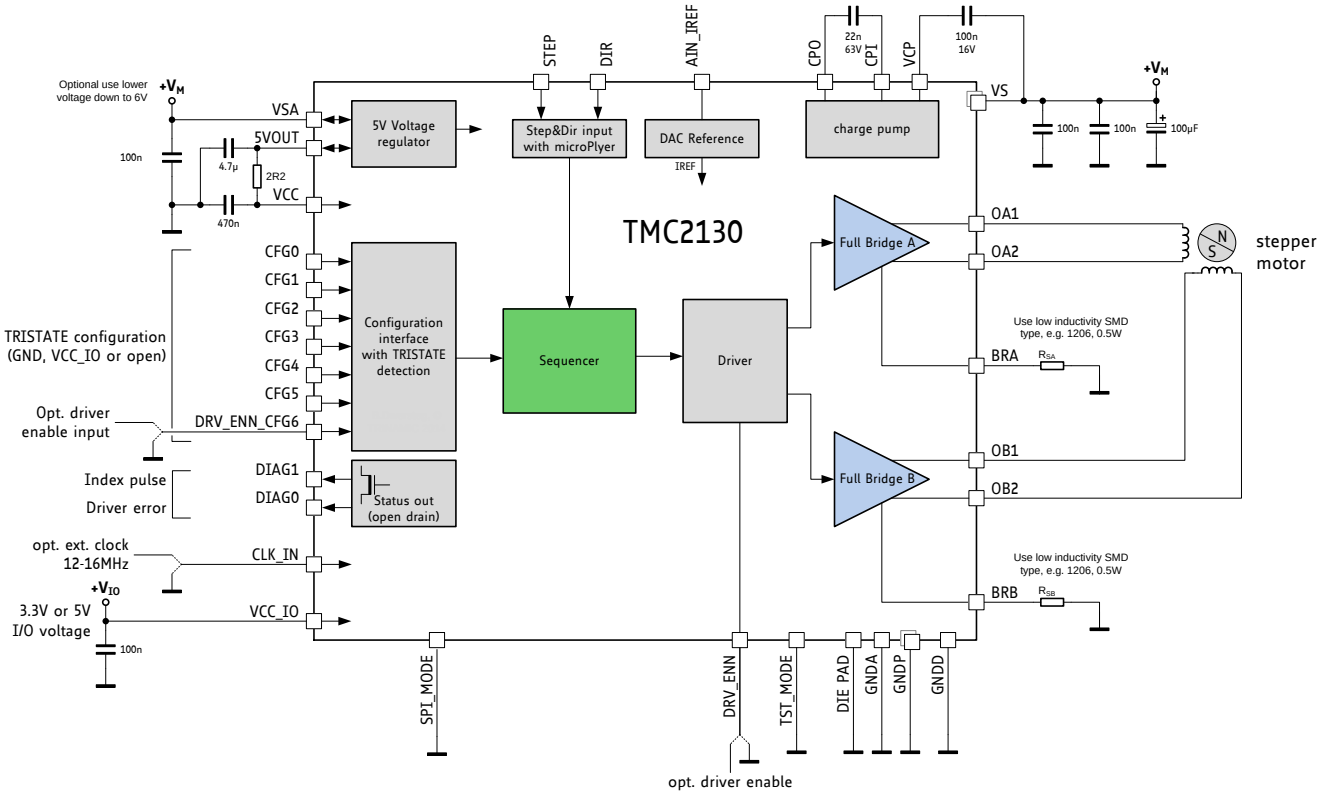


Figure 24.1 Standalone operation with TMC2130 (pins shown with their standalone mode names)

To activate standalone mode, tie pin SPI_MODE to GND. SPI is off. The driver works in spreadCycle mode or stealthChop mode. With regard to the register set, the following settings are activated:

GCONF settings:
GCONF.diag0_error = 1: DIAG0 works in open drain mode and signals driver error.
GCONF.diag1_index = 1: DIAG1 works in open drain mode and signals microstep table index position.

The following settings are affected by the CFG pins in order to ensure correct configuration:

CFG0: SETS CHOPPER OFF TIME (DURATION OF SLOW DECAY PHASE)		
CFG0	***TOFF* Setting**	**Registers**
GND	140 T_{CLK} (recommended, most universal choice)	*TOFF*=4
VCC_IO	236 T_{CLK}	*TOFF*=7
open	332 T_{CLK}	*TOFF*=10

CFG1 AND CFG2: SETS MICROSTEP RESOLUTION FOR STEP INPUT				
CFG2, CFG1	**Microsteps**	**Interpolation**	**Chopper Mode**	**Registers**
GND, GND	1 (Fullstep)	N	spreadCycle	*MRES*=8, *intpol*=0
GND, VCC_IO	2 (Halfstep)	N		*MRES*=7, *intpol*=0
GND, open	2 (Halfstep)	Y, to 256 µsteps		*MRES*=7, *intpol*=1
VCC_IO, GND	4 (Quarterstep)	N		*MRES*=6, *intpol*=0
VCC_IO, VCC_IO	16 µsteps	N		*MRES*=4, *intpol*=0
VCC_IO, open	4 (Quarterstep)	Y, to 256 µsteps		*MRES*=6, *intpol*=1
open, GND	16 µsteps	Y, to 256 µsteps		*MRES*=4, *intpol*=1
open, VCC_IO	4 (Quarterstep)	Y, to 256 µsteps	stealthChop	*MRES*=6, *intpol*=1, *en_PWM_mode*=1
open, open	16 µsteps	Y, to 256 µsteps		*MRES*=4, *intpol*=1, *en_PWM_mode*=1

CFG3: SETS MODE OF CURRENT SETTING		
CFG3	**Current Setting**	**Registers**
GND	Internal reference voltage. Current scale set by sense resistors, only.	
VCC_IO	Internal sense resistors. Use analog input current on AIN as reference current for internal sense resistor. This setting gives best results when combined with stealthChop voltage PWM chopper.	*internal_Rsense=1*
open	External reference voltage on pin AIN. Current scale set by sense resistors and scaled by AIN.	*I_scale_analog*=1

CFG4: SETS CHOPPER HYSTERESIS (TUNING OF ZERO CROSSING PRECISION)		
CFG4	***HEND* Setting**	**Registers**
GND	5 (recommended, most universal choice)	*HEND*=7
VCC_IO	9	*HEND*=11
open	13	*HEND*=15

CFG5: SETS CHOPPER BLANK TIME (DURATION OF BLANKING OF SWITCHING SPIKE)		
CFG5	**Blank time (in number of clock cycles)**	**Registers**
GND	16	*TBL*=%00
VCC_IO	24 (recommended, most universal choice)	*TBL*=%01
open	36	*TBL*=%10

CFG6_ENN: ENABLE PIN AND CONFIGURATION OF STANDSTILL POWER DOWN			
CFG6	**Motor driver enable**	**Standstill power down**	**Registers**
GND	Enable	N	*IRUN*=31, *IHOLD*=31
VCC_IO	Disable	- (Driver disable)	
open	Enable	Y, ramp down from 100% to 34% motor current in 44M clock cycles (3 to 4 seconds) if no step pulse for more than 1M clock cycles (standstill). In combination with stealthChop, be sure not to work with too low overall current setting, as regulation will not be able to measure the motor current after stand still current reduction. This will result in very low motor current after the stand-still period.	*IRUN*=31, *IHOLD*=11, *IHOLDDELAY*=8

While the parameters for spreadCycle can be configured for good microstep performance, stealthChop mode is configured with its power on default values (*PWMCONF*=0x00050480):

f_{PWM}=1/683 f_{CLK} (i.e. roughly 19kHz with internal clock)
pwm_autoscale=1
PWM_GRAD=4
PWM_AMPL=128

CFG0 and CFG4 settings do not influence the stealthChop configuration. This way, it is even possible to switch between spreadCycle and stealthChop mode by simply switching CFG1 and CFG2.

Hint
Be sure to allow the motor to rest for at least 100ms (assuming a minimum of 10MHz f_{CLK}) before starting a motion using stealthChop. This will allow the current regulation to set the initial motor current.

Example:
It is desired to do small motions in smooth and noiseless stealthChop mode. For quick motions, spreadCycle is to be used. The controller can deliver 1/16 microstep step signals. Tie together CFG1 and CFG2 and drive them with a three state driver. Switch both to VCC_IO to operate in spreadCycle, switch them to hi-Z (open) state for a motion in stealthChop.

25 External Reset

The chip is loaded with default values during power on via its internal power-on reset. In order to reset the chip to power on defaults, any of the supply voltages monitored by internal reset circuitry (VSA, +5VOUT or VCC_IO) must be cycled. VCC is not monitored. Therefore VCC must not be switched off during operation of the chip. As +5VOUT is the output of the internal voltage regulator, it cannot be cycled via an external source except by cycling VSA. It is easiest and safest to cycle VCC_IO in order to completely reset the chip. Also, 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, all inputs must be driven low during this reset operation. When this is not possible, an input protection resistor may be used to limit current flowing into the related inputs.

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.

26 Clock Oscillator and Clock Input

The clock is the timing reference for all functions: the chopper and the velocity thresholds. 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.

USING THE INTERNAL CLOCK

Directly tie the CLK input to GND near to the IC if the internal clock oscillator is to be used. The temperature dependency and ageing of the internal clock is comparatively low.

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

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 source 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.

26.1 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 TMC2130 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 (or even lower) can be considered.

27 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	49	V
Supply and bridge voltage max. *)	V_{VMAX}		50	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}		3.0	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.

28 Electrical Characteristics

28.1 Operational Range

Parameter	Symbol	Min	Max	Unit
Junction temperature	T_J	-40	125	°C
Supply voltage (using internal +5V regulator)	V_{VS}, V_{VSA}	5.5	46	V
Supply voltage (internal +5V regulator bridged: V_{VCC}=V_{VSA}=V_{VS})	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}		1.4	A
Peak output current per motor coil output (sine wave peak) using external or internal current sensing	I_{Ox}		2.0	A
Peak output current per motor coil output (sine wave peak) for short term operation. Limit $T_J \leq 105°C$, e.g. for 100ms short time acceleration phase below 50% duty cycle.	I_{Ox}		2.5	A

28.2 DC 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} = V_{VSA} = 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		15	22	mA
Total supply current, operating, I_{VS} + I_{VSA} + I_{VCC}	I_S	f_{CLK}=16MHz, 23.4kHz chopper, no load		19		mA
Idle supply current from VS, charge pump operating	I_{VS0}	f_{CLK}=0Hz, driver disabled		0.25	0.5	mA
Static supply current from VSA	I_{VSA0}	f_{CLK}=0Hz	1.4	2	3	mA
Supply current, driver disabled, dependency on CLK frequency	I_{VSA}	f_{CLK} variable, additional to I_{VSA0}		0.8		mA/MHz
Internal current consumption from 5V supply on VCC pin	I_{VCC}	f_{CLK}=16MHz, 23.4kHz chopper		16		mA
IO supply current (typ. at 5V)	I_{VIO}	no load on outputs, inputs at V_{IO} or GND Excludes pullup / pull-down resistors		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 (resistive load)	50	120	220	ns
slope, MOSFET turning off	t_{SLPOFF}	measured at 700mA load current (resistive load)	50	120	220	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_{VCC} - 0.3	V_{VCC}	V
Charge pump voltage threshold for undervoltage detection	V_{VCP}-V_{VS}	using internal 5V regulator voltage	3.3	3.6	3.8	V
Charge pump frequency	f_{CP}			1/16 f_{CLKOSC}		

Linear regulator	**DC-Characteristics** V_{VS} = V_{VSA} = 24.0V					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Output voltage	V_{5VOUT}	I_{5VOUT} = 0mA T_J = 25°C	4.80	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} = 16mA T_J = full range		+/-30	+/-100	mV
Deviation of output voltage over the full supply voltage range	$V_{5VOUT(DEV)}$	I_{5VOUT} = 0mA V_{VSA} = variable		+/-15	+/-30	mV / 10V
Deviation of output voltage over the full supply voltage range	$V_{5VOUT(DEV)}$	I_{5VOUT} = 16mA V_{VSA} = variable		-38	+/-75	mV / 10V

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_{CLK}	CLK driven to 0.1 V_{VIO} / 0.9 V_{VIO}	10			ns

Detector levels	**DC-Characteristics**					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
V_{VSA} undervoltage threshold for RESET	V_{UV_VSA}	V_{VSA} 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 (delay typ. 10µs)	2.1	2.55	3.0	V
V_{VCC_IO} undervoltage detector hysteresis	$V_{UV\_VIOHYST}$			0.3		V
Short to GND detector threshold (V_{VS} - V_{Ox})	V_{OS2G}		2	2.5	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 f_{CLK}=16MHz**					
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		325		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 -using internal reference	I_{COIL}	*I_scale_analog*=0, *vsense*=0	-5		+5	%
Sense input tolerance / motor current full scale tolerance -using external reference voltage	I_{COIL}	*I_scale_analog*=1, V_{AIN}=2V, *vsense*=0	-2		+2	%
Internal resistance from pin BRxy to internal sense comparator (additional to sense resistor)	R_{BRxy}			20		mΩ

Digital logic levels	**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
Pullup / pull-down resistors	R_{PU}/R_{PD}		132	166	200	kΩ

AIN/IREF input	**DC-Characteristics**					
Parameter	**Symbol**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
AIN_IREF input resistance to 2.5V (=5VOUT/2)	R_{AIN}	Measured to GND (*internalRsense*=0)	260	330	400	kΩ
AIN_IREF input voltage range for linear current scaling	V_{AIN}	Measured to GND (*IscaleAnalog*=1)	0	0.5-2.4	V_{5VOUT}/2	V
AIN_IREF open input voltage level	V_{AINO}	Open circuit voltage (*internalRsense*=0)		V_{5VOUT}/2		V
AIN_IREF input resistance to GND for reference current input	R_{IREF}	Measured to GND (*internalRsense*=1)	0.8	1	1.2	kΩ
AIN_IREF current amplification for reference current to coil current at maximum setting	$I_{REFAMPL}$	I_{IREF} = 0.25mA		3000		Times
Motor current full scale tolerance -using RDSon measurement	I_{COIL}	*Internal_Rsense*=1, *vsense*=0, I_{IREF} = 0.25mA	-10		+10	%

28.3 Thermal Characteristics

The following table shall give an idea on the thermal resistance of the package. The thermal resistance for a four layer board will provide a good idea on a typical application. Actual thermal characteristics will depend on the PCB layout, PCB type and PCB size. The thermal resistance will benefit from thicker CU (inner) layers for spreading heat horizontally within the PCB. Also, air flow will reduce thermal resistance.

A thermal resistance of 24K/W for a typical board means, that the package is capable of continuously dissipating 4.1W at an ambient temperature of 25°C with the die temperature staying below 125°C.

Parameter	Symbol	Conditions	Typ	Unit
Typical power dissipation	P_D	stealthChop or spreadCycle, 0.92A RMS in two phase motor, sinewave, 20kHz chopper, 24V, internal supply, 84°C peak surface of package (motor QSH4218-035-10-027)	2.6	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, 35µm CU, 84mm x 55mm, d=1.5mm)	24	K/W
Thermal resistance junction to board	R_{TJB}	PCB temperature measured within 1mm distance to the package	8	K/W
Thermal resistance junction to case	R_{TJC}	Junction temperature to heat slug of package	3	K/W

Table 28.1 Thermal Characteristics QFN5x6

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 spread-sheet for calculating TMC2130 power dissipation is available on www.trinamic.com.

29 Layout Considerations

29.1 Exposed Die Pad

The TMC2130 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.

29.2 Wiring GND

All signals of the TMC2130 are referenced to their respective GND. Directly connect all GND pins under the device 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 TMC2130. Optimally place them close to the IC, 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.

29.3 Supply Filtering

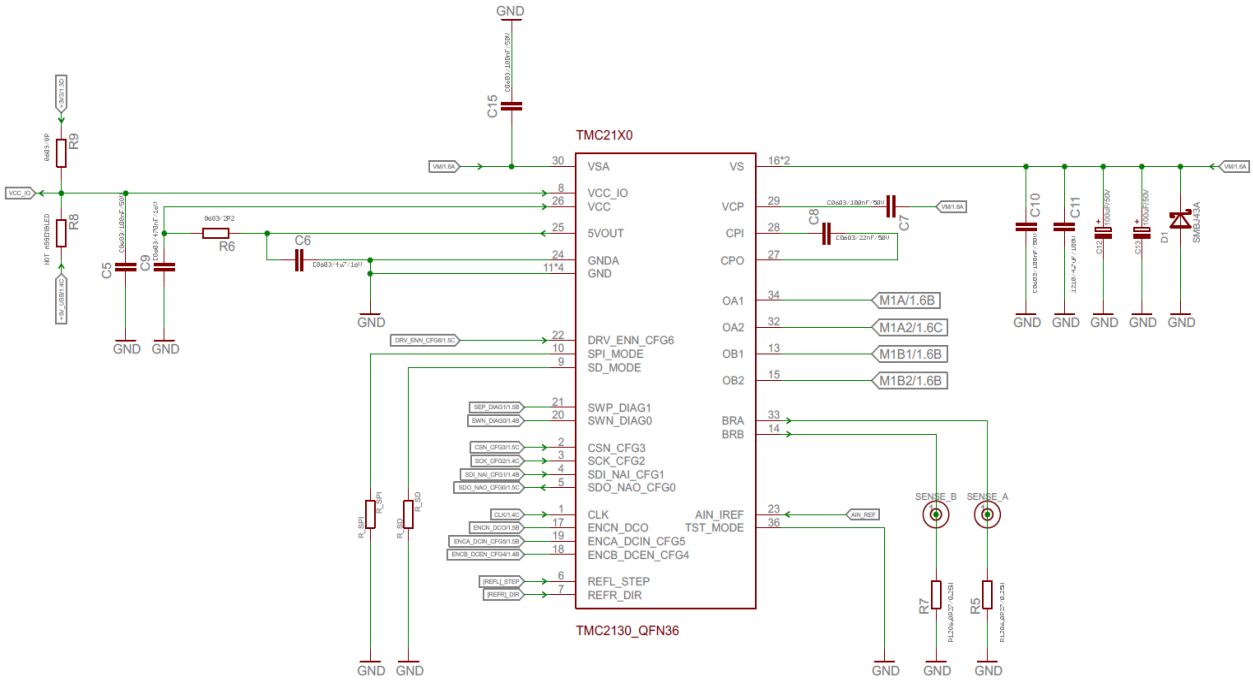
The 5VOUT output voltage ceramic filtering capacitor (4.7 µF recommended) should be placed as close as possible to the 5VOUT pin, with its GND return going directly to the GNDA pin. This ground connection shall not be shared with other loads or additional vias to the GND plan. Use as short and as thick connections as possible. For best microstepping performance and lowest chopper noise an additional filtering capacitor should 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 (1-2mm distance) to the VCC pin with GND return going to the ground plane. VCC can be coupled to 5VOUT using a 2.2 Ω or 3.3 Ω resistor in order 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.

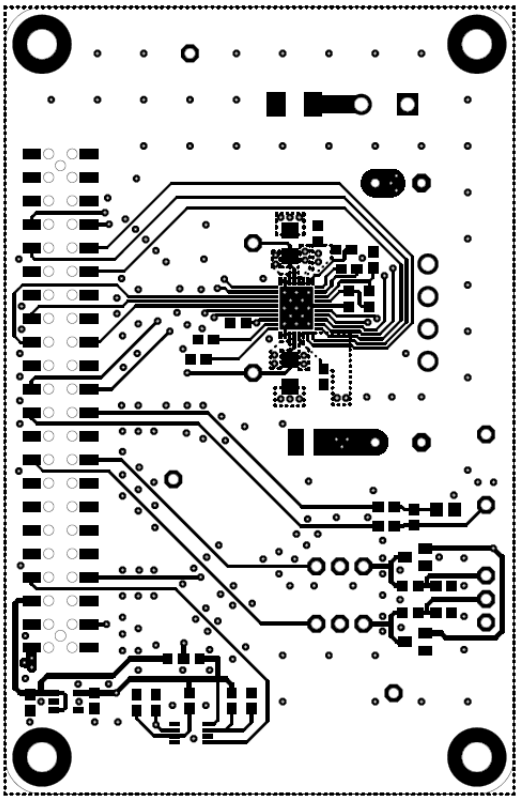
Take into account that the switching motor coil outputs have a high dV/dt. Thus capacitive stray into high resistive signals can occur, if the motor traces are near other traces over longer distances.

29.4 Layout Example

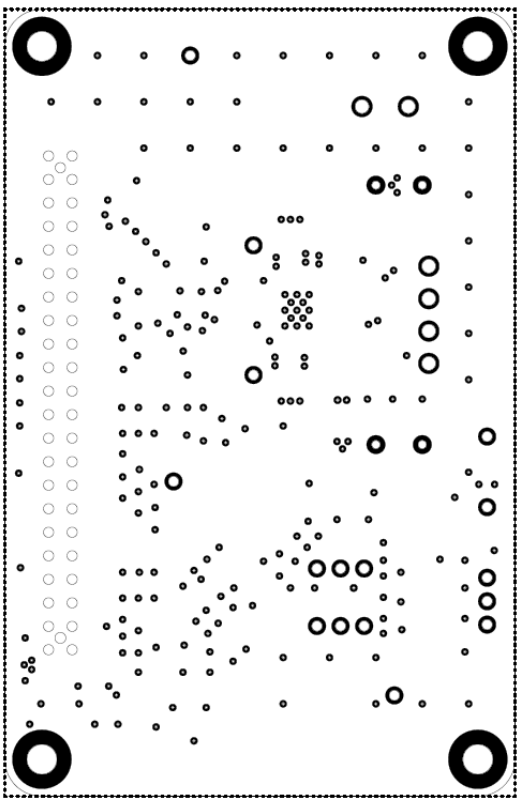
Schematic



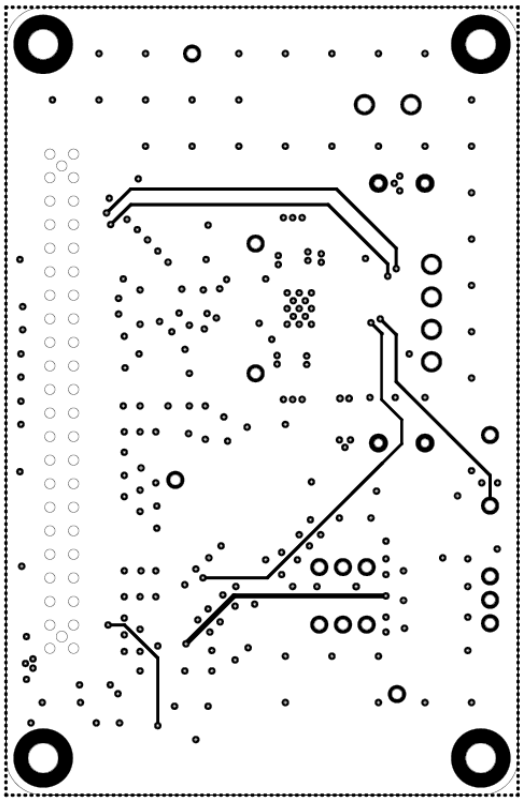
1 - Top Layer (assembly side)



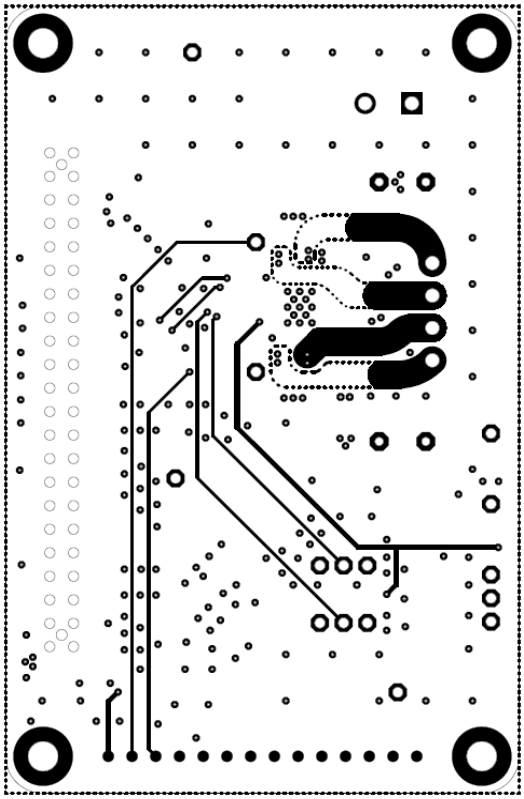
2 - Inner Layer 1



3 - Inner Layer 2



4 - Bottom Layer



Components / Silksceen Top

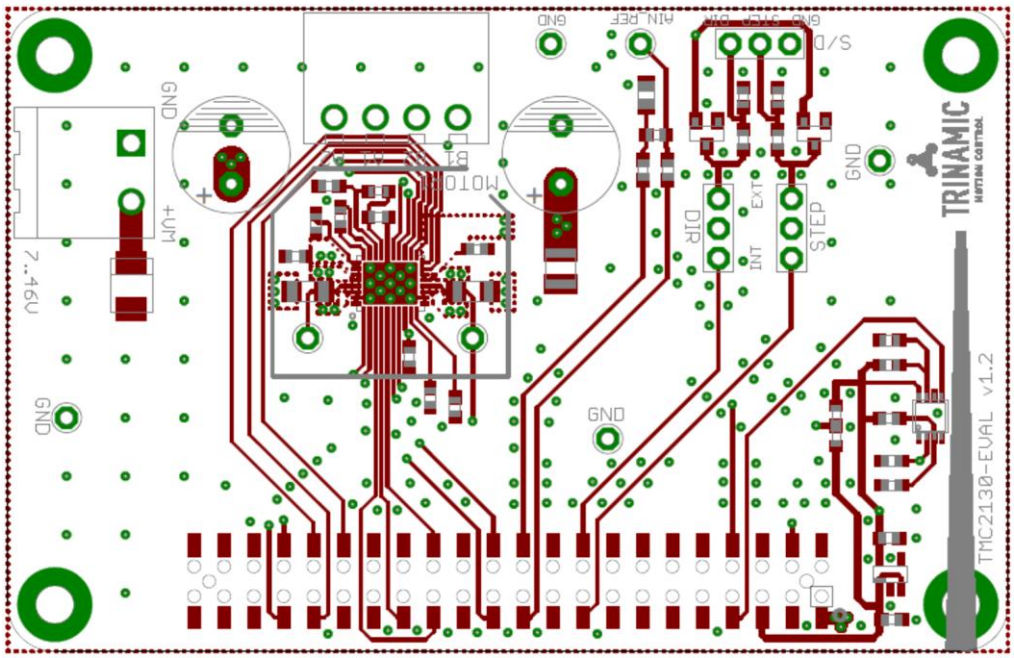


Figure 29.1 Layout example

30 Package Mechanical Data

30.1 Dimensional Drawings QFN36 5x6

Attention: Drawings not to scale.

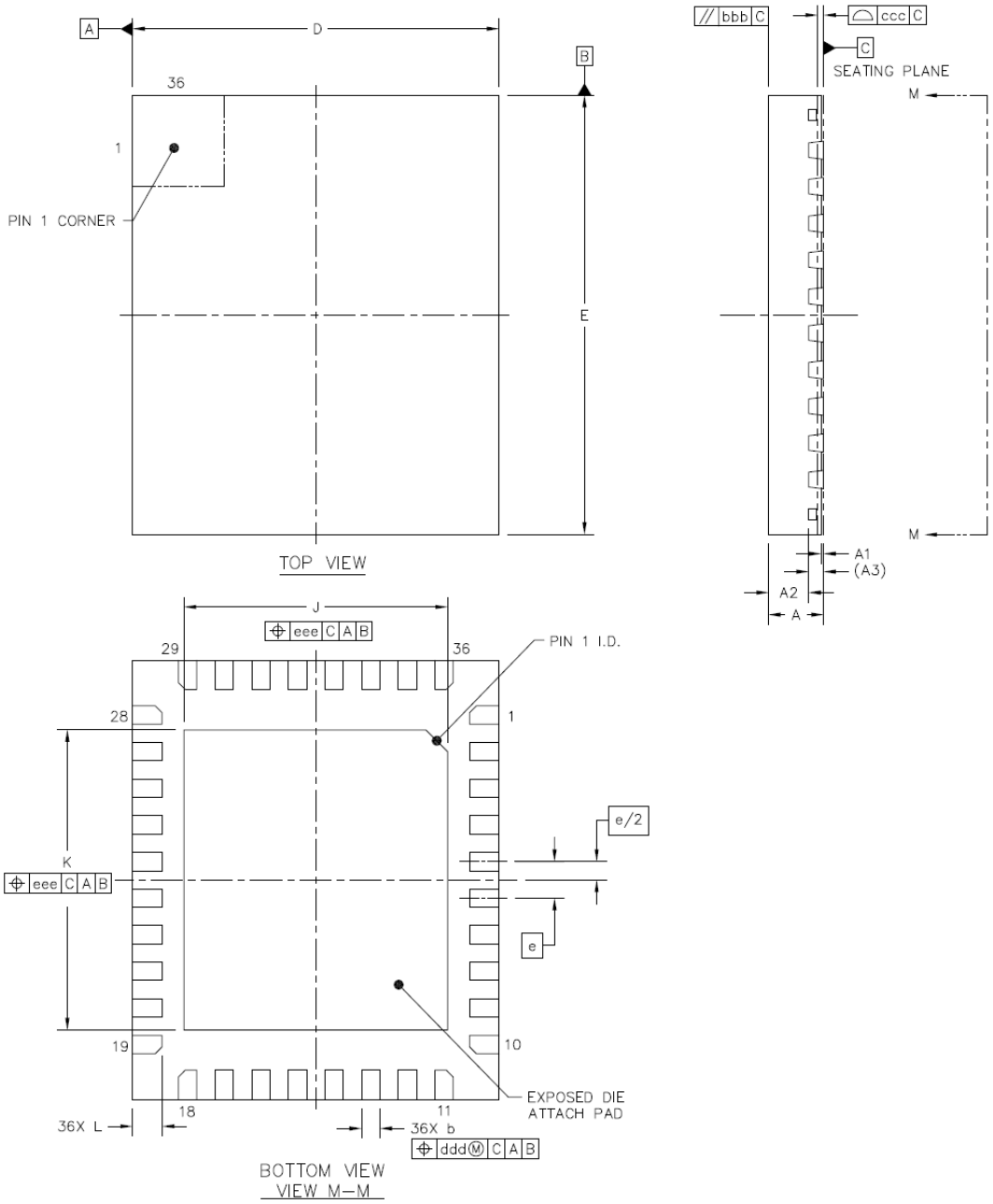


Figure 30.1 Dimensional drawings QFN 5x6

Parameter	Ref	Min	Nom	Max
total thickness	A	0.8	0.85	0.9
stand off	A1	0	0.035	0.05
mold thickness	A2	-	0.65	-
lead frame thickness	A3		0.203	
lead width	b	0.2	0.25	0.3
body size X	D	4.9	5	5.1
body size Y	E	5.9	6	6.1
lead pitch	e		0.5	
exposed die pad size X	J	3.5	3.6	3.7
exposed die pad size Y	K	4	4.1	4.2
lead length	L	0.35	0.4	0.45
mold flatness	bbb			0.1
coplanarity	ccc			0.08
lead offset	ddd			0.1
exposed pad offset	eee			0.1

30.2 Package Codes

Type	Package	Temperature range	Code & marking
TMC2130-LA	QFN36 (RoHS)	-40°C ... +125°C	TMC2130-LA

31 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.

32 ESD Sensitive Device

The TMC2130 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.

33 Table of Figures

34 Revision History

Version	Date	Author BD= Bernhard Dwersteg SD= Sonja Dwersteg	Description
V0.90	2014-AUG-21	SD	First version. Based on TMC5130 datasheet V0.42.
V0.91	2014-SEP-11	SD	Figure 1.1 and Figure 1.2 corrected.
V0.92	2014-SEP-25	BD	Clarified ramp generator source as step input, etc.
V1.00	2014-OCT-15	BD	Some detail corrections (removed wording 3 phase)
V1.01	2014-NOV-24	BD	Wording thermal shutdown, QFN36 table
V1.02	2014-DEC-08	BD	Some detail corrections for 2130 and stallGuard description
V1.03	2015-MAR-10	BD	Improved AN links, dcStep stallGuard description, blue blocks Renamed *TZEROWAIT* to *TPOWERDOWN*, Added References
V1.04	2015-APR-02	BD	More direct_mode info

Table 34.1 Document Revisions

35 References

[TMC2130-EVAL] TMC2130-EVAL Manual
[AN001] Trinamic Application Note 001 - Parameterization of spreadCycle™, www.trinamic.com
[AN002] Trinamic Application Note 002 - Parameterization of stallGuard2™ & coolStep™, www.trinamic.com
[AN003] Trinamic Application Note 003 - dcStep™, www.trinamic.com
Calculation sheet TMC5130_TMC2130_TMC2100_Calculations.xlsx