



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com

Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China



TMC429 DATASHEET

Intelligent Triple Stepper Motor Controller with Serial Peripheral Interfaces and Step/Direction
Full Compatible Successor of the TMC428

+



+

APPLICATIONS

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

+

+

FEATURES AND BENEFITS

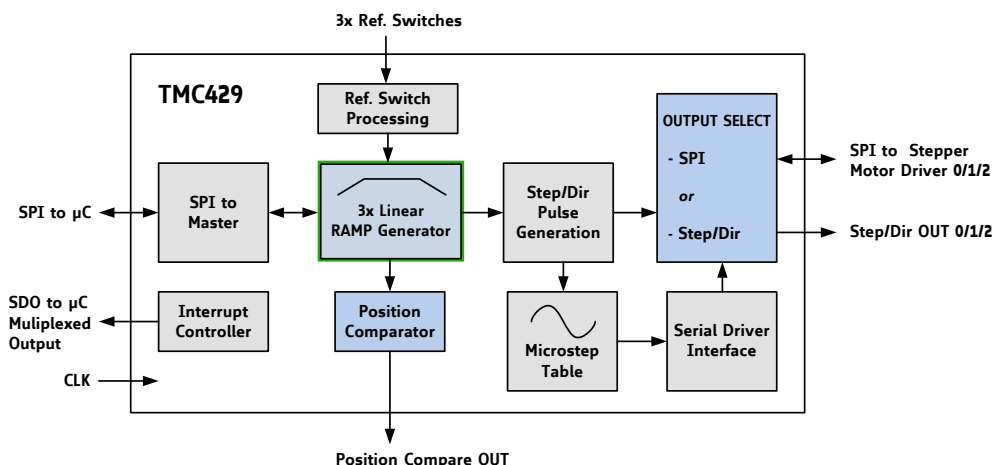
Controls up to three stepper motors
3.3 V or 5 V operation with CMOS / TTL compatible IOs
Serial 4-wire interface for μC with easy-to-use protocol
Interface for SPI™ motor drivers with data rates up to 1 Mbit/s
Step/Direction interface
Clock frequency: up to 32 MHz (can use CPU clock)
Internal position counters 24 bit wide
Microstep frequency up to 1 MHz
Read-out option for all motion parameters
Programmable 6 bit microstep table, up to 64 entries for a quarter sine wave period
Ramp generators for autonomous positioning / speed control
On-the-fly change of target motion parameters
Power boost automatic acceleration dependent current control
Low power operation: 1.25 mA at 4 MHz (typ.)
Compact Size: ultra small 16 pin SSOP package, small 24 pin SOP package, and 32 pin QFN 5x5 mm package
Directly controls TMC23x, TMC24x, TMC26x, and TMC389

DESCRIPTION

The TMC429 is a miniaturized stepper motor controller with an industry leading feature set. It controls up to three motors via SPI or Step/Direction interface. The SPI interface provides a programmable 6 bit microstep table (64 μsteps / fullstep) for best step accuracy with 2-phase stepper motors. Based on target positions and velocities - which can be altered on the fly - it performs all real time critical tasks autonomously. The TMC429 offers high level control functions for robust and reliable operation. Two separate 4 wire serial peripheral interfaces allow for communication with the microcontroller and with up to three daisy chained stepper motor drivers.

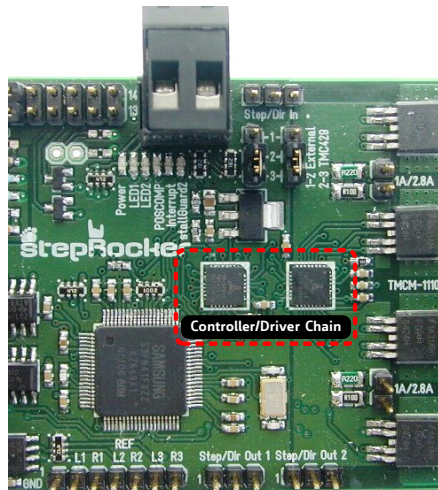
Together with a microcontroller the TMC429 forms a complete motion control system. High integration and small form factor allow for miniaturized designs for cost-effective and highly competitive solutions.

BLOCK DIAGRAM

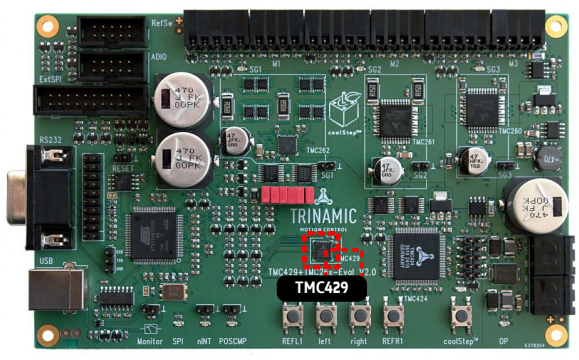


APPLICATION EXAMPLES: RELIABLE CONTROL FOR UP TO 3 MOTORS

The TMC429 scores with its autonomous handling of all real time critical tasks. By offloading the motion-control function to the TMC429, up to three motors can be operated reliably with very little demand for service from the microcontroller. Software only needs to send target positions, and the TMC429 generates precisely timed step pulses by hardware for up to three stepper motor driver chips. Parameters for each motor can be changed on the fly while software retains full control using an SPI bus. This way, high precision and reliable operation is achieved while costs are kept down.



Development platform with TMC262



Layout for Evaluation of TMC429 with TMC262, TMC261, and TMC260

STEPROCKER™

The TCM-1110 stepRocker is a single axis motor controller and driver board for 2-phase bipolar stepper motors. It features the TRINAMIC controller/driver chain consisting of TMC429 and TMC262. The Module is intended to be a fully functional development platform with 6A MOSFETs. Because of the TMC429s ability to control up to three motors the stepRocker can be extended to a full 3-axes system.

TMC429+TMC26x-EVAL

This evaluation board is a development platform for applications based on the TMC429 in combination with TMC260, TMC261, and TMC262. Common supply voltages are +12V DC / +24V DC / +48V DC (TMC261 only). The board features an embedded microcontroller with USB and RS232 interfaces. The control software provides a user-friendly GUI for setting control parameters and visualizing the dynamic responses of the motors.

Motor movements can be controlled via the step and direction interface using inputs from an external source or signals generated by the microcontroller acting as a step generator.

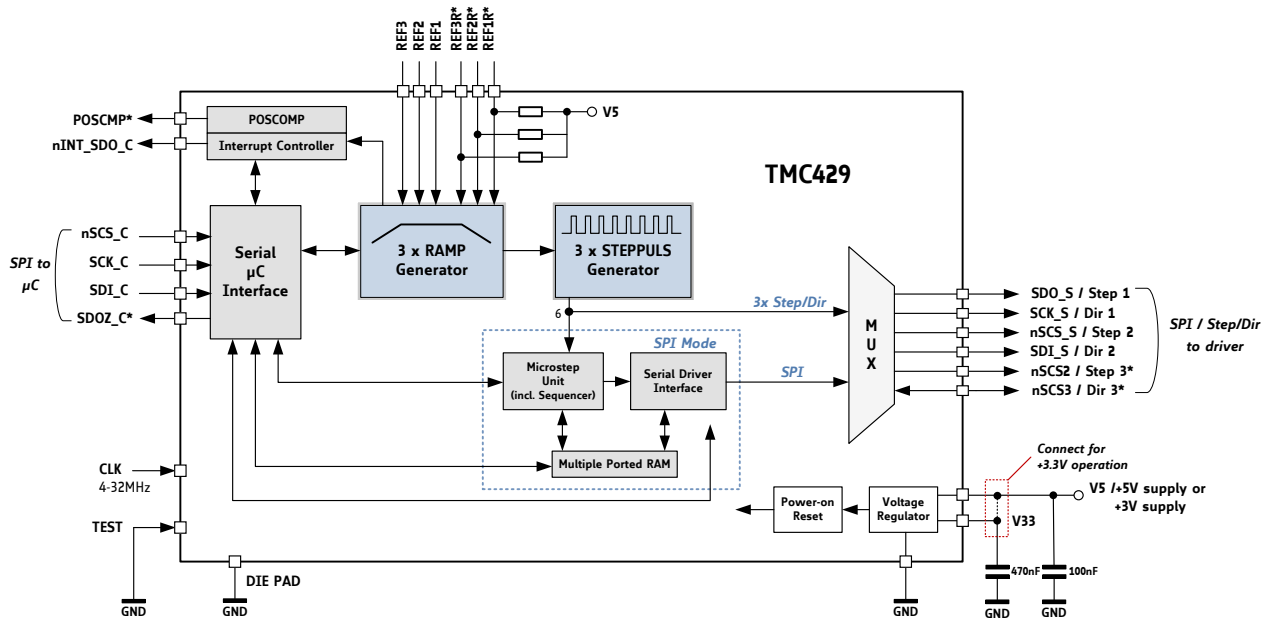
ORDER CODES

Order code	Description	Size
TMC429-LI	3-axis controller QFN32-package (5x5mm ²), full functionality	5 x 5 mm ²
TMC429-PI24	3-axis controller SOP24-package (TMC428 replacement possible)	15.5 x 10.5 mm ²
TMC429-I	3-axis controller SSOP16-package (SPI only, for TMC428 replacement)	6 x 5 mm ²
TMC429+26x-EVAL	Evaluation board for S/D chipset (TMC429with TMC260, TMC261, TMC262 and TMC424)	16 x 10 cm ²
TMC429+TMC24x-EVAL	Evaluation board for SPI chipset (TMC429, TMC246, and TMC249)	13.5 x 8,2 cm ²

TABLE OF CONTENTS

1	PRINCIPLES OF OPERATION	4	10	STEP/DIR DRIVERS	53
1.1	KEY CONCEPTS	4	10.1	TIMING	53
1.2	CONTROL INTERFACES	5	11	SPI MODE DRIVER INTERFACE	54
1.3	SOFTWARE VISIBILITY	6	11.1	BUS SIGNALS	54
1.4	STEP FREQUENCIES	6	11.2	TIMING	54
1.5	MOVING THE MOTOR	7	11.3	RAM ADDRESS PARTITIONING AND DATA ORGANIZATION	55
2	GENERAL DEFINITIONS, UNITS, AND NOTATIONS	9	11.4	STEPPER DRIVER SPI DATAGRAM CONFIGURATION	57
2.1	NOTATIONS	9	11.5	INITIALIZATION OF MICROSTEP LOOK-UP TABLE	62
2.2	SIGNAL POLARITIES	9	12	RUNNING A MOTOR	67
2.3	UNITS OF MOTION PARAMETERS	9	12.1	GETTING STARTED	67
2.4	REPRESENTATION OF SIGNED VALUES BY TWO'S COMPLEMENT	9	12.2	RUNNING A MOTOR WITH START-STOP-SPEED IN <i>RAMP_MODE</i>	67
3	PACKAGE VARIANTS	10	13	ON-CHIP VOLTAGE REGULATOR	68
4	PIN ASSIGNMENTS	10	14	POWER-ON RESET	69
4.1	PACKAGE OUTLINES	11	15	ABSOLUTE MAXIMUM RATINGS	70
4.2	SIGNAL DESCRIPTIONS	12	16	ELECTRICAL CHARACTERISTICS	70
5	SAMPLE CIRCUITS	13	16.1	POWER DISSIPATION	70
5.1	APPLICATION EXAMPLE: TMC429 IN QFN32 PACKAGE	13	16.2	DC CHARACTERISTICS	71
5.2	APPLICATION EXAMPLE: TMC429 IN SSOP16 PACKAGE	14	16.3	TIMING CHARACTERISTICS	72
5.3	APPLICATION EXAMPLE: TMC429 WITH DRIVERS WITHOUT SERIAL DATA OUTPUT (SDO)	14	18	PACKAGE MACHANICAL DATA	73
6	CONTROL INTERFACE	15	18.1	TMC429-LI / QFN32	73
6.1	BUS SIGNALS	15	18.2	TMC429-PI24 / SOP24	74
6.2	SERIAL PERIPHERAL INTERFACE FOR μ C	15	18.4	TMC429-I / SSOP16	75
7	ADDRESS SPACE PARTITIONS	20	19	MARKING	76
7.1	READ AND WRITE	20	20	COMPATIBILITY INFORMATION: TMC429 AND TMC428	77
7.2	REGISTER SET	20	20.1	SIGNAL DESCRIPTIONS: TMC428 vs. TMC429	77
7.3	REGISTER MAPPING	21	20.2	TMC428 SDO_C OUTPUT	78
8	REGISTER DESCRIPTION	22	20.3	UNUSED ADDRESSES	78
8.1	AXIS PARAMETER REGISTERS	22	20.4	GENERAL TIMING PARAMETERS	79
8.2	GLOBAL PARAMETER REGISTERS	39	21	DISCLAIMER	80
9	REFERENCE SWITCH INPUTS	49	22	ESD SENSITIVE DEVICE	80
9.1	REFERENCE SWITCH CONFIGURATION, <i>MOT1R</i> , AND <i>REFMUX</i>	49	23	TABLE OF FIGURES	81
9.2	TRIPLE SWITCH CONFIGURATION	51	24	REVISION HISTORY	82
9.3	HOMING PROCEDURE	52	25	REFERENCES	83
9.4	SIMULTANEOUS START OF UP TO THREE STEPPER MOTORS	52			

1 Principles of Operation



* Not available with all IC packages. Please refer to the package outlines.

Figure 1.1 TMC429 functional block diagram

The TMC429 is a miniaturized high performance stepper motor controller with an outstanding cost-performance ratio. It is designed for high volume automotive as well as for demanding industrial motion control applications. Once initialized the TMC429 controls up to three 2-phase stepper motors simultaneously. A programmable sequencer for 2-phase motors is integrated. The TMC429 motion controller is equipped with an SPI™ host interface with easy-to-use protocol and two driver interfaces (SPI and STEP/DIR) for addressing various stepper motor driver types.

1.1 Key Concepts

The TMC429 realizes real time critical tasks autonomously and guarantees for a robust and reliable drive. These following features contribute toward greater precision, greater efficiency, higher reliability, and smoother motion in many stepper motor applications.

- Initialization** Adapt the TMC429 to the driver type and configuration and send initial configuration data to SPI drivers. Configure microstep resolution and waveform for SPI drivers.
- Interfacing** The TMC429 offers application specific interfacing via Step/Dir or SPI.
- Positioning** The TMC429 operates the motors based on user specified target positions and velocities. Modify all motion target parameters on-the-fly during motion.
- Programming** Every parameter can be changed at any time. The uniform access to any TMC429 register simplifies application programming. A read-back option for all internal registers is available.
- Microstepping** Based on internal position counters the TMC429 performs up to $\pm 2^{23}$ (micro)steps completely independent from the microcontroller. Microstep resolutions are individually programmable for each stepper motor.
The range goes from full stepping (1 microstep = 1 full step) and half stepping (2 microsteps per full step) up to 6 bit micro stepping (64 microsteps per full step) for precise positioning and noiseless stepper motor rotation. With STEP/DIR drivers any microstep resolution is possible as supported by the driver.
The internal microstep table can be adapted to specific motor characteristics to further reduce torque ripple, if desired.

1.2 Control Interfaces

1.2.1 Serial μ C Interface

From the software point of view, the TMC429 provides a set of registers, accessed by a microcontroller via a serial interface in a uniform way. Each datagram contains address bits, a read-write selection bit, and data bits to access the registers and the on-chip memory. Each time the microcontroller sends a datagram to the TMC429 it simultaneously receives a datagram from the TMC429. This simplifies the communication with the TMC429 and makes programming easy. Most microcontrollers have an SPI hardware interface, which directly connects to the serial four wire microcontroller interface of the TMC429. For microcontrollers without SPI hardware software doing the serial communication is sufficient and can easily be implemented.

1.2.2 Step/Dir Driver Interface

The TMC429-LI controls the motor position by sending pulses on the STEP signal while indicating the direction on the DIR signal. A programmable step pulse length and step frequencies up to 1MHz allow operation at high speed and high microstep resolution. The driver chip converts these signals into the coil currents which control the position of the motor. The TMC429-LI perfectly fits to the TMC26x smart power Step/Dir driver family.

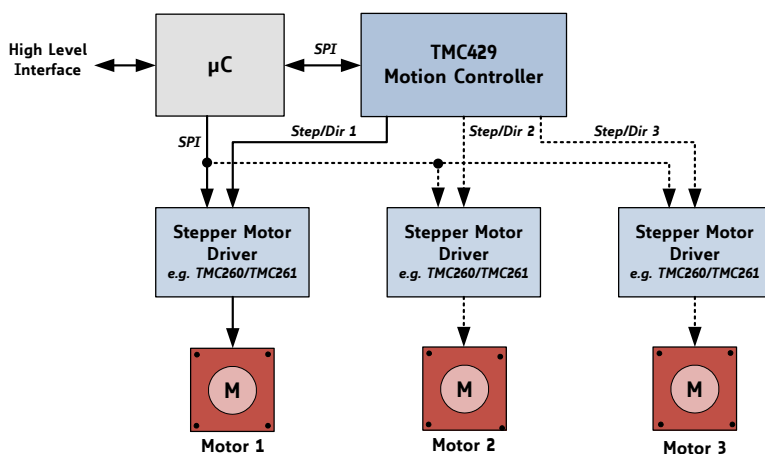


Figure 1.2 Application example using Step/Dir driver interface

1.2.3 Serial Driver Interface

The TMC429 automatically generates the required data-stream for SPI drivers and provides user configurable microstep waves and motor ramps for up to three motors. The serial interface to the motor drivers is flexibly configurable for different types (from different vendors) with up to 64 bit length for the SPI daisy chain. The TMC429-I perfectly fits to the TMC24x driver family.

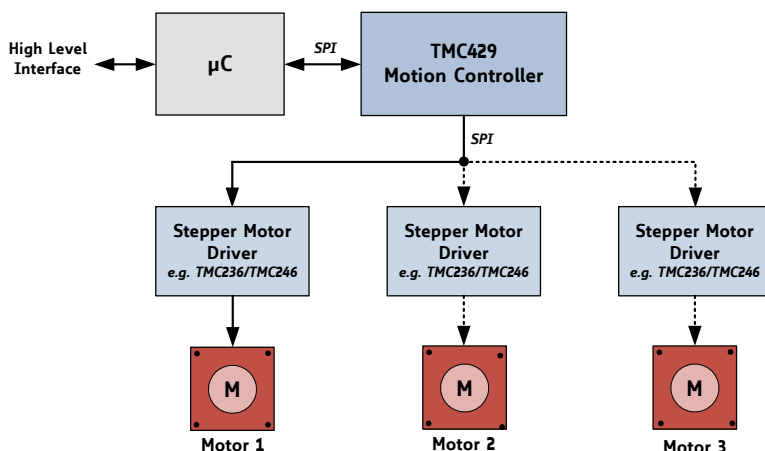


Figure 1.3 Application example using SPI driver interface

1.3 Software Visibility

From the software point of view the TMC429 provides a set of registers and on-chip RAM (see Figure 1.1), accessed via the serial μ C interface in a uniform way. The serial interface uses a simple protocol with fixed datagram length for the read- and write-access. These registers are used for initializing the chip as required by the hardware configuration. Afterwards the motor can be moved by writing target positions or velocity and acceleration values.

1.4 Step Frequencies

The desired motor velocity is an important design parameter of an application. Therefore it is important to understand the limiting factors.

1.4.1 Step Frequencies using the Step/Dir driver interface

The step pulses can directly be fed to a Step/Dir driver. The maximum full step rate (fsf_{max}) depends on the microstep resolution of the external driver chip.

The TMC429 microstep rate (μsf) is up to 1/32 of the clock frequency:

$$\mu sf_{max} = \frac{f_{CLK}}{32}$$

EXAMPLE FOR FULL STEP FREQUENCY CALCULATION

$f_{CLK} = 16 \text{ MHz}$

$\mu sf_{max} = 500 \text{ kHz}$

μ step resolution of external driver: 16

$$fsf_{max} = \frac{500 \text{ kHz}}{16} = 31.25 \text{ kHz}$$

With a standard motor with 1.8° per full step this results in up to 31.25kHz/200= 156 rotations per second, which is far above realistic motor velocities for this kind of motor and thus imposes no real limit on the application.

A 16 microsteps resolution can be extrapolated to 256 microsteps within the driver when using the TMC26x driver family.

1.4.2 Step frequencies using the SPI driver interface

The microstep unit with included sequencer processes step pulses from the pulse generator, which represent microsteps, half steps, or full steps (depending on the selected step resolution). The serial driver interface sends datagrams to the stepper motor driver chain whenever a step pulse comes.

The theoretical microstep frequency is identical to Step/Dir mode, but the achievable step frequency may be limited by the SPI data rate. Maximum SPI frequency (bit rate) is clock frequency divided by 16 (when CLK2DIV=7). An overhead of 1.5 bits is required per datagram. The maximum microstep transmission frequency depends on the total length of the datagrams sent to the SPI stepper motor driver chain.

EXAMPLE FOR SPI DATA RATE CALCULATION

At a clock frequency of 16 MHz, with a daisy chain of three SPI stepper motor drivers of 12 bit datagram length each (e.g. TMC246), the theoretical maximum SPI transmission frequency ($f_{SPI_{max}}$) is:

$$f_{SPI_{max}} = \frac{16 \text{ MHz}}{\frac{16}{3 \times 12 + 1.5}}$$

This is approximately 27 kHz. It is the theoretical upper limit for the fullstep frequency. In an application, the maximum desired fullstep frequency should be a factor 4 to 8 lower in order to avoid a beat between the step frequency and the SPI transmission rate.

The microstep rate may be higher than the SPI transmission frequency, even if the stepper motor driver does not note all microsteps due to the SPI data rate limit. At high step rates (respectively pulse rates) the differences between microstepping and full step excitation vanish.

1.5 Moving the Motor

Moving the motor is simple:

- To move a motor to a *new target position*, write the target position into the associated register by sending a datagram to the TMC429.
- To move a motor with a *new target velocity*, write the velocity into the register assigned to the stepper motor.

1.5.1 Motion Controller Functionality

The ramp generator monitors the motion parameters stored in its registers and calculates velocity profiles. Based on the actual ramp generator velocity a pulse generator supplies step pulses to the motor driver.

1.5.2 Modes of Motion – Individually Programmable for Each Axis

<i>ramp_mode</i>	For positioning applications the <i>ramp_mode</i> is most suitable. The user sets the position and the TMC429 calculates a trapezoidal velocity profile and drives autonomously to the target position. During motion, the position may be altered arbitrarily.
<i>velocity_mode</i>	For constant velocity applications the <i>velocity_mode</i> is most suitable. In <i>velocity_mode</i> , a target velocity is set by the user and the TMC429 takes into account user defined limits of velocity and acceleration.
<i>hold_mode</i>	In <i>hold_mode</i> , the user sets target velocities, but the TMC429 ignores any limits of velocity and acceleration, to realize arbitrary velocity profiles, controlled completely by the user.
<i>soft_mode</i>	The <i>soft_mode</i> is similar to the <i>ramp_mode</i> , but the decrease of the velocity during deceleration is done with a soft, exponentially shaped velocity profile.

1.5.3 Interrupts

The TMC429 has capabilities to generate interrupts. Interrupts are based on ramp generator conditions which can be set using an interrupt mask. The interrupt controller (which continuously monitors reference switches and ramp generator conditions) generates an interrupt if required.

SPECIAL HANDLING: TMC429-I / 16-PIN PACKAGE

- On 16-pin package the SDO_C signal becomes a low active interrupt signal called nINT_SDO_C while nSCS_C is high. Set SDO_INT=1 to access the non-multiplexed interrupt signal output nINT_SDO_C for the other packages.
- If the microcontroller disables the interrupt during access to the TMC429 and enables the interrupt otherwise, the multiplexed interrupt output of the TMC429 behaves like a dedicated interrupt output.
- For polling, the TMC429 sends the status of the interrupt signal to the microcontroller with each datagram.

1.5.4 Reference Switch Handling

The TMC429 has a left and a right reference switch input for each motor. Note, that these inputs are not available with all packages.

SPECIAL HANDLING: TMC429-I / 16-PIN PACKAGE

Because of its 16-pin package the TMC429-I has only three reference switch inputs: REF1, REF2, and REF3. Therefore the TMC429-I provides two different modes for reference switch handling:

- In the *Default Reference Switch Mode* the three reference switch inputs are defined as left side reference switches, one for each stepper motor.
- The *Second Reference Switch Mode* defines the first reference input REF1 as left reference switch input of motor one, the second reference input REF2 as left reference switch input of motor two, and the 3rd reference input REF3 as right reference switch input of motor one. In the second reference switch mode there is no reference switch input available for stepper motor three.
- With an external multiplexer 74HC157 any stepper motor may have a left and a right reference switch.

1.5.5 Integrated Programmable μ step Sequencer

The serial SPI interface to the stepper motor driver chain has to be configured by an initialization sequence which writes the configuration into the on-chip RAM. Once configured the serial driver interface works autonomously. The internal multiple port RAM controller of the TMC429 takes care of access scheduling. So, the user may read and write registers and on-chip RAM at any time. The registers hold global configuration parameters and motion parameters. The on-chip RAM stores the configuration of the serial driver interface and the microstep table.

The sequencer internally generates a number of control signals available for transmission to SPI driver ICs. These sequencer output signals are selected as configured by the internal stepper motor driver datagram configuration table.

During power-on reset, the TMC429 initializes a default configuration within the on-chip RAM for an SPI driver chain for TMC23x and TMC24x stepper motor drivers.

1.5.6 Access to Status and Error Bits

STEP/DIR

The microcontroller directly controls and monitors the stepper drivers. It also needs to take care for advanced current control, e.g. power down in stand still.

SPI

Many serial stepper motor drivers provide status bits (driver active, inactive...) and error bits (short to ground, wire open...), which are sent back from the stepper motor driver chain to the motion controller. To have access to error bits and datagrams with a total length up to 48 bits the TMC429 buffers the information by means of two 24 bit wide registers. The microcontroller has direct access to these registers.

Although, the TMC429 provides datagrams with up to 64 bits to the driver chain, only the last 48 bits sent back from the driver chain are buffered for read out by the microcontroller. Buffering of up to 48 bits is sufficient for a chain of three stepper motor drivers. For a chain of three TMC23x / TMC24x stepper motor driver chips all status bits are accessible.

2 General Definitions, Units, and Notations

2.1 Notations

- *Decimal numbers* are used as usual without additional identification.
- *Binary numbers* are identified by a prefixed % character.
- *Hexadecimal numbers* are identified by a prefixed \$ character.

EXAMPLE

Decimal: 42
Binary: %101010
Hexadecimal: \$2A

TMC429 DATAGRAMS ARE WRITTEN AS 32 BIT NUMBERS, E.G.:

\$1234ABCD = %0001 0010 0011 0100 1010 1011 1100 1101

TWO TO THE POWER OF N

In addition to the basic arithmetic operators (+, -, *, /) the operator *two to the power of n* is required at different sections of this data sheet. For better readability instead of 2^n the notation 2^n is used.

2.2 Signal Polarities

External and internal signals are high active per default, but the polarity of some signals is programmable to be inverted. A pre-fixed lower case *n* indicates low active signals (e.g. *nSCS_C*, *nSCS_S*). See chapter 8.2, too.

2.3 Units of Motion Parameters

The motion parameters *position*, *velocity*, and *acceleration* are given as integer values within TMC429 specific units. With a given stepper motor resolution one can calculate physical units for angle, angular velocity, angular acceleration. (See chapter 8.1.13)

2.4 Representation of Signed Values by Two's Complement

Motion parameters which have to cover negative and positive motion direction are processed as signed numbers represented by two's complement as usual. Limit motion parameters are represented as unsigned binary numbers.

SIGNED MOTION PARAMETERS ARE:

V_TARGET / *V_ACTUAL* / *A_ACTUAL* / *A_THRESHOLD*

UNSIGNED MOTION PARAMETERS ARE:

V_MIN / *V_MAX* / *A_MAX*

POSITIONS

X_TARGET / *X_ACTUAL* can be treated as signed or unsigned, as desired.

3 Package Variants

The TMC429 is available in three different package variants, qualified for the industrial temperature range. An additional variant is available for the automotive temperature range. All package variants are RoHS compliant.

Order code	Package	Characteristics	JEDEC Drawing
TMC429-LI	QFN32	5x5mm, 32 pins, plastic package, industrial (-40... +85°C)	
TMC429-PI24	SOP24	300 mils, 24 pins, plastic package, industrial (-40... +85°C)	MS-013 (300 mils)
TMC429-I	SSOP16	150 mils, 16 pins, plastic package, industrial (-40... +85°C)	MO-137 (150 mils)

4 Pin Assignments

The three package variants of the TMC429 offer different signal sets for various applications:

Type	Package	Compatibility	Remarks
TMC429-LI	QFN32		<ul style="list-style-type: none"> - Full functionality including SPI and Step/Dir driver interfaces for up to three stepper motor driver chips - Fits best to TMC26x and TMC389.
TMC429-PI24	SOP24	TMC428-PI24 replacement	<ul style="list-style-type: none"> - SPI interface for up to three stepper driver chips - STEP/DIR interface for up to three stepper driver chips - The right reference switch for motor 3 is not available.
TMC429-I	SSOP16	TMC428-I replacement	<ul style="list-style-type: none"> - SPI interface for up to three stepper motor driver chips (complements the TMC24x). - Step/Dir interface for up to two motors. - The additional reference right side switch inputs REF1R, REF2R, and REF3R are not available. - An additional multiplexer 74hc157 might be necessary. The multiplexing control signal is only available in SPI stepper motor driver chain mode.

Some third party SPI stepper motor drivers have no serial data output and therefore cannot simply be arranged in a daisy chain to drive more than one motor. The package variants SOP24 and QFN32 have two additional driver selection outputs nSCS2 and nSCS3 for stepper motor drivers without serial data output.

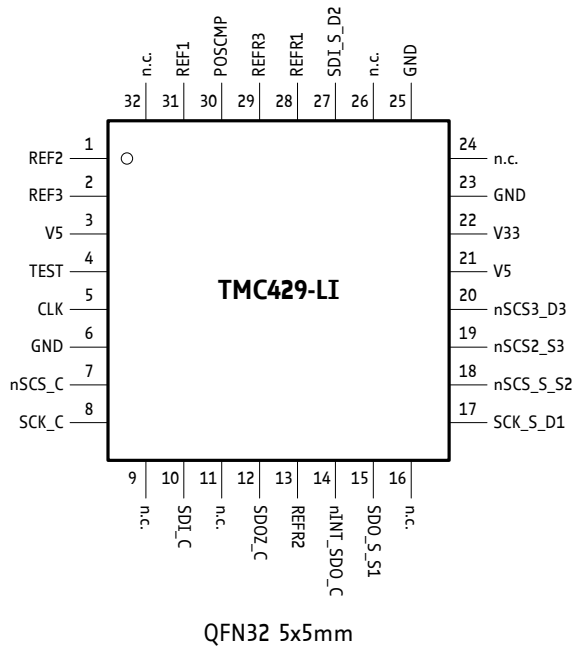
All inputs are Schmitt-Trigger. Unused inputs (REF1, REF2, REF3, and SDI_S) need to be connected to ground. Unused reference switch inputs have to be connected to ground, too. A pull-down resistor is necessary at the SDI_S input of the TMC429 for those serial peripheral interface stepper motor drivers that set their serial data output to high impedance Z while inactive.

STEP function outputs are S1, S2, and S3. Corresponding DIR outputs are D1, D2, and D3. The multiplexed output nINT_SDO_C of TMC429-LI and TMC429-PI24 can be configured in a de-multiplexed mode. An additional output named POSCMP is available for triggering when moving over a programmable position.

Attention

- After power on-reset, the TMC429 starts in TMC428 mode. That is, because the TMC429 is a 100% compatible successor of the TMC428 motion controller. *Additional outputs of the TMC429 including specific functions have to be activated by dedicated TMC429 configuration registers.*
- Preferably, long wires to the reference switch inputs (REF1, REF2, and REF3) should be avoided. For long wires, a low pass filter for spike suppression should be provided (refer the TMC429 evaluation board schematic as example).

4.1 Package Outlines



Please refer to the application note **PCB_Guidelines_TRINAMIC_packages** for a practical guideline for all available TRINAMIC IC packages and PCB footprints. The application note covers package dimensions, example footprints and general information on PCB footprints for these packages. It is available on www.trinamic.com.

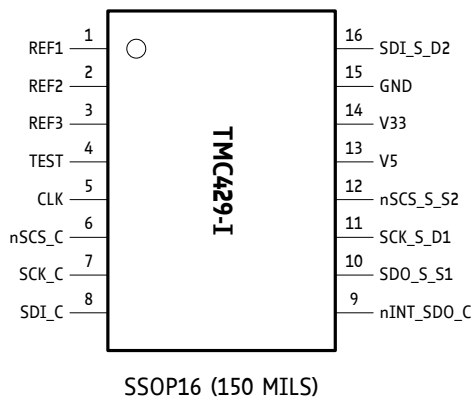
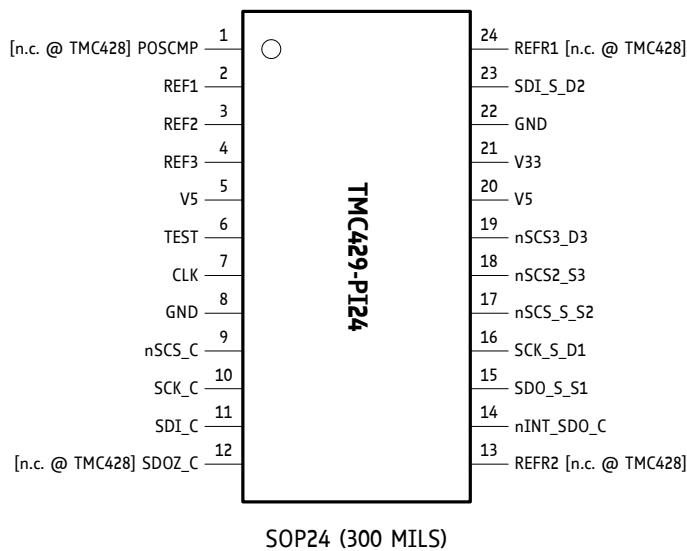


Figure 4.1 TMC429 pin out

4.2 Signal Descriptions

Pin	SSOP16	SOP24	QFN32	In/Out	Description
Reset	-	-	-	-	Internal power-on reset. No external reset input pin is available.
CLK	5	7	5	I	Clock input
nSCS_C	6	9	7	I	Low active SPI chip select input driven from μC
SCK_C	7	10	8	I	Serial data clock input driven from μC
SDI_C	8	11	10	I	Serial data input driven from μC
nINT_SDO_C	9	14	14	0	Serial data output to μC input / Multiplexed nINTERRUPT output if communication with μC is idle (resp. nSCS_C = 1) SDO_C will never be high impedance; the TMC429 is equipped with an additional pin named SDOZ_C that becomes high impedance when nSCS_C=1.
nSCS_S_S2	12	17	18	0	SPI chip select signal to stepper motor driving chain Step output S2 (for motor 2) in Step/Dir mode
nSCS2_S3	-	18	19	0	SPI chip select signal (SOP24 only) / Step output S3 (for motor 3) in Step/Dir mode
nSCS3_D3	-	19	20	0	SPI chip select signal (SOP24 only) / DIR output D3 (for motor 3) in Step/Dir mode
SCK_S_D1	11	16	17	0	Serial data clock output to SPI stepper motor driver chain / DIR output D1 (for motor 1) in Step/Dir mode
SDO_S_S1	10	15	15	0	Serial data output to SPI stepper motor driver chain / STEP output S1 (for motor 1) in Step/Dir mode
SDI_S_D2	16	23	27	I 0	Serial data input from SPI stepper motor driver chain (pull-up/down resistor at SDI_S avoids high impedance; SDI_S input is the power-on default) / DIR output D2 (for motor 2) in Step/Dir mode
REF1	1	2	31	I	Reference switch input 1 (no internal pull-up resistor)
REF2	2	3	1	I	Reference switch input 2 (no internal pull-up resistor)
REF3	3	4	2	I	Reference switch input 3 (no internal pull-up resistor)
V5	13	5, 20	3, 21		+5V supply / +3.3V supply
V33	14	21	22		470nF ceramic capacitor pin / +3.3V supply
GND	15	8, 22	6, 23, 25		Ground
TEST	4	6	4	I	<i>Must be connected to GND as close as possible to the chip. No user function.</i>
n.c.	-	-	9, 11, 16, 24, 26, 32	-	Not connected pins
POSCMP	-	1	30	n.c. / 0	Position compare output for SOP24 and QFN32 / Output for pos_comp function
SDOZ_C	-	12	12	0 / Z	SDOZ_C becomes high impedance (Z) when nSCS_C=1 / The nINT signal is not mapped to SDOZ_C pin / <i>The pin nINT_SDO_C can be configured with TMC429 register to give the nINT signal directly without multiplexing</i>
REFR1	-	24	28	I	Reference switch right 1 input Only available for TMC429 in SOP24 package and QFN32 package (with internal pull-up resistor)
REFR2	-	13	13	I	Reference switch right 2 input Only available for TMC429 in SOP24 package and QFN32 package (with internal pull-up resistor)
REFR3	-	-	29	I	Reference switch right 3 input Only available for TMC429 in QFN32 package (with internal pull-up resistor)

5 Sample Circuits

The sample circuits show the connection of the external components.

5.1 Application Example: TMC429 in QFN32 Package

All signals of the TMC429 are available with the QFN32 package. We recommend this package for applications using TRINAMICs TMC26x smart power driver family.

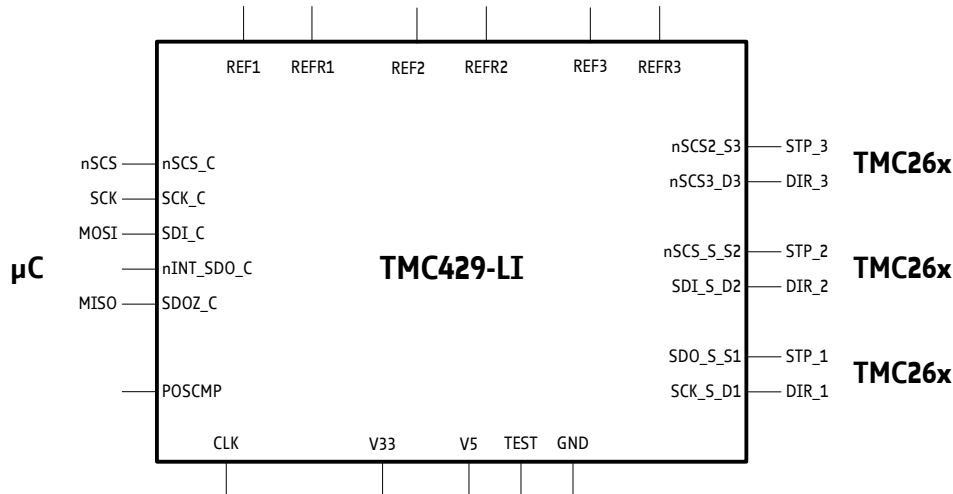


Figure 5.1 TMC429 within QFN32 package

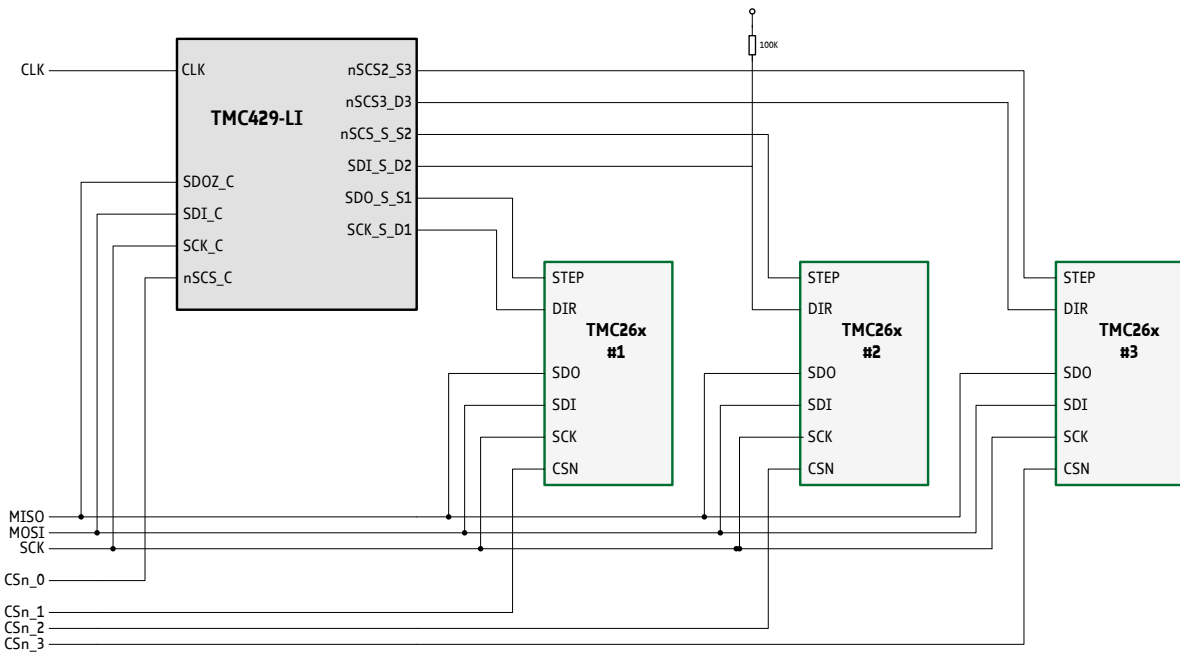


Figure 5.2 TMC429 / TMC26x outline for configuration via SPI and STEP/DIR for motion

APPLICATION ENVIRONMENT OF TMC429 (QFN32 PACKAGE) AND 3 X TMC26X STEPPER MOTOR DRIVER:

- One SPI chip select signal CSN_0 selects the TMC429 SPI microcontroller interface.
- Up to three SPI chip select signals (CSN_3, CSN_2, CSN1) select up to three TMC262 SPI for configuration.
- The TMC429 SDOZ_C is high impedance when nSCS_C is 1.

5.2 Application Example: TMC429 in SSOP16 Package

The low-priced TMC429-I is an optimum choice for SPI stepper motor drivers if the additional functions of the TMC429-LI are not required. We recommend this package for TRINAMICS TMC23x and TMC24x stepper driver family.

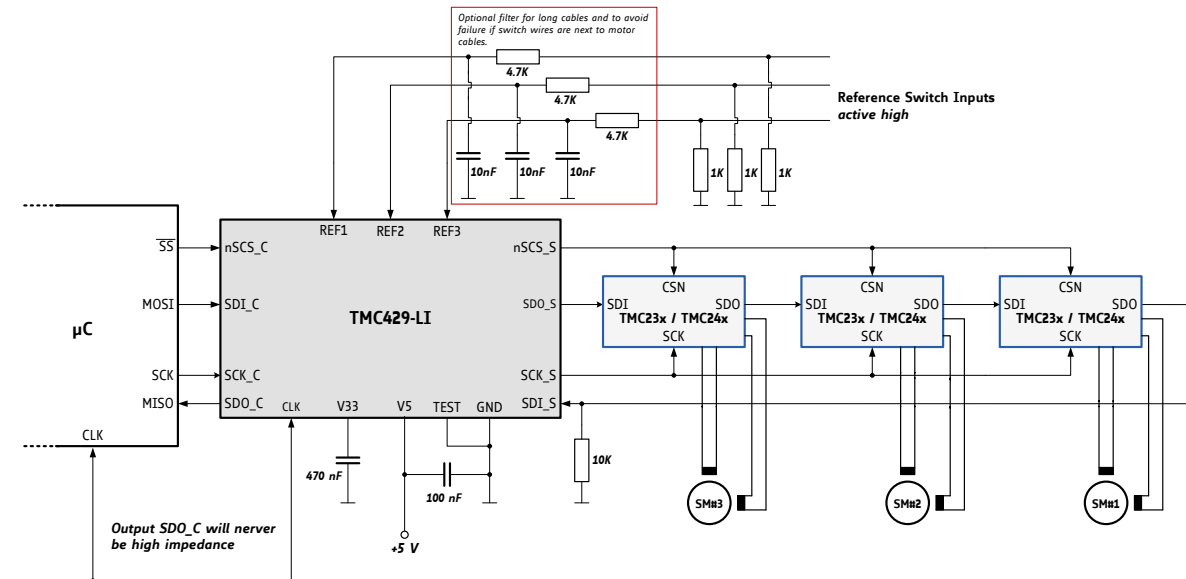


Figure 5.3 TMC429 application environment with TMC429 in SSOP16 package

5.3 Application Example: TMC429 with Drivers without Serial Data Output (SDO)

For driver chips without serial data output the TMC429-LI and the TMC429-PI24 with two additional chip select outputs are available. The TMC429 sends data to the driver chain on demand only, which minimizes the interface traffic and reduces the power consumption.

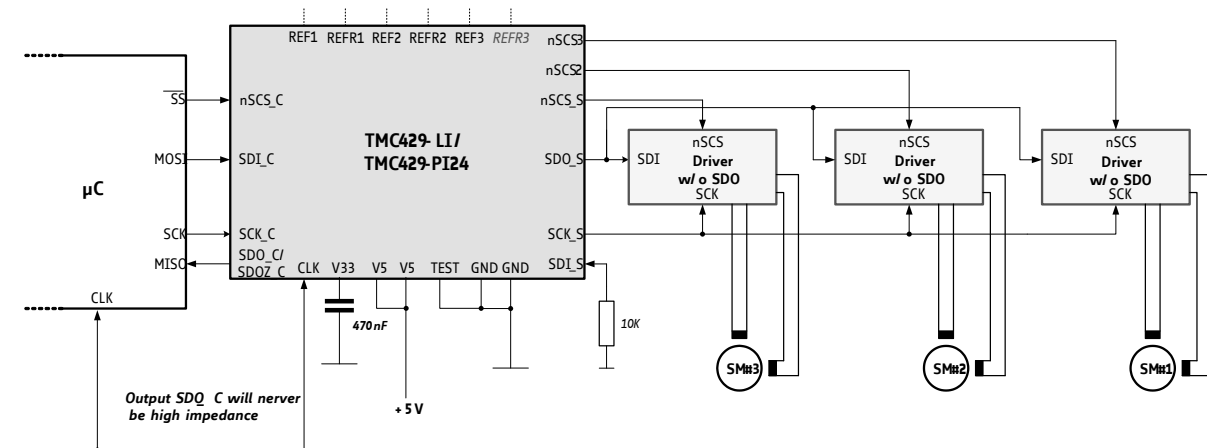


Figure 5.4 Usage of drivers without serial data output (SDO) with TMC429 in SOP24 or in QFN32 packages

6 Control Interface

The communication takes place via four wire serial interfaces and 32 bit datagrams of fixed length. Stepper motor drivers with parallel inputs can be used in connection with the TMC429 with some additional glue logic.

RESPONSIBILITIES ARE DEFINED AS FOLLOWS:

- The microcontroller is master of the TMC429.
- The TMC429 is master of the stepper motor driver daisy chain.

AUTOMATIC POWER-ON RESET:

- The TMC429 cannot be accessed before the power-on reset is completed and the clock is stable.
- All register bits are initialized with 0 during power-on-reset, except the SPI clock pre-divider `clk2_div` that is initialized with 15 (see section 8.2.5.3).

6.1 Bus Signals

Signal Description	TMC429 ↔ Microcontroller
Bus clock input	SCK_C
Serial data input	SDI_C
Serial data output	SDO_C
Chip select input	nSCS_C

6.2 Serial Peripheral Interface for μ C

The serial microcontroller interface of the TMC429 acts as a 32 bit shift register.

COMMUNICATION BETWEEN μ C AND THE TMC429

1. The serial μ C interface shifts serial data into SDI_C with each rising edge of the clock signal SCK_C.
2. Then, it copies the content of the 32 bit shift register into a buffer register with the rising edge of the selection signal nSCS_C.
3. The serial interface of the TMC429 immediately sends back data read from registers or read from internal RAM via the signal SDO_C.
4. The signal SDO_C can be sampled with the rising edge of SCK_C. SDO_C becomes valid at least four CLK clock cycles after SCK_C becomes low as outlined in the timing diagram.

6.2.1 Timing

A complete serial datagram frame has a fixed length of 32 bit. Because of on-the-fly processing of the input data stream, the serial μ C interface of the TMC429 requires the serial data clock signal SCK_C to have a minimum low / high time of three clock cycles. The SPI signals from the μ C interface may be asynchronous to the clock signal CLK of the TMC429.

If the microcontroller and the TMC429 work on different clock domains that run asynchronously by the timing of the SPI interface of the microcontroller should be made conservative in the way that the length of one SPI clock cycle equals 8 or more clock cycles of the TMC429 clock CLK.

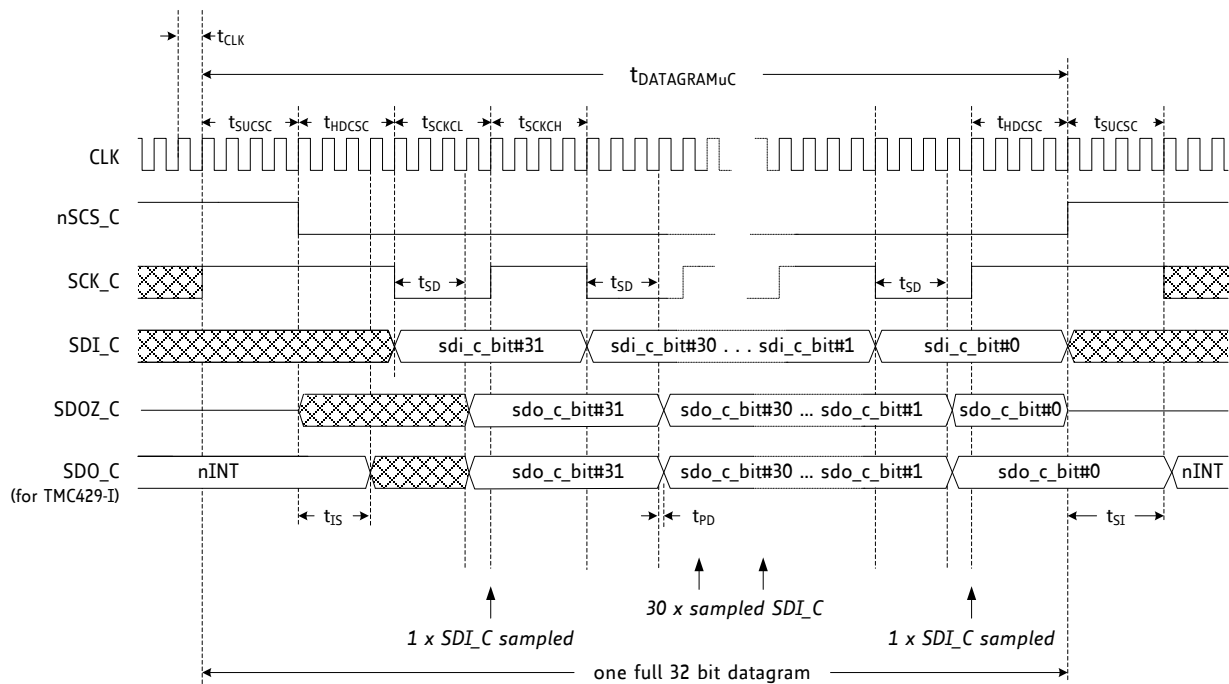


Figure 6.1 Timing diagram of the serial μ C interface

EXPLANATORY NOTES

- While the data transmission from the microcontroller to the TMC429 is idle, the low active serial chip select input nSCS_C and also the serial data clock signal SCK_C are set to high.
- While the signal nSCS_C is high, the TMC429 assigns the status of the internal low active interrupt signal nINT to the serial data output SDO_C.
- The data signal SDI_C driven by the microcontroller has to be valid at the rising edge of the serial data clock input SCK_C. The maximum duration of the serial data clock period is unlimited.
- While the μ C interface of the TMC429 is idle, the SDO_C signal is the (active low) interrupt status nINT of the integrated interrupt controller of the TMC429. The timing of the multiplexed interrupt status signal nINT is characterized by the parameters t_{IS} and t_{SI} (see chapter 16.3).

The following SPI clock frequencies are recommended in order to avoid possible issues concerning the SPI frequency between microcontroller and TMC429:

- For $f_{CLK} = 16\text{MHz}$ an upper SPI clock frequency of 1MHz is recommended.
- For $f_{CLK} = 32\text{MHz}$ an upper SPI clock frequency of 2MHz is recommended.

PROCEDURE OF DATA TRANSMISSION

1. The signal nSCS_C has to be high for at least three clock cycles before starting a datagram transmission. To initiate a transmission, the signal nSCS_C has to be set to low.
2. Three clock cycles later the serial data clock may go low.
3. The most significant bit (MSB) of a 32 bit wide datagram comes first and the least significant bit (LSB) is transmitted as the last one.
4. A data transmission is finished by setting nSCS_C high three or more CLK cycles after the last rising SCK_C slope.
5. So, nSCS_C and SCK_C change in opposite order from low to high at the end of a data transmission as these signals change from high to low at the beginning.

Information for TMC429-I / 16-pin package

In contrast to most other SPI compatible devices, the serial data output SDO_C of the TMC429-I is always driven. It will never be high impedance Z. If high impedance is required for the SDO_C connected to the microcontroller, it can be realized using a single gate 74HCT1G125. An additional pin named SDOZ_C is available for the TMC429 with an integrated high impedance driver.

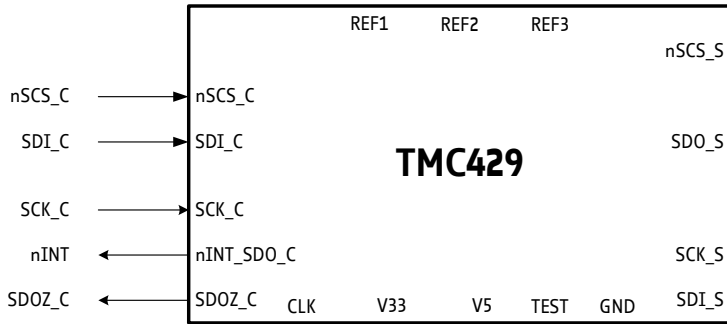


Figure 6.2 The TMC429 has a high impedance pin SDOZ_C. The nINT_SDO_C can be configured as non multiplexed interrupt output nINT if required.

TIMING CHARACTERISTICS OF THE SERIAL MICROCONTROLLER INTERFACE					
Symbol	Parameter	Min	Typ	Max	Unit
tSUCSC	Setup Clocks for nSCS_C	3		∞	CLK periods
tHDCSC	Hold Clocks for nSCS_C	3		∞	CLK periods
tSCKCL	Serial Clock Low	3		∞	CLK periods
tSCKCH	Serial Clock High	3		∞	CLK periods
tSD	SDO_C valid after SCK_C low	2.5		3.5	CLK periods
tIS	nINTERRUPT status valid after nSCS_C low	2.5			CLK periods
tSI	SDO_C valid after nSCS_C high			4.5	CLK periods
tDAMAGRAMuC	Datagram Length	$3+3+32*6= 198$		∞	CLK periods
tDAMAGRAMuC	Datagram Length	12.375		∞	μ s
fCLK	Clock Frequency	0		32	MHz
tCLK	Clock Period $t_{CLK} = 1 / f_{CLK}$	31.25		∞	ns
tPD	CLK-rising-edge-to-Output Propagation Delay		5		ns

6.2.2 Datagram Structure

The μ C communicates with the TMC429 via the four wire serial interface. Each datagram sent to the TMC429 via the pin SDI_C and each datagram received from the TMC429 via the pin SDO_C is 32 bits long.

The first bit sent is the *most significant bit (MSB)* sdi_c_bit#31. The last bit sent is the *least significant bit (LSB)* sdi_c_bit#0 (see Figure 6.1). During the reception of a datagram, the TMC429 immediately sends back a datagram of the same length to the microcontroller. This return datagram consists of requested read data in the lower 24 datagram bits and status bits in the higher 8 datagram bits. A read request is distinguished from a write request by the read/not write datagram bit (RW).

6.2.2.1 Datagrams Sent to the TMC429

The datagrams sent to the TMC429 are assorted in four groups of bits:

- RRS The *register RAM select (RRS) bit* selects either registers or the on-chip RAM.
- ADDRESS *Address bits* address memory within the register set or within the RAM area.
- RW The *read / not write (RW) bit* distinguishes between read access and write access:
read: RW = 1 / write RW = 0.
- DATA *Data bits* are only for write access. For read access these bits are not used (*don't care*) and should be set to 0.

MSB	32 BIT DATAGRAM SENT FROM μ C TO THE TMC429 VIA PIN SDI_C																															LSB				
	3	3	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	9	8	7	6		5	4	3	2
RRS	ADDRESS							RW	DATA																											

NOTE

- Different internal registers of the TMC429 have different lengths. For some registers only a subset of 24 data bits is used.
- Unused data bits should be set to 0.
- Some addresses select a couple of registers mapped together into the 24 data bit space.

6.2.2.2 Datagrams received by μ C from the TMC429

The datagrams received by the μ C from the TMC429 contain two groups of bits:

STATUS BITS The status bits, sent back with each datagram, comprehend the most important internal status bits of the TMC429 and the settings of the reference switches

DATA BITS Data bits are only for write access.

The most significant bit *MSB* is received first; the least significant bit *LSB* is received last. The TMC429 only sends datagrams on demand.

MSB	32 BIT DATAGRAM SENT BACK FROM THE TMC429 TO μ C VIA PIN SDO_C																								LSB						
	3	3	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1		1	0				
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
STATUS BITS										DATA BITS																					
INT	CDGW	SM3	SM2	SM1																											
		RS3	xEQt3	RS2	xEQt2	RS1	xEQt1																								

STATUS INFORMATION BITS

INT	The status bit <i>INT</i> is the <i>internal high active interrupt controller status output signal</i> . Handling of interrupt conditions without using interrupt techniques is possible by polling this status bit. The interrupt signal is also directly available at the SDO_C pin of the TMC429 (set SDO_INT=1 in if_configuration_429 register). The pin SDO_C may directly be connected to an interrupt input of the microcontroller. Since the SDO_C / nINT output on TMC429-I (16-pin package) is multiplexed, the microcontroller has to disable its interrupt input while it sends a datagram to the TMC429. The SDO_C signal driven by the TMC429 alternates during datagram transmission.
CDGW	The CDGW <i>cover datagram waiting</i> bit is a handshake signal for the microcontroller. It shows the state of a datagram covering mechanism that is necessary for direct configuration data transmission to the stepper motor driver chain, e.g. for configuring the drivers in the initialization phase. The CDGW status bit also gives the status of the <i>datagram_high_word</i> and <i>datagram_low_word</i> .
RS3, RS2, RS1	The status bits RS3, RS2 and RS1 represent the state of the left reference switch inputs. They are also accessible in register %1111100 as l3, l2 and l1.
xEQt3, xEQt2, xEQt1	The three status bits xEQt3, xEQt2, and xEQt1 indicate individually for each stepper motor, if it has reached its target position.

The status bits r1, r2, r3 and l1, l2, l3 and bits xEQt3, xEQt2, and xEQt1 can trigger an interrupt or enable simple polling techniques.

6.2.3 Simple Datagram Examples

The % prefix – normally indicating binary representation in this data sheet – is omitted for the following datagram examples. Assuming, one would like to write (RW=0) to a register (RRS=0) at the address %001101 the following data word %0000 0000 0000 0001 0010 0011, one would have to send the following 32 bit datagram

```
00011010000000000000000100100011
```

to the TMC429. With inactive interrupt (INT=0), no cover datagram waiting (CDGW=0), all reference switches inactive (RS3=0, RS2=0, RS1=0), and all stepper motors at target position (xEQt3=1, xEq2=1, xEq1=1) the status bits would be %10010101 the TMC429 would send back the 32 bit datagram:

```
10010101000000000000000000000000
```

To read (RW=1) back the register written before, one would have to send the 32 bit datagram

```
00011011000000000000000000000000
```

to the TMC429 and the TMC429 would reply with the datagram

```
10010101000000000000000100100011.
```

Write (RW=0) access to on-chip RAM (RRS=1) to an address %111111 occurs similar to register access, but with RRS=1. To write two 6 bit data words %100001 and %100011 to successive pair-wise RAM addresses %1111110 and %1111111 (%100001 to %1111110 and %100011 to %1111111) which are commonly addressed by one datagram, one would have to send the datagram

```
1111111000000000010001100100001.
```

To read (RW=1) from that on-chip memory address, one would have to send the datagram

```
11111111000000000000000000000000.
```

7 Address Space Partitions

The functionality of the TMC429 is mapped to registers which are combined to groups and mapped to the address space:

- Each stepper motor has a set of registers individually assigned to it and arranged within a contiguous address space.
- A set of registers within the address space holds the global parameters which are common for all stepper motors. A single dedicated global parameter register is essential for the configuration of the serial four wire stepper motor driver interface.
- One half of the on-chip RAM address space holds the configuration parameters for the stepper motor driver chain (*used for SPI mode, only*).
- The other half of the on-chip RAM address space is provided to store a microstep table if required (*used for SPI mode, only*).
- The first seven datagram bits (*sdi_c_bit#31 and sdi_c_bit#30 ... sdi_c_bit#25, respectively RRS and ADDRESS*) address the whole address space of the TMC429.

ADDRESS SPACE PARTITIONS				
Address ranges (incl. RRS)			Assignment	
%000 0000	...	%000 1111	16 registers for stepper motor #1	Registers with up to 24 bits
%001 0000	...	%001 1111	16 registers for stepper motor #2	
%010 0000	...	%010 1111	16 registers for stepper motor #3	
%011 0000	...	%011 1110	15 common registers	
		%011 1111	1 global parameter register	
%100 0000	...	%101 1111	32 addresses of 2x6 bit for driver chain configuration	RAM
%110 0000	...	%111 1111	32 addresses of 2x6 bit for microstep table	128x6 bit

CHANGING TARGET POSITION OR TARGET VELOCITY OF SINGLE MOTORS

The stepper motors are controlled directly by writing motion parameters into associated registers. Only one register write access is necessary for changing a target motion parameter. Thus the microcontroller has to send one 32 bit datagram to the TMC429 for altering the target position or the target velocity of one stepper motor.

CHANGING DRIVER CONFIGURATION OR MICROSTEP TABLE OF ALL MOTORS

Some parameters are packed together in a single data word at a single address. These parameters have to be initialized once and remain unchanged during operation. They have to be changed in common. The access to the on-chip RAM addresses concern two successive RAM addresses. So, always two data words are modified with each write access to the on-chip RAM.

Once initialized after power-up, the content of the RAM is usually left unchanged.

7.1 Read and Write

Read and write access is selected by the RW bit (*sdi_c_bit#24*) of the datagram sent from the μ C to the TMC429. The on-chip configuration RAM and the registers are writeable with read-back option. Some addresses are read-only. Write access (RW=0) to some of those read-only registers triggers additional functions, explained in detail later.

7.2 Register Set

The register address mapping is given in chapter 7.3. The registers are initialized internally during power-up. During power-up initialization, the TMC429 does not send any datagrams to the stepper motor driver chain.

The TMC429 loads a default RAM configuration for a TMC236 / TMC239 / TMC246 / TMC249 SPI driver chain on power-on reset. For a Step/Dir driver chain this is of no relevance.

7.3 Register Mapping

All register bits are initialized with 0 during power on reset, except the SPI clock pre-divider *clk2_div* (see section 8.2.5.3) that is initialized with 15. The on-chip RAM of the TMC429 is initialized internally during power-up. It can be modified by the microcontroller as required.

TMC429 REGISTER MAPPING																																															
32 BIT DATAGRAM SENT FROM μ C TO THE TMC429 VIA PIN SDI_C																																															
3	3	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0																
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R															
ADDRESS		RW	DATA																																												
SMDA		RW=0 : WRITE access / RW=1 : READ access	THREE STEPPER MOTOR REGISTER SETS (SMDA={00, 01, 10})																																												
IDX			<i>X_TARGET</i>																																												
0 0 0 0			<i>X_ACTUAL</i>																																												
0 0 0 1			<i>V_MIN</i>																																												
0 0 1 0			<i>V_MAX</i>																																												
0 0 1 1			<i>V_TARGET</i>																																												
0 1 0 0			<i>V_ACTUAL</i>																																												
0 1 0 1			<i>A_MAX</i>																																												
0 1 1 0			<i>A_ACTUAL</i>																																												
0 1 1 1			<i>A_THRESHOLD</i>																																												
1 0 0 0			<i>IS_AGTAT</i>							<i>IS_ALEAT</i>							<i>IS_v0</i>							<i>A_THRESHOLD</i>																							
1 0 0 1																	1				<i>PMUL</i>				<i>PDIV</i>																						
1 0 1 0																	<i>lp</i>				<i>REF_CONF</i>				<i>R_M</i>																						
1 0 1 1																	<i>INTERRUPT_MASK</i>								<i>INTERRUPT_FLAGS</i>																						
1 1 0 0																	<i>PULSE_DIV</i>				<i>RAMP_DIV</i>				<i>USRS</i>																						
1 1 0 1			<i>DX_REF_TOLERANCE</i>																																												
1 1 1 0			<i>X_LATCHED</i>																																												
1 1 1 1			<i>USTEP_COUNT_429</i>																																												
JDX			COMMON REGISTERS (SMDA=11)																																												
0 0 0 0			<i>DATAGRAM_LOW_WORD</i>																																												
0 0 0 1		<i>DATAGRAM_HIGH_WORD</i>																																													
0 0 1 0		<i>cw</i>														<i>COVER_POSITION</i>								<i>COVER_LEN</i>																							
0 0 1 1		<i>COVER_DATAGRAM</i>																																													
0 1 0 0		<i>IF_CONFIGURATION_429</i>																																													
0 1 0 1		<i>POS_COMP_429</i>																																													
0 1 1 0		<i>POS_COMP_INT_429</i>														<i>M</i>				<i>I</i>																											
1 0 0 0		POWER-DOWN																																													
1 0 0 1		<i>TYPE_VERSION_429</i> (= \$429101 for TMC429 version 1.01, read-only)																																													
1 1 1 0																																															
1 1 1 1		<i>motLr</i>														<i>refmux</i>				<i>cont_update</i>				<i>CLK2_DIV</i>				<i>cs ComInd</i>				<i>POLARITIES</i>				<i>LSMD</i>											
																				0 0 0 0				<i>STPDIV_429</i> (if <i>en_sd</i> =1)				<i>dac_ab</i>				<i>fd_ab</i>				<i>ph_ab</i>				<i>sk_s</i>				<i>nscs_s</i>			

□ unused bits

- SMDA = stepper motor driver address
- R_M = RAMP_MODE
- cw = cover waiting
- l1, l2, l3 = left switch 1/2/3 (read-out)
- M = mask
- I = interrupt
- RRS = register RAM select
- r1, r2, r3 = right switch 1/2/3 (read-out)

8 Register Description

The TMC429 provides axis parameter registers and global parameter registers.

8.1 Axis Parameter Registers

The registers hold binary coded numbers. Some are unsigned (positive) numbers, some are signed numbers in two's complement, and some are control bits or single flags. The functionality of different registers depends on the *RAMP_MODE* (refer to chapter 8.1.11).

OVERVIEW AXIS PARAMETER REGISTER MAPPING

REGISTER	R / W	TYPE	DESCRIPTION
<i>X_TARGET</i>	R/W	24 bit	This register holds the current target position in units of microsteps. Positions can be treated as signed or unsigned.
<i>X_ACTUAL</i>	R/W* ²	24 bit	The current position of each stepper motor is available by read out of this register. Positions can be treated as signed or unsigned.
<i>V_MIN</i>	R/W	11 bit unsigned	This register holds the absolute velocity value at or below which the stepper motor can be stopped abruptly.
<i>V_MAX</i>	R/W	11 bit unsigned	This parameter sets the maximum motor velocity.
<i>V_TARGET</i>	R/W	12 bit signed	The <i>V_TARGET</i> register holds the current target velocity. The use of <i>V_TARGET</i> depends on the chosen mode of operation.
<i>V_ACTUAL</i>	R* ¹	12 bit signed	This read-only register holds the current velocity of the associated stepper motor.
<i>A_MAX</i>	R/W	11 bit unsigned	This register defines the absolute value of the desired acceleration for <i>velocity_mode</i> and <i>ramp_mode</i> (resp. <i>soft_mode</i>) with a value range from 0 to 2047.
<i>A_ACTUAL</i>	R	12 bit signed	The actual acceleration can be read out by the microcontroller from the <i>A_ACTUAL</i> read-only register.
<i>IS_AGTAT</i> <i>IS_ALEAT</i> <i>IS_VO</i> <i>A_THRESHOLD</i>	R/W R/W R/W R/W	3 bit 3 bit 3 bit 11 bit unsigned	These parameters control the current scaling values <i>I_s</i> in SPI driver mode. Depending on the ramp phase they are applied to the motor by scaling the amplitudes of the internal sequencer.
<i>PMUL</i> <i>PDIV</i>	R/W R/W	1+7 bit 4 bit unsigned	These values form a floating point number with <i>PMUL</i> as mantissa and <i>PDIV</i> as exponent. <i>PMUL</i> and <i>PDIV</i> are used for calculating the deceleration ramp.
<i>RAMP_MODE</i> <i>REF_CONF</i> <i>lp</i>	R/W R/W R	2 bit 4 bit 1 bit	The two bits <i>RAMP_MODE</i> (<i>R_M</i>) select one of the four possible modes of operation. The configuration bits <i>REF_CONF</i> select the behavior of the reference switches. The bit called <i>lp</i> (latched position) is a read only status bit.
<i>INTERRUPT_MASK</i> <i>INTERRUPT_FLAGS</i>	R/W R/W	8 bit 8 bit	The TMC429 provides one interrupt register of eight flags for each stepper motor.
<i>RAMP_DIV</i> <i>PULSE_DIV</i> <i>USRS</i>	R/W R/W R/W	4 bit 4 bit 2 bit	The parameter <i>RAMP_DIV</i> scales the acceleration parameter <i>A_MAX</i> . The pulse generator clock – defining the maximum step pulse rate – is determined by the parameter <i>PULSE_DIV</i> . The parameter <i>PULSE_DIV</i> scales the velocity parameters. The parameter <i>USRS</i> (µstep resolution selection) is used for setting the microstep resolution in SPI mode.
<i>DX_REF_TOLERANCE</i>	R/W	12 bit	<i>DX_REF_TOLERANCE</i> excludes a motion range to allow motion near the reference position.
<i>X_LATCHED</i>	R	24 bit unsigned	This read-only register stores the actual position <i>X_ACTUAL</i> upon a change of the reference switch-state.
<i>USTEP_COUNT_429</i>	R/W	8 bit	The read-write register <i>USTEP_COUNT_429</i> holds the actual microstep pointer of the internal sequencer.

*¹ in *hold_mode* only, this register is a read-write register.

*² before overwriting *X_ACTUAL* choose *velocity_mode* or *hold_mode*. Refer to chapter 8.1.2.

8.1.1 *X_TARGET* (IDX=%0000)

This register holds the current target position in units of microsteps.

UNIT OF TARGET POSITION

The unit of the target position depends on the setting of the associated microstep resolution register *usrs*.

POSITIONING

- If the difference *X_TARGET* to *X_ACTUAL* is not zero and *R_M* = *ramp_mode* or *soft_mode*, the TMC429 moves the stepper motor in the direction of *X_TARGET* in order to position *X_ACTUAL* to *X_TARGET*. Usually *X_TARGET* is modified to start a positioning.
- The condition $|X_TARGET - X_ACTUAL| < 2^{23}$ must be satisfied for motion into correct direction.
- Target position *X_TARGET* and current position *X_ACTUAL* may be altered on the fly.
- To move from one position to another, the ramp generator of the TMC429 automatically generates ramp profiles in consideration of the velocity limits *V_MIN* and *V_MAX* and acceleration limit *A_MAX*.

The registers *X_TARGET*, *X_ACTUAL*, *V_MIN*, *V_MAX*, and *A_MAX* are initialized with zero after power up.

8.1.2 *X_ACTUAL* (IDX=%0001)

The current position of each stepper motor is available by read out of the registers called *X_ACTUAL*. The actual position can be overwritten by the microcontroller. This feature is important for the reference switch position calibration controlled by the microcontroller.

UNIT OF CURRENT POSITION

The unit of the target position depends on the setting of the associated microstep resolution register *usrs*.

Attention

Before overwriting *X_ACTUAL* choose *velocity_mode* or *hold_mode*.

If *X_ACTUAL* is overwritten in *ramp_mode* or *soft_mode* the motor directly drives to *X_TARGET*.

8.1.3 V_MIN (IDX=%0010)

This register holds the absolute velocity value at or below which the stepper motor can be stopped abruptly.

UNIT OF VELOCITY

The unit of velocity parameters is *steps per time unit*. The scale of velocity parameters (V_MIN , V_MAX , V_TARGET , V_ACTUAL) is defined by the parameter $PULSE_DIV$ (see page 8.1.13 for details) and depends on the clock frequency of the TMC429.

DECELERATION

- The parameter V_MIN is relevant for deceleration while reaching a target position. V_MIN should be set greater than zero.
- This control value allows reaching the target position faster because the stepper motor is not slowed down below V_MIN before the target is reached.
- Due to the finite numerical representation of integral relations the target position cannot be reached exactly, if the calculated velocity is less than one, before the target is reached. Setting V_MIN to at least one assures reaching each target position exactly.

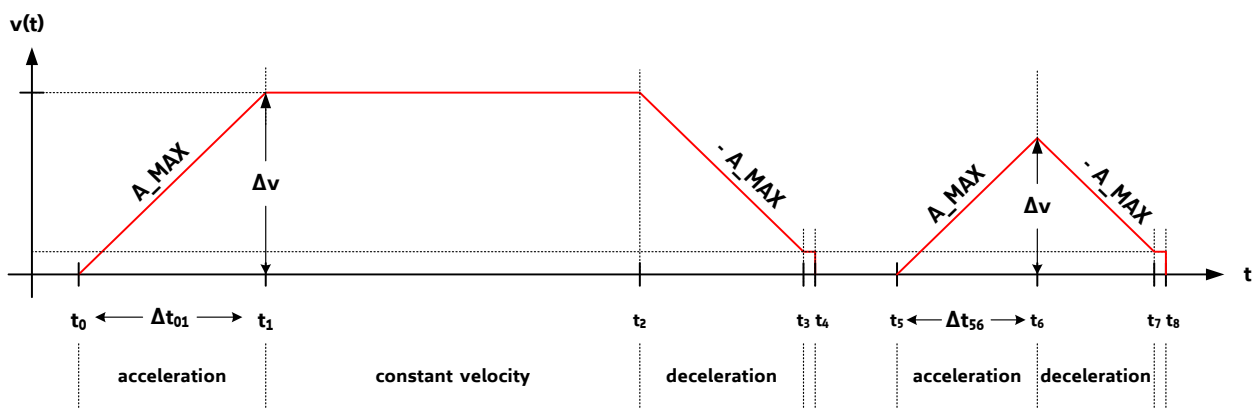


Figure 8.1 Velocity ramp parameters and velocity profiles

8.1.4 V_MAX (IDX=%0011)

This parameter sets the maximum motor velocity. The absolute value of the velocity will not exceed this limit, except if the limit V_MAX is changed during motion to a value below the current velocity.

UNIT OF VELOCITY

The unit of velocity parameters is *steps per time unit*. The scale of velocity parameters (V_MIN , V_MAX , V_TARGET , V_ACTUAL) is defined by the parameter $PULSE_DIV$ (see page 8.1.13 for details) and depends on the clock frequency of the TMC429.

HOMING PROCEDURE

To set target position X_TARGET and current position X_ACTUAL to an equivalent value (e.g. to set both to zero at a reference point) the assigned stepper motor should be stopped first and the parameter V_MAX should be set to zero to hold the assigned stepper motor at rest before writing into the register X_TARGET and X_ACTUAL .

Attention

Before overwriting X_ACTUAL choose *velocity_mode* or *hold_mode*.

If X_ACTUAL is overwritten in *ramp_mode* or *soft_mode* the motor directly drives to X_TARGET .

8.1.5 V_TARGET (IDX=%0100)

The use of V_TARGET depends on the chosen mode of operation:

Mode of operation	Functionality of V_TARGET
<i>ramp_mode</i>	The V_TARGET register holds the current target velocity calculated internally by the ramp generator.
<i>velocity_mode</i>	A target velocity can be written into the V_TARGET register. The associated stepper motor accelerates until it reaches the specified target velocity. The velocity is changed according to the motion parameter limits if the register V_TARGET is changed.
<i>hold_mode</i>	The register V_TARGET is ignored.
<i>soft_mode</i>	The V_TARGET register holds the current target velocity calculated internally by the ramp generator.

UNIT OF VELOCITY

The unit of velocity parameters is *steps per time unit*. The scale of velocity parameters (V_MIN , V_MAX , V_TARGET , V_ACTUAL) is defined by the parameter $PULSE_DIV$ (see chapter 8.1.13 for details) and depends on the clock frequency of the TMC429.

8.1.6 V_ACTUAL (IDX=%0101)

This read-only register holds the current velocity of the associated stepper motor. Internally, the ramp generator of the TMC429 processes with 20 bits while only 12 bits (the most significant bits) can be read out as V_ACTUAL .

In *hold_mode* only, this register is a read-write register. Writing zero to the register V_ACTUAL immediately stops the associated stepper motor, because hidden bits are set to zero with each write access to the register V_ACTUAL . In *hold_mode* motion parameters are ignored and the microcontroller has the full control to generate a ramp. The TMC429 only handles the microstepping and datagram generation for the associated stepper motor of the daisy chain.

UNIT

The unit of velocity parameters is *steps per time unit*. The scale of velocity parameters (V_MIN , V_MAX , V_TARGET , and V_ACTUAL) is defined by the parameter $PULSE_DIV$ (see chapter 8.1.13 for details) and depends on the clock frequency of the TMC429.

An actual velocity of zero read out by the microcontroller means that the *current velocity is in an interval between zero and one*. Therefore the actual velocity should not be used to detect a stop of a stepper motor. It is advised to detect the *target_reached* flag instead.

8.1.7 A_MAX (IDX=%0110)

This register defines the absolute value of the desired acceleration for *velocity_mode* and *ramp_mode* (resp. *soft_mode*) with a value range from 0 to 2047.

Note

The motion controller cannot stop the stepper motor if A_MAX is set to zero on the fly because afterwards the velocity cannot be changed automatically any more.

UNIT

The unit of the acceleration is *change of step frequency per time unit divided by 256*. The scale of acceleration parameters (A_MAX , A_ACTUAL , and $A_THRESHOLD$) is defined by the parameter $RAMP_DIV$ (see section 8.1.13) and depends on the clock frequency of the TMC429.

8.1.7.1 A_MAX in *ramp_mode*

As long as $RAMP_DIV \geq PULSE_DIV - 1$ is valid, any value of A_MAX within its range (0.. 2047) is allowed and there exists a valid pair $\{PMUL, PDIV\}$ for each A_MAX . The reason is that the acceleration scaling determined by $RAMP_DIV$ is compatible with the step velocity scaling determined by $PULSE_DIV$. A large $RAMP_DIV$ stands for low acceleration and a large $PULSE_DIV$ stands for low velocity. Low acceleration is compatible with low speed and high speed as well, but high acceleration is more compatible with high speed.