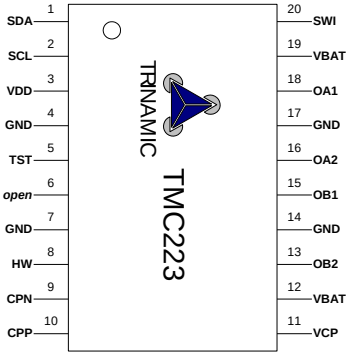
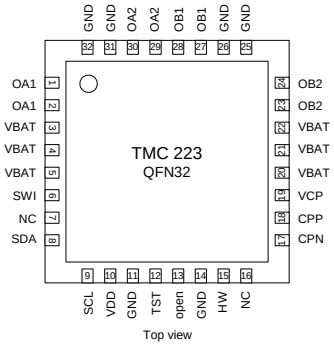
TMC223 – DATASHEET

Micro Stepping Stepper Motor Controller / Driver with Two Wire Serial Interface and Sensorless Stall Detection

TRINAMIC Motion Control GmbH & Co. KG
Hamburg, Germany





1 Features

The TMC223 is a combined micro-stepping stepper motor motion controller and driver with RAM and OTP memory and integrated sensorless stall detection. The RAM or OTP memory is used to store motor parameters and configuration settings. The TMC223 allows up to four bit of micro stepping and a coil current of up to 800 mA. After initialization it performs all time critical tasks autonomously based on target positions and velocity parameters. Communications to a host takes place via a two wire serial interface. Together with an inexpensive micro controller the TMC223 forms a complete motion control system. The main benefits of the TMC223 are:

- **Motor driver**
  - Controls one stepper motor with four bit micro stepping
  - Programmable Coil current up to 800 mA / Supply voltage range operating range 8V ... 29V
  - Fixed frequency PWM current control with automatic selection of fast and slow decay mode
  - Full step frequencies up to 1 kHz
  - High temperature, open circuit, short, over-current and under-voltage diagnostics
- **Motion controller**
  - Internal 16-bit wide position counter
  - Configurable speed and acceleration settings
  - Build-in ramp generator for autonomous positioning and speed control
  - On-the-fly alteration of target position
  - reference switch input available for read out
- **Two wire serial interface**
  - Transfer rates up to 350 kbps
  - Diagnostics and status information as well as motion parameters accessible
  - Field-programmable node addresses (32)
- **Sensorless Stall Detection**
    - GetFullStatus1 & GetFullStatus2 with parameters concerning stall detection
    - SetStallParam to set stall detection parameters

Life support policy

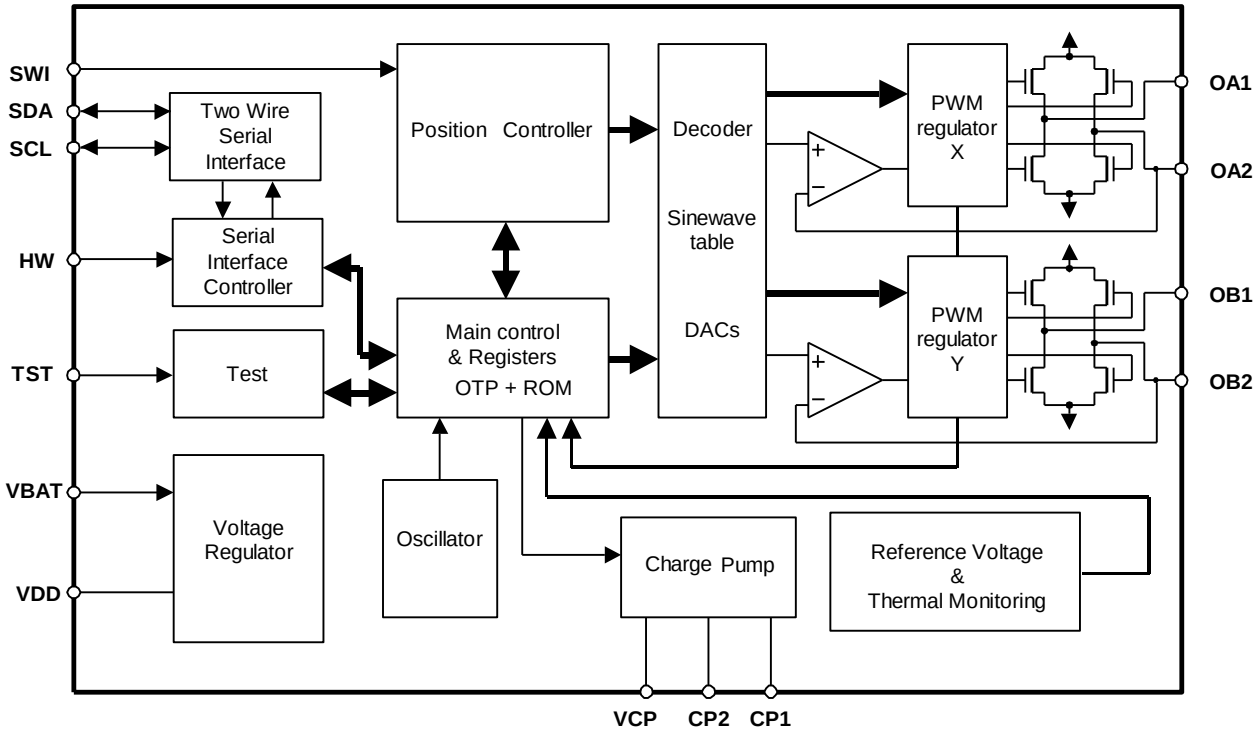
TRINAMIC Motion Control GmbH & Co. KG does not authorize or warrant any of its products for use in life support systems, without the specific written consent of TRINAMIC Motion Control GmbH & Co. KG.

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.

Table of Contents

2 General Description

2.1 Block Diagramm



2.2 Position Controller / Main Control

Motor parameters, e.g. acceleration, velocity and position parameters are passed to the main control block via the serial interface. These information are stored internally in RAM or OTP memory and are accessible by the position controller. This block takes over all time critical tasks to drive a stepper motor to the desired position under abiding the desired motion parameters.
The main controller gets feedback from the stepper motor driver block and is able to arrange internal actions in case of possible problems. Diagnostics information about problems and errors are transferred to the serial interface block.

2.3 Stepper Motor Driver

Two H-bridges are employed to drive both windings of a bipolar stepper motor. The internal transistors can reach an output current of up to 800 mA. The PWM principle is used to force the given current through the coils. The regulation loop performs a comparison between the sensed output current and the internal reference. The PWM signals to drive the power transistors are derived from the output of the current comparator.

2.4 Two Wire Serial Interface

Communication between a host and the TMC223 takes places via the two wire bi-directional serial interface. Motion instructions and diagnostics information are provided to or from the Main Control block. It is possible to connect up to 32 devices on the same bus. Slave addresses are programmable via OTP memory or an external pin.

2.5 Sensorless Stall Detection

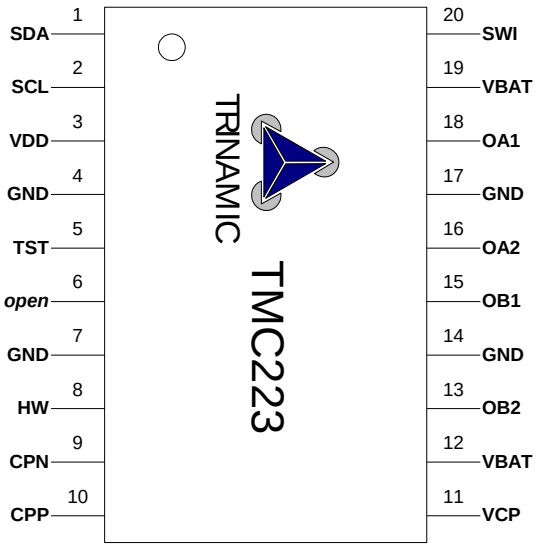
The TMC223 is equipped with a sensorless stall detection, to be used for noiseless reference search without reference switch and motion monitoring purposes as detection of motor blocking.

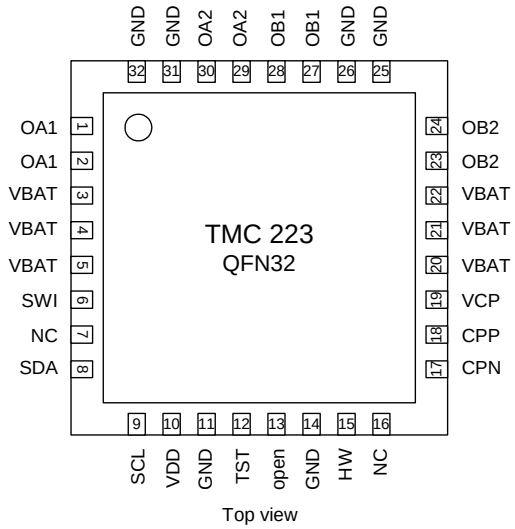
2.6 Miscellaneous

Besides the main blocks the TMC223 contains the following:

- an internal charge pump used to drive the high side transistors.
- an internal oscillator running at 4 MHz +/- 10% to clock the two wire serial interface, the positioning unit, and the main control block
- internal voltage reference for precise referencing
- a 5 Volts voltage regulator to supply the digital logic
- protection block featuring Thermal Shutdown, Power-On-Reset, etc.
- optional PWM jitter for reduction of EMI
- two programmable PWM frequencies (23 kHz and 46 kHz)

2.7 Pin and Signal Descriptions





Name	SOIC20	QFN32	Description
SDA	1	8	SDA Serial Data input/output
SCL	2	9	SCL Serial Clock input
VDD	3	10	internal supply (needs external decoupling capacitor)
GND	4,7,14,17	11,14,25,26, 31,32	ground, heat sink
TST	5	12	test pin (to be tied to ground in normal operation)
open	6	13	*must be left open*
HW	8	15	hard-wired serial interface address bit input **Hint:** The SWI is not a logic level input as usual; it needs to be connected via 1K resistor either to +VBAT or GND;
CPN	9	17	negative connection of external charge pump capacitor
CPP	10	18	positive connection of external charge pump capacitor
VCP	11	19	connection of external charge pump filter capacitor
VBAT	12, 19	3-5,20-22	battery voltage supply
OB2	13	23,24	negative end of phase B coil
OB1	15	27,28	positive end of phase B coil
OA2	16	29,30	negative end of phase A coil
OA1	18	1,2	positive end of phase A coil
SWI	20	6	reference switch input; **Hint:** The SWI is not a logic level input as usual; it needs to be connected via 1K resistor either to +VBAT or GND;
NC		7,16	internally not connected (shields when connected to ground)

Table 1: TMC223 Signal Description

3 Typical Application

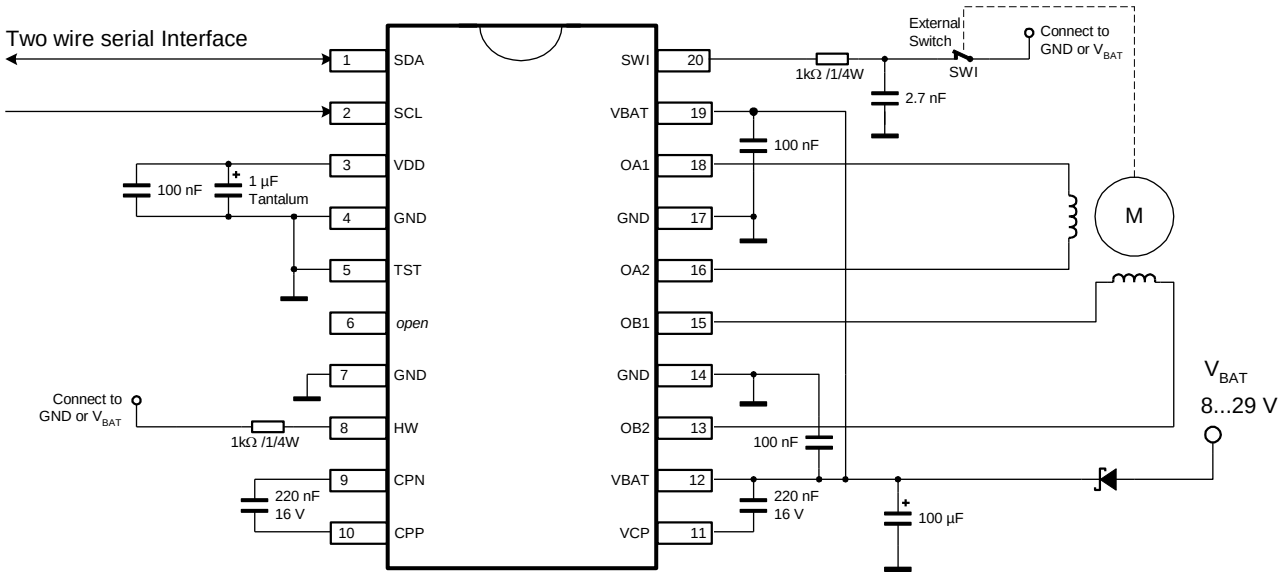


Figure 1: TMC223 Typical Application

Notes :

- Resistors tolerance +- 5%
- 2.7nF capacitors: 2.7nF is the minimum value, 10nF is the maximum value
- the 1µF and 100µF must have a low ESR value
- 100nF capacitors must be close to pins V_{BB} and V_{DD}
- 220nF capacitors must be as close as possible to pins CPN, CPP, V_{CP} and V_{BB} to reduce EMC radiation.

4 Ordering Information

Part No.	Package	Peak Current	Temperature Range
TMC223-SI	SOIC-20	800mA	-40°C..125°C
TMC223-LI	QFN32	800mA	-40°C..125°C

Table 2: Ordering Information

5 Functional Description

5.1 Position Controller and Main Controller

5.1.1 Stepping Modes

The TMC223 supports up to 16 micro steps per full step, which leads to smooth and low torque ripple motion of the stepping motor. Four stepping modes (micro step resolutions) are selectable by the user (see also Table 11):

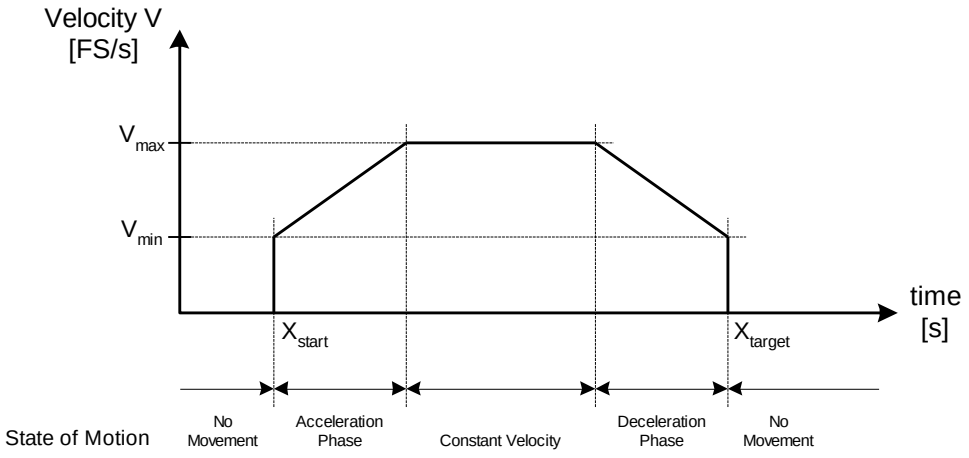
- Half step Mode
- 1/4 Micro stepping
- 1/8 Micro stepping
- 1/16 Micro stepping

5.1.2 Velocity Ramp

A common velocity ramp where a motor drives to a desired position is shown in the figure below. The motion consists of a acceleration phase, a phase of constant speed and a final deceleration phase. Both the acceleration and the deceleration are symmetrical. The acceleration factor can be chosen from a table with 16 entries. (Table 5: Acc Parameter on page 12). A typical motion begins with a start velocity Vmin. During acceleration phase the velocity is increased until Vmax is reached. After acceleration phase the motion is continued with velocity Vmax until the velocity has to be decreased in order to stop at the desired target position. Both velocity parameters Vmin and Vmax are programmable, whereas Vmin is a programmable ratio of Vmax. (See Table 3: Vmax Parameter on page 11 and Table 4: Vmin on page 12). The user has to take into account that Vmin is not allowed to change while a motion is ongoing. Vmax is only allowed to change under special circumstances. (See 5.1.4 Vmax Parameter on page 11).

The peak current value to be fed to each coil of the stepper-motor is selectable from a table with 16 possible values. It has to be distinguished between the run current Irun and the hold current Ihold. Irun is fed through the stepper motor coils while a motion is performed, whereas Ihold is the current to hold the stepper motor before or after a motion. More details about Irun and Ihold can be found in 5.3.1. and 5.3.2.

Velocity resp. acceleration parameters are accessable via the serial interface. These parameters are written via the SetMotorParam command (see 6.8.9) and read via the GetFullStatus1 command (see 6.8.1).



5.1.3 Examples for different Velocity Ramps

The following figures show some examples of typical motions under different conditions:

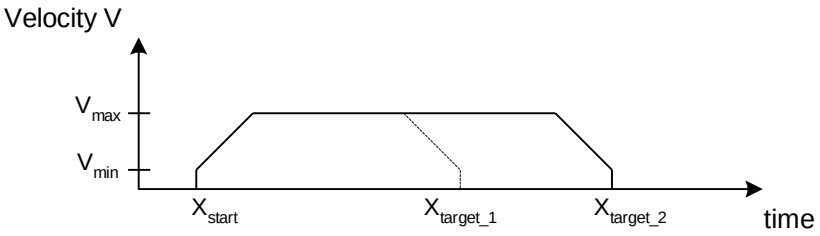


Figure 2: Motion with change of target position

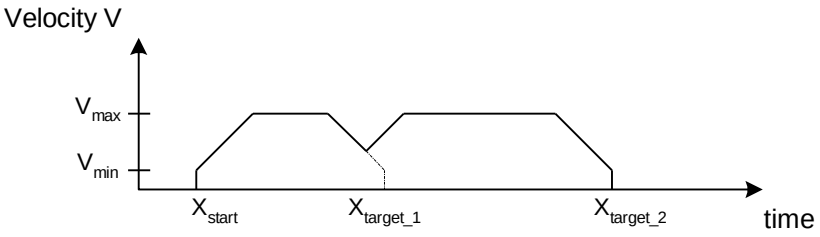


Figure 3: Motion with change of target position while in deceleration phase

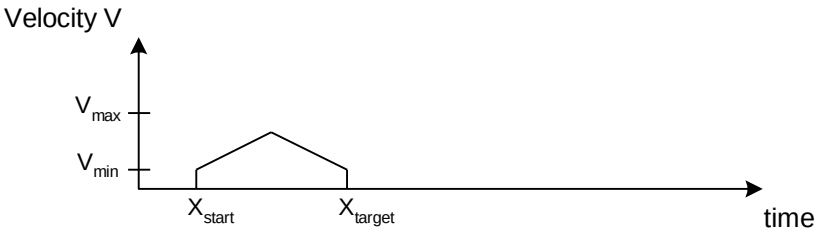


Figure 4: Short Motion Vmax is not reached

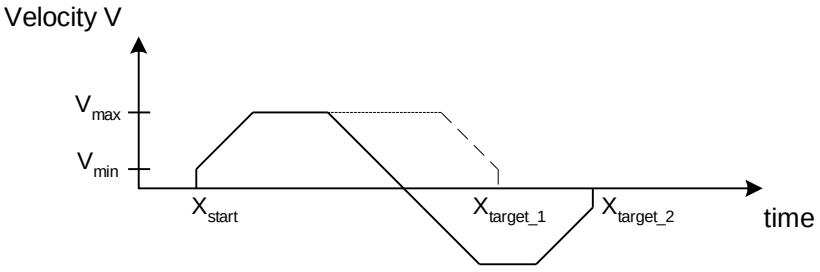


Figure 5: Linear Zero crossing (change of target position in opposite direction)

The motor crosses zero velocity with a linear shape. The velocity can be smaller than the programmed Vmin value during zero crossing. Linear zero crossing provides very low torque ripple to the stepper motor during crossing.

5.1.4 Vmax Parameter

The desired maximum velocity Vmax can be chosen from the table below:

Vmax index	Vmax [FS/s]	Vmax group	Stepping Mode			
			Half-Step Mode [half-steps/s]	1/4 micro stepping [micro-steps/s]	1/8 micro stepping [micro-steps/s]	1/16 micro stepping [micro-steps/s]
0	99	A	197	395	790	1579
1	136	B	273	546	1091	2182
2	167		334	668	1335	2670
3	197		395	790	1579	3159
4	213		425	851	1701	3403
5	228		456	912	1823	3647
6	243		486	973	1945	3891
7	273	C	546	1091	2182	4364
8	303		607	1213	2426	4852
9	334		668	1335	2670	5341
10	364		729	1457	2914	5829
11	395		790	1579	3159	6317
12	456		912	1823	3647	7294
13	546	D	1091	2182	4364	8728
14	729		1457	2914	5829	11658
15	973		1945	3891	7782	15564

Table 3: Vmax Parameter

Under special circumstances it is possible to change the Vmax parameters while a motion is ongoing. All 16 entries for the Vmax parameter are divided into four groups A, B, C and D. When changing Vmax during a motion take care that the new Vmax value is within the same group. Background: The TMC223 uses an internal pre-divider for positioning calculations. Within one group the pre-divider is equal. When changing Vmax between different groups during a motion, correct positioning is not ensured anymore.

5.1.5 Vmin Parameter

The minimum velocity parameter is a programmable ratio between 1/32 and 15/32 of Vmax. It is also possible to set Vmin to the same velocity as Vmax by setting Vmin index to zero. The table below shows the possible rounded values of Vmin given within unit [FS/s].

Vmin index	Vmax factor	Vmax group [A...D] and Vmax index [0…15]															
		A	B						C						D		
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	99	136	167	197	213	228	243	273	303	334	364	395	456	546	729	973
1	1/32	3	4	5	6	6	7	7	8	8	10	10	11	13	15	19	26
2	2/32	6	8	10	11	12	13	14	15	17	19	21	23	27	30	42	57
3	3/32	9	12	15	18	19	21	22	25	27	30	32	36	42	50	65	88
4	4/32	12	16	20	24	26	28	30	32	36	40	44	48	55	65	88	118
5	5/32	15	21	26	30	32	35	37	42	46	52	55	61	71	84	111	149
6	6/32	18	25	30	36	39	42	45	50	55	61	67	72	84	99	134	179
7	7/32	22	30	36	43	46	50	52	59	65	72	78	86	99	118	156	210
8	8/32	24	33	41	49	52	56	60	67	74	82	90	97	112	134	179	240
9	9/32	28	38	47	55	59	64	68	76	84	94	101	111	128	153	202	271
10	10/32	30	42	52	61	66	71	75	84	94	103	112	122	141	168	225	301
11	11/32	34	47	57	68	72	78	83	94	103	114	124	135	156	187	248	332
12	12/32	37	50	62	73	79	85	91	101	112	124	135	147	170	202	271	362
13	13/32	40	55	68	80	86	92	98	111	122	135	147	160	185	221	294	393
14	14/32	43	59	72	86	92	99	106	118	132	145	158	172	198	236	317	423
15	15/32	46	64	78	92	99	107	114	128	141	156	170	185	214	256	340	454

Table 4: Vmin values [FS/s] for all Vmin index – Vmax index combinations

5.1.6 Acceleration Parameter

The acceleration parameter can be chosen from a wide range of available values as described in the table below. Please note that the acceleration parameter is not to change while a motion is ongoing.

Acceleration Values in [FS/s²] dependent on Vmax																
Acc index	Vmax [FS/s]															
	99	136	167	197	213	228	243	273	303	334	364	395	456	546	729	973
0	49							106						473		
1	218													735		
2	1004															
3	3609															
4	6228															
5	8848															
6	11409															
7	13970															
8	16531															
9	14785	19092														
10		21886														
11		24447														
12		27008														
13		29570														
14		29570						34925								
15								40047								

Table 5: Acc Parameter

The amount of equivalent full steps during acceleration phase can be computed by the next equation:

$$N_{step} = \frac{V_{max}^2 - V_{min}^2}{2 \cdot Acc}$$

5.1.7 Position Ranges

Position information is coded by using two's complement format. Depending on the stepping mode (See 5.1.1) the position ranges are as listed in the following table:

Stepping Mode	Position Range	Full range excursion
Half-stepping	-4096…+4095 (-2^{12}…+2^{12}-1)	8192 half-steps 2^{13}
1/4 micro-stepping	-8192…+8191 (-2^{13}…+2^{13}-1)	16384 micro-steps 2^{14}
1/8 micro-stepping	-16384…+16383 (-2^{14}…+2^{14}-1)	32768 micro-steps 2^{15}
1/16 micro-stepping	-32768…+32767 (-2^{15}…+2^{15}-1)	65536 micro-steps 2^{16}

Table 6: Position Ranges

Target positions can be programmed via serial interface by using the SetPosition command (see 6.8.12). The actual motor position can be read by the GetFullStatus2 command (see 6.8.2).

5.1.8 Secure Position

The GotoSecurePosition command drives the motor to a pre-programmed secure position (see 6.8.4). The secure position is programmable by the user. Secure position is coded with 11 bits, therefore the resolution is lower than for normal positioning commands, as shown in the following table.

Stepping Mode	Secure Position Resolution
Half-stepping	4 half steps
1/4 micro stepping	8 micro steps (1/4th)
1/8 micro stepping	16 micro steps (1/8th)
1/16 micro stepping	32 micro steps (1/16th)

Table 7: Secure Position Resolution

5.1.9 External Switch

Pin SWI (see Figure 1, on page 8) will attempt to source and sink current in/from the external switch pin. This is to check whether the external switch is open or closed, resp. if the pin is connected to ground or Vbat. The status of the switch can be read by using the GetFullStatus1 command. As long as the switch is open, the <ESW> flag is set to zero.

The ESW flag just represents the status of the input switch. The SWI input is intended as a physical interface for a mechanical switch that requires a cleaning current for proper operation. The SWI input detects if the switch is open or connected either to ground or to Vbat. The SWI input is not a digital logic level input. The status of the switch does not automatically perform actions as latching of the actual position. Those actions have to be realized by the application software.

Important Hint: The SWI is not a logic level input as usual; it needs to be connected via 1K resistor either to +VBAT or GND;

5.1.10 Motor Shutdown Management

The TMC223 is set into motor shutdown mode as soon as one of the following condition occurs:

- The chip temperature rises above the thermal shutdown threshold T_{tsd}. See 5.1.12 Temperature Management on Page 16
- The battery voltage drops below UV2 See 5.1.13 Battery Voltage Management on Page 17.
- An electrical problem occurred, e.g. short circuit, open circuit, etc. In case of such an problem flag <ElDef> is set to one.
- Charge pump failure, indicated by <CPFail> flag set to one.

During motor shutdown the following actions are performed by the main controller:

- H-bridges are set into high impedance mode
- The target position register TagPos is loaded with the contents of the actual position register ActPos.

The two-wire-serial-interface remains active during motor shutdown. To leave the motor shutdown state the following conditions must be true:

- Conditions which led to a motor shutdown are not active anymore
- A GetFullStatus1 command is performed via serial interface.

Leaving the motor shutdown state initiates the following

- H-bridges in Ihold mode
- Clock for the motor control digital circuitry is enabled
- The charge pump is active again

Now the TMC223 is ready to execute any positioning command.

IMPORTANT NOTE:
First, a GetFullStatus1 command has to be executed after power-on to activate the TMC223.

5.1.11 Reference Search / Position initialization

A stepper motor does not provide information about the actual position of the motor. Therefore it is recommended to perform a reference drive after power-up or if a motor shutdown happened in case of a problem. The RunInit command initiates the reference search. The RunInit command consists of a Vmin and Vmax parameter and also position information about the end of first and second motion (6.8.8 RunInit).

A reference drive consists of two motions (Figure 6: RunInit): The first motion is to drive the motor into a stall position or a reference switch. The first motion is performed under compliance of the selected Vmax and Vmin parameter and the acceleration parameter specified in the RAM. The second motion has got a rectangular shape, without a acceleration phase and is to drive the motor out of the stall position or slowly towards the stall position again to compensate for the bouncing of the faster first motion to stop as close to the stall position as possible. The maximum velocity of the second motion equals to Vmin. The positions of Pos1 and Pos2 can be chosen freely (Pos1 > Pos2 or Pos1 < Pos2). After the second motion the actual position register is set to zero. Finally, the secure position will be traveled to if it is enabled (different from the most negative decimal value of –1024).

Once the RunInit command is started it can not be interrupted by any other command except a condition occurs which leads to a motor shutdown (See 5.1.10 Motor Shutdown Management) or a HardStop command is received. Furthermore the master has to ensure that the target position of the first motion is **not** equal to the actual position of the stepper motor and that the target positions of the first and the second motion are not equal. This is very important otherwise the circuit goes into a deadlock state. Once the circuit finds itself in a deadlock state only a HardStop command followed by a GetFullStatus1 command will cause the circuit to leave the deadlock state.

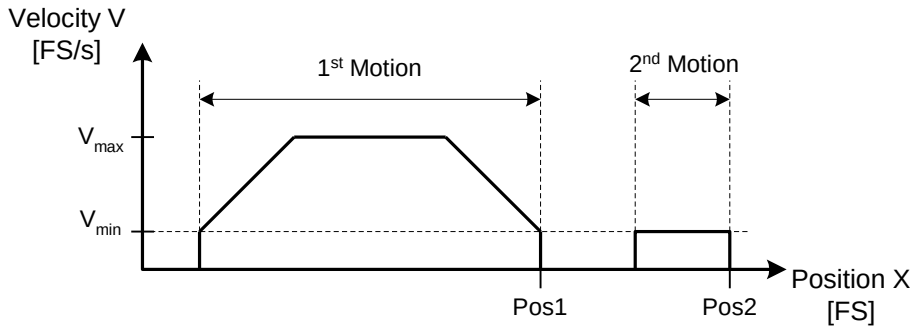
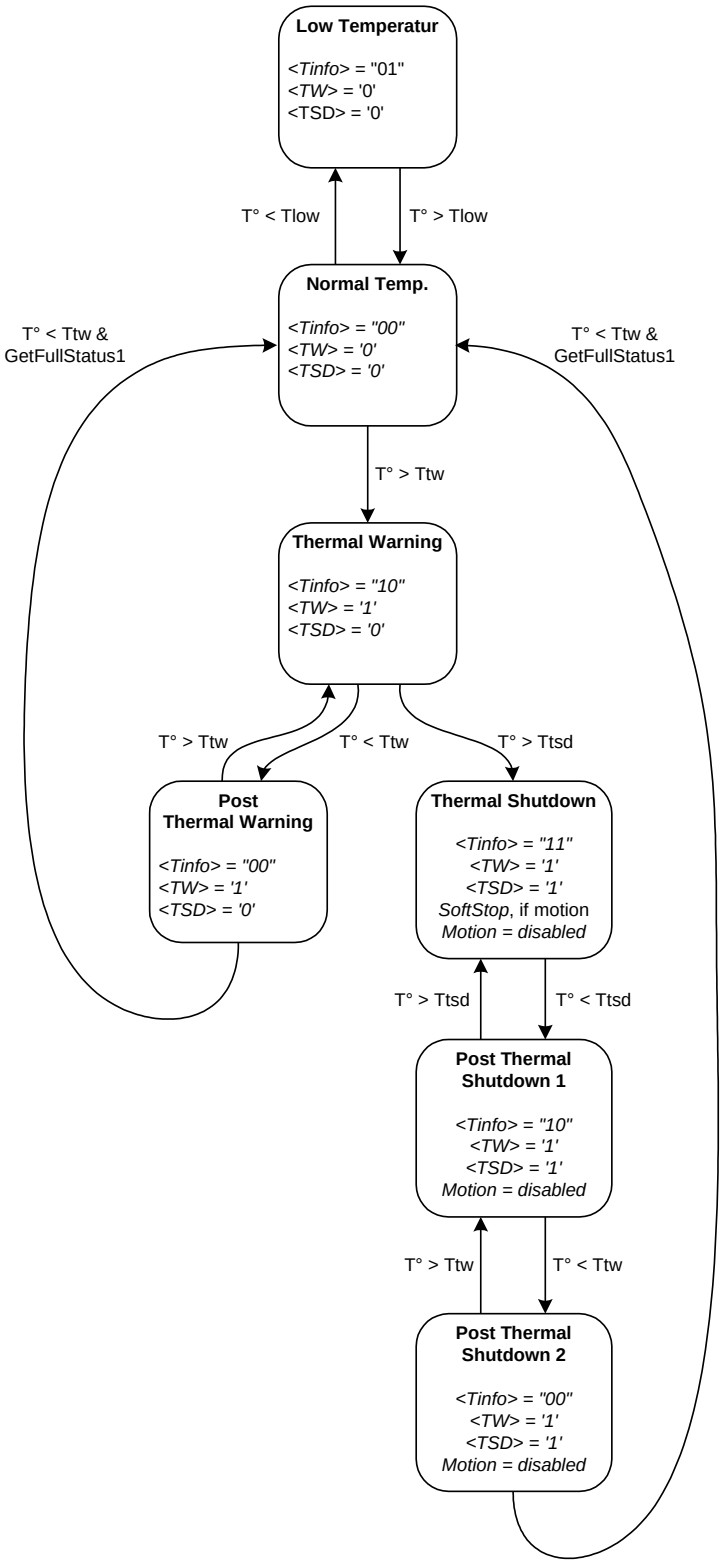


Figure 6: RunInit

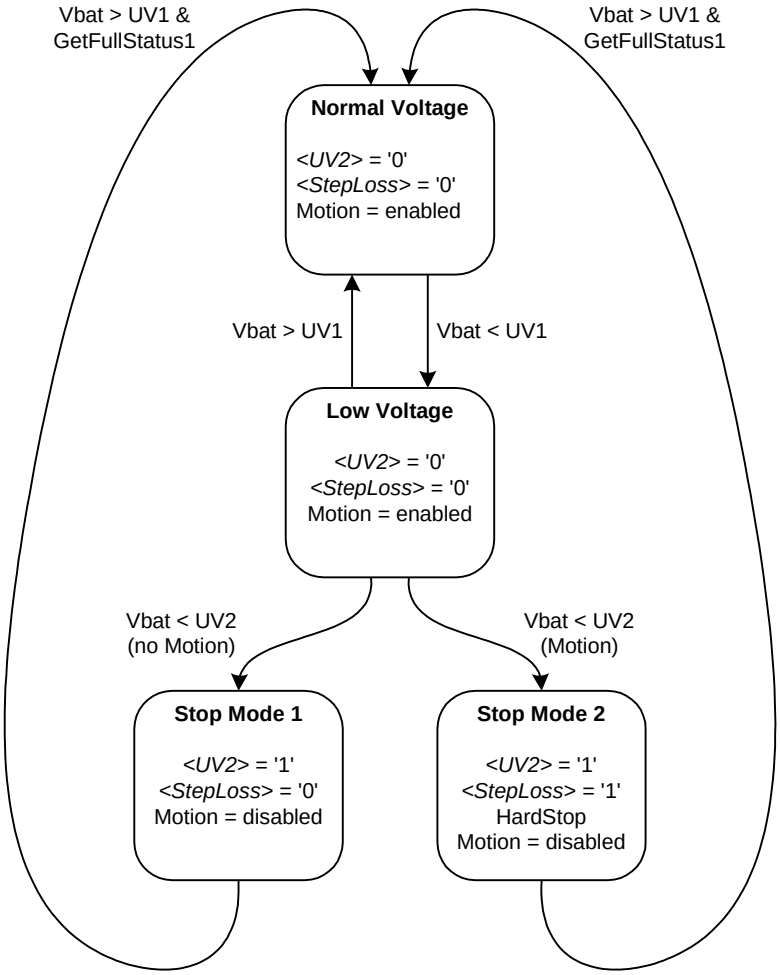
5.1.12 Temperature Management

The TMC223 provides an internal temperature monitoring. The circuit goes into shutdown mode if the temperature exceeds threshold T_{tsd}, furthermore two thresholds are implemented to generate a temperature pre-warning.



5.1.13 Battery Voltage Management

The TMC223 provides an internal battery voltage monitoring. The circuit goes into shutdown mode if the battery voltage falls below threshold UV2, furthermore one threshold UV1 is implemented to generate a low voltage warning.



5.1.14 Internal handling of commands and flags

The internal handling of commands and flags differs. Commands are handled with different priorities depending on the current state and the current status of internal flags, see figure below. SetPosition or GotoSecurePosition commands are ignored as long as the <StepLoss> flag is set. Details can be found in Table 8: Priority Encoder.

Note: A HardStop command is sent by the master or triggered internally in case of an electrical defect or over temperature.

A description of the available commands can be found in 6.8 Command Description. A list of the internal flags can be found in 5.2.2 Status Flags.

As an example: When the circuit drives the motor to its programmed target position, state “GotoPos” is entered. There are three events which can cause to leave this state: HardStop command received, SoftStop command received or target position reached. If all three events occur at the same time the HardStop command is executed since it has the highest priority. The Motion finished event (target position reached) has the lowest priority and thus will only cause transition to “Stopped” state when both other events do not occur.

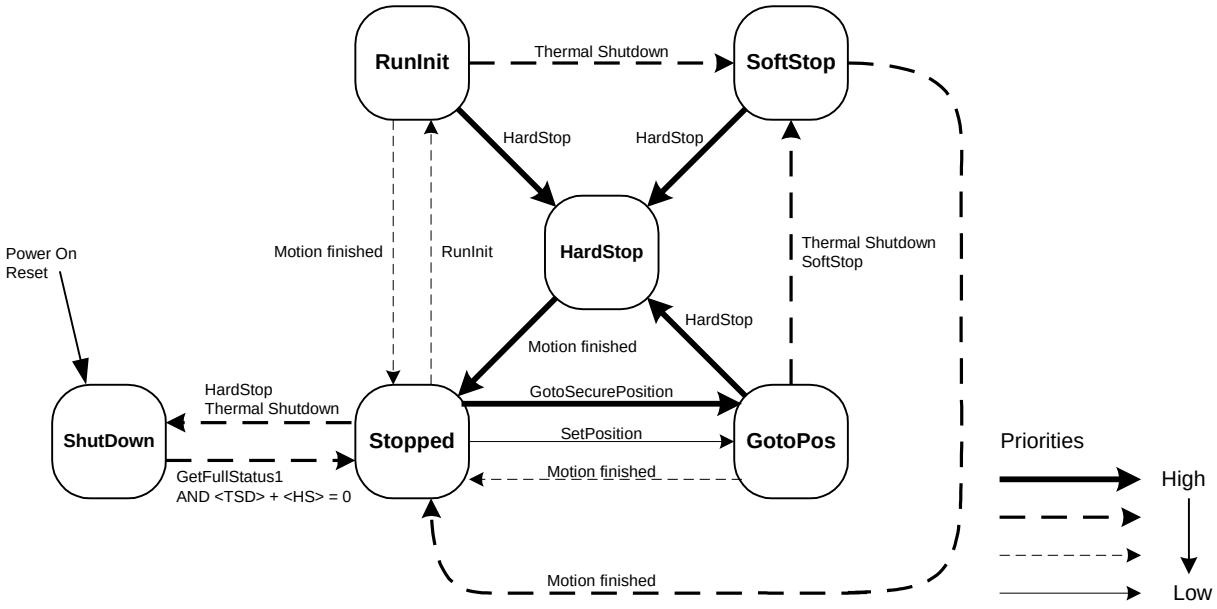


Figure 7: Internal handling of commands and flags

State →	Stopped	GotoPos	RunInit	SoftStop	HardStop	ShutDown
Command ↓	motor stopped, Ihold in coils	motor motion ongoing	no influence on RAM and TagPos	motor decelerating	motor forced to stop	motor stopped, H-bridges in Hi-Z
GetFullStatus2	I²C in-frame response	I²C in-frame response	I²C in-frame response	I²C in-frame response	I²C in-frame response	I²C in-frame response
GetOTPParam	OTP refresh; I²C in-frame	OTP refresh; I²C in-frame	OTP refresh; I²C in-frame	OTP refresh; I²C in-frame	OTP refresh; I²C in-frame	OTP refresh; I²C in-frame
GetFullStatus1 [attempt to clear <TSD> and <HS> flags]	I²C in-frame response	I²C in-frame response	I²C in-frame response	I²C in-frame response	I²C in-frame response	I²C in-frame response; if (<TSD> or <HS>) = '1' then **→ Stopped**
ResetToDefault [ActPos and TagPos are not altered]	OTP refresh; OTP to RAM; AccShape reset	OTP refresh; OTP to RAM; AccShape reset	OTP refresh; OTP to RAM; AccShape reset (note 2)	OTP refresh; OTP to RAM; AccShape reset	OTP refresh; OTP to RAM; AccShape reset	OTP refresh; OTP to RAM; AccShape reset
SetMotorParam [Master takes care about proper update]	RAM update	RAM update	RAM update	RAM update	RAM update	RAM update
ResetPosition	TagPos and ActPos reset					TagPos and ActPos reset
SetPosition	TagPos updated; **→GotoPos**	TagPos updated	TagPos updated			
GotoSecurePosition	If <SecEn> = '1' then TagPos = SecPos; **→GotoPos**	If <SecEn> = '1' then TagPos = SecPos	If <SecEn> = '1' then TagPos = SecPos			
RunInit	**→RunInit**					
HardStop		**→HardStop**; <StepLoss> = '1'	**→HardStop**; <StepLoss> = '1'	**→HardStop**; <StepLoss> = '1'		
SoftStop		**→SoftStop**				
HardStop [⇔ (<CPFail> or <UV2> or <ElDef>) = '1' ⇒ <HS> = '1']	**→Shutdown**	**→HardStop**	**→HardStop**	**→HardStop**		
Thermal shutdown [<TSD> = '1']	**→Shutdown**	**→SoftStop**	**→SoftStop**			
Motion finished	n.a.	**→Stopped**	**→Stopped**	**→Stopped**; TagPos =ActPos	**→Stopped**; TagPos =ActPos	n.a.

Table 8: Priority Encoder

Color code:

- Command ignored
- Transition to another state
- Master is responsible for proper update (see note 5)

Notes:

1 After Power on reset, the Shutdown state is entered. The Shutdown state can only be left after a GetFullStatus1 command (so that the Master could read the <VddReset> flag).

2 A RunInit sequence runs with a separate set of RAM registers. The parameters which are not specified in a RunInit command are loaded with the values stored in RAM at the moment the RunInit sequence starts. AccShape is forced to '1' during second motion even if a ResetToDefault command is issued during a RunInit sequence, in which case AccShape at '0' will be taken into account after the RunInit sequence. A GetFullStatus1 command will return the default parameters for Vmax and Vmin stored in RAM.

3 Shutdown state can be left only when <TSD> and <HS> flags are reset.

4 Flags can be reset only after the master could read them via a GetFullStatus1 command, and provided the physical conditions allow for it (normal temperature, correct battery voltage and no electrical or charge pump defect).

5 A SetMotorParam command sent while a motion is ongoing (state GotoPos) should not attempt to modify Acc and Vmin values. This can be done during a RunInit sequence since this motion uses its own parameters, the new parameters will be taken into account at the next SetPosition command.

6 <SecEn> = '1' when register SecPos is loaded with a value different from the most negative value (i.e. different from 0x400 = "100 0000 0000")

7 <Stop> flag allows to distinguish whether state Stopped was entered after HardStop/SoftStop or not. <Stop> is set to '1' when leaving state HardStop or SoftStop and is reset during first clock edge occurring in state Stopped.
8 While in state Stopped, if ActPos ≠ TagPos there is a transition to state GotoPos. This transition has the lowest priority, meaning that <Stop>, <TSD>, etc. are first evaluated for possible transitions.
9 If <StepLoss> is active, then SetPosition and GotoSecurePosition commands are ignored (they will not modify TagPos register whatever the state) and motion to secure position is forbidden. Other commands like RunInit or ResetPosition will be executed if allowed by current state. <StepLoss> can only be cleared by a GetFullStatus1 command.

5.2 RAM and OTP Memory

Some RAM registers (e.g. Ihold, Irun) are initialized with the content of the OTP (One Time Programmable) memory. The content of RAM registers that are initialized via OTP can be changed afterwards. This allows user initialization default values, whereas the default values are one time programmable by the user. Some OTP bits are address bits of the TMC223.

5.2.1 RAM Registers

Register	Mnemonic	Length (bit)	Related commands	Comment	Reset State
Actual Position	ActPos	16	GetFullStatus2 ResetPosition	Actual Position of the Stepper Motor. 16-bit signed	0x0000
Target Position	TagPos	16	SetPosition GetFullStatus2 ResetPosition	Target Position of the Stepper Motor. 16-bit signed	
Acceleration Shape	AccShape	1	GetFullStatus1 SetMotorParam ResetToDefault	0 = Acceleration with Acc Parameter. 1 = Velocity set to Vmin, without acceleration	
Coil Peak Current	Irun	4	GetFullStatus1 SetMotorParam ResetToDefault	Coil current when motion is ongoing (Table 12: Irun / Ihold Settings)	OTP Memory
Coil Hold Current	Ihold	4	GetFullStatus1 SetMotorParam ResetToDefault	Coil current when motor stands still (Table 12: Irun / Ihold Settings)	
Minimum Velocity	Vmin	4	GetFullStatus1 SetMotorParam ResetToDefault	Start Velocity of the stepper motor (Table 4: Vmin)	
Maximum Velocity	Vmax	4	GetFullStatus1 SetMotorParam ResetToDefault	Target Velocity of the stepper motor (Table 3: Vmax Parameter)	
Shaft	Shaft	1	GetFullStatus1 SetMotorParam ResetToDefault	Direction of motion	
Acceleration / Deceleration	Acc	4	GetFullStatus1 SetMotorParam ResetToDefault	Parameter for acceleration (Table 5: Acc Parameter)	
Secure Position	SecPos	11	GetFullStatus2 ResetToDefault	Target Position for GotoSecurePosition command (6.8.4 GotoSecurePosition); 11 MSBs of 16-bit position (LSBs fixed to '0')	
Stepping Mode	StepMode	2	GetFullStatus1 GetFullStatus2 ResetToDefault	Micro stepping mode (5.1.1 Stepping Modes)	

5.2.2 Status Flags

The table below shows the flags which are accessable by the serial interface in order to receive information about the internal status of the TMC223.

Flag	Mnemonic	Length (bit)	Related Command	Comment	Reset state
Digital supply Reset	VddReset	1	GetFullStatus1	Set to '1' after power-up or after a micro-cut in the supply voltage to warn that RAM contents may have been lost. Is set to '0' after GetFullStatus1 command.	'1'
Over current in coil A	OVC1	1	GetFullStatus1	Set to '1' if an over current in coil #1 was detected. Is set to '0' after GetFullStatus1 command.	'0'
Over current in coil B	OVC2	1	GetFullStatus1	Set to '1' if an over current in coil #2 was detected. Is set to '0' after GetFullStatus1 command.	'0'
StepLoss	StepLoss	1	GetFullStatus1	Set to '1' when under voltage, over current or over temperature event was detected. Is set to '0' after GetFullStatus1 command. SetPosition and GotoSecurePosition commands are ignored when <StepLoss> = 1	'0'
Secure position enabled	SecEn	1	Internal use	'0' if SecPos = "100 0000 0000" '1' otherwise	n.a.
Electrical Defect	ElDef	1	GetFullStatus1	Set to '1' if open circuit or a short was detected, (<OVC1> or <OVC2>). Is. Is set to '0' after GetFullStatus1 command.	'0'
Temperature Info	Tinfo	2	GetFullStatus1	Indicates the chip temperature "00" = normal temperature "01 = low temperature warning "10" = high temperature warning "11" = motor shutdown	"00"
Thermal Warning	TW	1	GetFullStatus1	Set to one if temperature raises above 145 °C. Is set to '0' after GetFullStatus1 command.	'0'
Thermal Shutdown	TSD	1	GetFullStatus1	Set to one if temperature raises above 155° C. Is set to '0' after GetFullStatus1 command and Tinfo = "00".	'0'
Motion Status	Motion	3	GetFullStatus1	Indicates the actual behavior of the position controller. "000": Actual Position = Target Position; Velocity = 0 "001": Positive Acceleration; Velocity > 0 "010": Negative Acceleration; Velocity > 0 "011": Acceleration = 0 Velocity = maximum pos Velocity "100": Actual Position /= Target Position; Velocity = 0 "101": Positive Acceleration; Velocity < 0 "110": Positive Acceleration; Velocity < 0 "111": Acceleration = 0 Velocity = maximum neg Velocity	"000"
External Switch Status	ESW	1	GetFullStatus1	Indicates the status of the external switch. '0' = open '1' = close	'0'
Charge Pump failure	CPFail	1	GetFullStatus1	'0' charge pump OK '1' charge pump failure	'0'
Electrical flag	HS	1	Internal use	<CPFail> or <UV2> or <ElDef>	'0'

5.2.3 OTP Memory Structure

The table below shows where the OTP parameters are stored in the OTP memory.

Note: If the OTP memory has not been programmed, or if the RAM has not be programmed by a SetMotorParam command, or if anyhow <VddReset> = '1', any positioning command will be ignored, in order to avoid any consequence due to unwanted RAM content. Please check that the correct supply voltage is applied to the circuit before zapping the OTP (See: Table 27: DC Parameters Supply and Voltage regulator on page 50), otherwise the circuit will be destroyed.

OTP Address	OTP Bit Order							
	7	6	5	4	3	2	1	0
0x00	OSC3	OSC2	OSC1	OSC0	IREF3	IREF2	IREF1	IREF0
0x01		TSD2	TSD1	TSD0	BG3	BG2	BG1	BG0
0x02	AbsThr3	AbsThr2	AbsThr1	AbsThr0	AD3	AD2	AD1	AD0
0x03	Irun3	Irun2	Irun1	Irun0	Ihold3	Ihold2	Ihold1	Ihold0
0x04	Vmax3	Vmax2	Vmax1	Vmax0	Vmin3	Vmin2	Vmin1	Vmin0
0x05	SecPos10.	SecPos9	SecPos8	Shaft	Acc3	Acc2	Acc1	Acc0
0x06	SecPos7	SecPos6	SecPos5	SecPos4	SecPos3	SecPos2		
0x07	DelThr3	DelThr2	DelThr1	DelThr0	StepMode1	StepMode0	LOCKBT	LOCKBG

Table 9: OTP Memory Structure

Parameters stored at address 0x00 and 0x01 and bit LOCKBT are already programmed in the OTP memory at circuit delivery, they correspond to the calibration of the circuit and are just documented here as an indication. Each OPT bit is at '0' when not zapped. Zapping a bit will set it to '1'. Thus only bits having to be at '1' must be zapped. Zapping of a bit already at '1' is disabled, to avoid any damage of the Zener diode. It is important to note that only one single OTP byte can be programmed at the same time (see command SetOTPParam).

Once OTP programming is completed, bit LOCKBG can be zapped, to disable unwanted future zapping, otherwise any OTP bit at '0' could still be zapped.

Lock bit	Protected byte
LOCKBT (zapped before delivery)	0x00 to 0x01
LOCKBG	0x02 to 0x07

Table 10: OTP Lock bits

The command used to load the application parameters via the serial bus into the RAM prior to an OTP Memory programming is SetMotorParam. This allows for a functional verification before using a SetOTPParam command to program and zap separately one OTP memory byte. A GetOTPParam command issued after each SetOTPParam command allows to verify the correct byte zapping.

5.3 Stepper Motor Driver

The StepMode parameter in SetMotorParam command (6.8.9 SetMotorParam on page 35) is used to select between different stepping modes. Following modes are available:

StepMode parameter	Mode
00	Half Stepping
01	1/4 µStepping
10	1/8 µStepping
11	1/16 µStepping

Table 11: StepMode

5.3.1 Coil current shapes

The next four figures show the current shapes fed to each coil of the motor in different stepping modes.

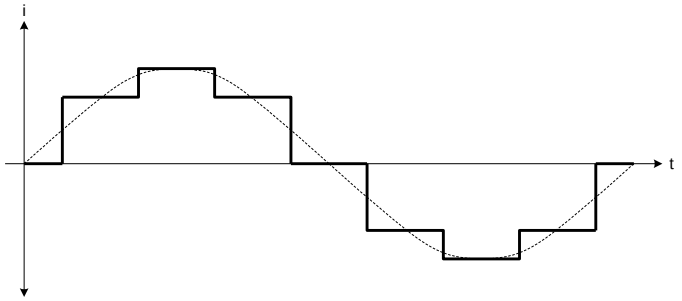


Figure 8: Coil Current for Half Stepping Mode

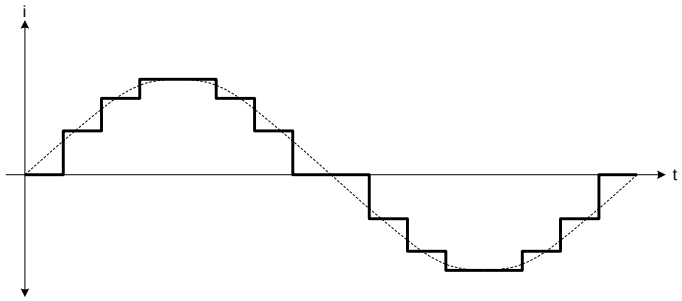


Figure 9: Coil Current for 1/4 Micro Stepping Mode

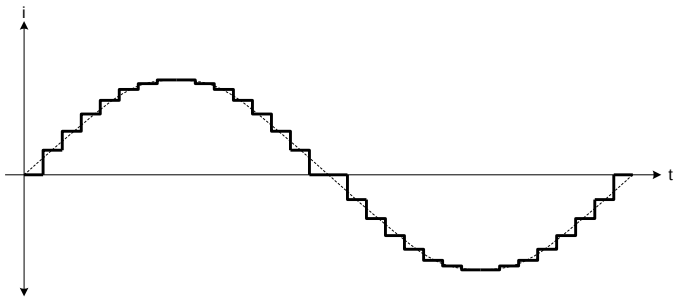


Figure 10: Coil Current for 1/8 Micro Stepping Mode

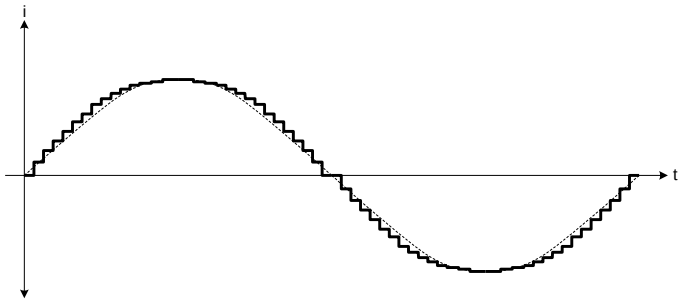


Figure 11: Coil Current for 1/16 Micro Stepping Mode

5.3.2 Transition Irun to Ihold

At the end of a motor motion the actual coil currents Irun are maintained in the coils at their actual DC level for a quarter of an electrical period (two half steps) at minimum velocity. Afterwards the currents are then set to their hold values Ihold. The figure below illustrates the mechanism:

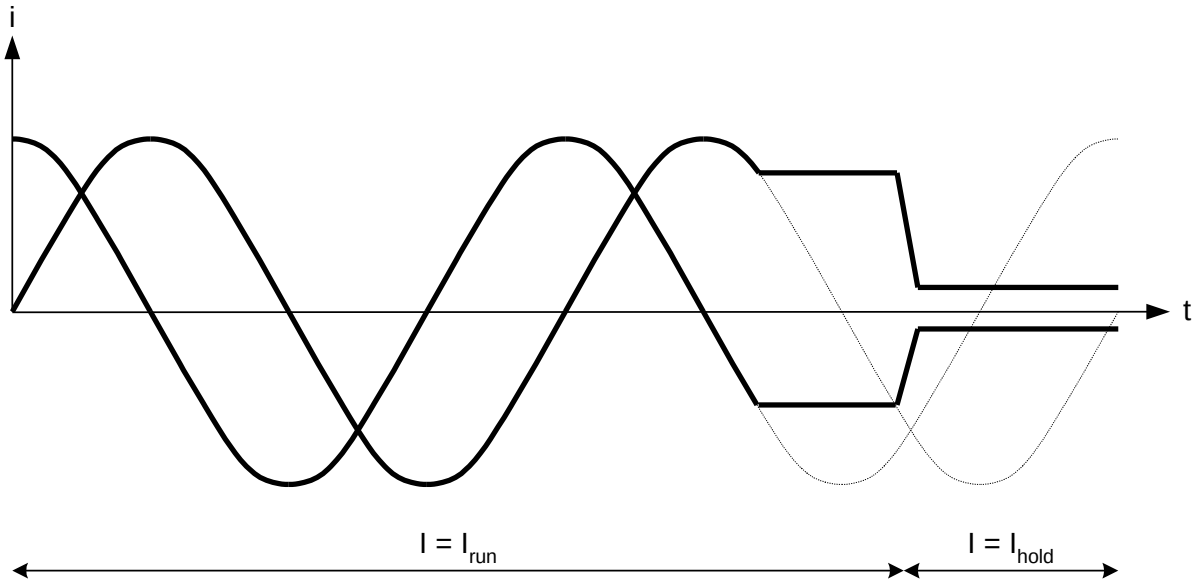


Figure 12: Transition Irun to Ihold

Both currents Irun and Ihold are parameterizeable using the command SetMotorParam. 16 values are available for Irun current and 16 values for Ihold current. The table below shows the corresponding current values.

Hint: The peak current of TMC223 is 0mA for setting Ihold = 0xF.

Irun / Ihold setting (hexadecimal)	Peak Current [mA]
0x0	59
0x1	71
0x2	84
0x3	100
0x4	119
0x5	141
0x6	168
0x7	200
0x8	238
0x9	283
0xA	336
0xB	400
0xC	476
0xD	566
0xE	673
0xF	0

Table 12: Irun / Ihold Settings

5.3.3 Chopper Mechanism

The chopper frequency is fixed as specified in chapter 11.4 AC Parameters on page 51. The TMC223 uses an intelligent chopper algorithm to provide a smooth operation with low resonance. The TMC223 uses internal measurements to derive current flowing through coils. If the current is less than the desired current, the TMC223 switches a H-bridge in a way that the current will increase. Otherwise if the current is too high, the H-bridge will be switched to decrease the current. For decreasing two modes are available, slow decay and fast decay, whereas fast decay decreases the current faster than slow decay. The figure below shows the chopper behavior.

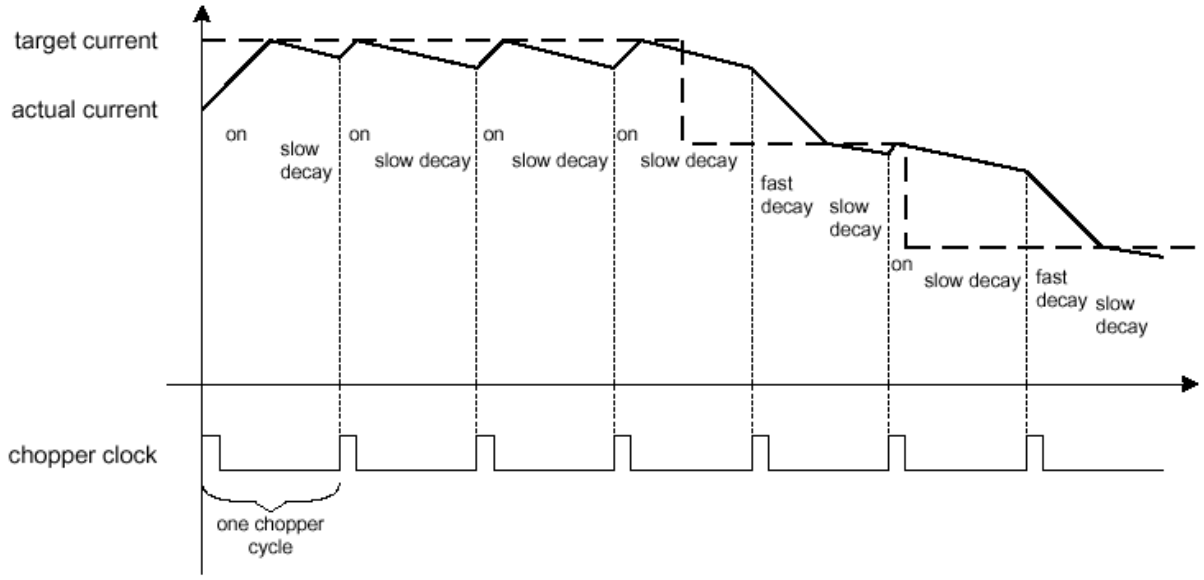


Figure 13: Different Chopper Cycles with Fast and Slow Decay

6 Two-Wire Serial Interface

6.1 Physical Layer

Both SDA and SCL lines are connected to positive supply voltage via a current source or pull-up resistor (see figure below). When there is no traffic on the bus both lines are high. Analog glitch filters are implemented to suppress spikes with a length of up to 50 ns.

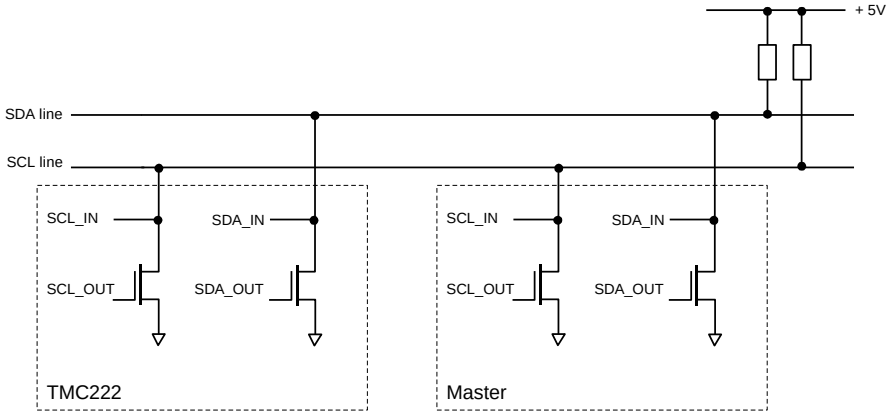


Figure 14: Two Wire Serial Interface - Physical Layer

6.2 Communication on Two Wire Serial Bus Interface

Each datagram starts with a Start condition and ends with a Stop condition. Both conditions are unique and cannot be confused with data. A high to low transition on the SDA line while SCL is high indicates a Start condition. A low to high transition on the SDA line while SCL is high defines a Stop condition (see figure below).

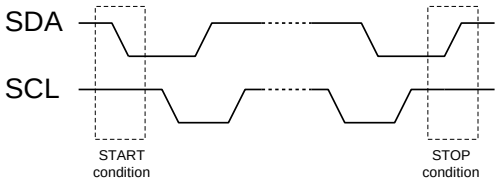


Figure 15: Two Wire Serial Interface - Start / Stop Conditions

The SCL clock is always generated by the master. On every rising transition of the SCL line the data on SDA is valid. Data on SDA line is only allowed to change as long as SCL is low (see figure below).

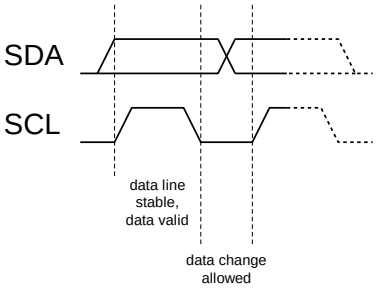


Figure 16: Two Wire Serial Interface - Bit transfer

Every byte put on the SDA line must have a length of 8 bits, where the most significant bit (MSB) is transferred first. The number of bytes that can be transmitted to the TCM222 is restricted to 8 bytes. Each byte is followed by an acknowledge bit, which is issued by the receiving node (see figure below).

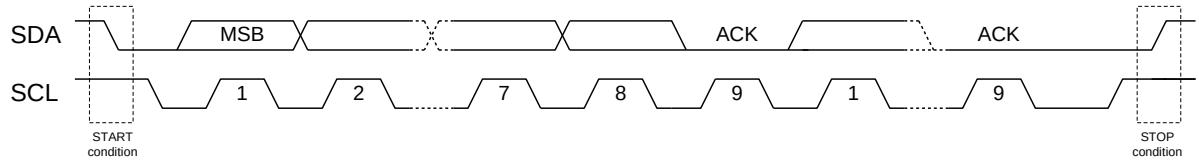


Figure 17: Two Wire Serial Interface - Data Transfer

6.3 Physical Address of the circuit

The circuit must be provided with a physical address in order to discriminate this circuit from other ones on the serial bus. This address is coded on seven bits (two bits are internally hardwired to ‘1’), yielding the theoretical possibility of 32 different circuits on the same bus. It is a combination of four OTP memory bits (see Table 9: OTP Memory Structure) and one hardwired address bit (pin HW). HW must either be connected to ground or Vbat. When HW is not connected and left floating correct functionality of the serial interface is not guaranteed. Pin HW uses the same principle to check whether it is connected to ground or Vbat like the SWI input (see 5.1.9 External Switch).

The TMC223 supports a “general call” address. Therefore the circuit is addressable using either the physical slave address or address “000 0000”.

AD6	AD5	AD4	AD3	AD2	AD1	AD0	Physical address
'1'	'1'	OTP_AD3	OTP_AD2	OTP_AD1	OTP_AD0		OTP Memory
						HW2	Hardwired Bit (Connect to 0 or 1)

Figure 18: Two Wire Serial Interface - Physical Address resp. Address Field

With un-programmed OTP address bits (OTP_AD3=o, OTP_AD2=o, OTP_AD1=o, OTP_AD0=o) and HW='0' (pin HW @ GND), the slave address resp. the address field of the TMC223 for reading is 11oooo01 (0xC1, 193) and for writing the slave address resp. the address field is 11oooo00 (0xC0, 192). The LSB of the address field selects read (='1') and write (='0'). With un-programmed OTP address bits and HW='1' (pin HW @ Vbat), the slave address resp. the address field of the TMC223 for reading is 11oooo11 (0xC3, 195) and for writing the salve address resp. the address field is 11oooo10 (0xC2, 194).

Important Hint: The HW is not a logic level input as usual; it needs to be connected via 1K resistor either to +VBAT or GND;

6.4 Write data to TMC223

A complete datagram consists of the following: a Start condition, the slave address (7 bit), a read/write bit (‘0’ = write, ‘1’ = read), an acknowledge bit, a number of data bytes (8 bit) each followed by an acknowledge bit, and a Stop condition. The acknowledge bit is used to signal to the transmitter the correct reception of the preceding byte, in this case the TMC223 pulls the SDA line low.

The TMC223 reads the incoming data at SDA with every rising edge of the SCL line. To finish the transmission the master has to transmit a Stop condition. Some commands for the TMC223 are supporting eight bytes of data, other commands are transmitting two bytes of data.

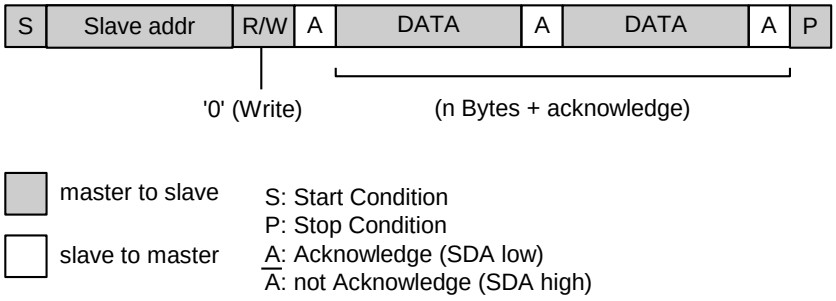


Figure 19: Two Wire Serial Interface - Writing Data to Slave

6.5 Read data from TMC223

When reading data from a slave two datagrams are needed. The first datagram consists of two bytes of data. The first byte consists of the slave address and the write bit. The second byte consists of the address of an internal register of the TMC223. The internal register address is stored in the circuits RAM. The second datagram consists of the slave address and the read bit. Then the master can read the data bits on the SDA line with every rising edge of the SCL line. After each byte of data the master has to acknowledge correct data reception by pulling SDA low. The last byte must not be acknowledged by the master so that the slave knows the end of transmission (see figure below).

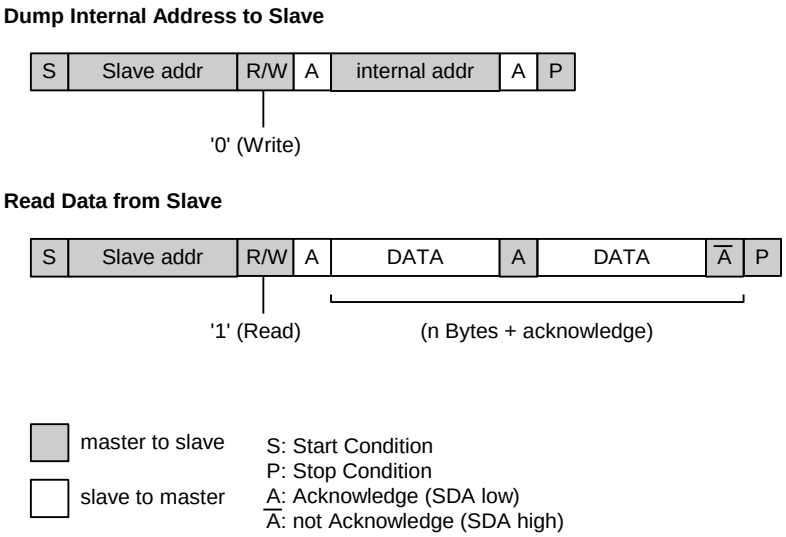


Figure 20: Two Wire Serial Interface - Read Data from Slave

6.6 Timing characteristics of the serial interface

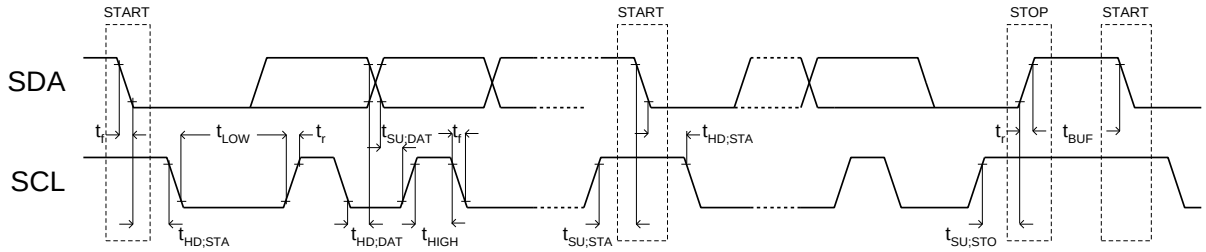


Figure 21: Definition of Timing

Parameter	Symbol	SCL Clk frequency <= 100KHz		SCL Clk frequency <= 350KHz		Unit
		Min.	Max.	Min.	Max.	
Low level input voltage: Fixed input levels	V_{IL}	-0.5(1)	1.5	-0.5(1)	$0.3V_{DD}$	V
High level input voltage: Fixed input levels	V_{IH}	3.0	(2)	$0.7V_{DD}$	(2)	V
Pulse width of spikes which must be suppressed by the input filter	t_{SP}	n/a	n/a	50	50	ns
Capacitance for each I/O pin	C_i	-	10	-	10	pF

Table 13: Two Wire Serial Interface - Characteristics of the SDA and SCL I/O Stages

Notes
(1): If Input voltage = < -0.3 Volts, then 20…100 Ohms resistor must be added in series
(2): Maximum $V_{IH} = V_{DDmax}$ + 0.5 Volt
n/a: not applicable

Parameter	Symbol	SCL Clk frequency <= 100KHz		SCL Clk frequency <= 350KHz		Unit
		Min.	Max.	Min.	Max.	
SCL clock frequency	f_{SCL}	0	100	0	350	KHz
Hold time (repeated) START condition. After this period, the first clock pulse is generated.	$t_{HD;STA}$	4.0	-	0.6	-	µs
LOW period of the SCL clock	t_{LOW}	4.7	-	1.3	-	µs
HIGH period of the SCL clock	t_{HIGH}	4.0	-	0.6	-	µs
Set-up time for a repeated START condition	$t_{SU;STA}$	4.7	-	0.6	-	µs
Data set-up time	$t_{SU;DAT}$	250	-	100	-	ns
Rise time of both SDA and SCL signals	t_r	-	1000	$20+0.1C_b$(1)	300	ns
Fall time of both SDA and SCL signals	t_f	-	300	$20+0.1C_b$(1)	300	ns
Set-up time for STOP condition	$t_{SU;STO}$	4.0	-	0.6	-	µs
Bus free time between a STOP and START condition	t_{BUF}	4.7	-	1.3	-	µs
Capacitive load for each bus line	C_b	0	400	-	400	pF
Noise margin at the LOW level for each connected device (including hysteresis)	V_{nL}	$0.1V_{DD}$	-	$0.1V_{DD}$	-	V
Noise margin at the HIGH level for each connected device (including hysteresis)	V_{nH}	$0.2V_{DD}$	-	$0.2V_{DD}$	-	V

Table 14: Two Wire Serial Interface - Characteristics of the SDA and SCL bus lines

Notes
(1): C_b = total capacitance of one bus line in pF.

6.7 Application Commands Overview

Communications between the TMC223 and a Two Wire Serial Bus Master takes place via a set of commands.

Reading commands are used to:
- Get actual status information, e.g. error flags
- Get actual position of the Stepper Motor
- Verify the right programming and configuration of the TMC223

Writing commands are used to:
- Program the OTP Memory
- Configure the TMC223 with motion parameters (e.g. max/min speed, acceleration, stepping mode, etc.)
- Provide target positions to the Stepper motor

Command Mnemonic	Function	Command Byte (hexadecimal)
GetFullStatus1	Returns complete status of the chip	0x81
GetFullStatus2	Returns actual, target and secure position	0xFC
GetOTPParam	Returns OTP parameter	0x82
GotoSecurePosition	Drives motor to secure position	0x84
HardStop	Immediate full stop	0x85
ResetPosition	Sets actual position to zero	0x86
ResetToDefault	Overwrites the chip RAM with OTP contents	0x87
RunInit	pre-programmed motion sequence to move to a mechanical limit	0x88
SetMotorParam	Sets motor parameter	0x89
SetOTP	Zaps the OTP memory	0x90
SetPosition	Programmers a target and secure position	0x8B
SetStallParam	Set Stall Detection Parameters	0x96
SoftStop	Motor stopping with deceleration phase	0x8F
TestBEMF	Outputs BEMF voltage on pin SWI (for debugging purposes only)	0x9F

Table 15: Two-Wire-Serial-Interface - Command Overview (in alphabetical order)

6.8 Command Description

There are data fields labeled as "N/A = not applicable". Within the command description tables, the contend is normally given as '1'. Data fields labeled by N/A might be reserved for later variants of the TMC223 and the content should be ignored for the TMC223.
Concerning response datagrams, the byte 0 is the slave address that is applied for addressing, where the byte 1 is the slave address that is sent back within the response data frame.

6.8.1 GetFullStatus1

This command is provided to the circuit by the Master to get a complete status of the circuit and of the stepper-motor. The parameters sent via the two wire serial bus to the Master are:

- coil peak and hold current values (Irun and Ihold)
- maximum and minimum velocities for the stepper-motor (Vmax and Vmin)
- direction of motion clockwise / counterclockwise (Shaft)
- stepping mode (StepMode) (Table 11: StepMode on page 22)
- acceleration (deceleration) for the Stepper motor (Acc)
- acceleration shape (AccShape)
- status information:
  - motion status <Motion [2:0]>
  - over current flags for coil A <OVC1> and coil B <OVC2>
  - digital supply reset <VddReset>
  - charge pump status <CPFail>
  - external switch status <ESW>
  - step loss <StepLoss>
  - electrical defect <ElDef>
  - under voltage <UV2>
  - temperature information <Tinfo>
  - temperature warning <TW>
  - temperature shutdown <TSD>
  - stall detection threshold parameters <AbsThr> and <DelThr>

GetFullStatus1 command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	GetFullStatus1	1	0	0	0	0	0	0	1

GetFullStatus1 command (Response)									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	1
1	Address	1	1	1	OTP3	OTP2	OTP1	OTP0	HW
2	Irun & Ihold	Irun (3:0)				Ihold (3:0)			
3	Vmax & Vmin	Vmax (3:0)				Vmin (3:0)			
4	Status 1	AccShape	StepMode(1:0)		Shaft	ACC(3:0)			
5	Status 2	VddReset	StepLoss	ElDef	UV2	TSD	TW	Tinfo(1:0)	
6	Status 3	Motion(2:0)			ESW	OVC1	OVC2	1	CPFail
7	N/A	1	1	1	1	1	1	1	1
8	N/A	AbsThr[3:0]				DelThr[3:0]			

Note: Slave Address is the address sent to device, but it will not be sent back - Address is sent back
Note: N/A = not applicable

6.8.2 GetFullStatus2

This command is provided to the circuit by the Master to get the actual position of the stepper-motor. The position is provided by the circuit in 16-bit format, with the 3 LSBs at '0' when in half stepping mode (StepMode = "00"). Furthermore programmed target position and secure position are also provided.

GetFullStatus2 command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	GetFullStatus2	1	1	1	1	1	1	0	0

GetFullStatus2 command (Response)									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	1
1	Address	1	1	1	OTP3	OTP2	OTP1	OTP0	HW
2	Actual Position 1	ActPos(15:8)							
3	Actual Position 2	ActPos(7:0)							
4	Target Position 1	TagPos(15:8)							
5	Target Position 2	TagPos(7:0)							
6	Secure Position	SecPos(7:0)							
7	Stall Detection / Secure Position	FS2StallEn[2:0]			1	DC100	SecPos(10:8)		
8	Stall Detection	AbsStall	DelStallLo	DelStallHi	MinSamples[2:0]			DC100StEn	PWMJEn

Note: Slave Address is the address sent to device, but it will not be sent back - Address is sent back
Note: N/A = not applicable

6.8.3 GetOTPParam

This command is provided to the circuit by the master to read the content of the OTP Memory. For more information refer to Table 9: OTP Memory Structure on page 22.

GetOTPParam command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	GetOTPParam	1	0	0	0	0	0	1	0

GetOTPParam command (Response)									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	1
1	OTP byte 0	OTP@0x00							
2	OTP byte 1	OTP@0x01							
3	OTP byte 2	OTP@0x02							
4	OTP byte 3	OTP@0x03							
5	OTP byte 4	OTP@0x04							
6	OTP byte 5	OTP@0x05							
7	OTP byte 6	OTP@0x06							
8	OTP byte 7	OTP@0x07							

Note: Slave Address is the address sent to device, but it will not be sent back - Address is sent back

6.8.4 GotoSecurePosition

This command is provided by the Master to one or all the stepper-motors to move to the secure position SecPos[10:0]. It can also be triggered at the end of a RunInit initialization phase. If SecPos[10:0] equals 0x400 (the most negative decimal value of -1024) the secure position is disabled and the GotoSecurePosition command is ignored.

GotoSecurePosition command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	GotoSecurePosition	1	0	0	0	0	1	0	0

6.8.5 HardStop

This command is internally triggered when an electrical problem is detected in one or both coils, leading to switch off the H-bridges. If this problem is detected while the motor is moving, the <StepLoss> flag is raised allowing to warn the Master that steps may have been lost at the next GetFullStatus1 command. A HardStop command can also be issued by the Master for some safety reasons.

HardStop command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	HardStop	1	0	0	0	0	1	0	1

6.8.6 ResetPosition

This command is provided to the circuit by the Master to reset ActPos and TagPos registers, in order to allow for an initialization of the stepper-motor position.

Hint: This command is ignored during motion. It has no effect during motion. The Status Flags (section 5.2.2, page 21) named 'Motion Status' indicate if the motor is at rest (velocity=0).

ResetPosition command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	ResetPosition	1	0	0	0	0	1	1	0

6.8.7 ResetToDefault

This command is provided to the circuit by the Master in order to reset the whole slave node into the initial state. ResetToDefault will for instance **overload the RAM** with the reset state of the register parameters. This is another way for the Master to initialize a slave node in case of emergency, or simply to refresh the RAM content.

Note: ActPos is not modified by a ResetToDefault command, and it's value is copied into TagPos register in order to avoid an attempt to position the motor to '0'.

ResetToDefault command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	ResetToDefault	1	0	0	0	0	1	1	1

6.8.8 RunInit

This command is provided to the circuit by the Master in order to initialize positioning of the motor by seeking the zero (or reference) position. Refer to 5.1.11 Reference Search / Position initialization on page 15. It leads to a sequence of the following commands:

- SetMotorParam(Vmax, Vmin);
- SetPosition(Pos1);
- SetMotorParam(Vmin, Vmin);
- SetPosition(Pos2);
- ResetPosition
- GotoSecurePosition

Once the RunInit command is started it can not be interrupted by any other command except when a condition occurs which leads to a motor shutdown (See 5.1.10 Motor Shutdown Management) or a HardStop command is received. If SecPos[10:0] equals 0x400 (the most negative decimal value of -1024) the final travel to the secure position is omitted.

The master has to ensure that the target position of the first motion is **not** equal to the actual position of the stepper motor and that the target positions of the first and second motion are different, too. This is very important otherwise the circuit goes into a deadlock state. Once the circuit is in deadlock state only a HardStop command followed by a GetFullStatus1 command will cause the circuit to leave the deadlock state.

RunInit command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	RunInit	1	0	0	0	1	0	0	0
2	N/A	1	1	1	1	1	1	1	1
3	N/A	1	1	1	1	1	1	1	1
4	Vmax Vmin	Vmax(3:0)				Vmin(3:0)			
5	Position1 byte 1	TagPos1(15:8)							
6	Position1 byte 2	TagPos1(7:0)							
7	Position2 byte 1	TagPos2(15:8)							
8	Position2 byte 2	TagPos2(7:0)							

Note: N/A = not applicable

6.8.9 SetMotorParam

This command is provided to the circuit by the Master to set the values for the following stepper motor parameters in RAM:

- coil peak current value (Irun)
- coil hold current value (Ihold)
- maximum velocity for the Stepper-motor (Vmax)
- minimum velocity for the Stepper-motor (Vmin)
- acceleration shape (AccShape)
- stepping mode (StepMode)
- direction of the Stepper-motor motion (Shaft)
- acceleration (deceleration) for the Stepper-motor (Acc)
- secure position for the Stepper-motor (SecPos)
- PWM frequency selection (PWMfreq)
- PWM jitter enable (PWMJEn) for low EMI

If SecPos[10:0] is set to 0x400 (the most negative decimal value of –1024) the secure position is disabled and the GotoSecurePosition command is ignored.

SetMotorParam command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	SetMotorParam	1	0	0	0	1	0	0	1
2	N/A	1	1	1	1	1	1	1	1
3	N/A	1	1	1	1	1	1	1	1
4	Irun & I hold	Irun(3:0)				Ihold(3:0)			
5	Vmax & Vmin	Vmax(3:0)				Vmin(3:0)			
6	Status	SecPos(10:8)			Shaft	Acc(3:0)			
7	SecurePos	SecPos(7:0)							
8	StepMode etc.	1	PWMfreq	1	AccShape	StepMode[1:0]		1	PWMJEn

Note: N/A = not applicable

6.8.10 SetStallParam

This command sets the relevant parameters for the sensorless stall detection. A description with an example, how to find a set of parameters for stall detection is given in section 7, page 39.

SetStallParam command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	SetStallParam	1	0	0	1	0	1	1	0
2	N/A	1	1	1	1	1	1	1	1
3	N/A	1	1	1	1	1	1	1	1
4	Irun & I hold	Irun(3:0)				Ihold(3:0)			
5	Vmax & Vmin	Vmax(3:0)				Vmin(3:0)			
6		MinSamples(2:0)			Shaft	Acc(3:0)			
7		AbsThr(3:0)				DelThr(3:0)			
8	StepMode	FS2StallEN(2:0)			AccShape	StepMode[1:0]		DC100En	PWMJEn

Note: The PWMfreq selection bit set via SetMotorParam has to be taken into account for MinSamples

A valid parameter set of stall detection parameters depends on the given type of motor and the motion settings for the motor (Irun, Ihold, Vmax, Vmin, Acc, …).

- timing parameter (MinSamples) has to be chosen for a given velocity (Vmax) and PWM frequency
- blanking parameter (FS2StallEN) has to be chosen depending on motor resonance characteristic
- absolute threshold (AbsThr) for detection of stall by mechanical limit
- relative threshold (DelThr) for motion monitoring purposes
- automatic masking of stall detection on 100% PWM by switch (DC100En)

The threshold parameters AbsThr resp. DelThr has to be chosen depending on the Back EMF constant of the motor.

6.8.11 SetOTPParam

This command is provided to the circuit by the Master in order to zap the OTP memory. The OTPA address (OTPA) addresses the OTP word (please refer section 5.2.3, page 22) within the OTP Memory structure. The Pbit byte represents the bit pattern to be programmed, where a one programs an un-programmed OTP bit. For example, if one wants to OTP the defaults to Irun := 0xD and Ihold = 0x5, one has to execute the SetOTPParam with OTPA = 0x03 and Pbit := 0xD5. Those OTP bits that are un-programmed can be programmed to '1' by corresponding Pbit chosen as '1' . For OTP the supply voltage Vbat has to be within the valid range specified as VbbOTP within Table 27, page 50.

SetOTPParam command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	SetOTPParam	1	0	0	1	0	0	0	0
2	N/A	1	1	1	1	1	1	1	1
3	N/A	1	1	1	1	1	1	1	1
4	OTP Address	1	1	1	1	1	OTPA(2:0)		
5	Pbit	Pbit(7:0)							

Note: N/A = not applicable

6.8.12 SetPosition

This command is provided to the circuit by the Master to drive the motor to a given position relative to the zero position, defined in number of half or micro steps, according to StepMode[1:0] value.
SetPosition will not be performed if one of the following flags is set to one:

- temperature shutdown <TSD>
- under voltage <UV2>
- step loss <StepLoss>
- electrical defect <ElDef>

SetPosition command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	SetPosition	1	0	0	0	1	0	1	1
2	N/A	1	1	1	1	1	1	1	1
3	N/A	1	1	1	1	1	1	1	1
4	Position byte1	TagPos(15:8)							
5	Position byte2	TagPos(7:0)							

Note: N/A = not applicable

6.8.13 SoftStop

If a SoftStop command occurs during a motion of the Stepper motor, it provokes an immediate deceleration to Vmin followed by a stop, regardless of the position reached. This command occurs in the following cases:

- The chip temperature rises above the Thermal shutdown threshold.
- The Master requests a SoftStop.

SoftStop command									
Byte	Content	Structure							
		bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	Slave Address	1	1	OTP3	OTP2	OTP1	OTP0	HW	0
1	SoftStop	1	0	0	0	1	1	1	1

6.9 Positioning Task Example

The TMC223 has to perform a positioning task, where the actual position of the stepper motor is unknown. The desired target position is 3000 µsteps away from position 0. See figure below.

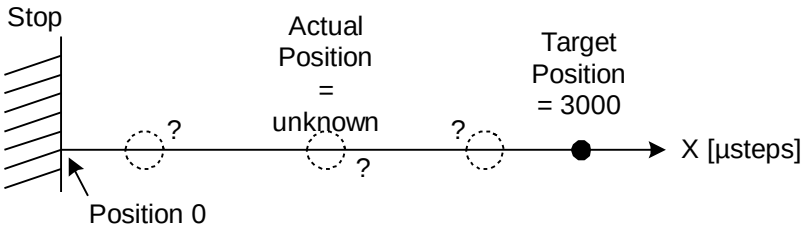


Figure 22: Positioning Example: Initial situation

The following sequence of commands has to be sent to the slave in order to complete the scenario described above (assumed after power on):

GetFullStatus1
The command is used to read the current status of the TMC223. Electrical or environmental problems will be reported, furthermore the circuit leaves the shutdown state and is ready for action. See 6.8.1 GetFullStatus1 on page 31.

GetFullStatus2
The circuit will enter a deadlock state if the actual position corresponds to the first target position of the RunInit command. This command is used to read the actual position. The master must take care that both positions are containing different values. For deadlock conditions see 5.1.11 Reference Search / Position initialization on page 15.

SetMotorParam
In order to drive the stepper motor with a desired motion parameters like torque, velocity, aso. the SetMotorParam command must issued. See 6.8.9 SetMotorParam on page 35.

RunInit
Hence the actual position is unknown, a position initialization has to be performed. The first motion must drive the stepper motor into the stop for sure. The second motion is a very short motion to bring the motor out of the stop. The actual position is then set to zero automatically after the second motion is finished. See 6.8.8 RunInit on page 34.

After reference search the situation is as depicted in the figure below. Actual position of the stepper motor corresponds to zero, the target position is 3000 µsteps away from the actual position.

Figure 23: Positioning Example: Situation after reference search

Now the positioning command SetPosition can be issued in order to drive the stepper motor to the desired position.

SetPosition
This command will cause the stepper motor to move to the desired target position. See 6.8.12 SetPosition on page 37. After the motion has been finished, the situation is as depicted in the figure below.

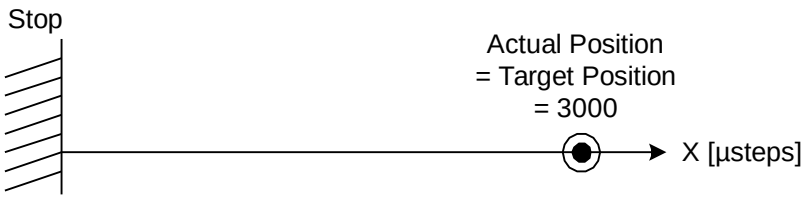


Figure 24: Positioning Example: Motion finished

Afterwards the actual status and position can be verified by GetFullStatus1 and GetFullStatus2 commands. The master can check if a problem, caused by electrical or temperature problems, occurred. Furthermore the actual position is read.

7 Sensorless Stall Detection

The sensorless stall detection of the TMC223 bases on the measurement of the Back EMF (electro magnetic force) voltage (U_{BEMF}) induced by the motion of the rotor. The amplitude of the Back EMF voltage is proportional to the speed of the rotor. The Back EMF constant of a stepper motor can determined by turning the axis of a non-connected motor measuring the amplitude and frequency of generated voltage at one phase of the motor.

If a motor stalls, the velocity of its rotor varies and the amplitude of the BEMF varies. The TMC223 detects a stall or disturbances of the motion measuring the variations of the Back EMF.

From physical point of view, a stepper motor is an anharmonic oscillator causing disturbances of the rotors motion. Although micro stepping cause low resonances - the measurement of the BEMF has to be masked to separate signals caused by resonance effects from those caused by stall.

7.1 Stall Detection Flags

A Stall is signalised by dedicated stall detection flags (Table 16: Stall Detection Flags). The stall detection flag named "Stall" is the logical function DelStallLo OR DelStallHi OR AbsStall.

The DelStallLo flag is set to one if the measured and averaged $U_{BEMF\_AVARAGE}$ becomes below the threshold $U_{BEMF} < (U_{BEMF\_AVARAGE} - DelThr)$. This occurs when breaking the motor axis in a way that tries to decelerate the rotor - when the stepper acts a motor. The DelStallHi is set to one if the averaged $U_{BEMF\_AVARAGE}$ becomes above the threshold ($U_{BEMF\_AVARAGE} - DelThr$) where $U_{BEMF\_AVARAGE}$. This occurs when a pulling torque is applied to the motor axis that tries to accelerate the rotor - when the stepper acts as a generator.

The AbsStall flag becomes one if the U_{BEMF} comes below the limit defined by AbsThr. This occurs when a motor come at rest at a limit. A stepper motor might jump back a multiple of for full steps when reaching a limit. This generates BEMF signals that might trigger the DelStallHi flag - depending on the parameterizing.

Hint: If the sensorless stall detection stops the motor, the StepLoss status bit be set in addition.

DelStallLo	DelStallHi	AbsStall	Stall	Condition
1	0	0	1	$U_{BEMF} < (U_{BEMF\_AVARAGE} - DelThr)$
0	1	0	1	$U_{BEMF} > (U_{BEMF\_AVARAGE} + DelThr)$
0	0	1	1	$U_{BEMF} < AbsThr$

Table 16: Stall Detection Flags

7.2 Stall Detection Parameters

The timing parameter MinSamples has to be set depending on the PWM frequency and the chosen velocity to the largest value below duration of on full step.

The stall detection blanking parameter FS2StallEN blanks variations of the BEMF due to oscillations that follows an acceleration phase that can cause faulty detection of a stall.

The parameter AbsThr represents an absolute voltage level of the BEMF. The parameter DelThr represents the difference (delta) threshold for processing of the BEMF based on the averaged BEMF voltage.

In addition, there is one important switch named **DC100En** that should be set to one if the PWM come close to 100%. With this, the stall detection is masked for PWM of 100%. This is because a PWM of 100% might faulty trigger the stall detection flags. On the other hand, a speed where the PWM duty cycle comes close to 100% should be avoided. This is because a PWM duty cycle of 100% indicated that the motor becomes within the zone where the torque decreases because the target current can not be reached for that speed.

MinSamples	Delay Time [µs]	
	PWMfreq = 0 (PWM frequency typ 22.8 kHz)	PWMfreq = 1 (PWM frequency typ 45.6 kHz)
0x0	87	43
0x1	130	65
0x2	174	87
0x3	217	109
0x4	261	130
0x5	304	152
0x6	348	174
0x7	391	196

Table 17: Stall Detection Timing Parameter MinSamples

FS2StallEN	Full Steps
0x0	0
0x1	1
0x2	2
0x3	3
0x4	4
0x5	5
0x6	6
0x7	7

Table 18: Stall Detection Blanking Parameter FS2StallEN

AbsThr	AbsThrLevel[V]
0x0	Disable
0x1	0.5
0x2	1.0
0x3	1.5
0x4	2.0
0x5	2.5
0x6	3.0
0x7	3.5
0x8	4.0
0x9	4.5
0xA	5.0
0xB	5.5
0xC	6.0
0xD	6.5
0xE	7.0
0xF	7.5

Table 19: Stall Detection Threshold Parameter AbsThr

DelThr	DelThrLevel[V]
0x0	Disable
0x1	0.25
0x2	0.50
0x3	0.75
0x4	1.00
0x5	1.25
0x6	1.50
0x7	1.75
0x8	2.00
0x9	2.25
0xA	2.50
0xB	2.75
0xC	3.00
0xD	3.25
0xE	3.50
0xF	3.75

Table 20: Stall Detection Threshold Parameter AbsThr

7.3 Example of Stall Detection Parameter Setting

The following set of parameters is given for an exemplary application. One should keep in mind that the parameterizing of sensorless stall detection might need some measurements and its successful parameterization also depends on the used type of motor and the given application environment.

Parameter	Value	Comment	Action
IRun	15	800 mA (peak current)	
IHold	9	283 mA (peak current)	
Vmin	1	8 FS/s	
Vmax	8	V[FS/s] = 303 FS/s ⇔ **206.3 µs/microstep**	=> Set MinSamples to 2 corresponding to 175.4 µs
Acc	6	11409 FS/s^2	
AccShape	0	ramping with Acc	
Shaft	0	cw (depending on motor)	
StepMode	3	1/16 µStepping	
PWMfreq	0	22.8 kHz	
Motor Parameter :			
K_{BEMF}	10,8 mV/FS/s	Back EMF constant of the motor	
U_{BEMF}	4,64 V	peak voltage **U_{BEMF}** = **K_{BEMF} * V[FS/s]**	=> Set AbsThr to 0x9 corresponding to 4.5V
Blanking Parameter :			
FS2StallEN	7	less sensitive to oscillations	try smaller values
DC100En	1	set to mask stall detection for 100% PWM	
PWMJEn	0	should be disabled for stall detection	

Table 21 : Example of Stall Detection Parameter Set

8 Frequently Asked Questions

8.1 Using the bus interface

Q: How many devices can be operated on the same bus?
A: 32 devices can be discriminated by means of the physical address. However, it depends on some factors if this high number really makes sense. First of all it has to be checked if each device can be serviced under any circumstances in the maximum allowed time taking the bus speed and the individual real-time requirements of each device into account. Second, the idea of reserving address 0 for OTP physical address programming during system installation and defective parts replacement reduces the number to 30.

Q: How to program the OTP physical address bits of a device if there are more devices connected to the same bus?
A: The problem here is that all new devices are shipped with the OTP physical address bits set to zero making it difficult to address just one device with the SetOTPParam command. Use HW input as chip select line to address just one device by SetOTPParam. If this is impractible since the HW input is hardwired or not controllable for any other reason the only alternative is to assemble and program one device after the other. I.e., assemble only first device and program the desired non-zero address, then assemble the second device and program the desired non-zero address, and so on until all devices are assembled and programmed. This is also a good service concept when replacing defective devices in the field: The idea is that all devices are programmed to different non-zero physical addresses at production/installation time. Once a defective device is being replaced the replacement part can easily be addressed by SetOTPParam since it is the only part with physical address zero.

8.2 General problems when getting started

Q: What is the meaning of ElDef?
A: The ElDef flag ('Electrical Defect') is the logical ORing of the OVC1 and OVC2 flags. OVC1 is set to one in case of an overcurrent (coil short) or open load condition (selected coil current is not reached) for coil A. OVC2 is the equivalent for coil B.

Q: What could be the reason for ElDef / OVC1 / OVC2 being set to one?
A: There are a number of possible causes:

- Motor not connected (→ open load)
- Connected motor has shorted coils (→ overcurrent) or broken coils (→ open load)
- Motor coils connected to the wrong device pins
- Selected coil current can not be reached (→ open load) due to high coil impedance or low supply voltage. Solution: Select a lower coil run/hold current or rise the supply voltage. Generally: the calculated voltage required to reach a desired coil current at a given coil resistance (V = I • R) must be significantly lower than actual supply voltage due to the coil inductivity.

Q: Should the external switch be normally closed or open when the reference position is hit?
A: The SWI input resp. the ESW flag have neither effect on any internal state machine nor on command processing, even not on the RunInit command. ESW must be polled by software using GetFullStatus1 command. The software can simply be adapted to whatever state the switch is in when the reference position is hit, i.e. closed or open.

8.3 Using the device

Q: What is the meaning of the 'Shaft' bit?
A: The Shaft bit determines the rotating direction of the motor, i.e. clockwise or counter-clockwise rotation.

Q: How to generate an interrupt when the target position is reached?
A: This is not possible. The device hasn't any interrupt output at all. Just poll ActPos or Motion[2:0] using an appropriate command.

Q: How can I ensure that I always get consistent data for ActPos and ESW?
A: There isn't a single command to read both ActPos and ESW simultaneously. GetFullStatus1 will read ESW whereas GetFullStatus2 will read ActPos. Thus it is not possible to read consistent values as long as a motion is in progress.

Q: How to specify a second target position to go to immediately after a first target position has been reached?
A: This is possible using the RunInit command. Note, that after the second target position has been reached the internal position counter ActPos is reset to zero.

Q: Is it possible to change Vmax on-the-fly?
A: Yes, it is, if the new velocity is in the same group as the old one (see Vmax Parameters). Otherwise correct positioning is not ensured anymore. Vmax values are divided into four groups:

- group A: Vmax index = 0
- group B: Vmax index = 1, 2, 3, 4 ,5 or 6
- group C: Vmax index = 7, 8, 9, 10 ,11 or 12
- group D: Vmax index = 13, 14 or 15

Q: Is it possible to change the stepping mode on-the-fly?
A: Yes, it is possible and it has immediate effect on the current motion.

Q: How to operate in continuous velocity mode rather than positioning (ramp) mode?
A: There is no velocity mode. The device was designed primarily for positioning tasks so for each motion there has to be specified a target position by the respective command. However, velocity mode can be emulated by repeating the following two commands again and again:

- Read ActPos using GetFullStatus2 command
- Set lower 16 bits of [ActPos+32767] as the next target position using SetPosition command

For real continuous motion this sequence has to be repeated before the current target position has been reached.

Q: Which units, formats and ranges does position information have?
A: All 16-bit position data fields in commands and responses are coded in two's complement format with bit 0 representing 1/16 micro-steps. Hence a position range of –32768…+32767 in units of 1/16 micro-steps is covered regardless of the selected stepping mode (1/2, 1/4, 1/8 or 1/16 micro-stepping). The difference between the stepping modes is the resolution resp. the position of the LSB in the 16-bit position data field: it's bit 0 for 1/16, bit 1 for 1/8, bit 2 for 1/4 and bit 3 for 1/2 micro-stepping. The position range can be regarded as a circle since position –32768 is just 1/16 micro-step away from position +32767. The device will always take the shortest way from the current to the target position, i.e., if the current position is +32767 and the target position is –32768 just 1/16 micro-step will be executed. 65535 1/16 micro-steps in the opposite direction can be achieved for example by two consecutive SetPosition commands with target positions 0 and –32768.

The 11-bit secure position data field can be treated as the upper 11 MSBs of the 16-bit position data fields described above with the 5 LSBs hardwired to zero. Hence it covers the same position range with a reduced resolution: The position range is –1024…+1023 in units of two full-steps.

8.4 Finding the reference position

Q: How do I find a reference position?
A: The recommended way is to use the RunInit command. Two motions are specified through RunInit. The first motion is to reach the mechanical stop. Its target position should be specified far away enough so that the mechanical stop will be reached from any possible starting position. There is no internal stall detection so that at the end of the first motion the step motor will bounce against the mechanical stop loosing steps until the internal target position is reached. The second motion then can be used either to drive in the opposite direction out of the mechanical stop right into the reference position which is a known number of steps away from the mechanical stop. Or the second motion can slowly drive a few steps in the same direction against the mechanical stop to compensate for the bouncing of the faster first motion and stop as close to the mechanical stop as possible.

Q: Can the SWI input help in finding a reference position?
Not directly. The current state of the SWI input is reflected by the ESW flag which can only be polled using the command GetFullStatus1. The SWI input resp. the ESW flag have neither influence on any internal state machine nor on command processing. The recommended way to find a reference position is to use the RunInit command. Alternatively one could initiate a long distance motion at very low speed using SetPosition and then poll ESW as frequently as possible to be able to stop the motion using HardStop right in the moment the switch position is reached. Then one would reset the internal position counters ActPos and TagPos using the ResetPosition command.

Q: What is the logic of the ESW flag?
A: The ESW flag reflects the state of the SWI input. ESW is set to one if SWI is high or low, i.e. pulled to VBAT or to GND. ESW is set to zero if SWI is left open, i.e. floating. ESW is updated synchronously with ActPos every 1024 µs.

Q: Is it possible to swap the logic of the ESW flag?
A: No, it's not. Actually this is not necessary since the ESW flag must be polled and evaluated by software anyway. The state of ESW has neither effect on any internal state machine nor on command processing.

Q: What else is important for the RunInit command?
A: The first target position of RunInit must be different from the current position before sending RunInit and the second target position must be different from the first one. Otherwise a deadlock situation can occur. During execution of RunInit only Get… commands should be sent to the device.

Q: Does the second motion of RunInit stop when the ESW flag changes, or does it continue into the mechanical stop?
A: Neither nor. The SWI input resp. the ESW flag have neither effect on any internal state machine nor on command processing, i.e. the RunInit command is not influenced by SWI / ESW. The same is true for the mechanical stop: as there isn't any internal stall detection the RunInit command can not detect a mechanical stop. When the mechanical stop is hit the first or second motion of RunInit (or the motion of any other motion command) will be continued until the internal position counter ActPos has reached the target position of this motion. This results in the motor bouncing against the mechanical stop and loosing steps. The intention of the second motion of RunInit is to drive out of the mechanical stop (reached by the first motion) to the desired reference position at a known distance from the mechanical stop or to drive slowly against the mechanical stop again to compensate for the bouncing of the first motion and to come to a standstill as close to the mechanical stop as possible.

Q: Does RunInit reset the position?
A: Yes, it does. After the second motion of RunInit has been finished the internal position counter ActPos is reset to zero.

9 Package Outline

9.1 SOIC-20

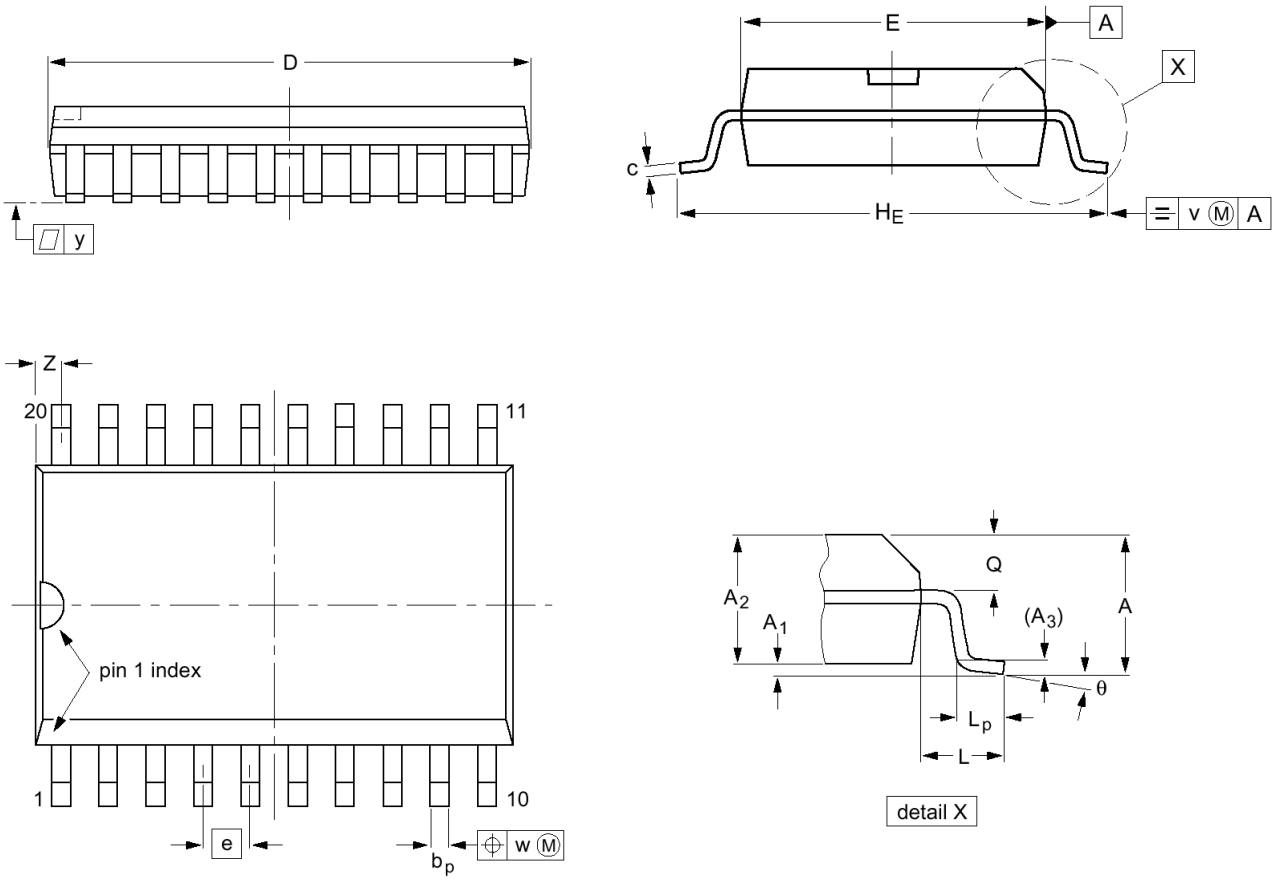


Figure 25: Package Outline SOIC-20

UNIT	A max.	A_1	A_2	A_3	b_p	c	$D^{(1)}$	$E^{(1)}$	e	H_E	L	L_p	Q	v	w	y	$Z^{(1)}$	theta
mm	2.65	0.30 0.10	2.45 2.25	0.25	0.49 0.36	0.32 0.23	13.0 12.6	7.6 7.4	1.27	10.65 10.00	1.4	1.1 0.4	1.1 1.0	0.25	0.25	0.1	0.9 0.4	8° 0°
inches	0.10	0.012 0.004	0.096 0.089	0.01	0.019 0.014	0.013 0.009	0.51 0.49	0.30 0.29	0.050	0.419 0.394	0.055	0.043 0.016	0.043 0.039	0.01	0.01	0.004	0.035 0.016	

Table 22: SOIC-20 Mechanical Data

Note: inch dimensions are derived from the original mm dimensions

9.2 QFN32

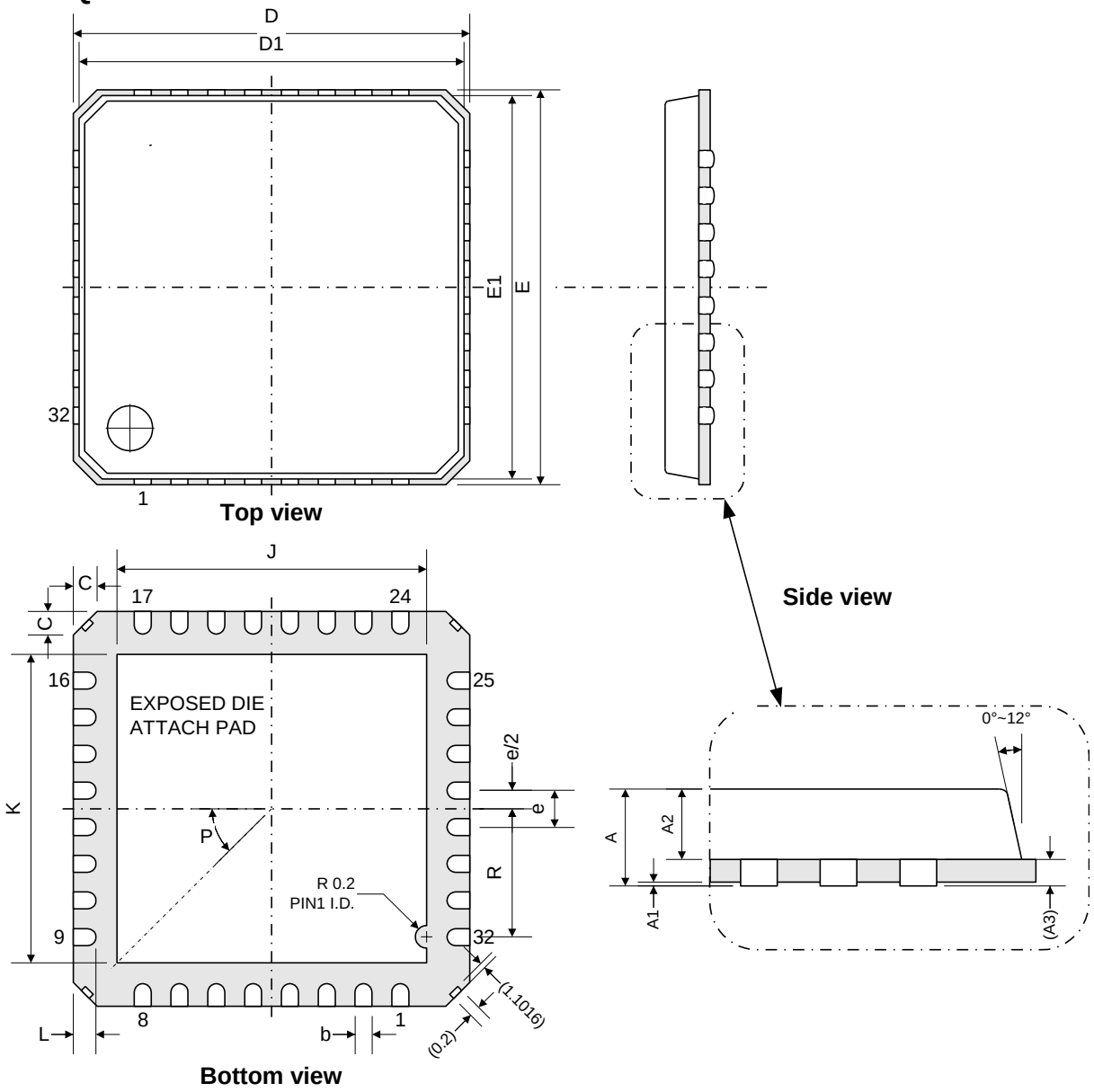


Figure 26: Package Outline QFN32

REF	A	A1	A2	A3	b	C	D	D1	E	E1	e	J	K	L	P	R
MIN	0.80	0.00	0.576		0.25	0.24						5.37	5.37	0.35		2.185
NOM		0.02	0.615	0.203	0.3	0.42	7	6.75	7	6.75	0.65	5.47	5.47	0.4	45	
MAX	0.90	0.05	0.654		0.35	0.6						5.57	5.57	0.45		2.385
Unit	mm	mm	mm	mm	mm	mm	mm	mm	mm	mm	mm	mm	mm	mm	deg.	mm

Hint: The exposed die attached pad is electrical ground. This pad should be connected to ground or can be left open. It is recommended to connect it to ground for cooling.

10 Package Thermal Resistance

10.1 SOIC-20 Package

The junction case thermal resistance is 28°C/W, leading to a junction ambient thermal resistance of 63°C/W, with the PCB ground plane layout condition given in the figure below and with

- PCB thickness = 1.6mm
- 1 layer
- Copper thickness = 35µm

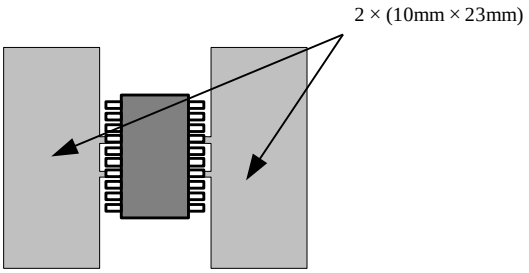


Figure 27: Layout consideration

11 Electrical Characteristics

11.1 Absolute Maximum Ratings

Parameter		Min	Max	Unit
Vbat	Supply Voltage	-0.3	+35	V
Tamb	Ambient temperature under bias (*)	-50	+150	°C
Tst	Storage temperature	-55	+160	°C
Vesd (**)	Electrostatic discharge voltage on pins	-2	+2	kV

Table 23: Absolute Maximum Ratings

(*) The circuit functionality is not guaranteed
(**) Human body model (100pF via 1.5 KΩ)

11.2 Operating Ranges

Parameter			Min	Max	Unit
Vbat	Supply Voltage (Vbb)		+8	+29	V
Top	Operating temperature range	Vbat <= 18V	-40	+125	°C
		Vbat <= 29V	-40	+85	°C

Table 24: Operating Ranges

11.3 DC Parameters

Motor Driver							
Symbol	**Pin(s)**	**Parameter**	**Test condition**	**Min**	**Typ**	**Max**	**Unit**
IMSmax Peak	OA1 OA2 OB1 OB2	Max current through motor coil in normal operation			800		mA
IMSmax RMS		Max RMS current through coil in normal operation			570		mA
RDSon		On resistance for each pin (including bond wire)	To be confirmed by characterization			1	Ω
IMSL		Leakage current	HZ Mode, 0V < V(pin) < Vbb	-50		+50	µA

Table 25: DC Parameters Motor Driver

Thermal Warning and shutdown							
Symbol	**Pin(s)**	**Parameter**	**Test condition**	**Min**	**Typ**	**Max**	**Unit**
Ttw		Thermal Warning		138	145	152	°C
Ttsd (*)		Thermal Shutdown			Ttw + 10		°C
Tlow		Low Temperature Warning			Ttw - 155		°C

Table 26: DC Parameters Thermal Warning and shutdown

(*) NO more than 100 cumulated hours in life time above Ttsd

Supply and Voltage regulator							
Symbol	**Pin(s)**	**Parameter**	**Test condition**	**Min**	**Typ**	**Max**	**Unit**
Vbb	VBB	Nominal operating supply range (*)		6.5		18	V
VbbT85		Nominal operating supply range (*) for temperature < 85°C		6.5		29	V
VbbOTP		Supply Voltage for OTP zapping		8.5		9.5	V
UV1		Low voltage high threshold		8.8	9.4	9.8	V
UV2		Stop voltage low threshold		8.1	8.5	8.9	V
Ibat		Total current consumption	Unloaded Outputs		10		mA
Vdd	VDD	Internal regulated output (**)	8V < Vbb < 18V Cload = 1µF (+100nF cer.)	4.75	5	5.25	V
IddStop		Digital current consumption	Vbb < UV2		2		mA
VddReset		Digital supply reset level (***)				4.4	V
IddLim		Current limitation	Pin shorted to ground			40	mA

Table 27: DC Parameters Supply and Voltage regulator

(*) Communication over serial bus is operating. Motor driver is disabled when Vbb < UV2.
(**) Pin VDD must not be used for any external supply.
(***) The RAM content will not be altered above this voltage

Switch Input and hardwired address input HW							
Symbol	**Pin(s)**	**Parameter**	**Test condition**	**Min**	**Typ**	**Max**	**Unit**
Rt_OFF	SWI HW	Switch OFF resistance (*)	Switch to GND or Vbat	10			kΩ
Rt_ON		Switch ON resistance (*)				2	kΩ
Vbb_sw		Vbb range for guaranteed operation of SWI and HW		6		18	V
Vmax_sw		Maximum Voltage	T < 1s			40	V
Ilim_sw		Current limitation	Short to GND or Vbat		30		mA

Table 28: DC Parameters Switch Input and hardwired address input

(*) External resistance value seen from pin SWI or HW, including 1kΩ series resistor

Test pin							
Symbol	**Pin(s)**	**Parameter**	**Test condition**	**Min**	**Typ**	**Max**	**Unit**
Vhigh	TST	Input level high		0.7			Vdd
Vlow		Input level low				0.3	Vdd
HWhyst		Hysteresis		0.075			Vdd

Table 29: DC Parameters Test pin

Charge Pump							
Symbol	**Pin(s)**	**Parameter**	**Test condition**	**Min**	**Typ**	**Max**	**Unit**
Vcp	VCP	Output Voltage	Vbb > 15V	Vbb+10	Vbb+12.5	Vbb+15	V
			Vbb > 8V	Vbb+5.8			V
Cbuffer		External Buffer Capacitor		220		470	nF
Cpump	CPP CPN	External pump Capacitor		220		470	nF

Table 30: DC Parameters Charge Pump

11.4 AC Parameters

Power-Up							
Symbol	**Pin(s)**	**Parameter**	**Test condition**	**Min**	**Typ**	**Max**	**Unit**
Tpu		Power-Up time				10	ms

Table 31: AC Parameters Power-Up

Switch Input and hardwired address input HW							
Symbol	**Pin(s)**	**Parameter**	**Test condition**	**Min**	**Typ**	**Max**	**Unit**
Tsw	SWI HW	Scan Pulse Period		921	1024	1127	µs
Tsw_on		Scan Pulse Duration			1/16		Tsw

Table 32: AC Parameters Switch Input and hardwired address input

Motor Driver							
Symbol	**Pin(s)**	**Parameter**	**Test condition**	**Min**	**Typ**	**Max**	**Unit**
Fpwm	OA1 OA2 OB1 OB2	PWM frequency (PWMfreq=0)		20.6	22.8	25.0	kHz
		PWM frequency (PWMfreq=1)		41.2	45.6	50.0	kHz
JTpwm		PWM frequency jitter modulation depth (PWMjen=1)			10%		
Tbrise		Turn-On transient time	Between 10% and 90%		350		ns
Tbfall		Turn-Off transient time			250		ns

Table 33: AC Parameters Motor Driver

Revision History

Version	Date	Comments
1.00	May 3, 2007 (LL)	Initial version, based on TMC222 datasheet (v. 1.06 / March 15, 2007)
1.01	December 16, 2008 (LL)	Section 7.2 Stall Detection Parameters, Table 21, page 42 : StepMode = 3 (was binary 11) and comment vmax : "V[FS/s] = 303 FS/s ⇔ 206.3 µs/microstep" corrected.
1.02	March 2nd, 2009 (LL)	GetActualPos (part of GetFullStatus2) corrected in Table 8: Priority Encoder, page 19 (~~GetActualPos~~ => GetFullStatus2)
1.03	November 25, 2009 (LL)	Hint concerning connecting exposed ground for cooling added (section 9.2, page 47); operating supply voltage range (VbbT85 parameter added for temperature < 85°C (VbbT85), Table 27, page 50)
1.04	December 11, 2009 (LL)	Hint concerning sensorless stall detection stop and <StepLoss> flag added (section 7.1, page 39).
1.05	March 7, 2011 (LL)	Hints concerning usage of HW pin and SWI pin added section 5.1.9 External Switch, page 13, section 6.3 Physical Address of the circuit, page 27;
1.06	August 21, 2017 (LL)	Hold current corrected in Table 12: Irun / Ihold Settings on page 24; hint added that the peak current of TMC223 is 0mA for setting Ihold = 0xF.

Please refer to www.trinamic.com for updated data sheets and application notes on this product and on other products.

The TMCtechLIB CD-ROM including data sheets, application notes, schematics of evaluation boards, software of evaluation boards, source code examples, parameter calculation spreadsheets, tools, and more is available from TRINAMIC Motion Control GmbH & Co. KG by request to info@trinamic.com