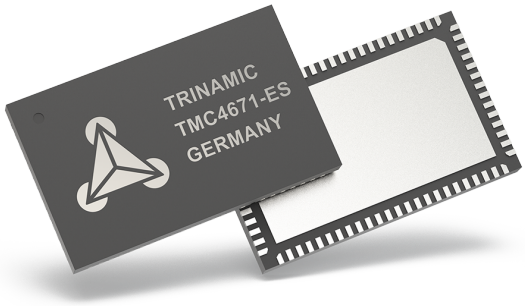
TMC4671 Datasheet

IC Version V1.00 | Document Revision V1.03 • 2018-Sept-06

The TMC4671 is a fully integrated servo controller, providing Field Oriented Control for BLDC/PMSM and 2-phase Stepper Motors as well as DC motors and voice coils. All control functions are implemented in hardware. Integrated ADCs, position sensor interfaces, position interpolators, enable a fully functional servo controller for a wide range of servo applications.



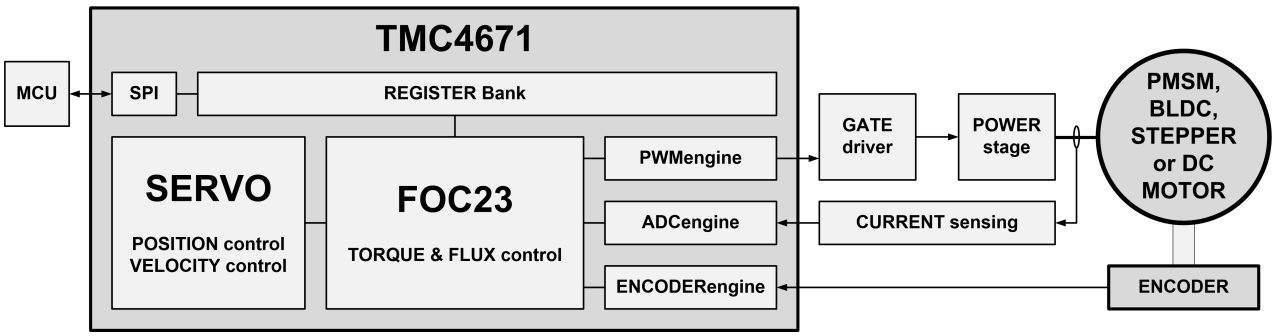
Features

- Servo Controller w/ Field Oriented Control (FOC)
- Torque Control (FOC), Velocity Control, Position Control
- Feed Forward Control Inputs
- Integrated ADCs, $\Delta\Sigma$-ADC Frontend
- Encoder Engine: Hall analog/digital, Encoder analog/digital
- Supports 3-Phase PMSM/BLDC, 2-Phase Stepper Motors, and DC Motors
- Advanced PWM Engine (25kHz...100kHz)
- Application SPI + Debug (UART, SPI)
- Step-Direction Interface (S/D)

Applications

- Robotics
- Pick and Place Machines
- Factory Automation
- E-Mobility
- Laboratory Automation
- Blowers
- Pumps

Simplified Block Diagram



Read entire documentation.

TRINAMIC
MOTION CONTROL

Contents

1 Order Codes

Order Code	Description	Size [mm^2]
TMC4671-ES	TMC4671 FOC Servo Controller IC	10.5 x 6.5
TMC4671-EVAL	TMC4671 Evaluation Board	55 x 85
TMC4671-BOB	TMC4671 Breakout Board	38 x 40

Table 1: Order codes

2 Functional Summary

- **Servo Controller with Field Oriented Control (FOC)**
  - Torque (and flux) control mode
  - Velocity control mode
  - Position control mode
  - update rate of current controller and PWM at maximum frequency of 100 kHz (speed and position controller update rate is configurable by setting a divider of current controller update rate)
- **Control Functions/PI Controllers**
  - Programmable clipping of inputs and outputs of interim results
  - Integrator windup protection for all controllers
  - Programmable field oriented voltage circular ($\sqrt{U_D^2 + U_Q^2}$) limiter
  - Feed-forward offsets for target values and feed-forward friction compensation
  - Advanced feed-forward control structure for optimal trajectory tracking performance
  - Extended IRQ event masking options and limiter status register
  - Advanced encoder initialization algorithms with Hall sensor or/and with minimal movement
- **Motion Control and Ramping**
  - Trapezoidal velocity ramps by control structure
  - Step/Direction interface for easy positioning
- **Supported Motor Types**
  - FOC3 : 3-phase permanent magnet synchronous motors (PMSM)
  - FOC2 : 2-phase stepper motors
  - DC1 : brushed DC motors, or linear voice coil motors
- **ADC Engine with Offset Correction and Scaling**
  - Integrated $\Delta\Sigma$ ADCs for current sense voltage, motor supply voltage, analog encoder, two AGPIs
  - Integrated $\Delta\Sigma$-Interface for external $\Delta\Sigma$-Modulators
- **Position Feedback Evaluation**
  - Open loop position generator (programmable [rpm], [rpm/s]) for initial setup
  - Digital incremental encoder (ABN resp. ABZ, up to 5 MHz)
  - Secondary digital incremental encoder
  - Digital Hall sensor interface (H_1, H_2, H_3 resp. H_U, H_V, H_W) with interpolation of interim positions
  - Analog encoder/analog Hall sensor interface (SinCos (0°, 90°) or 0°, 120°, 240°)
  - multi-turn position counter (32-bit)
  - Position target, velocity and target torque filters (Biquad)
- **PWM Engine Including SVPWM**
  - Programmable PWM frequency within the range of 20 kHz . . . 100 kHz
  - Programmable Brake-Before-Make (BBM) times (high side, low side) 0 ns . . . 2.5 μs in 10 ns steps and gate driver input signals

  - PWM auto scaling for transparent change of PWM frequency during motion

- **SPI Communication Interface**
  - 40-bit datagram length (1 ReadWrite bit + 7 address bits + 32 data bits)
  - Immediate SPI read response (register read access by single datagram)
  - SPI clock frequency up to 1MHz (8MHz in future version)

- **TRINAMIC RealTime Monitoring Interface**
  - High frequency sampling of real-time data via TRINAMIC's real-time monitoring system
  - Only single 10-pin high density connector on PCB needed
  - Advanced controller tuning support by frequency response identification and advanced auto tuning options with TRINAMIC's IDE

- **UART Debug Interface**
  - Three pin (GND, RxD, TxD) 3.3 V UART interface (1N8; 9600 (default), 115200, 921600, or 3M bps)
  - Transparent register access parallel to embedded user application interface (SPI)

- **Supply Voltages**
  - 5V and 3.3V; VCC_CORE is internally generated

- **IO Voltage**
  - 3.3V for all digital IOs (choosable by VCCIO Supply), 5V input range for differential analog inputs, 1.25V input range for single ended inputs

- **Clock Frequency**
  - 25 MHz (external oscillator needed)

- **Packages**
  - QFN76

3 FOC Basics

This section gives a short introduction into some basics of Field Oriented Control (FOC) of electric motors.

3.1 Why FOC?

The Field Oriented Control (FOC), alternatively named Vector Control (VC), is a method for the most energy-efficient way of turning an electric motor.

3.2 What is FOC?

The Field Oriented Control was independently developed by K. Hasse, TU Darmstadt, 1968, and by Felix Blaschke, TU Braunschweig, 1973. The FOC is a current regulation scheme for electro motors that takes the orientation of the magnetic field and the position of the rotor of the motor into account, regulating the strength in such way that the motor gives that amount of torque that is requested as target torque. The FOC maximizes active power and minimizes idle power - that finally results in power dissipation - by intelligent closed-loop control illustrated by figure 1.

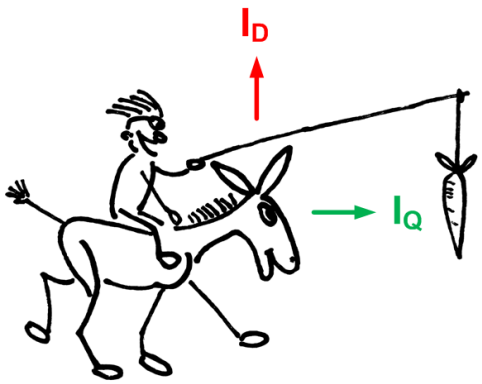


Figure 1: Illustration of the FOC basic principle by cartoon: Maximize active power and minimize idle power and power dissipation by intelligent closed-loop control.

3.3 Why FOC as pure Hardware Solution?

The initial setup of the FOC is usually very time consuming and complex, although source code is freely available for various processors. This is because the FOC has many degrees of freedom that all need to fit together in a chain in order to work.

The hardware FOC as an existing standard building block drastically reduces the effort in system setup. With that off the shelf building block, the starting point of FOC is the setup of the parameters for the FOC. Setting up and implement the FOC itself and building and programming required interface blocks is no longer necessary. The real parallel processing of hardware blocks de-couples the higher lever application software from high speed real-time tasks and simplifies the development of application software. With the TMC4671, the user is free to use its qualified CPU together with its qualified tool chain, freeing the user from fighting with processer-specific challenges concerning interrupt handling and direct memory access. There is no need for a dedicated tool chain to access the TMC4671 registers and to operate it - just SPI (or UART) communication needs to be enabled for any given CPU.

The integration of the FOC as a SoC (System-on-Chip) drastically reduces the number of required components and reduces the required PCB space. This is in contrast to classical FOC servos formed by motor

block and separate controller box wired with motor cable and encoder cable. The high integration of FOC, together with velocity controller and position controller as a SoC, enables the FOC as a standard peripheral component that transforms digital information into physical motion. Compact size together with high performance and energy efficiency especially for battery powered mobile systems are enabling factors when embedded goes autonomous.

3.4 How does FOC work?

Two force components act on the rotor of an electric motor. One component is just pulling in radial direction (ID) where the other component is applying torque by pulling tangentially (IQ). The ideal FOC performs a closed loop current control that results in a pure torque generating current IQ – without direct current ID.

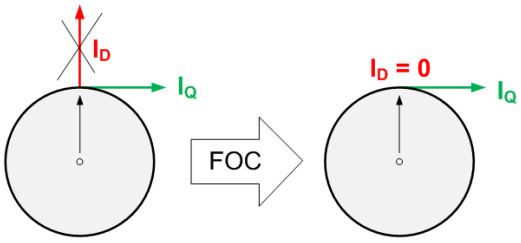


Figure 2: FOC optimizes torque by closed loop control while maximizing IQ and minimizing ID to 0

From top point of view, the FOC for 3-phase motors uses three phase currents of the stator interpreted as a current vector (Iu; Iv; Iw) and calculates three voltages interpreted as a voltage vector (Uu; Uv; Uw) taking the orientation of the rotor into account in a way that only a torque generating current IQ results.

From top point of view, the FOC for 2-phase motors uses two phase currents of the stator interpreted as a current vector (Ix; Iy) and calculates two voltages interpreted as a voltage vector (Ux; Uy) taking the orientation of the rotor into account in a way that only a torque generating current IQ results.

To do so, the knowledge of some static parameters (number of pole pairs of the motor, number of pulses per revolution of an used encoder, orientation of encoder relative to magnetic axis of the rotor, count direction of the encoder) is required together with some dynamic parameters (phase currents, orientation of the rotor).

The adjustment of P parameter P and I parameters of two PI controllers for closed loop control of the phase currents depends on electrical parameters of the motor (resistance, inductance, back EMF constant of the motor that is also the torque constant of the motor, supply voltage).

3.5 What is Required for FOC?

The FOC needs to know the direction of the magnetic axis of the rotor of the motor in refrence to the magnetic axis of the stator of the motor. The magnetic flux of the stator is calculated from the currents through the phases of the motor. The magnetic flux of the rotor is fixed to the rotor and thereby determined by an encoder device.

For the FOC, the user needs to measure the currents through the coils of the stator and the angle of the rotor. The measured angle of the rotor needs to be adjusted to the magnetic axes.

The challenge of the FOC is the high number of degrees of freedom in all parameters.

3.5.1 Coordinate Transformations - Clarke, Park, iClarke, iPark

The FOC requires different coordinate transformations formulated as a set of matrix multiplications. These are the Clarke Transformation (Clarke), the Park Transformation (Park), the inverse Park Transformation (iPark) and the inverse Clarke Transformation (iClarke). Some put Park and Clarke together as DQ transformation and Park and Clarke as inverse DQ transformation.

The TMC4671 takes care of the required transformations so the user no longer has to fight with implementation details of these transformations.

3.5.2 Measurement of Stator Coil Currents

The measurement of the stator coil currents is required for the FOC to calculate a magnetic axis out of the stator field caused by the currents flowing through the stator coils.

Coil current stands for motor torque in context of FOC. This is because motor torque is proportional to motor current, defined by the torque constant of a motor. In addition, the torque depends on the orientation of the rotor of the motor relative to the magnetic field produced by the current through the coils of the stator of the motor.

3.5.3 Stator Coil Currents I_U, I_V, I_W and Association to Terminal Voltages U_U, U_V, U_W

The correct association between stator terminal voltages U_U, U_V, U_W and stator coil currents I_U, I_V, I_W is essential for the FOC. In addition to the association, the signs of each current channel need to fit. Signs of the current can be adapted numerically by the ADC scaler. The mapping of ADC channels is programmable via configuration registers for the ADC selector. Initial setup is supported by the integrated open loop encoder block, that can support the user to turn a motor open loop.

3.5.3.1 Chain of Gains for ADC Raw Values

An ADC raw value is a result of a chain of gains that determine it. A coil current I_SENSE flowing through a sense resistor causes a voltage difference according to Ohm's law. The resulting ADC raw value is a result of the analog signal path according to

$$\text{ADC\_RAW} = (\text{I\_SENSE} * \text{ADC\_GAIN}) + \text{ADC\_OFFSET}. \tag{1}$$

The ADC_GAIN is a result of a chain of gains with individual signs. The sign of the ADC_GAIN is positive or negative, depending on the association of connections between sense amplifier inputs and the sense resistor terminals. The ADC_OFFSET is the result of electrical offsets of the phase current measurement signal path. For the TMC4671, the maximum ADC_RAW value ADC_RAW_MAX = $(2^{16} - 1)$ and the minimum ADC raw value is ADC_RAW_MIN = 0.

$$\begin{aligned} \text{ADC\_GAIN} = & \quad (\quad \text{I\_SENSE\_MAX} * \text{R\_SENSE} \quad) \\ & * \quad \text{SENSE\_AMPLIFIER\_GAIN} \\ & * \quad (\quad \text{ADC\_RAW\_MAX}/\text{ADC\_U\_MAX} \quad) \end{aligned} \tag{2}$$

For the FOC, the ADC_RAW is scaled by the ADC scaler of the TMC4671 together with subtraction of offset to compensate it. Internally, the TMC4671 FOC engine calculates with s16 values. Thus, the ADC scaling needs to be chosen so that the measured currents fit into the s16 range. With the ADC scaler, the user can choose a scaling with physical units like $[mA]$.

3.5.4 Measurement of Rotor Angle

Determination of the rotor angle is either done by sensors (digital encoder, analog encoder, digital Hall sensors, analog Hall sensors) or sensorless by a reconstruction of the rotor angle. Currently, there are no sensorless methods available for FOC that work in a general purpose way as a sensor down to velocity zero.

The TMC4671 does not support sensorless FOC.

3.5.5 Measured Rotor Angle vs. Magnetic Axis of Rotor vs. Magnetic Axis of Stator

The rotor angle, measured by an encoder, needs to be adjusted to the magnetic axis of the rotor. This is because an incremental encoder has an arbitrary orientation relative to the magnetic axis of the rotor, and the rotor has an arbitrary orientation to magnetic axis of the stator.

The direction of counting depends on the encoder, its mounting, and wiring and polarities of encoder signals and motor type. So, the direction of encoder counting is programmable for comfortable definition for a given combination of motor and encoder.

3.5.5.1 Direction of Motion - Magnetic Field vs. Position Sensor

For FOC it is essential, that the direction of revolution of the magnetic field is compatible with the direction of motion of the rotor position reconstructed from encoder signals: For revolution of magnetic field with positive direction, the decoder position needs to turn into the same positive direction. For revolution of magnetic field with negative direction, the decoder position needs to turn into the same negative direction.

With an absolute encoder, once adjusted to the relative orientation of the rotor and to the relative orientation of the stator, one could start the FOC without initialization of the relative orientations.

3.5.5.2 Bang-Bang Initialization of the Encoder

A Bang-Bang initialization is an initialization where the motor is forced with high current into a specific position. For Bang-Bang initialization, the user sets a current into direction D that is strong enough to move the rotor into the desired direction. Other initialization methods ramp up the current smoothly and adjust the current vector to rotor movement detected by the encoder.

3.5.5.3 Encoder Initialization using Hall Sensors

The encoder can be initialized using digital Hall sensor signals. Digital Hall sensor signals give absolute positions within each electrical period with a resolution of sixty degrees. If the Hall sensor signals are used to initialize the encoder position on the first change of a Hall sensor signal, an absolute reference within the electrical period for commutation is given.

3.5.5.4 Minimum Movement Initialization of the Encoder

For minimal movement initialization of the encoder, the user slowly increases a current into direction D and adjusts an offset of the measured angle in a way that the rotor of the motor does not move during initialization while the offset of the measured angle is determined.

3.5.6 Knowledge of Relevant Motor Parameters and Position Sensor (Encoder) Parameters

3.5.6.1 Number of Pole Pairs of a Motor

The number of pole pairs is an essential motor parameter. It defines the ratio between electrical revolutions and mechanical revolutions. For a motor with one pole pair, one mechanical revolution is equivalent to one electrical revolution. For a motor with npp pole pairs, one mechanical revolution is equivalent to npp electrical revolutions, with n = 1, 2, 3, 4, ...

Some define the number of poles NP instead of number of pole pairs NPP for a motor, which results in a factor of two that might cause confusion. For the TMC4671, we use NPP number of pole pairs.

3.5.6.2 Number of Encoder Positions per Revolution

For the encoder, the number of positions per revolution (PPR) is an essential parameter. The number of positions per revolution is essential for the FOC.

Some encoder vendors give the number of lines per revolution (LPR) or just named line count (LC) as encoder parameter. Line count and positions per revolution might differ by a factor of four. This is because of the quadrature encoding - A signal and B signal with phase shift - that give four positions per line, enabling the determination of the direction of revolution. Some encoder vendors associate counts per revolution (CPR) or pulses per revolution associated to PPR acronym.

The TMC4671 uses Positions Per Revolution (PPR) as encoder parameter.

3.5.7 Proportional Integral (PI) Controllers for Closed Loop Current Control

Last but not least, two PI controllers are required for the FOC. The TMC4671 is equipped with two PI controllers - one for control of torque generating current I_Q and one to control current I_D to zero.

3.5.8 Pulse Width Modulation (PWM) and Space Vector Pulse Width Modulation (SVPWM)

The PWM power stage is a must-have for energy efficient motor control. The PWM engine of the TMC4671 just needs a couple of parameters to set PWM frequency fPWM and switching pauses for both high side switches tBBM_H and low side switches tBBM_L. Some control bits are for the programming of power switch polarities for maximum flexibility in the selection in gate drivers for the power MOS-FETs. An additional control bit selects SVPWM on or off. The TMC4671 allows for change of PWM frequency by a single parameter during operation.

With this, the TMC4671 is advanced compared to software solutions where PWM and SVPM configuration of CPU internal peripherals normally needs settings of many parameters.

3.5.9 Orientations, Models of Motors, and Coordinate Transformations

The orientation of magnetic axes (U, V, W for FOC3 resp. X, Y for FOC2) is essential for the FOC together with the relative orientation of the rotor. Here, the rotor is modeled by a bar magnet with one pole pair (n_pole_pairs = 1) with magnetic axis in north-south direction.
The actual magnetic axis of the stator - formed by the motor coils - is determined by measurement of the coil currents.
The actual magnetic axis of the rotor is determined by incremental encoder or by Hall sensors. Incremental encoders need an initialization of orientation, where Hall sensors give an absolute orientation, but with low resolution. A combination of Hall sensor and incremental encoder is useful for start-up initialization.

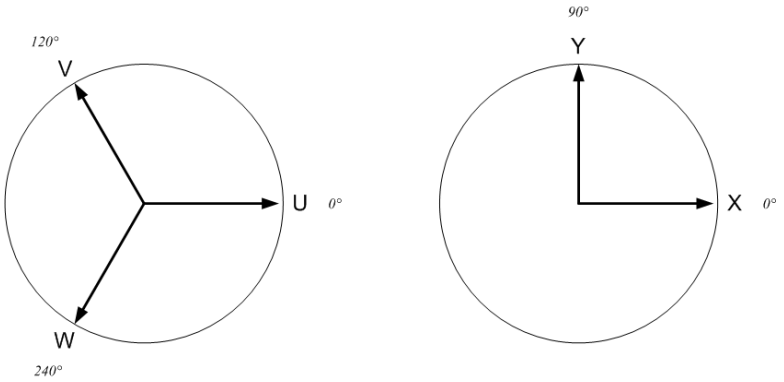


Figure 3: Orientations UVW (FOC3) and XY (FOC2)

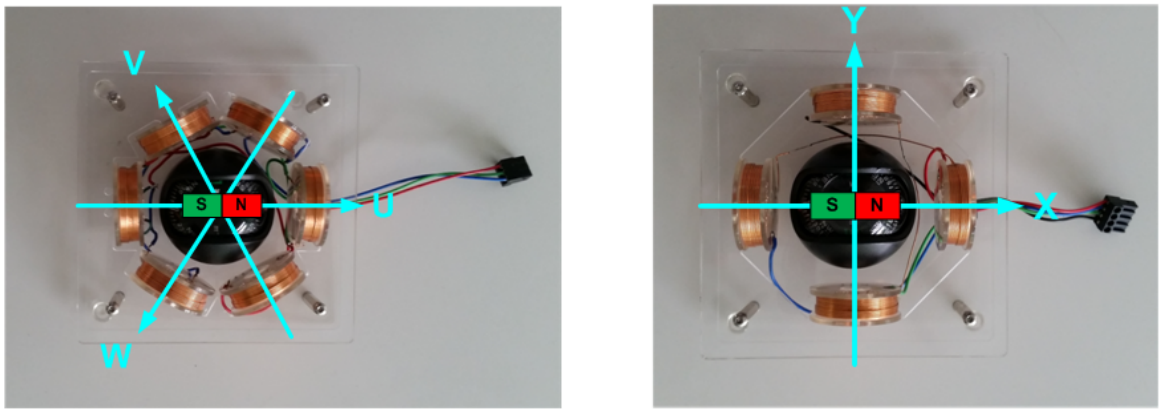


Figure 4: Compass Motor Model w/ 3 Phases UVW (FOC3) and Compass Motor Model w/ 2 Phases (FOC2)

4 Functional Description

The TMC4671 is a fully integrated controller for field-oriented control (FOC) of either one 2-phase stepper motor (FOC2) or one 3-phase brushless motor (FOC3), as well as DC motors or voice coil actuators. Containing the complete control loop core architecture (position, velocity, torque), the TMC4671 also has the required peripheral interfaces for communication with an application controller, for feedback (digital encoder, analog interpolator encoder, digital Hall with interpolator, analog inputs for current and voltage measurement), and helpful additional IOs. The TMC4671 supports highest control loop speed and PWM frequencies.

The TMC4671 is the building block which takes care of all real-time critical tasks of field-oriented motor control. It decouples the real-time field-oriented motor control and its real-time sub-tasks such as current measurement, real-time sensor signal processing, and real-time PWM signal generation from the user application layer as outlined by figure 5.

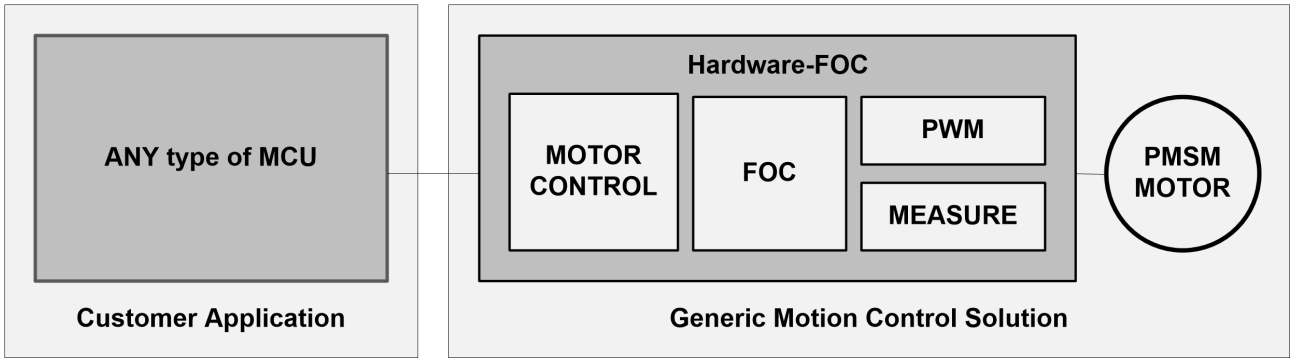


Figure 5: Hardware FOC Application Diagram

4.1 Functional Blocks

The Application interface, register bank, ADC engine, encoder engine, FOC torque PI controller, velocity PI controller, position P controller, and PWM engine make up the TMC4671.

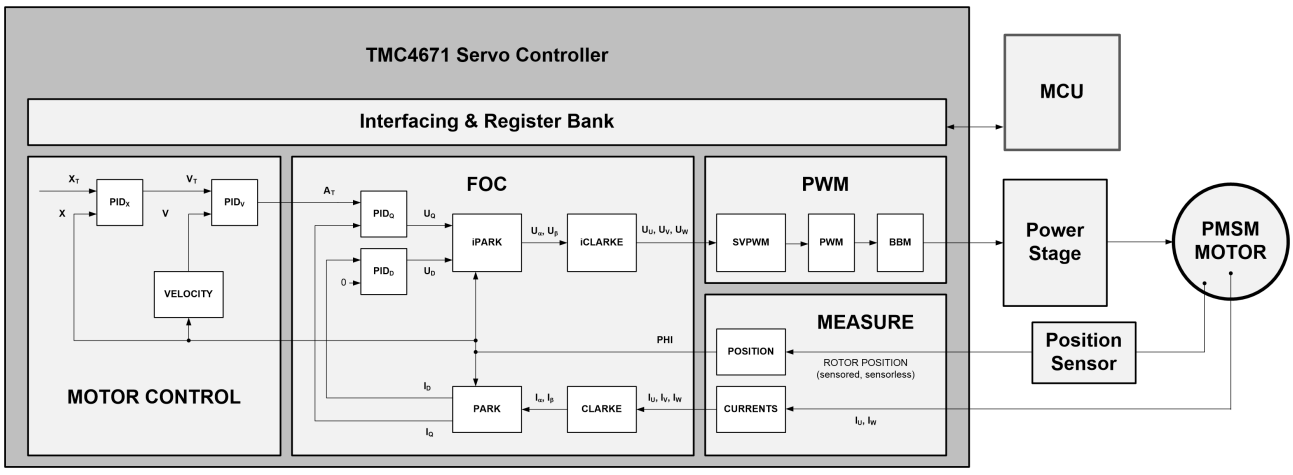


Figure 6: Hardware FOC Block Diagram

The ADC engine interfaces the integrated ADC channels and maps raw ADC values to signed 16 bit (s16) values for the inner FOC current control loop based on programmable offset and scaling factors. The FOC torque PI controller forms the inner base component including required transformations (Clark, Park, inverse Park, inverse Clark). All functional blocks are pure hardware.

4.2 Communication Interfaces

The TMC4671 is equipped with an SPI interface for access to all registers of the TMC4671. The SPI interface is the main application interface.
An additional UART interface is intended for system setup. With that interface, the user can access all registers of the TMC4671 in parallel to the application accessing them via the SPI communication interface - via the user's firmware or via evaluation boards and the TMCL-IDE. The data format of the UART interface is similar to the SPI communication interface - SPI 40 bit datagrams sent to the TMC4671 and SPI 40 bit datagrams received by the MCU vs. five bytes sent via UART and five bytes received via UART. Sending a burst of different real-time data for visualization and analysis via the TMCL-IDE can be triggered using special datagrams. With that, the user can set up an embedded application together with the TMCL-IDE, without having to write a complex set of visualization and analysis functions. The user can focus on its application.
The TMC4671 is also equipped with an additional SPI master interface (TRINAMIC Real-time Monitoring Interface, DBGSPI) for high-speed visualization of real-time data together with the TMCL-IDE.

4.2.1 SPI Slave User Interface

The SPI of the TMC4671 for the user application has an easy command and control structure. The TMC4671 user SPI acts as a slave. The SPI datagram length is 40 bit with a clock rate up to 1 MHz (8 MHz in future chip version).

- The MSB (bit#39) is sent first. The LSB (bit#0) is sent last.
- The MSB (bit#39) is the WRITE_notREAD (WRnRD) bit.
- The bits (bit#39 to bit#32) are the address bits (ADDR).
- Bits (bit#31) to (bit#0) are 32 data bits.

The SPI of the TMC4671 immediately responses within the actual SPI datagram on read and write for ease-of-use communication and uses SPI mode 3 with CPOL = 1 and CPHA = 1.

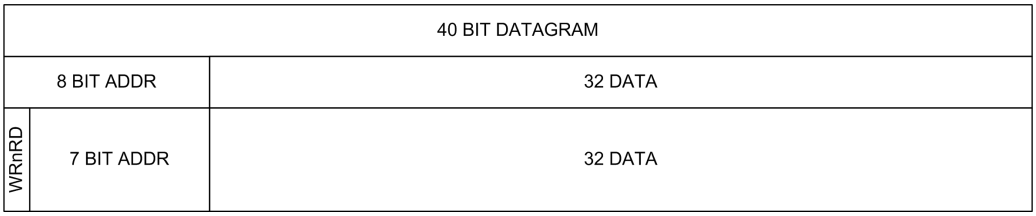


Figure 7: SPI Datagram Structure

A simple SPI datagram example:

```
0x8100000000 // 1st write 0x00000000 into address 0x01 (CHIPINFO_ADDR)
0x0000000000 // 2nd read register 0x00 (CHIPINFO_DATA), returns 0x34363731 <=> ACSII "4671"
```

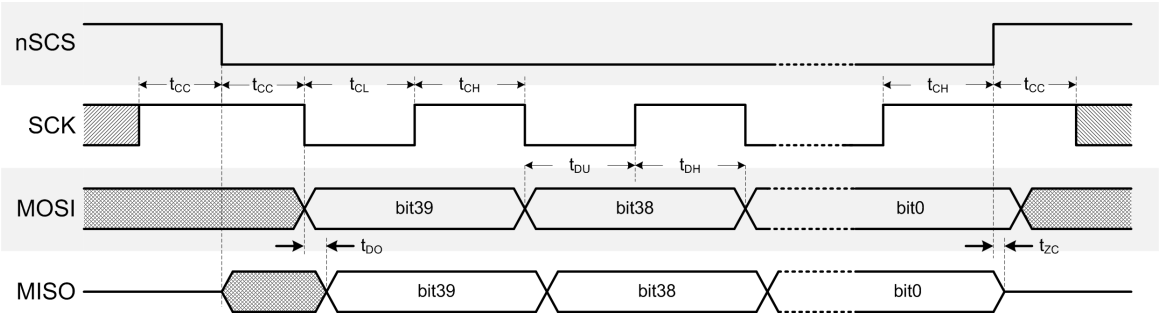


Figure 8: SPI Timing

SPI Interface Timing	Characteristics, fCLK = 25MHz					
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Unit**
SCK valid before or after change of nSCS	t_{CC}		62.5			ns
nSCS high time	t_{CSH}		62.5			ns
nSCS low time	t_{CSL}		62.5			ns
SCK high time	t_{CH}		62.5			ns
SCK low time	t_{CL}		62.5			ns
SCK low time	t_{CL}		62.5			ns
SCK frequency	f_{SCK}				8	MHz
MOSI setup time before rising edge of SCK	t_{DU}		62.5			ns
MOSI hold time after falling edge of SCK	t_{DH}		62.5			ns
MISO data valid time after falling edge of SCK	t_{DO}				10	ns

Table 2: SPI Timing Parameter

Info

The SPI in the TMC4671-ES shows following error: During transaction of read data the MSB (Bit#31) might get corrupted. This shows in two different ways. The first one being a 40 ns pulse (positive or negative) on MISO at the beginning of transfer of that particular bit. This pulse can corrupt the MSB of read data and this error can be avoided when SPI clock frequency is set to 1 MHz. The second error also corrupts MSB of read data when MSB of register is unstable. Such as current measurement noise around zero. In this case, MSB should be ignored when possible. Please also consider that e.g. actual torque value can be read from register PID_TORQUE_FLUX_ACTUAL or from INTERIM_DATA register, where it is showing up in the lower 16 bits. These errors will be fixed in the next IC version. SPI write access is not affected and can be performed at 8 MHz clock frequency.

4.2.2 TRINAMIC Real-Time Monitoring Interface (SPI Master)

The TRINAMIC Real-Time Monitoring Interface (RTMI, SPI Master) is an additional fast interface enabling real-time identification of motor and system parameters. The user can check configuration and access registers in the TMC4671 via the TMCL-IDE with its build-in configuration wizards for FOC setup in parallel to the user firmware. TRINAMIC provides a Monitoring Adapter to access the interface, which connects easily to a single 10 pin high density connector (Type: Hirose DF20F-10DP-1V) on the user's PCB or on the evaluation board. If the interface is not needed, pins can be left open or can be used as GPIOs according to the specification.

The connector needs to be placed near the TMC4671 and assignment needs to be as displayed in figure 9.

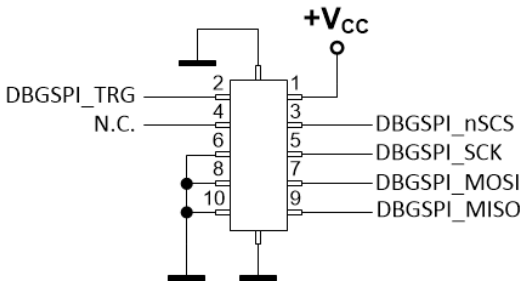


Figure 9: Connector for Real-Time Monitoring Interface (Connector Type: Hirose DF20F-10DP-1V)

Info	The TRINAMIC Real-Time Monitoring Interface can not be used with galvanic isolation, as the timing of SPI communication is too strict. This will be fixed in the next version so that galvanic isolation of SPI signals will be possible with a defined latency of isolators.

4.2.3 UART Interface

The UART interface is a simple three pin (GND, RxD, TxD) 3.3V UART interface with up to 3 Mbit/s transfer speed with one start bit, eight data bits, one stop bit, and no parity bits (1N8). The default speed is 9600 bps. Other supported speeds are 115200 bps, 921600 bps, and 3000000 bps.
With an 3.3V-UART-to-USB adapter cable (e.g. FTDI TTL-232R-RPi), the user can use the full maximum data rate. The UART port enables In-System-Setup-Support by multiple-ported register access.

An UART datagram consists of five bytes - similar to the datagrams of the SPI. In contrast to SPI, the UART interface has a time out feature. So, the five bytes of a UART datagram need to be send within one second. A pause of sending more than one second causes a time out and sets the UART protocol handler back into IDLE state. In other words, waiting for more than one second in sending via UART ensures that the UART protocol handler is in IDLE state.

A simple UART example (similar to the simple SPI example):

```
0x81 0x00 0x00 0x00 0x00 // 1st write 0x00000000 into address 0x01 (CHIPINFO_ADDR)
0x00 0x00 0x00 0x00 0x00 // 2nd read register 0x00 (CHIPINFO_DATA), returns 0x34363731
```

Why UART Interface? It might become necessary during the system setup phase to simply access some internal registers without disturbing the application, without changing the actual user application software, and without adding additional debugging code that might disturb the application software itself. The UART enables this supporting function. In addition, it also enables easy access for monitoring purposes with its very simple and direct five byte protocol. It is not recommended as standard communication interface due to low performance.

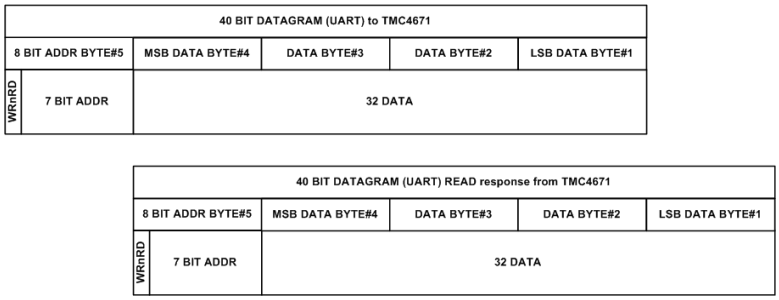


Figure 10: UART Read Datagram (TMC4671 register read via UART)

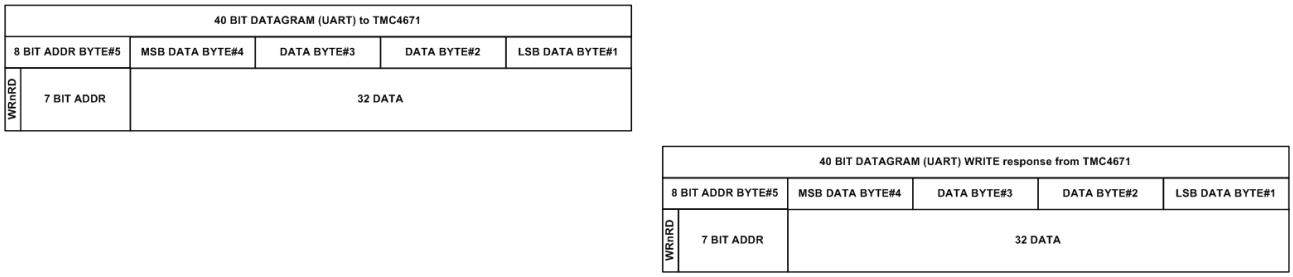


Figure 11: UART Write Datagram (TMC4671 register write via UART)

4.2.4 Step/Direction Interface

The user can manipulate the target position via the step direction interface. It can be enabled by setting the STEP_WIDTH (S32) register to a proper step width.

Info	The Step/Direction interface is not working properly, due to wrong mapping of internal signals. The target position is updated, but not fed into the position controller. This error will be fixed in next IC Version.

4.2.5 Single Pin Interface

The TMC4671 can be operated in Motion Modes in which the main target value is calculated from either a PWM input signal on PIN PWM_I or by analog input to AGPI_A.

Number	Motion Mode	Using PWM_I or AGPI_A
0	Stopped Mode	no
1	Torque Mode	no
2	Velocity Mode	no
3	Position Mode	no
4	PRBS Flux Mode	no
5	PRBS Torque Mode	no
6	PRBS Velocity Mode	no
7	PRBS Position Mode	no
8	UQ UD Ext Mode	no
9	Encoder Init Mini Move Mode	no
10	AGPI_A Torque Mode	AGPI_A
11	AGPI_A Velocity Mode	AGPI_A
12	AGPI_A Position Mode	AGPI_A
13	PWM_I Torque Mode	PWM_I
14	PWM_I Velocity Mode	PWM_I
15	PWM_I Position Mode	PWM_I

Table 3: Single Pin Interface Motion Modes

Registers SINGLE_PIN_IF_OFFSET and SINGLE_PIN_IF_SCALE can be used to scale the value to desired range. In case of the PWM input, a permanent low input signal or permanent high signal is treated as input error and chosen target value is set to zero.
Register SINGLE_PIN_IF_CFG configures the length of a digital filter for the PWM_I signal. Spikes on the signal can be thereby suppressed. Bit 0 in register SINGLE_PIN_IF_STATUS is set high when PWM_I is constant low, Bit 1 is set high when the PWM_I is constant high. Writing to this register resets these flags. Maximum PWM period of the PWM signal must be 65536 x 40 ns. The calculation of the normalized duty cycle is started on the rising edge of PWM_I. The PWM frequency needs to be constant as big variations (tolerance of 4 us in PWM period) in the PWM frequency are treated as error.

A duty cycle of 50% equals an input value of 32768. With the offset and scaling factors it can be mapped to desired range.

4.3 Numerical Representation, Electrical Angle, Mechanical Angle, and Pole Pairs

The TMC4671 uses different numerical representations for different parameters, measured values, and interim results. The terms electrical angle PHI_E, mechanical angle PHI_M, and number of pole pairs (N_POLE_PAIRS) of the motor are important for setup of FOC. This section describes the different numerical representations of parameters and terms.

4.3.1 Numerical Representation

The TMC4671 uses signed and unsigned values of different lengths and fixed point representations for parameters that require a non-integer granularity.

Symbol	Description	Min	Max
u16	unsigned 16 bit value	0	65535
s16	signed 16 bit values, 2'th complement	-32767	32767
u32	unsigned 32 bit value	0	2^{32} = 4294967296
s32	signed 32 bit values, 2'th complement	-2147483647	2^{31} - 1 = 2147483647
q8.8	signed fix point value with 8 bit integer part and 8 bit fractional part	-32767/256	32767/256
q4.12	signed fix point value with 4 bit integer part and 12 bit fractional part	-32767/4096	-32767/4096

Table 4: Numerical Representations

Info Two's complement of n bit is $-2^{(n-1)} \ldots -2^{(n-1)} - 1$. To avoid unwanted overflow, the range is clipped to $-2^{(n-1)} + 1 \ldots -2^{(n-1)} - 1$.

Because the zero is interpreted as a positive number for 2'th complement representation of integer n bit number, the smallest negative number is $-2^{(n-1)}$ where the largest positive number is $2^{(n-1)} - 1$. Using the smallest negative number $-2^{(n-1)}$ might cause critical underflow or overflow. Internal clipping takes this into account by mapping $-2^{(n-1)}$ to $-2^{(n-1)} + 1$.

Hexadecimal Value	u16	s16	q8.8	q4.12
$0x0000_h$	0	0	0.0	0.0
$0x0001_h$	1	1	1 / 256	1 / 4096
$0x0002_h$	2	2	2 / 256	2 / 4096
$0x0080_h$	128	128	0.5	0.03125
$0x0100_h$	256	256	1.0	0.0625
$0x0200_h$	512	512	2.0	0.125
$0x3FFF_h$	16383	16383	16383 / 256	16383 / 4096

Hexadecimal Value	u16	s16	q8.8	q4.12
$0x5A81_h$	23169	23169	23169 / 256	23169 / 4096
$0x7FFF_h$	32767	32767	32767 / 256	32767 / 4096
$0x8000_h$	32768	-32768	-32768 / 256	-32768 / 4096
$0x8001_h$	32769	-32767	-32767 / 256	-32767 / 4096
$0x8002_h$	32770	-32766	-32766 / 256	-32766 / 4096
$0xC001_h$	49153	-16383	-16383 / 256	-16383 / 4096
$0xFFFE_h$	65534	-2	-2 / 256	-2 / 4096
$0xFFFF_h$	65535	-1	-1 / 256	-1 / 4096

Table 5: Examples of u16, s16, q8.8, q4.12

The q8.8 and q4.12 are used for P and I parameters which are positive numbers. Note that q8.8 and q4.12 are used as signed numbers. This is because theses values are multiplied with signed error values resp. error integral values.

4.3.2 N_POLE_PAIRS, PHI_E, PHI_M

The parameter N_POLE_PAIRS defines the factor between electrical angle PHI_E and mechanical angle PHI_M of a motor (pls. refer figure 12).
A motor with one (1) pole pair turns once for each electrical period. A motor with two (2) pole pairs turns once for every two electrical periods. A motor with three (3) pole pairs turns once for every three electrical periods. A motor with four pole (4) pairs turns once for every four electrical periods.

The electrical angle PHI_E is relevant for the commutation of the motor. It is relevant for the torque control of the inner FOC loop.

$$\text{PHI\_E} = \text{PHI\_M} \cdot \text{N\_POLE\_PAIRS} \tag{3}$$

The mechanical angle PHI_M is primarily relevant for velocity control and for positioning. This is because one wants to control the motor speed in terms of mechanical turns and not in terms of electrical turns.

$$\text{PHI\_M} = \text{PHI\_E}/\text{N\_POLE\_PAIRS} \tag{4}$$

Different encoders give different kinds of position angles. Digital Hall sensors normally give the electrical position PHI_E that can be used for commutation. Analog encoders give - depending on their resolution - angles that have to be scaled first to mechanical angles PHI_M and to electrical angles PHI_E for commutation.

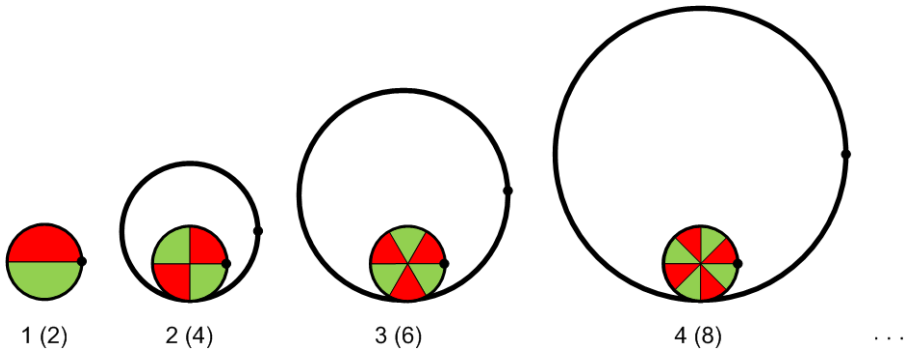


Figure 12: N_POLE_PAIRS - Number of Pole Pairs (Number of Poles)

4.3.3 Numerical Representation of Angles PHI

Electrical angles and mechanical angles are represented as 16 bit integer values. One full revolution of $360\,\mathrm{deg}$ is equivalent to $2^{16} = 65536$ steps. Any position coming from a sensor is mapped to this integer range. Adding an offset of PHI_OFFSET causes a rotation of an angle PHI_OFFSET$/2^{16}$. Subtraction of an offset causes a rotation of an angle PHI_OFFSET in opposite direction.

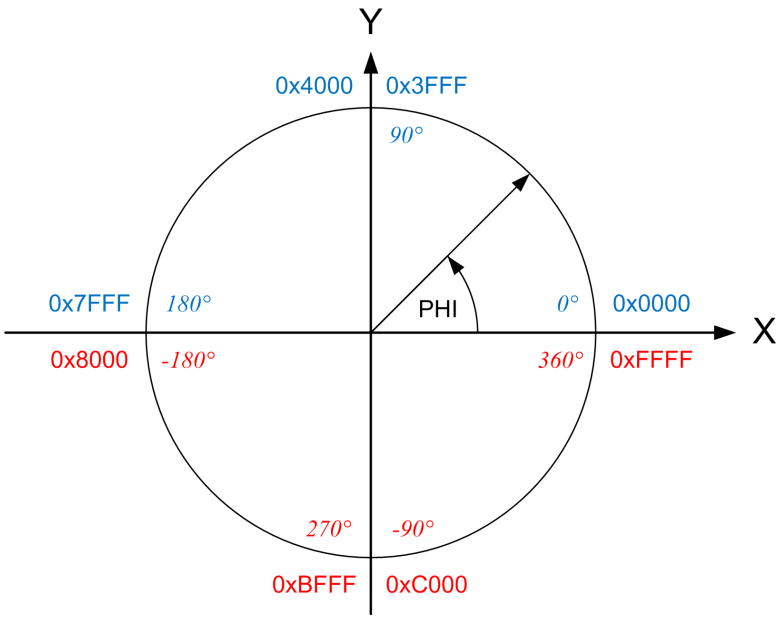


Figure 13: Integer Representation of Angles as 16 Bit signed (s16) resp. 16 Bit unsigned (u16)

Hexadecimal Value	u16	s16	PHI[°]	±PHI[°]
$0x0000_h$	0	0	0.0	0.0
$0x1555_h$	5461	5461	30.0	-330.0
$0x2AAA_h$	10922	10922	60.0	-300.0
$0x4000_h$	16384	16384	90.0	-270.0

Hexadecimal Value	u16	s16	PHI[°]	±PHI[°]
$0x5555_h$	21845	21845	120.0	-240.0
$0x6AAA_h$	27306	27768	150.0	-210.0
$0x8000_h$	32768	-32768	180.0	-180.0
$0x9555_h$	38229	-27307	210.0	-150.0
$0xAAAA_h$	43690	-21846	240.0	-120.0
$0xC000_h$	49152	-16384	270.0	-90.0
$0xD555_h$	54613	-10923	300.0	-60.0
$0xEAAA_h$	60074	-5462	330.0	-30.0

Table 6: Examples of u16, s16, q8.8

The option of adding an offset is for adjustment of angle shift between the motor and stator and the rotor and encoder. Finally, the relative orientations between the motor and stator and the rotor and encoder can be adjusted by just one offset. Alternatively, one can set the counter position of an incremental encoder to zero on initial position. For absolute encoders, one needs to use the offset to set an initial position.

4.4 ADC Engine

The ADC engine controls the sampling of different available ADC channels. The ADC channels (ADC_I0_POS, ADC_I0_NEG, ADC_I1_POS, ADC_I1_NEG) for current measurement are differential inputs. For analog Hall and for analog encoder, the ADC channels have differential inputs (AENC_UX_POS, AENC_UX_NEG, AENC_VN_POS, AENC_VN_NEG, AENC_WY_POS, AENC_WY_NEG). Two general purpose ADC channels are single-ended analog inputs (AGPI_A, AGPI_B). The ADC channel for measurement of supply voltage (ADC_VM) is associated with the brake chopper.

The FOC engine expects offset corrected ADC values, scaled into the FOC engine's 16 bit (s16) fixed point representation. The integrated scaler and offset compensator maps raw ADC samples of current measurement channels to 16 bit two's complement values (s16). While the offset is compensated by subtraction, the offset is represented as an unsigned value. The scaling value is signed to compensate wrong measurement direction. The s16 scaled ADC values are available for read out from the register by the user.

Info Wrong scaling factors (ADC_SCALE) or wrong offsets (ADC_OFFSET) might cause damages when the FOC is active. Integrated hardware limiters allow protection - especially in the setup phase when using careful limits.

ADC samples for measurement of supply voltage (VM) and the general purpose analog ADC inputs are available as raw values only without digital scaling. This is because these values are not processed by the FOC engine. They are just additional ADC channels for the user. The general purpose analog inputs (AGPI) are intended to monitor analog voltage signals representing MOSFET temperature or motor temperature. AGPI_A can also be used for the Single Pin Interface (please see section 4.8.10).

Info ADC_VM must be scaled down by voltage divider to the allowed voltage range, and might require additional supply voltage spike protection.

4.4.1 ADC Group A and ADC Group B

ADC inputs of the TMC4671 are grouped into two groups, to enable different sample rates for two groups of analog signals if needed. For all applications both groups should work with the same sampling rates. necessary to run its ADC channels with a much higher bandwidth than the ADC channels for current measurement.

4.4.2 Internal Delta Sigma ADCs

The TMC4671 is equipped with internal delta sigma ADCs for current measurement, supply voltage measurement, analog GPIs and analog encoder signal measurement. Delta sigma ADCs, as integrated within the TMC4671, together with programmable digital filters are flexible in parameterizing concerning resolution vs. speed. The advantage of delta sigma ADCs is that the user can adjust measurement from lower speed with higher resolution to higher speed with lower resolution. This fits with motor control application. Higher resolution is required for low speed signals, while lower resolution satisfies the needs for high speed signals.

Due to high oversampling, the analog input front-end is easier to implement than for successive approximation register ADCs as anti aliasing filters can be chosen to a much higher cutoff frequency. The ADC Engine processes all ADC channels in parallel hardware - avoiding phase shifts between the channels compared to ADC channels integrated in MCUs.

An analog voltage V_IN of an analog input is mapped to a raw ADC value ADC_RAW.

4.4.3 External Delta Sigma ADCs

The delta sigma front-end of the ADC engine supports external delta sigma modulators to enable isolated delta sigma modulators for the TMC4671. Additionally, the delta sigma front-end supports low-cost comparators together with two resistors and one capacitor (R-C-R-CMP) forming first order delta sigma modulators, as generic analog front-end for pure digital variants of the TMC4671 core.

4.4.3.1 ADC RAW

The sampled raw ADC values are available for read out by the user. This is important during the system setup phase to determine offset and scaling factors.

4.4.3.2 ADC EXT

The user can write ADC values into the ADC_EXT registers of the register bank from external sources. These values can be selected as raw current ADC values by selection. ADC_EXT registers are primarily intended for test purposes as optional inputs for external current measurement sources.

4.5 Delta Sigma Configuration and Timing Configuration

The delta sigma configuration is programmed via MCFG register that selects the mode (internal/external delta sigma modulator with fixed internal 100MHz system clock or with programmable MCLK; delta sigma modulator clock mode (MCLK output, MCLK input, MCLK used as MDAC output with external R-C-R-CMP configuration); delta sigma modulator clock and its polarity; and the polarity of the delta sigma modulator data signal MDAT).

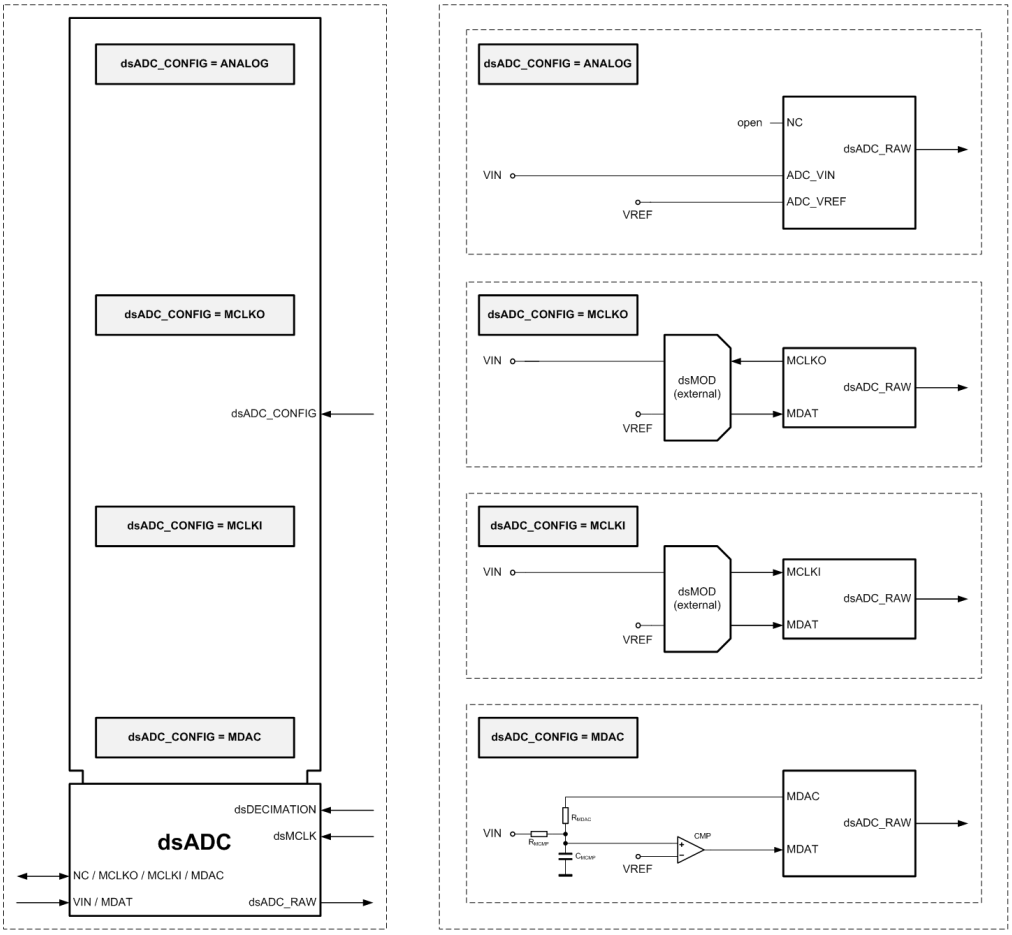


Figure 14: Delta Sigma ADC Configurations dsADC_CONFIG (internal: ANALOG vs. external: MCLKO, MCLKI, MDAC)

dsADC_CONGIG	Description	NC_MCLKO_MCLKI_MDAC	VIN_MDAT
ANALOG	integrated internal ADC mode, VIN_MDAT is analog input VIN	MCLK not connected (NC)	VIN (analog)
MCLKO	external dsModulator (e.g. AD7403) with MCLK input driven by MCLKO	MCLK output	MDAT input
MCLKI	external dsModulator (e.g. AD7400) with MCLK output that drives MCLKI	MCLK input	MDAT input
MDAC	external dsModulator (e.g. LM339, LM319) realized by external comparator CMP with two R and one C	MDAC output (= MCLK out)	MDAT input for CMP

Table 7: Delta Sigma ADC Configurations (figure 14), selected with dsADC_MCFG_A and dsADC_MCFG_B.

register	function
dsADC_MCFG_B	delta sigma modulator configuration MCFG (ANALOG, MCLKI, MCLKO, MDAC), group B
dsADC_MCFG_A	delta sigma modulator configuration MCFG (ANALOG, MCLKI, MCLKO, MDAC), group A
dsADC_MCLK_B	delta sigma modulator clock MCLK, group B
dsADC_MCLK_A	delta sigma modulator clock MCLK, group A
dsADC_MDEC_B	delta sigma decimation parameter MDEC, group B
dsADC_MDEC_A	delta sigma decimation parameter MDEC, group A

Table 8: Registers for Delta Sigma Configuration

4.5.0.1 Timing Configuration MCLK

When the programmable MCLK is selected, the MCLK_A and MCLK_B parameter registers define the programmable clock frequency fMCLK of the delta sigma modulator clock signal MCLK for delta sigma modulator group A and group B. For a given target delta sigma modulator frequency fMCLK, together with the internal clock frequency fCLK = 100MHz, the MCLK frequency parameter is calculated by

$$\text{MCLK} = 2^{31} * \text{fMCLK}[Hz]/\text{fCLK}[Hz] \tag{5}$$

Due to the 32 bit's length of the MCLK frequency parameter, the resulting frequency fMCLK might differ from the desired frequency fMCLK. The back calculation of the resulting frequency fMCLK for a calculated MCLK parameter with 32 bit length is defined by

$$\text{fMCLK}[Hz] = \text{fCLK}[Hz] * \text{MCLK}/2^{31} \tag{6}$$

The precise programming of the MCLK frequency is primarily intended for external delta sigma modulators to meet given EMI requirements. With that, the user can programm frequencies fMCLK with a resolution better than 0.1 Hz. This advantage concerning EMI might cause trouble when using external delta signal modulators if they are sensitive to slight frequency alternating. This is not an issue when using external first-order delta sigma modulators based on R-C-R-CMP (e.g. LM339). But for external second-order delta signal modulators, it is recommended to configure the MCLK parameter for frequencies fMCLK with kHz quantization (e.g. 10,001,000 Hz instead of 10,000,001 Hz). Table 9 gives an overview of MCLK parameter settings for different frequencies fMCLK.

fMCLK_target	MCLK	fMCLK_resulting	comment
25 MHz	0x20000000	25 MHz	w/o fMCLK frequency jitter, recommended
20 MHz	0x19000000	20 MHz -468750 Hz	recommended for ext. $\Delta\Sigma$ modulator
20 MHz	0x19999999	20 MHz -0.03 Hz	might be critical for ext. $\Delta\Sigma$ modulator
12.5 MHz	0x10000000	12.5 MHz	w/o fMCLK frequency jitter, recommended
10 MHz	0x0CCCCCCC	10 MHz -0.04 Hz	might be critical for ext. $\Delta\Sigma$ modulator
10 MHz	0x0CC00000	10 MHz -39062.5 Hz	recommended for ext. $\Delta\Sigma$ modulator

Table 9: Delta Sigma MCLK Configurations

Info	Parametrization of fMCLK will be changed in a future version of the chip to match usual modulator frequencies like 10MHz and 20MHz better. It is recommended to use a Modulatorfrequency of 25kHz for all applications. If the second ADC group is not needed, it is recommended to shut it off by setting the MCLK_B register to 0x0.

4.5.0.2 Decimation Configuration MDEC

The high oversampled single bit delta sigma data stream (MDAT) is digitally filtered by Sinc3 filters. To get raw ADC data, the actual digitally filtered values need to be sampled periodically with a lower rate called decimation ratio. The decimation is controlled by parameter MDEC_A for ADC group A and MDEC_B for ADC group B. A new ADC_RAW value is available after MDEC delta sigma pulses of MCLK. As such, the parameters MCLK and MDEC together define the sampling rate of the 16 bit ADC_RAW values.

The delta sigma modulator with Sinc3 filter works with best noise reduction performance when the length of the step response time tSINC3 of the Sinc3 filter is equal to the length of the PWM period tPWM = (PWM_MAXCNT+1) / fPWMCLK = ((PWM_MAXCNT+1) * 10 ns) of the period. The length of the step function response of a Sinc3 filter is

$$\text{tSINC3} = (3 \cdot (\text{MDEC} - 1) + 1) \cdot \text{tMCLK} \tag{7}$$

$$\text{MDEC}_{\text{recommended}} = \frac{\text{tPWM}}{3 \cdot \text{tMCLK}} - 2 \tag{8}$$

fMCLK	tMCLK	MDEC25 (25 kHz, 40μs)	MDEC50 (50 kHz, 20μs)	MDEC100 (100 kHz, 10μs)
50 MHz	20 ns	665	331	165
25 MHz	40 ns	331	165	81
20 MHz	50 ns	265	131	65
12.5 MHz	80 ns	165	81	40
10 MHz	100 ns	131	65	31

Table 10: Recommended Decimation Parameter MDEC (equation (8) for different PWM frequencies fPWM (MDEC25 for fPWM=25kHz w/ PWM_MAXCNT=3999, MDEC50 for fPWM=50kHz w/ PWM_MAXCNT=1999, MDEC100 for fPWM=100kHz w/ PWM_MAXCNT=999).

Info	Internal structure of the Sinc3 and synchronization to PWM will be enhanced in future version of the chip. This might need the user's application controller software to be changed.

4.5.1 Internal Delta Sigma Modulators - Mapping of V_RAW to ADC_RAW

Generally, delta sigma modulators work best for a typical input voltage range of 25% V_MAX . . . 75% V_MAX. For the integrated delta sigma modulators, this input voltage operation range is recommended with V_MAX

= 5V where V_MAX = 3.3V is possible. The table 11 defines the recommended voltage ranges for both 5V and 3.3V analog supply voltages.

V_SUPPLY[V]	(V_MIN[V])	V_MIN25%[V]	V_MAX50%[V]	V_MAX75%[V]	(V_MAX[V])
(3.3)	(0.0)	(0.825)	(1.65)	(2.75)	(3.3)
5.0	(0.0)	1.250	2.50	3.75	(5.0)

Table 11: Recommended input voltage range from V_MIN25%[V] to V_MAX75%[V] for internal Delta Sigma Modulators; V_SUPPLY[V] = 5V is recommended for the analog part of the TMC4671.

$$\text{V\_RAW} = \begin{cases} \text{V\_MAX} & \text{for} \quad \text{V\_IN} > \text{V\_MAX} \\ (\text{V\_IN} - \text{V\_REF}) & \text{for} \quad \text{V\_MIN} < (\text{V\_IN} - \text{V\_REF}) < \text{V\_MAX} \\ \text{V\_MIN} & \text{for} \quad \text{V\_IN} < \text{V\_MIN} \end{cases} \tag{9}$$

The resulting raw ADC value is

$$\text{ADC\_RAW} = (2^{16} - 1) \cdot \frac{\text{V\_RAW}}{\text{V\_MAX}} \quad \text{for} \quad \text{V\_MIN25\%[V]} < \text{V\_RAW} < \text{V\_MAX75\%[V]}. \tag{10}$$

The idealized expression (equation 9) is valid for recommended voltage ranges (table 11) neglecting deviations in linearities. These deviations primarily depend on different impedance on the analog signal path, but also on digital parameterization. Finally, the deviation is quantified in terms of resulting ADC resolution.

So, the Delta Sigma ADC engine maps the analog input voltages V_RAW = V_IN - V_REF of voltage range V_MIN < V_RAW < V_MAX to ADC_RAW values of range $\{0 \ldots (2^{16}) - 1\}$ <=> $\{0 \ldots 65535\}$ <=> 0x0000 . . . 0xFFFF.

Vmin[V]	Vref[V]	Vmax[V]	VIN[V]	DUTY[%]	ADC_RAW
0.0	2.5	5.0	(0.0)	(0%)	(0x0000)
0.0	2.5	5.0	1.0	25%	0x4000
0.0	2.5	5.0	2.5	50%	0x7fff
0.0	2.5	5.0	3.75	75%	0xC000
0.0	2.5	5.0	(5.0)	(100%)	(0xffff)

Table 12: Delta Sigma input voltage mapping of internal Delta Sigma Modulators)

Info For calibrating purposes, the input voltage of the delta sigma ADC inputs can be programmed to fixed voltages (25%, 50%, 75% of analog supply voltage) via the associated configuration register DS_ANALOG_INPUT_STAGE_CFG.

4.5.2 External Delta Sigma Modulator Interface

The TMC4671 is equipped with integrated digital filters for extraction of ADC raw values from delta sigma data stream for both internal and external delta sigma modulators. The interface for external delta sigma modulators is intended for external isolated sigma delta modulators, such as AD7401 (with MCLK input driven by TMC4671), or AD7402 (with MCLK output to drive TMC4671). In addition, the external delta sigma interface supports the use of simple comparator with a R-C-R network as external low cost delta sigma modulators (R-C-R-CMP, e.g. LM339).

Info	When selecting the external delta sigma ADC Interface, the high-performance Debug SPI Interface (RTMI) it not available in parallel due to pin sharing. The UART is always available, but with less performance than the RTMI.

Each external delta sigma modulator channel (dsMOD) has two signals (pls. refer figure 14), one dedicated input, and one programmable input/output. The configuration of the external delta sigma modulator interface is defined by programming associated registers. When selecting external delta signal ADC, the associated analog ADC inputs are configured as digital inputs for the delta sigma signal data stream MDAT.

4.5.3 ADC Configuration - MDAC

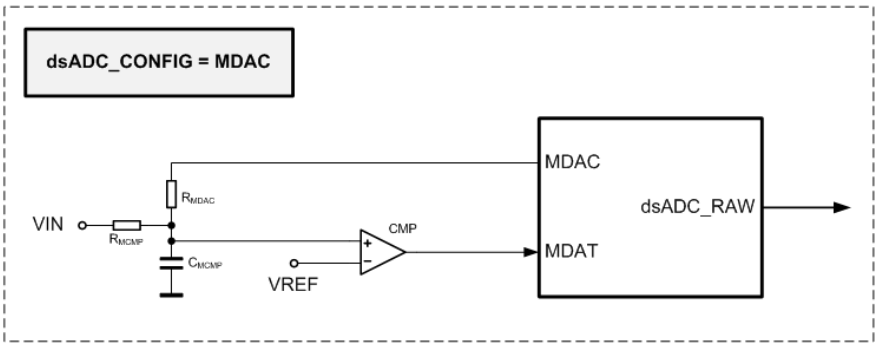


Figure 15: $\Delta\Sigma$ ADC Configuration - MDAC (Comparator-R-C-R as $\Delta\Sigma$-Modulator)

In the MDAC delta sigma modulator, the delay of the comparator CMP determines the MCLK of the comparator modulator. A capacitor C_{MCCMP} within a range of 100 pF . . . 1nF fits in most cases. The time constant τRC should be in a range of 0.1 tCMP . . . tCMP of the comparator. The resistors should be in the range of 1K to 10K. The fMAXtyp depends also on the choice of the decimation ratio.

CMP	tCMPtyp $[ns]$	R_{MCMP} $[k\Omega]$	R_{MDAC} $[k\Omega]$	C_{MCMP} $[pF]$	fMCLKmaxTYP
LM319	100	1	1	100	10 MHz
LM319	100	10	10	100	1 MHz
LM319	100	100	100	100	100 kHz
CMP	tCMPtyp $[ns]$	R_{MCMP} $[k\Omega]$	R_{MDAC} $[k\Omega]$	C_{MCMP} $[pF]$	fMCLKmaxTYP
LM339	1000	1	1	100	1 MHz

CMP	tCMPtyp $[ns]$	R_{MCMP} $[k\Omega]$	R_{MDAC} $[k\Omega]$	C_{MCMP} $[pF]$	fMCLKmaxTYP
LM339	1000	10	10	100	100 kHz
LM339	1000	100	100	100	10 kHz

Table 13: Delta Sigma R-C-R-CMP Configurations (pls. refer 14)

Vmin[V]	Vref[V]	Vmax[V]	VIN[V]	DUTY[%]	ADC_RAW
0.0	1.65	3.3	0.0	0%	0x0000
0.0	1.65	3.3	0.825	25%	0x4000
0.0	1.65	3.3	1.65	50%	0x7fff
0.0	1.65	3.3	2.475	75%	0xC000
0.0	1.65	3.3	3.3	100%	0xffff
Vmin[V]	Vref[V]	Vmax[V]	VIN[V]	DUTY[%]	ADC_RAW
0.0	2.5	5.0	0.0	0%	0x0000
0.0	2.5	5.0	1.0	25%	0x4000
0.0	2.5	5.0	2.5	50%	0x7fff
0.0	2.5	5.0	3.75	75%	0xC000
0.0	2.5	5.0	5.0	100%	0xffff

Table 14: Delta Sigma input voltage mapping of external comparator (CMP)

4.6 Analog Signal Conditioning

The range of measured coil currents, resp. the measured voltages of sense resistors, needs to be mapped to the valid input voltage range of the delta sigma ADC inputs. This analog preprocessing is the task of the analog signal conditioning.

4.6.0.1 Chain of Gains for ADC Raw Values

An ADC raw value is a result of a chain of gains that determine it. A coil current I_SENSE flowing through a sense resistor causes a voltage difference according to Ohm's law. Finally, a current is mapped to an ADC raw value

$$\text{ADC\_RAW} = (\text{I\_SENSE} * \text{ADC\_GAIN}) + \text{ADC\_OFFSET}. \tag{11}$$

The ADC_GAIN is a result of a chain of gains with individual signs. The sign of the ADC_GAIN is positive or negative, depending on the association of connections between sense amplifier inputs and the sense resistor terminals. The ADC_OFFSET is the result of electrical offsets of the phase current measurement signal path. For the TMC4671, the maximum ADC_RAW value is ADC_RAW_MAX = $(2^{16}-1)$ and the minimum ADC raw value is ADC_RAW_MIN = 0.

$$\begin{aligned} \text{ADC\_GAIN} &= \left(\text{I\_SENSE\_MAX} * \text{R\_SENSE} \right) \\ &* \text{SENSE\_AMPLIFIER\_GAIN} \\ &* \left(\text{ADC\_RAW\_MAX/ADC\_U\_MAX} \right) \end{aligned} \tag{12}$$

Rsense $[m\Omega]$	Isense $[A]$	Usense $[mV]$	GAIN$[V/V]$	ADC_GAIN$[A/V]$	Sense Amplifier
5	10	50	20	10	AD8204
10	5	50	20	5	AD8204

Table 15: Example Parameters for ADC_GAIN

For the FOC, the ADC_RAW is scaled by the ADC scaler of the TMC4671 together with subtraction of offset to compensate it. Internally, the TMC4671 FOC engine calculates with s16 values. So, the ADC scaling needs to be chosen so that the measured currents fit into the s16 range. With the ADC scaler, the user can choose a scaling with physical units like $[mA]$. A scaling to $[mA]$ covers a current range of $-32A \ldots +32A$ with $m[A]$ resolution. For higher currents, the user can choose unusual units like centi Ampere $[cA]$ covering $-327A \ldots +327A$ or deci Ampere $-3276A \ldots +3276A$.

ADC scaler and offset compensators are for mapping raw ADC values to s16 scaled and offset cleaned current measurement values that are adequate for the FOC.

4.6.1 FOC3 - Stator Coil Currents I_U, I_V, I_W and Association to Terminal Voltages U_U, U_V, U_W

The correct association between stator terminal voltages U_U, U_V, U_W and stator coil currents I_U, I_V, I_W is essential for the FOC.

For three-phase motors with three terminals U, V, W, the voltage U_U is in phase with the current I_U, U_V is in phase with I_V, and U_W is in phase with I_W according to equations (13) and (14) for FOC3.

$$\text{U\_UVW\_FOC3(U\_D, PHI\_E)} = \begin{cases} U_U(\phi_e) = U_D \cdot sin(\phi_e) \\ U_V(\phi_e) = U_D \cdot sin(\phi_e + 120^o) \\ U_W(\phi_e) = U_D \cdot sin(\phi_e - 120^o) \end{cases} \tag{13}$$

$$\text{I\_UVW\_FOC3(I\_D, PHI\_E)} = \begin{cases} I_U(\phi_e) = I_D \cdot sin(\phi_e) \\ I_V(\phi_e) = I_D \cdot sin(\phi_e + 120^o) \\ I_W(\phi_e) = I_D \cdot sin(\phi_e - 120^o) \end{cases} \tag{14}$$

4.6.2 Stator Coil Currents I_X, I_Y and Association to Terminal Voltages U_X, U_Y

For two-phase motors (stepper) with four terminals X1, X2, and Y1, Y2, voltage U_Ux = U_X1 - U_X2 is in phase with the measured current I_X and U_Wy = U_Y1 - U_Y2 is in phase with the measured current I_Y according to equations (15) and (16) for FOC2.

$$\text{U\_XY\_FOC2} = \begin{cases} U_X(\phi_e) = U_X * sin(\phi_e) \\ U_Y(\phi_e) = U_Y * sin(\phi_e + 90^o) \end{cases} \quad (15)$$

$$\text{I\_XY\_FOC2} = \begin{cases} I_X(\phi_e) = I_D * sin(\phi_e) \\ I_Y(\phi_e) = I_D * sin(\phi_e + 90^o) \end{cases} \quad (16)$$

4.6.3 ADC Selector & ADC Scaler w/ Offset Correction

The ADC selector selects ADC channels for FOC. The 3-phase FOC uses two of three ADC channels for measurement and calculates the third channel via Kirchhoff's Law using the scaled and offset-corrected ADC values. The 2-phase FOC just uses two ADC channels because for a 2-phase stepper motor, the two phases are independent from each other.

Note	The open-loop encoder is useful for setting up ADC channel selection, scaling, and offset by running a motor open-loop.

The FOC23 Engine processes currents as 16 bit signed (s16) values. Raw ADC values are expanded to 16 bit width, regardless of their resolution. With this, each ADC is available for read out as a 16 bit number. The ADC scaler w/ offset correction is for the preprocessing of measured raw current values. It might be used to map to user's own units (e.g. A or mA). For scaling, gains of current amplifiers, reference voltages, and offsets have to be taken into account.

Info	Raw ADC values generally are of 16 bit width, regardless of their real resolution.

Info	The ADC scaler maps raw ADC values to the 16 bit signed (s16) range and centers the values to zero by removing offsets.

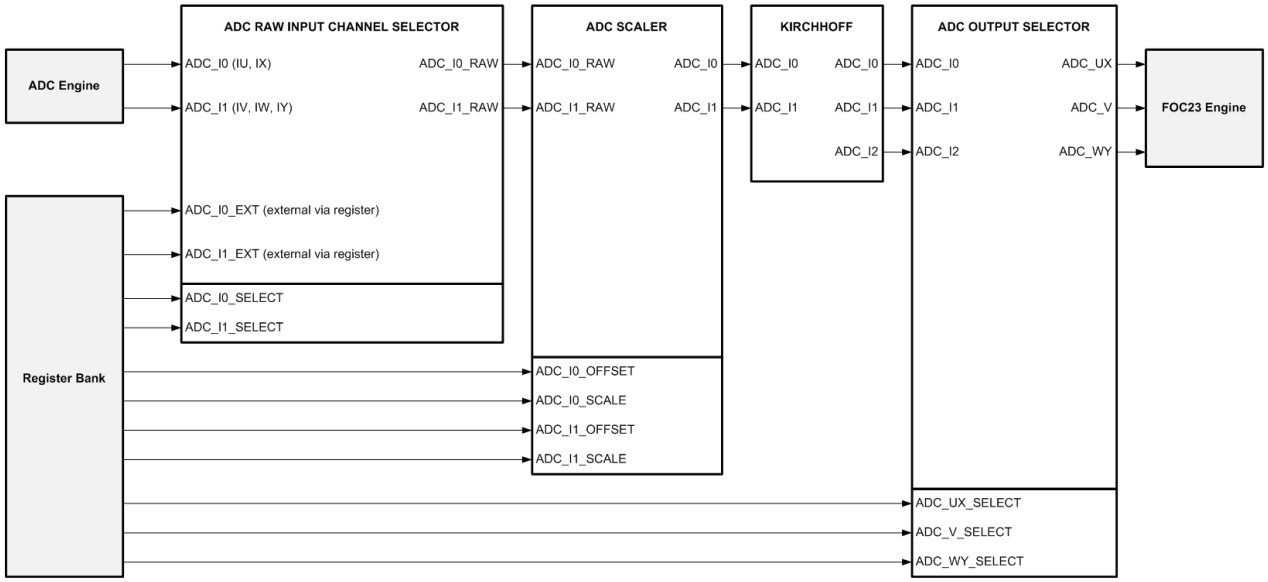


Figure 16: ADC Selector & Scaler w/ Offset Correction

ADC offsets and ADC scalers for the analog current measurement input channels need to be programmed into the associated registers. Each ADC_I_U, ADC_I_V, ADC_I_UX, ADC_I_WY, ADCSD_I_UX, ADCSD_I_WY, ADC_I0_EXT, and ADC_I1_EXT is mapped either to ADC_I0_RAW or to ADC_I1_RAW by ADC_I0_SELECT and ADC_I1_SELECT.

In addition, the ADC_OFFSET is for conversion of unsigned ADC values into signed ADC values as required for the FOC.

$$\text{ADC\_I0} = (\text{ADC\_I0\_RAW} + \text{ADC\_I0\_OFFSET}) \cdot \text{ADC\_I0\_SCALE} \tag{17}$$

$$\text{ADC\_I1} = (\text{ADC\_I1\_RAW} + \text{ADC\_I1\_OFFSET}) \cdot \text{ADC\_I1\_SCALE} \tag{18}$$

For FOC3, the third current ADC_I2 is calculated via Kirchhoff's Law. This requires the correct scaling and offset correction beforehand. For FOC2, there is no calculation of a third current.

The ADC_UX_SELECT selects one of the three ADC channels ADC_I0 ADC_I1, or ADC_I2 for ADC_UX.

The ADC_V_SELECT selects one of the three ADC channels ADC_I0 ADC_I1, or ADC_I2 for ADC_V.

The ADC_WY_SELECT selects one of the three ADC channels ADC_I0 ADC_I1, or ADC_I2 for ADC_WY.

The ADC_UX, ADC_V, and ADC_WY are for the FOC3 (U, V, W). The ADC_UX and ADC_WY (X, Y) are for the FOC2.

Note The open-loop encoder is useful to run a motor open loop for setting up the ADC channel selection with correct association between phase currents I_U, I_V, I_W and phase voltages U_U, U_V, U_W.

4.7 Encoder Engine

The encoder engine is an unified position sensor interface. It maps the selected encoder position information to electrical position (phi_e) and to mechanical position (phi_e). Both are 16 bit values. The encoder engine maps single turn positions from position sensors to multi-turn positions. The user can overwrite the multi-turn position for initialization.

The different position sensors are the position sources for torque and flux control via FOC, for velocity control, and for position control. The PHI_E_SELECTION selects the source of the electrical angel phi_e for the inner FOC control loop. VELOCITY_SELECTION selects the source for velocity measurement. With phi_e selected as source for velocity measurement, one gets the electrical velocity. With the mechanical angle phi_m selected as source for velocity measurement, one gets the mechanical velocity taking the set number of pole pairs (N_POLE_PAIRS) of the motor into account. Nevertheless, for a highly precise positioning, it might be useful to do positioning based on the electrical angel phi_e.

4.7.1 Open-Loop Encoder

For initial system setup, the encoder engine is equipped with an open-loop position generator. This allows for turning the motor open-loop by specifying speed in rpm and acceleration in rpm/s, together with a voltage UD_EXT in D direction. As such, the open-loop encoder is not a real encoder. It simply gives positions as an encoder does. The open-loop decoder has a direction bit to define direction of motion for the application.

Note	The open-loop encoder is useful for initial ADC setup, encoder setup, Hall signal validation, and for validation of the number of pole pairs of a motor. The open-loop encoder turns a motor open with programmable velocity in unit [RPM] with programmable acceleration in unit [RPM/s].

With the open-loop encoder, the user can turn a motor without any position sensor and without any current measurement as a first step of doing the system setup. With the turning motor, the user can adjust the ADC scales and offsets and set up positions sensors (Hall, incremental encoder, . . .) according to resolution, orientation, and direction of rotation.

4.7.2 Incremental ABN Encoder

The incremental encoders give two phase shifted incremental pulse signals A and B. Some incremental encoders have an additional null position signal N or zero pulse signal Z. An incremental encoder (called ABN encoder or ABZ encoder) has an individual number of incremental pulses per revolution. The number of incremental pulses define the number of positions per revolution (PPR). The PPR might mean pulses per revolution or periods per revolution. Instead of positions per revolution, some incremental encoder vendors call these CPR counts per revolution.

The PPR parameter is the most important parameter of the incremental encoder interface. With that, it forms a modulo (PPR) counter, counting from 0 to (PPR-1). Depending on the direction, it counts up or down. The modulo PPR counter is mapped into the register bank as a dual ported register. The user can overwrite it with an initial position. The ABN encoder interface provides both the electrical position and the multi-turn position, which are accessible through dual-ported read-write registers.

Note	The PPR parameter must be set exactly according to the used encoder.

The N pulse from an encoder triggers either sampling of the actual encoder count to fetch the position at the N pulse or it re-writes the fetched n position on an N pulse. The N pulse can either be used as stand

alone pulse or and-ed with NAB = N and A and B. It depends on the decoder what kind of N pulse has to be used - either N or NAB. For those encoders with precise N pulse within one AB quadrant, the N pulse must be used. For those encoders with N pulse over four AB quadrants the user can enhance the precision of the N pulse position detection by using NAB instead of N.

Note	Incremental encoders are available with N pulse and without N pulse.

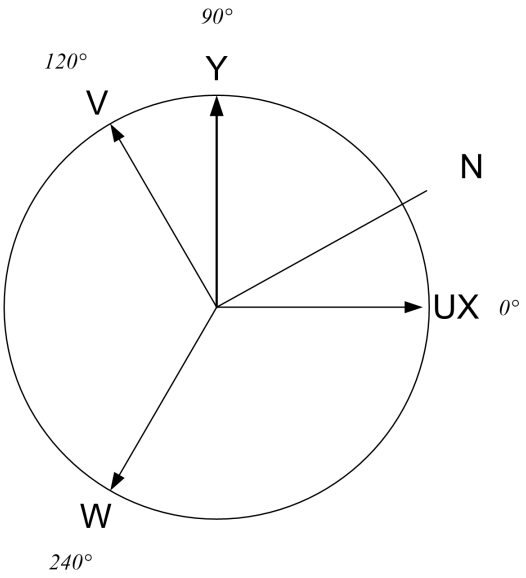


Figure 17: ABN Incremental Encoder N Pulse

The polarity of N pulse, A pulse and B pulse are programmable. The N pulse is for reinitialization with each turn of the motor. Once fetched, the ABN decoder can be configured to write back the fetched N pulse position with each N pulse.

Note	The ABN encoder interface has a direction bit to set to match wiring of motor to direction of encoder.

Logical ABN = A and B and N might be useful for incremental encoders with low resolution N pulse to enhance the resolution. On the other hand, for incremental encoders with high resolution N pulse a logical ABN = A and B and N might totally suppress the resulting N pulse.

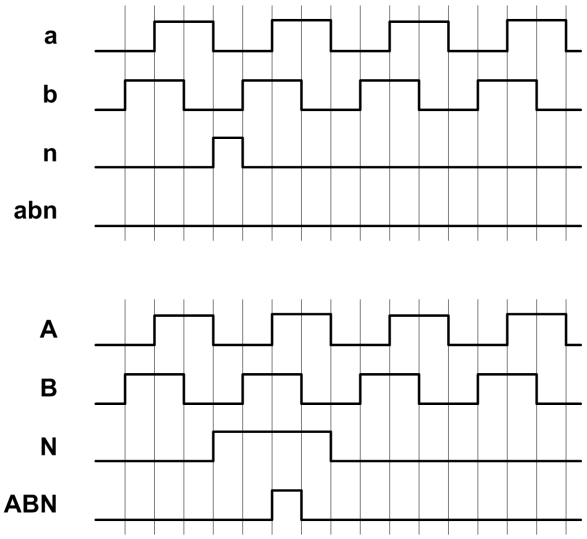


Figure 18: Encoder ABN Timing - high precise N pulse and less precise N pulse

4.7.3 Secondary Incremental ABN Encoder

For commutating a motor with FOC, the user selects a position sensor source (digital incremental encoder, digital Hall, analog Hall, analog incremental encoder, . . .) that is mounted close to the motor. The inner FOC loop controls torque and flux of the motor based on the measured phase currents and the electrical angle of the rotor.

The TMC4671 is equipped with a secondary incremental encoder interface. This secondary encoder interface is available as source for velocity control or position control. This is for applications where a motor with a gearing positions an object.

Info	The secondary incremental encoder is not available for commutation (phi_e) for the inner FOC. In others words, there is no electrical angle phi_e selectable from the secondary encoder.

4.7.4 Digital Hall Sensor Interface with optional Interim Position Interpolation

The digital Hall interface is the position sensor interface for digital Hall signals. The digital Hall signal interface first maps the digital Hall signals to an electrical position PHI_E_RAW. An offset PHI_E_OFFSET can be used to rotate the orientation of the Hall signal angle. The electrical angle PHI_E is for commutation. Optionally, the default electrical positions of the Hall sensors can be adjusted by writes into the associated registers.

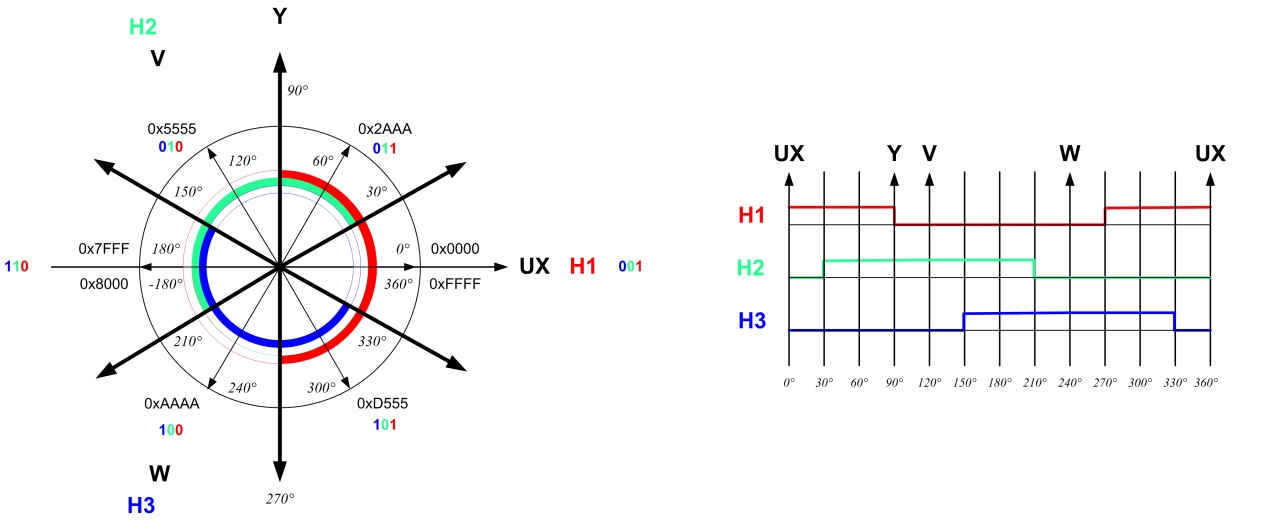


Figure 19: Hall Sensor Angles

Hall sensors give absolute positions within an electrical period with a resolution of 60° as 16 bit positions (s16 resp. u16) PHI. With activated interim Hall position interpolation, the user gets high resolution interim positions when the motor is running at a speed above 60 rpm.

4.7.5 Digital Hall Sensor - Interim Position Interpolation

For lower torque ripple the user can switch on the position interpolation of interim Hall positions. This function is useful for motors that are compatible with sine wave commutation, but equipped with digital Hall sensors.
When the position interpolation is switched on, it becomes active on speeds above 60 rpm. For lower speeds it automatically disables itself. This is especially important when the motor has to be at rest.
Hall sensor position interpolation might fail when Hall sensors are not properly placed in the motor. Please adjust Hall sensor positions for this case.

4.7.6 Digital Hall Sensors - Masking and Filtering

Sometimes digital Hall sensor signals get disturbed by switching events in the power stage. The TMC4671 can automatically mask switching distortions by correct setting of the HALL_MASKING register. When a switching event occurs, the Hall sensor signals are held for HALL_MASKING value times 10 ns. This way, Hall sensor distortions are eliminated. Uncorrelated distortions can be filtered via a digital filter of parameterizable length. If the input signal to the filter does not change for HALL_DIG_FILTER times 5 us, the signal can pass the filter. This filter eliminates issues with bouncing Hall signals.

4.7.7 Digital Hall Sensors together with Incremental Encoder

If a motor is equipped with both Hall sensors and incremental encoder, the Hall sensors can be used for the initialization as a low resolution absolute position sensor. Later on, the incremental encoder can be used as a high resolution sensor for commutation.

4.7.8 Analog Hall and Analog Encoder Interface (SinCos of 0° 90° or 0° 120° 240°)

An analog encoder interface is part of the decoder engine. It is able to handle analog position signals of 0° and 90° and of 0° 120° 240°. The analog decoder engine adds offsets and scales the raw analog encoder signals, while also calculating the electrical angle PHI_E from these analog position signals by an ATAN2 algorithm.

An individual signed offset is added to each associated raw ADC channel and scaled by its associated scaling factors according to

$$\text{AENC\_VALUE} = (\text{AENC\_RAW} + \text{AENC\_OFFSET}) \cdot \text{AENC\_SCALE} \tag{19}$$

In addition, the AENC_OFFSET is for conversion of unsigned ADC values into signed ADC values as required for the FOC.

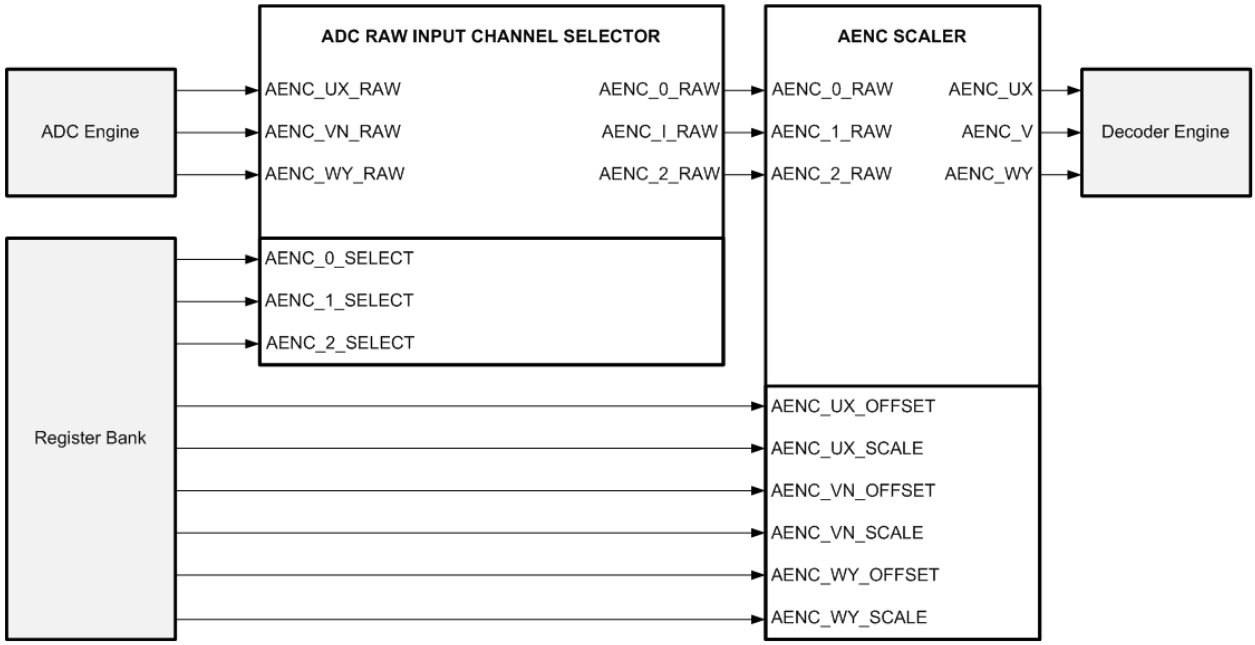


Figure 20: Analog Encoder (AENC) Selector & Scaler w/ Offset Correction

Info	The analog N pulse is just a raw ADC value. Handling of analog N pulse similar to N pulse handling of digital encoder N pulse is not implemented for analog encoder.

4.7.9 Analog Position Decoder (SinCos of 0°90° or 0°120°240°)

The extracted positions from the analog decoder are available for read out from registers.

4.7.9.1 Multi-Turn Counter

Electrical angles are mapped to a multi-turn position counter. The user can overwrite this multi-turn position for initialization purposes.

4.7.9.2 Encoder Engine Phi Selector

The angle selector selects the source for the commutation angle PHI_E. That electrical angle is available for commutation.

4.7.9.3 External Position Register

A register value written into the register bank via the application interface is available for commutation as well. With this, the user can interface to any encoder by just writing positions extracted from external encoder into this regulator. From the decoder engine point of view this is just one more selectable encoder source.

4.7.10 Encoder Initialization Support

The TMC4671 needs proper feedback for correct and stable operation. One main parameter is the commutation angle offset PHI_E_OFFSET. This offset must not be calculated when an absolute sensor system like analog or digital Hall sensors is used. All other supported feedback systems need to be initialized - their PHI_E_OFFSETs need to be identified. The user has several options to determine PHI_E_OFFSET with support of the TMC4671.

4.7.10.1 Encoder Initialization in Open-Loop Mode

In the case of a free driving motor, the motor can be switched to Open-Loop Mode. In this mode, the used commutation angle (PHI_OPEN_LOOP) can be used to match the measured PHI_E. This method is supported by the TMCL-IDE.

4.7.10.2 Encoder Initialization by Minimal Movement

If the motor shall not make a big move during initialization, the MOTION_MODE ENCODER_INIT_MINI_MOVE can be used which determines PHI_E_OFFSET by ramping up the flux and controlling the movement to a minimum by manipulating the used PHI_E_OFFSET. After the procedure is finished, the estimated PHI_E_OFFSET can be read from the register and used as the corresponding PHI_E_OFFSET for the feedback system.

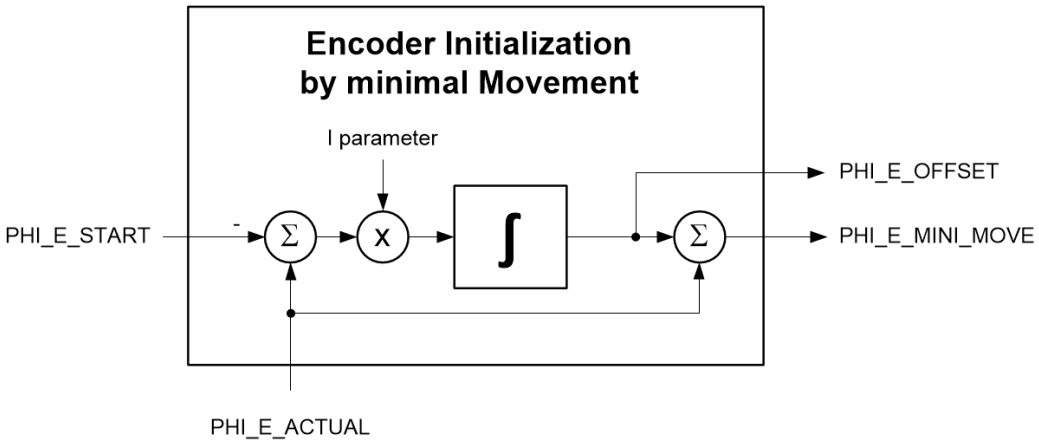


Figure 21: Encoder Initialization by minimal Movement

The flux ramping can be controlled by setting the U_D_INKR - which manipulates the slope of the ramp. The maximum voltage can be set by the parameter U_D_MAX. During operation, the current is monitored and the process is stopped when the current limit I_D_MAX is reached.

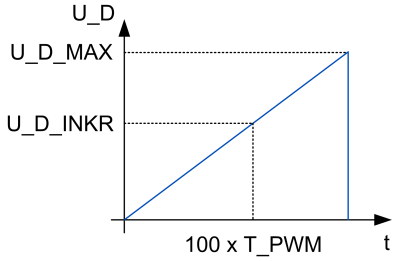


Figure 22: Flux Ramping

Info	For correct operation of this module a few parameters have to be set. Please try TRINAMIC TMCL-IDE Support for first usage and parameter tuning.

4.7.10.3 Encoder Initialization by Hall sensors

The TMC4671 can calculate PHI_E_OFFSET very precisely at a Hall state change for a second encoder system, when Hall sensors are correctly aligned. Therefore, the function needs to be enabled and calculate a new offset at the next Hall state change. After disabling of the module, the process can be started again. This function can also be used as a rough plausibility check during longer operation.

4.7.10.4 Encoder Initialization by N Pulse Detection

After determination of a correct offset, the value can be used again after power cycle. The encoder's N pulse can be used as reference for this. For starters the user can drive the motor in open-loop mode or by using digital Hall sensor signals. After passing the encoder's N pulse, the ABN encoder is initialized and can be used for operation.

4.7.11 Velocity Measurement

Servo control comprises position, velocity and current control. The position and the current are measured by separate sensors. The actual velocity has to be calculated by time discrete differentiation from the position signal. the user can choose a calculated position from the various encoder interfaces for velocity measurement by parameter VELOCITY_SELECTION.
The user can switch between two different velocity calculation algorithms with the parameter VELOCITY_METER_SELECTION. Default setting (VELOCITY_METER_SELECTION = 0) is the standard velocity meter, which calculates the velocity at a sampling rate of about 4369.067 Hz by differentiation. Output value is displayed in rpm (revolutions per minute). This option is recommended for usage with the standard PI controller structure.
By choosing the second option (VELOCITY_METER_SELECTION = 1), the sampling frequency is synchronized to the PWM frequency. This option is recommended for usage with the advanced PI controller structure. Otherwise, the controller structure might tend to be unstable due to non-matched sampling.
Velocity filters can be applied to reduce noise on velocity signals. Section 4.9 describes filtering opportunities in detail.

4.7.12 Reference Switches

The TMC4671 is equipped with three input pins for reference switches (REF_SW_L, REF_SW_H and REF_SW_R). These pins can be used to determine three reference positions. The TMC4671 displays the status of the reference switches in the register TMC_INPUTS_RAW and is able to store the actual position at rising edge of the corresponding signal. The signal polarities are programmable and the module reacts only on toggling the ENABLE register. The signals can be filtered with a configurable digital filter, which suppresses spike errors.

4.8 FOC23 Engine

Info	Support for the TMC4671 is integrated into the TMCL-IDE including wizards for set up and configuration. With the TMCL-IDE, configuration and operation can be done in a few steps and the user gets direct access to all registers of the TMC4671.

The FOC23 engine performs the inner current control loop for the torque current I_Q and the flux current I_D including the required transformations. Programmable limiters take care of clipping of interim results. Per default, the programmable circular limiter clips U_D and U_Q to U_D_R = $\sqrt{(2)}\cdot$ U_Q and U_R_R = $\sqrt{(2)}\cdot$ U_D. PI controllers perform the regulation tasks.

4.8.1 PI Controllers

PI controllers are used for current control and velocity control. A P controller is used for position control. The derivative part is not yet supported but might be added in the future. The user can choose between two PI controller structures: The classic PI controller structure, which is also used in the TMC4670, and the advanced PI controller structure. The advanced PI controller structure shows better performance in dynamics and is recommended for high performance applications. User can switch between controllers by setting register MODE_PID_TYPE. Controller type can not be switched individually for each cascade level.

4.8.2 PI Controller Calculations - Classic Structure

The PI controllers in the classic structure perform the following calculation

$$dXdT = P \cdot e + I \cdot \int_0^t e(t)\,dt \tag{20}$$

with

$$e = X\_TARGET - X \tag{21}$$

where X_TARGET stands for target flux, target torque, target velocity, or target position with error e, which is the difference between target value and actual values. The time constant dt is $1\mu s$ with the integral part is divided by 256.

Info Changing the I-parameter of the classic PI controller during operation causes the controller output to jump, as the control error is first integrated and then gained by the I parameter. Be careful during controller tuning or use the advanced PI controller structure instead. The normalization of the PI parameters might be changed due to low performance at high PWM frequencies. This will need small changes in user's application controller software.

4.8.3 PI Controller Calculations - Advanced Structure

The PI controllers in the advanced controller structure perform the calculation

$$dXdT = P \cdot e + \int_0^t P \cdot I \cdot e(t)\,dt \tag{22}$$

with

$$e = X\_TARGET - X \tag{23}$$

where X_TARGET represents target flux, target torque, target velocity, or target position with control error e, which is the difference between target value and actual values. The time constant dt is set according to the PWM period but can be downsampled for the velocity and position controller by register MODE_PID_SMPL. Velocity and position controller evaluation can be down-sampled by a constant factor when needed.

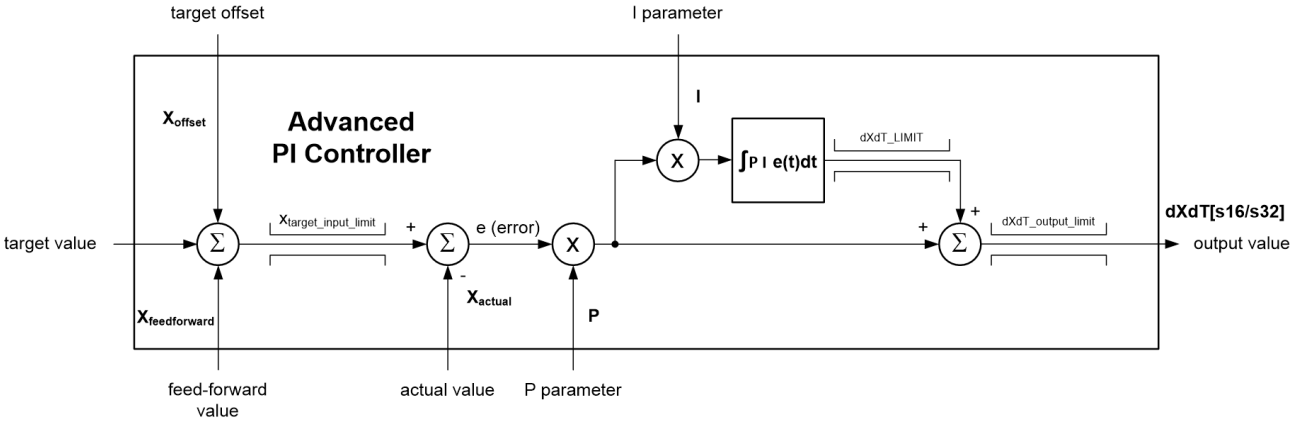


Figure 23: Advanced PI Controller

Info	The PI velocity controller will be given a derivative part (so it will be a PID controller) in a future version of the chip. Also, the normalization of the PI parameters might be changed due to low performance at high PWM frequencies. This will need changes in the user's application controller software.

Info	The P Factor in the advanced position controller is not properly scaled. Due to the high gain in velocity control loop, the position controller gain should be respectively low. The P Factor normalization of Q8.8 does not match these needs. This will be changed in a future version of the chip to a different Q format. This change will need changes in the user's application controller software. We recommend to use the classical PI control structure if performance is not sufficient.

4.8.4 PI Controller - Clipping

The limiting of target values for PI controllers and output values of PI controllers is programmable. Per power on default these limits are set to maximum values. During initialization, these limits should be set properly for correct operation and clipping.
The target input is clipped to X_TARGET_LIMIT. The output of a PI controller is named dXdT because it gives the desired derivative d/dt as a target value to the following stage: The position (x) controller gives velocity (dx/dt). The output of the PI Controller is clipped to dXdT_LIMIT. The error integral of (20) is clipped to dXdT_LIMIT / I in the classic controller structure, and the integrator output is clipped to dXdT_LIMIT in the advanced controller structure.

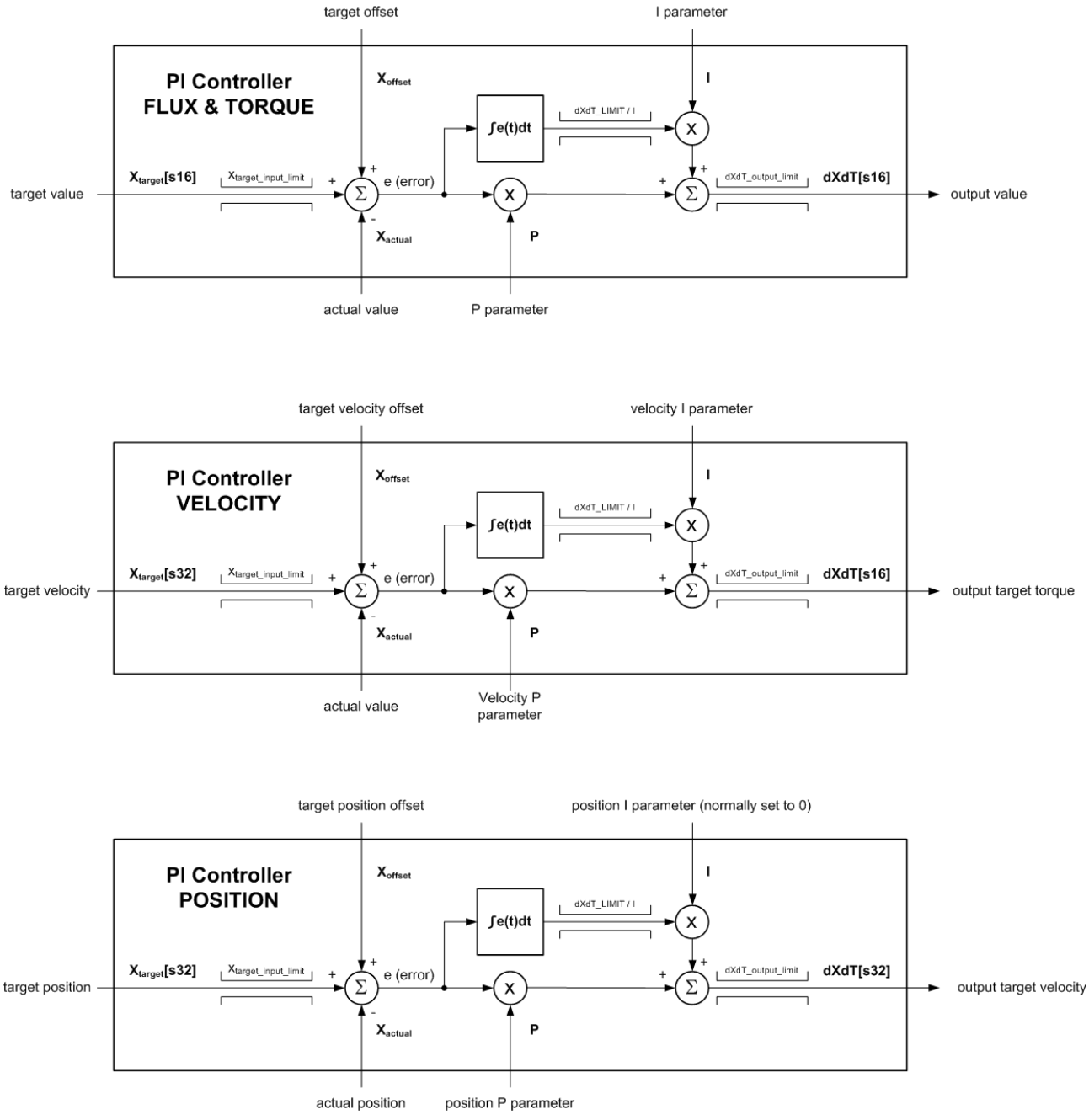


Figure 24: PI Architectures

4.8.5 PI Flux & PI Torque Controller

The P part is represented as q8.8 and I is the I part represented as q0.15.

4.8.6 PI Velocity Controller

The P part is represented as q8.8 and I is the I part represented as q0.15.

4.8.7 P Position Controller

For the position regulator, the P part is represented as q4.12 to be compatible with the high resolution positions - one single rotation is handled as an s16.

4.8.8 Inner FOC Control Loop - Flux & Torque

The inner FOC loop (figure 25) controls the flux current to the flux target value and the torque current to the desired torque target. The inner FOC loop performs the desired transformations according to figure 26 for 3-phase motors (FOC3). For 2-phase motors (FOC2) both Clarke (CLARKE) transformation and inverse Clarke (iCLARKE) are bypassed. For control of DC motors, transformations are bypassed and only the first full bridge (connected to X1 and X2) is used.

The inner FOC control loop gets a target torque value (I_Q_TARGET) which represents acceleration, the rotor position, and the measured currents as input data. Together with the programmed P and I parameters, the inner FOC loop calculates the target voltage values as input for the PWM engine.

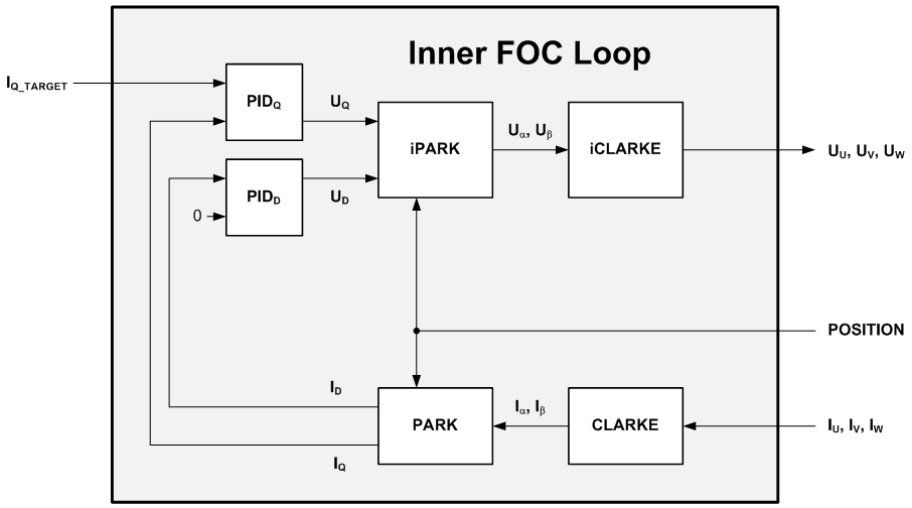


Figure 25: Inner FOC Control Loop

4.8.9 FOC Transformations and PI(D) for control of Flux & Torque

The Clarke transformation (CLARKE) maps three motor phase currents (I_U, I_V, I_W) to a two-dimensional coordinate system with two currents (I_α, I_β). Based on the actual rotor angle determined by an encoder or via sensorless techniques, the Park transformation (PARK) maps these two currents to a quasi-static coordinate system with two currents (I_D, I_Q). The current I_D represents flux and the current I_Q represents torque. The flux just pulls on the rotor but does not affect torque. The torque is affected by I_Q. Two PI controllers determine two voltages (U_D, U_Q) to drive desired currents for a target torque and a target flux. The determined voltages (U_D, U_Q) are re-transformed into the stator system by the inverse Park transformation (iPARK). The inverse Clarke Transformation (iCLARKE) transforms these two currents into three voltages (U_U, U_V, U_W). Theses three voltage are the input of the PWM engine to drive the power stage.

In case of the FOC2, Clarke transformation CLARKE and inverse Clarke Transformation iCLARKE are skipped.

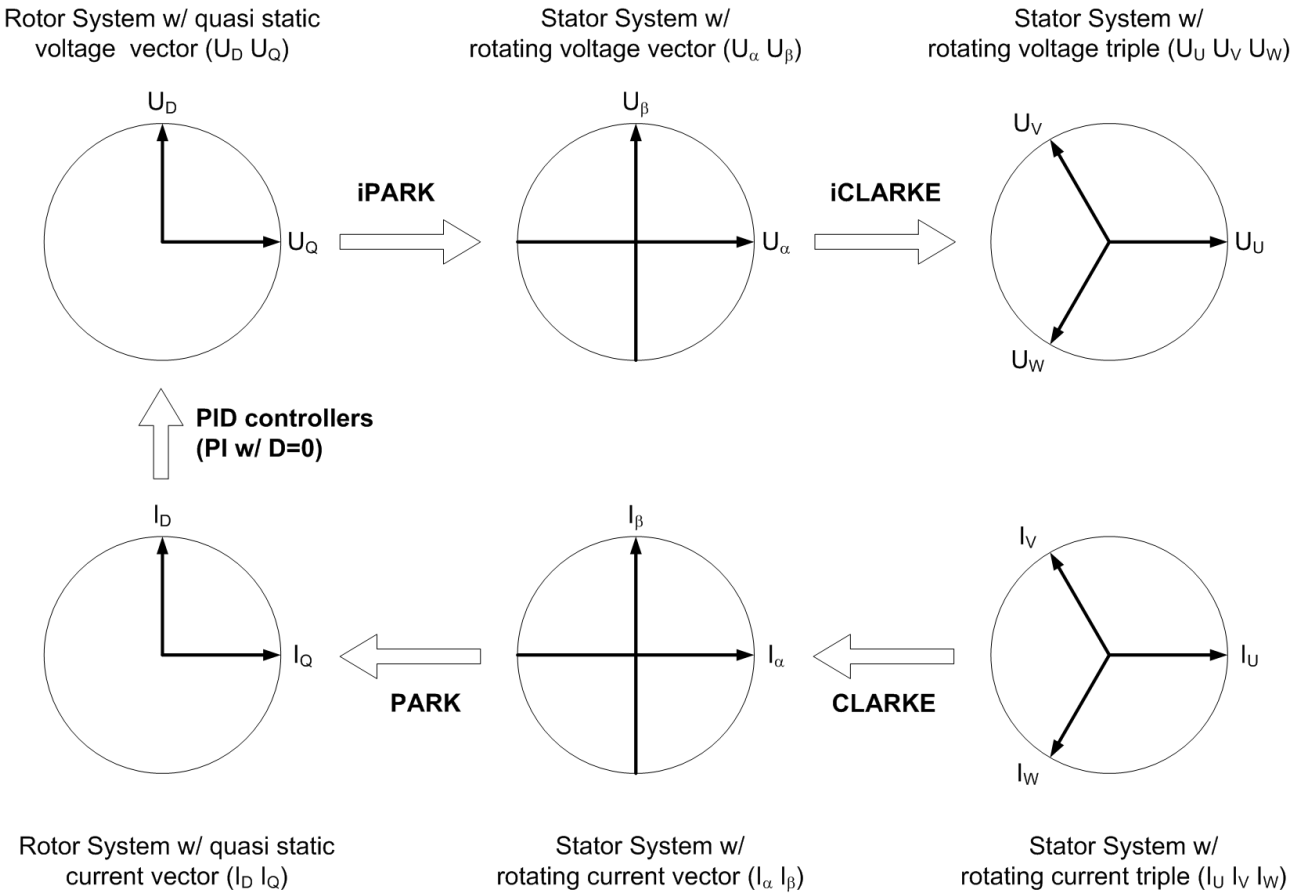


Figure 26: FOC3 Transformations (FOC2 just skips CLARKE and iCLARKE)

4.8.10 Motion Modes

The user can operate the TMC4671 in several motion modes. Standard motion modes are position control, velocity control and torque control, where target values are fed into the controllers via register access. The motion mode UD_UQ_EXTERN allows the user to set voltages for open-loop operation and for tests during setup.

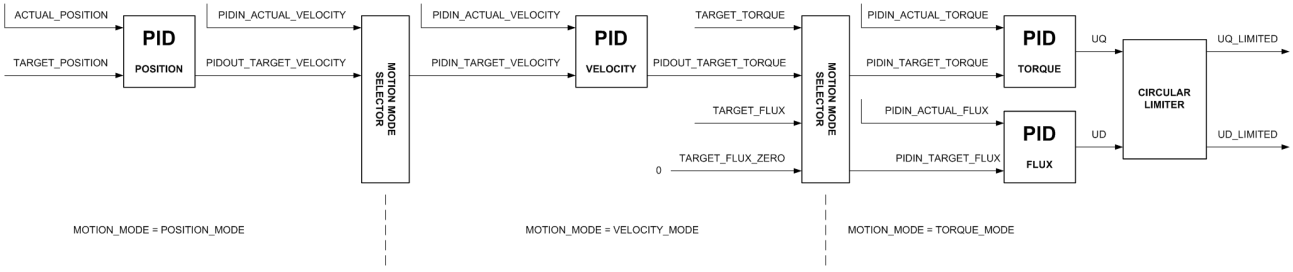


Figure 27: Standard Motion Modes

In position control mode, the user can feed the step and direction interface to generate a position target value for the controller cascade. Additional motion modes are the motion mode for encoder initialisation

(ENCODER_INIT_MINI_MOVE), and motion modes where target values are fed into the TMC4671 via PWM interface (Pin: PWM_IN) or analog input via pin AGPI_A.

There are additional motion modes, which are using input from the PWM_I input and the AGPI_A input. Input signals can be scaled via a standard scaler providing offset and gain correction. The interface can be configured via the registers SINGLE_PIN_IF_OFFSET_SCALE and SINGLE_PIN_IF_STATUS_CFG, where the status of the interface can be monitored as well. PWM input signals which are out of frequency range can be neglected. In case of wrong input data, last correct position is used or velocity and torque are set to zero.

Number	Motion Mode	Description
0	Stopped Mode	Disabling all controllers
1	Torque Mode	Standard Torque Control Mode
2	Velocity Mode	Standard Velocity Control Mode
3	Position Mode	Standard Position Control Mode
4	PRBS Flux Mode	PRBS Value is used as Target Flux Value for Ident.
5	PRBS Torque Mode	PRBS Value is used as Target Torque Value for Ident.
6	PRBS Velocity Mode	PRBS Value is used as Target Velocity Value for Ident.
7	PRBS Position Mode	PRBS Value is used as Target Position Value for Ident.
8	UQ UD Ext Mode	Voltage control mode (Software Mode)
9	Encoder Init Mini Move Mode	Encoder Initialization by minimal movement of the rotor.
10	AGPI_A Torque Mode	AGPI_A used as Target Torque value
11	AGPI_A Velocity Mode	AGPI_A used as Target Velocity value
12	AGPI_A Position Mode	AGPI_A used as Target Position value
13	PWM_I Torque Mode	PWM_I used as Target Torque value
14	PWM_I Velocity Mode	PWM_I used as Target Velocity value
15	PWM_I Position Mode	PWM_I used as Target Position value

Table 16: Motion Modes

4.8.11 Brake Chopper

During regenerative braking of the motor, current is driven into the DC link. If the power frontend is not actively controlled, the DC link voltage will rise. The brake chopper output pin (BRAKE) can be used for control of an external brake chopper, which burns energy over a brake resistor. The BRAKE pin is set to high for a complete PWM cycle if measured voltage is higher then ADC_VM_LIMIT_HIGH. Once active it will be deactivated when voltage drops below ADC_VM_LIMIT_LOW. This acts like a hysteresis. BRAKE can be deactivated by setting both registers to Zero. By setting proper values in the registers it is automatically enabled.

4.9 Filtering and Feed-Forward Control

The TMC4671 uses different filters for certain target and actual values. When using standard velocity meter, a standard velocity filter is used which is optimized for velocity signals from Hall sensors. Additional Biquad filters can be used to suppress measurement noise or damp resonances.

4.9.1 Biquad Filters

The TMC4671 uses standard biquad filters (standard IIR filter of second order) in the following structure.

$$Y(n) = X(n) \cdot b\_0 + X(n\text{-}1) \cdot b\_1 + X(n\text{-}2) \cdot b\_2 + Y(n\text{-}1) \cdot a\_1 + Y(n\text{-}2) \cdot a\_2 \tag{24}$$

In this equation X(n) is the actual input sample, while Y(n-1) is the filter output of the last cycle. All coefficients are S32 values and are normalized to a Q3.29 format. Users must take care of correct parametrization of the filter. There is no built-in plausibility or stability check. All filters can be disabled or enabled via register access. Biquad state variables are reset when parameters are changed. The TRINAMIC IDE supports parametrization with wizards.
A standard biquad filter has the following transfer function in the Laplace-Domain:

$$G(s) = \frac{b\_2\_cont \cdot s^2 + b\_1\_cont \cdot s + b\_0\_cont}{a\_2\_cont \cdot s^2 + a\_1\_cont \cdot s + a\_0\_cont} \tag{25}$$

The transfer function needs to be transformed to time discrete domain by Z-Transformation and coefficients need to be normalized. This is done by the following equations.

$$b\_2\_z = (b\_0\_cont \cdot T^2 + 2 \cdot b\_1\_cont \cdot T + 4 \cdot b\_2\_cont)/(T^2 - 2 \cdot a\_1\_cont \cdot T + 4 \cdot a\_2\_cont) \quad (26)$$

$$b\_1\_z = (2 \cdot b\_0\_cont \cdot T^2 - 8 \cdot b\_2\_cont)/(T^2 - 2 \cdot a\_1\_cont \cdot T + 4 \cdot a\_2\_cont) \quad (27)$$

$$b\_0\_z = (b\_0\_cont \cdot T^2 - 2 \cdot b\_1\_cont \cdot T + 4 \cdot b\_2\_cont)/(T^2 - 2 \cdot a\_1\_cont \cdot T + 4 \cdot a\_2\_cont) \quad (28)$$

$$a\_2\_z = (T^2 + 2 \cdot a\_1\_cont \cdot T + 4 \cdot a\_2\_cont)/(T^2 - 2 \cdot a\_1\_cont \cdot T + 4 \cdot a\_2\_cont) \quad (29)$$

$$a\_1\_z = (2 \cdot T^2 - 8 \cdot a\_2\_cont)/(T^2 - 2 \cdot a\_1\_cont \cdot T + 4 \cdot a\_2\_cont) \quad (30)$$

$$b\_0 = round(b\_0\_z \cdot 2^{29}) \quad (31)$$

$$b\_1 = round(b\_1\_z \cdot 2^{29}) \quad (32)$$

$$b\_2 = round(b\_2\_z \cdot 2^{29}) \quad (33)$$

$$a\_1 = round(-a\_1\_z \cdot 2^{29}) \quad (34)$$

$$a\_2 = round(-a\_2\_z \cdot 2^{29}) \quad (35)$$

while T is the sampling time according to PWM_MAX_COUNT · 10 ns and variables with index z are auxiliary variables.
There are four biquad filters in the control structure. Figure 28 illustrates their placement in the control structure.

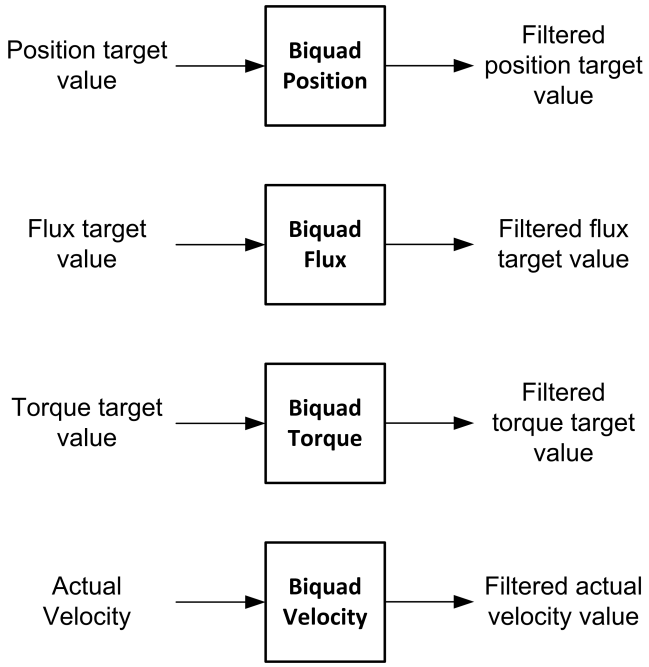


Figure 28: Biquad Filters

The biquad filter for the position target value is intended to be used as a low-pass filter for smoothening position input to the control structure. It is evaluated in every PWM cycle, or down-sampled according to the down-sampling factor for the velocity and position controllers. After powering on it is disabled.
The biquad filter for the flux target value is also intended to be used as a low-pass filter for input values from the user's microcontroller. Sampling frequency is fixed to the PWM frequency.
The biquad filter for the torque target value can be used as a low-pass filter for bandwidth limitation and noise suppression. Moreover, it can be designed to suppress a resonance or anti-resonance. Same

statements are correct for the velocity biquad filter. Both filters' sampling times are fixed to the PWM period.
The velocity target value biquad is configured as a second order low-pass with a cutoff frequency at 200 Hz - by default at a sampling frequency of 25 kHz. Biquad filters can be activated separately.

4.9.2 Standard Velocity Filter

By using the standard velocity measurement algorithm, the default velocity filter is enabled and can not be switched off. The standard velocity filter is a low-pass filter with a cutoff frequency of 20 Hz (slope of -20 dB/Decade). In this configuration, a new velocity is calculated at a sample rate of approx. 4369.067 Hz. This configuration is intended to be used in low-performance applications with a simple position feedback system like digital Hall sensors.

4.9.3 Feed-Forward Control Structure

The TMC4671 provides a feed-forward control structure for torque target value and velocity target value. The structure is intended to support controllers at high dynamic input profiles. It can be switched on when using the advanced PI controller structure. The feed-forward value is calculated with a DT1 (29) element. Each DT1 element can be parametrized with two parameters.

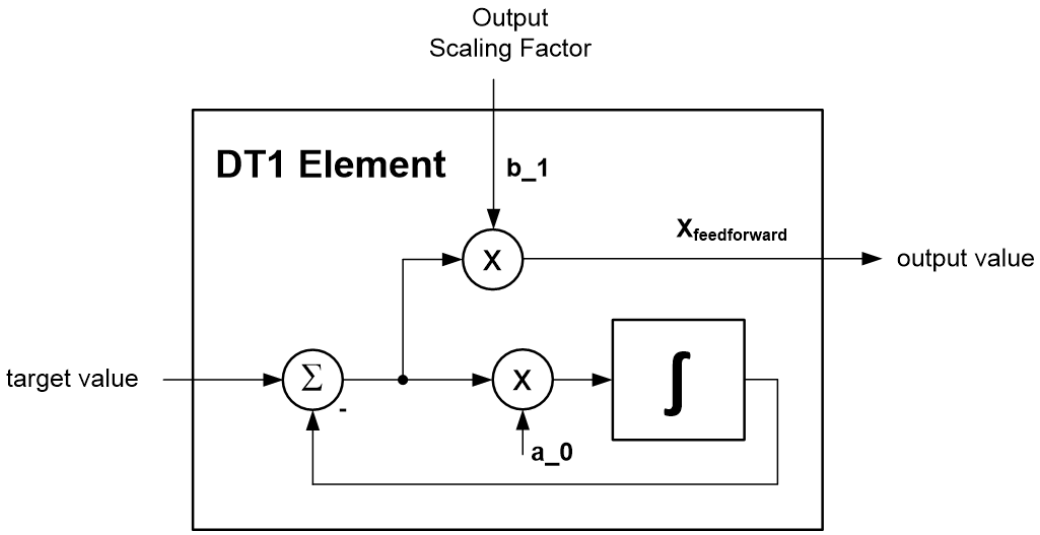


Figure 29: DT1 Element Structure

Equations:

$$e = X - int\_val \tag{36}$$

$$int\_val = \int e\, dt \tag{37}$$

$$Y = b\_1 \cdot e \tag{38}$$

Info Tuning of feed-forward control structure is supported by the TRINAMIC TMCL-IDE wizard.

The coefficients a_0 and b_1 are represented in Q2.30 format. Registers for parametrization of feed-forward control structure are feed_forward_velocity_gain, feed_forward_velocity_filter_constant, feed_forward_torque_gain, and feed_forward_torque_filter_constant.
The input target value to the velocity feed-forward entity is the filtered position target value. For the torque feed-forward entity the output of the velocity feed-forward entity is used. Sampling time for both entities' integrators is fixed to the PWM frequency.

4.10 PWM Engine

The PWM engine takes care of converting voltage vectors to pulse width modulated (PWM) control signals. These digital PWM signals control the gate drivers of the power stage. For a detailed description of the PWM control registers and PWM register control bits pls. refer section 6 page 56.

The ease-of-use PWM engine requires just a couple of parameter settings. Primarily, the polarities for the gate control signal of high-side and low-side must be set. The power on default PWM mode is 0, meaning PWM = OFF. For operation, the centered PWM mode must be switched on by setting the PWM mode to 7. A single bit switches the space vector PWM (SVPWM) on. For 3-phase PMSM, the SVPWM = ON gives more effective voltage. Nevertheless, for some applications it makes sense to switch the SVPWM = OFF to keep the star point voltage of a motor almost at rest.

4.10.1 PWM Polarities

The PWM polarities register (PWM_POLARITIES) controls the polarities of the control signals. Positive polarity for gate control means 1 represents ON and 0 represents OFF. The gate control signal polarities are individually programmable for low-side gate control and for high-side gate control. The PWM polarities register controls the polarity of other control signals as well.

4.10.2 PWM Frequency

The PWM counter maximum length register PWM_MAXCNT controls the PWM frequency. For a clock frequency fCLK = 25 MHz, the PWM frequency fPWM[Hz] = (4.0 · fCLK [Hz]) / (PWM_MAXCNT + 1). With fCLK = 25 MHz and power-on reset (POR) default of PWM_MAXCNT=3999, the PWM frequency fPWM = 25 kHz. The PWM frequency fPWM is recommended to be in the range of 25 kHz to 100 kHz by setting PWM_MAXCNT between 3999 to 999.

Note	The PWM frequency is the fundamental frequency of the control system. It can be changed at any time, also during motion for the classic PI controller structure. The advanced PI controller structure is tied to the PWM frequency and integrator gains have to be changed. Please make sure to set current measurement decimation rates to fit PWM period in high performance applications.
Info	Please be informed that later versions of the chip will support lower PWM frequencies. This might affect the user's software.

4.10.3 PWM Resolution

The base resolution of the PWM is 12 bit internally mapped to 16 bit range. The minimal PWM increment is 20ns due to the symmetrical PWM with 100 MHz counter frequency. MAX_PWMCNT = 4095 gives the full resolution of 12 bit with ≈ 25 kHz w/ fCLK=25 MHz. MAX_PWMCNT=2047 results in 11 bit resolution, but with ≈ 50kHz w/ fCLK=25 MHz. So the PWM_MAXCNT defines the PWM frequency, but also affects the resolution of the PWM.

Info	The PWM resolution might be increased in a future version of the chip.

4.10.4 PWM Modes

The power-on reset (POR) default of the PWM is OFF. The standard PWM scheme is the centered PWM. Passive braking and freewheeling modes are available on demand. Please refer to section 6 concerning the settings.

Info	The PWM modes might be changed in a future version of the chip to support so-called two-switch modulation or flat-bottom modulation.

4.10.5 Break-Before-Make (BBM)

One register controls BBM time for the high side, another register controls BBM time for the low side. The BBM times are programmable in 10 ns steps. The BBM time can be set to zero for gate drivers that have their own integrated BBM timers.

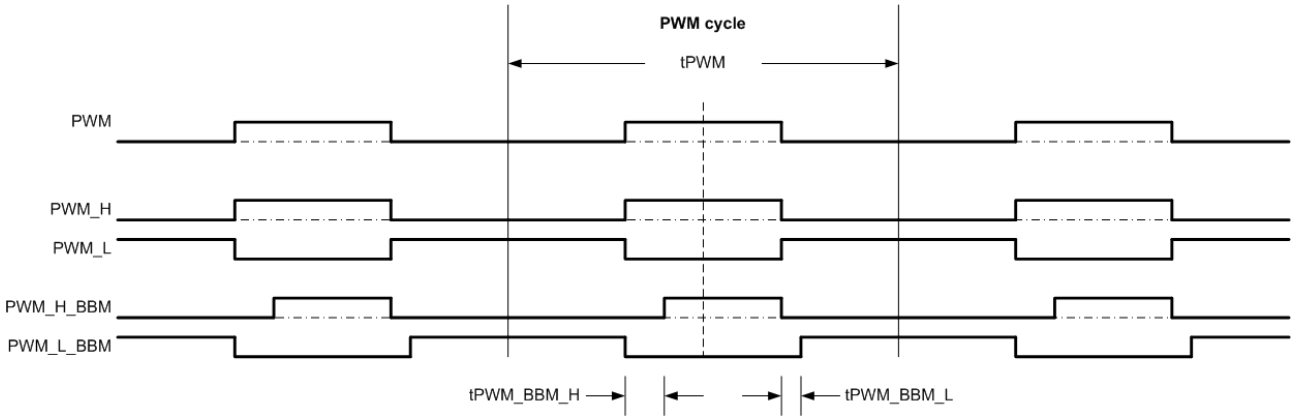


Figure 30: BBM Timing

Info	Measured BBM times at MOS-FET gates differ from programmed BBM times due to driver delays and possible additional gate driver BBM times. The programmed BBM times are for the digital control signals.
Note	Too short BBM times cause electrical shortcuts of the MOS-FET bridges - so called shoot through - that short the power supply and might damage the power stage and the power supply.
Info	BBM time registers might be changed in a future version of this chip to support longer BBM times then 2.55 us.

4.10.6 Space Vector PWM (SVPWM)

Note The Space Vector PWM does not allow higher voltage utilization. This will be fixed in next version of the chip.

A single bit enables the Space Vector PWM (SVPWM). No further settings are required for the space vector PWM - just ON or OFF. The power on default for the SVPWM is OFF. Space Vector PWM can be enabled to maximize voltage utilization in the case of an isolated star point of the motor. If the star point is not isolated, unintended current flows through the star point. Space Vector PWM is only used for three-phase motors. For other motors the SVPWM must be switched off.

5 Safety Functions

Different safety functions are integrated and mapped to status bits. A programmable mask register selects bits for activation of the STATUS output.

Internal hardware limiters for real time clipping and monitoring of interim values are available. LIMIT or LIMITS is part of register names of registers associated to internal limiters. Please refer to table 17.

Bit	Source
0	pid_x_target_limit
1	pid_x_target_ddt_limit
2	pid_x_errsum_limit
3	pid_x_output_limit
4	pid_v_target_limit
5	pid_v_target_ddt_limit
6	pid_v_errsum_limit
7	pid_v_output_limit
8	pid_id_target_limit
9	pid_id_target_ddt_limit
10	pid_id_errsum_limit
11	pid_id_output_limit
12	pid_iq_target_limit
13	pid_iq_target_ddt_limit
14	pid_iq_errsum_limit
15	pid_iq_output_limit
16	ipark_cirlim_limit_u_d
17	ipark_cirlim_limit_u_q
18	ipark_cirlim_limit_u_r
19	not_PLL_locked

20	ref_sw_r
21	ref_sw_h
22	ref_sw_l
23	——-
24	pwm_min
25	pwm_max
26	adc_i_clipped
27	adc_aenc_clipped
28	ENC_N
29	ENC2_N
30	AENC_N
31	wd_error

Table 17: Status Flags Register

All controllers have input limiters as offsets can be added to target values and they can be limited to stay in certain ranges. Also all controller outputs can be limited and the integrating parts (error sums) of the PI controllers are also limited to controller outputs. If d/dt-limiters are enabled they are also capable of limiting target values.
If one of these limiters gets active, the flag will go to high state. This is usually a normal operation, when controllers are working on the boarders of there working area. With STATUS_MASK register corresponding flags can be activated.
Other status flags go to high state whether the voltage limitation is reached (circular limiter in iPark transformation) or PWM is saturated (pwm_min and pwm_max). This is also usual operation as the current controller has to deal with voltage limitation at high velocity operation.
The user can also use the status output to generate an IRQ on reference switch or N-channel of encoder. Also ADC clipping can be monitored which is a good indicator of wrong or faulty behaviour.
Remaining wd_error status flag indicates an error on the clock input of the TMC4671 (see following section).

5.1 Watchdog

The TMC4671 uses an internal RC oscillator to monitor the clock input signal CLK. If during operation the CLK signal is lost, the user can program the TMC4671 for different responses via register WATCHDOG_CFG. Power on default action is: no action, otherwise the ENABLE_OUT signal can be removed to disable the power stage or the TMC4671 can be reset.

6 Register Map

The TMC4671 has an register address range of 128 addresses with registers up to 32 bit data width. Some registers hold 32 bit data, some hold 2 x 16 bit data and other hold combinations of data defined by data masks. This section descibes the register bank of the TMC4671.

Section 6.1 gives an overview over all registers and section 6.2 gives the detailed description of all registers.

6.1 Register Map Overview

Address	Registername	Access
$0x00_h$	CHIPINFO_DATA	R
$0x01_h$	CHIPINFO_ADDR	RW
$0x02_h$	ADC_RAW_DATA	R
$0x03_h$	ADC_RAW_ADDR	RW
$0x04_h$	dsADC_MCFG_B_MCFG_A	RW
$0x05_h$	dsADC_MCLK_A	RW
$0x06_h$	dsADC_MCLK_B	RW
$0x07_h$	dsADC_MDEC_B_MDEC_A	RW
$0x08_h$	ADC_I1_SCALE_OFFSET	RW
$0x09_h$	ADC_I0_SCALE_OFFSET	RW
$0x0A_h$	ADC_I_SELECT	RW
$0x0B_h$	ADC_I1_I0_EXT	RW
$0x0C_h$	DS_ANALOG_INPUT_STAGE_CFG	RW
$0x0D_h$	AENC_0_SCALE_OFFSET	RW
$0x0E_h$	AENC_1_SCALE_OFFSET	RW
$0x0F_h$	AENC_2_SCALE_OFFSET	RW
$0x11_h$	AENC_SELECT	RW
$0x12_h$	ADC_IWY_IUX	R
$0x13_h$	ADC_IV	R
$0x15_h$	AENC_WY_UX	R
$0x16_h$	AENC_VN	R
$0x17_h$	PWM_POLARITIES	RW
$0x18_h$	PWM_MAXCNT	RW
$0x19_h$	PWM_BBM_H_BBM_L	RW
$0x1A_h$	PWM_SV_CHOP	RW
$0x1B_h$	MOTOR_TYPE_N_POLE_PAIRS	RW
$0x1C_h$	PHI_E_EXT	RW

Address	Registername	Access
$0x1D_h$	PHI_M_EXT	RW
$0x1E_h$	POSITION_EXT	RW
$0x1F_h$	OPENLOOP_MODE	RW
$0x20_h$	OPENLOOP_ACCELERATION	RW
$0x21_h$	OPENLOOP_VELOCITY_TARGET	RW
$0x22_h$	OPENLOOP_VELOCITY_ACTUAL	RW
$0x23_h$	OPENLOOP_PHI	RWI
$0x24_h$	UQ_UD_EXT	RW
$0x25_h$	ABN_DECODER_MODE	RW
$0x26_h$	ABN_DECODER_PPR	RW
$0x27_h$	ABN_DECODER_COUNT	RW
$0x28_h$	ABN_DECODER_COUNT_N	RW
$0x29_h$	ABN_DECODER_PHI_E_PHI_M_OFFSET	RW
$0x2A_h$	ABN_DECODER_PHI_E_PHI_M	R
$0x2C_h$	ABN_2_DECODER_MODE	RW
$0x2D_h$	ABN_2_DECODER_PPR	RW
$0x2E_h$	ABN_2_DECODER_COUNT	RW
$0x2F_h$	ABN_2_DECODER_COUNT_N	RW
$0x30_h$	ABN_2_DECODER_PHI_M_OFFSET	RW
$0x31_h$	ABN_2_DECODER_PHI_M	R
$0x33_h$	HALL_MODE	RW
$0x34_h$	HALL_POSITION_060_000	RW
$0x35_h$	HALL_POSITION_180_120	RW
$0x36_h$	HALL_POSITION_300_240	RW
$0x37_h$	HALL_PHI_E_PHI_M_OFFSET	RW
$0x38_h$	HALL_DPHI_MAX	RW
$0x39_h$	HALL_PHI_E_INTERPOLATED_PHI_E	R
$0x3A_h$	HALL_PHI_M	R
$0x3B_h$	AENC_DECODER_MODE	RW
$0x3C_h$	AENC_DECODER_N_THRESHOLD	RW
$0x3D_h$	AENC_DECODER_PHI_A_RAW	R
$0x3E_h$	AENC_DECODER_PHI_A_OFFSET	RW
$0x3F_h$	AENC_DECODER_PHI_A	R
$0x40_h$	AENC_DECODER_PPR	RW

Address	Registername	Access
$0x41_h$	AENC_DECODER_COUNT	R
$0x42_h$	AENC_DECODER_COUNT_N	RW
$0x45_h$	AENC_DECODER_PHI_E_PHI_M_OFFSET	RW
$0x46_h$	AENC_DECODER_PHI_E_PHI_M	R
$0x47_h$	AENC_DECODER_POSITION	R
$0x4B_h$	PIDIN_VELOCITY_TARGET	R
$0x4C_h$	PIDIN_POSITION_TARGET	R
$0x4D_h$	CONFIG_DATA	RW
$0x4E_h$	CONFIG_ADDR	RW
$0x50_h$	VELOCITY_SELECTION	RW
$0x51_h$	POSITION_SELECTION	RW
$0x52_h$	PHI_E_SELECTION	RW
$0x53_h$	PHI_E	R
$0x54_h$	PID_FLUX_P_FLUX_I	RW
$0x56_h$	PID_TORQUE_P_TORQUE_I	RW
$0x58_h$	PID_VELOCITY_P_VELOCITY_I	RW
$0x5A_h$	PID_POSITION_P_POSITION_I	RW
$0x5C_h$	PID_TORQUE_FLUX_TARGET_DDT_LIMITS	RW
$0x5D_h$	PIDOUT_UQ_UD_LIMITS	RW
$0x5E_h$	PID_TORQUE_FLUX_LIMITS	RW
$0x5F_h$	PID_ACCELERATION_LIMIT	RW
$0x60_h$	PID_VELOCITY_LIMIT	RW
$0x61_h$	PID_POSITION_LIMIT_LOW	RW
$0x62_h$	PID_POSITION_LIMIT_HIGH	RW
$0x63_h$	MODE_RAMP_MODE_MOTION	RW
$0x64_h$	PID_TORQUE_FLUX_TARGET	RW
$0x65_h$	PID_TORQUE_FLUX_OFFSET	RW
$0x66_h$	PID_VELOCITY_TARGET	RW
$0x67_h$	PID_VELOCITY_OFFSET	RW
$0x68_h$	PID_POSITION_TARGET	RW
$0x69_h$	PID_TORQUE_FLUX_ACTUAL	R
$0x6A_h$	PID_VELOCITY_ACTUAL	R
$0x6B_h$	PID_POSITION_ACTUAL	RW
$0x6C_h$	PID_ERROR_DATA	R

Address	Registername	Access
$0x6D_h$	PID_ERROR_ADDR	RW
$0x6E_h$	INTERIM_DATA	RW
$0x6F_h$	INTERIM_ADDR	RW
$0x74_h$	WATCHDOG_CFG	RW
$0x75_h$	ADC_VM_LIMITS	RW
$0x76_h$	TMC4671_INPUTS_RAW	R
$0x77_h$	TMC4671_OUTPUTS_RAW	R
$0x78_h$	STEP_WIDTH	RW
$0x79_h$	UART_BPS	RW
$0x7A_h$	UART_ADDRS	RW
$0x7B_h$	GPIO_dsADCI_CONFIG	RW
$0x7C_h$	STATUS_FLAGS	RW
$0x7D_h$	STATUS_MASK	RW

Table 18: TMC4671 Registers

6.2 Register Map Full

Register Map for TMC4671					
Address	Registername				Access
$0x00_h$	CHIPINFO_DATA				R
	Variant 0				
	Mask	Name			Type
	$0xFFFFFFFF_h$	SI_TYPE			ASCII
		Min	Max	Default	Unit
		0	4294967295	0	
		Hardware type (ASCII).			
	Variant 1				
	Mask	Name			Type
	$0xFFFFFFFF_h$	SI_VERSION			Version
		Min	Max	Default	Unit
		0	4294967295	0	
		Hardware version (u16.u16).			
	Variant 2				
	Mask	Name			Type
	$0xFFFFFFFF_h$	SI_DATE			Date
		Min	Max	Default	Unit
		0	4294967295	0	
		Hardware date (nibble wise date stamp yyyymmdd).			
	Variant 3				
	Mask	Name			Type
	$0xFFFFFFFF_h$	SI_TIME			Time
		Min	Max	Default	Unit
		0	16777215	0	
		Hardware time (nibble wise time stamp –hhmmss)			
	Variant 4				
	Mask	Name			Type
	$0xFFFFFFFF_h$	SI_VARIANT			Unsigned
		Min	Max	Default	Unit
		0	4294967295	0	
	Variant 5				
	Mask	Name			Type

Address	Registername				Access
	0xFFFFFFFF$_h$	SI_BUILD		Unsigned	
		Min	Max	Default	Unit
		0	4294967295	0	
0x01$_h$	CHIPINFO_ADDR				RW
	Mask	Name		Type	
	0x000000FF$_h$	CHIP_INFO_ADDRESS		Choice	
		Min	Max	Default	Unit
		0	5	0	
		0: SI_TYPE 1: SI_VERSION 2: SI_DATE 3: SI_TIME 4: SI_VARIANT 5: SI_BUILD			
0x02$_h$	ADC_RAW_DATA				R
	Variant 0				
	Mask	Name		Type	
	0x0000FFFF$_h$	ADC_I0_RAW		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
		Raw phase current I0			
	Mask	Name		Type	
	0xFFFF0000$_h$	ADC_I1_RAW		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
		Raw phase current I1			
	Variant 1				
	Mask	Name		Type	
	0x0000FFFF$_h$	ADC_VM_RAW		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
		aw supply voltage value.			
	Mask	Name		Type	
	0xFFFF0000$_h$	ADC_AGPI_A_RAW		Unsigned	
		Min	Max	Default	Unit

Address	Registername				Access
		0	65535	0	
		Raw analog gpi A value.			
		Variant 2			
	Mask	Name			Type
	0x0000FFFF$_h$	ADC_AGPI_B_RAW			Unsigned
		Min	Max	Default	Unit
		0	65535	0	
		Raw analog gpi B value.			
	Mask	Name			Type
	0xFFFF0000$_h$	ADC_AENC_UX_RAW			Unsigned
		Min	Max	Default	Unit
		0	65535	0	
		Raw analog encoder signal.			
		Variant 3			
	Mask	Name			Type
	0x0000FFFF$_h$	ADC_AENC_VN_RAW			Unsigned
		Min	Max	Default	Unit
		0	65535	0	
		Raw analog encoder signal.			
	Mask	Name			Type
	0xFFFF0000$_h$	ADC_AENC_WY_RAW			Unsigned
		Min	Max	Default	Unit
		0	65535	0	
		Raw analog encoder signal.			
0x03$_h$	ADC_RAW_ADDR				RW
	Mask	Name			Type
	0x000000FF$_h$	ADC_RAW_ADDR			Choice
		Min	Max	Default	Unit
		0	3	0	
		0: ADC_I1_RAW & ADC_I0_RAW 1: ADC_AGPI_A_RAW & ADC_VM_RAW 2: ADC_AENC_UX_RAW & ADC_AGPI_B_RAW 3: ADC_AENC_WY_RAW & ADC_AENC_VN_RAW			
0x04$_h$	dsADC_MCFG_B_MCFG_A				RW
	Mask	Name			Type

Address	Registername				Access
	0x00000003$_h$	cfg_dsmodulator_a		Choice	
		Min	Max	Default	Unit
		0	3	0	
		0: int. dsMOD 1: ext. dsMOD with MCLK input 2: ext. dsMOD with MCLK output 3: ext. dsMOD with ext. CMP			
	Mask	Name		Type	
	0x00000004$_h$	mclk_polarity_a		Bool	
		Min	Max	Default	Unit
		0	1	0	
		0: Data is sampled on rising edge 1: Data is sampled on falling edge			
	Mask	Name		Type	
	0x00000008$_h$	mdat_polarity_a		Bool	
		Min	Max	Default	Unit
		0	1	0	
		0: MDAT is not inverted 1: MDAT is inverted			
	Mask	Name		Type	
	0x00000010$_h$	sel_nclk_mclk_i_a		Bool	
		Min	Max	Default	Unit
		0	1	0	
		0: MCLK is used (divided clock) 1: CLK (100 MHz) is used			
	Mask	Name		Type	
	0x000000FF00$_h$	blanking_a		Unsigned	
		Min	Max	Default	Unit
		0	255	0	
	Mask	Name		Type	
	0x00030000$_h$	cfg_dsmodulator_b		Choice	
		Min	Max	Default	Unit
		0	3	0	
		0: int. dsMOD 1: ext. dsMOD with MCLK input			

Address	Registername				Access
		2: ext. dsMOD with MCLK output 3: ext. dsMOD with ext. CMP			
	Mask	Name		Type	
	0x00040000$_h$	mclk_polarity_b		Bool	
		Min	Max	Default	Unit
		0	1	0	
		0: Data is sampled on rising edge 1: Data is sampled on falling edge			
	Mask	Name		Type	
	0x00080000$_h$	mdat_polarity_b		Bool	
		Min	Max	Default	Unit
		0	1	0	
		0: MDAT is not inverted 1: MDAT is inverted			
	Mask	Name		Type	
	0x00100000$_h$	sel_nclk_mclk_i_b		Bool	
		Min	Max	Default	Unit
		0	1	0	
		0: MCLK is used (divided clock) 1: CLK (100 MHz) is used			
	Mask	Name		Type	
	0xFF000000$_h$	blanking_b		Unsigned	
		Min	Max	Default	Unit
		0	255	0	
0x05$_h$	dsADC_MCLK_A				RW
	Mask	Name		Type	
	0xFFFFFFFF$_h$	dsADC_MCLK_A		Unsigned	
		Min	Max	Default	Unit
		0	4294967295	214748365	
		fMCLK_A = 2^{31} / (fCLK * (dsADC_MCLK_A+1)), dsADC_MCLK_A = (2^{31} / (fMCLK * fCLK)) - 1			
0x06$_h$	dsADC_MCLK_B				RW
	Mask	Name		Type	
	0xFFFFFFFF$_h$	dsADC_MCLK_B		Unsigned	
		Min	Max	Default	Unit

Address	Registername				Access
		0	4294967295	214748365	
		fMCLK_B = 2^{31} / (fCLK * (dsADC_MCLK_B+1)), dsADC_MCLK_B = (2^{31} / (fMCLK * fCLK)) - 1			
0x07$_h$	dsADC_MDEC_B_MDEC_A				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	dsADC_MDEC_A		Unsigned	
		Min	Max	Default	Unit
		0	65535	256	
	Mask	Name		Type	
	0xFFFF0000$_h$	dsADC_MDEC_B		Unsigned	
		Min	Max	Default	Unit
		0	65535	256	
0x08$_h$	ADC_I1_SCALE_OFFSET				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	ADC_I1_OFFSET		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
		Offset for current ADC channel 1.			
	Mask	Name		Type	
	0xFFFF0000$_h$	ADC_I1_SCALE		Signed	
		Min	Max	Default	Unit
		-32768	32767	256	
		Scaling factor for current ADC channel 1.			
0x09$_h$	ADC_I0_SCALE_OFFSET				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	ADC_I0_OFFSET		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
		Offset for current ADC channel 0.			
	Mask	Name		Type	
	0xFFFF0000$_h$	ADC_I0_SCALE		Signed	
		Min	Max	Default	Unit
		-32768	32767	256	
		Scaling factor for current ADC channel 0.			
0x0A$_h$	ADC_I_SELECT				RW

Address	Registername				Access
	Mask	Name		Type	
	$0x000000FF_h$	ADC_I0_SELECT		Choice	
		Min	Max	Default	Unit
		0	3	0	
		Select input for raw current ADC_I0_RAW. 0: ADCSD_I0_RAW (sigma delta ADC) 1: ADCSD_I1_RAW (sigma delta ADC) 2: ADC_I0_EXT (from register) 3: ADC_I1_EXT (from register)			
	Mask	Name		Type	
	$0x0000FF00_h$	ADC_I1_SELECT		Choice	
		Min	Max	Default	Unit
		0	3	1	
		Select input for raw current ADC_I1_RAW. 0: ADCSD_I0_RAW (sigma delta ADC) 1: ADCSD_I1_RAW (sigma delta ADC) 2: ADC_I0_EXT (from register) 3: ADC_I1_EXT (from register)			
	Mask	Name		Type	
	$0x03000000_h$	ADC_I_UX_SELECT		Choice	
		Min	Max	Default	Unit
		0	2	0	
		0: UX = ADC_I0 (default) 1: UX = ADC_I1 2: UX = ADC_I2			
	Mask	Name		Type	
	$0x0C000000_h$	ADC_I_V_SELECT		Choice	
		Min	Max	Default	Unit
		0	2	1	
		0: V = ADC_I0 1: V = ADC_I1 (default) 2: V = ADC_I2			
	Mask	Name		Type	
	$0x30000000_h$	ADC_I_WY_SELECT		Choice	
		Min	Max	Default	Unit

Address	Registername				Access
		0	2	2	
		0: WY = ADC_I0 1: WY = ADC_I1 2: WY = ADC_I2 (default)			
$0x0B_h$	ADC_I1_I0_EXT				RW
	Mask	Name		Type	
	$0x0000FFFF_h$	ADC_I0_EXT		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
		Register for write of ADC_I0 value from external source (eg. CPU).			
	Mask	Name		Type	
	$0xFFFF0000_h$	ADC_I1_EXT		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
		Register for write of ADC_I1 value from external source (eg. CPU).			
$0x0C_h$	DS_ANALOG_INPUT_STAGE_CFG				RW
	Mask	Name		Type	
	$0x0000000F_h$	ADC_I0		Choice	
		Min	Max	Default	Unit
		0	7	0	
		0: INP vs. INN 1: GND vs. INN 2: VDD/4 3: 3*VDD/4 4: INP vs. GND 5: VDD/2 6: VDD/4 7: 3*VDD/4			
	Mask	Name		Type	
	$0x000000F0_h$	ADC_I1		Choice	
		Min	Max	Default	Unit
		0	7	0	
		0: INP vs. INN 1: GND vs. INN			

Address	Registername				Access
	2: VDD/4 3: 3*VDD/4 4: INP vs. GND 5: VDD/2 6: VDD/4 7: 3*VDD/4				
	Mask	Name		Type	
	0x00000F00$_h$	ADC_VM		Choice	
		Min	Max	Default	Unit
		0	7	0	
	0: INP vs. INN 1: GND vs. INN 2: VDD/4 3: 3*VDD/4 4: INP vs. GND 5: VDD/2 6: VDD/4 7: 3*VDD/4				
	Mask	Name		Type	
	0x0000F000$_h$	ADC_AGPI_A		Choice	
		Min	Max	Default	Unit
		0	7	0	
	0: INP vs. INN 1: GND vs. INN 2: VDD/4 3: 3*VDD/4 4: INP vs. GND 5: VDD/2 6: VDD/4 7: 3*VDD/4				
	Mask	Name		Type	
	0x000F0000$_h$	ADC_AGPI_B		Choice	
		Min	Max	Default	Unit
		0	7	0	
	0: INP vs. INN				

Address	Registername				Access
		1: GND vs. INN 2: VDD/4 3: 3*VDD/4 4: INP vs. GND 5: VDD/2 6: VDD/4 7: 3*VDD/4			
	Mask	Name		Type	
	$0x00F00000_h$	ADC_AENC_UX		Choice	
		Min	Max	Default	Unit
		0	7	0	
		0: INP vs. INN 1: GND vs. INN 2: VDD/4 3: 3*VDD/4 4: INP vs. GND 5: VDD/2 6: VDD/4 7: 3*VDD/4			
	Mask	Name		Type	
	$0x0F000000_h$	ADC_AENC_VN		Choice	
		Min	Max	Default	Unit
		0	7	0	
		0: INP vs. INN 1: GND vs. INN 2: VDD/4 3: 3*VDD/4 4: INP vs. GND 5: VDD/2 6: VDD/4 7: 3*VDD/4			
	Mask	Name		Type	
	$0xF0000000_h$	ADC_AENC_WY		Choice	
		Min	Max	Default	Unit
		0	7	0	

Address	Registername				Access
		0: INP vs. INN 1: GND vs. INN 2: VDD/4 3: 3*VDD/4 4: INP vs. GND 5: VDD/2 6: VDD/4 7: 3*VDD/4			
$0x0D_h$	AENC_0_SCALE_OFFSET				RW
	Mask	Name		Type	
	$0x0000FFFF_h$	AENC_0_OFFSET		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
		Offset for Analog Encoder ADC channel 0.			
	Mask	Name		Type	
	$0xFFFF0000_h$	AENC_0_SCALE		Signed	
		Min	Max	Default	Unit
		-32768	32767	256	
		Scaling factor for Analog Encoder ADC channel 0.			
$0x0E_h$	AENC_1_SCALE_OFFSET				RW
	Mask	Name		Type	
	$0x0000FFFF_h$	AENC_1_OFFSET		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
		Offset for Analog Encoder ADC channel 1.			
	Mask	Name		Type	
	$0xFFFF0000_h$	AENC_1_SCALE		Signed	
		Min	Max	Default	Unit
		-32768	32767	256	
		Scaling factor for Analog Encoder ADC channel 1.			
$0x0F_h$	AENC_2_SCALE_OFFSET				RW
	Mask	Name		Type	
	$0x0000FFFF_h$	AENC_2_OFFSET		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	

Address	Registername				Access
		Offset for Analog Encoder ADC channel 2.			
	Mask	Name			Type
	0xFFFF0000$_h$	AENC_2_SCALE			Signed
		Min	Max	Default	Unit
		-32768	32767	256	
		Scaling factor for Analog Encoder ADC channel 2.			
0x11$_h$	AENC_SELECT				RW
	Mask	Name			Type
	0x000000FF$_h$	AENC_0_SELECT			Choice
		Min	Max	Default	Unit
		0	2	0	
		Select analog encoder ADC channel for raw analog encoder signal AENC_0_RAW. 0: AENC_UX_RAW (default) 1: AENC_VN_RAW 2: AENC_WY_RAW			
	Mask	Name			Type
	0x0000FF00$_h$	AENC_1_SELECT			Choice
		Min	Max	Default	Unit
		0	2	1	
		Select analog encoder ADC channel for raw analog encoder signal AENC_1_RAW. 0: AENC_UX_RAW 1: AENC_VN_RAW (default) 2: AENC_WY_RAW			
	Mask	Name			Type
	0x00FF0000$_h$	AENC_2_SELECT			Choice
		Min	Max	Default	Unit
		0	2	2	
		Select analog encoder ADC channel for raw analog encoder signal AENC_2_RAW. 0: AENC_UX_RAW 1: AENC_VN_RAW 2: AENC_WY_RAW (default)			
0x12$_h$	ADC_IWY_IUX				R
	Mask	Name			Type

Address	Registername				Access
	0x0000FFFF$_h$	ADC_IUX		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		Register of scaled current ADC value including signed added offset as input for the FOC.			
	Mask	Name		Type	
	0xFFFF0000$_h$	ADC_IWY		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		Register of scaled current ADC value including signed added offset as input for the FOC.			
0x13$_h$	ADC_IV				R
	Mask	Name		Type	
	0x0000FFFF$_h$	ADC_IV		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		Register of scaled current ADC value including signed added offset as input for the FOC.			
0x15$_h$	AENC_WY_UX				R
	Mask	Name		Type	
	0x0000FFFF$_h$	AENC_UX		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		Register of scaled analog encoder value including signed added offset as input for the interpolator.			
	Mask	Name		Type	
	0xFFFF0000$_h$	AENC_WY		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		Register of scaled analog encoder value including signed added offset as input for the interpolator.			
0x16$_h$	AENC_VN				R
	Mask	Name		Type	
	0x0000FFFF$_h$	AENC_VN		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	

Address	Registername				Access
		Register of scaled analog encoder value including signed added offset as input for the interpolator.			
$0x17_h$	PWM_POLARITIES				RW
	Mask	Name		Type	
	$0x00000001_h$	PWM_POLARITIES[0]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		polarity of Low Side (LS) gate control signal 0: off 1: on			
	Mask	Name		Type	
	$0x00000002_h$	PWM_POLARITIES[1]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		polarity of High Side (HS) gate control signal 0: off 1: on			
$0x18_h$	PWM_MAXCNT				RW
	Mask	Name		Type	
	$0x0000FFFF_h$	PWM_MAXCNT		Unsigned	
		Min	Max	Default	Unit
		0	65535	3999	
		PWM maximum (count-1), PWM frequency is fPWM[Hz] = 100MHz/(PWM_MAXCNT+1)			
$0x19_h$	PWM_BBM_H_BBM_L				RW
	Mask	Name		Type	
	$0x000000FF_h$	PWM_BBM_L		Unsigned	
		Min	Max	Default	Unit
		0	255	20	
		Break Before Make time tBBM_L[10ns] for low side MOS-FET gate control			
	Mask	Name		Type	
	$0x0000FF00_h$	PWM_BBM_H		Unsigned	
		Min	Max	Default	Unit
		0	255	20	

Address	Registername				Access
		Break Before Make time tBBM_H[10ns] for high side MOS-FET gate control			
$0x1A_h$	PWM_SV_CHOP				RW
	Mask	Name		Type	
	$0x000000FF_h$	PWM_CHOP		Choice	
		Min	Max	Default	Unit
		0	7	0	
		PWM chopper mode, defining how to chopper 0: PWM = OFF, free running 1: PWM = OFF, Low Side (LS) permanent = ON 2: PWM = OFF, High Side (HS) permanent = ON 3: PWM off, free running 4: PWM off, free running 5: PWM low side (LS) chopper only, high side (HS) off; not suitable for FOC 6: PWM high side (HS) chopper only, low side (LS) off; not suitable for FOC 7: centered PWM for FOC			
	Mask	Name		Type	
	$0x00000100_h$	PWM_SV		Bool	
		Min	Max	Default	Unit
		0	1	0	
		use Space Vector PWM 0: Space Vector PWM disabled 1: Space Vector PWM enabled			
$0x1B_h$	MOTOR_TYPE_N_POLE_PAIRS				RW
	Mask	Name		Type	
	$0x0000FFFF_h$	N_POLE_PAIRS		Unsigned	
		Min	Max	Default	Unit
		1	65535	1	
		Number n of pole pairs of the motor for calcualtion phi_e = phi_m / N_POLE_PAIRS.			
	Mask	Name		Type	
	$0x00FF0000_h$	MOTOR_TYPE		Choice	
		Min	Max	Default	Unit
		0	3	0	

<table>
<tr><th>Address</th><th colspan="5">Registername</th><th>Access</th></tr>
<tr><td></td><td></td><td colspan="4">0: No motor
1: Single phase DC motor
2: Two phase Stepper motor
3: Three phase BLDC motor</td><td></td></tr>
<tr><td>0x1C$_h$</td><td colspan="5">PHI_E_EXT</td><td>RW</td></tr>
<tr><td></td><td>Mask</td><td colspan="3">Name</td><td>Type</td><td></td></tr>
<tr><td></td><td>0x0000FFFF$_h$</td><td colspan="3">PHI_E_EXT</td><td>Signed</td><td></td></tr>
<tr><td></td><td></td><td>Min</td><td>Max</td><td>Default</td><td>Unit</td><td></td></tr>
<tr><td></td><td></td><td>-32768</td><td>32767</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td></td><td colspan="4">Electrical angle phi_e_ext for external writing into this register.</td><td></td></tr>
<tr><td>0x1D$_h$</td><td colspan="5">PHI_M_EXT</td><td>RW</td></tr>
<tr><td></td><td>Mask</td><td colspan="3">Name</td><td>Type</td><td></td></tr>
<tr><td></td><td>0x0000FFFF$_h$</td><td colspan="3">PHI_M_EXT</td><td>Signed</td><td></td></tr>
<tr><td></td><td></td><td>Min</td><td>Max</td><td>Default</td><td>Unit</td><td></td></tr>
<tr><td></td><td></td><td>-32768</td><td>32767</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td></td><td colspan="4">Mechanical angle phi_m_ext for external writing into this register.</td><td></td></tr>
<tr><td>0x1E$_h$</td><td colspan="5">POSITION_EXT</td><td>RW</td></tr>
<tr><td></td><td>Mask</td><td colspan="3">Name</td><td>Type</td><td></td></tr>
<tr><td></td><td>0xFFFFFFFF$_h$</td><td colspan="3">POSITION_EXT</td><td>Signed</td><td></td></tr>
<tr><td></td><td></td><td>Min</td><td>Max</td><td>Default</td><td>Unit</td><td></td></tr>
<tr><td></td><td></td><td>-2147483648</td><td>2147483647</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td></td><td colspan="4">Mechanical (multi turn) position for external writing into this register.</td><td></td></tr>
<tr><td>0x1F$_h$</td><td colspan="5">OPENLOOP_MODE</td><td>RW</td></tr>
<tr><td></td><td>Mask</td><td colspan="3">Name</td><td>Type</td><td></td></tr>
<tr><td></td><td>0x00001000$_h$</td><td colspan="3">OPENLOOP_PHI_DIRECTION</td><td>Bool</td><td></td></tr>
<tr><td></td><td></td><td>Min</td><td>Max</td><td>Default</td><td>Unit</td><td></td></tr>
<tr><td></td><td></td><td>0</td><td>1</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td></td><td colspan="4">Open loop phi direction.
0: positive
1: negative</td><td></td></tr>
<tr><td>0x20$_h$</td><td colspan="5">OPENLOOP_ACCELERATION</td><td>RW</td></tr>
<tr><td></td><td>Mask</td><td colspan="3">Name</td><td>Type</td><td></td></tr>
<tr><td></td><td>0xFFFFFFFF$_h$</td><td colspan="3">OPENLOOP_ACCELERATION</td><td>Unsigned</td><td></td></tr>
</table>

Address	Registername					Access
		Min	Max	Default	Unit	
		0	4294967295	0		
		Acceleration of open loop phi.				
0x21$_h$	OPENLOOP_VELOCITY_TARGET					RW
	Mask	Name			Type	
	0xFFFFFFFF$_h$	OPENLOOP_VELOCITY_TARGET			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
		Target velocity of open loop phi.				
0x22$_h$	OPENLOOP_VELOCITY_ACTUAL					RW
	Mask	Name			Type	
	0xFFFFFFFF$_h$	OPENLOOP_VELOCITY_ACTUAL			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
		Actual velocity of open loop generator.				
0x23$_h$	OPENLOOP_PHI					RWI
	Mask	Name			Type	
	0x0000FFFF$_h$	OPENLOOP_PHI			Signed	
		Min	Max	Default	Unit	
		-32768	32767	0		
		Angle phi open loop (either mapped to electrical angel phi_e or mechanical angle phi_m).				
0x24$_h$	UQ_UD_EXT					RW
	Mask	Name			Type	
	0x0000FFFF$_h$	UD_EXT			Signed	
		Min	Max	Default	Unit	
		-32768	32767	0		
		External writable parameter for open loop voltage control mode, usefull during system setup, U_D component.				
	Mask	Name			Type	
	0xFFFF0000$_h$	UQ_EXT			Signed	
		Min	Max	Default	Unit	
		-32768	32767	0		
		External writable parameter for open loop voltage control mode, usefull during system setup, U_Q component.				
0x25$_h$	ABN_DECODER_MODE					RW

Address	Registername				Access
	Mask	Name		Type	
	$0x00000001_h$	apol		Bool	
		Min	Max	Default	Unit
		0	1	0	
		Polarity of A pulse. 0: off 1: on			
	Mask	Name		Type	
	$0x00000002_h$	bpol		Bool	
		Min	Max	Default	Unit
		0	1	0	
		Polarity of B pulse. 0: off 1: on			
	Mask	Name		Type	
	$0x00000004_h$	npol		Bool	
		Min	Max	Default	Unit
		0	1	0	
		Polarity of N pulse. 0: off 1: on			
	Mask	Name		Type	
	$0x00000008_h$	use_abn_as_n		Bool	
		Min	Max	Default	Unit
		0	1	0	
		0: Ignore A and B polarity with Npulse = N, 1 : Npulse = N and A and B 0: Ignore A and B polarity with Npulse = N 1: Npulse = N and A and B			
	Mask	Name		Type	
	$0x00000100_h$	cln		Bool	
		Min	Max	Default	Unit
		0	1	0	
		Clear writes ABN_DECODER_COUNT_N into decoder count at Npulse. 0: off			

Address	Registername				Access
		1: on			
	Mask	Name		Type	
	0x00001000$_h$	direction		Bool	
		Min	Max	Default	Unit
		0	1	0	
		Decoder count direction. 0: positive 1: negative			
0x26$_h$	ABN_DECODER_PPR				RW
	Mask	Name		Type	
	0x00FFFFFF$_h$	ABN_DECODER_PPR		Unsigned	
		Min	Max	Default	Unit
		0	16777215	65536	
		Decoder pules per mechanical revolution.			
0x27$_h$	ABN_DECODER_COUNT				RW
	Mask	Name		Type	
	0x00FFFFFF$_h$	ABN_DECODER_COUNT		Unsigned	
		Min	Max	Default	Unit
		0	16777215	0	
		Raw decoder count; the digital decoder engine counts modulo (decoder_ppr).			
0x28$_h$	ABN_DECODER_COUNT_N				RW
	Mask	Name		Type	
	0x00FFFFFF$_h$	ABN_DECODER_COUNT_N		Unsigned	
		Min	Max	Default	Unit
		0	16777215	0	
		Decoder count latched on N pulse, when N pulse clears decoder_count also decoder_count_n is 0.			
0x29$_h$	ABN_DECODER_PHI_E_PHI_M_OFFSET				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	ABN_DECODER_PHI_M_OFFSET		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		ABN_DECODER_PHI_M_OFFSET to shift (rotate) angle DECODER_PHI_M.			
	Mask	Name		Type	

Address	Registername				Access
	0xFFFF0000$_h$	ABN_DECODER_PHI_E_OFFSET			Signed
		Min	Max	Default	Unit
		-32768	32767	0	
		ABN_DECODER_PHI_E_OFFSET to shift (rotate) angle DECODER_PHI_E.			
0x2A$_h$	ABN_DECODER_PHI_E_PHI_M				R
	Mask	Name			Type
	0x0000FFFF$_h$	ABN_DECODER_PHI_M			Signed
		Min	Max	Default	Unit
		-32768	32767	0	
		ABN_DECODER_PHI_M = ABN_DECODER_COUNT * 2î6 / ABN_DECODER_PPR + ABN_DECODER_PHI_M_OFFSET;			
	Mask	Name			Type
	0xFFFF0000$_h$	ABN_DECODER_PHI_E			Signed
		Min	Max	Default	Unit
		-32768	32767	0	
		ABN_DECODER_PHI_E = (ABN_DECODER_PHI_M * N_POLE_PAIRS_) + ABN_DECODER_PHI_E_OFFSET			
0x2C$_h$	ABN_2_DECODER_MODE				RW
	Mask	Name			Type
	0x00000001$_h$	apol			Bool
		Min	Max	Default	Unit
		0	1	0	
		Polarity of A pulse. 0: off 1: on			
	Mask	Name			Type
	0x00000002$_h$	bpol			Bool
		Min	Max	Default	Unit
		0	1	0	
		Polarity of B pulse. 0: off 1: on			
	Mask	Name			Type
	0x00000004$_h$	npol			Bool
		Min	Max	Default	Unit

Address	Registername				Access
		0	1	0	
		Polarity of N pulse. 0: off 1: on			
	Mask	Name			Type
	$0x00000008_h$	use_abn_as_n			Bool
		Min	Max	Default	Unit
		0	1	0	
		0: Ignore A and B polarity with Npulse = N, 1 : Npulse = N and A and B 0: Ignore A and B polarity with Npulse = N 1: Npulse = N and A and B			
	Mask	Name			Type
	$0x00000100_h$	cln			Bool
		Min	Max	Default	Unit
		0	1	0	
		Clear writes ABN_2_DECODER_COUNT_N into decoder count at Npulse. 0: off 1: on			
	Mask	Name			Type
	$0x00001000_h$	direction			Bool
		Min	Max	Default	Unit
		0	1	0	
		Decoder count direction. 0: positive 1: negative			
$0x2D_h$	ABN_2_DECODER_PPR				RW
	Mask	Name			Type
	$0x00FFFFFF_h$	ABN_2_DECODER_PPR			Unsigned
		Min	Max	Default	Unit
		1	16777215	65536	
		Decoder_2 pules per mechanical revolution. This 2nd ABN encoder interface is for positioning or velocity control but NOT for motor commutation.			
$0x2E_h$	ABN_2_DECODER_COUNT				RW

Address	Registername				Access
	Mask	Name		Type	
	0x00FFFFFF$_h$	ABN_2_DECODER_COUNT		Unsigned	
		Min	Max	Default	Unit
		0	16777215	0	
		Raw decoder_2 count; the digital decoder engine counts modulo (decoder_2_ppr).			
0x2F$_h$	ABN_2_DECODER_COUNT_N				RW
	Mask	Name		Type	
	0x00FFFFFF$_h$	ABN_2_DECODER_COUNT_N		Unsigned	
		Min	Max	Default	Unit
		0	16777215	0	
		Decoder_2 count latched on N pulse, when N pulse clears decoder_2_count also decoder_2_count_n is 0.			
0x30$_h$	ABN_2_DECODER_PHI_M_OFFSET				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	ABN_2_DECODER_PHI_M_OFFSET		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		ABN_2_DECODER_PHI_M_OFFSET to shift (rotate) angle DECODER_2_PHI_M.			
0x31$_h$	ABN_2_DECODER_PHI_M				R
	Mask	Name		Type	
	0x0000FFFF$_h$	ABN_2_DECODER_PHI_M		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		ABN_2_DECODER_PHI_M = ABN_2_DECODER_COUNT * 2î6 / ABN_2_DECODER_PPR + ABN_2_DECODER_PHI_M_OFFSET;			
0x33$_h$	HALL_MODE				RW
	Mask	Name		Type	
	0x00000001$_h$	polarity		Bool	
		Min	Max	Default	Unit
		0	1	0	
		polarity 0: off 1: on			
	Mask	Name		Type	

Address	Registername				Access
	0x00000100$_h$	interpolation		Bool	
		Min	Max	Default	Unit
		0	1	0	
		interpolation 0: off 1: on			
	Mask	Name		Type	
	0x00001000$_h$	direction		Bool	
		Min	Max	Default	Unit
		0	1	0	
		direction 0: off 1: on			
	Mask	Name		Type	
	0x0FFF0000$_h$	HALL_BLANK		Unsigned	
		Min	Max	Default	Unit
		0	4095	0	
		tBLANK = 10ns * HALL_BLANK			
0x34$_h$	HALL_POSITION_060_000				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	HALL_POSITION_000		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		s16 hall sensor position at 0°			
	Mask	Name		Type	
	0xFFFF0000$_h$	HALL_POSITION_060		Signed	
		Min	Max	Default	Unit
		-32768	32767	10922	
		s16 hall sensor position at 60°.			
0x35$_h$	HALL_POSITION_180_120				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	HALL_POSITION_120		Signed	
		Min	Max	Default	Unit
		-32768	32767	21845	
		s16 hall sensor position at 120°.			

Address	Registername				Access
	Mask	Name		Type	
	0xFFFF0000$_h$	HALL_POSITION_180		Signed	
		Min	Max	Default	Unit
		-32768	32767	-32768	
		s16 hall sensor position at 180°.			
0x36$_h$	HALL_POSITION_300_240				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	HALL_POSITION_240		Signed	
		Min	Max	Default	Unit
		-32768	32767	-21846	
		s16 hall sensor position at 240°.			
	Mask	Name		Type	
	0xFFFF0000$_h$	HALL_POSITION_300		Signed	
		Min	Max	Default	Unit
		-32768	32767	-10923	
		s16 hall sensor position at 300°.			
0x37$_h$	HALL_PHI_E_PHI_M_OFFSET				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	HALL_PHI_M_OFFSET		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		Offset of mechanical angle hall_phi_m of hall decoder.			
	Mask	Name		Type	
	0xFFFF0000$_h$	HALL_PHI_E_OFFSET		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		Offset for electrical angle hall_phi_e of hall decoder.			
0x38$_h$	HALL_DPHI_MAX				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	HALL_DPHI_MAX		Unsigned	
		Min	Max	Default	Unit
		0	65535	10922	
		Maximum dx for interpolation (default for digital hall: u16/6).			
0x39$_h$	HALL_PHI_E_INTERPOLATED_PHI_E				R
	Mask	Name		Type	

Address	Registername				Access
	0x0000FFFF$_h$	HALL_PHI_E			Signed
		Min	Max	Default	Unit
		-32768	32767	0	
		Raw electrical angle hall_phi_e of hall decoder, selection programmed via HALL_MODE control bit.			
	Mask	Name			Type
	0xFFFF0000$_h$	HALL_PHI_E_INTERPOLATED			Signed
		Min	Max	Default	Unit
		-32768	32767	0	
		Interpolated electrical angle hall_phi_e_interpolated, selection programmed via HALL_MODE control bit.			
0x3A$_h$	HALL_PHI_M				R
	Mask	Name			Type
	0x0000FFFF$_h$	HALL_PHI_M			Signed
		Min	Max	Default	Unit
		-32768	32767	0	
		Mechanical angle hall_phi_m of hall decoder.			
0x3B$_h$	AENC_DECODER_MODE				RW
	Mask	Name			Type
	0x00000001$_h$	AENC_DECODER_MODE[0]			Bool
		Min	Max	Default	Unit
		0	1	0	
		nXY_UVW : 0: SinCos Mode // 1: 0° 120° 240° Mode 0: off 1: on			
	Mask	Name			Type
	0x00001000$_h$	AENC_DECODER_MODE[12]			Bool
		Min	Max	Default	Unit
		0	1	0	
		decoder count direction 0: positive 1: negative			
0x3C$_h$	AENC_DECODER_N_THRESHOLD				RW
	Mask	Name			Type
	0x0000FFFF$_h$	AENC_DECODER_N_THRESHOLD			Unsigned

Address	Registername				Access
		Min	Max	Default	Unit
		0	65535	0	
		Threshold for generating of N pulse from analog AENC_N signal (only needed for analog SinCos encoders with analog N signal).			
	Mask	Name			Type
	0xFFFF0000$_h$	AENC_DECODER_N_MASK			Signed
		Min	Max	Default	Unit
		-32768	32767	0	
		Optional position mask (position) for the analog N pulse within phi_a period to be and-ed with the digital N pulse generated via aenc_decoder_n_threshold.			
0x3D$_h$	AENC_DECODER_PHI_A_RAW				R
	Mask	Name			Type
	0x0000FFFF$_h$	AENC_DECODER_PHI_A_RAW			Signed
		Min	Max	Default	Unit
		-32768	32767	0	
		Raw analog angle phi calculated from analog AENC inputs (analog hall, analog SinCos, ...).			
0x3E$_h$	AENC_DECODER_PHI_A_OFFSET				RW
	Mask	Name			Type
	0x0000FFFF$_h$	AENC_DECODER_PHI_A_OFFSET			Signed
		Min	Max	Default	Unit
		-32768	32767	0	
		Offset for angle phi from analog decoder (analog hall, analog SinCos, ...).			
0x3F$_h$	AENC_DECODER_PHI_A				R
	Mask	Name			Type
	0x0000FFFF$_h$	AENC_DECODER_PHI_A			Signed
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		Resulting phi available for the FOC (phi_e might need to be calculated from this angle via aenc_decoder_ppr, for analog hall sensors phi_a might be used directly as phi_e depends on analog hall signal type).			
0x40$_h$	AENC_DECODER_PPR				RW
	Mask	Name			Type
	0x0000FFFF$_h$	AENC_DECODER_PPR			Signed

Address	Registername					Access
		Min	Max	Default	Unit	
		-32768	32767	1		
		Number of periods per revolution also called lines per revolution (different nomenclatur compared to digital ABN encoders).				
$0x41_h$	AENC_DECODER_COUNT					R
	Mask	Name			Type	
	$0xFFFFFFFF_h$	AENC_DECODER_COUNT			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
		Decoder position, raw unscaled.				
$0x42_h$	AENC_DECODER_COUNT_N					RW
	Mask	Name			Type	
	$0xFFFFFFFF_h$	AENC_DECODER_COUNT_N			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
		Latched decoder position on analog N pulse event.				
$0x45_h$	AENC_DECODER_PHI_E_PHI_M_OFFSET					RW
	Mask	Name			Type	
	$0x0000FFFF_h$	AENC_DECODER_PHI_M_OFFSET			Signed	
		Min	Max	Default	Unit	
		-32768	32767	0		
		Offset for mechanical angle phi_m.				
	Mask	Name			Type	
	$0xFFFF0000_h$	AENC_DECODER_PHI_E_OFFSET			Signed	
		Min	Max	Default	Unit	
		-32768	32767	0		
		Offset for electrical angle phi_e.				
$0x46_h$	AENC_DECODER_PHI_E_PHI_M					R
	Mask	Name			Type	
	$0x0000FFFF_h$	AENC_DECODER_PHI_M			Signed	
		Min	Max	Default	Unit	
		-32768	32767	0		
		Resulting angle phi_m.				
	Mask	Name			Type	
	$0xFFFF0000_h$	AENC_DECODER_PHI_E			Signed	

Address	Registername					Access
		Min	Max	Default	Unit	
		-32768	32767	0		
		Resulting angle phi_e.				
0x47$_h$	AENC_DECODER_POSITION					R
	Mask	Name			Type	
	0xFFFFFFFF$_h$	AENC_DECODER_POSITION			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
		Multi-turn position.				
0x4B$_h$	PIDIN_VELOCITY_TARGET					R
	Mask	Name			Type	
	0xFFFFFFFF$_h$	PIDIN_VELOCITY_TARGET			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
		Target velocity at PI controller input.				
0x4C$_h$	PIDIN_POSITION_TARGET					R
	Mask	Name			Type	
	0xFFFFFFFF$_h$	PIDIN_POSITION_TARGET			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
		Target position at PI controller input.				
0x4D$_h$	CONFIG_DATA					RW
	Variant 1					
	Mask	Name			Type	
	0xFFFFFFFF$_h$	biquad_x_a_1			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
	Variant 2					
	Mask	Name			Type	
	0xFFFFFFFF$_h$	biquad_x_a_2			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
	Variant 4					
	Mask	Name			Type	
	0xFFFFFFFF$_h$	biquad_x_b_0			Signed	

Address	Registername				Access
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	Variant 5				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_x_b_1		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	Variant 6				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_x_b_2		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	Variant 7				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_x_enable		Bool	
	Min	Max	Default	Unit	
	0 0: off 1: on	1	0		
	Variant 9				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_v_a_1		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	Variant 10				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_v_a_2		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	Variant 12				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_v_b_0		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	Variant 13				

Address	Registername				Access
	Mask	Name		Type	
	$0xFFFFFFFF_h$	biquad_v_b_1		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 14				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	biquad_v_b_2		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 15				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	biquad_v_enable		Bool	
		Min	Max	Default	Unit
		0 0: off 1: on	1	0	
	Variant 17				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	biquad_t_a_1		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 18				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	biquad_t_a_2		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 20				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	biquad_t_b_0		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 21				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	biquad_t_b_1		Signed	
		Min	Max	Default	Unit

Address	Registername				Access
		-2147483648	2147483647	0	
	Variant 22				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_t_b_2		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 23				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_t_enable		Bool	
		Min	Max	Default	Unit
		0 0: off 1: on	1	0	
	Variant 25				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_f_a_1		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 26				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_f_a_2		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 28				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_f_b_0		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 29				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_f_b_1		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 30				
	Mask	Name		Type	

Address	Registername				Access
	0xFFFFFFFF$_h$	biquad_f_b_2		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 31				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	biquad_f_enable		Bool	
		Min	Max	Default	Unit
		0 0: off 1: on	1	0	
	Variant 32				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	prbs_amplitude		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 33				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	prbs_down_sampling_ratio		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 40				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	feed_forward_velocity_gain		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 41				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	feed_forward_velocity_filter_constant		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 42				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	feed_forward_torque_gain		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	

Address	Registername				Access
	Variant 43				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	feed_forward_torgue_filter_constant		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 50				
	Mask	Name		Type	
	$0x0000FFFF_h$	VELOCITY_METER_PPTM_MIN_POS_DEV		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Variant 51				
	Mask	Name		Type	
	$0x0000FFFF_h$	ref_switch_config		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Variant 52				
	Mask	Name		Type	
	$0x00000001_h$	Encoder_Init_hall_Enable		Bool	
		Min	Max	Default	Unit
		0	1	0	
		0: off 1: on			
	Variant 60				
	Mask	Name		Type	
	$0x000000FF_h$	SINGLE_PIN_IF_CFG		Unsigned	
		Min	Max	Default	Unit
		0	255	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	SINGLE_PIN_IF_STATUS		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Variant 61				
	Mask	Name		Type	
	$0x0000FFFF_h$	SINGLE_PIN_IF_OFFSET		Unsigned	
		Min	Max	Default	Unit

Address	Registername				Access
		0	65535	0	
		Offset for scaling of Single pin Interface input			
	Mask	Name			Type
	0xFFFF0000$_h$	SINGLE_PIN_IF_SCALE			Signed
		Min	Max	Default	Unit
		-32767	32767	0	
		Gain factor of Single pin Interface input			
0x4E$_h$	CONFIG_ADDR				RW
	Mask	Name			Type
	0xFFFFFFFF$_h$	CONFIG_ADDR			Choice
		Min	Max	Default	Unit
		1	52	0	
		1: biquad_x_a_1 2: biquad_x_a_2 4: biquad_x_b_0 5: biquad_x_b_1 6: biquad_x_b_2 7: biquad_x_enable 9: biquad_v_a_1 10: biquad_v_a_2 12: biquad_v_b_0 13: biquad_v_b_1 14: biquad_v_b_2 15: biquad_v_enable 17: biquad_t_a_1 18: biquad_t_a_2 20: biquad_t_b_0 21: biquad_t_b_1 22: biquad_t_b_2 23: biquad_t_enable 25: biquad_f_a_1 26: biquad_f_a_2 28: biquad_f_b_0 29: biquad_f_b_1 30: biquad_f_b_2			

<table>
<tr><th>Address</th><th colspan="5">Registername</th><th>Access</th></tr>
<tr><td></td><td colspan="5">31: biquad_f_enable
32: prbs_amplitude
33: prbs_down_sampling_ratio
40: feed_forward_velocity_gain
41: feed_forward_velicity_filter_constant
42: feed_forward_torque_gain
43: feed_forward_torgue_filter_constant
50: VELOCITY_METER_PPTM_MIN_POS_DEV
51: ref_switch_config
52: Encoder_Init_hall_Enable
60: SINGLE_PIN_IF_STATUS_CFG
61: SINGLE_PIN_IF_SCALE_OFFSET</td><td></td></tr>
<tr><td>0x50$_h$</td><td colspan="5">VELOCITY_SELECTION</td><td>RW</td></tr>
<tr><td></td><td>Mask</td><td colspan="3">Name</td><td>Type</td><td></td></tr>
<tr><td></td><td>0x000000FF$_h$</td><td colspan="3">VELOCITY_SELECTION</td><td>Choice</td><td></td></tr>
<tr><td></td><td></td><td>Min</td><td>Max</td><td>Default</td><td>Unit</td><td></td></tr>
<tr><td></td><td></td><td>0</td><td>12</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td colspan="5">Selects the source of the velocity source for velocity measurement.
0: phi_e selected via PHI_E_SELECTION
1: phi_e_ext
2: phi_e_openloop
3: phi_e_abn
4: reserved
5: phi_e_hal
6: phi_e_aenc
7: phi_a_aenc
8: reserved
9: phi_m_abn
10: phi_m_abn_2
11: phi_m_aenc
12: phi_m_hal</td><td></td></tr>
<tr><td></td><td>Mask</td><td colspan="3">Name</td><td>Type</td><td></td></tr>
<tr><td></td><td>0x0000FF00$_h$</td><td colspan="3">VELOCITY_METER_SELECTION</td><td>Choice</td><td></td></tr>
<tr><td></td><td></td><td>Min</td><td>Max</td><td>Default</td><td>Unit</td><td></td></tr>
</table>

Address	Registername				Access
		0	1	0	
		0: default velocity meter (fixed frequency sampling) 1: advanced velocity meter (time difference measurement)			
$0x51_h$	POSITION_SELECTION				RW
	Mask	Name		Type	
	$0x000000FF_h$	POSITION_SELECTION		Choice	
		Min	Max	Default	Unit
		0	12	0	
		0: phi_e selected via PHI_E_SELECTION 1: phi_e_ext 2: phi_e_openloop 3: phi_e_abn 4: reserved 5: phi_e_hal 6: phi_e_aenc 7: phi_a_aenc 8: reserved 9: phi_m_abn 10: phi_m_abn_2 11: phi_m_aenc 12: phi_m_hal			
$0x52_h$	PHI_E_SELECTION				RW
	Mask	Name		Type	
	$0x000000FF_h$	PHI_E_SELECTION		Choice	
		Min	Max	Default	Unit
		0	7	0	
		0: reserved 1: phi_e_ext 2: phi_e_openloop 3: phi_e_abn 4: reserved 5: phi_e_hal 6: phi_e_aenc 7: phi_a_aenc			
$0x53_h$	PHI_E				R

Address	Registername				Access
	Mask	Name		Type	
	0x0000FFFF$_h$	PHI_E		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
		Angle used for the inner FOC loop.			
0x54$_h$	PID_FLUX_P_FLUX_I				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	PID_FLUX_I		Signed	
		Min	Max	Default	Unit
		0	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	PID_FLUX_P		Signed	
		Min	Max	Default	Unit
		0	32767	0	
0x56$_h$	PID_TORQUE_P_TORQUE_I				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	PID_TORQUE_I		Signed	
		Min	Max	Default	Unit
		0	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	PID_TORQUE_P		Signed	
		Min	Max	Default	Unit
		0	32767	0	
0x58$_h$	PID_VELOCITY_P_VELOCITY_I				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	PID_VELOCITY_I		Signed	
		Min	Max	Default	Unit
		0	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	PID_VELOCITY_P		Signed	
		Min	Max	Default	Unit
		0	32767	0	
0x5A$_h$	PID_POSITION_P_POSITION_I				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	PID_POSITION_I		Signed	

Address	Registername					Access
		Min	Max	Default	Unit	
		0	32767	0		
	Mask	Name			Type	
	0xFFFF0000$_h$	PID_POSITION_P			Signed	
		Min	Max	Default	Unit	
		0	32767	0		
0x5C$_h$	PID_TORQUE_FLUX_TARGET_DDT_LIMITS					RW
	Mask	Name			Type	
	0xFFFFFFFF$_h$	PID_TORQUE_FLUX_TARGET_DDT_LIMITS			Unsigned	
		Min	Max	Default	Unit	
		0	32767	32767	[1/us]	
		Limits of change in time [d/dt] of the target torque and target flux.				
0x5D$_h$	PIDOUT_UQ_UD_LIMITS					RW
	Mask	Name			Type	
	0x0000FFFF$_h$	PIDOUT_UQ_UD_LIMITS			Unsigned	
		Min	Max	Default	Unit	
		0	32767	23169		
		Two dimensional circular limiter for inputs of iPark.				
0x5E$_h$	PID_TORQUE_FLUX_LIMITS					RW
	Mask	Name			Type	
	0x0000FFFF$_h$	PID_TORQUE_FLUX_LIMITS			Unsigned	
		Min	Max	Default	Unit	
		0	32767	32767		
		PID torque limt and PID flux limit, limits the target values coming from the target registers.				
0x5F$_h$	PID_ACCELERATION_LIMIT					RW
	Mask	Name			Type	
	0xFFFFFFFF$_h$	PID_ACCELERATION_LIMIT			Unsigned	
		Min	Max	Default	Unit	
		0	4294967295	2147483647		
		Acceleration limit.				
0x60$_h$	PID_VELOCITY_LIMIT					RW
	Mask	Name			Type	
	0xFFFFFFFF$_h$	PID_VELOCITY_LIMIT			Unsigned	

Address	Registername					Access
		Min	Max	Default	Unit	
		0	4294967295	2147483647		
		Velocity limit.				
0x61$_h$	PID_POSITION_LIMIT_LOW					RW
	Mask	Name			Type	
	0xFFFFFFFF$_h$	PID_POSITION_LIMIT_LOW			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	-2147483647		
		Position limit low, programmable positon barrier.				
0x62$_h$	PID_POSITION_LIMIT_HIGH					RW
	Mask	Name			Type	
	0xFFFFFFFF$_h$	PID_POSITION_LIMIT_HIGH			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	2147483647		
		Position limit high, programmable positon barrier.				
0x63$_h$	MODE_RAMP_MODE_MOTION					RW
	Mask	Name			Type	
	0x000000FF$_h$	MODE_MOTION			Choice	
		Min	Max	Default	Unit	
		0	15	0		
		0: stopped_mode 1: torque_mode 2: velocity_mode 3: position_mode 4: prbs_flux_mode 5: prbs_torque_mode 6: prbs_velocity_mode 7: prbs_position_mode 8: uq_ud_ext 9: enc_init_mini_move 10: AGPI_A torque_mode 11: AGPI_A velocity_mode 12: AGPI_A position_mode 13: PWM_I torque_mode 14: PWM_I velocity_mode				

Address	Registername				Access
		15: PWM_I position_mode			
	Mask	Name		Type	
	0x0000FF00$_h$	MODE_RAMP		Choice	
		Min	Max	Default	Unit
		0	7	0	
		0: no velocity ramping 1: reserved 2: reserved 3: reserved 4: reserved 5: reserved 6: reserved 7: reserved			
	Mask	Name		Type	
	0x00FF0000$_h$	MODE_FF		Choice	
		Min	Max	Default	Unit
		0	2	0	
		0: disabled 1: feed forward velocity control 2: feed forward torque control			
	Mask	Name		Type	
	0x7F000000$_h$	MODE_PID_SMPL		Unsigned	
		Min	Max	Default	Unit
		0	127	0	
		Controller downsampling factor for advanced velocity and position controller			
	Mask	Name		Type	
	0x80000000$_h$	MODE_PID_TYPE		Choice	
		Min	Max	Default	Unit
		0	1	0	
		0: Classic PI architecture 1: Advanced PI architecture			
0x64$_h$	PID_TORQUE_FLUX_TARGET				RW
	Mask	Name		Type	
	0x0000FFFF$_h$	PID_FLUX_TARGET		Signed	

Address	Registername					Access
		Min	Max	Default	Unit	
		-32768	32767	0		
	Mask	Name			Type	
	0xFFFF0000$_h$	PID_TORQUE_TARGET			Signed	
		Min	Max	Default	Unit	
		-32768	32767	0		
0x65$_h$	PID_TORQUE_FLUX_OFFSET					RW
	Mask	Name			Type	
	0x0000FFFF$_h$	PID_FLUX_OFFSET			Signed	
		Min	Max	Default	Unit	
		-32768	32767	0		
		Flux offset for feed forward control.				
	Mask	Name			Type	
	0xFFFF0000$_h$	PID_TORQUE_OFFSET			Signed	
		Min	Max	Default	Unit	
		-32768	32767	0		
		Torque offset for feed forward control.				
0x66$_h$	PID_VELOCITY_TARGET					RW
	Mask	Name			Type	
	0xFFFFFFFF$_h$	PID_VELOCITY_TARGET			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
		Target velocity register (for velocity mode).				
0x67$_h$	PID_VELOCITY_OFFSET					RW
	Mask	Name			Type	
	0xFFFFFFFF$_h$	PID_VELOCITY_OFFSET			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		
		Velocity offset for feed forward control.				
0x68$_h$	PID_POSITION_TARGET					RW
	Mask	Name			Type	
	0xFFFFFFFF$_h$	PID_POSITION_TARGET			Signed	
		Min	Max	Default	Unit	
		-2147483648	2147483647	0		

Address	Registername				Access
		Target position register (for position mode).			
$0x69_h$	PID_TORQUE_FLUX_ACTUAL				R
	Mask	Name		Type	
	$0x0000FFFF_h$	PID_FLUX_ACTUAL		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	PID_TORQUE_ACTUAL		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
$0x6A_h$	PID_VELOCITY_ACTUAL				R
	Mask	Name		Type	
	$0xFFFFFFFF_h$	PID_VELOCITY_ACTUAL		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		Actual velocity.			
$0x6B_h$	PID_POSITION_ACTUAL				RW
	Mask	Name		Type	
	$0xFFFFFFFF_h$	PID_POSITION_ACTUAL		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		Actual multi turn position for positioning. WRITE on PID_POSITION_ACTUAL writes same value into PID_POSITION_TARGET to avoid unwanted move.			
$0x6C_h$	PID_ERROR_DATA				R
	Variant 0				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	PID_TORQUE_ERROR		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		PID torque error.			
	Variant 1				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	PID_FLUX_ERROR		Signed	
		Min	Max	Default	Unit

Address	Registername				Access
	-2147483648	2147483647	0		
	PID flux error.				
	Variant 2				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PID_VELOCITY_ERROR		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	PID velocity error.				
	Variant 3				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PID_POSITION_ERROR		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	PID position error.				
	Variant 4				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PID_TORQUE_ERROR_SUM		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	PID torque error.				
	Variant 5				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PID_FLUX_ERROR_SUM		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	PID flux error sum.				
	Variant 6				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PID_VELOCITY_ERROR_SUM		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	PID velocity error sum.				
	Variant 7				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PID_POSITION_ERROR_SUM		Signed	

Address	Registername				Access
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		PID position error sum.			
0x6D$_h$	PID_ERROR_ADDR				RW
	Mask	Name			Type
	0x000000FF$_h$	PID_ERROR_ADDR			Choice
		Min	Max	Default	Unit
		0	7	0	
		0: PID_TORQUE_ERROR 1: PID_FLUX_ERROR 2: PID_VELOCITY_ERROR 3: PID_POSITION_ERROR 4: PID_TORQUE_ERROR_SUM 5: PID_FLUX_ERROR_SUM 6: PID_VELOCITY_ERROR_SUM 7: PID_POSITION_ERROR_SUM			
0x6E$_h$	INTERIM_DATA				RW
	Variant 0				
	Mask	Name			Type
	0xFFFFFFFF$_h$	PIDIN_TARGET_TORQUE			Signed
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		PIDIN target torque.			
	Variant 1				
	Mask	Name			Type
	0xFFFFFFFF$_h$	PIDIN_TARGET_FLUX			Signed
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		PIDIN target flux.			
	Variant 2				
	Mask	Name			Type
	0xFFFFFFFF$_h$	PIDIN_TARGET_VELOCITY			Signed
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		PIDIN target velocity.			

Address	Registername				Access
	Variant 3				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PIDIN_TARGET_POSITION		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		PIDIN target position.			
	Variant 4				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PIDOUT_TARGET_TORQUE		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		PIDOUT target torque.			
	Variant 5				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PIDOUT_TARGET_FLUX		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		PIDOUT target flux.			
	Variant 6				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PIDOUT_TARGET_VELOCITY		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		PIDOUT target velocity.			
	Variant 7				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	PIDOUT_TARGET_POSITION		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		PIDOUT target position.			
	Variant 8				
	Mask	Name		Type	
	0x0000FFFF$_h$	FOC_IUX		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	

Address	Registername				Access
	Mask	Name		Type	
	$0xFFFF0000_h$	FOC_IWY		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 9				
	Mask	Name		Type	
	$0x0000FFFF_h$	FOC_IV		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 10				
	Mask	Name		Type	
	$0x0000FFFF_h$	FOC_IA		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	FOC_IB		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 11				
	Mask	Name		Type	
	$0x0000FFFF_h$	FOC_ID		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	FOC_IQ		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 12				
	Mask	Name		Type	
	$0x0000FFFF_h$	FOC_UD		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	FOC_UQ		Signed	
		Min	Max	Default	Unit

Address	Registername				Access
		-32768	32767	0	
	Variant 13				
	Mask	Name		Type	
	0x0000FFFF$_h$	FOC_UD_LIMITED		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	FOC_UQ_LIMITED		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 14				
	Mask	Name		Type	
	0x0000FFFF$_h$	FOC_UA		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	FOC_UB		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 15				
	Mask	Name		Type	
	0x0000FFFF$_h$	FOC_UUX		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	FOC_UWY		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 16				
	Mask	Name		Type	
	0x0000FFFF$_h$	FOC_UV		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 17				
	Mask	Name		Type	

Address	Registername				Access
	0x0000FFFF$_h$	PWM_UX		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	PWM_WY		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 18				
	Mask	Name		Type	
	0x0000FFFF$_h$	PWM_V		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 19				
	Mask	Name		Type	
	0x0000FFFF$_h$	ADC_I_0		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	ADC_I_1		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 20				
	Mask	Name		Type	
	0x000000FF$_h$	PID_FLUX_ACTUAL_DIV256		Signed	
		Min	Max	Default	Unit
		-128	127	0	
	Mask	Name		Type	
	0x0000FF00$_h$	PID_TORQUE_ACTUAL_DIV256		Signed	
		Min	Max	Default	Unit
		-128	127	0	
	Mask	Name		Type	
	0x00FF0000$_h$	PID_FLUX_TARGET_DIV256		Signed	
		Min	Max	Default	Unit
		-128	127	0	
	Mask	Name		Type	

Address	Registername				Access
	$0xFF000000_h$	PID_TORQUE_TARGET_DIV256		Signed	
		Min	Max	Default	Unit
		-128	127	0	
	Variant 21				
	Mask	Name		Type	
	$0x0000FFFF_h$	PID_TORQUE_ACTUAL		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	PID_TORQUE_TARGET		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 22				
	Mask	Name		Type	
	$0x0000FFFF_h$	PID_FLUX_ACTUAL		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	PID_FLUX_TARGET		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 23				
	Mask	Name		Type	
	$0x0000FFFF_h$	PID_VELOCITY_ACTUAL_DIV256		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	PID_VELOCITY_TARGET_DIV256		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 24				
	Mask	Name		Type	
	$0x0000FFFF_h$	PID_VELOCITY_ACTUAL_LSB		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	

Address	Registername				Access
	Mask	Name		Type	
	$0xFFFF0000_h$	PID_VELOCITY_TARGET_LSB		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 25				
	Mask	Name		Type	
	$0x0000FFFF_h$	PID_POSITION_ACTUAL_DIV256		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	PID_POSITION_TARGET_DIV256		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 26				
	Mask	Name		Type	
	$0x0000FFFF_h$	PID_POSITION_ACTUAL_LSB		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	PID_POSITION_TARGET_LSB		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 27				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	FF_VELOCITY		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 28				
	Mask	Name		Type	
	$0x0000FFFF_h$	FF_TORQUE		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 29				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	ACTUAL_VELOCITY_PPTM		Signed	

Address	Registername				Access
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 30				
	Mask	Name			Type
	0x0000FFFF$_h$	REF_SWITCH_STATUS			Unsigned
		Min	Max	Default	Unit
		0	65535	0	
	Variant 31				
	Mask	Name			Type
	0xFFFFFFFF$_h$	HOME_POSITION			Signed
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 32				
	Mask	Name			Type
	0xFFFFFFFF$_h$	LEFT_POSITION			Signed
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 33				
	Mask	Name			Type
	0xFFFFFFFF$_h$	RIGHT_POSITION			Signed
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 34				
	Mask	Name			Type
	0x0000FFFF$_h$	ENC_INIT_HALL_STATUS			Unsigned
		Min	Max	Default	Unit
		0	65535	0	
	Variant 35				
	Mask	Name			Type
	0x0000FFFF$_h$	ENC_INIT_HALL_PHI_E_ABN_OFFSET			Unsigned
		Min	Max	Default	Unit
		0	65535	0	
	Variant 36				
	Mask	Name			Type
	0x0000FFFF$_h$	ENC_INIT_HALL_PHI_E_AENC_OFFSET			Unsigned

Address	Registername				Access
	Min	Max	Default	Unit	
	0	65535	0		
	Variant 37				
	Mask	Name		Type	
	0x0000FFFF$_h$	ENC_INIT_HALL_PHI_A_AENC_OFFSET		Unsigned	
	Min	Max	Default	Unit	
	0	65535	0		
	Variant 40				
	Mask	Name		Type	
	0x0000FFFF$_h$	ENC_INIT_MINI_MOVE_STATUS		Unsigned	
	Min	Max	Default	Unit	
	0	65535	0		
	Mask	Name		Type	
	0xFFFF0000$_h$	ENC_INIT_MINI_MOVE_U_D		Signed	
	Min	Max	Default	Unit	
	-32768	32767	0		
	Variant 41				
	Mask	Name		Type	
	0x0000FFFF$_h$	ENC_INIT_MINI_MOVE_PHI_E_OFFSET		Unsigned	
	Min	Max	Default	Unit	
	0	65535	0		
	Mask	Name		Type	
	0xFFFF0000$_h$	ENC_INIT_MINI_MOVE_PHI_E		Unsigned	
	Min	Max	Default	Unit	
	0	65535	0		
	Variant 42				
	Mask	Name		Type	
	0x0000FFFF$_h$	SINGLE_PIN_IF_TARGET_TORQUE		Signed	
	Min	Max	Default	Unit	
	-32768	32767	0		
	Mask	Name		Type	
	0xFFFF0000$_h$	SINGLE_PIN_IF_RAW_VALUE		Unsigned	
	Min	Max	Default	Unit	
	-32768	32767	0		
	Variant 43				

Address	Registername				Access
	Mask	Name		Type	
	$0xFFFFFFFF_h$	SINGLE_PIN_IF_TARGET_VELOCITY		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 44				
	Mask	Name		Type	
	$0xFFFFFFFF_h$	SINGLE_PIN_IF_TARGET_POSITION		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
	Variant 192				
	Mask	Name		Type	
	$0x0000FFFF_h$	DEBUG_VALUE_0		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	DEBUG_VALUE_1		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 193				
	Mask	Name		Type	
	$0x0000FFFF_h$	DEBUG_VALUE_2		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	DEBUG_VALUE_3		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 194				
	Mask	Name		Type	
	$0x0000FFFF_h$	DEBUG_VALUE_4		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	$0xFFFF0000_h$	DEBUG_VALUE_5		Signed	
		Min	Max	Default	Unit

Address	Registername				Access
		-32768	32767	0	
	Variant 195				
	Mask	Name		Type	
	0x0000FFFF$_h$	DEBUG_VALUE_6		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	DEBUG_VALUE_7		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 196				
	Mask	Name		Type	
	0x0000FFFF$_h$	DEBUG_VALUE_8		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	DEBUG_VALUE_9		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Variant 197				
	Mask	Name		Type	
	0x0000FFFF$_h$	DEBUG_VALUE_10		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	DEBUG_VALUE_11		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Variant 198				
	Mask	Name		Type	
	0x0000FFFF$_h$	DEBUG_VALUE_12		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	DEBUG_VALUE_13		Unsigned	

Address	Registername				Access
	Min	Max	Default	Unit	
	0	65535	0		
	Variant 199				
	Mask	Name		Type	
	0x0000FFFF$_h$	DEBUG_VALUE_14		Unsigned	
	Min	Max	Default	Unit	
	0	65535	0		
	Mask	Name		Type	
	0xFFFF0000$_h$	DEBUG_VALUE_15		Unsigned	
	Min	Max	Default	Unit	
	0	65535	0		
	Variant 200				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	DEBUG_VALUE_16		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	Variant 201				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	DEBUG_VALUE_17		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	Variant 202				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	DEBUG_VALUE_18		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	Variant 203				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	DEBUG_VALUE_19		Signed	
	Min	Max	Default	Unit	
	-2147483648	2147483647	0		
	Variant 208				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	CONFIG_REG_0		Unsigned	
	Min	Max	Default	Unit	

Address	Registername				Access
		0	4294967295	0	
	Variant 209				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	CONFIG_REG_1		Unsigned	
		Min	Max	Default	Unit
		0	4294967295	0	
	Variant 210				
	Mask	Name		Type	
	0x0000FFFF$_h$	CTRL_PARAM_0		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	CTRL_PARAM_1		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 211				
	Mask	Name		Type	
	0x0000FFFF$_h$	CTRL_PARAM_2		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	CTRL_PARAM_3		Signed	
		Min	Max	Default	Unit
		-32768	32767	0	
	Variant 212				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	STATUS_REG_0		Unsigned	
		Min	Max	Default	Unit
		0	4294967295	0	
	Variant 213				
	Mask	Name		Type	
	0xFFFFFFFF$_h$	STATUS_REG_1		Unsigned	
		Min	Max	Default	Unit
		0	4294967295	0	
	Variant 214				

Address	Registername				Access
	Mask	Name		Type	
	0x0000FFFF$_h$	STATUS_PARAM_0		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	STATUS_PARAM_1		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Variant 215				
	Mask	Name		Type	
	0x0000FFFF$_h$	STATUS_PARAM_2		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
	Mask	Name		Type	
	0xFFFF0000$_h$	STATUS_PARAM_3		Unsigned	
		Min	Max	Default	Unit
		0	65535	0	
0x6F$_h$	INTERIM_ADDR				RW
	Mask	Name		Type	
	0x000000FF$_h$	INTERIM_ADDR		Choice	
		Min	Max	Default	Unit
		0	215	0	
		0: PIDIN_TARGET_TORQUE 1: PIDIN_TARGET_FLUX 2: PIDIN_TARGET_VELOCITY 3: PIDIN_TARGET_POSITION 4: PIDOUT_TARGET_TORQUE 5: PIDOUT_TARGET_FLUX 6: PIDOUT_TARGET_VELOCITY 7: PIDOUT_TARGET_POSITION 8: FOC_IWY_IUX 9: FOC_IV 10: FOC_IB_IA 11: FOC_IQ_ID 12: FOC_UQ_UD			

Address	Registername	Access
	13: FOC_UQ_UD_LIMITED	
	14: FOC_UB_UA	
	15: FOC_UWY_UUX	
	16: FOC_UV	
	17: PWM_WY_UX	
	18: PWM_UV	
	19: ADC_I1_I0	
	20: PID_TORQUE_TARGET_FLUX_TARGET_TORQUE_ACTUAL_FLUX_ACTUAL_DIV256	
	21: PID_TORQUE_TARGET_TORQUE_ACTUAL	
	22: PID_FLUX_TARGET_FLUX_ACTUAL	
	23: PID_VELOCITY_TARGET_VELOCITY_ACTUAL_DIV256	
	24: PID_VELOCITY_TARGET_VELOCITY_ACTUAL	
	25: PID_POSITION_TARGET_POSITION_ACTUAL_DIV256	
	26: PID_POSITION_TARGET_POSITION_ACTUAL	
	27: FF_VELOCITY	
	28: FF_TORQUE	
	29: ACTUAL_VELOCITY_PPTM	
	30: REF_SWITCH_STATUS	
	31: HOME_POSITION	
	32: LEFT_POSITION	
	33: RIGHT_POSITION	
	34: ENC_INIT_HALL_STATUS	
	35: ENC_INIT_HALL_PHI_E_ABN_OFFSET	
	36: ENC_INIT_HALL_PHI_E_AENC_OFFSET	
	37: ENC_INIT_HALL_PHI_A_AENC_OFFSET	
	40: enc_init_mini_move_u_d_status	
	41: enc_init_mini_move_phi_e_phi_e_offset	
	42: SINGLE_PIN_IF_RAW_VALUE_TARGET_TORQUE	
	43: SINGLE_PIN_IF_TARGET_VELOCITY	
	44: SINGLE_PIN_IF_TARGET_POSITION	
	192: DEBUG_VALUE_1_0	
	193: DEBUG_VALUE_3_2	
	194: DEBUG_VALUE_5_4	
	195: DEBUG_VALUE_7_6	
	196: DEBUG_VALUE_9_8	

<table>
<tr><th>Address</th><th colspan="5">Registername</th><th>Access</th></tr>
<tr><td></td><td colspan="5">197: DEBUG_VALUE_11_10
198: DEBUG_VALUE_13_12
199: DEBUG_VALUE_15_14
200: DEBUG_VALUE_16
201: DEBUG_VALUE_17
202: DEBUG_VALUE_18
203: DEBUG_VALUE_19
208: CONFIG_REG_0
209: CONFIG_REG_1
210: CTRL_PARAM_10
211: CTRL_PARAM_32
212: STATUS_REG_0
213: STATUS_REG_1
214: STATUS_PARAM_10
215: STATUS_PARAM_32</td><td></td></tr>
<tr><td>0x74_h</td><td colspan="5">WATCHDOG_CFG</td><td>RW</td></tr>
<tr><td></td><td>Mask</td><td colspan="3">Name</td><td>Type</td><td></td></tr>
<tr><td></td><td>0x00000003_h</td><td colspan="3">WATCHDOG_CFG</td><td>Choice</td><td></td></tr>
<tr><td></td><td></td><td>Min</td><td>Max</td><td>Default</td><td>Unit</td><td></td></tr>
<tr><td></td><td></td><td>0</td><td>3</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td></td><td colspan="4">0: No action on watchdog error
1: PWM and power stage disable on watchdog error
2: Global reset on watchdog error
3: reserved</td><td></td></tr>
<tr><td>0x75_h</td><td colspan="5">ADC_VM_LIMITS</td><td>RW</td></tr>
<tr><td></td><td>Mask</td><td colspan="3">Name</td><td>Type</td><td></td></tr>
<tr><td></td><td>0x0000FFFF_h</td><td colspan="3">ADC_VM_LIMIT_LOW</td><td>Unsigned</td><td></td></tr>
<tr><td></td><td></td><td>Min</td><td>Max</td><td>Default</td><td>Unit</td><td></td></tr>
<tr><td></td><td></td><td>0</td><td>65535</td><td>65535</td><td></td><td></td></tr>
<tr><td></td><td></td><td colspan="4">Low limit for brake chopper output BRAKE_OUT.</td><td></td></tr>
<tr><td></td><td>Mask</td><td colspan="3">Name</td><td>Type</td><td></td></tr>
<tr><td></td><td>0xFFFF0000_h</td><td colspan="3">ADC_VM_LIMIT_HIGH</td><td>Unsigned</td><td></td></tr>
<tr><td></td><td></td><td>Min</td><td>Max</td><td>Default</td><td>Unit</td><td></td></tr>
<tr><td></td><td></td><td>0</td><td>65535</td><td>65535</td><td></td><td></td></tr>
<tr><td></td><td></td><td colspan="4">High limit for brake chopper output BRAKE_OUT.</td><td></td></tr>
</table>

Address	Registername				Access
0x76$_h$	TMC4671_INPUTS_RAW				R
	Mask	Name		Type	
	0x00000001$_h$	A of ABN_RAW		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	A of ABN_RAW 0: off 1: on				
	Mask	Name		Type	
	0x00000002$_h$	B of ABN_RAW		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	B of ABN_RAW 0: off 1: on				
	Mask	Name		Type	
	0x00000004$_h$	N of ABN_RAW		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	N of ABN_RAW 0: off 1: on				
	Mask	Name		Type	
	0x00000008$_h$	-		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	— 0: off 1: on				
	Mask	Name		Type	
	0x00000010$_h$	A of ABN_2_RAW		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	A of ABN_2_RAW 0: off				

Address	Registername				Access
	1: on				
Mask	Name			Type	
0x00000020$_h$	B of ABN_2_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	B of ABN_2_RAW 0: off 1: on				
Mask	Name			Type	
0x00000040$_h$	N of ABN_2_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	N of ABN_2_RAW 0: off 1: on				
Mask	Name			Type	
0x00000080$_h$	-			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	— 0: off 1: on				
Mask	Name			Type	
0x00000100$_h$	HALL_UX of HALL_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	HALL_UX of HALL_RAW 0: off 1: on				
Mask	Name			Type	
0x00000200$_h$	HALL_V of HALL_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	HALL_V of HALL_RAW 0: off				

Address	Registername				Access
	1: on				
Mask	Name			Type	
$0x00000400_h$	HALL_WY of HALL_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	HALL_WY of HALL_RAW 0: off 1: on				
Mask	Name			Type	
$0x00000800_h$	-			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	— 0: off 1: on				
Mask	Name			Type	
$0x00001000_h$	REF_SW_R_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	REF_SW_R_RAW 0: off 1: on				
Mask	Name			Type	
$0x00002000_h$	REF_SW_H_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	REF_SW_H_RAW 0: off 1: on				
Mask	Name			Type	
$0x00004000_h$	REF_SW_L_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	REF_SW_L_RAW 0: off				

Address	Registername				Access
	1: on				
Mask	Name			Type	
$0x00008000_h$	ENABLE_IN_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	ENABLE_IN_RAW 0: off 1: on				
Mask	Name			Type	
$0x00010000_h$	STP of DIRSTP_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	STP of DIRSTP_RAW 0: off 1: on				
Mask	Name			Type	
$0x00020000_h$	DIR of DIRSTP_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	DIR of DIRSTP_RAW 0: off 1: on				
Mask	Name			Type	
$0x00040000_h$	PWM_IN_RAW			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	PWM_IN_RAW 0: off 1: on				
Mask	Name			Type	
$0x00080000_h$	-			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	— 0: off				

Address	Registername				Access
	1: on				
Mask	Name			Type	
$0x00100000_h$	HALL_UX_FILT			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	ESI_0 of ESI_RAW 0: off 1: on				
Mask	Name			Type	
$0x00200000_h$	HALL_V_FILT			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	ESI_1 of ESI_RAW 0: off 1: on				
Mask	Name			Type	
$0x00400000_h$	HALL_WY_FILT			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	ESI_2 of ESI_RAW 0: off 1: on				
Mask	Name			Type	
$0x00800000_h$	-			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	— 0: off 1: on				
Mask	Name			Type	
$0x01000000_h$	-			Bool	
	Min	Max	Default	Unit	
	0	1	0		
	CFG_0 of CFG 0: off				

Address	Registername				Access
	1: on				
	Mask	Name		Type	
	$0x02000000_h$	-		Bool	
		Min	Max	Default	Unit
		0	1	0	
		CFG_1 of CFG 0: off 1: on			
	Mask	Name		Type	
	$0x04000000_h$	-		Bool	
		Min	Max	Default	Unit
		0	1	0	
		CFG_2 of CFG 0: off 1: on			
	Mask	Name		Type	
	$0x08000000_h$	-		Bool	
		Min	Max	Default	Unit
		0	1	0	
		CFG_3 of CFG 0: off 1: on			
	Mask	Name		Type	
	$0x10000000_h$	PWM_IDLE_L_RAW		Bool	
		Min	Max	Default	Unit
		0	1	0	
		PWM_IDLE_L_RAW 0: off 1: on			
	Mask	Name		Type	
	$0x20000000_h$	PWM_IDLE_H_RAW		Bool	
		Min	Max	Default	Unit
		0	1	0	
		PWM_IDLE_H_RAW 0: off			

Address	Registername				Access
		1: on			
	Mask	Name		Type	
	0x40000000$_h$	-		Bool	
		Min	Max	Default	Unit
		0	1	0	
		DRV_ERR_IN_RAW 0: off 1: on			
	Mask	Name		Type	
	0x80000000$_h$	-		Bool	
		Min	Max	Default	Unit
		0	1	0	
		— 0: off 1: on			
0x77$_h$	TMC4671_OUTPUTS_RAW				R
	Mask	Name		Type	
	0x00000001$_h$	TMC4671_OUTPUTS_RAW[0]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		PWM_UX1_L 0: off 1: on			
	Mask	Name		Type	
	0x00000002$_h$	TMC4671_OUTPUTS_RAW[1]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		PWM_UX1_H 0: off 1: on			
	Mask	Name		Type	
	0x00000004$_h$	TMC4671_OUTPUTS_RAW[2]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		PWM_VX2_L			

Address	Registername				Access
	0: off 1: on				
	Mask	Name		Type	
	0x00000008$_h$	TMC4671_OUTPUTS_RAW[3]		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	PWM_VX2_H 0: off 1: on				
	Mask	Name		Type	
	0x00000010$_h$	TMC4671_OUTPUTS_RAW[4]		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	PWM_WY1_L 0: off 1: on				
	Mask	Name		Type	
	0x00000020$_h$	TMC4671_OUTPUTS_RAW[5]		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	PWM_WY1_H 0: off 1: on				
	Mask	Name		Type	
	0x00000040$_h$	TMC4671_OUTPUTS_RAW[6]		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	PWM_Y2_L 0: off 1: on				
	Mask	Name		Type	
	0x00000080$_h$	TMC4671_OUTPUTS_RAW[7]		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	PWM_Y2_H				

Address	Registername				Access
		0: off 1: on			
$0x78_h$	STEP_WIDTH				RW
	Mask	Name		Type	
	$0xFFFFFFFF_h$	STEP_WIDTH		Signed	
		Min	Max	Default	Unit
		-2147483648	2147483647	0	
		STEP WIDTH = 0 => STP pulses ignored, resulting direction = DIR XOR sign(STEP_WIDTH), effects PID_POSITION_TARGET			
$0x79_h$	UART_BPS				RW
	Mask	Name		Type	
	$0x00FFFFFF_h$	UART_BPS		Unsigned	
		Min	Max	Default	Unit
		0	16777215	9600	
		9600, 115200, 921600, 3000000 (default=9600)			
$0x7A_h$	UART_ADDRS				RW
	Mask	Name		Type	
	$0x000000FF_h$	ADDR_A		Unsigned	
		Min	Max	Default	Unit
		0	255	0	
	Mask	Name		Type	
	$0x0000FF00_h$	ADDR_B		Unsigned	
		Min	Max	Default	Unit
		0	255	0	
	Mask	Name		Type	
	$0x00FF0000_h$	ADDR_C		Unsigned	
		Min	Max	Default	Unit
		0	255	0	
	Mask	Name		Type	
	$0xFF000000_h$	ADDR_D		Unsigned	
		Min	Max	Default	Unit
		0	255	0	
$0x7B_h$	GPIO_dsADCI_CONFIG				RW
	Mask	Name		Type	
	$0x00000001_h$	GPIO_dsADCI_CONFIG[0]		Bool	

Address	Registername				Access
	Min	Max	Default	Unit	
	0	1	0		
	SEL_nDBGSPIM_GPIO 0: off 1: on				
	Mask	Name		Type	
	$0x00000002_h$	GPIO_dsADCI_CONFIG[1]		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	SEL_nGPIO_dsADCS_A 0: off 1: on				
	Mask	Name		Type	
	$0x00000004_h$	GPIO_dsADCI_CONFIG[2]		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	SEL_nGPIO_dsADCS_B 0: off 1: on				
	Mask	Name		Type	
	$0x00000008_h$	GPIO_dsADCI_CONFIG[3]		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	SEL_GPIO_GROUP_A_nIN_OUT 0: off 1: on				
	Mask	Name		Type	
	$0x00000010_h$	GPIO_dsADCI_CONFIG[4]		Bool	
	Min	Max	Default	Unit	
	0	1	0		
	SEL_GPIO_GROUP_B_nIN_OUT 0: off 1: on				
	Mask	Name		Type	
	$0x00000020_h$	GPIO_dsADCI_CONFIG[5]		Bool	

Address	Registername				Access
		Min	Max	Default	Unit
		0	1	0	
		SEL_GROUP_A_DSADCS_nCLKIN_CLKOUT 0: off 1: on			
	Mask	Name			Type
	0x00000040$_h$	GPIO_dsADCI_CONFIG[6]			Bool
		Min	Max	Default	Unit
		0	1	0	
		SEL_GROUP_B_DSADCS_nCLKIN_CLKOUT 0: off 1: on			
	Mask	Name			Type
	0x00FF0000$_h$	GPO			Unsigned
		Min	Max	Default	Unit
		0	255	0	
	Mask	Name			Type
	0xFF000000$_h$	GPI			Unsigned
		Min	Max	Default	Unit
		0	255	0	
0x7C$_h$	STATUS_FLAGS				RW
	Mask	Name			Type
	0x00000001$_h$	STATUS_FLAGS[0]			Bool
		Min	Max	Default	Unit
		0	1	0	
		pid_x_target_limit 0: off 1: on			
	Mask	Name			Type
	0x00000002$_h$	STATUS_FLAGS[1]			Bool
		Min	Max	Default	Unit
		0	1	0	
		pid_x_target_ddt_limit 0: off 1: on			

Address	Registername				Access
	Mask	Name		Type	
	0x00000004$_h$	STATUS_FLAGS[2]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_x_errsum_limit 0: off 1: on			
	Mask	Name		Type	
	0x00000008$_h$	STATUS_FLAGS[3]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_x_output_limit 0: off 1: on			
	Mask	Name		Type	
	0x00000010$_h$	STATUS_FLAGS[4]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_v_target_limit 0: off 1: on			
	Mask	Name		Type	
	0x00000020$_h$	STATUS_FLAGS[5]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_v_target_ddt_limit 0: off 1: on			
	Mask	Name		Type	
	0x00000040$_h$	STATUS_FLAGS[6]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_v_errsum_limit 0: off 1: on			

Address	Registername				Access
	Mask	Name		Type	
	$0x00000080_h$	STATUS_FLAGS[7]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_v_output_limit 0: off 1: on			
	Mask	Name		Type	
	$0x00000100_h$	STATUS_FLAGS[8]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_id_target_limit 0: off 1: on			
	Mask	Name		Type	
	$0x00000200_h$	STATUS_FLAGS[9]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_id_target_ddt_limit 0: off 1: on			
	Mask	Name		Type	
	$0x00000400_h$	STATUS_FLAGS[10]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_id_errsum_limit 0: off 1: on			
	Mask	Name		Type	
	$0x00000800_h$	STATUS_FLAGS[11]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_id_output_limit 0: off 1: on			

Address	Registername				Access
	Mask	Name		Type	
	$0x00001000_h$	STATUS_FLAGS[12]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_iq_target_limit 0: off 1: on			
	Mask	Name		Type	
	$0x00002000_h$	STATUS_FLAGS[13]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_iq_target_ddt_limit 0: off 1: on			
	Mask	Name		Type	
	$0x00004000_h$	STATUS_FLAGS[14]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_iq_errsum_limit 0: off 1: on			
	Mask	Name		Type	
	$0x00008000_h$	STATUS_FLAGS[15]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pid_iq_output_limit 0: off 1: on			
	Mask	Name		Type	
	$0x00010000_h$	STATUS_FLAGS[16]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		ipark_cirlim_limit_u_d 0: off 1: on			

Address	Registername				Access
	Mask	Name		Type	
	$0x00020000_h$	STATUS_FLAGS[17]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		ipark_cirlim_limit_u_q 0: off 1: on			
	Mask	Name		Type	
	$0x00040000_h$	STATUS_FLAGS[18]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		ipark_cirlim_limit_u_r 0: off 1: on			
	Mask	Name		Type	
	$0x00080000_h$	STATUS_FLAGS[19]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		not_PLL_locked 0: off 1: on			
	Mask	Name		Type	
	$0x00100000_h$	STATUS_FLAGS[20]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		ref_sw_r 0: off 1: on			
	Mask	Name		Type	
	$0x00200000_h$	STATUS_FLAGS[21]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		ref_sw_h 0: off 1: on			

Address	Registername				Access
	Mask	Name		Type	
	$0x00400000_h$	STATUS_FLAGS[22]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		ref_sw_l 0: off 1: on			
	Mask	Name		Type	
	$0x00800000_h$	STATUS_FLAGS[23]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		— 0: off 1: on			
	Mask	Name		Type	
	$0x01000000_h$	STATUS_FLAGS[24]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pwm_min 0: off 1: on			
	Mask	Name		Type	
	$0x02000000_h$	STATUS_FLAGS[25]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		pwm_max 0: off 1: on			
	Mask	Name		Type	
	$0x04000000_h$	STATUS_FLAGS[26]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		adc_i_clipped 0: off 1: on			

Address	Registername				Access
	Mask	Name		Type	
	$0x08000000_h$	STATUS_FLAGS[27]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		aenc_clipped 0: off 1: on			
	Mask	Name		Type	
	$0x10000000_h$	STATUS_FLAGS[28]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		enc_n 0: off 1: on			
	Mask	Name		Type	
	$0x20000000_h$	STATUS_FLAGS[29]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		enc_2_n 0: off 1: on			
	Mask	Name		Type	
	$0x40000000_h$	STATUS_FLAGS[30]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		aenc_n 0: off 1: on			
	Mask	Name		Type	
	$0x80000000_h$	STATUS_FLAGS[31]		Bool	
		Min	Max	Default	Unit
		0	1	0	
		wd_error 0: off 1: on			

Address	Registername				Access
$0x7D_h$	STATUS_MASK				RW
	Mask	Name		Type	
	$0xFFFFFFFF_h$	WARNING_MASK		Unsigned	
		Min	Max	Default	Unit
		0	4294967295	0	

Table 19: Register Map for TMC4671

7 Pinning

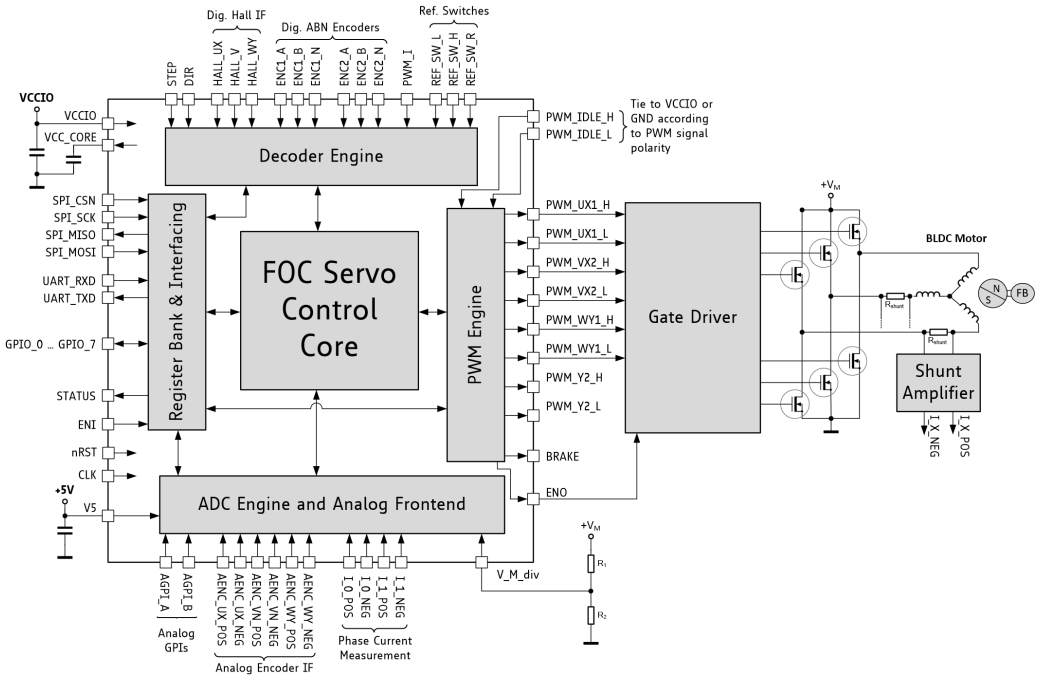


Figure 31: TMC4671 Pinout with 3 phase Power stage and BLDC Motor

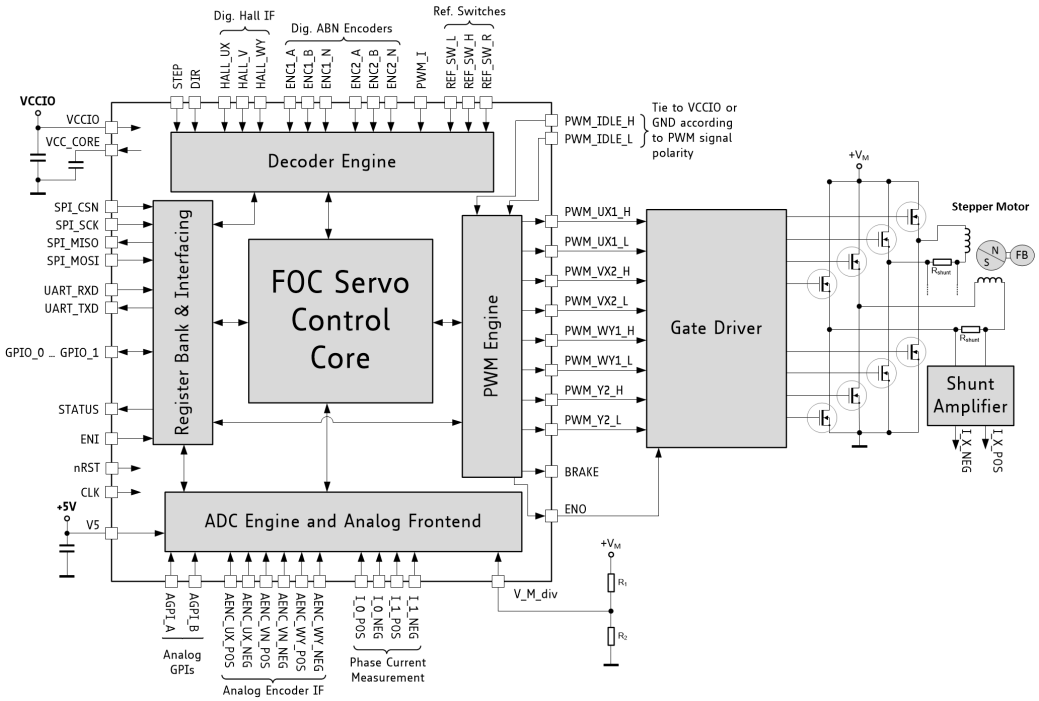


Figure 32: TMC4671 Pinout with Stepper Motor

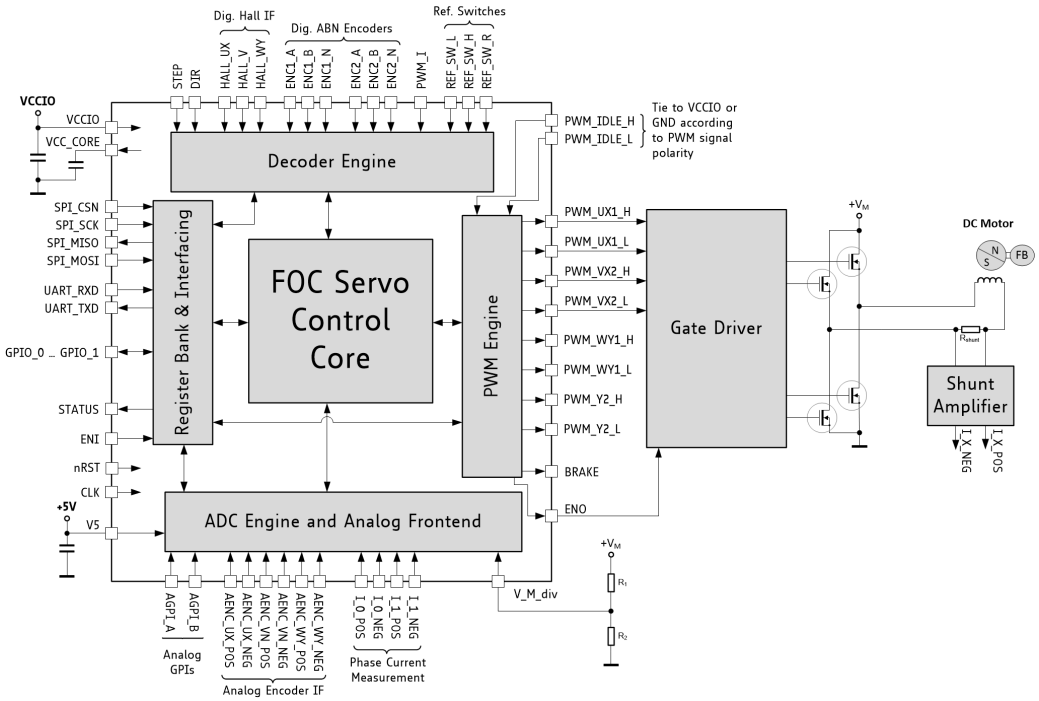


Figure 33: TMC4671 Pinout with DC Motor or Voice Coil

Info

All power supply pins (VCC, VCC_CORE) must be connected.

All ground pins (GND, GNDA, . . .) must be connected.

Analog inputs (AI) are 5V single ended or differential inputs (Input range: GNDA to V5). Use voltage dividers or operational amplifiers to scale down higher input voltages.

Digital inputs (I) resp. (IO) are 3.3V single ended inputs.

IO	Description
AI	analog input, 3.3V
I	digital input, 3.3V
IO	digital input or digital output, direction programmable, 3.3V
O	digital output, 3.3V

Table 20: Pin Type Definition

8 TMC4671 Pin Table

Name	Pin	IO	Description
nRST	50	I	active low reset input
CLK	51	I	clock input; needs to be 25 MHz for correct timing
TEST	54	I	TEST input, must be connected to GND
ENI	55	I	enable input
ENO	32	O	enable output
STATUS	12	O	output for interrupt of CPU (Warning & Status Change)
SPI_nSCS	6	I	SPI active low chip select input
SPI_SCK	7	I	SPI clock input
SPI_MOSI	8	I	SPI master out slave input
SPI_MISO	9	O	SPI master in slave output, high impedance, when SPI_nSCS = '1'
UART_RXD	10	I	UART receive data RxD for in-system-user communication channel
UART_TXD	11	O	UART transmit data TXD for in-system-user secondary communication channel
PWM_I	58	I	PWM input
DIR	56	I	direction input of step-direction interface
STP	57	I	step pulse input for step-direction interface
HALL_UX	38	I	digital hall input H1 for 3-phase (U) or 2-phase (X)
HALL_V	37	I	digital hall input H2 for 3-phase (V)
HALL_WY	36	I	digital hall input H3 for 3-phase (W) or 2-phase (Y)
ENC_A	35	I	A input of incremental encoder
ENC_B	34	I	B input of incremental encoder
ENC_N	33	I	N input of incremental encoder
ENC2_A	64	I	A input of incremental encoder
ENC2_B	65	I	B input of incremental encoder
ENC2_N	66	I	N input of incremental encoder
REF_L	67	I	Left (L) reference switch
REF_H	68	I	Home (H) reference switch
REF_R	69	I	Right (R) reference switch
ADC_I0_POS	16	AI	pos. input for phase current signal measurement I0 (I_U, I_X)

Name	Pin	IO	Description
ADC_I0_NEG	17	AI	neg. input for phase current signal measurement I0 (I_U, I_X)
ADC_I1_POS	18	AI	pos. input for phase current signal measurement I1 (I_V, I_W, I_Y)
ADC_I1_NEG	19	AI	neg. input for phase current signal measurement I1 (I_V, I_W, I_Y)
ADC_VM	20	AI	analog input for motor supply voltage divider (VM) measurement
AGPI_A	21	AI	analog general purpose input A (analog GPI)
AGPI_B	22	AI	analog general purpose input B (analog GPI)
AENC_UX_POS	25	AI	pos. analog input for Hall or analog encoder signal, 3-phase (U) or 2-phase (X (cos))
AENC_UX_NEG	26	AI	neg. analog input for Hall or analog encoder signal, 3-phase (U) or 2-phase (X (cos))
AENC_VN_POS	27	AI	pos. analog input for Hall or analog encoder signal, 3-phase (V) or 2-phase (N)
AENC_VN_NEG	28	AI	neg. analog input for Hall or analog encoder signal, 3-phase (V) or 2-phase (N)
AENC_WY_POS	29	AI	pos. analog input for Hall or analog encoder signal, 3-phase (W) or 2-phase (Y (sin))
AENC_WY_NEG	30	AI	neg. analog input for Hall or analog encoder signal, 3-phase (W) or 2-phase (Y (sin))
GPIO0 / ADC_I0_MCD	70	IO	GPIO or ΔΣ-Demodulator clock input MCLKI, clock output MCLKO, or single bit DAC output MDAC for ADC_I_0
GPIO1 / ADC_I1_MCD	71	IO	GPIO or ΔΣ-Demodulator clock input MCLKI, clock output MCLKO, or single bit DAC output MDAC for ADC_I_1
GPIO2 / ADC_VM_MCD	74	IO	GPIO or ΔΣ-Demodulator clock input MCLKI, clock output MCLKO, or single bit DAC output MDAC for ADC_VM_MCD
GPIO3 / AGPI_A_MCD / DBGSPI_nSCS	75	IO	GPIO or ΔΣ-Demodulator clock input MCLKI, clock output MCLKO, or single bit DAC output MDAC for AENC_UX_MCD, SPI debug port pin DBGSPI_nSCS
GPIO4 / AGPI_B_MCD / DBGSPI_SCK	76	IO	GPIO or ΔΣ-Demodulator clock input MCLKI, clock output MCLKO, or single bit DAC output MDAC for AENC_VN_MCD, SPI debug port pin DBGSPI_SCK
GPIO5 / AENC_UX_MCD / DBGSPI_MOSI	1	IO	GPIO or ΔΣ-Demodulator clock input MCLKI, clock output MCLKO, or single bit DAC output MDAC for AENC_WY_MCD, SPI debug port pin DBGSPI_MOSI

Name	Pin	IO	Description
GPIO6 / AENC_VN_MCD / DBGSPI_MISO	4	IO	GPIO or ΔΣ-Demodulator clock input MCLKI, clock output MCLKO, or single bit DAC output MDAC for AGPI_A_MCD, SPI debug port pin DBGSPI_MISO
GPIO7 / AENC_WY_MCD / DBGSPI_TRG	5	IO	GPIO or ΔΣ-Demodulator clock input MCLKI, clock output MCLKO, or single bit DAC output MDAC for AGPI_B_MCD, SPI debug port pin DBGSPI_TRG
PWM_IDLE_H	59	I	idle level of high side gate control signals
PWM_IDLE_L	60	I	idle level of low side gate control signals
PWM_UX1_H	39	O	high side gate control output U (3-phase) resp. X1 (2-phase)
PWM_UX1_L	40	O	low side gate control output U (3-phase) resp. X1 (2-phase)
PWM_VX2_H	41	O	high side gate control output V (3-phase) resp. X2 (2-phase)
PWM_VX2_L	42	O	low side gate control output V (3-phase) resp. X2 (2-phase)
PWM_WY1_H	46	O	high side gate control output W (3-phase) resp. Y1 (2-phase)
PWM_WY1_L	47	O	low side gate control output W (3-phase) resp. Y1 (2-phase)
PWM_Y2_H	48	O	high side gate control output Y2 (2-phase only)
PWM_Y2_L	49	O	low side gate control output Y2 (2-phase only)
BRAKE	31	O	brake chopper control output signal

Table 21: Functional Pin Description

Name	Pin	IO	Description
VCCIO1	2	3.3V	digital IO supply voltage; use 100 nF decoupling capacitor
VCCIO2	13	3.3V	digital IO supply voltage; use 100 nF decoupling capacitor
VCCIO3	43	3.3V	digital IO supply voltage; use 100 nF decoupling capacitor
VCCIO4	52	3.3V	digital IO supply voltage; use 100 nF decoupling capacitor
VCCIO5	61	3.3V	digital IO supply voltage; use 100 nF decoupling capacitor
VCCIO6	72	3.3V	digital IO supply voltage; use 100 nF decoupling capacitor
GNDIO1	3	0V	digital IO ground
GNDIO2	14	0V	digital IO ground
GNDIO3	44	0V	digital IO ground
GNDIO4	53	0V	digital IO ground
GNDIO5	62	0V	digital IO ground
GNDIO6	73	0V	digital IO ground
VCCCORE1	15	1.8V	digital core supply voltage output; use 100 nF decoupling capacitor
VCCCORE2	45	1.8V	digital core supply voltage output; use 100 nF decoupling capacitor
VCCCORE3	63	1.8V	digital core supply voltage output; use 100 nF decoupling capacitor
V5	23	5V	analog reference voltage
GNDA	24	0V	analog reference ground
GNDPAD	–	0V	bottom ground pad

Table 22: Supply Voltage Pins and Ground Pins

9 Electrical Characteristics

9.1 Absolute Maximum Ratings

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

Parameter	Symbol	Min	Max	Unit
Digital I/O supply voltage	VCCIO		3.6	V
Logic input voltage	VI		3.6	V
Maximum current drawn on VCCIO with no load on pins	I_IO		70	mA
Maximum current drawn on VCCIO with no load on pins and clock off	I_IO_0Hz		3	mA
Maximum current drawn on V5 at fCLK = 25MHz	I_V5		25	mA
Maximum current to / from digital pins and analog low voltage I/Os	IIO		10	mA
Junction temperature	TJ	-40	125	°C
Storage temperature	TSTG	-55	150	°C
ESD-Protection for interface pins (Human body model, HBM)	VESDAP		2	kV
ESD-Protection for handling (Human body model, HBM)	VESD1		2	kV
ADC input voltage	VAI	0	5	V

Table 23: Absolute Maximum Ratings

VCCCORE is generated internally from VCCIO and shall not be overpowered by external supply.

9.2 Electrical Characteristics

9.2.1 Operational Range

Parameter	Symbol	Min	Max	Unit
Junction temperature	TJ	-40	125	°C
Digital I/O 3.3V supply voltage	VIO3V	3.15	3.45	V
Core supply voltage	VCC_CORE	1.65	1.95	V

Table 24: Operational Range

The ΔΣ ADCs can operate in differential or single ended mode. In differential mode the differential input voltage range must be in between -2.5V and +2.5V. However, it is recommended to use the input voltage range from -1.25V to 1.25V, due to non-linearity of ΔΣ ADCs. In Single ended mode the operational input range of the positive input channel should be between 0V and 2.5V. Recommended maximum input voltage is 1.25V. ADCs have

9.2.2 DC Characteristics

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

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Input voltage low level	VINL	VCCIO = 3.3V	-0.3		0.8	V
Input voltage high level	VINH	VCCIO = 3.3V	2.3		3.6	V
Input with pull-down		VIN = 3.3V	5	30	110	μA
Input with pull-up		VIN = 0V	-110	-30	-5	μA
Input low current		VIN = 0V	-10		10	μA
Input high current		VIN = VCCIO	-10		10	μA
Output voltage low level	VOUTL	VCCIO = 3.3V	0.4			V
Output voltage high level	VOUTH	VCCIO = 3.3V	2.64			V
Output driver strength standard	IOUT_DRV		4			mA
Input impedance of Analog Input	R_ADC	TJ = 25°C	85	100	115	kΩ

Table 25: DC Characteristics

All I/O lines include Schmitt-Trigger inputs to enhance noise margin.

10 Sample Circuits

Please consider electrical characteristics while designing electrical circuitry. Most Sample Circuits in this chapter were taken from the Evalutation board for the TMC4671 (TMC4671-EVAL).

10.1 Supply Pins

Please provide VCCIO and V5 to the TMC4671. VCC_CORE is internally generated and needs just an external decoupling capacitor. Place one 100nF decoupling capacitor at every supply pin. Table 26 lists additional needed decoupling capacitors.

Pin Name	Supply Voltage	Additional Cap.
V5	5V	4.7uF
VCCIO	3.3V	4.7uF & 470nF
VCCCORE	1.8V	none

Table 26: Additional decoupling capacitors for supply voltages

10.2 Clock and Reset Circuitry

The TMC4671 needs an external oscillator for correct operation at 25 MHz. Lower frequency results in respective scaling of timings. Higher frequency is not supported. The internally generated active low reset can be externally overwritten. If users want to toggle the reset, a pulse length of at least 500 ns is recommended. When not used, please apply a 10k Pull up resistor and make sure all supply voltages are stable.

10.3 Digital Encoder, Hall Sensor Interface and Reference Switches

Digital encoders, Hall sensors and reference switches usually operate on a supply voltage of 5V. As the TMC4671 is usually operated at a VCCIO Voltage of 3.3V, a protection circuit for the TMC4671 input pin is needed. In fig. 34 a sample circuit for the ENC_A signal is shown, which can be reused for all encoder and Hall signals as well as for reference switch signals. Parametrization of the components is given in table 27 for different operations.

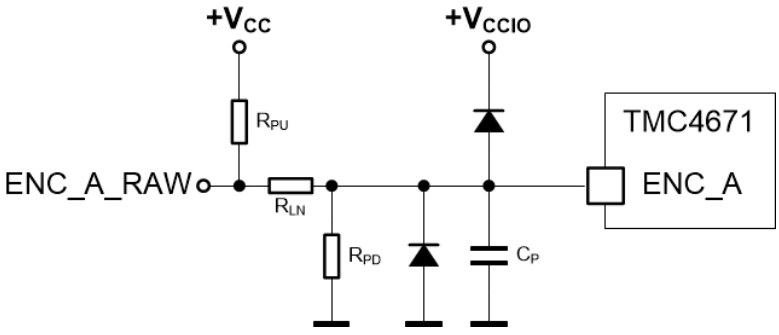


Figure 34: Sample Circuit for Interfacing of an Encoder Signal

Application	R_{PU}	R_{PD}	R_{LN}	C_P
5 V Encoder signal	4K7	n.c.	100R	100pF

Table 27: Reference Values for circuitry components

The raw signal (ENC_A_RAW) is divided by a voltage divider and filtered by a lowpass filter. A pull up resistor is applied for open collector encoder output signals. Diodes protect the input pin (ENC_A) against over- and undervoltage. The cutoff-frequency of the lowpass is:

$$f_c = \frac{1}{2\,\pi\,R_{PD}C_P} \tag{39}$$

10.4 Analog Frontend

Analog Encoders are encoding the motor position into sinusoidal signals. These signals need to be digitalized by the TMC4671 in order to determine the rotor position. The input voltage range depends on V5 input, which is usually 5V and GNDA (usually 0V). Due to nonlinearity issues of the ADC near input limits, an ADC input value from 1V to 4V is recommended. For a single ended application, the sample circuit from fig. 35 can be used. All single ended analog input pins (AGPI_A, AGPI_B and ADC_VM) have their negative input value tied to GNDA internally, so this sample circuit can also be used for them.

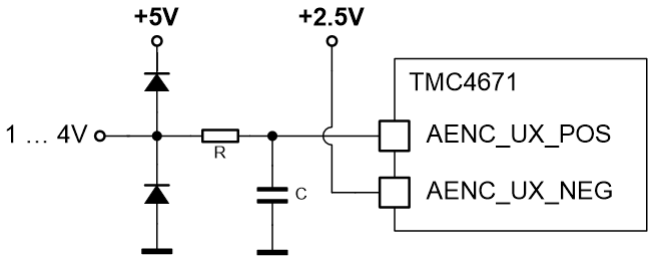


Figure 35: Sample Circuit for Interfacing of a single ended analog signal

If the power stage and the TMC4671 share a common ground, the ADC_VM input signal can be generated by a voltage divider to scale the voltage down to the needed range.
If the analog encoder has differential output signals, these can be used without signal conditioning (no OP AMPs), when voltage range matches. Differential analog inputs can be used to digitize differential analog input signals with high common mode voltage error suppression.

10.5 Phase Current Measurement

The TMC4671 requires two phase currents of a 2 or 3 phase motor to be measured. For a DC Motor only one current in the phase needs to be measured (see Fig. 37). In the ADC engine mapping of current signals to motor phases can be changed. Default setting is I0 to be the current running into the motor in phase U for a 3 phase motor. Respectively the current running into the motor from half-bridge X1 of a 2 phase motor. Figs. 36 and 37 illustrates the currents to be measured and their positive direction.

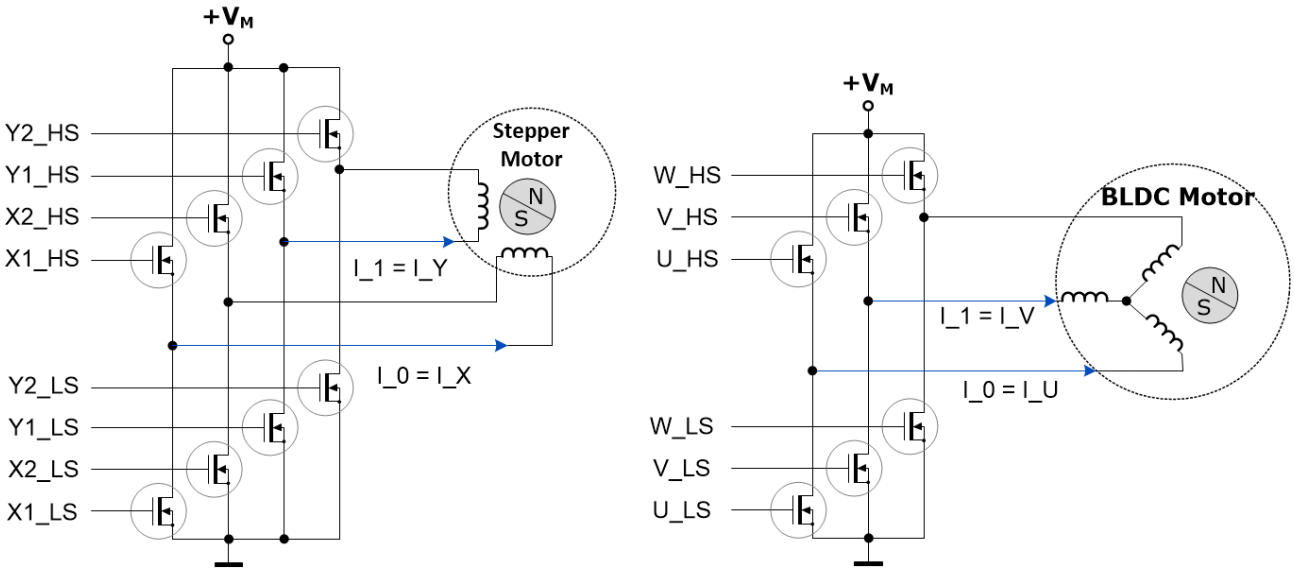


Figure 36: Phase current measurement: Current directions for 2 and 3 phase motors

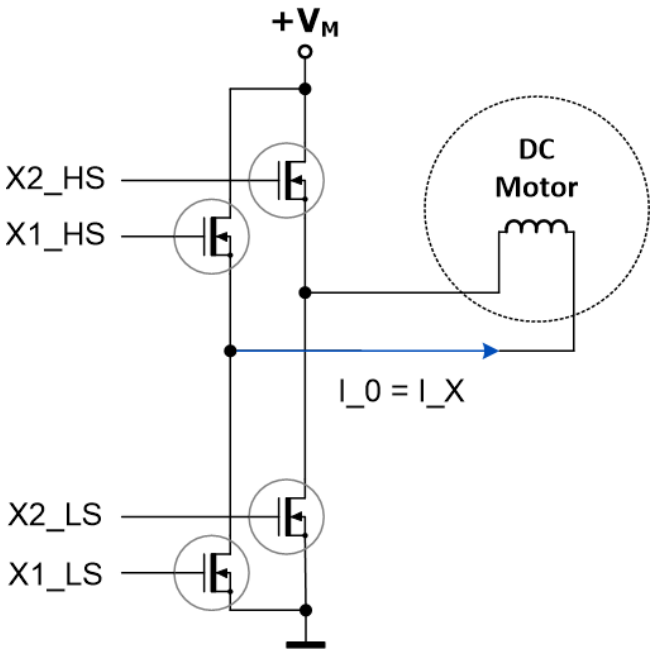


Figure 37: Phase current measurement: Current direction for DC or Voice Coil Motor

There are two main options for measuring the phase currents as described above. First option is to use a shunt resistor and a shunt amplifier like the LT1999 or the AD8418A. The other option is to use a real current sensor, which uses the Hall effect or other magnetic effects to implement an isolated current measurement. Shunt measurement might be the more cost-effective solution for low voltage applications up to 100V, while current sensors are more useful at higher voltage levels.
In general the sample circuit in fig. 38 can be used for shunt measurement circuitry. Please consider design guidelines of shunt amplifier supplier additionally. TRINAMIC also supplies power stage boards with current shunt measurement circuitry (TMC-UPS10A/70V-EVAL).

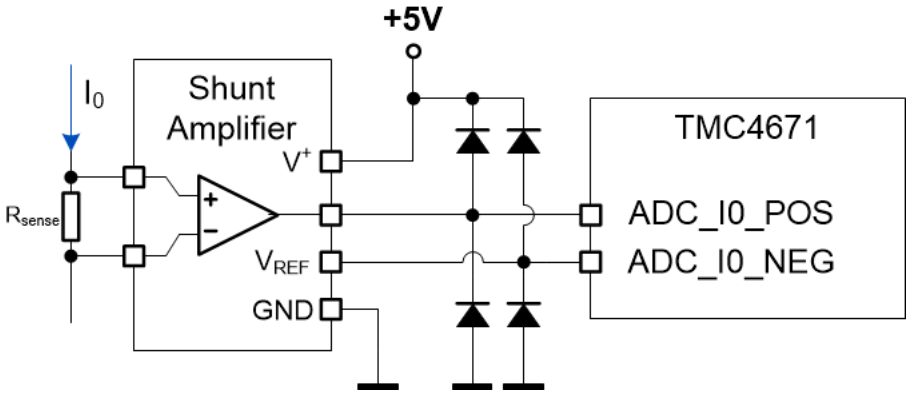


Figure 38: Current Shunt Amplifier Sample Circuit

10.6 Power Stage Interface

The TMC4671 is equipped with a configurable PWM engine for control of various gate drivers. Gate driver switch signals can be matched to power stage needs. This includes signal polarities, frequency, BBM-times for low and high side switches, and an enable signal. Please consider gate driver circuitry, when connecting to the TMC4671.

11 Setup Guidelines

For easy setup of the TMC4671 on a given hardware platform like the TMC4671 Evaluation-Kit, the user should follow these general guidelines in order to safely set up the system for various modes of operation.

Info	These guidelines fit to hardware platforms which are comparable to the TMC4671-Evaluation Kit. If system structure differs, configuration has to be adjusted. Please also make use of the RTMI Adapter and the TMCL IDE to setup the system as it reduces commissioning time significantly.

Step 0: Setup of SPI communication
As a first step of the configuration of the TMC4671 the SPI communication should be tested by reading and writing for example to the first registers for identification of the silicon. If communication fails, please check CLK and nRST signals. For easy software setup the TMC API provided on the TRINAMIC website can be used.
Step 1: Check connections
Register TMC_INPUTS_RAW can be accessed to see if all connected digital inputs are working correctly e.g. sensor signals can be checked by turning the motor manually.
Step 2: Setup of PWM and Gatedriver configuration
The user should choose the connected motor and the number of polepairs by setting register MOTOR_TYPE_N_POLE_PAIRS. For a DC motor the number of pole pairs should be set to one. The PWM can be configured with the corresponding registers PWM_POLARITIES (Gate Driver Polarities), PWM_MAXCNT (PWM Frequency), PWM_BBM_H_BBM_L (BBM times), and PWM_SV_CHOP (PWM mode). After setting the register PWM_SV_CHOP to 7 the PWM is on and ready to use.
Please check PWM outputs after turning on the PWM, if you are using a new hardware design.
Step 3: Open Loop Mode
In the Open Loop Mode the motor is turned by applying voltage to the motor. This mode is useful for test and setup of ADCs and position sensors. It is activated by setting the corresponding registers for PHI_E_SELECTION, and MODE_MOTION. With UD_EXT the applied voltage can be regulated upwards until the motor starts to turn. Acceleration and target velocity can be changed by their respective registers.
Step 4: Setup of ADC for current measurement
Please setup the current measurement by choosing your applications ADC configuration. Make sure to match decimation rate of the Delta Sigma ADCs to your choosen PWM frequency.
When the motor turns in Open Loop Mode the current measurement can be easily calibrated. Please match offset and gain of phase current signals by setting the corresponding registers. Please also make sure for a new hardware setup, that current measurements and PWM channels are matched. This can be done by matching phase voltages and phase currents. Register ADC_I_SELECT can be used to switch relations.
Step 5: Setup of Feedback Systems
In Open Loop Mode also the feedback systems can be checked for correct operation. Please configure registers related to used position sensor(s) and compare against Open Loop angles. Use encoder initialization routines to set angle offsets for relative position encoders according to application needs.
Step 6: Setup of FOC Controllers
Please configure your application's feedback system and configure position and velocity signal switches accordingly inside the FOC. Configure controller output limits according to you needs.
Setup PI controller parameters for used FOC controllers. Start with the current controller, followed by the velocity controller, followed by the position controller. Stop configuration at your desired cascade level.
TRINAMIC recommends to set the PI controller parameters by support of the RTMI, as it supports realtime access to registers and the TMCL IDE offers tools for automated controller tuning. Controller tuning without realtime access might lead to poor performance.
Please choose afterwards your desired Motion Mode and feed in reference values.
Step 7: Advanced Functions
For performance improvements Biquad filters and feed forward control can be applied.

12 Package Dimensions

Package: QFN76, 0.4 mm pitch, size 11.5 mm x 6.5 mm,
industrial temperature range 0°C . . . 85°C, RoHS compliant.

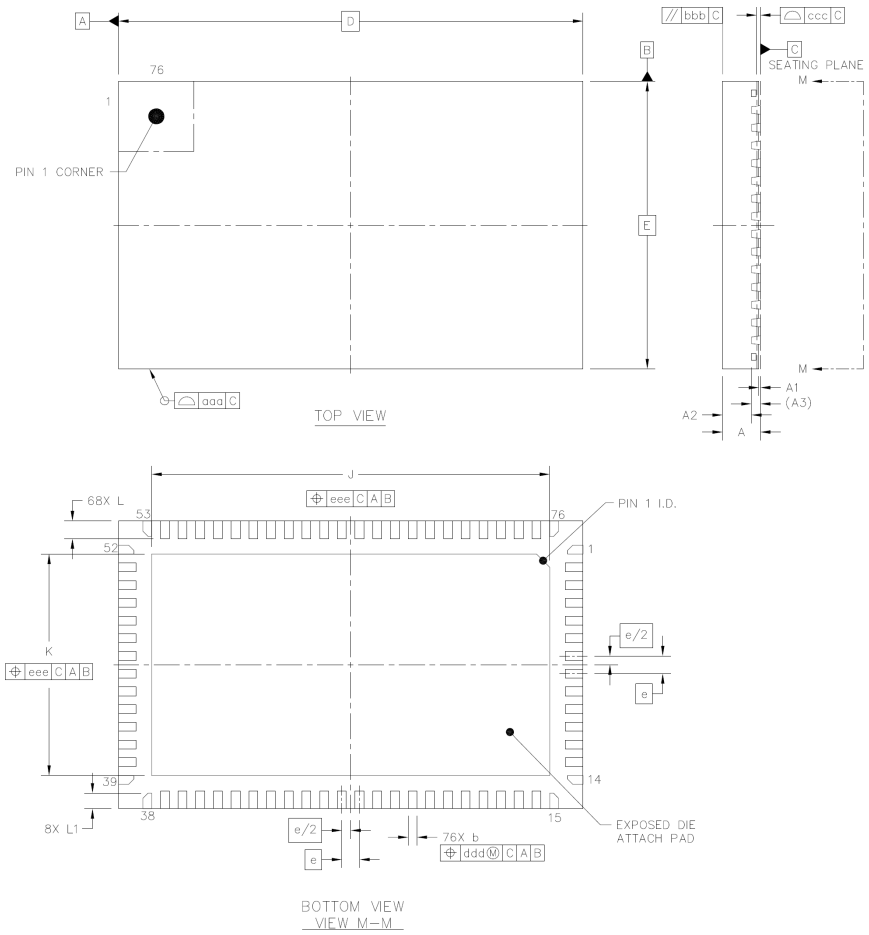


Figure 39: QFN76 Package Outline

QFN76 Package Dimensions in mm				
Description	Dimension[mm]	min.	typ.	max.
Total Thickness	A	0.80	0.85	0.90
Stand Off	A1	0.00	0.035	0.05
Mold Thickness	A2	—	0.65	—
L/F Thickness	A3		0.203 REF	
Lead Width	b	0.15	0.2	0.25
Body Width	D		10.5 BSC	
Body Length	E		6.5 BSC	
Lead Pitch	e		0.4 BSC	

QFN76 Package Dimensions in mm				
EP Size	J	8.9	9	9.1
EP Size	K	4.9	5	5.1
Lead Length	L	0.35	0.40	0.45
Lead Length	L1	0.30	0.35	0.40
Package Edge Tolerance	aaa		0.1	
Mold Flatness	bbb		0.1	
Coplanarity	ccc		0.08	
Lead Offset	ddd		0.1	
Exposed Pad Offset	eee		0.1	

Table 28: Package Outline Dimensions

Figure 40 shows the package from top view. decals for some CAD programs are available on the product's website.

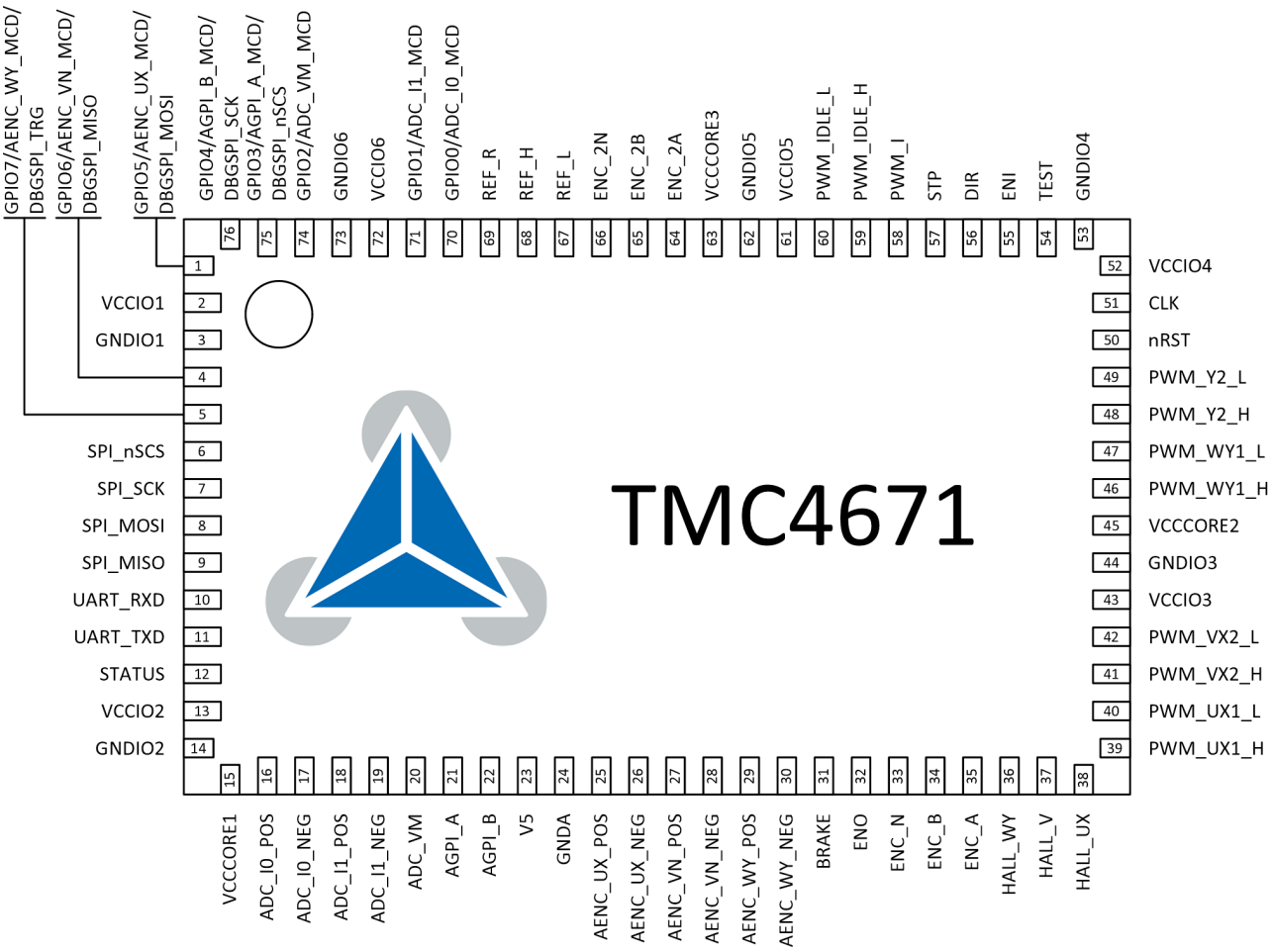


Figure 40: Pinout of TMC4671 (Top View)

13 Supplemental Directives

13.1 Producer Information

13.2 Copyright

13.3 Trademark Designations and Symbols

Trademark designations and symbols used in this documentation indicate that a product or feature is owned and registered as trademark and/or patent either by TRINAMIC or by other manufacturers, whose products are used or referred to in combination with TRINAMIC's products and TRINAMIC's product documentation.

This Datasheet is a non-commercial publication that seeks to provide concise scientific and technical user information to the target user. Thus, trademark designations and symbols are only entered in the Short Spec of this document that introduces the product at a quick glance. The trademark designation /symbol is also entered when the product or feature name occurs for the first time in the document. All trademarks and brand names used are property of their respective owners.

13.4 Target User

The documentation provided here, is for programmers and engineers only, who are equipped with the necessary skills and have been trained to work with this type of product.

The Target User knows how to responsibly make use of this product without causing harm to himself or others, and without causing damage to systems or devices, in which the user incorporates the product.

13.5 Disclaimer: Life Support Systems

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

Life support systems are equipment intended to support or sustain life, and whose failure to perform, when properly used in accordance with instructions provided, can be reasonably expected to result in personal injury or death.

Information given in this document is believed to be accurate and reliable. However, no responsibility is assumed for the consequences of its use nor for any infringement of patents or other rights of third parties which may result from its use. Specifications are subject to change without notice.

13.6 Disclaimer: Intended Use

The data specified in this user manual is intended solely for the purpose of product description. No representations or warranties, either express or implied, of merchantability, fitness for a particular purpose

or of any other nature are made hereunder with respect to information/specification or the products to which information refers and no guarantee with respect to compliance to the intended use is given.

In particular, this also applies to the stated possible applications or areas of applications of the product. TRINAMIC products are not designed for and must not be used in connection with any applications where the failure of such products would reasonably be expected to result in significant personal injury or death (safety-Critical Applications) without TRINAMIC's specific written consent.

TRINAMIC products are not designed nor intended for use in military or aerospace applications or environments or in automotive applications unless specifically designated for such use by TRINAMIC. TRINAMIC conveys no patent, copyright, mask work right or other trade mark right to this product. TRINAMIC assumes no liability for any patent and/or other trade mark rights of a third party resulting from processing or handling of the product and/or any other use of the product.

13.7 Collateral Documents & Tools

This product documentation is related and/or associated with additional tool kits, firmware and other items, as provided on the product page at: www.trinamic.com.

14 Errata

The Errata of the TMC4671-ES are listed here and in the particular descriptions they apply to.

1. SPI Slave Interface
   The SPI Slave Interface in the TMC4671-ES shows following error. During transaction of MSB of read data might get corrupted. This shows in two different ways. First one is a 40 ns pulse (positive or negative) on MISO at the beginning of transfer of that particular bit. This pulse can corrupt the MSB of read data and this error can be avoided when SPI clock frequency is set to 1 MHz. Second error also corrupts MSB of read data when MSB of regsiter is unstable. For example current measurement noise around zero. In this case MSB should be ignored when possible. Please also make sure that e.g. actual torque value can be read from register PID_TORQUE_FLUX_ACTUAL or from INTERIM_DATA register, where it is showing up in the lower 16 bits. These errors will be fixed in next IC version. SPI write access is not affected and can be performed at 8 MHz clock frequency.

2. Realtime Monitoring Interface
   The TRINAMIC Realtime Monitoring Interface can not be used with galvanic isolation as timing of SPI communication is too strict. This will be fixed in future version so galvanic isolation of SPI signals will be possible with defined latency of isolators.

3. PI Controllers
   The P Factor in the advanced position controller is not properly scaled. Due to the high gain in Velocity control loop, the position controller gain should be respectively low. The P Factor normalization of Q8.8 does not match these needs. This will be changed in a future version of the chip to a different Q Format. This change will cause changes in user's application controller software. We recommend to use the classical PI control structure if performance is not sufficient.

   The integrator in the advanced PI controller is not reset when P or I parameters are set to zero. As a workaround controller can be disabled by switching to stopped mode or to a lower cascade level and thereby the integrator is reset. This behaviour will be changed in the next IC version.

4. Inbuilt ADCs
   The inbuilt Delta Sigma ADCs show an error, where both groups are disturbing each other. When one group is deactivated, everything is fine, but with both groups being active ADC Data might be corrupted. This error occurs if clock signals of both groups are not in phase. Clock phase can be changed by toggling the dsADC_MCLK_B to a non-round figure like 0x30000001 and back to 0x20000000. This toggling has to be repeated until measurement is clean.

   If the second ADC Group is not needed, it can be shut down by setting dsADC_MCLK_B to 0. The distortion can be detected by monitoring measurement at reference voltage. Use register DS_ANALOG_INPUT_STAGE_CFG to switch on the reference voltage for monitoring.

5. Pins PWM_IDLE_H and PWM_IDLE_L without function
   Pins PWM_IDLE_H and PWM_IDLE_L are intended to determine Power on Reset Gate Driver Polarity. This feature is not working properly as the gate driver polarity always powers up to Low Side Polarity to be Active Low and High Side Polarity to be active high. This will be corrected in the next version of the chip.

6. Space Vector PWM does not allow higher voltage utilization
   The Space vector PWM does not allow higher voltage utilization. This will be fixed in next version of the chip.

7. Step Direction Counter not used as Target Position
   The step direction interface correctly counts up and down the target position, but the step direction counter position is not used as the target position for positioning as intended. The TMC4671-ES always uses the target position written via SPI, RTMI, or UART into the register bank as the target position for positioning. As a work around for evaluation of step direction target position control, the user can read out the target position periodically and write it back to the register bank as the

target position. But step loss can occur in this configuration as the step direction counter is also overwritten. This will be fixed in next version of the chip.

15 Figures Index

16 Tables Index

17 Revision History

17.1 IC Revision

Version	Date	Author	Description
V1.00	2017-JUL-03	LL, OM	Engineering samples TMC4671-ES (1v00 2017-07-03-19:43)

Table 29: IC Revision

17.2 Document Revision

Version	Date	Author	Description
V0.9	2017-SEP-29	LL, OM	Pre-liminary TMC4671-ES datasheet.
V0.91	2018-JAN-30	OM	Changed some typos and added some notes.
V0.92	2018-FEB-28	OM	Changed register descriptions.
V0.93	2018-MAR-07	OM	Changed some typos and bugs in graphics.
V0.94	2018-MAR-14	OM	Added Errata Section.
V0.95	2018-MAY-08	OM	Preparations for launch.
V1.00	2018-JUN-28	LL	Errata Section updated concerning Step/Dir.
V1.01	2018-JUL-19	OM	Added Description for Status Flags
V1.02	2018-JUL-31	OM	Added Description for Feed Forward Control Structure
V1.03	2018-SEP-06	OM	Description of single pin interface and motion modes added

Table 30: Document Revision