
Chipsmall
深圳市津利帝科技有限公司
Geelead Technology Limited
Chipsmall
采购部

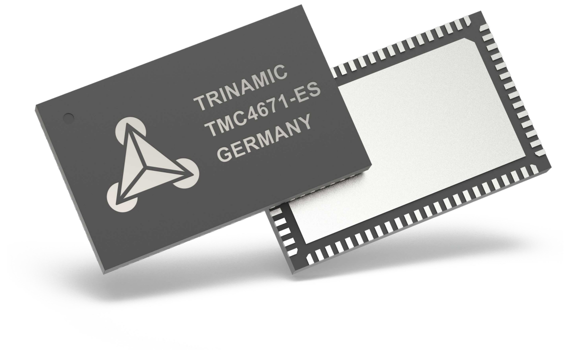
Dedicated Motion Controller for 2-/3-Phase PMSM

INTEGRATED CIRCUITS

TMC4671 Datasheet

IC Version V1.00 | Document Revision V1.00 • 2018-June-27

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.



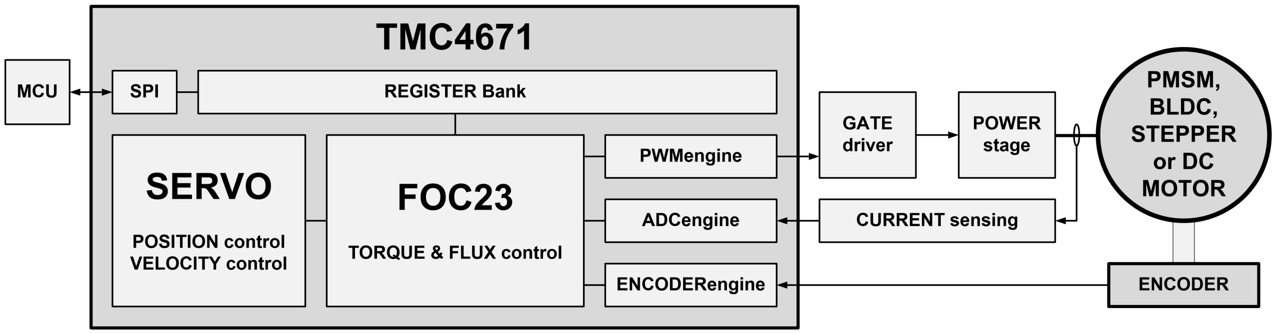
Features

- 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 (0Â°, 90Â°) or 0Â°, 120Â°, 240Â°)
  - 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.

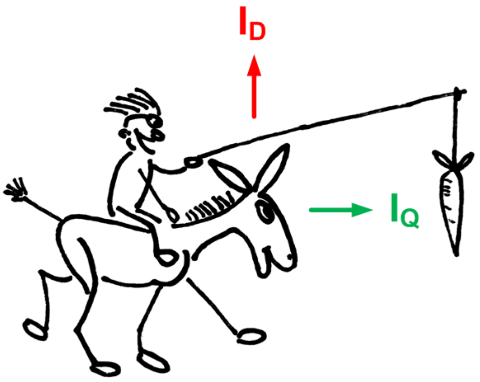


Figure 1: Illustration of the FOC basic principle by cartoon: Maximize active power and minimize idle power and power dissipation by intelligent closed-loop control.

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.

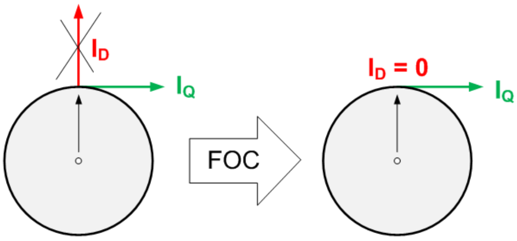


Figure 2: FOC optimizes torque by closed loop control while maximizing IQ and minimizing ID to 0

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.

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

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.

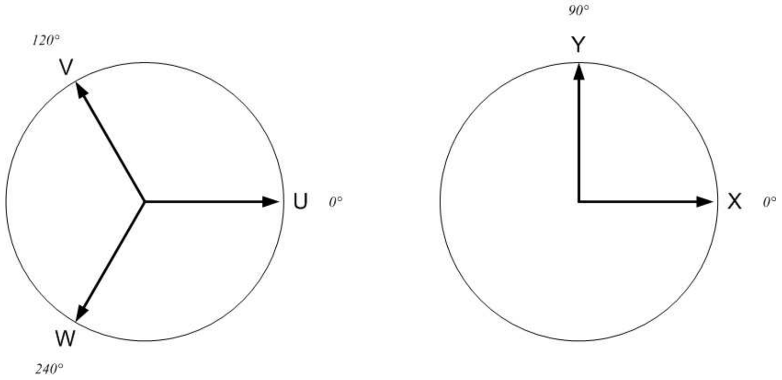


Figure 3: Orientations UVW (FOC3) and XY (FOC2)

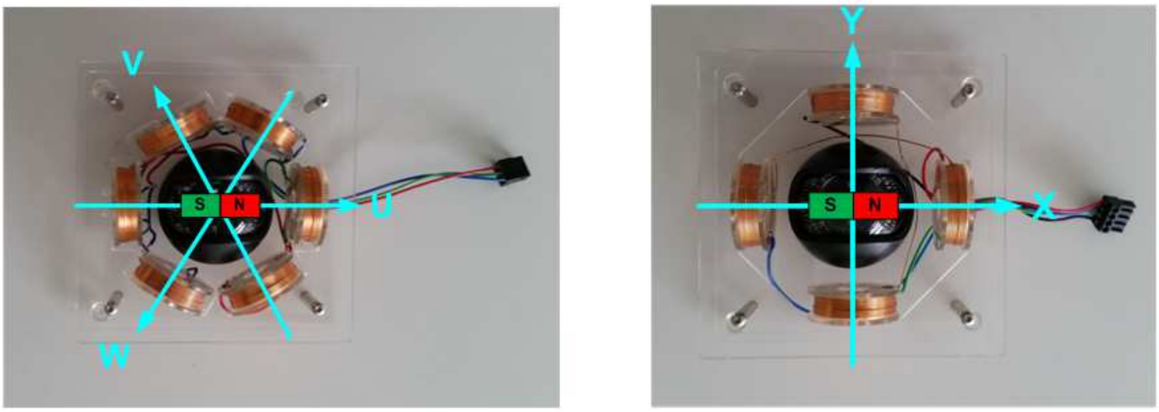


Figure 4: Compass Motor Model w/ 3 Phases UVW (FOC3) and Compass Motor Model w/ 2 Phases (FOC2)

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.

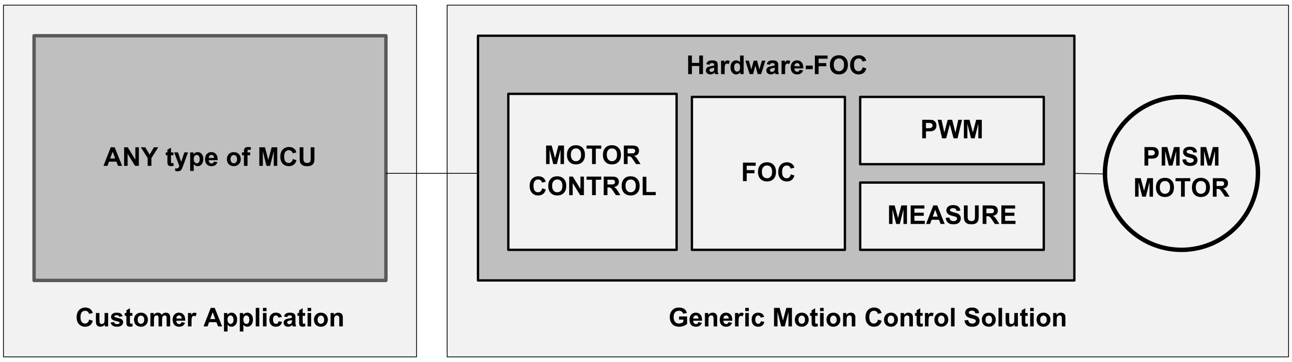


Figure 5: Hardware FOC Application Diagram

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.

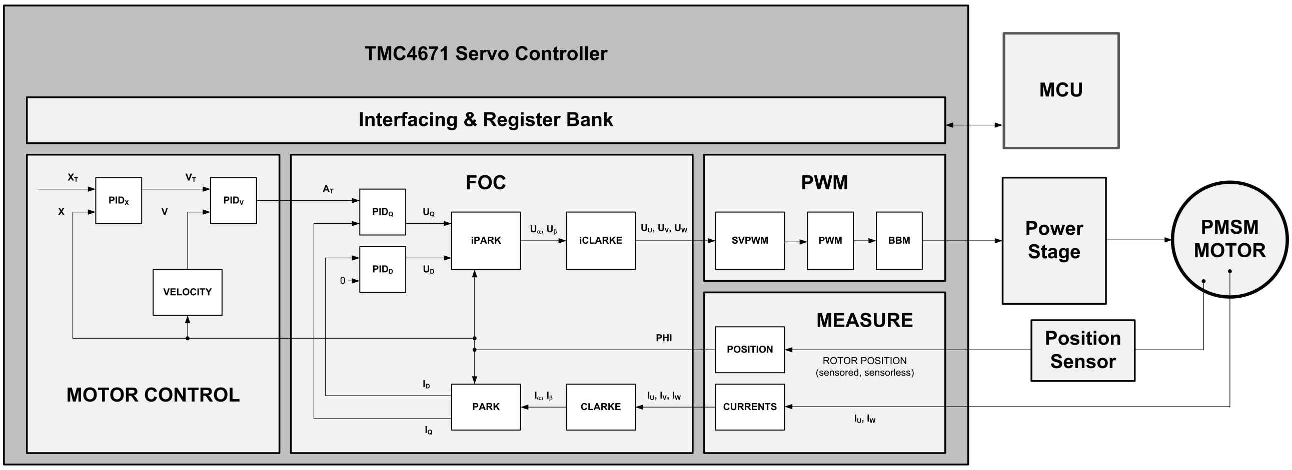


Figure 6: Hardware FOC Block Diagram

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.

<table>
<tr><td colspan="3">40 BIT DATAGRAM</td></tr>
<tr><td colspan="2">8 BIT ADDR</td><td>32 DATA</td></tr>
<tr><td>WRnRD</td><td>7 BIT ADDR</td><td>32 DATA</td></tr>
</table>

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"
```

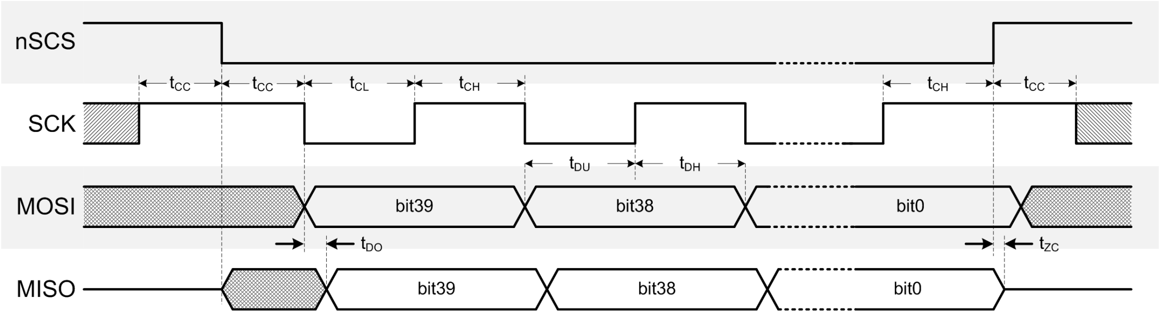


Figure 8: SPI Timing

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

ℹ ***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 (SPI Master) is an additional fast interface enabling real-time identification of motor parameters 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. If the interface is not needed, pins must 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.

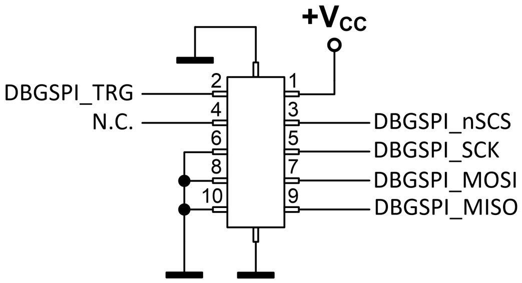


Figure 9: Connector for Real-Time Monitoring Interface (Connector Type: Hirose DF20F-10DP-1V)

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 Debug Interface

The UART debug 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 communicate with up to 3Mbps. The UART debug port enables In-System-Setup-Support by multiple-ported register access.

An UART datagram consists of five bytes - similar to the datagrams of the embedded user application interface (SPI). In contrast to the embedded user application interface (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.

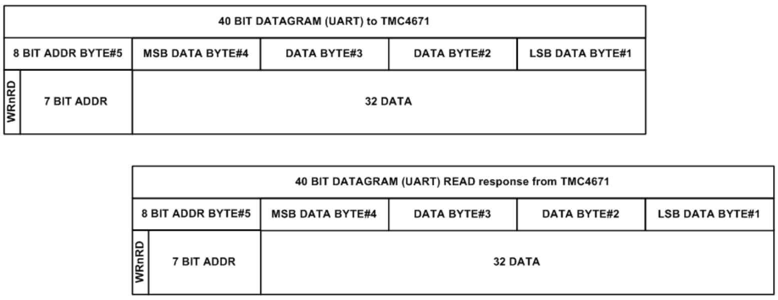


Figure 10: UART Read Datagram (TMC4671 register read via UART)

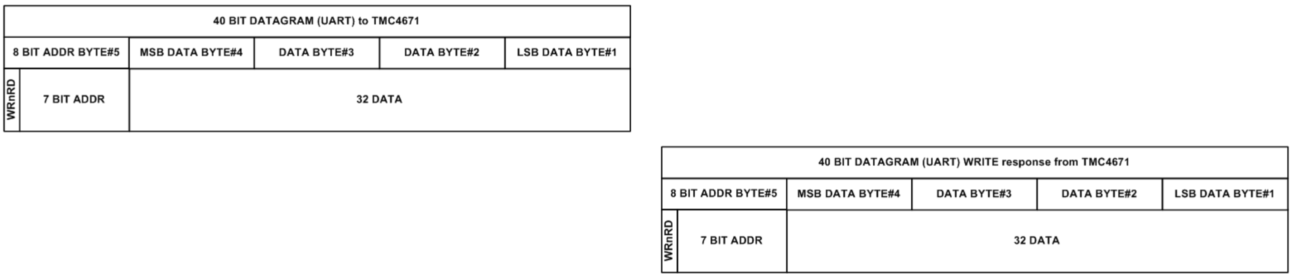


Figure 11: UART Write Datagram (TMC4671 register write via UART)

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.

ℹ ***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.

4.3 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 3: 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
0x5A81$_h$	23169	23169	23169 / 256	23169 / 4096
0x7FFF$_h$	32767	32767	32767 / 256	32767 / 4096

Hexadecimal Value	u16	s16	q8.8	q4.12
$0x8000_h$	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 4: 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.

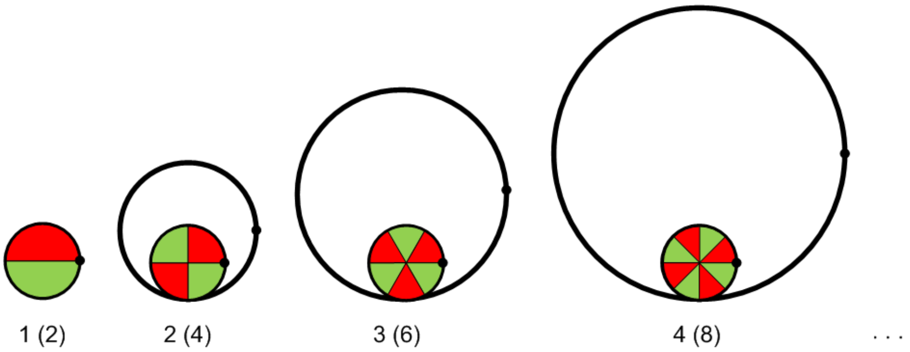


Figure 12: N_POLE_PAIRS - Number of Pole Pairs (Number of Poles)

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.

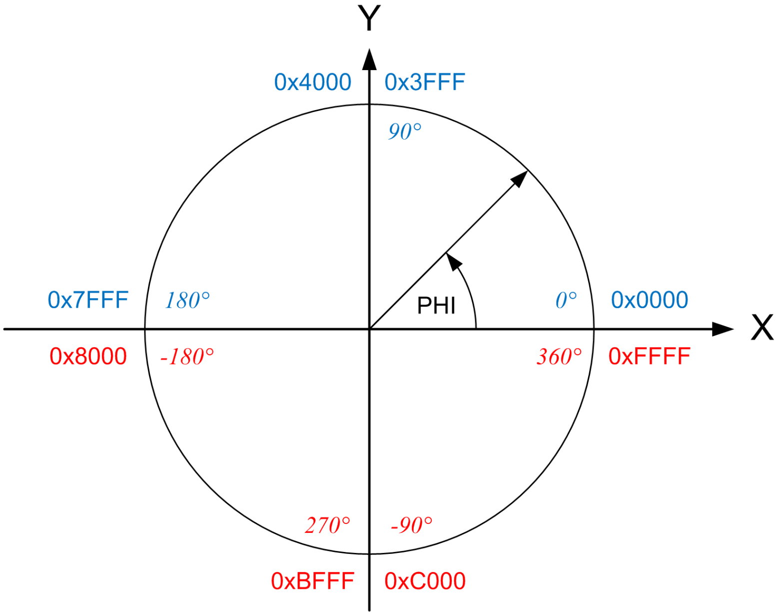


Figure 13: Integer Representation of Angles as 16 Bit signed (s16) resp. 16 Bit unsigned (u16)

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 5: 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.

ℹ ***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).

ℹ ***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).

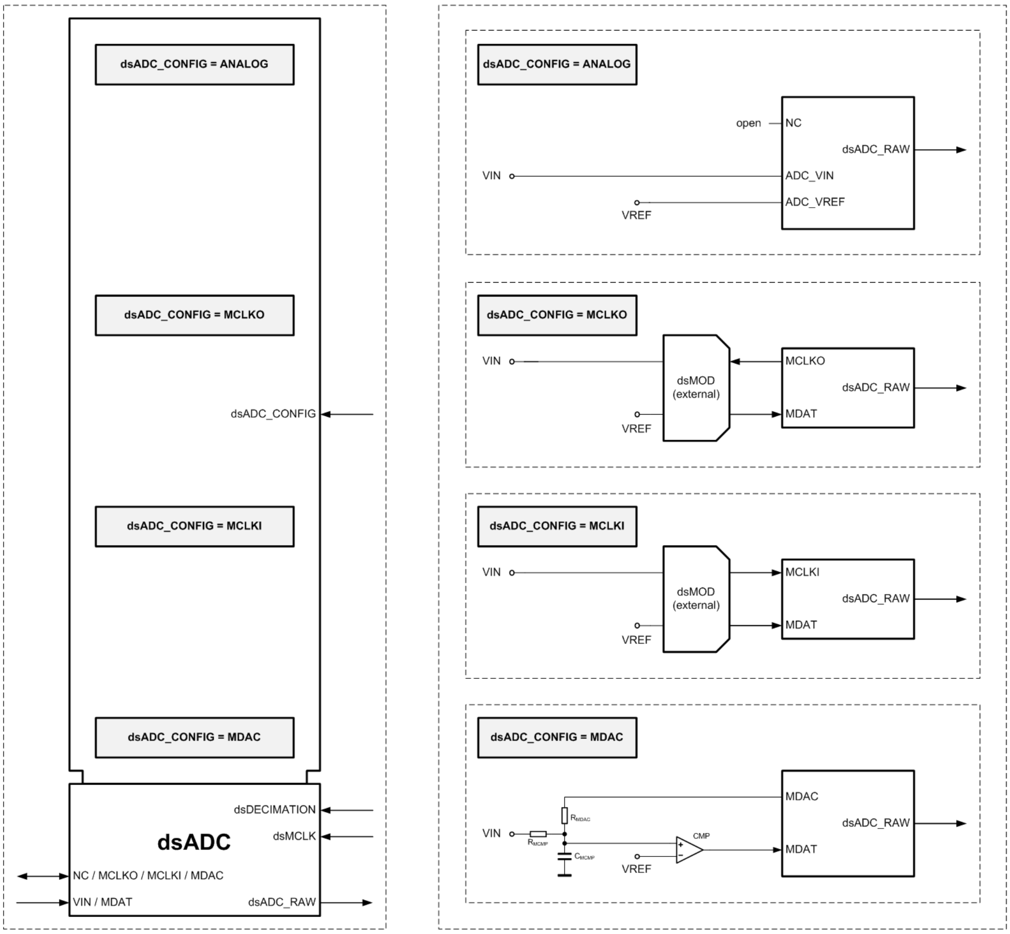


Figure 14: Delta Sigma ADC Configurations dsADC_CONFIG (internal: ANALOG vs. external: MCLKO, MCLKI, MDAC)

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