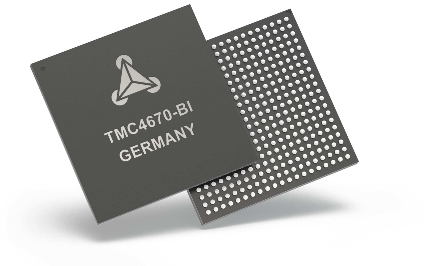

Chipsmall
深圳市津利帝科技有限公司
Geelead Technology Limited
Chipsmall
采购部
情暖大凉山
公益助学行动

TMC4670 Datasheet

IC Version V1.01 | Document Revision V1.00 • 2018-Mar-08

The TMC4670 is servo controller, providing Field Oriented Control for BLDC/PMSM and 2-phase Stepper Motors. Main control functions as torque, velocity and position control are implemented in hardware. Integrated ADCs, position sensor interfaces, position interpolators, enables a fully functional servo controller.



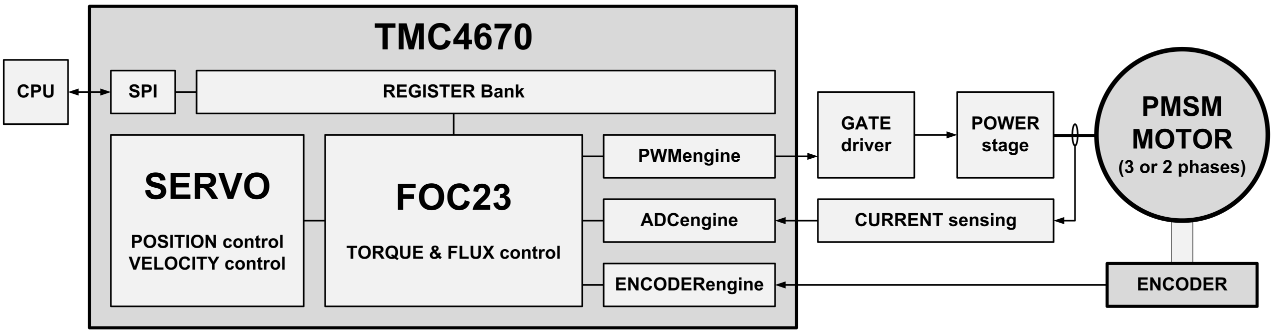
Features

- Field Oriented Control (FOC) w/ Servo Controller
- Torque Control (FOC), Velocity Control, Position Control
- Feed Forward Offsets
- Integraded ADCs
- Encoder Engine: Hall analog/digital, Encoder analog/digital, 2nd digital Encoder
- Supports 3-Phase PMSM (FOC3) and 2-Phase Stepper Motors (FOC2)
- PWM Engine including SVPWM
- SPI Communication Interface

Applications

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

Simplified Block Diagram



Read entire documentation.

Contents

1 Order Codes

Order Code	Description	Size [mm]
TMC4670-BI	TMC4670 FOC Servo Controller IC	17 x 17 x 2.5
TMC4670-EVAL	TMC4670 Evaluation Board	85 x 79

Table 1: Order codes

2 Functional Summary

- **Field Oriented Control (FOC) Servo Controller**
  - torque (and flux) control mode
  - velocity control mode
  - position control mode
  - update rate 200 kHz (w/ 4 kHz velocity meter sampling frequency)
- **PI Controllers**
  - programmable clipping of inputs and outputs of interim results
  - error sum (error integral over time) clipping
  - programmable target torque change in time (dTargetTorque/dt) limiter
  - programmable circular ($\sqrt{U_D^2 + U_Q^2}$) limiter
  - PI controller clipping status bit vector for real time Monitoring
  - Feed Forward Offsets for Target Values
- **Supported Motor Types**
  - FOC3 : three-phase permanent magnet synchronous motors (PMSM)
  - FOC2 : two-phase stepper motors
- **ADC Engine with Offset Correction and Scaling**
  - integrated 12 bit ADCs as analog interface for currents and analog encoders
  - interface to external AD (LTC2351, 14 bit or 12 bit)
  - ADC register to write externally sampled data via SPI communication interface
- **Encoder Engine**
  - open loop position generator (programmable [rpm], [rpm/s]) for initial setup
  - digital incremental encoder (ABN resp. ABZ, up to 5MHz)
  - secondary digital incremental encoder
  - digital hall sensor interface (H$_1$, H$_2$, H$_3$ resp. H$_U$, H$_V$, H$_W$)
  - digital hall sensor interface with interpolation of interim positions
  - analog encoder/analog hall sensor interface (SinCos (0°, 90°) or 0°, 120°, 240°)
  - position multi-turn counter (32 Bit)
- **PWM Engine including SVPWM**
  - programmable PWM frequency within range 25kHz . . . 200kHz
  - programmable Brake-Before-Make (BBM) times (high side, low side) 0ns . . . 2.5μs in 10ns steps
  - 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)
- **Supply Voltages** 3.3V, 2.5V, 1.2V
- **IO Voltage** 3.3V
- **Clock Frequency** 25MHz
- **Package** 17mm x 17mm BGA w/ 1mm ball pitch

3 Functional Description

TMC4670 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). It contains the complete control loop core architecture (position, velocity, torque) as well as required peripheral interfaces for communication with an application controller, for feedback (digital encoder, analog interpolator encoder, digital Hall, decoder Hall position interpolator, analog inputs for current and motor supply voltage measurement), and helpful additional IO. It supports highest control loop speed and PWM frequencies.

The TMC4670 is the building block for the user application that 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 sub-tasks as current measurement, real time sensor signal processing, real time PWM signal generation from the user application layer as outlined by figure 17.

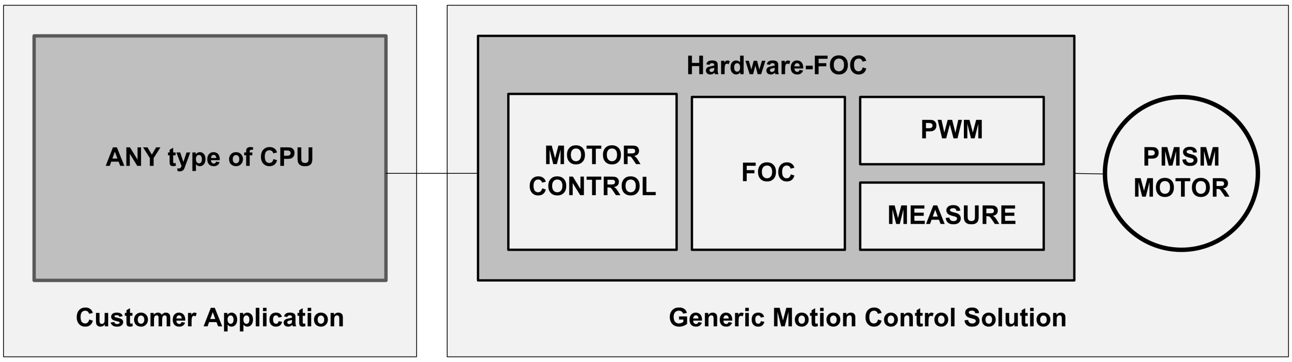


Figure 1: Hardware FOC Application Diagram

3.1 Functional Blocks

Application interface, register bank, ADC engine, encoder engine, FOC torque PI controller, velocity PI controller, position P controller, and PWM engine form the TMC4670. The TMC4670 supports 3-phase PMSM motors (FOC3) and 2-phase PMSM stepper motors (FOC2).

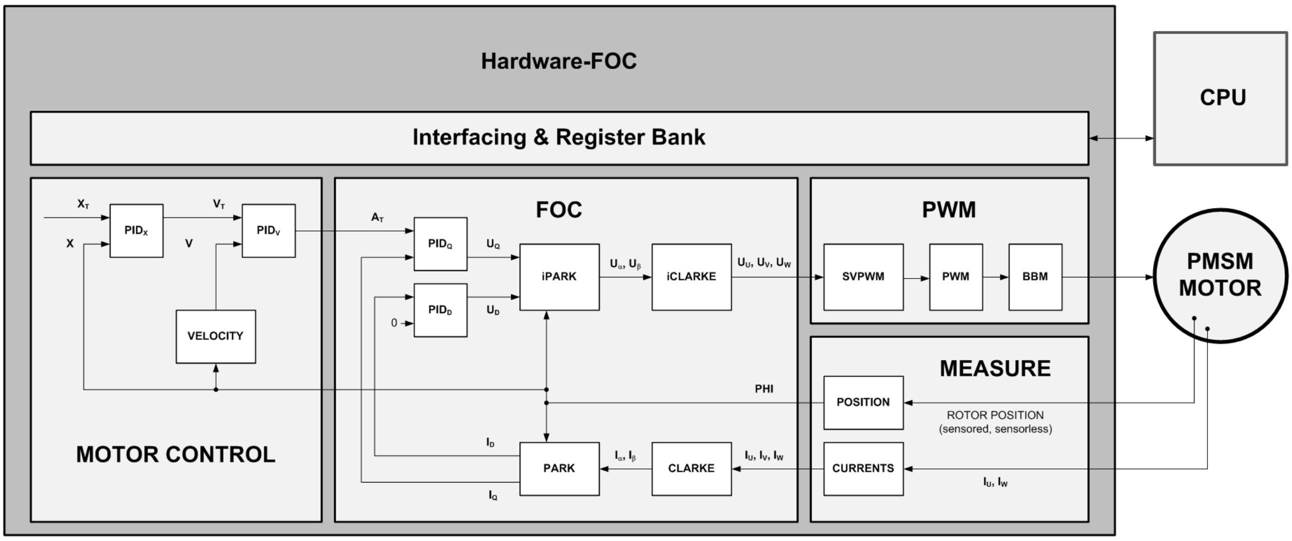


Figure 2: Hardware FOC Block Diagram

The ADC engine interfaces 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.

3.2 Communication Interface

The TMC4670 is equipped with an SPI interface for access to all registers of the TMC4670.

3.2.1 SPI Slave User Interface

The SPI of the TMC4670 for the user application has an easy command and control structure. The TMC4670 user SPI acts as a slave. The SPI datagram length is 40 bit with up to 2Mbit/s. 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 (up to) 32 data bits. The SPI of the TMC4670 immediately responses within the actual SPI datagram on read and write for ease-of-use communication.

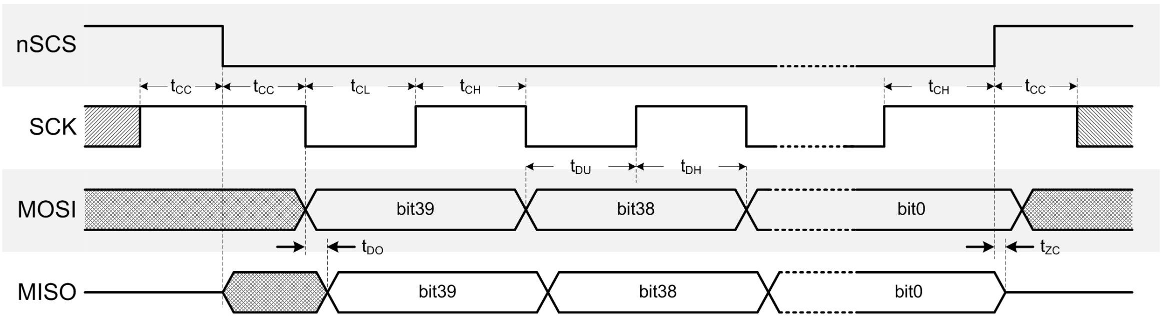


Figure 3: SPI Timing

SPI Interface Timing	**Characteristics, fCLK = 25MHz**					
Parameter	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Unit**
SCK valid before or after change of nSCS	t_{CC}		250			ns
nSCS high time	t_{CSH}		250			ns
nSCS low time	t_{CSL}		250			ns
SCK high time	t_{CH}		250			ns
SCK low time	t_{CL}		250			ns
SCK low time	t_{CL}		250			ns
SCK frequency	f_{SCK}				2	MHz
MOSI setup time before rising edge of SCK	t_{DU}		250			ns
MOSI hold time after falling edge of SCK	t_{DH}		250			ns
MISO data valid time after falling edge of SCK	t_{DO}				10	ns

Table 2: SPI Timing Parameter

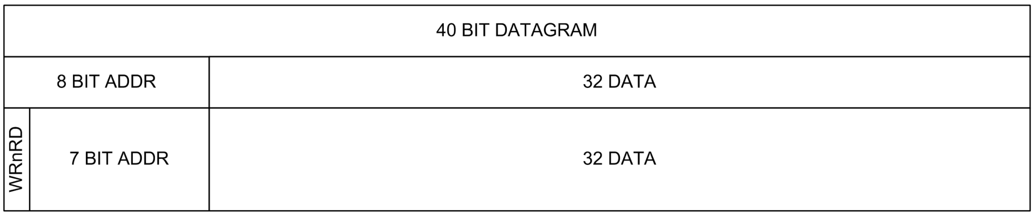


Figure 4: SPI Datagram Structure

3.3 Register Bank

Info This section gives a functional description as an overview. The section 5 starting page 27 gives the detailed description of each register.

The register bank is the interface to the user application. Each register of the TMC4670 has 8 bit address followed by up to 32 data bits. Some addresses hold more then one data registers for simultaneous access or composed control bit vectors. Section 5 page 27 describes all registers in detail.

During initialization, the user writes parameters into associated registers. These parameters are scaling factors and offsets, sensor configuration parameters, limits for clipping, selections, P and I parameters for the FOC torque controller, P and I parameters for velocity controller, and P parameter for the position controller.

During application, the user writes application parameters into associated registers. These are - depending on the motion mode - target torque, target velocity, or target position.

The TMC4670 has direct access registers and indirect access registers. Most registers are direct access registers with read or write access by a single datagram. Some less often used registers (e.g. silicon version registers, internal values for read out) are accessed via two registers: address register and data register. The address register selects the address, the second register holds the data.

3.3.1 Register Bank - Read and Write

From the access point of view there are two kind of registers: read-only and read-write. The most significant bit (MSB) of each register access datagram defines read (=0) or write (=1). So, there are 128 read addresses ($0x00_h \ldots 0x7F_h$) and 128 write addresses ($0x80_h \ldots 0xFF_h$) available. The TMC4670 ignores write accesses to read-only registers.

- Fixed Read Only Register (e.g. SILICON_TYPE, SILICON_VERSION, SILICON_DATE, SILICON_TIME)
- Read Only Register for internal values (e.q. scaled ADC values)
- Read Only Register for external Signals (e.g. ADC raw values, ABN encoder inputs, Hall signal inputs)
- Read Write Register for configurations (e.g. P and I parameter of PI controller, clipping parameters)
- Read Write Register for target values (e.g. target torque, target velocity, target position)
- Dual Ported Read Write Register (e.g. encoder count, actual position)

3.3.2 Register Access Datagram Examples

$0x0100000000_h$: reads data from address $0x01_h$
$0x8123456789_h$: writes data $0x23456789_h$ to address $0x01_h$

3.3.3 Identification of Silicon via Type, Version, Date, and Time

The read-only registers of silicon type, version with date and time identify the type of the silicon, the version of the silicon together with unique date stamp and time stamp. This enables the automatic identification of IC and version and enable the automatic handling of different IC and different versions.

3.3.4 Read of RAW Inputs & RAW Outputs

For ease-of-use while setting up the configuration, raw input signals and raw output signals are mapped into the register bank for user read out. With this, the user can initially check without a scope that the desired signals come into the TMC4670 as expected. Examples of readable raw input signals are digital Hall signals and incremental encoder signals.

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

The TMC4670 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.

3.4.1 Numerical Representation

The TMC4670 uses singed and unsigned values of different length and fixed point representation 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 interger part and 8 bit fractional part	-32767/256	32767/256
q4.12	signed fix point value with 4 bit interger 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 un-wanted 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 interger 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 under-flow or over-flow. 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
0x0001$_h$	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
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 but 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.

3.4.2 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 5).
A motor with one (1) pole pair turns once for each electrical period. A motor with two (2) pole pairs turns once for each two electrical periods. A motor with three (3) pole pairs turns once for each three electrical periods. A motor with four pole (4) pairs turns once for each 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.

$$\text{PHI\_E} = \text{PHI\_M} \cdot \text{N\_POLE\_PAIRS} \tag{1}$$

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.

$$\text{PHI\_M} = \text{PHI\_E}/\text{N\_POLE\_PAIRS} \tag{2}$$

Different encoders give different kind of position angles. Analog 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.

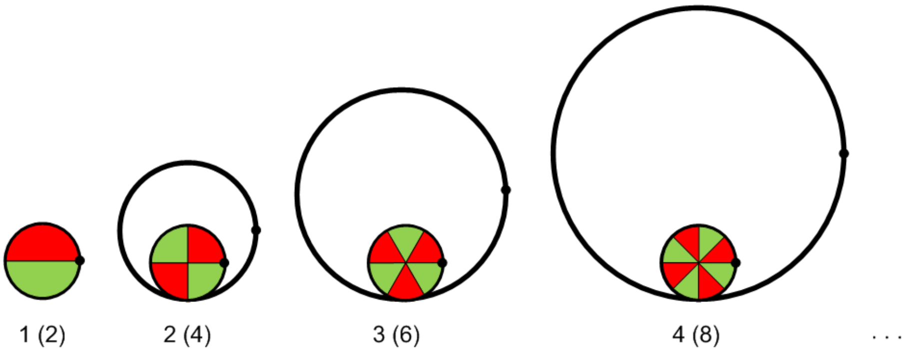


Figure 5: N_POLE_PAIRS - Number of Pole Pairs

3.4.3 Numerical Representation of Angles PHI

Electrical angels 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 interger 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.

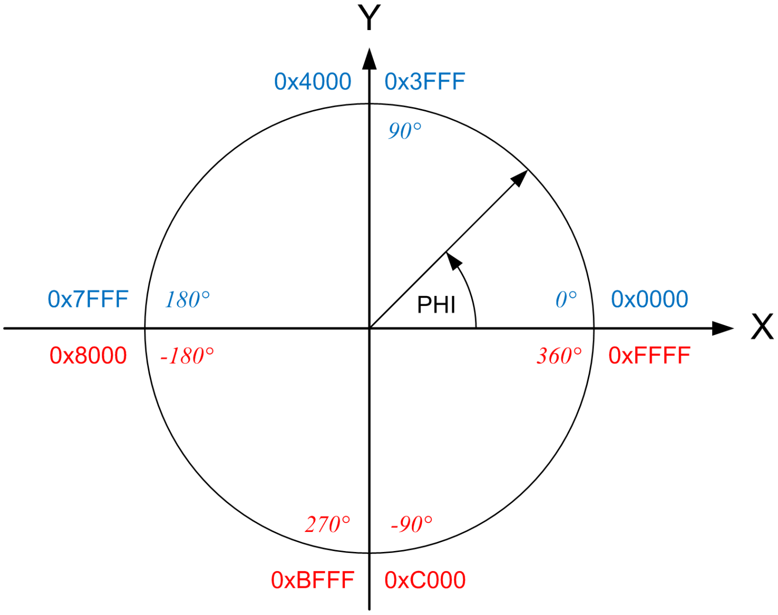


Figure 6: Integer Representation of Angles with 16 Bit as s16 resp. u16

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
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 motor and stator and rotor and encoder. Finally, the relative orientations between motor and stator and 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.

3.5 ADC Engine

The ADC engine controls the sampling of different available ADC channels.
The FOC engine expects offset corrected ADC values, scaled into the FOC engine 16 bit (s16) fixed point representation. The integrated scaler and offseter maps raw ADC samples of current measurement channels to 16 bit two's complement values (s16). Both, offset and scale calculations are signed. With this, the user can change the signs of current according to the application by the scaling factors.
The s16 scaled ADC values are available for read out from the register by the user. ADC samples for motor supply voltage (VM), MOSFET temperature, motor temperature, general purpose analog input (AIN) are only raw values without scaling.

- ADC samples of integrated ADC
- ADC samples from external ADC (LTC2351)
- ADC samples from external sources can be written into dedicated registers (ADC EXT)
- ADC values are for:
  - phase current measurement (most important task)
  - Supply voltage measurement (for monitoring or brake chopper)
  - Analog Hall signal measurement
  - analog Sine/Cosine encoder signal measurement
  - analog voltage input for MOS-FET temperature signal monitoring

  - analog voltage input for motor temperature signal monitoring

---

ℹ *Info*	Wrong scaling factors or wrong offsets might cause damages when the closed current regulation is active. Integrated hardware limiters allow protection especially in the setup phase when using carful limits.

3.5.1 Internal ADC

The TMC4670 is equipped with internal ADCs with input voltage range of 0V . . . 2.5V for current measurement, supply voltage measurement, analog hall signal measurement, analog encoder.

3.5.2 External ADC (LTC2351)

Alternatively to the integrated ADCs, the TMC4670 supports external SPI ADCs LTC2351 from Linear Technology for current measurements. This is intended for current sensing on separate power stages.

3.5.2.1 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.

3.5.2.2 ADC EXT

The user can write ADC values into the ADC EXT registers of the register bank from external ADC sources. For example it there are high precision ADC values availbale from an external ADC.

3.5.3 ADC Selector & ADC Scaler w/ Offset Correction

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

Note	The Open Loop Encoder is useful for setup of ADC channel selection, scaling, and offset by turning a motor open loop.

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

ℹ *Info*	Raw ADC values generally are of 16 bit width independent of their real resolution.

Info The job of the ADC scaler is to map raw ADC values to the 16 bit signed (s16) range and to center the values to zero by removing of offsets.

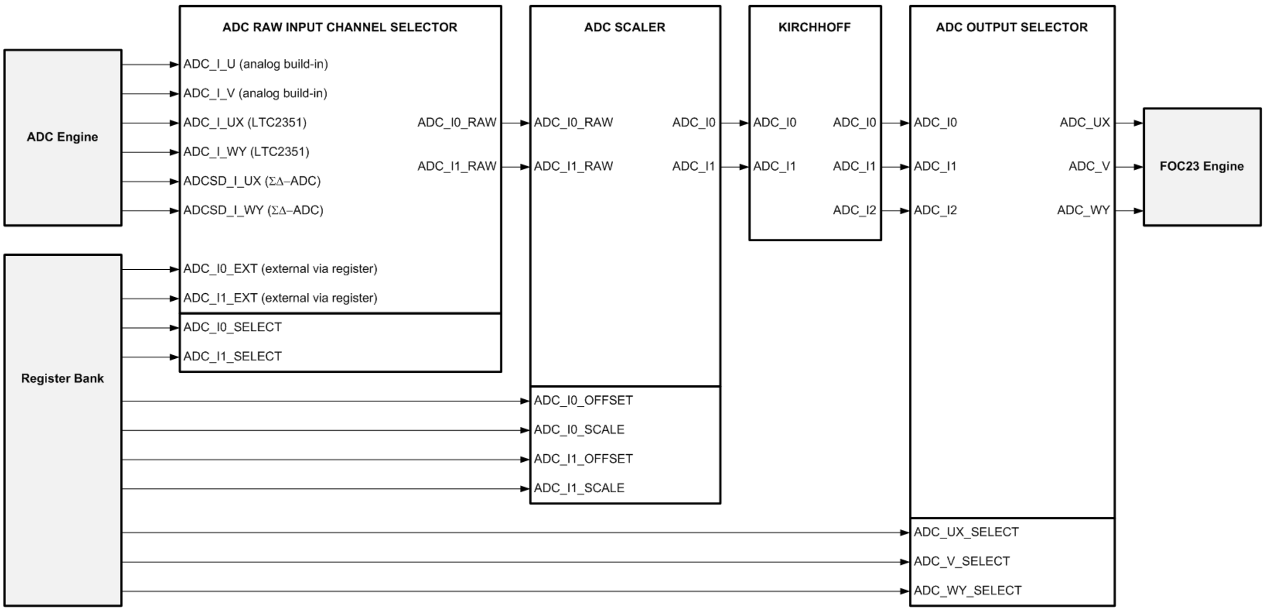


Figure 7: ADC Selector & Scaler w/ Offset Correction

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, 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.

$$\mathrm{ADC\_I0} = (\mathrm{ADC\_I0\_RAW} + \mathrm{ADC\_I0\_OFFSET}) \cdot \mathrm{ADC\_I0\_SCALE} \tag{3}$$

$$\mathrm{ADC\_I1} = (\mathrm{ADC\_I1\_RAW} + \mathrm{ADC\_I1\_OFFSET}) \cdot \mathrm{ADC\_I1\_SCALE} \tag{4}$$

For FOC3 the third current ADC_I2 is calculated via Kirchhoff's Law. This requires the correct scaling and offset correction before. 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, ADC_I2 for ADC_UX.

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

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

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

The ADC_UX, ADC_V, 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 for setup of ADC channel selection, scaling, and offset by turning a motor open loop.

3.6 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_M). Both are 16 bit values. The encoder engine maps single turn positions from position sensors to multi-turn position. 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 high precision position it might be useful to do positioning based on the electrical angel PHI_E.

3.6.1 Open Loop Encoder

For initial system setup the encoder engine is equipped with an open loop position generator. With one can turn the motor open-loop by specifying speed in rpm and acceleration in rpm/s together with a voltage UD_EXT in D direction. So, the open-loop encoder it is not a real encoder, it just gives positions as an encoder does. The open-loop decoder has a direction bit to define once the 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].

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

3.6.2 Incremental ABN Encoder

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 defines 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 are 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 on AB quadrature, the N pulse must be used. For those encoders with N pulse over four AB quadratures the user can enhance the precision of the N pulse position detection by using NAB instead of N, which is recommended.

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

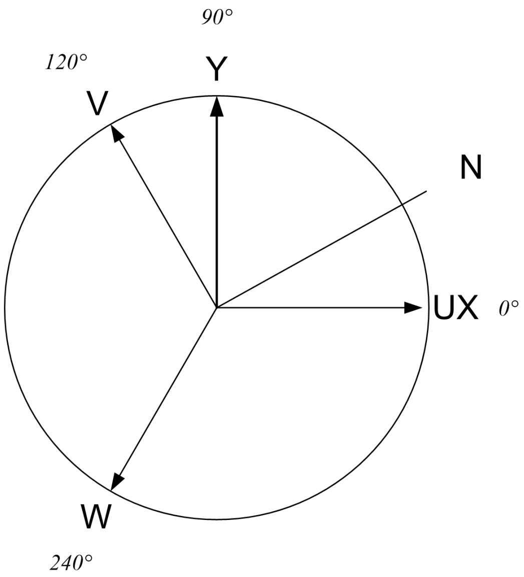


Figure 8: ABN Incremental Encoder N Pulse

The polarity of N pulse, A pulse and B pulse are programmable. The N pulse is for re-initialization 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 once the direction of motion for the application.

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.

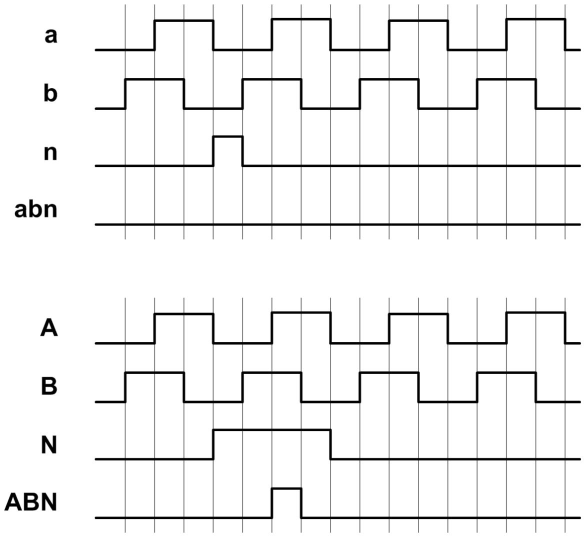


Figure 9: Encoder ABN Timing - high precise n pulse and less precise N pulse

3.6.3 Secondary Incremental ABN Encoder

For commutating a motor with FOC one 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 TMC4670 is equipped with a secondary incremental encoders interface. This secondary encoder interface is available as source for velocity control or position control. This is for applications where a motor turns an object with a gear to position the object. An example is a robot arm where a motor moves an angle with a the mechanical angle of the arm as the target.

ℹ ***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.

3.6.4 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.

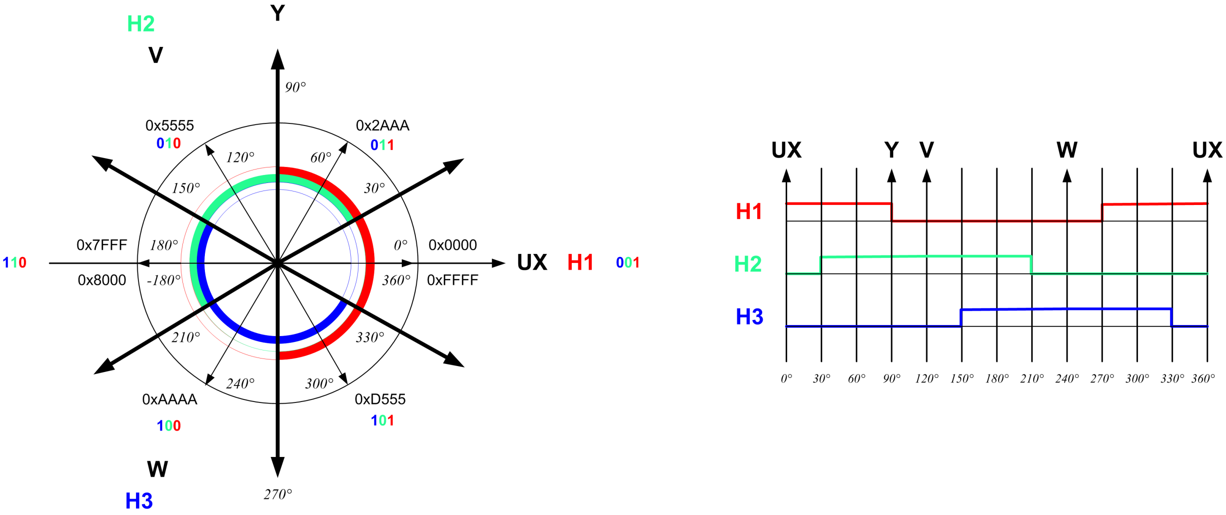


Figure 10: Hall Sensor Angles

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 TMC4670 additionally generates high resolution interim positions, when the motor is running at speed beyond 60 rpm.

3.6.5 Digital Hall Sensor - Interim Position Interpolation

For lower torque ripple the user can enable the position interpolation of interim hall positions. This function is useful for motors, which are compatible with sine wave commutation, but are equipped with digital hall sensors.
When the position interpolation is switched on, it becomes active on speed beyond 60 rpm. For lower speed it is automatically disabled. This is important especially when the motor has to be at rest.
Motors that are intended for block commutation might smarter turn with hall signal interpolation but the user should not expect too much for those motors.

3.6.6 Digital Hall 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 and later the incremental encoder can be used a a high resolution sensor for commutation.

3.6.7 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 0° 120° 240°. The analog decoder engine adds offset, scales the raw analog encoder signals and calculates the electrical angle PHI_E from these analog position signals.

ADC offsets and ADC scalers need to be programmed into the associated registers to use analog Hall sensors or analog encoders. Each AENC_0_SELECT, AENC_1_SELECT, AENC_2_SELECT, and AENC_3_SELECT, selects one raw analog ADC input channel AENC out of AENC_UX_RAW, AENC_VN_RAW, AENC_WY_RAW, AENC_N_RAW, or one AENC register channel AENC_UX_EXT, AENC_VN_EXT, AENC_WY_EXT, AENC_NEXT.

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

$$\mathrm{AENC\_VALUE} = (\mathrm{AENC\_RAW} + \mathrm{AENC\_OFFSET}) \cdot \mathrm{AENC\_SCALE} \tag{5}$$

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

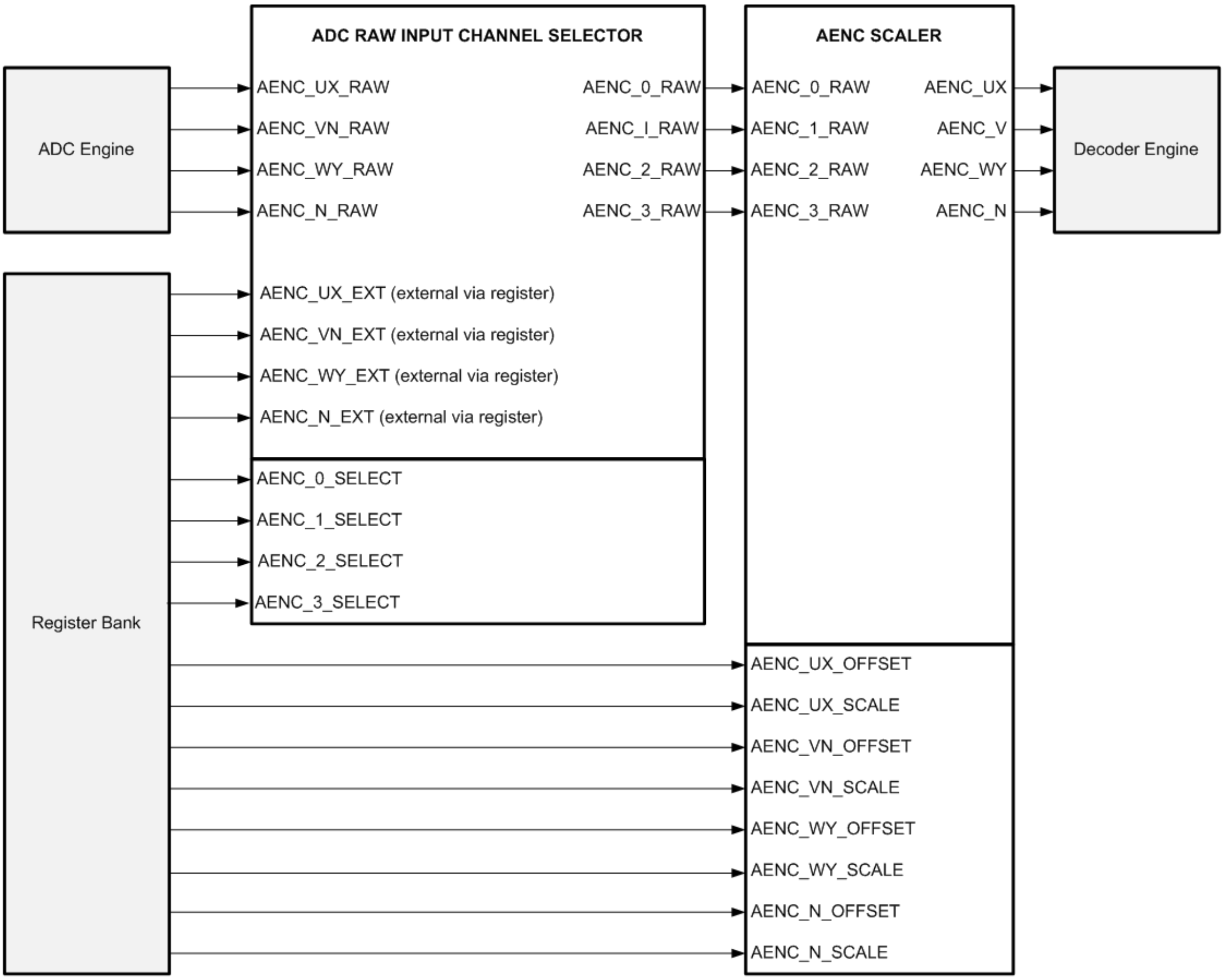


Figure 11: Analog Encoder (AENC) Selector & Scaler w/ Offset Correction

ℹ ***Info***	The analog N pulse is just a raw ADC value. Scaling, offset correction, manual handling of analog N pulse similar to N pulse handling of digital encoder N pulse is not implemented for analog encoder.

3.6.8 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.

3.6.8.1 Multi-Turn Counter

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

3.6.8.2 Encoder Engine Phi Selector

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

3.6.8.3 External Position Register

A register value written via the application interface into the register bank is available for commutation also. 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. As the application interface is not fast enough for high commutation frequencies, this mode of operation is only recommended for initialization.

3.7 FOC23 Engine

ℹ ***Info***	Support for the TMC4670 is integrated into the TMCL-IDE including wizards for system setup, which allow easy and fast commissioning and even turn the motor with a few steps. With the TMCL-IDE the user has direct access to all registers of the TMC4670.

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 control tasks.

3.7.1 PI(D) Controllers

PI controllers are used for current control and velocity control. A P controller is used for position control. The D part is not yet supported, it is just a register place holder for future variants.

3.7.2 PI(D) Controller Calculations

The PI controllers performs the calculation

$$\mathsf{dXdT} = \mathsf{P} \cdot \mathsf{e} + \mathsf{I} \cdot \int_0^t \mathsf{e(t)}\,\mathrm{d}t \tag{6}$$

with

$$\mathsf{e} = \mathsf{X\_TARGET} - \mathsf{X} \tag{7}$$

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

3.7.3 PI(D) Controller - Clipping

The limiting of target values for PI controllers and output values of PI controllers is programmable. Per power on default theses limiter are set to maximum values. Before one starts a motor one should set the limiters for clipping.
The target input is clipped to X_TARGET_LIMIT. The output of a PI(D) 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(D) is clipped to dXdT_LIMIT. The error integral of (6) is clipped to dXdT_LIMIT / I.

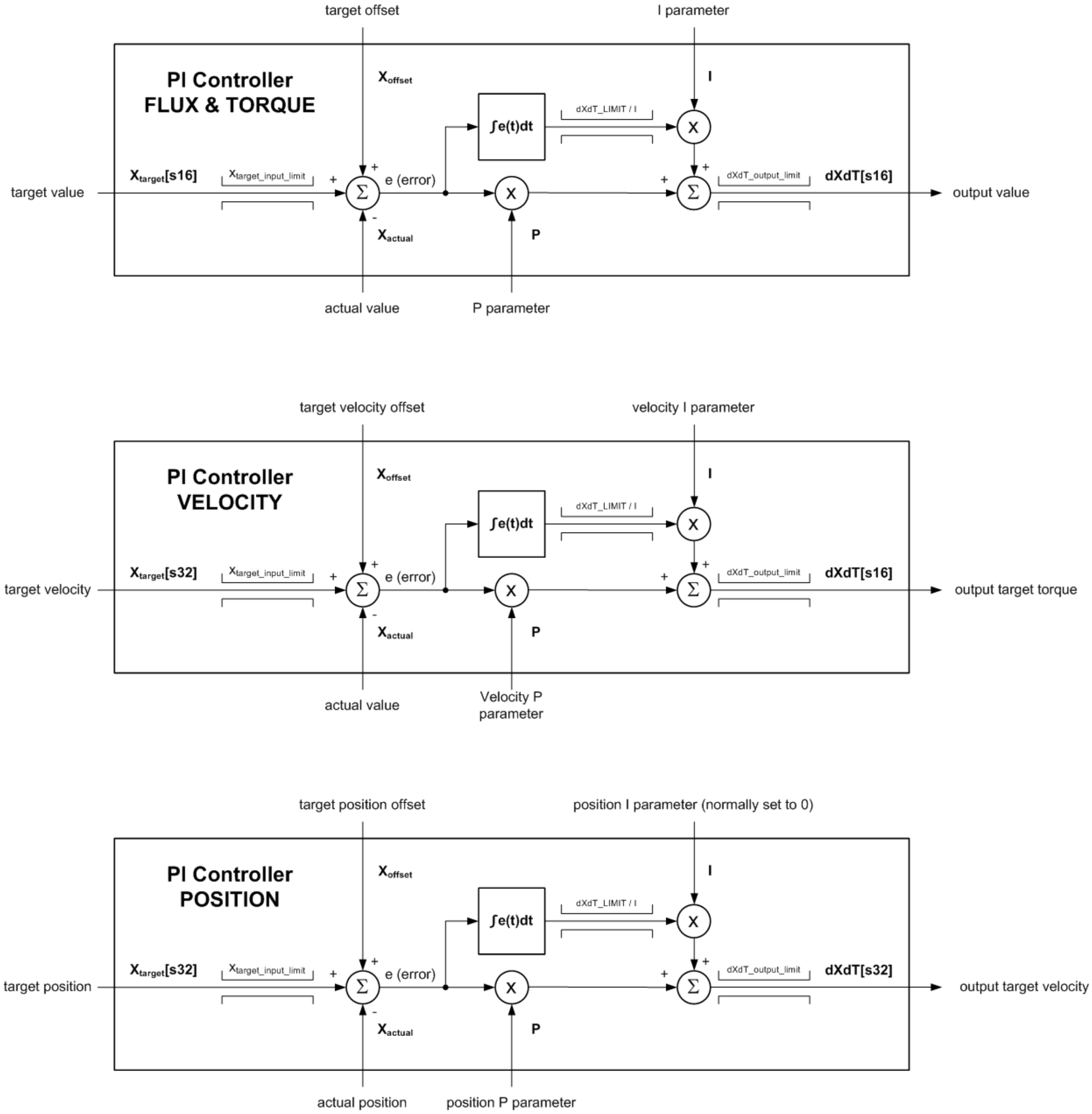


Figure 12: PID Architectures

3.7.4 PI Flux & PI Torque

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

3.7.5 PI Velocity

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

3.7.6 P(I) Position

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.

This is because e = x - x_target might result in larger e[s32] for x[s32] and x_target[s32] represented as s32 for e = x - x_target for x[s16] and x_target[s16] represented as s16.

3.7.7 Inner FOC Control Loop - Flux & Torque

The inner FOC loop (figure 13) controls the flux current to a flux target and the torque current to the desired torque target. The inner FOC loop performs the desired transformations according to figure 14 for 3-phase motors (FOC3). For 2-phase motors (FOC2) both Clarke (CLARKE) transformation and inverse Clarke (iCLARKE) a by-passed.

The inner FOC control loop gets a target torque value (I_Q_TARGET) that 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 three target voltage values as input for the PWM engine.

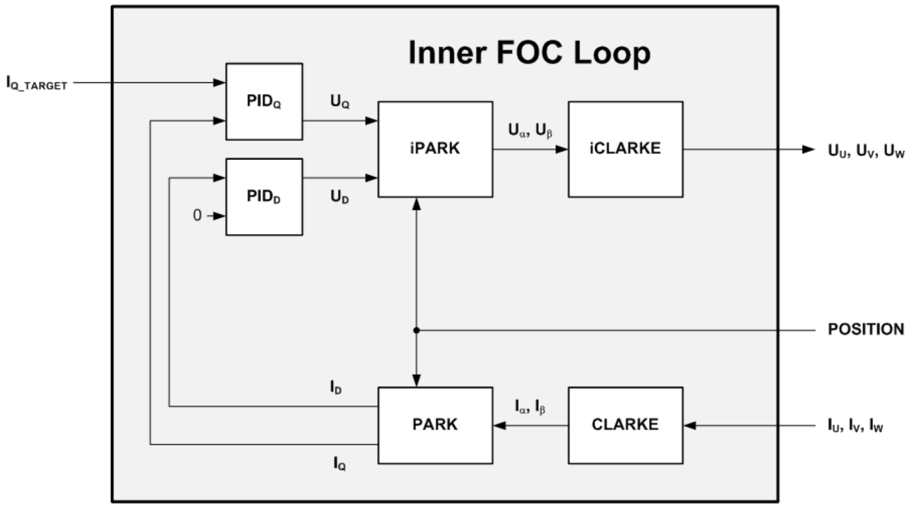


Figure 13: Inner FOC Control Loop

3.7.8 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 corresponds to the torque. The flux just pulls on the rotor and effects the torque constant. The torque is effected 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 of zero. The determined voltages (U_D, U_Q) are re-transformed into the stator system by the inverse Parke 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.

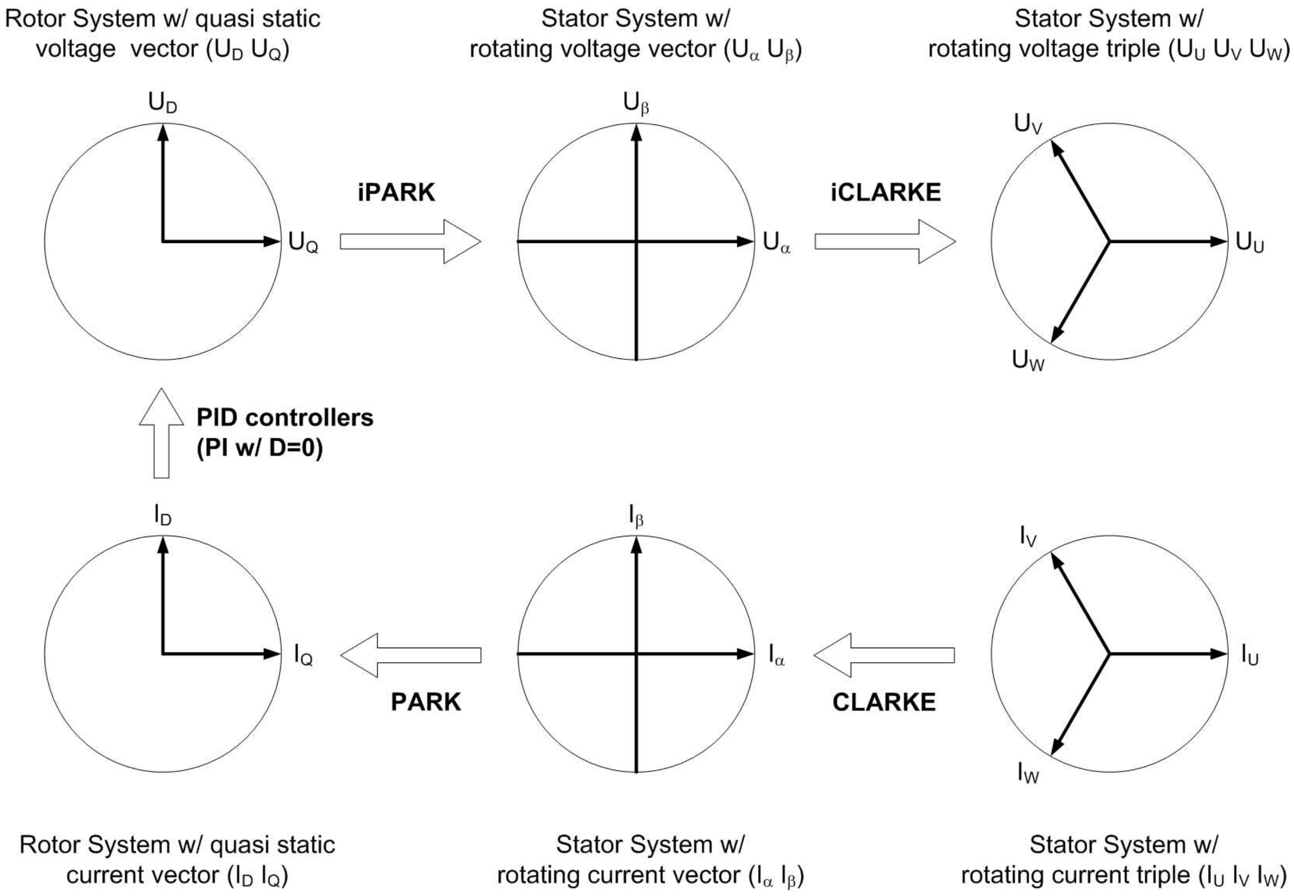


Figure 14: FOC3 Transformations (FOC2 just skips CLARKE and iCLARKE)

3.7.9 Motion Modes

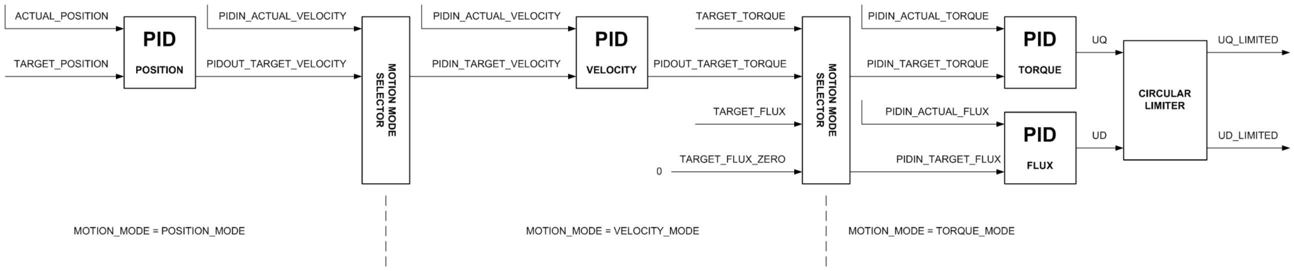


Figure 15: Motion Modes

3.8 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 detailed description of the PWM control registers and PWM register control bits pls. refer section 5 page 27.

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 that means PWM = OFF. For operation, the centered PWM mode must be set on by setting the PWM mode to 7. A single bit controls the Space vector PWM (SVPWM). 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 the motor almost at rest.

3.8.1 PWM Polarites

The PWM polarities register PWM_POLARITES 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 high side gate control and for low side gate control. The PWM polarities register controls the polarity of other control signals as well.

3.8.2 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] = is (4.0 * fCLK[Hz]) / (PWM_MAXCNT + 1). With fCLK = 25 MHz and power-on reset (POR) default of PWM_MAXCNT = 3999 the PWM frequency is fPWM = 25 kHz. The PWM frequency fPWM is recommended to be in the range of 25 kHz to 200 kHz by setting PWM_MAXCNT between 3999 to 499.

Note The PWM frequency can be changed any time also during motion.

3.8.3 PWM Resolution

The base resolution of the PWM is 12 bit internally mapped to 16 bit range. MAX_PWMCNT=4095 gives the full resolution of 12 bit with $\approx$ 25kHz w/ fCLK = 25 MHz. MAX_PWMCNT = 2047 results in 11 bit resolution but with $\approx$ 50 kHz w/ fCLK = 25 MHz. So the PWM_MAXCNT defines the PWM frequency but effects the resolution of the PWM.

3.8.4 PWM Modes

The power-on reset (POR) default of the PWM is OFF. The standard PWM scheme is the centered PWM. Passive Breaking and Free Wheeling Modes are available on demand. Please refer [**?**] concerning the settings.

3.8.5 Brake-Before-Make (BBM)

One register controls BBM time for the high side. One 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 there own integrated BBM timers.