

How to Use Analog Position Feedback with TMC4671

Field-oriented control (FOC) is a state-of-the-art control scheme to run two-phase stepper motors or three-phase permanent magnet synchronous motors (BLDC or PMSM) in closed loop. The position feedback is essential in a closed loop FOC motor drive. Using analog position feedback, the TMC4671 provides hardware closed loop torque control, velocity control, and position control decoupling motor control from application .

Contents

1	Analog Position Feedback and TMC4671	3
2	Analog Encoder with TMC4671	3
2.1	Connection	3
2.2	Analog Encoder Configuration	3
2.3	Analog Encoder Clock	4
2.4	Analog Encoder Mode	4
2.5	Encoder Resolution	4
2.5.1	Analog Encoder PPR Full Range	5
2.6	Analog Encoder ADC Offset and Scale	6
2.7	Encoder Initialization	6
2.7.1	TMC4671 Wizard	6
2.7.2	Automatic	6
2.7.3	One Time Initialization	8
2.8	Feedback Selection	8
3	How to Configure a BLDC with Analog Encoder with TMCL-IDE	8
3.1	TMC4671 Evaluation Board	8
3.2	Sense Resistors	9
3.3	Overview Wizard Configuration	10
3.4	Open TMC4671 Wizard	10
3.5	Wizard Selection	11
3.6	Settings - Powerstage and Motor	11
3.7	Open Loop	12
3.8	ADC Current Configuration	13
3.9	Analog Encoder	14
3.10	Encoder Test Drive (Torque Mode)	18
3.11	Save Configuration	19
3.12	Feedback Selection	20
3.13	Driving the Motor	20
4	Analog Hall with TMC4671	22
4.1	Connection	22
4.2	Analog Hall Configuration	22
4.3	Analog Hall Clock	22
4.4	Analog Hall Mode	23
4.5	Hall PPR	23
4.6	Analog Hall ADC Offset and Scale	23
4.7	Analog Hall Phase Offset	24
4.8	Feedback Selection	24
5	How to Configure a BLDC with Analog Hall with TMCL-IDE	24
5.1	TMC4671 Evaluation Board	24

5.2	Overview Motor Configuration	25
5.3	Open TMC4671 Wizard and Selection	25
5.4	Settings - Powerstage and Motor	25
5.5	Open Loop	26
5.6	ADC Current Configuration	27
5.7	Analog Hall	28
5.8	Analog Hall Test Drive (Torque Mode)	31
5.9	Save Configuration	31
5.10	Feedback Selection	32
5.11	Driving the Motor	33
6	How to Configure a BLDC with ABN Encoder Using an Analog Encoder	34
7	Additional Ressources	35
7.1	Automatic Analog Encoder Initialization TPC Script	35
7.2	Automatic Analog Hall Initialization TPC Script	36
7.3	ABN Incremental Encoder Initialization Using Analog Encoder TPC Script	38
7.4	Useful Links	40
8	Revision History	41

1 Analog Position Feedback and TMC4671

The TMC4671 controls a motor using the servo control modes of torque control, velocity control, and position control. To close the loop, the TMC4671 provides an ADC engine as well as multiple position feedback options that can be configured:

- Incremental encoder
- Digital hall
- Analog hall
- Analog encoder

This application note covers the configuration and operation of an analog encoder and analog hall with the TMC4671.

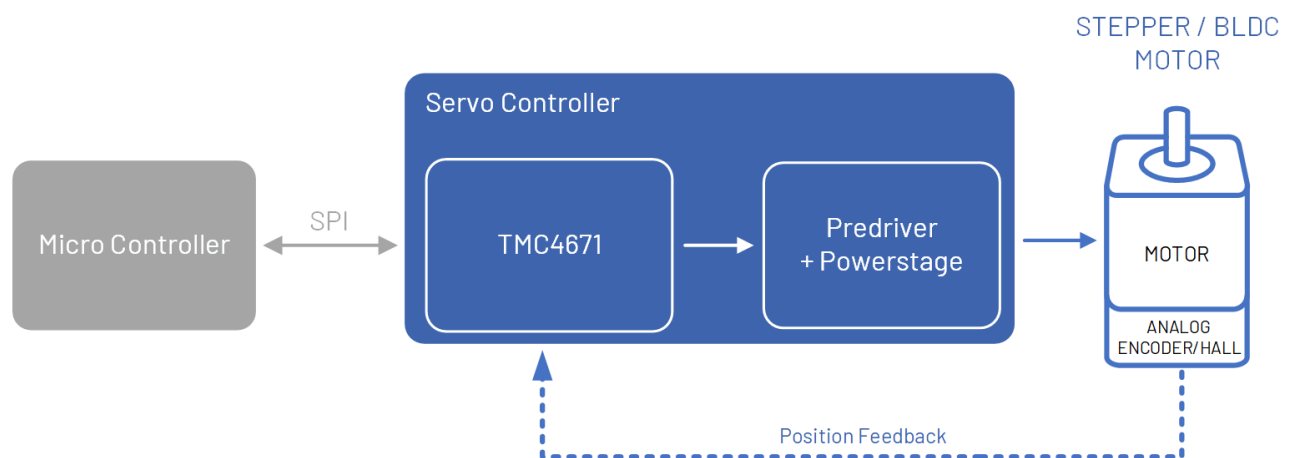


Figure 1: TMC4671 and Analog Position Feedback

2 Analog Encoder with TMC4671

In this architecture, a two-phase analog (sin/cos) encoder provides the feedback for commutation (respective to the torque loop) as well as velocity and position control of the TMC4671. Analog encoders provide high resolution and (depending on the pole pair) absolute position within a revolution, making them a good choice for high precision and lower speed application.

2.1 Connection

For two-phase analog sine wave and cosine wave signal (also called sin/cos encoder) channels, AENC_UX and AENC_WY are used. The signals can be used as differential or single-ended inputs with negative inputs (AENC_UX_NEG, AENC_WY_NEG) to GND.

NOTE

When the analog encoder is connected to AENC_VN, the wiring can be compensated in the TMC4671. An example is given below [step by step](#).

2.2 Analog Encoder Configuration

Table 1 gives an overview of the most important registers of the TMC4671 for an analog encoder:

Address	Register Name	Function
0x06 _h	dsADC_MCLK_B	Delta sigma modulator clock for group B (include analog encoder)
0x07 _h	dsADC_MDEC_B_MDEC_A	Filter for delta sigma modulators for group B (include analog encoder)
0x40 _h	AENC_DECODER_PPR	Number of periods per revolution for analog encoder
0x3B _h	AENC_DECODER_MODE	Basic configuration for the analog encoder
0x0D _h	AENC_0_SCALE_OFFSET	Calibration data for ADC channel AENC 0
0x0E _h	AENC_1_SCALE_OFFSET	Calibration data for ADC channel AENC 1
0x0F _h	AENC_2_SCALE_OFFSET	Calibration data for ADC channel AENC 2
0x11 _h	AENC_SELECT	Correct analog encoder ADC to compensate wiring twists
0x3E _h	AENC_DECODER_PHI_A_OFFSET	Offset of PHI_A for phase alignment
0x45 _h	AENC_DECODER_PHI_E_PHI_M_OFFSET	Offsets for electrical and mechanical angle for the analog encoder
0x50 _h	VELOCITY_SELECTION	Selection of velocity signal source
0x51 _h	POSITION_SELECTION	Selection of position signal source
0x52 _h	PHI_E_SELECTION	Selection of electrical angle used for commutation by FOC

Table 1: Registers Overview Analog Encoder

The following provides a guideline for the configuration.

2.3 Analog Encoder Clock

To set up the analog encoder clock, do as follows:

Action:

Set *dsADC_MCLK_B* register 0x06 = 0d536870912 = 0x20000000 (default value)

Result:

The analog encoder clock group B is active at 25Mhz.

2.4 Analog Encoder Mode

To set up a connected analog encoder, do as follows:

Action:

Set *AENC_DECODER_MODE[0]* = 0b0 (*AENC_DECODER_MODE* register 0x3B)

Result:

Selects processing of analog position signals of 0° and 90° (sin/cos encoder).

2.5 Encoder Resolution

For correct commutation, it is important to set the analog encoder resolution (*AENC_DECODER_PPR*, 0x40), which is defined by periods per mechanical revolution.

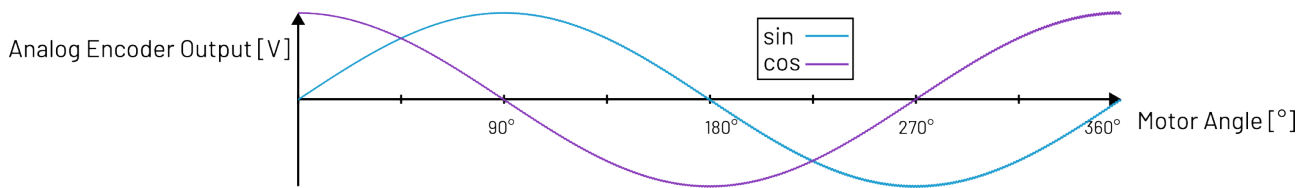


Figure 2: One Period per Revolution, PPR = 1

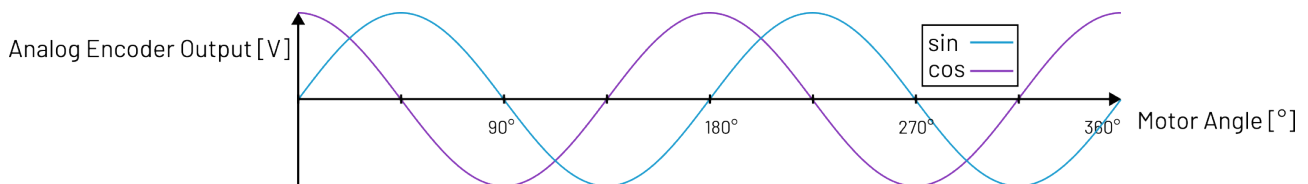


Figure 3: Two Periods per Revolution, PPR = 2

Refer to the manufacturer data sheet to determine the PPR.

NOTE

The used PPR in 0x40 AENC_DECODER_PPR must be 1 or a power of 2 (=1,2,4,8,...). When using different PPR AENC_DECODER_COUNT, must stay within $[-2^{31}; -2^{16}]$ and $[2^{16}; 2^{31}]$. For this case, 0x41 AENC_DECODER_COUNT must be overwritten during encoder initialization.

2.5.1 Analog Encoder PPR Full Range

Analog encoder PPR values that are not a power of 2 can be used only within a limited position range as follows:

1. Write AENC_DECODER_COUNT 0x41 = 2^{31} (or other value within $[-2^{31}; -2^{16}]$ and $[2^{16}; 2^{31}]$).
2. Configure the encoder initialization as described in 2.6.
3. Make sure that the AENC_DECODER_COUNT 0x41 stays within $[-2^{31}; -2^{16}]$ and $[2^{16}; 2^{31}]$ during motor movement.

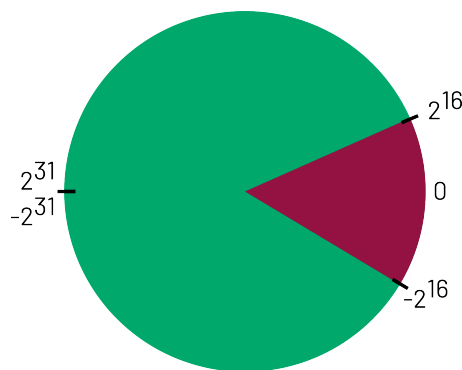


Figure 4: ABN_DECODER_COUNT, PPR Full Range

In this case, the maximum position range is $32767/PPR$ motor revolutions in each direction.

Example:

A motor has an analog encoder with 10 periods per revolution. The resulting maximum position range is $32767/10 = 3276.7$ revolutions in each direction.

2.6 Analog Encoder ADC Offset and Scale

The analog encoder offset is determined automatically using the TMCL-IDE wizard. For a code example, see the [Python code example](#).

The scale factor is set to 256 for analog encoder:

- AENC_0_SCALE in register 0x0D = 256
- AENC_1_SCALE in register 0x0E = 256
- AENC_2_SCALE in register 0x0F = 256

2.7 Encoder Initialization

The purpose of the encoder initialization is to obtain a known motor and encoder position within the motor electrical period.

2.7.1 TMC4671 Wizard

To initialize an analog encoder, the wizard in the TMCL-IDE provides inbuilt functions. The aim is to adapt AENC_DECODER_PHI_M_OFFSET so that OPENLOOP_PHI matches AENC_DECODER_PHI_E.

2.7.2 Automatic

The principle is to align the motor into a known position by applying a defined magnetic field. In this state, the analog encoder angle offset can be determined:

1. Clear analog encoder offsets
→ Set AENC_DECODER_PHI_E_PHI_M_OFFSET 0x45 = 0
2. Apply a fixed voltage at electrical position 0 .

For this initialization method, the motor should be able to move one pole pair. The resulting motor position is then well-defined as it is the beginning of the electrical period, as shown in [Figure 5](#):

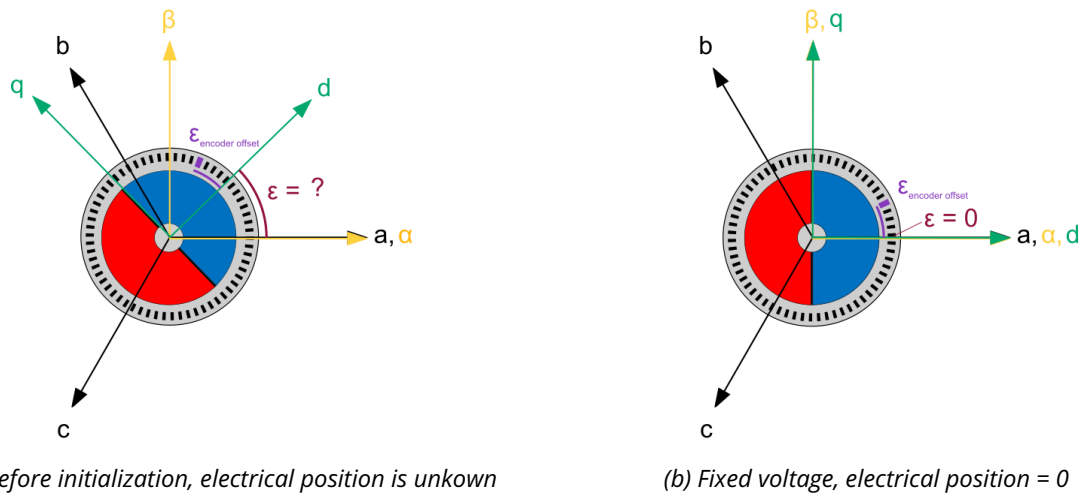


Figure 5: Analog Encoder Initialization

- Set PHI_E_SELECTION 0x52 = 1
- Set PHI_E_EXT 0x1C = 0
- Set UD_EXT in register UQ_UD_EXT 0x24

UD_EXT value depends on motor, determine suitable value for example from open loop drive.

3. Read out analog encoder value ② → Read AENC_DECODER_PHI_M from AENC_DECODER_PHI_E_PHI_M 0x46
4. Write negative analog encoder value as offset
→ Set AENC_DECODER_PHI_M_OFFSET in register AENC_DECODER_PHI_E_PHI_M_OFFSET 0x45

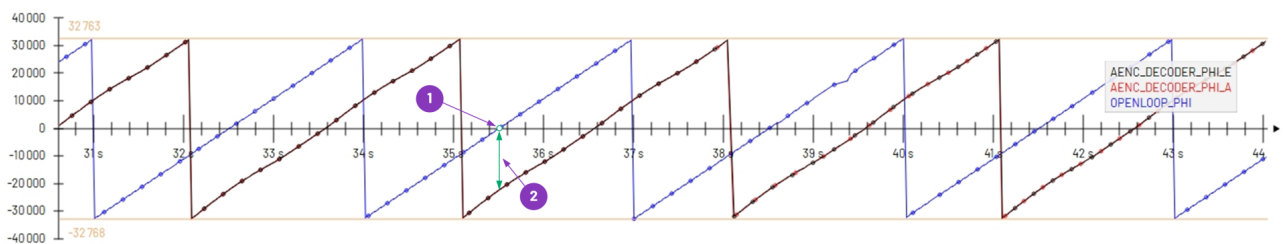


Figure 6: Encoder before Initialization

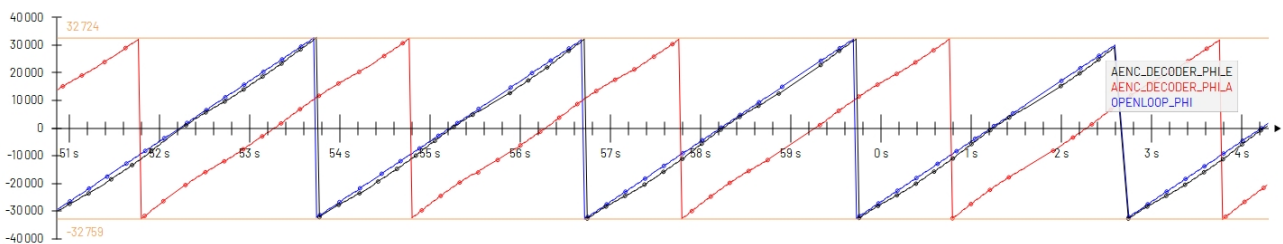


Figure 7: Encoder after Initialization

For a code example, refer to the:

- [TPC script](#).
- TMCL-IDE TMC4671 wizard summary page generates tpc and c-code examples.
- Function `tmc4671_doEncoderInitializationMode3` in the [TMC4671 API](#).

2.7.3 One Time Initialization

As the encoder offset usually changes with the motor position in general, the encoder initialization is required after every power up of the TMC4671.

However, the encoder offset is constant in the following cases:

- Analog encoder PPR = 1
- Analog encoder PPR = motor pole pair number
- Analog encoder PPR $\times$ N = motor pole pair number , N = 1,2,3,4...

In these cases, the encoder initialization can be determined once and the resulting offset is written to the TMC4671 at power up.

2.8 Feedback Selection

The feedback selection registers define the type of the position feedback (example, digital hall, analog position, incremental encoder). There are following options:

- 6: `phi_e_aenc` → electrical position (analog position within an electrical period) is used.
- 7: `phi_a_aenc` → analog position (analog value after scaling and offset) is used.
- 9: `phi_m_aenc` → mechanical position (analog position within an mechanical period) is used.

Usual recommendation is to configure the TMC4671 using the electrical position:

Action:

Address	Register Name	Value
0x50 _h	VELOCITY_SELECTION	6 (<code>phi_e_aenc</code>)
0x51 _h	POSITION_SELECTION	6 (<code>phi_e_aenc</code>)
0x52 _h	PHI_E_SELECTION	6 (<code>phi_e_aenc</code>)

Result:

Analog encoder electrical position is used as feedback for torque, velocity, and position modes.

3 How to Configure a BLDC with Analog Encoder with TMCL-IDE

3.1 TMC4671 Evaluation Board

The ADI Trinamic evaluation boards are being used as hardware platform for this guide. Following parts are used:

- [Landungsbruecke interface board](#)

- [TMC4671-EVAL](#)
- [TMC-UPS-2A24V-A-EVAL \(powerstage\)](#)
- [TMCL-IDE](#)
- BLDC motor with analog encoder [Faulhaber 2232S 024 BX4 - 5327](#)
 - Motor pole pair = 2
 - Motor maximum rated current = 0.54A
 - Analog encoder periods per revolution = 2

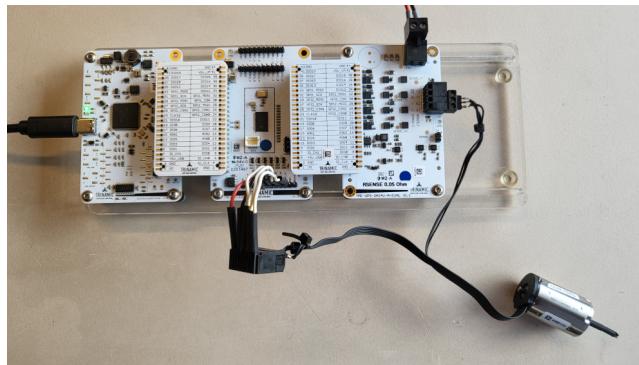


Figure 8: Hardware Setup Analog Encoder

3.2 Sense Resistors

The TMC-UPS-2A24V-A-EVAL powerstage is rated to 2A. To account for the smaller motor current, the parallel sense resistor R37 and are R35 removed.

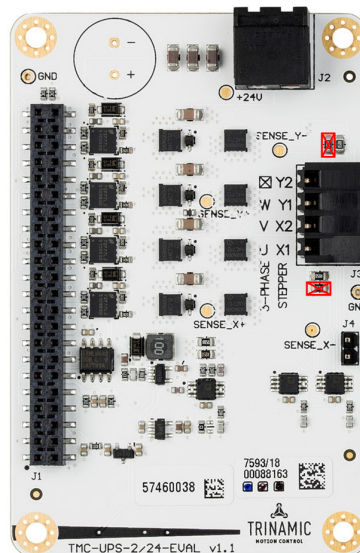


Figure 9: Remove R37 and R35 from the TMC-UPS-2A24V-A-EVAL

To calculate the resulting motor current, use the following equation ¹:

$$A_{\text{Peak}}/\text{LSB} = \frac{U_{\text{ADCRange}}}{R_{\text{Sense}} \times G_{\text{SenseAmp}} \times 32767} \quad (1)$$

$$A_{\text{Peak}}/\text{LSB} = \frac{2.5\text{V}}{0.05\Omega \times 20\text{V/V} \times 32767} = 0.076\text{mA/LSB} \quad (2)$$

The sense resistor value of 0.05Ω results in conversion factor 0.076mA/LSB . From the motor data sheet, the resulting current limit is determined and written to PID_TORQUE_FLUX_LIMITS $0x5E$:

$$\text{PID_TORQUE_FLUX_LIMITS } 0x5E = 500\text{mA} \times \frac{1}{0.076\text{mA/LSB}} = 6553\text{LSB} \quad (3)$$

3.3 Overview Wizard Configuration

Configuration by TMCL-IDE wizard

1. [Powerstage and motor](#)
2. [Motor current ADC](#)
3. [Analog encoder configuration and initialization](#)

Configuration with TMCL-IDE tools

1. [Feedback selection](#)
2. [PI tuning and position mode](#)

3.4 Open TMC4671 Wizard

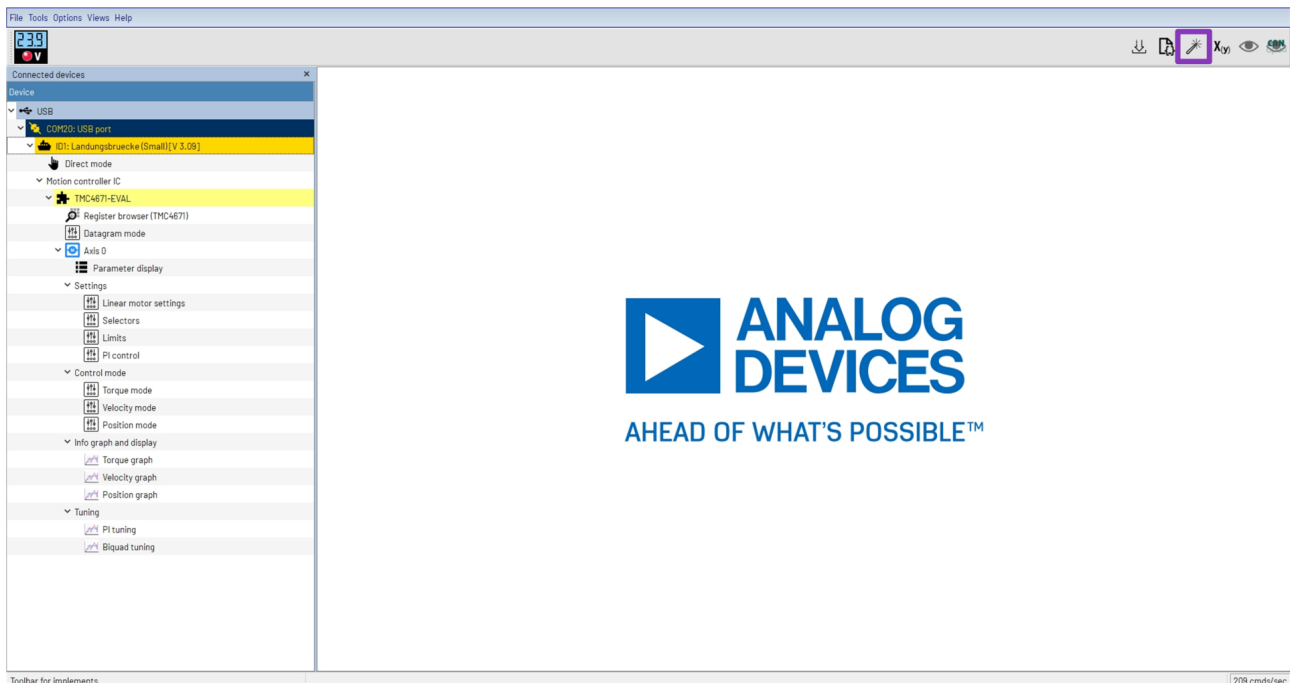


Figure 10: TMCL-IDE Wizard Location

¹ Refer to TMC4671 data sheet

3.5 Wizard Selection

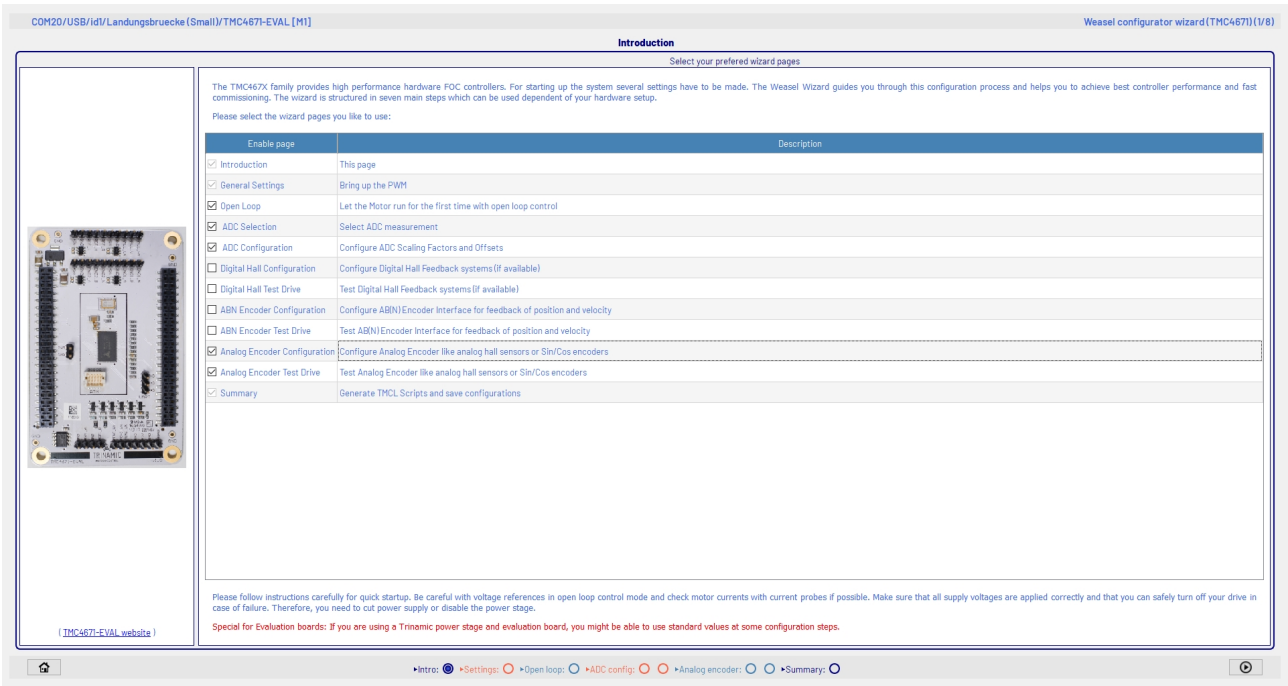


Figure 11: Selection of Function Blocks

3.6 Settings - Powerstage and Motor

Select the powerstage and set number of pole pairs = 2 for the BLDC. The pole pairs number is set according to the motor setup specified above. Adapt pole pairs to own motor (from motor data sheet).

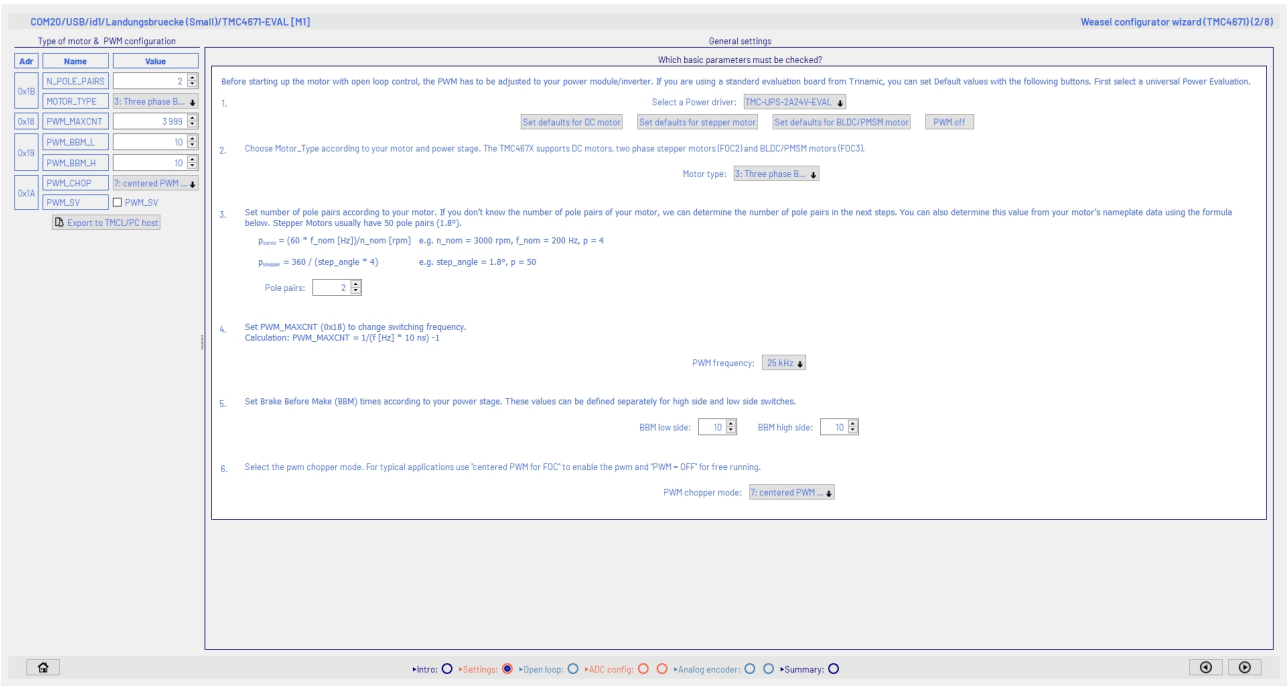


Figure 12: Powerstage and Motor Settings

3.7 Open Loop

Run the motor in open loop for ADC and encoder configuration.

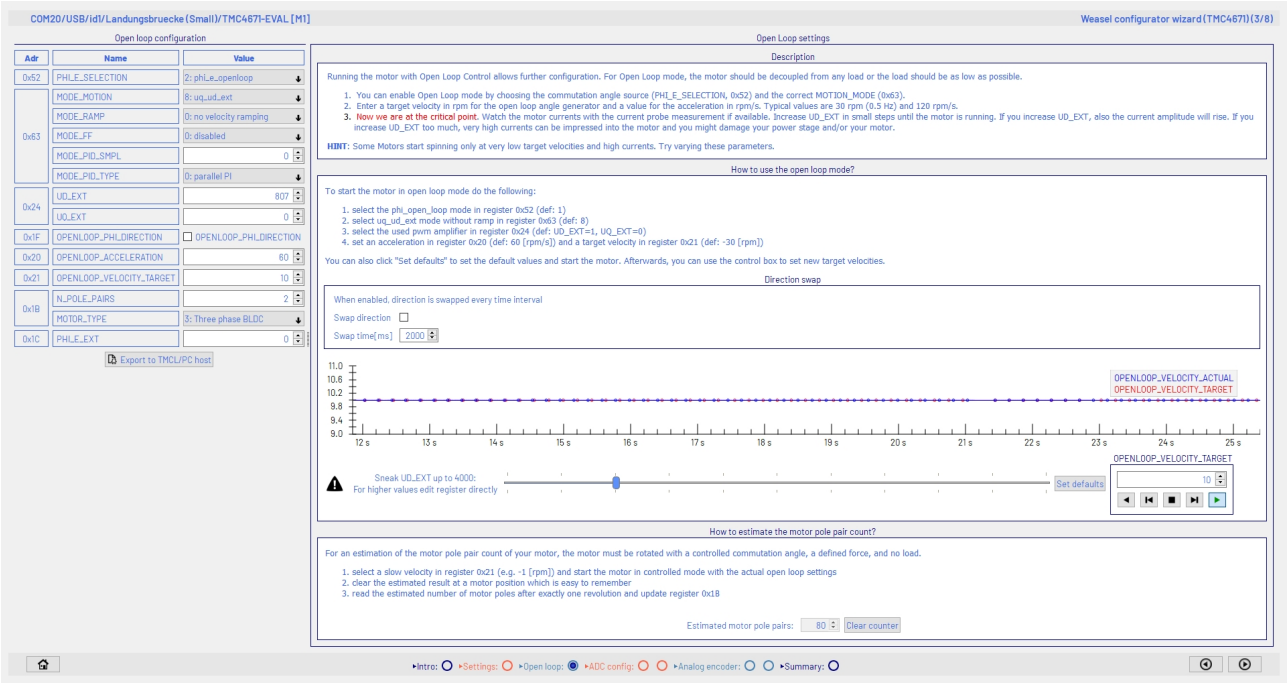


Figure 13: Open Loop Drive

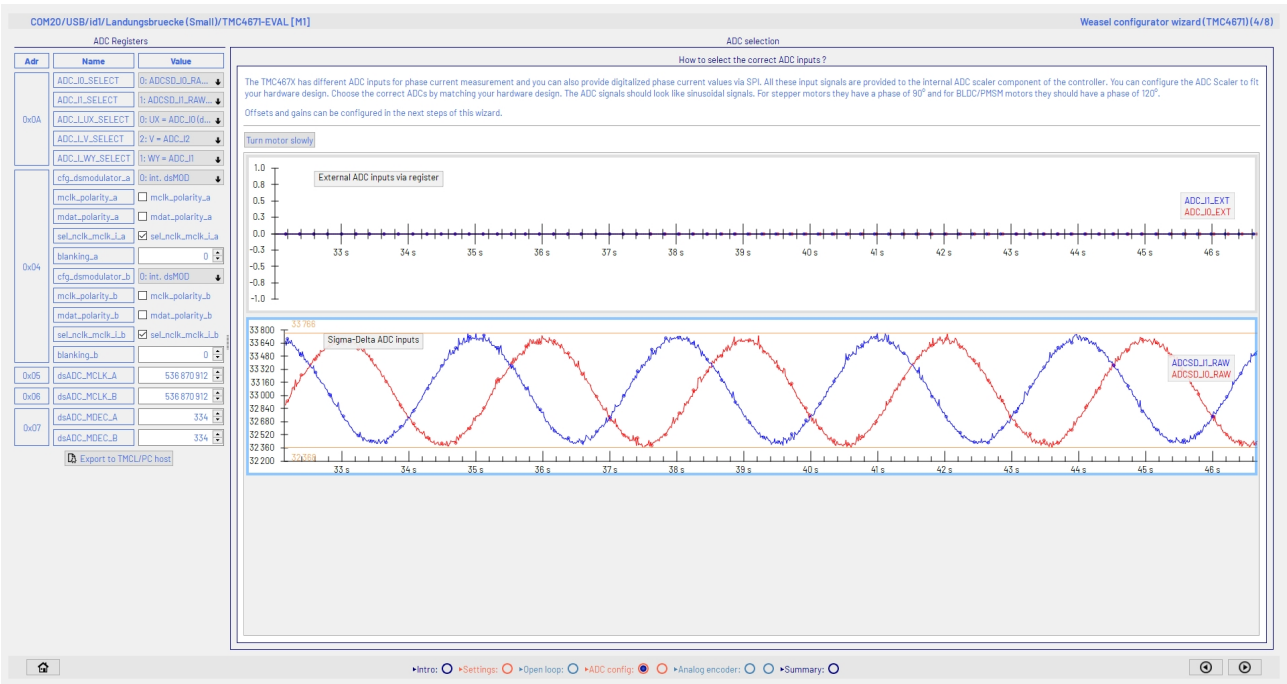


Figure 14: Open Loop RAW Motor Current

3.8 ADC Current Configuration

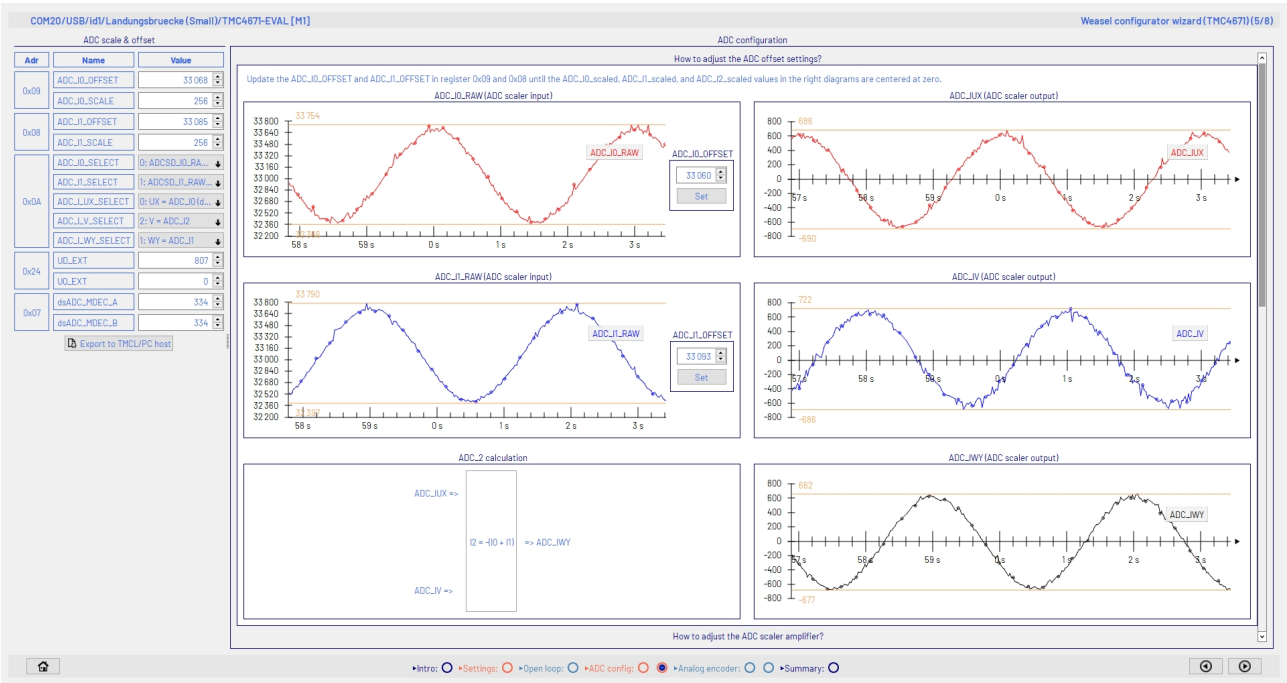


Figure 15: Motor Current Offset Configuration

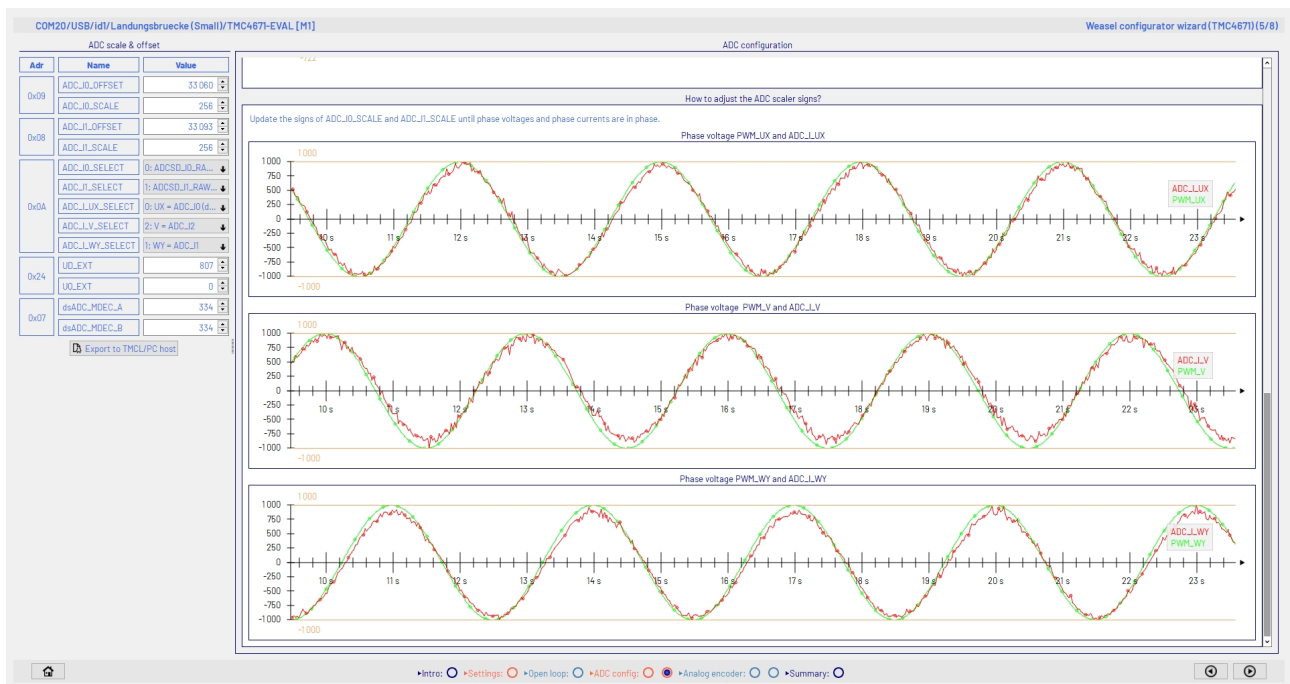


Figure 16: Motor Current Phase Selection

3.9 Analog Encoder

First the analog encoder CLK needs to be set ❶. Then AENC_DECODER_MODE (0x3B) Bit0 is set to 0 (= analog encoder) ❷. As next step, the offset of each phase is compensated using the Set button ❸.



Figure 17: Analog Encoder Configuration

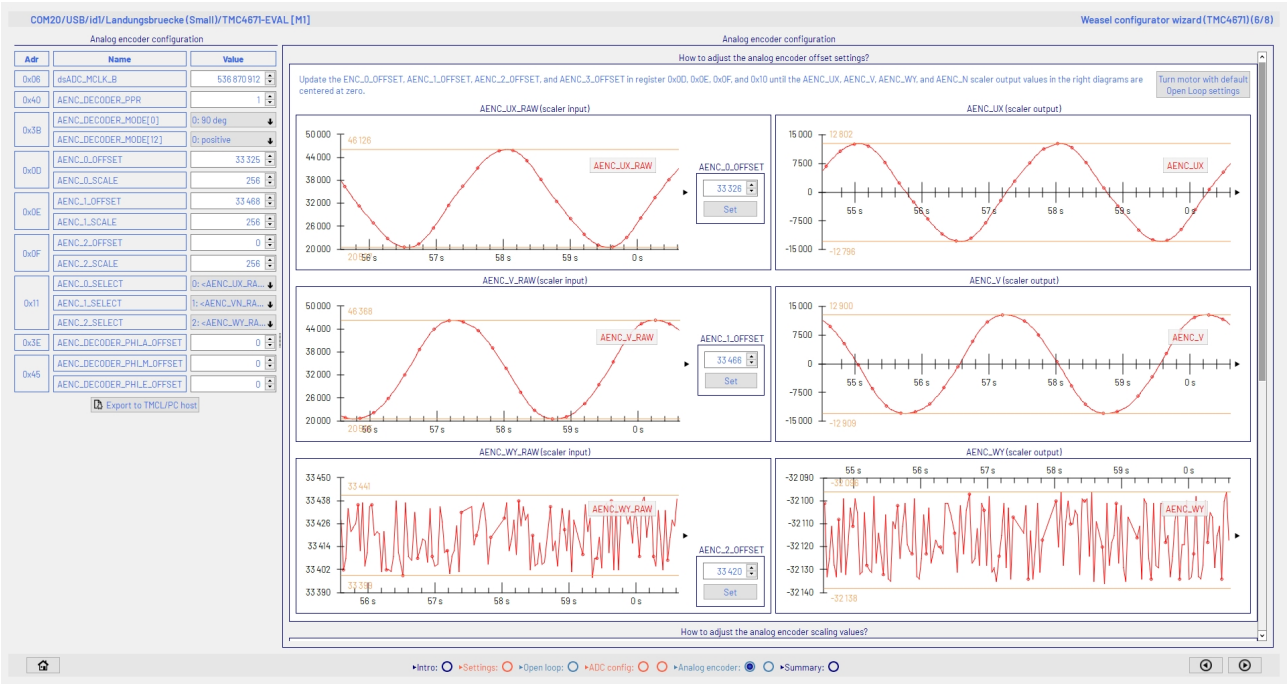


Figure 18: Analog Encoder Configuration after Offset Compensation

The used analog encoder has two periods per revolution. Thus, the PPR is set to two (ANALOG_DECODER_PPR) 1. As shown in the bottom graph, the resulting AENC_DECODER_PHI_E does not linear increase or decrease 2, which needs to be corrected.



Figure 19: Analog Encoder Example of an Unusable Signal

The analog encoder input expects the signals at the channels AENC_UX, AENC_WY. The current hardware connection (AENC_UX, AENC_VN) can be compensated as shown:

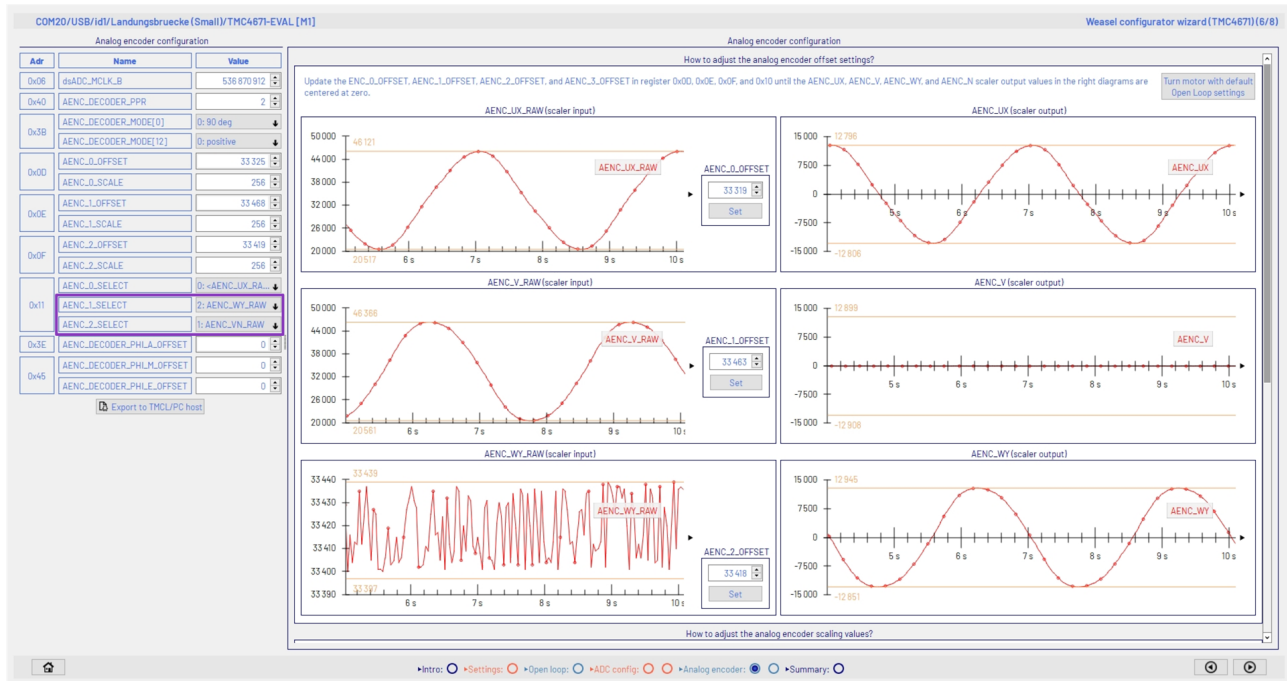


Figure 20: Analog Encoder Channel Compensation

The direction is compensated with setting bit 12 in register AENC_DECODER_MODE (0x3B):

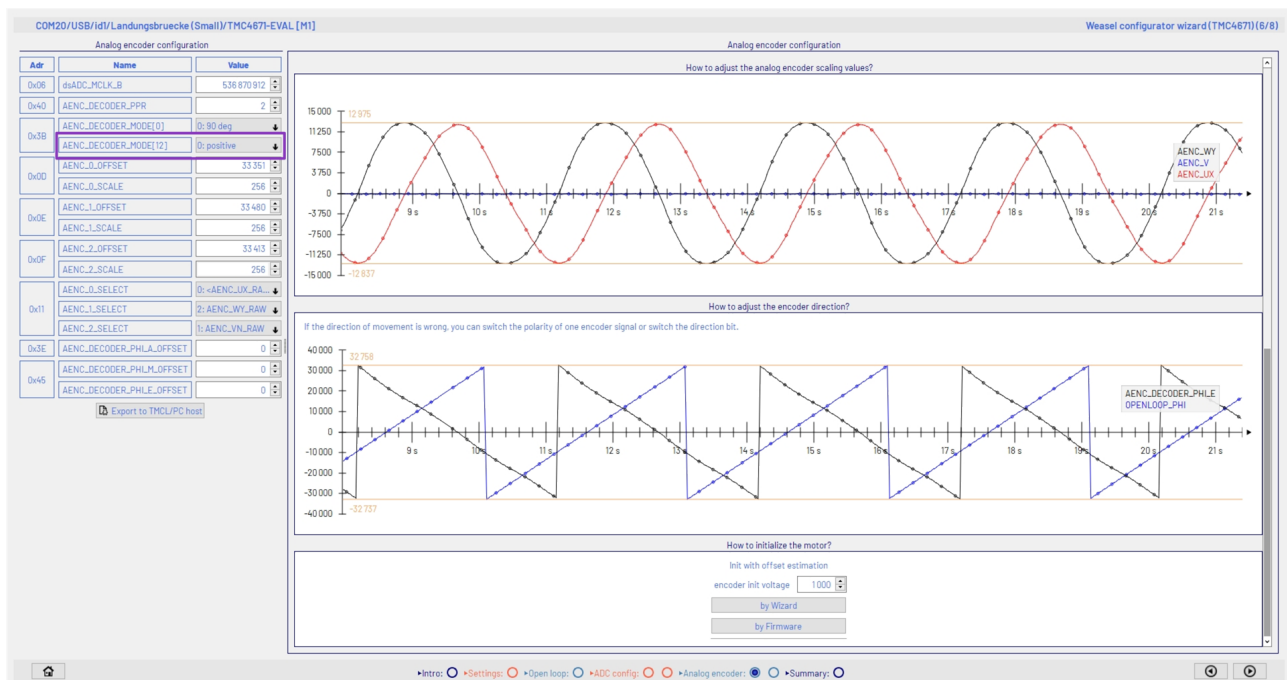


Figure 21: Analog Encoder Direction Setting

The offset is compensated by setting AENC_DECODER_PHI_M_OFFSET.

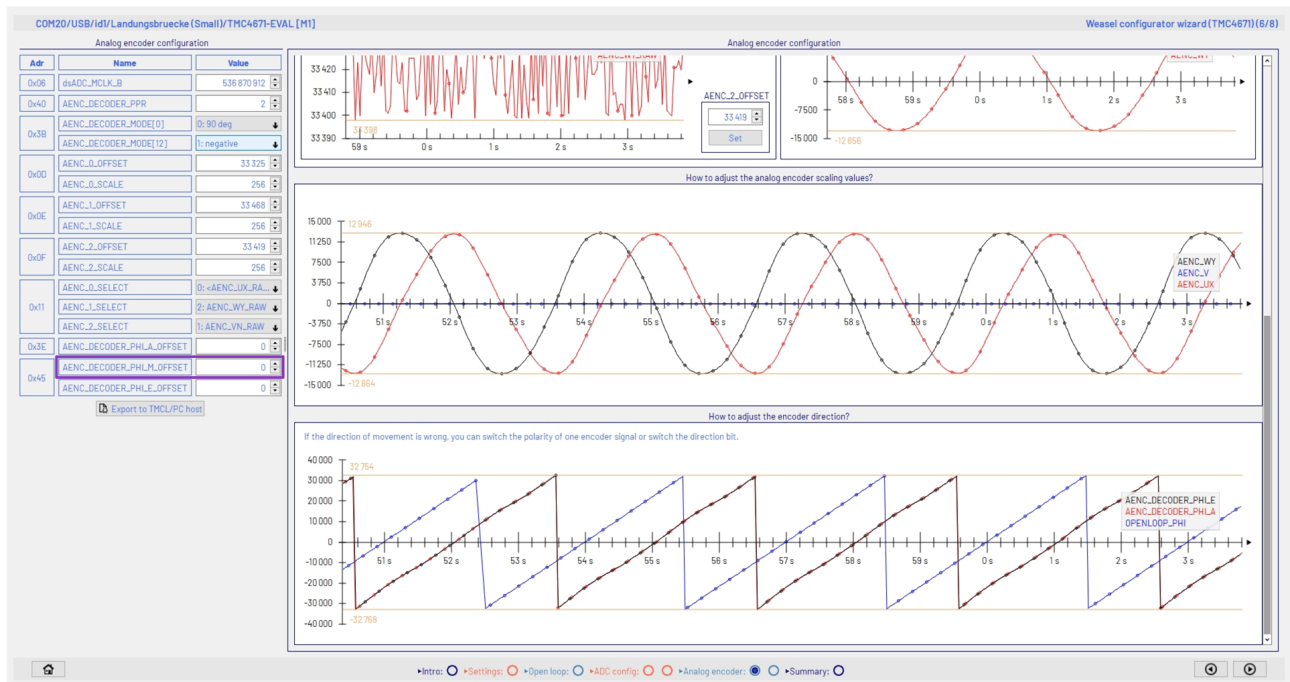


Figure 22: Analog Encoder before Offset Compensation

The offset can either be determined manually or automatically. For the automatic offset estimation, first set the *encoder init voltage* ①. Then use the *by Wizard* or *by Firmware* button ②. The *by Wizard* option sends the commands from the TMLC-IDE. The *by Firmware* option sends the commands from the Landungsbruecke. After the end of the automatic offset estimation, AENC_DECODER_PHI_M_OFFSET is updated ③.

The analog encoder is configured successfully when OPENLOOP_PHI matches AENC_DECODER_PHI_E ④:

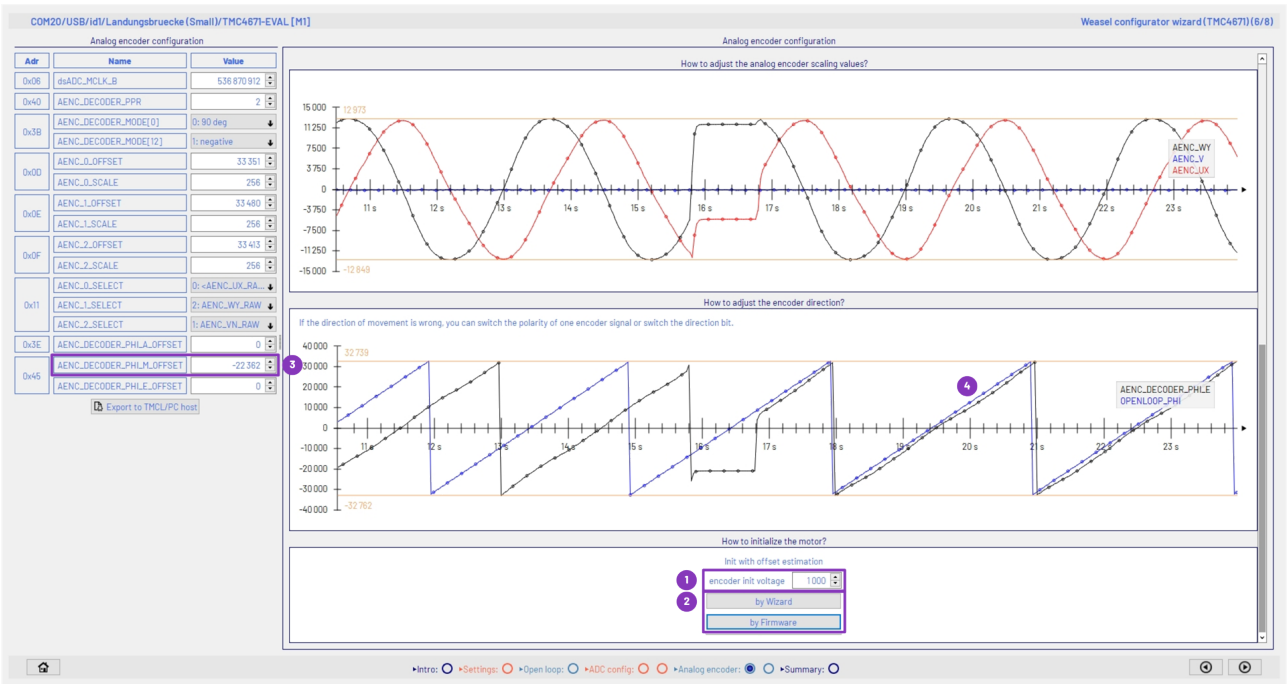


Figure 23: Configured Analog Encoder Automatic Offset Estimation

The analog encoder is configured successfully when OPENLOOP_PHI matches AENC_DECODER_PHI_E:

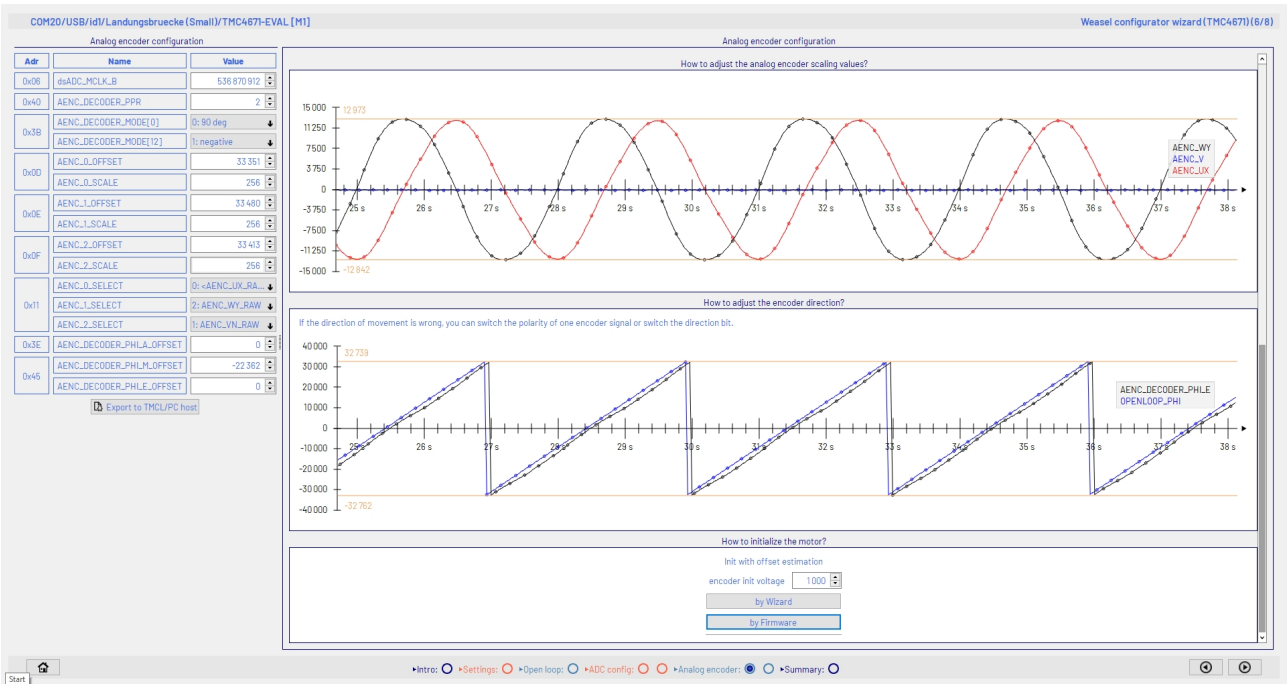
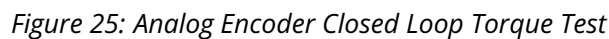


Figure 24: Analog Encoder, Configured

3.10 Encoder Test Drive (Torque Mode)

Using the analog encoder, the motor is driven in closed loop torque mode.



On the summary page, choose *analog encoder* for a preconfigured script, which includes the encoder offset compensation. Then, save the script using the *Export Script* button.



For an example of the exported script, see to the [example tpc script](#).

3.12 Feedback Selection

Close the wizard and open the *Feedback Selection* tool from the device tree view. Then, select the analog encoder (electrical positions) as depicted:

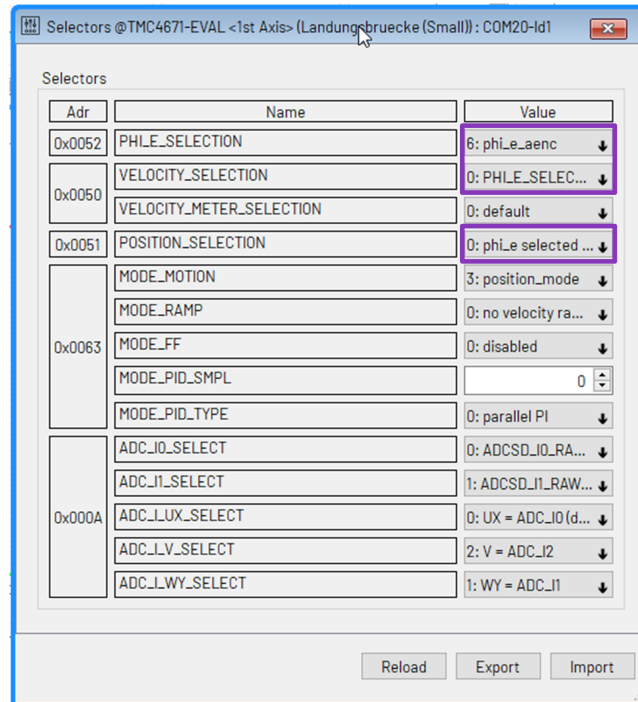


Figure 27: Feedback Selection for Analog Encoder

3.13 Driving the Motor

For PI tuning, refer to the [PI tuning appnote](#). Position mode step response test run with analog encoder:

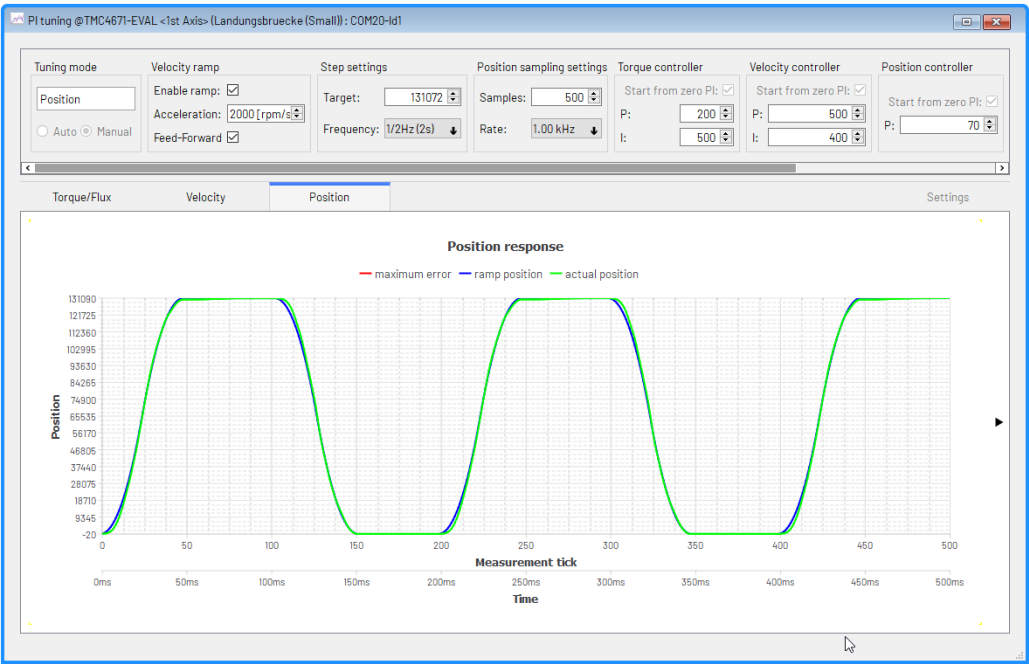


Figure 28: Position Step Control using the TMCL-IDE

4 Analog Hall with TMC4671

In this architecture, a three-phase analog hall sensor provides the feedback for commutation/torque loop as well as velocity and position control of the TMC4671.

4.1 Connection

For three-phase analog hall (with 120-phase shift) signals, following channels are used: AENC_UX, AENC_VN, and AENC_WY. The signals can be used as differential or single-ended inputs with negative inputs (AENC_UX_NEG, and AENC_VN_NEG, AENC_WY_NEG) to GND.

4.2 Analog Hall Configuration

Table 3 gives an overview of the most important registers of the TMC4671 for an analog hall sensor:

Address	Register Name	Function
0x06 _h	dsADC_MCLK_B	Delta sigma modulator clock for group B (include analog hall)
0x07 _h	dsADC_MDEC_B_MDEC_A	Filter for delta sigma modulators for group B (include analog hall)
0x40 _h	AENC_DECODER_PPR	Number of periods per revolution
0x3B _h	AENC_DECODER_MODE	Basic configuration for the analog hall
0x0D _h	AENC_0_SCALE_OFFSET	Calibration data for ADC channel AENC 0
0x0E _h	AENC_1_SCALE_OFFSET	Calibration data for ADC channel AENC 1
0x0F _h	AENC_2_SCALE_OFFSET	Calibration data for ADC channel AENC 2
0x11 _h	AENC_SELECT	Correct analog encoder ADC to compensate wiring twists
0x3E _h	AENC_DECODER_PHI_A_OFFSET	Offset of PHI_A for phase alignment
0x45 _h	AENC_DECODER_PHI_E_PHI_M_OFFSET	Offsets for electrical and mechanical angle for the analog encoder
0x50 _h	VELOCITY_SELECTION	Selection of velocity signal source
0x51 _h	POSITION_SELECTION	Selection of position signal source
0x52 _h	PHI_E_SELECTION	Selection of electrical angle used for commutation by FOC

Table 3: Register Overview Analog Hall

The following provides a guideline for the configuration.

4.3 Analog Hall Clock

To set up the analog encoder clock, do as follows:

Action:

Set *dsADC_MCLK_B* register 0x06 = 0d536870912 = 0x20000000 (default value)

Result:

The analog encoder clock group B is active at 25Mhz.

4.4 Analog Hall Mode

To set up a connected analog hall, do as follows:

Action:

Set `AENC_DECODER_MODE[0] = 0b1` (AENC_DECODER_MODE register 0x3B)

Result:

Selects processing of analog position signals of 0°, 120°, 240° (analog hall).

4.5 Hall PPR

For correct commutation, it is important to set the analog hall resolution (AENC_DECODER_PPR, 0x40), which is defined by periods per mechanical revolution. The analog hall periods per mechanical revolution depends on the motor pole pair number:

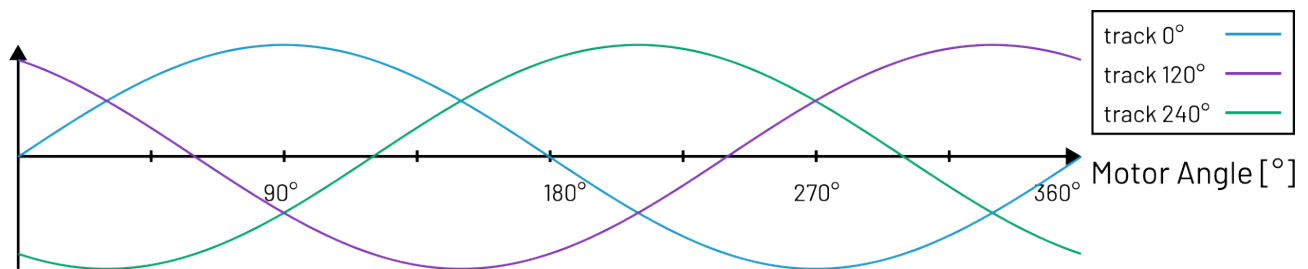


Figure 29: Analog Hall Signals, Motor Pole Pair = 1

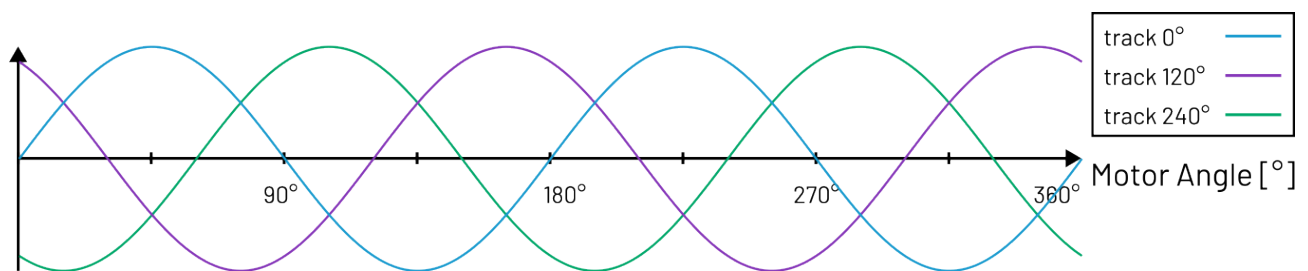


Figure 30: Analog Hall Signals, Motor Pole Pair = 2

Set the PPR to the motor pole pair number:

$$\text{AENC_DECODER_PPR} = \text{motor pole pair} \quad (4)$$

Refer to the motor data sheet to determine the motor pole pair number.

4.6 Analog Hall ADC Offset and Scale

The analog hall offset is determined automatically using the TMCL-IDE wizard. Also, refer to the [Python code example](#).

The scale factor is set to 256 for analog hall:

- AENC_0_SCALE in register 0x0D = 256
- AENC_1_SCALE in register 0x0E = 256
- AENC_2_SCALE in register 0x0F = 256

4.7 Analog Hall Phase Offset

The purpose of the hall offset configuration is to obtain a known motor and encoder position within the motor electrical period. The hall sensors are fixed within the motor. Thus, the hall offset is determined once.

To initialize the analog hall sensor, the wizard in the TMCL-IDE can be used.

The aim is to adapt `AENC_DECODER_PHI_M_OFFSET` manually so that `OPENLOOP_PHI` matches `AENC_DECODER_PHI_E`.

Usually, due to wiring, the hall offset is one of the following cases:

- $180^\circ \rightarrow \text{AENC_DECODER_PHI_E} = +/-32767$
- $120^\circ \rightarrow \text{AENC_DECODER_PHI_E} = +/-21844$
- $60^\circ \rightarrow \text{AENC_DECODER_PHI_E} = +/-10922$
- $30^\circ \rightarrow \text{AENC_DECODER_PHI_E} = +/-5461$

Due to motor tolerances, it may be required to tune these values.

The hall offset can be automatically determined. The procedure is the same as described in the section [analog encoder initialization](#). For a code example, see the [TPC script](#).

4.8 Feedback Selection

See the [analog encoder](#) section.

5 How to Configure a BLDC with Analog Hall with TMCL-IDE

5.1 TMC4671 Evaluation Board

The ADI Trinamic evaluation boards are being used as hardware platform for this guide. Following parts are used:

- [Landungsbruecke interface board](#)
- [TMC4671-EVAL](#)
- [TMC6100-EVAL \(powerstage\)](#)
- [TMCL-IDE](#)
- BLDC motor with analog hall [Faulhaber 4490 H 024 B K1155](#)
 - Motor pole pair = 1
 - Motor maximum rated current = 7.45A
 - Position feedback: analog hall

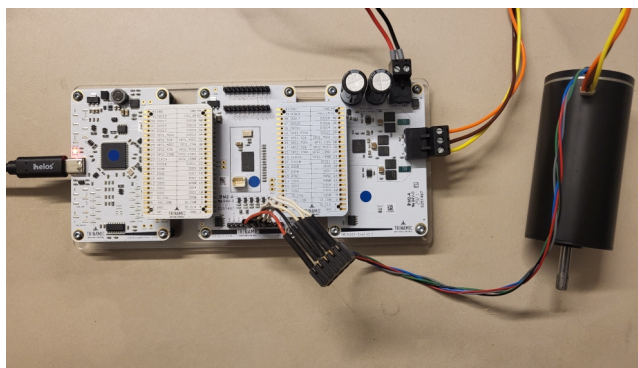


Figure 31: Hardware Setup Analog Hall

5.2 Overview Motor Configuration

Configuration by TMCL-IDE wizard

1. [Powerstage and motor](#)
2. [Motor current ADC](#)
3. [Analog Hall \(configuration and initialization\)](#)

Configuration with TMCL-IDE tools

1. [Feedback selection](#)
2. [PI Tuning and Position mode](#)

5.3 Open TMC4671 Wizard and Selection

See the steps in [analog encoder](#) section.

5.4 Settings - Powerstage and Motor

Select the TMC6100-EVAL as powerstage. For the motor in this example, *Set defaults for BLDC/PMSM motor* is used and number of pole pairs is set to 1.

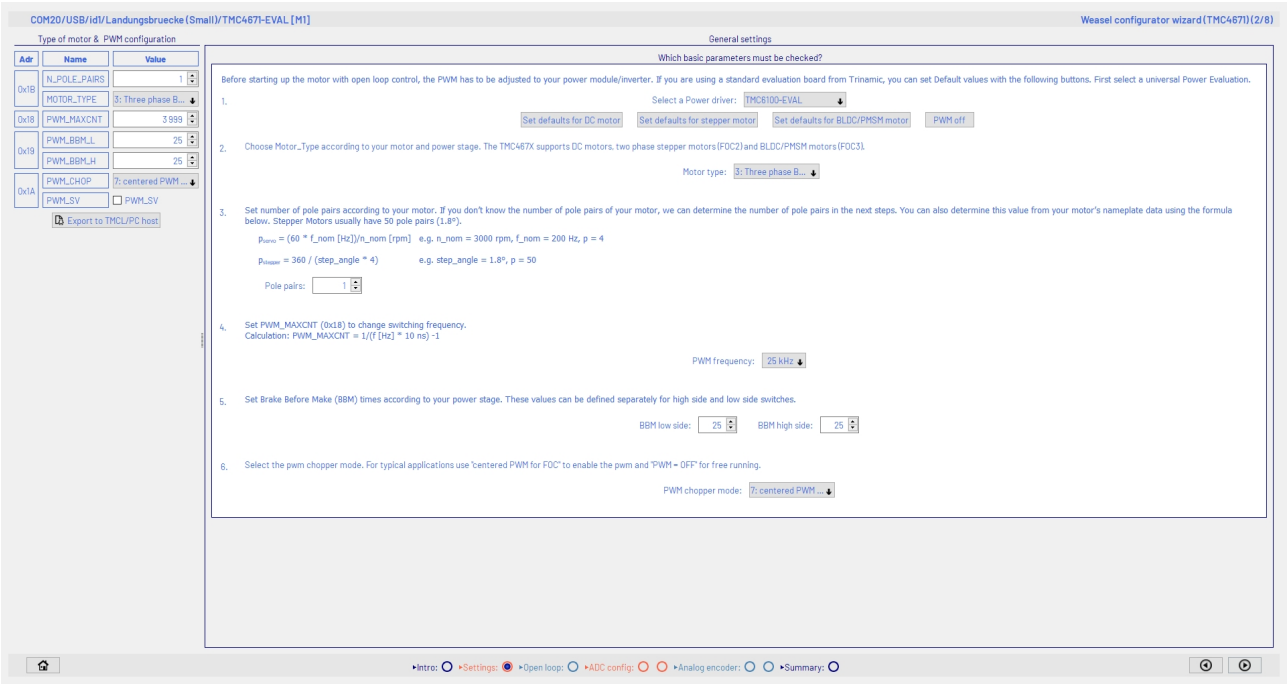


Figure 32: Powerstage and Motor Settings

5.5 Open Loop

Run the motor in open loop and configure the motor current ADC.

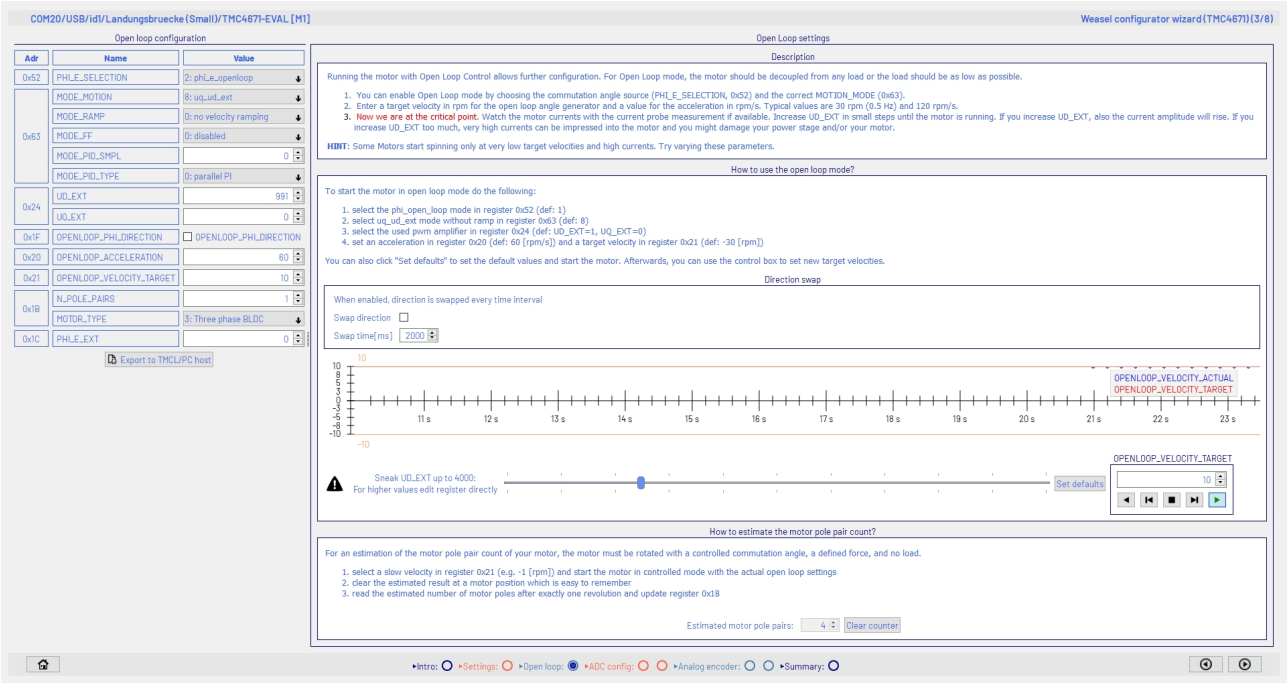


Figure 33: Open Loop Drive



Figure 35: Motor Current Offset Configuration

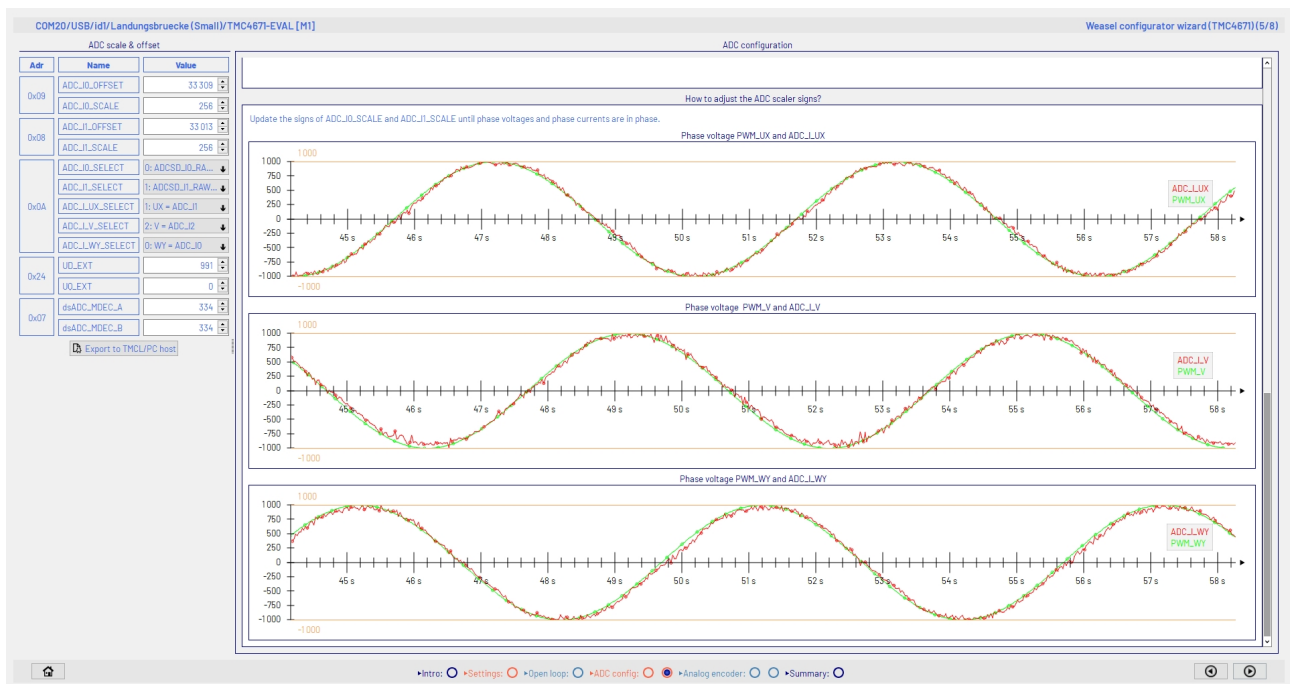


Figure 36: Motor Current Phase Selection

5.7 Analog Hall

First, the analog hall clock needs to be set **1**. The used motor has one motor pole pair per revolution. Thus, the PPR is set to one (ANALOG_DECODER_PPR) **2**. Then, AENC_DECODER_MODE (0x3B) Bit0 is set to 1 (= analog hall) **3**. Next, the offset of each phase is compensated using the Set button **4**.

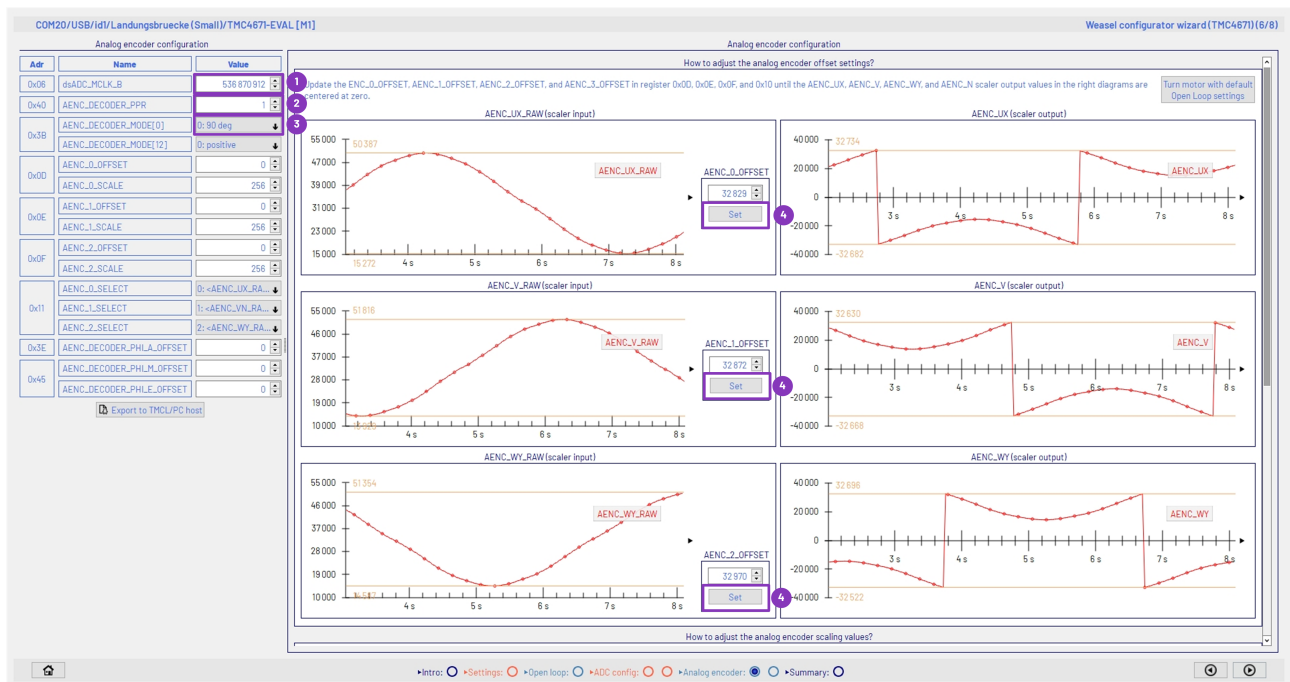
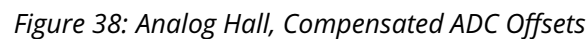


Figure 37: Analog Hall Configuration



COM20/USB/IdV/Landungsbruecke (Small)/TMC4671-EVAL [M1]
Weasel configurator wizard (TMC4671) [6/8]

Analog encoder configuration

Adr	Name	Value
0x08	2sADC_MCLK_LB	538 870 912
0x40	AENC_DECODER_PPR	1
0x3B	AENC_DECODER_MODE[0]	1: 120 deg
	AENC_DECODER_MODE[12]	0: positive
0x0C	AENC_0_OFFSET	32 844
	AENC_0_SCALE	256
0x0E	AENC_1_OFFSET	32 873
	AENC_1_SCALE	256
0x0F	AENC_2_OFFSET	32 987
	AENC_2_SCALE	256
0x10	AENC_0_SELECT	0: <AENC_0X_RA...>
	AENC_1_SELECT	1: <AENC_VIN_RA...>
	AENC_2_SELECT	2: <AENC_WY_RA...>
0x3C	AENC_DECODER_PHI_A_OFFSET	0
0x45	AENC_DECODER_PHI_H_OFFSET	0
	AENC_DECODER_PHI_L_OFFSET	0

Export to TMDL/PC host

Analog encoder configuration

How to adjust the analog encoder scaling values?

How to adjust the encoder direction?

If the direction of movement is wrong, you can switch the polarity of one encoder signal or switch the direction bit.

How to initialize the motor?

Init with offset estimation

encoder init voltage

by Wizard

by Firmware

Intro: ☐
Settings: ☒
Open loop: ☐
ADC config: ☐
Analog encoder: ☒
Summary: ☐

The offset can either be determined manually or automatically. For the automatic offset estimation, first set the *encoder init voltage* ❶. Then, use the *by Wizard* or *by Firmware* button ❷. The *by Wizard* option sends the commands from the TMLC-IDE. The *by Firmware* option sends the commands from the Landungs-

bruecke. After the end of the automatic offset estimation, AENC_DECODER_PHI_M_OFFSET is updated ³. The analog hall is configured successfully when OPENLOOP_PHI matches AENC_DECODER_PHI_E ⁴:

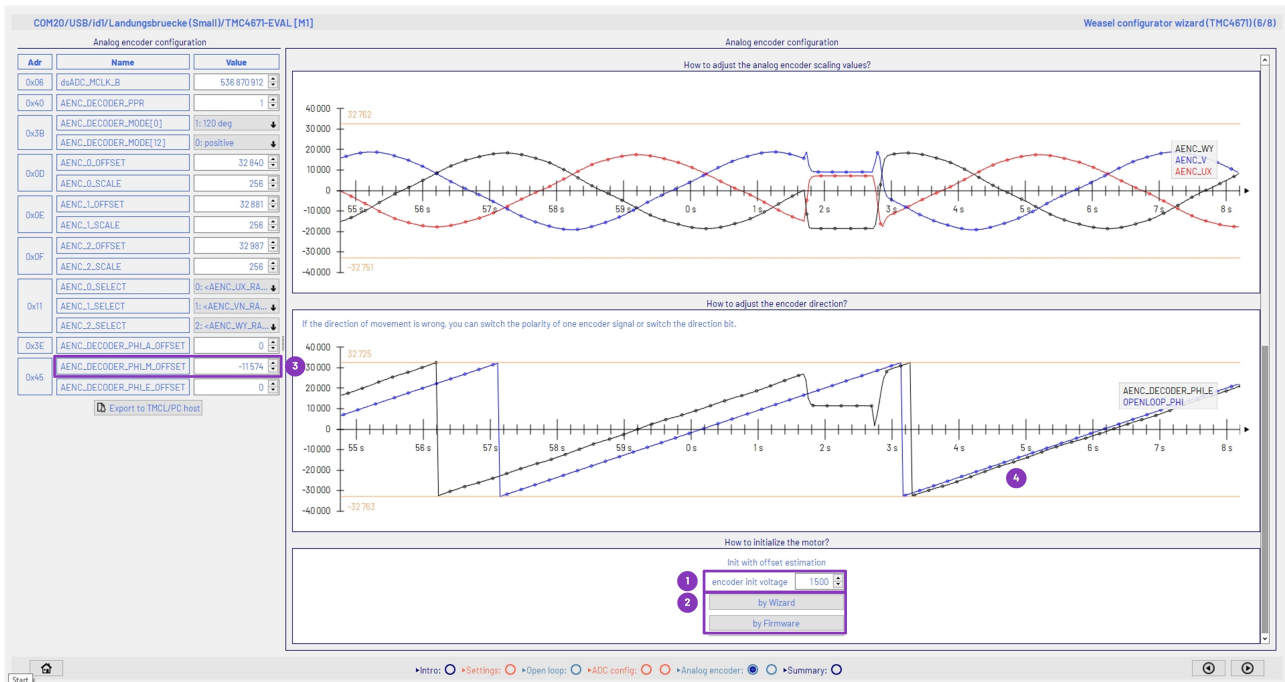


Figure 40: Configured Analog Hall with Automatic Offset Estimation

Alternatively, the offset is configured manually with one of the [common offset values](#).

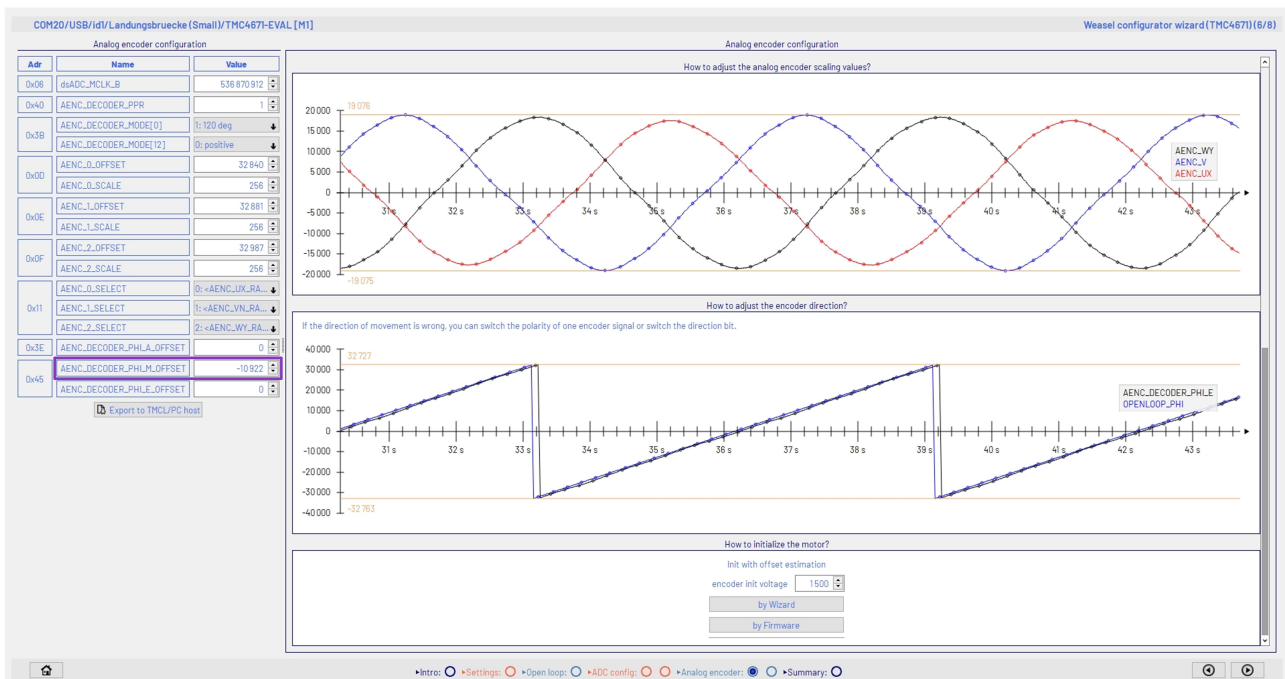


Figure 41: Configured Analog Hall with Manual Offset

5.8 Analog Hall Test Drive (Torque Mode)

Using the analog hall sensor, the motor is driven in closed loop torque mode.

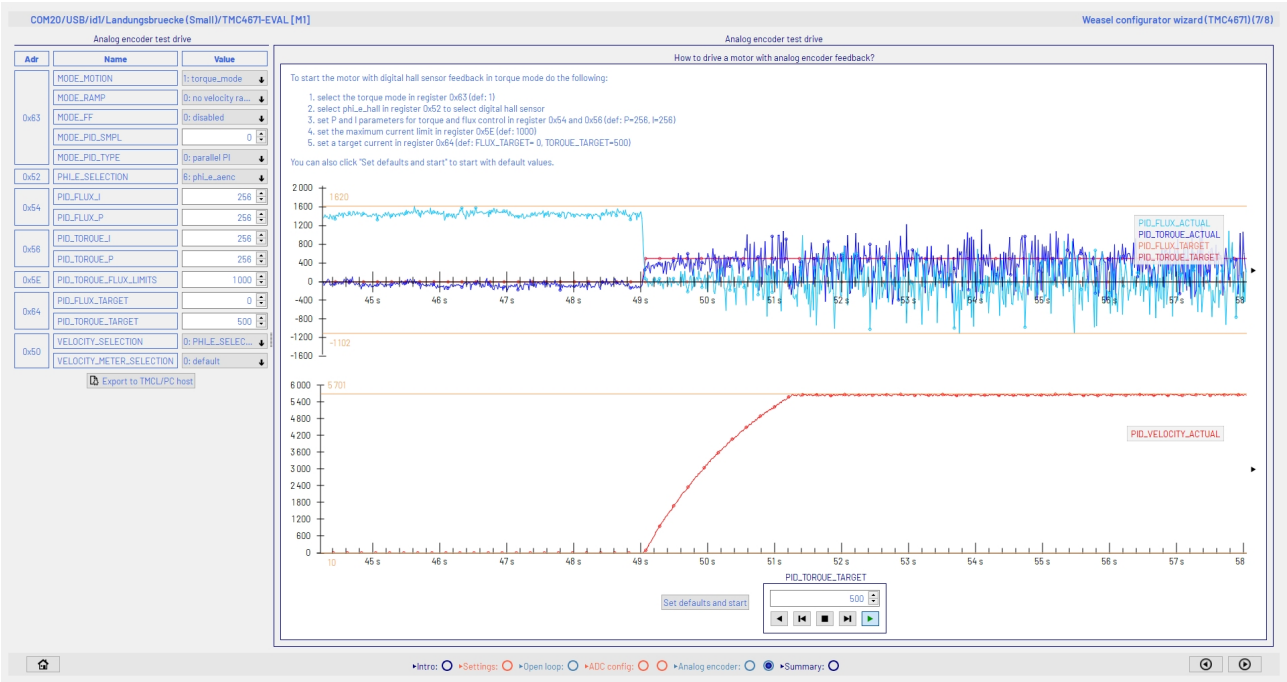


Figure 42: Analog Hall Closed Loop Torque Test

5.9 Save Configuration

On the summary page, choose *Analog encoder* for a preconfigured script. Then, save the script using the *Export Script* button.

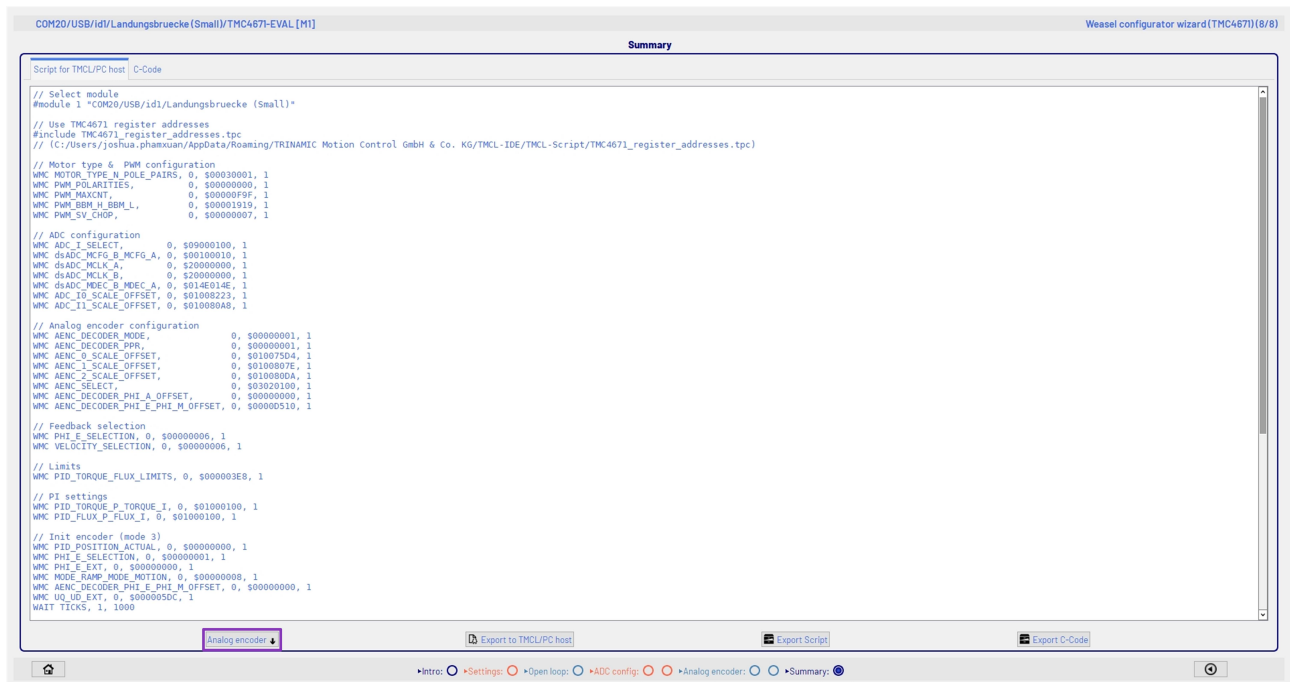


Figure 43: Save the Configuration to TPC script

For an example of the exported script, see the [example tpc script](#).

5.10 Feedback Selection

Close the wizard and open the *Feedback Selection* tool from the device treeview. Then, select the analog hall (electrical positions) as depicted:

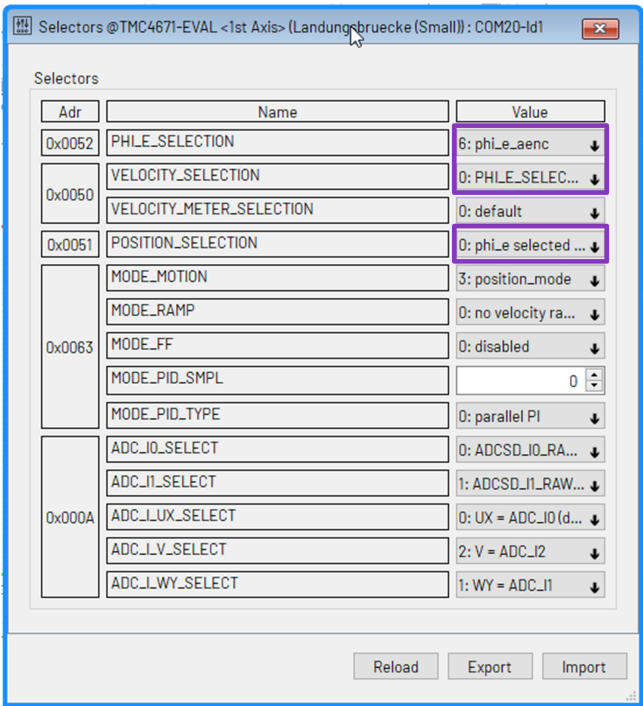


Figure 44: Feedback Selection for Analog Hall

5.11 Driving the Motor

For PI tuning, refer to the [PI tuning appnote](#). Position mode step response test run with analog hall:

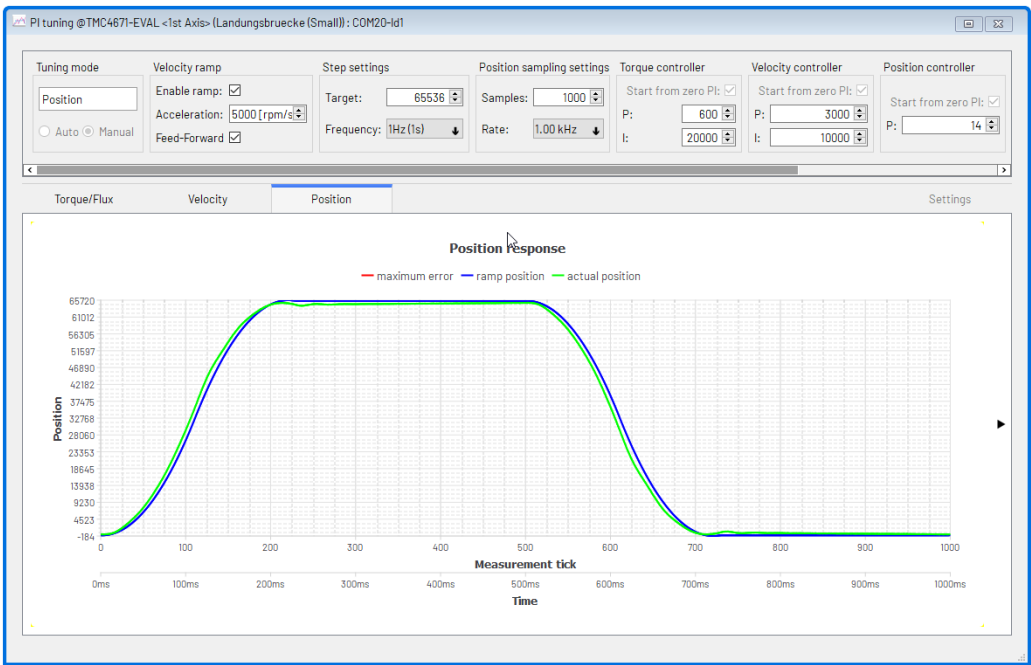


Figure 45: Position Step Control using the TMCL-IDE

6 How to Configure a BLDC with ABN Encoder Using an Analog Encoder

It is possible to initialize an ABN incremental encoder using an analog encoder. With this approach, there is no motor movement required to initialize the incremental encoder. This is helpful for applications where the motor movement is limited by mechanical end or where a motor movement at start-up should not happen (example, automated guided vehicle (AGV) or robots).

No motor movement during initialization can also be implemented by using hall sensor to initialize an incremental encoder. When using hall sensors, the motor is driven with the hall sensors for a short period (until the next hall sensor state change). In contrast, when using an analog encoder, the motor is controlled (commutated) from standstill using the incremental encoder.

It is required that the analog encoder offset is predetermined, which is for example the case if an encoder with a magnetic period of one is used (PPR = 1). Also, see section 2.7.3.

To initialize an incremental encoder using an analog encoder, proceed as follows:

1. Write ABN_DECODER_COUNT $0x27 = 0$.
2. Write ABN_DECODER_PHI_E_PHI_M_OFFSET $0x29 = 0$.
3. Write predetermined analog encoder offset AENC_DECODER_PHI_M_OFFSET $0x45$.
4. Read AENC_DECODER_PHI_M $0x46$.
5. Read ABN_DECODER_PHI_M $0x2A$.
6. Calculate offset: $AENC_DECODER_PHI_E - ABN_DECODER_PHI_E$.
7. Write calculated offset to ABN_DECODER_PHI_M_OFFSET $0x29$.
8. The initialization is successful if AENC_DECODER_PHI_M $0x46$ and ABN_DECODER_PHI_M $0x2A$ are aligned.

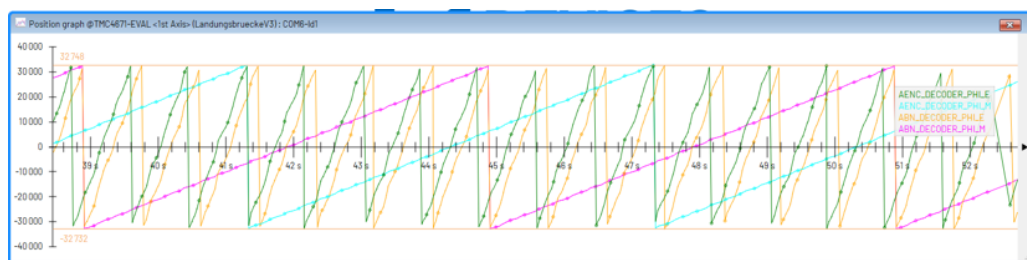


Figure 46: ABN Encoder before Initialization

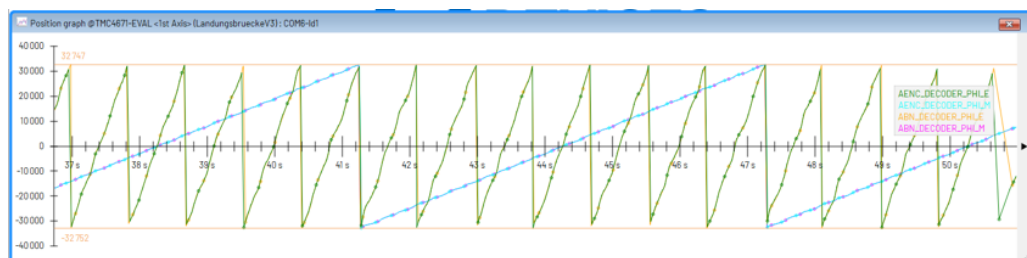


Figure 47: ABN Encoder before Initialization

For a code example, see the [example tpc script](#).

7 Additional Ressources

7.1 Automatic Analog Encoder Initialization TPC Script

Adapt motor/encoder configuration and UQ_UD_EXT value to own setup.

```

1 // Select module
#module 1 "COM20/USB/id1/Landungsbruecke (Small)"

3 // Use TMC4671 register addresses
5 #include TMC4671_register_addresses.tpc
// (C:/Users/user.name/AppData/Roaming/TRINAMIC Motion Control GmbH & Co. KG
  ↳ /TMCL-IDE/TMCL-Script/TMC4671_register_addresses.tpc)

7 // Motor type & PWM configuration
9 WMC MOTOR_TYPE_N_POLE_PAIRS, 0, $00030002, 1
WMC PWM_POLARITIES, 0, $00000000, 1
11 WMC PWM_MAXCNT, 0, $00000F9F, 1
WMC PWM_BBM_H_BBM_L, 0, $00000A0A, 1
13 WMC PWM_SV_CHOP, 0, $00000007, 1

15 // ADC configuration
WMC ADC_I_SELECT, 0, $18000100, 1
17 WMC dsADC_MCFG_B_MCFG_A, 0, $00100010, 1
WMC dsADC_MCLK_A, 0, $20000000, 1
19 WMC dsADC_MCLK_B, 0, $20000000, 1
WMC dsADC_MDEC_B_MDEC_A, 0, $014E014E, 1
21 WMC ADC_IO_SCALE_OFFSET, 0, $01008129, 1
WMC ADC_I1_SCALE_OFFSET, 0, $0100814A, 1

23 // Analog encoder configuration
25 WMC AENC_DECODER_MODE, 0, $00001000, 1
WMC AENC_DECODER_PPR, 0, $00000002, 1
27 WMC AENC_0_SCALE_OFFSET, 0, $01008247, 1
WMC AENC_1_SCALE_OFFSET, 0, $010082C8, 1
29 WMC AENC_2_SCALE_OFFSET, 0, $01008285, 1
WMC AENC_SELECT, 0, $03010200, 1
31 WMC AENC_DECODER_PHI_A_OFFSET, 0, $00000000, 1
WMC AENC_DECODER_PHI_E_PHI_M_OFFSET, 0, $0000A8A6, 1

33 // Feedback selection
35 WMC PHI_E_SELECTION, 0, $00000006, 1
WMC VELOCITY_SELECTION, 0, $00000006, 1

37 // Limits
39 WMC PID_TORQUE_FLUX_LIMITS, 0, $000003E8, 1

41 // PI settings
WMC PID_TORQUE_P_TORQUE_I, 0, $01000100, 1
43 WMC PID_FLUX_P_FLUX_I, 0, $01000100, 1

45 // Init encoder (mode 3)

```

```

WMC PID_POSITION_ACTUAL, 0, $00000000, 1
47 WMC PHI_E_SELECTION, 0, $00000001, 1 //0x52: Select phi_e_ext
WMC PHI_E_EXT, 0, $00000000, 1 //0x1C: Set electrical position to 0
49 WMC MODE_RAMP_MODE_MOTION, 0, $00000008, 1 //0x63: Select uq_ud_ext
WMC AENC_DECODER_PHI_E_PHI_M_OFFSET, 0, $00000000, 1 //0x45: Clear analog
    ↪ encoder offsets (phi_e, phi_m)
51 WMC UQ_UD_EXT, 0, $000003E8, 1 //0x24: Apply a fixed voltage at
    ↪ electrical position 0
WAIT TICKS, 1, 1000
53
// Calculate and use offset
55 RMC AENC_DECODER_PHI_E_PHI_M, 0, 1 //0x46: Read out analog encoder value
CALC NOT, 0
57 CALC AND, $0000FFFF
WMCA AENC_DECODER_PHI_E_PHI_M_OFFSET, 0, 1 //0x45: Set analog encoder offset
59
// Set voltage back to zero
61 WMC UQ_UD_EXT, 0, $00000000, 1

63 // Select PHI_E_AENC
WMC PHI_E_SELECTION, 0, $00000006, 1
65
//==== Analog encoder test drive ====
67
// Switch to torque mode
69 WMC MODE_RAMP_MODE_MOTION, 0, $00000001, 1

71 // Rotate right
WMC PID_TORQUE_FLUX_TARGET, 0, $03E80000, 1
73 WAIT TICKS, 1, 3000

75 // Rotate left
WMC PID_TORQUE_FLUX_TARGET, 0, $FC180000, 1
77 WAIT TICKS, 1, 3000

79 // Stop
WMC PID_TORQUE_FLUX_TARGET, 0, $00000000, 1

```

Listing 1: TPC Script Example

7.2 Automatic Analog Hall Initialization TPC Script

Adapt motor/encoder configuration and UQ_UD_EXT value to own setup.

```

1 // Select module
#module 1 "COM20/USB/id1/Landungsbruecke (Small)"
3
// Use TMC4671 register addresses
5 #include TMC4671_register_addresses.tpc
// (C:/Users/user.name/AppData/Roaming/TRINAMIC Motion Control GmbH & Co. KG
    ↪ /TMCL-IDE/TMCL-Script/TMC4671_register_addresses.tpc)
7
// Motor type & PWM configuration
9 WMC MOTOR_TYPE_N_POLE_PAIRS, 0, $00030001, 1

```



```

WMC PWM_POLARITIES,          0, $00000000, 1
11 WMC PWM_MAXCNT,            0, $00000F9F, 1
WMC PWM_BBM_H_BBM_L,        0, $00001919, 1
13 WMC PWM_SV_CHOP,           0, $00000007, 1

// ADC configuration
WMC ADC_I_SELECT,            0, $09000100, 1
17 WMC dsADC_MCFG_B_MCFG_A, 0, $00100010, 1
WMC dsADC_MCLK_A,            0, $20000000, 1
19 WMC dsADC_MCLK_B,          0, $20000000, 1
WMC dsADC_MDEC_B_MDEC_A,    0, $014E014E, 1
21 WMC ADC_IO_SCALE_OFFSET, 0, $01008223, 1
WMC ADC_I1_SCALE_OFFSET,    0, $010080A8, 1
23

// Analog encoder configuration
25 WMC AENC_DECODER_MODE,      0, $00000001, 1
WMC AENC_DECODER_PPR,        0, $00000001, 1
27 WMC AENC_0_SCALE_OFFSET,    0, $010075D4, 1
WMC AENC_1_SCALE_OFFSET,      0, $0100807E, 1
29 WMC AENC_2_SCALE_OFFSET,    0, $010080DA, 1
WMC AENC_SELECT,             0, $03020100, 1
31 WMC AENC_DECODER_PHI_A_OFFSET, 0, $00000000, 1
WMC AENC_DECODER_PHI_E_PHI_M_OFFSET, 0, $0000D85A, 1
33

// Feedback selection
35 WMC PHI_E_SELECTION, 0, $00000006, 1
WMC VELOCITY_SELECTION, 0, $00000006, 1
37

// Limits
39 WMC PID_TORQUE_FLUX_LIMITS, 0, $000003E8, 1

// PI settings
41 WMC PID_TORQUE_P_TORQUE_I, 0, $01000100, 1
43 WMC PID_FLUX_P_FLUX_I, 0, $01000100, 1

// Init encoder (mode 3)
WMC PID_POSITION_ACTUAL, 0, $00000000, 1
47 WMC PHI_E_SELECTION, 0, $00000001, 1 //0x52: Select phi_e_ext
WMC PHI_E_EXT, 0, $00000000, 1 //0x1C: Set electrical position to 0
49 WMC MODE_RAMP_MODE_MOTION, 0, $00000008, 1 //0x63: Select uq_ud_ext
WMC AENC_DECODER_PHI_E_PHI_M_OFFSET, 0, $00000000, 1 //0x45: Clear analog
    ↪ hall offsets (phi_e, phi_m)
51 WMC UQ_UD_EXT, 0, $000005DC, 1 //0x24: Apply a fixed voltage at
    ↪ electrical position 0
WAIT TICKS, 1, 1000
53

// Calculate and use offset
55 RMC AENC_DECODER_PHI_E_PHI_M, 0, 1 //0x46: Read out analog encoder value
CALC NOT, 0
57 CALC AND, $0000FFFF
WMCA AENC_DECODER_PHI_E_PHI_M_OFFSET, 0, 1 //0x45: Set analog encoder
    ↪ offset
59

// Set voltage back to zero

```

```

61 WMC UQ_UD_EXT, 0, $00000000, 1

63 // Select PHI_E_AENC
WMC PHI_E_SELECTION, 0, $00000006, 1

65 //==== Analog encoder test drive ====

67 // Switch to torque mode
69 WMC MODE_RAMP_MODE_MOTION, 0, $00000001, 1

71 // Rotate right
WMC PID_TORQUE_FLUX_TARGET, 0, $03E80000, 1
73 WAIT TICKS, 1, 3000

75 // Rotate left
WMC PID_TORQUE_FLUX_TARGET, 0, $FC180000, 1
77 WAIT TICKS, 1, 3000

79 // Stop
WMC PID_TORQUE_FLUX_TARGET, 0, $00000000, 1

```

Listing 2: TPC Script Example

7.3 ABN Incremental Encoder Initialization Using Analog Encoder TPC Script

```

1 // Select module
#module 1 COM6/USB/id1/LandungsbrueckeV3

3 // Use TMC4671 register addresses
5 #include TMC4671_register_addresses.tpc

7 MOTOR_RUN = 1

9 // User Variables
SETV 0,0,0 //init variable 0 --> ABN_DECODER_PHI_M
11 SETV 1,0,0 //init variable 1 --> AENC_DECODER_PHI_M
SETV 2,0,MOTOR_RUN //init variable 2 --> motor run flag

13

15 // Motor type & PWM configuration
WMC MOTOR_TYPE_N_POLE_PAIRS, 0, $00030007, 1
17 WMC PWM_POLARITIES, 0, $00000000, 1
WMC PWM_MAXCNT, 0, $00000F9F, 1
19 WMC PWM_BBM_H_BBM_L, 0, $00001919, 1
WMC PWM_SV_CHOP, 0, $00000007, 1

21 // ADC configuration
23 WMC ADC_I_SELECT, 0, $09000100, 1
WMC dsADC_MCFG_B_MCFG_A, 0, $00100010, 1
25 WMC dsADC_MCLK_A, 0, $20000000, 1
WMC dsADC_MCLK_B, 0, $20000000, 1
27 WMC dsADC_MDEC_B_MDEC_A, 0, $014E014E, 1
WMC ADC_IO_SCALE_OFFSET, 0, $010081DE, 1

```

```
29 WMC ADC_I1_SCALE_OFFSET, 0, $01008220, 1

31 // ABN encoder settings
WMC ABN_DECODER_MODE, 0, $00000000, 1
33 WMC ABN_DECODER_PPR, 0, $00009100, 1
WMC ABN_DECODER_COUNT, 0, $00000000, 1
35 WMC ABN_DECODER_PHI_E_PHI_M_OFFSET, 0, $00000000, 1

37 // Analog encoder configuration
WMC AENC_DECODER_MODE, 0, $00000000, 1
39 WMC AENC_DECODER_PPR, 0, $00000001, 1
WMC AENC_0_SCALE_OFFSET, 0, $01005211, 1
41 WMC AENC_1_SCALE_OFFSET, 0, $01000000, 1
WMC AENC_2_SCALE_OFFSET, 0, $01005106, 1
43 WMC AENC_SELECT, 0, $03020100, 1
WMC AENC_DECODER_PHI_A_OFFSET, 0, $00000000, 1
45 WMC AENC_DECODER_PHI_E_PHI_M_OFFSET, 0, $0000FA24, 1

47 // Limits
WMC PID_TORQUE_FLUX_LIMITS, 0, $000003E8, 1
49

// PI settings
51 WMC PID_TORQUE_P_TORQUE_I, 0, $01000100, 1
WMC PID_FLUX_P_FLUX_I, 0, $01000100, 1
53

55 // ==== ABN encoder init ====

57 // Stop the motor
WMC $63, 0, $00000000, 1 // write MODE_RAMP_MODE_MOTION
59 WAIT TICKS, 1, 500 // wait for motor stop

61 WMC ABN_DECODER_COUNT, 0, $00000000, 1

63 // Read abn encoder
RMC ABN_DECODER_PHI_E_PHI_M, 0, 1
65 CALC AND, $0000FFFF
COMP 32767
67 JC LE, positive1

69 CALC NOT, 0
CALC AND, $0000FFFF
71 CALC NOT, 0

73 positive1:
STOA 0, 0
75

// Read analog encoder
77 RMC AENC_DECODER_PHI_E_PHI_M, 0, 1
CALC AND, $0000FFFF
79 COMP 32767
JC LE, positive2
81

CALC NOT, 0
```

```
83 CALC AND, $0000FFFF
   CALC NOT, 0
85
   positive2:
87 STOA 1, 0

89 // Calculate abn_phi_m offset and write to register
   CALCAV SUB, 0
91 WMCA ABN_DECODER_PHI_E_PHI_M_OFFSET, 0, 1

93 // ==== motor test run ====

95 // Check if script should be continued
   CALCV COMP, 2, 0
97 JC EQ, script_end

99 // Feedback selection
   WMC PHI_E_SELECTION, 0, $00000003, 1
101 WMC VELOCITY_SELECTION, 0, $00000009, 1

103 // Switch to torque mode
   WMC MODE_RAMP_MODE_MOTION, 0, $00000001, 1
105
   // Rotate right
107 WMC PID_TORQUE_FLUX_TARGET, 0, $01F40000, 1
   WAIT TICKS, 1, 3000
109
   // Rotate left
111 //WMC PID_TORQUE_FLUX_TARGET, 0, $FC180000, 1
   //WAIT TICKS, 1, 3000
113
   // Stop
115 WMC PID_TORQUE_FLUX_TARGET, 0, $00000000, 1

117 script_end:
```

Listing 3: TPC Script Example

7.4 Useful Links

- [TM4671 product page](#)
- [TMC4671 PI tuning appnote](#)
- [Driving a linear stage with TMC4671](#)
- [TMC4671 API on github](#)
- [TMC4671 Python ressources](#)

8 Revision History

Revision Number	Revision Date	Description	Number of Pages Changed
Rev 0	03/25	Initial release	-

Table 4: Document Revision