A17810 END-OF-LINE CALIBRATION

By Solène Bastien
Allegro MicroSystems

INTRODUCTION

The A17810 is a high-precision inductive dual-angle position-sensor interface IC designed for steering-torque applications.

This application note provides the essential information needed to use this IC by describing:

- Usage of the IC
- Compensations available
- Compensation usage strategies
- How to calibrate and program compensations

USAGE OF THE IC

The A17810 is an inductive-position sensor interface IC for automotive electric power steering (EPS) applications. The internal algorithm measures the angular position of two metal targets placed near its transmit (TX) and receive (RX) coils, one on each side of the steering column. The angle difference between the two metal targets can be used to retrieve torque information.

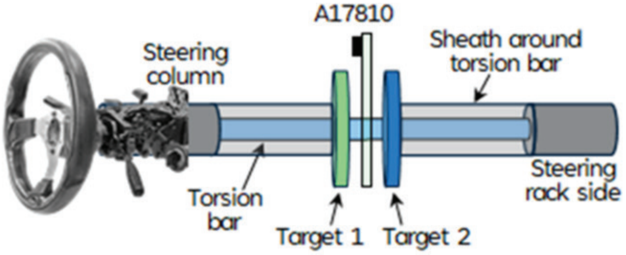


Figure 1: Dual-Angle Torque System

Table of Contents

The internal torque solving algorithms require the two metal targets to have a tooth-count ratio of 1:1, 2:1, or 3:1. For a 1:1 ratio, the TORQUE_ALG_SELECT EEPROM field must be set to 0; otherwise, TORQUE_ALG_SELECT must be set to 1. For any ratio not supported internally by the A17810 (such as when using 7 or 9 teeth targets, for example), the customer must calculate the torque information externally. In this case, the SENT data nibbles must be configured to provide both electrical angles on the output (refer to the A17810 datasheet).

COMPENSATIONS AVAILABLE

The A17810 contains nonvolatile EEPROM that stores configuration settings. This memory is preprogrammed by Allegro with default values that minimize the need for programming by the system implementer. The required programming depends on the application and coil system used.

The A17810 compensations listed in this section are static compensations that remain fixed throughout the device lifetime unless reprogrammed in EEPROM.

TX Driver Current: This adjusts the current in the transmitting coil to meet a defined reference voltage between the TXP and TXN pins. The reference voltage depends on target-to-coil air gap, system dimensions, and target material.

Front-End Amplifier Gain: This amplifies the received signals before analog-to-digital conversion to improve overall accuracy. The optimal code value depends on application air gap. In the A17810, two front-end gains are present, one for each receiving signal pair.

Channel Offset and Gain Trimming (OGT): This corrects for offset and amplitude mismatch between the RX signals. In general, OGT corrects for coil sensor imbalances in the printed circuit board (PCB). Gain-mismatch correction is between the two channels of the pair that senses the same target.

Electrical Angle Harmonic Compensation: This corrects periodic error that repeats each electrical cycle on the calculated angle. In the A17810, each calculated angle uses its own harmonic compensation trimming.

Zero-Angle Compensation: This applies an offset to the calculated angle, so that any arbitrary target position can represent 0°. In the A17810, two different offset values can be trimmed, one for each calculated angle.

Torque Linearity: When the internal torque solver is used in combination with EEPROM field TORQUE_ALG_SELECT = 0, this improves torque linearity across the full torque range. For TORQUE_ALG_SELECT = 1, this compensation is internally disabled.

- **Zero-Torque Compensation:** This removes IC offset error at zero torque and zero steering angle for true zero output at this position.
- **Output Gain:** This adjusts the application torque range to the IC full-scale output range and automatically clamps the output when outside of the range for gain values >1.

Input Phase Compensation (optional): This adjusts the phase delay in RX sampling with respect to the TX oscillator phase. RX signal is sampled at a fixed rate; the signal peaks may not be captured. This phase compensation allows the phase of the RX signal to be altered such that the signal peaks can be aligned with a sampling position, which allows the largest possible signal to be captured. By default, and programming simplicity, this compensation is internally bypassed. Allegro recommends to enable it only if small input signals are used.

Thresholds for Magnitude Checks (Optional): The magnitude check verifies that the IC input signals are within the range defined in the datasheet. The thresholds are EEPROM programmable and depend on the front-end (FE) gain applied.

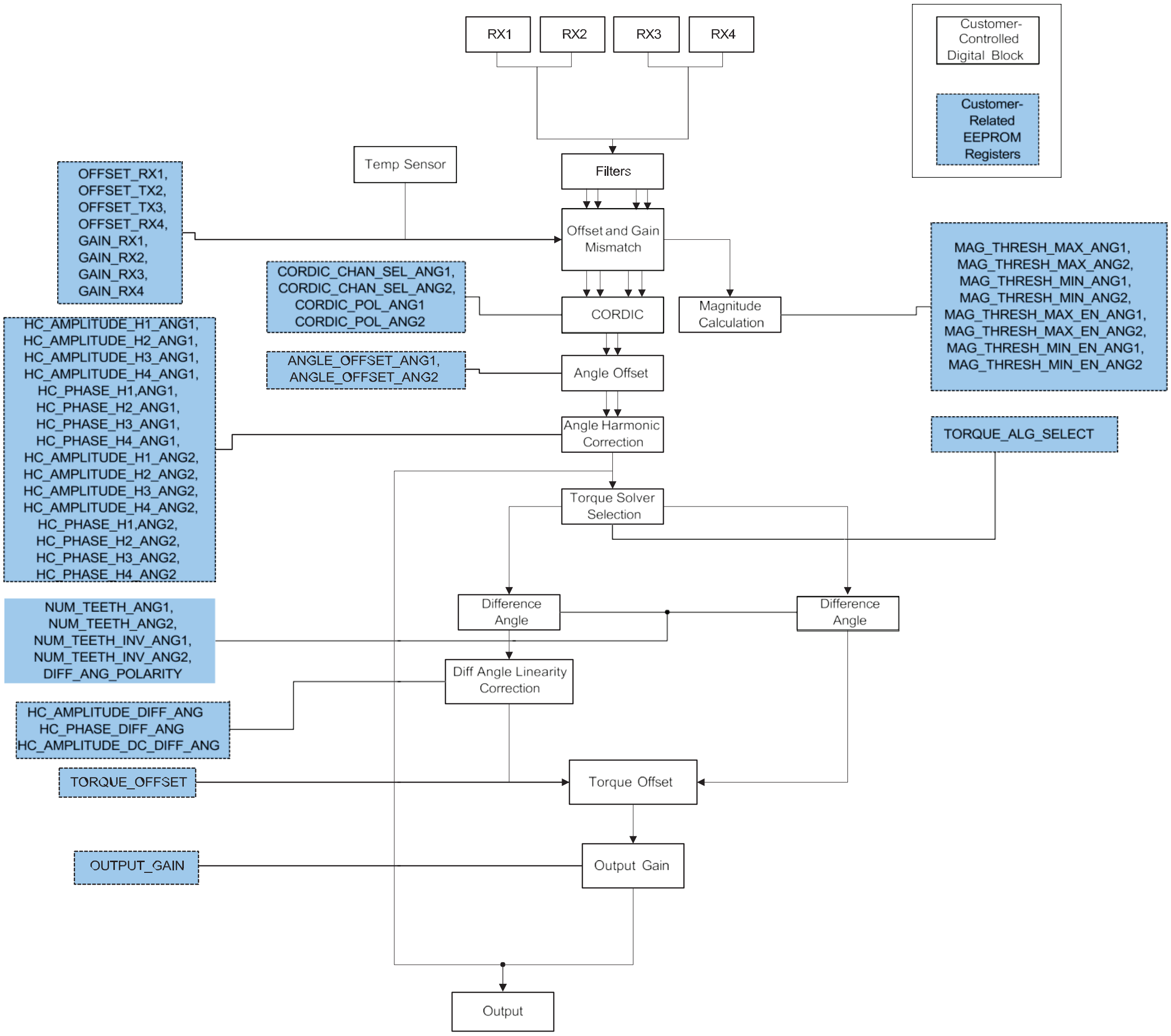


Figure 2: Compensations Flow Diagram: RX1/RX2 Senses One Target, and RX3/RX4 Senses the Other Target

COMPENSATION USAGE STRATEGIES

System implementers have the choice of which compensations to use when using the A17810, depending on accuracy requirements and system tolerances.

NOTE: Errors vary greatly from design to design, so the data shown here is only a reference.

Option 1: Basic Compensation

Only TX driver current and front-end amplifier gain are used. This is a baseline performance.

Option 2: Fixed Calibration Across System

A one-time characterization study determines the programmable compensations to apply to all units. This is effective at eliminating systematic errors while avoiding and/or reducing a measurement-based end-of-line calibration. TX current, FE amplifier-gain-trim channel-offset and gain-mismatch coefficients could be characterized during evaluation and trim value applied to all units end of line.

Option 3: Unique Calibration Per System

After end-of-line assembly, each system is measured and receives unique programmable compensations. This approach maximizes accuracy and combats errors introduced by mechanical misalignment.

Typically, electrical angle harmonic, the zero-angle, and torque linearity and zero-torque compensations benefit the most from unique calibration, while the other compensations can often use a fixed calibration approach.

HOW TO CALIBRATE AND PROGRAM COMPENSATIONS

The following describes how to collect data and set the programmable compensations.

Presample Acquisition Procedure

The following procedure is required for any calibration method and describes TX driver and front-end gain amplifier trimming. This provides baseline performances and does not require sample acquisition.

1. Send the EEPROM access code C6189D63 to the ACCESS field in memory register 0x72.
2. Before performing a sample acquisition, trim the transmitting driver current to provide operating condition oscillations of approximately 4.5 V peak-to-peak ($V_{pk\text{-}pk}$). TX current is determined by EEPROM field TX_TRIM (0x1A, bits [6:0]). Adjust TX_TRIM to obtain 4.5 $V_{pk\text{-}pk}$ between the TXP and TXN pins.

TX Current Trim Procedure

1. Set the shadow field TX_LOOP_EN = 0 (0x76, bit [24]). This disables the control loop on the TX current used for optimal performances during operation across temperature or dynamic shifts. In cases of multiple ICs connected to the same TX coil, this must be disabled on all ICs to correctly perform the TX trim.
2. The voltage can be directly measured via the TXP and TXN pins or by use of an internal test mode. To use the test mode, set the volatile bit to SIGMON_SAR_TRIGGER_VTX = 1 in the volatile memory address (0x94, bit [24]), immediately followed by a read of the SIGMON_SAR_VALUE in the volatile memory address (0x93, bits [8:0]). The 9-bit digital value can be converted to $V_{pk\text{-}pk}$ using:

$$V_{PK-PK} = \left(\frac{sigmon\_sar\_value}{511} \times 3.3V - 1.65\right) \times \frac{16}{3}$$

   Then, close the test mode by sending SIGMON_SAR_TRIGGER_VTX = 0 to the volatile memory address (0x94, bit [24]).
3. If the measured TX $V_{pk\text{-}pk}$ is less than 4.5 $V_{pk\text{-}pk}$, increase the TX_TRIM value in shadow (0x76, bits [6:0]); otherwise, lower its value. For cases where the TX coil is connected to two ICs, the TX_TRIM code must be updated only on one of the two ICs before the TX voltage is read; the other maintains TX_TRIM = 0.
4. If the TX $V_{pk\text{-}pk}$ is measured via the test mode, step 2B must be repeated each time a new TX_TRIM value is programmed to update the test mode value SIGMON_SAR_VALUE.
5. Once the TX voltage has reached 4.5 $V_{pk\text{-}pk}$, program the final TX_TRIM value in EEPORM (0x1A, bits [6:0]) on all ICs working with the measured TX coil.
6. Ensure TX_LOOP_EN = 1 in EEPROM (0x1A, bit [24]). For cases where the TX coil is connected to two ICs, only the IC with programmed TX_TRIM has the TX current control loop enabled (TX_LOOP_EN = 1) in EEPROM while the other IC remains disabled (TX_LOOP_EN = 0).

7. Program the EEPROM fields specific to the application system:

   A. NUM_TEETH_ANG(j), j = 1,2 (0x18, bits [4:0]; and 0x19, bits [4:0]). This is the target number of teeth used in the system minus 1. Some examples are:

Number of Teeth	NUM_TEETH_ANG1/2
1 tooth	0
2 teeth	1
3 teeth	2
...	...
31 teeth	30
32 teeth	31

   B. NUM_TEETH_INV_ANG(j), j = 1,2 (0x18, bits [22:5]) and [0x19, bits [22:5]). This is the inverse of the target number of teeth, with a maximum of two targets. This field is used in the torque solver when CUST_ALG_SEL = 1, for a tooth-count ratio of 2:1 or 3:1. For simplicity, the number of teeth intended to be used in the rewrap algorithm for each operational option has a value as follows:

Number of Teeth	1/Number of Teeth	NUM_TEETH_INV_ANG1/2 (binary)
1	1/1	100000000000000000
2	1/2	010000000000000000
3	1/3	001010101010101011
4	1/4	001000000000000000
5	1/5	000110011001100110
6	1/6	000101010101010101
7	1/7	000100100100100101
8	1/8	000100000000000000
9	1/9	000011100011100100
10	1/10	000011001100110011
11	1/11	000010111010001100
12	1/12	000010101010101011
13	1/13	000010011101100010
14	1/14	000010010010010010
15	1/15	000010001000100010
16	1/16	000010000000000000
17	1/17	000001111000011110
18	1/18	000001110001110010
19	1/19	000001101011110011
20	1/20	000001100110011010
21	1/21	000001100001100010
22	1/22	000001011101000110
23	1/23	000001011001000011
24	1/24	000001010101010101
25	1/25	000001010001111011
26	1/26	000001001110110001
27	1/27	000001001011110111
28	1/28	000001001001001001
29	1/29	000001000110101000
30	1/30	000001000100010001
31	1/31	000001000010000100
32	1/32	000001000000000000

C. CORDIC_POL_ANG1 (0x20, bit [0]) and CORDIC_POL_ANG2 (0x20, bit [1]) enable angle inversion for angle 1 and angle 2, respectively: 0 Increases the angle in the clockwise direction; 1 increases the angle in counter-clockwise direction.

D. OUTPUT_GAIN (0x2E, bits [16:1]). This adjusts the application output range to the IC full-scale range and automatically saturates the output when outside of the range for gain values >1; otherwise, the output rolls over after reaching the maximum of the range.

E. TORQUE_ALG_SELECT (0x18, bit [23]) must be set based on the tooth-count ratio used for the torque solver: For a 1:1 tooth-count ratio, set the logic to 0; for either a 2:1 or a 3:1 tooth-count ratio, set the logic to 1; and, for any other tooth-count ratio configuration, set the logic to 0.

8. Read all compensation fields in the table that follows. To ensure EOL calibration is performed correctly, if any compensation field does not equal 0, write it with the value 0.

Compensation	Field Name	EEPROM Address
Front-End Amplifier Gain	FE_AMP_GAIN_ANG1, FE_AMP_GAIN_ANG2	0x1B, bits [5:3], 0x1B, bits [8:6]
Channel Offset	OFFSET_RX1, OFFSET_RX2, OFFSET_RX3, OFFSET_RX4	0x1C, bits [13:0], 0x1D, bits [13:0], 0x1E, bits [13:0], 0x1F, bits [13:0]
Channel Gain	GAIN_RX1, GAIN_RX2, GAIN_RX3, GAIN_RX4	0x1C, bits [25:14], 0x1D, bits [25:14], 0x1E, bits [25:14], 0x1F, bits [25:14]
Angle Harmonic Amplitude	HC_AMPLITUDE_H1_ANG1, HC_AMPLITUDE_H2_ANG1, HC_AMPLITUDE_H3_ANG1, HC_AMPLITUDE_H4_ANG1, HC_AMPLITUDE_H1_ANG2, HC_AMPLITUDE_H2_ANG2, HC_AMPLITUDE_H3_ANG2, HC_AMPLITUDE_H4_ANG2	0x24, bits [24:11], 0x25, bits [24:11], 0x26, bits [24:11], 0x27, bits [24:11], 0x28, bits [24:11], 0x29, bits [24:11], 0x2A, bits [24:11], 0x2B, bits [24:11]
Angle Harmonic Phase	HC_PHASE_H1_ANG1, HC_PHASE_H2_ANG1, HC_PHASE_H3_ANG1, HC_PHASE_H4_ANG1, HC_PHASE_H1_ANG2, HC_PHASE_H2_ANG2, HC_PHASE_H3_ANG2, HC_PHASE_H4_ANG2	0x24, bits [10:0], 0x25, bits [10:0], 0x26, bits [10:0], 0x27, bits [10:0], 0x28, bits [10:0], 0x29, bits [10:0], 0x2A, bits [10:0], 0x2B, bits [10:0]
Zero Angle Offset	ANGLE_OFFSET_ANG1, ANGLE_OFFSET_ANG2	0x22, bits [16:0], 0x23, bits [16:0]
Torque Linearity Amplitude	HC_AMPLITUDE_DIFF_ANG	0x2D, bits [24:11]
Torque Linearity Phase	HC_PHASE_DIFF_ANG	0x2D, bits [10:0]
Torque Linearity DC	HC_AMPLITUDE_DC_DIFF_ANG	0x2C, bits [13:0]
Zero Torque Offset	TORQUE_OFFSET	0x1B, bits [25:13]

Front-End Amplifier Gain

To amplify the received signals before the analog-to-digital conversion, the front-end amplifier gain is trimmed as follows:

1. Read volatile complex-amplitude memory fields MAGNITUDE_ANG1 and MAGNITUDE_ANG2.

   For RX coils coupled with two TX coils, MAGNITUDE_ANG1 and MAGNITUDE_ANG2 must be collected with both TX drivers in the on state (ICs connected to each TX powered on), and volatile fields must be read on both ICs. The total complex amplitudes become:

   MAG1_TOTAL = MAGNITUDE_ANG1 of IC1
   + MAGNITUDE_ANG1 of IC2

   MAG2_TOTAL = MAGNITUDE_ANG2 of IC1
   + MAGNITUDE_ANG2 of IC2

2. Compare read values MAGNITUDE_ANG1 and MAGNITUDE_ANG2 (or, if RX is coupled with two TXs, compare read values MAG1_TOTAL and MAG2_TOTAL) with the table below, and select the appropriate gain code for each of the front-end amplifier gains, FE_AMP_GAIN_ANG1 and FE_AMP_GAIN_ANG2. If the input signals are less than a complex amplitude value of 1414.3 or higher than 29983.16, the input signals are too low or too high, respectively, for the IC to operate within the specified accuracy.

FE_AMP_GAIN_ANG Code	Acquired Value Code Range (LSB)
0	≤29983.16 and >20507.35
1	≤20507.35 and >13435.85
2	≤13435.85 and >10182.96
3	≤10182.96 and >8061.51
4	≤8061.51 and >6364.35
5	≤6364.35 and >5657.20
6	≤5657.20 and >4808.62
7	≤4808.62

3. Program the best gain code for each of the EEPROM fields FE_AMP_GAIN_ANG1 (0x1B, bits [5:3]) and FE_AMP_GAIN_ANG2 (0x1B, bits [8:6]).

Output Gain

Program the calculate torque to fit to the IC output range.

1. Perform a torque sweep along the full application torque range and collect the torque data through the IC SENT output.
2. Find the maximum absolute value from the collected data.
3. Calculate the output gain as follows:

   output gain decimal = max(abs(torque ideal value in code))/max((torque collected value in code))

   The torque ideal value in code is related to the SENT data nibble number of bits for $\text{diff}_{\text{Angle}}$. On SENT output, $\text{diff}_{\text{Angle}}$ is a signed representation from $-2^{\#\text{bits}}/2$ to $2^{\#\text{of bits}}/2 - 1$. Torque ideal value in codes is equal to $2^{\#\text{of bits}}/2 - 1$
4. Convert to fixed-length binary representation:

   output gain = Allegro_fi2bin(output gain decimal,0,16,10).
5. Write the calculated value in EEPROM field OUTPUT_GAIN (0x2E, bits [16:1]).

Thresholds for Magnitude Checks (Optional)

To detect input-signal variation, the IC offers the option to program minimum and maximum thresholds based on signal amplitude. A representation of the signal amplitudes related to RX1/2/3/4 is given in MAGNITUDE_ANG1 and MAGNITUDE_ANG2 volatile memory fields on 17 bits.

1. To use the detection option, the following fields must be set to 1:
   - MAG_THRESH_MAX_EN_ANG1 = (0x20, bit [3])
   - MAG_THRESH_MIN_EN_ANG1 = (0x20, bit [2])
   - MAG_THRESH_MAX_EN_ANG2 = (0x21, bit [3])
   - MAG_THRESH_MIN_EN_ANG2 = (0x21, bit [2])
2. Minimum and maximum thresholds for angle 1 and angle 2 can be calculated and converted to the fixed-length binary representation (see the Allegro_fi2bin function in Appendix B: Fixed-Point Binary Function):
   - MAG_THRESH_MIN_ANG1/2 = Allegro_fi2bin(MAG_THRESH_MIN_ANG1/2,0,11,11)
   - MAG_THRESH_MAX_ANG1/2 = Allegro_fi2bin(MAG_THRESH_MAX_ANG1/2,0,11,11)
3. Write each calculated parameter in the corresponding EEPROM field:

Fields	Type (si)	WL	FL	EEPROM Address
MAG_THRESH_MIN_ANG1	0	11	11	0x20, bits [14:4]
MAG_THRESH_MAX_ANG1	0	11	11	0x20, bits [25:15]
MAG_THRESH_MIN_ANG2	0	11	11	0x21, bits [14:4]
MAG_THRESH_MAX_ANG2	0	11	11	0x21, bits [25:15]

Sample Acquisition Procedure Using Controlled Angular Positions for 17810 Accuracy Optimization

- Mount the target(s) in a system equipped with an accurate rotary stage, depending on the application.

  NOTE: If the A17810 is used in an electric power-steering (EPS) torque application and with a tooth-count ratio of 1:1, the targets must be physically aligned with teeth and valley overlapping at the same position. Torque should not be applied on the torsion bar.
- Mount the sensor PCB precisely in front of the target(s) at the nominal air gap of the application. Ensure that any nonideal tilt in the PCB and/or target(s) is minimized
- To accurately calculate compensations, there must be an accurate reference system with the capability to rotate the system target(s) to unique, equally spaced angular positions over a complete mechanical revolution of 360°. If the intended end-of-line (EOL) calibration uses electrical angle harmonic compensation, there must be a sufficient number of angular positions, n, to resolve the content of a fourth harmonic in each electrical period. It is recommended that n angular positions be equal to or greater than 16 times the number of teeth, NTeeth, of the target-coil system.
- If the mechanical assembly of the target is sufficient (i.e., absent of both tilt and offset from the axis of rotation), measurement can be performed on a unique electrical period.
- To perform the calibration over continuous rotation, refer to Simplified Calibration Using SENT Output section.
- Repower the device, and send the access codes as in the Presample Acquisition Procedure, step 1.

Channel Offset and Gain Trimming

The A17810 compensates offset and gain nonidealities: in the balance of clockwise and counterclockwise windings; and between sine and cosine windings in the PCB coils. Compensation is performed directly at power-up with the offset and gain fields on each channel.

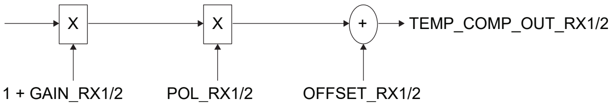
The offset and gain fields are programmed as follows:

1. Sweep the position of the target across the equally spaced n angular positions. At each position:
   A. Wait 10 µs.
   B. Read each volatile memory field TEMP_COMP_OUT_RX1/2/3/4.
   C. Uncomplement the read value (see the function in Appendix A: Uncomplement Function, uncomplement2C(double(TEMP_COMP_OUT_Rx1/2/3/4),17).
   D. Store the value of each field in an array, RX1/2/3/4, at the index of the current mechanical position:
      - TEMP_COMP_OUT_RX1 = (0x82, bits [16:0])
      - TEMP_COMP_OUT_RX2 = (0x83, bits [16:0])
      - TEMP_COMP_OUT_RX3 = (0x84, bits [16:0])
      - TEMP_COMP_OUT_RX4 = (0x85, bits [16:0])

      NOTE: If two ICs are connected to the same TX, both ICs must be powered on during data collection.
2. The mechanical position of each sample can be stored in an array called MECHANICAL_POSITION.
3. Each RX polarity can be individually controlled using the POL_RX1/2/3/4 EEPROM field.

Fields	EEPROM Address
POL_RX1	0x1B, bit [9]
POL _RX2	0x1B, bit [10]
POL _RX3	0x1B, bit [11]
POL _RX4	0x1B, bit [12]

4. Calculate gain and offset parameters for all ICs that apply according to the following channel-compensation path:



  - GAIN_RX1 = (Amplitude(RX2) / Amplitude(RX1)) – 1
  - GAIN_RX2 = 0
  - GAIN_RX3 = (Amplitude(RX4) / Amplitude(RX3)) – 1
  - GAIN_RX4 = 0

  Offsets on normalized signals are calculated as:

  - RX1/2/3/4_NORM = RX1/2/3/4 × (GAIN_RX1/2/3/4 + 1)
  - OFFSET_RX1/2/3/4 = –mean(RX1/2/3/4_NORM)

5. Convert each offset and gain calculated value to the fixed-length binary representation (see the Allegro_fi2bin function in Appendix B: Fixed-Point Binary Function).
  - OFFSET_RX(i)_SF = Allegro_fi2bin(OFFSET_RX(i),1,14,14), i = 1 to 4
  - GAIN_RX(i)_SF = Allegro_fi2bin(GAIN_RX(i),1,12,13), i = 1 to 4

6. Write each calculated parameter in the corresponding EEPROM field:

Fields	Type (si)	WL	FL	EEPROM Address
OFFSET_RX1	1	14	14	0x1C, bits [13:0]
OFFSET_RX2	1	14	14	0x1D, bits [13:0]
OFFSET_RX3	1	14	14	0x1E, bits [13:0]
OFFSET_RX4	1	14	14	0x1F, bits [13:0]
GAIN_RX1	1	12	13	0x1C, bits [25:14]
GAIN_RX2	1	12	13	0x1D, bits [25:14]
GAIN_RX3	1	12	13	0x1E, bits [25:14]
GAIN_RX4	1	12	13	0x1F, bits [25:14]

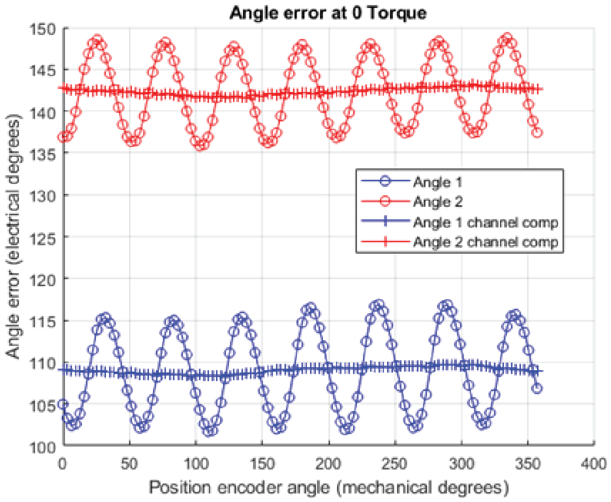


Figure 3: Angle Error with OGT Compensations

Angle Offset

The objective of the angle offset is to enable the rescale of an angle according to the reference.

To calibrate the correct angle offset, sweep the target position across the equally spaced n angular positions. At each position, for each IC:

1. Wait 10 µs , then:
    A. Read the volatile memory fields CORDIC_ANG1/2.
    B. Convert to degrees as:

    $\text{CORDIC\_ANG\_DEG1/2} = \text{modulo}((\text{cordic\_ang1/2})/2^{17} \times 360,360)$

    C. Store the value of each field in an array.
2. Calculate the error in degrees as:

    CORDIC_ERROR1/2 = modulo(CORDIC_ANG_DEG1/2 – refAngle1/2 + 180,360) – 180

    where refAngle is the reference angle in electrical degrees)
3. Calculate the offset on the angle as:

    ANGLE_OFFSET_ANG_DEG1/2 = mean(modulo(CORDIC_ERROR1/2,360));
4. Convert the value to the fixed-length binary representation:
    - ANGLE_OFFSET_ANG1 = Allegro_fi2bin(ANGLE_OFFSET_ANG_DEG1 /360,0,17,17)
    - ANGLE_OFFSET_ANG2 = Allegro_fi2bin(ANGLE_OFFSET_ANG_DEG2/360,0,17,17)
5. Write each obtained value into the respective EEPROM field:
    - ANGLE_OFFSET_ANG1 (0x22, bits [16:0])
    - ANGLE_OFFSET_ANG2 (0x23, bits [16:0])
6. The compensated offset angle (OFFS_ANG1/2) is calculated in the IC as:

    OFFS_ANG1/2 = CORDIC_ANG1/2 – ANGLE_OFFSET_ANG1/2
7. Optionally, the output angle rotation direction with respect to the input signals can be changed by changing the value of the EEPROM fields CORDIC_POL_ANG1 (0x20, bit [0]) and CORDIC_POL_ANG2 (0x21, bit [0]) for ANGLE1 and ANGLE2, respectively.

Electrical Angle Harmonic Compensation

The A17810 can compensate for electrical harmonic distortion in each angle due to the nonidealities of coil design, coil fabrication, and field uniformity across the coil surface. Compensation for the first, second, third, and fourth electrical harmonics can be made on each electrical angle.

To calculate the harmonic compensation parameters, the previous data acquisition is used. The objective is to calculate the amplitude and phase of each harmonic to apply the following equation:

$$\begin{aligned}\text{HC\_ANG1} = \text{OFFS\_ANG1} - (&\text{HC\_AMPLITUDE\_H1\_ANG1} \times \sin(w \times (\text{OFFS\_ANG1} + \text{HC\_PHASE\_H1\_ANG1}))\\ &+ \text{HC\_AMPLITUDE\_H2\_ANG1} \times \sin(w \times (2 \times \text{OFFS\_ANG1} + \text{HC\_PHASE\_H2\_ANG1}))\\ &+ \text{HC\_AMPLITUDE\_H3\_ANG1} \times \sin(w \times (3 \times \text{OFFS\_ANG1} + \text{HC\_PHASE\_H3\_ANG1}))\\ &+ \text{HC\_AMPLITUDE\_H4\_ANG1} \times \sin(w \times (4 \times \text{OFFS\_ANG1} + \text{HC\_PHASE\_H4\_ANG1}))\\ &)\end{aligned}$$

where OFF_ANG1 is the offset-corrected angle output, and HC_ANG1 is the output angle for target 1.

A similar equation applies for the angle of target 2.

1. Find parameters HC_AMPLITUDE_H_ANG and HC_PHASE_H_ANG to fit this function:

    CORDIC_ERROR1 = a0 + am1 × sind(1 × xm + pm1) + a1 × sind(1 × x + p1) + a2 × sind(2 × x + p2) + a3 × sind(3 × x + p3) + a4 × sind(4 × x + p4)

    where:
    - xm = reference angle in mechanical degrees
    - a1/2/3/4 = HC_AMPLITUDE_H1/2/3/4_ANG1
    - p1/2/3/4 = HC_PHASE_H1/2/3/4_ANG1 (±180°)
    - x = CORDIC_ANG_DEG1 – mean(modulo(CORDIC_ERROR1,360)) (= the offset-corrected angle output)
    - am1 and pm1 cannot be compensated internally because they correct for mechanical harmonics on error caused by target wobble, for example.
2. Convert the obtained values to the fixed-length binary representation and write them in EEPROM:
    - HC_AMPLITUDE_H1_ANG1 = Allegro_fi2bin(a1/45,1,14,14)
    - HC_PHASE_H1_ANG1 = Allegro_fi2bin(p1/360,1,11,11)

3. Repeat this process for the second angle.

Fields	Type (si)	WL	FL	EEPROM Address
HC_AMPLITUDE_H1_ANG1	1	14	14	0x24, bits [24:11]
HC_AMPLITUDE_H2_ANG1	1	14	14	0x25, bits [24:11]
HC_AMPLITUDE_H3_ANG1	1	14	14	0x26, bits [24:11]
HC_AMPLITUDE_H4_ANG1	1	14	14	0x27, bits [24:11]
HC_PHASE_H1_ANG1	1	11	11	0x24, bits [10:0]
HC_PHASE_H2_ANG1	1	11	11	0x25, bits [10:0]
HC_PHASE_H3_ANG1	1	11	11	0x26, bits [10:0]
HC_PHASE_H4_ANG1	1	11	11	0x27, bits [10:0]
HC_AMPLITUDE_H1_ANG2	1	14	14	0x28, bits [24:11]
HC_AMPLITUDE_H2_ANG2	1	14	14	0x29, bits [24:11]
HC_AMPLITUDE_H3_ANG2	1	14	14	0x2A, bits [24:11]
HC_AMPLITUDE_H4_ANG2	1	14	14	0x2B, bits [24:11]
HC_PHASE_H1_ANG2	1	11	11	0x28, bits [10:0]
HC_PHASE_H2_ANG2	1	11	11	0x29, bits [10:0]
HC_PHASE_H3_ANG2	1	11	11	0x2A, bits [10:0]
HC_PHASE_H4_ANG2	1	11	11	0x2B, bits [10:0]

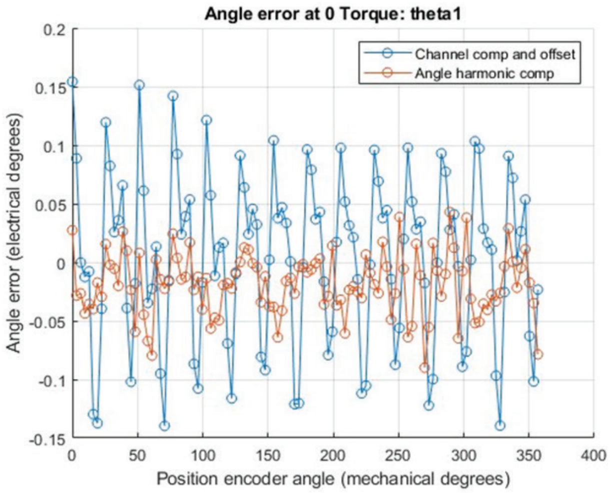


Figure 4: Angle Error After Harmonics Compensation

A17810 Torque Linearity Compensation

The A17810 integrates a torque solver that outputs the difference between the two electrical angles. A specific internal compensation on the torque linearity is possible when using the torque solver with a tooth-count ratio of 1:1.

1. To calculate compensation parameters, different torque must be applied with a precise angular reference system. The torque value can be sampled at multiple equally spaced torque positions. A minimum of five torque samples are needed, from minimum torque value to maximum torque value (minimum, minimum/2, zero, maximum/2,maximum) and with a torque step defined by the torque range divided by 4, in this case.
2. For each step, read DELTA_ANG in the volatile memory and store values in an array. The acquired DELTA_ANG array values are represented in two's complement on 17 bits. To calculate trim, parameter values must be uncomplemented:

   DELTA_ANG = uncomplement2C(DELTA_ANG, 17).
3. The uncomplemented data must be converted to electrical degrees as:

   DELTA_ANG = DELTA_ANG / 2^{Nbits} × 360

   where Nbits is defined either by 2^{17} in case of data sampled from volatile memory or by the number of bits from the SENT output).
4. Calculate TORQUE_ERROR = DELTA_ANG – APPLIED_TORQUE

   where the applied torque is in electrical degrees.
5. Find parameters HC_AMPLITUDE_DC_DIFF_ANG and HC_AMPLITUDE_DIFF_ANG, HC_PHASE_DIFF_ANG to fit this function:

   TORQUE_ERROR = a0 + a1 × sind(1 × x + p1)

   where:

   - a0 = HC_AMPLITUDE_DC_DIFF_ANG
   - a1 = HC_AMPLITUDE_DIFF_ANG
   - p1 = HC_PHASE_DIFF_ANG (±180°)
   - x = DELTA_ANG in electrical degrees

6. Convert the obtained values to the fixed-length binary representation and write them in EEPROM:
   - HC_AMPLITUDE_DIFF_AMP = Allegro_fi2bin(a1/45,1,14,14)
   - HC_PHASE_DIFF_ANG = Allegro_fi2bin(p1/360,1,11,11)
   - HC_AMPLITUDE_DC_DIFF_ANG = Allegro_fi2bin(a0/45,1,14,14)

Fields	Type (si)	WL	FL	EEPROM Address
HC_AMPLITUDE_DIFF_ANG	1	14	14	0x2D, bits [24:11]
HC_PHASE_DIFF_ANG	1	11	11	0x2D, bits [10:0]
HC_AMPLITUDE_DC_DIFF_ANG	1	14	14	0x2C, bits [13:0]

A17810 Zero-Torque Compensation

The zero-torque compensation gives the possibility for the A17810 to output exactly zero at the zero-torque and zero-steering-angle position, where best accuracy is usually required.

1. Ensure that neither torque nor zero steering angle are applied in the system.
2. Read the volatile memory field DELTA_ANG (0x8F, bits [16:0]), or read the SENT output directly, to collect the torque value at the zero-torque/zero-steering position.
3. The acquired DELTA_ANG value is represented in two's complement on 17 bits. To calculate trim, parameter values must be uncomplemented:

   DELTA_ANG = uncomplement2C(DELTA_ANG, 17).
4. Convert the obtained value to the fixed-length binary representation and write it in EEPROM:

   TORQUE_OFFSET = Allegro_fi2bin(DELTA_ANG,1,13,17)

Fields	Type (si)	WL	FL	EEPROM Address
TORQUE_OFFSET	1	13	17	0x1B, bits [25:13]

5. The compensated torque is calculated in the IC as:

   DELTA_ANG = DELTA_ANG – TORQUE_OFFSET

Simplified Calibration Using SENT Output

Advanced calibration requires use of the Manchester interface to read internal nodes before the angle calculation is performed at specific angle positions.

It is possible to perform a simplified calibration that does not use those nodes, by using SENT output data at specific angle positions.

Channel gain and channel offset cannot be directly compensated with this calibration, but the first and second harmonics compensation of the angle can resolve these specific errors depending on the gain mismatch and channel-offset amplitude.

1. Write EEPROM field OUTMSG_MODE = 6 to configure the SENT interface for the trigger SENT protocol, where data latch upon the falling edge of the trigger.
2. Write EEPROM field SENT_ DATA_CFG to 16. ANGLE1 and ANGLE2 become available with 16-bit resolution on alternative SENT frames: ANGLE1 becomes available with the even frames, and ANGLE2 becomes available with the odd frames.
3. Sweep the position of the target across the equally spaced n angular positions. Perform this sweet at each position, for each IC.
4. Using the trigger SENT message:
   A. Acquire data for ANGLE1/ANGLE2.

   B. Convert the data to degrees as:

      CORDIC_ANG_DEG1/2 = modulo((CORDIC_ANG1/2)/2^{16} × 360,360)

   C. Store the value of each field in an array.
5. Calculate the error in degrees as:

   CORDIC_ERROR1/2 = modulo(CORDIC_ANG_DEG1/2 – refAngle1/2 + 180,360) – 180

   where refAngle is the reference angle in electrical degrees.
6. Calculate the offset on angle as:

   ANGLE_OFFSET_ANG_DEG1/2 = mean(modulo(CORDIC_ERROR1/2,360)
7. Convert the value to the fixed-length binary representation:
   - ANGLE_OFFSET_ANG1 = Allegro_fi2bin(ANGLE_OFFSET_ANG_DEG1 /360,0,17,17)
   - ANGLE_OFFSET_ANG2 = Allegro_fi2bin(ANGLE_OFFSET_ANG_DEG2/360,0,17,17)
8. Write the value from step 7 in EEPROM.
   The compensated offset angle (OFFS_ANG) is calculated in the IC with: OFFS_ANG1/2 = CORDIC_ANG1/2 – ANGLE_OFFSET_ANG1/2
9. Using the previous acquisition, calculate the harmonic-compensation trims (amplitude and phases).
10. Find parameters HC_AMPLITUDE_H_ANG and HC_PHASE_H_ANG to fit the function:

    CORDIC_ERROR1 = a0 + am1 × sind(1 × xm + pm1) + a1 × sind(1 × x + p1) + a2 × sind(2 × x + p2) + a3 × sind(3 × x + p3) + a4 × sind(4 × x + p4)

    where:
    - am1 and pm1 are useful if there are mechanical harmonics on the error due to target wobble (cannot be compensated).
    - xm = reference angle is in mechanical degrees
    - a1/2/3/4 = HC_AMPLITUDE_H1/2/3/4_ANG1
    - p1/2/3/4 = HC_PHASE_H1/2/3/4_ANG1 (±180°)
    - x = CORDIC_ANG_DEG1 – mean(modulo(CORDIC_ERROR1,360)) (= the offset-corrected angle output)
11. Convert the values to the fixed-length binary representation and write them in EEPROM:
    - HC_AMPLITUDE_H1_ANG1 = Allegro_fi2bin(a1/45,1,14,14);
    - HC_PHASE_H1_ANG1 = Allegro_fi2bin(p1/360,1,14,14);
12. Repeat the process for ANG2.

APPENDIX A: UNCOMPLEMENT FUNCTION

MATLAB source code to uncomplement two's-complement signed field data. N_bits corresponds to the field number of bits.

```
function array_out = uncomplement2C(array_in, n_bits)
range = (2^n_bits);
threshold = range/2 - 1;
array_out = double(array_in);
for i = 1:length(array_in)
if array_out(i) > threshold
array_out(i) = array_out(i) - range;
end
end
end
```

APPENDIX B: FIXED-POINT BINARY FUNCTION

The MATLAB method below can be used to convert a decimal value into its fixed-point binary representation.

```
function [int_value] = Allegro_fi2bin(value,si,wl,fl)
% Convert a fixed point number to a binary value
% value - decimal value
% si - 0/1 0 = unsigned 1 = signed
% wl - The word length or number of bits
% fl - The fractional length (number of places to move the decimal)
if si
fi_max = (2^(wl-1)-1)/2^fl;
fi_min = -(2^(wl-1))/2^fl;
else
fi_max = (2^wl-1)/2^fl;
fi_min = 0;
end

if value > fi_max
value = NaN;
warning('OVERFLOW');
end

if value < fi_min
value = NaN;
warning(‘OVERFLOW’);
end

if isnan(value)
int_value = value;
else
negative_num = 0;
if si && value < 0
value = 2^wl/2^fl+value;
negative_num = 1;
end
value = round(2^fl * value);
int_value = value;
end
end
```

Revision History

Number	Date	Description
–	June 26, 2025	Initial release
1	December 9, 2025	Modified Table of Contents (page 1), Compensations Available (page 2), Compensation Usage Strategies (page 4), and Front-End Amplifier Gain (page 6) sections; added Output Gain section (page 7); and modified FL values (pages 11 and 12)