Contents

List of Figures

List of Tables

1. Introduction

This document explains the calibration and linearization process for the ZMID5201 using the analog output; it describes in detail the available algorithms and tools, and the considerations needed for correct calibration and linearization.

Recommendation: Read the *ZMID520x Datasheet* before using this document.

Important: These procedures are specifically for linearization and calibration using the analog output (the SOUT pin) or Spatialangle (SPA) that is read directly from register.

This document has three main parts:

- Theory and examples
- Considerations for correct measurements and clamping
- The *OneStepCalibration* algorithm for automatic calibration and linearization, which includes the algorithm for clamping calculations

For a correct calibration and linearization, a measurement of the target position during a complete sweep of the target is needed. Section 5 discusses the important considerations for a correct measurement.

The *OneStepCalibration* algorithm is designed for a fast and accurate calibration and linearization. The algorithm needs only a small number of configuration parameters, and it can function with SPA or analog output data.

OneStepCalibration algorithm is implemented in the following software:

- ZMID520x Graphical User Interface (GUI)
- Python® * using DLL (refer to the *ZMID520x User Guide: Calibration and Linearization with Python and the DLL* for details)
- LabVIEW™ † using DLL (refer to the *ZMID520x User Guide: Calibration and Linearization Using LabView and the Calibration DLL* for details)

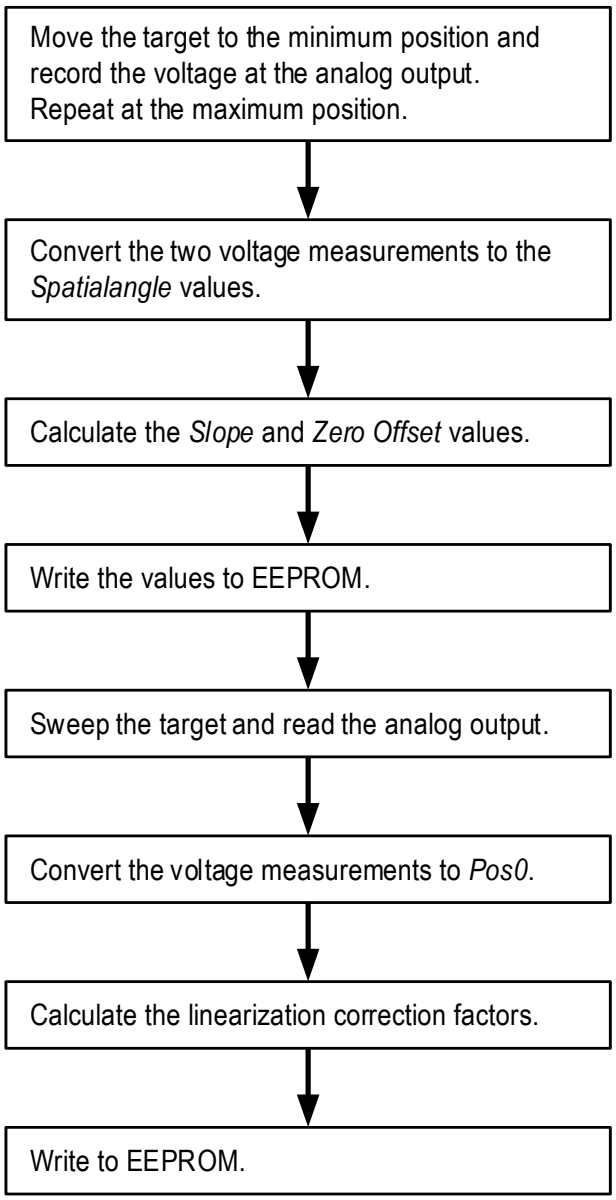
* Python™ is a trademark of the Python Software Foundation.

† LabVIEW™ is a trademark of National Instruments.

2. ZMID5201 Calibration and Linearization Theory

The flow diagram in Figure 1 shows the steps for calibration and linearization. Note: The flow that is implemented in the *OneStepCalibration* algorithm is different.

Figure 1. Calibration and Linearization Flow Diagram



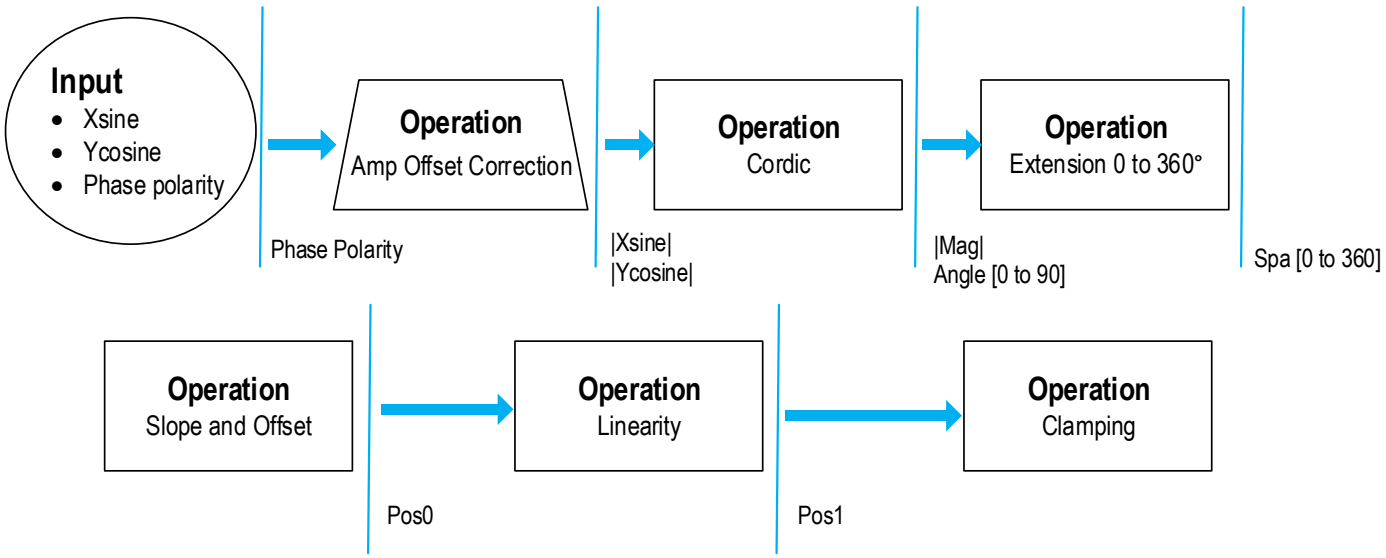
2.1 *Spa*, *Pos0*, and *Pos1* Registers

The processing of data in the ZMID5201 is represented in Figure 2.

Important registers:

- The *Spa* register represents the position value of the target without any calibration or linearization.
- The *Pos0* register represents the position value of the target after calibration for slope and zero offset.
- The *Pos1* register represents the position value of the target after calibration (slope and zero offset) and after linearization (corrections at 9 data points).

Figure 2. ZMID5201 Data Operations Block Diagram



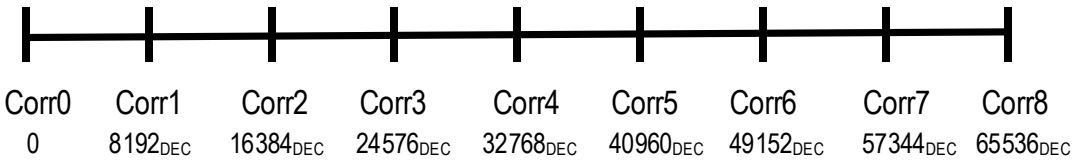
2.2 Nonlinearity Error Correction

The value of the measured position after calibration is stored in the *Pos0* register (shadow word register (SWR) address 19_{HEX}; read only). *Pos0* is a 16-bit value covering a range of output values [0, $2^{16} - 1$].

The measurement output range of 2^{16} is divided into 8 sections. Each section has a range of 2^{13}, and the boundaries are the correction points.

Correction points are fixed, and their values are expressed with respect to the maximum position output value 2^{16}. The linearization procedure is a "one-dimensional" procedure where only the values for the y-coordinate (i.e., the measured position values) can be corrected to match the ideal measurement curve. The position values of the correction points for the Y coordinates are defined by the EEPROM (E2P) address range 03_{HEX} to 07_{HEX}.

Figure 3. Correction Points



The Y correction factor is a number in the range of ±127 decimal, i.e., ±6.2% full scale (FS), and it is applied on the 11 MSB (most significant position bits) of the measured position.

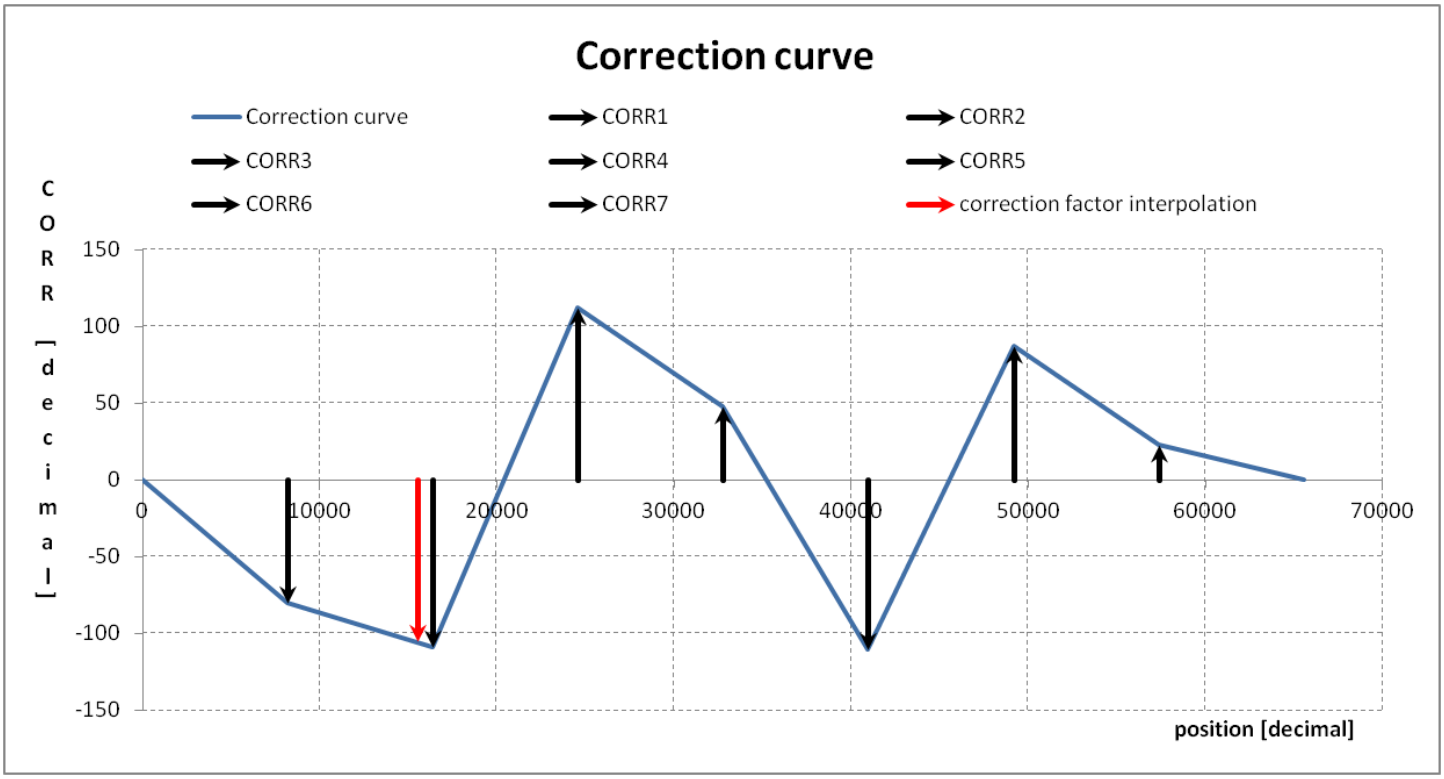
Table 1. Correction Factors

Parameter	x-coord (Fixed)	y-coord Range (Variable/11 MSB)
Corr0	0	$\pm 127_{DEC}$
Corr1	$1 \times 2^{13} = 8192_{DEC}$	$\pm 127_{DEC}$
Corr2	$2 \times 2^{13} = 16384_{DEC}$	$\pm 127_{DEC}$
Corr3	$3 \times 2^{13} = 24576_{DEC}$	$\pm 127_{DEC}$
Corr4	$4 \times 2^{13} = 32768_{DEC}$	$\pm 127_{DEC}$
Corr5	$5 \times 2^{13} = 40960_{DEC}$	$\pm 127_{DEC}$
Corr6	$6 \times 2^{13} = 49152_{DEC}$	$\pm 127_{DEC}$
Corr7	$7 \times 2^{13} = 57344_{DEC}$	$\pm 127_{DEC}$
Corr8	$8 \times 2^{13} = 65536_{DEC}$	$\pm 127_{DEC}$

Each section is identified by two correction points, so a linear interpolation function can be defined between each pair of corrections points. The set of linear interpolation functions connecting the correction points defines a linear piecewise function expressing the correction function over the whole measurement range of 2^{16}.

Applying the correction function to position value *Pos0* after the zero-point offset and slope calibration can improve the linearity by more than a factor of 2. After the correction is applied, the measured position can be monitored by reading the *Pos1* register.

Figure 4. Correction Curve



2.3 Zero-Point Offset and Slope Calibration

The goal of the calibration is to ensure that at the mechanical stroke ends, the measured positions are at zero and the maximum position respectively.

The receiver input signals are transformed by the arctan function into a linear position curve called the “spatial angle.” The spatial angle is transformed by the calibration process using the zero offset and slope to calculate the calibrated position:

position_cal = (*Spa* – *Zero Offset*) × *Slope*

2.3.1 Zero Offset Correction

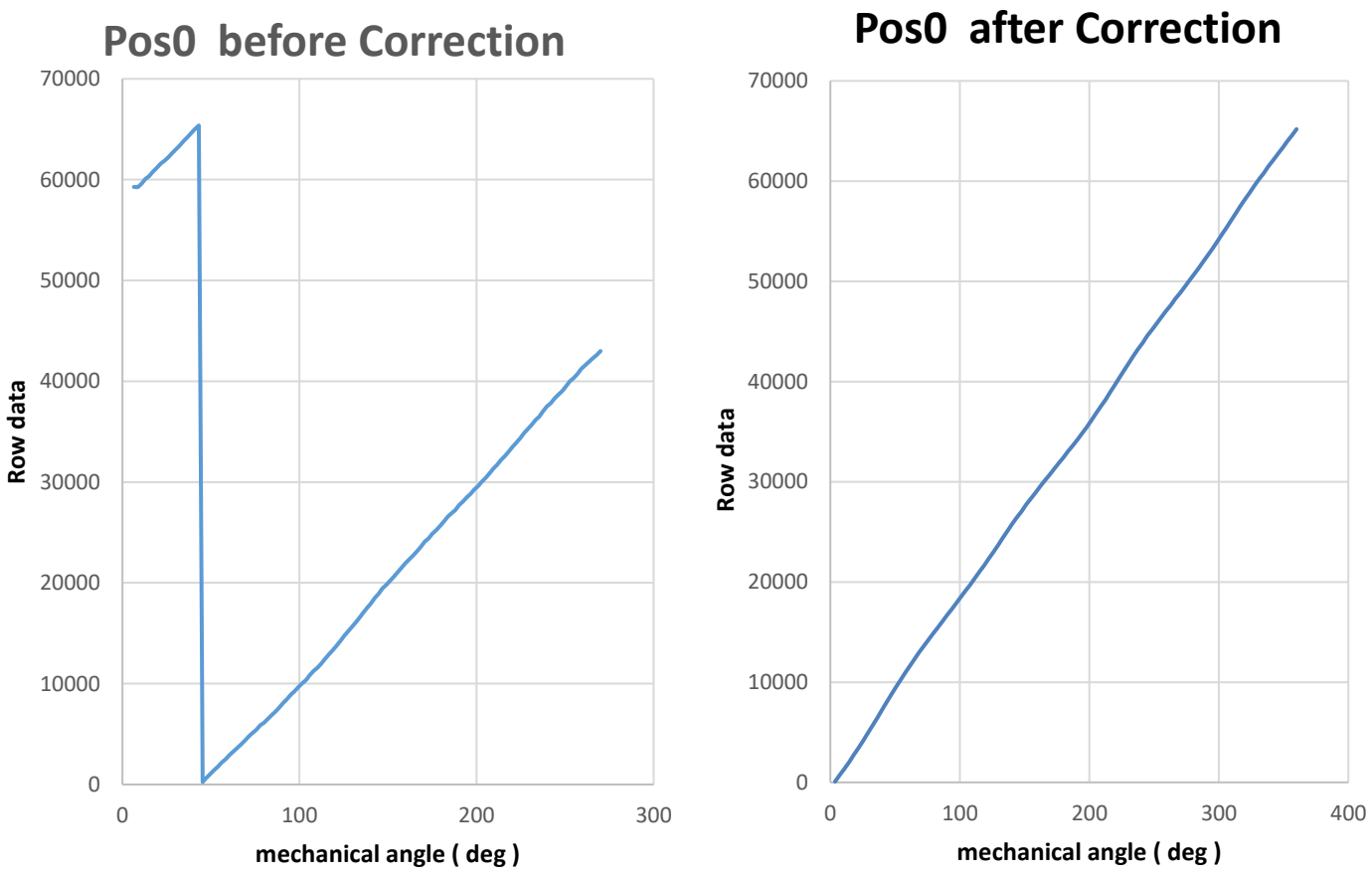
The zero offset correction is used for mapping the electric angle to the mechanical angle using the *Zero Offset* term.

Example: It is possible to design a 100° coil and to use a shorter range; e.g., 80°, 90°, 95°. The zero offset correction allows selecting the start point and the end point within this 100° range and allows moving the target in both directions (right to left or left to right).

Important: There is a limit for the designed selected range that depends on various parameters such as geometry, target length, and the maximum slope, which is four.

Figure 5 shows an example of the difference between before and after zero offset correction.

Figure 5. Example of Pos0 Values before and after Zero-Offset Correction



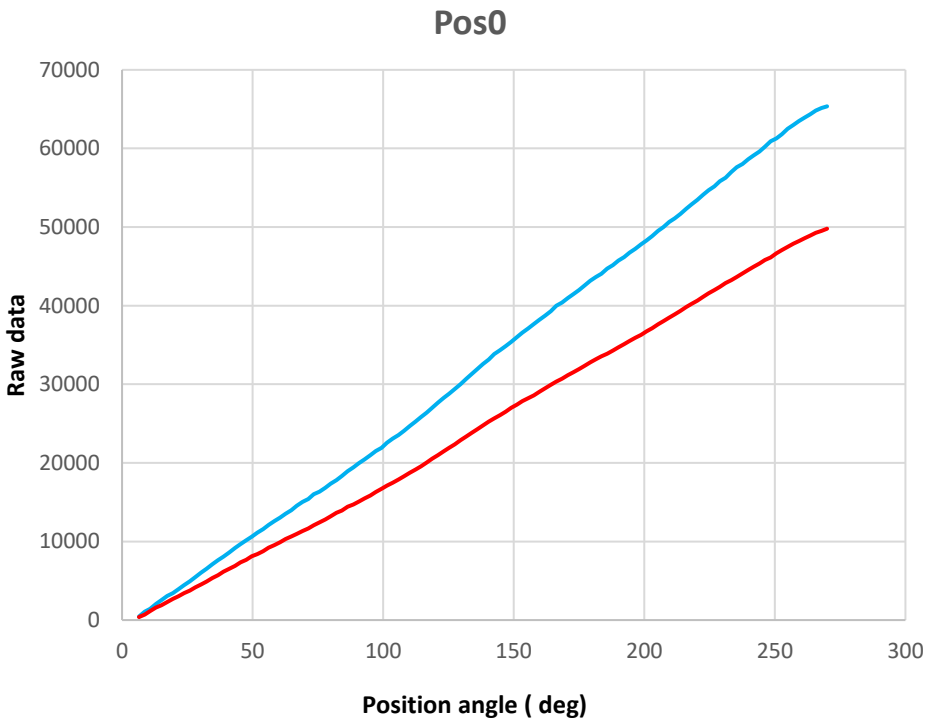
2.3.2 Slope Correction

The slope correction allows mapping the measured angle to the full-scale range for the measurement.

Example: In the graph in Figure 6, the red waveform is the mechanical movement data of the target before slope correction. It is mapped in a range from 0 to 50000_{DEC}. After slope correction, the same mechanical movement of the target is mapped to a range from 0 to 65535_{DEC} as shown by the blue waveform.

The *Slope* term is a simple multiplication factor.

Figure 6. Slope Correction Example



2.4 Coil Offset Compensation

The offset generated from the coils is called the coil offset or amplitude offset. Coil offset is the unwanted additive/subtraction voltage on the receiver's channel. The cause of this offset is the non-uniformity of the magnetic field. Ideally, the voltage measured on both receivers without any target should be zero.

In the ZMID520x, the measurement of the offset is done by reading the internal value of the sine and cosine channel without any metal target near the coils

This offset can be compensated by values programmed in the EEPROM. The maximum compensation value in the ZMID520x is $\pm 252_{DEC}$ for the raw data.

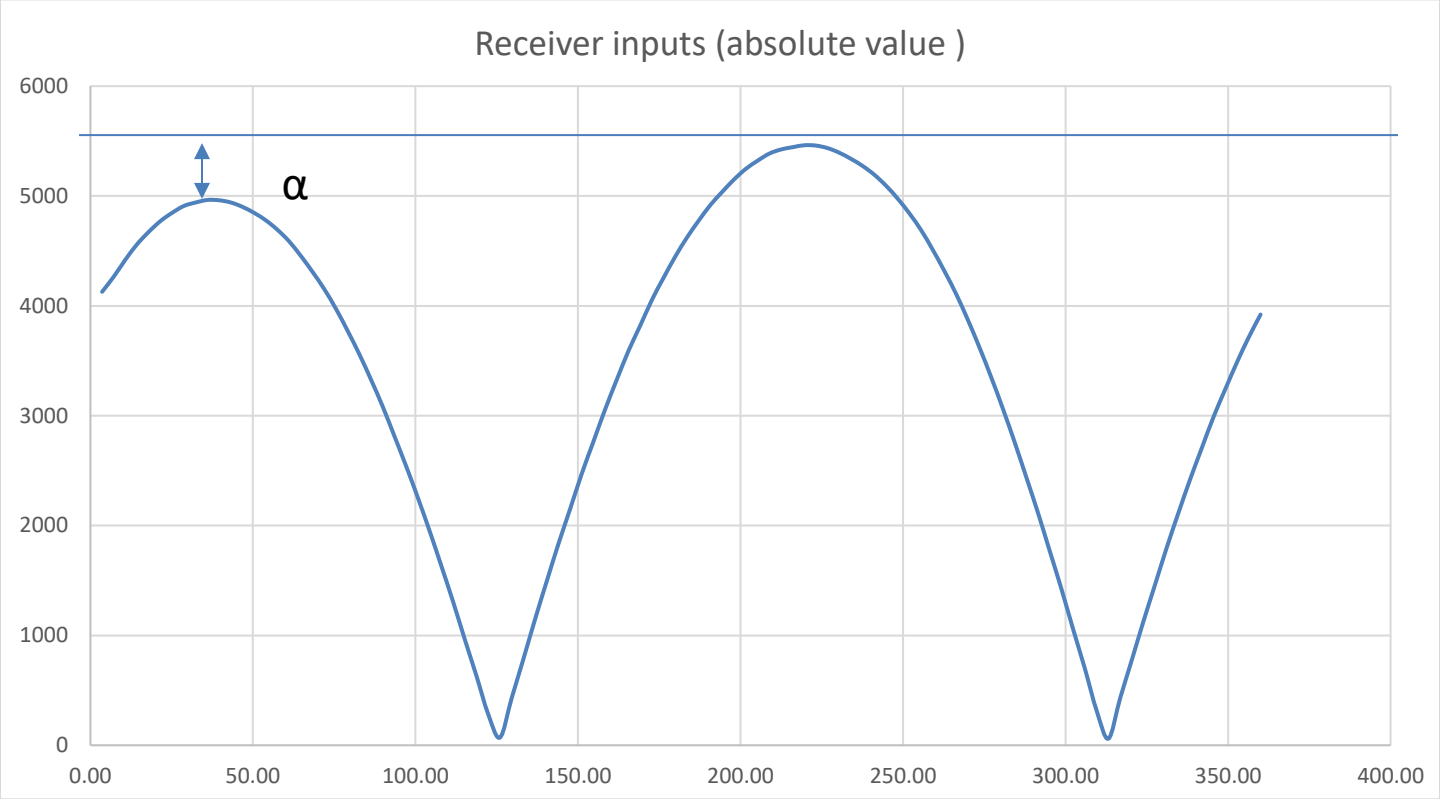
During a complete sweep of the target, the offset on the receiver is seen as a shift of the signal with respect to the x-axis.

Figure 7 shows an example where the shift is by α.

The compensation in this example should be

Offset compensation = α/2

Figure 7. Offset Compensation



3. Theory for Calibration and Linearization on the Analog Output

The general process follows this sequence:

1. Calibration
   - Zero offset
   - Slope
2. Linearization of correction points
   - Corr0
   - Corr1
   - …
   - Corr8

Important: This method of calibration and linearization is valid only for the analog output. The values are expressed in volts.

The linearization function is implemented in the digital domain, and the correction factors are applied at fixed digital points. In order to achieve the linearization, a conversion of the analog output data is required.

If the zero offset is equal to the origin (zero) and the slope is 1, the analog output (volts) is the converted digital raw data.

converted data = spatial value = Pos0 = Pos1

After the analog output measurement, it is possible to calculate the *Spa* value, which is used for calibration and linearization.

Example:

- Zero offset (address 00_{HEX}) = 2400_{HEX} for Linear Output Mode or (address 00_{HEX}) = 0000_{HEX} for Modulo360 Output Mode
- Slope (address 01_{HEX}) = 0400_{HEX}
- VDDE supply = 5V

For this case, the analog output voltage for each measured value is converted using these calculations:

Ratiometric output range = 5% of VDDE to 95% of VDDE = 0.25V to 4.75V

Full-scale voltage output range = 4.75 – 0.25 = 4.5V

This range is mapped to 0 to (2^{16} – 1):

$$converted\ data = \frac{(Volt\_meas\ -\ 0.25) * (2^{16} - 1)}{4.5\ V}$$

Where:

Volt_meas is the value measured at the analog output.

3.1 Zero Offset and Slope Calibration

Calibration consists of measurements at the two end points and determination of the zero point offset and slope:

1. Measure the actual analog output voltage when the target is in the first position (point A).
2. Measure the actual analog output voltage when the target is in the last position (point B).
3. Convert the analog output data in the *Spa* register.

   spatial_angle_pt.A is the *SPA* register value when the target is at point A.

   spatial_angle_pt.B is the *SPA* register value when the target is at point B.

```
If spatial_angle_pt.A < spatial_angle_pt.B
    spatial_angle_min = spatial_angle_pt.A
    spatial_angle_max = spatial_angle_pt.B
Else
    spatial_angle_min = spatial_angle_pt.B
    spatial_angle_max = spatial_angle_pt.A
```

4. Determine the offset of the zero point and save it to memory.

   Zero point offset:

```
offset_value = spatial_angle_min
```

   Map *offset_value* to register notation:

```
If (offset_value ≤ 2^15)
    Offset_parameter = 2^13 + 2^10 × [1 – (offset_value/2^15)]
else
    offset_parameter = 2^10 × [(offset_value/2^15) – 1]
```

   Write to EEPROM address:

```
address 0x00 = dec2hex [offset_parameter; 4 ]
```

5. Determine the slope correction factor and save it to memory.

   Slope:

```
Slope_value = 2^16 /(spatial_angle_max – spatial angle_min)
Slope_parameter = 2^10 × slope_value
```

   Write to EEPROM address:

```
address 0x01 = dec2hex[slope_parameter; 4]
```

   Optional: Change the slope polarity for the position output

```
offset_value = spatial_angle_max
slope_parameter = 2^12 + 2^10 × slope_value
```

3.2 Reference Angle Calculation

For the linearization, an ideal reference angle must be generated. The values of an ideal reference angle are defined as an array in the range from 0 to 65535 (*Pos0,* 16 bits). This array contains a given number of elements and a specific step between two adjacent values of the array.

Ideal Data Acquisition: The acquisition of data is done at equidistant time intervals, and the target is moved with constant velocity.

Table 2. Example of Ideal Data

Index	Ideal Pos0 Value
0	0
1	154.5637
2	309.1274
3	463.6911
4	618.2548
5	772.8185
6	927.3822
7	1081.9459
8	1236.5096
9	1391.0733
10	1545.637
11	1700.2007
12	1854.7644
13	2009.3281
14	2163.8918
15	2318.4555
16	2473.0192
17	2627.5829
18	2782.1466
19	2936.7103
20	3091.274
21	3245.8377
22	3400.4014
23	3554.9651
24	3709.5288
25	3864.0925
26	4018.6562
27	4173.2199

Definitions:

Step_value_Pos0: Step value between two adjacent values.

Pos0_ideal_elem_nr: The number of elements present in the ideal reference angle.

Spat_ele_nr: The number of samples that are read from the *Spatialangle* register during a complete sweep of the target.

Pos0_ideal_elem_nr is equal to *Spat_ele_nr.*

Step_value_Pos0 is calculated as *Step_value_Pos0* = 65535 / *Pos0_ideal_elem_nr*

Ideal_Pos0_Value is an array that is populated as follows:

```
index = 0
While (index < Pos0_ideal_elem_nr)
{
        Ideal_Pos0_Value[index] = index × Step_value_Pos0
        Index ++
}
```

3.3 Determining the Correction Factor for Linearization

Table 2 provides an example of the calculations for correction factors when *Pos0_ideal_elem_nr* is 383.

In Table 3, the "OutputCalPOS0" column contains the analog output converted value from a complete sweep of the target.

The "Difference" column value is calculated as *Ideal_pos0_Value[Index_calc] – OutputCalPOS0[Index_calc].* For the "Difference/32" column, 32 is a scaling factor inherent to the ZMID5201.

Table 3. Correction Factor Calculation

Index_calc	OutputCalPOS0	Ideal_pos0_Value	Difference	Difference/32	Difference/32 [Hexadecimal]
0	3277	3245			
1	3289	3400			
...	...	...	...	...	...
32	6775	8191.875	1417	44.26	Corr1 = 2C
...	...	...	...	...	...
85	14094	16383.75	2290	71.553	Corr2 = 47
...	...	...	...	...	...
...	...	...	...	...	...
...	...	...	...	...	...
381	62172				
382	62258				

The correction factor and the respective fixed point values are given in Table 1.

Linearization algorithm for Corr0 to Corr8:

Corr1 factor:

1. In the *Ideal_pos0_Value* array, find the closest value to 8192. (See Table 3.)
2. Calculate the difference between the found value and the value of the array *OutputCalPOS0* that has the same index.
3. *Corr1* = difference / 32
4. Convert the data to hexadecimal and write it to EEPROM.

Repeat the four steps for each remaining correction factor (*Corr0, Corr2, Corr3, … Corr8*).

Example for Corr2:

1. In the *Ideal_Pos0_Value* array, the element with the value 16384 is found and the *Index_calc* = 85 (see Table 3.)
2. In Table 3, the value of the array *OutputCalPOS0* at index = 85 is equal to 14094.
3. The difference between these two values divided by 32 is the correction factor for *Corr2*.

   (16383.75 – 14094) / 32 = 71.553
4. Convert the data to hexadecimal and write it to EEPROM.

4. Clamping Analog Output for the ZMID5201

The ZMID520x has a ratiometric analog output; i.e., the span of the electrical output signal is proportional to the excitation voltage applied, which depends on the supply voltage. The output is expressed in percentage with respect to VDDE. The maximum and minimum for the range of the output with respect to VDDE can be varied by clamping; i.e., maintaining the upper and lower limits of the electrical output signal at prescribed values.

4.1 The Relationship between the Internal Value and the Output

The internal DAC of the ZMID520x converts the internal register values (*Pos1*) into analog output data. Only 90% of VDDE is used for the analog output, so the full-scale (FS) range of the internal values is mapped to 90% of the VDDE range for the output.

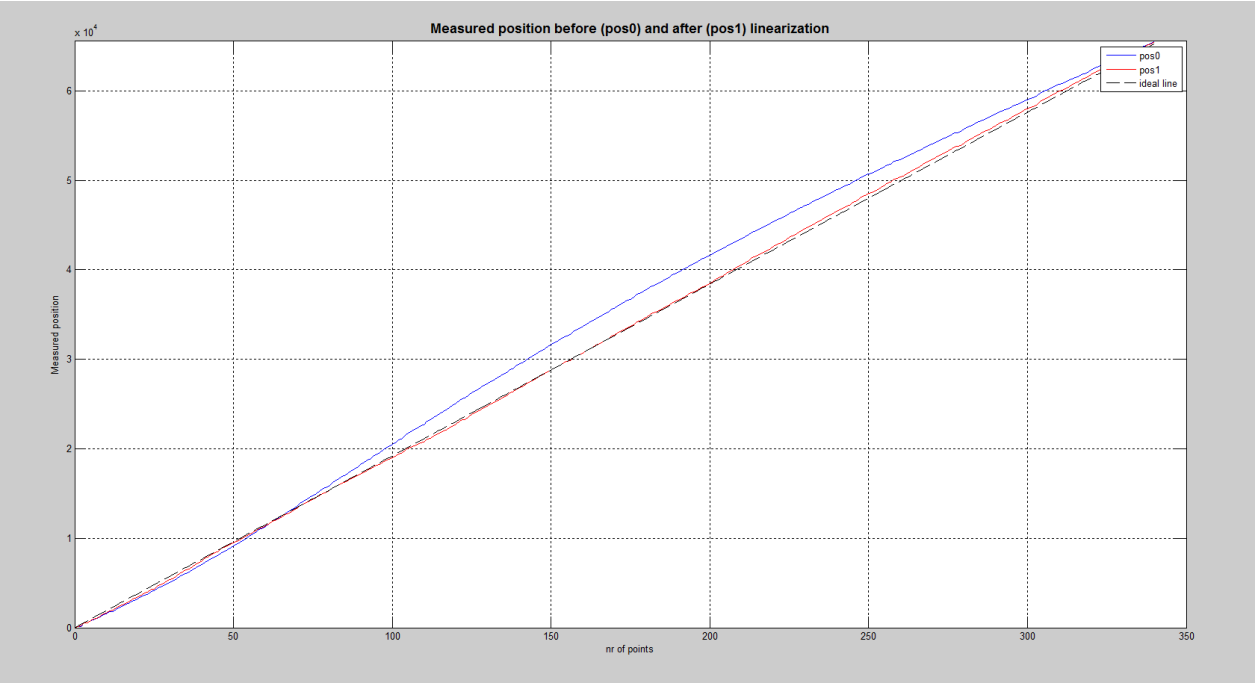
The minimum value for the *Pos1* register is zero, and the maximum is ($2^{16} - 1$).

Note: The minimum and maximum output limits will be respectively 5% to 95% of VDDE when there is no clamping. The output values are proportional to the internal values. When the internal range is compressed to a smaller range, the output range will also be decreased.

Clamping allows varying the output range without changing the target travel.

Figure 8. Pos1 Output Mapped to 5% to 95% VDDE (No Clamping)

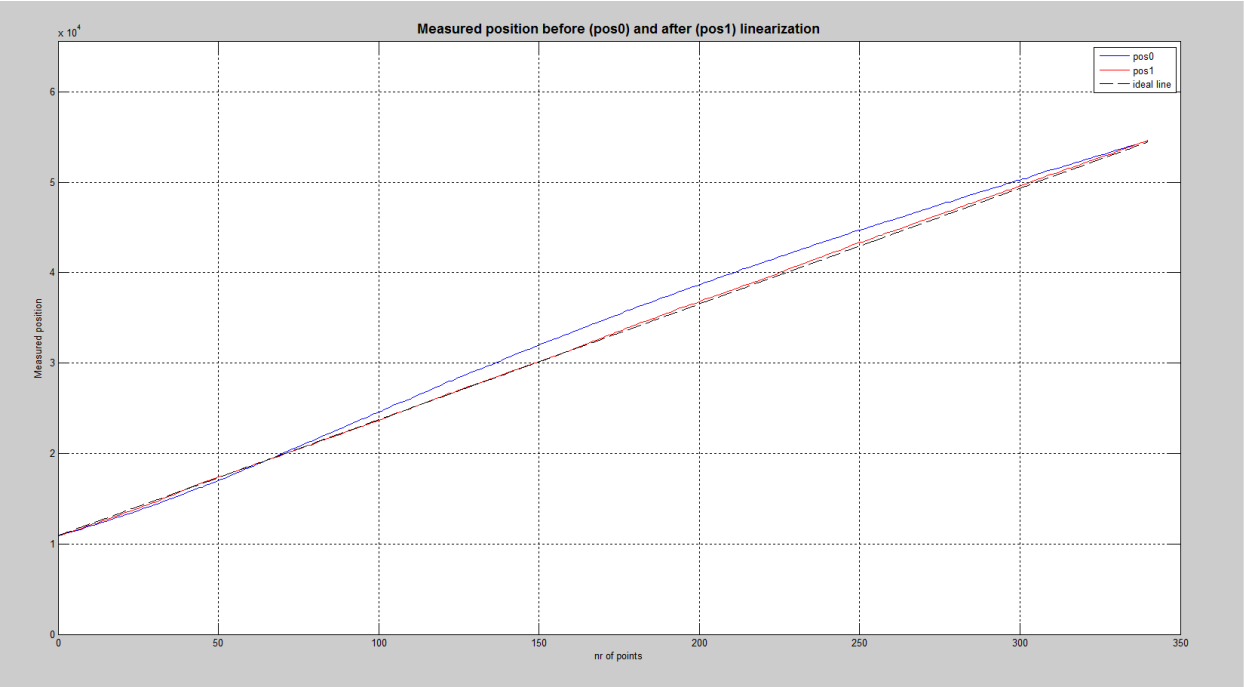
Note: See section 4.2.2 for a description of this type of clamping



Clamping is used in applications that require an output range that is different from the default of 5% to 95% VDDE. An example is shown in Figure 9. The decision to use clamping should be guided by the customer requirements or the electrical limits of the application/sensor.

Figure 9. Clamping in an Output of 20% to 80%VDDE and Clamping in *Pos1* to 16.667% to 83.33% FS

Note: See section 4.2.2 for a description of this type of clamping.



4.2 Applying the Clamping Process

Clamping in the ZMID520x is implemented by two different processes. Both are mandatory in the following order for correct clamping.

1. Calibration process by tuning the calibration parameters (see section 4.2.1)
2. Register clamping by configuring bits in EEPROM address 02_{HEX} (see section 4.2.2)

4.2.1 Calibration Process

The clamping by the calibration process is calculated based on the correlation between the internal value (*Pos1*) and the analog output. The *Pos1* register value represents the measured position as digital-format raw data.

The calibration process directly affects the internal raw data and indirectly affects the output clamping by limiting the minimum and maximum data in the *Pos1* register, which will also limit the output range.

The process for clamping by calibration determines a slope and zero offset that result in a specified minimum and maximum output for a given target movement.

4.2.2 Register Clamping

Register clamping can limit the output based on the VDDE reference regardless of where the target is positioned or how the slope and zero offset are defined. Register clamping uses only the VDDE of the ZMID5201 as the reference, without any link to the internal values.

An example of register clamping is shown In Figure 8 where the 0% to 100 % FS range for *Pos1* is correlated to 5% to 95% of the output. Another clamping example is given in Figure 9 where the 16.667% to 83.33 % FS range for *Pos1* is correlated to 20% to 80% of the output.

4.3 Proper Clamping Requirements

For proper clamping, both the calibration and register clamping must be applied. Clamping only by the calibration process is inadequate because whenever the target is moved out of the operation range (start to end), the calibration mapping will no longer be valid. The output value will exceed the limit defined by the calibration process mapping.

Clamping only by register clamping will decrease not only the output range but the measurement range as well.

The clamping requirements for the analog output can be satisfied only by applying the two different clamping methods simultaneously in the following sequence.

1. Clamping by the calibration process:
   a. *Clamp_low_calib* is defined by the calibration process
   b. *Clamp_high_calib* is defined by the calibration process
2. Clamping by register configuration:
   a. *Reg_low_value:* Value to be written in EEPROM register 02_{HEX}
   b. *Reg_high_value*: Value to be written in EEPROM register 02_{HEX}

The calculations for both these processes are automatically performed by the *OneStepCalibration* algorithm.

4.4 Clamping via Calibration

In the *OneStepCalibration* algorithm, the clamping is defined as follows:

Clamp_low = *Clamp_low_value* • % • VDDE

Clamp_low_value (no units)

Clamp_high = *Clamp_high_value* • % • VDDE

Clamp_high_value (no units)

The inputs for the *OneStepCalibration* algorithm are *Clamp_low_value* and *Clamp_high_value.* The outputs of the algorithm are the calibration parameters, Slope and Zero Offset.

The movement range of the target (start to end point) is verified as *Clamp_low_value* to *Clamp_high_value.*

Example:

VDDE = 5V

Clamp_low = 10% VDDE = 0.5 Volt

When the target position is at the start point, the output value will be 0.5 volt.

Clamp_high = 90% VDDE = 4.5 Volt

When the target position is at the end point the output, value will be 4.5 volt.

4.5 Clamping via Register Clamping

Clamping using the register is just a clamp of the output with respect to VDDE.

For register clamping, the inputs for the *OneStepCalibration* algorithm are *Clamp_low_value* and *Clamp_high_value*. The outputs of the algorithm are *Reg_low_value* and *Reg_high_value.*

4.5.1 Register Clamping Low

The user defines the clamping with respect to VDDE.

Clamp_low = *Clamp_low_value* • % • *VDDE*

Clamp_low_value = input value (no units).

$$Reg\_low\_value = \text{floor}\left(\frac{Clamp\_low\_value - 5}{0.9}\right) \text{ (in decimal)}$$

Where the function “floor” rounds the number to the next smaller integer.

Reg_low_value must be written to bits [7:0] in EEPROM register 02_{HEX}.

4.5.2 Register Clamping High

The user defines the clamping with respect to VDDE.

Clamp_high = *Clamp_high_value* • % • *VDDE* (This is the same for both register clamping and calibration mapping.)

Clamp_high_value = input value (no units).

$$Reg\_high\_value = \text{floor}\left(\frac{95 - Clamp\_high\_value}{0.9}\right) \text{ (in decimal)}$$

Reg_high_value must be written to the bits [15:8] in EEPROM register 02_{HEX}.

4.6 Calibration Clamping vs. Register Clamping

Important: Calibration clamping and register clamping have a different resolution step.

Register clamping has a clamping step of 0.9% of VDDE.

The step value with register clamping can be calculated as follows:

Clamp_low_reg = 5% • *VDDE* + 0.9% • *VDDE* • *Reg_low_value* (The low clamping output expressed in volts when *Reg_low_value* is written in bits [7:0] in EPROM register 02_{HEX} (integer).)

Clamp_high_reg = 95% • *VDDE* – 0.9% • *VDDE* • *Reg_high_value* (The high clamping output expressed in volts when *Reg_high_value* is written in bits[15:8] in EPROM register 02_{HEX} (integer).)

Calibration clamping has about x10 more resolution than register clamping and this leads to an imperfect match for these two types of calibration.

Example:

VDDE = 5V

Clamp_low = 10% VDDE = 0.5V	From specification
Clamp_low_value = 10	Definition
Reg_low_value = 5	Calculated by the equation given in section 4.5.1
Clamp_low_reg = 0.475V	Calculated by the equation above

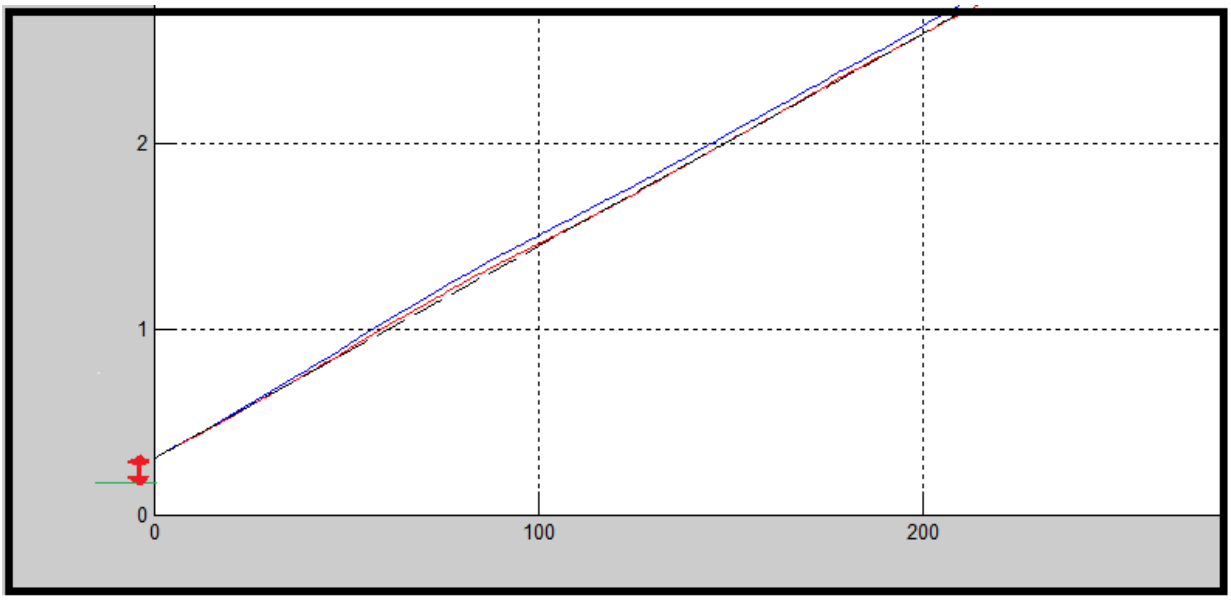
The difference between the calibration clamping and register clamping:

Difference = *Clamp_low* – *Clamp_low_reg* = 0.5 – 0.475 = 0.025V (see Figure 10)

The tolerance is 25mV, which is 0.555% FS where FS = 4.5V.

Figure 10. Difference between Calibration and Register Clamping

Note: In the following graph, the blue line indicates Pos0 data; the red line indicates Pos1 data; and the dashed line is the ideal measurement.



5. Requirements for Measurement of SPA and Analog Output When Using the *OneStepCalibration* Algorithm

Measuring the position during one sweep of the target is necessary for calibration and linearization. There are requirements for a correct measurement process.

The calibration and linearization process requires a mechanical test bench for executing a controlled and accurate movement of the target. The algorithm behind the process is implemented under the hypothesis of an ideal mechanical system. The quality of the calibration and linearization is directly related to the quality of the reference mechanical movement. The nonlinearity in the measured position is only associated with the sensor.

In practice, a motor/step motor is needed to execute the movement of the target under specific conditions including the following:

1. A specific air gap (distance between the coils and target)
2. Linear movement
3. Rotary movement

Synchronization: The acquisition of the SPA, analog voltage, and the target movement should be synchronized.

The starting point and end point are defined by the application.

The target should move from the starting point to the end point. After calibration, the start point will be the zero position and the end point will be the full scale position.

The measured, electrical data is linearly mapped to the mechanical movement.

An increase in the target travel range will decrease the resolution, as the same electrical output range will be mapped to a bigger movement range of the target.

In practice, there are two alternative ways to accomplish the measurement needed for calibration and linearization.

1. Synchronization between the target movement and the acquisition: The target should move in equidistant steps, and the acquisition should occur as soon as the target is stopped between each movement step.
2. Moving the target with a constant speed from the start point to the end and simultaneously performing the data acquisition with a constant speed. The acquired data should be considered linear to the target movement.

These two different acquisition methods are not dependent on the type of the acquisition output.

The number of acquisitioned samples in one target sweep should be about 300. Note that this is an approximate number; a larger number will result in better performance of the calibration and linearization. The GUI accepts a maximum of 1500 samples when uploading the .csv file that is used to transfer measurement data to the GUI. The calibration and linearization will proceed automatically.

5.1 EEPROM Configuration before Performing the Measurement

Important: General configuration is needed before the measurement including for the gain stage, post calibration, amplitude offset, and output mode.

Gain Stage

For a specific air gap, which is defined by the application, read the CORDIC Magnitude register when the target is on the top of the coils between the start point and the end point. Theoretically, the magnitude will be constant during the sweep of the target as long as the target is on the top of the receiver coils.

The optimal value should be 7000 to 10000 at room temperature. Define the gain stage to an appropriate stage to satisfy the optimal CORDIC Magnitude. (The range of the gain stage is 1 to 12)

Post Calibration

Strong recommendation: Use post calibration; i.e., apply the linearization after calibration.

Amplitude Offset or Coil Offset

The offset value should be zero if an ideal coil has been designed. In practice there is always some offset, and this can be compensated by a bit configuration in EEPROM.

Recommendation: Measure the offset by reading the internal value (sine and cosine channel) without the target after the gain stage has been defined.

Output Mode

There are the following two output mode:

- Linear Output mode: applicable in arc and linear sensors. Possible jumps in the spatial angle must be eliminated by a bit configuration change.
- Mode 360: only applicable in rotary 360-degree movement or with multi-period sensors. Do not use it for an angle narrower than 360°. If there is any mechanical tolerance in the target position at the start point or at the end point, a big jump in output from minimum to maximum will cause an error of 100%.

The Clamping Coefficient Configuration before Measuring

The clamping coefficient configuration should be zero before the measurement when the analog output is used. This has no influence on the internal value of the SPA.

The Calibration and Linearization Coefficient Configuration before Measuring

OWI: When the SPA register is read, ignore the bit configuration.

6. *OneStepCalibration* Automatic Calibration and Linearization

OneStepCalibration is a fast algorithm for calculating the calibration and linearization coefficient. The algorithm is very easily implemented.

For input, *OneStepCalibration* needs the spatial angle measurement during one sweep of the target and some basic configuration values including sign slope, output mode, reverse slope, and clamping output.

Figure 11. Inputs and Outputs for *OneStepCalibration*

The same algorithm can be used when the measurement data is analog

The *OneStepCalibration* algorithm should be used for performance validation and for end-of-line calibration.

Figure 12. General Flow Diagram for Using Internal Values for Calibration and Linearization

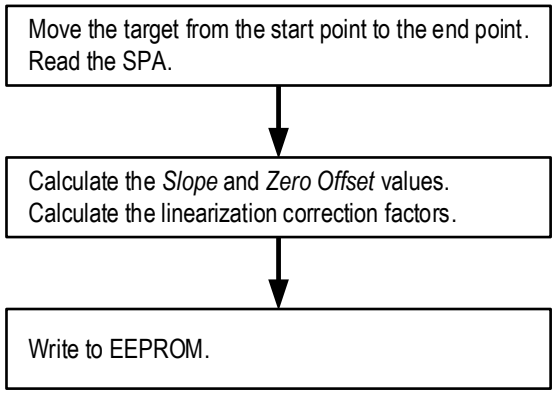
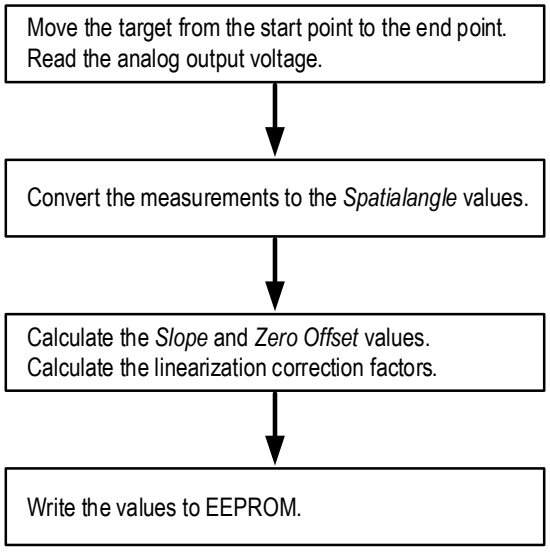


Figure 13. Flow Diagram When the Analog Output is used for Calibration and the Linearization



7. Validation: Error Calculation after Calibration and Linearization

After calibration and linearization, another measurement is needed to validate that the overall process has been performed correctly. The measurement should be performed in the same manner with the same output used for the calibration and linearization measurement. Any change in the target position (air gap, start point, end point) in this procedure could lead to an incorrect validation.

Once the measurement has been performed and the data is available, the error can be calculated as follows:

Elem = 0 :1: (*Nr_elem* – 1) *Matlab code*

Where *Nr_elem* is the number of the position acquisition during the sweep of the target.

Difference between Pos1 and the ideal line mapped to the FS in percentage:

FNL_elm = 100 • (*list_Pos1*[*elem*] - *Ideal_list*[*elem*]) / *list_Pos1*[*Nr_elem – 1*] //

Where

FNL is the final nonlinearity.

list_Pos1[*elem*] is the array with data from reading the *Pos1* register during the sweep of the target.

Ideal_list[*elem*] is an array with the same number of elements as *Pos1*; all the elements are part of the ideal straight line that is defined by the first element of the Pos1 and last element of the Pos1.

For analog the procedure is the same.

In the respective formula, use the output value instead of *Pos1[elem].*

Difference between Analog value and the Ideal line mapped to the FS in percentage

FNL_elm = 100 • (*list_Analog*[*elem*] – *Ideal_list*[*elem*]) / *list_Analog*[*Nr_elem* – 1] //

8. Revision History

Revision Date	Description of Change
October 10, 2019	Full revision.
March 7, 2018	Initial release.