Contents

List of Figures

List of Tables

1. Introduction

This document describes how to set up and use the Dynamic Link Library (DLL) combined with a Python® script* to perform calibration and linearization on the ZMID520x Inductive Position Sensor IC. After running the script successfully, it will produce the new coefficients to write into the memory of the chip.

1.1 User Requirements

- Windows® 7 or Windows® 10
- Recommended development environment: Anaconda® 5.2.0† with Spyder 3.6, 32-bit or 64-bit
- Development environment that supports Python 2.7, 3.4, 3.5, or 3.6

2. Getting Started

1. Navigate to the web page for the applicable product:

   ZMID5201: www.IDT.com/zmid5201

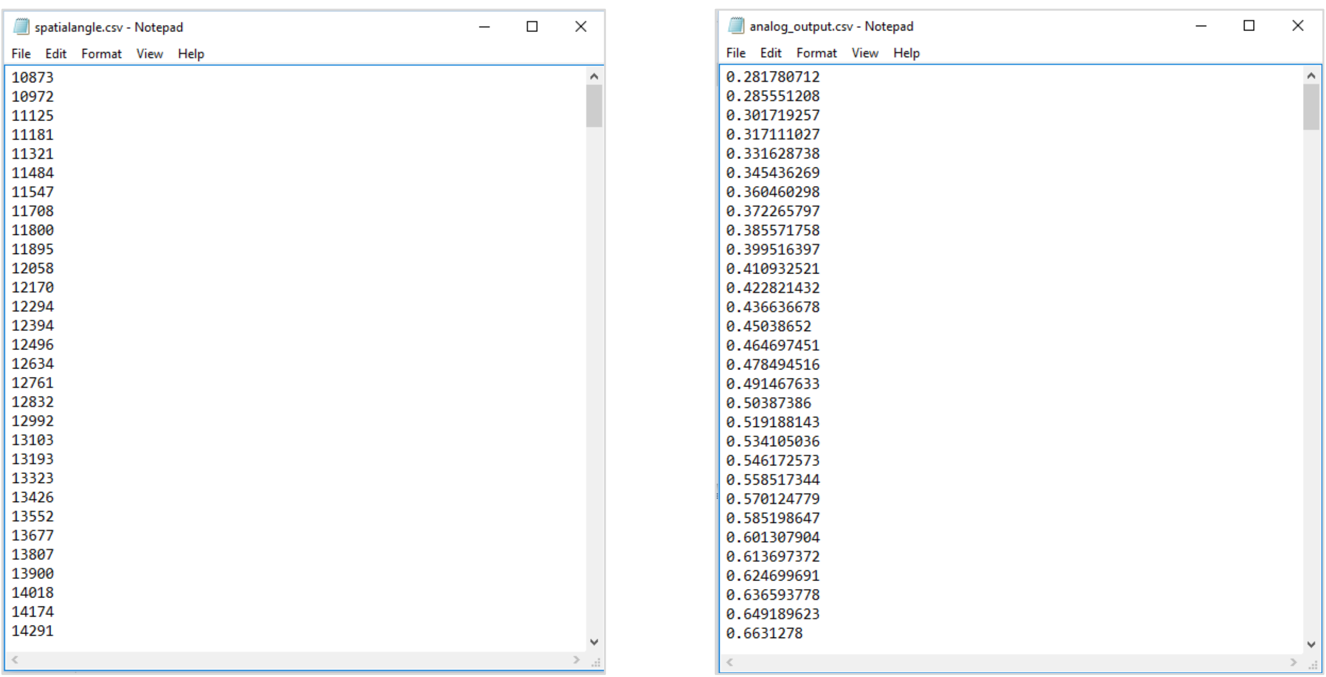
   ZMID5202: www.IDT.com/zmid5202

   ZMID5203: www.IDT.com/zmid5203

2. Under "Software and Tools," download the zip files for the DLL and the Python script. Extract the contents of the folder after downloading has completed. The DLL and the script must be in the same folder when the script is running.

3. A CSV file containing the spatial angle points or the analog output points is required as an input to run the script. Save the CSV file in the same folder as the Python script and the DLL. Figure 1 shows examples of spatial angle and analog values demonstrating the format that must be used. If the values have decimal points, a period must be used as the decimal separator (not a comma).

Figure 1. Example Content for Spatial Angle and Analog Output Values in CSV Files



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

† Anaconda® is a trademark of Anaconda, Inc.

3. Parameters

1. From the development environment, open *ZMID520x_calibration_and_linearization.py*.
2. There are ten parameters found at the beginning of the script, which are the user inputs as illustrated in the example given in Figure 2. Table 1 defines the parameters to set.
3. Run the script after the parameters have been set correctly. The result will appear in the console window.
4. Write the new coefficients in the respective memory in the chip. One method is via the *ZMID520x EVK Application Software*, which is a graphical user interface (GUI) provided on the IDT website.

Figure 2. Example of the Python Script

```
DLL = ctypes.WinDLL (r'C:\OneStepDLL\ZMID520X_OneStepCalibration_32.dll')
filename = 'spatialangle.csv'
spa_or_analog_input = 0
zmid_type = 1
position_slope = 0
reverse_slope = 0
out_mod = 0
clamping_percent_low = 5
clamping_percent_high = 95
Vdd_value = 4.994
```

Table 1. Parameters

Parameter	Description
DLL	Full path of the location of the DLL, including the filename with ".dll" at the end. Note: Write the path inside the parentheses. Keep "ctypes.WinDLL" as it is.
filename	Name of the file containing the spatial angle/analog values, including *.csv* at the end.
spa_or_analog_input	This parameter specifies which values the input file contains: 0 = Input file with spatial angle values 1 = Input file with analog values
zmid_type	Select the product: 1 = ZMID5201; analog output 2 = ZMID5202; PWM output 3 = ZMID5203; SENT output
position_slope	Position_slope defines the slope of the spatial angle values: 0 = Positive 1 = Negative
reverse_slope	This applies to the intended output of Position0 where 0 = Keep the slope 1 = Invert the slope

Parameter	Description
out_mod	Output mode of the chip: 0 = Linear Output Mode 1 = Modulo360 Output Mode
percent_input_low	Lower clamping in percentage. 5 = no clamping.
percent_input_high	Upper clamping in percentage. 95 = no clamping.
Vdd_value	Supply voltage when using analog output.

4. Example of Usage

The following lines are an example of completing the parameters using a ZMID5201 (analog output) and a positive spatial angle as input. In this example, the output mode is set to linear, the slope is not inverted, and clamping is set to 10% and 90%. The supply voltage is set to 5V; however in this example, it will not affect the calculation because the input file is the spatial angle. For all parameters used in this document, refer to the user guide for calibration and linearization.

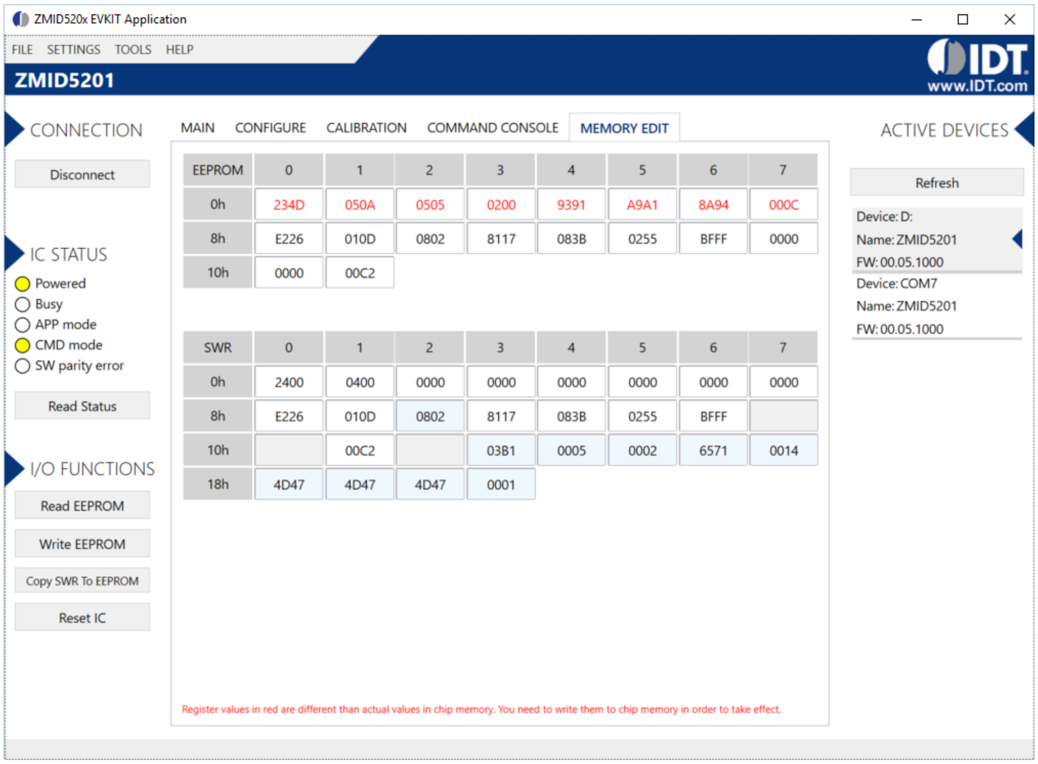
```
DLL = ctypes.WinDLL(r'C:\Users\Desktop\OneStepDLL\ZMID520X_OneStepCalibration_32.dll')
filename = 'spatialangle.csv'
spa_or_analog_input = 0
zmid_type = 1
position_slope = 0
reverse_slope = 0
out_mod = 0
clamping_percent_low = 10
clamping_percent_high = 90
Vdd_value = 5
```

After the parameters are entered and the script has run without errors, the new coefficients for the memory will appear in the Python console. The following lines are an example of what the output can look like when the script has run successfully.

Write to EEPROM:
Address 0x00: 234D
Address 0x01: 050A
Address 0x02: 0505
Address 0x03: 0200
Address 0x04: 9391
Address 0x05: A9A1
Address 0x06: 8A94
Address 0x07: 000C

The new coefficients can be written in the entry fields provided on the “MEMORY EDIT” tab in the GUI and saved in the ZMID520x memory by clicking the “Write EEPROM” button.

Figure 3. Example of Writing Coefficients Using the "MEMORY EDIT" Tab



5. Error Codes

Table 2 describes possible DLL error codes. A code of 0 indicates that the run was successful.

Table 2. Error Codes

Error Code	Error Name	Description
Error code -1	ERR_INVALID_INPUT	One or more of the inputs are invalid.
Error code -2	ERR_SLOPE	The calculated slope is out of range.
Error code -3	ERR_OFFSET	The calculated offset is out of range.
Error code -4	ERR_CORR	One or more of the calculated correction values are out of range.
Error code -5	ERR_POS0	One or more of the calculated Pos0 values are out of range.
Error code -6	ERR_POS1	One or more of the calculated Pos1 values are out of range.

6. Revision History

Revision Date	Description of Change
August 1, 2018	Initial release

IMPORTANT NOTICE AND DISCLAIMER

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.