Encoder error correction tables

Introduction

The quality of positioning control that a servo drive may achieve is limited by the resolution and accuracy of the position data provided by the encoder. For some high accuracy applications, it may be desirable to measure the encoder errors and provide a table of compensation values that the drive can use to correct for these errors in real time. Starting with firmware version 2.24, Copley feature set E drives (aka *plus* drives) support the use of such tables to compensate for encoder errors.

Table format

The encoder table is a binary formatted file that is uploaded to the drive's internal file system. This file consists of a short header section followed by a list of 16-bit offsets that will be added to the position information read from the encoder. The resulting values (position from encoder plus offset) will be used in place of the position data read from the encoder. This allows for known errors in the encoder position data to be compensated for by the drive.

The file header consists of 16 bytes which make up 4 32-bit words. These words are stored in the file in little endian order, and have the following meaning:

Word	Contents	Description
0	File revision	Identifies the file version. Should be set to 1.
1	Position increment	Number of input encoder counts / file entry.
2	First position	First encoder position to compensate.
3	Wrap value	Wrap the input position every N counts.

This header is followed by a list of 16-bit adjustment values. These adjustment values are stored as two bytes each, least significant byte first.

The maximum number of adjustment values that can be stored in a single encoder compensation file is limited to 2048 points on a single axis drive. For a dual axis drive, each axis can store a maximum of 1024 adjustment points.

When the encoder compensation table is uploaded to the drive, it must be given a special name to allow the drive firmware to identify it as an encoder adjustment table. For single axis drives, the table should be named “ENCADJ”. For multi-axis drives, the first axis' table should be named “ENCADJ0” and the second axis' table should be named “ENCADJ1”. If the encoder adjustment table specific for an axis number is not found, then the more generic “ENCADJ” file will be used if it exists. This allows the same table to be used for multiple axes on a multi-axis drive.

Table lookup method

When an encoder adjustment table is in use, the drive firmware adjusts the positions from the encoder as follows.

First, the position from the encoder is wrapped based on the wrap value in the file header if this wrap value is non-zero. For example, if the input position was 12345 and the wrap value was 10000, then the position after wrap would be 2345.

Next, the first position value from the file header is subtracted from this position.

Third, the resulting value is divided by the position increment from the file header resulting in an index into the offset data in the file. If the index is less than zero or greater than the length of the file data, then no adjustment is made.

Finally, the position adjustment is linearly interpolated from the contents of the file and the result is added to the original position from the encoder.

Typically, rotary encoders use a wrap value that is equal to the number of encoder counts / encoder revolution and a zero first position value. Linear encoders set the wrap value to zero and set the first position value to the lowest position that could be read from the encoder.

Drive configuration

In addition to the encoder adjustment table, a drive parameter must be set to enable the use of the table. This drive parameter is serial port parameter 0x13E. It can also be accessed as CANopen / EtherCAT object number 0x222A.

The encoder adjustment configuration parameter is bit-mapped as follows:

Bit(s)	Description
0	Set to 1 to enable the use of the encoder adjustment table.
1-2	These bits define the type of table being used: 0 - Standard encoder position compensation 1 - Resolver compensation table 2 - Cog compensation using position as input 3 - Cog compensation using phase angle as input
3-31	Reserved for future use. Set to zero.

CME *.cce file format

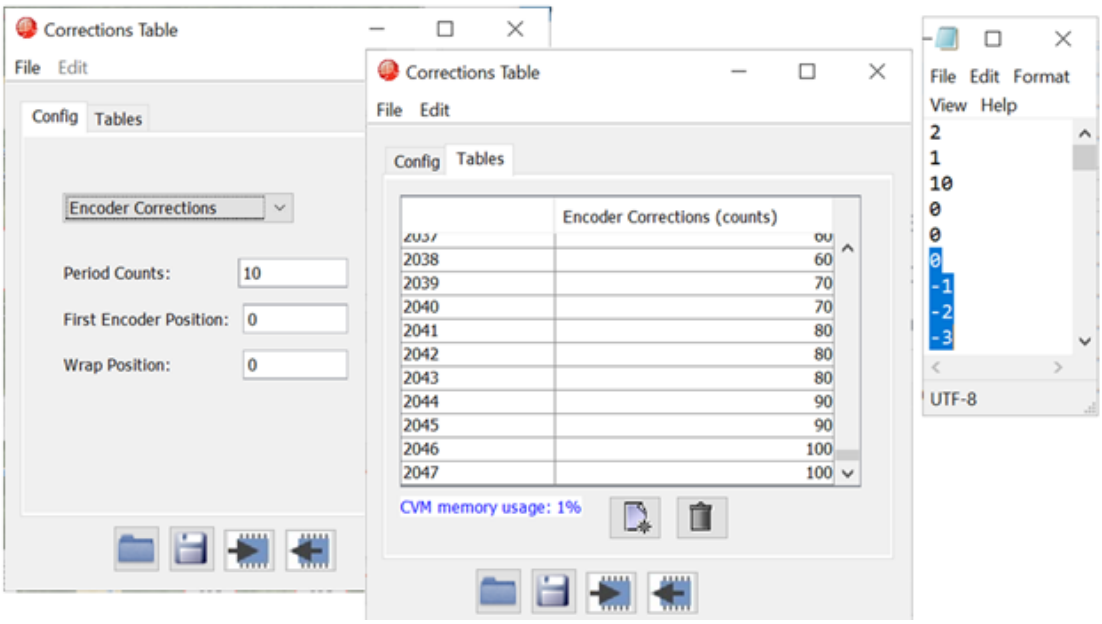
CME2 supports a *.cce file format that contains encoder correction information. This file type is structured similarly to the encoder corrections table format mentioned earlier in this document.

Rev1 Word	Rev2 Word	Contents	Description
0	0	File revision	Identifies the file version. Set to either 1 or 2.
N/A	1	Config options	Decimal value of drive configuration value. Only used in file version

			2 and above.
1	2	Position increment	Number of input encoder counts / file entry.
2	3	First position	First encoder position to compensate.
3	4	Wrap value	Wrap the input position every N counts.

It should be noted that this differs from the encoder corrections table binary format used by the drive despite its structural similarity. The *.cce file format is only used by CME2 software and for readability of encoder correction information. Values in *.cce file format are converted to decimal notation.

An example of a *.cce file being used in CME is provided below, along with the file itself in text format.



As can be seen from the header of the *.cce file, its first line specifies that the file version is rev2. Its config option bit is set, so the corrections table contains resolver angle adjustment information. The position increment of the table entries is set to 10 (see "Period Counts" field in the GUI), and its first position and wrap value words are set to 0. Decimal equivalents of the 16-bit adjustment values are listed after the header and are highlighted in blue in the text file.

Homing requirements

The use of the encoder compensation table assumes that the absolute position of the encoder is known. For this reason, the encoder adjustment will only be performed after the drive has had it's homing reference set.

For incremental encoders this requires a homing routine of some sort to be performed. For absolute encoders the homing can be done automatically as soon as the absolution position is read from the encoder. To enable this automatic homing, select 'Absolute encoder immediate home' in the CME2 homing configuration dialog box.

Special resolver compensation mode

Starting with firmware version 2.60, a special method of compensating resolver errors has been added to the drive firmware. This feature uses an encoder adjustment table but compensates the resolver angle rather than the encoder position.

The resolver angle is calculated as atan2(sin, cos) and scaled so that a value of 32768 corresponds to 180 degrees. When resolver angle compensation is enabled, this angle is used to look up an adjustment in exactly the same way that an encoder count would be used for encoder compensation. The resulting offset is then added to the resolver angle.

When using resolver angle compensation, the first position value in the file header should be set to zero. Also, the wrap value should be set to zero as resolver angles automatically wrap between 0 and 360 degrees (0 to 64k).

When resolver angle compensation is enabled, it will be used immediately. Homing is not required for resolver angle compensation.

Motor cogging compensation

Starting with firmware version 2.98, the encoder adjustment table can be used to compensate for motor cogging. In this mode the table consists of current values (in milliamp units) rather than position offsets.

Once the drive has been referenced, the position of the motor encoder is used to look up a position dependent current value in this table. That current is added to the input of the drive's current loop.

Motor cog compensation using phase angle

Version 4.30 firmware for the plus family of drives added another motor cogging compensation mode. This is similar to the previous cog compensation mode described above, but the motor phase angle is used as an input to the table rather than encoder position.

The phase angle is scaled as a 16-bit integer, so 32768 is 180 deg.

When using this mode, it’s not necessary to home the motor before the table may be used.

Revision History

Date	Version	Revision
01/17/2023	Rev 00	Initial Release