Firmware migration guide v4.x to v5.0

Technical Brief

This document describes the changes between v4 and v5 series of firmware releases and offers some suggestions for easing the transition.

Changelog

Date	Version	Description
2021-06-11	1.0	Finalized the initial version
2021-09-02	2.0	Added the encoder configuration refactor.
2021-11-04	2.1	Added the “BiSS encoder CRC” section.
2021-11-29	2.2	Added content to velocity reference frame Content reorganized
2021-12-15	2.3	Added formula how to adapt external scaled measurement

Table of contents

1. Motivation

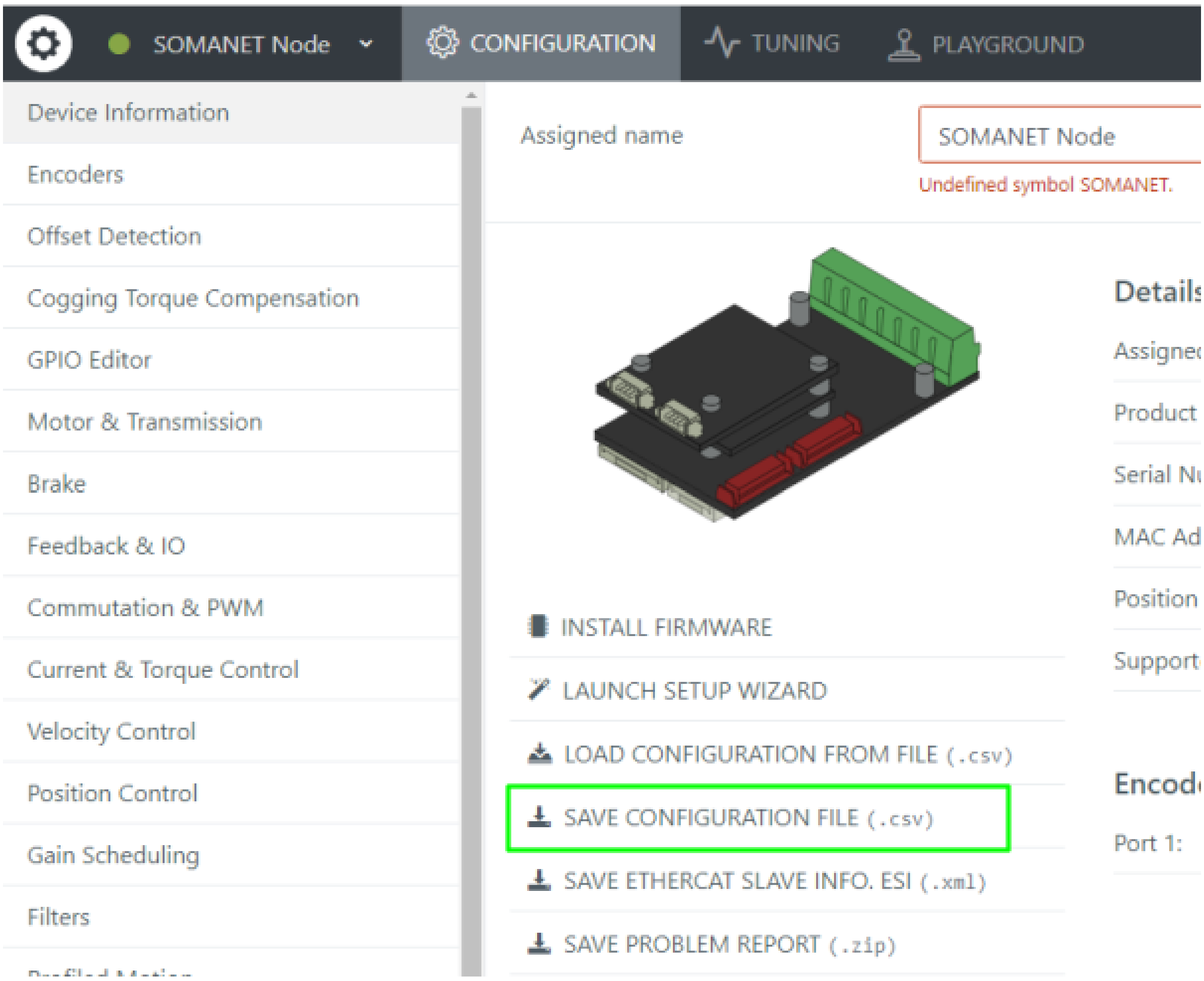
Several backwards-incompatible changes were made to the firmware. These changes were made for two main reasons, including

1. Improving the object dictionary conformance to the 2015 IEC 61800-7-201 (or CiA 402 profile) specification.
2. Add flexibility for incoming features to keep the object dictionary easy to understand.

These reasons should keep configuration of the device simple and clear, while also improving compatibility to existing EtherCAT master devices on the market.

The changes will necessitate re-tuning your control system, particularly the torque controller, as well as re-configuring some peripherals to get the previous behavior.

IMPORTANT NOTE! Please backup your drive's configuration before updating your firmware. You can accomplish this by navigating to OBLAC Drives → CONFIGURATION → Device Information → "SAVE CONFIGURATION FILE"



2. Change list

The following is a simple list of changes that will need further attention.

2.1. Torque controller

The torque controller tuning inputs have been completely reworked to support auto-tuning within the servo drive. This should make it easier to achieve the desired performance.
In torque controller object (0x2010) the previous subitems: Kp[1], Ki[2], Kd[3] are no longer supported and are replaced by subitems Settling time [10] and Damping ratio [11] (see documentation on object 0x2010 for details).

- Retune the torque control with OBLAC drives and save new parameters.

2.2. Digital inputs and outputs

2.2.1. Digital inputs

Digital input (0x60FD) replaces objects Digital input 1-4 (0x2501 through 0x2504)

- If you have mapped Digital inputs as TxPDO, the PDO mapping has to be changed
- If a Digital input is used in the user application, the reading of the Digital input has to be changed from single bit reading to a byte reading on a different object.

2.2.2. Digital outputs

Digital output (0x60FE) replaces objects Digital output 1-4 (0x2601 through 0x2604)

- If you have mapped Digital outputs as RxPDO, the PDO mapping has to be changed
- If a Digital output is used in the user application, the setting of the Digital outputs has to be changed from single bit to a byte on a different object.

2.2.3. Brake control - Manual output voltage mode

- When controlling the brake (or other device) in manual output voltage mode (see Brake options (0x2004) subindex Release strategy [4], set to 0), use the Digital output brake bit [0] to enable or disable a voltage applied to the brake output.

ⓘ Danger

In this mode, no automatic brake control is performed, even in the event of an emergency!

2.2.4. GPIO configuration

Configuration of a GPIO (0x2210) in v4.x with “input without pull-down” (value 1) will result in a lvldGpio error in v5.0.

- Use value 2 instead (there is no difference).

2.3. Offset-detection (auto-phasing)

2.3.1. General rework and renaming

The way to measure the commutation offset has been reworked and renamed to “auto-phasing” because the former routine was doing quite a lot of background diagnostics that were incompatible with the new auto-phasing workflow. In all cases, the aim of auto-phasing is to detect an accurate value for the Commutation angle offset (0x2001). You can execute this routine like any other diagnostic.

2.3.2. Diagnostics Opmode

Diagnostics (including commutation offset detection) are now performed by the OS Command object in Diagnostics Opmode. See our documentation on Diagnostics Opmode for details.

2.3.3. Faster execution of the phasing procedure

Additional diagnostics were removed from the pure commutation offset detection procedure to save time, but they can be triggered independently.

2.3.4. Additional methods

The commutation offset detection role is restricted to detect the relationship between the commutation encoder position and the phase angle of the rotor. There are now three methods (see subitem Measurement method 0x2009:3):

- **Method 0** doesn’t need tuning. Maximum rotation of the motor is 1 pole pair. This is the method that was also used in firmware versions < 5.0.
- **Method 1** requires that the subitems Phasing controller Kp [0x2009:4], Phasing controller Ki [0x2009:5], and Phasing controller Kd [0x2009:6] are tuned to achieve good performance. In this mode, good tuning can prevent any noticeable rotation of the rotor, while still achieving a reasonably good accuracy of the Commutation angle offset.

> ⓘ Attention
>
> `Method 1 is still in prototype phase and can be tested at user's own risk.`

- **Method 2** does not require any motion of the rotor. In fact, the brake can remain engaged while this procedure is executed. The accuracy reliability of this feature depends on the structure of the motor, so in most cases Method 1 is preferable. This method can be used in combination with incremental encoders to get a rough commutation angle location and skip the index detection routine.

> ⓘ Attention
>
> `Method 2 is still in prototype phase and can be tested at user's own risk.`

- Note that the Commutation angle offset (0x2001) found with v4 firmware is still valid for v5.
- The Auto-Phasing triggered in OBLAC drives will perform the required diagnostics and the commutation offset detection.

2.3.5. Required action

For manually triggering the Commutation Offset procedure: Instead of just setting the value -2 in object 0x6060 (what is now enabling the diagnostic mode), you have to additionally set the required OS Command for commutation offset detection (command 5).

It is recommended to perform the following OS Commands:

- Command 6: Open phase detection (recommended)
- Command 8: Phase resistance measurement (if needed)
- Command 9: Phase inductance measurement (if needed)
- Command 7: Pole pair detection (recommended)
- Command 4: Motor phase order detection (mandatory)
- Command 5: Commutation offset detection (mandatory)

2.4. Analog input range

Analog inputs were formally reported in the raw value that came from the ADC hardware component. This had the downside that every product had different valid ranges, depending on the circuitry behind them. In order to make implementation easier and present a unified interface for all products, the firmware is now scaling the raw ADC value to a 16-bit value.
0 will mean minimum input voltage, 65,535 is maximum voltage.
If you are using external scaled measurement, the parameters have to be adapted to the new range to keep the same data range after scaling.
If you use the external scaled measurement for a PT1000 or a KTY temperature sensor with a voltage divider, the external scaled measurement parameters are still valid and you do not need to change them. If you directly measure voltage, the external scaled measurement parameters have to be changed:

$A0' = A0.$
$A1' = A1/2^4$
$A2' = A2/2^8$
$A3' = A3/2^{12}$

2.5. External Scaled measurement limits

Additional limit thresholds for external scaled measurement are added:

- 0x2038:11: Upper error threshold (default value: 65,535)
- 0x2038:12: Lower error threshold (default value: 0)

If the scaled value is out of the limits given by the upper and lower thresholds, an “ExAnSnsr” fault is triggered.

If no coefficients a0 - a5 are defined for the 5th order polynomial calculation, the external analog value will stay within the default limits. But if there are some parameters defined that lead to negative scaled values (e.g. for a temperature sensor) and there is no sensor connected to the analog input, “ExAnSnsr” fault will be shown after update to FW 5.x.

Define valid upper and lower limit thresholds that match with the function of the connected sensor.

Not connected analog inputs might lead to floating values in analog input data that triggers the limit. Keep to the default values for external scaled measurements when analog input is not used.

2.6. Velocity units

There are additional data objects that are defined in user-defined units. They are:

- 0x2010:6 Torque controller / Field weakening starting speed
- 0x2010:7 Torque controller / Field weakening ending speed
- 0x2012:4 Position controller / Position loop integral limit
- 0x2013:6 Gain scheduling, Position loop integral limit 0
- 0x2013:15 Gain scheduling, Position loop integral limit 1

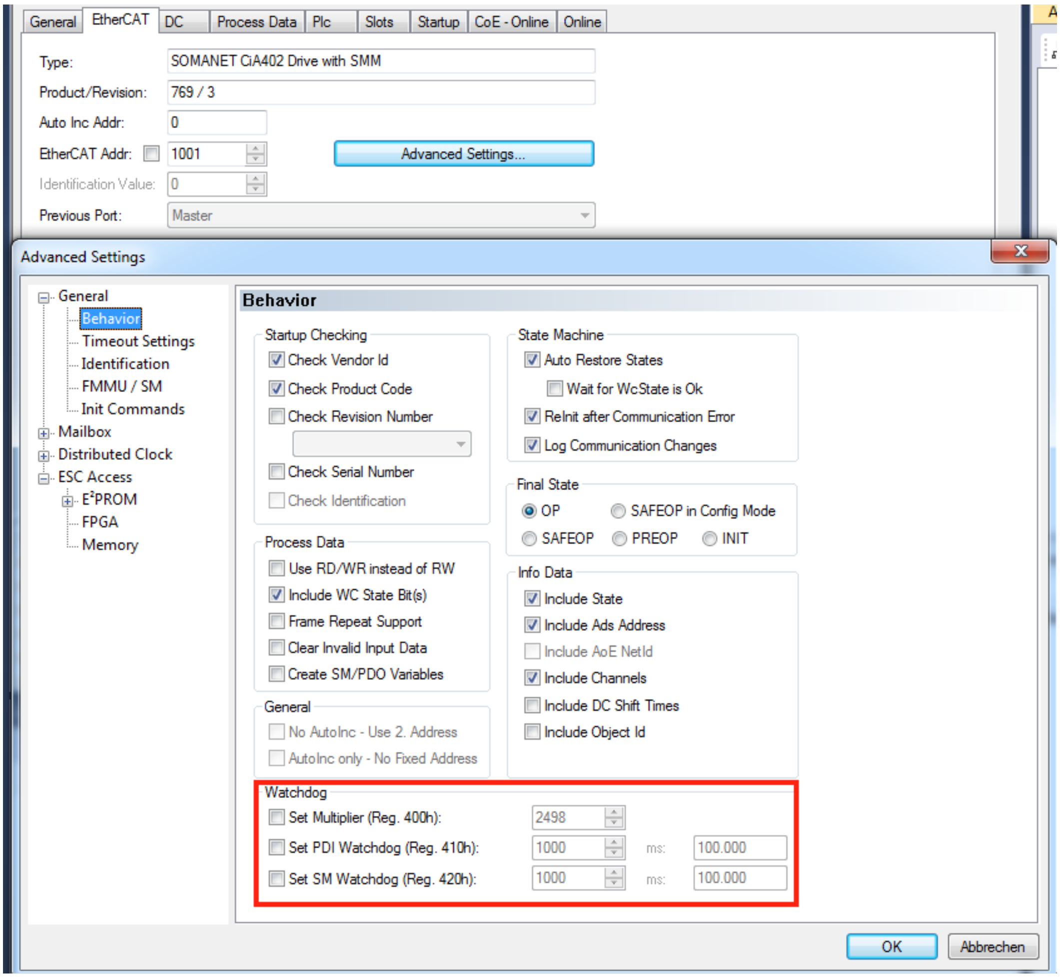
2.7. Velocity reference frame

In the v4.x firmware, the reported velocity was always that of the encoder configured for velocity control. So if the velocity encoder was mounted to the motor shaft, the velocity was reported as much higher than would be seen on the drive shaft. In v5.0, this is now unified at the drive shaft, as hinted in the IEC standard. This means that all target and reported velocities and accelerations/decelerations are described from the perspective of the position encoder. This affects only systems with gears and a dual encoder / dual loop setup.

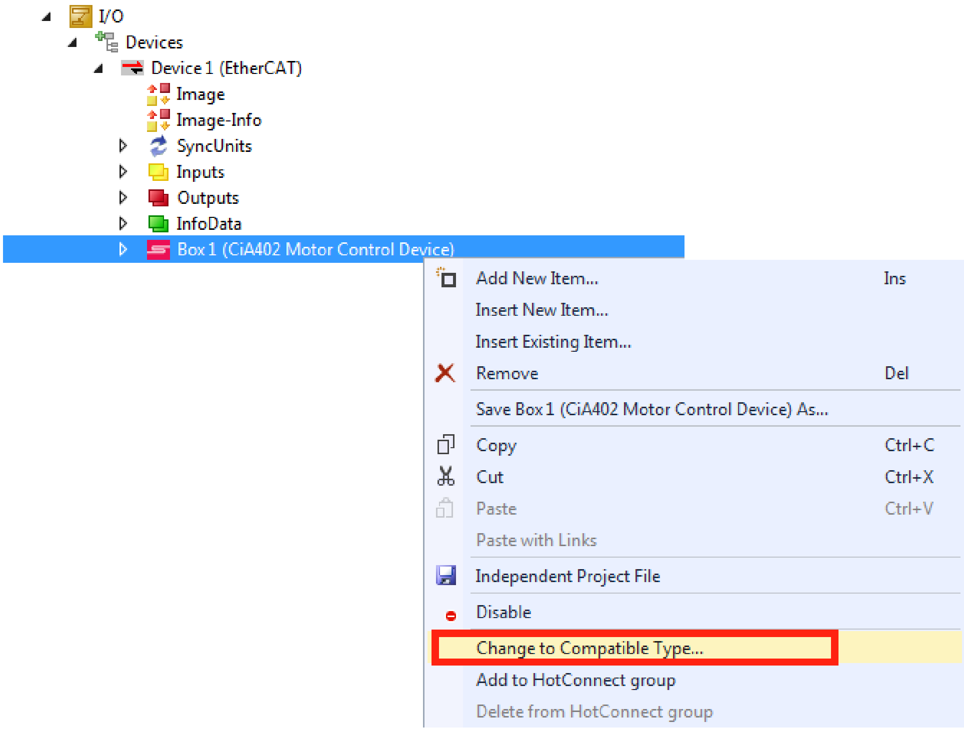
- Therefore, when migrating to 5.0 to get the same behavior of the system, the velocity-related command values including velocity, acceleration and deceleration in the master have to be divided by the factor gear ratio. Similarly, the feedback values such as Velocity actual value have to be multiplied by gear ratio to be converted to the motor reference frame. All velocity-related configurations, except Max motor speed, are also influenced, such as Quick-stop deceleration.
  Max motor speed is related to the motor itself and therefore still referenced to the motor reference frame.
- The values for acceleration and deceleration for Quick stop and in profile modes are now as well referenced to the drive shaft.
- If you have a gearbox, pay special attention to the SI unit velocity object (0x60A9). You may find that velocity units of RPM aren’t sufficient to finely control the velocity of the output shaft, and mili-RPM is a better choice.
- If you use OBLAC drives when changing the SI unit for velocity from the configuration page, all velocity related other objects (e.g. accelerations, limits, ...) will be changed to the new unit automatically by OBLAC drives.
  If the unit is changed by direct access to the SI unit object 0x60A9 (e.g. out of the master), all the other related objects have to be adapted manually.

2.8. Cyclic heartbeat (communication watchdog)

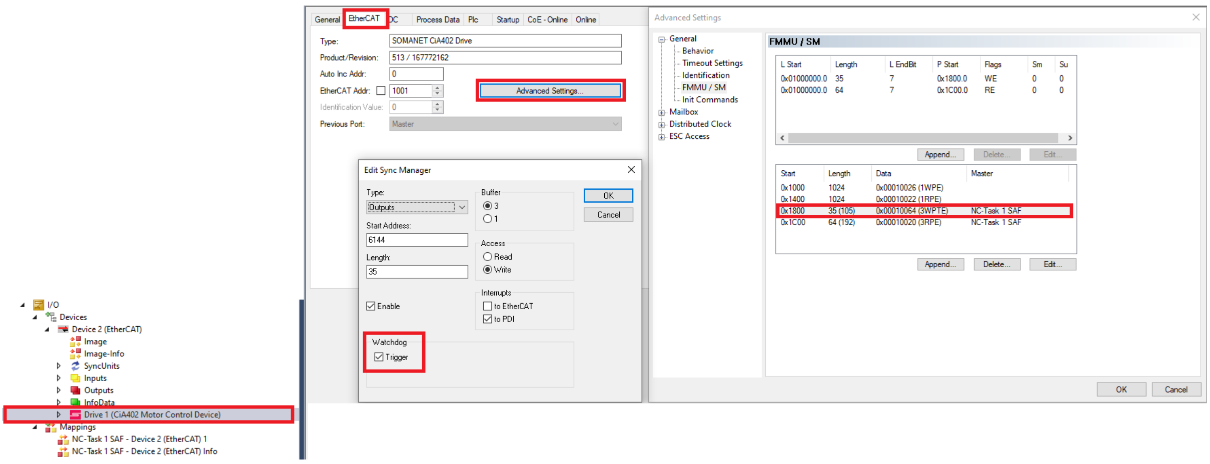
Cyclic heartbeat, the EtherCAT communication watchdog, now uses EtherCAT Slave Controller watchdog, and is therefore configurable. The figure below shows an example of how it's configured with TwinCAT 3.



Accordingly, the ESI (EtherCAT Slave Information) file is updated. For existing EtherCAT master projects, please update the project with the new EtherCAT Slave Information (Change to compatible type in TwinCAT 3).



The user can also choose to manually change the sync manager configuration. An example is shown below.



2.9. Encoder configuration

The way encoders are configured has been reworked to drastically simplify the procedure at a low level. There are now simply two configuration and two feedback objects.

- 0x2110 Encoder 1 configuration
- 0x2111 Encoder 1 feedback
- 0x2112 Encoder 2 configuration
- 0x2113 Encoder 2 feedback

There are three new subitems for the position, velocity, and torque controllers to select the encoder to use:

- 0x2012:9 Position controller: Encoder source
- 0x2011:5 Velocity controller: Encoder source
- 0x2010:12 Torque controller: Encoder source

OBLAC Drives will mostly remain the same, so that configuring your encoder is easy to do.

> ⓘ Note
>
> There will be no automatic conversion of the existing configuration when migrating! The encoder settings have to be done manually on every change between 4.x to 5.0 or vice versa. Please store the configuration before up- or downgrade if you want to fall back later.

2.9.1. Hall sensor + incremental encoder

To configure a hall sensor to work with an incremental encoder, first configure the incremental encoder. The parameter “Hall sensor port” can be used to specify the port on which the Hall sensor is connected, and the drive will automatically use it on startup to avoid the index detection routine. There is no need for additional configuration of the hall sensor on encoder port 2. Configuring the Hall additionally on port 2 will lead to a *HwRsrcEr.*

2.9.2. BiSS encoder CRC

The representation of the CRC polynomial has been changed. If you were using the default value of 48 (0x30) you can just go ahead and use the new default value, which is 67 (0x43).

In the unlikely event that your CRC polynomial parameter (subindex 9 of the appropriate BiSS encoder object) is different from 48, there are two different approaches you can follow.

If you know the CRC polynomial of your BiSS encoder, you should just represent it in binary form. For example, if your polynomial is $x^6+x^4+x^0$, the binary representation would be 0b1010001. This is the value you should write to the subindex 15 of the appropriate Encoder configuration object (0x2110 or 0x2112), which is 81 in decimal representation in this case.

On the other hand, if you don't know the CRC polynomial of your BiSS encoder, you should first represent the CRC polynomial parameter in binary form. For example, if you have a value of 18 in the subindex 9 of the BiSS encoder object, the binary representation would be 0b10010. Now you should add a "1" at the end of it like this: 0b100101. Once you've done that, you just need to revert this number, so that the LSB becomes the MSB and vice versa: 0b101001. And this is the value you should write to the subindex 15 of the appropriate Encoder configuration object (0x2110 or 0x2112), which is 41 in decimal representation in this case.

2.10. Deprecated items

- Object Supported homing methods (0x60E3) is no longer in the object dictionary. See the documentation on homing methods to see which modes are supported.
- Recuperation (0x2005) is no longer supported.
- Many objects related to encoder configuration are removed in favor of the new method. This includes 0x2100 Feedback sensor ports, and 0x2201 through 0x220E (encoder types for configuration).
- The secondary feedback values, 0x230A Secondary position value and 0x230B Secondary velocity value are removed. Instead, they can be found in the new objects 0x2111 Encoder 1 feedback and 0x2113 Encoder 2 feedback.

3. Behavior

After upgrading, some behaviors may be caused by a misconfigured device. The following table should help find the solution.

Table: Issues and solutions

Issue	Solution
The motor is buzzing when enabled.	Re-tune the torque controller. It has changed.
Error IvldGpio is raised.	Check that the configuration of the Digital inputs are correct.
ExAnSnsr	Connect the configured sensor on the analog input. Set the proper configuration for analog external scaled measurement. Adapt the analog input scaling to the changed data size of 16-bit.
New objects values are set to default after downgrading to V4.4 and back to 5.0	Store and export the configuration before downgrading to 4.x. and restore after upgrading to 5.0 again.
No encoders configured	Encoder configuration has to be done on every migration (see 2.8).
CyclicHb error with EtherCAT master	Check chapter 2.6., update the project with the new ESI or change the Sync Manager settings.
HwRsrcEr	Invalid encoder configuration E.g. configured a Hall sensor twice along with incremental encoder and on 2nd port.
Motor turns much faster than with previous v4.4	The reference frame for motor speed has moved to gears out. Adapt your target values to the new velocity reference frame. Divide the values with gear ratio.
Actual velocity is reported slower	The reference frame for motor speed has moved to gears out. Multiply the motor speed by gear ratio.
Faster acceleration and deceleration	The reference frame for motor speed has moved to gears out. This includes the acceleration and deceleration for Quick stop and in profile modes. Divide the values by the gear ratio.