

# ON Semiconductor

## Is Now

# onsemi™

To learn more about onsemi™, please visit our website at  
[www.onsemi.com](http://www.onsemi.com)

---

**onsemi** and **onsemi** and other names, marks, and brands are registered and/or common law trademarks of Semiconductor Components Industries, LLC dba "**onsemi**" or its affiliates and/or subsidiaries in the United States and/or other countries. **onsemi** owns the rights to a number of patents, trademarks, copyrights, trade secrets, and other intellectual property. A listing of **onsemi** product/patent coverage may be accessed at [www.onsemi.com/site/pdf/Patent-Marking.pdf](http://www.onsemi.com/site/pdf/Patent-Marking.pdf). **onsemi** reserves the right to make changes at any time to any products or information herein, without notice. The information herein is provided "as-is" and **onsemi** makes no warranty, representation or guarantee regarding the accuracy of the information, product features, availability, functionality, or suitability of its products for any particular purpose, nor does **onsemi** assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation special, consequential or incidental damages. Buyer is responsible for its products and applications using **onsemi** products, including compliance with all laws, regulations and safety requirements or standards, regardless of any support or applications information provided by **onsemi**. "Typical" parameters which may be provided in **onsemi** data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. **onsemi** does not convey any license under any of its intellectual property rights nor the rights of others. **onsemi** products are not designed, intended, or authorized for use as a critical component in life support systems or any FDA Class 3 medical devices or medical devices with a same or similar classification in a foreign jurisdiction or any devices intended for implantation in the human body. Should Buyer purchase or use **onsemi** products for any such unintended or unauthorized application, Buyer shall indemnify and hold **onsemi** and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that **onsemi** was negligent regarding the design or manufacture of the part. **onsemi** is an Equal Opportunity/Affirmative Action Employer. This literature is subject to all applicable copyright laws and is not for resale in any manner. Other names and brands may be claimed as the property of others.

## AMIS-30623/AMIS-30624/ NCV70627 Stall Detection



ON Semiconductor®

[www.onsemi.com](http://www.onsemi.com)

### APPLICATION NOTE

#### Introduction

The AMIS-3062x and NCV70627 are single-chip microstepping motordrivers with position controller and control/diagnostic interface. They are ready to build dedicated mechatronics solutions connected remotely with a LIN or I<sup>2</sup>C master. The chip receives positioning instructions through the bus and subsequently drives the motor coils to the desired position. The on-chip position controller is configurable (OTP or RAM) for different motor types, positioning ranges and parameters for speed, acceleration and deceleration. The motor drivers act as a slave on the LIN or I<sup>2</sup>C bus and the master can fetch specific status information like actual position, error flags, etc. from each individual slave node.

An integrated sensorless stall detection prevents the positioner from losing steps and stops the motor when running into stall. This enables silent, yet accurate position calibrations during a referencing run and allows semi-closed loop operation when approaching the mechanical end-stops.

This application note describes the stall detection operation and how to use it to enable the development of high reliable sensorless stepper motor applications\*.

\*This document is only intended as a guideline during development. The information as also the tips and tricks given in this document should always be verified by the customer under all operating conditions.

#### ELECTROMAGNETIC FORCE

Electromagnetic induction creates an electromagnetic force (EMF) when a conductor moves through a magnetic field (Faraday's Law of induction). The magnitude of this EMF is given by:

$$e = -\frac{d\Theta}{dt} \quad (\text{eq. 1})$$

where  $\Theta$  is the magnetic flux of the field.

The minus in above equation comes from Lenz's Law. If several conductors  $N$  are connected in series this gives:

$$e = -N\frac{d\theta}{dt} \quad (\text{eq. 2})$$

When  $\Theta = \Theta_m \sin \omega t$  is substituted this gives

$$e = E_m \cos \omega t \quad (\text{eq. 3})$$

where  $E_m = -N\omega\Theta_m$

The amplitude of this EMF is a function of the number of windings  $N$ , the magnetic flux  $\Theta_m$  of the field and the rotation speed  $\omega$ . Because  $N$  and  $\Theta_m$  are constant for a given stepper motor, EMF is a good representation of the angular speed of the stepper motor.

#### SENSING BEMF

As explained above, the EMF (also called Back EMF because of the minus sign (Lenz's Law)) is a good measure for the speed of a stepper motor. By measuring the BEMF, the stepper motor could sense if the motor is moving or not (stalled).

A two-phase stepper motor is driven by two H-bridges (one H-bridge for each coil). Both driver stages create a sine and cosine current. Figure 2 illustrates the coil current through the X- and Y-coil when 1/8 microstepping is used.

# AND8471/D

AMIS-30623 / AMIS-30624 / NCV70627

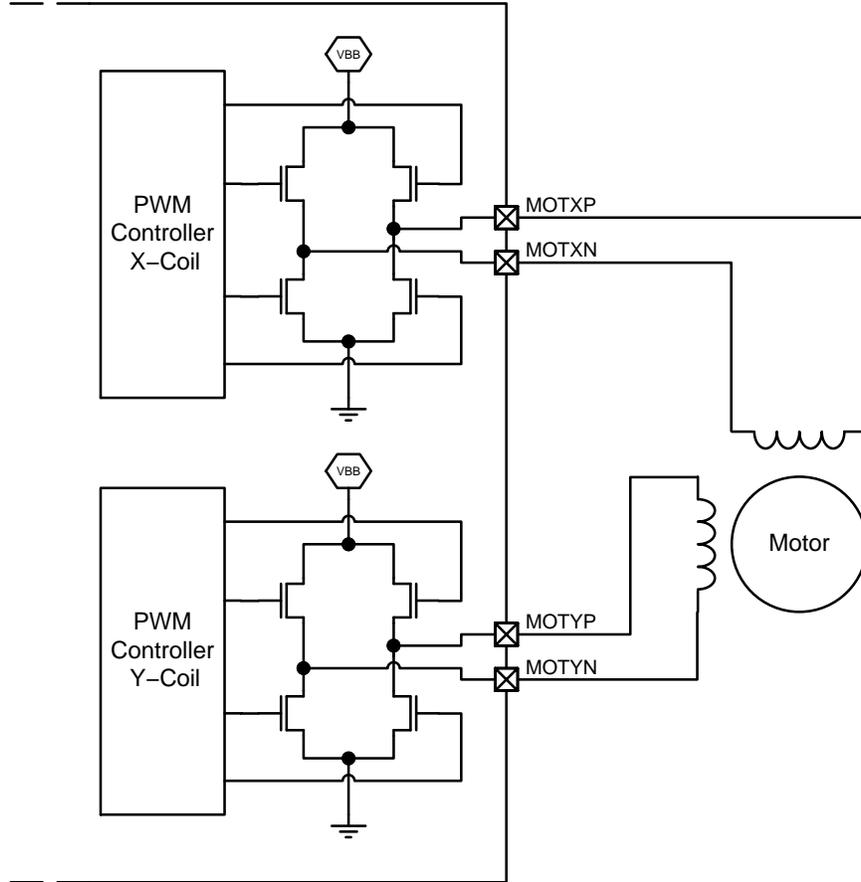


Figure 1. Driver Configuration

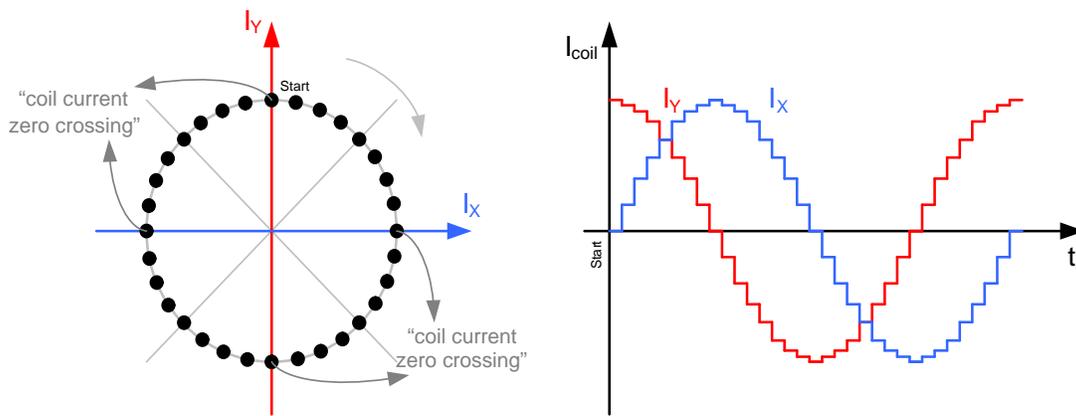


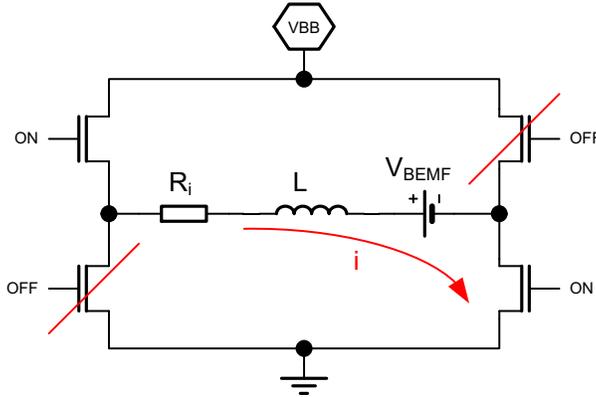
Figure 2. 1/8 Microstepping

# AND8471/D

At any moment in time the relation between voltage and current is given by:

$$U = V_{BEMF} + i \times R_i + L \times \frac{di}{dt}$$

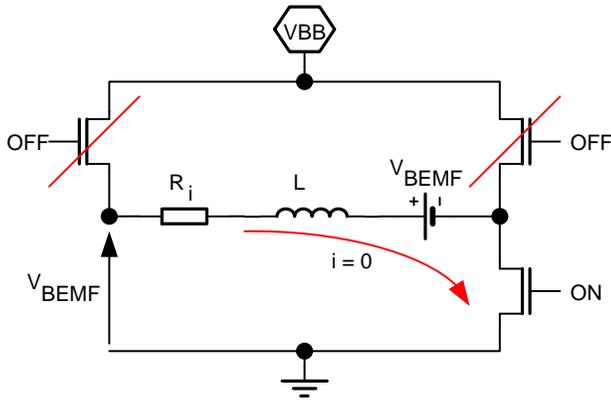
where:  $L$  = Motor Coil Inductance  
 $R_i$  = Serial Resistance of the Motor Coil  
 $i$  = Coil Current  
 $V_{BEMF}$  = Generated Back EMF



**Figure 3. Driving Coil Current**

As can be seen from Figure 2, in one electrical period (of both currents) there are 4 moments when no current is forced through one of the coils (the so called Coil Current Zero

Crossings). It's at this moment that the BEMF can be measured (see Figure 4).



$$U = V_{BEMF} + \cancel{i \times R_i} + L \times \cancel{\frac{di}{dt}}$$

**Figure 4. Measure BEMF During Coil Current Zero Crossing**

Above principle is used by AMIS-30623, AMIS-30624 and NCV70627 to measure the motor BEMF. This is done

on the 4 motor pins during one electrical period using a multiplexed sample and hold circuit (see Figure 5).

AMIS-30623 / AMIS-30624 / NCV70627

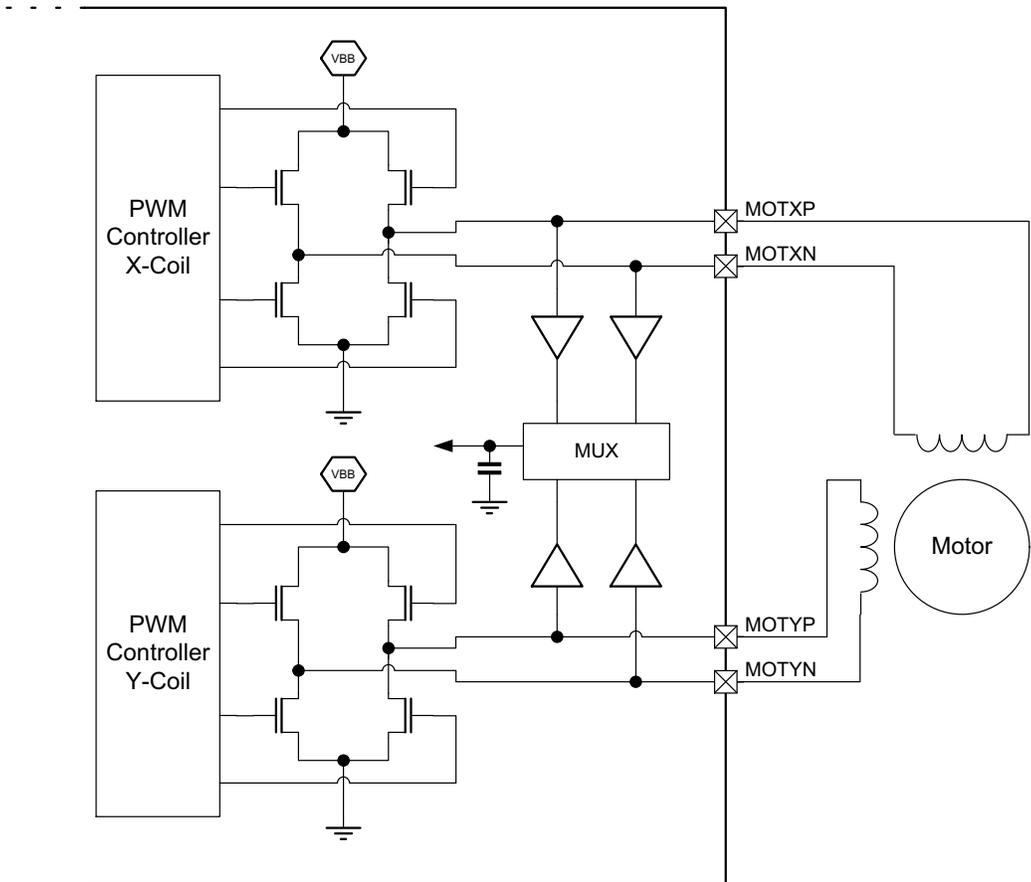


Figure 5. Multiplexed Sample and Hold Circuit to Sample BEMF

**STALL DETECTION PARAMETERS**

This chapter contains a technical description of the different parameters that can be set for stall detection. Next chapter gives a step-by-step explanation to define the stall parameters.

**Absolute Threshold**

When the motor is rotating a BEMF will be generated. In normal operation this BEMF would look something like given in Figure 6.

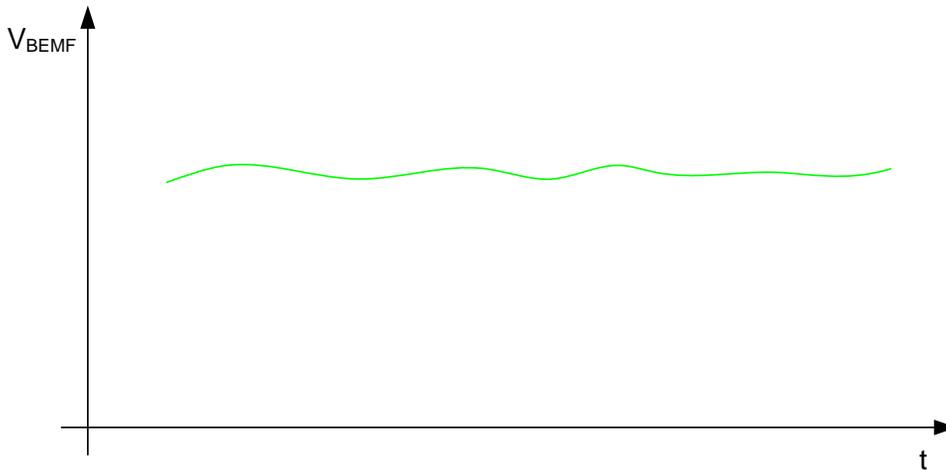


Figure 6. BEMF of a Rotating Motor

If the motor gets blocked at some point, the BEMF will drop (Figure 7).

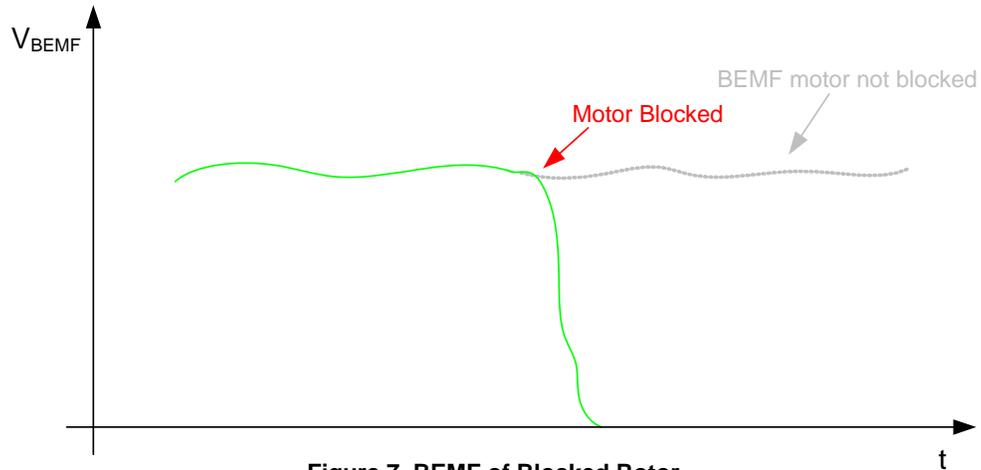


Figure 7. BEMF of Blocked Rotor

AMIS-3062x and NCV70627 have a stall parameter called absolute threshold (AbsThr) to detect this stall. If the

BEMF drops below the AbsThr level, stall will be detected by the motor driver and the motion will stop (Figure 8).

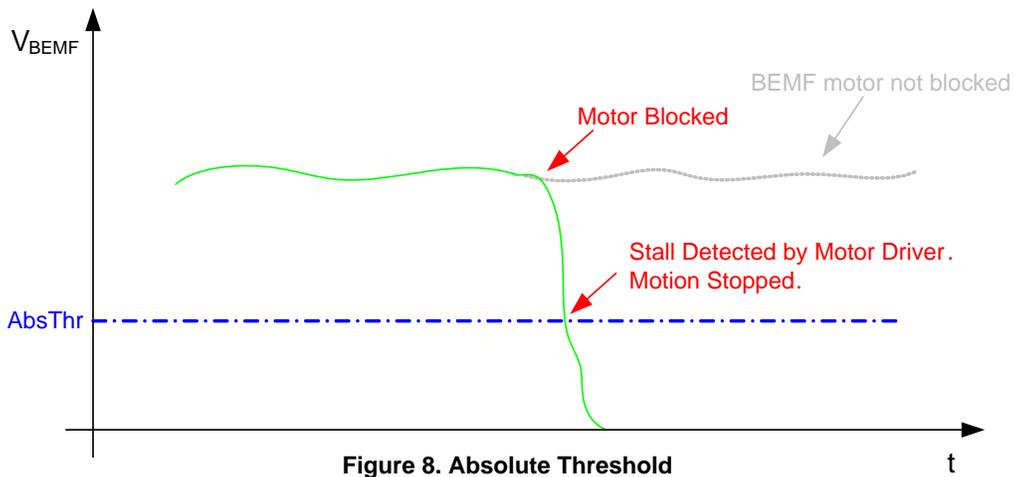


Figure 8. Absolute Threshold

There is a second reason why the absolute threshold is used, in case a motor is blocked from the beginning.

Figure 9 displays the theoretical and real velocity of the stepper motor during acceleration and the generated BEMF.

More information on this acceleration profile can be found in the stepper motor driver data sheets (see References). The real velocity represents the real movement of the rotor. The BEMF shape will be similar to this real velocity.

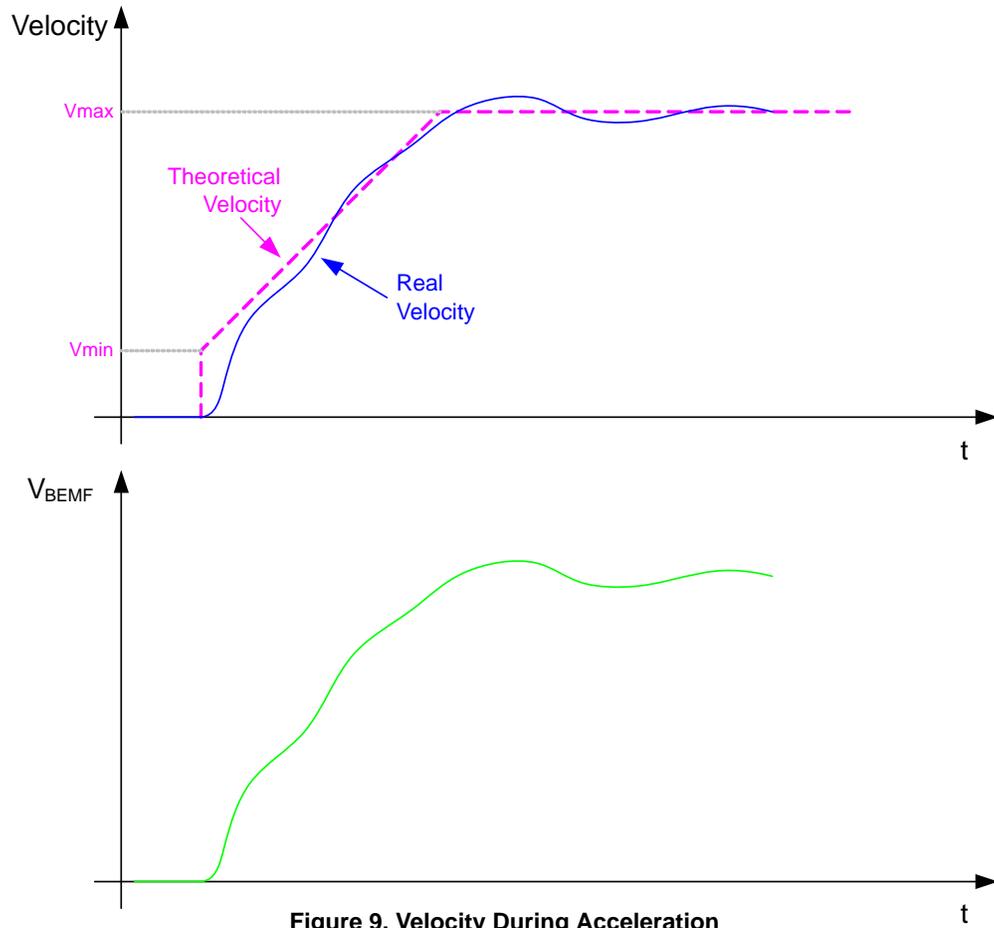


Figure 9. Velocity During Acceleration

Figure 10 displays what would happen if the rotor is blocked at start of the movement. Because the rotor is blocked, the rotor will not move and the BEMF will stay low. Because the BEMF is below the AbsThr level after

acceleration\*, stall will be detected by the motor driver and the movement will be stopped.

\*Stall detection is only enabled after the acceleration phase. During deceleration stall detection is also disabled.

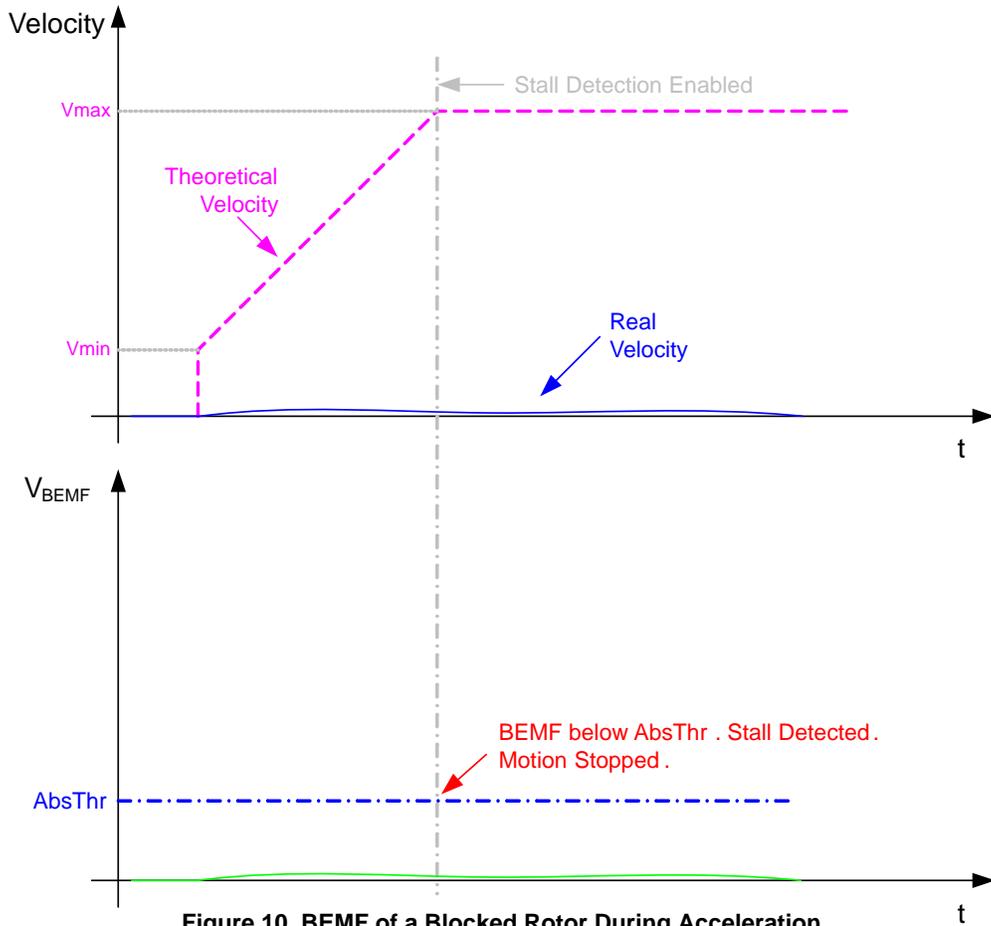


Figure 10. BEMF of a Blocked Rotor During Acceleration

**Delta Threshold**

Although stall detection will work by only using an absolute threshold level, it has some drawbacks. If different speeds are used, the AbsThr level always needs to be adjusted if the speed is adjusted. One could decide to define AbsThr based on the slowest speed (lowest BEMF) but this will have an effect on the stall detection accuracy at higher speeds.

Additionally, when a motor gets more loaded, the BEMF will shift (see Figure 11). Although the amplitude of the BEMF stays the same, the shift and the fact that the BEMF is only sampled during the coil current zero crossing could result in unwanted stall detection. Because of this, it's sometimes better to take the AbsThr value low to avoid falls stall under all operating conditions (see also Figure 14).

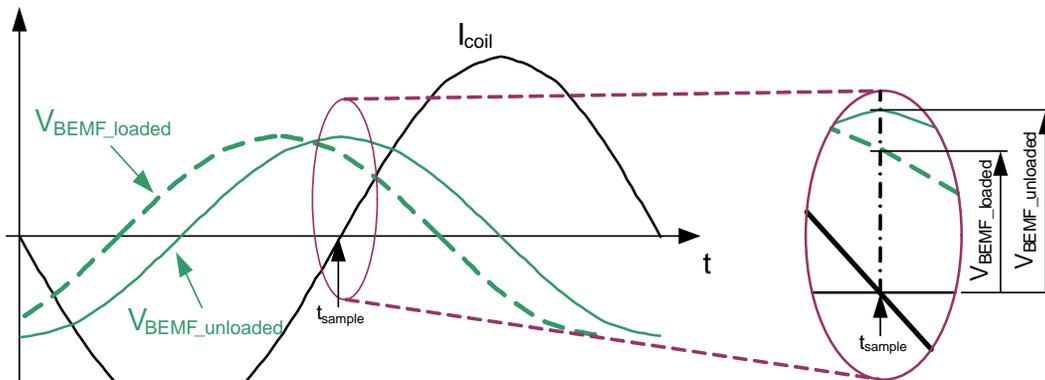


Figure 11. Effect of Motor Load on the BEMF

Figure 12 displays another issue. If by some external influence the motor is slowed down for a very short time, or accelerated, this will not be detected by the AbsThr level

(BEMF stays above AbsThr level). This effect can however result in steploss.

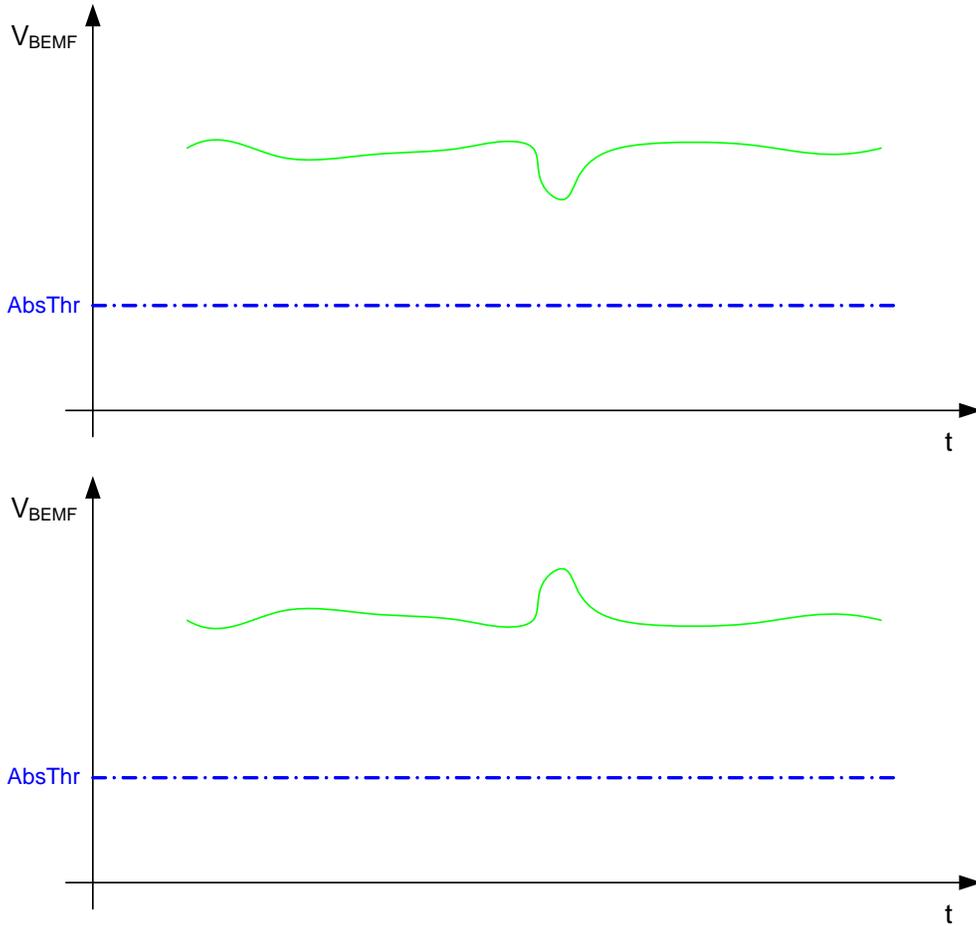


Figure 12. BEMF of a Rotating Motor (Top: Short Deceleration, Bottom: Short Acceleration)

To detect these short dips or peaks, AMIS-30623 and AMIS-30624 (not applicable for the NCV70627) will take the average value of the BEMF and add a delta on top and

below this level (DelThr). If the immediate BEMF value goes outside this window, stall is detected and the movement is stopped.

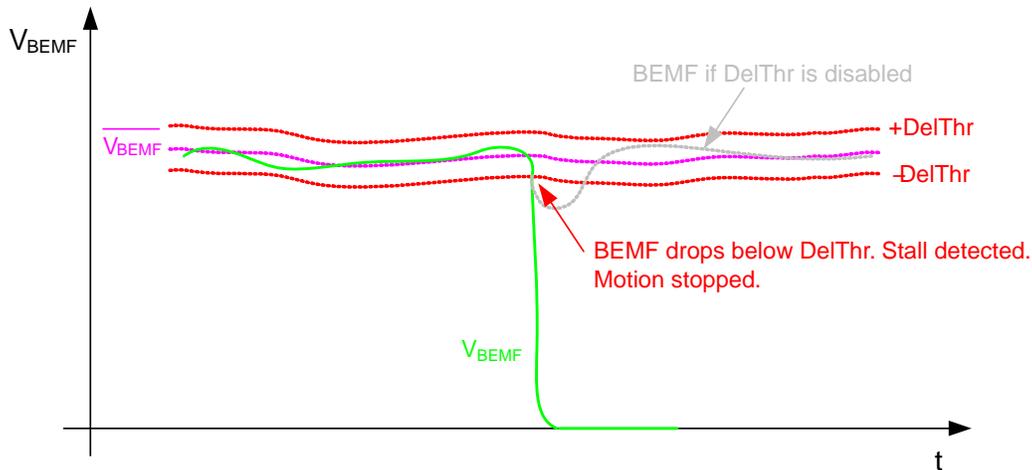


Figure 13. Delta Threshold (Not Applicable for the NCV70627)

If different speeds are used in the application, the AbsThr can be based on the slowest speed (lowest  $V_{max}$ ). The AbsThr level will only be intended to detect a blocked rotor at start of a rotation\*. DelThr will be used to prevent stploss

during motion. If AbsThr level is taken low enough, an increase in load will not trigger a falls stall (see Figure 14).

\*A slow decrease in speed will also trigger AbsThr if speed gets low enough.

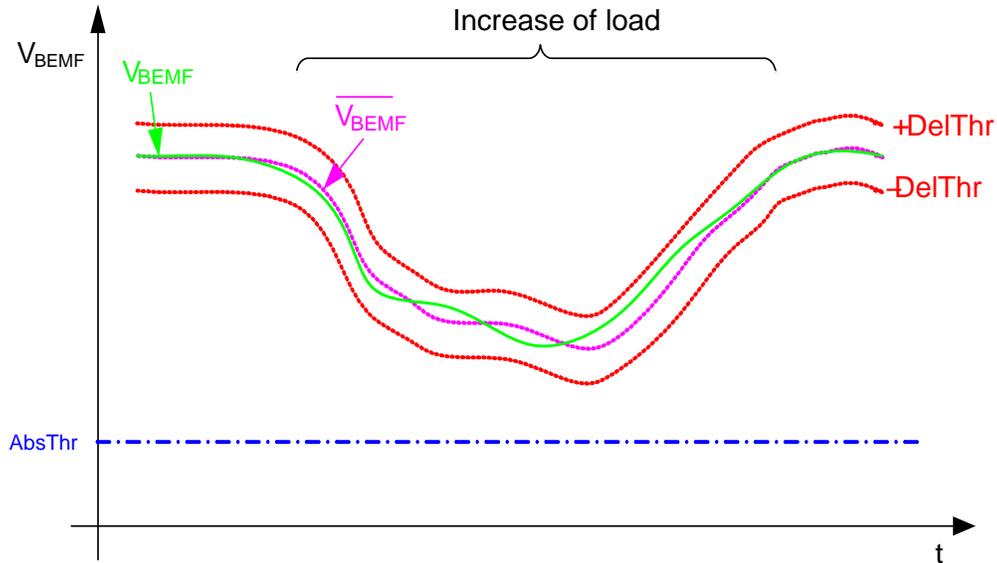


Figure 14. Use DelThr to Avoid Falls Stall Due to Increased Load (Not Applicable for the NCV70627)

**Full Steps to Stall Enable**

As mentioned before, stall detection is only enabled when the motor is at the (theoretical) maximum velocity ( $V_{max}$ ). Because a stepper motor is a 2<sup>nd</sup> order system (simplified),

a step response can lead to overshoots in the movement. Depending on the acceleration level, the speed of the motor can oscillate or ring around the  $V_{max}$  set-point. This could lead to falls stall detection (see Figure 15).

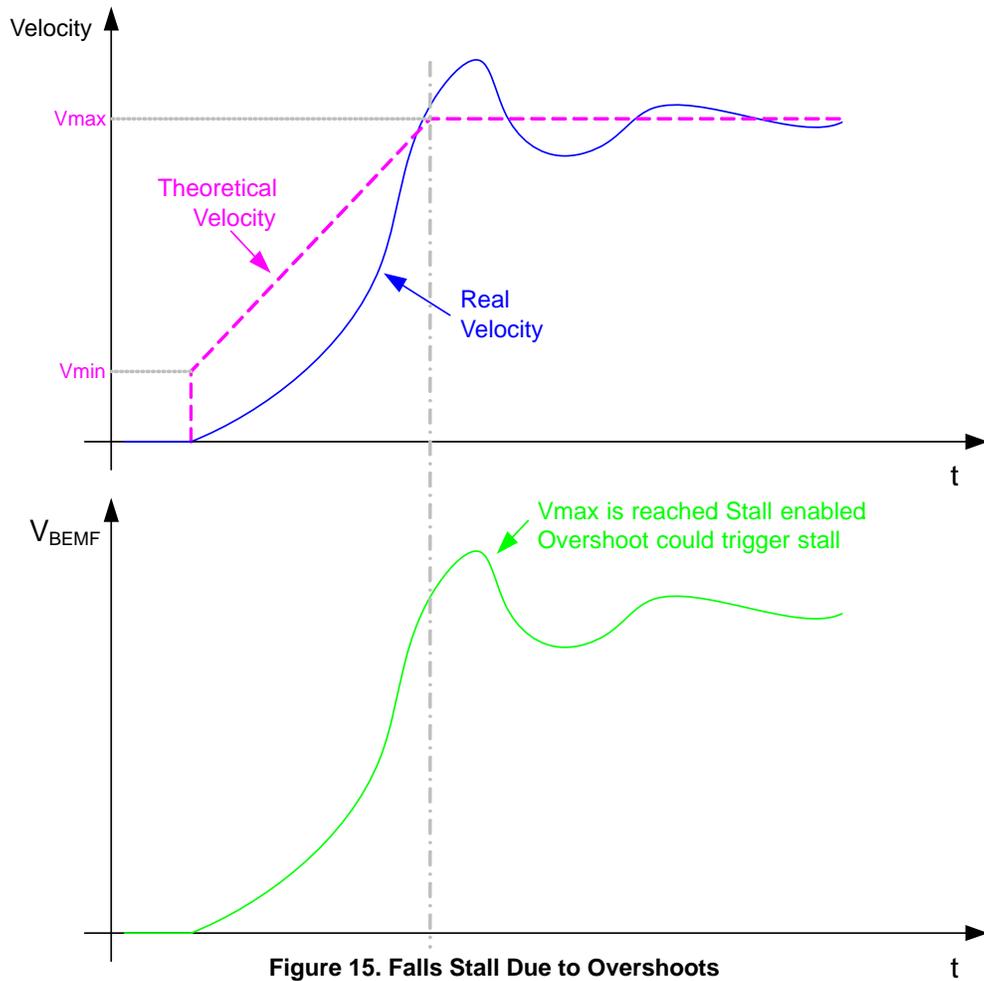


Figure 15. Falls Stall Due to Overshoots

A possible solution could be to make the DelThr wider or decrease AbsThr so the overshoots would not trigger the stall detection. However, because these overshoots will damp after a certain time, this would lead to a less sensitive stall detection.

Another solution could be to decrease the acceleration but for some applications it could be necessary to use a high acceleration (see application note [AND8404/D](#)). Keep in mind however that the BEMF represents the real movement

of the motor (see also Electromagnetic Force). If overshoots are seen in the BEMF, these overshoots will also be present in the real movement of the rotor (as can be seen in the blue curve of Figure 15). Too large overshoots could lead to steploss making it always important to minimize these overshoots (even if stall detection is not used).

To avoid falls triggering due to the overshoots, stall detection can be delayed with a certain amount of full steps (FS2StallEn).

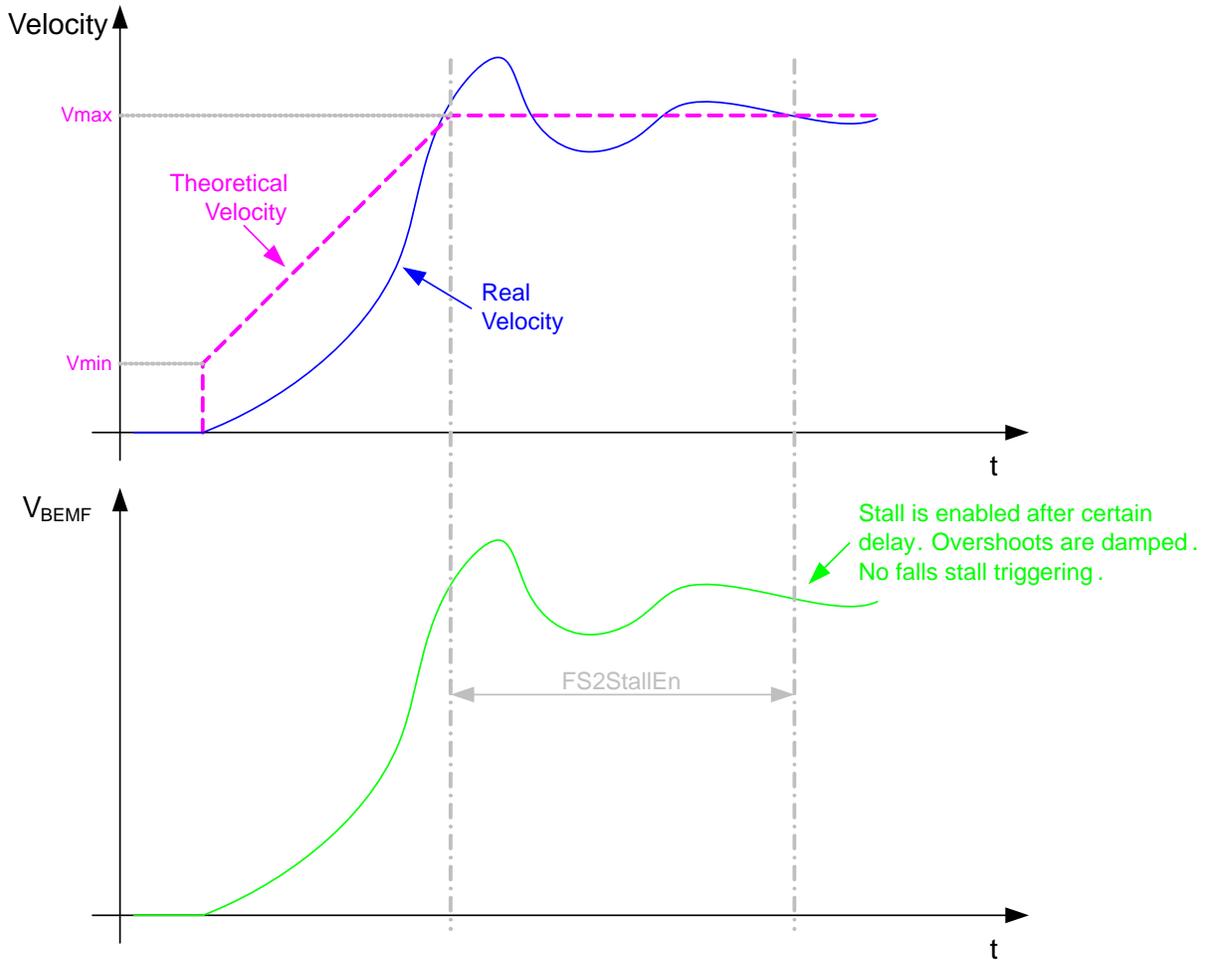


Figure 16. Delay Stall Detection

**Minimum Samples**

As mentioned before, the BEMF is only measured during a coil current zero crossing (see Figure 2). Figure 17 gives a more detailed view on the coil current zero crossing. The ideal coil current is given by the purple dashed line. At the moment the coil current zero crossing phase is initiated (purple dashed line goes to zero), one side of the coil is connected with ground, the other one is left open to measure the BEMF (see schematic at the right side of Figure 17). Because the coil current needs time to decay

to zero, the (real) coil current will not be zero at the beginning of the coil current zero crossing phase. This will give a spike in the measured coil voltage (coil voltage will clamp to  $V_{BB} + 0.6\text{ V}$ ).

When the real coil current has become zero a voltage transient is visible on the coil. The time this voltage transient is present depends on the coil inductance  $L$  and the parasitic resistance  $R_p$ . It is only after this voltage transient that the real BEMF can be measured!

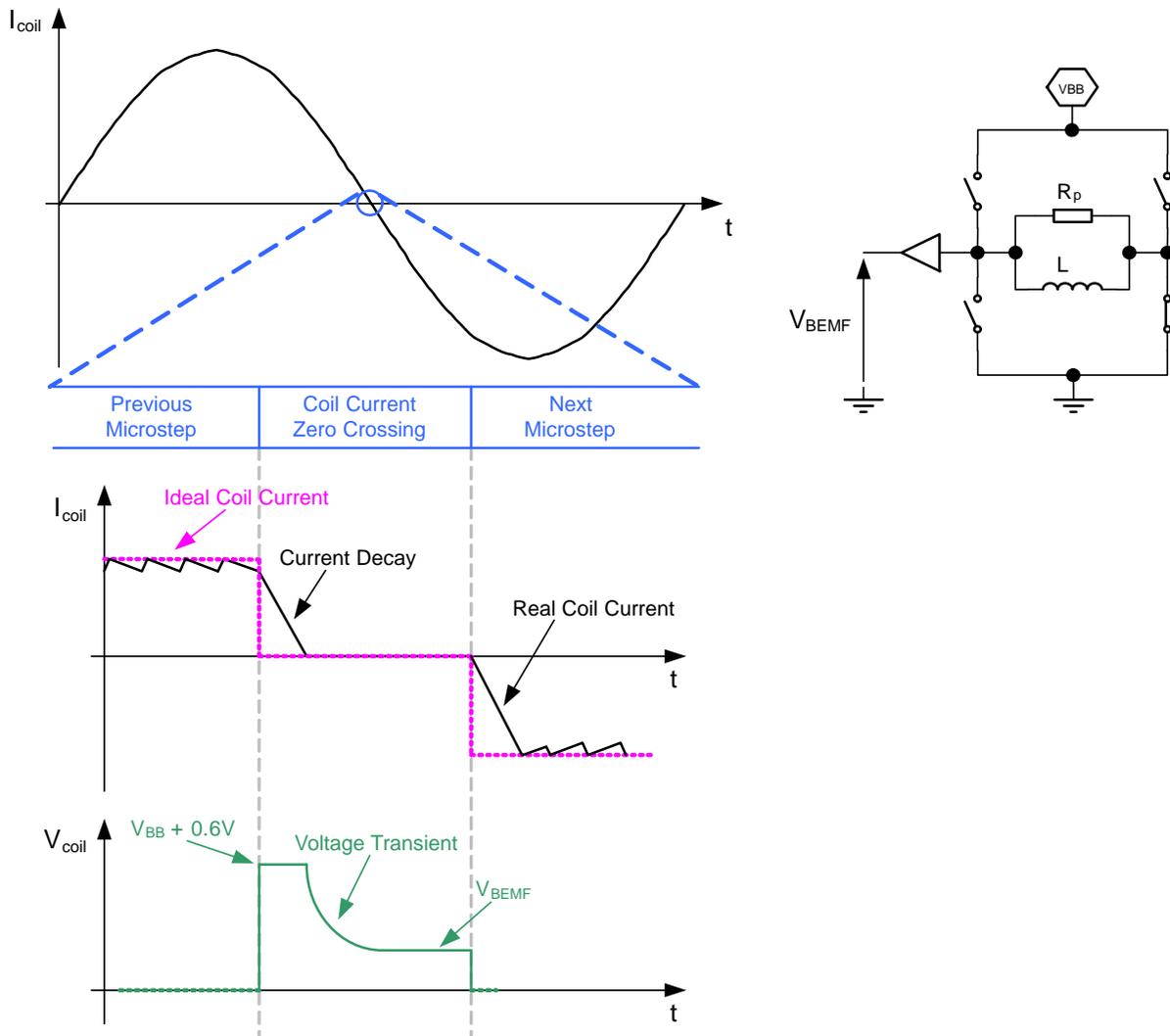
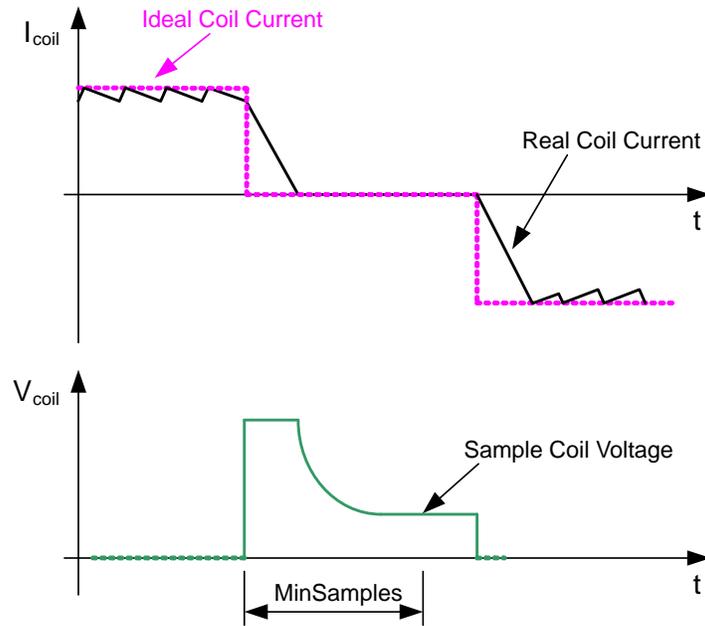


Figure 17. Coil Current Zero Crossing

Because AMIS-30623/AMIS-30624/NCV70627 does not know when the voltage transient has ended, the moment to measure the BEMF needs to be specified. This can be done by using the MinSamples parameter. The MinSamples

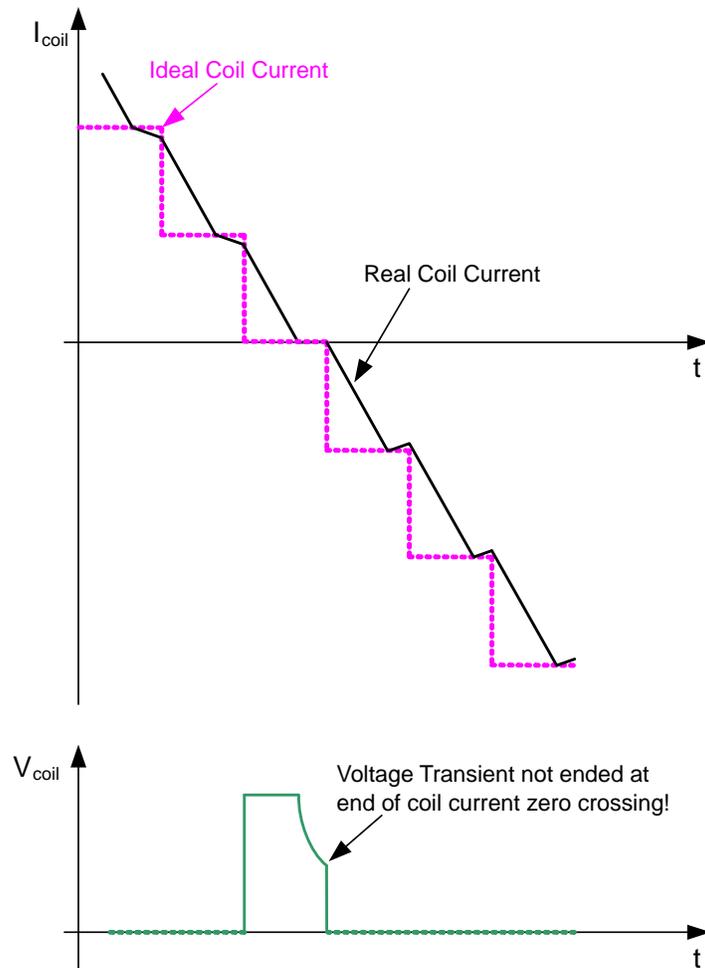
parameter specifies the delay time between the (theoretical) start of the zero crossing and sampling of the coil voltage (see Figure 18).



**Figure 18. MinSamples Parameter**

When MinSamples is set larger than the microstep time, the coil current zero crossing will be stretched. This makes it possible to do stall detection at high speeds but will have an effect on the coil current shape.

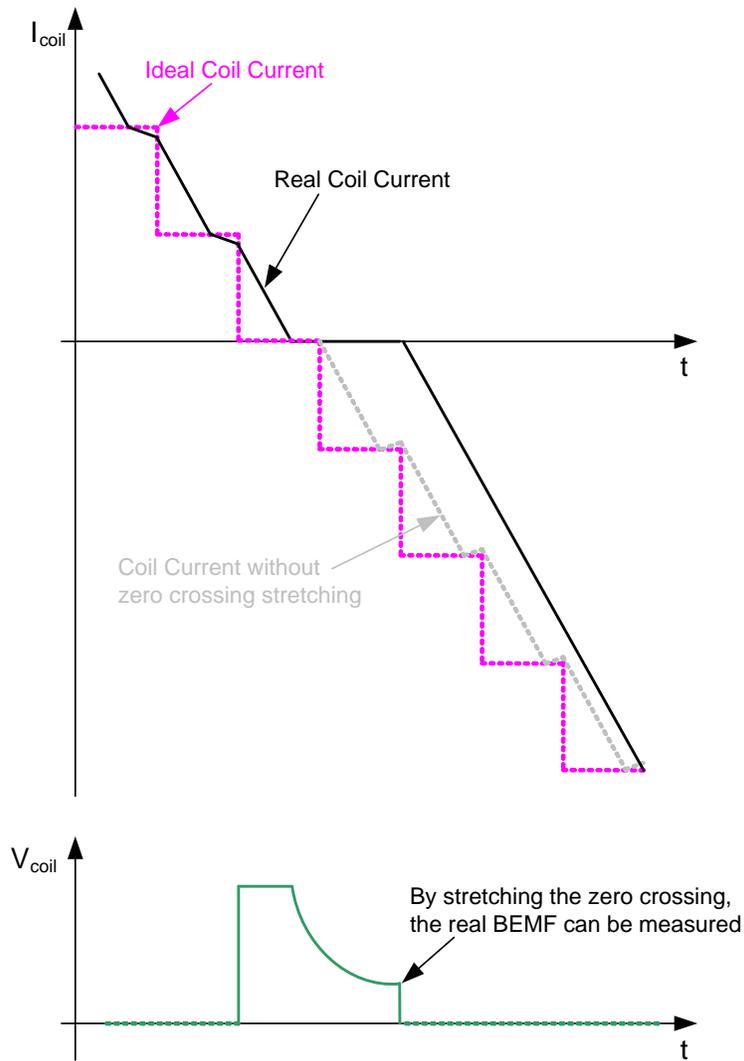
Figure 19 displays what could go wrong if the stepper motor is operated too fast. At the end of the coil current zero crossing the voltage transient has not yet finished. Because of this the real BEMF can not be sampled.



**Figure 19. Coil Current Zero Crossing Too Short for BEMF Measurement**

Figure 20 displays how the MinSamples setting can solve this issue. By setting MinSamples to a value higher than the clamp time + the voltage transient time, the real BEMF can

be sampled. Because MinSamples is in this case larger than one microstep, the coil current zero crossing will be stretched. Notice however that the speed did not change.



**Figure 20. Stretching the Coil Current Zero Crossing**

Stretching of the coil current zero crossing will not only be needed when the speed is set too high. If the supply voltage  $V_{BB}$  is too low, BEMF is high or the serial resistance of the coil  $R_i$  is high, the voltage across the coil will be too low resulting in a too slow current decay. If coil inductance is too high, this could also lead to a too slow current decay.

Similar effect will happen if the coil current is set too high. Figure 21 displays good BEMF sampling. The voltage transient phase has ended before the next microstep.

Figure 22 displays what would happen if the coil current is doubled (all other operating parameters are kept similar). The coil current decay will be as fast as in Figure 21 but because the coil current has to decay from a higher level to the zero current, it will take longer to reach the zero current. Because of this the voltage transient phase will start later (coil voltage is clamped longer to  $V_{BB} + 0.6\text{ V}$ ) and will not have ended before the next microstep. Figure 23 displays how MinSamples can solve this issue.

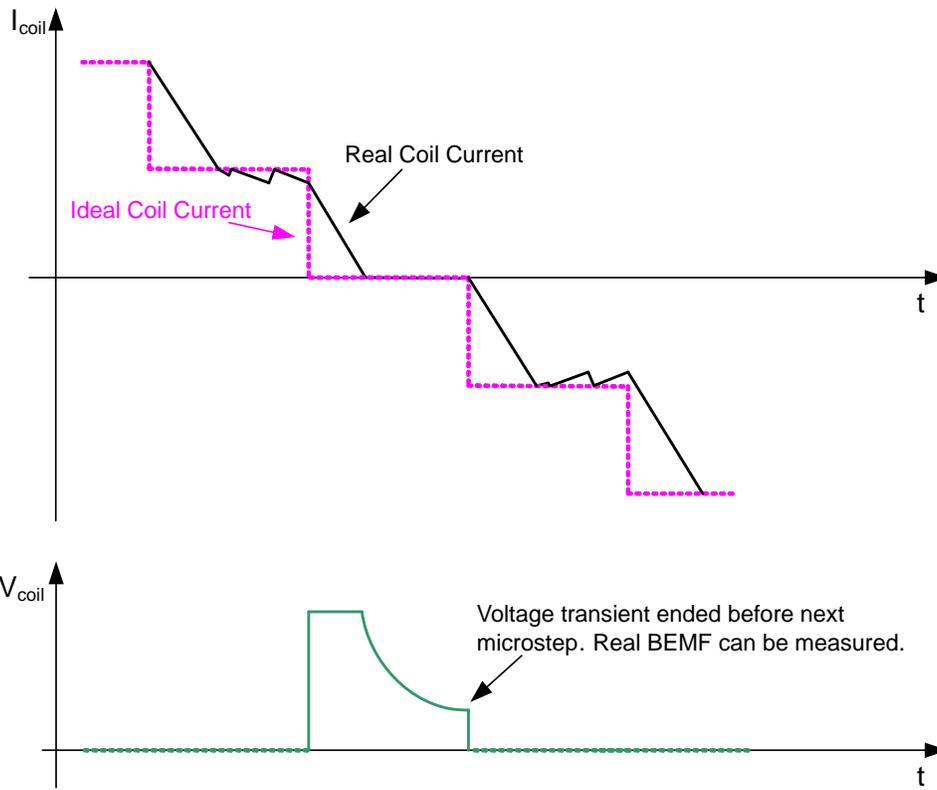


Figure 21. Correct BEMF Measurement

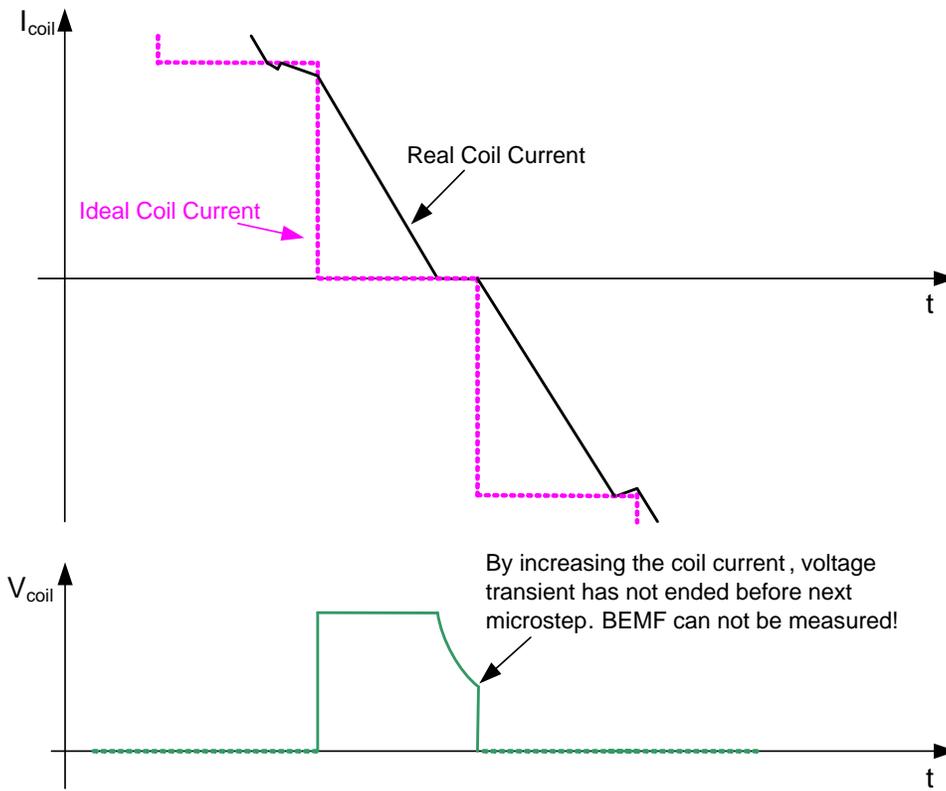


Figure 22. Incorrect BEMF Measurement Due to Too High Coil Current

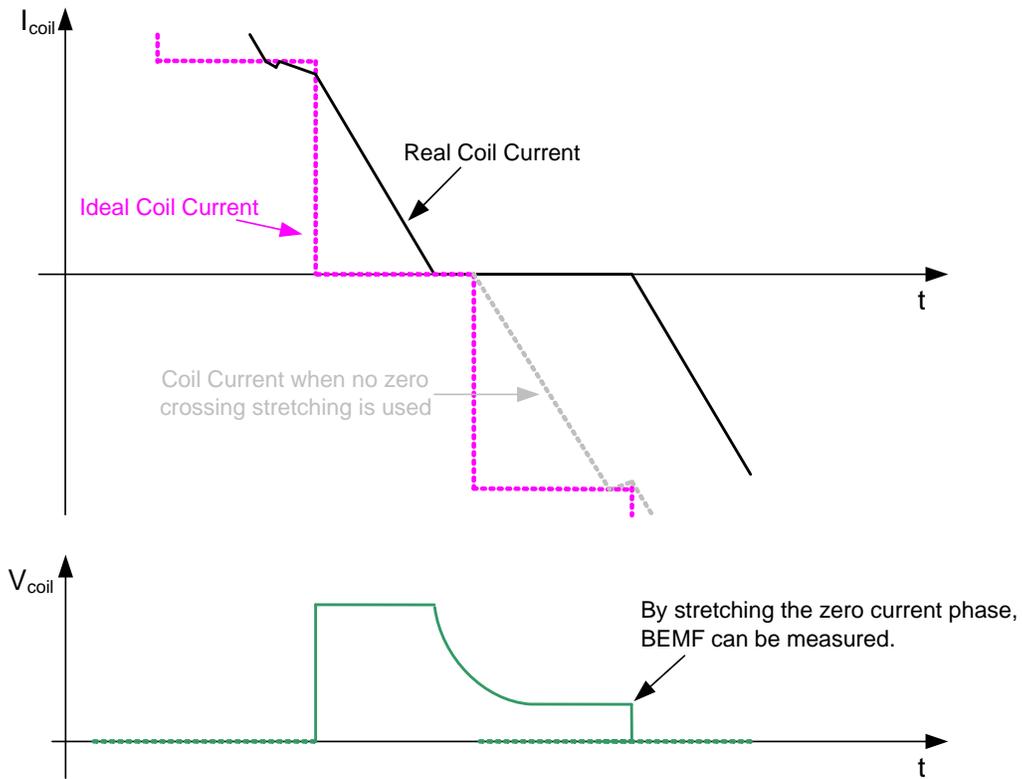


Figure 23. Correct BEMF Measurement at High Coil Current

Keep in mind that stretching the coil current zero crossing will not have an effect on the speed of the motor but will have an effect on the current shape and by this also on the motor performance.

**100% Duty Cycle**

AMIS-30623/AMIS-30624/NCV70627 will skip stall detection when 100% PWM duty cycle is detected because this could result in incorrect triggering of the stall detection.

Figure 24 gives an example of 100% duty cycle. During the zero crossing there is still current present in the coil. The BEMF can not be measured.

Figure 25 gives an oscilloscope plot of a 100% duty cycle. Because the motor driver detects this 100% PWM duty cycle, stall detection will be skipped at the next zero crossing to prevent falls stall triggering.

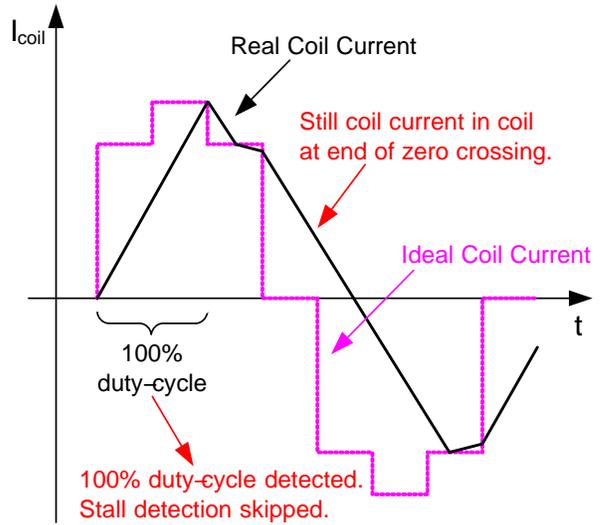


Figure 24. 100% PWM Duty Cycle

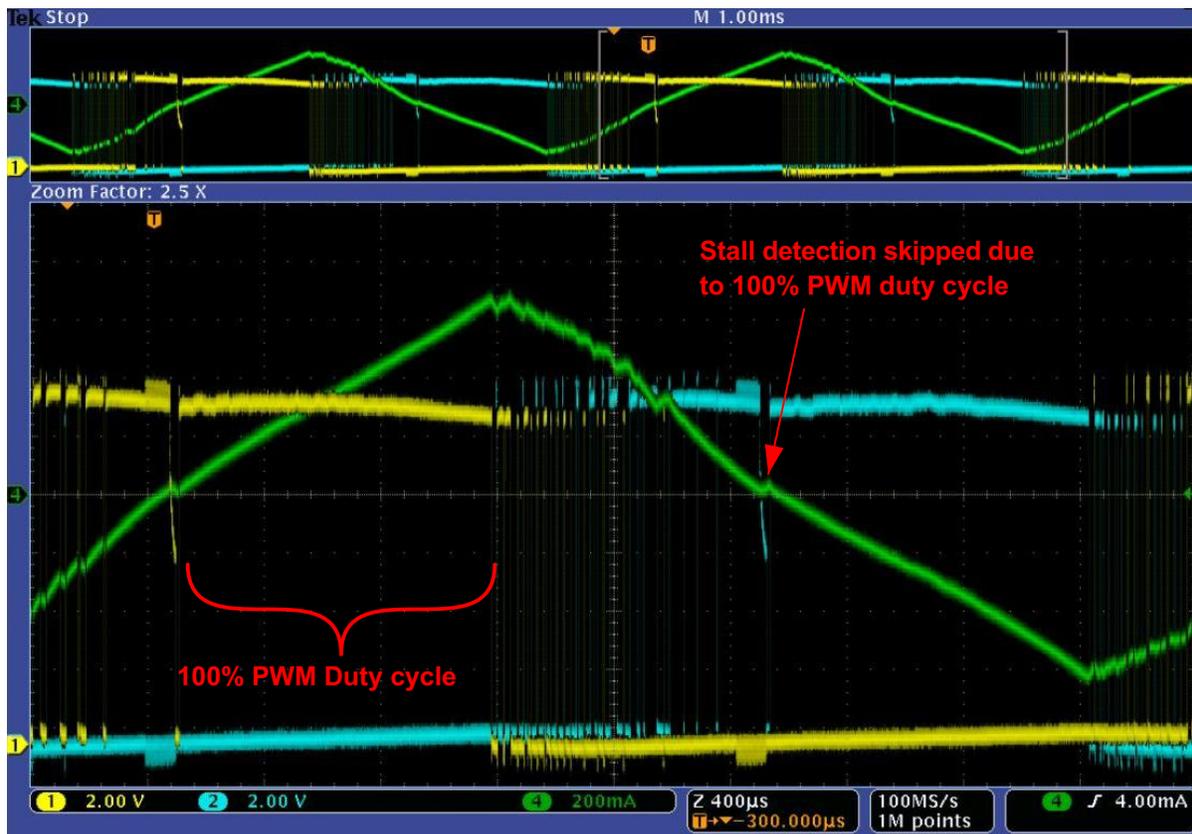


Figure 25. Oscilloscope Plot of 100% PWM Duty Cycle

If MinSamples is used to stretch the zero crossing, it's possible that the zero crossing can be made long enough to do a correct BEMF measurement.

In Figure 26 MinSamples is increased to stretch the zero crossing. Although the zero crossing is long enough to measure the BEMF, the motor driver will skip stall detection

due to the detection of 100% PWM duty cycle. It is however possible to enable stall detection all the time by setting the DC100StEn bit. This improves the sensitivity of the stall detection but can only be done if MinSamples can stretch the zero crossing long enough (see also Minimum Samples).



Figure 26. Oscilloscope Plot of 100% PWM Duty-Cycle

Ignoring the stall detection during 100% duty-cycle detection does not necessarily mean that stall can not be detected. If the motor gets blocked during a 100% duty-cycle, the BEMF drops to 0 V. Because of this the coil gets a larger voltage drop resulting in faster current slopes. In most cases this will result in a duty-cycle of less than 100% enabling stall detection again (Figure 27). If AbsThr is set stall will be detected (because BEMF is below AbsThr level).

Only in case the voltage supply is very low (too low to drive the motor in a correct way), the duty-cycle will still be 100% resulting in no stall detection.

Always keep in mind that driving a motor as given in Figures 25 and 26 will have an effect on the motor performance.

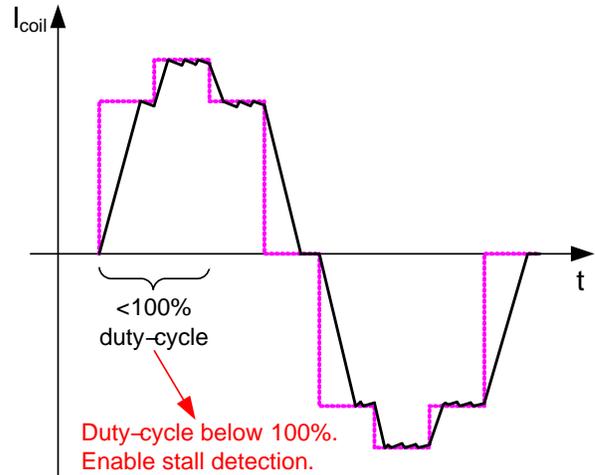


Figure 27. Blocked Rotor Result in Duty-Cycle Below 100%

## SET STALL PARAMETERS

This chapter will guide you step-by-step in defining the best stall parameters.

It is very important that stall detection parameters are verified under all operating conditions (voltage supply, temperature, ...). It is also important to define the stall parameters of the complete setup (and not only of the motor).

It is always important to operate a (stepper) motor application inside his operating range. Characterization of the complete setup has to be done to define this operating range. Application note [AND8371/D](#) can help you in characterizing the setup. Application note [AND8404/D](#) gives additional information in defining the motor parameters.

Using incorrect parameters or operating the stepper motor in an instable region will result in incorrect operation and incorrect stall detection.

To set the stall parameters an oscilloscope will be needed to measure the coil voltage, coil current (current probe) and the voltage on the SWI-pin. In all oscilloscope plots given in this document, the yellow and blue curves are the coil voltages measured on one coil, the green curve is the coil current through the same coil and the purple curve is the voltage measured on the SWI-pin of the motor driver.

It's assumed that the user has enough knowledge in using the stepper motor driver itself. The user should at least be confident with all AMIS-30623 and/or AMIS-30624 commands and parameters before continuing.

### Define MinSamples

The first parameter to be defined is MinSamples. It is very important to do this before defining AbsThr and DelThr. This because it's important to first make sure that the BEMF is measured and not the voltage transient (see Minimum Samples).

Step 1: Set the motor parameters by using the SetMotorParam command. Make sure the correct motor parameters are used (see application note [AND8404/D](#) for more info)

Step 2: Set all stall parameters to 0x00.

Step 3: To define MinSamples, the motor should be running at maximum velocity ( $V_{max}$ ). Use the RunVelocity command in case the stepper motor can rotate freely (no mechanical end positions).

If the stepper motor can not spin freely (mechanical end positions), the SetPosition command has to be used. Move the rotor between two positions without hitting a mechanical end position. Make sure that the movement is long enough so  $V_{max}$  is reached (see also Figure 35). It's possible that the SetPosition command has to be initiated several times to define the MinSamples parameter.

Step 4: When spinning at maximum velocity ( $V_{max}$ ), measure the coil voltage and coil current. Measure the time it takes for the voltage transient to end. Set MinSamples to the smallest possible value but high enough so the voltage transient has ended.

Figure 28 gives an oscilloscope plot of a zero current crossing. Notice that the voltage transient has ended before the zero current phase is left. Figure 29 is a more detailed zoom of the zero current crossing of Figure 28.

As can be seen in Figure 29, a BEMF sampling delay (MinSamples) of about 150  $\mu s$  is a good value. If MinSamples is set too low, it's possible that the voltage transient is measured instead of the BEMF resulting in incorrect stall detection.

In Figure 30 the speed ( $V_{max}$ ) and coil current ( $I_{run}$ ) are increased resulting in a too short zero crossing (MinSamples was set back to 0x00). The voltage transient has not ended before the next microstep is taken. By increasing the MinSamples to a value of about 200  $\mu s$ , the zero current phase is stretched and the BEMF can be measured (see Figure 31).

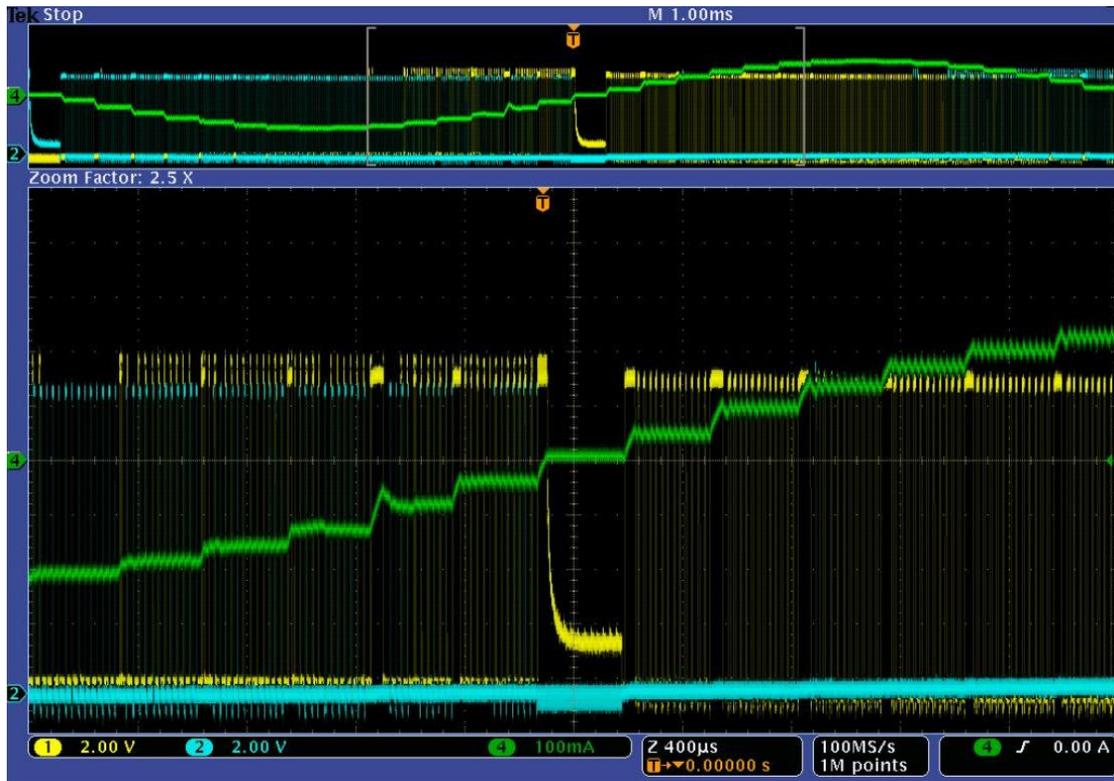


Figure 28. Oscilloscope Plot of Zero Current Crossing

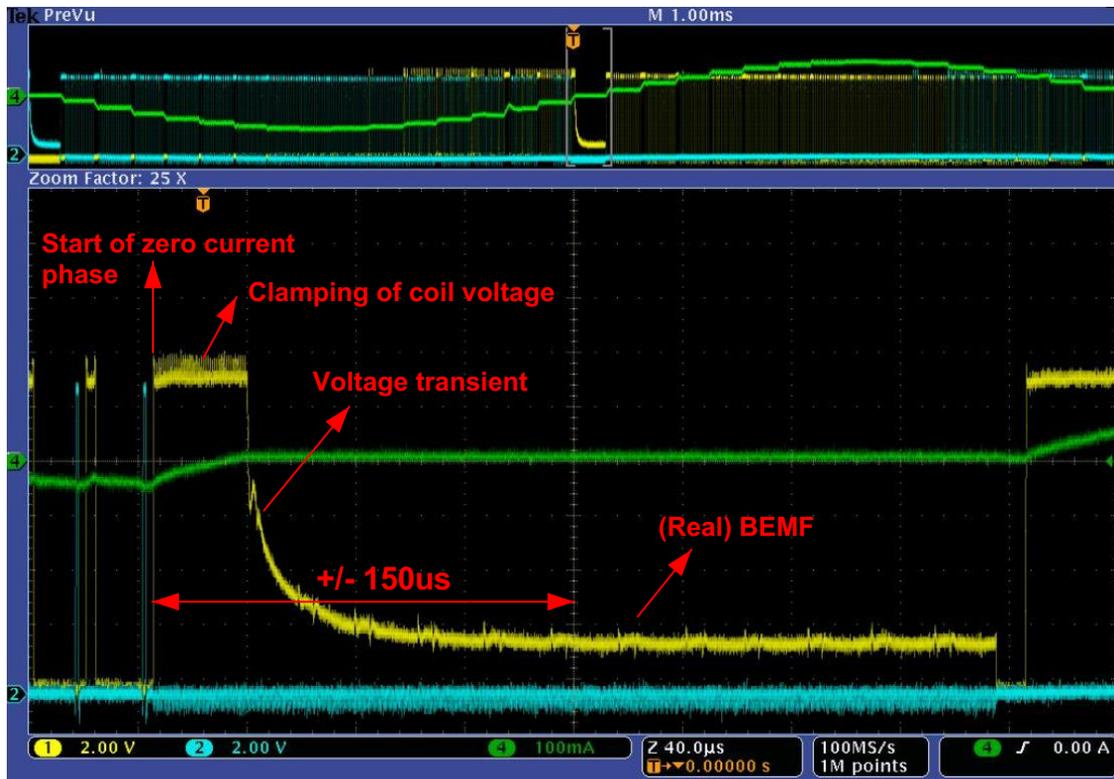


Figure 29. Zoom on Zero Current Crossing

# AND8471/D

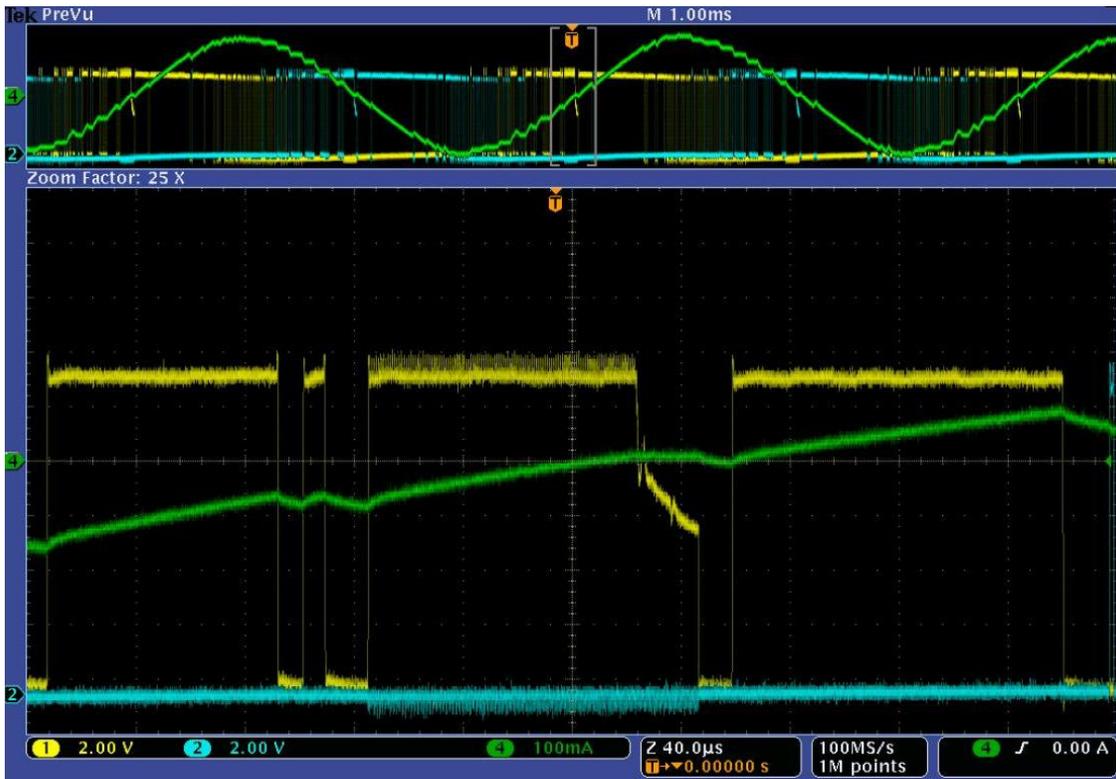


Figure 30. Too Short Zero Crossing

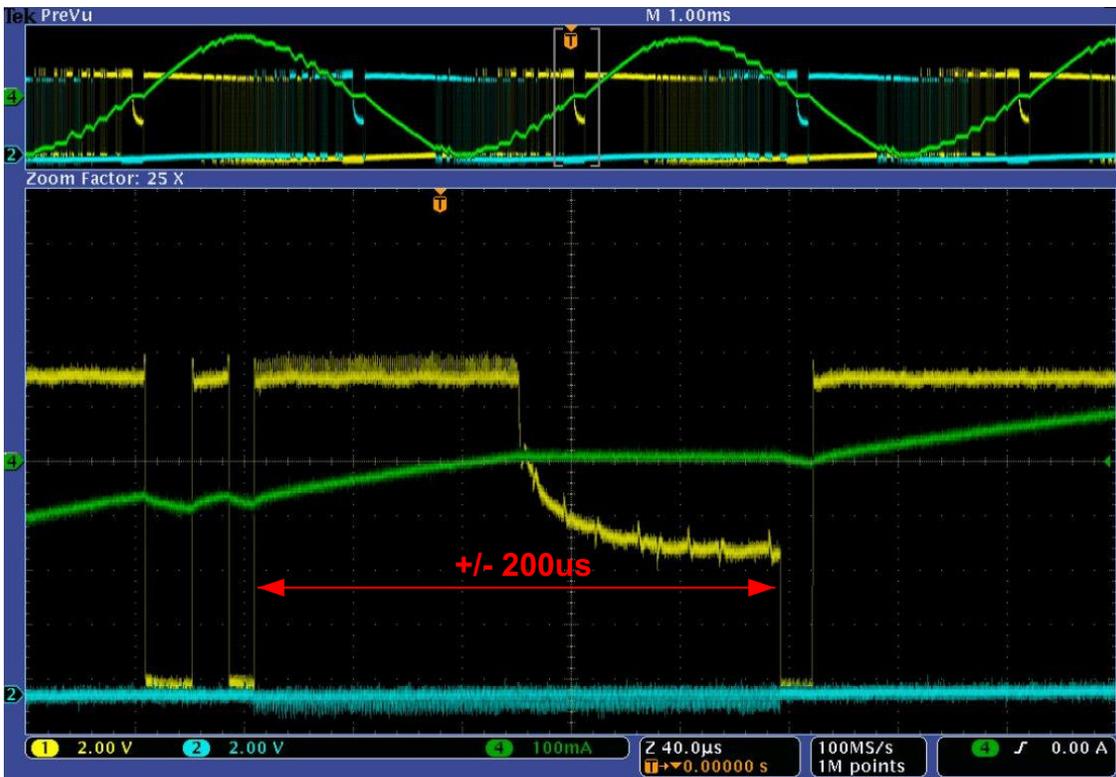


Figure 31. Zero Current Stretched

**Define FS2StallEn**

It's advised to define FS2StallEn before defining AbsThr and DelThr. Follow below steps to do this.

- Step 1: Set the motor parameters similar to the parameters taken to define MinSamples.
- Step 2: Set all stall parameters to 0x00 except for MinSamples (defined in previous section).
- Step 3: Send a TestBEMF command. This command will output the BEMF integrator of the motor driver on the SWI-pin. This can be used to observe the BEMF of the motor. See the motor driver datasheet for more info.
- Step 4: To define FS2StallEn, one must know when the acceleration phase ends. The amount of full steps the acceleration phase lasts can be calculated as next:

$$Nsteps = \frac{V_{max}^2 - V_{min}^2}{2 \times Acc}$$

- Step 5: Do a SetPosition and observe the BEMF on the SWI-pin during the acceleration phase and the beginning of the maximum velocity phase. Figure 32 displays an oscilloscope plot of the observed BEMF on the SWI-pin. Next settings where taken:

$$V_{max} = 395 \text{ FS/s}$$

$$V_{min} = 48 \text{ FS/s}$$

$$Acc = 19092 \text{ FS/s}^2$$

This will result in an acceleration phase of about 4 full steps.

Notice that the acceleration is fast and the BEMF is lagging behind (BEMF is not at his maximum level when acceleration phase has ended). Because it's best to enable stall detection only when the BEMF is at the (highest) level, FS2StallEn should be set to 7 full-steps in this example.

Keep in mind that the BEMF represents the real movement of the rotor. If the BEMF is lagging behind, this means that the real velocity of the rotor is also lagging behind on the desired (theoretical) velocity.

In Figure 33 acceleration was lowered to 6228 FS/s<sup>2</sup>. Acceleration phase will take 12 full steps. Notice that the BEMF is not lagging behind too much. MinSamples can be set to 3 full steps.

Figure 34 gives an example of bad acceleration. Very fast acceleration was done to a high velocity resulting in a lot of overshoots. If this happens it's best to reduce acceleration. If this can not be done, set FS2StallEn to the maximum (7 full steps). Keep in mind however that this will reduce stall accuracy. Also keep in mind that the BEMF represents the real movement of the rotor. If overshoots are seen on the BEMF, these overshoots will also be present in the real movement of the rotor.

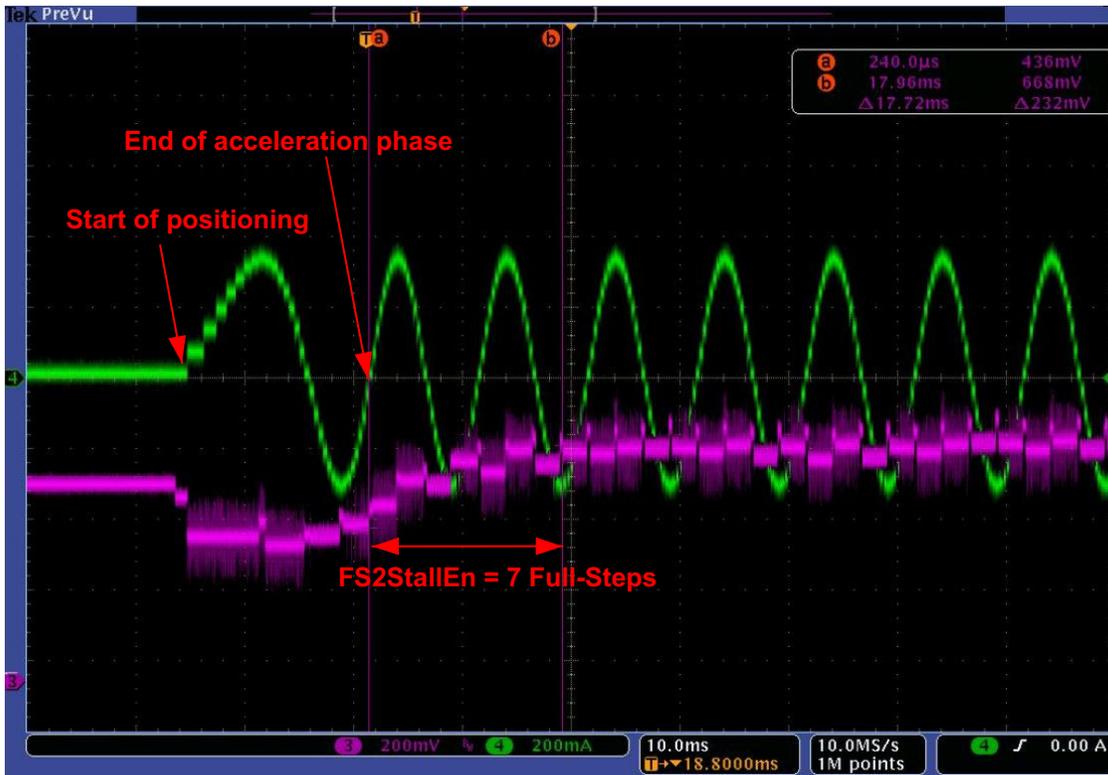


Figure 32. Observation of BEMF on SWI-Pin

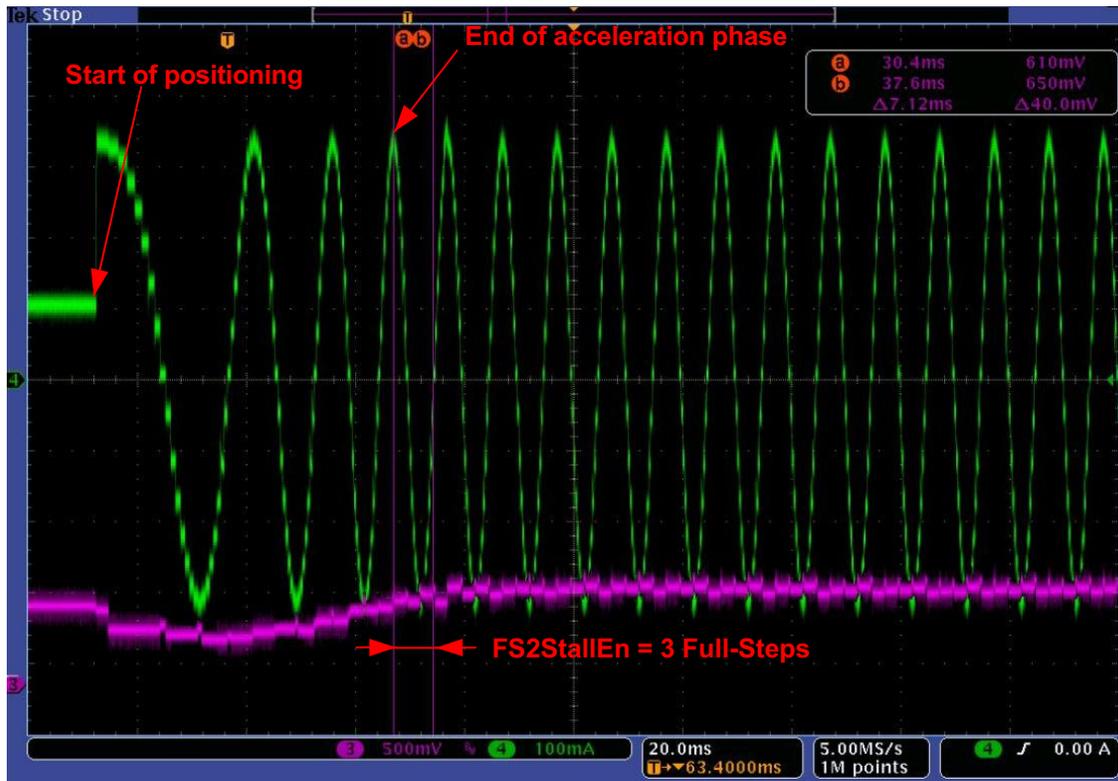


Figure 33. Slower Acceleration Results in Less Lagging Behind of the BEMF



Figure 34. Too Fast Acceleration Results in Overshoots

**Define AbsThr**

Follow next steps to define the absolute threshold.

Step 1: Set the motor parameters similar to the parameters taken to define MinSamples.

Step 2: Set all stall parameters to 0x00 except for MinSamples and FS2StallEn which are defined above.

Step 3: When the stepper motor can rotate freely (no mechanical end positions), send the RunVelocity command and go to Step 4a (do not execute Step 4b).

If the stepper motor can not spin freely (mechanical end positions), the SetPosition command has to be used. To define the AbsThr parameter, the motor has to be moved between two positions without hitting a mechanical end position. The movement has to be long enough to make sure stall detection is enabled. A good positioning is given by the purple curve in Figure 35. The grey curves give a too short positioning.

Go to Step 4b.

Step 4a: When spinning at maximum velocity ( $V_{max}$ ) increase AbsThr value by one. If the motor is still rotating, increase AbsThr again. Do this until the motor stalls without hitting an end position (= falls stall). If falls stall is detected, the AbsThr value is set too high. Decrease the (hexadecimal) value by 2 (if possible) to have a good AbsThr setting. See also Figure 36.

Although the best value for FS2StallEn was already defined in Define FS2StallEn, it's still possible that oscillations after acceleration trigger a falls stall. Do a SetPosition command and verify if the motion was done correctly. If not, oscillations triggered an absolute stall. Decrease AbsThr or increase FS2StallEn until a SetPosition is done correctly.

Step 4b: Set the AbsThr to the lowest value (0.5 V) and do a SetPosition. If the motion was completed correctly, increase AbsThr by one and repeat the SetPosition. Do this until a stall was detected without hitting an end position (= falls stall). If falls stall is detected, the AbsThr value is set too high. Decrease the (hexadecimal) value by 2 (if possible) to have a good AbsThr setting. Figure 37 gives a better view on this.

Remark: If the motor already stalls at the smallest AbsThr value (0.5 V), stall detection can not be used because BEMF is too low. Use a stepper motor which generates a higher BEMF or operate the motor at a higher speed.

It's possible that a stepper motor generates a high enough Bemf (higher than 0.5 V) but due to overshoots the absolute stall is triggered. See Define FS2StallEn (page 23) how this can be observed.

Stall parameters of a free running motor (no end positions) can also be defined by using the SetPosition command (as used for a non-free running stepper motor).

As given in Delta Threshold (page 7), sometimes it's best to set AbsThr level to the lowest value (0.5 V) and use it only to detect a blocked rotor at start of the rotation. DelThr will be used to detect a blocked rotor during rotation.

Hint: To detect if an Absolute stall occurred, the GetFullStatus(2) command can be used.

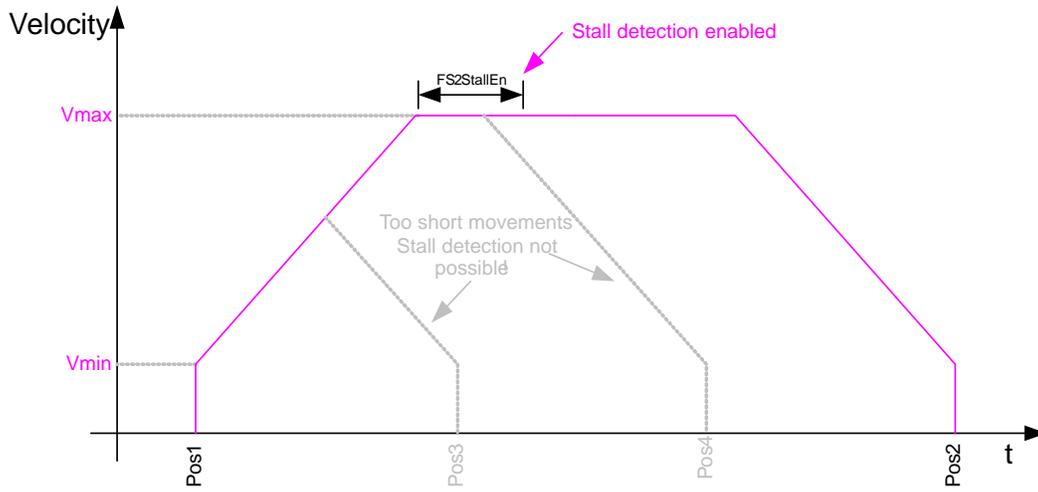


Figure 35. Define AbsThr by Using SetPosition Command

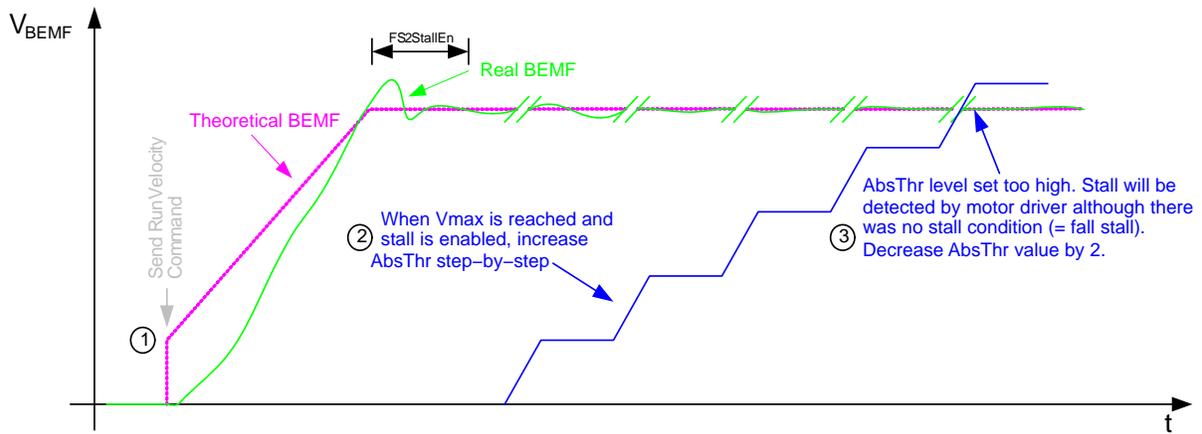


Figure 36. Define AbsThr For a Free Running Motor

# AND8471/D

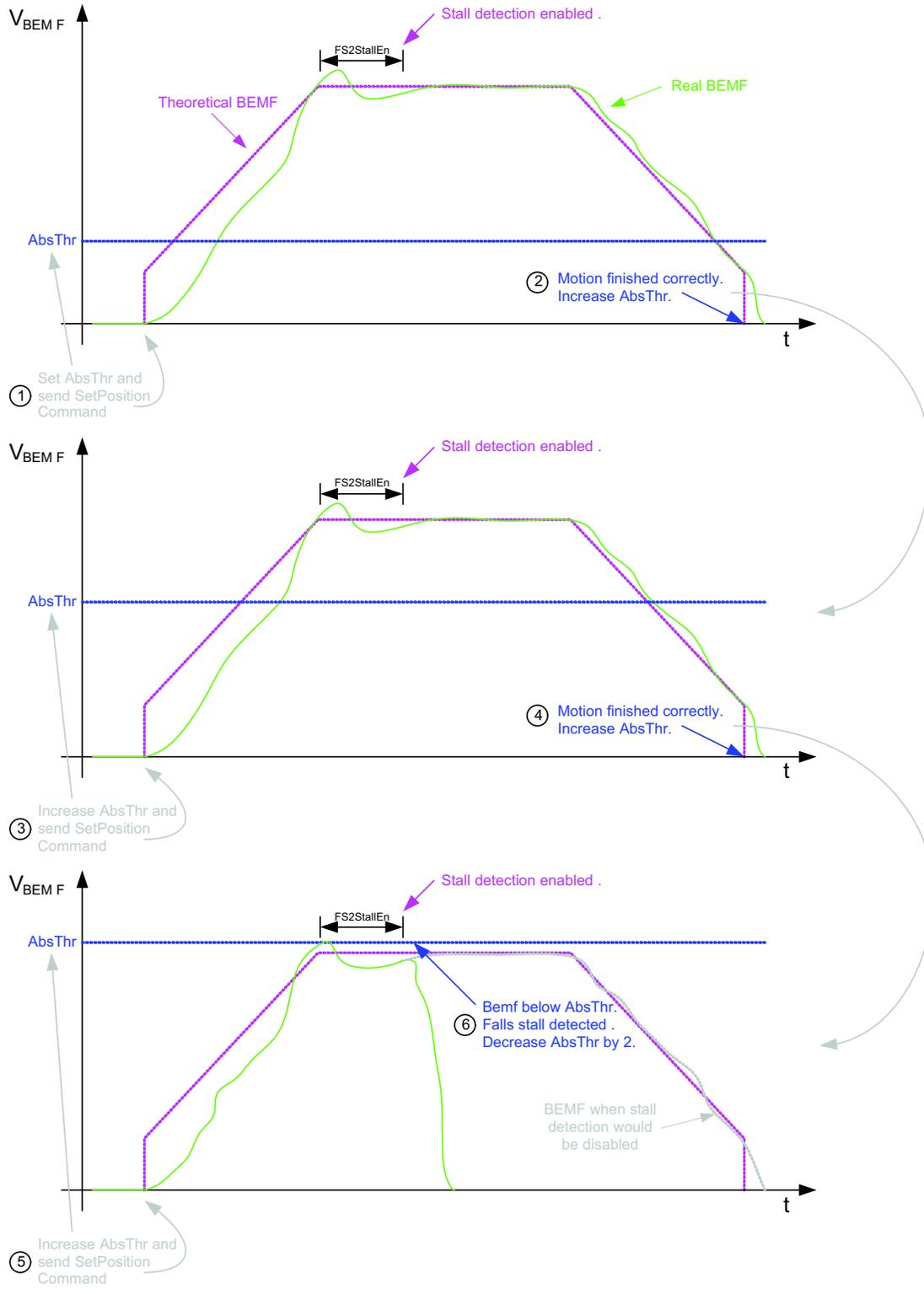


Figure 37. Define  $AbsThr$  for a Non-Free Running Motor (End Positions)

**Define DelThr (Not Applicable for the NCV70627)**

Follow next steps to define the delta threshold.

Step 1: Set the motor parameters similar to the parameters taken to define MinSamples.

Step 2: Set all Stall parameters to 0x00 except for MinSamples, FS2StallEn and AbsThr which were defined above.

Step 3: When the stepper motor can rotate freely (no mechanical end positions), send the RunVelocity command and go to Step 4a (do not execute Step 4b).

If the stepper motor can not spin freely (mechanical end positions), the SetPosition command has to be used.

To define the AbsThr parameter, the motor has to be moved between two positions without hitting a mechanical end position. The movement has to be long enough to make sure stall detection is enabled. A good positioning is given by the purple curve in Figure 35. The grey curves give a too short positioning.

Go to Step 4b.

Step 4a: When spinning at maximum velocity ( $V_{max}$ ), set DelThr to the maximum value. If the motor is still rotating, decrease DelThr. Do this until the motor stalls without hitting an end position (= falls stall). If falls stall is detected, the DelThr value is set too narrow. Increase the (hexadecimal) value by 2 (if possible) to have a good DelThr setting. See also Figure 38.

Although the best value for FS2StallEn was already defined in previous section, it's still possible that oscillations after acceleration trigger a falls stall. Do a SetPosition command and verify if the motion was done correctly. If not, oscillations triggered a delta stall. Increase DelThr or increase FS2StallEn until a SetPosition is done correctly.

Step 4b: Set DelThr to the maximum value and do a SetPosition. If the motion was completed correctly, decrease DelThr by one and repeat the SetPosition. Do this until a stall was detected without hitting an end position (= falls stall). If falls stall is detected, the DelThr value is set too high. Increase the (hexadecimal) value by 2 (if possible) to have a good DelThr setting. Figure 39 gives a better view on this.

Remark: If the motor already stalls at the highest DelThr value, this indicates that too many oscillations are present on the BEMF (and by this also on the real movement of the rotor). This can be caused due to too fast acceleration or because the motor is operated inside an instable region. See Define FS2StallEn how this can be observed.

Stall parameters of a free running motor (no end positions) can also be defined by using the SetPosition command (as used for a non-free running stepper motor).

Hint: To detect if an Delta stall occurred, the GetFullStatus(2) command can be used.

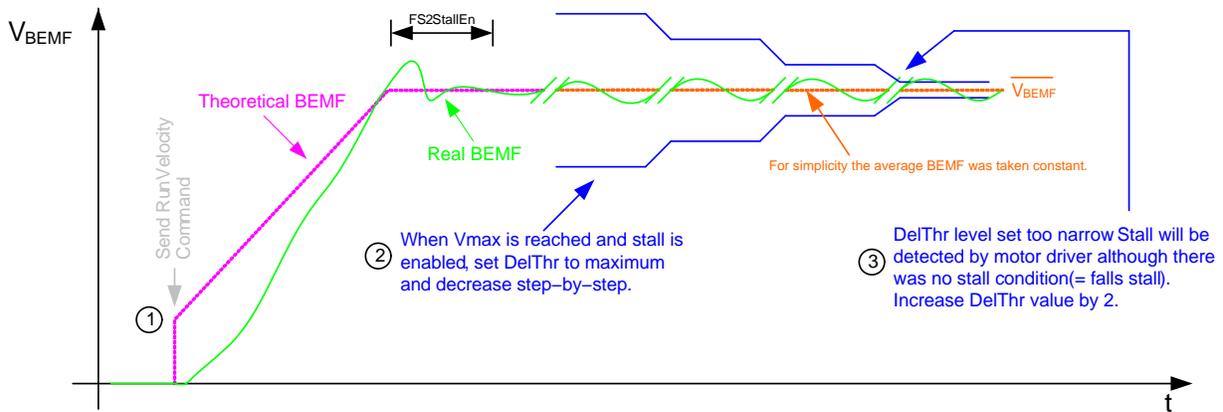


Figure 38. Defining DelThr for a Free Running Motor (Not Applicable for the NCV70627)

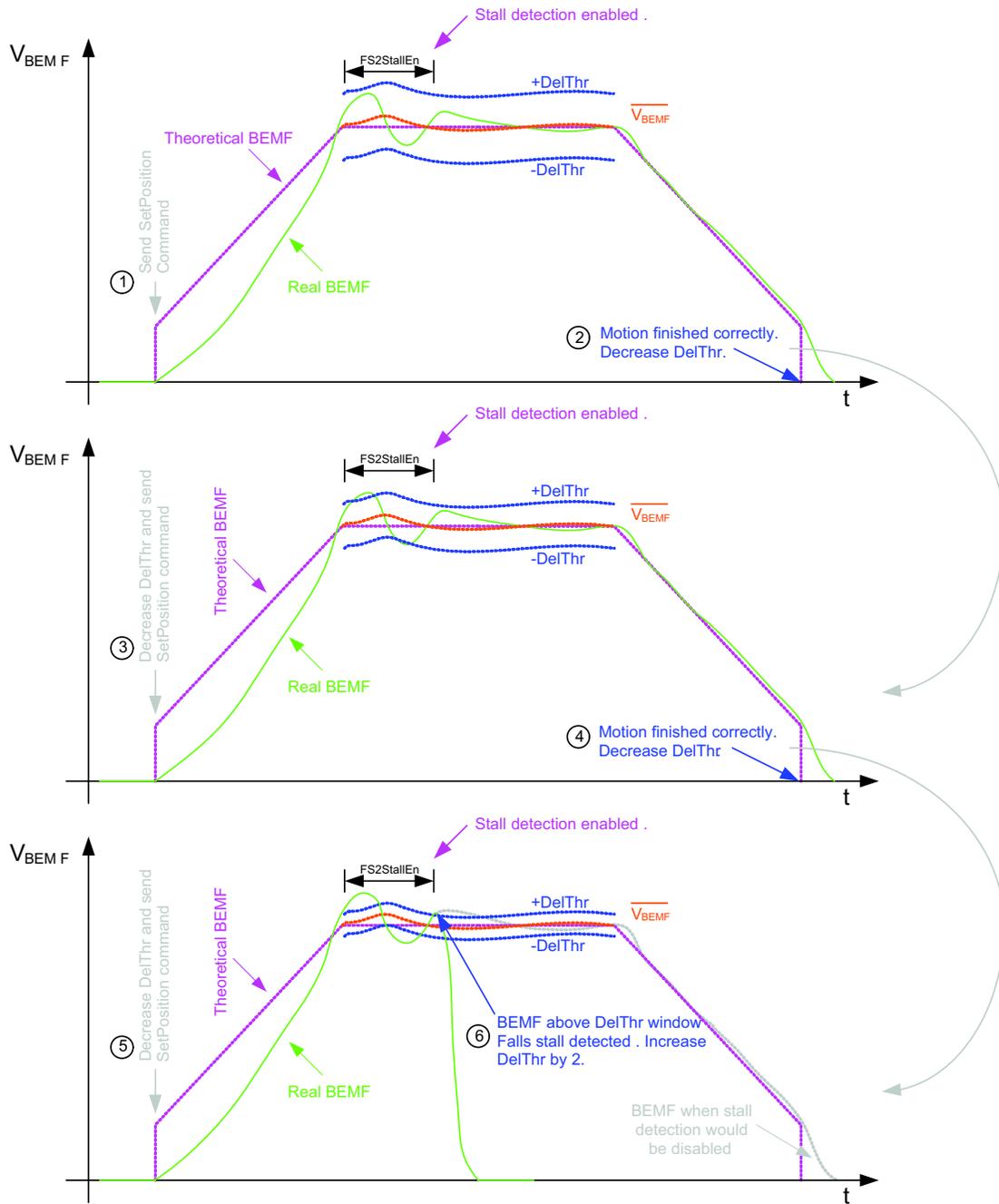


Figure 39. Define DelThr for a Non-Free Running Motor (Not Applicable for the NCV70627)

**Define DC100StEn**

If the BEMF can be sampled in a correct way under all operating conditions (voltage supply, temperature, ...), enable the DC100StEn bit. This makes sure that stall detection is performed every zero crossing.

If MinSamples can not be stretched long enough or if it's possible that under certain conditions BEMF can not be measured (e.g. voltage dips), clear DC100StEn bit. This makes sure that if 100% PWM duty cycle is noticed, stall detection is skipped to prevent falls stalls.

# AND8471/D

## REFERENCES

- [1] Data Sheet [AMIS-30623/D](#)
- [2] Data Sheet [AMIS-30624/D](#)
- [3] Data Sheet [NCV70627/D](#)
- [4] Application Note [AND8371/D](#), Stepper Motor Resonance Measurement Setup with the AMIS-3052x/NCV7052x Evaluation Kit.
- [5] Application Note [AND8404/D](#), AMIS-3062x Micro-Stepping Motor Driver Family, Robust Motion Control Using the AMIS-3062x.

ON Semiconductor and the  are registered trademarks of Semiconductor Components Industries, LLC (SCILLC) or its subsidiaries in the United States and/or other countries. SCILLC owns the rights to a number of patents, trademarks, copyrights, trade secrets, and other intellectual property. A listing of SCILLC's product/patent coverage may be accessed at [www.onsemi.com/site/pdf/Patent-Marking.pdf](http://www.onsemi.com/site/pdf/Patent-Marking.pdf). SCILLC reserves the right to make changes without further notice to any products herein. SCILLC makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does SCILLC assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation special, consequential or incidental damages. "Typical" parameters which may be provided in SCILLC data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. SCILLC does not convey any license under its patent rights nor the rights of others. SCILLC products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the SCILLC product could create a situation where personal injury or death may occur. Should Buyer purchase or use SCILLC products for any such unintended or unauthorized application, Buyer shall indemnify and hold SCILLC and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that SCILLC was negligent regarding the design or manufacture of the part. SCILLC is an Equal Opportunity/Affirmative Action Employer. This literature is subject to all applicable copyright laws and is not for resale in any manner.

## PUBLICATION ORDERING INFORMATION

### LITERATURE FULFILLMENT:

Literature Distribution Center for ON Semiconductor  
19521 E. 32nd Pkwy, Aurora, Colorado 80011 USA  
**Phone:** 303-675-2175 or 800-344-3860 Toll Free USA/Canada  
**Fax:** 303-675-2176 or 800-344-3867 Toll Free USA/Canada  
**Email:** [orderlit@onsemi.com](mailto:orderlit@onsemi.com)

**N. American Technical Support:** 800-282-9855 Toll Free  
USA/Canada  
**Europe, Middle East and Africa Technical Support:**  
Phone: 421 33 790 2910  
**Japan Customer Focus Center**  
Phone: 81-3-5817-1050

**ON Semiconductor Website:** [www.onsemi.com](http://www.onsemi.com)  
**Order Literature:** <http://www.onsemi.com/orderlit>  
For additional information, please contact your local Sales Representative