Driving high inertia loads with iMOTION™ 2.0

A guide for iMOTION, MCEWizard and MCEDesginer

About this document

Scope and purpose

This application note will help you drive high Inertia load applications such as fans with iMOTION™ 2.0

Intended audience

Power electronics engineers, motor control engineers, iMOTION™ learners.

Table of contents

1 Introduction

High inertia load applications such as ceiling fans can be quite tricky to understand in terms of control theory. Starting the motor or accurately attaining a stable speed can be challenging in these applications. In this application note we will review high inertia load theory, and how to use the Infineon iMOTION™ 2.0 ecosystem to run such a load.

A high inertia load represents a high resistance to acceleration. For a fan application, the high inertia can result from a heavy blade or a blade with large surface area representing a large amount of air friction. Starting a high inertia load from rest requires an understanding of how torque is applied to the rotor to move the load. The starting time is the time it takes for the rotor and load to go from rest to motion after a start command is issued. This starting time is a function of the load inertia, the torque capability of the motor and the current driving capability of the inverter circuit. For the same voltage class, a small inverter would drive a smaller current than a large inverter and hence require a longer starting time.

Most motors used in modern small home appliances are of the brushless DC (BLDC) or permanent magnet AC (PMAC) kind. Sensorless control of these motors requires the flux angle to be properly estimated from measured motor current via the inverter shunt resistance. Starting a motor with a high inertia load correctly and achieve reliable flux angle estimation can be challenging.

An easy solution for these problems is to select a larger motor and larger inverter. But this is impractical and not commercially viable. Using Infineon’s iMOTION™ 2.0 and with proper tuning of the motor control, we can achieve the right equilibrium between starting the motor under a heavy load, achieving a reliable flux angle estimate and a good response to target speed commands.

In this application note, we will study the startup of a 150 W high inertia fan with IMM101T-046, iMOTION™ Smart IPM.

2 EVAL-IMM101T-046

The EVAL-IMM101T starter kit is designed to be an easy-to-use motor-drive solution based on Infineon's IMM101T Smart IPM. The board is equipped with everything needed for sensorless field-oriented control (FOC). It contains a single-phase AC connector, EMI filter, rectifier and 3-phase output to connect to the motor. The power stage also contains a source shunt for current sensing and a voltage divider for DC-link voltage measurement.

The combination of the MCE2.0 iMOTION™ motion control engine integrated into the IMM101T devices, together with the gate driver and six MOSFETs offers a complete motor drive system in a compact 12 x 12 mm^2 surface mount package minimizing the number of external components and PCB area. The block diagram of the EVAL-IMM101T is depicted in Figure 1.

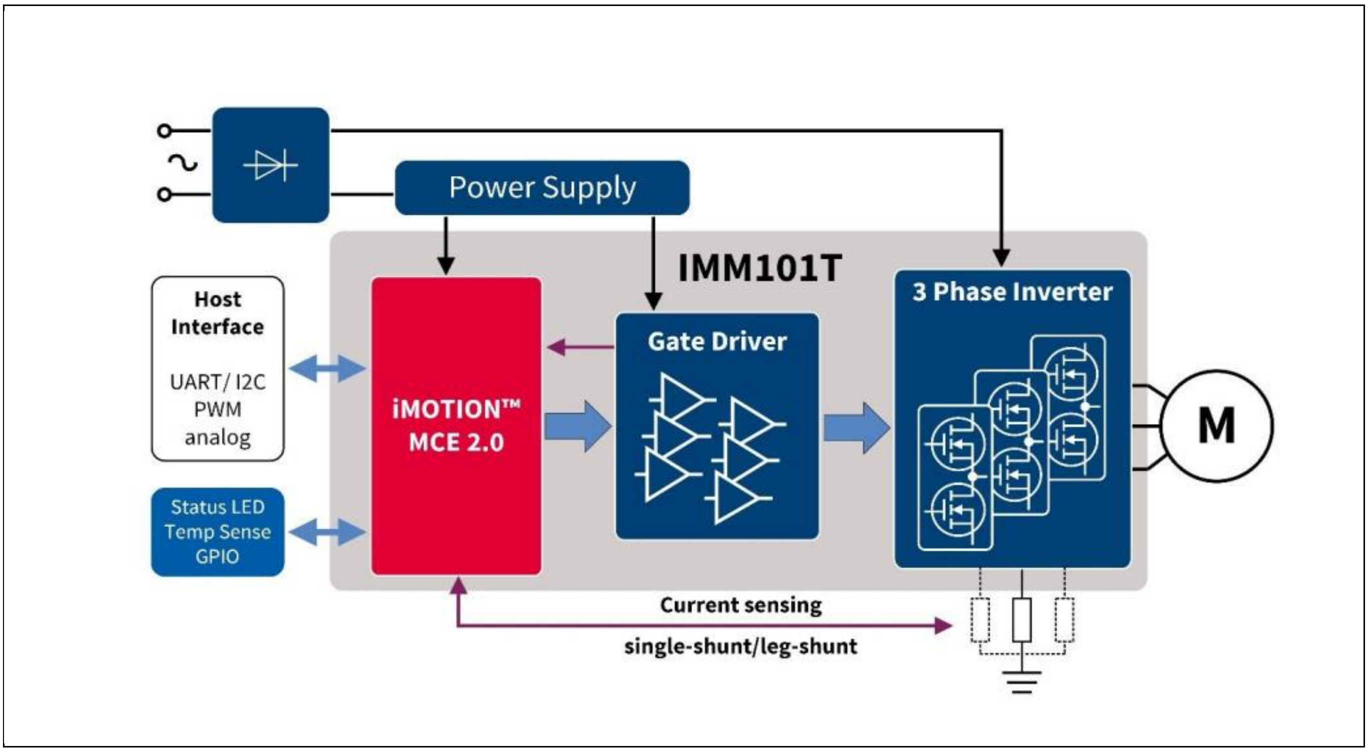


Figure 1 EVAL-IMM101T Block Diagram

2.1 Main features

EVAL-IMM101T Starter Kits are intended for evaluation of the IMM101T series of iMOTION™ Smart IPMs. The main features of the IMM101T series include:

- Motion control engine (MCE2.0) as ready-to-use controller solution for variable speed drives
- Field-oriented control (FOC) for permanent magnet synchronous motor (PMSM)
- Space vector PWM with sinusoidal commutation and integrated protection features
- Current sensing via single or leg shunt through direct interface
- Sensorless operation
- Support for digital Hall sensors
- Integrated analog comparators for overcurrent protection
- Over-/Undervoltage protection
- Rotor lock protection
- Built-in temperature sensor (over-temperature protection)
- Undervoltage lockout
- Integrated minimum dead time
- Shoot-through prevention
- 3 different power MOSFET options: 4.8 Ω / 500 V, 1.26 Ω /650 V and 0.86 Ω /65 0V (typical values at 25°C)
- 3.3 V or 5.0 V supply voltage options for controller
- 15 V supply voltage for gate driver
- Integrated bootstrap diode structure
- Flexible host interface options for speed commands: UART, SPI, PWM or analog signal
- Class B pre-certification for MCE2.0 firmware
- Isolation 1500 VRMS 1min
- Very compact 12x12 mm^2 PQFN package

The main characteristics of the evaluation board include:

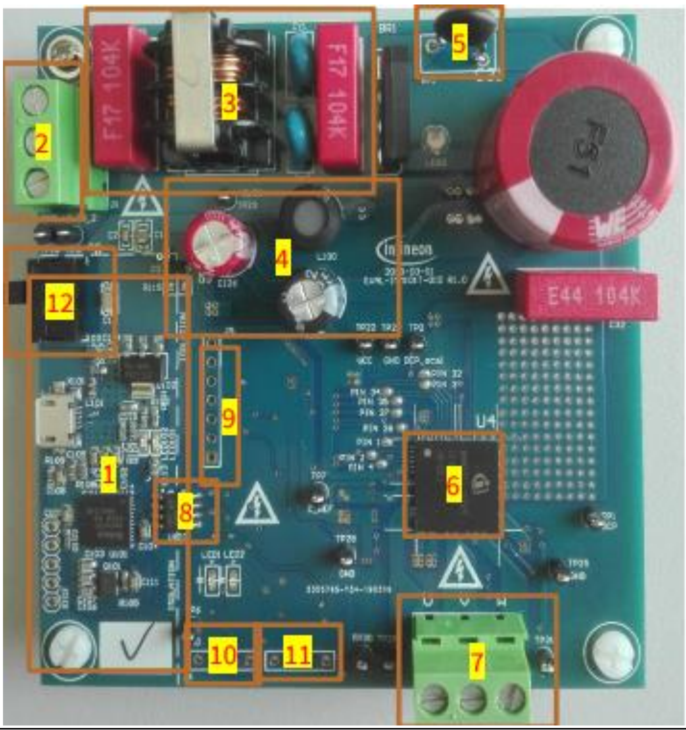
- Nominal input voltage 110 VAC – 230 VAC
- On-board EMI filter
- Single-shunt, current-sensing configuration by default
- Voltage divider for DC-link voltage sensing
- Measurement test points compatible with standard oscilloscope probes
- On-board debugger with 1 kV isolation for isolated communication to PC via USB
- NTC to reduce inrush current
- PCB size is 88.9 mm x 89.5 mm, 2 layers, 2 oz copper
- RoHS compliant

Table 1 lists all the key specifications of the evaluation board EVAL-IMM101T-046.

Table 1 EVAL-IMM101T-046

Parameters	Device	Values	Conditions
Input			
Voltage	EVAL-IMM101T-015	110 – 230 V_{AC}	
	EVAL-IMM101T-046		
Output			
Current per phase	EVAL-IMM101T-015	355 mA_{rms}	t_{amb} = 25°C, $t_{hotspot}$ = 105°C, V_{DCbus} = 300 V, 2phase modulation, 6kHz PWM
	EVAL-IMM101T-046	600 mA_{rms}	
DC Bus Voltage			
Maximum DC bus voltage	EVAL-IMM101T-015	380 V	
	EVAL-IMM101T-046		
Minimum DC bus voltage	EVAL-IMM101T-015	120 V	
	EVAL-IMM101T-046		
Current feedback			
Current sensing resistor RS1	EVAL-IMM101T-015	500 mΩ	The current sensing default configuration is single shunt, only RS1 is mounted
	EVAL-IMM101T-046	250 mΩ	
On board power supply			
15 V	EVAL-IMM101T-015	15V±5%, max. 50mA	Used for Smart IPM gate driver
	EVAL-IMM101T-046		
3.3 V	EVAL-IMM101T-015	3.3V±2%, max. 20mA	Supplying the 3.3V to the controller
	EVAL-IMM101T-046		
PCB characteristics			
Material	EVAL-IMM101T-015	FR4, 2 layers, 2oz opper	
	EVAL-IMM101T-046		
Dimension	EVAL-IMM101T-015	88.9mm x 89.5mm	
	EVAL-IMM101T-046		
System enviroment			
Ambient temperature	EVAL-IMM101T-015	From 0 to 60°C	Non-condensing, maximum RH of 95%
	EVAL-IMM101T-046		

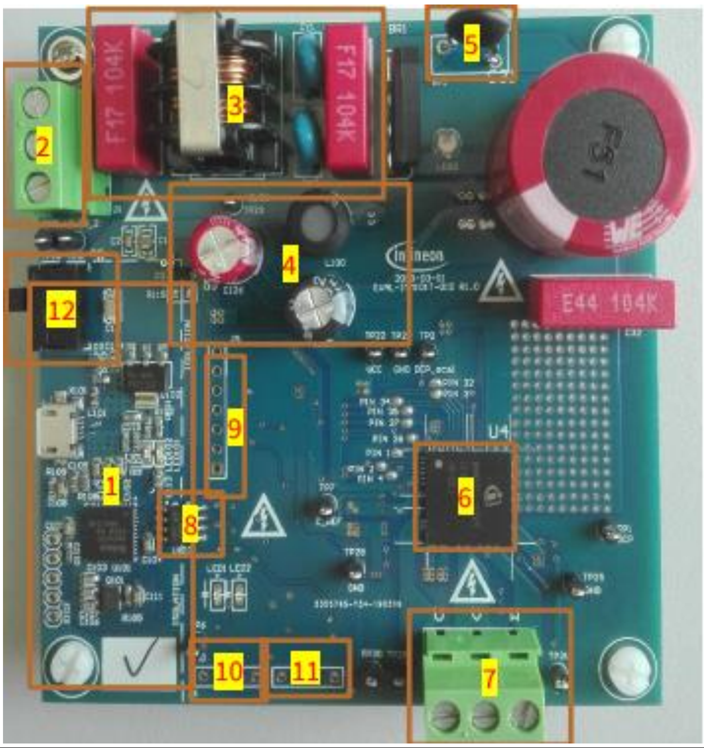
Figure 2 points out the functional groups on the top side of the EVAL-IMM101T-046 evaluation board.



1. Isolated on board debugger with USB connector
2. J1 – AC line connector
3. EMI filter
4. Auxiliary power supply capacitors
5. NTC
6. IMM101T-0xxM module
7. J3 – Motor Phase Connector
8. Optocoupler
9. J5 - connector
10. J6 - connector
11. J7 – connector
12. Isolated 3.3 V switch

Figure 2 Functional group of EVAL-IMM101T top side

Figure 3 points out the functional groups on the bottom side of the EvalIMM101T evaluation board.



1. Isolated on board debugger with USB connector
2. J1 – AC line connector
3. EMI filter
4. Auxiliary power supply capacitors
5. NTC
6. IMM101T-0xxM module
7. J3 – Motor Phase Connector
8. Optocoupler
9. J5 - connector
10. J6 - connector
11. J7 – connector
12. Isolated 3.3 V switch

Figure 3 Functional group of EVAL-IMM101T bottom side

Please refer to EVAL-IMM101T-046 for more information.

3 High inertia load control with iMOTION™

As explained in the introduction, a high inertia load can be challenging to control. Using iMOTION™ 2.0, we have to tune specific parameters correctly in order to have proper startup and speed control. Figure 4 shows some of the motor-specific parameters in MCEWizard, one of the iMOTION™ 2.0 development tools.

34 - Motor Rated Amps	0.625	Arms
35 - Motor Poles	8	
36 - Motor Stator Resistance	40	Ohms/phase
37 - Motor Lq Inductance	385	mH
38 - Motor Ld Inductance	368	mH
39 - Motor Back EMF Constant (Ke)	136	V(ln-rms)/krpm

Figure 4 MCEWizard motor parameters section for 150W motor

The parameters shown in the screenshot above indicate the motor has a high back EMF constant and large winding inductance. This implies that the motor has a low operating speed similar to one used in a ceiling fan. In a ceiling fan, the effect of gravity on the fan and a high air resistance create a high inertia load. The low running speed, low saliency (small discrepancy between L_d and L_q), plus the heavy fan load will make the startup and the speed loop unstable.

The first thing to do is run the motor unloaded to make sure all the entered motor parameters are correct. Once you have smooth motor operation without a load, the load can be introduced, and some parameters changed, to start up the system properly. We will discuss these parameter changes in Section 3.1.

It is very important that L_d, L_q and back EMF are precise values, as the angle estimation occurs at a very low speed for high inertia load motors. A good way to check if the parameters are correctly measured during the setup is to use MCEDesigner, and to check the Flx_M register value at nominal speed (Flx_M is dependent on speed). If all the motor parameters were measured correctly, the average Flx_M value should be 2048.

3.1 Startup settings

The speed controller calculates the motor torque required to follow the target speed, while the current loop drives the motor currents needed to generate this torque. The proportional plus integral (PI) speed loop compensator acts on the error between the target speed and the actual (estimated) speed. The integral term forces the steady state error to zero, while the proportional term improves the high-frequency response. Depending on the motor and load characteristics, the PI compensator gains are adjusted to meet the target dynamic performance. The limiting function on the output of the PI compensator prevents integral windup, and maintains the motor currents within the motor and drive capability.

The current controller calculates the inverter voltages to drive the motor currents needed to generate the desired torque. Field-oriented control (FOC) uses the Clarke transform and a vector rotation to transform the motor winding currents into two quasi components, an Id component that reinforces or weakens the rotor field, and an Iq component that generates motor torque.

3.1.1 Current loop initialization

Now that the unloaded motor is running well, we focus on the motor startup with the load. We will disable the speed loop and start the motor in current mode control. Once motor startup is working, we can include the speed loop to tune the response to speed commands.

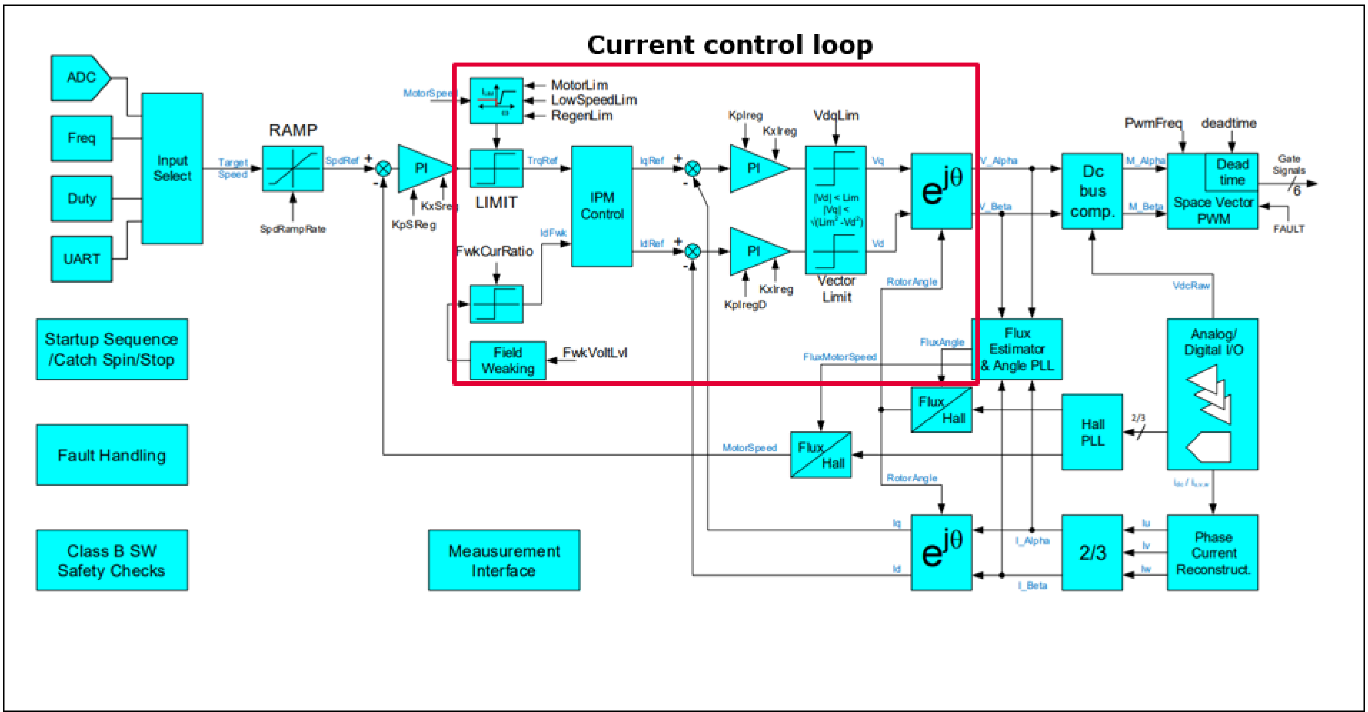


Figure 5 **Top level diagram of speed-control loop and sensorless FOC**

In order to deactivate the speed Loop, a current control startup procedure can be set up in MCEDesigner.

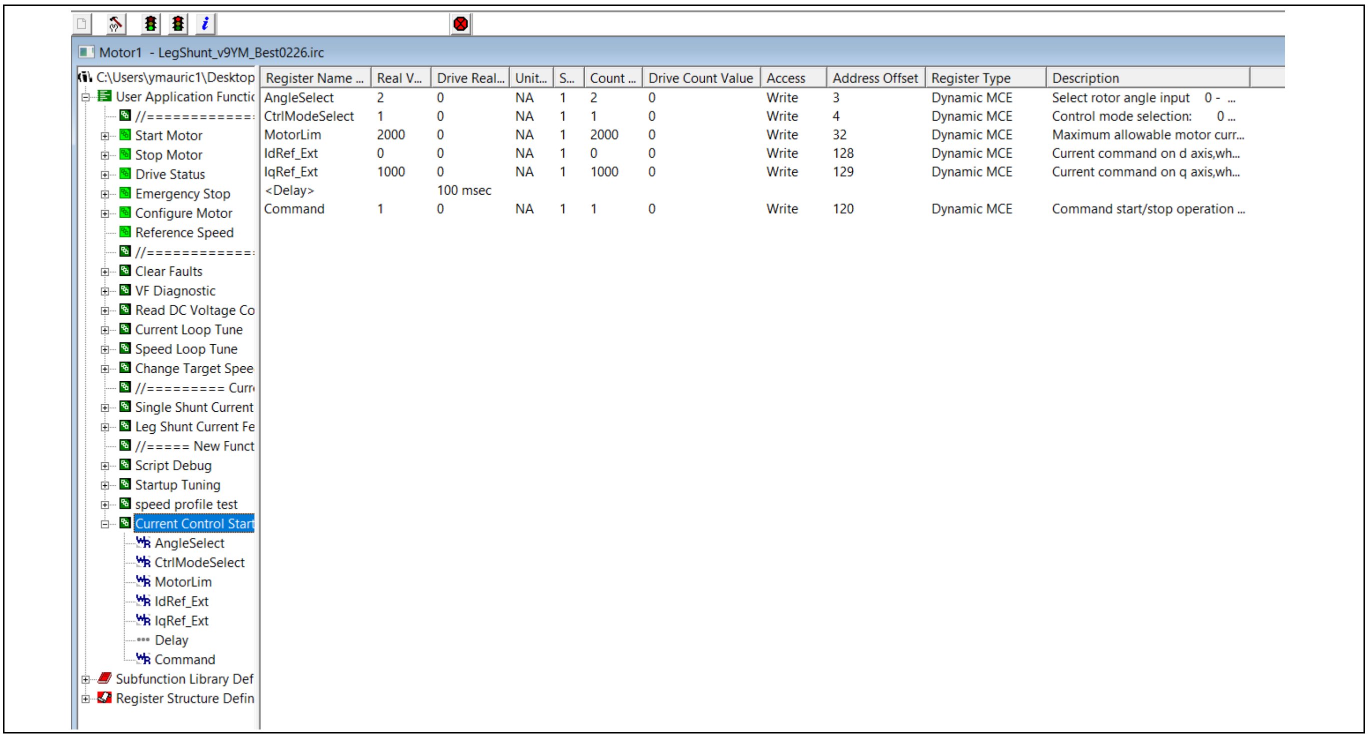


Figure 6 **Current control startup function definition in MCEDesigner**

Create a new function group, *Current Control Startup*, by right clicking on *User Application Function Definitions* and selecting *Create New Function.* Within the new function, add the following registers:

AngleSelect = 2. The rotor angle and position information are provided by the flux estimator with this setting. This setting also enables the use of additional parameters to tune the startup.

CtrlModeSelect = 1 in order to allow current control mode. Once current mode control is enabled, IdRef_Ext and IqRef_ext can be adjusted to tune the current control loops.

MotorLim is limiting the max current applied to the motor to avoid any over current at startup

IdRef_Ext = 0. The reconstructed DC current is only injected in the Q axis. The direct current or I_d has only a magnetizing effect on the reference flux, and should be kept to 0 for productive work. The main goal of field-oriented control is to maximize the torque production and minimize the magnetization in the motor winding for active and effective torque build-up. The voltage across the winding inductance due to the I_d current is in line with the back EMF, so it can directly add to or subtract from the back EMF voltage.

IqRef_Ext should be set to the current providing the desired startup torque. IqRef_Ext = 0 there is no startup current and IqRef_Ext = 4095 is the max rated RMS current entered question 2 in MCEWizard. The quadrature current or i_q is a torque-producing current, which allows the motor to rotate. The + or – sign will determine the motor startup direction. I_q current acts at 90° relative to the axis of the magnet, and produces torque. The voltage across the winding inductance due to the id current is in the quadrature with the back EMF so the motor terminal voltage is larger and out of phase with the back EMF.

Refer to the iMOTION™ Motion Control Engine Software Reference Manual for detailed descriptions of these registers.

When running the motor in current control mode, some careful steps should be taken in order to not damage the system. Make sure there is an overcurrent protection circuit outside iMOTION™. Most CIPOS™ IPMs have an overcurrent protection feature that can be used. Additionally, MotorLim and LowSpeedLim registers can be set up for protection as well. Please see the Software Reference Manual for more detailed instructions.

3.1.2 Angle sensing setting

Some fan applications require that motors start up reliably in the right direction, without a back-and-forth jitter. Using the traditional parking + open-loop method would cause undesired reverse motion in some cases. Using a direct-start method may fail due to insufficient back EMF at low motor speed range.

MCE offers a patented, initial angle-sensing function that estimates the rotor angle by injecting six current pulses at different angles for a duration of a few milliseconds before starting. The initial angle is then calculated based on the current amplitude of those sensing pulses. After ANGLE_SENSING state is completed, the motor state machine would shift to MOTOR_RUN state to run the closed loop FOC control directly.

The initial angle-sensing function can always start the motor in the right direction, and avoids potential reverse motion during parking when used in sensorless FOC control. The initial angle estimation relies on rotor magnetic saliency, and performs well when the motor L_d to L_q ratio is less than 95% and the average inductance is greater than 0.1 mH.

The relevant control parameters (IS_Pulses, IS_Duty, IS_IqInit) are automatically calculated by MCEWizard based on the L_d and L_q motor parameters entered. However, for some motors, the lower saliency and the very high back-EMF might require the user to tune the relevant angle-sensing parameters. Startup behavior can also depend on the initial rotor position. For best start up tuning, vary the initial rotor angle and evaluate the start up behavior. The "worst rotor position" lies where the stator flux and rotor magnetic flux have the largest angle, and the least torque is produced (Figure 9). The whole procedure tries to achieve a torque that is high enough to rotate the motor with current limitation, and at the same time properly estimate the rotor angle for the flux estimator to start working properly. Then the system can be controlled.

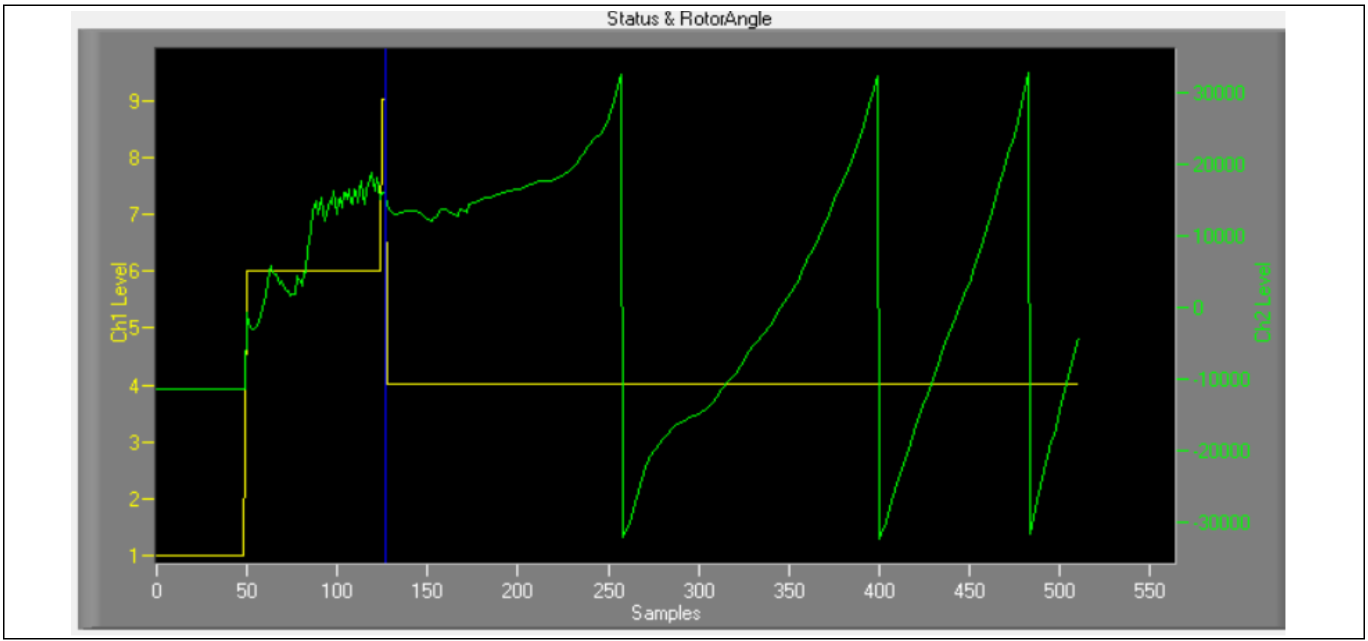


Figure 7 Aligned rotor and good startup position. Ch1 [Yellow] MCEDesigner SequencerState register to trigger on anle sensing. Ch2 [Green] RotorAngle register

Figure 7 shows a good startup situation where the rotor angle is properly estimated via the flux estimator, and the flux angle is properly aligned with the stator during the test. A torque can then be applied and the angle can be properly estimated.

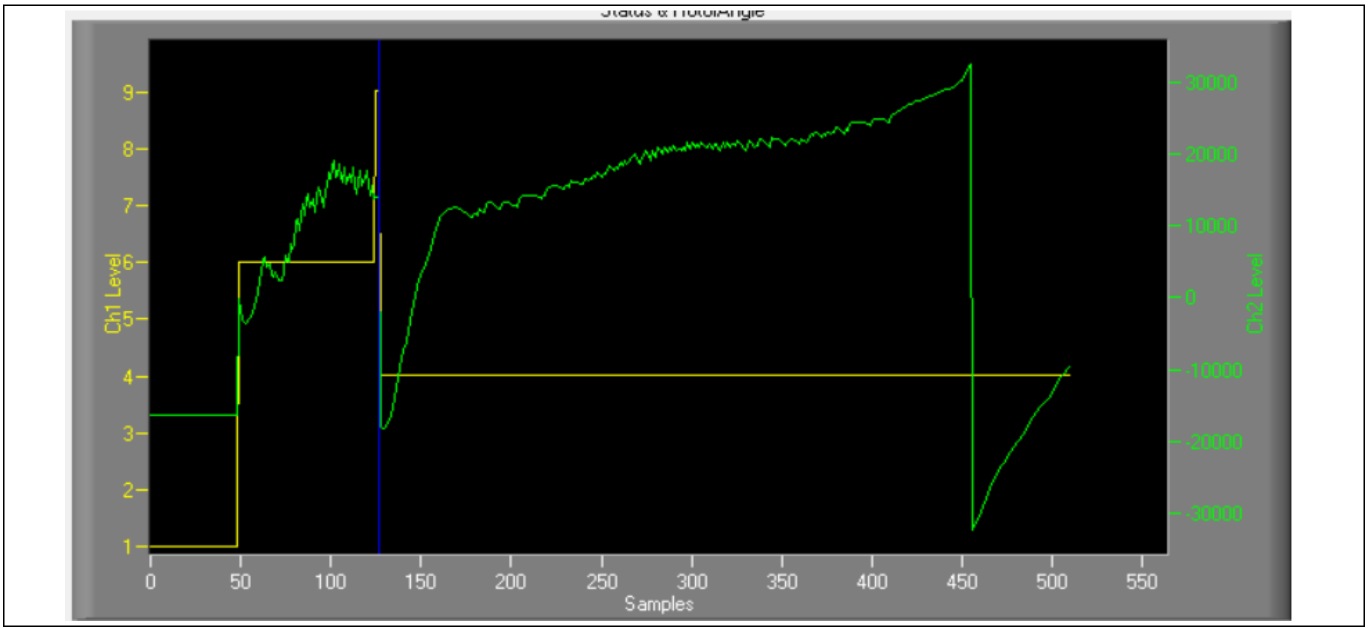


Figure 8 Rotor worst position during startup

Figure 8 shows the result starting from the worst rotor angle position. Looking at Ch2, we can see that the rotor angle cannot be properly estimated. This is because the amplitude of the current sensing pulse is too low. The motor cannot start properly if insufficient torque is applied. At this position, the stator magnetic field is in the same direction as the rotor magnetic field, and the least torque is produced.

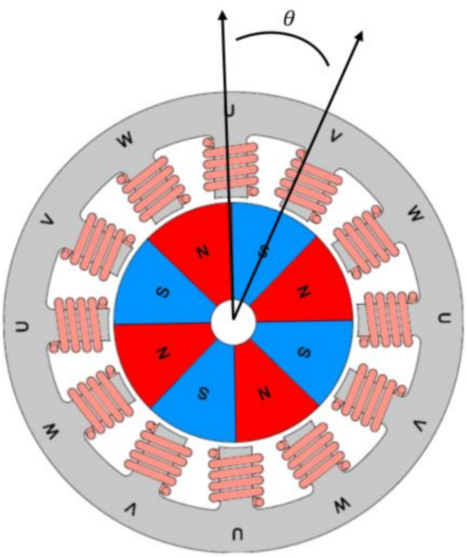


Figure 9 Angle between rotor magnetic flux and stator flux

IS_Duty: This parameter specifies the PWM duty cycle during the ANGLE_SENSING state. For better current sensing quality, in single shunt current sensing, the duty cycle of angle sensing should not be too low; otherwise the active vector is too short to sense the current. In leg shunt current sensing, the duty cycle should not be too high; otherwise there is not enough time to sense the current during zero vector.

IS_IqInit: This parameter specifies the initial torque that should be applied after having completed the ANGLE_SENSING state and before entering MOTOR_RUN state. Immediately after having completed the

ANGLE_SENSING state, the flux PLL has not locked onto the rotor angle. It takes some time to stabilize itself, and requires the motor speed to be sufficiently high. This means at the beginning of the MOTOR_RUN state, the flux PLL is not working properly, and it relies on initial torque to accelerate the motor in order for the PLL to lock. To achieve a reliable and smooth start, some tuning of the flux estimator and flux PLL is required.

Total equilibrium occurs when the correct amount of torque is applied. If there is too much, the rotor will try to rotate faster than it actually can due to inertia, and the angle estimation will be lost. If there is not enough, the angle cannot be estimated, and the rotor will stall.

During the startup test procedure, try to reduce the IS_IqInit up to the value that starts an initial rotation.

This method only deals with the initial angle measurement, so the flux estimator might also have to be tuned when the load is rotating. If the motor speed is not zero at the startup, the detected angle might not be accurate. It is recommended to use the catch-spin function in that scenario.

For a high inertia load startup, it is very important to delay the operation of the flux estimator in order to have the motor reach a speed where the back EMF can provide a useful signal used in the angle estimation for the flux estimator. The main register for reduction is called the flux estimator time constant, which can be found in the MCEWizard. The default value is set up at 15 ms. For a correct startup with high inertia loads, it is not unusual to see an increase of 5 times this value for obtaining a better low-frequency signal reading.

In our system example, we set the values as follows:

Flux estimator time constant = 50 ms

IS_Duty = 1000 counts

IS_IqInit = 200 counts

IqRef_Ext = -500 counts for counter clock wise rotation.

3.2 Setting the speed loop and running the motor

Once the startup procedure and angle estimation have been properly set, the motor is able to start and we should have the load rotating. Now it is time to tune the speed loop. Note that the following procedure is system dependent; we are using the EVAL-IMM101T starter kit as an example system. The same methodology can be applied to any motor control system.

The speed loop PI controller must first be properly determined. To do that, we need to do a speed response to a current step. We will use the IqRef_Ext register similar to Current Control Start function detailed in 3.1.1 to increase a step response and observe how the speed will settle.

In order to observe the following speed profile to current step, a monitor need to be set up in MCEDesigner. In the system window, right click on Monitor Definitions and select “Create New Monitor Group” that you can call IqRef_Ext and MotorSpeed. Then right click on the newly created monitor and select “Define Trace”. For CH1 Source (yellow) select IqRef_Ext register and for CH2 Source (green) select MotorSpeed register:

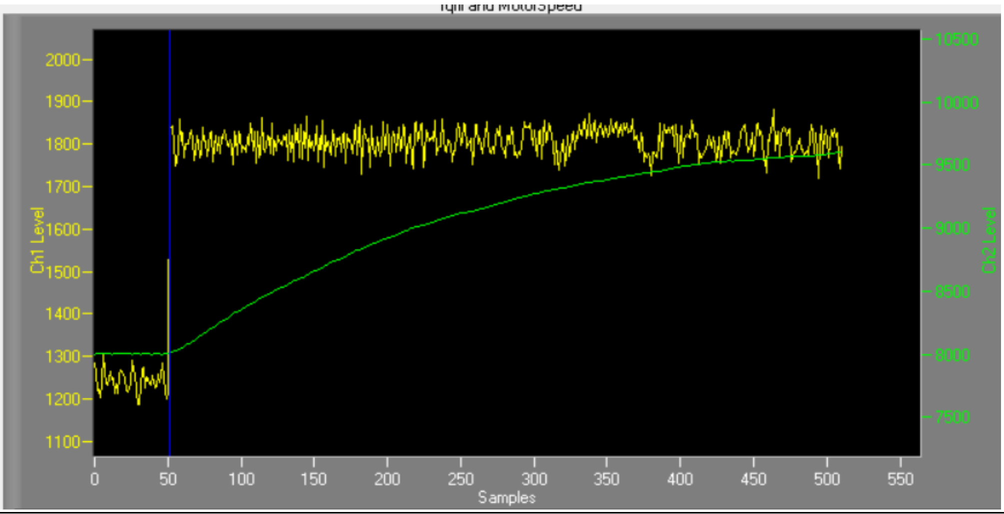


Figure 10 IqRef_Ext and motorspeed register during current step response test

We have observed that the startup is quite stable during current control mode.

Now let’s switch to including the speed control mode and tuning it.

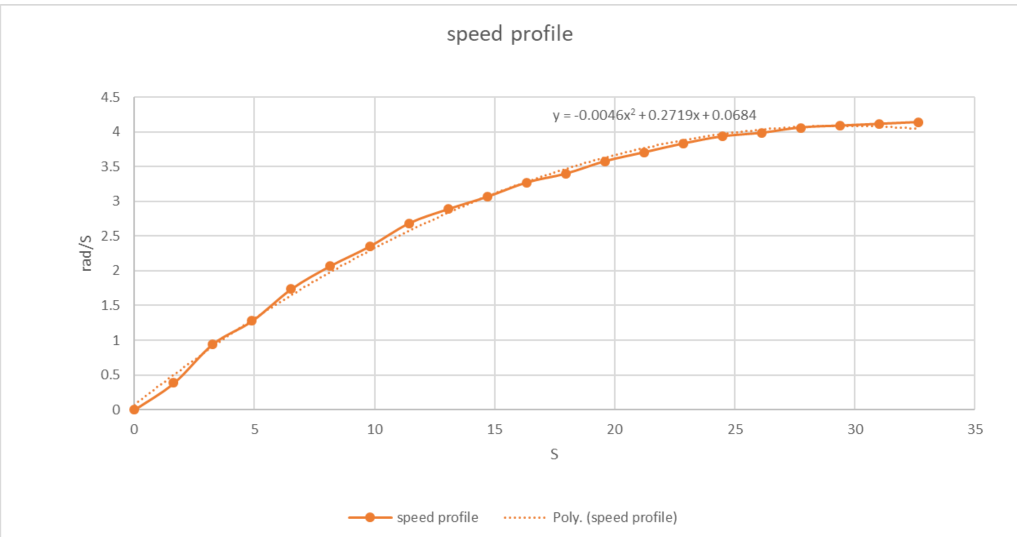


Figure 11 High Inertia fan speed response in SI

We then need to know the load profile of the fan to see if we can linearize the system. Fans have a polynomial load profile – motor power increases non-linearly with speed. In order to obtain the following measurements, we have to drive the fan at different speeds-changing TargetSpeed registers in MCEDesigner knowing that the maximum motor RPM values entered in the MCEWizard account for 16 383 counts. We then measure the output power applied to the fan at the corresponding speed.

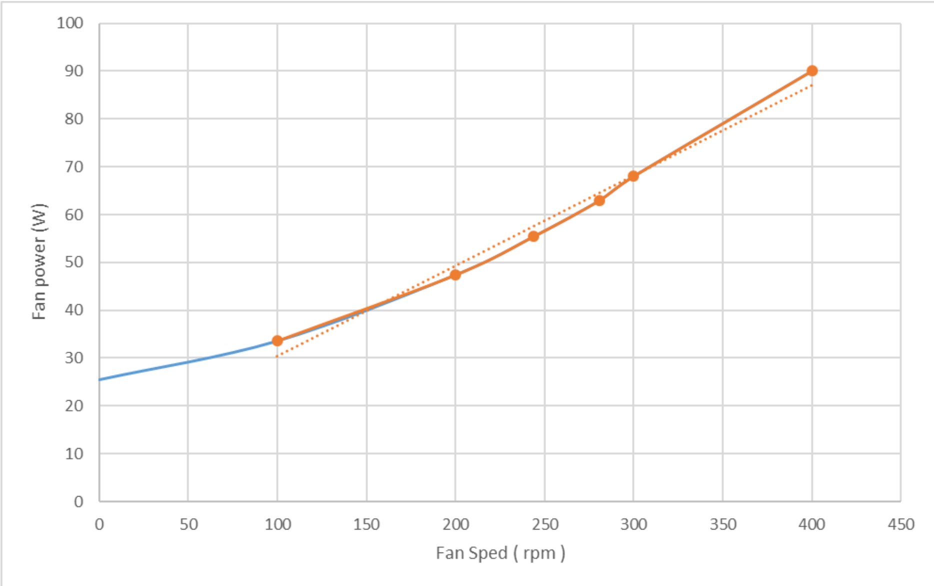


Figure 12 Fan speed vs Output power

To simplify the rest of the process, we have assumed a linear load profile for speed > 100 RPM. For speeds lower than 100 RPM, the fan will start up in current control mode, as we demonstrated in Section 3.1. This ensures proper current and torque control during the startup procedure. We will develop a script to start the system in current control mode and then switch to speed control mode.

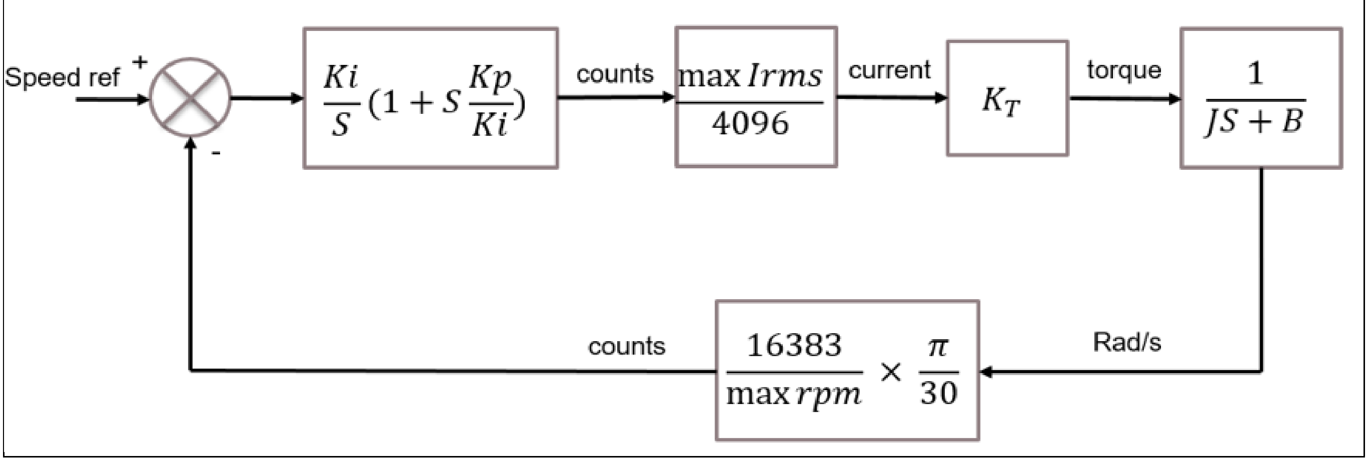


Figure 13 Linearized model of the system with iMOTION™ controller for speed control

The Kp and Ki can now be calculated for this system.

The fan is described in Laplace:

$$\frac{1}{JS + B}$$

With:

$$J = \frac{Kt\ (\Delta I)}{\frac{\Delta w}{\Delta t}}$$

$$B = \frac{J}{\tau}$$
$$Kt = 3\ \times Ke$$

We also need to know some values for our system in order to use SI during the determination process:
Switching frequency = 15Khz
Motor fast control rate = 4 this give us 3.75 Hz = 0.266 S
Our sampling time equal to 3.75/245 = 0.0153 Hz = 6.53ms
ΔI is the step current that define the speed response in Figure 11

All values can be extracted for the inertia and the friction coefficient from the speed test response.
Following the speed profile example in Figure 11:

$$Kt = 0.0427\ V\ / \frac{rad}{S}$$
$$\Delta I\ = 0.084\ A$$
$$\Delta w\ = 2.34\ rad/S$$
$$\Delta t\ = 10\ S$$
$$\tau = 11.2\ S$$

Hence, J = 0.015321 $kg.m^2$ and B = 0.01369 N
The integral regulator parameter should now have the same speed as the controller bandwidth:

$$Integral\ Gain = Ki = \frac{Band\ width}{\frac{Irms}{4096} \times \frac{16383}{maxspeed} \times \frac{Kt}{B}}$$

$$proportional\ gain = Kp = \ \tau Ki$$

We determine has arbitrary number that we want our speed response to be 10 time faster than our current sampling time. This give us :

$$Ki = \frac{0.0653}{\frac{0.625}{4096} \times \frac{16383}{41.87} \times \frac{0.0427}{0.01369}} = 0.03503$$

We then find Ki = 0.03503 and Kp = 0.3924. Converted into counts for iMOTION™, Kx = 1 and Kp = 100

Figure 14 MCEWizard settings for speed regulator

A step response speed check is done at this time to see if the speed loop is set properly. Proper behavior should include a speed regulation time based on the speed ramp rate, and a stable speed operation at steady state and at the desired speed command. For high inertia loads, it is important to slow down drastically the system to take into account the consequent delay that the load can generate. The speed ramp rate should be set to a small value adequate for the system. In our example, we have:

- Maximum motor RPM = 400 RPM
- Minimum running speed = 100 RPM (hence the linearization assumption demonstrated before)
- Speed ramp rate = 4 RPMs. it is very important to have a balance between the current injection, the angle estimation and the acceleration of the system at start up to beat the inertia. The max value can be found through iteration and testing but keep in mind that the speed feedback time constant is quite delayed due to the high inertia load. If the acceleration rate at start up is too high, the system will lose track of the angle estimation as the virtual reconstructed angle will rotate faster than the actual rotor.

Compared to floor standing fans for example, high inertia loads need the maximum torque at low speed, but if too much torque is applied, the startup angle sensing will lose control, and the motor cannot start. Again it is a question of finding the right balance. MCE provides a feature for configuring the dynamic motor current limit, which can be used to set the correct current limit in the low speed region, and prevent electronic damage.

In the motoring mode, when the absolute value of the motor speed is below the minimum speed specified by the parameter 'MinSpd' ($|MotorSpeed| \leq MinSpd$), the maximum motor current is limited to a threshold configured by the parameter 'LowSpeedLim.' When the absolute value of the motor speed is between the minimum speed and the low speed threshold, the motor current limit increases linearly as the speed increases, as per the following equation:

$$Motor\ Current\ Limit = LowSpeedLim + (|MotorSpeed| - MinSpd) \times LowSpeedGain$$

When the motor speed goes beyond the low speed threshold, the maximum motor current is limited to the upper boundary specified by the parameter 'MotorLim.'

In our example, the low speed current limit is set at 60%. Having the freedom to adjust the motor current limit in motoring mode allows users to tailor the acceleration torque to achieve optimal drive performance. If further customization of the motor current limit is required, users can take advantage of script code to program the motor current limit ('MotorLim' parameter) to any arbitrary profile.

Now that the speed regulator is set up, the last parameter to look at is the speed feedback filter time constant in the iMOTION™ wizard. As we are working with high inertia loads, the system response delay is significant. It is

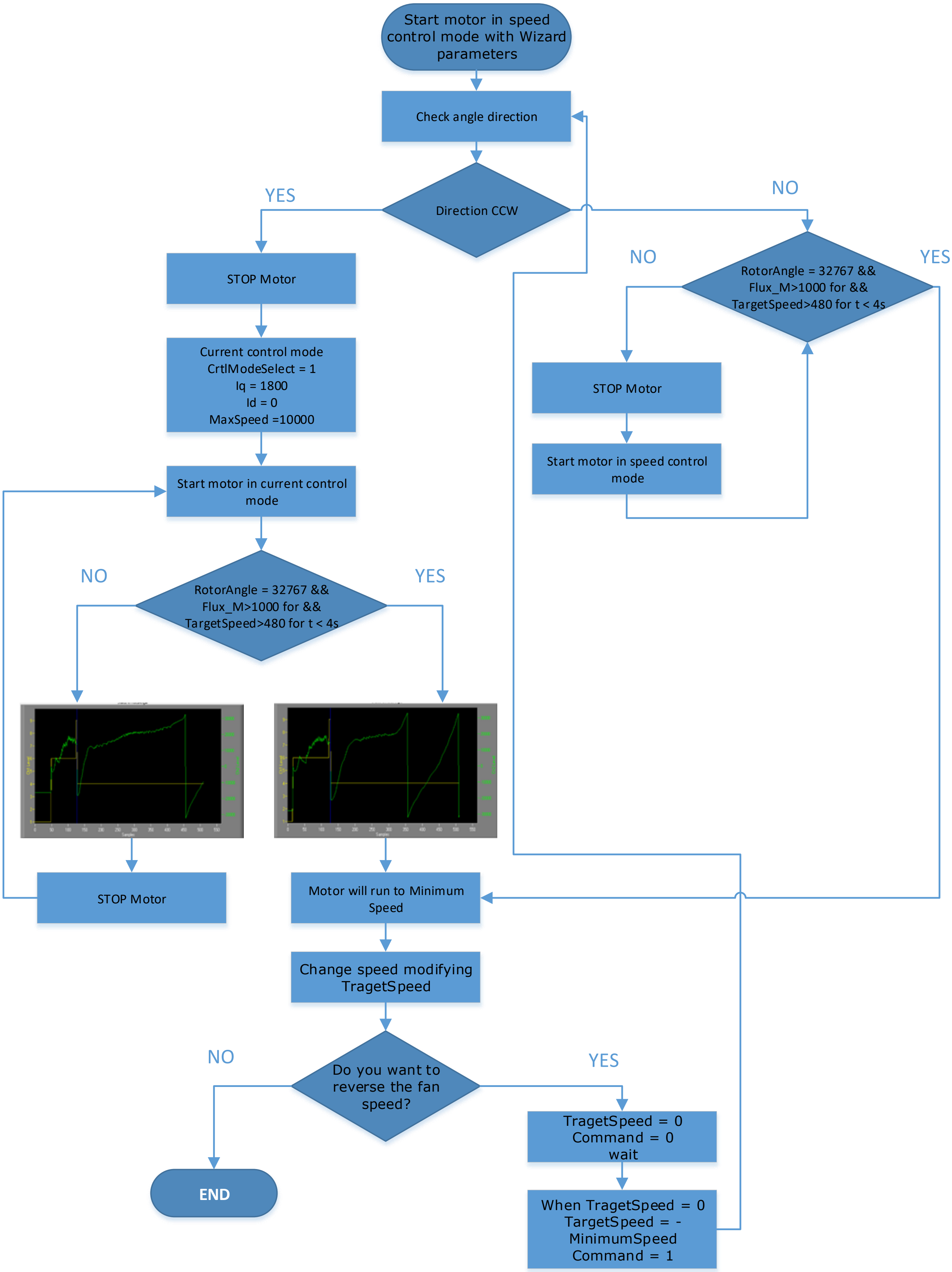
therefore important to delay the operation for the speed regulator, especially at startup, so that the flux estimator can receive the correct signals and process them in time. In our example we have the speed feedback filter time constant = 100 ms. As with the flux estimator time constant seen in Section 3.1.2, the slowdown of these values is extremely important at start up to prevent the flux estimator from losing control and working only with noise signals.

The motor should now be able to run with the proper startup and speed control. If there is an issue with startup, it is advisable to use a scripting function below the linearization of the speed controller, where current control startup is used. This can be done with the CrtlModeSelect register.

3.3 Flow chart script example

In keeping with the example used in this paper, the following script flow chart can be used to control high inertia loads, reverse speed, and do a proper startup either in speed loop or current loop control mode. Even if the rotor is in the worst angle start up position and the system is not fully set up, when trying to start the motor it will still slightly rotate and get out of the position. This physical phenomenon avoid to be locked in an infinite loop.

Please keep in mind that this can be done only after the preliminary work described in this paper has been done, such as setting the current control with startup current limit, setting angle sensing, and setting the speed loop.


Start motor in speed control mode with Wizard parameters
Check angle direction
Direction CCW
YES
NO
STOP Motor
Current control mode
CrtlModeSelect = 1
Iq = 1800
Id = 0
MaxSpeed =10000
Start motor in current control mode
NO
YES
RotorAngle = 32767 &&
Flux_M>1000 for &&
TargetSpeed>480 for t < 4s
NO
YES
RotorAngle = 32767 &&
Flux_M>1000 for &&
TargetSpeed>480 for t < 4s
STOP Motor
Start motor in speed control mode
STOP Motor
Motor will run to Minimum Speed
Change speed modifying TragetSpeed
NO
YES
Do you want to reverse the fan speed?
END
TragetSpeed = 0
Command = 0
wait
When TragetSpeed = 0
TargetSpeed = -MinimumSpeed
Command = 1

References

[1] iMOTION™ Motion Control Engine, Software reference manual

[2] EVAL-IMM101T-046 Reference manual

[3] PID Control Theory. Richard M. Murray - caltech

Revision history

Document version	Date of release	Description of changes
Ver 1.0	August. 2020	Released version

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2020-08-21

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference
AppNote Number