

# EVAL\_FAN\_XMC\_PFD7

Information and user guide

[www.infineon.com/eval-fan-xmc-pfd7](http://www.infineon.com/eval-fan-xmc-pfd7)



# Agenda

---

1 Introduction to fan driver evaluation board

2 Hardware and software

3 Tools

4 Getting started

5 Resources

# Agenda

---

1 Introduction to fan driver evaluation board

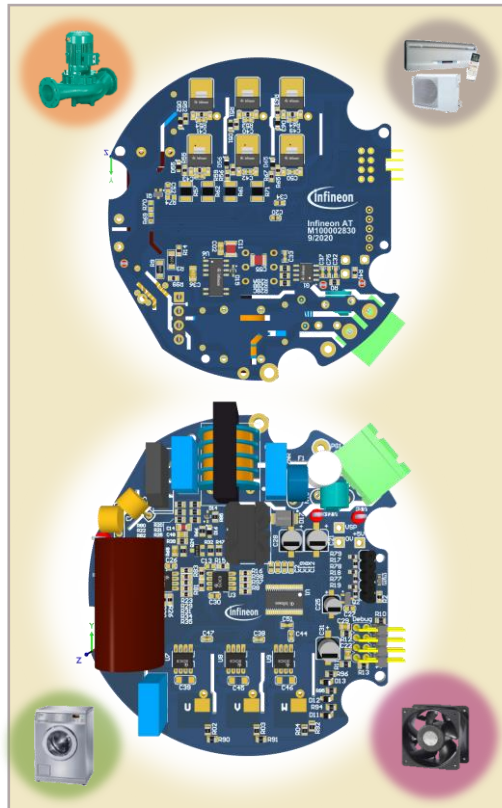
2 Hardware and software

3 Tools

4 Getting started

5 Resources

# Introduction



**Compact 3-phase motor drive** system solution up to 100 W

Designed for **sensor less or sensored** motor control

**Spin your motor** with easy-to-use GUI for tuning of motor

The hardware board and motor control software provides:

- Sensorless speed controlled direct FOC startup
- 7-segment / 5-segment SVM
- Over-current protection by CTRAP
- Speed controlled using  $\mu$ C/Probe™ GUI's slider

# Agenda

---

1 Introduction to fan driver evaluation board

2 Hardware and software

3 Tools

4 Getting started

5 Resources

# System overview

## Components

- › **XMC1302 Microcontroller**
  - Based on XMC1302 ARM® Cortex®-M0 MCU enabling sensored and sensorless motor control
  - Supports HALL
  - Includes J-Link debug interface by Segger
  - µC/Probe™-based GUI for parametrization and tuning
- › **Half bridge gate driver EiceDRIVER™ 2ED28073J06F**
- › **CoolIMOS™ IPN60R2K0PFD7S**
- › **CoolSet™ ICE5QR4770AG**
- › **Software package** (downloadable from [www.infineon.com/eva](http://www.infineon.com/eva))
- › Micrium µC/Probe™ for XMC™
  - .zip file with DAVE™ 4 project for the EVAL\_FAN\_XMC\_PFD7
  - kit (e.g. Eval-100W\_FOC\_XMC13\_v1.zip)

# System overview

For example the EVAL\_FAN\_XMC\_PFD7 Kit consists of:

## Hardware

- › EVAL\_FAN\_XMC\_PFD7 board
- › XMC™ Link
- › Micro-B USB cable

## Software

- › Micrium  $\mu$ C/Probe™ for XMC™
- › Eval-100W\_FOC\_XMC13.zip file  
(downloadable from <https://www.infineon.com/cms/en/p>)

# System overview

## Software package content description:

### › **Eval-100W\_FOC\_XMC13**

A DAVE™ 4 project for XMC1302-TO380200 device. The user can configure the motor parameter and startup mode in the source code. The compile firmware needs to be programed to EVAL\_FAN\_XMC\_PFD7 board via DAVE™ 4 IDE and XMC Link

### › **Eval-100W\_FOC\_XMC13.elf**

A debugging file used by µC/Probe™ GUI tool

### › **Eval-100W\_FOC\_XMC13.wspx**

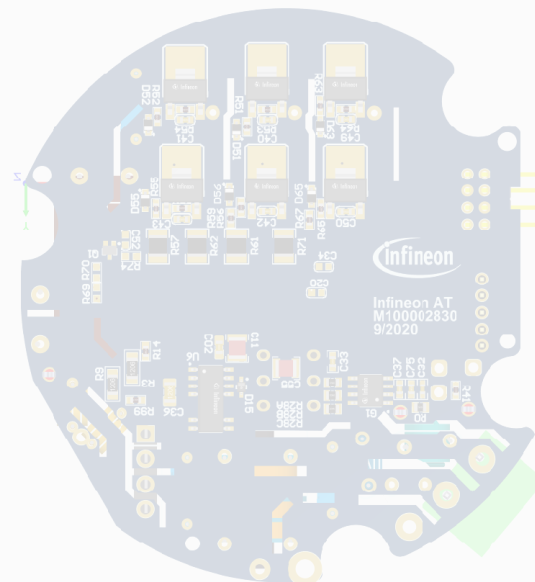
A µC/Probe™ project file for predefined GUI used for system parametrization and tuning



# Hardware overview

## Hardware

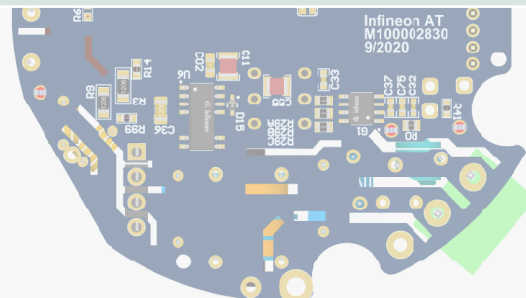
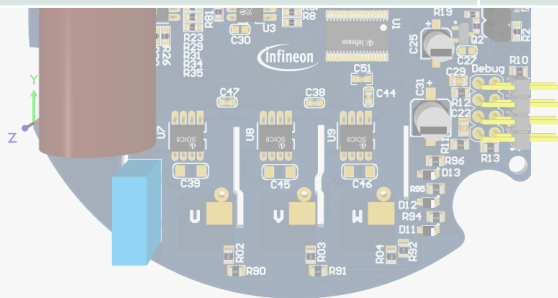
- › **Kit name**
  - EVAL\_FAN\_XMC\_PFD7
- › **Kit description**
  - EVAL\_FAN\_XMC\_PFD7 board
  - USB cable
  - XMC™ Link – Isolated Debug Probe
- › **Order number**
  - add
- › **Input voltage/output power**
  - 230 VAC/100 Watt max



# Hardware overview

Infineon parts utilized on EVAL\_FAN\_XMC\_PFD7 board:

Infineon Parts	Order Number
XMC1300 Microcontroller	XMC1302-T038F0200
600 V half bridge gate driver	2ED28073J06F
600 V CoolMOS™	IPN60R2K0PFD7S
Quasi-resonant CoolSET™	ICE5QR4770AG
5 V LDO regulator	IFX1763XEJV50



# Hardware overview

To properly setup the board, follow these steps:

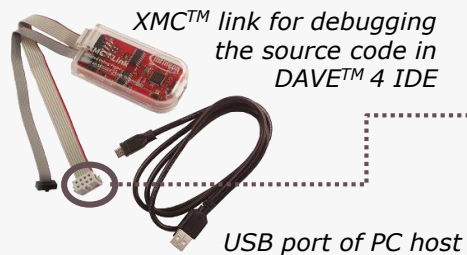
1  
Connect 3-phase motor wiring to 'U V W' connector

2a  
Connect XMCT<sup>™</sup> Link's ribbon cable to pin 1 of Debug connector

3  
Connect AC source to EVAL\_FAN\_XMC\_PFD7 board

AC power input 230 VAC

2b  
Connect to PC via micro-B USB cable



# Agenda

---

1 Introduction to fan driver evaluation board

2 Hardware and software

3 **Tools**

4 Getting started

5 Resources

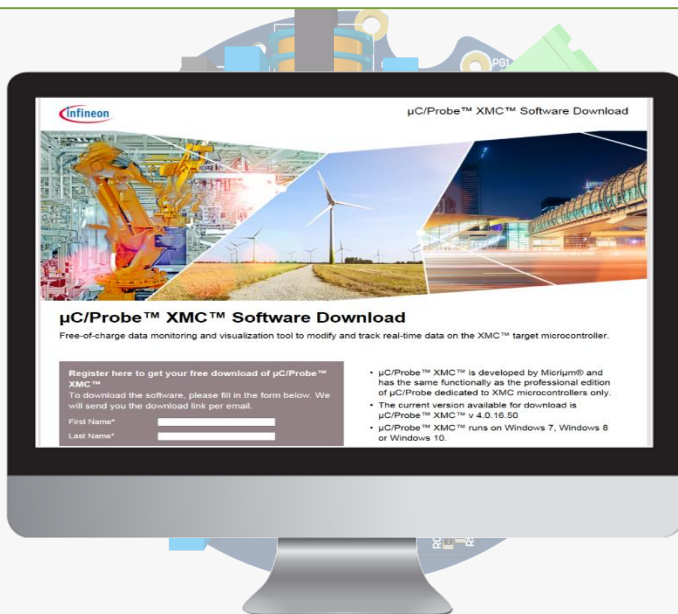
# DAVE™ 4 setup



# Micrium $\mu$ C/Probe™

**Download Micrium  $\mu$ C/Probe™ for XMC™ installer package from:**

[www.infineon.com/ucprobexmc](http://www.infineon.com/ucprobexmc)



## Installation requirements:

- > PC with Windows Vista, Windows 7, Windows 8, Windows 10 – 32 bit & 64 bit
- > RAM – 3 GB or more

# Agenda

---

1 Introduction to fan driver evaluation board

2 Hardware and software

3 Tools

4 **Getting started**

5 Resources

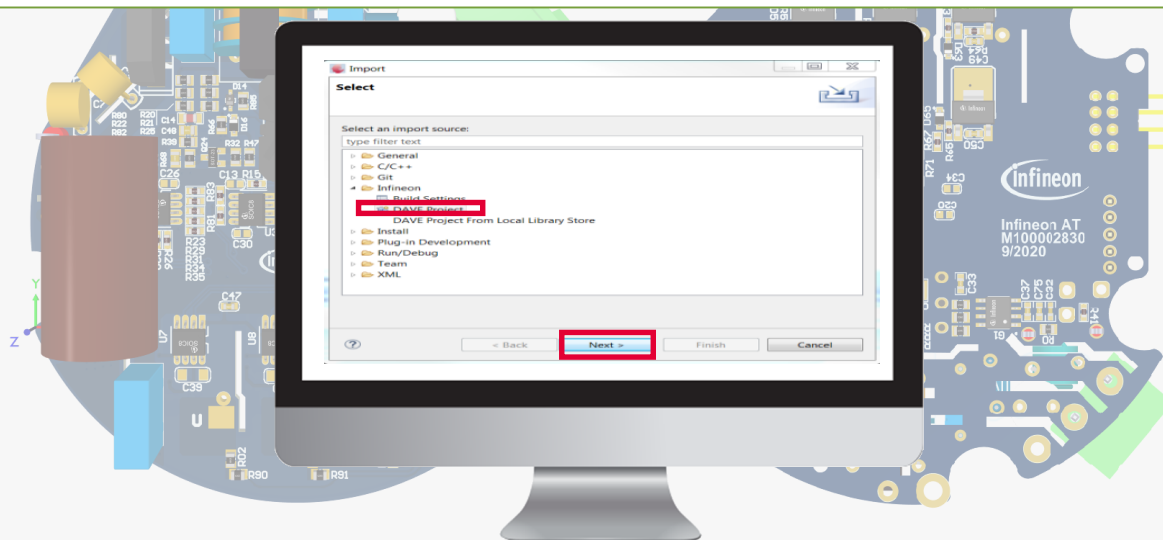
# Import the \*.zip project

1

Import the \*.zip file into DAVE™ 4 IDE by clicking on **File > Import**

2

Select **Infineon > DAVE Project** and click the **Next** button





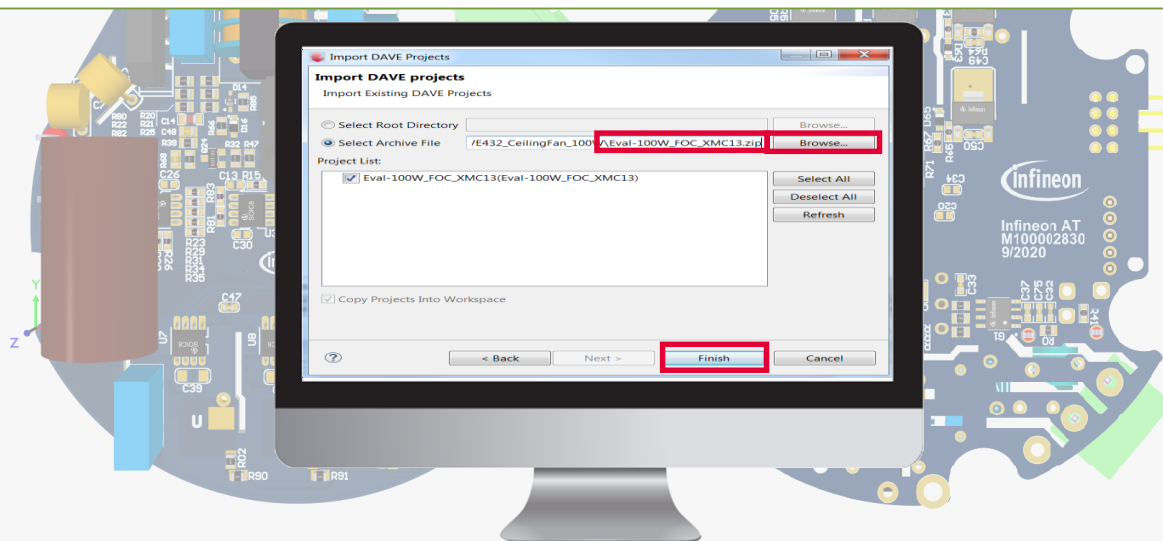
# Import the \*.zip project

3

Click **Browse** and navigate to the \*.zip file location

4

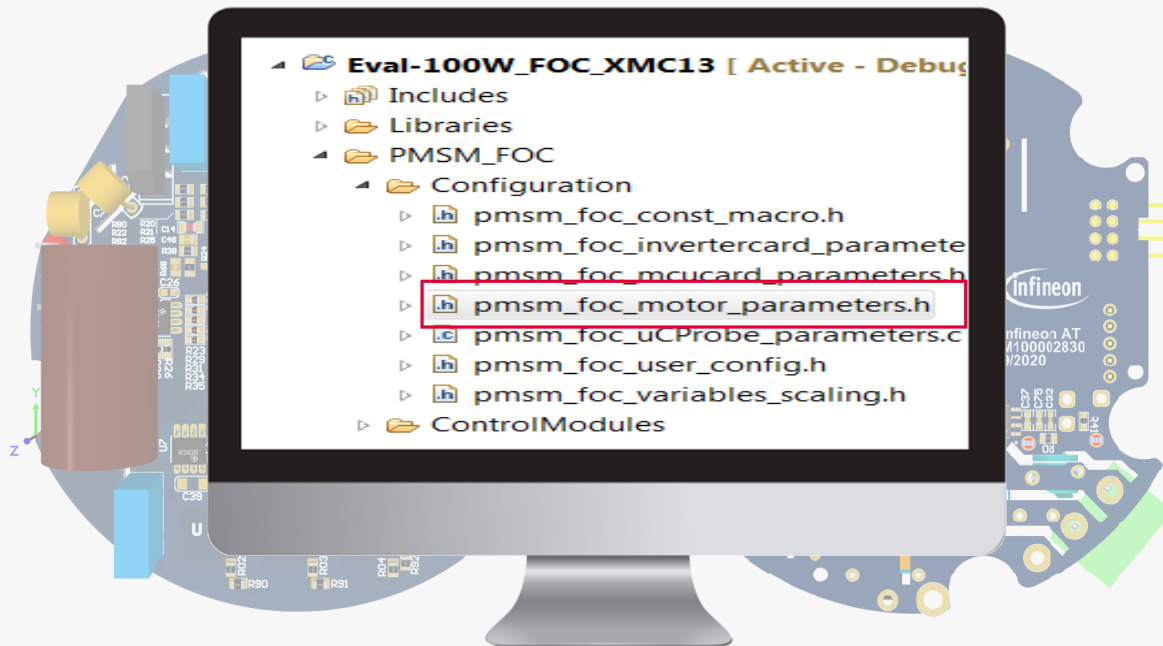
Click the **Finish** button



# Input motor parameters

1

Double click on **pmsm\_foc\_motor\_parameters.h**



# Input motor parameters

2

Scroll to **(MOTOR\_TYPE == CUSTOM\_MOTOR)** and update the parameters in the red box with your motor's data

```

204 /* ***** CUSTOM_MOTOR ***** */
205 #if(MOTOR_TYPE == CUSTOM_MOTOR)
206 /* ----- Motor Parameters ----- */
207 #define USER_MOTOR_R_PER_PHASE_OHM (23.0f) /* Motor Resistance per phase in Ohm */
208 #define USER_MOTOR_L_PER_PHASE_uH (55000.0f) /* Motor Inductance per phase in uH */
209 #define USER_MOTOR_POLE_PAIR (4.0f) /* Motor Pole Pairs */
210 /* ----- Constant Speed Control Mode (Used when Constant Speed Control is enabled) ----- */
211 /* POT ADC, or PWM to Adjust Speed */
212 #define USER_SPEED_HIGH_LIMIT_RPM (1050) /* Max Speed of User Motor */
213 #define USER_SPEED_LOW_LIMIT_RPM (wint32_t) (USER_SPEED_HIGH_LIMIT_RPM / 300)
214 #define USER_SPEED_RAMPUP_RPM_PER_S (300)
215 #define USER_SPEED_RAMPDOWN_RPM_PER_S (300)
216 #define USER_RATIO_S (10) /* Minimum ramp up and down ratio */
217 /* ----- V/F Start-Up Parameters ----- */
218 #define USER_STARTUP_SPEED_RPM (80)
219 #define USER_STARTUP_SPEED_THRESHOLD_RPM (150) /* threshold Speed to transit from Open Loop to closed Loop */
220 #define USER_STARTUP_VF_OFFSET_V (float) (USER_VDC_LINK_V * 0.05f-6) /* V/F offset */
221 #define USER_STARTUP_VF_SLEWRATE_V_PER_HZ (float) (USER_VDC_LINK_V / (ELECTRICAL_SPEED_FREQ_HZ * 2))
222 // #define USER_STARTUP_VF_OFFSET_V (0.4f) /* V/F startup offset in V */
223 // #define USER_STARTUP_VF_SLEWRATE_V_PER_HZ (0.07f) /* V/F start up slew rate in V/Hz */
224

```

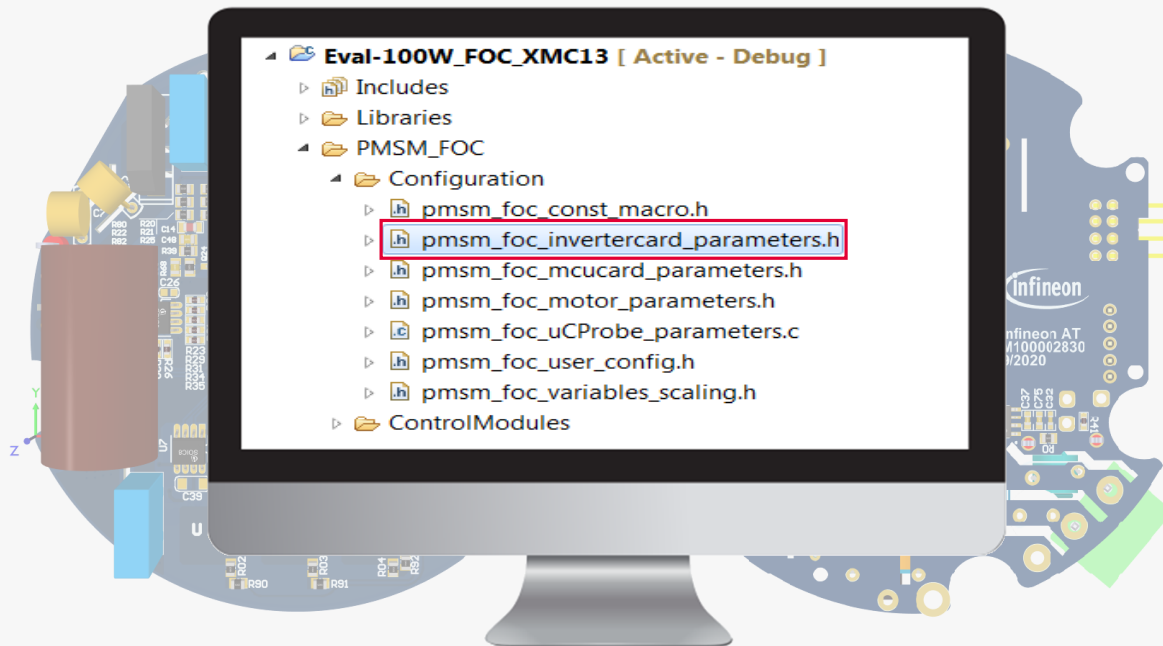
**i**

For more explanation of the parameter specified in the code, please refer to XMC1000 application note: 'AP32370-XMC1000-PMSM FOC motor control software using XMC™'

# Input inverter circuitry parameters

1

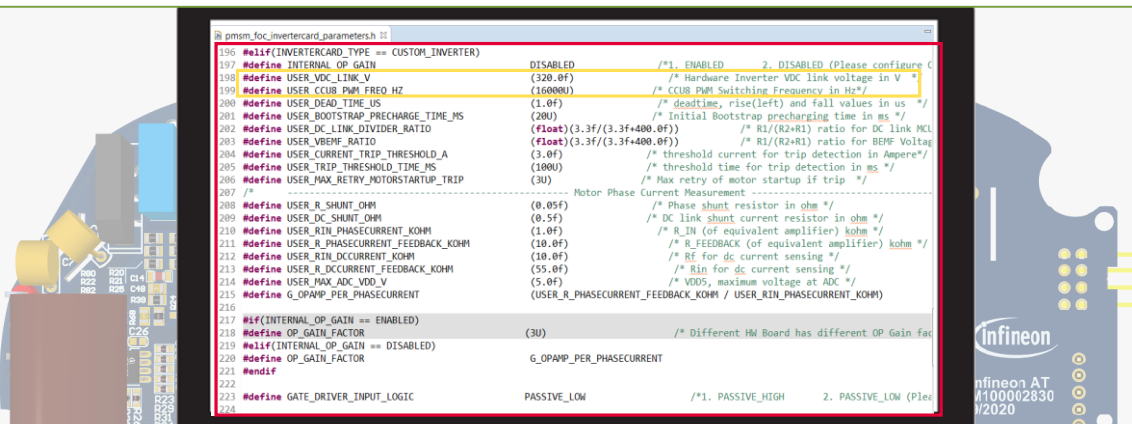
Double click on **pmsm\_foc\_invertercard\_parameters.h**



# Input inverter circuitry parameters

2

Scroll to **(INVERTERCARD\_TYPE == CUSTOM\_INVERTER)**



The inverter parameters is already preset to the EVAL\_FAN\_XMC\_PFD7 board's value, e.g. USER\_VDC\_LINK\_V is 320 V ( $\sqrt{2} \times 230$  V) with the assumption that AC input voltage is 230 V

The PWM frequency driving the power MOSFET is set to 16 kHz

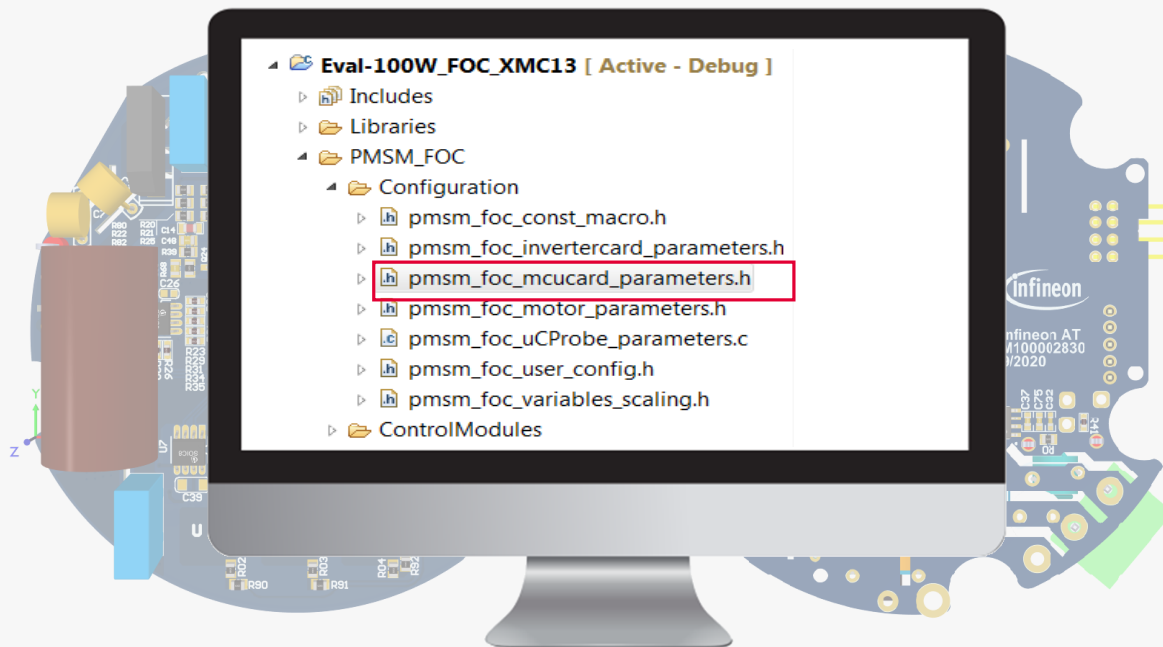
The minimum PWM frequency can be set is 6.6 kHz

The user does NOT need to change the value here if AC input voltage to EVAL\_FAN\_XMC\_PFD7 board is 230 V

# Input XMC1302 pin setting

1

Double click on **pmsm\_foc\_mcucard\_parameters.h**



# Input XMC1302 pin setting

2

Scroll to (MCUCARD\_TYPE == CUSTOM\_MCU)

```

770 #elif(MCUCARD_TYPE == CUSTOM_MCU)
771 /*****
772  * CUSTOM_MCU
773  * GPIO Resources Configuration
774 *****/
775 #define TRAP_PIN P0_12
776 #define INVERTER_EN_PIN P0_11
777 #define USE_INVERTER_EN_PIN 0
778
779 #define PHASE_U_HS_PIN P0_0
780 #define PHASE_U_HS_ALT_SELECT XMC_GPIO_MODE_OUTPUT_PUSH_PULL_ALT5
781
782 #define PHASE_U_LS_PIN P0_1
783 #define PHASE_U_LS_ALT_SELECT XMC_GPIO_MODE_OUTPUT_PUSH_PULL_ALT5
784
785 #define PHASE_V_HS_PIN P0_2
786 #define PHASE_V_HS_ALT_SELECT XMC_GPIO_MODE_OUTPUT_PUSH_PULL_ALT7
787
788 #define PHASE_V_LS_PIN P0_3
789 #define PHASE_V_LS_ALT_SELECT XMC_GPIO_MODE_OUTPUT_PUSH_PULL_ALT7
790
791 #define PHASE_W_HS_PIN P0_8
792 #define PHASE_W_HS_ALT_SELECT XMC_GPIO_MODE_OUTPUT_PUSH_PULL_ALT5
793
794 #define PHASE_W_LS_PIN P0_9
795 #define PHASE_W_LS_ALT_SELECT XMC_GPIO_MODE_OUTPUT_PUSH_PULL_ALT5
796
797 #define TEST_PIN P0_4
    
```



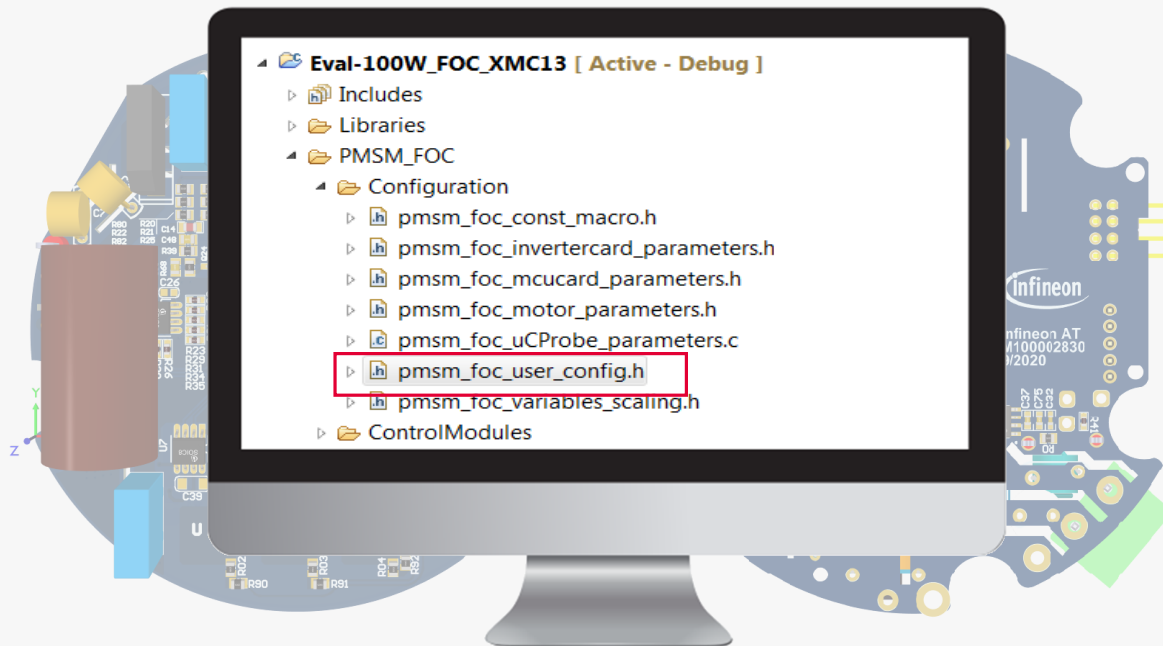
*INVERTER\_EN\_PIN is not available on this board. Hence, the USE\_INVERTER\_EN\_PIN is 0*

*The MCU pinout setting is already set according to the EVAL\_FAN\_XMC\_PFD7 board.  
The user does NOT need to change it!*

# Input motor startup: current sensing setting

1

Double click on **pmsm\_foc\_user\_config.h**





# Input motor startup: current sensing setting

2

Scroll to (MCUCARD\_TYPE == CUSTOM\_MCU)



```

78
79=====
80 * MACROS
81 =====
82 #define PMSM_FOC_HARDWARE_BOARD      CUSTOM_KIT          /*1. KIT_XMC1X_AK_MOTOR_001
83                                     2. KIT_XMC750MATT_MC_AK_V1
84                                     3. CUSTOM_KIT*/
85
86 /*----- Current feedback Sensing Mechanism -----*/
87 #define CURRENT_SENSING              USER_THREE_SHUNT_SYNC_CONV /*1. USER_SINGLE_SHUNT_CONV
88                                     2. USER_THREE_SHUNT_ASYNC_CONV
89                                     3. USER_THREE_SHUNT_SYNC_CONV*/
90
91 /*----- FOC Control and Startup Scheme (Only Select 1 Scheme at one time) -----*/
92 #define MY_FOC_CONTROL_SCHEME        SPEED_CONTROLLED_DIRECT_FOC /*1. SPEED_CONTROLLED_VF_ONLY,
93                                     2. SPEED_CONTROLLED_VF_MET_FOC
94                                     3. SPEED_CONTROLLED_DIRECT_FOC
95                                     4. TORQUE_CONTROLLED_DIRECT_FOC
96                                     5. VQ_CONTROLLED_DIRECT_FOC */
97
98 /*----- Micrium uC Probe Enable/Disable -----*/
99 #define UC_PROBE_GUI                  ENABLED              /*1. ENABLED 2. DISABLED
100
101 /*----- Reference Speed Adjustment Method -----*/
102 #define SETTING_TARGET_SPEED         MICRIUM_UC_ONLY       /*1. MICRIUM_UC_ONLY
103                                     2. BY_POT_ONLY */
  
```

The user can select 3 types of current sensing methods (as shown in green comment lines)

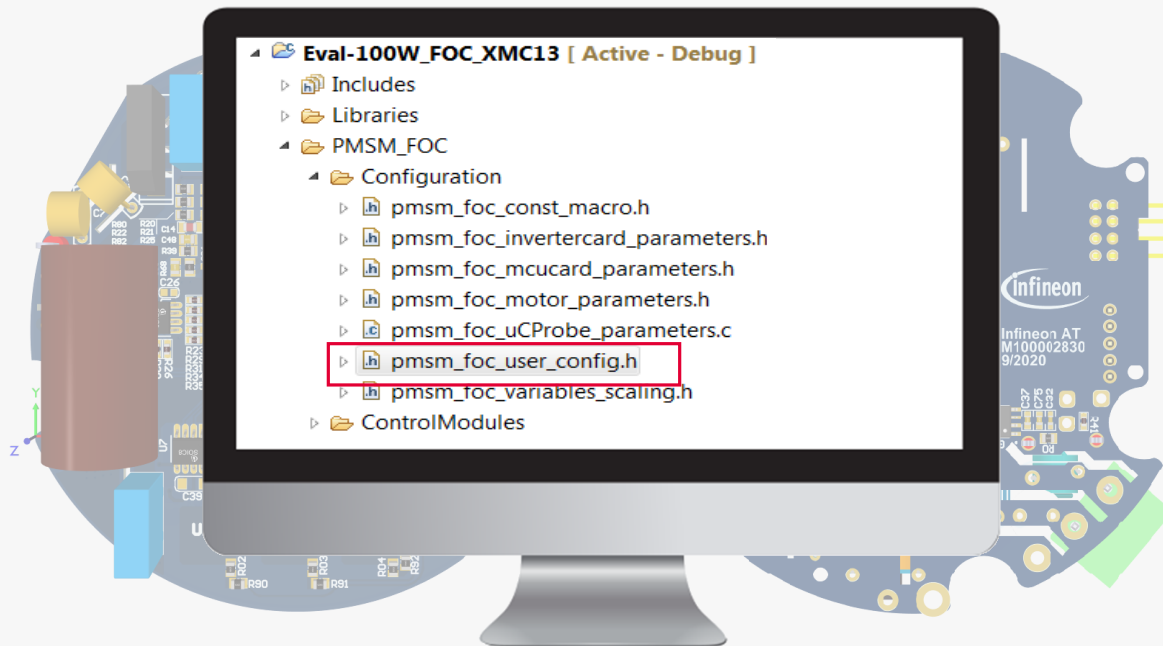
The user can select 5 types of motor control schemes (as shown in green comment lines)

3 shunt synchronous current sensing and speed controlled direct FOC startups have been selected by default. The user does NOT need to change these!

# Input SVM scheme

1

Double click on **pmsm\_foc\_user\_config.h**



# Input SVM scheme

2

The user can select 2 types of SVM scheme: 7-segment or 5-segment SVM (as shown in green comment lines)

```

123 /* ----- SVM Switching Sequences ----- */
124 #define SVM_SWITCHING_SCHEME STANDARD_SVM_7_SEGMENT /*1. STANDARD_SVM_7_SEGME
125
126 #define ADC_ALTERNATE_REFERENCE DISABLED /*1. ENABLED 2. DISA
127 /* ----- Advance Motor Stop Conditional Handling ----- */
128 #define ADVANCE_CONDITIONAL_MOTOR_STOP DISABLED /*1. ENABLED 2. DIS
129 /* ----- Recommended Configuration (Strongly Influenced by Execution Time) ----- */
130 #if(CURRENT_SENSING == USER_SINGLE_SHUNT_CONV)
131 /* ----- Reference Speed Adjustment Method ----- */
132 #define SETTING_TARGET_SPEED BY_POT_ONLY
133 /* ----- Add d-q voltage decoupling components ----- */
134 #define DQ_DECOUPLING DISABLED /*1. ENABLED 2. DISA
135 /* ----- Watch Dog Timer Activation ----- */
136 #define WATCH_DOG_TIMER DISABLED /*1. ENABLED 2. DISA
137 /* ----- FOC Control Safety Protection ----- */
138 #define VDC_UNDER_OVERVOLTAGE_PROTECTION DISABLED /*1. ENABLED 2. DISA
139 #define OVERCURRENT_PROTECTION DISABLED /*1. ENABLED 2. DISA
140
141 #else
142 /* ----- Add d-q voltage decoupling components ----- */
143 #define DQ_DECOUPLING ENABLED /*1. ENABLED 2. DI
144 #define WATCH_DOG_TIMER ENABLED /*1. ENABLED 2. DI
145 /* ----- FOC Control Safety Protection ----- */
146 #define VDC_UNDER_OVERVOLTAGE_PROTECTION ENABLED /*1. ENABLED 2. DI
147 #define OVERCURRENT_PROTECTION ENABLED /*1. ENABLED 2. DI
148 #endif
  
```

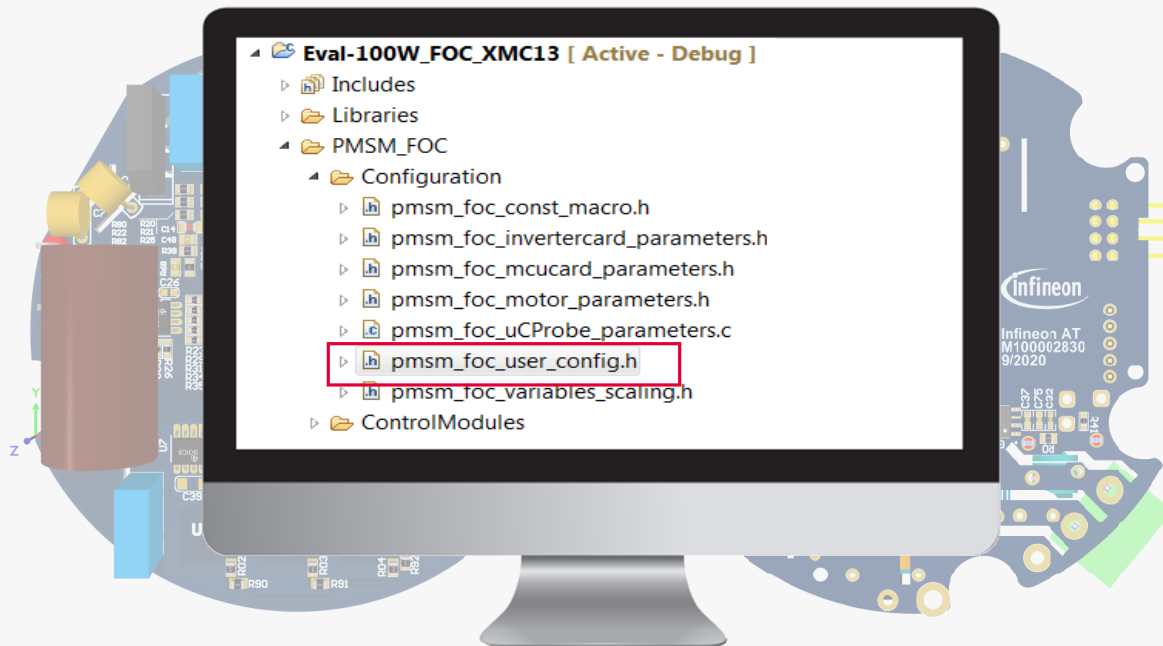
**i**

Standard 7-segment SVM is selected by default. The user does NOT need to change this!

# Input current and voltage protection setting

1

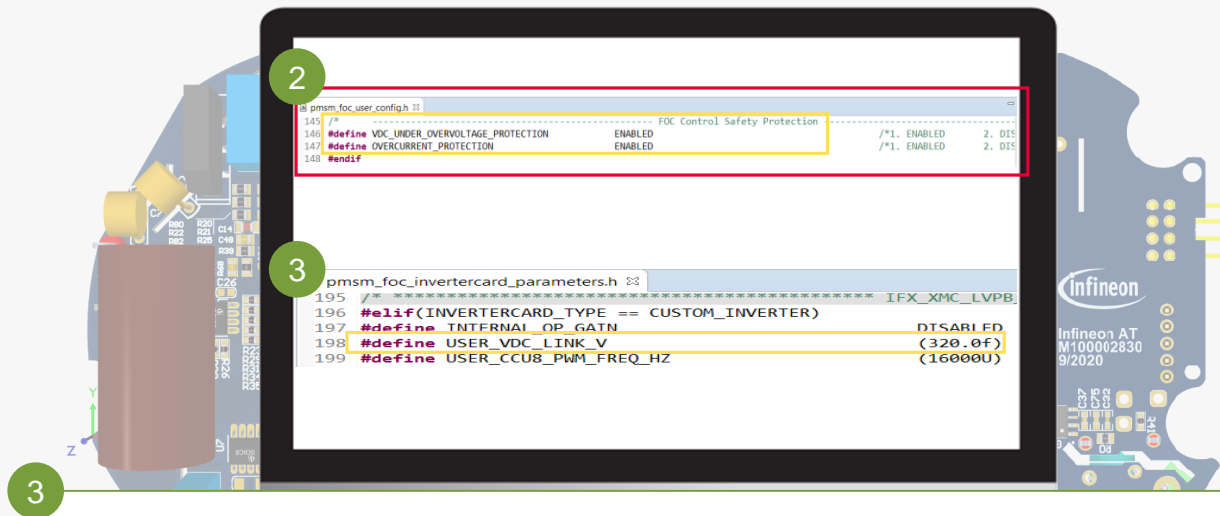
Double click on **pmsm\_foc\_user\_config.h**



# Input current and voltage protection setting

2

User can enable over-current and under-voltage protection (as shown in the green comment lines)



3

If the DC link voltage is  $<0.8$  or  $>1.2$  of `USER_VDC_LINK_V`, the PWM signal input to gate driver will go into high impedance and the motor will slowly stop

# Input current and voltage protection setting

4

If the average current flow through the DC link shunt resistor R57 is  $> \text{USER\_IDC\_MAXCURRENT\_A}$ , the reference speed will reduce until the current is  $< \text{USER\_IDC\_MAXCURRENT\_A}$

**i**

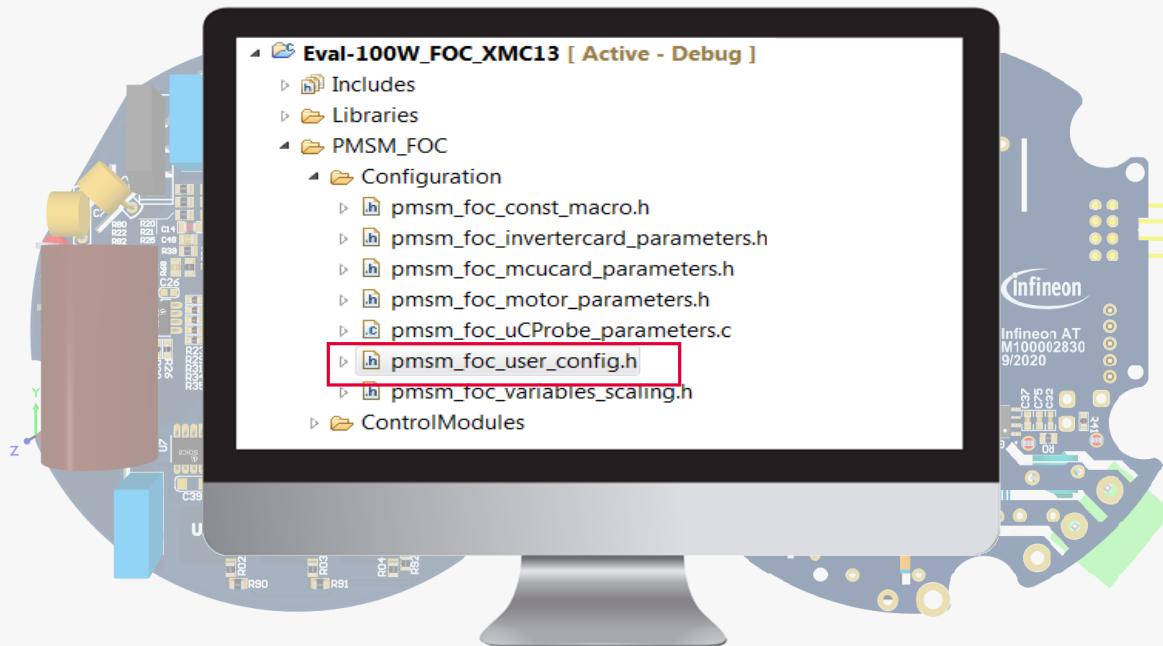
*Under-voltage and over-current protection have been selected by default.  
The user does NOT need to change this!*

*USER\_VDC\_LINK\_V and USER\_IDC\_MAXCURRENT\_A values might need to change depending on user motor setting*

# Input speed control setting

1

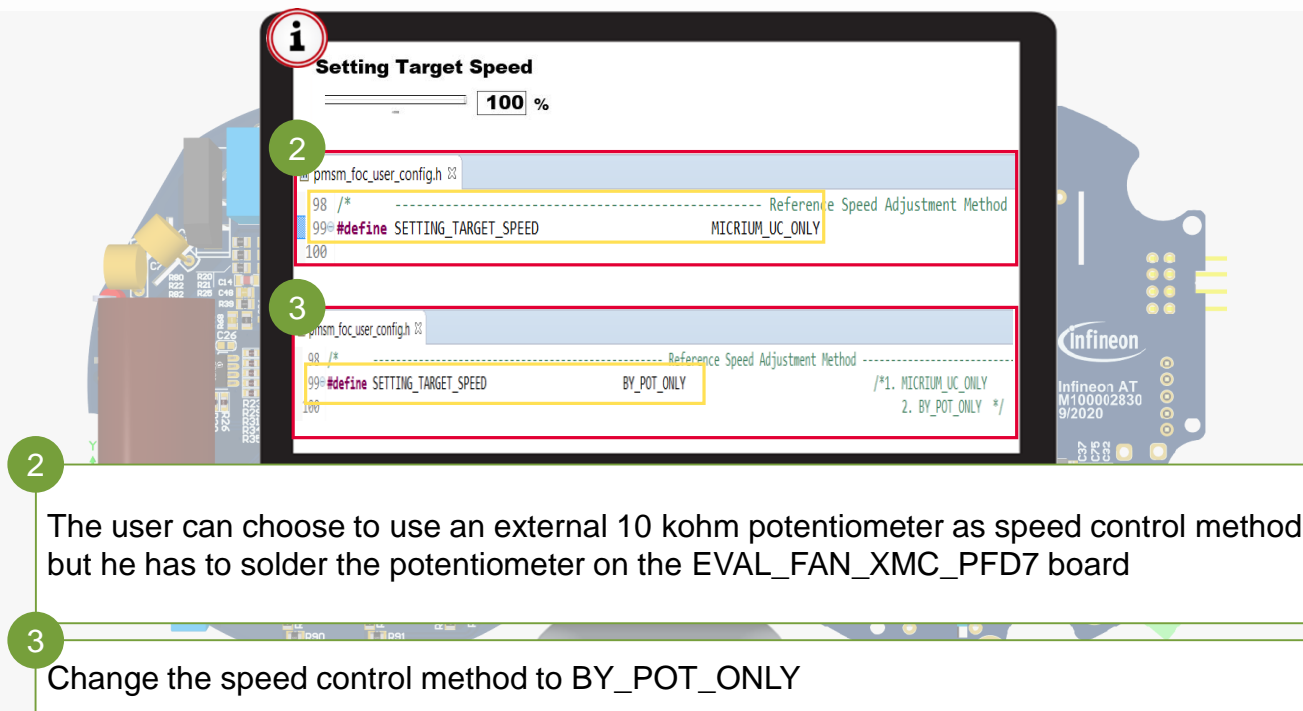
Double click on **pmsm\_foc\_user\_config.h**



# Input speed control setting



By default, the speed control method is using  $\mu$ C/Probe™ GUI's speed slider



**Setting Target Speed**

100 %

2

```

pism_foc_user_config.h
98 /* ----- Reference Speed Adjustment Method -----
99 #define SETTING_TARGET_SPEED MICRIUM_UC_ONLY
100

```

3

```

pism_foc_user_config.h
98 /* ----- Reference Speed Adjustment Method -----
99 #define SETTING_TARGET_SPEED BY_POT_ONLY /*1. MICRIUM_UC_ONLY
100 2. BY_POT_ONLY */

```

2

The user can choose to use an external 10 kohm potentiometer as speed control method but he has to solder the potentiometer on the EVAL\_FAN\_XMC\_PFD7 board

3

Change the speed control method to BY\_POT\_ONLY



# Compile and download the code

1

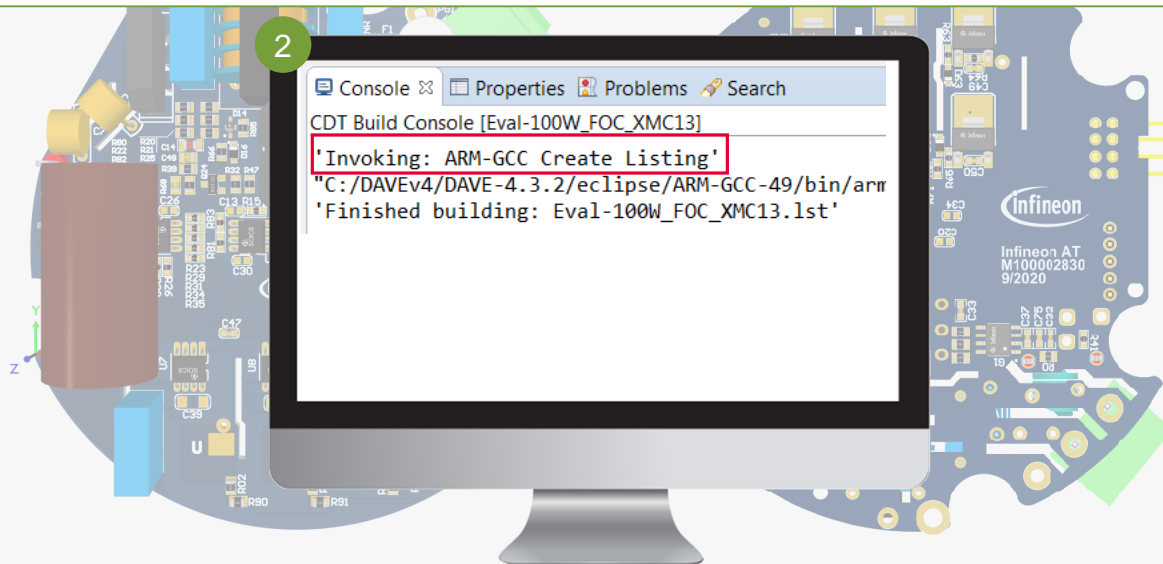
Click **Build Active Project**



2

Check **Eval-100W\_FOC\_XMC13.lst** created

2



# Compile and download the code

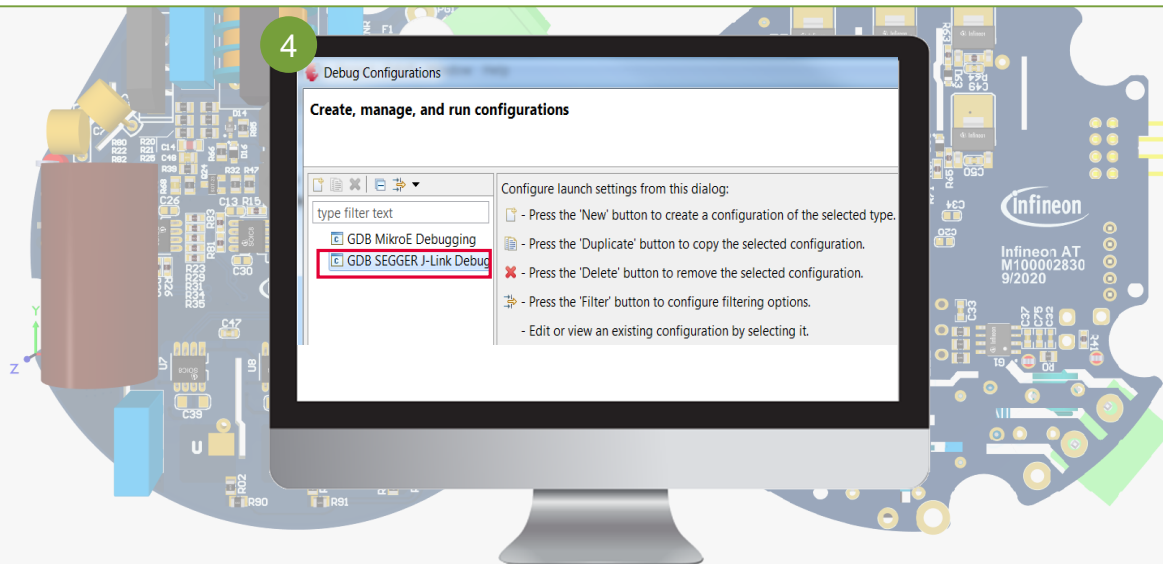
3

Click **Debug Configuration**



4

Select **GDB SEGGER J-Link Debugging**



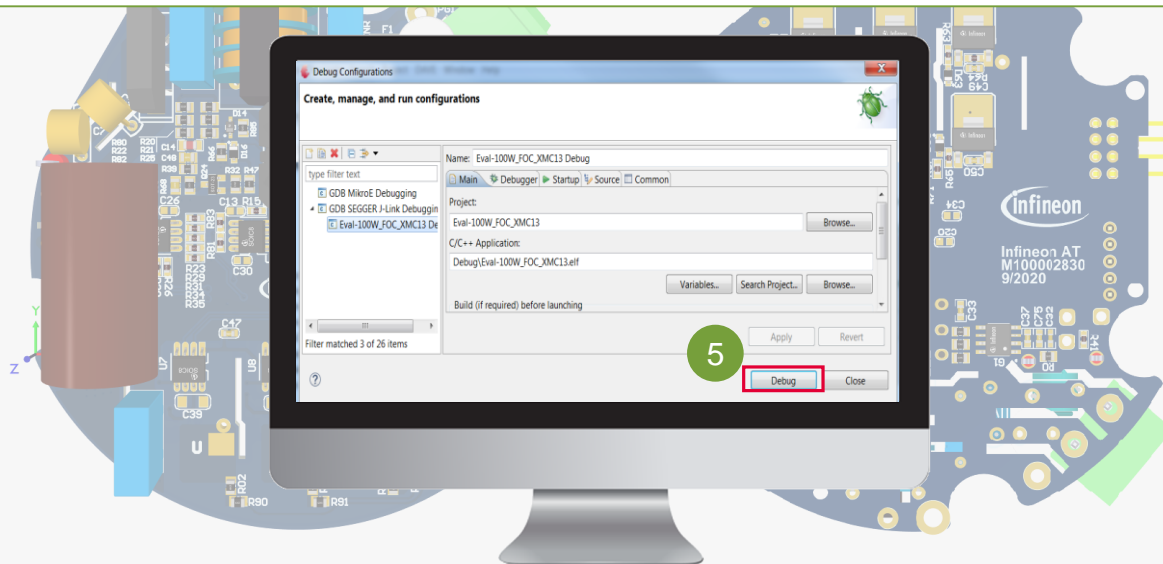
# Compile and download the code

5

Click the **Debug** button to download the code

6

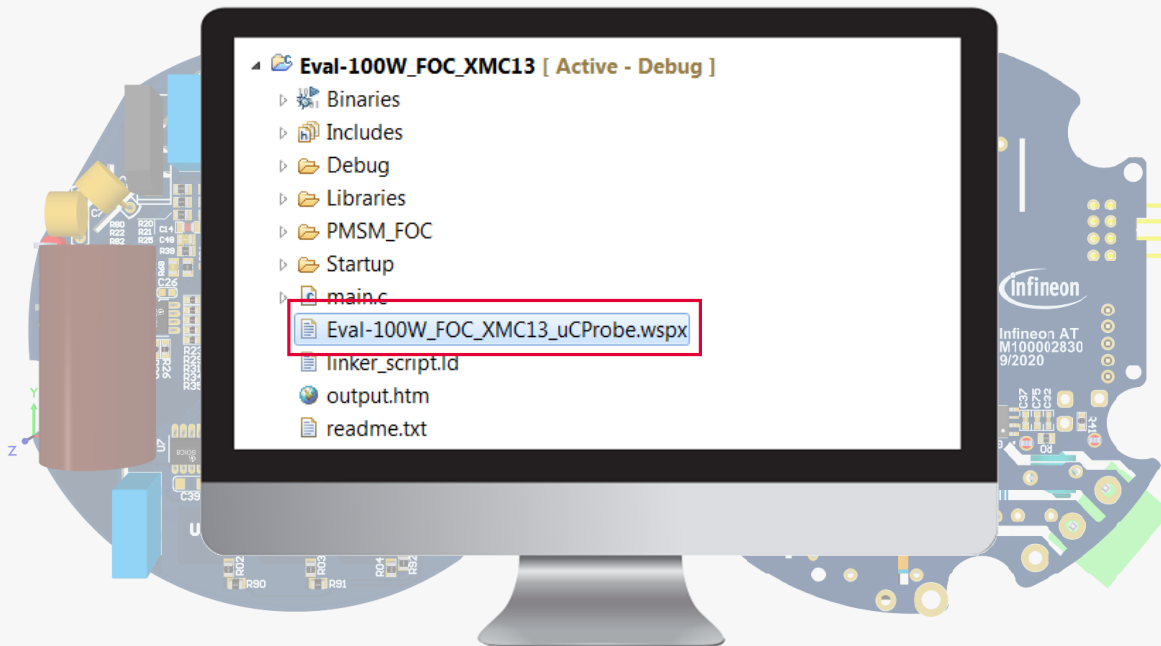
Click the **Resume** button to start the application



# Using $\mu$ C/Probe™ GUI

1

Open the \*.wspx  $\mu$ C/Probe™ project by double click on the \*.wspx file



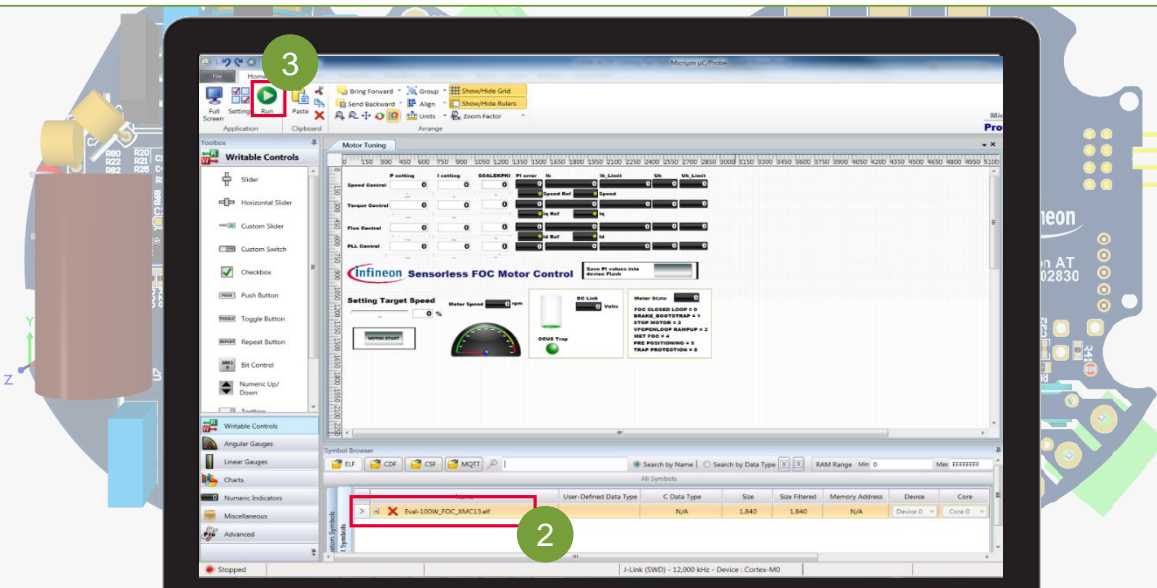
# Using $\mu$ C/Probe™ GUI

2

Check the \*.elf file is attached to  $\mu$ C/Probe™ project

3

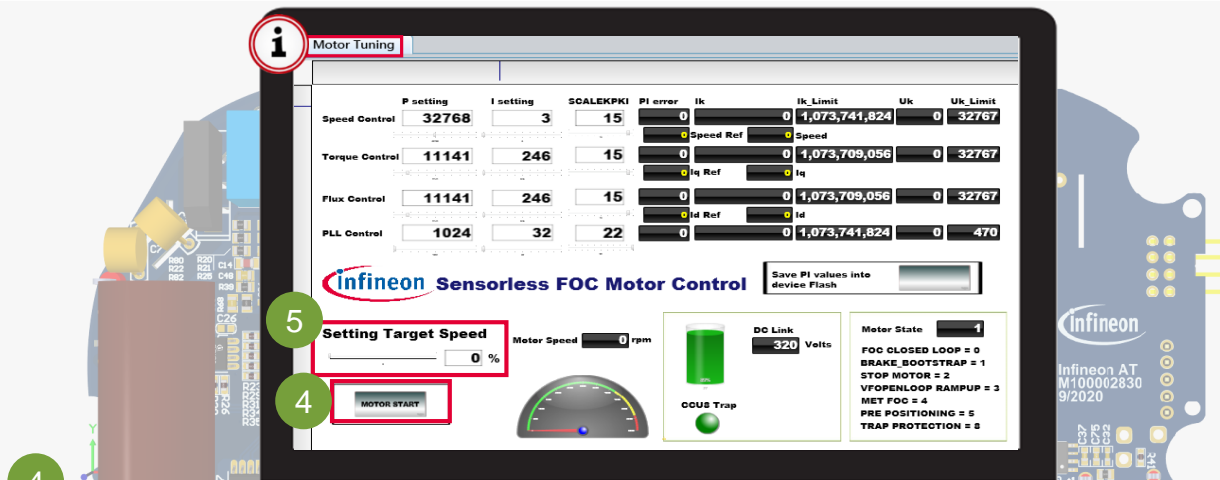
Click the **Run** button



# Using $\mu$ C/Probe™ GUI



The default  $K_p$ ,  $K_i$  values from the `pmsm_foc_motor_parameters.h` file are automatically displayed in the Motor Tuning page



4

Click the **MOTOR START** button

5

Then use the target speed slider or key in directly to set the percentage of maximum speed desired. The motor will start to spin

# Using $\mu$ C/Probe™ GUI

6

Modify the  $K_p$ ,  $K_i$  and **SCALEKPKI** values of the **Speed Control** and **PLL Control** loops to get optimum motor behaviour.  
The default  $K_p$ ,  $K_i$  and **SCALEKPKI** values for Torque Control and Flux Control normally do not need to change

7

Type in the desired values and hit **Enter**

8

Increasing the **SCALEKPKI** by 1 will reduce the gain by half

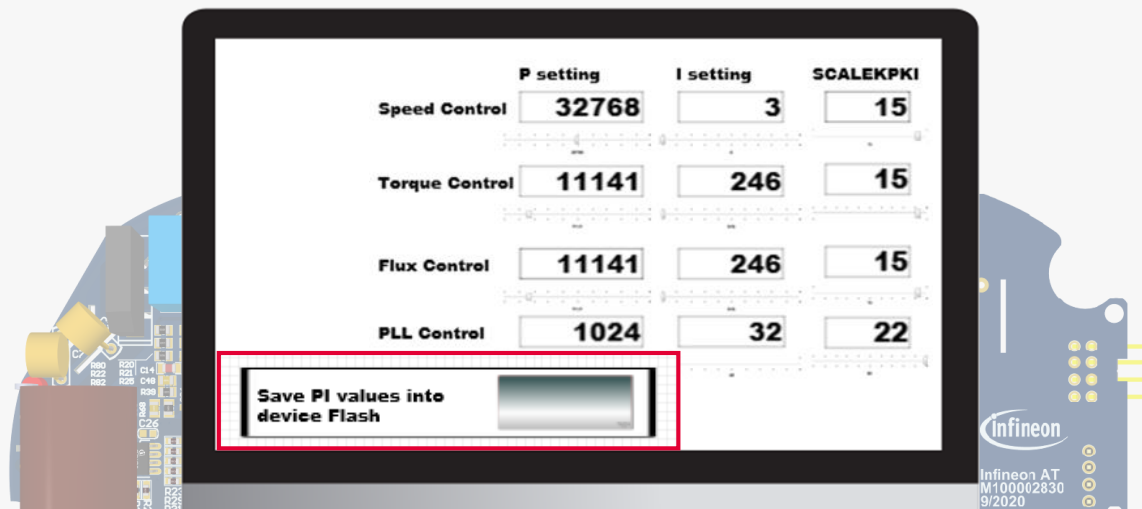


Under the **P setting** column are the  $K_p$  value for the 4 control loops

Under the **I setting** column are the  $K_i$  value for the 4 control loops

	P setting	I setting	SCALEKPKI
Speed Control	32768	3	15
Torque Control	11141	246	15
Flux Control	11141	246	15
PLL Control	1024	32	22

# Using $\mu$ C/Probe™ GUI



9

Click this button to save  $K_p$ ,  $K_i$ , SCALEKPKI values to Flash.  
The motor will stop

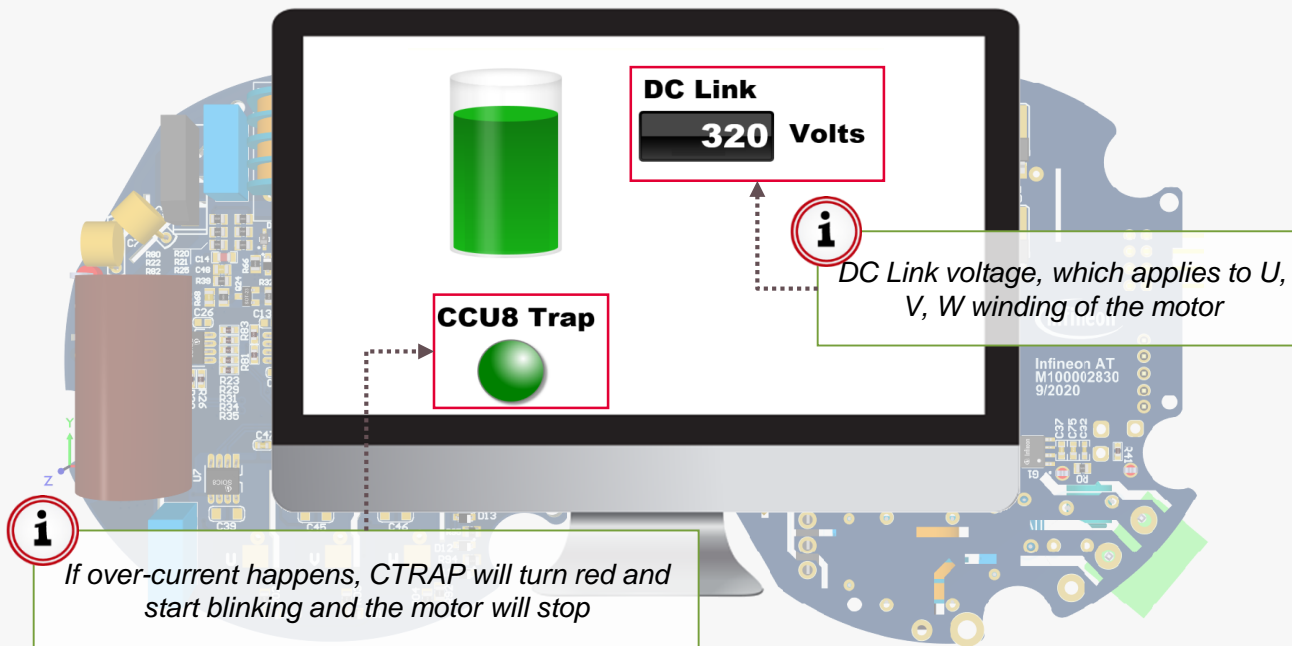
i

*This saving function only works if the motor is rotating normally*



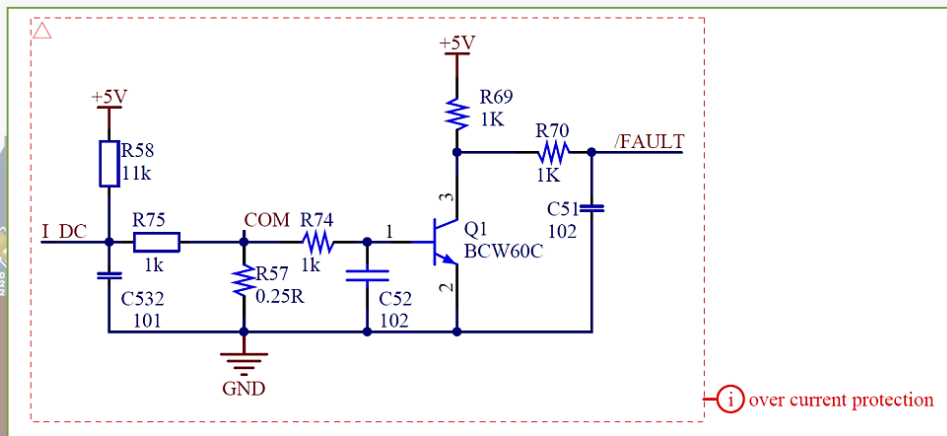
# Using $\mu$ C/Probe™ GUI

## Over-current and under-voltage protections



# Using $\mu\text{C}/\text{Probe}^{\text{TM}}$ GUI

## Over-current and under-voltage protections



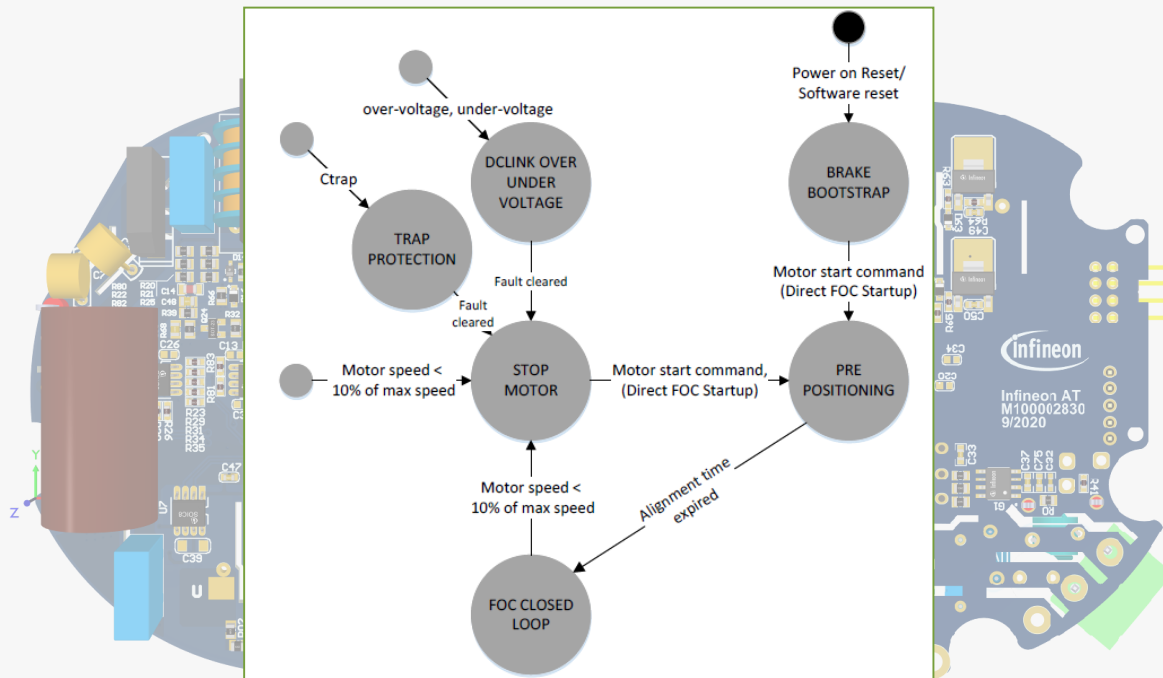
On the Eval-100W\_FOC board, the  $V_{BE(th)}$  of BCW60C is 0.7 V

The trip current  $I_{trip}$  is  $0.7/0.25=2.8$  A

When the 3-phase motor currents exceed its trip current, the CTRAP signal is activated and over-current detected

# Using $\mu$ C/Probe™ GUI

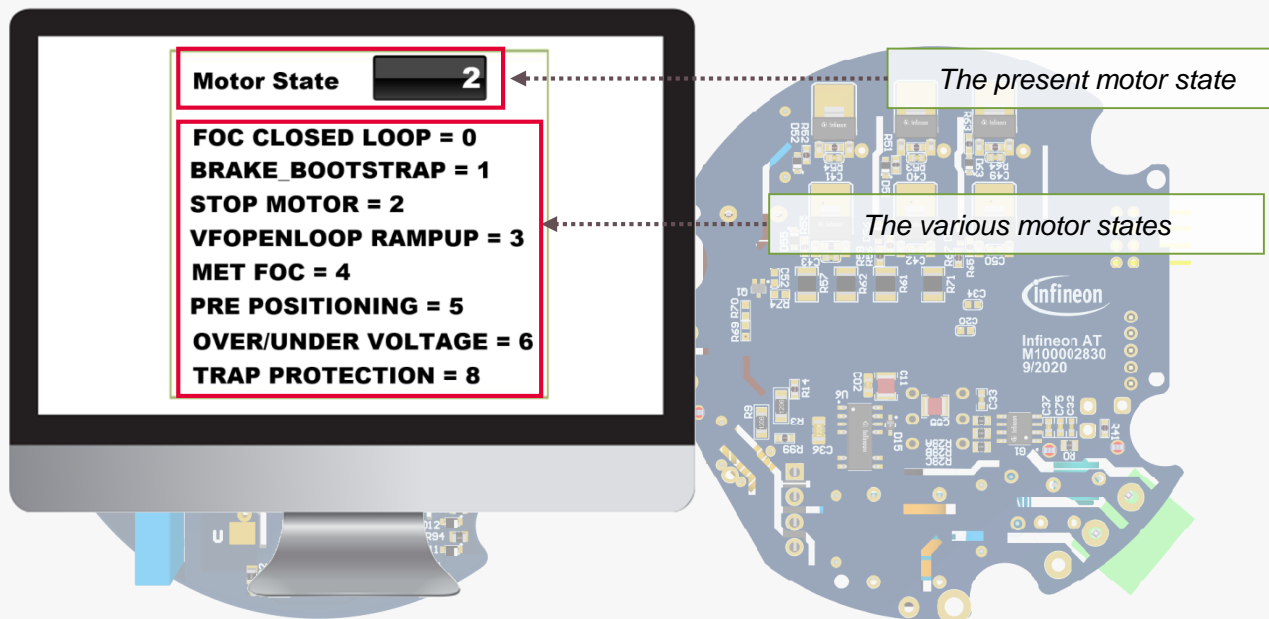
## PMSM FOC state machine panel



*Direct FOC startup state machine diagram*

# Using $\mu$ C/Probe™ GUI

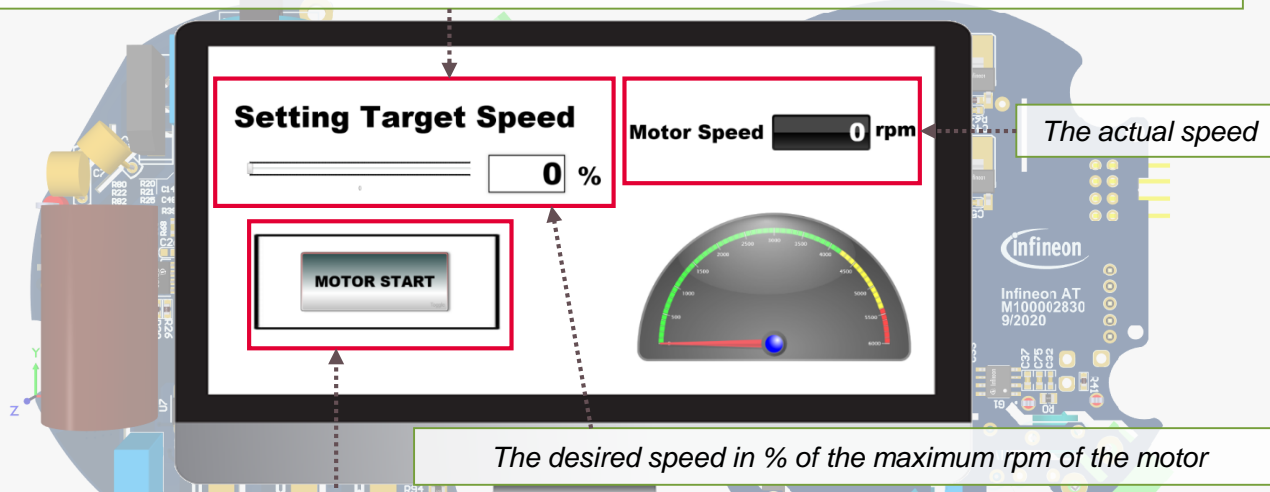
## PMSM FOC state machine panel



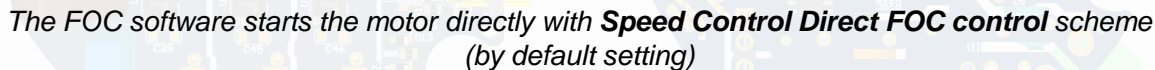
# Using $\mu$ C/Probe™ GUI

Motor speed, motor start/stop control panel

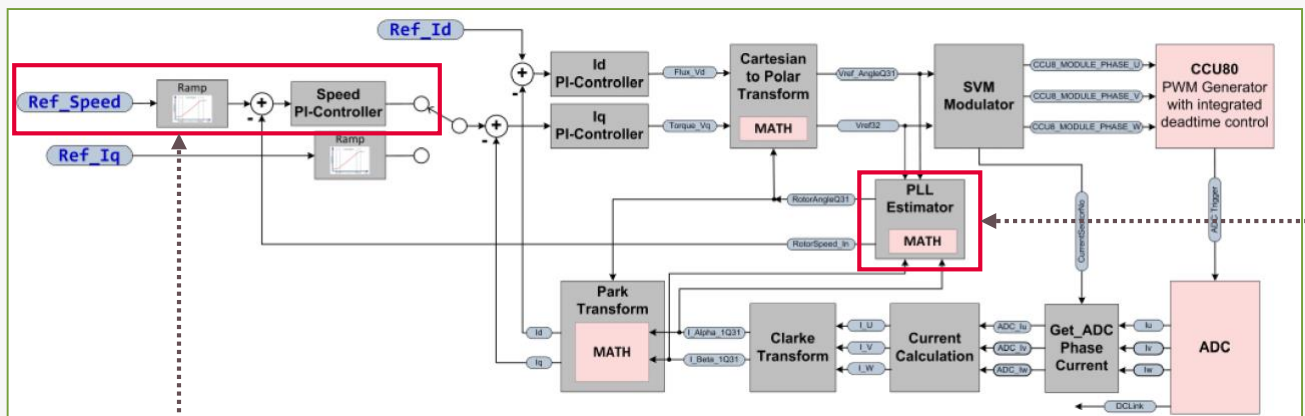
Use the slider to set the desired speed (% of motor maximum speed in rpm).  
The motor should start to run if **MOTOR START** is clicked



1) Click this **MOTOR START** button. 2) Then, set the desired speed for the motor to start. The wording on the button will change to **MOTOR STOP**. 3) Click this button to stop the motor



# Tuning of $K_P$ , $K_I$ value using the $\mu C/Probe^TM$



PMSM FOC block diagram



Remove the Speed control here if using  
**TORQUE\_CONTROLLED\_DIRECT\_FOC** scheme

Tune the PLL control loop here

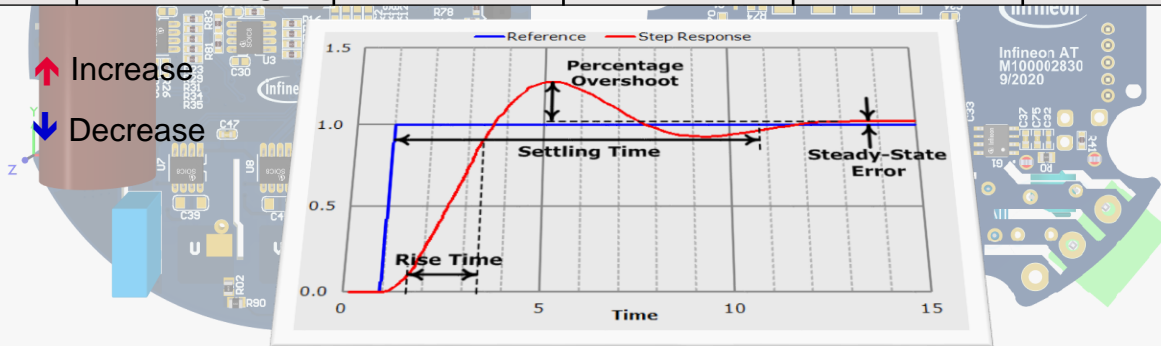
If the **MOTOR START** button was clicked and the motor does not spin after the desired speed slider is set to 30%, for example, the user should change the **My\_FOC\_CONTROL\_SCHEME** in `pmsm_foc_user_config.h` to **TORQUE\_CONTROLLED\_DIRECT\_FOC**, re-compile and reload

So that the user can tune the  $K_P$ ,  $K_I$  and SCALEKPKI of PLL, control loop first

# Tuning of $K_P$ , $K_I$ value using the $\mu C/Probe^{\text{TM}}$

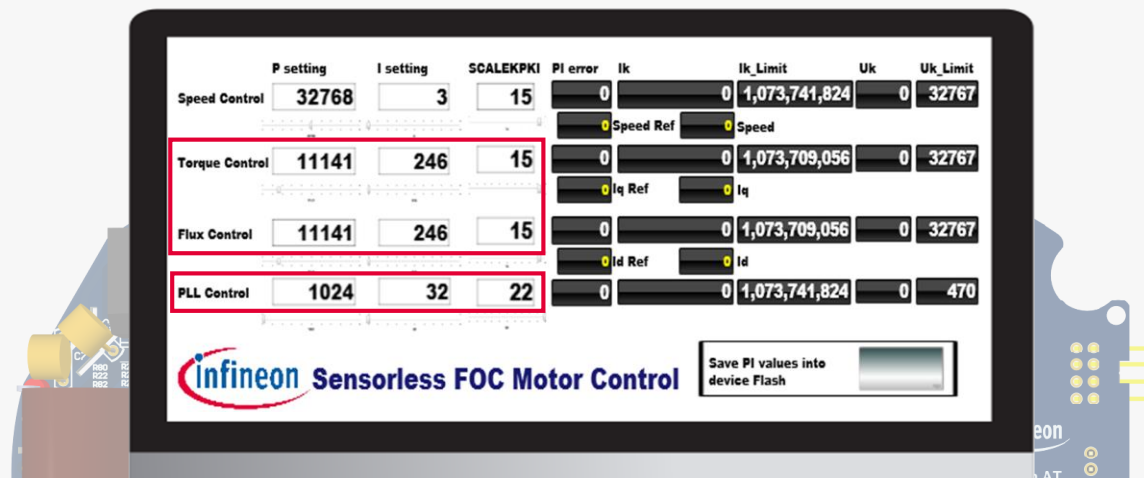
Effects of increasing proportional gain  $K_P$  or integral gain  $K_I$  of PI controller independently

	Gain change	Effects on step response characteristics			
		Rise time	Overshoot	Settling time	Steady-state error
1	$K_P \uparrow$ $K_I$ unchanged	$\downarrow$ 😊	$\uparrow$ ☹️	Minor Change 😊	$\downarrow$ 😊
2	$K_I \uparrow$ $K_P$ unchanged	$\downarrow$ 😊	$\uparrow$ ☹️	$\uparrow$ ☹️	Eliminate 😊





# Tuning of $K_P$ , $K_I$ value using the $\mu C/Probe^TM$



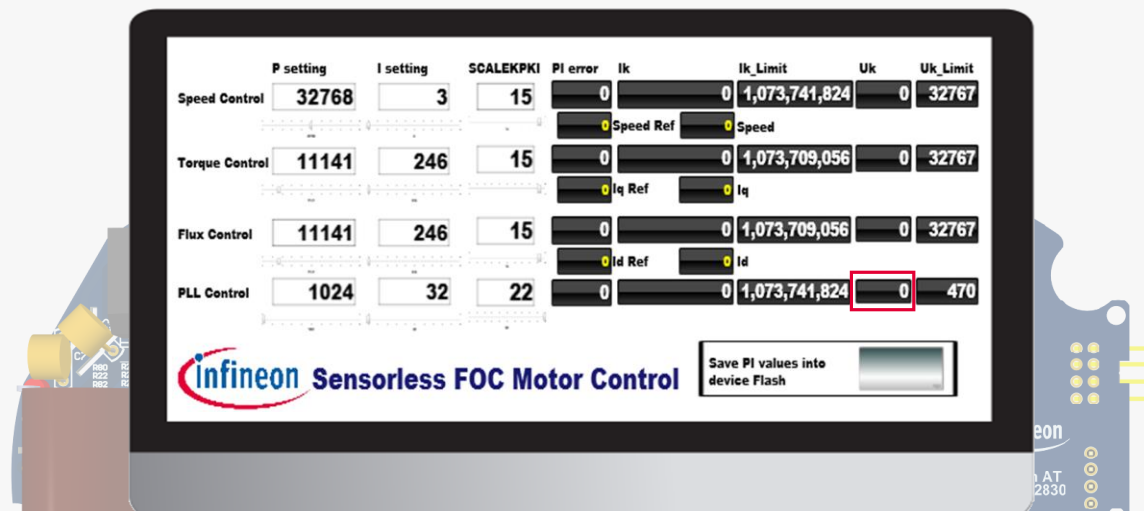
1

The P setting, I setting and SCALEKPKI values for Torque and Flux PI Controllers are calculated from the physical motor and system parameters, and typically don't need to be tuned in the first iteration

2

The PLL Control parameters  $K_P$ ,  $K_I$ , SCALEKPKI should start to be modified if the motor cannot startup in FOC closed-loop smoothly

# Tuning of $K_P$ , $K_I$ value using the $\mu C/Probe^TM$



3

The target speed slider becomes the target torque slider under TORQUE\_CONTROLLED\_DIRECT\_FOC mode

4

Make sure the Uk value of PLL control does not hit its Uk\_Limit while tuning the PLL control parameters

# Tuning of $K_P$ , $K_I$ value using the $\mu\text{C}/\text{Probe}^{\text{TM}}$

	P setting	I setting	SCALEKPKI
Speed Control	32768	3	15
Torque Control	11141	246	15
Flux Control	11141	246	15
PLL Control	1024	32	22

↑ this value by 1 will ↓ gain of Speed controller by half

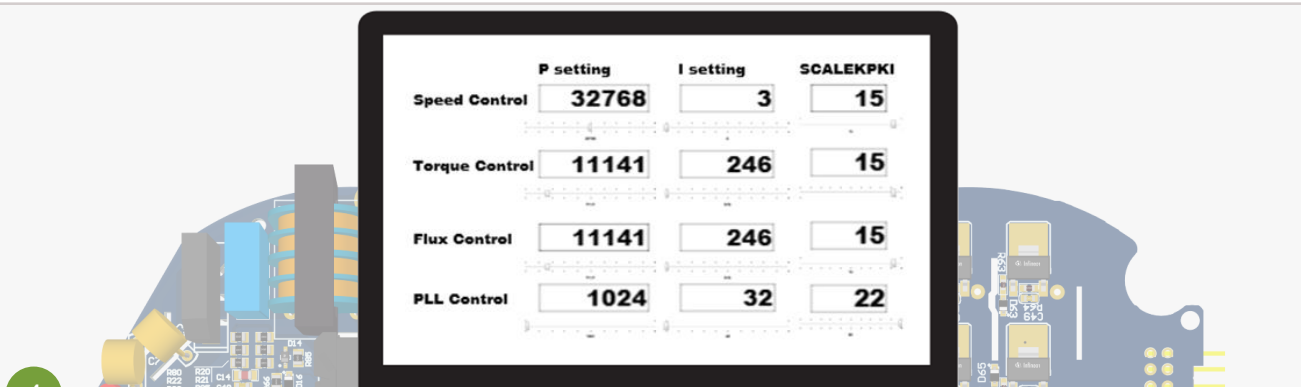
PI gains:  $K_p = \frac{P \text{ setting}}{2 \text{SCALEKPKI}}$ ,  $K_i = \frac{I \text{ setting}}{2 \text{SCALEKPKI}}$

↑ this value by 1 will ↓ gain of PLL estimator controller by half

3

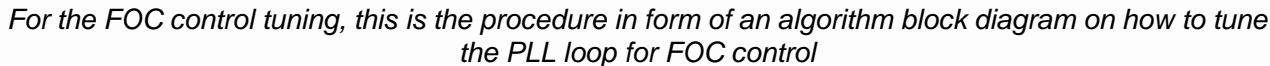
If the motor does not spin in FOC close loop, ↑ the SCALEKPKI of PLL Control and check the motor behavior. If the motor starts to move slowly, ↑ the SCALEKPKI further. Else, ↓ the SCALEKPKI.

# Tuning of $K_P$ , $K_I$ value using the $\mu C/Probe^TM$



	P setting	I setting	SCALEKPKI
Speed Control	32768	3	15
Torque Control	11141	246	15
Flux Control	11141	246	15
PLL Control	1024	32	22

- 4 After the motor is able to rotate smoothly under TORQUE\_CONTROLLED\_DIRECT\_FOC, remember to save the parameters by clicking the **Save PI** button
- 5 Change the My\_FOC\_CONTROL\_SCHEME in pmsm\_foc\_user\_config.h to SPEED\_CONTROLLED\_DIRECT\_FOC, re-compile and reload the code
- 6 Use the same tactic to tune the Speed Control parameters. No need to modify the PLL control parameters

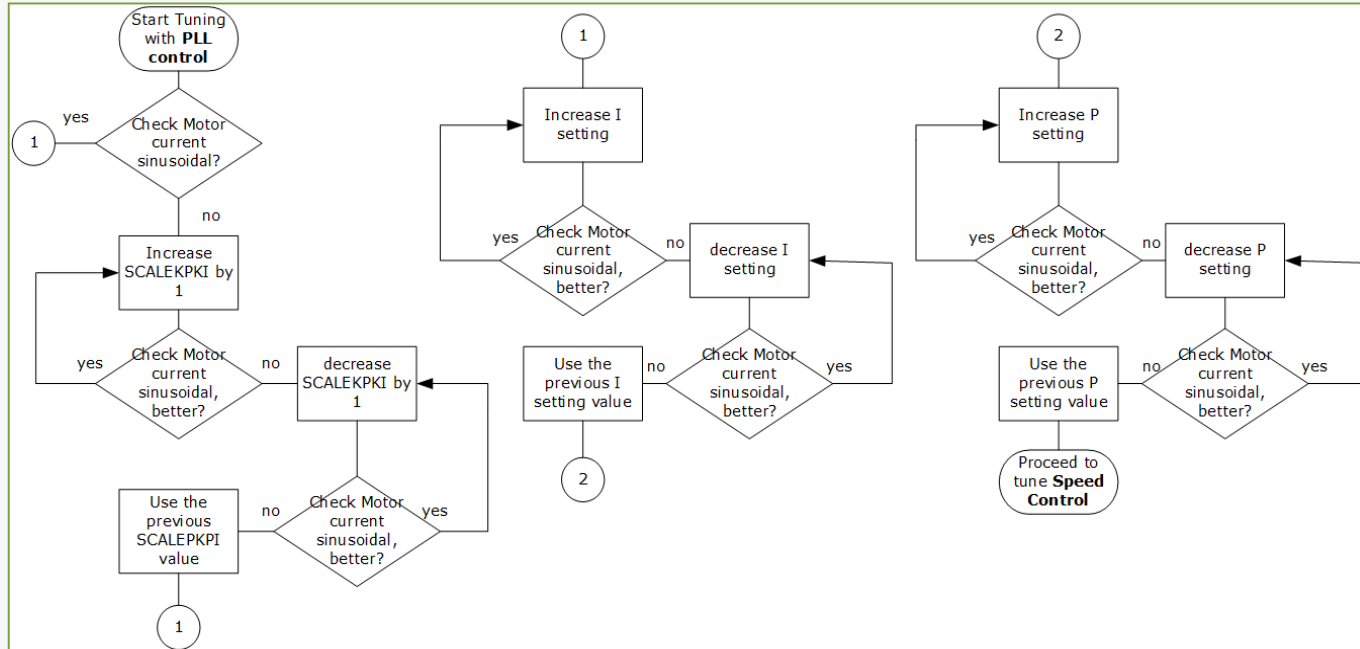


# General hints on tuning of SCALEKPKI, P, I value using the $\mu$ C/Probe™



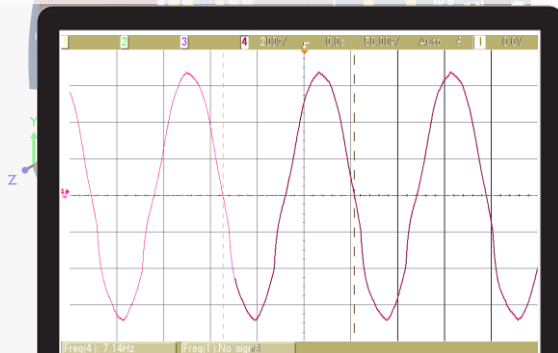
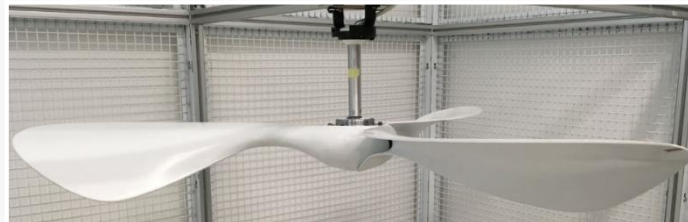
*For Speed Control tuning the user should follow the same flow*

*Typically, only these 2 PI control loops need to be tuned with the FOC software for EVAL\_FAN\_XMC\_PFD7 kit*

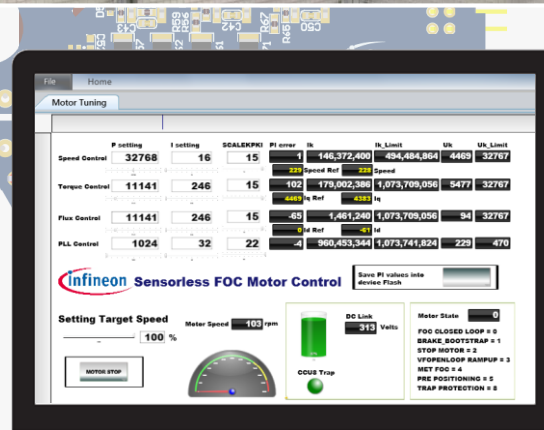


# Testing result with a ceiling fan

- › AC input: 230 V/50 Hz
- › Ceiling Fan motor specification:
  - Resistance/phase: 23 ohms
  - Inductance/phase: 65 mH
  - Pole pairs: 4
  - Max. speed: 105 rpm
- › Power consumption: 35 W @ 105 rpm



*The current waveform of phase U at max. speed*



# Agenda

---

1 Introduction to fan driver evaluation board

2 Hardware and software

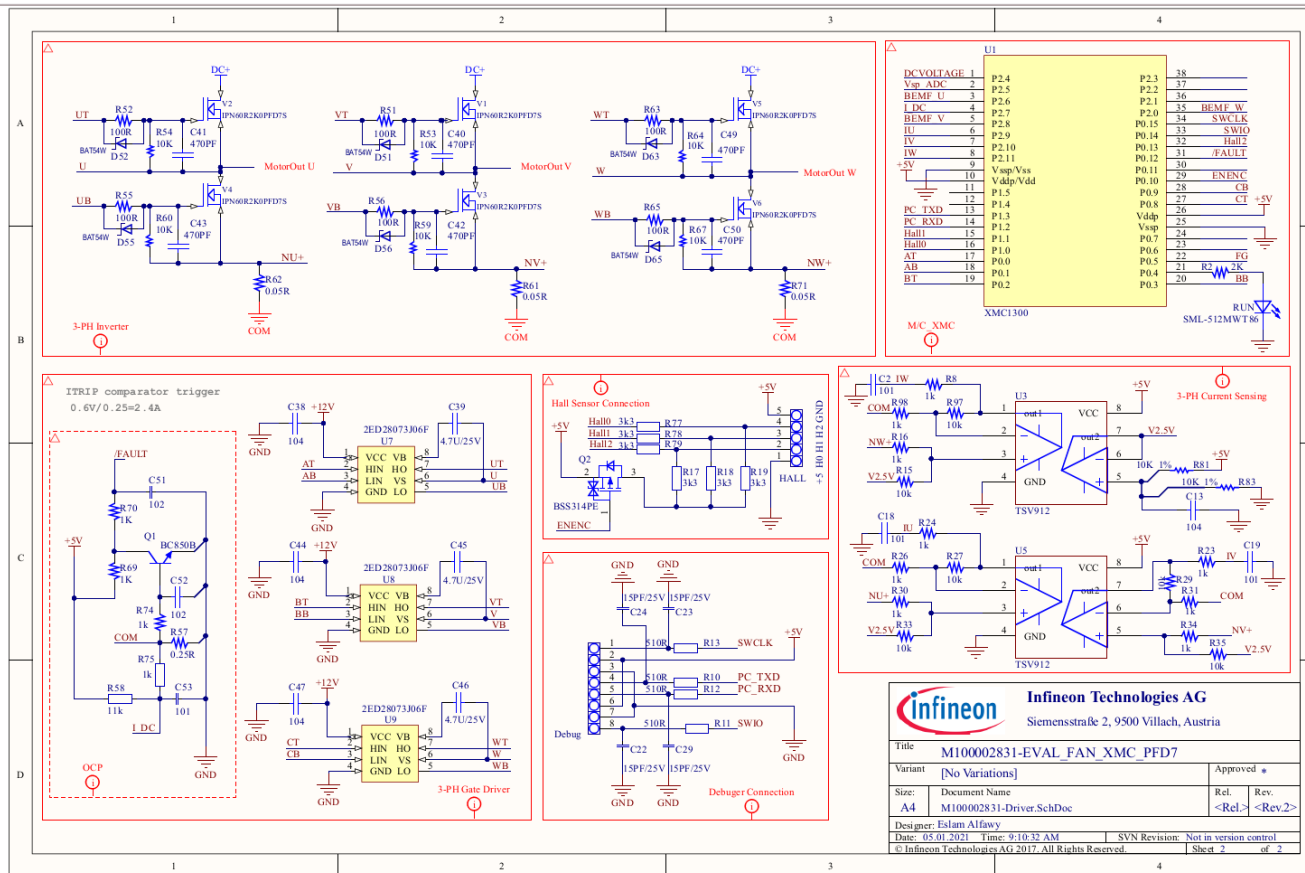
3 Tools

4 Getting started

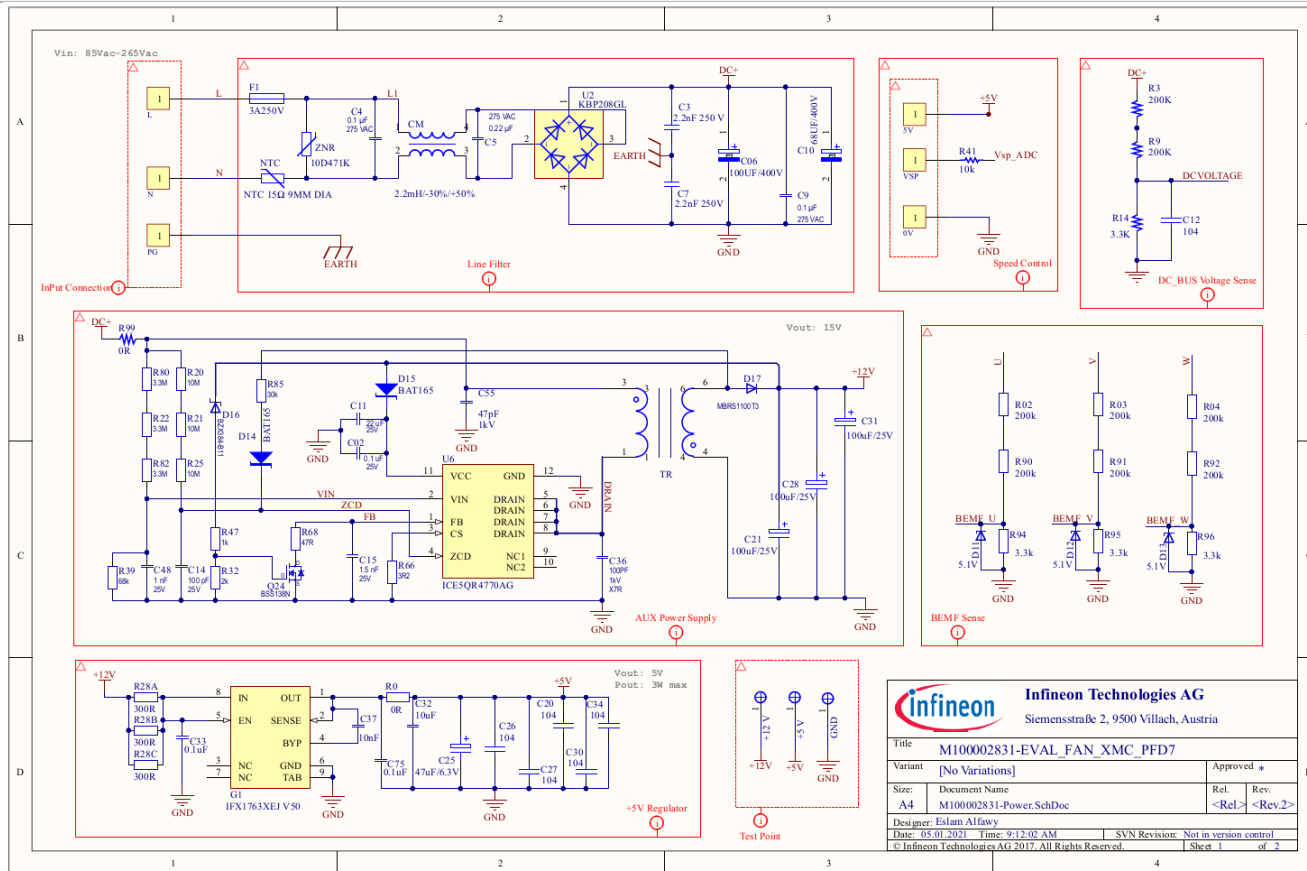
5 Resources



# Schematic of EVAL\_FAN\_XMC\_PFD7 board



# Schematic of EVAL\_FAN\_XMC\_PFD7 board



# Support material:

<p>Collaterals and brochures</p> <div> <div>Document</div> <div>Info</div> </div>	<ul style="list-style-type: none"> <li>› Product briefs</li> <li>› Selection guides</li> <li>› Application brochures</li> <li>› Presentations</li> <li>› Press releases, ads</li> </ul>	<ul style="list-style-type: none"> <li>› <a href="http://www.infineon.com/XMC">www.infineon.com/XMC</a></li> <li>› <a href="http://www.infineon.com/PFD7">www.infineon.com/PFD7</a></li> <li>› <a href="http://www.infineon.com/COOLSET">www.infineon.com/COOLSET</a></li> <li>› <a href="http://www.infineon.com/motorcontrol">www.infineon.com/motorcontrol</a></li> </ul>
<p>Technical material</p> <div> <div>Tools</div> <div>Evaluation Board</div> </div>	<ul style="list-style-type: none"> <li>› Application notes</li> <li>› Technical articles</li> <li>› Simulation models</li> <li>› Datasheet, MCDS files</li> <li>› PCB design data</li> </ul>	<ul style="list-style-type: none"> <li>› <a href="http://www.infineon.com/XMC">www.infineon.com/XMC</a></li> <li>› <a href="#">Kits and Boards</a></li> <li>› <a href="#">DAVE™</a></li> <li>› <a href="#">Software and Tool Ecosystem</a></li> </ul>
<p>Videos</p> <div>Play</div>	<ul style="list-style-type: none"> <li>› Technical videos</li> <li>› Product information Videos</li> </ul>	<ul style="list-style-type: none"> <li>› <a href="#">Infineon Media Center</a></li> <li>› <a href="#">XMC Mediathek</a></li> <li>› <a href="#">How to measure motor parameters</a></li> </ul>
<p>Contact</p> <div>Support</div>	<ul style="list-style-type: none"> <li>› Forums</li> <li>› Product support</li> </ul>	<ul style="list-style-type: none"> <li>› <a href="#">Infineon Forums</a></li> <li>› <a href="#">Technical Assistance Center (TAC)</a></li> </ul>

# Glossary abbreviations

Abbreviation	Full name
ADC	Analog-to-Digital Converter
FOC	Field-Oriented Control
PI Controller	Proportional–Integral Controller
PMSM	Permanent Magnet Synchronous Motor
PWM	Pulse Width Modulation
SVM	Space Vector Modulation
XMC™	Cross-Market Microcontrollers

# Disclaimer

---

The information given in this training materials is given as a hint for the implementation of the Infineon Technologies component only and shall not be regarded as any description or warranty of a certain functionality, condition or quality of the Infineon Technologies component.

Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this training material.

All the images used in this training are free for commercial use or free for use with attribution and were designed by Freepik.



Part of your life. Part of tomorrow.