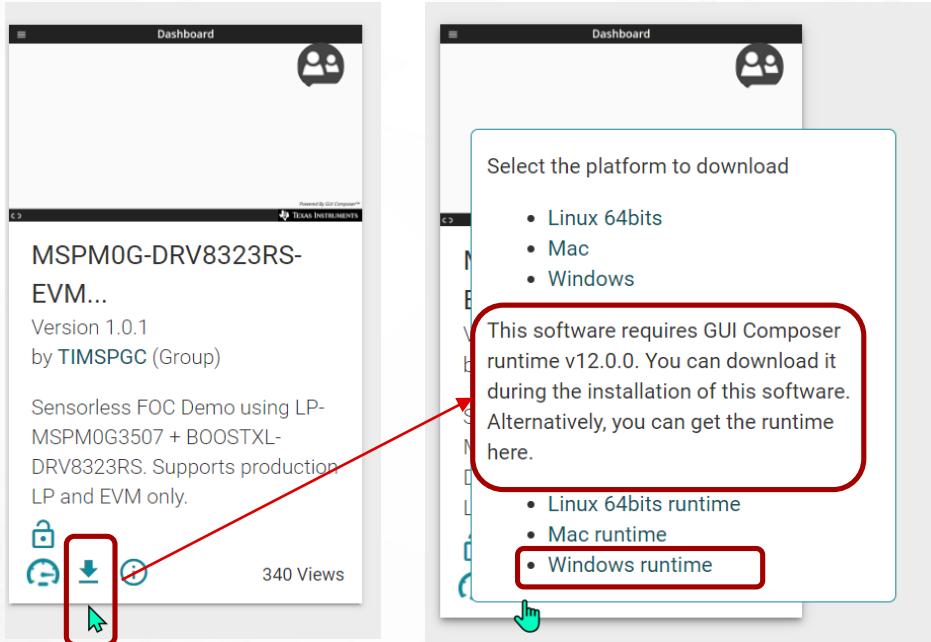


# MSPM0G-DRV8323RS GUI

# MSPM0G-DRV8323RS GUI

Access link:

<https://dev.ti.com/gallery/view/TIMSPGC/MSPM0G-DRV8323RS-EVM-GUI/ver/1.0.1/>



Install the software environment

# MSPM0G-DRV8323RS GUI

Access link:

<https://dev.ti.com/gallery/view/TIMSPGC/MSPM0G-DRV8323RS-EVM-GUI/ver/1.0.1/>

 MSPM0G-DRV8323RS-EVM-GUI

**Control**

Enable Motor  Disable Closed loop  Reverse

Outer Loop Speed  Motor State IDLE

Speed Reference (Hz) 0.00  Speed (Hz)

Id Reference (A) 0.000  Id Feedback (A)

Iq Reference (A) 0.000  Iq Feedback (A)

**Startup** Method IPD

IPD Thresh count 1  IPD Freq (Hz) 50

**Rampup**

Rampup current (A) 0.000  Rampup speed rate (Hz/s) 0.00000

Rampup Target (Hz) 0.00

**Tuning**

Speed Kp 0.000000  Speed Ki 0.000000

Id Kp 0.000000  Id Ki 0.000000

Iq Kp 0.000000  Iq Ki 0.000000

**CL**

PI Speed Divider 10  CL SpeedRef ramp (Hz/s) 0.00000

**Monitors**

VDC (V)

**Fault Status**

**CLEAR FAULTS**

Ext Fault OC OV UV

**Protection**

Over Current Limit (A) 0  Over Voltage Limit (V) 0  Under Voltage Limit (V) 0

**Speed (Hz)**

Speed (Hz) vs Time (s) graph showing a step increase from 0 to 4 Hz at t=0.

**Id Current (A)**

Id (A) vs Time (s) graph showing a step increase from 0 to 4 A at t=0.

**Iq Current (A)**

Iq (A) vs Time (s) graph showing a step increase from 0 to 4 A at t=0.

**TI Cloud Agent INSTALLER**

**Setup - TI Cloud Agent**

Welcome to the TI Cloud Agent Setup Wizard. Installing TI Cloud Agent will help connect TI cloud tools to your local hardware.

< Back Next > Cancel

**Might need install the TI Cloud Agent if this page shows the download request.**

 TEXAS INSTRUMENTS

# MSPM0G-DRV8323RS GUI

## Introduction

# MSPM0G-DRV8323RS GUI

## Introduction

### Control

Enable Motor

Disable Closed loop

Reverse

Outer Loop Speed  Motor State IDLE

Speed Reference (Hz) 0.00  Speed (Hz)

Id Reference (A) 0.000  Id Feedback (A)

Iq Reference (A) 0.000  Iq Feedback (A)

Startup Method IPD

IPD Thresh count 1  IPD Freq (Hz) 50

Rampup

Rampup current (A) 0.000  Rampup speed rate (Hz/s) 0.00000

Rampup Target (Hz) 0.00

Tuning

Speed Kp 0.00000  Speed Ki 0.00000

Id Kp 0.00000  Id Ki 0.00000

Iq Kp 0.00000  Iq Ki 0.00000

CL

PI Speed Divider 10  CL SpeedRef ramp (Hz/s) 0.00000  Monitors VDC (V)

### Motor & Drive Parameters

R (ohms) 0.00000  L (H) 0.000000  KE (V/Hz) 0.000000

Max Freq (Hz) 25  PWM Freq (Hz) 10000  Deadband (ns) 100

CSA Gain 0

Protection

Over Current Limit (A) 0  Over Voltage Limit (V) 0  Under Voltage Limit (V) 0

Fault Status

Ext Fault  OC  OV  UV

Speed (Hz)

Speed (Hz)

Current (A)

Current (A)

**Tuning**

Speed Kp 0.00000  Speed Ki 0.00000

Id Kp 0.00000  Id Ki 0.00000

Iq Kp 0.00000  Iq Ki 0.00000

**Regulator Params.**

The current loop params are related to motor params.

The speed loop params are related to motor, load, and machine structure.

Regulator Params.  
The current loop params are related to motor params.  
The speed loop params are related to motor, load, and machine structure.

# MSPM0G-DRV8323RS GUI

## Introduction

**Control**

Outer Loop: Speed, Motor State: IDLE

Speed Reference (Hz): 0.00, Speed (Hz): 0.00

Id Reference (A): 0.000, Id Feedback (A): 0.000

Iq Reference (A): 0.000, Iq Feedback (A): 0.000

Startup Method: IPD

IPD Thresh count: 1, IPD Freq (Hz): 50

Rampup: Rampup current (A): 0.000, Rampup speed rate (Hz/s): 0.00000

Rampup Target (Hz): 0.00

**Motor & Drive Parameters**

R (ohms): 0.00000, L (H): 0.000000, KE (V/Hz): 0.000000

Max Freq (Hz): 25, PWM Freq (Hz): 10000, Deadband (ns): 100

CSA Gain: 0

**Fault Status**

**Protection**

Over Current Limit (A): 0, Over Voltage Limit (V): 0, Under Voltage Limit (V): 0

**Control Params.**  
User can select the control loop and set the reference input for the loop.

User can select the start up method for different usage.  
Rampup stage can set the acceleration and loading current.

**CL**

PI Speed Divider: 10, CL SpeedRef ramp (Hz/s): 0.00000

**Monitors**

VDC (V): 0.00000

Id Current (A)

Iq Current (A)

User can set the speed loop divider – how many times current loop running with a speed loop running.

>Error: Failed to connect: CS\_DAP\_0: Error initializing emulator: (Error -260 @ 0x0)

TEXAS INSTRUMENTS

# MSPM0G-DRV8323RS GUI

## Introduction

**Control**

Enable Motor

Disable Closed loop

Reverse

**Motor & Drive Parameters**

R (ohms) 0.00000

L (H) 0.000000

KE (V/Hz) 0.000000

Max Freq (Hz) 25

PWM Freq (Hz) 10000

Deadband (ns) 100

CSA Gain 0

**Protection**

Over Current Limit (A) 0

Over Voltage Limit (V) 0

Under Voltage Limit (V) 0

**Fault Status**

Ext Fault

OC

OV

UV

**Startup** Method IPD

IPD Thresh count 1

IPD Freq (Hz) 50

**Rampup**

Rampup current (A) 0.000

Rampup speed rate (Hz/s) 0.00000

Rampup Target (Hz) 0.00

**Tuning**

Speed Kp 0.000000

Speed Ki 0.000000

Id Kp 0.000000

Id Ki 0.000000

Iq Kp 0.000000

Iq Ki 0.000000

**CL**

PI Speed Divider 10

CL SpeedRef ramp (Hz/s) 0.00000

**Monitors**

Speed (Hz)

Speed (Hz) graph: A line graph showing Speed (Hz) on the y-axis (from -1 to 4) and time on the x-axis. The line starts at 0, goes up to 1, then down to -1, and back to 0.

Id Current (A)

Id Current (A) graph: A line graph showing Id Current (A) on the y-axis (from 0 to 4) and time on the x-axis. The line starts at 0, goes up to 2, then down to 0, and back to 0.

**Fault status monitor.**  
It will show if there is any out of the protection limit.  
Click on the red button will clear the fault if exist.

## Fault status monitor.

It will show if there is any out of the protection limit or DRV ext fault occurs, and then stop the motor.

Click on the red button will clear the fault status if it doesn't exist.

# MSPM0G-DRV8323RS GUI

## Introduction

**MSPM0G-DRV8323RS-EVM-GUI**

**Control**

Enable Motor  Disable Closed loop  Reverse

Outer Loop Speed  Motor State IDLE

Speed Reference (Hz) 0.00  Speed (Hz)

Id Reference (A) 0.000  Id Feedback (A)

Iq Reference (A) 0.000  Iq Feedback (A)

**Startup** Method IPD  IPD Thresh count 1  IPD Freq (Hz) 50

**Rampup**

Rampup current (A) 0.000  Rampup speed rate (Hz/s) 0.00000

Rampup Target (Hz) 0.00

**Tuning**

Speed Kp 0.000000  Speed Ki 0.000000

Id Kp 0.000000  Id Ki 0.000000

Iq Kp 0.000000  Iq Ki 0.000000

**CL**

PI Speed Divider 10  CL SpeedRef ramp (Hz/s) 0.00000  VDC (V)

**Motor & Drive Parameters**

R (ohms) 0.00000  L (H)

Max Freq (Hz) 25  PWM Freq (Hz)

CSA Gain 0

Over Current Limit (A) 0  Over Voltage Limit (V)

**Fault Status** **CLEAR FAULTS**

**Watch Window**

User can timely get the current motor status, including the speed, d-axis current, q-axis current.

**Speed (Hz)**

Speed (Hz)

**Id Current (A)**

Id (A)

**Iq Current (A)**

Iq (A)

**Monitors**

PI Speed Divider 10  CL SpeedRef ramp (Hz/s) 0.00000  VDC (V)

**Errors**

Error: Failed to connect: CS\_DAP\_0: Error initializing emulator: (Error -260 @ 0x0)

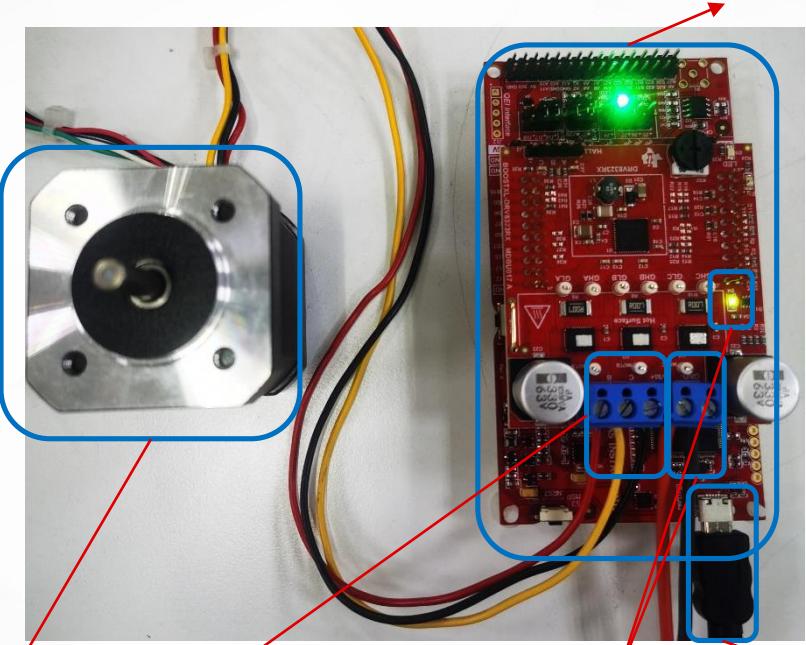
**TI** TEXAS INSTRUMENTS

# Step by Step Operations

## - GUI

# Hardware Setup

MSPM0G3507 LaunchPad +  
DRV8323RS EVM

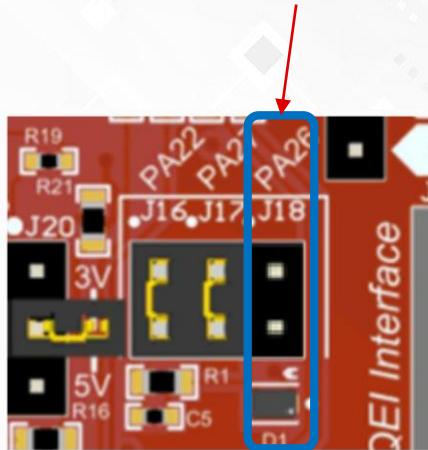


Motor  
Motor Phase U/V/W

Power supply 4.5~34V

Turn on the power supply. The green VM LED on the BOOSTXL-DRV8323RS should turn on.

Remove the PA26 jumper J18 in the LP-MSPM0G3507 as shown below



Connect USB to the PC for  
software download and debug

## Step2. Software setting

### 2-1. Connect the USB interface of LP-3507 with PC

Note:

Please power on firstly (bus dc power) and then load the program or connect board.

If connection error, please refresh the page and it will automatically reconnect.



Connection success

# Step2. Software setting

## 2-2. Set the correct parameters of the user motor and tuning params.

**MSPM0G-DRV8323RS-EVM-GUI**

**Control**

- Enable Motor
- Disable Closed loop
- Reverse

Outer Loop: Speed

Speed Reference (Hz): 160.00

Id Reference (A): 0.000

Iq Reference (A): 0.000

Startup Method: Align

AlignTime (sec): 0.500

Align Current (A): 2.000

**Rampup**

Rampup current (A): 2.000

Rampup speed rate (Hz/s): 20.00000

Rampup Target (Hz): 80.00

**Tuning**

Speed Kp: 0.083000

Speed Ki: 0.001000

Id Kp: 0.070000

Id Ki: 0.012000

Iq Kp: 0.070000

Iq Ki: 0.012000

**CL**

PI Speed Divider: 5

CL SpeedRef ramp (Hz/s): 20.00000

**Monitors**

VDC (V): 11.728

**Motor & Drive Parameters**

R (ohms): 0.37000

L (H): 0.000220

KE (V/Hz): 0.009867

Max Freq (Hz): 400

PWM Freq (Hz): 20000

Deadband (ns): 400

CSA Gain: 0

**Protection**

Over Current Limit (A): 12

Over Voltage Limit (V): 40

Under Voltage Limit (V): 7

**Fault Status**

CLEAR FAULTS

Ext Fault:  OC:  OV:  UV:

**Speed (Hz)**

**PI tuning regulator paras.**  
It is related to motor, load, and machine structure.

**Monitors**

**I<sub>d</sub> (A)**

**I<sub>q</sub> (A)**

# Step2. Software setting

## 2-3. Set the appropriate control method and reference input.

**MSPM0G-DRV8323RS-EVM-GUI**

**Control**

Enable Motor  Disable Closed loop  Reverse

Outer Loop Speed  Motor State STANDBY

Speed Reference (Hz) 160.00  Speed (Hz) 0.00

Id Reference (A) 0.000  Id Feedback (A) 0.000

Iq Reference (A) 0.000  Iq Feedback (A) 0.000

**Startup** Method Align   
AlignTime (sec) 0.500 Align Current (A) 2.000

**Rampup**  
Rampup current (A) 2.000  Rampup speed rate (Hz/s) 20.00000   
Rampup Target (Hz) 80.00

**Tuning**  
Speed Kp 0.083000  Speed Ki 0.001000   
Id Kp 0.070000  Id Ki 0.012000   
Iq Kp 0.070000  Iq Ki 0.012000

**CL**  
PI Speed Divider 5  CL SpeedRef ramp (Hz/s) 20.00000

**Monitors**

VDC (V) 11.728

**Motor & Drive Parameters**

R (ohms) 0.37000  L (H) 0.000220  KE (V/Hz) 0.009867   
Max Freq (Hz) 400  PWM Freq (Hz) 20000  Deadband (ns) 400   
CSA Gain 0

**Fault Status**

Ext Fault OC OV UV

**Protection**

Over Current Limit (A) 12  Over Voltage Limit (V) 40  Under Voltage Limit (V) 7

**Speed (Hz)**

Speed (Hz)

speed\_PI\_Fbk

speed\_PI\_Ref

**PI tuning regulator paras.**  
It is related to motor, load, and machine structure.

**Monitors**

**I<sub>d</sub> (A)**

I<sub>d</sub> (A)

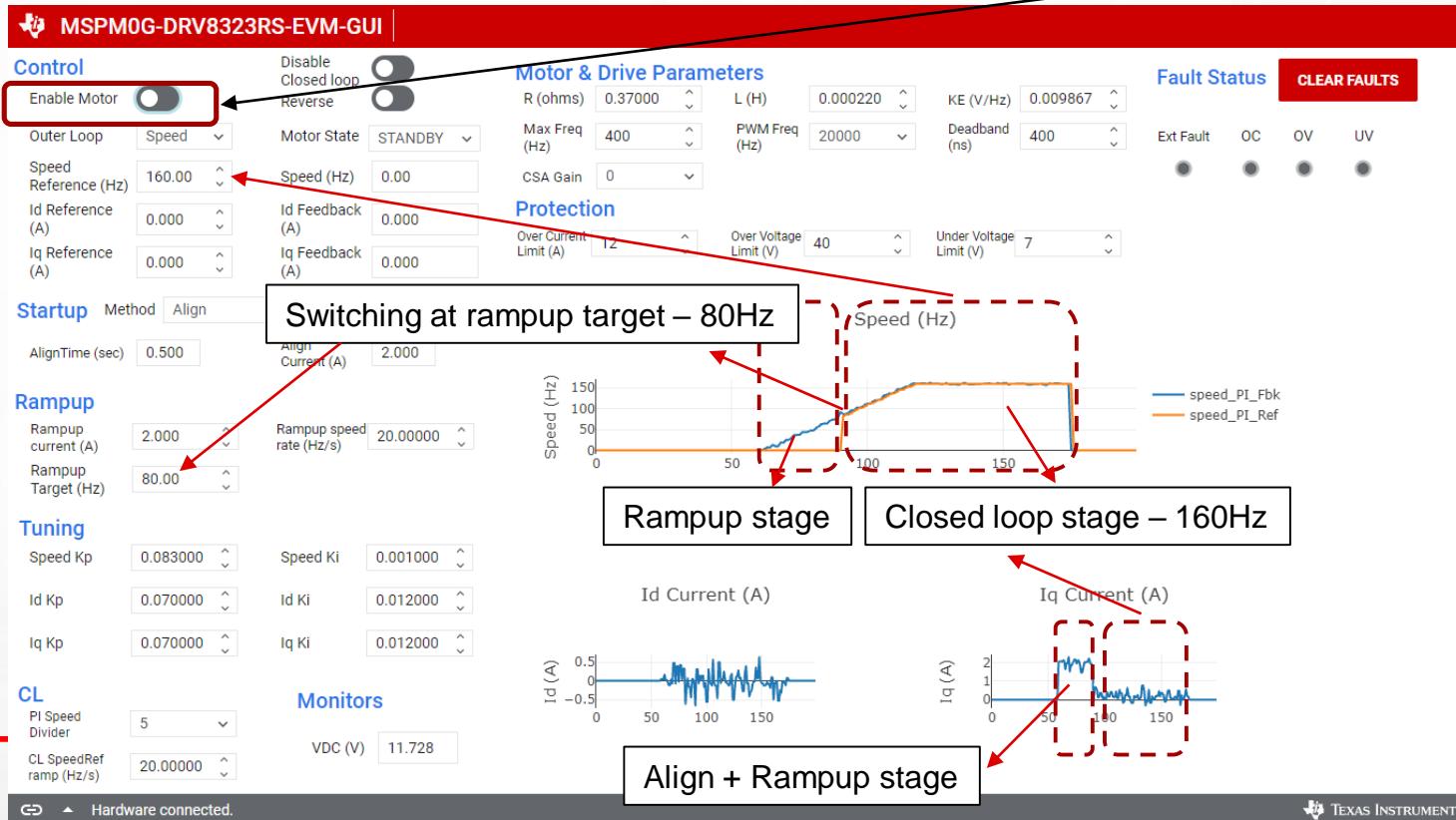
**I<sub>q</sub> (A)**

I<sub>q</sub> (A)

Hardware connected.

# Step3. Spin the motor

Here is the example select closed loop with speed loop, which means output of speed loop is the input of the current loop. Then click the **button** marked below to spin the motor



# Example of the GUI – Video

Control

Enable Motor

Closed loop

Reverse

Outer Loop

Speed

Speed Reference (Hz)

160.00

Id Reference (A)

0.000

Iq Reference (A)

0.000

Motor State

STANDBY

Speed (Hz)

0.00

Id Feedback (A)

0.000

Iq Feedback (A)

0.000

Startup

Method

Align

AlignTime (sec)

0.100

Align Current (A)

2.000

Rampup

Rampup current (A)

2.000

Rampup speed rate (Hz/s)

20.00000

Rampup Target (Hz)

80.00

Tuning

Speed Kp

0.083000

Speed Ki

0.001000

Id Kp

0.070000

Id Ki

0.012000

Iq Kp

0.070000

Iq Ki

0.012000

CL

PI Speed Divider

10

CL SpeedRef ramp (Hz/s)

40.00000

Monitors

VDC (V)

11.882

Speed (Hz)

Speed (Hz)

speed\_PI\_Fbk

speed\_PI\_Ref

Speed (Hz)

0 50 100 150

Id Current (A)

Id Current (A)

Iq Current (A)

Iq Current (A)

0 50 100 150

0 50 100 150

Hardware connected.

Texas INSTRUMENTS

# MSPM0G-DRV8323RS CCS Project

# Environment required

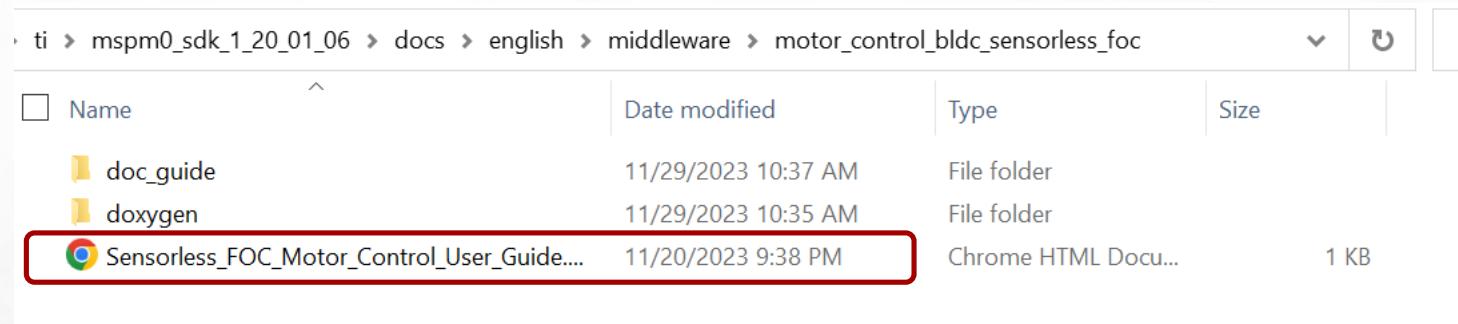
CCS latest version

MSPM0G3507 LaunchPad

SDK latest version

DRV8323RSEVM

## Document for the foc sensorless solution



<input type="checkbox"/> Name	Date modified	Type	Size
doc_guide	11/29/2023 10:37 AM	File folder	
doxygen	11/29/2023 10:35 AM	File folder	
Sensorless_FOC_Motor_Control_User_Guide....	11/20/2023 9:38 PM	Chrome HTML Docu...	1 KB

# MSPM0G-DRV8323RS GUI

Introduction for GUI: Download the SDK latest version and find the file path:

MSPM0 FOC software  
introduction (CCS Project)

MSPM0G-DRV8323RS GUI  
Introduction

MSPM0G3507 LaunchPad and DRV8323RS  
EVM hardware connection introduction

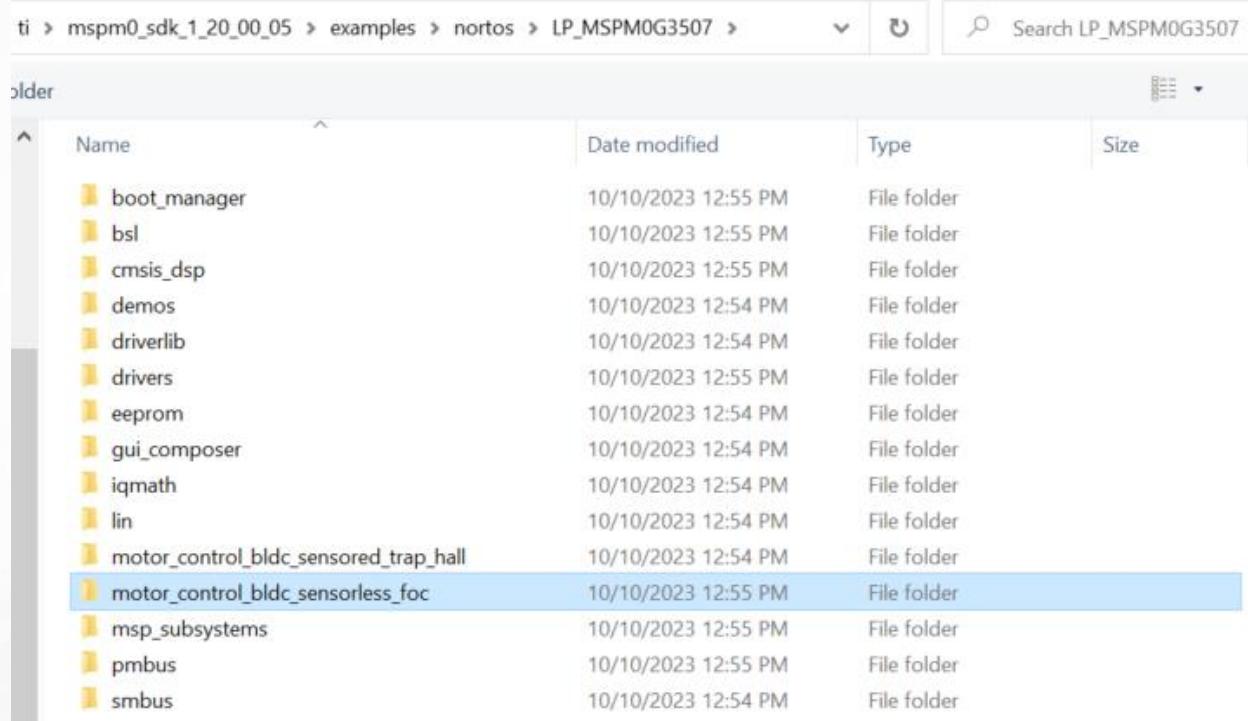


» Sensorless FOC Motor Control User Guide

## Sensorless FOC Motor Control User Guide

- MSPM0 Sensorless FOC Software User's Guide
  - CCS Project setup
  - Adding Initial Parameters (optional)
  - Starting the Project
  - Running the Project
  - 3. API Guide
  - 4. Known Issues
  - 5. Supported Devices
- Sensorless FOC Motor Control Library Overview
  - 1. Software Overview
- DRV8323RS GUI User Guide
  - 1. Overview
  - 1.1 Getting Started with GUI
  - 2. Using the GUI
  - 3. GUI Window
- DRV8323RS Hardware User Guide
  - 1. Hardware Required
  - 2. Hardware Setup

# CCS Project Path



ti > mspm0\_sdk\_1\_20\_00\_05 > examples > nortos > LP\_MSPM0G3507 > ▼ ⟳ 🔍 Search LP\_MSPM0G3507

older

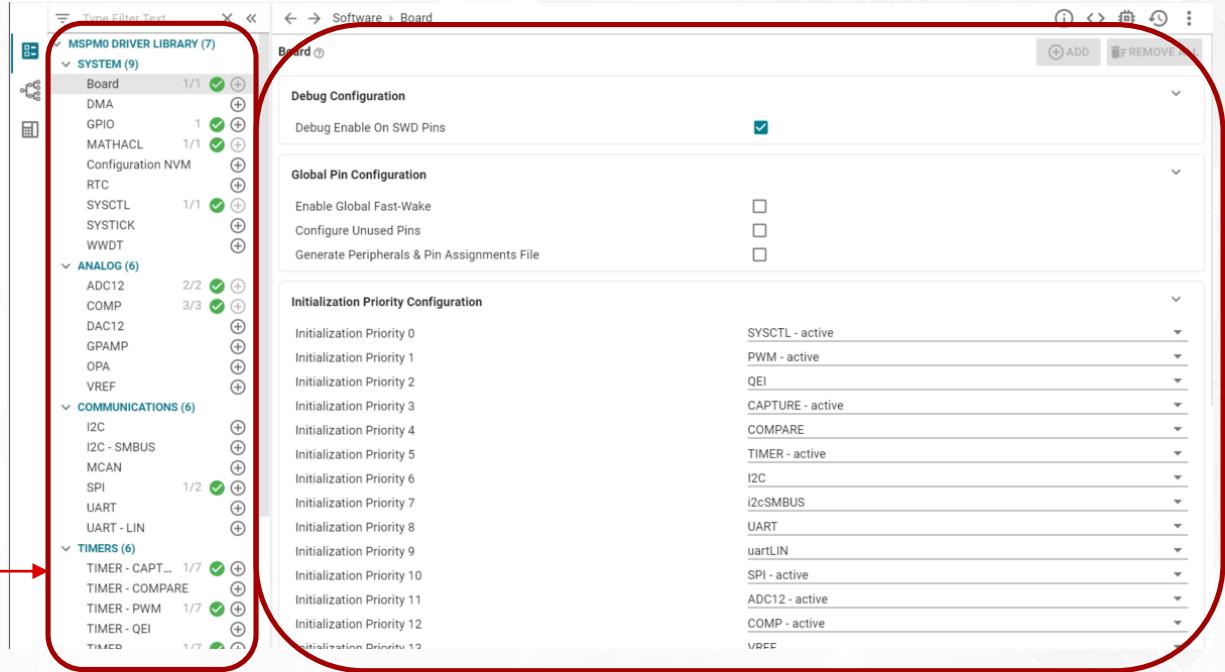
Name	Date modified	Type	Size
boot_manager	10/10/2023 12:55 PM	File folder	
bsl	10/10/2023 12:55 PM	File folder	
cmsis_DSP	10/10/2023 12:55 PM	File folder	
demos	10/10/2023 12:54 PM	File folder	
driverlib	10/10/2023 12:54 PM	File folder	
drivers	10/10/2023 12:55 PM	File folder	
eeprom	10/10/2023 12:54 PM	File folder	
gui_composer	10/10/2023 12:54 PM	File folder	
iqmath	10/10/2023 12:54 PM	File folder	
lin	10/10/2023 12:54 PM	File folder	
motor_control_bldc_sensored_trap_hall	10/10/2023 12:54 PM	File folder	
motor_control_bldc_sensorless_foc	10/10/2023 12:55 PM	File folder	
msp_subsystems	10/10/2023 12:55 PM	File folder	
pmbus	10/10/2023 12:55 PM	File folder	
smbus	10/10/2023 12:54 PM	File folder	

# CCS Project Overall

```
sensorless-foc-demo_LP_MSPM0G3507_nortos_ticlang
  Generated Source
  Includes
  Debug
  modules
    hal
    iqmath_rts
    motor
    motor_driver
    sensorless_foc
  targetConfigs
  ticlang
  drv8323rs-gui.c
  drv8323rs-gui.h
  main.c
  mspm0g3507.cmd
  README.html
  README.md
  sensorless-foc.syscfg
```

Module source code

Used for GUI communication



Sysconfig for peripheral setting  
(automatically generate the code)

# Software structure

Main.c

system initialization

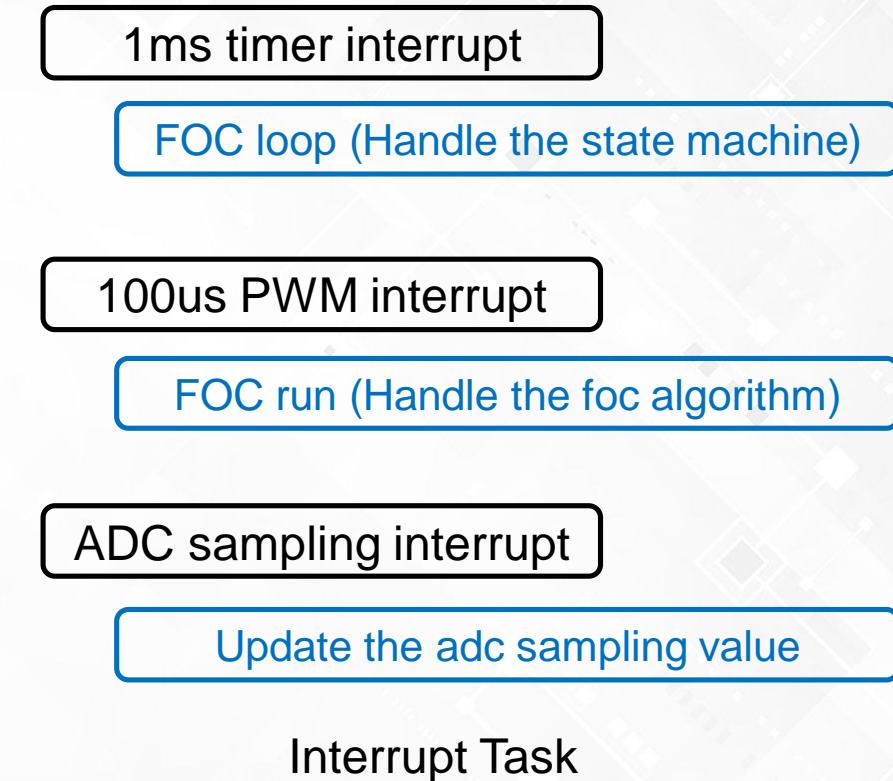


main loop

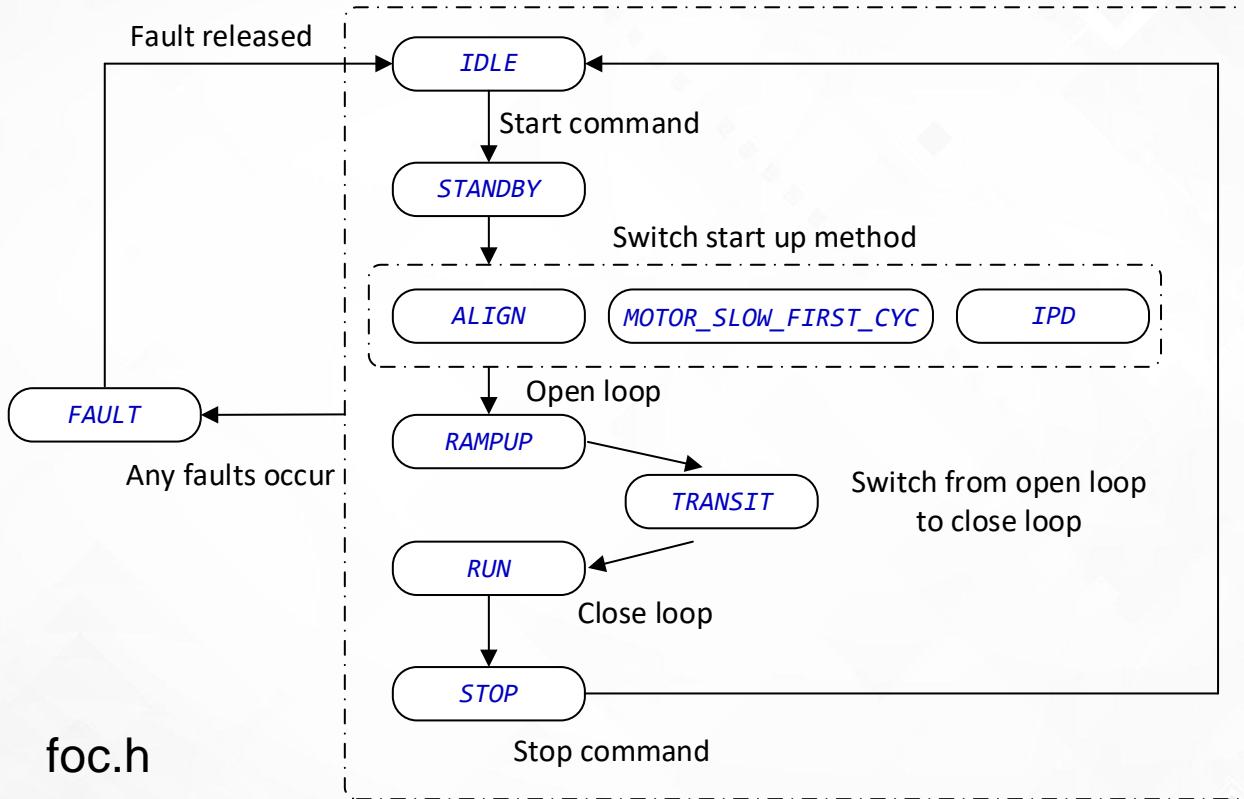
Update FOC control parameters  
Handle the GUI command

Polling Task

|-



# State Machine



FOC sensorless flow:

Three stage start with multi-state machines.

Based on SDK 1.20.01 version

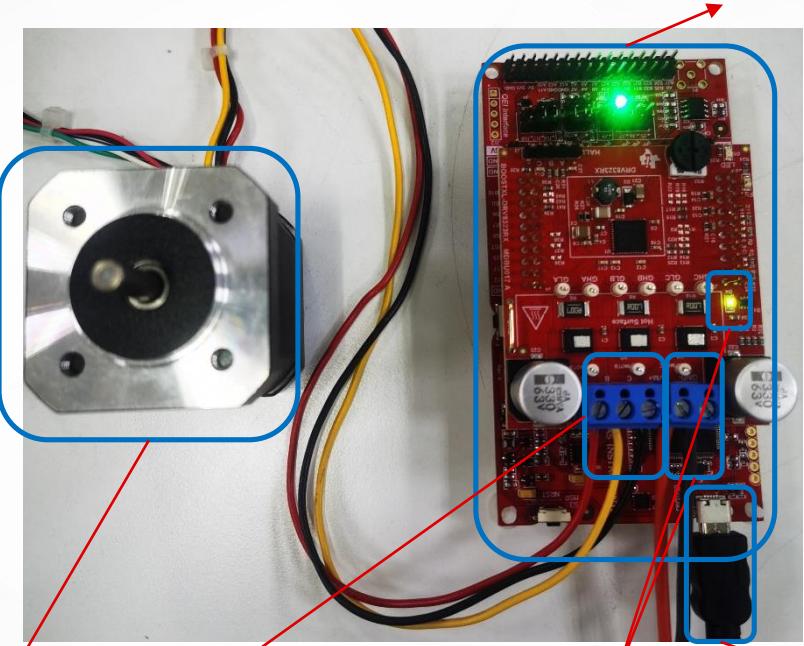
# Step by Step Operations - CCS Project

Note:

The instructions are included in ***Sensorless FOC Motor Control User Guide***

# Hardware Setup

MSPM0G3507 LaunchPad +  
DRV8323RS EVM

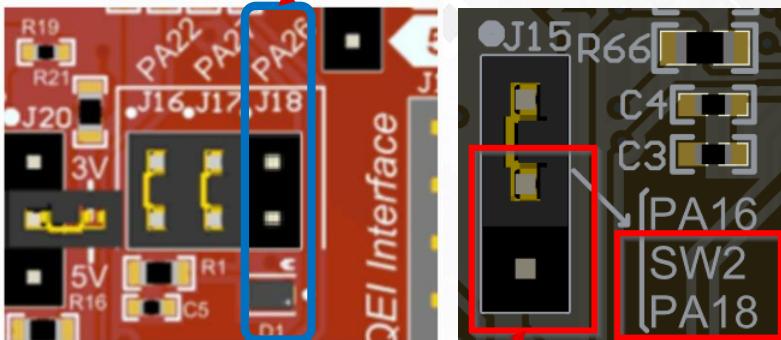


Motor  
Motor Phase U/V/W

Power supply 4.5~34V

Turn on the power supply. The green VM LED on the BOOSTXL-DRV8323RS should turn on.

Remove the PA26 jumper J18 in the LP-MSPM0G3507 as shown below

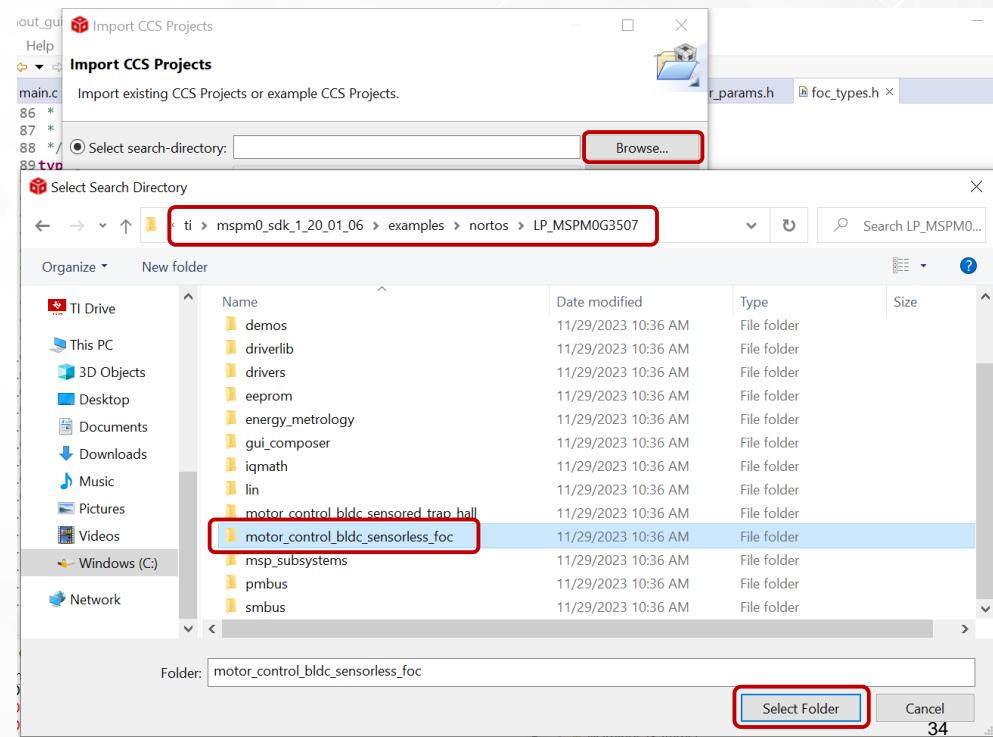
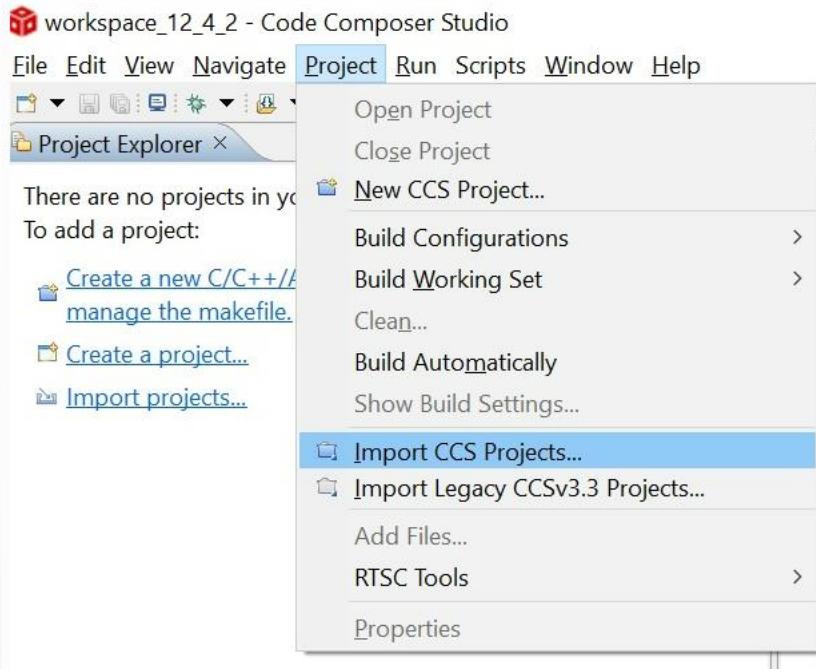


Connect PA18 with SW2 for ISENC  
In Jumper J15

Connect USB to the PC for  
software download and debug

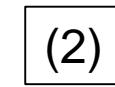
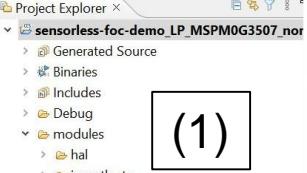
# CCS Project setup

## 1. Import CCS Projects as shown below

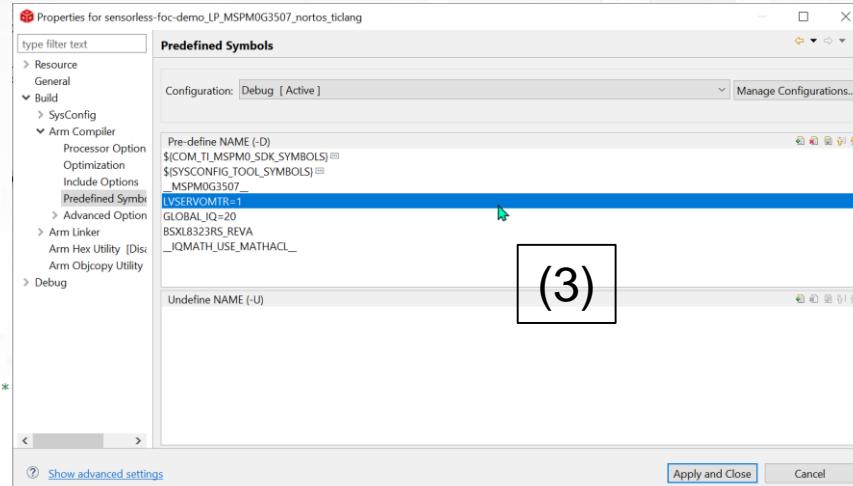


# CCS Project setup

## 2. Setting the Motor Parameters



```
43 ****
44 /* @defgroup MOTOR_MODULE Motor_Module
45 * @{
46 */
47 #ifndef MOTOR_PARAMS_H
48 #define MOTOR_PARAMS_H
49
50 #ifdef __cplusplus
51 extern "C" {
52#endif
53
54#if (LVSERVOMTR)
55/* Define the electrical motor parameters (Teknic 2310 Servomotor) */
56/* @brief Stator resistance (ohm) */
57#define MOTOR_PARA_RS (0.38)
58/* @brief Stator inductance (H) */
59#define MOTOR_PARA_Ls (0.000169)
60/* @brief Number of poles */
61#define MOTOR_PARA_POLES (8)
62/* @brief Back Emf constant in V/Hz */
63#define MOTOR_PARA_KE (0.069)
64/* @brief Maximum Frequency */
65#define MOTOR_PARA_MAX_FREQ (400)
66
```



The motor parameter can be found in the motor\_params.h file. These are the motor parameters required for Sensorless FOC:

- Stator resistance, Rs (ohm)
- Stator inductance, Ls (H)
- Number of motor poles (optional)
- Back Emf constant in V/Hz
- Maximum Frequency of the motor (Hz)

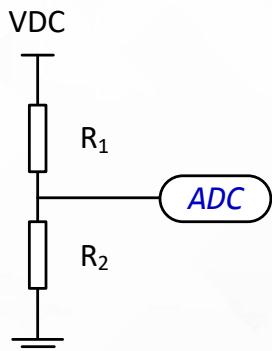
# CCS Project setup

### 3. Setting the circuit parameters (if use DRV8323RS EVM, then skip this step)

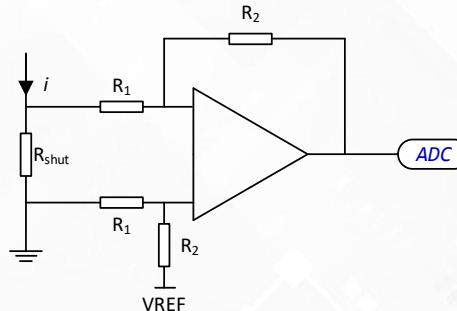


- sensorless-foc-demo\_LP.MSPM0G3507.nortos\_ticlang [Active - Debug]
  - >  Generated Source
  - >  Includes
  - >  Debug
  - >  modules
    - >  hal
    - >  iqmath\_rts
    - >  motor
    - >  motor\_driver
    - >  sensorless\_foc
      - >  estimator
      - >  foc
        - >  foc\_types.h
        - >  foc.h
        - >  transith
- >  ipd
- >  math
- >  parameter
- >  user
- >  targetConfigs
- >  drv8323rs-gui.c
- >  drv8323rs-gui.h
- >  main.c
- >  mspm0g3507.cmd
- >  startup\_mspm0g350x\_ticlang.c
-  README.html
-  README.md
-  sensorless-foc.susfn

```
126 /** @brief Computing voltage scale factor */
127 #define USER_DEFAULT_VOLTAGE_SF (HAL_ADC_DEFAULT_REF_VOLTAGE * \
128                                         (USER_DEFAULT_FOC_VOLT_RATIO))
129
130 /** @brief Computing current scale factor */
131 #define USER_DEFAULT_CURRENT_SF (HAL_ADC_DEFAULT_REF_VOLTAGE / \
132                                         (2 * USER_DEFAULT_DRV_RSHUNT * USER_DEFAULT_BASE_CSA_GAIN))
```



## Voltage scale circuit



## Current scale circuit

# CCS Project setup

## 3. Setting the circuit parameters

```
sensorless-foc-demo_LP.MSPM0G3507_nortos_ticlang [Active - Debug]
  > Generated Source
  > Includes
  > Debug
  > modules
    > hal
    > iqmath.rts
    > motor
    > motor_driver
  > sensorless_foc
    > estimator
    > foc
      > foc_types.h
        > foc.c
        > foc.h
        > transit.h
      > ipd
      > math
      > parameter
      > user
  > targetConfigs
  > drv8323rs-gui.c
  > drv8323rs-gui.h
  > main.c
  > mspm0g3507.cmd
  > startup_mspm0g3507_ticlang.c
  README.html
  README.md
  sensorless-foc.syscfg
```

```
126 /* @brief Computing voltage scale factor */
127 #define USER_DEFAULT_VOLTAGE_SF
128
129
130 /* @brief Computing current scale factor */
131 #define USER_DEFAULT_CURRENT_SF
132
  (HAL_ADC_DEFAULT_REF_VOLTAGE * \
  (USER_DEFAULT_FOC_VOLT_RATIO))

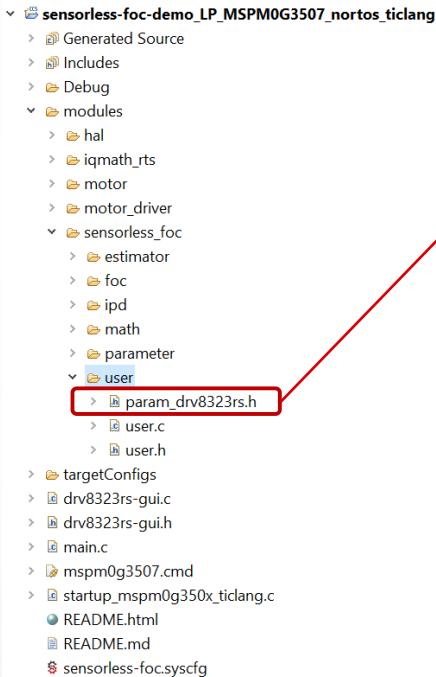
  (HAL_ADC_DEFAULT_REF_VOLTAGE/ \
  (2 * USER_DEFAULT_DRV_RSHUNT * USER_DEFAULT_BASE_CSA_GAIN))
```

The reason that there needs “2” for current scale:

In the current calculation formula, it has divided by “2”, so here gives a compensation.

# CCS Project setup

## 3. Setting the circuit parameters

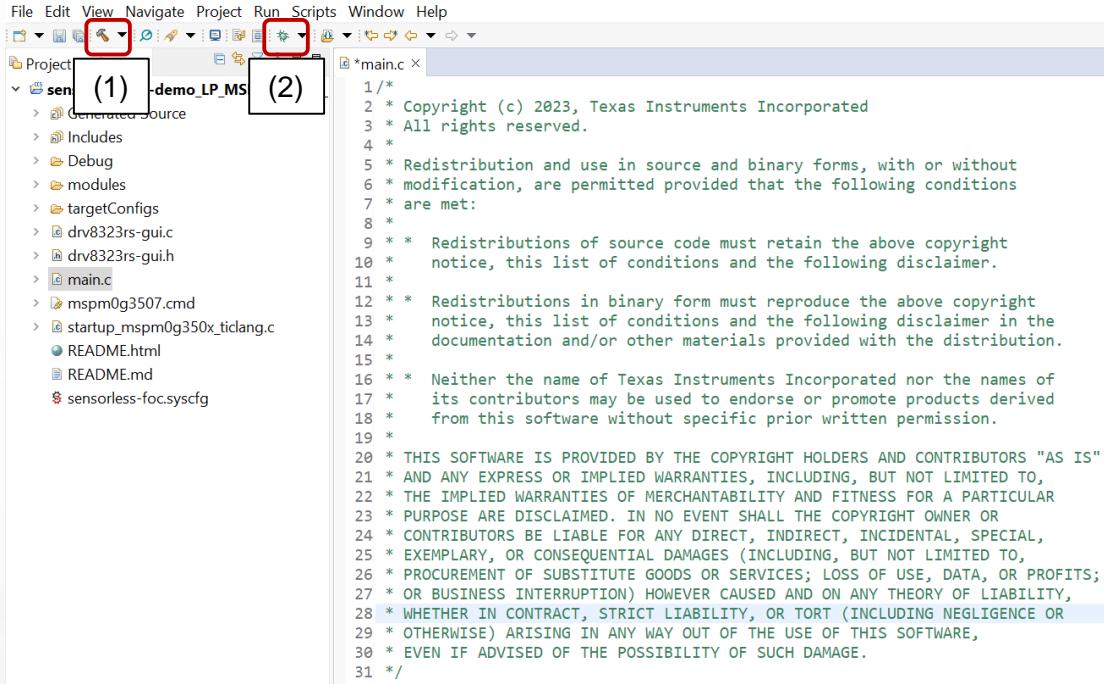


Basically, the foc control useful parameters is included in param\_drv8323rs.h, include the voltage/current scale circuit macro definition params; setting params in each motor control state, ...etc.

Please refers to “**MSPM0 Sensorless FOC Software User’s Guide**” for details instruction, or refers to the note in param\_drv8323rs.h.

# CCS Project run

## 1. Enter debug mode to run the demo



```
File Edit View Navigate Project Run Scripts Window Help
Project (1) demo_LP_MS (2)
*main.c x
1/*
2 * Copyright (c) 2023, Texas Instruments Incorporated
3 * All rights reserved.
4 *
5 * Redistribution and use in source and binary forms, with or without
6 * modification, are permitted provided that the following conditions
7 * are met:
8 *
9 * Redistributions of source code must retain the above copyright
10 * notice, this list of conditions and the following disclaimer.
11 *
12 * Redistributions in binary form must reproduce the above copyright
13 * notice, this list of conditions and the following disclaimer in the
14 * documentation and/or other materials provided with the distribution.
15 *
16 * Neither the name of Texas Instruments Incorporated nor the names of
17 * its contributors may be used to endorse or promote products derived
18 * from this software without specific prior written permission.
19 *
20 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
21 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
22 * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
23 * PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
24 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
25 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
26 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS;
27 * OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
28 * WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
29 * OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
30 * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
```

Connect the hardware and turn on the power supply.

Click on build button.

- Project should build with no errors.

Click on debug button.

# CCS Project run

## 1. Enter debug mode to run the demo

## The code halt position

Expression window, add the expression user want to observe

Switch the edit window  
and debug window

# CCS Project run

## 2. Add the expression for observing

If user want to observe register real-time value, open the Register window

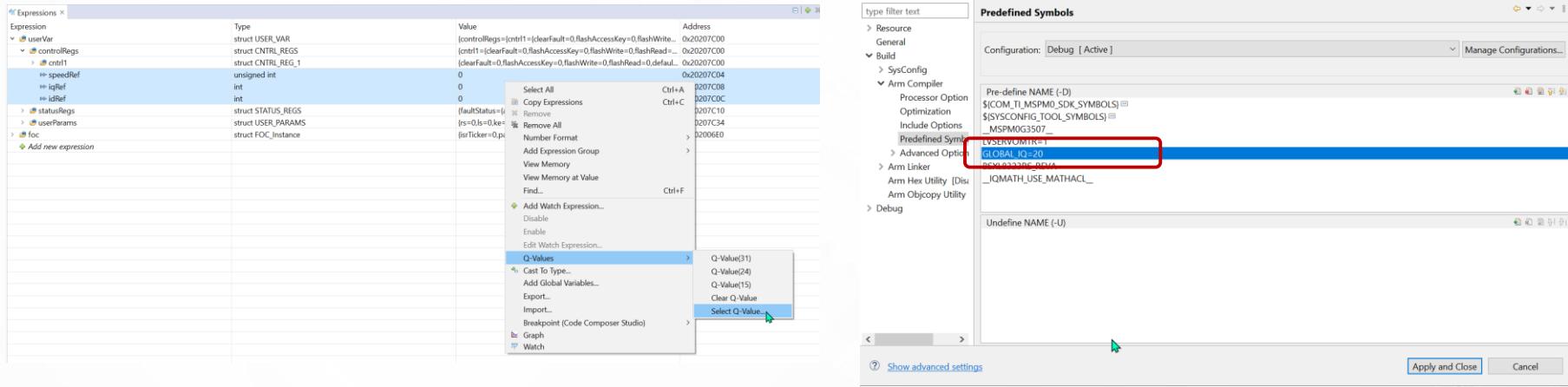
Add expressions:  
userVar;  
Foc;

If user want to observe any other variables, just add the expression here.

### Enable continuous refresh

# CCS Project run

## 2. Add the expression for observing



Some of the variables are in IQ20 format (Global default IQ format).

In order to be displayed and updated properly during debug, you should switch them to IQ(20) format in the CCS expressions window by right clicking the expression, select Q-Values, then click "Select Q-Value".

# CCS Project run

## 3. Running the project

- Start the code 

You should turn on the power firstly and then start, or the current offset detection might be wrong.  
Below is an example of the watch window:

Expression	Type	Value	Address
> userVar	struct USER_VAR		
> foc	struct FOC_Instance		
> isrticker	unsigned int	1894690	0x202006E0
> parameter	unsigned int[34]	[398458,177,72351,400,0,...]	0x202006E4
> state	enum MOTOR	MOTOR_STANDBY	0x2020076C
> faultStatus	union FAULT_STATUS	[all=0,bits={extFaultIn=0,uvl=0,ovlo=0,o...]	0x20200770
> cmd	struct COMMAND	[motorState=MOTOR_STATE_STOP,IqRef=...,	0x20200774
> control	struct CONTROL	[alignn=(cnt=0,alignTime=500,alignCurr=2,...	0x20200790
> vdcAdc	unsigned int	843	0x20200798
> iaAdc	unsigned int	2069	0x202007CC
> ibAdc	unsigned int	2050	0x202007D0
> icAdc	unsigned int	2070	0x202007D4
> offsetCalib	unsigned char	1 '\x01'	0x202007D8
> offsetA	unsigned int	2067	0x202007DC
> offsetB	unsigned int	2054	0x202007E0
> offsetC	unsigned int	2062	0x202007E4
> vdc	int	11.79782772 (Q-Value(20))	0x202007E8
> ia	int	0.1381130219 (Q-Value(20))	0x202007EC
> ib	int	-0.09207630157 (Q-Value(20))	0x202007F0
> overVoltageLimit	int	40.0 (Q-Value(20))	0x202007F4
> underVoltageLimit	int	7.0 (Q-Value(20))	0x202007F8
> overCurrentLimit	int	12.0 (Q-Value(20))	0x202007FC
> PISpdExecDivider	enum FOC_SPD_DIV	FOC_SPD_DIV_10	0x20200800
> PISpdExecCount	unsigned int	0	0x20200804
> enablePWM	unsigned char	0 '\x00'	0x20200808
> enablePWMSatus	unsigned char	0 '\x00'	0x20200809
> csaDiv	unsigned char	0 '\x00'	0x2020080A
> *drv_handle	void *	0x2020080C	0x2020080C
> hal	struct FOC_HAL	{pwmAHal=HAL_PWM_CHANNEL_1,pwm...	0x20200810

Motor status and Fault status

Actual current value converted by adc

Current offset for bipolar current sensing

Detected voltage / current value for motor control

Protection parameters

Note: FOC algorithm only use two phase currents.

# CCS Project run

## 4. Modify the parameters

Expression	Type	Value	Address
userVar	struct USER_VAR	{controlRegs={ctrl1={clearFault=0,flashAccessKey=0,fla...}}	0x20207C00
controlRegs	struct CNTRL_REGS	{ctrl1={clearFault=0,flashAccessKey=0,fla...}}	0x20207C00
statusRegs	struct STATUS_REGS	{faultStatus={all=0,bits={extFaultIn=0,uvlo...}}	0x20207C10
userParams	struct USER_PARAMS	{rs=387973,ls=230,ke=10346,maxFreq=400,PWMFreq=FOC_PWM_FREQ_10000,deadband=400,CSAGain=FOC_CSA_GAIN,outerLoop=FOC_OUTER_LOOP,speedReversal=FOC_DIRECTION, startupMethod=FOC_STARTUP,slowCycFreq=5242880,alignCur=2097152,alignTime=524288,IPDCurrThresh=FOC_IPD_THRESH_COUNT_16,IPDFreq=FOC_IPD_FREQ,rampupCur=2097152,rampupRate=41943040,rampupTarget=83886080,speedRefRampRate=83886080,piSpdKp=104857,piSpdKi=943,piSpdDiv=FOC_SPD_DIV_10,piIqKp=55574,piIqKi=12582,piIdKp=55574,piIdKi=12582,overVoltageLimit=41943040,underVoltageLimit=7340032,overCurrentLimit=12582912,writeFlag=2023406814}	0x20207C34
rs	unsigned int	0.3699998856 (Q-Value(20))	0x20207C40
ls	unsigned int	0.0002193450928 (Q-Value(20))	0x20207C40
ke	unsigned int	0.009866714478 (Q-Value(20))	0x20207C40
maxFreq	unsigned int	400	0x20207C40
PWMFreq	enum FOC_PWM_FREQ	FOC_PWM_FREQ_10000	0x20207C40
deadband	unsigned int	400	0x20207C40
CSAGain	enum FOC_CSA_GAIN	FOC_CSA_GAIN_0	0x20207C40
outerLoop	enum FOC_OUTER_LOOP	FOC_OUTER_LOOP_SPEED	0x20207C40
directionReversal	enum FOC_DIRECTION	FOC_DIRECTION_CW	0x20207C40
startupMethod	enum FOC_STARTUP	FOC_STARTUP_ALIGN	0x20207C40
slowCycFreq	unsigned int	5242880	0x20207C40
alignCur	unsigned int	2097152	0x20207C40
alignTime	unsigned int	524288	0x20207C40
IPDCurrThresh	enum FOC_IPD_THRESH_COUNT	FOC_IPD_THRESH_COUNT_16	0x20207C40
IPDFreq	enum FOC_IPD_FREQ	FOC_IPD_FREQ_4000	0x20207C40
rampupCur	unsigned int	2097152	0x20207C40
rampupRate	unsigned int	41943040	0x20207C40
rampupTarget	unsigned int	83886080	0x20207C40
speedRefRampRate	unsigned int	83886080	0x20207C40
piSpdKp	unsigned int	104857	0x20207C40
piSpdKi	unsigned int	943	0x20207C40
piSpdDiv	enum FOC_SPD_DIV	FOC_SPD_DIV_10	0x20207C40
piIqKp	unsigned int	55574	0x20207C40
piIqKi	unsigned int	12582	0x20207C40
piIdKp	unsigned int	55574	0x20207C40
piIdKi	unsigned int	12582	0x20207C40
overVoltageLimit	unsigned int	41943040	0x20207C40
underVoltageLimit	unsigned int	7340032	0x20207C40
overCurrentLimit	unsigned int	12582912	0x20207C40
writeFlag	unsigned int	2023406814	0x20207C40

User can change the params during the debug mode, select `userVar.userParams`

Motor params

PWM setting

Motor control setting

Start up params. (Align & IPD)

Ramp up params

PI regulator params

Protection params

What GUI has

# CCS Project run

## 4. Modify the parameters

User can change the params during the debug mode, select *userVar.userParams*

All the data listed in *userVar.userParams*, user can set it default value in the corresponding header files and macro definitions.

**CCS Project setup** describe some parameters definition.

If user can't find the definition you want to modify, you can use *userVar.userParams* to change during debug mode. It doesn't require reloading the program or restart the code, modifications will automatically take effect.

If user spin the motor through the GUI firstly, then he can use the same parameters in the GUI here.

# CCS Project run

## 4. Modify the parameters

### Examples:

Expression	Type	Value	Address
userVar	struct USER_VAR		
controlRegs	struct CNTRL_REGS		
statusRegs	struct STATUS_REGS		
userParams	struct USER_PARAMS		
rs	unsigned int	(controlRegs=<ctrl1=<clearFault=0,flashAccessKey=0,flashAccessKeySel=0,fla...	0x20207C00
ls	unsigned int	(ctrl1=<clearFault=0,flashAccessKey=0,flashAccessKeySel=0,fla...	0x20207C00
ke	unsigned int	(faultStatus=<all=0,bits=<extFault=0,uvlo=0,uvlo...	0x20207C10
maxFreq	unsigned int	(rs=387973,ls=230,ke=10346,maxFreq=4...	0x20207C34
PWMfreq	enum FOC_PWM_FREQ	0.3699998856 (Q-Value20)	0x20207C34
deadband	unsigned int	0.0002193450928 (Q-Value20)	0x20207C38
CSAGain	enum FOC_CSA_GAIN	0.009866714478 (Q-Value20)	0x20207C3C
outerLoop	enum FOC_OUTER_LOOP	400	0x20207C40
directionReversal	enum FOC_DIRECTION	FOC_PWM_FREQ_10000	0x20207C44
startupMethod	enum FOC_STARTUP	FOC_PWM_FREQ_10000	0x20207C48
slowCycFreq	unsigned int	FOC_PWM_FREQ_15000	0x20207C4C
alignCur	unsigned int	FOC_PWM_FREQ_20000	0x20207C50
alignTime	unsigned int	FOC_PWM_FREQ_25000	0x20207C54
IPDCurrThresh	enum FOC_IPD_THRESH_COUNT	FOC_PWM_FREQ_30000	0x20207C58
IPDfreq	enum FOC_IPD_FREQ	FOC_PWM_FREQ_35000	0x20207C5C
rampupCur	unsigned int	FOC_PWM_FREQ_40000	0x20207C60
rampupRate	unsigned int	FOC_PWM_FREQ_45000	0x20207C64
rampupTarget	unsigned int	FOC_PWM_FREQ_50000	0x20207C68
speedRefRampRate	unsigned int	FOC_PWM_FREQ_MAX	0x20207C6C
piSpdKp	unsigned int	FOC_PWM_FREQ_RESERVED	0x20207C70
piSpdKi	unsigned int	2097152	0x20207C70
piSpdDiv	unsigned int	41943040	0x20207C74
pilqKp	unsigned int	83886080	0x20207C7C
pilqKi	unsigned int	83886080	0x20207C7C
pildKp	unsigned int	104857	0x20207C80
pildKi	unsigned int	943	0x20207C84
pilqDiv	enum FOC_SPD_DIV	FOC_SPD_DIV_10	0x20207C88
pilqKp	unsigned int	55574	0x20207C8C
pilqKi	unsigned int	12582	0x20207C90
pildKp	unsigned int	55574	0x20207C94
pildKi	unsigned int	12582	0x20207C98
overVoltageLimit	unsigned int	41943040	0x20207C9C
underVoltageLimit	unsigned int	7340032	0x20207CA0
overCurrentLimit	unsigned int	12582912	0x20207CA4
writeFlag	unsigned int	2023406814	0x20207CA8

User can change the params during the debug mode, select `userVar.userParams`

An example to change the PWM frequency.

1. Double click on the value
2. Select the frequency you want

An example to change the rampup target speed

1. Select the correct IQ format for the data (IQ20)
2. Double click on the value
3. Input the value you want by keyboard

# CCS Project run

## 5. Spin the motor – open loop

User can control motor status by `userVar.controlRegs`

Expression	Type	Value	Address
<code>userVar</code>	struct USER_VAR		
<code>controlRegs</code>	struct CNTRL_REGS		0x20207C00
<code>ctrl1</code>	struct CNTRL_REG_1		0x20207C00
<code>clearFault</code>	unsigned int : 1	0	0x20207C00 bit 0
<code>flashAccessKey</code>	unsigned int : 4	0	0x20207C00 bit 1-4
<code>flashWrite</code>	unsigned int : 1	0	0x20207C00 bit 5
<code>flashRead</code>	unsigned int : 1	0	0x20207C00 bit 6
<code>defaultRead</code>	unsigned int : 1	0	0x20207C00 bit 7
<code>disCL</code>	unsigned int : 1	1	0x20207C00 bit 8
<code>enableMotor</code>	unsigned int : 1	1	0x20207C00 bit 9
<code>speedRef</code>	unsigned int	0.0 (Q-Value(20))	0x20207C04
<code>iqRef</code>	int	0.0 (Q-Value(20))	0x20207C08
<code>idRef</code>	int	0.0 (Q-Value(20))	0x20207C0C
<code>statusRegs</code>	struct STATUS_REGS		
<code>faultStatus</code>	union FAULT_STATUS		
<code>motorState</code>	enum MOTOR	MOTOR_RAMPUP	0x20207C10
<code>vdc</code>	int	11.91018772 (Q-Value(20))	0x20207C18
<code>speedFbk</code>	int	79.88662148 (Q-Value(20))	0x20207C1C
<code>speedRef</code>	int	0.0 (Q-Value(20))	0x20207C20
<code>iqCurr</code>	int	1.933246613 (Q-Value(20))	0x20207C24
<code>iqCurrRef</code>	int	2.0 (Q-Value(20))	0x20207C28
<code>idCurr</code>	int	0.1720647812 (Q-Value(20))	0x20207C2C
<code>idCurrRef</code>	int	0.0 (Q-Value(20))	0x20207C30
<code>userParams</code>	struct USER_PARAMS		
<code>rs</code>	float	{rs=387973,ls=230,ke=10346,maxFreq=4...	0x20207C34
<code>isrTicker</code>	unsigned int	387973181	0x202006E0
<code>parameter</code>	unsigned int[34]	[387973,230,10346,400,2...]	0x202006E4
<code>state</code>	enum MOTOR	MOTOR_RAMPUP	0x202006F0
<code>faultStatus</code>	union FAULT_STATUS	{all=0.bits={extFaultIn=0,uvlo=0,ovlo=0,o...	0x20200770
<code>cmd</code>	struct COMMAND	{motorState=MOTOR_STATE_START,IqRef=...	0x20200774
<code>control</code>	struct CONTROL	{align={cnt=500,alignTime=500,alignCurr...	0x20200790
<code>vdcAdc</code>	unsigned int	868	0x202007C8
<code>iaAdc</code>	unsigned int	2104	0x202007CC
<code>ibAdc</code>	unsigned int	1964	0x202007D0

Set `disCL = 1`, and then Set `enableMotor =1`.  
The motor will start to spin and end in open loop at the speed of the ramp up target you set.

User can control open loop speed by `userVar.userParams` ;

or control by  
`foc.control.rampup` ;

<code>foc</code>	struct FOC_Instance	
<code>isrTicker</code>	unsigned int	33144873.parameter=387973...
<code>parameter</code>	unsigned int[34]	33145646
<code>state</code>	enum MOTOR	[387973,230,10346,400,2...]
<code>faultStatus</code>	union FAULT_STATUS	MOTOR_STANDBY
<code>cmd</code>	struct COMMAND	{all=0.bits={extFaultIn=0,uvlo=0,ovlo=0,o...
<code>control</code>	struct CONTROL	{motorState=MOTOR_STATE_STOP,IqRef=...
<code>align</code>	struct ALIGN_Instance	{align={cnt=0,alignTime=500,alignCurr=2...
<code>rampup</code>	struct RAMPUP_Instance	{rampup={rampupTime=2097152,rampupRate=41920,target...
<code>rampupCurr</code>	int	2.0 (Q-Value(20))
<code>rate</code>	int	0.03997802734 (Q-Value(20))
<code>target</code>	int	80.0 (Q-Value(20))
<code>rampOut</code>	int	0

# CCS Project run

## 5. Spin the motor – close loop

User can control motor status by `userVar.controlRegs`

Expression	Type	Value	Address
userVar	struct USER_VAR	{controlRegs={ctrl1={clearFault=0,flashAc... 0x20207C00	
controlRegs	struct CNTRL_REGS	{ctrl1={clearFault=0,flashAccessKey=0,fla... 0x20207C00	
ctrl1	struct CNTRL_REG_1	{clearFault=0,flashAccessKey=0,flashWrite... 0x20207C00	
↳ clearFault	unsigned int : 1	0	0x20207C00 bit 0
↳ flashAccessKey	unsigned int : 4	0	0x20207C00 bit 1-4
↳ flashWrite	unsigned int : 1	0	0x20207C00 bit 5
↳ flashRead	unsigned int : 1	0	0x20207C00 bit 6
↳ defaultRead	unsigned int : 1	0	0x20207C00 bit 7
↳ disCL	unsigned int : 1	0	0x20207C00 bit 8
↳ enableMotor	unsigned int : 1	1	0x20207C00 bit 9
↳ speedRef	unsigned int	160.0 (Q-Value(20))	0x20207C04
↳ iqRef	int	0.0 (Q-Value(20))	0x20207C08
↳ idRef	int	0.0 (Q-Value(20))	0x20207C0C
statusRegs	struct STATUS_REGS	{faultStatus={all=0,bits={extFaultIn=0,uvlo... 0x20207C10	
↳ faultStatus	union FAULT_STATUS	{all=0,bits={extFaultIn=0,uvlo=0,ovlo=0,... 0x20207C10	
↳ motorState	enum MOTOR	MOTOR_RUN	0x20207C14
↳ vdc	int	11.96636772 (Q-Value(20))	0x20207C18
↳ speedFbk	int	159.7398071 (Q-Value(20))	0x20207C1C
↳ speedRef	int	159.9963379 (Q-Value(20))	0x20207C20
↳ iqCurr	int	0.06970119476 (Q-Value(20))	0x20207C24
↳ iqCurrRef	int	0.2961921692 (Q-Value(20))	0x20207C28
↳ idCurr	int	0.03873348236 (Q-Value(20))	0x20207C2C
↳ idCurrRef	int	0.0 (Q-Value(20))	0x20207C30
userParams	struct USER_PARAMS	{rs=387973,ls=230,ke=10346,maxFreq=4... 0x20207C34	
↳ foc	struct FOC_Instance	{srTicker=347764217,parameter={387973,... 0x202006E0	
↳ isrTicker	unsigned int	34765187	0x202006E0
↳ parameter	unsigned int[34]	387973,230,10346,400,2,...	0x202006E4
↳ state	enum MOTOR	MOTOR_RUN	0x2020076C
faultStatus	union FAULT_STATUS	{all=0,bits={extFaultIn=0,uvlo=0,ovlo=0,... 0x20200770	
cmd	struct COMMAND	{motorState=MOTOR_STATE_START,IqRef... 0x20200774	
control	struct CONTROL	{align={cnt=500,alignTime=500,alignCurr... 0x20200790	
↳ vdcAdc	unsigned int	846	0x202007C8
↳ iaAdc	unsigned int	2060	0x202007CC
↳ ibAdc	unsigned int	2064	0x202007D0

Set disCL = 0, set speedRef as you want, and then Set enableMotor =1.

The motor will start to spin and end in close loop at the speed of the speedRef you set.

User can change close loop speed by changing the speedRef.