life.augmented

UM1080
User manual

Quick start guide for STM32F PMSM single/dual FOC SDK v4.2

Introduction

This user manual provides information to facilitate the use and customization of the STM32 PMSM Field Oriented Control (FOC) SDK.

A complete documentation list is provided in *Section 2: Documentation architecture*. It is included in the software package (STSW-STM32100) and it is available on the ST web site.

Section 3: Motor profiler and One Touch Tuning explains how to enable and use the “Motor profiler” and the “One touch tuning” feature to startup an unknown motor from the scratch.

Section 4: On-the-fly Sensorless startup explains how to enable and use the “On-the-fly” Sensorless startup feature.

Section 5: Working environment and its customization explains the Motor Control workspace, its customization and download.

Section 6: How to download the full LCD user interface explains how to download a Graphical User Interface that allows run-time command execution and fine tune system parameters (note that this procedure has to be done just once on new evaluation boards) in the microcontroller Flash memory to STM32 evaluation boards fitted with an LCD display.

Section 7: Full LCD user interface explores the menu screens and controls.

Section 8: Introduction to the PMSM FOC drive provides the block diagram of the implemented Field Oriented Control.

Section 9: Current sensing and protection with embedded analog (STM32F3x) and *Section 10: Overvoltage protection with embedded analog (STM32F3x)* explain how the library can be easily configured to use STM32F30x's embedded analog peripheral set (fast comparators and Programmable Gain Amplifiers (PGA)).

Release note RN0085 lists all supported microcontrollers.

Contents

List of tables

List of figures

1 Motor control library features

- Motor profiler:
  - a new algorithm able to auto-measure electromechanical Parameters of PMSM Motors (only for STM32F30x and STM32F4xx).
- One touch tuning:
  - is a new algorithm that use a single parameter to set-up the speed controller according to the type of load. Together with the Motor profiler can be enabled to achieve the setup and run of an unknown motor from the scratch (only the STM32F30x and STM32F4xx).
- On-the-fly sensorless startup, a new algorithm able to detect if the motor is running before the startup and skip the acceleration phase if not necessary. The motor is run in FOC from the begin without need to stop it before the start. This feature is particular useful for fan application (any STM32F supported).
- Single or simultaneous Dual PMSM FOC
  - sensorless/sensored (Dual PMSM FOC only when running on STM32F103xx High-Density, STM32F103xx XL-Density, STM32F2xx, STM32F303xB/C or STM32F4xx)
- Speed feedbacks:
  - Sensorless (High Frequency Injection HFI plus B-EMF State Observer, PLL rotor speed/angle computation from B-EMF, only for STM32F30x or STM32F4xx);
  - Sensorless (B-EMF State Observer, PLL rotor speed/angle computation from B-EMF);
  - Sensorless (B-EMF State Observer, CORDIC rotor angle computation from B-EMF);
  - 60° or 120° displaced Hall sensors decoding, rising/falling edge responsiveness;
  - Quadrature incremental encoder;
  - For each motor, dual simultaneous speed feedback processing;
  - On-the-fly speed sensor switching capability;
- Current sampling methods:
  - Two ICS (only when running on STM32F103xx, STM32F2xx, or STM32F4xx);
  - Single, common DC-link shunt resistor (ST patented);
  - Three shunt resistors placed on the bottom of the three inverter legs (only when running on STM32F103xx, STM32F2xx, STM32F302xB/C, STM32F303xB/C or STM32F4xx);
- Embedded analog (STM32F30x only):
  - PGA (Programmable Gain Amplifiers) for current sensing: support for three-shunt and single shunt, internal and external gain;
  - Comparators for overcurrent protection: support for three-shunt and single shunt, internal and external threshold;
  - Comparators for overvoltage protection: support for motor phases short-circuiting mode and free-wheeling mode, internal and external threshold;
- FOC hardware acceleration (STM32F30x only);
  - ADC queue of context (ST patented architecture) support;
  - CCM (Core Coupled Memory) RAM support;

  - Advanced Timer structures for single shunt (ST patented) support;
- Flux weakening algorithm to attain higher than rated motor speed (optional);
- Feed-Forward, high performance current regulation algorithm (optional);
- SVPWM generation:
  - Centered PWM pattern type;
  - Adjustable PWM frequency;
- Torque control mode, speed control mode; on-the-fly switching capability;
- Brake strategies (optional):
  - Dissipative DC link brake resistor handling;
  - Motor phases short-circuiting (with optional hardware over-current protection disabling);
  - motor phases free-wheeling;
- When running Dual FOC, any combination of the above-mentioned speed feedback, current sampling, control mode, optional algorithm;
- Optimized I-PMSM and SM-PMSM drive;
- Programmable speed ramps (parameters duration and final target);
- Programmable torque ramps (parameters duration and final target);
- Real-time fine tuning of:
  - PID regulators;
  - Sensorless algorithm;
  - Flux weakening algorithm;
  - Start-up procedure (in case of sensorless);
- Fault conditions management:
  - Over-current;
  - Over-voltage;
  - Over-temperature;
  - Speed feedback reliability error;
  - FOC algorithm execution overrun;
- Easy customization of options, pin-out assignments, CPU clock frequency through ST MC Workbench GUI;
- C language code:
  - Compliant with MISRA-C 2004 rules;
  - Conforms strictly with ISO/ANSI;
  - Object-oriented programming architecture;

1.1 User project and interface features

There are two available options:

- FreeRTOS-based user project (for STM32F103xx and STM32F2xx only);
- SysTick-timer-easy-scheduler-based user project;

Available User Interface options (and combinations of them):

- Full LCD plus joystick;
- Light LCD plus joystick;
- Serial communication protocol bidirectional (compatible with ST MC Workbench GUI);
- Serial communication protocol fast unidirectional;
- Drive system variables logging/displaying via:
  – SPI;
  – DAC (DAC peripheral is not present in the STM32F103xx low or medium density; in this case, RC-filtered PWM signal option is available);

2 Documentation architecture

2.1 Where to find the information you need

Technical information about the MC SDK is distinguished and organized by topic. The following is a list of the documents that are available and the subjects they cover:

- STM32F PMSM single/dual FOC SDK (UM1052) provides the followings:
  - Features
  - Architecture
  - Workspace
  - Customization processes
  - Overview of algorithms implemented (FOC, current sensors, speed sensors, embedded analog topologies supported)
  - MC API
  - Demonstrative user project
  - Demonstrative LCD user interface
  - Demonstrative serial communication protocol
- Advanced developers guide for STM32F MCUs PMSM single/dual FOC library (UM1053). This provides the followings:
  - Object oriented programming style used for developing the MC library
  - Description of classes that belong to the MC library
  - Interactions between classes
  - Description of tasks of the MCA
- MC library source documentation (Doxygen compiled HTLM file). This provides a full description of the public interface of each class of the MC library (methods, parameters required for object creation).
- MC Application source documentation (Doxygen compiled HTML file). This provides a full description of the classes that make up the MC API.
- User Interface source documentation (Doxygen compiled HTML file). This provides a full description of the classes that make up the UI Library.
- STM32F0xx, STM32F10xx, STM32F2xx, STM32F30x or STM32F4xx Standard Peripherals Library source documentation (Doxygen compiled HTML file).
- ST MC Workbench GUI documentation. This is a field guide that describes the steps and parameters required to customize the library, as shown in the GUI.
- In-depth documentation about particular algorithms (sensorless position/speed detection, flux weakening, MTPA, feed-forward current regulation).

Please contact your nearest ST sales office or support team to obtain the documentation you are interested in if it was not already included in the software package you received or available on the ST web site (*www.st.com*).

2.2 Related documents

Available from www.arm.com

- Cortex®-M0 Technical Reference Manual, available from: http://infocenter.arm.com.
- Cortex®-M3 Technical Reference Manual, available from: *http://infocenter.arm.com.*
- Cortex®-M4 Technical Reference Manual, available from:http://infocenter.arm.com.

Available from *www.st.com* or your STMicroelectronics sales office

- STM32F051x datasheets
- STM32F100xx datasheet
- STM32F103xx datasheet
- STM32F20x and STM32F21x datasheets
- STM32F302x6/8 datasheet
- STM32F302xB/C datasheet
- STM32F303xB/C datasheet
- STM32F40x and STM32F41x datasheets
- STM32F051x user manual (RM0091)
- STM32F100xx user manual (RM0041)
- STM32F103xx user manual (RM0008)
- STM32F20x and STM32F21x user manual (RM0033)
- STM32F30x user manual (RM0316)
- STM32F40x and STM32F41x user manual (RM0090)
- STM32F103xx AC induction motor IFOC software library V2.0 (UM0483)
- STM32 and STM8 Flash Loader demonstrator (UM0462)

3 Motor profiler and One Touch Tuning

The “Motor profiler” (also called “Self-commissioning”) is a new algorithm able to auto-measure the electrical parameters of PMSM motors.

The “One touch tuning” is a new algorithm that use a single parameter to set-up the speed controller according the type of load.

They can be enabled together to achieve the run of an unknown motor from the scratch in few minutes.

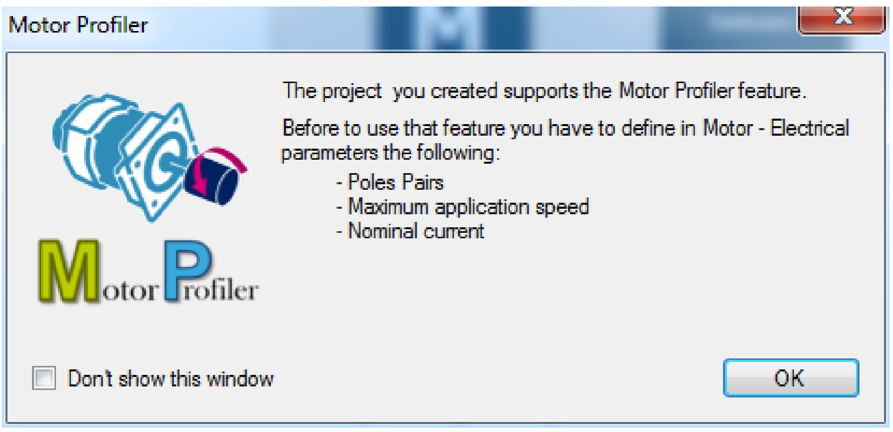
If enabled in the firmware this algorithms runs a set of electrical tests to determine the parameters required by the FOC and perform the auto tuning of the PI regulators (both current and speed). Starting from now the reference of both features is “Motor Profiler”.

The “Motor Profiler” algorithm will determine the following parameters:

- Stator resistance Rs
- Stator inductance Ls
- BEMF constant Ke
- KP and KI of speed controller
- Nominal speed of the motor

If the project supports the “Motor Profiler” feature, the window dialog shown in *Figure 1* will appear.

Figure 1. Motor profiler



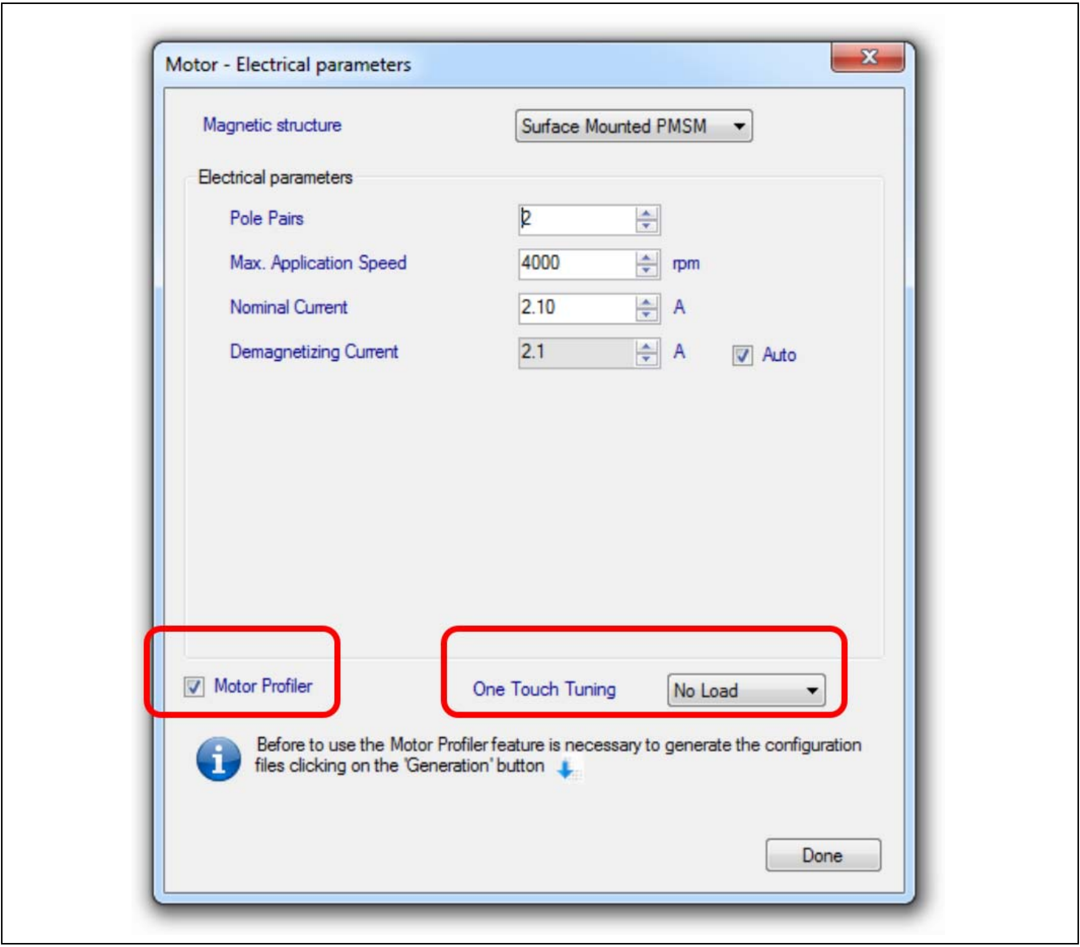
The project you created supports the Motor Profiler feature.

Before to use that feature you have to define in Motor - Electrical parameters the following:

- Poles Pairs
- Maximum application speed
- Nominal current

User can enable this functionality in any new Workbench project by checking the “Motor profiler” check box in the Motor – Electrical parameters dialog, like shown in *Figure 2*. When enabling “Motor profiler”, also enable the “One touch tuning” feature.

Figure 2. How to enable Motor Profiler



Motor - Electrical parameters

Magnetic structure: Surface Mounted PMSM

Electrical parameters

Pole Pairs	2		
Max. Application Speed	4000	rpm	
Nominal Current	2.10	A	
Demagnetizing Current	2.1	A	☑ Auto

☑ Motor Profiler

One Touch Tuning: No Load

Before to use the Motor Profiler feature is necessary to generate the configuration files clicking on the 'Generation' button

Enabling "Motor profiler", the electrical parameters required by the FOC (Rs, Ls and Ke) will measured by the FW and the relative edit box disappear from the Motor – Electrical parameters dialog. Other parameters like pole pairs, maximum application speed and nominal current, will not be measured and must be insert by the user. In case of Internal PMSM is possible to change the magnetic structure setting and insert manually the Ld/Lq ratio.

To setup the "One touch tuning" is necessary to indicate the kind of load connected to the motor in a qualitative way:

- No load (small/medium sized motor without any load)
- Medium load (medium sized motor connected with load like pump of small fan)
- Big load (medium/high sized motor connected with full load of big fan)

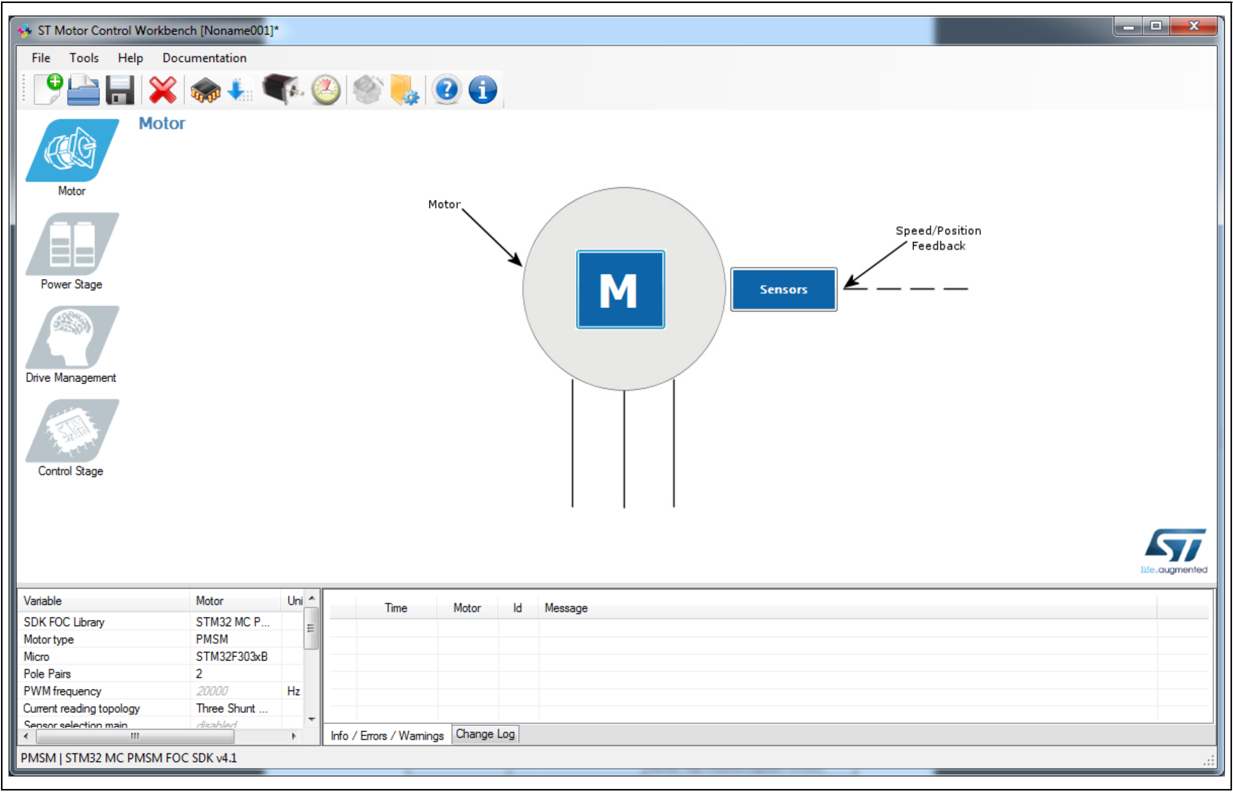
This can be selected with the drop down menu "One touch tuning" in Motor – Electrical parameters dialog, like shown in *Figure 2*.

In this case, the pre-computed PI coefficient of current and speed regulators done by ST MC Workbench will not be used to drive the motor but they will be computed by the "Motor profiler" algorithm.

With the "Motor profiler" feature enabled and before to run it is necessary, as usual, to generate the .h files into the "SystemDriveParams" folder, compile and download the executable into the microcontroller using a supported IDE (like IAR Embedded Workbench or Keil Microvision) as explained in the UM1052 – Chapter 9 – “Working environment”.

When the firmware is compiled and flashed into the micro is possible to use the ST MC Workbench real-time communication to send a “Motor profiler” command. To do this is necessary to Connect the control board with PC using RS232 (COM) null modem cable and press the Motor profiler button as shown in *Figure 3* starts the procedure.

Figure 3. Send a “Motor profiler” command



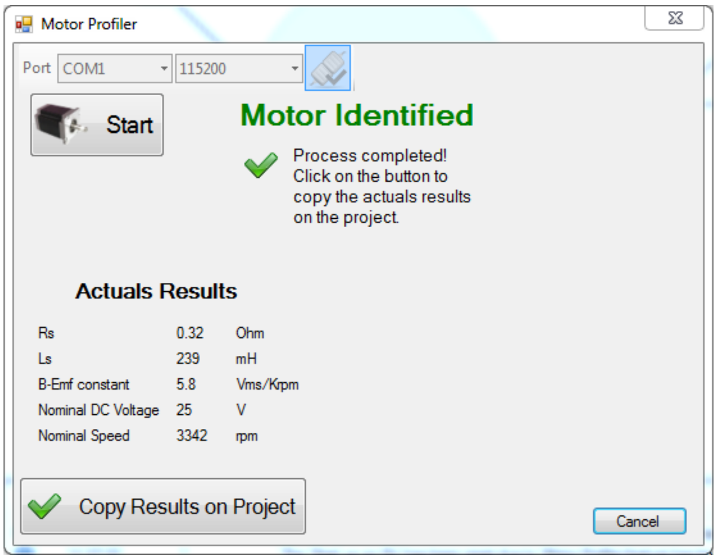
Variable	Motor	Uni
SDK FOC Library	STM32 MC P...	
Motor type	PMSM	
Micro	STM32F303xB	
Pole Pairs	2	
PWM frequency	20000	Hz
Current reading topology	Three Shunt ...	
Sensor selection main	disabled	

PMSM | STM32 MC PMSM FOC SDK v4.1

A progress bar will show the status of the procedure and at the end the motor runs and, a dialog shows the measured parameters like in *Figure 4*.

If some errors occur during the procedure, the fault indication will be shown and the “Clear Fault” button can be pressed to reset the fault state.

Figure 4. Dialog showing the measured parameter



Motor Identified

Process completed!
Click on the button to
copy the actuals results
on the project.

Actuals Results

Rs	0.32	Ohm
Ls	239	mH
B-Emf constant	5.8	Vms/Krpm
Nominal DC Voltage	25	V
Nominal Speed	3342	rpm

At the end of the procedure is also possible to import the measured parameter in the Workbench project pressing the “Copy Results on Project” button shown in *Figure 4*. Doing this the “Motor profiler” feature will be disabled and the firmware can be finalized removing the extra code required by that feature.

The process to generate the “.h” file, compile, download and to finalize the firmware is explained in the UM1052 – Chapter 9 – “Working environment”.

Note: When Motor Profiler is enabled is possible to run the motor in standalone mode (no Workbench connection) using the LCD and Joystick feature (if present in the board). Pressing the Key button the Motor Profiler procedure will be executed before to start the motor. After one successfully identification of the motor the next startup will be executed using the parameters already measured without executing new Motor Profiler identification.

3.1 Restrictions

“Motor profiler” can be enabled only:

- For new Workbench projects (or WB example projects) if the selected power board support that feature.
- If a microcontroller of the STM32F3 or STM32F4 family is used in the WB project.
- In a single drive WB project.
- Using three shunts or single shunt current regulation.

When enabling “Motor profiler” is not possible to:

- enable the Flux Weakening
- enable the “On-the-fly” startup
- define a customized startup sequence
- use “Embedded PGA”

4 On-the-fly Sensorless startup

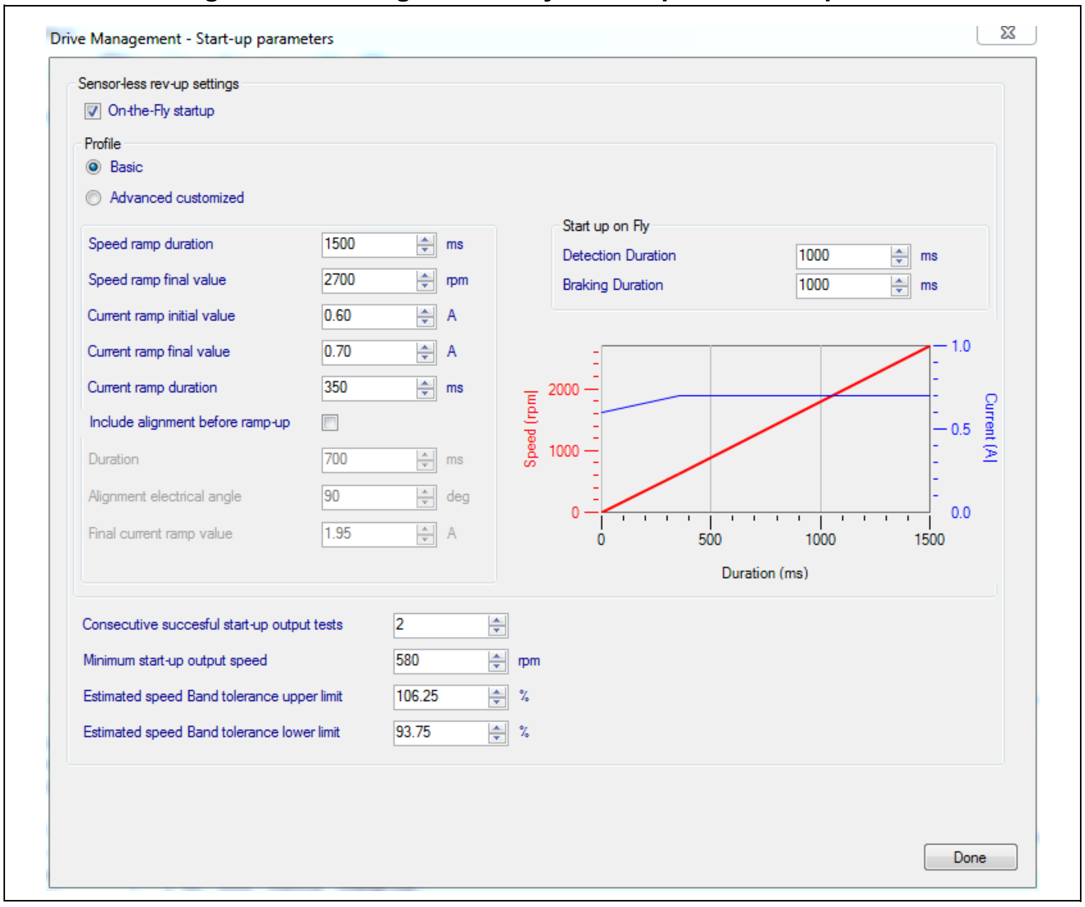
The “On-the-fly” sensorless startup is a new algorithm able to detect if the motor is running before the startup and skip the acceleration phase if not necessary. If the motor runs with a speed that is above the allowed threshold, the firmware apply the FOC from the beginning, without need to stop it and re-start.

This feature is particular useful for fan application.

It is possible to enable this feature only when Sensor-less (Observer+PLL) or Sensor-less (Observer+Cordic) is selected in the Drive Management – Speed Position Feedback Management dialog.

To enable this feature check the “startup on Fly” check box in the Drive management. Start-up parameters dialog like shown in *Figure 5* and *Figure 6*.

Figure 5. Enabling “On-the-fly” start-up with Basic profile



[✓] On-the-Fly startup

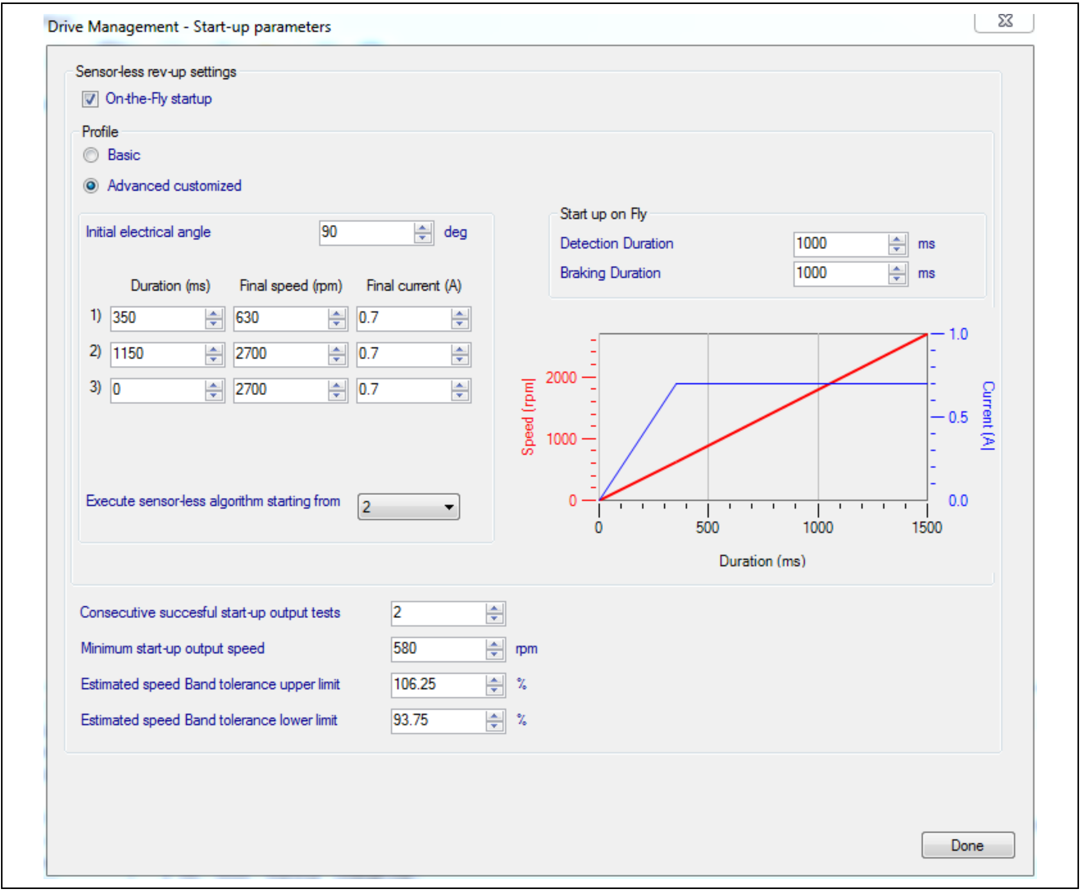
[● Basic] ○ Advanced customized

Speed ramp duration	1500	ms
Speed ramp final value	2700	rpm
Current ramp initial value	0.60	A
Current ramp final value	0.70	A
Current ramp duration	350	ms
Include alignment before ramp-up	☐	
Duration	700	ms
Alignment electrical angle	90	deg
Final current ramp value	1.95	A

Detection Duration	1000	ms
Braking Duration	1000	ms

Consecutive succesful start-up output tests	2	
Minimum start-up output speed	580	rpm
Estimated speed Band tolerance upper limit	106.25	%
Estimated speed Band tolerance lower limit	93.75	%

Figure 6. Enabling “On-the-fly” start-up with Advanced profile



Drive Management - Start-up parameters

Sensor-less rev-up settings

[✓] On-the-Fly startup

Profile

( ) Basic

(●) Advanced customized

Initial electrical angle: 90 deg

	Duration (ms)	Final speed (rpm)	Final current (A)
1)	350	630	0.7
2)	1150	2700	0.7
3)	0	2700	0.7

Execute sensor-less algorithm starting from: 2

Start up on Fly

Detection Duration	1000	ms
Braking Duration	1000	ms

Consecutive succesful start-up output tests	2	
Minimum start-up output speed	580	rpm
Estimated speed Band tolerance upper limit	106.25	%
Estimated speed Band tolerance lower limit	93.75	%

The speed threshold used to determine if is possible to skip the acceleration phase is the “Minimum startup-output speed” that represents the minimum speed for which the sensorless observer gives a reliable measurements. This can be select by the user according the nominal speed of the motor.

When enabling the “On-the-fly” startup two other parameter will appear in the dialog box:

- Detection duration
- Braking Duration

Both quantity are expressed in milliseconds and represents respectively:

- The duration of the “detection phase” of the OTF startup. Within this duration the reliability of the sensorless measurements are tested in order to validate the speed and run directly in FOC.
- The duration of the “braking phase” that is applied if the sensorless measurements doesn’t give reliable measurement during the “detection phase”. During the “braking phase” the motor is brake in order to stop it before the new acceleration.

Both, “Basic startup profile” and “Advanced startup profile”, can be used if the OTF startup is enabled. The two extra phases “detection phase” and “braking phase” are common of the two profiles. The startup profile can be set by the user to define the acceleration strategy if the speed of the motor is below the reliability threshold during the “detection phase”.

5 Working environment and its customization

The working environment for the Motor Control SDK is composed of:

- A PC
- A third-party integrated development environment (IDE)
- A third-party C-compiler
- A JTAG/SWD interface for debugging and programming
- An Application board with an STM32F0xx, STM32F100xx, STM32F103xx, STM32F2xx, STM32F30x or STM32F4xx properly designed to drive its power stage (PWM outputs to gate driver, ADC channels to read currents, DC bus voltage). Many evaluation boards are available from ST, some of them have an ST-link programmer onboard.
- A Three-phase PMSM motor

5.1 Motor control workspace

The Motor Control workspace is composed of two projects (as shown in *Figure 11*), which constitute the MC workspace.

Motor Control Library project: the collection of all the classes developed to implement all the features. It is built as a compiled library, not as an executable file.

User project: it contains both the MC Application layer and the demonstration program that makes use of that layer through its MC API and provides the required clockings and access to Interrupt Handlers. Parameters and configurations related to user's application are used here to create right objects in what is called the run-time system 'boot'. The Motor Control API is the set of commands granted to the upper layer. The program can run some useful functions (depending on user options), such as serial communication, LCD/keys interface, system variables displaying through DAC.

Two equivalent and alternative types of user projects exist. They differ in how they generate the clocks: one implements a simple time base itself; the other exploits an Operating System, FreeRTOS, to do it.
13 user project workspaces are available. They differ in the supported STM32 family, IDE supported, how they generate the clocks: a simple time base itself or an Operating System (FreeRTOS).
The fist 8 are for IAR™ EWARM IDE and are stored in the folder Project\EWARM:

- STM32F0xx_Workspace for STM32F0xx devices and simple time base;
- STM32F10x_Workspace for both STM32F100xx and STM32F103xx devices and simple time base;
- STM32F2xx_Workspace for STM32F2xx devices and simple time base;
- STM32F30x_Workspace for STM32F302/303 devices and simple time base;
- STM32F4xx_Workspace for STM32F4xx devices and simple time base;
- STM32F10x_RTOS_Workspace for both STM32F100xx and STM32F103xx devices and FreeRTOS;
- STM32F2xx_RTOS_Workspace for STM32F2xx devices and FreeRTOS;
- STM32F10x_Example for both STM32F100xx and STM32F103xx devices with simple time base and ready-to-use examples.

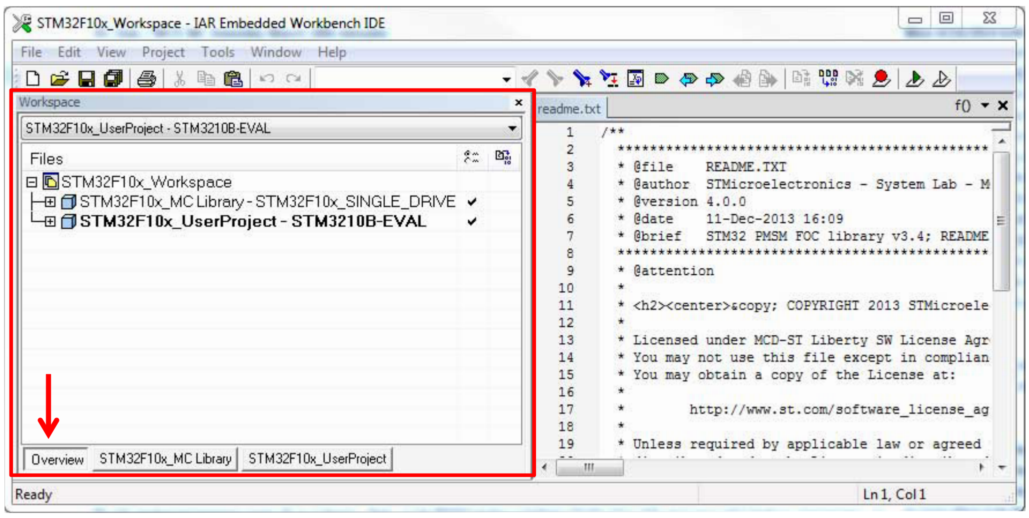
The remaining 5 are for Keil uVision and are stored in the folder Project\MDK-ARM:

- STM32F0xx_Workspace for STM32F0xx devices and simple time base;
- STM32F10x_Workspace for both STM32F100xx and STM32F103xx devices and simple time base;
- STM32F2xx_Workspace for STM32F2xx devices and simple time base;
- STM32F4xx_Workspace for STM32F4xx devices and simple time base;
- STM32F30x_Workspace for STM32F302/303 devices and simple time base.

Previously, the built ".lib" files are linked with the user project in order to generate the file that can be downloaded into microcontroller memory for execution.

Figure 7 provides an overview of the IAR EWARM IDE workspace (located in Installation folder `\Project\EWARM\STM32F10x_Workspace.eww`). The following sections provide details on this. The equivalent workspace based on FreeRTOS is located in Installation folder `\FreeRTOSProject\EWARM\ STM32F10x_RTOS_Workspace.eww`.

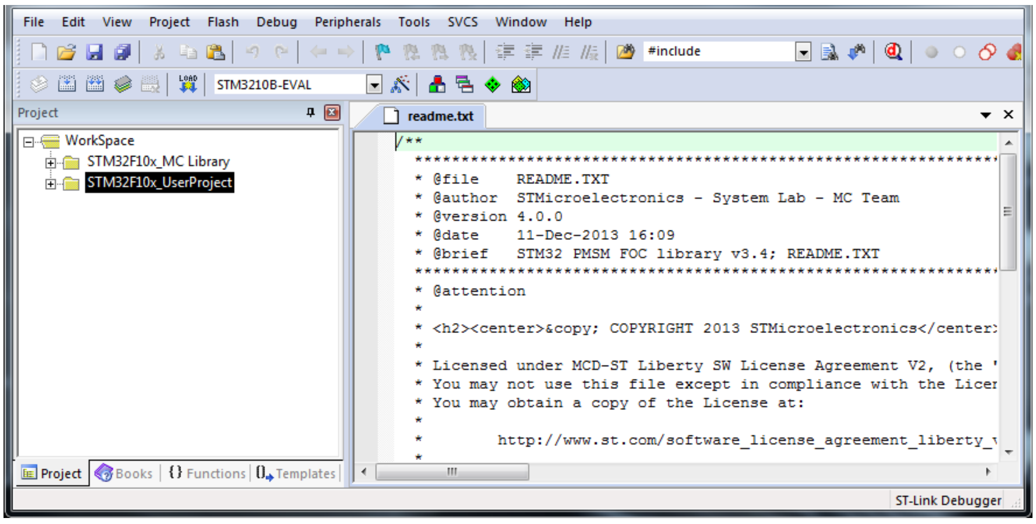
Figure 7. IAR EWARM IDE Workspace overview


STM32F10x_Workspace - IAR Embedded Workbench IDE
File Edit View Project Tools Window Help
Workspace
STM32F10x_UserProject - STM3210B-EVAL
Files
STM32F10x_Workspace
STM32F10x_MC Library - STM32F10x_SINGLE_DRIVE ✓
STM32F10x_UserProject - STM3210B-EVAL ✓
Overview
STM32F10x_MC Library
STM32F10x_UserProject
readme.txt
f()
Ready
Ln 1, Col 1
MS33855V1

```
/**
  ******************************************************
  * @file    README.TXT
  * @author  STMicroelectronics - System Lab - M
  * @version 4.0.0
  * @date    11-Dec-2013 16:09
  * @brief   STM32 PMSM FOC library v3.4; README
  ******************************************************
  * @attention
  *
  * <h2><center>&copy; COPYRIGHT 2013 STMicroele
  *
  * Licensed under MCD-ST Liberty SW License Agr
  * You may not use this file except in complian
  * You may obtain a copy of the License at:
  *
  *        http://www.st.com/software_license_ag
  *
  * Unless required by applicable law or agreed
```

Figure 8 provides an overview of the Keil uVision workspace (located in the Installation folder \Project\MDK-ARM\STM32F10x_Workspace.uvmpw).

Figure 8. Keil uVision workspace overview



```
/**
  ******************************************************************
  * @file    README.TXT
  * @author  STMicroelectronics - System Lab - MC Team
  * @version 4.0.0
  * @date    11-Dec-2013 16:09
  * @brief   STM32 PMSM FOC library v3.4; README.TXT
  ******************************************************************
  * @attention
  *
  * <h2><center>&copy; COPYRIGHT 2013 STMicroelectronics</center>
  *
  * Licensed under MCD-ST Liberty SW License Agreement V2, (the '
  * You may not use this file except in compliance with the Licer
  * You may obtain a copy of the License at:
  *
  *        http://www.st.com/software_license_agreement_liberty_v
  *
```

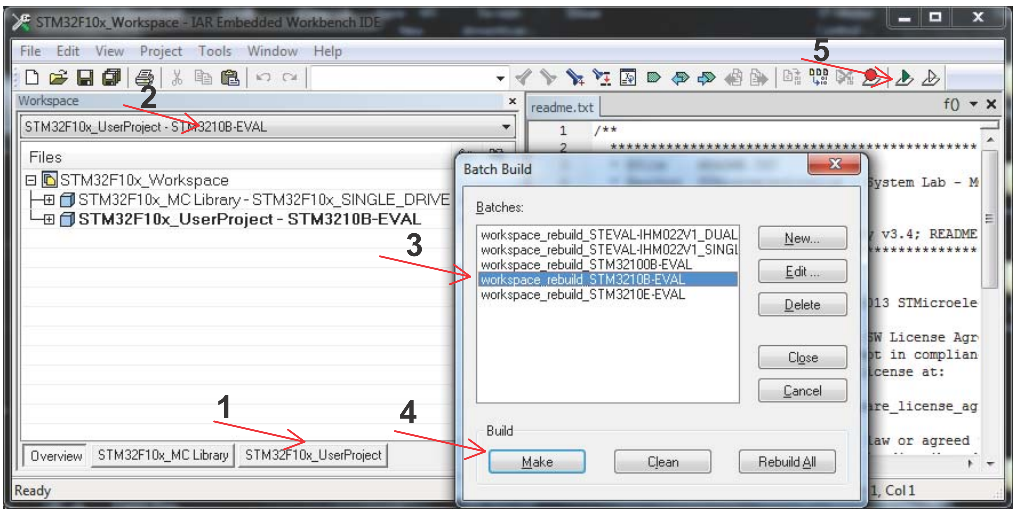
MS33856V1

5.2 MC SDK customization process

This section explains how to customize the Motor Control SDK using IAR EWARM IDE or Keil uVision, so that it corresponds with the user's current system.

1. Using the ST MC Workbench GUI configure the firmware according to the HW, motor and specific drive setting of the system. This part of the process ends by generating the .h parameters in the correct directory(`Installation folder\SystemDriveParams`).
2. If the system is configured to enable the LCD User Interface, download the specific firmware. See *Section 6: How to download the full LCD user interface* (this step is to be done only once).
   – Using IAR EWARM IDE follow the point: 3, 4, 5, 6.
   – Using Keil uVision follow the point 7, 8, 9, 10.
3. Open the MC workspace of choice:
   – Installation folder\FreeRTOS Project\EWARM\STM32F10x_RTOS_Workspace.eww
   – Installation folder\FreeRTOS Project\EWARM\STM32F2xx_RTOS_Workspace.eww
   – `Installation folder\Project\EWARM\STM32F0xx_Workspace.eww`
   – `Installation folder\Project\EWARM\STM32F10x_Workspace.eww`
   – `Installation folder\Project\EWARM\STM32F2xx_Workspace.eww`
   – `Installation folder\Project\EWARM\STM32F30x_Workspace.eww`
   – `Installation folder\Project\EWARM\STM32F4xx_Workspace.eww`
4. Enable the user project (call-out 1 in *Figure 9: Workspace batch build for IAR EWARM IDE*) and select the appropriate option from the combo-box (call-out 2 in *Figure 9*). If none of the boards displayed is in use, read *Table 1: Project configurations* to perform a correct configuration.
5. Press F8 to batch-build the entire workspace. The dialog box shown in *Figure 9* appears.
6. Select a batch command (call-out 3, *Figure 9*) as for step 4, then click the Make button to make the build (call-out 4, *Figure 9*). If no error or relevant warning appears, download the firmware (call-out 5, *Figure 9*) and do a test run.

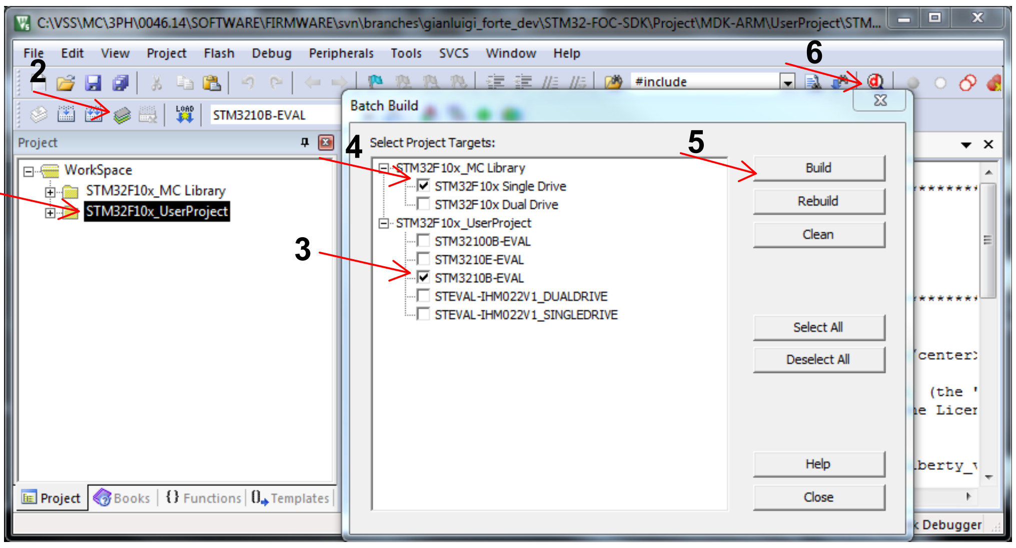
Figure 9. Workspace batch build for IAR EWARM IDE


STM32F10x_Workspace - IAR Embedded Workbench IDE
File Edit View Project Tools Window Help
Workspace
STM32F10x_UserProject - STM3210B-EVAL
Files
STM32F10x_Workspace
STM32F10x_MC Library - STM32F10x_SINGLE_DRIVE
STM32F10x_UserProject - STM3210B-EVAL
Overview
STM32F10x_MC Library
STM32F10x_UserProject
Ready
readme.txt
Batch Build
Batches:
workspace_rebuild_STEVAL-IHM022V1_DUAL
workspace_rebuild_STEVAL-IHM022V1_SINGL
workspace_rebuild_STM32100B-EVAL
workspace_rebuild_STM3210B-EVAL
workspace_rebuild_STM3210E-EVAL
New...
Edit ...
Delete
Close
Cancel
Build
Make
Clean
Rebuild All
1
2
3
4
5

MS33857V2

7. Open one of the MC workspaces:
   - Installation folder\Project\MDK-ARM\STM32F0xx_Workspace.uvmpw
   - Installation folder\Project\MDK-ARM\STM32F10x_Workspace.uvmpw
   - Installation folder\Project\MDK-ARM\STM32F2xx_Workspace.uvmpw
   - Installation folder\Project\MDK-ARM\STM32F3xx_Workspace.uvmpw
   - Installation folder\Project\MDK-ARM\STM32F4xx_Workspace.uvmpw
8. Enable the UserProject (callout 1 in *Figure 10*) right click on it and select "Set as Active Project" according the evaluation board used. If none of the boards displayed is in use, read *Table 1*: Project configurations to perform a correct configuration.
9. Press batch build button (callout 2 in *Figure 10*) The dialog box shown in *Figure 10*: Batch Build appears.
10. Select the configuration to be build according the step 7 (callout 3 in *Figure 10*) and selecting the proper conflagration of the MC Library (Single or Dual drive) (callout 4 in *Figure 10*). Then click Build button to make the build (callout 5 in *Figure 10*). If no error or relevant warning appears, download the firmware (callout 6, *Figure 10*) and do a test run.

Figure 10. Workspace batch build for Keil uVision


C:\VSS\MC\3PH\0046.14\SOFTWARE\FIRMWARE\svn\branches\gianluigi_forte_dev\STM32-FOC-SDK\Project\MDK-ARM\UserProject\STM...
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
#include
STM3210B-EVAL
Project
Project Books Functions Templates
Debugger
1
2
3
4
5
6
Build
Rebuild
Clean
Select All
Deselect All
Help
Close

Batch Build

Select Project Targets:

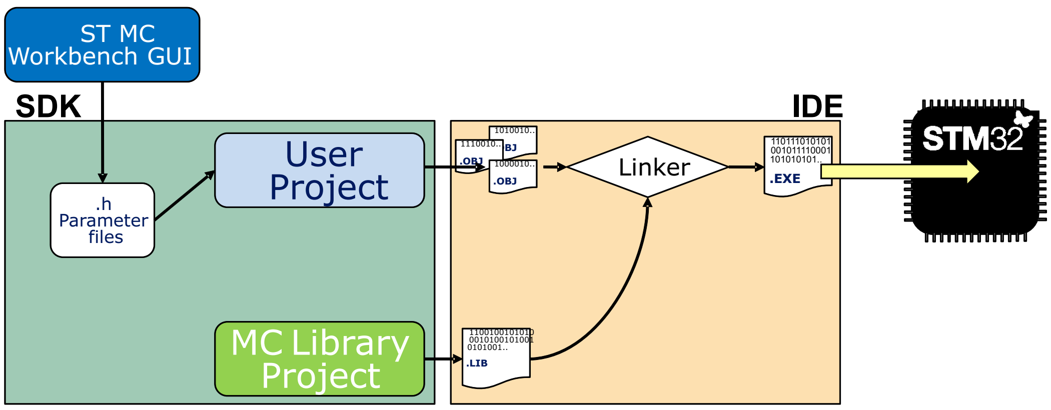
- STM32F10x_MC Library
  - [✓] STM32F10x Single Drive
  - [ ] STM32F10x Dual Drive
- STM32F10x_UserProject
  - [ ] STM32100B-EVAL
  - [ ] STM3210E-EVAL
  - [✓] STM3210B-EVAL
  - [ ] STEVAL-IHM022V1_DUALDRIVE
  - [ ] STEVAL-IHM022V1_SINGLEDRIVE

WorkSpace
- STM32F10x_MC Library
- STM32F10x_UserProject

MS33859V1

Note: *When the system configuration or parameters are modified, just the User project requires to be recompiled.The batch build procedure is requested just if the MC Library is provided as source code and only for the first compilation for both single and dual drive configuration. See Figure 11.*

Figure 11. Customization process



MS33858V1

Table 1. Project configurations

STM32 device part, single/dual drive selection	Viable configuration among existing
STM32F0xx, Single motor drive	STM320518-EVAL
STM32F100 low / medium / high density	STM32100B-EVAL
STM32F103 low density/medium density	STM3210B-EVAL
STM32F103 high density/XL density, Single motor drive	STM3210E-EVAL or STEVAL-IHM022V1_SINGLEDRIVE
STM32F103 high density/XL density, Dual motor drive	STEVAL-IHM022V1_DUALDRIVE
STM32F2xx, Single motor drive	STM322xG-EVAL
STM32F2xx, Dual motor drive	STM32F2xx_dual
STM32F302xB/C, Single motor drive	STM32302C_SINGLEDRIVE
STM32F302x6/8, Single motor drive	P-NUCLEO-IHM001_SINGLEDRIVE
STM32F303xB/C, Single motor drive	STM32303C-EVAL_SINGLEDRIVE
STM32F303xB/C, Dual motor drive	STM32303C-EVAL_DUALDRIVE
STM32F4xx, Single motor drive	STM324xG-EVAL STEVAL-IHM039V1_SINGLEDRIVE
STM32F4xx, Dual motor drive	STEVAL-IHM039V1_DUALDRIVE

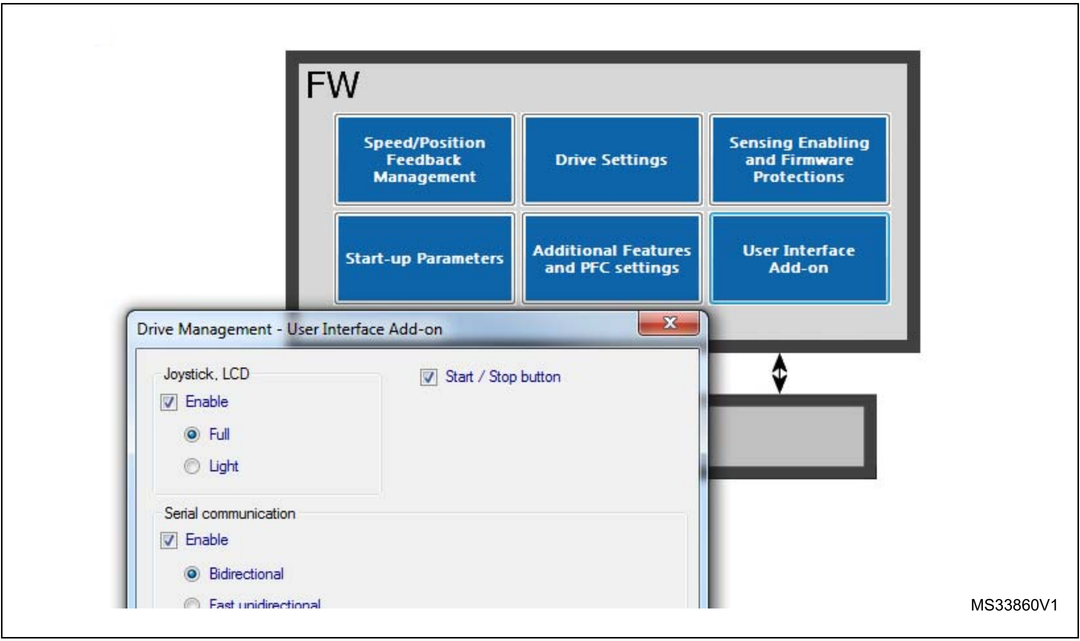
6 How to download the full LCD user interface

When an STM32 evaluation board equipped with LCD (such as STM3210B-EVAL, STM3210E-EVAL, STM32100B-EVAL, STEVAL-IHM022V1, STM322xG-EVAL, STM324xGEVAL, STEVAL-IHM039V1, or STM32303C-EVAL) is in use, you can enable the LCD plus Joystick User Interface—a useful feature of the demonstration user project that can be used as run-time command launcher, fine-tuning or monitoring tool (screens and functions are described in *Section 7*). This option can be selected via a setting in the ST MC Workbench GUI. See *Figure 12*.

In this case, download the LCD UI software (single or dual drive configuration) following the procedure explained below, in a reserved area in the microcontroller, located at the end of the addressable Flash memory. Unless you erase it or change the configuration from single-drive to dual-drive or vice-versa, there is no need to download it again. Even disabling the option with the GUI does not mean you need to flash it again when you re-enable the option.

The latest STM3210B-MCKIT Motor Control starter kits come with the Motor Control Library and LCD UI software (single-drive) pre-flashed. If your Motor Control kit has a previous version of Motor Control Library, you do not have the Motor Control kit but you are using one of the mentioned evaluation boards, or you are changing configuration (single-dual), you should follow one of the three procedures explained below to download the LCD UI.

Figure 12. Enabling the full LCD UI in the ST MC Workbench


FW
Speed/Position Feedback Management
Drive Settings
Sensing Enabling and Firmware Protections
Start-up Parameters
Additional Features and PFC settings
User Interface Add-on
Drive Management - User Interface Add-on
MS33860V1

Joystick, LCD
[✓] Enable
(●) Full
( ) Light

[✓] Start / Stop button

Serial communication
[✓] Enable
(●) Bidirectional
( ) Fast unidirectional

Option 1

Option 1 is straightforward and the preferred one.

1. Use the STM32 ST-LINK Utility tool to download the LCD pre-compiled file opening the proper MC workspaces, as explained in *Section 5.1*.
2. File->Open file...
3. Select the appropriate pre-compiled file (STM3210B-EVAL.hex, STM32100B-EVAL.hex, STM3210E-EVAL.hex, STM322xG-EVAL.hex,

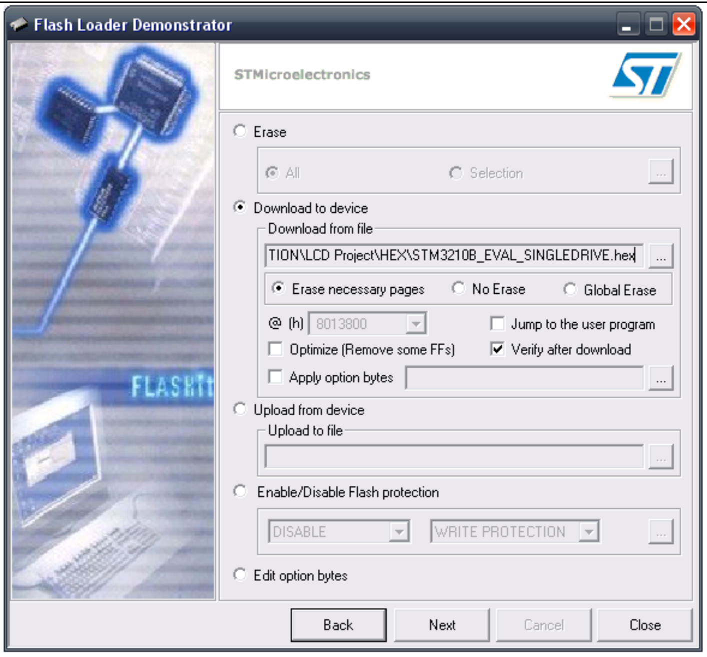
STM324xG-EVAL.hex, STM32303C-EVAL_SINGLEDRIVE.hex, STM32303C-EVAL_DUALDRIVE.hex, STEVAL-IHM022V1_SINGLEDRIVE.hex, STEVAL-IHM022V1_DUALDRIVE.hex, STEVAL-IHM039V1_SINGLEDRIVE.hex, STEVAL-IHM039V1_DUALDRIVE.hex).

Option 2

1. Use the STM32 and STM8 Flash loader demonstrator PC software package. This is available from the ST web site (*www.st.com* and in the \Installation folder\Utilities\Flash loader\.)

   The User Manual, UM0462 (included in the package), fully explains how to operate it. For communication purposes, you need to verify that you have an available COM port (RS232) on your PC.
2. After the program is installed, run the Flash loader demonstrator application from the Programs menu, making sure that the device is connected to your PC and that the boot configuration pins are set correctly to boot from the system memory (check the evaluation board user manual).
3. Reset the microcontroller to restart the system memory boot loader code.
4. When the connection is established, the wizard displays the available device information such as the target ID, the firmware version, the supported device, the memory map and the memory protection status. Select the target name in the target combo-box.
5. Click the **Download to device** radio button (see *Figure 13*) and browse to select the appropriate hexadecimal file (STM3210B_EVAL.hex, STM32100B_EVAL.hex, STM3210E_EVAL.hex, STEVAL_IHM022V1_SINGLEDRIVE.hex, STEVAL_IHM022V1_DUALDRIVE.hex, STM322xG-EVAL.hex, STM324xG-EVAL.hex, STEVAL_IHM039V1_SINGLEDRIVE.hex or STEVAL_IHM039V1_DUALDRIVE.hex) from Installation folder\LCD Project\Hex\.
6. Program the downloading to Flash memory. After the code is successfully flashed, setup the board to reboot from the user Flash memory and reset the microcontroller.

Figure 13. Flash loader wizard screen



( ) Erase
(● All) ( ) Selection

(●) Download to device

Download from file: TION\LCD Project\HEX\STM3210B_EVAL_SINGLEDRIVE.hex

(●) Erase necessary pages ( ) No Erase ( ) Global Erase

@ (h) 8013800

[ ] Jump to the user program

[ ] Optimize (Remove some FFs)

[✓] Verify after download

[ ] Apply option bytes

( ) Upload from device

Upload to file

( ) Enable/Disable Flash protection

DISABLE WRITE PROTECTION

( ) Edit option bytes

Option 3

This option is intended for users who want to modify the LCD UI code.

1. Use an IDE to rebuild and download the LCD UI.
2. After parameter files are generated by the GUI (to set the single/dual drive configuration) using KEIL uVision4 IDE, open the workspace located in:
   - `Installation folder\LCDProject\MDK-ARM\STM32F0xx_LCD Project.uvopt`
   - `Installation folder\LCDProject\MDK-ARM\STM32F10x_LCD Project.uvopt`
   - `Installation folder\LCDProject\MDK-ARM\STM32F2xx_LCD Project.uvopt`
   - `Installation folder\LCDProject\MDK-ARM\STM32F3xx_LCD Project.uvopt`
   - `Installation folder\LCDProject\MDK-ARM\STM32F4xx_LCD Project.uvopt.`

Figure 14. LCD UI project

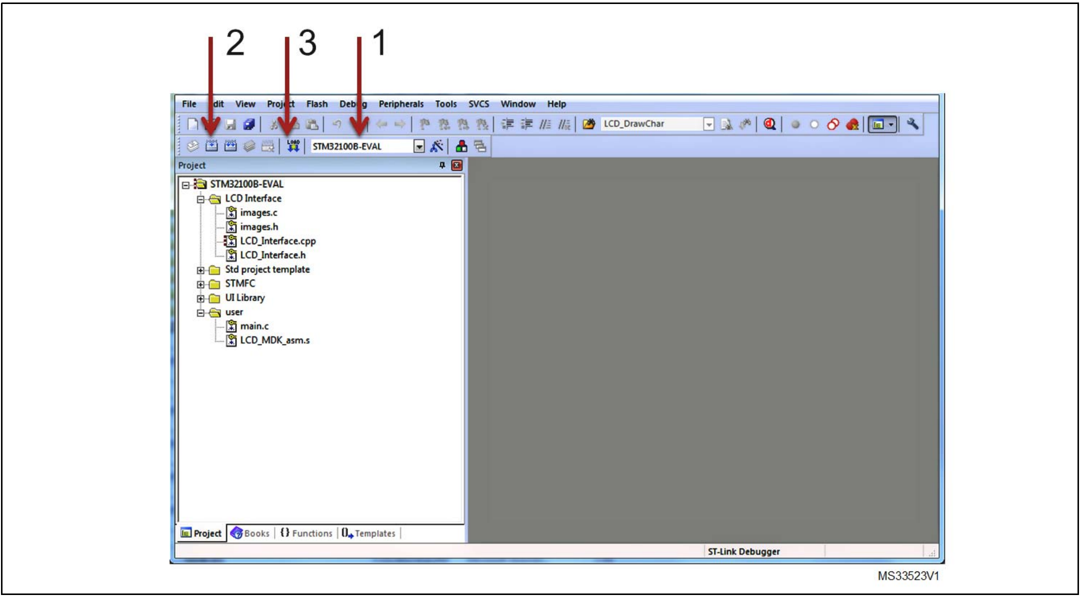


Figure 14 displays the logical arrangement of files (left-hand side) and actions that may be needed for set up and download.

Five project configurations are provided (call-out 1, *Figure 14*), one for each STM32 evaluation board that has been tested with the MC SDK:

– STM32F10B-EVAL
– STM32F10E-EVAL
– STM32F100B-EVAL
– STEVAL-IHM022V1_SINGLEDRIVE
– STEVAL-IHM022V1_DUALDRIVE

One project configuration is provided for the STM32F2xx_Workspace:

– STM322xG-EVAL

Three project configurations are provided for the STM32F4xx_Workspace:

– STM324xG-EVAL
– STEVAL-IHM039V1_SINGLEDRIVE
– STEVAL-IHM039V1_DUALDRIVE

One project configuration is provided for the STM32F0xx_Workspace:

– STM320518-EVAL

This configuration affects the LCD driver and linker file selection.

Two project configurations are provided for the STM32F3xx_Workspace:

– STM32303C-EVAL_SINGLEDRIVE
– STM32303C-EVAL_SINGLEDRIVE

3. Build the project (call-out 2, *Figure 14*), and download it to the microcontroller memory (call-out 3, *Figure 14*).
4. To test that the LCD UI is flashed correctly: open, build and download the user project (see *Section 5.2: MC SDK customization process*). From the debug session, run the firmware (F5) and then, after a while, stop debugging (CTRL+Shift+D). The LCD UI is not flashed properly if the program is stalled in a trap in UITask.c, line 195.

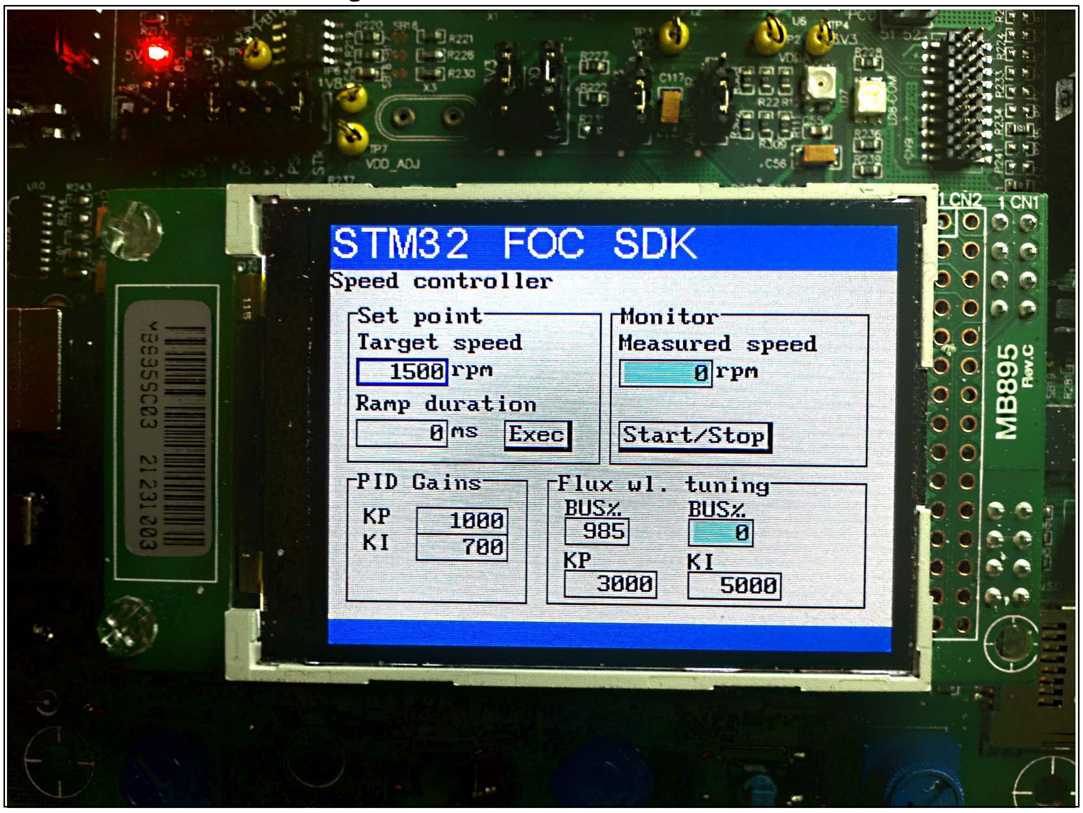
7 Full LCD user interface

7.1 Running the motor control firmware using the full LCD interface

The STM32 motor control library includes a demonstration program that enables you to display drive variables, customize the application by changing parameters, and enable and disable options in real time.

The user interface reference is the one present in the STM32 evaluation boards and is shown in *Figure 15*.

Figure 15. User interface reference



STM32 FOC SDK

Speed controller

Set point

Target speed

1500 rpm

Ramp duration

0 ms Exec

Monitor

Measured speed

0 rpm

Start/Stop

PID Gains

KP 1000

KI 700

Flux wl. tuning

BUS% 985

BUS% 0

KP 3000

KI 5000

The interface is composed of:

- A 320x240 pixel color LCD screen
- A joystick (see *Table 2* for the list of joystick actions and conventions)
- A push button (KEY button)

Table 2. Joystick actions and conventions

Keyword	User action
UP	Joystick pressed up
DOWN	Joystick pressed down
LEFT	Joystick pressed to the left

Table 2. Joystick actions and conventions (continued)

Keyword	User action
RIGHT	Joystick pressed to the right
JOYSEL	Joystick pushed
KEY	Press the KEY push button

In the default firmware configuration, the LCD management is enabled. It can be disabled using the STM32 MC Workbench or disabling the feature and manually changing the line: `define #define LCD_JOYSTICK_BUTTON_FUNCTIONALITY DISABLE` (line 316) of the Drive parameters.h file.

7.2 LCD User interface structure

The demonstration program is based on circular navigation pages.

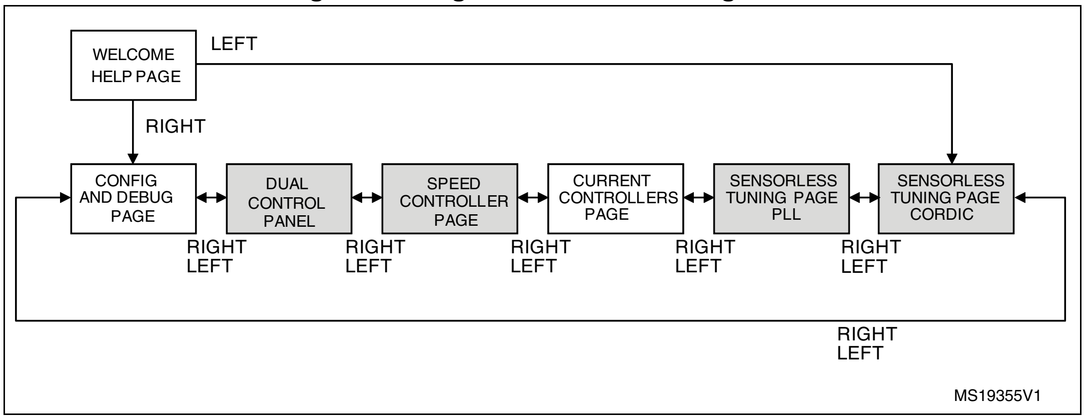
Figure 16 shows the page structure. The visibility of certain pages shown in *Figure 16* depends on the firmware configuration:

- Dual control panel is only present if the firmware is configured for dual motor drive.
- Speed controller page is only present when the firmware is configured in speed mode.
- Sensorless tuning page (PLL) is only present if the firmware is configured with state observer with PLL as primary or auxiliary speed sensor.
- Sensorless tuning page (CORDIC) is only present if the firmware is configured with state observer with CORDIC as primary or auxiliary speed sensor.

To navigate the help menus, use:

- RIGHT: navigate to the next page on the right
- LEFT: navigate to the next page on the left

Figure 16. Page structure and navigation



Each page is composed of a set of controls. *Table 3* presents the list of controls used in the LCD demonstration program.You can navigate between focusable controls in the page by pressing the joystick UP and DOWN. The focused control is highlighted with a blue rectangle. When focused, you can activate the control by pressing JOYSEL.

For some configurations such as STM32F100B-EVAL and STM320518-EVAL, a reduced set of LCD pages and/or controls is available.

Complete documentation about this LCD User Interface can be found in User manual STM32F PMSM single/dual FOC SDK (UM1052).

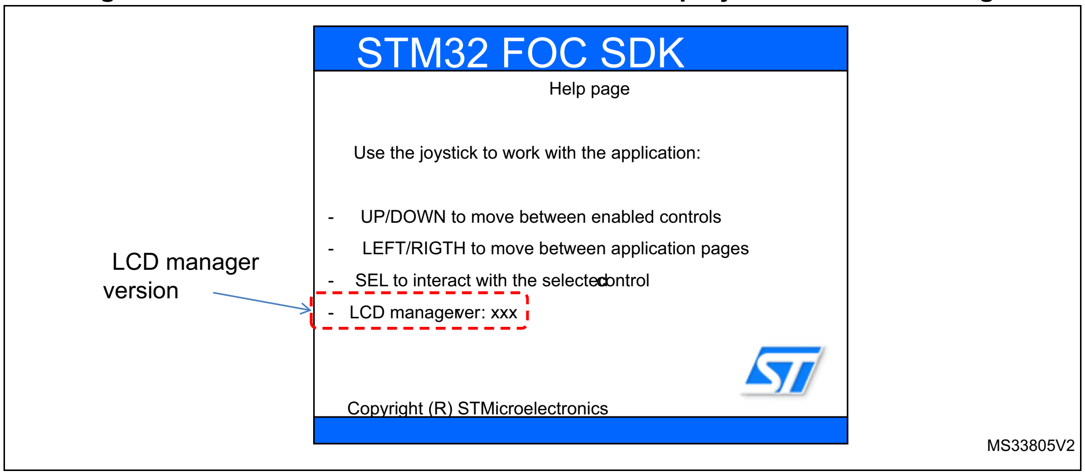
Table 3. List of controls used in the LCD demonstration program

Control name and examples	Description
Edit box 1500 rpm	Manages a numerical value. It can be “read only” or “read/write”. A read only edit box has a gray background and cannot be focused. A read/write edit box has a white background and can be focused. When a read/write edit box is focused, it can be activated for modification by pressing JOYSEL. An activated read/write edit box has a green background and its value can be modified pressing and/or keeping joystick UP/DOWN pressed. The new value is set to the motor control-related object instantaneously when the value changes, unless otherwise mentioned in this manual.
Combo-box Speed ▼	Manages a list of predefined values. For example, Speed or Torque control mode. When focused, it can be activated for modification by pressing JOYSEL. An activated combo-box has a green background and its value can be modified by pressing the joystick UP/DOWN. When the value changes, the new value is instantaneously set to the motor control-related object, unless otherwise mentioned in this manual.
Button Fault ack	Sends commands. For example, a start/stop button. A disabled button is drawn in light gray and cannot be focused. An enabled button is painted in black and can be focused. When focused, pressing JOYSEL corresponds to “pushing” the button and sending the related command.

7.2.1 Welcome message

After the STM32 evaluation board is powered on or reset, a welcome message displays on the LCD screen to inform the user about the firmware code loaded and the version of the release. See *Figure 17*.

Figure 17. STM32 Motor Control demonstration project welcome message


STM32 FOC SDK

Help page

Use the joystick to work with the application:

- UP/DOWN to move between enabled controls
- LEFT/RIGTH to move between application pages
- SEL to interact with the selectedcontrol
- LCD managerver: xxx

Copyright (R) STMicroelectronics

7.2.2 Configuration and debug page

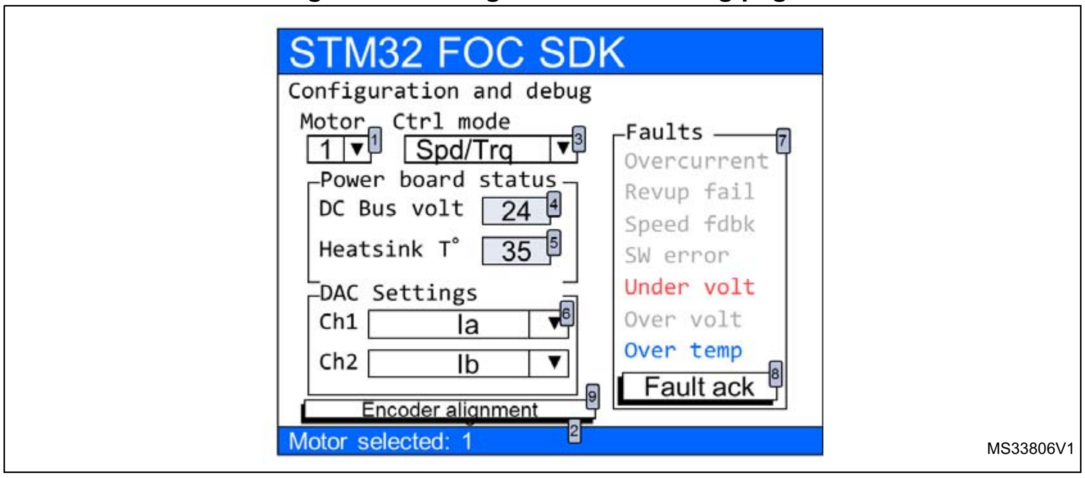
Press the joystick RIGHT from the welcome page to enter the Configuration and debug page.

To navigate between focusable controls on the page, press the joystick UP/DOWN.

Use the Configuration and debug page shown in *Figure 18* to:

- Select the active motor drive (1). This control is present only for dual motor control applications. This combo-box enables you to select the active motor drive. Once the active motor is selected, it is shown in the status bar present at the bottom of the screen (2). Commands performed on, or feedback from a control are only relative to the active motor.
- Select the control mode (3). Two control modes are available: speed and torque. You can change the control mode from speed to torque and vice versa on-the-fly even if the motor is already running.

Figure 18. Configuration and debug page



- Read the DC bus voltage value (4). This control is read-only.
- Read the heat sink temperature value (5). This control is read-only.
- Select the variables to be put in output through DAC channels (6). These controls are present only if the DAC option is enabled in the firmware. The list of variables also depends on firmware settings.
- It is possible to read the list of fault causes (7) if fault conditions have occurred, or if they are still present. The list of possible faults is summarized in *Table 4*. If a fault condition occurred and is over, the relative label is displayed in blue. If a fault condition is still present, the relative label is displayed in red. It is gray if there is no error.
- To acknowledge the fault condition, press the **Fault ack** button (8). If a fault condition occurs, the motor is stopped and it is no longer possible to navigate in the other pages. In this condition, it is not possible to restart the motor until the fault condition is over and the occurred faults have been acknowledged by the user, pushing the **Fault ack** button (8). If a fault condition is running, the **Fault ack** button is disabled.

Table 4. Fault conditions list

Fault	Description
Overcurrent	This fault occurs when the microcontroller break input signal is activated. It is usually used to indicate a hardware over current condition.
Revup fail	This fault occurs when the programmed rev-up sequence ends without validating the speed sensor information. The rev-up sequence is performed only when the state observer is configured as primary speed sensor.
Speed fdbk	This fault occurs only in RUN state when the sensor no longer meets the conditions of reliability.
SW error	This fault occurs when the software detects a general fault condition. In the present implementation, the software error is raised when the FOC frequency is too high to be sustainable by the microcontroller.
Under volt	This fault occurs when the DC bus voltage is below the configured threshold.
Over volt	This fault occurs when the DC bus voltage is above the configured threshold. If the dissipative brake resistor management is enabled, this fault is not raised.
Over temp	This fault occurs when the heat sink temperature is above the configured threshold.

- Execute encoder initialization. If the firmware is configured to use the encoder as the primary speed sensor or auxiliary speed sensor, the **Encoder alignment** button (9) is also present. In this case, the alignment of the encoder is required only once after each reset of the microcontroller.

7.2.3 Dual control panel page

This page is present only if the firmware is configured for dual motor drive.

To enter the Dual control panel page, press the joystick RIGHT from the Configuration and debug page.

It is possible to navigate between focusable controls present in the page by pressing the joystick UP/DOWN.

The Dual control panel page shown in *Figure 19* is used to send commands and get feedback from both motors. It is divided into three groups:

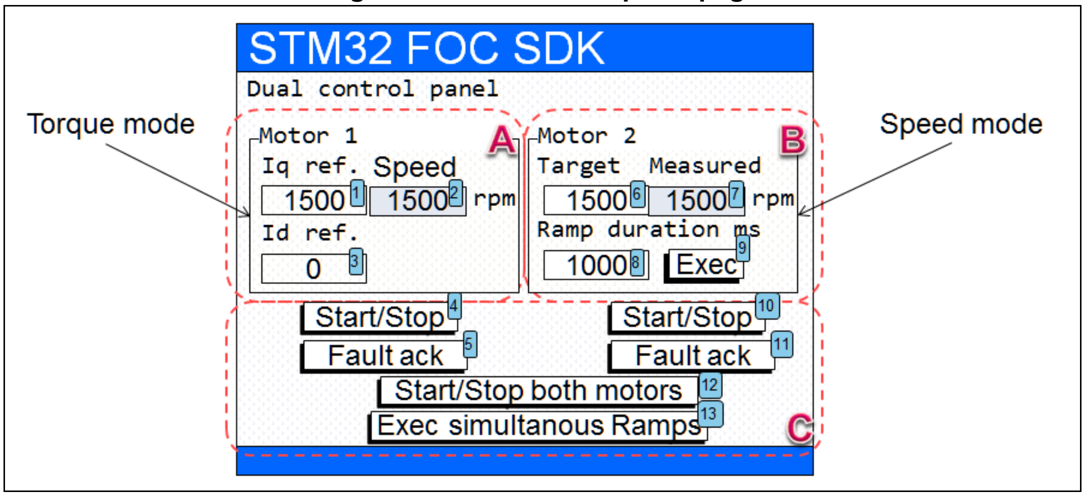
- Groups A and B depend on speed/torque settings. The group content is updated on-the-fly when the control mode (torque/speed) is changed in the Configuration and debug page. The control present in group A is related to the first motor. The control present in group B is related to the second motor.
- Group C does not depend on speed/torque settings. The control present in this group is related to both motors.

Figure 19 shows an example in which the first motor is set in torque mode and the second motor is set in speed mode.

The controls present in this page are used as follows:

- To set the I_q reference (1). This is related to motor 1 and is only present if motor 1 is set in torque mode. I_q reference is expressed in s16A. In this page, the current references are always expressed as Cartesian coordinates (I_q,I_d).

Figure 19. Dual control panel page



- To set the I_d reference (3). This is related to motor 1. This control is only present if motor 1 is set in torque mode. Id reference is expressed in s16A. In this page, the current references are always expressed as Cartesian coordinates (I_q,I_d).

Note: To convert current expressed in Amps to current expressed in digits, use the following formula:

*Current(s16A) = [Current(Amp) * 65536 * Rshunt * Aop] / Vdd micro.*

- Set the final motor speed of a speed ramp (6). This is related to motor 2. This control is only present if motor 2 is set in speed mode. Motor speed is expressed in RPM. The value set in this control is not automatically sent to the motor control related object but it is used to perform a speed ramp execution. See the Exec button description (9).
- Set the duration of a speed ramp (8). This is related to motor 2. This control is only present if motor 2 is set in speed mode. The duration is expressed in milliseconds. The value set in this control is not automatically sent to the motor control related object, but it is used to perform a speed ramp execution. See the Exec button description (9). It is possible to set a duration value of 0 to program a ramp with an instantaneous change in the speed reference from the current speed to the final motor speed (6).
- Execute a speed ramp by pushing the “Exec” button (9). This is related to motor 2. This control is only present if motor 2 is set in speed mode. The Exec speed ramp command is sent to the motor control related object together with the final motor speed and duration currently selected (6). The Exec speed ramp command performs a speed ramp from the current speed to the final motor speed in a time defined by the duration. The command is buffered and takes effect only when the motor is in RUN state.
- To read the motor speed, respectively (2) and (7) for motor 1 and motor 2. The motor speed is expressed in RPM. This control is read-only.
- Send a start/stop command, (4) for motor 1, (10) for motor 2. This is performed by pushing the start/stop button. A start/stop command means: start the motor if it is stopped, or stop the motor if it is running. If the drive is configured in speed mode when the motor starts, a speed ramp with the latest values of the final motor speed and duration is performed. If a fault condition occurs at any time, the motor is stopped (if running) and the start/stop button is disabled.
- When a fault condition is over, the **Fault ack** button, (5) for motor 1, (11) for motor 2, is enabled. Pushing this button acknowledges the fault conditions that have occurred. After the fault is acknowledged, the start/stop button becomes available again. When a

fault occurs and before it is acknowledged, it is only possible to navigate in the Dual control panel page and the Configuration and debug page.

- To start or stop both motors simultaneously, push the **Start/Stop both motors** button (12). This button is enabled only when the motors are both in Idle state or both in RUN state. If any of the motors is configured in speed mode when it starts, a speed ramp with the last values of the final motor speed and duration is performed. It is possible to stop both motors at any time by pushing the KEY button.
- To execute simultaneous speed ramps on both motors, push the **Exec simultaneous Ramps** button (13). This button is disabled when at least one of the two motors is configured in torque mode. The Exec speed ramp command is sent to both motor control objects together with the related final motor speed and the duration currently selected. The Exec speed ramp command performs a speed ramp from the current speed to the final motor speed in a time defined by the duration for each motor. The commands are buffered and take effect only when the related motor is in RUN state.

7.2.4 Speed controller page

This page is only present if the control mode set in *Figure 18* is the speed mode.

To enter the Speed controller page, press the joystick RIGHT from the Configuration and debug page (or from the Dual control panel page, if the firmware is configured in dual motor drive).

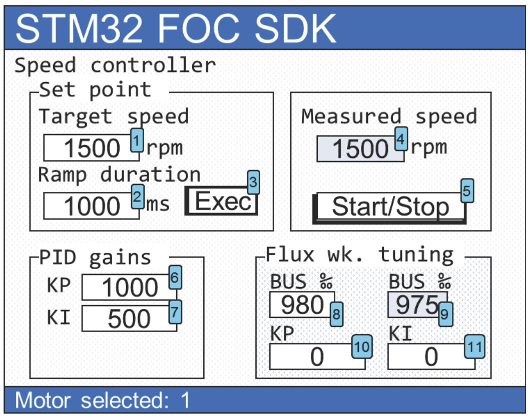
It is possible to navigate between focusable controls present in the page by pressing the joystick UP/DOWN.

The Speed controller page shown in *Figure 20* is used to send commands and get feedback related to the speed controller from the active motor. There are four groups of controls in this page:

Table 5. Control groups

Control group	Description
Set point	Used to configure and execute a speed ramp
PID gains	Used to change the speed controller gains in real- time
Flux wk. tuning	Used to tune the flux weakening related variables
Measured speed with start/stop button	Composed of two controls that are also present in the Current controllers page and in the sensorless tuning page; it provides a fast access to the measured speed and to the motor start/stop function

Figure 20. Speed controller page



If the firmware is configured as dual motor drive, it is possible to know which motor is active by reading the label at the bottom of the page. To change the active motor, go to the Configuration and debug page and change (1) in *Figure 18*.

Table 6 lists the actions that can be performed using this page.

Table 6. Speed controller page controls

Control	Description
Target speed (1 in *Figure 20*)	Sets the final motor speed of a speed ramp for the active motor. The motor speed is expressed in RPM. The value set in this control is not automatically sent to the motor control related object, but it is used to perform a speed ramp execution. See the **Exec** button description (3)
Ramp duration (2)	Sets the duration of a speed ramp for the active motor. The duration is expressed in milliseconds. The value set in this control is not automatically sent to the motor control related object, but it is used to perform a speed ramp execution. See the **Exec** button description (3). It is possible to set a duration value of 0 to program a ramp with an instantaneous change in the speed reference from the current speed to the final motor speed (1).
Exec button (3)	Executes a speed ramp for the active motor. The execute speed ramp command is sent to the motor control related object together with the final motor speed and duration presently selected (1) and (2). The execute speed ramp command performs a speed ramp from the current speed to the final motor speed in a time defined by duration. The command is buffered and takes effect only when the motor becomes in RUN state.
Measured speed (4)	Reads the motor speed for the active motor. The motor speed is expressed in RPM and is a read-only control.
Start/Stop button (5)	Sends a start/stop command for the active motor. A start/stop command starts the motor if it is stopped, or stops a running motor. Used with a motor start, a speed ramp with the last values of the final motor speed and duration is performed. If a fault condition occurs at any time, the motor is stopped (if running) and the Configuration and debug page displays.
Speed PID gain KP (6)	Sets the proportional coefficient of the speed controller for the active motor. The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of the speed controller.

Table 6. Speed controller page controls (continued)

Control	Description
Speed PID gain KI (7)	Sets the integral coefficient of the speed controller for the active motor. The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of the speed controller.
Bus‰ (8)	The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of flux weakening controller. The value is expressed in per mil (‰) of DC bus voltage.
Bus‰ (9)	DC bus voltage percentage presently used for the active motor; it is a read-only control. This control is present only if the flux weakening feature is enabled in the firmware. The value is actually expressed in per mil (‰) of DC bus voltage.
Flux wk PI gain KP (10)	The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of the flux weakening controller.
Flux wk PI gain KI (11)	The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of the flux weakening controller.

7.2.5 Current controllers page

To enter the Current controllers page, press the joystick RIGHT from the Speed controller page (or from one of the pages described above if the Speed controller page is not visible).

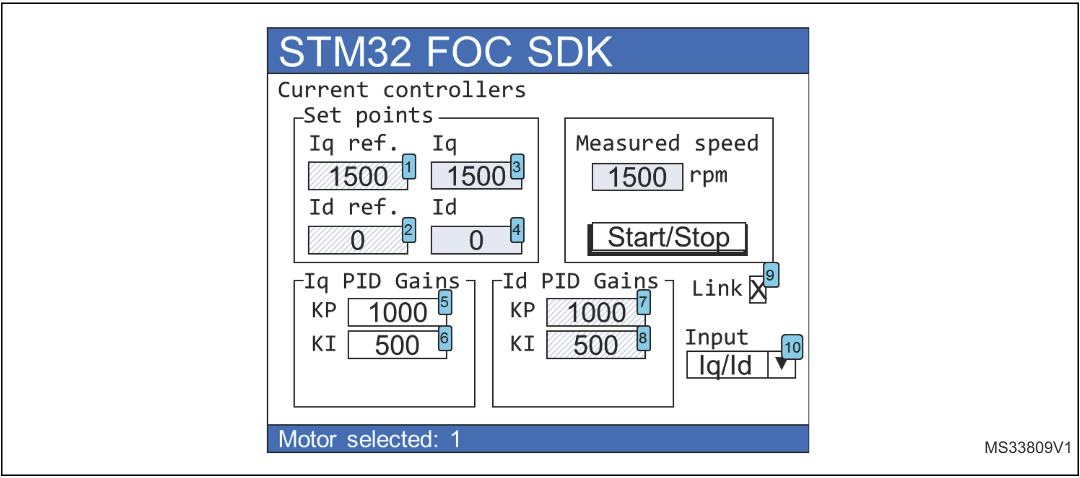
It is possible to navigate between focusable controls present in the page by pressing the joystick UP/DOWN.

The Current controllers page shown in *Figure 21* is used to send commands and get feedback related to current controllers, from the active motor. There are five control groups in this page:

Table 7. Control groups

Control group	Description
Set point	Used to set the current references and read the measured currents
Iq PID gains	Used to change the speed controller gains in real time
Id PID gains	
Measured speed with start/stop button	Composed of two controls that are also present in the Current controllers page and in the Sensorless tuning page. It provides a fast access to the measured speed and to the motor start/stop function

Figure 21. Current controllers page



If the firmware is configured as dual motor drive, it is possible to know which motor is active by reading the label at the bottom of the page. To change the active motor, go to the Configuration and debug page and change (1) in *Figure 21*.

Table 8 lists the actions that can be performed using this page.

Table 8. Current controllers page controls

Control	Description
I_q reference (1 in *Figure 21*)	Sets and reads the I_q reference for the active motor. This control is read-only if the active motor is set in speed mode, otherwise it can be modified. The I_q reference is expressed in s16A. To convert current expressed in Amps to current expressed in digits, use the formula: Current(s16A) = [Current(Amp) * 65536 * Rshunt * Aop] / Vdd micro
I_d reference (2)	Sets and reads the I_d reference for the active motor. This control is usually read-only if the active motor is set in speed mode, otherwise it can be modified. The I_d reference is expressed in s16A.
Measured I_q (3)	Reads the measured I_q for the active motor. Measured I_q is expressed in s16A and is a read-only control.
Iq PI(D) gain, KP (5)	Sets the proportional coefficient of the I_q current controller for the active motor. The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of the current controller.
Iq PI(D) gain, KI (6)	Sets the integral coefficient of the Iq current controller for the active motor. The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of the current controller.
Id PI(D) gain, KP (7)	Sets the proportional coefficient of the I_d current controller for the active motor. The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of the current controller. This control is only read if the link check box is checked.
Id PI(D) gain, KI (8)	Sets the integral coefficient of the I_d current controller for the active motor. The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of the current controller. This control is only read if the link check box is checked.

8 Introduction to the PMSM FOC drive

This software library is designed to achieve the high dynamic performance in AC permanent-magnet synchronous motor (PMSM) control offered by the well-established field oriented control (FOC) strategy.

With this approach, it can be stated that, by controlling the two currents i_{qs} and i_{ds}, which are mathematical transformations of the stator currents, it is possible to offer electromagnetic torque (T_e) regulation and, to some extent, flux weakening capability.

This resembles the favorable condition of a DC motor, where those roles are held by the armature and field currents.

Figure 22. Basic FOC algorithm structure, torque control

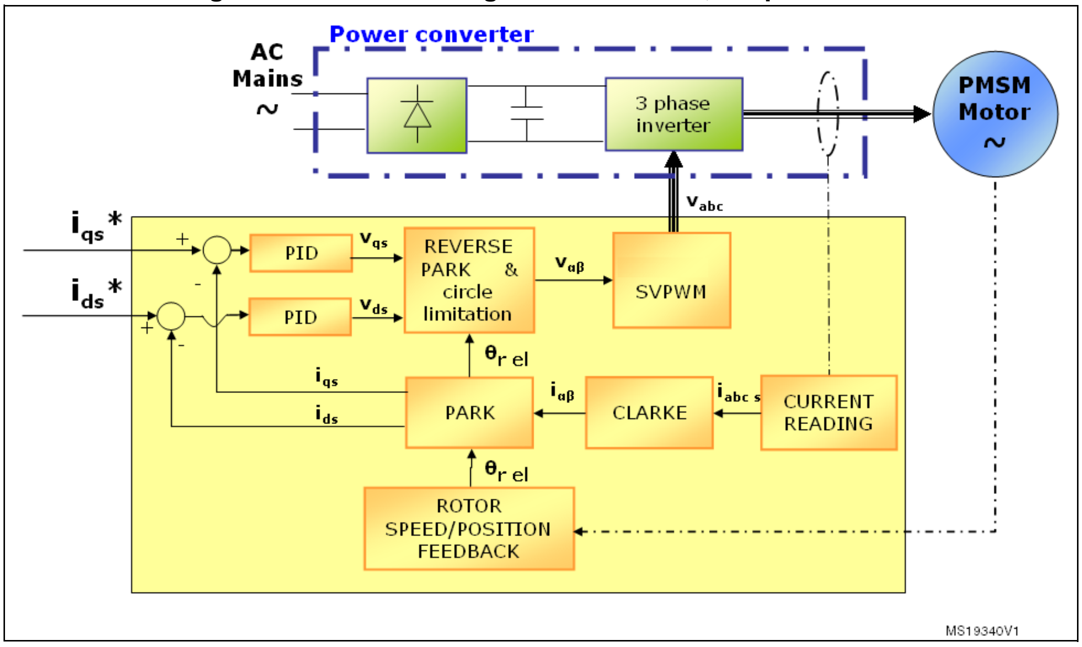
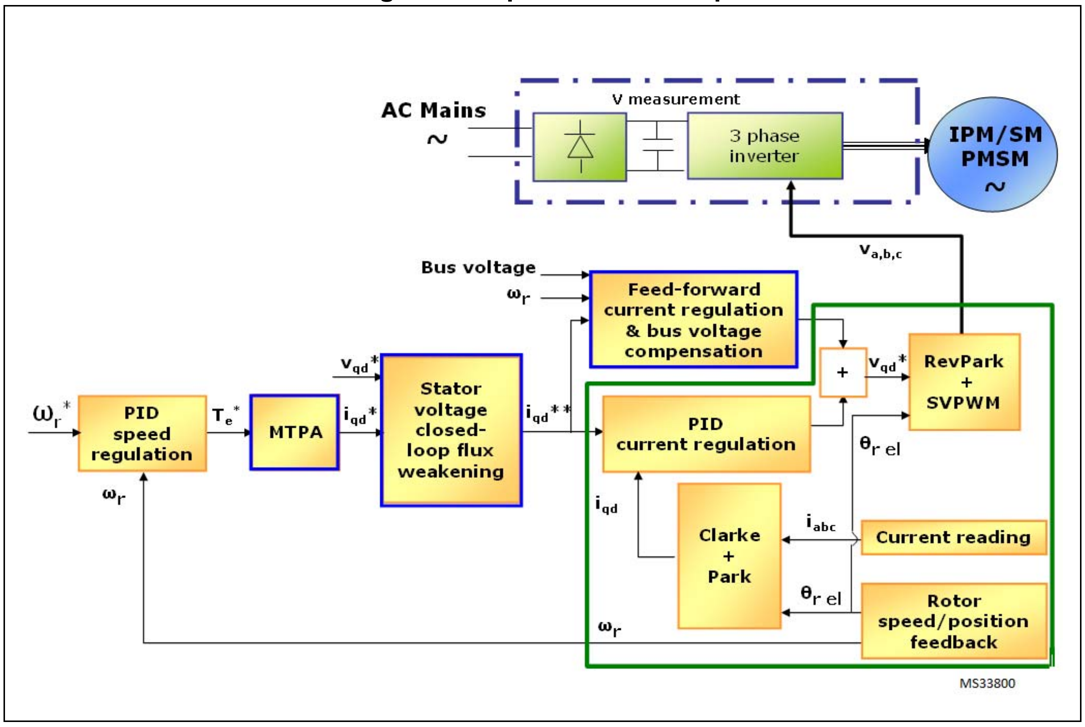


Figure 23. Speed control loop



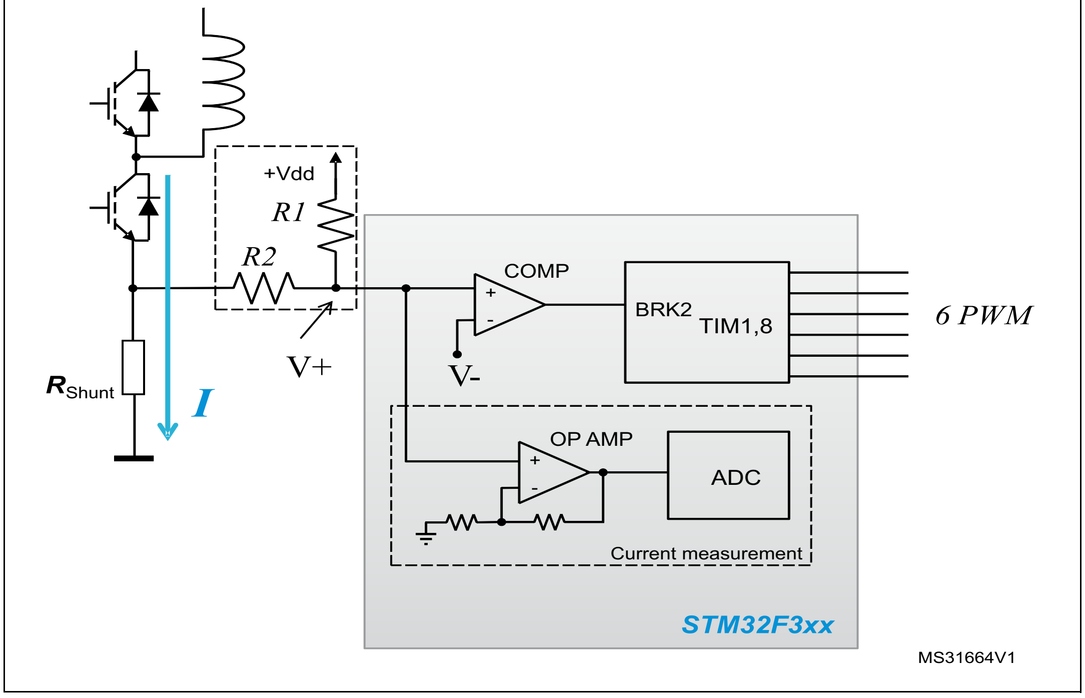
9 Current sensing and protection with embedded analog (STM32F3x)

9.1 Introduction

The STM32F302/303 microcontrollers features have an enhanced set of peripherals. They include comparators, PGAs, DACs and high-speed ADCs. This section describes how to use these peripherals accordingly to what is made available by the MC library

Figure 24. shows a current sensing and over-current protection network that can be implemented using the internal resources of the STM32F302/303: due to the motor phase current, the voltage drop on the shunt resistor can be either positive or negative, an offset is set by R1 and R2. Then, the signal is linked to a microcontroller pin that has both functionality of amplifier and comparator non-inverting input.

Figure 24. Current sensing and over-current protection with STM32F302/303



This configuration shows the optimization that can be reached, using STM32F3, both in terms of reduced number of external component and microcontroller pins assigned for the MC application.

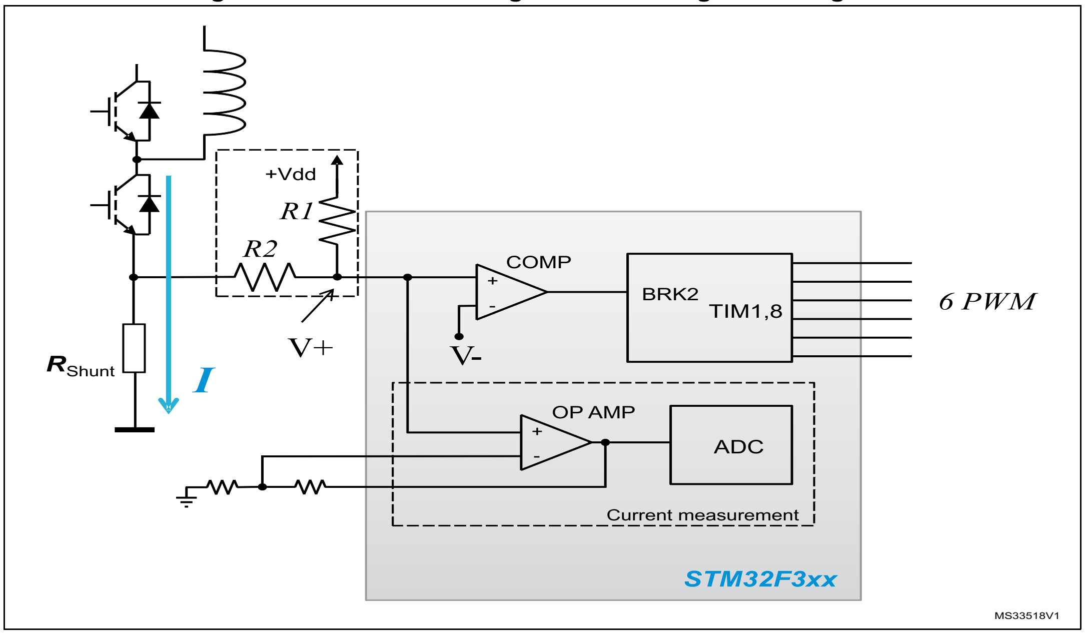
9.2 Current sensing

To maximize the resolution of the measurement, the embedded Programmable Gains Amplifier (PGA) can be used to adapt the level of voltage drop in the shunt resistor (Rshunt), caused by the motor current, up to the maximum range allowed by the analog to digital converter (ADC).

The PGA has a set of fixed internal gains (x2, x4, x8, x16). An alternative option in PGA mode allows you to route the central point of the resistive network on one of the I/Os connected to the non-inverting input: this feature can be used for instance to add a low-pass filter to PGA, as shown in *Figure 26.*

On the other hand, if a different value of amplification is required, it is possible to completely define the amplification network (for instance as it's shown in *Figure 25.*).

Figure 25. Current sensing network using external gains

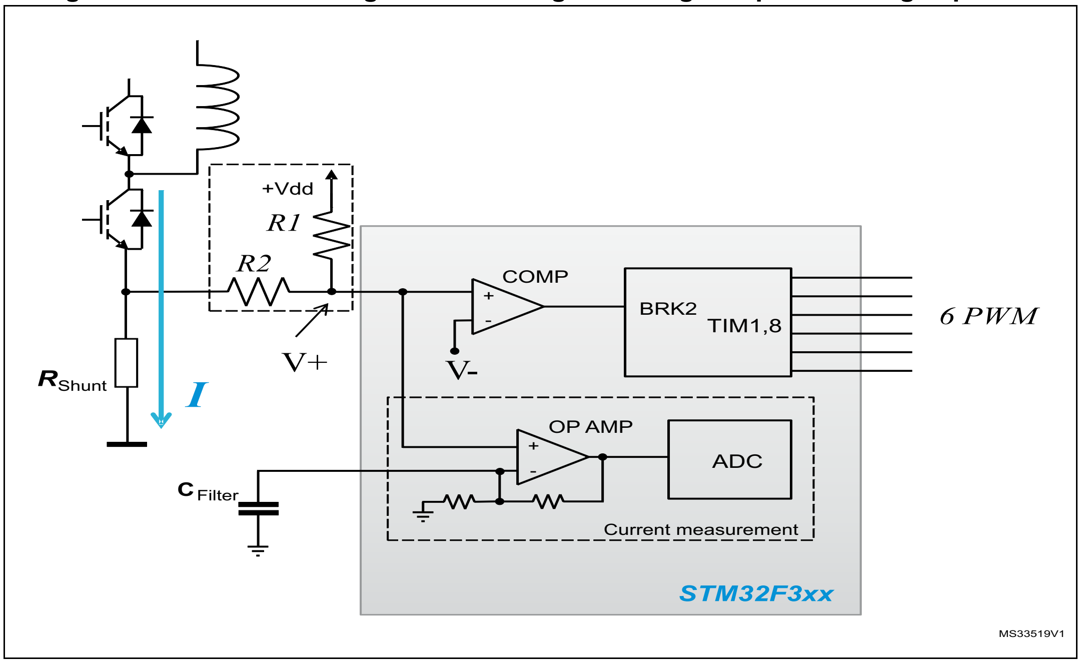


The MC library can be arranged to match all the configurations shown: by using the ST MC Workbench, creating a project based on STM32F302 or STM32F303, from the dialogue window located in Control Stage -> Analog Input -> Phase current feedback (*Figure 27.*) setting:

– “Embedded PGA” as current sensing topology;
– PGA internal gain (like in *Figure 24.*): Settling “Internal” in the “OPAMP Gain” drop down list;
– PGA external gain (like in *Figure 25.*): Settling “External” in the “OPAMP Gain” drop down list;
– PGA internal gain with external filtering capacitor (like in *Figure 26.*): Settling “Internal” in the “OPAMP Gain” drop down list and checking the “Feedback net filtering” check box in the same group.

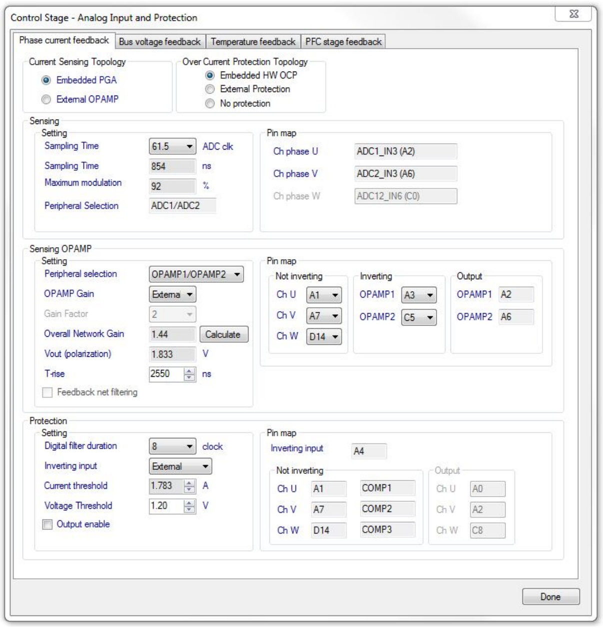
Just one of this setting is present in the workbench for each drives, since the configuration applies to each shunt resistor conditioning network.

Figure 26. Current sensing network using internal gains plus filtering capacitor



On the other hand, it is possible to setup the motor current measurement network to use external operational amplifiers. In this case the amplified signals are directly fed to the ADC channels. By using the ST MC Workbench, creating a project based on STM32F302 or STM32F303, from the dialogue window located in Control Stage -> Analog Input -> Phase current feedback, setting “External OPAMP” as current sensing topology.

Figure 27. STMCWB window related to PGA/COMP settings for motor currents



Control Stage - Analog Input and Protection

Phase current feedback | Bus voltage feedback | Temperature feedback | PFC stage feedback

Current Sensing Topology: [✓ Embedded PGA] External OPAMP

Over Current Protection Topology: [✓ Embedded HW OCP] External Protection No protection

Sensing

Setting		
Sampling Time	61.5	ADC clk
Sampling Time	854	ns
Maximum modulation	92	%
Peripheral Selection	ADC1/ADC2	

Pin map	
Ch phase U	ADC1_IN3 (A2)
Ch phase V	ADC2_IN3 (A6)
Ch phase W	ADC12_IN6 (C0)

Sensing OPAMP

Setting		
Peripheral selection	OPAMP1/OPAMP2	
OPAMP Gain	Externa	
Gain Factor	2	
Overall Network Gain	1.44	Calculate
Vout (polarization)	1.833	V
T-rise	2550	ns

Feedback net filtering

Pin map

Not inverting		Inverting		Output	
Ch U	A1	OPAMP1	A3	OPAMP1	A2
Ch V	A7	OPAMP2	C5	OPAMP2	A6
Ch W	D14				

Protection

Setting		
Digital filter duration	8	clock
Inverting input	External	
Current threshold	1.783	A
Voltage Threshold	1.20	V

Output enable

Pin map

Inverting input: A4

Not inverting			Output	
Ch U	A1	COMP1	Ch U	A0
Ch V	A7	COMP2	Ch V	A2
Ch W	D14	COMP3	Ch W	C8

Done

MS33802V1

9.3 Overcurrent protection

The basic principle of the hardware over-current protection mechanism can be summarized as follows:

- The phase current of the motor flows in the power transistor of the inverter bridge and passes through the shunt resistor (RShunt) producing a voltage drop (V+).
- This voltage drop is compared with a threshold (V-) defining the maximum admissible current.
- If the threshold is exceeded, a break signal stops the PWM generation putting the system in a safe state.

All of these actions can be performed using the internal resources of the STM32F302/303 and, in particular, the embedded comparators and the advanced timer break function (BRK2). As shown in *Figure 24.*, *Figure 25.*, and *Figure 26.* the same signal is fed to both not inverting input of embedded comparators and PGA.

The overcurrent threshold (V-) can be defined in three different ways:

- using one of the available internal voltage reference (1.2V, 0.9V, 0.6V, 0.3V);
- providing it externally using the inverting input pin of the comparator;
- programming a DAC channel.

The MC library can be arranged to match all the configurations shown by using the ST MC Workbench, creating a project based on STM32F302 or STM32F303, from the dialogue window located in Control Stage -> Analog Input -> Phase current feedback (*Figure 27.*), setting:

- “Embedded HW OCP” radio button as overcurrent protection topology;
- HW OCP internal threshold: selecting “Internal” in the “Inverting input” drop down list and choosing the internal voltage reference (among available values) in “Voltage Threshold”.
- HW OCP external threshold: selecting “External” in the “Inverting input” drop down list and editing the external voltage reference in “Voltage Threshold”.
- HW OCP internal threshold using DAC: selecting “DAC” in the “Inverting input” drop down list and editing the DAC voltage reference to be generated in “Voltage Threshold”. A DAC channel must be assigned for this functionality (OCP) from the related dialogue window located in Control stage -> DAC functionality (*Figure 30.*).

On the other hand, it is possible to setup the motor overcurrent protection network to use external components. In this case the overcurrent protection signal - coming from a comparator for instance - is directly fed to the advanced-timer's BKIN2 pin. By using the ST MC Workbench, creating a project based on STM32F302 or STM32F303, from the dialogue window located in Control Stage -> Analog Input -> Phase current feedback, setting “External protection” as OCP protection topology.

In any case, either using embedded comparators or external components, a digital filter, upstream the BKIN2 function, can be enabled and configured to reject noises.

9.4 Resources allocation - single drive

For project based on STM32F302/303 the current feedback network configurations supported by STM32 FOC SDK are single shunt and three shunt.

Single shunt topology: according to configuration (as explained in *Section 9.2* and *9.3*), one ADC, one OPAMP, one comparator, and one DAC channel must be assigned.

- If “Embedded PGA” is enabled, the selection of ADC peripheral (and input pin) is linked to that of PGA peripheral.
- If “Embedded HW OCP” and “Embedded PGA” are enabled, the selection of ADC and comparator (and their input and ‘+’ pins) is linked to that of PGA peripheral (and its '+' input).
- If “Embedded HW OCP” is enabled and “Embedded PGA” is disabled, the selection of comparator is free.
- If “Embedded HW OCP” and “Embedded PGA” are both disabled, the selection of comparator and ADC is free.
- If both PGA and comparator for OC protection are used they will share the same input pins for the motor current measurement signal.

Three shunts topology: according to configuration (as explained in *Section 9.2* and *9.3*), two ADCs, two OPAMPs, three comparators, and one DAC channel must be assigned.

- If “Embedded PGA” is enabled, the selection of ADC peripherals (and input pins) is linked to that of PGA peripherals.
- If “Embedded HW OCP” and “Embedded PGA” are enabled, the selection of ADCs and comparators (and their inputs and ‘+’ pins) is linked to that of PGA peripherals (and theirs '+' inputs).
- If “Embedded HW OCP” is enabled and “Embedded PGA” is disabled, the selection of comparators is free.
- If “Embedded HW OCP” and “Embedded PGA” are both disabled, the selection of comparators and ADCs is free.

The pair OPAMP1/OPAMP2 can be used in a project based on STM32F302 or STM32F303; the pair OPAMP3/OPAMP4 can be used additionally in a project based on STM32F303.

The pair ADC1/ADC2 can be used in a project based on STM32F302 or STM32F303; the pair ADC3/ADC4 can be used additionally in a project based on STM32F303.

If both PGA and comparator for OC protection are used they will share the same input pins for the motor current measurement signal.

9.5 Resources allocation - dual drive

If using a STM32F303 microcontroller is possible to create a dual drive project. The current feedback network configurations supported by STM32 FOC SDK are single shunt and three shunt.

Dual single shunt drive, dual three shunts drive and mixed “single plus three” shunts drives are allowed.

The sharing of peripherals between “single shunt drive” and “three shunt drive” is not allowed.

The sharing of peripherals between two “single shunt drive” is not allowed.

The sharing of peripherals between two “three shunt drive” is allowed, in the forms expressed below.

Single shunt topology: for each motor, according to configuration (as explained *Section 9.2* and *9.3*), one ADC, one OPAMP and one Comparator must be assigned.

- If “Embedded PGA” is enabled, the selection of ADC peripheral (and input pin) is linked to that of PGA peripheral.
- If “Embedded HW OCP” and “Embedded PGA” are enabled, the selection of ADC and comparator (and their input and ‘+’ pins) is linked to that of PGA peripheral (and its '+' input).
- If “Embedded HW OCP” is enabled and “Embedded PGA” is disabled, the selection of comparator is free.
- If “Embedded HW OCP” and “Embedded PGA” are both disabled, the selection of comparator and ADC is free.
- If both PGA and comparator for OC protection are used they will share the same input pins for the motor current measurement signal.

Three shunts topology mixed with Single shunt topology: according to configuration (as explained in *Section 9.2* and *9.3*), two ADCs, two OPAMPs, three Comparators, and one DAC channel must be assigned.

- If “Embedded PGA” is enabled, the selection of ADC peripherals (and input pins) is linked to that of PGA peripherals.
- If “Embedded HW OCP” and “Embedded PGA” are enabled, the selection of ADCs and comparators (and their inputs and ‘+’ pins) is linked to that of PGA peripherals (and theirs '+' inputs).
- If “Embedded HW OCP” is enabled and “Embedded PGA” is disabled, the selection of comparators is free.
- If “Embedded HW OCP” and “Embedded PGA” are both disabled, the selection of comparators and ADCs is free.
- The pair OPAMP1/OPAMP2 can be used in a project based on STM32F302 or STM32F303; the pair OPAMP3/OPAMP4 can be used additionally in a project based on STM32F303.
- The pair ADC1/ADC2 can be used in a project based on STM32F302 or STM32F303; the pair ADC3/ADC4 can be used additionally in a project based on STM32F303.
- If both PGA and comparator for OC protection are used they will share the same input pins for the motor current measurement signal.

Dual Three shunts topology, not shared resources: according to configuration (as explained in *Section 9.2* and *9.3*), four ADCs, four OPAMPs, six comparators, and two DAC channels must be assigned.

- If “Embedded PGA” is enabled, the selection of ADC peripherals (and input pins) is linked to that of PGA peripherals.
- If “Embedded HW OCP” and “Embedded PGA” are enabled, the selection of ADCs and comparators (and their inputs and ‘+’ pins) is linked to that of PGA peripherals (and theirs '+' inputs).
- If “Embedded HW OCP” is enabled and “Embedded PGA” is disabled, the selection of comparators is free.
- If “Embedded HW OCP” and “Embedded PGA” are both disabled, the selection of comparators and ADCs is free.

The pairs that can be used are:

- OPAMP1/OPAMP2 can be used in a project based on STM32F302 or STM32F303;
- OPAMP3/OPAMP4 can be used additionally in a project based on STM32F303.
- ADC1/ADC2 can be used in a project based on STM32F302 or STM32F303;
- ADC3/ADC4 can be used additionally in a project based on STM32F303.

If both PGA and comparator for OC protection are used they will share the same input pins for the motor current measurement signal.

Dual Three shunts topology, shared resources: If both drives are Three shunts it can be possible to share the ADC and/or the PGA to perform the motor current measurement. To do this, it is mandatory to have both use external operational amplifier or both use the embedded PGA for the motor current measurement signals amplification. It can be settled by the user in the ST MC Workbench clicking the “Shared resource” check box in the Control Stage -> Analog Input.

If shared resource is settled and external operational amplifier is used, it is possible to use the pairs ADC1/ADC2 or the pairs ADC3/ADC4 for both drivers. In this case ST MC Workbench proposes the allowed inputs pins for motor currents measurement.

If shared resource is settled and embedded PGA is used, the following configuration is used:

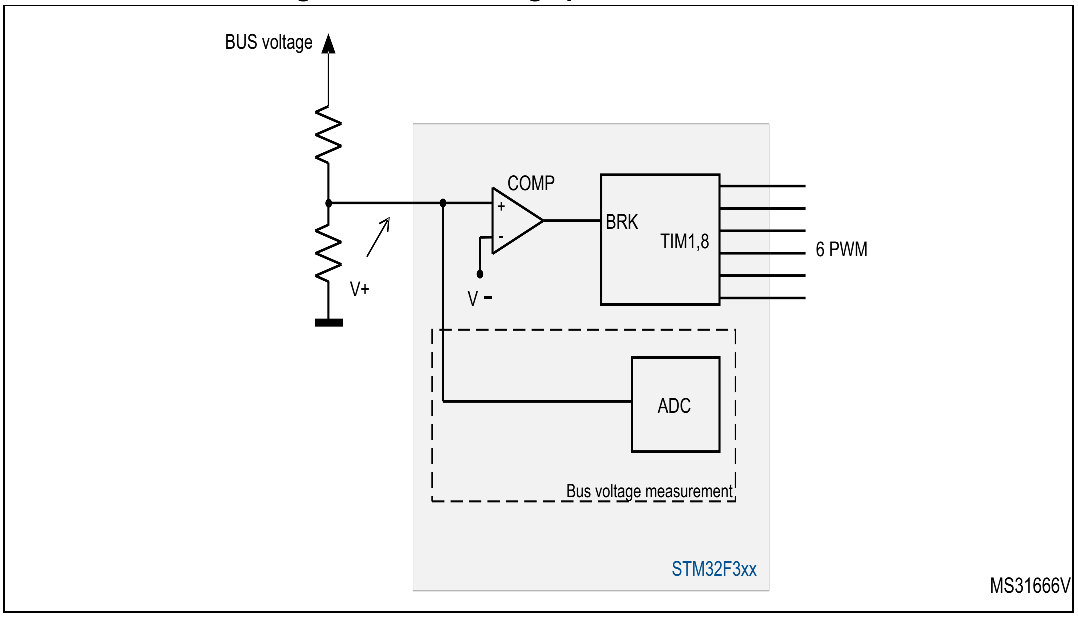
- The pair OPAMP1/OPAMP3 is used,
- OPAMP gains is only “Internal”,
- External capacitor filer is not allowed,
- Input pins are: PA5, PA7, and PB13 respectively U, V, W for motor 1 and PA1, PA3 and PB0 respectively U, V, W for motor 2.

In this case, if the hardware over current protection is managed by internal comparators, it is mandatory to connect externally the pins PA3 with one of the pins PB14 or PD14 and connect externally the pins PA5 with one of the pins PB11 or PD11. The pins selected can be settled in the workbench in Control Stage -> Analog Input -> Phase current feedback -> Protection.

10 Overvoltage protection with embedded analog (STM32F3x)

Figure 28. shows a basic implementation of over-voltage protection network that can be implemented using the internal resources of the STM32F30x.

Figure 28. Overvoltage protection network



In this case, the principle is similar to the one described in *Section 9.3*:

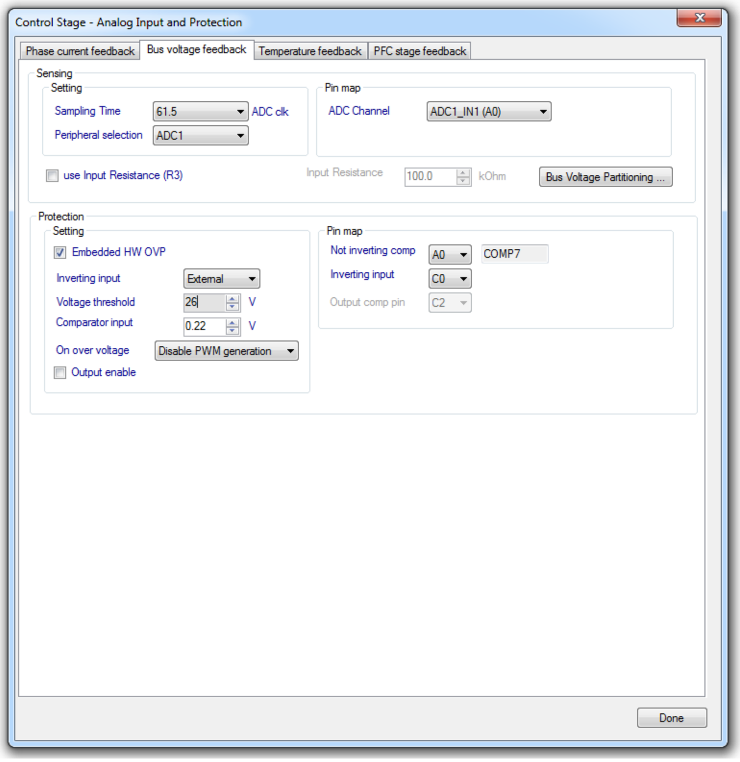
- A resistive voltage divider provides a signal proportional to the bus voltage. It must be sized depending on the bus voltage range requested by the target application, so that is never exceeds the MCU's input maximum admissible voltage level.
- This reading is compared to an over-voltage threshold to generate a fault signal.
- If the threshold is exceeded, a break signal stops the PWM generation putting the system in a safe state.

As mentioned before, these actions can be performed automatically using the internal comparator of the STM32F30x. In this case, it is convenient to use the second break functionality (BRK) of the advanced timer in order to differentiate the action to perform on the PWM signals in case of an over-current: disable PMW generation or turn-on low side switches.

The MC library can be arranged to match these configurations by using the ST MC Workbench, to create a project based on STM32F302 or STM32F303. From the dialogue window located in Control Stage -> Analog Input -> Bus voltage feedback (*Figure 29.*), setting:

– “Embedded HW OVP” checkbox;
– HW OVP internal threshold: selecting “Internal” in the “Inverting input” drop down list and choosing the internal voltage reference (among available values) in “Comparator Input”.
– HW OVP external threshold: selecting “External” in the “Inverting input” drop down list and editing the external voltage reference in “Comparator Input”.
– HW OVP internal threshold using DAC: selecting “DAC” in the “Inverting input” drop down list and editing the DAC voltage reference to be generated in “Comparator Input”. A DAC channel must be assigned for this functionality (OVP) from the related dialogue window located in Control stage -> DAC functionality (*Figure 30.*)
– The selection of 'not-inverting' input pin contextually picks the comparator to be used.
– The drive behavior when an overvoltage state is found: disable PWM generation, or turn-on low side switches;
– Enabling or disabling the comparator output has no effect on the overvoltage protection functionality itself.

Figure 29. STMCWB window related to ADC/COMP settings for DC bus voltage


Control Stage - Analog Input and Protection

Phase current feedback | Bus voltage feedback | Temperature feedback | PFC stage feedback

Sensing

Setting
Sampling Time: 61.5 ADC clk
Peripheral selection: ADC1

Pin map
ADC Channel: ADC1_IN1 (A0)

☐ use Input Resistance (R3)
Input Resistance: 100.0 kOhm
Bus Voltage Partitioning ...

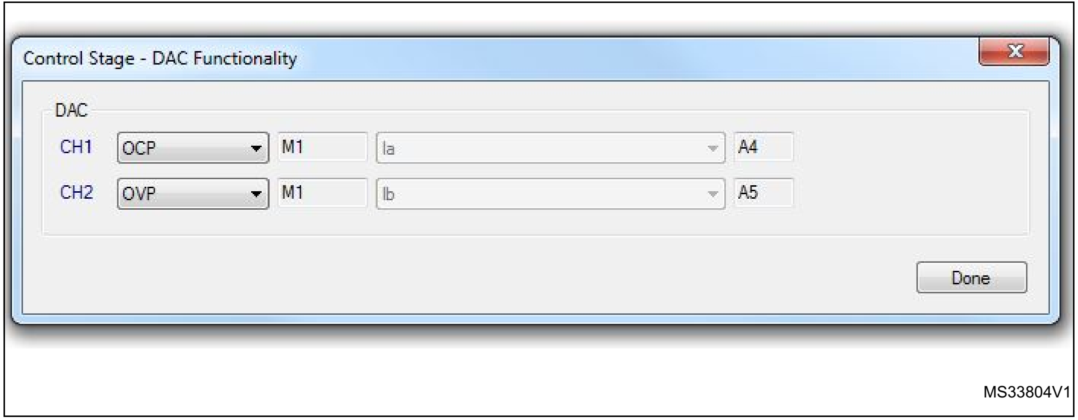
Protection

Setting
☑ Embedded HW OVP
Inverting input: External
Voltage threshold: 26 V
Comparator input: 0.22 V
On over voltage: Disable PWM generation
☐ Output enable

Pin map
Not inverting comp: A0 COMP7
Inverting input: C0
Output comp pin: C2

Done

Figure 30. STMCWB window related to DAC settings



Control Stage - DAC Functionality

DAC

CH1	OCP	M1	Ia	A4
CH2	OVP	M1	Ib	A5

11 Revision history

Table 9. Document revision history

Date	Revision	Changes
03-May-2011	1	Initial release.
14-Nov-2012	2	Added references to STM32F05xx, STM32F2xx and STM32F4xx in the title and throughout the document. Updated SDK V3.0 to V3.4. Updated Motor Control Library project and User project in *Section 5.1: Motor control workspace*. Updated *Section 5.2: MC SDK customization process*, including *Table 1: Project configurations*. Added Option 1 to *Section 6: How to download the full LCD user interface* and updated Option 2 and Option 3. Added text to *Section 7.2: LCD User interface structure*.
16-Dec-2013	3	Changed the software library version (from V3.3 to v3.4). Added references to STM32F303xB/C in the title and throughout the document. Added new available ST documents in *Section : Available from www.st.com or your STMicroelectronics sales office* Updated option in *Section 6: How to download the full LCD user interface*. Updated *Section 5.2: MC SDK customization process*. Added *Section 9: Current sensing and protection with embedded analog (STM32F3x)*. Added *Section 10: Overvoltage protection with embedded analog (STM32F3x)*. Updated: *Figure 15*, *Figure 17*, *Figure 18*, *Figure 19*, *Figure 20*, *Figure 21*, *Figure 27*, *Figure 29*, *Figure 30*
22-May-2014	4	Changed software library version from v3.4 to v4.0. Removed Table1 Applicable product. Added text in *Section 1: Motor control library features*. Modified *Section 1.1: User project and interface features*. Modified *Section 5.1: Motor control workspace*. Updated *Figure 7: IAR EWARM IDE Workspace overview* Added *Figure 8: Keil uVision workspace overview* Updated *Figure 9: Workspace batch build for IAR EWARM IDE* Added *Figure 10: Workspace batch build for Keil uVision* Updated *Figure 14: LCD UI project*, *Figure 15: User interface reference*, *Figure 17: STM32 Motor Control demonstration project welcome message* and *Figure 19: Dual control panel page*

Table 9. Document revision history (continued)

Date	Revision	Changes
18-May-2015	5	Updated *Introduction*; Updated *Section 1: Motor control library features* Updated *Section 2.2: Related documents* Added *Section 3: Motor profiler and One Touch Tuning* Added *Section 4: On-the-fly Sensorless startup* Added figure from *Figure 1: Motor profiler* to *Figure 6: Enabling “On-the-fly” start-up with Advanced profile* Updated *Figure 29: STMCWB window related to ADC/COMP settings for DC bus voltage*
11-Sept-2015	6	Updated – Document title – *Introduction* – *Table 1.: Project configurations* – *Section 2.2: Related documents*