



Introduction

This application note provides information to ease the use and customization of the STM32 PMSM Field Oriented Control (FOC) SDK V3.0.

A complete documentation list is provided in [Section 2: Documentation architecture](#). This is included in the software package or available on the ST web site.

[Section 3: Working environment and its customization](#) explains the Motor Control workspace, its customization and download.

[Section 4: How to download the 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 is to be done just once on new evaluation boards) in the microcontroller Flash memory to STM32 evaluation boards fitted with an LCD display.

[Section 5: LCD user interface](#) explores the menu screens and controls.

[Section 6: Introduction to the PMSM FOC drive](#) provides the block diagram of the implemented Field Oriented Control.

Contents

1	Motor control library features	3
1.1	User project and interface features	4
2	Documentation architecture	5
2.1	Where to find the information you need	5
2.2	Related documents	6
3	Working environment and its customization	7
3.1	Motor control workspace	7
3.2	MC SDK customization process	8
4	How to download the LCD user interface	10
5	LCD user interface	13
5.1	Running the motor control firmware using the LCD interface	13
5.2	LCD User interface structure	14
5.2.1	Welcome message	16
5.2.2	Configuration and debug page	16
5.2.3	Dual control panel page	18
5.2.4	Speed controller page	20
5.2.5	Current controllers page	22
6	Introduction to the PMSM FOC drive	24
7	Revision history	26

1 Motor control library features

- Single or simultaneous Dual PMSM FOC sensorless / sensed (Dual PMSM FOC only when running on STM32F103xx High-Density, STM32F103xx XL-Density)
- Speed feedbacks:
 - 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)
 - 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)
- 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)
- 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 STM32 performance line only)
- SysTick-timer-easy-scheduler-based user project

Available User Interface options (and combinations of them):

- LCD (C++ programmed) plus joystick
- Serial communication protocol
- Drive system variables logging/displaying via:
 - SPI
 - DAC (DAC peripheral available only on STM32F100xx or STM32F103xx High-Density and XL-Density; RC-filtered PWM signal option where not 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:

- STM32F103xx/STM32F100xx permanent-magnet synchronous motor single/dual FOC software library V3.0 (UM1052) provides the following:
 - Features
 - Architecture
 - Workspace
 - Customization processes
 - Overview of algorithms implemented (FOC, current sensors, speed sensors)
 - MC API
 - Demonstrative user project
 - Demonstrative LCD user interface
 - Demonstrative serial communication protocol
- *Advanced developers guide for STM32F103xx/STM32F100xx PMSM single/dual FOC library* (UM1053). This provides the following:
 - 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 html 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.
- STM32F10x 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™-M3 Technical Reference Manual, available from:
http://infocenter.arm.com/help/topic/com.arm.doc.ddi0337e/DDI0337E_cortex_m3_r1p1_trm.pdf

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

- *STM32F103xx datasheet*
- *STM32F100xx datasheet*
- *STM32F103xx user manual (RM0008)*
- *STM32F100xx user manual (RM0041)*
- *STM32F103xx AC induction motor IFOC software library V2.0 (UM0483)*
- *STM32 and STM8 Flash Loader demonstrator (UM0462)*

3 Working environment and its customization

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

- PC
- A third-party integrated development environment (IDE)
- A third-party C-compiler
- JTAG/SWD interface for debugging and programming
- Application board with an STM32F103xx/STM32F100xx 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.
- Three-phase PMSM motor

3.1 Motor control workspace

The Motor Control workspace is composed of three projects (as shown in [Figure 3](#)), which constitute the MC workspace.

Motor Control Library project: the collection of all the classes (37 among base and derivative classes) developed to implement all the features. It is built as a compiled library, not as an executable file.

Motor Control Application project: the application that uses the Motor Control Library layer. Parameters and configurations related to user's application are used here to create right objects.

The Motor Control API is the set of commands granted to the upper layer. This project is built as a compiled library, not as an executable file.

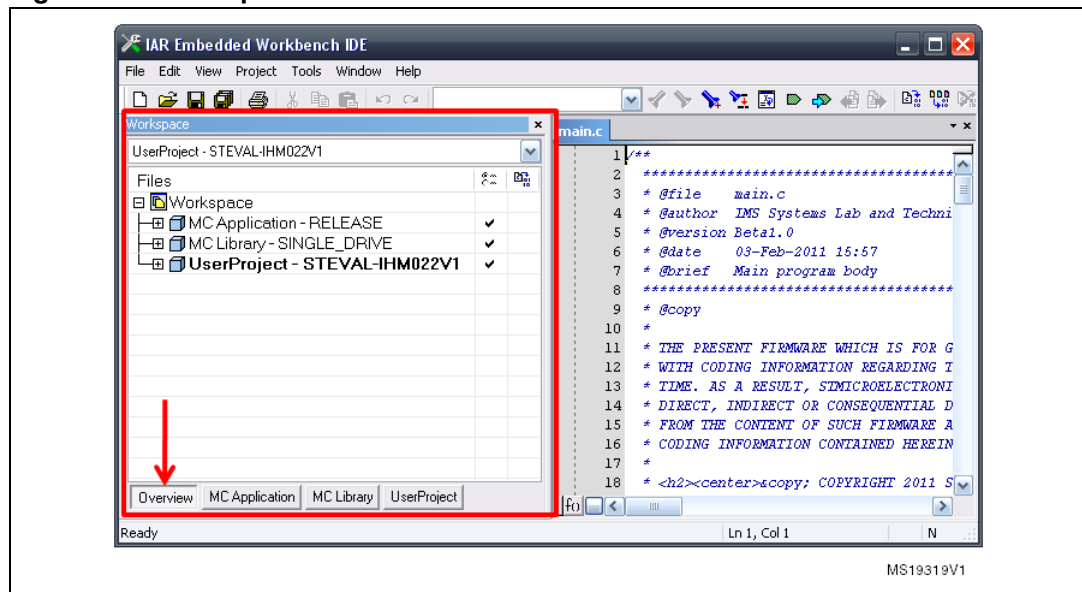
User project: the demonstration program included in the SDK that makes use of the Motor Control Application through its MC API and provides it the required clock frequencies and access to Interrupt Handlers. The program can run some useful functions (depending on user options), such as serial communication, LCD/keys interface, system variables displaying through DAC.

Two equivalent and alternative 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.

Previously, 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 1](#) provides an overview of the IAR EWARM IDE workspace (located in Installation folder `\Project\EWARM\Workspace.eww`) configured for dual FOC drive. The following sections provide details on this. The equivalent workspace based on FreeRTOS is located in Installation folder `\FreeRTOS Project\EWARM\RTOS_Workspace.eww`.

Figure 1. Workspace overview

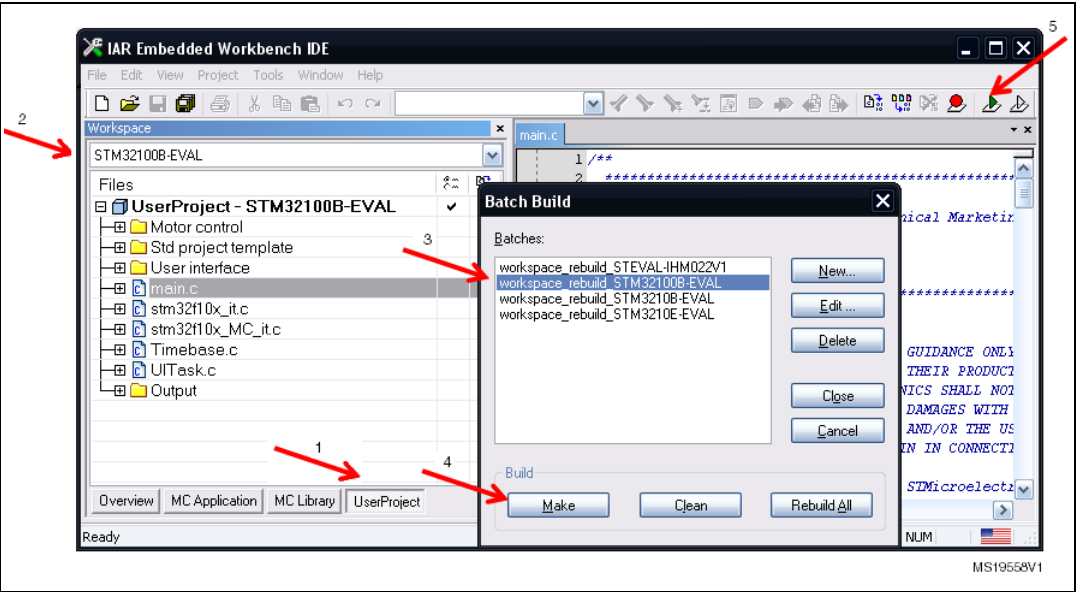


3.2 MC SDK customization process

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

1. Using the ST MC Workbench GUI, enter the page information to reflect the system configuration and parameters. This part of the process ends by generating the .h parameters in the correct directory (Installation folder\System & Drive Params).
2. If the system is configured to enable the LCD User Interface, download the specific firmware. See [Section 4: How to download the LCD user interface](#) (this step is to be done only once).
3. Open the MC workspace of choice:
 - FreeRTOS based: Installation folder\FreeRTOS
Project\EWARM\RTOS_Workspace.eww
 - Non-FreeRTOS: Installation folder\Project\EWARM\Workspace.eww
4. Enable the user project (callout 1 in [Figure 2: Workspace batch build](#)) and select the appropriate option from the combo-box (callout 2 in [Figure 2: Workspace batch build](#)). 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 2: Workspace batch build](#) appears.
6. Select a batch command (callout 3, [Figure 2](#)) as for step 4, then click the Make button to make the build (callout 4, [Figure 2](#)). If no error or relevant warning appears, download the firmware (callout 5, [Figure 2](#)) and do a test run.

Figure 2. Workspace batch build



It should be noted here that when system configuration or parameters are modified it may be necessary to rebuild all three projects or just one. The batch command conveniently builds all three, to avoid problems. This method is not time-consuming, from the compiler point of view, because if a project is not affected by the modification it is not recompiled. See [Figure 3](#).

Figure 3. Customization process

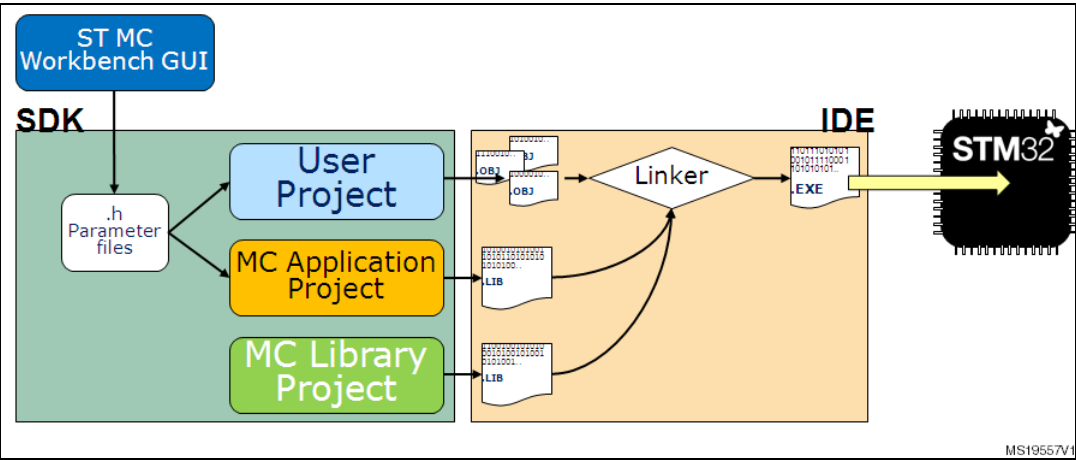


Table 1. Project configurations

STM32 device part, single/dual drive selection	Viable configuration among existing
STM32F103 low density/medium density	STM32F10B-EVAL
STM32F103 high density/XL density, Single motor drive	STM32F10E-EVAL
STM32F103 high density/XL density, Dual motor drive	STEVAL-IHM022V1
STM32F100 low / medium / high density	STM32F100B-EVAL

4 How to download the LCD user interface

When an STM32 evaluation board equipped with LCD (such as STM3210B-EVAL, STM3210E-EVAL, STM32100B-EVAL, STEVAL-IHM022V1) 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 5](#)). This option can be selected via a setting in the ST MC Workbench GUI.

In this case, the LCD UI software (single or dual drive configuration) is to be downloaded (following the procedure explained below) in the microcontroller in a reserved area, located at the end of 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 Motor Control Library V3.0 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 two procedures explained below to download the LCD UI.

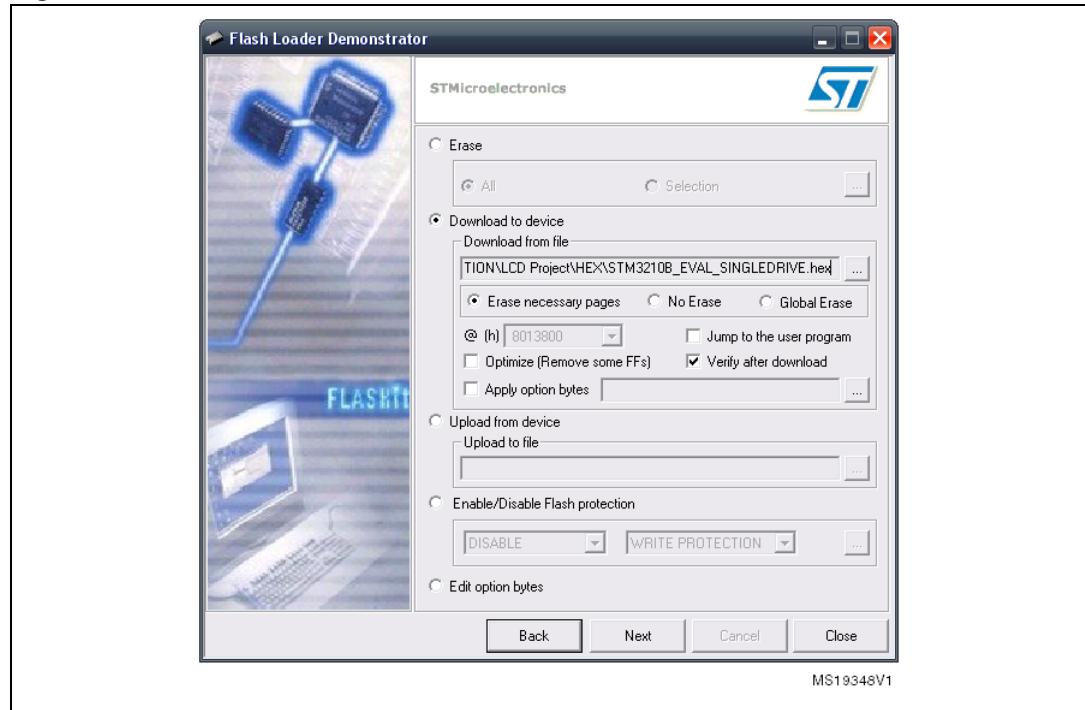
Option 1

This option is straightforward and the preferred one.

1. Use the STM32 and STM8 Flash loader demonstrator PC software package. This is available from the ST web site (http://www.st.com/internet/com/SOFTWARE_RESOURCES/SW_COMPONENT/SW_DEMO/um0462.zip) 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 4](#)) and browse to select the appropriate hexadecimal file (STM3210B_EVAL_SINGLEDRIIVE.hex, STM32100B_EVAL_SINGLEDRIIVE.hex, STM3210E_EVAL_SINGLEDRIIVE.hex, STEVAL_IHM022V1_SINGLEDRIIVE.hex, STEVAL_IHM022V1_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.
7. To test that the LCD UI is flashed correctly, for both option 1 and 2, open, build and download the user project (see [Section 3.2: MC SDK customization process](#)).

8. 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 inside UITask.c, line 133.

Figure 4. Flash loader wizard screen



Option 2

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 IAR EWARM IDE V5.50, open the workspace located in Installation folder\LCD Project\EWARM\UI Project.eww.

Figure 5. LCD UI project

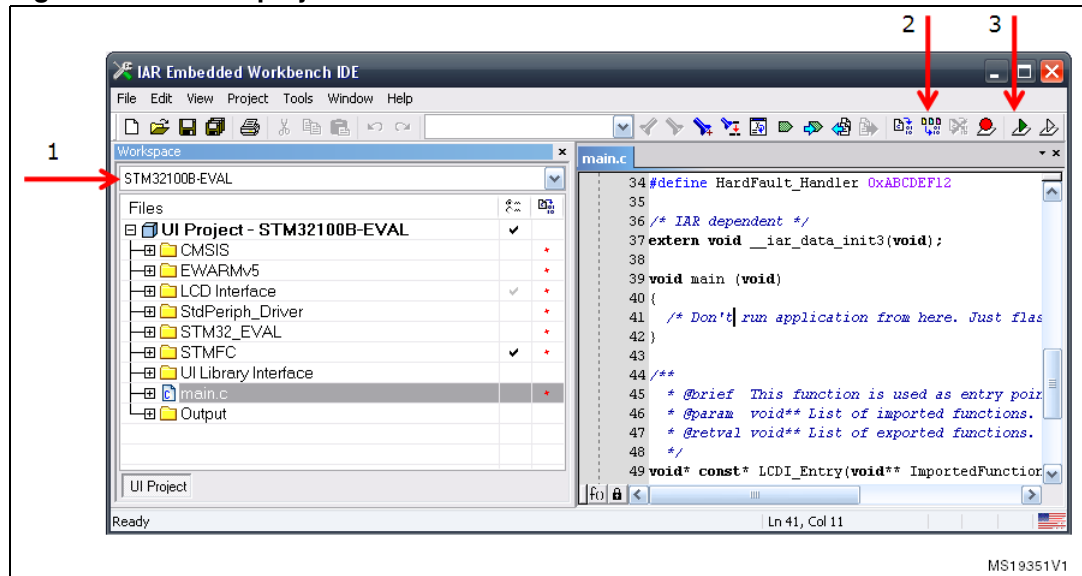


Figure 5 displays the logical arrangement of files (left-hand side) and actions that may be needed for set up and download.

Four project configurations are provided (callout 1, Figure 5), one for each STM32 evaluation board that has been tested with the MC SDK:

- STM32F10B-EVAL
- STM32F10E-EVAL
- STM32F100B-EVAL
- STEVAL-IHM022V1

This configuration affects LCD driver and linker file selection.

3. Build the project (callout 2, Figure 5), and download it to the microcontroller memory (callout 3, Figure 5).
4. To test that the LCD UI is flashed correctly, for both option 1 and 2, open, build and download the user project (see [Section 3.2: MC SDK customization process](#)).
5. 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 133.

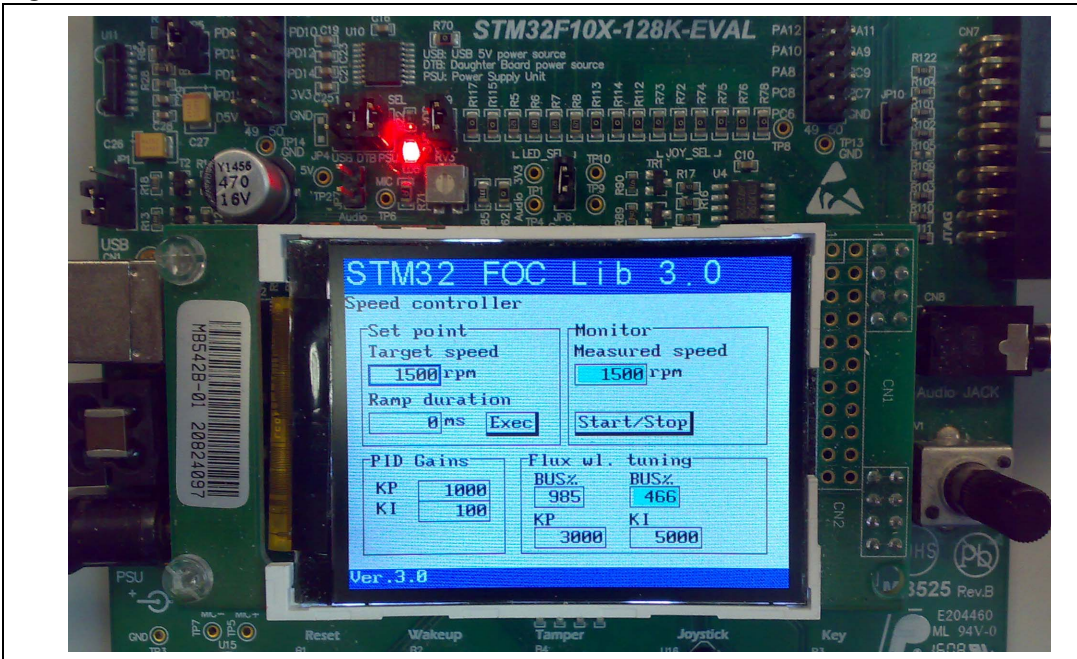
5 LCD user interface

5.1 Running the motor control firmware using the LCD interface

The STM32 motor control library (V3.0) 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 6](#).

Figure 6. User interface reference



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
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 LCD_JOYSTICK_BUTTON_FUNCTIONALITY DISABLE` (line 316) of the `Drive parameters.h` file.

5.2 LCD User interface structure

The demonstration program is based on circular navigation pages.

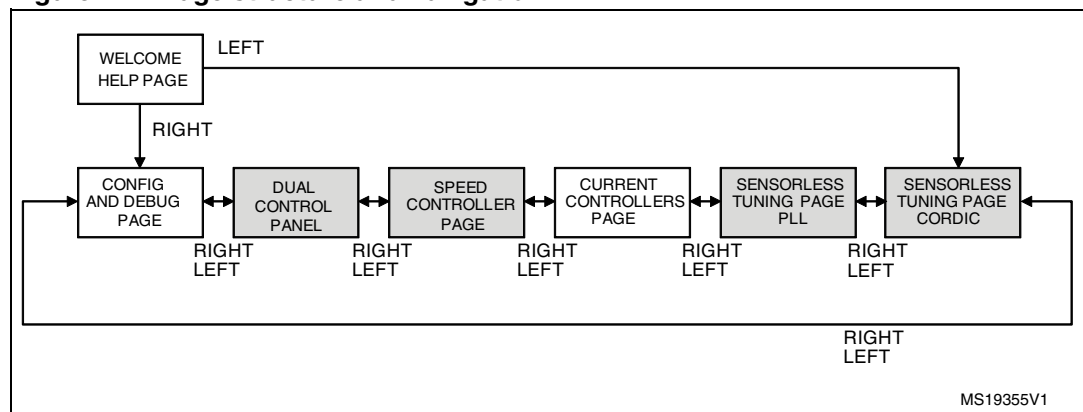
[Figure 7](#) shows the page structure. The visibility of certain pages shown in [Figure 7](#) depends on the firmware configuration:

- 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

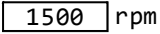
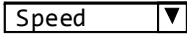
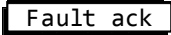
Figure 7. Page structure and navigation



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.

Complete documentation about this LCD User Interface can be found in *User manual STM32F103xx or STM32F100xx PMSM single/dual FOC SDK v3.0* (UM1052).

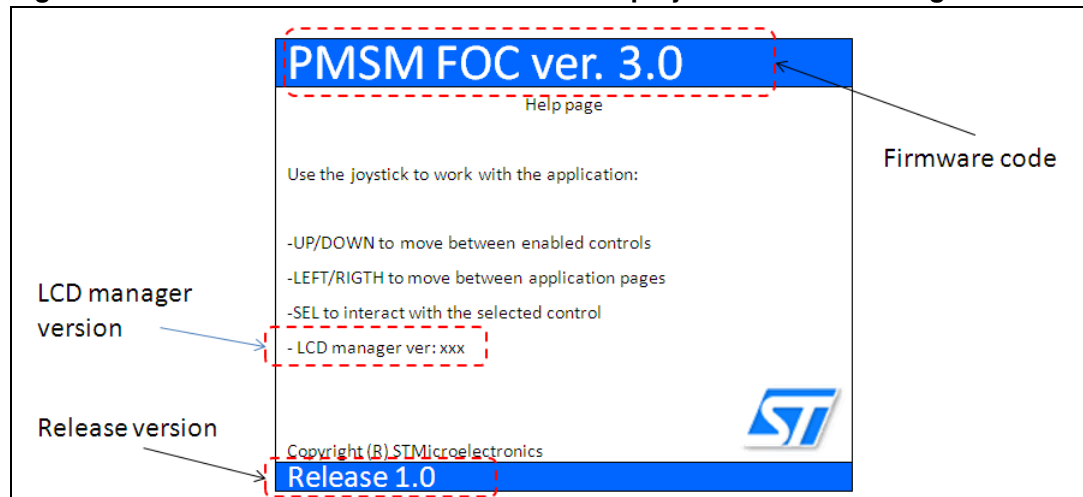
Table 3. List of controls used in the LCD demonstration program

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

5.2.1 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 8](#).

Figure 8. STM32 Motor Control demonstration project welcome message



5.2.2 Configuration and debug page

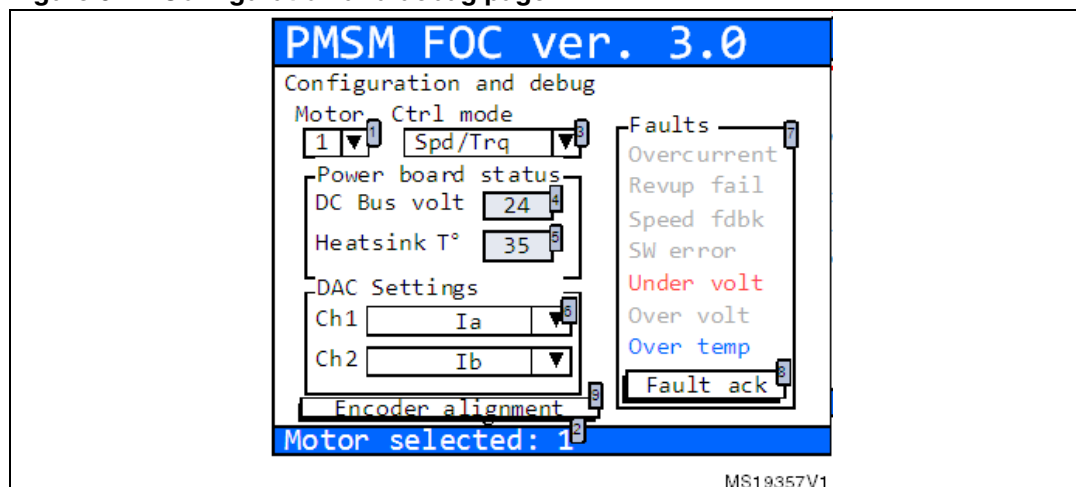
Press the joystick RIGHT from the welcome page to enter the configuration and debug page.

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

Use the configuration and debug page shown in [Figure 9](#) to:

- Select the active motor drive (field 1 in [Figure 9](#)). 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 (field 2 in [Figure 9](#)). Commands performed on, or feedback from a control are only relative to the active motor.
- Select the control mode (field 3 in [Figure 9](#)). 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 9. Configuration and debug page



- Read the DC bus voltage value (field 4 in [Figure 9](#)). This control is read-only.
- Read the heat sink temperature value (field 5 in [Figure 9](#)). This control is read-only.
- Select the variables to be put in output through DAC channels (field 6 in [Figure 9](#)). 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 (field 7 in [Figure 9](#)) 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 (field 8 in [Figure 9](#)). 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 (point 8 in [Figure 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 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 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 software detects a general fault condition. In 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 primary speed sensor or auxiliary speed sensor, the "encoder alignment" button is also present. In this case, the alignment of the encoder is required only once after each reset of the microcontroller.

5.2.3 Dual control panel page

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

To enter the dual control page, press the joystick RIGHT from configuration and debug page.

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

The dual control panel page shown [Figure 10](#) is used to send commands and get feedback from both motors. It is divided into three groups:

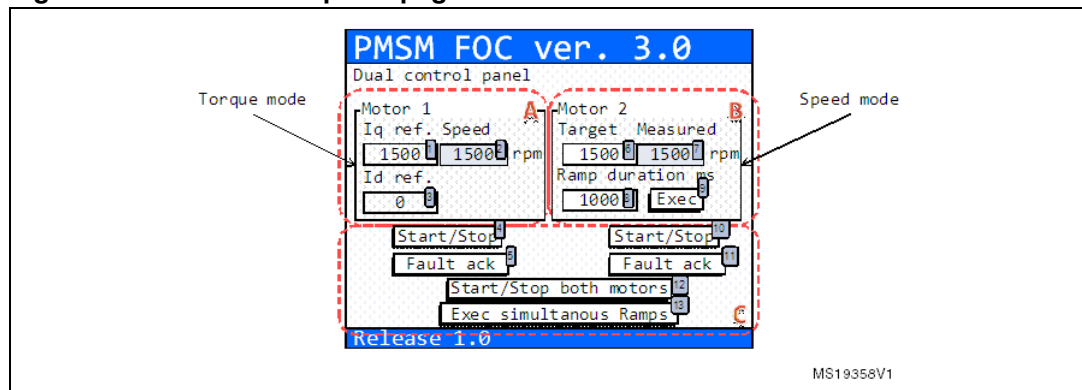
- Groups A and group B depend on speed/torque settings. The group content is updated on-the-fly when the control mode (torque/speed) is changed in the configuration and debug page. The control present in group A is related to the first motor. The control present in group B relates to the second motor.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

[Figure 10](#) 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 (field 1 in [Figure 10](#)). 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 10. Dual control panel page



- To set the I_d reference (field 3 in [Figure 10](#)). 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 expressed always as Cartesian coordinates (I_q, I_d).

Note: To convert current expressed in Amps to current expressed in digit is possible to use the formula:

$$\text{Current(s16A)} = [\text{Current(Amp)} * 65536 * R_{\text{shunt}} * A_{\text{op}}] / V_{\text{dd micro}}.$$

- Set the final motor speed of a speed ramp (field 6 in [Figure 10](#)). 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 (field 9 in [Figure 10](#)).
- Set the duration of a speed ramp (field 8 in [Figure 10](#)). This is related to motor 2. This control is only present if motor 2 is set in speed mode. 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 (field 9 in [Figure 10](#)). 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 (field 6 in [Figure 10](#)).
- Execute a speed ramp by pushing the “Exec” button (field 9 in [Figure 10](#)). 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 (field 6 and 8 in [Figure 10](#)). The Exec 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 is in RUN state.
- To read the motor speed (field 2 and 7 in [Figure 10](#) respectively for motor 1 and motor 2). Motor speed is expressed in RPM. This control is read-only.
- Send a start/stop command (field 4 for motor 1, field 10 for motor 2 in [Figure 10](#)). This is performed 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 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 ([Figure 10](#) field 5 for motor 1, field 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 (field 12 in [Figure 10](#)). This button is enabled only when the motors are both in Idle state or both in RUN state. If any of the motors are configured in speed mode when they start, a speed ramp with the last values of 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 (field 13 in [Figure 10](#)). 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 related final motor speed and 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 duration for each motor. The commands are buffered and take effect only when the related motor is in RUN state.

5.2.4 Speed controller page

This page is only present if the control mode set in Control mode (field 3 in [Figure 9](#)) is speed mode.

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

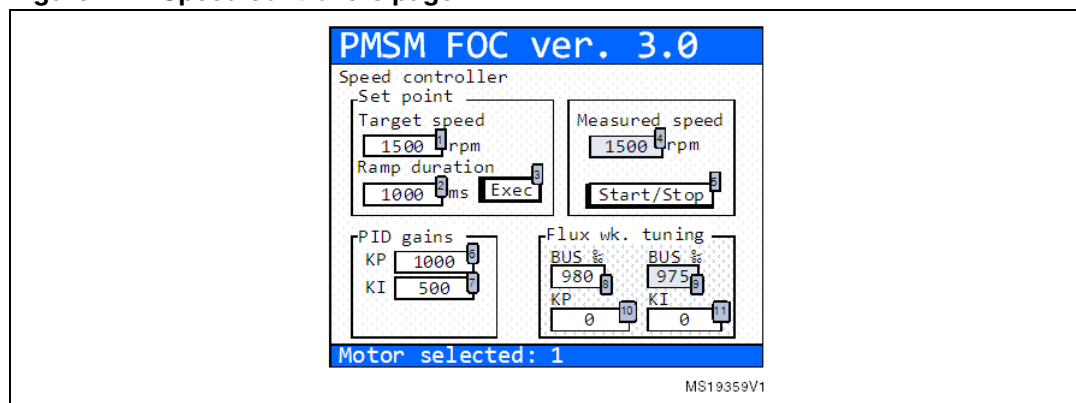
It is possible to navigate between focusable controls present in the page pressing joystick UP/DOWN.

The speed controller page shown in [Figure 11](#) is used to send commands and get feedback related to speed controller from the active motor. There are four groups of control 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, this provides for fast access to the measured speed and to the motor start/stop function

Figure 11. Speed controllers page



If the firmware is configured as dual motor drive, it is possible to know which is the active motor by reading the label at the bottom of the page. To change the active motor, change the motor field in the configuration and debug page (field 1, [Figure 11](#)).

[Table 6](#) lists the actions that can be performed using this page.

Table 6. Speed controllers page controls

Control	Description
Target speed (field 1 in Figure 11)	This sets the final motor speed of a speed ramp for the active motor. 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 Exec button description (field 3 in Figure 11)
Ramp duration (field 2 in Figure 11)	This sets the duration of a speed ramp for the active motor. Duration is expressed in milliseconds. The value set in this control is not automatically sent to motor control related object but it is used to perform a speed ramp execution. See description of “exec” button (field 3 in Figure 11). 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 (field 1 in Figure 11).
Exec button (field 3 in Figure 11)	This executes a speed ramp for the active motor. The execute speed ramp command is sent to the motor control related object together with final motor speed and duration presently selected (field 1 and 2 in Figure 11). 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 (field 4 in Figure 11)	This reads the motor speed for the active motor. Motor speed is expressed in RPM and is a read-only control.
Start/Stop button (field 5 in Figure 11)	This 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 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 (field 6 in Figure 11)	This sets the proportional coefficient of the speed controller for the active motor. The value set in this control is automatically sent to motor control related object, allowing the run-time tuning of the speed controller.

Table 6. Speed controllers page controls (continued)

Control	Description
Speed PID gain KI (field 7 in Figure 11)	This sets the integral coefficient of the speed controller for the active motor. The value set in this control is automatically sent to motor control related object, allowing the run-time tuning of speed controller.
Bus‰ (field 8 in Figure 11)	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‰ (field 9 in Figure 11)	DC bus voltage percentage presently used for the active motor and 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 (field 10 in Figure 11)	The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of flux weakening controller.
Flux wk PI gain KI (field 11 in Figure 11)	The value set in this control is automatically sent to the motor control related object, allowing the run-time tuning of flux weakening controller.

5.2.5 Current controllers page

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

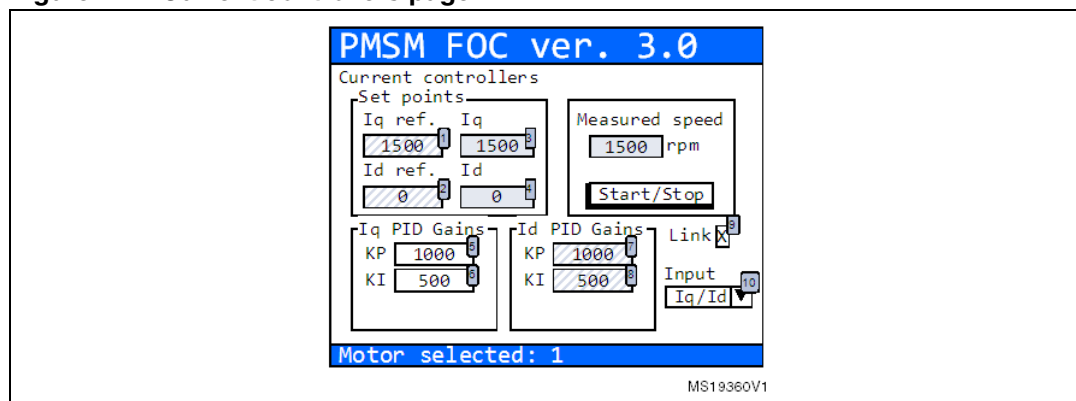
It is possible to navigate between focusable controls present in the page pressing joystick UP/DOWN.

The current controller page shown in [Figure 12](#) 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 measured currents
Iq PID gains	Used to change in real time the speed controllers gains
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, this provides for fast access to the measured speed and to the motor start/stop function

Figure 12. Current controllers page



If the firmware is configured as dual motor drive is possible to know which is the active motor reading the label at the bottom of the page. To change the active motor, the motor field in the configuration and debug page has to be changed ([Figure 12](#) field 1).

[Table 8](#) lists the actions that can be performed using this page.

Table 8. Current controllers page controls

Control	Description
I_q reference (field 1 in Figure 12)	To set and read 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: $\text{Current(s16A)} = [\text{Current(Amp)} * 65536 * \text{Rshunt} * \text{Aop}] / \text{Vdd micro}$
I_d reference (field 2 in Figure 12)	To set and read 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 (field 3 in Figure 12)	To read the measured I_q for the active motor. Measured I_q is expressed in s16A and is a read-only control.
I_q PI(D) gain, KP (field 5 in Figure 12)	To set 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 current controller.
I_q PI(D) gain, KI (field 6 in Figure 12)	To set the integral 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 current controller.
I_d PI(D) gain, KP (field 7 in Figure 12)	To set 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 current controller. This control is only read if the link check box is checked.
I_d PI(D) gain, KI (field 8 in Figure 12)	To set 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 current controller. This control is only read if the link check box is checked.

6 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 13. Basic FOC algorithm structure, torque control

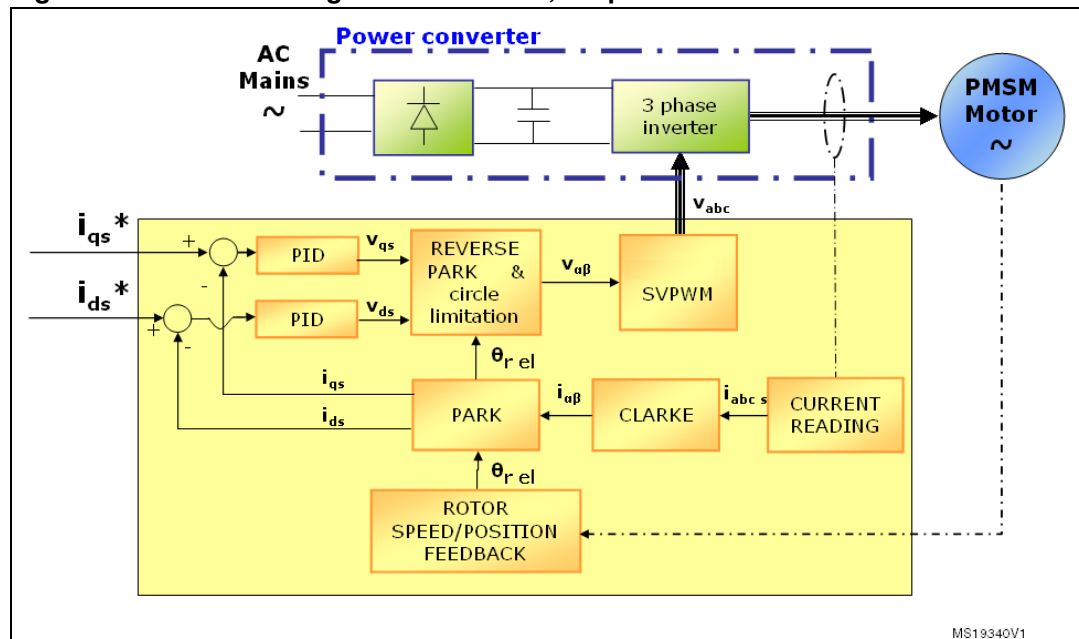
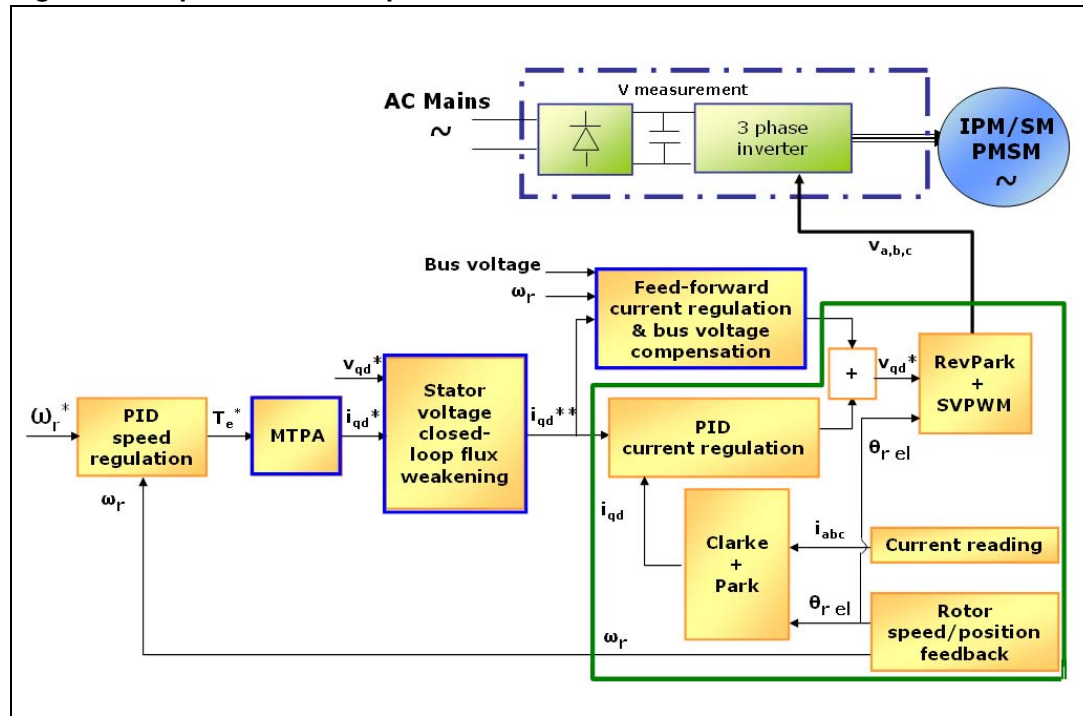


Figure 14. Speed control loop



7 Revision history

Table 9. Document revision history

Date	Revision	Changes
03-May-2011	1	Initial release.

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST's terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST'S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER'S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2011 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com

