Freescale Semiconductor, Inc.
User's Guide

Document Number: BLDCAPPACUG
Rev. 1, 11/2015

BLDC Motor Control Reference Solution Package

1. Introduction

This document provides instructions how to run and control a Brushless DC (BLDC) sensorless application with the Freescale Freedom and Tower System development boards shown in Table 1.

The required software, hardware setup, jumper settings, project arrangement, and user interface descriptions are outlined in the following chapters.

Contents

freescale™

Table 1. Supported development boards

		Platform		
		FRDM	**TWR**	**HVP**
Power Stage		FRDM-MC-LVBLDC	TWR-LV3PH	HVP-MC3PH
MCU	**KV10**	FRDM-KV10Z	TWR-KV10Z32	HVP-KV10Z
	KV11	—	TWR-KV11Z75M	—
	KV31	FRDM-KV31F	TWR-KV31F120	HVP-KV31F512
	KV46	—	TWR-KV46F150M	HVP-KV46F150

2. Hardware setup

The BLDC sensorless application runs on Freescale Tower and Freedom development platforms with a default 24 V Linix motor.

2.1. Linix 45ZWN24-40 motor

The BLDC Sensorless application uses the Linix 45ZWN24-40 motor described in the following table. The motor can be used with the Freescale Tower or Freedom development platforms.

Table 2. Linix 45ZWN24-40 motor parameters

Characteristic	Symbol	Value	Units
Rated Voltage	V_t	24	V
Rated Speed @ V_t	—	4000	RPM
Rated Torque	T	0.0924	Nm
Rated Power	P	40	W
Continuous Current	I_{cs}	2.34	A
Number of Pole Pairs	pp	2	—

Figure 1. Linix motor 45ZWN24-40

The motor has two types of cable connectors. One cable has three wires and is designated to power the motor. The second cable has five wires and is designated for Hall sensors signal sensing. For the BLDC sensorless application only the power input wires are needed.

2.2. Tower System

To run the BLDC application using the Freescale Tower development platform, the following Tower boards are required:

- Kinetis KV10Z Tower module, (for more information see the following website: TWR-KV10Z32), Kinetis KV11F75M Tower module (for more information see the following website: TWR-KV11F75M), or Kinetis KV31F Tower module (for more information see the following website: TWR-KV31F120M).
- Three-phase low voltage power module (for more information see the following website: TWR-MC-LV3PH) with included Linix motor.
- Tower elevator modules (for more information see the following website: TWR-ELEV).

2.2.1. TWR-MC-LV3PH module

The three-phase Low-Voltage Motor Control board (TWR-MC-LV3PH) is a peripheral Tower System Module, interchangeable across the Tower development platform. Phase voltage and current feedback signals are provided. These signals enable a variety of algorithms to control three-phase PMSM and BLDC motors. A high level of board protection (overcurrent, undervoltage, overtemperature) is provided by the MC33937 pre-driver. Before inserting the TWR-MC-LV3PH module into the Tower development platform, you must ensure the jumpers on your TWR-MC-LV3PH module are configured as follows:

Table 3. Jumper settings on TWR-MC-LV3PH

Jumper	Setting	Function
J2	1-2	Selects internal analog power supply
J3	1-2	Selects internal analog power reference (GND)
J10	2-3	Selects BEMF_SENSE_C
J11	2-3	Selects BEMF_SENSE_B
J12	2-3	Selects BEMF_SENSE_A
J13	2-3	Selects I_SENSE_DCB
J14	2-3	Selects V_SENSE_DCB_HALF

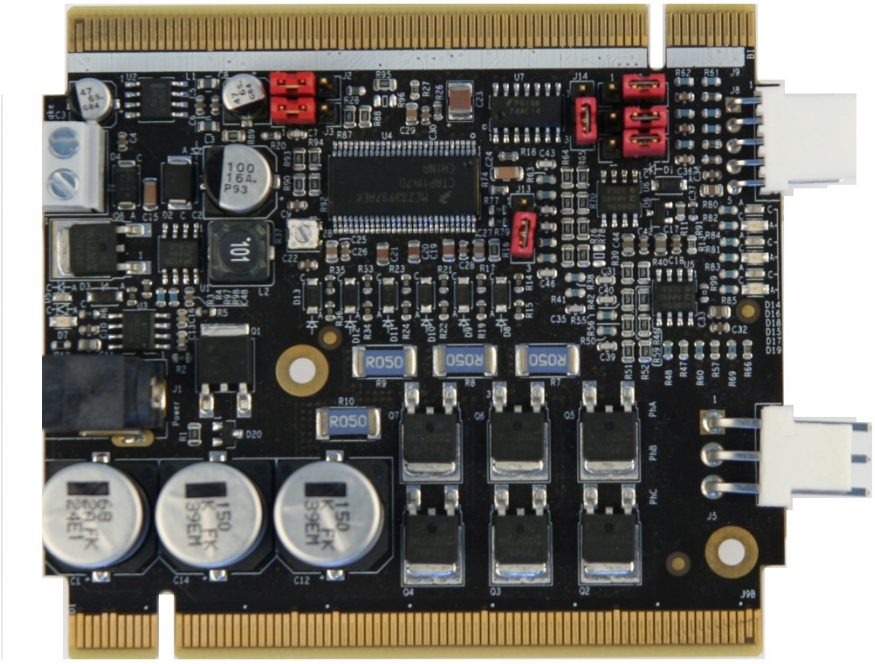

Figure 2. TWR-LV-MC3PH module's jumper setting for BLDC

2.2.2. TWR-KV10Z32 MCU module

The jumpers of the TWR-KV10Z32 MCU module and the TWR-MC-LV3PH module must be configured correctly. The following table lists the relevant jumpers and their settings for the TWR-KV10Z32 MCU module.

Table 4. TWR-KV10Z32 MCU module jumpers settings

Jumper	Setting	Function
J22	3-4	Selects UART1 for FreeMASTER connection through the OpenSDA terminal port
J21	3-4	Selects UART1 for FreeMASTER connection through the OpenSDA terminal port
J3	2-3	Connects PTB0 with on-board pushbutton SW2

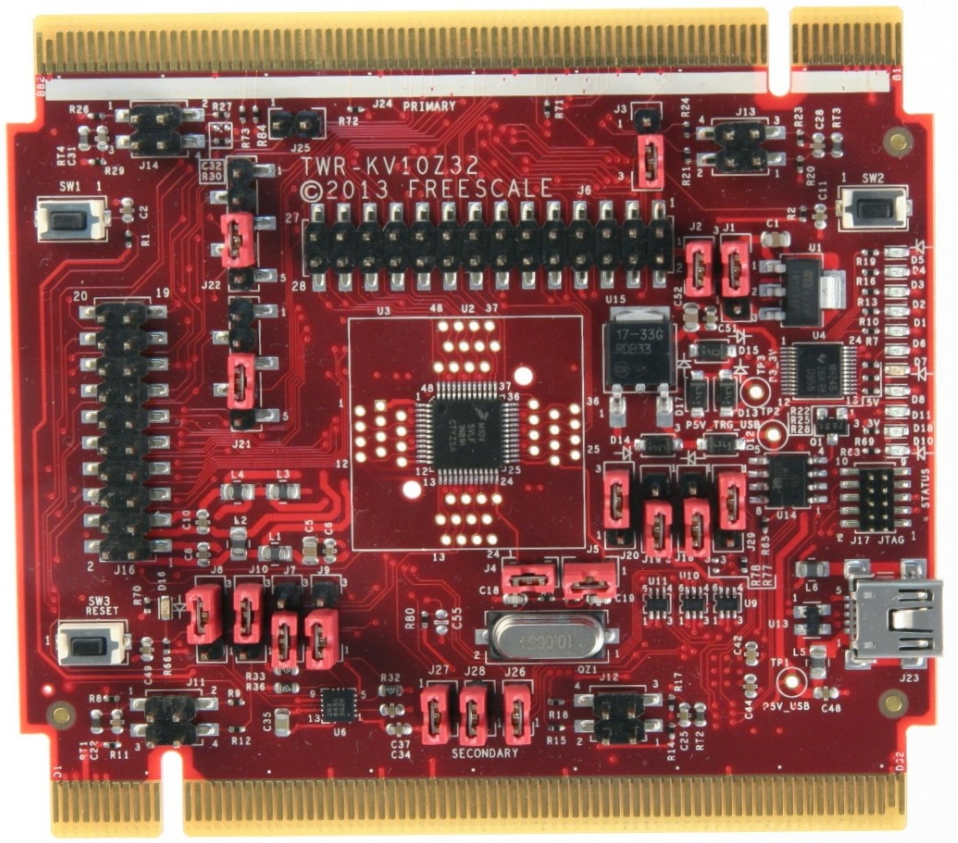

Figure 3. TWR-KV10Z32 MCU module

2.2.3. TWR-KV11Z75M MCU module

The jumpers of the TWR-KV11Z75M MCU module and the TWR-MC-LV3PH must be configured correctly. The following table lists the relevant jumpers and their settings for the TWR-KV11Z75M MCU module.

Table 5. TWR-KV11Z32 MCU module jumper settings

Jumper	Setting	Function
J505	2-3	Selects UART1 for FreeMASTER connection through the OpenSDA terminal port
J506	2-3	Selects UART1 for FreeMASTER connection through the OpenSDA terminal port
J3	2-3	Connects PTB0 with on-board pushbutton SW2

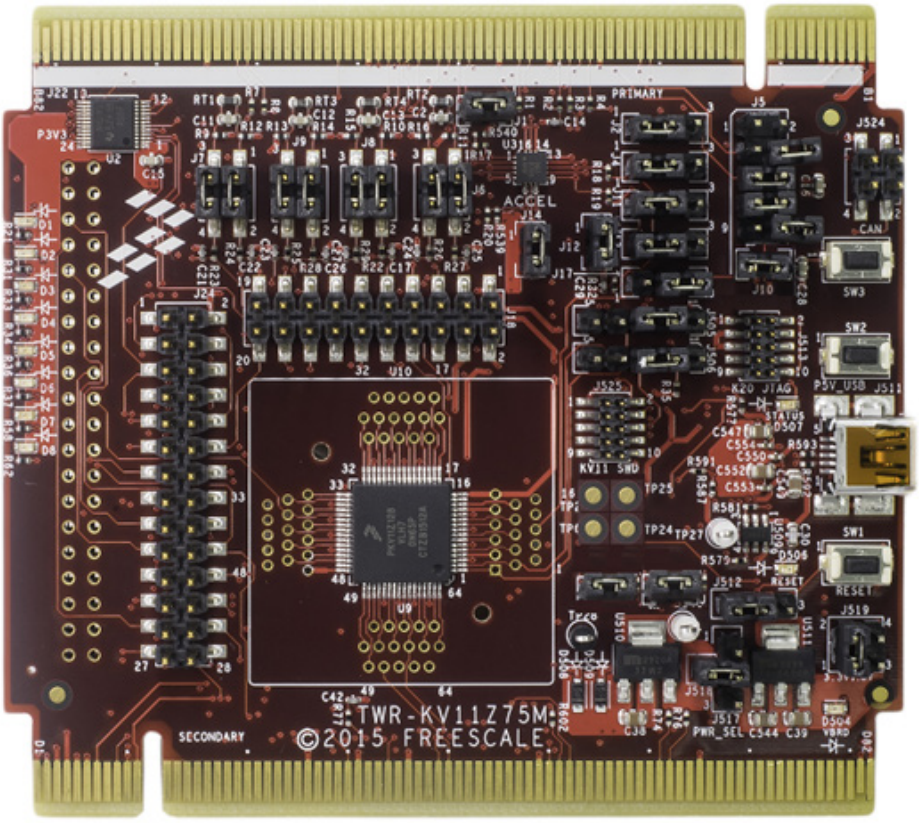

Figure 4. TWR-KV11Z75M MCU module

2.2.4. TWR-KV31F MCU module

The jumpers of the TWR-KV31F120 MCU module and the TWR-MC-LV3PH must be configured correctly. The following table lists the relevant jumpers and their settings for the TWR-KV31F120 MCU module.

Table 6. TWR-KV10Z32 MCU module jumpers settings

Jumper	Setting	Function
J22	2-3	Selects UART0 for FreeMASTER connection through the OpenSDA terminal port
J23	2-3	Selects UART0 for FreeMASTER connection through the OpenSDA terminal port

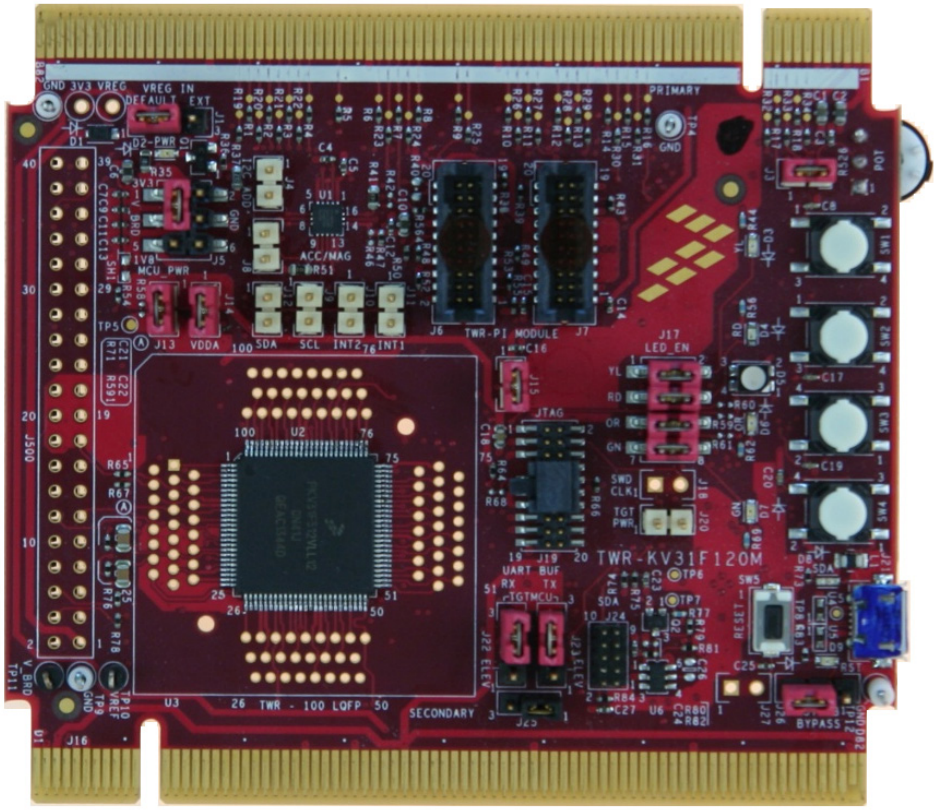

Figure 5. TWR-KV31F120M MCU module

2.2.5. TWR-KV46F150M MCU module

The jumpers on the TWR-KV46F150 MCU and TWR-MC-LV3PH modules must be configured correctly. The following table lists the relevant jumpers and their settings for the TWR-KV46F150 MCU module.

Table 7. TWR-KV10Z32 MCU module jumpers settings

Jumper	Setting	Function
J505	3-4	Selects UART1 for FreeMASTER connection through the OpenSDA terminal port
J506	3-4	Selects UART1 for FreeMASTER connection through the OpenSDA terminal port

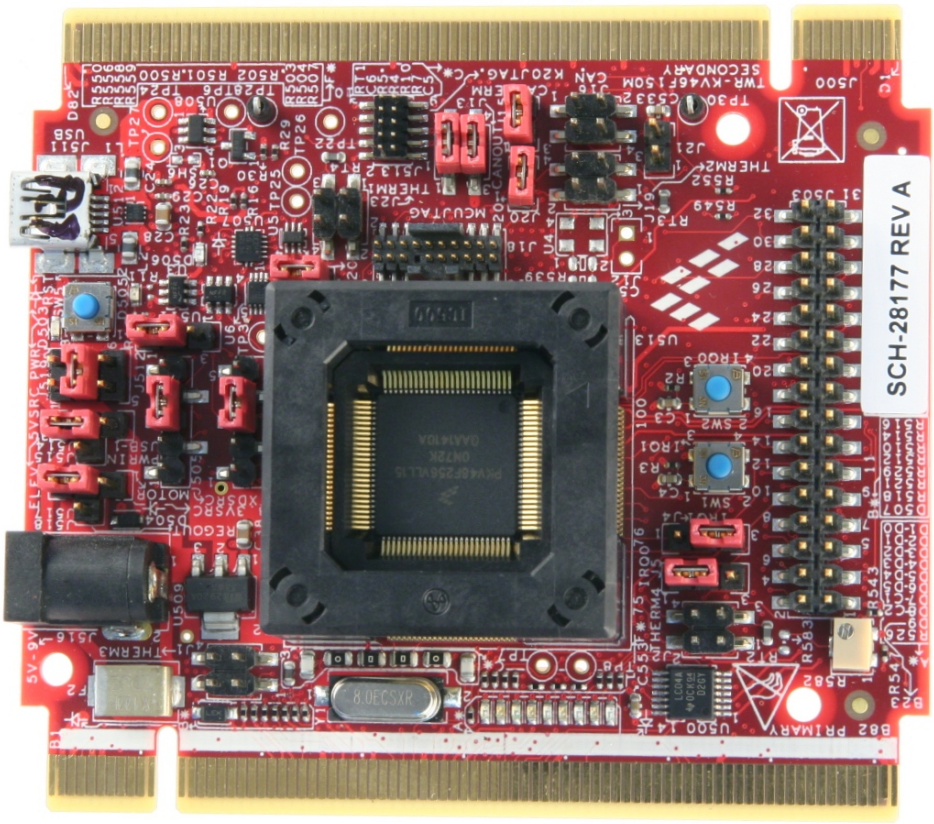

Figure 6. TWR-KV46F150M MCU module

2.3. Tower System assembly

1. Insert the TWR-KVxxXxx MCU module and the TWR-MC-LV3PH module into the TWR-ELEV card slots. Take care to ensure that the primary sides of the modules (marked by a white stripe) are inserted into the primary elevator card (marked by white connectors).
2. After you have completed assembly of the Tower System, connect the required cables as follows:
   - Connect the power input cable (three-wire connector) of the Linix motor to its corresponding connector (J5) on the TWR-MC-LV3PH motor control driver board.
   - Plug in the power supply cable that is attached to the TWR-MC-LV3PH system kit to the motor control peripheral board TWR-MC-LV3PH.
   - Connect the TWR MCU module to the host PC via a USB cable connected to J23 of the TWR-KV10Z32 MCU module or J21 of the TWR-KV31F120 MCU module and any USB port on the host PC.

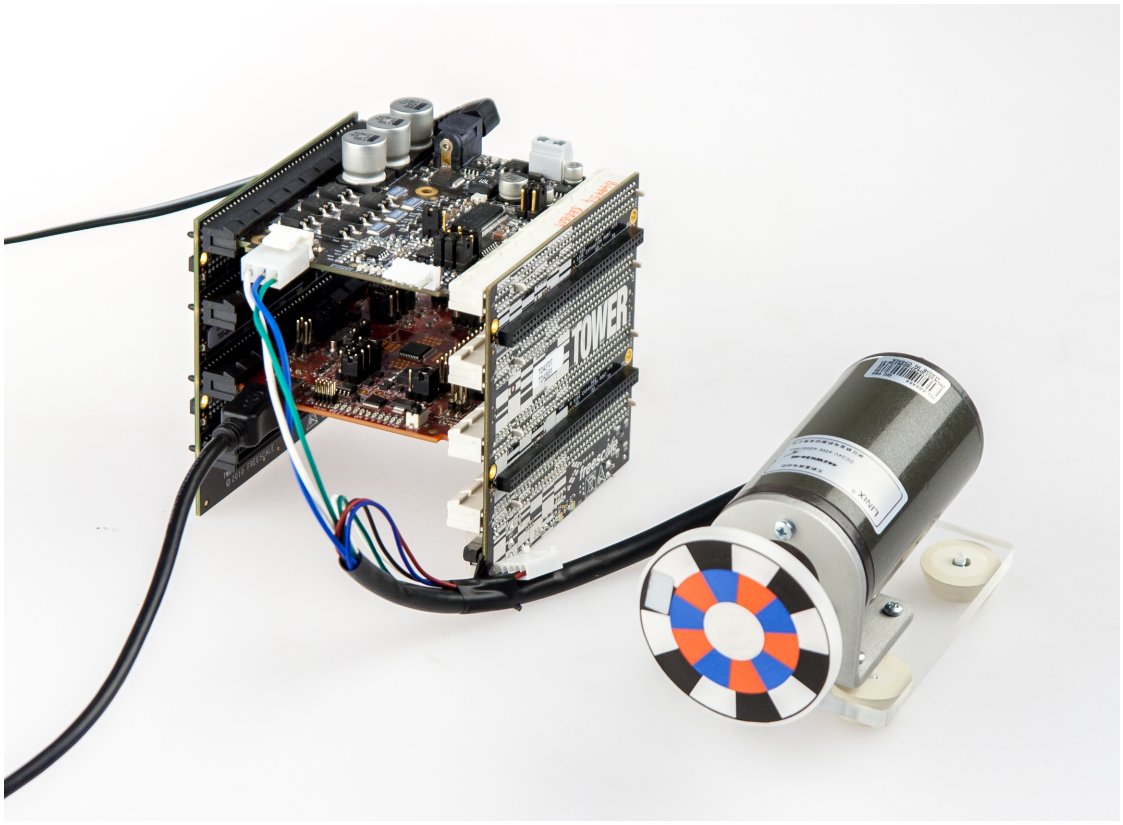

Figure 7. Assembled tower system

2.4. Freescale Freedom System

To run the BLDC application using the Freescale Freedom System, the following Freedom boards are required:

- Kinetis KV10Z Freedom board (see FRDM-KV10Z) or Kinetis KV31F Freedom board (see FRDM-KV31F)
- Three-phase low voltage power Freedom shield (see FRDM-MC-LV3PH) with included Linix motor.

2.4.1. FRDM-MC-LVBLDC low voltage evaluation board

The FRDM-MC-LVBLDC low voltage evaluation board, in a shield form factor, effectively turns a Freescale Freedom development platform into a complete motor control reference design, compatible with existing Freescale Freedom development platforms, FRDM-KV31F and FRDM-KV10Z.

The FRDM-MC-LVBLDC board does not require any hardware configuration or jumpers setting. It contains no jumpers.

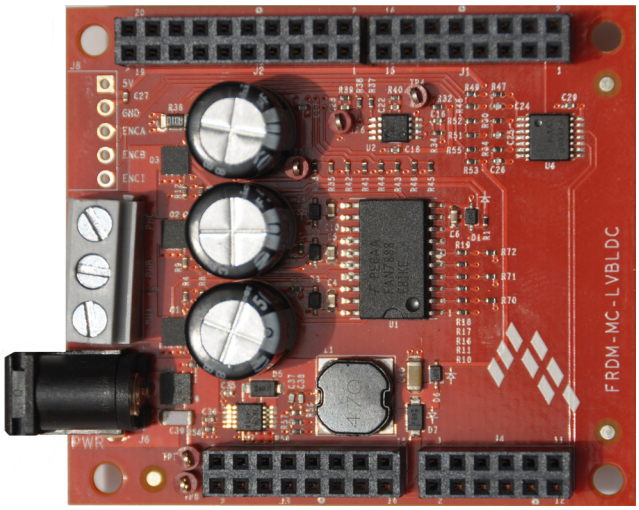

Figure 8. FRDM-MC-LVBLDC

2.4.2. FRDM-KV10Z board

The FRDM-KV10Z board is a low-cost development tool for the Kinetis V series KV1x MCU family built on the ARM® Cortex®-M0+ processor. The FRDM-KV10Z board's hardware is form-factor compatible with the Arduino™ R3 pin layout, providing a broad range of expansion board options. The FRDM-KV10Z platform features OpenSDA, the Freescale open source hardware embedded serial and debug adapter running an open source bootloader.

The FRDM-KV10Z board does not require any hardware configuration or jumpers setting. It contains no jumpers.

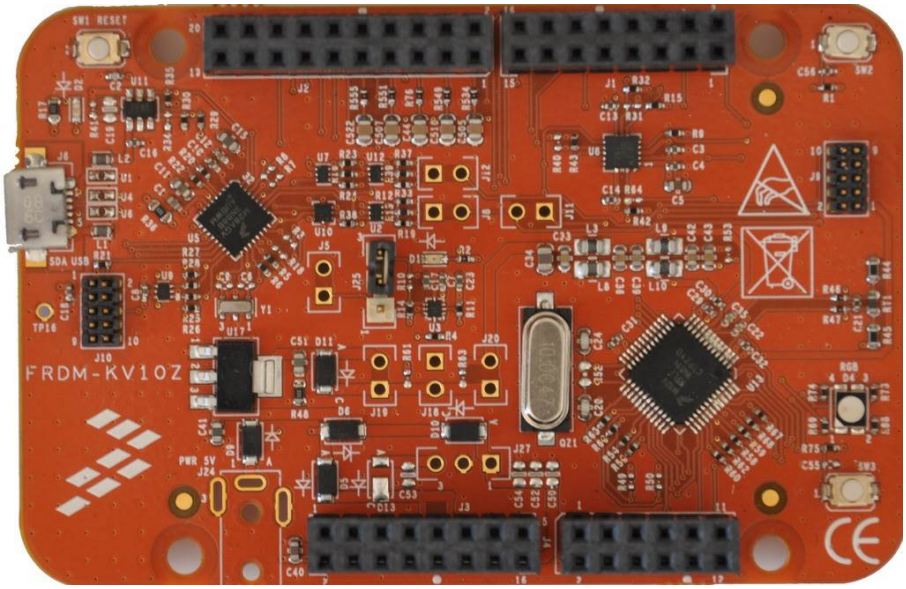

Figure 9. FRDM-KV10Z Freescale Freedom development board

2.4.3. FRDM-KV31F board

The FRDM-KV31F board is a low-cost development tool for the Kinetis V series KV3x MCU family built on the ARM Cortex-M4 processor. The FRDM-KV31F board's hardware is form-factor compatible with the Arduino™ R3 pin layout, providing a broad range of expansion board options, including FRDM-MC-LVPMSM and FRDM-MC-LVBLDC for permanent magnet and brushless DC motor control.

The FRDM-KV31F platform features OpenSDA, the Freescale open source hardware embedded serial and debug adapter running an open source bootloader. This circuit offers several options for serial communication, flash programming, and run-control debugging.

The FRDM-KV31F board does not require any hardware configuration or jumpers setting and contains no jumpers.

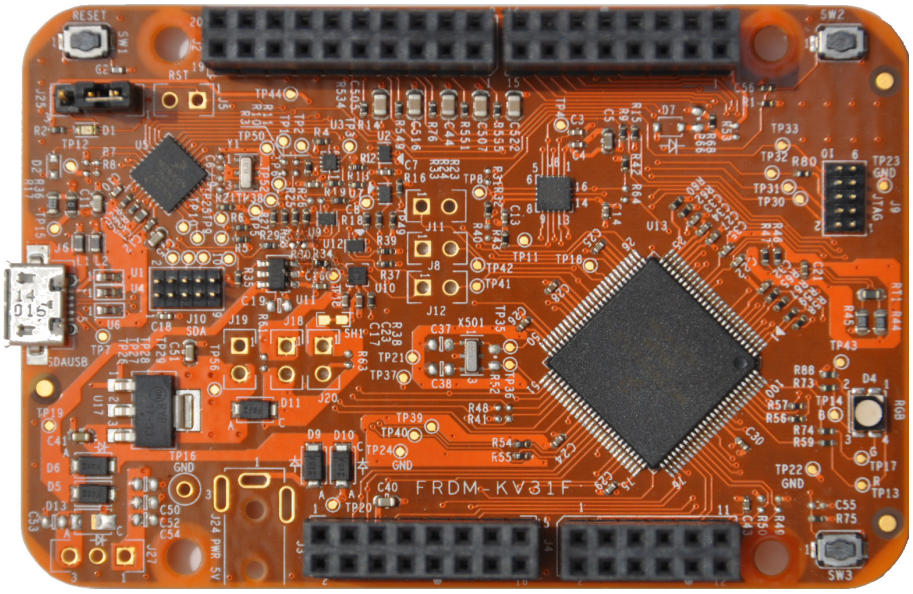

Figure 10. FRDM-KV31F Freescale Freedom development board

2.4.4. Freedom development platform assembly

1. Connect the FRDM-MC-LVBLDC shield on top of the FRDM-KVxxx board.
2. Connect the BLDC motor three-phase wires into the screw terminals on the board.
3. Plug in a USB cable from a USB host to the OpenSDA micro USB connector.
4. Plug in a 12 V DC power supply to the DC power jack.

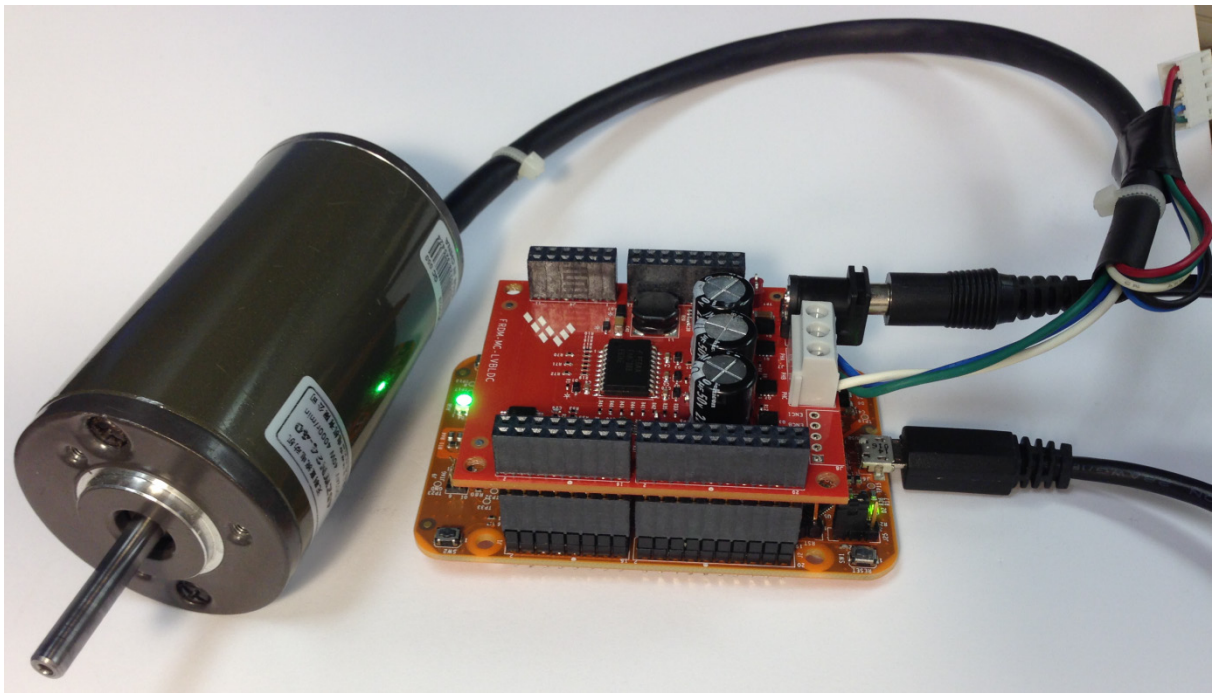

Figure 11. Assembled Freescale Freedom system

2.5. High-Voltage Platform

To run the BLDC application on the Freescale High-Voltage Platform, you need these components:

- Kinetis KV10Z high-voltage daughter board (HVP-KV10Z32), Kinetis KV31F high-voltage daughter board (HVP-KV31F512), or Kinetis KV46F high-voltage daughter board (HVP-KV46F150).
- High-Voltage Platform (HVP-MC-3PH) (motor not included).

Order all modules of the High-Voltage Platform from www.freescale.com or from distributors, and easily build the hardware platform for the target application.

2.5.1. HVP-MC3PH main board

The Freescale High-Voltage Platform is an evaluation and development platform for Kinetis V series MCUs. This platform enables the development of three-phase PMSM, BLDC, and ACIM motor-control and power-factor-correction solutions in a safe high-voltage environment. The board works in the default configuration, and you don't have to set any jumpers to run the attached application.
See *High-Voltage Motor Control Platform User's Guide* (document HVPMC3PHUG).

You don't have to set up the HVP-MC3PH high-voltage development board in any way. The board does not contain any jumpers.

Figure 12. HVP-MC3PH High-Voltage Platform

2.5.2. HVP-KV10Z daughter board

The HVP-KV10Z MCU daughter board contains a Kinetis KV10 family MCU built around the ARM Cortex-M0+ core. This daughter board is developed for use in motor-control applications, together with the High-Voltage Platform main board. This daughter board contains OpenSDA, which is Freescale open-source hardware embedded serial and debug adapter running an open-source bootloader.

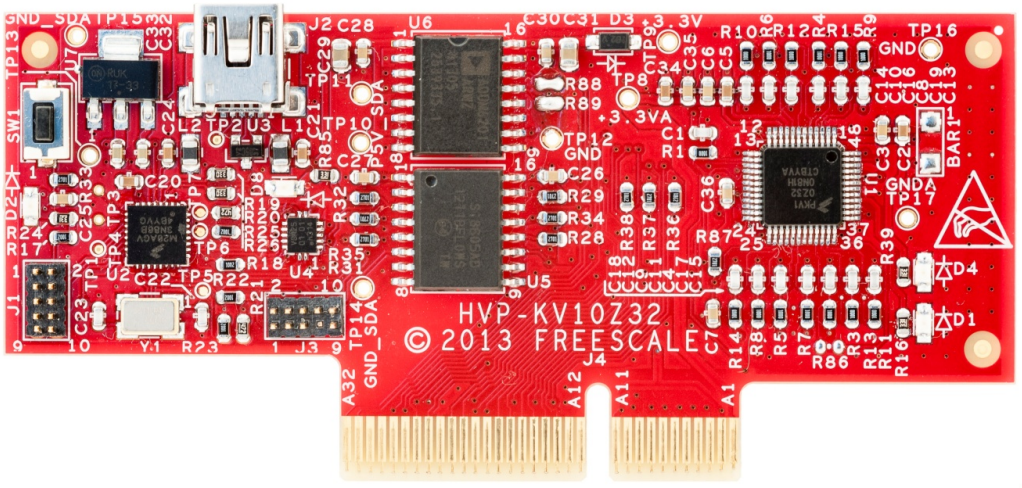

Figure 13. HVP-KV10Z32 daughter board

2.5.3. HVP-KV31F512

The HVP-KV31F512 MCU daughter board contains a Kinetis KV3x family MCU built around the ARM Cortex-CM4 core, which is a powerful chip with a lot of flash memory. The board has the capability to drive a motor. This daughter board is developed for use in motor-control applications, together with the High-Voltage Platform main board. This daughter board contains OpenSDA, which is Freescale open-source hardware embedded serial and debug adapter running an open-source bootloader.

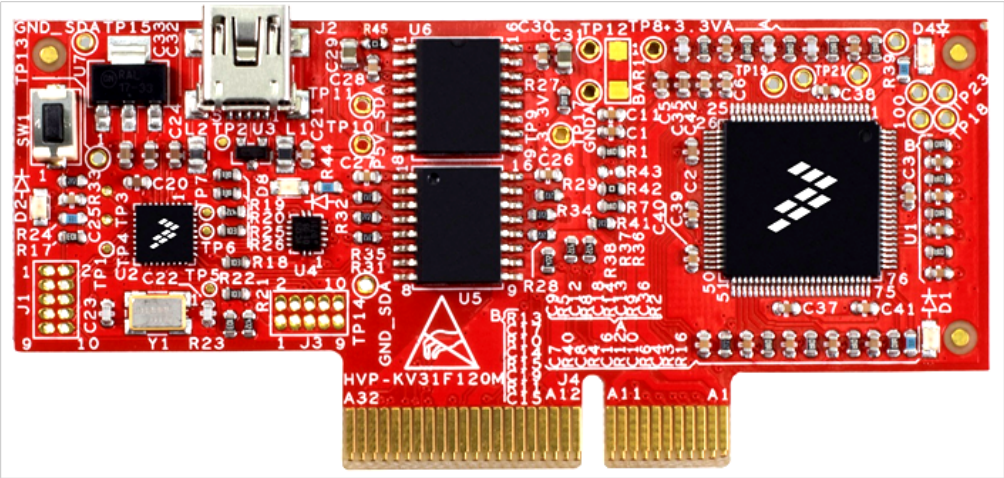

Figure 14. HVP-KV31F512 daughter board

2.5.4. HVP-KV46F150

The HVP-KV46F150 MCU daughter board contains a Kinetis KV4x family MCU built around the ARM Cortex-CM4 core, which is a powerful chip with a lot of flash memory. The board has the capability to drive a motor. This daughter board is developed for use in motor-control applications, together with the High-Voltage Platform main board. This daughter board features OpenSDA, which is Freescale open-source hardware embedded serial and debug adapter running an open-source bootloader.

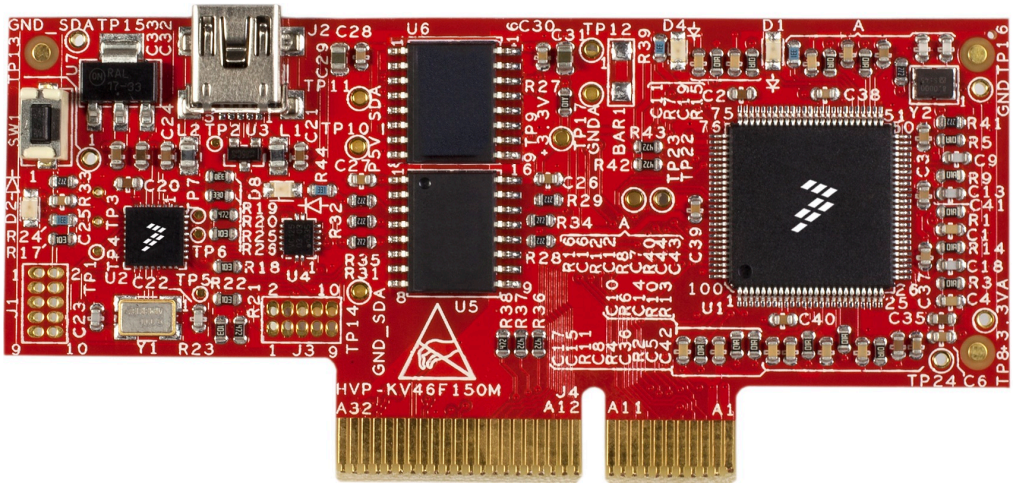

Figure 15. HVP-KV46F150 daughter board

2.5.5. High-Voltage Platform assembling

1. Make sure the HVP-MC3PH main board is unplugged from the voltage source.
2. Insert the HVP-KVxxx daughter board to the HVP-MC3PH main board (there is only one possible option).
3. Connect the BLDC motor three-phase wires to the screw terminals on the board.
4. Plug the USB cable from the USB host to the OpenSDA micro USB connector.
5. Plug a 230 V power supply to the power connector, and switch it on.

3. Project file structure

The BLDC package includes source codes for development boards produced by Freescale such as the Kinetis Tower and Freedom development platforms.

All necessary files are placed in one package which simplifies the distribution and size of the final package. The directory structure of the package is simple, easy to use, and logically organized. The folder structure does not fully correspond to the structures shown in the IDE workspace. A detailed description of the IDE structure is explained in the following chapters.

3.1. Structure organization in BLDC installation

A folder tree of the BLDC package installation is shown in this figure:

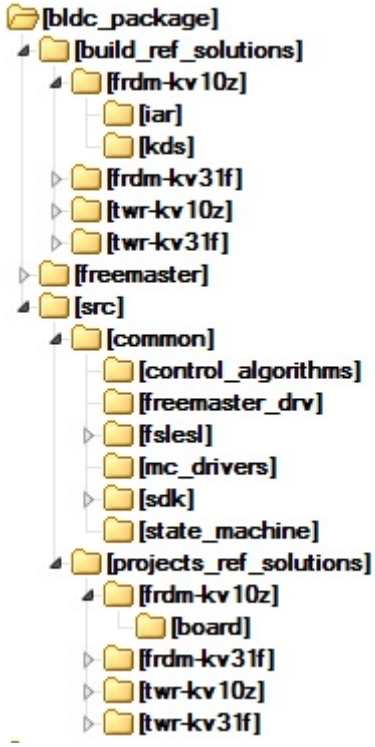

Figure 16. BLDC – package structure

The BLDC package is composed of three folders:

- build_ref_solutions
- freemaster
- src

The main demo project folder */build_ref_solutions\<platform_MCU>* contains these folders and files:

- IAR folder:
  - Contains configuration files for the IAR compiler, as well as the compiler's output executable and object files. If the IAR Embedded Workbench for ARM is installed on your computer double-click the workspace: "build_ref_solutions _<platform_MCU>.eww" to launch the IAR IDE.
  - This folder contains the Freemaster project *build_ref_solutions _<platform_MCU>.pmp* which can be opened by the FreeMASTER tool and is configured to tune a motor application compiled by IAR.
- KDS folder:
  - Contains configuration files for the KDS IDE, and launch configuration for debuggers. If the KDS is installed on your computer, the project must be opened with KDS IDE.
  - This folder contains the FreeMASTER project *build_ref_solutions _<platform_MCU>.pmp* that can be opened with the Freemaster tool and is configured to tune a motor application compiled by KDS.

Folder */freemaster/* contains auxiliary files for the FreeMaster tool.

Folder */src/* contains the subfolder with source and header files for each project:

- Common: Common source and header file used in all projects.
- Projects_ref_solutions: Folder with source code sorted by <platform_MCU>, these source code files are used in IDE as */build_ref_solutions\<platform_MCU>*.

Folder */src/common/* contains the subfolders which are common for the entire project in this package:

- Control_algorithms: Control algorithms used to control the BLDC motor.
- Freemaster_drv: FreeMASTER header and source code required for the FreeMASTER tool.
- Fslesl: Folder with mathematical functions used in the project. This folder includes required header files, source files and library files which are used in the project. It contains two subfolders *cm0* and *cm4*, where precompiled library files are fitted to the two ARM core CM0/CM4. The FSLESL folder is taken from official FSLESL release version 4.1 and it is fully compatible with official release. Library names are changed for better use in available IDEs in bldc_package. For more information about FSLESL, see: freescale.com/fslesl.
- Mc_drivers : This folder includes source and header files used to initialize and drive motor control application.
- Sdk: Contains functions for startup routines, header files, core specific functions, linker files used by available IDEs in this package and routines for core clock settings. The source and header files are obtained from the Kinetis SDK website, see: freescale.com/KSDK.

- State_machine : Contains state machine functions for Fault, Initialization, Stop and Run state.

The folder */src/projects_ref_solutions/\<platform MCU>* is linked with the *build_ref_solutions* project and contains all source files used in the workspace. These files are specific for each of the projects available in the build folders, they specify peripheral initialization routines, Freemaster initialization, motor definitions, and machine states. The source codes are explained in greater detail in the following paragraphs:

- M1_bldc_appconfig.h: Contains constant definitions for the application control processes (parameters of the motor and regulators, and the constants for BLDC sensorless control related algorithms). When the application is tailored for another motor using the Motor Control Application Tuning Tool, this file is generated by the tool at the end of the tuning process. The prefix “m1_” as “motor1” is used in one-motor application, in a multi-motor application this number counting to use motor number.
- M1_state_machine.c: Contains the software routines that are executed when the application is in a particular state or state transition.
- M1_state_machine.h: Header file and macros for m1_state_machine.c.
- Main.c: Contains basic application initialization (enabling interrupts), subroutines accessing the MCU peripherals, and interrupt service routines. The FreeMASTER communication is performed in the background infinite loop.
- Main.h: Header file for main.c.
- Motor_def.h: Fault macros, control structure.
- Board/app_init.c : Application init, not necessary for motor control – UART (FreeMASTER initialization), buttons and LEDs initialization functions.
- Board/app_init.h: Header file for app_init.h, LED and UART macros.
- Board/freemaster_cfg.h: FreeMASTER common configuration file.
- Board/mcdrv_<board_MCU>.c: Motor control driver peripherals initialization functions specific for the board and its MCU.
- Board/mcdrv_<board_MCU>.h: Header file for board/mcdrv_<board_MCU>.c, these files contains macros for changing a FlexTimer pwm periode and ADC channels assigned to phase currents and board voltage.

3.2. Structure organization in IDE workspaces

The structure organization in workspaces (see Section 5-Building and debugging the application) is different to the structure organization described in these sections, but uses the same files as mentioned before. A different organization has been chosen for better manipulation of folders and files in workplaces and to enable the possibility to add or remove files and directories by user.

4. Tools

The following list shows required software to be installed on your PC to run and control BLDC sensorless application properly.

1. Iar Embedded Workbench IDE v7.40 or higher
2. Kinetis Design Studio IDE v3.0 or higher
3. FreeMASTER Run-Time Debugging Tool 2.0

5. Building and debugging the application

The package contains projects for KDS and for IAR Embedded Workbench tools. Both of them are equivalent from motor control application point of view. IDE configurations are set as default and there are no special conditions to run or debug a demo application.

5.1. IAR Embedded Workbench

Users who have installed IAR Embedded Workbench can use this IDE to compile and run a reference solution project.

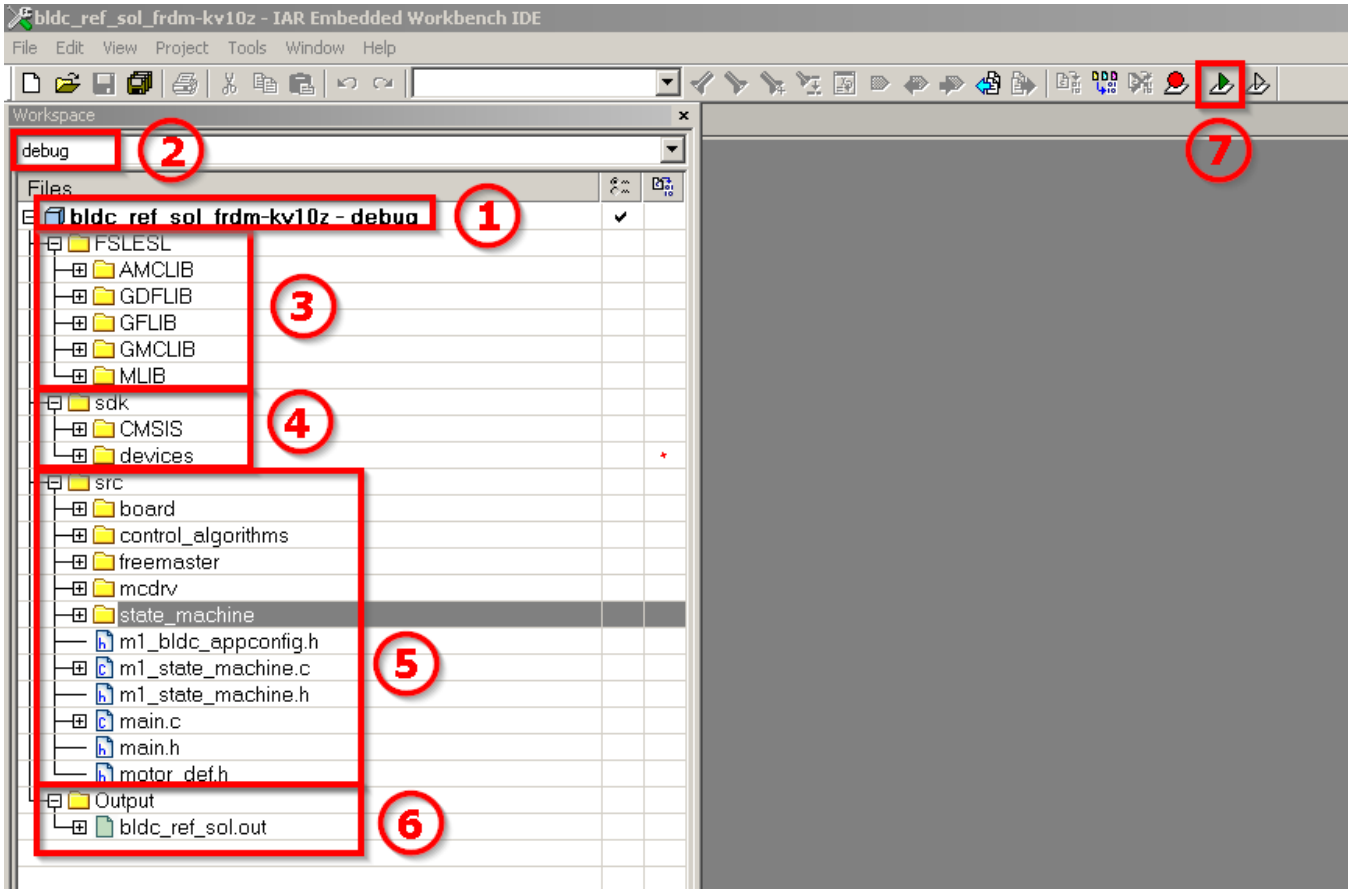

Figure 17. IAR Embedded Workbench

When you open a project in the IAR workbench it is fully configured. It includes all of the necessary source and header files required by the application, such as the startup code, the clock configuration, and the peripheral configuration. You can choose two compiling conditions “debug” and “release”, as shown in *Figure 12* point 2.

These two conditions have the following parameters:

- Debug condition : Used for debugging. Optimization has the flag, None – turned off.
- Release condition: Used for release. Optimization has the flag, High – Highest optimization for speed.

NOTE

Debug condition has optimization turned off. The output file may not be compatible with a MCU with lower amounts of Flash memory (for instance KV10Z32).

Points 3, 4 and 5 show source and header files used by the application that are necessary to run a project.

- Point 3 shows the FSLESL source folder that contains header files for mathematical and control functions used in this project, theory about their use and apply these functions are described in own user guides for all of these libraries such as “AMCLIB” and can be found freescale.com/fslesl.
- Point 4 shows the SDK folder which contains two subfolders, CMSIS and devices. Both of these subfolders are device and/or MCU-core specific, defining startup conditions of the MCU, the clock speed and the interrupt routines.
- Point 5 shows the source code of the application. This folder organization is different in comparison with that featured in section 3.1 Structure organization in BLDC installation. However, it is composed from these files and it is organized by fit for user interaction in IDE workspace.
- Point 6 shows a generated output file by the compiler and ready-to use via the default debugger – PEMicro – OpenSDA. This debugger is set as default in tower boards, and can be changed in project options by right-clicking on Point 1, Options and then menu Debugger is able to change debugger as is required.

5.2. Kinetis Design Studio (KDS)

The Kinetis Design Studio (KDS) is an IDE tool. KDS can be used for developing and testing applications with Freescale MCUs. KDS supports a wide range of Freescale MCUs from Kinetis devices including as the K family, low-power KL family and the family focused on motor control, KV. KDS includes tools for compiling, linking, and debugging source code and supports a wide range of debuggers such as PEMicro or Jlink. The latest release of KDS can be obtained from the Freescale website at freescale.com/kds. You can find further information on installation and configuration is in document *Kinetis Design Studio V3.0.0 User Guide* (document KDSUG).

To open a demo or solution project, it is necessary to select a development board and an MCU. For example to open a project for the Freedom development platform, MCU Kinetis KV10, the project is found at the following path: *bldc_package\build_ref_solutions\frdm-kv10z\kds*. You must then run KDS IDE from its default location and do the following:

Click on the File menu in the left-top corner of the IDE, then select Import from the File list menu, *Figure 13.*

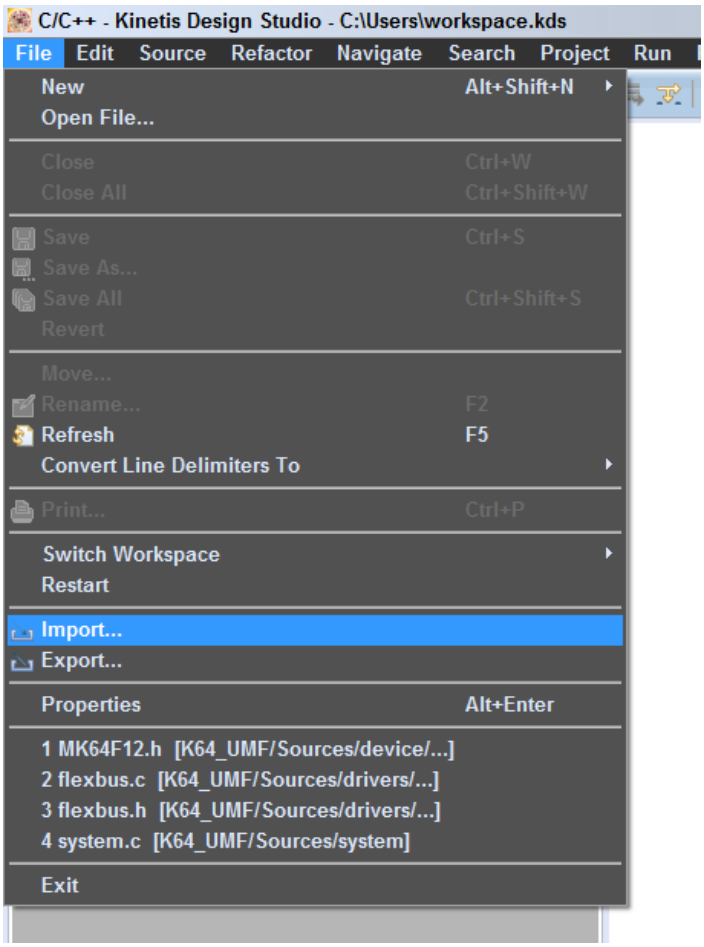

Figure 18. KDS import project

The import window opens, select Existing Projects into Workspace from the General folder. Click on the Next button, as shown in the following figure.

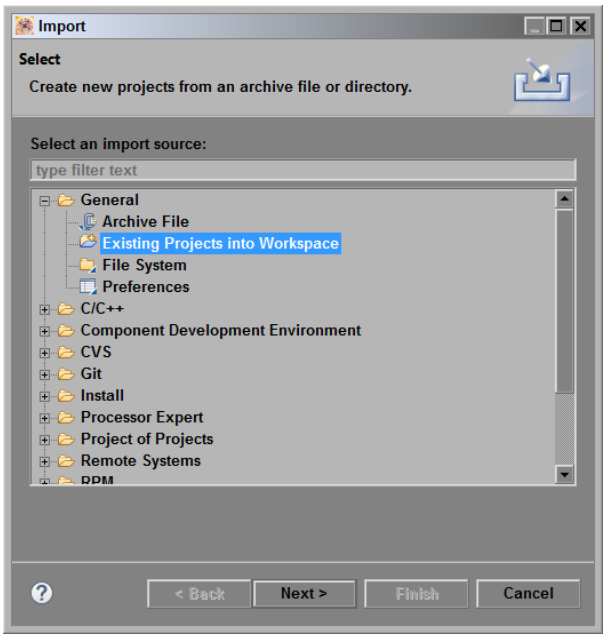

Figure 19. KDS import project 2

The Import window opens as shown in the following figure. Click on the Browse button and locate the chosen project "*//bldc_package\build_ref_solutions\frdm-kv10z\kds"*, confirm this by clicking on OK.

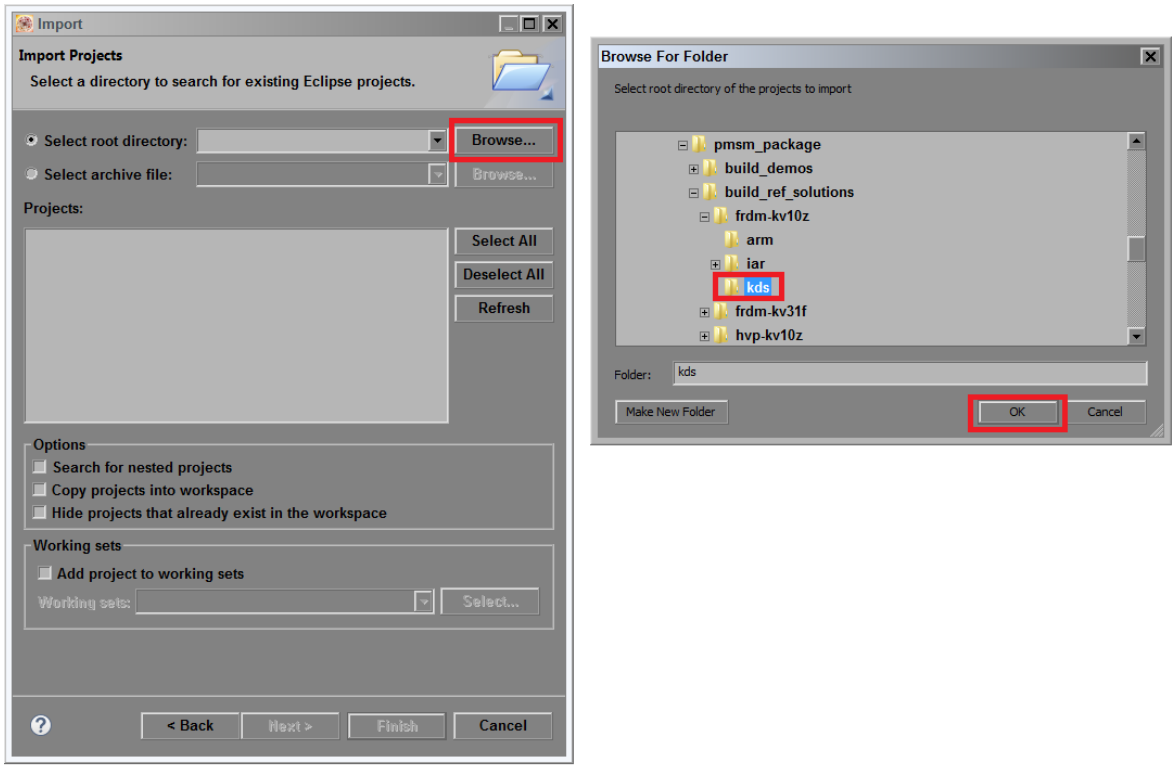

Figure 20. KDS import project 3

The final step is confirmation of the project by clicking on the Finish button, as shown in *Figure 16.*

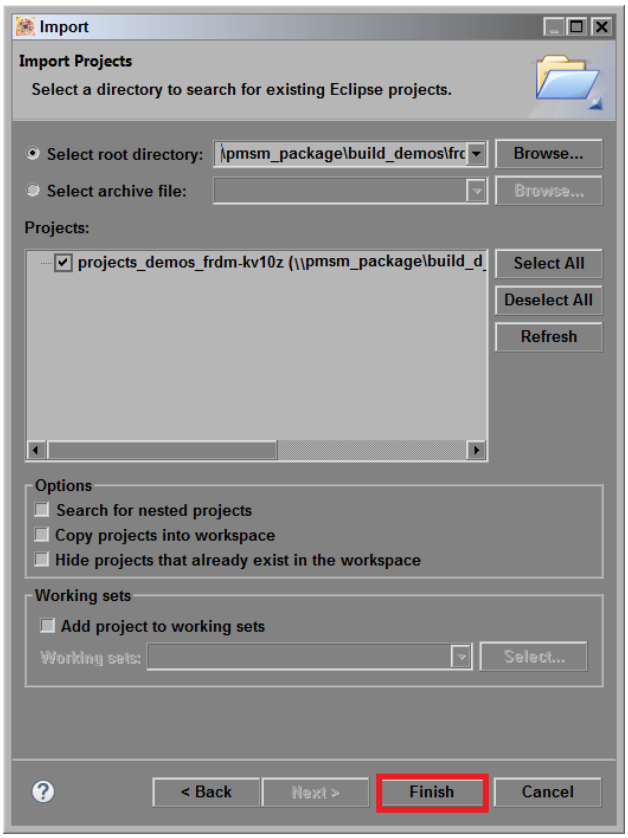

Figure 21. KDS import project 3

The project is now imported to Kinetis Design Studio as shown in *Figure 17*.

- Point 1 shows imported project in Project Explorer.
- Point 2 shows the source code of this project.
- The project can be built by clicking on the build icon, as shown in Point 3. The default configuration is set to debug. You can change the configuration to release.

These two conditions have the following parameters:

- Debug condition: Used for debugging. Optimization has the flag, None – turned off.
- Release condition: Used for release. Optimization has the flag, High – Highest optimization for speed".

NOTE

Debug condition has optimization turned off. The output file may not fit in be compatible with MCUs with lower amounts of Flash memory (for instance KV10Z32).

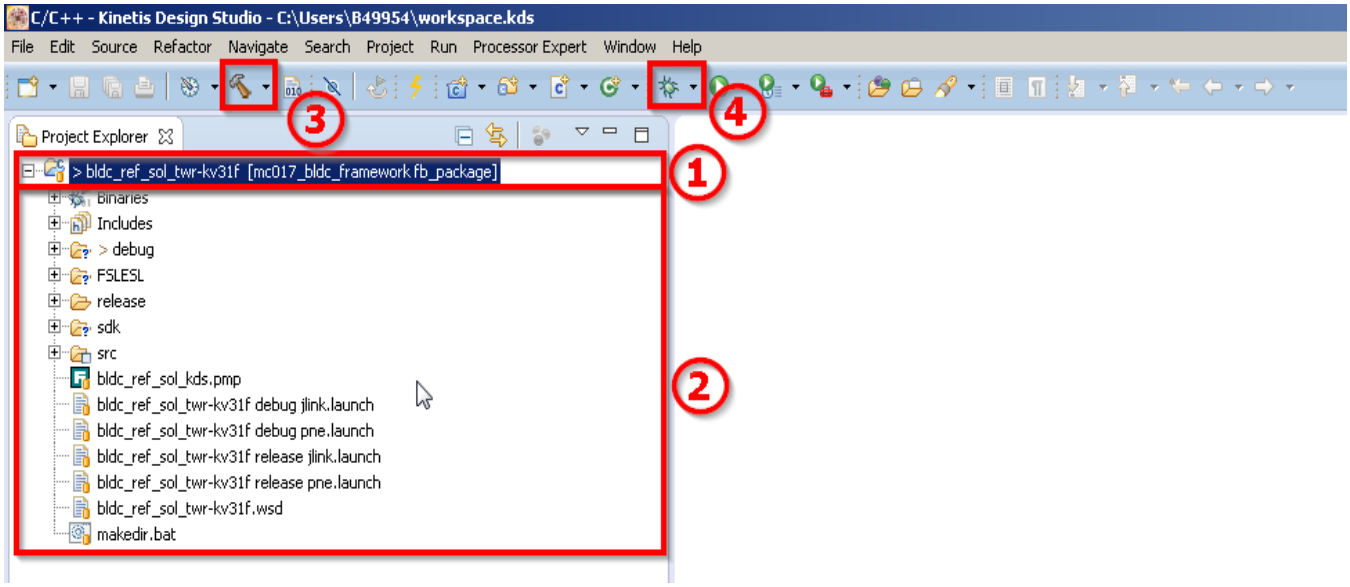

Figure 22. KDS BLDC project

After the project is compiled, the debug or release directory is created depending on which build configuration you have chosen. The *.elf binary file is created in one or both of these directories. You can now use the debugger with predefined debugger option as PEMicro – OpenSDA as the default debugger. User can use another from the menu or to define your own debugger in the top list menu, at Run-> Debug Configuration. You can also change debugging settings there, such as optimization level.

5.3. OpenSDA debugger

Freescale development boards include an OpenSDA serial and debugger adapter which serves as a bridge for serial and debug communication between the target processor and the Host computer. The debugger provides a virtual serial port between the host computer, and can be accessed as the COM port On the embedded target the debugger is connected to the UART peripheral and can control the debug interface that controls the JTAG or SWD debug interface in the target developing platform. The OpenSDA debug interface contains a Mass Storage Device Flash Programmer (MSD Flash Programmer). The MSD Flash Programmer offers an easy way to program applications into the flash memory of the target processor. It appears as a removable drive in the host operating system with a volume label that matches the board name. The raw binary Motorola S-Record files that are copied to the drive are programmed directly into the target memory. The MSD Flash Programmer is designed to program a specific target configuration. It does not support verification or configuration and is not recommended as a production programmer.

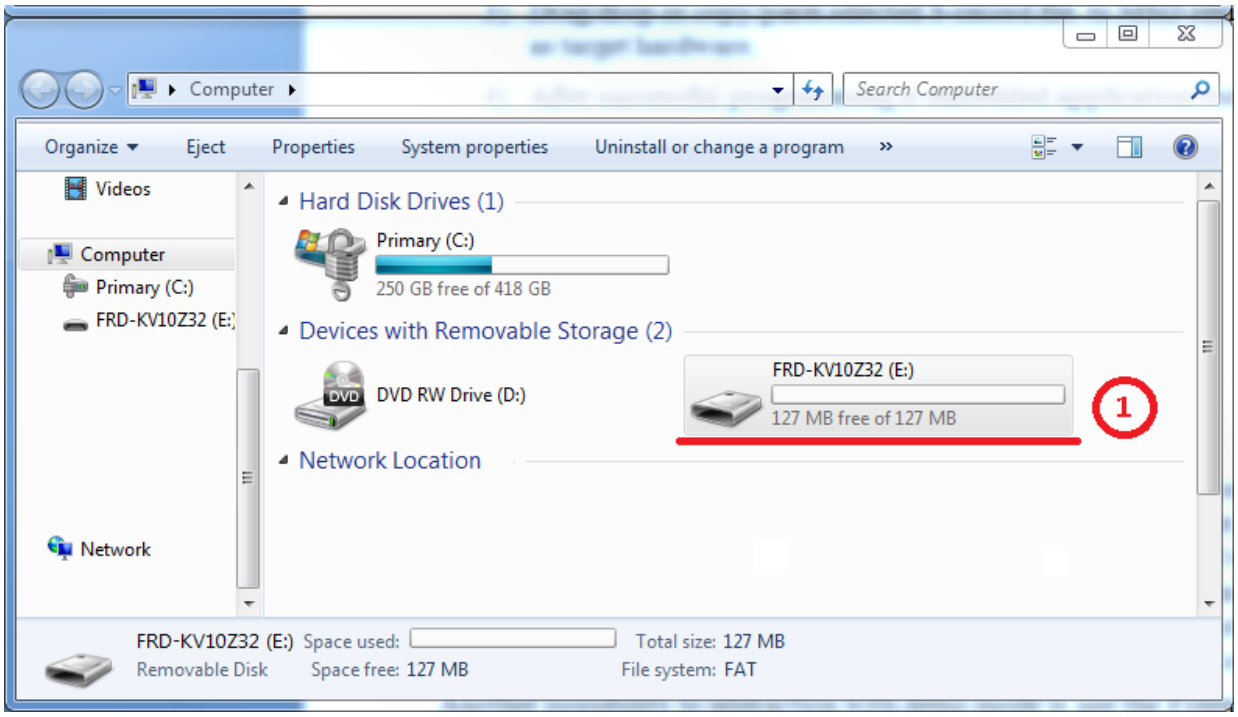

Figure 23. OpenSDA MSD

To load a generated application directly to a target MCU follow the next steps. These steps are applicable in a Windows host System.

1. Open Windows Explorer.
2. Locate the generated S-record file (file with extension .srec, .s19 or .bin) (for example *pmsm_package\build_ref_resolutions\<board_MCU>\<tool>\debug\projects_ref_solutions.sre*.
3. Drag and drop or copy and paste the selected S-record file to a MSD removable drive with a label of volume as target hardware, as shown in point 1 in the previous figure.
4. After successful programming the embedded application executes automatically.
5. Reconnect the target device.

5.4. Compiler warnings

Warnings are diagnostic messages that report instructions that are not inherently erroneous and warn about potential runtime, logic, and performance errors. In some cases warnings may be suspended and are not shown during the compiling process. One notable case is the warning, unused function. Functions and their body are implemented in the source code, but this function is not used. This case may occur when you implement a function for better usability, as a supporting function, but in some special cases you do not use the function.

The BLDC project contains reference solution projects. In these projects, especially in IDEs, some warnings are suppressed. These warnings are listed below, and other warning are setup to standard configuration.

IAR workbench has the following suppressed warnings:

- Pa082: Undefined behavior, the order of volatile accesses is undefined in this statement.
- Pe177: <entity> was declared but never referenced.

Kinetis Design Studio (KDS) has the following suppressed warnings:

- Parentheses: Warns if parentheses are omitted in certain contexts.
- Unused-function: Function is defined but not used.
- Uninitialized: Variable is used but uninitialized.
- Main: Due to optimization, main is void not integer, never return value.

During the compiling process there are no other warnings shown by default.

6. User interface

The application contains a demo application mode for demonstration purposes of motor rotating and can be operated either with the user button or by using a FreeMASTER Tool. Tower and Freedom boards feature a user button key associated with port interrupt, interrupt is generated whenever the button is pressed. Pressing the button causes the demo mode to begin; pressing the same button again causes the stop state and transition back to the STOP state. The location and number of user buttons is defined in Table 8.

The second way to access demo mode is by using the FreeMASTER tool. The FreeMASTER application consists of two parts: the PC application used for variable visualization, and the set of software drivers running in the embedded application. Data is transferred between the PC and the embedded application using the RS232 interface. This interface is provided by OpenSDA debugger, included in the boards.

The application can be controlled using two interfaces:

- Button control on Freescale Kinetis V Tower and Freedom development boards
- Remote control using FreeMASTER

These two options are explained in the following sections.

6.1. Button control

Pressing the control button switches on demonstration mode (or demonstration mode is switched off if it is currently switched on). The following table shows the correct switch button for your development board.

Table 8. Control button assignment

Board	Control button
TWR-KV10Z32	SW2
TWR-KV11Z75	SW2
TWR-KV31F120	SW2
TWR-KV46F150M	SW2
FRDM-KV10Z	SW2
FRDM-KV31F	SW2

6.2. Remote control using FreeMASTER

Remote operation can be provided by FreeMASTER software via the USB interface. FreeMASTER 2.0 is required for optimal operation of the application. FreeMASTER 2.0 application installation is available to download on www.freescale.com/freemaster.

Follow the following steps to control BLDC motor using FreeMASTER:

1. Open the FreeMASTER project file (.pmp). Open the project file for your IDE:
   - For KDS, the FreeMASTER project file is located at the following path:

   Project_directory/build/kds/.
   - For IAR, the FreeMASTER project is located at the following path:

   Project_directory/build/iar/.
2. Toggle the communication button (red STOP button in top left hand corner) to establish the communication.

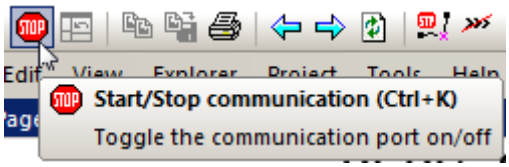

Figure 24. Red STOP button is placed in top left hand corner

After communication has been successfully established, the FreeMASTER communication status in the bottom right hand corner should change from not connected to RS232 UART Communication; COMxx; speed=19200, if the communication setup fails then a FreeMASTER warning popup window will appear.

Figure 25. FreeMASTER communication status when communication is established successfully

If the communication setup has failed, you must perform the following troubleshooting steps:

6. Go to Project->Options->Comm tab and make sure, that in port option is written SDA and communication speed is set to 19200 bps.

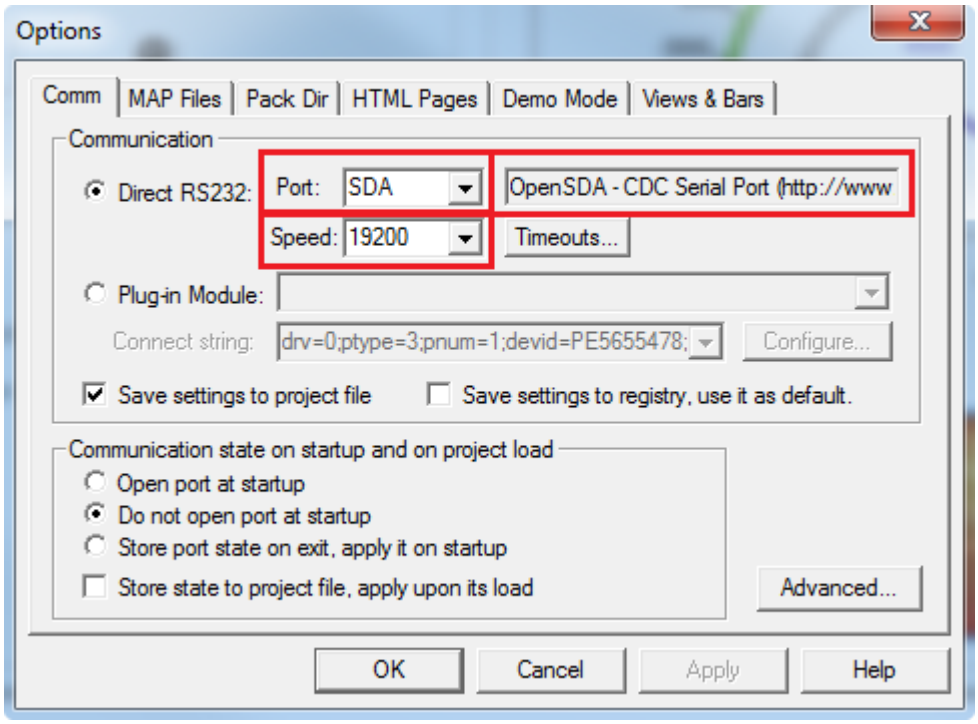

Figure 26. FreeMASTER communication setup window

7. If OpenSDA-CDC Serial Port is not displayed in the message window near to port listbox, unplug and plug in the USB cable and reopen the FreeMASTER project again.
8. Ensure that your development board is supplied with a sufficient power source. Sometimes the USB PC port does not have sufficient power to supply the development board.

6.2.1. Control page

After launching the application and performing all necessary settings the BLDC motor can be controlled from the FreeMASTER control page. The FreeMASTER control page contains:

- Speed gauge: showing actual and required speed.
- Speed slider: to set the required speed.
- DC bus voltage gauge: showing actual DC bus voltage.
- DC bus current gauge: showing actual DC bus current.
- DC bus current limitation slider: to setup DC bus current limit.
- Demo mode button: to turn on/off the demonstration mode.
- STOP button: to stop the whole application.
- Application State Notification: shows actual application state, and also shows faults.

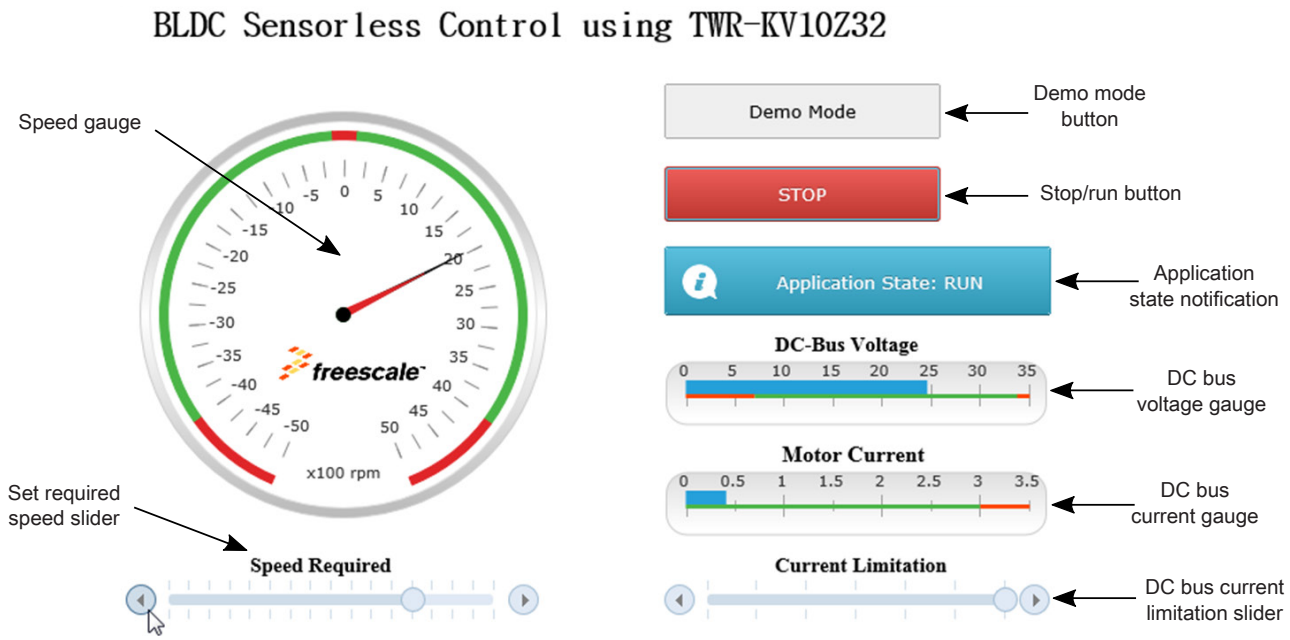

Figure 27. FreeMASTER control page

Basic instructions:

- To start the motor run, use the Speed Slider to setup required speed.
- In case a fault is indicated, click on the "CLEAR FAULT" button to clear the fault.
- Click on the Demo Mode button to turn on/off demonstration mode.
- Click on the Stop button to stop the motor.

7. Instructions

7.1. Running the motor

1. Assemble your Freescale hardware according to instructions in section 2, Hardware setup.
2. Download the correct project into your target using the OpenSDA debug interface.
3. Open the FreeMASTER project, establish the communication between MCU and PC following the instructions in section 6.2, Remote control using FreeMASTER.
4. Set up the required speed of the motor on the speed slider on the FreeMASTER control page.

7.2. Stopping the motor

1. Click on the Stop button on the FreeMASTER control page.

2. Require zero speed with using speed slider.
3. In emergency cases, turn off power supply.

7.3. Clearing a fault

To clear a fault click on the CLEAR FAULT button on the control page.

7.4. Turning on demonstration mode

1. If you use the FreeMASTER control page, click on the Demo Button.
2. If you do not use the FreeMASTER control page push the corresponding button on your development board to turn demonstration mode on/off.

8. References

The following documents can be found on freescale.com:

1. Kinetis Design Studio v3.0.0 User's Guide (document KDSUG)
2. Freescale Embedded Software Libraries User's Guides

9. Revision history

Table 9. Revision history

Revision number	Date	Substantive changes
0	10/2015	Initial release
1	11/2015	Added Section 2.5, "High-Voltage Platform", and Section 2.2.5, "TWR-KV46F150M". Updated Table 1. Updated Table 8.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: BLDCAPPACUG
Rev. 1
11/2015