NXP Semiconductors
User's Guide

Document Number: PMSMDEMOUG
Rev. 1, 06/2016

PMSM Control Demo User's Guide

Contents

1. Introduction

This user’s guide provides a step-by-step guide on how to open, compile, and run Permanent Magnet Synchronous Motor (PMSM) projects. It describes the basic compiling steps for IAR Embedded Workbench®, Kinetis Design Studio (KDS), and µVision® Keil® IDEs for a wide range of supported development platforms described in Section 2, “Supported Development Boards”. For more information, see Section 10, “References” or visit www.nxp.com/motorcontrol_pmsm.

2. Supported Development Boards

There are three supported development boards with two Kinetis KV series motor-control MCUs for motor-control applications. The development boards and supported MCUs are shown in Table 1. The Tower System modular development platform and NXP Freedom development platform are targeted for low-voltage and low-power applications with PMSM control type.

Table 1. Supported development platforms

—		Platform	
		Tower System	**Freedom**
Power stage		TWR-MC-LV3PH	FRDM-MC-LVPMSM
MCU	**KV10Z**	TWR-KV10Z32	FRDM-KV10Z
	KV31F	TWR-KV31F120M	FRDM-KV31F
	KV11Z	TWR-KV11Z75M	—
	KV46F	TWR-KV46F256	—
	KV58F	TWR-KV58F220M	—
	KE15Z	—	FRDM-KE15Z
	KE18F	TWR-KE18F	—

3. Motor Control versus MCUXpresso SDK Peripheral Drivers

Motor Control examples use MCUXpresso SDK peripheral drivers to configure general peripherals as clocks, SPI, SIM, ports. However, motor control requires critical application timing as most of control algorithm runs in 100us loop. To optimize CPU load, the maximum of peripheral hardware features are implemented for PWM signal generation, analogue signal sampling and synchronization between PWM and ADC units.

Standard MCUXpresso SDK peripheral drivers do not support configuration and handling all required features. Motor control drivers are designed to configure critical MC peripherals (eflexPWM, FTM, ADC, PDB).

It is highly recommended not to modify default configuration of allocated MC peripherals due to possible application timing conflict. The particular *mcdrv_< board&MCU >.c* source file contains configuration functions of allocated peripherals.

4. Hardware Setup

The PMSM sensorless application runs on Tower and Freedom development platforms with 24 V Linix Motor in the default configuration.

4.1. Linix 45ZWN24-40 motor

The Linix 45ZWN24-40 motor (described in Table 2**)** is a low-voltage three-phase motor used in BLDC and PMSM sensorless applications.

Table 2. Linix 45ZWN24-40 motor parameters

Characteristic	Symbol	Value	Units
Rated voltage	Vt	24	V
Rated speed @ Vt	—	4000	RPM
Rated torque	T	0.0924	Nm
Rated power	P	40	W
Continuous current	Ics	2.34	A
Number of pole pairs	pp	2	—

Figure 1. Linix motor

The motor has two types of connectors (cables). The first cable has three wires and it is designed to power the motor. The second cable has five wires and it is designed for Hall sensors signal sensing. For the PMSM sensorless application, you need only the power input wires.

4.2. Running PMSM application on Tower System

To run the PMSM application on the Tower System, you need the following Tower modules:

- Tower board with a Kinetis V series MCU (TWR-KV10Z32, TWR-KV11Z75M, TWR-KV31F120M, TWR-KV46F150M, or TWR-KV58F220M).
- Tower board with Kinetis E series MCU (TWR-KE18F).
- Three-phase low-voltage power module (TWR-MC-LV3PH) with included Linix motor.
- Tower elevator modules (TWR-ELEV).

You can order all Tower modules from www.nxp.com or from distributors to easily build the hardware platform for the target application.

4.2.1. TWR-MC-LV3PH module

The 3-Phase Low-Voltage Motor Control module (TWR-MC-LV3PH) is a peripheral Tower System module, interchangeable across the Tower System. The phase voltage and current feedback signals are provided. These signals enable a variety of algorithms to control the 3-phase PMSM and BLDC motors. A high level of board protection (over-current, under-voltage, over-temperature) is provided by the MC33937 pre-driver. Before you insert the TWR-MC-LV3PH module into the Tower System, ensure that the jumpers on your TWR-MC-LV3PH module are configured as follows:

Table 3. TWR-MC-LV3PH jumper settings

Jumper	Setting	Function
J2	1-2	Selects the internal analog power supply.
J3	1-2	Selects the internal analog power reference (GND).
J10	1-2	Selects I_SENSE_C.
J11	1-2	Selects I_SENSE_B.
J12	1-2	Selects I_SENSE_A.
J13	1-2	Selects I_SENSE_C.
J14	1-2	Selects I_SENSE_A.

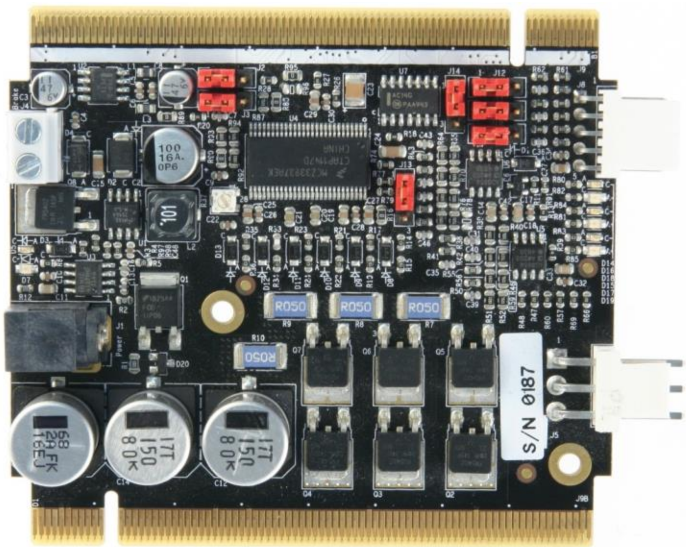

Figure 2. TWR-LV-MC3PH jumper settings

4.2.2. TWR-KV10Z Tower System module

The TWR-KV10Z32 is a development tool for the NXP Kinetis KV1x family of MCUs built around the ARM® Cortex®-M0+ core. This MCU has enough power for use in motor-control applications (such as PMSM or BLDC motors with simple or advanced control techniques). The MCU has a wide range of peripherals.

To begin, configure the jumpers on the TWR-KV10Z32 and TWR-MC-LV3PH Tower System modules properly. The following table lists the specific jumpers and their settings for the TWR-KV10Z32 Tower System module.

Table 4. TWR-KV10Z jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J1	2-3	J10	2-3	J21	3-4
J2	1-2	J11	open	J22	3-4
J3	2-3	J12	open	J25	open
J4	1-2	J13	open	J26	1-2
J5	1-2	J14	open	J27	1-2

J7	1-2	J18	2-3	J28	1-2
J8	2-3	J19	2-3	J29	1-2
J9	1-2	J20	2-3	—	—

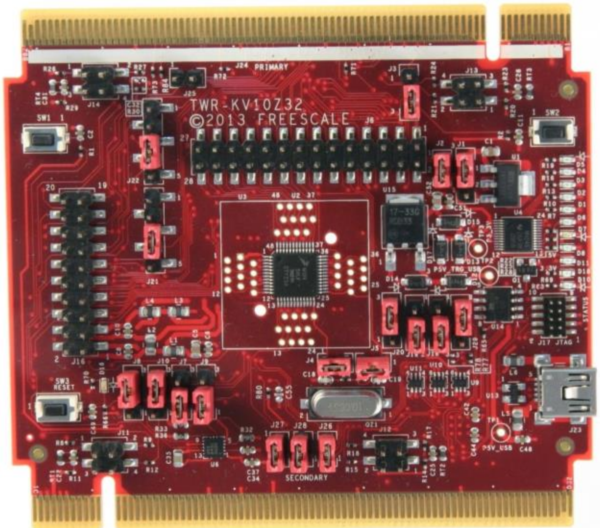

Figure 3. TWR-KV10Z Tower System module

4.2.3. TWR-KV11Z Tower System module

The TWR-KV11Z75M is a development tool for the NXP Kinetis KV1x family of MCUs built around the ARM Cortex-M0+ core. This MCU has enough power for use in motor-control applications (such as PMSM or BLDC motors with simple or advanced control techniques). The MCU has a wide range of peripherals.

To begin, configure the jumpers on the TWR-KV11Z75M and TWR-MC-LV3PH Tower System modules properly. The following table lists the specific jumpers and their settings for the TWR-KV11Z75M Tower System module.

Table 5. TWR-KV11Z jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J1	1-2	J9	open	J512	1-2
J2	2-3	J10	1-2	J517, J518	J518-J517(2)
J4	2-3	J11	2-3	J519	1-2
J5	5-6, 7-8, 9-10	J12	1-2	J523	1-2
J6	open	J13	2-3	J524	open
J7	open	J14	1-2	J526	1-2
J8	open	J17	2-3	—	—
J505	2-3	J506	2-3	—	—

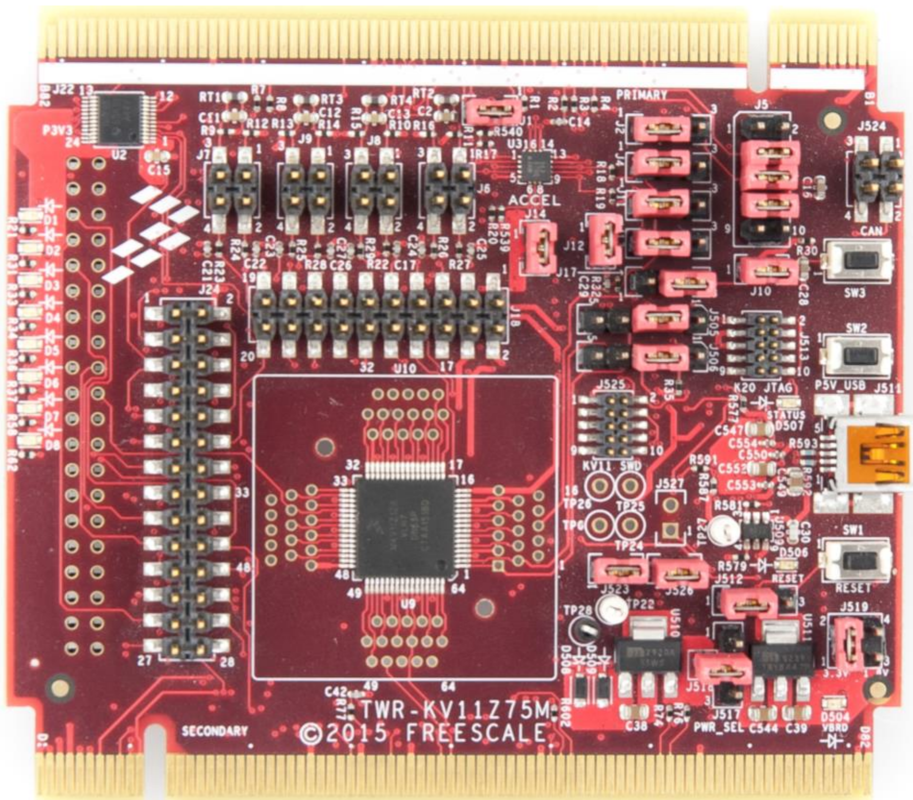

Figure 4. TWR-KV11Z Tower System module

4.2.4. TWR-KV31F Tower System module

The TWR-KV31F120M is a development tool for the NXP Kinetis KV3x family of MCUs built around the ARM Cortex-M4 core. This MCU has enough power for use in motor-control applications (such as PMSM or BLDC motors with simple or advanced control techniques). The MCU has a wide range of peripherals, lots of memory (depending on the model used), and a powerful core.

To begin, configure the jumpers on the TWR-KV31F120M and TWR-MC-LV3PH Tower System modules properly. The following table lists the specific jumpers and their settings for the TWR-KV31F120M Tower System module.

Table 6. TWR-KV31F jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J1	1-2	J10	open	J17	1-2, 3-4, 5-6, 7-8
J3	1-2	J11	open	J20	open
J4	open	J12	open	J22	2-3
J5	1-3	J13	1-2	J23	2-3
J8	open	J14	1-2	J25	1-2
J9	open	J15	1-2	J26	2-3

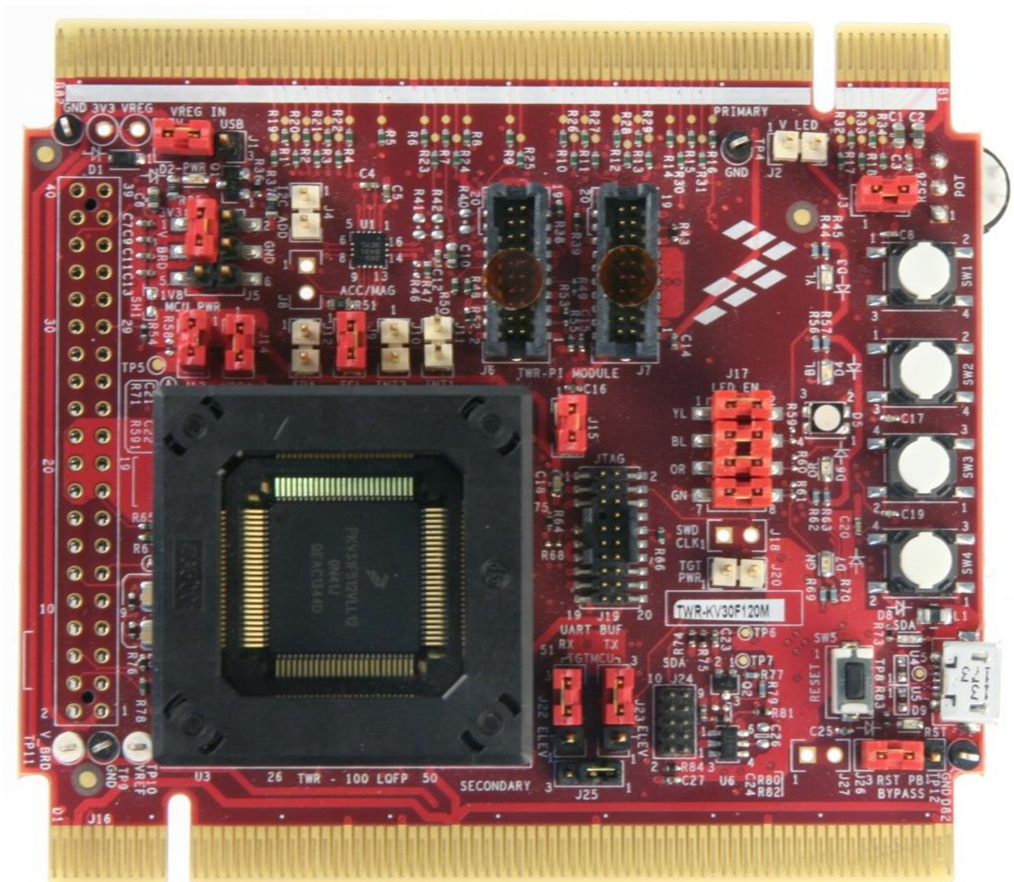

Figure 5. TWR-KV31F Tower System module

4.2.5. TWR-KV46F Tower System module

The TWR-KV46F150M is a development tool for the NXP Kinetis KV4x family of MCUs built around the ARM Cortex-M4 core. This MCU has enough power for use in motor-control applications (such as PMSM or BLDC motors with simple or advanced control techniques). The MCU has a wide range of motor-control peripherals, lots of memory (depending on the model used), and a powerful core.

To begin, configure the jumpers on the TWR-KV46F150M and TWR-MC-LV3PH Tower System modules properly. The following table lists the specific jumpers and their settings for the TWR-KV46F150M Tower System module.

Table 7. TWR-KV46F jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J1	open	J16	open	J505	3-4
J2	open	J19	open	J506	3-4
J4	2-3	J20	1-2	J512	1-2
J5	1-2	J21	open	J514	2-3
J13	1-2, 3-4	J23	open	J517	2-3
J15	1-2	—	—	J519	3-4

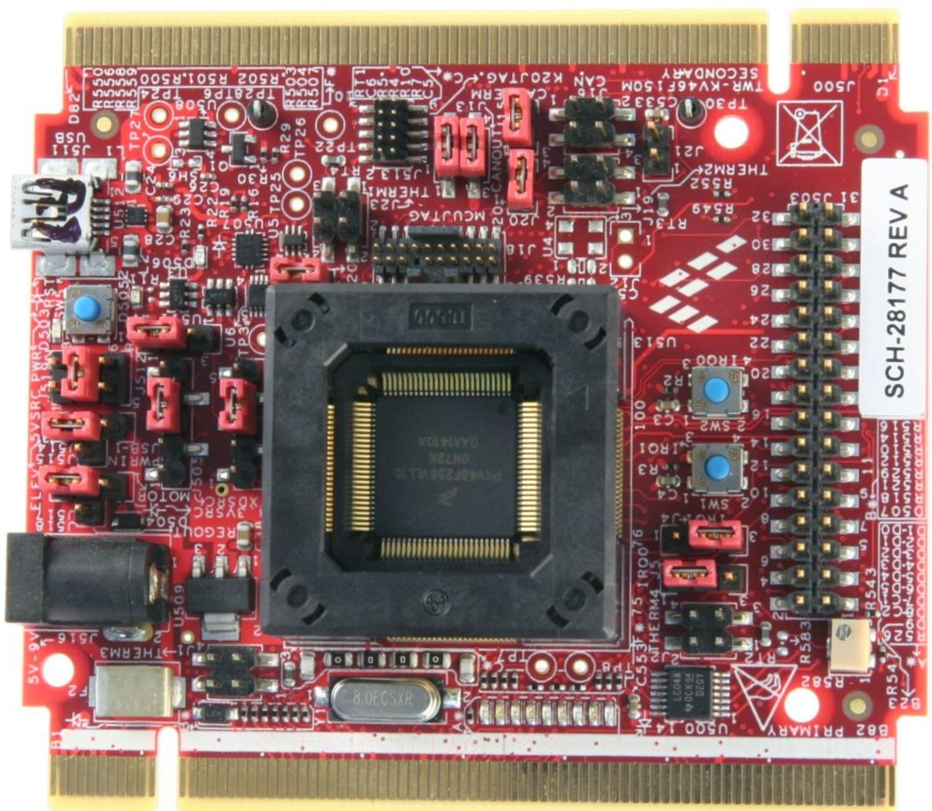

Figure 6. TWR-KV46F Tower System module

4.2.6. TWR-KV58F Tower System module

The TWR-KV58F220M is a development tool for the NXP Kinetis KV5x family of MCUs built around the ARM Cortex-M7 core. This MCU has enough power for use in multi-motor control applications (such as PMSM or BLDC motors with simple or advanced control techniques). The MCU has a wide range of motor-control peripherals, lots of memory (depending on the model used), and a powerful core.

To begin, configure the jumpers on the TWR-KV58F220M and TWR-MC-LV3PH Tower System modules properly. The following table lists the specific jumpers and their settings for the TWR-KV58F220M Tower System module.

Table 8. TWR-KV58F jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J1	1-2	J11	1-2	J23	2-3
J2	open	J12	1-2	J24	2-3
J3	1-2	J14	open	J25	2-3
J4	1-2	J17	open	J26	2-3
J7	1-2	J18	open	J28	1-2
J8	1-2	J19	open	J29	open
J9	open	J20	open	J30	1-2, 3-4, 5-6, 7-8
J10	1-2	J21	1-2	—	—

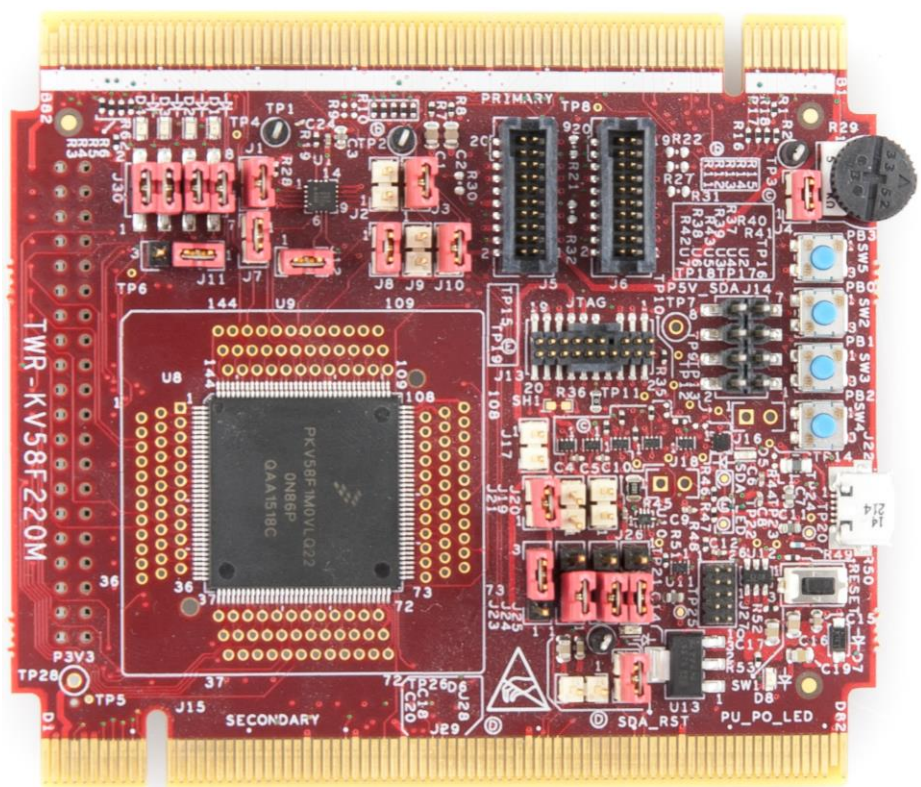

Figure 7. TWR-KV58F Tower System module

4.2.7. TWR-KE18F Tower System module

The TWR-KE18F is a development tool for the NXP Kinetis KE1x family of MCUs built around the ARM Cortex-M4 core. This MCU has enough power for use in motor-control applications (such as PMSM or BLDC motors with simple or advanced control techniques). The MCU has a wide range of peripherals, lots of memory (depending on the model used), and a powerful core.

To begin, configure the jumpers on the TWR-KE18F and TWR-MC-LV3PH Tower System modules properly. The following table lists the specific jumpers and their settings for the TWR-KE18F Tower System module.

Table 9. TWR-KE18F jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J3	2-3	J9	1-2	J18	1-2
J4	1-2	J11	1-2	J19	1-2
J5	2-3	J12	1-2	J21	open
J6	1-2	J13	1-2	J22	1-2
J7	1-2	J16	1-2	J23	open
J8	open	J17	1-2	J24	1-2

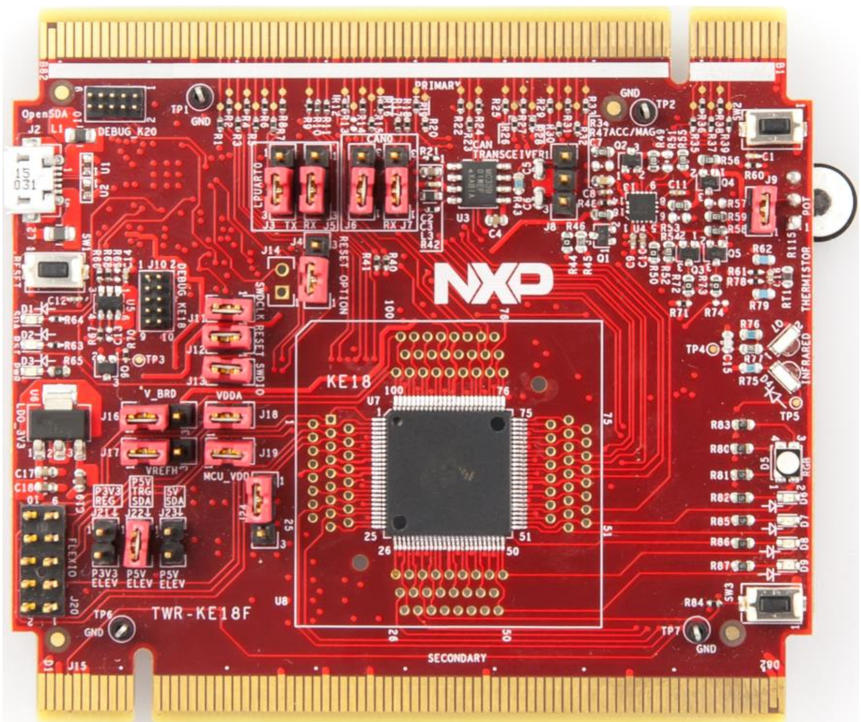

Figure 8. TWR-KE18F Tower System module

4.2.8. Tower System assembling

1. Insert the TWR-KVxxXxx MCU module and the TWR-MC-LV3PH peripheral module into the TWR-ELEV cards. 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 assembling 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 the power supply cable attached to the TWR-MC-LV3PH system kit to the motor-control peripheral board (TWR-MC-LV3PH).
   - Connect the TWR MCU module to any USB port on the host PC via a USB cable.

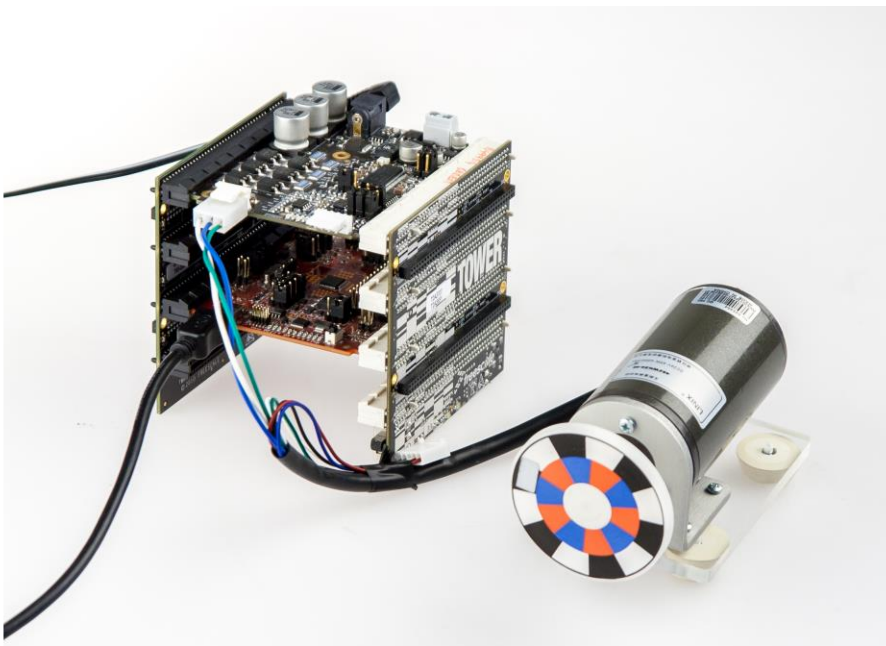

Figure 9. Assembled Tower System

4.3. NXP Freedom development platform

To run the PMSM application using the NXP Freedom development platform, you need these Freedom boards:

- Freedom board with a Kinetis V series MCU (FRDM-KV10Z or FRDM-KV31F).
- Freedom board with a Kinetis E series MCU (FRDM-KE15Z).
- Three-phase low-voltage power Freedom shield (FRDM-MC-LV3PH) with included Linix motor.

You can order all Freedom modules from nxp.com or from distributors, and easily build the hardware platform for the target application.

4.3.1. FRDM-MC-LVPMSM

The FRDM-MC- LVPMSM low-voltage evaluation board (in a shield form factor) turns an NXP Freedom development board into a complete motor-control reference design compatible with existing Freedom development platforms (FRDM-KV31F and FRDM-KV10Z).

The FRDM-MC-LVPMSM board does not require any hardware configuration or jumper setting.

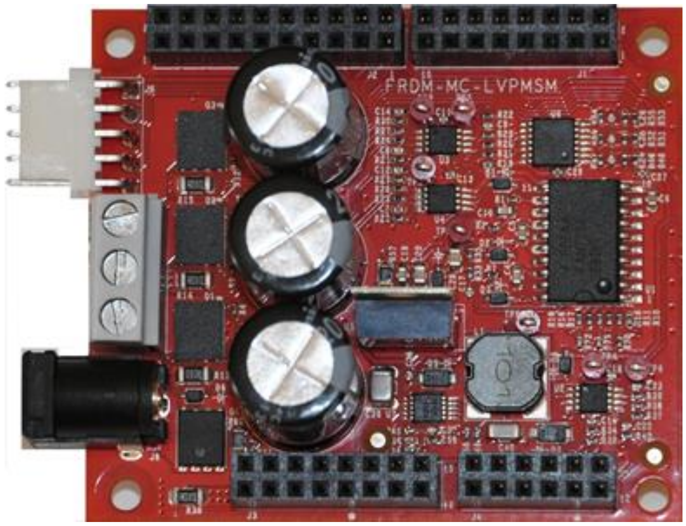

Figure 10. FRDM-MC-LVPMSM

4.3.2. FRDM-KV10Z

The FRDM-KV10Z is a low-cost development tool for Kinetis KV1x family of MCUs built around the ARM Cortex-M0+ core. The FRDM-KV10Z 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 NXP open-source hardware embedded serial and debug adapter running an open-source bootloader.

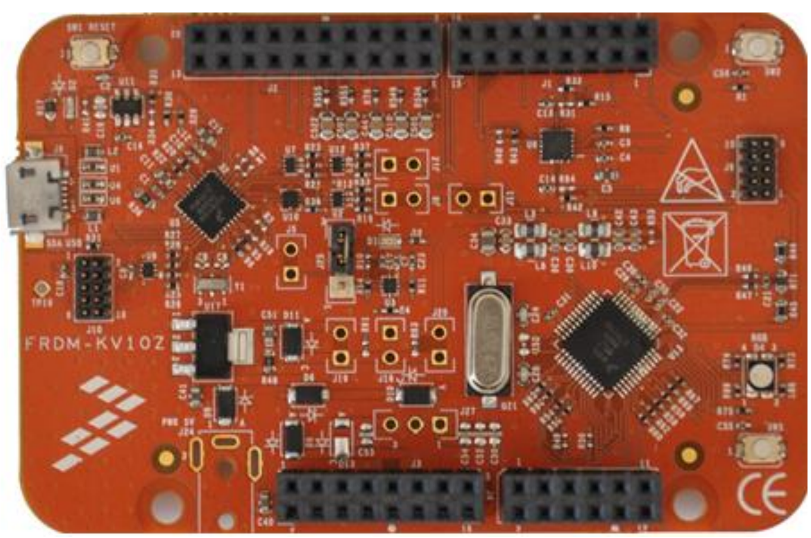

Figure 11. FRDM-KV10Z development board

4.3.3. FRDM-KV31F

FRDM-KV31F is a low-cost development tool for Kinetis KV3x family of MCUs built around the ARM Cortex-M4 core. FRDM-KV31F 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.

FRDM-KV31F features OpenSDA, the NXP 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.

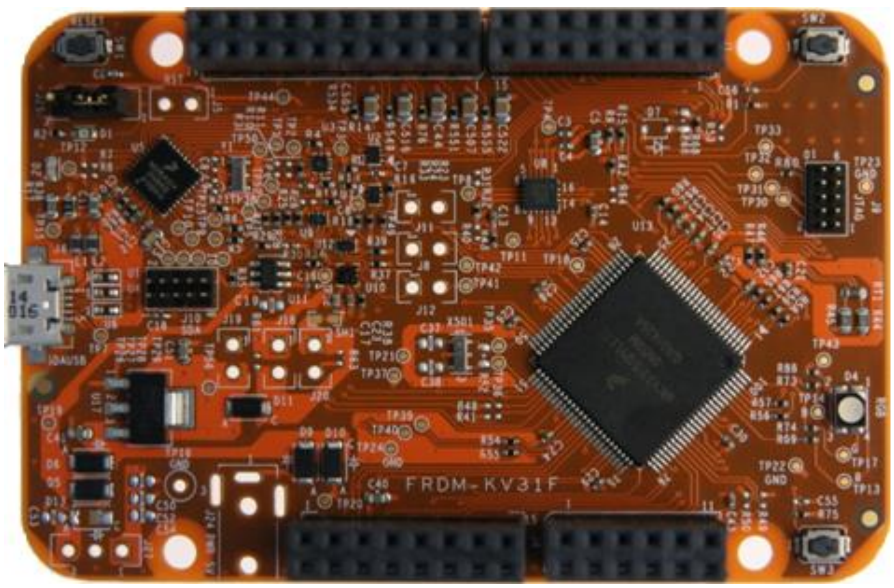

Figure 12. FRDM-KV31F development board

4.3.4. FRDM-KE15Z board

The FRDM-KE15Z is a low-cost development tool for Kinetis KE1x family of MCUs built around the ARM Cortex-M0+ core. The FRDM-KE15Z hardware is form-factor compatible with the Arduino™ R3 pin layout, providing a broad range of expansion board options. The FRDM-KE15Z platform features OpenSDA, the NXP open-source hardware embedded serial and debug adapter running an open-source bootloader.

To begin, configure the jumpers on the FRDM-KE15Z Freedom System module properly. The following table lists the specific jumpers and their settings for the FRDM-KE15Z Freedom System module.

Table 10. FRDM-KE15Z jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J7	1-2	J10	1-2	J15	2-3
J8	1-2	J14	1-2		

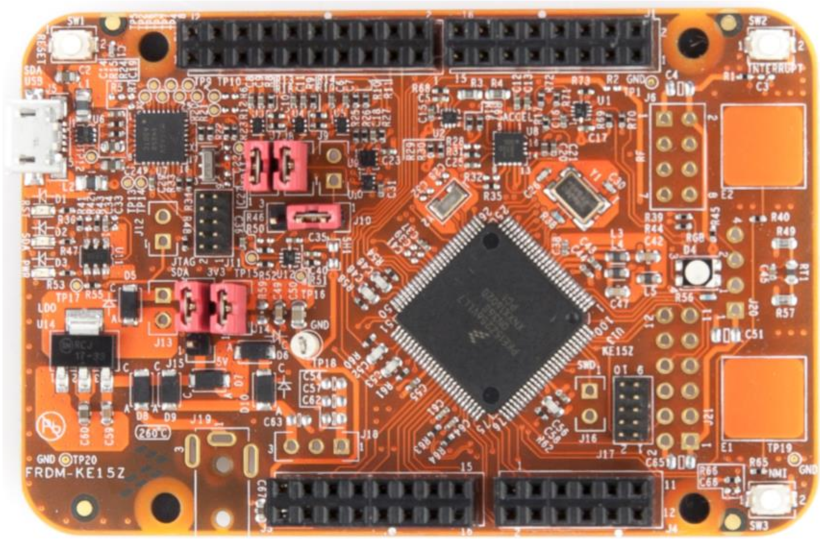

Figure 13. FRDM-KE15Z Freedom development board

4.3.5. NXP Freedom system assembling

1. Connect the FRDM-MC-LVPMSM shield on top of the FRDM-KVxxx board (there is only one possible option).
2. Connect the Linix motor three-phase wires into the screw terminals on the board.
3. Plug the USB cable from the USB host to the OpenSDA micro USB connector.
4. Plug the 24 V DC power supply to the DC Power connector.

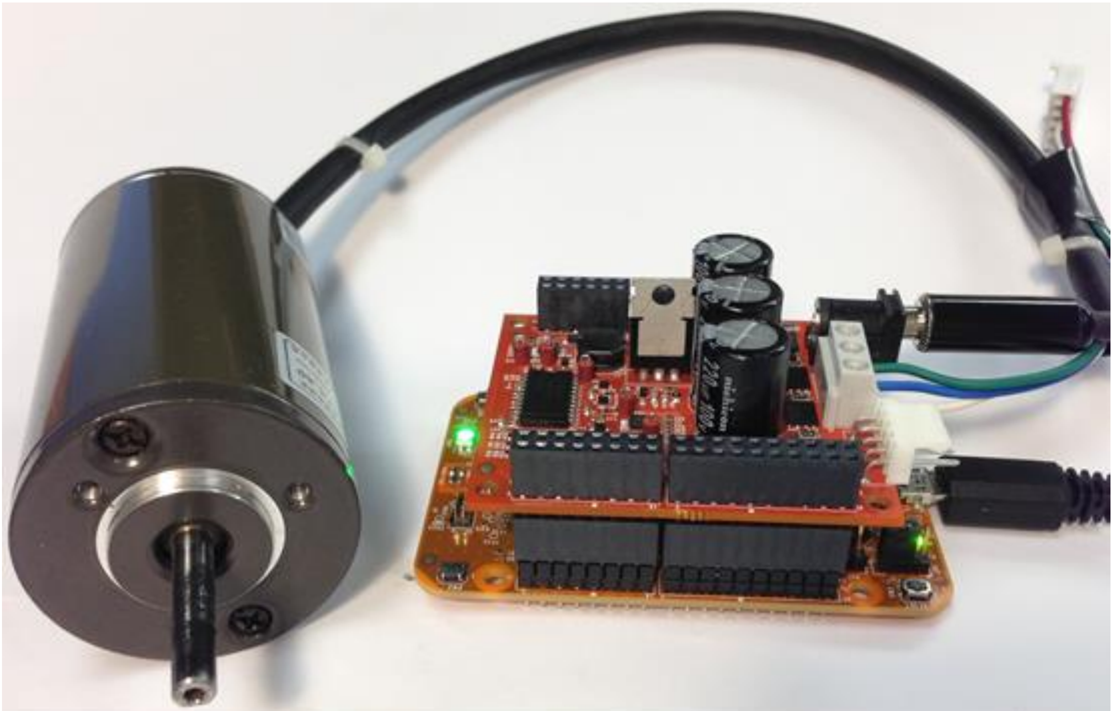

Figure 14. Assembled Freedom system

5. Project File Structure

The demo project folder (for example *<MCUXpressoSDK_install_folder>\boards\frdmkv10z\demo_apps\mc_pmsm*) contains these folders and files:

- *IAR* folder—contains the configuration files for IAR Embedded Workbench IDE. If IAR Embedded Workbench for ARM is installed on your computer, open the project using IAR IDE.
- *KDS* folder—contains the configuration files for KDS IDE and launch configuration for debuggers. If KDS is installed on your computer, open the project using KDS IDE.
- *MDK* folder—contains the configuration files for µVision Keil IDE. If Keil IDE is installed on your computer, open the project using Keil IDE.
- *Project files*—contains the device-specific files. They specify the peripheral initialization routines, motor definitions, and state machines. The source code contains a lot of comments. The functions of the particular files are explained in this list:
  - *m1_pmsm_appconfig.h*—contains the definitions of constants for the application control processes (parameters of the motor and regulators and the constants for other control-related algorithms).
  - *main.c*—contains the basic application initialization (enabling interrupts), subroutines for accessing the MCU peripherals, and interrupt service routines.
  - *mcdrv_< board&MCU >.c*—contains the motor-control driver peripherals' initialization functions, specific for the board and MCU used.
  - *mcdrv_< board&MCU >.h*—header file for *mcdrv_<board&MCU>.c*. This file contains the

macros for changing the PWM period and ADC channels assigned to the phase currents and board voltage.

- *pin_mux.c* – contains board initialization function for configuring pin routing. This file is generated with the Pins tool.
- *pin_mux.h* – header file for *pin_mux.c*.
- *board.c* – common MCUXpresso SDK file containing initialization of a debug console
- *board*.h – common MCUXpresso SDK file containing macros for specific board pinout
- *clock_config.c* – contains MCU clock configuration functions
- *clock_config.h* – header file for *clock_config.c*
- *readme.txt* – basic information about requirements, settings and demo.

The motor-control folder < *MCUXpressoSDK_install_folder*>*\middleware\motor_control\pmsm* contains these common source and header files used in all motor-control projects. The folder contains the subfolders common to the entire project in this package:

- *mc_algorithms*—contains the main control algorithms used to control the FOC and speed control loop.
- *mc_drivers*—contains the source and header files used to initialize and run motor-control applications.
- *mc_state_machine*—contains the software routines that are executed when the application is in a particular state or state transition.
- *state_machine*—contains the state machine functions for the Fault, Initialization, Stop, and Run states.

Each motor-control project is based on RTCESL (Real-Time Control Embedded Software Library) placed in the < *MCUXpressoSDK_install_folder*>*\middleware* folder. The library contains the functions used in the project. The *RTCESL* folder contains the library subfolders for the specific cores ("*cm0*", "*cm4*", and "*cm7*"). This subfolder includes the required header files and library files used in the project. RTCESL is fully compatible with the official release. The library names are changed for easier use in the available IDEs. See nxp.com/fslesl for more information about RTCESL.

6. Tools

Install the following software on your PC to run and control the PMSM sensorless application properly:

- Iar Embedded Workbench IDE v7.60 or higher
- Kinetis Design Studio IDE v3.2 or higher
- ARM-MDK - Keil µVision version 5.20

7. Building and Debugging Applications

The package contains projects for Kinetis Design Studio, IAR Embedded Workbench, and µVision Keil IDEs. Both of them are targeted for motor-control applications. The release configuration is the default one, and there are no special requirements needed to run and debug the demonstration applications.

7.1. IAR Embedded Workbench IDE

Use IAR Embedded Workbench IDE to compile and run the demonstration projects. The first step is to choose the demonstration, development board, and MCU. For example, to run a demonstration project for the Freedom development platform and Kinetis KV10 MCU, the project is located in the < *MCUXpressoSDK_install_folder>\boards\frdmkv10z\demo_apps\mc_pmsm\iar* folder, which contains all necessary files. Double-click "*mc_pmsm.eww*" to run this project. This figure shows the IAR workspace:

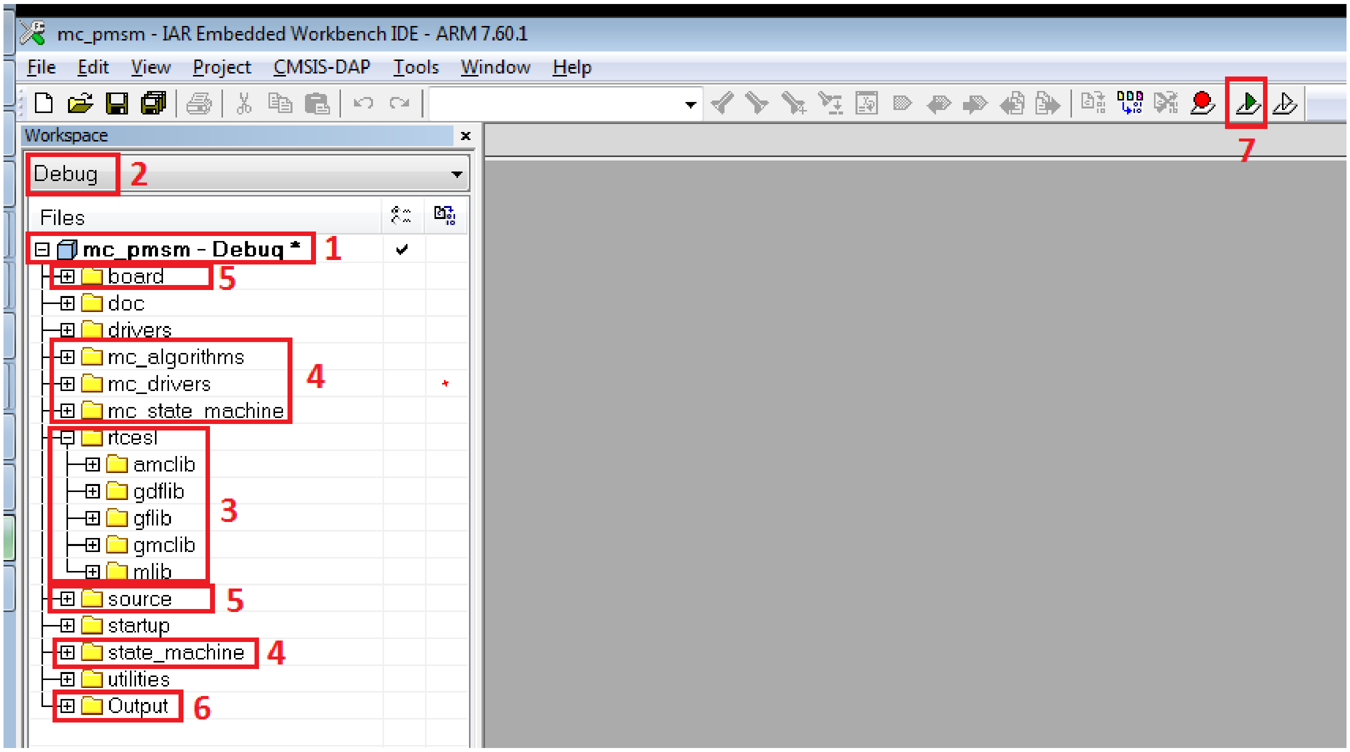

Figure 15. IAR Embedded Workbench

The project opened in IAR Embedded Workbench is fully configured and includes all source and header files required by the application, such as startup code, clock configuration, and peripherals' configuration. You can choose from two compiling conditions ("debug" or "release"), as shown in Figure 15, point 2. Each of the two conditions has its own setting:

- "debug"—used for debugging, the optimization has the "None – turned off" flag.
- "release"—used for releasing, the optimization has the "High – Highest optimization for speed" flag.

NOTE

The "debug" condition has the optimization turned off, and the output file may not fit into MCUs with a smaller flash (for example KV10Z32).

The source code shown in Figure 15 includes these source files and folders:

- Point 3—the RTCESL library source folder contains header files for the mathematical and control functions used in this project. The theory on using and applying these functions is described in the user's guides specific for each library. Find the user's guides at nxp.com/fslesl.

- Point 4—the board-independent source files contain the application source code. These files are placed in the < *MCUXpressoSDK_install_folder>\middleware\motor_control\pmsm* folder.
- Point 5—the device-specific files contain the application source code. These files are placed in the < *MCUXpressoSDK_install_folder>\boards\<board&MCU>\demo_apps\mc_pmsm* folder.

- Point 6—shows the output file generated by the compiler which is ready to use with the default debugger (P&E Micro OpenSDA). This debugger is set as default for the Tower System boards, and can be changed in the project options by right-clicking Point 1, selecting “Options”, and clicking “Debugger”. Start the project debugging by clicking Point 7 (Figure 15).

The installation package contains only the required project files. All cached files are deleted by default. From IAR IDE version 7.60 onwards, the updated P&E Micro OpenSDA driver requires selecting a valid device (see the following figure) during the first code build and loading to the MCU.

Figure 16. IAR select device alert

After you click the “OK” button, the “P&E Connection Manager” window opens (see the following figure). Click the “Select New Device” button and select the particular MCU according to Table 11.

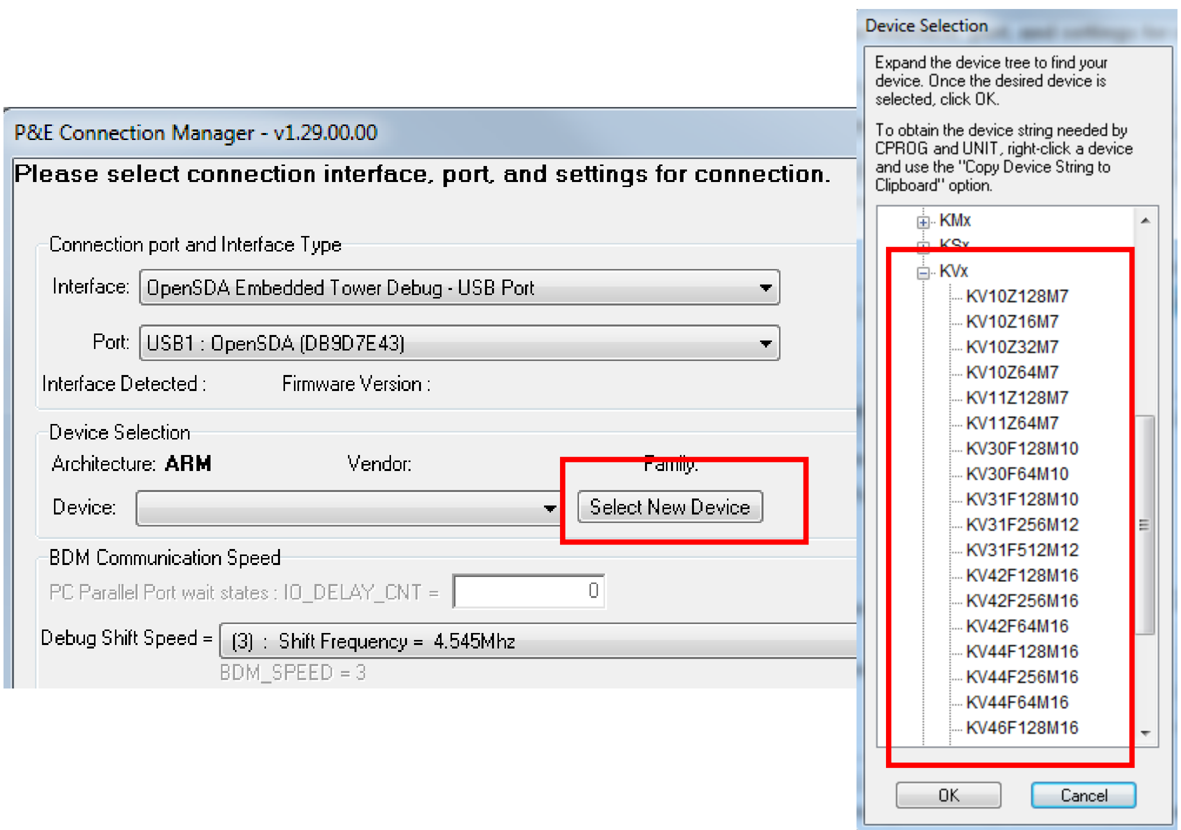

Figure 17. P&E Connection Manager

Table 11. Supported development platforms

—		Platform	
		Tower System	**Freedom**
MCU	**KV10Z**	KV10Z32M7	KV10Z32M7
	KV31F	KV31F512	KV31F512
	KV11Z	KV11Z128M7	—
	KV46F	KV46F256	—
	KV58F	KV58F1M	—
	KE15Z	—	KE15Z256
	KE18F	KE18F160M	—

After you select the right MCU, click the "OK" button and the setting is saved into the cache project files. The next time the code is built, the prompt window does not appear.

7.2. Kinetis Design Studio (KDS) IDE

Kinetis Design Studio (KDS) is an IDE tool that you can use to develop and test software for NXP MCUs. It supports a wide range of Kinetis devices, such as the powerful K series, low-power KL series, and KV series targeted for motor control. KDS includes the tools for compiling, linking, and debugging of source code. KDS supports a wide range of debuggers, such as P&E Micro or J-Link (and others). Download the latest release of KDS from the official NXP website nxp.com/kds. For installation and configuration, see *Kinetis Design Studio V3.2.0 User's Guide* (document KDSUG).

To open a demonstration, choose the development board and MCU. For example, if you want to open a demonstration project for the Freedom development platform and Kinetis KV10 MCU, locate the project in *< MCUXpressoSDK_install_folder>\boards\frdmkv10z\demo_apps\mc_pmsm\kds*, run KDS IDE from the default installation path or from the installed programs, and perform these steps:

- Click the "File" menu in the top-left corner of the IDE, and click "Import…".
- In the window that appears, highlight "Existing Projects into Workspace" in the "General" folder, and click the "Next" button:

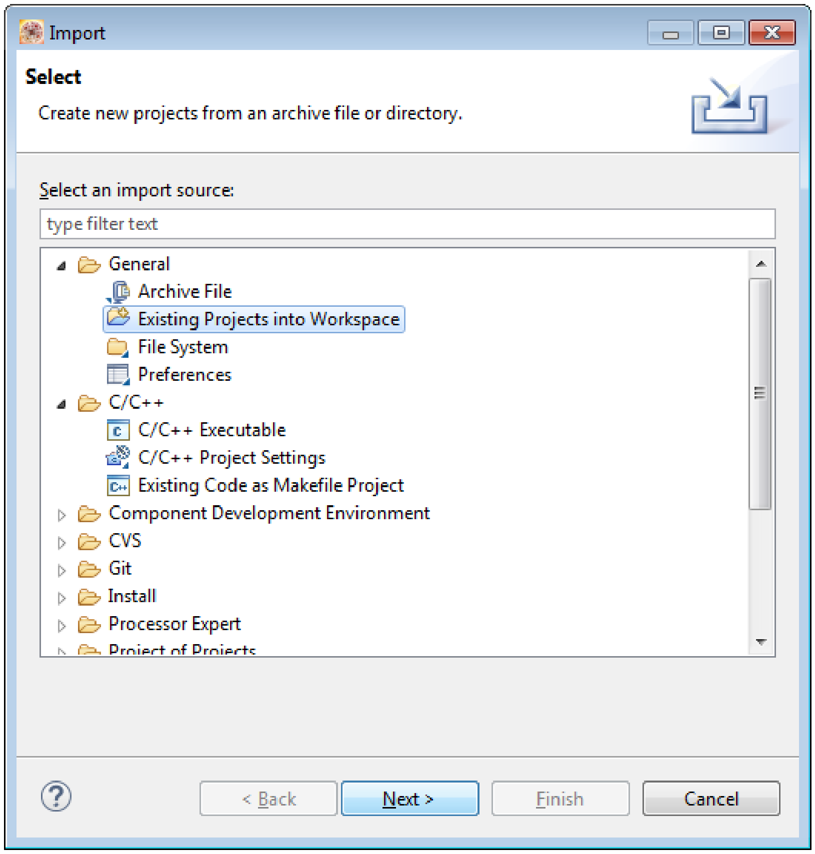

Figure 18. KDS—importing project

- The “Import” window opens. Click the “Browse” button, and then locate the project *< MCUXpressoSDK_install_folder>\boards\frdmkv10z\demo_apps\mc_pmsm\kds*. Click the “OK” button:

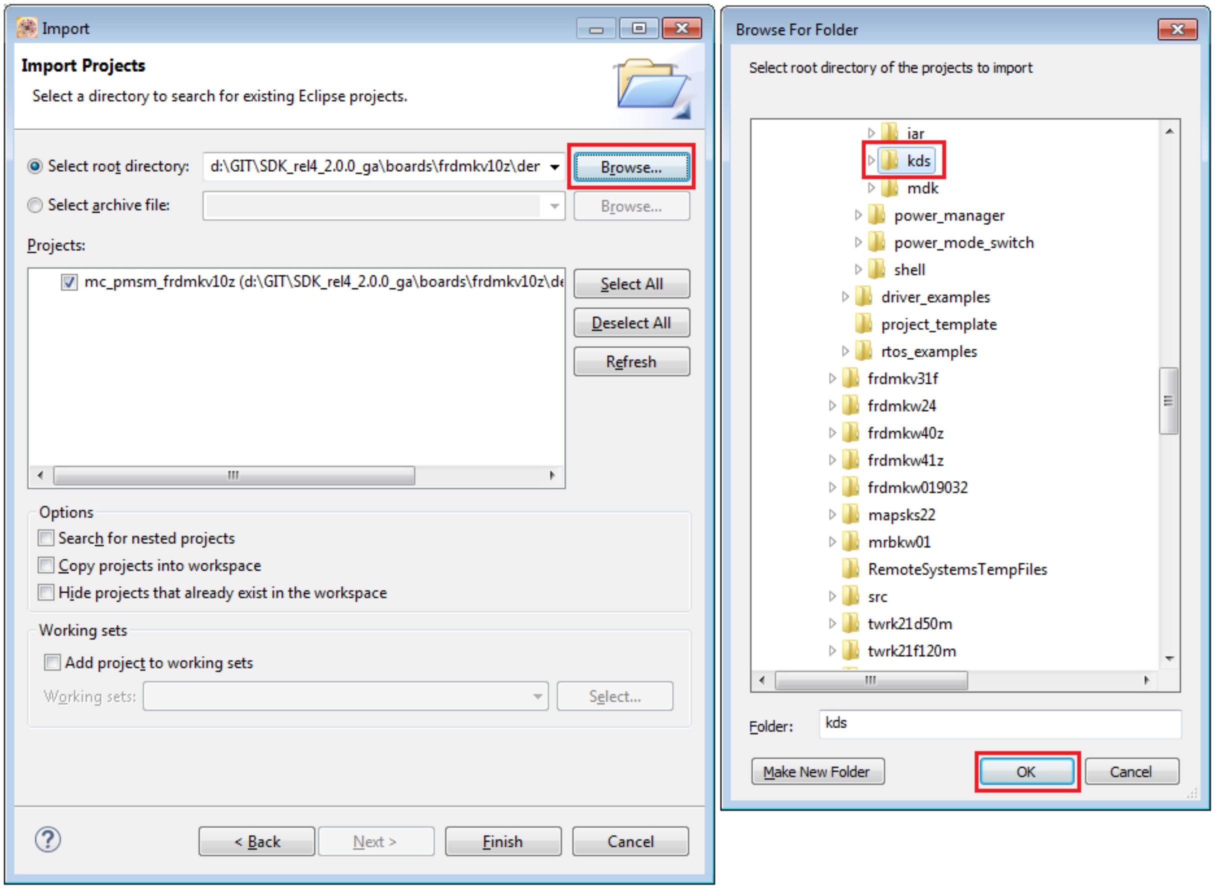

Figure 19. KDS import project

- Confirm the project by clicking “Finish”.

The project is now imported in Kinetis Design Studio (Figure 20). Point 1 shows the imported project in “Project Explorer”, and Point 2 shows the source code of this project. Build the project by clicking the “build” icon (Point 3) where the “release” configuration is set as default. You can change the configuration to “debug”. Each of these two conditions has its own setting:

- “debug”—used for debugging, optimization has the “None – turned off” flag.
- “release”—used for releasing, optimization has the “High – Highest optimization for speed” flag.

NOTE

The “debug” condition has the optimization turned off, and the output file may not fit into MCUs with a smaller flash (for example KV10Z32).

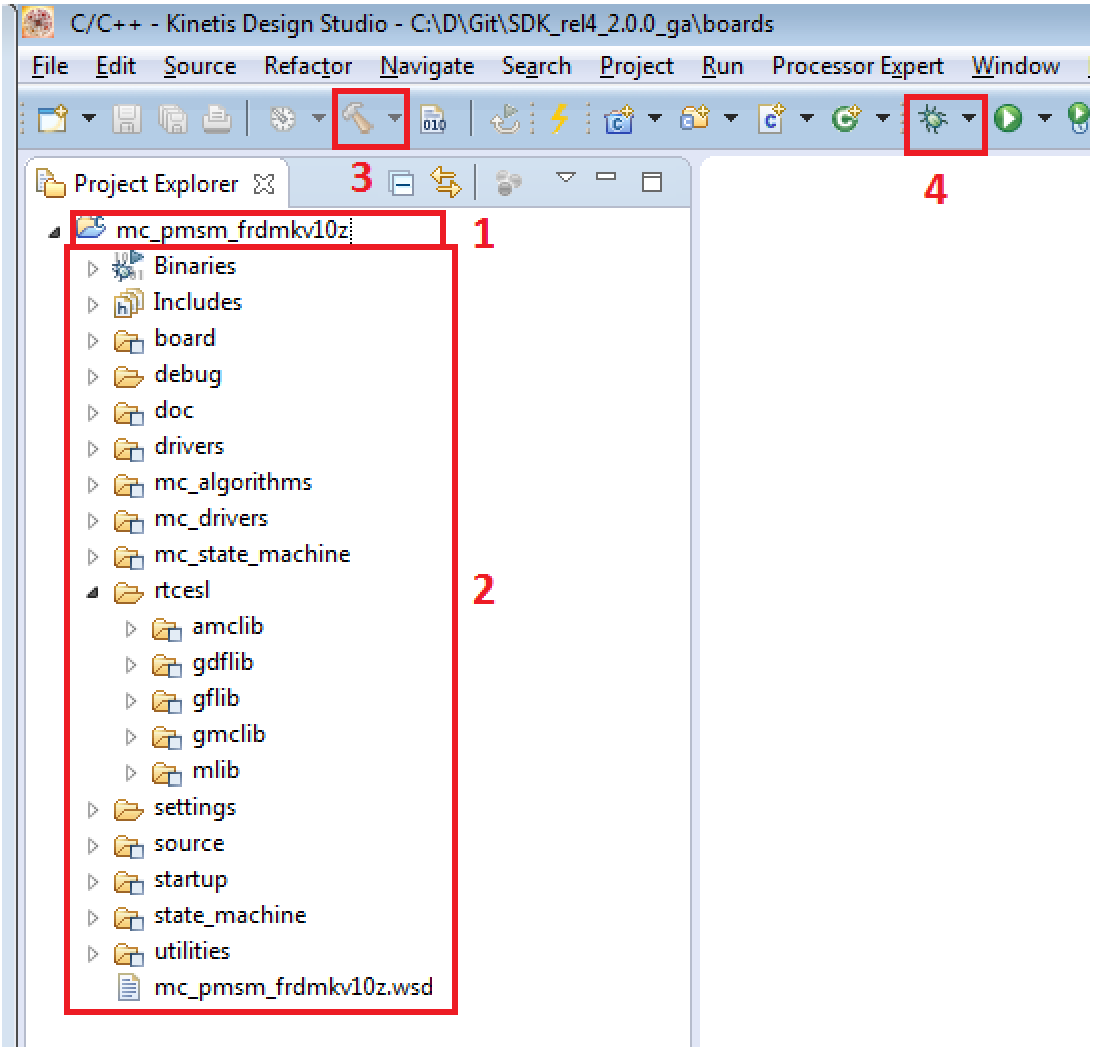

Figure 20. KDS PMSM project

When the project is compiled, either the *debug* or *release* folder is created (depending on the build condition selected). An **.elf* binary file is created in one or both of these folders. Use the debugger (Point 4 in Figure 20). You can use the predefined debugger (P&E Micro OpenSDA) or choose a different debugger from the menu. In the top list menu, select "Run-> Debug Configuration" to define a different type of debugger, or change the conditions of debugging (such as the optimization level).

7.3. ARM-MDK Keil µVision IDE

ARM-MDK Keil µVision IDE (Keil) is a software development and testing tool for various MCUs. It supports a wide range of Kinetis devices, such as the powerful K series, low-power KL series, and KV series targeted for motor control. Keil includes tools for compiling, linking, and debugging of source code. Keil supports a wide range of debuggers, such as P&E Micro or J-Link™ (and others). Download the latest release of Keil from the official Keil website www2.keil.com/mdk5/uvision.

To open the demonstration, choose the development board and MCU. For example, if you want to open a demonstration project for the Freedom development platform and Kinetis KV10 MCU, locate the project at < *MCUXpressoSDK_install_folder*>*\boards\frdmkv10z\demo_apps\mc_pmsm\mdk*:

- Double-click the *mc_pmsm.uvprojx* project file.
- The project opens. Click the “Build” button (Point 1) to compile the project. Then click the “Download” button (Point 2) to download the code to the target. Then click the “Debug” button to enter the debug session (Point 3):

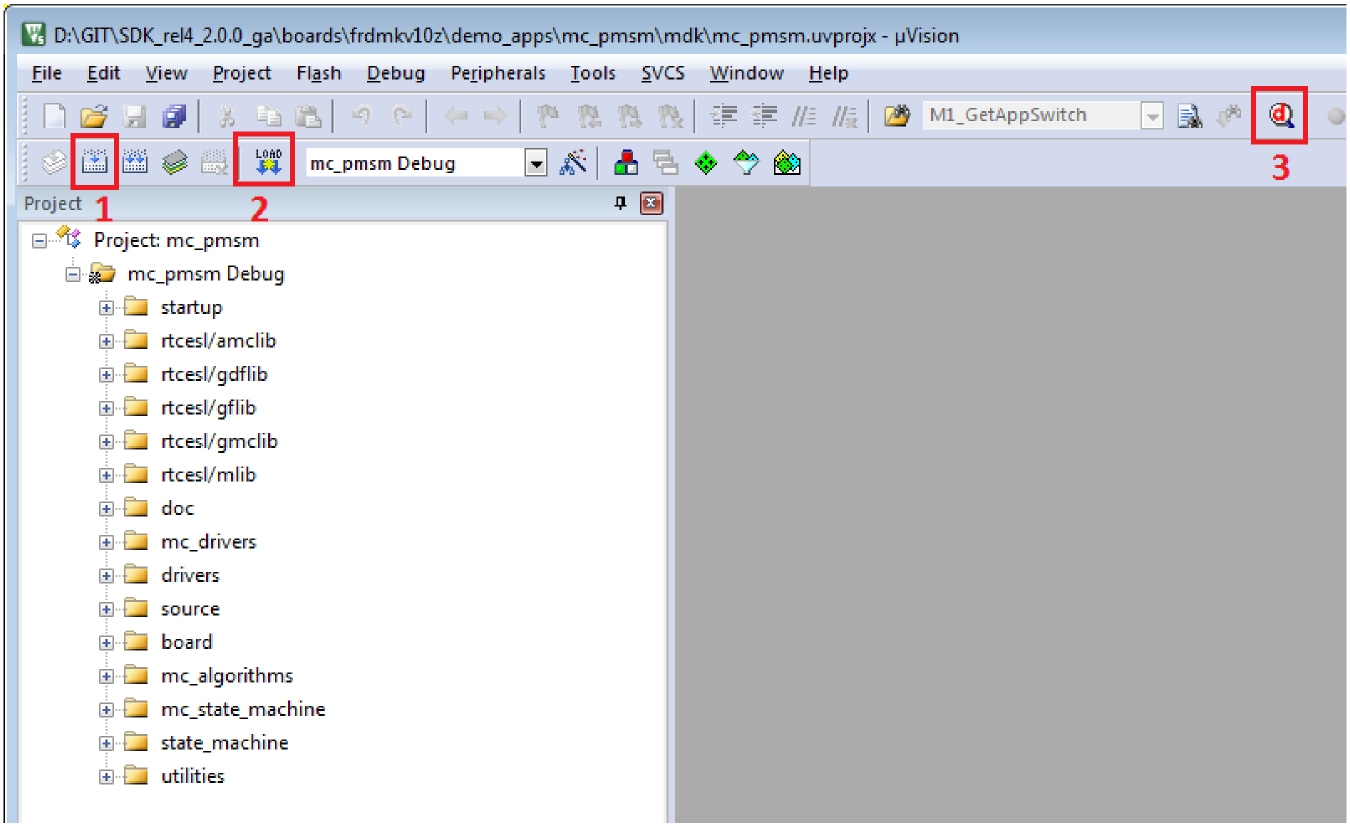

Figure 21. Keil PMSM project

- Before downloading the code to the target, select a proper target device (if using the P&E OpenSDA debugger):

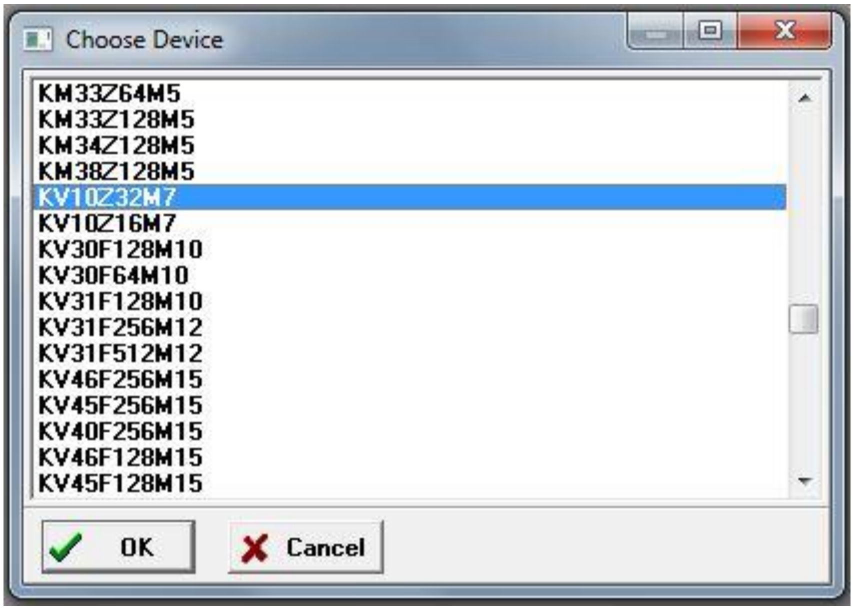

Figure 22. P&E debugger—target selection

- There are two project configurations:
  - "debug"—used for debugging, the optimization has the "None – turned off" flag.
  - "release"—used for releasing, the optimization has the "High – Highest optimization for speed" flag.
- Use the predefined debugger (P&E Micro OpenSDA) or choose a different debugger from the menu. In the top list menu, select "Run-> Debug Configuration" to define a different type of debugger, or change the conditions of debugging (such as the optimization level).

8. User Interface

The application contains the demo application mode to demonstrate the motor rotation. Operate it using the user button. The Tower System and Freedom boards include a user button associated with a port interrupt (generated whenever one of the buttons is pressed). At the beginning of the ISR, a simple logic executes, and the interrupt flag clears. When you press the SW2 button, the demo mode starts. When you press the same button again, the application stops and transitions back to the STOP state. There is also an LED indication of the current application state. The green continuous LED indicates that the application is in the RUN state, the flashing LED indicates the FAULT state, and the LED off (or red LED) indicates the STOP state.

Control the application using the buttons on NXP Kinetis V Tower and Freedom development boards.

8.1. Control button

When you press the SW2 button, the demonstration mode switches on (or off, if it is already switched on).

9. Acronyms and Abbreviations

Table 12. Acronyms and abbreviations

Term	Meaning
AC	Alternating Current
AN	Application Note
DRM	Design Reference Manual
FOC	Field-Oriented Control
MCAT	Motor Control Application Tuning tool
MCU	Microcontroller
MSD	Mass Storage Device
PMSM	Permanent Magnet Synchronous Motor

10. References

The following references are available at nxp.com:

- *Sensorless PMSM Field-Oriented Control* (document DRM148).
- *Tuning Three-Phase PMSM Sensorless Control Application Using MCAT Tool* (document AN4912).
- *Automated PMSM Parameters Identification* (document AN4986).
- *Motor Control Application Tuning (MCAT) Tool for 3-Phase PMSM* (document AN4642).

11. Revision History

The following table summarizes the changes done to this document since the initial release.

Table 13. Revision history

Revision number	Date	Substantive changes
1	06/2016	Initial release.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number: PMSMDEMOUG
Rev. 1
06/2016