PMSM Control Demo User's Guide

1. Introduction

This document provides instructions for running and controlling Permanent Magnet Synchronous Motor (PMSM) projects with Freedom, Tower System, and High Voltage Power development boards shown in Table 1. The required software, hardware setup, jumper settings, project arrangement, and user interface is described in the following sections. For more information, see Section 8, “References” or visit www.nxp.com/motorcontrol_pmsm.

Contents

1. Introduction 1
2. Supported development boards 2
3. Motor control vs. SDK peripheral drivers 2
4. Hardware setup 3
   4.1. Linix 45ZWN24-40 motor 3
   4.2. MIGE 60CST-MO1330 motor 3
   4.3. Teknic M-2310P motor 4
   4.4. Running PMSM application on Tower System 5
   4.5. NXP Freedom development platform 12
   4.6. High-Voltage Platform 18
   4.7. NXP i.MX RT evaluation kits 18
5. Project file structure 26
6. User interface 27
   6.1. Control button 28
   6.2. Remote control using FreeMASTER 28
   6.3. FreeMASTER TSA and user variables addition to FreeMASTER watch 29
   6.4. Control page 30
7. Acronyms and abbreviations 31
8. References 31
9. Revision history 31

2. Supported development boards

There are development boards supported for Kinetis KV, KE, and i.MX RT 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 the Freedom development platform are targeted for low-voltage and low-power applications with PMSM control type. The High-Voltage Platform (HVP) is designed to drive high-voltage (115/220 V) applications with up to 1 kW of power.

Table 1. Supported development boards

—		Platform			
		FRDM	**TWR**	**HVP**	**EVK**
Power Stage		FRDM-MC-LVBLDC	TWR-LV3PH	HVP-MC3PH	—
MCU	**KV11**	FRDM-KV11Z	TWR-KV11Z75M	—	—
	KV31	FRDM-KV31F	TWR-KV31F120	HVP-KV31F512	—
	KV46	—	TWR-KV46F150M	HVP-KV46F150	—
	KV58	—	TWR-KV58F220M	HVP-KV58F220M	—
	KE15Z	FRDM-KE15Z	—	—	—
	KE18F	—	TWR-KE18F	HVP-KE18F	—
	KE16Z	FRDM-KE16Z	—	—	—
	i.MX RT 1020	—	—	—	IMXRT1020-EVK
	i.MX RT 1050	—	—	—	IMXRT1050-EVK
	i.MX RT 1060	—	—	—	IMXRT1060-EVK

3. Motor control vs. SDK peripheral drivers

The motor-control examples use the MCUXpresso SDK peripheral drivers to configure the general peripherals (such as clocks, SPI, SIM, and ports). However, motor control requires critical application timing because most control algorithm run in a 100-µs loop. To optimize the CPU load, most peripheral hardware features are implemented for the PWM signal generation, analog signal sampling and synchronization between the PWM and ADC units.

The standard SDK peripheral drivers do not support configuration and handling of all required features. The motor-control drivers are designed to configure the critical MC peripherals (eflexPWM, FTM, ADC, and PDB).

It is highly recommended not to modify the default configuration of the allocated MC peripherals due to a 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, Freedom, or EVK development platforms with 24 V Linix Motor in the default configuration. The HVP platform runs with the default configuration for the MIGE 60CST-MO1330 motor.

4.1. Linix 45ZWN24-40 motor

The Linix 45ZWN24-40 motor (described in Table 2) is a low-voltage 3-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. MIGE 60CST-MO1330 motor

The MIGE 60CST-MO1330 motor (described in Table 3) is primarily used for the PMSM sensorless application but you can also use it for the BLDC sensorless application. You can also adapt the application to other motors, just by defining and changing the motor-related parameters. The motor is connected directly to the high-voltage development board via a flexible cable connected to the 3-wire development board connector.

Table 3. MIGE 60CST-MO1330 motor parameters

Characteristic	Symbol	Value	Units
Rated voltage	V_t	220	V
Rated speed @ V_t	—	3000	RPM
Rated power	P	400	W
Number of pole pairs	Pp	4	—

Figure 2. MIGE motor

4.3. Teknic M-2310P motor

The Teknic M-2310P-LN-04K motor is a low-voltage 3-phase permanent-magnet motor used in PMSM applications. The motor has two feedback sensors (hall and encoder). For information on the wiring of feedback sensors, see the datasheet on the manufacturer web page. The motor parameters are listed in Table 4.

Table 4. Teknic M-2310P motor parameters

Characteristic	Symbol	Value	Units
Rated voltage	Vt	40	V
Rated speed @ Vt	—	6000	RPM
Rated torque	T	0.274	Nm
Rated power	P	170	W
Continuous current	Ics	7.1	A
Number of pole pairs	pp	4	—

Figure 3. Teknic M-2310P permanent magnet synchronous motor

For the sensorless control mode, you need only the power input wires. If used with the hall or encoder sensors, connect also the sensor wires to the NXP Freedom power stage.

4.4. 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-KV11Z75M, TWR-KV31F120M, TWR-KV46F150M, or TWR-KV58F220M) or Tower board with a Kinetis E series MCU (TWR-KE18F).
- 3-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.

4.4.1. 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 5. 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.

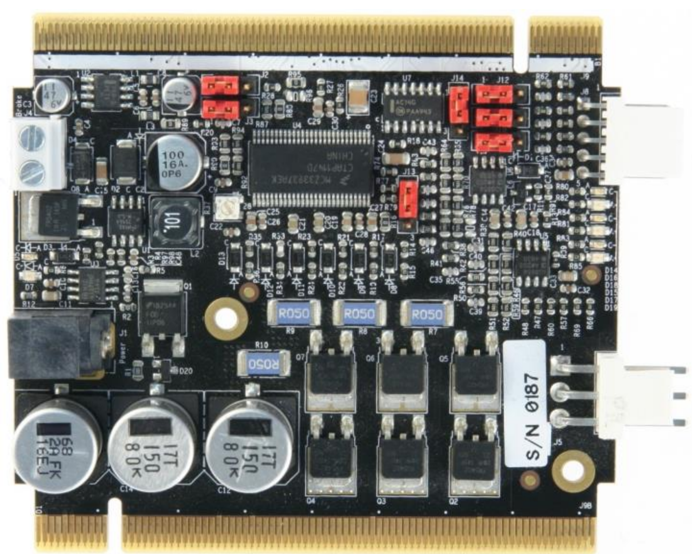

Figure 4. TWR-LV-MC3PH jumper settings

4.4.2. 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. Table 6 lists the specific jumpers and their settings for the TWR-KV11Z75M Tower System module.

Table 6. 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	—	—

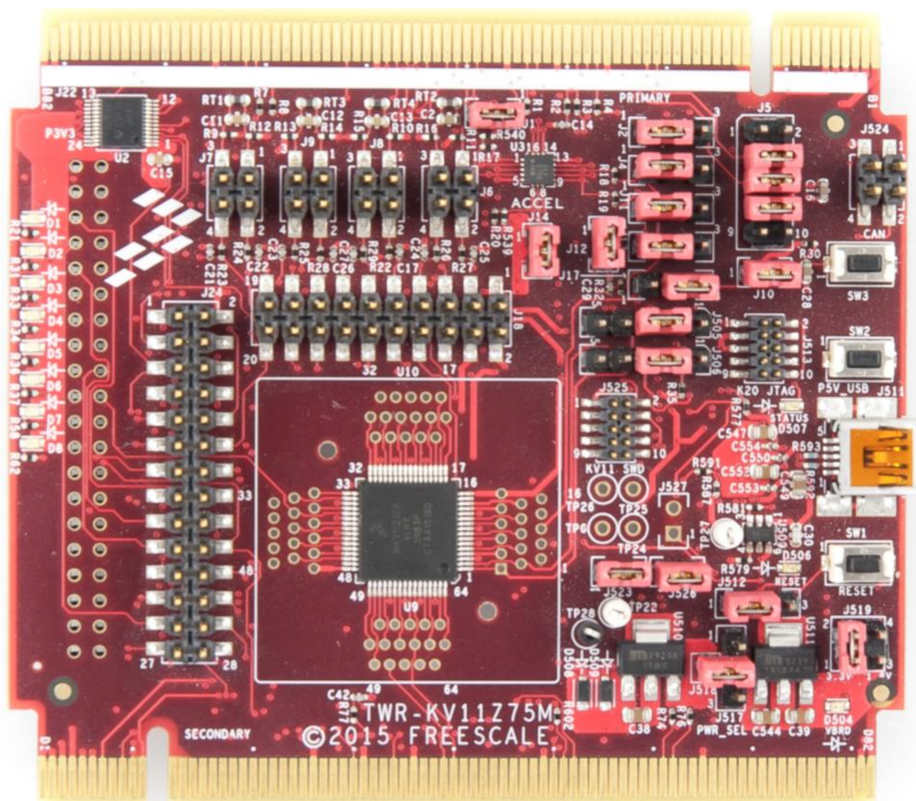

Figure 5. TWR-KV11Z Tower System module

4.4.3. 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. Table 7 lists the specific jumpers and their settings for the TWR-KV31F120M Tower System module.

Table 7. 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

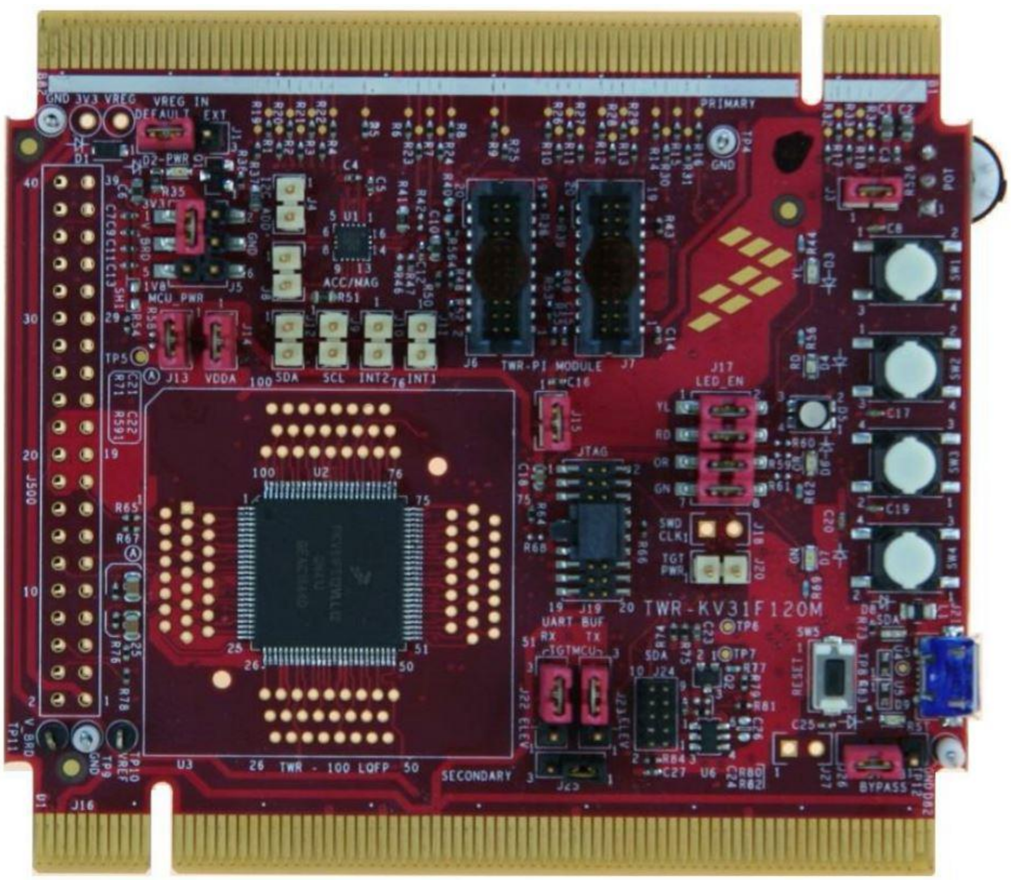

Figure 6. TWR-KV31F Tower System module

4.4.4. 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. Table 8 lists the specific jumpers and their settings for the TWR-KV46F150M Tower System module.

Table 8. 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	J520	1-2	J519	3-4
J520	1-2	J521	1-2	J522	1-2

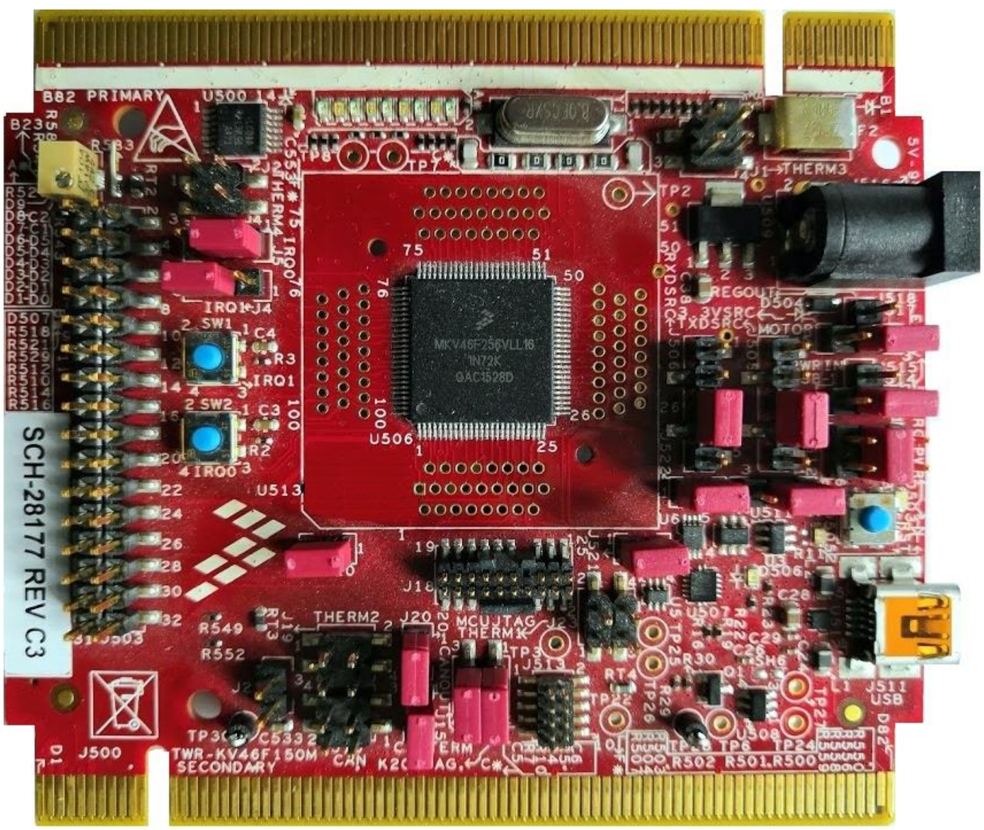

Figure 7. TWR-KV46F Tower System module

4.4.5. 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. Table 9 lists the specific jumpers and their settings for the TWR-KV58F220M Tower System module.

Table 9. 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	—	—

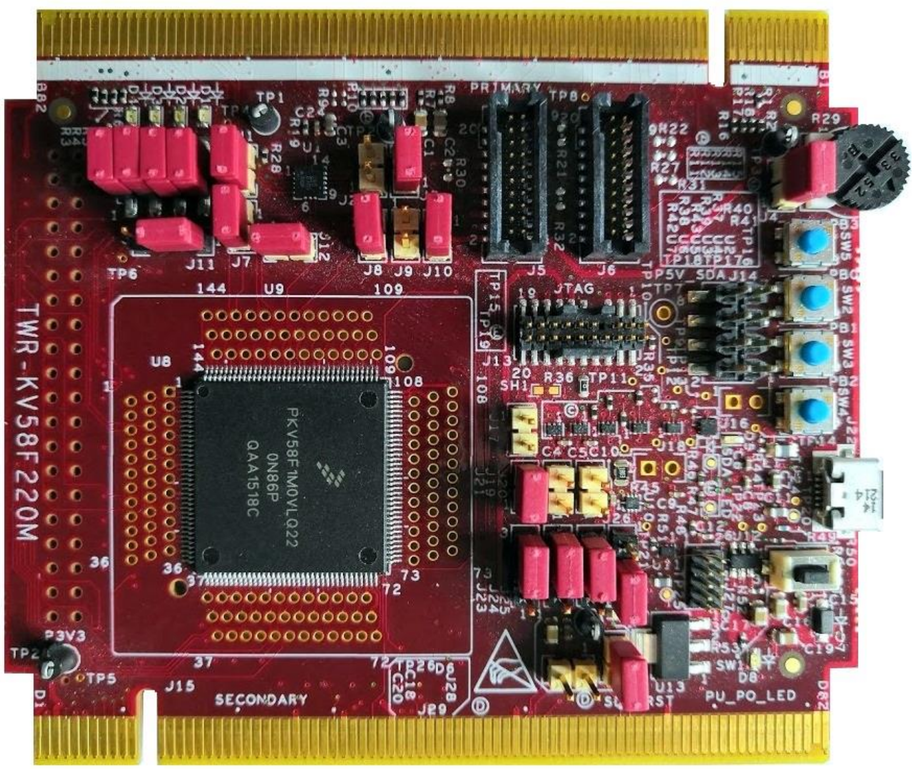

Figure 8. TWR-KV58F Tower System module

4.4.6. 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. Table 10 lists the specific jumpers and their settings for the TWR-KE18F Tower System module.

Table 10. 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	2-3	J13	1-2	J22	1-2
J7	2-3	J16	1-2	J23	open
J8	open	J17	1-2	J24	1-2

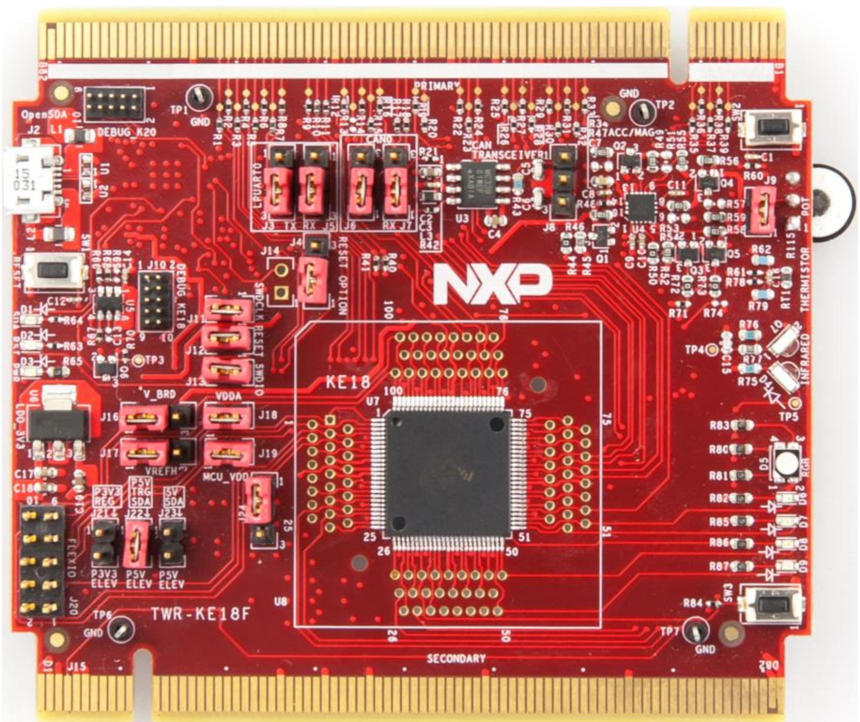

Figure 9. TWR-KE18F Tower System module

4.4.7. 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.

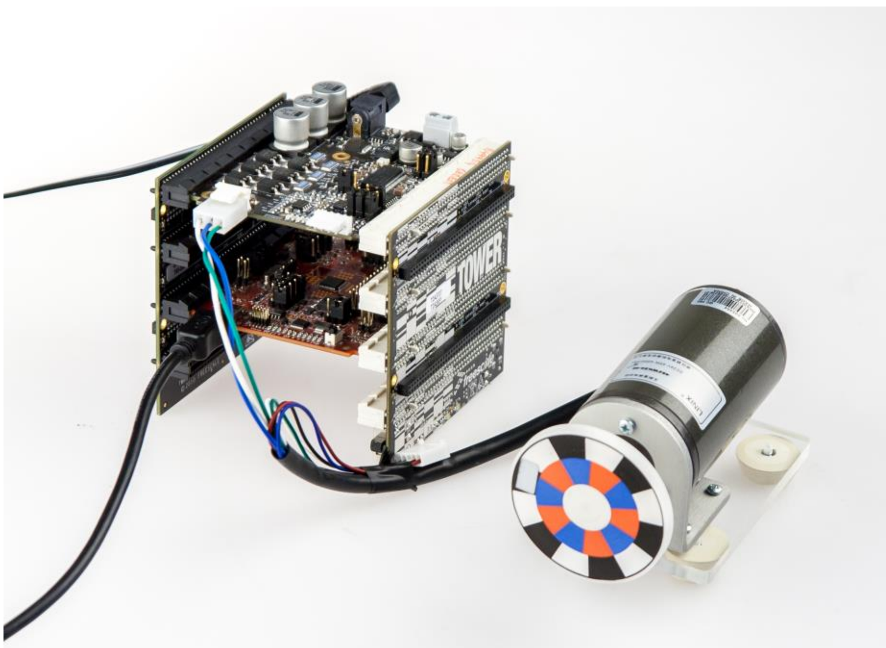

Figure 10. Assembled Tower System

4.5. 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-KV11Z or FRDM-KV31F) or Freedom board with a Kinetis E series MCU (FRDM-KE15Z, FRDM-KE16Z).
- 3-phase low-voltage power Freedom shield (FRDM-MC-LVPMSM) with included Linix motor.

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

4.5.1. 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, FRDM-KV11Z, FRDM-KE15Z, and FRDM-KE16Z).

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

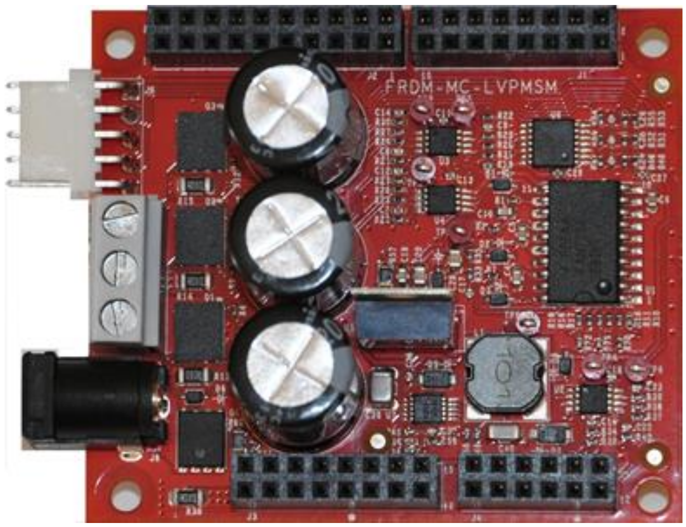

Figure 11. FRDM-MC-LVPMSM

4.5.2. FRDM-KV11Z board

The FRDM-KV11Z board is a low-cost development tool for the Kinetis V series KV1x MCU family built upon the Arm Cortex-M0+ processor. The FRDM-KV11Z board hardware is form-factor compatible with the Arduino R3 pin layout, providing a broad range of expansion board options. The FRDM-KV11Z platform features OpenSDA, the open-source hardware embedded serial and debug adapter running an open-source bootloader.

To begin, configure the jumpers on the FRDM-KV11Z Freedom System module properly. Table 11 lists the specific jumpers and their settings for the FRDM-KV11Z Freedom System module.

Table 11. FRDM-K11Z jumper settings

Jumper	Setting
J10	1-2

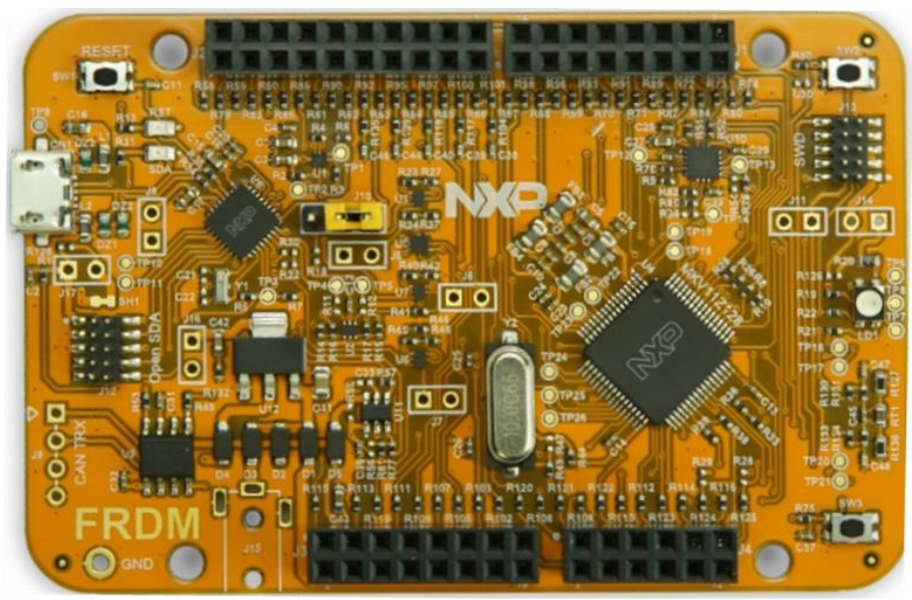

Figure 12. FRDM-KV11Z Freedom development board

4.5.3. 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.

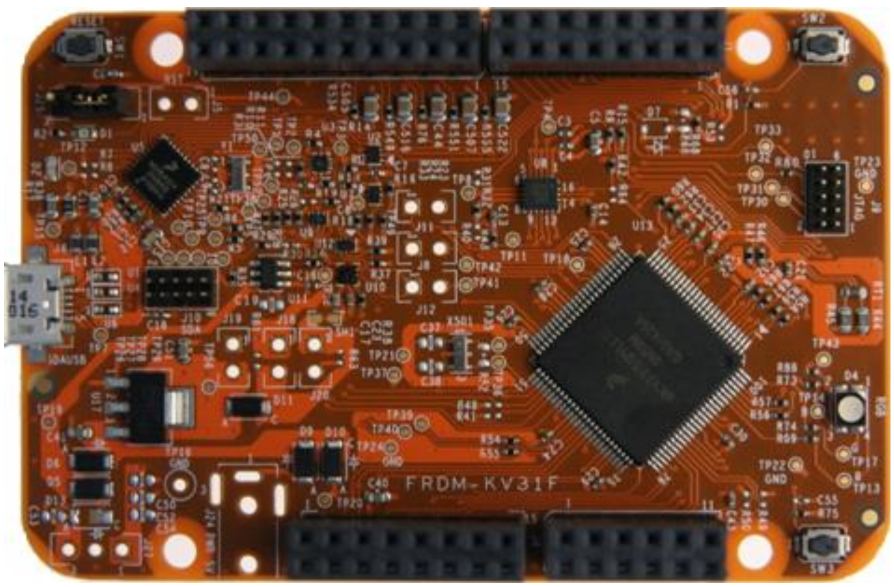

Figure 13. FRDM-KV31F development board

4.5.4. 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. Table 12 lists the specific jumpers and their settings for the FRDM-KE15Z Freedom System module.

Table 12. 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		

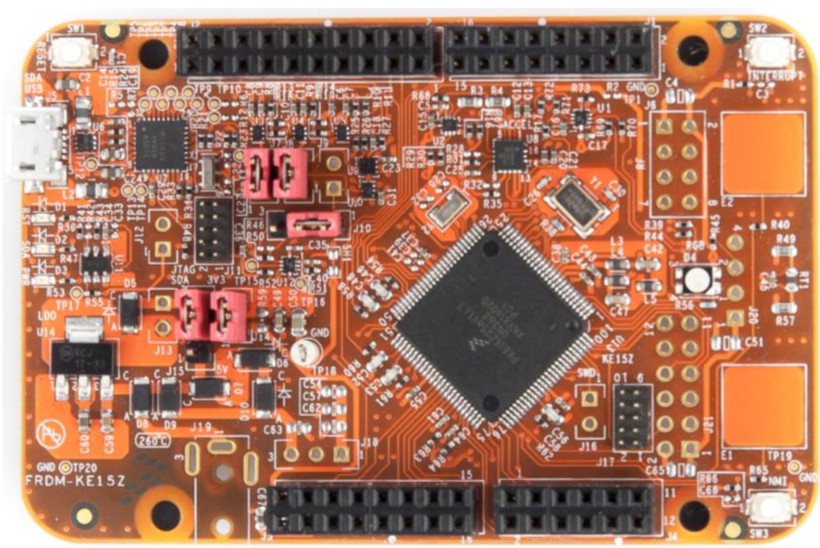

Figure 14. FRDM-KE15Z Freedom development board

4.5.5. FRDM-KE16Z board

The FRDM-KE16Z is a low-cost development tool for Kinetis KE1xZ family of MCUs built around the Arm Cortex-M0+ core. The FRDM-KE16Z hardware is form-factor compatible with the Arduino R3 pin layout, providing a broad range of expansion board options. The FRDM-KE16Z platform features OpenSDA, the NXP open-source hardware embedded serial and debug adapter running an open-source bootloader.

To begin, configure the jumpers and 0-Ω resistors on the FRDM-KE16Z Freedom System module properly. Table 13 lists the specific jumpers and Table 14 lists the 0-Ω resistor position settings for the FRDM-KE16Z Freedom System module.

Table 13. FRDM-KE16Z jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J7	1-2	J10	1-2	J19	2-3
J8	1-2	J17	1-2		

Table 14. FRDM-KE16Z 0-Ω resistor settings

MCU signal	Resistor	Add/remove
PTB4/RGB_GREEN	R24	Remove
PTB4/MC_PWM_CT	R25	Add
PTB5/RGB_RED	R28	Remove
PTB5/MC_PWM_CB	R29	Add
PTC0/GES_R0	R110	Remove
PTC0/MC_PWM_AT	R111	Add
PTC1/GES_R1	R8	Remove
PTC1/MC_PWM_AB	R13	Add
PTD1/RGB_BLUE	R6	Remove
PTD1/MC_PWM_BB	R9	Add
PTC0/GES_R0	R112	Remove
PTB2/MC_VOLT_DCB	R113	Add

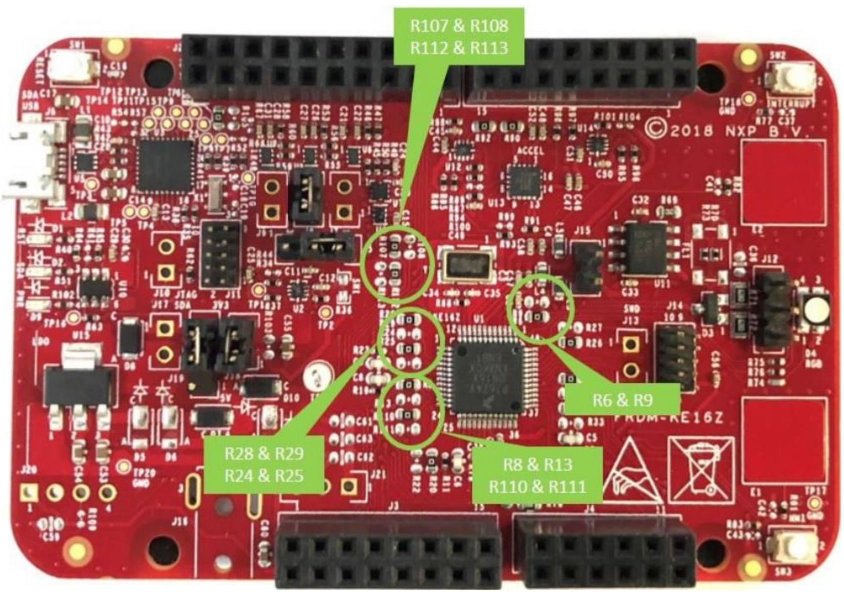

Figure 15. FRDM-KE16Z Freedom development board

4.5.6. 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 3-phase wires to 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.

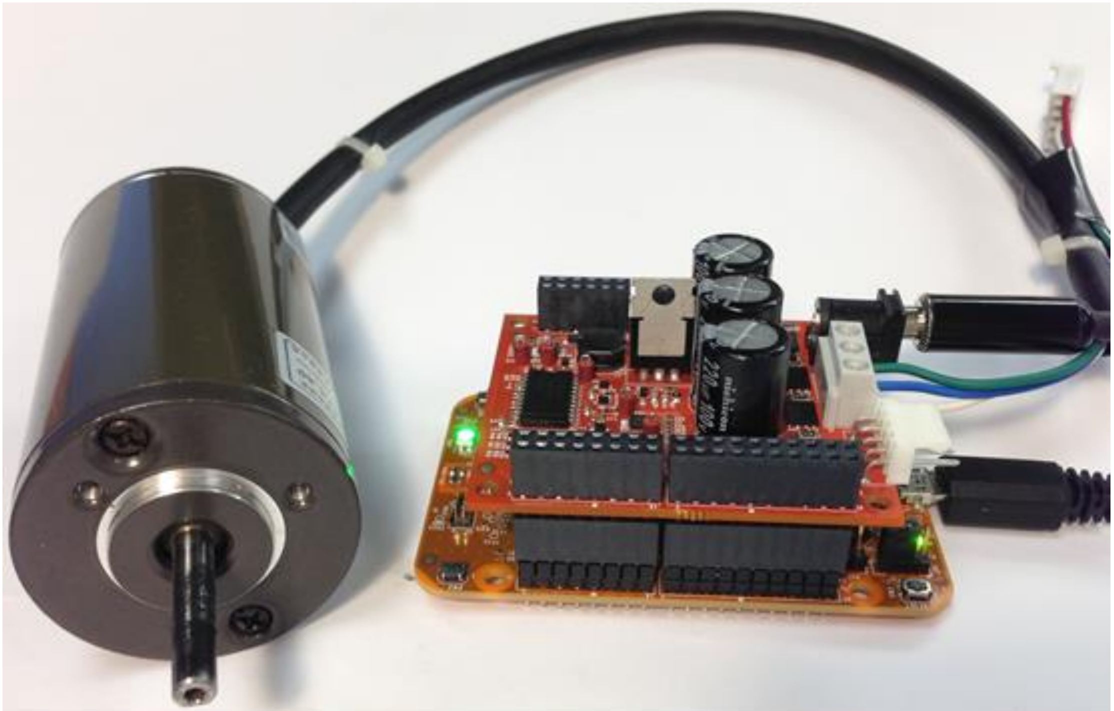

Figure 16. Assembled Freedom system

4.6. High-Voltage Platform

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

- Kinetis KV31F High-Voltage Daughter Board (HVP-KV31F512), Kinetis KV46F High-Voltage Daughter Board (HVP-KV46F150), Kinetis KV58F220M High-Voltage Daughter Board (HVP-KV58F220M), or Kinetis KE18F High-Voltage Daughter Board (HVP-KE18F).
- High-Voltage Platform (HVP-MC-3PH) (motor not included).

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

4.7. NXP i.MX RT evaluation kits

The MIMXRT EVK boards are a platform designed to showcase the most common features of the i.MX RT10xx processor in a small, low-cost package. The IMXRT10xx-EVK board is an entry-level development board which helps you to become familiar with the processor before investing a large amount of resources into more specific designs. The EVK board provides various memory types, particularly the 64-Mbit Quad SPI flash and the 512-Mbit hyper flash.

To run the PMSM application on the NXP i.MX RT evaluation kits, you need these boards:

- Evaluation kits with the i.MX RT crossover processor series (MIMXRT1020-EVK, MIMXRT1050-EVK, or MIMXRT1060-EVK).
- 3-phase low-voltage power Freedom shield (FRDM-MC-LVPMSM) with the Teknic motor.

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

4.7.1. MIMXRT1020-EVK

MIMXRT1020-EVK is a 2-layer, low-cost, hole-through, USB-powered PCB. It features the i.MX RT1020 crossover processor in a LQFP144 package, featuring NXP's advanced implementation of the Arm Cortex-M7 core. This core operates at speeds of up to 500 MHz to provide high CPU performance and the best real-time response.

Table 15. MIMXRT1020-EVK jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J1	5-6	J8	1-2	J26	1-2
J3	1-2	J21	open	J27	1-2
J4	1-2	J22	1-2	J28	1-2
J5	1-2	J24	1-2	J29	open
J6	1-2	J25	1-2	J37	1-2

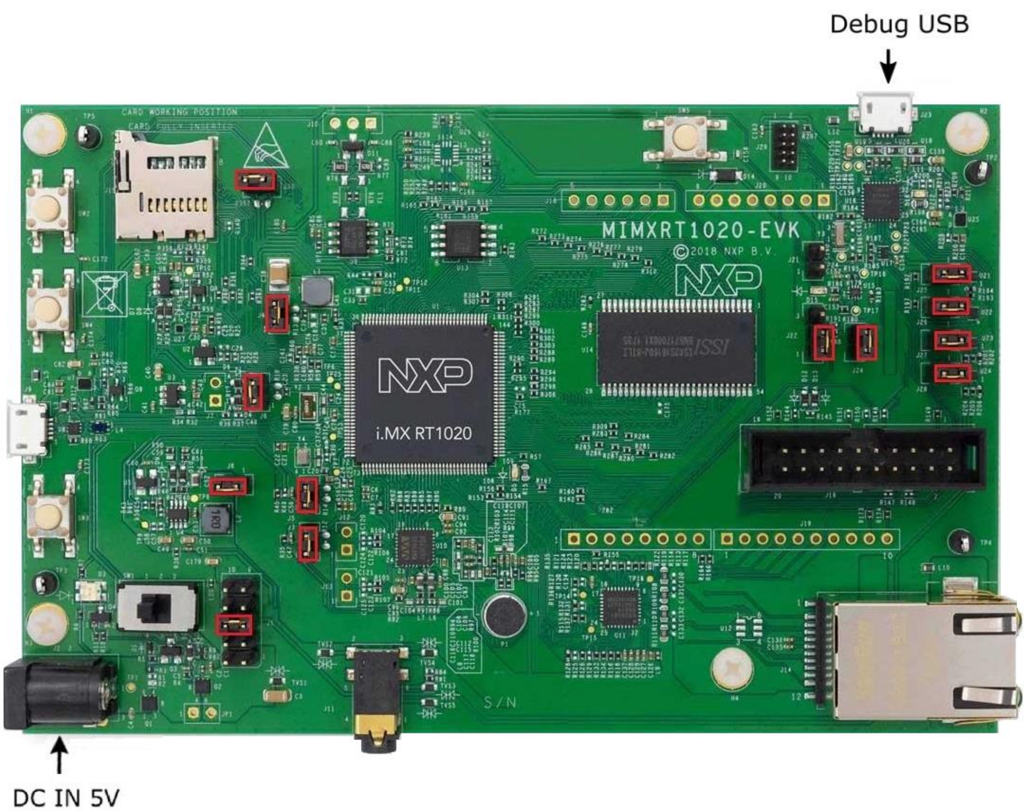

Figure 17. MIMXRT1020-EVK board with highlighted jumper settings

The ADC pins are shared with the next peripherals. Therefore, it is needed to remove resistors R65, R68, R69, and R74 from the board. These resistors are located on the top side of the EVK board.

For more details, see the schematic.

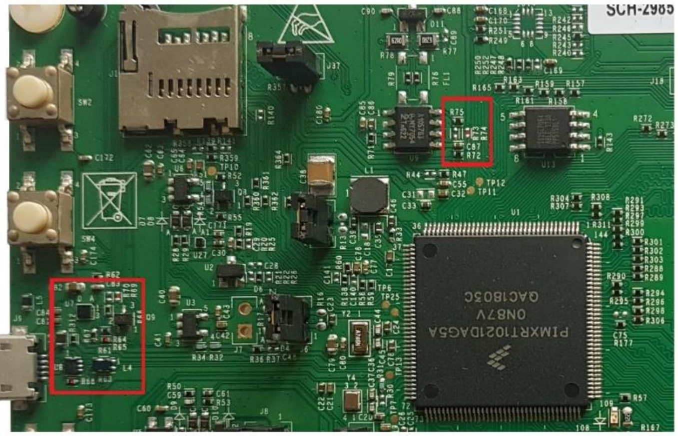

Figure 18. Removed resistor for proper operation of the ADC on the top side of the EVK board

For more information about the MIMXRT1020-EVK hardware (processor, peripherals, and so on), see the *MIMXRT1020 EVK Board Hardware User's Guide* (document MIMXRT1020EVKHUG).

4.7.1.1. Hardware assembling

1. Connect the FRDM-MC-LVPMSM power stage to the MIMXRT1020-EVK board by wires according to the pin assignment (Table 16) and the interconnection diagram (Figure 19).
2. Connect the 3-phase motor wires to the screw terminals (J7) on the Freedom PMSM power stage.
3. Plug the USB cable from the USB host to the OpenSDA micro USB connector (J23) on the EVK board.
4. Plug the 24-V DC power supply to the DC power connector on the Freedom PMSM power stage.

Table 16. MIMXRT1020-EVK pin assignment

FRDM-MC-LVPMSM	Connection	MIMXRT1020-EVK	
PWM_AT	J3, 15 <-> J17, 7	D6/AIN0/PWM/OC0A	GPIO_AD_B0_14
PWM_AB	J3, 13 <-> J19, 2	D9/OC1A/PWM	GPIO_AD_B0_15
PWM_BT	J3, 11 <-> J19, 4	D11/OC2A/PWM/SPI_MOSI	GPIO_AD_B0_12
PWM_BB	J3, 9 <-> J19, 5	D12/SPI_MISO	GPIO_AD_B0_13
PWM_CT	J3, 7 <-> J19, 6	D13/SPI_CLK	GPIO_AD_B0_10
PWM_CB	J3, 5 <-> J19, 3	D10/SPI_CS	GPIO_AD_B0_11
3V3	J3, 4 <-> J20, 4	3V3	3V3
ENC_A	J3, 3 <-> J17, 1	D0/UART_RX	GPIO_AD_B1_09
ENC_B	J3, 1 <-> J17, 2	D1/UART_TX	GPIO_AD_B1_08
GND	3, 14 <-> J20, 6	GND	GND
CUR_A	J2, 1 <-> J18, 2	A1/ADC1	GPIO_AD_B1_11
CUR_B	J2, 3 <-> J18, 3	A0/ADC0	GPIO_AD_B1_12
CUR_C	J2, 5 <-> J18, 4	A3/ADC3	GPIO_AD_B1_13
VOLT_DCB	J2, 7 <-> J18, 1	A4/ADC4/SDA	GPIO_AD_B1_10

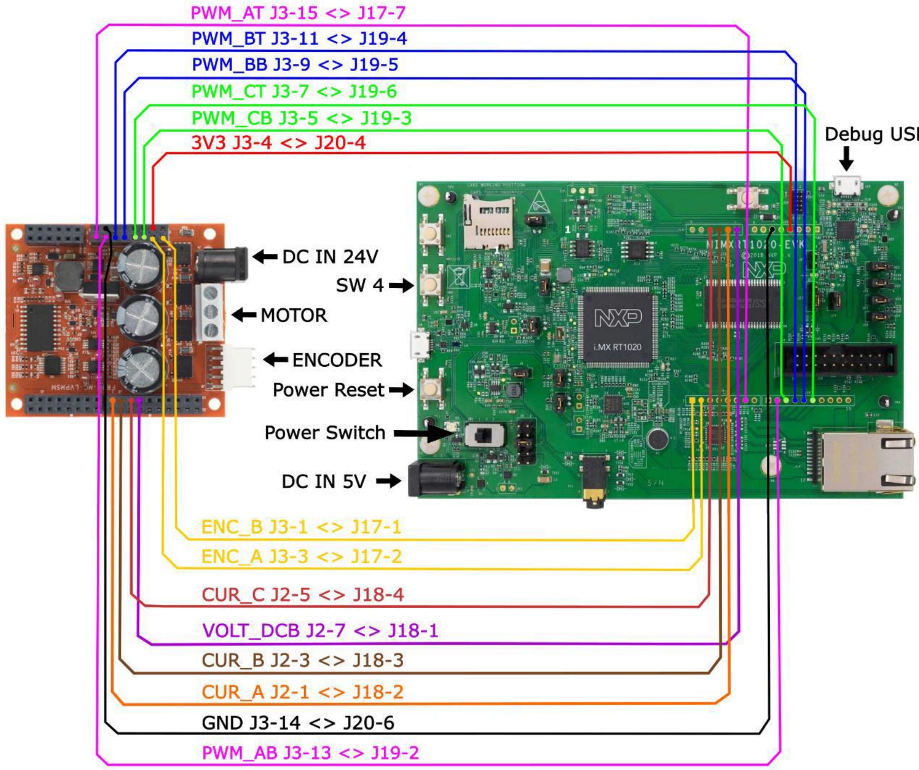

Figure 19. MIMXRT1020-EVK interconnection diagram

4.7.2. MIMXRT1050-EVK

MIMXRT1050-EVK is a 4-layer, hole-through, USB-powered PCB. It includes the i.MX RT1050 crossover processor, featuring NXP's advanced implementation of the Arm Cortex-M7 core. This core operates at speeds of up to 600 MHz to provide high CPU performance and the best real-time response.

Table 17. MIMXRT1050-EVK jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J1	5-6	J13	open	J30	1-2
J3	1-2	J15	open	J31	1-2
J4	1-2	J26	open	J32	1-2
J5	1-2	J27	1-2	J33	1-2
J7	1-2	J29	1-2	J36	1-2

NOTE

Jumper J1 in position 5-6 means that that the board is powered from the debug USB. You may also change J1 into position 1-2 for powering from the DC input (5 V).

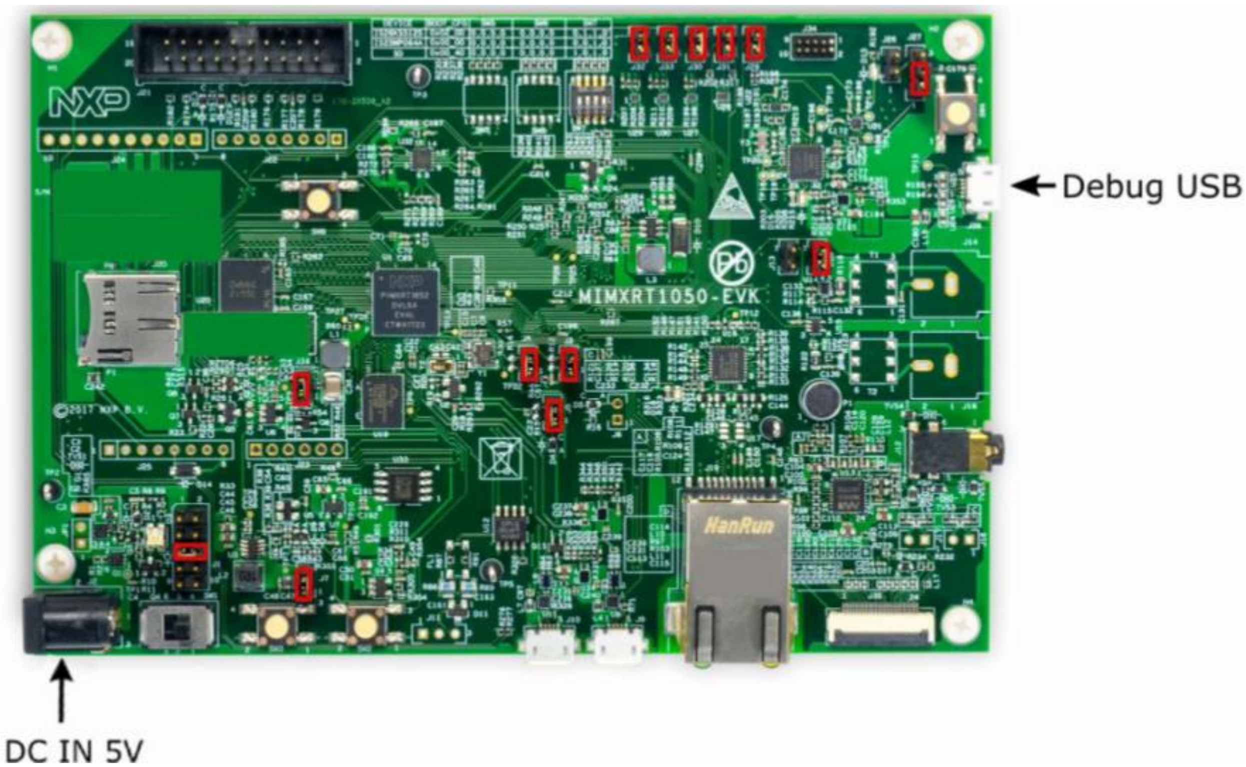

Figure 20. MIMXRT1050-EVK board with highlighted jumper settings

The motor control application requires the PWM signals to be connected from the MCU to the power stage. For a correct connection, it is necessary to solder the R278, R279, R280, and R281 resistors to the board. These resistors are located on the bottom side of the EVK board. For more details, see the schematic (revision A5).

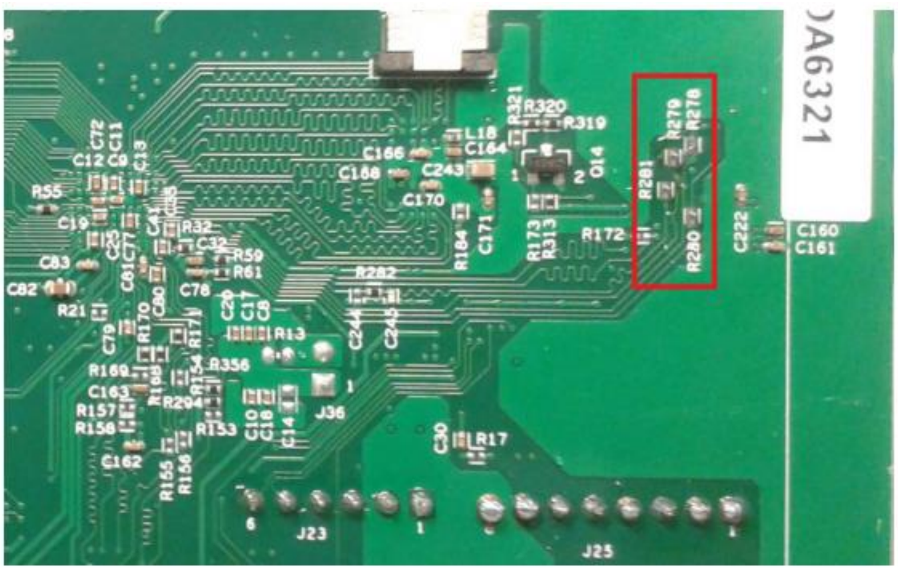

Figure 21. Resistors needed for the PWM on the bottom side of the EVK board

For more information about the MIMXRT1050-EVK hardware (processor, peripherals, and so on), see the *MIMXRT1050 EVK Board Hardware User's Guide* (document MIMXRT1050EVKHUG).

4.7.2.1. Hardware assembling

1. Connect the FRDM-MC-LVPMSM power stage to the MIMXRT1050-EVK board by wires according to the pin assignment (Table 18) and the interconnection diagram (Figure 22).
2. Connect the 3-phase motor wires to the screw terminals (J7) on the FRDM-MC-LVPMSM power stage.
3. Plug the USB cable from the USB host to the OpenSDA micro USB connector (J28) on the EVK board.
4. Plug the 24-V DC power supply to the DC power connector on the Freedom PMSM power stage.

Table 18. Pin assignment

MC-LVPMSM	Connection	MIMXRT1050-EVK	
PWM_AT	J3, 15 <-> J24, 6	D13/SPI_CLK	GPIO_SD_B0_00
PWM_AB	J3, 13 <-> J24, 3	D10/SPI_CS	GPIO_SD_B0_01
PWM_BT	J3, 11 <-> J24, 4	D11/OC2A/PWM/SPI_MOSI	GPIO_SD_B0_02
PWM_BB	J3, 9 <-> J24, 5	D12/SPI_MISO	GPIO_SD_B0_03
PWM_CT	J3, 7 <-> J22, 6	D5/TI/PWM	GPIO_AD_B0_10
PWM_CB	J3, 5 <-> J22, 3	D2/INT0	GPIO_AD_B0_11
3V3	J3, 4 <-> J24, 8	3V3	3V3
ENC_A	J3, 3 <-> J24, 10	D15/I2C_SCL	GPIO_AD_B0_00
ENC_B	J3, 1 <-> J24, 9	D14/I2C_SDA	GPIO_AD_B0_01
GND	J3, 14 <-> J25, 6	GND	GND
CUR_A	J2, 1 <-> J23, 2	A1/ADC1	GPIO_AD_B1_11
CUR_B	J2, 3 <-> J23, 6	A5/ADC5/SCL	GPIO_AD_B1_00
CUR_C	J2, 5 <-> J23, 5	A4/ADC4/SDA	GPIO_AD_B1_01
VOLT_DCB	J2, 7 <-> J22, 7	D6/AIN0/PWM/OC0A	GPIO_AD_B1_02

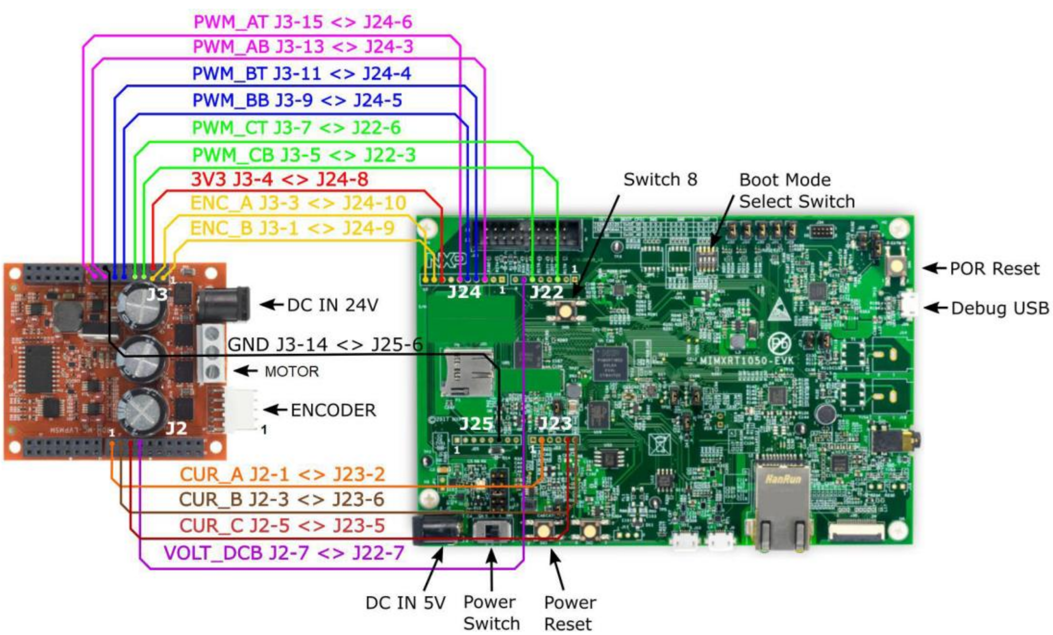

Figure 22. MIMXRT1050-EVK interconnection diagram

4.7.3. MIMXRT1060-EVK

MIMXRT1060-EVK is a 4-layer, hole-through, USB-powered PCB. It includes the i.MX RT1060 crossover processor, featuring NXP's advanced implementation of the Arm Cortex-M7 core. This core operates at speeds of up to 600 MHz to provide high CPU performance and the best real-time response.

Table 19. MIMXRT1060-EVK jumper settings

Jumper	Setting	Jumper	Setting	Jumper	Setting
J1	5-6	J13	open	J44	1-2
J3	1-2	J15	open	J47	1-2
J4	1-2	J26	open	J48	1-2
J5	1-2	J36	1-2	J49	1-2
J7	1-2	J43	1-2	J50	1-2

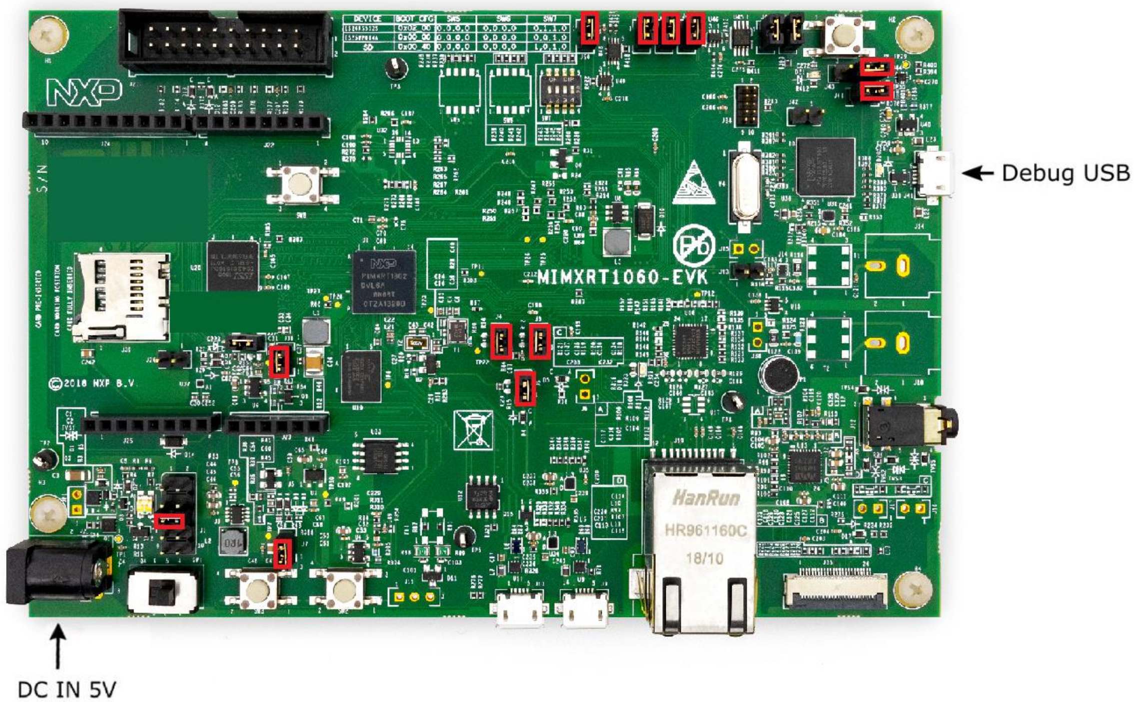

Figure 23. MIMXRT1060-EVK board with highlighted jumper settings

The motor control application requires the PWM signals to be connected from the MCU to the power stage. For correct connection, it is necessary to solder the R278, R279, R280, and R281 resistors to the board. These resistors are located on the bottom side of the EVK board. For more details, see the schematic.

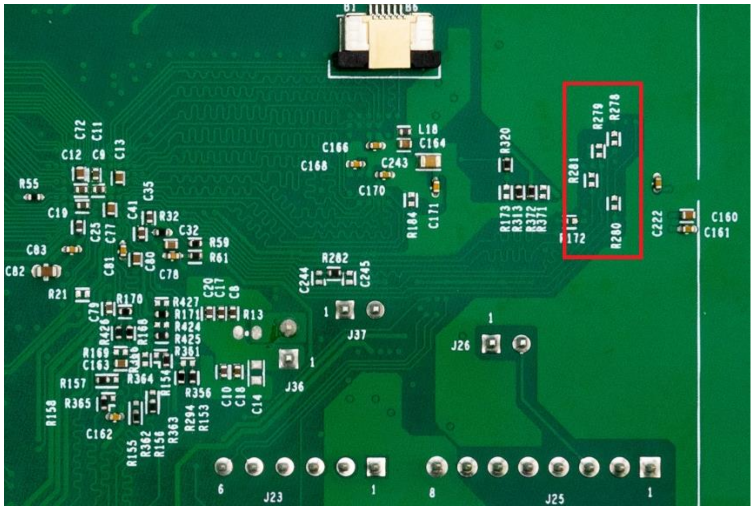

Figure 24. Resistors needed for the PWM on the bottom side of the EVK board

For more information about the MIMXRT1060-EVK hardware (processor, peripherals, and so on), see the *MIMXRT1060 EVK Board Hardware User's Guide* (document MIMXRT1060EVKHUG).

4.7.3.1. Hardware assembling

1. Connect the FRDM-MC-LVPMSM power stage to the MIMXRT1060-EVK board by wires according to the pin assignment (Table 20) and the interconnection diagram (Figure 25).
2. Connect the 3-phase motor wires to the screw terminals (J7) on the Freedom PMSM power stage.
3. Plug the USB cable from the USB host to the OpenSDA micro USB connector (J41) on the EVK board.
4. Plug the 24-V DC power supply to the DC power connector on the Freedom PMSM power stage.

Table 20. MIMXRT1060-EVK pin assignment

FRDM-MC-LVPMSM	Connection	MIMXRT1050-EVK	
PWM_AT	J3, 15 <-> J24, 6	D13/SPI_CLK	GPIO_SD_B0_00
PWM_AB	J3, 13 <-> J24, 3	D10/SPI_CS	GPIO_SD_B0_01
PWM_BT	J3, 11 <-> J24, 4	D11/OC2A/PWM/SPI_MOSI	GPIO_SD_B0_02
PWM_BB	J3, 9 <-> J24, 5	D12/SPI_MISO	GPIO_SD_B0_03
PWM_CT	J3, 7 <-> J22, 6	D5/TI/PWM	GPIO_AD_B0_10
PWM_CB	J3, 5 <-> J22, 3	D2/INT0	GPIO_AD_B0_11
3V3	J3, 4 <-> J24, 8	3V3	3V3
ENC_A	J3, 3 <-> J24, 10	D15/I2C_SCL	GPIO_AD_B0_00
ENC_B	J3, 1 <-> J24, 9	D14/I2C_SDA	GPIO_AD_B0_01
GND	J3, 14 <-> J25, 6	GND	GND
CUR_A	J2, 1 <-> J23, 2	A1/ADC1	GPIO_AD_B1_11
CUR_B	J2, 3 <-> J23, 6	A5/ADC5/SCL	GPIO_AD_B1_00
CUR_C	J2, 5 <-> J23, 5	A4/ADC4/SDA	GPIO_AD_B1_01
VOLT_DCB	J2, 7 <-> J22, 7	D6/AIN0/PWM/OC0A	GPIO_AD_B1_02

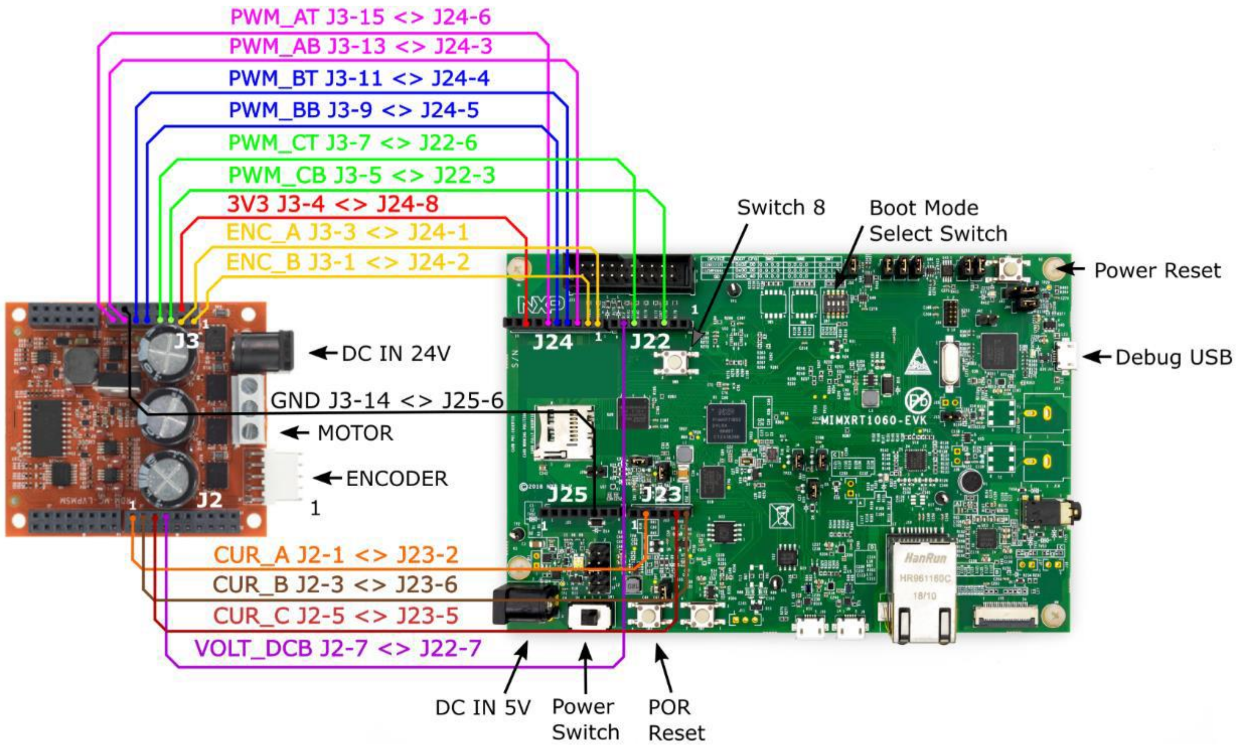

Figure 25. MIMXRT1060-EVK interconnection diagram

5. Project file structure

The demo project folder (for example; *<KSDK_install_folder>\boards\frdmkv11z\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.
- *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. These files 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 constants for the PMSM sensorless control-related algorithms).
  - *main.c*—contains the basic application initialization (enabling interrupts), the subroutines for accessing the MCU peripherals, and the interrupt service routines.
  - *mcdrv.h*—includes the specific *mcdrv_< board&MCU >.h* file into the project.
  - *mcdrv_< board&MCU >.c*—contains the motor-control driver peripherals' initialization functions specific to 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 back-EMF voltages and the board voltage and current.

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

The motor-control folder *<KSDK_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 *<KSDK_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 www.nxp.com/rtcesl for more information about RTCESL.

6. 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 SWx 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 the NXP Kinetis V Tower System and Freedom development boards. Because the HVP platform does not provide any push-buttons to control the application, the demo mode runs automatically after the HVP board is switched on.

6.1. Control button

Pressing the control button switches the demo mode on (or switches the demo mode off if it is currently switched on). Table 21 shows the correct switch button for your development board.

Table 21. Control button assignment

Board	Control button	LED state indication
TWR-KV11Z75	SW2	D5
TWR-KV31F120	SW2	D5
TWR-KV46F150M	SW2	D5
TWR-KV58F220M	SW2	D5
TWR-KE18F	SW2	D5
FRDM-KV10Z	SW2	D4
FRDM-KV11Z	SW2	D4
FRDM-KV31F	SW2	D4
FRDM-KE15Z	SW2	D4
FRDM-KE16Z	SW2	—
HVP-KV10Z	—	D20
HVP-KV31F	—	D20
HVP-KV46F	—	D20
HVP-KV58F	—	D20
HVP-KE18F	—	D20
MIMXRT1020-EVK	SW4	D18
MIMXRT1050-EVK	SW8	D18
MIMXRT1060-EVK	SW8	D18

6.2. Remote control using FreeMASTER

The remote operation is provided by FreeMASTER via the USB interface. FreeMASTER 2.0 is required for the application to operate properly. You can download FreeMASTER 2.0 at nxp.com/freemaster.

Perform these steps to control a PMSM motor using FreeMASTER:

1. Open the FreeMASTER project file (*pmsm.pmp*) in the *<KSDK_install_folder>boards\demo_apps\mc_pmsm* folder.
2. Click the communication button (the red STOP button in the top left-hand corner) to establish the communication.

Figure 26. Red STOP button placed in top left-hand corner

If the communication is established successfully, the FreeMASTER communication status in the bottom right-hand corner changes from "Not connected" to "RS232 UART Communication; COMxx; speed=115200". Otherwise, the FreeMASTER warning popup window appears.

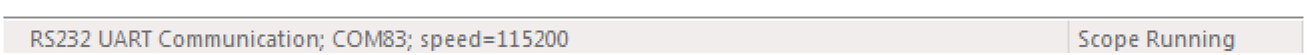

Figure 27. FreeMASTER—communication is established successfully

3. Control the PMSM motor as shown in Section 6.4, “Control page”.
4. If you rebuild and download a new code to the target, turn the application off and on.

If the communication is not established successfully, perform these steps:

1. Go to “Project->Options->Comm” tab and make sure that “SDA” is set in the “Port” option, and the communication speed is set to 115200 bps.

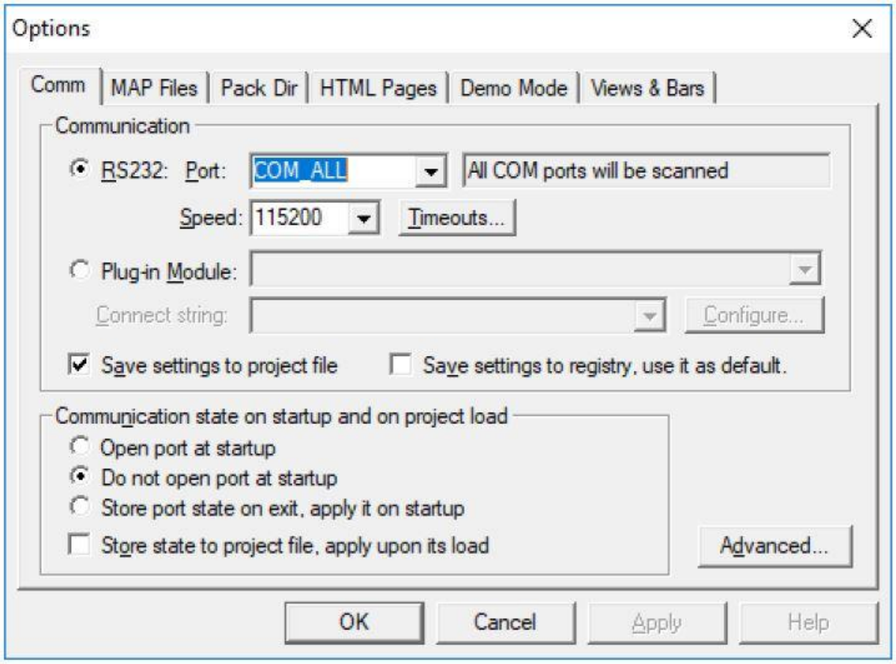

Figure 28. FreeMASTER communication setup window

2. If “OpenSDA-CDC Serial Port” is not printed out in the message box next to the “Port” dropdown menu, unplug and then plug in the USB cable, and reopen the FreeMASTER project.
3. Make sure to supply your development board from a sufficient energy source. Sometimes the PC USB port is not sufficient for supplying the development board.

6.3. FreeMASTER TSA and user variables addition to FreeMASTER watch

By default, all projects use the TSA (Target Side Addressing). This means that the information about the variables’ address and size are stored in the MCU flash memory. The TSA feature may be enabled or disabled by the BOARD_FMSTR_USE_TSA macro in the *board.h* file. Only the variables necessary for the control page functionality are stored in the TSA. Only these variables are visible in FreeMASTER. If you want to monitor your own variables, you can provide a symbol file which contains the information about the addresses of all variables in the project to FreeMASTER. The symbol files are generated during the build process to the *boards/demo_apps/mc_pmsm/<compiler>/<debug or release>* folder. The symbol files have different extensions for different compilers (IAR generates **.out*, MCUXpresso generates **.axf* and Keil generates the **.axf* file). For more information about the TSA, see *FreeMASTER Serial Communication Driver* (document FMSTERSCIDRVUG).

6.4. Control page

After launching the application and performing all necessary settings, you can control the PMSM motor using the FreeMASTER control page. The FreeMASTER control page contains:

Speed gauge—shows the actual and required speeds.

Required speed—sets up the required speed.

DC-bus voltage—shows the actual DC-bus voltage.

DC-bus motor current—shows the actual torque-producing current.

Current limitation—sets up the DC-bus current limit.

Demo mode—turns the demonstration mode on/off.

STOP button—stops the whole application.

Notification—shows the notification about the actual application state (or faults).

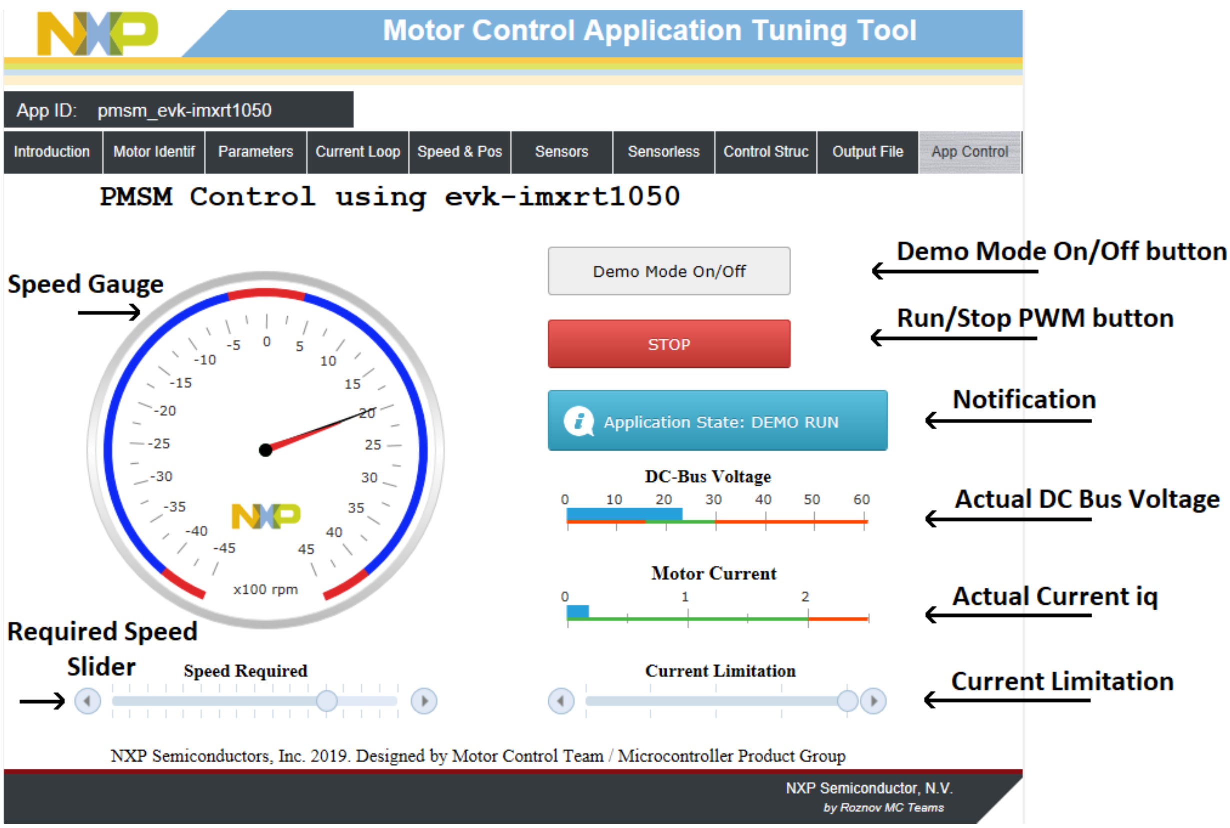

Figure 1. FreeMASTER control page

Here are the basic instructions:

To start the motor, set up the required speed using the speed slider.

In case of a fault, click on the fault notification to clear the fault.

Click the “Demo Mode” button to turn the demonstration mode on/off.

Click the “STOP” button to stop the motor.

7. Acronyms and abbreviations

Table 22. Acronyms and abbreviations

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

8. References

The following references are available at www.nxp.com:

- *Sensorless PMSM Field-Oriented Control* (document DRM148).
- *Tuning 3-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).
- *Getting Started with MCUXpresso SDK* (document MCUXSDKGSUG).
- Embedded Software Libraries User's Guides.
- *PMSM Field-Oriented Control on MIMXRT10xx EVK* (document AN12214).

9. Revision history

Table 23 summarizes the changes done to this document since the initial release.

Table 23. Revision history

Revision number	Date	Substantive changes
0	07/2016	Initial release.
1	03/2017	Updated the document with information about FRDM-KV11Z and the MCUXpresso IDE.
2	09/2017	Updated the document with information about the HVP platform.
3	04/2018	Removed the KDS IDE, added the i.MXRT board.
4	10/2018	Added FRDM-KE16Z, MIMXRT1020EVK, and MIMXRT1060EVK.
5	05/2019	Removed the building and debugging section, updated the TWR and EVK board figures.

How to Reach Us:

Home Page:
www.nxp.com

Web Support:
www.nxp.com/support

Document Number: PMSMDEMOUG
Rev. 5
05/2019

arm