

motorBench 2.50.0 User's Guide

READY

 30 Apr 2025

Table of contents:

- [Introduction](#)(see page 2)
- [Recommended reading](#)(see page 2)
- [Installation](#)(see page 2)
- [Hardware setup](#)(see page 3)
 - [MCLV-2](#)(see page 3)
 - [dsPIC33CK LVMC](#)(see page 7)
 - [MCLV-48V-300W](#)(see page 8)
 - [MCHV-2 and MCHV-3](#)(see page 9)
- [Operating instructions](#)(see page 14)
 - [Sample motorBench projects](#)(see page 14)
 - [MCC Classic](#)(see page 14)
 - [MCC Melody](#)(see page 14)
 - [Using MCC Classic](#)(see page 14)
 - [Using MCC Melody](#)(see page 15)
 - [Configuration](#)(see page 15)
 - [Board configuration](#)(see page 16)
 - [Default configuration](#)(see page 16)
 - [Supported development boards](#)(see page 16)
 - [Support for custom boards](#)(see page 17)
 - [Creating and using a custom board definition file](#)(see page 17)
 - [Making changes to custom board definition in an existing project](#)(see page 17)
 - [Comparing board definition](#)(see page 18)
 - [A note about system parameters](#)(see page 18)
- [Motor import and export](#)(see page 18)
 - [Importing a motor](#)(see page 19)
 - [Exporting a motor](#)(see page 19)
 - [Working with custom motors](#)(see page 19)
- [Motor parameter measurement](#)(see page 20)
 - [Operating instructions](#)(see page 20)
 - [Selective measurements feature](#)(see page 21)
 - [Inductance calculator widget](#)(see page 22)
 - [Common types of motor parameter measurement errors and reasons they can occur](#)(see page 22)
- [Autotuning](#)(see page 24)
 - [Managing errors in autotuning](#)(see page 25)
 - [How to identify autotuning errors](#)(see page 25)
 - [Common types of autotuning errors and reasons they can occur](#)(see page 26)
 - [How to report specific failure details to Microchip for assistance](#)(see page 26)
- [Customize](#)(see page 27)
 - [Position and velocity estimator selection](#)(see page 27)
 - [Advanced parameters](#)(see page 27)
 - [Normalized parameters](#)(see page 27)
 - [Advice](#)(see page 28)
 - [Additional information](#)(see page 28)
- [Generate code](#)(see page 28)
 - [Managing errors in code generation](#)(see page 29)

- [How to identify code generation errors](#)(see page 29)
- [Common types of code generation errors and reasons they can occur](#)(see page 30)
- [How to report specific failure details to Microchip for assistance](#)(see page 31)
- [Building code](#)(see page 31)
- [Running the Motor Control Application Framework](#)(see page 32)
 - [MCLV-2](#)(see page 32)
 - [dsPIC33CK LVMC and MCLV-48V-300W](#)(see page 32)
 - [MCHV-2 and MCHV-3](#)(see page 32)
- [Real-Time Diagnostics](#)(see page 32)
- [Troubleshooting issues with some motors](#)(see page 33)
- [Appendix](#)(see page 33)
 - [Motor parameter measurement - Calibration load specifications](#)(see page 33)

Introduction

This document describes how to set up hardware, configure, and operate motorBench® Development Suite as well as aid in troubleshooting issues with motors. This document does not advise how to use the resulting generated code. Please refer to the Motor Control Application Framework (MCAF) [User's Guide](#)¹ for additional information.

Recommended reading

This user's guide refers to motorBench operation only. Other useful documents are listed below:

- MPLAB® Code Configurator (MCC) User's Guide
- motorBench® v2.50.0 Release Notes
- [MCAF User's Guide](#)²

Installation

The following instructions outline the installation procedure for motorBench® Development Suite.

1. Verify that your internet connection is stable.
2. Open MPLAB X IDE
3. Open the Content Manager from Tools → MCC Content Manager.
4. Navigate to motorBench® Development Suite under 'Libraries' section. Select '2.50.0' from the dropdown menu and then click 'Apply'.
5. MCC will automatically attempt to download a copy of motorBench® Development Suite v2.50.0.
6. In the dropdown menu for motorBench versions, verify that v2.50.0 shows up as 'local'.

If the Content Manager fails to download and install v2.50.0 automatically, then follow the manual procedure below.

1. Download a copy of the motorBench® Development Suite v2.50.0 library file from <https://ww1.microchip.com/downloads/aemDocuments/documents/MCU16/ProductDocuments/SoftwareLibraries/Firmware/motorbench-2.50.0.mc3lib>
2. Open MPLAB X IDE
3. Go to Tools → Options → Plugins and then use the 'Install library' button to manually install v2.50.0.

¹ <https://microchiptech.github.io/mcaf-doc/latest/>

² <https://microchiptech.github.io/mcaf-doc/latest/>

Hardware setup

The following sections outline the hardware setup for [MCLV-2](#)³, [dsPIC33CK LVMC](#)⁴, [MCLV-48V-300W](#)⁵, [MCHV-2](#)⁶, and [MCHV-3](#)⁷.

MCLV-2

Hardware setup: MCLV-2

This section provides detailed steps that will help you set up your hardware to work with the motorBench® Development Suite:

1. The MCLV-2 board comes pre-installed out of the box with dsPIC33EP256MC506 Internal Op Amp Motor Control PIM (MA330031). This PIM may be replaced with any of the supported PIMs listed in the motorBench v2.50.0 Release Notes.
2. If using the dsPIC33EP256MC506 External Op Amp PIM, make sure that the PIM is populated with a silicon mask rev-A8 or later. To verify this, read out the device revision from MPLAB X and verify that the Device ID revision is equal to or greater than `0x4008`.

³ <https://www.microchip.com/en-us/development-tool/DM330021-2>

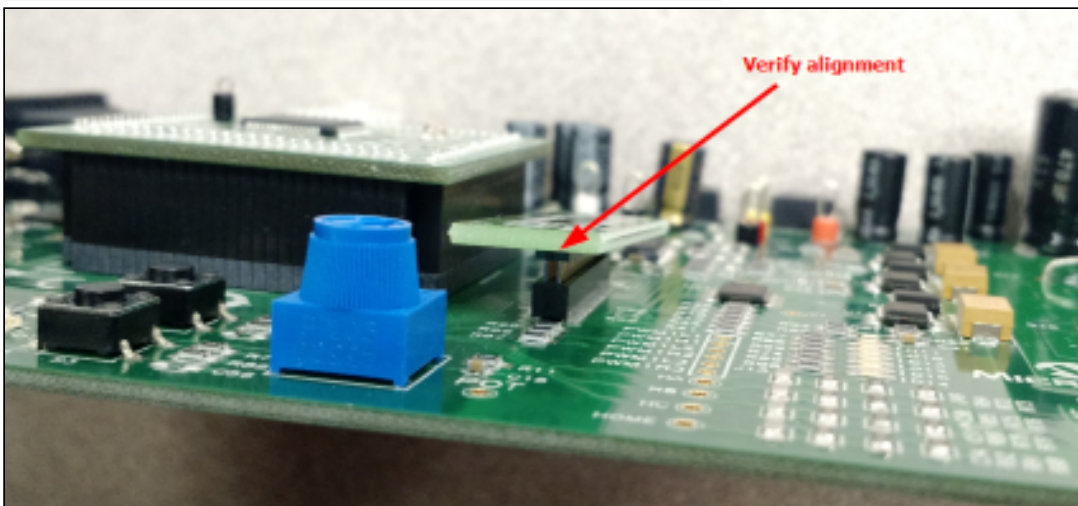
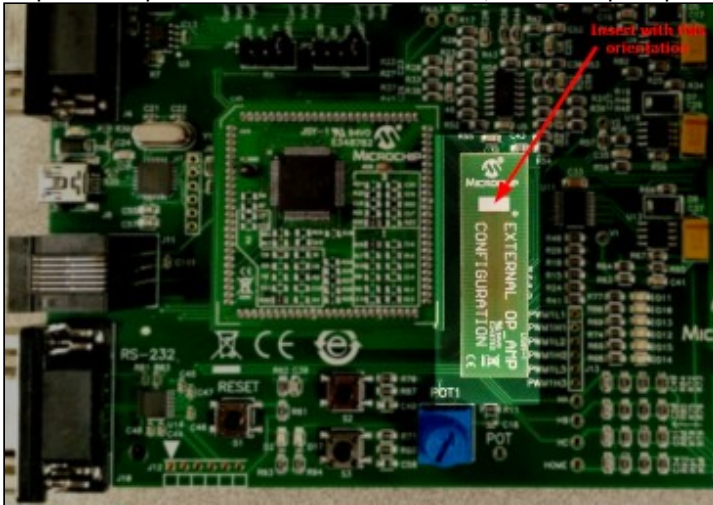
⁴ <https://www.microchip.com/en-us/development-tool/DM330031>

⁵ <https://www.microchip.com/en-us/development-tool/EV18H47A>

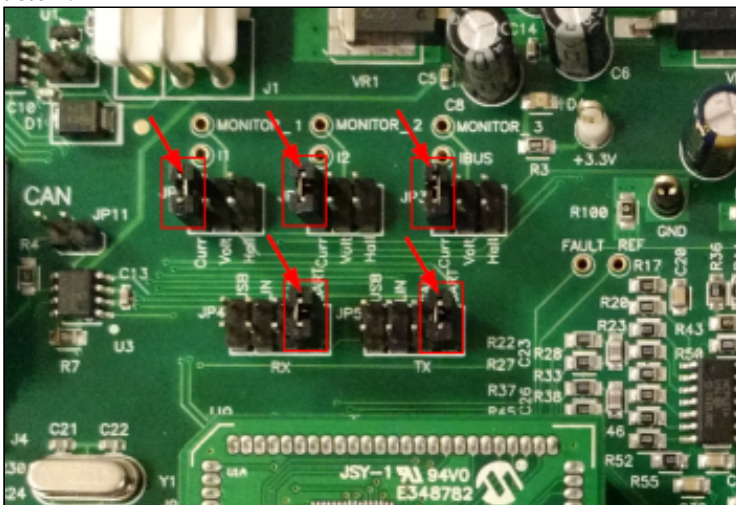
⁶ <https://www.microchip.com/en-us/development-tool/DM330023-2>

⁷ <https://www.microchip.com/en-us/development-tool/DM330023-3>

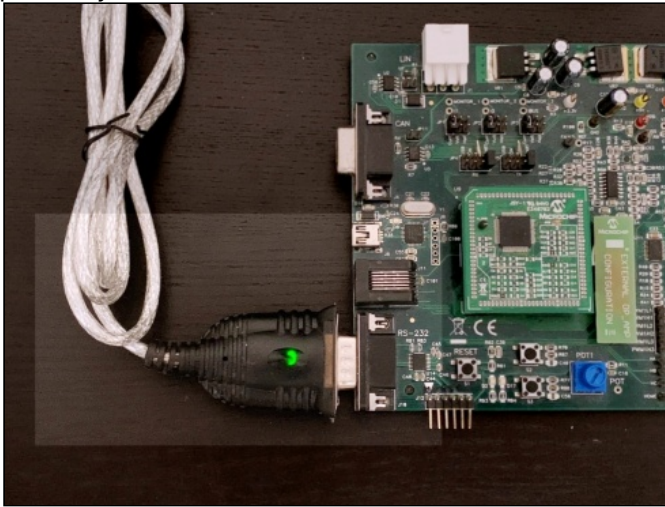
3. Ensure the Op Amp Configuration matrix board installed in **J14** matches the installed PIM. External Op Amp PIMs require the External matrix board; Internal Op Amp PIMs require the Internal matrix board.



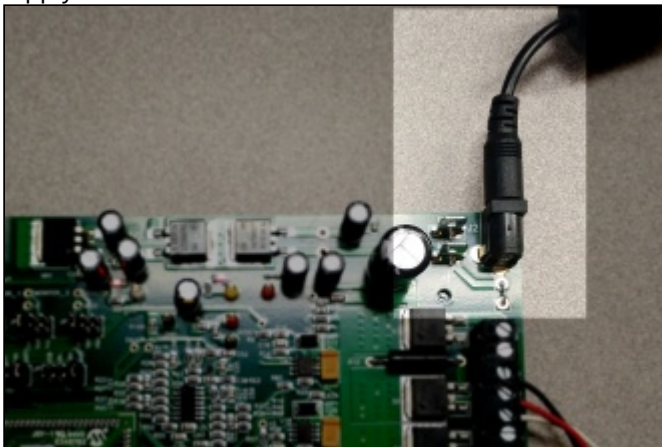
4. Update the jumpers **JP1-JP2-JP3** to **Curr** position and **JP4-JP5** to **UART** position as shown below:



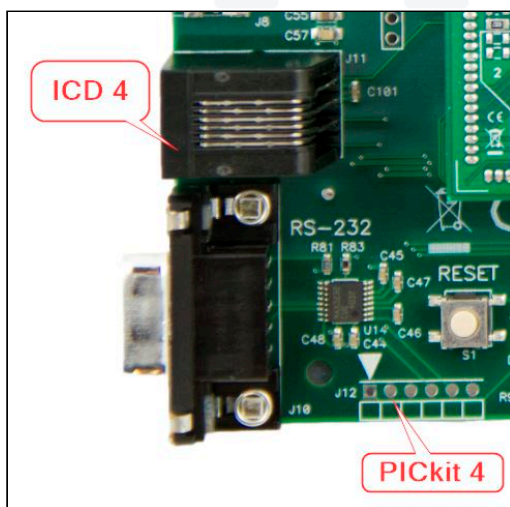
5. Connect the USB-to-logic-level-UART converter cable into **J10** of the MCLV-2 board and one of the USB ports on your PC:



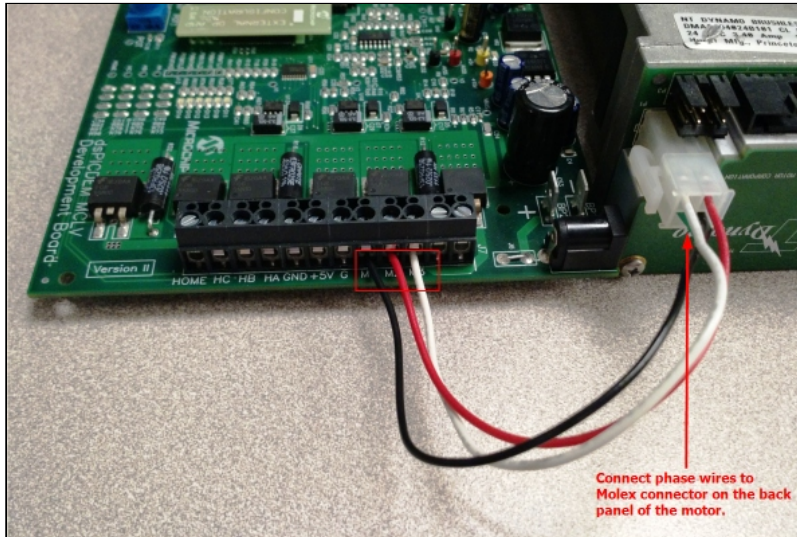
6. Plug in the 24V power supply barrel connector into **J2** of the MCLV-2 board and then plug the power supply into the mains outlet:



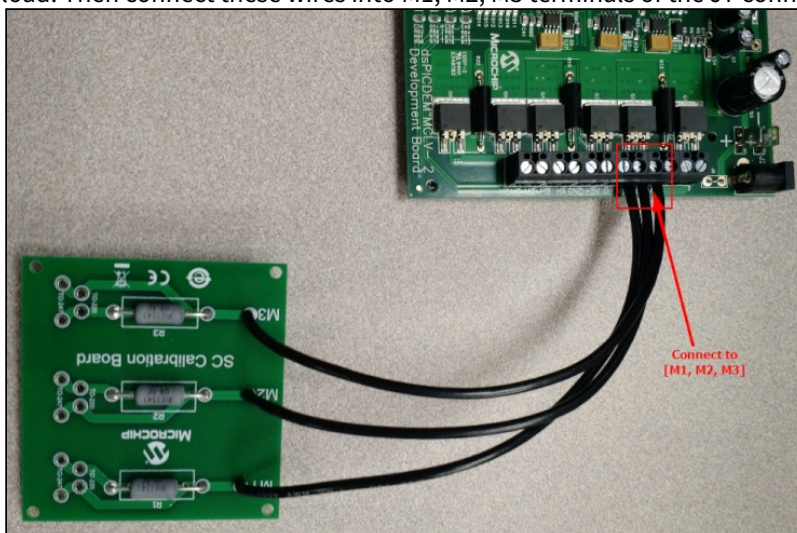
7. Connect PICKit4 or ICD4 to one of the USB ports on PC
8. Connect PICKit4 to **J12** or ICD4 to **J11** of the MCLV-2 board.



9. Connect the [black, red, white] phase wires of the 24V BLDC motor to [M1, M2, M3] terminals of J7 connector on the MCLV-2 board. You may leave the green color phase wire unconnected or connect it to the G terminal of J7 connector on the MCLV-2 board. Then, plug in the other end with a Molex connector to the 24V BLDC motor.



10. Keep the motor on a stable surface and use a clamp (if available) to secure the motor from jumping around. (If using a metal C-clamp, make sure there is a thin shim of rubber, cloth, wood, or other mechanically-compliant material between the clamp and the motor, to avoid deforming the motor housing.) Alternatively, you can also place the motor on a rubber mat. Also, do not disturb the motor or hold its shaft while motor parameter measurement is running.
11. While running the Board Calibration within motor parameter measurement feature in motorBench® Development Suite, you will need to use a calibration load of three equal-value resistors connected in place of the 24V BLDC motor. Please refer to the [Appendix section \(see page 33\)](#) of this document for more information on the calibration load. Once you have the calibration load on hand, you can start by stripping off the insulation on the wires coming from the M1, M2 and M3 terminals of the Calibration load. Then connect these wires into M1, M2, M3 terminals of the J7 connector on the MCLV-2 board.

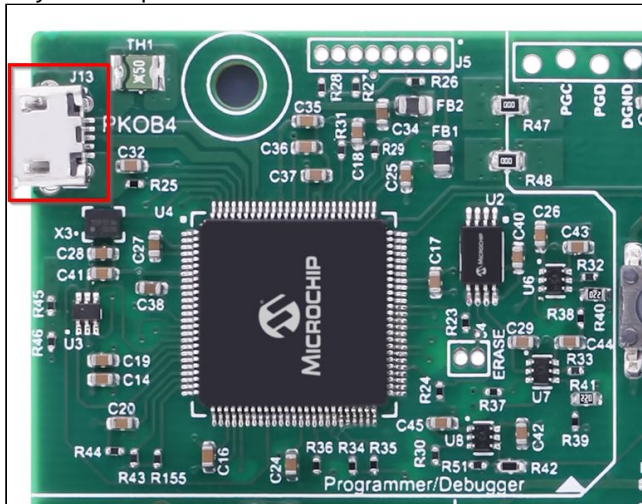


dsPIC33CK LVMC

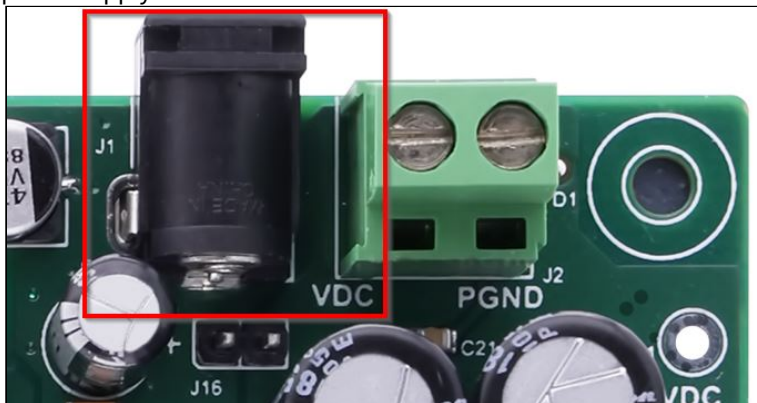
Hardware setup: dsPIC33CK LVMC

This section provides detailed steps that will help you set up your hardware to work with the motorBench® Development Suite:

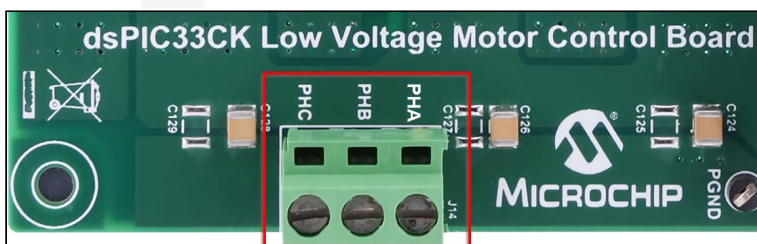
1. Connect the USB Micro-B cable between **J13** of the dsPIC33CK LVMC board and one of the USB ports on your computer:



2. Plug in the 24V power supply barrel connector into **J1** of the dsPIC33CK LVMC board and then plug the power supply into the mains outlet:



3. Connect the [black, red, white] phase wires of the 24V BLDC motor to [PHA, PHB, PHC] terminals of **J14** connector on the dsPIC33CK LVMC board.

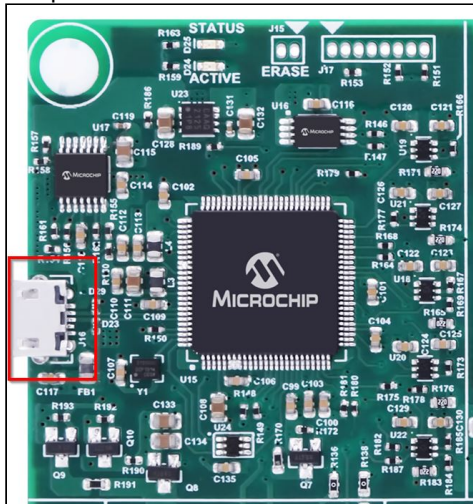


MCLV-48V-300W

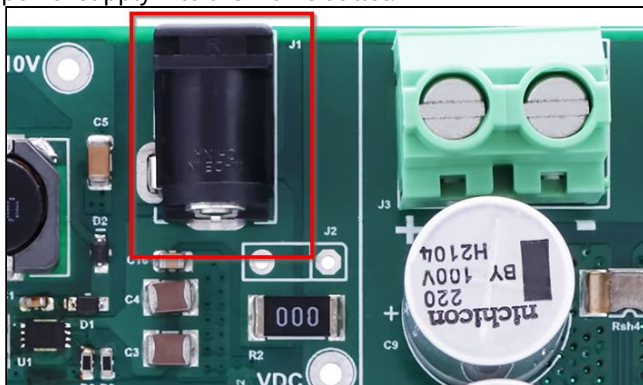
Hardware setup: MCLV-48V-300W

This section provides detailed steps that will help you set up your hardware to work with the motorBench® Development Suite:

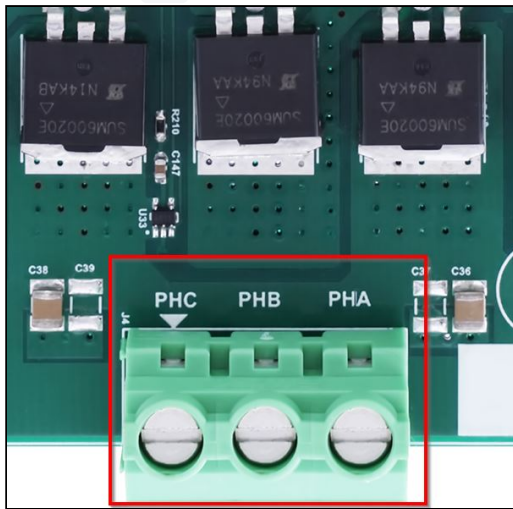
1. The MCLV-48V-300W board comes with the dsPIC33CK256MP508 Internal OpAmp Motor Control DIM (EV62P66A); install the DIM in **J8** on the MCLV-48V-300W board. This DIM may be replaced with any of the supported DIMs (Please see motorBench v2.50.0 Release Notes for additional information).
2. Connect the USB Micro-B cable to **J16** of the MCLV-48V-300W board and one of the USB ports on your computer:



3. Plug in the 24V power supply barrel connector into **J1** of the MCLV-48V-300W board and then plug the power supply into the mains outlet:



4. Connect the [black, red, white] phase wires of the 24V BLDC motor to [PHA, PHB, PHC] terminals of J4 connector on the MCLV-48V-300W board.



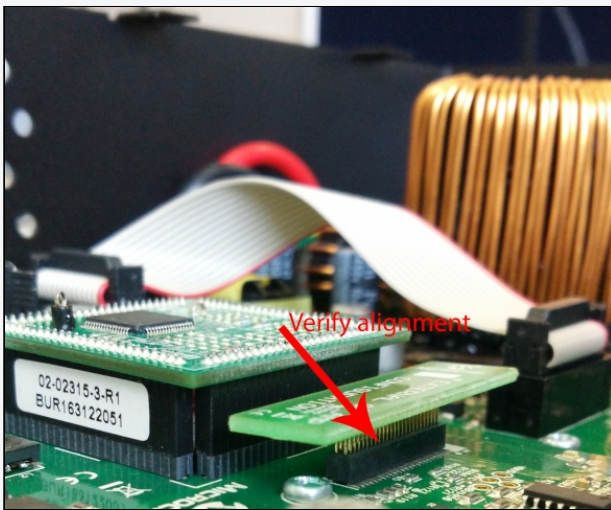
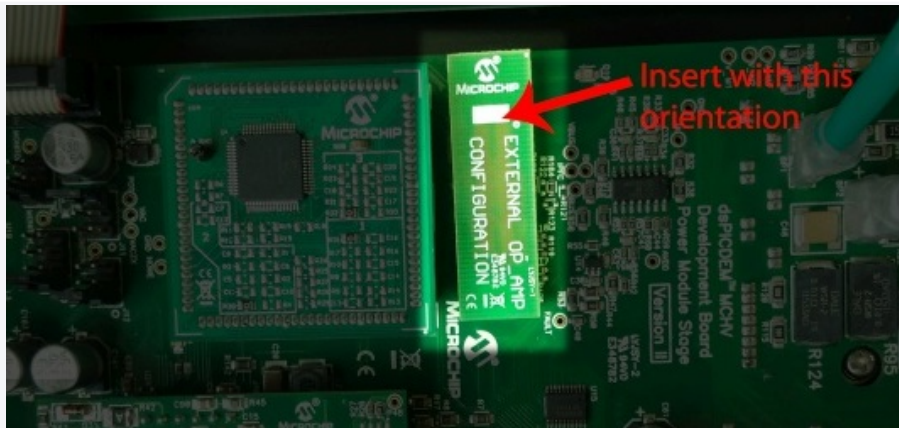
MCHV-2 and MCHV-3

Hardware setup: MCHV-2 and MCHV-3

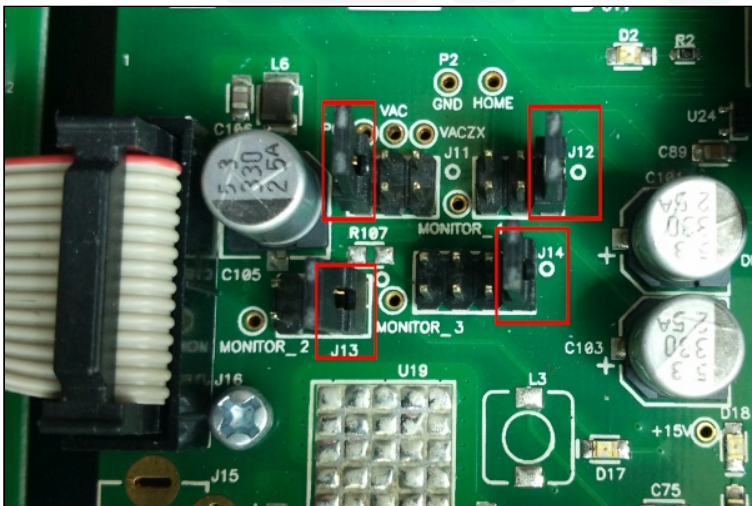
Hardware set up: MCHV-2 and MCHV-3

This section provides detailed steps that will help you set up your hardware to work with the motorBench® Development Suite: Although the setup is similar for MCHV-2 or MCHV-3, pictures of MCHV-3 are shown in the below section

1. MCHV-2 and MCHV-3 boards come pre-installed out of the box with dsPIC33EP256MC506 Internal Op Amp Motor Control PIM (MA330031); replace this PIM with any of the supported PIMs (Please see motorBench v2.50.0 Release Notes for additional information).
2. If using the dsPIC33EP256MC506 External Op Amp PIM, make sure that the PIM is populated with a silicon mask rev-A8 or later. To verify this, read out the device revision from MPLAB X and verify that the Device ID revision is equal to or greater than 0x4008 .
3. Ensure that the Op Amp Configuration matrix board installed in J14 matches the installed PIM. External Op Amp PIMs require the External matrix board; Internal Op Amp PIMs require the Internal matrix board.



4. Update the jumpers J12-J13-J14 to 1-2 position and J11 to 5-6 position as shown below:



5. Connect the USB-to-UART converter cable into RS232 labeled port on MCHV-3 board and one of the USB ports on your PC:
Please note that in the event of mechanical interference with the USB-to-UART cable, you may need to use an UART extension cable.



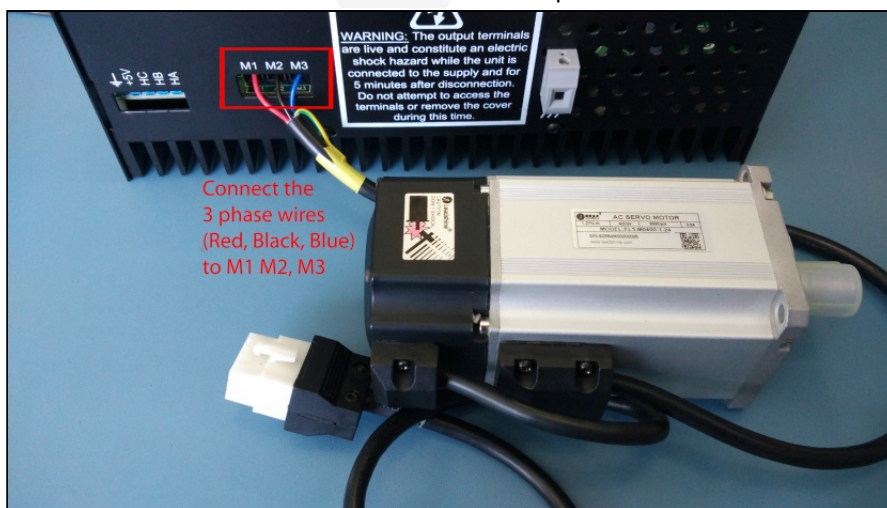
6. Plug in an appropriate ac power supply cable to MCHV-3 board and then plug in the AC power supply cable into the mains outlet:



7. Connect the provided USB cable to **PROGRAM/DEBUG** connector on MCHV-3 board and to one of the USB ports on your PC.



8. Connect the [red, black, blue] phase wires of the Leadshine 400W 220VAC Servo Motor (AC300025) to the [M1, M2, M3] terminals on the MCHV-3 board. Connect the green-yellow combination color wire to the **Ground** terminal provided on the MCHV-3 board.



9. Keep the motor on a stable surface and use a clamp (if available) to secure the motor from jumping around. (If using a metal C-clamp, make sure there is a thin shim of rubber, cloth, wood, or other mechanically-compliant material between the clamp and the motor, to avoid deforming the motor housing.) Alternatively, you can also place the motor on a rubber mat. Also, do not disturb the motor or hold its shaft while motor parameter measurement is running.
10. While running the **Board Calibration** within motor parameter measurement feature in motorBench® Development Suite, you will need to use a calibration load of three equal-value resistors connected in place of the BLDC motor. Please refer to the [Appendix section](#)(see page 33) of this document for more information on the calibration load. Once you have the calibration load on hand, you can start by stripping off the insulation on the wires coming from the M1, M2 and M3 terminals of the Calibration load. Then connect these wires into the M1, M2, M3 terminals of connector on the MCHV-3 board.

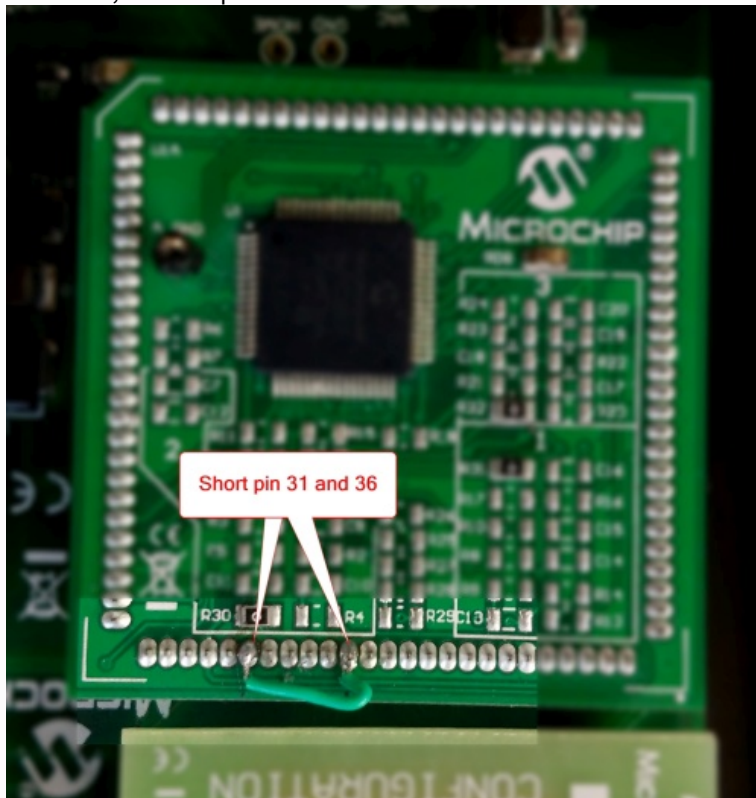
⚠ Hazard warning

Whenever connecting or disconnecting a motor or calibration load from MCHV-2/ MCHV-3 board, please disconnect the power from the MCHV-2/ MCHV-3 board. Also, use an insulated screwdriver that is recommended for high voltage rating.



- ⓘ There is a known hardware design limitation with MCHV-2 and MCHV-3 boards that can cause the device on a dsPIC33EP256MC506 External Op Amp Motor Control PIM to reset when running motors with large phase currents. This issue does not affect dsPIC33CK devices. If you are observing dsPIC device reset issues while using MCHV-2 or MCHV-3 board with this PIM and with certain test motors, **power down the board, unplug the AC power cable, wait until LED D13 is OFF** and then make the following modification:

Use a short length of jumper wire to connect digital ground signal to analog ground signal on the PIM. To do this, connect pins 31 and 36 on the dsPIC33EP256MC506 External Op Amp Motor Control PIM:



ⓘ Using an encoder

Note that if you are using an encoder, the sequence / order of the motor phase connections is important for the correct motor direction, as is the proper wiring of the encoder connections. Consult the encoder datasheet to determine the proper wiring.

Operating instructions

Device family being used	Use
dsPIC33EP	MCC Classic
dsPIC33C	MCC Melody

Sample motorBench projects

Sample projects are MPLAB X projects with preset compiler, linker and MCC configurations as required by motorBench. These projects are intended to help users to get started quickly while beginning with a new project using motorBench® Development Suite.

MCC Classic

MCC Classic is required when using motorBench® Development Suite with the supported dsPIC33EP device. In this case, starting with a sample project is strongly recommended. The sample motorBench project sets up all peripheral, system, compiler, and linker settings. Please navigate to the [release repository](#)⁸ and choose one of the projects below based on the development board that you want to use.

Device	MCLV-2	MCHV-2/3
dsPIC33EP256MC506	sample-mb-33ep256mc506-mclv2.X	sample-mb-33ep256mc506-mchv2.X

MCC Melody

motorBench® Development Suite automatically configures MCC Melody to set up the required peripherals while using one of the supported dsPIC33CK devices. Hence, a sample project is not necessary in this case.

The Motor Control Application Framework requires certain compiler and linker settings; without these settings it may not run correctly. Please refer to the motorBench Release Notes for compiler and linker settings.

Using MCC Classic

Please start with one of the sample MPLAB X projects for use with a dsPIC33EP device – the compiler, linker, and MCC System and Peripheral settings have been preset for use with the Motor Control Application Framework.

1. Unzip one of the dsPIC33EP sample motorBench projects onto your computer.
2. Open the sample motorBench project in MPLAB X.
3. Right-click the project and click **Set as the main project** in MPLAB X.
4. Right-click and open project properties. Select an in-circuit programmer (please see the motorBench v2.50.0 Release Notes for recommended in-circuit programmers) and XC16 / XC-DSC compiler version.
5. MCC can be launched from the MPLAB® X Tools menu under the Embedded selection, or by Clicking on the MCC icon in MPLAB X Toolbar.

⁸ <https://github.com/microchip-pic-avr-solutions/motorbench-release-collateral>

6. MCC help can be invoked by clicking on the



Using MCC Melody

1. **Create a New Project**
 - Go to **File > New Project...**
 - Select **Application Project(s)**, choose the desired **Device** from **dsPIC33C** family.
2. **Select Hardware Tool and Compiler**
 - Select a **Hardware Tool** that will use for programming the device
 - Select an **XC16 / XC-DSC compiler** version
3. **Select project name and location**
 - Select **Set as main project**
 - Unselect **Open MCC on Finish**
4. **Verify Project Properties:**
 - Right-click the project → **Properties**.
 - Confirm **Hardware Tool** and **compiler**.
5. **Open MPLAB Code Configurator (MCC):**
 - Click the **MCC** icon in the toolbar.
6. **Select the compatible content versions in MCC Content Manager:**
 - Click **Load Manifest**, select the `mcc-manifest-autosave.yml` file from motorBench release repository.
 - Verify the content version(s), click **Apply**.
 - Wait for the Content Manager to finish updating the content.
7. Add motorBench® Development Suite to project resources to get started.

The Motor Control Application Framework requires certain compiler and linker settings; without these settings it may not run correctly. Please refer to the motorBench Release Notes for compiler and linker settings.

Configuration

1. Select motorBench® Development Suite in Project Resources under Libraries.
 - The sample projects, if applicable, already have the motorBench library added.
 - If the motorBench library is not present in Project Resources, then add it by locating it in Device Resources under Libraries, and double-clicking on it.
2. If you haven't registered, you will be given the option to [register motorBench](https://www.microchip.com/en-us/solutions/motor-control-and-drive/motorbench-development-suite#:~:text=Registration%20Instructions)⁹, or enter a 30 day trial period. For registration steps on an offline machine please contact your local Microchip representative.
3. motorBench® Development Suite has three separate views
 - a. Composer view: This allows users to define and measure the system characteristics, tune the system characteristics and customize MCAF generation parameters. In composer view, the user can
 - i. Configure the system: Allows user to configure the motor and board. Users have the option to measure the electrical and mechanical parameters of their motor. Users can export motors and boards from their project.
 - ii. Tune the system: If desired, tuning parameters for the motor current and velocity loops can be adjusted. This is an optional step.
 - iii. Customize the MCAF parameters: Allows users to customize MCAF parameters that are used in code generation.
 - b. Pin Manager View: This allows users to see the various pins that are being used by the system.

⁹ <https://www.microchip.com/en-us/solutions/motor-control-and-drive/motorbench-development-suite#:~:text=Registration%20Instructions>

- c. Summary View: This allows users to see the various settings that are made by the user in the system.
4. Once you register, the "Home" page of the Composer view would show "Registered", otherwise it would show "Not yet Registered"
5. You can navigate between the "Configure", "Tune" and "Customize" stages using their respective buttons.
6. Once all the components fully defined and there are no configuration errors, a completed puzzle icon will appear with the status "Ready to Generate". If the puzzle icon doesn't appear complete, check if there are any messages present in the "Notifications[MCC]" tab below that prevent completion, and resolve them.

Board configuration

While motorBench has integrated support for Microchip development boards, it also supports custom boards that users can define using text files serialized in YAML format.

When starting a new project, users will have three options --

1. Select from one of the supported Microchip development boards.
2. Import one of the board definition files shared on the [motorBench release repository](#)¹⁰.
3. Import one of the existing board definition files specifically created for a given custom board.

Selected board configuration is serialized and saved within the MCC project file.

Default configuration

Upon creating a new project, motorBench® Development Suite always defaults to the dsPICDEM™ MCLV-2 Development Board, and a supported PIM if the selected device in the project is available on one of the supported PIMs.

Supported development boards

motorBench® Development Suite has integrated support for the following Microchip development boards

- dsPIC33CK Low Voltage Motor Control Board ([DM330031](#))¹¹

Boards supporting Plug-in Modules (PIMs):

- dsPICDEM™ MCLV-2 Development Board ([DM330021-2](#))¹²
- dsPICDEM™ MCHV-2 Development Board ([DM330023-2](#))¹³
- dsPICDEM™ MCHV-3 Development Board ([DM330023-3](#))¹⁴

Boards supporting Dual Inline Modules (DIMs):

- MCS MCLV-48V-300W Development Board ([EV18H47A](#))¹⁵

¹⁰ <https://github.com/microchip-pic-avr-solutions/motorbench-release-collateral>

¹¹ <https://www.microchip.com/en-us/development-tool/DM330031>

¹² <https://www.microchip.com/en-us/development-tool/DM330021-2>

¹³ <https://www.microchip.com/en-us/development-tool/DM330023-2>

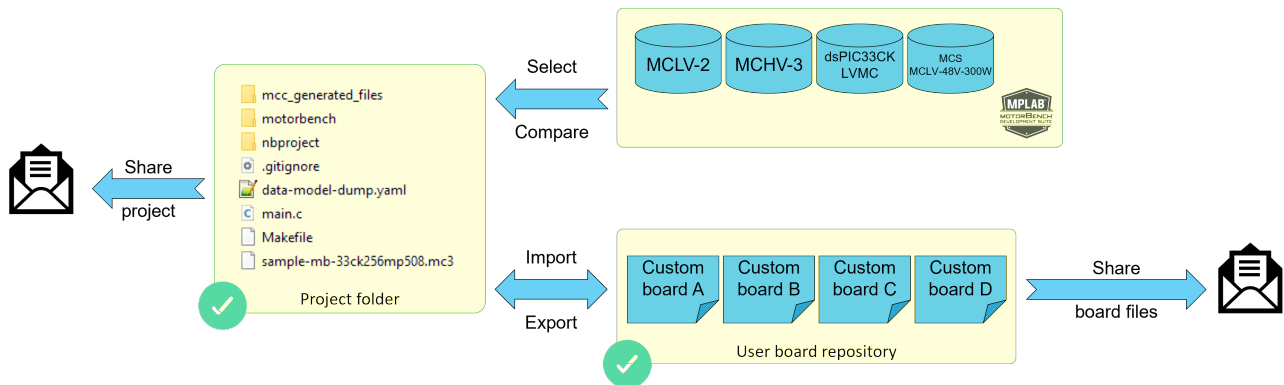
¹⁴ <https://www.microchip.com/en-us/development-tool/DM330023-3>

¹⁵ <https://www.microchip.com/en-us/development-tool/EV18H47A>

Support for custom boards

motorBench® Development Suite supports custom motor drive hardware using one of the following workflow steps

1. Export board data from a project into a board definition file
2. Import board data from a board definition file into the project
3. Compare the selected board in a project with one of the supported Microchip development boards



See [MCAF user's guide](#)¹⁶ chapter "Custom board support" for more information.

Creating and using a custom board definition file

1. Obtain a copy of the starter board file from the [motorBench release repository](#)¹⁷.
2. Open it in a text editor like VS Code
3. Make necessary changes to the starter board definition without removing any content.
4. Add new sections to completely define the custom board
5. Save the custom board definition file and close the text editor.

In Configure page of motorBench, use the 'Import Board' button to import the custom board definition file to begin using it.


i motorBench allows a maximum of one custom board to be saved into a project. Importing another custom board will overwrite the existing custom board definition in the project.

Making changes to custom board definition in an existing project

1. Open the project with MPLAB X IDE, launch MCC and open motorBench
2. In Configure page, select the custom board from the drop down menu.
3. Select 'Export board' and save the board definition data into a file on the local disk.
4. Open this board definition file in a text editor and make necessary changes.
5. Select 'Import board' and import the modified board definition file.

¹⁶ <https://microchiptech.github.io/mcaf-doc/latest/>

¹⁷ <https://github.com/microchip-pic-avr-solutions/motorbench-release-collateral>

 Board parameters displayed in the Configure page of motorBench are presented in a read-only format in this release of motorBench.

Comparing board definition

motorBench® Development Suite includes a feature to compare the currently selected board definition with one of the four supported Microchip development boards to help identify discrepancies or troubleshoot issues with board formatting.

1. Click the **Compare Boards** button to open the **File Difference Viewer**.
2. Select one of the four supported Microchip boards and a compatible PIM/DIM to compare the current board against.
3. Click the 'Compare' button to launch a diff window that highlights the differences in between the two boards under comparison.

A note about system parameters

System parameters are application-specific properties that can be viewed and edited in the Configure page of motorBench® Development Suite. They include parameters such as PWM frequency and dead time, which are application choices, rather than inherent, fixed properties of a board. For this reason they are saved as a part of the project and are not a part of the board definition — but they do need to meet the constraints of the selected board. motorBench will try to retain the values of System parameters when user imports or selects another board. However, in cases where the prevailing values of one or more System parameters cannot be supported within the limitations of the selected board, motorBench will adjust the value of these parameters to the minimum or maximum value (whichever is closer) that can be supported by the selected board.

Motor import and export

motorBench® Development Suite includes motor data for the following five reference motors.

Reference motor	For use with
hurst300	Low voltage boards example: MCLV-2, dsPIC33CK LVMC, MCLV-48V-300W
hurst075	Low voltage boards example: MCLV-2, dsPIC33CK LVMC, MCLV-48V-300W
elra-ipm80s-24v	Low voltage boards example: MCLV-2, dsPIC33CK LVMC, MCLV-48V-300W
linix-45zwn24-40	Low voltage boards example: MCLV-2, dsPIC33CK LVMC, MCLV-48V-300W

Reference motor	For use with
leadshine400	High voltage boards example: MCHV-2 and MCHV-3

Importing a motor

When a new project is created for the first time, there is no motor selected. Click on the 'Import Motor' button to select one of the motor data files included within motorBench® Development Suite, or a motor data file for a custom motor.

Select your desired motor, and click 'Open'. The motor will be imported into motorBench® Development Suite. motorBench® Development Suite is now ready to generate code.

Exporting a motor

Once an existing motor is imported, users may choose to change its parameter values.


Motor and your project

Note that once a motor is imported into motorBench® Development Suite, there is no link between the project and the file from which motor parameters were imported. 'Output - MPLAB® Code Configurator' window will show a log event with details of the filesystem path from which motor parameters were imported, but any changes made in the UI are local to the project and do not affect the original file.

All the motor parameters displayed under the 'Active Values' column are editable and changes to these parameters will be saved to the main project. Parameters listed under the 'Measured Values' column are automatically calculated by the motor parameter measurement process.


All numeric values are shown in scientific notation. When the mouse pointer is hovered over each of the individual parameters, the text box will display the value in double-precision floating point. Currently, motorBench supports entering motor parameter values in the units shown next to each parameter.


Users can choose to export the motor data if they need to use the same motor for other/new projects. Clicking the 'Export Motor' button will bring up a dialog box that will allow users to save a copy of the motor data to a file.

 Note that only the 'Active Values' in the 'Electrical and Mechanical Parameters' section are exported to the file. 'Measured Values' are saved only in the main project.

Working with custom motors

To start working with a custom motor, start by importing the configuration for one of the reference motors. Then, update all of the identification and nameplate data from on the datasheet or specification sheet of the custom motor. After this, update the electrical and mechanical parameters of the motor under 'Active values' column by

selecting  button after motor parameter measurement (Self-Commissioning) completes, or by looking up these values from the motor datasheet.

Save all changes to the project by pressing  button in the MPLAB X IDE toolbar. After this, the user may choose to export the configuration of this new motor to a file for future reference.

Editing exported files

Manual editing of the motor data file is not recommended. If needed, import the motor data file into the project, make the necessary edits and then export the motor data to the same/another file.

Motor parameter measurement

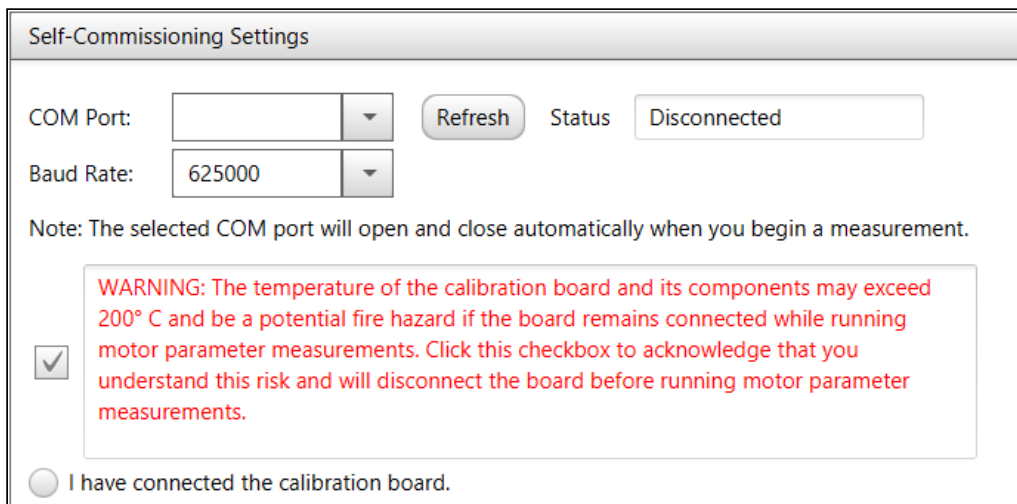
Also referred to as Self-Commissioning, this feature of motorBench can automatically measure electrical and mechanical parameters of a wide range of custom motors.

Currently, this feature is supported only on the dsPIC33EP256MC506 External Op Amp PIM with MCLV-2/MCHV-2/MCHV-3 development boards.

Operating instructions


After importing a motor, click on the "Measure Now" button in the motor's Electrical and Mechanical Parameters group to open up the motor parameter measurement window.

1. Before beginning the measurement, make sure that programming tool and a compiler are selected in the project properties.
2. Select the appropriate COM port from the combo box. Click 'Refresh' to update the list of COM ports.
3. Acknowledge the heat hazard warning. This is necessary to proceed with the measurement.



The dialog box titled "Self-Commissioning Settings" contains the following elements:

- COM Port:** A text box with a dropdown arrow.
- Baud Rate:** A text box with a dropdown arrow, currently showing "625000".
- Refresh:** A button next to the COM Port field.
- Status:** A text box showing "Disconnected".
- Note:** "The selected COM port will open and close automatically when you begin a measurement."
- Warning:** A red text box stating: "WARNING: The temperature of the calibration board and its components may exceed 200° C and be a potential fire hazard if the board remains connected while running motor parameter measurements. Click this checkbox to acknowledge that you understand this risk and will disconnect the board before running motor parameter measurements."
- Acknowledgment:** A checked checkbox next to the warning text.
- Radio Button:** An unchecked radio button labeled "I have connected the calibration board."



-  The default communication baud rate setting during motor parameter measurement is 625000. However, the RS232 transceiver devices used in MCLV-2 and MCHV-2 development boards (i.e. MAX3232CUE) are specified to operate reliably at baud rates up to 120kbps. Due to this limitation, while using MCLV-2 development board, MCHV-2 development board or MCHV-3 development board during motor parameter measurement, in case of serial communication related issues, retry measurement after reducing the communication baud rate to a lower setting.

4. Verify that the Calibration Bus Voltage is correctly set for your hardware setup. **This step is necessary even if the Calibration process is not being run.** To update this value, edit the value in [Configure → Board → Voltage Source → Output](#). For the dsPICDEM MCHV-2 and MCHV-3 Development boards, this value can be estimated from the nominal value of AC input voltage using the following equation or the table below:

$$V_{source} = V_{RMS} \times \sqrt{2}$$

Nominal AC input voltage (V_{RMS})	Value to be entered in Configure → Board → Voltage Source → Output = V_{source}
110	155.5
120	169.7
220	311.1
240	339.4

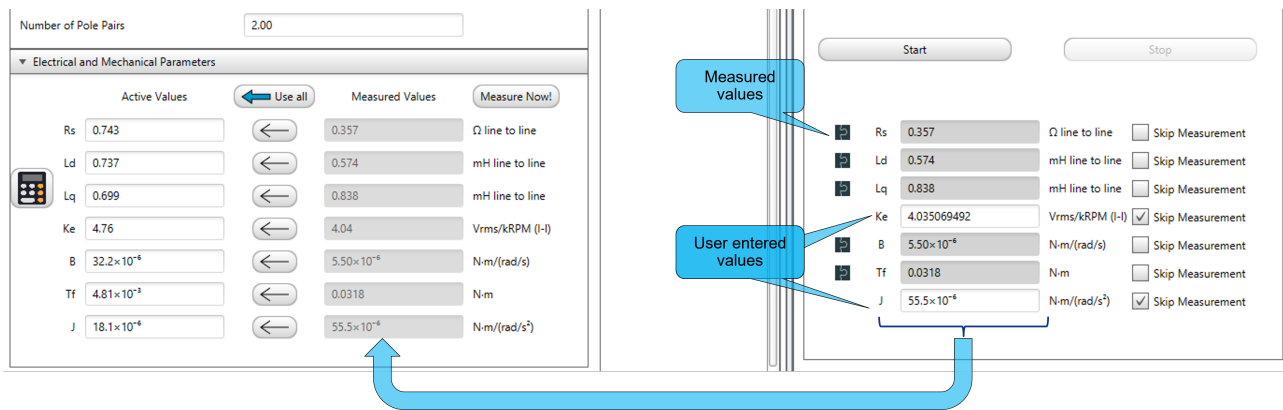
5. To calibrate, connect a [valid calibration load](#) (see [page 33](#)) (see Appendix for recommendations) and click the “I have connected the calibration board” radio button. Click 'Calibrate'.
6. To run the motor parameter measurements,
- Disconnect the calibration load and connect the motor instead.
 - Click the “I have disconnected the calibration board” and “I have connected the motor” radio buttons.
 - Click the 'Start' button to begin measurement. Measured values will sequentially appear next to each parameter as and when measurement complete.
7. The measured values are updated in the 'Measured Values' of the Motor's Electrical and Mechanical

Parameters section. Click on the  icon to set a measured value as the active value. The  button can be used to set all measured values as active values. The 'Active Values' are used for Autotuning and code generation.


Selective measurements feature

Motor parameter measurement feature of motorBench allows users to optionally skip one or more motor parameter measurements in the measurement sequence using the 'Skip measurement' option. This feature can be useful in cases where the motor parameter measurement feature fails to measure specific motor parameters, or if the measurements are not accurate enough.

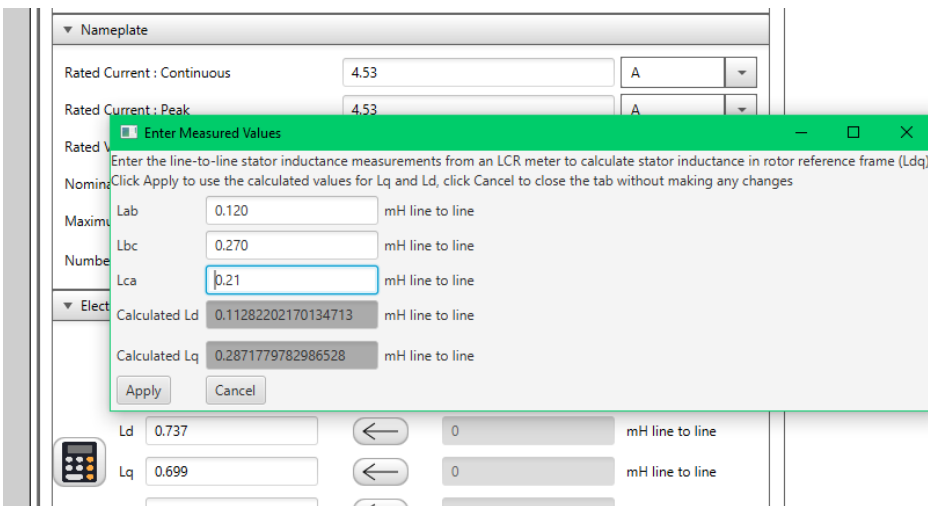
To use this feature, select the 'Skip measurement' checkbox and then manually enter a parameter value into the text box for measurements that you want motorBench to *not* measure before clicking on the 'Start' button to start the parameter measurement process. motorBench will use the user-entered parameter values for internal calculations while measuring the remaining motor parameter values. Once the measurement is complete, motorBench will reflect the measured as well as user-entered parameter value under the 'Measured values' column of the Configure page.



Inductance calculator widget

This widget helps to calculate the rotor reference frame (Ldq) inductance values from line to line measurements of three phase stator reference frame (Labc) inductance values, typically obtained using an LCR meter. Clicking on the  icon in the Configure page opens up a widget window. The widget will instantly calculate Ld and Lq as soon as values of Lab, Lbc and Lca are entered. Click on 'Apply' to insert the calculated values of Ld and Lq into the 'Active Values' column of motor parameters.

Note: inductance measurements should generally be performed with excitation signals in the 1 kHz - 20 kHz range. Before using an LCR meter, check to see what frequency it uses for measurement. Use of low-frequency (≤ 100 Hz) measurement may give incorrect readings because of motor shaft vibration and generated back-emf. Use of high-frequency measurement (≥ 100 kHz) may give incorrect readings because of parasitic capacitances.



Common types of motor parameter measurement errors and reasons they can occur

General guidance: The usual cause of errors during motor parameter measurement is due to incorrect definition of motor and/or board specifications in the "Configure" page, specifically these parameters:

1. Configure → PMSM Motor → Rated current: Continuous
2. Configure → PMSM Motor → Number of pole pairs

3. Configure → PMSM Motor → Nominal speed
4. Configure → Board → Voltage Source → Output

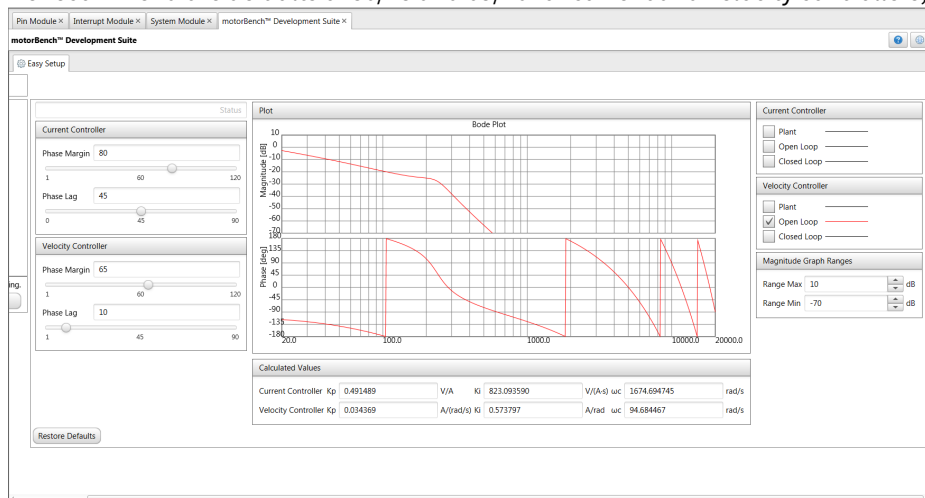
Table below provides some more specific examples of typical issues and their possible causes.

Symptom	Possible causes	Possible fix
Hardware over-current fault during R, Ld or Lq measurements (Fault Code #10)	Rated current specification of the motor is close to / higher than the full scale current range of the board. In this case, with motors that have relatively low values of electrical time constant, current ripple can trigger the over-current comparator on the board.	Retry the measurement after reducing the value in Configure → PMSM Motor → Rated current: Continuous to a lower value.
Motor stalls during inertia measurement with a Fault Code #107	The specified Nominal speed of the motor is either too low or too high for the given voltage source output at the inverter.	Correct the value in Configure → PMSM Motor → Nominal speed and retry the measurement.
Motor stalls during Ke measurement with a Fault Code #10, #102, #103, #104, #105 or #106	If the test motor has a relatively low value of stator inductance (typical Ld and Lq ~ 100µH or less), then transients in the current ripple can trigger the hardware over-current fault comparator causing the motor to stall during startup.	Increase the Measure → Motor Parameters → Current Controller → Phase Margin (slider) from its default value of 95° in 5° increments up to a maximum value of 115° and retry the measurement.
	If the test motor has a relatively high value of stator inductance (typical Ld and Lq ~ 1mH or more), then the current controller may need to be more aggressive during the startup phase in order to be able to suppress any open loop oscillations that may cause disturbances in the current control loop.	Decrease the Measure → Motor Parameters → Current Controller → Phase Margin (slider) from its default value of 95° to 91° and retry the measurement.

Symptom	Possible causes	Possible fix
	If the test motor has significant inertia (typical $J \sim 100\mu\text{Nm}/(\text{rad}/\text{s}^2)$ or more) then, Measure → Motor Parameters → Approx. Spin Down Time parameter may be set to value that is too low for the given motor.	Increase the value that is set in Measure → Motor Parameters → Approx. Spin Down Time and retry the measurement. This will allow the measurement algorithm to wait for the motor to spin down completely before attempting to restart the motor in between measurements.
Measurement fails during R measurement with a message "Unable to detect a motor on the board inverter output. Please verify that the leads of the motor under test are connected securely to the board inverter output terminals."	All three phase wires of the test motor are not correctly connected to the inverter output terminals.	Verify that all three phase wires of the motor are connected correctly to the inverter output terminals.
	Test motor has a stator inductance value that is out of supported measurement range.	-
	Voltage source that is connected to the inverter stage is not powered on.	Verify if the inverter stage is connected to a voltage source that is powered on.

Autotuning

1. Click the 'Tune' button to go to the Tuning stage. There is no required user action in this stage. You can check out the Bode plots, or use different settings for phase margin or PI phase lag at crossover, if you wish to do so.
2. We recommend the defaults of 80/45 and 65/10 for current and velocity controllers, respectively.



Managing errors in autotuning

Autotuning may fail in certain extreme cases:

- Motor parameters are extreme values, either because the motor parameter measurement process did not work correctly, or because the motor itself is unusual
- Tuning parameters (phase margin and PI phase lag at crossover) are extreme values
- Customize page parameters are extreme values

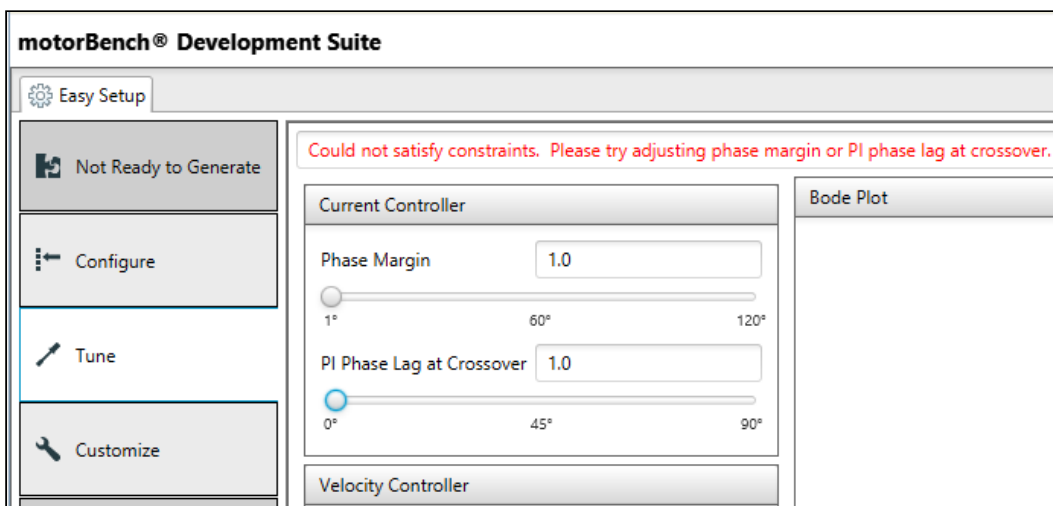
Important things to know are

- How to recognize that an error has occurred
- Common types of autotuning errors and reasons they can occur
- How to report specific failure details to Microchip for assistance

How to identify autotuning errors

If an error has occurred during autotuning, the following will occur:

- motorBench will show "Not Ready to Generate"
- the Tune page will display a short status message in red, and no Bode plot will be drawn on the graphs
- (in most cases) the Notifications [MCC] tab will show a list of problems



Notifications	Output - MPLAB® Code Configurator	Notifications [MCC] ×	Pin Manager: Grid View
Category	Source	Type: ALL	Description
	DMA	HINT	Configure EX1_INT module for DMA Channel: 3
	motorBench® Developm...	WARNING	motorBench® Development Suite configuration is not yet complete. We do not recommend generating code at this time, since it is likely to produce erroneous results.
	motorBench® Developm...	WARNING	Unable to tune the velocity or current loop: Could not satisfy constraints. Please try adjusting phase margin or PI phase lag at crossover. Refer to the motorBench Release Notes or MCAF User's Guide, available from the Microchip website, for further guidance on this issue.
	ADC1	HINT	To use OA2 output for CH123SA, select CMP2 and set it to ORAMP mode

Common types of autotuning errors and reasons they can occur

Symptom	Possible cause	Possible fix
Tune page: "No motor imported"	A motor has not been imported on the Configure page.	Import a motor on the Configure page.
Tune page: "Could not satisfy constraints. Please try adjusting phase margin or PI phase lag at crossover."	Tune page parameters (phase margin or PI phase lag) may be unusually low or high	Change tuning parameters to be closer to their default values. (The "Restore Defaults" button will set these values back to their defaults.)
	Estimator parameters on the Customize page may be unusually low or high	Change estimator parameters to be closer to their default values. (Customizable estimator parameters are AN1292 PLL time constant and bandwidth and Quadrature encoder tracking loop time constant — all of which are shown only if the "Show advanced parameters" checkbox is checked.)
	Certain other parameters (minimum operating velocity) on the Customize page may be unusually low or high	Change parameters to be closer to their default values.
Tune page: "Could not evaluate transfer function."	May occur if minimum operating velocity is zero.	Choose a minimum operating velocity greater than zero.
Other symptoms (for example, "Could not locate transfer function logic" or "Could not construct transfer function")		Please report to Microchip – these are unexpected errors.

How to report specific failure details to Microchip for assistance

If contacting Microchip staff for assistance, please copy the entire text of the Tune page status message and the description of any MCC warnings – not just a screenshot. (Click on the table cell in the MCC Notifications pane, so that it is highlighted as shown below, and press Ctrl+C to copy the text to the system clipboard.)

Notifications [MCC] × Pin Manager: Grid View	
Type: ALL	Description
WARNING	<p>not recommend generating code at this time, since it is likely to produce erroneous results.</p> <p>Unable to tune the velocity or current loop: Could not evaluate transfer function. Refer to the motorBench Release Notes or MCAF User's Guide, available from the Microchip website, for further guidance on this issue.</p> <p>Failed to findRoot.</p>
HINT	To use OA2 output for CH123SA, select CMP2 and set it to OPAMP mode

This provides important clues that may indicate the cause of the problem and how to address it.

Customize

The Customize page allows users to modify parameters used for MCAF code generation. This is an optional step to improve the behavior of the generated code for some motors; none of the parameters on the Customize page need to be modified in order to generate code. The default values found in the Customize page are good for most motors.

Position and velocity estimator selection

One estimator must be selected as the "primary" estimator used for commutation and velocity feedback. Additional estimators may be selected as secondary estimators for comparison of angle and velocity information.

Refer to the [MCAF User's Guide](#)¹⁸ for more detailed information about the different position and velocity estimators.

Advanced parameters


There are several dozen customizable parameters, but most of these are considered "advanced" and are used only in rare cases to solve specific problems. Only a handful of parameters are shown by default; to show all parameters, click on the checkbox marked "Show advanced parameters" at the top of the Customize page.

Normalized parameters

Many parameters are normalized to motor or system values, in order to ensure consistent default behavior. For example, the startup current I_{q0} is specified as a fraction of maximum current. The Customize page will display the actual value in real-world engineering units as a visual and computational aid.

▼ Motor startup		
Note: See sample graph below, which illustrates many of these startup parameters.		
Current I_{q0}	<input type="text" value="0.75"/>	$\times I_{max} = 1.7175 \text{ A}$
Nominal startup current, normalized to maximum current I_{max} , where I_{max} = minimum of motor and drive continuous ratings		
Min accel time t_{acc}	<input type="text" value="250"/>	$\times L/R = 241.51 \text{ ms}$
Determines the minimum allowable acceleration time, which affects the maximum acceleration during startup. Acceleration rates are determined using motor mechanical parameters, and can be slower, but not faster than this.		

¹⁸ <https://microchiptech.github.io/mcaf-doc/latest/>

Each of these parameters has a default value. When editing parameters, if the value is different from the default, an "undo" arrow  will be displayed. Hover over the green circular arrow to view the default value. To restore the value to its default, click on the green circular arrow.

Advice

The "Advice" section in the Customize page provides guidance on the effect of certain system quantities such as ripple current, as shown below:

Ripple current at maximum DC link voltage

$$I_R = \frac{V_{DC} T_{PWM}}{12L} = 0.1318 \times I_{max}$$

$I_R < 0.2I_{max}$	Low ripple current (< 1.3% additional I^2R loss)
$0.2I_{max} \leq I_R < 0.4I_{max}$	Moderate ripple current (< 5.3% additional I^2R loss)
$I_R \geq 0.4I_{max}$	High ripple current ($\geq 5.3\%$ additional I^2R loss)

I_R describes the worst-case peak amplitude of ripple current, which occurs when the three motor phases are switching at some permutation of (0%, 50%, 100%). Ripple current can approach this value at high modulation indices. The RMS value of ripple

This information is provided to highlight specific quantities that may cause problems in some systems. These are general guidelines and may not apply to all motor control systems.

Additional information

Refer to the [MCAF User's Guide](#)¹⁹ for more detailed guidance on parameter customization.

Generate code

- To generate code, click "Generate" under the Project Resources section. Look for the generation progress under the MPLAB® Code Configurator console window. Once generation is complete, you will see a banner comment like so -

Notifications	Output	Notifications [MCC]	Pin Manager: Grid View
Project Loading Warning			
MPLAB® Code Configurator			
16:34:59.647	INFO:	*****	
16:34:59.648	INFO:	Generation complete (total time: 27250 milliseconds)	
16:34:59.648	INFO:	*****	
16:34:59.648	INFO:	Generation complete.	

You will also be able to see the generated files added to your project.

- If you want to run generation again, and have not made any manual edits, consider removing previously-generated files under the mcc_generated_files/motorBench directory. This will avoid having to merge newly-generated files with earlier-generated files.
- You are now ready to build and run the project, and spin the motor.

¹⁹ <https://microchiptech.github.io/mcaf-doc/latest/>

Managing errors in code generation

Code generation may fail in certain extreme cases:

- Motor parameters are extreme values, either because the motor parameter measurement process did not work correctly, or because the motor itself is unusual
- Tuning parameters (phase margin and PI phase lag at crossover) are extreme values
- Customize page parameters are extreme values

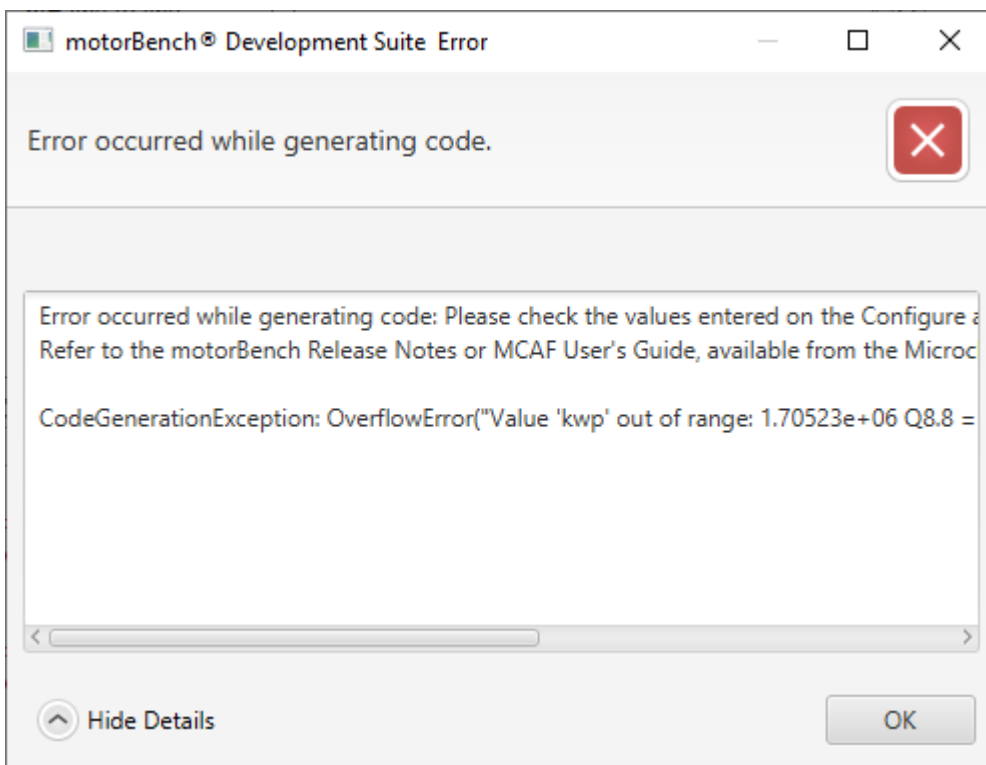
Important things to know are

- How to recognize that an error has occurred
- How to report specific failure details to Microchip for assistance
- Common types of code generation errors and reasons they can occur

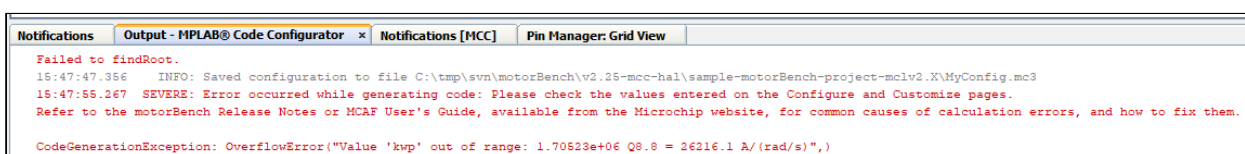
How to identify code generation errors

If an error has occurred during code generation, the following will occur:

- A dialog box will pop up with an error message:



- MPLAB® Code Configurator console window will have a SEVERE category message like shown below -



Common types of code generation errors and reasons they can occur

General guidance: The usual cause of code generation errors is a motor parameter that is extremely high or extremely low. Please see the section "Motor Control Limitations" which discusses ranges of motor parameters that work well with motorBench® Development Suite. There are a few reasons a motor parameter could cause a code generation error:

- The parameter has not been measured or entered correctly. (For example, a datasheet states 50μH but is entered as 50mH = 0.05H.)
- The motor has unusual motor parameters. (Slotless motors may fall into this category; they have very low inductance.)
- The motor is not well-matched to the motor drive circuitry, for example use of a 6V motor with a 24V motor drive.

The Customize page parameters may be at an extremely high or low value either.

Looking at the error message may help to diagnose and check for a fix.

Symptom: category	Symptom: specific message	Possible cause	Possible fix
ZeroDivisionError		One of the motor parameters is zero. (This should never be the case.)	Make sure none of the motor parameters are zero.
ZeroDivisionError		One of the Customize page parameters is zero	Check whether there are Customize page parameters that are zero, but should be greater than zero.
CodeGenerationException: OverflowError	kwp out of range	This is the velocity loop proportional gain. Out-of-range errors can occur if the inertia (J) is very high.	Make sure the inertia value is reasonable. An increase or decrease in velocity loop phase margin in the Autotuning page may be required for high-inertia motors. (Increasing phase margin generally lowers kwp whereas decreasing phase margin raises it.)
CodeGenerationException: OverflowError	velocitySlowrateLimitDecel out of range	May occur if Coulomb friction torque (Tfr) is out of range.	Make sure the Coulomb friction value is reasonable.

Symptom: category	Symptom: specific message	Possible cause	Possible fix
CodeGenerationException: OverflowError	normLsdT out of range	May occur if inductance (Ld and/or Lq) is out of range.	Make sure the motor inductance values are reasonable and well-matched to the motor drive.
CodeGenerationException: OverflowError	normRs out of range	May occur if resistance (Rs) is out of range.	Make sure the motor resistance values are reasonable and well-matched to the motor drive.
CodeGenerationException: OverflowError	(Other messages)	A parameter on the Customize page is at an extreme value	Recheck values on the Customize page.
Other errors			Please report to Microchip – these are unexpected errors.


How to report specific failure details to Microchip for assistance

If contacting Microchip staff for assistance, please copy the entire text of the dialog box or MCC Output window — not just a screenshot.

This provides important clues that may indicate the cause of the problem and how to address it.

Building code

At this point, work in motorBench® Development Suite is complete. In MPLAB X, click the Clean button and then the Run button to build and program the device.

 **Reminder:** the Motor Control Application Framework requires certain compiler and linker settings; without these settings it may not run correctly. Please refer to the motorBench Release Notes for compiler and linker settings.

Running the Motor Control Application Framework

Directions for using MCAF with the supported boards are slightly different:

MCLV-2

- Press button S2 to start/stop the motor.
- Press button S3 to reverse the direction of rotation.
- Turn the potentiometer to control speed.

dsPIC33CK LVMC and MCLV-48V-300W

- Press button SW1 to start/stop the motor.
- Press button SW2 to reverse the direction of rotation.
- Turn the potentiometer to control speed.

MCHV-2 and MCHV-3

- Press the button labeled PUSHBUTTON to start/stop the motor.
- Hold down the button PUSHBUTTON (minimum 3 seconds) to reverse the direction of rotation.
- Turn the potentiometer to control speed.

In the event of an error, both LEDs will flash together to indicate an error code; see the 'Error Codes' documentation in the [MCAF User's Guide](#)²⁰ for further information. The component number for the LEDs used for error indication on each board are listed below.

Board	LED 1	LED 2
MCLV-2	D2	D17
dsPIC33CK LVMC	LD10	LD11
MCLV-48V-300W	LD2	LD3
MCHV-2 and MCHV-3	D17	D19

Real-Time Diagnostics

The Motor Control Application Framework includes out-of-the-box support for [X2Cscope](#)²¹, a third-party plugin for MPLAB X which facilitates real-time diagnostics. X2Cscope is available in the same **Available Plugins** tab used to install motorBench® Development Suite.

²⁰ <https://microchiptech.github.io/mcaf-doc/latest/>

²¹ <https://x2cscope.github.io/>

Troubleshooting issues with some motors

We have observed some issues during our testing under the following circumstances:

1. Starting up small motors with large inertia loads
2. Starting up certain motors with low speed reference
3. Instability in current loop

These are some solutions that may work for some motors:

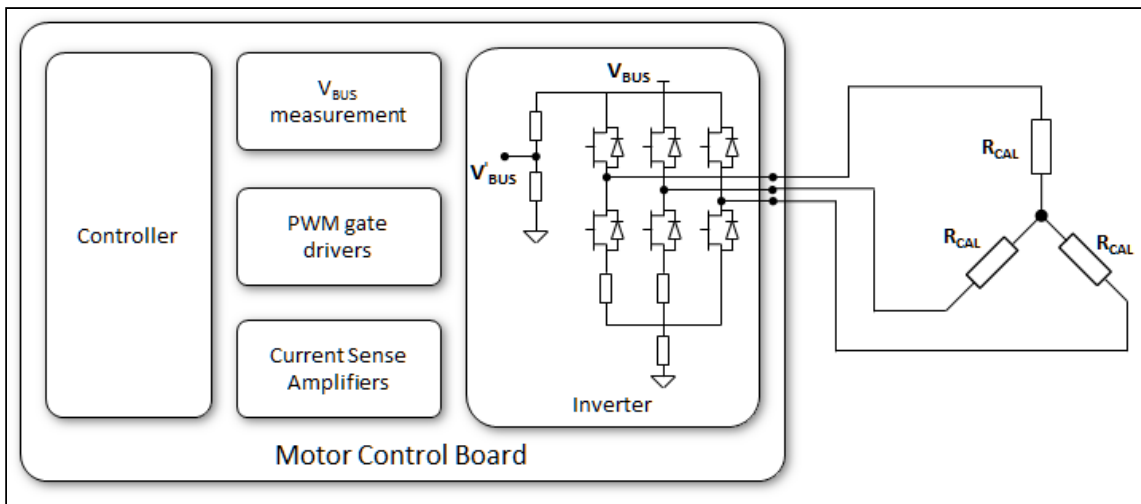
1. Starting up small motors with large inertia loads using the classic startup method may have a difficult time with the transition from forced commutation to closed-loop commutation. Use the Weathervane startup method in these circumstances.
2. Starting up certain motors with low speed reference fails to start. This may be due to either an issue in the startup method or the behavior of the sensorless estimator during the transition to closed-loop commutation. Workarounds for this issue include: use the Weathervane startup method, ensure that the speed command is not at the low end of its range, or reduce the DC link voltage if possible.
3. If instability in current loop is observed an increase of 5° or 10° in current loop phase margin may help.

Appendix

Motor parameter measurement - Calibration load specifications

The board calibration process analyses the hardware to measure any deviation from its design values of board parameters. This will help improve the accuracy of motor parameter measurements. The board calibration step of motor parameter measurement is optional. However, it is recommended that at the least, it is run once every time there is a change in the development board.

The board calibration process requires three equal-value resistors to be connected to the three-phase inverter output as shown in the diagram below.



The following equation provides the recommended range of calibration resistance values based on parameters of the board that is being calibrated:

$$\frac{V_{BUS}}{0.20 \times I_{fullscale} \times 2} \leq R_{CAL} \leq \frac{V_{BUS}}{0.80 \times I_{fullscale} \times 2}$$

where $I_{fullscale}$ and V_{BUS} are defined as follows:

- $I_{fullscale}$ — full-scale current for the board
- V_{BUS} — voltage applied to the inverter's three-phase bridge.

During the calibration process, the specialized firmware applies a series of voltage pulses on the calibration resistors and measures the resulting current. The pulse train has the following properties:

- Amplitude of V_{BUS}
- Pulse train duration of 2 ms with each pulse lasting 50 μ s
- Duty cycle of 46%

Since the pulses applied have an amplitude of V_{BUS} , the calibration resistors must be voltage rated to handle at least half of V_{BUS} . The pulse train also imposes resistor size requirements due to the energy in the pulses that need to be dissipated with a nominal temperature rise to prevent resistance changes. Since the voltages for low and high voltage applications vary vastly, the resistor values vary as seen below. Factoring all the above requirements we have the following values of resistors for the boards supported:

dsPICDEM™ MCLV-2 Development Board - 10 Ω and 21 millijoules

dsPICDEM™ MCHV-2 Development Board and dsPICDEM™ MCHV-3 Development Board - 50 Ω and 450 millijoules

In order to ensure accurate calibration, use resistors with

- 1% tolerance specification or better
- Low temperature coefficient