


KMS and KV31

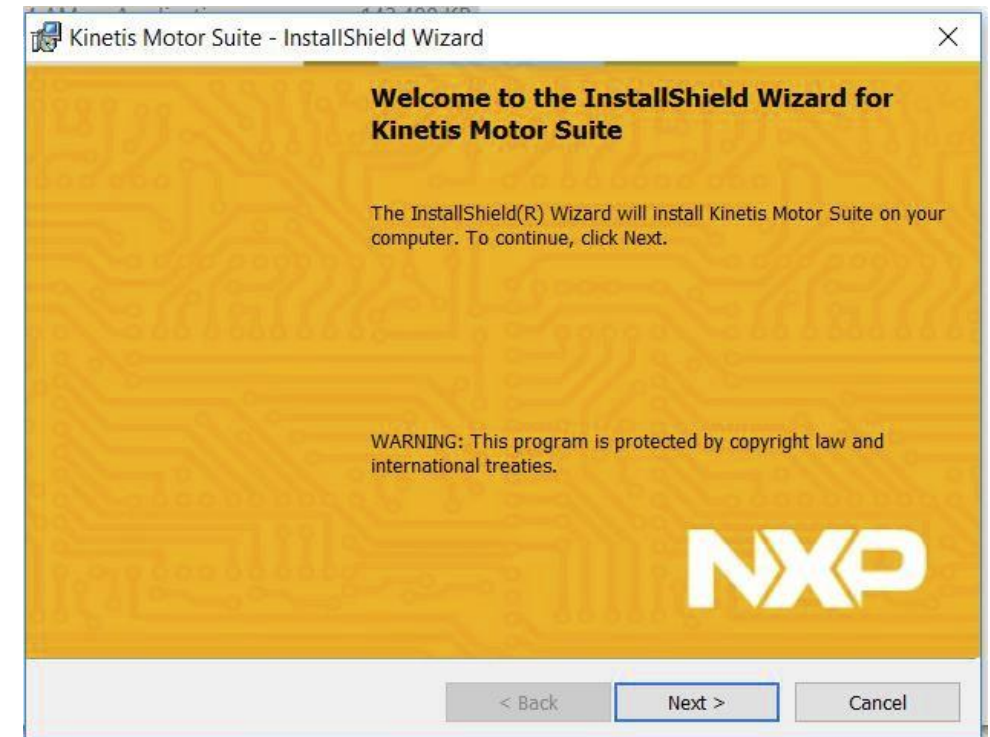
Part I: Setup the Environment

1. Software

1.1. Kinetis Motor Suite PC software

- Kinetis Motor Suite PC software refers to the PC executable that runs the KMS user interface, communicates with the selected microcontroller (MCU), and adapts embedded firmware reference projects for your system.
- Locate the “Kinetis-Motor-Suite-1.1.0-Installer” at your course website, download it and follow the steps to install it.

 Kinetis-Motor-Suite-1.1.0-Installer



1.2. Kinetis Software Development Kit (SDK)

- KMS builds on the Kinetis software development kit (KSDK) to configure hardware and peripherals for the supported microcontrollers.
- If you did not yet install “Kinetis SDK 1.3.0” on your computer then locate it on your course website, download it and install it.



Kinetis SDK 1.3.0 Mainline - Windows

1.3. Development tools

- In order to access the open-source embedded firmware provided as part of KMS and to compile KMS firmware, your PC must have an integrated development environment (IDE) such as Kinetis Design Studio (KDS).
- It is a free IDE based on the Eclipse software framework and distributed by NXP.
- If you did not yet install the “Kinetis-design-studio_3.2.0” on your computer then locate it on your course website, download it to and install it.



1.1.4. Communication

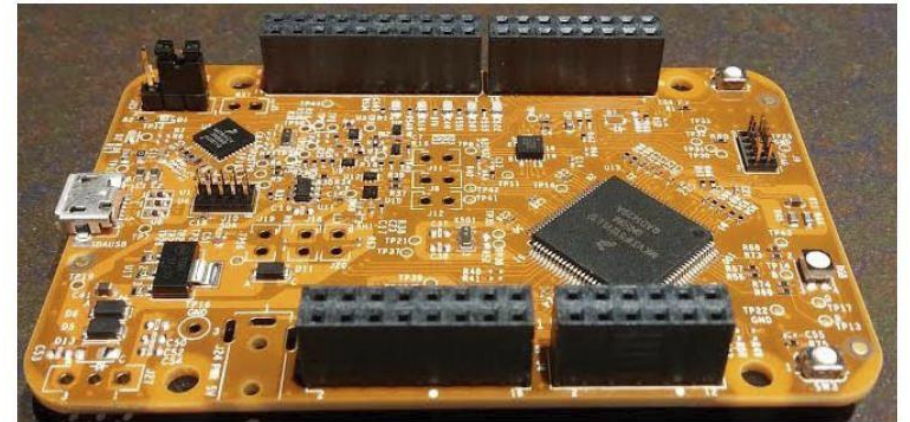
- Kinetis development platforms are already preloaded with P&E Microcomputer Systems (P&E Micro) openSDA software which communicate serially via USB with your PC using P&E Micro Windows Hardware Interface Drivers.

2. Locate your hardware:

- FRDM-MC-LVPMSM: Low-Voltage, 3 Phase Motor Control Module
- FRDM-KV31F: Kinetis KV3x FRDM Development Module
- Power supply with 24V, 5A output and 5.5 x 2.1mm barrel connector



Power Stage (FRDM-MC-LVPMSM)



Control Board (FRDM-KV31F)



Figure 1: FRDM-KV31F Callouts



Figure 2: FRDM-KV31F Pin-Out

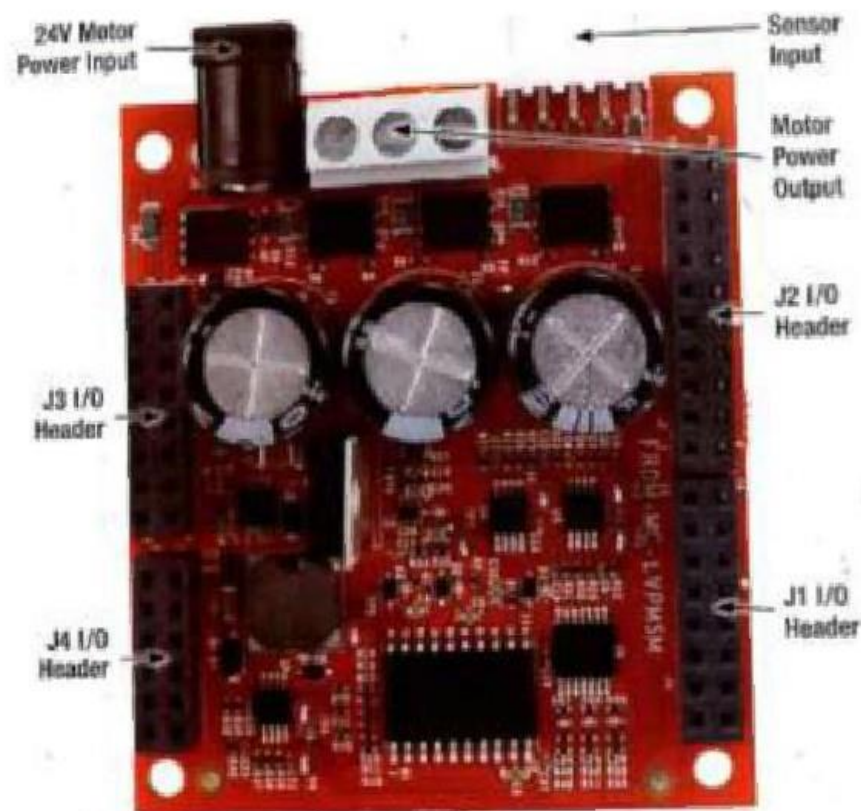


Figure 1: FRDM-MC-LVPMSM Callouts

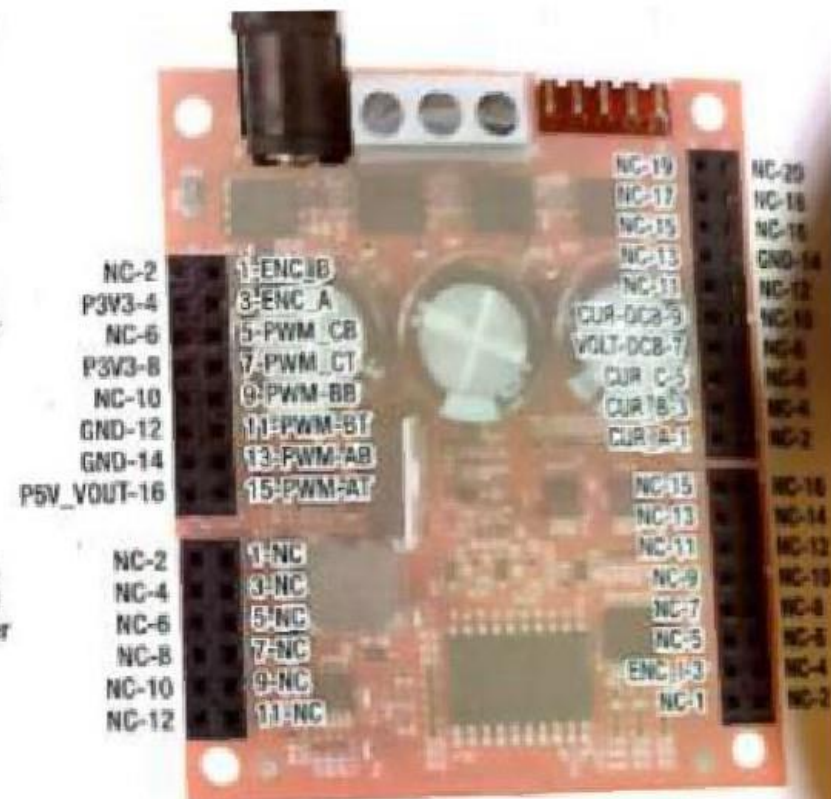


Figure 2: FRDM-MC-LVPMSM Pin-Out

Linux 45ZWN24-40 motor

The application uses the Linux 45ZWN24-40 motor described in the following table. The motor can be used with the Freedom development platforms.

Linux 45ZWN24-40 motor parameters

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



Linux motor 45ZWN24-40

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

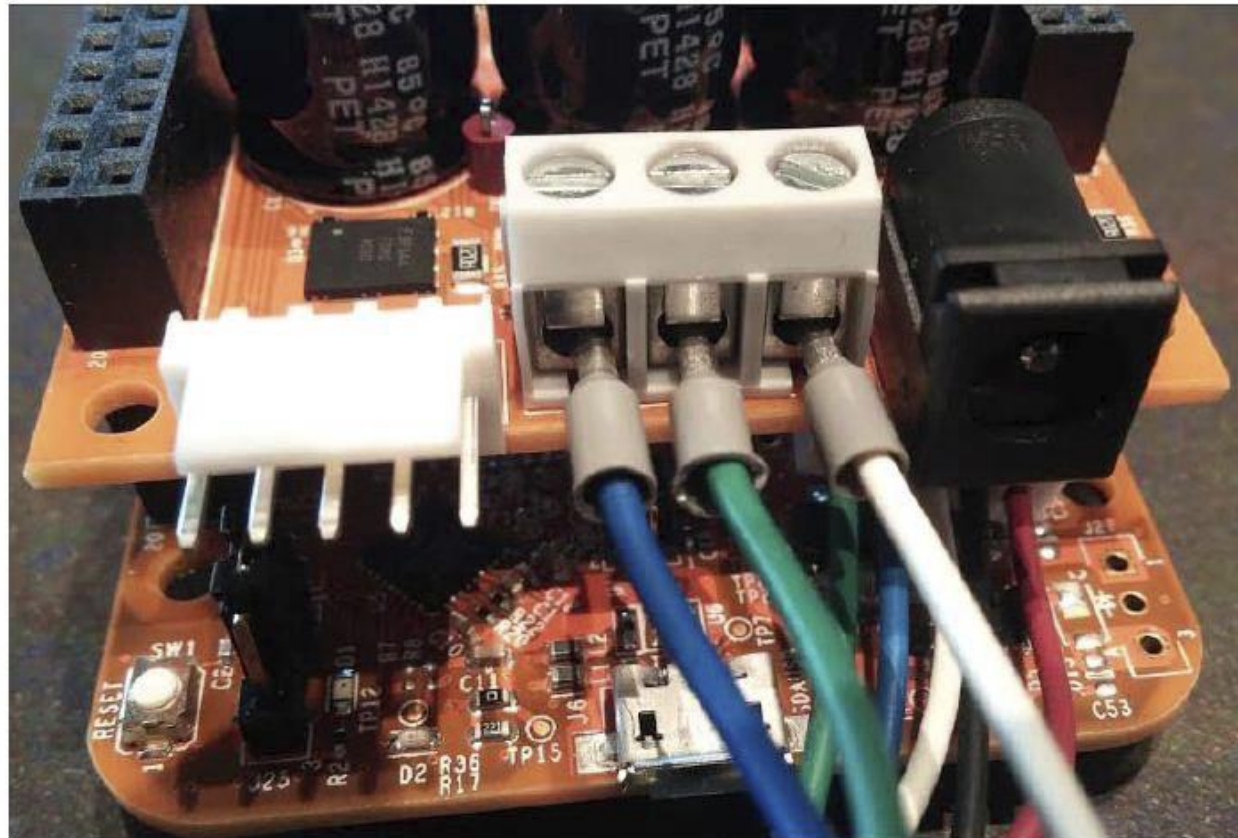
3. Connect your hardware.

- Connect the Power Stage and the Control Board by inserting the pins from the bottom of the Power Stage into the top of connectors on the Control Board. The default jumper configuration is appropriate.



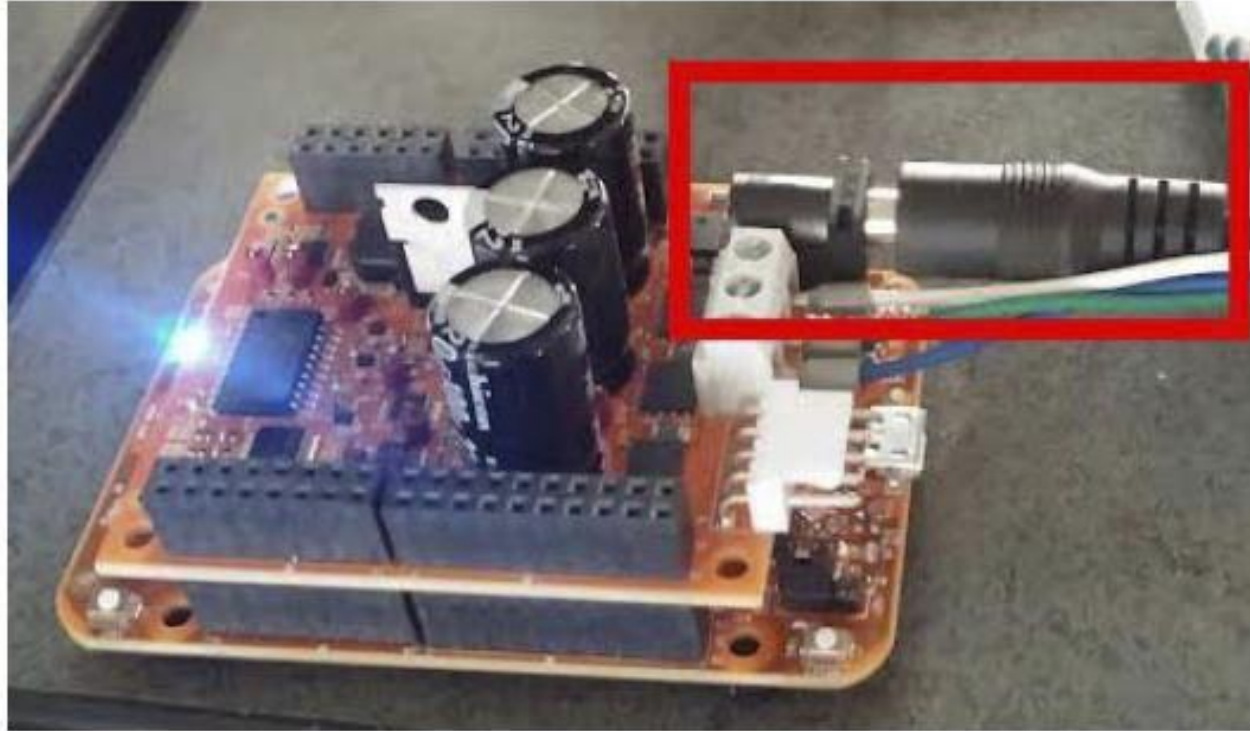
Plug Power Stage into Control Board

- Connect the power supply to the FRDM-MC-LVPMSM.



Connect leads to Power Stage

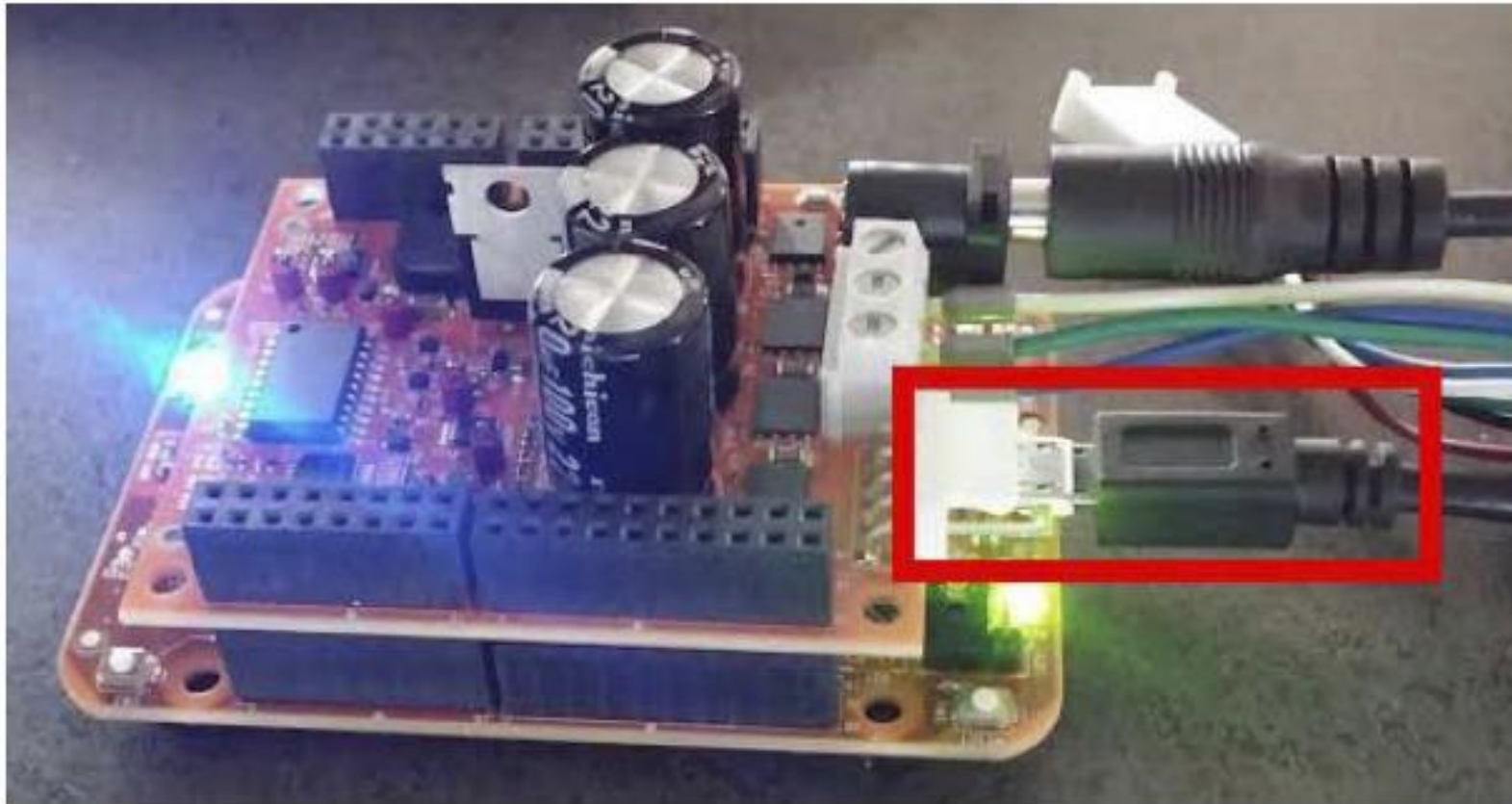
- Plug in Power Supply and connect it to the Power Stage.



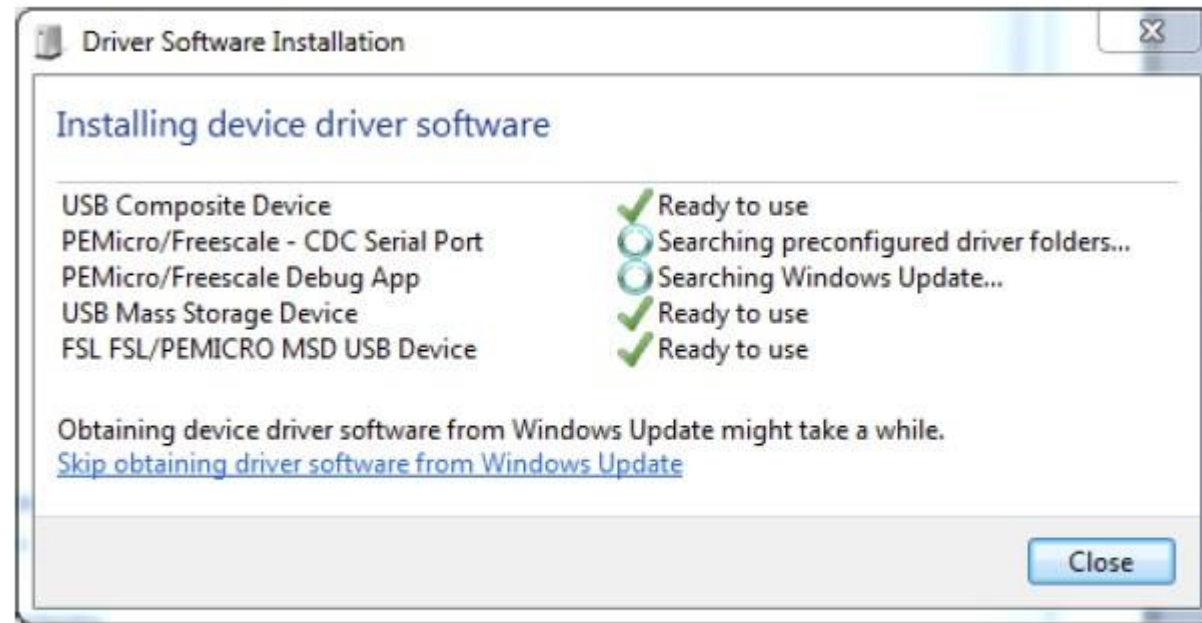
Connect power supply to power stage

Note: Be sure that you are using the correct 24 V power supply. Please check with your instructor.

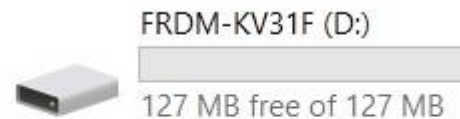
- Connect the Control Board to the computer using the USB cable.



Connect MCU to computer

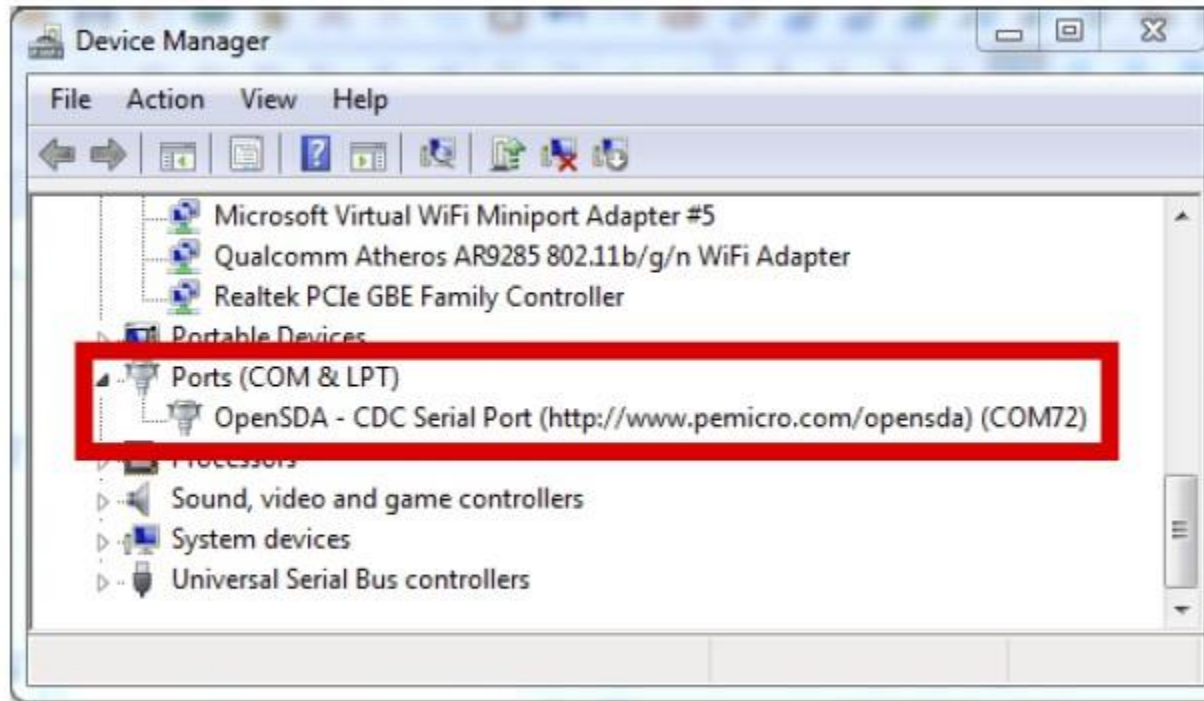


Installation window for P&E Micro driver



Check that FRDM-KV31F appears as a drive.

- Confirm successful connection by verifying that a COM port has been assigned to a P&E Micro OpenSDA Serial Port



Communication port

Part II: Run your project on MCU.

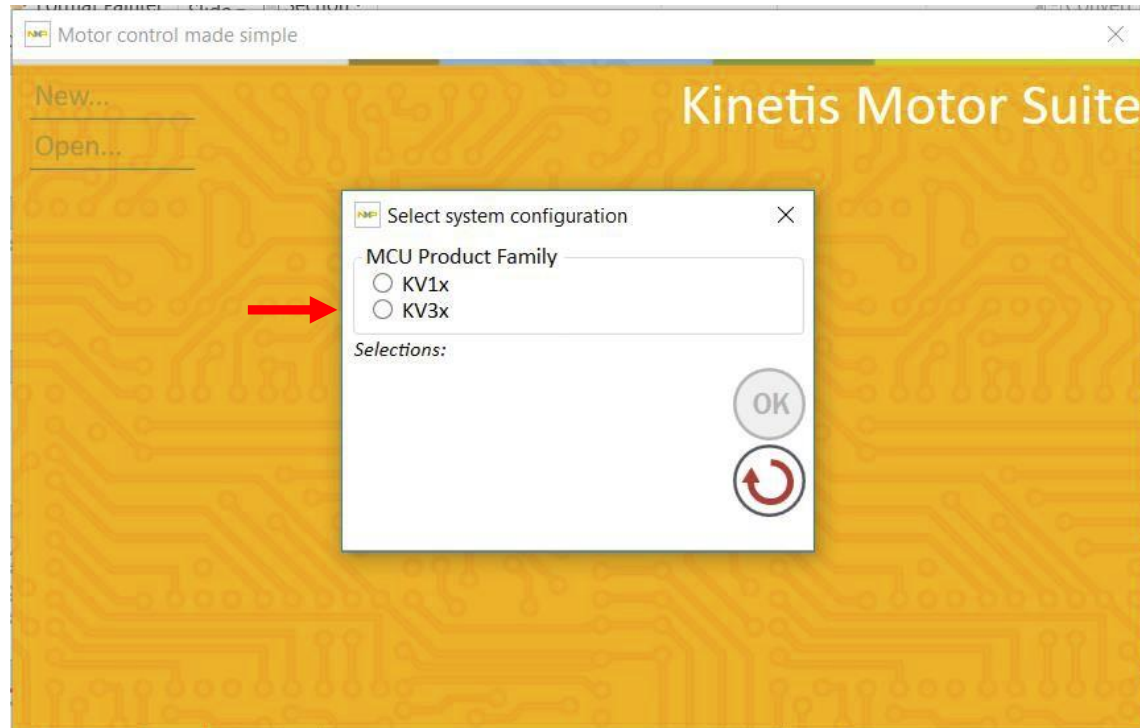
1. In your Kinetis Motor Suite, Click Project -> Run Project on MCU. This will Download the software required to run the Kinetis Motor Suite to the board
2. Wait for the download to finish. After download, the LED on the board should start glowing in various colors rather than a blinking LED.
3. KMS relies on the Kinetis Software Development Kit (KSDK) version 1.3.0.

Part III: Run your Motor using Motor Tuner.

Velocity lab: Spin your motor using Motor Tuner.

- Just getting a motor to spin is typically challenging. However, Kinetis Motor Suite makes it easy.
- We are using KMS Motor Tuner to characterize a motor and then operate it across the speed range - with minimal user configuration and without coding.
- Launch KMS by double-clicking the Desktop icon.

- Select New* and specify your desired system configuration



Select KV31F.

Select system configuration

Development Platform

☒ Freedom

☐ High Voltage

☐ Tower

Selections:

- KV3x

OK

↺

Select system configuration

Control Type

☐ Sensored Position

☒ Sensorless Velocity

☐ Sensored Velocity

Selections:

- KV3x

- Freedom

OK

↺

Select system configuration

Motor Type

☒ PMSM

Selections:

- KV3x

- Freedom

- Sensorless Velocity

OK

↺

Select system configuration

Development Environment

☐ IAR Embedded Workbench

☒ Kinetis Design Studio

Selections:

- KV3x

- Freedom

- Sensorless Velocity

- PMSM

OK

↺

Select system configuration

Click OK to confirm your configuration and place KMS reference project at path below.

Selections:

- KV3x

- Freedom

- Sensorless Velocity

- PMSM

- Kinetis Design Studio

- 1.1.0.312

OK

↺

Project Name:

FRDMKV31F_SNLESSVEL_KDS_1_1_0_312

Project Path:

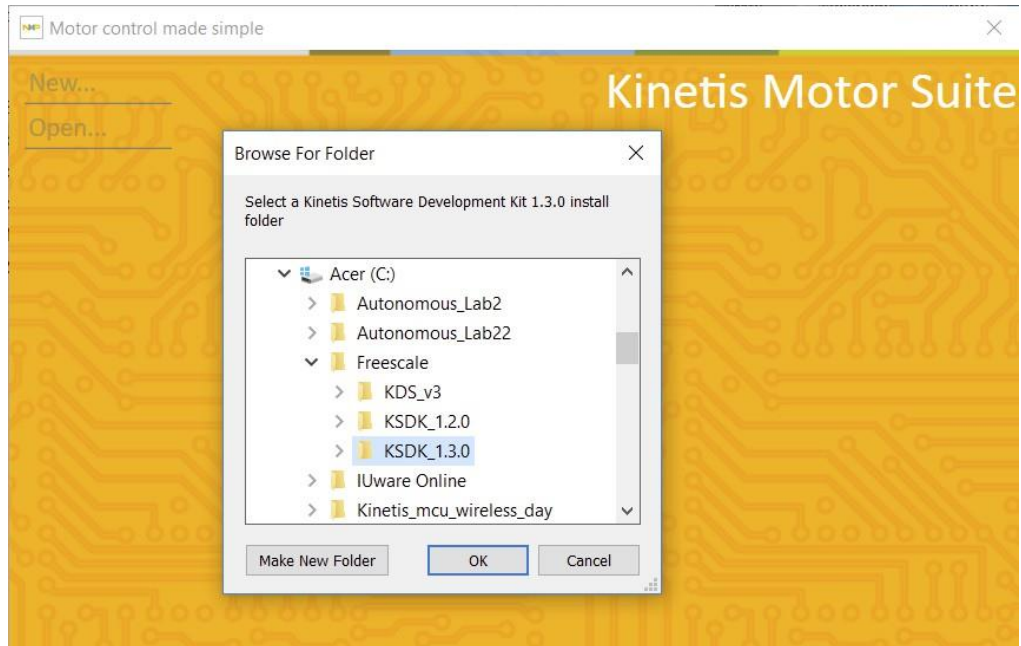
C:\Users\Sharkawy\Documents...

📁

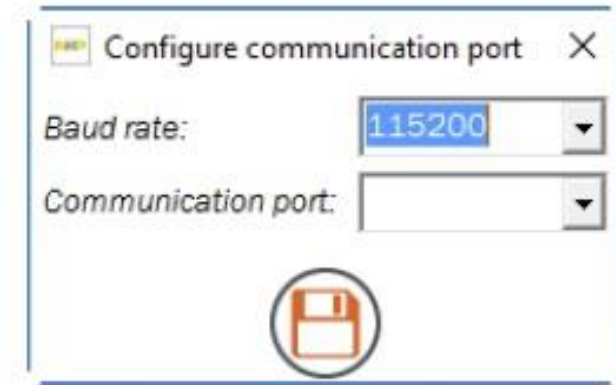
- After selecting all the above configuration select path for your project to be saved, also you can set the project name.

- Go to Project → Select Path → Kinetis SDK
- Find and select the folder named KSDK_1.3.0. The default installation location is:
C:\Freescale\KSDK_1.3.0.

Specify the location of KSDK 1.3.0

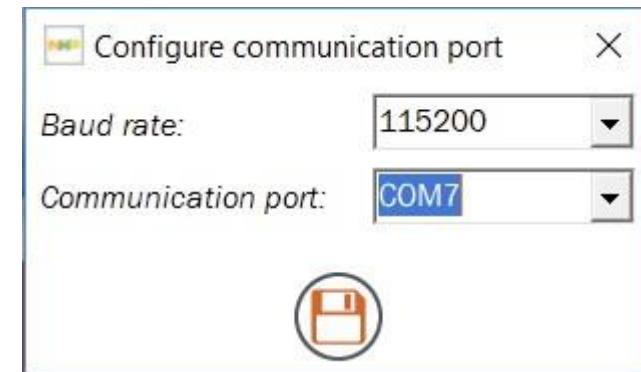


- Wait a minute or two for the unzipping and placement of the KMS reference project.
- Select the proper communication port for your board from the dropdown menu.

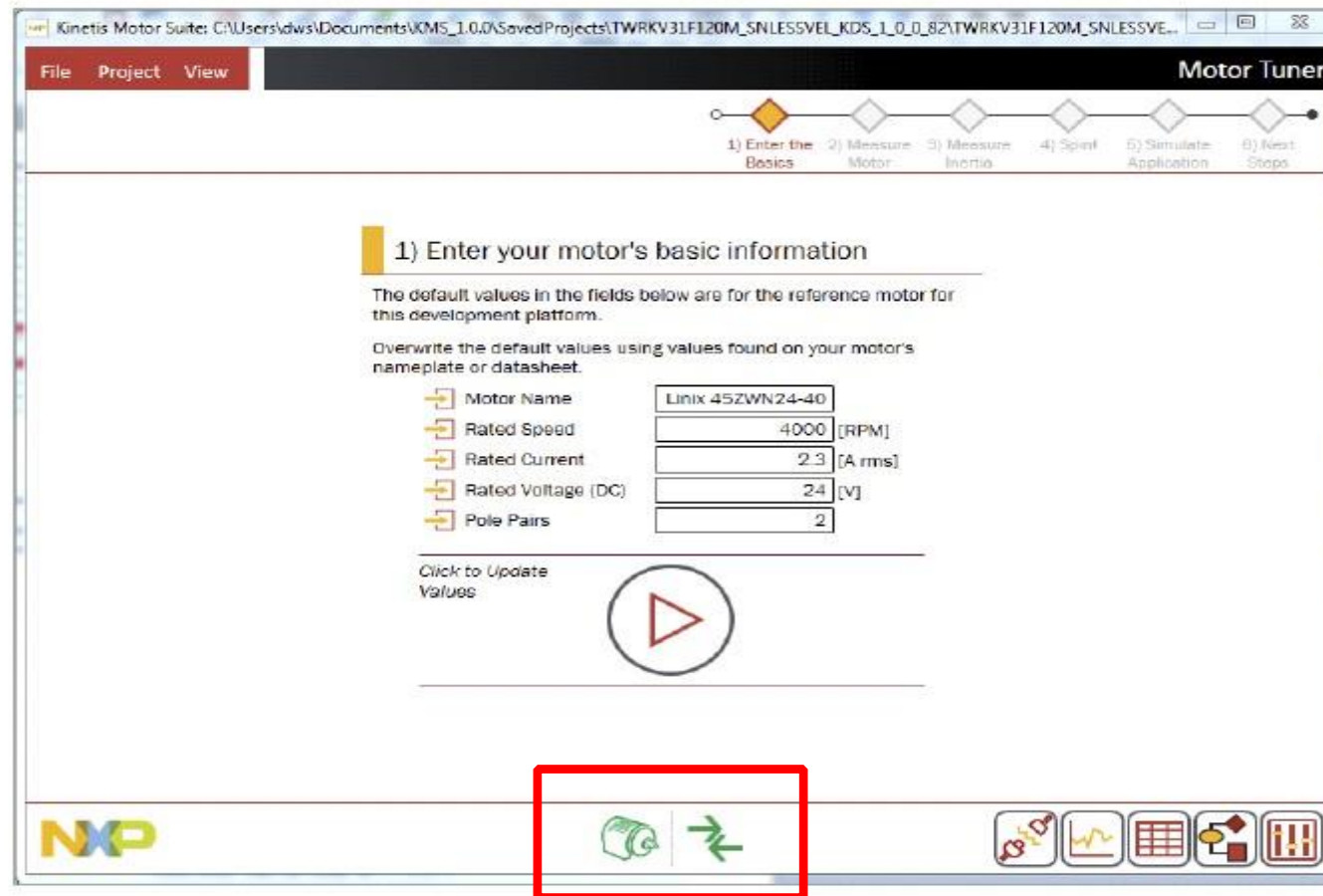


Select COM port (KV3)

- The proper communication port can be determined by looking in your Windows Device Manager (accessible from Control Panel) and finding the OpenSDA communication port.



- KMS should open to its Main Window and indicate successful communication by virtue of green arrows at the bottom of the screen.



Successful opening of
KMS main window

1. Enter the basics.

- KMS opens to the first page of Motor Tuner, the wizard-style interface for identifying and running your motor that is the subject of this lab.
- The first step in commissioning your system is to enter your motor's basic information.
- Linux 45ZWN24-40 information is prepopulated in KMS for the Freedom development platform (sensor less velocity).

1) Enter your motor's basic information

The default values in the fields below are for the reference motor for this development platform.

Overwrite the default values using values found on your motor's nameplate or datasheet.

→ Motor Name	<input type="text" value="Your Motor"/>
→ Rated Speed	<input type="text" value="4000"/> [RPM]
→ Rated Current	<input type="text" value="2.30"/> [A rms]
→ Rated Voltage (DC)	<input type="text" value="24"/> [V]
→ Pole Pairs	<input type="text" value="2"/>

Click to Update
Values



- Click to update the basic motor values.



2. Measure motor

- Motor Tuner applies voltage to measure the motor's resistance and inductance. Motor Tuner then rotates the motor's shaft at a low speed to measure the rotor flux. Click to measure your motor's characteristics.

2) Measure your motor's characteristics

Success! Motor Tuner will now energize and rotate your motor to measure its electrical characteristics. The motor should be bareshaft for this measurement.

This typically takes 20-30 seconds.

Start Motor
Measurement



→ Stator Resistance	0.719999 [Ohms]
→ Stator Inductance	0.000544 [H]
→ Rotor Flux	0.009882 [Wb]

3. Measure inertia

- Inertia is important for controlling the motion of the application but is often neglected by traditional approaches.
- Motor Tuner uses inertia as a direct input to create an appropriate model of the system for advanced control.
- Your motor is running without anything connected to the motor shaft, then Motor Tuner measures the inertia represented by the motor shaft.

- Click to start inertia measurement. The motor spins briefly. If successful in the default configuration, KMS also updates the reference project that you created upon launching KMS with your motor and inertia characteristics.

3) Measure your system's inertia*

Success! Motor Tuner will now quickly accelerate and decelerate your system to measure its mechanical characteristics. After this step is complete, all motor control settings will be saved.

Run this step with the motor bareshaft or connected to your application inertia. Do not apply load.*

*Start Inertia
Measurement*



 Inertia

0.000015 [A/(rpm/s)]

*Inertia is different from load. Think of a washing machine: the drum is the system inertia; the clothes are the load. Inertia is directly coupled to the motor and rotates with it; load is typically not coupled and impedes motion. Only inertia is relevant for this measurement.

4. Spin

- Motor Tuner automatically advances to the next step when the system inertia measurement is successful.
- In this step, Motor Tuner runs your motor to its rated speed, which you entered or confirmed on step 1) Enter the Basics.
- Click to run your motor to the rated speed.

Note: If it is not desirable to run your motor to rated speed (e.g., if your application is attached and the rated speed of the motor is too high for your application), overwrite the Target Speed value to a more suitable value before clicking to Run.

4) Spin your motor

Success! You are now ready to attempt to run your motor to its rated speed.

The KMS Software Oscilloscope will plot desired speed vs. speed feedback for approximately 15 seconds before stopping.

*Run Motor to Rated
Speed*



→ Target Speed

4000 [RPM]

→ Motor Speed

0 [RPM]

- Motor Tuner automatically activates the Software Oscilloscope so that you can watch the performance of your motor as it transitions to the rated speed. Commanded speed is plotted in red and estimated actual speed is plotted in green.
- When operating under sensor less velocity control, KMS transitions from open to closed loop speed control upon reaching 10% of the motor's rated speed.



Open to closed loop in TWR sensorless velocity control

- This results in a noticeable bump in the previous plot as the system attempts to lock onto a signal that is a function of speed and thus small when running at low speed.
- This transition from open to closed loop control is a key challenge of sensor less velocity control.
- By contrast, when running sensor-based velocity control, the motor is always in closed loop control due to the external sensor.
- This results in smoother operation at low speed, which is a chief benefit of sensor-based control. Nevertheless, the encoder must catch up when starting from zero, so shows a small divergence from commanded speed near zero.
- The motor spins to the Target Speed and remains at that speed for 5 seconds. After that, Motor Tuner closes the Software Oscilloscope and proceeds to the next step.

5. Simulate application.

- Motion Sequence Builder is one of the tools available in KMS. It allows you to easily build complex motion sequences and automatically generate application code.
- One of the provided motion sequences simulates simple washing machine behavior. The motor:
 - remains at 0 rpm speed during the "water fill" stage.
 - “agitates” by ramping to a certain speed, then reversing direction to reach the same speed in the opposite direction (repeating this behavior several times).
 - ramps to twice the agitation speed during the “spin cycle”.
 - comes to a halt and concludes the motion sequence.

- Click to start the washing machine trajectory.

5) Simulate an application

Success! But a real application requires operation at more than just rated speed.

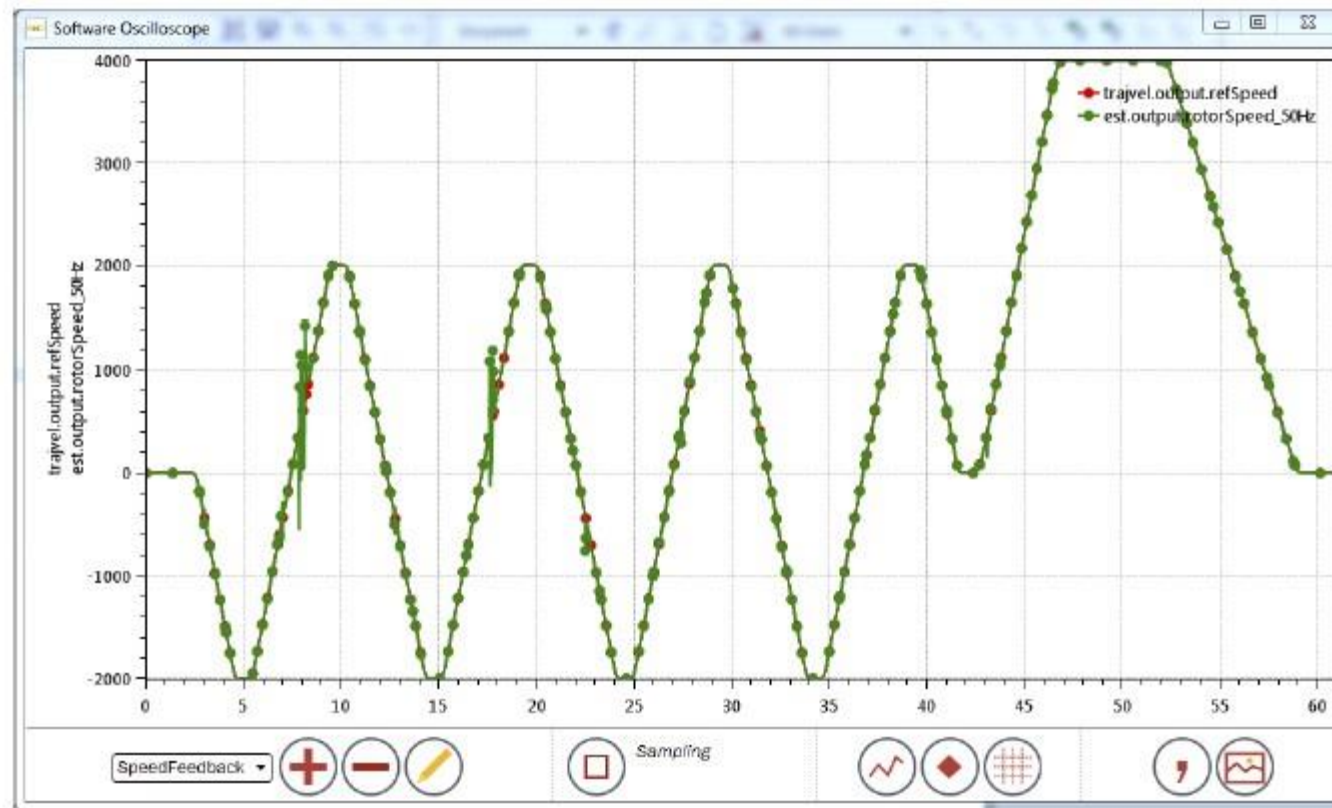
Click below to command your motor to follow a simple, one-minute trajectory that approximates washing machine motion.

*Start Washing
Machine Trajectory*



Click to start the washing machine trajectory

- Motor Tuner automatically launches the Software Oscilloscope so that you can observe your motor's performance as it executes the simulation.



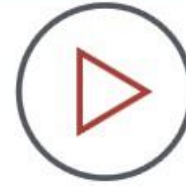
Sensorless velocity control plot of washing machine motion

Congratulations!

If you made it this far, your motor is running successfully! From here, you may choose to:

- Create your own application's trajectory with Motion Sequence Builder

*Launch Motion
Sequence Builder*



- Check performance at various speeds and optimize your motor control settings with Motor Manager

*Open Motor
Manager*



Motor Manager will be used in Part IV of this lab

Summary

- In this lab, you performed the following steps to spin your motor:
 - Entered basic information about the motor
 - Characterized the electrical and mechanical system
 - Ran the motor and validated its ability to achieve and hold rated speed

Part IV: Other methods for running your motor in KMS

Objective

- KMS is intended to allow flexibility for both beginner and expert users.
- In this part of the lab, spin your motor using **Motor Manager**, which is a superset of the functionality available in Motor Tuner, and the Watch Window, which allows runtime access to MCU variables.

1. Start Motor Manager

- Click on Open Motor Manager.

Congratulations!

If you made it this far, your motor is running successfully! From here, you may choose to:

- Create your own application's trajectory with Motion Sequence Builder

Launch Motion
Sequence Builder

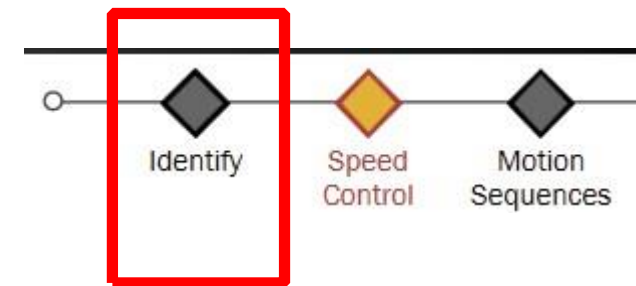
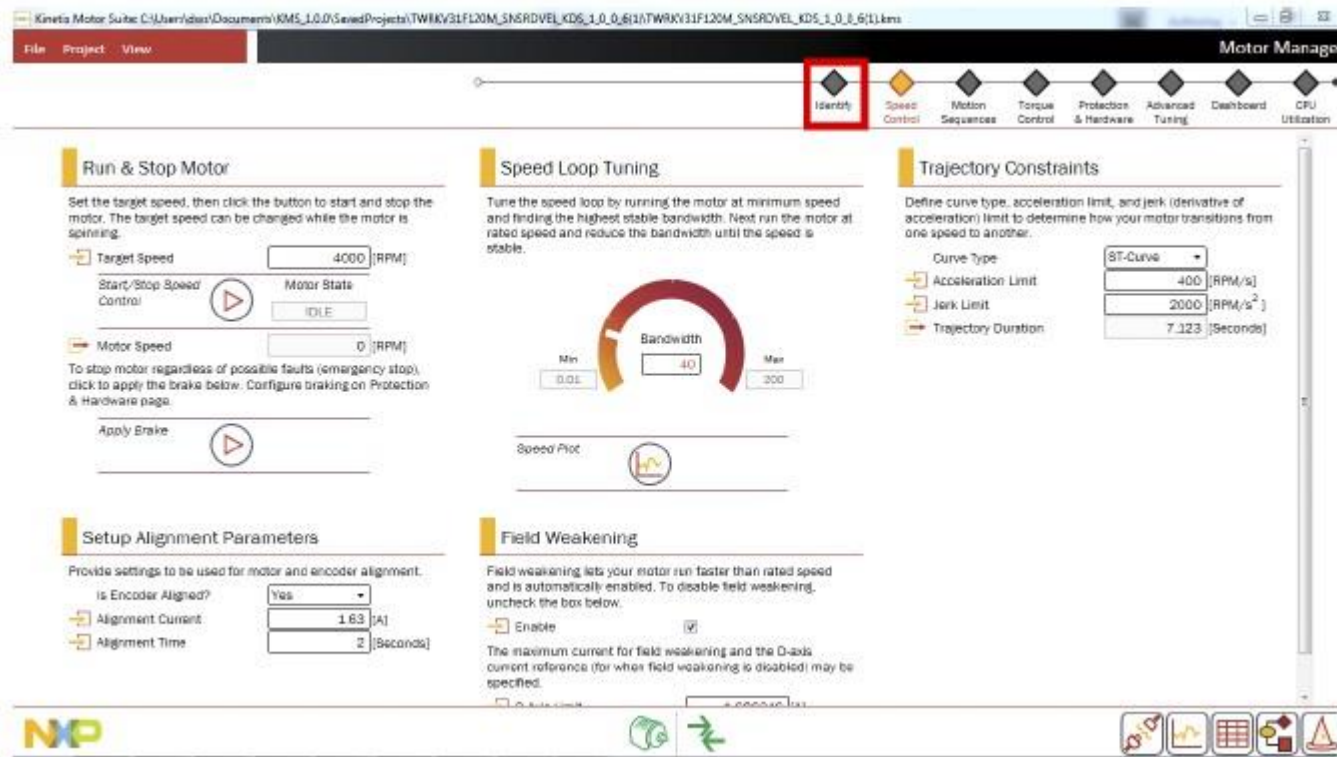


- Check performance at various speeds and optimize your motor control settings with Motor Manager

Open Motor
Manager



- Navigate to the Identify page by clicking on the appropriate selection in the navigation bar at top right.



Navigate to identify page

1. Basic motor information

- When using Motor Tuner to get your motor spinning, the first step was to enter the motor's basic information. The same is true for Motor Manager. Enter the basic information for your motor in Basic Motor Parameters.

Basic Motor Information

The default values in the fields below are for the reference motor for this development platform.

Overwrite the default values using values found on your motor's nameplate or datasheet.

→ Motor Name	<input type="text" value="Your Motor"/>
→ Rated Speed	<input type="text" value="4000"/> [RPM]
→ Rated Current	<input type="text" value="2.30"/> [A rms]
→ Rated Voltage (DC)	<input type="text" value="24"/> [V]
→ Pole Pairs	<input type="text" value="2"/>

2. Automatic parameter measurement

- Motor Tuner applied a voltage to measure the motor's resistance and inductance, and then rotated the motor's shaft at a low speed to measure the rotor flux. Motor Manager does the same thing.
- Click to start the motor measurement.

Automatic Parameter Measurement

Use the text fields below to configure how Motor Manager measures your motor's electrical characteristics, then click the button to start measurement. The motor should be bareshaft.

Inter values for two following parameters




 Rs Identification Current	<input type="text" value="80.00"/> [%]
 Ls Identification Current	<input type="text" value="80.00"/> [%]
 Enable Rotation	<input checked="" type="checkbox"/>
 Flux Identification Speed	<input type="text" value="15.00"/> [%]

Start Motor
Measurement






Status

Idle

 Stator Resistance	<input type="text" value="0.422109"/> [Ohms]
 Stator Inductance	<input type="text" value="0.000325"/> [H]
 Rotor Flux	<input type="text" value="0.013401"/> [Wb]

After identification, the recalculated motor drive tuning parameters are automatically loaded to the RAM of the MCU.

- After Motor Manager measures the motor parameters, they are updated on the screen

 Stator Resistance	<input type="text" value="0.425758"/> [Ohms]
 Stator Inductance	<input type="text" value="0.000332"/> [H]
 Rotor Flux	<input type="text" value="0.013412"/> [Wb]

After identification, the recalculated motor drive tuning parameters are automatically loaded to the RAM of the MCU.

3. System inertia measurement

- In lab 1, part II: “spin your motor using Motor Tuner”, we learned that Motor Tuner uses inertia as a direct input to create an appropriate model of the system for advanced control. Motor Manager does the same.







Note: When developing your application (not at this lab), connect the motor to the application inertia (anything that spins directly with the motor during operation) before running System Inertia Measurement.

- Scroll down to System Inertia Measurement then click to start the inertia measurement.
- During system inertia measurement, the motor accelerates and decelerates in a manner governed by the configuration parameters inertia identification speed and ramp time.
- Unlike Motor Tuner, Motor manager allows you to make manual adjustments to system inertia measurement configuration.




System Inertia Measurement

Connect the application inertia to the motor shaft but keep it unloaded. Click the button to start inertia measurement; your motor will quickly accelerate then decelerate.

If the measurement does not succeed, you may manually change the speed to which the motor will try to ramp and the time allotted for ramping.

	Inertia Identification Speed	<input type="text" value="4000.0"/>	[RPM]
	Ramp Time	<input type="text" value="3.5"/>	[s]
<div>Start Inertia Measurement</div> <div></div>		<div>Status</div> <div>Incomplete</div>	
	Inertia	<input type="text" value="0.000015"/>	[A/(rpm/s)]
	Friction	<input type="text" value="0.000065"/>	[A/rpm]
	Inertia Identification Error	<input type="text" value="0"/>	

- After Motor Manager identifies the system inertia, the inertia and friction values are displayed on the screen

 Inertia	<input type="text" value="0.000014"/> [A/(rpm/s)]
 Friction	<input type="text" value="0.000107"/> [A/rpm]
 Inertia Identification Error	<input type="text" value="0"/>

- KMS updates configuration values based on the measured motor parameters and inertia. These values can be saved to the Motor Observer reference project (created upon launching KMS) which can then be edited from your IDE. This happens automatically in Motor Tuner.
- In Motor Manager, you must click the Store Motor Information button to push your system settings into the Motor Observer reference project. You receive a notification that the header file that contains this information has been saved in a particular location within the directory you defined after launching KMS.
- Click OK after receiving this notification.

Store Motor Parameters

After identifying the motor parameters and system inertia, click the button below to generate a header file that will update the KMS firmware reference project with your system's settings.

*Store Motor
Information*



Store motor parameters in a header file



Notification



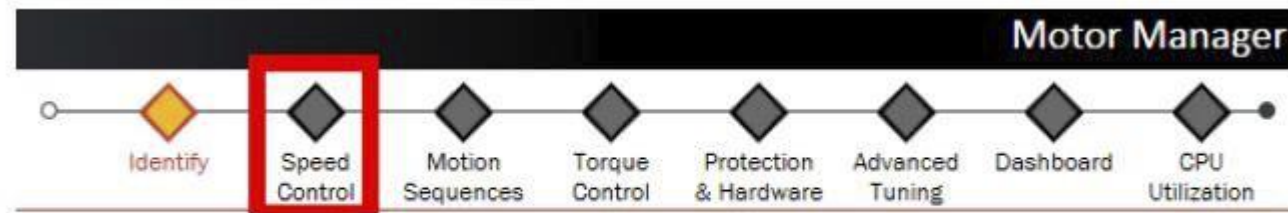
Saved system.h to C:\Users\Sharkawy\Documents\KMS_1.1.0
\SavedProjects\FRDMKV31F_SNLESSVEL_KDS_1_1_0_312(9)
\inc\system.h



Notification that system settings have been updated

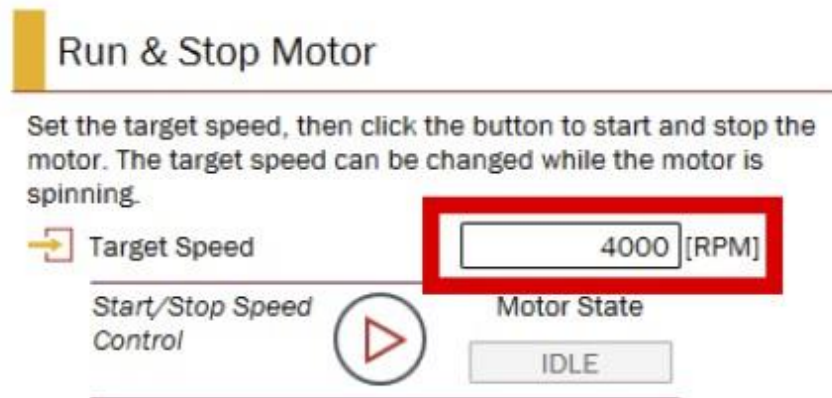
4. Start Motor.

- Click on the Speed Control page icon at top



Navigate to the Speed Control page

- Enter the motor's rated speed as its Target Speed

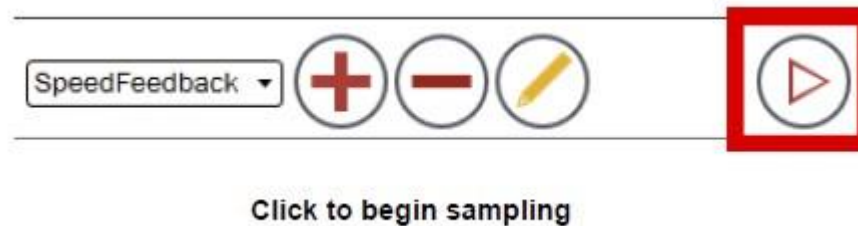


Enter target speed

- Click to activate the Software Oscilloscope's Speed Plot.



- Use the Software Oscilloscope to view your motor's performance. Click the Run button at the bottom of the oscilloscope display to start sampling.



- Back in Motor Manager, click to Start Speed Control.


Run & Stop Motor

Set the target speed, then click the button to start and stop the motor. The target speed can be changed while the motor is spinning.

→ Target Speed

[RPM]

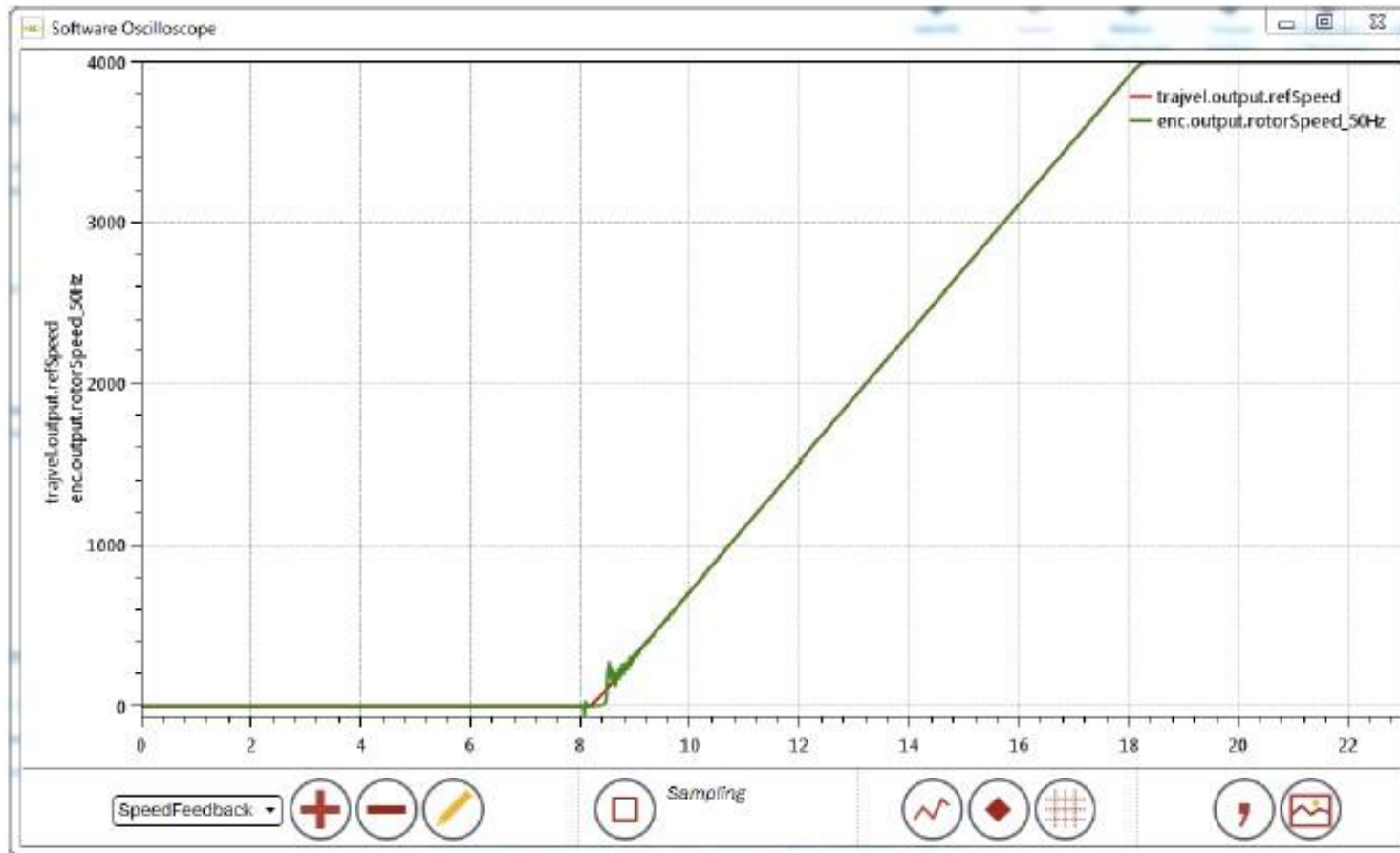
Start/Stop Speed Control

A square button with a red border containing a play icon (a right-pointing triangle inside a circle).

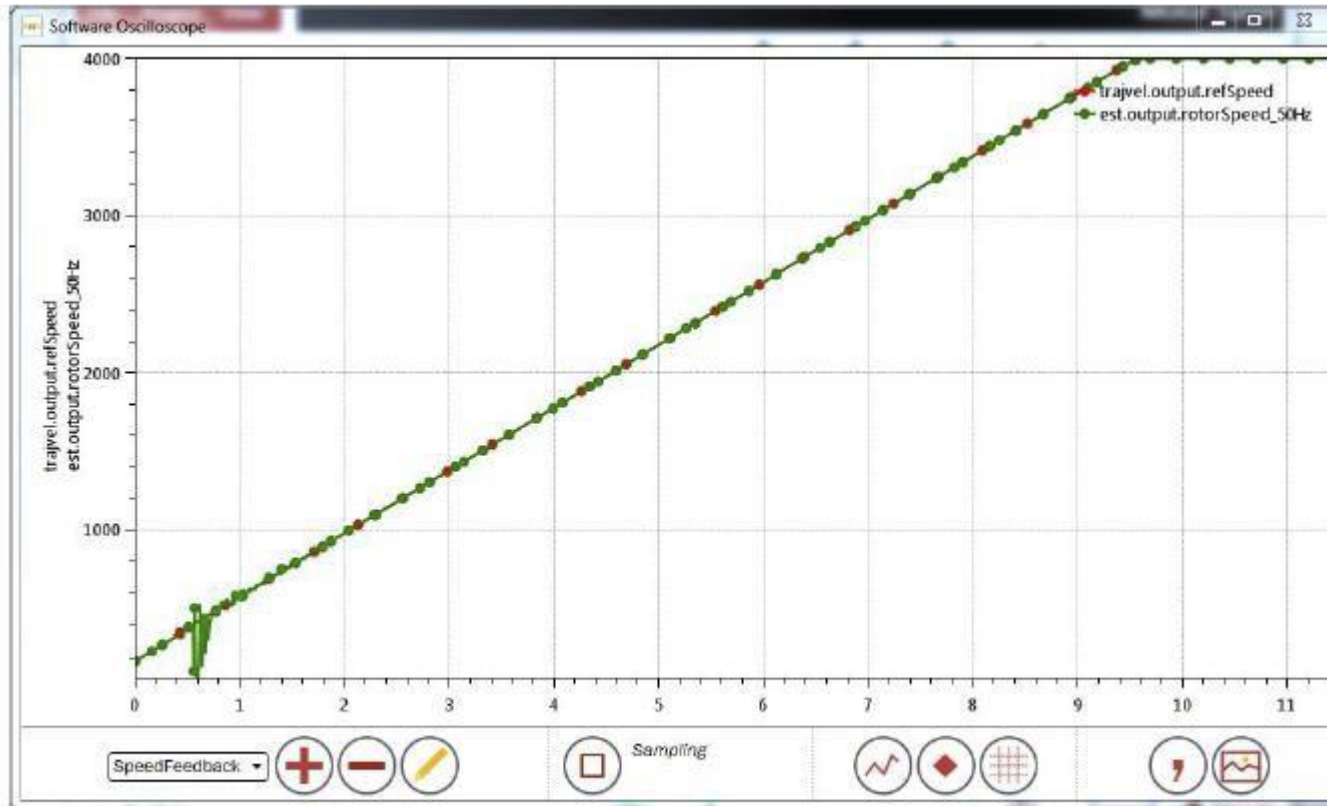
Motor State

Click to start the motor

- You should see your motor ramp to the target speed.



Ramp to target speed (sensored)



Ramp to target speed (sensorless)

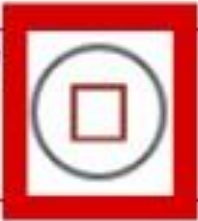
- After the motor reaches and holds rated speed, as in Motor Tuner, click to Stop sampling.



- In Motor Manager, click to Stop the motor.

Run & Stop Motor

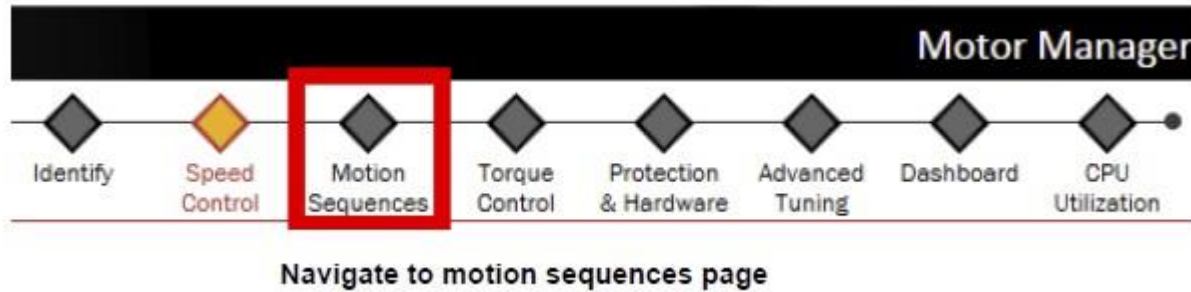
Set the target speed, then click the button to start and stop the motor. The target speed can be changed while the motor is spinning.

Target Speed	<input type="text" value="4000"/>	[RPM]
Start/Stop Speed Control		Motor State
	<input type="button" value="RUN SPEED"/>	
Motor Speed	<input type="text" value="4000"/>	[RPM]

Click to stop motor

5. Run motion sequence

- Click on the Motion Sequences page icon at top.

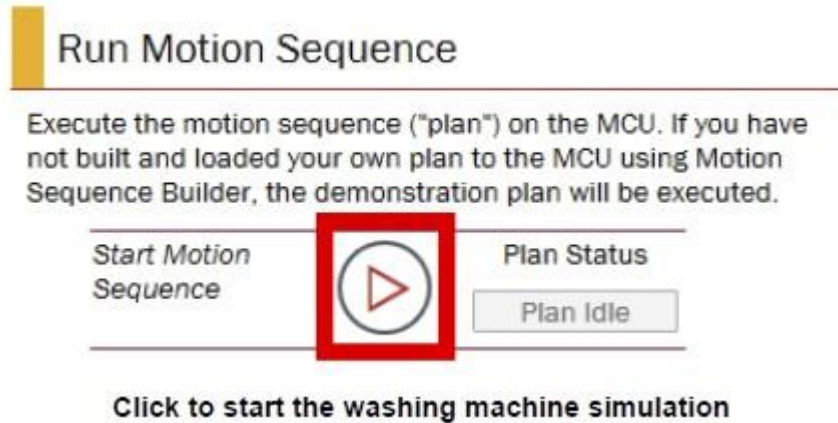


- In the Software Oscilloscope, click to restart sampling.



Click to begin sampling button

- Click to Start the washing machine simulation

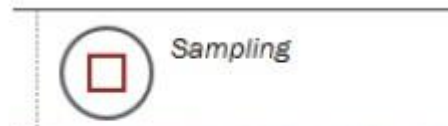


- Return to the Software Oscilloscope so that you can view your motor's performance.
- Watch your new motor run the same simple washing machine simulation as in "Velocity lab : spin your motor using Motor Tuner".



Sensored velocity example of washing machine simulation

- Click to stop sampling.



Click to stop sampling

6. Start Watch Window

- While Motor Manager offers more granular control of the core functions performed by Motor Tuner, the Watch Window goes even further.
- Via runtime access to MCU variables, the Watch Window can be used to set in motion the same functions as Motor Tuner and Motor Manager.
- However, the Watch Window exercises these functions as discrete elements, instead of as an integrated system.
- For example, identifying motor parameters from the Watch Window will not automatically adapt the motor drive configuration based on updated values, as occurs in Motor Tuner and Motor Manager.
- Thus, to use the Watch Window properly, you should have some familiarity with the KMS firmware and with motor control principles.

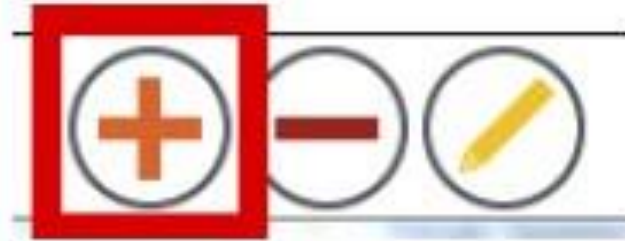
- The following steps, wherein only a subset of the functions described for Motor Tuner and Motor Manager are exercised, serve to reinforce the discrete nature of operation from the Watch Window.
- Click to activate the Watch Window using the button at bottom right



Click to activate Watch Window

7. Automatic parameter measurement - flux only

- Click the Add button at bottom left to show a list of the parsed variables available from the MCU.

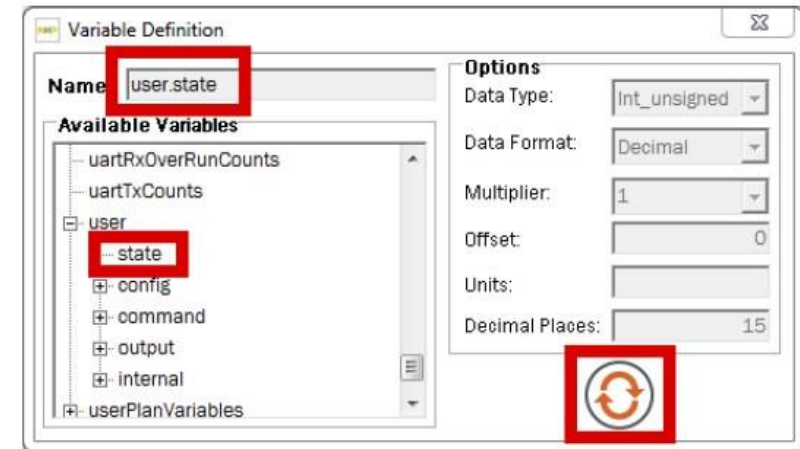


Click to add an MCU variable to the Watch Window

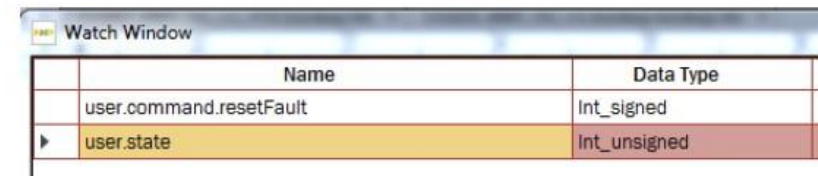
- Scroll down to locate the “user” group, which aggregates variables found in the User module of Motor Observer firmware.

- Expand the user group, select “state,” and click to refresh.

- The variable user.state is added to the Watch Window.

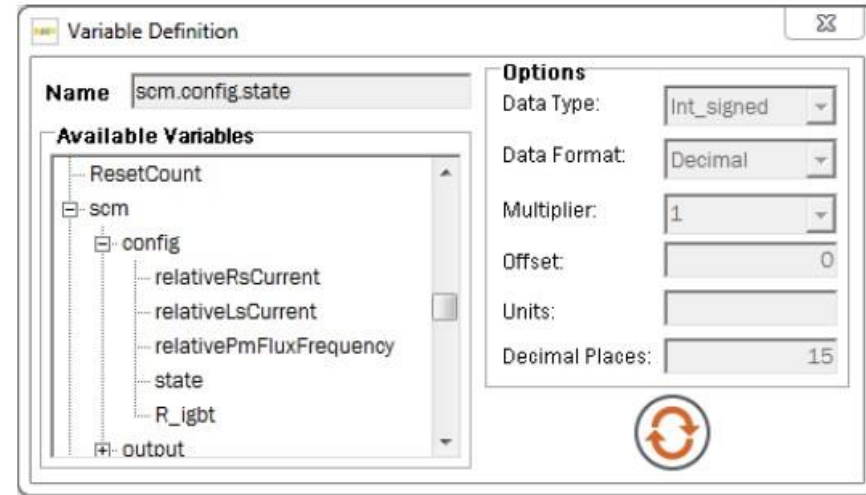


Add user.state to Watch Window



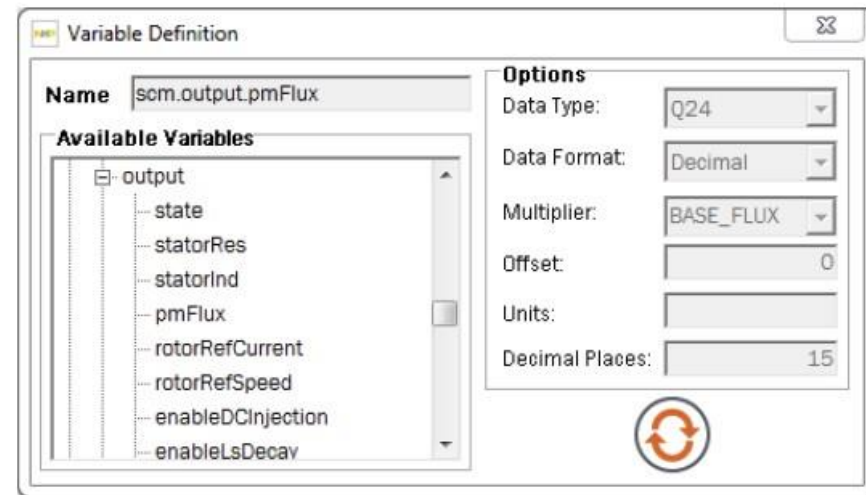
User.state added to Watch Window

- The variable `user.state` is intended to provide easy access to KMS' operating modes, including several that are responsible for the Motor Tuner and Motor Manager functions you have already seen.



Add scm.config.state

- To replicate a part of the motor measurement routine, click the Add button again and find the variable `scm.config.state`.
- Do the same for `scm.output.pmFlux`.



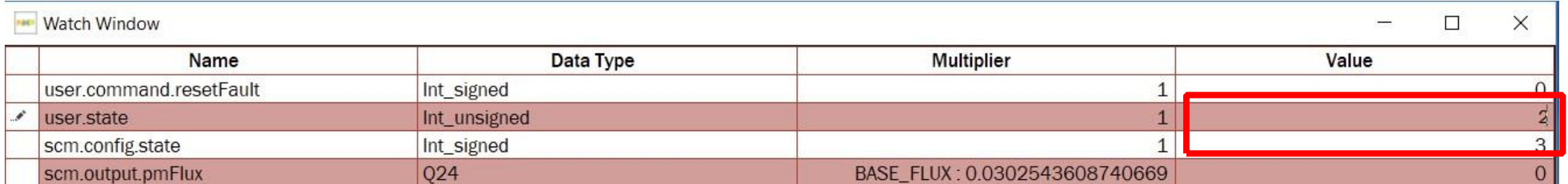
Add scm.output.pmFlux

- The term “scm” refers to variables used in the Self Commissioning module of KMS firmware.
- Scm.config.state determines which value the system is trying to measure (resistance, inductance, or flux), and scm.output.pmFlux is the result of the flux measurement.
- To run the flux measurement from the Watch Window, first click the Run button to activate live updating of the Watch Window variables.



Click to start live update of the Watch Window

- Type 3 into the Value column for scm.config.state. This specifies flux as the desired measurement.
- Type 2 into the Value column for user.state. This tells KMS firmware to run in Self Commissioning mode - and thus measure flux given the value configured in the previous step.

A screenshot of a 'Watch Window' from a debugger. It contains a table with four columns: Name, Data Type, Multiplier, and Value. The rows are: user.command.resetFault (Int_signed, 1, 0), user.state (Int_unsigned, 1, 2), scm.config.state (Int_signed, 1, 3), and scm.output.pmFlux (Q24, BASE_FLUX : 0.0302543608740669, 0). The 'user.state' and 'scm.config.state' rows are highlighted with a red box, and the 'Value' column for 'user.state' is also highlighted with a red box.

Name	Data Type	Multiplier	Value
user.command.resetFault	Int_signed	1	0
user.state	Int_unsigned	1	2
scm.config.state	Int_signed	1	3
scm.output.pmFlux	Q24	BASE_FLUX : 0.0302543608740669	0

(you may need to stop and run again)

- The motor spins and on completion of the measurement, the value for scm.output.pmFlux will update in the Watch Window.
- User.state and scm.config.state return to the idle, 0 state.

	Name	Data Type	Multiplier	Value
	user.command.resetFault	Int_signed	1	0
▶	user.state	Int_unsigned	1	0
	scm.config.state	Int_signed	1	0
	scm.output.pmFlux	Q24	BASE_FLUX : 0.0302543608740669	0.0138101057032259

Updated flux value

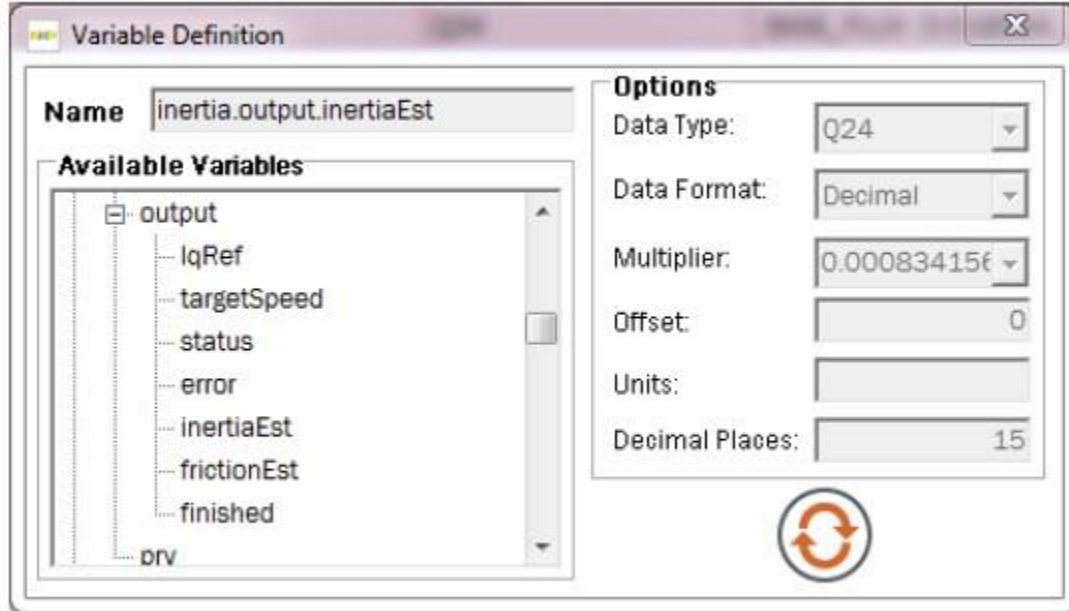
8. System inertia measurement

- Click Stop button at bottom middle of Watch Window to halt live updating of MCU variables.



Click to stop updating in Watch Window

- Now add the variable inertia.output.inertiaEst.
- This represents the value resulting from the system inertia measurement



Add inertia estimate to Watch Window

- Click to Run live updating in the Watch Window.



Click to start live update of the Watch Window

- Enter a value of 3 into the Value column for user.state.
- This will place the firmware into inertia estimation mode.
- The motor spins and the value for inertia.est.inertiaEst updates.

inertia.output.inertiaEst	Q24	0.000859375	1.41683127731085E-05
---------------------------	-----	-------------	----------------------

Updated inertia value

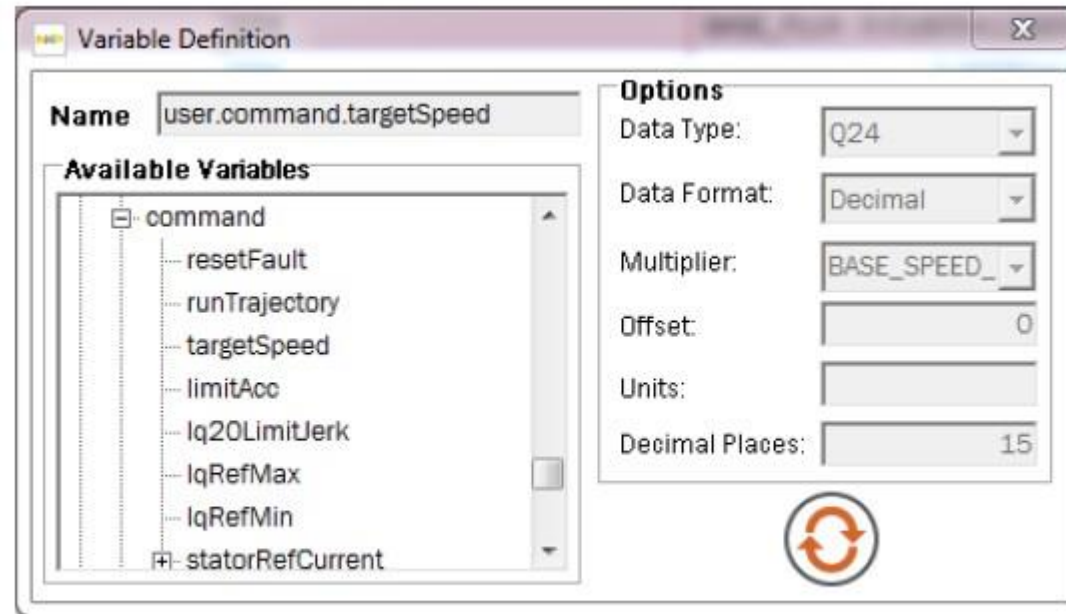
9. Start Motor

- Click to Stop updating the Watch Window again



Click to stop updating in Watch Window

- Find and add the variable `user.command.targetSpeed`.
- This represents the speed command being sent to the motor.



Add user.command.targetSpeed

- Find and add the variable showing the motor's feedback speed. In sensorless operation, this information comes from the Estimator module (EST).

Motor speed feedback variable names

Control type	Variable name
Sensorless velocity	est.output.rotorSpeed_50Hz
Sensored velocity	enc.output.rotorSpeed_50Hz

Variable Definition

Name

est.output.rotorSpeed_50Hz

Available Variables

output

rotorAngle

rotorSpeed

rotorSpeed_50Hz

statorCurrentMagnitude

statorVoltageMagnitude

statorRefVoltageMagnitude

torque

activePower

Options

Data Type:

Q24

Data Format:

Decimal

Multiplier:

BASE_SPEED_


Offset:

0

Units:

Decimal Places:

15



Add motor speed feedback variable (sensorless velocity example)

- Click to Run live updating in the Watch Window.

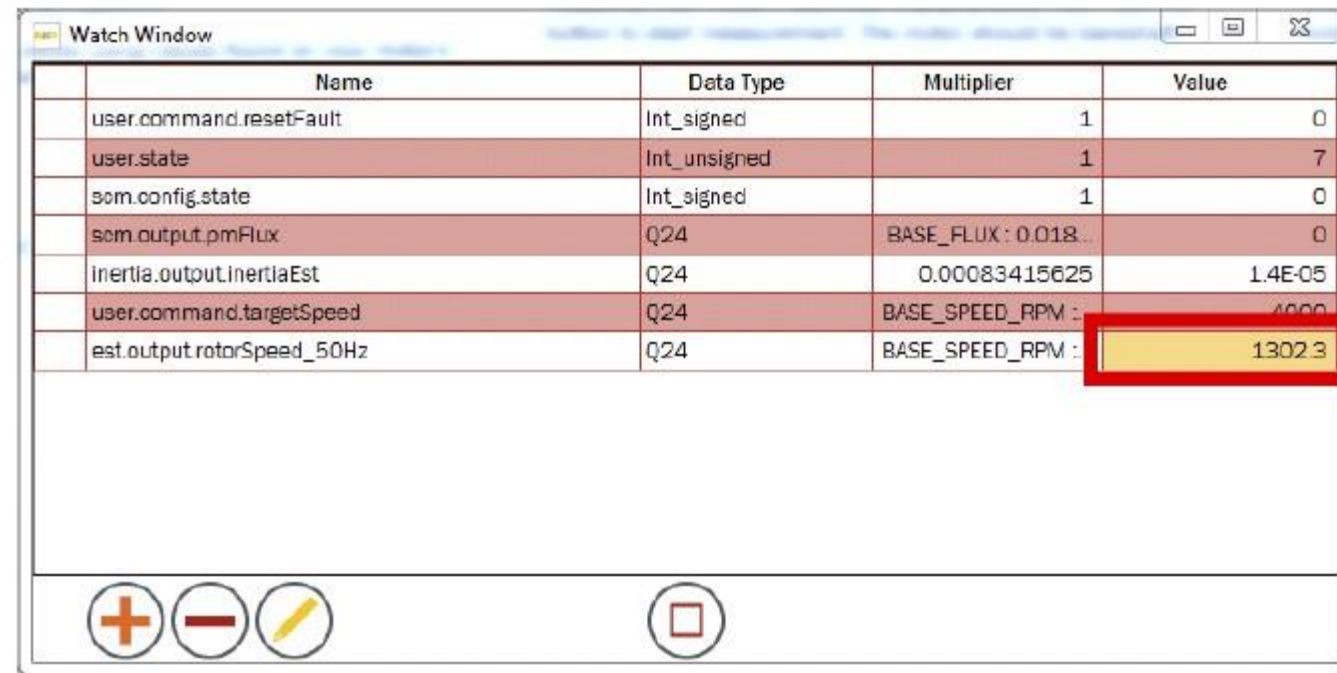


Click to start live update of the Watch Window

- Enter the Rated Speed for your motor into the Value column for user.command.targetSpeed, then type a value of 7 into the Value column for user.state.
- This value for user.state places the motor into Speed Control mode.

Watch Window				
Name	Data Type	Multiplier	Value	
user.command.resetFault	Int_signed			
user.state	Int_unsigned			7
sem.config.state	Int_signed			0
sem.output.pwmFlux	Q24	BASE_FLUX: 0.03805413880698		0
inertia.output.inertiaEst	Q24	0.00083415825		0
user.command.targetSpeed	Q24	BASE_SPEED_RPM: 960		4000

- The motor spins to Rated Speed just as it did in Motor Tuner and Motor Manager.
- Observe the motor speed feedback variable to see it gradually reach the Target Speed.

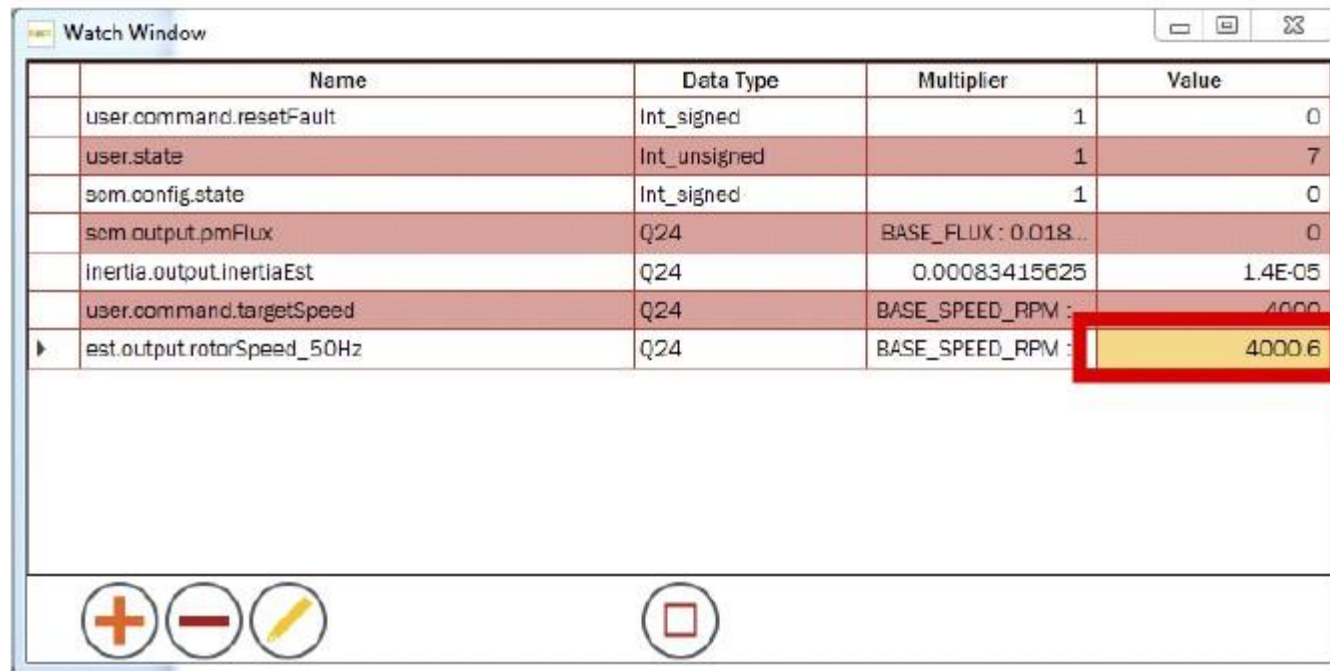


Name	Data Type	Multiplier	Value
user.command.resetFault	Int_signed	1	0
user.state	Int_unsigned	1	7
scm.config.state	Int_signed	1	0
scm.output.pmFlux	Q24	BASE_FLUX : 0.018...	0
Inertia.output.InertiaEst	Q24	0.00083415625	1.4E-05
user.command.targetSpeed	Q24	BASE_SPEED_RPM :	1000
est.output.rotorSpeed_50Hz	Q24	BASE_SPEED_RPM :	1302.3

Motor speed feedback during acceleration to Target Speed (sensorless velocity)

10. Run motion sequence

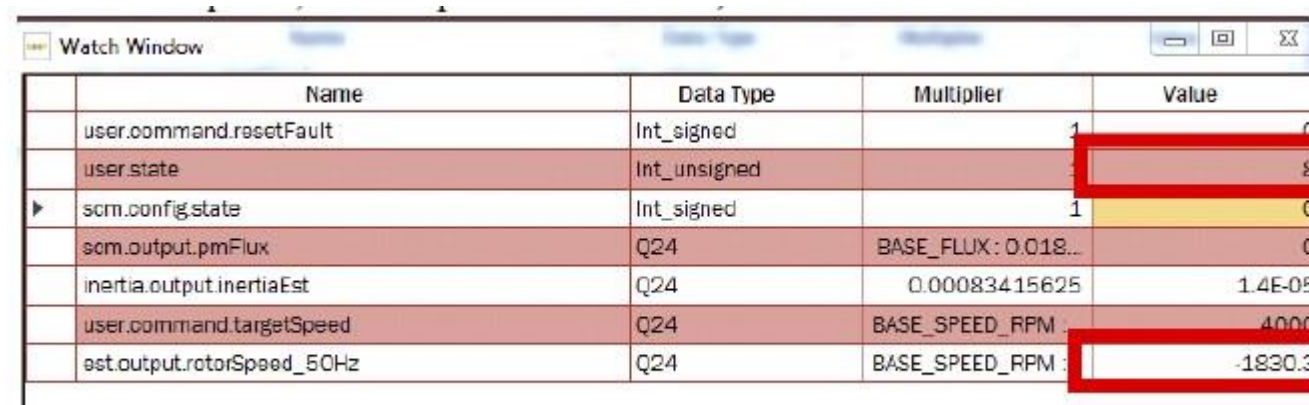
- Wait for the motor to reach Rated Speed according to your speed feedback value.



Name	Data Type	Multiplier	Value
user.command.resetFault	Int_signed	1	0
user.state	Int_unsigned	1	7
scm.config.state	Int_signed	1	0
scm.output.pmFlux	Q24	BASE_FLUX : 0.018...	0
inertia.output.inertiaEst	Q24	0.00083415625	1.4E-05
user.command.targetSpeed	Q24	BASE_SPEED_RPM :	4000
est.output.rotorSpeed_50Hz	Q24	BASE_SPEED_RPM :	4000.6

Motor running at Rated Speed (sensorless velocity)

- With the Watch Window still live updating, type a value of 8 into the Value column for user.state.
- This starts the application motion sequence.
- First the motor slows to a stop, then proceeds through the washing machine sequence. Verify this by watching motor speed feedback (user.command.targetSpeed does not update because the speed reference is now coming from the motion sequence, not an explicit user command).



The screenshot shows a 'Watch Window' with a table of variables. The 'user.state' variable is highlighted in red, and its value '8' is also highlighted in a red box. The 'user.command.targetSpeed' variable is highlighted in red, and its value '4000' is also highlighted in a red box. The 'est.output.rotorSpeed_50Hz' variable is highlighted in red, and its value '-1830.3' is also highlighted in a red box.

Name	Data Type	Multiplier	Value
user.command.resetFault	Int_signed	1	0
user.state	Int_unsigned		8
scm.config.state	Int_signed	1	0
scm.output.pmFlux	Q24	BASE_FLUX : 0.018...	0
inertia.output.inertiaEst	Q24	0.00083415625	1.4E-05
user.command.targetSpeed	Q24	BASE_SPEED_RPM :	4000
est.output.rotorSpeed_50Hz	Q24	BASE_SPEED_RPM :	-1830.3

Motor running washing machine trajectory

- Click to Stop updating the Watch Window after the motion sequence is complete.



Click to stop updating in Watch Window

Summary

- In this lab, you performed the following steps to spin your motor outside of the Motor Tuner wizard:
 - measured motor and inertia from Motor Manager's Identify page
 - commanded the motor to run at a configurable speed from Motor Manager's Speed Control page
 - ran the example motion sequence from Motor Manager's Motion Sequences page
 - used the Watch Window to explore how MCU variables relate to PC GUI functions
 - triggered motor & inertia measurements, speed control, and motion sequence operation from the Watch Window