Motor Control Demonstrations Help

MPLAB Harmony Integrated Software Framework

Volume I: Getting Started With MPLAB Harmony Libraries and Applications

This volume introduces the MPLAB® Harmony Integrated Software Framework.

Description

MPLAB Harmony is a layered framework of modular libraries that provide flexible and interoperable software "building blocks" for developing embedded PIC32 applications. MPLAB Harmony is also part of a broad and expandable ecosystem, providing demonstration applications, third-party offerings, and convenient development tools, such as the MPLAB Harmony Configurator (MHC), which integrate with the MPLAB X IDE and MPLAB XC32 language tools.

Legal Notices

Please review the *Software License Agreement* prior to using MPLAB Harmony. It is the responsibility of the end-user to know and understand the software license agreement terms regarding the Microchip and third-party software that is provided in this installation. A copy of the agreement is available in the `<install-dir>/doc` folder of your MPLAB Harmony installation.

The OPENRTOS® demonstrations provided in MPLAB Harmony use the OPENRTOS evaluation license, which is meant for demonstration purposes only. Customers desiring development and production on OPENRTOS must procure a suitable license. Please refer to one of the following documents, which are located in the `<install-dir>/third_party/rtos/OPENRTOS/Documents` folder of your MPLAB Harmony installation, for information on obtaining an evaluation license for your device:

- `OpenRTOS Click Thru Eval License PIC32MXxx.pdf`
- `OpenRTOS Click Thru Eval License PIC32MZxx.pdf`

TIP! Throughout this documentation, occurrences of `<install-dir>` refer to the default MPLAB Harmony installation path:

- Windows: `C:/microchip/harmony/<version>`
- Mac OS/Linux: `~/microchip/harmony/<version>`

Applications Help

This section provides information on the various application demonstrations that are included in MPLAB Harmony.

Description

Applications determine how MPLAB Harmony libraries (device drivers, middleware, and system services) are used to do something useful. In a MPLAB Harmony system, there may be one main application, there may be multiple independent applications or there may be one or more Operating System (OS) specific applications. Applications interact with MPLAB Harmony libraries through well defined interfaces. Applications may operate in a strictly polling environment, they may be interrupt driven, they may be executed in OS-specific threads, or they may be written so as to be flexible and easily configured for any of these environments. Applications generally fit into one of the following categories.

Demonstration Applications

Demonstration applications are provided (with MPLAB Harmony or in separate installations) to demonstrate typical or interesting usage models of one or more MPLAB Harmony libraries. Demonstration applications can demonstrate realistic solutions to real-life problems.

Sample Applications

Sample applications are extremely simple applications provided with MPLAB Harmony as examples of how to use individual features of a library. They will not normally accomplish anything useful on their own. They are provided primarily as documentation to show how to use a library.

Motor Control Demonstrations

This section provides descriptions of the Motor Control demonstrations.

MPLAB Harmony is available for download from the Microchip website by visiting: http://www.microchip.com/mplabharmony. Once you are on the site, click the Downloads tab to access the appropriate download for your operating system. For additional information on this demonstration, refer to the "Applications Help" section in the MPLAB Harmony Help.

Introduction

Motor Control Demonstration Applications Help.

Description

This installation of MPLAB Harmony consists of two broad categories of motor control demonstrations.

Stand-alone Sensorless Field Oriented Control of PMSM Motor Using a PLL-based Estimator

This demonstration is designed to work with PIC32MK 100-pin Motor Control PIM mounted on the dsPICDEM™ MCLV-2 Development Board for Small Hurst Motor (DMB0224C10002), and has two variants:

- Dual Shunt PLL Estimator-based FOC demonstration on PIC32MK using an external Op amp configuration
- Dual Shunt PLL Estimator-based FOC demonstration on PIC32MK using an internal Op amp configuration

Please refer to the Microchip application note, AN2520 *Sensorless Field Oriented Control (FOC) for a Permanent Magnet Synchronous Motor (PMSM) Using a PLL Estimator and Flux Weakening (FW)* for more details

Integrated Application of Average Current Mode Power Factor Correction and Sensorless Field Oriented Control of a PMSM Motor

This demonstration is designed to work with PIC32MK 100-pin Motor Control PIM mounted on the dsPICDEM™ MCHV-3 Development Board (High Voltage), and has two variants:

- Integrated PFC and FOC demonstration on PIC32MK using an external Op amp configuration
- Integrated PFC and FOC demonstration on PIC32MK using an internal Op amp configuration

Please refer to the Microchip application note, *Integrated Power Factor Correction (PFC) and Sensorless Field Oriented Control (FOC) for Microchip 32-bit Microcontrollers* for more details.

To learn more about MPLAB Harmony stacks and libraries refer to the related documentation in Volume V: MPLAB Harmony Framework Reference.

Demonstrations

This topic provides information on how to run the Motor Control demonstration applications included in this release.

dualshunt_pll_foc_mclv2_ext_opamp

This section provides information on the supported demonstration boards, how to configure the hardware, and how to run the demonstration.

Description

This demonstration implements Sensorless Field Oriented Control (FOC) of a PMSM motor using a dual shunt configuration, and utilizes external (off-chip) Op amps for current sensing.

Building the Application

This section identifies the MPLAB X IDE project name and location and lists and describes the available configurations for the demonstration.

Description

To build this project, you must open the `dualshunt_pll_foc_mclv2_ext_opamp.X` project in MPLAB X IDE, and then select the desired configuration.

The following tables list and describe the project and supported configurations. The parent folder for these files is `<install-dir>/apps/motor_control/dualshunt_pll_foc_mclv2_ext_opamp`.

MPLAB X IDE Project

This table lists the name and location of the MPLAB X IDE project folder for the demonstration.

Project Name	Location
`dualshunt_pll_foc_mclv2_ext_opamp.X`	`<install-dir>/apps/motor_control/dualshunt_pll_foc_mclv2_ext_opamp/firmware`

MPLAB X IDE Project Configurations

This table lists and describes the supported configurations of the demonstration, which are within `./firmware/src/system_config`.

Project Configuration Name	BSP(s) Used	Description
small_hurst_motor_mclv2_ext_opamp	Not applicable.	Demonstrates dual shunt PLL estimator-based FOC of a Small Hurst Motor (DMB0224C10002) using external op-amps.

Configuring the Hardware

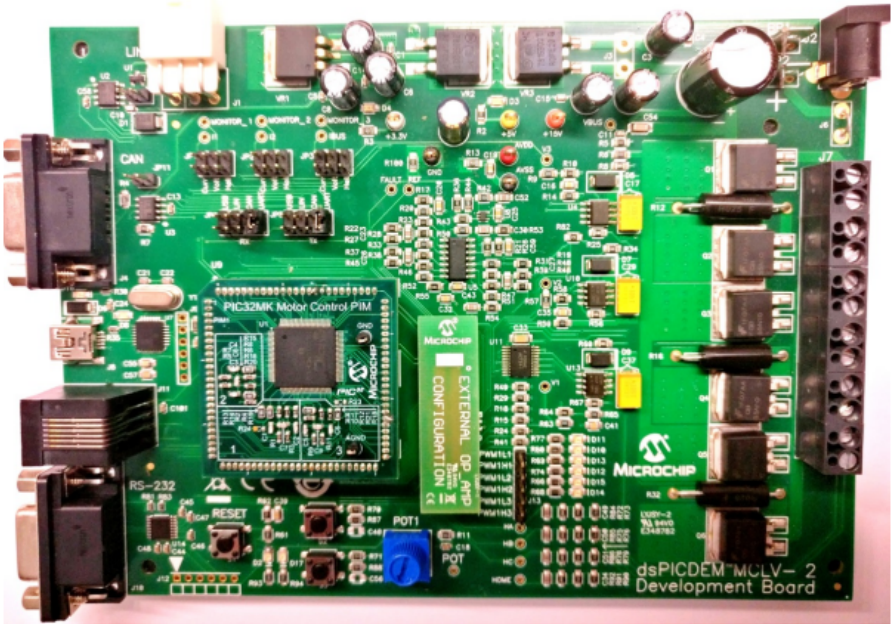
Describes how to configure the supported hardware.

Description

PIC32MK 100-pin Motor Control Plug-in Module (PIM) with the dsPICDEM MCLV-2 Development Board

1. Mount the PIC32MK 100-pin Motor Control PIM on the dsPICDEM MCLV-2 Development Board, as shown in the following figure.
2. Connect the External Op amp Configuration Matrix Board to header J14.
3. Connect the three-phase wires coming from P1 header of the small Hurst motor (i.e., Red, White and Black to M1, M2 and M3 ports, respectively). Leave the green wire from the motor unconnected.
4. Connect the programmer/debugger using the J11 connector.
5. Apply 24V DC at BP1-BP2/J2.
6. This application supports communication with the X2C-Scope plug-in to monitor or plot any variables used in the application. The communication between the target and the X2C-Scope plug-in can be established using the RS-232 serial port (J10) or mini-USB port (J8). To use the RS-232 serial port, connect JP4 and JP5 to the UART position. To use the mini USB port, connect JP4 and JP5 to the USB position. Refer to Using the X2C Scope for details on how to use X2C Scope in this demonstration.

PIC32MK MC PIM with dsPICDEM™ MCLV-2 using the External OPAMP Configuration



Running the Demonstration

Provides instructions on how to build and run the demonstration.

Description

Do the following to run the demonstration:

1. Compile and build the project.
2. Program the target device.
3. Press the ‘S2’ switch to start spinning the motor.
4. Vary the potentiometer, P1, to change the motor speed.
5. Press the 'S1’ switch to stop the motor.

Modifying the System Parameters

The system parameters, such as motor coefficients, maximum current, etc., can be modified in the header file `mc_app.h`, as shown in the following code example.

```
//==============================================================
// Following parameters for MCLV-2 board
// Gain of op amp = 15
// Shunt resistor = 0.025 ohms
// DC offset = 1.65V
// Max current = x
// (x * 0.025 * 15) + 1.65V = 3.3V
// x = 4.4Amps
#define      MAX_BOARD_CURRENT                 (float)(4.4)
#define      MAX_MOTOR_CURRENT                 (float)(4.2)
#define      MAX_MOTOR_CURRENT_SQUARED         (float)((float)MAX_MOTOR_CURRENT*
                                               (float)MAX_MOTOR_CURRENT)
#define      VREF_DAC_VALUE                    (int) 2048
#define      ADC_CURRENT_SCALE                 (float)(MAX_BOARD_CURRENT/(float)2048)
#define      CURRENT_LIMIT_CMP_REF             (int)(((float)2048*
                                               (MAX_MOTOR_CURRENT/MAX_BOARD_CURRENT))
                                               +VREF_DAC_VALUE)
#define      MOTOR_PER_PHASE_RESISTANCE        ((float)2.10)     // Resistance in Ohms
#define      MOTOR_PER_PHASE_INDUCTANCE        ((float)0.00192)  // Inductance in Henrys
#define      MOTOR_BACK_EMF_CONSTANT_Vpeak_Line_Line_KRPM_MECH   (float)7.24
                                               // Back EMF Constant in Vpeak/KRPM
#define      NOPOLESPAIRS                      5
#define      MAX_ADC_COUNT                     (float)4095     // for 12-bit ADC
#define      MAX_ADC_INPUT_VOLTAGE             (float)3.3      // volts
```

Application Parameters

Parameter Name	Description	Units
PWM_FREQ	PWM frequency	Hz
DEADTIME_SEC	Dead time	Seconds
MAX_BOARD_CURRENT	Maximum inverter DC bus current through the board that can be measured without saturating the ADC	A
MAX_MOTOR_CURRENT	Maximum motor phase current	A
MOTOR_PER_PHASE_RESISTANCE	Motor per phase resistance	O
MOTOR_PER_PHASE_INDUCTANCE	Motor per phase inductance	H
MOTOR_BACK_EMF_CONSTANT_Vpeak_Line_Line_KRPM_MECH	Motor Back EMF constant	Vpeak(line-line)/KRPM
NOPOLESPAIRS	Number of motor pole pairs	-
MAX_ADC_COUNT	Full scale ADC count, 2n bit ADC --1 (For 12 bit ADC, 212-1= 4095	-
MAX_ADC_INPUT_VOLTAGE	ADC reference voltage	V
DCBUS_SENSE_TOP_RESISTOR	High-side resistance of DC BUS Sense voltage divider	O
DCBUS_SENSE_BOTTOM_RESISTOR	Low-side resistance of DC BUS Sense voltage divider	O
LOCK_TIME_IN_SEC	Rotor lock time to a forced rotor angle before spinning the motor	Seconds
END_SPEED_RPM	Motor speed in Open Loop mode at which the algorithm switches to closed loop	RPM
RAMP_TIME_IN_SEC	Time to reach open loop end speed (END_SPEED_RPM) during open loop operation	Seconds
Q_CURRENT_REF_OPENLOOP	Q-axis current during open loop operation	A
NOMINAL_SPEED_RPM	Maximum rated speed in constant torque mode (without field weakening)	RPM
MOVING_AVG_WINDOW_SIZE	Total number of current samples used to calculate moving average of the current to obtain ADC current offset = 2MOVING_AVG_WINDOW_SIZE	-
CURRENT_OFFSET_MAX	Maximum limit of the ADC current offset	-
CURRENT_OFFET_MIN	Minimum limit of the ADC current offset	-
CURRENT_OFFSET_INIT	Initial value of the ADC current offset	-
KFILTER_ESDQ	First order low-pass filter coefficient for Ed, Eq estimator parameters	-
KFILTER_VELESTIM	First order low-pass filter coefficient for speed estimation	-
FW_SPEED_RPM	Maximum rated speed in Field Weakening mode	RPM
MAX_FW_NEGATIVE_ID_REF	Maximum applicable negative D-axis current	A
D_CURRCNTR_PTERM	D-axis current controller proportional gain	-
D_CURRCNTR_ITERM	D-axis current controller integral gain	-
D_CURRCNTR_CTERM	D-axis current controller anti-windup gain	-
D_CURRCNTR_OUTMAX	D-axis current controller saturation limit	-
Q_CURRCNTR_PTERM	Q-axis current controller proportional gain	-
Q_CURRCNTR_ITERM	Q-axis current controller integral gain	-
Q_CURRCNTR_CTERM	Q-axis current controller anti-windup gain	-
Q_CURRCNTR_OUTMAX	Q-axis current controller saturation limit	-
SPEEDCNTR_PTERM	Speed controller proportional gain	-
SPEEDCNTR_ITERM	Speed controller integral gain	-

SPEEDCNTR_CTERM	Speed current controller anti-windup gain	-
SPEEDCNTR_OUTMAX	Speed current controller saturation limit	-

Class B Tests

This demonstration runs the following Class B tests during startup:

- Clock Test
- Program Counter Test
- CPU Registers Test
- Flash Test.
- Checkerboard RAM Test
- March B Test
- March C Test

All Class B memory tests in this demonstration are non-destructive.

Faults

In this demonstration, the ON state of LED D2 indicates a Fault detection. This demonstration is equipped to detect the Class B test fail and overcurrent faults. Class B tests are run immediately after the device power-up. If any of the Class B tests fail, it will be indicated by LED D2 turning ON. Also, failure of Class B tests would inhibit the start of the motor.

Once the Class B tests have passed successfully, the motor can be started by pressing the switch ‘S2’. While the motor is spinning, if there is an overcurrent condition detected, this will cause the MCPWM to shutdown and LED D2 will turn ON indicating a Fault.

dualshunt_pll_foc_mclv2_int_opamp

This section provides information on the supported demonstration boards, how to configure the hardware (if needed), and how to run the demonstration.

Description

This demonstration implements Sensorless Field Oriented Control (FOC) of a PMSM motor using a dual shunt configuration, and utilizes internal (on-chip) Op amps for current sensing.

Building the Application

This section identifies the MPLAB X IDE project name and location and lists and describes the available configurations for the demonstration.

Description

To build this project, you must open the `dualshunt_pll_foc_mclv2_int_opamp.X` project in MPLAB X IDE, and then select the desired configuration.

The following tables list and describe the project and supported configurations. The parent folder for these files is `<install-dir>/apps/motor_control/dualshunt_pll_foc_mclv2_int_opamp`.

MPLAB X IDE Project

This table lists the name and location of the MPLAB X IDE project folder for the demonstration.

Project Name	Location
`dualshunt_pll_foc_mclv2_int_opamp.X`	`<install-dir>/apps/motor_control/dualshunt_pll_foc_mclv2_int_opamp/firmware`

MPLAB X IDE Project Configurations

This table lists and describes the supported configurations of the demonstration, which are within `./firmware/src/system_config`.

Project Configuration Name	BSP(s) Used	Description
small_hurst_motor_mclv2_int_opamp	Not applicable.	Demonstrates dual shunt PLL estimator-based FOC of a Small Hurst Motor (DMB0224C10002) using internal op-amps.

Configuring the Hardware

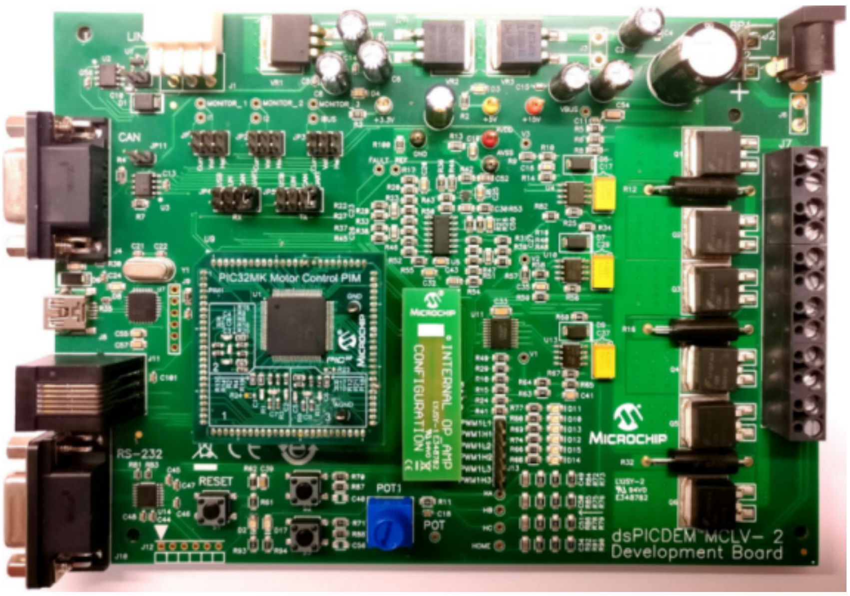
Describes how to configure the supported hardware.

Description

PIC32MK 100-pin Motor Control Plug-in Module (PIM) with the dsPICDEM MCLV-2 Development Board

1. Mount the PIC32MK 100-pin Motor Control PIM on the dsPICDEM MCLV-2 Development Board, as shown in the following figure.
2. Connect the Internal Op amp Configuration Matrix Board to header J14.
3. Connect the three-phase wires coming from P1 header of the small Hurst motor (i.e., Red, White, and Black to M1, M2, and M3 ports, respectively). Leave the green wire from the motor unconnected.
4. Connect the programmer/debugger using the J11 connector.
5. Apply 24V DC at BP1-BP2/J2.
6. This application supports communication with DMCI to monitor/plot any variables used in the application. The communication between the target and the DMCI plug-in can be established using the RS-232 serial port (J10) or mini-USB port (J8). If using the RS-232 serial port, connect JP4 and JP5 to the UART position. If using the mini-USB port, connect JP4 and JP5 to the USB position.

PIC32MK MC PIM with dsPICDEM™ MCLV-2 the Internal Op amp Configuration



Running the Demonstration

Provides instructions on how to build and run the demonstration.

Description

For instructions, refer to the Running the Demonstration section for the dualshunt_pll_foc_mclv2_ext_opamp demonstration.

integrated_pfc_foc_mhcv3_ext_opamp

This section provides information on the supported demonstration boards, how to configure the hardware, and how to run the demonstration.

Description

This demonstration implements integrated application of average current mode PFC and Sensorless Field Oriented Control (FOC) of a PMSM motor using a dual shunt configuration, and utilizes external (off-chip) Op amps for current sensing.

Building the Application

This section identifies the MPLAB X IDE project name and location and lists and describes the available configurations for the demonstration.

Description

To build this project, you must open the `integrated_pfc_foc_mchv3_ext_opamp.X` project in MPLAB X IDE, and then select the desired configuration.

The following tables list and describe the project and supported configurations. The parent folder for these files is:

`<install-dir>/apps/motor_control/integrated_pfc_foc_mchv3_ext_opamp`.

MPLAB X IDE Project

The following table lists the name and location of the MPLAB X IDE project folder for the demonstration.

Project Name	Location
`integrated_pfc_foc_mchv3_ext_opamp.X`	`<install-dir>/apps/motor_control/integrated_pfc_foc_mchv3_ext_opamp/firmware`

MPLAB X IDE Project Configurations

The following table lists and describes the supported configurations of the demonstration, which are within `./firmware/src/system_config`.

Project Configuration Name	BSP(s) Used	Description
motor_80_mchv3_ext_opamp	N/A	Demonstrates integrated application of average current mode PFC and sensorless FOC of PMSM motor using external op-amps.

Configuring the Hardware

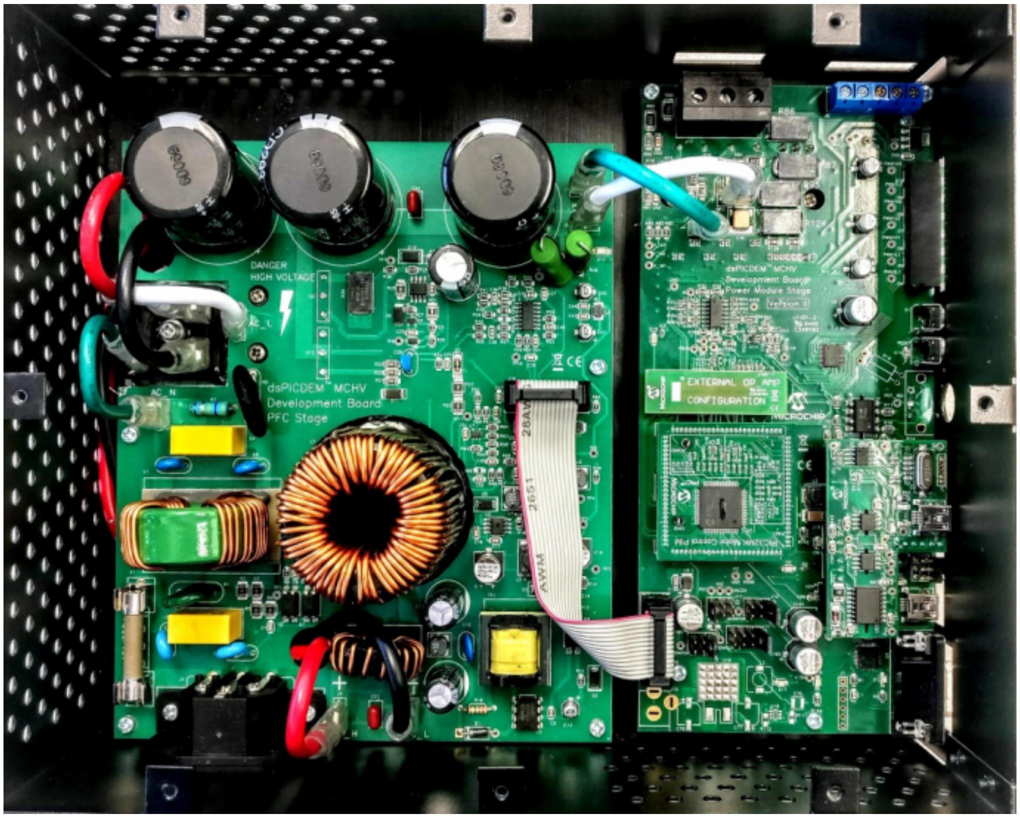
This section describes how to configure the supported hardware.

Description

PIC32MK 100-pin Motor Control PIM with the dsPICDEM MCHV-3 Development Board

1. Mount the PIC32MK 100-pin Motor Control PIM on the dsPICDEM MCHV-3 Development Board, as shown in the following figure.
2. Connect the External Op amp Configuration Matrix Board to header J4.
3. Connect the three-phase wires of the PMSM motor to M1, M2 and M3 ports.
4. Connect the on board programmer and debugger to J20 using USB Mini B cable.
5. Connect the AC mains using J1.
6. This application supports communication with the X2C-Scope plug-in to monitor and plot any variables used in the application. The communication between the target and the X2C-Scope plug-in can be established using the RS-232 serial port (J8) or mini-USB port (J6). Refer to the jumper positions marked on the development board chassis to select between the RS-232 serial port (J8) and the mini-USB port (J6). Refer to Using the X2C-Scope Plug-in for details on how to use the X2C-Scope in this demonstration.

Note:	Please refer to dsPICDEM MCHV-3 Development Board User's Guide for safety and operating details of the development board.



Running the Demonstration

This section provides instructions on how to build and run the demonstration.

Description

Do the following to run the demonstration:

1. Compile and build the project.
2. Program the target device.
3. Press the ‘S1’ switch to start the PFC. Once the bus voltage has reached the set point, the motor will begin spinning.
4. Vary the potentiometer, POT1, to change the motor speed.
5. Press the ‘Reset’ switch to stop the motor.

Modifying the System Parameters

The system parameters, such as motor coefficients, maximum current, etc., can be modified in the header file `mc_app.h`, as shown in the following code example

```
//================================================================
// Following parameters for MCHV-3 board (External Opamp Configuration)
// Gain of opamp = 10.06
// shunt resistor = 0.01 ohms
// DC offset = 1.65V
// max current = x
// (x * 0.01 * 10.06) + 1.65V = 3.3V
// x = 16.4 Amps
#define      MAX_BOARD_CURRENT                          (float)(16.4)
#define      MAX_MOTOR_CURRENT                          (float)(8)
#define      MAX_MOTOR_CURRENT_SQUARED
(float)((float)MAX_MOTOR_CURRENT*(float)MAX_MOTOR_CURRENT)

#define      ADC_CURRENT_SCALE                          (float)(MAX_BOARD_CURRENT/(float)2048)

#define      CURRENT_LIMIT_CMP_REF
(int)(((float)2048*(MAX_MOTOR_CURRENT/MAX_BOARD_CURRENT))+VREF_DAC_VALUE)

// Resistance in Ohms
#define      MOTOR_PER_PHASE_RESISTANCE                 ((float)16)


// Inductance in Henrys
#define      MOTOR_PER_PHASE_INDUCTANCE                 ((float)0.022)

#define      MOTOR_PER_PHASE_INDUCTANCE_DIV_2_PI
((float)(MOTOR_PER_PHASE_INDUCTANCE/(2*M_PI)))

// Back EMF Constant in Vpeak/KRPM
#define      MOTOR_BACK_EMF_CONSTANT_Vpeak_Line_Line_KRPM_MECH   (float)75

#define      NOPOLESPAIRS                               2
#define      MAX_ADC_COUNT                              (float)4095     // for 12-bit ADC
#define      MAX_ADC_INPUT_VOLTAGE                  (float)3.3      // volts
```

Application Parameters

Parameter Name	Description	Units
PWM_FREQ	PWM Frequency	Hz
DEADTIME_SEC	Dead Time	Seconds
MAX_BOARD_CURRENT	Maximum Inverter DC bus current through the board that can be measured without saturating the ADC	A
MAX_MOTOR_CURRENT	Maximum Motor Phase Current	A
MOTOR_PER_PHASE_RESISTANCE	Motor per phase resistance	O
MOTOR_PER_PHASE_INDUCTANCE	Motor per phase inductance	H
MOTOR_BACK_EMF_CONSTANT_Vpeak_Line_Line_KRPM_MECH	Motor Back EMF constant	Vpeak(line-line)/KRPM

NOPOLESPAIRS	Number of Motor Pole Pairs	-
MAX_ADC_COUNT	Full Scale ADC Count, 2n bit ADC -1 (For 12 bit ADC, 212-1= 4095	-
MAX_ADC_INPUT_VOLTAGE	ADC Reference voltage	V
DCBUS_SENSE_TOP_RESISTOR	High side resistance of DC BUS Sense voltage divider	O
DCBUS_SENSE_BOTTOM_RESISTOR	Low side resistance of DC BUS Sense voltage divider	O
LOCK_TIME_IN_SEC	Rotor lock time to a forced rotor angle before spinning the motor	Seconds
END_SPEED_RPM	Motor speed in Open Loop mode at which the algorithm switches to closed loop	RPM
RAMP_TIME_IN_SEC	Time to reach Open Loop End Speed (END_SPEED_RPM) during open loop operation	Seconds
Q_CURRENT_REF_OPENLOOP	Q axis current during Open Loop operation	A
NOMINAL_SPEED_RPM	Maximum rated speed in Constant Torque mode (without field weakening)	RPM
MOVING_AVG_WINDOW_SIZE	Total number of current samples used to calculate moving average of the current to obtain ADC Current Offset = 2MOVING_AVG_WINDOW_SIZE	-
CURRENT_OFFSET_MAX	Maximum limit of the ADC Current Offset	-
CURRENT_OFFSET_MIN	Minimum limit of the ADC Current Offset	-
CURRENT_OFFSET_INIT	Initial value of the ADC Current Offset	-
KFILTER_ESDQ	First order low pass filter coefficient for Ed, Eq estimator parameters	-
KFILTER_VELESTIM	First order low pass filter coefficient for speed estimation	-
FW_SPEED_RPM	Maximum rated speed in Field Weakening mode	RPM
MAX_FW_NEGATIVE_ID_REF	Maximum applicable negative D-axis current	A
D_CURRCNTR_PTERM	D-axis Current Controller Proportional Gain	-
D_CURRCNTR_ITERM	D-axis Current Controller Integral Gain	-
D_CURRCNTR_CTERM	D-axis Current Controller Anti-Windup Gain	-
D_CURRCNTR_OUTMAX	D-axis Current Controller Saturation Limit	-
Q_CURRCNTR_PTERM	Q-axis Current Controller Proportional Gain	-
Q_CURRCNTR_ITERM	Q-axis Current Controller Integral Gain	-
Q_CURRCNTR_CTERM	Q-axis Current Controller Anti-Windup Gain	-
Q_CURRCNTR_OUTMAX	Q-axis Current Controller Saturation Limit	-
SPEEDCNTR_PTERM	Speed Controller Proportional Gain	-
SPEEDCNTR_ITERM	Speed Controller Integral Gain	-
SPEEDCNTR_CTERM	Speed Current Controller Anti-Windup Gain	-
SPEEDCNTR_OUTMAX	Speed Current Controller Saturation Limit	-

Class B Tests

This demonstration runs the following Class B tests during startup:

- Clock Test
- Program Counter Test
- CPU Registers Test
- Flash Test
- Checkerboard RAM Test
- March B Test
- March C Test

All Class B memory tests in this demonstration are non-destructive.

Faults

In this demonstration, the ON state of LED D2 indicates a Fault detection. This demonstration is equipped to detect a Class B test fail, and PFC or Motor Control faults. Class B tests are run immediately after the device power-up. If any of the Class B tests fail, it will be indicated by LED D2 turning ON. Also, the failure of Class B tests would inhibit the start of the PFC or motor. Once the Class B tests have passed successfully, the motor can be started by pressing the switch 'S1'. While the PFC and motor are operational, if there is an Overcurrent or over-voltage condition detected on the PFC or Motor, this will cause the MCPWM to shutdown and the LED D2 will turn ON indicating a Fault.

integrated_pfc_foc_mchv3_int_opamp

This section provides information on the supported demonstration boards, how to configure the hardware, and how to run the demonstration.

Description

This demonstration implements integrated application of average current mode PFC and Sensorless Field Oriented Control (FOC) of a PMSM using a dual shunt configuration, and utilizes internal (on-chip) Op amps for current sensing.

Building the Application

This section identifies the MPLAB X IDE project name and location. It also lists and describes the available configurations for the demonstration.

Description

To build this project, you must open the `integrated_pfc_foc_mchv3_int_opamp.X` project in MPLAB X IDE, and then select the desired configuration.

The following tables list and describe the project and supported configurations. The parent folder for these files is:

`<install-dir>/apps/motor_control/integrated_pfc_foc_mchv3_int_opamp`

MPLAB X IDE Project

This table lists the name and location of the MPLAB X IDE project folder for the demonstration.

Project Name	Location
`integrated_pfc_foc_mchv3_int_opamp.X`	`<install-dir>/apps/motor_control/integrated_pfc_foc_mchv3_int_opamp/firmware`

MPLAB X IDE Project Configurations

This table lists and describes the supported configurations of the demonstration, which are within `./firmware/src/system_config`

Project Configuration Name	BSP (s) Used	Description
motor_80_mchv3_int_opamp	N/A	Demonstrates integrated application of average current mode PFC and sensorless FOC of PMSM motor using internal op-amps.

Configuring the Hardware

This section describes how to configure the supported hardware.

Description

PIC32MK 100-pin Motor Control PIM with the dsPICDEM MCHV-3 Development Board

1. Mount the PIC32MK 100-pin Motor Control PIM on the dsPICDEM MCHV-3 Development Board, as shown in the following Figure.
2. Connect the Internal Op amp Configuration Matrix Board to header J4.
3. Connect the three-phase wires of the PMSM motor to M1, M2 and M3 ports.
4. Connect to the on board programmer and debugger to J20 using USB Mini B cable.
5. Connect the AC mains using J1.
6. This application supports communication with the X2C-Scope plug-in to monitor and plot any variables used in the application. The communication between the target and the X2C-Scope plug-in can be established using the RS-232 serial port (J8) or mini-USB port (J6). Refer to the jumper positions marked on the development board chassis to select between the RS-232 serial port (J8) and the mini-USB port (J6). Refer to Using the X2C-Scope Plug-in for details on how to use the X2C-Scope in this demonstration.

Note:	Please refer to dsPICDEM MCHV-3 Development Board User's Guide for safety and operating details of the development board.

Running the Demonstration

This section provides instructions on how to build and run the demonstration.

Description

For instructions, refer to the Running the Demonstration section for the integrated_pfc_foc_mchv3_int_opamp demonstration.

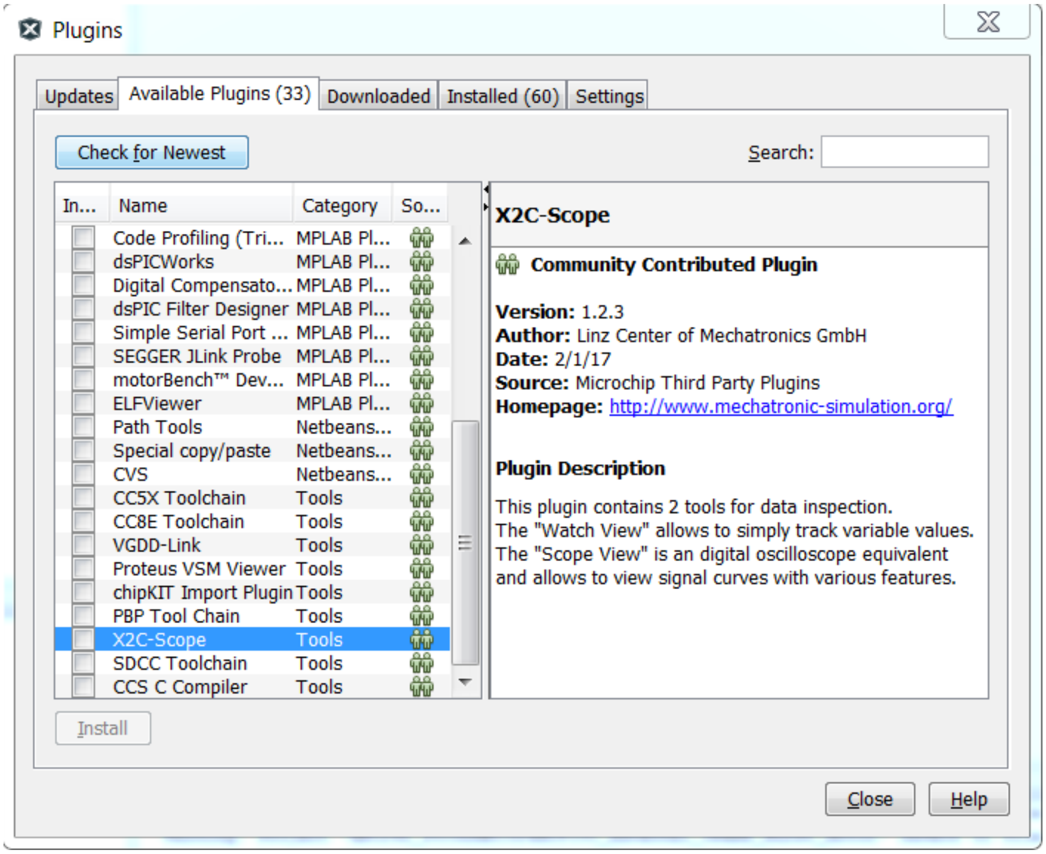
X2C-Scope Plug-in

This topic describe how to use the X2C-Scope MPLAB X IDE plug-in.

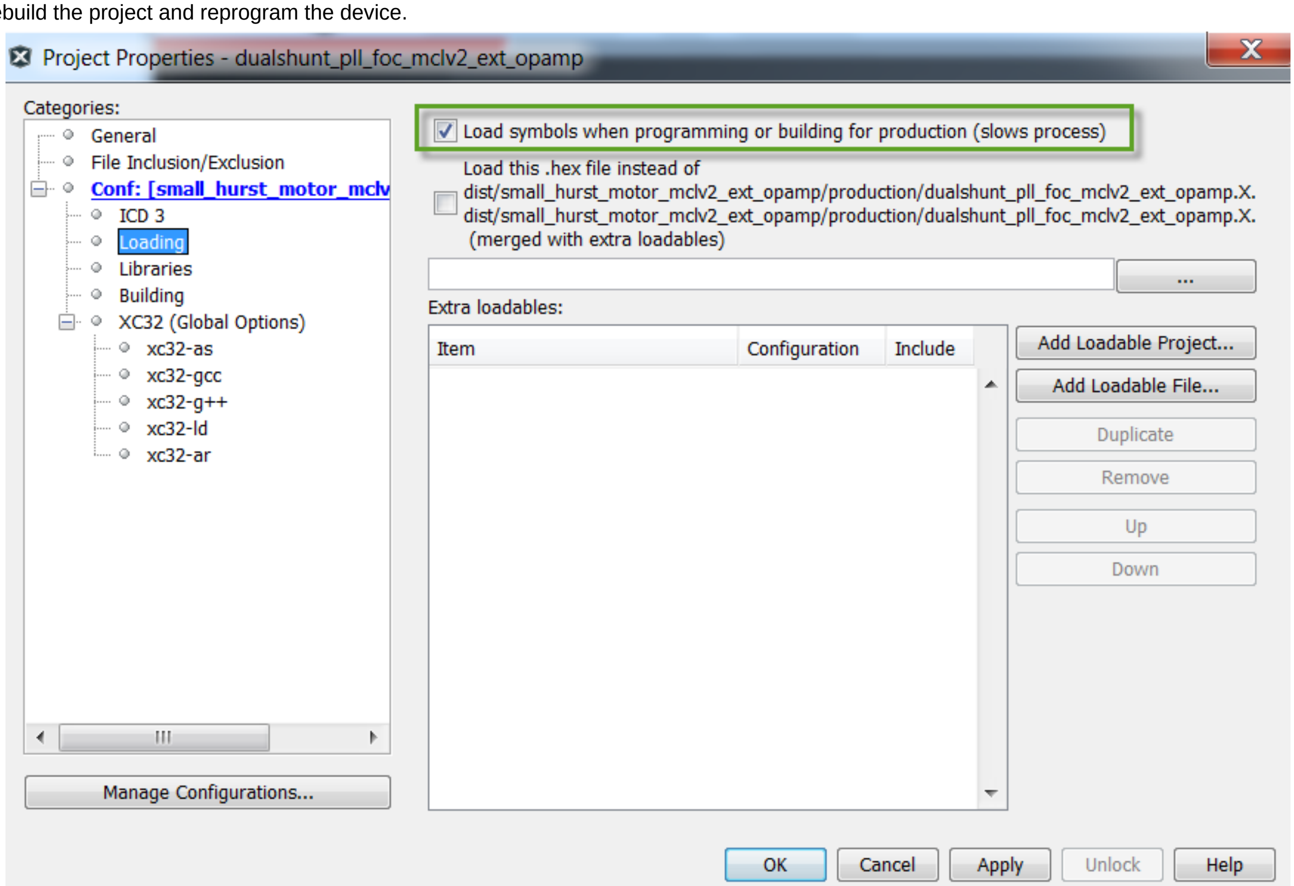
Description

The MPLAB X IDE enables use of the X2C-Scope plug-in to read, write, and plot global variables in real time. In this demonstration, the X2C-Scope communicates with the target using UART Channel 2. The baud rate of the UART communication can be modified using the MPLAB Harmony Configurator (MHC) (the default value is 38400 bps).

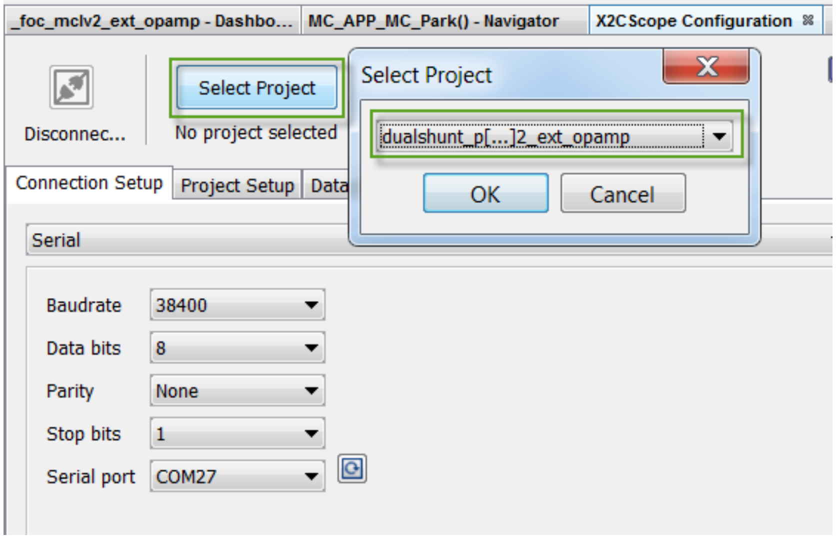
1. If not already installed, in MPLAB X IDE, select *Tools > Plugins > Available Plugins > Select X2C-Scope> Install*.
2. Restart MPLAB X IDE to complete the plug-in installation.



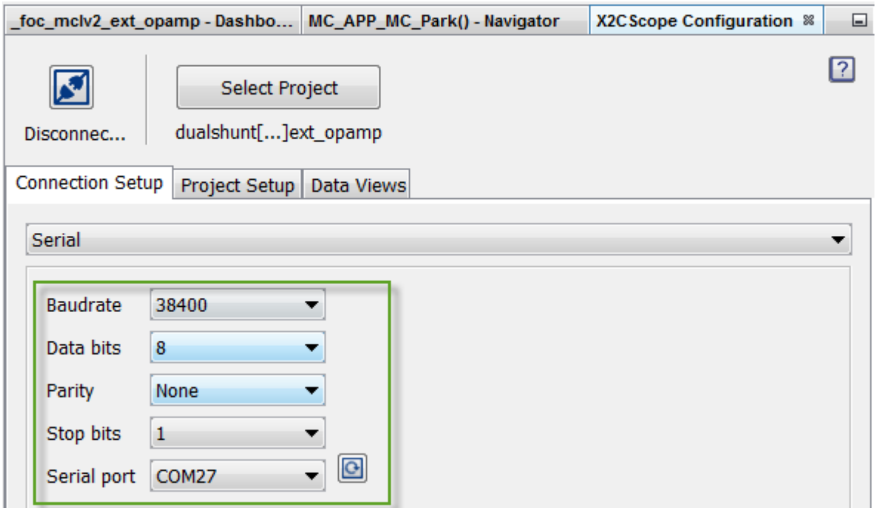
3. Open the X2C-Scope by selecting *Tools > Embedded > X2C-Scope*.
4. Ensure the symbols are loaded during the project build by selecting *Project Properties > Selected Configuration > Loading*, and ensure that the check box for *Load Symbols when programming or building for production (slows process)* is selected. If this check box was not selected, rebuild the project and reprogram the device.



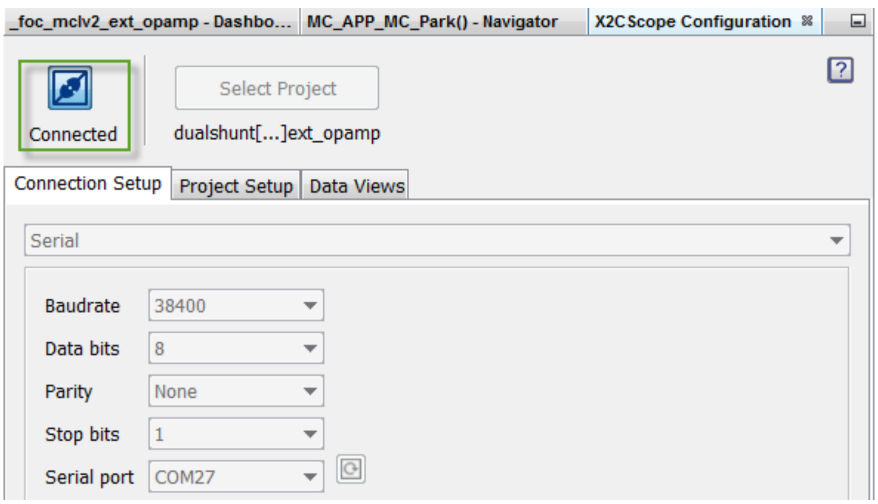
5. In the *X2C-Scope Configuration* tab, select the project.



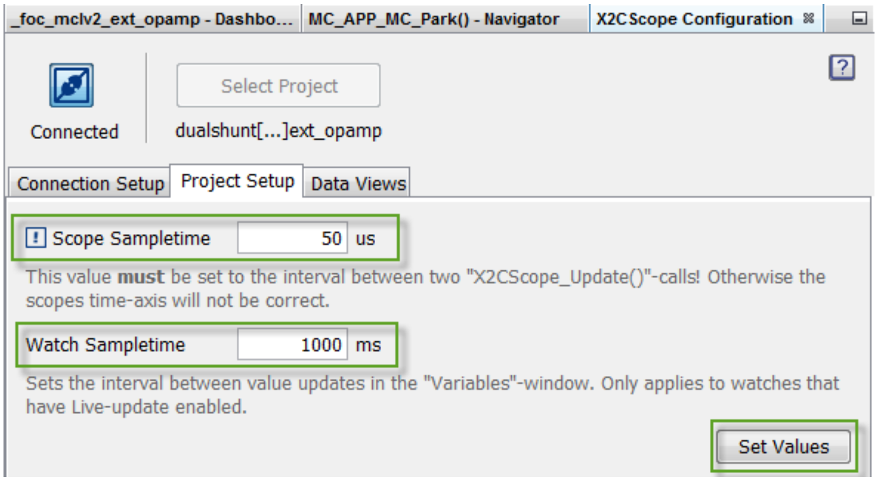
6. In Connection Setup, set the *Baud Rate* to '38400', *Data bits* to '8', *Parity* to 'None', *Stop Bits* to '1', and then select the associated COM port.



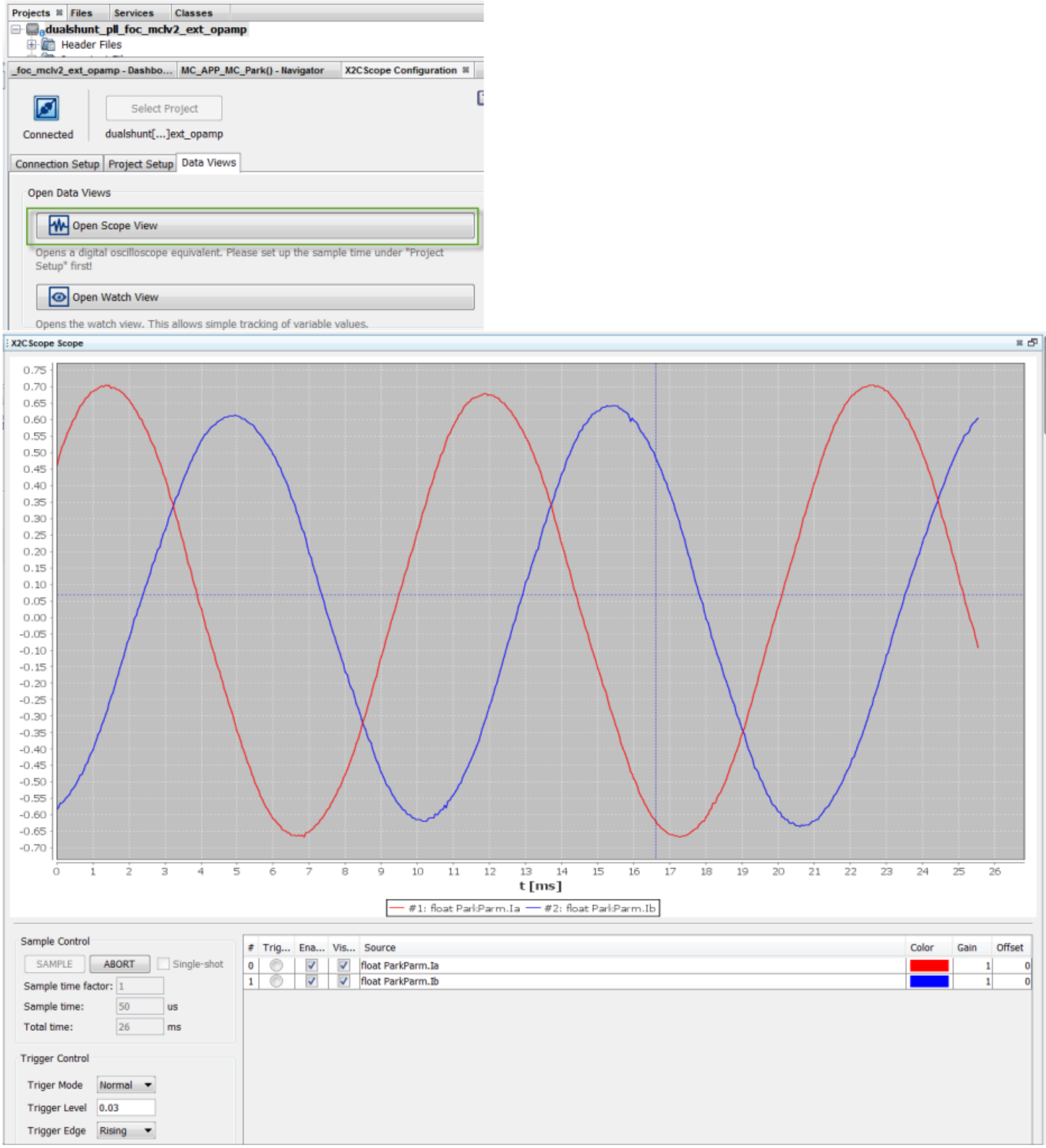
7. Connect to the target by clicking **Connect** (or **Disconnect**).



8. In *Project Setup*, Set the *Scope Sampletime* to '50 us.' In this demonstration, the `X2CScope_Update` function is called from the motor control ISR, which executes every 50 µs. Set the *Watch Sample time* to a value at which you want to update the watch window variables (the default is 1000 ms). Click **Set Values** after setting the *Scope Sampletime* and *Watch Sampletime*.

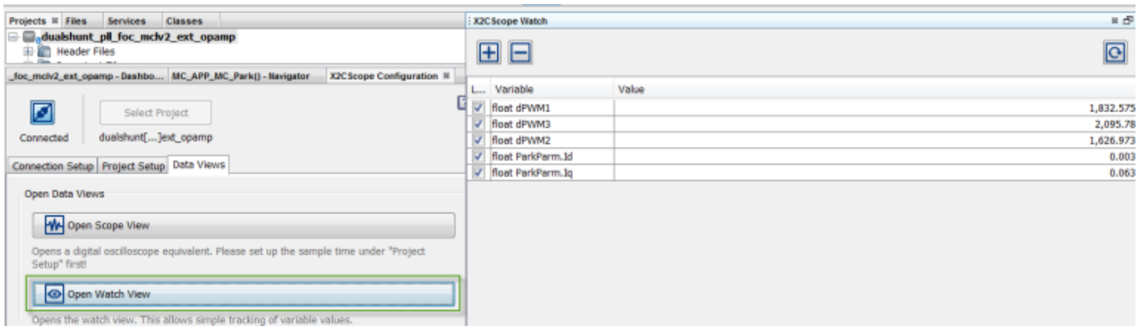


9. In *Data Views*, select *Open Scope View* to plot any two global variables in run-time.



Note:	The free version of X2C-Scope allows up to two global variables to be plotted simultaneously. The professional Version of X2C-Scope allows up to seven global variables to be plotted simultaneously.

11. Under *Data Views*, select *Open Watch View* to read or write to any global variables in run time.


Projects
Files
Services
Classes
dualshunt_pll_foc_mclv2_ext_opamp
Header Files
_foc_mclv2_ext_opamp - Dashbo...
MC_APP_MC_Park() - Navigator
X2CScope Configuration
Select Project
Connected
dualshunt[....]ext_opamp
Connection Setup
Project Setup
Data Views
Open Data Views
Open Scope View
Opens a digital oscilloscope equivalent. Please set up the sample time under "Project Setup" first!
Open Watch View
Opens the watch view. This allows simple tracking of variable values.
X2CScope Watch
Variable
Value
float dPWM1
1,832.575
float dPWM3
2,095.78
float dPWM2
1,626.973
float ParkParm.Id
0.003
float ParkParm.Iq
0.063

Index

A

B

C

D

I

M

R

V

X