

---

# AN1292 Demonstration ReadMe for dsPICDEM™ Low Voltage Motor Control Bundle with the dsPIC33EV256GM106 Internal Op Amp PIM (MPLAB X)

---

## 1.1 INTRODUCTION

This document describes the setup requirements for running the Sensorless FOC algorithm with a PLL Estimator, which is referenced in AN1292 “*Sensorless Field Oriented Control (FOC) for a Permanent Magnet Synchronous Motor (PMSM) Using a PLL Estimator and Field Weakening (FW)*” using a Low Voltage Motor Control Bundle.

## 1.2 SUGGESTED DEMONSTRATION REQUIREMENTS

MPLAB and XC16 versions used:

- MPLABX version 2.26 (or later)
- XC16 version 1.23 (or later)

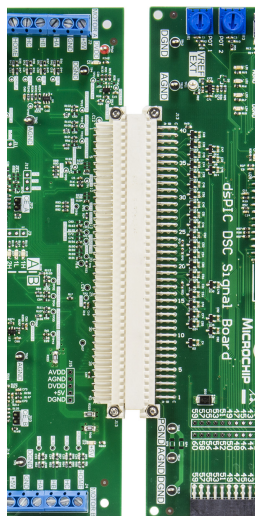
Hardware used with part numbers, available from [www.microchipdirect.com](http://www.microchipdirect.com):

- dsPICDEM Low Voltage Motor Control Bundle (DV330100)
- dsPIC33EV256GM106 Internal Op amp PIM (MA330036)
- 24V Power supply (AC002013)
- 24V Hurst motor (AC300020)

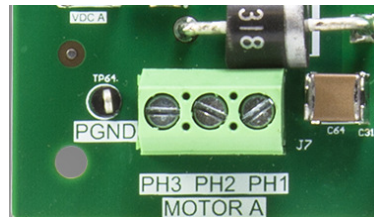
## 1.3 HARDWARE SETUP

The following hardware setup allows the Sensorless FOC algorithm to run on the dsPICDEM Low Voltage Motor Control Bundle.

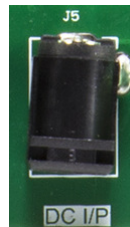
1. Place the Motor control 10-24V driver board (Dual/Single) and the dsPIC DSC signal board on a sturdy insulated platform.
2. Connect dsPIC DSC Signal Board and Motor control 10-24V Driver Board (Dual/Single) together using Connector J13 (White Connector).



3. Make sure that the dsPIC33EV256GM106 Plug in Module is mounted in the U2 socket of dsPIC DSC signal board.
4. Ensure Jumper J15 on dsPIC DSC Signal Board is populated
5. Connect Motor A (three-phase, 10 pole, 24V Hurst motor AC300020) wires to J7 on Motor Control 10-24V Driver Board. As this is a sensorless algorithm, the motor phase wires (Red, Black, and White) can be connected to PH1, PH2, and PH3 in any order. . The Green wire does not have internal connection in the motor, so it can be left unconnected.

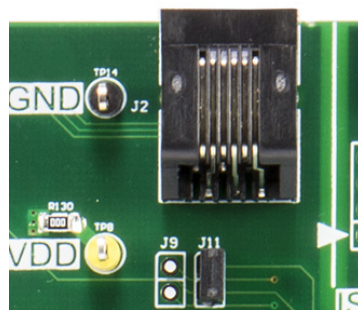


6. Connect the 24V power supply (AC002013) to the Motor Control 10-24 V Driver Board (Dual/Single) connector J5.

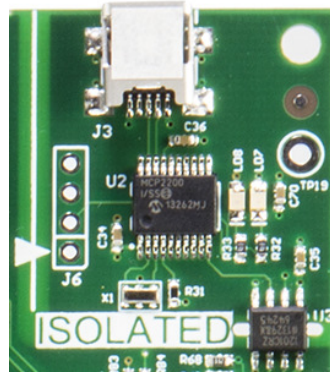


**Note: Power Supply connected to Motor Control 10-24 V Driver Board(Dual/Single) can supply power to both boards. So power supply connection to dsPIC DSC Signal Board is optional**

7. Connect the programmer/debugger to the J2 connector on dsPIC DSC Signal Board.



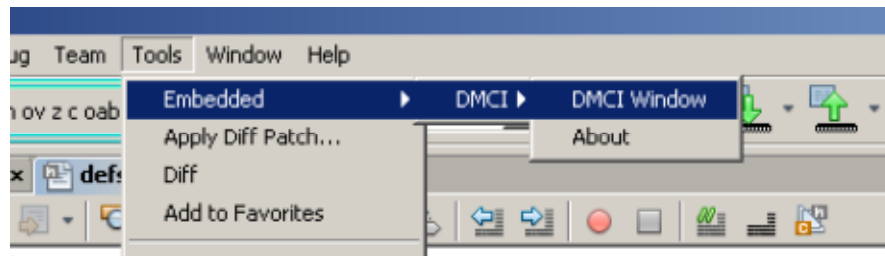
8. For enhanced demonstration, the application requires the Real-Time Data Monitor (RTDM) along with DMCI (Data Monitor Control Interface) Plug-in available in MPLABX IDE.
  - Users can connect a mini-USB cable from their computer to the J3 connector on the dsPIC DSC Signal Board.
  - Ensure Jumper J11 on dsPIC DSC Signal Board is populated, to establish this communication link



Notice that when the development board is powered and connected to the USB host for the first time, the driver needs to be installed on the host for proper operation. When the USB driver is installed, a new COM port should show up in Windows device hardware manager. This should be the COM port used for Enhanced Demonstration.

USB driver can be found at <http://www.microchip.com/MCP2200>

- Also ensure DMCI plug-in available in MPLABX IDE is installed for debugging applications in real-time. Steps to install the plugin can be found in MPLABX IDE Help MPLABX IDE->Additional Tasks -> Add Plug-In Tools-> Add Plug-Ins
- After successful installation,DMCI plug-in is available under Tools->Embedded->DMCI



- In MPLAB X IDE, you will need to do the following:
  - This is applicable for MPLAB X IDE version 2.10 or later. Check the checkbox "Load symbols when programming or building for production (slows process)" under the "Loading" category in the Project Properties dialog (right click on the project name and select "Properties"). This will load debug symbols for building and programming.
  - Uncheck the checkbox "Maintain active connection to hardware tool" under Tools>Options, Embedded button, Generic Settings tab.
- For additional information on DMCI ,refer to MPLABX IDE Plug-in-Tools help for 'Data Monitor and Control Interface'

## 1.4 SOFTWARE SETUP AND RUN

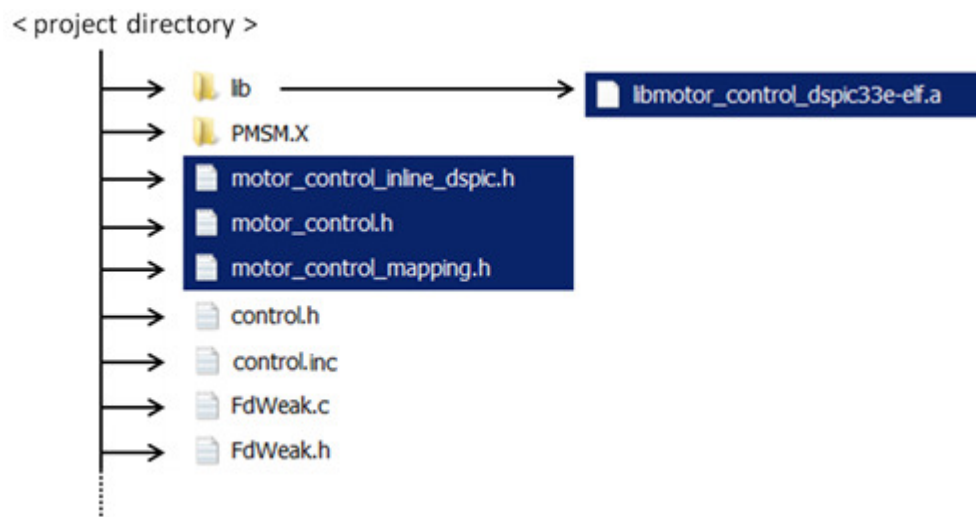
### 1.4.1 Basic Demonstration

This demonstration consists of running the motor using a push button and varying the speed with a potentiometer. The software, which is available for download from the Microchip website, is already configured for enabling the basic demonstration.

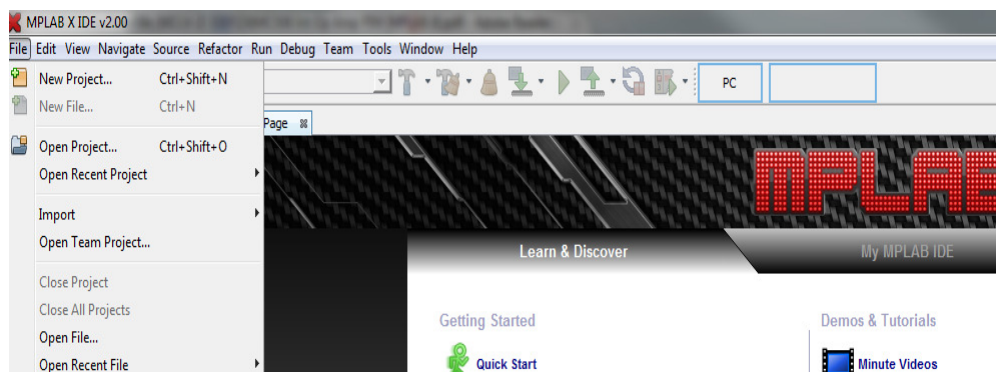
**NOTE: Code uses Motor Control Library for some of the functions. To run the code the motor control library files have to be included in the project. Details on usage of motor control library can be found on [www.microchip.com/motor](http://www.microchip.com/motor)**

**Copy the Motor Control library header files into the MPLABX project directory**

- **Include a copy of the Motor Control library archive file (libmotor\_control\_dspic33e-elf.a) in the “lib” folder located in the MPLABX project directory.**

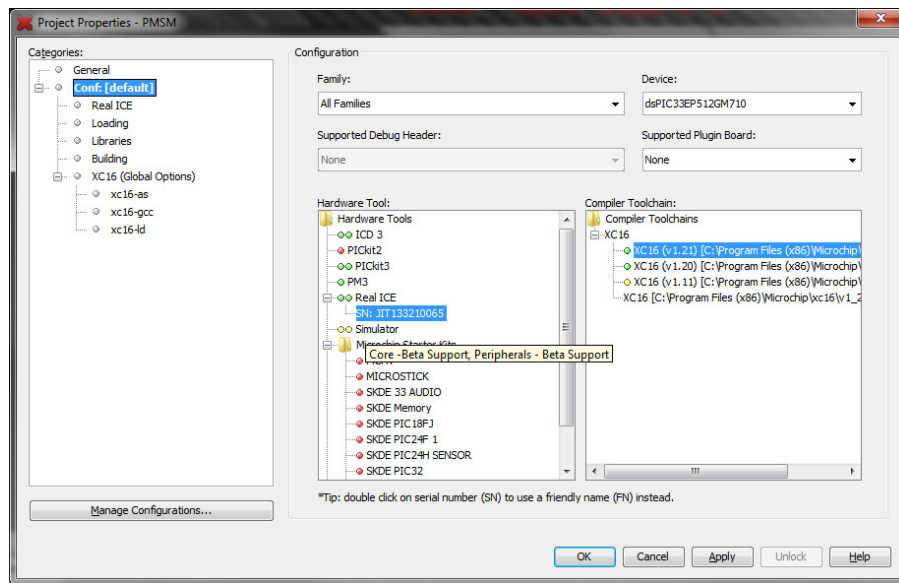


1. Start MPLABX IDE and open the PMSM.X workspace.



2. Right click on PMSM.X project on the left tab called “Project”, and select “Properties”. On the “conf” page you can select the programmer/debugger and the compiler tool chain. In this particular case, REAL ICE™ is the selected programmer and XC16 is the selected toolchain for building the project.

# Demonstration ReadMe



3. Make sure that `RTDM` are not defined in the `user_parms.h` file. Refer to Enhanced Demonstration Using Real-Time Data Monitor (R) if the RTDM demonstration is required.
4. In the `user_parms.h` file, ensure that `BIDIRECTIONAL_SPEED`, `TUNING`, `OPEN_LOOP_FUNCTIONING`,

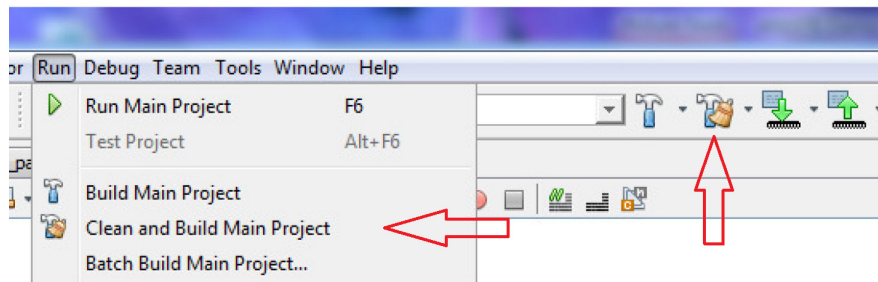
```

//***** Real Time Data M
// This defin
// to handle
// informatio

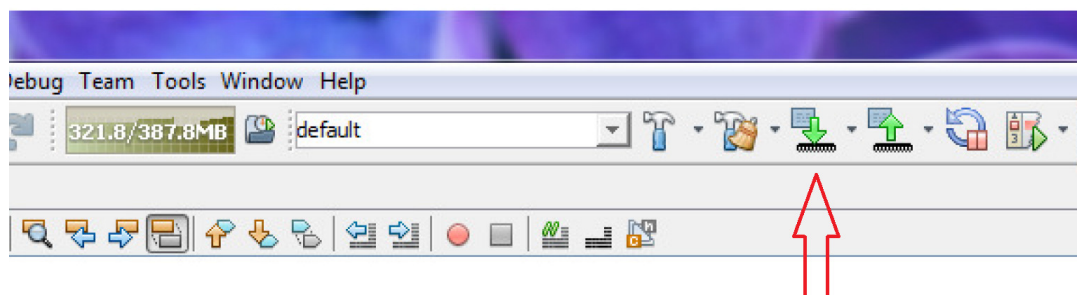
```

and `TORQUE_MODE` are not defined.

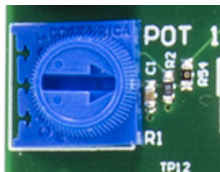
5. Build the code by selecting the “Clean and Build Project” button found either on the toolbar or in the “Run” Menu.



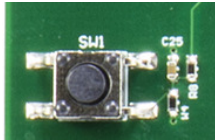
6. After a successful build, download the code to the target device on the dsPIC DSC signal Board by selecting the “Make and Program the device main project” button on the toolbar.



7. Turn and set the Potentiometer POT1 on dsPIC DSC Signal Board to the middle



8. Press SW1 button on dsPIC DSC Signal Board. The 'MOTOR A' should start spinning smoothly in one direction.



9. Vary the speed of 'MOTOR A' using the potentiometer1 (labeled POT 1) on dsPIC DSC Signal Board.
10. To double the speed of the 'MOTOR A', press SW2 on dsPIC DSC Signal Board. Pressing SW2 again will reduce the speed of the motor by 50%.



11. Press SW1 on dsPIC DSC Signal Board to stop the 'MOTOR A'



# Demonstration ReadMe

## 1.4.2 Enhanced Demonstration Using Real-Time Data Monitor (RTDM) and Dynamic Monitor and Control Interface (DMCI)

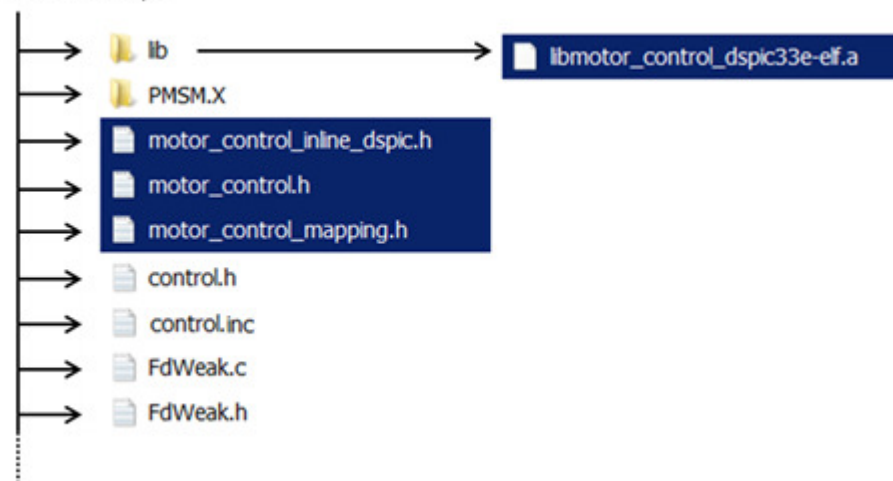
Make sure you have the correct hardware setup as previously described in [Section 1.3 “Hardware Setup”](#)

**NOTE:** Code uses Motor Control Library for some of the functions. To run the code the **motor control** library files have to be included in the project. Details on usage of motor control library can be found on [www.microchip.com/motor](http://www.microchip.com/motor)

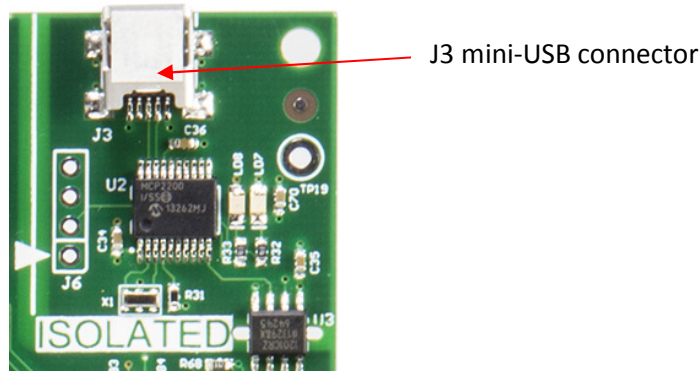
Copy the Motor Control library header files into the MPLABX project directory

- Include a copy of the Motor Control library archive file (libmotor\_control\_dspic33e-elf.a) in the “lib” folder located in the MPLABX project directory.

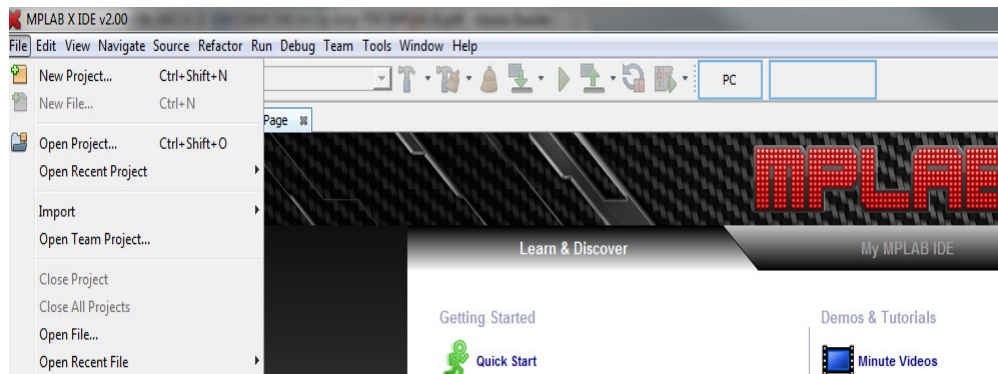
< project directory >



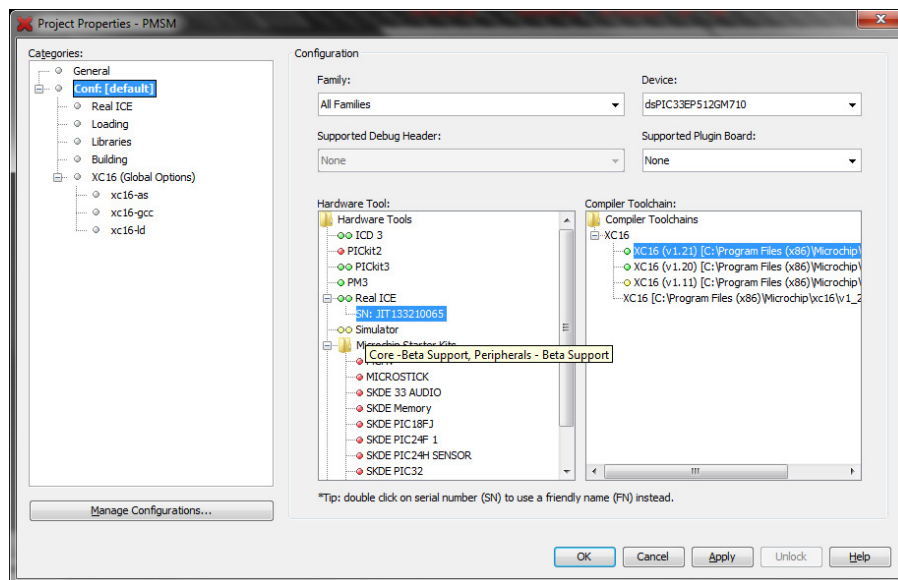
1. In order to utilize RTDM communication for this demonstration, a mini-USB connection is required. Connect a mini-USB cable from your computer to the J3 connector on the dsPIC DSC Signal Board. Also ensure Jumper J11 on dsPIC DSC Signal Board is populated.



2. Start MPLABX IDE and open the PMSM.X workspace.



3. Right click on PMSM.X project on the left tab called "Project", and select "Properties". On the "conf" page you can select the programmer/debugger and the compiler tool chain. In this particular case, REAL ICE™ is the selected programmer and XC16 is the selected toolchain for building the project.



4. Make sure that RTDM are defined in the user\_parms.h file. If this is not defined the DMCI data between the target and the host will not be exchanged.

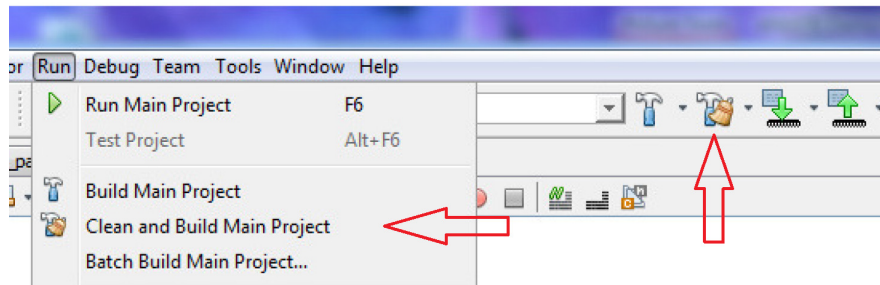
```
#define RTDM // This defines
           // to handle R
           // information
```

5. In the user\_parms.h file, ensure that BIDIRECTIONAL\_SPEED, TUNING, OPEN\_LOOP\_FUNCTIONING, and TORQUE\_MODE are not defined.

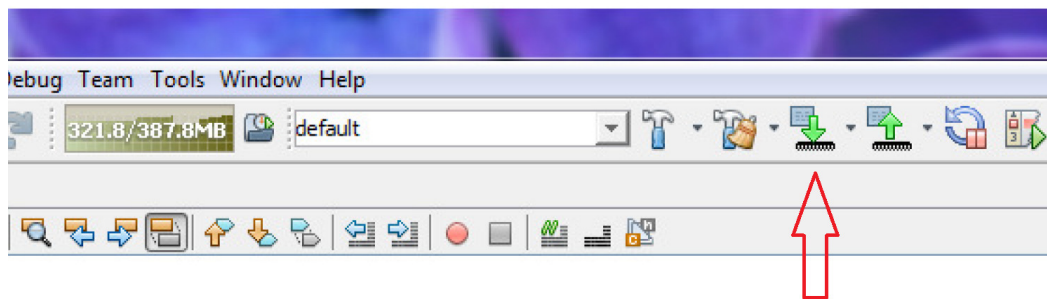


## Demonstration ReadMe

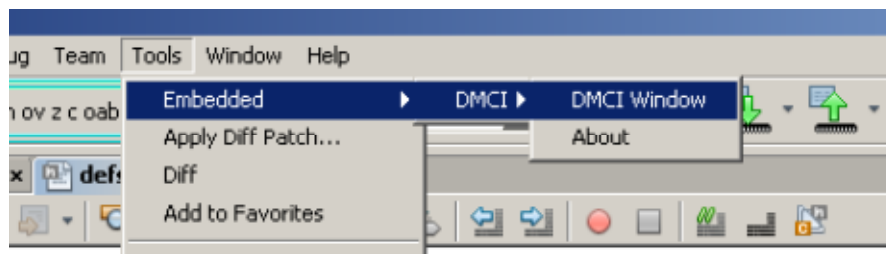
6. Build the code by selecting the “Clean and Build Project” button found either on the toolbar or in the “Run” Menu.



7. After a successful build, download the code to the target device on the dsPIC DSC signal Board by selecting the “Make and Program the device main project” button on the toolbar.



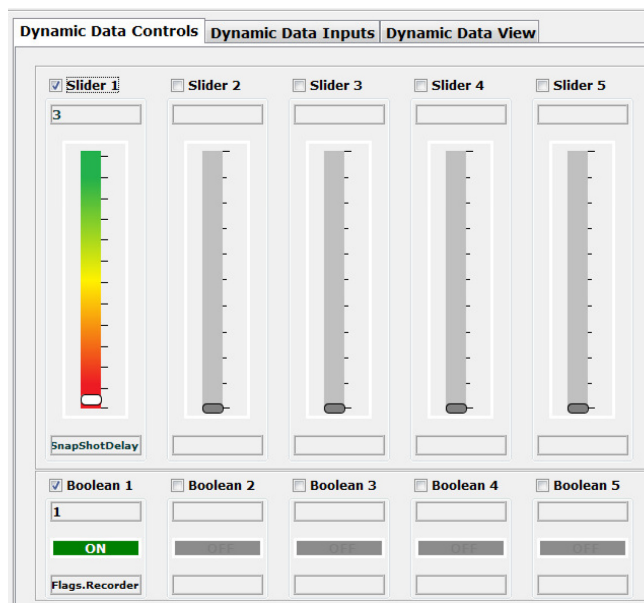
8. Follow steps 8-12 in [Section “1.4.1 Basic Demonstration”](#), for 'MOTOR A' control such as start, stop and vary the motor speed etc
9. Open the DMCI window by selecting Tools>Embedded>DMCI>DMCI Window.



10. Click the Load Profile icon, and from the same folder where your project resides, load the DEMO.xml file in PMSM.X folder, which contains a previously configured profile



11. The DMCI window appears as follows:

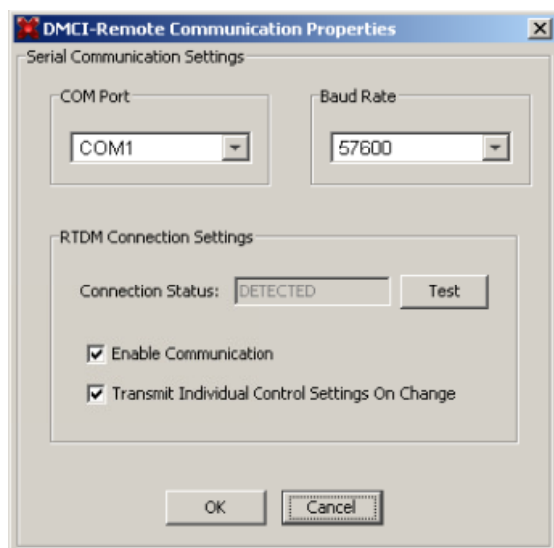


Please consult the *"Real-Time Data Monitor User's Guide"* (DS70567) for additional settings needed for a RTDM connection. This document explains the steps needed for the proper communication settings between the Host and Embedded side.

12. Select Serial Settings to connect RTDM with your computer..



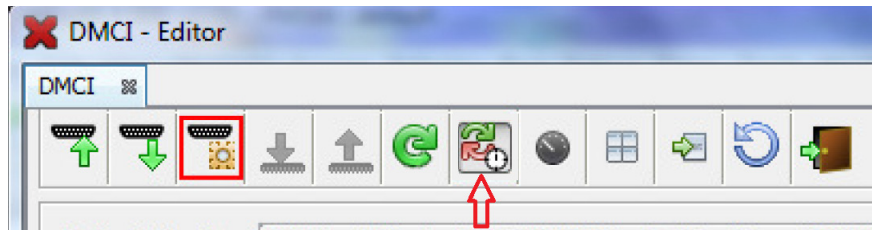
13. Remote Communication needs to be established, as indicated in the following figure (the communication baud rate should be set to 57600, while the COM port used depends on your particular settings).



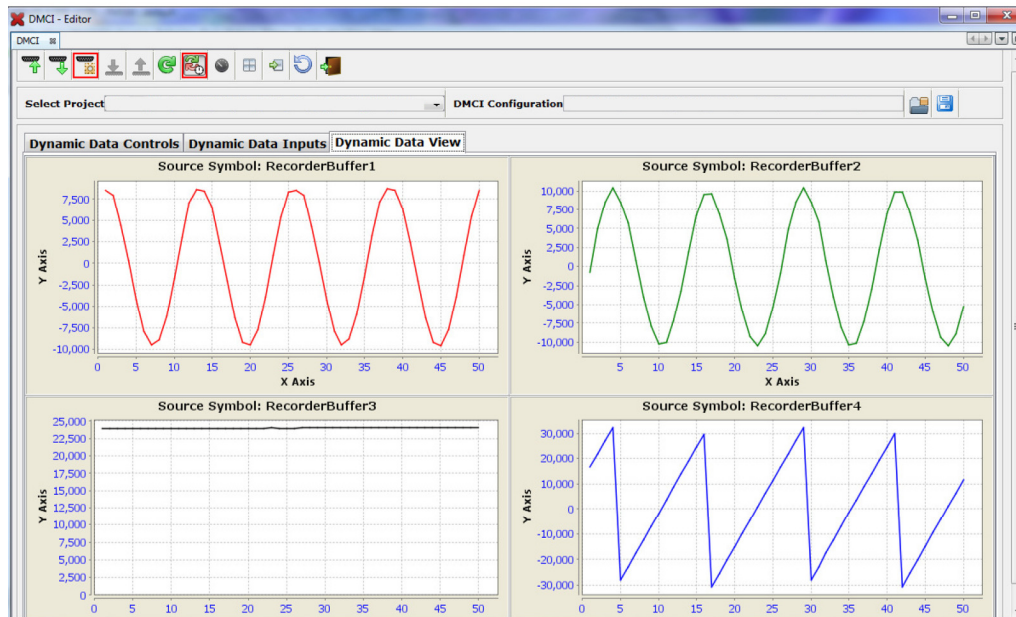
14. Once communication is detected, make sure the **Enable Communication** box is checked and click **OK**.

# Demonstration ReadMe

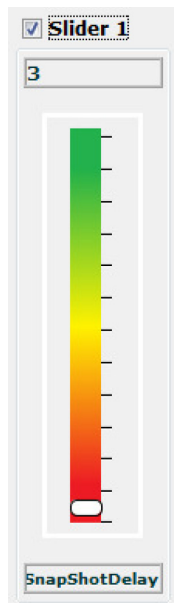
15. To plot variables in real time, enable Automated Event Control by clicking the automatic event execution icon found on the toolbar.



16. The DMCI window shows variables plotted in real time, which is updated automatically.



17. To change the time window to see more time on each plot, change the value of the SnapShotDelay, which controls how the buffers are being filled in the code.



## 1.5 I/O CONFIGURATION

### 1.5.1 Analog I/O Configuration

Refer PIM information sheet for the details regarding the analog circuitry and PIM schematics. The PIM information sheet document is available at [www.microchip.com/pims](http://www.microchip.com/pims).

### 1.5.2 Digital I/O Configuration

Functional Description	Device Pin Function	Input/output
<b>INVERTER A SECTION</b>		
PWM	RB10 through RB15	Output
Switch SW1	RC4	Input
Switch SW2	RG6	Input
POT1	RE13	Input
UART 2 TX (DE2-MCP8024 U8)	RG8	Output
UART 2 RX (DE2-MCP8024 U8)	RG7	Input
CE_A (Chip Enable MCP8024 U8)	RC13	Output
FAULT_MA	RC7/RP55	Output
<b>INVERTER B SECTION</b>		
PWM	RA7,RA10 & RD1 through RD4	Output
Switch SW3	RD15	Input
Switch SW4	RD13	Input
POT2	RE14	Input
UART 3 TX (DE2-MCP8024 U9)	RB8	Output
UART 4 RX (DE2-MCP8024 U9)	RE0	Input
CE_B (Chip Enable MCP8024 U9)	RF6	Output
FAULT_MB	RE1	Input
<b>OTHERS</b>		
UART 1 TX (RTDM)	RF1	Output
UART 1 RX (RTDM)	RC5	Input
Debug LED1	RD6	Output
Debug LED2	RD8	Output
Debug LED3	RG11	Output
Debug LED4	RF13	Output