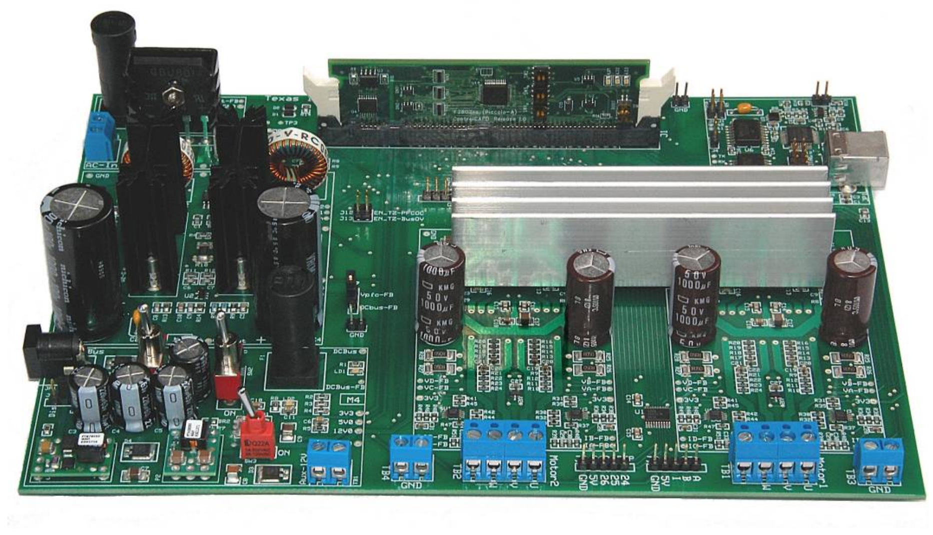
Motor Control and PFC Developer's Kit Overview

June 2009



The **Motor Control and PFC Developer's Kit** gives a great way to begin learning about digital motor control. This kit contains a motherboard that can accept any of the C2000 series controlCARDs. This board is divided into a Power Factor Correction (PFC) stage and two identical three-phase inverter stages.

Features of the Motor Control and PFC Developer's Kit board:

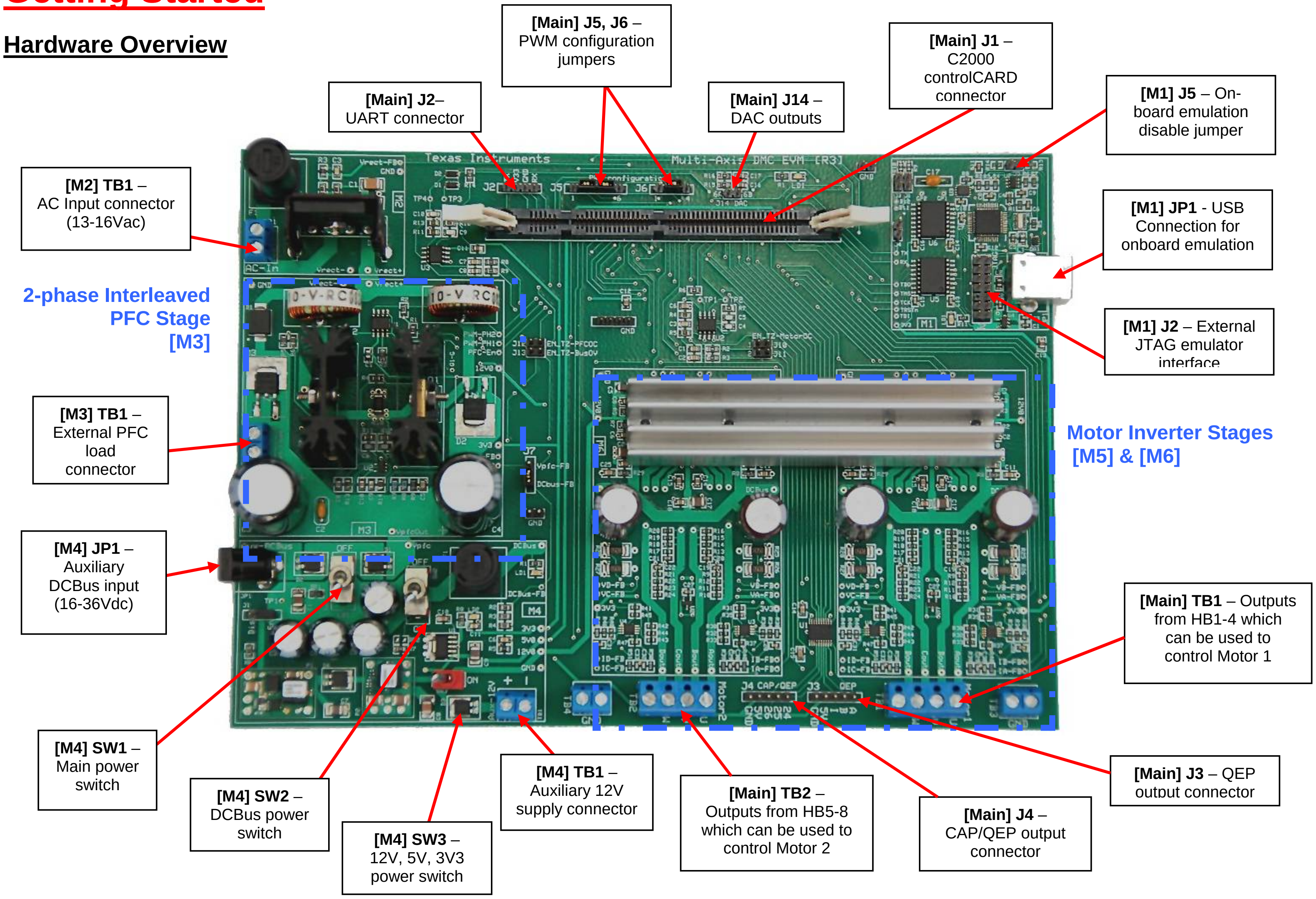
- Sensorless field oriented control of 2 permanent magnet motors using the Texas Instruments DRV8402 IPM module
- PFC provides current-shaping of the AC input and regulates the DCbus.
- Closed-loop digital control with feedback using the C2000's on-chip PWM and ADC peripherals
- On-board isolated JTAG emulation
- Over-current and over-voltage protection for the PFC stage and over-current protection for each inverter section
- UART communication header available for host-control
- Hardware Developer's Package is available and includes schematics, bill of materials, Gerber files, etc.

WARNING

This equipment may generate voltages and currents that can be injurious to humans, and must be used with caution. The user must employ appropriate safeguards to avoid serious injury.

Getting Started

Hardware Overview


[Main] J5, J6 – PWM configuration jumpers
[Main] J2– UART connector
[Main] J14 – DAC outputs
[Main] J1 – C2000 controlCARD connector
[M1] J5 – On-board emulation disable jumper
[M2] TB1 – AC Input connector (13-16Vac)
[M1] JP1 - USB Connection for onboard emulation
2-phase Interleaved PFC Stage [M3]
[M1] J2 – External JTAG emulator interface
[M3] TB1 – External PFC load connector
Motor Inverter Stages [M5] & [M6]
[M4] JP1 – Auxiliary DCBus input (16-36Vdc)
[Main] TB1 – Outputs from HB1-4 which can be used to control Motor 1
[M4] SW1 – Main power switch
[M4] SW2 – DCBus power switch
[M4] SW3 – 12V, 5V, 3V3 power switch
[M4] TB1 – Auxiliary 12V supply connector
[Main] TB2 – Outputs from HB5-8 which can be used to control Motor 2
[Main] J4 – CAP/QEP output connector
[Main] J3 – QEP output connector

More information on the Motor Control and PFC Developer’s Kit can be found in the document:
C:\TI_F28xxx_SysHW\Multi-Axis-HWdevPkg\Multi-Axis_HWGuide.pdf

Software Installation

The **Motor Control and PFC Developer's Kit** application software example, step-by-step lab style documentation, and other useful soft collateral can all be found on the TI website. The target mother board included in this kit is designed to be run in the Code Composer Studio v3.3 IDE.

To run any of the application specific software in the CCStudio IDE, you will also need to install Code Composer Studio v3.3, the Baseline Software Package for C2000 kits, and the Board Specific Software Package. A 32KB-limited version of the Code Composer IDE has been included with this kit. The Baseline software package contains the header files, libraries, etc necessary for the Code Composer project to compile. This baseline installer is common to all C2000 development kits and may not need to be downloaded if it is already installer to your computer.

The kit's software includes three separate projects to make the learning process easier. The 2xPM_Motors project and PFC2PHIL project each thoroughly go over the theory and process of learning motor control and PFC separately. The final project, PFC+2PM_Motors, shows the integrated and project and its documentation shows how this integration was achieved.

1) **Baseline** soft collateral and hardware installer
   - On an Internet browser type: http://www.ti.com/f28xkits
   - At the C2000 collateral page search for the Motor Control and PFC Developer's Kit and download the "Baseline Software" for this kit.
   - Save the .zip file to the directory of your choice
   - Unzip the file and run the install program Baseline Software Setup
   - The installer will create the following default directories:

```
C:\TI_F28xxx_SysSW
      ~Docs                       - contains general software documentation
      ~SupportFiles               - contains C2000 header files, key libraries, etc
C:\TI_F28xxx_SysHW                - contains schematics, etc for all controlCARDs
```

2) Install Code **Composer Studio v3.3**
   - Place the Code Composer trial version CD into your CD-ROM drive
   - Follow the automated installer through the rest of the install
   - See the document "QSG-CodeComposerC2000.pdf" for more information (C:\TI_F28xxx_SysSW\~Docs)

3) **Motor Control and PFC Developer's Kit** board specific collateral and hardware installer
   - On an Internet browser type: http://www.ti.com/f28xkits
   - At the C2000 collateral page search for the Motor Control and PFC Developer's Kit and download the "Board Specific" software download for this kit.
   - Save the .zip file to the directory of your choice
   - Unzip the file and run the install program Multi-AxisDMC.exe
   - The installer will create the following default directories:

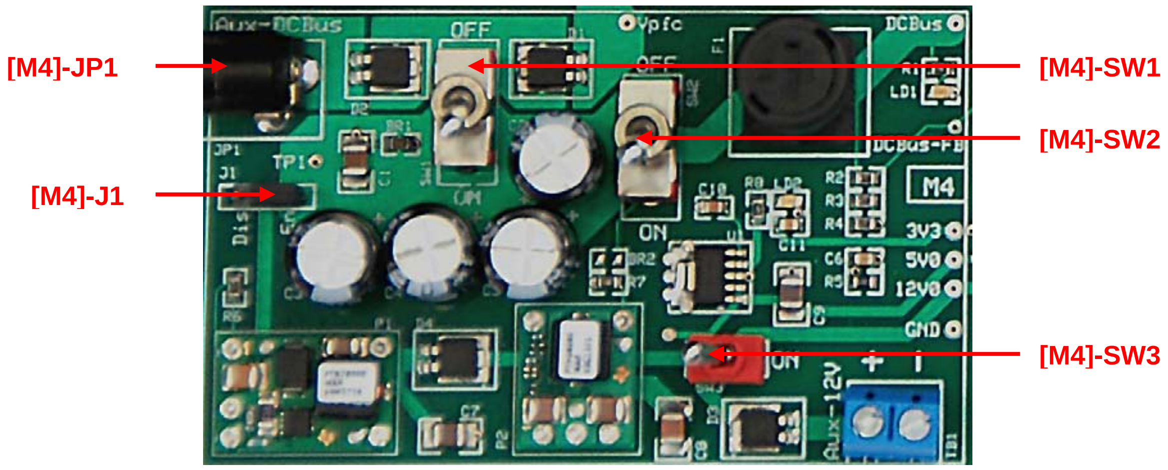
```
C:\TI_F28xxx_SysSW
      MotorCtrl+PfcKit
            ~Docs
            2xPM_Motors
                  ~Docs
            PFC2PHIL
                  ~Docs
            PFC+2PM_Motors
                  ~Docs

C:\TI_F28xxx_SysHW
      Multi-Axis-HWdevPkg
```

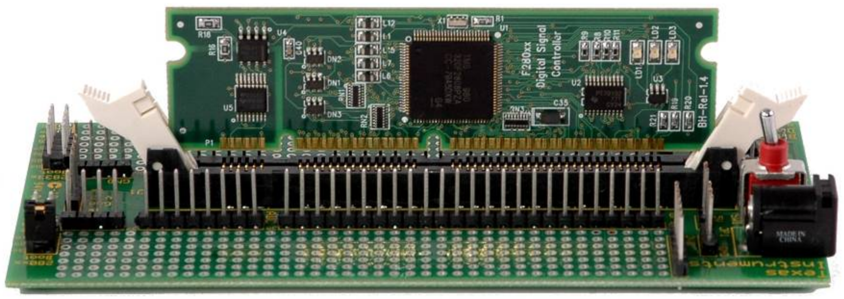
Hardware Setup

Note that the Multi-Axis DMC kit is separated into multiple function-specific macro blocks. Components shown below will be referred to with their macro number in brackets. For example, [M3]-C1 would refer to the C1 located in the macro M3.

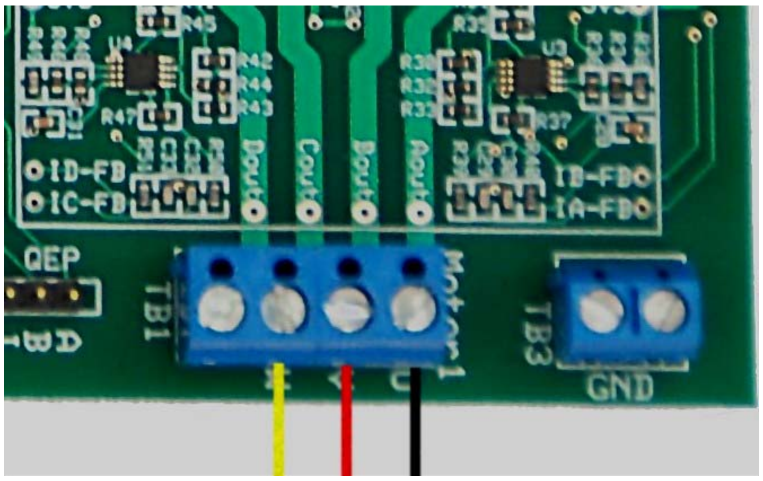
1) Ensure that [M4]-SW1, [M4]-SW2, and [M4]-SW3 are in the “Off” position. Ensure that [M4]-J1 is put in the “En” position.



2) Ensure that [Main]-J5 has 3 jumpers attached to it and [Main]-J6 has 2 jumpers attached to it.
3) Ensure that [Main]-J7 is jumpered in the “DCbus-FB” position.
4) Unpack the DIMM style controlCARD and place it loosely in the connector slot of [Main]-J1. See picture below.



5) Push vertically down using even pressure from both ends of the card until the clips snap and lock. (to remove the card simply spread open the retaining clip with thumbs)
6) Connect the motors to the board. Each motor will have several wires. Find the larger gauge wires of the motor (they should be yellow, black, and red) and connect the motor’s power wires to the U, V, and W terminals of [Main]-TB1 & [Main]-TB2 respectively.



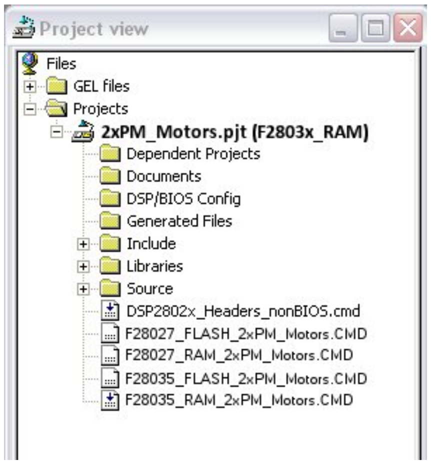
7) Connect 24Vdc supply to [M4]-JP1.
8) Turn [M4]-SW1 on. This enables power to be sent to the other two switches in [M4].
9) Turn on [M4]-SW2. This will enable DC power to be sent into the inverter stages. [M4]-LD1 should turn on.
10) Turn on [M4]-SW3. This will enable generation of the 12V, 5V, and 3.3V power rails, and turn on the controlCARD. [M4]-LD2 should turn on.
11) Connect a USB cable to connector [M1]-JP1. This will enable isolated JTAG emulation to the C2000 device. [M1]-LD1 should turn on.
    - If the included Code Composer Studio is installed the drivers for the onboard JTAG emulation will automatically be installed.
    - If a windows installation window appears try to automatically install drivers from those already on your computer. The emulation drivers are found at http://www.ftdichip.com/Drivers/D2XX.htm. The correct driver is the one listed to support the FT2232.

Note: For full details (schematics, pin-out table, etc) of the hardware please refer to the Multi-Axis_HWGuide and the Hardware Developer's Package, MultiAxis-HWdevPkg. See References for download location.

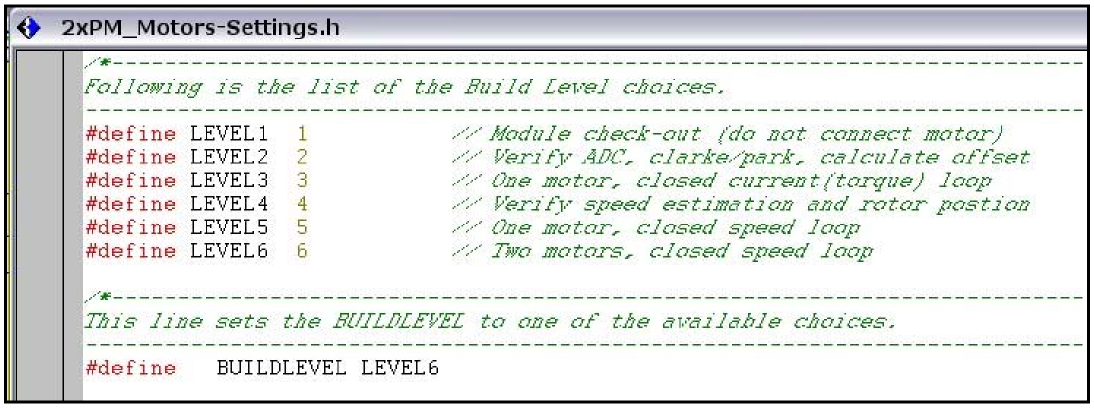
Running the Application

This document goes through how to configure and run the 2xPM_Motors.pjt project. To configure and run either of the other two projects please see its project specific documentation.

1. Configure Code Composer to work with an XDS100 emulator and a Piccolo F28035 MCU.
   - For details on setting up Code Composer Studio please see "QSG-CodeComposerC2000.pdf" (C:\TI_F28xxx_SysSW\~Docs)
2. Open Code Composer and attempt to connect (Debug -> Connect).
3. Load workspace 2xPM_Motors.wks, found in C:\TI_F28xxx_SysSW\MotorCtrl+PfcKit\2xPM_Motors\   (File->Load Workspace) This workspace will populate the watch window and graphs with variables useful in debugging.
4. Confirm that F2803x_RAM configuration is chosen from the "Select Active Configuration" dropdown menu. (this dropdown box should be located near the top of Code Composer's toolbar). At this point F2803x_RAM is the only supported build option.


Project view
Files
GEL files
Projects
2xPM_Motors.pjt (F2803x_RAM)
Dependent Projects
Documents
DSP/BIOS Config
Generated Files
Include
Libraries
Source
DSP2802x_Headers_nonBIOS.cmd
F28027_FLASH_2xPM_Motors.CMD
F28027_RAM_2xPM_Motors.CMD
F28035_FLASH_2xPM_Motors.CMD
F28035_RAM_2xPM_Motors.CMD

5. In the project window, expand the Include directory. In this directory open the file "2PM_Motors-Settings.h. Confirm incremental build is 5. Line 29 should be `#define   BUILDLEVEL LEVEL6`



2xPM_Motors-Settings.h

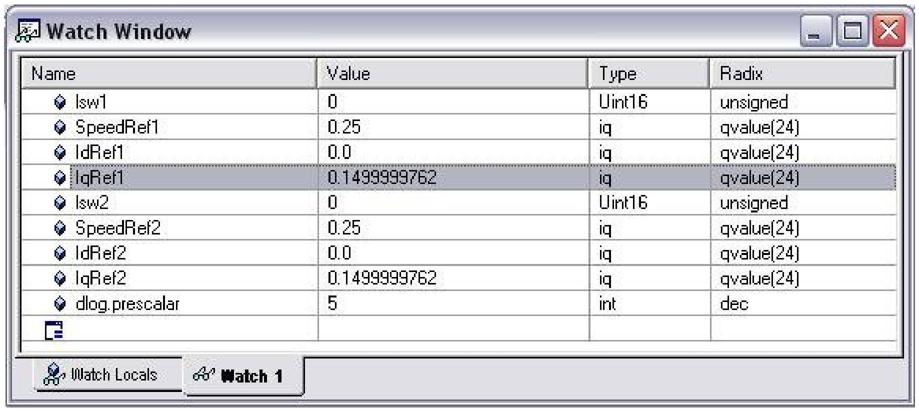
```
/*-----------------------------------------------------------------------------
Following is the list of the Build Level choices.
-----------------------------------------------------------------------------*/
#define LEVEL1  1           // Module check-out (do not connect motor)
#define LEVEL2  2           // Verify ADC, clarke/park, calculate offset
#define LEVEL3  3           // One motor, closed current(torque) loop
#define LEVEL4  4           // Verify speed estimation and rotor postion
#define LEVEL5  5           // One motor, closed speed loop
#define LEVEL6  6           // Two motors, closed speed loop

/*-----------------------------------------------------------------------------
This line sets the BUILDLEVEL to one of the available choices.
-----------------------------------------------------------------------------*/
#define   BUILDLEVEL LEVEL6
```

6. Prepare to run the software by clicking:
   - Project -> Rebuild All
   - File -> Load Program -> "2803x_RAM\ 2xPM_Motors.out"
   - Debug -> Reset CPU

- Debug -> Restart
- Debug -> Go Main
- Debug -> Real-time Mode.
  (A message box *may* appear. If so, select YES to enable debug events. This will set bit 1 (DGBM bit) of status register 1 (ST1) to a "0". The DGBM is the debug enable mask bit. When the DGBM bit is set to "0", memory and register values can be passed to the host processor for updating the debugger windows.)
- Debug -> Run
- On the watch window and all graphs, right-click and select "Continuous Refresh".

7. Both motors should currently be in a locked rotor state (the rotor should resist any attempt at rotor rotation).



Watch Window

Name	Value	Type	Radix
Isw1	0	Uint16	unsigned
SpeedRef1	0.25	iq	qvalue(24)
IdRef1	0.0	iq	qvalue(24)
IqRef1	0.1499999762	iq	qvalue(24)
Isw2	0	Uint16	unsigned
SpeedRef2	0.25	iq	qvalue(24)
IdRef2	0.0	iq	qvalue(24)
IqRef2	0.1499999762	iq	qvalue(24)
dlog.prescalar	5	int	dec

8. Change the variable "Isw1" to 1. This will close the current (torque) loop of motor 1 and force constant torque through motor 1. This does however mean that as load increases speed may not remain constant.
9. Change the variable "Isw1" to 2. This closes the speed loop of motor 1. Now speed will remain constant and torque will adapt to the load even as the load changes.
10. The default value of SpeedRef1, the speed control variable, is 0.25. In the watch window, change the speed reference, SpeedRef1, to 0.5. The motor will speed up by a factor of 2. As load increases on the motor, the torque will compensate and the motor will stay at a set speed reference.
    - Look at the graph window located in your workspace and note the estimated rotor angle and current waveform of Phase U (phase A of the DRV8402). If the load is low, minimal current will be delivered to the load and the current waveform and angle waveform will seem distorted. As the load increases, these waveforms will become less distorted. Slightly changing the variable dlog.prescaler in the watch window may become necessary to view full waveforms clearly.
    - Because of variations in board components and motor build quality there may be an offset on the current waveform. This offset can cause minor performance issues if it is not compensated. Please see Phase 2C of the document "Field Oriented Control of PM Motors.
11. Continue experimenting by editing the motor1's speed reference. Note that the control parameters are optimized for the region from 0.2 to 0.8. Outside of this range, the PID and sensorless algorithm coefficients may need further tuning.
12. Next experiment with motor 2. Follow step 11 and 12 with Isw2 (controls the motor 2's status) and SpeedRef2 (controls motor 2's speed).
13. To shut down the board
    - Set Isw1 and Isw2 to 0
    - Debug -> Halt
    - Debug -> Real-time Mode
    - Debug -> Reset CPU

- Debug -> Disconnect
- Turn off [M4]-SW3, then [M4]-SW2, then [M4]-SW1.

Board is now unpowered. Remove cables and connectors as necessary

References

For more information please see the following guides:

- **QSG-CodeComposerC2000** – a step-by-step instruction guide on how to install Code Composer and get C2000 onboard emulators to work with Code Composer Studio.

  C:\TI_28xxx_SysSW\~Docs\QSG-CodeComposerC2000.pdf

- **2xPM_Motors –** provides detailed information on the 2xPM_Motors project within an easy to use lab-style format.

  C:\TI_F28xxx_SysSW\MotorCtrl+PfcKit\2xPM_Motors\ ~Docs\2xPM_Motors.pdf

- **PFC2PHIL –** provides detailed information on the PFC2PHIL project within an easy to use lab-style format.

  C:\TI_28xxx_SysSW\ MotorCtrl+PfcKit\PFC2PHIL\~Docs\ PFC2PHIL.pdf

- **PFC+2PM_Motors –** provides detailed information on the PFC+2PM_Motors project within an easy to use lab-style format.

  C:\TI_28xxx_SysSW\ MotorCtrl+PfcKit\PFC+2PM_Motors\~Docs\ PFC+2PM_Motors.pdf

- **Multi-Axis_HWGuide** – gives more information on the hardware of the Motor Control and PFC Developer's Kit. Includes details on all the various connectors, hardware block diagrams, etc

  C:\TI_28xxx_SysHW\MultiAxis-HWdevPkg\Mult-Axis_HWGuide.pdf

- **Multi-Axis-HWdevPkg** – a folder containing various files related to the hardware on the Motor Control and PFC Developer's Kit board (schematics, bill of materials, Gerber files, PCB layout, etc).

  C:\TI_28xxx_SysHW\MultiAxis-HWdevPkg\