

CROSSHAIRS []
EMBEDDED

Invaluable [insight] through precise software solutions

[**Crosshairs Embedded**
Dual Motor Control and PFC Developer's Kit
User Guide]

Contents

Contents	2
Requirements	3
Crosshairs [debugger] and Crosshairs [interface designer]	3
Code Composer Studio 4	3
The Dual Motor Control and PFC Developer's Kit	3
controlSUITE Installation	4
Setup and Configuration	6
Working with CCS4	8
Configuring your workspace	8
Importing example project.....	8
Understanding the example	9
Crosshairs [commros] integration.....	10
Integration overview	10
The Crosshairs [commros] user manual.....	10
Crosshairs [commros] configuration (Commros_user.c).....	10
Changes to the original 2xPM_Motors example code (2xPM_Motors.c)	11
Compiling	12
Programming the Motor Control Kit.....	12
Using the Crosshairs [debugger]	13
Creating a debug session.....	13
Debugging the application with the Real-time Monitor	14
Using the Data Logger.....	16
FAQ	19
Customer Contact.....	20

Requirements

Before proceeding, please make sure you have the following software and hardware properly installed and configured on your computer.

Crosshairs [debugger] and Crosshairs [interface designer]

The Crosshairs [debugger] and Crosshairs [interface designer] may be downloaded for free (30-day trial) from the following location:

<http://crosshairembedded.com/free-trial>

Please note that after registering for a free trial, the download links are only valid for a limited time. Download and install the Crosshairs [debugger] and Crosshairs [interface designer]. The stand-alone installation of the Crosshairs [engine] is not required for this tutorial.

Code Composer Studio 4

This may be downloaded for free from Texas Instruments at the following location:

<http://focus.ti.com/docs/toolsw/folders/print/ccstudio.html>

In this tutorial, Code Composer Studio 4 Microcontroller Limited edition was used (TMD FCCS-MCULTD.)

The Dual Motor Control and PFC Developer's Kit

This guide is applicable for both the single motor and dual motor version of the Motor Control Kit.

The Dual Motor Control and PFC Developer's Kit (TMDS2MTRPFCKIT) is available from Texas Instruments or one of its suppliers. For ordering information, please refer to the following URL:

<http://focus.ti.com/docs/toolsw/folders/print/tmds2mtrpfckit.html>

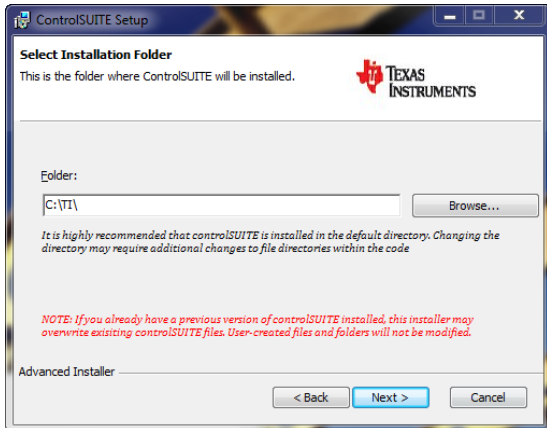
Bundled with the Dual Motor Control and PFC Developer's kit is a DIMM style TI Piccolo Microcontroller (TMS320F28035). Specifications and other information about this controlCARD can be found at the following location:

<http://focus.ti.com/docs/prod/folders/print/tms320f28035.html>

controlSUITE Installation

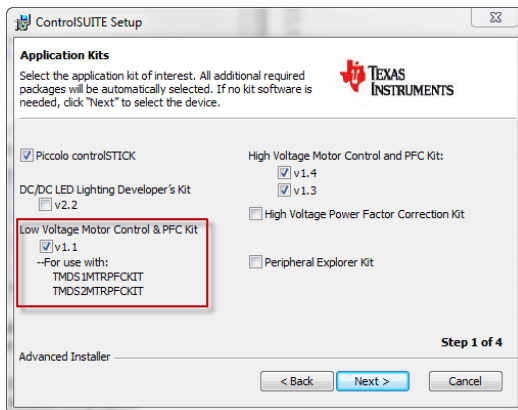
The relevant software and resource files needed for development on the Dual Motor Control and PFC Developer's Kit (besides the CCS4 installation itself) is an installation of the TI controlSUITE software package. This is available from the link below:

<http://focus.ti.com/docs/toolsw/folders/print/controlsuite.html>

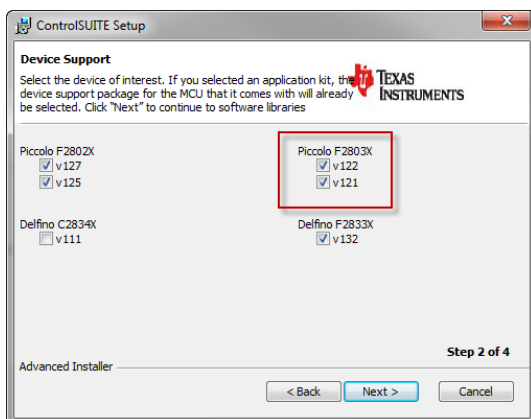


We recommend using the default path for installation "C:\TI\" for simplicity. If this installation path is changed certain include folders and library folders may need to be changed in the example projects for the example application to work.

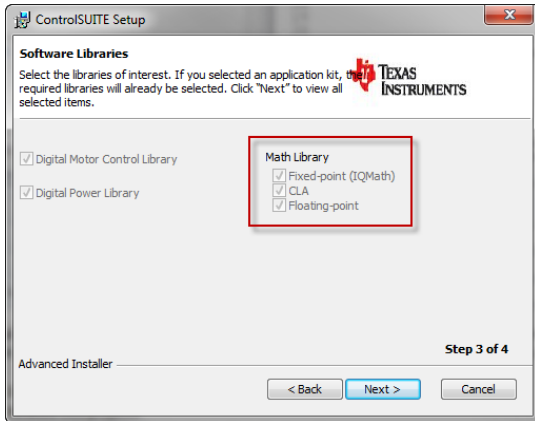
When installing controlSUITE please ensure that these checkboxes are checked:



Low Voltage Motor Control and PFC Kit



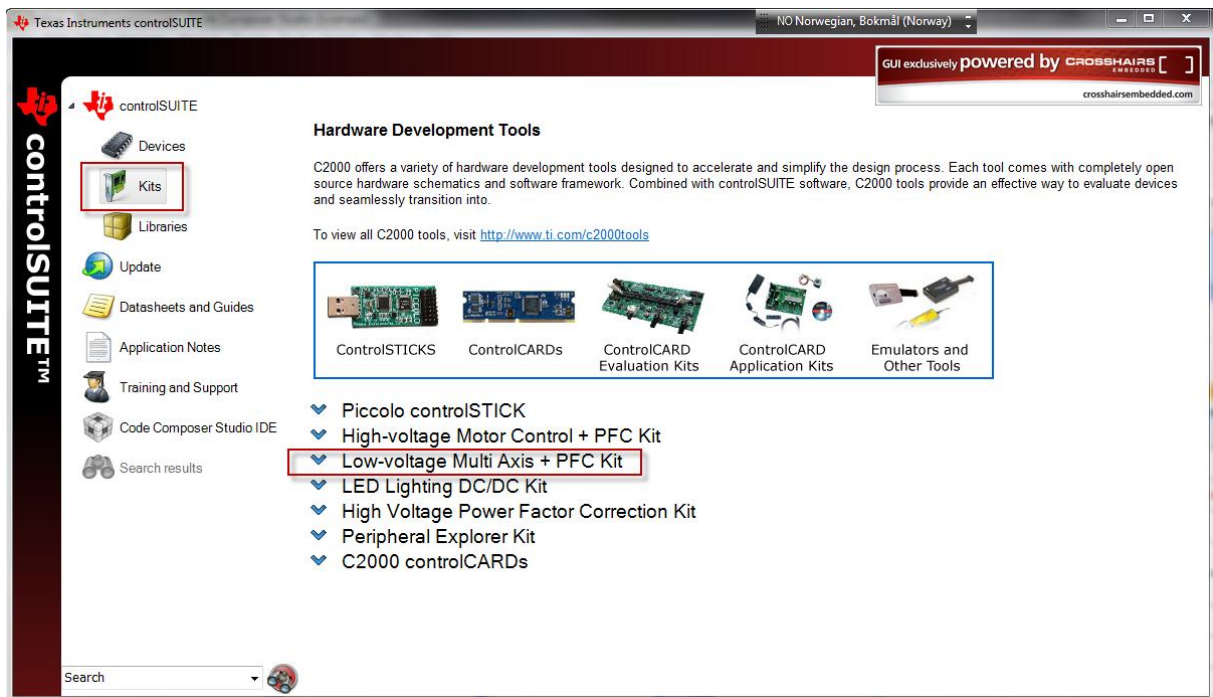
Pico F2803x support files



Digital Motor Control Library, Fixed-point Math library CLA and Floating-point Math library

After a successful install you can browse the installed files for the development kits and the support files for the different controlCARDs etc. through the controlSUITE Desktop application.

Open up the controlSUITE Desktop and expand the controlSUITE node and select Kits. In Kits press the arrow to expand Low-voltage Multi Axis + PFC Kit:



There you will find a lot of resources for the example used, the device kit and How to run-guides to help you get this kit up and running.

Note that the *Commros_v2.2_regular_logger_PFC-MotorControlKit* example that we are going to use in this user-guide is a self-contained version of the 2xPM_Motors example provided by the aforementioned software packages. In this project Crosshairs [commros] has been integrated and all the relevant header files, command files, assembly snippets and libraries are already inside the project.

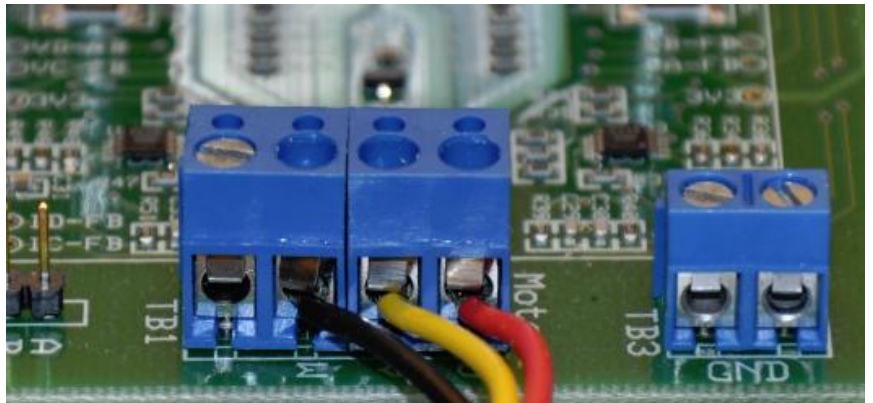
Setup and Configuration

Connecting the motor

After unboxing the Motor Control and PFC Developer's Kit, the provided motor(s) needs to be connected to the board. In this example we will only be using Motor1, which needs to be connected to the Dual Motor Control Developers Kit.

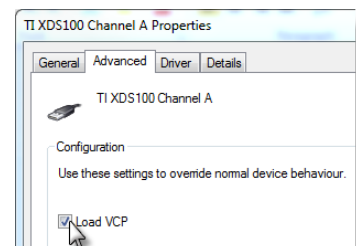
Connecting the DIMM style controlCARD

The DIMM style TMS320F28035 controlCARD must be slotted in J1



Setting up the virtual serial ports

After installing the Motor Control and PFC Developer's Kit drivers on your system and connecting the Motor Control board to the computer using the USB cable, connect the power supply and power the on the board. When the board has been turned on and connected to the computer, new USB devices should appear on your system. These are the virtual JTAG/Serial ports available via the USB-connection. Open the Windows Device Manager. Expand the Universal Serial Bus Controllers section and find the two TI XDS100 entries (Channel A and Channel B.)

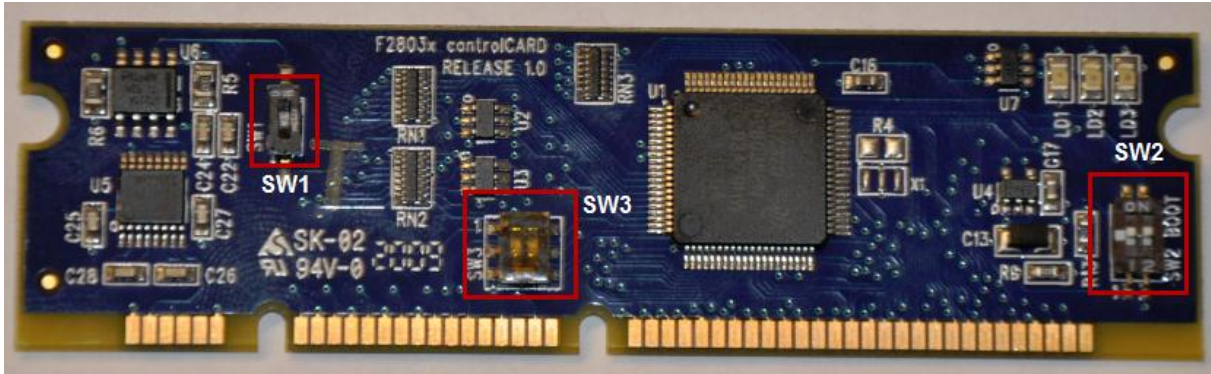


For both entries, open the Properties dialog and select the Advanced tab. Click to enable "Load VCP" as shown in the screenshot. This enables the virtual serial port for the device.

After enabling VCP for both channels, unplug and replug the USB-cable for the Motor Control and PFC Developer's Kit. You should now see two new COM-ports in your Device Manager under the Ports (COM & LPT) section.

Jumpers and Switches

Please make sure that the following jumpers and switches are in the correct positions



SW1	
POS	1
ON	
OFF	

SW3		
POS	1	2
ON		
OFF		

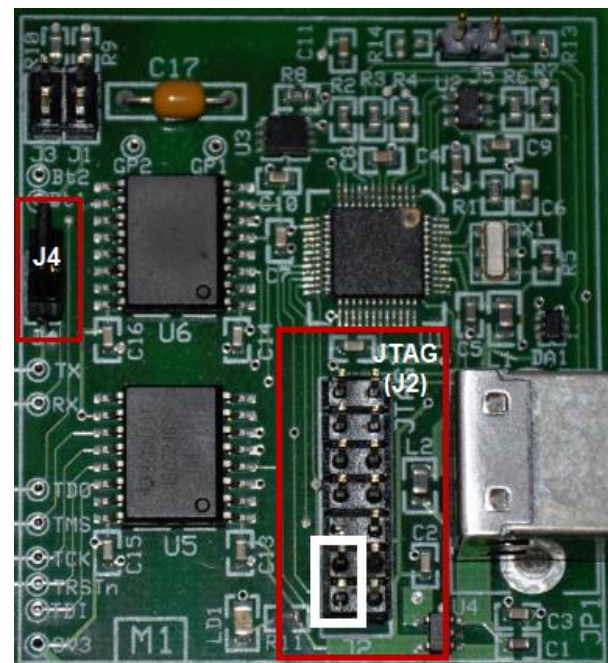
SW2		
POS	1	2
ON		
OFF		

SW1: Controls whether GPIO-28 is used for RS232 or not. Turning this **off** releases GPIO-28 and let us use the FTDI-chip for serial communication

SW2: Controls boot mode, currently set to boot from FLASH.

[M1] J4: Putting a jumper over J4 sets the FTDI-chip in Serial/JTAG mode. This is required to be able to connect to the kit through the Crosshairs [debugger] and Crosshairs [interface designer] using the USB-cable and Virtual Serial Port emulation.

[M1] J2: If a jumper is placed over the pins enclosed in white in the screenshot, JTAG will be permanently disabled. Disabling JTAG is sometimes required to ensure correct boot from flash. Note that the when 2xPM_Motors example is running the LED marked LD3 will blink red on the controlCARD. The jumper over [M1] J2 must be removed to be able to program the controlCARD in Code Composer Studio using the JTAG emulator (over USB Cable).



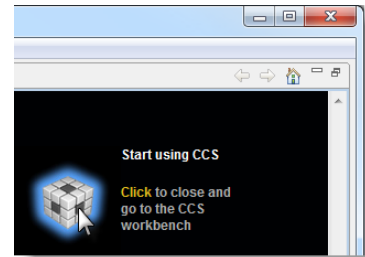
Working with CCS4

After successfully installing Code Composer Studio 4 and completing the Requirements section above, please start CCS4.

Configuring your workspace

When you start CCS4, you will by default be prompted to select the workspace you would like to use. Select a folder you would like to use and click OK. Please note that if you are an existing CCS4 user, you might want to create a new workspace, for example `\crosshairs_examples`.

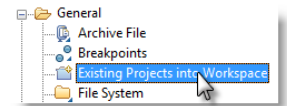
When CCS4 loads and the welcome screen is visible, click the cube in the top-right corner to access the CCS Workbench.



Importing example project

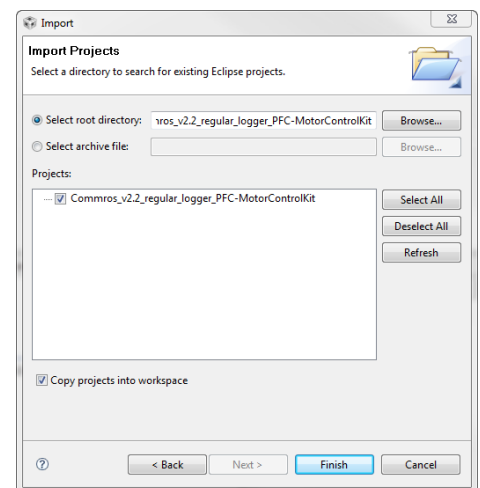
With a clean workspace, open the File menu and select **Import...**

Expand the General folder and select Existing Projects into Workspace and click Next.



In the next step, we browse for a project folder. Set the root directory to `Documents\Crosshairs Embedded\crosshairs [commros] \TMS320F28xxx\Examples\CCS4\Commros_v2.2_regular_logger_PFC-MotorControlKit`.

Note that by default, CCS4 does not make a copy of imported projects into your workbench. Rather than editing the original files, we recommend checking the Copy projects into workspace checkbox before clicking Finish.



Understanding the example

In the C/C++ Projects tab, you will now see the Motor Control example. A description of the files follows below:

src commros

- commros_regular.h
 - Include for the regular version of Commros. This version of Commros includes Datalogger functionality
- Commros_user.h/.c
 - User-specific Commros configurations
- ProcessorDependent.h
 - Defines known data sizes.

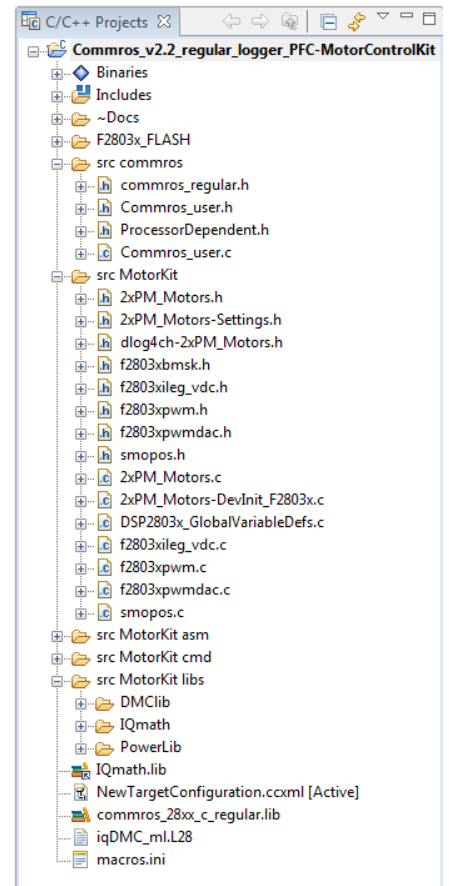
src MotorKit

This folder contains files from the original 2xPM_Motors example and includes, libraries and sources from the *Baseline Software Setup* and *Motor Control and PFC Developer's Kit Software* required for operation.

2xPM_motors.c holds the entry point of the application.

others

The rest of the files in this project are command files to facilitate placing the code in flash on a F28035, various assembly snippets for processor specific functionality, IQMath, DMCLib and PowerLib libraries and finally the commros_28xx_c_regular.lib



For a simpler example of integrating Crosshairs [commros] to an empty application, please look at the user-guide for the [F2803x Piccolo Experimenter's Kit](#). Even though the guide is written for a different development kit, the software should function the same way when running on the Motor Control Kit.

Crosshairs [commros] integration

Integration overview

- The 2xPM_Motors example has been modified to integrate the Crosshairs [commros] library.
- Serial Interface A (SCIA GPIO) of the Motor Control Kit has been dedicated to Crosshairs [commros].
- Probe points have been configured for on-board high resolution data logging.
- Calls to service routines for Crosshairs [commros] and the DataLogger has been added to the original example code (2PM_Motors).
- The Crosshairs [commros] regular version library is added to the project.

The Crosshairs [commros] user manual

Before proceeding, please read the Crosshairs [commros] User Guide that is installed with all Crosshairs products. This is found in the Crosshairs Embedded program group on the start menu.

Crosshairs [commros] configuration (Commros_user.c)

Two probe-points for the data-logger have been defined, together with a buffer where logs from the data-logger is stored on-board.

```
struct LoggerVarStruct loggerVars[4];  
struct LoggerProbeStruct probes[2];  
  
#define SIZE_OF_BUFFER 0x400  
unsigned char buffer[SIZE_OF_BUFFER];
```

The serial port is configured and activated in *SetupSerialPort*.

The functions *SCITransmitByte*, *SCIReceiveByte* and *SCIDataAvailable* point to where data can be read from or written to using the serial port.

```
probes[0].pName = "Mainloop";  
probes[0].sampleTime = 1.0;  
probes[1].pName = "MainISR";  
probes[1].sampleTime = 0.00010;  
  
AddProbes(&commros.m_datalogger, probes, 2);  
AddLoggerVariableBuffers(&commros.m_datalogger, loggerVars, 4);  
DataloggerSetBuffer(&commros.m_datalogger, buffer, SIZE_OF_BUFFER);
```

In *InitCommros* the Crosshairs [commros] instance is initialized and the probe points are configured and activated. There are two probe points. One probe point configured to run in the main-loop slated to sample of every second. One probe-point is configured to run in the Main Interrupt Service Routine (MainISR), slated to sample every 0.00010 seconds.

Changes to the original 2xPM_Motors example code (2xPM_Motors.c)

Changes (additions) to the original code are shown in snippets where the additions have been emphasized in **yellow**. An include for the Crosshairs [commros] library has been added.

```
#include "IQmathLib.h"  
#include "2xPM_Motors.h"  
#include "2xPM_Motors-Settings.h"  
#include <math.h>  
#include "Commros_user.h"
```

Initialization of Commros has been added before the Main loop.

```
// Enable CNT_zero interrupt using EPWM1 Time-base  
EPwm1Regs.ETSEL.bit.INTEN = 1; // Enable EPWM1INT generation  
EPwm1Regs.ETSEL.bit.INTSEL = 1; // Enable interrupt CNT_zero event  
EPwm1Regs.ETPS.bit.INTPRD = 1; // Generate interrupt on the 1st event  
EPwm1Regs.ETCLR.bit.INT = 1; // Enable more interrupts  
  
InitCommros();  
  
// Enable CPU INT3 for EPWM1_INT:  
IER |= M_INT3;  
// Enable global Interrupts and higher priority real-time debug events:  
EINT; // Enable Global interrupt INTM  
ERTM; // Enable Global realtime interrupt DBGM
```

A call to run the Crosshairs [commros] service routine has been added in the Main-loop.
A call to run the datalogger with the first probe-point has been added to the Main-loop.

```
for(;;) //infinite loop  
{  
    // State machine entry & exit point  
    //=====  
    (*Alpha_State_Ptr)(); // jump to an Alpha state (A0,B0,...)  
    //=====  
    ServiceRoutine(&commros);  
    Datalogger(&commros.m_datalogger,0);  
}
```

A call to run the Datalogger with the second probe-point has been added to the MainISR function.

```
// -----  
// Call the PWMDAC update function.  
// -----  
// pwmdac1.update(&pwmdac1);  
  
// -----  
// Call the DATALOG update function.  
// -----  
dlog.update(&dlog);  
Datalogger(&commros.m_datalogger,1);  
  
#if (DSP2803x_DEVICE_H==1)  
// Enable more interrupts from this timer
```

Compiling

There should be no need to change any of the code. Just compile the example by right-click on the Project and select Build Project or open the Project menu and select Build Project from there. If there are compile errors, please refer to the FAQ on page 14 or contact support@crosshairseembedded.com

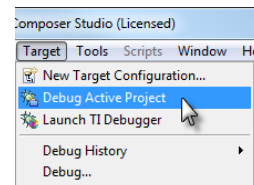
The out-file and be generated in the folder F2803x_FLASH.

Programming the Motor Control Kit

To program the application to the Dual Motor Control and PFC Developer's Kit, open the Target menu and select Debug Active Project.

If prompted to create a new target configuration, click *Yes* and *Finish*. In the target configuration, change the Connection and Device as follows:

- Connection: Texas Instruments XDS100v1 USB Emulator
- Device: Developer's Kit- Dual Motor Control and PFC (F28035)



Click Save and launch the debug session again (open the Target menu and select Debug Active Project.)

If the debug session is launched correctly, you should now be in the Debug perspective and the program counter should be located on the first line in the main function. To verify that the application runs correctly, add a hardware breakpoint inside the Main-loop and click run a few times and see that the reaches the breakpoint.

Before proceeding to the Crosshairs products, disconnect the debug session inside CCS4. You may also close CCS4 at this point. Please note that if you halted the application before exiting unplug and re-plug the USB-cable for the Dual Motor Control and PFC Developer's Kit and restart the board by flipping [Main] SW3 off and on in order to restart the application.

The LED marked LD3 on the controlCARD should be blinking red after the program has started up and is running without any JTAG breakpoints.



If the LED marked LD3 isn't blinking red, the program hasn't booted. You may need to revisit the chapter called [Jumpers and Switches](#) earlier in this document, to ensure that the jumpers on the controlCARD is set correctly and that JTAG is disabled.

Using the Crosshairs [debugger]

For more comprehensive usage instructions, please refer to the Crosshairs [debugger] User Guide that is installed with the product.

Creating a debug session

After launching the Crosshairs [debugger] open the File menu and select New Project. Give your project a name and provide an (optional) description.

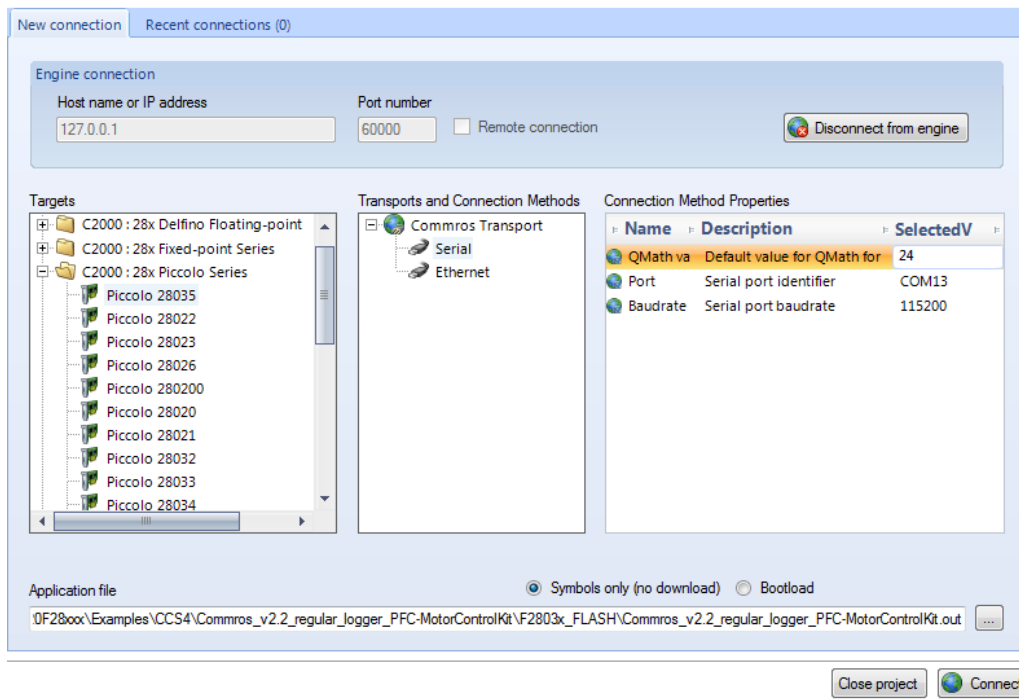
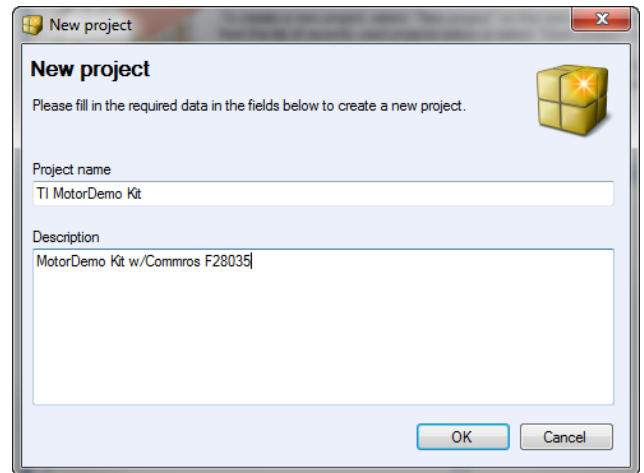
After clicking OK, you are taken to the Connection configuration screen. This is where the connection to the Crosshairs [engine] and, in turn, the hardware is established.

If you are connecting to an engine running on your local computer, leave the Engine Connection settings as they are and click the Connect to Engine button. If prompted to start the engine, click Yes.

In the Targets list, expand the C2000:28x Piccolo Series and select Piccolo 28035.

Select Commros Serial as the transport and select the correct COM-port in the Connection method properties pane.

Next, select the Application file (out-file) by clicking the [...] button in the lower right corner. Browse to the \F2803x_Flash folder of the CCS4 project that was created in the previous sections. Once everything is configured properly, click the Connect button.



If the connection fails, please refer to the FAQ on page 19 or contact support@crosshairseembedded.com for assistance.

Debugging the application with the Real-time Monitor

Once the connection is established, you will be in the Monitor module of the Crosshairs [debugger].

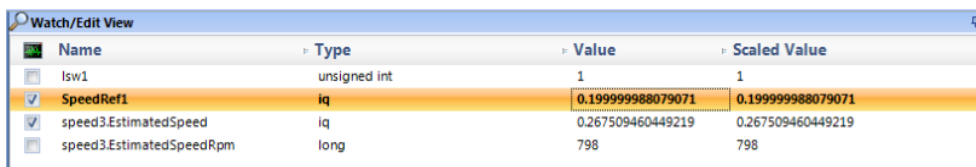
On the left side, you will find the Symbols View. Variables to be watched or edited can be dragged into the Watch and Edit view from the Symbols View.

Some values of interest:


lsw1	The mode of operation of the motor. 0 = Motor off 1 = Tuning algorithm 1 2 = Tuning algorithm 2
SpeedRef1	Normalized ref speed of the motor (between 0 and 1)
speed3.EstimatedSpeed	Normalized estimated speed of the motor
speed3.EstimatedSpeedRPM	Estimated RPM of the motor


Add these variables to the Watch/Edit view by either dragging them to the view, or right-clicking and selecting Add to Watch View.

The variables should now be visible in the Watch/Edit View. The leftmost column of the Watch/Edit view is used to control which variables are graphed once updating is turned on. In this example SpeedRef1 and speed3.EstimatedSpeed is checked for graphing.

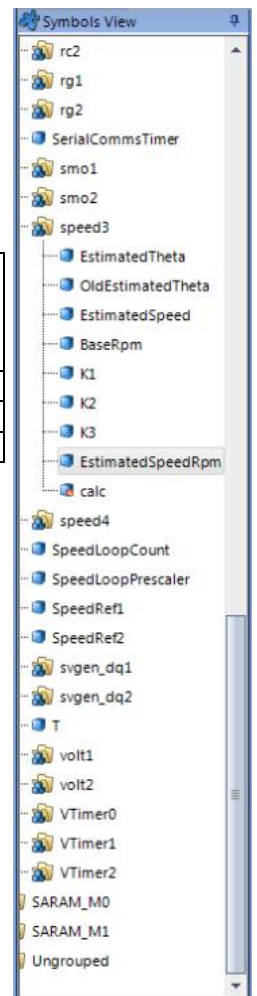


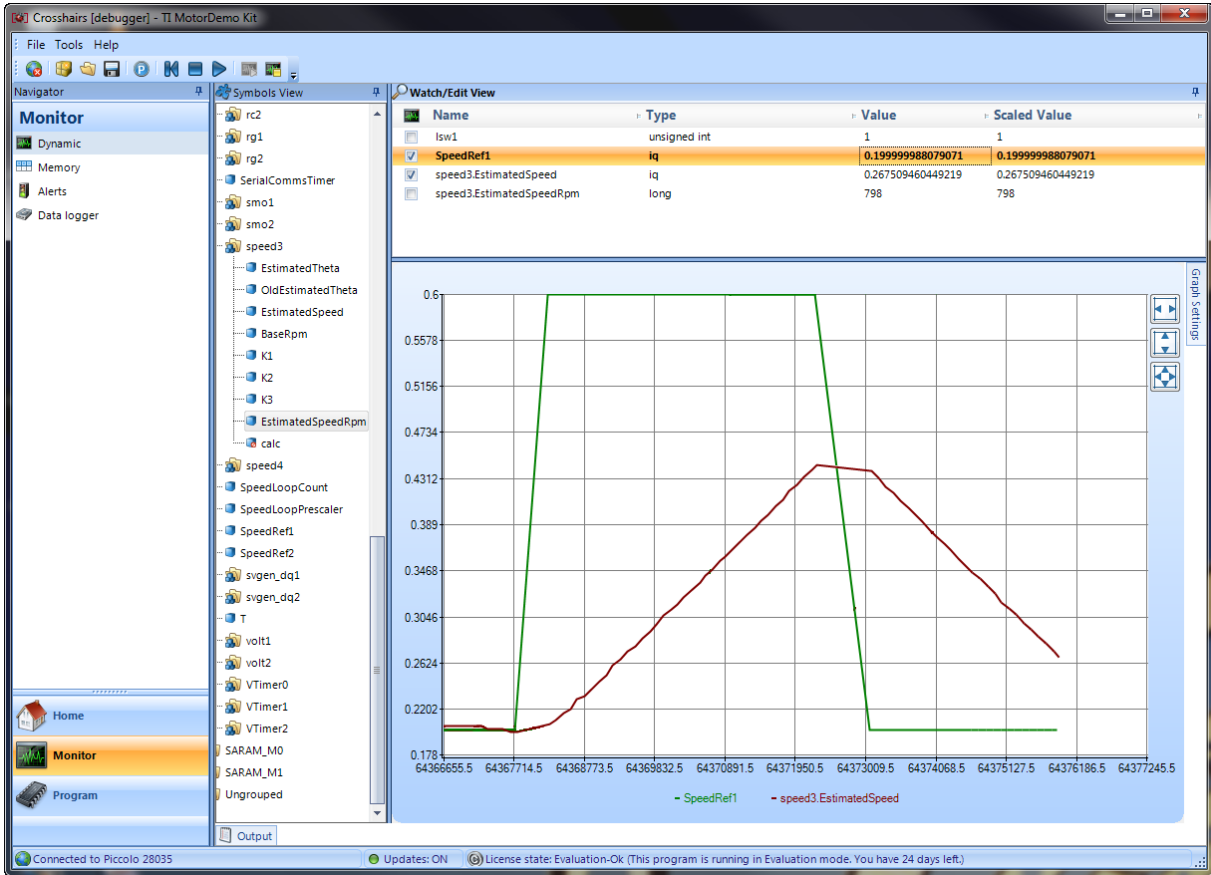
Name	Type	Value	Scaled Value
<input type="checkbox"/> lsw1	unsigned int	1	1
<input checked="" type="checkbox"/> SpeedRef1	iq	0.199999988079071	0.199999988079071
<input checked="" type="checkbox"/> speed3.EstimatedSpeed	iq	0.267509460449219	0.267509460449219
<input type="checkbox"/> speed3.EstimatedSpeedRpm	long	798	798

Real-time updating can be set either by clicking the Start update button (), or by opening the Tools menu and selecting Start Update. To start the motor click the cell under Value for lsw1 and set the value to 1

You should now see values updating as well as the graph plotting SpeedRef1 and speed3.EstimatedSpeed. Note that if the graph plotted is too flat, click the *Fit to window* button  on the graph.

It is now possible to control the speed of the motor by editing SpeedRef1. You can do this by clicking the cell under Value and inputting a value between 0 and 1.

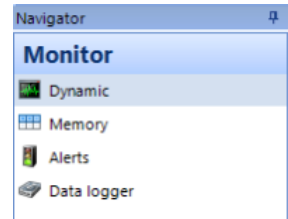




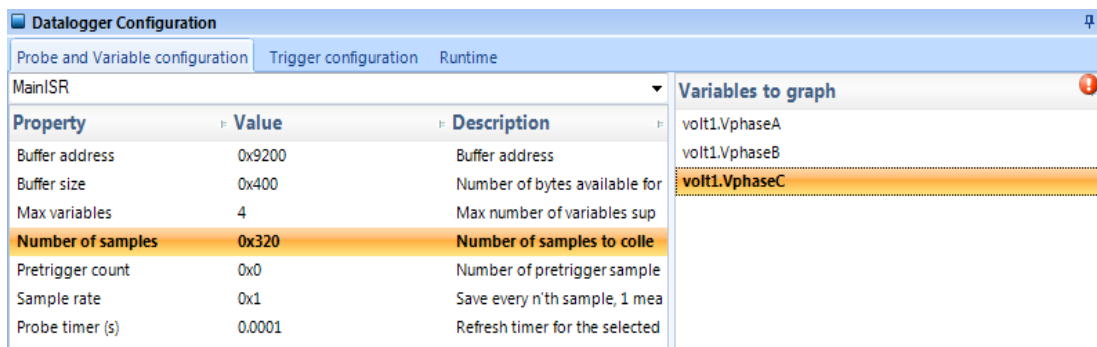
Using the Data Logger

Using the Data Logger requires integration of the regular or full version of Crosshairs [commros] into the application running on the embedded target and probe-points configured and utilized according to chapter called [Crosshairs \[commros\] integration](#).

Enter into the Data logger Module by clicking Data Logger in the navigator. The available probe points will be selectable in the drop down under Probe and Variable Configuration. Select the probe-point called MainISR.

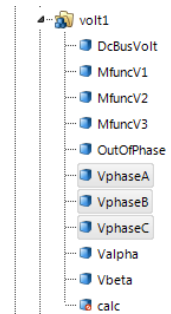


Configure the properties in the grid according to this example:



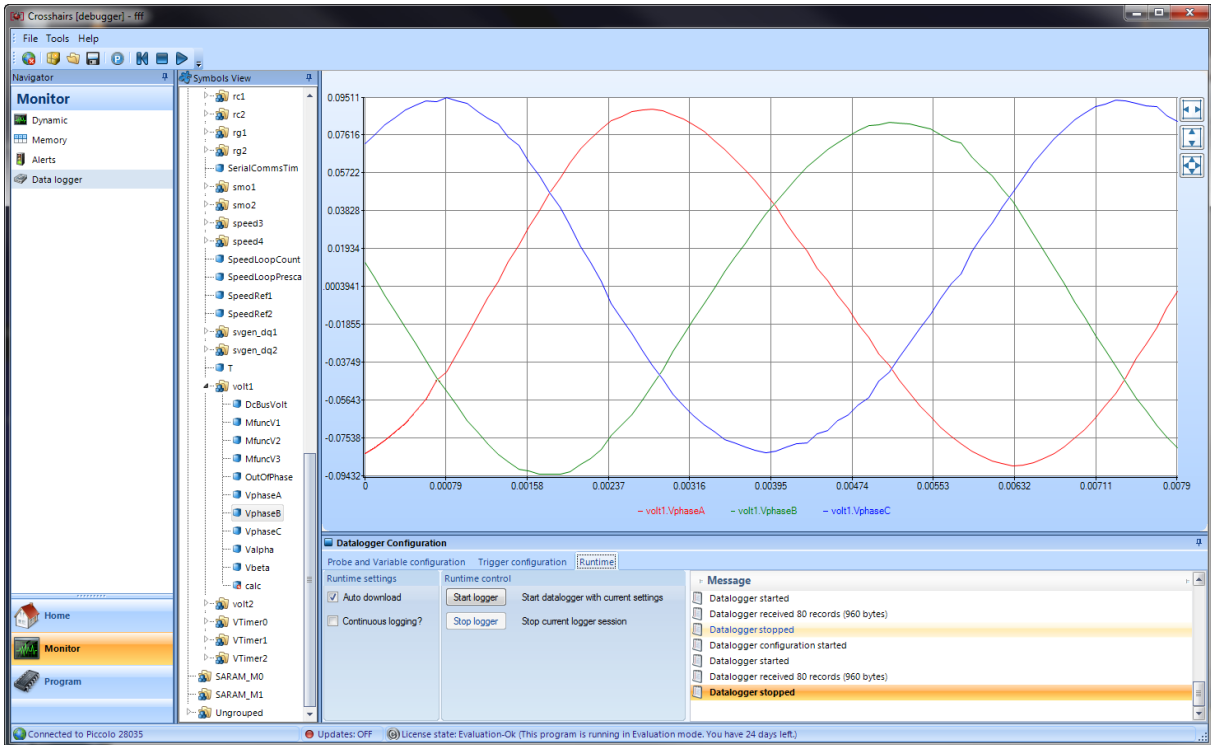
The variables volt1.VphaseA, volt1.VphaseB and volt1.VphaseC are variables that shows the three voltage phases for the motor. These can be added for Data Logging by either dragging them from the symbols view or selecting them, right clicking with the mouse and select to add them for graphing.

It is possible to make the Data Logger start up when certain conditions are met in the application. These configurations can be set in the Trigger configuration, where triggering based on value-comparison or raising or falling edges of variables is allowed. If the Data Logger is meant to trigger at any time, there is no need to do any trigger configuration.



In the Runtime tab press the Start Logger button. This transfers the configurations to Crosshairs [commros] where data will be logged when trigger-conditions are met. When the buffer is full, the resulting values will be sent back and displayed in the graph-view.

Here is an example output of the voltage phases of the motor logged by the Data Logger:

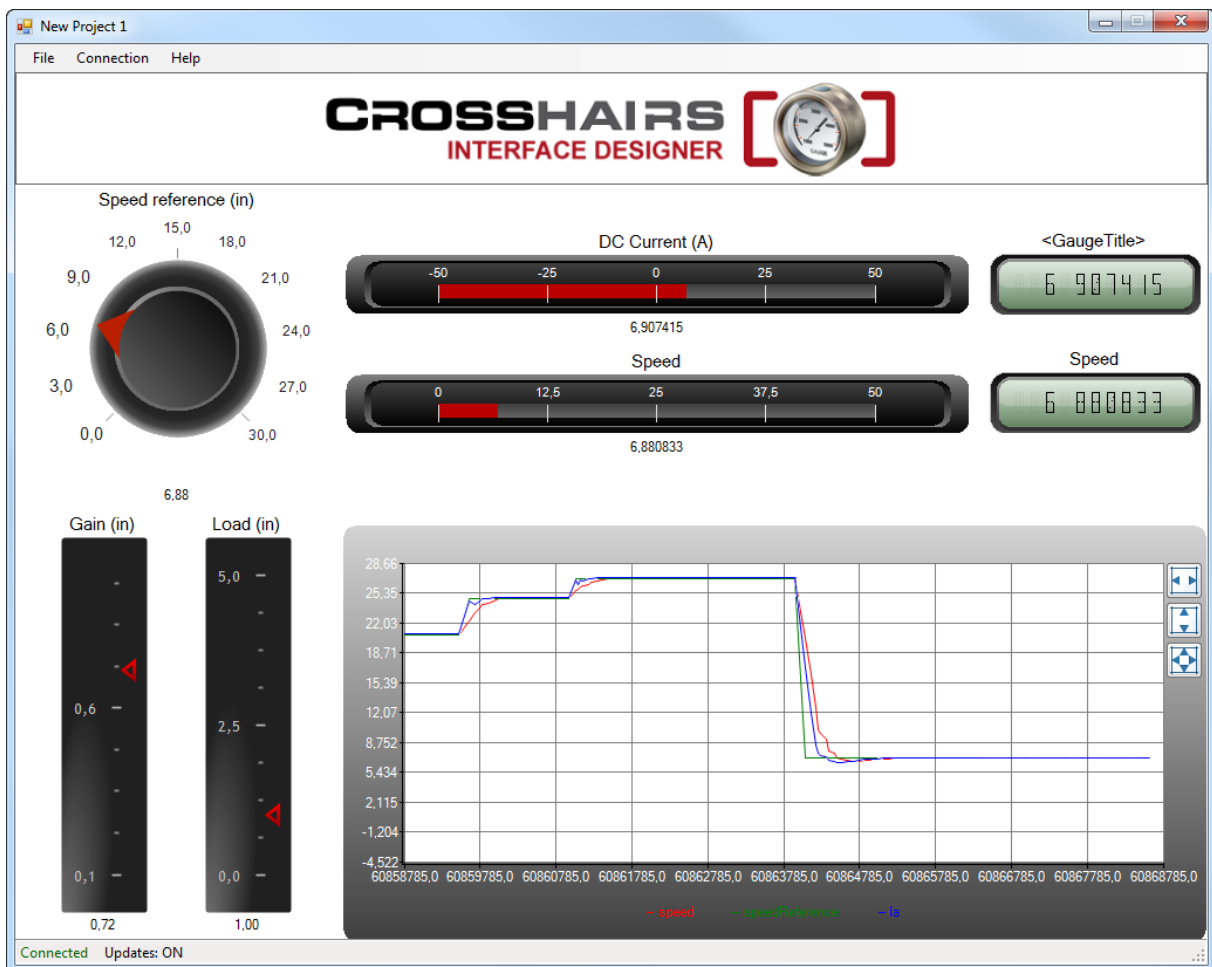


Crosshairs [interface designer]

The Crosshairs [interface designer] is available for trial download from the Crosshairs Embedded website. When using the Crosshairs [interface designer] you have a wealth of high quality graphical components available for your use in designing crisp, user friendly dashboards for your hardware.

For more information, please visit the following link: <http://crosshairseembedded.com/interface-designer>

For trial download, please register at the following link or contact sales@crosshairseembedded.com
<http://crosshairseembedded.com/free-trial>



FAQ

Q: Why can't I connect to the Motor Control Kit through the Crosshairs[debugger] after I've programmed the example on the controlCARD?

A1: (LD3 on the controlCARD is blinking red). Please ensure that the virtual communication port has been enabled for the TI XDS100 device under Control Panel Devices, and that the COM-port(s) for the Motor Control Kit is visible in COM & LPT (For information on how to do this please read the chapter called: [Setting up virtual serial ports](#)). Please be sure to select the right COM-port as well. When COM ports Channel A and Channel B has been set in VCP mode, it is normally the COM port with the highest value that is the correct one to use.

A: (LD3 on the controlCARD isn't blinking) It is possible to start the Motor Control Kit from Code Composer Studio by starting up a debug session and choosing to let the program run free. If the Motor Control Kit is started outside from Code Composer Studio, the JTAG-emulator will prevent the Motor Demo Kit from booting up. Please go to the chapter called [Jumpers and Switches](#) for information about how to disable JTAG.

Q: I downloaded the controlSUITE software from Texas Instruments but the installer failed to run properly. Is there another way to get the controlSUITE resources installed?

A: Yes, you may also download a ZIP archive of the controlSUITE for manual install from this link: <http://www.ti.com/litv/zip/sprca85>

Customer Contact

If you have any questions regarding this document or any of the products and/or services described herein, please do not hesitate to contact us at support@crosshairseembedded.com

All other enquiries including license sales and enterprise solutions please contact us at sales@crosshairseembedded.com.

During business hours we may also be reached at the following phone numbers

- Sales +47 984 15 109
- General information +47 736 05 906

We are committed to total customer satisfaction and a member of our team will be in touch within one working day.

You may also find up-to-date news, company and product information on our website at www.crosshairseembedded.com

